МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

До захисту допускається
Завідувач кафедри
_____Скарга-Бандурова І.С.
«____»_____ 2018 р.

ДИПЛОМНИЙ ПРОЕКТ (РОБОТА) БАКАЛАВРА

ПОЯСНЮВАЛЬНА ЗАПИСКА

НА ТЕМУ:

Інформаційно-довідкова система навчального закладу

Освітньо-кваліфікаційний рівень "бакалавр"
Напрям 6.050102 – "Комп'ютерна інженерія"

Керівник проекту:	_____ (підпис)	доц. Сафонова С.О. (ініціали, прізвище)
Консультант з охорони праці:	_____ (підпис)	ст.викл. Критська Я.О. (ініціали, прізвище)
Здобувач вищої освіти:	_____ (підпис)	Ганієва А.А. (ініціали, прізвище)
Група:		КІ-14бд

Сєвєродонецьк 2018

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет Інформаційних технологій та електроніки
Кафедра Комп'ютерних наук та інженерії
Освітньо-кваліфікаційний рівень бакалавр
Напрям підготовки 6.050102 Комп'ютерна інженерія
(шифр і назва)
Спеціальність
(шифр і назва)

ЗАТВЕРДЖУЮ:
Завідувач кафедри
І.С. Скарга-Бандурова
«_____» _____2018 р.

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) БАКАЛАВРА

Ганієвій Аліні Альбертівні
(прізвище, ім'я, по батькові)

1. Тема роботи Інформаційно-довідкова система навчального закладу

керівник проекту (роботи) Сафонова С.О., к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від " 14 " 05 2018 р. №
2. Термін подання студентом роботи
3. Вихідні дані до роботи матеріали переддипломної практики

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Аналіз і розробка технічного завдання. Вибір і обґрунтування програмних засобів реалізації проекту. Розробка інформаційної системи. Посібник користувача системи. Охорона праці.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Електронні плакати

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	ст.викл. кафедри КНІ Критська Я.О.		

7. Дата видачі завдання _____

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Аналіз і розробка технічного завдання	14.05.18-16.05.18	
2	Вибір і обґрунтування програмних засобів реалізації проекту	17.05.18-20.05.18	
3	Розробка структури бази даних	21.05.18-24.06.18	
4	Розробка основних компонентів системи	25.06.18-30.06.18	
5	Посібник користувача системи	01.06.18-06.06.18	
6	Розробка розділу охорона праці	07.06.18-09.06.18	
7	Оформлення електронних плакатів	10.06.18-12.06.18	
8	Оформлення пояснювальної записки	13.06.18-14.06.18	

Здобувач вищої освіти _____ Ганієва А.А.
(підпис) (прізвище та ініціали)

Керівник _____ Сафонова С.О.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломного проекту (роботи) бакалавра: 130 с., 32 рис., 8 табл., 26 бібліографічних джерел посилань, 2 додатка.

Об'єкт дослідження: процеси обміну інформацією між учасниками учбового процесу середньої школи, а так само батьками.

Мета роботи: розробка програмного забезпечення інформаційно-довідкової системи для середньої школи.

В проекті виконано:

1. Проаналізовано поточний процес обміну інформацією. На основі проведеного аналізу виконана постановка задачі та сформульовані технічні вимоги.
2. Обрані технології для реалізації проекту. Були обрані скриптова мова, СУБД і програма веб-сервер.
3. Розроблена інформаційно-довідкова система.
4. Визначені умови безпечної трудової діяльності.

Отримано наступні результати: була розроблена система, яка забезпечує можливість управління учбовими процесами і надає інформацію про успішність віддалено (через мережу Internet), що дозволило батькам, учням і викладачам мати доступ до інформації по школі з будь-якого комп'ютера, маючого доступ до мережі Internet.

Практичне значення, галузь застосування роботи: результати дипломного проекту можуть використовуватися в роботі будь-якої середньої школи для ефективнішого обміну інформацією.

Ключові слова: СЕРЕДНЯ ШКОЛА, ІНФОРМАЦІЙНА СИСТЕМА, ВЕБ-СЕРВЕР, СИСТЕМА КЕРУВАННЯ БАЗАМИ ДАНИХ, СКРИПТОВА МОВА, HTML СТОРІНКА, PHP.

Умови одержання дипломного проекту: СНУ ім. В. Даля, пр. Центральний 59-А,. м. Сєвєродонецьк, 93400.

ЗМІСТ

ВСТУП

На сьогоднішній день обмін інформацією про успішність учнів освітньої установи відбувається лише за допомогою прямого спілкування або телефонного зв'язку батьків і викладачів. При цьому не завжди батьки здатні прийти до школи для того, щоб повною мірою проконтролювати успішність і відвідуваність своїх дітей. Інформаційні бази даних школи в більшості випадків доступні лише викладацькому складу школи. Швидкість контролю успішності керівництвом школи при цьому істотно понижена. Ухвалення управлінських рішень по «проблемних» учнях відбувається приблизно раз на тиждень, що не дає повної картини по школі.

Використовуючи сучасні засоби програмування і мережні технології, можливо створити систему, яка забезпечуватиме можливість управління учбовими процесами і надаватиме інформацію про успішність віддалено (через мережу Internet). Це дозволить батькам, учням і викладачам мати доступ до інформації по школі з будь-якого комп'ютера, що має доступ до мережі Internet.

В дипломній роботі вирішується задача створення інформаційно-довідкової системи для школи, що дозволить автоматизувати обмін інформацією між всіма учасниками учбового процесу. А так само дозволить істотно зменшити час на обробку даних по успішності і відвідуваності учнів, збільшить швидкість ухвалення рішень керівництвом школи.

1 АНАЛІЗ І РОЗРОБКА ТЕХНІЧНОГО ЗАВДАННЯ

1.1 Призначення інформаційно-довідкового веб-сервера для школи

Основна мета інформаційно-довідкового веб-сервера - це створення єдиної для всієї школи бази даних, що містить інформацію про різні аспекти учбово-виховного процесу: відомості про співробітників, дітях, що вчаться і батьках, учбовому плані, електронні класні журнали, розклад, всілякі звіти і т.п.

Шкільна інформаційно-довідкова система повинна надавати комплекс засобів інтерактивної взаємодії адміністрації школи і вчителів з одного боку, батьків і учнів з іншого боку.

Для батьків - це оперативний контроль успішності і відвідуваності свого дитяти через його електронний щоденник і звіти по успішності, а так само можливість отримувати оперативну інформацію про майбутні заходи і інші події.

Для викладачів - це ведення електронного класного журналу, здобуття звітів про успішність і відвідуваність, доступ до останньої версії свого розкладу, перегляд шкільних і класних заходів.

Для учнів - це перегляд поточного розкладу, шкільних і класних заходів, здобуття підсумкових і поточних звітів про свою успішність і відвідуваність, доступ до свого електронного щоденника, куди автоматично виставляються оцінки і поміщаються домашні завдання.

Для адміністрації школи - це оперативне здобуття і аналіз інформації про учбовий процес для ухвалення управлінських рішень, ведення розкладу, перегляд розкладу з різних точок зору, ведення шкільних і класних заходів.

Так само інформаційно-довідковий веб-сервер повинен надавати єдине середовище обміну інформацією в рамках школи, наприклад, дошки оголошень, внутрішньої електронної пошти, списку іменинників і т.п.

1.2 Аналіз існуючих рішень

Аналіз програмних продуктів, які дозволили б сформувати інформаційне середовище школи, показав, що на даний момент не існує універсального програмного продукту здатного задовольнити абсолютно всі потреби учбового закладу. Дані продукти не надають можливості зміни функціональності унаслідок закритості коду або використання платних програмних рішень.

На даний момент існує два типи програмних продуктів, що надають можливості управляти процесами в межах учбового закладу: локальні і мережні.

Локальні програмні продукти в більшості випадків надають зручний інтерфейс для редагування шкільного розкладу або управління контингентом і кадровим складом школи. При цьому всі дані зберігаються на одному комп'ютері, а, отже, важкий є процес поширення створених даних. У загальному випадку цього типа програмних продуктів надає можливість збереження даних у вигляді Exel таблиць з подальшим поширенням за допомогою змінних носіїв або через мережу. До таких програмних продуктів відносяться “Школьный диспетчер”, ”Экспресс-расписание”, ”ХроноГраф” та інші.

Мережні вирішення програмних продуктів керівників процесами в школі надають доступ до даних в межах мережі (локальної або глобальної). Вони так само містять засоби управління школою. Для забезпечення працездатності даних засобів використовується комп'ютер - сервер зі встановленим на ньому програмним комплексом. Доступ здійснюється через мережу, при цьому користувачі системи отримують дані або за допомогою програм інтерфейсів системи, або при допомозі Інтернет браузера. До таких рішень відноситься система «Net Школа», реалізована на базі Microsoft Office і «1С Школа Online». Система існує з 2004 року і вже введена в багатьох школах Росії і України. До відмітних особливостей даної системи можна віднести можливість доступу до електронних щоденників не лише учнів, але і батьків. Так само в даному рішенні для батьків передбачена можливість отримувати інформацію на мобільний телефон через SMS. Проте дане програмне рішення є платним. Крім того,

працездатність забезпечується лише на операційній системі Windows, а, отже, маються на увазі додаткові витрати на програмне забезпечення. Залежність від операційної системи Windows робить неможливою установку даного програмного комплексу на платні хостінги з метою економії засобів, які буде потрібно на покупку і обслуговування комп'ютера, - сервера системи. Іншим можливим рішенням даної задачі є «1С Школа Online». Система розроблена на базі програмного комплексу «ХроноГраф», отже, має всі необхідні можливості при складанні розкладу і управління учбовим процесом. Проте дана система зроблена переважно для управління школою, документообігом і не надає доступу до інформації ззовні.

Можна додати, що всі вище перелічені системи мають закритий вихідний код, а, отже, не надають можливості зміни під конкретні потреби. Вартість даних програм хоч і невелика, але вимоги до програмного забезпечення вимагають додаткових фінансових вкладень. Саме це є основною причиною доцільності створення інформаційно-довідкового веб-сервера, здатного надавати інформацію за допомогою мережі Internet.

1.3 Вимоги до проектованої системи

Для створення ефективної інформаційно-довідкової системи, необхідно щоб вона відповідала наступним вимогам:

- мала мережне рішення і базувалася на технології клієнт-сервер;
- була комплексним рішенням, тобто забезпечувала ведення єдиної бази шкільних учбових і адміністративних даних;
- надавала повну інформацію про розклад, успішність, відвідуваність учнів;
- надавала можливість зміни або створення нових функціональних модулів;
- забезпечувала безпеку і збереження даних системи;
- мала низьку вартість установки і експлуатації системи;
- мала можливість здобуття статистичних даних для їх аналізу і ухвалення рішень на всіх рівнях управління.

1.3.1 Вимоги до функціональності системи

Інформаційно-довідкова система для середньої школи повинна здійснювати наступні завдання:

- ведення шкільного розкладу;
- зберігання і оперативне представлення інформації про контингент і кадровий склад школи;
- ведення журналу успішності;
- ведення електронних щоденників;
- ведення статистики по предметах;
- формування відомостей о класах;
- формування звітності по успішності;
- ведення персональної інформації про співробітників і учнів;
- формування штатного розкладу;
- надавати батькам дані про успішність і відвідуваність їх дітей;
- надавати інформацію по останніх діях, здійснених в системі;
- здійснення взаємодії між всіма класами користувачів за рахунок внутрішньої пошти, дошок оголошень і ін.;
- розмежування об'єму призначених для користувача прав і можливість виконувати роботу незалежно від інших користувачів.

1.3.2 Вимоги до безпеки даних системи

Інформаційно-довідкова система школи містить конфіденційну інформацію про учнів, тому необхідно максимально забезпечити дані від несанкціонованого доступу. Для цього необхідне дотримання вимог:

- розмежування доступу до даних системи по типах користувачів.
- можливість заборони доступу до системи, як окремого користувача, так і адреси або групи адрес мережі internet.
- максимально можлива ідентифікація користувача.

- заборона одночасної роботи двох і більш користувачів з одного аккаунта.
- тимчасове блокування входу в систему, якщо була вироблена спроба підбору пароля.
- перевірка вхідних параметрів. захист від sql-injection.
- видалення повідомлень про помилки для утаєння внутрішньої структури.

1.3.3 Вимоги до апаратного і програмного забезпечення

Інформаційно-довідкова система не має строгої залежності від програмного забезпечення і операційної системи, на якій планується установка сервера системи. Від операційної системи залежить лише можливий вибір програми реалізовуючої функції веб-сервера, що дозволить збільшити швидкість роботи всієї системи. Окрім цього вибір операційної системи може накладати обмеження на використання деяких програмних продуктів. У загальному випадку рекомендується використання операційних систем сімейства UNIX. Причиною цьому служить покращувана підтримка асинхронних методів передачі даних. Зокрема використання технологій select, epoll, kqueue. Для операційних систем Windows існує аналогічні технології, але за даними статистики лідирує epoll. Дані технології забезпечують одночасне підключення великої кількості користувачів, так для веб-сервера «nginx» кількість підключених одночасно користувачів обмежено лише ресурсами комп'ютера.

Відповідно до вибору програмного забезпечення сервера та необхідності одночасної роботи декількох сотень користувачів одночасно, були встановлені мінімальні вимоги, що рекомендуються, до апаратного забезпечення представлені в таблиці 1.1 та 1.2.

Таблиця 1.1 – Мінімальні вимоги до апаратного забезпечення

Процесор	Intel Celeron 1.40 ГГц
ОЗУ	512 Мб
Мережна карта	Ethernet карта 10/100Mbps

У таблиці 1.1 представлені вимоги до апаратного забезпечення, які дозволять запустити операційну систему (ОС) з невеликим запасом ресурсів для забезпечення роботи інформаційно-довідкової системи. При цьому кількість користувачів системи буде мінімальною, і дана конфігурація устаткування може бути використана в цілях розробки додаткових компонентів або функціонуванні з малою кількістю користувачів.

Таблиця 1.2 - Вимоги, що рекомендуються, до апаратного забезпечення

Процесор	Intel Xeon 3.4 ГГц
ОЗУ	4096 Мб
Мережна карта	Intel PRO/100+ Dual Port Server Adapter

У таблиці 1.2 представлені вимоги до апаратного забезпечення. Вони дозволять запустити систему в штатному режимі і забезпечать кількість одночасних з'єднань вище 500, а так само нададуть виконання завдань, передбачених системою, без затримок.

В цілому, вибір апаратного забезпечення залежить від операційної системи. Так для ОС Linux (наприклад, Gentoo) скомпільованою з мінімальною кількістю компонентів система працюватиме і на процесорі з тактовою частотою нижче 500 ГГц і 64мб оперативної пам'яті.

Якщо ж вже є комп'ютер, то програмне забезпечення вибирається для забезпечення максимальної продуктивності системи. Так для комп'ютерів з тактовою частотою нижче 2 ГГц і ОЗУ менше ніж 1024Мб бажане використання ОС сімейства UNIX, які збираються з модулів і можуть використовувати мінімум ресурсів комп'ютера. Для комп'ютерів з ОЗУ більше 2 Гб і тактовою частотою вище 2ГГц можливе використання ОС сімейства Windows, при цьому ресурси комп'ютера буде вистачати як для внутрішніх потреб операційної системи, так і для програмного забезпечення того, що використовується інформаційною системою.

1.3.4 Вимоги до методичного забезпечення

Інформаційно-довідкова система надає велику кількість функцій. Для забезпечення зручного використання системи необхідно створити:

- посібник користувача системи;
- документацію по установці інформаційної системи;
- описи внутрішніх компонентів системи;
- систему інтерактивної допомоги користувачеві системи.

Керівництво по установці і використанню дозволить полегшити процес установки і використання системи. При цьому керівництво повинне надавати повний перелік можливих дій в системі. Інтерактивна допомога вбудовується в саму інформаційно довідкову систему і дозволяє отримувати підказки прямо під час роботи з системою. Опис внутрішніх компонентів системи дозволить створювати нові компоненти, що вже існують, що дозволить додавати функціональність системі залежно від конкретних потреб учбової установи.

2 ВИБІР І ОБҐРУНТУВАННЯ ПРОГРАМНИХ ЗАСОБІВ РЕАЛІЗАЦІЇ ПРОЕКТУ

2.1 Структура інформаційно-довідкового веб-сервера

Інформаційно-довідковий веб-сервер вимагає наявність наступних компонентів:

1) системи управління базами даних (СУБД) для зберігання і управління даними системи;

2) веб-серверу для прийому та обробки запитів від користувачів, а так само виведення інформації;

3) скриптової мови програмування для виконання логічних операцій, взаємодії з базою даних, генерування HTML сторінок.

Крім того, можливо використовувати допоміжні технології на стороні клієнта для зменшення навантаження на сервер і спрощення управління даними інформаційної системи.

2.2 Аналіз і вибір СУБД для реалізації проекту

Сьогодні бази даних (БД) - основа будь-якої крупної інформаційної системи, що зберігає і оброблює всілякі дані. Це системи електронного документообігу і бухгалтерського обліку, білінгові системи і системи управління вмістом веб-сайтів, системи управління технологічними процесами на виробництві, інформаційні каталоги. Оскільки в базах даних зберігається критична для ведення бізнесу інформація, то першочергове завдання адміністраторів баз даних - підтримка конфіденційності, цілісності і доступності цих даних. Окрім того, що в разі успішної атаки потенційний порушник може дістати доступ до інформації, що зберігається в базі даних, існує можливість виконання довільних команд на сервері з привілеями процесу СУБД, що у свою

чергу, може привести до здобуття порушником повного доступу безпосередньо до сервера, на якому встановлена СУБД.

Для зберігання даних про користувачів системи, розклад, щоденників і ін. необхідна наявність СУБД, яка дозволить зберігати і управляти даними, а також взаємодіяти з останніми компонентами системи. Вимоги до СУБД для інформаційно - довідкової системи наступні:

1. Спосіб доступу до БД – клієнт-сервер.

2. Модель даних – реляційна.

3. Найменша залежність від операційної системи і програмного забезпечення.

Нижче будуть проаналізовані деякі з можливих СУБД для реалізації проекту інформаційно-довідкового веб-сервера.

2.2.1 СУБД Microsoft SQL

СУБД MS SQL розроблена корпорацією Microsoft. Основна використовувана мова запитів — TRANSACT-SQL, створена спільно Microsoft і Sybase. TRANSACT-SQL є реалізацією стандарту Ansi/iso по структурованій мові запитів (SQL) з розширеннями. Використовується для невеликих і середніх за розміром баз даних.

Microsoft SQL Server як мова запитів використовує версію SQL, що отримала назву TRANSACT-SQL (скорочено T-SQL), є реалізацією Sql-92 (стандарт ISO для SQL) з множинними розширеннями. T-SQL дозволяє використовувати додатковий синтаксис для збережених процедур, і забезпечує підтримку транзакцій (взаємодія бази даних із застосуванням, що управляє). Microsoft SQL Server і Sybase ASE для взаємодії з мережею використовують протокол рівня додатка під назвою Tabular Data Stream (TDS, протокол передачі табличних даних). Протокол TDS також був реалізований в проекті FREETDS з метою забезпечити різним застосуванням можливість взаємодії з базами даних Microsoft SQL Server і Sybase.

SQL Server підтримує дзеркалірование і кластеризацію баз даних. Використання кластеризації дозволяє розподілити робоче навантаження між декількома серверами. Всі сервера мають одне віртуальне ім'я, і дані розподіляються по IP адресам машин кластера протягом робочого циклу. Також в разі відмови або збою на одному з серверів кластера доступне автоматичне перенесення навантаження на інший сервер.

SQL Server підтримує надлишкове дублювання даних по трьох сценаріях:

1. *Знімок:* виробляється «знімок» бази даних, який сервер відправляє одержувачам.

2. *Історія змін:* всі зміни бази даних безперервно передаються користувачам.

3. *Синхронізація з іншими серверами:* бази даних декількох серверів синхронізуються між собою. Зміни всіх баз даних відбуваються незалежно один від одного на кожному сервері, а при синхронізації відбувається звірка даних. Даного типа дублювання передбачає можливість вирішення протиріч між БД.

У SQL Server вбудована підтримка .NET Framework. Завдяки цьому, процедури БД, що зберігаються, можуть бути написані на будь-якій мові платформи .NET, використовуючи повний набір бібліотек, доступних для .NET Framework, включаючи Common Type System (система поводження з типами даних в Microsoft .NET Framework). Проте, на відміну від інших процесів .NET Framework, будучи базисною системою для SQL Server, виділяє додаткову пам'ять і вибудовує засоби управління SQL Server замість того, щоб використовувати вбудовані засоби Windows. Це підвищує продуктивність порівняно із загальними алгоритмами Windows, оскільки алгоритми розподілу ресурсів спеціально налагоджені для використання в структурах SQL Server.

MS SQL дозволяє створювати *stored procedures* - це набір скомпільованих команд T-SQL, доступних безпосередньо sql-серверу. Команди розміщуються в процедурі, що зберігається, і виконуються як одне ціле або підпрограма по аналогії з іншими мовами програмування. Процедури, що зберігаються, знаходяться на сервері СУБД і використовуються, коли необхідно часто

виконувати запити, що повторюються в певному порядку до сервера MS SQL. Переваги таких процедур у малій кількості даних переданих від клієнта серверу БД і можливість розподілити навантаження сервера. Недоліком даної СУБД є можливість роботи лише під ОС сімейства Windows. Окрім цього дана СУБД вимоглива до ресурсів комп'ютера і складна в налаштуванні і забезпеченні безпеки. Проте вона забезпечує хорошу продуктивність і можливість перенесення частини логіки додатка в базу даних. Останнє забезпечується за рахунок *stored procedures.*

2.2.2 СУБД MySQL

MySQL вільна система управління базами даних. MySQL є власністю компанії Sun Microsystems, що здійснює розробку і підтримку додатка. Поширюється під GNU General Public License і під власною комерційною ліцензією, на вибір. Окрім цього компанія MySQL AB розробляє функціональність за замовленням ліцензійних користувачів, саме завдяки такому замовленню майже в найраніших версіях з'явився механізм реплікації.

MYSQL є рішенням для малих і середніх застосувань. Зазвичай MySQL використовується як сервер, до якого звертаються локальні або видалені клієнти, проте в дистрибутив входить бібліотека внутрішнього сервера, що дозволяє включати MySQL в автономні програми.

Гнучкість СУБД MySQL забезпечується підтримкою великої кількості типів таблиць: користувачі можуть вибрати як таблиці типа MyISAM, що підтримують повнотекстовий пошук, так і таблиці InnoDB, що підтримують транзакції на рівні окремих записів. Більш того, СУБД MySQL поставляється із спеціальним типом таблиць EXAMPLE, що демонструє принципи створення нових типів таблиць. Завдяки відкритій архітектурі і gpl-ліцензуванню, в СУБД MySQL постійно з'являються нові типи таблиць.

MySQL має подвійне ліцензування. MySQL може поширюватися відповідно до умов ліцензії GPL. Проте за умовами GPL, якщо яка-небудь програма включає вихідні коди MySQL, то вона теж повинна поширюватися за ліцензією GPL.

MySQL портирована на велику кількість платформ: AIX, Bsdi, FREEBSD, HP-UX, Gnu/linux, Mac OS X, NETBSD, OPENBSD, Os/2 Warp, SGI IRIX, Solaris, SUNOS, SCO Openserver, SCO Unixware, Tru64, Windows 95, Windows 98, Windows NT, Windows 2000, Windows XP, Windows Server 2003 і Windows Vista. Існує також порт MySQL до OpenVMS. Поважно відзначити, що компанія MySQL AB надає для вільного завантаження не лише вихідні коди СУБД, але і готові виконувані модулі, що відкомпілювалися і оптимізовані під конкретні операційні системи, які можна завантажити з офіційного сайту.

MySQL має API для мов Delphi, C, C++, Эйфель, Java, Лісп, Perl, PHP, Python, Ruby, Smalltalk і Tcl, бібліотеки для мов платформи .NET, а також забезпечує підтримку для ODBC за допомогою odbc-драйвера MyODBC.

MySQL підтримує мову запитів SQL в стандарті ANSI 92, і окрім цього має безліч розширень до цього стандарту, яких немає ні в одній іншій СУБД.

Короткий перелік можливостей MySQL:

- підтримується необмежена кількість користувачів, що одночасно працюють з базою даних;
- кількість рядків в таблицях може досягати 50 млн;
- швидке виконання команд;
- проста і ефективна система безпеки.

Внутрішні характеристики і переносимість:

- написаний на c і c++. протестований на безлічі різних компіляторів;
- працює на різних платформах;
- для забезпечення переносимості використовується gnu automake, autoconf і libtool;
- api для c, c++, eiffel, java, perl, php, python, ruby и tcl;

- повністю багатопотоковий, з використанням потоків ядра. Це означає, що, якщо така можливість забезпечується, можна легко організувати роботу з декількома процесорами;

- дуже швидкі дискові таблиці на основі b-дерев із стискуванням індексів;

- дуже швидка система розподілу пам'яті, що базується на потоках;

- дуже швидкі з'єднання, що використовують оптимізований метод однопрохідного мультіз'єднання(one-sweep multi-join);

- хеш-таблиці в пам'яті, використовувані як тимчасові таблиці;

- sql-функції реалізовані за допомогою добре оптимізованої бібліотеки класів, тому вони виконуються настільки швидко, наскільки це можливо;

- mysql- код протестований з використанням purify (комерційний детектор витоку пам'яті), а також valgrind, одного з gpl-інструментів.

Так само слід особливо відзначити, що розміри таблиць визначаються граничним розміром файлу у файловій системі операційної системи, під якою працює MySQL. Проте слід зауважити, що операційні системи мають свої власні обмеження по розмірах файлів.

Таким чином, основні важливі сторони пакету MySQL - це багатопоточність, підтримка декількох одночасних запитів, оптимізація зв'язків з приєднанням багатьох даних за один прохід, записи фіксованої і змінної довжини, ODBC драйвер в комплекті з ісходником, гнучка система привілеїв і паролів, до 16 ключів в таблиці, кожен ключ може мати до 15 полів. Також є підтримка ключових полів і спеціальних полів в операторові CREATE, підтримка чисел довгою від 1 до 4, рядків змінної довжини і міток часу, інтерфейс з мовами C і perl. Заснована на потоках, швидка система пам'яті, утиліта перевірки і ремонту таблиці, всі дані зберігаються у форматі Iso8859_1. Всі операції роботи з рядками не звертають уваги на регістр символів в оброблюваних рядках, псевдоніми застосовні як до таблиць, так і до окремих колонок в таблиці, всі поля мають значення за умовчанням. INSERT можна використовувати на будь-якій підмножині полів. Легкість управління таблицею, включаючи додавання і видалення ключів і полів.

Проаналізувавши вищеперелічені СУБД, для реалізації проекту була вибрана MySQL. Основні причини вибору це:

- підтримка безлічі мов програмування;
- простота налаштування і управлінням безпекою даних;
- швидкість роботи;
- можливість безкоштовного використання за ліцензією gpl.

Крім того, дана СУБД надає досить функціональності для розробки веб-сервера-додатків і є однією з провідних СУБД використовуваних для використання у веб-сервері-додатках. Враховуючи особистий досвід роботи з даною СУБД, вона показала відмінну продуктивність і безпеку даних при невисоких показниках апаратного забезпечення. Все вищеперелічене дає можливість зробити вибір саме СУБД MySQL.

2.3 Вибір програми веб-сервера для реалізації проекту

Програма Http-сервер (він же веб-сервер) являється невід'ємною частиною розроблювальної інформаційно-довідкової системи. Саме вона надає можливість клієнтам системи робити запити й отримувати необхідну інформацію у вигляді HTML сторінок. Це дає можливість користувачам системи без установлення додаткового програмного забезпечення користуватися ресурсами інформаційно-довідкової системи. Основні вимоги до Http-серверу наступні:

- багатопоточність - можливість одночасно обробляти запити декількох користувачів;
- по можливості незалежність від ос і пз сервера;
- підтримка скриптових мов, тому що саме вони будуть забезпечувати генерування html сторінок на основі інформації з баз даних, і запиту отриманого http-сервером від клієнта.

Далі буде розглянуте програмне забезпечення, що надає необхідні можливості.

2.3.1 Веб-сервер Apache

Основними гідностями Apache вважаються надійність і гнучкість конфігурації. Він дозволяє підключати зовнішні модулі для надання даних, використовувати СУБД для аутентифікації користувачів, модифікувати повідомлення про помилки і т.д. Недоліком найбільше часто називається відсутність зручного стандартного інтерфейсу для адміністратора.

Веб-сервер Apache розробляється й підтримується відкритим співтовариством розроблювачів під егідою Apache Software Foundation і включений у багато програмних продуктів, серед яких СУБД Oracle і IBM Websphere.

На UNIX системах, які підтримують потоки стандарту POSIX, Apache починаючи з версії 2.0 може виконуватися в гібридному багатопроцесово-багатопоточному режимі. Це сприяє розширюваності системи для багатьох, але не для всіх конфігурацій.

Модулі Apache можна написати так, що вони будуть виконувати роль фільтрів, що обробляють потоки даних, які приходять або йдуть із сервера. Це дозволяє бути обробленими SSI фільтром INCLUDES, надаваним модулем mod_include. Модуль mod_ext_filter дозволяє зовнішнім програмам виконувати роль фільтрів точно в такий же спосіб, яким CGI програми діють як оброблювачі.

Повідомлення про помилки, що посилають браузеру, тепер представлено кількома мовами й використовують SSI технологію. Вони можуть бути легко відредаговані адміністратором під свої потреби.

Apache 2.0 на Windows NT тепер використовує кодування utf-8 для роботи з іменами файлів. Це дозволяє використовувати файлову систему, що працює у форматі Unicode, що надає підтримку сервером багатомовності для всіх NT-систем.

До складу Apache 2.0 була включена бібліотека для роботи з Perl-сумісними регулярними вираженями (PCRE). Всі регулярні вирази тепер використовують потужніший синтаксис Perl 5.

Компіляція і установка Apache в Unix-системах стала простішою і аналогічною процесу збірки інших програмних продуктів. Це досягається тим що нині вона грунтується на методиках autoconf і libtool.

Модулі Apache можуть писатися як фільтри, що переглядають всі дані, що приходять або вирушають з сервера. Завдяки цій можливості можна обробляти SSII-фільтром Includes сторінки, які до цього були оброблені Phpi або CGI-скріптом. Раніше таке поєднання було неможливим.

Сервер Apache є ідеальним вирішенням для платформи MS Windows, проте на UNIX системах він поступається в продуктивності деяким Http-серверам.

2.3.2 Веб-сервер nginx

nginx (engine x) — вільний веб-сервер і поштовий проксі-сервер, що працює на Unix-подобних операційних системах.

Основним компонентом даного продукту є Http-сервер, що надає наступні можливості:

- обслуговування статичних запитів, індексних файлів, автоматичне створення списку файлів, кеш дескрипторів відкритих файлів;
- можливість розподілу навантаження і відмовостійкість;
- підтримка FASTCGI і memcached серверів;
- фільтри, у тому числі стискування (gzip), byte-ranges (докачка), chunked відповіді, Http-аутентіфікация, Ssi-фільтр;
- декілька підзапитів на одній сторінці, оброблювані в Ssi-фільтрі через прокси або FASTCGI, виконуються паралельно;
- підтримка SSL;
- експериментальна підтримка вбудованого Perl.

Конфігурація Http-сервера nginx розділяється на віртуальні сервери (директива server). Віртуальні сервери розділяються на location'и (location). Для віртуального сервера можливо задати адреси і порти, на яких прийматимуться з'єднання, а так само імена які можу включати * для позначення довільної

послідовності в першій і останній частині, або задаватися регулярним вираженням.

Для ефективного управління пам'яттю nginx використовує пули. Пул — це послідовність заздалегідь виділених блоків динамічної пам'яті. Довжина блоку варіюється від 1 до 16 кілобайт. Спочатку під пул виробиться лише один блок. Блок розділяється на зайняту область і незаняту. Виділення дрібних об'єктів виконується шляхом просування покажчика на незайняту область з урахуванням вирівнювання. Якщо розмір об'єкту, що виділяється, перевищує значення константи Ngx_max_alloc_from_pool або довжину блоку, то він повністю виділяється з купи.

Таким чином, дрібні об'єкти виділяються дуже швидко і мають накладні витрати лише на вирівнювання.

nginx містить модуль географічної класифікації клієнтів по Ip-адресу. У його основу входить база даних відповідності Ip-адресів географічному регіону, представлену у вигляді Radix tree в оперативній пам'яті. nginx заздалегідь розподіляє перші декілька рівнів дерева, таким чином щоб вони займали рівно 1 сторінку пам'яті. Це гарантує, що при пошуку Ip-адреса для перших декількох вузлів при трансляція адреси завжди знайдеться запис в TLB.

Обидва, описаних вище, веб-сервера можуть використовуватися для створення інформаційно-довідкової системи для школи. Вибір конкретного залежатиме від операційної системи комп'ютера сервера. Для UNIX систем раціонально використовувати nginx оскільки він забезпечує велику продуктивність (за рахунок використання асинхронних методів передачі даних) і при цьому використовує менше ресурсів. Для систем Windows можливо використовувати веб-сервер Apache. Для ОС сімейства Windows можливо так само використання програмних комплексів при установці яких ставиться одночасно веб-сервер і база даних.

2.4 Аналіз і вибір мов програмування

Для забезпечення динамічного створення HTML сторінок необхідне використання скриптової мови, що виконується перед виведенням сторінок веб-сервером клієнтові. Вимоги до такої мови наступні: підтримка вибраним веб-сервером; можливість роботи з субд; достатня функціональність.

Далі розглядаються мови програмування, які можливо використовувати для реалізації проекту інформаційно-довідкового веб-сервера.

2.4.1 Препроцесор гіпертексту PHP

«PHP: препроцессор гипертекста» — мова програмування, створена для генерування HTML-сторінок на веб-сервері та роботах з базами даних. В даний час підтримується переважною більшістю хостінг-провайдеров. Входить в LAMP — є одним з компонентів найбільш розповсюдженого набору для створення веб-сайтів (Linux, Apache, MYSQL, PHP (Python або Perl)).

Група розробників PHP складається з безлічі людей, що добровільно працюють над ядром і розширеннями PHP і суміжними проектами. В області програмування для мережі, PHP — одна з популярних скриптових мов (поряд з JSP, Perl і мовами, використовуваними в ASP.NET) завдяки своїй простоті, швидкості виконання, багатій функціональності і поширенню початкових кодів на основі ліцензії PHP. PHP відрізняється наявністю ядра і модулів, що підключаються «расширений»: для роботи з базами даних, сокетами, динамічною графікою, криптографічними бібліотеками, документами формату PDF. Існують сотні розширень, проте в стандартне постачання входить лише декілька десятків тих, що добре зарекомендували себе.

В даний час PHP використовується сотнями тисяч розробників. Порядка 20 мільйонів сайтів повідомляють про роботу з PHP, що складає більше п'ятої долі доменів Інтернету.

Синтаксис PHP подібний до синтаксису мови C. Деякі елементи, такі як асоціативні масиви і цикл foreach, запозичені з Perl. Для роботи програми не потрібно описувати будь-які змінні, використовувані модулі, і тому подібне. Будь-яка програма може починатися безпосередньо з оператора PHP. PHP виконує код, що знаходиться усередині обмежувачів, таких як <?php ?>. Все, що знаходиться поза обмежувачами, виводиться без змін. В основному це використовується для вставки PHP-кода в HTML-документ. Окрім обмежувачів <?php ?>, допускається використання додаткових варіантів, таких як <? ?> і <script language="php"> </script>.

Імена змінних починаються з символу $, типа змінної оголошувати не потрібно. На відміну від імен функцій і класів, імена змінних чутливі до регістра. Змінні обробляються в рядках, в подвійних лапках, і heredoc-строках (рядках, створених за допомогою оператора <<<).

PHP розглядує перехід на новий рядок як пропуск, так само як HTML і інші мови з вільним форматом. Інструкції розділяються за допомогою крапки з комою (;), за винятком деяких випадків.

PHP підтримує три типи коментарів: у стилі мови Сі (обмежені /* */) C++ (що починаються з // і рядками, що йдуть до кінця) і оболонки UNIX (з # до кінця рядка).

PHP є мовою програмування з динамічною типізацією, що не вимагає вказівки типа при оголошенні змінних, так само як і самого оголошення змінних. Перетворення між скалярними типами здійснюються неявно без додаткових зусиль (втім PHP надає широкі можливості і для явного перетворення типів).

До скалярних типів даних відносяться: цілий тип (integer); речовий тип даних (float, double); логічний тип (boolean); рядковий тип (string); спеціальний тип NULL. До не скалярних типів відносяться: «ресурс» (*resource*); масив (*array*); объект (object).

Заслання на зовнішні ресурси мають типа «ресурс» (resource). Змінні даного типа, як правило, є дескриптором, що дозволяє управляти зовнішніми об'єктами, такими як файли, динамічні зображення, результуючі таблиці бази даних і т. п.

Масиви (array) підтримують числові і рядкові ключі і є гетерогенними. Масиви можуть містити значення будь-яких типів, включаючи інші масиви. Велика частина суперглобальних масивів містить вхідні дані запиту користувача (параметри Get-запроса, поля форм при посилці методом POST і т. п.).

Нижче приведений список суперглобальних масивів з описом даних в них, що зберігаються:

1. $_GLOBALS - масив всіх глобальних змінних (у тому числі і призначених для користувача).

2. $_SERVER (застарілий аналог — $HTTP_SERVER_VARS) - містить змінні оточення, які операційна система передає серверу.

3. $_ENV (уст. $HTTP_ENV_VARS) - поточна змінна середа (*Environment variables*). Їх набір специфічний платформі, на якій виконується скрипт.

4. $_GET (уст. $HTTP_GET_VARS) - містить параметри Get-запроса, передані в URI після знаку питання «?».

5. $_POST (уст. $HTTP_POST_VARS) - асоціативний масив значень полів Html-форми при відправки методом POST. Індекси елементів відповідають значенню атрибуту name елементів управління Html-форм.

6. $_FILES (уст. $HTTP_POST_FILES) - асоціативний масив з відомостями про відправлених методом POST файлах. Кожен елемент має індекс ідентичний значенню атрибуту «name» у формі і, у свою чергу, також є масивом.

7. $_COOKIE (уст. $HTTP_COOKIE_VARS) - асоціативний масив з переданим агентом користувача значеннями cookies.

8. $_REQUEST - містить елементи з масивів $_GET, $_POST, $_COOKIE. З версії PHP 4.1 включає $_FILES.

9. $_SESSION (уст. $HTTP_SESSION_VARS) - містить дані сесії.

PHP підтримує широкі об'єктно-орієнтовані можливості, повна підтримка яких була введена в п'ятій версії мови. Клас в PHP оголошується за допомогою ключового слова class. Методи і поля класу можуть бути загальнодоступними (public, за умовчанням), захищеними (protected) і прихованими (private). PHP підтримує всі три основні механізми ООП — інкапсуляцію поліморфізм і

спадкоємство (батьківський клас вказується за допомогою ключового слова extends після імені класу). Вирішується оголошення фінальних, абстрактних методів і класів. Множинне спадкоємство класів не підтримується, проте клас може реалізовувати декілька інтерфейсів. Для звернення до методів батьківського класу використовується ключове слово parent.

PHP надає розробникам велику кількість найрізноманітніших функцій, які потрапили в мову з розширень, що створюються різними групами програмістів. В результаті синтаксис мови не погоджений.

Код, створений для раніших версій мови, частенько не працює або працює некоректно з пізнішими версіями мови. У пізніших версіях виключаються конструкції, методики, функції, що застосовувалися раніше. В результаті, застосування, створені кілька років тому практично втрачають працездатність для сучасних версій мови і вимагають значної модифікації. Слід зазначити, що відсутність зворотної сумісності взагалі характерно для сучасних мов, що інтерпретуються.

Підтримка Unicode-строк реалізується через розширення mbstring. При цьому замість стандартних функцій роботи з рядками використовуються аналогічні функції, але з префіксом mb_. Самі рядки не зберігають інформацію про своє кодування, і її необхідно вказувати вручну при виклику функцій розширення mbstring.

У мові не передбачена можливість створення многопоточних застосувань. Є різні обхідні вирішення, проте PHP поширений головним чином в області Web-розробки, де часто проблему многопоточності бере на себе веб-сервер.

2.4.2 Мова програмування Perl

Perl — високорівнева динамічна мова програмування спільного призначення, що інтерпретується, створена Ларрі Уоллом, лінгвістом за освітою. Назва мови є абревіатурою, яка розшифровується як Practical Extraction and Report Language «практична мова для витягання даних і складання звітів».

Основною особливістю мови вважаються його багаті можливості для роботи з текстом, у тому числі реалізовані за допомогою регулярних виразів. Perl також знаменитий величезною колекцією додаткових модулів, що розширюють функціональність. Перл успадкував багато властивостей від мов Сі, shell script, awk. Синтаксис Perl має багато спільного з синтаксисом мов Си, Awk, sed и shell.

Основні типи даних: скаляр, масив, хеш-таблиця, функція, файловий дескриптор. Змінні різних типів відрізняються знаком, який стоїть перед ім'ям змінної.

1. Скалярні змінні використовуються для зберігання одиночних значень. Вони можуть містити числа, рядки і посилання на інші об'єкти. Перед ім'ям скалярної змінної необхідно ставити знак долара '$'. Тип скалярної змінної не фіксований (на відміну від, наприклад, мови С) і визначається динамічно залежно від контексту.

2. Масив є впорядкованим списком скалярів. Кожен елемент масиву має порядковий індекс, за допомогою якого до нього можна дістати доступ. Нумерація елементів починається з нуля. Перед ім'ям змінної -массиву необхідно ставити знак '@', а для доступу до певного елементу масиву рекомендується ставити знак '$', оскільки певний елемент масиву є скаляром. Багатовимірні масиви можна змоделювати, поміщаючи в список посилання на інші списки.

3. Хеш-таблиця є асоціативним масивом, що дозволяє асоціювати рядок зі скаляром. Рядок називається ключем, а скаляр - значенням. Перед ім'ям змінної - списку необхідно ставити знак відсотка '%', а для доступу до певного елементу масиву рекомендується ставити знак '$'. Фізично Хеш-таблиця є масивом, де в непарних позиціях знаходяться ключі, а в парних — значення.

4. Функція є фрагментом виконуваного коду. Функція завжди повертає яке-небудь значення або UNDEF. Якщо значення, що повертається, явно не вказане оператором return, повертається останнє обчислене значення. Константа є незміним значенням. Константа не є вбудованим типом мови і емулюється за допомогою функцій.

5. Файловим дескриптором є покажчик на файл, пристрій або PIPE канал, відкриті для запису, читання або для запису і читання.

Perl - мова, що інтерпретується, пристосована для обробки довільних текстових файлів, витягання з них необхідної інформациії і видачі повідомлень. Perl також зручний для написання різних системних програм. Ця мова проста у використанні, ефективна, але про неї важко сказати що вона елегантна і компактна. Perl вміщує в собі кращі риси C, shell, sed і awk. Сінтаксис виразів Perl-а близький до синтаксису C. На відміну від більшості утиліт ОС UNIX Perl не ставить обмежень на об'єм оброблюваних даних, і якщо вистачає ресурсів, то весь файл обробляється як один рядок. Рекурсія може бути довільної глибини. Хоча Perl пристосований для сканування текстових файлів, він може обробляти так само двійкові дані і створювати .dbm файли, подібні до асоціативних масивів. Perl дозволяє використовувати регулярні вирази, створювати об'єкти, вставляти в програму на C або C++ шматки коду на Perl-е, а також дозволяє здійснювати доступ до баз даних, у тому числі Oracle. Проте підтримка різних СУБД досягається за рахунок використання розширень.

Як скриптова мова для програмної реалізації інформаційно-довідкового веб-сервера був вибраний PHP. Вибір обумовлений простотою синтаксису, великою кількістю функцій для роботи з текстом, базами даних. PHP дає можливість генерувати не лише HTML сторінки, але і текстові документи графічні файли і ін. Так само простота використання і наявність початкового коду дозволить простіше змінювати, відладжувати або додавати нові компоненти в інформаційну систему.

2.5 Допоміжні технології

Для того, щоб спростити створення сторінок, що передаються клієнтові, а також відокремити HTML сторінки від виконання логічних операцій, в проекті, можливе використання Smarty-компілюючого обробника шаблонів для PHP. Він

дозволить винести теги HTML в окремий файл. Крім того це дозволить додати функціональності в генерування HTML сторінок.

Так само для зменшення навантаження на сервер і впорядкування даних на HTML сторінці, на стороні клієнта потрібне використання технології Javascript і AJAX. Це дозволить зменшити кількість файлів на сервері, а так само зменшити навантаження на базу даних.

2.5.1 Smarty – обробник шаблонів для PHP

Одне з призначень Smarty — це відділення логіки застосування від вистави. Звичайно ж, шаблони можуть містити в собі логіку, але лише за умови, що ця логіка необхідна для правильного представлення даних. Такі завдання, як підключення інших шаблонів, забарвлення рядків, що чергується, в таблиці приведення букв до верхнього регістра, циклічний прохід по масиву для його відображення і так далі — все це є прикладом логіки вистави. Не слід думати, що Smarty заставляє вас розділяти логіку застосування і виставу. Smarty не бачить різниці між цими речами отже поміщати або не поміщати логіку застосування в шаблони вирішує програміст.

Одна з унікальних можливостей Smarty — компіляція шаблонів. Це означає, що Smarty читає файли шаблонів і створює Php-код на їх основі. Код створюється один раз і потім лише виконується. Тому немає необхідності обробляти файл шаблону для кожного запиту і кожен шаблон може користуватися всіма перевагами таких кешируючих вирішень, як eaccelerator або PHP Accelerator.

Деякі особливості Smarty:

- швидкість роботи;
- ефективність, оскільки обробник php робить за нього основну роботу;
- жодної зайвої обробки шаблонів, вони компілюються лише один раз;
- перекомпілюються лише ті шаблони, які змінилися;
- можна створювати призначені для користувача функції і модифікатори, що робить мову шаблонів надзвичайно розширюваною;

- роздільники тегов шаблону, що набудовуються;
- конструкції *if/elseif/else/endif* передаються обробникові php, так що синтаксис вираження *{if ...}* може бути настільки простим або складним, наскільки це необхідно;
- допустиме неограниченноє вкладення секцій, умов і т. д.;
- існує можливість включення php-кода прямо в шаблон, проте зазвичай в цьому немає необхідності (і це не рекомендується), оскільки движок вельми гнучкий і розширюваний;
- вбудований механізм кешування;
- довільні джерела шаблонів;
- призначені для користувача функції кешування;
- компонентна архітектура.

Для того, щоб працювати з шаблонами Smarty, потрібно зробити ці бібліотеки доступними для всіх скриптів, в яких вони використовуються.

Smarty - не просто клас для обробки шаблонів, він визначає цілу мову побудови шаблонів. При цьому шаблон Smarty це набір спеціальних конструкцій (змінних викликів функцій і методів і т.п) і HTML-тегов. За замовчуванням це символи фігурних дужок “{” і “}”, але їх можна змінити. Все, що не поміщене в такі обмежувачі, Smarty розглядує як константи не вимагаючи обробки. Коментарі в Smarty записуються між двома зірочками: *{* Це коментар. *}*

Кожен Smarty тег або виводить значення змінної, або викликає яку-небудь функцію. Функція записується таким чином:

{імя_функциі атрібут1=”значеніє1" атрібут2=”значеніє2"}

Змінні в шаблоні можуть бути декількох типів:

1. Змінні, значення яким привласнюється в php-скрипті користувача, повинні мати перед ім'ям знак долара.

Наприклад*: {$first_name}*

2. Елементи масиву, значення яких були привласнені в php-скрипті користувача, доступні в шаблоні за допомогою синтаксису {$імя_массива.ассоциатівний_ключ}.

Наприклад*: {$person.last_name}*

3. Елементи не асоціативного масиву доступні за допомогою синтаксису квадратних дужок: {імя_массива[числової_індекс]}.

Наприклад*: {$person[2]}*

4. Властивості об'єктів, задані в php-скрипті, доступні в шаблоні за допомогою такого синтаксису: {імя_об'екта->імя_свойства}.

5. Змінні, завантажені з конфігураційних файлів, полягають між символами #. Також вони доступні як елементи асоціативного масиву $smarty.config.

Наприклад: {#bodyBgColor#} или {$smarty.config.bodyBgColor}

6. Крім того, існує змінна {$smarty}, зарезервована для деяких спеціальних змінних шаблону, таких як змінні HTTP запиту, дати і часу, і т.п.

У шаблонах Smarty визначений ряд модифікаторів, які можна застосовувати до змінних, призначених для користувача функцій або рядків з тим, щоб модифікувати їх значення. Щоб застосувати модифікатор, потрібно вказати його назву після вертикальної межі, наступної за ім'ям змінної, функції або рядком до якої він застосовується. Список усіх функцій и модифікаторів можно знайти в документації Smarty.

2.5.2 Скріптова мова JavaScript

Javascript — скриптова мова найчастіше використовується при створенні сценаріїв поведінці браузера, вбудованих у веб-сторінки. Є однією з реалізацій мови Ecmascript.

Javascript в даний момент повністю займає нішу браузерних мов. Javascript також знаходить вживання як скриптова мова доступу до об'єктів застосувань. Платформа Mozilla (Xul/gecko) використовує Javascript. Серед сторонніх продуктів, наприклад, Java, починаючи з версії 6 містить вбудований інтерпретатор Javascript на базі Rhino. Сценарії Javascript підтримуються в таких

програмах Adobe, как Adobe Photoshop, Adobe Dreamweaver, Adobe Illustrator або Adobe InDesign.

Javascript володіє рядом властивостей об'єктно-орієнтованої мови, але підтримка об'єктів в ній відрізняється від традиційних об'єктно-орієнтованих мов. Крім того, Javascript має ряд властивостей, що властиві функціональним мовам — функції як об'єкти першого рівня, об'єкти як списки, анонімні функції замикання (closures) — що додає мові додаткову гнучкість.

Javascript має С-подобний синтаксис, але в порівнянні з мовою С має наступні корінні відзнаки:

- об'єкти, з можливістю інтроспективної і динамічної зміни типа через механізм прототипів;
- функції як об'єкти першого класу;
- автоматичне приведення типів;
- автоматична збірка сміття;
- анонімні функції.

2.5.3 Технологія AJAX

AJAX — це модна назва для набору техніки розробки веб-інтерфейсів, що дозволяють робити динамічні запити до сервера без видимого перезавантаження веб-сторінки: користувач не помічає, коли його браузер запрошує дані.

Сайт, зроблений за допомогою AJAX, суб'єктивно працює набагато швидше за звичайний. Принаймні, він швидше відгукується на будь-які дії користувача. Класичні і, мабуть кращі приклади використання AJAX - проекти Google Maps і Gmail. Запити користувачів обробляються дуже швидко, бо використання ідеології AJAX дозволяє не перезавантажувати сторінку цілком, а оновлювати на ній лише ті елементи, які вимагають оновлення.

AJAX розшифровується як Asynchronous Javascript + XML (асинхронний Javascript+xml). Це просто абревіатура, що позначає підхід до створення веб-додатків за допомогою наступних технологій:

- стандартизована вистава силами XHTML і CSS;
- динамічне відображення і взаємодія з користувачем за допомогою DOM;
- обмін і обробка даних у вигляді XML і XSLT;
- асинхронні запити з допомогою XMLHttpRequest;
- Javascript, що зв'язує всі разом.

Якщо в стандартному веб-додатку обробкою всієї інформації займається сервер, тоді як браузер відповідає лише за взаємодію з користувачем, передачу запитів і виведення HTML, то в Ajax-додатку між користувачем і сервером з'являється ще один посередник - движок AJAX. Він визначає, які запити можна обробити "на місці", а за якими необхідно звертатися на сервер.

Поведінка сервера теж змінилася. Якщо раніше на кожен запит сервер видавав нову сторінку, то тепер він посилає лише ті дані, які потрібні клієнтові, а HTML з них прямо в браузері формує движок AJAX.

Асинхронність виявляється в тому, що далеко не кожен клік користувача доходить до сервера, причому зворотне теж справедливо - далеко не кожна реакція сервера обумовлена запитом користувача. Велику частину запитів формує движок AJAX, причому його можна написати так що він завантажуватиме інформацію превентивно, передбачаючи дії користувача.

Зрозуміло, що з такою схемою роботи якісне навантаження на сервер змінюється - якщо раніше запитів було мало, але кожен з них вимагав значних ресурсів, то тепер завдання сервера спрощується (формувати веб-сторінки не потрібно, та і об'єм передаваних даних менший), але запитів обробляти доводиться більше.

2.6 Програмні засоби, що використовувалися для розробки проекту

Написання коду, складання моделі бази даних, написання стилів елементів HTML можливо максимально спростити шляхом використання спеціалізованих програм. Це дозволить істотно заощадити час програміста.

2.6.1 Редактори коду і стилів веб-сторінок

Для редагування скриптів PHP, Javascript, SQL запитів, а також гіпертекстової розмітки HTML можливо використовувати програму Phpdesinger. PHP Designer - середа розробки, об'єднуюча багато потужних особливостей таких як: схеми синтаксису PHP, HTML, XHTML, CSS, Perl, C *, Javascript, VB Java і SQL (Ingres, Interbase, MSSQL, MySQL, Oracle, Sybase і Стандартний SQL), class/include браузер, тестування і відладка скриптів за допомогою інтерпретатора PHP, інтеграція керівництва PHP, автоматичне закриття близьких дужок доступ до спільних бібліотек code/script - все це об'єднано в одній програмі. Також ідеально підходить для Wamp/lamp- і Ajax-розробників, оскільки володіє засобами для автоматичного підсвічування коду. Програма має об'ємні бібліотеки, що містять більш ніж 3 тис. функцій, доступ до яких можна легко здійснювати в процесі програмування. Інтерфейс програми створений з вбудованими помічниками, щоб полегшити процес написання коду. Крім того, програма перекладена на безліч мов, включаючи російську і українську мови.

Особливості програми:

1) підтримка php, html, mysql, xml, css, javascript, vbscript, java, c, python и ruby;
2) інтелектуальне підсвічування синтаксису коду;
3) відладка скриптів;
4) автоматичне закриття близьких дужок;
5) вбудовані помічники;
6) мультимовний інтерфейс, включаючи російську і українську мови;
7) можливість зміни оформлення інтерфейсу.

Для створення каскадних таблиць стилів, що використовуються при відображенні HTML сторінок можливе використання програми Rapid CSS, Dreamwaver MX, PHP Designer, проте лише Rapid CSS надає можливість інтерактивного передогляду.

2.6.2 Програми роботи з базами даних

Для управлінням базами даних можливо використовувати такі програми як Sqlyog і Navicat. Вони надають графічний інтерфейс управління базами даних і дозволяють зручно переглядати структуру або дані з таблиць бази даних. Для створення моделі бази даних, а так само управління створеними базами даних використовується Allfusion Erwin Data Modeler і Navicat for MYSQL. Дані програми дозволяють полегшити працю розробника і адміністратора СУБД.

ERwin Data Modeler надає наступні переваги:

1. Збільшена продуктивність завдяки зручній у використанні графічній середі, яка спрощує проектування баз даних і автоматизує багато трудомістких завдань. Allfusion Erwin Data Modeler прискорює процес створення високоякісних і високопродуктивних баз даних і сховищ даних.

2. Ефективне спілкування між адміністраторами баз даних і розробниками завдяки спільному і повторному використанню моделей, а також графічному відображенню громіздких і складних масивів корпоративних даних в зручному для розуміння і супроводу форматі.

3. Швидке реагування на змінні потреби бізнесу завдяки покращуваному розумінню впливу зміни властивостей інформації в масштабі всієї організації і полегшеному швидкому впровадженню цих змін.

Можливості ERwin:

1. Керівники проектів можуть за допомогою Erwin Data Modeler ретельно задокументувати структуру БД, отримати звіти презентаційної якості і забезпечити ефективне управління проектом, використовуючи середу для спільного проектування Allfusion Model Manager.

2. Оскільки Erwin Data Modeler підтримує роботу з БД на фізичному рівні, враховуючи особливості кожної конкретної СУБД, адміністратори БД можуть з його допомогою максимально підвищити продуктивність інформаційної системи.

3. Користувач описує структуру даних візуально.

4. Erwin Data Modeler дозволяє по вже існуючих файлах БД відновлювати логічну структуру даних. Це називається зворотнім проектуванням. Воно дозволяє, по-перше, переносити структуру БД з однієї СУБД в іншу і, по-друге, досліджувати старі проекти. Цей процес найбільш поширений при переході з однієї технології на іншу (з файл-сервер на клієнт-сервер), а також при зміні сервера БД.

5. Erwin підтримує пряме і зворотне проектування 20 типів баз даних різних виробників, від настольних до реляційних СУБД і спеціалізованих СУБД, призначених для створення сховищ даних.

Основні характеристики AllFusion ERwin Data Modeler:

1. Підтримка стандартної нотації Idef1x для Er-діаграмм моделей даних, нотації IE і спеціальній нотації, призначеній для проектування сховищ даних, - Dimensional.

2. Підтримка проектування інформаційних сховищ (на основі Red Brick і Teradata).

3. Підтримка спільного проектування (версія для Modelmart).

4. Підтримка тригерів, процедур, що зберігаються, і шаблонів.

5. Розвинені засоби перевірки коректності моделей даних Reverse Engineering (генерація моделі даних на основі аналізу існуючої бази даних), включаючи відновлення зв'язків по індексах.

6. Автоматична генерація SQL DDL для створення баз даних.

7. Повна сумісність і підтримка 20-ті типів СУБД на основі прямого доступу до системного каталога баз даних (відпадає потребам у використанні ODBC).

Відмітні особливості:

1. Архітектура рівня проектування (Design Layer). Забезпечується гнучкість генерації моделей даних, повністю відповідна потребам організації. Продукт підтримує роздільні логічні і фізичні моделі, поряд із змішаними моделями. Зберігається знання стосунків і хронологія всього процесу проектування, що дозволяє користувачеві швидко визначати вплив змін, зроблених на одному рівні, на наступний рівень.

2. Технологія трансформації (Transform Technology). Фізична структура бази даних рідко відповідає оригінальній логічній структурі. Для досягнення прийнятної продуктивності сучасні ebusiness-застосування вимагають денормалізації таблиць. Технологія трансформації Erwin Data Modeler дозволяє реалізувати цього типа змін, підтримуючи цілісність структури оригіналу.

3. Визначення стандартів (Defining Standarts). Erwin Data Modeler забезпечує визначення і подальшу підтримку стандартів за допомогою словника доменів (Domain Dictionary), редактора стандартів іменування (Naming Standards Editor) і редактора стандартів типів даних (Datatype Standards Editor). Словник доменів містить повторно використовувані атрибути і забезпечує вживання несуперечливих імен і визначень на всьому протязі проектування бази даних. Редактор стандартів іменування дозволяє користувачам створювати словник допустимих слів, скорочень і правил іменування, які можуть бути використані на всьому протязі процесу моделювання даних підприємства.

4. Управління великими моделями. Erwin Data Modeler полегшує управління моделями великих підприємств за рахунок використання наочних областей (Subject Areas) і відображень, що зберігаються (Stored Displays). Наочні області надають індивідуальним проектувальникам можливість сфокусованого погляду розділяючи модель на дрібніші. Відображення, що зберігаються, надають множинні графічні представлення моделі або її наочних областей, тим самим, полегшуючи обмін інформацією між спеціалізованими групами користувачів.

5. Повне порівняння (Complete Compare). Ця технологія автоматизує синхронізацію моделі і бази даних. Вона порівнює модель з базою даних, відображує відмінності і дозволяє користувачеві вибрати, які відмінності необхідно перемістити в модель, а які згенерувати в базі даних. Якщо зміни моделі переміщені в базу даних, автоматично генерується скрипт, що змінює базу даних.

6. Генерація схеми бази даних. У Erwin Data Modeler включені оптимізовані шаблони тригерів посилальної цілісності, і потужна міжплатформенна макро-мова, що підтримує налаштування тригерів і процедур, що зберігаються.

7. Проектування сховищ і вітрин даних. Продуктивність, придатність для використання, а отже, і цінність сховищ даних визначається лежачими в їх основі проектними вирішеннями. Erwin Data Modeler надає техніку моделювання специфічну для проектування сховищ даних - таку як розмірне моделювання за схемою "зірки" або "сніжинки" - додаючи упевненість проектувальникам, що сховище даних оптимізоване як по продуктивності, так і по аналітичних можливостях. Крім того Erwin Data Modeler здатний збирати і документувати широкий спектр інформації про сховище даних, включаючи джерела даних, логіку трансформації даних і правила управління даними.

Засоби розрахунку об'єму дозволяють точно оцінити первинний розмір і характер зростання бази даних або сховища, полегшуючи ефективний розподіл ресурсів системи і планування потужності.

Для управління базами даних і створення таблиць використовується програма Premiumsoft Navicat MySQL.

Premiumsoft Navicat MySQL – це графічна утиліта для роботи з базами даних MySQL. Легкий і інтуітивно зрозумілий інтерфейс робить Navicat незамінним інструментом для роботи. Окрім звичайних функцій для адміністрування баз даних, Navicat також пропонує функції імпорта/експорта створення резервних копій і пересилки даних за допомогою зручних помічників, дозволяє конвертувати Acess в MySQL, MS SQL в MySQL, Excel в MySQL і синхронізувати.

Основні функції PremiumSoft Navicat: HTTP Tunnel; SSH Tunnel; синхронізація даних і структури; SQL консоль; підтримка всіх версій MySQL; підтримка множинних з'єднань для локальних і видаленних MySQL серверів; створення і видалення баз даних, таблиць, індексів і користувачів; підтримка Unicode; імпорт/експорт даних в 5 найбільш популярних форматів: XLS, CSV, TXT, DBF і XML; створення і запуск SQL запитів; можливість виконання основних завдань за розкладом; підтримка перенесення даних з одного MySQL сервера на іншій; створення резервних копій і відновлення баз даних; управління правами.

3 РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Етапи розробки

Інформаційно-довідковий веб-сервер для школи повинен мати два обов'язкові компоненти: база даних для зберігання інформації про користувачів, розклади і щоденники і програма веб-сервер,що відповідає за прийом запитів від користувачів і передачу користувачам, що генеруються php-скріптамі HTML сторінок із запрошуваною інформацією. Структура інформаційно-довідкового веб-сервера, що розробляється, представлена на рисунку 3.1.

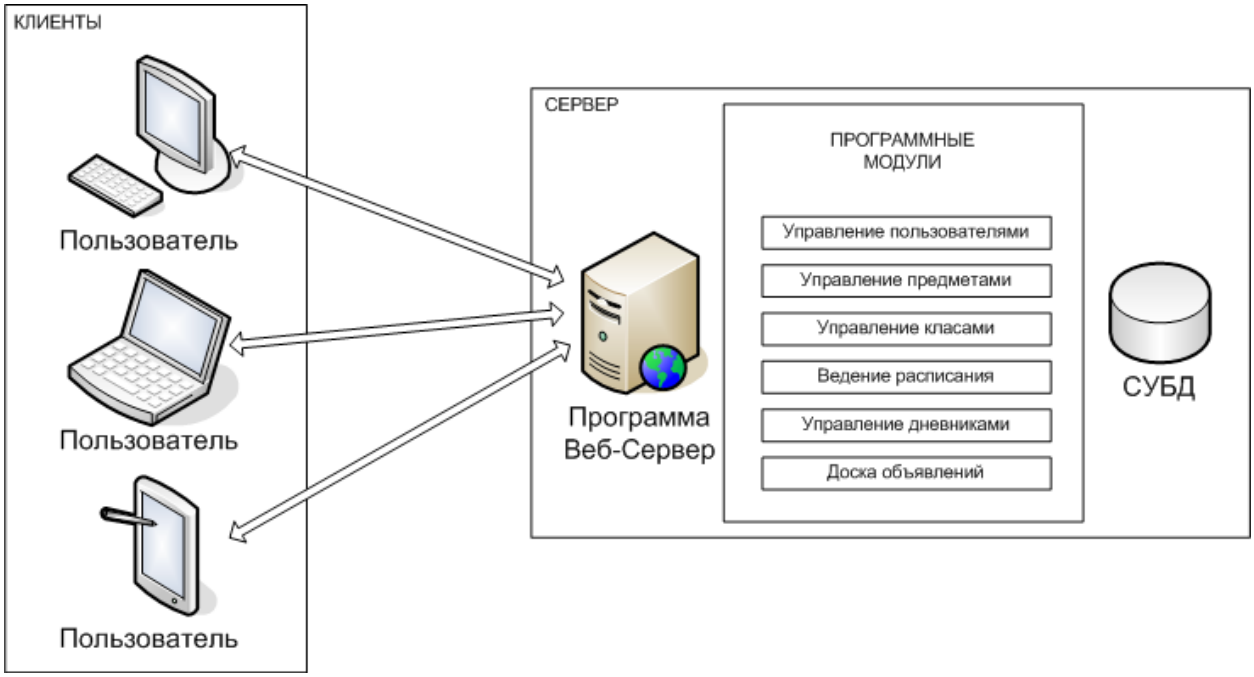


Рисунок 3.1 Структура інформаційно-довідкового веб-сервера

Дана структура дозволяє всім «клієнтам» інформаційної системи одночасно отримувати дані. Принцип взаємодії клієнта з сервером представлен на рисунку 3.2.

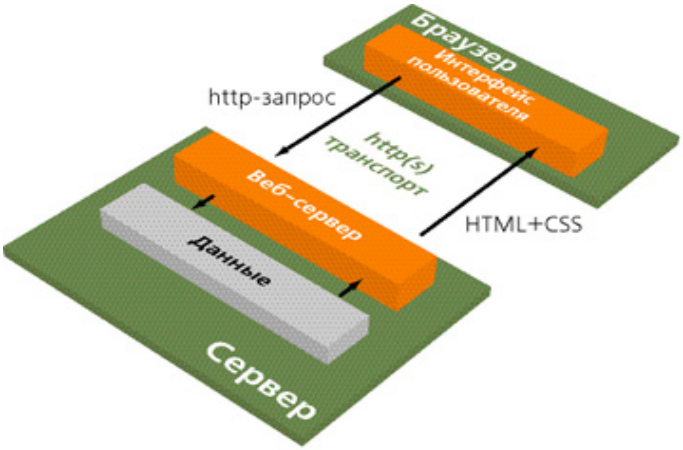


Рисунок 3.2 – Принцип обміну даними між сервером і клієнтами

Процес роботи з системою є послідовною відправкою http-запроса і здобуття відповіді у вигляді HTML сторінки і додаткових файлів (css, js, та інш.). Веб-сервер обробляє запит і передає його запрошуваному php-скрипту. Далі скрипт виконує необхідні дії, отримує дані від СУБД, формує з них HTML сторінку. Потім готова сторінка передається клієнту.

На основі вибраних програмних засобів інформаційно-довідковим веб-сервером буде набір PHP-скриптів, файлів та шаблонів TPL (що беруть участь в генеруванні сторінки на основі інформації з бази даних), а так само графіки і скриптів Javascript (переданих браузеру клієнта у разі потреби).

Процес створення сайту інформаційно-довідкової системи можна розділити на наступні етапи:

1) розробка структури бази даних з використанням CASE- засобів;
2) створення таблиць в базі даних відповідно до розробленої структури;
3) створення структури окремого компоненту (генеруємі сторінки) системи;
4) створення допоміжних функцій і класів, спільних для кожного компоненту системи;
5) створення спільного шаблону компоненту системи;
6) створення компонентів, що дозволяють управляти користувачами, класами, предметами;

7) створення компонентів, що дозволяють управляти розкладом, щоденниками, звітами;

8) створення додаткових компонентів системи (повідомлення, дошка оголошень, протоколювання дій в системі).

3.2 Розробка структури бази даних

Обробка рахунків, електронна торгівля, аналіз даних, управління знаннями - все це неможливо без використання баз даних. Системи з архітектурою клієнт/сервер будуються на основі реляційних серверних СУБД. Застосування для Internet і інтрасетей здійснюють доступ і динамічне оновлення даних. Пакети програм необхідно адаптувати і інтегрувати з існуючими системами. Сховища даних об'єднують і інтегрують безліч баз даних, забезпечуючи необхідні бізнесу гнучкість і інтелектуальність. Успіх вживання всіх цих застосувань залежить від того наскільки добре спроектована база даних.

База даних є основою інформаційно-довідкової системи, тому розробка починається саме з її структури. Насамперед необхідно створити модель даних майбутньої бази даних. Для представлення інформаційної моделі даних використовується Case-засіб Allfusion Erwin Data Modeler. З його допомогою при проектуванні моделі інформаційно-довідкової системи «Школа» була створена логічна модель.

База даних представлена у вигляді сутностей, їх атрибутів і зв'язків між ними. Кожна суть представляє безліч подібних об'єктів, званих екземплярами. Кожен екземпляр індивідуальний і повинен відрізнятися від всіх останніх. Атрибут виражає певну властивість об'єкту. З точки зору моделі бази даних сутності відповідає таблиця (наприклад «schools», «users»), екземпляру сутності – рядок в таблиці, а атрибуту – колонка таблиці. В результаті проектування були створені сутності представлені в таблиці 3.1.

Таблиця 3.1 – Сутності проектованої бази даних

Назва сутності	**Опис сутності**
schools	Школи, для яких створюватиметься інформаційна система
users	Властивості користувача. П.І.П, Рівень доступу.
auth	Логіни, паролі для користувачів, яким дозволено заходити на сайт.
profiles	Додаткова інформація про користувача. Номери телефонів, будинок. адреса та інші.
classes	Властивості класів (назва, рік початку навчання)
lessons	Властивості предметів. Назва, викладачі, призначені звістки даний предмет
periods	Періоди навчання, що визначають проміжки часу. На ці проміжки часу можливе виставляння спільних оцінок або створення звітів по успішності
schedule	«Расписание». Призначається по трьох властивостях: тиждень, день, номер уроку. Так само даная сутність містить інформацію про домашні завдання для класу.
diary	«Дневник». Безпосередньо залежить від Розкладу. Містить оцінки і персональне завдання, а так само прапор присутності на уроці і можливу причину відсутності.
actions	Містить записи про зміни в розкладі, щоденниках. На основі цих даних, можливо робити сповіщення про останні зміни подій в інформаційній системі.
messages	Містить тексти повідомлень внутрішньої пошти. Крім того, має прапор прочитання листа і видалення відправником або одержувачем.
holidays	Містить дати, які будуть відмічені в розкладі і щоденниках як святкові дні.

Зв'язок на діаграмі відображує логічну залежність однієї сутності від іншої. У Idef1x розрізняють залежну і незалежну сутність. Тип сутності визначається її зв'язком з іншою сутностю. Ідентифікуючий зв'язок встановлюється між незалежною (батьківський кінець зв'язку) і залежною (дочірній кінець зв'язку) сутностю. Екземпляр залежної сутності визначається лише через відношення до батьківської сутності. Залежна сутність змальовується на діаграмі прямокутником з закругленими кутами.

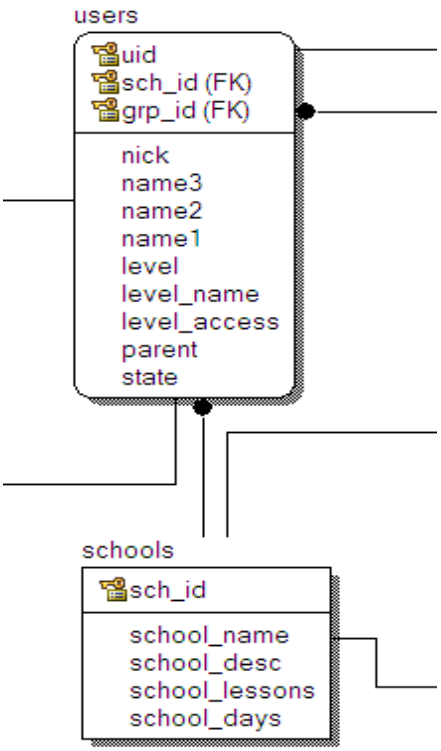


Рисунок 3.3 – Створення залежностей

На рисунку 3.3 залежною сутністю є: «users». Батьківськими для неї є сутність «schools». При встановленні неідентифікуючого зв'язку дочірня сутність залишається незалежною, а атрибути первинного ключа батьківської сутності мігрують до складу не ключових компонентів. Неідентифікуючий зв'язок служить для скріплення незалежних сутностей.

Для того, щоб однозначно ідентифікувати екземпляр сутності використовується первинний ключ (атрибут або група атрибутів). Атрибути первинного ключа на діаграмі не вимагають спеціального позначення - це ті атрибути, які знаходяться в списку атрибутів вище за горизонтальну лінію.

При встановленні ідентифікуючого зв'язку атрибути первинного ключа батьківської сутності автоматично переносяться до складу первинного ключа дочірньої сутності. Ця операція доповнення атрибутів дочірньої сутності при створенні зв'язку називається міграцією атрибутів. У дочірній сутності нові атрибути позначаються як зовнішній ключ - (FK).

В результаті розробки була отримана модель бази даних, що забезпечує основну функціональність інформаційної системи. Так само була визначена сутність, надаюча можливості ведення календаря подій і обміну повідомленнями.

Кінцева модель бази даних представлена на рисунку 3.4.

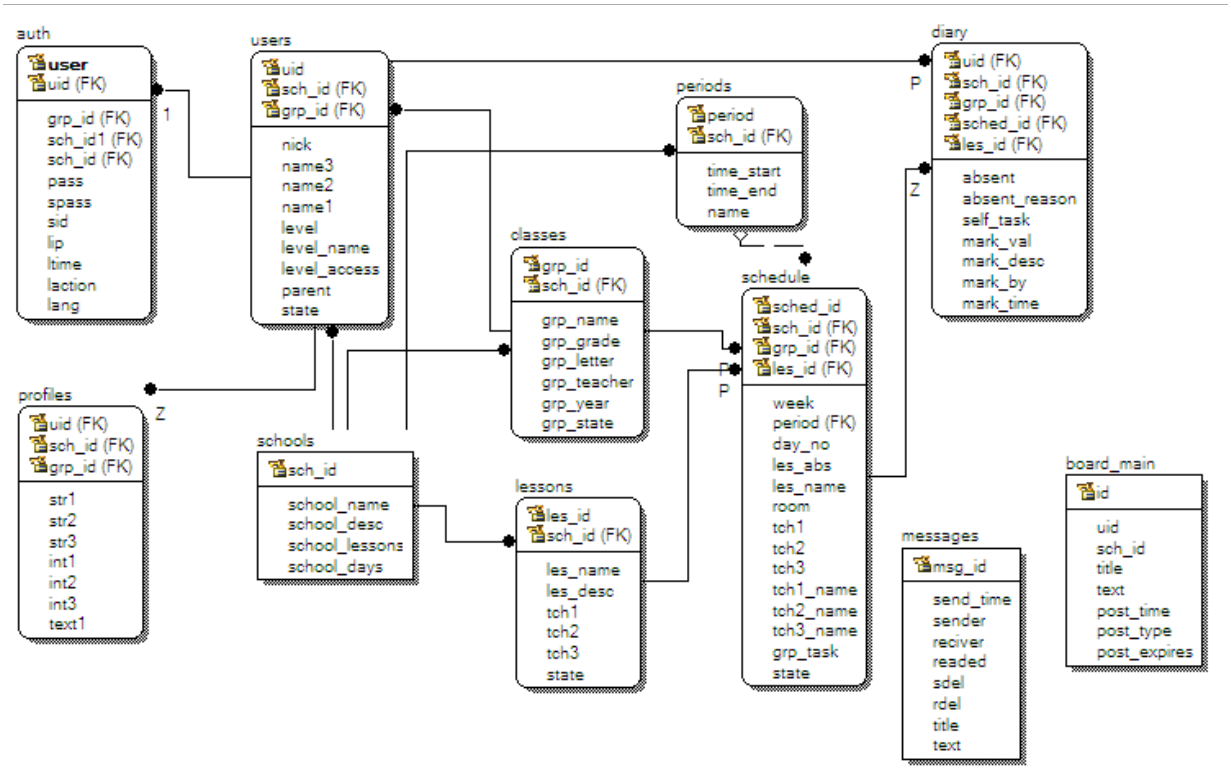


Рисунок 3.4 – Кінцева модель бази даних інформаційної системи

На основі створеної моделі були створені таблиці в базі даних. Для створення таблиць використовувалася програма Premiumsoft Navicat. Дана програма має зручний інтерфейс для створення баз даних, а так само редагування таблиць. Має підтримку всіх функцій СУБД MySQL. А так само дозволяє створювати резервні копії баз даних і відновлювати їх. При створенні бази даних користувачеві надають вибрати кодування за умовчанням для таблиць даної бази даних. Форма створення бази даних представлена на рисунку 3.5.

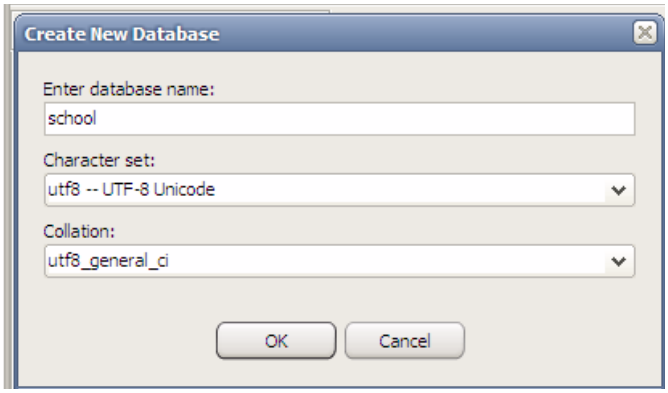


Рисунок 3.5 – Створення бази даних за допомогою Premiumsoft Navicat

В даному випадку кодуванням була вибрана utf8 для забезпечення підтримки спеціальних символів в «Домашних заданиях» таблиць «Расписание» і «Дневник».

Далі в базі даних створюються таблиці. Редактор таблиць представлений на рисунку 3.6.

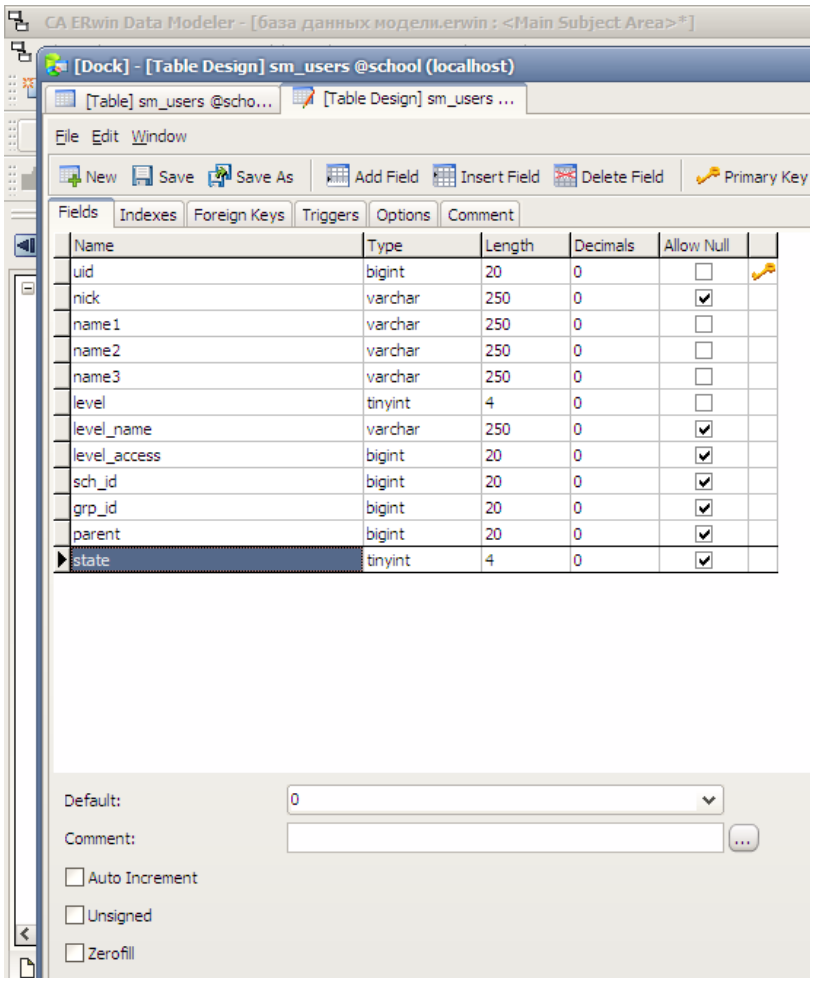


Рисунок 3.6 – Редактор таблиць в PremiumSoft Navicat

У редакторові таблиць у вкладці Fields задаються назва рядків таблиці, тип даних, розмір, значення за умовчанням, а так само первинні ключі. Так само можлива установка індексів і Foreign Keys. У вкладці Options надається можливість вибору типа таблиці, кодування і встановлення початку лічильника. На рисунку 3.7 показані спільні для всіх створених таблиць властивості.

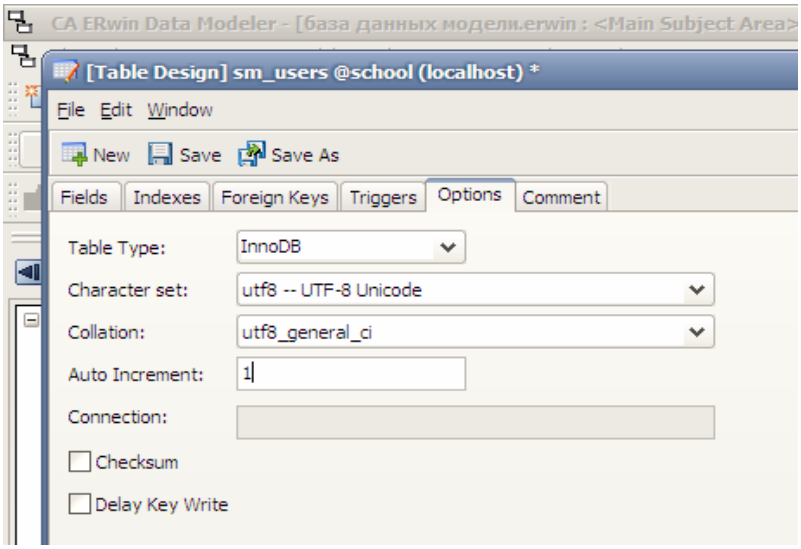


Рисунок 3.7 – Спільний вид редактора властивостей таблиці

Як тип таблиці (Table Type) вибраний тип INNODB, що дозволяє використовувати механізм транзакцій. Проте основна причина вибору саме цього типу таблиць полягає в покращуваному захисті від втрати даних (наприклад, при відключенні живлення комп'ютера).

Використовуючи розроблену модель даних, були створені наступні таблиці: sm_schools, sm_users, sm_auth, am_profiles, sm_classes, sm_lessons, sm_periods, sm_schedule, sm_diary.

Цих таблиць вистачає для повного функціонування системи. Таблиці sm_holidays, sm_messages, sm_board_main, sm_boad_class, sm_actions надають додаткові можливості системі, такі як оголошення, внутрішню пошту, календарні події.

До назви таблиць спеціально додається префікс “sm_”, для того, щоб виключити збіги імен таблиць, якщо використовується спільна база даних з декількома проектами. Префікс таблиць визначається у файлі конфігурації сайту.

3.3. Розробка шаблону компоненту системи

Кожен компонент інформаційної системи - це PHP-скрипт або група скриптів об'єднаних типом виконуваних завдань або типом інформації, що відображується. Так само в кожному скрипту зіставлений шаблон або група шаблонів TPL. В результаті роботи основного скрипта виходить HTML сторінка відправляється браузеру. На стороні браузера можуть завантажуватися специфічні файли Javascript, що містять функції управління даними на сторінці. Необхідність підключення Javascript файлу до HTML сторінці визначається в основному скрипті через установку масива даних Smarty

$smarty->assign("page_js",’названіє скріпта’);

При обробці шаблону перед відправкою користувачеві перевіряється наявність встановленої змінної *$page_js* і відповідно додається тег *<script>*

{if $page_js|count_characters>0}

<script type="text/javascript" src="{$page_js}.js"></script>

{/if}

Схема алгоритму функціонування основного скрипта компоненту представлена на рисунку 3.8. Використовуючи даний алгоритм роботи, можливо робити нові компоненти, створюючи лише основну логіку і вказуючи в шаблонах ім'я нового скрипта.

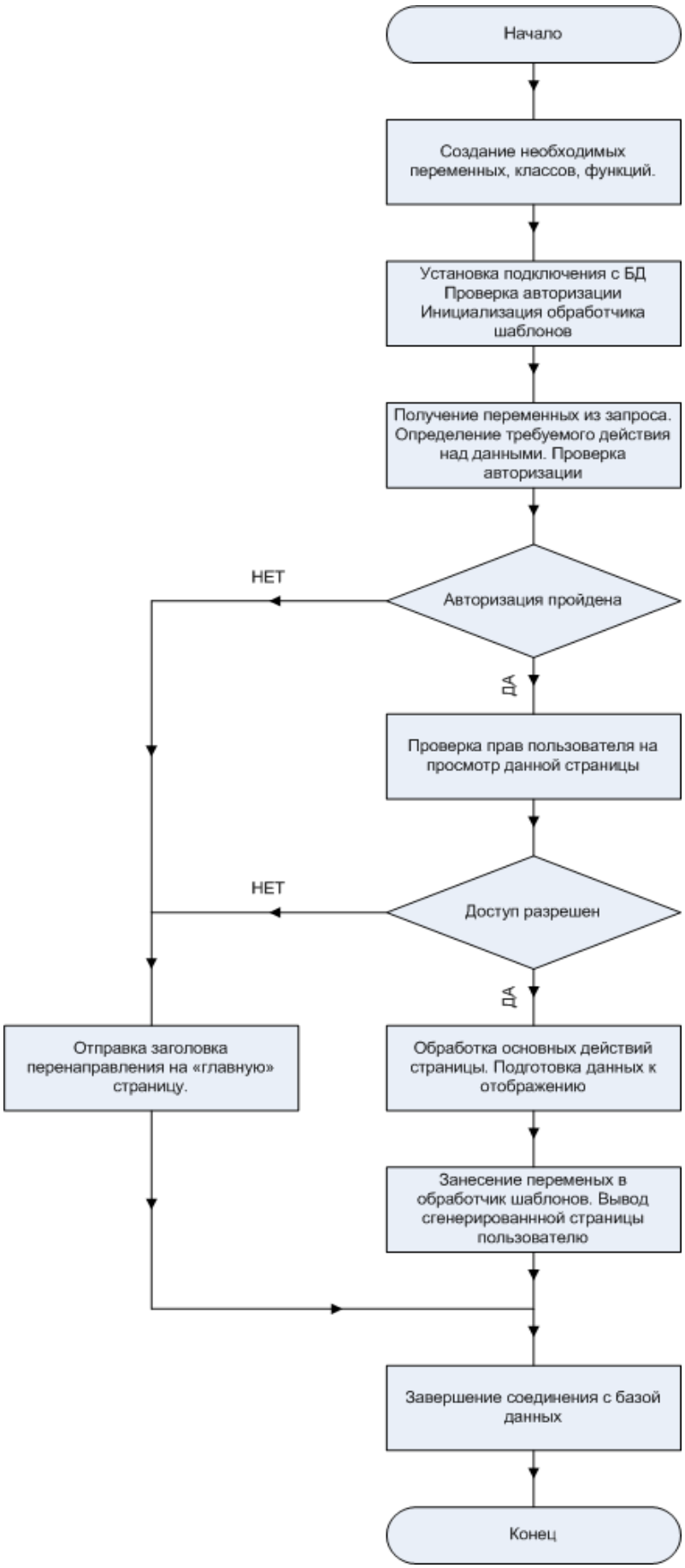


Рисунок 3.8 – Алгоритм роботи основного скрипта компоненту системи

Даний підхід забезпечує простоту зміни логіки компоненту, а так само просте створення нових компонентів, використовуючи структуру тих, що вже існують. При цьому процес обробки даних відокремлюється від створення HTML коду сторінок, що генеруються.

При створенні інформаційної системи для написання PHP скриптів використовується програма PHP Desinger. Основною перевагою використання саме цієї програми є можливість створення проектів. При цьому функції і класи, оголошені в якому-небудь файлі проекту вбудовуються в систему підказок. Таким чином полегшується процес написання коду скриптів. Спільний вид вікна програми з проектом представлений на рисунку 3.9.

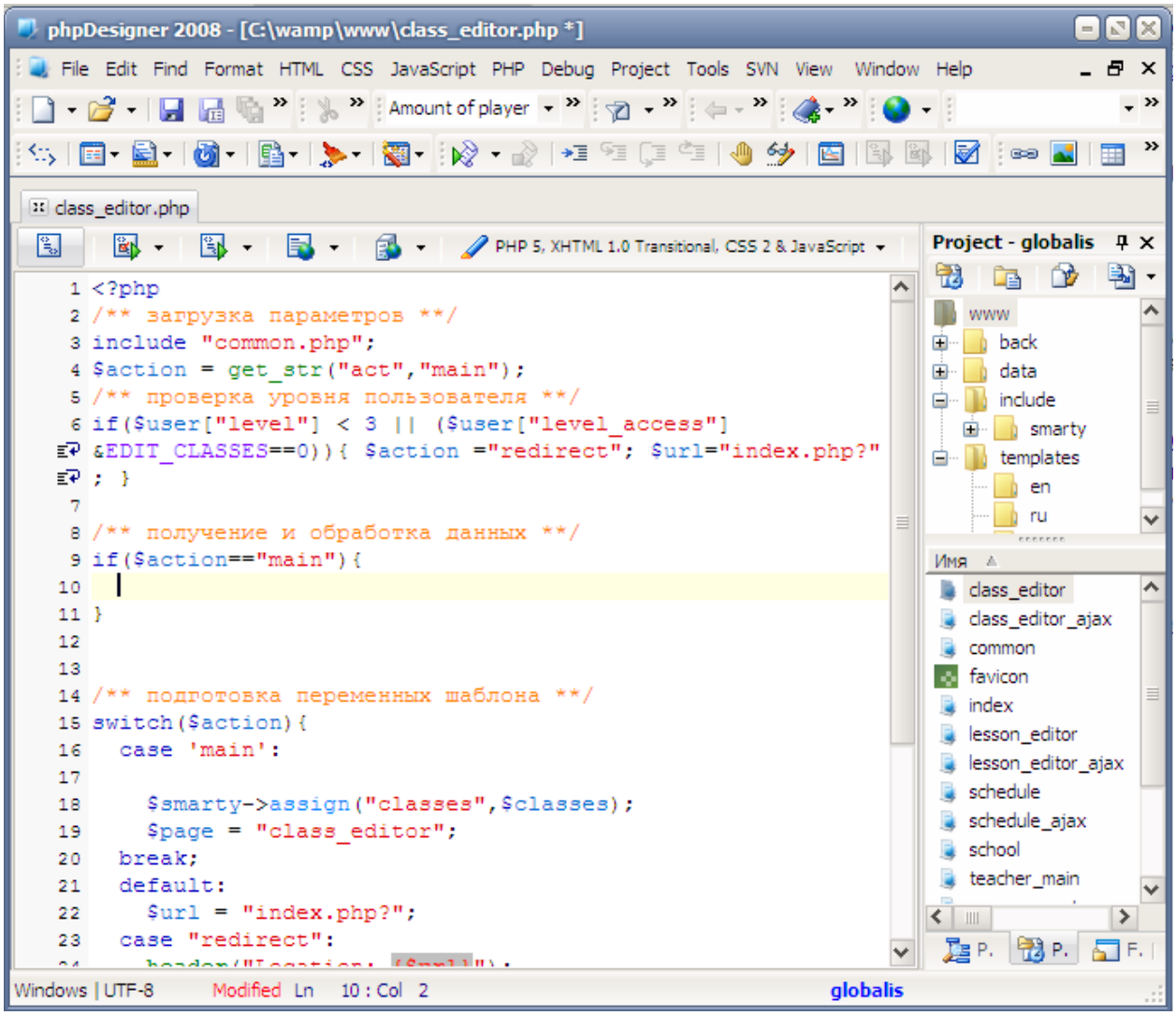


Рисунок 3.9 – Редагування php скрипта в PHP Designer

На рисунку справа, у верхній частині відображує структура тек проекту, а в нижче список файлів в теці. Так само є хороше підсвічування синтаксису, що значно полегшує розробку скриптів.

Виконання PHP скрипта починається з підключення файлу ***common.php,*** який у свою чергу завантажує параметри підключення до бази даних, визначає часто використовувані функції, а так само проводить перевірку авторизації користувача. Після завантаження змінних, скрипт запрошує із змінних, переданих користувачем, тип дії і заносить його в змінну $action. Якщо дія не визначена користувачем явно, то $action виставляєт значення за умовчанням “main”. Далі скрипт виконує операції відповідні конкретній дії, після чого заносить необхідні змінні в обробник шаблонів Smarty. В кінці файлу викликається функція відображення шаблону і закриття з'єднання з базою даних.

Файл **common.php**, що підключається на початку кожного скрипта, виконує функції **h** файлів при програмуванні на мові програмування C++.

Він виконує строгу послідовність дій:

1. Визначає шлях до теки скриптів. Це необхідно для подальшого підключення скриптів, що ініціалізували функції, класи і обробник шаблонів.

2. Підключення файлу конфігурації (логін і пароль для з'єднання з сервером бази даних, назва сесії, мова сайту за умовчанням і ін.), оголошення класу роботи з базою даних і допоміжних функцій (функції роботи з часом, рядками, запитами, а так само функція перевірки авторизації).

3. Створення класу роботи з базою даних і підключення до бази даних. На цьому ж етапі встановлюється кодування роботи з базою даних. Крім того, це дає можливість припинити виконання скрипта з виведенням помилки, якщо з'єднання з базою даних неможливе, а, отже неможлива робота з системою.

4. Перевірка авторизації користувача. На цьому етапі в масив $user заносяться дані користувача системи, або ж, якщо користувач не розпізнаний, то в масив заносяться дані, що містять нульовий ідентифікатор і тип користувача, а так само мова системи за умовчанням.

5. Підключення обробника шаблонів Smarty. Це робиться після здобуття даних користувача, оскільки при підключенні цієї бібліотеки вже буде встановлена змінна, що відповідає за мову відображення сторінок. Залежно від цієї змінної буде встановлена тека що містить шаблони для вибраної мови.

Обробивши файл **common.php**, інтерпретатор повертається в основний скрипт. Потім виконуються основні дії компоненту. І в кінці роботи основного скрипта в клас $smarty заносяться структуровані дані, отримані в результаті роботи основної частини скрипта. Після цього виконується обробка шаблону із занесеними в клас $smarty змінними. Завершується скрипт відправкою готової сторінки HTML і закриттям з'єднання з сервером бази даних.

3.3.1. Клас роботи з базою даних

Для забезпечення взаємодії з сервером бази даних використовується клас Mysql_database визначення, якого відбувається при підключенні файлу **class_database.php**, що відбувається на початку кожної сторінки. При цьому встановлюється з'єднання з сервером бази даних в кожному скрипті і завершується при завершенні роботи скрипта. Такий підхід дозволяє унеможливити вчасно видати повідомлення про помилку з'єднання з базою даних і заздалегідь завершити роботу скрипта без відображення помилок пов'язаних з відсутністю з'єднання, а, отже, приховати структуру сайту. Крім того, це дає можливість швидкого переходу на іншого типа бази даних. При цьому необхідно буде створити клас з такою ж структурою, але що використовує інші функції роботи з базами даних. При цьому текст основних скриптів не міняється. Так само використання окремого класу дає можливість протоколювати запити до бази даних (якщо в цьому є необхідність) або відображувати статистику за часом виконання запиту. У основних скриптах, що виконують обробку даних, звернення до бази йде через функцію db_query()

$db->db_query("строка SQL запроса");

Клас Mysql_database має наступну структуру:

```
class Mysql_database {
    var $database_connection;
    function Mysql_database
    function db_connect
    function db_select
    function db_query
    function db_fetch_array
    function db_fetch_assoc
    function db_num_rows
    function db_affected_rows
    function db_set_charset
    function db_real_escape_string
    function db_insert_id
    function db_error
    function db_close
}
```

Ця структура надає достатній набір функцій для повного доступу до сервера бази даних MySQL.

3.3.2. Функція перевірки авторизації користувача

Для розмежування доступу до даних інформаційно-довідкової системи існує система пізнання користувача по логіну і паролю. На початку кожного скрипта визиваєьтся функція авторизації, ідентифікуюча користувача або по введеному логіну і паролю або по приховано переданому ідентифікатору сесії користувача. Реалізація пізнання користувача системи заснована на технології php сесій.

Сесії - це механізм, який дозволить створювати і використовувати змінні, що зберігають своє значення протягом всього часу роботи користувача з сайтом. При цьому у кожного користувача сайту ці змінні будуть власними, тобто їх зона видимості (variable scope) поширюється на весь час знаходження на сайті

конкретного користувача, причому для кожного заходу користувача на сайт ці змінні будуть різними. Кажучи простіше, ці змінні належать конкретній сесії роботи конкретного користувача з сайтом.

У основі всього механізму сесій лежить рішення задачі про ідентифікацію того, від кого саме прийшов запит на сервер. Якщо це буде точно відомо, то вже не виникне великої проблеми в тому, щоб надати скрипту інформацію, що відноситься саме до цього конкретного користувача. Дане завдання вирішується шляхом привласнення кожній сесії унікального ідентифікатора SID (Session Identifier), який створюється в той момент, коли користувач заходить на сайт, і знищується в мить, коли користувач вирушає з сайту. Цей ідентифікатор передається на сервер разом з кожним запитом з боку клієнта і повертається на машину клієнта разом з результатами обробки запиту. Алгоритм генерації SID (а в PHP як ідентифікатор сесії використовується GUID (Global Unique Identifier)) дозволяє гарантувати його унікальність, тому унеможливлено те, що дві сесії матимуть один і той же ідентифікатор сесії.

Можливе використання двох різних механізмів таких, як "транспортний засіб" для передачі Sid:cookies і Параметр *query string*(рядок запиту). Cookies зручніший спосіб передачі ідентифікатора. При цьому SID зберігається "усередині" браузера і залишається непомітним для користувача. Але підтримка cookies - це необов’язкова умова для браузера, вона може бути відсутньою або бути відключена у когось з відвідувачів, тому в спільному випадку спиратися на них не можна. В цьому випадку можна використовувати менш "красивий", але надійніший спосіб - передачу SID як один з параметрів запиту. PHP має можливість автоматично додавати SID до всіх посилань в сторінках, що генеруються, HTML, тому, як правило, не потрібно буде піклуватися про те, щоб додавати цей ідентифікатор до кожного посилання вручну. Отримати ідентифікатор сесії можна у будь-який час з константи SID або з функції *session_id()*. Алгоритм роботи функції перевірки авторизації користувача представлений на рисунку 3.10.

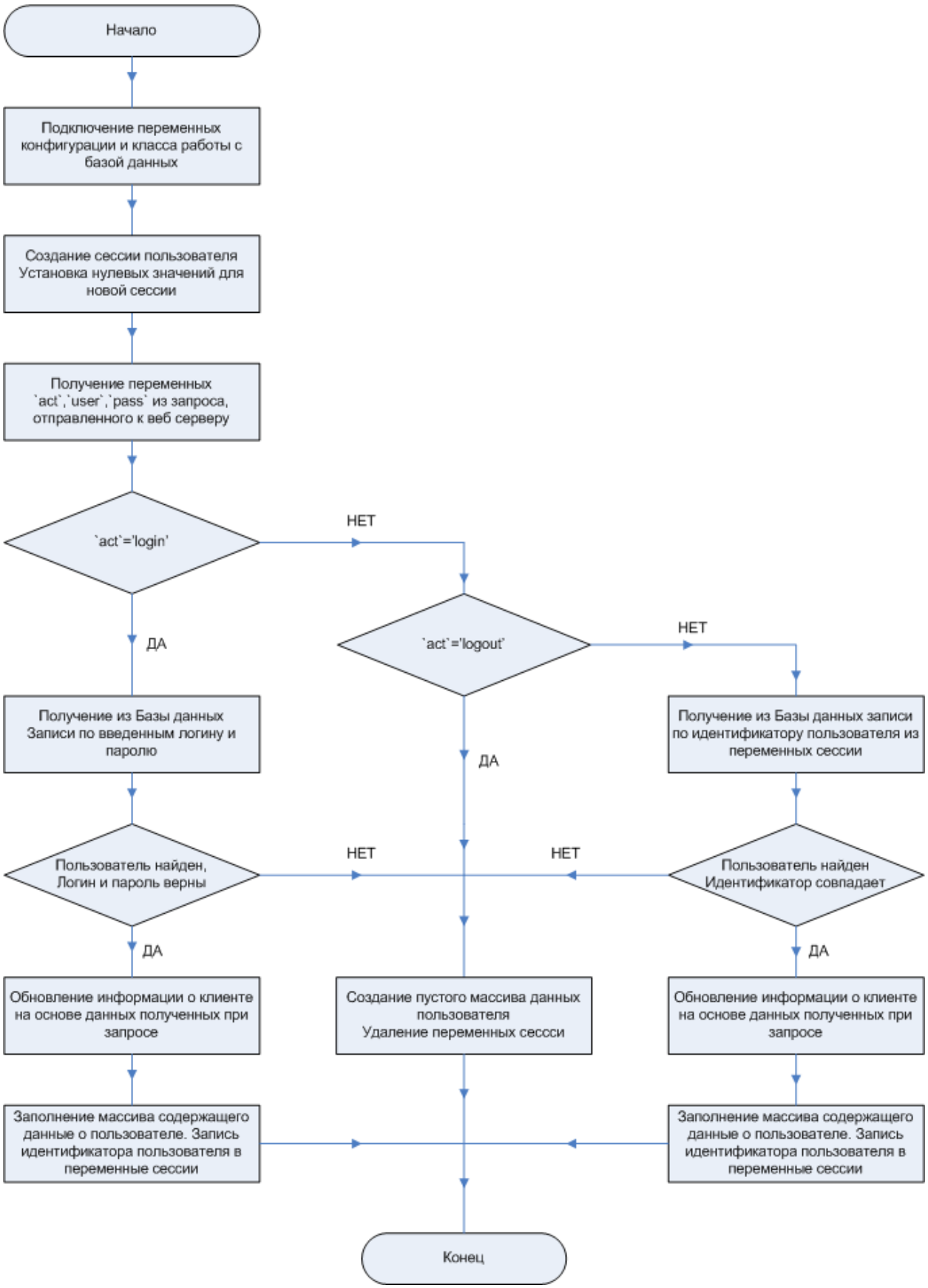


Рисунок 3.10 - Алгоритм роботи функції авторизації

На самому початку функції до її зони видимості підключаються змінні конфігурації сайту і клас роботи з базою даних. Параметри конфігурації сайту містять мову за умовчанням для користувачів, які не минули авторизацію і назву змінної сесії користувача, що містить ідентифікатор. Потім перевіряється чи встановлений в змінних даної сесії ідентифікатор користувача. В разі відсутності, змінна створюється з нульовим ідентифікатором.

Функція авторизації отримує із запиту, що прийшов від користувача, змінні `act`,`user` і `pass`. Якщо ці змінні оголошені в запиті, то відбувається перевірка змінних на правильність введення і довжину. Так для змінної `user` можливо використовувати лише символи латинського алфавіту і цифри. Змінна $act може набувати значень «login», «logout». Залежно від змінної $act виконується перевірка існування користувача з даними ім'ям користувача і паролем. Для змінної $act рівною «logout» відбувається установка ідентифікатора користувача в нуль. При цьому авторизація вважається такою, що завершилася невдало, і масив користувача містить нульові значення і мову сторінок за умовчанням. Якщо змінна $act не встановлена (це відбувається в більшості запитів), то виконується перевірка ідентифікатора користувача, що міститься в змінних сесії. Якщо користувач знайдений в базі даних, то заповнюється масив даних користувача і функція завершується з поверненням масиву, інакше повертається масив з нульовими значеннями.

3.3.3. Допоміжні функції

Так само на початку роботи скрипта визначаються функції роботи з часом, текстом, здобуттям даних із запитів. Визначення функцій, які оброблюють змінні запиту, дозволяє зменшити кількість коду в основних скриптах. Дані функції виконують перевірку на існування змінної в запиті користувача до веб-серверу, відповідність змінній даному типові або установці значень за умовчанням.

Список допоміжних функцій у файлі fuctions_global.php приведений в таблиці 3.2.

Табліця 3.2 – Список допоміжних функцій

Назва функції	Опис функції
draw_page	Викликає функцію компіляції шаблону з ім'ям, встановленим в змінній $page. Після обробки сторінка відправляється користувачеві. Постійне з'єднання з базою даних завершується. Дана функція нічого не повертає і, як правило, використовується в кінці виконання основного скрипта.
exit_script	Функція використовується для екстреного завершення роботи скрипта (наприклад, при виникненні помилки). Дана функція нічого не псує користувачеві, а просто завершує з'єднання з базою даних і зупиняє інтерпретатор PHP.
get_int	Функція отримує із змінних, що прийшли в запиті, змінну з ім'ям $name у вигляді цілого позитивного числа. При цьому, якщо знаходження змінної неможливе або вона не є цілим позитивним числом, встановлюється значення за замовчуванням.
get_str	Аналогічна попередній функції, але функція повертає рядок. Якщо набуття значення змінної неможливе, повертається значення за умовчанням.
md_day	Функція проводить коректування значення часу UNIX до найближчого початку дня. Робиться це для сумісності з часом в базі даних.
md_week md_month_b md_month_w	Функції аналогічні попередній, але вирівнювання ведеться по початку тижня, першому дню місяця, першому дню першого тижня місяця.

3.3.4. Скрипти, що виконуються на стороні клієнта

Окрім скриптів, створюючих HTML сторінки на стороні сервера, використовуються скрипти, що виконуються в браузері клієнта. Призначення даних скриптів – зменшити візуальний розмір сторінки, забезпечити можливість редагування, не покидаючи сторінки, створити елементи, що розкриваються в таблиці. Як мова написання даного типа скриптів використовується Javascript. Файли скриптів оголошуються на початку HTML сторінок, відправлених Веб-сервером. Після цього браузер підвантажує дані файли скриптів. Як правило, виклик Javascipt функцій відбувається як реакція на події. Найбільш часто

використовувані функції виведені в окремий файл, що оголошується перед файлом з функціями для конкретної сторінки. Прикладом такої часто використовуваної функції, є функція здобуття об'єкту HTML сторінки по імені (властивості id тега). При цьому в останніх виклик функції може виглядати так:

ge(“table_cell”).innerHTML=”text_in_cell”;

Назва функції ge отримана скороченням назви Getelement. Скорочення імен функцій так само здатне зменшити розмір даних, що передаються.

Функції, які виконують розкриття і приховування тексту домашнього завдання використовують CSS атрибути об'єктів сторінки, а саме style. Так виглядатиме приховування блоку тексту ge (“элемент з текстом») style.display=”none”; при цьому введений текст не знищується, а просто не відображується на сторінці.

Так само у файлах Javascript оголошуються функції для роботи з AJAX.

function ajax(url,method,params,onload);

Так за допомогою всього чотирьох параметрів можна відправити «невидимий» запит на адресу «url» методом «method» з параметрами «params». А відповідь отримати у функції вказаної в «onload» через параметр responsetext.

Javascript, використовуючи набір функцій для відображення спливаючих підказок, може створювати «Диалоговые окна» для забезпечення зручного редагування даних. На рисунку 3.11 представлено вікно редагування прав вчителя.

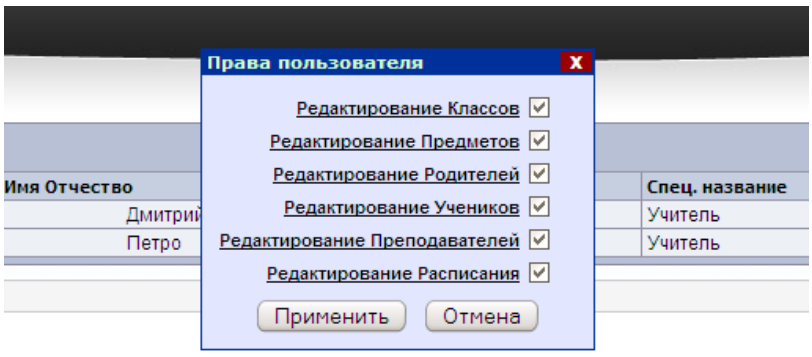


Рисунок 3.11 – Спливаюче вікно редагування прав користувача

3.4. Розробка основних компонентів системи

Після того, як всі допоміжні функції і класи створені, маємо можливість створення основних скриптів системи. Дані скрипти виконуватимуть дії над БД і запитами користувача і формуватимуть дані. Проте створювати сторінки слід в наступній послідовності:

1) Управління користувачами.

2) Управління предметами і класами.

3) Управління розкладом.

4) Управління щоденниками.

5) Додаткові операції.

Такий підхід дозволить, послідовно створюючи компоненти, мати можливість відладжувати всю систему.

3.4.1. Сторінки управління користувачами

Сторінки управління користувачами розділені на три категорії по рівню користувача (вчитель, учень, батько). Зроблено це для зменшення розміру як клієнтської частини (HTML і Javascript), так і для спрощення написання коду скриптів. Так для кожного типа користувача створюються окремі файли: users_teachers.php users_students.php і users_parents.php відповідно. Для цих файлів робляться окремі шаблони сторінок. При цьому кожна таблиця має свої стовпці, і не відбувається нагромадження даних так, як би всі користувачі виводилися в одній таблиці.

Кожен скрипт отримує дані за конкретним типом користувачів і, якщо це потрібно, то і додаткові дані, такі як призначені учні (для батьків) або призначені предмети (для вчителів). Додавання, зміна установки властивостей користувача проводиться з самої сторінки за допомогою технології AJAX. Вона робить запити на файли users_teachers_ajax.php, users_students_ajax.php і users_parents_ajax.php. Це дозволяє не виконувати перезавантаження сторінки, а, отже, повторну вибірку

даних з бази що зменшує навантаження на базу і дозволяє зменшити обмін трафіком між сервером і клієнтом. На рисунку 3.12 показано вікно призначення учня батькам.

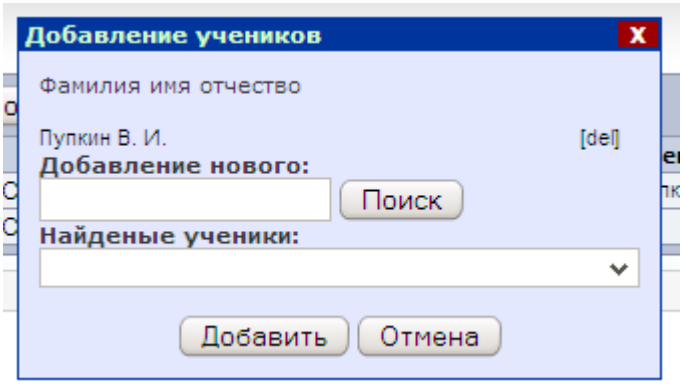


Рисунок 3.12 – Вікно додавання учня батькам

Дане спливаюче вікно дозволяє додати учня батькам. Дана функціональність забезпечується відразу декількома функціями. Так після введення тексту і натиснення кнопки «Поиск» відбувається прихований запит до скрипта users_parents_ajax.php, і якщо по введеному тексту знайдені учні, їх дані передаються назад в javascript. Після цього заповнюється список «Найденные ученики». Натиснення кнопки «Добавить» викличе функцію, що відправляє запит у файл users_parents_ajax.php з поточним ідентифікатором користувача вибраного в списку «Найденные ученики». При цьому, якщо це можливо, вибраному учневі буде призначений Батько. Якщо ж кількість вибраних учнів вже рівна трьом, то призначення не відбувається. У верхній частині вікна показано П.І.П. редагованого батька і список вже призначених учнів. Праворуч від імені учня є кнопка яка дозволяє відмінити прив'язку учня до батька. Виконання цієї функції так само відбувається в «фоновом» режимі, відправкою запиту з вказівкою необхідної дії і ідентифікатора учня.

Окремий випадок, який вимагає перезавантаження сторінки, - це додавання/ видалення користувача. При цьому повинна змінюватися кількість рядків таблиці. В тім, це завдання теж може бути вирішено засобами JavaScript і AJAX. Але

вимагає додавання скриптів як на стороні сервера, так і на стороні клієнта. Зміна інших значень, таких як «права доступа», авторизація і профіль не вимагають зміни кількості рядків, а, отже, можуть бути виконані без перезавантаження сторінки.

Редагування користувачів складається з декількох етапів. Основний етап - це завдання П.І.П. і імен, що відображуються. При цьому не робиться перевірка на збіг з вже існуючими даними. У випадку, якщо виконується додавання користувача, відбувається перезавантаження сторінки. Після створення користувач може бути знайдений за допомогою пошуку по користувачах. Пошук здійснюється по прізвищу, імені, по батькові і імені користувача, що відображується. Для створених користувачів доступне меню управління доступом до сайту (установка логіна і пароля), установка прав на окремі частини сайту установка властивостей профілю, або для батьків, додавання дітей, за якими вони зможуть спостерігати і отримувати дані. Меню управління користувачем виконане у виглядіі контекстного меню провідника Windows. Проте виконано це за допомогою спливаючих вікон і каскадних стилів. Приклад меню управління користувачем представлений на рисунку 3.13.

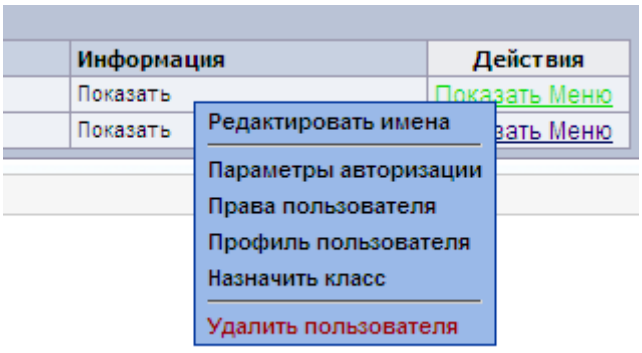


Рисунок 3.13 – Меню редагування властивостей учня

Кожен елемент меню викликає певну функцію, що виводить вікно редагування специфічних властивостей. Так для операції «Удаления пользователя» буде виведений запит підтвердження видалення користувача.

Після створення даних сторінок можливо створення трьох типів користувачів і вже з'являється можливість входу на сайт інформаційної системи, і виявлення помилок, які допущені в створених скриптах. Інформація про допущені помилки виводиться у верхній частині сторінки проте в робочому варіанті виведення помилок навмисно вимкнене. Зроблено це для приховування внутрішньої структури сайту, оскільки як правило, в тексті помилки вказується так само шлях до файлу, в якому сталася помилка.

На сторінці редагування учнів, викладач має можливість перейти на сторінку створення звітності по учневі, сторінку перегляду щоденника або перегляду класу. Можливість переходу включено в меню управління користувачем. При натисненні на елемент меню, що відповідає за перехід на сторінку створення звітів, відбувається генерація посилання з використанням ідентифікатора учня. Після створення посилання відбувається перехід.

Слід зазначити, що для доступу до сторінок управління користувачем обов'язково рівень користувача має бути вище, ніж рівень учня і виставлені права редагування конкретного типа користувачів.

3.4.2. Сторінки управління предметами і класами

Предмети і класи є основою для складання розкладу і ведення щоденників. Файли edit_lessons.php edit_classes.php відповідають за відображення списку класів і предметів, і так же, як і у випадку з користувачами, дозволяють додавати і змінювати класи і предмети. Призначення учнів в клас відбувається на сторінці редагування користувачів. Проте на сторінці редагування класів вчитель може проглянути список учнів в класі і змінити його. Так само із сторінки редагування класів можливий перехід на сторінки виставляння оцінок класу і створення звіту успішності по класу.

Редактор предметів дозволяє змінювати або створювати нові предмети. Спільний вид вікна створення/редагування предмету представлений на рисунку 3.14.

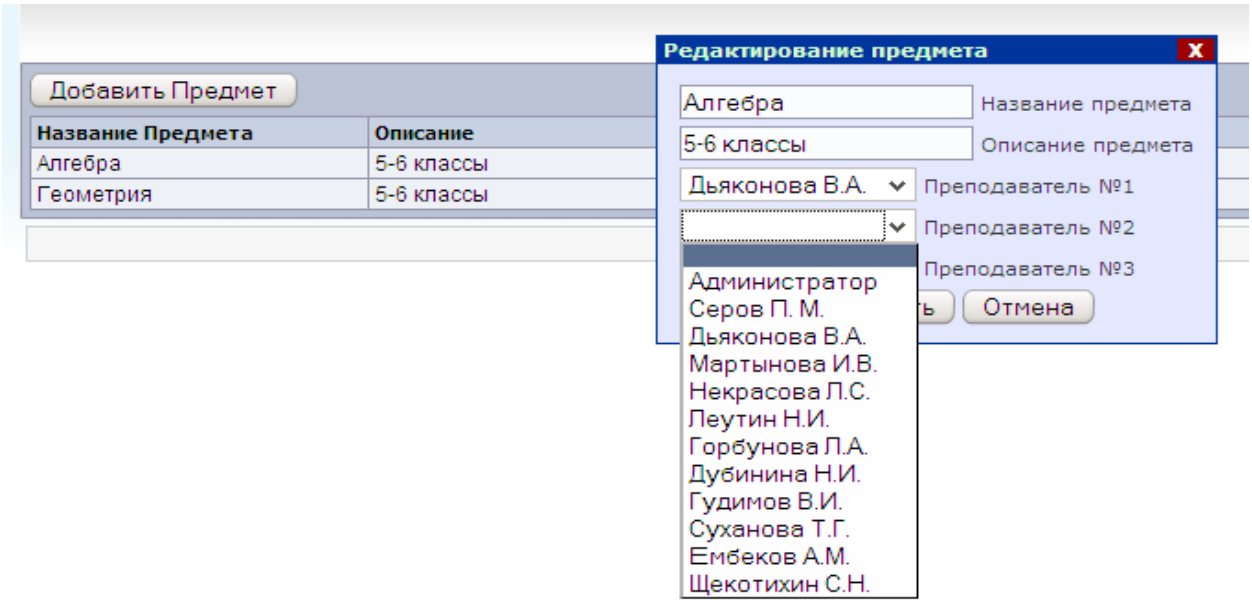


Рисунок 3.14 – Вікно редагування предмету

При редагуванні на предмет можливо додавання до трьох вчителів. Всі вибрані вчителі отримують можливість виставляти оцінки і додавати/змінювати домашнє завдання для цього предмету. Так само на сторінці редагування предметів вчитель отримує можливість проглянути спільну кількість уроків на поточний проміжок часу, що дозволить якісніше стежити за виконанням учбового плану.

3.4.3. Сторінки редагування розкладу

Файли schedule.php, schedule_periods.php забезпечують редагування розкладу і створення періодів навчання. Періодом навчання може бути чверть, півріччя або будь-який інший проміжок часу. На сторінці редагування періодів навчання задається початок і кінець періоду, при цьому періоди не перетинаються. Вікно редагування періоду часу показане на рисунку 3.15.

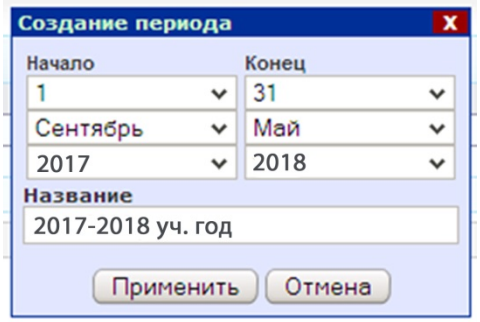


Рисунок 3.15 – Вікно редагування періоду навчання

Кнопка «Применить» відправляє прихований запит скрипту schedule_periods_ajax.php. Запит у файлі schedule_periods_ajax.php обробляється. Перевіряються всі введені значення. У випадку, якщо вказані невірні значення (наприклад, дата початка більше дати кінця періоду), то повертається код результату. Залежно від отриманого коду, проводиться або виведення підказки у верхній частині вікна редагування, або перезавантаження сторінки. У створених періодах можливе ведення статистики, а також виставляння підсумкових оцінок, враховуючи дані з таблиці щоденників.

Основою інформаційної системи є редактор розкладу. Саме редактор розкладу дозволяє створювати розклад. Розклад призначається класу на конкретний тиждень. Сторінка редактора розкладу представлена на рисунку 3.16.

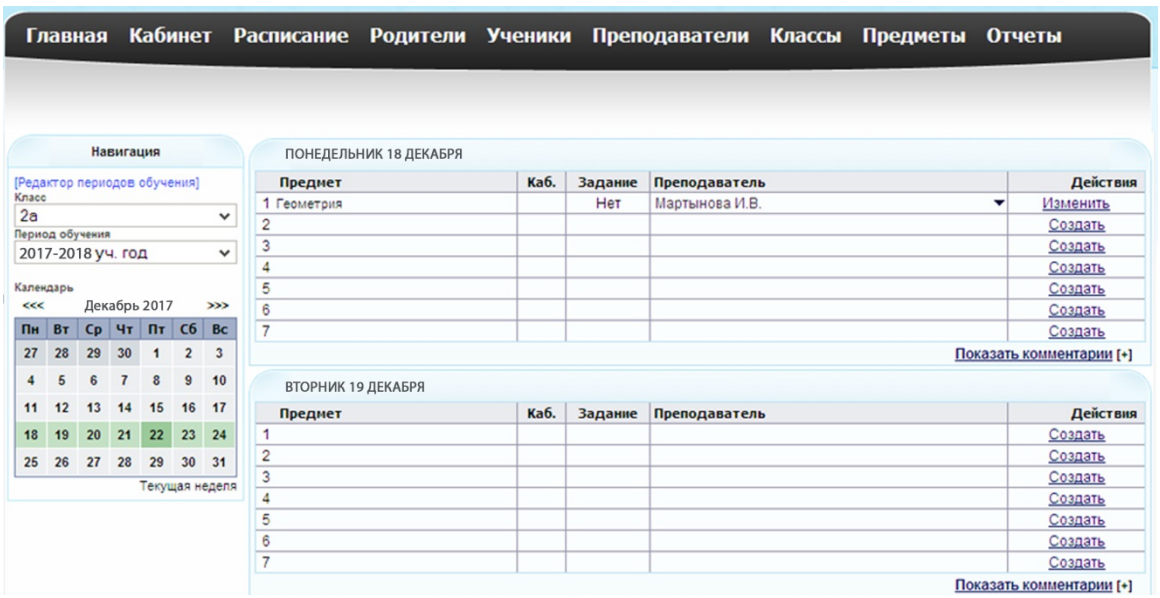


Рисунок 3.16 – Сторінка редагування розкладу класу

На сторінці редагування розкладу в лівій частині розташовується календар, і списки класів і періодів навчання. При зміні активного класу відбувається перезавантаження сторінки, при цьому в рядку запиту до сервера буде вказаний новий ідентифікатор класу *(grp_id=ідентіфікатор)*. Календар дозволяє проводити перехід по тижнях. При цьому активний тиждень виділений зеленим кольором. Скрипт виконує вибірку з бази даних значень відповідних вибраному тижню навчання. Якщо тиждень не визначений в рядку запиту до веб-серверу, то береться поточна дата і відносно її виводиться розклад.

В центрі вікна розташовуються основні таблиці розкладу. Кожна таблиця відображає розклад уроків одного дня. Натиснення на вікно колонки «Каб», «Задание» виводить вікна редагування номера кабінету і домашнього завдання для класу. Колонка «учителя» відображує призначених на конкретний урок вчителів. «Действия» містять список можливих дій для даного запису - для порожнього запису - це буде лише «Создать», а для вже створеного, меню «Изменить».

Скрипти виконуючі дії додавання і зміни запису розкладу знаходяться у файлі schedule_ajax.php. Виконавши необхідні операції, скрипт повертає створені або змінені значення, і вони заносяться в таблицю розкладу за допомогою рядка

ge(“ячейка табліци”).innerHTML = “новое значеніє”

При цьому зміни запису відбуваються майже миттєво, за рахунок того, що передаються лише необхідні для відображення дані.

На кожен день можливе додавання коментаря. Коментарі зберігаються в таблиці sm_schedule_comm бази даних. Додавання коментаря на день надає можливість сповіщення, як учнів, так і батьків про класні заходи. Додавання нового коментаря так само реалізоване у вигляді функцій тих, що виводять вікно редагування, відправка тексту запиту на сервер і здобуття коду результату виконання операції.

3.4.4. Сторінки редагування і перегляду щоденників

Сторінки diary_student.php diary_teacher.php diary_parent.php мають однаковий алгоритм вибірки даних з таблиць бази даних, але відрізняються способом визначення користувача, для якого виводиться щоденник. Так для diary_student.php проводиться вибірка по ідентифікатору користувача, що увійшов до системи (масив $user), а для останніх ідентифікатор користувача має бути переданий в рядку запиту до сервера. Якщо ідентифікатор не переданий або не існує в базі даних, то здійснюється перехід на «Главную страницу». Відрізняються так само і шаблони для виведення інформації щоденника. Для «Учителя» вони містять додаткові вікна редагування. Виставляння оцінки, маркіровка уроку як пропущеного, установка персонального завдання, коментарі – це ті додаткові функції, які надаються користувачеві з рівнем «Учитель». Учні можуть лише змінювати персональне завдання. Батьки можуть лише переглядати щоденник. На рисунку 3.17 показаний спільний вид редактора щоденника у викладача.

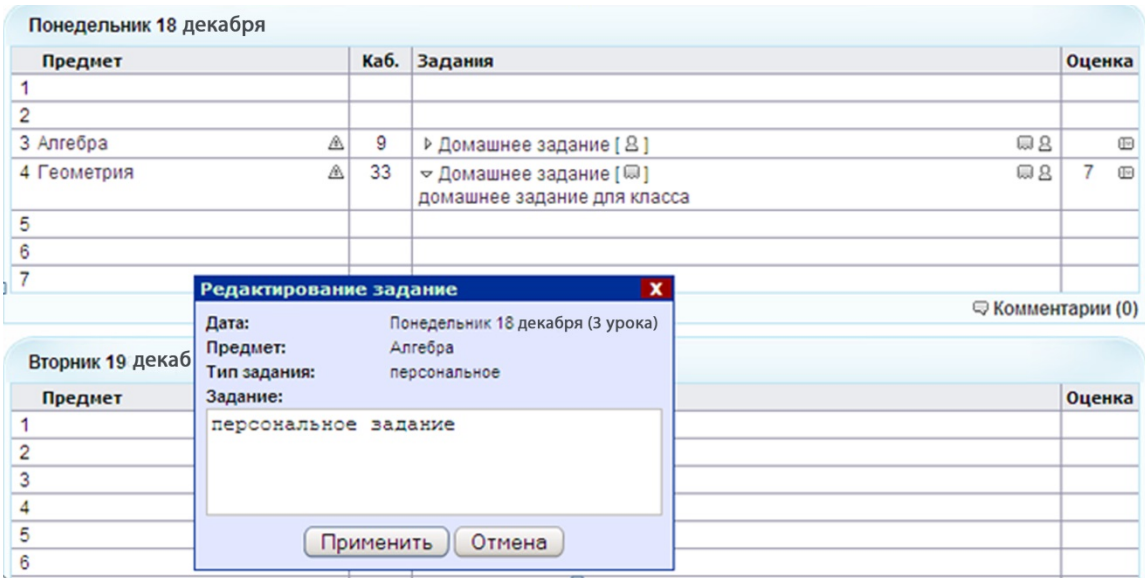


Рисунок 3.17 – Редагування щоденника учня

Редактор щоденника є щоденником з додатковими елементами управління. Так вчитель може відзначити відсутність учня, змінити завдання для класу або

персональне завдання. При натисненні на ікону в полі оцінки відкривається вікно з вибором оцінки, що виставляється, і полем введення коментаря до оцінки. Додавання нової оцінки відбувається без повторного завантаження сторінки за рахунок технології AJAX. Дані про запис, що виставляється або видаляється, передаються у файл teacher_diary_ajax.php, і відбувається або додавання запису, або видалення. Окрім цього вчитель має можливість написати новий персональний коментар на день. Даний коментар буде видний учневі і батькам при перегляді щоденника.

При додаванні оцінки, коментаря або домашнього завдання відбувається протоколювання дії вчителя. Дані про дію заносяться в таблицю actions БД. Ці дії надалі використовуються в системі сповіщення про події на сторінках «Главная», «Клас», і при перегляді профілю користувачів. Протоколювання дій дозволяє учням бути в курсі останніх змін в розкладі. Для батьків це спрощує контроль за своїми дітьми.

3.4.5. Додаткові компоненти системи

До додаткових компонентів системи відносяться повідомлення, мій кабінет, мій клас, мої діти, звіти та інші. Ці компоненти не впливають на розклад і щоденники і носять інформаційний характер.

Так на головній сторінці відображуються оголошення для школи і для всього сайту. Крім того, користувачам, що пройшли авторизацію, показуються останні події, що мають до них пряме або косвене відношення. Наприклад, для батьків - це будуть повідомлення про виставляння оцінки учневі або додавання нового оголошення для класу.

Головна сторінка має відмінну від останніх сторінок схему функціонування. Головна сторінка може бути доступна для всіх типів користувачів системи, а так само користувачів, що не пройшли авторизацію.

Сторінка Повідомлень надає можливість обмінюватися повідомленнями в межах школи і дає можливість спілкування між «Учениками», «Учителями» і

«Родителями». Дана сторінка виконана у вигляді вікна поштового клієнта та відображує вхідні і вихідні повідомлення. Скрипти на стороні клієнта дозволяють переглядати повідомлення без перезавантаження сторінки. У списку вихідних повідомлень є колонка, що інформує про прочитання листа, що дає можливість взнати чи прочитаний лист одержувачем. Прапори видалення (sdel, rdel) дозволяють приховувати повідомлення із списку. При цьому одне повідомлення може бути видалене у одержувача, але залишатися у витікаючих у відправника. Вид сторінки управління повідомленнями представлений на рисунку 3.18.

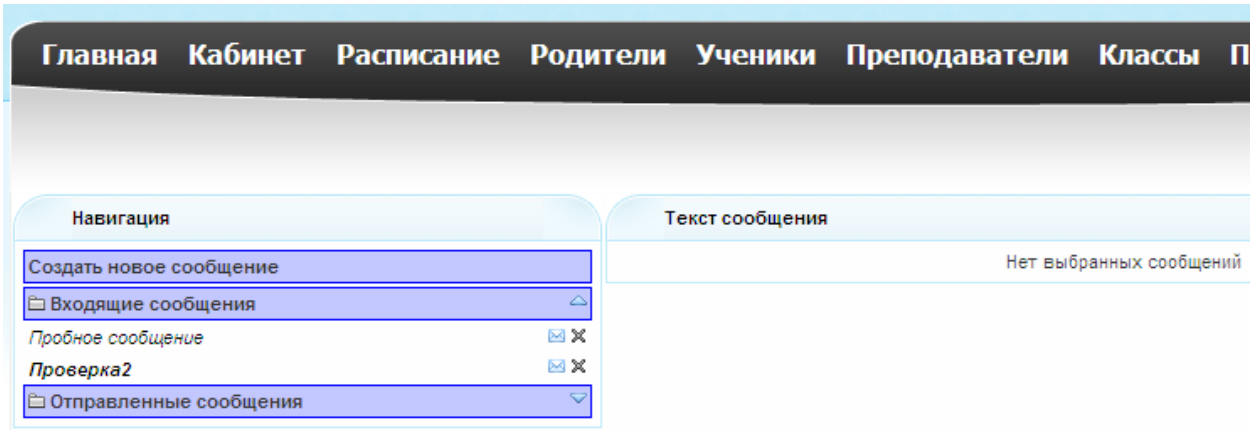


Рисунок 3.18 – Сторінка управління повідомленнями

Сторінка «Мой кабинет» доступна лише вчителеві і містить інформацію по предметах, які веде вчитель в межах періоду навчання, класів, в яких є ці предмети, а так само учнів цих класів. Списки уроків, класів і учнів представлені у вигляді дерева. На рисунку 3.19 показаний блок дерева уроків.

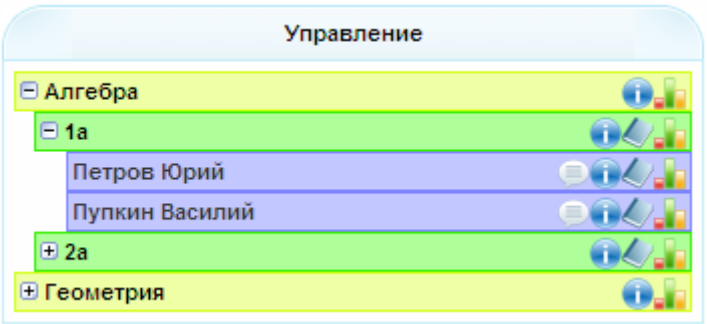


Рисунок 3.19 – Управління уроками

На рисунку жовтим кольором виділені уроки, зеленим класи, синім учні. При цьому в правій частині рядка показані можливі дії. Так для учня це «повідомлення, інформація, редактор щоденника, статистика». Для економії місця елементи за умовчанням згорнуті. Розкриття відбувається при натисненні на будь-яку частину блоку елементу. При розкритті блоку класу відбувається прихований запит до сервера. В результаті запиту приходить список учнів класу. Зроблено це для того, щоб зменшити розмір сторінки. Якщо дерево уроків зробити статичним (всі елементи завантажуються відразу), то розмір сторінки складатиме декілька мегабайт. Крім того, такий підхід дозволить зменшити навантаження на базу даних за рахунок зменшення складності запиту.

Сторінка «Мой класс» доступна учневі і надає інформацію про останні події в класі, оголошення для класу, розклад і відображує список учнів класу. Ця ж сторінка доступна і останнім користувачам системи, але в цьому випадку в рядку запиту має бути переданий ідентифікатор класу, а користувач, що запитав дану сторінку, повинен мати права на перегляд класів. Якщо користувач не має достатніх прав для перегляду, відбувається перенаправлення на головну сторінку. На рисунку 3.20 представлена частина сторінки, що відображує список учнів.

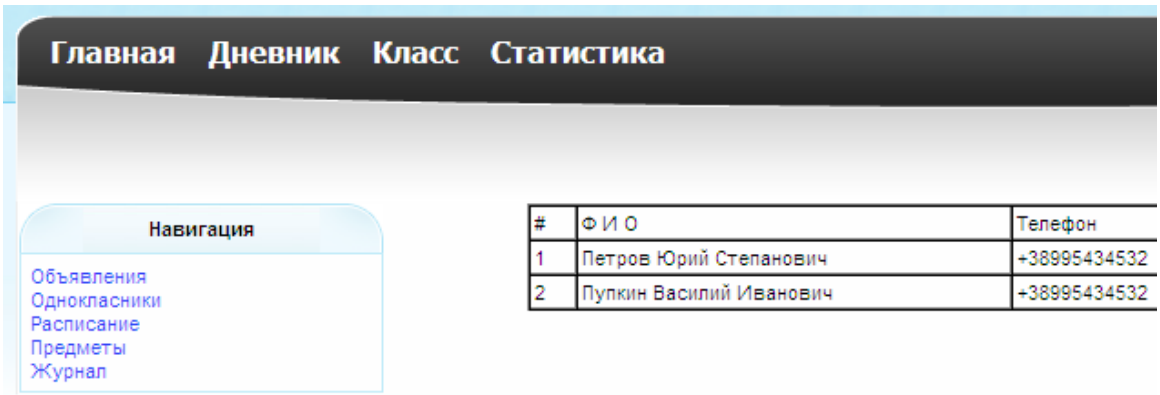


Рисунок 3.20 – Меню класу і список однокласників

У списку однокласників можуть відображуватися додаткові поля, такі як домашня адреса або контактний телефон. З цієї ж сторінки можна відправити приватне повідомлення.

Звіти - це набір скриптів, що мають схожу назву report_{тип звіту}.php і сторінок, що виконують створення та містять інформацію про відвідуваність, успішність учня, або для класу - це предмети з кількістю годин на поточний період навчання. Скрипт report_main надає можливість вибрати один з можливих варіантів звітності (рис. 3.21).

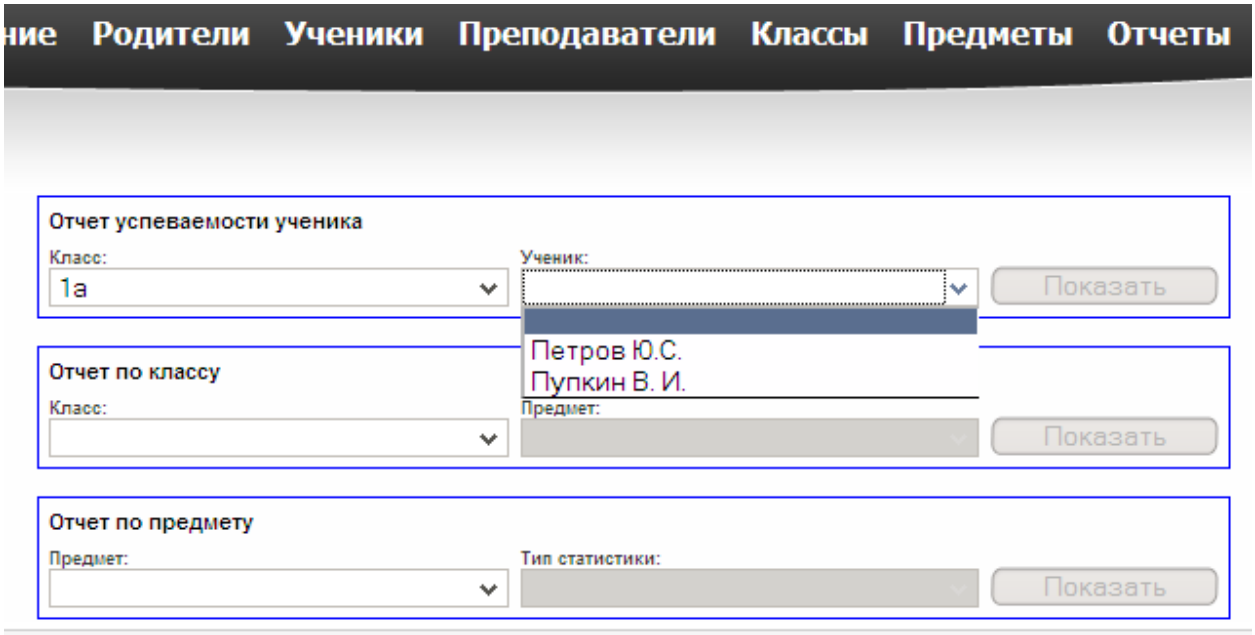


Рисунок 3.21 – Сторінка вибору типу звіту

Для створення звітів використовуються дані з таблиць Розклади і Щоденник. Для генерації звітів беруться всі виставлені оцінки і на їх основі заповнюється шаблон звіту успішності.

3.4.6. Карта сайту

Карта сайту – це повний каталог всіх розділів сайту, з коротким описом кожного розділу. Карта сайту надає модель структури сайту. За допомогою карти сайту можна значно полегшити працю відвідувача по дослідженню складових модулів Інтернет-проекта.

Для батьків

- Главная
    - Мои дети
        - Просмотр дневника
        - Просмотр статистики
        - Обзор класса
            - Расписание
            - Ученики
            - Объявления
    - Сообщения

Для учнів

- Главная
    - Дневник
    - Класс
        - Одноклассники
        - Расписание
        - Преподаватели
        - Объявления
    - Отчеты
        - Успеваемость по предмету
        - Успеваемость класса
    - Сообщения

Для викладачів

- Главная
    - Мой кабинет
        - Редагування щоденників
        - Классный журнал
        - Просмотр класса
            - Ученики
            - Расписание
            - Объявления
        - Мое расписание
        - Мои предметы
    - Редактор расписания
        - Редактор периодов обучения
    - Редактор пользователей
        - Родители
        - Ученики
        - Преподаватели
    - Редактор классов
    - Редактор предметов
    - Отчеты
        - По ученику
        - По классу
        - По предмету
    - Сообщения

3.5. Посібник користувача системи

Інформаційно-довідковий веб-сервер надає можливість обміну інформацією між контенгентом учбового закладу і батьками учнів. Доступ до інформаційної системи здійснюється при допомозі Інтернет браузера. Нижче приведений список підтримуваних браузерів.

1) InternetExplorer версії 6 і вище;
2) Opera версії 7 і вище;
3) Mozila FireFox версії 2 і вище;
4) Safari;
5) Google Chrome.

В цілому доступ до системи можливий з будь-якого браузера, що підтримує технологію Javascript. Дозвіл екрану, що рекомендується, 1024x768 і вище.

Інформаційно-довідковий веб-сервер є закритою системою. При цьому доступ до даних системи можливий лише після ідентифікації користувача. Для ідентифікації користувача використовуються унікальний логін і пароль. Для того, щоб увійти до системи, користувачеві необхідно вести свої логін і пароль у форму, показану на рисунку 3.22.

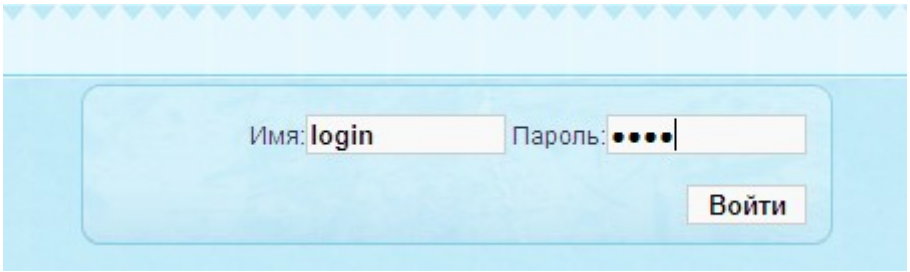


Рисунок 3.22 – Форма перевірки авторизації

При цьому користувачеві варто пам'ятати, що він має п'ять спроб введення правильних даних, після чого доступ до системи буде тимчасово обмежений. Після того, як вхід в систему здійснений, користувач потрапляє на головну сторінку користувача. При цьому форма входу вже не міститиме полів для

введення логіна і пароля. У формі відображуватиме ім'я поточного користувача і кнопка «Выход». Слід завжди виходити з системи після завершення роботи. Такі дії не дозволять нікому працювати від чужого імені і збільшать безпеку системи.

Так само після входу розширюється меню користувача, так для викладача вид меню показаний на рисунку 3.23.

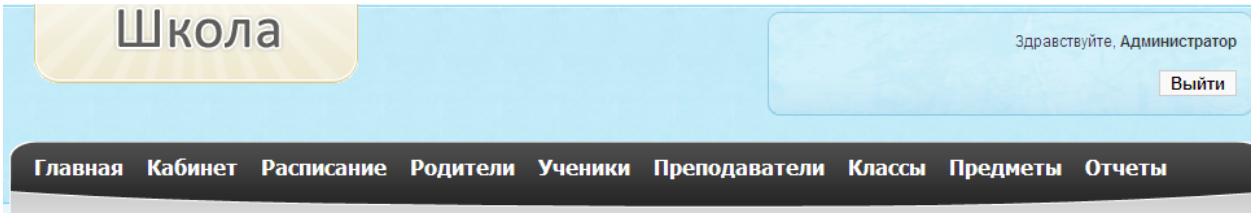


Рисунок 3.23 – Меню навігації для вчителя

Для кожного типа користувачів меню навігації виглядатиме по-різному.

Спільною для всіх типів користувачів є сторінка управління повідомленнями представлена на рисунку 3.24.

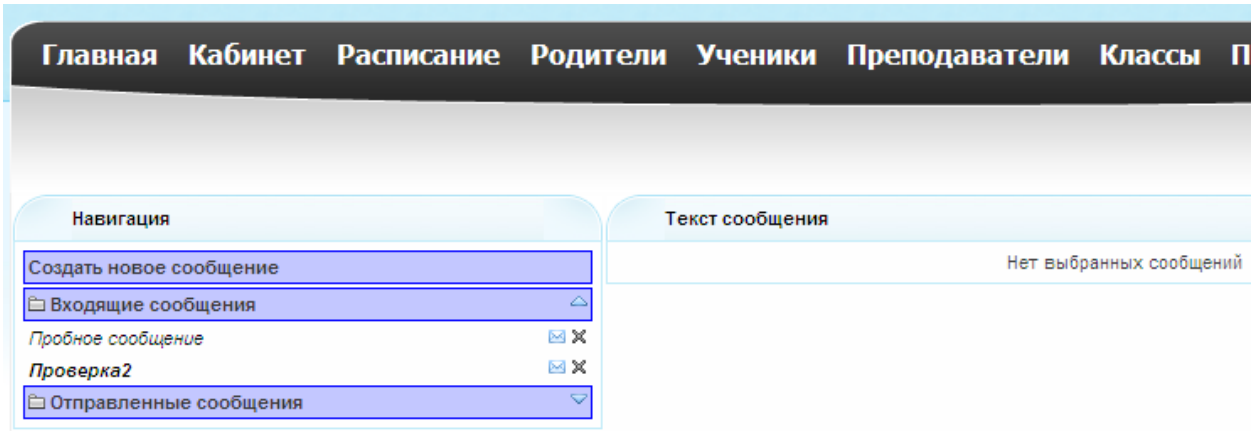


Рисунок 3.24 – Сторінка управління внутрішньою поштою

На рисунку 3.24 в лівій частині розташований блок, що містить кнопку створення нового повідомлення, а так само два списки, що розкриваються. При розкритті списків відображуються вхідні або вихідні повідомлення. Заголовки непрочитанних повідомлень виділені жирним шрифтом, і вони поміщаються вверх списку. Праворуч від заголовка повідомлення відображуються кнопки дій.

Для вхідних повідомлень це кнопки відповіді і видалення. Для вихідних повідомлень це прапор прочитання і видалення. Вибравши будь-яке з повідомлень, текст повідомлення буде завантажений в правий блок сторінки. При цьому у верхній частині відображується докладна інформація про повідомлення і кнопки «Ответ», «Удаление», «Переслать». При натисненні на кнопки «Ответ», «Создать новое» відкривається вікно редактора повідомлення. У верхній частині розташовується рядок пошуку користувача, або ж якщо натискувана кнопка відповідь, то там вже вписано ім'я користувача, якому вирушає дане повідомлення. Нижче розташовується поле введення заголовка повідомлення і поле введення тексту повідомлення. При цьому заголовок і текст не можуть бути порожніми. Якщо користувач намагається відправити порожнє повідомлення, то буде виведена підказка з описом помилки.

3.5.1. Сторінки викладачів

Після входу в систему, вчитель дістає можливість виконувати дії над розкладом, користувачами, класами, і ін. У верхній частині сторінки з'являються посилання на доступні сторінки: Кабінет; Розклад; Батьки; Учні; Викладачі; Класи; Предмети; Звіти.

Кожне із посилань веде на свій розділ сайту. Нижче описані основні сторінки вчителя.

Кабінет. Ця сторінка надає можливість управління уроками, класами і учнями, а так само дозволяє стежити за останніми шкільними подіями, що мають пряме або побічне відношення до користувача. У лівій частині розташовано дерево уроків. При клацанні мишки по порожньому місцю рядка уроку відбувається розкриття списку класів, в яких на поточному проміжку навчання є хоч би один такий предмет.

Натиснення на конки  дозволяють перейти на сторінки статистики предмету і інформації по предмету. На сторінках статистики предмету є

можливість отримати дані по кількості годин у вибраного класу. А так само згенерувати версію для друку.

У розкритому списку класів напроти кожного класу так само є ікони, що дозволяють переходити на інші сторінки. Так натиснення на іконі приведе до переходу до сторінки журналу успішності класу. Така ж ікона напроти прізвища і імені учня веде до редактора щоденника.

У розкритому списку учнів класу так само присутня ікона. Натиснення на дану ікону відкриє вікно створення персонального повідомлення. У верхній частині сторінки так само розташовуються посилання, що ведуть на сторінки відображення розкладу вчителя, що виводять список предметів і сторінки звітів.

У розкладі вчителя відображуються лише предмети, на які призначений даний вчитель. Дана сторінка не має жодних засобів редагування і створена виключно для інформування про розпорядок дня вчителя. При цьому в рядках розкладу вказується предмет, кабінет, назва класу і домашнє завдання. У колонці дій знаходяться посилання на інформацію по класу і предмету.

Розклад. Даний пункт меню видно лише у тих викладачів, які мають права редагування розкладу. Спільний вид сторінки редагування розкладу представлений на рисунку 3.16.

У блоці «Навигация» можливий вибір класу, для якого редагується розклад і періоду навчання. Так само в цьому блоці розташовується посилання, яке веде до редактора періодів навчання.

У випадку, якщо на даний момент немає створених періодів навчання або класів, то в лівій частині сторінки відображується підказка про те, що необхідно створити новий період або додати класи. У колонці «Действия» показані можливі дії для запису в розкладі. Так для порожнього запису це буде «Создать», а для вже заповненого «Изменить». Посилання «Изменить» відкриє вікно редагування запису розкладу представлене на рисунку 3.25.

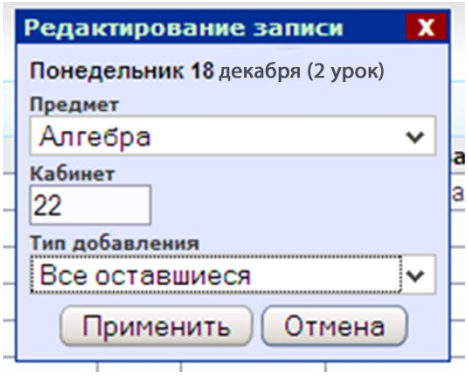


Рисунок 3.25 – Редагування запису розкладу

У вікні редагування вчитель може вибрати предмет, призначити кабінет і встановити типи додавання запису. Є два можливі типи додавання: «Одна запись» і «Все оставшиеся». Тип додавання «Все оставшиеся» встановить запис не лише на редагований тиждень, але і на всі тижні до кінця вибраного періоду навчання. При видаленні запису так само буде запропонований вибір «Одна запись» і «Все оставшиеся».

На кожен день в розкладі може бути доданий коментар класу. На рисунку 3.26 показано вікно створення коментаря на день.

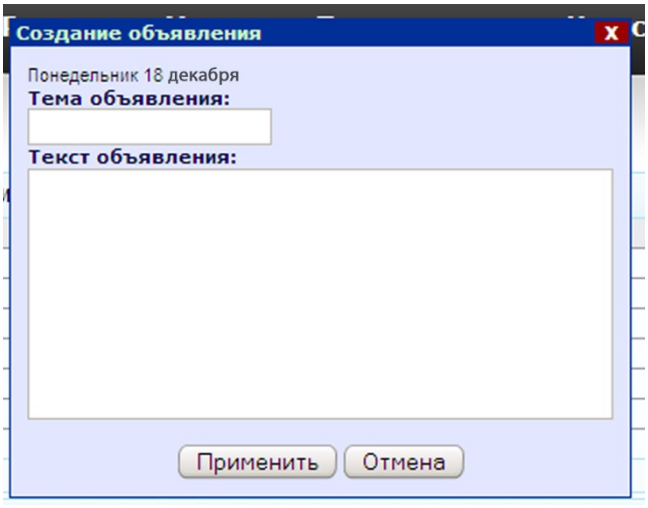


Рисунок 3.26 – Створення коментаря на день

При додаванні коментаря обов'язково мають бути заповнені тема і текст оголошення. В разі виникнення помилки буде показана підказка у верхній частині

вікна. При успішному додаванні вікно зникне, а всі користувачі, переглядаючі щоденники/розклад вже зможуть побачити створене оголошення.

Батьки, Учні, Викладачі. Дані сторінки надають можливість додавання або зміни інформації про учнів, викладачів і батьків. Доступ до даних сторінок може бути обмежений залежно від прав викладача. У верхній частині таблиці користувачів розташовані кнопка додавання користувача, форма пошуку користувача. Так само на сторінці редагування учнів є можливість сортування користувачів залежно від класу. За замовчуванням відображується десять користувачів на одній сторінці. Якщо кількість користувачів перевищує це число, то під таблицею з'являється список сторінок. На рисунку 3.27 представлена сторінка редагування учнів.

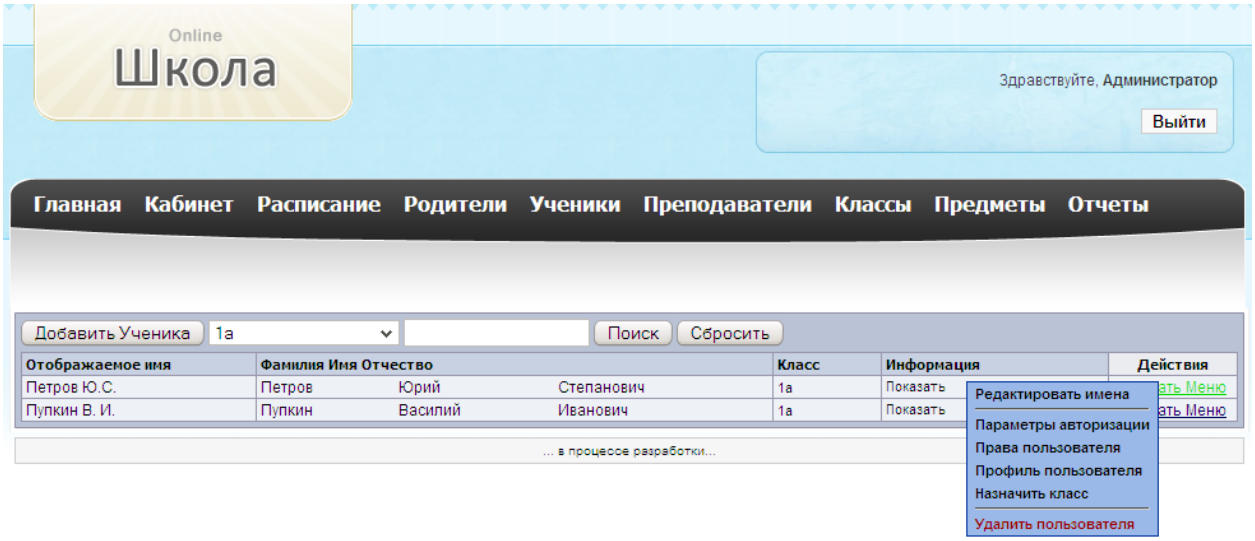


Рисунок 3.27 – Сторінка редагування учнів

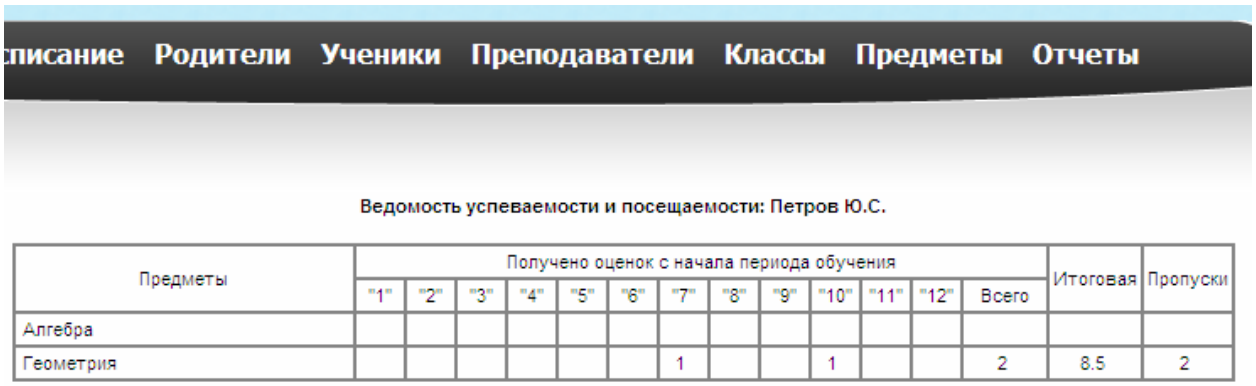
Посилання «Показать Меню» виводить список можливих дій. Так вибравши «Параметры авторизации» буде показане вікно призначення користувачеві логіна і пароля для входу в систему. «Профиль пользователя» відображує вікно введення додаткових даних, таких як дата народження, адреса мешкання, контактний телефон та інші. «Назначить класс» виводить вікно вибору класу для користувача. Колонка «Информация» дає можливість перегляду всієї доступної інформації (Наявність логіна і пароля, дані профілю та інші).

Сторінки редагування викладачів і батьків мають схожу структуру і відрізняються лише списком можливих дій в меню.

Предмети. Класи. Редактор предметів дозволяє створювати і редагувати предмети. При редагуванні призначається назва і список викладачів (до трьох чоловік), які дістануть можливість редагувати домашнє завдання для цих предметів і виставляти оцінки.

Редактор класів дозволяє створювати нові класи, а так само змінювати властивості вже існуючих класів. Як додаткові параметри при редагуванні використовується рік початку навчання, класний керівник і опис. Окрім цього з даної сторінки можливий перехід на сторінку перегляду інформації класу.

Звіти. Сторінка звітів дозволяє викладачеві бути в курсі успішності по конкретному учневі або класу. Існує три типи звітів: «Статистика класса» «Статистика ученика» і «Статистика предмета». На рисунку 3.28 представлений звіт по успішності учня.



Предметы	Получено оценок с начала периода обучения													Итоговая	Пропуски
	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	"9"	"10"	"11"	"12"	Всего		
Алгебра															
Геометрия							1			1			2	8.5	2

Рисунок 3.28 – Звіт по успішності учня

У лівій частині перераховані предмети, по яких йде навчання. Колонки таблиці - це оцінки, отримані за даний період навчання і середній бал по предмету. Після таблиці вказується кількість пропусків предметів. Даний звіт може згенерувати як на весь поточний період навчання, так і на вибраний інтервал часу.

До додаткових сторінок викладача відносяться: класний журнал і редактор щоденника.

Редактор щоденника. Перехід на дану сторінку можливий із сторінки «Кабинет». Вид сторінки редагування щоденника користувача представлений на рисунку 3.29.

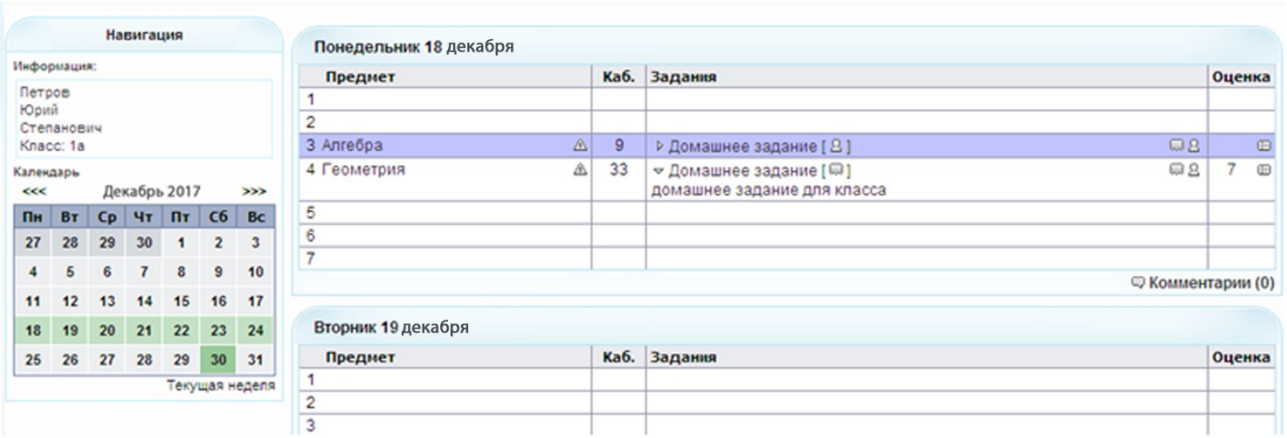


Рисунок 3.29 – Редактор щоденника

Дана сторінка надає можливість виставляння оцінок (колонка «Оценка»), призначення домашнього завдання, і маркіровки уроку як пропущеного. Окрім цього на кожен день можливе додавання коментаря. Проте доданий коментар, у порівнянні зі сторінкою редагування розкладу, буде доступний лише учневі, чий щоденник редагується. Маркіровка уроку як пропущеного робить рядок уроку виділеним сірим кольором. Так само при установці уроку як пропущеного можливо ввести причину відсутності.

Виставляння оцінки здійснюється натисненням на кнопку . При цьому відображується вікно, показане на рисунку 3.30.

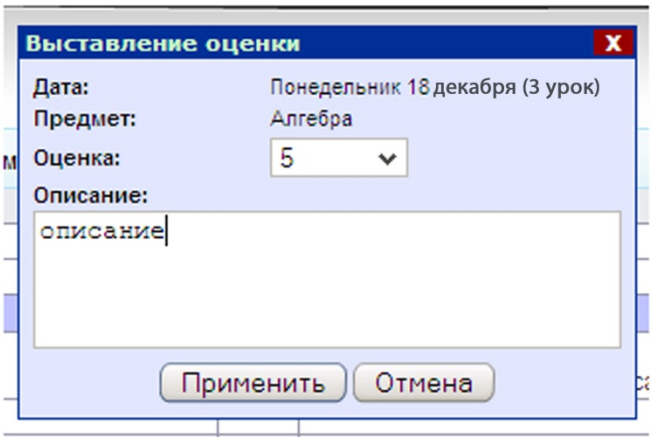


Рисунок 3.30 – Вікно виставлення оцінки в щоденник

Вчитель має можливість не лише виставити оцінку, але і написати опис. Слід зазначити, що виставляння оцінок в більшості випадків доступно лише на ті предмети, на які призначений викладач. Проте є можливість виставити викладачеві права на виставляння оцінок по всіх предметах.

3.5.2. Сторінки учня

У інформаційно-довідковій системі учень дістає можливість отримувати поточний розклад і переглядати електронну версію свого щоденника. Окрім цього при вході в систему він отримує повідомлення про нові події (наприклад, виставляння оцінки, зміни в розкладі, новому оголошенні для класу).

Для учня передбачено три основні компоненти меню: Щоденник; Клас; Статистика. Сторінка щоденника представлена на рисунку 3.31.

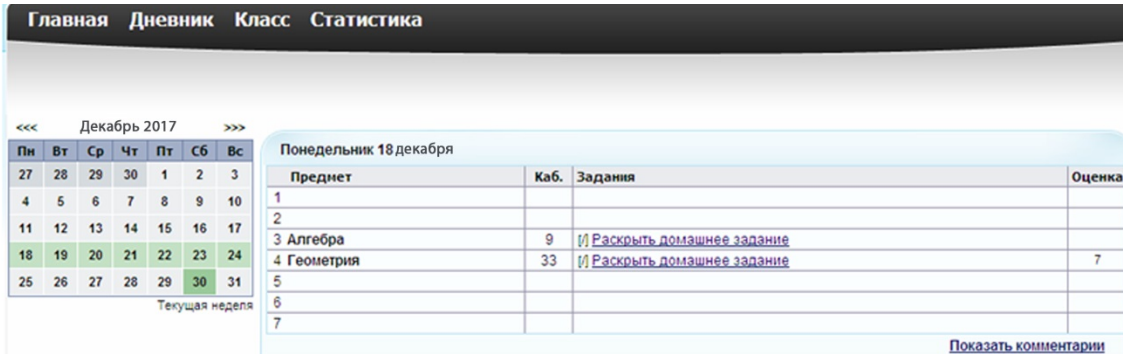


Рисунок 3.31 – Сторінка перегляду щоденника

На даній сторінці учень може проглянути свої поточні оцінки, домашнє завдання, а також розклад з вказівкою вчителів і кабінетів.

Так само під таблицею учбового дня розташовується блок з коментарями, розкривши цей блок учень отримує список приміток або планованих заходів на цей день.

«Класс» надає можливість бути в курсі останніх планованих заходів класу, а так само з допомогою підміню «одноклассники» мати можливість спілкування з однокласниками і отримувати додаткову інформацію, наприклад номери телефонів.

Із сторінки «Класс» є можливість переходу на сторінку списку предметів. На даній сторінці учневі відображується список предметів на поточний період навчання і імена викладачів призначених на ці предмети.

На сторінці статистики учень має можливість оцінити поточну успішність і проглянути середній бал, який виходить на даний момент.

3.5.3. Сторінки батьків

Батьки мають всього один пункт меню – Мої діти. На даній сторінці показані учні, які були призначені даному користувачеві. При цьому на головній сторінці показуються останні дії, що відносяться до призначених учнів.

Із сторінки «Мои дети» батько може перейти до перегляду щоденника вибраного учня, а так само класу. Окрім цього, батьки дістає можливість автоматично згенерувати звіт про успішність вибраного учня.

Приклад сторінки звіту успішності представлений на рисунку 3.32.

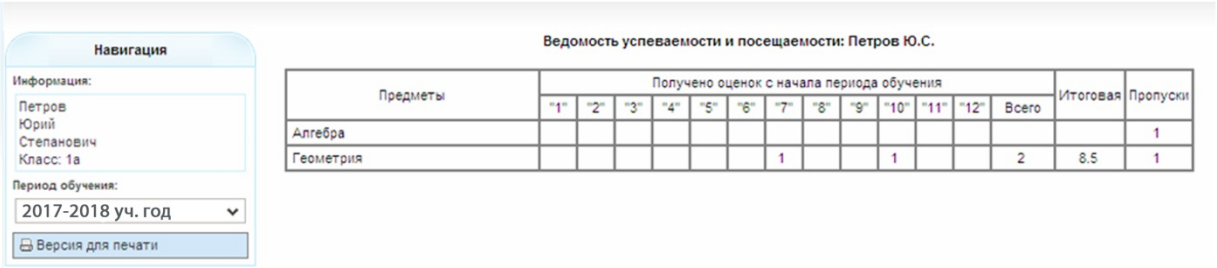


Рисунок 3.32 – Звіт про успішність учня

У даному звіті відображують оцінки і їх кількість, а так само середню оцінку на даний момент. Так само окремим пунктом відображується кількість пропусків даного предмету.

Для батьків так само передбачена система персональних повідомлень. Відповідно викладачі можуть відправити повідомлення батькам. Кількість нових вхідних повідомлень показана у верхній частині сторінки у вигляді рядка Повідомлення(x), де x – це число нових непрочитанних повідомлень.

Перегляд даних класу може бути обмежений керівництвом школи. При цьому можуть бути приховані персональні дані.

3.5.4. Вимоги до програмного і апаратного забезпечення користувача системи

Для коректної роботи і відображення сайту інформаційно-довідкової системи комп'ютер користувача повинен відповідати наступним параметрам апаратного і програмного забезпечення:

- частота процесора від 800 мгц і вище;
- оперативна пам'ять від 256 мбайт і вище;
- монітор з дозволом екрану не нижче 1024x768;
- один з перерахованих нижче інтернет-браузеров з вбудованою підтримкою javascript і cookies: MS Internet Explorer 6.0 і вище; Mozilla Firefox 2.0 і вище; Opera 7 і вище; Safari 2.0.0 і вище.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

В даному розділі проведено аналіз потенційних небезпечних та шкідливих виробничих факторів, причин пожеж. Розглянуті заходи, які дозволяють забезпечити гігієну праці і виробничу санітарію. На підставі аналізу, розроблені заходи з техніки безпеки та рекомендації з пожежної профілактики.

Завданням даної роботи бакалавра була розробка програмного забезпечення інформаційно-довідкової системи для середньої школи. Так як в процесі проектування використовувалося комп’ютерне обладнання, то аналіз потенційно небезпечних і шкідливих виробничих чинників виконується для персонального комп'ютера, на якому буде виконуватися розробка.

4.1 Загальні питання з охорони праці

Умови праці на робочому місці, безпека технологічних процесів, механізмів, устаткування та інших засобів, що використовуються працівником, а також санітарно-побутові умови повинні відповідати вимогам нормативних актів про охорону праці. В законі України «Про охорону праці» визначається, що охорона праці - це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності.

В організації/підприємстві проводиться навчання і перевірка знань з питань охорони праці. Також впроваджені організаційні заходи з пожежної безпеки - навчання і перевірку знань відповідно до вимог Типового положення про інструктажі, спеціальне навчання та перевірку знань з питань пожежної безпеки на підприємствах, в установах та організаціях України.

4.2 Аналіз стану умов праці

Робота над створенням інформаційно-довідкової системи для середньої школи проходитиме в приміщенні багатоквартирного будинку. Для даної роботи достатньо однієї людини, для якої надано робоче місце зі стаціонарним комп'ютером. Геометричні розміри приміщення зазначені в табл. 4.1.

Таблиця 4.1 – Розміри приміщення

Найменування	Значення
Довжина, м	5
Ширина, м	5
Висота, м	3
Площа, $м^2$	25
Об'єм, $м^3$	75

Згідно з [14] розмір площі для одного робочого місця оператора персонального комп'ютера має бути не менше 6 кв. м, а об'єм — не менше 20 куб. м. Отже, дане приміщення цілком відповідає зазначеним нормам.

При порівнянні відповідності характеристик робочого місця нормативним основні вимоги до організації робочого місця за [12] (табл. 4.2) і відповідними фактичними значеннями для робочого місця, констатуємо повну відповідність.

Таблиця 4.2 - Характеристики робочого місця

Найменування параметра	Фактичне значення	Нормативне значення
Висота робочої поверхні, мм	700	680 ÷800
Висота простору для ніг, мм	700	не менше 600
Ширина простору для ніг, мм	600	не менше 500
Глибина простору для ніг, мм	720	не менше 650
Висота поверхні сидіння, мм	450	400 ÷500
Ширина сидіння, мм	420	не менше 400
Глибина сидіння, мм	420	не менше 400

Продовження таблиці 4.2 - Характеристики робочого місця

Найменування параметра	Фактичне значення	Нормативне значення
Висота поверхні спинки, мм	500	не менше 300
Ширина опорної поверхні спинки, мм	450	не менше 380
Радіус кривини спинки в горизонтальній площині, мм	400	400
Відстань від очей до екрану дисплея, мм	750	700 ÷800

Приміщення кабінету знаходиться на третьому поверсі чотирьох поверхової будівлі, має об'єм 75 м^3, площу 25 м^2. Температура протягом року коливається у межах 18-25°С, відносна вологість - близько 50%. Швидкість руху повітря не перевищує 0,2 м/с. Шум у приміщенні знаходиться на рівні 50 дБА. Система вентилювання - природна неорганізована, а опалення - централізоване.

За фізичним навантаженням виконання випускної роботи бакалавра відноситься до категорії легкі роботи (Іа), її виконують сидячи з періодичним ходінням, але щодо характеру організування, підпадає під нав'язаний режим, оскільки певні розділи роботи необхідно виконати у встановлені терміни.

Рекомендовано застосування екранних фільтрів, локальних світлофільтрів (засобів індивідуального захисту очей) та інших засобів.

Роботу над дипломним проектом визнано такою, що займає 50% часу робочого дня та при восьмигодинній робочій зміні рекомендовано встановити додаткові регламентовані перерви - для розробників програм тривалістю 15 хв. через кожну годину роботи.

4.3 Виробнича санітарія

На підставі аналізу небезпечних та шкідливих факторів при виробництві (експлуатації), пожежної безпеки можуть бути надалі вирішені питання необхідності забезпечення працюючих достатньою кількістю освітлення,

вентиляції повітря, організації заземлення, тощо.

Аналіз небезпечних та шкідливих виробничих факторів виконується у табличній формі (табл. 4.3). Основними робочими характеристиками персонального комп'ютера є: робоча напруга U=+220В +-5%, робочий струм I=2А, споживана потужність Р=350 Вт.

Таблиця 4.3 – Аналіз небезпечних і шкідливих виробничих факторів

Небезпечні і шкідливі виробничі фактори	Джерела факторів (види робіт)	Кількісна оцінка	Нормативні документи
1	2	3	4
фізичні			
- підвищена температура поверхонь обладнання	експлуатація ЕОМ, принтерів, сканерів чи/або серверного обладнання для роботи	2	[14]
- підвищений рівень електромагнітного випромінення	-//-	2	[24]
- підвищений рівень напруги електричної мережі, замикання якої може відбутися через тіло людини	-//-	4	[25] [14]
- підвищений рівень статичної електрики	-//-	2	[25]
- підвищена напруженість електричного поля	-//-	2	[24]
- підвищена напруженість магнітного поля	-//-	2	[24]
- підвищена яскравість світла	порушення умов праці (організації місця праці-налагодження моніторів)	1	[12]
- знижена контрастність	-//-	1	[12]
психофізіологічні:			
- нервово-психічна перевантаження (розумове, перенапруження аналізаторів-зорових)	- пошук інформації для постановки теми; - пошук та аналіз аналогів і літератури; - пошук наявних технологій, моделювання, аналіз алгоритмів; - виконання роботи за темою диплома, тестування; - оформлення роботи	4	[26] [12]
- фізичні (статичне – сидіння)	порушення умов праці та організації робочого часу - безперервна робота	2	[26] [12]

Небезпека розвитку пожежі на обчислювальному центрі обумовлюється застосуванням розгалужених систем електроживлення ЕОМ, вентиляції і кондиціювання. Запобігти утворенню горючого середовища (замінити горючі речовини і матеріали на негорючі і важкогорючі) не надається технічно можливим. Тому проектом передбачаються способи і засоби запобігання утворення (або внесення) в горюче середовище джерел запалювання.

На робочому місці виконуються наступні вимоги електробезпеки: ПК, периферійні пристрої та устаткування для обслуговування, електропроводи і кабелі за виконанням та ступенем захисту відповідають класу зони за ПУЕ (правила улаштування електроустановок), мають апаратуру захисту від струму короткого замикання та інших аварійних режимів. Лінія електромережі для живлення ПК, периферійних пристроїв і устаткування для обслуговування, виконана як окрема групова три-провідна мережа, шляхом прокладання фазового, нульового робочого та нульового захисного провідників. Нульовий захисний провідник використовується для заземлення електроприймачів.

4.4 Гігієнічні вимоги до параметрів виробничого середовища

4.4.1 Мікроклімат

Мікроклімат робочих приміщень – це клімат внутрішнього середовища цих приміщень, що визначається діючої на організм людини температури, вологості та швидкості переміщення повітря. Оптимальні значення для температури, відносної вологості й рухливості повітря для зазначеного робочого місця відповідають [13] і наведені в табл. 4.4:

Таблиця 4.4 – Норми мікроклімату робочої зони об’єкту

Період року	Категорія робіт	Температура C^0	Відносна вологість %	Швидкість руху повітря, м/с
Холодна	легка-1 а	22 - 24	40 – 60	0,1
Тепла	легка-1 а	23 - 25	40 – 60	0,1

Дане приміщення обладнане системами опалення, кондиціювання повітря або припливно-витяжною вентиляцією. У приміщенні на робочому місці забезпечуються оптимальні значення параметрів мікроклімату: температури, відносної вологості й рухливості повітря у відповідності до [13]. Рівні позитивних і негативних іонів у повітрі мають відповідати [13].

4.4.2 Освітлення

Світло є природною умовою існування людини. Воно впливає на стан вищих психічних функцій і фізіологічні процеси в організмі. Хороше освітлення діє тонізуюче, створює гарний настрій, покращує протікання основних процесів вищої нервової діяльності.

У приміщенні, де розташовані ЕОМ передбачається природне бічне освітлення, рівень якого відповідає [11]. Джерелом природного освітлення є сонячне світло. Регулярно повинен проводитися контроль освітленості, який підтверджує, що рівень освітленості задовольняє ДБН і для даного приміщення в світлий час доби достатньо природного освітлення.

Розрахунок освітлення.

Для виробничих та адміністративних приміщень світловий коефіцієнт приймається не менше -1/8, в побутових – 1/10:

$$S_b = \left(\frac{1}{5} \div \frac{1}{10} \right) \cdot S_n, \tag{4.1}$$

де S_b – площа віконних прорізів, м^2;

S_n – площа підлоги, м^2.

$S_n = a \cdot b = 5 \cdot 5 = 25$ м^2,

$S = 1/8 \cdot 25 = 3{,}125$ м^2.

Приймаємо 2 вікна площею S=1,6 м^2 кожне.

Розрахунок кількості світильників n виробляється по формулі (4.2):

$$n = \frac{E \cdot S \cdot Z \cdot K}{F \cdot U \cdot M}, \tag{4.2}$$

де E – нормована освітленість робочої поверхні, визначається нормами – 300 лк;

S – освітлювана площа, м^2; S = 25 м^2;

Z – поправочний коефіцієнт світильника (Z = 1,15 для ламп розжарювання та ДРЛ; Z = 1,1 для люмінесцентних ламп) приймаємо 1,1;

K – коефіцієнт запасу, що враховує зниження освітленості в процесі експлуатації – 1,5;

U – коефіцієнт використання, залежний від типу світильника, показника індексу приміщення і т.п. – 0,575

M – число люмінесцентних ламп в світильнику – 2;

F – світловий потік лампи – 5400лм (для ЛБ-80).

Підставивши числові значення у формулу (4.2), отримуємо:

$$n = \frac{300 \cdot 25 \cdot 1{,}1 \cdot 1{,}5}{5400 \cdot 0{,}575 \cdot 2} \approx 2{,}0$$

Приймаємо освітлювальну установку, яка складається з 2-х світильників, які складаються з двох люмінесцентних ламп загальною потужністю 160 Вт, напругою – 220 В.

4.4.3 Шум та вібрація, вентилювання

Рівень шуму, що супроводжує роботу користувачів персональних комп’ютерів (зумовлений як роботою системних блоків, клавіатури, так і друкуванням на принтерах, а також зовнішніми чинниками), коливається у межах

50–65 дБА [13]. У залах опрацювання інформації та комп’ютерного набору рівні шуму не повинні перевищувати 65 дБА. Віброізоляцію можливо здійснювати за допомогою спеціальної прокладки під системний блок, який послаблює передачу вібрацій робочого столу. Вібрація на робочому місці в приміщенні, що розглядається, відповідає нормам [13].

У приміщенні, де знаходяться ЕОМ, повітрообмін реалізується за допомогою природної організованої вентиляції (вентиляційні шахти), тобто при V приміщення > 40 м3 на одного працюючого допускається природна вентиляція. Також має здійснюватися провітрювання приміщення, в залежності від погодних умов, тривалістю не менше 10 хв.

4.5 Заходи з організації виробничого середовища та попередження виникнення надзвичайних ситуацій

При раптовому припиненні подачі електричної енергії вимкнути всі пристрої ПК. Витягнути вилки з розеток. При наявності ознак горіння (дим, запах горілого) необхідно вимкнути всі пристрої ПК, знайти місце загоряння і виконати всі можливі заходи для його ліквідації, попередивши терміново про це керівництво.

При замиканні, перевантаженні електричного струму на електричному обладнанні, внаслідок ураження грозової блискавки та ймовірної небезпеки ураженням електричним струмом, попередження замикання здійснюється правильним вибором, монтажем експлуатації мереж, а також застосування захисту схем у вигляді швидкодіючих реле та вимикачів, плавких запобіжників.

Застосовують різні електричні захисні засоби від ураження струмом: ізолюючі, основні, запобіжні.

Розрахунок захисного заземлення (забезпечення електробезпеки будівлі)

Згідно з класифікацією приміщень за ступенем небезпеки ураження електричним струмом [19], приміщення в якому проводяться всі роботи відноситься до першого класу (без підвищеної небезпеки). Коефіцієнт використання вертикальних заземлювачів $\eta_в$ в залежності від їх розміщення та кількості знаходиться в межах 0,4…0,99. Взаємну екрануючу дію горизонтального заземлювача (з'єднувальної смуги) враховують за допомогою коефіцієнта використання горизонтального заземлювача η_c.

Послідовність розрахунку.

1) Визначається необхідний опір штучних заземлювачів $R_{шт.з.}$:

$$R_{шт.з.} = \frac{R_д \cdot R_{пр.з.}}{R_{пр.з.} - R_д}, \qquad (4.3)$$

де $R_{пр.з.}$ – опір природних заземлювачів; $R_д$ – допустимий опір заземлення. Якщо природні заземлювачі відсутні, то $R_{шт.з.}=R_д$.

Підставивши числові значення у формулу (4.3), отримуємо:

$$R_{шт.з.} = \frac{4 \cdot 40}{40 - 4} \approx 4 \text{ Ом}$$

2) Опір заземлення залежить від питомого опору ґрунту ρ, Ом·м. Значення питомого опору приймаємо ρ=40 Ом•м (табличне значення).

3) Розрахунковий питомий опір ґрунту, $\rho_{розр}$, Ом·м, визначається відповідно для вертикальних і горизонтальних і визначається за формулою:

$$\rho_{розр..} = \psi \cdot \rho, \qquad (4.4)$$

де ψ – коефіцієнт сезонності для вертикальних заземлювачів і кліматичної зони з нормальною вологістю землі, приймається для вертикальних заземлювачів $\rho_{розр.в}$=1,7 і горизонтальних $\rho_{розр.г}$=5,5 Ом·м.

$$\rho_{розр.в} = 1,7 \cdot 40 = 68 \text{ Ом•м;}$$

$$\rho_{розр.г} = 5,5 \cdot 40 = 220 \text{ Ом•м}$$

4) Розраховується опір розтікання струму вертикального заземлювача $R_в$, Ом, за формулою (4.5).

$$R_в = \frac{\rho_{розр.в}}{2 \cdot \pi \cdot l_в} \cdot \left(\ln \frac{2 \cdot l_в}{d_{ст}} + \frac{1}{2} \cdot \ln \frac{4 \cdot t + l_в}{4 \cdot t - l_в} \right), \quad (4.5)$$

де $l_в$ – довжина вертикального заземлювача (для труб - 2−3 м; $l_в$=3 м);

$d_{ст}$ – діаметр стержня (для труб - 0,03−0,05 м; $d_{ст}$=0,05 м);

t – відстань від поверхні землі до середини заземлювача, яка визначається за формулою (4.6):

$$t = h_в + \frac{l_в}{2}, \quad (4.6)$$

де $h_в$ – глибина закладання вертикальних заземлювачів (0,8 м); тоді

$$t = 0,8 + \frac{3}{2} = 2,3 \text{ м}$$

$$R_в = \frac{68}{2 \cdot \pi \cdot 3} \cdot \left(\ln \frac{2 \cdot 3}{0,05} + \frac{1}{2} \cdot \ln \frac{4 \cdot 2,3 + 3}{4 \cdot 2,3 - 3} \right) = 18,5 \text{ Ом}$$

5) Визначається теоретична кількість вертикальних заземлювачів n штук, без урахування коефіцієнта використання $\eta_в$:

$$n = \frac{2 \cdot R_в}{R_д} = \frac{2 \cdot 18,5}{4} = 9,25 \quad (4.7)$$

І визначається коефіцієнт використання вертикальних електродів групового заземлювача $\eta_в$ =0,57 (табличне значення).

6) Визначається необхідна кількість вертикальних заземлювачів з урахуванням коефіцієнта використання $n_в$, шт:

$$n_в = \frac{2 \cdot R_в}{R_д \cdot \eta_в} = \frac{2 \cdot 18,5}{4 \cdot 0,57} = 16,2 \approx 16 \quad (4.8)$$

7) Визначається довжина з'єднувальної стрічки $l_с$, м:

$$l_с = 1,05 \cdot L_в \cdot (n_в - 1), \quad (4.9)$$

де $L_в$ – відстань між вертикальними заземлювачами, (беремо $L_в$ = 3м);

$n_в$ – необхідна кількість вертикальних заземлювачів.

$$l_с = 1,05 \cdot 3 \cdot (16 - 1) \approx 48 \text{ м}$$

8) Визначається опір розтікання струму горизонтального заземлювача (з'єднувальної стрічки) $R_г$, Ом:

$$R_г = \frac{\rho_{розр.г}}{2 \cdot \pi \cdot l_с} \cdot \ln \frac{2 \cdot l_с^2}{d_{см} \cdot h_г}, \quad (4.10)$$

де $d_{см}$ – еквівалентний діаметр смуги шириною b, $d_{см}$ =0,95b, b = 0,15 м;

$h_г$ – глибина закладання горизонтальних заземлювачів (0,5 м);

$l_с$ - довжина з'єднувальної стрічки горизонтального заземлювача $l_с$, м

$$R_г = \frac{220}{2 \cdot \pi \cdot 48} \cdot \ln \frac{2 \cdot 48^2}{0,95 \cdot 0,15 \cdot 0,5} = 8,1 \text{ Ом}$$

9) Визначається коефіцієнт використання горизонтального заземлювача $\eta_{с.}$ відповідно до необхідної кількості вертикальних заземлювачів $n_в$. Коефіцієнт використання $\eta_с$=0,3 (табличне значення).

10) Розраховується результуючий опір заземлювального:

$$R_{заг} = \frac{R_в \cdot R_г}{R_в \cdot \eta_с + R_г \cdot n_в \cdot \eta_в} \le R_д. \quad (4.11)$$

Висновок: дане захисне заземлення буде забезпечувати електробезпеку будівлі, так як виконується умова: $R_{заг}$ <4 Ом, а саме:

$$R_{заг} = \frac{18,5 \cdot 8,1}{18,5 \cdot 0,3 + 8,1 \cdot 16 \cdot 0,57} = 1,9 \le R_д$$

Висновки до розділу 4

В результаті проведеної роботи було зроблено аналіз умов праці, шкідливих та небезпечних чинників, з якими стикається робітник. Було визначено параметри і певні характеристики приміщення для роботи над запропонованим проектом, описано, які заходи потрібно зробити для того, щоб дане приміщення відповідало

необхідним нормам і було комфортним і безпечним для робітника. Приведені рекомендації щодо організації робочого місця, а також важливу інформацію щодо пожежної та електробезпеки. Були наведені розміри приміщення та наведено значення температури, вологості й рухливості повітря, необхідна кількість і потужність ламп та інші параметри, значення яких впливає на умови праці робітника, а також – наведені інструкції з охорони праці, техніки безпеки при роботі на комп’ютері.

ВИСНОВКИ

Завданням даної роботи бакалавра була розробка програмного забезпечення інформаційно-довідкової системи для середньої школи.

Проаналізувавши існуючі на даний момент типи програмних продуктів, що надають можливість управляти процесами в межах учбового закладу, зроблено вивід, що всі ці системи мають закритий вихідний код і не надають можливості зміни під конкретні потреби. Вартість даних програм хоч і невелика, але вимоги до програмного забезпечення вимагають додаткових фінансових вкладень. Саме це стало основною причиною доцільності створення інформаційно-довідкового веб-сервера, здатного надавати інформацію за допомогою мережі Internet.

Сьогодні бази даних - основа будь-якої крупної інформаційної системи, що зберігає і оброблює дані. В процесі роботи зроблено вибір СУБД MySQL, тому що, дана СУБД надає досить функціональності для розробки веб-сервера додатків і є однією з провідних СУБД. Вона показала відмінну продуктивність і безпеку даних при невисоких показниках апаратного забезпечення.

Для програмної реалізації інформаційно-довідкового веб-сервера була вибрана скриптова мова PHP. Вибір обумовлений простотою синтаксису, великою кількістю функцій для роботи з текстом, базами даних. PHP дає можливість генерувати не лише HTML сторінки, але і текстові документи графічні файли і ін. Так само простота використання і наявність початкового коду дозволили простіше змінювати, відладжувати або додавати нові компоненти в інформаційну систему.

Створена інформаційно-довідкова система для школи дозволила автоматизувати обмін інформацією між всіма учасниками учбового процесу, істотно зменшити час на обробку даних по успішності і відвідуваності учнів, збільшити швидкість ухвалення рішень керівництвом школи.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Вендров, А.М. CASE-технологии. Современные методы и средства проектирования информационных систем/ А.М. Вендров. - М.: Финансы и статистика, 1998.–176 с.

2. Маклаков, С.В. BPWin и ERWin. Case-средства разработки информационных систем/ С.В.Маклаков-М.: ДИАЛОГ–МИФИ, 1999.–256с.

3. Орлов, С.А. Технологии разработки программного обеспечения/ С.А. Орлов–СПб.: Питер, 2002.–464 с.

4. Гарсиа-Молина Г., Ульман Дж., Уидом Дж. Системы баз данных. Полный курс / Гарсиа-Молина Г, Ульман Дж, Уидом Дж. — М.: "Вильямс", 2003. – 229 с.

5. Дейт. К. Дж. Введение в системы баз данных / К. Дж. Дейт. — "Вильямс", 2001. – 426 с.

6. Харрингтон Д. Л Проектирование реляционных баз данных. Просто и доступно / Д. Л. Харрингтон. – М.: ЛОРИ, 2000. – 277 с.

7. Коннолли Т. М, Бегг К. Базы данных. Проектирование, реализация и сопровождение. Теория и практика / Т. М. Коннолли, К. Бегг. – М.: Издательский дом "Вильямс", 2003. – 261 с.

8. Калянов Г. Н. CASE. Структурный системный анализ (автоматизация и применение) / Г. Н. Калянов. – М.: "Лори", 2006. – 175 с.

9. Черемных, С.В. Структурный анализ систем: IDEF-технологии. /С.В. Черемных, И.О.Семенов, В.С. Ручкин-М.: Финансы и статистика, 2003.–208 с.

10. ГОСТ 12.1.044-89 ССБТ. Пожежовибухонебезпека речовин і матеріалів. Номенклатура показників і методи їх визначення.

11. ДБН В.2.5-28:2015 Природнє і штучне освітлення.

12. ДСанПіН 3.3.2.007-98 Гігієнічні вимоги до організації роботи з візуальними дисплейними терміналами електронно-обчислювальних машин.

13. ДСН 3.3.6.037-99 Санітарні норми виробничого шуму, ультразвуку та

інфразвуку.

14. ДСН 3.3.6.042-99 Санітарні норми мікроклімату виробничих.

15. НПАОП 0.00-4.12-05 Типове положення про порядок проведення навчання і перевірки знань з питань охорони праці.

16. НПАОП 0.00-4.15-98 Про розробку інструкцій з охорони праці.

17. НПАОП 0.00-6.03-93 Порядок опрацювання та затвердження власником нормативних актів про охорону праці.

18. НАПБ Б.03.002-2007 Норми визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою.

19. НПАОП 40.1-1.01-97 Правила безопасной эксплуатации електроустановок.

20. НПАОП 40.1-1.32-01 Правила устройства электроустановок. Электрооборудование специальных установок.

21. ДСН 3.3.6.039-99 Санітарні норми виробничої загальної та локальної вібрації.

22. ДСТУ ГОСТ 12.1.012-90 ССБТ. Вібраційна безпека. Загальні вимоги.

23. ДБН В.2.5-67:2013 Опалення, вентиляція та кондиціонування.

24. ГОСТ 12.1.006-84 ССБТ. Електромагнітні поля радіочастот. Загальні вимоги безпеки. Допустимі рівні на робочих місцях і вимоги до проведення контролю.

25. ГОСТ 12.1.030-81 ССБТ. Електробезпечність. Захисне заземлення. Занулення.

26. НПАОП 0.00-1.28-10 Правила охорони праці під час експлуатації електронно- обчислювальних машин.

ДОДАТОК А

Лістинг скриптів компонентів інформаційно-довідкового WEB-сервера

А.1 Скрипт ініціалізував класи і функції системи (common.php)

```
1   <?php
2   /** Путь к файлам **/
3   $fullpath = realpath("./");
4   /** Конфигурация **/
5   include "{$fullpath}include/config.php";
6   include "{$fullpath}include/class_database.php";
7   include "{$fullpath}include/functions_global.php";
8   include "{$fullpath}include/text_arrays.php";
9   include "{$fullpath}include/level_masking.php";
10  $db = new Mysql_database(
11  $conf["dbhost"],
12  $conf["dbuser"],
13  $conf["dbpass"],
14  $conf["dbname"]);
15  $db->db_set_charset($conf["dbchar"]);
16  /* Проверка авторизации */
17  $user = auth_module();
18  if(!isset($page)) $page="empty";
19  /*система шаблонов*/
20  include "{$fullpath}include/smarty/smarty_config.php";
21  ?>
```

А.2 Скрипт, що містить основні функції (function_global.php)

```
1   <?php
2   /** функции получения параметров запроса **/
3   function get_str($name, $default = "")
4   {
5   if (isset($_REQUEST[$name]))
6   return $_REQUEST[$name];
7   return $default;
8   }
9   function get_int($name, $default = -1)
10  {
11  if (isset($_REQUEST[$name]) && ctype_digit($_REQUEST[$name]))
12  return $_REQUEST[$name];
13  return $default;
14  }
15  /** функция проверки авторизации пользователя **/
16  function auth_module()
17  {
18  global $db, $conf, $lang;
```

```
19    session_name($conf['sid_name']);
20    session_start();
21    if (!isset($_SESSION["UID"]))
22    $_SESSION["UID"] = 0;
23    /* Проверка правильности ID пользователя */
24    if (!ctype_digit($_SESSION["UID"]))
25    $_SESSION["UID"] = 0;
26    /* Выполнение действий входа/выхода */
27    $f_user = preg_replace("/[^a-z0-9_]/usi", '', get_str("user"));
28    $f_pass = get_str("pass");
29    $f_act = get_str("act");
30    if (($f_user == "" || $f_pass == "") && $f_act == "login")
31    $f_act = "";
32    $f_lang = get_str("lang", "");
33    if (!in_array($f_lang, $conf["aval_langs"]))
34    $f_lang = "";
35    switch ($f_act) {
36    case 'login':
37    $f_pass = md5($f_pass . $conf['solt']);
38    $sql = "
39    SELECT a.`ltime`,a.`uid`,a.`lang`,u.*
40    FROM {$conf['prefix']}auth AS a
41    LEFT JOIN {$conf['prefix']}users AS u ON a.uid=u.uid
42    WHERE a.`user`='$f_user' AND a.`pass`='{$f_pass}';";
43    $data = $db->db_fetch_assoc($db->db_query($sql));
44    if (is_array($data) && $data['uid'] > 0) {
45    $ltime = time();
46    $IP = $_SERVER['REMOTE_ADDR'];
47    $SID = $db->db_real_escape_string(session_id());
48    $UID = $data['uid'];
49    $_SESSION["UID"] = $UID;
50    //lang
51    if ($data["lang"] == "") {
52    $data["lang"] = $conf["deflang"];
53    }
54    $sql = "
55    UPDATE {$conf['prefix']}auth
56    SET `ltime`='{$ltime}',
57    `sid`='{$SID}',
58    `lip`='{$IP}',
59    `lang`='{$data['lang']}'
60    WHERE `uid`='{$UID}';";
61    $db->db_query($sql);
62    $lang = $data['lang'];
63    return $data;
64    }
65    break;
66    case 'logout':
67    break;
68    default:
69    /** CHECK IF LOGGED IN **/
70    $IP = @$_SERVER['REMOTE_ADDR'];
```

```
71  $SID = $db->db_real_escape_string(session_id());
72  $UID = $_SESSION['UID'];
73  if ($UID > 0 && ctype_digit($UID)) {
74  $ltime = time();
75  $sql = "
76  SELECT a.`ltime`,a.`uid`,a.`lang`,u.*
77  FROM {$conf['prefix']}auth AS a
78  LEFT JOIN {$conf['prefix']}users AS u ON a.uid=u.uid
79  WHERE a.`sid`='{$SID}'
80  AND a.`lip`='{$IP}'
81  AND a.`uid`='{$UID}';";
82  $data = $db->db_fetch_assoc($db->db_query($sql));
83  if (is_array($data)) {
84  if ($data["lang"] == "") {
85  if ($f_lang == "")
86  $f_lang = $conf["deflang"];
87  $data["lang"] = $f_lang;
88  } else {
89  if ($f_lang != $data["lang"] && $f_lang != "")
90  $data["lang"] = $f_lang;
91  }
92  $sql = "
93  UPDATE {$conf['prefix']}auth
94  SET `ltime`='{$ltime}',
95  `lang`='{$data['lang']}'
96  WHERE `uid`='{$UID}'";
97  $db->db_query($sql);
98  $lang = $data['lang'];
99  return $data;
100 }
101 }
102 break;
103 }
104 $_SESSION['UID'] = 0;
105 $lang = $conf["deflang"];
106 return array("uid" => 0, "level" => 0, "lang" => $conf["deflang"]);
107 }
108 /** функции работы с временем **/
109 function md_month_w($stamp)
110 { //Начало недели месяца
111 if (date("I", $stamp) == 0)
112 $stamp += 60 * 60;
113 $stamp = strtotime(date("1 F Y", $stamp));
114 $dn = date("w", $stamp);
115 if ($dn == 0)
116 $dn = 7;
117 $stamp -= (($dn - 1) * (60 * 60 * 24));
118 return strtotime(date("j F Y", $stamp));
119 }
120 function md_month_b($stamp)
121 { //Первый день месяца
122 if (date("I", $stamp) == 0)
```

```
123     $stamp += 60 * 60;
124     return strtotime(date("1 F Y", $stamp));
125     }
126     function md_month_e($stamp)
127     { //Последний день месяца
128     if (date("I", $stamp) == 0)
129     $stamp += 60 * 60;
130     $ed = date("t", $stamp);
131     return strtotime(date("{$ed} F Y", $stamp));
132     }
133     function md_week($stamp)
134     { // Начало недели
135     if (date("I", $stamp) == 0)
136     $stamp += 60 * 60;
137     $dn = date("w", $stamp);
138     if ($dn == 0)
139     $dn = 7;
140     $stamp -= (($dn - 1) * (60 * 60 * 24));
141     return strtotime(date("j F Y", $stamp));
142     }
143     function md_day($stamp)
144     { //Начало дня
145     if (date("I", $stamp) == 0)
146     $stamp += 60 * 60;
147     return strtotime(date("j F Y", $stamp));
148     }
149     /** Текстовое представление даты/времени **/
150     function date_format1($time){
151       $days = array(
152         1 => "Понедельник",
153         2 => "Вторник",
154         3 => "Среда",
155         4 => "Четверг",
156         5 => "Пятница",
157         6 => "Суббота",
158         7 => "Воскресенье");
159       $mons1 = array(
160         1 => "января",
161         2 => "февраля",
162         3 => "марта",
163         4 => "апреля",
164         5 => "мая",
165         6 => "июня",
166         7 => "июля",
167         8 => "августа",
168         9 => "сентября",
169         10 => "октября",
170         11 => "ноября",
171         12 => "декабря");
172       $dn = date("w", $time);
173       if ($dn == 0)
174         $dn = 7;
```

```
175     $mn = date("n", $time);
176     $dd = date("j", $time);
177     return $days[$dn] . " {$dd} " . $mons1[$mn];
178   }
179   /** функция вывода шаблона и зкрытия базы данных **/
180   function draw_page()
181   {
182   global $page, $db, $conf, $smarty;
183   $pg = $page . ".tpl";
184   $smarty->display($pg);
185   if (isset($db))
186   $db->db_close();
187   }
188   /** Функция завершения скрипта без вывода шаблона **/
189   function exit_script(){
190   global $db;
191   if (isset($db))
192   $db->db_close();
193   exit();
194   }
195   /** функции преобразования **/
196   function db2html($text)
197   {
198   return htmlentities($text, ENT_QUOTES, 'UTF-8');
199   }
200   /** функция смены кодировки **/
201   function utf8_win($s)
202   {
203   $out = "";
204   $c1 = "";
205   $byte2 = false;
206   for ($c = 0; $c < strlen($s); $c++) {
207   $i = ord($s[$c]);
208   if ($i <= 127)
209   $out .= $s[$c];
210   if ($byte2) {
211   $new_c2 = ($c1 & 3) * 64 + ($i & 63);
212   $new_c1 = ($c1 >> 2) & 5;
213   $new_i = $new_c1 * 256 + $new_c2;
214   if ($new_i == 1025) {
215   $out_i = 168;
216   } else {
217   if ($new_i == 1105) {
218   $out_i = 184;
219   } else {
220   $out_i = $new_i - 848;
221   }
222   }
223   $out .= chr($out_i);
224   $byte2 = false;
225   }
226   if (($i >> 5) == 6) {
```

```
227     $c1 = $i;
228     $byte2 = true;
229     }
230     }
231     return $out;
232     }
233     function win_utf8($in_text)
234     {
235     $output = "";
236     $other[1025] = "Ё";
237     $other[1105] = "ё";
238     $other[1028] = "Є";
239     $other[1108] = "є";
240     $other[1030] = "I";
241     $other[1110] = "i";
242     $other[1031] = "Ї";
243     $other[1111] = "ї";
244     for ($i = 0; $i < strlen($in_text); $i++) {
245     if (ord($in_text{$i}) > 191) {
246     $output .= "&#" . (ord($in_text{$i}) + 848) . ";";
247     } else {
248     if (array_search($in_text{$i}, $other) === false) {
249     $output .= $in_text{$i};
250     } else {
251     $output .= "&#" . array_search($in_text{$i}, $other) . ";";
252     }
253     }
254     }
255     return $output;
256     }
257     //Генератор календаря.
258     function get_calendar($ctime)
259     {
260     global $array_monthes;
261     $cur_day = md_day($ctime);
262     $cur_week = md_week($cur_day);
263     $cweek = $cur_day;
264     $wb = md_month_w($cweek);
265     $wbs = $wb;
266     $mon_n = date("n", $cweek);
267     $cal = array();
268     while (date("n", $wb) == $mon_n || $wb <= ($wbs + 60 * 60 * 24 * 14))
269     {
270     $wk = array("stamp" => $wb, "days" => array(), "this" => 0);
271     if ($cur_week == $wb)
272     $wk["this"] = 1;
273     for ($i = 0; $i <= 6; $i++) {
274     $stm = md_day($wb + 60 * 60 * 24 * $i);
275     $mon_s = date("n", $stm);
276     $day_s = date("j", $stm);
277     if ($mon_n == $mon_s)
278     $mon_s = 1;
```

```
279     else
280     $mon_s = 0;
281     if ($stm == md_day(time()))
282     $day_t = 1;
283     else
284     $day_t = 0;
285     $wk["days"][] = array(
286     "this" => $day_t,
287     "day" => $day_s,
288     "month" => $mon_s,
289     "stamp" => $stm,
290     "holiday" => 0);
291     }
292     $cal[] = $wk;
293     $wb = md_week($wb + 60 * 60 * 24 * 7);
294     }
295     $calendar = $cal;
296     $cal_date = $array_monthes[$mon_n] . " " . date("Y", $cweek);
297     $cal_mplus = md_month_b(md_month_b($cweek) + 60 * 60 * 24 * 32);
298     $cal_mminus = md_month_b(md_month_b($cweek) - 60 * 60 * 24 * 2);
299     return array(
300       "calendar" => $cal,
301       "mnext" => $cal_mplus,
302       "mprev" => $cal_mminus,
303       "date_txt" => $cal_date,
304       "cur_day" => $cur_day,
305       "cur_week" => $cur_week);
306     }
307     ?>
```

A.3 Скрипт, що відображує розклад класу на тиждень (schedule.php)

```
1       <?php
2       /** загрузка параметров **/
3       include "common.php";
4       $action = get_str("act","main");
5       /** проверка уровня пользователя **/
6       if($user["level"] < 3 || ($user["level_access"]&EDIT_SCHEDULE==0))
7       { $action ="redirect"; $url="index.php?"; }
8       if($action=="main"){
9       $ctime = time();
10      $cur_day = get_int("time",md_day($ctime));
11      if($cur_day!=md_day($cur_day)) $cur_day = md_day($ctime);
12      $cur_week = md_week($cur_day);
13      //PERIODS
14      $n_period = get_int("per",0);
15      $sql = "SELECT * FROM {$conf['prefix']}periods
16      WHERE `sch_id`={$user['sch_id']} AND `time_end`>{$ctime}
17      AND `state`=0 ORDER BY `time_begin` ASC;";
18      $result = $db->db_query($sql);
19      $periods = array();
20      $cur_period = array();
```

```
21  while($row=$db->db_fetch_assoc($result)){
22  if($n_period == 0) $n_period = $row['period'];
23  if($row['period']==$n_period){
24  $cur_period = $row;
25  if($cur_day>$row['time_end']) $cur_day=$row['time_end'];
26  if($cur_day<$row['time_begin']) $cur_day=$row['time_begin'];
27  $cur_week = md_week($cur_day);
28  }
29  $periods[]=$row;
30  }
31  //CLASSES
32  $n_class = get_int("grp",0);
33  $sql = "SELECT * FROM {$conf['prefix']}classes
34  WHERE `sch_id`={$user['sch_id']} AND `grp_state`=0 ORDER BY `grp_grade` DESC;";
35  $result = $db->db_query($sql);
36  $classes = array();
37  $cur_class = array();
38  while($row=$db->db_fetch_assoc($result)){
39  if($n_class == 0) $n_class = $row['grp_id'];
40  if($row['grp_id']==$n_class) $cur_class = $row;
41  $classes[]=$row;
42  }
43  //CALENDAR
44  $cweek = $cur_day;
45  $wb = md_month_w($cweek);
46  $wbs = $wb;
47  $mon_n = date("n",$cweek);
48  $cal = array();
49  while(date("n",$wb)==$mon_n || $wb<=($wbs+60*60*24*14)){
50  $wk = array("stamp"=>$wb,"days"=>array(),"this"=>0);
51  if($cur_week==$wb) $wk["this"]=1;
52  for($i=0;$i<=6;$i++){
53  $stm = md_day($wb+60*60*24*$i);
54  $mon_s = date("n",$stm);
55  $day_s = date("j",$stm);
56  if($mon_n==$mon_s) $mon_s = 1; else $mon_s = 0;
57  if($stm == md_day(time())) $day_t = 1; else $day_t=0;
58  $wk["days"]
    []=array("this"=>$day_t,"day"=>$day_s,"month"=>$mon_s,"stamp"=>$stm,"holiday"=>0);
59  }
60  $cal[]=$wk;
61  $wb = md_week($wb+60*60*24*7);
62  }
63  $calendar=$cal;
64  $cal_date=$array_monthes[$mon_n]." ".date("Y",$cweek);
65  $cal_mplus=md_month_b(md_month_b($cweek)+60*60*24*32);
66  $cal_mminus=md_month_b(md_month_b($cweek)-60*60*24*2);
67  //MAINFRAME
68  $schedule = array();
69  if(count($classes)>0 && count($periods)>0){
70  //GET SCHOOL_INFO
71  $sql = "SELECT * FROM {$conf['prefix']}schools WHERE `sch_id`={$user['sch_id']};";
```

```
72    $school = $db->db_fetch_assoc($db->db_query($sql));
73    //CREATE STRUCTURE
74    $n_lesson=array("no"=>0,"no_abs"=>0,"les_id"=>0,"les_name"=>"","room"=>"",
75    "group_task"=>"","tch1_id"=>0,"tch2_id"=>0,"tch3_id"=>0,"tch1_name"=>"",
76    "tch2_name"=>"","tch3_name"=>"","can_edit"=>1,"can_task"=>1,
77    "can_teachers"=>1,"can_room"=>1,"sched"=>0);
78    $n_les = array_fill(0,$school['school_lessons'],$n_lesson);
79    $n_day = array("no"=>0,"holiday"=>0,"holiday_name"=>"",
80    "date_text"=>"","time"=>0,"lessons"=>$n_les,"comments"=>array(),"can_comment"=>1);
81    $schedule = array_fill(0, $school['school_days'], $n_day);
82    //FILL STRUCTURE
83    foreach($schedule as $k=>$v){
84    $schedule[$k]["no"]=$k+1;
85    $schedule[$k]["time"]=md_day($cur_week+(60*60*24*($k)));
86    $schedule[$k]["date_text"]=date_format1($schedule[$k]["time"]);
87    //end;
88    foreach($schedule[$k]["lessons"] as $i=>$j){
89    $schedule[$k]["lessons"][$i]["no"]=$i+1;
90    $schedule[$k]["lessons"][$i]["no_abs"]=$i+1;}
91    }
92    //LOAD DATA FROM DATABASE;
93    $sql = "
94    SELECT *
95    FROM {$conf['prefix']}schedule
96    WHERE `sch_id`={$user['sch_id']}
97    AND `grp_id`={$cur_class['grp_id']}
98    AND `week`={$cur_week}
99    AND `period`={$cur_period['period']}
100   AND `state`=0;";
101   $result = $db->db_query($sql);
102   while($row=$db->db_fetch_assoc($result)){
103   $xd = $row['day_no']-1;
104   $xl = $row['les_abs']-1;
105   $schedule[$xd]["lessons"][$xl]["les_id"]=$row['les_id'];
106   $schedule[$xd]["lessons"][$xl]["les_name"]=db2html($row['les_name']);
107   $schedule[$xd]["lessons"][$xl]["room"]=db2html($row['room']);
108   //teachers:
109   $schedule[$xd]["lessons"][$xl]["tch1_id"]=$row['tch1'];
110   $schedule[$xd]["lessons"][$xl]["tch2_id"]=$row['tch2'];
111   $schedule[$xd]["lessons"][$xl]["tch3_id"]=$row['tch3'];
112   $schedule[$xd]["lessons"][$xl]["tch1_name"]=db2html($row['tch1_name']);
113   $schedule[$xd]["lessons"][$xl]["tch2_name"]=db2html($row['tch2_name']);
114   $schedule[$xd]["lessons"][$xl]["tch3_name"]=db2html($row['tch3_name']);
115   //task:
116   $schedule[$xd]["lessons"][$xl]["group_task"]=db2html($row['grp_task']);
117   $schedule[$xd]["lessons"][$xl]["sched"]=$row['sched_id'];
118   }
119   //COMMENTS,HOLIDAYS
120   $sql = "SELECT * FROM {$conf['prefix']}schedule_comm
121   WHERE `comm_type`=1 AND `grp_id`={$cur_class['grp_id']}
122   AND `week`={$cur_week} AND `uid`=0 ORDER BY `post_time` DESC;";
123   $result = $db->db_query($sql);
```

```
124     while($row=$db->db_fetch_assoc($result)){
125     $schedule[$row['day_no']-1]["comments"][]=$row;
126     }
127     /** LOAD LESSONS,TEACHERS **/
128     $sql = "SELECT * FROM {$conf['prefix']}users
129     WHERE `sch_id`={$user['sch_id']} AND `level`=3
130     AND `state`=0 ORDER BY `nick` ASC;";
131     $result = $db->db_query($sql);
132     $teachers = array();
133     while($row=$db->db_fetch_assoc($result)){
134     $row['udn']=db2html($row['nick']);
135     $teachers[]=$row;
136     }
137     $sql = "SELECT * FROM {$conf['prefix']}lessons
138     WHERE `sch_id`={$user['sch_id']} AND `state`=0 ORDER BY `les_name` ASC;";
139     $result = $db->db_query($sql);
140     $lessons = array();
141     while($row=$db->db_fetch_assoc($result)){
142     $lessons[]=$row;
143     }
144     }
145     }
146     //print_r($schedule);
147     switch($action){
148     case 'main':
149     $smarty->assign("periods",$periods);
150     $smarty->assign("classes",$classes);
151     $smarty->assign("cperiod",$cur_period);
152     $smarty->assign("cclass",$cur_class);
153     $smarty->assign("schedule",$schedule);
154     $smarty->assign("lessons",$lessons);
155     $smarty->assign("teachers",$teachers);
156     //calndar
157     $smarty->assign("cal_date",$cal_date);
158     $smarty->assign("cal_mplus",$cal_mplus);
159     $smarty->assign("cal_mminus",$cal_mminus);
160     $smarty->assign("cal",$calendar);
161     $smarty->assign("cur_day",$cur_day);
162     $smarty->assign("cur_week",$cur_week);
163     $page = "schedule";
164     break;
165     default:
166     $url = "index.php?";
167     case "redirect":
168     header("Location: {$url}");
169     exit_script();
170     break;
171     }
172     $smarty->assign("page_js",$page);
173     $smarty->assign("user",$user);
174     $smarty->assign("dpath",$conf["data_url"]);
175     $smarty->assign("lang",$lang);
```

```
176   draw_page();
177   ?>
```

А.4 Скрипт, що виконує приховані дії редактора щоденника (teacher_diary_ajax.php)

```
1     <?php
2     /** загрузка параметров **/
3     include "common.php";
4     $action = get_str("act","main");
5     /** проверка уровня пользователя **/
6     if($user["level"] < 3 || $user["level_access"]&EDIT_SCHEDULE==0){ $action ="access"; }
7     /** получение и обработка данных **/
8     $msg = "";
9     switch($action){
10    case 'edit':
11    $n_lid = get_int("lid",0);
12    $n_week = get_int("week",0);
13    $n_period = get_int("per",0);
14    $n_grp = get_int("grp",0);
15    $n_day = get_int("day",0);
16    $n_les = get_int("les",0);
17    $n_room = urldecode(get_str("room"));
18    $n_type = get_int("type",0);
19    //Проверка входящих значений
20    $all_ok = true;
21    if($n_week!=md_week($n_week)) $all_ok = false;
22    if($n_grp==0 || $n_period==0 || $n_week==0 || $n_day==0 || $n_les==0 || $n_type>1)
      $all_ok=false;
23    //TODO: n_day,n_les based on school_info;
24    if($all_ok){
25    $sql = "SELECT
26    (SELECT 'yes' FROM {$conf['prefix']}lessons WHERE `les_id`={$n_lid} AND
      `sch_id`={$user['sch_id']} AND `state`=0),
27    (SELECT 'yes' FROM {$conf['prefix']}classes WHERE `grp_id`={$n_grp} AND
      `sch_id`={$user['sch_id']} AND `grp_state`=0),
28    (SELECT 'yes' FROM {$conf['prefix']}periods WHERE `period`={$n_period} AND
      `sch_id`={$user['sch_id']} AND `time_begin`<={$n_week} AND `time_end`>={$n_week}
      AND `state`=0)
29    ";
30    $info = $db->db_fetch_array($db->db_query($sql));
31    if($info[0]!='yes' || $info[1]!='yes' || $info[2]!='yes') $all_ok = false;
32    }
33    if(!$all_ok){
34      $aciton=”error”;
35      $msg = “Ошибка параметров”;
36    }
37    //Дополнительные проверки возможности редактирования
38    //Получение необходимых данных;
39    if($n_lid>0){ //Установить значения;
40    //информация об уроке
41    $sql = "SELECT * FROM {$conf['prefix']}lessons WHERE `sch_id`={$user['sch_id']} AND
```

```
        `state`=0 AND `les_id`={$n_lid};";
42      $lesson = $db->db_fetch_assoc($db->db_query($sql));
43      if($n_type>0){
44      $sql = "SELECT `time_begin`,`time_end` FROM {$conf['prefix']}periods WHERE
        `period`={$n_period};";
45      $period = $db->db_fetch_assoc($db->db_query($sql));
46      $weeks = array();
47      $st=$n_week;
48      while($st<=$period['time_end']){
49      $weeks[]=md_week($st);
50      $st+=60*60*24*7; //прибавляем неделю
51      }
52      }else{
53      $weeks = array(0=>$n_week);
54      }
55      $w1 = implode(",",$weeks);
56      $sql = "DELETE FROM {$conf['prefix']}schedule WHERE `sch_id`={$user['sch_id']} AND
        `week` IN ({$w1}) AND `les_abs`={$n_les} AND `day_no`={$n_day} AND `state`=0;";
57      $db->db_query($sql);
58      $sql="INSERT INTO {$conf['prefix']}schedule
59      (`sch_id`,`grp_id`,`week`,`period`,`day_no`,`les_abs`,`les_id`,`les_name`,`room`,`tch1`,
60      `tch2`,`tch3`,`tch1_name`,`tch2_name`,`tch3_name`)VALUES";
61      $da = array();
62      foreach($weeks as $w){
63      $d_lesname = $db->db_real_escape_string($lesson['les_name']);
64      $d_room = $db->db_real_escape_string($n_room);
65      $d_tch1 = $db->db_real_escape_string($lesson['tch1_name']);
66      $d_tch2 = $db->db_real_escape_string($lesson['tch2_name']);
67      $d_tch3 = $db->db_real_escape_string($lesson['tch3_name']);
68      $da[]= "({$user['sch_id']},{$n_grp},{$w},{$n_period},{$n_day},{$n_les},
69      {$lesson['les_id']},'{$d_lesname}','{$d_room}',{$lesson['tch1']},
70      {$lesson['tch2']},{$lesson['tch3']},'{$d_tch1}','{$d_tch2}','{$d_tch3}')";
71      }
72      $sql .= implode(",",$da);
73      $db->db_query($sql);
74      }else{ //Очистка расписания
75      if($n_type>0){
76      $weeks = "AND `week`>={$n_week}";
77      }else{
78      $weeks = "AND `week`={$n_week}";
79      }
80      $sql = "DELETE FROM {$conf['prefix']}schedule WHERE
81       `sch_id`={$user['sch_id']} AND `period`={$n_period} AND `grp_id`={$n_grp}
82      AND `day_no`={$n_day} AND `les_abs`={$n_les} {$weeks};";
83      $db->db_query($sql);
84      }
85      //Создаем данные для отображения
86      if($n_lid>0){
87      $o_act = 1; //place;
88      $o_day = $n_day;
89      $o_les = $n_les;
90      $o_lid = $n_lid;
```

```
91    $o_name = rawurlencode($lesson['les_name']);
92    $o_room = rawurlencode($n_room);
93    $o_task = 0;
94    $o_tch1 = rawurlencode($lesson['tch1_name']);
95    $o_tch2 = rawurlencode($lesson['tch2_name']);
96    $o_tch3 = rawurlencode($lesson['tch3_name']);
97    //this schedule
98    $sql = "SELECT `sched_id`,`les_id` FROM {$conf['prefix']}schedule
99    WHERE `period`={$n_period} AND `week`={$n_week} AND `day_no`={$n_day}
100   AND `les_abs`={$n_les} AND `state`=0;";
101   $nrec = $db->db_fetch_assoc($db->db_query($sql));
102   $msg = "ret_code={$o_act};les={$o_les};day={$o_day};lid={$o_lid};
103   name='{$o_name}';room='{$o_room}';task=0;tch1={$lesson['tch1']};
104   tch2={$lesson['tch2']};tch3={$lesson['tch3']};tch1n='{$o_tch1}';
105   tch2n='{$o_tch2}';tch3n='{$o_tch3}';sched={$nrec['sched_id']}";
106   }else{
107   $o_act = 2; //clear all;
108   $o_day = $n_day;
109   $o_les = $n_les;
110   $msg = "ret_code={$o_act};les={$o_les};day={$o_day};";
111   }
112   break;
113   case 'delete':
114   break;
115   case 'task':
116   $text = urldecode(get_str("text"));
117   $sched = get_int("sched",0);
118   if($sched==0){ $msg = "ret_code=9;";break; }
119   #check exist
120   $sql = "SELECT `day_no`,`les_abs`,`tch1`,`tch2`,`tch3` FROM {$conf['prefix']}schedule
121   WHERE `sch_id`={$user['sch_id']} AND `sched_id`={$sched} AND `state`=0;";
122   $record = $db->db_fetch_assoc($db->db_query($sql));
123   if(!is_array($record)){$msg = "ret_code=9;";break;}
124   #check rights;
125   #place task;
126   $ntext = $db->db_real_escape_string($text);
127   $sql = "UPDATE {$conf['prefix']}schedule SET `grp_task`='{$ntext}'
128   WHERE `sched_id`={$sched};";
129   $db->db_query($sql);
130   $ntext = rawurlencode($text);
131   $msg = "ret_code=1;text='{$ntext}';day={$record['day_no']};les={$record['les_abs']};";
132   break;
133   case 'room':
134   $text = urldecode(get_str("room"));
135   $sched = get_int("sched",0);
136   if($sched==0){ $msg = "ret_code=9;";break; }
137   #check exist
138   $sql = "SELECT `day_no`,`les_abs`,`tch1`,`tch2`,`tch3` FROM {$conf['prefix']}schedule
139   WHERE `sch_id`={$user['sch_id']} AND `sched_id`={$sched} AND `state`=0;";
140   $record = $db->db_fetch_assoc($db->db_query($sql));
141   if(!is_array($record)){$msg = "ret_code=9;";break;}
142   #check rights;
```

```
143 #place room;
144 $ntext = $db->db_real_escape_string($text);
145 $sql = "UPDATE {$conf['prefix']}schedule SET `room`='{$ntext}'
146 WHERE `sched_id`={$sched};";
147 $db->db_query($sql);
148 $ntext = rawurlencode($text);
149 $msg = "ret_code=1;text='{$ntext}';day={$record['day_no']};les={$record['les_abs']};";
150 break;
151 case 'teachers':
152 $tch1 = get_int("tch1",0);
153 $tch2 = get_int("tch2",0);
154 $tch3 = get_int("tch3",0);
155 $sched = get_int("sched",0);
156 if($sched==0){ $msg = "ret_code=9;";break; }
157 #check exist
158 $sql = "SELECT `day_no`,`les_abs`,`tch1`,`tch2`,`tch3` FROM {$conf['prefix']}schedule
159 WHERE `sch_id`={$user['sch_id']} AND `sched_id`={$sched} AND `state`=0;";
160 $record = $db->db_fetch_assoc($db->db_query($sql));
161 if(!is_array($record)){$msg = "ret_code=9;";break;}
162 #check rights;
163 #get_teachers_names;
164 $imp = "{$tch1},{$tch2},{$tch3}";
165 $sql = "SELECT * FROM {$conf['prefix']}users
166 WHERE (`uid`={$tch1} OR `uid`={$tch2} OR `uid`={$tch3})
167 AND `level`=3 AND `sch_id`={$user['sch_id']};";
168 $result = $db->db_query($sql);
169 $tcn = array(0=>"");
170 while($row=$db->db_fetch_assoc($result)){
171 $tcn[$row['uid']]=$row['nick'];
172 }
173 $tcn1 = $db->db_real_escape_string($tcn[$tch1]);
174 $tcn2 = $db->db_real_escape_string($tcn[$tch2]);
175 $tcn3 = $db->db_real_escape_string($tcn[$tch3]);
176 #place room;
177 $sql = "UPDATE {$conf['prefix']}schedule SET
178  `tch1_name`='{$tcn1}',`tch2_name`='{$tcn2}',`tch3_name`='{$tcn3}',
179 `tch1`={$tch1},`tch2`={$tch2},`tch3`={$tch3} WHERE `sched_id`={$sched};";
180 $db->db_query($sql);
181 $ntch1 = rawurlencode($tcn[$tch1]);
182 $ntch2 = rawurlencode($tcn[$tch2]);
183 $ntch3 = rawurlencode($tcn[$tch3]);
184 $msg = "ret_code=1;tch1n='{$ntch1}';tch2n='{$ntch2}';tch3n='{$ntch3}';
185 tch1={$tch1};tch2={$tch2};tch3={$tch3};
186 day={$record['day_no']};les={$record['les_abs']};";
187 break;
188 case 'comment':
189 $text = urldecode(get_str("text"));
190 $title = urldecode(get_str("title"));
191 $week = get_int("week",0);
192 $grp = get_int("grp",0);
193 $day = get_int("day",0);
194 if($week==0){ $msg = "ret_code=9;";break; }
```

```
195  if($text==""){ $msg = "ret_code=9;";break; }
196  if($title==""){ $msg = "ret_code=9;";break; }
197  $ntext = $db->db_real_escape_string($text);
198  $ntitle = $db->db_real_escape_string($title);
199  $nposter = $db->db_real_escape_string($user['nick']);
200  $ptime = time();
201  $sql = "
202  INSERT INTO {$conf['prefix']}schedule_comm
203  (`comm_type`,`week`,`day_no`,`grp_id`,`uid`,`post_uid`,`post_uname`,`post_title`,
204  `post_text`,`post_time`)
205  VALUES(1,{$week},{$day},{$grp},0,{$user['uid']},'{$nposter}','{$ntitle}',
206  '{$ntext}',{$ptime});";
207  $db->db_query($sql);
208  $nn = $db->db_insert_id();
209  $ntext = rawurlencode($text);
210  $ntitle = rawurlencode($title);
211  $nposter = rawurlencode($user['nick']);
212  $nptime = "nn-22-3000";
213  $msg = "ret_code=1;text='{$ntext}';
214  title='{$ntitle}';poster='{$nposter}';day={$day};cid={$nn};";
215  break;
216  case 'commdel':
217  $cid = get_int("cid",0);
218  if($cid==0){ $msg = "ret_code=9;";break; }
219  $sql = "DELETE FROM {$conf['prefix']}schedule_comm
220   WHERE `comm_id`={$cid};";
221  $db->db_query($sql);
222  $msg = "ret_code=1;id={$cid}";
223  break;
224  case 'access':
225  $msg = 'alert("access failed;");';
226  break;
227  default:
228  $msg = 'current_action=0;';
229  break;
230  }
231  echo $msg;
232  exit_script();
233  ?>
```

А.5 Скрипт виконується на стороні клієнта. Функції елементів редактора щоденника. (teacher_diary.js)

```
1   var page_name = "teacher_diary";
2   var current_action = 0;
3   //NAVIGTOR
4   function change_view(){
5   location="schedule.php?time="+ge("cur_day").value+"&grp="+ge("sel_class").value+
6   "&per="+ge("sel_period").value;
7   }
```

```
8       //COMMENTS EXPAND;
9       function exp_comments(day){
10      if(ge("d"+day+"_comm_block").style.display=="none"){
11      ge("d"+day+"_comm_block").style.display="block";
12      }else{
13      ge("d"+day+"_comm_block").style.display="none";
14      }
15      }
16      //TASK EXPAND
17      function exp_tasks(day,les){
18      if(ge("d"+day+"l"+les+"_task_block").style.display=="none"){
19      ge("d"+day+"l"+les+"_task_block").style.display="block";
20      ge("d"+day+"l"+les+"_task_img_exp").src = ge("d"+day+"l"+les+
21      "_task_img_exp").src.replace("_collapse.gif","_expand.gif");
22      }else{
23      ge("d"+day+"l"+les+"_task_block").style.display="none";
24      ge("d"+day+"l"+les+"_task_img_exp").src = ge("d"+day+"l"+les+
25      "_task_img_exp").src.replace("_expand.gif","_collapse.gif");
26      }
27      }
28      //WINDOWS FUNCTIONS
29      function win_mark(day,les){
30      if(current_action>0) return;
31      if(ge("d"+day+"l"+les+"_lid").value==0) return;
32      TagToTip('win_mark', TITLEFONTCOLOR, '#CCFFCC',CLOSEBTN, true, STICKY,
33      true,PADDING,6, TITLE,'Выставление оценки',CENTERWINDOW,true,DELAY, 0);
34      ge("wmark_sched").value = ge("d"+day+"l"+les+"_sched_id").value;
35      ge("wmark_date").innerHTML = ge("d"+day+"_day_name").innerHTML+
36      " ("+les+" урок)";;
37      ge("wmark_lesson").innerHTML = ge("d"+day+"l"+les+"_lesname").innerHTML;
38      ge("wmark_desc").value = ge("d"+day+"l"+les+"_mark_desc").innerHTML;
39      ge("wmark_submit").disabled = false;
40      var mark = parseInt(ge("d"+day+"l"+les+"_mark").innerHTML);
41      if(mark==NaN) mark=0;
42      set_selector("wmark_mark",mark);
43      }
44      function set_mark(){
45      var uid = ge("wmark_uid").value;
46      var sched = ge("wmark_sched").value;
47      var mark = ge("wmark_mark").value;
48      var desc = encodeURIComponent(ge("wmark_desc").value);
49      var send_params = "act=mark&uid="+uid+"&sched="+sched+"&mark="
50      +mark+"&desc="+desc;
51      current_action = 1;
52      ge("wmark_submit").disabled=true;
53      ajax("teacher_diary_ajax.php","post",send_params,set_mark_done);
54      }
55      function set_mark_done(){
56      current_action = 0;
57      ge("wmark_submit").disabled = false;
58      var ret_code,sub_code,day,les,mark,desc;
59      try{
```

```
60      eval(req.responseText);
61      }catch(err){
62      alert(req.responseText+err);
63      }
64      if(ret_code==1){ //mark_set;
65      ge("d"+day+"l"+les+"_mark_desc").innerHTML = html_encode(
66      decodeURIComponent(desc));
67      ge("d"+day+"l"+les+"_mark").innerHTML = mark;
68      }
69      if(ret_code==9){ //error;
70      if(sub_code==1){
71      }
72      }
73      UnTip();
74      }
75      function win_task(day,les,type){
76      if(current_action>0) return;
77      if(ge("d"+day+"l"+les+"_lid").value==0) return;
78      TagToTip('win_task', TITLEFONTCOLOR, '#CCFFCC',CLOSEBTN, true, STICKY,
79      true,PADDING,6, TITLE,'Редактирование задание',CENTERWINDOW,true,
80      DELAY, 0);
81      ge("wtask_sched").value = ge("d"+day+"l"+les+"_sched_id").value;
82      ge("wtask_date").innerHTML = ge("d"+day+"_day_name").innerHTML+
83      " ("+les+" урок)";;
84      ge("wtask_lesson").innerHTML = ge("d"+day+"l"+les+"_lesname").innerHTML;
85      if(type==1){//group
86      ge("wtask_type").value = 1;
87      ge("wtask_typetext").innerHTML = "для класса";
88      ge("wtask_text").innerHTML = ge("d"+day+"l"+les+"_gtask").innerHTML;
89      }else{//personal
90      ge("wtask_type").value = 2;
91      ge("wtask_typetext").innerHTML = "персональное";
92      ge("wtask_text").innerHTML = ge("d"+day+"l"+les+"_stask").innerHTML;
93      }
94      ge("wtask_submit").disabled = false;
95      }
96      function set_task(){
97      var uid = ge("wtask_uid").value;
98      var sched = ge("wtask_sched").value;
99      var type = ge("wtask_type").value;
100     var text = encodeURIComponent(ge("wtask_text").value);
101     var send_params;
102     if(type==1){
103     send_params = "act=gtask&sched="+sched+"&text="+text;
104     }else{
105     send_params = "act=stask&sched="+sched+"&uid="+uid+"&text="+text;
106     }
107     current_action = 2;
108     ge("wtask_submit").disabled=true;
109     ajax("teacher_diary_ajax.php","post",send_params,set_task_done);
110     }
111     function set_task_done(){
```

```
112  current_action = 0;
113  ge("wtask_submit").disabled = false;
114  var ret_code,sub_code,day,les,text;
115  try{
116  eval(req.responseText);
117  }catch(err){
118  alert(req.responseText+err);
119  }
120  if(ret_code==1){ //task_set;
121  ge("d"+day+"l"+les+"_gtask").innerHTML = html_encode(decodeURIComponent(text));
122  if(text.length>0){
123  ge("d"+day+"l"+les+"_gtask_m").style.display="inline";
124  }else{
125  ge("d"+day+"l"+les+"_gtask_m").style.display="none";
126  }
127  }
128  if(ret_code==2){ //self
129  ge("d"+day+"l"+les+"_stask").innerHTML = html_encode(decodeURIComponent(text));
130  if(text.length>0){
131  ge("d"+day+"l"+les+"_stask_m").style.display="inline";
132  }else{
133  ge("d"+day+"l"+les+"_stask_m").style.display="none";
134  }
135  }
136  if(ret_code==9){ //error;
137  if(sub_code==1){
138  }
139  }
140  UnTip();
141  }
142  function win_absent(day,les){
143  if(current_action>0) return;
144  if(ge("d"+day+"l"+les+"_lid").value==0) return;
145  TagToTip('win_absent', TITLEFONTCOLOR, '#CCFFCC',CLOSEBTN, true, STICKY,
146  true,PADDING,6, TITLE,'Маркер отсутствия',CENTERWINDOW,true,DELAY, 0);
147  ge("wabsent_sched").value = ge("d"+day+"l"+les+"_sched_id").value;
148  ge("wabsent_date").innerHTML = ge("d"+day+"_day_name").innerHTML+" ("+les+
149  " урок)";;
150  ge("wabsent_lesson").innerHTML = ge("d"+day+"l"+les+"_lesname").innerHTML;
151  ge("wabsent_set").disabled=false;
152  ge("wabsent_reset").disabled=false;
153  }
154  function set_absent(type){
155  var uid = ge("wabsent_uid").value;
156  var sched = ge("wabsent_sched").value;
157  var text = encodeURIComponent(ge("wabsent_reason").value);
158  var send_params = "act=absent&sched="+sched+"&uid="+uid+
159  "&type="+type+"&text="+text;
160  current_action = 3;
161  ge("wabsent_set").disabled=true;
162  ge("wabsent_reset").disabled=true;
163  ajax("teacher_diary_ajax.php","post",send_params,set_absent_done);
```

```
164  }
165  function set_absent_done(){
166  current_action = 0;
167  ge("wabsent_set").disabled=false;
168  ge("wabsent_reset").disabled=false;
169  var ret_code,sub_code,day,les,reason;
170  try{
171  eval(req.responseText);
172  }catch(err){
173  alert(req.responseText+err);
174  }
175  if(ret_code==1){ //exist;
176  ge("d"+day+"l"+les+"_areason").innerHTML = "";
177  ge("d"+day+"l"+les+"_tr").style.backgroundColor = "#FFFFFF";
178  }
179  if(ret_code==2){ //absent;
180  ge("d"+day+"l"+les+"_areason").innerHTML = html_encode(
181  decodeURIComponent(reason));
182  ge("d"+day+"l"+les+"_tr").style.backgroundColor = "#C0C4FF";
183  }
184  if(ret_code==9){ //error;
185  if(sub_code==1){
186  }
187  }
188  UnTip();
189  }
190  function win_comment(day){
191  if(current_action>0) return;
192  TagToTip('win_comment', TITLEFONTCOLOR, '#CCFFCC',CLOSEBTN, true, STICKY,
193  true,PADDING,6, TITLE,'Добавление комментария',CENTERWINDOW,true,
194  DELAY, 0);
195  ge("wcomm_date").innerHTML = ge("d"+day+"_day_name").innerHTML;
196  ge("wcomm_day").value = day;
197  ge("wcomm_submit").disabled=false;
198  }
199  function set_comment(){
200  var uid = ge("wcomm_uid").value;
201  var week = ge("wcomm_week").value;
202  var day = ge("wcomm_day").value;
203  var title = encodeURIComponent(ge("wcomm_title").value);
204  var text = encodeURIComponent(ge("wcomm_text").value);
205  var send_params = "act=comment&week="+week+"&uid="+uid+
206  "&day="+day+"&title="+title+"&text="+text;
207  current_action = 4;
208  ge("wcomm_submit").disabled=true;
209  ajax("teacher_diary_ajax.php","post",send_params,set_comm_done);
210  }
211  function set_comm_done(){
212  current_action = 0;
213  ge("wcomm_submit").disabled=false;
214  var ret_code,sub_code,day,comm_id,title,text,owner,oid,time,dpath;
215  try{
```

```
eval(req.responseText);
}catch(err){
alert(req.responseText+err);
}
if(ret_code==1){ //exist;
var nblock =
"<div id=\"c"+comm_id+"_div\" style=\"padding:3px;\">"+
"<div style=\"background-color:#B9EAE4;padding:3px;border:1px solid #1083C7;\">"+
"<div style=\"float:right;width:12px;padding:2px;\">"+
"<img src=\""+decodeURIComponent(dpath)+"images/icon/trash.gif\"
onclick=\"del_comment("+comm_id+")\"/>"+
"</div>"+
"<div style=\"float:right;width:220px;\">"+
html_encode(decodeURIComponent(owner))+
"</div>"+
html_encode(decodeURIComponent(title))+
"</div>"+
"<div style=\"border-bottom:1px solid #1083C7;border-left:1px solid #1083C7;
border-right:1px solid #1083C7;padding:2px;\">"+
html_encode(decodeURIComponent(text))+"</div>"+
"</div>";
ge("d"+day+"_comm_block").innerHTML = nblock+ge("d"+day+
"_comm_block").innerHTML;
}
if(ret_code==9){ //error;
if(sub_code==1){
}
}
UnTip();
}
function del_comment(id){
var send_params = "act=comment_del&cid="+id;
current_action=5;
ajax("teacher_diary_ajax.php","post",send_params,del_comment_done);
}
function del_comment_done(){
current_action = 0;
var ret_code,sub_code,comm_id;
try{
eval(req.responseText);
}catch(err){
alert(req.responseText+err);
}
if(ret_code==1){ //exist;
ge("c"+comm_id+"_div").innerHTML = "";
ge("c"+comm_id+"_div").style.display = "none";
}
if(ret_code==9){ //error;
if(sub_code==1){
}
}
UnTip();
```

```
268     }
```

A.6 Шаблон звіту успішності учня. (report_student_period.tpl)

```
1       {include file='html_header.tpl'}
2       <table width="100%" cellpadding="1" cellspacing="0">
3         <tr>
4           <td width="200" valign="top">
5             {*navibar*}
6            <table width="100%"  border="0" cellpadding="0" cellspacing="0">
7              <tr>
8                <td class="bar-l"></td><td class="bar-m">Навигация</td><td class="bar-r"></td>
9              </tr>
10         </table>
11         <table class="bor1" cellpadding="0" cellspacing="0">
12           <tr>
13             <td>
14               <div><b><small>Информация:</small></b></div>
15               <div style="padding:5px 2px 5px 2px;">
16                 <div style="padding:2px;border:1px solid #D0F1FA;">
17                   {$owner.name1}<br />
18                   {$owner.name2}<br />
19                   {$owner.name3}<br />
20                   Класс: {$owner.grp_grade}{$owner.grp_letter}
21                 </div>
22               </div>
23               <div><b><small>Период обучения:</small></b></div>
24               <div style="padding:5px 2px 5px 2px;">
25                 <select id="period_id"
26        onchange="location='report_student_period.php?uid={$owner.uid}&period='+
27      this.value" style="width:100%">
28                   {section name=i loop=$periods}
29                   <option value="{$periods[i].period}"{if $periods[i].cur==1} selected="true"{/if}>
30                     {$periods[i].name}
31                   </option>
32                   {/section}
33                 </select>
34               </div>
35               <div>
36                 <a href="report_student_period.php?uid={$owner.uid}&period={$cur_period.period}
37      &print=1" target="_blank" style="text-decoration:none;">
38                 <div style="border:1px solid gray;padding:2px;background-color: #D2E8F8;
39      margin-top:2px;cursor: pointer"> 
40                   <img src="{$dpath}images/icon/printer.gif" style="padding-top:2px;" border="0"/>
41                  <div style="display:inline;vertical-align: top;">
42                    Версия для печати 
43                  </div>
44                </div>
45                </a>
46              </div>
47            </td>
48          </tr>
```

```
49      </table>
50      {*navibar end*}
51      </td>
52      <td valign="top" align="center">
53      <h4>Ведомость успеваемости и посещаемости: {$owner.nick}</h4>
54      <table style="text-align:center; border:1px solid gray;" border="0" cellpadding="0"
        cellspacing="0">
55      <tr>
56      <td colspan="1" rowspan="2" style="width:200px;border:1px solid gray;">Предметы</td>
57      <td style="padding:2px;border:1px solid gray;">
58      Получено оценок с начала периода обучения</td>
59      <td colspan="1" rowspan="2" style="padding:2px;border:1px solid gray;">Итоговая</td>
60      <td colspan="1" rowspan="2" style="padding:2px;border:1px solid gray;">Пропуски</td>
61      </tr>
62      <tr>
63      <td>
64      <table border="0" cellpadding="0" cellspacing="0" id="report1">
65      <tr>
66      <td style="width:24px;">"1"</td><td style="width:24px;">"2"</td>
67      <td style="width:24px;">"3"</td><td style="width:24px;">"4"</td>
68      <td style="width:24px;">"5"</td><td style="width:24px;">"6"</td>
69      <td style="width:24px;">"7"</td><td style="width:24px;">"8"</td>
70      <td style="width:24px;">"9"</td><td style="width:24px;">"10"</td>
71      <td style="width:24px;">"11"</td><td style="width:24px;">"12"</td>
72      <td style="width:45px;">Всего</td>
73      </tr>
74      </table>
75      </td>
76      </tr>
77      <tr>
78      <td>
79      <table width="100%" cellspacing="0" cellpadding="0"
80      style="text-align:center;" id="report2">
81      {section name=i loop=$report}
82      {assign var="r" value=$report[i]}
83      <tr>
84      <td style="text-align:left;padding-left:4px;">{$r.name}</td>
85      </tr>
86      {/section}
87      </table>
88      </td>
89      <td>
90      <table cellspacing="0" cellpadding="0" style="text-align:center;" id="report1">
91      {section name=i loop=$report}
92      {assign var="r" value=$report[i]}
93      <tr>
94      <td>{$r.1} </td>
95      <td>{$r.2} </td>
96      <td>{$r.3} </td>
97      <td>{$r.4} </td>
98      <td>{$r.5} </td>
99      <td>{$r.6} </td>
```

```
100     <td>{$r.7} </td>
101     <td>{$r.8} </td>
102     <td>{$r.9} </td>
103     <td>{$r.10} </td>
104     <td>{$r.11} </td>
105     <td>{$r.12} </td>
106     <td style="width:45px;">{$r.total} </td>
107     </tr>
108     {/section}
109     </table>
110     </td>
111     <td>
112     <table width="100%" cellspacing="0" cellpadding="0"
113     style="text-align:center;" id="report2">
114     {section name=i loop=$report}
115     {assign var="r" value=$report[i]}
116     <tr>
117     <td>{$r.result} </td>
118     </tr>
119     {/section}
120     </table>
121     </td>
122     <td>
123     <table width="100%" cellspacing="0" cellpadding="0"
124     style="text-align:center;" id="report2">
125     {section name=i loop=$report}
126     {assign var="r" value=$report[i]}
127     <tr>
128     <td>{$r.absent} </td>
129     </tr>
130     {/section}
131     </table>
132     </td>
133     </tr>
134     </table>
135     <br />
136     </td>
137     </tr>
138     </table>
139     {include file='html_footer.tpl'}
```

ДОДАТОК Б

Комп'ютерна презентація

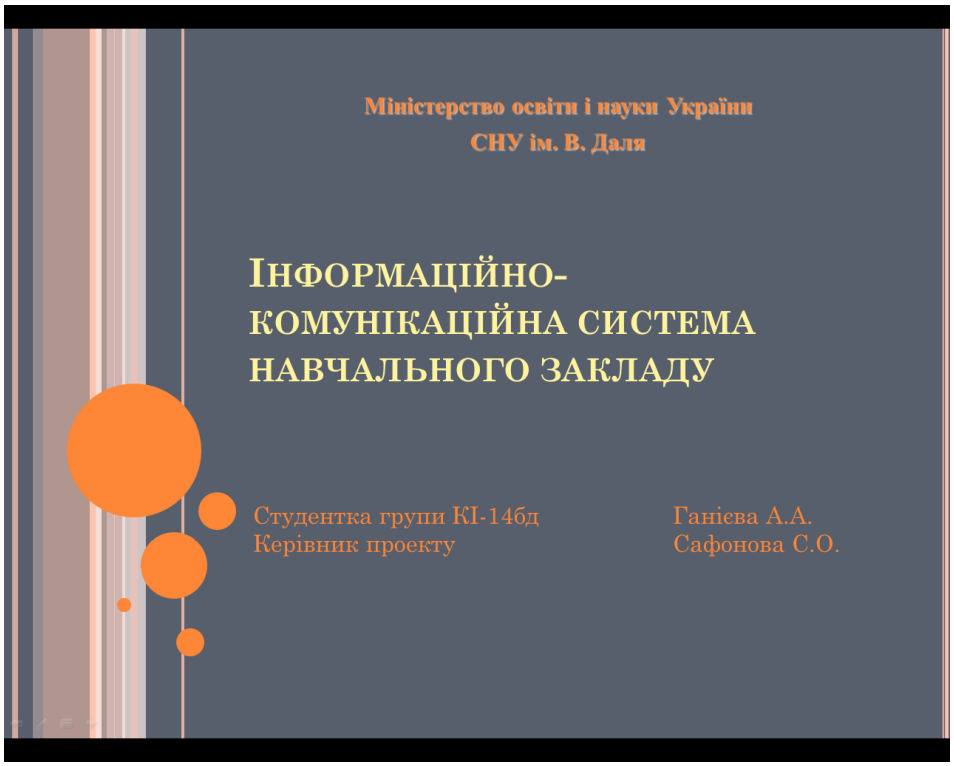


Рисунок Б.1. Слайд № 1

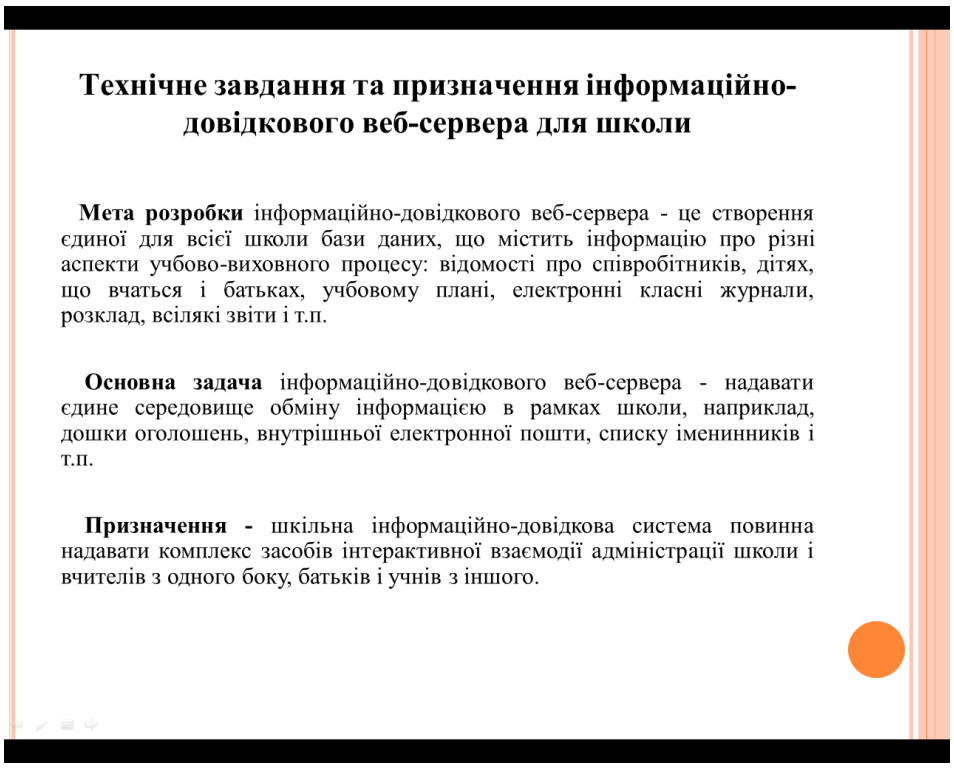


Рисунок Б.2. Слайд № 2

Аналіз і вибір СУБД для реалізації проекту

Сьогодні бази даних (БД) - основа будь-якої крупної інформаційної системи, що зберігає і обробляє всілякі дані.

Вимоги до СУБД для інформаційно - довідкової системи наступні:

- Спосіб доступу до БД – клієнт-сервер.
- Модель даних – реляційна.
- Найменша залежність від операційної системи і програмного забезпечення.

Короткий перелік можливостей MySQL:

- підтримується необмежена кількість користувачів, що одночасно працюють з базою даних;
- кількість рядків в таблицях може досягати 50 млн;
- швидке виконання команд;
- проста і ефективна система безпеки.

В процесі роботи я зробила свій вибір саме на СУБД MySQL.

Допоміжні технології

Також у процесі розробки проекту були використані допоміжні технології. Використання Smarty — компілюючого обробника шаблонів для PHP, дозволило винести теги HTML в окремий файл. Крім того, додати функціональності в генерування HTML сторінок.

На стороні клієнта потрібне було використання технології Javascript і AJAX. Це дозволило зменшити кількість файлів скриптів на сервері, а так само зменшити навантаження на базу даних.

Для управління базами даних і створення таблиць використовується програма Premiumsoft Navicat MySQL.

Premiumsoft Navicat MySQL – це графічна утиліта для роботи з базами даних MySQL.

Рисунок Б.3. Слайд № 3

Розробка інформаційної системи

Інформаційно-довідковий веб-сервер для школи повинен мати два обов'язкові компоненти: база даних для зберігання інформації про користувачів, розклади і щоденники і програма веб-сервер,що відповідає за прийом запитів від користувачів і передачу користувачам, що генеруються php-скріптами HTML сторінок із запрошуваною інформацією. Розробляєма структура інформаційно-довідкового веб-сервера представлена на рисунку 1.

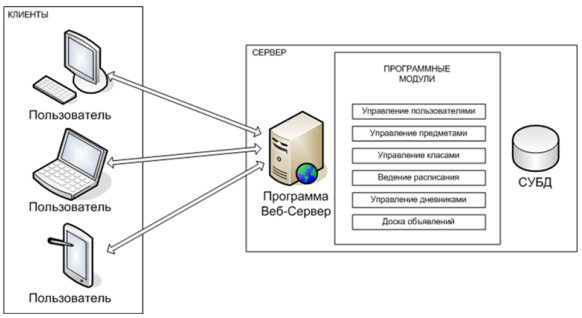


Рисунок 1 Структура інформаційно-довідкового веб-сервера

Рисунок Б.4. Слайд № 4

Дана структура дозволяє всім «клиентам» інформаційної системи одночасно отримувати дані. Принцип взаємодії клієнта з сервером представлен на рисунку 2.

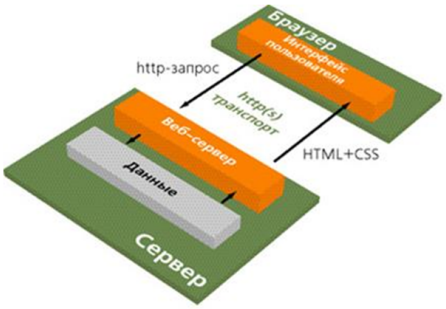


Рисунок 2 – Принцип обміну даними між сервером і клієнтами

Рисунок Б.5. Слайд № 5

ПРОЦЕС СТВОРЕННЯ САЙТУ ІНФОРМАЦІЙНО-ДОВІДКОВОЇ СИСТЕМИ МОЖНА РОЗДІЛИТИ НА НАСТУПНІ ЕТАПИ:

- розробка структури бази даних з використанням CASE- засобів;
- створення таблиць в базі даних відповідно до розробленої структури;
- створення структури окремого компоненту (генеруємі сторінки) системи;
- створення допоміжних функцій і класів, спільних для кожного компоненту системи;
- створення спільного шаблону компоненту системи;
- створення компонентів, що дозволяють управляти користувачами, класами, предметами;
- створення компонентів, що дозволяють управляти розкладом, щоденниками, звітами;
- створення додаткових компонентів системи (повідомлення, дошка оголошень, протоколювання дій в системі).

Рисунок Б.6. Слайд № 6

Розробка структури бази даних

База даних є основою інформаційно-довідкової системи, тому розробка починається саме з її структури.

В результаті розробки була отримана модель бази даних що забезпечує основну функціональність інформаційної системи. Так само була визначена сутність, котра дає можливості ведення календаря подій і обміну повідомленнями.

Кінцева модель бази даних представлена на рисунку 3.

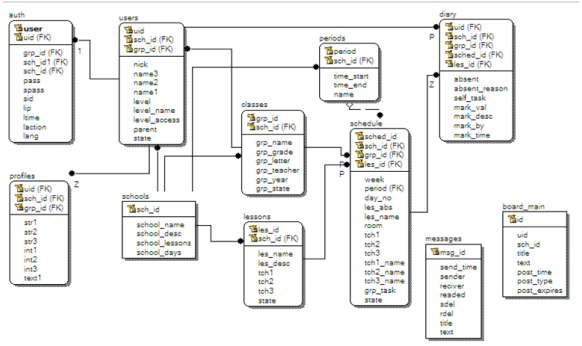
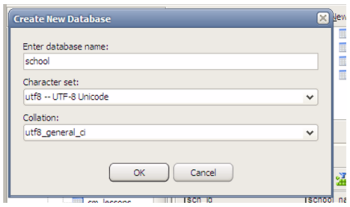


Рисунок 3 – Кінцева модель бази даних інформаційної системи

Рисунок Б.7. Слайд № 7

На основі створеної моделі були зроблені таблиці в базі даних. Для створення таблиць використовувалася програма Premiumsoft Navicat. Результат роботи показан на рисунку 4.

Далі в базі даних створюються таблиці. Редактор таблиць представлений на рисунку 5.



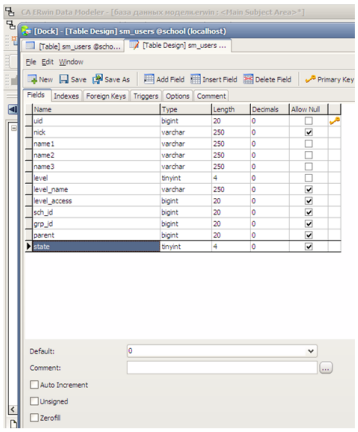


Рисунок 4 – Створення бази даних за допомогою Premiumsoft Navicat

Рисунок 5 – Редактор таблиць в PremiumSoft Navicat

Рисунок Б.8. Слайд № 8

Розробка шаблону компоненту системи

При створенні інформаційної системи для написання PHP скриптів використовується програма PHP Desinger. Основною перевагою використання саме цієї програми є можливість створення проектів. При цьому функції і класи, оголошені в якому-небудь файлі проекту вбудовуються в систему підказок. Таким чином полегшується процес написання коду скриптів. Спільний вид вікна програми з проектом представлений на рисунку 6.

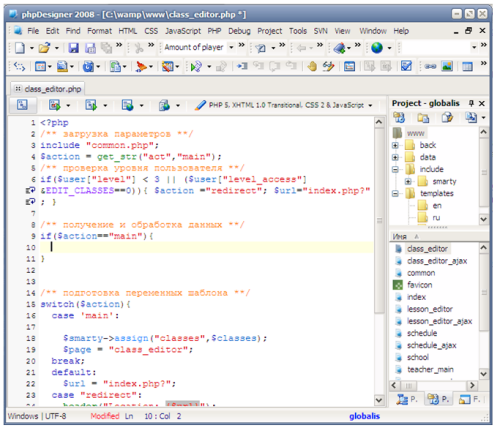


Рисунок 6 – Редагування php скрипта в PHP Designer

Рисунок Б.9. Слайд № 9

Розробка основних компонентів системи

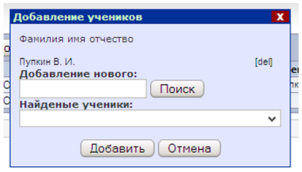


Рисунок 7 – Вікно додавання учня батьку

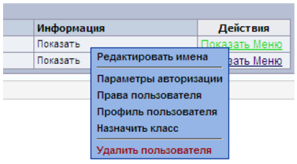


Рисунок 8 – Меню редагування властивостей учня

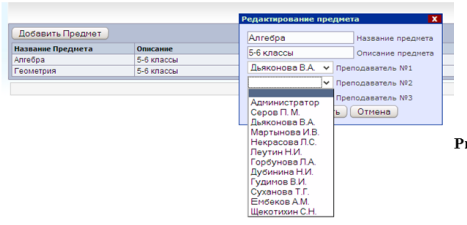


Рисунок 9 – Вікно редагування предмету

Рисунок 10 – Вікно редагування періоду навчання

Рисунок Б.10. Слайд № 10

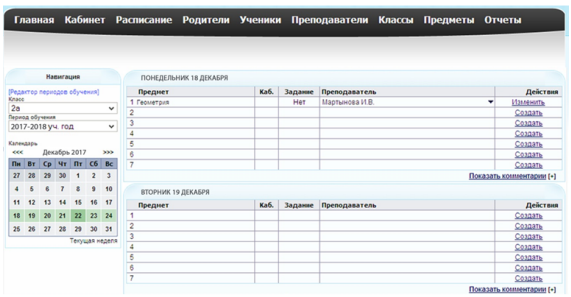


Рисунок 11 – Сторінка редагування розкладу класу

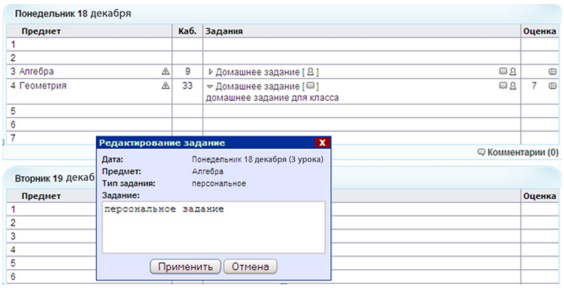


Рисунок 12 – Редагування щоденника учня

Рисунок Б.11. Слайд № 11

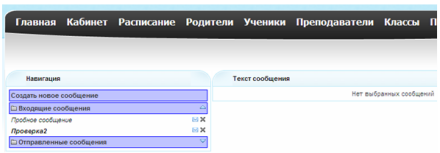


Рисунок 13 – Сторінка управління повідомленнями

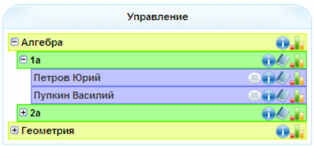


Рисунок 14 – Управління уроками

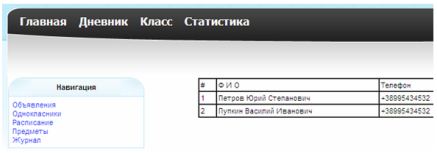


Рисунок 15 – Меню класу і список однокласників

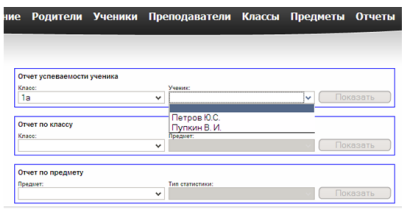


Рисунок 16 – Сторінка вибору типу звіту

Рисунок Б.12. Слайд № 12

Посібник користувача системи

Інформаційно-довідковий веб-сервер є закритою системою. При цьому доступ до даних системи можливий лише після ідентифікації користувача. Для пізнання користувача використовуються унікальний логін і пароль. Для того, щоб увійти до системи користувачеві необхідно вести свої логін і пароль у форму, показану на рисунку 17. Так само після входу розширюється меню користувача, так для викладача вид меню показаний на рисунку 18. Для кожного типа користувачів меню навігації виглядатиме по-різному. Спільною для всіх типів користувачів є сторінка управління повідомленнями представлена на рисунку 19.

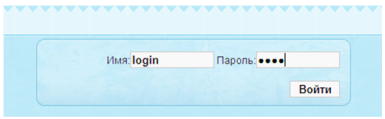


Рисунок 17 – Форма перевірки авторизації

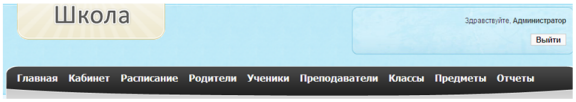


Рисунок 18 – Меню навігації для вчителя

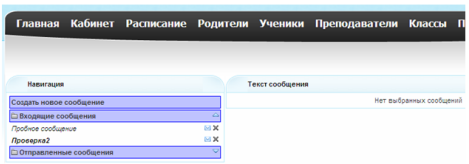


Рисунок 19 – Сторінка управління внутрішньою поштою

Рисунок Б.13. Слайд № 13

Сторінки викладачів

Після входу в систему, вчитель дістає можливість виконувати дії над розкладом, користувачами, класами, і ін. У верхній частині сторінки з'являються посилання на доступні сторінки:

- Кабінет;
- Розклад;
- Батьки;
- Учні;
- Викладачі;
- Класи;
- Предмети;
- Звіти.

Кожне із посилань веде на свій розділ сайту.

Сторінки учня

У інформаційно-довідковій системі учень дістає можливість отримувати поточний розклад і переглядати електронну версію свого щоденника. Окрім цього при вході в систему він отримує повідомлення про нові події (наприклад, виставляння оцінки, зміни в розкладі, новому оголошенні для класу).

Для учня передбачено три основні компоненти меню:

- Щоденник;
- Клас;
- Статистика.

Сторінки батьків

Батьки мають всього один пункт меню – Мої діти. На даній сторінці показані учні, які були призначені даному користувачеві. При цьому на головній сторінці показуються останні дії, що відносяться до призначених учнів.

Рисунок Б.14. Слайд № 14

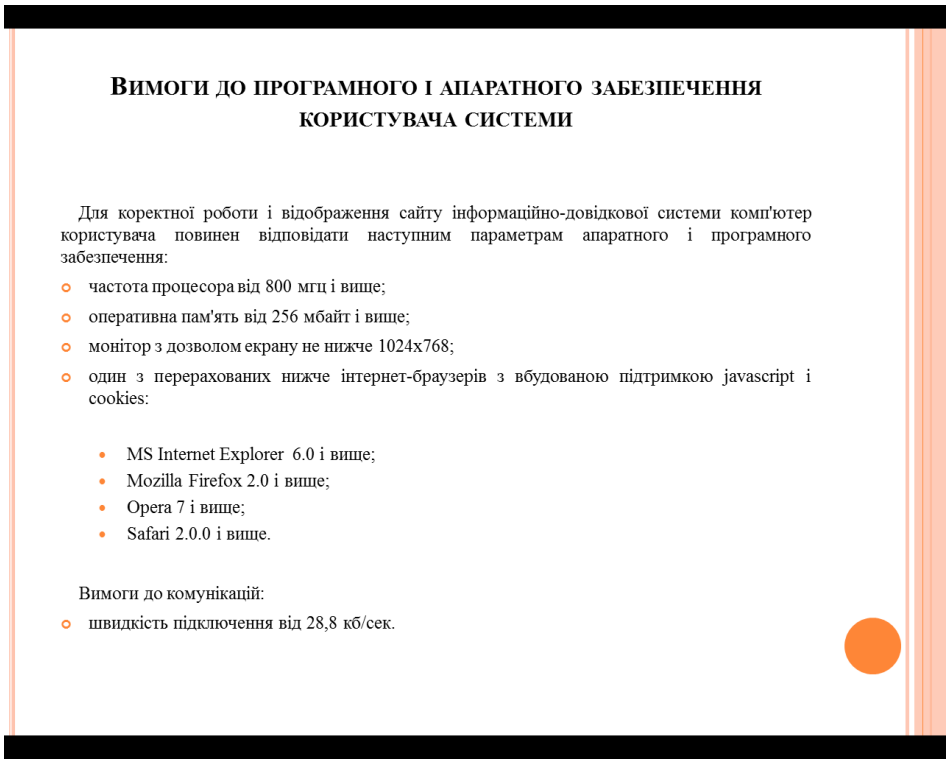


Рисунок Б.15. Слайд № 15

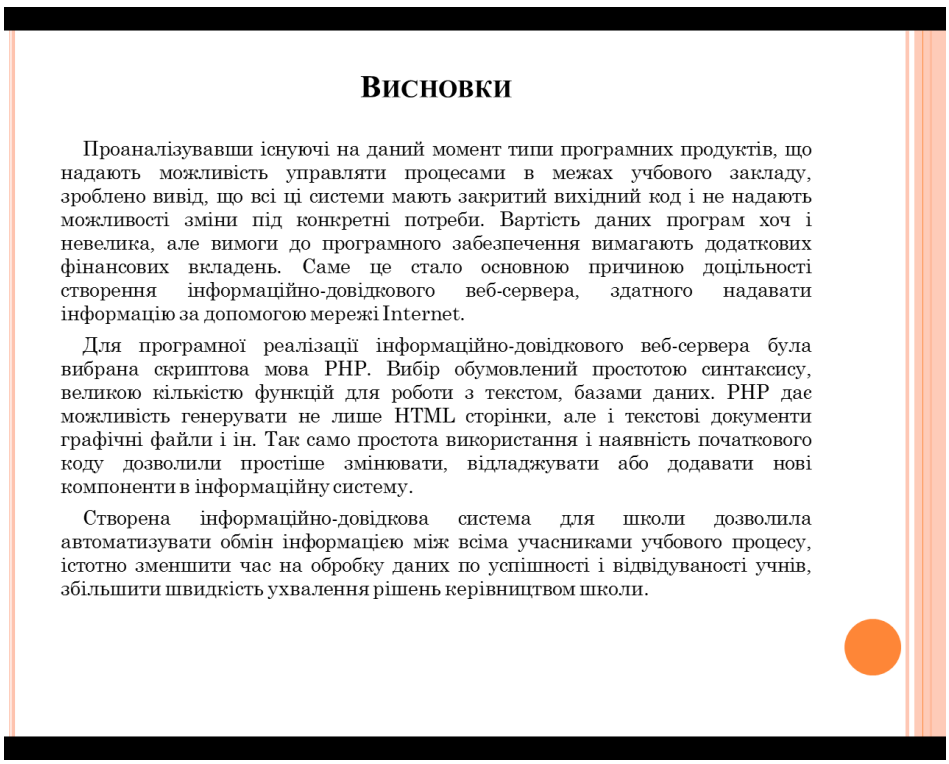


Рисунок Б.16. Слайд № 16