СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Навчально-науковий інститут (факультет) інформаційних технологій та електроніки

Кафедра інформаційних технологій та програмування

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної випускної роботи

освітній ступінь бакалавр
спеціальність 121 „Інженерія програмного забезпечення”
(шифр і назва спеціальності)

на тему « Веб-сайт для комп’ютерної гри з використанням JavaScript фреймворку»

Виконав: студент групи ІПЗ-20д _____ А.Р. Павлов
(підпис) (ініціали і прізвище)

Керівник _____ В.О. Лифар
(підпис) (ініціали і прізвище)

Завідувач кафедри _____ О.І. Захожай
(підпис) (ініціали і прізвище)

Рецензент Ратов Д.В.

Київ – 2024

СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Навчально-науковий інститут (факультет) інформаційних технологій та електроніки
Кафедра інформаційних технологій та програмування

Освітній ступінь бакалавр
спеціальність 121 „Інженерія програмного забезпечення”
(шифр і назва спеціальності)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Захожай О.І.
“___” _____ 2024 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ ВИПУСКНУ РОБОТУ СТУДЕНТУ

Павлов Артур Романович
(прізвище, ім’я, по батькові)

1. Тема роботи: Веб-сайт для комп’ютерної гри з використанням JavaScript фреймворку
Керівник роботи Лифар Володимир Олексійович, д.т.н., доцент
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
Затверджений наказом університету від “06” травня 2024 року №171/15.15-С

2. Строк подання студентом роботи: 08.06.2024р.
3. Вихідні дані до роботи: Об’єктом даної роботи є процес створення веб-сайту для комп’ютерної гри

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити): Вступ. Аналітичний огляд, з висвітленням наступних питань: Основні популярні JavaScript фреймворки, плюси та мінуси кожного фреймворку, обгрунтування вибору фреймворка. Основна частина, в якій висвітлити інформаційну та функціональну модель додатку, етап створення додатку та тестування. Висновки. Перелік використаних джерел.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслеників)

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 30.03.2024р

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання кваліфікаційної випускної роботи	Строк виконання етапів	Примітка
1	Одержання завдання на виконання роботи	30.03.24	виконано
2	Укладання і погодження з керівником плану і етапів виконання роботи	06.04.24	виконано
3	Узагальнення даних літературних джерел, укладання розділу «Аналіз предметної галузі»	13.04.24	виконано
4	Аналіз шляхів виконання завдання. Вибір і погодження з керівником оптимального шляху	20.04.24	виконано
5	Укладання та тестування програмного продукту	27.04.24	виконано
6	Укладання, оформлення та погодження пояснювальної записки з керівником	07.06.24	виконано
7	Здача готової пояснювальної записки на кафедру	08.06.24	виконано
8	Укладання доповіді і презентації	12.06.24	виконано

Студент _____ А.Р. Павлов
підпис (ініціали і прізвище)

Керівник роботи _____ В.О. Лифар
підпис (ініціали і прізвище)

ЛИСТ ПОГОДЖЕННЯ І ОЦІНЮВАННЯ

дипломної роботи студента гр. ІПЗ-20д Павлов А.Р.

Науковий керівник:

Професор каф.ІТП, д.т.н. _____ Лифар В.О.

Оцінка наукового керівника: _____

Рецензент: Ратов Д.В., к.т.н., доцент каф. ІТП СНУ ім. В.Даля
ПІБ, місто роботи, посада

Оцінка рецензента: _____

Кінцева оцінка за результатами захисту:

_____

Голова ЕК
Професор кафедри ІТП
д.т.н. _____ Меняйленко О.С.

ЗМІСТ

Вступ

На сьогодні часто можна побачити що на сайті в інтернеті використовуються технології які допомагають написати SPA додаток. Односторінковий додаток (англ. односторінковий додаток, SPA) — це веб-додаток або веб-сайт, що використовує єдиний HTML-документ як оболонку для всіх веб-сторінок і організує взаємодію з користувачем через методи, що підвантажуються HTML, CSS, JavaScript, зазвичай, через AJAX. ОП нагадують рідні (рідні) додатки, з тією лише різницею, які виконуються у браузері, а не у власному процесі вбудованої системи.

Фреймворк допомагає підвищити перше враження від сайта. Через те що сайт завантажується всього 1 раз, користувачу не треба чекати для того щоб побачити якусь інформацію на своїй сторінці, це тому що фреймворки використовують Virtual DOM.

Що таке DOM

Перед тим, як ми почнемо вникати в те, що являє собою DOM віртуальний, давайте трохи поговоримо про те, чим є DOM реальний.

DOM (абревіатура від Document Object Model) - спосіб представлення структурного документа за допомогою об'єктів. Це кросплатформна та мовно-незалежна угода для подання та взаємодії з даними в HTML, XML і т.д.

Веб-браузери обробляють компоненти DOM, і ми можемо взаємодіяти з ними, використовуючи JavaScript та CSS. Ми можемо працювати з вузлами документа, змінювати їх дані, видаляти та вставляти нові вузли. У наші дні DOM API є практично кросплатформним та кросбраузерним.

Проблема DOM

Головна проблема DOM — він ніколи не був розрахований для створення динамічного інтерфейсу користувача (UI). Ми можемо працювати

з ним, використовуючи JavaScript та бібліотеки на зразок jQuery, але їх використання не вирішує проблем із продуктивністю.

Подивіться на сучасні соціальні мережі, такі як Twitter, Facebook чи Pinterest.

Після невеликого скролінгу ми матимемо десятки тисяч DOM-вузлів, ефективно взаємодіяти з якими — завдання не з легких.

Наприклад, спробуйте перемістити 1000 div-блоків на 5 пікселів вліво.

Це може зайняти більше секунди – це дуже багато для сучасного інтернету. Ви можете оптимізувати скрипт і використовувати деякі прийоми, але в результаті це викличе лише головний біль під час роботи з величезними сторінками та динамічним UI.

Чи можемо ми вирішити цю проблему? Схоже, що можемо.

В даний час W3C працює над новим стандартом Shadow DOM.

Shadow DOM - це робоча чернетка стандарту W3C. Специфікація, що описує метод поєднання кількох DOM-дерев в одну ієрархію і як ці дерева взаємодіють один з одним у межах документа, що дозволяє краще скомпонувати DOM.

Інший варіант полягає у використанні підходу з Virtual DOM.

Virtual DOM

Замість взаємодіяти з DOM безпосередньо, ми працюємо з його легковажною копією. Ми можемо вносити зміни в копію, виходячи з наших потреб, а потім застосовувати зміни до реального DOM.

При цьому відбувається порівняння DOM-дерева з його віртуальною копією, визначається різниця та запускається перемальовка того, що було змінено.

Такий підхід працює швидше, тому що не включає всі великовагові частини реального DOM.

1.Популярні JavaScript фреймворки

1.1 Angular

Angular - це кросплатформний фреймворк, який дотримується MVC-шаблону проектування і заохочує слабкий зв'язок між поданням, даними та логікою компонентів (складових частин програми).

Angular переносить на клієнтську сторону частину серверної служби. Отже, зменшується навантаження на сервер і веб-додаток стає легшим.1

Завдяки TypeScript код проектів стає чистим, зручним для розуміння розробників і містить менше помилок.

Сильні сторони Angular: чудова документація, підтримка Google, величезний набір інструментів для розробки (Material UI, CLI і т.д.)

Однією із слабких сторін є високий поріг входження до проектів для розробників, тому що, наприклад, потрібно знати TypeScript. Отже, це ускладнює розробку проектів, особливо, якщо проект передається від однієї команди до іншої.

Друга проблема Angular – дуже частий реліз нових версій. У червні 2019 року вийшла вже восьма версія фреймворку. Цей факт також говорить про те, що проекти на Angular складніше підтримувати.

Рисунок 1.1 Доля трафіка на документацію Angular

1.2 React.js

React - це JavaScript бібліотека з відкритим вихідним кодом для розробки інтерфейсів користувача. Вперше React був використаний у 2011 році у стрічці Facebook, потім у 2012 у стрічці Instagram.

React досі розробляється та підтримується Facebook та Instagram. React може використовуватися не тільки для браузерних веб-застосунків, але також для мобільних додатків. Мета React – надати високу швидкість, простоту та працездатність додатків на різних платформах.

Але React це не повноцінний фреймворк, а бібліотека функцій. Щоб використовувати його як фреймворк, необхідно додатково підключати сторонні бібліотеки.

Сильні сторони React: швидкість роботи, легковагість, кросплатформність та велике ком'юніті.

Слабкою стороною є той факт, що для повноцінної роботи потрібні сторонні Javascript-бібліотеки, що ускладнює процес розробки. Другий мінус бібліотеки - це відсутність дотримання стандартів у написанні коду на HTML і CSS, яке є, наприклад, Angular і Vue.js.

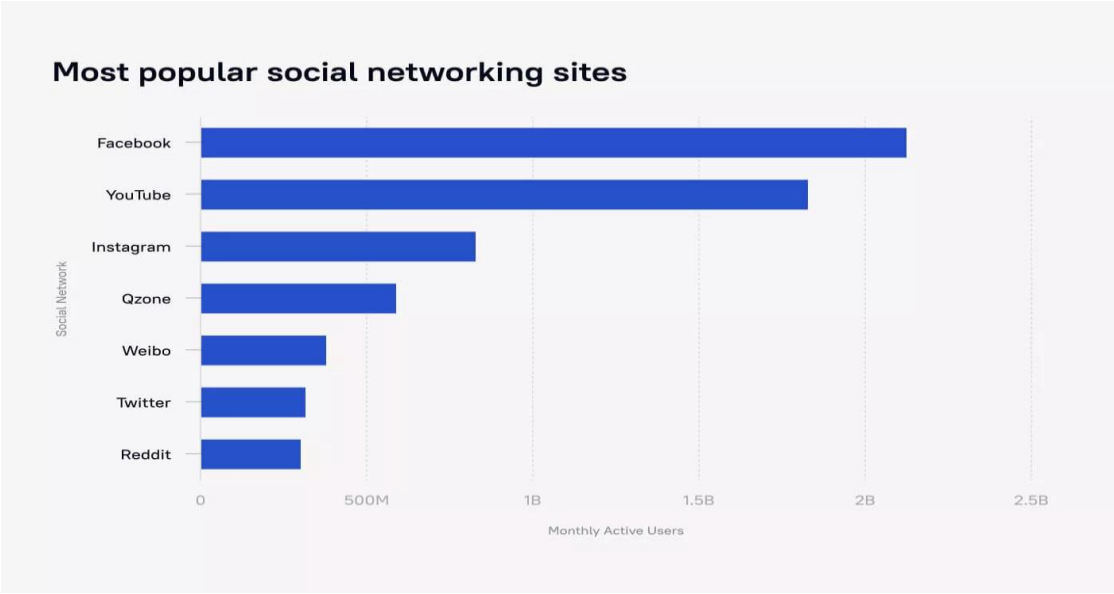

Рисунок 1.2 Список популярних сайтів

1.3 Vue.js

Vue - це прогресивний фреймворк для створення інтерфейсів користувача. Vue створений придатним для поступового впровадження, на відміну від Angular чи React. Це означає, що впроваджувати цей фреймворк можна поетапно з певних сторінок, що значно спрощує розробку.

Автором Vue.js є Іван Йу. Vue широко використовується серед китайських компаній, наприклад Alibaba, Baidu, Xiaomi та ін. Нещодавно система управління репозиторіями GitLab теж перейшла на Vue.js.

Vue насамперед вирішує завдання рівня представлення (view), що спрощує інтеграцію коїться з іншими бібліотеками та існуючими проектами.

Vue увібрав у себе найкращі сторони Angular та React: швидкість, легковажність, можливість підтримки таких технологій, як TypeScript та JSX. Але, при цьому, Vue залишився вірним стандартам написання коду на HTML і CSS, що полегшує процес розробки та підтримки проекту.

Таким чином, для будь-яких розробників, які знають основи фронтенд-технологій, не буде проблемою розробити або взяти на підтримку проекти на Vue.

1.4 Що ж вибрати?

Кожен проект по-своєму унікальний і потребує детального аналізу та оцінки перед стартом. Але є деякі ключові моменти, якими можна заздалегідь оцінити вибір певної технології для розробки:

Швидкість та легкість розробки та підтримки проекту

Швидкість розробки залежить від технологій на вашому проекті. Якщо у вашому проекті вже використовуються такі технології, як TypeScript або JSX, вам краще підійдуть Angular і React відповідно.

Але якщо ви починаєте проект з нуля, то, як було сказано вище, завдяки дотримання стандартів найкраще вам підійде Vue.js.

Тренди

Всі три фреймворки є популярними і найближчими роками займатимуть провідні місця на ринку розробки. Однак Vue стрімко набирає популярності, за ним слідує React. Тому, обравши будь-який із них, ви не помилитеся. Нижче наведено графік популярності фреймворків на Гітхабі за кількістю лайків за останні кілька років:

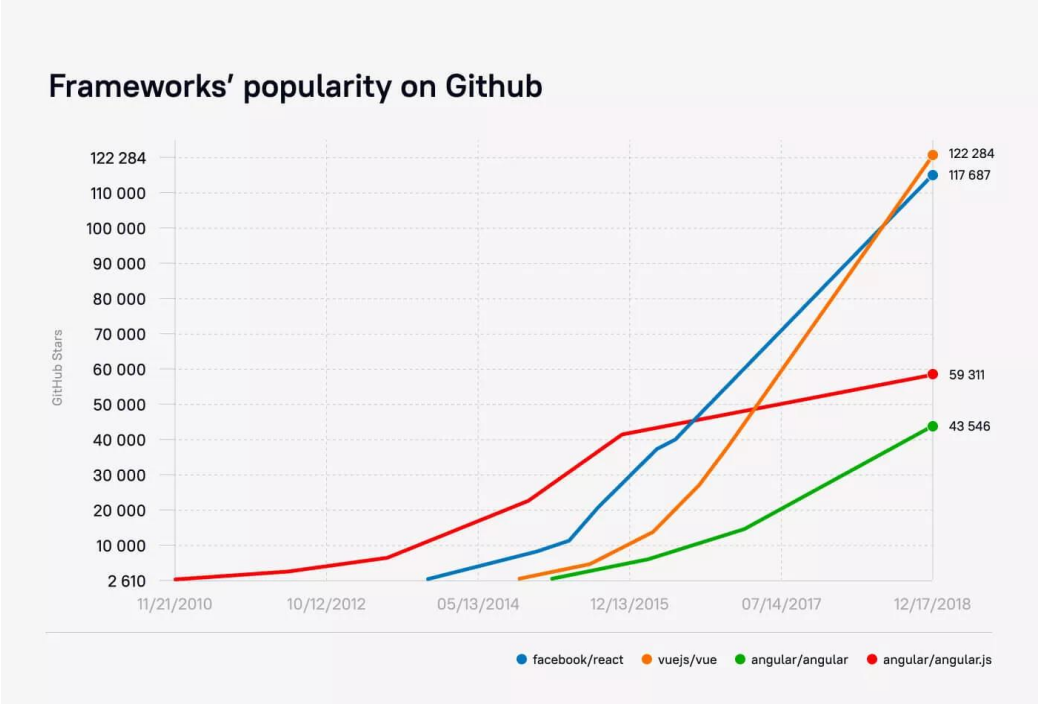

Рисунок 1.3 Графік використання фреймворків

2. Плюси та мінуси популярних JavaScript фреймворків

2.1 Angular 2+

Сильні сторони

Головна перевага Angular 2+ – це його популярність. Можна говорити про те, що з ним пов'язане ім'я компанії Google, і це впливає на те, як його сприймають. Angular 1 швидко став популярним, оскільки ті, хто прийшов з інших середовищ розробки, виявили в ньому знайомий шаблон MVC для створення односторінкових додатків. Після модернізації Angular 1 та перепроектування деяких частин фреймворку, Angular 2+ буквально вистрілив. Вражає кількість тренінгів щодо нього, офіційних та неофіційних. На ринку є серйозна потреба в Angular-розробниках. Крім того, це один з небагатьох фреймворків, розглянутих у цьому матеріалі, у якого є офіційний набір багатих можливостями компонентів для створення інтерфейсів користувача.

Слабкі сторони та можливі складнощі при впровадженні

Ми вважаємо, що Angular зосереджений на створенні інтерфейсів для користувача односторінкових додатків і не відповідає потребам розробників більших проектів. Це може призвести до складності підтримки проектів, якщо базові принципи, на яких вони засновані, не були чітко сформульовані на початку їх розробки. На практиці розробникам доводиться вдаватися до чудес винахідливості для того, щоб змусити додаток на Angular робити те, що не є частиною фреймворку. Це, крім того, знижує інтерес розробників до TypeScript, на якому написано фреймворк.

2.2 React + Redux

Сильні сторони

Основна перевага React і Redux полягає в їхній порівняльній простоті і тому, що вони спрямовані на вирішення одного завдання, на розробку інтерфейсів. Якщо поставити собі за мету знайти щось, що робить щось одне,

але робить це добре, можна сказати, що обидві бібліотеки чудово роблять те, чого від них чекають. У той час, як для когось підхід, пов'язаний з використанням контейнера стану може здатися незнайомим, більшість розробників можуть легко розібратися в цій концепції та зрозуміти переваги архітектури, заснованої на односпрямованому потоці даних, і те, як такий підхід може спростити додатки зі складними користувальницькими інтерфейсами.

Слабкі сторони та можливі складнощі при впровадженні

Найбільші мінуси React та Redux полягають не в особливостях реалізації того, що вони вміють, а в тому, чого вони не можуть. Для того, щоб створити складну веб-додаток, вам знадобиться багато інших технологій. Як тільки ви відійдете від основних функцій React, Redux та пари інших бібліотек, ви зіткнетеся з безліччю думок про «правильні інструменти», з незліченною кількістю рішень і шаблонів, які іноді легко інтегрувати в програму, а іноді ні.

Отже, оскільки React і Redux — бібліотеки, які зосереджені на вирішенні вузького кола спеціалізованих завдань, недосвідчені команди можуть дуже легко зробити на їх основі щось непідтримуване, не знаючи, що рішення, які вони приймають, ведуть до погіршення продуктивності веб-додатків. або до помилок. Навіть досвідчені розробники можуть зіткнутися з тим, що недолік чіткого архітектурного планування рішення або суворих правил на початку розробки можуть дуже неприємно позначитися на проекті в майбутньому.

Легко обдурити себе уявною економією часу та ресурсів, яка полягає в тому, що використання React та Redux у всій організації пом'якшить проблеми з ефективністю розробки. Без ретельно опрацьованих угод та стандартизації інших бібліотек та шаблонів, перехід на React і Redux схожий

на заяву: «Ми переходимо на JavaScript для того, щоб писати програми та підвищити ефективність роботи».

Vue.js

Сильні сторони

Ймовірно, головний плюс цього фреймворку полягає у можливості його поступового впровадження. Vue відрізняється зрозумілою та раціональною архітектурою, яку нескладно освоїти та просто застосовувати на практиці.

Існує згуртована спільнота ентузіастів та сторонні проекти, які роблять Vue.js ще цікавішими. Крім того, різні розробки, орієнтовані на Vue, досить просто поєднувати в більш складних рішеннях при створенні нових проектів.

Слабкі сторони та можливі складнощі при впровадженні

Бажання якось вивернутися між ідеями програм, заснованих на шаблоні MVC, та додатках, заснованих на контейнерах стану, може заплутати. Здається, що розробники фреймворку мають прагнення зробити все якісно, при цьому не даючи переваги одному шаблону розробки додатків перед іншим. Нам здається, що це, як мінімум, збиває з пантелику тих, хто шукає в Vue.js платформу для повномасштабного веб-рішення, і може призвести до застосування різних шаблонів, що у результаті ускладнить підтримку програми.

Одна з основних проблем Vue.js полягає в тому, що проект залежить від однієї людини. Зрозуміло, що інші фреймворки теж залежать від когось, але зазвичай це — організації. Навколо Vue.js склалося широке співтовариство, тут є з безліччю інноваційних додаткових проектів, але технологія ядра повністю лежить на плечах єдиного розробника.

3. Чому я вибрав Vue для свого проєкту

Vue.js - це JavaScript-фреймворк модель-подання-подання (MVVM) для створення інтерфейсів користувача (UI) і односторінкових додатків. Творцем Vue.js є колишній інженер Google Еван Ю, який швидко розчарувався у використанні AngularJS у проектах Google. Він вирішив отримати кращі риси Angular і створити щось неймовірно легке.

Vue.js – це легкий інтерфейсний JavaScript-фреймворк.

Архітектура MVVM Vue.js забезпечує відділення логіки уявлення від бізнес-логіки, яка представляє уявлення та модель відповідно. Потім модель уявлення діє як посередник, готуючи об'єкти даних до подання. Але, крім архітектури, найбільш примітною особливістю Vue.js є директиви. Директиви розширюють HTML за допомогою HTML-атрибутів, розширюючи функціональні можливості HTML-додатків. Ці директиви є вбудованими, але вони також можуть бути визначені користувачем. По суті вони дозволяють маніпулювати об'єктною моделлю документа (DOM). DOM – це свого роду інтерфейс, зокрема API для документів HTML та XML, який інструктує машини про те, як структурувати текст. Концепція директив існує як в Angular, так і Vue.js, але ви можете підключити Vue.js до будь-якої частини серверного додатка і вручну налаштувати більш інтерактивний досвід для споживачів. Коротше кажучи, ця функція дозволяє елементам HTML інкапсулювати повторно використовуваний код. Фреймворки JavaScript, такі як Vue.js, сприяють більшій організації, коли доходить до розробки інтерфейсу. Vue.js може взяти веб-сторінку і розбити її на компоненти, що повторно використовуються, кожен з яких має свої власні елементи HTML, CSS і JavaScript. Інші розумні інструменти Vue.js включають Vuex та Vue-Router. Вони працюють з основним програмуванням Vue.js, надаючи наступне:

Керування станом, коли елементи керування інтерфейсу користувача, такі як текстові поля, кнопки OK і т. д., можуть вимагати управління за межами поточної сторінки, яку відвідує користувач (Vuex).

Маршрутизація — процес, який відбувається, коли вам потрібно синхронізувати URL-адресу з поданнями у вашій програмі (Vue-Router).

3.1 Навіщо використовується Vue.js?

Vue.js — універсальна мова, тому ви можете використовувати її для різних цілей.

По-перше, Vue.js добре підходить для будь-яких проектів, в яких задіяні HTML, CSS та JavaScript. І саме тому Vue.js в основному використовується для створення інтерфейсів і будь-яких веб-розробок. Але оскільки Vue.js легкий, це також добрий вибір для швидкого прототипування. Скажімо, ви хотіли створити мінімально життєздатний продукт (MVP) з тієї чи іншої причини. Vue.js зробив би це в одну мить.

Односторінкові програми (SPA) також знаходяться у сфері компетенції Vue.js. У SPA є лише одна сторінка для перегляду інформації. Користувачі отримують доступ до нової інформації, оскільки сайт динамічно переписує сторінку при взаємодії з користувачем. Як і React Native, Vue.js дозволяє розробляти нативні мобільні програми. Розробники можуть використовувати NativeScript для створення мобільних додатків на Android та iOS із загальним кодом JavaScript. Може бути корисно знати, що глобальні веб-сайти, що використовують Vue.js, включають такі, як Facebook, Netflix і Google, що демонструє, наскільки потужна ця структура.

3.2 Чому Vue.js є популярним?

Розробники використовують Vue.js з низки причин. Зрозуміло, що Vue.js є популярним, але його справжня привабливість полягає в його технічній проникливості. Ось деякі особливості, що відображають можливості Vue.js:

Легкий дизайн

Vue.js називають прогресивним, тому що він призначений для поступового впровадження. Це означає, що програмування на Vue.js означає починати з малого та масштабувати за власним бажанням. В результаті Vue.js досить простий і не відрізняється складністю більших фреймворків. Плюс варто відзначити, що розмір фреймворку Vue.js менше 21 кілобайта. Так що будьте впевнені, вам не доведеться мати справу із затримкою важкого програмного забезпечення на ваших робочих машинах.

Двостороння прив'язка даних

Маніпуляції з DOM – це двонаправлена подія. Те, що відбувається у виставі, впливає на модель, і навпаки. Хоча потік даних в одному напрямку може бути менш складним на мозковому рівні, одностороння прив'язка даних перешкоджає процесу розробки. Зміни моделі відбиваються на уявленні, але з навпаки. Отже, внесення змін до інтерфейсу користувача вимагає додаткової роботи.

DOM-рендерінг

Vue.js – постачальник віртуального DOM. (Vue.js безкоштовний і має відкритий вихідний код, тому плата не стягується) Віртуальний DOM - це засіб зберігання уявлення інтерфейсу користувача в пам'яті. Через узгодження віртуальний DOM синхронізує зміни із «справжнім» DOM пізніше. Віртуальні DOM вигідні, тому що негайне оновлення «реальної» DOM може бути стомлюючим та повільним. Використовуючи віртуальний DOM, розробники можуть бачити зміни, які вони вносять у виставу, фактично не надсилаючи ці зміни для постійного рендерингу.

Компонентний

Кожна частина програми або веб-сторінки Vue.js діє як окремий компонент. Перевага тут у тому, що ви можете повторно використовувати компоненти на власний розсуд. В цілому, такий підхід до розробки

забезпечує кращу читаність коду, оскільки буде менше безладу. Простіше модульне тестування є додатковою перевагою, тому що ви можете перевірити, як найдрібніші частини програми працюють самі по собі.

Vue.js має велику та активну спільноту

І останнє, але не менш важливе: ця спільнота заповнена талановитими розробниками Vue.js, а це означає, що вам легко знайти таланти для вашої команди розробників програмного забезпечення.

3.3 Vue.js проти інших фреймворків JavaScript

Найбільшими конкурентами Vue.js є інші фреймворки JavaScript, такі як React та Angular.

Vue.js проти Angular

Angluar — це фреймворк для веб-застосунків на основі TypeScript . Він кроссплатформенний, що означає, що ви можете повторно використовувати код TypeScript для Інтернету, мобільного Інтернету, власного мобільного та настільного використання. Фреймворк також обіцяє швидкість, продуктивність та корисні інструменти, такі як декларативні шаблони та відгуки, специфічні для Angular, для його редакторів та інтегрованих середовищ розробки (IDE). Але в порівнянні з Vue.js досвідчені розробники вважають, що Angular:

Важко вчитися

Схильний до жаргонної документації

Vue.js проти React

React — це зовнішня бібліотека JavaScript для створення інтерфейсів користувача. Його компонентний дизайн дозволяє розробникам легко підключати компоненти інтерфейсу користувача і створювати складні інтерфейси користувача без необхідності керування станом. JSX також є основним аспектом React. Подібно до XML, JSX поєднує HTML, CSS і

JavaScript в одну мову. Незважаючи на все це, розробники, які віддають перевагу Vue.js, думають:

Використання JSX робить код менш зручним для супроводу

Відсутні важливі функції, такі як навігація та керування станом.

Досі Vue.js успішно замовчував проблеми інших фреймворків JavaScript. Для розчарованих розробників це має стати привабливою причиною переходу на Vue.js. Компанії, як правило, повинні довіряти технічним лідерам у своїй організації та враховувати їхній вибір технологій.

4 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

4.1 Вибір і опис методології проектування

При створенні веб-сайту для комп'ютерної гри з використанням JavaScript фреймворку, була обрана методологія Agile. Agile є найбільш популярною методологією у сфері програмування, оскільки вона сприяє гнучкості, швидкості та постійному вдосконаленню продукту. Основними принципами методології Agile, які були використані при проектуванні веб-сайту, є постійне вдосконалення функціоналу, гнучке реагування на зміни вимог та постійне вдосконалення продукту через ітераційний процес розробки.

Для реалізації методології Agile у процесі проектування веб-сайту було обрано SCRUM як фреймворк для управління проектом. SCRUM дозволяє розбити процес розробки на невеликі ітерації, які називаються спринтами, кожен з яких має чітко визначені завдання та терміни виконання. Крім того, при проектуванні веб-сайту було застосовано принципи Test-Driven Development (TDD), що дозволяє писати тести перед реалізацією функціоналу, щоб забезпечити його коректну роботу та покриття коду тестами. Використання методології Agile та фреймворку SCRUM дозволило забезпечити ефективне управління проектом, швидку адаптацію до змін та

постійне вдосконалення веб-сайту для комп'ютерної гри з використанням JavaScript фреймворку.

Agile - класифікація методології

Методологія Agile є підхід до управління проектами, який використовує для організації проектів чотири цінності та 12 принципів.

У маніфесті Agile (Agile Manifesto) заявлено такі чотири цінності:

1. Люди та взаємодія важливіші за процеси та інструменти.
2. Працюючий продукт важливіший за вичерпну документацію.
3. Співпраця із замовником важливіша за узгодження умов контракту.
4. Готовність до змін важливіша за проходження початкового плану.

Метод Agile полягає у розбитті планування та виконання проектів на ітерації (спринти), що дозволяє безперервно адаптувати та вдосконалювати план, обсяг робіт та структуру протягом усього проекту. Agile-проекти вимагають ітераційного підходу, який забезпечує регулярне та послідовне надання замовникам чи клієнтам дедалі більш працездатних продуктів. Цей інноваційний метод управління проектами гарантує вашій проектній групі можливість щоразу випускати конкретні продукти без затримок через зміни та уточнення вимог.

Гнучка методологія передбачає активну участь клієнтів та часту перевірку ходу робіт як з боку проектної групи, так і з боку клієнта.

Для керування Agile-проектами можна використовувати різні фреймворки. До найпопулярніших відносяться:

- Scrum
- Канбан
- Екстремальне програмування
- DSDM

Що таке Scrum

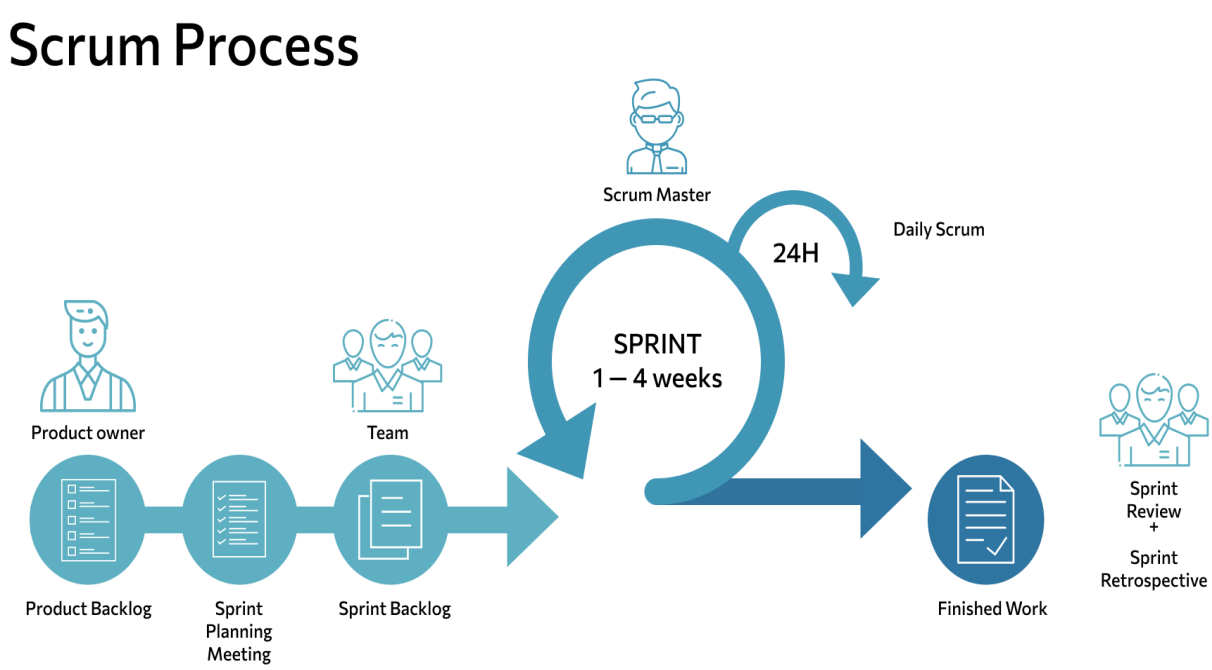

Рис. 4.1 – етапи проекту при використанні методики Scrum

Scrum – це методика гнучкого управління проектами, яка допомагає командам структурувати свою роботу та керувати нею за допомогою набору цінностей, принципів та практик. Як спортивна команда готується до вирішальної гри (до речі, scrum — англ. «битва», елемент гри в регбі), так і команда співробітників компанії повинна отримувати уроки з набутого досвіду, освоювати принципи самоорганізації, працюючи над вирішенням проблеми, та аналізувати свої успіхи та провали, щоб постійно вдосконалюватись. Scrum сприяє цьому.

Методологію Scrum найчастіше застосовують команди розробників додатків, але принципи та досвід її використання можна застосувати до командної роботи будь-якого роду. Це одна із причин такої популярності Scrum. Scrum часто представляють як платформу для управління проектами методології agile. Учасники команди Scrum проводять збори, використовують спеціальні інструменти та приймають на себе особливі ролі, щоб організувати роботу та управляти.

4.2 Проектування структури системи

Проєктува́ння — це комплекс робіт який складається з пошуку, досліджень, розрахунків та розрахування з метою отримання опису достатнього для створення нового об'єкту або виробу, його реконструкції, модернізації, що відповідає заданим вимогам.

у техніці — розробка проєктної, конструкторської та іншої технічної документації, призначеної для забезпечення будівництва, створення нових видів та зразків.

В процесі проєктування виконуються технічні та економічні розрахунки, схеми, графіки, пояснювальні записки, кошториси, калькуляції та описи.

У психології — сукупність і послідовність розумових та психомоторних дій, внаслідок чого створюються образні схеми або знакові системи.

Стадії проєктування

Стадії проектування регламентовані стандартами ГОСТ 2.103-2013 та ГОСТ Р 15.301-2016. Послідовність виконання всіх стадій утворює офіційну структуру процесу розробки проектної документації, яка зазвичай використовується при офіційних взаєминах між замовником і виконавцем або між співвиконавцями робіт. Сама документація необхідна для звіту перед замовником про виконану роботу, можливість перевірки чи повторення розробок іншими виконавцями, підготовки виробництва та обслуговування виробу під час експлуатації.

Стадії створення інших систем регламентуються своїми стандартами, наприклад, для автоматизованих систем - ГОСТ 34601-90 .

Структура встановлює стадії розробки конструкторської документації на вироби всіх галузей промисловості та етапи виконання робіт у кожній стадії, тобто склад документації та види робіт, що допомагає відповісти на

запитання «Що потрібно робити?» у процесі проектування. Основні стадії структури включають:

- Ескізне проектування полягає у створенні ескізного проекту, що складається з сукупності документів, що містять принципові рішення і дають загальне уявлення про пристрій і принцип роботи об'єкта, що розробляється, а також дані, що визначають його призначення, основні параметри і габаритні розміри. У разі великої складності об'єкта цьому етапу може передувати аван-проект (передпроектне дослідження), що містить теоретичні дослідження, призначені для обґрунтування принципової можливості і доцільності створення даного об'єкта. При необхідності проводять виготовлення та випробування макетів об'єкта, що розробляється.
- Технічне проектування полягає у створенні технічного проекту, утвореного сукупністю документів, які мають містити остаточні технічні рішення, що дають повне уявлення про пристрій об'єкта, що проектується, вихідні дані для розробки робочої документації.
- На стадії робочого проектування (робочого проекту) спочатку розробляють докладну конструкторську документацію виготовлення дослідного зразка і його випробування. Випробування проводять у низку етапів (від заводських до приймально-здавальних), за результатами яких коригують проектні документи. Далі розробляють робочу документацію виготовлення установчої серії, її випробування, оснащення виробничого процесу основних складових частин вироби. За результатами цього етапу знову коригують проектні документи та розробляють робочу документацію для виготовлення та

випробування головної (контрольної) серії. На основі документів остаточно відпрацьованих і перевірених у виробництві виробів, виготовлених за зафіксованим і повністю оснащеним технологічним процесом, розробляють завершальну робочу документацію виробництва, що встановився.

- Завершує цикл робіт етап, що підбиває підсумок проектної діяльності, сертифікація. Її призначення — визначення рівня якості виробленого виробу та підтвердження його відповідності вимогам тих країн, де передбачається його подальша реалізація. Необхідність виділення цього етапу як самостійного викликана тим, що у час експорт продукції чи її реалізація всередині країни у часто неприпустимі без наявності в неї сертифіката якості. Сертифікація може бути обов'язковою чи добровільною. Обов'язковій сертифікації підлягають товари, на які законами чи стандартами встановлено вимоги, що забезпечують безпеку життя та здоров'я споживачів, охорону навколишнього середовища, запобігання заподіянню шкоди майну споживача. Добровільна сертифікація проводиться із ініціативи підприємств. Зазвичай це робиться з метою офіційного підтвердження показників продукції, що виготовляється підприємством, і, як наслідок, підвищення довіри до неї у споживачів.

4.3 Проектування схеми і опис бази даних

Виходячи з вимог технічного завдання, як СУБД був обраний продукт – MySQL. Для роботи з базою даних буде використовуватися phpMyAdmin.

Навіщо потрібна база даних

Будь-який сайт складається з однієї та більше HTML-сторінок. У всіх є щось спільне, наприклад, каркас, але на кожній сторінці є своя інформація — контент. Найчастіше на сайті використовуються графічні скрипти, які на

основі тих чи інших дій відвідувачів сайту генерують вимоги. Такі скрипти називають CMS або системою управління сайтом, адмінкою. Система керування сайтом забезпечує такий інтерфейс редагування інформації, який не потребує зручних знань мови розмітки сторінок HTML.

Дані в такій системі можуть зберігатися у файлах і базі даних, але найчастіше використовуються бази даних, оскільки СУБД (система управління базою даних) має зручні і швидкі засоби пошуку потрібних записів, додавання та видалення відомостей, сортування.

MySQL - вільна реляційна система управління базами даних (СУБД). Під словом "вільна" мається на увазі її безкоштовність, під "реляційна" - робота з базами даних, заснованих на двовимірних таблицях. Система випущена 1995 року, її розробка активно продовжується.

MySQL має ряд переваг:

1. Висока швидкість роботи.
2. Підтримка майже всіх CMS.
3. Безкоштовна ліцензія.
4. Надійна та проста система безпеки.
5. Підтримка кількох типів таблиць: MyISAM, InnoDB.
6. Плагіни, що дозволяють спростити та налаштувати роботу під себе.
7. В одній таблиці може бути кілька мільйонів записів.

MySQL універсальна та застосовується при розробці веб-сайтів, веб-додатків та корпоративних баз даних початкового рівня.

Що таке phpMyAdmin

phpMyAdmin - це веб-додаток, призначений для віддаленого адміністрування баз даних на MySQL-сервері. phpMyAdmin дозволяє проводити різні операції: створення та видалення таблиць, додавання, редагування та

видалення записів, імпорт даних на сервер та створення дампа бази даних (експорт) та ін.

Інші варіанти баз даних

PostgreSQL

PostgreSQL є одним з кількох безкоштовних популярних варіантів СУБД, які часто використовуються для ведення баз даних веб-сайтів. Це дуже стара система, тому в даний час вона добре розвинена, і дозволяє користувачам управляти як структурованими, так і неструктурованими даними. Може використовуватися на більшості основних платформ, включаючи Linux (де особливо добре проявляється продуктивність). Чудово справляється із завданнями імпорту інформації з інших типів баз даних за допомогою власного інструментарію.

Переваги

- Є масштабованим рішенням і дозволяє обробляти терабайти даних.
- Підтримує формат JSON.
- Існує безліч зумовлених функцій.
- Доступний ряд інтерфейсів.

Недоліки

- Документація туманна, тож, можливо, відповіді на деякі питання доведеться шукати в інтернеті.
- Конфігурація може збентежити непідготовленого користувача.
- Швидкість роботи може падати під час пакетних операцій або виконання запитів читання.
- Ідеально підходить для організацій з обмеженим бюджетом, але вимагає залучення кваліфікованих фахівців, коли потрібна можливість вибрати унікальний інтерфейс та використовувати json.

MongoDB

Ще одна безкоштовна система, яка має комерційну версію – MongoDB. Вважається одним із класичних прикладів NoSQL-систем, використовує JSON-подібні документи та схему бази даних. Написана мовою C++. Вона призначена для програм, які використовують як структуровані, так і неструктуровані дані. Ядро є дуже гнучким і працює при підключенні бази даних до програм через драйвери MongoDB. Існує широкий вибір доступних драйверів, тому легко знайти драйвер, який працюватиме з необхідною мовою програмування.

Оскільки система MongoDB не була розроблена для обробки моделей реляційних даних (хоча може це виконувати), можуть виникнути проблеми продуктивності, якщо спробувати використовувати її таким чином. Однак, двигун призначений для обробки різних даних, які не можна віднести до реляційних, і може добре справлятися там, де інші двигуни працюють повільно або безсилі.

MongoDB 5.0 - це остання версія (на липень 2021 р.), і вона має нову систему двигунів зберігання, що підключається. Документи можуть бути перевірені в процесі оновлення або виконання вставок, а функції текстового пошуку покращено. Нова здатність часткового індексування може призвести до більш високої продуктивності зменшуючи розмір індексів.

Переваги

- Швидкість та простота у використанні
- Двигун підтримує json та інші традиційні документи NoSQL.
- Дані будь-якої структури можуть бути збережені/прочитані швидко та легко.

Недоліки

- SQL не використовується як мови запитів.

- Інструменти для перекладу SQL-запитів до MongoDB доступні, але їх слід розглядати як доповнення.
- Програма встановлення може тривати багато часу.
- Підходить для організацій, які працюють із різнорідними даними, які важко піддаються класифікації. Для впровадження будуть потрібні висококласні фахівці.

MariaDB

Ця СУБД є безкоштовною, але, як і багато інших безкоштовних програм, пропонує платні версії. Є безліч доступних плагінів розширень, мабуть, це СУБД, що найбільш швидко розвивається на даний момент.

MariaDB фактично - це відгалуження від СУБД MySQL, яке розробляється спільнотою під ліцензією GNU GPL. Розробку та підтримку MariaDB здійснює компанія MariaDB Corporation Ab та фонд MariaDB Foundation. Поштовхом до створення стала необхідність забезпечення вільного статусу СУБД, на противагу політиці ліцензування MySQL компанією Oracle. Система ліцензування MariaDB зобов'язує учасників, які бажають додати свій код до основної гілки СУБД, обмінюватися своїми авторськими правами з MariaDB Foundation для охорони ліцензії та можливості створювати критичні виправлення для MySQL.

Провідний розробник – Майкл Віденіус, автор оригінальної версії MySQL та засновник компанії Monty Program AB.

Ядро бази даних дозволяє вибирати із кількох систем зберігання, і це робить використання ресурсів більш оптимізованим, що підвищує продуктивність запитів та обробки. До складу MariaDB включено підсистеми зберігання даних XtraDB для можливості заміни InnoDB як основної підсистеми зберігання. Також включені підсистеми Aria, PBXT та FederateX. Вона повністю сумісна з MySQL, і цілком підходить як заміна, т.к. повністю клонований як набір команд, і API. Багато розробників MySQL були залучені до процесу розробки, а зараз беруть участь у розвитку.

Переваги

- Продуктивність
- Індикатори дадуть вам знати, як обробляється запит.
- Розширювана архітектура та плагіни дозволяють налаштовувати інструмент відповідно до ваших потреб.
- Шифрування доступне в мережі, сервері та рівні програми.

Недоліки

- На сьогодні стабільність нижча, ніж у MySQL, тому навіть на нових проектах можна рекомендувати встановлювати mysql.
- Двигун досить новий, тому поки що немає жодних гарантій подальших оновлень.
- Як і в багатьох інших безкоштовних баз даних, вам доведеться платити за підтримку.

Ідеальна як альтернатива MySQL, якщо MySQL не влаштовує з якихось причин.

Oracle 12c

Не дивно, що Oracle пропонує однойменний продукт, з якого зазвичай починається розгляд варіантів популярних СУБД. Першу версію Oracle було створено наприкінці 70-х років. На даний момент цей продукт має блискучу репутацію. Крім того, існує кілька версій цього продукту для задоволення потреб конкретної організації.

Актуальна версія Oracle на момент написання цієї статті – 12c – призначена для хмарних середовищ і може бути розміщена на одному або кількох серверах, це дозволяє керувати базами даних, які містять мільярди записів. Деякі з функцій новітньої версії Oracle включають grid framework і використання як фізичних, так і логічних структур.

Це означає, що фізичне управління даними не впливає на доступ до логічних структур. Крім того, безпека в цій версії доведена до найвищого рівня, тому що кожну транзакцію ізольовано від інших.

Переваги

- Найсвіжіші інновації та вражаючий функціонал уже впроваджено у цьому продукті, оскільки компанія Oracle прагне тримати планку навіть на тлі інших розробників СУБД.
- Оракул є вкрай надійним, фактично це еталон надійності серед подібних систем.

Недоліки

- Вартість Oracle може бути непомірно високою, особливо для невеликих організацій.
- Система може вимагати значних ресурсів вже відразу після встановлення, тому можливо, потрібно буде модернізувати обладнання для впровадження Oracle.

Ідеально підходить для великих організацій, які працюють з величезними базами даних та різноманітними функціями.

4.4 Проектування функціональної частини

Функціональна частина сайту буде складатися з профілю користувача, переліку ігрових персонажів, чату між користувачами та функціонал порівняння персонажів за силою.

Сайт буде нагадувати мессенджери, але тільки для гравців певної гри. Для сайту буде написана API, окрім багатьох інших ендпоінтів, у API буде реалізована аутентифікація через Bearer token

Bearer-токен - це тип авторизаційного токена, який надається користувачеві після успішної автентифікації для доступу до захищених ресурсів. Цей токен зазвичай є рядком, який користувач повинен передавати в заголовку HTTP-запиту для кожного запиту до захищеного сервісу або додатку.

Що таке автентифікація на основі токенів?

Аутентифікація на основі токенів полегшує процес аутентифікації для вже відомих користувачів. Для початку роботи користувач надсилає запит до сервера, вказавши ім'я користувача та пароль. Потім сервер підтверджує їх на підставі значень, зареєстрованих у базі даних ідентифікаційної

інформації. Якщо ідентифікаційні дані підтверджені, сервер повертає токен аутентифікації (який також зберігається у базі даних).

Коли той же користувач надалі надсилає запити на доступ до захищених ресурсів, ці запити можуть бути авторизовані за допомогою токена автентифікації замість імені користувача та пароля. Сервер звіряє токен із зареєстрованим у базі даних токеном та надає доступ. Аутентифікацію можна реалізувати на основі різних типів токенів, наприклад, OAuth та JSON Web Tokens (JWT).

JWT використовує безпечний спосіб, заснований на підписаних токена, що дозволяє з легкістю виявляти модифікації. Апаратні токени можуть містити ідентифікаційні дані або генерувати одноразовий пароль.

Система входу до профиля користувача

Розвитком політики виборчого доступу є управління доступом з урахуванням ролей (RBAC), де доступом до об'єктів системи формується з урахуванням специфіки їх застосування з урахуванням ролі суб'єктів у кожен час. Ролі дозволяють визначити зрозумілі для користувачів правила розмежування доступу. Роль поєднує властивості виборчого управління доступом, ставлячи у відповідність суб'єктам об'єкти, і мандатного, при зміні ролей зміниться і доступ до групи файлів, але цей тип доступу більш гнучкий, порівняно з попередніми, і може їх моделювати. Зараз RBAC широко використовується для управління привілеями користувача в межах єдиної системи або програми. Список таких систем включає Microsoft Active Directory, SELinux, FreeBSD, Solaris, СУБД Oracle, PostgreSQL 8.1, SAP R/3, Lotus Notes і безліч інших.

5 РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАСОБУ

5.1 Обґрунтування вибору інструментальних засобів

Для реалізації було вибрано безліч інструментальних засобів, основні вимоги до яких полягають в продуктивності, масштабованості і

досить високою абстрактності компонентів для прискореної реалізації функціональних частин.

Практично вся API частина була написана на мові PHP. В цьому проекті була використана 8.1 версія. На сьогоднішній день 7 версія мови програмування PHP є застарілою, і використовується в рідких випадках.

PHP 8 став ще швидшим та надійнішим. У порівнянні з версією PHP 7 з'явилося багато нових і корисних речей, які обов'язково будуть затребувані користувачами. Розглянемо докладніше ці нововведення.

Типи Union 2.0 (Об'єднані типи)

Існує безліч випадків коли використання типів об'єднання може бути корисним. Замість анотацій PHPDoc для об'єднаних типів тепер можна використовувати оголошення union type, які перевіряються відразу під час виконання.

Union Types – це сукупність двох або більше типів, які вказують, що ви можете використовувати будь-який із них.

JIT – компілятор

JIT (Just In Time - "в потрібний момент") - це динамічна компіляція байт коду в машинний. JIT компілятор значно покращує продуктивність роботи програм. Перевірка на синтетичних бенчмарках показує поліпшення продуктивності приблизно в 3 рази і в 1,5-2 рази для деяких програм, що довго працюють.

У JIT код перекладається спочатку в проміжне уявлення і потім у машинний код. Тобто виконується безпосередньо на процесорі, а не на віртуальній машині Zend VM.

JIT реалізований як незалежна частина OPcache. Його можна включати/вимикати під час компіляції або виконання скрипту.

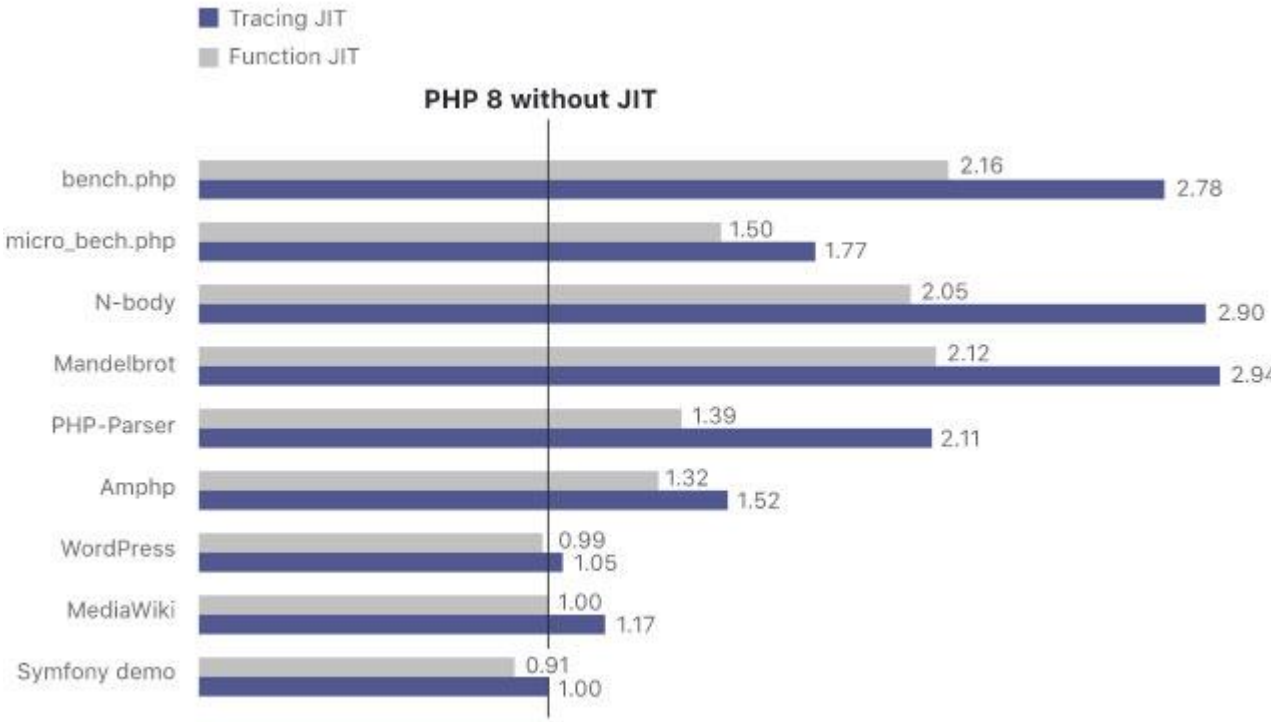

Рис. 5.1 – PHP 8 без JIT

Використання ::class для об'єктів

У PHP 8 тепер можна використовувати ::class для об'єктів замість того, щоб використовувати get_class() як раніше. Працює так само, як і get_class().

Атрибути

У PHP 8 тепер замість анотацій PHPDoc можна використовувати структурні метадані з нативним синтаксисом PHP.

Вираз Match

У PHP 8 новий вираз match схожий на оператор switch, але має свої особливості:

- Match - це вираз і його результат може бути збережений у змінній або повернутий.
- Умови match підтримують тільки однорядкові вирази, для яких не потрібна конструкція, що управляє break;
- Вираз match використовує суворе порівняння.

Оператор Nullsafe

У PHP 8 тепер замість перевірки на null можна використовувати послідовність викликів з новим оператором Nullsafe. Якщо один із елементів у послідовності повертає null, то виконання переривається і вся послідовність повертає null.

Поліпшене порівняння рядків та чисел

Тепер у порівнянні з числовим рядком PHP 8 використовує порівняння чисел. В іншому випадку число перетворюється на рядок і використовується порівняння рядків.

Помилки узгодженості типів для вбудованих функцій

У PHP 8 більшість внутрішніх функцій тепер викидають виняток Error, якщо під час перевірки параметра виникає помилка.

Іменовані аргументи

У PHP 8 тепер можна:

- Вказувати лише необхідні параметри та пропускати необов'язкові.
- Порядок аргументів не важливий, аргументи самодокументовані.

Оголошення властивостей у конструкторі

У новій версії PHP 8 тепер використовується набагато менше шаблонного коду для визначення та ініціалізації властивостей.

Новий тип повернення static

Раніше в PHP вже були типи, що повертаються — self і parent, але static не був допустимим типом повернення для цієї мови програмування. Тепер є.

Throw-вирази

У PHP оператор throw не може створювати винятки там, де були дозволені лише вирази, наприклад, стрілочні функції, оператор об'єднання та тернарний оператор.

У новому PHP 8 тепер ви можете перетворювати інструкцію throw на вираз.

Weak maps

Weak maps - це набір об'єктів, на які в коді слабко посилаються ключі, що може призвести до їх видалення збирачами сміття. PHP 8 додає клас WeakMaps для створення збереження посилання на об'єкт. У цьому вона перешкоджає видалення самого об'єкта.

Завершальна кома у списках параметрів

Раніше під час виклику функції у списках параметрів відсутня підтримка коми. У PHP 8 тепер це дозволено.

Новий Stringable інтерфейс

У PHP 8 тепер можна використовувати інтерфейс Stringable для анотації типів або імплементації методу __toString (). І більше не потрібно реалізовувати його вручну.

Нові функції в PHP 8

get_debug_type() - повертає тип змінної. На відміну від gettype() повертає корисніший висновок для масивів, рядків, анонімних класів та об'єктів.

str_starts_with() — перевіряє, чи починається рядок із певного рядка. Повертає TRUE/FALSE.

str_ends_with() — перевіряє, чи закінчується рядок із певного рядка. Повертає TRUE/FALSE.

str_contains() — перевіряє, чи містить рядок інший рядок.

fdiv() - ця функція схожа на fmod() і intdiv(), що дозволяє ділити на 0. Але замість помилок отримуємо INF, -INF або NAN, залежно від випадку.

get_resource_id() — раніше ресурсам надається ідентифікатор, але для цього необхідно було перетворення ресурсу в int. У PHP 8 додана функція get_resource_id(), яка робить цю операцію наочнішою і безпечнішою для типів.

5.2 Розробка проєкту

Форма реєстрації користувача на сайті

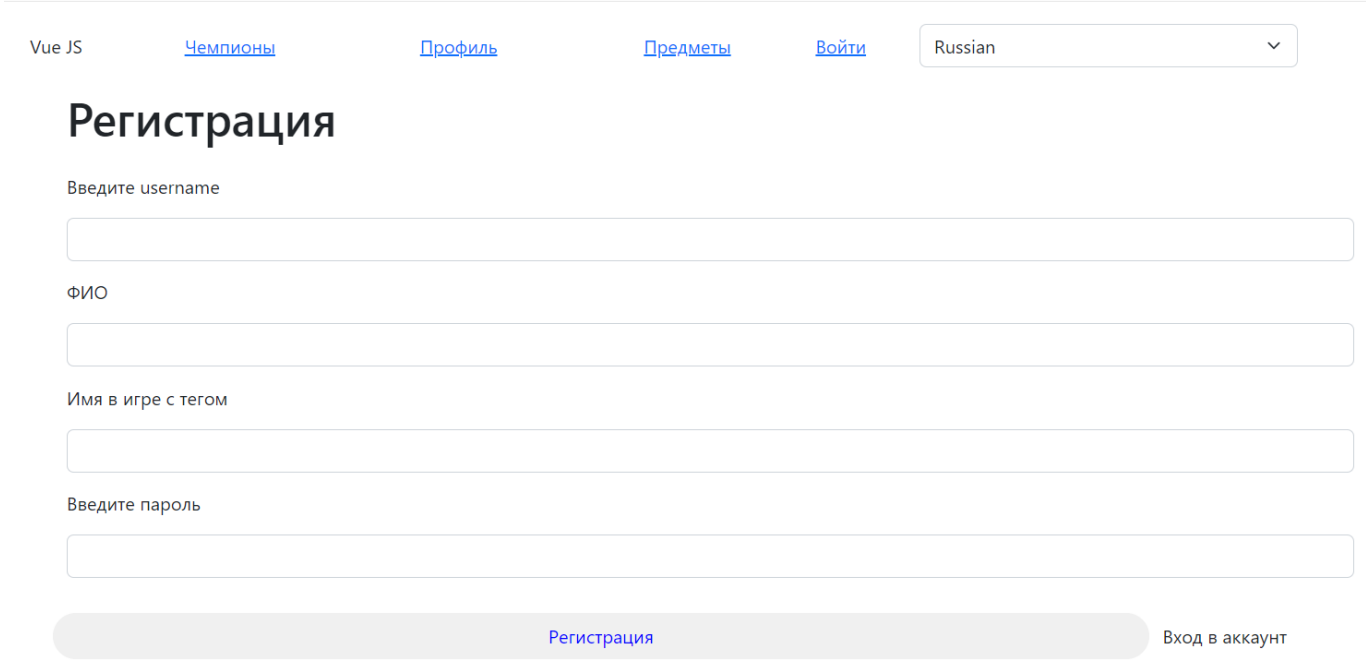

Рис. 5.2 – Форма реєстрації

Код функції реєстрації користувача

```
public function registrationUser(){
    switch($this->method){
        case "POST":
            $_POST = json_decode(file_get_contents('php://input'), true);
            $name = $_POST["name"];
            $username = $_POST["username"];
            $setlolName = $_POST["setlolName"];
            $password = hash("sha1", $_POST["password"]);
            $checkduplicate = mysqli_fetch_assoc(mysqli_query($this->connect,
"SELECT * FROM `users` WHERE `username` = '$username'"));
```

```
            if(empty($checkduplicate)){
                if(mysqli_query($this->connect,"INSERT INTO `users` (`id`,
`name`, `username`, `setlolName`, `password`) VALUES
(NULL,'$name','$username','$setlolName', '$password')")){
                    $lastUser = mysqli_fetch_assoc(mysqli_query($this-
>connect,"SELECT * FROM `users` ORDER BY id DESC LIMIT 1"));
                    $token = $this->getToken($lastUser["id"], $this-
>connect);
                    $this->requestData["data"] = $token;
                    $this->requestData["status"] = true;
                }
                else{
                    http_response_code(400);
                    $this->requestData["textError"] = "При создании
пользователя произошла ошибка, повторите попытку позднее";
                    $this->requestData["status"] = false;
                }
                echo json_encode($this->requestData);
                break;
            }
            else{
                http_response_code(400);
                $this->requestData["textError"] = "Пользователь с таким
username уже существует";
                $this->requestData["status"] = false;
                echo json_encode($this->requestData);
            }
            break;
    }
}
```

Форма логіну користувача на сайті

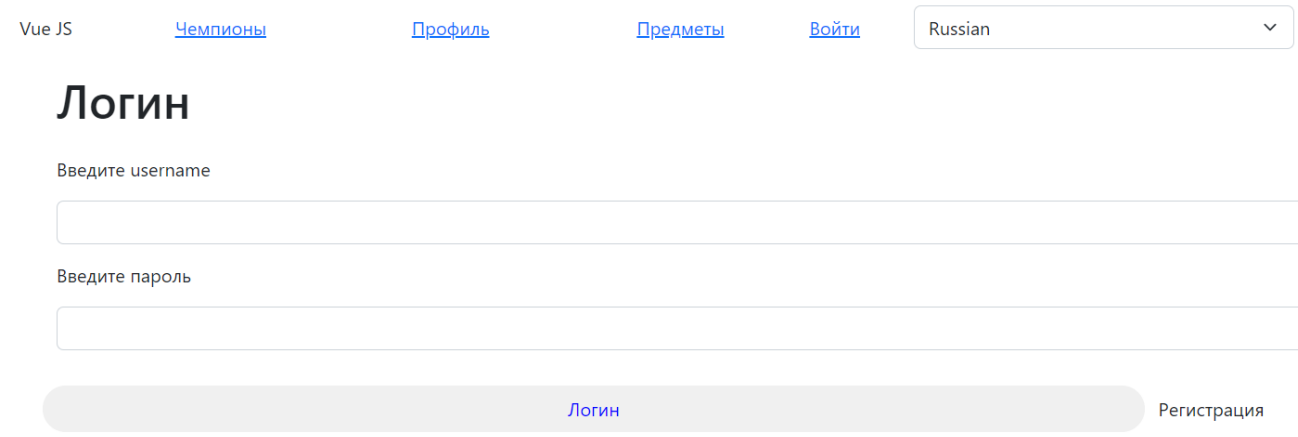

Рис. 5.3 – Форма логіну

Код логіну користувача на сайті

```
public function authToProfile()
{
    switch ($this->method) {
        case "GET":
            $username = $_GET["username"];
            $password = hash("sha1", $_GET["password"]);
            $user = mysqli_fetch_assoc(mysqli_query($this->connect, "SELECT *
FROM users WHERE users.username='$username' AND users.password='$password'
LIMIT 1"));
```

```
            if($user){
                $token = $this->getToken($user["id"]);
                $this->requestData["data"] = $token;
                $this->requestData["status"] = true;
            }
            else{
                $this->requestData["textError"] = "Данные не верны";
                $this->requestData["status"] = false;
            }
            echo json_encode($this->requestData);
            break;
    }
}
```

Сторінка переліку усіх героїв комп’ютерної гри

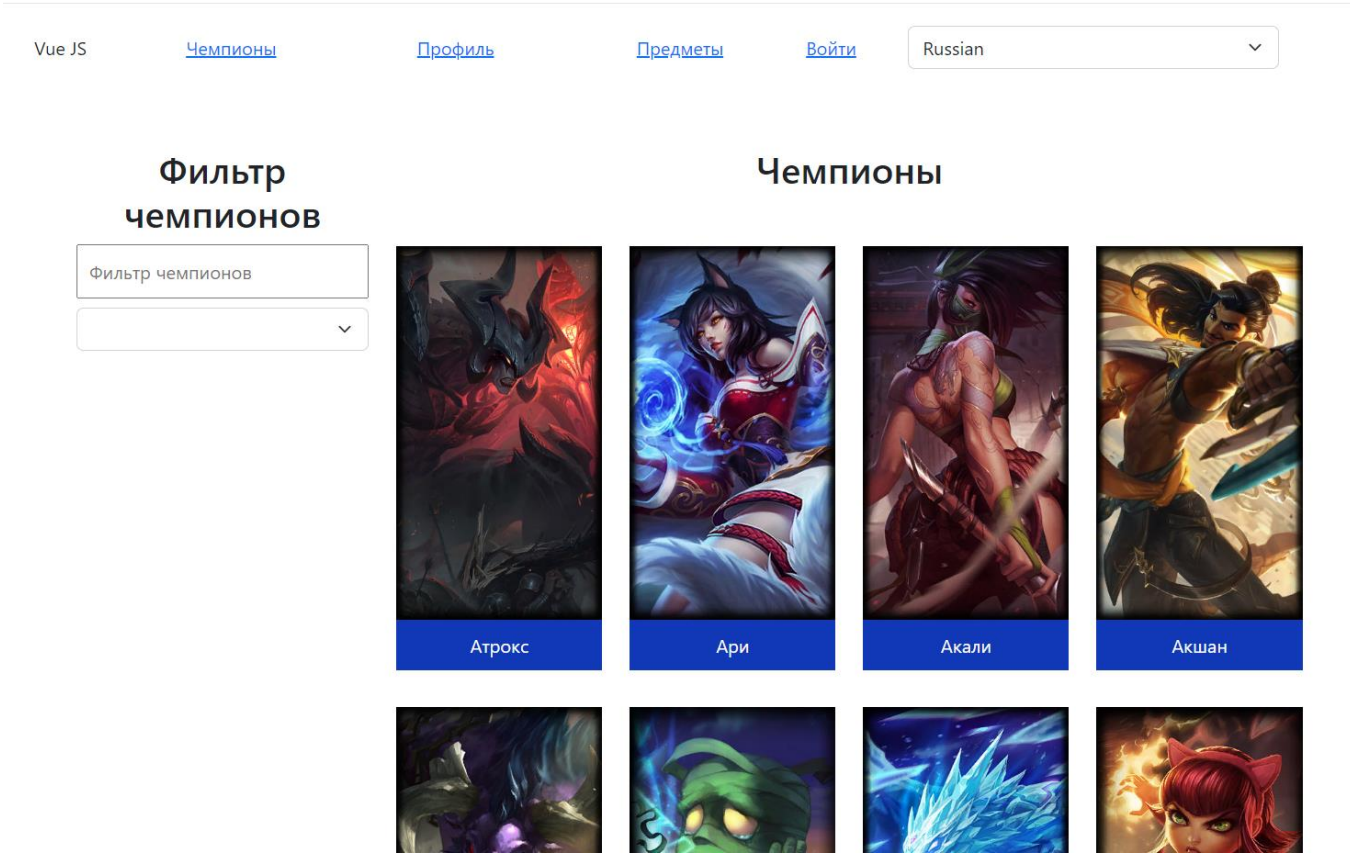

Рис. 5.4 – Сторінка перегляду героїв

На цій сторінці використовується API комп’ютерної гри League of Legends для переліку усіх героїв цієї гри, також для цієї сторінки булі зроблені 2 фільтри, 1 фільтр дозволяє відфільтрувати героїв по їх імені, 2 реалізує фільтрацію по ролі героя. Важливо що ці фільтри між собою зв’язані, тобто якщо вибрати певну роль і ввести послідовність букв, то ми побачимо героїв с цією ролю і послідовністю букв

Приклад реалізації фільтрів

```
textSelect(state, getters) {
    const asArray = Object.entries(state.champions);
    if (state.sortValue !== "") {
        let filterArray = asArray.filter(([key, value]) => {
            for (let i = 0; i < value.tags.length; i++) {
                if (value.tags[i] == state.sortValue) {
                    return [key, value]
                }
            }
        });
        return Object.fromEntries(filterArray)
    } else {
        return Object.fromEntries(asArray);
    }
},
textFilterArray(state, getters) {
    const asArray = Object.entries(getters.textSelect);
    let filterArray = asArray.filter(([key, value]) =>
[key][0].toLowerCase().includes(state.textFilter.toLowerCase()));
    return Object.fromEntries(filterArray)
}
```

Сторінка детального перегляду героя

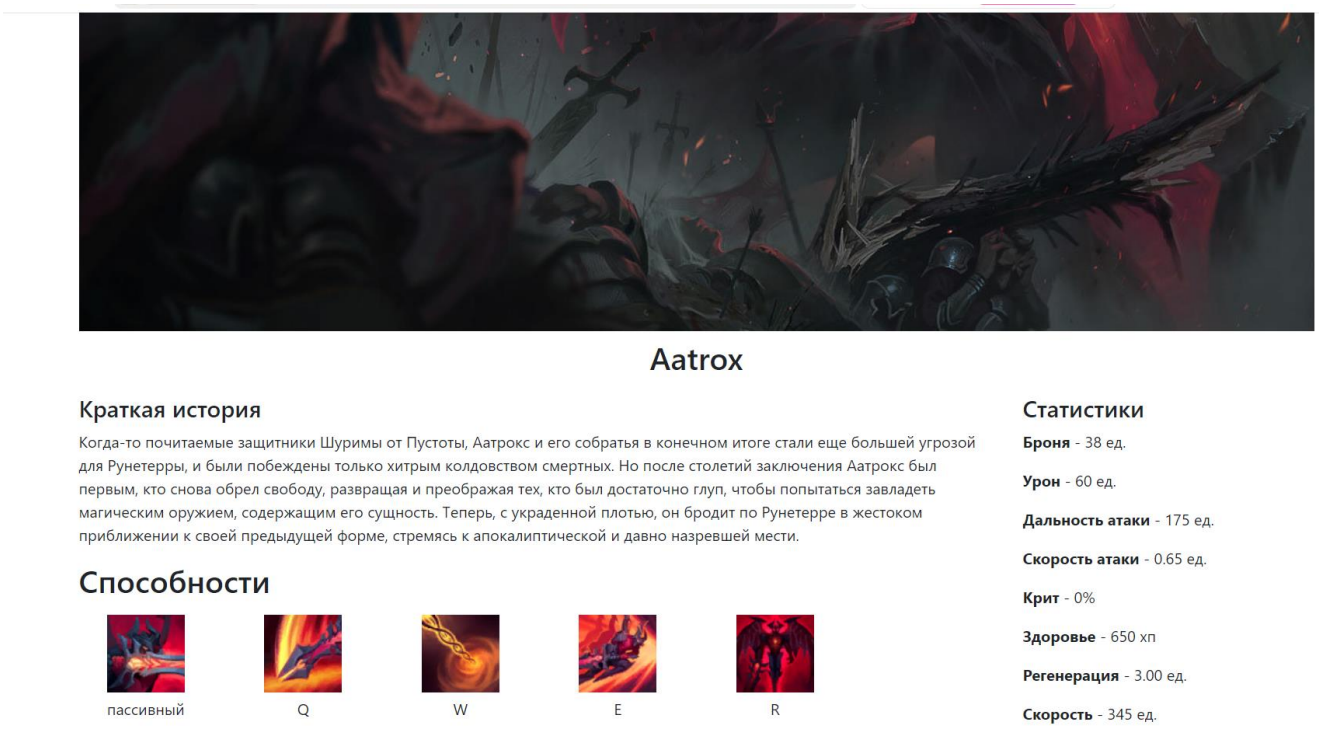

Рис. 5.5 – Сторінка детального перегляду героя

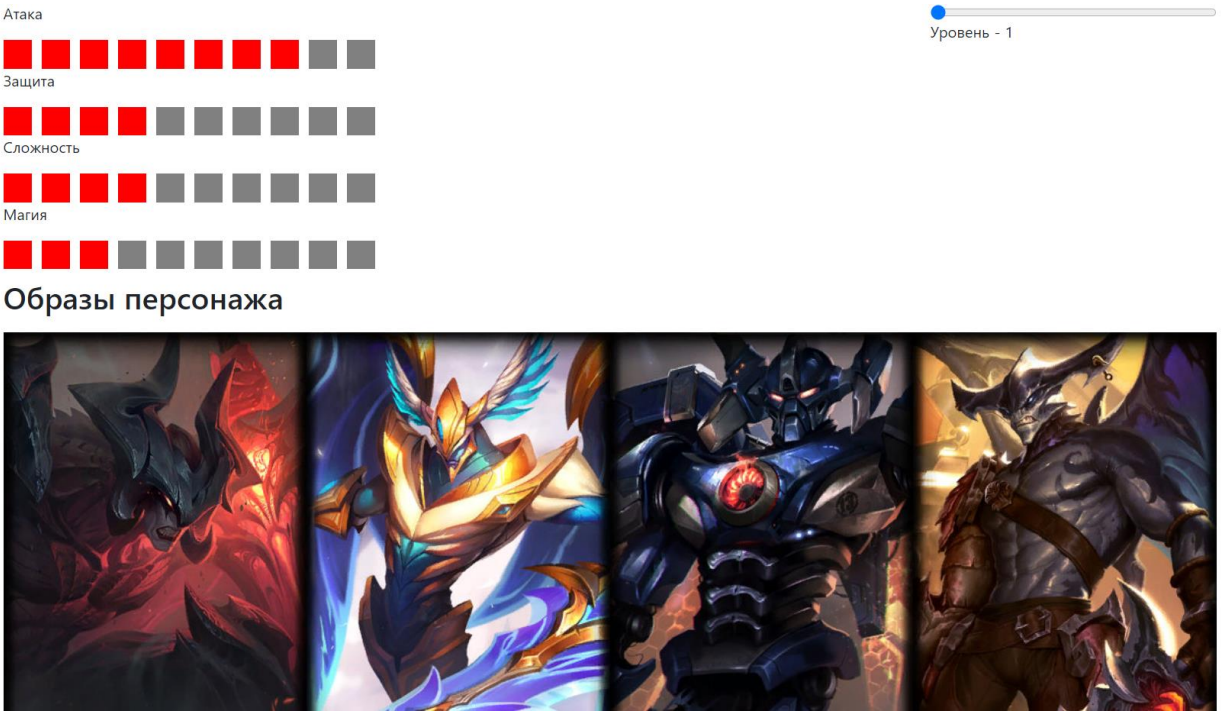

Рис. 5.6 – Сторінка детального перегляду героя

Функція підрахунку характеристик герою

```
ChampionLevelFunc({state, commit}, level) {
    state.statsDefault.forEach(([key, value]) => {
        state.champion.stats[key] = state.champion.stats[key + 'perlevel']
            ? value + ((state.champion.stats[key + 'perlevel'] *
Math.floor(level)) - state.champion.stats[key + 'perlevel'])
            : value;
    });
}
```

5.3 Тестування сайту

Види тестування сайту

5.3.1. Функціональне тестування

Цей вид тестування спрямований на перевірку того, чи всі можливості та функції сайту працюють так, як було задумано. Тестування функціональності включає кілька параметрів, таких як перевірка інтерфейсу користувача, API, тестування бази даних, тестування безпеки, тестування сервера, а також всіх базових функцій платформи. Під час роботи фахівці застосовують як ручне, і автоматизоване тестування.

Наприклад, функціональне тестування сайту передбачає перевірку всіх ссылок на страницах, анкет и форм, файлов Cookies, HTML и CSS, базы даних, негативних та позитивних сценаріїв поведінки користувачів на сайті.

5.3.2. Юзабіліті-тестування

Тестування юзабіліті: оцінює, наскільки відвідувачу зручні та інтуїтивно зрозумілі макет, дизайн та загальна навігація сайту. Юзабіліті-тести можуть проводити як тестувальники, так і невелика фокус-група, схожа на вашу цільову аудиторію.

До перевірки навігації відноситься тестування меню, кнопок або посилань, що ведуть різні сторінки сайту. Вони повинні бути легко помітними та однакової форми та розміру на всіх веб-сторінках. Текст має бути коректним та без помилок.

5.3.3. Тестування інтерфейсу користувача

Цей вид тестування передбачає, що QA-команда перевіряє всі елементи інтерфейсу користувача, застосовуючи всілякі тестові сценарії. Перевірка елементів інтерфейсу передбачає тестування:

- форми на коректність введених даних та процедури входу в систему
- текстових полів та посилань
- закликів до дії
- віконних вікон
- навігації по сайту та рядки пошуку
- баз даних

Процес тестування перелічених елементів вручну застосовується для невеликих сайтів. Проте комплексне тестування великих проектів передбачає використання інструментів автоматизованого тестування.

5.3.4. Тестування сумісності

Тести на сумісність гарантують, що ваш сайт коректно відображатиметься на різних пристроях. Це передбачає:

- Тест на сумісність із браузерами: один і той же веб-сайт у різних браузерах відображатиметься інакше. Необхідно перевірити, чи правильно відображаються сторінки в браузерах, чи працює JavaScript, AJAX та автентифікація. Також можна оцінити сумісність із мобільними браузерами.
- Відображення веб-елементів, таких як кнопки, текстові поля тощо, змінюється з вибором іншої операційної системи. Переконайтеся, що сайт коректно працює в різних поєднаннях ОС, таких як Windows, Linux, Mac і браузерах Яндекс, Opera, Firefox, Internet Explorer, Safari і т.д.

5.3.5. Тестирование производительности

Тестування продуктивності дозволить забезпечити стабільну роботу сайту за будь-яких навантажень у вигляді великої кількості користувачів або операцій.

Тестування включатиме, але не обмежуватиметься такими аспектами:

- Час відповіді (відгуку) веб-програми на різних швидкостях з'єднання
- Навантажувальне та стрес-тестування сайту для визначення його стабільності при очікуваних та максимальних навантаженнях
- Тестування того, як платформа відновлюється після збою внаслідок пікового навантаження
- Перевірка того, що методи оптимізації, такі як стиснення gzip, кеш браузера та сервера, включені для скорочення часу завантаження сторінок та файлів.

Тестування продуктивності може застосовуватися для розуміння масштабованості сайту або для оцінки продуктивності серед інших

продуктів, таких як сервери і проміжне ПЗ для потенційних покупок.

5.3.6. Тестування безпеки

Тестування безпеки особливо необхідне для банківських сайтів, сайтів авіакомпаній або майданчиків електронної комерції, які зберігають конфіденційну інформацію про клієнтів, наприклад, кредитні картки, дані паспортів тощо. Тестування безпеки передбачає перевірку:

- несанкціонованого доступу до захищених сторінок
- файлів з обмеженим доступом: щоб їх не можна було завантажувати без відповідного права
- автоматичного завершення сеансів після тривалої бездіяльності користувача
- під час використання SSL-сертифікатів сайт повинен перенаправляти на зашифровані SSL-сторінки

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Дунаев, Вадим Сценарии для Web-сайта. PHP и JavaScript / Вадим Дунаев. - М.: БХВ-Петербург, 2008. - 335 с.
2. Изучаем Node.js. - М.: Питер, 2013. - 400 с.
3. Никсон, Р. Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5 / Р. Никсон. - М.: Питер, 2015. - 931 с.
4. Vue.js в действии (pdf+epub) | Хэнчетт Эрик, Листуон
5. Дэвид Скляр. Изучаем PHP 7. Руководство по созданию интерактивных веб-сайтов

ДОДАТОК 1. API НОВОГО ПРОЕКТУ

Класс конфігу проекту.

```php
<?php
class Config
{
    public $connect;
    public $method;
    protected $requestData = ["text"=>"","status"=>false];
    function __construct()
    {
        $this->connect = mysqli_connect("localhost", "root", "", "vue_db");
        $this->method = $_SERVER["REQUEST_METHOD"];
    }
    protected function getBearerToken(){
        $headers = array_change_key_case(getallheaders(), CASE_LOWER);
        if (!isset($headers['authorization'])) {
            return null;
        }

        return trim(str_replace('Bearer', '', $headers['authorization']));
    }
}
```

Класс опціональних функцій та класс функцій користувача

```php
<?php
class OpthionalFunc extends Config
{
    protected function setToken($id)
    {
        $token = bin2hex(random_bytes(18));
        $idUser = $id;
        $today = date("Y-m-d", time()+60*60*24*7);
        mysqli_query($this->connect,"DELETE FROM `tokens` WHERE
`user_id`=$idUser");
        mysqli_query($this->connect,"INSERT INTO `tokens` (`value`,
`user_id`, `valid_until`) VALUES ('$token', '$idUser', '$today')");
        return mysqli_fetch_assoc(mysqli_query($this->connect, "SELECT
tokens.value FROM `tokens` WHERE `user_id`='$id' LIMIT 1"));
    }
    protected function getToken($id)
    {
        $token = mysqli_fetch_assoc(mysqli_query($this->connect, "SELECT *
FROM `tokens` WHERE `user_id`='$id' LIMIT 1"));
        if($token){
            if($token["valid_until"] < date("Y-m-d", time())){
                return $this->setToken($id);
            }
            else{
                return $token["value"];
            }
        }
        else{
            return $this->setToken($id);
        }
    }
    protected function getProfileByToken($token)
    {
        $user_id = mysqli_fetch_assoc(mysqli_query($this->connect, "SELECT
`user_id`, `valid_until` FROM `tokens` WHERE `value`='$token' LIMIT 1"));
        if($user_id){
            if($user_id["valid_until"] < date("Y-m-d", time())){
                $this->setToken($user_id["user_id"], );
```

```
                $this->requestData["textError"] = "Токен перегенерирован";
                $this->requestData["status"] = false;
            }
            else{
                $user_id = $user_id["user_id"];
                $user_object = mysqli_fetch_assoc(mysqli_query($this-
>connect, "SELECT * FROM `users` WHERE `id`='$user_id' LIMIT 1"));
                $path = './user-images/user-'. $user_id;
                if(file_exists($path)){
                    $user_object["img"] = "http://localhost/vueback/user-
images/user-".$user_id."/icon.jpg";
                }
                else{
                    $user_object["img"] = "http://localhost/vueback/user-
images/default.jpg";
                }
                $this->requestData["data"] = $user_object;
                $this->requestData["status"] = true;
            }
        }
        else{
            $this->requestData["textError"] = "Токен не найден";
            $this->requestData["status"] = false;
        }
        return $this->requestData;
    }
}
class UserProfile extends OpthionalFunc{
    public function usetProfileImage()
    {
        switch ($this->method){
            case "POST":
                $id = $_POST['userId'];
                $user = mysqli_fetch_assoc(mysqli_query($this->connect,
"SELECT * FROM `users` WHERE `id` = '$id'"));
                $path = './user-images/user-'.$user['id'];
                $globalPath = "http://localhost/vueback/user-images/user-
".$user['id']."/";
                if(!file_exists($path)){
                    mkdir($path);
                }
                switch ($_FILES['image']['type']){
                    case 'image/jpeg':
                        $fileName = $path. "/". "icon.jpg";
                        $globalPath .= "icon.jpg";
                        break;
                    case 'image/png':
                        $fileName = $path. "/". "icon.png";
                        $globalPath .= "icon.png";
                        break;
                    default:
                        http_response_code(400);
                        $this->requestData["textError"] = "Формат не
поддерживается";
                        $this->requestData["status"] = false;
                        echo json_encode($this->requestData);
                        break;
                }
                if(isset($fileName)){
                    if(move_uploaded_file($_FILES['image']['tmp_name'],
$fileName)){
                        var_dump($_FILES['image']['tmp_name']);
                        $this->requestData["data"] = ["image_src" =>
$globalPath];
                        $this->requestData["status"] = true;
```

```
                    echo json_encode($this->requestData);
                }
            }
            break;
        }
    }
    public function setUserData(){
        switch ($this->method){
            case "PUT":
                $token = $this->getBearerToken();
                if(isset($token)){
                    $data = json_decode(file_get_contents('php://input'),
true);
                    $profile = $this->getProfileByToken($token);
                    $profile_id = $profile["data"]["id"];
                    $username = $data["username"];
                    $lolName = $data["lolName"];
                    if(mysqli_query($this->connect, "UPDATE `users` SET
`username` = '$username', `setlolName` = '$lolName' WHERE `id` =
'$profile_id'")){
                        $profile = $this->getProfileByToken($token);
                        echo json_encode($profile["data"]);
                    }
                }
                else{
                    $this->requestData["textError"] = "Токен не передан";
                    $this->requestData["status"] = false;
                    echo json_encode($this->requestData);
                }
                break;
        }
    }
    public function registrationUser(){
        switch($this->method){
            case "POST":
                $_POST = json_decode(file_get_contents('php://input'), true);
                $name = $ POST["name"];
                $username = $_POST["username"];
                $setlolName = $_POST["setlolName"];
                $password = hash("sha1", $_POST["password"]);
                $checkduplicate = mysqli_fetch_assoc(mysqli_query($this-
>connect, "SELECT * FROM `users` WHERE `username` = '$username'"));
                if(empty($checkduplicate)){
                    if(mysqli_query($this->connect,"INSERT INTO `users`
(`id`, `name`, `username`, `setlolName`, `password`) VALUES
(NULL,'$name','$username','$setlolName', '$password')")){
                        $lastUser = mysqli_fetch_assoc(mysqli_query($this-
>connect,"SELECT * FROM `users` ORDER BY id DESC LIMIT 1"));
                        $token = $this->getToken($lastUser["id"], $this-
>connect);
                        $this->requestData["data"] = $token;
                        $this->requestData["status"] = true;
                    }
                    else{
                        http_response_code(400);
                        $this->requestData["textError"] = "При создании
пользователя произошла ошибка, повторите попытку позднее";
                        $this->requestData["status"] = false;
                    }
                    echo json_encode($this->requestData);
                    break;
                }
                else{
                    http_response_code(400);
                    $this->requestData["textError"] = "Пользователь с таким
```

```
username уже существует";
                    $this->requestData["status"] = false;
                    echo json_encode($this->requestData);
                }
                break;
        }
    }
    public function authToProfile()
    {
        switch ($this->method) {
            case "GET":
                $username = $_GET["username"];
                $password = hash("sha1", $_GET["password"]);
                $user = mysqli_fetch_assoc(mysqli_query($this->connect,
"SELECT * FROM users WHERE users.username='$username' AND
users.password='$password' LIMIT 1"));
                if($user){
                    $token = $this->getToken($user["id"]);
                    $this->requestData["data"] = $token;
                    $this->requestData["status"] = true;
                }
                else{
                    $this->requestData["textError"] = "Данные не верны";
                    $this->requestData["status"] = false;
                }
                echo json_encode($this->requestData);
                break;
        }
    }
    public function getProfileInfo()
    {
        switch ($this->method){
            case "GET":
                $token = $this->getBearerToken();
                if(isset($token)){
                    echo json_encode($this->getProfileByToken($token));
                }
                else{
                    $this->requestData["textError"] = "Токен не передан";
                    $this->requestData["status"] = false;
                    echo json_encode($this->requestData);
                }
        }
    }
    public function getAlertsList(){
        switch ($this->method) {
            case "GET":
                $token = $this->getBearerToken();
                if(isset($token)) {
                    $profileArray = $this->getProfileByToken($token);
                    if($profileArray["status"]){
                        $user_id = $profileArray["data"]["id"];
                        $profileArray = $this->getAlerts($user_id);
                        echo json_encode($profileArray);
                    }
                }
                else{
                    $this->requestData["textError"] = "Токен не передан";
                    $this->requestData["status"] = false;
                    echo json_encode($this->requestData);
                }
        }
    }
    public function deleteAnswer()
    {
```

```
        switch ($this->method) {
            case "DELETE":
                $alertID = $_GET["idAnswer"];
                $token = $this->getBearerToken();
                if(isset($token)){
                    $profileArray = $this->getProfileByToken($token);
                    if($profileArray["status"]) {
                        $user_id = $profileArray["data"]["id"];
                        if(mysqli_query($this->connect, "DELETE FROM
`message_users` WHERE `message_for_user`=$user_id AND `id`=$alertID LIMIT
1")){
                            $this->requestData = $this->getAlerts($user_id);
                            $this->requestData["status"] = true;
                            echo json_encode($this->requestData);
                        }
                        else{
                            $this->requestData["status"] = false;
                            $this->requestData["textError"] = "Произошла
ошибка при удалении повторите попытку позднее";
                            echo json_encode($this->requestData);
                        }
                    }
                }
                else{
                    $this->requestData["textError"] = "Токен не передан";
                    $this->requestData["status"] = false;
                    echo json_encode($this->requestData);
                }
        }
    }
    public function getAlerts($user_id)
    {
        $array = mysqli_query($this->connect, "SELECT * FROM `message_users`
WHERE `message_for_user`=$user_id ORDER BY `alert_time_create` DESC");
        $profileArray["data"] = [];
        if(isset($array)){
            while($row = mysqli_fetch_assoc($array)){
                $profileArray["data"][] = $row;
            }
            return $profileArray;
        }
    }
}
```

Класс взаємодії користувача з друзями

```
<?php
class FriendMethods extends OpthionalFunc
{
    public function getFriedsList(){
        $array = mysqli_query($this->connect, "SELECT * FROM `users`");
        if(isset($array)){
            while($row = mysqli_fetch_assoc($array)){
                $path = './user-images/user-'. $row["id"];
                if(file_exists($path))
                    $row["img"] = "http://localhost/vueback/user-images/user-
".$row["id"]."/icon.jpg";
                else
                    $row["img"] = "http://localhost/vueback/user-
images/default.jpg";
                $requestData["data"][] = $row;
            }
            $requestData["status"] = true;
            echo json_encode($requestData);
        }
```

```
    }
    public function getUserFriends(){
        switch ($this->method){
            case "GET":
                $profileArray = $this->getProfileByToken($this-
>getBearerToken());
                if($profileArray["status"]){
                    $user_id = $profileArray["data"]["id"];
                    $friendsArray = mysqli_query($this->connect,"SELECT
friends.requested_user_id, users.id, users.name, users.username,
users.setlolName FROM `users` INNER JOIN `friends` WHERE users.id =
'$user_id' AND friends.accepted_user_id = users.id AND
friends.accepted_status = 1 UNION SELECT friends.accepted_user_id, users.id,
users.name, users.username, users.setlolName FROM `users` INNER JOIN
`friends` WHERE users.id = '$user_id' AND friends.requested_user_id =
users.id AND friends.accepted_status = 1");
                    if (mysqli_num_rows($friendsArray) > 0) {
                        $profileArray["data"] = array();
                        while ($item = mysqli_fetch_assoc($friendsArray)) {
                            $friend_id = $item["requested_user_id"];
                            $array = mysqli_fetch_assoc(mysqli_query($this-
>connect, "SELECT * FROM `users` WHERE `id`=$friend_id"));
                            $path = './user-images/user-'. $friend_id;
                            if(file_exists($path)){
                                $array["img"] =
"http://localhost/vueback/user-".$friend_id."/icon.jpg";
                            }
                            else{
                                $array["img"] =
"http://localhost/vueback/user-images/default.jpg";
                            }
                            $profileArray["data"][] = $array;

                        }
                    }
                    echo json_encode($profileArray);
                }
        }
    }
}
```

ДОДАТОК 2. СТВОРЕННЯ НОВОГО ПРОЕКТУ І ЙОГО ЗАПИС У БД

```
-- phpMyAdmin SQL Dump
-- version 5.2.0
-- https://www.phpmyadmin.net/
--
-- Хост: 127.0.0.1:3306
-- Время создания: Июн 12 2024 г., 13:11
-- Версия сервера: 10.5.17-MariaDB
-- Версия PHP: 8.0.22

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
```

```sql
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- База данных: `vue_db`
--

-- --------------------------------------------------------

--
-- Структура таблицы `friends`
--

CREATE TABLE `friends` (
  `id` int(11) NOT NULL,
  `requested_user_id` int(10) NOT NULL,
  `accepted_user_id` int(10) NOT NULL,
  `accepted_status` tinyint(1) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

--
-- Дамп данных таблицы `friends`
--

INSERT INTO `friends` (`id`, `requested_user_id`, `accepted_user_id`,
`accepted_status`) VALUES
(1, 15, 14, 1),
(2, 13, 15, 1),
(3, 13, 1, 0);

-- --------------------------------------------------------

--
-- Структура таблицы `message_users`
--

CREATE TABLE `message_users` (
  `id` int(11) NOT NULL,
  `message_type` varchar(40) COLLATE utf8mb4_unicode_ci NOT NULL,
  `message_for_user` int(10) NOT NULL,
  `text_message` varchar(260) COLLATE utf8mb4_unicode_ci NOT NULL,
  `title_message` varchar(40) COLLATE utf8mb4_unicode_ci NOT NULL,
  `alert_time_create` timestamp NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

--
-- Дамп данных таблицы `message_users`
```

```
--

INSERT INTO `message_users` (`id`, `message_type`, `message_for_user`,
`text_message`, `title_message`, `alert_time_create`) VALUES
(4, 'register_alert', 14, 'Вы успешно зарегистрировались. Чтобы изменить данные
профиля нажмите <a href=\"\">здесь</a>', 'Поздравляем!', '2024-05-19 14:34:10');

-- --------------------------------------------------------

--
-- Структура таблицы `tokens`
--

CREATE TABLE `tokens` (
  `value` varchar(40) COLLATE utf8mb4_unicode_ci NOT NULL,
  `user_id` int(10) NOT NULL,
  `valid_until` timestamp NULL DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

--
-- Дамп данных таблицы `tokens`
--

INSERT INTO `tokens` (`value`, `user_id`, `valid_until`) VALUES
('d8b093a9c4b9e85b4811cb1dcfab65cf4474', 12, '2024-05-16 21:00:00'),
('b9ae097590e9469d4c10fe46758f85d6f725', 15, '2024-05-24 21:00:00'),
('c58de9eb44ea85d2cca886bc7e4cb589afdd', 17, '2024-06-01 21:00:00'),
('c78d4a7783d953c91dba20607278094e4782', 14, '2024-06-10 21:00:00');

-- --------------------------------------------------------

--
-- Структура таблицы `users`
--

CREATE TABLE `users` (
  `id` int(11) NOT NULL,
  `name` varchar(40) COLLATE utf8mb4_unicode_ci NOT NULL,
  `username` varchar(40) COLLATE utf8mb4_unicode_ci NOT NULL,
  `about_me` varchar(50) COLLATE utf8mb4_unicode_ci NOT NULL,
  `favorit_champ` varchar(40) COLLATE utf8mb4_unicode_ci NOT NULL,
  `setlolName` varchar(40) COLLATE utf8mb4_unicode_ci NOT NULL,
  `password` varchar(120) COLLATE utf8mb4_unicode_ci NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

--
-- Дамп данных таблицы `users`
--

INSERT INTO `users` (`id`, `name`, `username`, `about_me`, `favorit_champ`,
`setlolName`, `password`) VALUES
(1, 'Artur', 'Pavlov', '', '', 'Romanovich', ''),
```

```
(7, 'testName', 'test', '', '', 'test', 'sarmatiK1'),
(8, 'testName', 'test', '', '', 'test', '123456'),
(9, 'Павлов Артур', 'salatik951', '', '', 'lol951', '123456'),
(11, 'Pavlov', 'salatik95', '', '', 'salaik#951', 'sarmatiK1'),
(12, 'Pavlov', 'salatik9', '', '', 'salaik#951',
'8c933d5e5668d401c93edd2431d15c1a1d121a29'),
(13, 'Pavlov', 'salatik91', '', '', 'salaik#951',
'8c933d5e5668d401c93edd2431d15c1a1d121a29'),
(14, 'Pavlov', 'salatik915', 'Text', '', 'lolName',
'8c933d5e5668d401c93edd2431d15c1a1d121a29'),
(15, 'Pavlov Artur', 'salatik9511', '', '', 'salatik951',
'8c933d5e5668d401c93edd2431d15c1a1d121a29'),
(16, 'Pavlov', 'salatik95111', '', '', 'salaik#951',
'8c933d5e5668d401c93edd2431d15c1a1d121a29'),
(17, 'Pavlov', 'salatik945', '', '', 'salatik967',
'8c933d5e5668d401c93edd2431d15c1a1d121a29');

--
-- Индексы сохранённых таблиц
--

--
-- Индексы таблицы `friends`
--
ALTER TABLE `friends`
  ADD PRIMARY KEY (`id`),
  ADD KEY `accept_user_fk` (`accepted_user_id`),
  ADD KEY `requested_user_fk` (`requested_user_id`);

--
-- Индексы таблицы `message_users`
--
ALTER TABLE `message_users`
  ADD PRIMARY KEY (`id`),
  ADD KEY `users_fk` (`message_for_user`);

--
-- Индексы таблицы `tokens`
--
ALTER TABLE `tokens`
  ADD KEY `user_id_fk` (`user_id`);

--
-- Индексы таблицы `users`
--
ALTER TABLE `users`
  ADD PRIMARY KEY (`id`);

--
-- AUTO_INCREMENT для сохранённых таблиц
--
```

```
--
-- AUTO_INCREMENT для таблицы `friends`
--
ALTER TABLE `friends`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=4;

--
-- AUTO_INCREMENT для таблицы `message_users`
--
ALTER TABLE `message_users`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=5;

--
-- AUTO_INCREMENT для таблицы `users`
--
ALTER TABLE `users`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=18;

--
-- Ограничения внешнего ключа сохраненных таблиц
--

--
-- Ограничения внешнего ключа таблицы `friends`
--
ALTER TABLE `friends`
  ADD CONSTRAINT `accept_user_fk` FOREIGN KEY (`accepted_user_id`) REFERENCES
`users` (`id`),
  ADD CONSTRAINT `requested_user_fk` FOREIGN KEY (`requested_user_id`) REFERENCES
`users` (`id`);

--
-- Ограничения внешнего ключа таблицы `message_users`
--
ALTER TABLE `message_users`
  ADD CONSTRAINT `users_fk` FOREIGN KEY (`message_for_user`) REFERENCES `users`
(`id`);

--
-- Ограничения внешнего ключа таблицы `tokens`
--
ALTER TABLE `tokens`
  ADD CONSTRAINT `user_id_fk` FOREIGN KEY (`user_id`) REFERENCES `users` (`id`);
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```