МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА

ЛУЦКОВ МАКСИМ ПАВЛОВИЧ

Допускається до захисту:
завідувач кафедри
інформаційних технологій,
д-р техн. наук, доцент
_____ Тетяна НЕСКОРОДЄВА
«_____» _____ 2022р.

РОЗРОБКА КЛІЄНТСЬКОЇ ЧАСТИНИ ВЕБ-ФОРУМУ ДЛЯ ІТ-СПЕЦІАЛІСТІВ

Спеціальність 122 «Комп’ютерні науки»

Кваліфікаційна (бакалаврська) робота

Керівник:
Павло РИМАР, старший викладач
кафедри інформаційних технологій

Оцінка: _____ / _____ / _____
(бали за шкалою ЄКТС / за національною шкалою)
Голова ЕК: _____
(підпис)

Вінниця – 2022

АНОТАЦІЯ

Луцков М.П. Розробка клієнтської частини веб-форуму для IT-Спеціалістів. Спеціальність 122 «Комп'ютерні науки». Донецький національний університет імені Василя Стуса, Вінниця, 2021.

Основною метою виконання кваліфікаційної роботи є створення веб-форуму для IT-спеціалістів, оскільки, кількість людей даної предметної області дуже швидко росте, та підвищує попит на форуми даного типу.

У вступі наведено актуальність розробки саме веб-застосунка. У першому розділі розглянемо основних конкурентів та аналіз предметної області. У другому розділі розглянуто інструменти та технології, які полегшили створення даного застосунка. Третій розділ посвячено опису користувацького інтерфейсу, реалізації застосунка з клієнтської частини.

Ключові слова: Веб-форум, Website, Vue, IT-Спеціалісти, Server Side Rendering.

ABSTRACT

Lutskov MP Development of the client part of the web forum for IT specialists. Specialty 122 "Computer Science". Vasyl Stus Donetsk National University, Vinnytsia, 2021.

The main purpose of the qualification work is to create a web forum for IT professionals, as the number of people in this subject area is growing very fast, and increases the demand for forums of this type.

The introduction shows the relevance of developing a web application. The second section discusses the tools and technologies that have facilitated the creation of this application. The third section is devoted to the description of the user interface, the implementation of the application from the client side.

Keywords: Web Forum, Website, Vue, IT Specialists, Server Side Rendering.

ЗМІСТ

ВСТУП

В Україні в останні часи, дуже швидко розвивається IT-індустрія з абсолютно різних напрямків. IT-індустрія приковує все більше поглядів до себе. Велика кількість людей починають вивчати різні мови програмування та подібні речі, щоб почати працювати в індустрії інформаційних технологій.

Для вивчення цих технологій є велика кількість ресурсів, від онлайн курсів до книг. Але які б не були гарні курси чи книги, у людини, яка тільки починає чомусь навчатись в неї виникає велика кількість питань. На ці питання найкраще може відповісти досвідчений спеціаліст. І саме таку проблему може вирішити Веб-форум для спеціалістів різних рівнів. Кожен може задати питання, яке його бентежить або знайти вже готову відповідь на схоже питання по даній темі. Окрім отримування відповідей на свої питання.

Можна ознайомитись з іншими предметними областями чи іншими мовами програмування та побачити загальний рівень проблем. У наш час, дуже важливо бути продуктивними навіть в моменти коли ми не за комп'ютером та не можемо самостійно, щось писати та вчити. Але на веб-форумі, навіть з мобільного телефона можна підготовитись до майбутніх проблем, або ж допомогти комусь своєю відповіддю на питання користувача даної платформи.

Ринок IT-спеціалістів по всьому світі росте доволі високим темпом. Україна в тому числі не виняток. В зв'язку з тим, що в цілому користувачів різного програмного забезпечення дуже багато по всьому світу, відповідно і з'являється велика кількість людей, що повинна обслуговувати весь інтернет трафік. До нього входять Веб-додатки, програмне забезпечення, IOS та Android додатки. Все це потребує величезну кількість розробників, тестувальників та інших людей, які працюють на IT-індустрію. Особливо гостро бажання працювати в IT в людей з'явилось через пандемію Covid-19, тому що саме ця професія дає можливість працювати навіть не виходячи з дому. Оскільки пропозиція породжує попит, багато людей почали вивчати різні напрямки програмування тестування та інших областей, для влаштування на роботу. Навчання на

розробника, потребує багато часу та уваги. Далеко не всі готові пройти цей шлях від «Hello World» до дійсно великих та потужних програм. Саме на цьому шляху і з'являються основні питання етапів розробки. Звичайно в інтернеті є багато навчальних ресурсів, де можна знайти відповідь на своє питання в якісь статті, але вже давно на цьому ринку є спеціальні форуми для обговорення тих чи інших питань, які трапляються на етапі навчання чи навіть роботи розробника. Але враховуючи, кількість нових спеціалістів та людей, які навчаються даному ремеслу, підвищується і попит на нові, більш сучасні та потужні форуми, де можна знайти відповідь на своє питання в одному місці, не звертаючись до інших джерел. Тому саме наш форум, допоможе вирішити проблеми багатьох людей, з більш сучасним функціоналом та дизайном. Нижче приведений графік росту кількості IT-ФОПів по роках в Україні. [1]

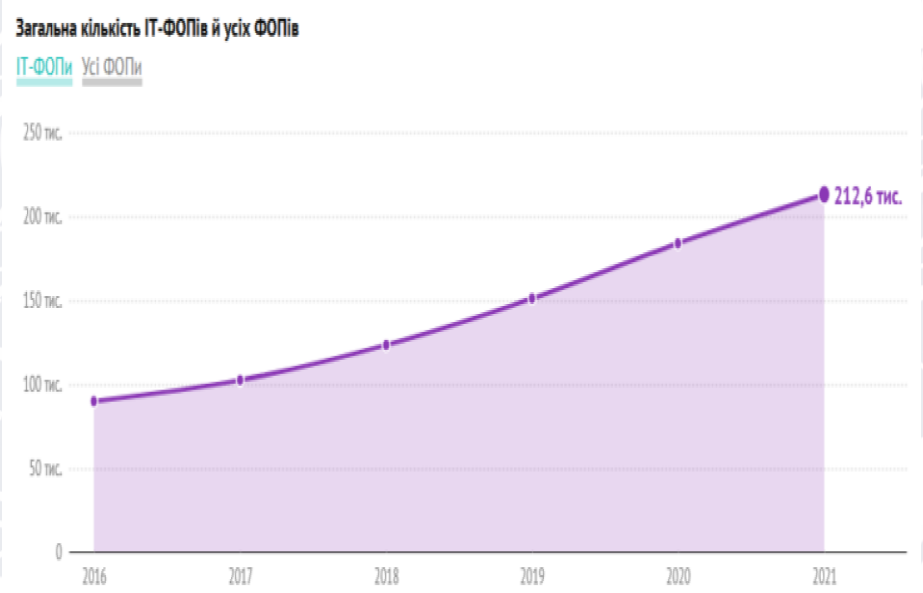

Рисунок 1 – Графік росту IT – спеціалістів

Як ми можемо бачити, аналізуючи графік, є стабільно високий ріст щороку, і з кожним наступним, він тільки збільшується, також зазначимо, що не всі IT-спеціалісти відкривають ФОП, тому реальна кількість спеціалістів в Україні є ще більшою.

У рамках бакалаврської роботи мета полягає у наступному:

1 Поглиблене вивчення технологій розробки сучасного веб-застосунку.

2 Огляд існуючих додатків на дану тематику, виявлення їх переваг та недоліків.

3 Написання клієнтської частини сайту та інтегрування з його серверною частиною.

Задачами дослідження є:

- Огляд існуючих веб-форумів для IT-спеціалістів, виявлення їх переваг і недоліків.
- Вивчення та використання спеціалізованого програмного забезпечення для розробки.
- Розробка веб-застосунку, що дозволить IT-спеціалістам допомагати початківцям у їх розвитку, та вирішення питань у сфері IT.

Об’єкт дослідження: веб-форум для запитань та відповідей на тему IT.

Предмет дослідження: процес розробки веб-форумів.

Структура роботи: кваліфікаційна робота складається зі вступу, трьох розділів, висновків, списку використаних джерел.

Робота містить 47 сторінок, 56 рисунків, 1 таблицю та список літератури з 23 джерел.

РОЗДІЛ 1
ПОСТАНОВКА ЗАДАЧІ З РОЗРОБКИ ВЕБ-ФОРУМУ

1.1 Постановка задачі

Необхідно реалізувати клієнтську частину веб-форуму. Застосунок повинен виконувати наступні функції:

1 Мати сучасний дизайн.
2 Бути повністю адаптивним та крос браузерним.
3 Логін та авторизація.
4 Створення запитань та можливість залишати відповіді під іншими запитаннями.
5 Фільтрація запитань по темах та позначках.
6 Ведення статистики кожного користувача та рівень його корисності на платформі.

1.2 Огляд існуючих аналогів

Перші популярні форуми почали з’являтись ще на початку 2000 років. Тому можемо розглянути декілька реально потужних інструментів для пошуку відповідей на свої питання.

Stack overflow [2]

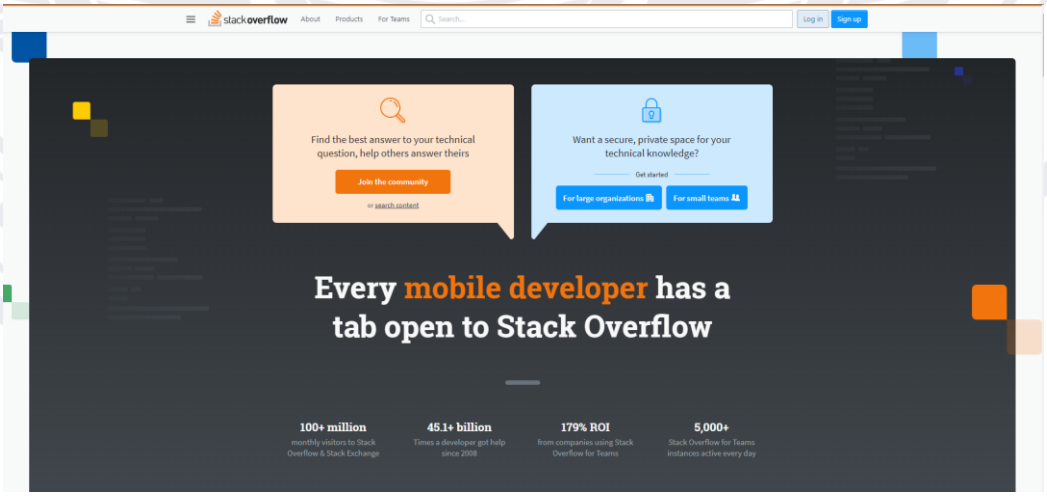

Рисунок 1.1 – Головна сторінка форуму «Stack overflow»

Stack overflow – найбільший форум для IT-спеціалістів. Його основна перевага це величезна база користувачів та питань з відповідей. Здається, що не має жодного програміста, який би не знав що таке Stack overflow. До недоліків можна віднести зайвий функціонал, який не дає одразу знайти потрібний тобі розділ, деякі функції застосунка не є безкоштовними.

CyberForum

Рисунок 1.2 – Логотип форуму «CyberForum»

CyberForum – досі доволі популярна платформа для запитань та відповідей розробників програмного забезпечення. Має велику базу відповідей на найбільш популярні запитання користувачів – це єдина сильна перевага, також весь контент є абсолютно безкоштовним. Щодо недоліків то це застарілий інтерфейс, який не зручно використовувати сучасному користувачу.

iForum.pro

iForum.pro – ще один сайт, з можливістю питань та відповідей. Нажаль на даний час має низьку активність користувачів, відповідно низька кількість вирішених актуальних запитань. Не адаптивний під сучасність дизайн та інтерфейс користувача.

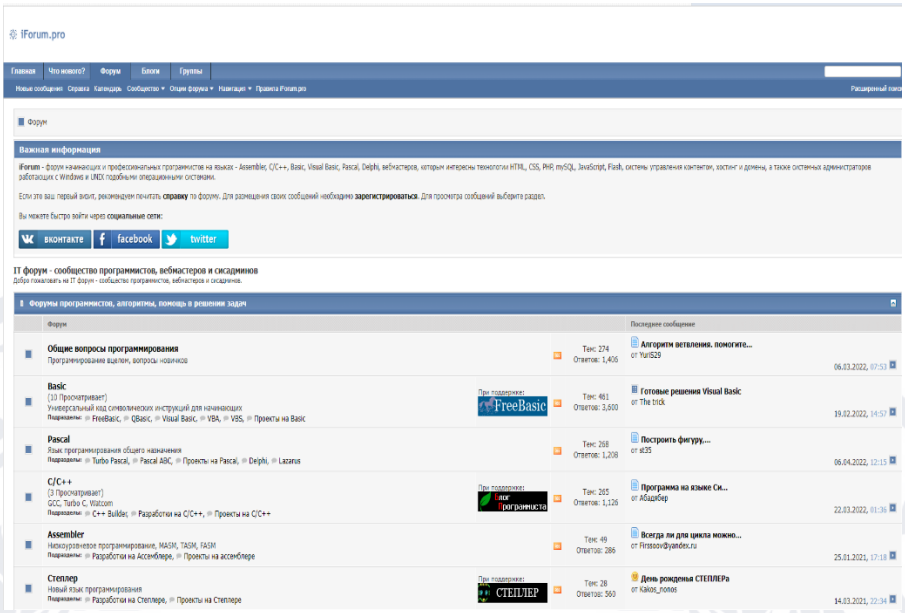

Рисунок 1.3 – Головна сторінка форуму «iForum.pro»

Розглянувши альтернативи, ми можемо скласти таблицю для порівняння та встановлення переваг та недоліків даних програм:

Таблиця 1.1 – Порівняння аналогів

	Stack overflow	CyberForum	iForum.pro
Велика кількість онлайн користувачів	+	-	-
User-Friendly	-	-	-
Можливість бачити свою статистику	+	+	-
Велика ймовірність знайти вже готову відповідь на своє питання	+	-	-

З таблиці ми можемо бачити, що навіть найкращий з сучасних форумів має свої недоліки, а актуальних форумів з сучасним дизайном зовсім не багато.

Висновок до розділу 1

У цьому розділі сформульовано постановку задачі з розробки веб-форму для ІТ-спеціалістів. Також було розглянуто, які функції має виконувати сучасний веб форум для програмістів, та проаналізували, які готові рішення є на просторах інтернету та їх актуальність. Також зрозуміли, що попит на таку платформу може бути доволі великим.

РОЗДІЛ 2
ОПИС ВИКОРИСТАНИХ ІНСТРУМЕНТІВ ТА ТЕХНОЛОГІЙ РОЗРОБКИ

2.1 Інструменти та утиліти для роботи

Web-програмування є одним з найбільш популярних серед розробників напрямком. Відповідно є великий попит на програмне забезпечення для роботи з веб-сайтами. Різні компанії пропонують свої рішення програмного забезпечення, яке прискорює роботу розробника, завдяки своєму розширеному функціоналу. Тому розглянемо основні IDE для розробки веб-додатків.

Sublime Text

Sublime text – це дуже швидкий текстовий редактор. Був розроблений у 2008 році під Windows. При появі Sublime text 2 з'явилась підтримка і інших операційних систем таких як Linux та MacOS. У 2013 році була представлена третя версія Sublime Text. IDE можна завантажити і користуватися безкоштовно, хоча ранні версії цього програмного забезпечення були платними. [3]

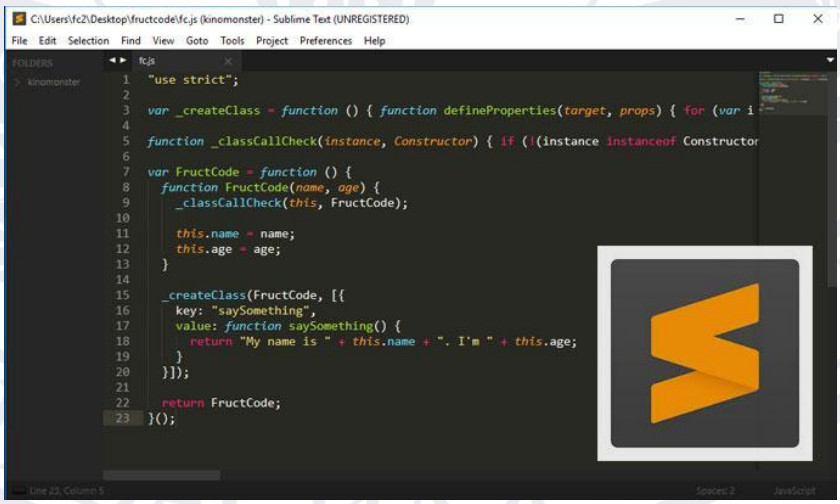

Рисунок 2.1 – Приклад JavaScript коду в Sublime Text 4

Найбільші його переваги – це розмір самої програми та його швидкість. В його переваги також можна додати влаштований «emit», зручна підсвітка коду та зручне дерево з файлів вашого проекту.

Середовище Sublime Text призначено скоріше для не великих проектів та навчання працювати з кодом. Також він добре працює на повільних комп'ютерах, що дає змогу займатись програмуванням навіть не маючи потужної техніки.

Звісно він має низку недоліків, які не дають йому право називатись найкращими IDE, тому що в першу чергу – це текстовий редактор, який працює з кодом. Можна сказати, що це зручний та простий текстовий редактор для коду на різних мовах програмування , які призначенні для розробки веб-сайтів.

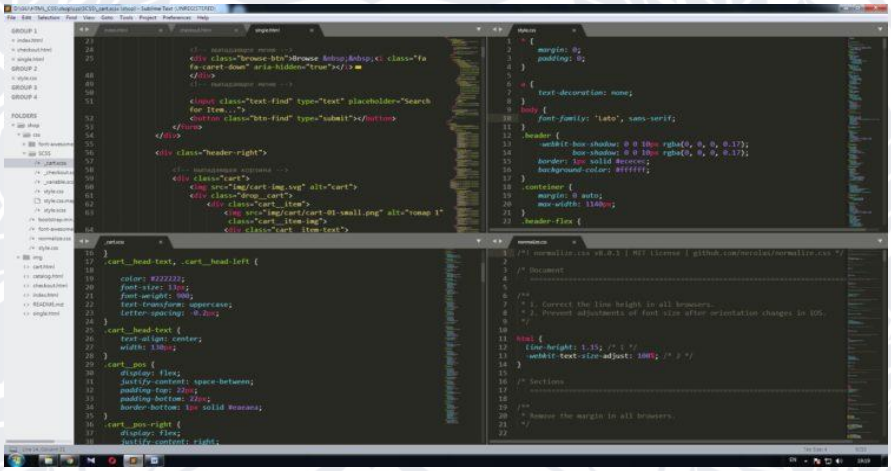

Рисунок 2.2 – Текстовий редактор Sublime Text

WebStorm

Webstorm – інтегроване середовище розробки від JetBrains, платформа розроблена на базі IntelliJ IDEA. Webstorm є спеціальної версією PhpStorm, основна різниця встановленні з коробки усі основні JavaScript розширення. Дане середовище підтримує більш розширений список мов програмування , таких як JavaScript, CoffeScript, TypeScript, Dart.

Переваги Webstorm в тому, що це потужне середовище, яке підтримує створення інструментів для збірок Gulp або NPM. Забезпечує підтримку з контролем версій GIT. Повністю налаштована робоча поверхня, яка дає можливість писати код, майже в будь-якій видозміні на розсуд користувача. Стартовий пакет разом з шаблонами для відправки.

Webstorm – це розумний редактор, він може проводити рефакторинг та генерацію кода, його аналіз під час роботи, можливість користуватись модульним тестуванням та запуск «Run & Debug». На ньому працюють доволі великі компанії в першу чергу. В ньому зручно працювати команді, в тому числі завдяки зручного терміналу для роботи з контролем версій. До його недоліків можна віднести високу ціну на продукцію JetBrains.

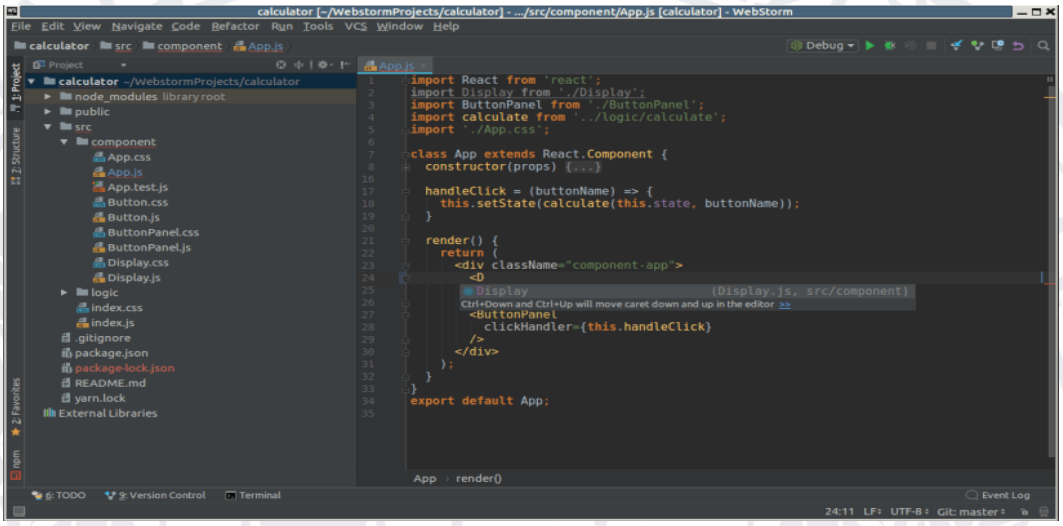

Рисунок 2.3 – Проект у WebStorm

Visual Studio Code

Visual Studio Code – застосунок для розробки веб-застосунків і програм для хмарних систем. Це безкоштовна IDE і доступний для всіх платформ Windows, Linux,OS X. [4] Вперше Visual Studio Code було представлено у квітні 2015 року на конференції Build 2015. В його основі використовувались напрацювання проєкту «Atom», яка належить компанії GitHub. В тому числі ця IDE є надбудовою над Atom Shell, яка в свою чергу використовує браузерний двигун Chromium і Node.js.

Рисунок 2.4 – Логотип Visual Studio Code

Переваги Visual Studio Code

- офіційна IDE для розробки веб-застосунків;
- середовище розробки підтримує роботу з декількома мовами програмування, до яких відносяться найпопулярніші – JavaScript C / C ++, Phyton, PHP.
- зручна робота із системами версії контролю;
- зручний та функціональний редактор коду;
- підказки до коду та автозавершення;
- велика кількість додаткових модулів, які роблять розробку швидшою;
- підтримка усіх нововведень;
- зручний інтерфейс та робота із структурою проекту.
- безліч «гарячих клавіш» для швидкості роботи.

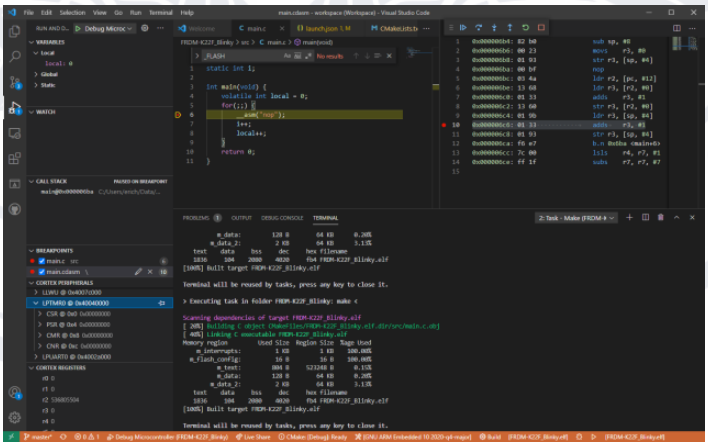

Рисунок 2.5 – Інтерфейс розробки Visual Studio Code

Отже, це найзручніший та найбільш ефективний, та потужний засіб для створення веб застосунків, а найголовніше він є повністю безкоштовним. Тому саме на цій IDE і буде вестись вся розробка нашого форуму. Нижче можна побачити рисунки, на яких видно ключові переваги Visual Studio Code

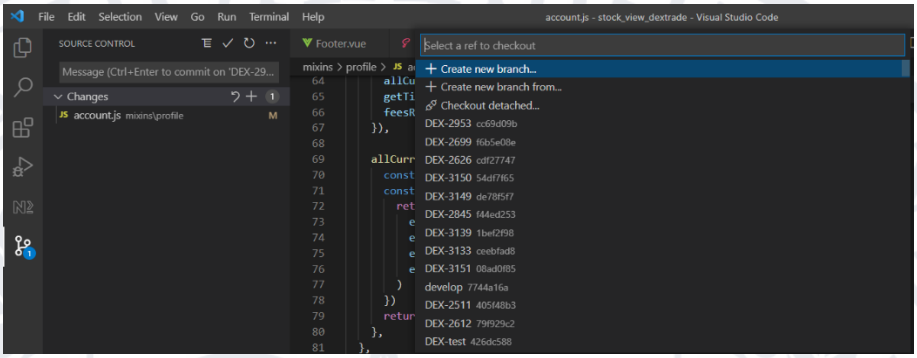

Рисунок 2.6 – Робота із Git в VS code

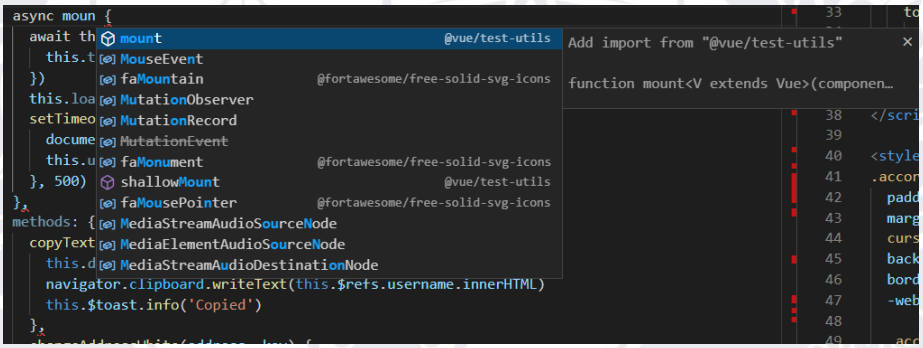

Рисунок 2.7 – Автозавершення відповідно до коду із бібліотек та framework`s

2.2 Технології

Основна мова програмування для веб розробки JavaScript, але оскільки він має свої недоліки, зараз популярні JavaScript фреймворки такі як React.js, Angular.js, Vue.js [5]

JS фреймворк – інструмент, який ми можемо використовувати для розробки сучасних веб-додатків. Раніше, коли інтерфейс застосунка був доволі простим, більшість розробників без проблем писали його на звичайному JavaScript, але с кожним роком інтерфейси та їх логіка стали надто складними для ванільного JavaScript. Тоді на допомогу і почали з’являтись фреймворки.

Розглянемо основні переваги JavaScript фреймворків:

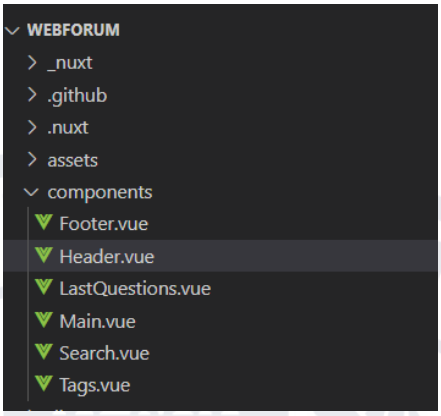

Рисунок 2.8 – Приклад компонентів на Vue.js

1. Web-компоненти. Компонентний підхід в розробці використовується майже у всіх мовах програмування ,окрім JavaScript. Замість того, щоб на кожній сторінці повторювати код, наприклад, шапки сайту, достатньо зробити компонент цієї шапки, і одної стрічкою підключати його на всіх необхідні файли, так працює з будь-якими частинами сайту, які ми можемо перевикористовувати.
2. Маршрутизація – сучасні фреймворки дають можливість використовувати систему навігації, без перезавантаження сторінок. Тобто є можливість створити багатосторінковий сайт, при цьому він буде мати всі переваги одно сторінкового, так звані SPA(Single Page Aplication)

```
<router-link :to="'/home'">Home</router-link>

<!-- same as above -->
<router-link :to="{ path: '/home' }">Home</router-link>

<!-- named route -->
<router-link :to="{ name: 'user', params: { userId: '123' }}">User</router-link>

<!-- with query, resulting in `/register?plan=private` -->
<router-link :to="{ path: '/register', query: { plan: 'private' }}">
  Register
</router-link>
```

Рисунок 2.9 – Приклад маршрутизації на фреймворках

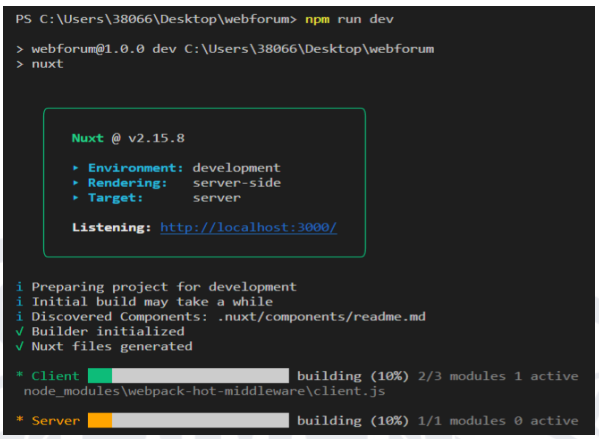

Рисунок 2.10 – Приклад процесу збірки компонентів

3. Зручні інструменти для збірки – компонентний підхід потребує інструменту, який буде збирати всі ці файли в один, та компілювати в розмітку доступну браузеру. Основні фреймворки мають свої збірники, які дуже легко встановлюються , та розширюють певний функціонал.

React.js – це бібліотека для створення користувацьких інтерфейсів.

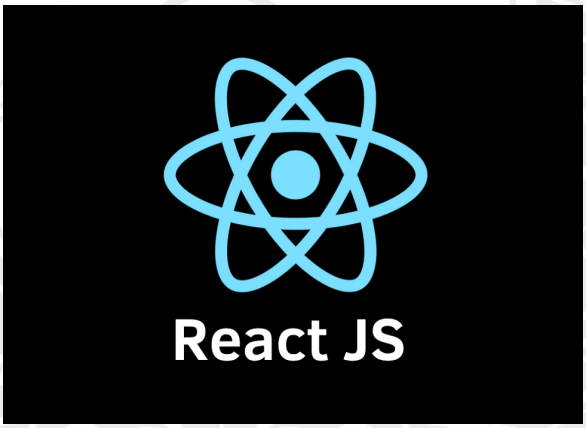

Рисунок 2.11 – Логотип React.js

Це навіть не повноцінний фреймворк. Він використовується для візуалізації та зв’язків з іншими бібліотеками. Щоб створювати веб сторінки його використовують в тандемі з ReactDOM. React дозволяє розробникам створювати великі веб-застосунки, які використовують дані, котрі змінюються з часом, без перезавантаження сторінки. Мета його полягає у тому, щоб бути швидким, масштабним та простим.

Переваги React.js

Одною із переваг React.js є Віртуальна об’єктна модель документа. Об’єктна модель документа (DOM) визначає деревовидну структуру HTML-документа, відправлений клієнту сервером після відповідного запиту. DOM представляє веб-сторінку в об’єктно-орієнтованому форматі, щоб мова програмування могли взаємодіяти із нею.

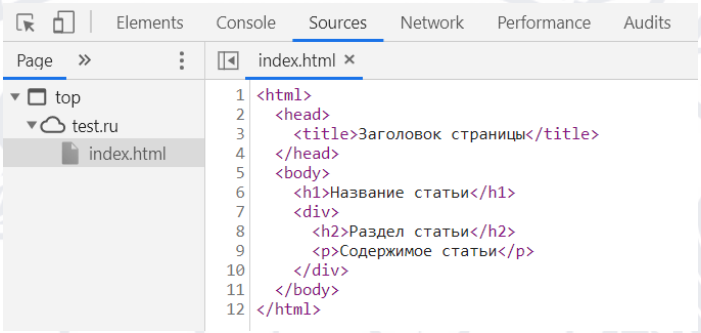

Рисунок 2.12 – Приклад DOM

Замість повільних і не комфортних взаємодій безпосередньо з реальною об’єктною моделлю документа. React.js взаємодіє з її полегшеною копією – віртуальний DOM, тому реальний DOM оновлюється тільки після взаємодії з віртуальним DOM.

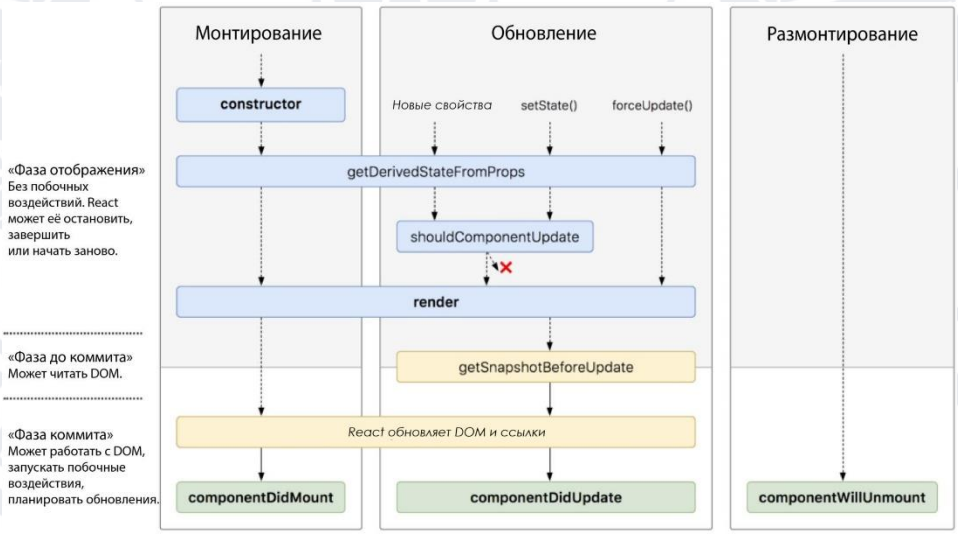

Рисунок 2.13 – Приклад DOM

При роботі з React.js створюються багаторазові компоненти: частіше всього, компонент інтерфейсу користувача можна використовувати в інших частинах коду або ж навіть в різних проєктах практично без змін.

Односторонній потік даних в React – це ще одна корисна функція. Такий потік даних також називають <зверху вниз> або <від батька до дитини >.

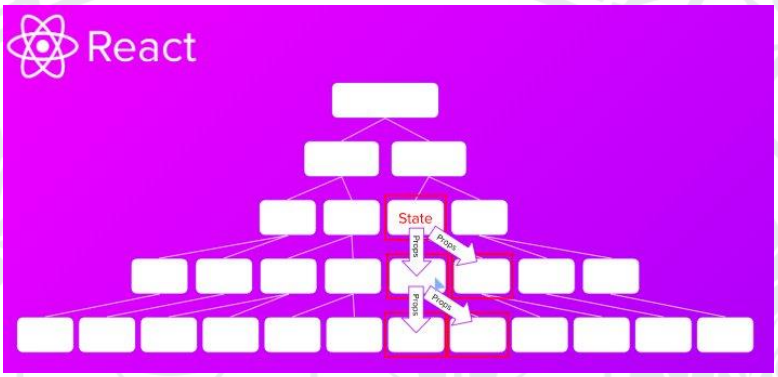

Рисунок 2.14 – Архітектура потоків даних

Браузерні інструменти React-розробника

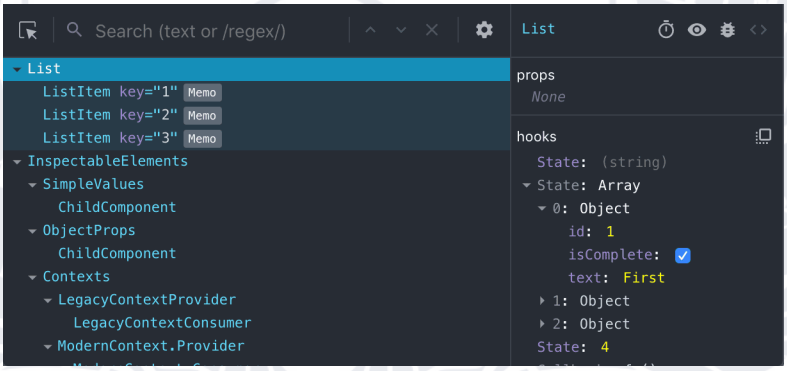

Рисунок 2.15 – React Developer Tools

React Developer Tools – це безкоштовне розширення для Chrome і Firefox, яке презентує цілий набір віджетів перевірки. Розширення спрощує налагодження, дозволяючи розробникам не тільки шукати по списку всіх компонентів, але і переглядати глибоко вкладені компоненти в браузері . [6]

Недоліки React.js

Як і будь яка інша технологія , React має :

1 Незадовільна документація
2 Заплутаний синтаксис JSX
3 Складність пошукової оптимізації
4 Фокусування на інтерфейсі користувача
5 Стандарти та складність використання
6 Складні оновлення
7 Велике використання оперативної пам’яті

```
class BookList extends Component {

    constructor(props) {
        super(props)
        //this.props = props;
    }

    renderList() {
        return this.props.books.map((book) => {
            return (
                <li key={book.title} className="list-group-item">{book.title}</li>
            )
        })
    }

    render() {
        return (
            <ul className="list-group col-sm-4">
                {this.renderList()}
            </ul>
        )
    }

}

// function is the glue between react and redux
function mapStateToProps(state) {
    // Whatever gets retrieved from here will show up as props inside
    // of book-list

    return {
        books: state.books
    }
}

function mapDispatchToProps(dispatch) {
    return bindActionCreators({selectBook: selectBook}, dispatch)
}

export default connect(mapStateToProps, mapDispatchToProps)(BookList);
```

Рисунок 2.16 – Синтаксис JSX

Як висновок, можна сказати, що дані недоліки є доволі критичні, і використання реакту не є найкращим рішенням.

Angular

Написаний на Type-script front-end фреймворк з відкритим кодом, який розробляється під керівництво Angular team, архівавано 18 серпеня 2021 року і Weyback Machine. До переваг його можна віднести наступне:

Декларативний стиль коду. При створенні шаблонів Angular.js застосовується декларативна парадигма програмування. Це робить код

легковажнішим, полегшує його читання і підтримку, оскільки описується необхідний кінцевий результат, а не всі кроки по його досягненню.

example_component.ts

```
import {Component, OnInit} from '@angular/core';

@Component({
  selector: 'example',
  templateUrl: 'example.component.html'
})

export class ExampleComponent implements OnInit {
  constructor() {
  }

  ngOnInit() {
  }
}
```

Рисунок 2.17 – Приклад коду на Angular

Використання директив. Директиви дозволяють сконцентруватися на опрацюванні логіки і працювати продуктивніше. Їх можна використовувати повторно, що також підвищує читабельність коду.

Рисунок 2.18 – Директиви на Angular.js

Висока швидкість розробки. При правильному підході за допомогою Angular.js можна швидко розробляти великі програми.

MVC з коробки. В AngularJS використовується схема MVC, що розділяє логіку, представлення та дані програми: [7]

Схема Модель-Вид-Контролер. Це дозволяє створювати односторінкові веб-програми (Single Page Application). Angular.js має службу $http, яка забезпечує взаємодію з віддаленими HTTP-серверами за допомогою XMLHttpRequest або JSONP.

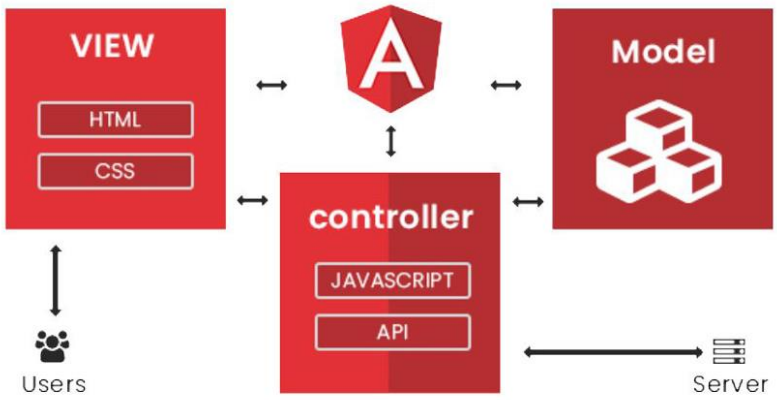

Рисунок 2.19 – Структура для SPA

Незважаючи на вищезгадані особливості AngularJS, у нього є кілька недоліків:

1. Складність освоєння. Труднощі зазвичай виникають у тих, хто раніше використовував бібліотеку jQuery, адже, на відміну від Angular.js, вона покладається на виконання маніпуляцій з деревом DOM.
2. Уповільнення роботи з використанням понад 2000 вотчерів (або слухачів подій).
3. Відсутність зворотної сумісності із другою версією. Зрозуміло, що ви можете почати підготовку свого коду до міграції вже зараз, але гарантій, що вона пройде без проблем, немає.

Vue.js

Vue.js - ще один з популярних JavaScript фреймворків для створення користувацьких інтерфейсів. Ядро Vue включає в себе бібліотеку ядра та маршрутизатор, а сам фреймворк можна навіть поступово інтегрувати в вашу систему, на відмінну від аналогів монолітів.

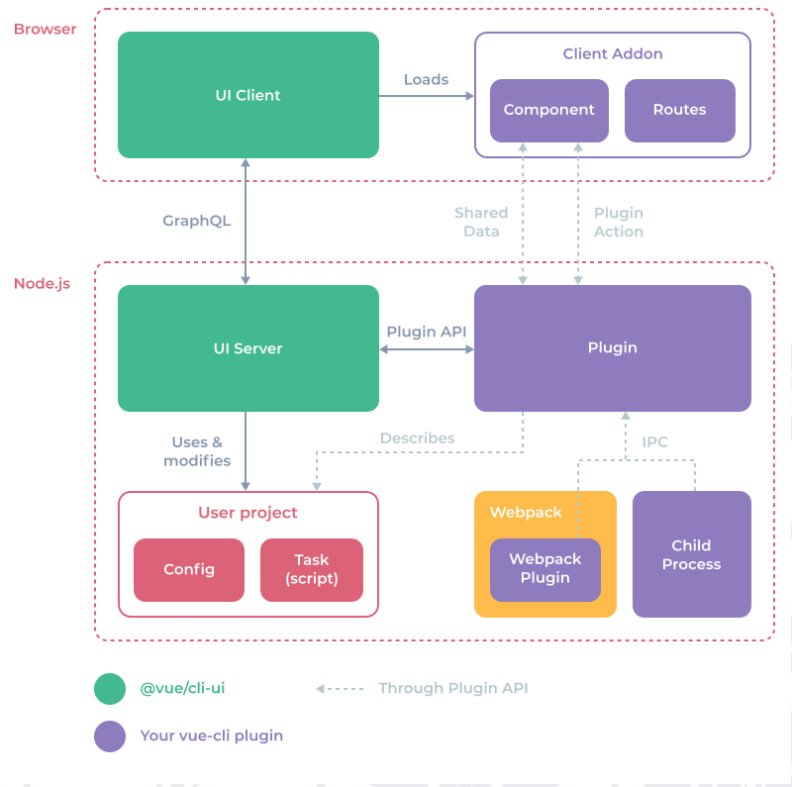

Рисунок 2.20 – Структура Vue.js

Директиви – спеціальні атрибути для додавання логіки елементам html. Директиви мають префікс ”-v” в якості аргументів можуть бути атрибути html або події. Розглянемо основні з них.

1. V-bind – динамічно зв'язується з одним або декількома атрибутами.
2. V-cloak – ховає «вусаті» вирази, поки не підтягнулися дані
3. V-if – умова для рендеру елемента
4. V-else – означає «else блок» для v-if
5. V-for – циклічно проходить масив об’єктів
6. V-model – зв’язує стан з input елементом
7. V-on – пов’язує слухача події з елементом
8. V-once – рендерит елемент тільки спочатку і більше не слідкує за ним
9. V-pre – не компілює елемент та його дочірні елементи
10. V-show – перемикає видимість елемента, змінюючи властивість CSS display
11. V-text – оновлює textContent елемент

```
  <div id="app">
    <div v-if="flag">foo</div>
    <div v-else>bar</div>
    <button @click="flag=!flag">Click Me!</button>
  </div>
<script type="text/javascript">
new Vue({
  el: '#app',
  data:{
    flag : true
  }
})
</script>
```

Рисунок 2.21 – Приклад роботи директив Vue.js

Компоненти – як в інших фреймворках, основною зручністю фреймворка є компоненти. Вони допомагають розширити основні html елементи і інтегрувати багато використовуваний код, тобто часто використовуванні UI елементи.

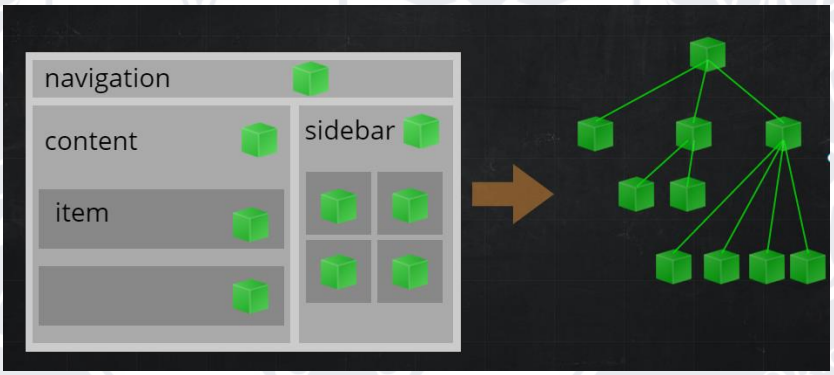

Рисунок 2.22 – Приклад структури компонентного підходу Vue.js

Переходи. Не меншої важливості, в сучасних веб-додатках потребують анімації та плавність елементів сторінки. Vue надає можливість різноманітних способів анімації елементів і не важливо в якому вони стані: коли вони на сторінці, чи коли оновленні , або навіть видаленні з DOM дерева. До них входять такі інструменти:

1. Автоматичне застосування класів для CSS-переходів і анімацій.
2. Інтеграція сторонніх бібліотек для CSS-анімацій, такий як ‘Animate.css’
3. Використання JavaScript для маніпуляції DOM
4. Інтеграція сторонніх JavaScript бібліотек для анімацій, такий як “Velocity.js”

html

```
<div id="demo">
  <button @click="show = !show">Toggle show</button>
  <transition name="bounce">
    <p v-if="show">Look at me!</p>
  </transition>
</div>
```

js

```
new Vue({
  el: '#demo',
  data: {
    show: true
  }
})
```

Рисунок 2.23 – Використання анімацій у Vue.js

Роутинг. у Vue.js є спеціальний пакет Vue-router , який відповідає за маршрутизацію компонентів. Його суть в підтримці вкладних маршрутів до вкладних компонентів, пропонує спрощене API для хуків навігації. [8]

app.js

```
import Vue from 'vue'
import VueRouter from 'vue-router'
import App from './app.vue'
import ViewA from './view-a.vue'
import ViewB from './view-b.vue'

Vue.use(VueRouter)

const router = new VueRouter()

router.map({
  '/a': { component: ViewA },
  '/b': { component: ViewB }
})

router.start(App, '#app')
```

Рисунок 2.24 – Приклад використання маршрутизації.

Ajax – запити. В Vue.js існує розширення vue-resource – воно дає можливість створювати веб-запити і оброблювати відповіді на них за допомогою MLHttpRequest чи JSONP. До його особливостей також входять підтримка Promise API и URI шаблонів. [0]

```
{
  // GET /someUrl
  this.$http.get('/someUrl').then((response) => {
    // успех
  }, (response) => {
    // или ошибка
  });
}
```

Рисунок 2.25 – Використання vue-resource

Управління станом через Vuex. Vuex – це бібліотека та паттерн управління станом для додактів на Vue.js. По суті це централізований стан для всіх компонентів і застосунка і правила, які задають передбачувані зміни стану. [11]. Розглянемо рисунок нижче, можемо побачити застосунок, разом з Vuex та наступними долями:

Стан (State) – він є єдиним джерелом даних для компонентів.

Сам компонент, він лише є декларативним відображенням стану.

Дія(Actions), яка ловить подію, що відбулась. Збирає дані з інших API та застосовує спеціальні мутації.

Мутації (Mutations) – це частина, яка може змінювати стан подай, і отримав данні від Actions, використовує їх на стані.

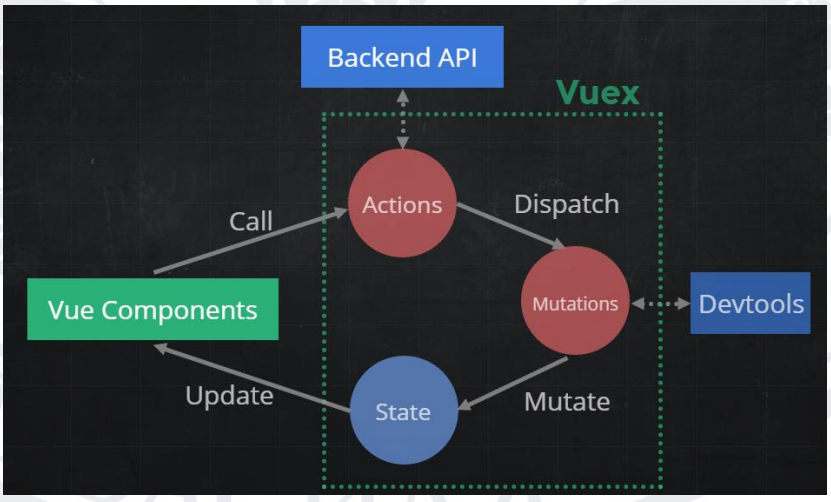

Рисунок 2.26 – Приклад роботи структури Vuex

Як висновок до вибору фреймворків, можна зрозуміти, що це найбільш прогресивний та потужний інструмент для розробки даного сайту. Звісно фреймворк, це не є єдиною технологію , яку можна використовувати самостійно.

HTML 5 – це мова розмітки гіпертексту. Вона є стандартизованою для розробки веб-сторінок. По суті це фундамент сайтів. Ми будем використовувати найкращі практики при розробці форуму, такі як валідність , семантика та доступність HTML- коду [19]

Рисунок 2.27 – Логотип HTML

CSS 3 (каскадні таблиці стилів) – спеціальна мова для задавання стилістичних особливостей блоків та тексту. Окрім вигляду, дає можливість змінювати розташування цих файлів на сторінці. [20]

Рисунок 2.28 – Логотип CSS 3

JavaScript – динамічна мова програмування, реалізація стандарту ECMAScript. Найбільшу реалізацію отримала в створенні різних сценаріїв на

веб-сторінках, що на дає на стороні клієнта взаємодіяти з користувачем, керувати браузером, тощо. [12]

Рисунок 2.29 – Логотип JavaScript

Для створення макетів сторінок використовувався редактор Figma. Це векторний онлайн-сервіс розробки інтерфейсів та прототипування з можливістю організації спільної роботи, що розробляється однойменною компанією. Працює у двох форматах: у браузері та як клієнтський додаток на десктопі користувача. Зберігає онлайн-версії файлів, з якими працював користувач.[13] В ньому можна розробляти дизайн сторінок, який за допомогою інструментів програми можна зручно переносити стилі шрифтів або блоків на реальну сторінку сайту.

Рисунок 2.30 – Логотип «Figma»

Висновок до розділу 2

В цьому розділі було розглянуто та зроблено порівняльний аналіз всіх необхідних інструментів та технології для вирішення задачі. Зрозуміли різницю між тими чи іншими технологіями та обрали найбільш правильні технології для розробки веб-форуму.

РОЗДІЛ 3
ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБ-ФОРУМУ

3.1 Опис дизайну користувача

Однією з умов нашої задачі було мати сучасний та красивий дизайн. Яким буде приємно користуватись абсолютно різним групам користувачів, таким як жінкам та чоловікам, молодим та більш дорослим людям. Увесь дизайн було виконано в Figma.

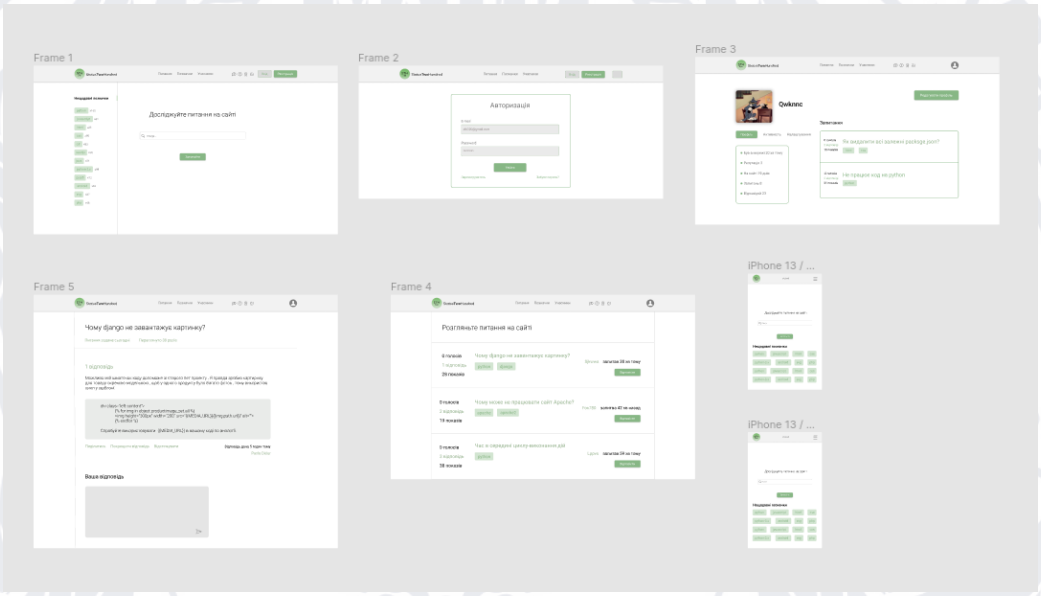

Рисунок 3.1 – Основні сторінки дизайну в Figma

Задля приємної взаємодії користувача з сайтом було підібрано палітру кольорів, яка використовується для задання єдиного стилю усім сторінкам.

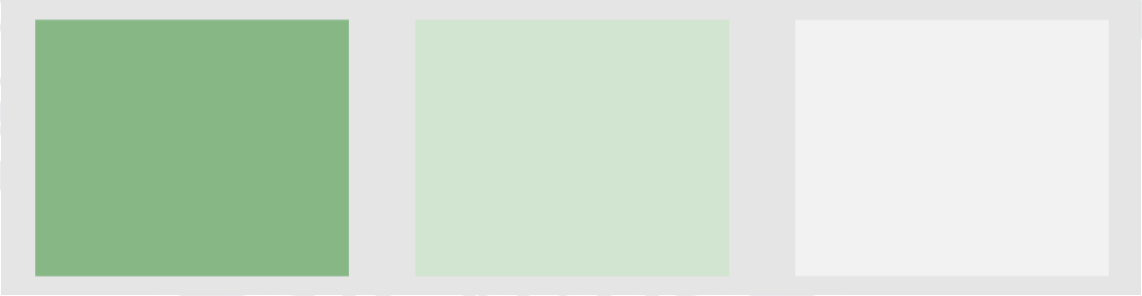

Рисунок 3.2 – Кольорова палітра для блоків сайту

З кожним роком дедалі більше користувачів інтернету використовують мобільні пристрої для перегляду різної інформації. ІТ – спеціалістам це є не

менш актуальним, адже можна отримувати інформацію, навіть будучи, не біля комп’ютера, або не дома, у громадському транспорті тощо.

Тому наш форум, повинен бути повністю функціональним як на великих стаціонарних моніторах, маленьких ноутбуках та абсолютно різних смартфонах та планшетних пристроях. Саме по цій причині, був розроблений мобільний дизайн, який адаптується до всіх типів пристроїв, наша платформа повинна бути повністю адаптивна та крос браузерна, приклад головної сторінки можемо бачити на рисунку нижче. [14]

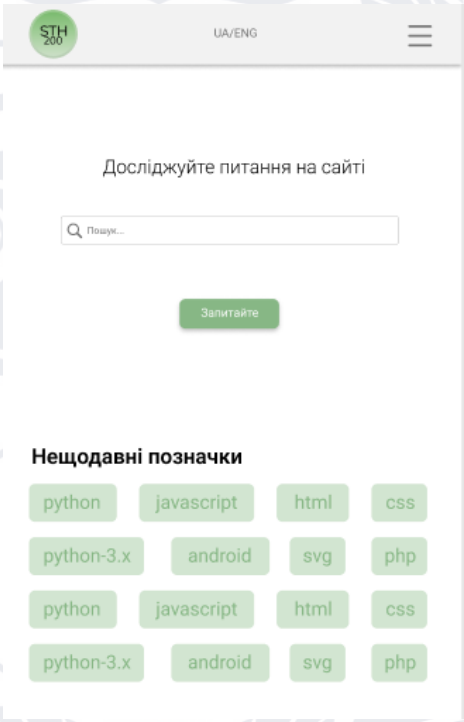

Рисунок 3.3 – Мобільна версія головної сторінки нашого сайту

Як висновок до дизайну форуму, можемо зазначити, що він виконаний в стилі мінімалізму, що відповідає канонам сучасного дизайну. З коректно підібраною кольоровою палітрою. Дизайн підготовлений для різних типів пристроїв, що є необхідністю для успішного веб-сайту.

3.2 Програмна реалізація

Для початку програмної реалізації необхідно створити архітектуру нашого проекту. До архітектури будемо відносити такі частини:

1. API [15]
2. Верстка дизайну
3. Написання логіки авторизації
4. Розробка можливості додавання та редагування відповідей та запитань
5. Routing.

Рисунок 3.4 – Структура проєкту

API – це опис методів, за допомогою яких можна взаємодіяти з сервером. Його будемо використовувати для написання логіки сайту. Більш детальніше розберемо у реалізації авторизації та додавання питань і відповідей.

Для розробки сайту за допомогою API ми відповідно потребуємо певного Server-side rendering [16]. Це дає можливість розвертати проект на стороні сервера, і відправляти вже зібрані дані до клієнта. Все це ми реалізовуємо завдяки Nuxt.js.

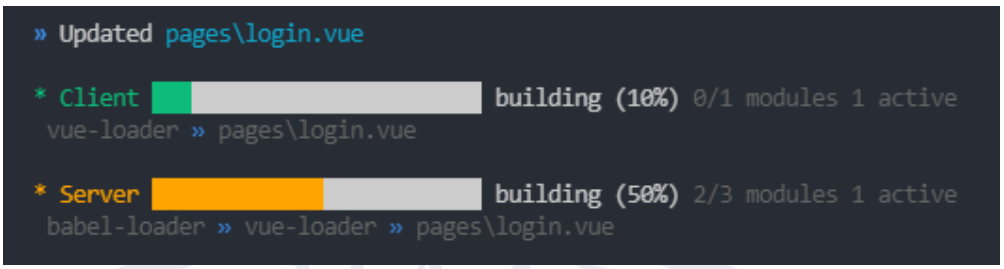

Рисунок 3.5 – Процес Server-side rendering

Верстка дизайну. Для верстки даного проекту будемо взаємодіяти з Figma для швидкості розробки. Завдяки їй ми можемо швидко підбирати основні стилі текстів та блоків.

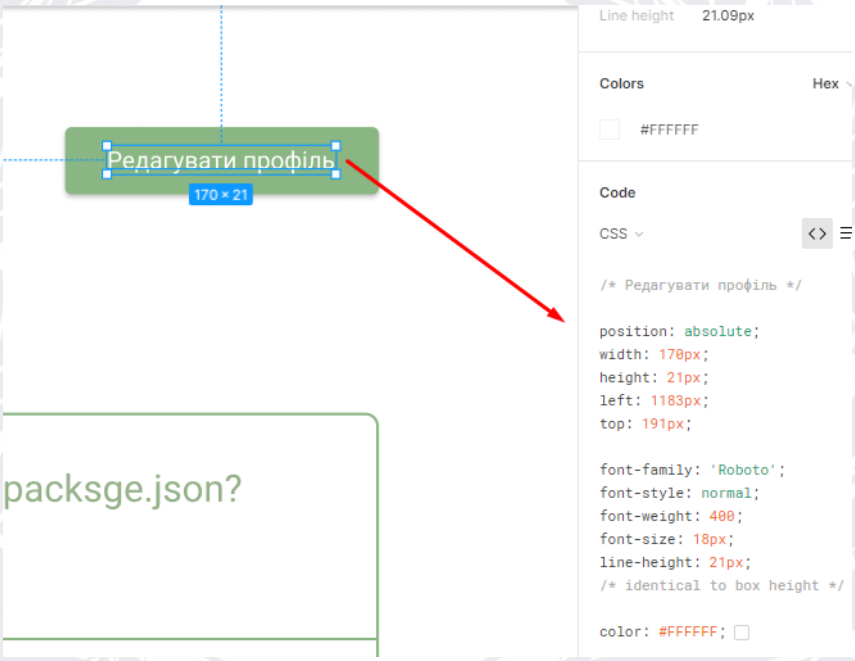

Рисунок 3.6 – Приклад взаємодії Figma з кодом CSS

Для написання розмітки використовуємо HTML5 та методологію БЕМ. БЕМ – це методологія, за допомогою якої ми можемо розбивати інтерфейс на відповідні блоки. БЕМ потребує спеціальних назв та синтаксису блоків, які ми відповідно використовуєм.[17] Зараз можемо побачити розмітку сторінки авторизації.

```
<div class="login">
  <div class="container">
    <div class="login__body">
      <h2>Авторизація</h2>

      <form action="/">
        <div class="login__email">
          <h3>E-mail</h3>
          <input
          class="login__input"
            v-model="userEmail"
            type="text"
            placeholder="Example@gmail.com"
          />
        </div>
        <div class="login__email">
          <h3>Password</h3>
          <input
          class="login__input"
            v-model="userPassword"
            type="text"
            placeholder="Example@gmail.com"
          />
        </div>
        <button class="btn__submit" type="submit">Увійти</button>
      </form>
      <div class="login__tooltip">
        <nuxt-link :to="'/registration'">Зареєструватись</nuxt-link>
        <nuxt-link :to="'/restore'">Забули пароль?</nuxt-link>
      </div>
    </div>
  </div>
</div>
```

Рисунок 3.7 – Лістинг коду сторінки Авторизації

Процес розробки авторизації.

Для розробки авторизації ми будемо використовувати API з endpoint. За допомогою Vue.js ми можемо отримувати дані які будуть вписуватись, у поля для авторизації. Та за допомогою Axios запитів, відправляти дані з полів до сервера, та перевіряти чи є користувач зареєстрований, якщо ми отримаємо статус запита «succsess» то користувач авторизувався та отримує доступ до системи та свого облікового запису. Якщо ні буде запропоновано зареєструватись, або ж перевірити правильність введення своїх даних.

```
actions: {
  login({commit}, user){
    return new Promise((resolve, reject) => {
      commit('auth_request')
      axios({url: 'http://localhost:3000/login', data: user, method: 'POST' })
      .then(resp => {
        const token = resp.data.token
        const user = resp.data.user
        localStorage.setItem('token', token)
        axios.defaults.headers.common['Authorization'] = token
        commit('auth_success', token, user)
        resolve(resp)
      })
      .catch(err => {
        commit('auth_error')
        localStorage.removeItem('token')
        reject(err)
      })
    })
  }
}
```

Рисунок 3.8 – Лістинг коду запита на логінізацію

Відповідно, щоб зробити вихід з особистого акаунта ми будемо використовувати метод logout(), будемо видаляти з local Storage наш токен логінізації.

```
logout({commit}){
  return new Promise((resolve, reject) => {
    commit('logout')
    localStorage.removeItem('token')
    delete axios.defaults.headers.common['Authorization']
    resolve()
  })
}
```

Рисунок 3.9 – Лістинг коду методу виходу із системи

Процес розробки додавання питань та відповідей.

Для створення власного питання, ми використовуємо поля для вводу тексту, та вставлення рисунків для більш детального опису вашої проблеми.

Після натискання кнопки створити запитання, наш код відправляє запит, з питанням на сервер, сервер повертає це питання для відображення на сайті. Після чого це питання відобразиться на основній сторінці з питаннями.

```
            <div class="quest__form">
                <input v-model="formQuest">
            </div>
        </div>
    </div>
</template>

<script>
export default {
  data() {
    return {
      formQuest: '',
    }
  },
}
</script>
```

Рисунок 3.10 – Отримання даних з форми до змінної «formQuest»

Routing

Оскільки наш сайт є багатосторінковим, на ньому встановлені переходи між сторінками. Зазвичай, перехід на сторінку, потребує нового рендерингу сторінки, що призводить до тривалого завантаження контенту та функціоналу певної сторінки.

Оскільки ми використовуємо Nuxt.js та Server-side rendering. Ми можемо використовувати влаштовану маршрутизацію. Це нам дає можливість лише один раз завантажити дані сайту, та виконувати перехід по сторінках без додаткового завантаження сторінки. Перехід та відображення даних сайту буде відбуватись моментально, що значно покращує користувацький досвід з платформою.

Для роутингу використовується спеціальний тег “nuxt-link” з атрибутом “:to()”[18]. Також до переваг Server-side rendering можна віднести кращу роботу SEO роботів, які можуть швидше та краще проіндексувати нашу сторінку.

```
<div class="header__menu mobile__none">
  <ul>
    <li><nuxt-link to="/login">Питання</nuxt-link></li>
    <li><nuxt-link to="/mark">Позначки</nuxt-link></li>
    <li><nuxt-link to="/people">Учасники</nuxt-link></li>
  </ul>
</div>
<div class="header__login">
  <nuxt-link class="btn_in header__btn" to="/login">Вхід</nuxt-link>
  <nuxt-link class="btn_out header__btn" to="/registration">Реєстрація</nuxt-link>
</div>
</div>
```

Рисунок 3.11 – Робота з роутингом

Для роботи роутинга, в Nuxt.js [22], всі сторінки повинні бути розміщенні у теці Pages, саме тоді ми зможемо використовувати дану технологію переходів між сторінками.

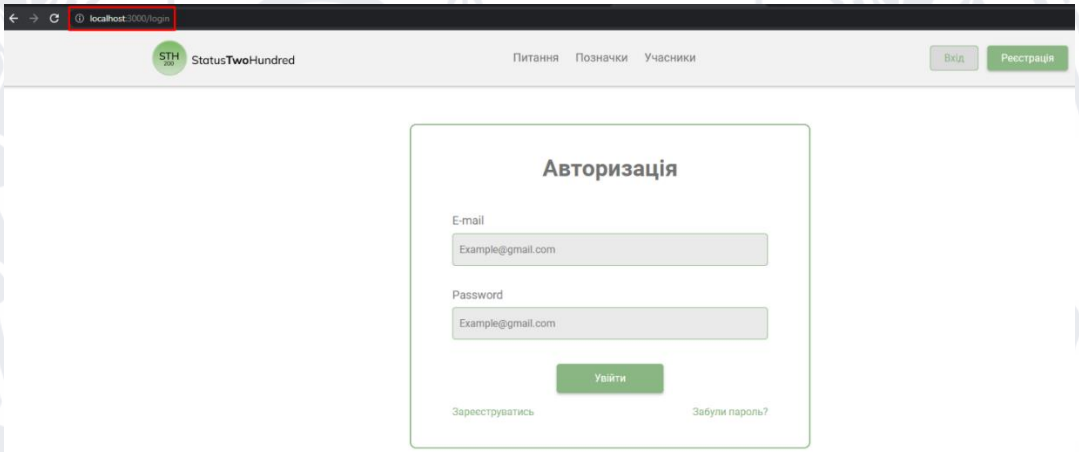

Рисунок 3.12 – Демонстрація шляху до сторінки за її назвою

Відповідно, назва файлу буде дорівнювати шляху, по якому можна звернутись до нашого файлу. Компоненти відобразити без сторінки ми не зможемо.

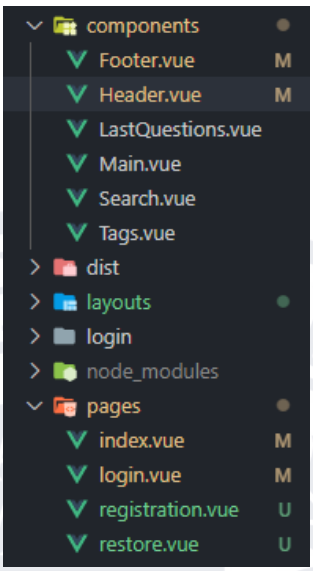

Рисунок 3.13 – Структура компонентів та сторінок

Оскільки, ми використовуємо компонентний підхід до розробки сайту, у нас є певні компоненти які повинні використовуватись на кожній сторінці без виключення, наприклад як header та footer [21]. Для цього ми також створили шаблон, який буде використовуватись автоматично, header та footer, між ним буде знаходитись весь контент нашого компоненту або сторінки.

layouts > default.vue > ...

```
<template>
  <div class="wrapper">
    <Header />
    <div class="main">
      <nuxt />
    </div>
    <Footer />
  </div>
</template>

<style lang="scss" scoped>
.wrapper {
  min-height: 100%;
  display: flex;
  flex-direction: column;
}

.main {
  flex: 1 1 auto;
}
</style>
```

Рисунок 3.14 – Лістинг коду нашого шаблону

Як можемо бачити, в нас є підключенні два компоненти header та footer. Між ними в тегу з класом “main” є тег <nuxt /> саме замість нього буде відображатись весь контент сторінки, на яку буде здійснено перехід.

3.3 Опис функціональної частини сайту

Для користувачів, які вперше зайдуть на сайт, буде переадресація на сторінку Авторизації та Реєстрації.

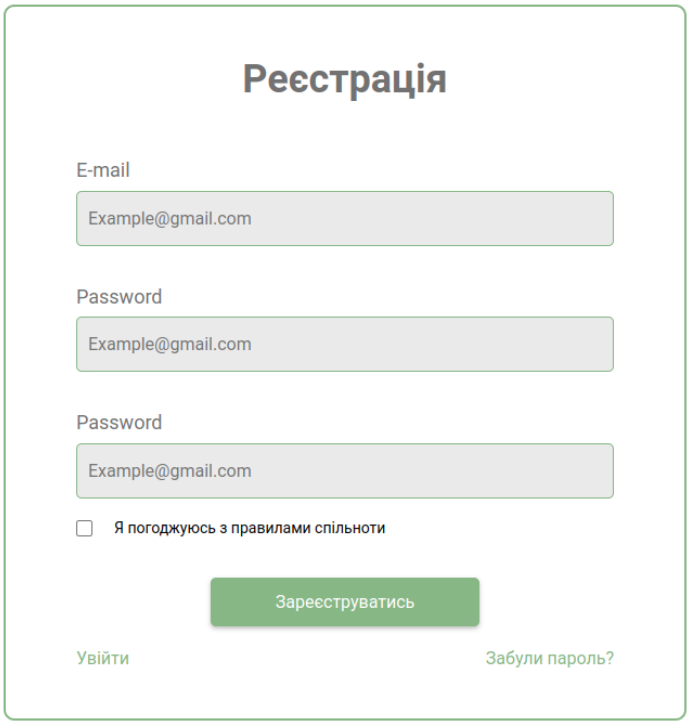

Рисунок 3.15 – Вікно Реєстрації

Можемо побачити поля вводу електронної пошти та пароля, чек бокс для погодження з правилами нашої спільноти. Також доступні поля для відновлення пароля та входу, при наявному акаунті.

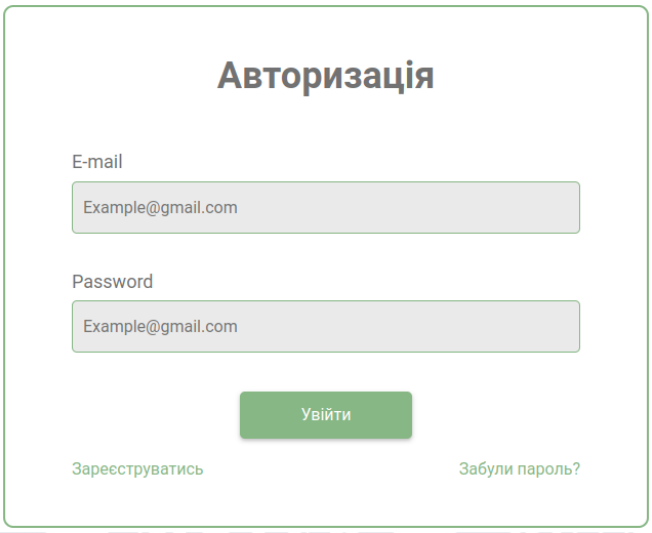

Рисунок 3.16 – Блок авторизації

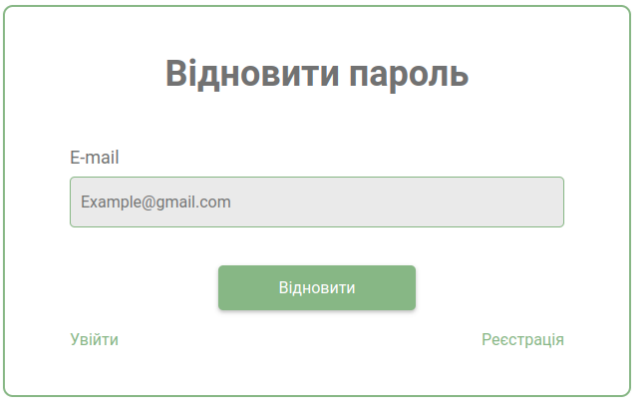

Рисунок 3.17 – Блок для відновлення паролю

При введені зареєстрованої пошти, користувач може отримати унікальне посилання на пошту, після переходу по даному посиланню, можна буде змінити пароль на новий. Після входу або реєстрації на даному форумі, користувач матиме доступ до наступного функціоналу:

1. Особистий кабінет з інформацією по особистому профілю, побачити список своїх поставлених запитань та відповідей на запитання інших користувачів. Має можливість змінити особисті данні, та налаштування конфіденційності. Також, доступна певна статистика профіля, така як остання активність користувача, його рівень репутації (бали, які можна отримати за корисні відповіді на запитання інших учасників), як давно

користувач використовує дану платформу та кількість його запитань та відповідей за весь час. Саме ці функціональні особливості платформи, будуть залучати більшу кількість користувачів, яким буде цікаво відповідати на запитання інших людей, та піднімати свою репутацію, як спеціаліста

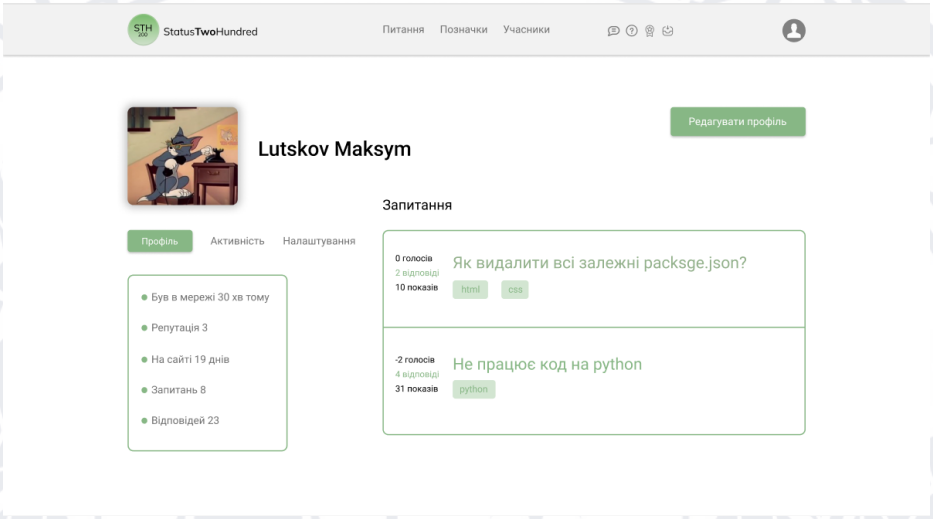

Рисунок 3.18 – Вигляд особистого кабінету користувача

2. Користувач матиме можливість задавати питання з можливістю вставлення коду, та позначок, за допомогою яких інші користувачі, зможу відповідати на питання, по темах, в яких вони розбираються найкраще, що економить час, та є найбільш ефективним способом, фільтрувати питання. [23]

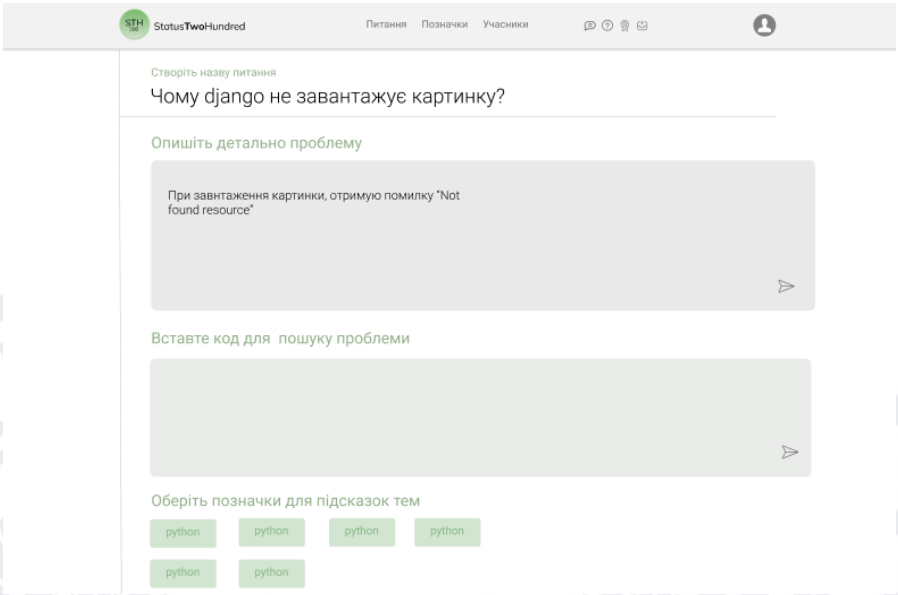

Рисунок 3.19 – Вигляд створення питання на форумі

3. Можливість фільтрації питань за позначками та ключовими словами. Користувач на головній сторінці, може знайти питання, яке його цікавить, або ж продивитись усі питання, які були позначенні певними позначками тем. Варто зазначити, що пошук питань за позначками, або по ключових словах, можуть використовувати навіть не зареєстровані користувачі, але давати відповідь на питання, необхідно авторизуватись на нашій платформі. До того ж можна відкрити загальний список питань, де в першу чергу видаватимуться найновіші запитання користувачів.

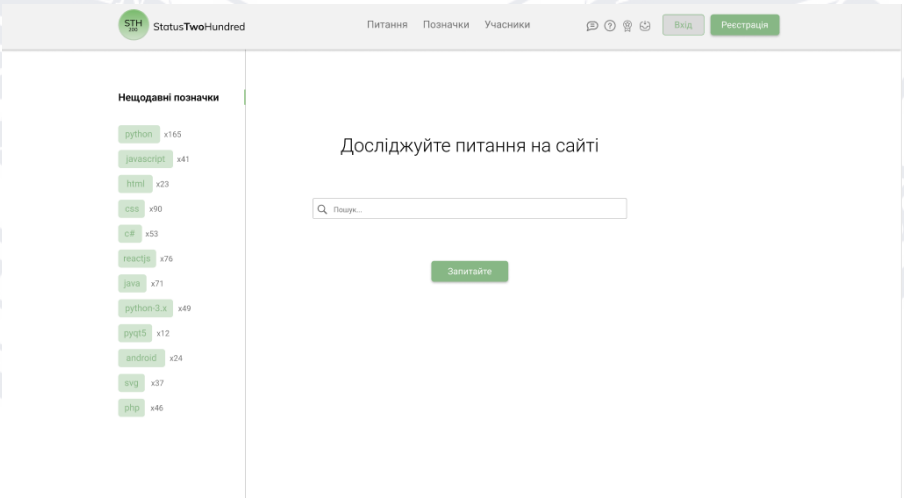

Рисунок 3.20 – Головна сторінка з пошуком, та позначками

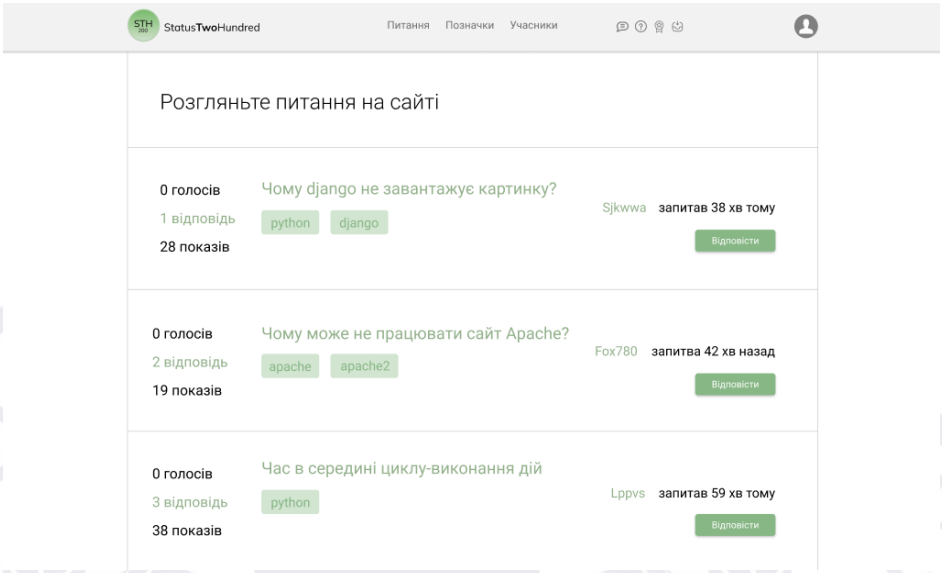

Рисунок 3.21 – Список найновіших питань у розділі «Питання»

4. Можливість давати відповіді на запитання інших користувачів. Перейшовши, на сторінку окремого питання, можна побачити детальний опис проблеми користувача, та відповіді інших користувачів, якщо вони були. Також можна побачити поле, вводу тексту, та вставки для коду, щоб дати більш розвернену відповідь на запитання користувача, якщо ваша відповідь буде корисною, і вона буде допомагати користувачам платформи, ваш рейтинг буде підвищено, і таким чином довіра до вашої відповіді буде більша, і відповідь буде виділена зеленим кольором, як найбільш корисна.

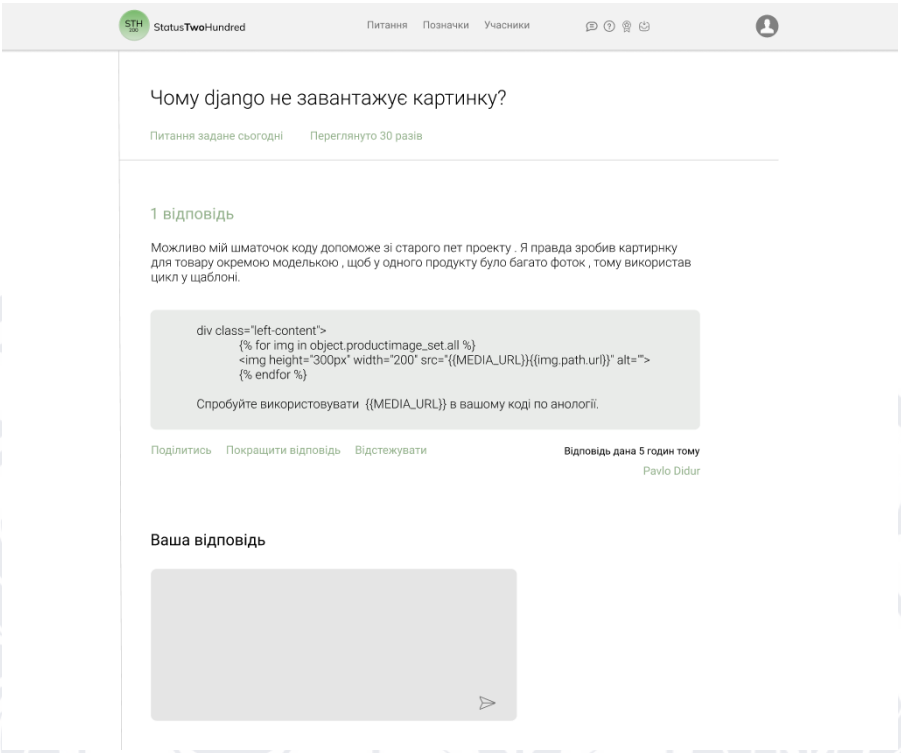

Рисунок 3.22 – Вигляд інтерфейсу в середині питання

Також можемо бачити як давно була додана відповідь, можливість підписатись на зміни в цьому питанні та слідкувати за іншими відповідями. Можливо поділитись відгуком з іншими користувачами.

Висновок до розділу 3

В цьому розділі було описано, основні дизайнерські рішення, для успішності візуальної частину сайту. Також був наведений весь функціонал даної платформи, що робить її доволі конкурентоспроможною на фоні інших схожих форумів.

ВИСНОВКИ

В результаті роботи, було виконано поставленні перед нами завдання. Ми дослідили актуальність даної платформи, та її майбутню конкурентоспроможність з іншими платформами. Актуальність даної роботи, супроводжується збільшенням цікавістю людей до IT, та бажання приєднатись до цієї спільноти IT – спеціалістів, що в свою чергу дає попит на платформи даного типу. Така платформа вирішує проблеми користувачів, завдяки відповідям інших користувачів, які також на своєму шляху ставили собі подібні запитання. Також до актуальності можна віднести мобільність платформи. Величезна кількість інформації поглинається з мобільних пристроїв, і дана платформа, будучи адаптивною, може використовуватись навіть в дорозі чи в інших сценаріях, коли комп'ютер не доступний, для навчання своєї улюбленої справи.

Для розробки клієнтської частини були досліджені методи дизайну сайтів. Розібрано методики проектування архітектури. Основними задачами даної роботи було вивчення методів розробки сучасних веб-застосунків, огляд існуючих форумів, та визначення їх переваг та недоліків. Та основна задача – це розробка веб-форуму, який мав би переваги перед конкурентними платформами. Задача виконана та протестована відповідно до принципів UI/UX.

В результаті тестування було виявлено, кілька не великих проблем та виконано їх вирішення. Весь функціонал працює та відповідає опису та принципам розробки. Набуті навички по впровадженні найновіших патернів та концепцій розробки сучасних сайтів.

У ході розробки було використано наступні інструменти та мови програмування: Visual Studio Code, Мова розмітки та стилів HTML, CSS, JavaScript, фреймворк Vue.js , збірник Nuxt, Менеджер стану VueX , фреймворк для CSS – SCSS.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Скільки IT-фопів в Україні [Електронний ресурс] – Режим доступу https://dou.ua/lenta/articles/how-many-devs-in-ukraine-2020/

2. StackOverFlow – найпопулярніший веб-форум [Електронний ресурс] – Режим доступу до ресурсу https://stackoverflow.com/

3. Редактор Sublime Text [Електронний ресурс] – Режим доступу до ресурсу: https://www.sublimetext.com/

4. «Visual Studio Code» - засіб для створення, редагування та зневадження сучасних веб-застосунків і програм для хмарних систем [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Visual_Studio_Code

5. Книга David Herron “Node.js Web Development – 4”

6. React Developing tools [Електронний ресурс] Режим доступу до ресурсу: https://chrome.google.com/webstore/detail/react-developer-tools/fmkadmapgofadopljbjfkapdkoienihi?hl=ru

7. MVC в Ангуляр [Електронний ресурс] – Режим доступу до ресурсу: https://dev.to/angular/understanding-mvc-services-for-frontend-angular-3e8a

8. Використання роутингу у Vue.js [Електронний ресурс] – Режим доступу до ресурсуhttps://router.vuejs.org/

9. Принцип роботи технології AJAX [Електронний ресурс] – Режим доступу до ресурсу: https://www.hostinger.com.ua/rukovodstva/chto-takoje-ajax/

10. Принцип роботи технології AJAX [Електронний ресурс] – Режим доступу до ресурсу: https://www.hostinger.com.ua/rukovodstva/chto-takoje-ajax/

11. Vuex та використання в розробці [Електронний ресурс] – Режим доступу до ресурсу https://docs.gitlab.com/ee/development/fe_guide/vuex.html

12. Визначення JavaScript [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/JavaScript

13. Figma – векторний онлайн-сервіс розробки інтерфейсів та прототипування [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Figma

14. Адаптивність та кросбраузерність у веб-розробці [Електронний ресурс] / wikipedia.org – Режим доступу до ресурсу: http://comscienceatschool.blogspot.com/p/22.html

15. API – прикладний програмний інтерфейс [Електронний ресурс] – Режим доступу до ресурсу https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BA%D0%BB%D0%B0%D0%B4%D0%BD%D0%B8%D0%B9_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81

16. Server-side rendering [Електронний ресурс] – Режим доступу до ресурсу: https://vuejs.org/guide/scaling-up/ssr.html

17. Методологія БЕМ (Блок Елемент Модифікатор) [Електронний ресурс] – Режим доступу до ресурсу: https://avivi.pro/ua/blog/metodologiya-bem-v-deystvii/

18. Routing Nuxt.js [Електронний ресурс] – Режим доступу до ресурсу: https://nuxtjs.org/docs/get-cypc]

19. HTML – це стандартизована мова розмітки документів для перегляду веб-сторінок у браузері [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/HTML

20. CSS стилі для розмітки. [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/CSS

21. Header та footer як layouts [Електронний ресурс] – Режим доступу до ресурсу: https://nuxtjs.org/docs/directory-structure/layouts/

22. Nuxt.js як основа Server-side rendering [Електронний ресурс] – Режим доступу до ресурсу: https://nuxtjs.org/

23. Для фільтрації використовується інструмент filter [Електронний ресурс] – Режим доступу до ресурсу: https://v2.vuejs.org/v2/guide/filters.html