ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра інженерії програмного забезпечення

Пояснювальна записка
до магістерської роботи
на ступінь вищої освіти магістр

на тему: «Оптимізація розробки Web-проектів та складних додатків за допомгою фреймворків»

Виконав: студент 6 курсу, групи ПДМ-61
спеціальності:
121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

Стоян М.О.
(прізвище та ініціали)

Керівник Негоденко О.В.
(прізвище та ініціали)

Рецензент_____
(прізвище та ініціали)

Київ – 2022

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМІЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення
Ступінь вищої освіти - «Магістр»
Спеціальність – 121 Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри
інженерії програмного забезпечення
Негоденко О.В.

“____” _____2022 року

ЗАВДАННЯ
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Стоян Микола Олександрович
(прізвище, ім’я, по батькові)

1. Тема магістерської роботи: «Оптимізація розробки Web-проектів та складних додатків за допомогою фреймворків»

Керівник роботи Негоденко Олена Василівна, завідувач кафедри, кандидат технічних наук, доцент
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої оствіти від “11” жовтня 2022 року № 170

2. Строк подання студентом роботи

3. Вихідні дані до роботи:
Науково-технічна література з питань, пов’язаних з побудовою архітектури додатків та технологій орієнтованих на дані.

4.Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
4.1 Аналіз наявних засобів та технологій для розробки веб-додатків або використання фреймворків в складних проектах
4.2 Дослідження інформаційно-орієнтованих технологій.
4.3 Дослідження коли слід використовувати фрймворки, а коли це зовсім недоцільно і не має ніякого сенсу.

5. Перелік графічного матеріалу
1.Титульний слайд
2.Мета, об’єкт дослідження, предмет дослідження
3.Актуальність роботи.
4.Аналіз існуючих рішень
5.Проектування та розрока веб-додатків
5.Дослідження особливостей розробки веб проектів
5.Економічний аналіз.
6. Висновки

6. Дата видачі завдання

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1.	Вибір теми магістерської роботи	05.10.2021р.	
2.	Вивчення літературних джерел, збирання та обробка інформації	10.10.2021р.	
3.	Складання плану роботи	15.10.2021р.	
4.	Узгодження плану роботи та списску використаних джерел з науковим керівником	25.10.2021р.	
5.	Аналіз існуючих рішень	20.09.2021р.	
6.	Дослідження інформаційно-орієнтованих технологій Unity DOTS	10.10.2021р.	
7.	Розробка оптимізованої архітектури додатку	24.10.2021р.	
8.	Вступ, висновки, реферат	05.12.2021р.	
9.	Розробка обов’язкових демонстраційних матеріалів	13.12.2021р.	
10.	Попередній захист роботи		
11.	Здача роботи в деканат		

Студент _____ Стоян М.О.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Негоденко О.В
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина магістерської роботи: 70 сторінок, 24 рисунків, 15 джерел, 8 таблиць.

Об'єкт дослідження – ANGULAR 2 фреймворк.

Предметом дослідження – методи Web-розробки на основі фреймворків для створення складних веб-проектів.

Методи дослідження – методи аналізу, імітаційного моделювання та математичної статистики.

Мета роботи − підвищення швидкості розробки та безпеки функціонування Web-проектів та складних додатків.

У роботі проведено аналіз існуючих фреймворків, які наразі є найпопулярнішими і на них створено не один веб-проект, якими вже користуються тисячі компаній, які створюю веб-додатки.

Головною перевагою фреймворків є те, що вони намагаються боротися з однопоточністю мови програмування JavaScript, що в подальшому значно робить простішим програмування складних проектів. Компонентний підхид зробить розробку набагато читабильной та приємнішою в користуванні для кінцевого користувача.

Дослідження полягає у розробленні теоретико-методичних та практичних рекомендацій які дозволяють розробити архітектурне рішення для розробки веб-проектів за допомогою найкращих фреймворків.

КЛЮЧОВІ СЛОВА: JAVASCRIPT, REACT, ANGULAR 2, VUE.JS, TYPESCRIPT, ASP.NET, HTML, CSS, FRAMEWORK, SPA, SVELTE.

ЗМІСТ

Стор.

ВСТУП

Обгрунтування вибору теми та її актуальність: Сучасні сайти, програми, послуги стають все більш складними, динамічними, багатофункціональними. Вони повинні активно взаємодіяти з користувачем, бути здатними підлаштовуватися під середовище, що змінюється, що обумовлює постійний розвиток методів їх розробки, підтримки. Прагнучи відповідати зростаючим потребам ринку IT-технологій, фахівці розробили найновіший інструмент – фреймворк. Він є каркас, платформа для створення веб-продуктів нового покоління, їх ефективної підтримки. Він призначений для складних, масштабних проектів, що дозволяє реалізувати нестандартні рішення.

Фреймворки використовуються спеціально для створення онлайн-проектів, будь то веб-дизайн сторінки чи більш конкретні веб-сервіси. Хоча існують різні типи фреймворків для веб-додатків, немає перешкод у використанні фреймворку, спочатку задуманого для мови програмування на іншій мові. Причина в тому, що розробнику легше адаптувати одну мову до іншої, ніж змінювати проект з іншими цілями.

Сутність вивчення проблеми: Фреймворки використовуються спеціально для створення онлайн-проектів, будь то веб-дизайн сторінки чи більш конкретні веб-сервіси. Хоча існують різні типи фреймворків для веб-додатків, немає перешкод у використанні фреймворку, спочатку задуманого для мови програмування на іншій мові. Причина в тому, що розробнику легше адаптувати одну мову до іншої, ніж змінювати проект з іншими цілями.

Для досягнення поставленої мети необхідно вирішити такі завдання:

1. Провести порівняльний аналіз ефективності існуючих методів розробки.
2. Розробити проект за допомогою найкращого фреймворку.
3. Оцінити ефективність алгоритму за допомогою аналіза рівня якості варіантів реалізації.

Наукова новизна роботи полягає у розробленні теоретично-методичних та практичних рекомендацій, які допоможуть в виборі найкращих методів розробки веб-проектів, та складних додатків.

Практична значущість результатів: Запропановані заходи удосконалення можуть бути використанні у розробці складних додатків за допомогою фреймворків.

1 АНАЛІЗ ФРЕЙМВОРКІВ ДЛЯ РОЗРОБКИ ВЕБ-ПРОЕКТІВ

1.1 Опис

Досвід користувача є першочерговим пріоритетом для кожного бізнесу, який розробляє веб-сайт. Незалежно від того, наскільки багатогранні операції та функції відбуваються у фоновому режимі, погляди та досвід користувачів мають бути безперешкодними.

Це вимагає використання інтерфейсних фреймворків, які спрощують розробку інтерактивних веб-сайтів, орієнтованих на користувачів.

Front-end фреймворки є головними блоками процесу розробки програмного забезпечення. Але є багато варіантів, які можливо вибрати, якщо потрібно створити візуально привабливі програми, які мають найвищий рейтинг за користувацьким вражненням.

Веб-розробка – це процес перетворення даних у графічний інтерфейс за допомогою CSS, HTML, JavaScript, щоб користувачі могли спостерігати та працювати з цими даними.

Існує безліч платформ та інструментів розробки, таких як Joomla, WordPress і Drupal, які можна використовувати для створення інтерфейсу веб-сайту. Розуміння того, які інструменти накраще підходять для виконання точних задач, свідчить про відмінність між добре розробленим масштабним сайтом і розробкою зламаного сайту.

Розробники інтерфейсу враховують ці моменти, використовуючи доступні методи та інструменти для досягнення цілей. З розробкою мобільних пристроїв, таких як планшети та смартфони, дизайнери вимагають, щоб сайт правильно функціонував у браузерах на всіх пристроях. Це можливо швидко зробити, створивши сприйнятливий веб-дизайн з використанням таблиць стилів у CSS. Цілі продуктивності в основному стосуються часу візуалізації, розгортання CSS, HTML і JavaScript, щоб гарантувати швидке відкриття веб-сайту.

Фронт-енд розробка, також позначається як «CSS фреймворки», є пакетами, що містять попередньо написаний однорідний код у папках і файлах. Вони пропонують основу для розвитку, водночас забезпечуючи гнучкість з остаточним дизайном. Характерно, що фреймворки інтерфейсу охоплюють такі компоненти:

- Сітка дозволяє легко встановити особливості дизайну сайту;
- Добре визначені стилі шрифтів, які відрізняються за призначенням;
- Попередньо вбудовані компоненти сайту, як-от кнопки, панелі навігації та бічні панелі.

Основне використання інтерфейсних фреймворків полягає в тому, що вони створюють інтерактивні інструменти та розробляють адаптивні веб-сайти. Він створює послідовні продукти для збільшення трафіку та покращує зовнішній вигляд веб-сайтів і мобільних додатків.

Крім того, наступні переваги використання front-end фреймворків:

- Фунції та програми є значно більш чуйними. Останній інтерфейсний фреймворк дає можливість веб-розробникам розробляти програми з функціями, що швидко реагують. Вони гнучкі та швидкі в збиранні, що дозволяє додатку швидко реагувати натисканням і діяти відповідно до цього;
- Швидкий розвиток із наслідками, керованими користувачами. За допомогою нинішнього фреймворку можна прискорити розвиток численних елементів сайту. Аналогічно, це скорочує реальний час, необхідний для процедури розробки. Наприклад, частина інтерфейсу потребує одночасної розробки серверної частини програми, і якщо останній налаштований на роботу, то простір інтерфейсу також за кілька днів розквітне;
- Забезпечує програмування в режимі реального часу. Розробники веб-сайтів, які отримують неабияку користь від розробки інтерфейсних веб-сайтів. Від виявлення всіх змін у браузері до того, щоб не боятися втратити позицію веб-програми разом із повторним завантаженням сторінок веб-браузера, веб-розробник може досягти всього;
- Технологія з кількома перевагами. Однією із значущих переваг інтерфейсної

системи веб-розробки є те, що вона підтримується технологією, яку легко масштабувати, вивчати та використовувати. Ці технології пройшли через незліченну кількість детальних повторів, і згодом вони забезпечують зручний досвід у побудові шарів;

- Забезпечений. Серед кількох переваг кодування інтерфейсної веб-розробки є відносно захищеним . Це одна з головних переваг на етапі, коли кібер-сканери чекають лазівки для атаки на ваш цифровий режим. Завдяки цій розробці ви гарантуєте безперебійну роботу. і безпечний сайт, що працює у веб-браузері.

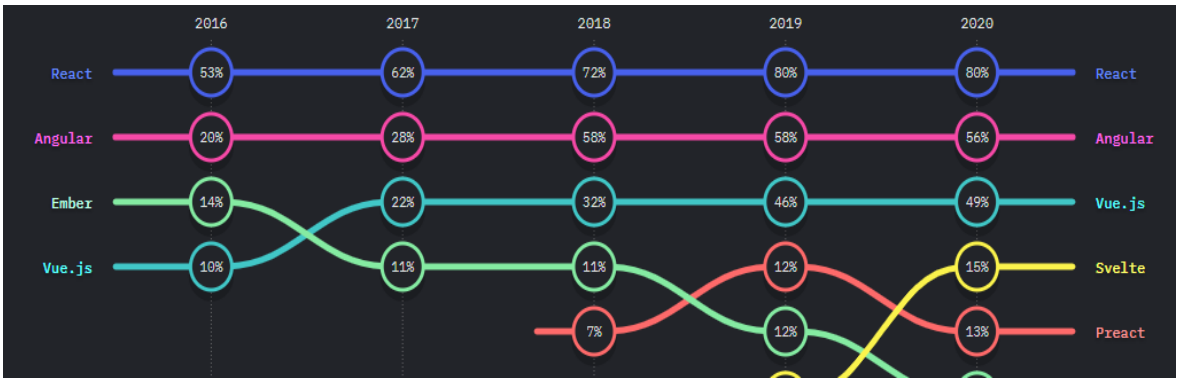


Рисунок 1.1 Рейтинг фреймворків за даними використання розробниками для створення веб-проектів.

1.2 Основні фреймворки для створення веб-додатків

1.2.1 Фреймворк React

До розробки ReactJS, або просто React, як його сьогодні здебільшого називають, Facebook зіткнувся з основним завданням користувацького досвіду – створення динамічного інтерфейсу користувача з максимально можилово високою продуктивністю. Наприклад, інженери хотіли, щоб оновлення стрічки новин відбувалося одночасно з людьми, які використовують чат.

Щоб досягти цього, Facebook довелося оптимізувати сам процес розробки, і Джордан Волке вирішив зробити це за допомогою JavaScript. Він запропонував помістити XHP, синтаксис розмітки Facebook, у систему координат JS. Ідея здавалася неможливою, але в 2011 році його команда випустила бібліотеку ReactJS на основі симбіозу JavaScript і XHP. Тоді Facebook зрозумів, що ReactJS працює швидше, ніж будь-яка інша реалізація такого роду. У 2013 році Facebook випустив React як інструмент JavaScript з відкритим кодом [1].

React – це фреймворк з відкритим вихідним кодом, розроблений і створений командою розробників Facebook. Цей фреймворк є найкращим фреймворком інтерфейсу користувача на сьогодні, який використовується більшістю розробників інтерфейсу згідно з опитуванням Stack Overflow Developer 2021.

Команда розробників Facebook була стурбована забезпеченням відмінної продуктивності шляхом формування корисного інтерфейсу користувача. Основною метою було виправити проблеми з підтримкою коду через постійне додавання функцій у програмі.

Інтерфейсний фреймворк React виділяється своєю віртуальною об’єктною моделлю документа (DOM), яка представляє чудову функціональність. Це ідеальний фреймворк для тих, хто очікує високого трафіку та потребує стабільної платформи для керування ним. Крім того, ця структура є зручною для нових розробників - посібники також допомагають впоратися з будь-якими ускладненнями, які виникають під час процесу навчання.

В цього фреймворку є як і плюси так і мінуси, плюси:

- Економія часу при повторному використанні компонентів;
- Віртуальним DOM покращує як досвід користувачів, так і роботу розробника;
- Бібліотека з відкритим кодом із різноманітними інструментами;
- Стійкий код забезпечується рухом даних в одному напрямку.

Мінуси цього фреймворку:

- Відсутність документації через звичайний розвиток темпів;

- Довга крива навчання;
- Розробникам важко зрозуміти складності JSX.

Також валижливо розумі коли використовувати, а саме React використовується для розробки інтерфейсу користувача, особливо коли потрібно створити односторінкові програми. Це найнадійніший фреймворк інтерфейсу, коли потрібно створити інтерактивний інтерфейс за менший час, оскільки є можливість переробити компоненти.

Найважливіше розуміти коли не потрібно використовувати цей фреймворк, а саме, Якщо немає практичних знань із JavaScript, React не є запропонованою альтернативою. Аналогічно, для недосвідчених розробників крива навчання JSX є серйозною.

Щоб працювати з React потрібно і дотримуватись структури, Кореневий каталог абсолютно однаковий для кожного проекту, і про нього не ведеться багато дискусій, оскільки йдеться лише про конфігурацію. Тож важливо розібрати папку в якій все зберігається, а саме

Рисунок 1.2. Правильна структура веб-проекту з використання фреймворка React.

TSX — це розширення файлів TypeScript, що містять код JSX.

Файл index.tsx – це точка входу в проект. В цьому файлі ініціалізують бібліотеки, які використовують, в основному це:

- Styled-components, це бібліотека CSS-In-JS, яка дозволяє писати звичайний CSS і приєднувати його до компонентів JavsScript. За допомогою styled-components з'являється можливість використовувати CSS, я яким всі знайомі, замість того, щоб вивчати нову структуру стилів. Styled-components дає змогу стилізувати роботу, не турбуючись про те, чи суперечать назви класів або чи підходять вони для певного випадку використання. Це також допомогає при налагодженні, тому що як тільки з'ясується, який компонент має проблему, не є проблемою знайти йього стилі;
- Redux. Це передбачуваний контейнер стану, призначений для того, щоб допомогти писати програми JavaScript, які працюють послідовно в клієнтських, серверних і рідних середовищах, та легко тестуються. Хоча він здебільшого використовується як інструмент керування станом із React, найлоговніше, що є можливість використовувати йього з будь-яким фреймворком або бібліотекою JavaScript. Він не велекий і займає вього 2КБ, тому не доведеться турбуватися про те, що він збільшить розмір активу розробки.
- Reacr Router. Це повнофункціональна бібліотека маршрутизації на стороні клієнта та сервера для React, бібліотеки JavaScript для створення інтерфейсів користувача. React Router працює скрізь, де працює React; в Інтернеті, на сервері з node.js і на React Native.

Файл app.tsx, містить компонент React Router, тут реалізуються функції, які використовуються в усій програмі, як-от модальна система, контейнер сповіщень, сліжбові тощо.

Файл router.jsx, складається з Switch і основних Route компонентів проекту. Якщо потрібно підмаршрути, вони обробляються контейнерами, які їх використовують.

Папка активів, Без великого сюрпризу. Очевидно, що ця папка містить зображення, значки, шрифти тощо. Але оскільки не потрібно, щоб він став величезним кошиком для сміття, куди команда просто викидає речі випадковим чином у міру зростання програми, активи сортуються за контекстом.

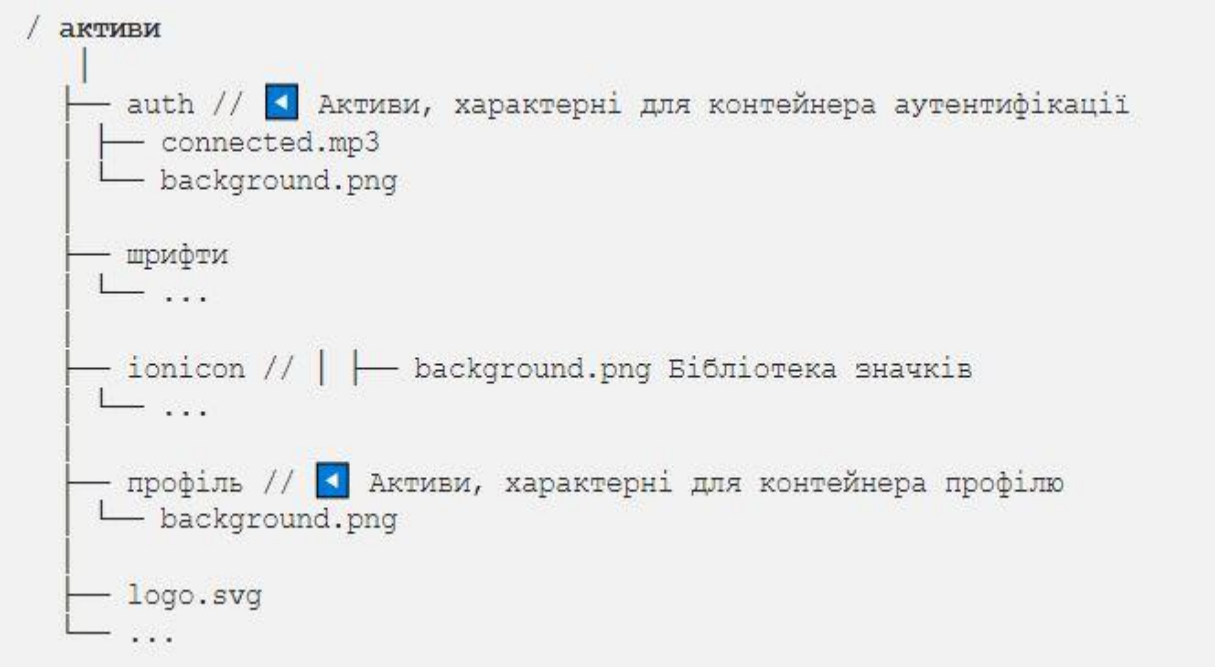


Рисунок 1.3. Правильна структура папки активи.

Правило таке, якщо актив використовується лише в певному контейнері, він належить до папки з назвою контейнера. Наприклад, якщо програма використовує логотип, його, ймовірно, можна використовувати в будь-якому місці цього проекту. З цієї причини цей файл належить до кореня ./assets папки. Але фонове зображення, яке використовується лише в auth контейнері, належить до auth підпапки. Цей метод також полегшує пошук активів, які більше не використовуються. Спочатку це не повинно бути проблемою, але коли проект значно збільщуєьбся: розробники постійно забувають очистити цю папку.

Та найголовніше, це папки компонентів та контейнерів. Важливо розуміти, що багато різних смаків по групуванні компонентів, але React документація пропонує два варіанти:

- Групування файлів за функціями або маршрутами;

- Групування файлів за типом, тобто, CSS, компоненти, тести;

Вважається накращим рішенням, це перший варіант, тобто групування файлів за функціями або маршрутами.

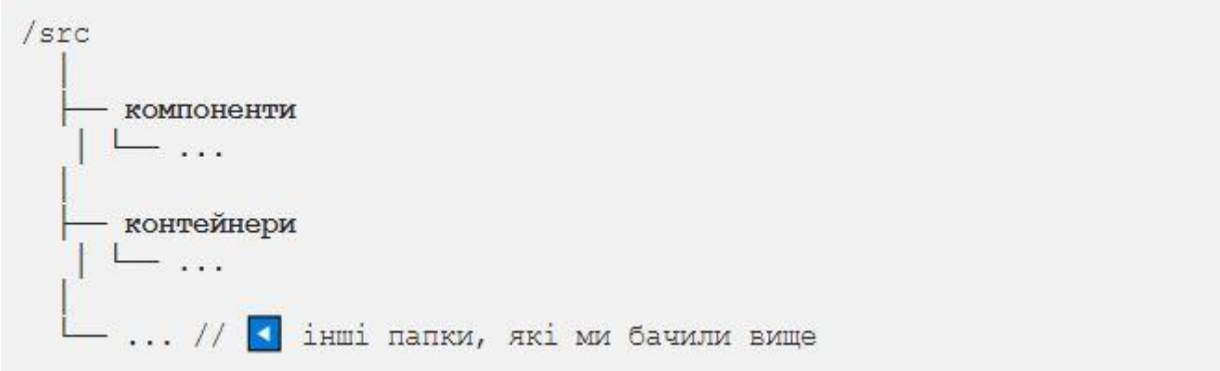


Рисунок 1.4. Правильна структура папки компонентів та контейнерів.

Щоб компонент React зайняв своє місце в components папці, він повинен підкорятися двом правилам, по-перше, це має бути презентаційний компонент , що означає, що він не пов’язаний із станом програми і, звичайно, не отримує та не публікує дані. Він може взаємодіяти з батьківським контейнером (наприклад, запускаючи функцію, яка була передана в Props). По-друге, Його необхідно використовувати для кількох компонентів або контейнерів.

Важливо розуміти, що Компоненти забезпечують гарний вигляд, а контейнери допомагають працювати. Ще одна перевага, яку Facebook представив у React, — це можливість у будь-який час повторно використовувати компоненти коду іншого рівня, ще один значущий ефект економії часу. Розробникам легко керувати оновленнями, оскільки всі компоненти React ізольовані, і зміни в одному не впливають на інші. Це дозволяє повторно використовувати компоненти, які самі по собі не змінюють, щоб зробити програмування більш точним, ергономічним і зручним для розробників.

1.2.2 Фреймворк Angular

Angular є одним із найвідоміших рішень для розробки SPA (односторінкових додатків), окрім React і Vue.js. Він існує вже майже 10 років, і з тих пір він зазнав незліченних змін. Перша версія фреймворка – AngularJS – стартувала ще в 2009 році і заклала основу сучасної розробки фронтенд-додатків.

Перший стабільний випуск AngularJS (версія 0.9.0, також відомий як *dragon-breath*) був випущений на GitHub у жовтні 2010 року за ліцензією MIT; Коли AngularJS 1.0.0 вийшов у червні 2012 року, фреймворк вже здобув величезну популярність у спільнотах веб-розробників у всьому світі [2].

Складно описати причини такої популярності, просте основні ключові моменти:

- Ін'єкція залежності. Важливо розуміти, що AngularJS був першим фреймворком на стороні клієнта, який реалізував його: це, безсумнівно, було величезною перевагою перед конкурентами, включаючи бібліотеки, що маніпулюють DOM, такі як JQuery. За допомогою AngularJS розробники могли писати слабо зв’язані та легко перевірені компоненти, залишаючи на фреймворку завдання їх створення, вирішення їхніх залежностей і передачу іншим компонентам за запитом.
- Директиви**.** Їх можна описати як маркери на конкретних елементах DOM, таких як елементи, атрибути, стилі тощо: потужна функція, яка може використовуватися для визначення користувацьких і багаторазових елементів, подібних до HTML, і атрибутів, які визначають прив’язки даних та/або іншу специфічну поведінку. компонентів презентації.
- Двостороннє прив’язування даних. Автоматична синхронізація даних між компонентами моделі та представлення: коли дані в моделі змінюються, подання відображає зміну; коли дані у представленні змінюються, модель також оновлюється. Це відбувається миттєво й автоматично, що гарантує, що модель і вигляд постійно оновлюються.
- Односторінковий підхід. AngularJS був першим фреймворком, який повністю

усунув необхідність перезавантаження сторінок. Це дало великі переваги як на стороні сервера (менша і менша кількість запитів до мережі), так і на рівні клієнта (плавніші переходи, більш чуйний досвід) і проклало шлях для шаблону односторінкового додатка**,** який також буде прийнятий у React , Vue. JS та інші фреймворки, що зайняли друге місце пізніше.

- Дружній до кешу. Вся магія AngularJS мала відбуватися на стороні клієнта, без будь-яких зусиль на стороні сервера для створення частин UI/UX; саме з цієї причини всі веб-сайти AngularJS можна кешувати в будь-якому місці та/або робити доступними через CDN.

Новий випуск AngularJS, випущений 14 вересня 2016 року і відомий як Angular 2 , був повністю переписаним попереднього, повністю заснованим на новій версії ECMAScript 6 (офіційно ECMAScript 2015) специфікаціях. Подібно до перепису ASP.NET Core, революція принесла таку кількість руйнівних змін на архітектурному рівні, обробці конвеєра HTTP, життєвому циклі програми, управлінні станом, що перенесення старого коду на новий було майже неможливим: незважаючи на збереження його колишня назва, нова версія Angular була абсолютно новою структурою, яка майже не мала спільного з попередньою [2].

Версій Angular було багато, проте важливо знати про найголовніше та про найновіше, щоб вдало користуватися фреймворком і розуміти для чого він створений.

І останнє, але не менш важливе, Angular 8 , який був випущений 29 травня 2019 року і наразі є останньою версією. Новий випуск здебільшого стосується Ivy , довгоочікуваного нового компілятора/часу виконання Angular: хоча версія 8 є постійним проектом з Angular 5, версія 8 є першою, яка офіційно пропонує перемикач часу виконання, щоб фактично підключитися до використання Ivy — що можливо, стане середовищем виконання за замовчуванням, починаючи з Angular 9 [2].

Покращення останьої версії фреймворку, що дає ще більше можливостей для розробки:

- Підтримка Babel. Angular 8 підтримує Bazel , безкоштовний програмний інструмент, розроблений і використовуваний компанією Google для автоматизації створення та тестування програмного забезпечення: він може бути дуже корисним для розробників, які прагнуть автоматизувати свій конвеєр доставки, оскільки він дозволяє інкрементно створювати та тестувати і навіть можливість налаштувати віддалені збірки (і кеш) на фермі збірок.
- Маршрутизація. маршрутизатор Angular тепер приймає новий синтаксис для оголошення маршрутів із відкладеним завантаженням , використовуючи синтаксис import() із TypeScript 2.4+ замість того, щоб покладатися на рядковий літерал: старий синтаксис збережено для зворотної сумісності, але буде можливо, скоро впав.
- Service Workers. була представлена нова стратегія реєстрації, яка дозволяє розробникам вибирати, коли реєструвати своїх працівників, замість того, щоб робити це автоматично в кінці життєвого циклу запуску програми; також можна обійти сервіс-воркер для конкретного HTTP-запиту за допомогою нового ngsw-bypass заголовка.
- API робочого простору. Новий і більш зручний спосіб читати та змінювати конфігурацію робочої області Angular замість того, щоб вручну змінювати angular.json файл.

Основні переваги використання Angular в проекті:

- Полегшення процедури кодування завдяки службам рефакторингу та розширеній навігації;
- Компонентний шаблон санкцій Angular утворює інтерфейс користувача з окремих компонентів;
- Велика екосистема;
- Angular material реорганізує виробництво інтерфейсу Material Design;
- Висока ефективність.

Але як є і переваги так є і мінуси:

- Angular ускладнює;
- Переміщення застарілих схем з AngularJS на Angular;
- Документація CLI не чітко визначена;
- Навчальні зусилля.

Потрібно розуміти коли потрібно використовувати цей фреймворк, а коли ні. Angular розширює виконання програм на основі браузера, енергійно модернізуючи вміст за менший час, оскільки використовує двостороннє прив'язування даних. Для корпоративного та активного веб-додатка використання Angular підходить. Проте є випадки в яких цей фреймворк не можна використовувати. Angular — це комплексне рішення як інтерфейсний фреймворк. Якщо потрібно розробляти програми з недостатнім обсягом, не є можливим використовувати ресурси, які надає Angular.

1.2.3 Фреймворк Vue.js

Сьогодні одним з найпростіших фреймворків є Vue.js. Варто усунути тонкощі, з якими стикаються розробники програмного забезпечення Angular.

Він має невеликий розмір і має дві основні переваги – візуальний DOM і компонентний. Він також використовує двостороннє прив'язування даних. Цей інтерфейсний фреймворк є універсальним і допомагає розробникам виконувати різноманітні завдання під час створення веб-додатків.

Від створення веб-додатків і мобільних додатків до прогресивних веб-додатків (PWA) він може легко керувати як динамічними, так і простими розробками. Незважаючи на те, що він створений для вирішення складнощів і підвищення продуктивності додатків, він зазвичай не популярний серед гігантів галузі. Тим не менш, Xiaomi, Alibaba, Reuters і 9Gag є користувачами цього фреймворку. Vue.js продовжує поширюватися з точки зору прийняття, незважаючи на те, що з Кремнієвої долини менше користувачів.

Історія Vue.js починається в 2013 році, коли Еван Ю працював у Google, створюючи безліч прототипів прямо в браузері. З цією метою Еван використав

зручні практики з інших фреймворків, з якими він працював, і офіційно випустив Vue.js у 2014 році.

Vue.js — це прогресивний фреймворк для JavaScript, який використовується для створення веб-інтерфейсів і односторінкових програм. Vue.js також використовується не тільки для веб-інтерфейсів для розробки настільних і мобільних додатків за допомогою Electron framework. Розширення HTML і база JS швидко зробили Vue улюбленим інструментом інтерфейсу, про що свідчить прийняття такими гігантами, як Adobe, Behance, Alibaba, Gitlab і Xiaomi.

Назва фреймворка Vue англійською збігається фонетично з *view* і відповідає традиційній архітектурі Model-View-Controller (MVC). Простіше кажучи, view — це інтерфейс програми/веб-сайту, а основна бібліотека Vue.js за замовчуванням фокусує рівень перегляду. Але MVC не означає, що Vue.js не можна використовувати з іншим архітектурним підходом, як-от Component Based Architecture (CBA), що використовується в React [3].

У своєму нинішньому вигляді Vue.js, розроблений Еваном Ю на повний робочий день, отримує користь від внесків членів спільноти та фінансується в основному Patreon. Без фінансової підтримки таких підприємств, як Facebook (React) і Google (Angular), Vue все ще отримав широке поширення на Github .

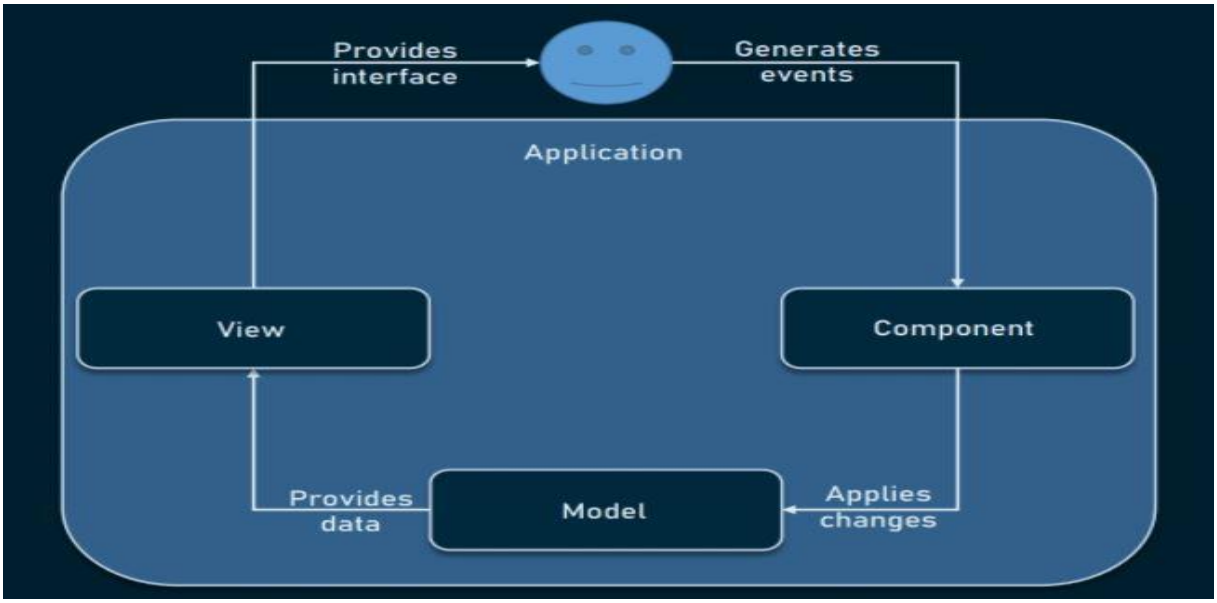


Рисунок 1.5. Архітектура контролера перегляду моделі.

Як і будь-яка популярна технологія, програмування Vue.js викликає суперечки у всьому співтоваристві розробників, які створюють веб-сторінки. І є причини, чому Vue став другим найпопулярнішим фреймворком у 2019 році.

По-перше, головний плюс фреймворку, це маленький розмір. Ця точка буде невелика, як і сам Vue: завантажений zip з фреймворком важить 18 КБ. Будучи легкою вагою, фреймворк не тільки швидко завантажує та встановлює бібліотеку, але й позитивно впливає на SEO та UX.

По-друге, віртуальний рендеринг і продуктивність DOM. Document Object Model (DOM) є те , що кожен розробник ймовірно знає, тому що постійной працює з візуалізацією веб - сторінок. DOM – це представлення сторінок HTML з його стилями, елементами та вмістом сторінки як об’єктами. Об’єкти, що зберігаються у вигляді деревоподібної структури, генеруються браузером під час завантаження сторінки.

Коли користувач взаємодіє зі сторінкою, об’єкти змінюють свій стан, тому браузер повинен оновлювати інформацію та відображати її на екрані. Але оновлення всього DOM є громіздким. Заради швидкості Vue.js використовує віртуальну DOM: подумайте про це як про копію оригінальної DOM, яка визначає, які елементи оновлювати, без повторного відтворення всього DOM. Такий підхід робить рендеринг сторінки досить швидким і покращує продуктивність програми.

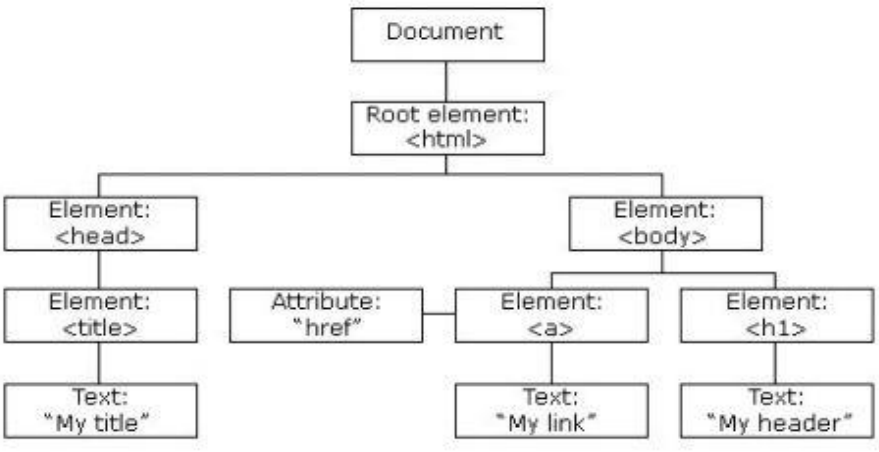


Рисунок 1.6. Дерево об’єктів HTML DOM.

Продуктивність є одним із ключових факторів, які можуть зумовити вибір фреймворка. Фактичні контрольні показники наведено на сторінці порівняння Vue. Наприклад, під час тестування компонентів DOM, пов'язаних із оновленими даними, Vue.js здається більш продуктивним, ніж Angular та React. Звичайно, це далеко не лідируючі позиції, де вкладається Vanilla.js , а тест включає старі версії фреймворків, тому саме це потрібно врахувати. Але загальна картина виглядає оптимістично.

Ще однією перевагою маніпуляцій DOM є двостороннє прив'язування даних, успадковане Vue від Angular. Двостороннє прив'язування даних – це зв'язок між оновленнями даних моделі та переглядом (UI). Зв'язані компоненти містять дані, які час від часу можна оновлювати. За допомогою двостороннього зв'язування даних легше оновлювати пов'язані компоненти та відстежувати оновлення даних.

У Vue зв'язані дані оновлюються реактивно, як і об'єкти DOM, що чудово підходить для будь-якої програми, яка потребує оновлення в режимі реального часу. З точки зору розробників, реактивність Vue зробить оновлення даних більш зрозумілим і легшим у виконанні. Для того, щоб реактивність працювала, застосовуються деякі правила.

Потрібно пам'ятати про однофайлеві компоненти та читабельність. Кожна частина майбутньої програми/веб-сторінки у Vue є компонентом. Компоненти представляють інкапсульовані елементи інтерфейсу. У Vue.js компоненти можна писати на HTML, CSS та JavaScript, не розділяючи їх на окремі файли. Поділ коду програми насправді є архітектурним підходом, який називається Component Based Architecture (CBA), і він також використовується в Angular і React. Такий архітектурний підхід має масу переваг:

- Можливість повторного використання компонентів в веб-проекті. Інкапсульовані компоненти, це в основному фрагменти коду, які можна повторно використовувати як шаблони для подібних системних елементів;
- Читабільність коду. Оскільки всі компоненти зберігаються в окремих файлах (а кожен компонент є лише одним файлом), код легше читати та розуміти, що

полегшує його обслуговування та виправлення;

- Добре підходить для модульного тестування. Юніт-тестування — це дія контролю якості, спрямована на перевірку того, як найменші частини програми працюють самостійно. Наявність компонентів значно спрощує це завдання.

Важливим аспектом будь-якої нової технології є можливість інтеграції з існуючими програмами. З Vue.js це так само просто, як пиріг, тому що він покладається лише на JavaScript і не вимагає жодних інших інструментів для роботи. Vue також дозволяє писати шаблони за бажанням: використовуючи HTML, JS або JSX (розширення синтаксису JavaScript). Серед компонентів і легкої природи Vue можна використовувати майже в будь-якому проекті. І найголовніша перевага цього фреймворку, що перехід з React або Angular не буде великою проблемою, оскільки внутрішня організація Vue в основному є сумішшю цих двох.

Найголовніші переваги Vue для розробки новачкам, які прагнуть розібратися без великого зусилля:

- Легко навчитися. Інструмент може досягти масового поширення лише тоді, коли його легко зрозуміти, що може бути у випадку вивчення Vue.js. Щоб почати кодування, Vue не вимагає глибоких знань про бібліотеки, JSX і TypeScript, як це часто буває з іншими інтерфейсними технологіями. Все, що новому розробнику потрібно для початку, це базові знання HTML, CSS і JavaScript, звичайно. Найпопулярніші редактори коду, такі як Sublime text , Visual Studio та Atom, підтримують Vue, що полегшує спробу розібратися та працювати в подальшому;
- Коротка документація. Всі розробники повинні віддати належне Vue.js документації. Незалежно від того, чи новачок, який збирається вивчати фреймворк, чи технічно підкований хлопець, який шукає довідку про проблему: документація Vue допоможе кожному розробнику розібратися без зайвих зусиль. Він добре структурований і охоплює всі можливі теми, точно описуючи все, від встановлення до більш глибоких речей, таких як

реактивність і масштабування програми. Що ще важливіше, є також розділ, у якому порівнюється Vue з іншими фреймворками JS та надається спільне для них (наприклад, віртуальний DOM у Vue та React, синтаксис шаблону у Vue та Angular);

- Та найголовніше, це підтримка громади. Vue.js існує завдяки своїй спільноті. Через це учасники спільноти досить активні як у чаті Discord, так і на форумі. Як доказ діяльності спільноти, просто потрібно подивитися на кількість тегів Vue.js на Stack Overflow, яка зараз налічує понад 41 тисячу.

Проте, як і в кожному фреймворку чи в кожній мові програмування є як і світла сторона медалі так і темна, отже, потрібно знати і мінеси Vue.js, а вони такі:

- Мовний бар'єр. Впровадження Vue на таких підприємствах, як Xiaomi і Alibaba, допомогло популяризувати фреймворк і створило попит на ринку праці. Оскільки Vue.js стає все більш популярним у Китаї, значна частина його вмісту та обговорень, як не дивно, китайська. Великий китайський брандмауер робить речі в цій країні дещо іншими, тому що багато популярних ресурсів зараз там недоступні. Це ускладнює вивчення та використання React або Angular. Vue є більш кращим варіантом. Отже, під час пошуку вмісту Vue, розробник, очевидно, зіткнетеся з обговореннями на форумі, описами плагінів та інструкціями китайською мовою. Це може бути проблемою для інженерів, які володіють лише англійською мовою.
- Складність реактивності. двостороннє прив'язування даних, реалізоване у Vue для керування оновленнями DOM. Хоча це зручний інструмент для синхронізації компонентів, є одна проблема, що стосується того, як працює система реактивності, як її називають. Зрозумілою мовою, програма Vue.js складається з компонентів, з якими може взаємодіяти користувач. Кожен компонент має свого спостерігача, який повторно відтворює дані щоразу, коли користувач запускає компонент. Система реактивності повторно відтворює лише ті фрагменти даних, які були запущені. Справа в тому, що він не такий розумний і часто робить помилки під час зчитування даних, тому

вимагає зведення даних. Однак це відома проблема, і вона вирішена в документації Vue, де є вказівки щодо того, як правильно налаштувати реактивність.

- Відсутність підтримки маштабних проектів. Vue.js ще молодий фреймворк. Розмір спільноти та команди розробників все ще непорівнянний з більш зрілим Angular. Також не користується фінансовою підтримкою великих підприємств. Для того, щоб технологія була прийнята у великомасштабних проектах, технологія повинна бути стабільною та сильно підтримуваною, щоб проблеми можна було швидко вирішувати. Хоча у Vue не так багато проблем з цим, і навіть є попит від таких підприємств, як IBM і Adobe, він все ще використовується у відносно невеликих проектах.
- Ризик надмірної гнучкості. Якщо згадувати про гнучкість, яка є спірною якістю для великого проекту. Надання команді розробників занадто багато варіантів може призвести до філософськи різних підходів до програмування, які боротимуться всередині однієї команди. І, як результат, це остаточний обнулювач замість працюючого програмного забезпечення.
- Обмежені ресурси. Хоча екосистема досить широка, і є всі необхідні інструменти для початку розробки за допомогою Vue, вона все ще не така велика, як React або Angular. Щоб бути більш точним, просто потрібно порівняти кількість плагінів, доступних для React і Vue.js: різниця в сотнях. Існуючі плагіни, які можна використовувати з іншими фреймворками, часто також не підтримуються, але це може бути лише питанням часу.
- Брак досвідчених розробників. Vue.js – відносно молода технологія, яка тільки почала набирати популярність. Але, здається, нам доведеться почекати кілька років, поки він не буде масово прийнятий на ринок праці, наповнений досвідченими розробниками Vue.js.

Враховуючи всі концептуальні та технічні аспекти Vue.js, розробники можуть поставити запитання, де він може найкраще підходити. Крім прямого

призначення, створення односторінкових програм і роботи на веб-сторінках, він також підходить для виконання ряду завдань. Наприклад, робота з протипами, перш за все, Vue.js був розроблений для створення прототипів. При правильному зв'язуванні даних також чудово працювати з великою кількістю анімацій, інтерактивних елементів та графіки. Вивчіть інтерфейс, встановіть Vue CLI, і ви готові працювати з інтерактивними прототипами.

Також, Vue.js зосереджується на роботі з інтерфейсом користувача, оскільки для роботи з ним потрібні лише HTML, CSS та JS, без зайвих матеріалів, що стосуються Vue. Наприклад, IBM використовувала Vue як інтерфейсний фреймворк для своєї гібридної хмари через його невелику вагу, м'яку криву навчання та залежність від HTML, CSS та JS.

Якщо у розробників є існуюча програма, і вони хочуть додати до неї інтерактивність, Vue може допомогти з цим. Оскільки він заснований на JavaScript, його можна легко інтегрувати в будь-який проект за допомогою JS. Крім того, він сумісний з багатьма внутрішніми технологіями та фреймворками, такими як Laravel, Express, Rails та Django.

Навіть враховуючи недоліки Vue.js, його все одно можна використовувати у великих проектах. Більшість питань вирішується в документації, тому це питання пошуку.

1.2.4 Фреймворк Svelte

Нарешті у розробників з'явився претендент, готовий приєднатися до Великої трійки – React, Vue та Angular. Svelte — ще один популярний фреймворк веб-розробки для розробників завдяки чіткому для читання шаблонам HTMLX та естетично привабливому вигляду. Але чи це відповідна структура для розробників проекту? Потрібно зрозуміти переваги та недоліки.

А саме, популярним цей фреймворк роблять шаблони Svelte, це є наднабором HTML Якщо розробники знають HTML і CSS, крива навчання буде здаватися відносно простою. Філософія дизайну Svelte подібна до Python. Існує чудовий підручник з інтерфейсом у режимі реального часу, який допомагає розробникам

швидко вивчити фреймворк. Також, великий плюс, розширення мови JS, CSS і HTML призначені для покращення роботи розробника. Фреймворк став популярним завдяки своїм невеликим пакетам і продуктивності. Для підвищення продуктивності режим відтворення на стороні сервера Svelte об'єднує компоненти та компілює їх у форматі рядка.

Також, невід'ємною частиною популярності є API, надійні в користуванні та красиві. Починаючи від функцій до семантики, синтаксису та розміру встановлення, все естетично.

Віртуальний DOM часто має певні недоліки. Найкращою особливістю Svelte є те, що він оптимізує код під час компіляції, таким чином поширюючи зміни з мінімальними накладними витратами під час виконання.

Svelte може бути не таким революційним, як AngularJS чи React у наші дні на JQuery, оскільки, зрештою, він майже виконує ту ж роботу, що і більшість інших фреймворків MVVM, таких як React, Vue чи Angular. Якщо чесно, Svelte спирається на уроки, отримані від своїх попередників. Але справа не в цьому. Що відрізняє його, так це підхід, який він використовує, і переваги, які він надає, які можуть бути незначними або помітними залежно від конкретного випадку використання.

Щоб зрозуміти, як його використовувати в проекті та чи потрібен він, розробники повинні розуміти переваги та недоліки цього фреймворку, а саме про переваги:

- Компілятор проти віртуального DOM. Бути компілятором і позбутися VirtualDOM є найважливішою перевагою Svelte, яка полегшує багато інших переваг. Концепція стає настільки популярною, що Angular і Ember почали рухатися до компіляції в своїх останніх версіях.
- Легкий і продуктивний. Svelte створює дуже оптимізований ванільний JS з мінімальними накладними витратами під час виконання. Це означає невеликі розміри пакетів, низький обсяг пам'яті, а також швидке завантаження та роботу програми.

- Менше шаблонного. З Svelte немає потреби додавати клейовий код, як-от гачки, або складне керування станом тощо. Шаблон, необхідний для компонентів, дуже мінімальний і майже близький до звичайного HTML/JS. Svelte також підтримує додаткові двосторонні прив’язки, що полегшують створення форм.

```
<style>
  h1 {
    color: blue;
  }
</style>
<script>
  let name = "world";
</script>

<input bind:value="{name}" />

<h1>Hello {name}!</h1>
```

Рисунок 1.7. Простий компонент у Svelte з двостороннюю прив’язкою введення.

- Справді реактивний. Svelte реагує за замовчуванням. DOM автоматично оновлюється при зміні стану будь-якої змінної верхнього рівня компонента. Для цього навіть не потрібно додавати спеціальний код. Таким чином працюють лише прямі призначення верхнього рівня, а мутації посилання, наприклад array.push, не працюватимуть. Це означає, що, мутації будуть більш чіткими і легшими для розуміння. Svelte також підтримує похідні декларації та оператори, які повторно обчислюються при зміні стану за допомогою спеціальної мітки ($:).
- Низька крива навчання. На відміну від React або Angular, крива навчання Svelte досить низька. Немає спеціального синтаксису, такого як JSX, для вивчення або складних API, як-от Angular, щоб запам’ятати. Все написано з використанням стандартно-сумісних JS/TS, CSS і HTML з додатковим синтаксисом для директив і логіки шаблонів. API компонента простий і

зрозумілий. Документація також досить гарна, і її легко слідувати.

- Компонентний підхід. Svelte дотримується принципу «перший компонент», що робить його ідеальним для створення нових веб-програм або для додавання веб-компонентів до існуючих програм. Стилі за замовчуванням розраховані на компоненти, що робить Svelte ідеальним для веб-компонентів.
- Вбудована анімація та ефекти. Svelte надає вбудовані анімації та ефекти, які полегшують створення зручних інтерфейсів користувача та інтерактивних візуалізацій. Ну, фреймворк спочатку був створений для створення інтерактивної графіки для The Guardian. Цей підхід забезпечує набагато кращий досвід розробника, ніж щось на кшталт React, і його набагато легше використовувати. Ось простий приклад використання ефекту переходу:

```
<script>
  import { fade } from "svelte/transition";
  let visible = true;
</script>

<label>
  <input type="checkbox" bind:checked="{visible}" />
  visible
</label>

{#if visible}
<p transition:fade>Fades in and out</p>
{/if}
```

Рисунок 1.8. Приклад використання ефекту переходу.

- Вбудований магазин Reactive. Svelte надає як змінні, так і незмінні реактивні сховища з коробки, що спрощує більш складне керування станом у вашій програмі. Магазини підтримують ручні та автоматичні підписки та двосторонні прив’язки, що робить їх дуже гнучкими. Реалізація також дає можливість перейти до іншого рішення для управління станом, наприклад, RxJS.

- Кілька вихідних цілей. Будучи компілятором, легко змінювати вихідні цілі, не змінюючи код вашого компонента. Наприклад, Svelte підтримує відтворення на стороні сервера з коробки, надаючи для нього режим компілятора (domvs ssr). Існує навіть інтеграція NativeScript для Svelte, яка використовує цю гнучкість для створення цілей за межами domta ssr. Існує також фреймворк Sapper, який незабаром стане SvelteKit , від команди Svelte, який схожий на Next.js, але оптимізований для роботи з філософією Svelte. Sapper підтримує SSR, прогресивні веб-програми, розділення коду тощо.

Усі ті переваги, які були описані, не означають, що недоліків немає, кожен фреймворк має компроміси. Основними недоліками Svelte є:

- Молодий каркас. Svelte дуже молодий, а це означає, що він не такий перевірений у боях, як React чи Angular, і ви можете час від часу зіткнутися з деякими стінами. Це означає, що він, ймовірно, не підходить для дуже складних або критично важливих додатків, які очікують масштабування. Це може не бути довгостроковою проблемою, оскільки популярність фреймворка зростає, а впровадження Sapper допогло вирішити проблеми масштабування. Хоча, на думку багатьох розробників, плутанини навколо Sapper проти SvelteKit можна було уникнути.
- Менша спільнота та екосистема. Будучи молодим фреймворком, він має меншу спільноту та базу користувачів, а також меншу екосистему. Тому розробники можуть не знайти стільки інструментів або бібліотек, як у React, або стільки допомоги в Stack Overflow, коли розробники застрягли на якійсь складній проблемі.
- Примхи. Svelte відповідає веб-стандартам, він не вводить нічого нового, як JSX. Але це змінює деяку стандартну семантику, щоб працювати по-іншому, і це може збити з пантелику нових користувачів. Наприклад, він використовує export ключове слово по-різному, і є такі особливості, як необхідність використовувати on:click замість onClick і так далі. Але їх

майже не уникнути в будь-яких рамках. Він також використовує мітку JS ($:) для того, щоб похідні оператори/декларації працювали, це може виглядати чужорідним, оскільки деякі розробники JS, ймовірно, навіть не знають, що мітки існують в JS, оскільки ми їх рідко використовуємо.

Висновок до розділу 1

В даному розділі ми розібралися в сутності роботи з фреймворками, дізналися для чого вони створенні, і для яких потреб слід використовувати той чи інший інструмент. З’являється розуміння, завдяки якому фреймворку необхідно менше ресурсів для виконання разніх завдань-проектів, на яких дійсно буде легше працювати з командою, який фреймворк розвивається і немає пережевання, що потім необхідно переносити проект на іншу систему.

В кожному фреймворку є як плюси так і мінуси, один зроблений для швидких рішень, інший змінює екосистему, в кожному з наведених вище фреймів є багато плюсів так мінусів, багато людей користується кожним із них і немає конкретної відповіді, який фреймворк є найкрутішим, проте є відповіді для яких цілей на сьогоднішній день краще використовувати той чи інший.

Висновок, що наразі із описаних фреймів, краще використовувати React і Angular для великих проектів, та React для створення проектів компонентним підходом та створення веб-додатків та додатків на мобільні пристрої. Vue.js для невеликих проектів або для тих, які потребують швидкого рішення.

2 ПРОЕКТУВАННЯ ТА РОЗРОБКА WEB-ДОДАТКІВ

На етапі аналізу предметної області були визначені основні завдання, які вирішують веб-додатки та вирішують технічні вимоги. Моделювання предметної області полегшує вибір інструментів і технологій.

2.1 Архітектура веб-додатків

Нижче наведений опис етапів створення кінцевої програми буде базуватися на ітеративному підході, який базується на поступовому ускладнення та деталізація початкової структури додатка.

Основою розробленої системи є архітектура «клієнт-сервер», в який розподіл навантаження на мережу між постачальниками послуг (сервісами), іменовані сервери, а клієнти послуг називаються клієнтами

Клієнт — це браузер користувача, де користувач взаємодіє з даними за допомогою DOM і JavaScript. Використання JavaScript не є обов'язковим саме по собі, але воно працює більшість веб-додатків без JS неможливі. Як Середовище взаємодії клієнта з сервером використовує Інтернет (рис 2.1).

Ви також можете використовувати архітектуру для вирішення цієї проблеми «Трьохрівневої архітектури», але в цьому випадку буде схожий підхід випущено за відсутності великого фрагмента додатків бізнес-логіки.

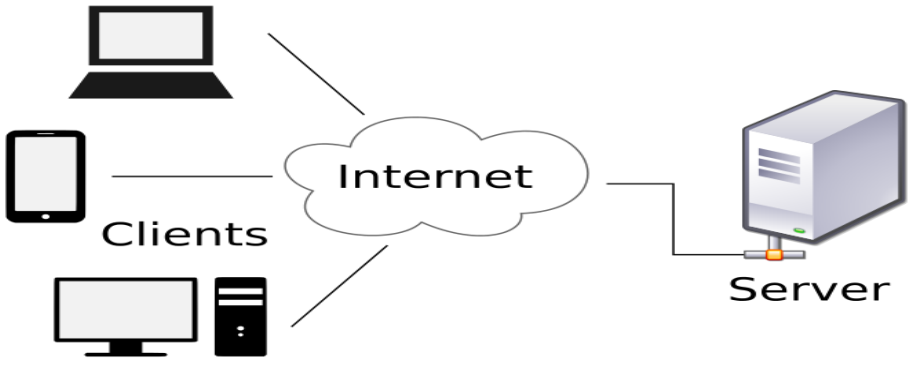


Рисунок 2.1. Загальна архітектура додатку.

Основними перевагами архітектури клієнт-сервер є:

Здатність, у більшості випадків, розрізняє функції комп’ютерної системи між кількома незалежними комп’ютерами в мережі. Це спрощує обслуговування комп'ютерної системи. Зокрема, заміна, ремонт, оновлення або переміщення сервера не впливає на клієнтів.

Усі дані зберігаються на сервері, який зазвичай набагато краще захищений, ніж великі клієнти. Сервер легше забезпечити повний контроль, щоб дозволити доступ до даних тільки від клієнтів з відповідні права доступу.

Дозволяє комбінувати різних клієнтів. Клієнти з різними апаратними платформами, операційними системами тощо часто можуть використовувати ресурси одного сервера.

Основні недоліки архітектури клієнт-сервер:

У разі використання централізованої системи вихід з ладу головного сервера може зробити всі веб-додатки непрацездатними.

Адміністрування цієї системи вимагає кваліфікації професіоналізм та висока вартість обслуговування.

У процесі вибору апаратної платформи будуть реалізовані пропозиції рішення, які мінімізують ймовірність виходу зі структури сервера частини веб-додатка, а також знижує вартість обладнання до оплати мінімально необхідного рівня режиму

Основна мета програми – надання послуг засобами Інтернет, необхідно включити в його серверну частину веб-сервер - апаратно-програмний комплекс, призначений для обслуговування HTTP-запитів. HTTP-запит - запит до сервера, сформований за протоколом HTTP/1.1 на попередньо визначений порт (за замовчуванням, порти 80 і 8080) для виконання будь-якої віддаленої дії (маніпулювання інформацією, виконання певної команди тощо). Зазвичай такі запити надсилає браузер клієнта.

Оскільки Web-приклад передбачає велику кількість операцій над читати, записувати і змінювати значний обсяг даних, найзручніше варіантом буде включення технологій баз даних у серверну частину.

На рисунку 2.2 показано процес деталізації початкової архітектури Web-прикладів, а також її серверної частини. Серверна частина тримається сам веб-сервер і сервер баз даних.

До завдань веб-сервера входять:

- Отримання і відповідь на HTTP запитів;
- Перенаправлення запитів на потрібний веб-додаток, зазвичай призначається певному домену або субдомену;
- Надання Web-прикладів доступу до необхідних модулів;
- Авторизація та аутендифікація користувачів;
- Реалізація функцій файл-сервера.

До завдань веб-сервера входять:

- Запити на послуги для маніпулювання даними на основі мов SQL;
- Обслуговування БД;
- Забезпечування цілісності даних;
- Надання утиліт для управління базою даних.

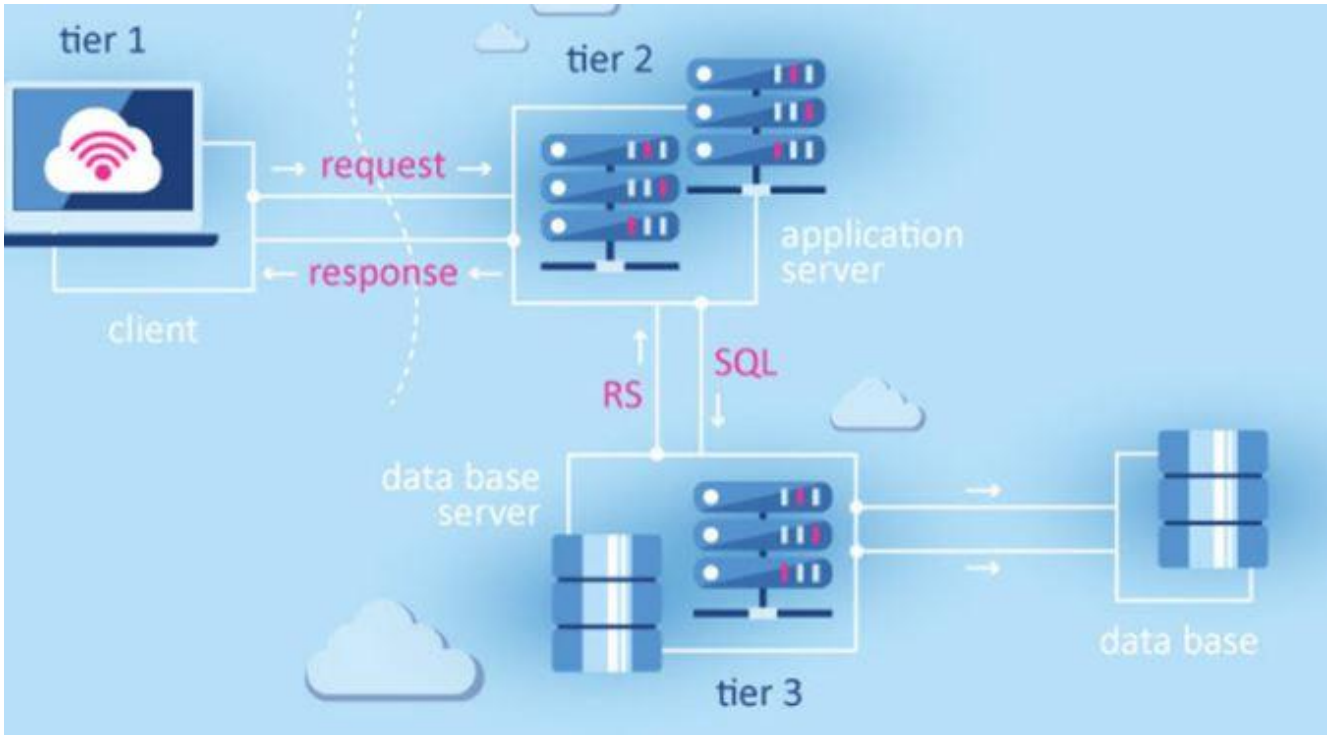


Рисунок 2.2. Деталізація серверної частини додатків.

Клієнтська частина програми повинна підтримувати такі технології:

- Доступ до мережі Інтернет;
- Можливість роботи по протоколу HTTP/1.1;
- Отримання інформації;
- Підтримка додатків для отримання данних;
- Проектування бази даних.

Вивчив предметну область та розробив структуру Web-додатку, можна переходити до наступного кроку - проектування структури бази даних. Як метод проектування бази даних обрано метод ER-діаграм.

Також потрібно розуміти типи контенту в WordPress. Перш ніж дізнатися, які дані містяться в базі даних WordPress, проаналізуйте типи контенту. Існують такі стандартні типи контенту:

- Помітки;
- Сторінки;
- Спеціальні типи записів;
- Вкладення ;
- Посиалання;
- Елементи меню.

Ці типи контенту мають такі дані:

- Категорії;
- Мітки;
- Спеціальні таксономії;
- Метадані.

Крім того існують типи контенту, які зберігаються в іншому вигляді:

- Віджети;
- Опції;
- Користувачі;
- Сторінки ждя MU WordPress;
- Спеціальний вміст який додає деякі теми або плагіни;

- Сторонній контент.

Усі ці типи вмісту зберігаються в таблицях або файлах бази даних налаштування теми/плагіна. Кожен тип може бути представлений як окремим записом у таблиці, так і його частиною. Крім того, вони можуть бути пов’язані з даними в інші таблиці. Наприклад, дані запису пов’язані з даними користувача, щоб WordPress знав, хто є автором і який запис.

WordPress використовує кілька взаємозалежних таблиць. Між ними встановлюються з'єднання "один до багатьох". Наприклад, на одній сторінці може бути багато коментарів.

На рисунку 2.3 показана логічна модель спроектованої бази дані.

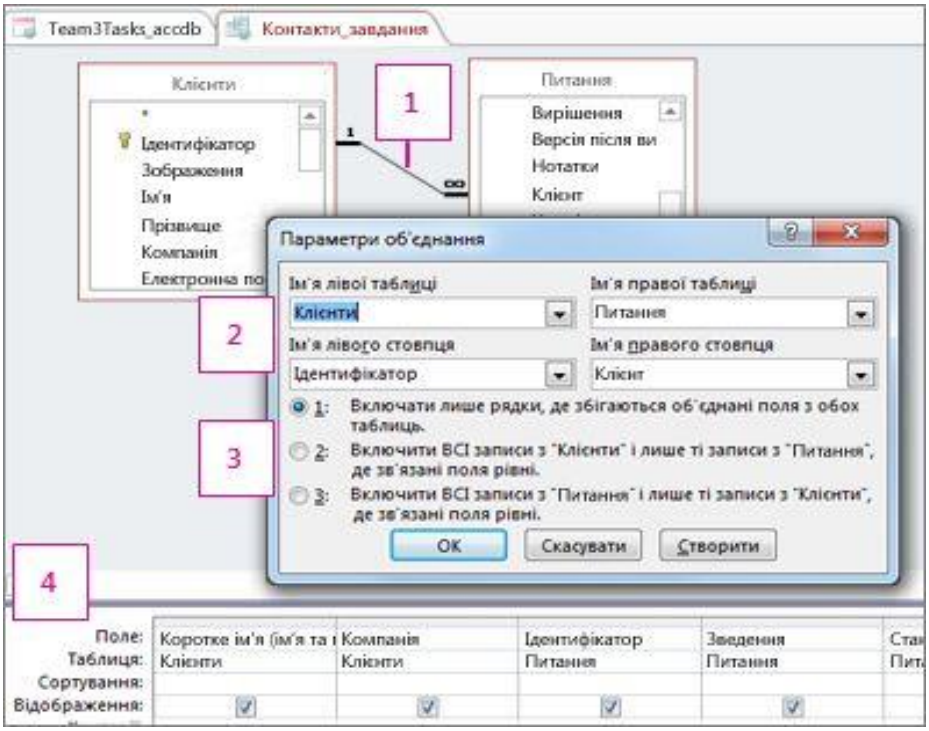


Рисунок 2.3. Уявлення таблиць.

Зміст руху бази даних WordPress набагато більше - це тому що вже є записи, сторінки, коментарі та встановлені плагіни, які також працюють на основі запитів до бази даних. Ось найпоширеніші лист 9.03.04.2017.104.ПЗ 17 WordPress налаштовує базу даних і запити, з якими вона насправді працює Веб-приклад створення WordPress:

- SELECT – вибрати строку з таблиці;

- INSERT – додати строку до таблиці. В яку саме таблицю, вказується в запросі;
- UPDATE – зміна строки в вибраній таблиці;
- DELETE – видалити строку в заданій таблиці;

Це SQL-запити (Structured Query Language - спеціальна мова для маніпулювання даними). Більшість таблиць пов’язані з однією або кількома іншими за допомогою одного поля. Це поле буде унікальним ідентифікатором для кожного запису (приклад post_id). У таблиці 2.1 представлені дані зв'язку:

Таблиця 2.1 – Таблиця зв’язку

Таблиця	Дані	Зв’язок з другими таблицями
m6sf_posts	Записи, сторінки, вкладення, редагування, користувацькі записи	m6sf_postmeta через post_id m6sf_term_relationships через post_id
m6sf_postmeta	Метадані публікації, сторінки тощо.	m6sf_posts через post_id
m6sf_comments	Коментарі	m6sf_posts через post_id
m6sf_commentmeta	Метадані коментарів	m6sf_comments через comment_id
m6sf_term_relations hips	Посилання між таксономіями та записами, сторінками тощо.	m6sf_posts через post_id m6sf_term_taxonomy через term_taxonomy_id
m6sf_term_taxonom y	Таксономія	m6sf_term_relationships через term_taxonomy_id
m6sf_terms	Ваші категорії, теги та Терміни користувацьких таксономій	m6sf_term_taxonomy через term_id

m6sf_links	Посилання у вашому блоці (Як правило, не зараз використаний)	m6sf _term_relationships через link_id
m6sf_users	Користувачі	m6sf_posts через post_author
m6sf_user_meta	Метадані для кожного користувача	m6sf_users через user_id
m6sf_options	Налаштування сайту	Немає

Зверніть увагу на деякі особливості цієї таблиці:

- Таблиці бази даних за замовчуванням мають префікс wp_. вимагається змінити його.
- Таблиця m6sf_posts являється найголовгішою. Саме в ній зберігаються більшисть даних.
- Тільки одна таблиця не пов’язана з іншими – таблиця m6sf_options. Вони зберігають дані про сайт і налаштування WordPress, які не стосуються записів або користувачів.
- Дві таблиці використовуються для зберігання даних про таксиномію.
- Таблиці в m6sf_users і m6sf_comments не пов’язані. IN у налаштуваннях WordPress є можливість вказати, що лише зареєстровані киристувачі можуть залишити коментарі. Однак WordPress не підтримає зв’язок коментарі та користувач, який їх опублікував.

Прочитавши типи вмісту в WordPress і таблицю баз даних, які використовуються для їх зберігання, ви можете почерпнути відповіді між ними. У наступній таблиці 2.2 показано, які таблиці бази даних для збереження типу вмісту:

Таблиця 2.2 – Таблиця для типу контенту

№	Тип контенту	Таблиця
1	Записи	m6sf_posts
2	Сторінки	m6sf_posts

3	Користувачеві типи записів	m6sf_posts
4	Вкладення	m6sf_posts
5	Посилання	m6sf_links
6	Елементи меню	m6sf_posts
7	Категорії	m6sf_terms
8	Мітки	m6sf_terms
9	Користувацькі таксономії	m6sf_term_taxonomy
10	Терміни користувацьких таксиномій	m6sf_terms
11	Методані	m6sf_postmeta
12	Віджети	m6sf_options
13	Налаштування	m6sf_options
14	Користувачі	m6sf_users
15	Нестандартний контент	m6sf_posts, m6sf_options, файли тем або плагинів
16	Інший контент	m6sf_posts, m6sf_options, файли тем або плагинів

Зверніть увагу, що не всі таблиці використовуються в списку. Це тому, що деякі з них використовуються для зберігання метаданих. Інші використовуються для зберігання з'єднань

2.2 Технологічне забезпечення роботи

У цьому розділі відповідно до розробленого веб-додаток буде відповідати вимогам до додатків технологічної платформи (програмний компонент), а також у відповідних підрозділах огляд наявних рішень і вибір найбільш оптимального варіанту.

Операційна система (далі - ОС) (від англ. Operating system) – складна програми, які забезпечують базовий набір функцій керування обладнанням за допомогою комп'ютера. ОС є сполучною ланкою між програмами і комп'ютерне обладнання. Основні вимоги до ОС, призначеної для використання в серверна частина програми:

- Підтримка повного спектру мережевих технологій (мережеве обладнання, протоколи, доступ до віддалених ресурсів і сервісів тощо);
- Багатозадачність;
- Багатопоточність;
- Багатокористувацький режим;
- Підтримка різних апаратних платформ (64=розрядні процеси, системи зберігання даних на жорстких дисках тощо);
- Розширюваність;
- Доступний рівень безпеки (наявність механізмів авторизації, аутентифікації, аудит);
- Низька вартість копії ліцензії;
- Підтримка повного спектру мережевих технологій (мережеве обладнання, протоколи, доступ до віддалених ресурсіві сервісів тощо).

Сучасні операційні системи можна розділити на дві великі групи: системи Windows і Unix. Серед серверних рішень Windows 10 виділяються такі операційні системи:

Windows 2016 Standard — серверна операційна система від Microsoft. Система є частиною сімейства Windows NT і розробляється одночасно Windows 10. Це хмарна операційна система, яка може розвинути лист 09.03.04.2017.104.ПЗ 21 рівень безпеки та більш ефективна робота. Пропонує можливості корпоративного класу з обмеженими правами віртуалізації.

Windows Server 2016 Essentials — це економічно ефективне технічне рішення для бізнесу. Це найкраще рішення для першого сервера, підключеного до області.

Це видання призначене для власників невеликих компаній та IT-фахівці, які їх підтримують.

Серед систем Unix для вирішення поставлених завдань можна виділити дві Тип ОС: MacOS X Server і ОС на базі Linux.

Серед операційних систем на базі ядра Linux можна виділити:

Debian — це система, яка поєднує новітні технології з максимальною ефективністю. В даний час він є одним з найнадійніших і складні дистрибутиви Linux з найкращою системою управління пакетами, великою спільнотою, високими стандартами та відповідальним ставленням до тестування пакетів. Це безкоштовна високоякісна Unix-сумісна операційна система з повним набором програм. Установка Debian досить заплутана і складна, ця система підготовлена користувачів.

Поширюється на безкоштовній основі.

Ubuntu Server — це дистрибутив операційної системи Linux на основі Debian GNU / Linux. Генеральним спонсором Ubuntu є Canonical Ltd., заснована Марком Шаттлвортом. Проект активно розвивається та підтримується вільною спільнотою.

Ubuntu Server — це серверна версія дистрибутива, яка дозволяє менше ніж за 10 хвилин налаштувати і запустити повноцінний веб-сервер і сервер баз даних.

Red Hat Enterprise Linux — це дистрибутив Red Hat GNU/Linux.

Цей дистрибутив призначений для корпоративного використання. Нові версії виходять кожні 3 роки.

Основна особливість дистрибуції - наявність комерційної підтримки. Багато постачальників програмного та апаратного забезпечення включені RHEL є одним із підтримуваних ними дистрибутивів GNU/Linux.

Найкращий варіант - Ubuntu 16.04 LTS. Основні з них фактори, пов'язані з рішенням:

- Безкоштовність рішення;
- Підтримка цієї ОС багатьма хостинг-провейдерами;
- Наявність докладної документації;

- Наявність виликої кількості тематичних спільнот;
- Версія LTS (Довгострокова підтримка) період означає, що для розповсюдження версії 16.04 до 2022 року будуть випущені оновлення программного забезпечення та безпеки;
- У комплекті поширене програмне забезчення, що означає, що для встановлення програми з такого пакету потрібен лише один файл, інші є або вже є в системі, або будуть завантажені автоматично, а також така система дозволяє оновлювати програмне забезпечення в автоматичному режимі;
- Велика кількість BD-серверів і веб-серверів, які можна використовувати з цим дистрибутивом як програмний пакет.

Наразі існує велика кількість СУБД.

СУБД – це набір мовних і програмних засобів, розроблених створюватипідтримувати та ділитися базою даних багатьма користувачами. Зазвичай СУБД розрізняють за використовуваною моделлю даних. Таким чином, СУБД, засновані на використанні реляційних моделей даних, називаються реляційними СУБД.

Критерії для підбору СУБД:

- стабільність;
- функціональність;
- вартість.

Найпопулярніші СУБД:

- Microsoft SQL Server Express;
- MySQL;
- PHPMyAdmin.

Microsoft SQL Server Express - система управління реляційною базою даних База даних (СУБД), розроблена Microsoft. Основний використаний Мова запитів - Transact-SQL, створена у співпраці з Microsoft і Sybase. Transact-SQL є реалізацією стандарту ANSI / ISO для мови структурованих запитів (SQL) з розширеннями.

Використовується для роботи з базами даних розмірів від персональних до великих корпоративних баз даних; конкурує з інші СУБД у цьому сегменті ринку.

SQL Server Express включає 10 ГБ для кожної бази даних даних, зручні функції резервного копіювання та відновлення відповідає всім випускам баз даних SQL Server і Microsoft Azure SQL. Поширюється на безкоштовній основі.

MySQL — це безкоштовна система керування реляційними базами даних. MySQL розроблено та підтримується корпорацією Oracle, отримано лист 09.03.04.2017.104.ПЗ 24 права на торговельну марку разом із поглибленою Sun Microsystems, яка раніше придбала шведську компанію MySQL AB. Товар поширюється як п.п GNU General Public License та за власною комерційною ліцензією. Крім того, розробники створюють індивідуальні функціональні можливості користувачів. Саме завдяки такому порядку майже в самих ранніх версіях з'явився механізм реплікації.

Гнучкість бази даних MySQL підтримується великою кількістю опор Типи таблиць: користувачі можуть вибрати як таблиці MyISAM, які підтримують повнотекстовий пошук, так і таблиці InnoDB, які підтримують транзакції на рівні транзакцій. Крім того, надається база даних MySQL особливий тип таблиці EXAMPLE, що демонструє принципи створення нові типи столів. Завдяки відкритій архітектурі та ліцензуванню GPL, Бази даних MySQL - це постійно нові типи таблиць.

MySQL має подвійне ліцензування. MySQL можна поширювати в відповідно до умов ліцензії GPL. Однак, згідно з умовами GPL, якщо будь-яка програма містить вихідний код MySQL, вона також має бути розширена відповідно до ліцензії GPL. Його можна витратити на плани розробників, ні бажаючи відкрити вихідний код своїх програм. Для таких випадків надається комерційна ліцензія, яка також забезпечує якісну сервісну підтримку. Поширюється на безкоштовній основі.

На основі отриманих даних був обраний для розробки програми MySql, оскільки він є повнофункціональним і безкоштовним, також не має жодних обмежень щодо розміру бази даних. Також для цієї реляційної системи є багато

додаткових спрощень базового адміністрування дані. Найчастіше це робить PhpMyAdmin.

PHPMyAdmin — це безкоштовна програма з відкритим кодом, призначена для адміністрування баз даних MySQL. PHPMyAdmin є веб-інтерфейси, за допомогою яких можна адмініструвати сервер MySQL, запускати команду та переглядати вміст таблиць і баз даних через браузер. Це лист 09.03.04.2017.104.ПЗ 25 Програма дуже популярна через свої переваги: можливість керувати базами даних MySQL без безпосереднього введення команди SQL, оскільки панель керування PHPMyAdmin надає можливість адмініструвати вибрані База даних, інтенсивна розробка, можливість інтегрувати PHPMyAdmin у власні розробки завдяки GNU General Public License та іншим функціям.

На основі цієї інформації PhpMyAdmin був обраний як веб-інтерфейс для бази даних MySql, оскільки цей веб-приклад дуже популярний і поширюється безкоштовно.

Web-приклад складається з двох еквівалентних частин: сервера і клієнта. Можливо, використовувати одну мову програмування для всіх додатків, однак, це найбільш неоптимальний варіант.

Висновок до розділу 2

Таким чином з’являється розуміння, як працюють веб-розробники, які в них задачі окрім створення сайтів за допомогою HTML та CSS, а з’являється і багато інших завдань, як роботи з базами і так далі. В цьому розділі йдеться, про накращі інструменти для роботи на сьогодні, описані переваги та описані недоліки, щоб розуміти, що і коли використовувати та для чого воно створене.

3 ДОСЛІДЖЕННЯ ОСОБЛИВОСТЕЙ РОЗРОБКИ ВЕБ-ДОДАТКУ З ВИКОРИСТАННЯМ ANGULAR 2

3.1 Робота з Angular

На основі офіційних прикладів підготовлено робочий документ приклад веб-додатка, який розгляне особливості розробки односторонній веб-додаток, який використовує фреймворк Angular 2.

3.1.1 Встановлення програмного забезпечення

Як згадувалося вище, фреймворк Angular 2 пропонує розробнику повну архітектура та структура веб-додатків. Як серверна технологія Angular 2 пропонує використовувати платформу Node.js на базі машини JavaScript 8.

Ви можете встановити його за допомогою інсталятора з офіційного сайту https://nodejs.org/. Щоб запустити приклад, необхідно встановити версії Node.js 5.k.k і npm 3.k.k або новіші. Щоб перевірити встановлену версію Node.js і npm необхідно ввести в консольні команди, показані на малюнку 21.

Інші залежності можна налаштувати автоматично, завдяки вбудований менеджер пакетів npm.

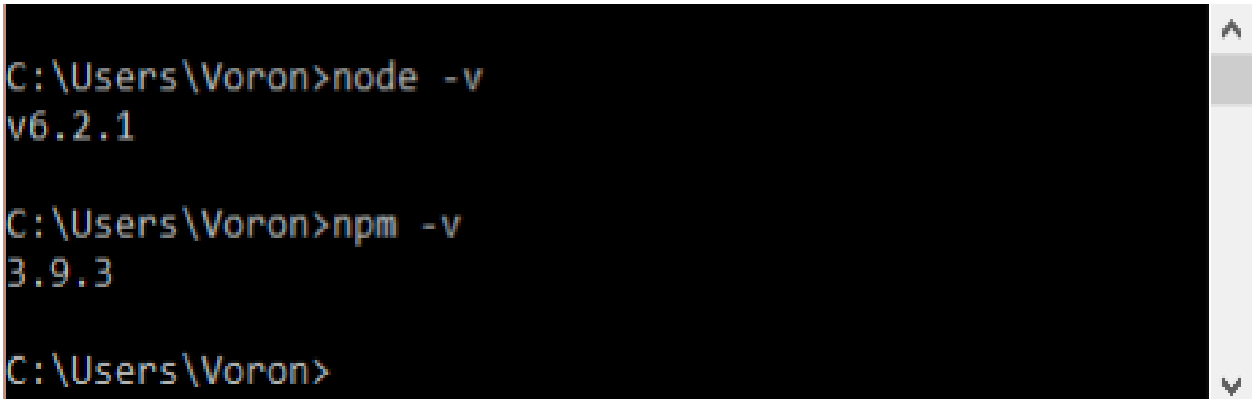


Рисунок 3.1. Таблиця.

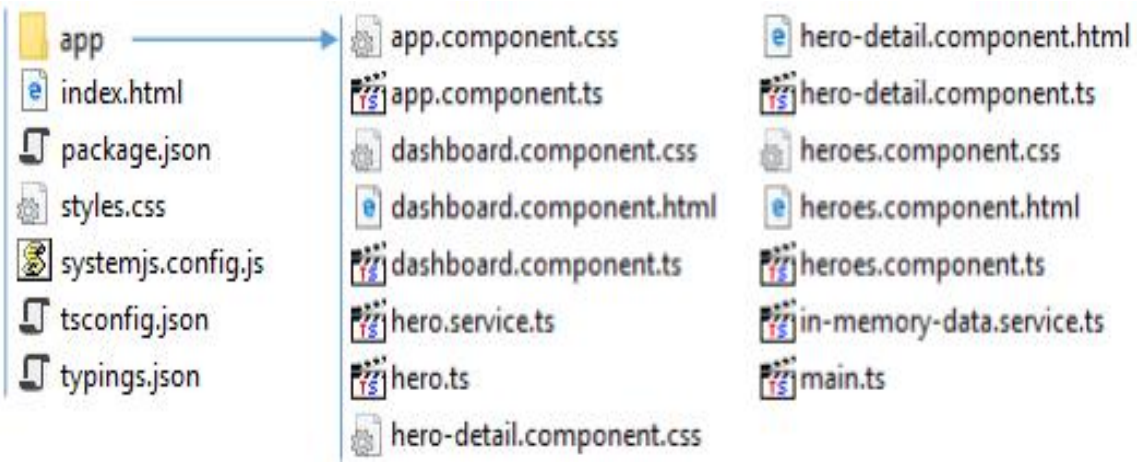


Рисунок 3.2. Структура файлів.

Список усіх зразків файлів можна знайти в Додатку А. Файли, придатні для роботи сервера:

- Package.json – це файл менеджера пакетів npm, який містить список і версії залежностей, необхідних для запуску прикладу, і також деякі команди для створення та запуску серверів.
- Tsconfig.json – TypeScript компіляторна конфігурація.
- Sistemjs.config.js – це конфігурація системного драйвера модуля SistemJS.

За роботу клієнтської частини відповідають наступні файли:

- Index.html – це головна сторінка веб-програми, яка містить тільки необхідні основні теги та сценарії, які завантажуються синхронно ресурси для повнного огляду програми.
- Main.ts – містить код, який завантажує інфраструктуру фреймворку Angular 2 підтримує і запускає надбудову з основним компонентом AppComponent. За це відповідає функція bootstrap.
- Heroes.ts – має TypeScript код клсу Hero.
- hero.service.ts - містить TypeScript код сервісу HeroService.
- in-memory-data.service.ts - містить TypeScript код сервісу InMemoryDataService, який надає набір даних для прикладу.
- styles.css - містить загальні спільні стилі необхідні для роботи додатку.

Кожна компонента AppComponent, HeroesComponent, HeroDetailComponent, DashboardComponent мають файл, "* .ts", який містить Yogo Введіть код, який може містити файли "* .html" і "* .css", які містять HTML Шаблон стилю SSS і компонент відповідно. Відповідно до рекомендованих норм кодування компонентів Angular 2 може містити в назві "компонент".

Управління Node.js і npm здійснюється через консоль. Отже, перш за все вам потрібно відкрити консоль і перейти в папку проекту. Вам потрібно буде запустити його вперше або після внесення змін до файлу package.json виконати процедуру відновлення залежностей, під час якої він буде завантажений необхідні версії всіх необхідних залежностей. Щоб виконати цю процедуру ви повинні ввести команду npm install у консолі.

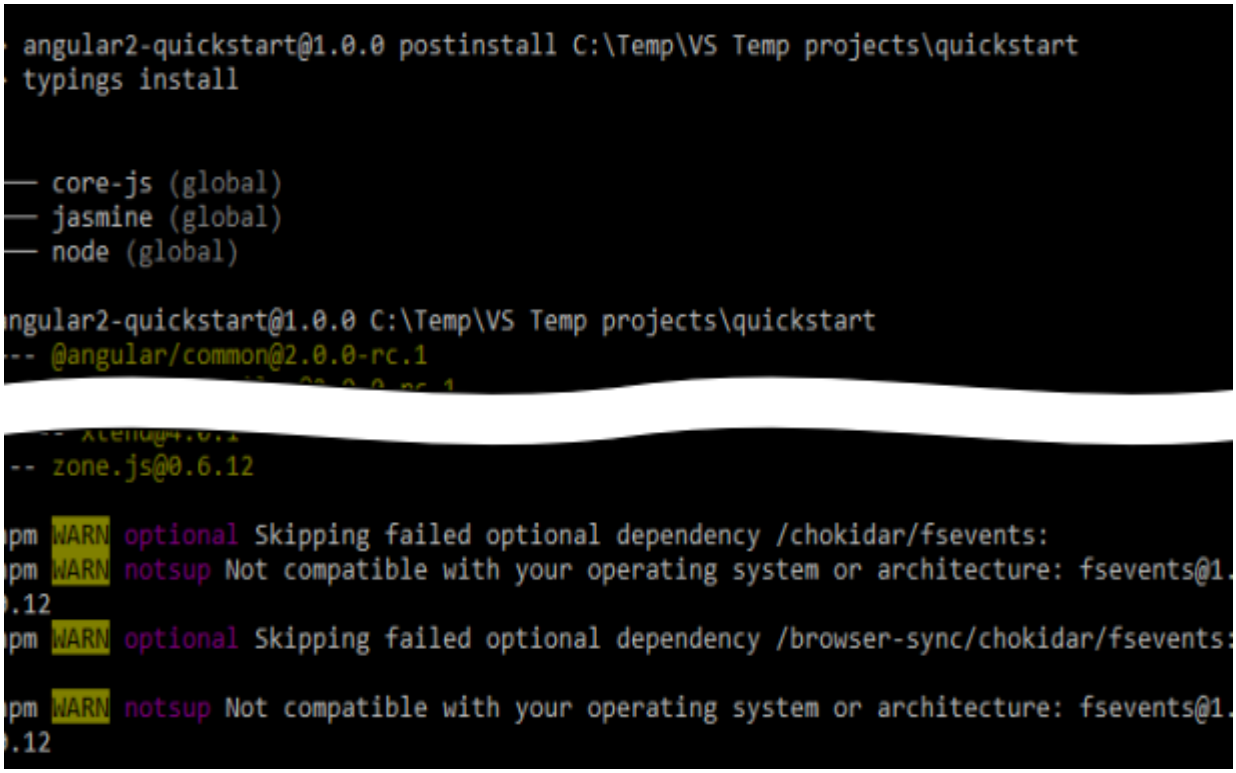


Рисунок 3.3. результат команди npm install.

Для того, щоб запустит проект потрбна команда npm start.

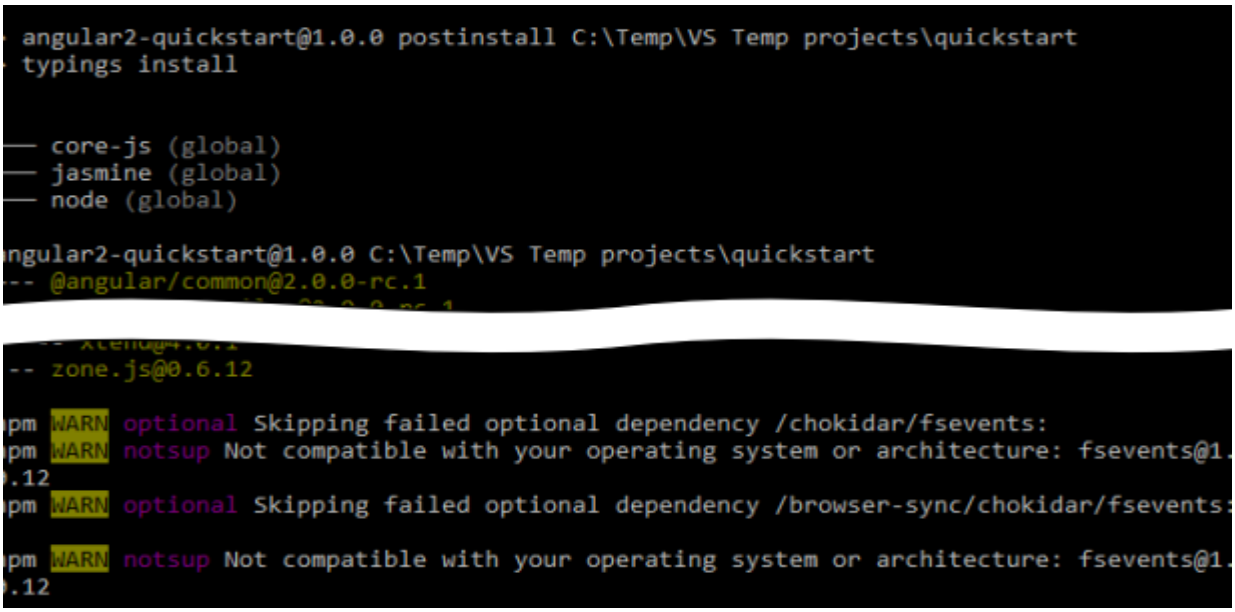


Рисунок 3.4. дані консолі при запуску.

Після успішного запуску сервера веб-сайт буде доступний за такими адресами. Також для додаткових адрес, які також відображаються на консолі, буде доступний графічний інтерфейс керування сервером.

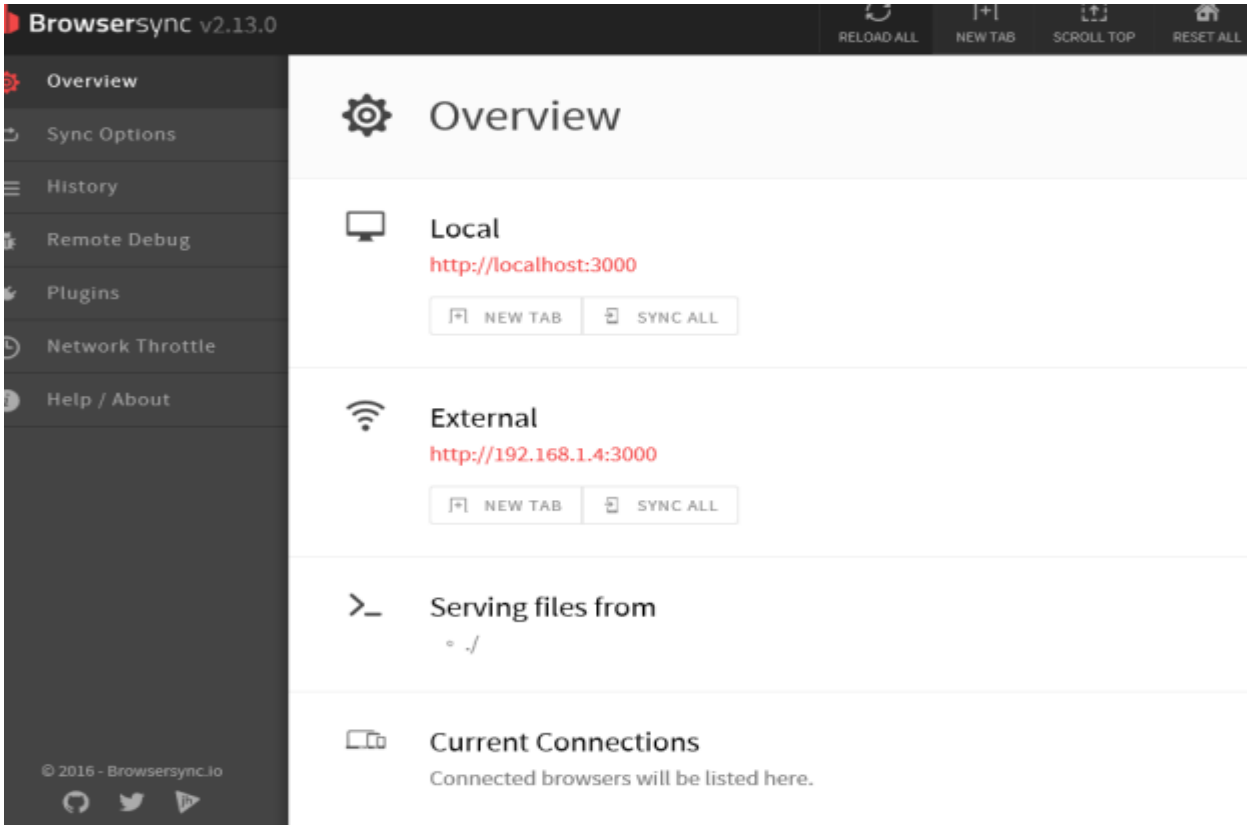


Рисунок 3.5. Інтерфейс керування сервером.

3.1.2 Робота з компонентами

Доповнення складається з основного компонента AppComponent, який містить налаштування маршрутизації клієнта та є контейнером для компоненти, які надають поточний вигляд або відчуття програми:

- DashboardComponent – компонент-представництво для дії. Містить кнопки навігації для компонентів HeroesComponent і список найкращих героїв 40 натиснувши на які компоненти дисплея відкриються HeroDetailComponent з подробицями про героя.
- HeroesComponent – компонент дипсплея, який містить кнопку перейдіть до компонента Dashboard список всіх героїв, натиснути на який можна побачити їх деталі Компонент перегляду HeroDetailComponent.
- HeroDetailComponent – цу компонент дисплея, яякий відображає і дозволяє щмінювати деталі конкретного персонажа і містить кнопки повернення до огляду та до компонентів DashboardCompont або HeroesComponent.

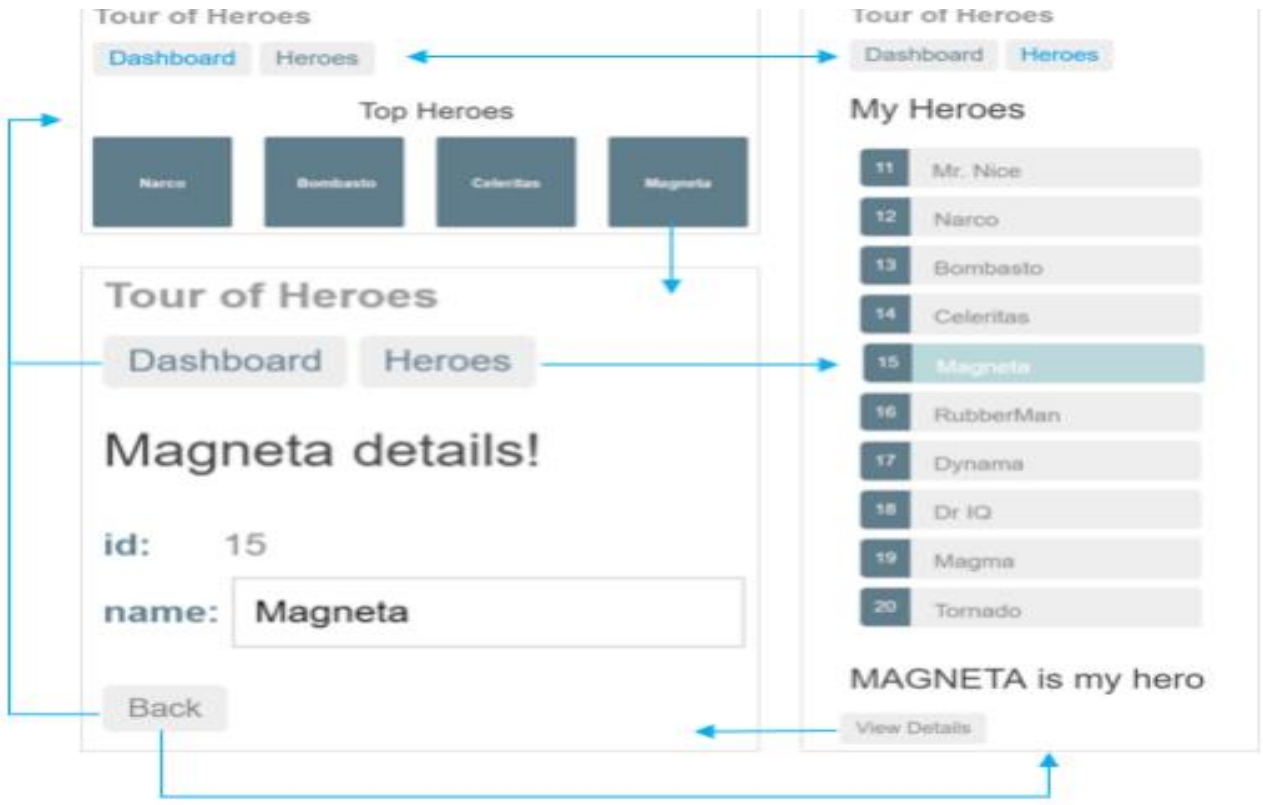


Рисунок 3.6. Інтерфейс керування сервером.

3.2 Модульність з Angular

Angular 2 рекомендує використовувати модульний підхід і самі модулі ES2015, які мають хорошу підтримку в TipeScript, хоча вона не потрібна. Використання масштабів TipeScript можна замінити стандартним JavaScript, якщо необхідно, оскільки будь-який код TypeScript компілюється в JavaScript. Використання модулів і TipeScript значно спрощує написання та структурування і відмовлятися від їх підтримки не бажано.

Модулі подібні до просторів імен у мовах програмування, таких як C# і S ++. Вони мають внутрішній і зовнішній код (класи, функції, значення), тобто які інші модулі можна використовувати. Щоб позначити зовнішній код модуль експорту використовує модифікатор "export", наприклад:

export class AppComponent { }

Щоб підключити частину іншого модуля, необхідно використовувати наступний синтаксис:

import { AppComponent } from './app.component';

Оскільки модулі ES2015 підтримуватимуть майбутнього, підтримувати їх в сучасних версіях пошукових систем необхідно додатково використовуйте менеджер модулів, такий як SystemJS.

3.3 Архітектура Angular

Веб-додаток Angular 2 складається з компонентів, директив і послуги. Компоненти контролюють певну частину елементів DOM і містять їх логіку їхнього мислення та поведінки. Директиви, на відміну від компонентів вони містять лише поведінку. Сервіси розроблені для легкого впровадження екземпляри класу в компонентах і директиви програми, які залежать від них, тобто сервіси реалізують шаблон оформлення Один.

Для позначення класу за допомогою компонента, директиви або служби Angular 2 використовуються метадані і потрібна сама функція декоратора Параметри перед класом.

- *ngFor - структурна директива що інструктує фреймворк створити DOM елемент для кожного елементу з вказаного списку. Приклад:
  <div *ngFor="let hero of heroes">
- *ngIf - Структурна директива, яка вимагає від структур створення HOME елемент лише для певного розуму. Приклад:
  <hero-detail *ngIf="selectedHero"></hero-detail>
- [(…)] = "..." Це директива атрибутів, яка оголошує двостороннє відношення. Приклад:
  <input [(ngModel)]= "hero.name">
- [(…)] = "..." "Це директива атрибутів, яка оголошує двостороннє відношення.

Структурна директива, яка вимагає від структур створення HOME елемент лише для певного розуму. приклад:

Рисунок 3.7. Angular архітектура додатку.

Компоненти можуть використовуватися іншими компонентами як діти, складання відповідної ієрархії елементів програми. Краватки працюють так само а також між батьківським і дочірнім компонентом.

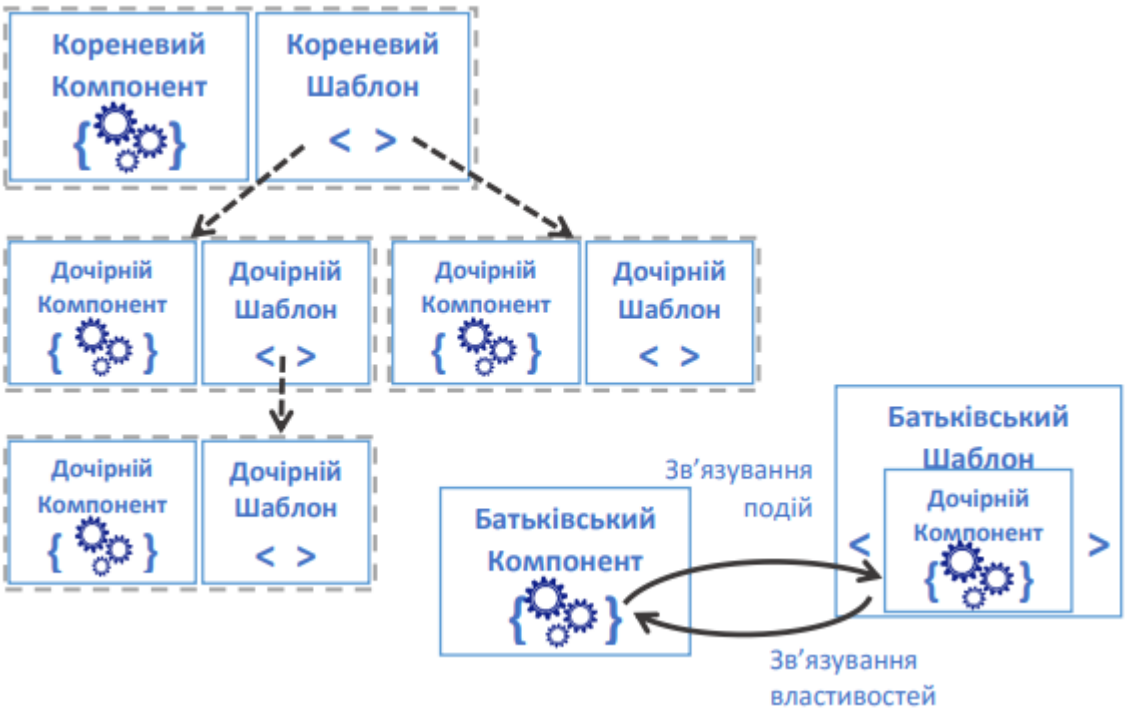


Рисунок 3.8. Ієрархія та зв'язок компонентів Angular 2 додатку.

Щоб створити компонент, вам потрібно обернути клас, який ви хочете, щоб він містив поведінка компонента як декоратора @ Component.

Розглянемо наступний приклад компонента HeroesComponent.

```
@Component({
    selector: 'hero-list',
     templateUrl: 'app/hero-list.component.html',
     directives: [HeroDetailComponent],
     providers: [HeroService]
})
export class HeroesComponent { ... }
```

Деякі параметри декоратору компонента:

Selector визначає селектор css, для якого буде вібрувати фрейм html елементи, які будуть відтворювати компоненти, тобто бути контейнерами для компонента. Селектор у прикладі каже вам відтворити кадр тег Hero-list> </ hero-list> як компонент HeroesComponent.

templateUrl вказує посилання на шаблон html для відтворення компонент. Його зручно використовувати для невеликих шаблонів альтернативний варіант шаблон.

Directives визначає модулі, які використовуються в компоненті.

Providers вказує необхідні компоненти послуги, екземпляри реалізації які рамки здати дизайнеру.

Директиви поділяються на структурні, які додаються, вилучаються або змінюються Елементи та атрибути DOM, які змінюють вигляд або поведінку існуючого html елементів. Використовувати достатньо вбудованих звичок для створення плагінів директива. Деякі популярні директиви:

- *ngFor - структурна директива що інструктує фреймворк створити DOM елемент для кожного елементу з вказаного списку.
- *ngIf - структурна директива що інструктує фреймворк створити DOM елемент лише за певної умови.
- . «[(…)]=”…”» - атрибутна директива що декларує двостороннє зв’язування.

Для того щоб створити власну директиву треба обернути клас який буде містити поведінку директиви у функцію-декоратор «@Directive»

3.4 Шаблони Angular

Angular 2 має ряд вбудованих директив, які дозволяють зв’язувати властивості, атрибути та події, гнучко та зручно керувати зовнішнім виглядом компонентів завдяки шаблонам. Якщо це при необхідності користувач може створювати власні директиви.

Angular 2 дозволяє зв’язувати властивості, атрибути та події елементи шаблону компонента з властивостями та функціями компонента та дочірніх компонента. З'єднання є декларативним за використанням спеціальні синтаксис в шаблоні:

1. Інтерполяція - дозволяє в односторонньому порядку оновлювати властивості та вміст Елементи DOM під час оновлення значень компонентів.
2. Властивості зв'язування - дозволяє в односторонньому порядку пов'язувати властивості елементів DOM і дочірніх компонентів з властивостями компонента.
3. Пов'язування подій - дозволяє зв'язувати події елемента DOM компонентні методи.
4. Двостороннє зв'язування - дозволяє двостороннє зв'язування властивостей DOM-елементи та підкомпоненти з компонентом.
5. Прив'язка атрибутів - дозволяє встановити значення атрибута DOM елемент при створенні.
6. Клас зв'язування - дозволяє включати та виключати клас з DOM елемент залежно від значення виразу.
7. Прив'язка стилю - дозволяє вмикати і вимикати стиль з DOM елемент залежно від значення виразу.

Структурні директиви дозволяють додавати, видаляти, повторювати або ви змінюєте елементи DOM залежно від значень певних змінних, властивості компонентів або їх вирази.

1. Умовна директива - дозволяє додавати та видаляти елемент DOM залежно від значення логічного виразу.
2. Directive switch - дозволяє змінити елемент DOM на залежності від значень виразів.
3. Директива Repeat - дозволяє створити елемент DOM для кожного елемента списку.

Для організації маршрутизації клієнтів в Angular 2 є спеціальний компонент "ComponentRouter", який дозволяє вам маршрутизувати до базові компоненти.
Для роботи необхідно виконати кілька кроків:

1. Включити "" до розділу "<base-href="/ ">" сторінки "index.html".
2. Підключіть такі модулі: імпорт {RouteConfig, ROUTER_DIRECTIVES, ROUTER_PROVIDERS} від "@ angular / routerprecated".
3. До компонента, який використовуватиме ComponentRouter, необхідно додати такі директиви та служби, такі як директиви: [ROUTER_DIRECTIVES], постачальники: [ROUTER_PROVIDERS].
4. Додати конфігурацію маршрутів і компонентів, які їм підходять, такі як шлях: "/ герої", Ім'я: "герої", компонент: HeroesComponent.
5. Додати до шаблону мітку "Router-outlet>". компоненти, які буде використовувати ComponentRouter.

Після цього при переході до «/heroes» компонент маршрутизації завантажить компонент "HeroesComponent" в елемент DOM.

Висновок до розділу 3

У цьому розділі розглядаються характеристики розробки за допомогою фреймворку Angular 2 на прикладі офіційних прикладів фрейму. Також обговорювалася переважна більшість основних характеристик каркаса та спосіб їх використання. Детально описано структуру програми, програмне забезпечення, необхідне для його роботи, а також процес створення та запуску програми.

Можна зробити висновок, що фреймворк Angular 2 дуже гнучкий і зручний і має багато можливостей для створення сучасних веб-додатків на одній сторінці. Найбільшим недоліком фреймворка є те, що на момент написання він все ще знаходиться в розробці, що робить його ризикованим для використання у великих проектах, але дає надію на розширення його можливостей у остаточній версії.

4 ДОСЛІДЖЕННЯ ОСОБЛИВОСТЕЙ РОЗРОБКИ ВЕБ-ДОДАТКУ З ВИКОРИСТАННЯМ ANGULAR 2

У цьому розділі оцінюються основні особливості програмного продукту, призначеного для демонстрації концепції односторінкових вебдодатків. Прототипи ігор були розроблені за допомогою Unity та Construct2.

Програмні продукти призначені для особистого використання комп'ютери, які використовують будь-яку операційну систему.

Нижче наведено аналіз різних варіантів впровадження модуля з метою вибору найкращого, враховуючи як економічні фактори, так і характеристики продукту, що впливають на продуктивність і його сумісність з обладнанням. Для цього використано апарат функціонально-вартісного аналізу.

Функціональний аналіз витрат (FVA) – це технологія, яка дозволяє оцінити фактичну вартість товару чи послуги, незалежно від організаційної структура компанії. Як прямі, так і побічні витрати розподіляються на продукти і послуги залежно від кількості ресурсів, необхідних на кожному етапі виробництва. Дії, що виконуються на цих фазах у контексті методу FVA, називаються функціями.

Метою FVA є забезпечення належного розподілу призначених ресурсів виробництво або надання послуг, прямі та непрямі витрати. У цьому case - аналіз функцій програмного продукту та ідентифікація всіх витрати на реалізацію цих функцій.

Фактично цей метод працює за таким алгоритмом:

- визначає порядок виконання функцій, необхідних для виробництва продукції. Спочатку – можливо все, потім вони поділяються на дві групи: ті, що впливають на ціну товару, і ті, що ні. На цьому ж етапі оптимізується сама послідовність 49 зменшення кроків, які не впливають на значення і, відповідно, витрати;
- для кожної функції визначаються загальні річні витрати та кількість робочі години;

- для кожної функції визначається на основі оцінки з попереднього пункту кількісна характеристика джерел витрат;
- після того, як будуть визначені джерела їх вартості для кожної функції, проводиться остаточний розрахунок витрат на виробництво продукції.

4.1 Постановка з проблеми техніко-економічного аналізу

Для техніко-економічного аналізу використовується метод FVA система аналізу нелінійних нестаціонарних процесів. Оскільки основний проектні рішення стосуються всієї системи, кожної окремої підсистеми це повинно їх задовольнити. Отже, справжній аналіз — це функціональний аналіз програмний продукт, призначений для збору, обробки та аналізу гетероскедастичні процеси в економіці та фінансах.

Відповідно, слід вибрати систему індикаторів якості програмного забезпечення продукт.

Технічні вимоги до продукту такі:

1. Програмний продукт повинен працювати на персональному комп'ютері зі стандартним набором компонентів.
2. Забезпечують високу швидкість обробки даних і реагування користувача в режимі реального часу.
3. Забезпечують зручність і легкість взаємодії.
4. Користувачеві надати зручну можливість масштабування сервісу.
5. Забезпечити мінімальні витрати на впровадження програмного забезпечення продукт.

4.1.1 Пояснення функцій програмного продукту

Основна функція $Ф_0$- розробка програмного продукту, що аналізує процес на на основі вхідних даних і будує її модель для подальшого прогнозування. На основі Для конкретних цілей можна виділити такі основні функції ПП:

- $Ф_1$ - вибір серверної технології;
- $Ф_2$ - вибір клієнтської рамки;
- Ф3 – Вибір менеджер клієнтських модулів. Кожна з основних функцій може мати кілька варіантів реалізації.

Кожна з основних функцій може мати кілька варіантів реалізації. Функція $Ф_1$:

1. ASP.NET.
2. PHP.
3. Node.js.

Функція $Ф_2$:

1. Angular 2.
2. Backbone.
3. Ember.

Функція $Ф_3$:

1. Webpack.
2. Browserify.
3. SystemJS.

4.1.2 Варіанти реалізації основних функцій

Варіанти реалізації основних функцій наведені в морфологічній карті системи. На основі цієї карти було побудовано позитив-негатив матриця варіантів основних функцій.

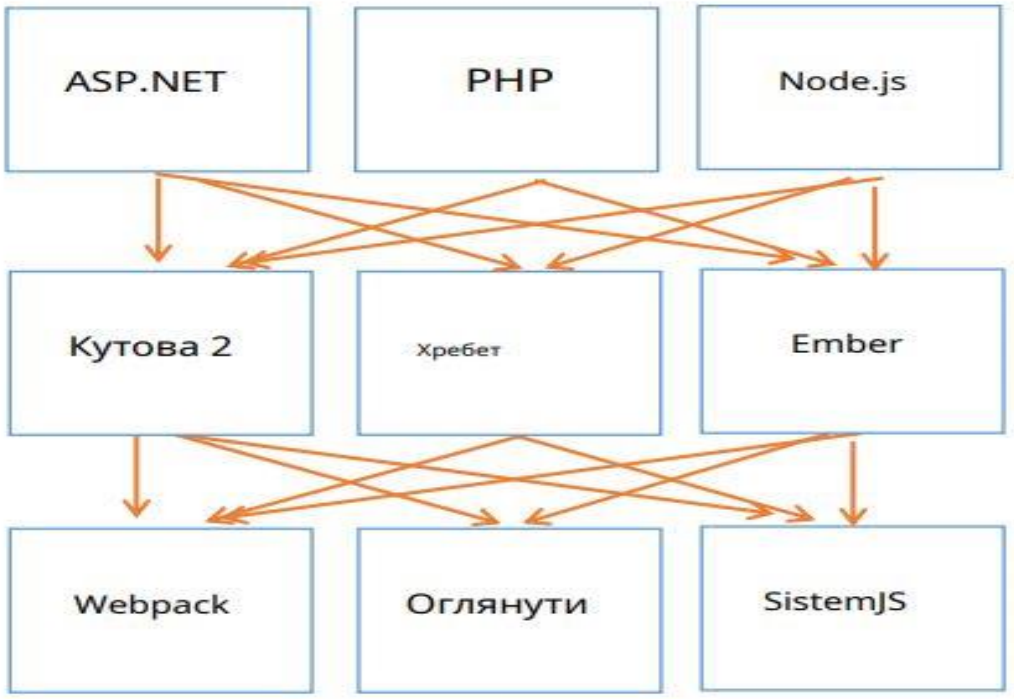


Рисунок 4.1. Ієрархія та зв’язок компонентів Angular 2 додатку

Морфологічна карта показує всі можливі комбінації варіантів для реалізація функцій, що складають повний набір варіантів ПП.

Таблиця 4.1 - Позитивно-негативна матриця

Основні функції	Варіанти реалізації	Переваги	Недоліки
$Ф_1$	А	Великі можливості платформи та зручне масштабування	Прив’язка до платформи .Net
	Б	Низький поріг входу для новачків,	Важче відладити
	В	Кросплатформеність, спільна мова для клієнта та сервера	Погана швидкість та можливості відладки

$Ф_2$	А	Потужний найновіший фреймворк з готовою архітектурою	Продукт ще знаходиться в стадії розробки
	Б	Бібліотека не потребує певної архітектури, використовує інші відомі бібліотеки	Найстаріший з обраних варіантів
	В	Зручність, надійність	Не використовую найсучасніші технології
$Ф_3$	А	Багаті можливості налаштування та пакування	Необхідність складного налаштування
	Б	Динамічне оновлення скриптів при змінах, зручність розробки	Втрачає популярність на користь більш новим бібліотекам, не підтримує майбутній формат модулів
	В	Реалізовую синтаксис який в майбутньому буде підтримуватися браузера без використання додаткових бібліотек	Погана можливість налаштування

На основі аналізу позитивно-негативної матриці робимо висновок при розробці програмного продукту деякі варіанти реалізації функції слід відхилити,

оскільки вони не відповідають цілям програмного забезпечення продукт. Ці варіанти позначені на морфологічній карті.

Функція $Ф_1$:

Тому що для вивчення трьох серверів потрібно занадто багато часу технологій, вибираємо лише 2 варіанти А і Б.

Функція $Ф_2$:

Також виберіть лише 2 кадри, параметри А та В.

Функція $Ф_3$:

Варіант А занадто складний для налаштування, варіант Б застарів і втрачає популярність, тому залишимо варіант В.

Отже, ми розглянемо наступні варіанти реалізації ПП:

1. Ф1а - Ф2а - Ф3в
2. Ф1б – Ф2в – Ф3в

Для оцінки якості розглянутих функцій була обрана описана система параметрів нижче.

4.2 Обґрунтування системи параметрів ПП

4.2.1 Опис параметрів

На основі даних про основні функції, які він має реалізувати програмний продукт, вимоги до нього, визначають основні параметри продукту який буде використаний для розрахунку коефіцієнта технічного рівня.

Для характеристики програмного продукту будемо використовувати такі параметри: На основі даних про основні функції, які повинен реалізувати програмний продукт, вимоги до нього визначимо основні параметри продукту, які будуть використовуватися для розрахунку технічного рівня.

Для характеристики програмного продукту будемо використовувати наступні параметри:

1. X1 - швидкість мови програмування серверної мови;
2. X2 - обсяг пам'яті для зберігання даних;

3. X3 - час обробки даних;
4. X4 - потенційний обсяг програмного коду.

X1:Відображає швидкість виконання операцій мовою програмування залежно від обрані серверні технології.

X2:Відображає обсяг пам’яті персонального комп’ютера RAM, необхідний для зберігання та обробки даних під час виконання програми.

X3:Показує час, витрачений на дії.

X4:Відображає розмір програмного коду, який потрібно створити безпосередньо програмісту

4.2.2 Кількісна оцінка параметрів

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію ПП як показано у табл. 4.

Таблиця 4.2 - Основні параметри ПП

Назва параметра	Умовні позначення	Одиниці виміру	Значення параметра		
			гірші	середні	кращі
Швидкодія мови програмування	X1	Оп/мс	19000	11000	2000
Об’єм пам’яті для збереження даних	X2	мб	32	16	8
Час обробки запитів користувача	X3	мс	200	100	50
Потенційний об’єм програмного коду	X4	кількість строк коду	2000	1500	1000

За даними таблиці 4 будуються графічні характеристики параметрів – рис. 5.1 – рис. 5.4.

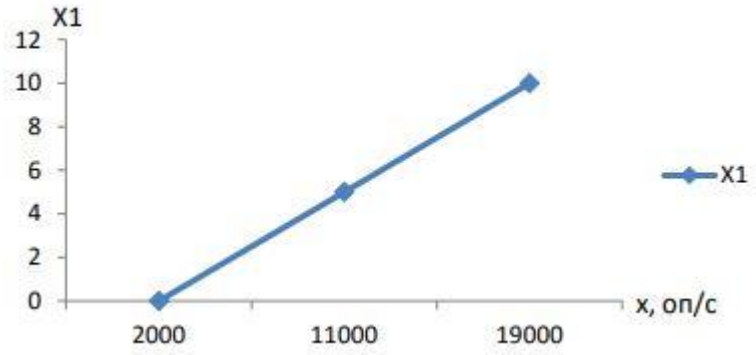


Рисунок 5.1. X1, швидкодія мови програмування.

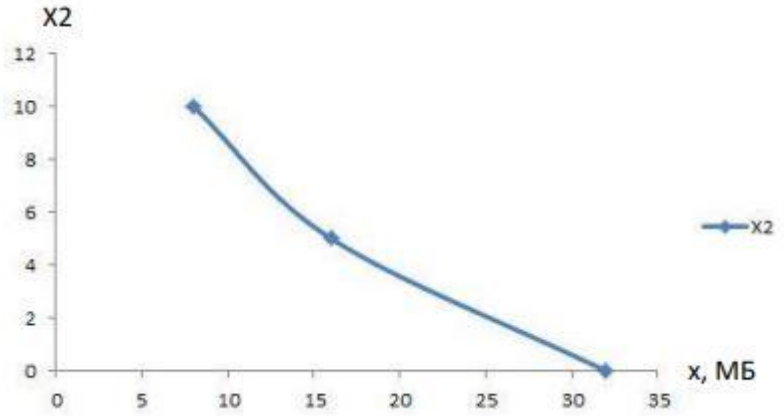


Рисунок 5.2. X2, об’єм пам’яті для збереження даних.

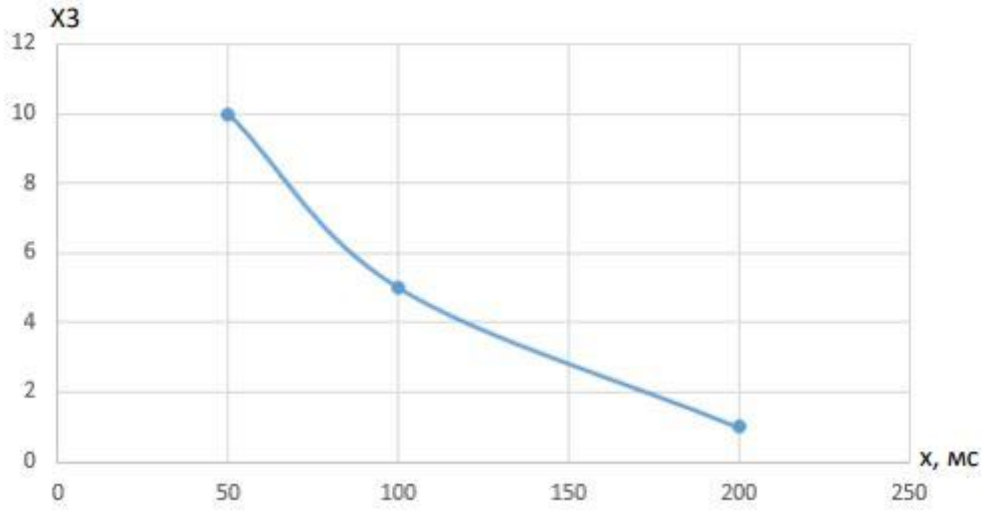


Рисунок 5.3. X3, час обробки запитів користувача.

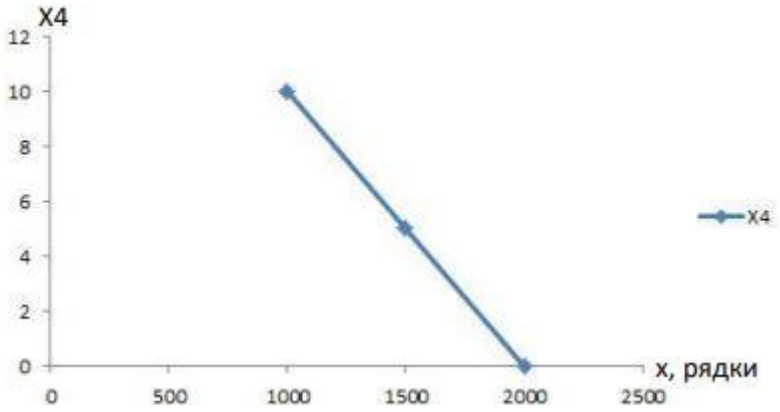


Рисунок 5.4. X4, потенційний об’єм програмного коду.

Після детального обговорення та аналізу кожен експерт оцінює важливість кожного параметра для певної мети – розробки програмного продукту який має найбільш зручний і зрозумілий інтерфейс взаємодія з користувачем.

Значення кожного параметра визначається методом парного порівняння. Оцінку проводить експертна комісія у складі 7 осіб. Визначення коефіцієнтів значущості включає:

- визначення рівнів значущості параметрів шляхом присвоєння різних звання;
- перевірка придатності експертних оцінок для подальшого використання;
- визначення оцінки парного пріоритету параметрів;
- обробка результатів і визначення значущих факторів.

Таблиця 4.3 - Результати ранжування параметрів

Позначення параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів Ri	Відхилення Δi	Δi^2
			1	2	3	4	5	6	7			
X1	Швидкодія мови програмування	Оп/мс	3	2	2	1	1	2	2	13	-4,5	20,25

X2	Об’єм пам’яті для збереження даних	Мб	1	1	1	2	2	1	1	9	-8,5	72,25
X3	Час обробки запитів користувача	Мс	2	3	3	3	3	3	4	21	3,5	12,25
X4	Потенційний об’єм програмного коду	кількість строк коду	4	4	4	4	4	4	3	27	9,5	20,25
	Разом		10	10	10	10	10	10	10	70	0	195

Для перевірки достовірності експертних оцінок, визначимо наступні параметри:

1. сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{J=1}^{N} R_{i_j} = \frac{Nn(n+1)}{2} = 70$$

де N - кількість експертів, n - кількість параметрів;

2. середня сума рангів:

$$T = \frac{1}{1} R_1 j = 17.5$$

3. відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T$$

Сума відхилень по всім параметрам повинна дорівнювати 0;

4. загальна сума квадратів відхилення:

$$s = \sum_{i=1}^{N} \Delta_i^2 = 195$$

Порахуємо коефіцієнт узгодженості:

$$w = \frac{12S}{N^2(n^3 - n)} = \frac{12 * 195}{7^2(4^3 - q)} = 0.8 > w_k = 0.67$$

Рейтинг можна вважати надійним через знайдений коефіцієнт консистенція перевищує нормативну, яка дорівнює 0,67

Використовуючи результати ранжування, проведемо попарне порівняння всіх параметрів і результатів, занесених до таблиці 6.

Таблиця 4.4 - Порівняння параметрів попарно.

Параметри	Експерти							Фінальний рейтинг	Числове значення
	1	2	3	4	5	6	7		
X1 і X2	<	<	<	>	>	<	<	<	0,5
X1 і X3	<	>	>	>	>	>	>	>	1,5
X1 і X4	>	>	>	>	>	>	>	>	1,5
X2 і X3	>	>	>	>	>	>	>	>	1,5
X2 і X4	>	>	>	>	>	>	>	>	1,5
X3 і X4	>	>	>	>	>	>	<	>	1,5

Числове значення, яке визначає ступінь переваги і параметра над j - Це визначається за формулою:

$$a_i = \begin{cases} 1.5 \text{ на } Xi > X \\ 1.0 \text{ при } Xi = Xj \\ 0.5 \text{ при } Xi < Xj \end{cases}$$

З отриманих чисельних оцінок переваг складемо матрицю A = $\| a_{ij} \|$.

Розраховуємо вагу для кожного параметра K_{mu} за такою формулою:

Kві = $bi \sum bi\ n\ i$=1 ,де $bi = \sum aij\ N\ i$=1 .

Відносні оцінки обчислюються кілька разів, поки вони є. значення не будуть незначно відрізнятися від попередніх (менше 2%). на другому та наступних кроках відносні оцінки розраховуються відповідно до наступна формула:

Kві = $bi\ ' \sum bi\ n\ '\ i$=1 ,де $bi\ ' = \sum aijbj\ N\ i$=1 .

Як видно з таблиці 7, різниця між значеннями ваги не перевищує 2%, тому більше ітерацій не потрібно

Таблиця 4.5 - Розрахунок вагомості параметрів

Параметри xi	Параметри xj				Перша ітер.		Друга ітер.		Третя ітер.	
	X1	X2	X3	X4	bi	Кві	bi 1	Kві 1	bi 2	Kві 2
X1	1,0	0,5	1,5	1,5	4,5	0,281	16,25	0,275	59,125	0,274
X2	1,5	1,0	1,5	1,5	5,5	0,344	21,25	0,36	77,875	0,361
X3	0,5	0,5	1,0	1,5	3,5	0,219	12,25	0,208	44,875	0,207
X4	0,5	0,5	0,5	1,0	2,5	0,156	9,25	0,157	34,125	0,158
Всього					16	1	59	1	216	1

4.3 Аналіз рівня якості варіантів реалізації функцій

Визначаємо рівень якості кожного варіанту основних функцій окремо.

Абсолютні значення параметрів Н2 (об'єм пам'яті для зберігання даних) і Ks1 (швидкість мови програмування) відповідають технічним вимогам умов роботи цього пп.

Абсолютне значення параметра Н3 (потенційна кількість програмного коду) найгірший (а не максимальний) не вибирається, тобто значення відповідає або варіант а) 2000 або варіант б) 1500

Абсолютне значення параметра Н4 (час обробки запиту користувача) найкращий (не мінімальний) не вибирається, тобто це значення відповідає або варіант а) 50 мс або варіант б) 100 мс

Коефіцієнт технічного рівня для кожного варіанту реалізації ПП розраховується таким чином (табл. 8):

$$K\text{к}(j) = \sum K\text{в}i,jBi,j \; n \; i\text{=}1 \; ,$$

де n – кількість параметрів;$K\text{в}i$– коефіцієнт вагомості і–го параметра; Bi – оцінка і–го параметра в балах.

Таблиця 4.6 - Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Параметри, що беруть участь у реалізації функцій	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1	Б	X1	100	5	0,274	1,37
F2	А	X2	500	6.7	0,361	2,42
F3	А	X3	1500	2.5	0,158	0,395
	Б		800	6.4	0,158	1,01
	А	X4	50	7.5	0,207	1,55
	Б		100	5.5	0,207	1,14

За даними з таблиці 8 за формуло

$$Кк = Кту[F1k\ ] + Кту[F2k\ ] + \ldots + Кту[Fzk],$$

визначаємо рівень якості кожного з варіантів:

Кк1 = 1,37 + 2,42 + 1, 55 + 0,395 = 5,725

Кк2 = 1,37 + 2,42 + 1,14 + 1,01 = 5,95

Як видно з прикладу, краще другий варіант, для якого коефіцієнт технічний рівень має першорядне значення.

4.4 Економічний аналіз варіантів розробки ПП

Щоб визначити вартість розробки програмного забезпечення, спочатку розраховуємо інтенсивність роботи.

Всі варіанти включають в себе два різних завдання:

1. Розробка проекту програмного продукту;
2. Розробка програмної оболонки;

Завдання 1 за ступенем новизни відноситься до групи А, завдання 2 – до група В. Необхідна складність алгоритмів, використаних у Завданні 1 в групі 1; а в завданні 2 - до 3 груп.

Довідкова інформація і використовується для виконання завдання 1 Завдання 2 використовує інформацію у вигляді даних

Розрахуємо норми часу на розробку та програмування кожних із завдань.

Розрахуємо норми часу на розробку та програмування кожного із завдань. Загальна складність обчислюється як

То = Тр· Кп· Кск· Км· Кст· Кст.м,

- де Тр – трудомісткість розробки ПП;
- Кп – поправочний коефіцієнт;
- Кск – коефіцієнт на складність вхідної інформації;
- Км – коефіцієнт рівня мови програмування;
- Кст – коефіцієнт використання стандартних модулів і прикладних програм;

- Кст.м – коефіцієнт стандартного математичного забезпечення.

Для першого завдання, виходячи з норм часу обчислювальних завдань характер ступеня новизни А та групи складності алгоритму 1, складність 62 дорівнює: Tr = 90 людино-днів. Поправочний коефіцієнт, що враховує вид нормативно-довідкова інформація для першого завдання: КП = 1,7. Коригувальний коефіцієнт, що враховує складність управління вхідною та вихідною інформацією для всіх напівзавдань дорівнює 1: Кск = 1. З моменту розробки першого завдання використовуються стандартні модулі, це враховується коефіцієнтом Kst = 0,8. Тоді, згідно з формулою 5.1, на першому місці загальна складність програмування завдання те саме:

$T_1 = 90 \cdot 1.7 \cdot 0.8 = 122.4$ людино-днів.

Подібні розрахунки зробимо для подальших завдань.

Для другого завдання (використовується алгоритм третьої групи складність, рівень новизни Б), тобто $T_р = 27$ людино-днів, Кп = 0,9, $К_{ск} = 1$, $К_{ст} = 0,8$:

$T_2 = 27 \cdot 0.9 \cdot 0.8 = 19.44$ людино-днів.

Додамо складність відповідних завдань для кожного з вибраних варіанти реалізації програм для отримання їх складності:

$T_I = (122.4 + 19.44 + 4.8 + 19.44) \cdot 8 = 1328,64$ людино-годин;

$T_{II} = (122.4 + 19.44 + 6.91 + 19.44) \cdot 8 = 1345.52$ людино-годин;

Найбільш високу трудомісткість має варіант II.

Один забудовник із зарплатою 6 тис. грн та один задіяний у розробці фінансовий аналітик із заробітною платою 9000 грн. Визначити заробітну плату за рік за формулою:

$С_ч = М / T_m \cdot t$ грн.,

de M - місячна заробітна плата працівників; Tm - кількість робочих днів на тиждень; t - кількість робочих годин на добу.

$С_ч = 6000 + 9000 / 2 \cdot 21 \cdot 8 = 44,64$ грн.

Тоді, розрахуємо заробітну плату за формулою

$С_{ЗП} = С_ч \cdot T_i \cdot К_Д$

де $С_ч$– величина погодинної оплати праці програміста; $Т_і$ – трудомісткість відповідного завдання; $К_Д$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

1. $С_{ЗП} = 44,64 \cdot 1328.64 \cdot 1.2 = 71172,59$ грн.
2. $С_{ЗП} = 44,64 \cdot 1345.52 \cdot 1.2 = 72076,82$ грн.

Відрахування на єдиний соціальний внесок залежно від групи професійний ризик (II клас) становить 22%:

1. $С_{ВІД} = С_{ЗП} \cdot 0.22 = 71172,59 \cdot 0.22 = 24429$ грн
2. $С_{ВІД} = С_{ЗП} \cdot 0.3677 = 67281,38 \cdot 0.22 = 24739$ грн.

Тепер вирішуємо заплатити за один машинний рік. (SM)

Один розробник із заробітною платою 6000 грн. виконує роль однієї МНВ, с при коефіцієнті зайнятості 0,2, то на один автомобіль отримуємо:

$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 6000 \cdot 0,2 = 16800$ грн.

З урахуванням додаткової заробітної плати:

$С_{ЗП} = С_Г \cdot (1 + К_З) = 16800 \cdot (1 + 0.2) = 20280$ грн.

Відрахування на єдиний соціальний внесок:

$С_{ВІД} = С_{ЗП} \cdot 0.22 = 20280 \cdot 0,22 = 4461,6$ грн.

Амортизаційні відрахування нараховуються з амортизацією 25% і ціна ЕОМ 8000 грн.

$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.15 \cdot 0.25 \cdot 8000 = 2300$ грн.,

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача; $К_А$– річна норма амортизації; $Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

СР = КТМ·ЦПР · КР = 1.15 · 8000 · 0.05 = 460 грн.,

де КР– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 104 – 8 – 16) \cdot 8 \cdot 0.9 = 1706.4$ годин, де $Д_К$ – календарна кількість днів у році; ДВ, ДС – відповідно кількість вихідних та святкових днів; $Д_Р$ – кількість днів планових ремонтів устаткування;

t –кількість робочих годин в день; КВ– коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot К_З \cdot Ц_{ЕН}$ =1706,4 · 0,156 · 0,9733 · 2,0218 = 523.83 грн.,

де $N_С$ – середньо-споживча потужність приладу; $К_З$– коефіцієнтом зайнятості приладу; $Ц_{ЕН}$ – тариф за 1 $КВ_т$-годин електроенергії.

Накладні витрати розраховуємо за формулою:

СН = ЦПР·0.67 = 8000· 0,67 =5360 грн.

Тоді, річні експлуатаційні витрати будуть:

$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н$

$С_{ЕКС}$ = 17280 + 6353,86 + 2300 + 460 + 523.83 + 5360 = 33883,55 грн.

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$С_{М-Г} = С_{ЕКС} / Т_{ЕФ}$ = 33883,55/1706,4 = 20,68 грн/час.

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$С_М = С_{М-Г} \cdot Т$

1. $С_М$ = 20,68· 1328,64 = 27476 грн.;
2. $С_М$ = 20,68· 1345.52 = 27825 грн.;

Накладні витрати складають 67% від заробітної плати:

$С_Н = С_{ЗП} \cdot 0,67$

1. $С_Н$ = 71172,59 · 0,67 = 47685,63 грн.;
2. $С_Н$ = 72076,82 · 0,67 = 48291,47 грн.;

Отже, вартість розробки ПП за варіантами становить:

$С_{ПП} = С_{ЗП} + С_{ВІД} + С_М + С_Н$

1. $С_{ПП}$ = 71172,59 + 4461,6 + 27476 + 47685,63 = 150795,82 грн.;
2. $С_{ПП}$ = 72076,82 + 4461,6 + 27825 + 48291,47 = 152654,89 грн.;

4.5 Вибір кращого варіанта ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$К_{ТЕРj} = К_{Кj} / С_{Фj}$,

$К_{ТЕР1} = 5,325 / 150795,82 = 0,35 \cdot 10^{-4}$;

$К_{ТЕР2} = 6,35 / 152654,89 = 0,42 \cdot 10^{-4}$;

Як бачимо, найбільш ефективним є другий варіант реалізації програми з коефіцієнтом техніко-економічного рівня $К_{ТЕР1} = 0,42 \cdot 10^{-4}$.

Висновок до розділу 4

У цьому розділі представлено повний функціональний аналіз та аналіз вартості ПП, який розроблено в рамках дипломного проекту. Можливий процес аналізу умовно розділений на дві частини.

Першим з них було вивчення ПП з технічної точки зору: визначаються основні функції ПП та формується набір варіантів їх реалізації; на на основі розрахункових значень параметрів, а також експертних оцінок їх значущості розраховано коефіцієнт технічного рівня, який уможливив визначення оптимальна з технічної точки зору, альтернатива реалізації функцій ПП.

Друга частина FVA присвячена вибору альтернатив здійснення найбільш економічно виправданого. Порівняння запропонованого варіанти реалізації в рамках цієї частини виконуються на коефіцієнті ККД, для розрахунку якого розраховуються наступні допоміжні параметри, як довгострокові, вилучити зарплати, видалити рахунки.

Після виконання функціонально-вартісного аналізу програми складний у розвитку, можна зробити висновок, що з альтернатив, що залишився після першого вибору двох версій програмиперша версія програмного продукту є оптимальною. IN це виявилося найкращим показником техніко-економічного рівня якості $К_{ТЕР} = 0,42 \cdot 10^{-4}$

Цей варіант реалізації програмного продукту має такі параметри:

- Серверна технологія ASP.NET;
- Клієнтський фреймворк Angular 2;
- Клієнтський менеджер модулів SystemJS.

Обраний варіант є оптимальний й найкраще відповідає вимогам поставленим в ході цієї роботи.

ВИСНОВКИ

В ході даної магістерської роботи було досліджено методи розробки веб-проектів, порівняльний аналіз існуючих рішень розробки. Досліджено накращі методи розробки на сьогоднішній день, та проаналізовано, який метод краще підходить для вирішення саме потрібного завдання, для зменшення витрачених ресурсів на розробку.

Наведено приклад розробки за домогою фреймворка Angular 2, який наразі є популярним методом у більшості компаній світу, наведено приклад, найефективнішого використання цього методу розробки для великих проектів, також, описано основні переваги використання саме цього методу розробки для складних проектів.

Зроблена оцінка алгоритма методів розробки, ефективності та витрачених ресурсів для виконання поставлених завдань, розробки веб-проектів, зроблений функціональний аналіз та аналіз вартості розробки за допомгою різних методів розробки. Визначаються основні функції та формується набір варіантів їх реалізації, на основі розрахункових значень параметрів, а також експертних оцінок їх значущості розраховано коефіцієнт технічного рівня, який уможливив визначення оптимальна з технічної точки зору, альтернатива реалізації функцій.

Якщо зробити висновки, з’являється розуміння для чого створені ці методи розробки та коли потрібно їх використовувати для найкращого результату в еквіваленті витрачених ресурсів, що є найголовнішим для розробників.

ПЕРЕЛІК ПОСИЛАНЬ

1. React Documentation [Електронний ресурс] : [інтернет портал]. – Електронні дані. – React from Facebook Guide. – Режим доступу: https://reactjs.org/docs/getting-started.html
2. Angular Documentation [Електронний ресурс] : [інтернет портал]. – Електронні дані. – Angular Get Statred. – Режим доступу: https://angular.io/start
3. Vue.js Documentation [Електронний ресурс] : [інтернет портал]. – Електронні дані. – Vue.js for begginers. – Режим доступу: https://vuejs.org/v2/guide/
4. Односторінкові додатки [Електронний ресурс] : [інтернет портал]. – Електронні дані. – Create Single Page Application from ASP.NET. – Режим доступу: http://www.interface/home.asp?artId=35504/
5. Що таке SPA або односторінковий портал. [Електронний ресурс] : [інтернет портал]. – Електронні дані. – Create Single Page Application – Режим доступу: https://www.calabonga.net/site
6. Офіційний сайт Knockout. [Електронний ресурс] : [інтернет портал]. – Електронні дані. – Knockout Guide. – Режим доступу: https://knockoutjs.com/
7. Офіційний сайт TypeScript. [Електронний ресурс] : [інтернет портал]. – Електронні дані. – TypeScript Guide. – Режим доступу: https://www.typescriptlang.org/
8. Williamson K., Learning AngularJS. // Williamson K. - O'Reilly Media, 2015 – 212 с.
9. Gechev M., Switching to Angular 2 // Gechev M. - Packt Publishing, 2016 – 254 с.
10. Herrington J., Learning AngularJS. // Herrington J. - Packt Publishing, 2015 – 235 с.
11. Black C., Building a Single Page Web Application with Knockout.js // Black C., Ly D. - Packt Publishing, 2014 – 152 с.
12. Monteiro F., Learning Single-page Web Application Developmen // Monteiro F. - Packt Publishing, 2014 – 214 с.

13. Козловский П., Разработка веб-приложений с использованием AngularJS // Козловский П., Дарвин П. - ДМК Пресс, 2014 – 394 с.
14. Миковски М., Разработка одностраничных веб-приложений // Миковски М. - ДМК Пресс, 2014 – с. 512
15. Knol A., Dependency Injection with AngularJS // Knol A. - Packt Publishing, 2015 – с. 78

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

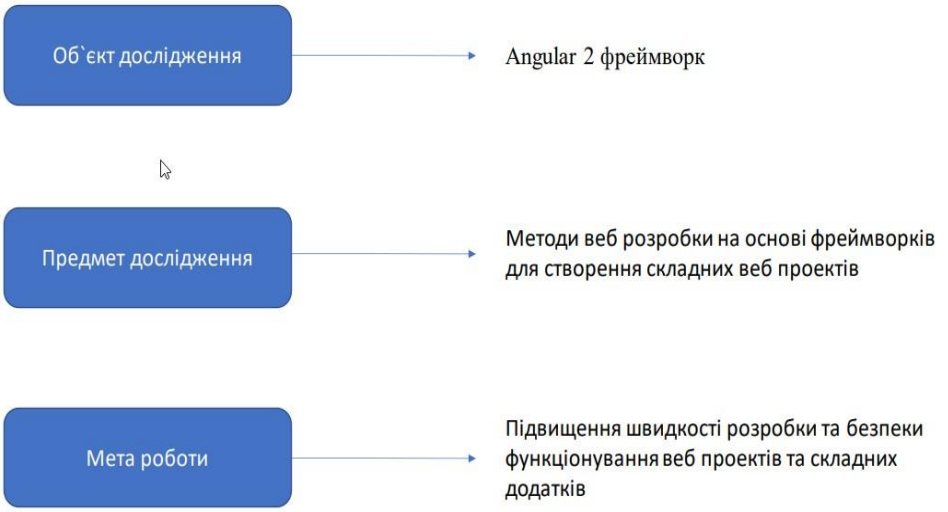
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра інженерії програмного забезпечення

МАГІСТЕРСЬКА РОБОТА
«ОПТИМІЗАЦІЯ РОЗРОБКИ WEB-ПРОЕКТІВ ТА СКЛАДНИХ ДОДАТКІВ ЗА ДОПОМГОЮ ФРЕЙМВОРКІВ»

Виконав: студент групи ПДМ-61,
Стоян М. О
Керівник: кандидат технічних наук, доцент
Негоденко О.В



Актуальність роботи

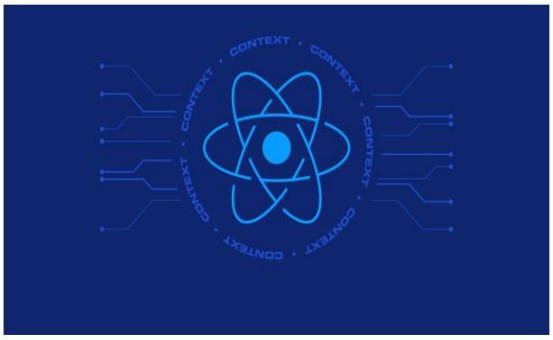
Дослідження полягає у розробленні теоретико-методичних та практичних рекомендацій, які дозволяють розробити архітектурне рішення для розробки веб-проектів за допомогою накращих фрейморків. На сьогоднішній день майже всі компанії світу використовують фреймворки в своїх проектах для зручності роботи в команді та швидкості ствоерння додатків.

ЗАВДАННЯ

- Провести порівняльний аналіз ефективності існуючих методів розробки
- Розробити проект за допомогою найкращого фреймворку
- Оцінити ефективність алгоритму за допомогою аналіза рівня якості варіантів реалізації

Фрейморки для проектів веб-розробки

ReactJS



JavaScript-бібліотека для створення користувацьких інтерфейсів

Переваги

Декларативний
React спрощує створення інтерактивних інтерфейсів. Вам потрібно лише описати, як різні частини інтерфейсу виглядають у кожному стані вашого додатку і React ефективно оновить та відрендерить лише потрібні компоненти, коли ваші дані зміняться.
Декларативні інтерфейси роблять ваш код більш передбачуваним і його набагато легше налагоджувати.

Заснований на компонентах
Створюйте інкапсульовані компоненти, які керують власним станом, а з них будуйте складні інтерфейси.
Оскільки логіка компонентів написана на JavaScript, замість шаблонів, ви з легкістю можете передавати складні дані у вашому додатку і зберігати стан окремо від DOM.

Фрейморки для проектів веб-розробки

AngularJs

Сучасна платформа веб-розробника Angular

Переваги

РОЗРОБКА НА ВСІХ ПЛАТФОРМАХ
Дізнайтеся, як створювати програми за допомогою Angular і повторно використовувати свій код і можливості для створення програм для будь-якої цілі розгортання. Для Інтернету, мобільного Інтернету, рідного мобільного та рідного комп'ютера.

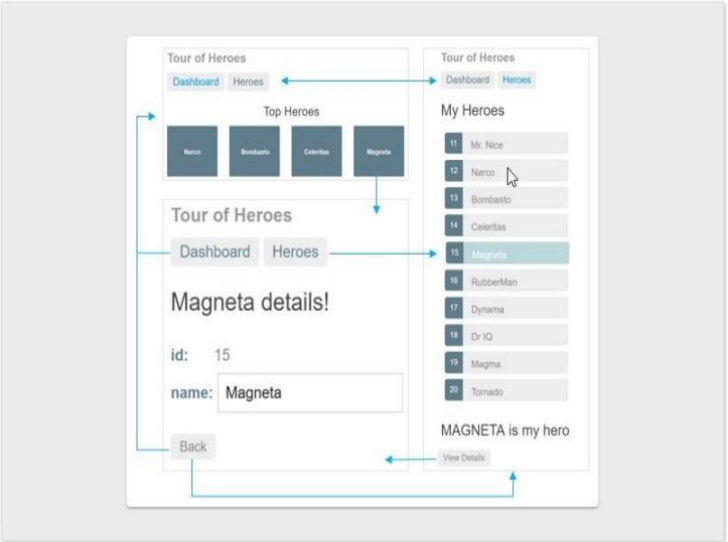
ШВИДКІСТЬ І ПРОДУКТИВНІСТЬ
Досягніть максимально можливої швидкості на веб-платформі сьогодні та розвивайте її далі за допомогою Web Workers та рендерингу на стороні сервера.
Angular дає вам контроль над масштабованістю. Задовольняйте величезні вимоги до даних, створюючи моделі даних на RxJS, Immutable.js або іншій push-моделі.

Порівняння самих популярних фреймворків

Angular			ReactJS
Google	Developer	Facebook	
2016	Released	2013	
Framework	Type	Frontend-library	
High	Learning curve	Medium	
HTML + TypeScript	Template	JSX + JS (ES5/ES6)	
Bi-directional (2-way)	Data-binding	Uni-directional (1-way)	
Regular DOM	DOM	Virtual DOM	
Strong	Abstraction	Medium	

Розробити проект за допомогою найкращого фреймворку

Розроблено проект в якому використані основні переваги фреймворку для ефективного використання додатку.



Директиви

Односторонній підхід. Angular був перший у світі, який повністю усунув необхідність перезавантажевання сторінки.

Компонентний підхід

Підвищення швидкості розробки та безпеки функціонування веб проектів та складних додатків

За допомогою фреймворків з'являється можливість розробляти веб проекти набагато швидше та зручніше, якщо розібратися.

Для вдалого використання фреймворків потрібно розуміти, який саме підходить для розробки

- В основі є прості мови програмування.
- Надзвичайна гнучкість програми.
- Використання DOM.
- Програма витримує великі навантаження.
- Фрейморки добре ладять з СЕО. Пошуковим роботам простіше переглядати сайти, покращується взаємодія користувачів з Вашим ресурсом.
- Забезпечує незмінність батьківських даних.
- Має відкриту бібліотеку даних.
- Невелика вага бази даних
- Забезпечує просту міграцію між версіями.
- Гібридні мобільні додатки зовні майже не відрізняються від нативних.

Висновки

В ході магістерської роботи було досліджено методи розробки веб-проектів, порівняльний аналіз існуючих рішень розробки. Досліджено найкращі методи розробки на сьогоднішній день, та проаналізовано, який метод краще для вирішення саме потрібного завдання, для зменшення витрачених ресурсів на розробку

Дякую за увагу!