МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
Факультет кібербезпеки, комп'ютерної та програмної інженерії (ЗФН)
Кафедра комп'ютерних інформаційних технологій

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
_____ Аліна САВЧЕНКО
"____"_____2021 р.

ДИПЛОМНА РОБОТА

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ

"МАГІСТРА"

ЗА ОСВІТНЬО-ПРОФЕСІЙНОЮ ПРОГРАМОЮ "ІНФОРМАЦІЙНІ УПРАВЛЯЮЧІ СИСТЕМИ ТА ТЕХНОЛОГІЇ"

Тема: "WEB-додаток на основі React для управління робочим процесом підприємства"

Виконавець: Гавлицький Олександр Васильович

Керівник: к.т.н., доцент Колісник Олена Василівна

Нормоконтролер: _____ Ігор РАЙЧЕВ

Київ - 2021

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, комп'ютерної та програмної інженерії (ЗФН)

Кафедра Комп'ютерних інформаційних технологій

Галузь знань, спеціальність, освітньо-професійна програма: 12 "Інформаційні технології", 122 "Комп'ютерні науки", "Інформаційні управляючі системи та технології" .

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Аліна САВЧЕНКО

«____»_____2021р.

ЗАВДАННЯ

на виконання дипломної роботи студента

Гавлицького Олександра Васильовича

(прізвище, ім'я, по батькові)

1. Тема роботи: «WEB-додаток на основі React для управління робочим процесом підприємства» затверджена наказом . ректора від 12.10.2021 за № 2229/ст.

2. Термін виконання роботи: з 12.10.2021 по 31.12.2021.

3. Вихідні дані до роботи: . Операційна система сімейства Windows XP та новіше, Visual Studio Code, встановлений фреймворк React JS, MongoDB.

4. Зміст пояснювальної записки: вступ, огляд теоретичної бази побудови веб-додатків, функціональна модель програмного комплексу, проектування рішень розробки, опис технологій розробки, тестування додатку.

5. Перелік обов'язкового ілюстративного матеріалу: слайди, презентація.

6. Календарний план-графік

№ п/п	Завдання	Термін виконання	Підпис керівника
1.	Проаналізувати літературу та джерела за темою дипломного проекту.	12.10.2021 – 15.10.2021	
2.	Розроблення та затвердження плану дипломного проекту.	16.10.2021 – 19.10.2021	
3.	Огляд та створення концепції репозитарію патернів альтернативних архітектур ПС.	20.10.2021 – 24.10.2021	
4.	Провести консультації з науковим керівником щодо створення першого розділу.	25.10.2021 – 31.10.2021	
5.	Написання Розділу 1 та 2 дипломної роботи.	01.11.2021 – 07.11.2021	
6.	Розробка додатку	08.11.2021 – 17.11.2021	
7.	Написання Розділу 3 та 4 дипломної роботи. Завершення створення пояснювальної записки.	18.11.2021 – 01.12.2021	
8.	Оформлення та друк пояснювальної записки.	02.12.2021 – 11.12.2021	
9.	Створення презентації, доповіді та підготовка до захисту дипломної роботи	12.12.2021 – 20.12.2021	

7. Дата видачі завдання: 12.10.2021р.

Керівник дипломної роботи _____ Колісник Олена

(підпис керівника)

Завдання прийняв до виконання _____Гавлицький Олександр.

РЕФЕРАТ

Пояснювальна записка до дипломної роботи роботи «WEB-додаток на основі React для управління робочим процесом підприємства»» викладена на 93 сторінках, містить 43 рисунків, 1 таблиця та 10 літературних джерел.

Ключові слова: WEB-ДОДАТОК, РОЗРОБКА, СИСТЕМА, РІШЕННЯ, АРХІТЕКТУРА.

Об'єкт дослідження: процес створення та ведення профілю користувача, процес створення задач, процес комунікації.

Предмет дослідження: web-додаток для управлыння робочим процесом підприємства.

Мета роботи: створити Web-додаток, здатний конструювати опитування зі різноманітними типами питань та з подальшою можливістю розсилання.

Методи дослідження: Розробка інтерактивного додатку який допоможе в управлінні робочим процесом на підприємстві.

Отримані результати: реалізовано і введено в експлуатацію Web-додаток, який дозволяє зручно керувати робочим процесом компанії.

Результати дипломної роботи планується використовувати для подальшого розвитку, додання та розширеня функціонал

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД – база даних

ОС – операційна система

ПЗ – програмне забезпечення

ПК – персональний комп’ютер

ООП – об’єктно-орієнтоване програмування

HTML – HyperText Markup Language

CSS – Cascade Style Sheets

HTTP – HyperText Transfer Protocol

JS – JavaScript

HTTP – протокол передачі даних

ERP - enterprise resource planning

CRM - Customer Relationship Management

URL - Uniform Resource Locator

DNS - Domain Name System

IP - Internet Protocol

TCP - Transmission Control Protocol

IDE – Інтегроване середовище розробки (Integrated Development Environment)

ВСТУП

Дистанційна робота в останні роки набирає популярності, і багато компаній по всьому світу обрали саме цей формат. Є три основні переваги розширення середовища пошуку персоналу, ефективності, балансу життя та роботи та лояльності команди. Щоб визначити найкращий підхід до організації, потрібно орієнтуватися на потреби керівників і співробітників, приймати чіткі рішення та оптимізувати, встановлювати терміни і гарантувати, що всі співробітники мають рівний доступ до проекту, ставити цілі та мінімізувати час для досягнення цілей. Вони. Тому останнім часом онлайн-сервіси стали такими популярними.

Онлайн-сервіси дозволяють автоматизувати процес компанії, це виводить якість і швидкість розробки проекту на абсолютно новий рівень. Управління та моніторинг завдань у режимі реального часу дозволяє швидко та вчасно вносити корективи, досягаючи процесу впровадження та цілей. Слідкуйте за часом, ресурсами та витратами на персонал для виконання конкретних завдань. Відстежуйте, аналізуйте та покращуйте робочу силу на кожному етапі розвитку

Актуальність обраної для дослідження теми полягає в тому, що сотні компаній щодня переходять у віддалений режим, а це означає, що процес можна виконувати, не виходячи з дому. Організація роботи залишається основним завданням, тому на допомогу приходять додатки. За допомогою спеціального програмного забезпечення можна легко керувати базою даних персоналу, швидко створювати та розподіляти робочі місця, контролювати процес впровадження, здійснювати ділові дзвінки та розмови тощо..

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Поняття «Web-додаток» та «Web-сайт»

Веб-додаток — це будь-яка комп'ютерна програма, яка виконує певну функцію за допомогою веб-браузера як клієнта, незалежно від пристрою чи платформи, на якій відкрито браузер. Програма може бути простою, як дошка оголошень або контактна форма, текстовий процесор або програма для мобільних ігор для кількох гравців, яку ви завантажуєте на свій телефон. [1]

Веб-сайт – це група загальнодоступних, взаємопов'язаних веб-сторінок, які мають спільне доменне ім'я. Він може бути розроблений окремою особою, бізнесом або організацією. Веб-сайт призначений для різних цілей. Прикладом є сайти-візитки, блоги, новинні портали тощо. Основні функції сайту:

1) Ефективний спосіб демонструвати свої продукти та послуги.
2) Розробка сайту допоможе створити соціальний доказ.
3) Допомагає у брендінгу бізнесу.
4) Допомагає досягти ділових цілей.
5) Дозволяє розширити підтримку клієнтів

1.2. Основні відмінності web- додатку від web-сайту

1.2.1. Інтерактивність

Перша відмінність – це різні рівні взаємодії зі сторінкою. Хоча веб-сайт містить текстовий і візуальний вміст, до якого користувач не може отримати доступ, веб-додатки дозволяють користувачеві не тільки читати, але й змінювати та додавати інформацію на сторінки.

Його можна назвати інтернет-магазином, веб-додатком, що дозволяє користувачам купувати товари, здійснювати пошук у каталогах. Ще один цікавий приклад – соціальні мережі. Сюди входять функції блогу, чати, вміст, який вибирають користувачі, і можливість ділитися цим вмістом.

Сьогодні велика кількість веб-сайтів є інтерактивними. Тому що користувачеві це подобається. Для цього власники веб-сайтів додають на свої веб-сайти невеликі веб-додатки.

Деякі ресторани мають доступ до Google Maps, щоб допомогти користувачам знайти дорогу до ресторану. У той же час більшість сайтів є більш інформативними, ніж інтерактивними. Це робить відвідувачів сайту більш зацікавленими у перегляді, читанні або прослуховуванні інформації. А відвідувачі веб-додатків зосереджуються на взаємодії з користувачами.

1.2.2. Інтеграція

Інтеграція – це процес об’єднання простих компонентів в один комплекс. Розробники можуть інтегрувати веб-додатки та сайти з програмами, включаючи ERP, CRM. Однак у більшості випадків інтеграція відбувається з веб-додатками,

оскільки їх складні функції часто вимагають інформації з інших систем. Найпопулярніший вид інтеграції в електронній комерції інтегрує веб-додатки з системою керування взаємодією з клієнтами (CRM). Це допомагає зберігати дані клієнтів, інформацію про замовлення та покращувати загальні продажі. Завдяки інтеграції інформація про користувача автоматично збирається та зберігається в CRM-системі. Така інтеграція дозволяє відділу продажів дізнатися більше про поведінку клієнтів і ефективніше працювати з негативними відгуками. Уміння спілкуватися з інформацією про клієнтів може збільшити продажі та покращити процес роботи в інтернет-магазині.

У деяких випадках власники веб-сайтів використовують інтеграцію CRM, щоб надати відвідувачам більш персоналізований вміст. Але на відміну від веб-програм, така інтеграція з веб-сторінками є додатковою функцією, а не невід'ємною частиною основної функції.

1.2.3. Авторизація

Цей процес передбачає введення даних користувача для доступу до веб-сайтів або систем. Ця функція є важливою для систем, які потребують персональної інформації клієнта. На цьому етапі важливо звернути особливу увагу на безпеку. Важливо мінімізувати доступ сторонніх осіб до персональних даних користувачів.

На відміну від веб-сайтів, веб-додатки пропонують користувачам більше функцій, ніж веб-сторінки, тому їм знадобиться процес авторизації. Наприклад, при використанні соціальних мереж системи можуть попереджати про вразливості паролів. Ігнорування таких повідомлень дозволить хакерам увійти у ваш обліковий запис.

Більшість сайтів використовують авторизацію. У деяких випадках вона може використовуватися для надання додаткових функцій, які недоступні для неавторизованого користувача. Якщо не зареєстровані користувачі можуть переглядати лише статті, зареєстровані користувачі залишають коментарі, ділитися статтями в соціальних мережах тощо. Це гарне рішення для блокування спаму. Тому дозвіл потрібен як для веб-сайтів, так і для веб-програм. Але в той же час веб-програми потребують ліцензії для захисту даних.

У сучасному світі Інтернету практично не існує соціальних мереж. Навпаки, багато веб-додатків часто передбачають доступ до інформації. Однак веб-сайти все ще залишаються джерелом інформації, а веб-додатки залишаються користувацьким інструментом.

Можна сказати, що всі веб-додатки є веб-сторінками, але не всі веб-додатки є веб-додатками. Унікальним його є те, що неможливо розробити веб-додаток з нуля. У будь-якому випадку вам потрібно використовувати не тільки поширені мови розмітки та програмування, такі як HTML, CSS і JavaScript, а й технологічні бібліотеки від сторонніх розробників. Існує ряд компаній, які спеціалізуються на створенні подібних технологій для початківців і малого бізнесу.

1.3. Переваги та недоліки Web-додатку

1.3.1. Переваги створення Web-додатків

Веб-додатки розробляються на таких мовах програмування, як HTML і CSS, які користуються популярністю серед IT-фахівців.

На відміну від рідних програм, веб-додатки можна використовувати на всіх пристроях. Він запрограмований для роботи на будь-якій операційній системі. Повинен бути сумісний з iOS, Android, Windows Phone та іншими системами.

Веб-додатки не вимогливі до ресурсів і не пред’являють ніяких вимог до апаратної платформи.

Немає проблем із підтримкою та сумісністю для старих версій програм. Коли надходить нова версія програми, користувачам часто доводиться стикатися з проблемами оновлення встановленої версії на своїх пристроях. У додатку браузера таких проблем не виникає - є лише одна версія, в якій працюють усі користувачі, і якщо вона нова, то всі без винятку ходять до неї, іноді навіть не знаючи про це.

На пристрої не потрібно встановлювати веб-програму.

Ці програми запускаються у веб-браузері пристрою через просту URL-адресу.

Їх не потрібно завантажувати та встановлювати з магазинів додатків таких як Google Play або Apple Store. Це економить гроші, оскільки пряме посилання через веб-додаток безкоштовне.

Ви також можете відкривати веб-сайти. Це означає, що їх не потрібно оновлювати, як це роблять стандартні програми.

Високий рівень розвитку мережевих з'єднань і надійність web-технологій.

Розробка веб-додатків є дешевим видом розробки додатків. Це передбачає створення посилання або кількох посилань між програмою та URL-адресою. Створення рідної або перекладеної програми коштує великих грошей, але шанси на успіх програми дуже високі.

Веб-додатки дозволяють своїм користувачам бути по-справжньому мобільними. Ви можете зберігати результати своєї роботи на сервері, а при необхідності матимете доступ до них з будь-якої точки планети з доступом до Інтернету.

1.3.2. Недоліки створення Web-додатків

Як згадувалося вище, веб-додаток можна використовувати на всіх пристроях. Але, звичайно, веб-сайт має бути запрограмований на відображення незалежно від операційної системи пристрою. Якщо це не адаптивний веб-сайт, у вас можуть виникнути проблеми з його відображенням на iOS, Android або Windows Phone.

Необхідно буде запустити підключення до Інтернету. Інакше ви не зможете переглядати веб-сайт, а веб-додаток не принесе вам користі. На жаль, наприклад, у нашій країні Інтернет доступний не скрізь. Це ускладнює роботу з веб-додатками, наприклад, під час подорожі.

Кожне перезавантаження (або оновлення сторінки) призведе до значних затримок, оскільки вам потрібно буде встановити з’єднання HTTP, обробити запит на сервері, надіслати HTTP-повідомлення до мережі та перезавантажити сторінку в браузері. Це створює режим переривання і сповільнюється.

Крім того, існують деякі обмеження доступу до певних апаратних функцій, на яких працює пристрій.

Існує багато програм, які неможливо реалізувати в Інтернеті, наприклад, можливість створювати складні 3D-моделі в браузері.

1.4. Архітектура та принципи роботи типового web-додатку.

Web-додаток являється клієнт-серверним, в я кому клієнтом виступає браузер, а сервером web-сервер. Логіка web-додатку розподілена між сервером та клієнтом. Збереження даних переважно здійснюється на сервері. Обмін інформації проходить у мережі Інтерент.

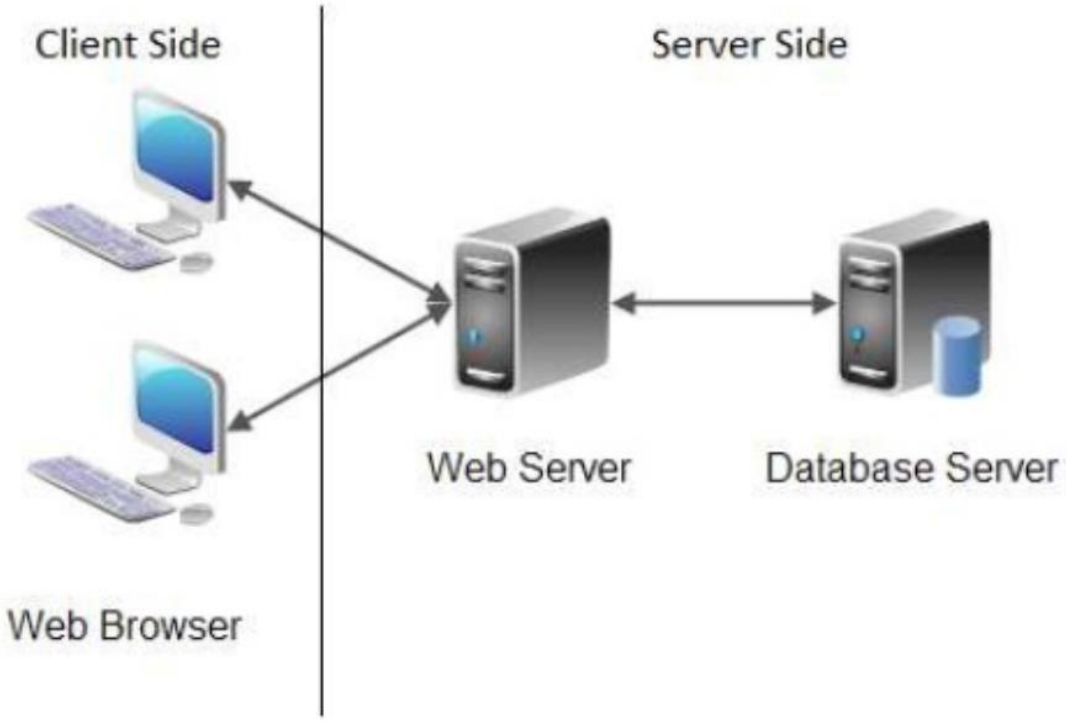

Рис. 1.1. Діаграма типового web-додатку

Є дві програми, які одночасно працюють у web-додатках:

- код на сервері і відповідає на HTTP-запити;
- код у браузері, який реагує на введення користувача.

Код сервера не видимий для клієнта, він може відповідати лише на HTTP-запити з певною URL-адресою, а не на будь-який вхід користувача. Цей код зазвичай пишеться на таких мовах програмування та фреймворках: Java JavaScript (Node.js), Python (Django), PHP, Ruby On Rails, Java, C# та інші.

Код клієнта аналізується браузером користувача. На відміну від сервера, він може реагувати на введення користувача, доступний користувачеві для повного перегляду та редагування. Файли сервера неможливо прочитати безпосередньо, сервер повинен обробляти HTTP-запити. Мови програмування друкують HTML, CSS, JavaScript.

Технологічний стек (Рис. 1.2) представляє собою:

- операційну систему (файлову систему). Частіше всього це Linux;
- web-сервер (Apache, NGINX, IIS та ін.);
- база данних (MySql, MongoDB, MS Sql, Oracle, PostgreSql, Redis та ін.)

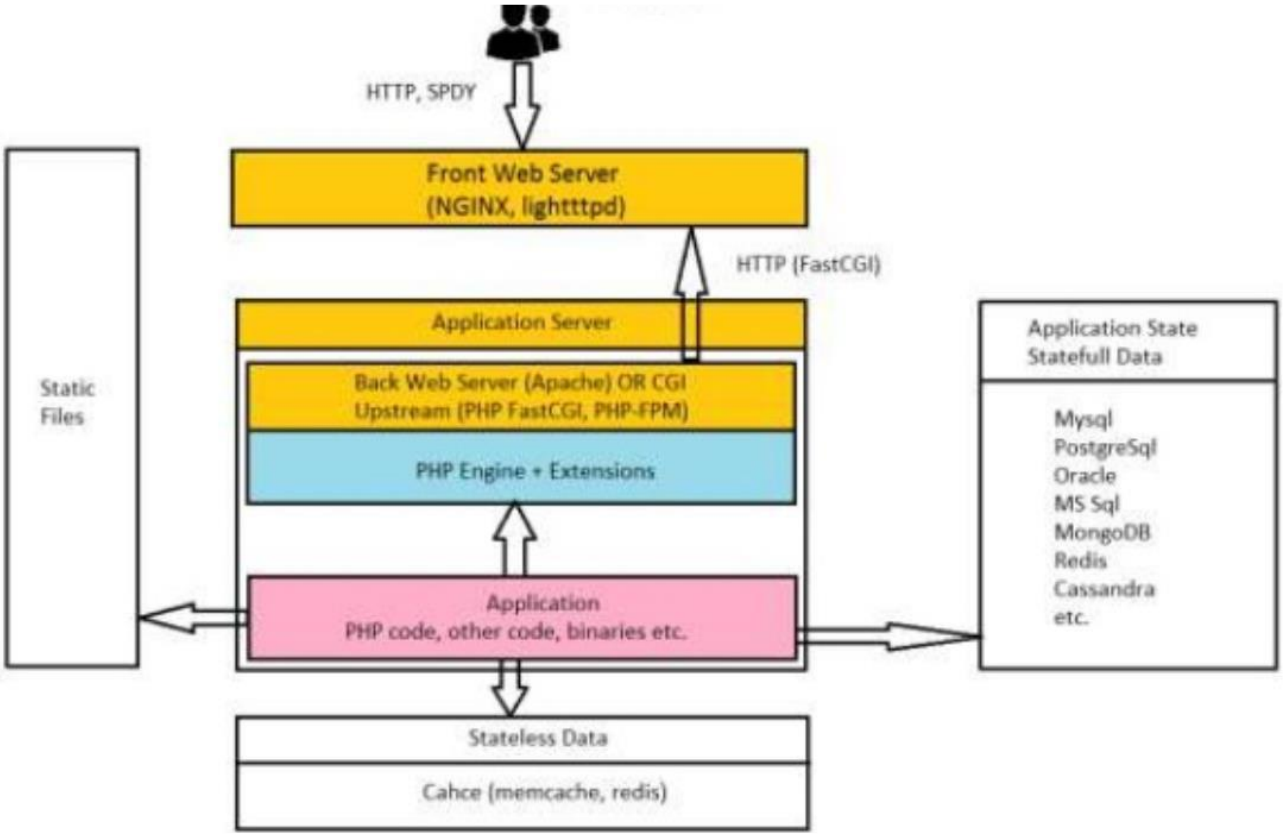

Рис. 1.2. Технологічний стек

Шлях запиту:

1. DNS.
2. HTTP, SPDY.
3. Front Server (NGINX).
4. Back Server or Application Server (Apache, PHP-FPM).
5. PHP code, opcode.
6. Response.

DNS (Domain Name System) - ієрархічна розподілена система перетворення імені хоста (комп'ютера або іншого мережевого пристрою) в IP-адресу. Всі комп'ютери, підключені до Інтернету, включаючи смартфони, настільні комп'ютери і сервери, що надають контент для величезних торгових веб-сайтів, знаходять один одного і обмінюються інформацією за допомогою цифр. Ці цифри називаються IP-адресами. Щоб відкрити веб-сайт в браузері, не потрібно запам'ятовувати довгі набори цифр. Досить ввести доменне ім'я, наприклад example.com, і браузер відкриє потрібну сторінку.

Шлях користувацького запиту починається з DNS:

1. Користувач звертається до DNS для отримки IP адреси.
2. DNS віддає IP адресу користувачеві.
3. Користувач звертається до web-додатка по IP через TCP/IP.
4. Відкривається мережеве з’єднання (HTTP, SPDY).

Стек протоколів TCP/IP – це набір мережевих протоколів, що забезпечують передачу даних, які використовуються в мережах, включаючи мережу Інтернет. Назва походить від двох найважніших протоколів: TCP (Transmission Control Protocol), IP (Internet Protocol).

IP – протокол, який лежить у основі Інтернета, його назва так і розшифровується: Internet Protocol. Наразі використовується дві версії протоколу IP:

1. IPv6 – порівняно нова (опублікована в 1998); IP-адреса має розрядність 128 біт і записується у виді восьми 16-ти бітних полів, з використанням шістнадцяткової системи числення і з можливістю скороченням двох чи більше послідовних нульових полів до «::». Приклад: 2001:db8:1234:5::1:1.
2. IPv4 – опублікована в 1981р; IP- адреса має розрядність 32 біта і записується у виді чотирьох десятичних чисел в діапазоні 0…255 через точку; приклад: 192.0.3.43.

Кожен вузол може бути підключений тільки безпосередньо до вузлів мережі (наприклад, підключений до однієї і тієї ж частини Інтернету), що визначає мережеву адресу - частину IP-адреси, в залежності від маски мережі. Підключення до інших мереж здійснюється через допоміжні вузли - маршрутизатори.

TCP – це протокол, який базується на IP для доставки пакетів, але добавляє дві важливі речі: встановлення з’єднання – це дозволяє йому, на відміну від IP, гарантувати доставку пакетів; порти – дозволяють обмінюватись пакетами між застосунками, а не просто вузлами. Протокол TCP призначений для обміну даними – це надійний протокол, тому що:

1. Забезпечує надійну доставку даних, так як передбачає встановлення логічного з’єднання.
2. Нумерує пакети і підтверджує їх прийом квитанцією, а у випадку загублення пакету організує повторну передачу.
3. Ділить переданий потік байтів на частини – сегменти, і переда їх нижньому рівню, на прийомній стороні знову збирає їх у потік байтів. З’єднання двух вузлів починається з handshake (рукопотискання):
4. Вузол А посилає вузлу В спеціальний пакет SYN – запрошення до з’єднання.

5. В відповідає пакетом SYN-ACK – згодою на установлення з’єднання.
6. А посилає пакет ACK – підтвердження, що згоду отримано.

Після цього TCP з’єднання рахується установленим, і додатки, працюючі на цих вузлах, можуть відправляти один одному пакети з даними. З’єднання означає, що вузли пям’ятають один одного, нумерують усі пакети, які ідуть в обидві сторони, посилають підтвердження про отримання кожного пакету і знову посилають загублені по дорозі пакети. Для вузла А це з’єднання називається вихіднім, а для вузла В – вхідним.

HTTP (Hyper Text Transfer Protocol) – протокол прикладного рівня передачі даних (початково – у виді гіпертекстових документів в форматі «HTML», наразі використовується для передачі довільних даних).

SPDY (читається як «speedy», «спіді») – протокол прикладного рівня для передачі веб-контенту. Протокол розроблено корпорацією Google. По задуму розробників, даний протокол позиціонується як заміна деяких частин протоколу HTTP – таких, як управління з’єднаннями і форматами передачі даних.

Клієнт спілкується з сервером через HTTP-повідомлення. Обмін повідомленнями йде по звичайній схемі «запит-відповідь».

Кожне HTTP-повідомлення складається із чотирьох частин, які передаються у вказанному порядку:

1. Стартовий рядок (англ. Starting line) – оприділяє тип повідомлення;
2. Заголовки (англ. Headers) – характеризує тіло повідомлення, параметри передачі та інші дані.
3. Тіло повідомлення (англ. Message Body) – безпосередньо дані повідомлення. Обов’язково має відділятись від заголовків пустим рядком.

Заголовки та компоненти повідомлень можуть бути відсутніми, але головна панель є обов’язковою, оскільки вона вказує на тип запиту. Відмінність

полягає в версії 0.9 протоколу, в якій повідомлення запиту містить лише перший рядок, а повідомлення-відповідь є лише частиною повідомлення.

Стартові рядки розрізняються для запиту і для відповіді. Рядок запиту виглядає так:

- GET URI – для версії 0.9;
- Метод URI HTTP/Версія – для інших версій.

Тут:

Метод – назва запиту, одне слово заглавними буквами. У версії HTTP 0.9 використовується тільки метод GET;

URI оприділяє шлях до запитаного документу;

Версія – пара розділених точкою цифр (Наприклад: 1.0);

Стартова строка відповіді має наступний формат: HTTP/Версія, код стану, пояснення, де:

- Версія – пара розділених точкою цифр, як в запиті;
- Код стану – три цифри. По коду стану оприділяється подальший вміст повідомлення і поведінка клієнта;
- Пояснення – текстове коротке пояснення до коду відповіді для користувача. Ніяк не впливає на повідомлення і являється необов’язковим.

Методи HTTP - будь-яка послідовність символів, крім елементів керування та обмежень, які показують основну функцію ресурсу. Кожен сервер повинен підтримувати принаймні методи GET і HEAD. Якщо сервер не знає методу, зазначеного клієнтом, має бути повернута умова 501 (не виконується). Якщо сервер знає метод, але його неможливо застосувати до певного джерела, буде

повернуто повідомлення з кодом 405 (неавторизований метод). Крім методів GET і HEAD, часто використовується метод POST.

1.5. Види інтернет-додатків

eCommerce - додатки електронної комерції, до них відносяться інтернет-магазини, маркетплейси, B2B-портали і онлайн-аукціони. Для таких сайтів важлива інтеграція з платіжними системами і службами доставки. Інші пріоритети включають забезпечення безпеки і створення максимально комфортного користувацького досвіду.

Системи автоматизації бизнесу – це CRM, ERP-системи, програми для бухгалтерського обліку і платформи електронного документообігу також можна встановлювати в хмарі. Таким чином всі дані будуть консолідовані на безпечному сервері. А співробітники і партнери з необхідним рівнем доступу зможуть користуватися можливостями системи без прив'язки до конкретного комп'ютера.

SaaS-платформи - SaaS-додатки поширюються не в формі, як традиційний софт, а завантажуються безпосередньо в браузер користувача з вашого сервера. Така модель вигідна як постачальнику ПО, так і користувачам, але щоб побудувати SaaS-продукт, вам знадобиться кваліфікований розробник з досвідом реалізації складних проектів.

MVP - оптимальний спосіб запустити стартап - розробка MVP - мінімального життєздатного продукту. Такий веб-додаток буде мати всі необхідні функції для тестування вашої ідеї і залучення перших користувачів. Ми можемо виступити вашим технологічним партнером і побудувати для вас робочу версію продукту, з якої буде набагато простіше залучити інвесторів.

Агрегатори даних – сайти-агрегатори - автоматизовані платформи, які беруть дані від новинних агентств, тематичних сайтів і інших постачальників контенту. З точки зору користувачів, перевага агрегатора в тому, що всі матеріали зібрані на єдиному майданчику в зручному форматі. У той же час, у власника агрегатора немає необхідності утримувати власну редакцію авторів.

B2B-портали – це інтернет-майданчик, в межах якого реалізуються угоди між компаніями (наприклад, виробником і гуртковиком, між гуртковиком і роздрібним продавцем). До порталу можна підключити партнерів з різних рівнів ланцюга розподілу: дистриб'юторів, дилерів, незалежних гуртовиків, рітейлерів. На порталі здійснюються оптові продажі, оформлення і попередня обробка замовлень, обмін документами і довідковою інформацією. Постачальник може скоротити час на взаємодію з покупцем, розвантажити відділи продажів і підтримки. Замовник може знаходити потрібну продукцію і всю інформацію про неї в інтерактивному каталозі з фільтрами і сортуванням, моментально отримувати довідки без необхідності дзвінків постачаль

Висновки до розділу 1

У даному розділі було розібрано основні теоретичні питання предметної області, а саме поняттю веб-додаток, його переваги та недоліки, види та відмінності.

РОЗДІЛ 2. ПОРІВНЯЛЬНИЙ АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ, ІНСТРУМЕНТІВ ТА ТЕХНОЛОГІЙ

2.1. Аналітичний огляд інструментів та технологій для розробки веб-додатку

2.1.1. Hypertext Markup Language

HTML або Hypertext Markup Language – це код, який використовується для створення веб-сторінок і веб-додатків. HTML був створений у всесвітній мережі наприкінці 1980-х і на початку 1990-х років. Ця мова дозволяє веб-розробникам вказувати веб-браузерам, як відображати такі функції, як зображення, текст, форми та інтерактивні функції.

Веб-майстри зазвичай використовують HTML разом із двома іншими мовами програмування, CSS і JavaScript, для створення веб-сторінок і програм, доступ до яких можна отримати через веб-браузери, такі як Chrome, Firefox, Safari та Edge. Стандарти HTML і CSS історично підтримували World Wide Web Consortium (W3C).

У 1980 році Тім Бернер-Лі почав працювати над прототипом HTML для Європейської організації ядерних досліджень. Протягом десятиліття Burners Lee розробив всюдисущу мову, а також інтернет-браузер і серверне програмне забезпечення.

Протягом наступних двох десятиліть Консорціум World Wide Web Consortium (W3C) продовжить розробляти міжнародні стандарти коду. Друге

видання HTML було випущено в 1995 році, а протягом наступних кількох років відбулося ще кілька оновлень.

Хоча HTML заснований на правилах SGML (стандартна загальна мова розмітки, яку можна використовувати для опису деяких металевих мов для маркування документів), XHTML заснований на правилах XML, суворому підрозділі правил SGML. Документи XHTML повинні бути точними XML-документами, тому, на відміну від HTML, їх можна запускати за допомогою стандартних інструментів обробки документів XMTML, що вимагає відносно складнішого, складного та повільнішого аналізу. XHTML часто розглядається як інтерфейс HTML і XML, оскільки ця вимога є покращенням HTML за допомогою XML. XHTML 1.0 Рекомендований консорціумом W3C 26 січня 2000 р. XHTML 1.1 став рекомендацією W3C 31 травня 2001 року.

HTML перекладається браузером і виглядає як зручний для людини документ.

HTML – це програма SGML (стандартна загальна мова аутентифікації) і відповідає міжнародним стандартам ISO 8879.

HTML: документ – це текстовий файл, позначений спеціальними (природними, текстовими) командами. Текстовий формат для вибору веб-документів вибирається виходячи з основних вимог до веб-документації: простота, можливість безпосереднього перекладу в будь-яку операційну систему, мінімальний розмір файлу, простота редагування та інтерпретації.

Мова розмітки гіпертексту HTML дозволяє вказувати різні типи елементів, які забезпечують функції документації. Елементи HTML визначаються за допомогою тегів (від англійського label - тег). Усі теги HTML знайдені в тексті документа перекладаються браузером при відображенні документа.

HTML – це мова розмітки, фраза, яка використовується для опису того, як програміст використовує код для позначення тексту. Мови розмітки, такі як

HTML і XML, читаються і відрізняються від машинних мов, які є шістнадцятковим або двійковим кодом.

Створюючи веб-сторінку з використанням HTML, автор використовує ряд ключових елементів або функцій, які може прочитати будь-який Інтернет-браузер, який має доступ до Інтернету. Більшість із цих функцій поєднуються з «Початковим тегом» і «Останнім тегом».

Наприклад, автор може використовувати букву «P» у дужках (Рис. 2.1), щоб позначити початок абзацу, а другу «P» у кінці другого набору дужок. Перший фрагмент коду відкриває абзац, а другий фрагмент коду закриває абзац.

```
<p>Текст</p>
```

Рис. 2.1. Приклад використання

HTML 5 є найбільш відповідним стандартом для цієї мови жестів сьогодні. За цим критерієм базується найбільша кількість веб-сайтів в Інтернеті. Ми отримуємо підтримку мультимедіа (аудіо та відео) в останній версії HTML 5. Основною особливістю веб-сторінок сьогодні є HTML 5. Забезпечує послідовний перегляд веб-сторінок. А всі інші стилі та функції додаються за допомогою CSS, PHP, JavaScript. Тому вам потрібно вивчити HTML, перш ніж ви зможете вивчати ці мови.

Можна сказати, що HTML є основним будівельним блоком веб-розробки. Усі сучасні сервіси веб-дизайну, такі як WordPress, Wiki, Webli, використовують HTML 5 для створення веб-сайту.

Елемент TML відрізняється від іншого тексту в документі «тегами» за назвою навколишнього елемента «<» і «>». Назва елемента в тегу не чутлива до регістру. Тобто це можуть бути великі або малі літери чи комбінації. Наприклад, тег title можна записати як "Title", "Title" або будь-яким іншим способом.

Розробник веб-сайту може використовувати функції та елементи HTML, щоб визначити кожен аспект сторінки та те, як вона виглядає для користувача. Для створення веб-сайту програміст може використовувати CSS і JavaScript, а також додаткові мови, такі як PHP, jQuery та XML.

Програмісти-початківці вивчатимуть HTML як оригінальну мову програмування та продовжуватимуть вивчати розширені посібники з JavaScript та CSS.

Вивчення основ HTML може бути корисним для майбутніх програмістів, а також для тих, хто хоче зрозуміти, що потрібно для створення веб-сайтів, які можна читати на комп’ютері чи смартфоні щодня.

2.1.2. Cascading style sheets

CSS – це мова каскадних таблиць стилів (англ. cascading style sheets) , яка використовується для опису стилів багаторазового використання для подання документів, написаних мовою розмітки. Його концепція була створена компанією Håkon Wium Lie у 1994 році. У грудні 1996 року WSC розробила специфікацію для CSS і сьогодні дозволяє веб-розробникам змінювати макет і зовнішній вигляд своїх веб-сторінок. Наприклад, CSS може використовуватися для зміни шрифту, який використовується в певному HTML-елементі, а також його розміру та кольору. Один CSS-файл може бути пов'язаний з декількома сторінками, що дозволяє розробнику одночасно змінювати зовнішній вигляд усіх сторінок. Наведене нижче поле містить основний приклад використання коду CSS для визначення шрифтів, кольору гіперпосилань та кольору посилання, коли курсор миші наводить курсор миші. У цьому конкретному прикладі ми змінюємо лише теги HTML <a> та <body>, а не створюємо нові селектори класу чи id. [ ]

```
body {
 font: normal 100% "trebuchet ms", Arial, Helvetica, sans-serif;
}
a {
 color: #000000;
}
A:visited {
 color: #005177;
}
a:hover {
 color: #005177;
}
```

Рис. 2.2. Поле з CSS кодом

Переваги CSS:

1. CSS економить час - ви можете написати CSS один раз, а потім повторно використовувати той же аркуш на декількох HTML-сторінках. Ви можете визначити стиль для кожного елемента HTML і застосувати його до якомога більше веб-сторінок.

2. Сторінки завантажуються швидше - якщо ви використовуєте CSS, вам не потрібно щоразу писати атрибути тегів HTML. Просто напишіть одне правило CSS тегу та застосуйте його до всіх випадків цього тегу. Так менше коду означає швидше завантаження.

3. Простота обслуговування - Щоб зробити глобальну зміну, просто змініть стиль, і всі елементи на всіх веб-сторінках будуть оновлені автоматично.

4. Покращені стилі до HTML - CSS має набагато ширший набір атрибутів, ніж HTML, тому ви можете надати набагато кращий погляд на свою сторінку HTML порівняно з атрибутами HTML.

5. Сумісність декількох пристроїв - аркуші стилів дозволяють оптимізувати вміст для більш ніж одного типу пристроїв використовуючи один і той же документ HTML, різні версії веб-сайту можуть бути представлені для портативних пристроїв, таких як КПК та мобільних телефонів або для друку.

Розглянемо версії CSS. Каскадні таблиці стилів рівня 1 (CSS1) вийшли з W3C як рекомендація в грудні 1996 року. Ця версія описує мову CSS, а також просту модель візуального форматування для всіх тегів HTML.

CSS2 став рекомендацією W3C у травні 1998 року та базується на CSS1. Ця версія додає підтримку для специфічних таблиць стилів, наприклад, принтери та аудіопристрої, завантажувані шрифти, розміщення елементів та таблиці.

2.1.3. JavaScript

JavaScript є найбільш часто використовуваною мовою сценаріїв для створення скриптів веб-браузера, вбудованих у веб-сторінки. JavaScript Sun Microsystems, Inc - це зареєстрована торгова марка.

JavaScript створено для «оновлення веб-сторінок». Програми на цій мові називаються скриптами. Вони можуть бути записані безпосередньо на веб-сторінки HTML і автоматично запускатися, коли сторінка завантажується. Скрипти надаються і будуть реалізовані як простий текст. Для запуску вони не вимагають спеціальної підготовки чи компіляції. У цьому відношенні JavaScript сильно відрізняється від будь-якої іншої мови під назвою Java.

Коли JavaScript був створений, він спочатку називався «Живий скрипт». Але в той час Java була дуже популярна, тому було вирішено, що це допоможе зберегти нову мову як Java «молодший брат».

Але в міру покращення JavaScript став повністю незалежною мовою з власним ECMAScript і тепер не має нічого спільного з Java.

JavaScript зазвичай використовується як сценарна мова клієнта. Це означає, що код JavaScript написаний на сторінці HTML. Коли користувач шукає HTML-сторінку з JavaScript, сценарій надсилається браузеру, і браузер повинен вирішити щось зробити.

Оскільки сценарій знаходиться на сторінці HTML, ваші сценарії можуть переглядати та копіювати будь-хто, хто переглядає вашу сторінку. Однак я вважаю, що ця прозорість є великою перевагою, оскільки ви можете бачити, вивчати та використовувати скрипт Java, який знайдете на WWW.

JavaScript можна використовувати в контекстах за межами веб-браузера. JavaScript на стороні сервера Netscape, створений як мова CGI, може виконувати те саме, що Perl або ASP. Немає жодних причин, чому JavaScript не слід використовувати для написання реальних і складних програм. Однак цей сайт стосується лише використання JavaScript у веб-браузерах.

Сьогодні JavaScript працює не тільки в браузері, а й на сервері або будь-якому іншому пристрої за допомогою спеціальної програми під назвою JavaScript Mechanical. Браузер має вбудований «движок», який іноді називають віртуальною машиною JavaScript. Різні двигуни мають різні «кодові назви». приклад:

- V8 - в Chrome і Opera.
- SpiderMonkey - у Firefox.

Сучасний JavaScript є «безпечною» мовою програмування. Низькорівневий доступ до пам’яті або ЦП не надається, оскільки він розроблений для браузерів, яким це не потрібно.

Можливості JavaScript значною мірою залежать від середовища, в якому він використовується. Наприклад, Node.js JavaScript підтримує функції, які дозволяють читати/записувати випадкові файли, надсилати мережеві запити тощо.

У браузері JavaScript може робити все, що стосується шахрайства на веб-сайті, взаємодії з користувачем та веб-сервера. Наприклад, JavaScript може наступне у веб-браузері:

1. Додати новий HTML на сторінку, змінити існуючий вміст, змінити стилі;
2. Реагувати на дії користувача(натискання миші, рух вказівника, натискання клавіш);
3. Надсилати запити по мережі на віддалених серверів, закачувати та завантажувати файли (так звані технології AJAX та COMET);
4. Отримати та встановити файли cookie, задавати питання відвідувачеві, показувати повідомлення;
5. Запам’ятовувати дані на стороні клієнта («локальне зберігання»).

2.2. Необхідність застосування фрейморку у веб-розробці

2.2.1. Поняття фреймворку

Фреймворки – це програмні продукти, які полегшують створення та обслуговування технічно складних проектів. Фреймворк зазвичай містить лише базові програмні модулі, а всі компоненти на основі проекту реалізовані розробником. Це забезпечує не тільки високі темпи зростання, але й високу продуктивність та надійність рішень.

Web Framework – це платформа для створення веб-сторінок та веб-додатків, які полегшують розробку та інтеграцію різних компонентів великого програмного проекту. Завдяки широкому спектру можливостей і високій продуктивності в бізнес-логіці ця платформа особливо підходить для створення складних сайтів, бізнес-додатків і веб-сервісів.

2.2.2. Важливість фреймоврків у розробці

З комерційної точки зору розробка на фреймворку є більш економічно ефективною та якісною, ніж написання чистою мовою програмування без використання будь-яких платформ. Розробка без використання платформи може бути ефективним вирішенням лише двох проблем - проект дуже простий і не потребує подальшої розробки або дуже зайнятий і вимагає оптимізації дуже низького рівня (наприклад, десятки тисяч викликів веб-сервісу в секунду) . У всіх інших випадках розробка програмної платформи відбувається швидше і краще.

Якщо порівнювати фреймворки з іншими классовими платформами - SaaS, CMS або CMF - фреймворки ефективніше використовувати в складній бізнес-логіці та проектах з високими вимогами до швидкості, надійності та безпеки. Але без значних вимог у простих і звичайних проектах темпи зростання і вартість на фреймворку будуть вище, ніж у SaaS або CMS.

Однією з головних переваг використання фреймів є те, що фреймворк визначає інтегровану структуру для додатків, побудованих на основі. Тому додатки на фреймворках дуже легко виправити і відфільтрувати, тому що стандартна структура класу зрозуміла всім розробникам на цій платформі і не потрібно багато часу, щоб знайти місце для реалізації іншого функціоналу. Більшість фреймворків в розробці веб-додатків використовують методологію MVC (model-view-control) – тобто багато фреймворків мають подібний підхід до організації елементів програмування, що дозволяє легко зрозуміти архітектуру програми навіть для незнайомого розробника фреймворка. [2]

Розробка архітектури програмного забезпечення також дуже проста при створенні на основі фреймворка – дана методологія часто включає в себе найкращі методи розробки програмного забезпечення і легко дотримуючись цих правил усуває багато проблем і помилок у проектуванні. ПО своїй суті являє

собою сукупність конкретних і абстрактних класів, пов'язаних між собою і впорядкованих за каркасним методом. Конкретні класи зазвичай реалізують взаємні відносини між класами, а абстрактні класи являють собою точки розширення, в яких закладений у фреймворк базовий функціонал може бути використаний «як є» або адаптований під завдання конкретного додатка. Для забезпечення можливостей у більшості фреймів використовуються методи об’єктно-орієнтованого програмування — наприклад, компоненти програми можуть бути успадковані від базових компонентів фреймворка або окремі модулі можуть бути з’єднані як сміття.

Веб-екосистеми також багаті на багатофункціональні готові програми. Розробникам не потрібно «винаходити велосипеди» під час співпраці, оскільки вони можуть використовувати вже створений додаток спільноти I це не тільки заощаджує час і гроші, але й робить рішення більш стабільним — компонент, який використовують тисячі інших розробників, часто краще реалізується та краще тестується в різноманітних ситуаціях як у випадку одного розробника так і навіть невеликою групою.

2.2.3. Найактуальніші фреймворки JavaScript

Рис. 2.3. Логотип Angular

Angular - одне з найпотужніших середовищ JavaScript. Google використовує цю платформу для розробки односторінкової програми (SPA). Це середовище розробки відоме насамперед тому, що воно надає розробникам

найкращі умови для об'єднання JavaScript з HTML та CSS. Понад півмільйона сайтів, таких як google.com, youtube.com тощо, використовують Angular.

Це також переважне середовище інтерфейсу JavaScript для розробників додатків Google. Angular має компонентну структуру, як і React. Можна маніпулювати, вкладати та використовувати їх у міру потреби. Потрібно буде використовувати TypeScript, щоб написати програму в Angular. Це розширений набір JavaScript, який використовує той же синтаксис, але також підтримує статичну типізацію та класи. У TypeScript даються модифікатори доступу, перерахування, узагальнення, гібридні типи та багато іншого. Простіше кажучи, Angular це фантастична платформа, на яку можна поглянути, якщо ви новий розробник.

Переваги:

1) допомагає створювати прогресивні програми (PWA);
2) зручно маніпулювати DOM-елементами;
3) висока швидкість та продуктивність;
4) крута крива навчання;
5) вбудований механізм застосування залежностей;
6) підтримка Google та потужна екосистема

Рис. 2.4. Логотип React

В даний час лідером у галузі інфраструктури JavaScript UI є React. Спочатку розробники Facebook почали працювати над цим, щоби спростити свою роботу. Програма під назвою Facebook Ads росла дуже швидко, що означало складне управління та підтримку. В результаті команда почала створювати

структуру, яка допоможе їм ефективно. У них був ранній прототип до 2011 року, а через два роки, структура була з відкритим вихідним кодом і доступна для громадськості. В даний час його використовують багато бізнес-гіганти: AirBNB, PayPal, Netflix і т.д.

React базується на компоненті багаторазового використання. Простіше кажучи, це блоки коду, які можна класифікувати як класи чи функції. Кожен компонент представляє певну частину сторінки, таку як логотип, кнопку або поле введення. Використані параметри називаються реквізитами, що означає властивості. Говорячи про синтакс, більшість розробників сходяться на думці, що React легко освоїти, коли вже володієте JavaScript.

React використовує JSX, синтаксис XML, який поєднує JavaScript і HTML. Це не шаблон JavaScript; це повний JavaScript. Спочатку деякі нові розробники можуть знайти JSX трохи заплутаним. Однак, попрацювавши з ним якийсь час, розробник зрозумієте, наскільки це корисно. Це бібліотека, на яку розробники повинні звернути увагу, якщо ви займаєтеся інтерфейсною веб-розробкою

Переваги:

1) допомагає створювати прогресивні програми (PWA);
2) зручно маніпулювати DOM-елементами;
3) висока швидкість та продуктивність;
4) крута крива навчання;
5) вбудований механізм застосування залежностей;
6) підтримка Google та потужна екосистема.

Рис. 2.5. Логотип Aurelia

Aurelia — це інтерфейсне середовище з відкритим кодом, розроблене Робом Айзенбергом. Aurelia 1.0 була випущена вперше у 2016 році. Aurelia складається з функціонально-орієнтованих модулів, таких як плагіни, маршрутизація, тестування, впровадження залежностей та багато іншого. Можемо використовувати середовище Aurelia JS для розробки веб-додатків, програм для мобільних пристроїв та комп'ютерів. Модульність також дозволяє створювати програми різних розмірів. Aurelia підтримує як Babel, і TypeScript. Він також повністю сумісний із майбутніми версіями JavaScript.

З варіантів, представлених у цьому розгляді, Aurelia може бути одним із наймолодших: ми отримали версію 1.0 тільки в середині 2016 року. Однак це дає йому невелику перевагу: за словами його творців, Aurelia пропонує компонентну модель, яка найкраще відповідає найкращим веб-стандартам. Він використовує безліч нових функцій ECMAScript (ECMAScript допомагає визначати стандарти JavaScript) і рекомендує використовувати ці нові функції для написання коду. Це добре, тому що не потрібно вивчати власний синтаксис, який працює лише з Aurelia

Переваги:

1) висока продуктивність;
2) велика спільнота;
3) адаптивна прив'язка даних;
4) ефективність пам'яті.

Рис. 2.6. Логотип Vue

Vue – це JavaScript-фреймворк з відкритим вихідним кодом для створення креативного інтерфейсу. Інтеграція з Vue у проектах, які використовують інші бібліотеки JavaScript, спрощена, оскільки вона розроблена для адаптації. Більше 36 000 веб-сайтів зараз використовують Vue. Такі компанії, як Stackoverflow, PlayStation і т.д., покладаються на Vue для своїх сайтів інтерфейсу користувача.

Vue.js також досить простий у освоєнні: все, що потрібно, це JavaScript і HTML. Іншою сильною стороною Vue.js є його інтерфейс командного рядка (CLI). Це базовий інструмент, який прискорює розробку, пропонуючи масу плагінів, пресетів, миттєвого прототипування та інтерактивного інструменту розробки проектів. Деякі з його функцій включають компоненти, шаблони, переходи та двостороннє зв'язування даних, а також фокус реактивності. Реактивність виникає при зміні або оновленні будь-якого з об'єктів JavaScript у Vue. Vue.js використовує те, що називається Shadow DOM, що робить рендеринг сторінки швидким.

Переваги:

1) високий ступінь настроюваності;
2) легко вчити;
3) підтримка CSS переходів та анімації;
4) гнучкість та модульність.

Рис. 2.7. Логотип JQuery

JQuery, мабуть, найпопулярніша бібліотека JavaScript з такою кількістю функцій для сучасної розробки. JQuery - це швидка та лаконічна бібліотека JavaScript, створена Джоном Резігом у 2006 році. Це кроссплатформенна бібліотека JavaScript, призначена для спрощення HTML-скриптингу на стороні клієнта. Понад 19 мільйонів веб-сайтів зараз використовують jQuery! Такі компанії, як WordPress, Facebook, Google, IBM та багато інших, покладаються на jQuery для забезпечення свого роду перегляду веб-сторінок.

Ви можете використовувати API jQuery для обробки, анімації та маніпулювання подією в HTML-документі, також відомому як DOM. Крім того, jQuery використовується з будівельними інструментами Angular та React App. Одним словом, одна з найважливіших бібліотек JavaScript для веб-розробки.

Рис. 2.8. Логотип Ember

Ember.js це JavaScript-фреймворк з відкритим вихідним кодом, який був спочатку випущений Єгудою Кацем в 2011 році. Спочатку він називався SproutCore 2.0, перш ніж став називатися Ember.js. Робота над Ember Framework почалася в 2011 році, а версія 1.0 була випущена через два роки.

Він також добре масштабується і може використовуватися для великих проектів. Apple Music, яка має понад 60 мільйонів передплатників у всьому світі, була побудована на Ember. Якщо ви хочете побачити, що ви зможете, є інструмент під назвою Ember Inspector, який дозволяє вам ближче познайомитися з проектами Ember в Інтернеті.

Ember має відмінний інструмент для складання, запозичений з багатьох інших середовищ SPA, званий Ember CLI. Цей інструмент збирання має все необхідне для початку роботи. Вам потрібний роутер? Там вбудовано. Потрібно пройти тестування? Він також убудований. Вам потрібно працювати із внутрішніми даними? Є дані на Ембер. Ember.js дотримується багатьох тих же принципів, що і Ruby в Rails. Він дуже продуманий, гнучкий.

Переваги:

1) висока продуктивність;
2) адаптивність та гнучкість;
3) дозволяє писати розширені HTML-теги;
4) можна виконувати тести усередині браузера;
5) швидко зростаюче співтовариство.

Рис. 2.9. Логотип Meteor

Веб-фреймворк під назвою Skybreak було випущено наприкінці 2011 року. За кілька місяців команда змінила назву на Meteor. Використовуючи простий JavaScript, ви можете створювати програми, які можна використовувати на різних платформах, включаючи, крім іншого, Android та iOS. Середа Meteor JS надає рішення з повним стеком. Використання цієї інфраструктури включає важливі області, такі як внутрішня розробка, управління базами даних, бізнес-логіка та зовнішній рендеринг.

Meteor позиціонує себе як найшвидший спосіб створення програм JavaScript. Він завойовує популярність на ринку з більш ніж 13 000 веб-сайтів, які використовують Meteor. Такі сайти, як mtv.com, meteofrance.com і т.д., використовують Meteor для створення свого інтерфейсу користувача.

Переваги:

1) екосистема ізоморфного розвитку (IDevE);
2) вбудоване перезавантаження браузера;
3) користувальницький менеджер пакетів;
4) потужна хмарна платформа для розгортання, масштабування та моніторингу клієнтських програм;
5) реактивні візерунки.

2.3. Аналіз веб–додатків зі сторони фінансових рішень

Онлайн-сервіс для організації та нагляду за робочим процесом є основою для побудови успішної бізнес-команди. Тому все більше IT-компаній впроваджують у свою роботу такі програми. Перед керівництвом компаній постало непросте питання вибору такої послуги. Тому враховуйте критерії відбору.

Перше і найважливіше – це функціонал, який може забезпечити реальний сервіс. Кожен споживач шукає додаток, який дає йому можливість виконувати свої унікальні завдання. По друге доступність, адже для когось безкоштовний продукт єдина можливість, а для когось – це функціонал сервісу, за якість якого часто доводиться платити. Існує багато видів покупок товарів, тому їх можна розділити на:

1) одноразова покупка;
2) покупка певного обьєму (лімітовану чи безлімітну кількість користувачів);
3) підписка на певний період часу;
4) підписка на певний пакет послуг (функціональні можливості, які надаються).

Розглянемо веб-додаток Asana, програмне забезпечення, розроблене для спільної роботи над проектами електронної пошти в Інтернеті та мобільних пристроях. Його створили засновник Facebook Дастін Москвіц і колишній інженер Джастін Роузстайн. Це означає, що ця програма надає можливість організовувати робочі процеси в команді, але це платне. Місячна підписка коштує не менше 11 доларів і може коштувати до 30,5 доларів на людину, залежно від типу підписки. Звичайно, для невеликих компаній це дорого, тому багато хто відмовляється від використання програми. Веб-додаток Asana можна використовувати безкоштовно, але він значно знижений у функціональності і для деяких користувачів - 15 осіб, тому ця програма не корисна для компаній з невеликим капіталом.

Додаток Trello є однією з найпопулярніших онлайн-систем управління проектами серед невеликих компаній і новачків. Він використовує парад управління проектами, відомий як каннабіс. Проекти представлені дошками з деталями. Список містить картки дій. Картки повинні переміщатися від

попереднього списку до наступного (перетягування), щоб показувати переміщення дії від ідеї до експерименту. Картка може бути присвоєна користувачам, відповідальним за неї. Користувачів і дошки можна згрупувати разом. Шлях можна протестувати безкоштовно, користувач має можливість використовувати 10 дощок, але з обмеженою функціональністю. Після того, як ви скористаєтеся 10 дошками, і вам доведеться підписатися, щоб користуватися повною функціональністю програми, ціна становитиме 12,5 доларів на місяць і збільшиться в функціональності.

У ході аналізу існуючих додатків стає зрозумілим, що для невеликих компаній не має повноцінних безкоштовних додатків.

2.4. Вітчизняний і зарубіжний досвід застосування веб–додатку для організації та контролю робочого процесу компанії

Веб-додатки для організації та моніторингу діяльності компанії є звичним явищем в IT-індустрії, а також у великих корпораціях та організаціях.

Вітчизняних веб-додатків для організації та моніторингу діяльності компанії немає, але українські компанії активно використовують такі програми, як Danone, Vodafone та Aven. Крім того, майже всі IT-компанії у своєму бізнесі використовують один із додатків: Trello, Asana, Slack.

Розглянемо зовнішні додатки. Одним з найпопулярніших веб-додатків для організації та моніторингу роботи компанії є додаток Jira, який дозволяє швидко створювати завдання, призначати їх певному виконавцю або групі, визначати час і терміновість виконання, а також виконувати одночасно. Ви також можете визначити пріоритети завдань.

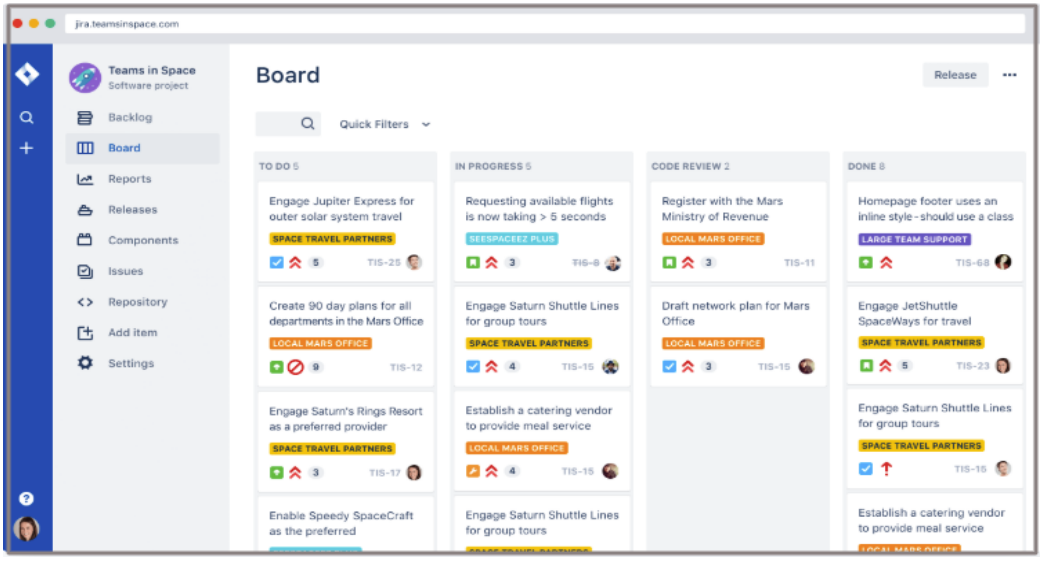

Рис. 2.10. Головна сторінка Jira

Аналогічний сервіс Trello, дозволяє створювати «дошки дій», щоб визначити їх пріоритети, визначити час виконання та призначити керівників. Ви також можете поспілкуватися в програмі та залишити повідомлення про конкретне завдання, що скорочує час на обговорення деталей в іншій програмі.

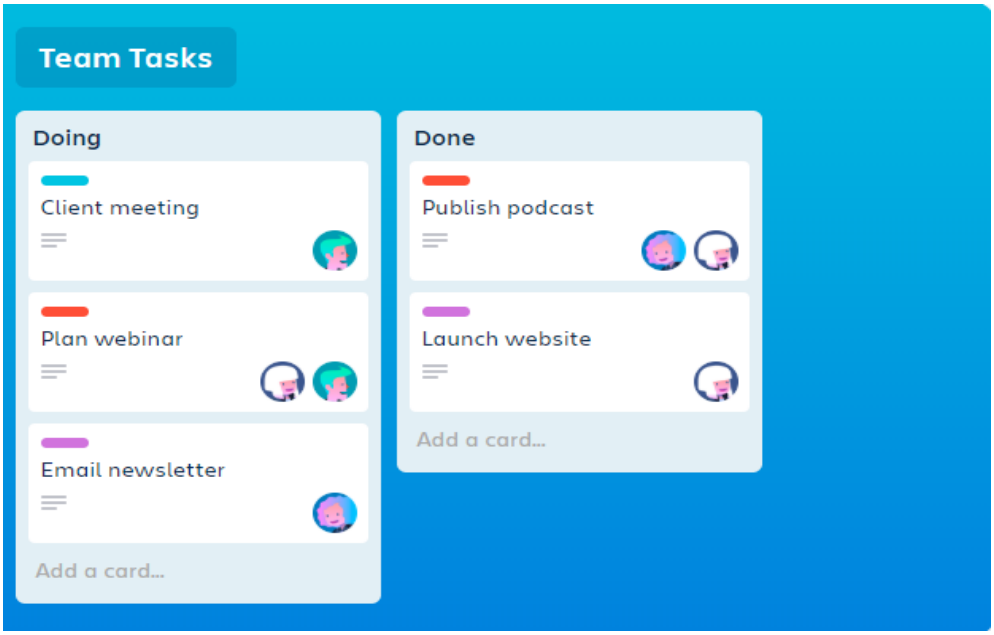

Рис. 2.11. Дошка завдань Trello

На пріклад Asana допомагає вам створювати, призначати та керувати завданнями. Вона дозволяє помістити кожне завдання у власний контекст і контекст проекту, з яким вона пов'язана. Завдяки цьому ви можете письмово обговорювати завдання з членом команди, не втрачаючи контексту. Це допомагає підтримувати конструктивну розмову. Ви також можете завантажувати або експортувати файли з комп'ютера, диска Google, Dropbox та інших хмарних сховищ.

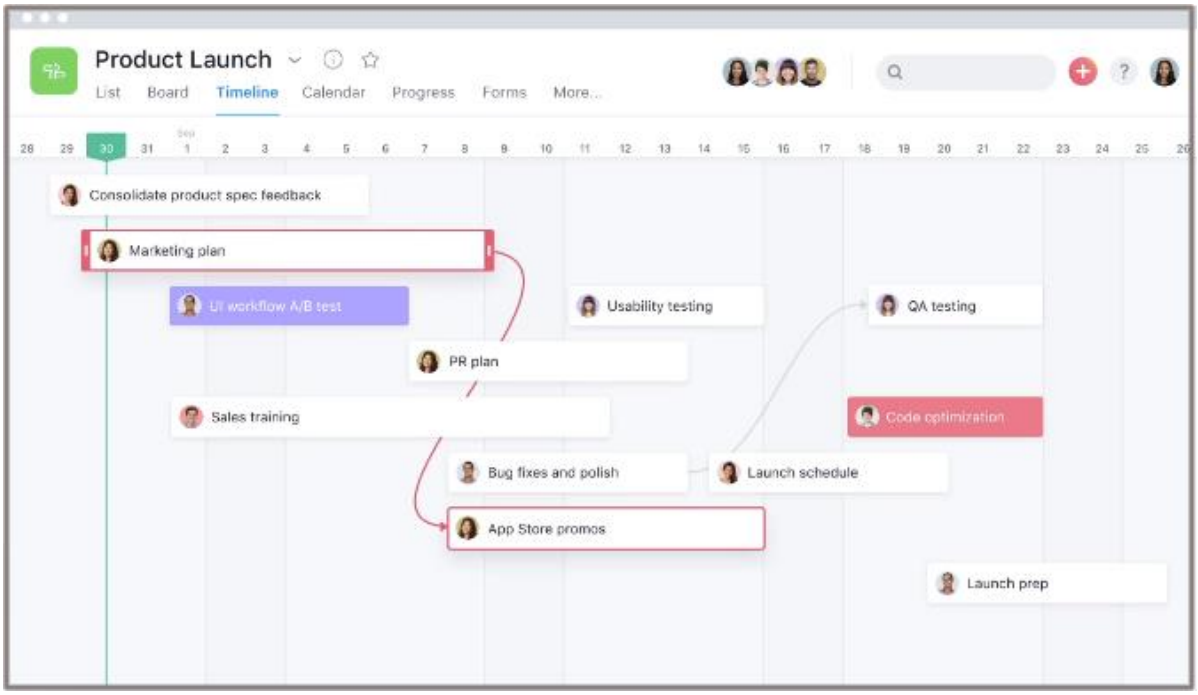

Рис. 2.12. Сторінка менеджменту часу Asana

Slack – це інструмент для обміну миттєвими повідомленнями, дистанційного спілкування команд та віддалених працівників. Він має відмінний функціонал для управління проектами, призначення команд, планування релізів та запусків, перегляду інформації про співробітників та продажу.

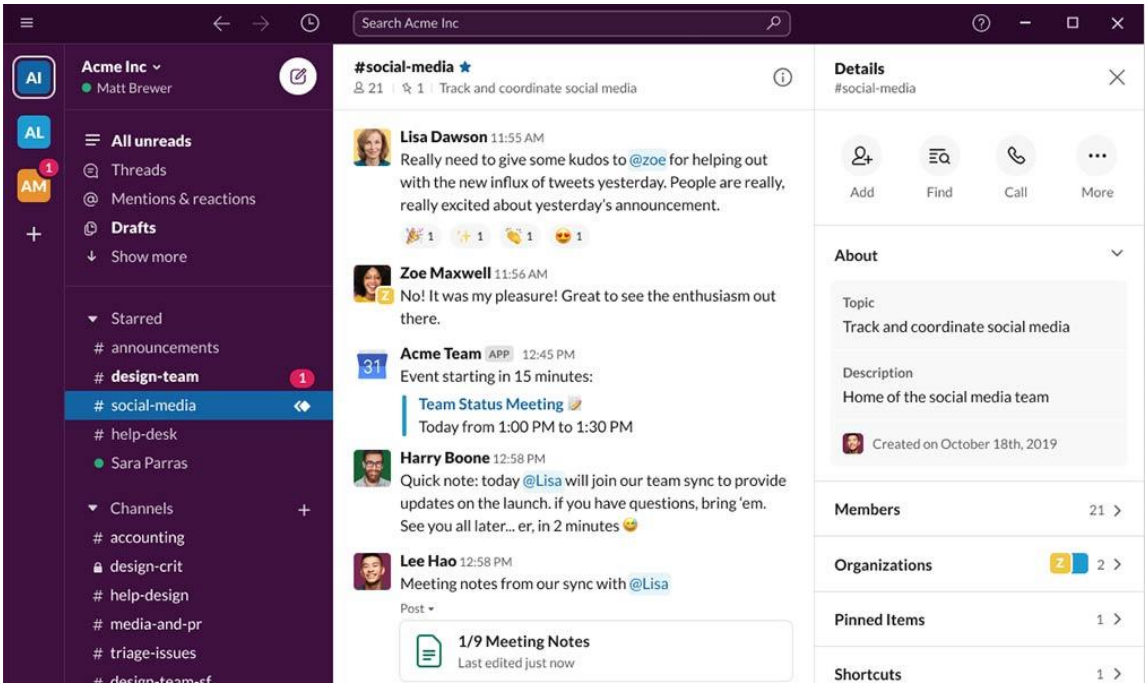

Рис. 2.13. Ілюстрація інтерфейсу Slack

Wrike - це онлайн-інструмент корпоративного рівня для управління проектами та спільної роботи, призначений для масштабування та прискорення впливу на бізнес. Безкоштовна версія Wrike дозволяє одночасно працювати п'ятьом користувачам. Він пропонує безліч функцій через веб-додатки або мобільні програми, такі як управління завданнями, перегляд дошки та електронної таблиці, потік активності в реальному часі, спільне використання файлів та 2 ГБ дискового простору. Він також забезпечує базову інтеграцію програмного забезпечення (Google Drive, MSFT Office 365 тощо) та інтеграцію з хмарним сховищем. Функції, що настроюються - існують різні способи перегляду проектів, а також зрозуміле інформування про потреби і завдання. Можливості перевірки, що дозволяють користувачам легко обмінюватися даними та співпрацювати

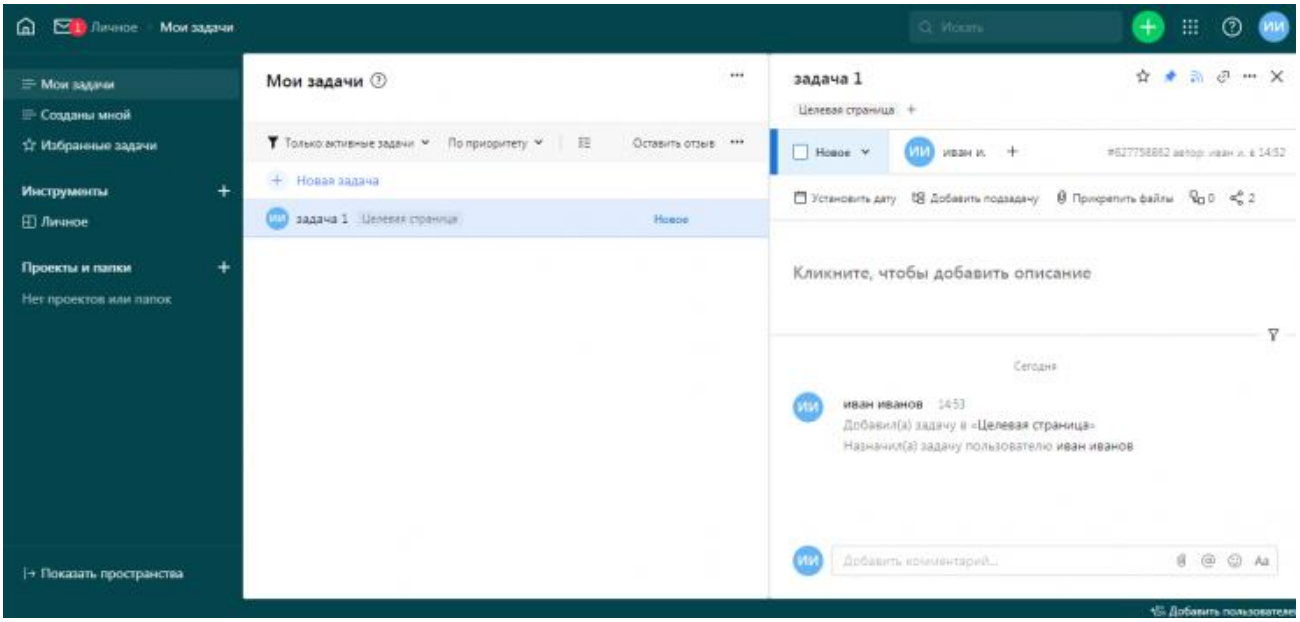

Рис. 2.14. Розділ «мої задачі» у додатку Wrike

Airtable - це платформа для спільної роботи, розроблена спеціально для допомоги компаніям в управлінні проектами, конвеєрі контенту, дослідженнях та багато іншого. Airtable є гібридом бази даних та електронної таблиці. Поля даних в Airtable схожі на комірки в електронній таблиці, але мають такі типи, як «прапорець», «номер телефону» і «список, що розкривається», і можуть посилатися на вкладення файлів, наприклад зображення Безкоштовна версія інструмента пропонує важливі функції, такі як спільна робота, підтримка електронною поштою, різні налаштування перегляду, необмежену кількість баз, а також історія змін та знімків терміном до 2 тижнів. Доб плюсів додатку можна віднести можливість пов'язувати зовнішні або внутрішні документи у дашбордах/проектах та чистота і візуально привабливий інтерфейс користувача.

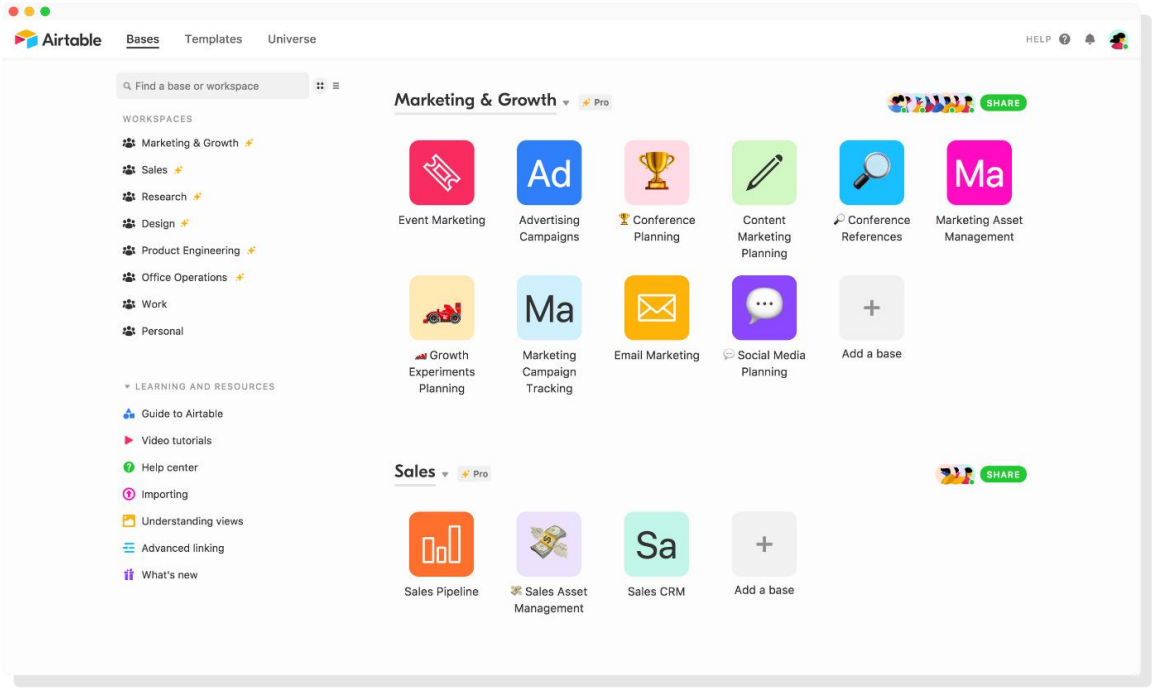

Рис. 2.15. Головна сторінка Airtable

2.5 Порівняльна характеристика

У таблиці 1.1 представлена порівняльна характеристика функціоналу найпривабливіших додатків с оглянутих раніше.

Таблиця 1.1

Додаток/ Функціонал	Asana	Jira	Trello	Slack
Постановка задач	+	+	+	–
Інтеграція з іншими додатками	+	+	+	+
Комунікація	–	–	+	+
Профіль співробітника	+	–	–	+

Безкоштовність	–	–	–	–

В результаті аналізу таблиці можна зробити висновок, що кожне з розглянутих додатків має як переваги, так і недоліки, але основним недоліком цих додатків є висока вартість покупки. Тому вирішено налаштувати додаток із певною функцією, яка надаватиме основні функції, наприклад, зберігати профіль співробітника, створювати завдання, підключатися безпосередньо в програмі та бути безкоштовним.

2.6. Постановка задачі та опис діяльності організації

Метою дослідження є веб-додаток для організації та контролю діяльності компанії.

Предметом дослідження є процес створення та ведення профілю користувача, процес створення завдань, процес спілкування.

Завдання даної роботи полягає у реалізації власного веб-додатка для організації та моніторингу діяльності компанії на основі переваг та недоліків розроблених рішень. Вивчивши готові рішення та методи їх реалізації, ми можемо зробити висновок, що доцільно було б розробити безкоштовний додаток з невеликою кількістю практичного, зручного інтерфейсу та високим рівнем безпеки. Це пов'язано з бажанням надати користувачам безпечний зв'язок і зберігання даних у додатку, оскільки такі системи в першу чергу використовуються для організації роботи в IT-компаніях, для яких конфіденційність і безпека ключ до успіху. Тому розробка повинна відповідати наступним основним вимогам:

1) безкошновність;
2) безпечність;
3) функціональність.

Метою роботи є підвищення ефективності та якості додатка в мережі Internet

Для дослідження використовується один із методів теоретичного аналізу – формалізація.

Формалізація як метод дослідження має багато переваг:

1. Дає загальний огляд конкретної проблемної області, підхід до вирішення.
2. Базується на використанні спеціальних умовних позначень, що надають стислість і наочність знань;
3. Пов'язані з вираженням певних символів для їх символів або систем, що усуває багатозначність слів у загальноприйнятих мовах;
4. Дозволяє створювати символічні моделі об'єктів, а також вивчати реальні об'єкти і процеси на заміну вивчення цих моделей.

Формалізація є невід'ємною частиною формальної логіки.

Для організації та моніторингу діяльності компанії веб-додаток буде містити веб-сторінки, що містять створений контент, контент буде додаватися користувачем і наповнюватися під час роботи з ним. Остаточний вміст полягає в тому, що відвідувач сайту заповнює необхідну інформацію, наприклад, створює профіль персоналу чи вакансії тощо. Така сторінка називається динамічною.

Працюючи в програмі, користувачі можуть отримати доступ до всієї необхідної інформації про існуючі функції та процеси, створювати та видаляти завдання, переглядайте профілі співробітників, ведіть ділові розмови або користуйтеся послугами безпосередньо в додатку. Якщо у вас є права, ви можете створювати, редагувати та видаляти профілі чи завдання співробітників, змінювати пріоритети.

Висновки до розділу 2

У даному розділі був проведений порівняльний аналіз існуючих веб-додатків для управління процесом компанії, також проаналізували стек технологій з коротким їх порівнянням, та визначили основне завдання роботи.

РОЗДІЛ 3. ПРОЕКТУВАННЯ РІШЕНЬ ДЛЯ РЕАЛІЗАЦІЇ ВЕБ-ДОДАТКУ

3.1. Концептуальна модель додатку

Концептуальна модель - це певна безліч понять і зв'язків між ними, що є смисловою структурою аналізованої предметної області. Це модель предметної області, що складається з переліку взаємопов'язаних понять, що використовуються для опису цієї області, разом з властивостями і характеристиками, класифікацією цих понять, за типами, ситуаціями, ознаками в цій галузі та законів протікання процесів у ній. Змістовна модель - це абстрактна модель, що визначає структуру моделюється системи, властивості її елементів і причинно-наслідкові зв'язки, властиві системі і суттєві для досягнення мети моделювання. Концептуальна модель веб-додатку представлена на (Рис. 3.1). [4]

Оскільки системи стають дедалі складнішими, роль концептуального моделювання різко зросла. За такої розширеної присутності реалізується ефективність концептуального моделювання при захопленні основ системи. На основі цієї реалізації, були створені численні методи концептуального моделювання. Ці методи можуть бути застосовані в різних дисциплінах для поліпшення розуміння користувачем системи, що моделюється.

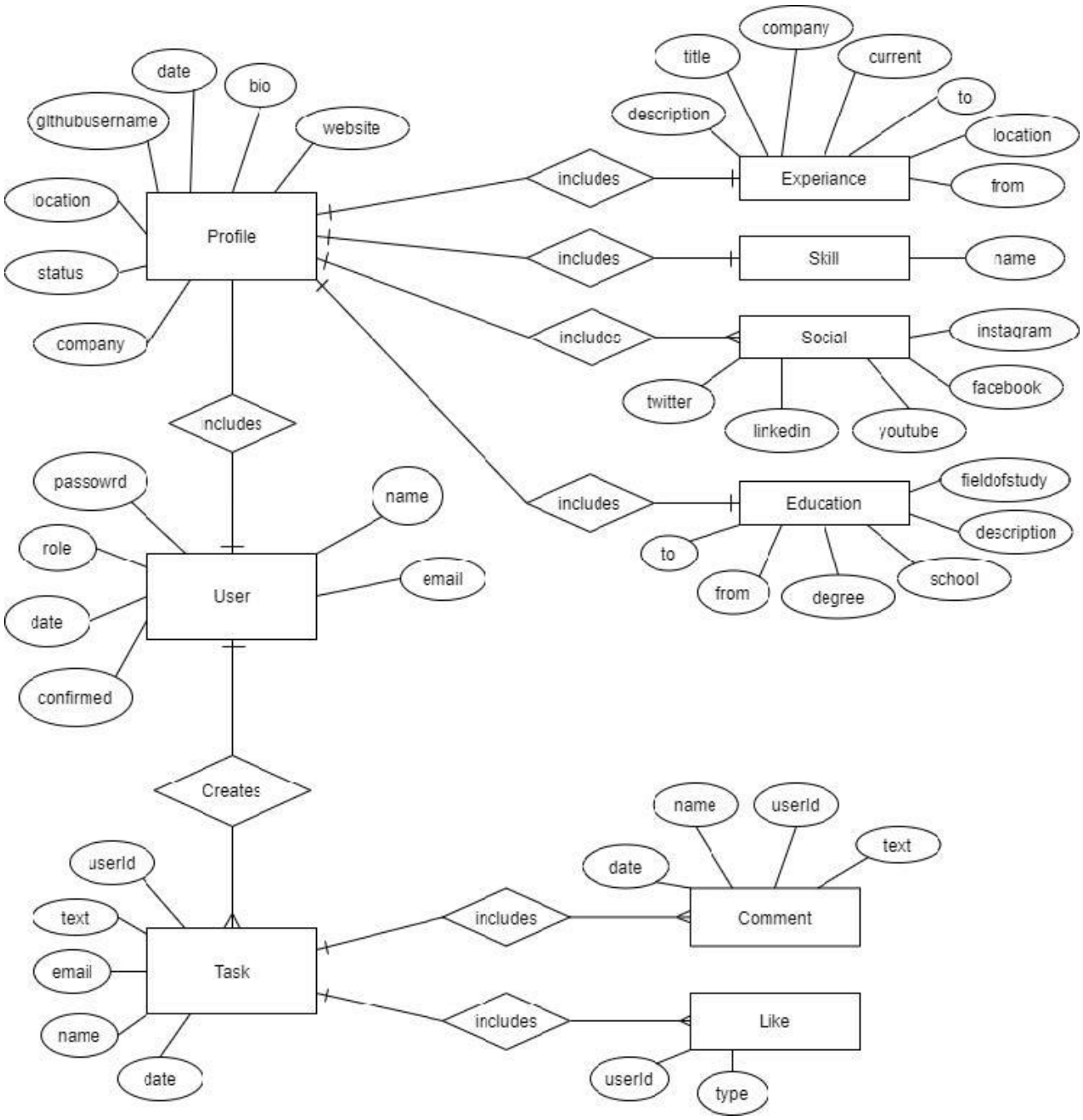

Рис. 3.1. Концептуальна модель у вигляді діаграми сутність-зв'язок

3.2. Логічна модель веб-додатку

Логічний рівень - це абстрактний погляд на дані, на ньому дані видаються так, як виглядають у реальному світі, і можуть називатися так, як вони називаються у реальному світі. За допомогою програмного пакета ERwin можна

створити логічну модель даних у вигляді юридичного зв’язку на основі критеріїв IDEF1x, а потім побудувати на її основі фізичну базу даних практично для будь-якої бази даних. Цей параметр дозволяє перейти від фізичної моделі до логічної інверсії. Це дуже цінна якість. Бази даних зараз дуже поширені і будуються в різних програмних середовищах. Використовуючи цю властивість, ви можете перетворити існуючі бази даних, вбудовані в одне середовище бази даних, у логічну модель, а потім у зовсім іншу базу даних, яку підтримує пакет ERwin. Іншими словами, існуючі бази даних можна автоматично перенести в нове середовище програмування без використання мов програмування.На схемі логічної моделі зображено:

1. Сутність та їх атрибути;
2. типи даних;
3. ключові атрибути;
4. зв'язки між сутностями;

Для побудови логічної моделі проаналізували діаграму сутність–зв'язок та з’ясували необхідні таблиці та атрибути для реалізаці. Логічна модель даних веб-додатку представлена на (Рис. 3.2).

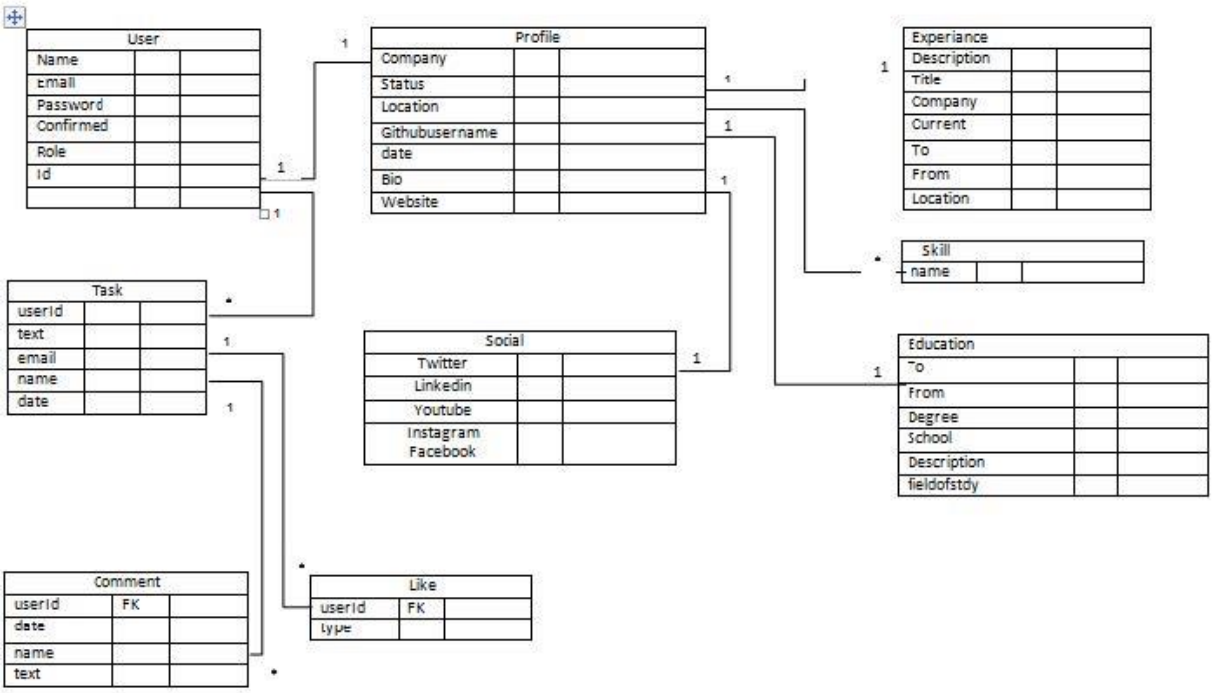

Рис. 3.2. Логічна модель данних розробляємого веб-додатку

3.3. Фізична модель веб-додатку

Фізична модель визначає розміщення даних у зовнішній пам'яті. Її також називають внутрішньою моделлю системи, а форма представлення базується на обраному СУБД. Якщо вибрано СУБД, що підтримує модель даних підключення, підключення таблиць до функцій і таблиць необхідно перенести в середовище СУБД, враховуючи вимоги до відповідного об’єкта бази даних. Тому теги (імена) таблиць і полів повинні відповідати вимогам до бази даних, типам даних, розмірам полів і обмеженням, отриманим у цій базі даних. Потім описуються стратегії індексації, а також зв’язки між таблицями, первинними та зовнішніми ключами на основі моделі даних, описаної в моделі даних, і з урахуванням методів, що використовуються у обраній СУБД. На рівні фізичного проектування особливу увагу слід приділяти перевірці точності бази даних. Цілісність даних у базі даних перевіряється обмеженнями чистоти, тобто набір правил, які

забезпечують точність даних і взаємозв’язок між ними. Наведено приклад заповненої БД з урахуванням встановлених обмежень. [6]

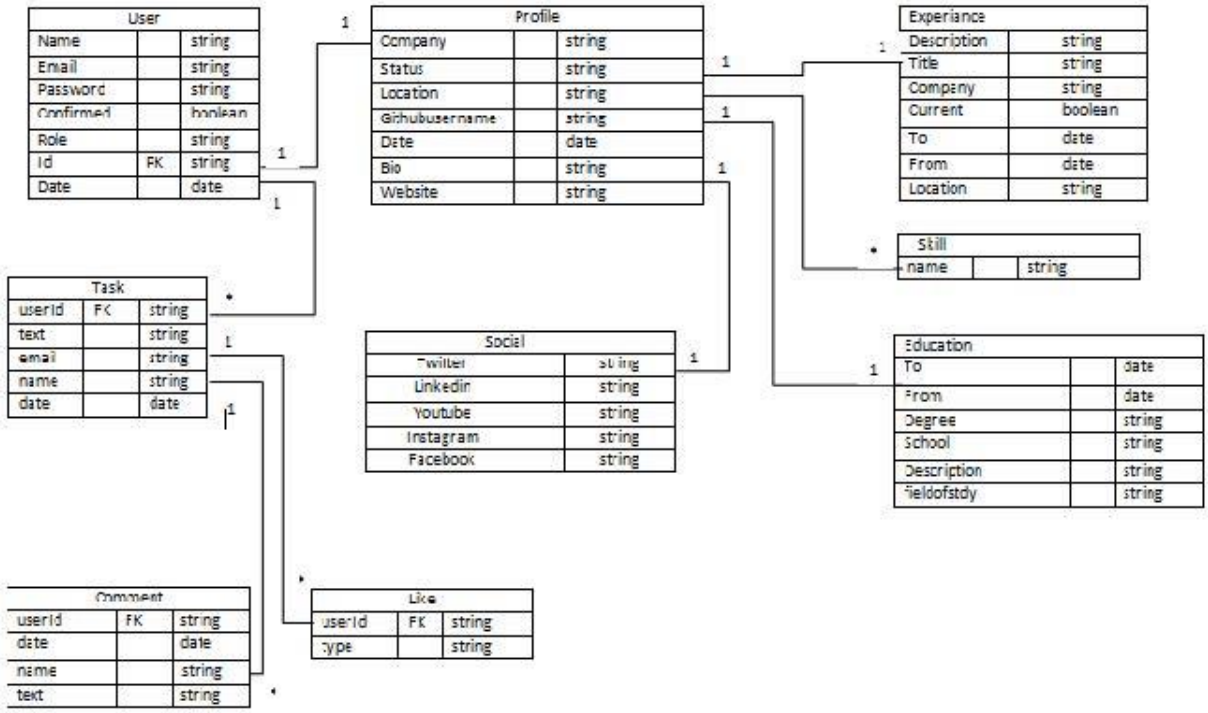

Рис. 3.3. Фізична модель данних додатку

3.4. Діаграма класів

Діаграма класів - структурна діаграма мови моделювання UML, що демонструє загальну структуру ієрархії класів системи, їх кооперацій, атрибутів (полів), методів, інтерфейсів та взаємозв'язків між ними. Широко застосовується не тільки для документування та візуалізації, але також для конструювання за допомогою прямого чи зворотного проектування.

Таблиця виглядає як символи класів, які називаються значками та зв’язками між ними. Значок слова вказує на стандартизовану, фіксовану форму, легке для розуміння візуальне зображення певного поняття (скажімо, ієрогліф). Значок класу має прямокутну форму, яку можна розділити на дві або три частини.

Верхній обов’язковий, містить ім’я класу. Другу і третю частини прямокутника можна задати або залишити і можна утримувати: друга - список функцій кімнати, третя - список функцій кімнати.

Існує два види діаграм класів:

1) Статичний вид діаграми розглядає логічні взаємозв'язки класів між собою;
2) Аналітичний вид діаграми розглядає загальний вигляд і взаємозв'язку класів, що входять в систему.

Існують різні точки зору на побудову діаграм класів в залежності від цілей їх застосування:

1) Концептуальна точка зору – діаграма класів описує модель предметної області, в ній присутні тільки класи прикладних об'єктів;
2) Точка зору специфікації – діаграма класів застосовується при проектуванні інформаційних систем;
3) Точка зору реалізації – діаграма класів містить класи, використовувані безпосередньо в програмному коді (при використанні об'єктно–орієнтованих мов програмування).

Діаграма класів є одним із найбільш послідовних виразів системи з точки зору показу структури системи. Діаграма класів не показує динамічний характер навчальних кімнат, описаних на ній. Діаграми класів показують розділи, діалоги та зв’язки між ними. Схема компонентів програми представлена на (рисунок 3.4).

Діаграма класів розроблена відповідно до вимог середовища RationalRose.

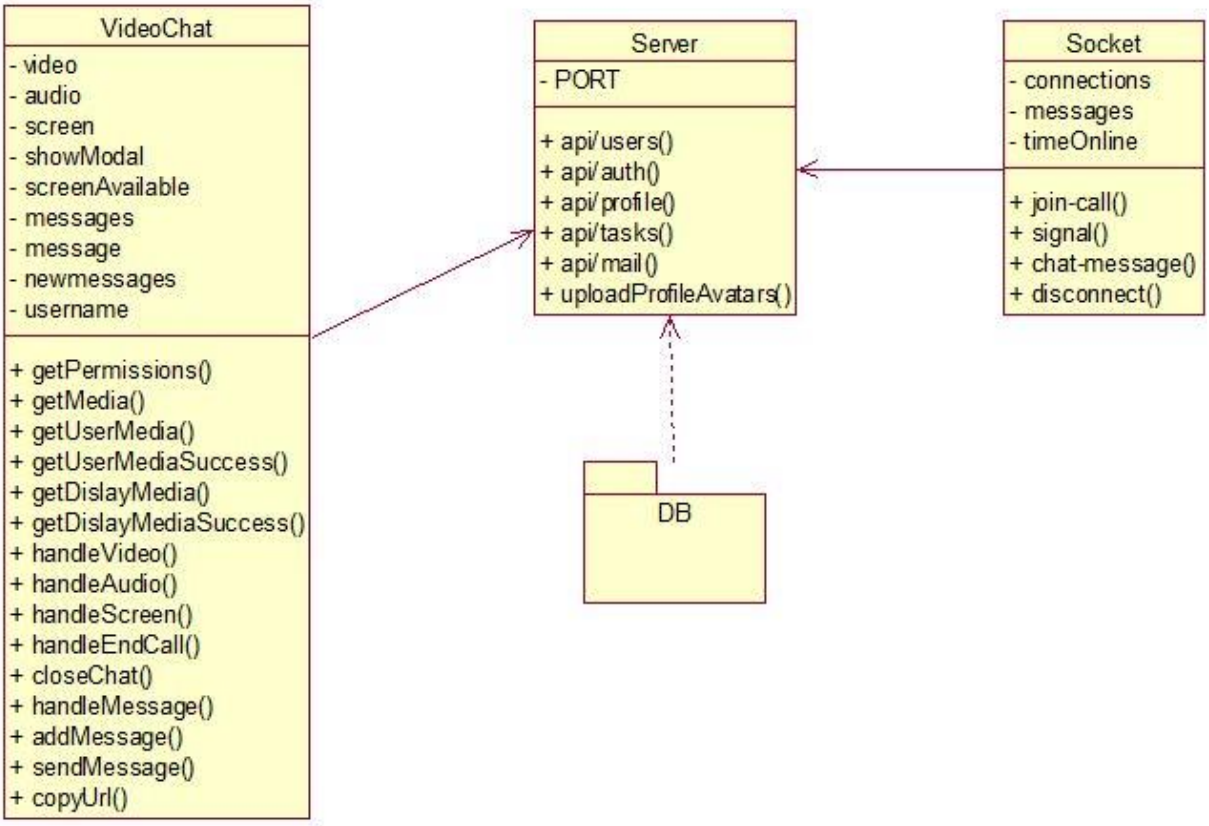

Рис. 3.4. Діаграма класів додатку

3.5. Діаграма послідовності

Діаграма послідовності відображає взаємодії об'єктів впорядкованих за часом. Зокрема, такі діаграми відображають задіяні об'єкти та послідовність відправлених повідомлень. Діаграми послідовностей – це відмінний засіб документування поведінки системи, деталізації логіки сценаріїв використання, але є ще один спосіб – використовувати діаграми взаємодії. Діаграма взаємодії показує потік повідомлень між об'єктами системи і основні асоціації між ними і по суті, як вже було сказано вище, є альтернативою діаграми послідовностей. Слід зазначити, що використання діаграми послідовностей або діаграми взаємодії – особистий вибір кожного проектувальника і залежить від індивідуального стилю проектування. [7]

Щодо приміток, використаних у діаграмі зв’язку, то на їх привід не варто зволикати увагу. Тут все нормально: речі позначаються прямокутниками (щоб відокремити їх від класу), зв’язки між об’єктами з’являються у вигляді ліній зв’язку, а над ними може бути стрілка, що вказує назву повідомлення та порядковий номер. Важливість номера повідомлення пояснюється дуже просто - на відміну від діаграми послідовності, час, витрачений на діаграму зв'язку, не розглядається як окремий параметр.

На рисунку 3.5 показана послідовність варіантів створення профілю для співробітника.

Діаграма послідовності розроблена відповідно до вимог середовища RationalRose.

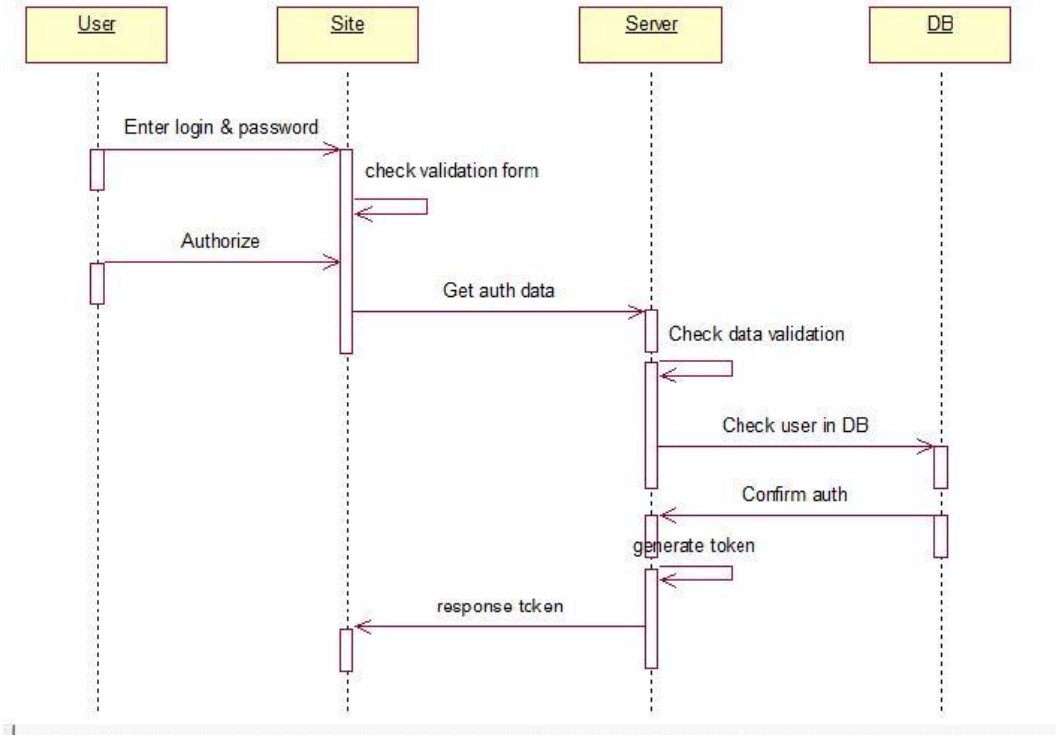

Рис. 3.5. Діаграма послідовності для варіанту використання створення профілю співробітника

3.6. Рішення по шифровці даних

Шифрування даних – це метод захисту, в якому інформація кодується і може бути відновлена або розшифрована лише користувачем з правильним ключем дешифрування. Зашифровані дані, також відомі як шифротекст, зашифровані або нечитані неавторизованими особами або організаціями. Шифрування даних використовується для запобігання розголошенню конфіденційної інформації зловмисниками або небажаними людьми. В архітектурі кібербезпеки необхідна лінія захисту – шифрування – робить хакерську інформацію максимально складною. Програмне забезпечення для шифрування, також відоме як алгоритм шифрування або шифр, теоретично використовується для розробки плану шифрування, доступ до якого може отримати лише потужний комп'ютер. Шифрування використовується для зберігання важливої інформації із безпечних джерел і для її передачі небезпечними засобами. Такі обміни є двома зворотними процесами: перед відправкою або зберіганням інформації в режимі онлайн її можна зашифрувати. Хоча інформацію все ще можна зламати, вона заплутана і тому не корисна для шпигунів чи хакерів. Ключ шифрування можна порівняти зі стандартним паролем, наприклад, який використовується для електронної пошти. Ключ є ключем до кодування та дешифрування даних. Ключі шифрування створюються абсолютно унікальним способом. Ключ шифрування використовується для шифрування або розділення даних. Тобто ключ шифрування може змішувати інформацію в нечитані символи і відновлювати ці символи до чіткості.

Для системи нашого додатку був обраний асиметричний алгоритм RSA завдяки його простоті, надійності та легкої реалізації.

На (Рис. 3.6) представлена схема шифрування та дешифрування ключа асиметричним алгоритмом.

Рис. 3.6. Схема шифрування асиметричним алгоритом

Найбільша перевага використання асинхронних алгоритмів полягає в тому, що вам не потрібно надсилати або пересилати ключ шифрування через шкідливий канал. Це зменшує ймовірність пограбування.

Для генерації пари ключів потрібно виконати наступне:

1) обрати два великі прості числа p та q, кожне довжиною в 512б;

2) обчислити їх добуток за формулою $n=pq$;

3) обчислити функція Ейлера $\varphi(n)= (p–1)(q–1)$;

4) обрати ціле число e, таке, що $1<e< \varphi(n)$ та e , взаємно просте з $\varphi(n)$;

5) знайти число d, таке, що $ed=1(mod\ \varphi(n))$ за допомогою алгоритму Евкліда.

Число n називається модулем, а числа e і d– відкритою і секретною експонентами відповідно. Пари чисел (n,e) називаються відкритою частиною ключа, а (n, d) – секретною. Числа p і q не будуть більше використовуватися після

генерації пари ключів і можуть бути знищені, але ні в якому випадку не не повинні буди розсекречені.

Алгоритм шифрації данних:

1) взяти відкритий ключ (n,e) та повідомлення m;
2) зашифрувати повідомлення $c= E(m)=m_e \, mod \, n$.

Алгоритм дешифрації данних:

1) взяти закритий ключ (n, d) та повідомлення m;
2) розшифрувати повідомлення $m=D(c)=c_d \, mod \, n$.

Було вирішено, що новий ключ користувача буде шифруватися в базі даних щоразу, коли програма дозволяє користувачам безпечно спілкуватися під час відеоконференції, не турбуючись про ключ. Згенерований ключ буде скопійовано іншим співробітникам або співробітник отримає ключ і відправить його на приватну конференцію з розробником-відправником ключа. Ключ для шифрування та дешифрування цих ключів не зберігається на сервері, вони також не надсилаються на сервер, а шифрування-дешифрування відбувається безпосередньо на комп'ютерах користувачів.

3.7. Загальносистемні рішення

3.7.1. Представлення схеми організаційної структури

Додаток для організації та контролю діяльності компанії не ділиться за правами доступу, усі користувачі мають однакові права. Щоб почати працювати з колегами в додатку, новий співробітник повинен спочатку увійти в систему, після чого адміністратор знаходить нового користувача в базі даних і перевіряє профіль, цей крок додається в цілях безпеки. Адже зареєстрований користувач з непідтвердженим профілем не має можливості спілкуватися з робочими радами компанії, профілями співробітників і. Співробітники компанії є основними та

основними користувачами програми, менеджер додатків не є безпосереднім користувачем, лише для ідентифікації користувачів та доступу до програми, наданої на (рисунок 3.7).

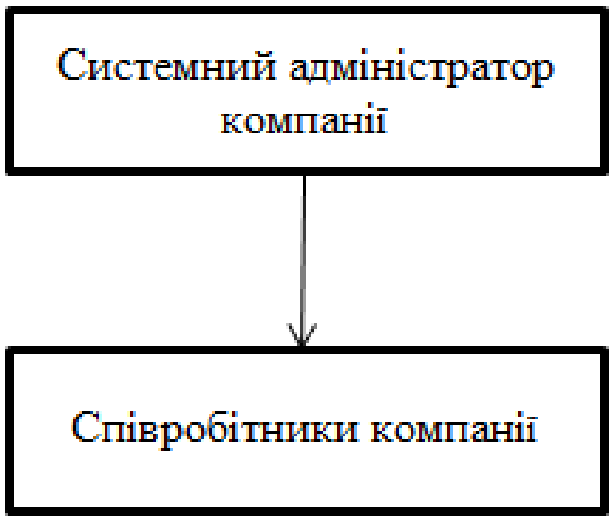

Рис. 3.7. Схема організаційної структури

3.7.2. Представлення схеми функціональної структури

Функціональна схема – це схема взаємодії компонентів програмного забезпечення з описом інформаційних потоків, складу даних в потоках. Функціональні схеми більш інформативні, ніж структурні. На (Рис. 3.8) представлена схема функціональної структури для додатку для організації та контролю робочого процесу компанії.[ ]

Рис. 3.8. Схема функціональної структури

3.7.3. Описання функцій, що автоматизуються

Додаток призначений для автоматизації ручної роботи та оптимізації процесу спілкування з віддаленим робочим процесом компанії. Наступні процеси є автоматичними:

1) процес ідентифікації працівників;
2) процес створення та призначення завдань працівникам;
3) процес моніторингу діяльності;
4) додаткова можливість здійснювати дзвінки та торгувати в додатку .
5) Процес спілкування співробітників компанії;
6) Процес ведення кадрової бази.

3.7.4. Загальний опис системи

Додаток розроблено для компаній, насамперед IT-індустрії. Програма дозволяє робити наступне:

1) додавати, редагувати та видаляти інформацію;
2) створювати, переглядати, коментувати та видаляти вакансії;
3) створювати та підтримувати профіль співробітника, переглядати та видаляти інформацію;
4) Перегляд інформації про компанію та її співробітників;
5) Здійснюйте онлайн-дзвінки та конференції, використовуючи високе шифрування даних.

Для роботи в додатку створюється профіль менеджера з безпеки компанії, який є системним адміністратором або менеджером. Спочатку працівник реєструється, перевіряється профіль, а потім він може ввійти і працювати зі сторінкою, на панелі завдань, на сторінці компанії (на певних правах) і в конференц-залі.

3.7.5. Рішення з організаційного забезпечення

Графічний інтерфейс користувача – тип інтерфейсу, який дозволяє користувачам взаємодіяти з електронними пристроями через графічні зображення та візуальні вказівки, на відміну від текстових інтерфейсів, заснованих на використанні тексту, текстовому наборі команд та текстовій навігатор. [8]

Цей термін використовується в багатьох галузях науки і техніки. Мається на увазі з'єднання всіх компонентів будь-якої взаємодії (як природних, так і апаратних, і людино-машинних). Залежно від контексту, концепція застосовується як до одноелементного інтерфейсу, так і до інтерфейсу

компонентного інтерфейсу. Користувачеві елемента не потрібно знати, як керувати використаним елементом, але використаний елемент повинен забезпечувати інтерфейс керування.

Інтерфейс програми повинен бути лаконічним і зрозумілим, щоб користувач міг швидко знайти необхідну інформацію, сторінку чи форму та легко з нею працювати. Інтерфейс має бути у вигляді сторінок з формами та полями.

3.8. Описання використаних технічних засобів для розробки

При розробці додатку для організації і контролю робочого процесу компанії використовувались наступні технічні засоби:

1) JavaScript – високорівнева мова веб програмування;
2) Фреймворк React JS;
3) MongoDB – СКБД;
4) Visual Studio Code – середа розробки та підключення додатків на мові JavaScript;
5) XML – мова розмітки що була використана (є офіційним стандартом) для розробки візуальної складової додатку (дизайну).

Висновки до розділу 3

У даному розділі представлені схеми організаційної та функціональної структури. Описані функції, що автоматизувалися у розробці. Були спроектовані та побудовані: модель варіантів використання та концептуальна модель, логічна і фізична моделі, модель переходів станів, діаграми послідовності та класів..

РОЗДІЛ 4. ВИПРОБУВАННЯ ТА ДЕМОНСТРАЦІЯ ВЕБ ДОДАТКУ ДЛЯ КОНТРОЛЮ ТА ОРГАНІЗАЦІЇ РОБОЧОГО ПРОЦЕСУ

4.1. Загальні положення випробування веб-додатку

Тестування розробленого додатку проводилось за методикою «чорної скриньки» - це метод тестування програмного забезпечення, що фокусується на аналізі функціональності програмного забезпечення, а не на внутрішніх системних механізмах. Тестування чорної скриньки було розроблено як метод аналізу вимог, специфікацій та стратегій проектування високого рівня клієнта. Тестувальник програмного забезпечення чорної скриньки вибере дійсний та недійсний вхідний набір та умови виконання коду, а також перевірить правильні вихідні відповіді. [5]

Об’єктом тестування є веб–додаток для управління робочим процесом підприємтсва, в нашому випадку на прикладі ІТ компанії.

Метою тестування буде перевірити програмне забезпечення на наявність багів інтерфейсу, критичних помилок та відповідність функціоналу зазначеного у вимогах до програми.

Вимогами до програмного забезпечення являється відповідність до функціоналу зазначеного у розділі «Вимоги до функціональних характеристик». При проведенні тестування будуть розглядатись всі реалізовані функції додатку, а саме:

1) Функції «Реєстрації та авторизації».

2) Функція «Додання інформації у формах персональної інформації, досвід роботи та освіта».

3) Функція «Створення, перегляд, видалення задач».

4) Функція «Створення приватного дзінка».

Вимоги до програмної документації буде документація яка пропонуються до тестування програмного забезпечення, а саме:

– завдання та вимоги до розробки;

– опис програмного забезпечення;

– текст програмного забезпечення;

– програму та методику випробувань;

– інструкцію користувача.

До засобів необхідних для тестування веб-додатку відносяться наступні технічні залежності:

- IBM–сумісний комп'ютер з тактовою частотою процесора 1.2 МГц;
- Операційна система Microsoft Windows 7 та вище»;
- Обсяг оперативної пам'яті 512 МБ;
- 200 МБ простору на жорсткому диску;
- Монітор;
- Клавіатура;
- Миша;

4.2. Функціональне тестування

Тестування функцій «Реєстрації та входу до системи»

Для того щоб створити нового користувача потрібно в меню обрати вкладку «Sing Up», після чого відкриється форма реєстрації де потрібно

заповнити всі поля, прийняти політику конфіденційності та підтвердити натиснувши відповідну кнопку. Даний функціонал можна вважати реалізованим якщо:

1. Коректно відображається та працює посилання на політику конфіденційності, при натисканні розгортується сторінка з правилами конфіденційності;
2. Система перевірила дані кожного поля на дублікат та у відсутності таких проінформувала про це користувача надписом зеленого кольору, у випадку збігання видалила дані поля та проінформувала червоним кольором про це;
3. Система перевірила користувача чи не заблокований він в системі;
4. Кожне поле проходить правила валідації, серед яких максимальна та мінімальна довжина, діапазон допустимих символів, спецсимволи та обов'язковість заповнення;

Після того як система проінформує про підтвердження профілю потрібно буде зайти до особистого профілю через розділ авторизації натиснувши кнопку «Login». Для цього розділу специфіка перевірки реалізації така сама як і для авторизації, тому функцію входу можна важати реалізованою якщо дотримані правила описані вище. Далі як всі необхідні процедури зроблено без помилок ми потрапляємо на персональну сторінку.

Тестування функції «Особистої інформації»

На сторінці «Your dashboard» (персональна інформація) обираємо доступний розділ інформації про користувача, досвід роботи чи освіти, далі відобразиться відповідна форма для внесення інформації, після заповнення всіх полів додаємо інформацію до розділу. Форми для внесення інформації враховуються реалізованими якщо всі заповнені поля перенесли дані до відповідного поля розділу особистої інформації, форма заповнення фотографії

підтягує вибір розділу на персональному комп'ютері та відображається обрана фотографія. Сам розділ рахується реалізованим коли можливий запуск повторного виклику форм для зміни даних, та оновлення самих даних, працездатні кнопки видалення акаунту та конкретних полів.

Тестування функціоналу «Створення, перегляду та видалення задач»

Після того як увійшли до системи натискаємо кнопку «Tasks», нас переправляє на сторінку де розташовані всі задачі, даний розділ вважатиметься реалізованим коли:

1. При створенні нової задачі вона відображається з права в колонці;
2. При натисканні кнопки «Discussion» викликається чат для обговорювання завдання;
3. При натискані кнопки з червоним хрестиком вона видаляється та помічається як виконана для особи яка ставила завдання;
4. При заповненні форми та натискання кнопки «submit» публікуэться завдання;

Тестування функції «Створення приватного дзінка»

У додатку натиснути кнопку «VideoChat», після цього нас переправить до розділу відео–конференцій. За допомогою кнопки «Copy invite link» ми отримаємо адресу необхідну для підключення відеодзвінка, надсилаємо її адресанту за допомогою кнопки «Invite», тоді ми автоматично перейдемо до вікна бесіди з обраним користувачем. При успішній роботі функції всі кнопки повинні функціонувати, адресант id-адреси після натискання на неї повинен перейти у режим дзвінку з організатором, у формі веб-додатку повинно відображатись вікно видео передачі з онлайн трансляціею та наявність аудіозвуку.

Була проведена загальна перевірка зручності використання яка передбачала перевірку:

1. Орфографічних та граматичних помилок, всі сторінки мають коректні заголовки;
2. Вирівнювання картинок, шрифтів, текстів;
3. Відступи між полями, колонками, рядами та повідомленнями про помилки.
4. Кнопки мають стандартний розмір, колір.
5. На сайті немає битих посилань та зображень.
6. Неактивні поля відображаються сірим кольором.
7. Скролл повинен з'являтися лише тоді, коли він потрібний.
8. Переходи та навігація між сторінками та розділами меню.

4.3. Нефункціональне тестування

Перевірка безпеки - ця перевірка націлена на пошук недоліків та пробілів з точки зору безпеки веб-додатку, було перевірено наступні пункти:

1. Користувач не може авторизуватись під старим паролем чи якщо заблокований у сервісі.
2. Пароль прихований астерисками на сторінках: реєстрація, забули пароль, зміна пароля.
3. Коректне відображення повідомлень про помилки.
4. Завершення сесії після розлогіну.
5. Доступ до закритих розділів додатку.
6. Ролі користувачів та доступ до контенту.

Провели крос-платформне тестування щоб переконатися, що додаток сумісний з іншими браузерами, різними оболонками чи апаратним забезпеченням пристрою. Деталі викладені у пунктах:

1. Портували в різних браузерах таких як Firefox, Chrome, Opera, перевірили анімацію, верстку, шрифти, сповіщення та ін.

2. Перевиріли у різних версіях ОС: Windows, Mac, Linux.
3. Java Script код коректно відпрацьовує у різних браузерах.
4. Перегляд з мобільного пристрою здійснюється без нарікань.

Для проведення навантажувального тестування використовувався ресурс LoadStorm де настроїли сценарій користувача та запустили на перевірку 50-ти потоків одночасно, система без зайвих проблем витримала до 3-х запитів на секунду, а середній час відгук сервера становив близько 0.3 секунди.

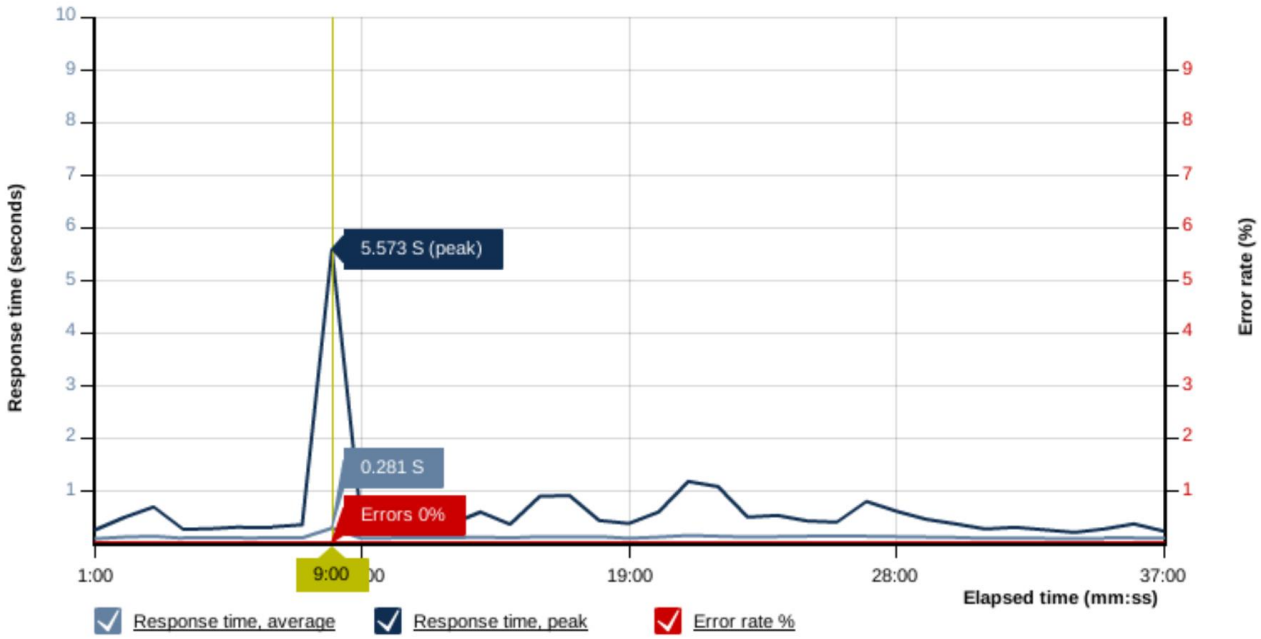

Рис. 4.1. Графік відгуку сервера

4.4. Огляд розробленого веб-додатку

Розроблений програмний продукт в процесі написання дипломної роботи складається з розділів, в яких міститься певний функціонал необхідний для покращення взаємодії між співробітниками підприємства.

Для початку роботи с додатком потрібно його запустити, після нетривалого очікування користувача перекине на головну сторінку веб-додатку (Рис. 4.2), на

якій відображено його назву, навігаційне меню та форми реєстрації та входу до особистого кабінету.

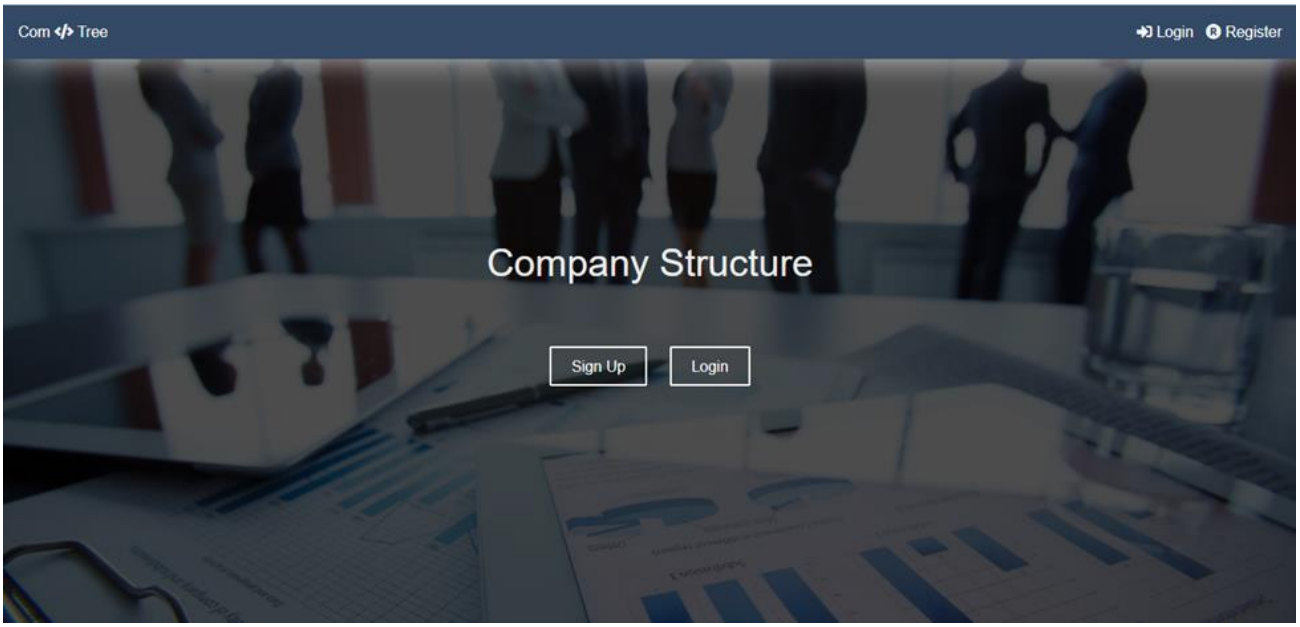

Рис. 4.2. Відображення головної сторінки

Далі потрібно зайти до особистого профілю, або зареєструватися якщо користувач новий. У вікні реєстрації (Рис. 4.3) потрібно ввести свої дані, у даному вікні налаштована можливість швидкої реєстрації за допомогою особистого профілю на Google або Facebook.

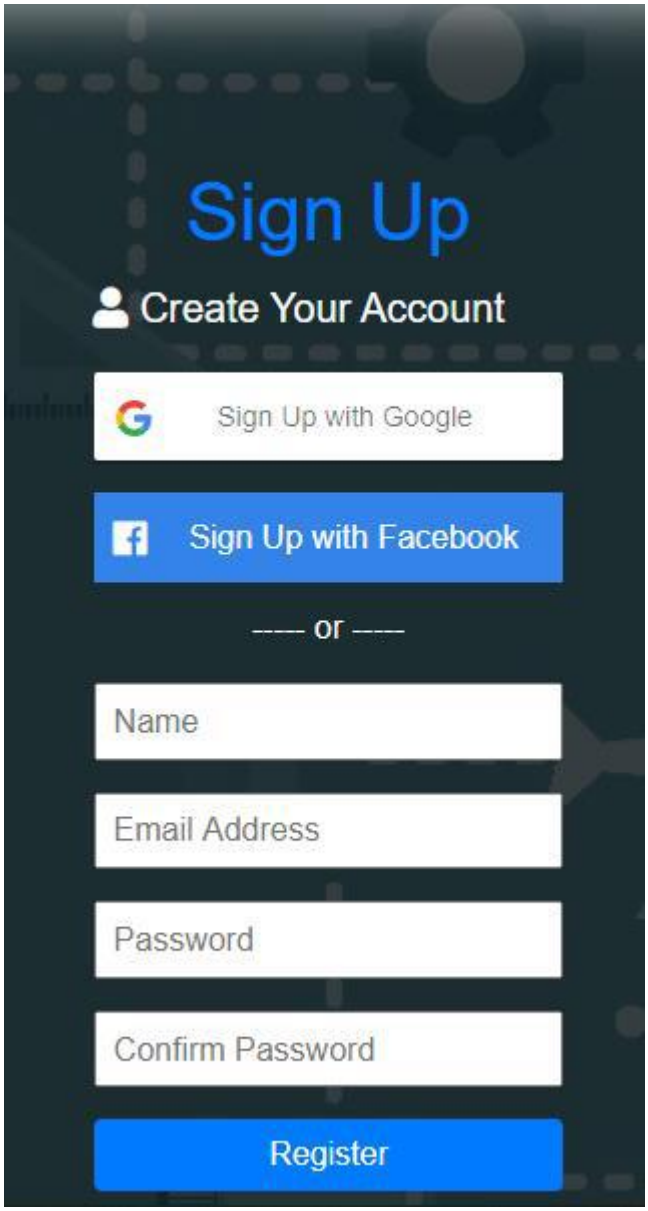

Рис. 4.3. Форма створення особистого профілю

Далі потрібно увійти до системи (Рис. 4.4) за допомогою данних створених у попередній формі, або також скористатися функцією швидкого підтягування данних з проіллюстрованих профілів.

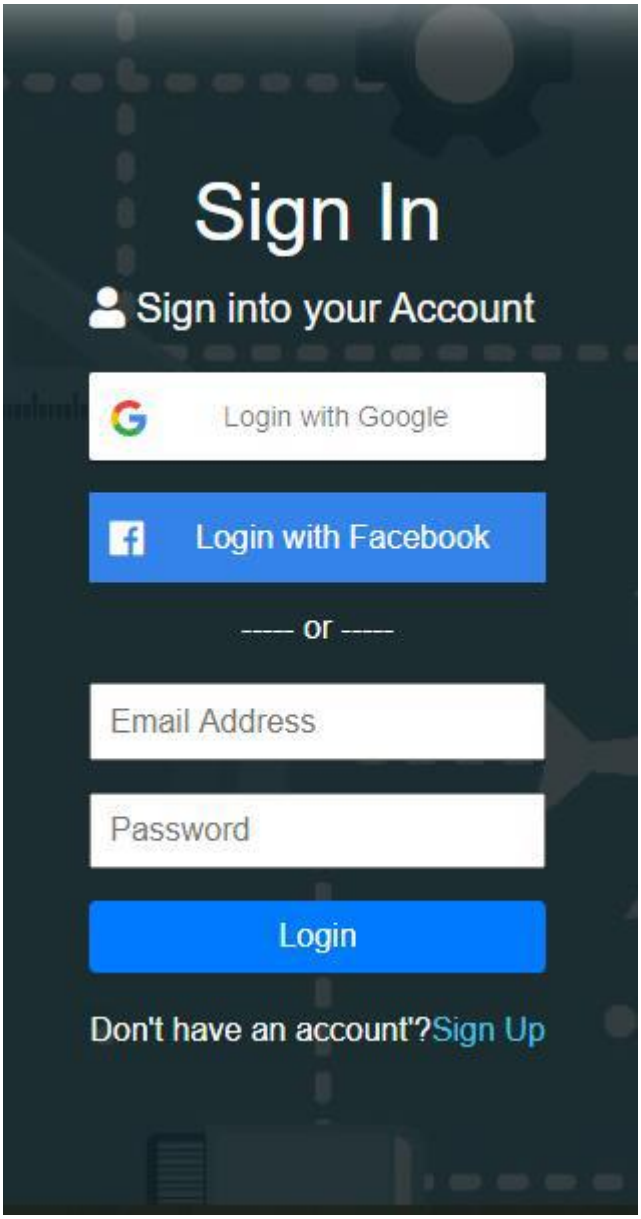

Рис. 4.4. Вікно входу до системи.

Після успішного проходження реєстрації та авторизації користувача переправляє на сторінку особистого профілю (Рис. 4.5).

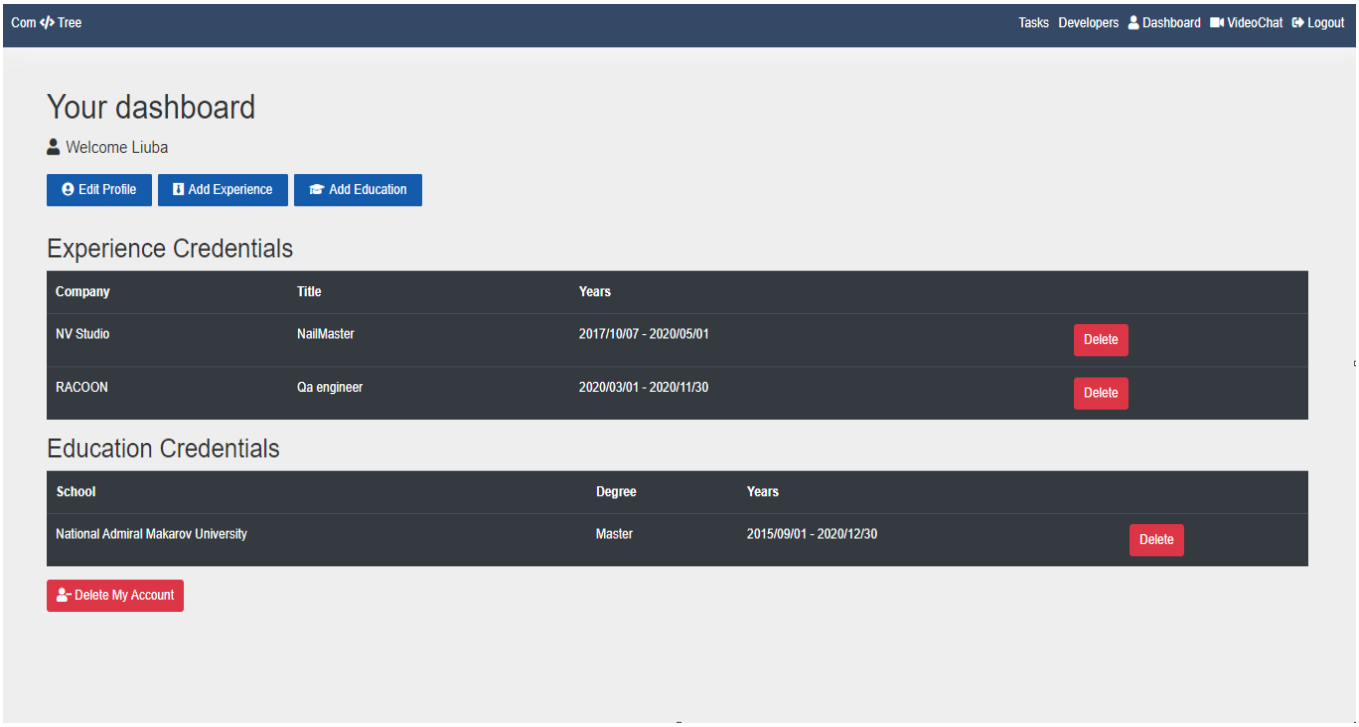

Рис. 4.5. Сторінка особистого профілю користувача

На сторінці профілю доступні розділи персональної інформації які необхідно заповнити при створенні нового користувача, доступно також редагування та видалення як інформації так і самого профілю. Після натискання «Edit Profile» з’являться форми для внесення інформації (Рис. 4.6 – 4.9).

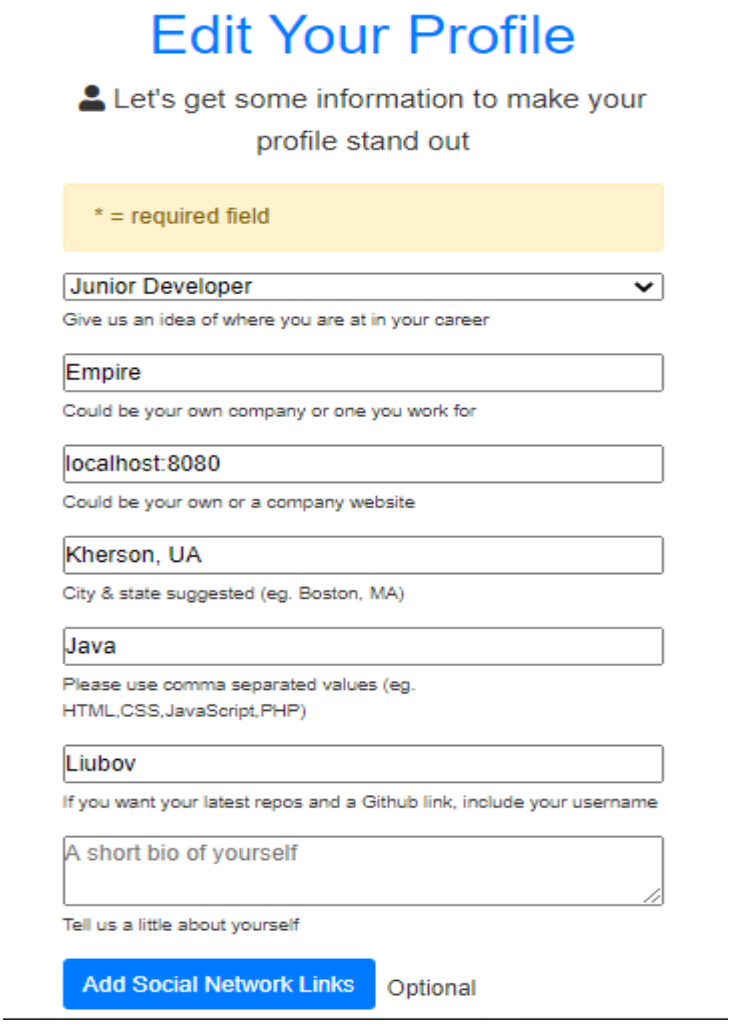

Рис. 4.6. Форма внесення персональної інформації

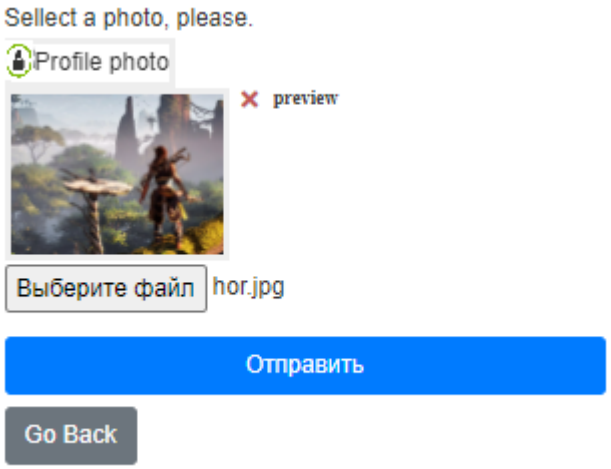

Рис. 4.7. Форма редагування фото профілю

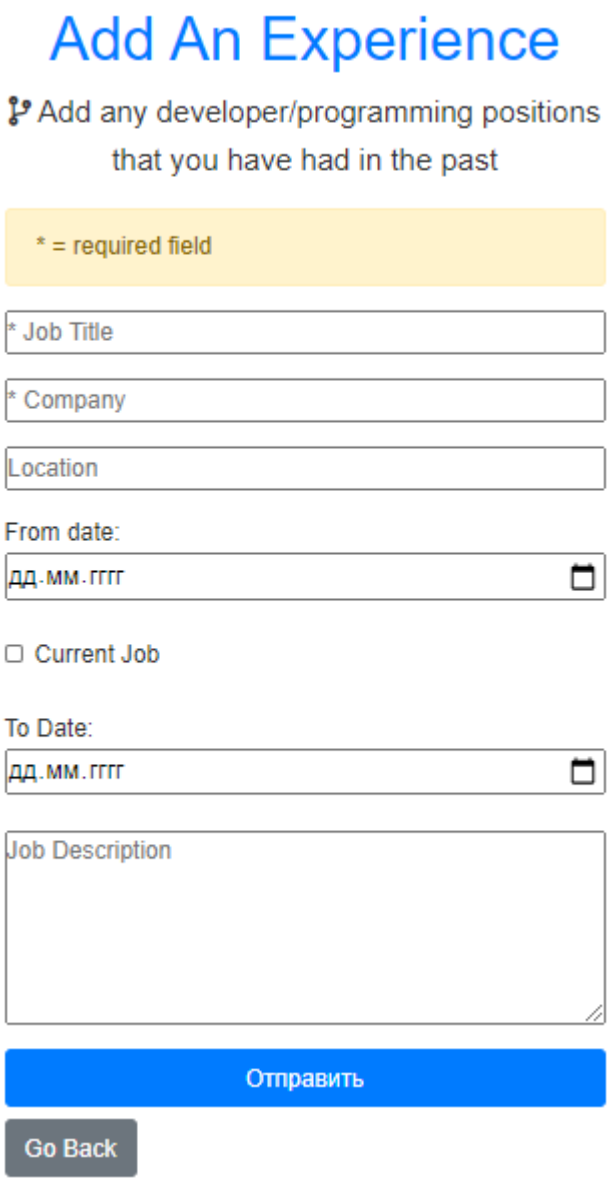

Рис. 4.8. Форма внесення даних про досвід роботи

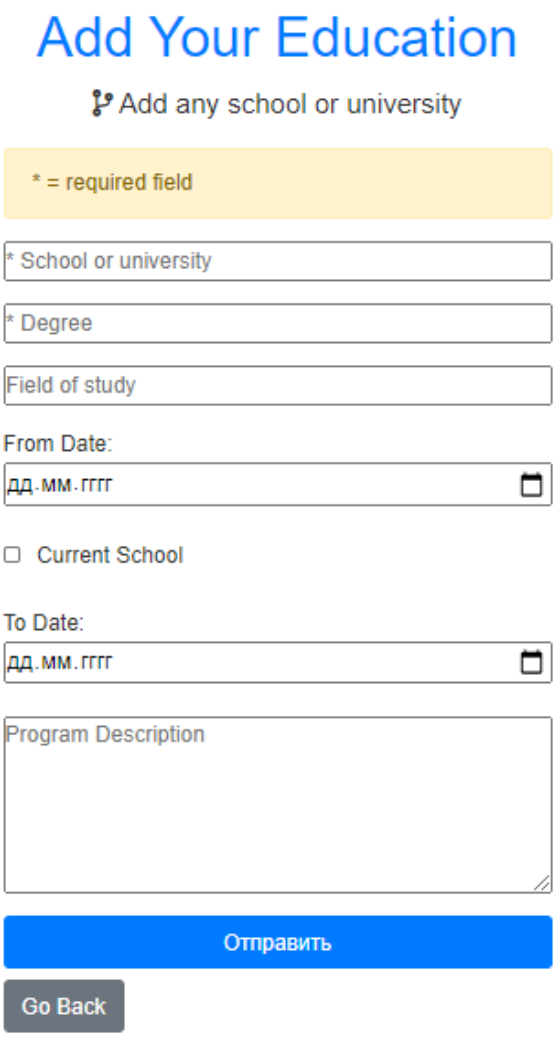

Рис. 4.9. Форма внесення даних про освіту

Головним розділом додатку є сторінка «Tasks» (Рис. 4.10. та 4.11.), де розміщені пости з завданнями, на яких вказано ім’я, дату, коротку інформацію та коментарі. Також у цьому розділі розташований інструмент для розміщення самих постів з завданням.

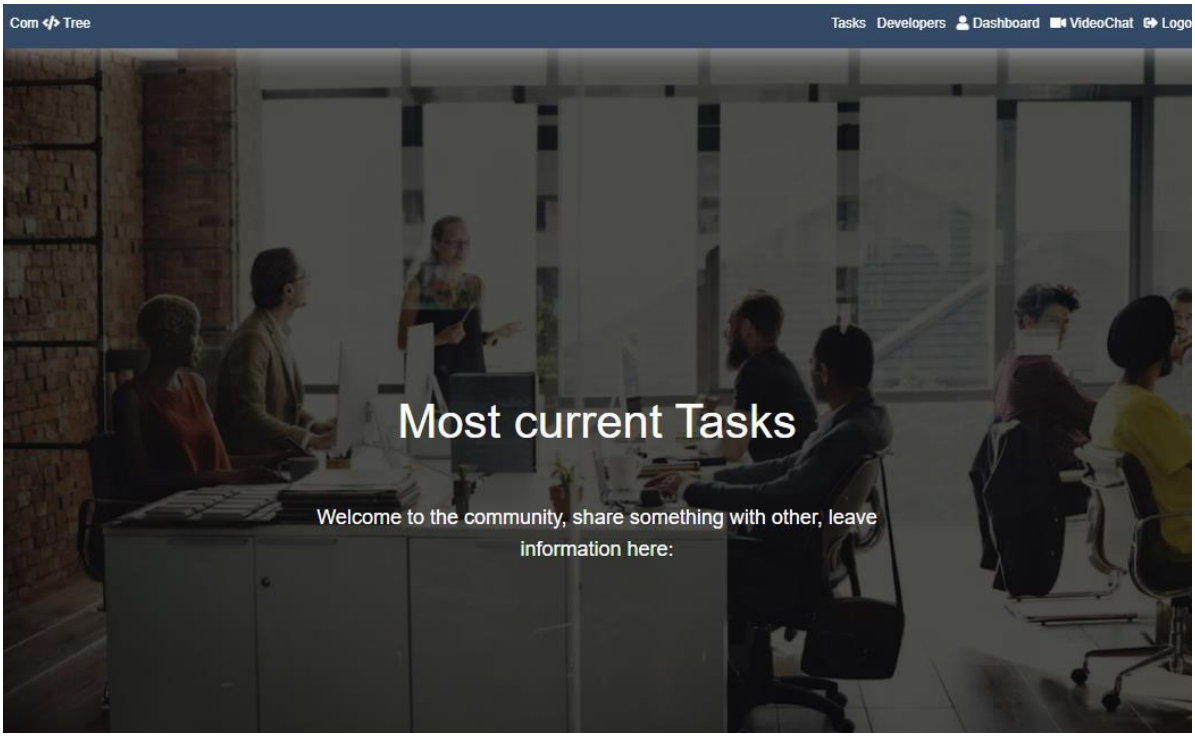

Рис. 4.10. Дошка задач 1

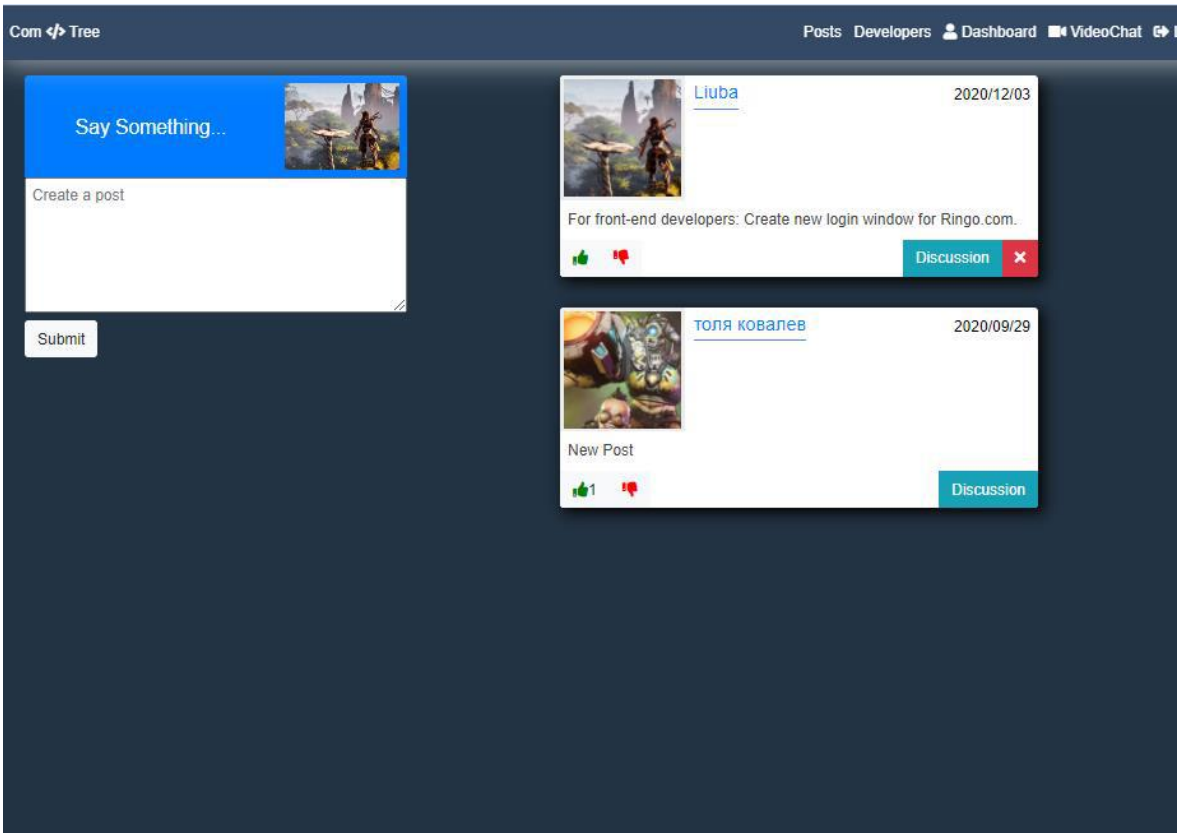

Рис. 4.11. Дошка задач 2

Також у доступі є можливість перегляду детальної інформації про компанію та інших співробітників на сторінці «Developers» (Рис. 12).

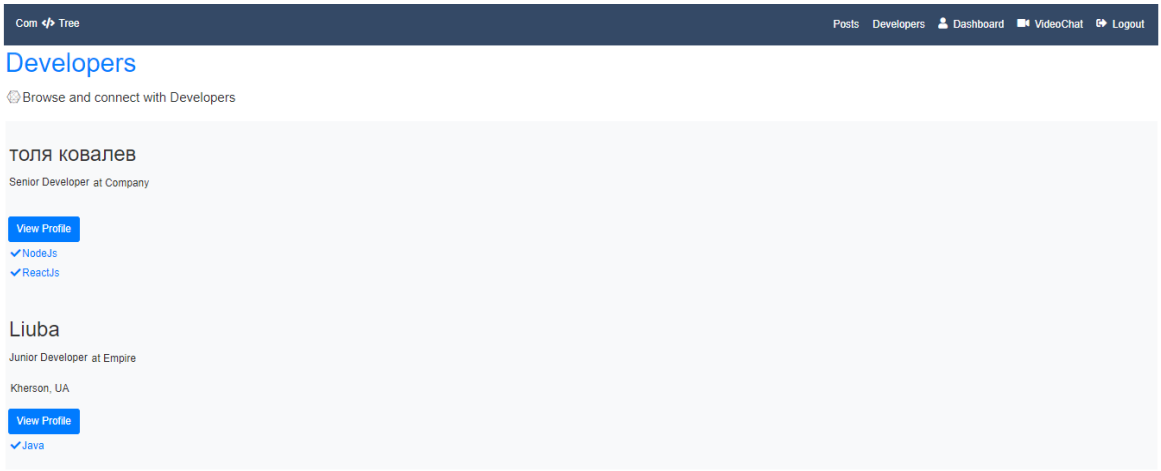

Рис. 4.12. Сторінка компанії

У додатку влаштована функція відео-конференцій у розділі «VideoChat», в ній також присутня можливість приватних бесід.

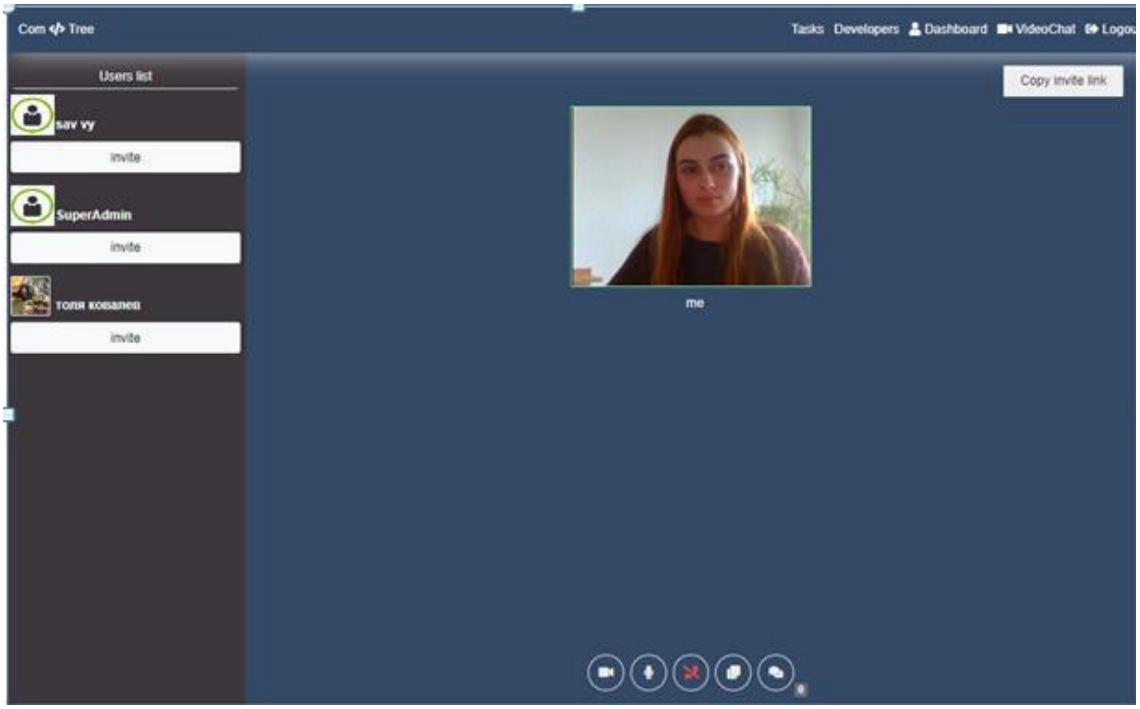

Рис. 4.13. Розділ відео–конференцій

Для завершення роботи користувач може вийти з профілю за допомогою кнопки «Logout».

Висновки до розділу 4

У даному розділі було проведено функціональне та нефункціональне тестування додатку та висвітлено огляд створеного функціоналу.

ВИСНОВКИ

У ході виконання дипломного проекту були досліджені та випробувані методи створення Web-додатку для контролю робочого процесу підприємства. Проведено дослідження ефективності використання різних технологій для реалізації функціонального і готового до експлуатації Web-додатку. Для цього було проаналізовано переваги програмного забезпечення, редакторів вихідного коду, фреймворків та засобів, які допомагають полегшити розробку.

Також були розглянуті питання необхідності такої системи в наші часи, описано важливість покращення середовища для робітників та своєчасного виявлення проблем в компанії, що в свою чергу дозволить покращити виробництво підприємства.

Розглянуті та використані технології показали, що розумний підбір технологій для розробки проекту може полегшити процес розробки та навіть зробити його цікавим.

В результаті дипломної роботи було отримано працездатний web-додаток, який можна використовувати як систему керування робочим процесом компанії.

Незаперечно цей проект буде і надалі розвиватись, в планах розвивати готовий функціонал, додавати новий, робити можливість інтегрування проекту у інші, покращувати зовнішній вид та зручність використання.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Д. Приемы обьектно–ориентированного проектирования. Паттерны проектирования. – СПб: Питер, 2010. – 366с.
2. Фаулер, Мартин. Архитектура корпоративных программных приложений.: Пер. с англ. – М.: Издательский дом "Вильяме", 2006. – 544с.
3. Руководство Microsoft по проектированию архитектуры приложений. 2-е издание. – Корпорация «Майкрософт», 2009. – 529с.
4. Брауде Е. Технология разработки программного обеспечения / Е. Брауде. – СПб. :Изд-во "Питер", 2004. – 655с.
5. Лаврищева К.М. Програмна інженерія / Лаврищева К.М. –К: Академперіодика, 2008. – 319с.
6. Патерни проектування. – режим доступу: http://design-pattern.ru/
7. Патерни проектування.: – режим доступу: http://cpp-reference.ru/patterns/
8. Харченко О.Г. Інструментальний засіб порівняльного оцінювання і багатокритеріального вибору архітектури програмних систем / О.Г.Харченко, І.О.Боднарчук, І.Е.Райчев, І.О.Галай // Інженерія програмного забезпечення. –2015. –№1(21). – С. 10–24.
9. Харченко О.Г. Службовий твір Комп’ютерна програма “Архітектор програмних систем” / О.Г.Харченко, І.Е.Райчев, О.А.Щербак, Б.С.Павленко, І.О.Боднарчук // Свідоцтво про реєстрацію авторського права на твір №59631. Видане державною службою інтелектуальної власності України 13.05.2015, м.Київ, НАУ.
10. Optimization of Software Architecture Selection for the System Under Design and Reengineering / Kharchenko O., Raichev I., Bodnarchuk I.,

Zagorodna N. // 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET), Lviv – Slavske, Ukraine, February 20-24, 2018. – pp.1245-1248.

Додаток А

Програмний код компоненту «профіль користувача» додатку:

```
import React, { useEffect, Fragment } from 'react';
import PropTypes from 'prop-types';
import { connect } from 'react-redux';
import { Link } from 'react-router-dom';
import Spinner from '../common/spinner/Spinner';
import { getProfileById } from '../../redux/actions/profileActions';
import ProfileTop from './ProfileTop';
import ProfileAbout from './ProfileAbout';
import ProfileExperience from './ProfileExperience';
import ProfileEducation from './ProfileEducation';
import ProfileGithub from './ProfileGithub';
const Profile = ({
  auth,
  getProfileById,
  profile: { loading, profile },
  match,
}) => {
  useEffect(() => {
    getProfileById(match.params.id);
  }, [getProfileById]);
  return (
    <Fragment>
      {' '}
      {profile === null || loading ? (
        <Spinner />
```

```
) : (
  <Fragment>
    {' '}
    <Link to="/profiles" className="btn btn-light">
      {' '}
      Back to Profiles{' '}
    </Link>{' '}
    {auth.isAuthenticated &&
      auth.loading === false &&
      auth.user._id === profile.user._id && (
        <Link to="/edit-profile" className="btn btn-dark">
          {' '}
          Edit Profile{' '}
        </Link>
      )}{' '}
    <div className="profile-grid">
      {' '}
      <ProfileTop profile={profile} /> <ProfileAbout profile={profile} />{' '}
      <div className="profile-exp bg-white p-2">
        {' '}
        <h2 className="text-primary"> Experiance </h2>{' '}
        {profile.experience.length > 0 ? (
          <Fragment>
            {' '}
            {profile.experience.map((experience) => (
              <ProfileExperience
                key={experience._id}
                experience={experience}
```

```
          />
        ))}{' '}
      </Fragment>
    ) : (
      <h4> No experience </h4>
    )}{' '}
  </div>{' '}
  <div className="profile-edu bg-white p-2">
    {' '}
    <h2 className="text-primary"> Education </h2>{' '}
    {profile.education.length > 0 ? (
      <Fragment>
        {' '}
        {profile.education.map((education) => (
          <ProfileEducation
            key={education._id}
            education={education}
          />
        ))}{' '}
      </Fragment>
    ) : (
      <h4> No Education </h4>
    )}{' '}
  </div>{' '}
  {profile.githubusername && (
    <ProfileGithub username={profile.githubusername} />
  )}{' '}
</div>{' '}
```

```
        </Fragment>
      )}{' '}
    </Fragment>
  );
};
Profile.propTypes = {
  getProfileById: PropTypes.func.isRequired,
  profile: PropTypes.object.isRequired,
  auth: PropTypes.object.isRequired,
};
const mapStateToProps = (state) => ({
  profile: state.profile,
  auth: state.auth,
});
export default connect(mapStateToProps, {
  getProfileById,
})(Profile);
```

Додаток Б

Програмний код компоненту «відео-чат» додатку:

```
import React from "react";
import { connect } from "react-redux";
import { isEmpty } from "lodash";
import { Modal, ModalHeader, ModalBody, ModalFooter } from "reactstrap";
const ChatModal = ({
showModal,
messages,
handleMessage,
closeChat,
sendMessage,
message,
}) => (
<Modal isOpen={showModal} className="video-chat-modal">
<ModalHeader>Chat Room</ModalHeader>
<ModalBody>
{isEmpty(messages) ? (
<p>No messages yet</p>
) : (
messages.map((item, index) => (
<div key={index} className="video-chat-modal__message">
<b>{item.sender}</b>: <span>{item.data}</span>
</div>
))
)}
</ModalBody>
```

```
<ModalFooter className="div-send-msg">
<input
placeholder="Message..."
value={message}
onChange={(e) => handleMessage(e)}
/>
<button onClick={() => sendMessage()}>Send</button>
<button onClick={() => closeChat()}>Close</button>
</ModalFooter>
</Modal>
);
export default ChatModal;
import React, { useState, useEffect, lazy } from "react";
import { isChrome } from "../../utils/videoChatConst";
import { connect } from "react-redux";
import "./VideoChat.scss";
const UsersList = lazy(() => import("./UsersList"));
const VideoChatContent = lazy(() => import("./VideoChatContent"));
const VideoChat = ({ user, profile }) => {
if (!isChrome())
<div className="video-not-support">
<h1>Sorry, yourbrowser doesnt support video chat</h1>
</div>;
return (
<div className="video-chat-wrapper">
<UsersList />
<VideoChatContent user={user} profile={profile} />
</div>
```

```
);
};
const mapStateToProps = (state) => ({
user: state.auth.user,
profile: state.profile,
});
export default connect(mapStateToProps, {})(VideoChat);
import React, { useEffect, useState } from "react";
import { connect } from "react-redux";
import { getAllUsersList } from "../../redux/actions/profileActions";
import { profilePhotoUrl } from "../../utils/urls";
import "./UsersList.scss";
const UsersList = ({ auth: { user }, getAllUsersList, profile: { users } }) => {
useEffect(() => {
getAllUsersList();
}, []);
return (
<div className="users-list-wrapper">
<h3>Users list</h3>
<ul>
{users.map(
(userItem, index) =>
user._id !== userItem._id && (
<li key={index} className="user-list__item">
<img
className="user-list__photo"
alt="photo"
src={profilePhotoUrl(userItem.email)}
```

```
/>
<p>{userItem.name}</p>
<button className="btn btn-light">invite</button>
</li>
)
)}
</ul>
</div>
);
};
const mapStateToProps = (state) => ({
profile: state.profile,
auth: state.auth,
});
export default connect(mapStateToProps, { getAllUsersList })(UsersList);
@import "../../variables";
.users-list-wrapper {
display: flex;
background: $darkBlueV1;
color: #fff;
flex-direction: column;
width: fit-content;
min-width: 300px;
padding: 0.4rem;
h3 {
text-align: center;
font-size: 1rem;
border-bottom: 2px solid #eee;
```

```
border-radius: 4px;
margin: 0.5rem 0.2rem;
white-space: nowrap;
padding: 5px 0.8rem;
}
.user-list__item {
display: flex;
flex-wrap: wrap;
flex-direction: row;
align-items: flex-end;
margin-bottom: 1rem;
font-weight: bold;
&:nth-last-child(1) {
margin-bottom: 0;
}
.btn-light {
margin-top: .5rem ;
width: 100%;
}
.user-list__photo {
overflow: hidden;
border-radius: 5%;
object-fit: cover;
width: 50px;
height: 50px;
border: 2px solid #eee;
}
p {
```

```
margin-left: 0.4rem;
margin-bottom: 0;
height: min-content;
}
}
}
```