НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
РАДІОТЕХНІЧНИЙ ФАКУЛЬТЕТ
КАФЕДРА ПРИКЛАДНОЇ РАДІОЕЛЕКТРОНІКИ

До захисту допущено:
В.о.зав. кафедри
_____ Михайло СТЕПАНОВ
«___»_____20__ р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньою-професійною програмою «Радіозв'язок і оброблення сигналів»

спеціальності 172 Телекомунікації та радіотехніка

на тему: «Мережа мультимедійних модерованих інформаційних панелей**»**

Виконав (-ла):
студент (-ка) IV курсу, групи РА-81

Горбенко Артур Юрійович _____
Прізвище, ім'я та по батькові підпис

Керівник:
Доц. к.т.н Мосійчук В.С. _____
Посада, науковий ступінь, вчене звання, Прізвище, ім'я та по батькові підпис

Рецензент:
Доц. д.т.н., Сушко О.Ю. _____
Посада, науковий ступінь, вчене звання, Прізвище, ім'я та по батькові підпис

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) Горбенко А.Ю.

Київ – 2022 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Радіотехнічний факультет
Кафедра прикладної радіоелектроніки

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 172 Телекомунікації та радіотехніка

Освітньо-професійна програма «Радіозв'язок і оброблення сигналів»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Михайло СТЕПАНОВ
«___»_____20__ р.

ЗАВДАННЯ
на дипломну роботу студенту
Горбенку Артуру Юрійовичу

1. Тема роботи «Мережа мультимедійних модерованих інформаційних панелей», керівник роботи доц.,к.т.н., Мосійчук В.С., затверджені наказом по університету від 822-с від 01.06.2022р

2. Термін подання студентом роботи 09 червня 2022 року

3. Вихідні дані до роботи нормальні умови експлуатації; підтримка FullHD екранів;підтримка усіх версій Raspberry PI;можливість розміщувати серверну частину локально;функціонал на додавання фото, відео та текстових слайдів;можливість керувати усіма панелями з адмін панелі.

4. Зміст пояснювальної записки
   1. Огляд сучасних аналогів об'єкту що проектується.
   2. Розробка та техніко-економічне обґрунтування технічних вимог проекту
   3. Вибір компонентів обладнання та програмних застосунків і технологій
   4. Огляд засобів розробки

5. Опис програмної реалізації

6. Робота користувача з системою

7. Висновок.

5. Дата видачі завдання 01 травня 2022 року

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1	Огляд існуючих рішень	3.05.22 – 13.05.22	
2	Розробка та техніко-економічне обґрунтування технічних вимог проекту	15.05.22 – 18.05.22	
3	Вибір компонентів обладнання та програмних застосунків і технологій	20.05.22 – 30.05.22	
4	Огляд засобів розробки	30.05.22 – 6.06.22	
5	Опис програмної реалізації	8.06.22 – 12.06.22	
6	Робота користувача з системою	12.06.22 – 15.06.22	
7	Оформлення документації	14.06.20 – 20.06.22	

Студент Горбенко А.Ю.

Керівник Мосійчук В.С.

АНОТАЦІЯ

Метою дипломної роботи є створення технічної документації на додаток керування мережею мультимедійних модерованих інформаційних панелей. Програмний додаток розроблено для подальшого використання радіотехнічним факультетом. Панелі будуть виконувати функції модерованої дошки оголошень, які вони будуть отримувати з сервера факультету, привітання студентів з святами та демонстрація досягнень факультету на дні відкритих дверей. Наявність необхідного обладнання на факультеті зменшує вартість розробки такого додатку.

Annontation

The purpose of the thesis is to create technical documentation in addition to managing a network of multimedia moderated information panels. The software application is designed for further use by the Faculty of Radio Engineering. The panels will serve as a moderated bulletin board, which they will receive from the faculty server, greet students with the holidays and demonstrate the achievements of the faculty at the open house. The availability of the necessary equipment at the faculty reduces the cost of developing such an application.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ОК – Одноплатний комп’ютер

JS – JavaScript, мова програмування

PHP - Hypertext Preprocessor, мова програмування

CSS – Cascading Style Sheets, мова стилізації

OS – Operation system, операційна система

GPIO – General purpose input, output, загально-цільовий вхід/вихід

MySQL — вільна система керування реляційними базами даних, яка була розроблена компанією «TcX» для підвищення швидкодії обробки великих баз даних

SSH — (англ. Secure SHell — «безпечна оболонка») — мережевий протокол рівня застосунків, що дозволяє проводити віддалене управління комп'ютером і тунелювання TCP-з'єднань (наприклад, для передачі файлів).

Full HD (Full High Definition) — роздільна здатність 1920x1080 точок (пікселів) і частотою кадрів не менше 24/сек

ПЕРЕЛІК ПОНЯТЬ

Віджет - (англ. widget) — контент-модуль, що вбудовується у вебсторінку або у браузер

Двофакторна автентифікація - це різновид багатофакторної автентифікації, яка грунтується на підтвердженні особистості користувача за допомогою двох, незалежних один від одного, факторів.

Модеровані панелі – панелі, що керуються, редагуються та підтримуются користувачем.

Canvas (англ. canvas — «полотно») — елемент HTML5, який можна застосовувати для малювання графіки використовуючи скрипти (переважно JavaScript). Наприклад його можна застосувати для малювання графів, створення фотокомпозицій а також анімації.

Apache Software License v2 - дозволена ліцензія на вільне програмне забезпечення, написана Apache Software Foundation (ASF).

Адмін-панель — інструмент для керування веб-ресурсом та його налаштуваннями, додавання нових та видалення старих сторінок, зміни зовнішнього вигляду веб-ресурсу та редагування контенту.

Linux – загальна назва UNIX-подібних операційних систем на основі однойменного ядра.

Android - операційна система і платформа для мобільних телефонів та планшетних комп'ютерів, створена компанією Google на базі ядра Linux.

DOM — Об'єктна модель документа (англ. Document Object Model,) — специфікація прикладного програмного інтерфейсу для роботи зі структурованими документами (як правило, документами XML). Визначається ця специфікація консорціумом W3C.

Слайд-шоу (англ. Slide show) — відеокліп, що формується з фотографій або відео.

ВСТУП

У наші непрості часи розповсюдження цифрової інформації є одною із важливіших завдань сучасних технологій. Тому набули певної популярності таки пристрої як цифрові вивіски, або як їх ще іноді називають інформаційні панелі.

Цифрові вивіски відносяться до технологій відображення, таких як світлодіодні стіни (або відеостіни), проекційні та рідкокристалічні монітори (РК-монітори) для яскравого відображення веб-сторінок, відео, маршрутів, меню ресторанів, маркетингових повідомлень або повідомлень о авіаційній небезпеці.

Цифрові вивіски функціонують у різних місцях — у громадських місцях, музеях, спортивних аренах, церквах, навчальних корпусах, роздрібних магазинах, корпоративних приміщеннях та ресторанах — щоб запропонувати шляхи пошуку, обмін повідомленнями, маркетинг і зовнішню рекламу.

Приклади таких інформаційних панелей можна використовувати для надання публічної інформації, передачі внутрішньої комунікації або обміну інформацією про продукт для покращення обслуговування клієнтів, рекламних акцій та пізнаваності бренду.

Отже розробки у цій галузі сучасних технологій є актуальними і іноді мають вузькоспеціалізовані потреби.

1. ОГЛЯД СУЧАСНИХ АНАЛОГІВ ОБ'ЄКТУ ЩО ПРОЕКТУЄТЬСЯ

У цьому розділі буде проведений огляд сучасних аналогів додатків до цифрових вивісок. Аналіз буде проведено за групами : безкоштовні та платні.

1.1 Платні додатки для цифрових вивісок.

Yodeck — це програмне забезпечення для цифрових вивісок світового класу. Це хмарне, просте у використанні програмне забезпечення, яке допомагає розробляти, планувати та відображати вміст в Інтернеті, який впливає на відвідувачів. Він допомагає виділити веб-сайти чи екрани за допомогою своїх безкоштовних віджетів, таких як дата й час, щоденна погода, аналоговий годинник та багато іншого.

Особливості:

- Показує медіа (зображення, аудіо, відео, файли документів, веб-сторінки тощо) на екрані за допомогою простого перетягування.
- Забезпечує легке налаштування за допомогою безкоштовних шаблонів і віджетів, які підтримують будь-яку роздільну здатність екрана.
- Легкий моніторинг та керування екранами за допомогою інтуїтивно зрозумілого планування, автоматичних оновлень тощо.
- Забезпечує чудову безпеку за допомогою захищених IP-адрес, двофакторної автентифікації, безпечного блокування гравця та обмеженого доступу.

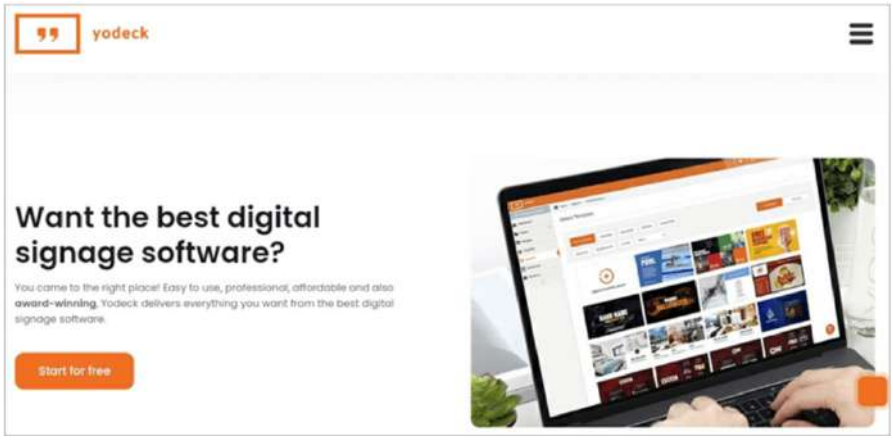

Рис 1.1.1 – Головна сторінка Yodesk

Вартість: 8$ у місяц

Novisign — це безпечне та просте програмне забезпечення для цифрових вивісок. Він полегшує візуальну комунікацію у школах, охороні здоров'я, роздрібній торгівлі тощо. Він надає такі функції, як студія цифрових вивісок, шаблони, віджети, планування списків відтворення, інтеграція та сенсорний екран.

Вартість : 10$ у місяць

Він відстежує екран і надає детальні звіти та аналітику та забезпечує безкоштовну підтримку кваліфікованої команди.

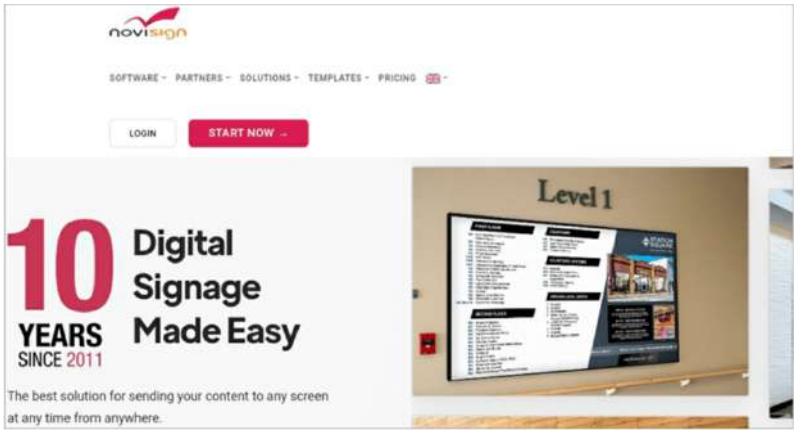

Рис 1.2.1— Головна сторінка Novosign

Особливості:

- Полегшує корпоративне спілкування за допомогою обміну оновленнями кадрів, бюлетенями компанії тощо.
- Допомагає в шкільному спілкуванні, розповсюджуючи новини факультету та оголошення викладачів.
- Створює цифрові дошки меню, додаючи описи, ціни та зображення.
- Простий у використанні та доступний.
- Надає безкоштовні шаблони та віджети.
- Планує список відтворення за лічені секунди.
- Забезпечує сенсорні системи лобі.

TelemetryTV — це потужне хмарне програмне забезпечення для вивісок, яке допомагає спілкуватися та керувати вмістом. Його використовували провідні

компанії по всьому світу, такі як Starbucks, Databricks та багато інших. Він допомагає керувати вмістом і контролювати його, щоб зробити його потужним та інтуїтивно зрозумілим.

Він поставляється з додатками та інтеграціями разом із набором інструментів. Завдяки цьому користувачі можуть дуже легко керувати мережею за допомогою автоматичного надання, звітів про безперебійну роботу та роботи в автономному режимі.

Вартість : 54$ у місяць

Рис 1.3.1 — Головна сторінка TelemetryTV

Особливості:

- Створює привабливий вміст за допомогою функцій перетягування або додавання відео.
- Дозволяє налаштовувати, контролювати та підтримувати вміст свіжим.
- Надійно відображає понад 70 інтегрованих додатків і дозволяє створювати будь-що за допомогою Canvas.
- Допомагає в масштабному управлінні мережею за допомогою автоматичного надання та своєчасного звітування.

- Полегшує планування вмісту списків відтворення.
- Надає організаціям потужні API для створення спеціальних програм, наборів правил і протоколів надання.

ScreenCloud — це програмне забезпечення для цифрових вивісок, яке сприяє залученню, продуктивності та продажам за допомогою цифрових вивісок. Воно забезпечує безпеку корпоративного рівня та просте керування вмістом також підтримує та безпечно показує різні інформаційні панелі з різних програм. Покращує продуктивність команд за рахунок обміну даними в режимі реального часу, які допомагають у прийнятті рішень і в кінцевому підсумку призводять до довгострокового підвищення продуктивності або продуктивності.

Вартість : 60$ у місяц

Особливості:

- Легко доступний з будь-якого обладнання після створення мережі.
- ScreenCloud Studio дозволяє легко й інтуїтивно керувати вмістом та екраном.
- Забезпечує безпеку облікового запису за допомогою системи єдиного входу, спеціальних дозволів, журналів аудиту та відповідності стандарту SOC2.
- Допомагає легко транслювати зустрічі та події в прямому ефірі.
- Надає інші корисні функції, такі як відображення інформаційних панелей, GraphQL API, 60+ інтеграцій тощо.

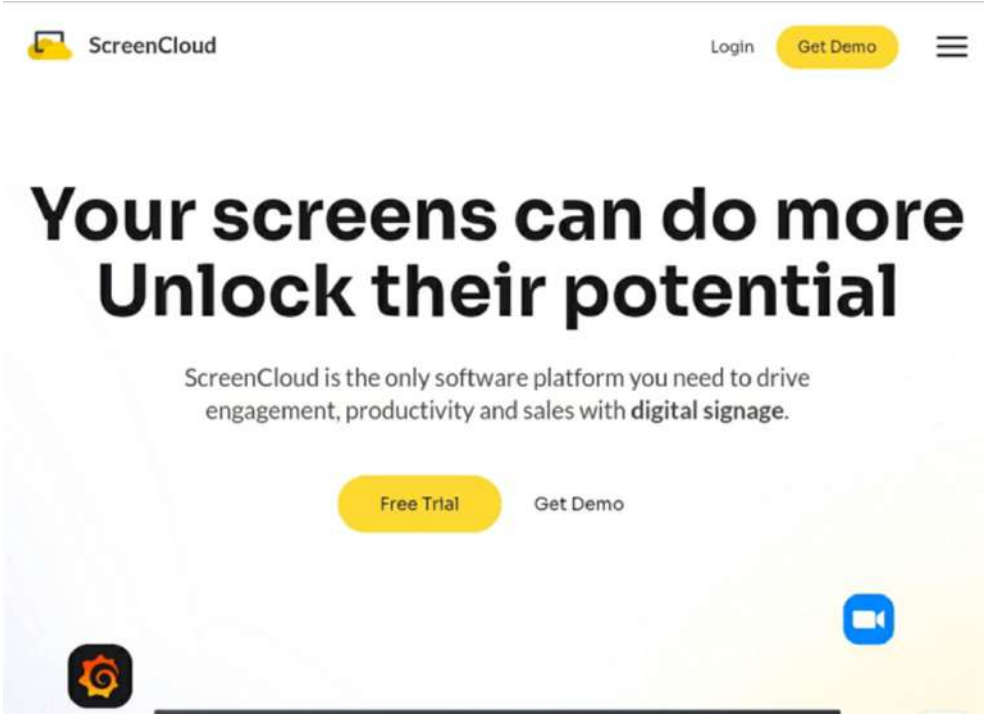

Рис 1.4.1. — Головна сторінка ScreenCloud

Вартість: 60$ у місяц

OptiSigns — це програмне забезпечення для вивісок, яке дозволяє користувачам створювати привабливий контент і керувати цифровими вивісками за допомогою простих і легких у використанні функцій. Він робить екрани за допомогою таких програм, як Google Data Studio, Facebook, Trello, Instagram та багато інших.

Він забезпечує розширену аналітику та штучний інтелект за допомогою звітів про відтворення в режимі реального часу та аналізу аудиторії, що допомагає змінювати/відображати відповідний вміст у реальному часі.

Вартість : 60$ у місяц

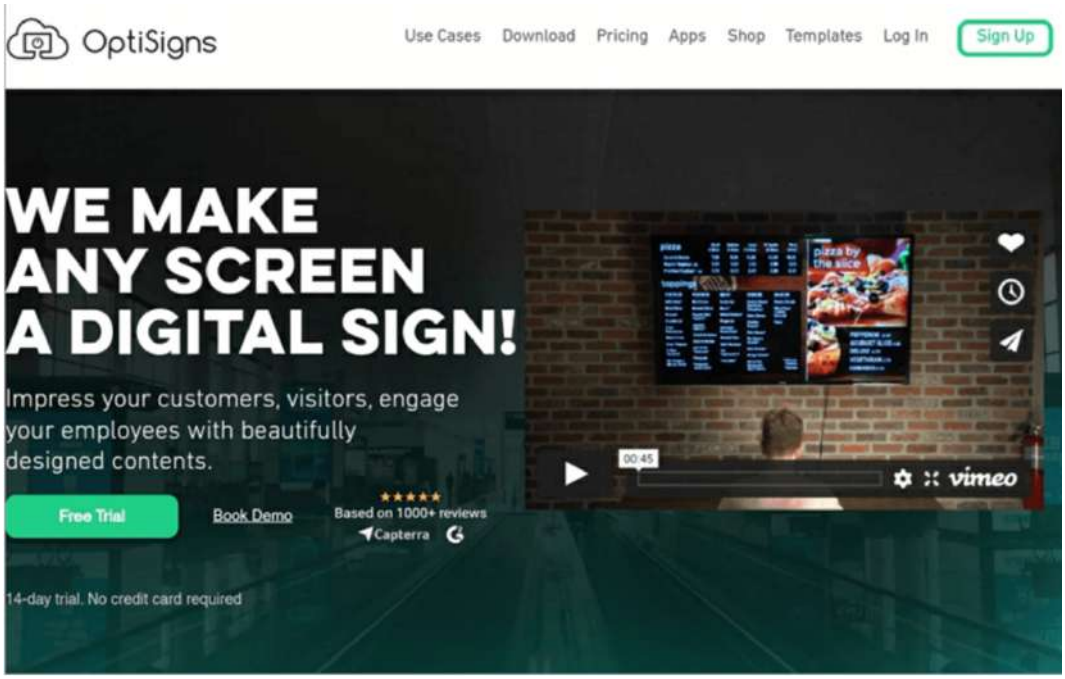

Рис 1.5.1 — Головна сторінка OptiSigns

Особливості:

- Підтримує різні формати аудіо, відео та зображень
- Надає функцію додатків для розміщення корисного вмісту на таких екранах, як соціальні мережі, погода та Google Презентації.
- Дозволяє створювати налаштований список відтворення, який включає зображення, відео, веб-посилання, програми.
- Вміст для планування надається гнучкими та потужними параметрами планування.
- Інші цінні функції включають підтримку орієнтації екрана, зони екрана, роботу в автономному режимі, автоматичний перезапуск тощо.
- Відстежує відтворення вмісту за допомогою функції звіту про підтвердження відтворення.

OnSign TV — це професійна платформа CMS для керування екранами вивісок. Він повністю наповнений потужними функціями, щоб розширити можливості професійних операторів вивісок.

Може бути корисно для моніторингу або керування користувачами та групами користувачів за допомогою таких функцій, як створення ролей для

кількох користувачів, детальне керування користувачами тощо. Він створює різноманітні звіти для кращого прийняття рішень, як-от звіти про підтвердження відтворення, автоматичні звіти за розкладом тощо.

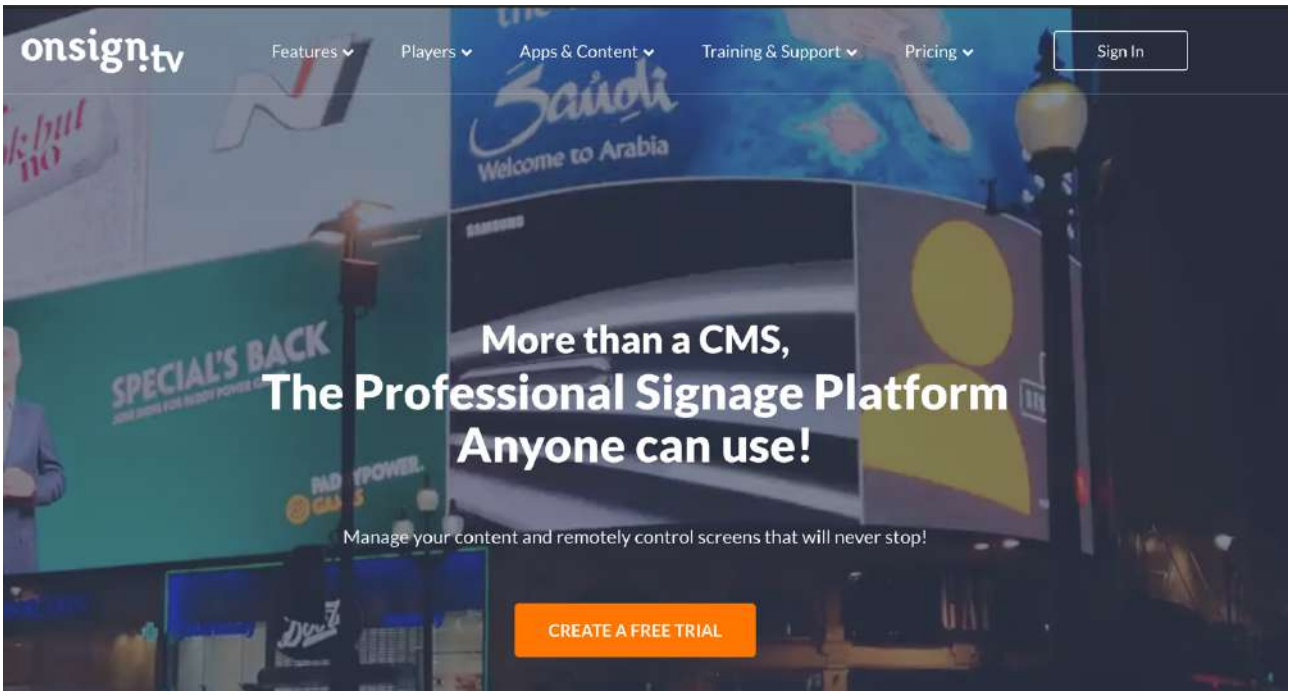

Рис 1.7.1 — Головна сторінка onSignTv

Особливості:

- Легке підключення або залучення спільноти за допомогою графічних, відео та текстових повідомлень.
- Веб-програмне забезпечення з відкритим кодом.
- Виділяє одне повідомлення з пулом інших повідомлень для різних місць і аудиторій.
- За допомогою цього програмного забезпечення можна розміщувати повідомлення на плоских телевізорах, мобільних телефонах, персональних заставках та інших веб-сайтах.
- Зменшує вартість, оскільки використання безкоштовне, потрібно лише заплатити за обладнання.
- Повністю сумісний зі смартфонами або мобільними пристроями нового покоління.

Viewneo — це хмарне програмне забезпечення для цифрових вивісок, яке допомагає користувачам легко створювати екрани цифрових вивісок. Він надає рішення для всіх галузей, включаючи охорону здоров'я, роздрібну торгівлю, державний сектор тощо.

Він надає настроюванні рішення з індивідуальними пакетами з прозорими цінами. Він забезпечує розробку онлайн-контенту, RSS-канали, понад 200 розроблених шаблонів, плагіни шаблонів, контроль доступу, звіти, плагін API та багато інших потужних функцій

Вартість : 18$ у місяць

Рис 1.9.1—Головна сторінка Viewneo

Особливості:

- Підтримує формати файлів, включаючи JPEG, відео, PowerPoint тощо.
- Забезпечує керування користувачами та необмежений список відтворення.
- Підтримує всі види інформації для публікації на екранах.
- Забезпечує настроюванні індивідуальні пакети з прозорими цінами.
- Включає понад 200 вбудованих шаблонів і дозволяє налаштувати один, який можна використовувати повторно.
- Надає функцію для створення онлайн-контенту для ваших презентацій.

- Легко розпочніть роботу за три кроки: створіть вміст, упорядкуйте список відтворення та підключіть програвач.

1.2 Безкоштовні додатки для цифрових вивісок

DigitalSignage.com — це безкоштовне програмне забезпечення для цифрових вивісок, яке дає змогу користувачам розробляти, поширювати й аналізувати вміст без написання жодного рядка коду.

Він дає змогу користувачам створювати сенсорні програми за допомогою інтерактивних рішень для цифрових вивісок і кіосків, які включають вибір шаблонів і роботу за хвилину, плавність роботи на GPU, дизайн перетягування та детальні звіти.

Вартість: програмне забезпечення безкоштовне

Рис 1.6.1—Головна сторінка DigitalSignage.com

Особливості:

- Перетягування інформації на екран або в список відтворення, яка має значення для аудиторії.
- Створення роботи один раз, щоб використовувати її в будь-якому місці у браузерах, ПК, планшетах, мобільних пристроях, Smart TV та на телеприставках.

- Надає детальні аналітичні звіти ваших кампаній для вимірювання натискань, ефективності кампаній та рентабельності інвестицій.
- Забезпечує перспективні рішення для кіосків із будь-якою потребою в навичках кодування.
- Підтримує новітні веб-стандарти
- Інші функції включають звіти підтвердження відтворення, контроль доступу, анімацію, дотик, необмежене хмарне сховище тощо.

Concerto — це безкоштовне веб-програмне забезпечення цифрових вивісок, яке допомагає своїм користувачам транслювати повідомлення про події, послуги та інші необхідні елементи на екранах.

Він надає свої послуги безкоштовно, оскільки випущений під ліцензією Apache Software License v2, тому це програмне забезпечення з відкритим кодом, яке можна розповсюджувати або змінювати. Додаток спрямований на обмін інформацією по всьому світу за допомогою сучасних веб-технологій та API.

Рис 1.8.1—Головна сторінка Concreto

Особливості:

- Легке підключення спільноти за допомогою графічних, відео та текстових повідомлень.
- Веб-програмне забезпечення з відкритим кодом.

- Виділяє одне повідомлення з пулом інших повідомлень для різних місць і аудиторій.
- За допомогою цього програмного забезпечення можна розміщувати повідомлення на плоских телевізорах, мобільних телефонах, персональних заставках та інших веб-сайтах.
- Зменшує вартість, оскільки використання безкоштовне, потрібно лише заплатити за обладнання.
- Повністю сумісний зі смартфонами або мобільними пристроями нового покоління.

1.3 Висновки

Під час аналізу існуючих рішень було з’ясовано, що існує велике різноманіття цифрових модерованих інформаційних панелей. Різне програмне забезпечення постачається з різними функціями та ціновими планами. Є широкий вибір продуктів, починаючи від безкоштовних версій з обмеженим функціоналом, так і дорогі розробки, призначені для масивного використання.

Деякі найкраще забезпечують такі функції, як звіти про відтворення, підтримка різних форматів файлів тощо. Деякі найкращі в легких налаштуваннях, як Yodeck і NoviSign. Деякі добре вміють керувати вмістом, як наприклад, Telemetry TV, ScreenCloud тощо.

Виходячи з аналізу аналогів, було узято до уваги, що майже усі існуючі рішення потребують специфічного обладнання та базуються на хмарних додатках. Отже, вони не є гнучкими відносно наприклад, існуючого обладнання користувача. Також, відсутня можливість розміщення серверної частини локально у мережі клієнта, що могло б значно пришвидшити оновлення контенту на панелі, а також прибрало б залежність від хмарних сервісів, оскільки при значному віддалені від серверів з’явиться велика затримка.

2. РОЗРОБКА ТА ТЕХНІКО-ЕКОНОМІЧНЕ ОБҐРУНТУВАННЯ ТЕХНІЧНИХ ВИМОГ ПРОЕКТУ

Розроблений програмний додаток буде мати менший функціонал відносно існуючих рішень у першій версії застосунку. Розроблений додаток буде придатний для використання на будь яких версіях мікрокомп’ютера Raspberry PI та на будь яких моделях моніторів, дисплеїв, телевізорів, смарт-телевізорів тощо з підтримкою HDMI.

2.1 Склад продукції

До складу входить:

- Raspberry PI 3 Model B
- Кабель mini-HDMI -- HDMI

2.2 Технічні вимоги

2.3.1 Вимоги до параметрів одноплатного комп’ютера.

Таблиця 2.3.1 – Параметри одноплатного комп’ютера.

№ п/п.	Параметр або характеристика	Одиниця виміру	Норма	Прим.
1	Порти для підключення панелей	шт	2	HDMI
2	RAM (Оперативна пам’ять)	ГБ	2	
3	Можливості вихода FULLHD зображення та відео			

Одноплатний комп’ютер для локального встановлення повинен мати RJ-45 інтерфейс та WIFI модуль для використання з хмарним сервером.

Для подальшого розвинення додатку за допомогою використання інших периферійних пристроїв як кнопки, клавіатури і так далі пристрій має бути обладнаним USB та GPIO програмованими портами.

2.3.2 Вимоги до серверної частини додатку

Серверна частина додатку має бути спроектована на сучасних серверних технологіях з використанням спеціалізованих мов програмування. Водночас з тим, серверна частина має працювати з базою даних MySQL як з найвідомішою СУБД. На серверній частині має бути впроваджена аутентифікація за допомогою унікального набору ключів.

2.3.3 Вимоги до клієнтської частини додатку

Клієнтська частина додатку має представляти собою адмін-панель з наступними можливостями

- Створення різного контенту для різних панелей
- Можливість додавати фото, відео, текст на панель
- Можливість зміни контенту та можливість корегування швидкості зміни

Клієнтська частина додатку має працювати в усіх сучасних браузерах.

2.3.4 Вимоги до додатка для Rasperry PI

Додаток для Raspberry PI має підтримувати версію програмного забезпечення, встановленого на Raspbian OS. Також передбачається підтримка аутентифікації, перезапуск та віддаленого інсталяції доданку за допомогою ssh. Розроблена програма має запускати браузер у повновіковому режимі без користувацького інтерфейсу, курсору миші та віконних кнопок.

2.4 Призначення системи

Система додатків призначена для виведення інформації на панелі для ознайомлення у різних видах мультимедіа форматів. Додатки складаються з наступних частин: клієнтська частина (адмін панель), серверна частина, додаток для Raspberri PI для запуску браузера у повноекранному режимі. З обладнання

рекомендується використовувати Raspberry PI 3 B, HDMI 2.0 кабель та будь який монітор з підтримкою HDMI та маючий FULLHD розширення.

2.5 Повний опис додатку

1. Raspberry PI підключаєтеся до панелі за допомогою HDMI.На карту пам’яті встановлюється остання доступна версія операційної системи Raspbian OS with Desktop.

2. Управління панеллю здійснюється через web-доданок, встановлений або локально у мережі факультету або на віддалених хмарових сервісах. За допомогою адмін-панелі, користувач може створювати нові картинки, відео або інші види слайдів для панелі. Також налаштовувати швидкість зміни контенту.

3. За допомогою технології ssh користувач запускає скрипт на Raspberry PI, котрий вже у свою чергу запускає браузер Chrome у повноекранному режимі на потрібній адресі.

4. Серверна частина підтримує систему, зберігаючи створений контент на сервері.

3. ВИБІР КОМПОНЕНТІВ ОБЛАДНАННЯ ТА ПРОГРАМНИХ ЗАСТОСУНКІВ І ТЕХНОЛОГІЙ

У даному розділі були розглянуті існуючи варіанти одноплатних комп'ютерів та був зроблений вибір, базуючись на вимогах до одоплатного комп'ютеру, описаних у розділі 2. Також проведений аналіз можливих підходів до розробки програмних застосунків, набору технологій та прийняті відповідні рішення щодо шляху реалізації поставленого завдання.

3.1 Огляд існуючих одноплатних ком'ютерів

Одноплатний комп'ютер (ОК) — це повноцінний комп'ютер, побудований на одній друкованій платі, з мікропроцесором, пам'яттю, введенням-виводом (вводом-виводом) та іншими функціями, необхідними для функціонального комп'ютера. До аналізу декілька рішень від відомих виробників, задовольняючих вимогам даної роботи, таки як Rasperry PI 4, Beaglebone Black, NVIDIA Jetson Nano, Google Coral Dev Board та Asus Tinker Board 2.

3.1.1 Порівняння за підтримкою операційних систем

З усіх цих одноплатних комп'ютерів Raspberry Pi, є тим, на якому можна запустити більшість операційних систем, такі як ОС Raspberry Pi, раніше відома як Raspbian, офіційна підтримувана операційна система для Raspberry Pi.
Він поставляється в різних версіях, які включають лише робочий стіл або робочий стіл із рекомендованим програмним забезпеченням, залежно від завантаженої версії.

Він також може запускати різні інші дистрибутиви на базі Linux та Android. Asus Tinker Board може працювати під керуванням операційних систем Linux або Android 10. Він був розроблений для роботи з Android 10.
Як і Raspberry Pi, Beaglebone Black може працювати з кількома операційними системами. Він поставляється з попередньо встановленим дистрибутивом Debian, але інші сумісні операційні системи включають Ubuntu, Android та інші.

На відміну від комп’ютерів вище, NVIDIA Jetson Nano і Google Coral Dev Board обмежені щодо до операційних систем. NVIDIA поставляється з попередньо встановленим Linux4Tegra, який базується на Ubuntu (окремий дистрибутив Linux) . Тому він повинен мати можливість запускати звичайний дистрибутив Ubuntu, але Linux4Tegra був спеціально розроблений для роботи на обладнанні NVIDIA.

Coral Dev Board від Google була розроблена для спеціальної операційної системи Google Mendel, яка призначена для використання з нейронною мережею, і постачається в стандартній комплектації.

3.1.2 Порівняння за продуктивність

Що стосується продуктивності, то Asus Tinker Board 2 має найвищі характеристики з шести ядерною системою Rockchip RK3399 на чіпі. Це дає Tinker Board 2 вищу продуктивність порівняно з іншими одноплатними комп’ютерами.

Другою на черзі, оскільки вона розроблена для програм машинного навчання з Edge TPU, є Coral Dev Board від Google. Він просто пропонує потужність і є одним із потужних одноплатних комп’ютерів на ринку.

Наступним на черзі є Raspberry Pi 4. Його процесор Cortex A72 пропонує більш високу продуктивність і швидкість тактової роботи порівняно з попередніми версіями. Крім того Raspberry Pi 4 можна легко розігнати, що призводить до набагато вищих результатів у тестуванні продуктивності.

За Raspberry Pi знаходиться NVIDIA Jetson Nano. Його процесор на покоління відстає від процесора Raspberry Pi 4, тому йому може не вистачити потужності Raspberry Pi.

У нижній частині знаходиться Beaglebone Black. Його ARM Cortex-A8 з частотою 1 ГГц не задовольняє вимогам до одноплатного комп’ютеру, оскільки не може відтворювати зображення на достатньому розширенні монітору(панелі).

3.1.3 Порівняння за відтворенням відео.

NVIDIA Jetson Nano має надзвичайно потужний графічний процесор у вигляді 128-ядерного графічного процесора Maxwell. Це робить його придатним для машинного навчання та програм штучного інтелекту, де необхідний потужний графічний процесор.

Однак проблема полягає в тому, що він поставляється лише з одним відеовиходом, тому використовувати подвійні дисплеї з Jetson Nano неможливо. Відповідно до вимог, ОК має мати мінімум 2 відеовиходи.

Наступний представник Asus Tinker Board 2. Його графічний процесор, Mali T-860, що працює на частоті 800 МГц, є багатоядерним графічним процесором, який є найвищою моделлю, побудованою на архітектурі Arm Midgard.

Він пропонує повну підтримку подвійного дисплея з роздільною здатністю 4K UHD.

Як і Asus Tinker Board 2, Raspberry Pi підтримує подвійний дисплей. Це робить обидва ці комп'ютери можливими для використання для цифрових вивісок.

Beaglebone Black має графічний процесор PowerVR SGX530, який працює на частоті 200 МГц, хоча і має деяке апаратне прискорення 3D, не може відповідати характеристикам інших одноплатних комп'ютерів у цьому порівнянні.

Coral Dev Board є надзвичайно потужною машиною, і вона має необхідні можливості графічного процесора для запуску TensorFlow (бібліотека машинного навчання) і використання в додатках машинного навчання та штучного інтелекту.

Однак проблема Coral Dev Board полягає в тому, що в ній відсутній настільна операційна система. Згідно з офіційною документацією Coral Dev Board, не рекомендується підключати монітор і клавіатуру до плати, а користувачам слід використовувати лише мережеві з'єднання. Отже, хоча він може бути потужним, його головне призначення — не відображати відео.

3.1.4 Порівняння за пам'яттю.

Що стосується пам’яті, Raspberry Pi 4 пропонує найбільше опцій. Він пропонує 2 ГБ, 4 ГБ або 8 ГБ пам’яті LPDDR4. Далі йде Asus Tinker Board 2, який має варіанти двоканальної пам’яті LPDDR 2 ГБ або 4 ГБ.

NVIDIA Jetson Nano пропонує лише один варіант із 4 ГБ пам'яті LPDDR. Coral Dev Board від Google пропонує 1 ГБ пам’яті, але, за їхніми словами, незабаром з’являться варіанти на 2 ГБ та 4 ГБ. Знову виховуючи задню частину, Beaglebone Black з лише 512 МБ пам'яті.

Окрім Beaglebone Black, усі ці комп’ютери мають достатньо пам’яті для завдань, які поставлені у вимогах до ОК.

3.1.5 Порівняння за варіативністю підключення

Raspberry Pi 4 пропонує два порти USB 3.0, два порти USB 2.0, стандартний 40-контактний роз’єм GPIO Raspberry Pi, який повністю сумісний із попередніми моделями, два порти micro-HDMI, 2-смуговий порт MIPI, двосмуговий Порт камери MIPI CSI, порт стереозвуку та композитного відео, Gigabit Ethernet, Wi-Fi та Bluetooth 5.0.

NVIDIA Jetson Nano пропонує чотири порти USB 3.0, один порт USB 2.0, порт HDMI і дисплея, два порти для камер MIPI CSI, Gigabit Ethernet і Wi-Fi.

Asus Tinker Board 2 пропонує один USB 3.2 type-C, три порти USB 3.2 type-A, один порт камери MIPI CSI, один 40-контактний роз'єм, один аудіо вихід HDMI, один порт HDMI, один порт дисплея USB типу C, один Порт дисплея MIPI DSI, Gigabit Ethernet, Wi-Fi і Bluetooth 5.0.

Coral Dev Board від Google пропонує два порти USB типу C, один порт USB 3.0 типу A і порт USB micro-B. Він також має порт HDMI 2.0, порт дисплея MIPI DSI, порт камери MIPI CSI, аудіо роз’єм 3,5 мм, два цифрових мікрофони PDM, чотири контактний роз'єм для стерео динаміків, Gigabit Ethernet, Wi-Fi і Bluetooth 4.2.

Beaglebone Black має порт USB для живлення та зв’язку, хост-порт USB, Ethernet, HDMI і два 46-контактних роз’єму. Beaglebone Black не оснащений Wi-

Fi або Bluetooth, тому потрібно розглядати Beaglebone Black Wireless, який стандартно має Bluetooth і Wi-Fi.

Виходячи з аналізу виводів різних ОК, майже усі вони відповідають вимогам пункту 2.

3.1.6 Порівняння за можливостями зберігання інформації

Усі ці одноплатні комп'ютери мають роз'єми для карт пам'яті micro-SD. Крім того, Beaglebone Black поставляється з 4 ГБ 8-розрядної флеш-пам'яті eMMC, а Coral Dev Board — з 8 ГБ вбудованої пам'яті eMMC. Крім того, Asus Tinker Board 2S поставляється з 16 ГБ вбудованої пам'яті eMMC.

Всі вони пропонують можливість налаштувати обсяг пам'яті відповідно до вимог за допомогою карти micro-SD відповідного розміру. Raspberry Pi, Asus Tinker Board 2 і NVIDIA Jetson Nano операційна система займе відповідне місце на карті SD.

3.1.7 Порівняння за вартістю

Ціна Raspberry Pi 4 Model B починається від 35 доларів за модель 2 ГБ. Ця ціна сягає 55 доларів за модель з 4 ГБ і 75 доларів за модель з 8 ГБ.

Ціна комплекту розробника NVIDIA Jetson Nano становить 99 доларів. Важливо відзначити, що NVIDIA також випустила комплект розробників NVIDIA Jetson Nano 2 ГБ. Окрім меншої пам'яті, він також має менше функцій, ніж комплект розробника Jetson Nano. Ціна на цю модель становить 59 доларів.

Ціна на Beaglebone Black становить близько 60,00 доларів. Ціна на Coral Dev Board становить 129,99 доларів США. Як згадувалося вище, планується запустити моделі Coral Dev Board на 2 ГБ і 4 ГБ.

На момент написання цієї роботи ціни та наявність Asus Tinker Board 2 ще не були доступні, але орієнтовно Asus спробує запропонувати його ринку за ціною, подібною до попередньої моделі. Ця модель стартувала за ціною 55 доларів.

3.1.8 Висновки про одноплатний комп’ютер.

Проаналізувавши існуючи ОК на ринку, було з’ясовано, що відповідно вимогам підходять лише деякі моделі. Основна проблема NVIDIA Jetson Nano полягає у відсутності підтримки подвійного відеовиходу.
Згідно документації Coral Dev Board пристрій не призначено для відображення відео.

Beaglebone Black не відповідає поставленим вимогам по декілька параметрів : не достатньо оперативної пам’яті, занадто слабкий процесор, що не підтримує необхідний рівень відображення, відсутність WIFI у базовій комплектації.

Отже, залишається обрати між Asus Tinker Board 2 і Raspberri PI. Обидві моделі мають 2 відеовиходи, підтримують широкий спектр операційних систем та мають достатній об’єм оперативної пам’яті. Основним відмінністю стає вартість обох пристроїв, де Raspberry PI буде дешевше приблизно на 40%.

Отже, у ході аналізу було обрано Raspberry PI як найоптимальніший ОК, що відповідає усім вимогам пункту 2 цієї дипломної роботи.

3.2 Огляд підходів розробки веб-додатків

Ефективна, якісна веб-архітектура стала де-факто показником хорошого продукту і дійсно є важливою для потоку даних та інформації для досягнення поставлених завдань. Вона діє як план потоку даних та інформації, який може ефективно вирішувати проблеми. Архітектура з продуманими функціями та інтуїтивно зрозумілим інтерфейсом забезпечує безперебійну роботу користувача. Це також зменшує ймовірність збою програми, щоб уникнути простоїв.

3.2 Архітектура веб-додатків

Архітектура веб-додатків — це план одночасних взаємодій між компонентами, базами даних, системами проміжного програмного забезпечення,

інтерфейсами користувача та серверами в додатку. Його також можна описати як макет, який логічна визначає з’єднання між сервером і клієнтською стороною для кращої роботи в Інтернеті.

Веб-додатку потрібна архітектура, щоб закласти міцний фундамент, і без неї додаток буде викликати проблеми кожного разу при імплементації кожної нової функціональності.

Добре продумана архітектура веб-додатків може впоратися з різними навантаженнями та вміло адаптуватися до мінливих вимог, щоб забезпечити швидкий досвід користувача, що ще більше покращує продуктивність програми. Також можна взяти кілька завдань розробки одночасно, розділивши структуру на кілька невеликих модулів, зрештою скоротивши час розробки. Крім того, стає легше інтегрувати нові функції, не впливаючи на загальну структуру.

Веб-архітектура ділить програму на кілька блоків, які захищені окремо, щоб мінімізувати загрози безпеці, включаючи ризик, який створює шкідливий код. Крім того, програми з архітектурою, що забезпечує майбутнє, дають можливість додавати нові функції та підтримувати низьку затримку, навіть коли кількість користувачів збільшується.

Усі програми складаються з двох основних компонентів:

- Клієнтська сторона, яку ще називають інтерфейс, де код пишеться на HTML, CSS, JavaScript і зберігається в браузері. Тут відбувається взаємодія користувача з додатком.
- На стороні сервера, також відомий як бекенд, керує бізнес-логікою і відповідає на запити HTTP. Код на стороні сервера написаний на Java, PHP, Ruby, Python тощо.

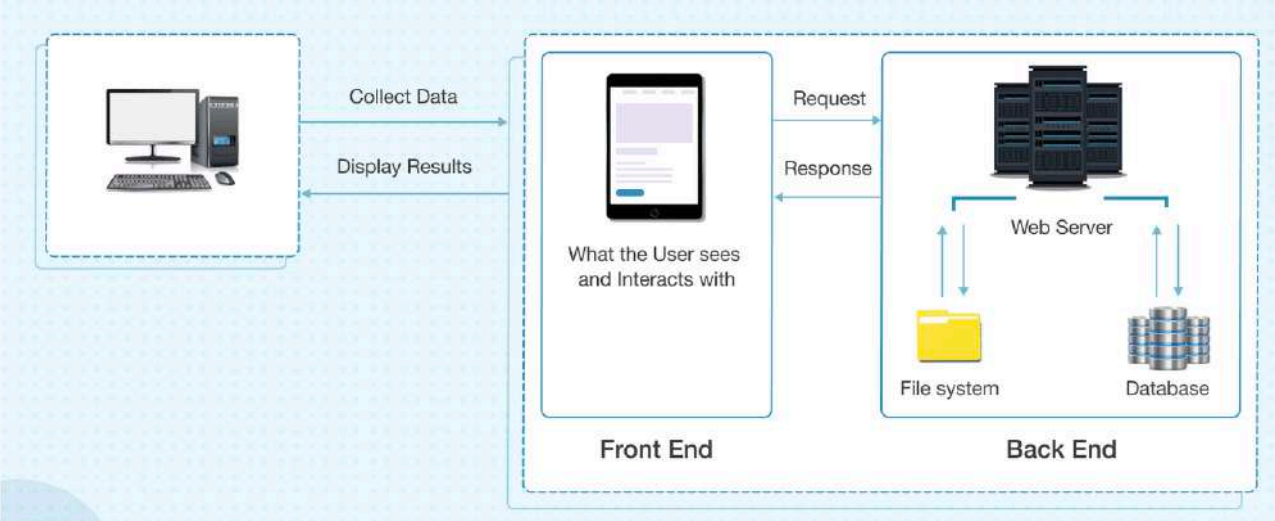

Рис 3.2—Схема типового web-додатку.

Крім цього, є додатковий компонент, тобто сервер бази даних, який надсилає запитані дані на сервер.

Принцип роботи полягаю в наступному: після вводу URL-адресу, наприклад, «kpi.ua.com» у браузері та натиску клавіши Enter, браузер відправить запит на сервер доменних імен, який розпізнає IP-адресу, а потім відправить запит на сервер, де знаходиться сайт КПІ. Після цього сервер прийме запит і відправить його в сховище даних, щоб знайти сторінку, і запитає дані для відображення в браузері. Після цього на екрані відобразиться сторінка з необхідною інформацією.

3.3 Шари архітектури веб-додатку.

Сучасні веб-додатки мають багатошарову архітектуру, яка складається з рівнів презентації, бізнесу, збереження та бази даних. Невеликі додатки мають три рівні, де в деяких випадках шари бізнесу та постійності діють як один рівень, тоді як складні програми мають п’ять або шість шарів.

- Рівень презентації, створений за допомогою HTML, CSS, JavaScript та його фреймворків, забезпечує взаємодію між інтерфейсом і браузером для полегшення взаємодії з користувачем.

- Бізнес-рівень визначає бізнес-логіку та правила. Він обробляє запити браузера, виконує бізнес-логіку, пов’язану із запитами, а потім надсилає її на рівень презентації.
- Рівень збереження відповідає за збереження даних і також відомий як рівень доступу до даних. Він тісно пов’язаний з бізнес-рівнем і має сервер бази даних, який отримує дані з відповідних серверів.
- Рівень бази даних, також відомий як рівень служби даних, містить усі дані та забезпечує безпеку даних, відокремлюючи бізнес-логіку від клієнтської сторони.

Кожен із цих шарів працює ізольовано. Це означає, що кожен шар працює незалежно. Компоненти одного шару закриті і мають справу з логікою відповідного шару. Наприклад, компоненти, що знаходяться на рівні презентації, мають справу з логікою презентації, тоді як компоненти на бізнес-рівні мають справу з бізнес-логікою.

Це також зменшує майбутнє навантаження, поки потрібні зміни у веб-додатку. Отже, зміни можна вносити в одному шарі, не зачіпаючи компоненти інших шарів.

3.4 Типи архітектури веб-додатків

Завжди корисно вибирати найбільш підходящу архітектуру, враховуючи різні фактори, такі як логіка програми, функції, функціональні можливості, бізнес-вимоги тощо. Правильна архітектура визначає ціль продукту в цілому.

3.4.1 Односторінкова архітектура програми

SPA (односторінкові програми) було введено для подолання традиційних обмежень для досягнення плавної роботи програми, інтуїтивно зрозумілого та інтерактивного користувацького досвіду.

Замість того, щоб завантажувати нову сторінку, SPA завантажують одну веб-сторінку та перезавантажують запитані дані на тій же сторінці з динамічна обновлюваним вмістом. Решта веб-сторінки залишається недоторканою. Вони

розроблені на стороні клієнта з використанням фреймворків JavaScript, оскільки вся логіка завжди переноситься на інтерфейс.

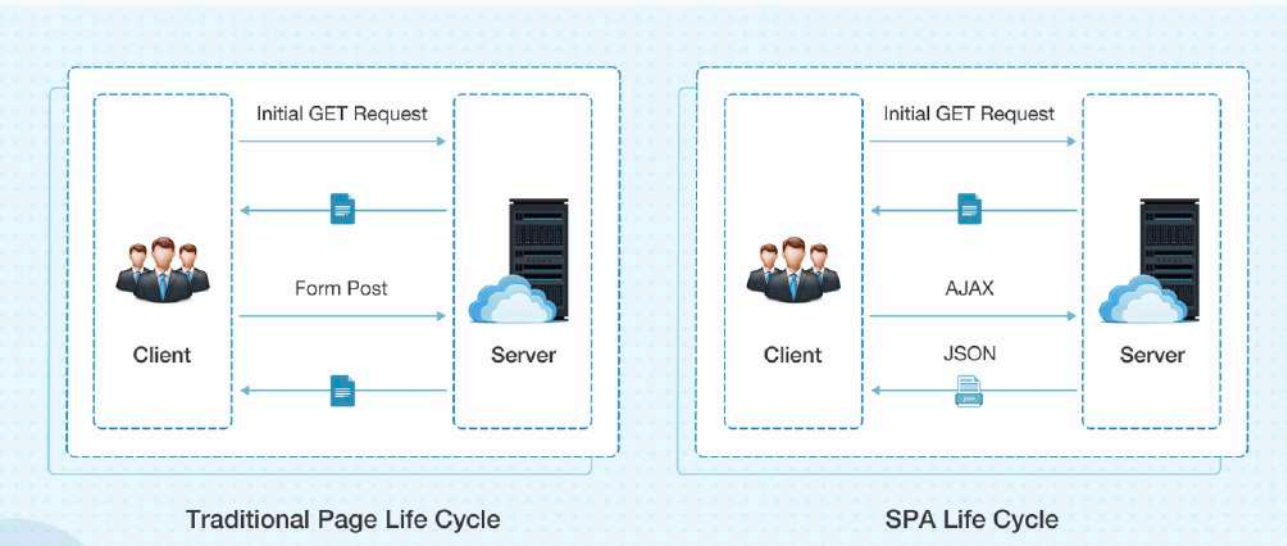

Рис 3.4.1 — Відмінність класичної архітектури і односторінкової.

3.4.2 Архітектура мікросервісів

Мікросервісна архітектура стала найкращою альтернативою сервіс-орієнтованій архітектурі (SOA) і монолітній архітектурі. Служби слабо пов’язані між собою, щоб їх розробляли, тестували, підтримували та впроваджували незалежно. Такі служби також можуть взаємодіяти з іншими службами через API для вирішення складних бізнес-задач.

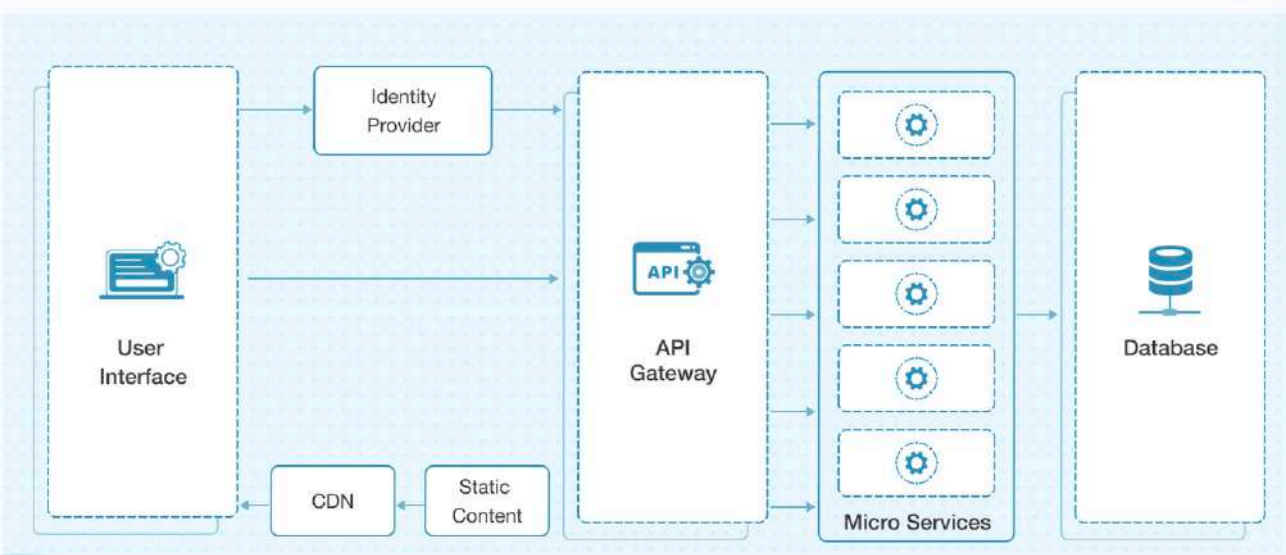

Рис 3.4.2 — Діаграма мікросервісної архітектури.

Розгортання веб-програм з використанням монолітної архітектури є громіздким завданням через тісно пов’язані компоненти. Мікросервіси вирішили цю проблему, розділивши програму на кілька окремих компонентів служби. Це додатково спрощує підключення між компонентами сервісу та усуває потребу в оркестровці послуг.

3.4.3 Безсерверна архітектура

Це архітектура, де все виконання коду здійснюється постачальниками хмарних послуг – не потрібно розгортати їх вручну на сервері. Безсерверна архітектура — це шаблон проектування, в якому програми створюються та запускаються без будь-якого ручного втручання на серверах, якими керують сторонні постачальники хмарних послуг, як-от Amazon та Microsoft.

Рис 3.4.2 — Діаграма мікросервісної архітектури.

Такий підхід дозволяє більше зосередитися на якості та складності продукту, щоб зробити їх масштабованими та надійними. Загалом він

поділяється на два типи – Backend-as-a-Service (BaaS) і Function-as-a-Service (FaaS).

BaaS дозволяє розробникам зосередитися на завданнях розробки інтерфейсу, виключаючи операції, що виконуються на серверній частині. AWS Amplify, наприклад, є популярним продуктом BaaS. FaaS, з іншого боку, — це модель, керована подіями, яка дозволяє розробникам розбивати програми на невеликі функції, щоб зосередитися на коді та тригерах подій. Решту зроблять такі постачальники послуг FaaS, як AWS Lambda і Microsoft Azure.

3.4.4 Висновки

Виходячи з проведеного аналізу та оцінки технічного завдання, найбільш відповідною є архітектура односторінкового додатку. Вона зазвичай використовується в малонавантаженних додатках з обмеженим функціоналом. Таким чином буде досягнута плавність інтуїтивність інтерфейсу, можливість розширення додатку залишиться необмеженою.

4. ОГЛЯД ЗАСОБІВ РОЗРОБКИ

У даному розділі буде проведений аналіз обраних засобів розробки та технологій.

4.1 Середовище розробки

Visual Studio Code, або також VS Code,— це редактор вихідного коду, створений Microsoft для Windows, Linux і macOS. Функції включають підтримку налагодження, виділення синтаксису, інтелектуальне завершення коду, фрагменти, редагування коду та вбудовану систему контроля версій. Користувачі можуть змінювати тему, комбінації клавіш, параметри та встановлювати розширення, які додають додаткові функції.

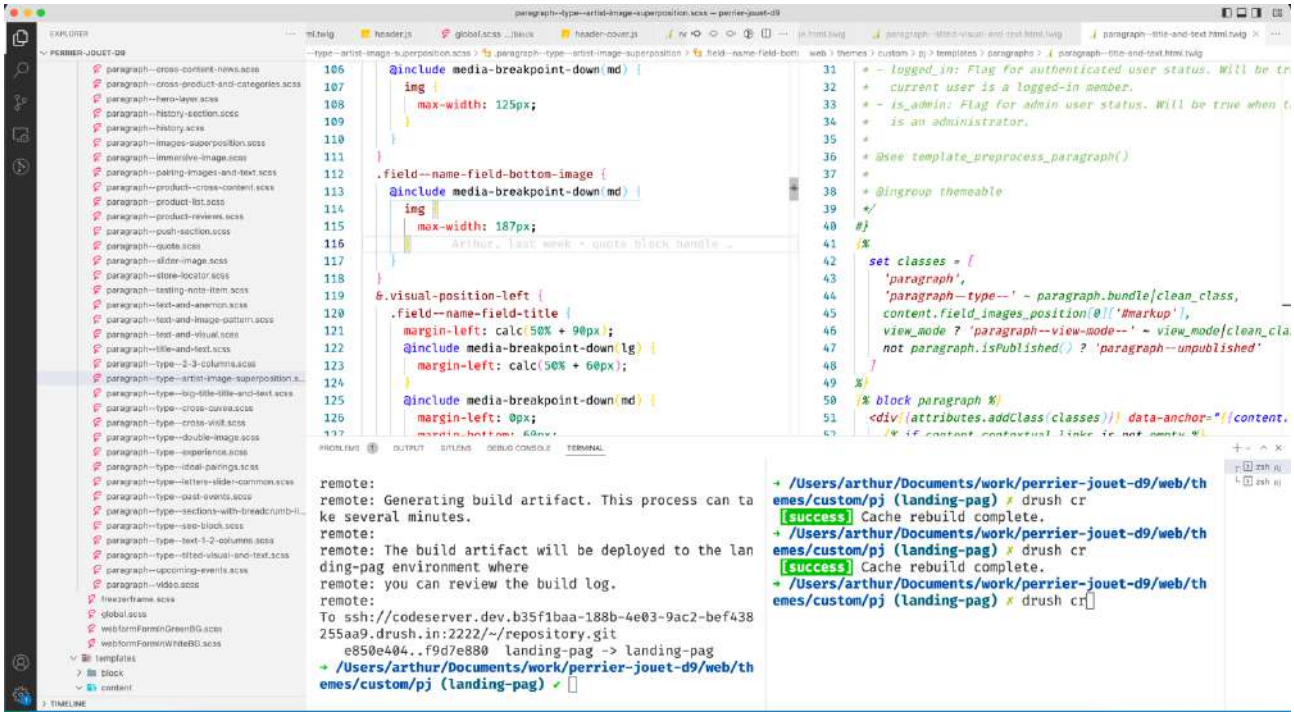

Рис 4.1.1— Відкритий проект у VS code.

У опитуванні розробників Stack Overflow 2021 Visual Studio Code був визнаний найпопулярнішим інструментом середовища розробника, при цьому 70% з 82 000 респондентів повідомили, що вони його використовують.

Visual Studio Code — це редактор вихідного коду, який можна використовувати з різними мовами програмування, включаючи Java, JavaScript, Go, Node.js,

Python, C++ і Fortran.Він заснований на фреймворку Electron,[20] який використовується для розробки веб-додатків Node.js, які працюють на механізмі макета Blink.

Visual Studio Code включає базову підтримку більшості поширених мов програмування. Підтримка додаткових мов може бути забезпечена безкоштовними розширеннями на VS Code Marketplace.

Visual Studio Code можна розширити за допомогою доповнень, доступних через центральне сховище. Це включає доповнення до редактора та підтримку мови. Примітною особливістю є можливість створювати розширення, які додають підтримку нових мов, виконувати статичний аналіз коду.

Контроль джерела є вбудованою функцією Visual Studio Code. Він має окрему вкладку всередині рядка меню, де можно отримати доступ до
Visual Studio Code включає кілька розширень для FTP, що дозволяє використовувати програмне забезпечення як безкоштовну альтернативу для веб-розробки. Код можна синхронізувати між редактором і сервером без завантаження будь-якого додаткового програмного забезпечення.

Visual Studio Code дозволяє користувачам встановлювати кодову сторінку, на якій зберігається активний документ, символ нового рядка та мову програмування активного документа. Це дозволяє використовувати його на будь-якій платформі, у будь-якій мові та для будь-якої мови програмування.

Visual Studio Code збирає дані про використання та надсилає їх до Microsoft, хоча це можна вимкнути. Через природу програми з відкритим вихідним кодом, код телеметрії доступний для розробників, яка може бачити, що саме зібрано.

4.2 Мова програмування php

Термін PHP є акронімом від PHP: Hypertext Preprocessor. PHP — це мова сценаріїв на стороні сервера, розроблена спеціально для веб-розробки. Це відкритий вихідний код, що означає, що його можна безкоштовно завантажувати

та використовувати. Він дуже простий у навчанні та використанні. Файли мають розширення «.php».

Расмус Лердорф надихнув створення першої версії PHP і брав участь у пізніших версіях. Це інтерпретована мова, і для неї не потрібен компілятор.

- PHP код виконується на сервері.
- Його можна інтегрувати з багатьма базами даних, такими як Oracle, Microsoft SQL Server, MySQL, PostgreSQL, Sybase, Informix.
- Він підтримує основні протоколи, такі як HTTP Basic, HTTP Digest, IMAP, FTP та інші.
- Такі веб-сайти, як www.facebook.com, www.yahoo.com, також створені на PHP.
- Однією з основних причин цього є те, що PHP можна легко вбудувати у файли HTML, а HTML-коди також можна записати у файл PHP.
- Річ, що відрізняє PHP від клієнтської мови, як наприклад HTML, полягає в тому, що коди PHP виконуються на сервері, тоді як HTML-коди відображаються безпосередньо в браузері. PHP коди спочатку виконуються на сервері, а потім результат повертається браузеру.
- Єдина інформація, яку знає клієнт або браузер, - це результат, повернутий після виконання PHP-скрипту на сервері, а не фактичні коди PHP, присутні у файлі PHP. Крім того, файли PHP можуть підтримувати інші мови сценаріїв на стороні клієнта, такі як CSS і JavaScript.

4.3 Веб-фреймворк symfony

Веб-фреймворк PHP — це набір класів, які допомагають розробити веб-додаток. Symfony — це фреймворк MVC з відкритим кодом для швидкого розвитку сучасних веб-додатків. Symfony — це повноцінний веб-фреймворк. Він містить набір компонентів PHP для повторного використання. Ви можете використовувати будь-які компоненти Symfony у додатках, незалежно від фреймворку.

Symfony має величезну кількість функціональних можливостей і активну спільноту. Він має гнучку конфігурацію з використанням YAML, XML або анотацій. Symfony інтегрується з незалежною бібліотекою та PHP Unit. Symfony в основному надихається Ruby on Rails, Django і фреймворками веб-додатків Spring. Компоненти Symfony використовуються багатьма проектами з відкритим кодом, які включають Composer, Drupal і phpBB.

Фреймворк Symfony складається з кількох компонентів, таких як компонент HttpFoundation, який розуміє HTTP і пропонує гарний об’єкт запиту та відповіді, який використовується іншими компонентами. По суті, ядро є «основним класом», який керує середовищем і несе відповідальність за обробку HTTP-запиту.

Добре організована структура Symfony, чистий код і хороші методи програмування полегшують веб-розробку. Symfony дуже гнучкий, використовується для створення мікросайтів і роботи з корпоративними додатками з мільярдами підключень.

Деякі з важливих функцій Symfony Framework полягають у наступному:

- Система на основі моделі-вигляду-контролера
- Високопродуктивний PHP фреймворк
- Гнучка маршрутизація URI
- Код багаторазового використання і його легше обслуговувати
- Управління сесією
- Помилка реєстрації
- Повнофункціональні класи баз даних з підтримкою кількох платформ
- Підтримує величезну та активну спільноту
- Набір роз’єднаних і багаторазових компонентів
- Стандартизація та сумісність програм
- Безпека від підробки міжсайтових запитів та інших атак
- Механізм шаблонів

Symfony пропонує розробникам велику гнучкість. Він має можливості для налагодження, читабельності коду та розробки розширюваних програм.

Symfony — це повноцінний веб-фреймворк; це дуже ефективний інструмент для створення веб-додатків.

Нижче наведено деякі переваги, які є у використання Symfony Framework.

Microframework — Symfony можна використовувати для розробки певної функціональності. Вам не потрібно переробляти або встановлювати весь каркас. Зменшує накладні витрати часу на розробку.

Надзвичайно зрілий механізм шаблонів і швидко доставляє вміст користувачам. Сумісний і розширюваний — розробники можуть легко розширити всі класи фреймворку.

4.4 Мова програмування JavaScript

JavaScript (часто скорочується до JS) — це легка, інтерпретована, об’єктно-орієнтована мова з першокласними функціями, найбільш відома як мова сценаріїв для веб-сторінок, але вона також використовується в багатьох середовищах, що не є браузерами. Це заснована на прототипах, мультипарадигмена мова сценаріїв, яка є динамічною та підтримує об’єктно-орієнтований, імперативний та функціональний стилі програмування.

JavaScript працює на стороні клієнта в Інтернеті, що можна використовувати для розробки/програмування поведінки веб-сторінок під час виникнення події. JavaScript — це легка для вивчення, а також потужна мова сценаріїв, широко використовується для керування поведінкою веб-сторінки.

Всупереч поширеній помилковій думці, JavaScript не є «інтерпретованою Java». JavaScript – це динамічна мова сценаріїв, яка підтримує конструювання об’єктів на основі прототипів. Основний синтаксис навмисно схожий як на Java, так і на C++, щоб зменшити кількість нових понять, необхідних для вивчення мови. Мовні конструкції, такі як оператори `if`, цикли `for` і `while`, а також блоки switch і `try ... catch` функціонують так само, як і в цих мовах.

JavaScript може функціонувати як процедурна, так і як об'єктно-орієнтована мова. Об’єкти створюються програмно в JavaScript, шляхом приєднання методів і властивостей до порожніх об’єктів під час виконання, на відміну від визначень синтаксичних класів, поширених у компільованих мовах, таких як C++ і Java. Після створення об’єкта його можна використовувати як план (або прототип) для створення подібних об’єктів.

Динамічні можливості JavaScript включають побудову об’єктів під час виконання, списки змінних параметрів, змінні функцій, створення динамічного сценарію (через `eval`), інтроспекцію об’єкта (через `for ... in`) і відновлення вихідного коду (програми JavaScript можуть декомпілювати тіла функцій назад у вихідний текст).

4.5 Бібліотека React

ReactJS — це бібліотека JavaScript, що використовується для створення компонентів інтерфейсу користувача, які можна повторно використовувати. Згідно з офіційною документацією React, нижче наведено визначення −

React — це бібліотека для створення компонованих інтерфейсів користувача. Це заохочує до створення повторно використовуваних компонентів інтерфейсу користувача, які представляють дані, які змінюються з часом. Багато людей використовують React як V в MVC. React абстрагує від DOM, пропонуючи просту модель програмування та кращу продуктивність. React також може відтворювати на сервері за допомогою Node, і він може запускати власні програми за допомогою React Native. React реалізує односторонній реактивний потік даних, що зменшує шаблон і легше міркувати, ніж традиційне прив’язування даних.

Особливості React :

- JSX – JSX є розширенням синтаксису JavaScript. Використовувати JSX у розробці React не обов’язково, але рекомендується.

- Компоненти – React. Розробник повинен думати про все як про компонент. Це допоможе підтримувати код під час роботи над великими проектами.
- Односпрямований потік даних і Flux – React реалізує односторонній потік даних, що дозволяє легко міркувати про програму. Flux — це шаблон, який допомагає підтримувати однонаправленість даних.
- Ліцензія – React ліцензується відповідно до Facebook Inc. Документація ліцензована відповідно до CC BY 4.0.

Переваги React

- Використовує віртуальний DOM, який є об’єктом JavaScript. Це покращить продуктивність програм, оскільки віртуальний DOM JavaScript швидше, ніж звичайний DOM.
- Може використовуватися на стороні клієнта і сервера, а також з іншими фреймворками.
- Шаблони компонентів і даних покращують читабельність, що допомагає підтримувати більші програми.
- Обмеження реакції
- Охоплює лише рівень перегляду програми, тому вам все одно потрібно вибрати інші технології, щоб отримати повний набір інструментів для розробки.
- Використовує вбудовані шаблони та JSX, що може здатися незручним деяким розробникам

4.6 СУРБД MySQL

База даних — це окрема програма, яка зберігає набір даних. Кожна база даних має один або кілька окремих API для створення, доступу, керування, пошуку та тиражування даних, які вона зберігає.

Можна також використовувати інші типи сховищ даних, наприклад файли у файловій системі або великі хеш-таблиці в пам'яті, але вибірка та запис даних були б не такими швидкими та простими з такими системами.

Сьогодні ми використовуємо системи керування реляційними базами даних (RDBMS) для зберігання та керування величезними обсягами даних. Це називається реляційною базою даних, оскільки всі дані зберігаються в різних таблицях, а відносини встановлюються за допомогою первинних ключів або інших ключів, відомих як зовнішні ключі.

Система управління реляційною базою даних (СУРБД) — це програмне забезпечення, яке −

- Дозволяє реалізувати базу даних з таблицями, стовпцями та індексами.
- Гарантує цілісність посилань між рядками різних таблиць.
- Оновлює індекси автоматично.
- Інтерпретує запит SQL і поєднує інформацію з різних таблиць.

MySQL — це швидка, проста у використанні СУБД, яка використовується для багатьох малих і великих підприємств. MySQL розробляється, продається і підтримується компанією MySQL AB, яка є шведською компанією. MySQL стає таким популярним через багато вагомих причин –

- MySQL випускається під ліцензією з відкритим кодом.
- MySQL є потужним додатком , він обробляє велику частину функціональних можливостей найдорожчих і потужних пакетів баз даних.
- MySQL використовує стандартну форму відомої мови даних SQL.
- MySQL працює на багатьох операційних системах і з багатьма мовами, включаючи PHP, PERL, C, C++, JAVA тощо.

- MySQL працює дуже швидко і впорається з великими наборами даних.
- MySQL добре інтегрується з PHP
- MySQL підтримує великі бази даних, до 50 мільйонів рядків або більше в таблиці. Обмеження розміру файлу за замовчуванням для таблиці становить 4 ГБ, але є можливість збільшити його до теоретичного обмеження в 8 мільйонів терабайт (ТБ).
- MySQL можна налаштувати. Ліцензія GPL з відкритим вихідним кодом дозволяє програмістам модифікувати програмне забезпечення MySQL відповідно до свого власного середовища.

4.7 Висновки

У ході огляду обраних рішень було встановлено, що застосовані технології мають велику спільну інтеграцію, отже разом складають надійну і сучасну систему додатків. Обрана мова серверної розробки є сучасною і поширеною технологією, та добре працює у парі з MySQL, що відповідає вимогам пункту 2.3.2. Для реалізації клієнтської частини була обрана сучасна бібліотека компонентів React, яка дозволить реалізувати усі вимоги, описані у п. 2.3.3.

5. ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

У даному розділі буде приведений опис програмної реалізації застосунку та опис застосованих підходів до реалізацій.

5.1 Архітектура додатку роботи з БД.

Було вирішено використовувати symfony вбудовану систему Doctrine для роботи з базою даних. Приклад створення нової сутності наведений на Рис 5.1.1

```
$ php bin/console make:entity

Class name of the entity to create or update:
> Product

New property name (press <return> to stop adding fields):
> name

Field type (enter ? to see all types) [string]:
> string

Field length [255]:
> 255

Can this field be null in the database (nullable) (yes/no) [no]:
> no

New property name (press <return> to stop adding fields):
> price

Field type (enter ? to see all types) [string]:
> integer

Can this field be null in the database (nullable) (yes/no) [no]:
> no

New property name (press <return> to stop adding fields):
>
(press enter again to finish)
```

Рис 5.1.1— Приклад створення нової сутності у БД за допомогою консольних команд

Таким чином створені основні сутності проекту, приведені на Рис 5.1.2

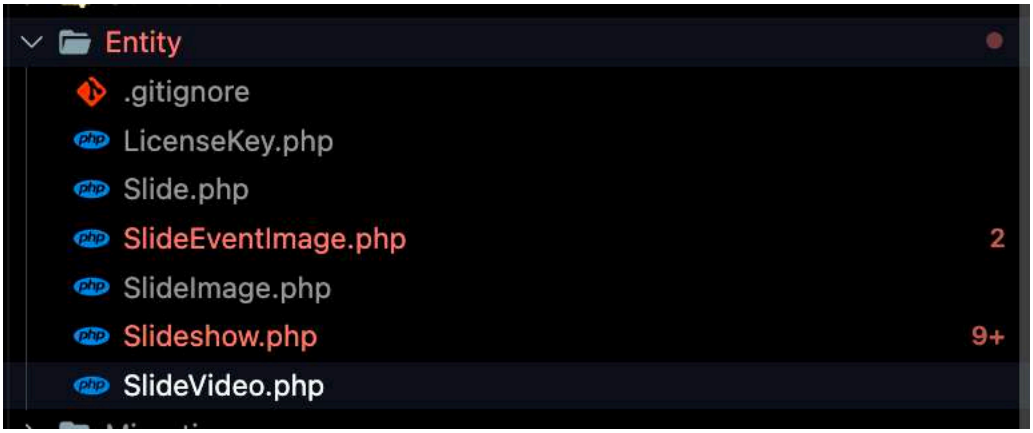

Рис 5.1.2—Основі сутності доданку.

Сутність — це об’єкт, який представляє базові дані (як сказав @perovic: рівно один рядок даних з однієї таблиці, об’єднаний з даними з інших таблиць).

У Symfony вся модель і оновлюється і керується через Doctrine.

Приклад нагенерованої сутності наведений нижче на Рис 5.1.3 або у додатку 1.

Дивлячись на автогенерований код для сутності да репозиторію видно, що вони мають базовий набор методів для роботи з базою даних для здійснення таких операцій як створювання, редагування, видалення та читання даних з бази даних. Фактично, кожний ключ кожної сутності відповідає стовпчику у таблиці MySQL, а сама сутність відповідає кожній табличці. Також symfony має свій інструмент для створення викликів у базу даних. Він оптимізований, зручніший у використанні та дозволяє створювати складні запроси у більш зрозумілій формі. На даному етапі розробки було вирішено зробити зміну інформації на цифровій вивісці у вигляді слайд-шоу, отже основною сутністю є слайдшоу, до котрої у свою чергу прив’язані інші залежні типи сутностей як слайди з відео, картинками та інші. Також було створено сутність LicenseKey, яка відповідає за збереження унікальної пари ключів, необхідної для формування посилання на якесь конкретне слайдшоу.

Також для зв’язування серверної частини і клієнтської є шар бізнес-логіки, котрий відповідає за обробку запитів з клієнтської частини доданку та керування змінами у базі даних. Для цього було створені необхідні контролери, приведені на Рис 5.1.4.

```php
<?php

namespace App\Entity;

use Doctrine\ORM\Mapping as ORM;

/**
 * @ORM\Entity(repositoryClass="App\Repository\SlideVideoRepository")
 */
class SlideVideo
{
    /**
     * @ORM\Id()
     * @ORM\GeneratedValue()
     * @ORM\Column(type="integer")
     */
    private $id;

    /**
     * @ORM\Column(type="string", length=255)
     */
    private $video_link;

    /**
     * @ORM\OneToOne(targetEntity="App\Entity\Slide", cascade={"persist", "remove"})
     * @ORM\JoinColumn(nullable=false,onDelete="CASCADE")
     */
    private $slide;

    public function getId(): ?int
    {
        return $this->id;
    }

    public function getVideoLink(): ?string
    {
        return $this->video_link;
    }

    public function setVideoLink(string $video_link): self
    {
        $this->video_link = $video_link;

        return $this;
    }
```

Рис 5.1.3 — Приклад створенної сутності

Як видно, набір контролерів є більшим. Це викликано ширшим спектром завдань, покладеним на бізнес-логіку серверної частини доданку. Наприклад, FileUploadControlller.php відповідає за збереження на сервері усіх медіа файлів, завантажених через клієнтську частину для відображення на цифрових вивісках.

Окрім цього, також було реалізовано система контролю версій інформації, завдяки якої відбувається оновлення контенту на панелях кожні 5 секунд.

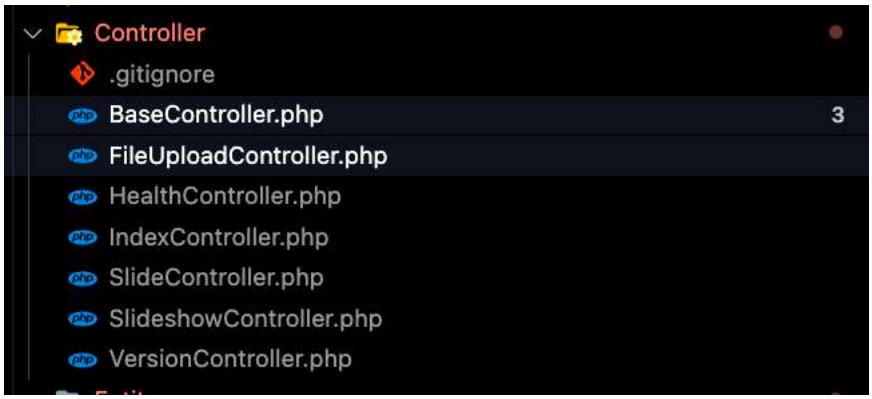

Рис 5.1.4 — Набір створених контролерів

5.2 Реалізація відображення контенту.

Також на базі серверного додатку було створено систему відтворення інформації з бази даних до звичайної веб-сторінки. Вона складається з 3 компонентів, як і усі сторінки в мережі Інтернет : hmtl, css, js.

На Рис 5.2.1 приведена структура цієї системи.

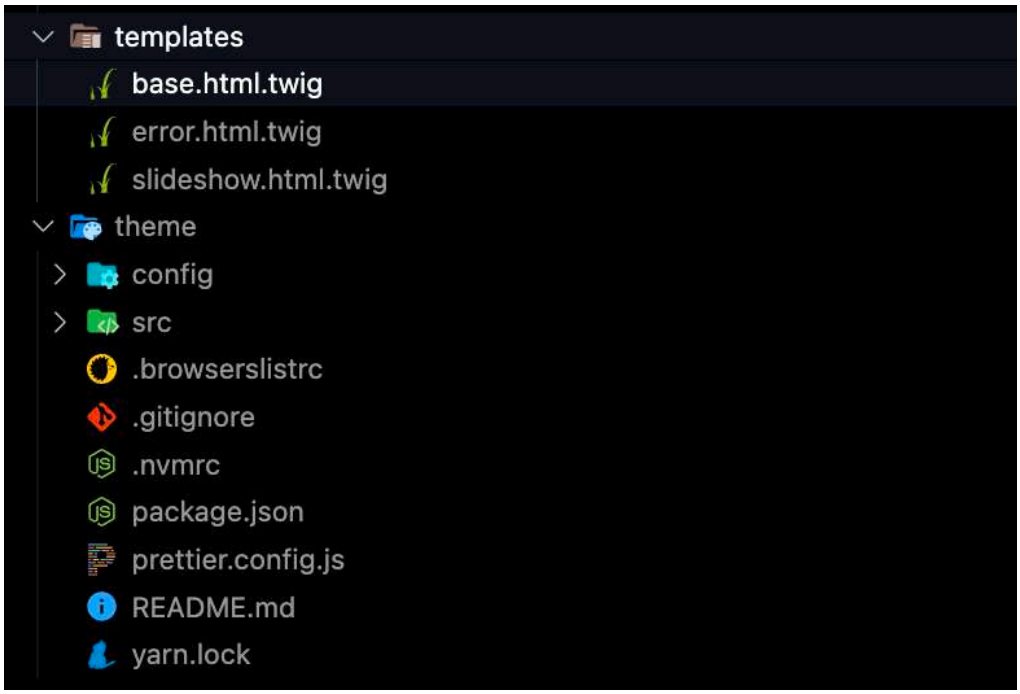

Рис 5.2.1 — Структура системи відображення інформації

У папці templates знаходяться усі частини сторінок, потрібних для відображення інформації. Папка theme має усі стилі, призначені для покращення

відображення та скрипти з розширенням .js, необхідні для коректної роботи як і слайдшоу, так і системи контроля версій. На Рис 5.2.2 приведена реалізація системи контроля версій, алгоритм працює таким чином : після запиту до сервера про існуючу версію слайдшоу і порівняння її з присутньою приймаєтеся рішення, або перезавантажити сторінку, або залишити роботу без зміни.

```
const {origin, search} = location;
const credentials = new URLSearchParams(search);
const uuid = credentials.get('uuid');
const slideshowId = credentials.get('slideshow_id');
const {version} = app.datas;

const checkVersion = () => {
  fetch(`${origin}/api/version/${uuid}/${slideshowId}`)
    .then(res => {
      if (res.ok && res.status < 203) {
        return res.json();
      }
    })
    .then(({version: newVersion}) => newVersion != version && location.reload(true));
};

if (uuid && slideshowId) {
  setInterval(checkVersion, 5000);
}
```

Рис 5.2.2—Реалізація системи контроля версій

5.3 Архітектура клієнтської частини додатку

Використання бібліотеки React має на увазі самостійне формування структури проекту і архітектури. Зазвичай, є розподіл на компоненти, модулі та окремо зберігається бізнес-логіка для роботи з API та стан додатку. Було обрано наступну структуру, показану на Рис 5.3.1.

Розподіл додатку наступний :

- Assets – відповідає за зберігання усіх картинок, використаних у інтерфейсі;
- Config – усі налаштування доданку;
- Lib — усі абстракції та допоміжні функції
- Messages — переводи українською і англійскою
- Modules — основні сутності проекту

- Theme — налаштування стилізації проекту
- Ui — компоненти для повторного використання

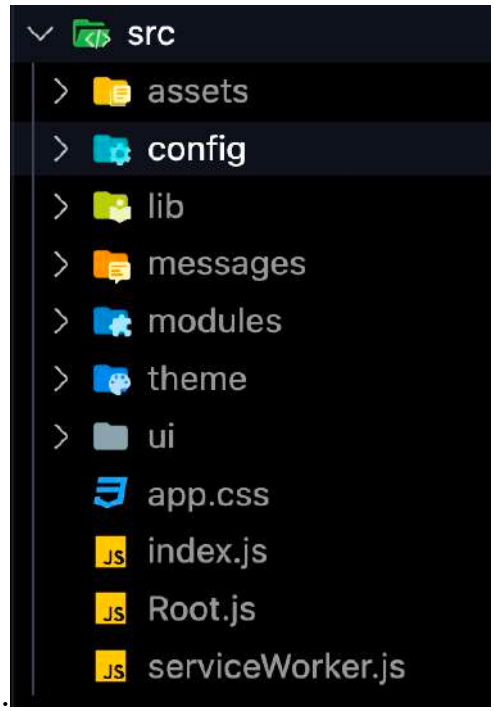

Рис 5.3.1—Структура клієнтської частини

Реалізовано увесь функціонал, зазначений у вимогах до проекту. Додаток розроблено за сучасними стандартами і підходами розробки клієнтської частини. Детальніше клієнтській інтефейс буде описан у п.6.

Для прикладу розглянемо один з основних модулей – слайдшоу.

Структура представлена на Рис 5.4.1.

Папка command відповідає за інтерфейс для встановлення системи на Raspberri PI. Create, Edit, List, View – відповідно за створення, редагування, відображення у списку та окремо більше детальної кожного створеного слайдшоу. Form є абстракцією для форми створення або зміни сладшоу.

У ducks.js реалізована уся бізнес-логіка стосовно роботи з API, роботи зі станом додатку. withSlideshow.js і withSlideshows.js відповідають за зв’язування модуля слайдшоу з іншими, залежними від нього.

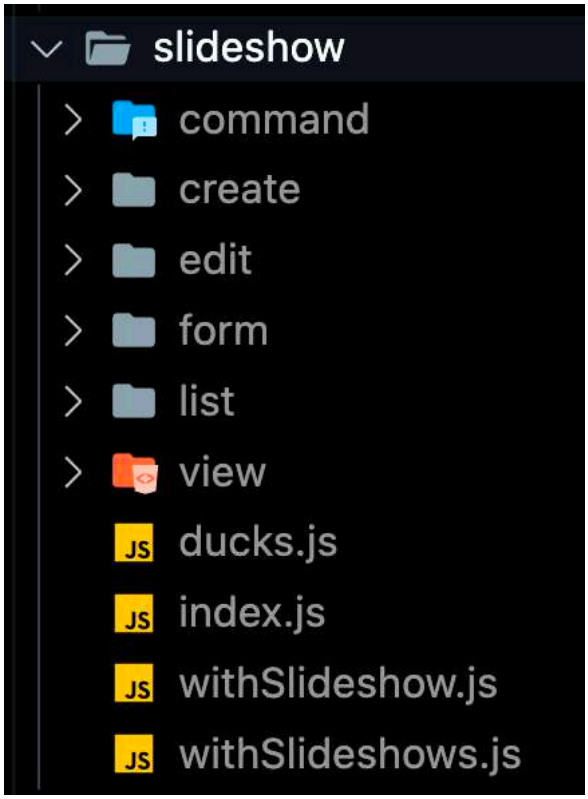

Рис 5.4.2

5.4 Реалізація додатку для Raspberry PI

Було вирішено використати звичайні unix команди для автоматичного запуску браузера на Raspberry PI. Нижче наведено приклад такої команди, яка автоматично генерується з адмін панелі.

```
ssh <IP> -l pi "sudo sed -i '/^@chromium/d' /etc/xdg/lxsession/LXDE-pi/autostart; echo '@chromium-browser --kiosk --disable-features=TranslateUI --no-default-browser-check --no-first-run --disable-infobars --disable-session-crashed-bubble --disable-restore-session-state --disable-plugins --noerrordialogs --force-device-scale-factor=1 --disable-popup-blocking --disable-tab-switcher --disable-translate --check-for-update-interval=31536000 --disable-component-update --incognito https://localhost:8000/?uuid=27becb79-50b7-4c65-8b58-9bdbde28678e&slideshow_id=4' | sudo tee -a /etc/xdg/lxsession/LXDE-pi/autostart"
```

Якщо іти по алгоритму, то маємо :

`sudo sed -i '/^@chromium/d'` – Запуск Chrome браузера.

`/etc/xdg/lxsession/LXDE-pi/autostart` – Додавання Chrome до автозапуску

Далі іде список з параметрів запуску браузера, необхідних для ввімкнення курсора миші, хования інтерфейсу браузера та інші.

`--incognito  https://localhost:8000/?uuid=27becb79-50b7-4c65-8b58-9bdbde28678e&slideshow_id=4'`- Відкриття специфічної адреси для показу інформації.

Отже маємо відкритий браузер у повноекранному режимі на моніторі, панелі, телевізорі на потрібній адресі. По цьому адресу відображається увесь створений контент через адмін панель.

5.5 Висновки

У відповідності до вимог п.2 було реалізовано мережу інформаційних модерованих панелей. Серверна частина відповідає за зберігання інформації, клієнтська частина додатку реалізує користувальницький інтерфейс для створення слайдів зі різним наповненням.

6. РОБОТА КОРИСТУВАЧА З СИСТЕМОЮ

У даному розділі буде приведений опис типової роботи користувача з адмін панеллю.

6.1 Створення слайдів для інформаційної панелі.

Передбачається, що серверна частина встановлена на локальний сервер факультету як і клієнтська частина. Це означає, що адмін панель має бути доступна за адресою локального сервера за виділеним для неї портом.
Наприклад, «`http://localhost:3000/auth/root_uuid/root_license`».
По перше потрібно створити декілька слайдів на вкладці Слайд у адмін панелі. Приклад наведений на рис 6.1.1.Далі користувач має створити на вкладці Екрани одне слайдшоу.(Рис 6.1.2.) та додати необхідні слайди до нього. (Рис 6.1.3.) Після цього потрібно налаштувати Raspberry PI відповідно до інструкцій на сторінці створення слайдшоу. Рис 6.1.4.

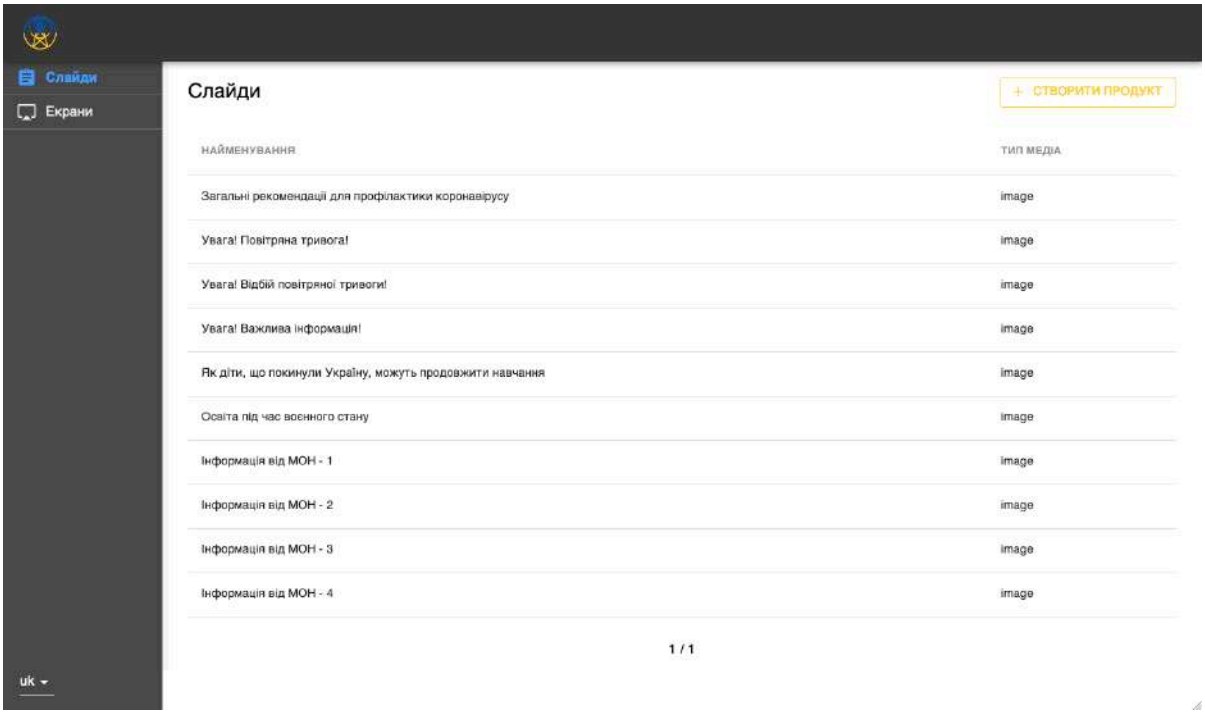

Рис 6.1.1. — Сторінка створення слайдів

Рис 6.1.2. — Сторінка створення слайдшоу(екранів)

Також інсує можливість зміни мови у нижньому лівому куті. Ще на Рис 6.1.3 продемонстровано додаткові можливості керування слайдшоу.

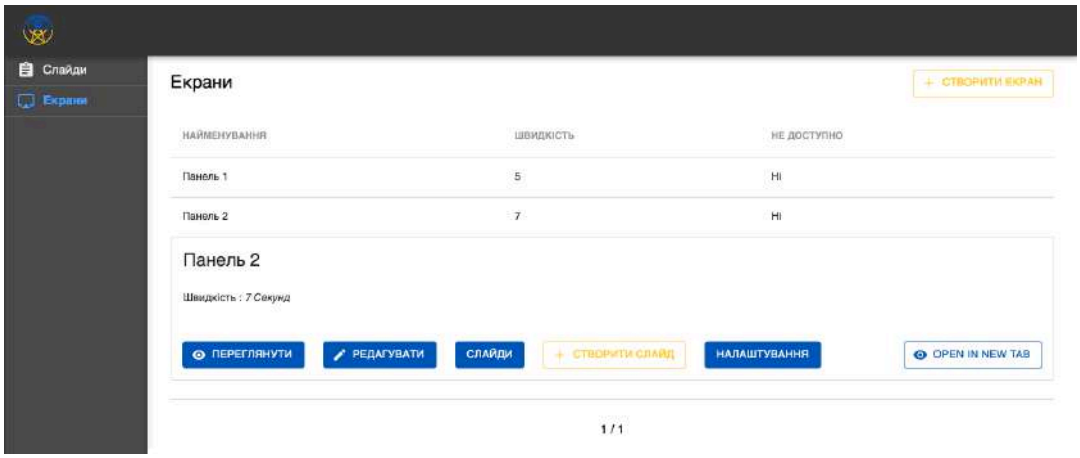

Рис 6.1.3. — Сторінка додавання слайдів до слайдшоу

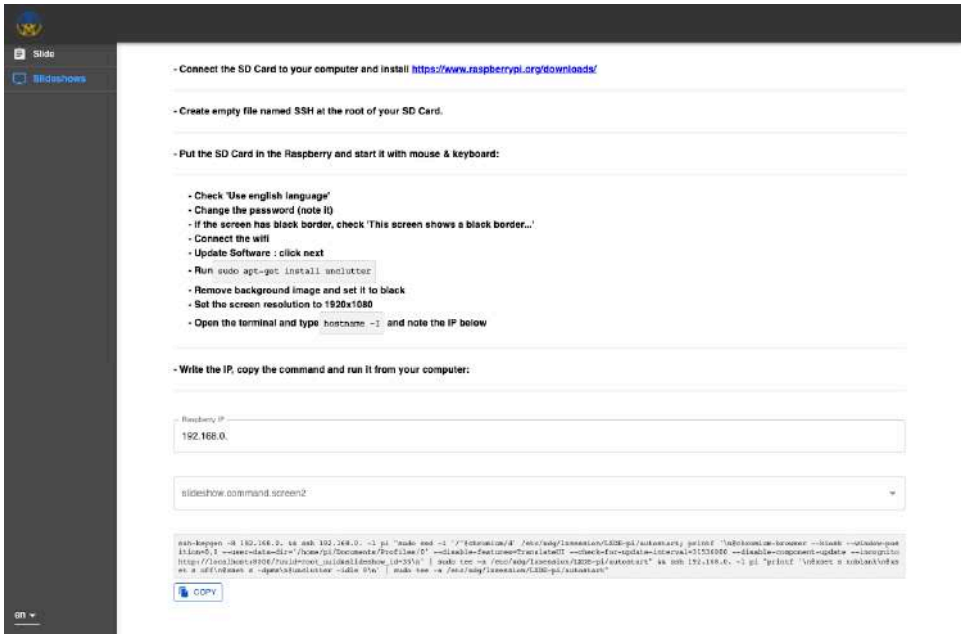

Рис 6.1.4. — Сторінка налаштувань Raspberry PI

6.2 Демонстрація слайдшоу на моніторі.

На Рис 6.2.1 та Рис 6.2.2. представлена робота системи на моніторі комп’ютера. Як видно, є різні напрямки використання такого додатку.

Рис 6.2.1. —Слайд

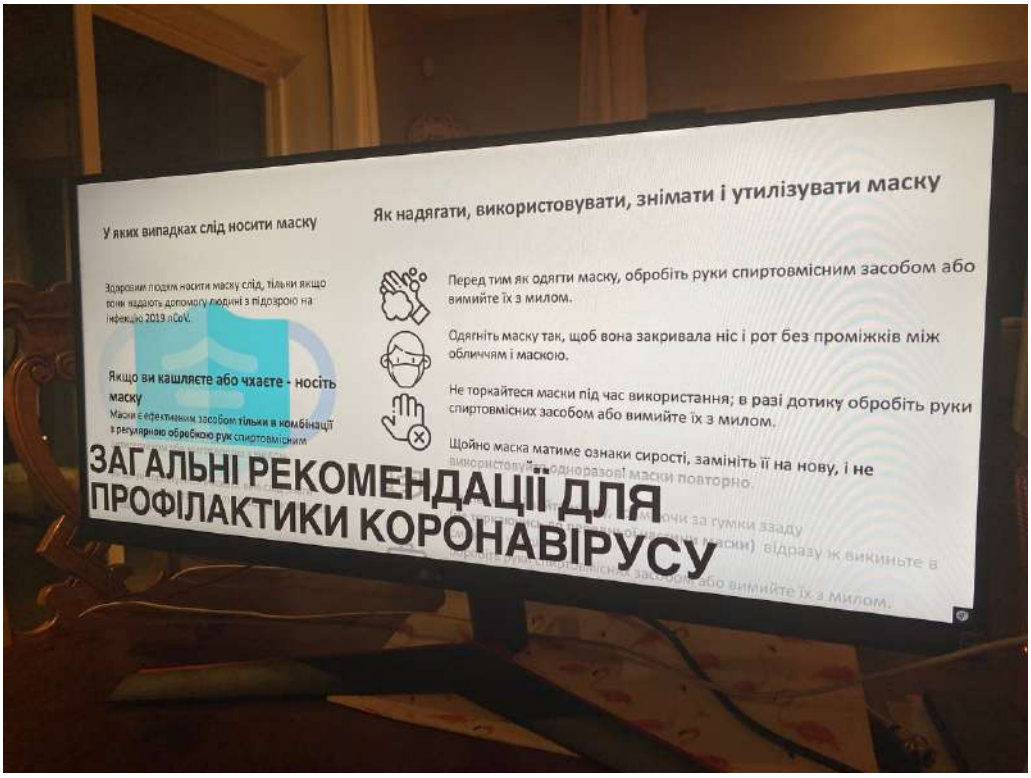

Рис 6.2.2. — Слайд

Також є можливість завантажувати відео та звичайні картинки.

6.3 Висновки

У цьому розділі було продемонстровано основні реалізовані функціональності, показані базові кроки, необхідні для початку роботи з системою. Як видно, інтерфейс має інтуїтивно зрозумілий вигляд і не потребує особливих навиків роботи з комп’ютером.

ВИСНОВОК

У ході цієї дипломної роботи було проаналізовано актуальність розробки нового підходу до реалізації цифрових вивісок. Виходячи з пророблено аналізу виявлено нішу, котра має місце для розвитку, а отже було прийняте рішення про реалізацію. Актуальність такої розробки є високою, оскільки розповсюдження інформації у наші військові часи є іноді запорукою перемоги та збереження життя, як, наприклад, відображення на великому екрані останньої карти до найближчого бомбосховища.

Особливість саме такої реалізації є у спрощених вимогах до місця встановлення, необмеженими можливостями розширення системи та доступних опцій. Такий додаток можна встановити як у приміщенні факультету, маючи тільки звичайний монітор і Raspberry PI молодшого покоління, так й у магазині або іншому закладі, та при відсутності такого обладнання це не складе великих витрат при придбанні. Необмеженість можливостей полягає у різноманітті доступних опцій по розробці нових функцій такої системи, як наприклад відображення останньої карти бомбосховищ, додавання звукового супроводження або ж інтерактивного квесту по вивченню основних запобіжних заходів при хімічної небезпеці.

ПЕРЕЛІК ДЖЕРЕЛ

1. Цифрові вивески [Електронний ресурс] // sourceforge. – Режим доступу: https://sourceforge.net/software/digital-signage/?__cf_chl_tk=5ocs8Ah2.H1PPnkrmGZtCcbJqWtvCSmJX0vCXuGbDUw-1653691174-0-gaNycGzNCFE. – Назва з екрана.
2. Порівняння цифрових вивесок [Електронний ресурс] // softwaretestinghelp. – Режим доступу: https://www.softwaretestinghelp.com/best-digital-signage-software/. – Назва з екрана.
3. Користувальницькі цифрові вивіски-функції [Електронний ресурс] // Шеньчжень Resscreen Промислова група Co., Ltd. – Режим доступу: http://ua.lcdtouchkiosk.com/info/custom-digital-signage-features-52894072.html (дата звернення: 27.05.2022). – Назва з екрана.
4. 4 головних тенденції в області рекламних вивісок 2021 року, про яких вам потрібно знати - Стаття від Гудвіл Дизайн [Електронний ресурс] // Гудвіл Дизайн сервіс. – Режим доступу: https://gudvil.com.ua/ua/blog/4-holovnykh-tendentsiyi-v-oblasti-reklamn/ (дата звернення: 27.05.2022). – Назва з екрана.
5. 7 найкращих методів роботи з цифровими вивісками • TechLila [Електронний ресурс] // TechLila. – Режим доступу: https://www.techlila.com/uk/best-practices-for-digital-signage/ (дата звернення: 27.05.2022). – Назва з екрана.
6. Прогнозування і розробка програм : метод. рек. Київ : Наук. світ, 2000. 468 с.
7. Програмування. К., 1994. 36 с.
8. Структура web-додатку. *Stud*. URL: https://stud.com.ua/97612/informatika/struktura_dodatku (дата звернення: 14.06.2022).

9. Учасники проектів Вікімедіа. Архітектура програмного забезпечення – Вікіпедія. *Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Архітектура_програмного_забезпечення (дата звернення: 14.06.2022).
10. Фронтенд і бекенд: у чому різниця?. *Створення та розробка сайтів на Drupal ma WordPress– Internetdevels*. URL: https://internetdevels.ua/blog/front-end-development-vs-back-end-development (дата звернення: 14.06.2022).
11. Ab M. MySQL language reference. Upper Saddle River : Pearson Education, 2005.
12. Bash scripting cheatsheet. *Devhints.io cheatsheets*. URL: https://devhints.io/bash (date of access: 14.06.2022).
13. Front-end web developer - Learn web development | MDN. *MDN Web Docs*. URL: https://developer.mozilla.org/en-US/docs/Learn/Front-end_web_developer (date of access: 14.06.2022).
14. Linux.org. *Linux.org*. URL: https://www.linux.org/ (date of access: 14.06.2022).
15. Operating system images â□□ Raspberry Pi. *Raspberry Pi*. URL: https://www.raspberrypi.com/software/operating-systems/ (date of access: 14.06.2022).
16. Teach, learn, and make with raspberry pi. *Raspberry Pi*. URL: https://www.raspberrypi.org/ (date of access: 14.06.2022).

ДОДАТОК-1

```php
<?php

namespace App\Entity;

use Doctrine\Common\Collections\ArrayCollection;
use Doctrine\Common\Collections\Collection;
use Doctrine\ORM\Mapping as ORM;

/**
 * @ORM\Entity(repositoryClass="App\Repository\SlideshowRepository")
 */
class Slideshow
{
  /**
   * @ORM\Id()
   * @ORM\GeneratedValue()
   * @ORM\Column(type="integer")
   */
  private $id;

  /**
   * @ORM\Column(type="string", length=255)
   */
  private $name;

  /**
   * @ORM\Column(type="integer")
   */
  private $speed;

  /**
   * @ORM\ManyToOne(targetEntity="App\Entity\Shop")
   * @ORM\JoinColumn(nullable=false)
   */
  private $shop;

  /**
   * @ORM\Column(type="boolean")
   */
  private $disabled;

  /**
   * @ORM\Column(type="integer")
   */
  private $version;

  /**
   * @ORM\ManyToMany(targetEntity="App\Entity\Category")
   */
  private $categories;
```

```php
/**
 * @ORM\ManyToOne(targetEntity="App\Entity\Design")
 */
private $design;

/**
 * @ORM\OneToMany(targetEntity="App\Entity\Slide", mappedBy="slideshow")
 */
private $slides;

public function __construct()
{
  $this->categories = new ArrayCollection();
  $this->slides = new ArrayCollection();
}

public function getId(): ?int
{
  return $this->id;
}

public function getName(): ?string
{
  return $this->name;
}

public function setName(string $name): self
{
  $this->name = $name;

  return $this;
}

public function getSpeed(): ?int
{
  return $this->speed;
}

public function setSpeed(int $speed): self
{
  $this->speed = $speed;

  return $this;
}

public function getShop(): ?Shop
{
  return $this->shop;
}

public function setShop(?Shop $shop): self
{
  $this->shop = $shop;
```

```
  return $this;
}

public function getDisabled(): ?bool
{
  return $this->disabled;
}

public function setDisabled(bool $disabled): self
{
  $this->disabled = $disabled;

  return $this;
}

public function getVersion(): ?int
{
  return $this->version;
}

public function setVersion(int $version): self
{
  $this->version = $version;

  return $this;
}

/**
 * @return Collection|Slide[]
 */
public function getSlides(): Collection
{
  return $this->slides;
}

/**
 * @return Collection|Category[]
 */
public function getCategories(): Collection
{
  return $this->categories;
}

public function addCategory(Category $category): self
{
  if (!$this->categories->contains($category)) {
    $this->categories[] = $category;
  }

  return $this;
}

public function removeCategory(Category $category): self
{
```

```
    if ($this->categories->contains($category)) {
      $this->categories->removeElement($category);
    }

    return $this;
  }

  public function getDesign(): ?Design
  {
    return $this->design;
  }

  public function setDesign(?Design $design): self
  {
    $this->design = $design;

    return $this;
  }
}
```

ДОДАТОК-2

```
import {prop, isEmpty} from 'ramda';
import {createSelector} from 'reselect';
import {createReducer, createAction} from 'lib/redux-helper';
import {takeLatest, put, call, select} from 'redux-saga/effects';
import {get, post, update, del} from 'lib/fetch';
import {setError} from 'modules/network';
import {startLoading, stopLoading} from 'ui/LoadingOverlay';
import {push} from 'connected-react-router';

export const [SLIDES_FETCH, fetchSlides] =
createAction('SLIDES_FETCH', 'allSlides');
export const [SLIDESHOW_ID_SET, setSlideshowId] =
createAction('SLIDESHOW_ID_SET', 'slideshowId');
export const [SLIDE_FETCH, fetchSlide] =
createAction('SLIDE_FETCH', 'id');
export const [SLIDES_SET, setSlides] = createAction('SLIDES_SET',
'items', 'total', 'totalPages');
export const [SLIDES_SET_ITEMS, setAllSlides] =
createAction('SLIDES_SET_ITEMS', 'items');
export const [SLIDES_CREATE, createSlide] =
createAction('SLIDES_CREATE', 'item');
export const [SLIDES_DELETE, deleteSlide] =
createAction('SLIDES_DELETE', 'id');
export const [SLIDES_POST, postSlide] =
createAction('SLIDES_POST', 'formData');
export const [SLIDES_PATCH, patchSlide] =
createAction('SLIDES_PATCH', 'payload');
export const [SLIDES_PAGE_SET, setPage] =
createAction('SLIDES_PAGE_SET', 'page');
export const [SLIDES_ORDER_POST, postSlidesOrder] =
createAction('SLIDES_ORDER_POST', 'order');

const initialState = {
  items: null,
  total: null,
  totalPages: null,
  page: 1,
  slideshowId: null,
};

export const reducer = createReducer(initialState, {
```

```
  [SLIDES_SET]: (s, {items, total, totalPages}) => ({...s, items,
total, totalPages}),
  [SLIDES_SET_ITEMS]: (s, {items}) => ({...s, items}),
  [SLIDES_CREATE]: (s, {item}) => ({...s, items: [item]}),
  [SLIDES_PAGE_SET]: (s, {page}) => ({...s, page}),
  [SLIDESHOW_ID_SET]: (s, {slideshowId}) => ({...s, slideshowId}),
});

const slides = prop('slides');
export const getSlides = createSelector(slides, prop('items'));
export const getTotal = createSelector(slides, prop('total'));
export const getTotalPages = createSelector(slides,
prop('totalPages'));
export const getPage = createSelector(slides, prop('page'));
export const getSlideshowId = createSelector(slides,
prop('slideshowId'));
export const getItemById = createSelector(
  getSlides,
  (_, slideId) => slideId,
  (items, id) => (items ? items.find(el => el.id === +id) : {}),
);

function* fetchSlidesSaga({allSlides = false}) {
  const page = yield select(getPage);
  const slideshowId = yield select(getSlideshowId);

  try {
    yield put(startLoading());

    const {data, total_items, items_per_page, error} = yield call(
      get,
      `slideshows/${slideshowId}/slides?page=${page}`,
    );
    if (error) {
      yield put(setError(error));
      return;
    }

    yield put(stopLoading());
    if (allSlides) {
      yield put(setAllSlides(data));
      return;
    }
```

```
    const totalPages = Math.ceil(total_items / items_per_page);

    yield put(setSlides(data, total_items, totalPages));
  } catch (e) {
    yield put(setError(e));
  }
}

function* fetchSlidesaga({id}) {
  try {
    const slideshowId = yield select(getSlideshowId);
    yield put(startLoading());
    const {data, error} = yield call(get,
`slideshows/${slideshowId}/slides/${id}`);

    if (error || isEmpty(data)) {
      yield put(setError(error));
      return;
    }
    yield put(createSlide(data));
    yield put(stopLoading());
  } catch (e) {
    yield put(setError(e));
  }
}

function* postSlidesSaga({formData}) {
  try {
    yield put(startLoading());
    const slideshowId = yield select(getSlideshowId);

    const {data, error} = yield call(post,
`slideshows/${slideshowId}/slides`, JSON.stringify(formData));
    if (error || isEmpty(data)) {
      yield put(setError(error));
      return;
    }
    yield put(createSlide(data));
    yield put(stopLoading());
    yield
put(push(`/slideshow/${slideshowId}/slides/view/${data.id}`));
  } catch (e) {
    yield put(setError(e));
```

```
  }
}

function* patchSlidesSaga({payload}) {
  const {values, slideId} = payload;
  try {
    yield put(startLoading());
    const slideshowId = yield select(getSlideshowId);

    const {data, error} = yield call(
      update,
      `slideshows/${slideshowId}/slides/${slideId}`,
      JSON.stringify(values),
    );

    if (error || isEmpty(data)) {
      yield put(setError(error));
      return;
    }
    yield put(createSlide(data));
    yield put(stopLoading());
    yield
put(push(`/slideshow/${slideshowId}/slides/view/${data.id}`));
  } catch (e) {
    yield put(setError(e));
  }
}

function* deleteSlidesSaga({id}) {
  try {
    yield put(startLoading());
    const slideshowId = yield select(getSlideshowId);

    const {error} = yield call(del,
`slideshows/${slideshowId}/slides/${id}`);

    if (error) {
      yield put(setError(error));
      return;
    }

    yield put(stopLoading());
    yield put(fetchSlides());
```

```
  } catch (e) {
    yield put(setError(e));
  }
}

function* postSlidesOrderSaga({order}) {
  try {
    // yield put(startLoading());
    const slideshowId = yield select(getSlideshowId);

    const {error} = yield call(post,
`slideshows/${slideshowId}/slides/order`,
JSON.stringify({order}));
    if (error) {
      yield put(setError(error));
      return;
    }
    // yield put(stopLoading());
    // yield put(fetchSlides(slideshowId));
  } catch (e) {
    yield put(setError(e));
  }
}

export function* saga() {
  yield takeLatest([SLIDES_PAGE_SET, SLIDES_FETCH],
fetchSlidesSaga);
  yield takeLatest(SLIDE_FETCH, fetchSlidesaga);
  yield takeLatest(SLIDES_POST, postSlidesSaga);
  yield takeLatest(SLIDES_PATCH, patchSlidesSaga);
  yield takeLatest(SLIDES_DELETE, deleteSlidesSaga);
  yield takeLatest(SLIDES_ORDER_POST, postSlidesOrderSaga);
}
```