МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра інженерії програмного забезпечення

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри, канд. техн. наук,
доцент _____ Є. О. Давиденко
«___»____2022 р.

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

КОМПЛЕКС ЗАСТОСУНКІВ АВТОМАТИЗАЦІЇ БІЗНЕС-ПРОЦЕСІВ КІНОТЕАТРУ

Спеціальність «Інженерія програмного забезпечення»

121 – КРБ.1 – 409.21810912

Студент _____ А. А. Жлуктарьов
«___»_____2022 р.

Керівник ст. викладач кафедри інженерії програмного забезпечення _____ С. Ю. Боровльова
«___»_____2022 р.

Консультант канд. техн. наук, доцент _____ А. О. Алексєєва
«___»_____2022 р.

Миколаїв 2022

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет ім. П. Могили

Факультет комп'ютерних наук

Кафедра інженерії програмного забезпечення

ЗАТВЕРДЖУЮ

Зав. кафедри _____ Є. О. Давиденко

«____» _____2022 р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи бакалавра

Видано студенту групи 409 факультету комп'ютерних наук

Жлуктарьову Антону Андрійовичу

(прізвище, ім'я, по батькові студента)

1. Тема кваліфікаційної роботи

«Комплекс застосунків автоматизації бізнес-процесів кінотеатру»

Затверджена наказом по ЧНУ від «01» грудня 2021 р. № 314

2. Строк представлення кваліфікаційної роботи «____» _____ 2022 р.

3. Очікуваний результат роботи та початкові дані, якщо такі потрібні:

комплекс застосунків для автоматизації бізнес-процесів кінотеатру

4. Перелік питань, що підлягають розробці

– аналіз предметної області та існуючих аналогів;

– специфікація вимог для створюваного ПЗ на основі проведеного аналізу;

– проєктування комплексу взаємопов'язаних сервісів відповідно до вимог;

– моделювання користувацьких інтерфейсів та API;

– розробка backend сервісу для редагування та зберігання даних інформаційної системи;

– створення адміністративної панелі для редагування контенту контент-менеджером;

– розробка frontend застосунку для взаємодії клієнтів з кінотеатром;

– розробка мобільного застосунку для перевірки квитків адміністраторами зали кінотеатру;

– документація створених артефактів.

5. Перелік графічних матеріалів

презентація

6. Завдання до спеціальної частини

аналіз питання охорони праці розробника програмного забезпечення

7. Консультанти:

Консультант	Кафедра (організація)	Частина роботи
канд. техн. наук, доцент Алексеєва Анна Олександрівна	Кафедра екології	Спеціальна частина з охорони праці

Керівник роботи старший викладач Боровльова Світлана Юріївна

(посада, прізвище, ім'я, по батькові)

(підпис)

Завдання прийнято до виконання

Жлуктарьов Антон Андрійович

(прізвище, ім'я, по батькові студента)

(підпис)

Дата видачі завдання «_» _____ 2022 р.

КАЛЕНДАРНИЙ ПЛАН
виконання кваліфікаційної роботи

Тема: «Комплекс застосунків автоматизації бізнес-процесів кінотеатру»

№	Найменування роботи	Початок	Закінчення	Примітки
1.	Затвердження завдання на виконання КРБ	14.01.22	19.01.22	Виконано
2.	Складання календарного плану роботи на весь період виконання КРБ	20.01.22	21.01.22	Виконано
3.	Аналіз існуючих аналогів застосунків для управління кінотеатром	22.01.22	13.02.22	Виконано
4.	Специфікація вимог для створюваного ПЗ	14.02.22	22.02.22	Виконано
5.	Розробка сервісу для зберігання та редагування даних інформаційної системи	23.02.22	20.03.22	Виконано
6.	Реалізація вебзастосунку для взаємодії клієнта з кінотеатром	21.03.22	11.04.22	Виконано
7.	Реалізація вебзастосунку для редагування та наповнення вмісту сервісу контент-менеджером	12.04.22	22.05.22	Виконано
8.	Реалізація системи купівлі та перевірки квитків адміністраторами	23.05.22	01.06.22	Виконано
9.	Формування документації для розроблених застосунків	02.06.22	04.06.22	Виконано

10.	Розробка спеціальної частини з охорони праці	05.06.22	09.06.22	Виконано
11.	Оформлення КРБ та презентації	10.06.22	19.06.22	Виконано
12.	Рецензування	20.06.22	25.06.22	Виконано
13.	Захист КРБ	29.06.22	29.06.22	Виконано

Розробив студент Жлуктарьов Антон Андрійович _____

(прізвище, ім'я, по батькові студента) (*підпис*)

«____» _____ 2022 р.

Керівник роботи ст. викладач Боровльова Світлана Юріївна _____

(ступень, звання, прізвище, ім'я, по батькові) *(підпис)*

«____» _____ 2022 р.

АНОТАЦІЯ

до кваліфікаційної роботи бакалавра

«Комплекс застосунків автоматизації бізнес-процесів кінотеатру»

Студент 409 гр.: Жлуктарьов Антон Андрійович

Керівник: старший викладач Боровльова Світлана Юріївна

Кваліфікаційна робота бакалавра присвячена створенню комплексу застосунків для автоматизації бізнес-процесів кінотеатру. Робота особливо актуальна під час тривалої кризи в індустрії українських кінотеатрів, яка виникла внаслідок війни та пандемії COVID-19, оскільки розроблюване ПЗ дозволяє економити кошти за рахунок автоматизації праці співробітників кінотеатру та приваблює клієнтів підвищеною зручністю використання його сервісу.

Об’єкт роботи – бізнес-процеси кінотеатру.

Предмет роботи – підходи та інформаційні технології для автоматизації бізнес-процесів кінотеатру.

Мета – автоматизація бізнес-процесів кінотеатру шляхом розробки комплексу веб та мобільних застосунків.

У першому розділі розглянуто актуальність автоматизації бізнес-процесів кінотеатру, а також проведено порівняльний аналіз існуючих аналогів. У другому розділі детально розглянуто сценарії використання ПЗ та створено прототипи його інтерфейсу. У третьому розділі досліджуються необхідні архітектурні рішення та концепції з інженерії програмного забезпечення. В четвертому розділі описано процеси кодування спроєктованого застосунку та створення необхідної документації. В останньому розділі розглянуто необхідні норми з охорони праці й техніки безпеки для роботи розробника ПЗ.

В результаті виконаної роботи реалізовано комплекс застосунків та були зроблені висновки щодо можливості використання інформаційних технологій для автоматизації бізнес-процесів кінотеатру.

КРБ викладена на 60 сторінок, вона містить 4 розділи, 29 ілюстрацій, 9 таблиць, 25 джерел в переліку посилань.

Ключові слова: кінотеатр, CQRS, Clean Architecture, SPA, WS.

ABSTRACT

of the Bachelor's Thesis

«Complex of applications for automatization of cinema business-processes»

Student of group 409: Zhluktarov Anton Andriiovych

Supervisor: Senior Instructor Borovlova Svitlana Yuriivna

The Bachelor's Thesis is dedicated to constructing a complex of applications designed to automate cinema business-processes. This paper is especially of interest nowadays during the lasting crisis of Ukrainian cinematic industry, which was caused by the ongoing war and the COVID-19 pandemic, as constructed software allows cutting the costs via automatization of cinema workers' labour, and it also attracts customers because of the increased comfort of cinema's service usage.

The object of thesis are cinema business-processes.

The subject of thesis are approaches and information technologies for automatization of cinema business-processes.

The goal is to automatize cinema business-processes via developing a complex of Web and Mobile applications.

The first section overviews relevance of automatization of cinema business process and performs a comparative analysis of already existing alternatives. The second section describes use-cases of the designed complex of applications and contains user interface prototypes. The third section researches necessary architectural solutions and concepts from the field of software engineering. The fourth section presents processed of coding the constructed complex and creating necessary documentation. The last section overviews the necessary norms of labor protection and safety for the job of software engineer.

As a result of the work performed, a complex of applications has been developed and conclusions about the possibilities of using information technologies for automatization of cinema business processes.

The Bachelor's Thesis spans 60 pages, it contains 4 sections, 29 illustrations, 9 tables, 25 sources in the reference list.

Keywords: cinema, CQRS, Clean Architecture, SPA, WS.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

БД – база даних

КРБ – кваліфікаційна робота бакалавра

ПЗ – програмне забезпечення

СКБД – система керування базами даних

API – Application Programming Interface

AJAX – Asynchronous JavaScript and XML

CQRS – Command and Query Responsibility Segregation

ER – Entity-Relationship

HTTP – HyperText Transfer Protocol

ORM – Object-Relational Mapping

REST – Representational State Transfer

SPA – Single-Page Application

UI – User Interface

UX – User Experience

WS – WebSockets

ВСТУП

У наш час все більше і більше бізнесів оцифровуються та використовують інформаційні технології для створення систем автоматизації, оптимізації вже існуючих бізнес-процесів та створення нових. За допомогою інформаційних систем можна підвищити продуктивність працівників, зменшити або повністю нівелювати ймовірність помилки в їх операціях або заохочувати клієнтів підвищеним комфортом користуванням сервісом. Особливо актуальне оцифровування зараз [25], оскільки під час пандемії вірусу COVID-19:

– багато людей вимушені дотримуватись карантинного режиму, щоб зменшити ризик захворювання, але все ж таки можуть користуватись послугами онлайн;

– велика кількість секторів економіки знаходиться в кризі, а тому вони мають наскільки це можливо зменшити витрати та надавати послуги в цих умовах.

Одним з таких бізнесів є кінотеатр, оскільки використовуючи комплекс спеціально розроблених застосунків можна автоматизувати такі процеси як презентацію та редагування кіноафіші, купівлю та перевірку придбаних квитків тощо.

Також в КРБ розглянуто впровадження протоколу коммунікації WebSockets [15] для роботи механізму купівлі квитків в режимі реального часу, оскількі це відносно нова технологія, яка ще не використовується у сфері кінотеатрів.

Об'єктом роботи є бізнес-процеси кінотеатру.

Предметом роботи є підходи та інформаційні технології для автоматизації бізнес-процесів кінотеатру.

Метою роботи є автоматизація бізнес-процесів кінотеатру шляхом розробки комплексу веб та мобільних застосунків.

Для досягнення поставленої мети необхідно виконати наступні **завдання**:

– аналіз предметної області та існуючих аналогів;

– специфікація вимог для створюваного ПЗ на основі проведеного аналізу;

– проєктування комплексу взаємопов’язаних сервісів відповідно до вимог;

– моделювання користувацьких інтерфейсів та API;

– розробка backend-сервісу для редагування та зберігання даних інформаційної системи;

– створення адміністративної панелі для редагування контенту контент-менеджером;

– розробка вебзастосунку для взаємодії клієнтів з кінотеатром;

– розробка мобільного застосунку для перевірки квитків адміністраторами зали кінотеатру;

– документація створених артефактів.

Апробація результатів. Окремі аспекти роботи були висвітлені на тезах міжнародної конференції «Ольвійський форум»:

– Боровльова С. Ю., Жлуктарьов А. А. Принципи використання Clean Architecture. Ольвійський форум–2022: стратегії країн Причорноморського регіону в геополітичному просторі : тези доп. XVI Міжнар.наук. конф. / Чорном. нац. ун-т ім. Петра Могили, Миколаїв, 23–26 червня 2022 р. Миколаїв : Чорном. нац. ун-т ім. Петра Могили, 2022. С. XX–XX. (прийнято до друку).

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд застосунків-аналогів

Задля створення специфікації вимог для створюваного ПЗ необхідно спершу провести аналіз вже існуючих аналогів. Завдяки цьому можна виділити головні функції, які потрібні користувачам та підмітити можливі недоліки. Успішний проєкт має не тільки реалізовувати відповідні вимоги, але й уникати виявлених недоліків. Внаслідок дослідження суміжних з обраною темою ресурсів виявлено такі аналоги як: Планета Кіно [22] (табл 1.1), Multiplex [19] (табл 1.2) й КіноАфіша [24] (табл 1.3).

1.1.1 Планета Кіно

Таблиця 1.1 – Аналіз вебзастосунку «Планета Кіно»

Назва характеристики	**Значення характеристики**
Назва	Планета Кіно
Відомі особливості реалізації	Frontend використовує мову програмування JavaScript разом з бібліотекою jQuery. Backend переважно генерує сторінки та відсилає її браузеру, але інколи дані відправляються за методологією REST. Для комунікації використовується технологія AJAX.
Функції	– перегляд кіноафіші; – купівля квитків онлайн; – перегляд детальної інформації про фільм; – ознайомлення з використовуваними технологіями перегляду кіно (IMAX, 4DX, Cinetech+);

Кінець таблиці 1.1

Назва характеристики	**Значення характеристики**
Переваги	– автоматично визначає місто користувача; – надає можливість купувати місця онлайн; дозволяє більш гнучко демонструвати планування зали та різні типи місць.
Недоліки	– інформація про куплені місця не оновлюється в режимі реального часу; – інтерфейс незручний та не відповідає стандартам гарного UX.

Головна сторінка застосунку «Планета Кіно» має інтерфейс зображений на рисунку 1.1.

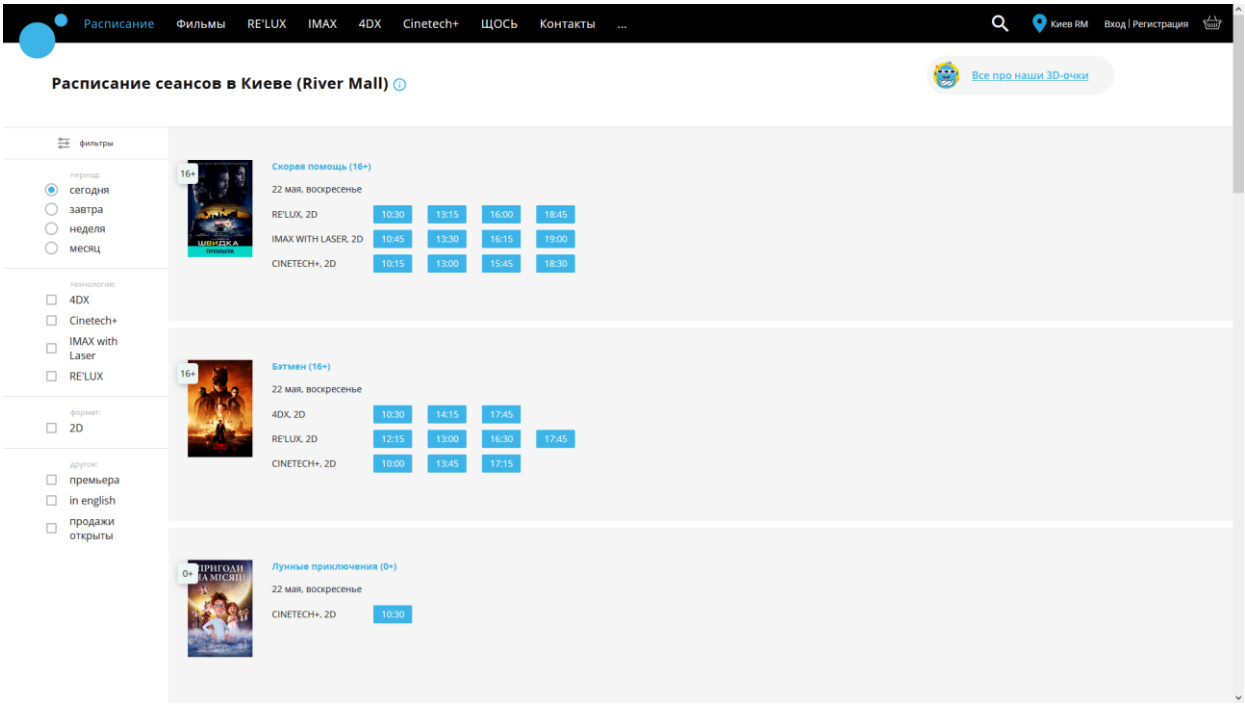


Рисунок 1.1 – Інтерфейс застосунку «Планета Кіно»

Як можна побачити з наведеного скріншоту, дизайн головної сторінки застосунку неефективний, оскільки одночасно можна побачити лише 3 фільми з афіші.

1.1.2 Multiplex

Таблиця 1.2 – Аналіз вебзастосунку «Multiplex»

Назва характеристики	Значення характеристики
Назва	Multiplex
Відомі особливості реалізації	Frontend використовує мову програмування JavaScript разом з бібліотекою jQuery. Backend генерує сторінки та відсилає їх браузеру. Використовується AJAX.
Функції	– перегляд кіноафіші; – купівля квитків онлайн; – перегляд детальної інформації про фільм;
Переваги	– автоматично визначає місто користувача; – надає можливість купувати квитки онлайн; – має лаконічний та сучасний дизайн.
Недоліки	– інформація про куплені місця не оновлюється в режимі реального часу; – сайт погано адаптується до малого розміру екрану.

Інтерфейс кіноафіші застосунку «Multiplex» представлений на рисунку 1.2.

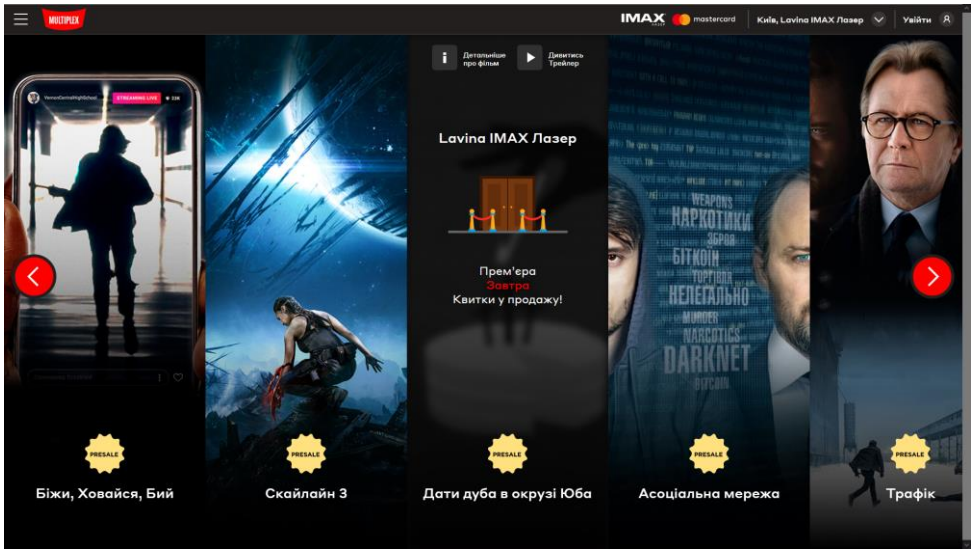


Рисунок 1.2 – Інтерфейс застосунку «Multiplex»

Кіноафіша представлена у вигляді каруселі з постерів фільмів, при наведенні на які виводиться час найближчого сеансу та додаткові посилання.

1.1.3 КіноАфіша

Таблиця 1.3 – Аналіз вебзастосунку «КіноАфіша»

Назва характеристики	**Значення характеристики**
Назва	КіноАфіша
Відомі особливості реалізації	Frontend використовує мову програмування JavaScript разом з бібліотекою jQuery. Backend генерує сторінки та відсилає її браузеру. Для комунікації використовується технологія AJAX.
Функції	– перегляд кіноафіші; – купівля квитків онлайн; – коментування фільмів; – відображення новин кіноіндустрії.
Переваги	– автоматично визначає місто користувача; – надає можливість купувати квитки онлайн; – має потужну базу даних з світу кіно.
Недоліки	– інформація про куплені місця не оновлюється в режимі реального часу; – для покупки квитків сайт перенаправляє на зовнішній ресурс, таким чином, КіноАфіша залежить від його стану для виконання однієї з своїх основних функцій.

Сервіс «КіноАфіша» надає інформацію про фільми в прокаті у представленні наведеному на рисунку 1.3.

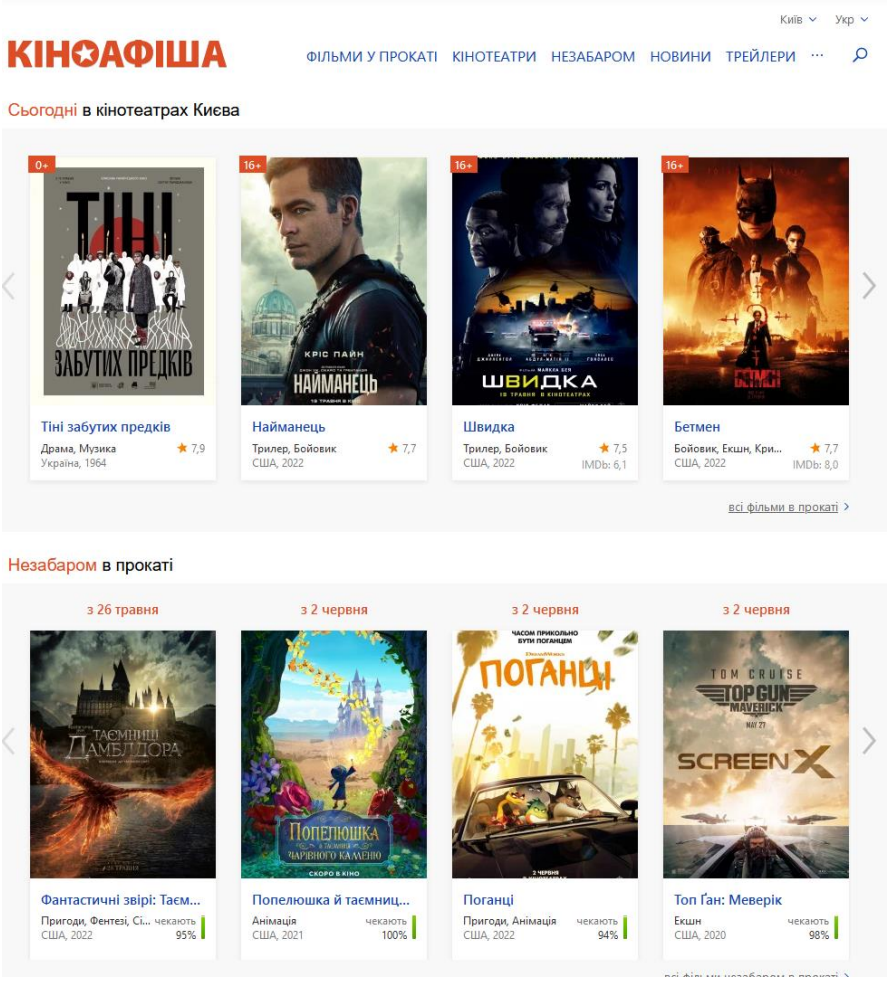


Рисунок 1.3 – Інтерфейс застосунку «КіноАфіша»

Кіноафіша представлена двома каруселями: одна показує вже вийшовші в прокат фільми, а інша – заплановані фільми.

1.2 Аналіз створюваної системи

Проаналізувавши аналоги, можемо відокремити мінімальний обсяг головних функції, які потребують клієнти як-от:

– перегляд кіноафіші;

– перегляд детальної інформації про кінострічку;

– купівля білетів онлайн.

Також необхідно звернути увагу на головниі недоліки аналогів:

– інформація про куплені місця не передається у режимі реального часу і, таким чином, це може призводити до ситуацій, коли два чи більше клієнта

бажають викупити одне й те ж місце, або навпаки: вже вільне місце буде помічене як куплене, якщо користувач відвідав сторінку оформлення замовлення квитку до того як інший користувач її скасував;

– інтерфейс застосунку погано адаптується під малі розміри екранів, що особливо незручно для мобільних користувачів, які складають велику частку всіх користувачів Інтернету.

Комплекс застосунків має одночасно працювати з кількома користувачами. Він має працювати 99% часу та сумісним з найпоширенішими браузерами (Google Chrome, Firefox, Opera, Edge) [23] та ОС (Windows, Android, iOS) [21] тощо.

В результаті функціонального аналізу створювану систему описано в таблиці 1.4.

Таблиця 1.4 – Опис створюваної системи

Користувачі	– клієнт; – контент-менеджер; – адміністратор зали.
Функції	– перегляд кіноафіші; – перегляд детальної інформації про кінострічку; – купівля квитків онлайн в режимі реального часу; – додавання та редагування кіноафіші; – додавання та редагування наявних в кінотеатрі зал; – призначення сеансів; – перевірка квитків на справжність та коректність; – авторизація адміністраторів зали та контент менеджерів у відповідних частинах комплексу застосунків.

Задовільнивши вказані функції, ствоерний застосунок можна вважати кращим за існуючі застосунки-аналоги.

1.3 Опис етапів розробки проєкту

При використанні інженерного методу до створення застосунків можна виявити низку стандартних етапів, через які треба пройти для створення найбільш вдалого, задовольняючого всі вимоги та придатного для розширення ПЗ:

а) Специфікація вимог. На основі аналізу застосунків-конкурентів, співпраці з замовником та власного дослідження створюється список функціональних та нефункціональних вимог яким система, що створюється, має відповідати. Можуть бути зазначені бажаний архітектурний стиль, інструменти для розробки, очікувані обмеження та програмні, апаратні та користувацькі інтерфейси тощо.

б) Прототипування користувацького інтерфейсу. Етап, в рамках якого швидко продумуються екрани застосунку. Головна увага приділяється правильності структури, зручності та інтуїтивності використання інтерфейсу й через декілька ітеративних покращень отримується бажаний схематичний вигляд UI з урахуванням бажаного UX.

в) Детальна розробка дизайну. Створені прототипи уточнюють, конкретизують потрібні іконки та зображення, підбирають потрібні кольорові палітри враховуючи принципи доступності та теорію кольору. Також демонструється яким чином програма має реагувати на різні ситуації:

1) Як інтерфейс має адаптуватись до різних розмірів екранів?
2) Як оброблювати занадто довгий текст?
3) Як виводити помилки?

г) Моделювання сутностей та їх відношення. Моделюється концептуальна схема сутностей, які будуть представлені в системі, що розроблюється. Для цього зазвичай використовують такі нотації як UML та ERD, які потім будуть використані розробникам для створення відповідної бази даних.

д) Розробка архітектури системи. Визначивши концептуальну модель сутностей та спираючись на специфікацію вимог, узгоджується загальний вигляд системи, її підсистем та яким чином вони взаємодіють між собою. Це особливо важливо для створення легкого в супроводі та розширення застосунку.

е) Програмна реалізація. Спираючись на визначену вище інформацію система кодується.

ж) Налаштування системи розгортання. Налаштовуються процеси за допомогою яких система має впроваджуватись в експлуатацію, враховуючи взаємодію створених сервісів та потенційні відмінності у програмних середовищах.

з) Тестування та виправлення виявлених недоліків. Створену систему перевіряють на дефекти методами тестування та виправляють знайдені. Цей процес має ітеративний характер і після кожного виправлення недоліків система має знову бути протестована для підтвердження її справності.

и) Документування. На цьому етапі створюються документи, записи та коментарі, які мають допомогти супроводжуючим розробникам у полагодженні несправностей та розширенні системи, а також зрозуміла, дружелюбна до користувача інструкція з використання.

1.4 Специфікація вимог до створюваного проєкту

ПРИЗНАЧЕННЯ ТА МЕЖІ ПРОЄКТУ

Призначення системи (застосунку), для якої розробляється програмне забезпечення

Система призначена для автоматизації бізнес-процесів кінотеатру за рахунок використання вебтехнологій.

Погодження, що ухвалені в програмній документації

Було погоджено, що для передачі даних про куплені місця має використовуватись протокол WS.

Межі проєкту ПЗ

Проєкт має реалізовувати основні функції застосунків-аналогів та вирішувати їх головні недоліки: відсутність оновлення даних про статус доступності місць на певний сеанс й погану адаптивність під малі екрани застосунку для клієнтів.

ЗАГАЛЬНИЙ ОПИС

Сфера застосування

Створюваний проєкт має використостовуватись кінотеатрами.

Загальна структура і склад системи

Система має складатись з БД, backend сервісу для забезпечення API кінотеатру, frontend сервісу для взаємодії клієнта з кінотеатром, frontend сервісу адміністративної панелі для контент менджерів, мобільного застосунку для перевірки квитків адміністраторами зали та зворотнього проксі.

ВИМОГИ ДО ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ

Джерела і зміст вхідної інформації (даних)

Джералами даних є користувачі ПЗ: як клієнти, так і контент-менеджери кінотеатру. Клієнти купують квитки, а контент-менеджери заповнюють необхідні форми для детальної інформації про фільми, сеанси, зали тощо. Також для отримання рейтингу фільму використовується зовнішній ресурс – IMDB API [2].

Вимоги до способів організації, збереження та ведення інформації

Інформація має зберігатись в реляціонній базі даних.

ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Архітектура програмної системи

Програмна система має складатися з чотирьох ланок: клієнтські застосунки, зворотній проксі-сервер, API сервер та БД.

Мережне програмне забезпечення

Для комунікації має бути використаним зворотній проксі-сервер Nginx.

Програмне забезпечення ведення інформаційної бази

Ведення інформаційної бази має проводитись за допомогою системи управління базами даних Microsoft SQL Server.

Мова і технологія розробки ПЗ

API сервер має використовувати мову програмування C# та використовувати вебфреймворк ASP.NET Core. Клієнтські застосунки мають використовувати мову програмування TypeScript. Застосунок для користувача-клієнта має використовувати фреймворк React, адміністративна панель має бути створена за допомогою фреймворку Vue, а мобільний застосунок для адміністраторів зали має бути побудованим на технології React Native.

ВИМОГИ ДО ЗОВНІШНІХ ІНТЕРФЕЙСІВ

Інтерфейс користувача

Користувацький інтерфейс має задовольняти стандартні UI та UX вимоги. Маніпуляція розкладом сеансів має бути реалізована у вигляді діаграми Ґанта [18], а розклад зали має бути представленим у вигляді спеціально адаптованого компоненту-таблиці.

Програмний інтерфейс

Застосунок має надавати програмний інтерфейс за допомогою окремого API сервера. Основні операції з сутностями мають бути визначеними за методологією REST. Також треба створити окремий ресурс для надання метаінформації про програмний інтерфейс у формі OpenAPI документу. Всі відповіді від API мають бути серіалізованими у форматі JSON.

Комунікаційний протокол

Комунікація з застосунком має проводитися за допомогою протоколу HTTP. В місцях де потрібен режим оновлення даних в реальному часі передбачено використання протоколу WS.

ВЛАСТИВОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Доступність

Від ПЗ вимагається доступність для будь-якого користувача, в якого є доступ до мережі Інтернет.

Супроводжуваність

Комплекс застосунків має бути легко супроводжуваним та придатним для подальшої модифікації чи доповнення.

Переносимість

Програмне забезпечення має бути кроссплатформеним. Клієнтські застосунки можуть бути використані будь-яким ПК, мобільним телефоном, планшетом тощо. а серверна частина має розгортатися на будь-якому настільному чи хмарному сервері.

Продуктивність

Система має підтримувати обробку від 100 запитів в секунду, кожен з яких має оброблятись до 4 секунд.

Надійність

Створюваний проєкт має належним чином оброблювати помилки та продовжувати справно працювати навіть у разі виникнення нефатальних помилок та демонструвати відповідну помилку, якщо така є. Потенційно небезпечні для системи помилки не мають бути показані звичайним клієнтам кінотеатру.

Безпека

Адміністративна панель, мобільний застосунок для адміністраторів зали та використовуване ними API має вимогати відподної авторизації.

Висновки до розділу 1

В першому розділі КРБ проаналізовано існуючі застосунки-аналоги для автоматизації бізнес-процесів кінотеатру. Виявлено їх головні функції, переваги, недоліки та деталі реалізації. На основі проведеного аналізу створено список функцій, які створюване ПЗ має виконувати, та ролі його користувачів. Зазначено як виправити недоліки застосунків-аналогів.

Також в розділі сформовано специфікацію вимог до ПЗ, що розробляється, в якій зазначено описано призначення, загальні характеристики

проєкту та вимоги до його інформаційного та програмного забезпечення.

2 МОДЕЛЮВАННЯ ФУНКЦІЙ ТА ПРОТОТИПУВАННЯ КОРИСТУВАЦЬКОГО ІНТЕРЕЙСУ

2.1 Діаграма прецедентів

Для представленння функцій системи, її користувачів та взаємодії між ними створено дігграму прецедентів, яку наведено на рисунку 2.1.

Рисунок 2.1 – Діаграма прецедентів

Діаграма прецедентів [5] – це граф, на якому позначаються актори, варіанти використання проєктуємої системи, відношення між ними та асоціації акторів з їх варіантами використання.

В стандартній нотації для тлумачення діаграми визначено такі концепії:

а) актор – це користувач системи будь-то людина чи зовнішня система;

б) прецедент – це послуга, виконання якої забезпечує система. Варто зазначити, що прецеденти на діаграмі мають бути зображені узагальнено, оскільки діаграма призначена для поверхневого опису функціоналу.

2.2 Детальний опис сценаріїв використання

Для більш детальної специфікації вимог до кожного прецеденту варто представити їх у вигляді сценаріїв використання [10]. Сценарії використання створюваної системи описано на таблицях 2.1-2.5.

Таблиця 2.1 – Опис сценарію авторизації користувача

Назва	Авторизація
Актори	Контент-менеджер, адміністратор зали
Мета виконання	Надати доступ до функціоналу, який заборонено використовувати клієнтам.
Передумови	Користувач попередньо неавторизований.
Успішний сценарій: а) Контент-менеджер або адміністратор зали переходить на екран з формою авторизації в їх відповідних застосунках. б) Користувач вводить свій логін та пароль. в) Користувач відправляє форму за допомогою POST методу. г) Система шукає користувача з таким ж логіном в базі даних. д) Система шифрує відправлений користувачом пароль, використовуючи криптографічну сіль. е) Зашифровані версії паролів порівнюються. ж) Користувачу видається cookie, який свідчить, що користувач авторизований.	
Результат	Користувач має доступ до функціоналу, який заборонено використовувати клієнтам. Йому видано cookie, який буде надаватись з кожним наступним запитом користувача.

Кінець таблиці 2.1

Розширення:	
а	Користувач некорректно заповнює форму: не вводить логін і/або пароль; введені дані не задовільняють правила для логіну чи паролю тощо. Результат: відповідні поля підсвічуються червоним та під ними виводяться повідомлення, які описують виявлені помилки.
б	Користувача з введеним логіном не знайдено. Результат: виводиться відповідна помилка у повідомлені зверху форми.
в	Зашифровані версії паролів не співпадають. Результат: виводиться помилка про неправильність паролю у повідомлені зверху форми.

Таблиця 2.2 – Опис сценарію купівлі квитка

Назва	Купівля квитка
Актори	Клієнт
Мета виконання	Купити виток на потрібний сеанс на обрані місця
Передумови	Клієнт обрав фільм, квиток на який він хоче купити.
Успішний сценарій: а) Клієнт переходить на екран купівлі квитка. б) Клієнт обирає потрібний сеанс. в) Клієнт обирає бажані місця в залі. г) Клієнт вводить свою поштову адресу в форму. д) Клієнт натискає кнопку купівлі квитка. е) Клієнт сплачує замовлення будь-яким доступним шляхом. ж) Система надсилає клієнтові квиток з QR-кодом номера замовлення на зазначену поштову адресу.	

Кінець таблиці 2.2

Результат	Клієнт успішно купив квиток та отримав на пошту електроного листа, який підтверджує купівлю та містить QR-код з номером замовлення. Кошти за купівлю перейшли на рахунок кінотеатру.
Розширення:	
а	Клієнт вводить некоректну поштову адресу. Результат: відповідне поле підсвічується червоним та під ним виводиться повідомлення з помилкою.
б	Клієнт переходить до платіжного шлюзу, але не сплачує квиток. Результат: через 10 хвилин після переходу клієнту надсилається електроний лист, який сповіщає про помилку. В базі даних його замовлення позначається як несплачене. Обрані місця знову стають доступними для купівлі.

Таблиця 2.3 – Опис сценарію перевірки квитка

Назва	Перевірка квитка
Актори	Адміністратор зали
Мета виконання	Перевірити квиток на оригінальність та відповідність залі
Передумови	Адміністратора зали авторизовано в системі
Успішний сценарій: а) Адміністратор зали обирає залу, яку він обслуговує. б) Адміністратор зали сканує камерою телефону QR-код, наданий клієнтом кінотеатру. в) Система підтверджує оригінальність квитка та його відповідність обраній залі.	

Кінець таблиці 2.3

Результат	На екрані застосунку з’являється повідомлення про коректність просканованого квитка
Розширення:	
а	Клієнтський QR-код вказує на неіснуюче або несплачене замовлення. Результат: на екрані застосунку з’являється відповідне повідомлення.
б	Клієнтський QR-код вказує на замовлення, сеанс якого проходить в іншій залі. Результат: на екрані застосунку з’являється відповідне повідомлення з зазначенням правильної зали.

Таблиця 2.4 – Опис сценарію створення сеансу

Назва	Створення сеансу
Актори	Контент-менеджер
Мета виконання	Створити сеанс фільму на обраний час в обраній залі
Передумови	Контент-менеджера авторизовано в системі
Успішний сценарій: а) Контент-менеджер обирає з комбінованого списку потрібний фільм. б) Контент-менеджер натискає кнопку створення сеансу. в) Система створює сеанс та поміщає його у рядок нерозподілених сеансів. г) Контент-менеджер переносить сеанс у потрібну залу на потрібне місце.	
Результат	Дані про створений сеанс збережено в базі даних.
Розширення:	
а	Контент-менеджер намагається приначити сеанс на вже зарезервований час в обраній залі. Результат: застосунок запобігає цьому та залишає переміщуваний сеанс на його попередньому місці.

Таблиця 2.5 – Опис сценарію редагування зали

Назва	Редагування зали
Актори	Контент-менеджер
Мета виконання	Оновити назву та/або схему зали
Передумови	Контент-менеджера авторизовано в системі
Успішний сценарій: а) Контент-менеджер вводить потрібну назву зали. б) Контент-менеджер регулює сітку схеми зали: вказує необхідну кількість стовпчиків та рядків. в) Контент-менеджер призначає тип кожному місцю за допомогою кліків: кожен клік змінює тип місця у такому порядку: 1) пусте; 2) звичайне; 3) VIP; 4) диван. г) Контент-менеджер натискає кнопку збереження змін.	
Результат	Дані про залу оновлено в базі даних.

Описані сценарії можна використати під час перевірки застосунку на відповідність поставленим вимогам.

2.3 Прототипування макетів користувацького інтерфейсу

Наступним етапом розробки проєкту є прототипування макетів користувацького інтерфейсу. Для цього було обрано найважливіші екрани для кожного застосунку, розроблено їх дизайн та вербально описано його концепцію та нюанси. Прототипування виконано за допомогою ПЗ Balsamiq Cloud [3].

2.3.1 Сторінка кіноафіші

Сторінка кіноафіші показує фільми, які знаходяться у прокаті. Прототип її інтерфейсу зображено на рисунку 2.2.

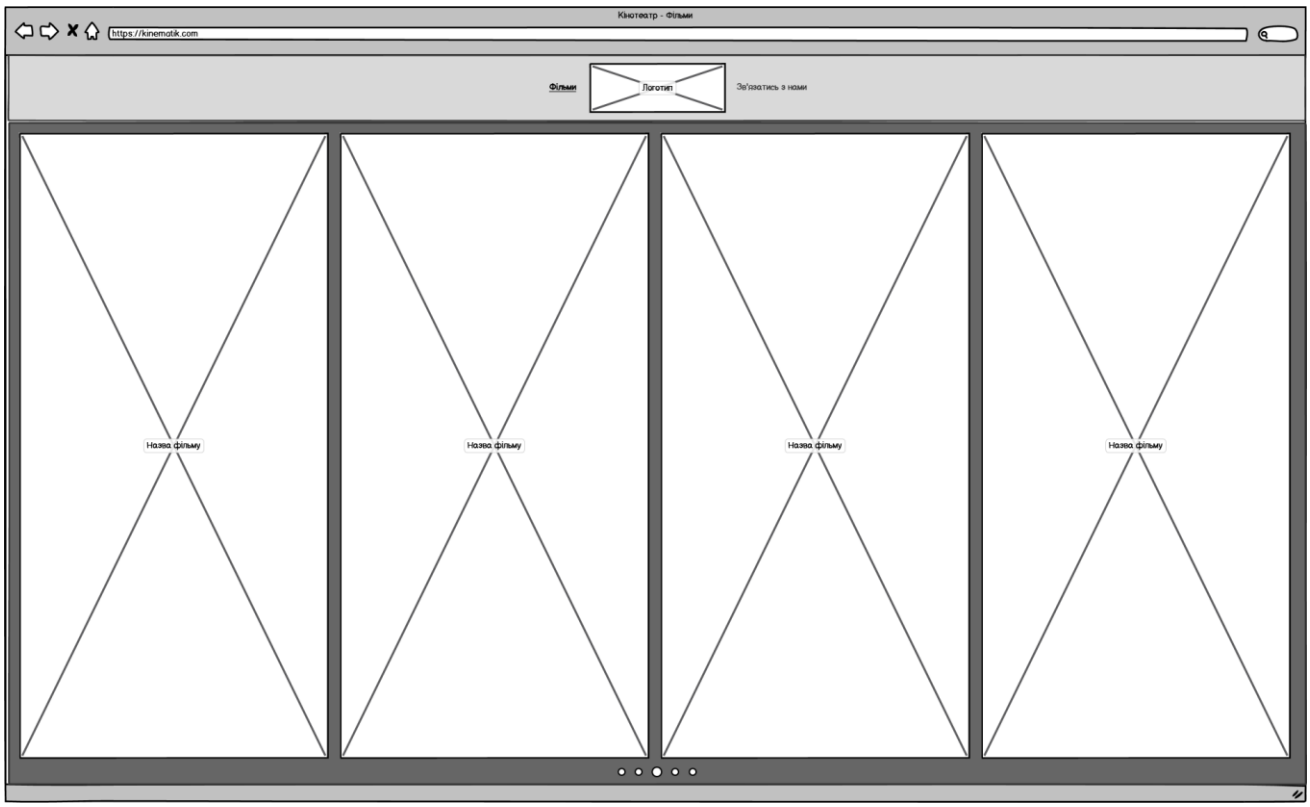


Рисунок 2.2 – Сторінка кіноафіші

Користувач бачить горизонтальну карусель з чотирма великими зображеннями постерів фільмів, назви яких розташовані посередині. Карусель автоматично прокручується, але зупиняється при наведенні на будь-який постер. Також нею можна управляти за допомогою круглих кнопок знизу, одна з яких показує на якій позиції знаходиться карусель. На всіх сторінках клієнтського та адміністративного застосунку зверху розташована навігаційна панель з посиланнями на основні сторінки. Активна сторінка виділяється підкресленням та коліром, а посередині панелі знаходиться логотип кінотеатру, який використовує створене ПЗ.

2.3.2 Сторінка опису фільму

Сторінка опису фільму показує такі відомості про фільм як:

– назва;

– опис;

– рейтинг;

– жанри;

– тривалість;

– альтернативний постер;

– цікаві факти.

Інтерфейс цієї сторінки показаний на рисунку 2.3.

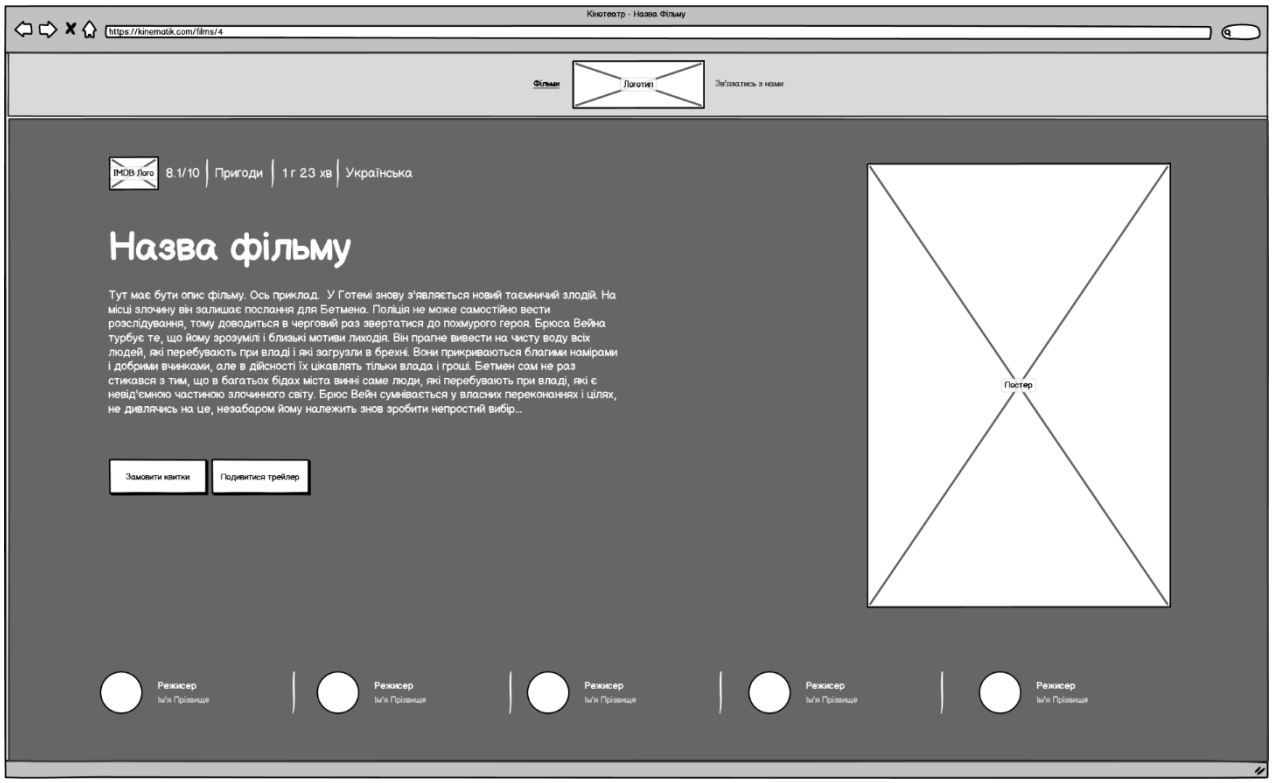


Рисунок 2.3 – Сторінка опису фільму

Метадані знаходяться зверху, а під ними розташовані назва та опис фільму. Постер розміщений окремо справа, задля того щоб приваблювати увагу клієнта. Під описом фільму знаходяться дві кнопки: одна перенаправляє користувача на сторінку купівлі квитків, а інша показує трейлер у модальному вікні. У «підвалі» сторінки розташовані блоки цікавих фактів про фільм.

2.3.3 Сторінка купівлі квитків

Сторінка покупки квитків дозволяє клієнту кінотеатра переглянути всі сеанси на обраний фільм та купити квиток онлайн на бажані місця. Макет цієї сторінки представлено на рисунку 2.4.

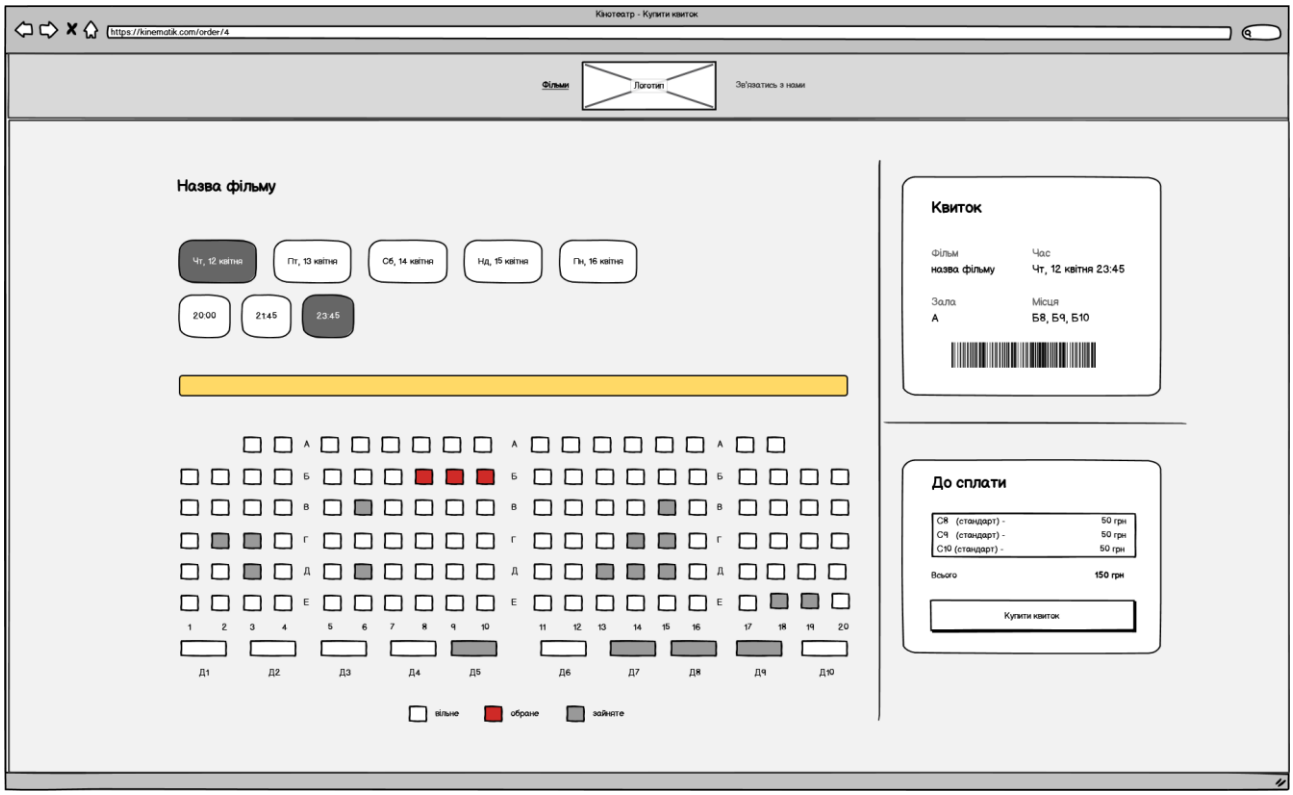


Рисунок 2.4 – Сторінка покупки квитків

Зверху сторінки виводиться назва обраного фільму, під якої розташовані кнопки вибіру дати та часу потрібного сеансу.

Під ними знаходиться схема стану місць зали,в якій місця зали представлені різними видами іконок для різних типів місць: звичайне,VIP або диванне. Знизу схеми відображається легенда, яка тлумачить яким чином виділяються різні стани місць.

Справа знаходяться дві панелі: одна дозволяє стилістично показати створений квиток, щоб користувач мав змогу перевірити чи правильну дату, час та місця він обрав, а друга виводить ціну за кожне місце, суму замовлення та містить кнопки для купівлі квитка.

2.3.4 Сторінка редагування даних про фільм

Сторінка редагування фільму дозволяє контент-менеджерам кінотеатру редагувати відомості про фільм, які потім виводяться користувачам при перегляді кіноафіші. Її інтерфейс зображено на рисунку 2.5.

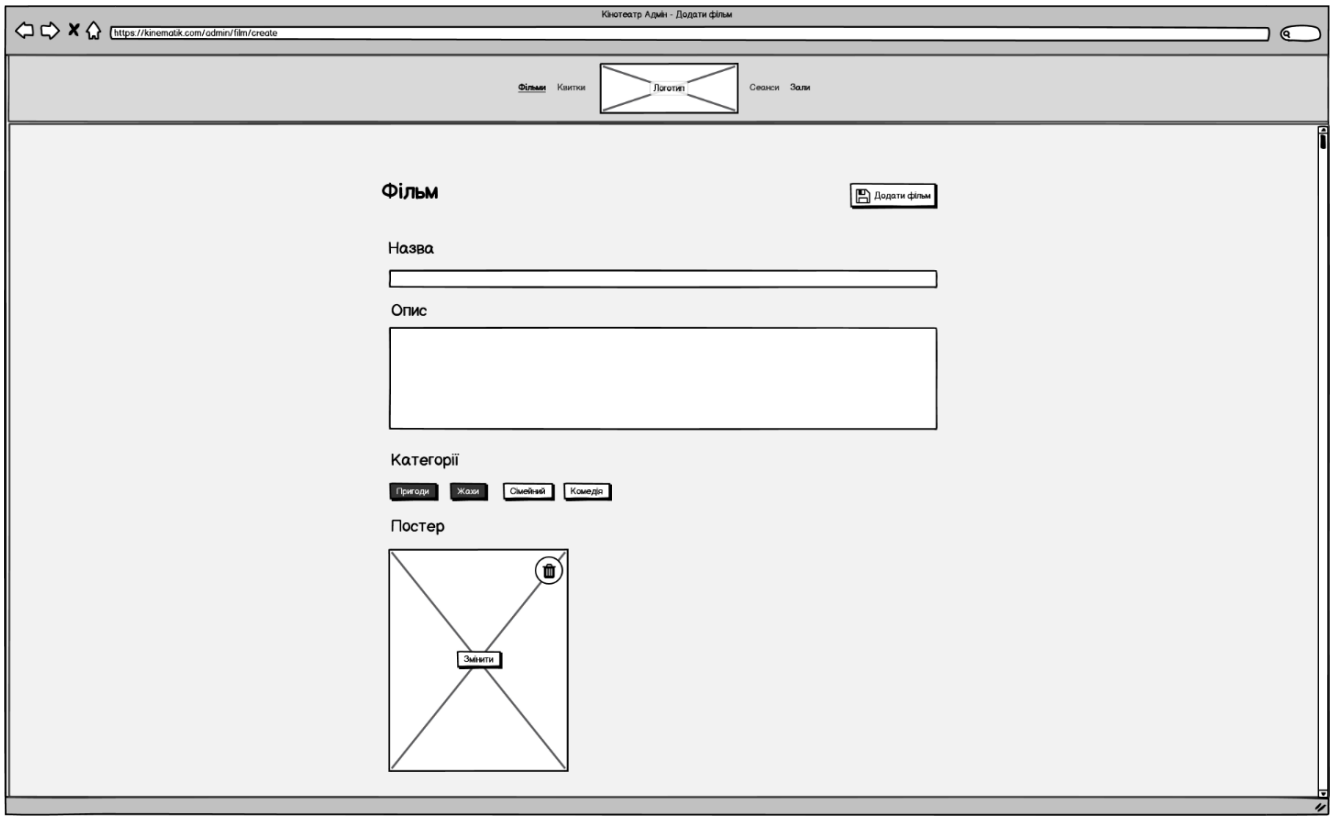


Рисунок 2.5 – Сторінка редагування даних про фільм

Зверху виводиться назва фільму, який редагується, а також кнопка для збереження змін. Знизу ж розташовані полядля редагування таких властивостей фільму: назви, короткого опису, категорій тощо. Для кожної характеристики використовуються різні типі полів: наприклад, категорії змінюються перемиканням стану кнопок-тегів, а опис фільму містить більше рядків ніж звичайне текстове поле. Задля зручнішої зміни постеру, застосунок виводить його прев’ю, змасштабувавши його до необхідного розміру.

2.3.5 Сторінка управління сеансами

На сторінці управління сеансами користувач може побачити всі

заплановані сеанси й маніпулювати ними: створювати нові, видаляти існуючі або ж переміщати їх у часі чи між залами. Макет цієї сторінки наведено на рисунку 2.6.

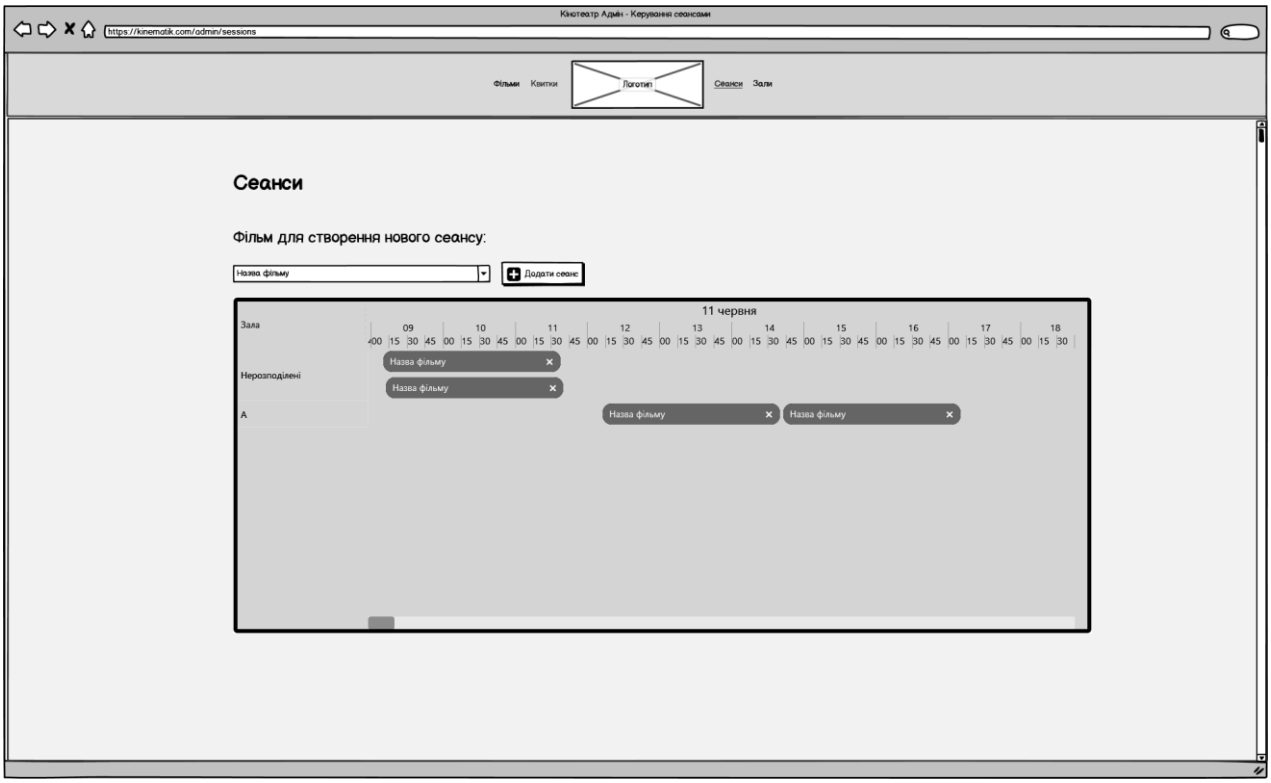


Рисунок 2.6 – Сторінка управління сеансами

Всі заплановані сеансі виводяться за допомогою діаграми Ґанта. Для створення нового сеансу використовується пара з комбінованого списку фільмів та кнопки, яка додає обраний фільм до діаграми. Елементи діаграми позначають в якій залі буде проводиться сеанс та коли він починається, а їх довжина автоматично визначається відовідно до тривалості фільму. Фігури можна переміщати в часі та між залами, але в одному залі не може проводитись одночасно кілька сеансів. Існує окремий рядок, на якій можна перемістити сеанси які мають проводитись, але для яких ще не визначено залу. Щоб видалити сеанс, користувач має клікнути на іконку хрестика в кінці елементу діаграми. Всі зміни автоматично зберігаються у базі даних.

2.3.6 Сторінка для перегляду стану місць по сеансам

На сторінці для перегляду стану місць по сеансам співробітник кінотеатру може переглянути статистику по купленим місць на кожен запланований сеанс. Прототип її дизайну можна побачити на рисунку 2.7.

Рисунок 2.7 – Сторінка перегляду стану місць по сеансам

На цій сторінці наведено всі публічні сеанси, дані про які перелічено у вигляді таблиці, в якій зазначено ідентифікатор сеансу, фільм, зала в якій буде проводитись сеанс, наскільки заповнена зала та час початку. При кліку на кнопку «Показати стан місць» знизу відповідного рядка виводиться схема зали та стан кожного з місць, а при наведенні вказівника миші на викуплене місце виводиться інформація про відповідне замовлення: його ідентифікатор, а також електронна пошта та телефон клієнта.

2.3.7 Інтерфейс мобільного застосунку для перевірки квитків

Для адміністраторів зали розроблено окремий мобільний застосунок. Після авторизації в системі, користувача перенаправляє на екран макет якого

представлено на рисунку 2.8.

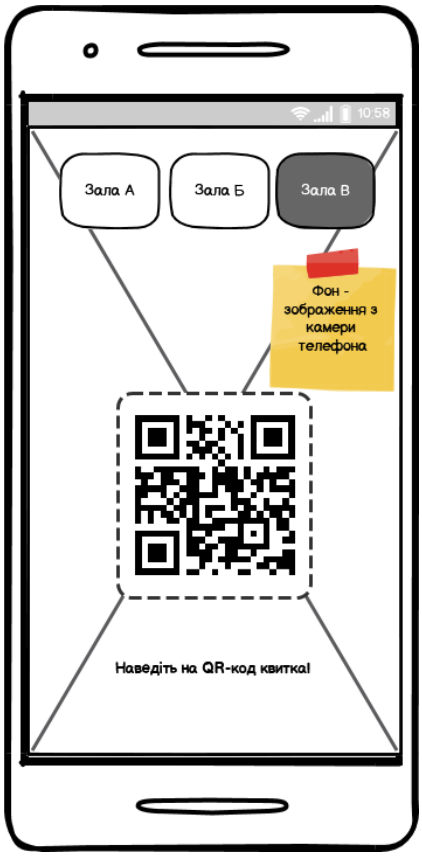


Рисунок 2.8 – Інтерфейс мобільного застосунку для перевірки квитків

На цьому екрані адміністратори перевіряють квитки клієнтів перед запуском їх до зали. Зображення з камери телефона виводиться на задній фон застосунку, а в його центрі розташована зона для сканування QR-коду, яка позначається прямокутником потрібної форми. Зверху знаходяться кнопки для зміни зали, для якої потрібно перевіряти квитки, щоб клієнти не переплутали зали.

2.4 Створення моделі бази даних

Наступним кроком моделювання системи є проєктування бази даних, під час якого створюється ER-модель. Вона описує кожну сутність у системі, її атрибути, обмеження (наприклад, на унікальність), відношення до інших сутностей тощо. Потім отриманий артефакт може бути перетворений в фізичну модель, яка буде підлаштована під обраний варіант СКБД.

Проаналізувавши прикладну сферу кінотеатру, зокрема її частину пов’язану з оцифровуємим функціоналом: кіноафішею та купівлюю квитків – побудовано ER-модель зображену на рис. 2.9.

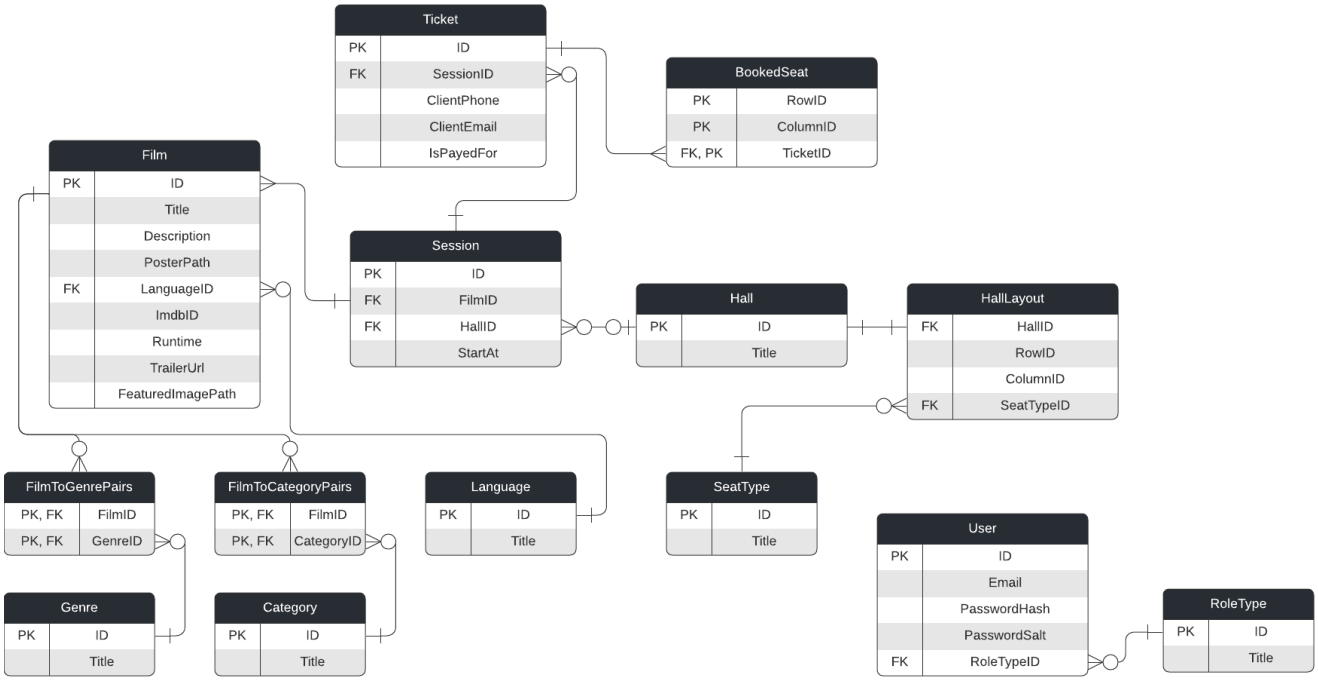


Рисунок 2.9 – ER-діаграма бази даних

Проаналізувавши наведену схему бази даних, можна побачити велику кількість відношень між сутностями, через що для розролюваного застосунку варто обрати реляційну СКБД, наприклад, Microsoft SQL Server.

Висновки до розділу 2

В другому розділі КРБ детально проаналізовано функціонал створюваної системи. Його представлено в діаграмі прецедентів, яка зображує основні прецеденти, їх акторів та яким чином вони пов’язані.

Створено прототипи інтерфейси основних екранів для всіх застосунків в комплексі за допомогою вайрфреймів.

Також в розділі спроєктовано модель бази даних, яка демонструє яким чином дані системи мають зберігатись, та представлено її у вигляді ER-діаграми, на основі якої аргументовано вибір реляційної СКБД Microsoft SQL Server.

3 АРХІТЕКТУРА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Склад комплексу застосунків

Проаналізувавши функціональні вимоги, вирішено створити комплекс застосунків, що складається з чотирьох частин:

а) backend API;

б) frontend інтерфейс для управління контентом;

в) мобільний застосунок для перевірки квитків;

г) сайт для клієнтів кінотеатру.

Схематично структуру комплексу наведено на рисунку 3.1.

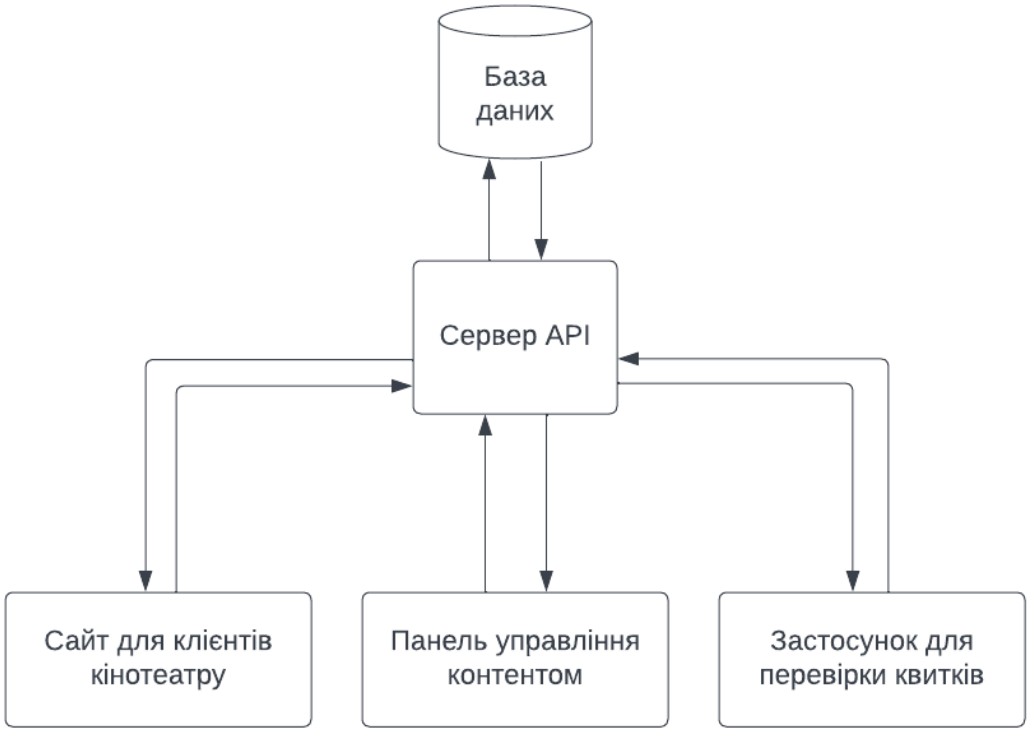


Рисунок 3.1 – Структура комплексу застосунків

Такий розподіл підвищує складність розробки системи, але також забезпечує такі переваги як:

– більша пристосованість ПЗ до потреб саме свого типу користувачів;

– можливість масштабування сервісів відповідно до їх навантаження: адміністраторів зал в кінотеатрі буде набагато менше ніж клієнтів, а тому на їх обслуговування можно виділити менше ресурсів;

– стійкість до помилок: клієнти кінотеатру зможуть користуватись сайтом, навіть якщо в частині для контент-менеджерів виникне фатальна помилка чи вона взагалі не буде вімкнена;

– можливість до впровадження змін в окремі частини комплексу: наприклад, розроблені зміни для клієнтського сайту не вимагатимуть перезавантаження серверу API, який зможе продовжити обслуговувати інші застосунки та не втратиме дані, що зберігаються в пам’яті як-от кеш.

3.2 Розгортання комплексу застосунків

Docker – це технологія, яка дозволяє розгортати ПЗ в ізольованих контейнерах, які інкапсулюють програми. За рахунок цього застосунки виконуються в стандартизованому середовищі, через що зникають помилки чи незручності пов’язані з різницею в операційних системах, версіях середи виконання тощо, оскільки вони однакові для всіх розробників, на серверах тестування та для вже впровадженої системи [14].

Є альтернатива цьому підходу – віртуалізація, але на відміну від віртуальних машин, Docker контейнери використовують не повну версію операційної системи, а лише один двигун контейнерів. Через це вони легші за розміром як на диску, так і у оперативній пам’яті. Принцип роботи віртуалізації зображено на рисунку 3.2, а контейнерізації – на рисунку 3.3.

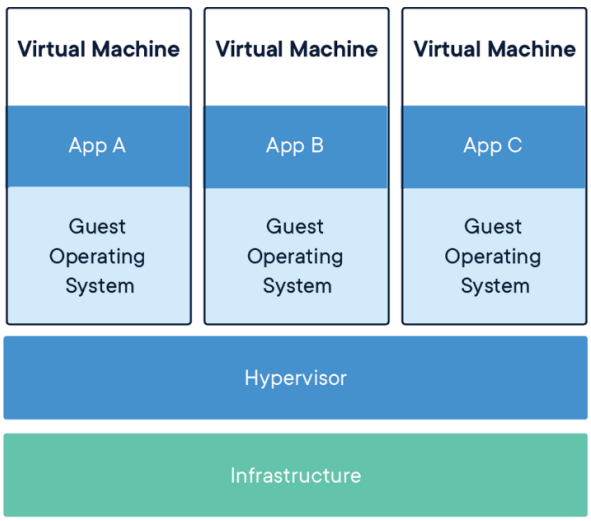


Рисунок 3.2 – Принцип роботи віртуалізації

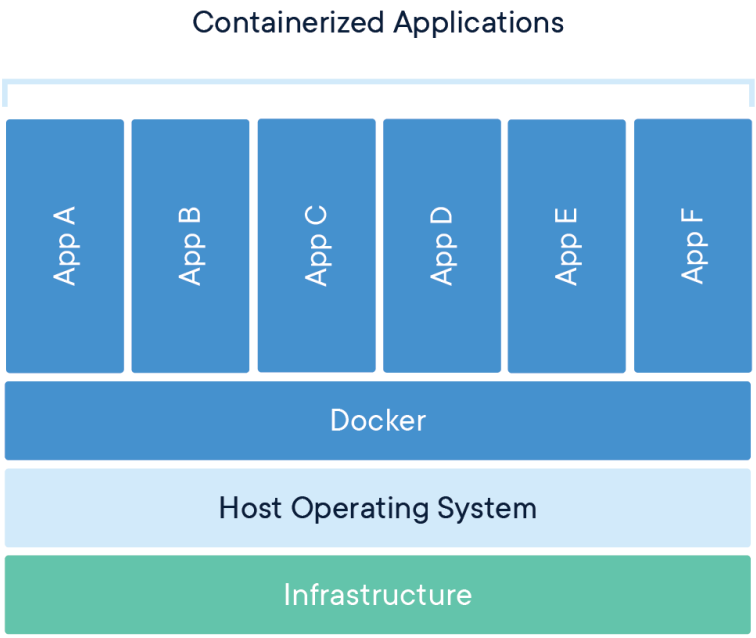


Рисунок 3.3 – Принцип роботи контейнерізації

Розробники вже створили величезну кількість контейнерів для різних застосунків від баз даних та середовищ виконання до проксі-серверів, брокерів повідомлень, кешу тощо. Більшість з них доступна в спеціальному публічному репозитарію, який називається Docker Hub. Будь-який користувач Docker може їх використовувати у своєму проєкті, лише завантаживши їх: немає ніякої потреби в безспосередньо встановлені їх на машину, на якій запускатиметься контейнерізований застосунок.

Разом з Docker часто використовується **Docker Compose** – інструмент для опису та запуску систем з кількох Docker-контейнерів. Його конфігурація дозволяє визначити налаштування кожного з контейнерів, що входять до складу комплексу, а також пов’язати їх окремою мережею. Кількома командами в консолі чи кліками в графічному інтерфейсі можна вмикати та вимикати цілі графи контейнерізованих застосунків.

3.2 Backend API

Backend API реалізує збереження та редагування даних, інтеграцію з платіжною системою, такі елементи сценаріїв викоистання як генерація QR-коду квитка та надсилання його на електрону поштову адресу тощо.

3.2.1 Чиста архітектура

Застосунок побудовано з використанням архітектурного стилю **«чиста (гексагональна) архітектура»** [13]. Він поєднав в собі декілька інших підходів, які сходяться у таких положеннях:

– Логіка застосунку має бути незалежною від UI частини, якою вона буде використовуватись: це може бути як графічний інтерфейс, так й консольний, чи навіть просто спілкування через HTTP тощо.

– Архітектура має бути незалежною від обраного інструментарію для інфраструктури: структура застосунку має передбачати зміну обраної бази даних, поштового серверу тощо.

– Має бути можливість тестувати будь-який компонент системи легко та з мінімальною кількістю об’єктів-заглушок.

Такі вимоги досягаються завдяки абстракції класів у інтерфейси, використання їх у якості контрактів та дотримання принципу інверсії залежності: шари мають бути незалежними від конкретних реалізацій, лише від контрактів. Завдяки цьому, розробник може дуже легко просто замінити одну реалізацію іншою чи створити нову.

Є декілька назв запропонованих шарів, але всі вони описують одні й ті ж концепти. Найбільш популярним розподілом є поділ на 4 частини:

– **Домен**. В ньому зберігаються моделі, які описують сутності предметної сфери застосунку. Також сюди можуть додаватись, наприклад, такі компоненти як Доменні Сервіси, якщо треба інкапсулювати якусь логіку обробки моделей для подальшого перевикористання в шарі Застосунку.

– **Застосунок**. Цей шар реалізує сценарії використання застосунку, використовуючи шари Домену та Інфраструктури. Він також містить Порти – контракти, через які спілкуються шари Користувацького Інтерфейсу й Інфраструктури з Застосунком, та використовуються цим шаром для реєстрації залежностей.

– **Користувацький Інтерфейс**. Він описує яким чином користувач, будь-то людина чи зовнішня система, може взаємодіяти з програмою: через графічний або консольний інтерфейс, за допомогою HTTP спілкування тощо.

Інфраструктура. Категорія Інфраструктури містить реалізації взаємодії з зовнішніми сервісами будь-то черга повідомлень, ORM чи поштовий сервер. Оскільки інтерфейси для використання конкретних реалізацій не завжди збігаються з тим, що зручно використовувати в контексті Застосунку, то часто їх абстрагують за допомогою шаблону проєктування «Адаптер». Він дозволяє поєднати Порт та інфраструктурний сервіс завдяки імплементації потрібного контракту функціями обраної бібліотеки чи фреймворку.

Схема роботи застосунку побудованого на основі гексагональної архітектури зображено на рисунку 3.4.

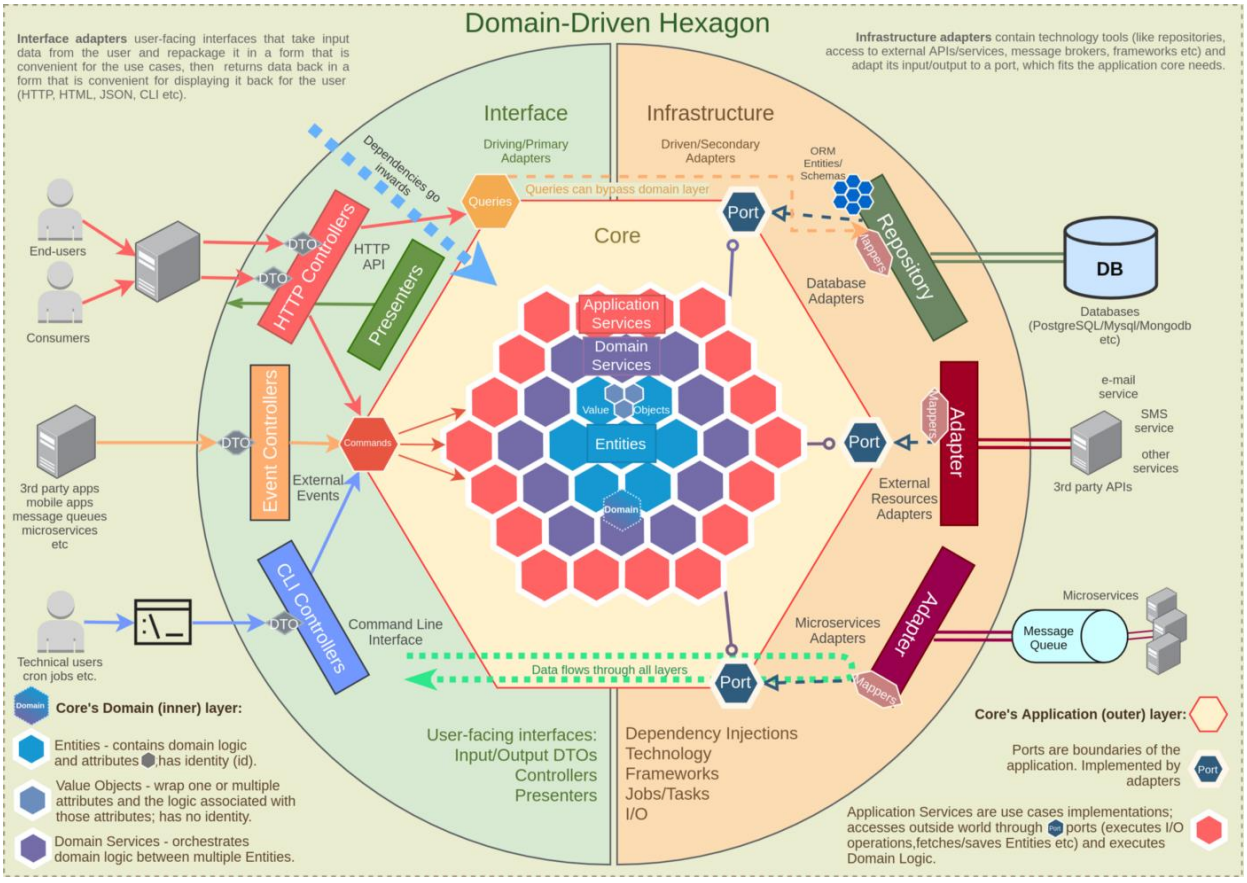


Рисунок 3.4 – Схема роботи застосунку побудованого на основі гексагональної архітектури

На наведеній схемі видно, що потік залежностей направлений всередину

застосунку, а тому моделі та сценарії використання, нічого не знають про оточуючі їх шари окрім їх контрактів. Через цю особливість ахітектури можна, наприклад, вільно змінювати користувацький інтерфейс та обрану базу даних. Також варто відмітити, що кожен шар відокремлений та містить в собі все потрібне для тестування свого функціоналу.

3.2.2 CQRS

У більшості інформаційних систем операції читання та операції запису мають різні вимоги:

– читання використовується часто й воно використовує агреговані дані, а не сутності домену;

– запис ж навпроти: використовується рідше, але має оперувати доменними сутностями, приймаючи до уваги їх обмеження тощо.

Для когнітивного спрощення кодування цих операцій та використання найбільш підходящих інструментів для кожної задачі можна використати **CQRS** – принцип побудови застосунків, у яких читання (query) та запис (command) оброблюються окремо.

Найбільша перевага цього патерну заключається саме в семантичному ропозділу, але також завдяки цьому розподілу можна пристосовуватись до потреб конкретного виду операцій, наприклад:

– використати різні технології отримання даних: для записів використовувати ORM, а для читання – чисті SQL запити;

– використати різні сховища даних: для записів використовувати реляційну базу даних та публікувати події при зміні даних, а для читання зберігати копії даних в NoSQL сховищі для кешування.

Приклад потоків даних при використанні цього підходу наведено на рисунку 3.5.

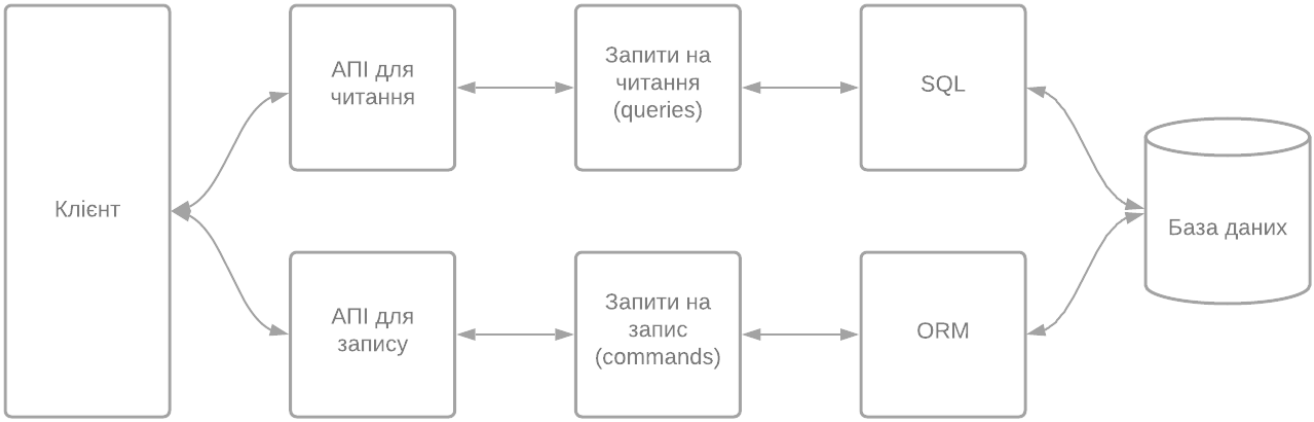


Рисунок 3.5 – Приклад потіку даних при використанні CQRS

В застосунку цей патерн реалізовано за допомогою іншого патерну – «Посередник». Його використовують для зменшення зв’язаності компонентів програми шляхом заміни їх на залежність від однієї сутності – Посередника, який собою поєднує ці об’єкти.

3.2.3 Вибір мови програмування та фреймворку

Реалізацію гексагональної архітектури та CQRS патерну зручно виконати за допомогою комбінації мови програмування **C#** та вебфреймворку **ASP.NET Core**. Обрана мова програмування статично типізована, що запобігає помилкам, які можна передбачити при компіляції. Вона постійно розвивається та щороку поповнюється новими можлиовстями, наприклад: елементи функціонального програмування, автоматичний вивід типів, анонімні типи, інтерполяція рядків тозо. Також C# має дуже обширну екосистему, яка містить безліч корисних інструментів: наприклад, бібліотеку **MediatR,** яка дозволяє зручно використовувати патерн «Посередник» у своїх застосунках.

Більшість вебзастосунків написаних на C# побудовані на ASP.NET, заради вбудованого контейнеру залежностей, HTTP контролерів з автоматичним маппінгом запитів, проміжних обробників HTTP запитів тощо.

3.2.4 ORM

Для виконання сценаріїв використання потрібно взаємодіяти з базою

даних: зберігати інформацію о фільмах, залах, сеансах, замовлення квитків тощо. Таку взаємодію можна реалізувати за допомогою прямих звертань до бази даних через SQL запити, але це дуже важко, оскільки для цього треба створювати велику кількість шаблонного коду, що неминуче призводить до помилок через неуважність.

Також при зміні бази даних треба переписувати більшість запитів, оскільки SQL діалекти між собою несумісні. Використання системи ORM вирішує всі ці проблеми, оскільки вона автоматично створює й оптимізує необхідні запити, а також синхронізує дані у пам’яті та базі даних. Ця поведінка реалізована через прошарок між базою даних та застосунком, який використовує цю технологію. Схема взаємодії з базою даних при використанні ORM наведена на рисунку 3.6.

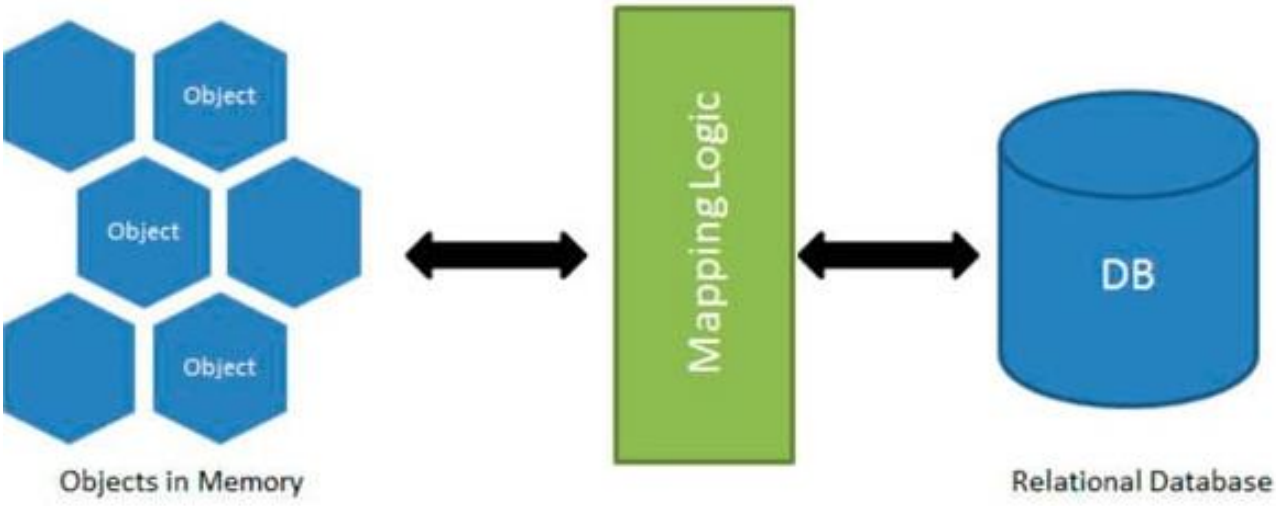


Рисунок 3.6 – Схема взаємодії з базою даних при використанні ORM

Разом з цими функціями можна використовувати ORM для маніпуляції з схемою бази даних: її можна генерувати та оновлювати з моделей сутностей за допомогою механізму міграцій. Саме цей підхід використовується в сервері API комплексу застосунків для автоматизації бізнес-процесів кінотеатру.

В екосистемі .NET головною реалізацією ORM є **Entity Framework**.

3.3 Інтерфейс для управління контентом

Контент-менеджери можуть управляти контентом в спеціальній панелі. В ній можна застосовувати стандартні CRUD операції для фільмів, зал, сеансів та переглядати стан заповненості сеансів.

3.3.1 Вибір мови програмування та бібліотек

Для розробки цієї частини комплексу застосунків використано мову програмування **TypeScript** – надбудову над JavaScript, яка дозволяє використовувати типи в коді. Статична типізація суттєво зменшує кількість помилок, а також покращує процес розробки підказками автодоповнення коду.

Застосунок побудовано на основі фреймворка **Vue** з використанням компонентної бібліотеки Bootstrap. Разом їх пов’язує бібліотека **BootstrapVue**, яка дозволяє розробникам декларативно працювати з компонентами. Обрано саме такі технології, оскільки Bootstrap задовільняє містить набір найвикористованіших елементів: кнопки, комбіновані списки, вспливаючі підказки тощо. Також ця бібліотека містить утилітарні класи, за допомогою яких зручно вносити маленькі зміни в стилі елементів сторінки без звичайної каскадності CSS.

3.3.2 SPA

Інтерфейс для управління контентом побудовано згідно з концепцією **SPA** [7]: сторінка завантажується лише один раз, а потім при переході на інші сторінки застосунку звертається до серверу API за потрібними даними та відповідно змінює свою структуру. Таким чином, статичні ресурси як-от: таблиці стилів, скрипти, файли шрифтів тощо – завантажуються лише один раз, що пришвидшує роботу застосунку та зменшує навантаження на сервер. Також за допомогою цього підходу застосунок краще розділяє логіку між backend та frontend, оскільки розмітка не генерується на бекенді. Найбільш популярним та функціональним засобом реалізації SPA для Vue є **Vue Router**.

3.3.3 Діаграма Ґанта

Інтерфейс для управління розкладом сеансів використовує діаграму Ґанта для зручності представлення сеансів в часі та призначення їм зали. В неї є кілька рядків та шкала часу, а елементи діаграми зображуються прямокутниками з довжиною пропорціональною тривалості елементу. Для розкладу сеансів рядками є зали. Приклад цього виду діаграм наведено на рис. 3.7.

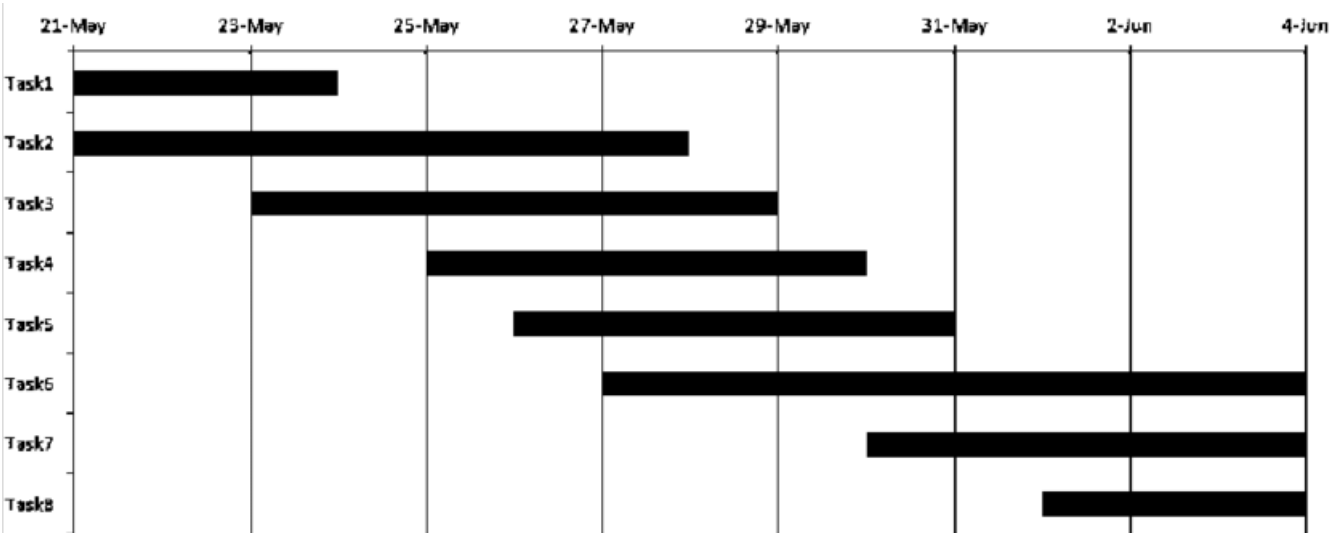


Рисунок 3.7 – Приклад діаграми Ґанта

З метою інтеграції цієї діаграми в застосунок та прив’язки його до Vue використано бібліотеку **gantt-schedule-timeline-calendar**.

3.4 Мобільний застосунок для перевірки квитків

Задля валідації квитків користувачів, адміністратори зал використовують мобільний застосунок. Після авторизації їм пропонується обрати залу для якої вони перевірятимуть квитки, а потім при скануванні QR-коду квитка система виведе повідомлення з результатом перевірки.

Мобільний застосунок побудовано на технології **React Native**, оскільки ця технологія забезпечує кросс-платформеність, за винятком унікальних можливостей мобільних платформ, які не потрібні для реалізації функціоналу

перевірки квітки. Ця технологія перетворює React компоненти на компоненти платформи, на якій запускається застосунок. Більш детально цей процес описано на рисунку 3.8.

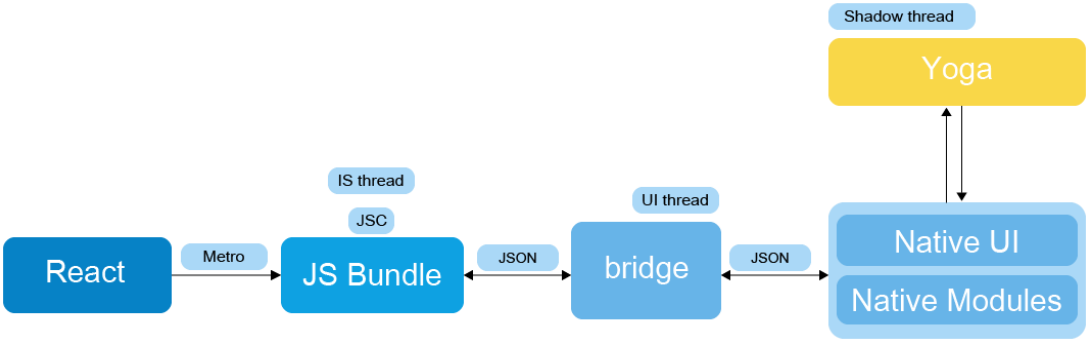


Рисунок 3.8 – Процес роботи React Native

За описаною вище логікою в якості мови програмування обрано **TypeScript**.

3.5 Сайт для клієнтів кінотеатру

Обслуговуванням потреб клієнтів користувачу займається окремий сайт. За допомогою нього можно передивитися кіноафішу, деталі про фільм, купити квиток тощо.

Оскільки клієнтський сайт має більшу потребу в красоті дизайну та кастомізації, то в якості фреймворку обрано **React**. Окрім зазначених вище переваг, використання React для сайту клієнтів користувачів та React Native для мобільного застосунку для перевірки квитків дозволяє консолідувати спільну логіку, стилі, компоненти тощо.

За такими ж резонами ж як і в інтерфейсі для управління контентом, в застосунку використовується мова програмування **TypeScript** та методологія **SPA,** яка реалізується за допомогою бібліотеки **React Router**.

Так як цей сайт потребує більш глибокої стилізації, замість утилітарних класів використовується звичайний CSS, неконтрольована каскадність якого нівелюється технологією CSS Modules. Вона додає класам кожного компоненту додатковий суфікс: спеціальний хеш. Приклад розмітки з

згенерованими класами наведено на рисунку 3.9.

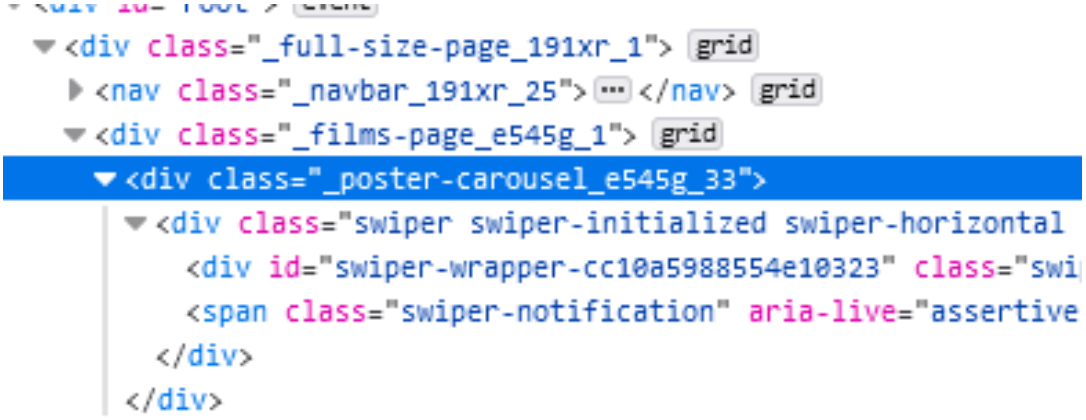


Рисунок 3.9 – Приклад згенерованої розмітки при використанні CSS Modules

За замовчуванням при створенні хешу-суфіксу використовується шлях до оброблюваємого файлу, завдяки чому гарантується унікальність створюваній назві класу.

Висновки до розділу 3

В третьому розділі КРБ детально описано архітектуру створюваного комплексу застосунків: вказано його складові, яким чином вони взаємодіють між собою та метод його розгортання.

Відповідно до функціональних вимог обрано мову програмування та допоміжні технології, бібліотеки, підходи до розробки тощо для кожного з компонентів системи. Також розглянуто такі концепції з інженерії ПЗ: гексагональна архітектура, CQRS, ORM та SPA.

4 КОДУВАННЯ ТА ДОКУМЕНТАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Серверне API

4.1.1 Налаштування застосунку

Точка входу в застосунок серверного API налаштовує вебфреймворк ASP.NET Core таким чином:

а) описується порядок виконання дій та правила обробки запитів:

1) авторизація;

2) роутінг;

3) Cross-Origin Resource Sharing;

б) вказуються джерела статичних файлів;

в) підключаються та конфігуруються сторонні сервіси:

1) для ORM EntityFramework вказується рядок підключення до бази даних;

2) для бібліотеки спілкування в режимі реального часу SignalR налагоджуються центри обробки запитів та адреси для взаємодії з ними;

3) для бібліотеки опису HTTP API Swagger наводяться опції щодо генерації документації;

4) для сервісу роботи з електронною поштою вказується бажана адреса відправника, його ім’я та авторизаційний ключ;

г) додаються необхідні елементи до вбудованого контейнеру ін’єкції залежностей.

4.1.2 CQRS та шаблон проєктування «Посередник»

В створюваному проєкті використовується спрощена версія патерну CQRS [6] без розподілу на різні типи баз даних. Вона реалізована за допомогою бібліотеки MediatR, яка інкапсулює в собі шаблон проєктування «Посередник»,

необхнідні абстракції та методи для реєстрації екземплярів цих абстракцій в контейнері ін'єкції залежностей.

Окрім команд та запитів MediatR надає змогу використовувати механізм підписки на повідомлення, за допомогою якого, наприклад, реалізовано надсилання електронного листа при оплаті квитка та автоматичного оновлення статусу місця на клієнтському сайті при створенні замовлення.

Стандартну структуру запитів, команд та обробників повідомлень на прикладі запиту про детальну інформацію про певну залу наведено на наступному лістингу:

```
namespace Kinematik.Application.Queries.Admin.Halls
{
    public class GetHallQueryInput : IRequest<GetHallQueryOutput>
    {
        public int HallID { get; set; }
    }

    public class GetHallQueryHandler : IRequestHandler<GetHallQueryInput,
GetHallQueryOutput>
    {
        private readonly KinematikDbContext _dbContext;

        public GetHallQueryHandler(
            KinematikDbContext dbContext
        )
        {
            _dbContext = dbContext;
        }

        public async Task<GetHallQueryOutput> Handle(GetHallQueryInput request,
CancellationToken cancellationToken)
        {
            GetHallQueryOutput output = new GetHallQueryOutput();

            var hall = await _dbContext.Halls
                .Where(hall => hall.ID == request.HallID)
                .Select(hall =>
                    new
                    {
                        hall.Title,
                        HallLayoutItems = hall.LayoutItems.Select(hallLayoutItem
=> new GetHallQueryOutput.MappedHallLayoutItem
                        {
                            RowID = hallLayoutItem.RowID,
                            ColumnID = hallLayoutItem.ColumnID,
                            SeatType = hallLayoutItem.SeatType,
                        })
                    })
                .SingleOrDefaultAsync(cancellationToken);

            output.Title = hall.Title;
            output.LayoutItems = hall.HallLayoutItems;

            return output;
        }
```

```
    }

    public class GetHallQueryOutput
    {
        public string Title { get; set; }
        public IEnumerable<MappedHallLayoutItem> LayoutItems { get; set; }

        public class MappedHallLayoutItem
        {
            public int RowID { get; set; }
            public int ColumnID { get; set; }
            public SeatType SeatType { get; set; }
        }
    }
}
```

Всі класи в цих сутностях угруповано в одному файлі за принципом вертикального розрізу (англ. vertical slicing) [16]: класи поєднуться за причетністю до певного функціоналу, замість розподілу по типу класу.

4.1.3 HTTP API

HTTP API застосунку також структуровано за принципом вертикального розрізу.

Оскільки в застосунку використано шаблон проєктування «Посередник», то самі методи контроллерів створюються дуже лаконічними: вони лише перетворюють запит в сприятливий формат, відправляють за допомогою посереднику потрібну команду або запит та перетворюють результат у відповідь.

4.1.4 WebSockets

Однією з критичних переваг розроблюваного застосунку є оновлення даних о статусі місць на певний сеанс у режимі реального часу. Це досягається за допомогою використання протоколу WS [15], робота з яким в екосистемі .NET полегшується бібліотекою SignalR [1]. Вона спрощує управління підключеннями, організацію цих підключень в групи, збереження інформації про контекст поточної операції тощо.

Аналогічна контроллерам концепція в SignalR називається хабом [9]. Хаб також надає можливість викликати процедури на сервері клієнтом, але на

відміну від контроллерів, також можна й сервером ініціювати запуск процедури на клієнті. Варто також відмітити, що самого поняття відповіді в хабах немає, але можливо, наприклад, наприкінці методу викликати на клієнті функцію отримання даних.

Хаб для оновлення стану бронювання місць наведено на наступному лістингу:

```
namespace Kinematik.RealTimeApi.Bookings
{
    public class BookingHub : Hub
    {
        private readonly IMediator _mediator;

        public BookingHub(IMediator mediator)
        {
            _mediator = mediator;
        }

        public async Task SubscribeToBookingStatuses(int sessionID)
        {
            GetBookingStatusesResponse response = new
GetBookingStatusesResponse();

            GetBookingStatusesQueryOutput queryOutput = await _mediator.Send(
                new GetBookingStatusesQueryInput
                {
                    SessionID = sessionID
                }
            );

            response.BookingStatuses =
queryOutput.BookingStatuses.Select(bookingStatus => new
GetBookingStatusesResponseBookingStatus
            {
                RowID = bookingStatus.RowID,
                ColumnID = bookingStatus.ColumnID,
                SeatTypeID = (int)bookingStatus.SeatType,
                IsFree = bookingStatus.IsFree
            });

            await this.Groups.AddToGroupAsync(this.Context.ConnectionId,
$"subscriber_to_bookings_{sessionID}");
            await this.Clients.Caller.SendAsync("GetBookingStatuses", response);
        }
    }
}
```

З цього уривку можна побачити, що вони майже нічим не відрізяються від контроллерів за виключенням використання додаткових методів для додавання користувача у групу та дещо екзотичного варіанту повернення відповіді.

4.1.5 Платіжний шлюз

В застосунку для оплати квитків реалізовано інтеграцію з платіжною системою LiqPay [12]. За допомогою її документації реалізовано процес оплати шляхом перенаправлення користувача на персоналізовану платіжну сторінку [11]. Його етапи детально проілюстровано на рисунку 4.1.

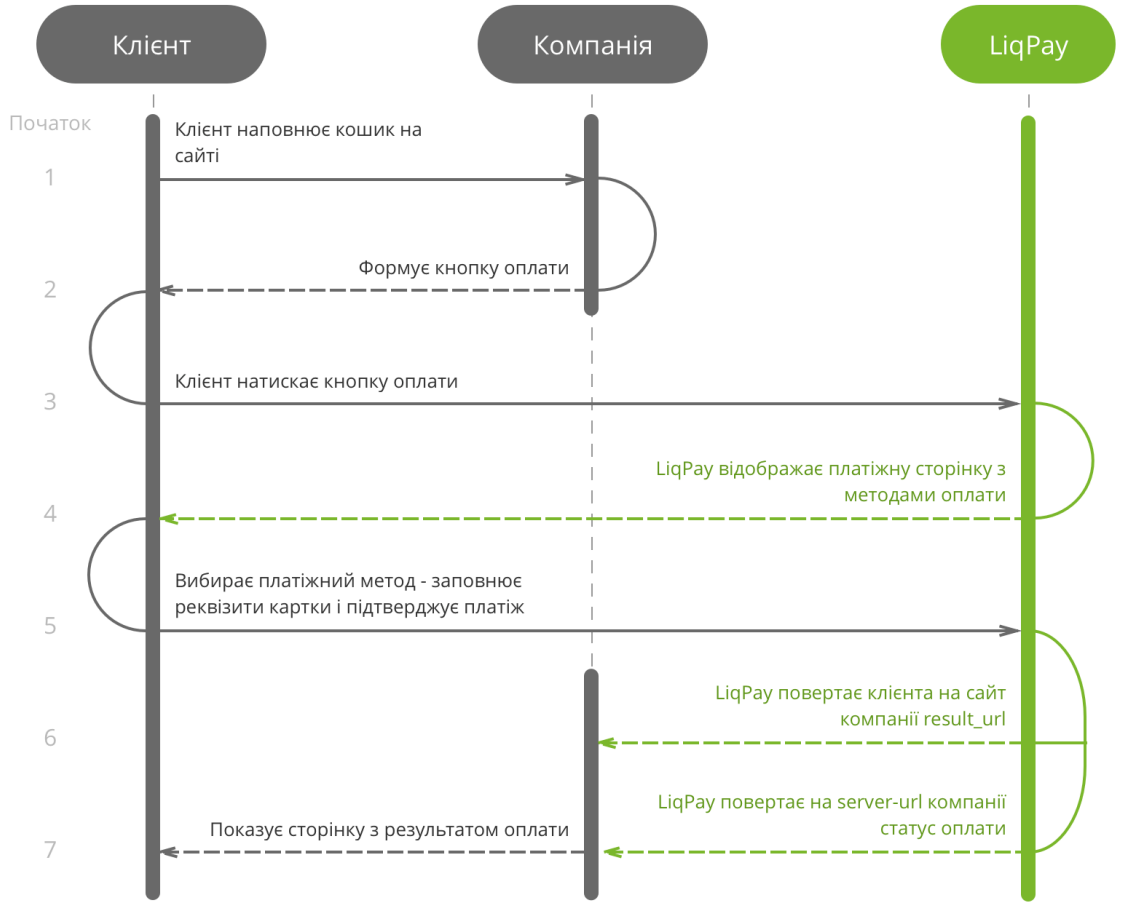


Рисунок 4.1 – Етапи оплати квитка через LiqPay

При кліку на кнопку купівлі квитка запис про клієнтське замовлення зберігається у базі даних, а разом з ним формується форма з необхідними параметрами, які генеруються відповідно до документації LiqPay: створюється об’єкт запиту, серіалізується у вигляді JSON, а потім отримані рядки разом з приватним ключом кодуються у base64 та/або шифруються за допомогою шифру SHA-1.

Потім ця форма відправляється та перенаправляє користувача на сторінку платіжного шлюзу, вигляд якої наведено на рисунку 4.2.

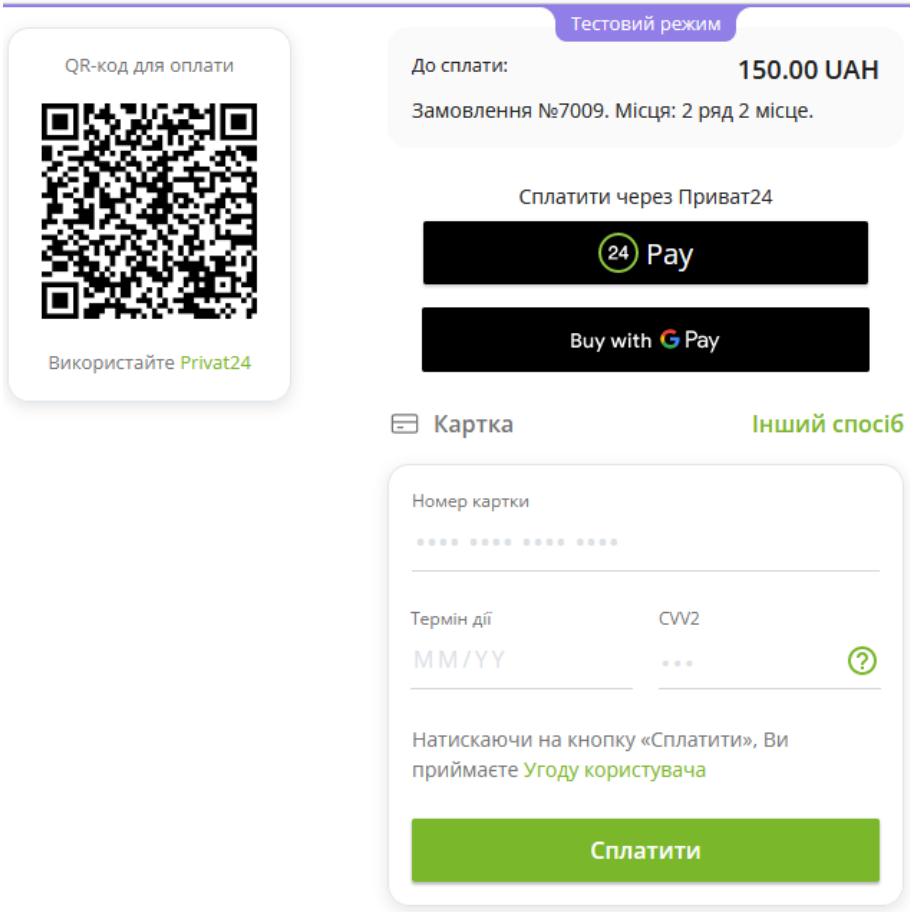


Рисунок 4.2 – Форма оплати квитка

В результаті цього потоку даних, система оплати звертається до API кінотеатру по вказаному колбеку та передає дані щодо транзакції та її статусу. Якщо оплата пройшла успішно, то в базі даних це замовлення помічується як оплачене, а клієнту кінотеатру надсилається відповідного електронного листа.

4.1.6 Відправка квитка електронним листом

Для авторизації клієнта кінотеатру при заході в залу сеанса, він має спершу показати відповідний QR-код адміністратору зали. Його він отримує після оплати квитка у електроному листі разом з інформацією про транзакцію, обрані місця, номером та часом замовлення тощо. Це реалізовано за допомогою обробника повідомлень MediatR, який у відповідь на подію оплати квитка знаходить замовлення, обрані місця, електрону пошту користувача тощо і трансформує їх у розмітку листа за допомогою двигуну шаблонів Razor.

Приклад створеної розмітки показано на рисунку 4.3.

Рисунок 4.3 – Вміст електронного листа про придбання квитка

Сам лист відправляється за допомогою сервісу Mailgun [17], а процес роботи з його розміткою та метаданими лаконічно абстраговано бібліотекою FluentEmail.

4.2 Клієнтський сайт

Клієнтський сайт побудовано на найкращих принципах створення сучасних frontend застосунків: SPA, фреймворк для декларативного оновлення розмітки тощо.

Особлива увага при розробці цього застосунку приділялась його зовнішньому вигляду, через що використано користувацькі стилі, які

оброблюються технологією CSS Modules. Також, завдяки цьому підходу дуже легко контролювати адаптивність дизайну, оскільки можна легко маніпулювати макетом сторінки за допомогою CSS Grid [20], яка дозволяє змінювати розміри колонок та стовпчиків сітки макету та навіть змінювати розташування елементів сторінки в цій сітці. Аналогічним чином, але для більш локальних випадків, використовується CSS Flexbox. Приклад такої адаптації наведено на рисунку 4.4.

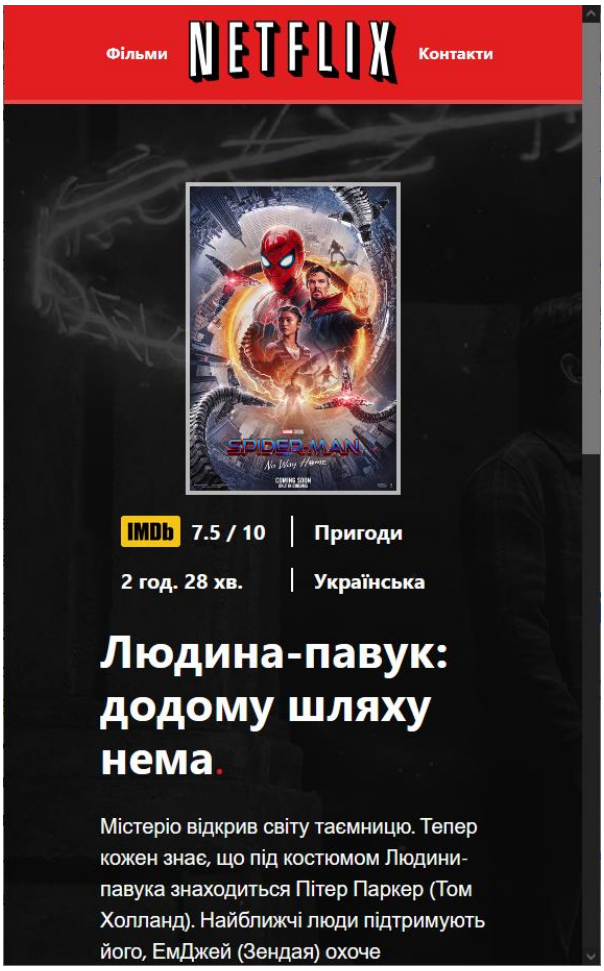


Рисунок 4.4 – Приклад адаптації сторінки до малого розміру екрану

При виборі певного сеансу на сторінці купівлі квитків застосунок підписується на оновлення стану місць для в режимі реального часу. Це виконується за допомогою клієнтською частини білбліотеки SignalR.

```
useEffect(async () => {
  if (selectedTime !== null && selectedTime !== undefined) {
    (async () => {
```

```
      const connection = new signalR.HubConnectionBuilder()
        .withUrl(`${window.apiBaseUrl}/real-time/bookings`)
        .build();
      connection.on("GetBookingStatuses", (response) => {
        const bookingStatuses = response.bookingStatuses.map(
          (rawLayoutItem) => {
            return {
              rowID: rawLayoutItem.rowID,
              columnID: rawLayoutItem.columnID,
              seatType: rawLayoutItem.seatTypeID as SeatType,
              seatStatus: rawLayoutItem.isFree
                ? SeatStatus.FREE
                : SeatStatus.BOOKED,
            };
          }
        );

        const newLayoutRows = splitBookingStatusesIntoLayoutRows(bookingStatuses);
        setLayoutRows(newLayoutRows);
      });

      await connection.start();

      await connection.send(
        "SubscribeToBookingStatuses",
        selectedTime.sessionID
      );
    })();
  }
}, [selectedTime]);
```

Таким чином, після забронювання місця одним клієнтом, це одночасно відобразиться на екрані всіх інших клієнтів.

4.3 Панель керування контентом

За вищезгаданими принципами побудовано й панель керування контентом, але на відміну від клієнтського сайту для неї швидкість змін та консистентість важливіші за зовнішній вигляд, через що для реалізації обрано зв’язку Vue + BootstrapVue.

Особливої уваги заслуговує сторінка для управління сеансами, оскільки вона містить віджет діаграми Ґанта. Інтерфейс цього віджету наведено на рисунку 4.5.

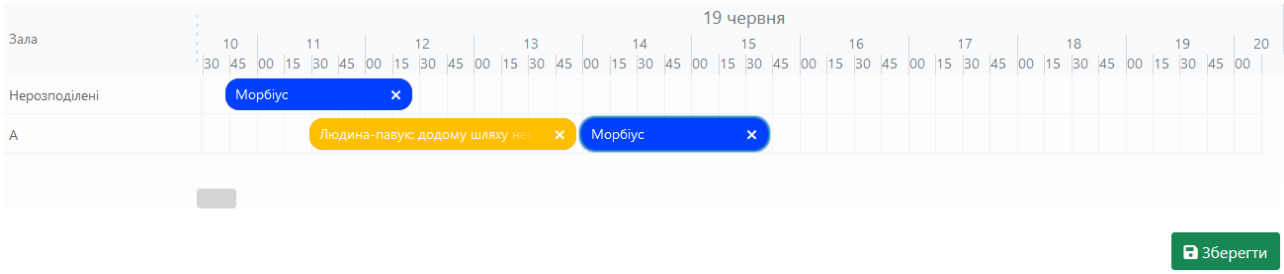


Рисунок 4.5 – Інтерфейс віджету для управління сеансами

При завантаженні сторінки робиться низка запитів до серверу API, один з яких отримує дані про розклад сеансів, після чого ця інформація споживається компонентом діаграми Ґанта.

Для уникнення призначення неправильного часу шляхом переміщення сеансу по діаграмі, використовується функція-колбек, яка перевіряє коректність нової дати початку сеансу.

```
onMove({ items }) {
  return items.before.map((beforeMovementItem, index) => {
    const afterMovementItem = items.after[index];
    const myItem: Item = GSTC.api.merge({}, afterMovementItem);

    const shouldPreventCollisionAfterMoving =
      !isItemUnassigned(afterMovementItem) && isCollision(myItem);
    const shouldPreventMovingOutOfBounds =
      dayjs(myItem.time.start).isBefore(periodStart);
    const shouldPrevent =
      shouldPreventCollisionAfterMoving || shouldPreventMovingOutOfBounds;

    if (shouldPrevent) {
      myItem.time = { ...beforeMovementItem.time };
      myItem.rowId = beforeMovementItem.rowId;
    }

    return myItem;
  });
}
```

В наведеному коді використовується бібліотека dayjs, яка значно розширює можливості TypeScript щодо роботи з часом та датами.

4.4 Застосунок для перевірки квитків

Застосунок для перевірки квитків сконструйовано з існуючих

компонентів в фреймворку React Native та його розширенню – Expo[8]. Це розширення надає більшу кількість готових компонентів, у тому числі сканнер для QR-коду, а також робить процес розброки набагато комфортнішим.

При старті застосунку робиться запит на API сервер задля отримання переліку зал які потім виводяться у вигляді кнопок поверх компоненту для сканування.

```
export async function getAvailableHalls() {
  const response = await axios.get("/api/halls");
  const parsedResponse: AvailableHallsResponse = response.data;

  return parsedResponse.halls.map((availableHall) => {
    return {
      id: availableHall.id,
      title: availableHall.title,
    } as Hall;
  });
}
```

Після вибору зали, адміністратор отримує змогу перевіряти продемонстровані квитки. При скануванні QR-коду надсилається запит для перевірки коректності квитка, а результат перевірки виводиться на екран у вигляді вспливаючого повідомлення (рис. 4.6).

```
handleQrCodeScanned = async (scanningResult: BarCodeScanningResult) => {
  if (this.state.checkedHallID !== null) {
    this.setState({
      shouldScan: false,
    });

    await this.checkTicket(Number.parseInt(scanningResult.data));

    this.setState({
      shouldScan: true,
    });
  }
};
```

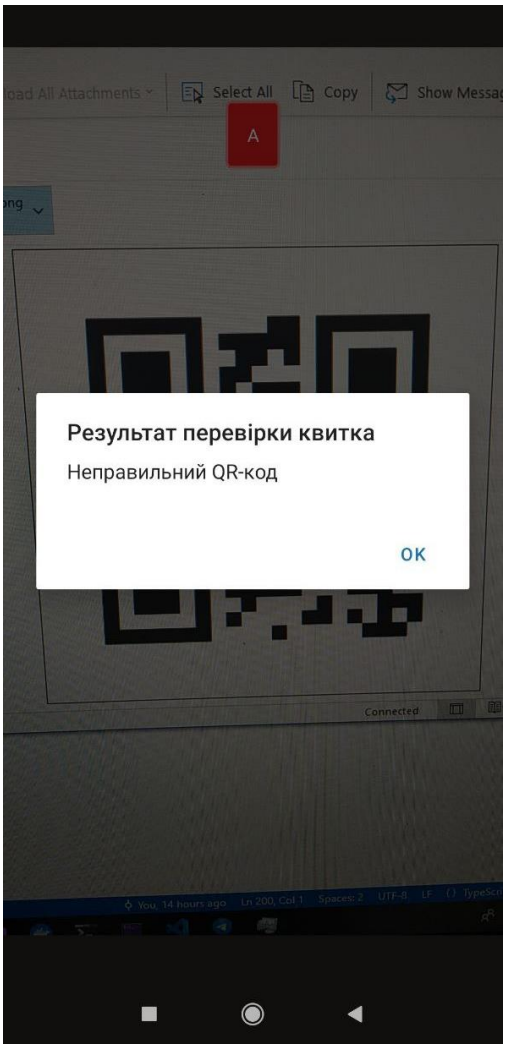


Рисунок 4.6 – Результат перевірки квитка

Окрім цієї помилки можуть бути ще такі результати:

– неоплачене замовлення;

– неправильна зала;

– неправильний сеанс.

4.5 Документація

Для документації створеного HTTP API використано набір розробницьких інтерфейсів Swagger [13], який автоматично генерує специфікацію API та спеціальну сторінку SwaggerUI, інтерфейс якой зображено на рисунку 4.7.

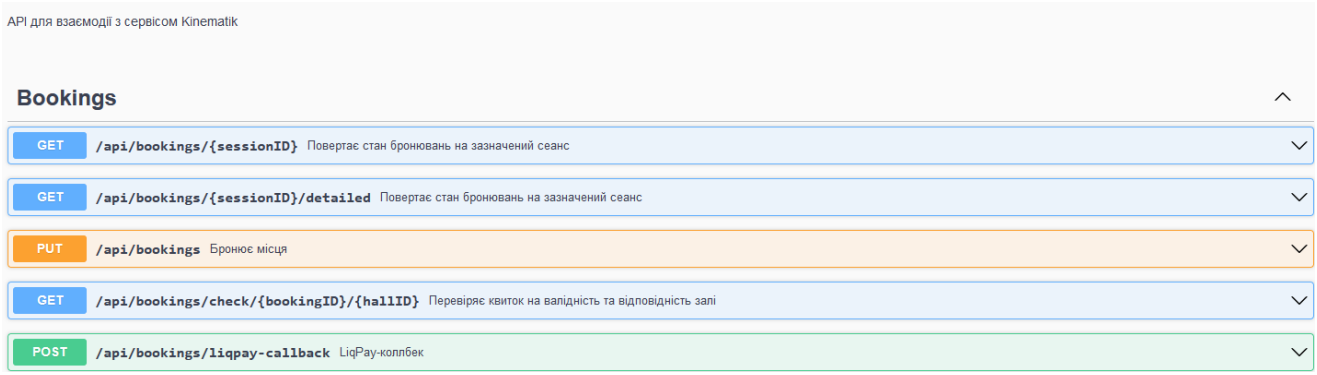


Рисунок 4.7 – Інтерфейс SwaggerUI

Тут розробник може швидко переглянути всі описані операції разом з їх методами, шляхами, схемами запитів/відповідей тощо, а також зручно протестувати API прямо в браузері.

Також в проєктах з розвинутою архітектуру варто документувати, наприклад, за допомогою документів Architecture Desion Record (ARD) [4]. Приклад такого документу наведено на рисунку 4.8.

Architecture Decision Record: Використання спілкування в режимі реального часу

Контекст

В системі автоматизації бізнес процесів кінотеатру Kinematik клієнти можуть створювати замовлення на купівлю квитків. Варто відзначити, що ці замовлення можуть бути скасовані. Бажано, щоб користувач відразу бачив зміну стану місця, щоб мати змогу обрати місце, квиток на яке було щойно скасовано, та запобіти конфліктній ситуації, в якій кілька клієнтів замовлюють квиток на одне й теж місце.

Рішення

Вирішено використовувати такі технології спілкування в режимі вільного часу (в порядку пріорітетності)

1. **WebSockets**;
2. *Long Pooling*;
3. *Server-Side Events*.

Для абстракції вирішено використати бібліотеку **SignalR**.

Статус

Реалізовано

Наслідки

1. В застосунок backend API додано бібліотеку SignalR.
2. Створено хаб(и), які реалізують API для оновлення даних в режимі реального часу.

Рисунок 4.8 – Приклад ARD

Зазвичай ці документи формують у вигляді Markdown текстів, які могуть складаться з таких частин:

– назва;

– контекст проблеми, яке описуване рішення має подолати;

– опис самого рішення;

– статус;

– наслідки.

За допомогою цього підходу будь-який розробник може дізнатися більше про архітектуру та не витрачати ресурси на дослідження рішення, яке попередньо вже було відмовлено з певних причин.

Висновки до розділу 4

В четвертому розділі КРБ описано найбільш визначні особливості кодування комплексу застосунків. На прикладі створеного backend API розглянуто варіанти реалізації гексагонального архітектурного стилю, патерну CQRS, а також особливості структурування застосунку: глобальний розподіл застсоунку на команди, запита та обробники подій, локальний розподіл за вертикальним принципом тощо. Докладно пояснено нюанси реалізації таких механізмів:

– синхронізації стану місць в режимі реального часу;

– інтеграції з платіжним шлюзом LiqPay;

– відправки електронних листів з QR-кодом квитка.

Також наведено методолію розробки адаптивних стилів для клієнтського сайту з використанням CSS Grid та CSS Flexbox.

Проведено процес документації розробленого застосунку за допомогою API специфікації – Swagger та спеціальної панелі для її перегляду або тестування – SwaggerUI. Також розглянуто важливість використання записів про архітекурні рішення.

ВИСНОВКИ

В процесі виконання кваліфікаційної роботи бакалавра автоматизовано бізнес-процеси кінотеатру шляхом розробки комплексу застосунків.

Для досягнення цієї мети вирішено наступні завдання:

– аналіз предметної області та існуючих аналогів;

– специфікація вимог для створюваного ПЗ на основі проведеного аналізу;

– проєктування комплексу взаємопов’язаних сервісів відповідно до вимог;

– моделювання користувацьких інтерфейсів та API;

– розробка backend сервісу для редагування та зберігання даних інформаційної системи;

– створення адміністративної панелі для редагування контенту контент-менеджером;

– розробка frontend застосунку для взаємодії клієнтів з кінотеатром;

– розробка мобільного застосунку для перевірки квитків адміністраторами зали кінотеатру;

– документація створених артефактів.

Систему розроблено за інженерним методом: перед кодуванням розроблено оптимальну архітектуру, змодельовано сутності прикладної сфери кінотеатру та їх відношення, а також створено прототипи користувацького інтерфейсу для кожного компоненту з комплексу застосунків.

Розглянуто такі концепції інженерії програмного забезпечення: гексагональна архітектура, CQRS, SPA, ORM, вертикальний принцип розподілу, шаблон проєктування «Посередник» тощо. Відповідно до специфікації вимог та обраної архітектури, визначено технології, фреймворки та бібліотеки для реалізації проєкту та наведено аргументацію щодо прийнятих рішень.

Результатом проведеної роботи є комплекс застосунків, який складається з чотирьох частин: backend API, сайт для клієнтів кінотеатру, адміністративна

панель для редагування контенту та мобільний застосунок для перевірки квитків.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Aguilar J. M. SignalR Programming in Microsoft ASP. NET. Microsoft Press, 2014. С. 17–18.

2. API for IMDb, TMDb, Wikipedia and more - IMDb API. URL: https://imdb-api.com/ (дата звернення 29.05.2022).

3. Balsamiq Cloud. URL: https://balsamiq.cloud/ (дата звернення 11.06.2022).

4. Clements P., Garlan D., Little R. et al. Documenting software architectures: views and beyond. *25th International Conference on Software Engineering, 2003. Proceedings.*(2003). IEEE, 2003. ISBN 076951877X. С. 740–741.

5. Cockburn A. Writing effective use cases. Pearson Education India, 2001. ISBN 8177589075.

6. Debski A., Szczepanik B., Malawski M. et al. A scalable, reactive architecture for cloud applications. *IEEE Software*. Vol. 35, Issue 2. С. 62–71.

7. Gavrilă V., Băjenaru L., Dobre C. Modern single page application architecture: a case study. *Studies in Informatics and Control*. Vol. 28, Issue 2. С. 231–238.

8. GitHub - expo/expo: An open-source platform for making universal native apps with React. Expo runs on Android, iOS, and the web. URL: https://github.com/expo/expo (дата звернення 10.06.2022).

9. Ingebrigtsen E. SignalR–Real-time Application Development. Packt Publishing Ltd, 2015. С. 37–49. ISBN 1785288628.

10. Jacobson I., Spence I., Kerr B. Use-case 2.0. *Communications of the ACM*. Vol. 59, Issue 5. С. 27–28.

11. LiqPay - Документація API інтернет-еквайрингу. URL: https://www.liqpay.ua/documentation/api/aquiring/ (дата звернення 05.06.2022).

12. LiqPay - Єдине платіжне рішення для вашого бізнесу. URL: https://www.liqpay.ua/ (дата звернення 05.06.2022).

13. Martin R. C., Grenning J., Brown S. et al. Clean architecture: a craftsman’s guide to software structure and design. Prentice Hall, 2018. 636 с.

14. Miell I., Sayers A. Docker in practice. Simon and Schuster, 2019. 8 с. ISBN 1638356300.

15. Murley P., Ma Z., Mason J. et al. Websocket adoption and the landscape of the real-time web. *Proceedings of the Web Conference 2021*(2021). С. 15–23.

16. Ratner I. M., Harvey J. Vertical slicing: Smaller is better. *2011 Agile Conference*(2011). IEEE, 2011. ISBN 161284426X. С. 240–245.

17. Transactional Email API Service For Developers | Mailgun. URL: https://www.mailgun.com/ (дата звернення 09.06.2022).

18. Великодний С. С., Бурлаченко Ж. В., Зайцева-Великодна С. С. Розробка архітектури програмного засобу для управління мережевим плануванням реінжинірингу програмного проекту. *Сучасний стан наукових досліджень та технологій в промисловості.* Vol. 2, Issue 8. 26 с.

19. Зараз у кіно, Афіша - Кінотеатр Multiplex | Київ, Харків, Дніпро, Херсон і вся Україна. URL: https://multiplex.ua/ (дата звернення 13.05.2022).

20. Ічанська Н. В., Сіровий С. С. Адаптивна сітка CSS як новий підхід у розробці дизайну сайту. 2019.

21. Краліна Г., Баков Н. ОПЕРАЦІЙНІ СИСТЕМИ: ПОПУЛЯРНІСТЬ, ХАРАКТЕРИСТИКИ, ВИБІР. *InterConf*. 2020.

22. Планета Кіно в Києві (Blockbuster) — мережа кінотеатрів. URL: https://planetakino.ua/ (дата звернення 13.05.2022).

23. Статистика найбільш популярних браузерів в світі • Marketer. URL: https://marketer.ua/ua/stats-of-browsers-2017/ (дата звернення 23.05.2022).

24. Фільми в кінотеатрах України на KINOafisha.ua. URL: https://kinoafisha.ua/ (дата звернення 13.05.2022).

25. Солодковський Ю. М., Веремієнко Т. С. Діджиталізація міжнародного економічного розвитку. ДВНЗ «Київський національний економічний університет імені Вадима Гетьмана», 2020. ISBN 9669263166.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра інженерії програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

КОМПЛЕКС ЗАСТОСУНКІВ АВТОМАТИЗАЦІЇ БІЗНЕС-ПРОЦЕСІВ КІНОТЕАТРУ

СПЕЦІАЛЬНА ЧАСТИНА З ОХОРОНИ ПРАЦІ

ПИТАННЯ ОХОРОНИ ПРАЦІ РОЗРОБНИКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Спеціальність «Інженерія програмного забезпечення»

121 – КРБ.1– 409.21810912

Студент

_____ А. А. Жлуктарьов
підпис
«__»_____2022р.

Консультант канд. техн. наук**,** доцент _____А. О. Алексєєва
підпис
«__»_____2022р.

Миколаїв – 2022

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

АУП – автоматична установка пожежогасіння

дБ – децибел

ЕОМ – електронно-обчислювальна машина

ПК – персональний комп’ютер

ВСТУП

В Законі України «Про охорону праці»[1] визначається, що охорона праці – це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності.

Охорона праці – це дисципліна, яка забезпечує відповідність умов праці на робочому місці нормативним актами, а також дотримання безпеки технологічних процесів, машин, механізмів, стан засобів індивідуального захисту, що використовуються працівником під час роботи.

Метою охорони праці є науковий аналіз умов праці, технологічних процесів, апаратури і обладнання тощо задля того щоб передбачити можливу появу небезпечних факторів та запобігти виникненню інцидентів. Спираючись на цей аналіз визначаються можливі аварійні ситуації на виробництві, небезпечні ділянки виробництва, а також розробляються заходи направлені на їх усунення.

На кожному робочому місці мають бути заходи протидії можливого впливу небезпечних та/або шкідливих факторів. Рівні цих факторів повинні задовольняти визначених в відповідних документах технічними та санітарно-технічними нормами.

Ці нормативні документи описують яким чином необхідно створювати умови праці на робочому місці так, щоб вплив небезпечних та шкідливих факторів на співробітників було або усунуто повністю, або утримано в допустимих межах.

Задля реалізації комплексу застосунків для автоматизації бізнес-процесів кінотеатру було розглянуто наступні питання:

– аналіз пожежної безпеки під час використання ПК;

– розрахунок рівня звукового тиску;

– визначення ергономічних вимог під час використання ПК;

– аналіз рівня освітленості робочого місця;

– аналіз необхідних параметрів мікроклімату.

1 ПОЖЕЖНА БЕЗПЕКА ПІД ЧАС ВИКОРИСТАННЯ ПК

У сучасних комп'ютерах дуже висока щільність розміщення елементів електронних систем, в безпосередній близькості один від одного розташовуються сполучні дроти, комунікаційні кабелі, що може посприяти небезпеці різного роду загорянь. При цьому можливі оплавлення ізоляції сполучних проводів, їх оголення і, як наслідок, коротке замикання, яке супроводжується іскрінням, яке веде до неприпустимих перевантажень елементів електронних схем. При перенагріванні, вони згорають з розбризкуванням іскор.

У разі пожежі спрацьовує автоматична установка пожежогасіння. Найчастіше застосовуються газові АУП вони забезпечені світловою та звуковою сигналізацією. Для запобігання поширенню вогню під час пожежі з однієї частини будівлі на іншу влаштовують протипожежні перепони у вигляді протипожежних стін, перегородок, перекриттів, зон, тамбурів-шлюзів, дверей, вікон, люків, клапанів. У будівлі на випадок виникнення пожежі передбачається не менше двох евакуаційних виходів; але через машинний зал, який має теж менше двох виходів, не повинні проходити шляхи евакуації співробітників, працюючих в інших підрозділах.

Проходи, коридори і робочі місця не слід захаращувати архівними матеріалами, папером. На евакуаційних шляхах встановлюють як природне, так і штучне аварійне освітлення. В інших виробничих приміщеннях допускається проектувати один вихід, якщо відстань від найбільш віддаленого місця до виходу не перевищує 25 м, а кількість працюючих в зміні не більше 25 осіб [2]. Для зберігання носіїв інформації використовуються вогнетривкі металеві шафи, двері в сховище також повинні бути вогнетривкими. Комплекс організаційних і технічних заходів пожежної профілактики дозволяє запобігти пожежі, а в разі його виникнення забезпечити безпеку людей, обмежити поширення вогню, а також створити умови для успішного гасіння пожежі.

2 ЕРГОНОМІКА РОБОЧОГО МІСЦЯ

Проєктування робочих місць відноситься до числа найважливіших проблем ергономічного проєктування в області обчислювальної техніки.

Ергономічними аспектами проєктування робочих місць, зокрема, є: висота робочої поверхні, розміри простору для ніг, вимоги до розташування документів на робочому місці (наявність і розміри підставки для документів, можливість різного розміщення документів, відстань від очей користувача до екрану, документа, клавіатури тощо.), характеристики робочого крісла, вимоги до поверхні робочого столу, можливість регулювання елементів робочого місця [5].

Головними елементами робочого місця програміста є стіл і крісло. Основним робочим положенням є положення сидячи.

Робоча поза сидячи викликає мінімальне стомлення програміста. Раціональне планування робочого місця передбачає чіткий порядок і сталість розміщення предметів, засобів праці і документації. Те, що потрібно для виконання робіт частіше, розташоване в зоні легкої досяжності робочого простору.

Моторне поле – простір робочого місця, в якому можуть здійснюватися рухові дії людини.

Максимальна зона досяжності рук – це частина моторного поля робочого місця, обмеженого дугами, що описуються максимально витягнутими руками при русі їх в плечовому суглобі.

Оптимальна зона – частина моторного поля робочого місця, обмеженого дугами, описуваними передпліччями при русі в ліктьових суглобах з опорою в точці ліктя і з відносно нерухомим плечем.

Схематично зони досяжності наведено на рисунку 2.1. Розглянемо оптимальне розміщення предметів праці і документації в зонах досяжності рук.

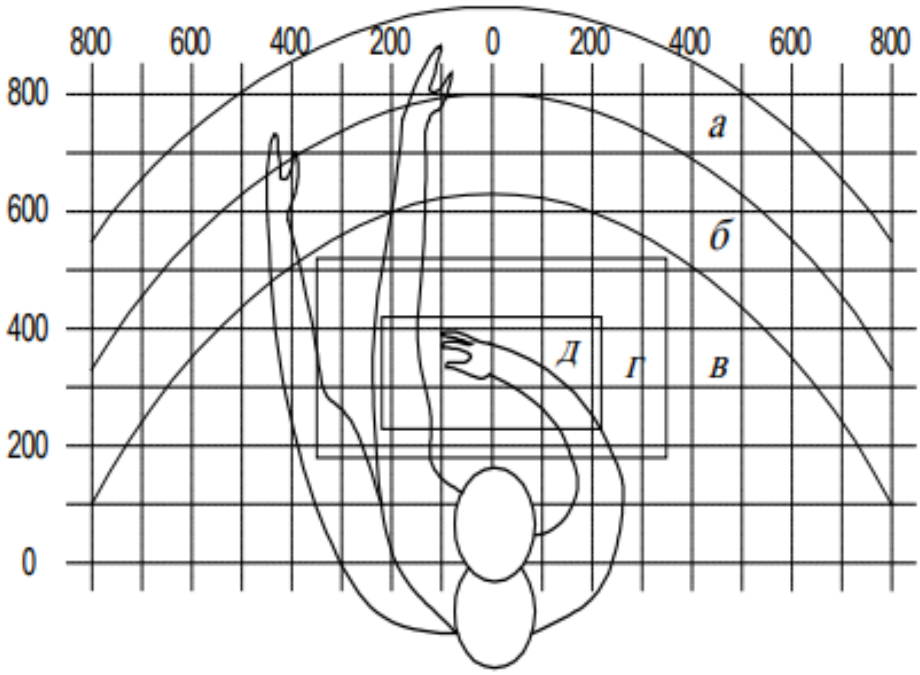


Рисунок 2.1 – Зони досяжності рук у горизонтальній площині:

а – зона максимальної досяжності; б – зона досяжності пальців при витягнутої руці; в – зона легкої досяжності долоні; р – оптимальний простір для грубої ручної роботи; д – оптимальний простір для тонкої ручної роботи.

а) дисплей розміщується в зоні а (у центрі);

б) клавіатура – у зоні г;

в) системний блок розміщується в зоні б (ліворуч);

г) література і документація, необхідна при роботі – в зоні в (ліворуч);

Для комфортної роботи стіл повинен задовольняти наступним умовам:

– висота столу повинна бути вибрана з урахуванням можливості сидіти вільно, в зручній позі, при необхідності спираючись на підлокітники;

– нижня частина столу повинна бути сконструйована так, щоб програміст міг зручно сидіти, ні змушений підбирати ноги;

– поверхня стола повинна мати властивості, що виключають появу відблисків в полі зору програміста;

– конструкція столу повинна передбачати наявність висувних ящиків.

– висота робочої поверхні рекомендується в межах 680-760мм. Висота поверхні, на яку встановлюється клавіатура, повинна бути біля 650мм.

Положення екрану визначається:

– відстанню зчитування (0,6 ... 0,7 м);

– кутом зчитування, напрямком погляду на 20 ° нижче горизонталі до центру екрану, причому екран перпендикулярний цьому напряму.

Велике значення також надається правильній робочій позі користувача. При незручній робочій позі можуть з'явитися болі в м'язах, суглобах і сухожиллях. Вимоги до робочої пози користувача відеотермінала наступні:

– голова не повинна бути нахилена більш ніж на 20 °;

– плечі повинні бути розслаблені;

– лікті – під кутом 80 ° ... 100 °;

– передпліччя і кисті рук – в горизонтальному положенні.

Причина неправильної пози користувачів обумовлена наступними факторами: немає гарної підставки для документів, клавіатура знаходиться дуже високо, а документи – низько, нікуди покласти руки і кисті, недостатній простір для ніг.

З метою подолання зазначених недоліків даються загальні рекомендації: краще пересувна клавіатура; повинні бути передбачені спеціальні пристосування для регулювання висоти столу, клавіатури і екрану, а також підставка для рук [6].

Під час користування комп'ютером медики радять встановлювати монітор на відстані 50–60 см від очей. Фахівці також вважають, що верхня частина дисплея повинна бути на рівні очей або трохи нижче. Коли людина дивиться прямо перед собою, його очі відкриваються ширше, ніж коли він дивиться вниз. За рахунок цього площа огляду значно збільшується, викликаючи обезводнення очей. До того ж якщо екран встановлений високо, а очі широко відкриті, порушується функція моргання. Це означає, що очі не закриваються повністю, не омиваються слізною рідиною, не отримують достатнього зволоження, що приводить до їх швидкої стомлюваності. [7]

Створення сприятливих умов праці і правильне естетичне оформлення робочих місць на виробництві має велике значення як для полегшення праці, так і для підвищення його привабливості, позитивно впливає на продуктивність праці.

3 АНАЛІЗ НОРМ ШКІДЛИВИХ ЧИННИКІВ

3.1 Параметри мікроклімату

Параметри мікроклімату можуть змінюватися в широких межах, в той час як необхідною умовою життєдіяльності людини є підтримка постійності температури тіла завдяки терморегуляції, тобто здатності організму регулювати віддачу тепла в навколишнє середовище. Принцип нормування мікроклімату – створення оптимальних умов для теплообміну тіла людини з навколишнім середовищем.

Обчислювальна техніка є джерелом тепла, що може приводить до підвищення температури і зниження відносної вологості в приміщенні, через що у приміщеннях, де встановлені комп'ютери, повинні дотримуватися певні параметри мікроклімату.

У санітарних нормах СН-245-71 встановлені величини параметрів мікроклімату, що створюють комфортні умови. Ці норми встановлюються в залежності від пори року, характеру трудового процесу і характеру виробничого приміщення (див. табл. 3.1).

Таблиця 3.1 – Норми мікроклімату робочої зони об’єкту

Період року	Категорія робіт	Температура C^0	Відносна вологість %	Швидкість руху повітря, м/с
Холодна	легка-1 а	22-24	40-60	0,1
Тепла	легка-1 а	23-25	40-60	0,1

Обсяг приміщень, в яких розміщені працівники обчислювальних центрів, не повинен бути менше 19,5м3 / людина з урахуванням максимального числа одночасно працюючих в зміну.

Норми подачі свіжого повітря в приміщення, де розташовані комп'ютери, приведені в табл. 3.2.

Таблиця 3.2 – Норми подачі свіжого повітря в приміщення, де розташовані комп'ютери

Характеристика приміщення	Об'ємна витрата подається в приміщення свіжого повітря, м ^{3} / на одну людину в годину
Об'єм до 20м ^{3}на особу	Не менше 30
20 ... 40м ^{3} на особу	Не менше 20
Більш 40м ^{3} на особу	Природна вентиляція

Для забезпечення комфортних умов використовуються як організаційні методи (раціональна організація проведення робіт залежно від пори року і доби, чергування праці і відпочинку), так і технічні засоби (вентиляція, кондиціонування повітря, опалювальна система).

3.2 Рівень освітленості робочого місця

Недостатність освітлення призводить до напруги зору, послаблює увагу, приводить до настання передчасної стомленості. Надмірно яскраве освітлення викликає осліплення, роздратування і різь в очах. Неправильний напрямок світла на робочому місці може створювати різкі тіні, відблиски, дезорієнтувати працюючого. Всі ці причини можуть призвести до нещасного випадку або профзахворювань, тому настільки важливий правильний розрахунок освітленості. Створення умов освітлення за стандартами покращує умови роботи зорової системи людини, що поліпшує продуктивність та впливає на психологічний та моральний стан людини.

Розрахуємо штучне освітлення приміщення у якому було виконано поставлену роботу. Площа приміщення складає 20 м^2, ширина якої 4м, довжина 5м, а висота 3м. Скористаємось відповідною формулою (3.1) [8].

$$F = \frac{E*K*S*Z}{n} \quad (3.1)$$

де F – світловий потік, що розраховується, лм;

E – мінімальна освітленість, лк; за стандартом E = 300лк;

S – площа приміщення (20 м^2);

Z – відношення середнього значення освітленості до мінімального (Z = 1.2);

K – коефіцієнт запасу, що враховує хибні чинники, що можуть впливати на світловий потік (K = 1.5);

n – коефіцієнт використання світлового потоку, що є відношенням світлового потоку яке падає на поверхню; має залежність від усіх характеристик ліхтаря, світильника, лампи, тощо. А також від розмірів приміщення, оздоблення стін і стелі, які можна охарактеризувати спеціалізованими коефіцієнтами відбиття від стін та стелі. $P_{стін}$ = 25%, $P_{стелі}$ = 60%.

Також задля аналізу приміщення потрібно обчислити індекс приміщення за формулою (3.2) [9].

$$I = \frac{S}{h(A+B)} \qquad (3.2)$$

де, S – площа приміщення (20 м^2);

h – висота приміщення (3 м);

A – ширина приміщення (4 м);

B – довжина приміщення (5 м);

$$I = \frac{20}{3*(5+4)} = 0.74 \qquad (3.3)$$

Спираючись на отримані дані та на табличні значення (таблиця 3.3), можна зазначити, що коефіцієнт становить n = 0,3.

Таблиця 3.3 – Значення коефіцієнту використання світлового потоку

	Коефіцієнт відображення, %							Коефіцієнт приміщення і
Стеля	70%				50%		30%	
Стіни	50%		30%		50%	30%	10%	
Підлога	30%	10%	30%	10%	10%			
Коефіцієнт використання світлового потоку	0,26	0,25	0,20	0,19	0,17	0,13	0,06	0,5
	0,3	0,28	0,24	0,23	0,2	0,16	0,08	0,6
	0,34	0,32	0,28	0,27	0,22	0,19	0,10	0,7
	0,38	0,36	0,31	0,30	0,24	0,21	0,11	0,8
	0,40	0,38	0,34	0,33	0,26	0,23	0,12	0,9
	0,43	0,41	0,37	0,35	0,28	0,25	0,13	1,0
	0,46	0,43	0,39	0,37	0,30	0,26	0,14	1,1

Використовуючи отримані дані, розрахуємо світловий потік. Підставимо значення до відповідної формули (3.4):

$$F = \frac{E*K*S*Z}{n} = \frac{300*1.5*20*1.2}{0.3} = 36\ 000 \quad (3.4)$$

Використовувані лампи мають світловий потік 4200Лк. Виходячи з цього, розрахуємо необхідну кількість ламп (3.5).

$$N = \frac{F_1}{F_2} \quad (3.5)$$

де, F_1 – необхідний потік;

F_2 – світловий потік лампи.

$$N = \frac{36000}{4200} \approx 9 \text{ ламп} \quad (3.6)$$

Було проаналізовано умови освітлення приміщення та виявлено, що для відповідної роботи приміщення має містити 9 ламп.

3.3 Рівень шумового тиску

Шумовий тиск має негативний вплив на розумову діяльність та її загальну продуктивність, оскільки він може викликати подразливість, складнощі у концентрації та навіть головні болі [3]. Щоб запобігти цьому, відповідно до нормативних джерел середній рівень шуму має не перевищувати 65 дБ [4].

Найшумнішими джерелами є навколишнє середовище та комп’ютер.

Для розрахунку приблизного рівня шуму на робочому місті визначимо головні джерела та їх приблизний рівень та сформуємо наступну таблицю (табл 3.4).

Таблиця 3.4 – Рівень шуму від різноманітних джерел

Джерело	**Рівень шуму**
Клавіатура	30
Миша	20
Кулер	20
Навколишнє середовище	60

Рівень шуму, підраховується за допомогою формули рівня звукового тиску. Формула дає можливість зрозуміти чи є показники допустимими задля роботи у приміщенні.

$$L = 10 \lg \sum_{i=1}^{n} 10^{0.1 \cdot L_i} \quad (3.7)$$

Де L_i – рівень звукового тиску;

n – кількість джерел.

Проведемо розрахунки для визначення рівня звукового тиску:

$$L = 10 \lg(10^3 + 10^2 + 10^2 + 10^{1.5}) = 10 \lg(1001200) \approx 60 \text{ дБ} \quad (3.8)$$

Оскільки отримане значення менше за середній допустимий рівень шуму, то можна стверджувати що умови праці задовольняють норми щодо шумового тиску.

ВИСНОВКИ

В спеціальній частині КРБ розглянуто необхідність дотримання норм охорони праці на робочому середовищі заради протидії множині шкідливих та/або небезпечних чинників.

Розглянуто питання пожежної безпеки під час використання ПК та у серверних приміщеннях, потенціальні джерела виникнення пожеж, а також сформовано план дій, який спрямовано на нівелювання можливості такого інциденту. Наведено поради, яких рекомендованно дотримуватись у разі виникнення пожежі.

Також проаналізовано ергономічні норми для роботи за столом ПК: оптимальне розміщення предметів праці на робочому місці програміста, висоту та конструкцію столу, коректну позу працівника тощо. Для цього аналізу наведено нормативні документи та схеми.

Наприкінці досліджено вплив таких шкідливих та потенціально небезпечних чинників як:

– недостатня освітленість приміщення;

– несприятливий мікроклімат;

– завеликий рівень шуму.

Для приміщення, в якому проведено розробку ПЗ, визначено приблизний рівень шуму та його відповідність нормам, необхідну кількість ламп для коректного освітлення.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Закон України Про охорону праці від 14.08.2021 №49 URL: https://zakon.rada.gov.ua/laws/show/2694-12#Text (дата звернення: 13.06.2022).

2. Про затвердження Правил пожежної безпеки в Україні: Наказ міністерства внутрішніх справ України від 30.12.2014 № 1417 URL: https://zakon.rada.gov.ua/laws/show/z0252-15#Text (дата звернення: 13.06.2022).

3. Стаття про фізіологічні зміни в організмі від рівня шуму. URL: https://www.unian.ua/health/worldnews/874784-pidvischeniy-riven-shumu-zumovlyue-fiziologichni-zmini-v-organizmi.html/ (дата звернення: 13.06.2022).

4. Допустимі норми шуму в житлових та громадських приміщеннях. URL: https://biz.ligazakon.net/news/188395_dopustim-normi-shumu-v-zhitlovikh-ta-gromadskikh-primshchennyakh/ (дата звернення: 13.06.2022).

5. Зинченко В. П., Мунипов В. М. Основы эргономики. *М.: Издательство Московского университета*. 1979.

6. Стаття про вимоги до робочого місця програміста. URL: https://vuzlit.com/87295/trebovaniya_rabochemu_mestu_programmista (дата звернення: 13.06.2022).

7. Стаття про правила безпечної роботи за комп’ютером. URL: https://pedcollege.kiev.ua/index.php/77-robota-koledzhu/okhorona-pratsi/589-pravyla-bezpechnoi-roboty-na-kompiuteri (дата звернення: 13.06.2022).

8. Стаття про розрахунок освітленості приміщення. URL: http://ekobil.com.ua/kalkulyatori-rozrahunku-osvitlenosti-primishhennya/ (дата звернення: 13.06.2022).

9. Стаття про розрахунок освітленості приміщення. URL: https://www.brille.ua/ua/kak-rasschitat-uroven-osveshennosti-pomesheniya/ (дата звернення: 13.06.2022).

ЗВІТ

про перевірку на унікальність кваліфікаційної роботи бакалавра на тему:
«Комплекс застосунків автоматизації бізнес-процесів кінотеатру»

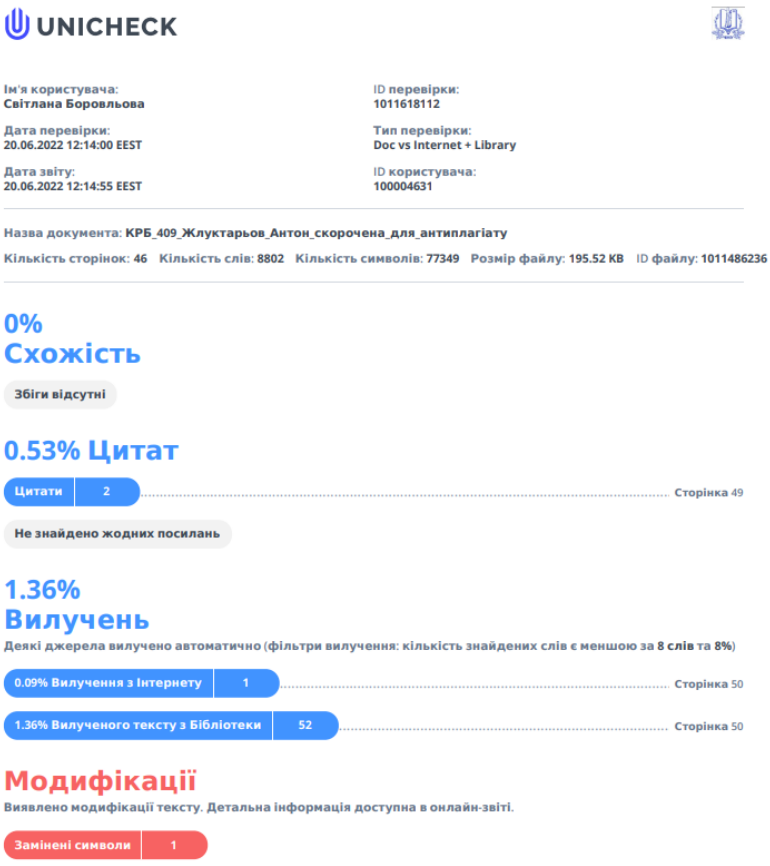
студента спеціальності 121 «Інженерія програмного забезпечення», 409 групи

Жлуктарьова Антона Андрійовича

прізвище, ім’я, по батькові

Перевірку тексту здійснено сервісом: онлайн-сервіс Unicheck

Результат перевірки тексту кваліфікаційної роботи бакалавра: схожість складає 0%.



UNICHECK

Ім'я користувача: Світлана Боровльова
ID перевірки: 1011618112
Дата перевірки: 20.06.2022 12:14:00 EEST
Тип перевірки: Doc vs Internet + Library
Дата звіту: 20.06.2022 12:14:55 EEST
ID користувача: 100004631

Назва документа: КРБ_409_Жлуктарьов_Антон_скорочена_для_антиплагіату

Кількість сторінок: 46 Кількість слів: 8802 Кількість символів: 77349 Розмір файлу: 195.52 KB ID файлу: 1011486236

0%
Схожість

Збіги відсутні

0.53% Цитат

Цитати 2 Сторінка 49

Не знайдено жодних посилань

1.36%
Вилучень

Деякі джерела вилучено автоматично (фільтри вилучення: кількість знайдених слів є меншою за 8 слів та 8%)

0.09% Вилучення з Інтернету 1 Сторінка 50

1.36% Вилученого тексту з Бібліотеки 52 Сторінка 50

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи 1

Студент:

_____ А. А. Жлуктарьов
підпис ініціали, прізвище

Керівник:
ст. викладач кафедри інженерії програмного забезпечення

_____С. Ю. Боровльова
підпис ініціали, прізвище

Дата: «___» _____ 2022 р.

АНОТАЦІЯ

до кваліфікаційної роботи бакалавра

«Комплекс застосунків автоматизації бізнес-процесів кінотеатру»

Студент 409 гр.: Жлуктарьов Антон Андрійович

Керівник: старший викладач Боровльова Світлана Юріївна

Кваліфікаційна робота бакалавра присвячена створенню комплексу застосунків для автоматизації бізнес-процесів кінотеатру. Робота особливо актуальна під час тривалої кризи в індустрії українських кінотеатрів, яка виникла внаслідок війни та пандемії COVID-19, оскільки розроблюване ПЗ дозволяє економити кошти за рахунок автоматизації праці співробітників кінотеатру та приваблює клієнтів підвищеною зручністю використання його сервісу.

Об’єкт роботи – бізнес-процеси кінотеатру.

Предмет роботи – підходи та інформаційні технології для автоматизації бізнес-процесів кінотеатру.

Мета – автоматизація бізнес-процесів кінотеатру шляхом розробки комплексу веб та мобільних застосунків.

У першому розділі розглянуто актуальність автоматизації бізнес-процесів кінотеатру, а також проведено порівняльний аналіз існуючих аналогів. У другому розділі детально розглянуто сценарії використання ПЗ та створено прототипи його інтерфейсу. У третьому розділі досліджуються необхідні архітектурні рішення та концепції з інженерії програмного забезпечення. В четвертому розділі описано процеси кодування спроєктованого застосунку та створення необхідної документації. В останньому розділі розглянуто необхідні норми з охорони праці й техніки безпеки для роботи розробника ПЗ.

В результаті виконаної роботи реалізовано комплекс застосунків та були зроблені висновки щодо можливості використання інформаційних технологій для автоматизації бізнес-процесів кінотеатру.

КРБ викладена на 60 сторінок, вона містить 4 розділи, 29 ілюстрацій, 9 таблиць, 25 джерел в переліку посилань.

Ключові слова: кінотеатр, CQRS, Clean Architecture, SPA, WS.

ABSTRACT

of the Bachelor's Thesis

«Complex of applications for automatization of cinema business-processes»

Student of group 409: Zhluktarov Anton Andriiovych

Supervisor: Senior Instructor Borovlova Svitlana Yuriivna

The Bachelor's Thesis is dedicated to constructing a complex of applications designed to automate cinema business-processes. This paper is especially of interest nowadays during the lasting crisis of Ukrainian cinematic industry, which was caused by the ongoing war and the COVID-19 pandemic, as constructed software allows cutting the costs via automatization of cinema workers' labour, and it also attracts customers because of the increased comfort of cinema's service usage.

The object of thesis are cinema business-processes.

The subject of thesis are approaches and information technologies for automatization of cinema business-processes.

The goal is to automatize cinema business-processes via developing a complex of Web and Mobile applications.

The first section overviews relevance of automatization of cinema business process and performs a comparative analysis of already existing alternatives. The second section describes use-cases of the designed complex of applications and contains user interface prototypes. The third section researches necessary architectural solutions and concepts from the field of software engineering. The fourth section presents processed of coding the constructed complex and creating necessary documentation. The last section overviews the necessary norms of labor protection and safety for the job of software engineer.

As a result of the work performed, a complex of applications has been developed and conclusions about the possibilities of using information technologies for automatization of cinema business processes.

The Bachelor's Thesis spans 60 pages, it contains 4 sections, 29 illustrations, 9 tables, 25 sources in the reference list.

Keywords: cinema, CQRS, Clean Architecture, SPA, WS.

ВІДГУК
на кваліфікаційну роботу бакалавра
студента групи 409 ЧНУ імені Петра Могили
Жлуктарьова Антона Андрійовича
«Комплекс застосунків автоматизації бізнес-процесів кінотеатру»

Тема кваліфікаційної роботи бакалавра Жлуктарьова Антона Андрійовича присвячена проєктуванню та розробці комплексу застосунків для кінотеатру, метою якого є автоматизація бізнес-процесів. Тема роботи є актуальною, оскільки вона описує використання сучасних технологій для вирішення задачі синхронізації даних в режимі реального часу та має пом'якшити кризу кінотеатрів, спричинену війною та пандемією COVID-19.

Кваліфікаційна робота складається з чотирьох розділів, в яких розкрито актуальність теми, наведено аналіз застосунків-аналогів, створено специфікацію вимог, а також описано розроблену архітектуру та вибрані технології кодування. Крім того, в рамках окремої підзадачі описано реалізацію механізму автоматичної синхронізації даних в режимі реального часу.

Під час виконання роботи Жлуктарьов Антон Андрійович дотримувався термінів календарного плану, дослідив достатню кількість наукових робіт (в тому числі англомовних) та використав останні дослідження і технології в написані кваліфікаційної роботи, продемонстрував високий рівень професійних знань та практичних навичок, старанність, наполегливість та дисциплінованість, професійну підготовку та спроможність самостійно приймати проєктні рішення. При роботі над проєктом використовувався наступний стек технологій, мов, та фреймворків: C#, ASP.NET Core, Microsoft SQL Server, TypeScript, Vue, BootstrapVue, React, NGINX, Docker.

Теоретична значимість роботи полягає в детальному описі побудови архітектури системи з використанням сучасних методів та дослідженні використання механізмів для спілкування в режимі реального часу.

Практична значимість роботи полягає в використанні оптимальних інструментів для реалізації архітектури та роботі з інструментами спілкування в режимі реального часу.

Вміст кваліфікаційної роботи бакалавра викладено грамотно та структуровано, з використанням достатньої кількості графічних та схематичних матеріалів. Пояснювальна записка оформлена відповідно до існуючих вимог до оформлення технічної документації.

Кваліфікаційна робота бакалавра Жлуктарьова А. А. виконана на високому професійному рівні. Вважаю за можливе допустити роботу Жлуктарьова А. А. до захисту та присвоїти освітню кваліфікацію «бакалавр з інженерії програмного забезпечення» за спеціальністю 121 «Інженерія програмного забезпечення» в галузі знань 12 «Інформаційні технології».

Керівник кваліфікаційної роботи бакалавра:
ст. викладач кафедри інженерії
програмного забезпечення
ЧНУ імені Петра Могили С. Ю. Боровльова

РЕЦЕНЗІЯ

на кваліфікаційну роботу бакалавра

студента групи 409 ЧНУ імені Петра Могили

Жлуктарьова Антона Андрійовича

«Комплекс застосунків автоматизації бізнес-процесів кінотеатру»

Кваліфікаційна робота присвячена розробці комплексу застосунків з метою автоматизації бізнес-процесів кінотеатру. Дана задача є актуальною, оскільки вона описує використання сучасних технологій для вирішення задачі синхронізації даних в режимі реального часу та має пом’якшити кризу кінотеатрів, спричинену війною та пандемією COVID-19.

В першому розділі здійснено пошук та аналіз застосунків-аналогів, описано стандартний цикл розробки вебзастосунків при використанні інженерного підходу, а також створено специфікацію вимог до розроблюваного програмного забезпечення.

В другому розділі проводиться деталізація специфікації вимог шляхом складання діаграми прецедентів, сценаріїв використання та прототипів користувацького інтерфейсу для кожного компоненту з комплексу. Спираючись на функціональні вимоги, виокремлено необхідну частину прикладної сфери кінотеатру та спроєктовано модель бази даних.

У третьому розділі розглядаються сучасні концепції інженерії програмного забезпечення та шляхи їх використання. Описується склад комплексу застосунків, методи його розгортання, а також досліджується оптимальна архітектура рішення. Для кожного з створюваних компонентів наведено аргументацію щодо вибраних технологій, мов розробки, бібліотек, фреймворків тощо.

У четвертому розділі наведено уривки з вихідного коду створеної системи, які ілюструють найбільш визначні з точки зору кодування особливості розробки, разом з знімками екрану, які демонструють процес роботи програмного забезпечення.

Структура кваліфікаційної роботи бакалавра відповідає вимогам та поставленому завданню. Робота виконана в повному обсязі та у встановлений

термін. Обрані програмні засоби є ефективними та обґрунтованими, поставлена мета досягнута.

Серед недоліків слід відмітити відносно недостатню кількість розглянутих методів розгортання застосунку. Відзначений недолік не зменшує в цілому позитивне враження від роботи, яку виконано на високому рівні

Автор демонструє високий рівень володіння набутими знаннями з навчального матеріалу та здатність до їх практичного використання.

Враховуючи вищенаведене, вважаю, що кваліфікаційна робота є завершеною і заслуговує оцінки відмінно, а її автор Жлуктарьов А. А. − присвоєння освітньої кваліфікації «бакалавр з інженерії програмного забезпечення» за спеціальністю 121 «Інженерія програмного забезпечення» галузі знань 12 «Інформаційні технології».

Рецензент:
д-р фіз.-мат. наук, проф.
професор кафедри ІІС
ЧНУ ім. Петра Могили Е.А. Лисенков