Реалізація проекту на React

Модель кінотеатру з бронюванням квитків в кінозалі

Загальний опис проєкту

Створити веб-застосунок для моделювання роботи кінотеатру, який дозволяє користувачам переглядати картки фільмів, вибирати місця в залі та бронювати квитки. Проєкт реалізується з використанням git за допомогою React, з використанням State Management, Routing, API-запитів та інтерактивного UI.

Оцінювання проекту та наступний розвиток

Проект включає три лабораторні роботи 8, 9, 10 (по 4 бали) та модуль 2 (5 балів) – це перевірка репозиторію GitHub (вимоги нижче).

Додатково

За виявленим бажанням студента робота Модель кінотеатру (лаб. 8-9-10) може бути додатково розглянута ментором, а також продовжена за подвійною згодою у вигляді літньої школи та літньої практики в IT компанії (надіслати посилання на github у відповідне поле на moodle).

Вимоги

Використання GitHub для розробки проекту

- Створіть публічний репозиторій на GitHub.
- Всі зміни потрібно зберігати в цьому репозиторії з самого початку роботи над проєктом.
- Створюйте коміти достатньо часто, але тільки з логічно завершеними змінами.
- Використовуйте осмислені повідомлення до комітів.
- Синхронізовуйте локальні коміти з репозиторієм на GitHub регулярно.
- Для виконання завдань використовуйте JavaScript/TypeScript та React.

Відображення списку актуальних фільмів через створення карток для фільмів. Опційно: сеансів фільмів, можливістю фільтрації…. Лаб.8

Створення кінозалу з можливістю бронювання квитків. Інтерактивна схема залу з можливістю вибору місць. Колірна індикація для - вільні місця, заброньовані, обрані користувачем. Лаб.9

Технічні вимоги до проекту

1. Архітектура
   - Використовувати компонентний підхід. Створення окремих компонентів для відображення картки фільму та списку: MovieCard,
     Компонентний підхід (розбиття на окремі компоненти: MovieCard, SeatSelection, BookingForm).
     State Management:
     useState для локальних змін.
     useContext або Redux для глобального стану (обране місце, дані користувача).
     React Router для маршрутизації сторінок.
2. Збереження даних:
   Використання localStorage для тимчасового збереження вибору користувача.
   JSON-файл або Firebase для зберігання бронювань.
3. UI/UX
   Використання CSS Modules, Styled Components або Tailwind CSS.
   - Стилізація блоків, робота з **display: flex / grid**.
   - Використання **box-shadow, border-radius, hover, transition** для створення ефектів.

   Інтерактивні елементи (hover, active), адаптивність через media queries.
   4. API-запити
4. Використання fetch() або Axios для отримання списку фільмів та бронювання місць.
5. Запити до mock-сервера (json-server) або API для отримання фільмів чи розкладу.

Теоретичні відомості

- React: основи компонентного підходу, JSX.
- Props та State у компонентах.
- Робота з useState та useEffect для оновлення інтерфейсу.

- Методи масивів JavaScript (map(), filter(), find()) для роботи зі списками.
- LocalStorage для збереження даних.
- Основи CSS Flexbox/Grid для розташування елементів.

Лабораторна робота 8

Картки фільмів та дошка з картками

1. Налаштування орієнтовної структури проєкту:

```
/src
├── components
│   ├── MovieCard.jsx
│   ├── MovieList.jsx
├── data
│   ├── movies.js
├── App.jsx
├── index.css
├── main.jsx
```

2. Реалізація компонентів

3. Компонент MovieCard.jsx (Одна картка фільму)

Створити компонент MovieCard, який отримує дані через props та відображає:

- Зображення постера.
- Назву фільму.
- Опис.
- Жанр.
- Дату та час сеансу

4. Компонент MovieList.jsx (Список карток фільмів)

Цей компонент повинен:

- Приймати список фільмів (movies) у props.
- Використовувати .map() для рендерингу MovieCard.

5. Реалізувати пошук за назвою фільму у відповідному полі зверху списку

5. Файл movies.js (Дані фільмів)

Створити масив з кількома фільмами для відображення у MovieList.

6. Компонент App.jsx (Головний компонент)

- Імпортувати список фільмів з movies.js.
- Передавати список у MovieList та відобразити його

7. Оформлення стилів

- Використати CSS Grid або Flexbox для розташування карток.
- Створити ефекти при наведенні (hover, box-shadow).
- Забезпечити адаптивність через @media.

Лабораторна робота 9

Реалізація кінозалу з відображенням вільних та заброньованих місць

Завдання

1. Оновити структуру проекту

```
/src
├── components
│   ├── MovieList.jsx
│   ├── MovieCard.jsx
│   ├── CinemaHall.jsx
├── pages
│   ├── Home.jsx
│   ├── Booking.jsx
├── App.jsx
```

├── main.jsx
2. Реалізація сторінок та компонентів
3. Налаштування роутінгу (App.jsx)
   - Налаштувати маршрутизацію за допомогою react-router-dom.
   - Створити дві сторінки:
   - Головна (Home.jsx) – містить список фільмів, які відображені за допомогою компоненти MovieList.
   - Сторінка бронювання (Booking.jsx) – показує кінозал та дозволяє вибрати місця.
4. Компонент CinemaHall.jsx (кінозал)
   - Відображати сітку місць (сидіння).
   - Додавати можливість вибору місць (зміна стану useState).
   - Список вибраних місць повинен відображатися та оновлюватися, якщо користувач робить зміни.
5. Оновлення MovieCard.jsx
   - Додати кнопку "Забронювати", яка перенаправляє на /booking/:id.
6. Оновлення Booking.jsx
   - Отримувати id фільму через React Router params.
   - Відображати кінозал з можливістю вибору місць.
7. Оформлення стилів
   - Використати CSS Grid для розміщення місць у кінозалі.
   - Виділити місця вибрані (синій) доступні (зелений), .
   - Додати адаптивність для мобільних пристроїв.

Очікуваний результат

Після виконання роботи студент отримає React-додаток, що дозволяє переглядати список фільмів, деталі про окремий фільм, схему місць в кінозалі та вибирати місця для бронювання.

Лабораторна робота 10

Реалізація бронюванням квитків користувачем

Завдання

1. Оновити структуру проекту

/src
├── components
│ ├── MovieList.jsx
│ ├── MovieCard.jsx
│ └── CinemaHall.jsx
├── pages
│ ├── Home.jsx
│ └── Booking.jsx
services
BookingService.js
├── App.jsx
├── main.jsx

2. Реалізація сторінок та компонентів
3. Тут реалізувати збереження в LocalStorage чи FireBase. Отримувати звідти дані про бронювання та відображати в кінозалі

3. Реалізувати BookingSrvice з методами збереження бронювання та отримання забронюваних місць для фільму за id фільму

3. Реалізувати збереження вибраних місць для бронювання в LocalStorage чи Firebase
   - Користувач вибирає список місць для бронювання
   - Натискає кнопку “Забронювати”
   - Користувачу показується форма для введення
     - Імя
     - Телефон
     - Емейл
   - Всі поля є обов'язковими. Перевірити чи емейл введено в правильному форматі
   - Якщо поля не заповнені, або у невірному форматі показати помилку валідації
   - Коли всі поля введено правильно зберегти бронювання використовуючи BookingService

- Після успішного збереження показати сповіщення, використовуючи https://www.npmjs.com/package/react-toastify або аналог

4. Реалізувати відображення заброньованих та вільних місць
   - Отримувати id фільму через React Router params
   - Під час завантаження сторінки Booking, показувати заброньовані та вільні місця
   - Дані отримати з BookingService
6. Оформлення стилів
   - Виділити заброньовані місця (червоний) .
   - Додати адаптивність для мобільних пристроїв.