Lab 03 - React

Tworzenie projektu React o tematyce gier komputerowych

Dzisiaj stworzymy nowy projekt w React i zbudujemy kilka prostych komponentów związanych z tematyką gier komputerowych. Skupimy się na stanie komponentów i dodawaniu własnych styli. Projekt ten będziemy rozwijać przez dalszą część semestru, aby utworzyć jedną, większą, bardziej złożoną aplikację.

1. Inicjalizacja nowego projektu

Na początek utwórzmy nowy projekt React zgodnie z instrukcją podaną podczas ostatnich zajęć.

1. Otwórz WebStorm.
2. Kliknij Create New Project.
3. Wybierz React App.
4. Wybierz lokalizację projektu
5. Nadaj nazwę projektu: games-project
6. Kliknij Create.
7. Upewnij się, że WebStorm automatycznie uruchomi
   npx create-react-app games-project
8. W terminalu wpisz
   cd games-project
9. Uruchom aplikację
   npm start

Teraz nasza aplikacja powinna się uruchomić pod localhost:3000.

Powinniśmy zobaczyć startowy defaultowy projekt React w przeglądarce.

2. Struktura projektu

W katalogu src/ utworzymy folder components/, w którym będziemy przechowywać nasze komponenty.

Dodaj folder *components* do src.

Na koniec dzisiejszych zajęć powinniśmy otrzymać poniższą strukturę w tym katalogu:

src/

|-- components/

| |-- Header.js

| |-- GameList.js

| |-- LoadMore.js

| |-- GameCard.js

| |-- Footer.js

| |-- FavoriteGames.js

| |-- ThemeToggle.js

| |-- RandomGame.js

W katalogu src utwórz również folder *styles*, w którym będziemy przechowywać style komponentów.

Teraz możemy przejść do tworzenia odpowiednich komponentów!

3. Tworzenie komponentu nagłówka

Zacznijmy od utworzenia komponentu, będącego nagłówkiem naszej aplikacji. Element ten będzie zawierać nazwę naszego projektu oraz krótkie przywitanie osób odwiedzających stronę.

Dodaj nowy plik Header.js do folderu components.

Utwórz także plik Header.css w folderze styles, który wykorzystamy do nadania styli.

Header.js:

```
import React from 'react';

const Header = () =>  {
  return (
    <header className="header">
      <h1>games-project</h1>
      <h2>Witamy w świecie gier!</h2>
      <p>Odkrywaj i poznawaj nowe tytuły.</p>
    </header>
  );
};
export default Header;
```

Umieść komponent Header (Header.js) w komponencie App (App.js).

Następnie nadaj style dla tego komponentu.

Header.css:

```
.header {
  background: #282c34;
  color: white;
  padding: 20px;
  text-align: center;
}
```

Na koniec dodaj import Header.css do Header.js.

```
import '../styles/Header.css';
```

Następnie zobacz jak wygląda ostylowany komponent na stronie internetowej.

4. Tworzenie komponentu GameCard pozwalającego wyswietlić pojedynczy kafelek gry

Utworzymy teraz komponent, który będzie otrzymywać pojedynczą grę jako props oraz wyświetla jej tytuł i gatunek.

Utwórz nowy plik GameCard.js.

```
import React from 'react';
const GameCard = ({ game }) => {
  return (
    <div className="game-card">
      <h3>{game.title}</h3>
      <p>Gatunek: {game.genre}</p>
    </div>
  );
};
export default GameCard;
```

Następnie dodajmy plik GameCard.css, w którym znajdą się style tego komponentu.

```
.game-card {
  border: 1px solid #ddd;
  padding: 10px;
  margin: 10px;
  border-radius: 5px;
  background-color: #f9f9f9;
  transition: 0.3s;
}
```

```
.game-card:hover {
  background-color: #e0e0e0;
}
```

Dodaj import GameCard.css w GameCard.js.

5. Tworzenie listy gier - GameList

Utworzymy komponent, który wyświetla na stronie listę gier, wykorzystując GameCard. Lista gier znajdzie się w osobnym pliku. Elementy będzie przekazywać do GameList jako props.

Dodajmy nowy plik GameList.js.

```
import React from 'react';
import GameCard from './GameCard';
const GameList = ( {gamesData}  ) => {
  return (
    <div>
      <h2>Lista gier</h2>
      <ul>
        { gamesData.map((game) => (
          <GameCard key={game.id} game={game} />
        ))}
      </ul>
    </div>
  );
};
export default GameList;
```

Teraz utworzymy plik z danymi dla GameList.

Dodaj plik GamesData.js.

```
const gamesData = [
  { id: 1, title: 'The Witcher 3', genre: 'RPG' },
  { id: 2, title: 'Cyberpunk 2077', genre: 'Sci-Fi RPG' },
  { id: 3, title: 'Dark Souls III', genre: 'Action RPG' },
];
export default gamesData;
```

Następnie umieść komponent GamesList w komponencie App przekazując gamesData jako props.

5. Tworzenie przycisku "Załaduj więcej"

Utworzymy teraz komponent, który pozwala użytkownikowi wczytać więcej gier do listy. Gry zostaną załadowane po kliknięciu w przycisk.

Ten komponent będzie zmieniać swoje zachowaniu w zależności od stanu kliknięcia w przycisk. Do implementacji użyjemy więc useState, pozwalający na przechowanie stanu komponentu. W poniższym przykładzie:
const [loaded, setLoaded] = useState(false);
loaded to nazwa zmiennej-stanu, natomiast *setLoaded* jest metodą pozwalającą na zmianę stanu *loaded*. Za pomocą *useState(false)* określamy początkowy stan na false – lista nie będzie wyświetlana, dopóki użytkownik nie kliknie w przycisk.

Utwórz nowy plik LoadMore.js:

```
import React, { useState } from 'react';
import GameList from "./GameList";
const extraGames = [
   { id: 4, title: 'Elden Ring', genre: 'RPG' },
   { id: 5, title: 'Horizon Zero Dawn', genre: 'Action RPG' }
];
```

```
const LoadMore = () => {
    const [loaded, setLoaded] = useState(false);
    const loadMoreGames = () => { setLoaded(!loaded); };
    return ( <div>
        <button onClick={loadMoreGames}>
            {loaded ? 'Schowaj'  : 'Załaduj więcej gier'}
        </button>
            {loaded ? <GameList gamesData={extraGames} /> : '' }
        </div>
    );
};
export default LoadMore;
```

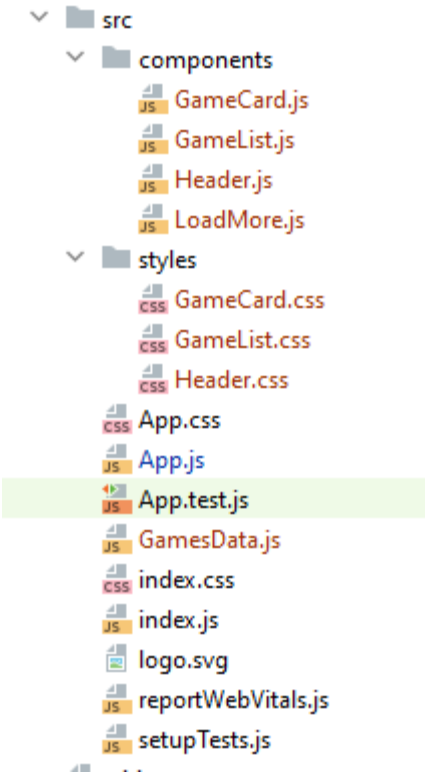
Następnie umieść komponent w App i przetestuj jego działanie.

Zadania do samodzielnego rozwiązania:

1. Dodaj prosty komponent Footer, będący stopką naszej aplikacji.
2. Utwórz komponent FavouriteGames, który po kliknięciu w przycisk wyświetli listę Twoich ulubionych gier.
3. Utwórz komponent ThemeToggle, który przy użyciu przycisku pozwoli się przełączać pomiędzy dwoma backgroud-color.
4. Utwórz komponent RandomGame, który po kliknięciu wyświetla nazwę jednej losowej gry z listy gier.

POMOC:

Struktura projektu katalogu src/



Header.js

```
import React from 'react';
import '../styles/Header.css';
const Header = () =>  {
    return (
        <header className="header">
            <h1>games-project</h1>
            <h2>Witamy w świecie gier!</h2>
            <p>Odkrywaj i poznawaj nowe tytuły.</p>
        </header>
    );
};
export default Header;
```

Header.js Header.css

```css
.header {
    background: #282c34;
    color: white;
    padding: 20px;
    text-align: center;
}
```

GameCard.js

```jsx
import React from 'react';
import '../styles/GameCard.css';
const GameCard = ({ game }) => {
    return (
        <div className="game-card">
            <h3>{game.title}</h3>
            <p>Gatunek: {game.genre}</p>
        </div>
    );
};
export default GameCard;
```

GameCard.css

```css
.game-card {
    border: 1px solid #ddd;
    padding: 10px;
    margin: 10px;
    border-radius: 5px;
    background-color: #f9f9f9;
    transition: 0.3s;
}

.game-card:hover {
    background-color: #e0e0e0;
}
```

GamesData.js

```
const gamesData = [
    { id: 1, title: 'The Witcher 3', genre: 'RPG' },
    { id: 2, title: 'Cyberpunk 2077', genre: 'Sci-Fi RPG' },
    { id: 3, title: 'Dark Souls III', genre: 'Action RPG' },
];
export default gamesData;
```

GameList.js

```
import React from 'react';
import GameCard from './GameCard';
const GameList = ( {gamesData}  ) => {
    return (
        <div>
            <h2>Lista gier</h2>
            <ul>
                { gamesData.map((game) => (
                    <GameCard key={game.id} game={game} />
                ))}
            </ul>
        </div>
    );
};
export default GameList;
```

LoadMore.js

```jsx
import React, { useState } from 'react';
import GameList from "./GameList";

const extraGames = [
    { id: 4, title: 'Elden Ring', genre: 'RPG' },
    { id: 5, title: 'Horizon Zero Dawn', genre: 'Action RPG' }
];
const LoadMore = () => {
    const [loaded, setLoaded] = useState( initialState: false);
    const loadMoreGames = () => { setLoaded(!loaded); };
    return ( <div>
        <button onClick={loadMoreGames}>
            {loaded ? 'Schowaj'  : 'Załaduj więcej gier'}
        </button>
            {loaded ? <GameList gamesData={extraGames} /> : '' }
        </div>
    );
};
export default LoadMore;
```

App.js

```jsx
import './App.css';
import Header from "./components/Header";
import gamesData from "./GamesData";
import GameList from "./components/GameList";
import LoadMore from "./components/LoadMore";

function App() {
  return (
    <div className="App">
      <Header />
        <GameList gamesData={gamesData} />
        <LoadMore />
    </div>
  );
}

export default App;
```