Final Project Presentation

Created by Wiseman Chali Shoto

For our final project from the Dynamic Web Applications Module, we had to create a Podcast Web App. The Podcast App allows us to browse through various genres of podcasts, play some episodes and even track your favourites episodes.

In the next page, you will be presented with the user stories we were given to consider this project as successful.

Following that I will take you through how I built Podcast app.

User Stories

- User can view the details of a show broken down into seasons, sorted by number
- User can listen to any episode in a season of a show
- User can see a view where only episodes for a specifically selected season are shown
- User can toggle between different seasons for the same show
- User can see the name of all available shows on the platform
- User sees preview image of shows when browsing
- User sees the amount of seasons per show as a number when browsing
- User sees a human-readable date as to when a show was last updated

User Stories

- User sees what genres (as genre titles) a show is associated with when browsing
- User sees a preview image of seasons for a specific show
- User sees the amount of episodes in a season as a number
- User can go back to a show view from a season-specific view
- User can mark specific episodes as favourites to find them again
- User can visit a view where they see all their favourites
- User can see the show and season of any episode in their favourites list
- Episodes related by season/show are grouped in favourites

User Stories

- User is able to remove episodes from their favourites
- User can arrange the list of shows by title from A-Z
- User can arrange the list of shows by title from Z-A
- User can arrange the list of shows by date updated in ascending order
- User can arrange the list of shows by date updated in descending order
- User can filter shows by title through a text input
- User can find shows based on fuzzy matching of strings (you can use something like

https://fusejs.io/)

User Stories

- User sees the date and time that an episode was added to their favourites list
- User can arrange favourites by show titles from A-Z
- User can arrange favourites by show titles from Z-A
- User can arrange favourites by date updated in ascending order
- User can arrange favourites by date updated in descending order
- Audio player shows current progress and episode length as timestamps
- Audio player is always visible, so the user can listen to episodes while they browse
- User is prompted to confirm they want to close the page when audio is playing
- App remembers which show and episode the user listened to last when returning to the platform
- App remembers which shows and episodes the user listened to all the way through

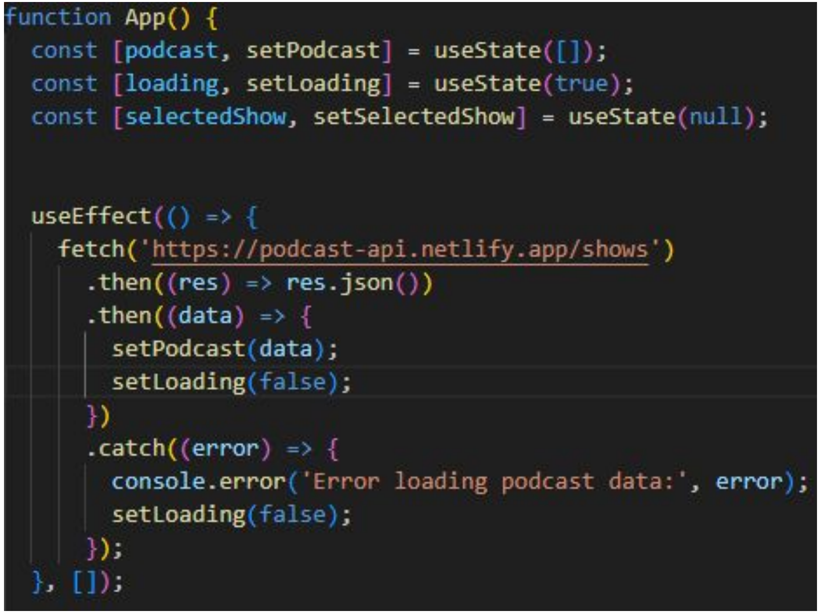
User Stories

- App remembers the timestamp where the user stopped listening within a 10-second accuracy period of closing
- App remembers and shows the timestamp progress of any episode the user has started listening to
- User can "reset" all their progress, effectively removing their listening history
- User is presented with a sliding carousel of possible shows they might be interested in on the landing page
- User can log in via https://app.supabase.com authentication
- User favourites are stored in the https://app.supabase.com database
- User favourites are automatically synced when logged in, ensuring that they share favourites between devices
- Users can share their favourites as a publicly accessible URL

Fetch Calls and Data loading

The function displayed is named App that fetches podcast data from an API when the component mounts. It uses hooks (useState and useEffect) to manage state and perform side effects. The fetched data is stored in the podcast state variable, and the loading state variable is used to track whether the data is still loading.

The component also defines a function called handleSelectedShow to fetch the details of a selected show when called with its id. The show details are stored in the selectedShow state variable.

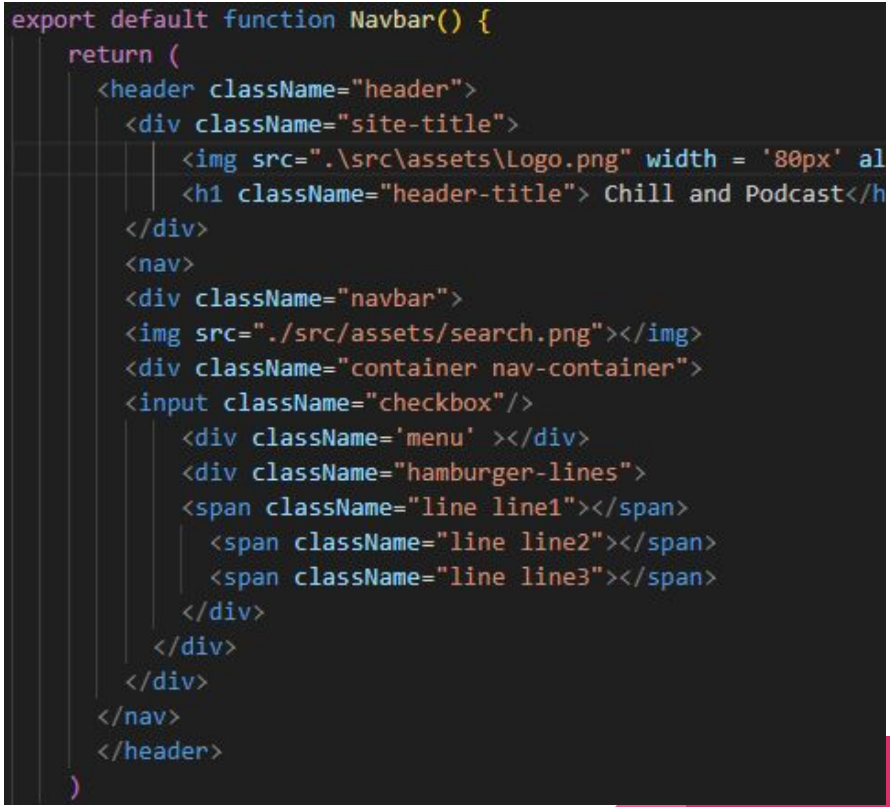


```
function App() {
  const [podcast, setPodcast] = useState([]);
  const [loading, setLoading] = useState(true);
  const [selectedShow, setSelectedShow] = useState(null);


  useEffect(() => {
    fetch('https://podcast-api.netlify.app/shows')
      .then((res) => res.json())
      .then((data) => {
        setPodcast(data);
        setLoading(false);
      })
      .catch((error) => {
        console.error('Error loading podcast data:', error);
        setLoading(false);
      });
  }, []);
```

Navbar function

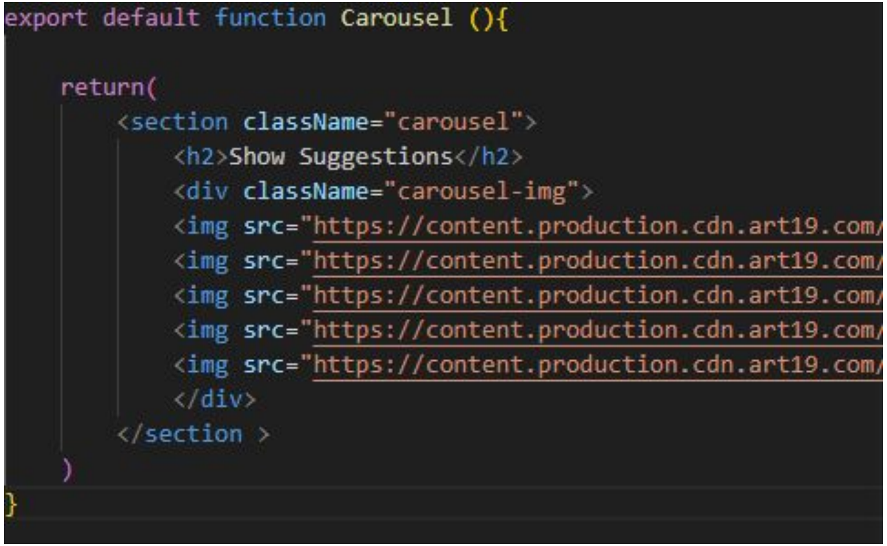
The function contains Search Icons, Logo and Classes



```
export default function Navbar() {
    return (
      <header className="header">
        <div className="site-title">
            <img src=".\src\assets\Logo.png" width = '80px' al
            <h1 className="header-title"> Chill and Podcast</h
        </div>
        <nav>
        <div className="navbar">
        <img src="./src/assets/search.png"></img>
        <div className="container nav-container">
        <input className="checkbox"/>
            <div className='menu' ></div>
            <div className="hamburger-lines">
            <span className="line line1"></span>
              <span className="line line2"></span>
              <span className="line line3"></span>
            </div>
          </div>
        </div>
      </nav>
      </header>
    )
```

Carousel Function

Carousel displays Show suggestions



```
export default function Carousel (){

    return(
        <section className="carousel">
            <h2>Show Suggestions</h2>
            <div className="carousel-img">
            <img src="https://content.production.cdn.art19.com/
            <img src="https://content.production.cdn.art19.com/
            <img src="https://content.production.cdn.art19.com/
            <img src="https://content.production.cdn.art19.com/
            <img src="https://content.production.cdn.art19.com/
            </div>
        </section >
    )
}
```

Card Function

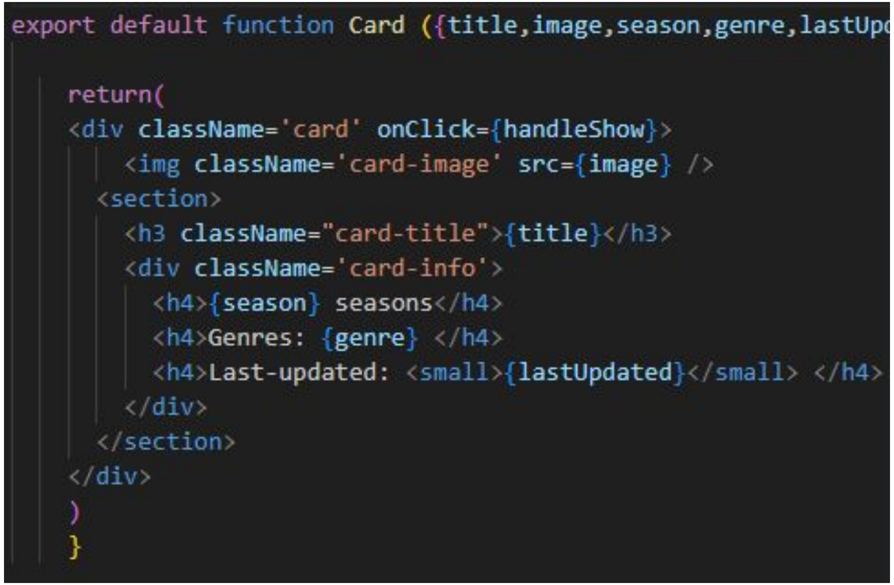
Card function displays

A picture associated with the podcast

Podcast Genre

Number of Seasons

Podcast Title

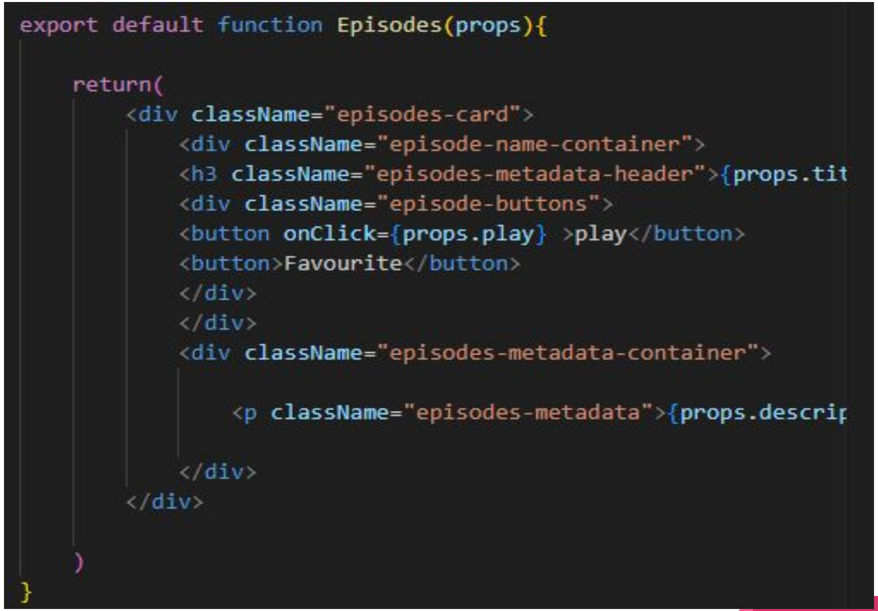


```
export default function Card ({title,image,season,genre,lastUpd

    return(
    <div className='card' onClick={handleShow}>
        <img className='card-image' src={image} />
      <section>
        <h3 className="card-title">{title}</h3>
        <div className='card-info'>
          <h4>{season} seasons</h4>
          <h4>Genres: {genre} </h4>
          <h4>Last-updated: <small>{lastUpdated}</small> </h4>
        </div>
      </section>
    </div>
    )
    }
```

Episodes Function

When you click on the selected podcast it displays the amount of episodes available and also the play and favourites button



```
export default function Episodes(props){

    return(
        <div className="episodes-card">
            <div className="episode-name-container">
            <h3 className="episodes-metadata-header">{props.tit
            <div className="episode-buttons">
            <button onClick={props.play} >play</button>
            <button>Favourite</button>
            </div>
            </div>
            <div className="episodes-metadata-container">

                <p className="episodes-metadata">{props.descrip

            </div>
        </div>

    )
}
```