

Netflix-Clone

• MOHAN RAAJI B

Introduction

- This project replicates Netflix's core features, including a dynamic home page, user authentication, and movie trailer playback, showcasing the integration of modern web development technologies.
- To create a scalable, user-friendly platform for streaming content.



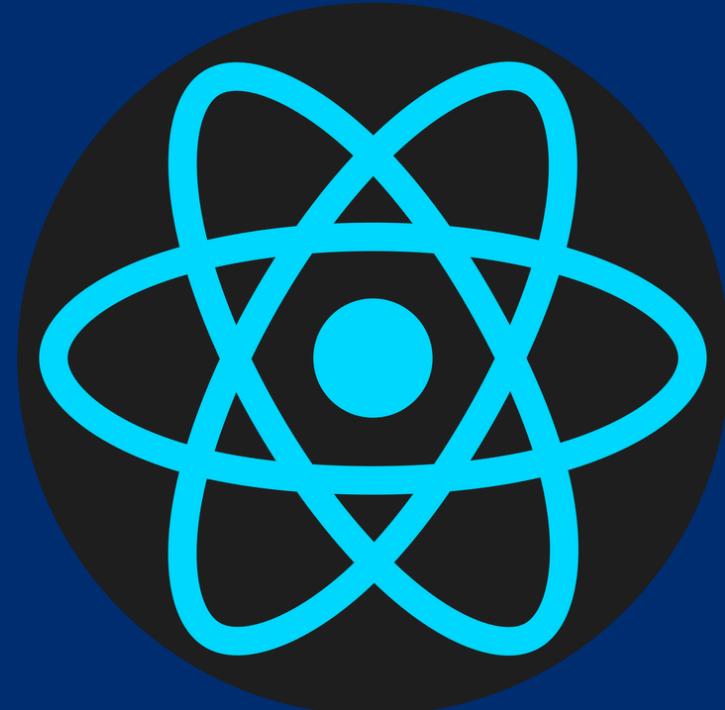
Problem Statement

- Building a platform that combines seamless UI/UX design, secure user authentication, and dynamic content delivery.
- Highlight challenges like API integration, responsive design, and efficient state management.



Technology Stack

- 01 **Frontend:** React, React Router, CSS, Vite
- 02 **Backend** Firebase Firestore, Firebase Authentication
- 03 **API Integration:** TMDB API for movie data



Features



- **Home Page:** Dynamic movie categories (e.g., Blockbusters, Top Hits).
- **Login/Register:** Secure user authentication using Firebase.
- **Player:** Embedded YouTube trailers fetched dynamically.
- **Responsive Design:** Optimized for devices of all sizes.

System Architecture

- Frontend components dynamically fetch and display data.
- Firebase Firestore stores user data and supports authentication.
- TMDB API provides real-time movie data.

```
render() {
  return (
    <React.Fragment>
      <div className="py-5">
        <div className="container">
          <Title name="our" title="prod">
            <div className="row">
              <ProductConsumer>
                {(value) => {
                  console.log(value);
                }}
              </ProductConsumer>
            </div>
          </Title>
        </div>
      </div>
    </React.Fragment>
  );
}
```



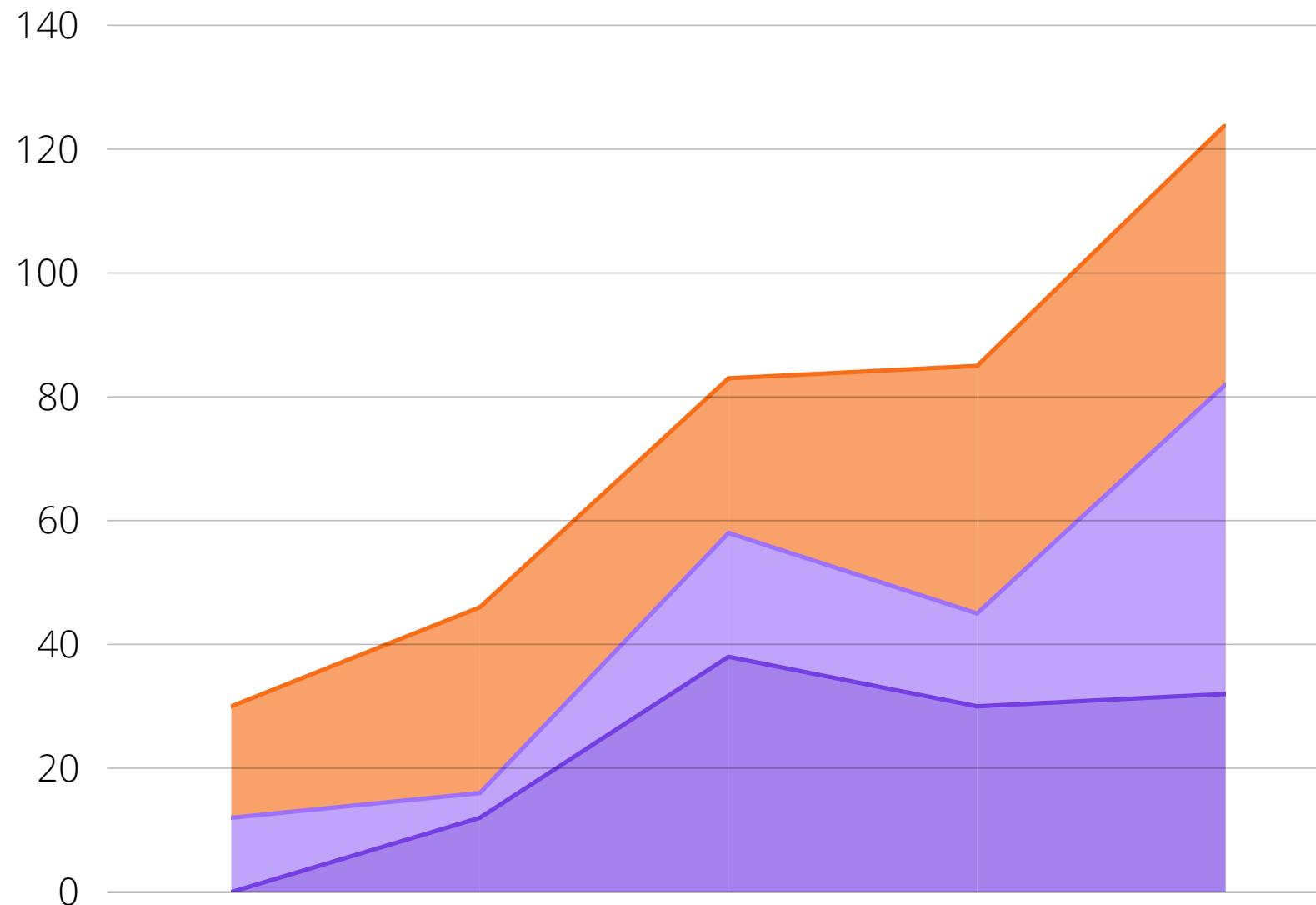
Implementation



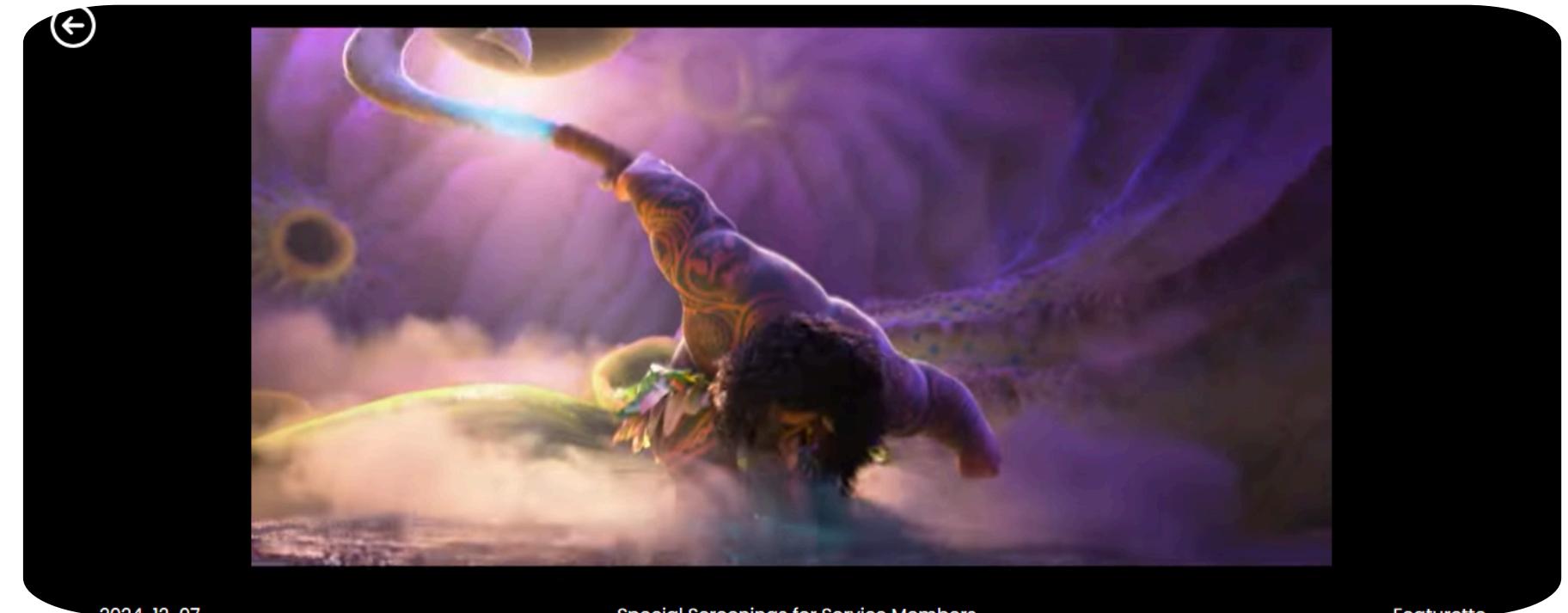
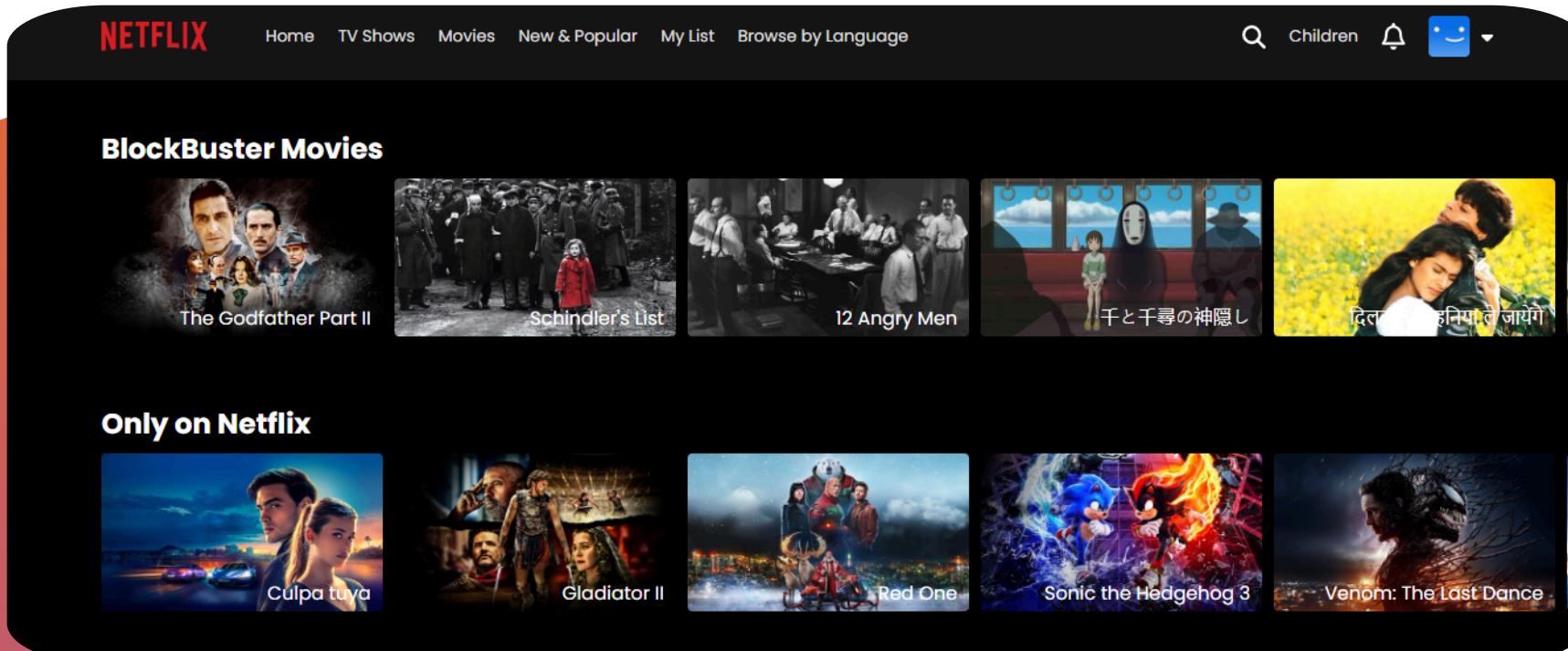
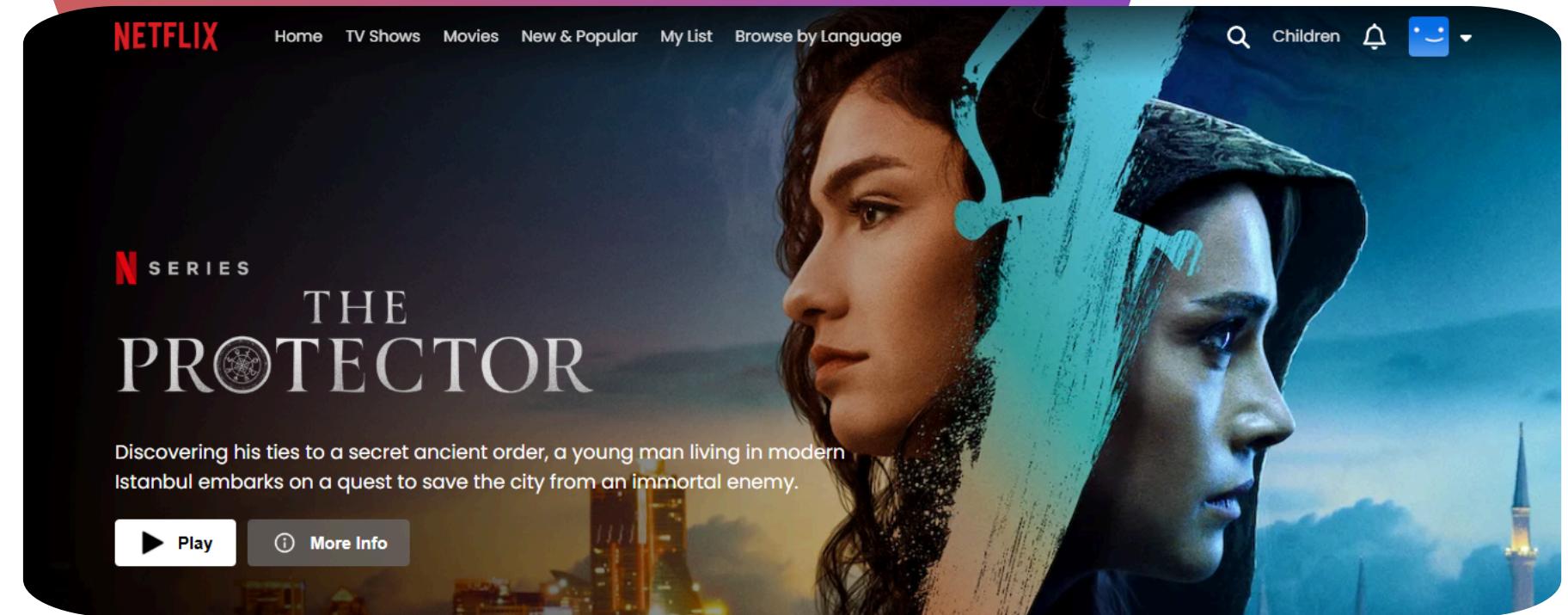
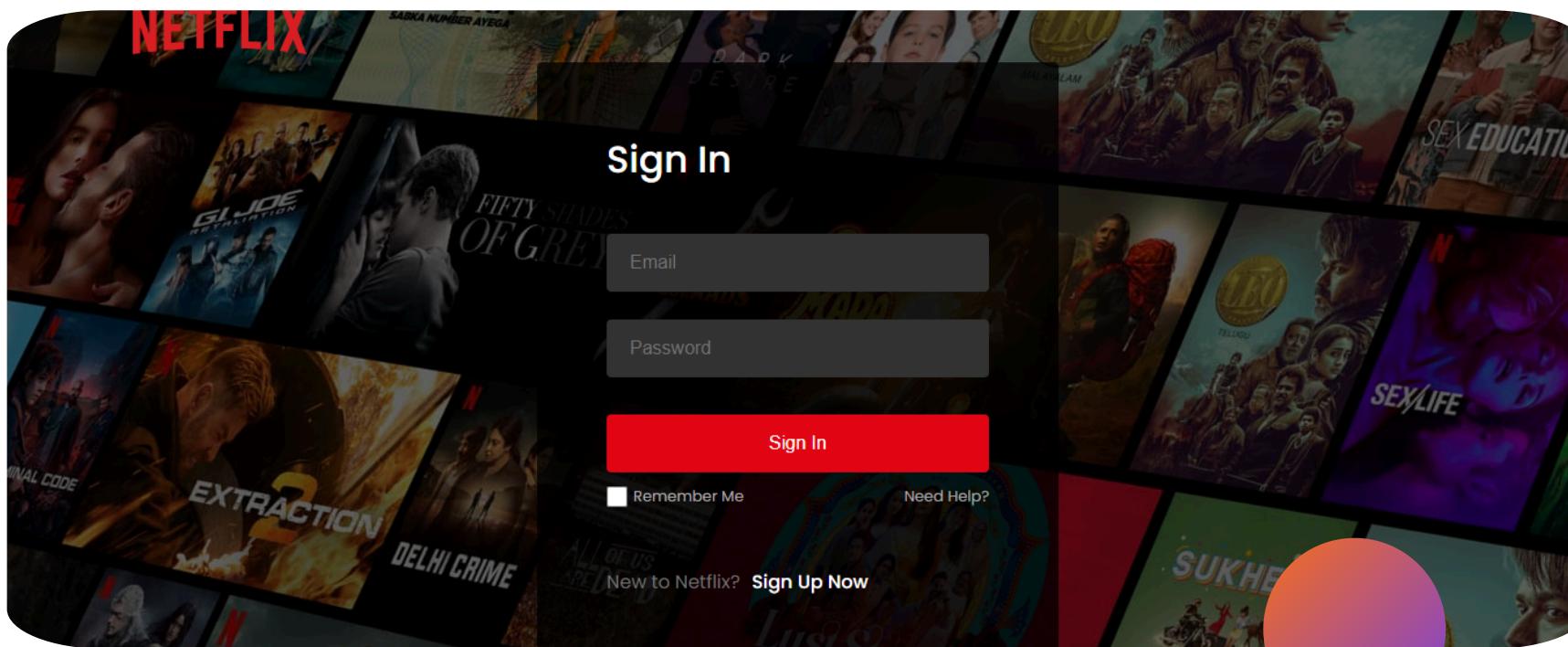
- **Home Page:** Modular components like Navbar, Hero, and TitleCards for rendering content.
- **Login/Register:** Unified page with form validation and Firebase integration.
- **Player:** Fetches and plays trailers using TMDB API and embeds YouTube videos.

Challenges

- **API Data Handling:** Added fallback mechanisms and error handling.
- **Responsive Design:** Used media queries and CSS flex/grid for consistency.
- **Dynamic Rendering:** Managed loading states and asynchronous operations effectively.



Demo



Thank You

- MOHAN RAAJI B