Assignment Category: Camellia

Chill Gamer: A Game Review Application

Project Theme

Build **Chill Gamer**, a user-friendly **game review application**. The goal of this project is to design a platform where users can explore and share game reviews. You need to develop key features such as user authentication and review management to enhance the user experience. The application should have a clean and responsive UI, focusing on simplicity and functionality to provide a ”chill” experience.

Key Rules:

- **GitHub Commits:**
    - Include a minimum of 15 notable GitHub commits on the client side.
    - Include a minimum of 8 notable GitHub commits on the server side
- **Readme.md:** Add a meaningful readme.md file with the name of your website and a live site URL. Include a minimum of five bullet points to feature your website.
- **Responsiveness:** Make it responsive for all devices, including mobile, tablet, and desktop views.
- **Environment Variables:** The Environment variable hides the Firebase config keys and Mongodb credentials.
- **Lorem Text:** Don’t use any Lorem ipsum text; you can not use the default alert to show any error or success message.
- **Unique Design:** Create a unique Design but remember, your website idea shouldn’t be similar to any projects you’ve done before or to any examples in our modules or conceptual sessions.

★ You can also look for free resources on blogs to help with your website.

- **Host your Application:** You can choose deployment systems like Netlify, Surge, and Firebase for client-side hosting and vercel for server-side hosting. As you develop a single-page application
    - Ensure the page doesn't throw any error when reloading from any routes.
    - Add your domain for authorization to Firebase if you use Netlify / surge
    - Logged in User must not redirect to Login on reloading any private route

---

Main Requirements:

- ### Layout Structure

1. **Navbar: The** Navbar will have the-
    - Website name/logo
    - Home
    - All Reviews
    - Add Review **(Private/Protected Route)**
    - My Reviews **(Private/Protected Route)**
    - Game WatchList **(Private/Protected Route)**
    - Login/Register **(Conditional)**:
    - If the user is not logged in, show:
        - Login: Redirects to the login page.
        - Register: Redirects to the register page.
    - If the user is logged in, show:
        - User Avatar (photoURL): Display the user's profile picture.On hovering, display the user's `displayName` in a tooltip or dropdown.
        - Log Out: Logs the user out and redirects to the homepage.

2. **Main Section:** Main Section will show different pages based on routes.
3. **Footer:** A Footer with all relevant information and eye-catching design.

- ## Home Page:

1. **Home Page:** Besides the navbar and footer the **Home Page** will contain a banner/Slider, a Highest Rated Game Section, 2 Extra Sections,
   🎯 **Make sure to keep the navbar and footer on all the pages except the 404 page.**

2. **Banner:** Add a slider (you can use any type of slider/carousel) with a minimum of 3 slides and meaningful information

3. **Highest Rated Game Section:** You need to show at least 6 cards of Highest Rated Game data based on their rating. Here you will show the data from the database. Which data you want to show on the card is up to you. Each card will contain a "Explore Details" button.

   For showing 6 data you can use the limit operator of MongoDB

4. **Extra Section:** Add 2 relevant and meaningful extra sections that are mentioned above on the Home page.

Authentication

5. **Login Page:** When you click the login button on the navbar it redirects to the login page. You have to use a password and email-based authentication to log in. The login page will have-
   a. Email
   b. Password
   c. Google login, GitHub - implement any of one
   d. A link that will redirect to the Register page

   🎯**Here the email and password should match with the registered email and password. If it doesn't match, show an error. You can show an error by using toast/sweet alert if you want.**

6. **Register Page:** You have to use a password and email-based authentication to register. The Register page will have the following -
    - Name
    - Email
    - photoURL
    - password
    - A Link that will redirect to the login page
    - ★ For password verification you need to follow this -
        - Must have an Uppercase letter in the password
        - Must have a Lowercase letter in the password
        - Length must be at least 6 character
    - ★ If any of this isn't fulfilled it will show an error/toast
    - ★ After successful login or Register you need to show toast/sweet alert

🎯**Don't implement email verification or forget password method as it will inconvenience the examiner. If you want, you can add these after receiving the assignment result.**

---

7. **Add Review Page (/addReview):**
   Create an **Add New Review** page where users can submit reviews for games. The form will include:

    - Game Cover Image/Thumbnail (a URL for the game cover)
    - Game Title/ Name (string)
    - Review Description (text): A detailed review of the game.
    - Rating (number): Allow users to provide a rating (e.g., 1-5 or 1-10).
    - Publishing year: (Ex: 2021, 2024)
    - Genres (dropdown): Users can select one (e.g., Action, RPG, Adventure).
    - User Email (Read Only): Pre-filled with the logged-in user's email address.

- User Name (Read Only): Pre-filled with the logged-in user's name.
- **Submit Button**: On clicking, the review data will be stored in the database, and a success message will be shown using **toast** or **sweet alert**.

This is a **private/protected route**, and only logged-in users can access it. Redirect non-logged-in users to the login page.

8. **Review Details page (/review/:id)**

The **Review Details Page** will display all the information stored in the database for a specific review.

- Show the game cover image, title, review description, rating, genre, reviewer's name, and email.
- Include a "Add to WatchList" button for logged-in users to like the review.
- After Clicking on "Add to WatchList" the review data will be stored on the database(with logged-in email and username) in the **watchlist collection** with the user email and username.

9. **All Reviews(/reviews)**

Create an All Reviews page to display all the reviews added by users.

- Fetch all the Reviews data from the database and display it in a card format.
- Show 3-4 properties you want to show and an "Explore Details" button linking to the review details page.

10. **My Reviews(/myReviews):**

A **private/protected route** where a user can see all the Reviews he/she has added to the database. here a user can only see his/her added data, but he/she can not access other's data.

- Display reviews in a table format, showing:
    - 3-4 data as you want

    - **Update** button
    - **Delete** button

Update Feature:

- Clicking the "Update" button will redirect to an **Update Review Page** (`/updateReview/:id`) or open a modal where users can edit their review.

Delete Feature:

- Clicking the "Delete" button will remove the review after a confirmation prompt.

11. **Update Review Page (/updateReview/:id)**
    Create an **Update Review Page** where users can update their previously submitted reviews.

    - The form should include:
        - All fields from the "Add Review Page" (game title, description, rating, genres, etc.).
        - Pre-fill the fields with existing data from the database.
        - **User Email** and **User Name** should remain read-only.
    - Clicking the "Update" button will save changes and display a success message.

Optional: Use a modal instead of a separate page for updating reviews.

12. Game Watchlist Page (/myWatchlist)

The **Game Watchlist Page** is a **private/protected route** where logged-in users can manage a list of games they have added to the Watchlist from the Review Details page. You need to show all the data in table format. here a user can only see his/her added data, but he/she can not access other's data. Which data you want to show is your choice.

13. Other Requirements

- **404 page:** Add a 404 page/Not Found Page
- **Loading Spinner:** Show a loading spinner when the data is in a loading state.

Challenges

- Implement a dark/light theme toggle for the home page.
- Explore this package and implement at least 2-
    - Lottie React
    - React-simple-typewriter
    - React Awesome reveal
    - React-tooltip
- At the top of the **All Reviews** page, there will be a dropdown button where you will implement sort functionality based on “Rating” and “year”. You can sort by ascending or descending or, it’s up to you.
- At the top of the “All Reviews” page, Add a dropdown menu where you will implement filter functionality based on the Genres field the game.

What to submit:

- Your client-side code GitHub repository
- Your server-side code GitHub repository
- Your live website link