Brain Boosting Game Web App for Kids

Final Report: CS526 (Advanced Web Programming) Project

Supervisor: Prof. Pragati Dharmale

Prepared by: Aderonke Babatunde (136640), Sruthi Puthiyandy (162489),

Introduction

The rapid advancement of digital entertainment has significantly reshaped how children engage with games. While modern games offer impressive graphics and fast-paced interaction, they often lack the developmental value associated with traditional, cognitive-based games. In response to this shift, our project — Brain Boosting Game Web App for Kids — is a web-based application designed to rekindle interest in classic games that encourage critical thinking, memory, and decision-making among Gen-Z children.

Through this platform, players can interact with vibrant, animation-rich interfaces, track their scores, manage their profiles, and enjoy a mobile-friendly, responsive design. The application not only entertains but also supports cognitive development in children by enhancing skills such as memory retention, pattern recognition, strategic thinking, and motor coordination.

Objective and Functionality

The primary objective of the Brain Boosting Game Web App for Kids is to create an engaging, educational platform that reintroduces classic cognitive games in a digital format tailored for the Gen-Z generation while also demonstrate ability to apply in-class knowledge to a real-world scenario of building a full stack web application.

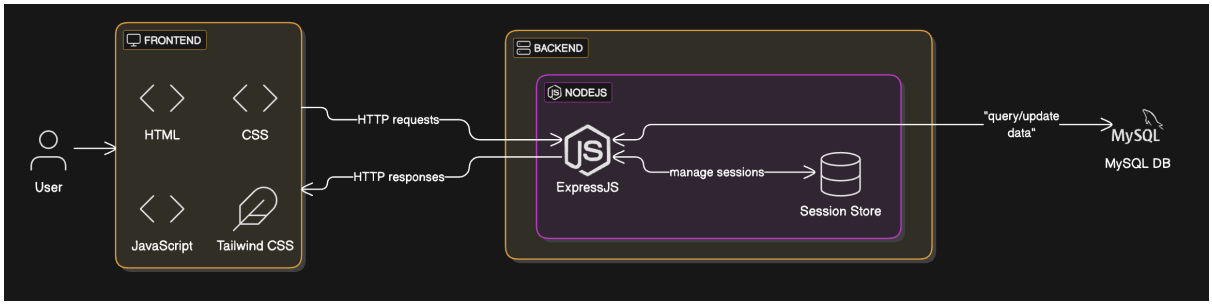
This web application includes a comprehensive set of functionalities both on the frontend (client-facing) and backend (server-side), providing a complete user experience. Key functionalities include:

- User registration with password confirmation and username uniqueness validation.
- Login authentication using session-based storage.

- Integration of a themed game interface styled with advanced animations.
- Profile dropdown with the ability to view history and delete account.
- Use of real-time validation feedback for errors such as password mismatch or existing usernames.
- Responsive and themed landing page
- Classic games built with animations and interactive UI
- Backend integration with a MySQL database for user data persistence.

System Architecture

The application is built on a classic client-server architecture model. The architecture is based on a client-server model. The frontend, developed in HTML, CSS, JavaScript, and React, interacts with the backend via RESTful APIs. The backend, built with Node.js and connected to an MSSQL Server database, handles logic, user data, and persistence.



Tools and Frameworks

Frontend:

- HTML5 & CSS3: Markup and styling of the interface
- JavaScript: Client-side scripting for all interactivity
- Tailwind CSS: Utility-first CSS framework for rapid UI development
- Framer Motion for rich animations

Backend:

- Node.js & Express: Server-side environment and web framework
- MySQL: Relational database for user data management
- REST API for frontend-backend interaction

Development Tools:

- Visual Studio Code: Code editor
- GitHub: Version control and collaboration

Team Roles

The project was completed by a two-member team and both members contributed equally to the development of both frontend and backend functionalities.

- Frontend: Designing UI components, implementing Tailwind CSS styling, JavaScript functionality, and animations.
- Backend: Creating REST API endpoints, handling database operations, and server configuration.
- Testing and Debugging: Shared responsibility for identifying and fixing bugs and

Some key contributions include:

- Sruthi focused on creating the DB, implementing the profile management endpoints and handling the game logic and animations.
- Aderonke handled design and implementation of the landing page and other pages, form validation and backend to frontend connection.

Challenges, Learnings, and Future Improvements

Challenges:

- Handling asynchronous JavaScript and consistent error feedback.
- Managing session storage across multiple JavaScript files.

- Handling session management and secure authentication.
- Debugging API endpoint failures.
- Coordinating real-time validations between frontend and backend.

Learnings:

- Deepened understanding of REST API integration and asynchronous programming.
- Improved skills in responsive design using Tailwind CSS.
- Experience with full-stack architecture and team collaboration workflows.
- Improved skills in debugging and testing asynchronous JavaScript functions.

Future Improvements:

- Add OAuth (Google/GitHub) authentication.
- Improve backend functionality such as game analytics and user history tracking.
- Deploy on a cloud platform.

Conclusion

By building this full-stack application, we have applied and expanded our technical knowledge across frontend and backend technologies, gaining hands-on experience in real-world software development. This report outlines the system's architecture, the tools and frameworks utilized, the division of responsibilities, challenges encountered, key learnings, and areas for future improvement.

Additionally, this full-stack project provided an innovative platform to revive meaningful classic games for modern children. The intuitive interface, impressive use of animations with Framer Motion, and robust backend architecture collectively deliver a polished and engaging experience. The web app also exemplifies the effective use of JavaScript for both logic and interactivity.