**GenAI Interactive Learning Games - Project Documentation**

**Overview**

A GenAi-React-based interactive learning platform built with Vite, featuring gamified educational content. Key highlights:

Modern Stack: React 19 + Vite (with HMR), ESLint for code quality

Engaging UI: Includes animations (`framer-motion`), carousels (`react-slick`), and confetti effects (`react-confetti`)

API Integration: Axios for backend communication (proxied via Vite)

Responsive Design: Uses `@react-hook/window-size` for adaptive layouts

Client-Side Routing: React Router DOM for seamless navigation

**Key Components**

React 19 : Core framework for UI components and state management

Vite: Fast build tool with HMR and proxying (`/api` → `localhost:5000`)

React Router DOM: Client-side navigation and routing

ESLint: Enforces code quality with React-specific rules (hooks, refresh)

Framer Motion: Adds animations for interactive learning elements

React Slick: Carousel components for content slides

Axios: Handles API calls to backend services

**Project Structure**

public/ # Static assets (e.g., vite.svg)

src/

pages/ # Route components

Home.jsx # Landing page

Game.jsx # Dynamic game page

main.jsx # Router setup

index.html # Entry HTML (with Slick Carousel CSS)

vite.config.js # Vite config (proxy, plugins)

eslint.config.js # ESLint rules (React hooks, browser globals)

package.json # Dependencies + scripts

.gitignore # Ignores logs, node\_modules, IDE files

**Installation**

1. Clone the repository

2. Install dependencies:

bash

npm install

3. Start the dev server:

bash

npm run dev

**Configuration**

Vite Proxy Setup

Routes `/api` requests to `http://localhost:5000` (backend):

js

vite.config.js

server: {

proxy: {

'/api': {

target: 'http://localhost:5000',

rewrite: (path) => path.replace(/^\/api/, '')

}

}

}

**ESLint Rules**

Enforces React best practices (hooks, refresh) and browser-compatible JS:

```js

eslint.config.js

rules: {

'react-refresh/only-export-components': ['warn', { allowConstantExport: true }],

'no-unused-vars': ['error', { varsIgnorePattern: '^[A-Z\_]' }]

}

```

**Routing**

The app uses \*\*React Router DOM v7\*\* for client-side navigation, enabling multi-page interactions without full reloads.

Key Features

Dynamic Routes: Load components based on URLs (e.g., `/game`, `/quiz`)

Programmatic Navigation: Navigate between pages via hooks (`useNavigate`)

Nested Routes: Support for complex layouts (future implementation)

**Implementation**

1. Router Setup (in `src/main.jsx`):

```jsx

import { createBrowserRouter, RouterProvider } from 'react-router-dom';

import Home from './pages/Home';

import Game from './pages/Game';

const router = createBrowserRouter([

{ path: '/', element: <Home /> },

{ path: '/game/:id', element: <Game /> },

]);

ReactDOM.createRoot(document.getElementById('root')).render(

<RouterProvider router={router} />

);

```

2. Navigation Methods:

```jsx

import { Link } from 'react-router-dom';

<Link to="/game/123">Start Game</Link>

```

Hook (Imperative):

```jsx

import { useNavigate } from 'react-router-dom';

const navigate = useNavigate();

navigate('/game/123');

```

3. Accessing Route Parameters:

```jsx

import { useParams } from 'react-router-dom';

const { id } = useParams(); // Gets `123` from `/game/123`

```

* No Nested Routes
* Open the front end and back end part in windows of vs code separately
* For quiz http://localhost:5713/quiz/dashboard

**Features**

Gamification: Confetti effects (`canvas-confetti`), emoji picker (`emoji-picker-react`)

Routing: Multi-page navigation with React Router DOM

Responsive: Adapts to window size changes (`@react-hook/window-size`)

UI Enhancements: Slick carousels, motion animations

**Dependencies**

Core

- `react@19`, `react-dom@19`, `vite`, `eslint`

Routing=

- `react-router-dom@7.4.1`

UI & Utilities

- `framer-motion`, `react-slick`, `react-confetti`, `emoji-picker-react`

API & State

- `axios`, `js-cookie`, `react-cookie`, `uuid`

Dev Tools

- `@vitejs/plugin-react`, `eslint-plugin-react-hooks`, `globals`

**Quick Start**

```bash

npm run dev # Start dev server (http://localhost:3000)

npm run lint # Check code quality

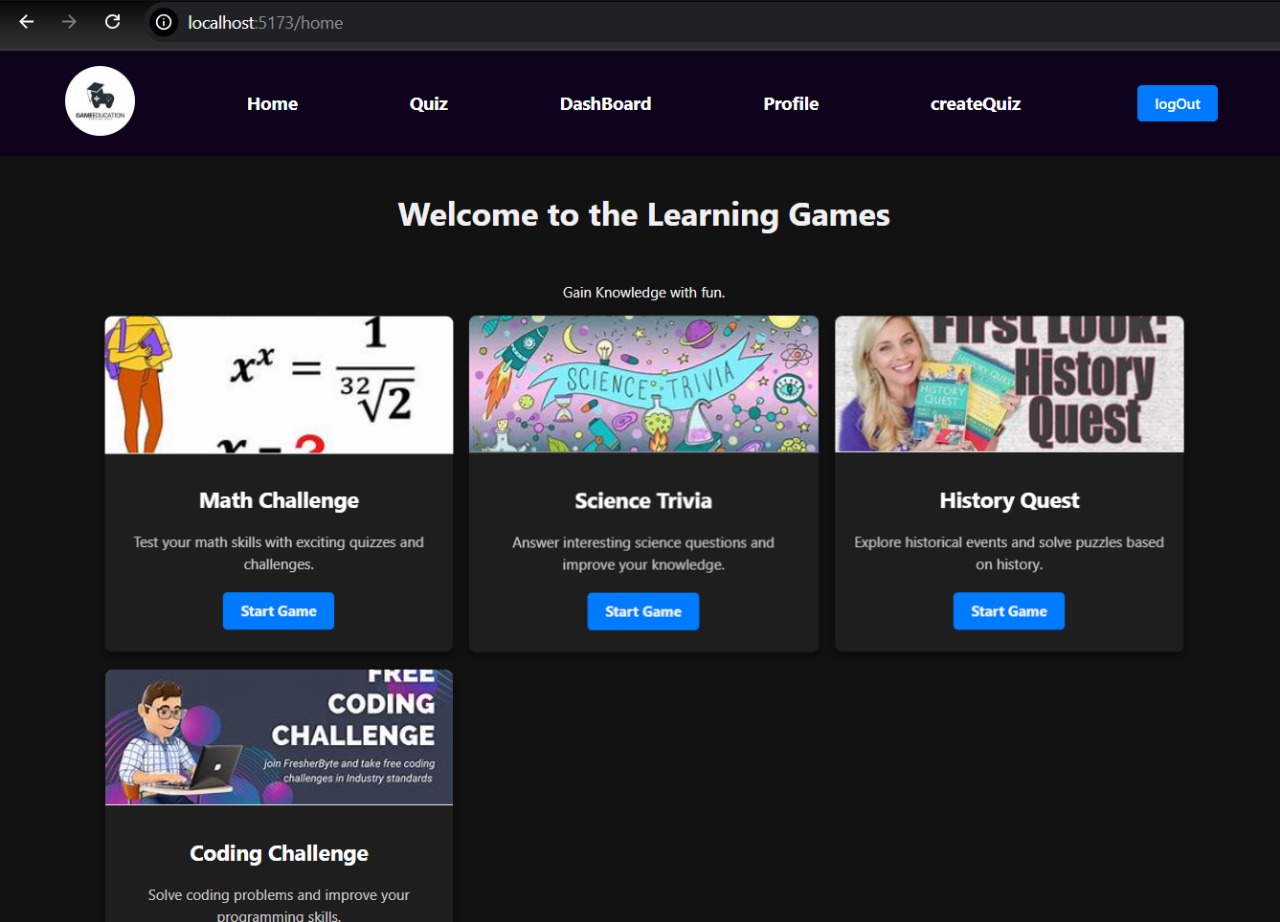
npm run build # Production build

**Credintials:**

**mail:** [nencheppanuga123@gmail.com](mailto:nencheppanuga123@gmail.com)

**Password**: password

**Images/ Screen shots: **

****

