Developer Documentation for TaskBuddy

Project Name: TaskBuddy
Version: 1.0
Date: 04-09-2024
Author: Rushikesh Gajanan Suryawanshi
Description: TaskBuddy is a web application designed for efficient daily task management. It allows users to sign up, log in, and manage their tasks with various features like marking tasks as important, tracking task completion, and more.

1. Architecture

1.1 Frontend

- **Framework:** React.js
- **UI Components:** Custom components built for task management, including task cards, forms, and a sidebar for navigation.
- **Routing:** React Router is used for navigation between different pages like Home, Important Tasks, Completed Tasks, etc.
- **State Management:** Managed using React's built-in state management features (useState, useEffect, etc.).
- **Styling:** Tailwind CSS are used for styling the components.

1.2 Backend

- **Framework:** Express.js
- **Database:** MySQL
- **API Structure:** RESTful API endpoints for handling user authentication (signup, login) and task management (CRUD operations).
- **Authentication:** JWT (JSON Web Tokens) for session management.
- **Data Validation:** Handled using middleware in Express.js to ensure secure and accurate data handling.

2. Key Features

2.1 User Authentication

- **Sign-Up:** Users can sign up by providing a username, email, and password (minimum 4 characters).
- **Login:** Users can log in using their registered email and password. Error messages are displayed for incorrect credentials.

2.2 Task Management

- **Create Task:** Users can create a new task by providing a title and description.
- **Update Task:** Users can update the title and description of existing tasks.

- **Delete Task:** Users can delete tasks as needed.
- **Mark as Important:** Users can mark tasks as important, which starts a 3-hour timer displayed on the task card.
- **Mark as Complete/Incomplete:** Users can toggle the completion status of tasks.

2.3 Pages and Sections

- **Home Page:** Displays all tasks by default.
- **Important Tasks Page:** Displays tasks marked as important.
- **Completed Tasks Page:** Displays tasks marked as completed.
- **Incomplete Tasks Page:** Displays tasks that are incomplete.
- **Sidebar Navigation:** Allows quick access to different task categories and logout functionality.

3. Code Structure

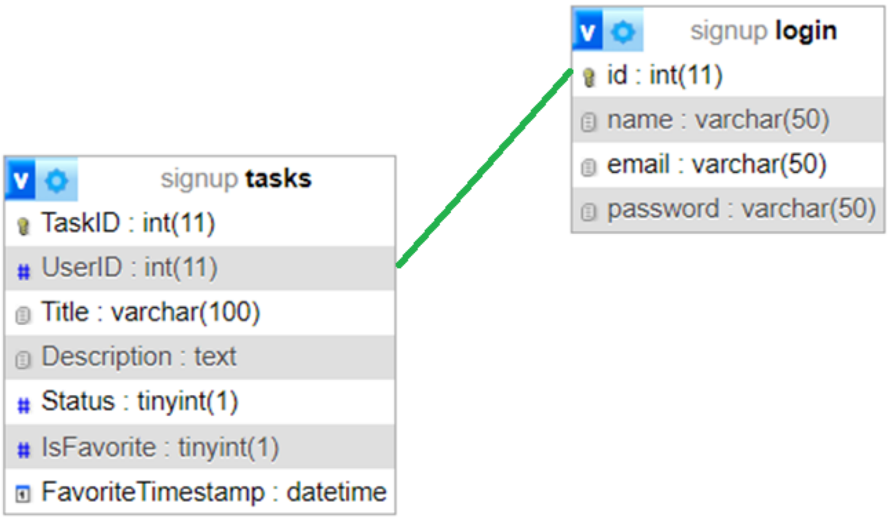
3.1 Frontend

- **Components:**
    - **Card.jsx:** Displays individual task details with options to update, delete, mark as important, or toggle completion status.
    - **Inputdata.jsx:** Handles the input of task details for creating or updating tasks.
    - **Sidebar.jsx:** Manages navigation between different sections.
- **Pages:**
    - **Alltask.jsx:** The main page where all tasks are displayed.
    - **ImportantTasks.jsx:** Displays tasks marked as important.
    - **CompletedTasks.jsx:** Displays completed tasks.
    - **IncompleteTasks.jsx:** Displays incomplete tasks.

3.2 Backend

- **Routes:**
    - **/api/auth/signup:** Handles user registration.
    - **/api/auth/signin:** Handles user login.
    - **/api/tasks/:userId:** CRUD operations for managing tasks (Create, Read, Update, Delete).
- **Middleware:**
    - **Auth Middleware:** Verifies JWT tokens for protected routes.
    - **Validation Middleware:** Ensures data validity during signup and task management.

- **Database Schema:**
  - **Users Table:** Stores user information including username, email, password.
  - **Tasks Table:** Stores task information including title, description, status (complete/incomplete), important flag, userId, and timestamps.



4. Deployment

4.21Environment Variables

- **JWT_SECRET:** Secret key used for signing JWT tokens.
- **DB_HOST, DB_USER, DB_DATABASE:** Database connection details.
- **PORT:** Port number on which the backend server runs.