Task Manager App - Software Requirements Specification (SRS)

============================================================

**1. Introduction**

**1.1 Purpose**  
The Task Manager App is designed to help individuals and families manage and track tasks efficiently. It allows users to create, assign, and monitor tasks, manage family members, and view progress through dashboards. This document provides the complete software requirements specification for the Task Manager App.

**1.2 Scope**

* Web-based application with frontend built in React (JSX + Tailwind) and backend using Node.js with Express.
* Database managed using Neon (PostgreSQL).
* Features include user authentication (JWT), task management, family management, dashboards, and progress tracking.

**1.3 Definitions, Acronyms, Abbreviations**

* JWT: JSON Web Token for authentication.
* API: Application Programming Interface.
* Neon: Cloud-hosted PostgreSQL database service.

**1.4 References**

* React Documentation: <https://reactjs.org/docs/getting-started.html>
* Node.js Documentation: <https://nodejs.org/en/docs/>
* Express Documentation: <https://expressjs.com/>
* Neon Documentation: <https://neon.tech/docs/>

**2. Overall Description**

**2.1 Product Perspective**  
The Task Manager App is a standalone web application designed to assist users in managing tasks and collaborating with family members.

**2.2 Product Functions**

* User Signup and Login with JWT authentication.
* Task CRUD operations (Create, Read, Update, Delete).
* Family management (create family, join family via invitation code).
* Dashboard for task completion tracking and family progress visualization.
* Responsive UI compatible with desktop and mobile.

**2.3 User Classes and Characteristics**

* Regular User: Can create tasks, mark tasks completed, join a family.
* Family Leader: Can create families, assign tasks to members, view family dashboard.

**2.4 Operating Environment**

* Frontend: React + Tailwind CSS.
* Backend: Node.js + Express.
* Database: Neon PostgreSQL.
* Browser support: Chrome, Firefox, Edge, Safari.

**2.5 Design and Implementation Constraints**

* Must use JWT for secure authentication.
* All sensitive data must be stored in the Neon database.
* RESTful API design for backend endpoints.

**3. System Features & Requirements**

**3.1 Functional Requirements**

**3.1.1 User Authentication**

* Users can sign up and log in.
* JWT token verification for secure API access.

**3.1.2 Task Management**

* Create, edit, delete, and assign tasks.
* Task prioritization (Low, Medium, High).
* Mark tasks as completed or pending.

**3.1.3 Family Management**

* Create a new family.
* Join an existing family using invitation code.
* Family leader can assign tasks to members.

**3.1.4 Dashboard & Progress Tracking**

* View tasks with priority and status.
* Visual representation of family member completion rates.
* Task distribution among family members.

**3.2 Non-Functional Requirements**

* **Security:** JWT authentication, HTTPS support.
* **Performance:** Fast API responses, optimized queries.
* **Usability:** Intuitive UI with responsive design.
* **Reliability:** Handle errors gracefully, proper logging.

**4. External Interface Requirements**

**4.1 User Interfaces**

* Forms for login, signup, task creation, and family management.
* Modals for creating and joining families.
* Dashboard visualization using charts.

**4.2 Hardware Interfaces**

* Web browser access; compatible with desktops, laptops, and mobile devices.

**4.3 Software Interfaces**

* REST API endpoints for tasks, families, and authentication.
* Neon PostgreSQL database access through Node.js backend.

**5. Other Requirements**

**5.1 Backup & Recovery**

* Daily backup of database to prevent data loss.

**5.2 Logging & Error Handling**

* Centralized logging for API requests and errors.
* Proper error messages for frontend display.

**5.3 Future Enhancements**

* AI-based task suggestions.
* Push notifications and email reminders.
* Mobile app version.