**Smart Task Manager: A Web-Based To-Do List Application**

**Abstract**

This project presents a web-based application built using modern JavaScript technologies. It includes multiple static pages such as login, registration, home, and dashboard, and simulates backend operations using a local JSON file (db.json). The objective is to demonstrate the design, development, and deployment of a modular and responsive frontend interface.

**3. Table of Contents**

* Abstract
* Introduction
* Objectives
* System Requirements
* System Design
* Implementation
* Testing
* Conclusion
* Future Enhancements
* References
* Appendix

**4. Introduction**

The application demonstrates the fundamental structure of a single-page or multi-page web application. Built using HTML, CSS, and JavaScript (possibly React), it includes routing between pages and a mocked backend for simulating dynamic content.

**5. Objectives**

* Build a responsive and interactive frontend interface.
* Implement user flows such as login, registration, and dashboard viewing.
* Simulate backend functionality using local mock data (db.json).
* Organize the project using best practices for code and folder structure.

**6. System Requirements**

**Hardware:**

* Minimum 4 GB RAM
* Intel i3 or above processor

**Software:**

* Node.js (v14 or above)
* Code Editor (VS Code)
* Browser (Chrome, Firefox)

**7. System Design**

**Folder Structure:**

project/

├── data/

│ └── db.json (mock database)

├── public/

│ ├── \*.html (static views)

│ └── images/

├── src/

│ ├── index.js (main logic)

│ └── index.css (styles)

├── package.json (project metadata)

**Design Approach:**

* MVC-inspired separation (data, logic, view)
* Mocked RESTful data using JSON
* Responsive layout with modular CSS

**8. Implementation**

* index.js: Initializes the application and handles event-based logic
* index.css: Stylesheets for the UI
* HTML files: Serve individual sections of the website
* db.json: Simulates server-side data (can be accessed via json-server)

**9. Testing**

Manual testing was conducted:

* Verified page navigation and link integrity
* Validated form inputs for login and registration
* Ensured responsiveness on different screen sizes

**10. Conclusion**

The project successfully demonstrates the development of a client-side application with simulated data services. It lays a foundation for integrating real backend services or deploying to production.

**11. Future Enhancements**

* Connect to a real database with Express.js or Firebase
* Add authentication and session handling
* Integrate chart libraries for data visualization on the dashboard

**12. References**

* [Node.js Documentation](https://nodejs.org/)
* [React Documentation](https://reactjs.org/)
* [json-server GitHub](https://github.com/typicode/json-server)

**13. Appendix**

Include:

* Code snippets
* Screenshots of each page
* JSON sample from db.json