**Daily Todo Management**

**Project Overview**

This web application enables users to manage their daily tasks efficiently. Users can create, update, delete view their daily todos, as well as mark them as complete. This application will be utilizing local storage to ensure data persistence.

**Project Plan**

**Day 1: Planning:**

**Development Setup:**

1. Set up development environment -completed  
   - Create react-daily-todo app using @latestvite
2. Initialize project on github -completed  
   - Create new repository named task management  
   - Commit any recent updates onto new repository

**System Planning:**

1. Draft system architecture – completed  
   - Identify key components (frontend, backend, database)
2. Draft system Requirements - completed  
   - Identify target users  
   -List functional requirements (users can add, delete, edit todo)  
   -List non-functional requirements(User friendly interface)

**Design:**

1. Create wireframes  
   - Design the user interface layout

**Project Requirements:**

1. Outline how to use application.  
   -Provide instructions on how to add, edit, delete and mark a todo as complete

**Project Plan - Day 2  
Define the Requirements**

**Features:**

* Add a new to-do item.
* Display a list of to-do items.
* Mark a to-do item as completed.
* Remove a to-do item.
* Persist to-do items in local Storage.
* Fetch and update to-do items via a REST API.

**Break Down Frontend Components into Smaller Fragments**

1. **Header Component**

* Create a header component which includes a basic greeting.  
    
  **App.js Component**
  + This will be my main component where everything comes together.
  + State for todos will be created within this component so that it may be utilized by the children components

1. **Task List**
   * Display todo list within a table
   * Display an edit, complete and delete button next to the new task.  
     - Implement state and functions to handle editing, handle delete, toggle complete   
     - Integrate try catch and fetch todos from an API
2. **Add Todo**
   * Create a form component to add new todos.
   * Include input fields for adding a todo.
   * Implement a handlesumbit function to validate the form input.
3. **Persist Todos to Local Storage**
   * incorporate local storage

**Main Challenge On Day 2**

* Incorporating a toggle feature within the frontend

**Project Plan - Day 3**

**Backend RESTFUL API using Node.js and Express. js**

1. **Backend**

* Create a server within the backend of my project in a file named index.js.
* Install Express.js
* Implement basic api endpoints to GET, POST, PUT and DELETE todos
* Create an in memory database to store all the todos

**Main Challenge For Day 3**

* Upon integrating RESTFul API's, I've encountered an issue related to data persistence. It appears that data stored in the in-memory array (todos) on the server does not persist across server when it restarts.

**Improvements**

* Adding try catch to the endpoints, so that errors maybe handled effectively.

**Day 4: Incorporating MySQL Database**

**Objectives:**

* Set up MySQL database.
* Integrate MySQL with the backend.
* Modify existing API endpoints to interact with MySQL.

**Tasks:**

* Set Up MySQL Database
* Install MySQL on my local machine
* Create a database for my project using workbench
* Create tables for storing todos with appropriate fields (e.g., id, task, completed).

CREATE TABLE todos ( id INT AUTO\_INCREMENT PRIMARY KEY, task VARCHAR(255) NOT NULL, completed BOOLEAN NOT NULL DEFAULT FALSE );

* Integrate MySQL with Backend
* Install MySQL Node.js driver (mysql2).
* Establish a connection to the MySQL database from Node.js server.
* Update the API endpoints to use MySQL for data persistence.
* Modify API Endpoints
* GET /todos: Retrieve todos from the MySQL database.
* POST /todos: Insert new todos into the MySQL database.
* PUT /todos/ : Update existing todos in the MySQL database.
* DELETE /todos/ : Delete todos from the MySQL database.

**Main Challenge for Day 4:**

* Learning MySQL and ensuring proper integration with your Node.js backend.

**Day 5: Finalizing the Project and Testing**

* Finalized the database integration.
* Implemented remaining features and improvements.  
  I learned some more sql