**Task Manager Application Documentation**

**Link to assignment:** **https://drive.google.com/drive/folders/1A9MmMRdBymqBIAoUFXgeo8UwXb8pmmej?usp=sharing**

**Overview**

The Task Manager application is designed to help users manage their tasks effectively. It provides functionalities for creating, editing, and deleting tasks, with a user-friendly interface.

**Directory Structure**

TaskManager/

│

├── index.html # Landing page for the Task Manager application

├── style.css # Cascading Style Sheets (CSS) for styling the application

├── script.js # JavaScript file for frontend interactions

│

└── task\_manager\_api/ # Backend API for Task Manager

│

├── \_\_init\_\_.py # Initialization file for Flask application

├── app.py # Main Flask application file

├── models.py # SQLAlchemy models for database

├── routes.py # API routes definition

├── requirements.txt # List of Python dependencies

└── static/

├── style.css # Static CSS files for Flask application

└── script.js # Static JavaScript files for Flask application

## Backend Overview

The backend of the Task Manager application is built using Flask, a Python web framework. It utilizes SQLAlchemy for database operations and provides RESTful API endpoints for frontend interactions.

### **Key Files**

* **app.py**: Initializes the Flask application, configures database, registers routes, and starts the development server.
* **models.py**: Defines SQLAlchemy models (Task) for tasks in the application, including attributes like id, title, description, and due\_date.
* **routes.py**: Defines API endpoints (/api/tasks) for CRUD operations (Create, Read, Update, Delete) on tasks. Handles requests, interacts with the database, and returns JSON responses.

## Frontend Overview

The frontend of the Task Manager application is developed using HTML, CSS, and JavaScript. It interacts with the backend through API calls (fetch) to perform operations on tasks.

### **Key Files**

* **index.html**: Provides the user interface for Task Manager, including task creation form and task listing.
* **style.css**: Styles the HTML elements, making the application visually appealing and responsive.
* **script.js**: Handles frontend logic, such as fetching tasks from the backend, rendering tasks dynamically, and handling user interactions like task creation, editing, and deletion.

## Usage

1. **Setup and Installation**:
   * Clone the repository from GitHub.
   * Set up a virtual environment (venv) and install dependencies from requirements.txt.
   * Configure the Flask application (app.py) to use the appropriate database URI (sqlite:///tasks.db).
2. **Running the Application**:
   * Activate the virtual environment (venv).
   * Run python app.py to start the Flask development server.
   * Access the Task Manager application in your web browser at http://localhost:5000.
3. **Interacting with the Application**:
   * Create tasks using the form on the landing page (index.html).
   * Edit or delete tasks using the corresponding buttons on each task card.
   * All frontend actions are communicated to the backend through API calls (fetch).

## Dependencies

* Flask
* Flask-CORS
* Flask-SQLAlchemy
* Python 3.x