# Task Management System

# Group names

**Shadarien Williams**

**Computer Programming Capstone**

**24/Fall –**

Instructor: Dr. Mubarak Banisakher

November 3,2024

Table of Contents

**Type chapter title (level 1)1**

Type chapter title (level 2)2

Type chapter title (level 3)3

**Type chapter title (level 1)4**

Type chapter title (level 2)5

Type chapter title (level 3)6

## Purpose of the project:

The purpose of this project is to develop a Task Management System that allows users to organize, prioritize, and track their tasks efficiently. The system is designed to help individuals and teams manage their workflows by providing features such as task creation, categorization, due dates, priority levels, and status tracking. The application will be intuitive and user-friendly, enabling users to maintain productivity and ensure that all tasks are completed on time.

## System workflow:

The Task Management System will follow a straightforward workflow where users can create tasks, assign categories, set due dates, and prioritize their workload. Users can also mark tasks as complete and filter or sort them based on various criteria such as due date, priority, or status. The system will store all task data in a database, ensuring that users can access and manage their tasks from any device connected to the internet.

## Platform (e.g., Windows, Mac, iOS):

The system will be built using the Django framework and will run on Linux-based operating systems. It will be accessible through a web browser, making it platform-independent and usable on various devices, including desktops, laptops, and mobile devices

## Intended customer/user (that is, who is the program for?):

## The Task Management System is intended for individuals and teams who need an efficient way to manage their tasks and projects. It is suitable for professionals, students, and anyone looking to enhance their productivity and organization. The system will be particularly beneficial for small teams that need to collaborate and keep track of their collective tasks

Financial traders and individual investors interested in both organizing their trading portfolios and with developing well planned and tested trading strategies. Specifically, it is for my father and for myself (see below).

## Source of the idea for the project:

The idea for this project stems from the need for a simple yet effective tool to manage tasks and projects. Many existing task management applications are either too complex or lack the necessary features for effective task tracking. This project aims to bridge that gap by providing a balanced solution that is both easy to use and comprehensive enough to meet the needs of various users. A prime example of this idea is the Canvas platform. The platform contains a todo list section however, it's only available in the mobile version. I wanted to make a platform that was universally accessible and completely customizable.

## Development environment and/or tools to be used:

* Django: The primary web framework for building the application.
* Python 3.x: The programming language used to develop the system.
* SQLite: A lightweight database engine for storing task data.
* HTML/CSS/JavaScript: For designing the front-end interface.
* VS Code: The integrated development environment (IDE) used for coding.
* Git/GitHub: Version control and code collaboration tools.

## Development language(s) and major 3rd party libraries you plan on using:

* **Python 3.x:** For backend logic and database interactions.
* **Django ORM:** For managing database operations.
  + **Link:** [Django documentation | Django documentation | Django (djangoproject.com)](https://docs.djangoproject.com/en/5.1/)
* **HTML5/CSS3:** For creating the user interface.
* **JavaScript:** For enhancing interactivity and handling client-side logic.

**NOTE:** Although numerous libraries and some standard Python frameworks are used (e.g., Backtester) it is anticipated there will be well in excess of 500 lines of original code generated in this project which will more than meet the original code requirement.

## limitations or risks anticipated:

**Limitations:**

* **Scalability:** The system might need adjustments if scaled for larger teams or more complex workflows.
* **Data Security:** Ensuring that user data is secure will be a priority, but potential vulnerabilities may exist.
* **Browser Compatibility:** Ensuring consistent performance across different browsers.
* **Time Constraints:** Balancing the scope of the project with the available development time.

**IMPORTANT:** To complete this somewhat complex ensemble of codes within the relatively short class development period, ALL parts of this code structure will be “bare-bones” and simple without many “bells and whistles” (minimum functionality only will be achieved). This will form a code base on which additional modifications and refactoring can be done for future enhancements.

## Schedule (built around the course milestones):

Stage 1:

* Write proposal.
* Study code from similar projects using the same or similar libraries.
* Select libraries and API’s that will be used in the system.
* Set up of development environment. Stage 2:
* Specify initial functional requirements.
* Create UMLs for use cases, defines uses cases.
* Research and finalize libraries and tools
* Review and finalize requirements

Stage 3:

* Begin Preliminary design, including wireframes and database schema
* Create a test matrix for use cases.
* Document the design process.

Stage 4:

* Develop the backend logic and database models.
* Implement user authentication and task management features.
* Begin frontend development
* Go over preliminary design.
* Create Preliminary Design Document (the “How” the system works- flowcharts, pseudo-code, etc.).
* Document any new information about system. Stage 5:
* Integrate the frontend with the backend
* Implement task filtering, sorting, and search functionalities.
* Conduct unit testing

Stage 6:

* Conduct user acceptance
* Finalize documentation and user manual
* Document any new information about system. Stage 7:
* Prepare the final presentation
* Demonstrate the project
* Submit the final project and documentation

## Estimated number of lines of code:

It is anticipated that the project will involve more than 500 lines of original code, including backend logic, frontend design, and database interactions. The use of open-source libraries will be limited to standard tools such as Django and Bootstrap, with no more than 30% of the code being modified from existing sources

## Documentation:

Documentation will include detailed comments within the code, a user manual, and technical documentation covering the system architecture, database schema, and API endpoints. All documents will be made available in PDF format.

## User training:

Training will be provided through detailed user manuals and in-person or virtual sessions as needed. The documentation will cover all aspects of using the system, from task creation to advanced features like task filtering and prioritization.

## Delivery/installation plan:

The Task Management System will be delivered as a web application hosted on a server. Users will be able to access it via a web browser, with installation instructions provided for setting up the system locally or on a cloud-based server..