AnGp211 – Python Programming Project

Development of an Engineering Task Management Interface

Context:

In the field of engineering, managing tasks efficiently is crucial to ensure that projects meet deadlines, maintain high quality, and stay within budget.

Whether in aerospace, automotive, or systems engineering, engineers need to coordinate tasks across different departments (design, testing, manufacturing, etc.), monitor progress, and address dependencies between tasks.

Objective:

The goal of this project is to develop a desktop application using Python and Tkinter that serves as a task management tool for engineering teams. The application will allow users to create, assign, and monitor tasks related to various stages of an engineering project.

Pedagogical objectives

- To refresh and further fundamental knowledge of your Python courses (In211) such as algorithms, functions, and object-oriented programming.
- Learn the software development lifecycle(various phases of development).
- Use Paper prototyping to design your interface.

> ***Paper prototyping*** *is the process of developing ideas and designing user flows using hand-sketched “screens” that represent a digital product.*

- Modeling your programs with the class diagrams (UML).

Development Requirements

- Python3, Tkinter
- Code Editors : Python IDLE, Visual studio, Spyder or other IDE

Project Requirements

1. *Task Creation and Assignment:*

The interface should allow users to create tasks with details such as:

- Task title
- Description
- Priority (e.g., Low, Medium, High)
- Deadline

2. *Task Status and Progress Tracking :*

- Each task should have a status (e.g., Not Started, In Progress, Completed)
- Users should be able to update the status of each task through the interface.

3. *Task Dependencies :*

- The application should allow users to set dependencies between tasks (e.g., Task B cannot start until Task A is complete).
- The interface should indicate when a task is blocked by another task's incomplete status.

4. *Data Persistence:*

- Task data should be saved in a file format such as CSV or JSON, so that tasks are not lost when the application is closed.

5. *User-Friendly Interface:*

- The interface should be intuitive and easy to navigate.
- It should feature buttons for common actions like creating, deleting, or updating tasks.
- The task list should be sortable by priority, department, or deadline.

6. *Notifications and Alerts (Optional):*

The interface should generate alerts for:

- Overdue tasks (past their deadline).
- Tasks with high priority that have not yet been started.
- Notifications can be displayed as pop-ups or highlighted in the task list.

Planning

Session 1 :

- Project description and sessions planning
- Setting up of working groups(2/3 students) and specifying the application objectives.
- Approve the project topic proposed byeach group
- Initial planning: the main implementation steps
- Preparing development environment (project folder, files, tools, modules)
- Submit your project topic on Moodle (session1)

Session 2 :

- Paper prototyping to define interface architecture (what this app will look like and its components)
- Initialize the UML diagrams for the Model and start the development with Python.
- Test your initial model with data sets.
- Submit your program(python files) + UML diagrams on Moodle (session2)

Session 3 :

- Create the user interface with Python and Tkinter.
- Test your model with interface and external data.
- Debug your application.
- Submit your program(python files) + UML diagrams on Moodle (session3)

Session 4 :

- Test and debug your application with multiple data sets.
- Add extra functionalities to your app.
- Prepare your presentation (slideshow).
- Submit your final project(python files) + UML diagrams on Moodle (session4)

Session 5 :

- Presentation and Demo.

Presentation

You will present your project orally (in **10-15 minutes**) with :

- Slideshow (PowerPoint or other) using diagrams that you created during your project.
- Application demo.

/!\ all slides should be numbered /!\ :

- slide 1: Title + Author’s name + school logo + subject title (AnGp211) + year + etc.
- slide 2: Context and objectives of project with steps / tasks to do.
- slide 3 to x (x≤6):
  - application structure, UML diagrams(of class definition and algorithms), external files, or libraries.
- Application demonstration
- slide x+1: conclusion:
  - project improvements, achievements.
  - your feedback from project, difficulties, achievements.

Grading criteria

Presentation (8 pt)

- Follow the instructions and clarity of presentation.
- Presentation of application structure and quality of UML diagrams
- Questions

Project (12 pt)

- Session’s submission
- Quality of application
- Code organization and respect of coding rules in Python.
- Code structure (modularity), use comments and documentations.

Documentations

Python

- *https://docs.python.org/3/tutorial/index.html*
- *https://python.sdv.univ-paris-diderot.fr/*

Tkinter

- *https://www.pythontutorial.net/tkinter/*
- *https://www.tutorialspoint.com/python/python_gui_programming.htm*
- *https://www.geeksforgeeks.org/python-tkinter-tutorial/*

Modern Tkinter

- *https://ttkbootstrap.readthedocs.io/en/latest/*
- *https://customtkinter.tomschimansky.com/*