Rapportmall: Systemutveckling i Python – Individuell slutuppgift

Studentinformation

Namn: **Umaporn Sratongyung**
Klass: **DevOps (DOE25)**
Datum: **23 October 2025**
GitHub-länk till projektet: **https://github.com/FahUmaAngel/Python-assignment**

1. Introduction

Briefly describe what the assignment was about.

Answer:

In this assignment, I developed a system monitoring program using Python.
The program starts by checking the system status ("List monitoring"), creates and displays alerts for CPU, RAM, and DISK usage ("Create Alarm" and "Show Alarm"), and allows users to remove unnecessary alerts ("Remove Alarm"). It also includes functionality to manage the monitoring mode ("Start Monitoring Mode").

The purpose of this assignment is to practice using Python for system development within the DevOps concept by applying functions and object-oriented programming (OOP), as well as importing and using code from other files.

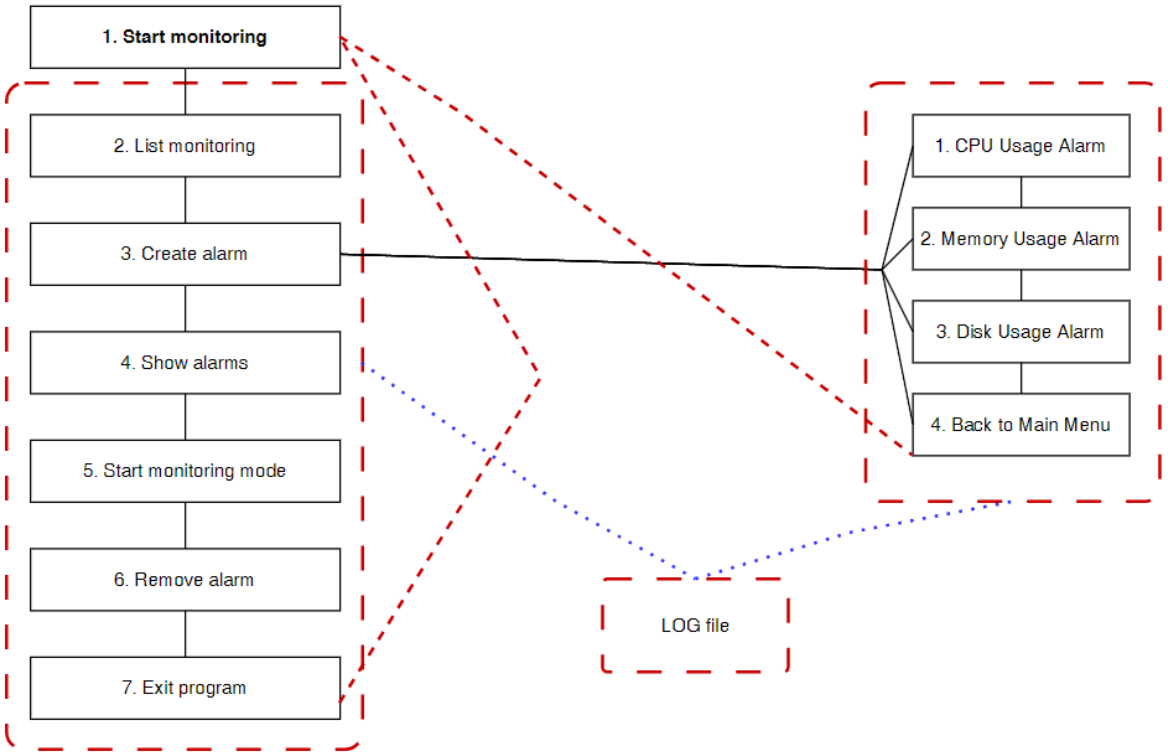
2. Planning and Design

How did you plan your work?

Answer:

I started by translating the assignment from Swedish to English to gain a clearer understanding of the requirements. After that, I tested the assignment commands on https://app.base44.com/ to see how the assignment works in practice and what it is supposed to do. This helped me visualize the overall functionality of the program, which I then used as a reference for designing the program's flowchart.

Afterward, I designed the program's flowchart and planned the functions step by step for each menu option.



3. Planning and Design

How did you plan your work?

Answer:

The program is divided into several files: **main.py, alarm.py, logger.py, monitor.py, and storage.py.**

The **main.py** file serves as the main menu and the entry point of the program. The **MonitoringApp** class is responsible for reading system information using the psutil library (imported from ***monitor.py***) and contains functions (*def*) for handling the commands of each menu option, as well as loading and saving alarms.

In **alarm.py**, the Alarm class manages the created alerts and is called from ***main.py***.

The **monitor.py** file contains functions that display the performance and usage of the CPU, RAM, and DISK in percentages and in gigabytes (GB) of total usage.

The **storage.py** file is responsible for saving and loading alarm data using a JSON file.

The **logger.py** file handles logging of various events and saves them into a .txt log file.

4. Important Functions or Classes

Choose 2–3 key parts of your code and describe them in more detail.
Briefly explain what they do and why you chose to implement them in that way.
Answer:

One of the key components in my program is the **AppLogger** class in the **logger.py** file.

This class is responsible for creating and managing log files that record system events and actions performed by the user.

When an instance of AppLogger is created, it first checks whether the directory named “logs” exists, if not, it automatically creates it.

Then, it generates a new log file with a unique name based on the current date and time (for example: **log_20251023_153045.txt**).

The log() method is used to write messages or events into this log file. Each log entry is saved together with a timestamp in the format **YYYY-MM-DD HH:MM:SS**, making it easy to track when each event occurred.

I chose to implement this functionality in a separate class so that logging can be easily reused throughout the program, improving both organization and maintainability.

5. Libraries and Tools

Which external or built-in libraries did you use, and for what?

Answer:

Libraries and Tools Used:

os: for handling files and working with the operating system.
datetime: for working with dates and times, such as creating timestamps for logs.
threading: for running tasks in parallel, for example monitoring system resources while still allowing user interaction.
psutil: to read system resources such as CPU, RAM, and disk usage.
json: to save and load alarm data in a structured format.

Version Control:

The project was managed using Git for version control and GitHub for remote repository hosting. Each feature or update was committed with descriptive messages, allowing me to track changes and revert to previous versions if needed. This workflow ensured that the project remained organized and easy to maintain.

“git init

git add .
git commit -m "comment"
git branch -M main
git remote add origin https://github.com/FahUmaAngel/Python-assignment.git
git push -u origin main
”

6. Testing and Debugging

How did you test your program?
● How did you ensure that the functions work as intended?
● Did you test different user inputs?
● Did you have to handle any bugs or errors?
Answer:

I tested my program by building it step by step, one menu at a time. At each step, I used print() statements to display outputs and verify that each function worked correctly before moving on to the next. The program was run in the Git Bash terminal to check outputs and test interactions.

I also tested various user inputs to ensure that the program could handle unexpected or invalid entries. To prevent crashes, I implemented try/except blocks for input handling.

When bugs or errors occurred, I first read the error messages directly from the terminal to understand the issue. If I could not figure it out, I asked Copilot for guidance.

Throughout the development process, I learned from books, online resources, and GitHub Copilot, which helped me understand Python concepts, program structure, and best practices for debugging and testing.

7. Results

What works well in the program?
Are there any parts you are particularly satisfied with?
Answer:

I am satisfied that the program works exactly according to the commands I wrote and executes the workflow in the correct order. I am especially pleased with the main.py, because I believe that organizing it correctly and understanding the program’s structure from the beginning helps ensure that the operations follow the proper sequence efficiently.

I am also very proud of myself for writing a program in Python for the first time.

8. Reflection and Lessons Learned

What have you learned during the project?
● About Python and programming
● About structuring code
● About testing, Git, and workflows
Answer:

From this project, I learned a lot about programming with Python for the first time. Creating a system monitoring program helped me understand how to use functions and classes to organize my code clearly and make it easier to maintain.

I also learned the importance of planning before coding, like drawing a flowchart of the program and breaking the menu into sections so that everything runs step by step efficiently.

In addition, I practiced testing and debugging by running the program menu by menu, using print() to check the outputs, and then gradually replacing them with functions for each menu as needed. Using Git and GitHub helped me keep track of project versions in an organized way, making it easy to go back or compare changes.

9. Possible Improvements and Further Development

If you had more time, what would you like to add or improve?

Answer:

If I had more time, I would like to add a GUI to the program to make it more user friendly. I have started studying GUI development, but it is quite complex and takes a lot of time to learn properly.

10. Summary

Answer:

This project shows that I can use Python to create a simple but working system monitoring program.

While working on it, I practiced object-oriented programming, file handling, error handling, and using Git and GitHub for version control.