



General

[Collapse all](#)



[Ankündigungen](#)



[Lecture Link](#)



[GitHub Repository](#)



vhb Course: IT Fundamentals for Students of Economics

Please refer to [this link](#) to get more information about registration. **ONLY** chapters 2, 3, 4, 11 and 12.

ATTENTION: Registration is only possible until 30th November



1: Foundations



Introduction and Setup



[Einführung-in-Python-and-Setup-mit-Visual-Studio-Code-en-US](#)



Data Types and Variables



Control Structures



Unit Tests



[Unit-Tests-in-Python-Quality-Assurance-From-the-Start](#)



GitHub Repository



[Understanding-Git-and-GitHub-A-Beginners-Guide](#)



2: Modularisation and Functions



3: Data and Object Oriented Programming



4: APIs, Deployment and Automatisation



[Streamlit-The-Future-of-Interactive-Data-Applications](#)



5: Best Practices



Exam: Final Project



[Grouping](#)

Please build the groups with minimum 4 and maximum 5 Persons for your examination work.



[Group & GitHub Names](#)

Please fill out the document by adding your **name**, your **group** and your **gitHub username**.

Date of examination: 17th January, 2026

Time: 13:30

The examination consists of two parts:

1. Virtual written Vhb test (individual work) - **ONLY** chapters 2, 3, 4, 11 and 12 --> **PLEASE REGISTER URGENTLY!!!**
2. Project work (group work) --> Grouping is possible [here](#)

Please organise yourselves and form groups of 4-5 people. Individual work is not permitted. As a group, select **ONE** of the following topics and work on it. **I will provide you a gitHub space which you have to push your code.**

Topics

1. Develop a Python programme that accepts a list of medications (either entered manually or loaded from a database/CSV file) and checks them for potential known drug interactions.
Use LLMs to help build the initial data structure, optimise the search algorithms, and write the user interface (e.g., with flask and streamlit).
2. Create a simple SIR (Susceptible-Infected-Recovered) model simulator or a similar epidemiological model that simulates the spread of a disease (e.g. Ebola or any other viral disease) over time. The results should be visualised (e.g. using Matplotlib or Plotly). As a user, you can adjust parameters such as infection rate, recovery rate and population size.
Use LLMs in the mathematical formulation of the model (the differential equations), the creation of code snippets for the numerical solution (e.g. Euler method) and the creation of the plot functions.
3. Develop a system that accepts appointment requests from patients and schedules them as efficiently as possible in a doctor's calendar (data structure for time slots), taking into account priorities (e.g., emergency vs. routine) and desired times. The result is a list of confirmed appointments.
Use LLMs to research optimisation algorithms (e.g. simple greedy algorithms) or the structuring of time objects (datetime module).

Evaluation Criteria

Code quality and functionality

Criterion	weighting	Description
Functionality and correctness	20%	The programme must meet all core requirements specified in the project specification and function without errors. This is primarily demonstrated by passing all unit tests.
Code style and formatting	15%	Compliance with Python conventions (PEP 8), clean naming of variables/functions and consistent formatting.
Modularity and design	15%	The code is structured logically. Separation of data processing, logic and user interface. Sensible use of functions and classes (object-oriented design).

Understanding the algorithms

Criterion	weighting	Description
Core algorithms and logic	15%	Evidence of own implementation of central, project-specific logic. Explanations in the technical report.
Code Coverage	10%	The proportion of code covered by the written tests (at least 80% coverage).
LLM use and reflection	10%	The technical report must clearly document where LLMs were used and where the solutions proposed by LLMs were adapted and improved.

Documentation and project management

Criterion	weighting	Description
Technical documentation	5%	A clear README.md file with installation instructions, usage examples, and a brief overview of the architecture.
Technical report (PDF)	5%	A short document (max. 3 pages) explaining the design decisions, data structures and technical rationale for the chosen algorithms.
project management (Git-Historie)	5%	Balanced Git commit history with equal participation from all group members, use of pull requests, and meaningful commit messages (evidence of collaboration).

 [Contact site support](#) 

You are logged in as [Awaadallah Yusef](#) ([Log out](#))

[Get the mobile app](#)

[Impressum](#)

Moodle Version 4.5