House work

Objective: Build the development environment(s) and obtain the results up to a web application based on scripts that allows full replication, starting from an original setup with VirtualBox and Vagrant.

1) Starting point is installing VirtualBox and Vagrant on your user account in the ICT network or on the private computer

2) Vagrant: Creating a basic box:
   - Debian-11 (others with justification)
   - German localized
   - GUI / X11 desktop

   bonus:
   - Cleanup for the lowest possible disk space requirements of the box / VMs

3) Vagrant: Create a derived box equipped with
   - Dockers
   - current docker compose
   - other required tools: Wireshark, make, etc.
     (if you need additional information later, please update the vagrant file accordingly)

House work

4) Vagrant:
Create the development VM from the box of task 3 as a development computer for all further tasks.
Docker:
Creating a base image as the basis for all of the following:
- Debian-11
Bonus:
- Precaution for simple operation when building the image and starting, stopping and removing a container as well as when calling a shell in the running container, e.g. via Makefile

5) Docker: create three derived images for containers of kind
- 'Host' with Python installed,
- Database server - e.g. Redis - for persistence,
- web application server with flask,
their networking and proof of accessibility (ping / Wireshark).
Organization of the configuration and networking, via dockercompose.yml.

6) Create a simple application running in the 'host' container with database connectivity to demonstrate the functionality of the data banking solution and proof of the achievement of persistence.

House work

7) Flask: Create a simple Hello World application for the web application framework flask to demonstrate the basic functionality (for tasks 7...10 based on HTTP & HTML)

8) Create an application that
- allowed to enter usernames via POST / WTForms,
- enters new names in the database and generates a corresponding feedback page and
- If the name is already known, reports the number of visits and the date and time of the last visit.

9) Improve your application in terms of content preparation and appearance by using
- Jinja2 templates
- CSS

(if not already done; possibly reference to corresponding results in the previous solutions)

10) Extend your application with registration/login/logout functionality so that only authenticated users have access.

House work

Bonus: (optional, transfer)

Replace or supplement the HTTP/HTML functionality of your application with a web API that allows all application actions to be performed.
Base the data exchange on JSON. Use curl as a client (possibly Postman, Insomnia or similar with justification)

Additional bonuses:

- Replacement of User:Password-based Web API authentication with token-based one
- Replacement of the Redis database with an RDBMS and connection via SQLAlchemy
- Write a simple, socket-based client program that addresses your server and interacts with it (GET, possibly also POST)
- Replacement of make / Makefile with another tool, e.g. Ansible