Summerterm 2023
Prof. Dr. Andreas Wölfl (andreas.woelfl@th-deg.de)

TECHNISCHE HOCHSCHULE DEGGENDORF

Software Engineering

"'Praktikumsleistung"'

Complete **Task 1** and **Task 2** to complete the project work. ("Praktikumsleistung")

Note the following rules:

- Groups of 6-7 students must make a joint submission.
- The submission must include:
  - The names and matriculation numbers of all group members.
  - All source code and configuration files.
  - The presentation document (as PDF).
- Submit the solution in the form of a ZIP archive via iLearn.

Successful completion of **Task 3** carries a 10% grading bonus.

Task 1 - Enigma Engineering

Implement a software emulation of the *Enigma* cipher machine.

The following **functional** requirements shall apply:

- The software shall have a graphical user interface (GUI).
- The GUI shall display a virtual keyboard and lamp panel.
- While a user presses a key on the physical keyboard, the corresponding key of the Enigma's virtual keyboard as well as the cipher-character (i.e., the scrambled key) in the lamp panel shall be highlighted.
- While a user performs a mouse click on a key of the Enigma's virtual keyboard, the cipher-character in the lamp panel shall be highlighted.
- The GUI shall display the history of the last 140 key/cipher-character pairs.
- The key scrambling shall be implemented based on an entry disk, multiple rotors, a reflector, and an optional plugboard. The functions are described here[1].
- The software shall support multiple variants. A variant shall differ in the presence or absence of the plugboard, the number of installable rotors, as well as the set of selectable rotors and reflectors.

[1] https://www.cryptomuseum.com/crypto/enigma/i/index.htm

- The variants shall be configurable in a sense that no modification of the source code is required to add, modify, or delete a variant.
- The software shall ship with variants for the *Enigma B*, the *Enigma I* and the *Enigma M3*. See this article[2] for details.
- A user shall be able to apply a ring setting (rotor & reflector selection).
- A user shall be able to apply a key setting (initial position of the rotors).
- A user shall be able to apply a plugboard configuration (up to 10 plugs).
- Closing and re-opening the application shall retain the state of the machine.
- A user shall be able to reset the machine to its initial state.

The following **non-functional** requirements shall apply:

- The source code shall be written in one (or more) of the following programming languages: Java, C++, Python, JavaScript.
- The source code shall be managed in `mygit`[3].
- The application shall be containerized using Docker images.
- The Docker images shall be built with Gitlab CI and pushed to the registry.
- The application shall be shipped with a Docker *compose* specification.
- All functional requirements shall be verified in terms of E2E tests.
- The project shall be conducted according to the *Scrum* framework using time-based capacity planning with a total of 3 sprints.
- A Product Backlog shall be created and maintained according to this document.
- All non-code project resources shall be managed and maintained with Jira[4]. At any time, the assets in Jira shall reflect the current state of the project.
- The dates and deadlines published in iLearn under the section *Dates and Deadlines* shall apply. Note that all deadlines are hard.

Task 2 - Presentation

Create a presentation document of 10-15 slides. It shall demonstrate the software developed in **Task 1**, provide insights on the architecture of the system, and highlight aspects that make the solution special. Furthermore, each group member is tasked to create and present a slide detailing the personal contribution to the project.

[2] `https://www.cryptomuseum.com/crypto/enigma/wiring.htm#12`
[3] `https://mygit.th-deg.de`
[4] `https://jira-stud.th-deg.de`

Task 3 - Quality & Security

In addition to Task 1, the following **non-functional** requirements shall apply:

- The CI system shall perform nightly builds.
- Test coverage (integration and unit tests combined) shall be above 75%.
- The Docker image shall be based on Alpine Linux[5].
- As part of the CI pipeline, static code analysis shall be performed. Conduct a market analysis and propose a suitable solution to the Product Owner.
- As part of the CI pipeline, the Docker image shall be scanned for CVEs. Conduct a market analysis and propose a suitable solution to the Product Owner.
- At submission time, all Docker images must be free from CVEs.

[5] `https://alpinelinux.com`