Network Infrastructure

Timing: 2nd year of study **Scope: 10** ECTS

Content: This course covers switching, routing, and wireless technologies, along with security fundamentals. It also introduces enterprise network design, WAN technologies, QoS, and modern concepts like virtualization, automation, and software-defined networking.

Learning objectives:

Knowledge

The student must have knowledge of:

- Principles of design and implementation of distributed systems; and
- Fundamental network concepts

Skills

The student can:

- Include relevant technological aspects in the development of distributed systems, including:
  - Central security concepts and threats;
  - Use of virtualisation;
  - Use of services and programming interface for communication; and
  - Application of common application protocols.

Competencies

The student can:

- Reflect on choice of infrastructure in connection with development of distributed systems.

The compulsory educational component is completed with an exam (Programming exam).

The examination:

Internal written exam consisting of two CCNA final exams (multiple choice).

Assessment:

7-point grading scale. Grading is based on the two exam scores, which are added up and divided by 2 to find the average score.

Internet of Things (IoT)

Timing: 2nd year of study **Scope: 10** ECTS

Content:

IoT aims at equipping students with the necessary concrete and abstract skills to be competent to develop prototypes of IoT systems, ultimately to explore new ways of creating value for business.

Learning objectives

Knowledge

The student must have an:

- Overview of the IoT eco-system and current usages
- Understanding of IoT's role in business

Skills

The student has acquired the skills in the following areas:

- Programming real-time microcontrollers. (No operating system, no scheduling, etc.)
- Connecting to different sensors like temperature, air-pressure, humidity, rain, distance, etc.
- Connecting to motors, fans, pumps, lights, etc.
- Web scraping
- Digital vs. analog electronics in practical use. Digital/analogue conversion
- Basic understanding of driver software for different hardware
- Decentralized communication protocol (MQTT)
- Data collection

Competencies

The student can:

- Design prototypes of IoT devices
- Advise on core technical implementations of IoT in a business setting

The examination:

Exam is held as an individual, project and synopsis based, oral exam. The project is a group project and the synopsis is individual. The oral exam begins with an 8-minutes group presentation and demonstration, followed by a 17-minutes individual exam including evaluation.

Assessment:

7-point grading scale. The assessment is based on an overall evaluation of the oral exam, the synopsis and the written project.

Fullstack Development

Timing: 2nd year of study **Scope: 10** ECTS

Content: The purpose of the course is to introduce the student to real-time communication systems between browsers and web servers and advanced software architecture, such that students can develop novel business applications using modern technologies.
Learning objectives: *Knowledge* The student must have knowledge of: • Protocols and technologies used to facilitate real-time web communication • Code organization and architecture planning *Skills* The student can: • Test and debug API's communicating bi-directionally with web clients • Implement and scale stateful server applications • Reflect on advantages and disadvantages of architectural decisions • Structure and debug event-based systems *Competencies* The student can: • Conduct technical decision-making and systems design for web development • Integrate heterogeneous components in web-based systems

The examination:
Oral exam is based on written project submission. The student submits a link to a version control repository with source code and system documentation. The submission must include a client and server application built using the tech stack used throughout the course. The exam duration is a total of 20 minutes including grading.

Assessment:
7-point grading scale. The evaluation is an overall assessment based on written and oral performance.

Mobile Development

Timing: 2nd year of study **Scope: 10** ECTS

Content: In this course the student will learn how to make cross-platform mobile (smartphone) apps. Main focus is programming, structure and state management. User experience will also be taken into consideration.

Learning objectives:

Knowledge

The student must have knowledge of:

- Core concepts of Dart programming language.
- Building UI with widgets in Flutter.
- Stack based navigation.
- Principles for separating logic and presentation.
- Reactive programming with BLoC pattern.

Skills

The student can:

- Design user interfaces suitable for smartphones.
- Utilize phone features such as location, camera and notifications.
- Manage application state.
- Communicate with external APIs.
- Test and debug apps.
- Use packages and plugins.

Competencies

The student can:

- Develop useful apps suited for smartphones.
- Reflect on design, structure and architecture of apps.

The examination:

Exam is an individual oral presentation of self-made mobile application. The presentation is based on a written synopsis. The exam duration is 15 minutes including grading, wherein the student is expected to present for 5 minutes.

Prerequisites for the exam:

A synopsis for the project must be submitted along with a link to the code. The synopsis must have the full name of all contributors and a link to the source code on the front page.

Assessment:

7-point grading scale. The evaluation is based on the oral performance.

Applications of Artificial Intelligence

Timing: 2nd year of study **Scope: 10** ECTS

Content: In this course the students will learn how to incorporate AI into their applications using Large Language Models (LLM) and how to orchestrate all crucial components in a proper manner. The students will also learn about LLMs and their parameters so they can determine the best LLM(s) for their concrete hardware and needs.

Learning objectives:

Knowledge

The student must have knowledge of:

- What a large language model (LLM) is.
- Which providers there are on the market.
- Parameters stating the size and heaviness setting requirements for the hardware processing them.
- Where to find existing open-source models and compare them.
- Which systems do we have that can process open-source models.
- Sustainable aspects of using AI.

Skills

The student can:

- Develop basic applications using Python.
- Structure an LLM powered application in such a way that external data can be used for further improvement and memory functionalities.
- Use proper Python libraries and tools for LLM communication.
- Use basic prompting techniques making LLM responses more concise and have control of the response.
- Deploy python applications.
- Use tools for testing and asserting an LLMs and providers in terms of responses.

Competencies

The student can:

- Develop applications that uses the power of LLMs.
- Determine which LLMs can be used for a specific hardware configuration.

The examination:

Oral exam based on project and written submission. The exam duration is 15 minutes (the student presents 5 minutes, Q'n'A 5 minutes, assessment 5 minutes).

Formal requirements to the project:

A Python application that is empowered by LLMs using orchestrations tools and quality measures learned in class.

Formal requirements to the written submission:

A mini report at a maximum of 10 pages.

Assessment:

7-point grading scale. The evaluation is an overall assessment based on the project, the report and the oral performance.

DevOps

Timing: 2nd year of study **Scope: 10** ECTS

Content: This course discusses the concept of DevOps as an approach to delivering software in a fast and secure manner without compromising quality and stability of the deliveries. Our primary tools to succeed in that journey will be automation through continuous integration, delivery and deployment combined with strategies to recover if something doesn't work out as expected.

Learning objectives:

Knowledge

The student must have knowledge of:

- The overall objective of DevOps and how working with automation, measurement and recovery can help achieve these problems.
- The concept of pipelines and how it relates to DevOps.
- Different aspects within containerisation and how these aspects relate to DevOps.
- How company culture has an impact on DevOps.

Skills

The student can:

- Implement a full DevOps pipeline in GitHub Actions from integration through delivery to deployment.
- Integrate different types of software testing automatically into the software development life cycle.
- Measure the DevOps-performance of any software project.
- Monitor their applications and use technology to revert failing changes in production.

Competencies

The student:

- Plan and automate processes to quickly deliver new features without compromising quality.
- Monitor their software solutions and implement a shift-left approach to detect and resolve bugs early in the development process.

Formal requirements for the project

An application must be developed that implements subject-specific elements from the course. A link to the application's source code on GitHub must be submitted.

Formal requirements for the written submission

The written submission is a short list of pains and gains, used as the group's own reflection on the project work.

The examination: Oral exam based on project and written submission. Time perspective for the oral exam:

- 3 minutes: Presentation of an interesting aspect of the project of the students own choice.
- 7 minutes: Exam dialogue.
- 5 minutes: Voting and grading.

Assessment:

7-point grading scale. Grading is based on the product and oral examination.