# SYLLABUS

**REGARDING THE QUALIFICATION CYCLE FROM 2023TO 2025**

**ACADEMIC YEAR 2024/2025**

## 1. BASIC COURSE/MODULE INFORMATION

| | |
|---|---|
| Course/Module title | *Elective Course 2: Microservices-Based Web Application Design* |
| Course/Module code * | |
| Faculty (name of the unit offering the field of study) | *College of Natural Sciences* |
| Name of the unit running the course | *Institute of Computer Science* |
| Field of study | *Computer Science* |
| Qualification level | 2nd degree |
| Profile | *general academic profile* |
| Study mode | *full-time studies* |
| Year and semester of studies | *year 2, semester 3* |
| Course type | *major subject to elect* |
| Language of instruction | English |
| Coordinator | Piotr Lasek, PhD Eng |
| Course instructor | Piotr Lasek, PhD Eng |

* - as agreed at the faculty

### 1.1. Learning format – number of hours and ECTS credits

| Semester (no.) | Lectures | Classes | Colloquia | Lab classes | Seminars | Practical classes | Internships | others | **ECTS credits** |
|---|---|---|---|---|---|---|---|---|---|
| 3 | 15 | | | 30 | | | | | 4 |

### 1.2. Course delivery methods

conducted in a traditional way

### 1.3. Course/Module assessment

pass with a grade

## 2. PREREQUISITES

| |
|---|
| Web applications, knowledge of selected technology for the implementation of web applications (e.g., Java, .Net) |

## 3. OBJECTIVES, LEARNING OUTCOMES, COURSE CONTENT, AND INSTRUCTIONAL METHODS

### 3.1. Course/Module objectives

| | |
|---|---|
| O1 | Familiarization with the basic concepts, characteristics, and advantages of web system architectures based on microservices, such as independent deployment, flexibility, security, monitoring. |
| O2 | Mastering knowledge and developing skills that allow for modeling microservices using the Domain Driven Design technique. |
| O3 | Familiarization with the methods of transitioning from classic monolithic applications to monolithic architecture. |
| O4 | Familiarization with patterns related to workflow (orchestration, choreography, saga pattern) and the role of continuous integration and continuous delivery in the context of microservices applications. |

### 3.2. Course/Module learning outcomes

| Learning Outcome | The description of the learning outcome defined for the course/module | Relation to the degree programme outcomes |
|---|---|---|
| LO_01 | The student knows the basic concepts and problems associated with creating software in a microservices architecture. | K_W01 |
| LO_02 | The student knows and correctly identifies technologies and tools for creating systems based on microservices architecture. | K_W02 |
| LO_03 | The student is able to correctly implement a specified range of system functionalities in the form of a microservice, using various sources of documentation. | K_U01, K_U02 |
| LO_04 | The student is able to design a system consisting of several microservices and is able to communicate all the basic components of such a system (gateway, microservices registry, microservices, and others). | K_W01, K_U02, K_K01 |
| LO_05 | The student understands the challenges and the need for constant tracking of the latest achievements with microservices, which can be used to create modern and scalable web systems. | K_W03, K_K01 |

### 3.3. Course content

A. Lectures

| |
|---|
| 1. Key concepts related to microservices (independent deployment, size, flexibility, adaptation to team organization) |
| 2. The role and advantages of monolithic architecture (single-processor monoliths, modular, distributed) |
| 3. Characteristics of microservices architectures (security, monitoring, data consistency, testing) |
| 4. Defining microservice boundaries (cohesion, coupling) |

| |
|---|
| 5. Domain-Driven Design (Domain Driven Design, Aggregates, shared models, Event Storming) |
| 6. Design patterns Do not Repeat Yourself, serialization formats, contract (contract changes, versioning, early detection of changes) |
| 7. Migrating from a monolith to microservices |
| 8. Methods of microservices communication (synchronous and asynchronous communication, queues and message brokers) |
| 9. Workflow (orchestration and choreography, role of database transactions, Saga pattern) |
| 10. The role of continuous integration and continuous delivery (source code management, Monorepo and Multirepo pattern, automatic microservices deployment) |
| 11. Integrating microservices (database integration, choreography and orchestration, Postel's law, integration with external software, Strangler pattern) |
| 12. User interface models in the context of systems based on microservices (monolithic frontend, micro-frontends, aggregating gateway pattern, BFF pattern) |
| 13. Containerization of microservices (physical machines, virtual machines, containers, PaaS, SaaS) |
| 14. Introduction to Kubernetes and container orchestration |

B. Laboratory classes

| |
|---|
| 1. Creating a basic microservice (defining API, ports, data storage) |
| 2. Data persistence and applying caching mechanisms using a selected database engine and a caching memory management framework (e.g., Caffeide) |
| 3. Service discovery and registration using the Eureka server |
| 4. Configuration and launch of the API Gateway |
| 5. Communication between microservices - using the RabbitMQ message broker. |
| 6. Containerization of microservices using the Docker engine. |

### 3.4. Methods of instruction

Lecture: a lecture supported by a multimedia presentation
Laboratory classes: individual and group work designing and conducting experiments using sample libraries

## 4. Assessment Techniques and Criteria

### 4.1. Methods of evaluating learning outcomes

| Learning outcome | Methods of assessment of learning outcomes (e.g. test, oral exam, written exam, project, report, observation during classes) | Learning format (lectures, classes,...) |
|---|---|---|
| LO-01 | Test | lecture |
| LO-02 | Test, observation during classes | lecture |
| LO-03 | Colloquium, observation during classes | lab |
| LO-04 | Colloquium, observation during classes | lab |
| LO-05 | Test | lecture |

### 4.2. Course assessment criteria

Lecture - knowledge test:
Effect LO_01, LO_02, LO_05
By solving the final test, students demonstrate that they know the basic concepts and problems associated with creating software in a microservice architecture and identifies technologies and tools for creating systems based on this architecture. It also shows that they understand the challenges and the need for constant tracking of the latest achievements with microservices, which can be used to create modern and scalable web systems.
Students can earn additional points based on homework assignments that may be given during lectures. These points can be added to the final test result, increasing the number of collected points.
Laboratory:
Effect LO_03, LO_04
During the laboratory classes, students construct a small system utilizing a microservice architecture. In the final sessions, students develop its documentation, which, together with the source code, serves as the foundation for passing. This process verifies that students can accurately implement the specified scope of system functionalities within a microservice architecture.
For the assessment, the instructor evaluates several key elements:

1. The completeness, clarity, and quality of the source code: Each student is expected to implement at least two distinct microservices and all other necessary components of the system, which can contribute up to 60 points.
2. The quality of the documentation, assessed based on its completeness, clarity, and readability, contributing up to 40 points.

The final laboratory grade is assigned in accordance with a grading table, given below.

| Achieved result [%] | Grade |
|---|---|
| 0 - 49 | F (2.0) |
| 50 - 59 | E (3.0) |
| 60 - 69 | D (3.5) |
| 70 - 79 | C (4.0) |
| 80 - 89 | B (4.5) |
| 90 - 100 | A (5.0) |

## 5. TOTAL STUDENT WORKLOAD NEEDED TO ACHIEVE THE INTENDED LEARNING OUTCOMES – NUMBER OF HOURS AND ECTS CREDITS

| Activity | Number of hours |
|---|---|
| Scheduled course contact hours | 45 |
| Other contact hours involving the teacher (consultation hours) | 3 |
| Non-contact hours - student's own work (preparation for classes or examinations, projects, etc.) | 55 |
| Total number of hours | 103 |
| Total number of ECTS credits | 4 |

* One ECTS point corresponds to 25-30 hours of total student workload

## 6. INTERNSHIPS RELATED TO THE COURSE/MODULE

| | |
|---|---|
| Number of hours | ------ |
| Internship regulations and procedures | ------ |

## 7. INSTRUCTIONAL MATERIALS

Compulsory literature:

1. Sam Newman. 2015. Building Microservices (1st. ed.). O'Reilly Media, Inc.

Complementary literature:

1. Ronnie Mitra, Irakli Nadareishvili. 2020. Microservices: Up and Running: A Step-By-Step Guide to Building a Microservices Architecture, O'Reilly Media, Inc.
2. Chris Richardson. 2019. Microservice Patterns: With examples in Java, Manning Publications

Approved by the Head of the Department or an authorised person