COURSE DESCRIPTION CARD - SYLLABUS

Course name
Virtualization tools [S1MiKC1E>NW]

Course

Field of study
Microelectronics and Digital Communication

Year/Semester
3/5

Area of study (specialization)
–

Profile of study
general academic

Level of study
first-cycle

Course offered in
English

Form of study
full-time

Requirements
elective

Number of hours

Lecture
15

Laboratory classes
0

Other
0

Tutorials
0

Projects/seminars
15

Number of credit points

2,00

Coordinators

dr inż. Łukasz Kułacz
lukasz.kulacz@put.poznan.pl

Lecturers

Prerequisites

The student should have a basic knowledge of programming, especially the concepts of variables, classes and functions. He should have the ability to implement simple programs and web applications.

Course objective

The purpose of the course is to provide students with basic knowledge of vitualization and containerization. Through virtualization, the student can isolate the designed application or service from the host operating system, so that it is possible to ensure that the code runs independently of hardware, system versions and drivers. In addition, the course will present issues related to containerization, primarily, in the context of scalability and independence of applications designed as so-called microservices.

Course-related learning outcomes

Knowledge:
The student has basic theoretical and practical knowledge of software virtualization, containerization and container orchestration. The student knows the basic tools to create, use and manage containers. The student knows the features and limitations of using containers to deploy a finished solution.

Skills:
The student is able to create, use and manage virtual operating systems. The student is able to prepare simple applications and then place them in containers, run and update them. The student is able to combine several microservices into a working system and swap system components without starting the whole system from scratch.

Social competences:
The student is aware of the opportunities and limitations associated with developing software and placing it inside containers. Understands the potential gains associated with scalability and increased reliability of the system while recognizing the risks associated with using such a solution.

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

In terms of the project, the verification of the established learning outcomes is carried out by: substantive assessment of the implementation of the individual project. The project consists in the realization of an application built from several interconnected microservices. Each element of the application is assigned an appropriate number of points. The combination of independent elements of the application is also scored.
In terms of the lecture, verification of the established learning outcomes is carried out through a written assessment in the form of a test.
Passing (applies to both parts) requires obtaining at least 50% of the points. The adopted grading scale is:
2.0 <0%; 50%>
3.0 (50%; 60%>
3.5 (60%; 70%>
4.0 (70%; 80%>
4.5 (80%; 90%>
5.0 (90%; 100%>

Programme content

The course will introduce the principles of operating system virtualization, the principles of containerization, the concept of microservices, and selected issues related to container orchestration.

Course topics

Lectures:
1. Principles of virtualization of operating systems. (1 class unit)
2. The concept of cotenery. Docker - the concept of image and container. (2 class units)
3. Comparison of monolith and microservices concepts. (1 class unit)
4. Fundamentals of container orchestration. Kubernetes - container management. (1 class unit)
5. Creating a service. Docker compose and Helm tools - combining multiple containers to create a service. (1 class unit)
6. integration of microservices. (1 class unit)
Project:
Preparation (in a group of several people) of a project on a selected topic realized with the help of virtualization and containerization tools.

Teaching methods

Lecture: multimedia presentation, supplemented by examples and additional explanations based on code snippets.
Project: working at the computer, completing an individual task. As part of the course, students have the opportunity to get guidance and assistance from the tutor in the implementation of the assigned task. In addition, also outside of class students can take advantage of additional consultations.

Bibliography

Basic:
1. Rozwijanie mikrousług w Pythonie. Budowa, testowanie, instalacja i skalowanie. Tarek Ziade

2. Architektura aplikacji w Pythonie. TDD, DDD i rozwój mikrousług reaktywnych. Harry Percival
3. Nauka Dockera w miesiąc. Elton Stoneman
4. Nauka Kubernetesa w miesiąc. Elton Stoneman

Additional:
1. Docker. Niezawodne kontenery produkcyjne. Praktyczne zastosowania. Sean Kane

Breakdown of average student's workload

	Hours	ECTS
Total workload	60	2,00
Classes requiring direct contact with the teacher	30	1,00
Student's own work (literature studies, preparation for laboratory classes/ tutorials, preparation for tests/exam, project preparation)	30	1,00