Akademia Humanistyczno Ekonomiczna w Łodzi

2026-2027

ACADEMIC OFFER

COMPUTER SCIENCE

COMPUTER SCIENCE

Language: **English** | Spring **2026-2027** | **Computer Science Bachelor**

1ST YEAR / 2ND SEMESTER

ZDW: Operating Systems/Systems for managing computer hardware and applications	lecture
ZDW: Operating Systems/Systems for managing computer hardware and applications	workshops
ZDW: Operating Systems/Systems for managing computer hardware and applications	project
Numerical Methods in Computer Science	lecture
Numerical Methods in Computer Science	exercises
Numerical Methods in Computer Science	project
Discrete Mathematics	lecture
Discrete Mathematics	project
Basics of Programming II	lecture
Basics of Programming II	exercises
Basics of Programming II	project
ZDW: Intellectual Property Protection	lecture
Communication and Relations building	workshops

COMPUTER SCIENCE

Language: **English** | Spring **2026-2027** | **Computer Science Bachelor**

2ND YEAR / 4TH SEMESTER

ZDW: Basics of Management Information Systems/ Information Management Technology	lecture
ZDW: Basics of Management Information Systems/ Information Management Technology	project
ZDW: Data protection/Data safety	lecture
ZDW: Data protection/Data safety	project
Probability and statistics	lecture
Probability and statistics	
Object-oriented programming 1	workshops
Object-oriented programming 1	project

3RD YEAR / 6TH SEMESTER

ZDW: Embedded Systems / Design of Special-Purpose Systems	lecture
ZDW: Embedded Systems / Design of Special-Purpose Systems	workshops
ZDW: Embedded Systems / Design of Special-Purpose Systems	project
Advanced programming technologies	lecture
Advanced programming technologies	project
Team project	lecture
Team project	project
Electronic Transaction Systems	lecture
Electronic Transaction Systems	project
Conducting IT Projects	lecture
Conducting IT Projects	project

ZDW: Operating Systems/Systems for managing computer hardware and applications

lecture

2 ECTS

Teaching methods	demonstrations / tutorials / practical exercises / individual work
Method of verifying education	colloquium / assignments / activity during classes

OBJECTIVES

The aim of the course is to is to equip students with fundamental knowledge and practical skills in the effective use of office applications (word processor, spreadsheet, and presentation software), as well as to develop an understanding of the role of information technologies and multimedia in everyday life and professional work.

COURSE CONTENT

- Using IT tools and applications in education. Editing text documents in MS Word: typing, editing, proofreading, autocorrect, formatting, inserting objects into text, creating lists and tables, working with headers, sections, and page numbering, print preview. Working with multi-page documents: footnotes, bookmarks, hyperlinks, tables of contents, bibliographies, indexes, etc.
- Using IT tools and applications in education – creating multimedia presentations (PowerPoint and Prezi): principles of presentation design; using graphics, audio, and animations; adding hyperlinks and charts; slide masters and templates; organizing and running a slideshow; setting up automatic presentations; saving presentations in various formats.
- Using IT tools and applications in education – creating spreadsheets: data types, operators, arithmetic, logical and text expressions, function arguments and function values, extracting parameters in task solutions, referencing methods (including cross-sheet referencing), formulas, built-in functions, autofill, formatting cells and ranges, XY charts. Spreadsheets as simple databases: forms, searching, filtering, and multi-level sorting.
- Protection of intellectual property in IT in education: creating, sharing, and using source materials; types of licenses.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of selected functionalities of office tools (word processor, spreadsheet, presentation software) and their applications in data analysis and document preparation.
- Has advanced knowledge and understanding of selected concepts and issues related to information technologies, multimedia, and media convergence, as well as the role of IT in work organization. Possesses basic knowledge of different types of licenses and the principles of intellectual property protection in the creation and use of materials in the context of IT.

In terms of skills:

- Is able to independently use advanced features of a word processor to create complex documents (e.g., engineering reports), applying styles, tables of contents, footnotes, and embedded objects.
- Is able to effectively design multimedia presentations, applying principles of visual design, graphics, audio, animations, and hyperlinks, as well as adapting presentations to different formats.
- Is able to consciously select and apply advanced spreadsheet functions (formulas, built-in functions, referencing methods, charts, filtering, sorting) for data analysis, data processing, and problem-solving.
- Is able to independently search for information and expand their knowledge using various sources, including academic publications, online resources (netography), tutorials, and other professional materials.

In terms of social competencies:

- Is ready to critically evaluate their own knowledge and continuously expand it.
- Is prepared to uphold high ethical standards and respects the principles of intellectual property protection when using source materials and applying licenses in work with IT applications.
- Ensures a professional standard in the preparation of digital materials and takes responsibility for the accuracy and quality of their content.

ZDW: Operating Systems/Systems for managing computer hardware and applications

workshops

2 ECTS

Teaching methods	demonstrations / tutorials / practical exercises / individual work
Method of verifying education	colloquium / assignments / activity during classes

OBJECTIVES

The aim of the course is to is to equip students with fundamental knowledge and practical skills in the effective use of office applications (word processor, spreadsheet, and presentation software), as well as to develop an understanding of the role of information technologies and multimedia in everyday life and professional work.

COURSE CONTENT

- Using IT tools and applications in education. Editing text documents in MS Word: typing, editing, proofreading, autocorrect, formatting, inserting objects into text, creating lists and tables, working with headers, sections, and page numbering, print preview. Working with multi-page documents: footnotes, bookmarks, hyperlinks, tables of contents, bibliographies, indexes, etc.
- Using IT tools and applications in education – creating multimedia presentations (PowerPoint and Prezi): principles of presentation design; using graphics, audio, and animations; adding hyperlinks and charts; slide masters and templates; organizing and running a slideshow; setting up automatic presentations; saving presentations in various formats.
- Using IT tools and applications in education – creating spreadsheets: data types, operators, arithmetic, logical and text expressions, function arguments and function values, extracting parameters in task solutions, referencing methods (including cross-sheet referencing), formulas, built-in functions, autofill, formatting cells and ranges, XY charts. Spreadsheets as simple databases: forms, searching, filtering, and multi-level sorting.
- Protection of intellectual property in IT in education: creating, sharing, and using source materials; types of licenses.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of selected functionalities of office tools (word processor, spreadsheet, presentation software) and their applications in data analysis and document preparation.
- Has advanced knowledge and understanding of selected concepts and issues related to information technologies, multimedia, and media convergence, as well as the role of IT in work organization. Possesses basic knowledge of different types of licenses and the principles of intellectual property protection in the creation and use of materials in the context of IT.

In terms of skills:

- Is able to independently use advanced features of a word processor to create complex documents (e.g., engineering reports), applying styles, tables of contents, footnotes, and embedded objects.
- Is able to effectively design multimedia presentations, applying principles of visual design, graphics, audio, animations, and hyperlinks, as well as adapting presentations to different formats.
- Is able to consciously select and apply advanced spreadsheet functions (formulas, built-in functions, referencing methods, charts, filtering, sorting) for data analysis, data processing, and problem-solving.
- Is able to independently search for information and expand their knowledge using various sources, including academic publications, online resources (netography), tutorials, and other professional materials.

In terms of social competencies:

- Is ready to critically evaluate their own knowledge and continuously expand it.
- Is prepared to uphold high ethical standards and respects the principles of intellectual property protection when using source materials and applying licenses in work with IT applications.
- Ensures a professional standard in the preparation of digital materials and takes responsibility for the accuracy and quality of their content.

ZDW: Operating Systems/Systems for managing computer hardware and applications

project

3 ECTS

Teaching methods	demonstrations / tutorials / practical exercises / individual work
Method of verifying education	colloquium / assignments / activity during classes

OBJECTIVES

The aim of the course is to is to equip students with fundamental knowledge and practical skills in the effective use of office applications (word processor, spreadsheet, and presentation software), as well as to develop an understanding of the role of information technologies and multimedia in everyday life and professional work.

COURSE CONTENT

- Using IT tools and applications in education. Editing text documents in MS Word: typing, editing, proofreading, autocorrect, formatting, inserting objects into text, creating lists and tables, working with headers, sections, and page numbering, print preview. Working with multi-page documents: footnotes, bookmarks, hyperlinks, tables of contents, bibliographies, indexes, etc.
- Using IT tools and applications in education – creating multimedia presentations (PowerPoint and Prezi): principles of presentation design; using graphics, audio, and animations; adding hyperlinks and charts; slide masters and templates; organizing and running a slideshow; setting up automatic presentations; saving presentations in various formats.
- Using IT tools and applications in education – creating spreadsheets: data types, operators, arithmetic, logical and text expressions, function arguments and function values, extracting parameters in task solutions, referencing methods (including cross-sheet referencing), formulas, built-in functions, autofill, formatting cells and ranges, XY charts. Spreadsheets as simple databases: forms, searching, filtering, and multi-level sorting.
- Protection of intellectual property in IT in education: creating, sharing, and using source materials; types of licenses.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of selected functionalities of office tools (word processor, spreadsheet, presentation software) and their applications in data analysis and document preparation.
- Has advanced knowledge and understanding of selected concepts and issues related to information technologies, multimedia, and media convergence, as well as the role of IT in work organization. Possesses basic knowledge of different types of licenses and the principles of intellectual property protection in the creation and use of materials in the context of IT.

In terms of skills:

- Is able to independently use advanced features of a word processor to create complex documents (e.g., engineering reports), applying styles, tables of contents, footnotes, and embedded objects.
- Is able to effectively design multimedia presentations, applying principles of visual design, graphics, audio, animations, and hyperlinks, as well as adapting presentations to different formats.
- Is able to consciously select and apply advanced spreadsheet functions (formulas, built-in functions, referencing methods, charts, filtering, sorting) for data analysis, data processing, and problem-solving.
- Is able to independently search for information and expand their knowledge using various sources, including academic publications, online resources (netography), tutorials, and other professional materials.

In terms of social competencies:

- Is ready to critically evaluate their own knowledge and continuously expand it.
- Is prepared to uphold high ethical standards and respects the principles of intellectual property protection when using source materials and applying licenses in work with IT applications.
- Ensures a professional standard in the preparation of digital materials and takes responsibility for the accuracy and quality of their content.

Numerical Methods in Computer Science

lecture

2 ECTS

Teaching methods	lecture / conversational lecture
Method of verifying education	exam / assignments / activity during classes

OBJECTIVES

Familiarizing students with numerical methods used in solving computational problems, forming a foundation for further study in scientific computing and simulation. The course aims to develop the ability to implement numerical algorithms, analyze their accuracy, and apply them to practical tasks such as solving equations or approximating functions.

COURSE CONTENT

- Calculations of lengths, areas of plane figures, and volumes of solids of revolution.
- Factorization techniques.
- Iterative methods of approximation, interpolation, and extrapolation.
- Mathematical formulation of optimization problems: criteria, objective functions, classification.
- Solving first-order differential equations.
- Numerical series – calculation of derivative values at a point, the number π, factorial, Napier's number (e). Numerical integration methods.
- Optimization methods for multidimensional nonlinear problems with constraints: gradient descent with backtracking, penalty functions.
- Calculations of lengths, areas of plane figures, and volumes of solids of revolution.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of calculating lengths, areas of plane figures, and volumes of solids of revolution using Monte Carlo methods and the least squares method.
- Has advanced knowledge and understanding of the mathematical formulation of approximation, interpolation, and extrapolation problems.
- Has advanced knowledge and understanding of the principles of operation and computational complexity of optimization algorithms, as well as algorithmic computations using Lagrange and Euler methods and Chebyshev methods in linear interpolation..

In terms of skills:

- Is able to implement and apply numerical methods for solving systems of linear equations and first-order differential equations.
- Is able to implement and apply optimization algorithms for counting and data processing tasks.
- Is able to analyze, formulate, and classify optimization problems in mathematical form, selecting appropriate numerical methods.

In terms of social competencies:

- Demonstrates rigor in source code and responsibility in the numerical analysis of the stability, convergence, and complexity of implemented optimization algorithms.

Numerical Methods in Computer Science

exercises

2 ECTS

Teaching methods	problem-solving method / individual and group work
Method of verifying education	assignments / activity during classes

OBJECTIVES

Familiarizing students with numerical methods used in solving computational problems, forming a foundation for further study in scientific computing and simulation. The course aims to develop the ability to implement numerical algorithms, analyze their accuracy, and apply them to practical tasks such as solving equations or approximating functions.

COURSE CONTENT

- Calculations of lengths, areas of plane figures, and volumes of solids of revolution.
- Factorization techniques.
- Iterative methods of approximation, interpolation, and extrapolation.
- Mathematical formulation of optimization problems: criteria, objective functions, classification.
- Solving first-order differential equations.
- Numerical series – calculation of derivative values at a point, the number π, factorial, Napier's number (e). Numerical integration methods.
- Optimization methods for multidimensional nonlinear problems with constraints: gradient descent with backtracking, penalty functions.
- Calculations of lengths, areas of plane figures, and volumes of solids of revolution.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of calculating lengths, areas of plane figures, and volumes of solids of revolution using Monte Carlo methods and the least squares method.
- Has advanced knowledge and understanding of the mathematical formulation of approximation, interpolation, and extrapolation problems.
- Has advanced knowledge and understanding of the principles of operation and computational complexity of optimization algorithms, as well as algorithmic computations using Lagrange and Euler methods and Chebyshev methods in linear interpolation..

In terms of skills:

- Is able to implement and apply numerical methods for solving systems of linear equations and first-order differential equations.
- Is able to implement and apply optimization algorithms for counting and data processing tasks.
- Is able to analyze, formulate, and classify optimization problems in mathematical form, selecting appropriate numerical methods.

In terms of social competencies:

- Demonstrates rigor in source code and responsibility in the numerical analysis of the stability, convergence, and complexity of implemented optimization algorithms.

Numerical Methods in Computer Science

project

2 ECTS

Teaching methods	workshop method / problem-solving method
Method of verifying education	project / activity during classes

OBJECTIVES

Familiarizing students with numerical methods used in solving computational problems, forming a foundation for further study in scientific computing and simulation. The course aims to develop the ability to implement numerical algorithms, analyze their accuracy, and apply them to practical tasks such as solving equations or approximating functions.

COURSE CONTENT

- Calculations of lengths, areas of plane figures, and volumes of solids of revolution.
- Factorization techniques.
- Iterative methods of approximation, interpolation, and extrapolation.
- Mathematical formulation of optimization problems: criteria, objective functions, classification.
- Solving first-order differential equations.
- Numerical series – calculation of derivative values at a point, the number π, factorial, Napier's number (e). Numerical integration methods.
- Optimization methods for multidimensional nonlinear problems with constraints: gradient descent with backtracking, penalty functions.
- Calculations of lengths, areas of plane figures, and volumes of solids of revolution.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of calculating lengths, areas of plane figures, and volumes of solids of revolution using Monte Carlo methods and the least squares method.
- Has advanced knowledge and understanding of the mathematical formulation of approximation, interpolation, and extrapolation problems.
- Has advanced knowledge and understanding of the principles of operation and computational complexity of optimization algorithms, as well as algorithmic computations using Lagrange and Euler methods and Chebyshev methods in linear interpolation..

In terms of skills:

- Is able to implement and apply numerical methods for solving systems of linear equations and first-order differential equations.
- Is able to implement and apply optimization algorithms for counting and data processing tasks.
- Is able to analyze, formulate, and classify optimization problems in mathematical form, selecting appropriate numerical methods.

In terms of social competencies:

- Demonstrates rigor in source code and responsibility in the numerical analysis of the stability, convergence, and complexity of implemented optimization algorithms.

Discrete Mathematics

lecture

3 ECTS

Teaching methods	lecture / conversational lecture
Method of verifying education	exam / assignments / activity during classes

OBJECTIVES

Familiarizing students with discrete mathematics, which forms the foundation for algorithms, graph theory, cryptography in computer science, and mathematical induction. The course aims to develop the ability to analyze discrete structures such as classical logic, sequences, sets, relations, and graphs, as well as to introduce binary algebra, binary logic, and binary graphs. Additionally, it aims to develop analytical and creative mathematical thinking and the ability to translate advanced mathematical structures into algorithmic structures.

COURSE CONTENT

- Introduction to discrete mathematics and its applications in computer science.
- Basics of sets, numerical sequences, set operations and relations, Venn diagrams.
- Functions and their properties in a discrete context.
- Fundamentals of logic: operators, tautologies, inference.
- Combinatorics: permutations, combinations, variations, Stirling numbers of the first and second kind, the inclusion–exclusion principle.
- Introduction to graph theory: definitions, types of graphs, graph formulas, graph representations.
- Trees and their applications in computer science.
- Practical exercises in implementing discrete structures.
- Applications of discrete mathematics in algorithms and cryptography.
- Introduction to discrete mathematics and its applications in computer science.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of selected topics in discrete mathematics, including classical logic, sets, relations, and functions.
- Has knowledge and understanding of methods for representing numerical sequences (e.g., finite sequences of prime numbers) and algebraic sorting of weakly monotonic sequences.
- Has advanced knowledge and understanding of the theory and formulas of planar and spatial graphs and their applications in computer science.
- Has advanced knowledge and understanding of the principles of logic, combinatorics, and RSA encryption.
- Has advanced knowledge and understanding of the principles of calculating and applying Stirling numbers of the first and second kind.
- Has advanced knowledge and understanding of the theory of adjacency and incidence matrices in graph theory.

In terms of skills:

- Is able to analyze and model computational problems using discrete structures.
- Is able to apply graph theory to solve practical problems.
- Is able to implement discrete algorithms in a selected programming language.
- Is able to independently search for information and expand their knowledge using various sources, including academic publications, online resources, tutorials, etc.
- Is able to independently formulate and analyze problems, as well as propose solutions to less complex problems in predictable conditions.

In terms of social competencies:

- Is ready to critically evaluate and expand their knowledge and understands the need for continuous improvement of mathematical skills in computer science.
- Is prepared to maintain high ethical standards in professional work and engineering practice.
- Is ready to take responsibility for tasks undertaken in professional activity.

Discrete Mathematics

project

4 ECTS

Teaching methods	project method / problem-solving method / individual and group work
Method of verifying education	project / assignments / activity during classes

OBJECTIVES

Familiarizing students with discrete mathematics, which forms the foundation for algorithms, graph theory, cryptography in computer science, and mathematical induction. The course aims to develop the ability to analyze discrete structures such as classical logic, sequences, sets, relations, and graphs, as well as to introduce binary algebra, binary logic, and binary graphs. Additionally, it aims to develop analytical and creative mathematical thinking and the ability to translate advanced mathematical structures into algorithmic structures.

COURSE CONTENT

- Introduction to discrete mathematics and its applications in computer science.
- Basics of sets, numerical sequences, set operations and relations, Venn diagrams.
- Functions and their properties in a discrete context.
- Fundamentals of logic: operators, tautologies, inference.
- Combinatorics: permutations, combinations, variations, Stirling numbers of the first and second kind, the inclusion–exclusion principle.
- Introduction to graph theory: definitions, types of graphs, graph formulas, graph representations.
- Trees and their applications in computer science.
- Practical exercises in implementing discrete structures.
- Applications of discrete mathematics in algorithms and cryptography.
- Introduction to discrete mathematics and its applications in computer science.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of selected topics in discrete mathematics, including classical logic, sets, relations, and functions.
- Has knowledge and understanding of methods for representing numerical sequences (e.g., finite sequences of prime numbers) and algebraic sorting of weakly monotonic sequences.
- Has advanced knowledge and understanding of the theory and formulas of planar and spatial graphs and their applications in computer science.
- Has advanced knowledge and understanding of the principles of logic, combinatorics, and RSA encryption.
- Has advanced knowledge and understanding of the principles of calculating and applying Stirling numbers of the first and second kind.
- Has advanced knowledge and understanding of the theory of adjacency and incidence matrices in graph theory.

In terms of skills:

- Is able to analyze and model computational problems using discrete structures.
- Is able to apply graph theory to solve practical problems.
- Is able to implement discrete algorithms in a selected programming language.
- Is able to independently search for information and expand their knowledge using various sources, including academic publications, online resources, tutorials, etc.
- Is able to independently formulate and analyze problems, as well as propose solutions to less complex problems in predictable conditions.

In terms of social competencies:

- Is ready to critically evaluate and expand their knowledge and understands the need for continuous improvement of mathematical skills in computer science.
- Is prepared to maintain high ethical standards in professional work and engineering practice.
- Is ready to take responsibility for tasks undertaken in professional activity.

Basics of programming 2

lecture

2 ECTS

Teaching methods	lecture / conversational lecture
Method of verifying education	colloquium / assignments / activity during classes

OBJECTIVES

The course introduces students to the Python programming language, with an emphasis on functional and object-oriented paradigms. It aims to develop students' skills in writing more complex programs, working with Python libraries, designing data structures and algorithms, and preparing for practical applications in computer science. An important component of the course is fostering awareness of effective use of documentation and online resources, as well as preparing students for more advanced courses, such as application programming, in subsequent semesters.

COURSE CONTENT

- Introduction to Python and its paradigms (functional, object-oriented)
- Python syntax basics: variables, data types, functions
- Working with Python standard libraries (e.g., math, os)
- Advanced data structures in Python (lists, dictionaries, sets)
- Implementation of algorithms in Python (sorting, searching)
- Object-oriented programming in Python (classes, inheritance)
- File handling and exception management in Python
- Practical exercises with external libraries (e.g., NumPy, Pandas)
- Project: solving a computing problem using Python
- Code optimization and debugging in Python

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of the Python language, including its syntax and object-oriented and functional paradigms
- Has advanced knowledge and understanding of Python's standard libraries and their applications
- Has advanced knowledge and understanding of data structures and algorithms in Python

In terms of skills:

- Can write, debug, and optimize programs in Python
- Can design and implement advanced data structures using Python
- Can use Python libraries to solve computing problems
- Can independently seek information and expand knowledge using various sources, including scientific publications, online resources, and tutorials
- Can independently formulate and analyze problems, as well as propose solutions to uncomplicated problems in predictable conditions

In terms of social competencies:

- Is prepared to critically evaluate their own knowledge and work on its expansion
- Is prepared to uphold high ethical standards in professional work, particularly in engineering practice
- Is prepared to take responsibility when performing tasks in professional activities

Basics of programming 2

exercises

2 ECTS

Teaching methods	workshop method / individual work
Method of verifying education	assignments / activity during classes

OBJECTIVES

The course introduces students to the Python programming language, with an emphasis on functional and object-oriented paradigms. It aims to develop students' skills in writing more complex programs, working with Python libraries, designing data structures and algorithms, and preparing for practical applications in computer science. An important component of the course is fostering awareness of effective use of documentation and online resources, as well as preparing students for more advanced courses, such as application programming, in subsequent semesters.

COURSE CONTENT

- Introduction to Python and its paradigms (functional, object-oriented)
- Python syntax basics: variables, data types, functions
- Working with Python standard libraries (e.g., math, os)
- Advanced data structures in Python (lists, dictionaries, sets)
- Implementation of algorithms in Python (sorting, searching)
- Object-oriented programming in Python (classes, inheritance)
- File handling and exception management in Python
- Practical exercises with external libraries (e.g., NumPy, Pandas)
- Project: solving a computing problem using Python
- Code optimization and debugging in Python

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of the Python language, including its syntax and object-oriented and functional paradigms
- Has advanced knowledge and understanding of Python's standard libraries and their applications
- Has advanced knowledge and understanding of data structures and algorithms in Python

In terms of skills:

- Can write, debug, and optimize programs in Python
- Can design and implement advanced data structures using Python
- Can use Python libraries to solve computing problems
- Can independently seek information and expand knowledge using various sources, including scientific publications, online resources, and tutorials
- Can independently formulate and analyze problems, as well as propose solutions to uncomplicated problems in predictable conditions

In terms of social competencies:

- Is prepared to critically evaluate their own knowledge and work on its expansion
- Is prepared to uphold high ethical standards in professional work, particularly in engineering practice
- Is prepared to take responsibility when performing tasks in professional activities

Basics of programming 2

project

2 ECTS

Teaching methods	project method / individual work
Method of verifying education	project / assignments / activity during classes

OBJECTIVES

The course introduces students to the Python programming language, with an emphasis on functional and object-oriented paradigms. It aims to develop students' skills in writing more complex programs, working with Python libraries, designing data structures and algorithms, and preparing for practical applications in computer science. An important component of the course is fostering awareness of effective use of documentation and online resources, as well as preparing students for more advanced courses, such as application programming, in subsequent semesters.

COURSE CONTENT

- Introduction to Python and its paradigms (functional, object-oriented)
- Python syntax basics: variables, data types, functions
- Working with Python standard libraries (e.g., math, os)
- Advanced data structures in Python (lists, dictionaries, sets)
- Implementation of algorithms in Python (sorting, searching)
- Object-oriented programming in Python (classes, inheritance)
- File handling and exception management in Python
- Practical exercises with external libraries (e.g., NumPy, Pandas)
- Project: solving a computing problem using Python
- Code optimization and debugging in Python

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of the Python language, including its syntax and object-oriented and functional paradigms
- Has advanced knowledge and understanding of Python's standard libraries and their applications
- Has advanced knowledge and understanding of data structures and algorithms in Python

In terms of skills:

- Can write, debug, and optimize programs in Python
- Can design and implement advanced data structures using Python
- Can use Python libraries to solve computing problems
- Can independently seek information and expand knowledge using various sources, including scientific publications, online resources, and tutorials
- Can independently formulate and analyze problems, as well as propose solutions to uncomplicated problems in predictable conditions

In terms of social competencies:

- Is prepared to critically evaluate their own knowledge and work on its expansion
- Is prepared to uphold high ethical standards in professional work, particularly in engineering practice
- Is prepared to take responsibility when performing tasks in professional activities

ZDW: Intellectual Property Protection

lecture

1 ECTS

Teaching methods	lecture
Method of veryfying education	exam

OBJECTIVES

Students will learn the concept of intellectual property. The course also covers issues: the role and importance of intellectual property in education, commercial activities and research, and the general scope of protection by exclusive rights, including time and territory. Students will also learn about intellectual property categories and exhaustion of intellectual property rights. Students will learn international treaties and other legal provisions.

COURSE CONTENT

- The historical development of intangible goods protection.
- International and national aspects of intellectual property protection.
- The origins and position of contemporary copyright law and related rights.
- The relationship between intellectual property protection, competition policy, unemployment reduction, innovation, and economic growth.
- The subject matter and entities of copyright law; basic definitions.
- Authors' moral rights to protected works.
- The concept and basic catalogue of economic rights and fields of exploitation of a work; selected issues concerning licenses.
- Forms of infringement of moral and economic copyright: the concepts of plagiarism, piracy, and databases; the role of collective copyright management organizations.
- The concept and principles of permissible private and public use of a work; the rights of libraries and schools; the quotation right.
- Special protection of computer programs, images, and correspondence.
- Protection of inventions, trademarks, and industrial designs; the Community trademark.
- Civil and criminal liability rules for infringement of intellectual property rights.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Students know and understand concepts in the field of intellectual property, including copyright law.
- They know the relationship between intellectual property protection, fair competition, innovation, and economic growth.
- They know the principles of intellectual property protection.
- They know and understand the differences between moral rights and economic (property) copyrights.

In terms of abilities:

- Student: is able to accurately define a work and other objects of intellectual property in both legal and economic terms.
- Is able to determine which works are not protected by copyright law and provide a reasoned justification.
- Is able to select relevant information and statistical data to analyze the economic impact of intellectual property rights.

In terms of social competencies:

- Student is aware of their knowledge of the social and economic role of intellectual property protection.
- Acts in a professional manner, respecting intellectual property.
- Is able to produce academic texts and simple informational content without infringing copyright law.

COMMUNICATION AND RELATIONS BUILDING

workshops

1 ECTS

Teaching methods	lecture / didactic discussion / case study
Method of veryfying education	group and individual assgnments / project / acticity during classes

OBJECTIVES

The course will explore strategies for establishing and maintaining connections through effective communication. It covers communication tools such as naming emotions, asking questions, and active listening, techniques like paraphrasing and mirroring. The importance of being precise and clear in communication will also be emphesised. Verbal and non-verbal communication means are going to be discovered, discussed and explored together with communication barriers, including cross-cultural context. The course participants will be equipped with practical tools and insights to communicate more effectively, build rapport and trust in order to establish relationships in both personal and professional setting.

COURSE CONTENT

- Ways to effectively establish contact with another person.
- Tools for effective communication: naming feelings, using open questions, active listening: paraphrase, mirroring, precision of the message, "I" message.
- The role of verbal and non-verbal communication.
- Communication barriers.
- The role and importance of emotions in the process of communication and integration.
- Online communication.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- The student has knowledge of verbal and non-verbal communication, including methods and styles of communication as well as communication barriers.

In terms of skills :

- The student establishes and deepens relationships with the group.
- The student chooses an effective communication strategy.
- The student uses selected tools for effective communication.
- The student is able to present themselves in an appropriate way to the situation.
- The student organizes teamwork.

In terms of social competencies

- The student shows openness to solving individual and group communication problems.
- The student engages in teamwork and plays various group roles.

ZDW: Basics of Management Information Systems/ Information Management Technology

lecture

1 ECTS

Teaching methods	lecture / conversational lecture
Method of verifying education	exam / assignments / activity during classes

OBJECTIVES

The course introduces students to information systems supporting management (ERP, CRM), with an emphasis on their applications in computer science. It aims to develop students' skills in designing and implementing simple data management systems, understanding business processes, recognizing their role in decision-making, and integrating these systems with databases.

COURSE CONTENT

- Introduction and characteristics of management systems: role and significance of information systems in enterprises (ERP, CRM, SCM)
- Modularity and process-oriented approach: modularity and integration of systems (ERP) and process-oriented organization reflected in system modules
- Business process modeling: fundamentals of business processes, their role in decision-making, and modeling using tools (e.g., BPMN)
- Data architecture and integration: architecture of data management systems and integration with databases (relational, NoSQL)
- System and process design: fundamentals of designing management system infrastructures and processes; examples of methods and tools supporting design
- Implementation, deployment, and configuration: distribution models (cloud, on-premise); examples of implementations and practical project in system configuration
- Data security and ethics: data protection in management systems and ethical and legal aspects of data use

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of management information systems, including ERP and CRM, and their role within organizations
- Has advanced knowledge and understanding of business processes, their modeling, and their impact on managerial decisions
- Has advanced knowledge and understanding of data management system architecture, as well as the fundamentals of security and ethics

In terms of skills:

- Can design and configure a basic ERP/CRM system
- Can model business processes using tools such as BPMN
- Can integrate systems with databases (relational and NoSQL) and analyze business data
- Can independently seek information and expand knowledge using various sources, including scientific publications, online resources, and tutorials
- Can independently formulate and analyze problems, as well as propose solutions to uncomplicated problems in predictable conditions

In terms of social competencies:

- Is prepared to critically evaluate their own knowledge and work on its expansion
- Is prepared to uphold high ethical standards in professional work, particularly in engineering practice
- Is prepared to take responsibility when performing tasks in professional activities

ZDW: Basics of Management Information Systems/ Information Management Technology

project

2 ECTS

Teaching methods	individual and group work / project method
Method of verifying education	project / assignments / activity during classes

OBJECTIVES

The course introduces students to information systems supporting management (ERP, CRM), with an emphasis on their applications in computer science. It aims to develop students' skills in designing and implementing simple data management systems, understanding business processes, recognizing their role in decision-making, and integrating these systems with databases.

COURSE CONTENT

- Introduction and characteristics of management systems: role and significance of information systems in enterprises (ERP, CRM, SCM)
- Modularity and process-oriented approach: modularity and integration of systems (ERP) and process-oriented organization reflected in system modules
- Business process modeling: fundamentals of business processes, their role in decision-making, and modeling using tools (e.g., BPMN)
- Data architecture and integration: architecture of data management systems and integration with databases (relational, NoSQL)
- System and process design: fundamentals of designing management system infrastructures and processes; examples of methods and tools supporting design
- Implementation, deployment, and configuration: distribution models (cloud, on-premise); examples of implementations and practical project in system configuration
- Data security and ethics: data protection in management systems and ethical and legal aspects of data use

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of management information systems, including ERP and CRM, and their role within organizations
- Has advanced knowledge and understanding of business processes, their modeling, and their impact on managerial decisions
- Has advanced knowledge and understanding of data management system architecture, as well as the fundamentals of security and ethics

In terms of skills:

- Can design and configure a basic ERP/CRM system
- Can model business processes using tools such as BPMN
- Can integrate systems with databases (relational and NoSQL) and analyze business data
- Can independently seek information and expand knowledge using various sources, including scientific publications, online resources, and tutorials
- Can independently formulate and analyze problems, as well as propose solutions to uncomplicated problems in predictable conditions

In terms of social competencies:

- Is prepared to critically evaluate their own knowledge and work on its expansion
- Is prepared to uphold high ethical standards in professional work, particularly in engineering practice
- Is prepared to take responsibility when performing tasks in professional activities

ZDW: Data protection/data safety

lecture

1 ECTS

Teaching methods	lecture / conversational lecture
Method of verifying education	colloquium / assignments / activity during classes

OBJECTIVES

The course focuses on introducing students to the technical aspects of data protection in information systems. It aims to develop students' skills in implementing security measures (e.g., encryption, authentication), analyzing threats, and designing secure code.

COURSE CONTENT

- Introduction and fundamentals: introduction to technical data protection in IT, discussion of security models
- Threat and risk analysis: analysis of threats (attacks, code errors, security vulnerabilities – e.g., OWASP Top 10) and risk management, including identification and mitigation
- Cryptographic algorithms and implementation: theory and practical implementation of encryption algorithms (symmetric, asymmetric, hash functions) in programming
- Authentication and authorization: implementation of secure authentication (e.g., salted password hashing, tokens) and session management in code
- Practical cryptography labs: implementing encryption in a chosen programming language (e.g., Python with the cryptography library) and verifying implementation correctness
- Secure coding design: principles to avoid common injection flaws and vulnerabilities (e.g., SQL injection, XSS), input validation, and secure configurations
- Security testing and verification: code security testing (static analysis, scanning tools) and performing basic testing
- Project scenario: securing applications/APIs: comprehensive semester project involving auditing, refactoring, and securing an unsecured application to apply all acquired skills
- Ethics and responsibility: ethics in secure software development, engineer's responsibility for data (e.g., GDPR compliance), and the need for continuous professional development

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of technical aspects of data protection, including common threats, security vulnerabilities (e.g., SQL Injection, XSS), and methods for prevention and mitigation
- Has advanced knowledge and understanding of cryptographic algorithms and techniques (e.g., symmetric and asymmetric encryption, hash functions) as well as user authentication and authorization methods
- Has advanced knowledge and understanding of secure coding principles (Secure Coding) and security risk management methods in IT projects

In terms of skills:

- Can identify, analyze, and classify technical data threats (e.g., network attacks, social engineering) and propose appropriate countermeasures
- Can implement technical data protection measures and cryptographic mechanisms in code (e.g., encryption, hashing, authentication) using programming libraries (e.g., in Python)
- Can design and test secure code (penetration tests, security unit tests) within the software development lifecycle
- Can independently seek out and critically evaluate information sources (including security standards and scientific publications) to expand knowledge
- Can independently analyze and formulate security problems and propose appropriate solutions for uncomplicated incidents

In terms of social competencies:

- Is prepared to critically evaluate their own knowledge and expand it, understanding the need for continuous improvement of technical data protection skills
- Is prepared to uphold high ethical standards in professional work, particularly in engineering practice
- Is prepared to take responsibility when performing tasks in professional activities

ZDW: Data protection/data safety

project

2 ECTS

Teaching methods	problem-solving method / project method
Method of verifying education	project / assignments / activity during classes

OBJECTIVES

The course focuses on introducing students to the technical aspects of data protection in information systems. It aims to develop students' skills in implementing security measures (e.g., encryption, authentication), analyzing threats, and designing secure code.

COURSE CONTENT

- Introduction and fundamentals: introduction to technical data protection in IT, discussion of security models
- Threat and risk analysis: analysis of threats (attacks, code errors, security vulnerabilities – e.g., OWASP Top 10) and risk management, including identification and mitigation
- Cryptographic algorithms and implementation: theory and practical implementation of encryption algorithms (symmetric, asymmetric, hash functions) in programming
- Authentication and authorization: implementation of secure authentication (e.g., salted password hashing, tokens) and session management in code
- Practical cryptography labs: implementing encryption in a chosen programming language (e.g., Python with the cryptography library) and verifying implementation correctness
- Secure coding design: principles to avoid common injection flaws and vulnerabilities (e.g., SQL injection, XSS), input validation, and secure configurations
- Security testing and verification: code security testing (static analysis, scanning tools) and performing basic testing
- Project scenario: securing applications/APIs: comprehensive semester project involving auditing, refactoring, and securing an unsecured application to apply all acquired skills
- Ethics and responsibility: ethics in secure software development, engineer's responsibility for data (e.g., GDPR compliance), and the need for continuous professional development

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of technical aspects of data protection, including common threats, security vulnerabilities (e.g., SQL Injection, XSS), and methods for prevention and mitigation
- Has advanced knowledge and understanding of cryptographic algorithms and techniques (e.g., symmetric and asymmetric encryption, hash functions) as well as user authentication and authorization methods
- Has advanced knowledge and understanding of secure coding principles (Secure Coding) and security risk management methods in IT projects

In terms of skills:

- Can identify, analyze, and classify technical data threats (e.g., network attacks, social engineering) and propose appropriate countermeasures
- Can implement technical data protection measures and cryptographic mechanisms in code (e.g., encryption, hashing, authentication) using programming libraries (e.g., in Python)
- Can design and test secure code (penetration tests, security unit tests) within the software development lifecycle
- Can independently seek out and critically evaluate information sources (including security standards and scientific publications) to expand knowledge
- Can independently analyze and formulate security problems and propose appropriate solutions for uncomplicated incidents

In terms of social competencies:

- Is prepared to critically evaluate their own knowledge and expand it, understanding the need for continuous improvement of technical data protection skills
- Is prepared to uphold high ethical standards in professional work, particularly in engineering practice
- Is prepared to take responsibility when performing tasks in professional activities

Probability and statistics

lecture

2 ECTS

Teaching methods	lecture / conversational lecture
Method of verifying education	exam / assignments / activity during classes

OBJECTIVES

The course introduces students to advanced probability and statistics, with an emphasis on their applications in computer science, such as data analysis and machine learning. It aims to develop students' skills in calculating probabilities, performing statistical analyses, and interpreting results in practical contexts.

COURSE CONTENT

- Probability and statistics as a language for describing events
- Population as the sample space of events; statistical sample as a specific subset; empirical distributions
- Random events and the probabilistic space
- Concept and definitions of probability; Kolmogorov axioms; probability of the sum and product of events; conditional, total, and Bayes' probability; independence of events
- One-dimensional random variables: discrete and continuous; distributions of random variables: Bernoulli, binomial, Poisson, Student's t, normal, and standard normal distributions
- Parameters of random variables: expected value, variance, standard deviation; cumulative distribution function (CDF) and its properties
- Stochastic processes, stochastic matrices and graphs; Markov processes and Poisson processes
- Parameter estimation: estimators (properties), point and interval estimation (e.g., mean and variance), maximum likelihood estimation (MLE)
- Statistical hypothesis testing: key concepts (null and alternative hypotheses, significance level, type I and II errors); selected parametric tests (e.g., Student's t-test) and non-parametric tests (e.g., χ^2 test) and their applications in computer science

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of selected topics in probability theory, including definitions of random events, random variables, probability distributions, basic distributions, and parameters of random variables.
- Has advanced knowledge and understanding of selected topics in mathematical statistics, including parameter estimation, hypothesis testing, and the relationships between the sample space and statistical samples.
- Has advanced knowledge and understanding of selected applications of statistics in computer science, such as data analysis and the fundamentals of machine learning.

In terms of skills:

- Can calculate probabilities and analyze random variables.
- Can apply statistical methods to data analysis.
- Can calculate probabilities of random events and parameters in probability distributions, and analyze relationships between the population and a statistical sample.
- Can independently search for information and expand knowledge using various sources, including scientific publications, online resources, and tutorials.
- Can independently formulate and analyze problems, as well as propose solutions to simple problems in predictable conditions.

In terms of social competencies:

- Is ready to critically evaluate and expand their knowledge, understanding the need for continuous improvement of statistical skills in computer science.
- Is prepared to uphold high ethical standards in professional practice.
- Is ready to take responsibility for tasks in their professional activities.

Probability and statistics

workshops

2 ECTS

Teaching methods	workshop method / individual and group work
Method of verifying education	assignments / activity during classes

OBJECTIVES

The course introduces students to advanced probability and statistics, with an emphasis on their applications in computer science, such as data analysis and machine learning. It aims to develop students' skills in calculating probabilities, performing statistical analyses, and interpreting results in practical contexts.

COURSE CONTENT

- Probability and statistics as a language for describing events
- Population as the sample space of events; statistical sample as a specific subset; empirical distributions
- Random events and the probabilistic space
- Concept and definitions of probability; Kolmogorov axioms; probability of the sum and product of events; conditional, total, and Bayes' probability; independence of events
- One-dimensional random variables: discrete and continuous; distributions of random variables: Bernoulli, binomial, Poisson, Student's t, normal, and standard normal distributions
- Parameters of random variables: expected value, variance, standard deviation; cumulative distribution function (CDF) and its properties
- Stochastic processes, stochastic matrices and graphs; Markov processes and Poisson processes
- Parameter estimation: estimators (properties), point and interval estimation (e.g., mean and variance), maximum likelihood estimation (MLE)
- Statistical hypothesis testing: key concepts (null and alternative hypotheses, significance level, type I and II errors); selected parametric tests (e.g., Student's t-test) and non-parametric tests (e.g., χ^2 test) and their applications in computer science

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of selected topics in probability theory, including definitions of random events, random variables, probability distributions, basic distributions, and parameters of random variables.
- Has advanced knowledge and understanding of selected topics in mathematical statistics, including parameter estimation, hypothesis testing, and the relationships between the sample space and statistical samples.
- Has advanced knowledge and understanding of selected applications of statistics in computer science, such as data analysis and the fundamentals of machine learning.

In terms of skills:

- Can calculate probabilities and analyze random variables.
- Can apply statistical methods to data analysis.
- Can calculate probabilities of random events and parameters in probability distributions, and analyze relationships between the population and a statistical sample.
- Can independently search for information and expand knowledge using various sources, including scientific publications, online resources, and tutorials.
- Can independently formulate and analyze problems, as well as propose solutions to simple problems in predictable conditions.

In terms of social competencies:

- Is ready to critically evaluate and expand their knowledge, understanding the need for continuous improvement of statistical skills in computer science.
- Is prepared to uphold high ethical standards in professional practice.
- Is ready to take responsibility for tasks in their professional activities.

Object-oriented programming 1

workshops

2 ECTS

Teaching methods	conversational lecture / individual and group work
Method of verifying education	assignments / activity during classes

OBJECTIVES

The course focuses on introducing students to the object-oriented programming (OOP) paradigm using the Python language, incorporating selected aspects of Agile methodology, such as creating user stories, and documentation in UML. The aim of the course is to develop skills in designing and implementing classes, objects, and relationships between them, including associations, generalizations, dependencies, aggregations, compositions, and realizations, as well as understanding the core principles of OOP: abstraction, encapsulation, inheritance, and polymorphism.

COURSE CONTENT

- Introduction to object-oriented programming (OOP) using C++ and Python.
- Fundamentals of classes and objects, including definitions and instances.
- Principles of encapsulation, covering privacy and properties.
- Principles of abstraction, including interfaces and abstract classes.
- Inheritance: basics and class hierarchies (generalizations).
- Polymorphism: method and function overloading, dynamic polymorphism (method overriding).
- Relationships between objects, such as associations, generalizations, dependencies, aggregations, compositions, and realizations.
- Introduction to Agile methodology and creation of user stories.
- Introduction to UML documentation and generation of diagrams from code (sequence diagrams, activity diagrams).
- Design of classes, objects, and relationships using user stories, with coding examples.
- Implementation of inheritance, polymorphism, encapsulation, and relationships integrated with Agile practices, with coding examples.
- Development of a console-based "Zoo Garden" application using OOP principles (abstraction, encapsulation, inheritance, polymorphism, associations, generalizations, dependencies, aggregations, compositions, realizations), Agile (user stories), and UML documentation (sequence and activity diagrams), including diagram generation from code.
- Debugging and optimization of object-oriented code within the project.
- Applications of OOP, Agile, and UML in IT project development.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has an advanced understanding of the object-oriented programming (OOP) paradigm, including encapsulation, inheritance, polymorphism, and abstraction, along with the mechanism of classes and objects, with knowledge of C++ and Python syntax.
- Has an advanced understanding of the principles for designing and implementing relationships between classes, including associations, generalizations, dependencies, aggregations, compositions, and realizations.
- Has an advanced understanding of selected Agile methodology elements, such as user stories, and advanced knowledge of UML documentation principles, including use case diagrams and class diagrams, applied in object-oriented projects.

In terms of skills:

- Can design and implement classes, objects, and complex relationships between them, including associations, generalizations, dependencies, aggregations, compositions, and realizations.
- Can effectively apply the pillars of OOP—inheritance, polymorphism, and encapsulation—in the implementation of complex programming projects.
- Can develop and verify project documentation using UML notation, including leveraging AI tools (e.g., ChatGPT) to generate PlantUML or XML code from source code.
- Can independently search for and critically evaluate information sources, including scientific publications, online resources, and tutorials, to expand knowledge.
- Can independently analyze and formulate uncomplicated problems and propose appropriate solutions in a predictable project environment.

In terms of social competencies:

- Is willing to critically assess their knowledge and expand it, understanding the need for continuous improvement of object-oriented programming skills.
- Is prepared to uphold high ethical standards in professional work within engineering practice.
- Is ready to take responsibility for tasks in professional activities.

Object-oriented programming 1

project

2 ECTS

Teaching methods	individual and group work / project method
Method of verifying education	project / assignments / activity during classes

OBJECTIVES

The course focuses on introducing students to the object-oriented programming (OOP) paradigm using the Python language, incorporating selected aspects of Agile methodology, such as creating user stories, and documentation in UML. The aim of the course is to develop skills in designing and implementing classes, objects, and relationships between them, including associations, generalizations, dependencies, aggregations, compositions, and realizations, as well as understanding the core principles of OOP: abstraction, encapsulation, inheritance, and polymorphism.

COURSE CONTENT

- Introduction to object-oriented programming (OOP) using C++ and Python.
- Fundamentals of classes and objects, including definitions and instances.
- Principles of encapsulation, covering privacy and properties.
- Principles of abstraction, including interfaces and abstract classes.
- Inheritance: basics and class hierarchies (generalizations).
- Polymorphism: method and function overloading, dynamic polymorphism (method overriding).
- Relationships between objects, such as associations, generalizations, dependencies, aggregations, compositions, and realizations.
- Introduction to Agile methodology and creation of user stories.
- Introduction to UML documentation and generation of diagrams from code (sequence diagrams, activity diagrams).
- Design of classes, objects, and relationships using user stories, with coding examples.
- Implementation of inheritance, polymorphism, encapsulation, and relationships integrated with Agile practices, with coding examples.
- Development of a console-based "Zoo Garden" application using OOP principles (abstraction, encapsulation, inheritance, polymorphism, associations, generalizations, dependencies, aggregations, compositions, realizations), Agile (user stories), and UML documentation (sequence and activity diagrams), including diagram generation from code.
- Debugging and optimization of object-oriented code within the project.
- Applications of OOP, Agile, and UML in IT project development.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has an advanced understanding of the object-oriented programming (OOP) paradigm, including encapsulation, inheritance, polymorphism, and abstraction, along with the mechanism of classes and objects, with knowledge of C++ and Python syntax.
- Has an advanced understanding of the principles for designing and implementing relationships between classes, including associations, generalizations, dependencies, aggregations, compositions, and realizations.
- Has an advanced understanding of selected Agile methodology elements, such as user stories, and advanced knowledge of UML documentation principles, including use case diagrams and class diagrams, applied in object-oriented projects.

In terms of skills:

- Can design and implement classes, objects, and complex relationships between them, including associations, generalizations, dependencies, aggregations, compositions, and realizations.
- Can effectively apply the pillars of OOP—inheritance, polymorphism, and encapsulation—in the implementation of complex programming projects.
- Can develop and verify project documentation using UML notation, including leveraging AI tools (e.g., ChatGPT) to generate PlantUML or XML code from source code.
- Can independently search for and critically evaluate information sources, including scientific publications, online resources, and tutorials, to expand knowledge.
- Can independently analyze and formulate uncomplicated problems and propose appropriate solutions in a predictable project environment.

In terms of social competencies:

- Is willing to critically assess their knowledge and expand it, understanding the need for continuous improvement of object-oriented programming skills.
- Is prepared to uphold high ethical standards in professional work within engineering practice.
- Is ready to take responsibility for tasks in professional activities.

ZDW: Embeded Systems / Design of special-purpose systems

lecture

3 ECTS

Teaching methods	lecture / conversational lecture
Method of verifying education	exam / assignments / activity during classes

OBJECTIVES

The course focuses on preparing students for the rigorous engineering work required in the design of systems whose failure could lead to significant losses (e.g., real-time systems, safety-critical systems, embedded industrial systems). It covers the creation of precise requirements documentation, architectural modeling using formal methods (e.g., Statecharts, Petri Nets), and the execution of validation and verification procedures.

COURSE CONTENT

- Introduction to special-purpose systems: definitions and classification of critical systems, including medical, military, industrial, and real-time systems, as well as their life cycles and main differences in engineering.
- Analysis and specification of critical requirements: methods for eliciting and documenting non-functional requirements, including Safety, Reliability, Availability, and Real-Time. Laboratory exercises include developing and validating requirement specifications for a selected system.
- Modeling the architecture of critical systems: architectural patterns (layered, microkernel, real-time patterns) and formal methods, including an introduction to Statecharts and Petri Nets. Laboratory exercises focus on modeling complex system states and transitions using UML-RT or simulation tools.
- Designing for reliability and safety: fault tolerance and redundancy techniques, risk analysis, and technology selection in the context of limited resources (e.g., embedded systems). Laboratory exercises include performing FMEA (Failure Mode and Effects Analysis) for a critical module.
- Verification and validation (V&V) of critical systems: critical testing strategies, simulation methods such as Model-in-the-Loop (MIL) and Hardware-in-the-Loop (HIL), and formal verification. Industry standards such as ISO 26262 and DO-178C are covered. Laboratory exercises involve designing and simulating V&V procedures and evaluating compliance with critical non-functional requirements.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of methods for analyzing and specifying requirements for special-purpose systems, including both functional and non-functional requirements such as safety, reliability, and real-time constraints.
- Has advanced knowledge and understanding of specialized architectural design methods, including patterns for real-time systems, as well as formal modeling techniques such as Statecharts and Petri Nets.
- Has advanced knowledge and understanding of verification and validation techniques for critical systems, including in-the-loop testing, simulations, and formal verification methods.

In terms of skills:

- Can create complete requirements documentation and specifications for a special-purpose system, taking into account relevant industry standards.
- Can apply selected methodologies and tools to model the architecture of a critical system (e.g., advanced UML diagrams, domain-specific languages).
- Can design and execute testing procedures (verification and validation) to ensure compliance with critical non-functional requirements.
- Can select technologies and tools for implementing systems with constrained resources or strict safety requirements.
- Can independently search for and critically evaluate information on specialized methods and tools in systems engineering.
- Can independently analyze and propose solutions to complex design problems.

In terms of social competencies:

- Is ready to critically evaluate their own knowledge and expand it, understanding the need for continuous improvement of skills in embedded systems.
- Is prepared to uphold high ethical standards in professional work within engineering practice.

ZDW: Embeded Systems / Design of special-purpose systems

workshops

2 ECTS

Teaching methods	individual and group work / workshop method
Method of verifying education	assignments / activity during classes

OBJECTIVES

The course focuses on preparing students for the rigorous engineering work required in the design of systems whose failure could lead to significant losses (e.g., real-time systems, safety-critical systems, embedded industrial systems). It covers the creation of precise requirements documentation, architectural modeling using formal methods (e.g., Statecharts, Petri Nets), and the execution of validation and verification procedures.

COURSE CONTENT

- Introduction to special-purpose systems: definitions and classification of critical systems, including medical, military, industrial, and real-time systems, as well as their life cycles and main differences in engineering.
- Analysis and specification of critical requirements: methods for eliciting and documenting non-functional requirements, including Safety, Reliability, Availability, and Real-Time. Laboratory exercises include developing and validating requirement specifications for a selected system.
- Modeling the architecture of critical systems: architectural patterns (layered, microkernel, real-time patterns) and formal methods, including an introduction to Statecharts and Petri Nets. Laboratory exercises focus on modeling complex system states and transitions using UML-RT or simulation tools.
- Designing for reliability and safety: fault tolerance and redundancy techniques, risk analysis, and technology selection in the context of limited resources (e.g., embedded systems). Laboratory exercises include performing FMEA (Failure Mode and Effects Analysis) for a critical module.
- Verification and validation (V&V) of critical systems: critical testing strategies, simulation methods such as Model-in-the-Loop (MIL) and Hardware-in-the-Loop (HIL), and formal verification. Industry standards such as ISO 26262 and DO-178C are covered. Laboratory exercises involve designing and simulating V&V procedures and evaluating compliance with critical non-functional requirements.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of methods for analyzing and specifying requirements for special-purpose systems, including both functional and non-functional requirements such as safety, reliability, and real-time constraints.
- Has advanced knowledge and understanding of specialized architectural design methods, including patterns for real-time systems, as well as formal modeling techniques such as Statecharts and Petri Nets.
- Has advanced knowledge and understanding of verification and validation techniques for critical systems, including in-the-loop testing, simulations, and formal verification methods.

In terms of skills:

- Can create complete requirements documentation and specifications for a special-purpose system, taking into account relevant industry standards.
- Can apply selected methodologies and tools to model the architecture of a critical system (e.g., advanced UML diagrams, domain-specific languages).
- Can design and execute testing procedures (verification and validation) to ensure compliance with critical non-functional requirements.
- Can select technologies and tools for implementing systems with constrained resources or strict safety requirements.
- Can independently search for and critically evaluate information on specialized methods and tools in systems engineering.
- Can independently analyze and propose solutions to complex design problems.

In terms of social competencies:

- Is ready to critically evaluate their own knowledge and expand it, understanding the need for continuous improvement of skills in embedded systems.
- Is prepared to uphold high ethical standards in professional work within engineering practice.

ZDW: Embeded Systems / Design of special-purpose systems

project

3 ECTS

Teaching methods	individual and group work / project method
Method of verifying education	project / assignments / activity during classes

OBJECTIVES

The course focuses on preparing students for the rigorous engineering work required in the design of systems whose failure could lead to significant losses (e.g., real-time systems, safety-critical systems, embedded industrial systems). It covers the creation of precise requirements documentation, architectural modeling using formal methods (e.g., Statecharts, Petri Nets), and the execution of validation and verification procedures.

COURSE CONTENT

- Introduction to special-purpose systems: definitions and classification of critical systems, including medical, military, industrial, and real-time systems, as well as their life cycles and main differences in engineering.
- Analysis and specification of critical requirements: methods for eliciting and documenting non-functional requirements, including Safety, Reliability, Availability, and Real-Time. Laboratory exercises include developing and validating requirement specifications for a selected system.
- Modeling the architecture of critical systems: architectural patterns (layered, microkernel, real-time patterns) and formal methods, including an introduction to Statecharts and Petri Nets. Laboratory exercises focus on modeling complex system states and transitions using UML-RT or simulation tools.
- Designing for reliability and safety: fault tolerance and redundancy techniques, risk analysis, and technology selection in the context of limited resources (e.g., embedded systems). Laboratory exercises include performing FMEA (Failure Mode and Effects Analysis) for a critical module.
- Verification and validation (V&V) of critical systems: critical testing strategies, simulation methods such as Model-in-the-Loop (MIL) and Hardware-in-the-Loop (HIL), and formal verification. Industry standards such as ISO 26262 and DO-178C are covered. Laboratory exercises involve designing and simulating V&V procedures and evaluating compliance with critical non-functional requirements.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of methods for analyzing and specifying requirements for special-purpose systems, including both functional and non-functional requirements such as safety, reliability, and real-time constraints.
- Has advanced knowledge and understanding of specialized architectural design methods, including patterns for real-time systems, as well as formal modeling techniques such as Statecharts and Petri Nets.
- Has advanced knowledge and understanding of verification and validation techniques for critical systems, including in-the-loop testing, simulations, and formal verification methods.

In terms of skills:

- Can create complete requirements documentation and specifications for a special-purpose system, taking into account relevant industry standards.
- Can apply selected methodologies and tools to model the architecture of a critical system (e.g., advanced UML diagrams, domain-specific languages).
- Can design and execute testing procedures (verification and validation) to ensure compliance with critical non-functional requirements.
- Can select technologies and tools for implementing systems with constrained resources or strict safety requirements.
- Can independently search for and critically evaluate information on specialized methods and tools in systems engineering.
- Can independently analyze and propose solutions to complex design problems.

In terms of social competencies:

- Is ready to critically evaluate their own knowledge and expand it, understanding the need for continuous improvement of skills in embedded systems.
- Is prepared to uphold high ethical standards in professional work within engineering practice.

Advanced programming technologies

lecture

1 ECTS

Teaching methods	lecture / conversational lecture
Method of verifying education	exam / colloquium / assignments / activity during classes

OBJECTIVES

The course introduces students to programming in Java, with an emphasis on individual work and practical applications in computer science. It aims to develop students' ability to write Java code, understand object-oriented programming principles (classes, objects, inheritance), and handle exceptions. Students will learn to model real-world scenarios using classes, objects, and interfaces, and gain an understanding of how the Java Virtual Machine (JVM) operates.

COURSE CONTENT

- The course covers Java as a successor to C++ and Rust, emphasizing its safety features such as the absence of pointers, automatic memory management via garbage collection (GC), and the Java Virtual Machine (JVM). Students learn the basic syntax of Java and foundational object-oriented concepts: classes as templates, objects as instances, fields, methods, constructors, and the use of the this keyword for references (safe "pointers").
- Encapsulation and packages are addressed through access modifiers (private, public, protected, default), along with the use of getters and setters versus direct field access, highlighting API stability and implementation hiding. Relationships between objects, including composition and aggregation, demonstrate how objects interact without global variables.
- Inheritance and polymorphism are explored via class extension (extends), method overriding (@Override), and object casting. Abstraction is introduced through abstract classes versus interfaces, emphasizing Java's design for interface-oriented programming and the principle of "top-down" design, with interfaces defining an object's capabilities.
- Exception handling is covered using try-catch-finally blocks, differentiating between critical errors and checked exceptions. Collections and generics, including ArrayList, HashSet, and HashMap, are presented, with an introduction to type safety through generics.
- Engineering tools are introduced with Maven, demonstrating standard project and dependency management practices in Java development.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of the JVM architecture and can distinguish between code compilation and interpretation.
- Has advanced knowledge and understanding of object-oriented programming (OOP) fundamentals, including encapsulation, inheritance, polymorphism, and abstraction.
- Has advanced knowledge and understanding of the Java Collections Framework and basic exception handling mechanisms.
- Understands the concept of an interface as a contract and knows the difference between a class and an interface.

In terms of skills:

- Can design and implement a class hierarchy to solve a given problem.
- Can use interfaces as contracts within application architecture.
- Can use collections (List, Map, Set) instead of raw arrays known from languages like C.
- Can integrate Java code with the Maven build tool and version control systems.
- Can independently search for information and expand knowledge using various sources, including scientific publications, online resources, and tutorials.
- Can independently formulate and analyze problems, as well as propose solutions to straightforward problems in predictable conditions.

In terms of social competencies:

- Is willing to critically evaluate their knowledge and expand it, understanding the need for continuous improvement of Java programming skills.
- Is aware of the importance of writing readable and modular code within a development team.
- Is committed to maintaining high ethical standards in professional work and engineering practice.
- Is ready to responsibly undertake tasks in professional activities.

Advanced programming technologies

project

2 ECTS

Teaching methods	workshop method / projecy method / individual work
Method of verifying education	project / assignments / activity during classes

OBJECTIVES

The course introduces students to programming in Java, with an emphasis on individual work and practical applications in computer science. It aims to develop students' ability to write Java code, understand object-oriented programming principles (classes, objects, inheritance), and handle exceptions. Students will learn to model real-world scenarios using classes, objects, and interfaces, and gain an understanding of how the Java Virtual Machine (JVM) operates.

COURSE CONTENT

- The course covers Java as a successor to C++ and Rust, emphasizing its safety features such as the absence of pointers, automatic memory management via garbage collection (GC), and the Java Virtual Machine (JVM). Students learn the basic syntax of Java and foundational object-oriented concepts: classes as templates, objects as instances, fields, methods, constructors, and the use of the this keyword for references (safe "pointers").
- Encapsulation and packages are addressed through access modifiers (private, public, protected, default), along with the use of getters and setters versus direct field access, highlighting API stability and implementation hiding. Relationships between objects, including composition and aggregation, demonstrate how objects interact without global variables.
- Inheritance and polymorphism are explored via class extension (extends), method overriding (@Override), and object casting. Abstraction is introduced through abstract classes versus interfaces, emphasizing Java's design for interface-oriented programming and the principle of "top-down" design, with interfaces defining an object's capabilities.
- Exception handling is covered using try-catch-finally blocks, differentiating between critical errors and checked exceptions. Collections and generics, including ArrayList, HashSet, and HashMap, are presented, with an introduction to type safety through generics.
- Engineering tools are introduced with Maven, demonstrating standard project and dependency management practices in Java development.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of the JVM architecture and can distinguish between code compilation and interpretation.
- Has advanced knowledge and understanding of object-oriented programming (OOP) fundamentals, including encapsulation, inheritance, polymorphism, and abstraction.
- Has advanced knowledge and understanding of the Java Collections Framework and basic exception handling mechanisms.
- Understands the concept of an interface as a contract and knows the difference between a class and an interface.

In terms of skills:

- Can design and implement a class hierarchy to solve a given problem.
- Can use interfaces as contracts within application architecture.
- Can use collections (List, Map, Set) instead of raw arrays known from languages like C.
- Can integrate Java code with the Maven build tool and version control systems.
- Can independently search for information and expand knowledge using various sources, including scientific publications, online resources, and tutorials.
- Can independently formulate and analyze problems, as well as propose solutions to straightforward problems in predictable conditions.

In terms of social competencies:

- Is willing to critically evaluate their knowledge and expand it, understanding the need for continuous improvement of Java programming skills.
- Is aware of the importance of writing readable and modular code within a development team.
- Is committed to maintaining high ethical standards in professional work and engineering practice.
- Is ready to responsibly undertake tasks in professional activities.

Team project

lecture

3 ECTS

Teaching methods	lecture / problem-solving method
Method of verifying education	exam / assignments / activity during classes

OBJECTIVES

The course focuses on preparing and guiding students to independently solve engineering problems. It equips students with both theoretical knowledge and practical skills in managing IT projects using Prince2 and PMI standards, as well as CASE-class software tools that support the effective planning and execution of projects.

COURSE CONTENT

- Introduction to IT project management, covering the project lifecycle and differences between traditional methodologies (Waterfall) and agile approaches (Scrum, Kanban). Lab: selecting and justifying a methodology for a team project.
- Project standards: PRINCE2 and PMI. Principles and processes of PRINCE2 (Initiating, Controlling, Closing) and key PMI knowledge areas (PMBOK). Role of the project manager. Lab: project initiation—creating a Project Charter and defining the project scope.
- Project planning and estimation using Project Libre. Techniques for decomposition (WBS), time and resource estimation, and risk management. Lab: preparing a detailed work plan, risk register, Gantt chart, and schedule in Project Libre Desktop.
- CASE and DevOps tools in team projects. Introduction to CASE tools (modeling), task management systems (Jira/Trello), Git and repositories, and CI/CD processes. Lab: configuring a DevOps environment, implementing a code repository, and integrating with a task management system.
- Project execution, monitoring, and closure. Performing engineering tasks according to assignments, progress monitoring (metrics, EVM), and change management. Lab: executing the main project phase, status reporting, and technical and organizational project closure.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of IT project management methodologies (both traditional and agile), including the key processes and principles of PRINCE2 and PMI.
- Has advanced knowledge and understanding of project planning, team organization, resource estimation, and risk management in IT projects.
- Has advanced knowledge of the functions of IT tools and can select appropriate CASE and DevOps tools to support the analysis, design, implementation, and monitoring phases of a project.

In terms of skills:

- Can plan, estimate costs and resources, and define risks in an IT project in accordance with methodologies (e.g., using Project Libre).
- Can use CASE and DevOps tools (e.g., code repositories, task management systems, automated testing) effectively in team-based work.
- Can independently carry out assigned engineering tasks in a project while meeting deadlines and quality standards.
- Can document all project phases (initiation, analysis, execution, and closure) according to established project standards.
- Can communicate effectively within a project team and manage personal work time efficiently.

In terms of social competencies:

- Is ready to responsibly and punctually carry out assigned tasks while fulfilling a role within a project team.
- Is prepared to critically assess their own contribution to the project and to accept constructive feedback.
- Understands the need for continuous improvement of knowledge and skills in project management and engineering techniques.

Team project

project

3 ECTS

Teaching methods	individual and group work / project method
Method of verifying education	project / assignments / activity during classes

OBJECTIVES

The course focuses on preparing and guiding students to independently solve engineering problems. It equips students with both theoretical knowledge and practical skills in managing IT projects using Prince2 and PMI standards, as well as CASE-class software tools that support the effective planning and execution of projects.

COURSE CONTENT

- Introduction to IT project management, covering the project lifecycle and differences between traditional methodologies (Waterfall) and agile approaches (Scrum, Kanban). Lab: selecting and justifying a methodology for a team project.
- Project standards: PRINCE2 and PMI. Principles and processes of PRINCE2 (Initiating, Controlling, Closing) and key PMI knowledge areas (PMBOK). Role of the project manager. Lab: project initiation—creating a Project Charter and defining the project scope.
- Project planning and estimation using Project Libre. Techniques for decomposition (WBS), time and resource estimation, and risk management. Lab: preparing a detailed work plan, risk register, Gantt chart, and schedule in Project Libre Desktop.
- CASE and DevOps tools in team projects. Introduction to CASE tools (modeling), task management systems (Jira/Trello), Git and repositories, and CI/CD processes. Lab: configuring a DevOps environment, implementing a code repository, and integrating with a task management system.
- Project execution, monitoring, and closure. Performing engineering tasks according to assignments, progress monitoring (metrics, EVM), and change management. Lab: executing the main project phase, status reporting, and technical and organizational project closure.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of IT project management methodologies (both traditional and agile), including the key processes and principles of PRINCE2 and PMI.
- Has advanced knowledge and understanding of project planning, team organization, resource estimation, and risk management in IT projects.
- Has advanced knowledge of the functions of IT tools and can select appropriate CASE and DevOps tools to support the analysis, design, implementation, and monitoring phases of a project.

In terms of skills:

- Can plan, estimate costs and resources, and define risks in an IT project in accordance with methodologies (e.g., using Project Libre).
- Can use CASE and DevOps tools (e.g., code repositories, task management systems, automated testing) effectively in team-based work.
- Can independently carry out assigned engineering tasks in a project while meeting deadlines and quality standards.
- Can document all project phases (initiation, analysis, execution, and closure) according to established project standards.
- Can communicate effectively within a project team and manage personal work time efficiently.

In terms of social competencies:

- Is ready to responsibly and punctually carry out assigned tasks while fulfilling a role within a project team.
- Is prepared to critically assess their own contribution to the project and to accept constructive feedback.
- Understands the need for continuous improvement of knowledge and skills in project management and engineering techniques.

Electronic transaction systems

lecture

1 ECTS

Teaching methods	lecture / conversational lecture
Method of verifying education	exam / assignments / activity during classes

OBJECTIVES

Introduction for students to the design, implementation, and security of modern electronic transaction systems, including integration with external payment gateways (e.g., Stripe, PayPal) and e-SIM or m-commerce platforms. Through project-based work, students learn the principles of transaction management (ACID), security standards (PCI DSS, PSD2), and transaction authorization mechanisms, preparing them to develop advanced financial and e-commerce systems while adhering to engineering ethics.

COURSE CONTENT

- Introduction to REST architecture and FastAPI: principles of building stateless transactional systems, handling asynchronous requests (async/await), and automatic OpenAPI/Swagger documentation.
- Data modeling and ACID principles: using Pydantic for financial data validation and implementing transactional mechanisms (ACID) in databases to ensure operation consistency.
- Security and eSIM technology: SSL/TLS standards, card data tokenization, and secure provisioning of eSIM profiles through external APIs (e.g., Airalo).
- Integration with payment systems (Sandbox): configuring Stripe/PayPal test environments, managing authorization keys, and handling asynchronous Webhook notifications.
- Simulation and process testing: practical use of test card numbers to verify critical paths (successful payment, declined, 3D Secure error) in a FastAPI environment.
- Legal and ethical aspects in FinTech: PCI DSS standards, PSD2 (Open Banking), financial data protection (GDPR), and the engineer's responsibility for transaction security.
- Project phases:
- Initiation and setup: configuring an asynchronous FastAPI server, database structure, and connecting provider API keys (Stripe Sandbox, Airalo API).
- Purchase logic implementation: building REST endpoints for cart management, input data validation, and integration with the Checkout process.
- Digital delivery and Webhooks: automating issuance of eSIM profiles (QR code) upon payment confirmation from external servers.
- Testing and demo: documenting test scenarios with different card types, analyzing system logs, and presenting a fully operational transactional system.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of transactional system architectures, e-commerce models, and the principles of secure electronic payments within a REST architecture. Has advanced knowledge and understanding of security standards (PCI DSS, PSD2), eSIM technology (Airalo), and the role of asynchronous data processing in FastAPI.

In terms of skills:

- Can integrate an application with external payment systems (Stripe/PayPal) using asynchronous REST APIs and handle Webhook notifications.
- Can design and implement mechanisms ensuring transaction consistency and atomicity (ACID) in database systems.
- Can independently research technical API documentation (e.g., Stripe, Airalo) to implement tokenization and automate eSIM sales.
- Can set up a Sandbox environment, conduct tests using test card numbers, and analyze asynchronous transaction flows.
- Can independently seek out and expand knowledge using various sources, including scientific publications, online resources, and technical tutorials.
- Can independently formulate and analyze engineering problems and propose solutions for predictable problem scenarios.

In terms of social competencies:

- Is ready to critically evaluate their own knowledge and understands the need for continuous improvement in modern programming technologies and FinTech.
- Is prepared to take responsibility for the security and confidentiality of financial data and to uphold ethical standards as an IT engineer.

Electronic transaction systems

project

2 ECTS

Teaching methods	conversational lecture / individual work
Method of verifying education	project / assignments / activity during classes

OBJECTIVES

Introduction for students to the design, implementation, and security of modern electronic transaction systems, including integration with external payment gateways (e.g., Stripe, PayPal) and e-SIM or m-commerce platforms. Through project-based work, students learn the principles of transaction management (ACID), security standards (PCI DSS, PSD2), and transaction authorization mechanisms, preparing them to develop advanced financial and e-commerce systems while adhering to engineering ethics.

COURSE CONTENT

- Introduction to REST architecture and FastAPI: principles of building stateless transactional systems, handling asynchronous requests (async/await), and automatic OpenAPI/Swagger documentation.
- Data modeling and ACID principles: using Pydantic for financial data validation and implementing transactional mechanisms (ACID) in databases to ensure operation consistency.
- Security and eSIM technology: SSL/TLS standards, card data tokenization, and secure provisioning of eSIM profiles through external APIs (e.g., Airalo).
- Integration with payment systems (Sandbox): configuring Stripe/PayPal test environments, managing authorization keys, and handling asynchronous Webhook notifications.
- Simulation and process testing: practical use of test card numbers to verify critical paths (successful payment, declined, 3D Secure error) in a FastAPI environment.
- Legal and ethical aspects in FinTech: PCI DSS standards, PSD2 (Open Banking), financial data protection (GDPR), and the engineer's responsibility for transaction security.
- Project phases:
- Initiation and setup: configuring an asynchronous FastAPI server, database structure, and connecting provider API keys (Stripe Sandbox, Airalo API).
- Purchase logic implementation: building REST endpoints for cart management, input data validation, and integration with the Checkout process.
- Digital delivery and Webhooks: automating issuance of eSIM profiles (QR code) upon payment confirmation from external servers.
- Testing and demo: documenting test scenarios with different card types, analyzing system logs, and presenting a fully operational transactional system.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of transactional system architectures, e-commerce models, and the principles of secure electronic payments within a REST architecture. Has advanced knowledge and understanding of security standards (PCI DSS, PSD2), eSIM technology (Airalo), and the role of asynchronous data processing in FastAPI.

In terms of skills:

- Can integrate an application with external payment systems (Stripe/PayPal) using asynchronous REST APIs and handle Webhook notifications.
- Can design and implement mechanisms ensuring transaction consistency and atomicity (ACID) in database systems.
- Can independently research technical API documentation (e.g., Stripe, Airalo) to implement tokenization and automate eSIM sales.
- Can set up a Sandbox environment, conduct tests using test card numbers, and analyze asynchronous transaction flows.
- Can independently seek out and expand knowledge using various sources, including scientific publications, online resources, and technical tutorials.
- Can independently formulate and analyze engineering problems and propose solutions for predictable problem scenarios.

In terms of social competencies:

- Is ready to critically evaluate their own knowledge and understands the need for continuous improvement in modern programming technologies and FinTech.
- Is prepared to take responsibility for the security and confidentiality of financial data and to uphold ethical standards as an IT engineer.

Conducting IT projects

lecture

1 ECTS

Teaching methods	lecture / conversational lecture / problem-solving method
Method of verifying education	exam / assignments / activity during classes

OBJECTIVES

Introduction for students to the practical planning, execution, and monitoring of IT projects using the latest editions of management standards (PRINCE2® 7, PMBOK® 7) and agile methodologies (Scrum), with a particular emphasis on proficiency in professional ALM tools such as Jira and Azure DevOps.

COURSE CONTENT

- Contemporary approach to project management (PRINCE2® 7): principles, practices, and processes adapted to digital projects.
- Value delivery system (PMBOK® 7): shift from processes to engineering principles and performance domains.
- Agile methodologies (Scrum 2020): roles, events, and artifacts in iterative development of web systems.
- Scope and product management: creating the Product Backlog and defining the Definition of Done (DoD).
- Risk and change management: risk register and handling change requests in a dynamic IT environment.
- Supporting tools (Jira/Azure DevOps): workflow automation and progress tracking (Burndown chart).
- Ethical and legal aspects: engineer's responsibility, licensing, and intellectual property protection.
- Project: initiation and planning – preparing the Project Initiation Document (PID), stakeholder analysis, and risk assessment.
- Project: execution and monitoring – simulating Sprint work, managing tasks in Jira, and reporting progress.
- Project: closure and evaluation – presenting results (Demo), conducting "Lessons Learned" analysis, and preparing post-project documentation.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of the principles, practices, and processes of the latest project management standards (PRINCE2® 7, PMBOK® 7) as well as agile methodologies (Scrum).
- Has advanced knowledge and understanding of the IT project lifecycle, including legal, ethical, and economic conditions for conducting business.

In terms of skills:

- Can develop comprehensive documentation for IT tasks (e.g., PID, Backlog) and present results clearly to both specialists and non-specialists.
- Can plan, verify solutions, and assess the limitations of standard management methods in specific project contexts.
- Can use modern IT tools (e.g., Jira, Azure DevOps) for task management and project execution.
- Can independently seek information and expand knowledge using various sources, including scientific publications, online resources, and technical tutorials.
- Can independently formulate and analyze problems and propose solutions for uncomplicated problems in predictable project environments.

In terms of social competencies:

- Is ready to critically evaluate their own knowledge and to continuously expand it. Understands the need for ongoing improvement of IT project management skills.
- Is prepared for active teamwork, taking on various roles within a team, and accepting responsibility for the outcomes of collective efforts.

Conducting IT projects

project

2 ECTS

Teaching methods	problem-solving method / project method / individual and group work
Method of verifying education	project / assignments / activity during classes

OBJECTIVES

Introduction for students to the practical planning, execution, and monitoring of IT projects using the latest editions of management standards (PRINCE2® 7, PMBOK® 7) and agile methodologies (Scrum), with a particular emphasis on proficiency in professional ALM tools such as Jira and Azure DevOps.

COURSE CONTENT

- Contemporary approach to project management (PRINCE2® 7): principles, practices, and processes adapted to digital projects.
- Value delivery system (PMBOK® 7): shift from processes to engineering principles and performance domains.
- Agile methodologies (Scrum 2020): roles, events, and artifacts in iterative development of web systems.
- Scope and product management: creating the Product Backlog and defining the Definition of Done (DoD).
- Risk and change management: risk register and handling change requests in a dynamic IT environment.
- Supporting tools (Jira/Azure DevOps): workflow automation and progress tracking (Burndown chart).
- Ethical and legal aspects: engineer's responsibility, licensing, and intellectual property protection.
- Project: initiation and planning – preparing the Project Initiation Document (PID), stakeholder analysis, and risk assessment.
- Project: execution and monitoring – simulating Sprint work, managing tasks in Jira, and reporting progress.
- Project: closure and evaluation – presenting results (Demo), conducting "Lessons Learned" analysis, and preparing post-project documentation.

DESCRIPTION OF THE EXPECTED LEARNING RESULTS

In terms of knowledge:

- Has advanced knowledge and understanding of the principles, practices, and processes of the latest project management standards (PRINCE2® 7, PMBOK® 7) as well as agile methodologies (Scrum).
- Has advanced knowledge and understanding of the IT project lifecycle, including legal, ethical, and economic conditions for conducting business.

In terms of skills:

- Can develop comprehensive documentation for IT tasks (e.g., PID, Backlog) and present results clearly to both specialists and non-specialists.
- Can plan, verify solutions, and assess the limitations of standard management methods in specific project contexts.
- Can use modern IT tools (e.g., Jira, Azure DevOps) for task management and project execution.
- Can independently seek information and expand knowledge using various sources, including scientific publications, online resources, and technical tutorials.
- Can independently formulate and analyze problems and propose solutions for uncomplicated problems in predictable project environments.

In terms of social competencies:

- Is ready to critically evaluate their own knowledge and to continuously expand it. Understands the need for ongoing improvement of IT project management skills.
- Is prepared for active teamwork, taking on various roles within a team, and accepting responsibility for the outcomes of collective efforts.