**SYLLABUS**

**regarding the qualification cycle FROM 2023TO 2024**

1. Basic Course/Module Information

|  |  |
| --- | --- |
| Course/Module title | *MODELLING AND ANALYSIS OF INFORMATION SYSTEMS* |
| Course/Module code \* |  |
| Faculty (name of the unit offering the field of study) | *College of Natural Sciences* |
| Name of the unit running the course | *Institute of Computer Science* |
| Field of study | *Computer Science & Computer Science and Econometrics* |
| Qualification level | *Second degree* |
| Profile | *Academic* |
| Study mode | *Full-time* |
| Year and semester of studies | *Year 2, semester 3* |
| Course type | *Major* |
| Language of instruction | *English* |
| Coordinator | *MSc. Ewa Żesławska* |
| Course instructor | *MSc. Ewa Żesławska* |

\* - as agreed at the faculty

1.1.Learning format – number of hours and ECTS credits

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Semester  (n0.) | Lectures | Classes | Colloquia | Lab classes | Seminars | Practical classes | Internships | others | **ECTS credits** |
| 3 |  |  |  | 30 |  |  |  |  | 4 |

1.2. Course delivery methods

- conducted in a traditional way

- involving distance education methods and techniques

1.3. Course/Module assessment (exam, pass with a grade, pass without a grade)

Lectures - exam

Lab classes - Pass with a grade.

2. Prerequisites

|  |
| --- |
| Knowledge of algorithms and data structures.  Ability to program it applications using object-oriented language.  Skills in the creation of basic UML diagrams.  Skills in the field of software testing.  Skills related to creating project documentation. |

3. Objectives, Learning Outcomes, Course Content, and Instructional Methods

3.1. Course/Module objectives

|  |  |
| --- | --- |
| O1 | To familiarize the student with the methodology of design, requirements specification, modelling and the processes of creating a complex IT system in relation to the real conditions of these processes. |
| O2 | Shaping practical skills in identifying and modelling system processes and their parameters. |
| O3 | Developing practical skills to evaluate the decisions made related to the risk of the designed IT systems and their effectiveness. |
| O4 | Shaping practical, appropriate skills in the field of prototyping and cost estimation of the analysed IT system. |
| O5 | Developing practical skills in using CASE tools in the process of designing, creating and implementing an IT system. |
| O6 | Developing practical skills to integrate knowledge from various sources while formulating and solving tasks related to modelling and designing information systems. |

3.2. Course/Module Learning Outcomes (to be completed by the coordinator)

|  |  |  |
| --- | --- | --- |
| Learning Outcome | The description of the learning outcome  defined for the course/module | Relation to the degree programme outcomes |
| LO\_01 | Apply existing theories, models, tools, and formal methods to the process of developing requirements specifications and modeling and designing a complex computer system with real-world requirements. | K\_W03, K\_W05 |
| LO\_02 | Apply appropriate design methodology to complex information systems, understands the relationships between system components; knows methods and tools for designing information systems. | K\_W06 |
| LO\_03 | Describe the organization of the various stages of an information system project life cycle including implementation and cost estimation. | K\_W08 |
| LO\_04 | Conduct a research experiment by interpreting the results obtained. | K\_U02, K\_K01 |
| LO\_05 | Evaluate and compare design solutions and software development processes, with respect to given usability and economic criteria (complexity of algorithms, speed of operation, time consumption, cost, etc.). | K\_U04 |
| LO\_06 | Design and select appropriate CASE tools supporting the process of analysis, design and development of an information system. | K\_U06 |

**3.3. Course content (to be completed by the coordinator)**

1. Lectures

|  |
| --- |
| Content outline |
| Principles of object oriented modelling |
| Formulation and implementation of requirements |
| Modelling the system's environment and its functionality |
| Designing the system architecture |
| Use of components and patterns project based system implementation |
| Using API (application programming interface) |
| Tools and software development environments |
| Taking care of the process and product quality |
| Methods of software development |
| Validation and testing of software |
| Evolution of software |
| Managing a programming venture |

1. Classes, tutorials/seminars, colloquia, laboratories, practical classes

|  |
| --- |
| Content outline |
| Preparation of the application concept: name, goals, functionalities, potential clients, interface prototype; a collection of initial system requirements. Preparation of project documentation |
| Presentation of the application concept in the group forum. Common discussion on the presented idea: defining the advantages, disadvantages, proposed changes; introducing changes to the concept that take into account the results of discussions |
| Modelling of the system and its environment with UML diagrams |
| Determining the project risk - risk table |
| Defining tasks and preparing a project work schedule |
| Work on GUI (graphical user interface) |
| Creating the application structure. Functional implementation. Testing |
| Preparation of the poster and presentation promoting the application |
| Presentation of the project in a public forum |

3.4. Methods of Instruction

e.g.

*Lecture: a problem-solving lecture/a lecture supported by a multimedia presentation/ distance learning*

*Classes: text analysis and discussion/project work (research project, implementation project, practical project)/ group work (problem solving, case study, discussion)/didactic games/ distance learning*

*Laboratory classes: designing and conducting experiments*

Lecture: lecture with multimedia presentation

Laboratory: discussion, individual work, IT project preparation and its presentation.

4. Assessment techniques and criteria

4.1 Methods of evaluating learning outcomes

|  |  |  |
| --- | --- | --- |
| Learning outcome | Methods of assessment of learning outcomes (e.g. test, oral exam, written exam, project, report, observation during classes) | Learning format (lectures, classes,…) |
| LO\_01 | *Written exam in a test form* | Lectures |
| LO\_02 | *Written exam in a test form* | Lectures |
| LO\_03 | *Written exam in a test form* | Lectures |
| LO\_04 | *Project lab* | Laboratory |
| LO\_05 | *Project lab* | Laboratory |
| LO\_06 | *Project lab* | Laboratory |

4.2 Course assessment criteria

|  |
| --- |
| Credit for the lecture will be carried out in a form of a test. The pass mark is set at 50% of correct answers + 1.  Credit for the laboratory will take place on a basis of a completed project in which the individuals solve specific problem and take a test. |

5. Total student workload needed to achieve the intended learning outcomes

– number of hours and ECTS credits

|  |  |
| --- | --- |
| Activity | Number of hours |
| Scheduled course contact hours | 30 |
| Other contact hours involving the teacher (consultation hours, examinations) | 20 |
| Non-contact hours - student's own work (preparation for classes or examinations, projects, etc.) | 50 |
| Total number of hours | 100 |
| Total number of ECTS credits | 4 |

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

6. Internships related to the course/module

|  |  |
| --- | --- |
| Number of hours | *N/A* |
| Internship regulations and procedures | *N/A* |

7. Instructional materials

|  |
| --- |
| Compulsory literature:  Sommerville I.: Software Engineering, Pearson, 2015 or newer  Bass L., Clements P., Kazman R. ,Software architecture in practice, Addison-Wesley, Upper Saddle River, NJ, 2013 |
| Complementary literature:  Martin Fowler, Patterns of Enterprise Application Architecture, Addison-Wesley Professional, 2002 or newer  Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley Professional, 1994 or newer |

Approved by the Head of the Department or an authorised person