ANEXA 1

SYLLABUS
SOFTWARE ENGINEERING, 2024-2025
Year III / II

1. Information on academic programme

1.1. University	**„1 Decembrie 1918”**
1.2. Faculty	**Faculty of Informatics and Engineering**
1.3. Department	**Informatics, Mathematics and Electronics Department**
1.4. Field of Study	**Computer Science**
1.5. Cycle of Study	**Undergraduate**
1.6. Academic programme / Qualification	**Computer Science** **ESCO-08: 2511/ Systems Analyst, 2512/ Software developers Analyst 251201** **Computer System Programmer 251204** **Computer System Engineer 251203**

2. Information of Course Matter

2.1. **Course**	*Software Engineering*			2.2. **Code**		CSE 311	
2.3. **Course Leader**		Lect. dr. Domsa Ovidiu					
2.4. **Seminar Tutor**		Lect.drd. Cristea Daniela					
2.5. Academic Year	**III**	2.6. Semester	**II**	2.7. Type of Evaluation (E – final exam/ CE - colloquy examination / CA -continuous assessment)	**E**	2.8. Type of course (**C**– Compulsory, **Op** – optional, **F** - Facultative)	**C**

3. Course Structure (Weekly number of hours)

3.1. Weekly number of hours	*5*	3.2. course	***2***	3.3. seminar, laboratory	***3***
3.4. Total number of hours in the curriculum	*60*	3.5. course	***24***	3.6. seminar, laboratory	***36***
Allocation of time:					
Individual study of readers					***20***
Documentation (library)					***20***
Home assignments, Essays, Portfolios, projects					***20***
Tutorials					***20***
Assessment (examinations)					***10***
Other activities…….					

3.7 Total number of hours for individual study	90
3.8 Total number of hours in the curriculum	60
3.9 Total number of hours per semester	150
3.10 Number of ECTS	6

4. Prerequisites (*where applicable*)

4.1. curriculum-based	***INFO209, INFO207***
4.2. competence-based	*Room equipped with video projector / board* *Laboratory – computer, Project Management applications.*

5.Requisites (*where applicable*)

5.1. course-related	Room equipped with video projector / boar
5.2. seminar/laboratory-based	*Laboratory – computer, Software: Microsoft Project.*

6.Specific competences to be aquired (chosen by the course leader from the programme general competences grid)

Professional competences	C2.2 The identification and explanation of appropriate mechanisms for software analysis, design and development. C3.4. UML Data and model's description. C3.5. The development of software engineering components for business projects.
Transversal competences	C6.2. The identification and explanation of base architectures, structures, organizing and management systems for software development stages. C6.3. The use of various techniques for installing, configuring and managing different software tools

7.Course objectives (as per the program specific competences grid)

7.1 General objectives of the course	Abilities to develop and manage all stack for problems solving regarding information's structuring, storing, processing, and documentation and date description.
7.2 Specific objectives of the course	Explain basic concepts in the field of software engineering and process stages software development to describe and compare models of software development processes Analyze user requirements,identify solutions, compare and select tools appropriate software to resolve a given issue. Use proper UML core charts (UC, activity, classes, sequences, states) in system analysis and design software. To argue the importance of the field software engineering and ethical principles of the engineering profession software. Develop a correct relationship with clients.

8.Course contents

8.1 Course (learning units)	**Teaching methods**	**Remarks**
1. Introduction to software engineering **1.1 Development of software systems** **1.2 Software engineering features** **1.3 Notes on the development of a software product**	***Lecture, conversation, exemplification***	
2. The life cycle of a software product **2.1 Phases of the life cycle** **2.2 Cascade models**	***Lecture, conversation, exemplification***	

2.3 Iterative models **2.4 Extreme Programming Methodology**		
3. Requirements engineering **3.1 Specific issues** **3.2 Types of requirements** **3.3 Requirements analysis** **3.4 Specification of requirements**	***Lecture, conversation, exemplification***	
4. Software modeling **4.1 Modeling languages** **4.2 Structured modeling** **4.3 Object Oriented Modeling** **4.4 UML Language**	***Lecture, conversation, exemplification***	
5. Designing software systems **5.1 Software architectures** **5.2 Characteristics of a software system** **5.3 Architectural Styles** **5.4 Architectural models**	***Lecture, conversation, exemplification***	
6. Development of software systems **6.1 RAD** **6.2 Incremental development** **6.3 Prototyping** **6.4 Agile methods** **6.5 Development cycle in extreme programming** **6.6 Reuse in the development of a software system**	***Lecture, conversation, exemplification***	
7. Testing and validation **7.1 Verification and Validation Process** **7.2 Static and dynamic verification** **7.3 Testing and debugging** **7.4 Planning the test** **7.5 Static analysis** **7.6 Testing and validating systems**	***Lecture, conversation, exemplification***	
3. Case study	***Lecture, conversation, exemplification***	
Seminars-laboratories	**Teaching methods**	
Microsoft project and different tools, general presentation, description of the functionalities, examples	***Project-work, computer-based activities, laboratory activities***	
Applications frame and project design using project management tools	***Project-work, computer-based activities, laboratory activities***	
UML description using software tools, Use proper UML core charts (UC, activity, classes, sequences, states)	***Project-work, computer-based activities, laboratory activities***	
Designing tools. Designing objects – based content.	***Project-work, computer-based activities, laboratory activities***	
Designing software systems, Software architectures, Arhitectural Style, Architectural models	***Project-work, computer-based activities, laboratory activities***	
Agile methods, tool for monitoring and planning tasks.(Jira, Mantis, Scrum monitoring)	***Project-work, computer-based activities, laboratory activities***	
Testing and validation tools	***Project-work, computer-based activities, laboratory activities***	
Complet case study. Project.	***Project-work, computer-based activities, laboratory activities***	

References
1. BASS, L., CLEMENTS, P., KAZMAN R.: Software Architecture in Practice, 2nd ed., Addison-Wesley, 2003
2. MARTIN, ROBERT CECIL: Agile software development: principles, patterns, and practices, Pearson Education, 2002
3. McCONNELL, STEVE: Code Complete, 2nd ed., Microsoft Press, 2004
4. OTERO, C.E.: Software Engineering Design, CRC Press, 2012.
5. Gillian Lemke, The Software Development Life Cycle and Its Application, Eastern Michigan University, 2018.

site: http://softwareengineeringdesign.com/Default.htm (2019)
https://creately.com/blog/diagrams/uml-diagram-types-examples/ (2023)
https://staruml.io/ (2023)

9.Corroboration of course contents with the expectations of the epistemic community's significant representatives, professional associations and employers in the field of the academic programme

Not applicable

10.Assessment

Activity	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Percentage of final grade
10.4 Course	*Final evaluation*	*Project*	60%
	-	-	-
10.5 Seminar/laboratory	*Continuous assessment*	*Laboratory activities portfolio*	40%
	-		-
10.6 Minimum performance standard:			
Implementation and documentation of the software units in a web applications including object oriented programming language and efficiently using the related concepts.			

Submission date | Course leader signature | Seminar tutor signature

Date of approval by Department members | Department director signature