COURSE OUTLINE

(1) GENERAL

SCHOOL	SCHOOL OF ENGINEERING		
ACADEMIC UNIT	Department of Informatics and Computer Engineering		
LEVEL OF STUDIES	Undergraduate		
COURSE CODE	ICE-7105	**SEMESTER**	**7th,9th**
COURSE TITLE	Special Topics in Software Engineering		

INDEPENDENT TEACHING ACTIVITIES if credits are awarded for separate components of the course, e.g. lectures, laboratory exercises, etc. If the credits are awarded for the whole of the course, give the weekly teaching hours and the total credits	**WEEKLY TEACHING HOURS**	**CREDITS**
Lectures	2	
Laboratories	2	
	4	5
Add rows if necessary. The organisation of teaching and the teaching methods used are described in detail at (d).		

COURSE TYPE general background, special background, specialised general knowledge, skills development	Scientific Area, Skills Development, Specialization
PREREQUISITE COURSES:	
LANGUAGE OF INSTRUCTION and EXAMINATIONS:	Greek
IS THE COURSE OFFERED TO ERASMUS STUDENTS	No
COURSE WEBSITE (URL)	https://eclass.uniwa.gr/courses/MSCCSE107/

(2) LEARNING OUTCOMES

Learning outcomes

The course learning outcomes, specific knowledge, skills and competences of an appropriate level, which the students will acquire with the successful completion of the course are described.

Consult Appendix A

- Description of the level of learning outcomes for each qualifications cycle, according to the Qualifications Framework of the European Higher Education Area
- Descriptors for Levels 6, 7 & 8 of the European Qualifications Framework for Lifelong Learning and Appendix B
- Guidelines for writing Learning Outcomes

The course aims to enrich the knowledge that students already have about Software Engineering from the relevant course in the 6th semester. Specialized topics are discussed which cover the entire life cycle of a modern software application / an information system.

The main learning objectives of the course are:

- Understanding empirical conclusions as well as widespread perceptions regarding the software development process
- Ability to apply flexible (agile) software development methodologies and in particular with the SCRUM methodology.
- Knowledge and application of design standards (patterns) during software design and development.
- Assimilation of the main approaches regarding enterprise system architectures and their application in solving information systems design problems.
- Understanding and applying code writing best practices while using techniques such as Dependency Injection (DI) and Aspect Oriented Programming (AOP).
- Knowledge and application of unit control and integration control procedures.
- Familiarity with programming in the Computing Cloud (cloud)
- Knowledge of the legal and administrative issues related to the management of software projects and the ability to apply them in practice.

General Competences

Taking into consideration the general competences that the degree-holder must acquire (as these appear in the Diploma Supplement and appear below), at which of the following does the course aim?

Search for, analysis and synthesis of data and information, with the use of the necessary technology
Adapting to new situations
Decision-making
Working independently
Team work
Working in an international environment
Working in an interdisciplinary environment
Production of new research ideas

Project planning and management
Respect for difference and multiculturalism
Respect for the natural environment
Showing social, professional and ethical responsibility and sensitivity to gender issues
Criticism and self-criticism
Production of free, creative and inductive thinking
……
Others…
…….

- Search, analysis and synthesis of data and information, using the necessary technologies
- Autonomous work
- Teamwork
- Planning and Project Management
- Generation of new research ideas

(3) SYLLABUS

- Empirical conclusions and popular perceptions regarding software development
- SCRUM methodology (roles, processes, requirements analysis)
- Patterns
- Enterprise application architectures
- Coding best practices
- Dependency Injection – Aspect Oriented Programming
- Unit Testing and Integration Testing
- Development of Cloud Computing services
- Legal and administrative issues of software project management

(4) TEACHING and LEARNING METHODS - EVALUATION

<table>
<tr><td><b>DELIVERY</b>
Face-to-face, Distance learning, etc.</td><td>Face-to-face</td></tr>
<tr><td><b>USE OF INFORMATION AND COMMUNICATIONS TECHNOLOGY</b>

Use of ICT in teaching, laboratory education, communication with students</td><td>▪ Use of electronic material to support courses (slides and material in electronic format)
▪ Use the C# programming language for .Net
▪ Use of Azure cloud infrastructure</td></tr>
<tr><td><b>TEACHING METHODS</b>
The manner and methods of teaching are described in detail.
Lectures, seminars, laboratory practice, fieldwork, study and analysis of bibliography, tutorials, placements, clinical practice, art workshop, interactive teaching, educational visits, project, essay writing, artistic creativity, etc.

The student's study hours for each learning activity are given as well as the hours of non-directed study according to the principles of the ECTS</td><td>
<table>
<tr><th>Activity</th><th>Semester workload</th></tr>
<tr><td>Lectures</td><td>26</td></tr>
<tr><td>Laboratory practice</td><td>26</td></tr>
<tr><td>Assignments</td><td>30</td></tr>
<tr><td>Independent personal study</td><td>43</td></tr>
<tr><td></td><td></td></tr>
<tr><td></td><td></td></tr>
<tr><td></td><td></td></tr>
<tr><td></td><td></td></tr>
<tr><td></td><td></td></tr>
<tr><td>Course total</td><td><b>125</b></td></tr>
</table>
</td></tr>
</table>

STUDENT PERFORMANCE EVALUATION Description of the evaluation procedure Language of evaluation, methods of evaluation, summative or conclusive, multiple choice questionnaires, short-answer questions, open-ended questions, problem solving, written work, essay/report, oral examination, public presentation, laboratory work, clinical examination of patient, art interpretation, other Specifically-defined evaluation criteria are given, and if and where they are accessible to students.	I. Written final exams (100%) that includes: – Theory questions – Problem solving – Comparative evaluation of theory elements

(5) ATTACHED BIBLIOGRAPHY

- Suggested bibliography

1. Ian Sommerville, “Engineering Software Products: An Introduction to Modern Software Engineering”, Pearson, 2019
2. Robert. L. Glass, “Facts and Fallacies of Software Engineering”, Addison Wesley, ISBN:0 – 321 – 11742 - 5
3. Kenneth S. Rubin, “Essential Scrum”, Addison Wesley, ISBN: 978-0-13-704329-3
4. Steve McConnell, “Code Complete 2”, Microsoft Press
5. E. Gamma, R. Helm, R. Johnson, J. Vlissides, “Design Patterns: Elements of Reusable Object Oriented Software”, Addison Wesley, 978-0201633610
6. Mark Seemann, “Dependency Injection in .Net”, Manning, ISBN: 9781935182504
7. Roy Osherove, “The Art of Unit Testing: With Examples in .NET”, Manning, ISBN: 9781933988276