UNIVERSITY of NICOSIA

Course Syllabus

Course Code	Course Title	ECTS Credits
COMP-401	Software Engineering	6
Prerequisites ▪ COMP 113 ▪ COMP 201 ▪ Junior standing	**Department** Computer Science	**Semester** Spring
Type of Course Required	**Field** Computer Science	**Language of Instruction** English
Level of Course 1st Cycle	**Lecturer(s)** Dr Ioannis Katakis	**Year of Study** 3 or 4
Mode of Delivery Face to Face	**Work Placement** N/A	**Corequisites** None

Course Objectives:

This course aims to provide students with the application of theory, knowledge, and practice to develop software systems that satisfy the requirements of users and customers in an effective and efficient way. The main objectives of this course are to:

- Describe all phases of the life cycle of a software system, including requirements analysis and specification, design, construction, testing, deployment, and operation and maintenance.
- Demonstrate tools for managing software development; analyzing and modeling software artifacts; assessing and controlling quality; and for ensuring a disciplined, controlled approach to software evolution and reuse.
- Present the “good practice” tools, methods, and approaches that are most applicable for a given development environment.

Learning Outcomes:

Upon completion of this course students are expected to be able to:

1. Explain the concept of a software life cycle and provide an example, illustrating its phases including the deliverables that are produced.

2. Select, with justification the software development models and process elements most

appropriate for the development and maintenance of a diverse range of software products.

3. Explain the role of process maturity models. Develop a medium-size software product using a software requirement specification, an accepted program design methodology (e.g., structured or object-oriented), and appropriate design notation.

4. Design appropriate UML diagrams for a medium-sized software system.

5. Discuss the properties of good software design including the nature and the role of associated documentation.

6. Evaluate the quality of multiple software designs based on key design principles and concepts.

7. Distinguish between program validation and verification.

8. Describe the role that tools can play in the validation of software.

9. Distinguish between the different types and levels of testing (unit, integration, systems, and acceptance) for medium-size software products and related materials.

10. Create, evaluate, and implement a test plan for a medium-size code segment.

Course Content:

- Software Engineering Basic Concepts
- Project Management
- Project Metrics
- Project Planning
- Risk Analysis and Management
- Project Scheduling and Tracking
- Software Analysis
- Software Design
- Software Testing Techniques and Strategies

Learning Activities and Teaching Methods:

Lectures, Collaborative Learning through a Group Project, Articles, In-Class Exercises, Student-led Presentations

Assessment Methods:

Mid-term exam, Project, Assignments/Quizzes, Final Exam.

Required Textbooks / Readings:

Title	Authors	Publisher	Year	ISBN
Software Engineering: A Practitioner's Approach	Pressman, R. S. and Maxim, B. R.	McGraw Hill, 8th Edition	2014	978-0078022128

Recommended Textbooks / Readings:

Title	Authors	Publisher	Year	ISBN
Ian Sommerville	Software Engineering	Pearson; 10th edition	2015	978-0133943030
Software Engineering: Principles and Practice	H. van Vliet	John Wiley & Sons, Third edition	2008	978-0470031469