|  |  |
| --- | --- |
| Department: Software Engineering | |
| Module Title: | Module Code: |
| Module ECTS: 24 |
| Course Title: Software Engineering-II | Course Code:ITSE-2231 |
| Course ECTS: 6 |
| Course Duration:16 Weeks |

|  |  |
| --- | --- |
| **Instructor’s Contact Information:**  Name: **Dr. V Sunkari**  Office: Pharmacy Bldg.  Email**:** [**v.sunkari@aait. edu.et**](mailto:v.sunkari@aait.%20edu.et)  Office Hours:  Monday [1:00 Am – 12:00 Pm]  Friday [1:00Am- 12:00pm] | **Course Information:**  Academic Year: 2019-2020  Semester: II  Meeting day: Office Hour  Meeting time: Office Hour  Meeting location: Pharmacy Bldg. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Course Description** | | | | | | |
| This course stresses on uncovering the approaches and techniques to software Testing. Students will be introduced to various testing strategies, types and tools.  The course mainly bases on foundations given on software engineering I. Students are required to undertake two software development projects as a way to exploit what they got in prerequisites. Moreover, the interim as well as the final deliverables of the aforementioned projects will be used as a play ground for Unit Testing, Integration Testing, and System Testing. | | | | | | |
| **Course Objectives** | | | | | | |
| After completing this course:  ⮚Students will know basic verification and validation process.  ⮚Students will be comfortable with various testing techniques.  ⮚Students will understand that how to generate test cases. | | | | | | |
| **Learning Outcomes**  Upon completion of this course each student will be able to:   * Understand the processes, principles and techniques of software testing, software verification and validation * Develop software testing plans and models * Generate test cases according to various testing criteria * Apply different types of testing approaches * Develop and apply continuous verification methods * Conduct test reviews and formalize documentation   Efficiently use software verification and validation tools | | | | | | |
| **Practical Skills** | | | | **Transferable Skills** | | |
| **Prerequisites:** | | | | | | |
| **Student Work Load** | | | | | | |
| **Lecture:32 hrs** | **Tutorial: 32hrs** | | | **Home Study:65 hrs** | | **Laboratory: 48hrs** |
| **Course Content** | | | | | | |
| **Introduction**  **Chapter One: Architectural Styles, Design Patterns, and Objects: Preliminaries**  Architectural Design: Architectural Design Decisions, Architectural Views, Architectural Patterns: Layered Architecture, Repository architecture, Client–server architecture, Pipe and filter architecture and Application architectures, Object Oriented Design: Object - Oriented Design Using the UML, Design Patterns.  **Assignment 1.1: Software Requirement and Design Specification Document**  **Chapter Two – Software Verification and Validation:** Quality Revolution, Software Quality, Distinguishing Verification and Validation, Limitations of Verification and Validation, The Role of V & V in Software Evolution, Types of Products and Objectives of V & V  TESTING CONCEPTS, ISSUES, AND TECHNIQUES: Purposes of Testing, Terms in Testing, Failure, Error, Fault, and Defect, Generic Testing Processes, Functional Vs. Structural Testing, Test-Case Design [4], Concept of Complete Testing, When to Stop Testing, Notion of Software Reliability.  **Assignment II: Software Testing; Seminar I**  **Chapter Three - Theory of Program Testing:** Basic Concepts In Testing Theory, Theory of Goodenough And Gerhart, Theory of Weyuker And Ostrand, Theory of Gourlay, Adequacy of Testing, Limitations of Testing.  **Chapter Four – Unit Testing:** Concept of Unit Testing, Phases of Unit Testing, Static Unit Testing, Program Inspections, Walkthroughs, and Reviews [4], Dynamic Unit Testing, Test-Case Design Techniques [4], Mutation Testing, Debugging [4], Debugging by Brute Force, Debugging by Induction, Debugging by Deduction, Debugging by Backtracking, Debugging by Testing, Extreme Testing.  **Assignment 4.1: Document Review Assignment 4.2: Code Review**  **Chapter Five - Test Case Design Techniques:** Control Flow Testing, Control Flow graph,Paths and Path Selection Criteria, Generating Test Input, Infeasible Paths, Data Flow testing, Dynamic Dataflow Testing, Dataflow Graph, Dataflow Testing Criteria, Domain Testing: Testing for Domain Error, Sources and types of domain error, Test Selection Criterion  **Assignment 5.1: Literature Review**  **Assignment 5.2: Software Test Plan**  **Chapter Six – System Integration Testing:** Concept of Integration Testing, Interfaces and Interface Errors, System Integration Techniques, Software and Hardware Integration Testing, Off-the-Shelf Component Integration Testing.  **Chapter Seven – System Testing:** Basic Testing, Functionality Testing, Robustness Testing, Interoperability Testing, Performance Testing, Scalability Testing, Stress Testing, Load and Stability, Reliability, Regression Testing, Documentation, Regulatory Testing. | | | | | | |
| **Schedule –** optional | | | | | | |
| **Week** | | **Lecture Topics** | Laboratory Exercises | | Assessment | |
| Week 1 | | Chapter One |  | |  | |
| Week 2 | | Chapter One |  | |  | |
| Week 3 | | Chapter Two |  | |  | |
| Week 4 | | Chapter Two |  | | Assessment Test-1 | |
| Week 5 | | Chapter Three |  | |  | |
| Week 6 | | Chapter Four |  | | Assessment Test-2 | |
| Week 7 | | Chapter Five |  | |  | |
| Week 8 | | Chapter Six |  | |  | |
| Week 9 | | Chapter Seven |  | |  | |
| Week 10 | |  |  | | **Final Examination [30]** | |
| **Assessment Method:** | | | | | | |
| Assessment Examination: (40 %)  Final Examination: (30 %)  Lab (30 %) | | | | | | |
| **Course Policies** | | | | | | |
| * **Attendance:** It is compulsory to attend class in time and every time. Missing more than three classes during the term causes readmission for that course. * **Assignments:** No Late Assignment will be accepted * **Test/Quizzes:** Rarely re examination schedules will be arranged for those who missed the exam by accidental or uncontrollable situation. * **Cheating/Plagiarism:** No second Chance or excuses. | | | | | | |
| **References** | | | | | | |
| 1. Jeff Tian, “Software Quality Engineering - Testing, Quality Assurance, and Quantifiable Improvement”, Southern Methodist University - Department of Computer Science and Engineering, 2005. 2. Ian Sommerville, “Software Engineering”, Pearson Education, Ed. 9, 2011. 3. Kshirasagar Naik and Priyadarshi Tripathy, “Software Testing and Quality Assurance - Theory and Practice”, University of Waterloo, 2008. 4. Glenford J. Myers, “The Art of Software Testing”, John Wiley & Sons, Inc., 2nd Ed., 2004. | | | | | | |