20MCA107	ADVANCED SOFTWARE ENGINEERING	CATEGORY	L	T	P	CREDIT
		GENERAL	3	1	0	4

Preamble:
Most of the programs on Computer Applications do not give due importance to teach Software Engineering in an Industry perspective. But this course, built upon the tools and techniques prevalent in Industry today, is supposed to make students Industry-ready.

Prerequisite: Programming proficiency in at least one of C, C++, Java, Python or PHP programming languages.

Course Outcomes: After the completion of the course the student will be able to

CO 1	Get a full view of the Software life cycle
CO 2	Gain a deep knowledge of Software Planning, Analysis and Design and Software Engineering Models
CO 3	Have a great comprehension of Coding Practices, Version Control using 'git' and Software Quality
CO 4	Acquire ample grasp of Design Patterns
CO 5	Get deeply familiarised with Software Testing and its automation
CO 6	Start using Agile Methodology
CO 7	Begin to apply CI/CD techniques in Software development

Mapping of course outcomes with program outcomes

	PO 1	**PO 2**	**PO 3**	**PO 4**	**PO 5**	**PO 6**	**PO 7**	**PO 8**	**PO 9**	**PO 10**	**PO 11**	**PO 12**
CO 1		2	2					3			1	1
CO 2		3	3					3				
CO 3					3				3	2	2	
CO 4			3		3							
CO 5					3					2	3	
CO 6					2			2	2		2	3
CO 7					3			1		2		

Assessment Pattern

Bloom's Category	**Continuous Assessment Tests**		**End Semester Examination**
	1	**2**	
Remember	10	10	10
Understand	20	20	20
Apply	10	10	20
Analyse			
Evaluate			
Create	10	10	10

Mark distribution

Total Marks	**CIE**	**ESE**	**ESE Duration**
100	40	60	3 hours

Continuous Internal Evaluation Pattern:

Attendance	: 8 marks
Continuous Assessment Test (2 numbers)	: 20 marks
Assignment/Quiz/Course project	: 12 marks

End Semester Examination Pattern: There will be two parts; Part A and Part B. Part A contains 10 compulsory short answer questions, 2 from each module. Each question carries 3 marks. Part B contains 2 questions from each module of which student should answer any one. Each question can have maximum 2 sub-divisions and carry 6 marks.

Course Level Assessment Questions

Course Outcome 1 (CO1):

(a) Understand the software development as an engineering process and its stages.

(b) Understand Software development lifecycle (SDLC).

(c) Understand software engineering models.

(d) Learn how to prepare software requirements specification, approaches and methodologies to prepare requirement specifications document.

Course Outcome 2 (CO2)

(a) Understand writing industry-grade software programs, following style guides and coding standards.

(b) Learn core concepts of software version control system and common operations with Git distributed version control system.

(c) Understanding software quality concepts with respect to software requirement specifications document, what to conform to at various stages of SDLC.

(d) Understand what to ensure at various stage of SDLC to ensure quality of developed software system.

Course Outcome 3(CO3):

(a) Learn Object Oriented Programming concepts comprehensively.

(b) Learn the concept of Design Patterns, category of patterns, and how to select appropriate design patterns.

(c) Understand Unit testing concepts and xUnit architecture.

(d) Learn Unit testing frameworks and writing unit testing for Java and one of PHP or Python.

(e) Understand the concepts Continuous Integration and Continuous Delivery (CICD).

Course Outcome 4 (CO4):

(a) Knowledge of Git distributed version control system to use in a product environment.

(b) Knowledge of OOP paradigm and software Design Patterns to design the software system.

(c) Knowledge of unit testing frameworks such as Junit, uniitest, phpdbg for wiring units tests in a software production environment.

(d) Knowledge of software testing CI/CD practices.

Course Outcome 5 (CO5):

(a) Understand software testing concepts and principles.

(b) Learn common approaches to ensure software quality through testing.

(c) In-depth understanding of various types of testing methodologies.

(d) Learn about testing automation and understand commonly used test automation types.

(e) Learn to use Robot framework.

Course Outcome 6 (CO5):

(a) Understand the concepts of Agile methodology.

(b) Learn to use Scrum framework for implementing Agile methodology for executing a software development process.

(c) Learn to monitor a software development project using a Scrum tool.

Course Outcome 7 (CO5):

(a) Understand the concepts of Software Configuration Management.

(b) Learn about build and deployment environments.

(c) Understand the concepts of Continuous Integration and essential practices.

(d) Understand the concepts of deployment automation and learn to use Ansible.

Model Question paper

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY
First Semester MCA Degree Examination (R&S)

Course Code: 20MCA107
Course Name: ADVANCED SOFTWARE ENGINEERING

Total Marks: 60 Duration: 3 Hours

PART A
Answer all questions, each carries3 marks. Marks

1. Why is Software Engineering important? (3)
2. What are the desired requirements of a good software engineering model? (3)
3. What is the purpose of a version control system? (3)
4. Explain the different ways to fix commits in Git (3)
5. What is anti-pattern? (3)
6. What is an abstract test? (3)
7. Distinguish between black box testing and white box testing. (3)
8. Draw a model Sprint Backlog for the login module of a simple web portal (3)

9. Write a short note on release candidate (3)
10. Differentiate continuous delivery and continuous deployment (3)

PART B

Required to answer one question from each module in full.
Each module carries 6 marks for either of the questions.

Module I

11. Prepare a basic Software Requirements Specification for Savings Bank accounts. (6)

OR

12. How is Use Case different from User Stories? Enlist the advantage of each. (6)

Module II

13. How do you create, switch and view branches in Git? explain how to merge commits between branches. (6)

OR

14. You have cloned a repository which was then modified by another developer. You make changes locally and try to execute push. What are the possible outputs? How will you solve the problems, if any? (6)

Module III

15. Explain the important design patterns. (6)

OR

16. When are assertions and expected error tests used in Unit tests? (6)

Module IV

17. Write down the scrum. (6)

OR

18. Differentiate Black box testing and White box testing. Give appropriate example for each for "only black box testing is possible" and "necessary to do white box testing" scenarios. (6)

Module V

19. Explain the strategy for implementing Continuous integration. (6)

OR

20. What is a deployment pipeline? Explain the anatomy of a deployment pipeline with a (6)
neat diagram. Comment on the various stages of a deployment pipeline.

Syllabus

Module 1 [8 hrs]

Introduction to Software Engineering: What is Software Engineering, Characteristics of Software.

Life cycle of a software system: software design, development, testing, deployment, Maintenance.

Project planning phase: project objectives, scope of the software system, empirical estimation models, COCOMO, staffing and personnel planning.

Software Engineering models: Predictive software engineering models, model approaches, prerequisites, predictive and adaptive waterfall, waterfall with feedback (Sashimi), incremental waterfall, V model; Prototyping and prototyping models.

Software requirements specification, Eliciting Software requirements, Requirement specifications, Software requirements engineering concepts, Requirements modelling, Requirements documentation. Use cases and User stories.

Module 2 [10 hrs]

Programming Style Guides and Coding Standards; Literate programming and Software documentation; Documentation generators, Javadoc, phpDocumentor.

Version control systems basic concepts; Concept of Distributed version control system and Git; Setting up Git; Core operations in Git version control system using command line interface (CLI): Clone a repository; View history; Modifying files; Branching; Push changes, Clone operation, add, commit, log, diff commands, conflict resolution. Pushing changes to the master; Using Git in IDEs and UI based tools.

Software Quality: Understanding and ensuring requirements specification quality, design quality, quality in software development, conformance quality.

Module 3 [10 hrs]

OOP Concepts; Design Patterns: Basic concepts of Design patterns, How to select a design pattern, Creational patterns, Structural patterns, Behavioural patterns. Concept of Anti-patterns.

Unit testing and Unit Testing frameworks, The xUnit Architecture, Writing Unit Tests using at least one of Junit (for Java), unittest (for Python) or phpdbg (PHP). Writing tests with Assertions, defining and using Custom Assertions, single condition tests, testing for expected errors, Abstract test.

Module 4 [10 hrs]

Concepts of Agile Development methodology; Scrum Framework.

Software testing principles, Program inspections, Program walkthroughs, Program reviews; Blackbox testing: Equivalence class testing, Boundary value testing, Decision table testing, Pairwise testing, State transition testing, Use-case testing; White box testing: control flow testing, Data flow testing.

Testing automation: Defect life cycle; Regression testing, Testing automation; Testing non-functional requirements.

Module 5[10 hrs]

Software Configuration Management: Using version control, Managing dependencies, Managing software configuration, Managing build and deployment environments.

Continuous Integration: Prerequisites for continuous integration, Essential practices.

Continuous Delivery: Principles of Software delivery, Introduction and concepts.

Build and deployment automation, Learn to use Ansible for configuration management.

Test automation (as part of continuous integration), Learn to set up test automation cases using Robot Framework.

Notes

1. At the end of Module 1, conduct the following class work with appropriate evaluation points: Prepare Software Specification Document for a moderately complex process flow system (*e.g.* Broadband fault booking and resolution system covering technical, operational and commercial aspects, covering organizational and subscriber use cases).

2. At the end of Module 2, clone an open source project using Git and perform all based operations.

Reference Books

1. Philip A. Laplante, *What Every Engineer Should Know about Software Engineering*, CRC Press [Module 1]
2. Murali Chemuturi, *Mastering Software Quality Assurance: Best Practices, Tools and Technique for Software Developers*, J Ross Publishing [Module 2]
3. Ben Straub, Scott Chacon, *Pro Git*, 2nd Edition, Apress [Module 2]
4. Erich Gamma et. al., *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley [Moule 3]
5. Vaskaran Sarcar, *Java Design Patterns: A Hands-On Experience with Real-World Examples*, Apress [Module 3]
6. Alistair Cockburn and Robert Cecil Martin, *Agile Software Development: The Cooperative Game (2nd edition)*, Addition Wesley [Module 4]

7. Ken Schwaber , *Agile Software Development with Scrum*, Pearson [Module 4]
8. Lisa Crispin, Agile Testing: *A Practical Guide for Testers and Agile Teams*, Adison Wesley
9. Paul Hamill, *Unit Test Frameworks*, O'Reilly Media [Module 4]
10. Glenford J. Myers, et. al., *The Art of Software Testing*, Wiley [Module 4, 5]
11. Lee Copeland, *A Practitioner's Guide to Software Test Design*, Artech House Publishers [Module 4, 5]
12. Jez Humble and David Farley, *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*, Pearson Education [Module 5]

Web-based Resources

1. *Git Handbook* https://guides.github.com/introduction/git-handbook/ Retrieved 8 July 2020 [Module 2]
2. *Git User Manual* https://mirrors.edge.kernel.org/pub/software/scm/git/docs/user-manual.html Retrieved 8 July 2020 [Module 2]
3. *Introduction to Software Engineering/Quality* https://en.wikibooks.org/wiki/Introduction_to_Software_Engineering/Quality Retrieved 8 July 2020 [Module 2]
4. *Understanding software design patterns* https://opensource.com/article/19/7/understanding-software-design-patterns Retrieved 8 July 2020 [Module 3]
5. *The Scrum Guide* https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf Retrieved 8 July 2020 [Module 4]
6. *unittest — Unit testing framework* https://docs.python.org/3/library/unittest.html Retrieved 8 July 2020 [Module 4]
7. What is CI/CD? https://www.redhat.com/en/topics/devops/what-is-ci-cd Retrieved 8 July 2020 [Module 5]

Course Contents and Lecture Schedule

No	Topic	No. of Lecture Hours
1	**Software Engineering**	
1.1	What is Software Engineering, Characteristics of Software Engineering	1
1.2	Life cycle of a software system	1
1.3	Project planning	1
1.4	Software Engineering Models	2
1.5	Software Requirements Specification	3
2	**Industry Best Practices**	
2.1	Programming style guides and coding standards	1
2.2	Software version control systems, basic concepts	1
2.3	Git distributed version control system, introduction	2
2.4	Common operations in Git	4

No	Topic	No. of Lecture Hours
2.5	Software quality, achieving	2
3	**System Design Methodologies**	
3.1	Object Oriented Programming	1
3.2	Software Design Patterns	4
3.3	Unit Testing concepts and xUnit architecture	1
3.4	Unit testing frameworks: Junit, unittest, phpdbg	2
3.5	Writing unit test code	2
4	**Agile Development Methodology**	
4.1	Agile Development methodology, introduction	2
4.2	Scrum framework	5
4.3	Automated testing	3
5	**Continuous Integration and Continuous Development (CI/CD)**	
5.1	Configuration Management	2
5.2	Continuous Integration, concepts and practices	2
5.3	Continuous Delivery, concepts and practices	2
5.4	Build and deployment automation	2
5.5	Test automation for CI/CD	2