CS3383: AI FOR SOFTWARE ENGINEERING

Effective Term
Semester B 2025/26

Part I Course Overview

Course Title
AI for Software Engineering

Subject Code
CS - Computer Science
Course Number
3383

Academic Unit
Computer Science (CS)

College/School
College of Computing (CC)

Course Duration
One Semester

Credit Units
3

Level
B1, B2, B3, B4 - Bachelor's Degree

Medium of Instruction
English

Medium of Assessment
English

Prerequisites
Nil

Precursors
CS3342 Software Design or equivalent

Equivalent Courses
CS3343 Software Engineering Practice

Exclusive Courses
Nil

Part II Course Details

Abstract
CS3383: AI for Software Engineering empowers students to harness the latest AI technologies and automation tools to transform the software engineering process. Through collaborative, project-based learning, students will explore how AI can

streamline requirements analysis, software design, code generation, testing, documentation, and project management. By engaging with state-of-the-art AI platforms and real-world scenarios, students will learn to critically evaluate, integrate, and manage AI-driven workflows, preparing them for the next generation of software development practices.

Course Intended Learning Outcomes (CILOs)

	CILOs	Weighting (if app.)	DEC-A1	DEC-A2	DEC-A3
1	Apply AI technologies and automation tools to enhance the software engineering lifecycle, including requirements analysis, design, code generation, testing, and documentation, to deliver high-quality software solutions.			x	x
2	Critically evaluate and integrate AI-assisted workflows and tools into software engineering projects, demonstrating awareness of their strengths, limitations, and ethical considerations.		x	x	
3	Produce clear, concise, and professional technical documentation and project deliverables by leveraging AI-powered documentation and collaboration platforms.			x	x
4	Collaborate effectively within a software development team to plan, manage, and deliver projects using AI-augmented project management and communication tools, reflecting on team dynamics and individual contributions.		x	x	x

A1: Attitude
Develop an attitude of discovery/innovation/creativity, as demonstrated by students possessing a strong sense of curiosity, asking questions actively, challenging assumptions or engaging in inquiry together with teachers.

A2: Ability
Develop the ability/skill needed to discover/innovate/create, as demonstrated by students possessing critical thinking skills to assess ideas, acquiring research skills, synthesizing knowledge across disciplines or applying academic knowledge to real-life problems.

A3: Accomplishments
Demonstrate accomplishment of discovery/innovation/creativity through producing /constructing creative works/new artefacts, effective solutions to real-life problems or new processes.

Learning and Teaching Activities (LTAs)

	LTAs	Brief Description	CILO No.	Hours/week (if applicable)
1	Lectures	Students will participate in lectures introducing the latest AI technologies, large language models (LLMs), and automation tools reshaping the software engineering landscape. Lectures will cover AI-driven approaches to requirements engineering, design, code and test generation, project management, and documentation, alongside critical issues such as AI ethics, limitations, and practical integration.	1, 2, 3, 4	
2	Tutorials and Hands-on Labs	Students will engage in hands-on tutorials and lab sessions designed to deepen their practical skills with AI-based software engineering tools and platforms. These activities will include prompt engineering, automated code and test generation, AI-assisted design, and collaborative workflow exercises. Students will also learn to critically evaluate AI-generated artifacts and develop strategies for effective and ethical AI integration.	1, 2, 3, 4	

3	Group Project	Working in teams, students will complete a substantial software engineering project that leverages AI and LLMs throughout the development lifecycle. Teams will utilize AI tools for requirements gathering, design, coding, testing, bug reporting, and documentation. Each group will clearly define member roles, maintain a project log detailing AI usage, challenges, and solutions, and produce a comprehensive final report reflecting on both technical outcomes and the impact of AI on their workflow.	1, 2, 3, 4	

Assessment Tasks / Activities (ATs)

	ATs	CILO No.	Weighting (%)	Remarks ("-" for nil entry)	Allow Use of GenAI?
1	In-Class Test	1, 2, 3, 4	15	-	No
2	Group Project	1, 2, 3, 4	35	30 Report + 5 Presentation	Yes

Continuous Assessment (%)

50

Examination (%)

50

Examination Duration (Hours)

2

Minimum Continuous Assessment Passing Requirement (%)

40

Minimum Examination Passing Requirement (%)

30

Additional Information for ATs

To pass the course, a student must achieve at least 40% of the maximum possible marks in the continuous assessment component and at least 30% of the maximum possible marks in the examination component.

For students who are unable to participate (for any reason) in a group project and choose to work and submit individually, the maximum allowable weighting for the project component is 25% instead of 35% (as a penalty).

Assessment Rubrics (AR)

Assessment Task

In-Class Test

Criterion

Criterion:
1.1 CAPACITY for SELF-DIRECTED LEARNING to understand and utilize AI-powered tools and practices in software engineering.
1.2 ABILITY to EXPLAIN AND APPLY AI techniques (such as prompt engineering, code generation, and AI-based testing) in practical software engineering scenarios.
1.3 CAPACITY to CRITICALLY EVALUATE the effectiveness and limitations of AI-generated software artifacts.

Excellent (A+, A, A-)

High

Good (B+, B, B-)

Significant

Fair (C+, C, C-)

Moderate

Marginal (D)

Basic

Failure (F)

Unacceptable

Assessment Task

Group project

Criterion

2.1 CAPACITY for SELF-DIRECTED LEARNING to integrate AI tools and automation throughout the software engineering lifecycle, and to produce technically sound and well-documented solutions.
2.2 ABILITY to EXPLAIN AND DEMONSTRATE IN DETAIL the integration of AI techniques in project management, collaboration, and workflow optimization within a team environment.
2.3 ABILITY to APPLY AND REFLECT ON the strengths, limitations, and ethical implications of AI-driven software engineering practices in a collaborative project setting.

Excellent (A+, A, A-)

High

Good (B+, B, B-)

Significant

Fair (C+, C, C-)

Moderate

Marginal (D)

Basic

Failure (F)

Unacceptable

Assessment Task

Examination

Criterion

3.1 CAPACITY for SELF-DIRECTED LEARNING to understand, compare, and critique AI-powered software engineering tools and methodologies.
3.2 ABILITY to EXPLAIN AND APPLY concepts and best practices for using AI in requirements analysis, design, code generation, testing, and project management.
3.3 CAPACITY to ANALYZE AND EVALUATE the ethical, professional, and practical challenges associated with AI-augmented software engineering.

Excellent (A+, A, A-)

High

Good (B+, B, B-)

Significant

Fair (C+, C, C-)

Moderate

Marginal (D)

Basic

Failure (F)

Unacceptable

Part III Other Information

Keyword Syllabus

Generative Ai for Software Engineering, AI-Augmented Software Engineering, Large Language Models (LLMs), AI-Driven Code Generation, Automated Software Testing with AI, AI in Project Management, AI-Assisted Requirement Engineering and Design, AI-Enhanced Object-Oriented Modeling, AI-Driven Refactoring and Debugging, Technical Documentation Automation, AI Ethics in Software Engineering, Prompt Engineering, Human-AI Collaboration, Web-based Application Development with GenAi.

Syllabus

- **AI-Augmented Software Engineering**
  - Introduction to AI and LLMs in software engineering
  - Overview of current AI tools (e.g., GitHub Copilot, ChatGPT etc.)
  - Human-AI collaboration dynamics
- **Automated Software Testing and Debugging with AI**
  - AI-powered test case generation (e.g., with LLMs)
  - Test coverage and comprehensiveness using AI tools
  - AI-assisted debugging, bug detection, and reporting
- **AI-Enhanced Software Project Management**
  - Agile practices and AI integration
  - Sprint planning and backlog management with AI
  - AI-assisted version control and collaboration
- **Requirement Engineering and Design with AI**
  - Eliciting and specifying requirements with AI support
  - Prototyping and design generation using AI tools
  - Object-oriented modeling and architecture with AI assistance
- **Coding, Implementation, and Refactoring**

- AI-generated code and code review
- Refactoring and code quality assessment using AI
- Modern IDEs and AI plugin integration

- **Technical Documentation Automation**
  - AI-generated documentation and code comments
  - Knowledge management and doc quality review
- **Ethics, Limitations, and Professional Practice**
  - Ethical issues in AI-augmented SE
  - Bias, transparency, and reliability in AI-driven workflows
  - Best practices for the responsible use of AI in software projects
- **Project Presentation and Peer Review**
  - Group project demos and reflective presentations
  - Peer review of AI-generated artifacts

Reading List

Compulsory Readings

	Title
1	Nil

Additional Readings

	Title
1	Sommerville, I. (2020). Software Engineering. 10th Edition, Addison Wesley. [Classic SE reference]
2	Russell, S., & Norvig, P. (2021). Artificial Intelligence: A Modern Approach. 4th Edition, Pearson. [AI foundations]
3	Amershi, S., et al. (2019). "Software Engineering for Machine Learning: A Case Study." In Proceedings of ICSE, IEEE. [AI in SE practice]
4	Allamanis, M., Barr, E. T., Devanbu, P., & Sutton, C. (2018). "A Survey of Machine Learning for Big Code and Naturalness." ACM Computing Surveys, 51(4). [ML for code]
5	Pearce, H., et al. (2021). "Asleep at the Keyboard? Assessing the Security of GitHub Copilot’s Code Contributions." arXiv preprint arXiv:2108.09293. [AI code generation risks]
6	Bender, E. M., & Koller, A. (2020). "Climbing towards NLU: On Meaning, Form, and Understanding in the Age of Data." ACL. [Understanding LLMs]
7	Rajan, A., et al. (2023). "AI-Augmented Software Engineering: Opportunities and Challenges." IEEE Software, 40(2), 16-23. [Overview article]
8	Larman, C. (2012). Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development. 3rd Edition, Pearson.
9	Christiansen, M., & Thayer, R. H. (2002). A Manager's Guide to Software Engineering’s Best Practice. Wiley–IEEE Computer Society.
10	OpenAI documentation: https://platform.openai.com/docs/
11	GitHub Copilot documentation: https://docs.github.com/en/copilot