STATE UNIVERSITY OF NEW YORK
COLLEGE OF TECHNOLOGY
CANTON, NEW YORK

MASTER SYLLABUS

COURSE NUMBER: ELEC 388
COURSE TITLE: SOFTWARE ENGINEERING

CIP Code: 14.1001

Created by: Dr. Stephen Frempong
Updated

School: SCHOOL OF ENGINEERING TECHNOLOGY
Department: ELECTRICAL ENGINEERING TECHNOLOGY & ENGINEERING SCIENCE
Implementation Semester/Year: FALL 2027

A. COURSE TITLE: Software Engineering

B. COURSE NUMBER: ELEC 388

C. CREDIT HOURS (# Hours of Lecture, Laboratory, Recitation, Clinical):

# Credit Hours per Week	3
# Lecture Hours per Week	3
# Lab Hours per Week	
Other per Week	

D. GRADING METHOD:

A – F	☑
Pass/Fail	☐
Other:	

E. WRITING INTENSIVE COURSE:

Yes	☐
No	☑

F. GER CATEGORY:

Does the course satisfy a GER category? If so, please select all that apply.
(https://www.canton.edu/provost/assessment/ger/)

[1-2] Communication	☐
[3] Diversity: Equity, Inclusion & Social Justice	☐
[4] Mathematics & Quantitative Reasoning	☐
[5] Natural Science & Scientific Reasoning	☐
[6] Humanities	☐
[7] Social Sciences	☐
[8] Arts	☐
[9] US History & Civic Engagement	☐
[10] World History & Global Awareness	☐
[11] World Languages	☐

CORE COMPETENCIES (Required starting in Fall 2026**):**

[12] Critical Thinking and Reasoning	☑
[13] Information Literacy	☐
[14] Civic Discourse	☐

G. APPLIED LEARNING COMPONENT (High-Impact Practices):

Yes	☐
No	☑

If Yes, select [X] one or more of the following Curricular Attribute categories:
HIPs definitions found here: https://www.suny.edu/applied-learning/resources/

Capstone	
Creative Works	☐
For-Credit Internship	☐
Practicum	
Practicum [Clinical Placement]	☐
Practicum [Non-Clinical Placement]	☐
Research & Field Study	
Field Research	☐
Research	☐
Undergraduate Research	☐
Service or Community	
Service Learning	☐
Community Service	☐
Civic Engagement	☐
Study Abroad	
International and Domestic Travel/Exchange	☐
COIL	☐

H. SEMESTER(S) OFFERED:

Fall	☐
Spring	☑
Fall and spring	☐

I. **COURSE DESCRIPTION:** This course introduces the principles and practices of software engineering used to design, develop, test, and maintain software systems. Topics include the software development life cycle, requirements analysis, system design, testing, quality assurance, and project management. Students gain practical experience through team-based projects and the use of modern development tools and methodologies**.**

J. **PRE-REQUISITES:** ENGS 102 Programming for Engineers, or CITA 180 Introduction to Programming, and CITA 380 Integrated Programming for Engineers, or CITA 342 Visual Programming, and MATH 162 (Calculus II), or permission of instructor.

CO-REQUISITES: None

K. LEARNING OUTCOMES:

SLO Statement	ABET/ PLO	ISLO	Subset	GER
1. Explain the software development life cycle (SDLC) and its phases,	7. an ability to acquire and apply new knowledge as needed, using appropriate learning strategies	5. Industry, Professional, Discipline-Specific Knowledge and Skills.		
2. Analyze user requirements and translate them into clear software specifications	6. an ability to develop and conduct appropriate experimentation, analyze and interpret data, and use engineering judgment to draw conclusions	3. Foundational Skills	PS	
3. Apply software design principles and architectural patterns to develop maintainable systems.	6. an ability to develop and conduct appropriate experimentation, analyze and interpret data, and use engineering judgment to draw conclusions.	2. Critical Thinking 5. Industry, Professional, Discipline-Specific Knowledge and Skills.	PS	
4. Apply modern software development tools such as version control systems and issue tracking tools.	7. an ability to acquire and apply new knowledge as needed, using appropriate learning strategies	5. Industry, Professional, Discipline-Specific Knowledge and Skills	CA	
5. Implement testing and debugging techniques to ensure software quality.	7. an ability to acquire and apply new knowledge as needed, using appropriate learning strategies	2. Critical Thinking 5. Industry, Professional, Discipline-Specific Knowledge and Skills.	PS	

6. Work effectively in team-based software development environments.	5. an ability to function effectively in a team whose members together provide leadership, create a collaborative environment, establish goals, plan tasks, and meet objectives.	4. Social Responsibility	Teamwork	
7. Apply project management practices including planning, scheduling, and risk management.	1. an ability to identify, formulate, and solve complex engineering problems by applying principles of engineering, science, and mathematics.	2. Critical Thinking 5. Industry, Professional, Discipline-Specific Knowledge and Skills.	PS	
8. Apply professional, ethical, and documentation standards in software development.	7. an ability to acquire and apply new knowledge as needed, using appropriate learning strategies.	5. Industry, Professional, Discipline-Specific Knowledge and Skills.		

L. TEXTS: To be determined by instructor

M. SUGGESTED INSTRUCTIONAL MATERIALS:

David C. Kung, Software Engineering, 2/Edition
Publisher: McGraw Hill
ISBN: 9781264556953

KEY	
SLO	**Student Learning Outcomes**
PLO	**Program Learning Outcome**
ISLO	**Institutional Student Learning Outcomes [ISLO 1 – 5]**
ISLO #	**ISLO and Subsets**
1	Communication Skills: • Oral [O] • Written [W]
2	Critical Thinking: • Critical Analysis [CA] • Inquiry & Analysis [IA] • Problem Solving [PS]
3	Foundational Skills: • Information Management [IM] • Quantitative Lit, /Reasoning [QTR]
4	Social Responsibility • Ethical Reasoning [ER] • Global Learning [GL] • Intercultural Knowledge [IK] • Teamwork [T]
5	Industry, Professional, Discipline Specific Knowledge and Skills
GER	**General Education Requirements**: Refer to Listing, Section F

N. EQUIPMENT: University Supplied Equipment / Learning Management System

O. SUGGESTED MEASUREMENT CRITERIA/METHODS:

Quiz	☐
Exam	☑
Assignment	☑
Other:	☑

P. DETAILED COURSE OUTLINE:

Introduction to Software Engineering

- Overview of software engineering
- Importance of software processes
- Software engineering ethics and professionalism

Software Development Life Cycle (SDLC)

- SDLC models (Waterfall, Incremental, Spiral)
- Agile methodologies overview
- Comparison of development models

Agile Software Development

- Agile principles and values
- Scrum framework (roles, artifacts, ceremonies)
- Agile vs traditional approaches

Requirements Engineering

- Requirement types (functional and non-functional)
- Requirement elicitation techniques
- Stakeholder analysis

Requirements Analysis and Specification

- Writing Software Requirement Specifications (SRS)
- Use cases and user stories
- Requirements validation

Software Design Fundamentals

- Design principles (modularity, abstraction, cohesion, coupling)

- Software architecture basics
- Design documentation

Modeling with UML

- UML diagrams (use case, class, sequence, activity diagrams)
- Modeling system behavior and structure

Software Architecture and Design Patterns

- Architectural styles (layered, client-server, microservices)
- Introduction to design patterns

Implementation and Coding Practices

- Coding standards and documentation
- Version control systems (e.g., Git)
- Collaborative development

Software Testing

- Testing levels (unit, integration, system, acceptance)
- Test case design
- Automated testing basics

Software Quality Assurance

- Quality models and metrics
- Code reviews and inspections
- Continuous integration

Software Project Management

- Project planning and scheduling
- Risk management
- Cost estimation

Q. LABORATORY OUTLINE: None