CS4389: DECENTRALIZED APPLICATIONS DEVELOPMENT

Effective Term
Semester A 2024/25

Part I Course Overview

Course Title
Decentralized Applications Development

Subject Code
CS - Computer Science
Course Number
4389

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
CS3343 Software Engineering Practice

Precursors
CS2204 Fundamentals of Internet Applications Development

Equivalent Courses
Nil

Exclusive Courses
Nil

Part II Course Details

Abstract

Decentralized applications (Dapps) are software applications that are run by multiple users on decentralized networks. Dapps have found successful usage scenarios in many domains such as software development, finance, customer services, and logistics. Many Dapps scenarios include the use of a kind of programs called smart contracts to perform transactions involving multiple users. The aim of this course is to provide a comprehensive study on the software design and development of Dapps, its associated software engineering practices, programming languages, development and testing environments, tools, evaluation, and current trends and issues in the aspect of software engineering. Students are expected to design, implement, test, maintain and evaluate programs that meet the constraints and requirements of high quality decentralized applications.

Course Intended Learning Outcomes (CILOs)

	CILOs	Weighting (if app.)	DEC-A1	DEC-A2	DEC-A3
1	Describe the essential requirements, constraints and technology stack in developing Dapps.		x		
2	Demonstrate working knowledge on the development, test, and deployment environments of Dapps.			x	
3	Design, implement and maintain Dapps.				x
4	Evaluate the correctness and performance of Dapps.		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 engage in lectures to gain knowledge about the essential constraints, requirement knowledge, system models, and frameworks for the development and deployment of Dapps.	1, 2, 3, 4	3 hours/week

2	Tutorials	Students will engage in tutorials to discuss and practice various skills in Dapps software development. They will use a series of hands-on practices on developing and testing a Dapp from scratch to strengthen their knowledge in the field.	1, 2, 3, 4	8 hours/semester
3	Quiz	Students will answer questions in a quiz that will cover all topics learned in lectures and practices gained via tutorials, as well as the working knowledge in setting up the decentralized applications' environment in the group project.	2, 3, 4	
4	Project	Students will participate in groups to conduct a comprehensive software engineering project to learn how to collaborate and share in their learning process in a software development context. They will practice all major topics from design to coding and testing learned in the course.	1, 2, 3, 4	
5	Presentation	Students will deliver presentations that cover all aspects of the project, including design decisions and rationale, as well as the justification, implementation, and evaluation of the project.	2, 3, 4	

Assessment Tasks / Activities (ATs)

	ATs	CILO No.	Weighting (%)	Remarks (e.g. Parameter for GenAI use)
1	Quiz	2, 3, 4	15	
2	Project	1, 2, 3, 4	40	
3	Presentation	1, 2, 3, 4	5	

Continuous Assessment (%)

60

Examination (%)
40

Examination Duration (Hours)
2

Additional Information for ATs
For a student to pass the course, at least 30% of the maximum mark for the examination must be obtained.

Assessment Rubrics (AR)

Assessment Task
Quiz

Criterion
1.1 ABILITY to describe, analyse and apply software engineering techniques and write programs and tests for decentralized applications

Excellent (A+, A, A-)
High

Good (B+, B, B-)
Significant

Fair (C+, C, C-)
Moderate

Marginal (D)
Basic

Failure (F)
Not even reaching marginal levels

Assessment Task
Project

Criterion
2.1 ABILITY to set up and apply software engineering environment, tools, techniques, practices, and programming languages to develop and deploy decentralized applications
2.2 ABILITY to function as a team of developers
2.3 ABILITY to report in an organised and logical way

Excellent (A+, A, A-)
High

Good (B+, B, B-)
Significant

Fair (C+, C, C-)
Moderate

Marginal (D)
Basic

Failure (F)
Not even reaching marginal levels

Assessment Task
Presentation

Criterion
3.1 ABILITY to summarize and present complex technical ideas systematically

Excellent (A+, A, A-)
High

Good (B+, B, B-)
Significant

Fair (C+, C, C-)
Moderate

Marginal (D)
Basic

Failure (F)
Not even reaching marginal levels

Assessment Task
Examination

Criterion
4.1 ABILITY to explain software development and deployment technology stack
4.2 ABILITY to apply software design techniques and write code for decentralized applications
4.3 ABILITY to ensure the correctness and performance of decentralized applications

Excellent (A+, A, A-)
High

Good (B+, B, B-)
Significant

Fair (C+, C, C-)
Moderate

Marginal (D)
Basic

Failure (F)
Not even reaching marginal levels

Part III Other Information

Keyword Syllabus

Decentralized applications, blockchain, blockstack, transactions, smart contract, wallet, testing, debugging, performance, programming language, tools and frameworks.

Syllabus

- Technology stack of decentralized applications
  Overview of different kinds of application styles including centralized, decentralized and distributed. Technology stacks including decentralized filesystem, decentralized networks, software architecture, programming language, platform, wallet, virtual machine, API framework and library, development and testing environments.
- Problem solving through decentralized application programming
  Solidity programming language. Implementation of wallet, transaction, smart contract, and backend and frontend application logics.
- Software correctness and performance
  Analysis, code review, debugging, testing, fuzzing, and maintenance.

Reading List

Compulsory Readings

	Title
1	Nil

Additional Readings

	Title
1	Ethereum Smart Contract Development: Build blockchain-based decentralized applications using solidity. Mayukh Mukhopadhyay. Packt Publishing Ltd, 2018. ISBN 1788472624.
2	Introducing Ethereum and Solidity: Foundations of Cryptocurrency and Blockchain Programming for Beginner. Chris Dannen. Apress; 1st ed. Edition, 2017. ISBN 1484225341.
3	Solidity. https://github.com/ethereum/solidity and https://solidity.readthedocs.io/en/v0.4.24/
4	Ethereum. https://www.ethereum.org/
5	Decentralized applications: Harnessing bitcoin’ s blockchain technology. Siraj Raval. O’ Reilly. 2016. ISBN 9781491924549.