University at Buffalo
The State University of New York

CSE426/526: Blockchain Application Development

COURSE INFORMATION

3 Credits
Lecture – 3 credits

INSTRUCTOR INFORMATION

Dr. Bina Ramamurthy

- *Email:* bina@buffalo.edu
- *Webpage:* http://www.cse.buffalo.edu/~bina
- *Office:* Davis 345
- *Office hours:* Mon and Tue: 2.00-3.30PM

COURSE DESCRIPTION

This course is intended for students interested in learning about blockchain technology and in developing applications using the blockchain concepts. It begins with the definition of the blockchain as trust layer over the internet for working with distributed resources with decentralized and disintermediated control. Topics include: Definition of a blockchain in terms of transactions, blocks and chain of blocks, data structures enabling the blockchain protocol and operational details involving algorithms and techniques such as peer-to-peer transactions, cryptography, digital signing and hashing, and consensus mechanisms. All of these concepts will be illustrated using Bitcoin and Ethereum blockchain. In the second part of the course, we introduce the concept of code execution the blockchain and the program module called smart contract and a language, Solidity, for writing smart contracts, compiling, deploying and testing the smart contracts on Ethereum blockchain. In the last part of the course, we introduce a decentralized application (Dapp) stack and explore problem solving using blockchain. This involves design and development of a Dapp stack with the computational logic represented by the smart contract code, a user interface and support for off-chain data access, and decentralized file systems. Students will work on hands-on end-to-end Dapp projects using Ethereum blockchain and Truffle integrated development environment (IDE). The course will also discuss standards, best practices, and current challenges, such as scalability and interoperability, and the respective solutions.

On completion of the course, a student will be able to analyze a problem, develop, and implement an end-to-end blockchain solution on Ethereum blockchain.

Course Prerequisites: CSE250 Data Structures, equivalent, or permission of the instructor

COURSE ORGANIZATION / SCHEDULE

Week#	Description
1	History of blockchain; Blockchain as a method for enabling peer-peer transactions; blocks of transactions, chain of blocks and distributed ledger technology; explore transactions on real-world blockchain.
2	Bitcoin protocol: UTXO (unspent transaction output) model; Mining, verification and validation; and proof of work; Ethereum protocol: concept of account and addresses; incentive model and the cost of trust; immutability and anonymity.
3	Issues: block size, timestamp, scalability, hard and soft fork, governance.
4	Code execution on the blockchain; Introduction to smart contract and Ethereum Virtual Machine (EVM); alternative implementations of smart contract in chain code; application-specific validation and verification.

5	Introduction to Solidity, a Turing-complete language for writing smart contracts: Basic data types, blockchain-specific data structures such as mapping, address, message.
6	Problem solving using finite state machine design and transforming the design into smart contract; Solidity programming details: access modifiers and function modifiers. Defining events and pushing notifications and listeners.
7	Remix integrated development environment (IDE) for smart contract development and testing.
8	Best practices in development of smart contracts: when to use blockchain and what to store on the blockchain? Privacy, integrity, and confidentiality.
9	Blockchain server concept; setting up a blockchain network; working of a blockchain node; go-Ethereum node (geth); Ethereum APIs: web3 API for integrating traditional web to a blockchain network; Ropsten public network;
10	Definition of decentralized application stack; Truffle development environment; End-end application development process using Truffle; test-driven application development.
11	Off chain and on-chain data. Infura: blockchain as a service; Interplanetary File system (IPFS) for decentralized data
12	Application models, distributed apps (Dapps), side chains; Use of Multi-sig, escrow, oracles.
13	Blockchain platforms: Linux foundation's Hyperledger project: IBM's Fabric; Intel Sawtooth; Use cases in healthcare and Fintech application domains.
14	Blockchain case studies – throughout the semester
15	Advanced topics

STUDENT LEARNING OUTCOMES

Course Learning Outcome	**Program Outcomes (CS / CEN)**	**Assessment Method(s)**	**Assessment types**
Explain the history of blockchain	1(CS); 1(CEN)	Summative quiz, Exam	Quizzes, Exam
Explain a transaction, block and blockchain and distributed ledger.	1(CS); 1(CEN)	Summative quiz, Exam	Quizzes, Exam
Discuss how immutability and consensus are achieved, and the transaction verification process.	2(CS); 2(CEN)	Summative quiz, Exam	Quizzes, Exam
Design a blockchain solution using a FSM design, implement a smart contract for the design using Solidity and test it on the Remix IDE.	1,2 (CS); 1,2 (CEN)	Graded lab, Exam	Programming assignments, exam
Analyze a problem and implement a decentralized application and deploy it on a test blockchain	1,2(CS); 1,2(CEN)	Graded lab, Exam	Programming assignments, exam

Program outcomes correspond to student outcomes provided by ABET: Blank: no coverage, 1: introduced 2: practices or reinforced 3: mastered

TEXT BOOK: Blockchain in Action, Bina Ramamurthy, Manning Publishing, https://www.manning.com/books/blockchain-in-action, ISBN: 9781617296338, 2020.

COURSE REQUIREMENTS

- Students will be assessed through quizzes, graded lab assignments and exams and attendance.
- **Exams:** There will be two exams, one midterm and another a comprehensive final exam.
- **Lab Assignments**: Students will setup, test, deploy and transact on a blockchain as well as design, program and test a smart-contract on this blockchain. They will work on problem solving using blockchain by designing, developing and testing a Dapp.

- **Quizzes**: There will be at least 4 quizzes throughout this course on weeks when there are no graded labs. Quizzes will be used to assess students on learning outcomes that are not amenable to be assessed through programming assignments.
- **Attendance is very important and is assessed through various mechanisms.**

GRADING POLICY

Learning assessments will be graded based on rubric criteria and weighted according to the following break-down.

Weight	Assessment / Assignment
45%	Exams (2) – 20 +25
45%	Lab assignments (Programming projects) (3)
10%	Quizzes (4) including class attendance

Final Grades:

Grade	Quality Points	Percentage
A	4.0	93.0% -100.00%
A-	3.67	90.0% - 92.9%
B+	3.33	87.0% - 89.9%
B	3.00	83.0% - 86.9%
B-	2.67	80.0% - 82.9%
C+	2.33	77.0% - 79.9%
C	2.00	73.0% - 76.9%
C-	1.67	70.0% - 72.9%
D+	1.33	67.0% - 69.9%
D	1.00	60.0% - 66.9%
F	0	59.9 or below

Incompletes (I/IU): A grade of incomplete ("I") indicates that additional course work is required to fulfill the requirements of a given course. Students may only be given an "I" grade if they have a passing average in coursework that has been completed and have well-defined parameters to complete the course requirements that could result in a grade better than the default grade. An "I" grade may not be assigned to a student who did not attend the course. Incompletes (I/IU): Prior to the end of the semester, students must initiate the request for an "I" grade and receive the instructor's approval. Assignment of an "I" grade is at the discretion of the instructor.

ACADEMIC INTEGRITY

Academic integrity is a fundamental university value. Through the honest completion of academic work, students sustain the integrity of the university while facilitating the university's imperative for the transmission of knowledge and culture based upon the generation of new and innovative ideas. See the link for more details: https://catalog.buffalo.edu/policies/academic_integrity_2019-20.html

EXAMS

There will be a midterm that will be administered and graded before the resign date. Exam1 - **10/9/2019** during the lecture time will cover all lecture and reading assignments before the exam, as well as concepts from the lab assignments. The Exam2 - **12/9/2019: 8-11AM, NSC 215** is a comprehensive exam, covering all lecture, lab, and homework areas. All exams are closed book, closed notes, and closed neighbor. We do not give make up exams for any reason. If you miss an exam, you will receive a zero for that portion of the grade.

PROJECTS

The due date for each project will be announced when it is assigned. All the source code, documentation, makefile, data files, and README files are to be submitted on-line. The details of how to submit given along with your first project. You

will have to follow the rules for the other projects too. I reserve the right to change the project specifications at any point before the due date to address problems that may arise during the course of the project. If your design is modular the changes will not be difficult to implement. A detailed grading guideline will be given to you along with the project specification. Use this as a guide for your design and implementation. It is necessary to keep up with the programming projects in the class. There will be a 25 percent deduction for each day the project is late after the due date.
Develop your code using the ***Incremental Development technique.*** Do not try to sit down and code the entire assignment in one sitting. Instead, take one section at a time, implement, test it, back up the code, and move on to the next section. You will turn in each lab before 11:59 PM on the due date via the departmental ***submit*** command. You must also include appropriate testing programs to show the validity of your solution. In addition, you must include external documentation discussing the "how's and why's" of your design and implementation. You will be required to demonstrate your lab to your TA. The TA will also run test examples against your code to check your solution's overall correctness. The TA will provide a demo schedule. It is your responsibility to demo your project, or you will receive a zero for that portion of the grade.

OFFICE HOUR POLICY

If you can't meet during these hours, you will have to communicate with us via Email. Office hours are intended to resolve questions about the material that could not be answered in lecture or recitation. Come to office hours prepared! Office hours are NOT for the following: to repeat missed lecture material, to repeat missed recitation material or to have the instructor or TA solve an assigned problem for you. During office hours, we will NOT write or debug your code for you! Instead, we will direct you as to where to concentrate your debugging efforts.

GRADING POLICY

All assignments will be graded and returned in a timely manner. When an assignment is returned, you will have a period of one week to contest any portion of the grade. The TA who graded your assignment will be the first person to resolve a grading conflict. If the conflict cannot be resolved, the instructor will mediate the dispute. The judgment of the instructor will be final in all such cases. When contesting a grade, you must be able to demonstrate how your particular solution is correct. Also, when contesting a grade, the instructor or TA reserves the right to re-evaluate the entire lab or exam, not just the portion in dispute.

ACCESSIBILITY RESOURCES

If you have any disability which requires reasonable accommodations to enable you to participate in this course, please contact the Office of Accessibility Resources, 60 Capen Hall, 645-2608, and also the instructor of this course. The office will provide you with information and review appropriate arrangements for reasonable accommodations. See http://www.buffalo.edu/studentlife/who-we-are/departments/accessibility.html

COURSE MATERIALS : Course Website: https://www.cse.buffalo.edu/~bina/cse426/fall2019