EE 461L - Software Engineering and Design Laboratory
Course Syllabus: Unique Sections 17680, 17685, 17690, 17695, 17700, 17705
Spring 2022

Version 1.0 01/18/2022

Instructor:

Dr. Abhay Samant
Email: abhay.samant @utexas.edu (the best way to reach me outside of office hours)
Office Location: Online
Office hours: W 4:00-5:00 pm (online Zoom meeting) for 17680, 17685, 17690
Th 4:00-5:00 pm (online Zoom meeting) for 17695, 17700, 17705

Teaching Assistants:

Evan King (e.king@utexas.edu)
Taniya Bhosale (taniya@utexas.edu)
Santhosh Saravanan (sharnushtak@utexas.edu)

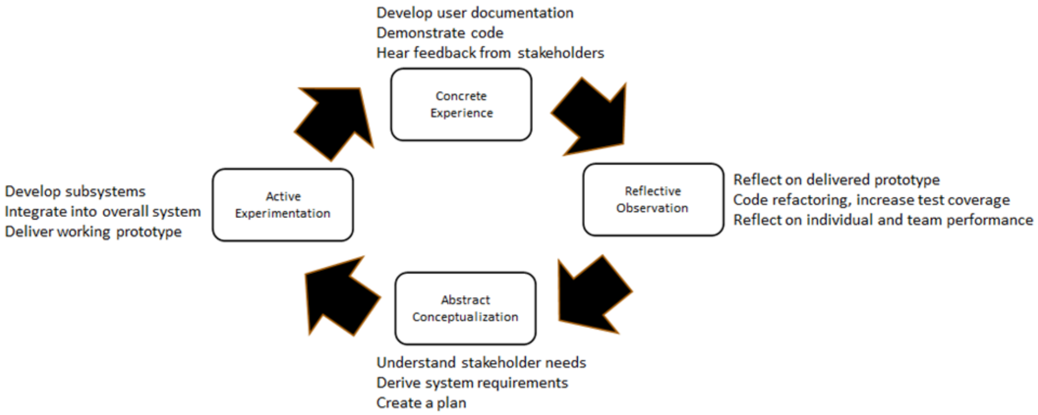
Time and TAs:

Section	Class Time	Lab Time	TA
17680	MW 5-6:30	T 3:30-6:30	Taniya
17685	MW 5-6:30	Th 3:30-6:30	Evan
17690	MW 5-6:30	T 6:30-9:30	Santhosh
17695	TTH 5-6:30	F 12:00-3:00	Evan
17700	TTH 5-6:30	W 6:30-9:30	Santhosh
17705	TTH 5-6:30	Th 6:30-9:30	Taniya

Course Description

This course focuses on providing hands-on experience in designing and developing large-scale software systems. Specifically, the course studies tools and techniques that enable large-scale software development. Specific topics include software development life cycle, model-based system engineering, version control systems, testing tools, tools for debugging, object-oriented design and analysis, design patterns and refactoring.

The team project, a key component of the course, is based on principles of David Kolb's cycle of experiential learning, as shown in the figure below.


Develop user documentation
Demonstrate code
Hear feedback from stakeholders
Concrete Experience
Reflective Observation
Reflect on delivered prototype
Code refactoring, increase test coverage
Reflect on individual and team performance
Abstract Conceptualization
Understand stakeholder needs
Derive system requirements
Create a plan
Active Experimentation
Develop subsystems
Integrate into overall system
Deliver working prototype

Experiential learning is an engaged learning process whereby students "learn by doing" and by reflecting on the experience. As per David Kolb, learning that is considered "experiential" contain all the following elements:

1. Reflection, critical analysis and synthesis.
2. Opportunities for students to take initiative, make decisions, and be accountable for the results.
3. Opportunities for students to engage intellectually, creatively, emotionally, socially, or physically.
4. A designed learning experience that includes the possibility to learn from natural consequences, mistakes, and successes.

Course Objectives

Teach practical software development methods and tools in the context of developing software applications in collaboration with team members.

- Practical software design, construction, and evolution methods
    - Software Development Life Cycle (SDLC)
    - Requirements Gathering and Feature Definition methodology
    - Software Design Patterns (Traditional and Novel)
    - GNU/Linux operating system environment, command-line utilities, bash shell scripting
    - Code refactoring
    - Object-oriented programming
- Practical software development tools
    - Tool support for collaboration (GIT, Jira/Confluence)
    - SW development using Python
    - Unit testing and regression testing, test coverage and test generation (pytest)
    - Web application development and hosting. Front-end development tools (React.js), Back-end development tools (Python Flask)
    - Database management and programming (MongoDB NoSQL programming)

Learning Outcomes

By the end of the course, students will be able to

- Identify different phases in the Software Development Life Cycle (SDLC)
- Recognize the necessity of agile development
- Demonstrate the use of object-oriented programming techniques for scalable and modular software applications
- Identify the right tools for front-end, back-end, database, and cloud deployment for a scalable web application
- Develop software that meets system requirements and satisfies stakeholder needs.
- Construct a project plan for working in cross-functional teams on collaborative software design and development
- Formulate a testing and code refactoring strategy for long term supportability of the designed software
- Prioritize work items to deliver large-scale software systems in a timely manner
- Analyze and reflect on the project to understand what worked well and what can be improved upon further
- Critique self and team member performance with the objective of continuous improvement
- Justify how the developed software adds value to the stakeholders and addresses their needs

Prerequisites

EE 422C or CS 336 with a grade of at least C-; credit with a grade of at least C- for M325K.

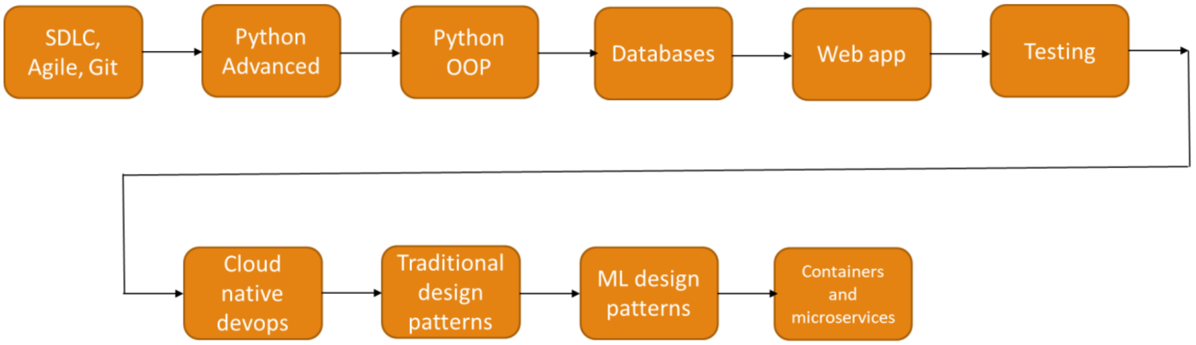
Course Materials

All course materials will be made available through Canvas. These materials are shared solely for the purpose of meeting the course objectives, and access to the materials should be restricted to registered students in the course. Sharing or distributing lecture notes or other course-specific materials in any other way is not approved and is a violation of both the University of Texas Honor Code and in some cases, copyright law.

Textbooks

There are no required textbooks.

Syllabus and Schedule



Wk	Date	Class Topic	Component	Points
1	17-Jan	Welcome, Introductions, Syllabus, Source Code Management (git)		
2	24-Jan	Software Development Life Cycle (SDLC)	HW1	6
		Project Discussions, Overview of all tools needed		
		Agile Software Development		
4	31-Jan	Python Programming	Project Teams Finalized	
		FileIO, Lists, Dictionaries, Iterators, Generators. Object Oriented Programming		
5	7-Feb	Python Programming	HW2	7
		Inheritence, Polymorphism, UML Diagrams, Class Definition, calling external modules (cryptography, retrieving data from public datasets)		
6	14-Feb	Databases	HW3	7
		NoSQL (MongoDB), Distributed data storage (MapReduce), Difference between SQL and NoSQL	Project Checkpoint1	10
7	21-Feb	Web Development	HW4	6
		Introduction to toolchain, HTML/CSS, Frontend (React.js)		
8	28-Feb	Web Development	HW5	6
		Backend (Python Flask), Cloud Deployment (GCP, Heroku)		
9	7-Mar	Midterm	Midterm	20
	14-Mar	Spring Break		
10	21-Mar	Testing Frameworks (pytest), Information Hiding, Refactoring	HW6	6
11	28-Mar	Cloud Native Devops (CICD, Containers)	HW7	6
12	4-Apr	Traditional Design Patterns (Factory, Singleton, Command, Adapter)	Project Checkpoint2	10
13	11-Apr	Machine Learning Design Patterns	HW8	6
14	18-Apr	Cloud Native Devops (Industry Perspective)		
15	25-Apr	Project Presentations	Project Checkpoint3	10
				100
16	2-May			

Labs

There will be weekly homework assignments in most of the laboratory sections. You should be able to complete most of them during your lab session time. These assignments will be submitted for grading on Canvas. Attendance in your lab section is highly recommended.

The primary purpose of the labs is for working through tutorials. You are expected to attend your lab meetings, and unless otherwise specified, to work on the weekly tutorial during your lab. The first 1 1/2 hours of every lab period is highly recommended, and you must attend the lab for which you are registered. The last half of each three-hour lab session is allocated for TA office hours, and any student may attend.

HW Late Turn-in Policy

Homework is due on the date mentioned on the Canvas assignment page for that homework assignment. Canvas submissions will be disabled after the due date. Home late turn-in policy is as follows

0 to 4 hours after the due date: 25% penalty
4 to 8 hours after the due date: 50% penalty
8 to 12 hours after the due date: 75% penalty
12 to 24 hours after the due date: 90% penalty

24 hours after due date: 0 points awarded

Please email your homework after the due date to your TA (not instructor). Please put your instructor's email in the cc field.

Project

There will be one project with three checkpoints. Throughout the semester, you will work in a team on an extensive software project. It is not necessarily the case that all members of a given team will receive the same project grade. Each project phase will receive a score, and individual scores may be, at the instructor's discretion, weighted/reduced based on the contributions of the team member as indicated by the peer evaluations.

Exam

There will be one mid-term exam during the semester. There will not be a final exam. The mid-term exam will be worth 20% of your course grade will occur approximately mid-semester. Any material from the course is considered fair game for the exams.

Assignment and Exam Regrade Requests

The grade you are given on an exam, assignment or as your final course grade is not the starting point of a negotiation. It is your grade unless a concrete error has been made. Do not ask for a better grade because you want one or you feel you deserve it. Request a regrade only if you can document a specific error in grading or in recording your scores. Keep in mind that errors can be made either in your favor or not. It's possible that if you ask to have a piece of work regraded, your grade will go down rather than up. Regrade requests will be considered for only one week after the assignment is returned. This is defined as the date on which the graded assignment was returned to you or the date the assignment grades are released on Canvas (whichever is earliest.) The entire assignment or exam will be regraded. Please pay attention to your grades on Canvas, and respond promptly to any emails sent to you by the instructor or TAs. Regrade requests must be sent via email on Canvas to the instructor and all TAs, and will only be considered if supporting documentation which explains the grading error is included. For group work, regrade request submissions must be agreed upon and submitted by the entire group since it affects every group member's grade.

Remember that the most important characteristic of any grading scheme is that it be fair. Keep this in mind if you're thinking of asking, for example, for more partial credit points on a problem. The important thing is not the exact number of points that were taken off for each kind of mistake. The important thing is that the number was the same for everyone. Partial credit decisions won't be changed once grading is done and the exams and assignments are returned.

Final Grades

Final grades will be assigned according to the following standard criteria:

Final Average	Letter Grade
90-100	A
80-89	B
70-79	C
60-69	D
0-59	F

There is a possibility that the final grade cutoffs will be lower than the criteria discussed above, but this will not be determined until the end of the semester. Plus and minus grades will be used for course averages near a letter grade cutoff at the instructor's discretion. Nonacademic explanations for poor class performance will have no bearing on the assignment of grades.

Communication

Piazza is an official form of communication in EE 461L. You are responsible for any announcements or other information posted on Piazza. Use Piazza for any question of general interest, and email for any question that pertains only to you. Do not expect to get last-minute help on assignments from the instructor or TAs. Do not expect detailed answers to technical questions via email. Students are encouraged to discuss important matters in person, typically during office hours. If you must send an email, ensure that it is both brief and clear. Put "EE461L" in the subject of your email. Sign the email with your full name. While it's easy for you to dash off an email question, it takes time to answer it. The instructor and TAs will not respond to email questions to which you can find the answer somewhere else (e.g., lecture notes, the class web page,piazza.)

Academic Honesty

Plagiarism and other forms of cheating will not be tolerated. This includes using code downloaded from the internet without explicit permission from the instructor or without adhering to the code's distribution license. Please be sure to adhere to the stated collaboration policy for each assignment.

If you do a team or pair assignment, you are not permitted to have one person (or a proper subset of the team members) do the assignment. Each team member must contribute to the assignment solution in a meaningful way, and work on each part of the assignment.

Another form of academic dishonesty is misrepresenting the contributions of teammates on project peer evaluations. You are expected to honestly evaluate the work of each team member at the end of project phases. These peer evaluations may impact grades. Do not, for example, help a friend out by giving them a higher score than they deserve. Your project peer evaluations are assignments that are part of your project grade.

Every piece of work that you turn in with your name on it must be yours and yours alone (or that of you and your project team, if it's team work) unless explicitly allowed by the instructor. Specifically, unless otherwise authorized by the instructor.

Students may not acquire from any source (e.g., another student or an internet site) a partial or complete solution to a problem or project that has been assigned.

You are responsible for complying with this policy in two ways:

You must not turn in work that is not yours.

You must not enable someone else to turn in work that is not theirs. Do not share your work with anyone else. Protect your files. After you have finished a class, do not share your work or publish answers to assignments. This means that you do not post your solution code to any public website such as public repositories on GitHub.

The penalty for academic dishonesty will be a 0 on the portion of your grade allocated to the type of assignment you cheated on, as well as a referral of the case to the Dean of Students. Further penalties, including suspension or expulsion from the university may be imposed by that office.

This policy is not intended to discourage students from learning from each other, nor does it ignore the fact that most significant work in industry is done by teams of people working together. Because of our need to assign individual grades, we are forced to impose an otherwise artificial requirement for individual work.

You are encouraged to study for tests together, to discuss methods for solving the assignments, to help each other in using the software, and to discuss methods for debugging code. Essentially if you talk about an assignment with anyone else you are okay, but the moment you start looking at someone else's source code or showing someone else your source code you have crossed the line into cheating. You should not ask anyone to give you a copy of their code or give your code to another student. Similarly, you should not discuss detailed algorithmic strategies to such an extent that you and your collaborators end up turning in essentially the same code. Discuss high level approaches together but do the coding on your own.

Examples of cheating are many and include accessing another student's account, looking at someone else's solution code, copying or downloading someone else's solution code, referring to solutions from previous semesters, having another student walk you through the solution and how to code it, having another student perform significant debugging of your code, having another student write your code for you and/or allowing others to copy or access your solution code. This means you shall not look on the internet for code to solve your problems.

Examples of allowable collaboration include discussions and debate of general concepts and solution strategies.

- The code you can reuse in this class:
- You may use any code you develop with the instructor or TAs.
- You may use code (with attribution) from the class slides and the class coding examples.
- You shall not make use of code you find from other sources including the world wide web.

Materials from the web should only be used for educational purposes. Thus, you can read about JavaScript and look at examples of JavaScript, but you must not copy any code from the web or be looking at any of this code from the web when writing anything you turn in. For more information on Scholastic Honesty and the UT Honor code see the University Policy on Scholastic Dishonesty

Help when You’re Struggling, or Have a Crisis or Emergency:

Please, when something bad happens, or when you’re feeling overwhelmed, get help. Don’t endure it on your own. Even talking through the situation often helps. Here are some options:

- See me. Come by office hours, or book an appointment (just email me.)
- Visit the ECE advisors. They’re not just there to help you register; they can really help in many situations.

Tel: 512-471-1851
Office: EER 2.884 Hours: Monday–Friday 8 a.m–5 p.m. (closed for lunch 12-1 pm)

- Talk to Student Emergency Services. They are there to help you with all kinds of life's troubles (family, housing, health, money, stress, etc.) whether it's a crisis or not.
  Tel: 512-471-5017 E-mail: studentemergency@austin.utexas.edu
  Office: SSB 4.104 Hours: Monday–Friday 8 a.m–5 p.m.

Religious Holidays: By UT Austin policy, you must notify me via email of your pending absence at least fourteen days prior to the date of observance of a religious holy day. If you must miss a class, an examination, a work assignment, or a project in order to observe a religious holy day, you will be given an opportunity to complete the missed work within a reasonable time after the absence.

Students with Disabilities: Students with disabilities may request appropriate academic accommodation from the Division of Diversity and Community Engagement, Services for Students with Disabilities, 471-6259, www.utexas.edu/diversity/ddce/ssd/. Provide me with your accommodation letter during the first week of classes.