Last updated May 25, 2023

Welcome to 18652 - Foundations of Software Engineering (FSE)

18-652 is a central (and mandatory) course in the MS-SE program. It is a prerequisite or a corequisite for the courses in the 18-65X series as well as 18664 and 18668. All incoming MS-SE students must take it during their first semester. Below is the information to prepare you for the course and facilitate ramp-up.

Before you Attend the First Class

- ❏ Learn JavaScript and Node.js.
- ❏ Complete the Pre-Registration Programming Assignment and video.
- ❏ Learn Git and obtain a Github account.
- ❏ Create a CMU Zoom account.

You may need up to six weeks, depending on your background and skill level, to comfortably complete these ramp-up tasks. ***We strongly advise you to start early and not wait until the last moment. We will not accept a late start as an excuse for failure to complete the tasks on time. The deadline is strict!***

Textbooks and Readings

There are no mandatory textbooks. Necessary readings will be provided throughout the course.

Programming Skills

We will assume that you have significant experience with one or more modern programming languages. You will need to know how to build a web application end-to-end and be familiar with programming in **JavaScript** and **Node.js** to complete the team project component. ***To prepare you for the course, you will build a simple web application from scratch, using a specific development stack before you take the course. Refer to the Pre-Registration Programming Assignment below. Important: If you cannot pass this assignment, you will not be admitted to the course. If already enrolled in the course, you will have the option to drop the course and take it the following semester; but if you do so, your other courses that require 18652 as a prerequisite or corequisite will be affected.***

Git & Github

You will use Git & GitHub for version control and issue tracking. ***Obtain a GitHub account, add your full name and clear headshot photo (above shoulder showing your face clearly and in a recognizable manner) to your Github profile (this is important). Complete a Git tutorial, review the official Git reference, and experiment with Git.***

- Git reference: https://git-scm.com/doc
- Here is a free tutorial on Git: https://www.pluralsight.com/courses/how-git-works
- If you need more or something different, PluralSight also has a free course on Git: https://www.pluralsight.com/courses/code-school-git-real

- A compact, no-frills starter guide to git is here: http://rogerdudler.github.io/git-guide/
- Github Desktop client also has an interactive tutorial for beginners that some students find very helpful. It's available upon installation of the client.

18652 Pre-Registration Programming Assignment

The purpose of the assignment is to provide you with a context to demonstrate proficiency in a modern programming language, and to make sure that you are ready to tackle the course project. You will not succeed in the course project if you are not ready. While programming is not the main focus of the course, a lot of coding is involved. You will use specific technologies and require proficiency in them. ***This assignment will be graded. It is due at the end of the first week of classes. This deadline is strict -- please start early to make sure you can meet it!***

Instructions

- Build a **web application** that implements the requirements described below. You must show that you can engineer a web application essentially from scratch rather than hacking something together from scraps of code borrowed from various internet sources.
- You will use the following development stack to complete this assignment:
    - Client side: the standard "web stack": **HTML, CSS, JavaScript (JS).**
    - Server side: **Node.js** with **express.js** web development framework.
    - Database: any database, SQL or No-SQL, that runs on Linux/MacOS.
    - You may additionally use other basic/common JavaScript libraries, node modules, express.js middleware, or standard plug-ins. Typical example is JQuery.
  - The use of **express.js** is mandatory. You may **not** substitute it with another web development framework.
  - You may **not** use a front-end framework like React, Vue, or Angular. You may **not** use any UI layout libraries or frameworks (e.g., Bootstrap, SemanticUI, Material). The client-side must be done in pure CSS, HTML, and JavaScript (exceptions: you may use JQuery, other similar DOM manipulation libraries, a templating engine like *pug* or *ejs* for server-side rendering, or other common JavaScript libraries that do not provide packaged UI layouts or UI components).
- You must acquire **at least beginner/novice-level knowledge** in Javascript, HTML, CSS (the Web Stack) and Node.js and express.js, and attest to this in a team formation survey we will deploy at the beginning of the course.
- **Push your code to a GitHub repository, become comfortable using GitHub and git on a day-to-day basis.**
- Create a **short demo video** (max 5 mins) for your application. Describe your application's **purpose, design**, **code structure**, choice of **technologies**, and demo your **application**. The video must have **audio overlay (audio must be understandable)**, not only a screen recording. We need to hear you explain your application and code. All the **highlighted** components above must be present in the video.

- The user interface must be **web-based**: users will use the application through a **browser**. Non-web-based (command-line or native GUI) will not be accepted.
- Organize your code neatly. Use meaningful variable and module/file names, and good coding style.
- **You should be prepared to show and explain your code to a teaching assistant**. If you are not able to explain your code and demonstrate sufficient understanding of the technologies used, we may determine that your deliverable is unoriginal and copied from another source. You may also be subjected to an Academic Integrity Violation investigation.
- **Your code must be original. READ THIS SECTION CAREFULLY!** You may model your code's organization on the structure of an online example, but you may not copy code verbatim from an online example and superficially change the content. You may use AI-generated code only for small portions and only if you fully understand it and adapt it fully to your context. Your resulting code should be unique and significantly different from any online resource, from other students' code, and from previous students' submissions. **If your code is determined to be significantly similar to that of another student or of an online source or mostly AI-generated, you will fail the assignment, and your actions will be subject to university's applicable academic integrity policies. *A plagiarism tool may be used to analyze the originality of your code by checking it against known online examples and previous submissions.*** If you based your application's structure on an online example in any way, **you must cite the URL of the example in your video and explain how you reused that example. If you fail to cite the online source, and we determine that your code's structure is based on an online example, you will fail the assignment, and your actions may be subject to applicable university and course academic integrity policies.**

Application Requirements

Implement a simple FSE Chat Room application. The system should allow a user to perform the following functions:

- Register for the chat room with a new username (one that does not already exist) and password
- Enter the chat room (login) with his/her username and password
- See other users' chat messages
- Post a chat message
- Leave the chat room gracefully (logout) without simply closing the browser tab

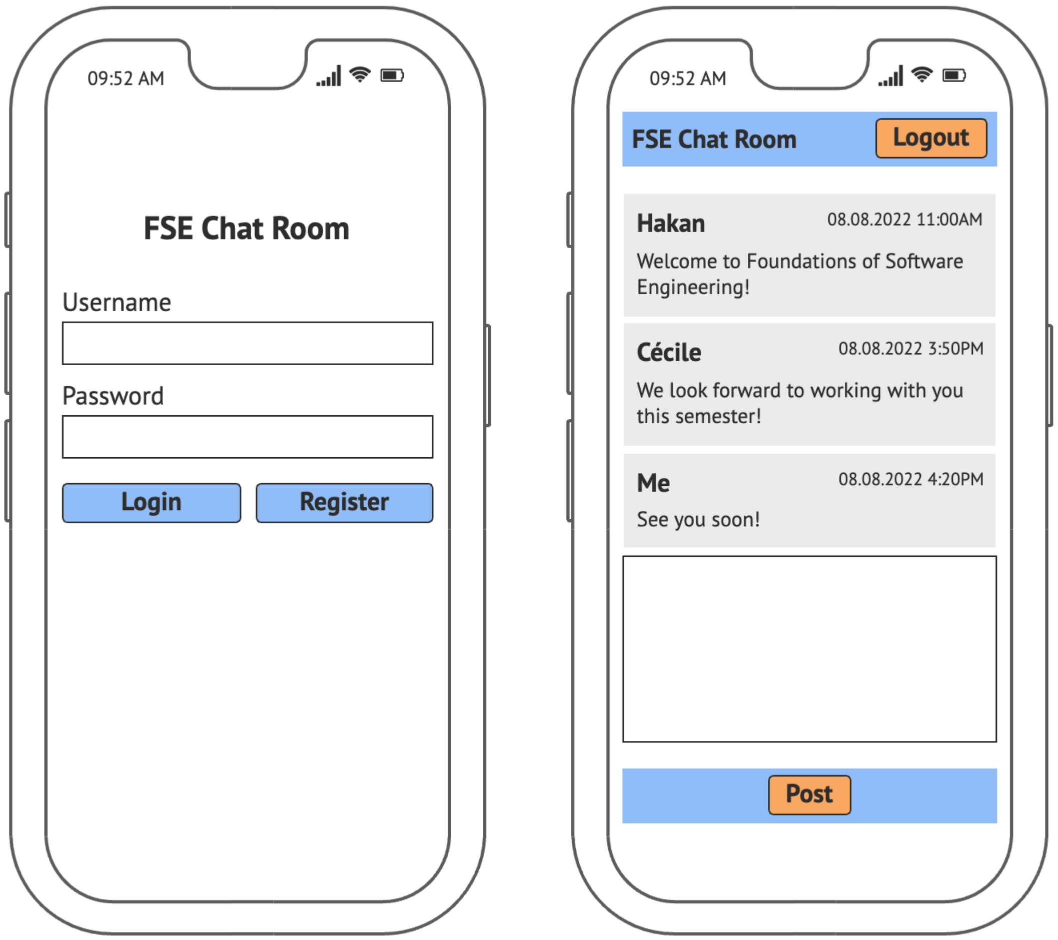
Do NOT add extra features that you think may be "cool" or "fun."

The application should satisfy these rules:

- Registration is persistent: once a user is registered, the user's info must be stored on the server in a database, and the user must be able to login again and be recognized.
- When a user posts a chat message, the text is displayed together with the user's username and a timestamp.

- When there is a new post, the screens of all of the users currently in the chat room are **dynamically updated** (updates are real time on all clients). You should demo this aspect using two different browsers (e.g., Firefox and Chrome) or using a combination of the normal and private/incognito mode of a single browser with one user accessing the system from the first tab and another user accessing it from a second tab.
- Chat messages are persistent. All posted chat messages must be stored on the server in a database and re-loaded when a user exits (logs out) and re-enters (logs into) the chat room.

A mockup of the chat room's main screens is shown below. Your application's screens should very closely resemble the mockup, with the same UI elements, layout, and look-and-feel. It must work on Chrome. The UI design should be compatible with a mobile view and work both on a phone's Chrome app or on Chrome running on a laptop/desktop while emulating a phone screen. If using Chrome on a laptop, desktop, or tablet, use the responsive design or mobile mode and choose a familiar phone screen format to demo your application.



Hakan 08.08.2022 11:00AM
Welcome to Foundations of Software Engineering!

Cécile 08.08.2022 3:50PM
We look forward to working with you this semester!

Me 08.08.2022 4:20PM
See you soon!

Checklist

Here is a checklist for your Pre-Registration Programming Assignment:

- ❏ *Application satisfies its functional requirements.

- ❏ *Application's screens very closely resemble the provided mockup above, with the same UI elements, layout, and look-and-feel. This is important to demonstrate your basic front-end skills.
- ❏ *Application uses real-time dynamic updates to display new chat messages instantly on other users' browser screens.
- ❏ *Application uses a database and persists the data.
- ❏ *Your code is original, you haven't copied the code from another source, student or previous student: it's your own! (Be extremely careful with this checklist item not be subjected to an Academic Integrity Violation!)
- ❏ *A 5-min video is created and uploaded to youtube.
- ❏ Application's purpose and requirements are briefly presented at the beginning of the video.
- ❏ Application high-level design is described in the video.
- ❏ The code structure is reviewed in the video.
- ❏ Technologies used together with their purposes are explained in the video.
- ❏ Application functionality is demonstrated in the video (again, using a phone screen).
- ❏ Your code is pushed to a **private** repo on github (free private repos are possible with github's Student Developer Pack - see https://education.github.com/pack)
- ❏ If applicable, you cited the URL of the online example on which your application's structure is based (this is critical since you may fail the assignment otherwise).

You have completed the assignment only if you can tick off all items on the list above. Not satisfying the checklist items marked with an asterisk (*) will result in automatic failure of the assignment.

You do not need to submit your code unless we ask. We will ask you to add us to your github repo when we need to review your code.

Demonstration and Code Review

We may ask you to demonstrate your running application in person and discuss your design and code at the beginning of the semester. We may review your code and ask questions about it: you should be prepared to explain all elements of your code. If you cannot answer the teaching team's questions during a code review in a satisfactory manner that convinces us that you have built the application yourself and you thoroughly understand the codebase, you will fail the assignment. If we determine that you haven't built the application yourself, we will initiate an Academic Integrity Violation case!)

Additional Resources

In addition to **express.js** (a lightweight web application framework for **Node.js**), you may find the following middleware and other JavaScript libraries useful:

- jQuery (JavaScript library for HTML/json manipulation and traversal)

- Pug or EJS or another Node.js templating engine for server-side rendering of HTML
- underscore (JavaScript library for convenient functional programming constructs)
- socket.io (JavaScript-based middleware for real-time client-server communication for dynamic web-based applications)
- Authentication and session/cookie-management middleware, such as passport.js.

Other alternatives are available too, but the above are good standard choices. Please email hakane@andrew.cmu.edu if you are unsure whether you can use a library not listed above.

If you have never used JavaScript, starter guides to plain JavaScript are below (skip if you are already familiar with JavaScript):

- JavaScript Tutorial (with a cloud IDE to instantly try out examples, make sure to review JavaScript Objects and JavaScript Classes sections)
- A re-introduction to JavaScript (JavaScript tutorial from MDN)
- JavaScript resources (various resources and guides from MDN)

PluralSight has decent, reasonably priced online courses on JavaScript, Node.js, express.js, and HTML5/CSS.You may want to get a one-month subscription to access everything, and then move onto free online sources. See the list here: https://www.pluralsight.com/product/paths. We suggest these paths, depending on your knowledge: *JavaScript Core Language*, *Working with Node.js*, and *Building Websites with HTML, CSS, and Javascript*.

For express.js, below are some resources. Start with the guide available at the first link. The second site gives a brief overview, and links to expressjs.com for details. The third is for Bootstrap, a library for responsive user interfaces and for a clean, professional-looking layout (you may use it during the course project later, but not for this pre-registration assignment).

- http://expressjs.com/
- http://webapplog.com/express-js-fundamentals/
- https://www.pluralsight.com/courses/responsive-websites-bootstrap3 (paid tutorial: if you wish to use a responsive design framework with express.js later in the course)

There are plentiful other online resources and tutorials that will help you get expertise in Node.js and JavaScript. Most tutorials illustrate these technologies by showing how to develop a simple, real-time chat application, like the one you are supposed to create for this assignment. W3Schools.com also has a series of simple JavaScript and client-side web-based programming tutorials (HTML, CSS).

Using Object Orientation (OO) with JavaScript

This will be extremely important during the course to give structure to your JavaScript code. **In particular, your server-side code written in Node.js must be well structured**. A caveat of JavaScript is that it allows many kinds of unsafe, unstructured coding practices that lead to “spaghetti code.” JavaScript is a multi-paradigm language that also supports OO, albeit with some effort. Keep your JavaScript code structured by using objects properly and adhering to

OO principles. You must use recent versions of Nodel.js and JavaScript (based on ES6, aka ES2015, or later). These versions support OO better. You will be required to use these versions in your course project. Here are some resources on ES6 (ES2015):

- https://babeljs.io/learn-es2015/ (free overview)
- https://scotch.io/tutorials/better-javascript-with-es6-pt-ii-a-deep-dive-into-classes (a good tutorial explaining how new syntax makes prototype inheritance in JavaScript almost look like inheritance in Java)
- https://www.pluralsight.com/courses/javascript-fundamentals-es6 (paid tutorial)
- https://developer.mozilla.org/en-US/docs/Web/JavaScript (free MDN guide and references, covers ES6)

If you have any questions, please email the course instructors.

Good luck!