

**Software Engineering II**

**Baylor University**

**Fall 2022**

Table of Contents

1. Objective 2

2. Expectations 3

2.1 Students 3

2.2 Baylor 3

2.3 Credera 3

3. Project Details 4

3.1 Project Assessment 4

3.1.1 Problem Statement 4

3.1.2 Milestones 4

3.1.3 Schedule 5

3.2 Office Hours 6

3.3 Group Member Roles 6

3.4 Product Backlog 7

4. Technology Frameworks 8

4.1 Technology Stack 8

4.1.1 Spring 8

4.1.2 React. 8

4.1.3 REST 8

4.1.4 Google Cloud Platform 8

4.1.5 MySQL…………………………………………………………………………………….8

5. Credera Mentors 9

# 1. Objective

Credera seeks to interact with students and faculty to develop deeper relationships through a semester-long project in Software Engineering II. Students will cooperate in teams, with the professor and Credera mentors.

# 2. Expectations

## 2.1 Students

Students should expect to face a real-world situation where they will be given a problem and are tasked with developing a solution. They will need to understand how the problem is approached, propose a solution, and implement it. They will work in teams, just as commonly occurs on Credera projects, to solve the problem and present their findings, emulating actual work practices on Credera projects. This will not be an easy task. Students will be expected to research and find solutions to their team’s issues. Mentors are there to assist, but not to solve challenges. Students will be expected to come to their office hour meetings prepared with specific questions and deliverables complete.

## 2.2 Baylor

Baylor will partner with Credera to manage the project to better prepare their students for jobs following graduation. Professors will work as a point of contact for Credera and the students on all project questions. Professor will work with Credera to ensure project meets all class expectations.

## 2.3 Credera

Credera will need four employees to participate in on-campus activities and regular phone calls with the students. Through weekly calls with the students, Credera employees will be expected to mentor in best practices and problem-solving solutions. Credera will be expected to work with professors to meet class expectations. Deliverables will be given to the students and mentors are expected to review and offer feedback to students.

# 3. Project Details

## 3.1 Project Assessment

### 3.1.1 Problem Statement

A new company, TrailBlazers, Inc., wants to create a custom experience for road trippers everywhere, so they can plan their route for optimized efficiency and fun. You’ve been hired as a consultant to build out a web application for this project that plans stops, playlist recommendations, route options, and more. The client wants your team to create a user-friendly web app for users to register under the appropriate classification, input their start and end location, and choose their desired stops. In addition, your team can enhance the application with other features at your discretion.

### 

### 3.1.2 Milestones

This semester will be broken out into four milestones for technical implementation. Each team will be responsible for completing each milestone by the assigned week (see project schedule for dates). At the end of each milestone, each team will present their current implementation to the mentors and professor during scrum. A formal presentation is not needed during the first three milestone checkpoints, but a thorough walk-through of current complete functionality will be expected.

1. **Milestone 1:**

* Have team repository setup
* Have pages scaffolded out with routing (can be empty, home page just says “Home”, etc…)
* Call example Java endpoint from client (ask the mentors for more details on this in our meetings)
  + - * 1. **Demo:**
* Demonstrate that all the pages exist by directly hitting the URLs
* Display the result of the test endpoint service call

1. **Milestone 2:**

* Have all the pages implemented – sign-in/registration, profile page, home page (at a minimum)
* In the service layer, be hitting a MySQL Database and be able to create/edit account, and add/edit trips, create/edit stops, set dates, review stops, etc.
* Have deployed the template project to GCP
* Have continuous integration setup (a Git commit triggers a build, which triggers a GCP deployment)
* In the front-end, the individual should be able to
  + Register and log in
  + Add and edit trips
  + Specify preferences for their stops, dates, etc.
  + Rate the trip and the stops suggested
* Add unit tests in Java layer and add a testing step to the continuous integration deployment pipeline (maintain and update tests moving forward)
  + - 1. **Demo:**
* Show the app running live on GCP
* Change the message on the home page and commit the change, then check if the continuous integration is working and the message changed on the live site
* Demonstrate creating a new account and logging in
* Demonstrate adding a new trip and providing details (date, start and stop destinations)
* Demonstrate deployment, with all tests passing

1. **Milestone 3:**

* In the service layer, be able to suggest stops, schedule and cancel trips, send notifications, and submit ratings for stops
* Recommendations: A critical component of this project is a sophisticated system for matching stops to a trip. Typically, teams achieve this by allowing the individual to create a trip, assign start and end destinations, and select preferences for stops. Then, the system can recommend a list and the individual can choose their stops. You can take a different approach from the one above and even add components like a road trip playlist, but it needs to be able to create a road trip effectively and also support a large user base.
* In the front-end, individuals should be able to:
  + Review recommended stops
  + Apply recommendations to a trip
  + Get trip notifications
  + Edit or cancel a trip
  + Rate the stops chosen after the trip
  + Update existing and add new unit tests in Java layer

**Demo:**

* Suggestions for stops related to preferences and journey details
* Reminders before and during trip
* Rating the stops during and after the trip

1. **Milestone 4 (due at time of Final Presentation):**

* Enforce security (viewing other user’s trips, only editing user-owned trips, unauthenticated user viewing the site, etc.)
* Final UI/UX design
* Bonus features

**Demo:**

* Demonstrate security
* Demonstrate bonus features
* The UI should be complete
* Product and functionality should be in a complete state

1. **Bonus Feature Examples:**

* Socializing/gamifying the app
* Tutorial
* OAuth sign in through a social media site like Google/Facebook/Twitter
  + There is an approval process to set up some OAuth integrations, if you choose to do this be sure to start early
* Social media integrations
* Mobile notifications
* Playlist generator
* Password reset
* Mobile application
* Promotions
* Use of the Alexa service

### 

### 3.1.3 Schedule

The following is the schedule for this semester along with the due dates of milestones and weekly deliverables.

|  |  |  |
| --- | --- | --- |
| **Date** | **Week** | **Credera Assignment** |
| 8/30 | 1 | **Project Kick Off!** (3:30 – 4:45pm)   * Class introduction * Project kick-off presentation Role assignments and requirements clarifications |
| 9/5 | 2 | * First scrum meeting * Continued requirements clarification for Milestone 1 * Milestone 1 development |
| 9/12 | 3 | * Milestone 1 development |
| 9/19 | 4 | **Milestone 1 Due**   * Continued requirements clarification for Milestone 2 * Begin Milestone 2 development |
| 9/26 | 5 | * Milestone 2 development |
| 10/3 | 6 | * Milestone 2 development |
| 10/10 | 7 | **Milestone 2 Due**   * Continued requirements clarification for Milestone 3 * Begin Milestone 3 development |
| 10/17 | 8 | * Milestone 3 development |
| 10/24 | 9 | * Milestone 3 development |
| 10/31 | 10 | * Milestone 3 development |
| 11/7 | 11 | **Milestone 3 Due**   * Continued requirements clarification for Milestone 4 * Begin Milestone 4 development |
| 11/14 | 12 | * Milestone 4 development |
| 11/21 | 13 | * Thanksgiving break – no scrum |
| 11/28 | 14 | * Milestone 4 development |
| 12/6 | 15 | **Final Presentation (3:30-4:45pm)** |

## 

## 3.2 Office Hours

Office hours will occur weekly. Teams will have a 20-minute slot on a day determined by the students in the evening. At least two mentors will be present for every scrum, but an individual mentor will not be assigned to a specific team; rather all mentors will be available for all teams. Every team member is responsible for stating what they accomplished last week, what they plan to accomplish this week, and any roadblocks. The remainder of the time will be used on some of the following:

1. Each team should have questions ready (i.e., requirements, Spring, APIs) to discuss during the meeting for further understanding
   1. Feel free to send the mentors questions ahead of time so that we can better utilize scrum time
   2. Any code needed for questions should be available in source control so that the mentors can look at it directly and run it locally
2. Review weekly deliverable or discuss the upcoming deliverable
3. Provide a brief demo of new functionality

Questions submitted asynchronously will be answered in a timely manner, but please note project mentors will most likely be confined to business hours, meaning questions may not be answered until 24 hours later. Please plan questions outside of office hours accordingly.

## 3.3 Group Member Roles

The minimum prescribed roles in each group are as follows:

1. **Project Manager** will be responsible for coordinating the activities of the group and for liaison with the professor/coach to resolve issues affecting the group from time to time. The project manager should be: professional and responsible, a good organizer, and an effective communicator. His/her functions include: 1) Coordinating group activities, 2) Overseeing and managing activities, 3) Allocating roles, activities and responsibilities, Coordinating and managing documentation activities (You may refer to [What a team leader does/does not](http://cs.baylor.edu/~song/courses/3372-13f/project/3372-TeamLeaders.htm) as well.)
2. **Requirements Engineer** leads the requirements effort
3. **Design Engineer** leads the design effort
4. **Quality Assurance Engineer** oversees test case design, validation of requirements, design, etc.
5. **Project Librarian** keeps all meeting logs and makes all design artifacts available for the team

The initial role assignment will be decided within the group and can change during the semester, if needed, after consulting with the professor/mentors. Each student is also required to summarize the activities they carried out and give the times spent on each. Regardless of group member role, each team member is expected to make contributions to the code-base.

## 3.4 Product Backlog

The product backlog includes a list of artifacts that must be completed this semester. The following list must be included in each team’s final deliverable.

|  |
| --- |
| **Product Backlog** |
| **Build activity diagrams** - Initial use cases outlining, in general, core user interactions with the new feature |
| **Build use case documents** - Brief listing of requirements generated from use case diagrams |
| **User interface mockups (if applicable)** - UI mocks based on the use cases to help showcase how the feature will be implemented |
| **Create requirements class diagrams** - Class diagrams be built off of the requirements & use case document. Students need to keep in mind that requirements class diagrams are meant to show how the domain is related, not how it is implemented, and the ER diagrams show the actual database tables; correlating them may take some work |
| **Develop architecture diagram** - An initial architecture diagram that outlines major elements of the feature and how they relate |
| **Build sequence diagrams** - Initial sequence diagrams showcasing major flows in design class diagram |
| **Document project plan** – The project plan outlines deliverables. The students should indicate how much of the feature they can complete and what parts of the feature will be accomplished by each deliverable in detail |
| **Create test plan** – The test plan details how features are tested in each milestone. Testing should include both functional and unit testing, and a milestone should be considered complete when the tests are satisfied |
| **Document design patterns used** – Documentation showcasing which patterns you have selected and how you are implementing |
| **Create User Manual –** Guide should detail operations and maintenance of your project |
| **Final project documentation** - Document organizing all prior deliverables |
| **Final project source code** |
| **Final presentation** - PowerPoint presentation or otherwise that demos your project, you should talk about the software development process and any risks/setbacks/difficulties encountered along the way |
|  |

# 4. Technology Frameworks

## 4.1 Technology Stack

### 4.1.1 Spring

Spring (aka Spring Framework) is an open source framework and inversion of control container for Java that was created to address the complexity of developing enterprise applications. Spring has layered architecture that allows developers to leverage certain components while not utilizing others that they may not care about. Spring’s biggest features include:

1. Transaction management – Spring allows for pluggable transaction managers to deal with transactions
2. Inversion of control container – Spring allows for dependency injection, which helps with the configuration and management of Java objects
3. Data access – Easy integration with Hibernate and JDBC
4. Model-View-Controller (MVC) – Spring provides a framework for extending and customizing web applications
5. Messaging – Spring is able to leverage existing technologies, such as Java Messaging Service (JMS) for sending messages

Some of these technologies will be vital to your project, and Spring allows for easy integration with other useful technologies and frameworks.

### 4.1.2 React

React is a JavaScript-based open-source web application framework for the application’s front-end that allows you to create dynamic views for your single-page application. It is a very widely-used platform and is currently one of the most popular projects on GitHub.

### 4.1.3 REST

Representational state transfer (REST) is an architecture approach (not a tool or language) that consists of a coordinated set of constraints that applies to components, connectors, and data elements. The REST architecture has been applied to developing web services and is commonly used as an alternative to SOAP. REST allows for data to be quickly obtained via point-to-point communications with clients. For this project, REST will be used to obtain data from the service layer.

### 4.1.4 Google Cloud Platform (GCP)

GCP is Google’s cloud platform. You will use GCP to host your application so that it will be publicly accessible on the web. You will be able to control how the application deploys, runs, and connects with your database and any other resources.

### 4.1.5 MySQL

MySQL is a popular open-source relational database management system. It is reliable and easy to use and provides official Docker images maintained by the MySQL team for Linux platforms. MySQL is a leading choice for SQL databases and is used in a wide variety of applications.

# 5. Credera Mentors



