

Project Report Milestone 3

Tues-18.30-3

Oscar Ling, Indirakshi Sreeram, Yuepeng Du, Andhika Rakha  
 | Software Engineering Process and Tools | 16th October 2020

Table of Contents

[Project Vision 2](#_Toc53952293)

[System Architecture/Design 3](#_Toc53952294)

[Description of Scrum Process 4](#_Toc53952295)

[GitFlow organization 5](#_Toc53952296)

[Deployment Pipeline 6](#_Toc53952297)

[Evidence of test execution 7](#_Toc53952298)

# Project Vision

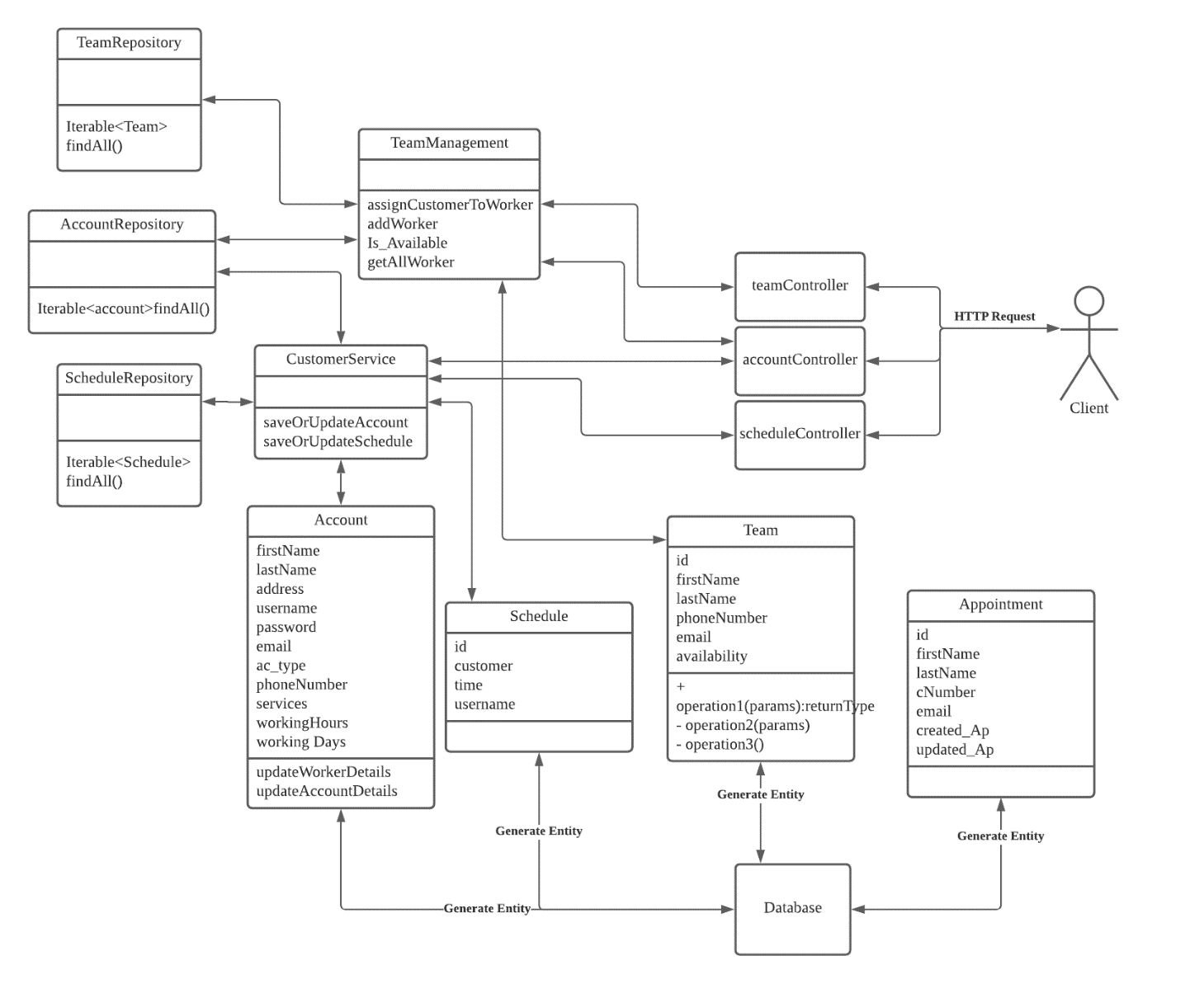
AGME is a company that requires a 24/7 web-based application system to provide various services and bookings for their clients and can be used by any service provider. This will allow clients to book appointments for many services plus an assigned service provider. The various business services provided include hairdressers, doctor appointments, gym consultation/appointments, real estate appointments so on and so forth

The business will be given two open API providers. One for the business management side which enables users to add/edit/remove types of services, employees and their schedules. Another API will be catered towards the customer’s needs such as booking a service. This API being implemented, enables the business to integrate it with their own individual website. AGME has reached out to us to deliver an application “Online Appointment Booking System” as our team has had experience and history with delivering high quality projects to various businesses, no matter the challenge.

They have requested a service that is accessible to all such as managing users which include both admin and customers who can perform different functionalities and work differently to one another. The logged in customer must also be able to have a look at their own dashboard with custom booking details, contact information and other functions

Our team has developed an application using technologies such as CSS, HTML, JavaScript, SpringBoot and NodeJS. The overall application is designed in a way that allows for a smooth sailing experience for all users young or old.

# System Architecture/Design



# Description of Scrum Process

Our group consisted of four members and the Scrum Master being Oscar Ling (s3604409)

The Scrum Master set up a meeting every week , on Saturday outside of the general Tuesday tutorials and during these meetings the group made major progress by defining objectives clearly for upcoming sessions, providing weekly updates to all team members ,conducting a scrum meeting (what was accomplished, what went wrong, how can you go about this task/ if any help was needed , what was left to do) and a backlog refinement meeting. The meetings were held twice a week (During/After the general Tuesday tutorials and on Saturday evenings). For each sprint, the group ensured to meet twice a week and to use Microsoft Teams to communicate frequently and discuss any bugs/ quick fixes throughout the weeks.

At the start of each sprint we conducted a Sprint Planning to understand the goal/ vision, what is expected of us in this sprint, what features we were going to implement (with their estimation points). At the end of each sprint, we conducted a sprint Retro where we all discussed how we refine and further improve our work for the next sprint

# GitFlow organization

In terms of the organization of our remote repository on GitHub, We committed under certain conditions, which are as follows:

* Every time we either finished a section that was delegated to us, this includes features or bug fixes
* Every time we were to save our progress
* Every time we needed to share around files that were needed by another colleague
* Commits in order to merge and push to dev or master

Therefore, as shown in our GitHub logs, there are numerous commits each detailing what their purpose of committing is.

We encourage each other to utilize the structure that required a push to dev before pushing master to minimize bug spreads and errors to occur in the master branch. From there we would each create our own branches off of dev or master and work within those branches until the code is implemented and repeat the process again of committing and merging to dev, before merging to master. Each branch that is created also should include the developer’s name that is working in the branch and should be clear so that others know the owner of a certain branch other than feature, dev and master.

We believe this to be the most effective way to locate the timing of the bugs and errors that occurred and allowed a rollback as a safety net so we do not corrupt all coding within the repository.

# Deployment Pipeline

A picture containing text

Description automatically generated

Although we did not fully complete our pipeline, the above diagram presents the intended pipeline in which we would deploy our application.

Using our IDE and continuous updates from the GitHub repository, we would implement, develop and fix the code for features and releases before pushing back to GitHub, in turn, CircleCI will be connected to the repository and automate the testing for every push. Which will then connect to Elastic Beanstalk within AWS to automate deployment of the release.

While on the other hand, Terraform will be used to continuously update our AWS based database server instance and any other AWS infrastructure, including web servers and Elastic Beanstalk.

# Evidence of test execution

