Topmate Clone - Final Report

Team Roles

Scrum Master - Dalton Avery
Product Owner - Sudhanva Rajesh
Developers: Ashutosh Punyani, Braeden Smith, Geetesh Challur, Zepu Ma

There were no changes in role throughout the project.

Links

Heroku - https://topmate-clone-779709a9559a.herokuapp.com/
GitHub - https://github.com/tamu-edu-students/TopMate-Clone/
GitHub Projects - https://github.com/orgs/tamu-edu-students/projects/10/
Slack - https://topmateclone.slack.com/
Presentation/Demo Video - https://youtu.be/JNPXHDWeIp8

Summary

This semester, our team has endeavored to develop a clone of Topmate (https://topmate.io/), a web service that facilitates service sales between service providers and clients, for Manik Taneja. At this stage, just as when we began, our only stakeholders are Manik, his clients, and any future users of the application. The aim of the project was to copy the core features of the website into a format that could be utilized by Manik for his services. These core features include posting services, setting availability, and booking appointments. We are happy to report that our team has successfully implemented all of these requested features!

Specifically, our application has implemented the following: Admins are able to describe the services they offer, including their durations and price. They are also able to define their availability periods for clients to book time. Lastly, they can manage and maintain their appointments, and customize their public page. Clients, on the other hand, are able to book appointments through the admins' public page and manage these appointments through an emailed link. All of these features have been thoroughly tested and are ready for use today.

User Stories

All of the user stories undertaken by our team were able to be completed before the deadline. Below, they are broken down by iteration and their outcomes are described.

Iteration 0

Create slack and invite teammates – 0 Story Points – Completed by Braeden

Created our slack team (https://topmateclone.slack.com/) and invited necessary members.

Reach out to client and set up meeting – 0 Story Points – Completed by Sudhanva

Reached out to our client Manik Taneja, and set up an initial meeting for introductions.

Initialize Rails Project – 0 Story Points – Completed by Dalton

Generated the rails application and set up basic configuration.

Initialize frontend with basic features – 0 Story Points – Completed by Zepu

Added basic features to the front-end to prepare for major features.

Deploy Rails Project to Heroku – 0 Story Points – Completed by Ashutosh

Deploy a Ruby on Rails project to Heroku for online accessibility.

Iteration 1

Admins can log in – 5 Story Points – Completed by Ashutosh and Sudhanva

Created a login functionality for the user, Upon successful login the user is redirected to the dashboard. In the case of incorrect credentials, the user is prompted to try logging in again.

User can reset password: Email – 3 Story Points – Completed by Dalton

Added interface to allow users to submit their email to reset their password.

User can reset password: Send Email – 3 Story Points – Completed by Zepu

Set up the password reset page to send an email when requesting a reset.

User can reset password: New Password – 3 Story Points – Completed by Geetesh

Set up the reset password page and ensure it successfully changes the password in the database.

User can access dashboard – 1 Story Point – Completed by Ashutosh and Sudhanva

Created a dashboard that should be presented to the user after a successful login. The user can navigate to other parts of the application from this dashboard.

Iteration 2

Admins can maintain public page – 3 Story Points – Completed by Dalton

Added wildcard routes that searched the database for usernames and set up a public profile page displaying the user's services.

Admins can add services via the dashboard – 3 Story Points – Completed by Sudhanva

Created a dashboard where the admin can view existing stories and their details, and added the functionality for the admin to add new services to be listed.

Admins can edit existing services via the dashboard – 3 Story Points – Completed by Geetesh

Created the form that allows users to edit services they have already created.

Add gmail stmp server for sending emails – 3 Story Points – Completed by Zepu

Set up an email server through Gmail to support password resets and eventually handle booking notifications.

Admin should be able to define time slots – 3 Story Points – Completed by Braeden

Created a page that allows admins to specify their availability times.

Admins can edit public page via the dashboard – 2 Story Points – Completed by Ashutosh

Client can edit his public page information inside the dashboard to update his information to the current.

Admins can navigate dashboard – 1 Story Points – Completed by Ashutosh

Client should be able to easily navigate through all web pages from the admin dashboard

Iteration 3

Users can view a service from the public page – 3 Story Points – Completed by Braeden

Prior to this story, services created by admins could not be viewed on their public page. Now, if a user publishes a new service, it will be visible from the public page to be interacted with by clients.

Admins can delete a Service from the dashboard – 3 Story Points – Completed by Sudhanva

Added functionality for the admin to hide a story from his dashboard. By doing so, the appointments related to this service can still be accessible, but new appointments cannot be created for that service.

UI changes – 3 Story Points – Completed by Geetesh

Implemented new styling to replace placeholder UI across the application.

Users can select a slot from admin's availability – 8 Story Points – Completed by Ashutosh and Dalton

Added all scheduling functionality, including removing the appointment times from the admins availability to provide the most accurate times available.

Tests – 3 Story Points – Completed by Zepu

Added additional RSpec tests to cover gaps in coverage for our controllers.

Iteration 4

Appointments page – 3 Story Points – Completed by Geetesh

Created a page within the dashboard to display all of the current and previously booked appointments.

Clients can edit their appointment – 8 Story Points – Completed by Dalton and Sudhanva

Added functionality for the user to edit any booked appointment. The user will be sent an email confirming their booking, and this email would contain a link to edit the appointment, this way only the user can edit the booked appointment.

Security fixes – 1 Story Point – Completed by Braeden

Our development processes left a lot of routes open to exploitation. After this story, the insecure routes were closed off and authentication/authorization checks were added where necessary.

Availability Updates – 1 Story Point – Completed by Braeden

After demonstrating our availability page to the Manik, he had some critiques on its layout and usability. This story resolved those concerns.

Services Usability Updates – 2 Story Points – Completed by Braeden

After demonstrating our services page to the Manik, he had some critiques on its layout and usability. This story resolved those concerns.

Scheduling updates – 3 Story Points – Completed by Ashutosh

When booking appointments, split the date and time to avoid slowing down the website. Users pick a date first, and then choose a time slot. If no slots are open, it shows 'No available timings for the selected date.'

Iteration 5

Appointment Scheduling: Mandatory Date and Time Selection – 1 Story Point – Completed by Ashutosh

Users can submit the appointment only when all the fields are filled, and errors will be thrown if they try to submit the form with missing information.

Rspec coverage improvement – 2 Story Points – Completed by Ashutosh

Improved RSpec coverage by writing tests, increasing code coverage from 38% to 98%.

User notification and validation updates – 3 Story Points – Completed by Braeden

At this stage in the project, we had two problems: First, our forms all had inconsistent ways of reporting success/failure. Second, they had little-to-no input validation. This story resolved these problems by making user notification consistent and adding validation to forms.

Scheduling Time Zone Bug Fixes – 3 Story Points – Completed by Braeden

There were some major issues with how appointments were updated in the database after making an update in the edit page. This story resolved them by cleaning up how the form input was displayed on the front-end and updated on the back-end.

fixing bugs while deleting hours – 3 Story Points – Completed by Sudhanva

Added validations and made sure that hours with scheduled appointments cannot be deleted.

Finalize Documentation – 1 Story Points – Completed by Dalton

Created the getting started documentation so future devs can quickly get the application up locally and deployed to heroku.

Seed Main User – 0.5 Story Points – Completed by Dalton

Finalize details with the client to manually insert admin user information to the database.

Bug: Fix Appt Filtering – 0.5 Story Points – Completed by Dalton

Corrected a bug where the filtering on appointments does not work.

UI update – 3 Story Points – Completed by Geetesh

Updated parts of the UI that did not implement the application's consistent style.

Adding buttons for connectivity between pages – 1 Story Points – Completed by Geetesh

Created buttons to improve navigation where it was lacking.

Iteration Summaries

Our project was completed over the course of five iterations, concluding every other Friday. Each iteration, our team was able to make progress toward the completion of our project. A summary of each of these iterations can be found below.

Iteration 0

Iteration 0 gave us time to initialize our application and teamwork environment. We set up our GitHub repository and project, as well as a Slack channel for communication. We also initialized our Rails project with basic features and deployed it Heroku. Finally, we spoke with Manik to establish a basic idea of the feature set and write some initial user stories.

Iteration 1

During iteration 1, the team worked to implement the authentication and authorization system. We created a user entity in the database and developed a way for the user to log in and reset their password. We also developed a primitive dashboard for the user to access after logging in.

Iteration 2

Iteration 2 focused on establishing the basic set of features that should be accessible to the admin. Part of our team worked on developing a dashboard page for viewing, adding, and editing services. Also in the dashboard, a member of our team set up a page for viewing, adding, and editing availability. A separate group worked on creating a public page and allowing the admin to change the data that is presented on it. Lastly, a member of our team set up the SMTP server to deliver automated email for our application.

Iteration 3

Iteration 3 split our efforts between application support and client functionality. To improve the stability of the application, we improved test coverage for both Cucumber and RSpec. We also updated the UI across the application to ensure a consistent and aesthetic experience for the user. We also added the beginnings of our booking functionality, allowing the client to view and select time slots for a service.

Iteration 4

In iteration 4, we primarily focused on expanding upon and improving the appointment feature. On the admin's side, we added a page to view appointments clients have booked. These can be filtered for easier viewing. On the client's side, we implemented a page to change the date and details of booked appointments. We also made some upgrades on the back-end and increased code coverage for this feature. In addition to this, Manik gave the team some feedback which we mostly resolved. The remainder will be resolved in iteration 5.

Iteration 5

During iteration 5, we made a number of miscellaneous changes to round out the edges of our application and prepare it for future support. We finished addressing the remainder of Manik's feedback and seeded his account into the database so that he could use it. We also fixed a few miscellaneous bugs and oversights that popped up throughout production but were never resolved. Besides that, we also improved navigation, updated inconsistent styles on certain pages, and improved testing coverage. Finally, we wrote detailed deployment documentation to assist future developers in expanding and maintaining the application.

Work Completed

The following table is a breakdown of the work done by each team member during each iteration. In general, team members contributed a roughly equal amount of effort each iteration, ensuring that our project velocity stayed on track.

Team Member	Iteration 1	Iteration 2	Iteration 3	Iteration 4	Iteration 5	Totals
Dalton Avery	3	3	4	4	2	16
Ashutosh Punyani	3	3	4	3	3	16
Sudhanva Rajesh	3	3	3	4	3	16
Zepu Ma	3	3	3	0	0	9
Braeden Smith	0	3	3	4	6	16
Geetesh Challur	3	3	3	3	4	16
Total	15	18	20	18	18	89

Meetings

In addition to in-class meetings, our team held meetings with Manik on a roughly weekly basis to discuss progress and new features. Summaries of each of these meetings with Manik can be found below.

18th September 2023

In this meeting, our client gave us a couple of features we could start implementing within the first iteration, and we decided on implementing a login functionality for the user, where the user's dashboard is presented upon a successful login, and the functionality to reset a user's password, should they forget or want to do so.

28th September 2023

In this meeting, we showed the client the implemented features that were requested in the previous meeting. We initially showed our client the login feature, displaying that a user hitting the domain will be redirected to the login page if a session did not exist for them. The user could then login which would then take them to the dashboard screen. The user also could log out, directing them back to the login page. We also displayed the functionality for submitting their email when they need to reset their password. From the login screen the user can select "forgot password" and submit their email, which will then be validated against the users table to ensure a user with that email exists before starting a reset password session.

1st October 2023

In this meeting, we spoke to our client and discussed features that could be implemented in the second iteration and agreed on completing the following requirements:

a. Implementing a Public page for the user. The user should be able to display his/her information, and details on the services offered by them.
b. Implementing functionality for the user to add services to his profile, The service should include details such as information about it, price and the duration of the service.
c. Implementing functionality for the user to view and edit listed services.
d. Implementing functionality for the user to edit his/her information
e. Allowing users to set their availability in terms of time slots for any given day of the week.

12th October 2023

In this meeting, we showed our client the implemented features in the previous iteration, which included the requirements that our client had asked for in the previous meeting. We also discussed features that could be implemented in the third iteration and agreed on completing the following requirements:

a. Allowing the details of a service to be viewed from the public page
b. Implementing booking a service from the public page of a user
c. Implementing functionality for returning the available time slots of a users availability after removing the time of booked appointments
d. Refactoring of UI components to provide a consistent visual model

26th October 2023

In this meeting, we showed our client the implemented features in the previous iteration, which included the requirements that our client had asked for in the previous meeting. We also discussed features that could be implemented in the fourth iteration and agreed on completing the following requirements:

a. Allowing a user who has booked an appointment with the client, to edit the appointment.
b. Fixing Issues that out client found with the application
c. Having a page where the client can view all the appointments booked, and their status
d. Fixing bugs in the UI for the appointments page, and make some logical changes

8th November 2023

In this meeting, we showed our client the implemented features in the previous iteration, which included the requirements that our client had asked for in the previous meeting. Overall our client was very satisfied with our progress so far.

19th November 2023

In this meeting, we discussed features implemented during the final iteration, and any changes/requirements that our client suggested these included:

a. Rewriting technical documentation for ease of future developers
b. Creating a user profile for client
c. Fixing the functionality to filter new appointments in the dashboard
d. Enhancing RSpec tests to increase test coverage from 38% to 95%.
e. Updating the appointment creation form logic to prevent submission if the form is not filled completely.
f. Improving user notification when taking page actions and input validation
g. Fixing time zones being improperly interpreted when making appointments
h. Fixing service listing in the public page to not show deleted services
i. Restricting deletion of availability if an appointment is scheduled
j. Updating styling for pages that have fallen out of date
k. Improving navigation between pages
l. Increasing cucumber test coverage
m. Improving the email notification for new appointments

Testing Process

We utilized the Cucumber gem as our behavior-driven development (BDD) tool and the RSpec gem as our test-driven development (TDD) tool. For each new feature or change, we wrote tests in both tools to validate our additions and then confirmed they passed successfully after completion. Finally, we would ensure that our code did not cause any other tests to fail, and address any failures if they appeared. This approach had the benefit of not only ensuring our approach did not overlook any edge cases, but also that it did not break any other features.

There was one major issue that we encountered with this approach, though it was mostly a result of our inexperience and improved over the course of the project. The problem was that it could be really easy to write the tests wrong, and then fail them in spite of proper implementation of the feature. We would then need to spend time fixing the test instead of the feature. This is overcome by simply writing the tests better, but it is a consideration that should certainly be taken when choosing a new testing tool.

Configuration Management Approach

For configuration management, our team used Git and GitHub. This helped us keep track of changes over time. To plan changes, we wrote user stories for each new feature and fix and added it to GitHub projects, assigning story point values and a developer. For each user story, the assigned developer would split a branch off of main, implement their change, and then submit a pull request back to main for review by another team member. This ensured the integrity of our main branch.

At any given time, we would have the main branch and about 4-5 feature or fix branches. Our project was completed over the course of five two-week sprints. At the end of each sprint, we confirmed that all branches were safely merged into main and made a major release of our website.

Our team did not need to complete any spikes throughout the development process. Any required research was completed by the developers as needed to complete their assigned stories.

Tools and Issues

Deployment of our project was a smooth process. We did not run into any issues using Heroku as our live hosting tool. Configuration management with GitHub and project management with GitHub Projects was also without issue. In general, the tools we used did not give us any major issues.

The three main gems we used were Cucumber, RSpec, and SimpleCov. Cucumber and RSpec were used for BDD and TDD, respectively, and have already been discussed. We utilized SimpleCov to identify our code coverage percentage and ensure we were covering all branches of logic. This turned out to be really useful in detecting blind spots in our testing.

Repository Contents and Deployment

TopMate Clone is structured as a basic Ruby on Rails project following the Model-View-Controller pattern. All of the files supporting the page-to-page functionality of the application can be found in the /app folder. Configuration files can be found in the /config folder. Specifically, the /config/database.yml file may need to be modified locally depending on your Postgres installation.

Files for testing can all be found in the /features and /spec folders, with the former containing Cucumber tests and the latter containing RSpec tests. Cucumber tests are structured in the [feature].feature files and described in the /features/step_definitions folder. The /spec folder mirrors the contents of the /app folder and tests can be found in the relevant sub-folder.

We do not use any special scripts to deploy our code. Heroku is set up to pull any new pushes to main from our repository and automatically deploy it. It will run with just the files within the repository.