Milestone 3

Links

Task board

https://app.clickup.com/6915733/v/b/s/6938372

Code repository

https://github.com/RMIT-SEPT/majorproject-mon-1530-7

CI pipelines

https://app.circleci.com/pipelines/github/RMIT-SEPT/majorproject-mon-1530-7/

Wireframes

https://www.figma.com/file/q9Tfd9aEHZHrOtHqKuW1AK/SEPT-Wireframes

Documentation

✏️ Meeting Notes

👦 User stories

✅ Acceptance tests

📊 Test Case Tracking

REST API

Backend models

📋 CI/CD evidence

Peer assessment

▼ **Amy Nguyen (s3783694)**

Contribution: 20%

- Enhance and refine website UI
- Created static page for 'Manage Employees'

- Updated wireframes
- Dynamically populated labels on admin upcoming bookings page
- Created static page for 'Edit Profile'
- Fix FrontEnd Code Warnings
- Wrote Vision Statement

▼ **Christopher Mellor (s3491308)**

Contribution: 20%

- Makefiles
- Code review
- Development assistance
- Prepared milestone submission
- Refactoring
- API methods to create and edit bookings
- API method to edit users
- Backend testing

▼ **Joel Whitley (s3787473)**

Contribution: 20%

- Login functionality separated based on user type
- Population of admin past booking page
- Toggle button for switching booking type on admin dashboard
- Page restrictions based on user type and logged in status
- Profile page loading

▼ **Thomas Testa (s3785277)**

Contribution: 20%

- Check and realign UI pages with wireframe designs
- Filling in missing backend unit tests
- Admin past bookings summary page frontend (static)
- Add more unit tests for increased coverage (frontend)

    - Dynamic population of upcoming bookings on admin dashboard
    - New Employee back end functionality (not released)

- ▼ **Yashwin Sudarshan (s3781718)**

  Contribution: 20%

    - Completed the following backend - frontend integration tasks:
        - Customer bookings page
        - Customer past booking page
        - Admin manage employees page
    - Completed backend API method to edit employee services
    - Configured CI/CD pipeline (partially completed):
        - Builds, tests, and deploys React Js application to AWS ECR
        - Builds, tests, and deploys Spring Boot application and Postgres database to AWS ECR
    - Multiple code reviews on Pull Requests
    - Maintained all meeting minutes

Changelog

User stories

-

Acceptance tests

-

Functionality

- New manage employees page
- Staff details dynamically populated on manage employees page
- Customer past and upcoming booking details are populated dynamically on their dashboard and past bookings page
- Customers can create a new booking
- Login functionality completed for administrator and customer accounts

- Enhanced website UI
- Past and upcoming booking details are populated dynamically on the admin dashboard, and the user can toggle to see either upcoming, or past bookings
- Backend API method for editing employee services by admin users

Files modified

`git diff --name-only 0.0.2 0.0.1`

Report

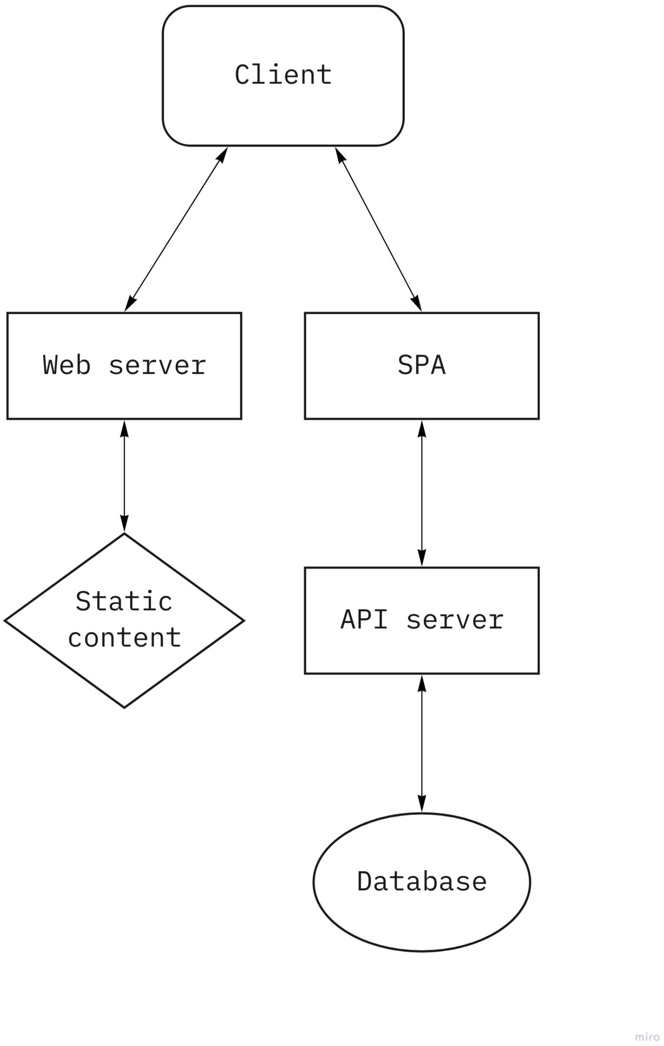
Vision statement

Our vision is to provide our clients with an online booking service that will improve their business's lifestyle. Our product is designed to maintain and enhance our client's management and organisation of bookings.

The product offers a variety of features catered across a range of users that will be interacting with our product including customers, employees, and administrators. For customers, features integrated consists of making a new booking, a dashboard with a summary of bookings, and past bookings. Additionally, employees are able to view their schedule and their assigned bookings through the dashboard. Administrators, on the other hand, are able to efficiently manage employees and their schedules, thus allowing smooth management of the overall service. Through the implementation of our simple and user-friendly interface, we aim to provide a booking system that is easily accessible for customers, employees, and administrators.

Our product is a valuable service as it allows clients to integrate an intuitive and efficient booking API in order to deliver 24/7 contact for its customers. With the incorporation of a 24/7 booking service, customers are able to make a booking at any time throughout the day which best suits their needs.

System architecture



Refactoring

Efforts were taken to move, and keep as much code out of, the Spring Boot controllers. We also refactored the list bookings API method in backend to filter more efficiently, so filtering code won't appear in the front end , as we wanted to keep database filtering as close as possible to the backend framework. Minor changes were also made in certain backend methods to enhance efficiency when handling lists, such as using sets instead of lists where appropriate. In the front end, we refactored components that contained duplicated code, into higher-order components which rendered the appropriate sub-components based on different views, which removed the presence of duplicated code.

Git flow

There were measures taken to enhance the organisation of our Gitflow.
These measures are listed as follows:

- Separate branches for managing code versions and new features
  - master, develop, release branches, feature branches, bug fix branches, hotfix branches
- Prefixes were used to specify the the type of branch
  - 'feature' - feature branches
  - 'bugfix' - bugfix branches
  - 'hotfix' - hotfix branches
  - 'release' - release branches
- The standard organisational merging process is as follows:
  - When a new branch is created, a pull request is made for it in draft mode until it is ready,
  - this pull request is laid out in the agreed format, where branch type and a checklist of requirements are specified
  - Once it is ready, reviews are requested from group members
  - When a reviewer is happy that the work is complete, they approve the PR
  - A squash merge is used to move the PR onto develop

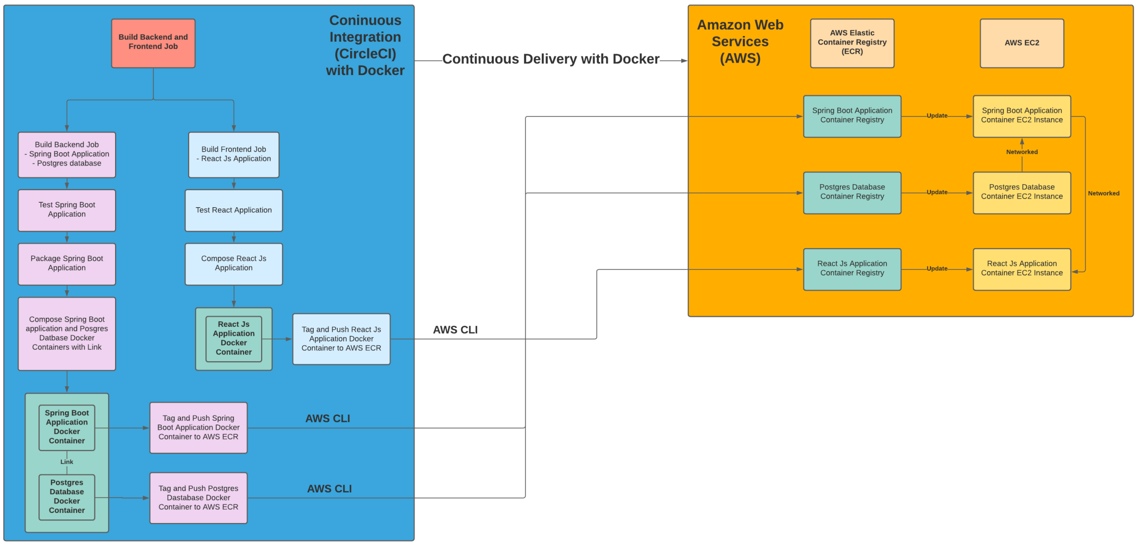
Scrum process

Scrum Master: @Yashwin Sudarshan

Regular meetings on Mondays and Fridays, plus midweek and weekend meetings as required.

Deployment pipeline

Visualisation of CI/CD Pipeline


Coninuous Integration (CircleCI) with Docker
Build Backend and Frontend Job
Build Backend Job
- Spring Boot Application
- Postgres database
Test Spring Boot Application
Package Spring Boot Application
Compose Spring Boot application and Posgres Datbase Docker Containers with Link
Spring Boot Application Docker Container
Link
Postgres Database Docker Container
Tag and Push Spring Boot Application Docker Container to AWS ECR
Tag and Push Postgres Dastabase Docker Container to AWS ECR
Build Frontend Job
- React Js Application
Test React Application
Compose React Js Application
React Js Application Docker Container
Tag and Push React Js Application Docker Container to AWS ECR
AWS CLI
AWS CLI
AWS CLI
Continuous Delivery with Docker
Amazon Web Services (AWS)
AWS Elastic Container Registry (ECR)
AWS EC2
Spring Boot Application Container Registry
Update
Spring Boot Application Container EC2 Instance
Networked
Postgres Database Container Registry
Update
Postgres Database Container EC2 Instance
Networked
React Js Application Container Registry
Update
React Js Application Container EC2 Instance

Acceptance tests

✅ Acceptance tests

📊 Test Case Tracking