PartyPlanner

by Emily Bradfield

INTRODUCTION

Who am I, and what is PartyPlanner?

- A website designed for assisting with planning parties and managing guest RSVPs
- Simple, effective, with database integration through an API
- Separate account types for host and their guest
  - Login redirects to appropriate page
  - Different functionality allocated to each

- Host has full control over guest list (create, read, update, and delete)
- Guests have control over account settings
- Responses sent from guests through website
- Host can amend and share party details as they see fit
- Each guest can only update their own account

.Objective

The overall objective of the project is the following:

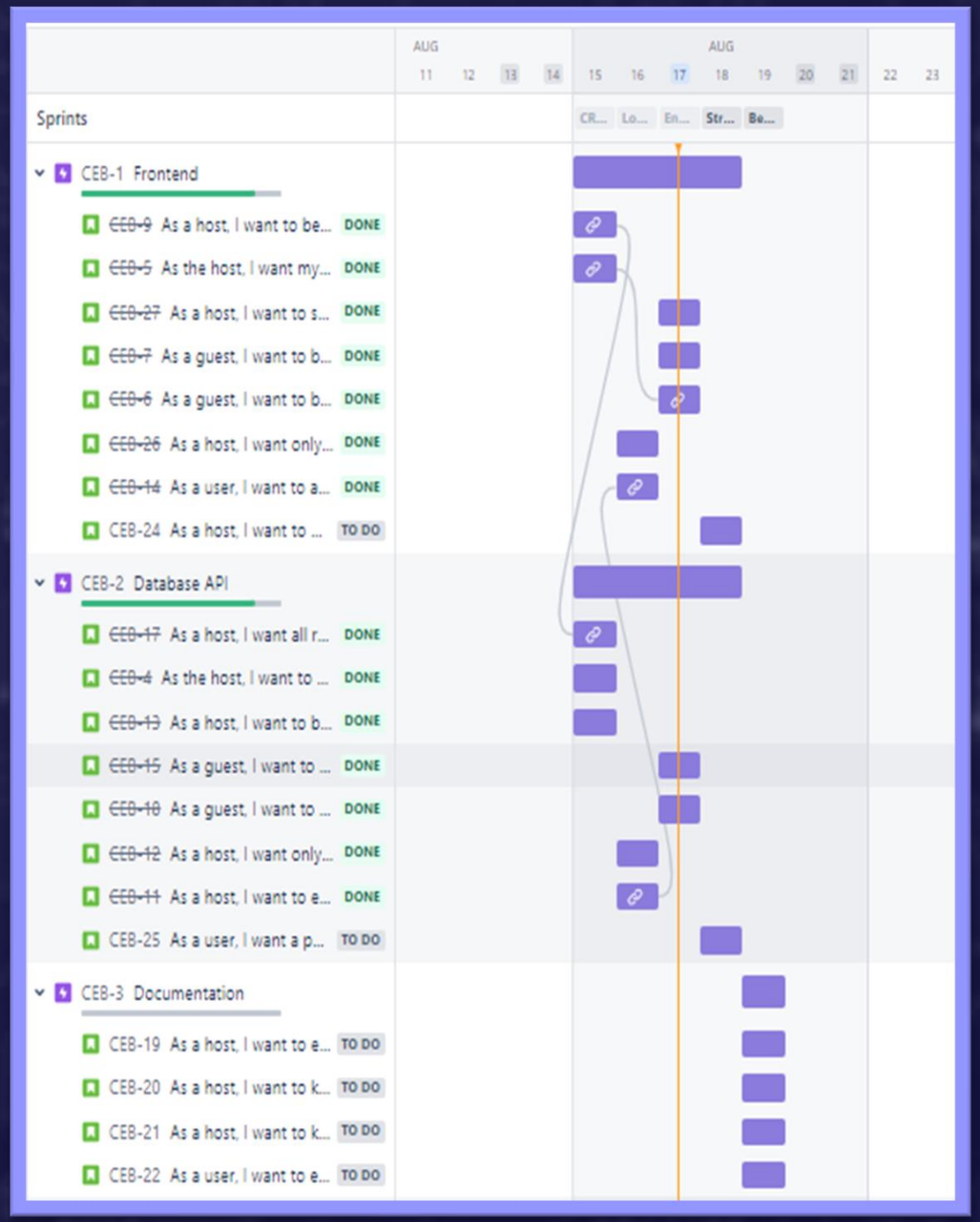
To create an OOP-based web application, with utilisation of supporting tools, methodologies, and technologies, that encapsulates all fundamental and practical modules covered during training.

Specifically, you are required to create a full-stack web application following the Enterprise Architecture Model, using:

- An application back-end developed using the Java (Spring Boot Framework).
- A test suite utilising JUnit and Mockito for integration and unit tests of the back-end.
- A managed MySQL database hosted locally.
- A front-end developed using Javascript, HTML and CSS.

If you wish to use any technologies which have not been covered as part of your training, you must consult your trainer first.

You must plan the approach you will take to complete this project using the design techniques you have learned.



CONCEPTUALISATION

How did I approach the task?

1. Assess and consider MVP -> CRUD functionality
2. Separate MVP from final concept and break down individual tasks
3. Consider priority of non-MVP tasks
4. Develop Jira Board (scrum) with 5 sprints (one per day)
5. Ensure sprint one produces MVP
6. Separate remainder of concept into features; allocate to sprints
7. Compare goals with current knowledge, resources, and timeline
8. Review priority and storypoints, note down specific tasks required of each user story

SPRINT ONE: CRUD (MVP)

Database API / CEB-4

As the host, I want to be able to invite, delete, and view all guests, so my guest list is accurate

To Do

Description

TASKS

1. Create database to store user/guest information
2. Develop CRUD function within the API to the database
3. Develop section in front-end to make all aforementioned changes
4. Use JavaScript to connect the frontend and database

Sprint: CRUD (MVP)

MoS-CoW Priority: Must

Story point estimate: 50

Development: Create branch, Create commit

Edit sprint: CRUD (MVP)

Sprint name *: CRUD (MVP)

Duration: custom

Start date: 8/15/2022 10:30 AM

End date: 8/15/2022 4:30 PM

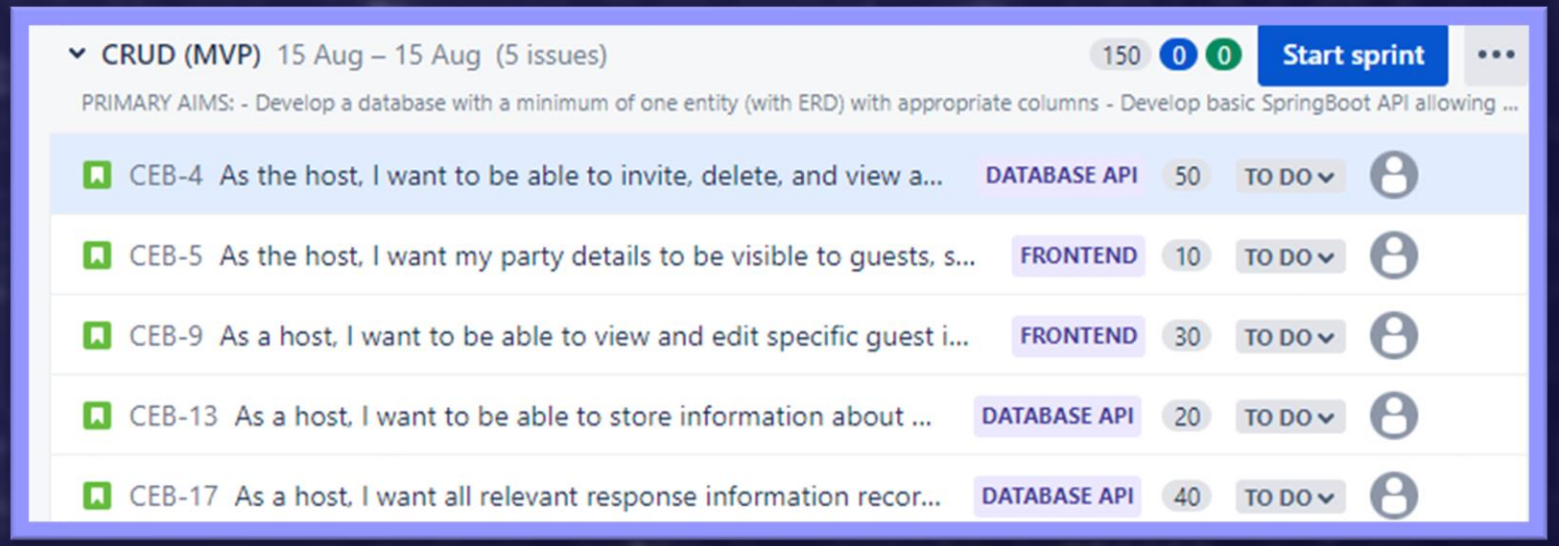
Sprint goal

PRIMARY AIMS:
- Develop a database with a minimum of one entity (with ERD) with appropriate columns
- Develop basic SpringBoot API allowing guests to be added, viewed, amended, or removed
- Create simple, single-page front-end enabling users to interact with and access the CRUD functions using JavaScript
- Ensure front-end includes some text with party details (can be static and not editable by users for time being)

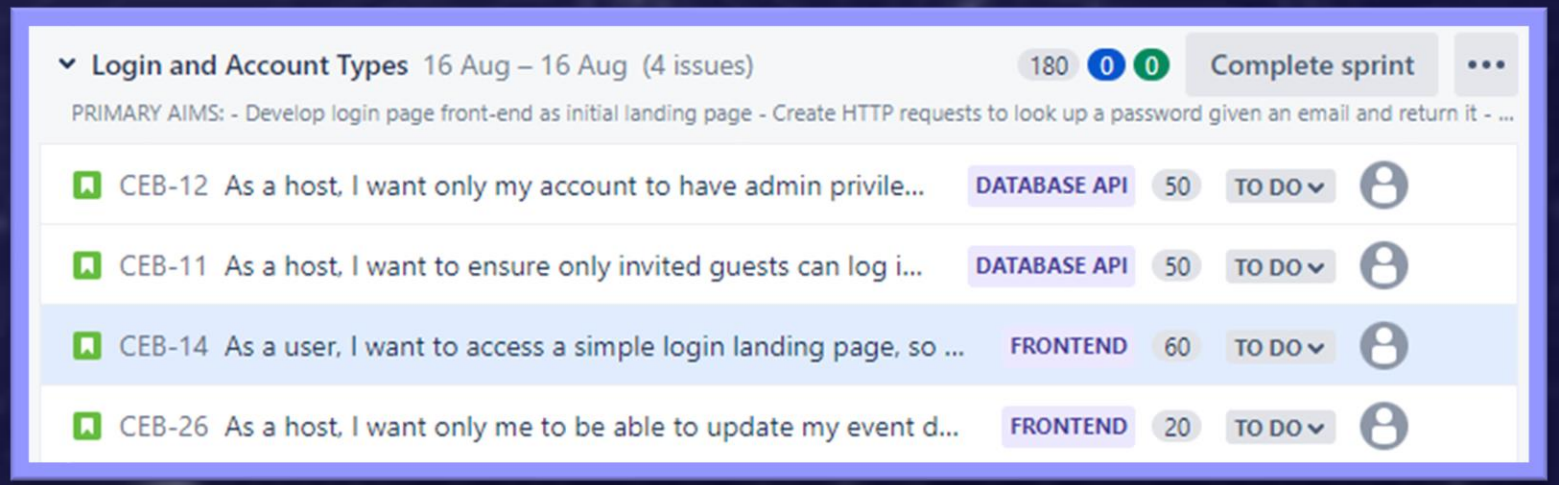
SECONDARY:
Enhance front-end to appear neater and require less reliance on guest ID to update, view, or delete; editing of party details text enabled

Update Cancel

ALL SPRINTS – MY ROADMAP

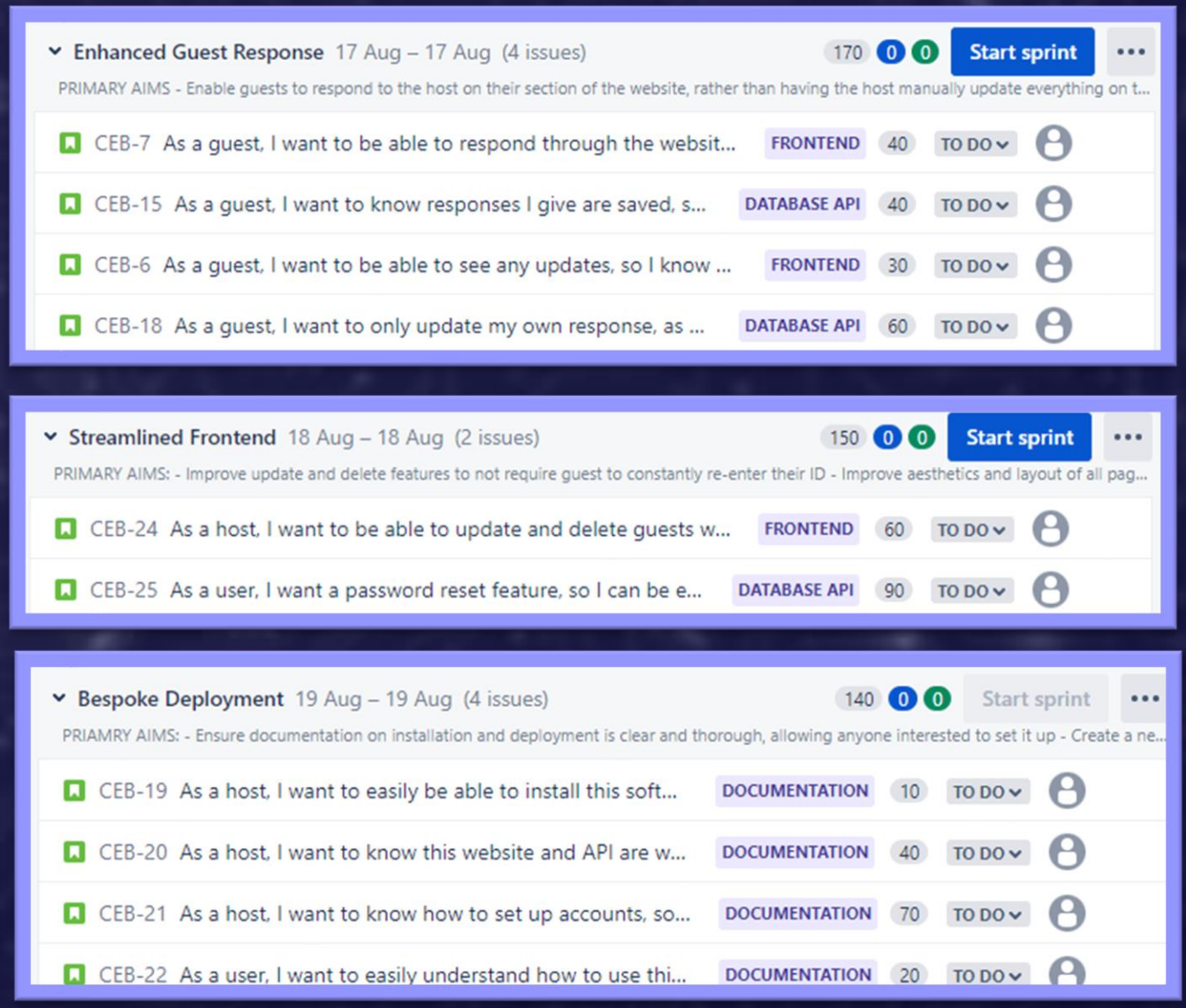


THE GOAL:
Create the minimum viable product; a website with CRUD functionality communicated through an API to a Database



THE GOAL:
Develop a functional login page, redirecting users to the appropriate section, and separating host/guest function between

ALL SPRINTS – MY ROADMAP



Enhanced Guest Response 17 Aug – 17 Aug (4 issues) 170 0 0 Start sprint

PRIMARY AIMS - Enable guests to respond to the host on their section of the website, rather than having the host manually update everything on t...

CEB-7	As a guest, I want to be able to respond through the websit...	FRONTEND	40	TO DO
CEB-15	As a guest, I want to know responses I give are saved, s...	DATABASE API	40	TO DO
CEB-6	As a guest, I want to be able to see any updates, so I know ...	FRONTEND	30	TO DO
CEB-18	As a guest, I want to only update my own response, as ...	DATABASE API	60	TO DO

Streamlined Frontend 18 Aug – 18 Aug (2 issues) 150 0 0 Start sprint

PRIMARY AIMS: - Improve update and delete features to not require guest to constantly re-enter their ID - Improve aesthetics and layout of all pag...

CEB-24	As a host, I want to be able to update and delete guests w...	FRONTEND	60	TO DO
CEB-25	As a user, I want a password reset feature, so I can be e...	DATABASE API	90	TO DO

Bespoke Deployment 19 Aug – 19 Aug (4 issues) 140 0 0 Start sprint

PRIAMRY AIMS: - Ensure documentation on installation and deployment is clear and thorough, allowing anyone interested to set it up - Create a ne...

CEB-19	As a host, I want to easily be able to install this soft...	DOCUMENTATION	10	TO DO
CEB-20	As a host, I want to know this website and API are w...	DOCUMENTATION	40	TO DO
CEB-21	As a host, I want to know how to set up accounts, so...	DOCUMENTATION	70	TO DO
CEB-22	As a user, I want to easily understand how to use thi...	DOCUMENTATION	20	TO DO

THE GOAL:
Relocate the response function to guests and introducing “edit event details” to the host

THE GOAL:
Finetune frontend design and enhance user experience

THE GOAL:
Finalise post-production documentation, introduce initial setup features enabling easier implementation

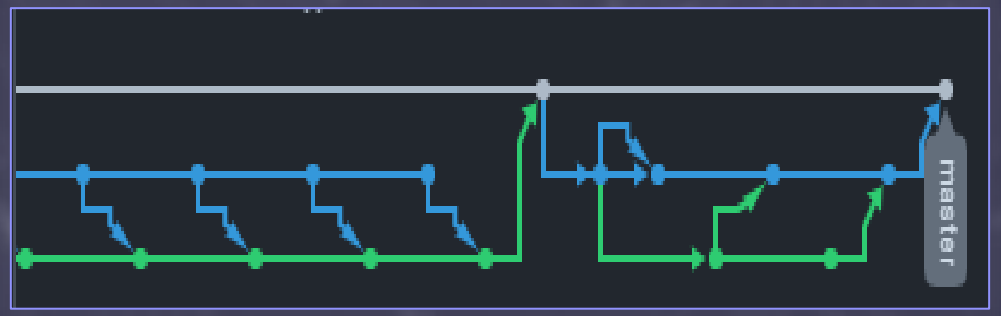
CONSULTANT JOURNEY

Front End

- HTML: creates the structure
  - Applying skills from QA
  - Understanding DOM
- CSS: creates the style
  - Improved bootstrap implementation
- JavaScript: creates the control
  - Understanding functions how they interact
  - Working with Axios for HTTP requests
  - Using localStorage to enable features
  - Using DOM to my advantage

Database and API

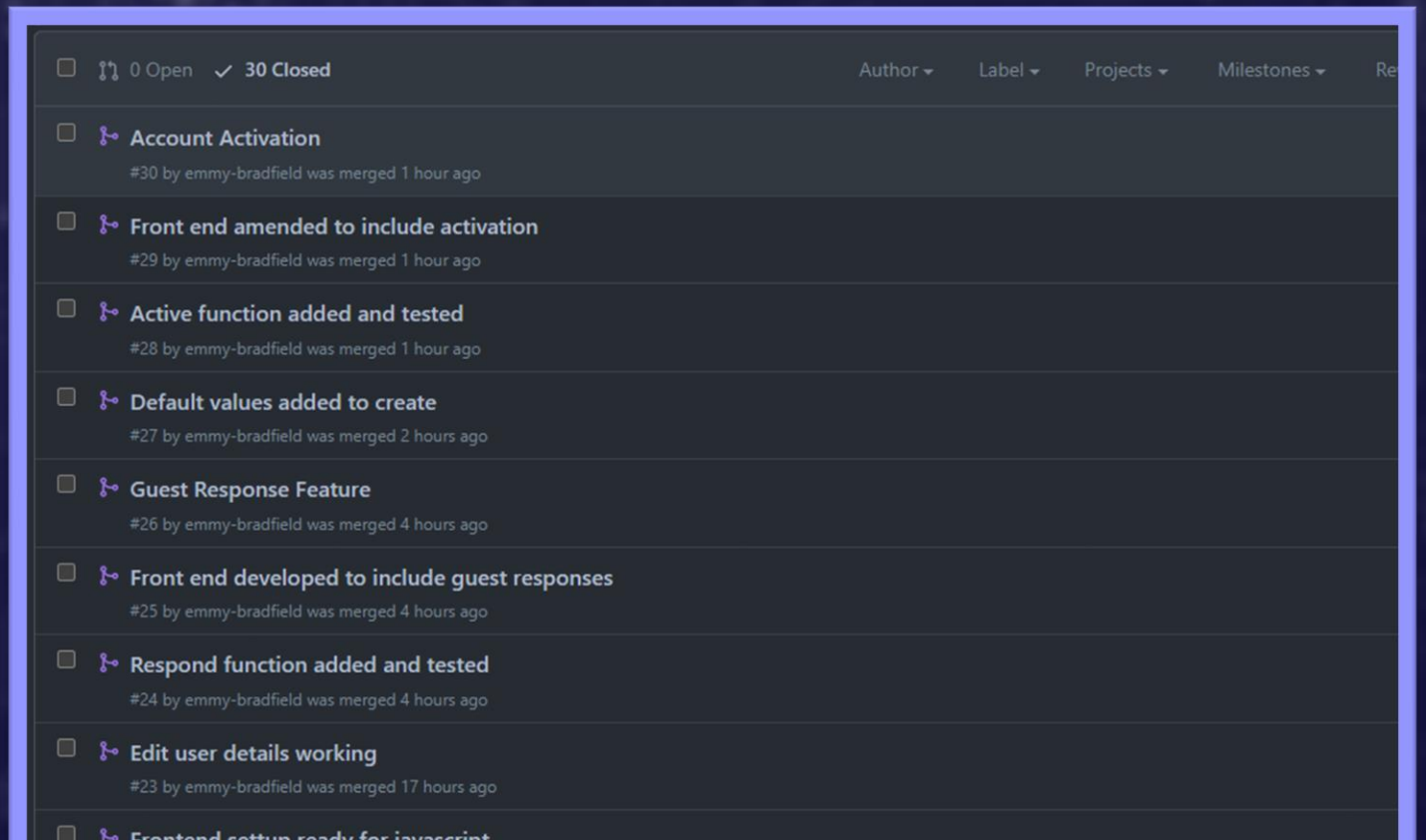
- MySQL: the database
  - Applying previous skills and knowledge
  - Implementing JPA to communicate with Java
  - Understanding entity relationships
- SpringBoot: the API
  - Developing CRUD functionality through HTTP requests
  - Use of repos, services, controllers, and domains
  - Ensuring separation of function as appropriate
  - Repeated and consistent testing throughout development



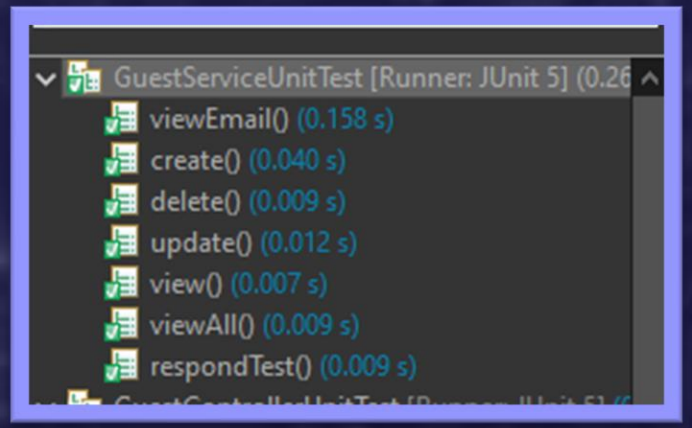
VERSION CONTROL

Through Git Bash and GitHub, the master/dev/feature model can be implemented to reduce risks when changing code or introducing features.

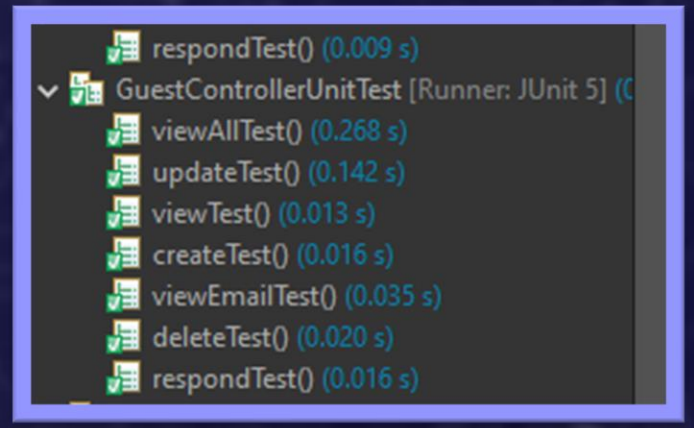
Document branch additionally used to separate development and documentation



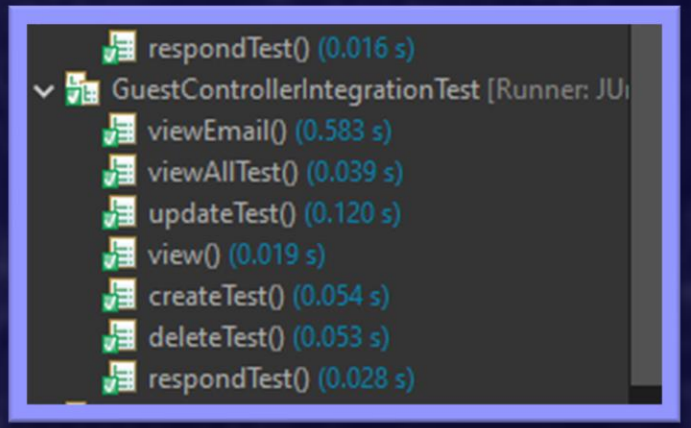
TESTING



Unit Testing on GuestService with Mockito and Assert Equals



Unit Testing on GuestController with Mockito and MockMvc



Integration Testing on GuestController with Mockito and MockMvc

FINAL COVERAGE:

Demonstration

Sprint Review

HOW DID IT GO?

SPRINT ONE
All tasks completed by end of day one and documented throughout

SPRINT TWO
All tasks completed by end of day, UMLs and Risk Assessment additionally completes and committed

SPRINT THREE
All tasks including “would” task completed by hallway through, presentation developed

SPRINT FOUR
Started sprint early, did not complete one user story – added to final sprint

SPRINT FIVE
Did not complete user story added from previous sprint, but completed all other sprint 5 tasks

Sprint Retrospective

HOW DID I LEARN?

PROJECT STRENGTHS

- Completed all tasks in 4/5 sprints
- Had sufficient time and room to expand on initial concept
- Development focused on one feature at a time
- MVP achieved and exceeded

AREAS OF GROWTH

- Define epics to avoid homogeneity between them
- Review approach to storypoint allocation
- Balance goals and timeframe

CONCLUSION

MY FINAL TAKEAWAY

What went well…

- Finished product which look good and works well
- Developed own knowledge through research
- Effectively managed time and MVP

I can grow from…

- Reviewing approach to storypoint rating
- Understand impact of documentation on planning

Project potential…

- Introduction of SMTP protocol
- Expansion of response options
- Party entity and structured party details

Thank you for your time!

ANY QUESTIONS?