

**Milestone 3**

*-*

By

Kelly Smith (s3898340)

Myat Theingi Nwe (Gigi) (s3963447)

Tanya Tran (s3843142)

Huy Do (s3894502)

Antoni Giannakopoulos (Toni) (s3895923)

Kevin Chen (s3780646)

Course: COSC2299 Software Engineering: Process and Tools

[Vision Statement 2](#_Toc700279239)

[Architecture/Design 2](#_Toc730234225)

[Refactoring 2](#_Toc598100085)

[Delta between milestone 1, 2 and 3 3](#_Toc1844521278)

[Git Organization 3](#_Toc925630353)

[Scrum Process 4](#_Toc70260240)

[Team Velocity: 5](#_Toc1350981059)

[Deployment Setup 6](#_Toc681449933)

[Docker image Upload to AWS ECR 7](#_Toc1684176513)

[Completed Custom User story 9](#_Toc526336785)

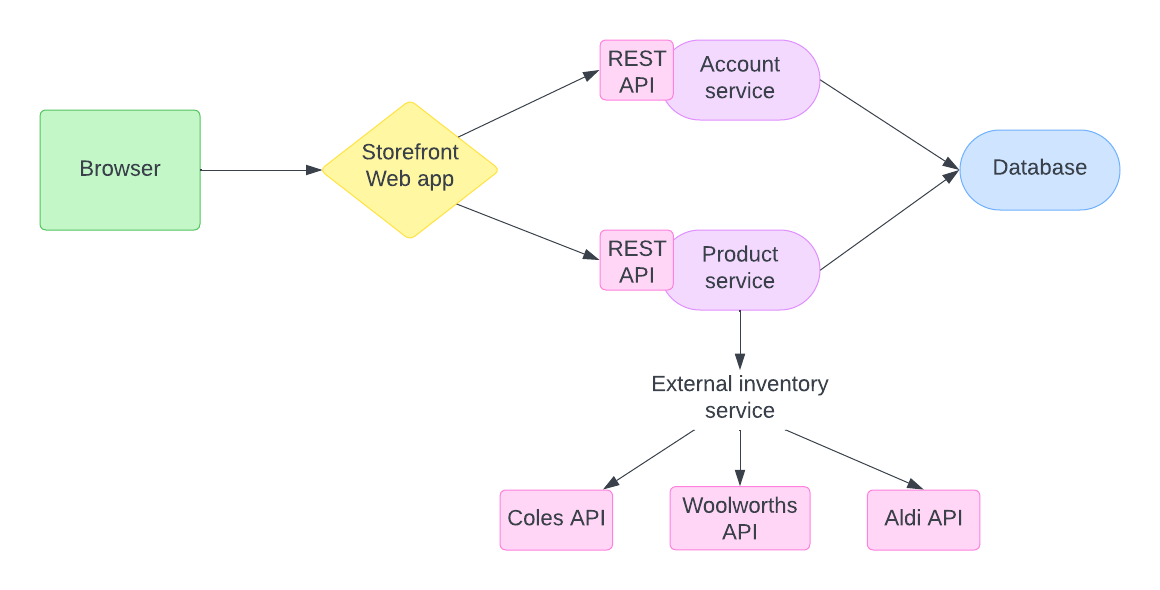
# Vision Statement

In today’s bustling world, the pursuit of value extends beyond mere costs; it encompasses time, effort, and overall experience. SuperPrice was developed with an understanding of the importance of saving time and providing optimal shopping experience without the hassle of surfing through multiple sites or visiting numerous stores.

Melbourne, with its diverse and dynamic population, stands at the forefront of our journey. By centralizing the price comparison of local supermarkets, we are not just ensuring users get the best financial deals but also promising them the gift of time and the removal of guesswork. Say goodbye to purchases and opportunities. Every time our users open the SuperPrice app they are guaranteed to find the value at their fingertips.

SuperPrice goes beyond being an app. It represents a promise. It’s about valuing convenience and revolutionizing the way we shop. Our vision, for Melbourne and eventually other cities is to create a community where grocery shopping is no longer a task but a smarter and more effortless choice, with SuperPrice.

# Architecture/Design



The architecture for the application has evolved to now draw from a single database. There are still two separate services which enable the web app to interact with the business logic.

# Refactoring

Once the base of the application was created in sprint 1, sprint 2 was where we focused on improving the functionality of the code base and increasing its resilience to errors. This meant expanding on the scope of each API and polishing up the front end to allow for a cleaner looking application. The addition of error handling meant that the application could now handle bad requests and conflicts coming from either the rest request or the database. These improvements were tasked due to either technical or non-functional requirements that were specified by the client. From there, quality assurance and tests revealed the weakness in the code and where there were gaps in error handling or implementation.

# Delta between milestone 1, 2 and 3

In Milestone 1, sprint 1 was conducted. The purpose of this sprint was to determine the scope, scale and constraints of developing such an application. In this time, the scrum leader was appointed, and the 6-person team was split up into three groups, two for backend, three for frontend and one for database. The user stories were created along with an SRS and an architecture style was decided on.

In Milestone 2, the framework for the application was built. Both the account and product APIs were created using an mvc structure. Methods were implemented to include GET, PUT and POST requests. On the front end, the landing, signup and login pages were implemented, and notification and profile pages were set up. The database was set up to include the tables for each of the relevant API’s and appropriate connections to the backend were established. By the end there was a working but unrefined application.

In Milestone 3, the code that was created in milestone 2 was improved upon and refined to create a complete application. Error handling and further unit tests were added, further features like DELETE requests, search and filtering were added in the backend. The front end was completed with styling and the finalization of the remaining pages. The database was updated with pictures and chain API’s within the database.

# Git Organization

This report provides an insightful exploration into the Git organization of our project, "SuperPrice." The project's primary objective is to create a comprehensive web application, encompassing frontend, backend, and database components. Developed collaboratively with students within the course.

The project repository, hosted on Git at the URL [https://github.com/cosc2299-sept-2023/team-project-group-p05-01], is the repository which the project lives.

Over the course of the semester, our team has made a commendable total of 35 commits, reflecting our dedication and progress in the development of "SuperPrice." With an average of 28 commits each week. A total of 59 pull requests across the front end and backend.

The git architecture can be found in CONTRIBUTING.md

contributors follow a structured process:

1. Create an Issue: Begin by creating an issue on GitHub to outline the problem or enhancement.
2. Branch Creation: Generate a branch using a specific naming convention reflecting the area, ticket number, and a brief description.  
   - Example: feature/frontend/25\_create\_contribution  
   - Example: bugfix/backend/123\_fix\_missing\_semicolon
3. Coding: Implement the changes required for the issue.
4. Commit Changes: Commit often using a message template: Task {TICKET\_NUMBER}: {DESCRIPTION}.  
   - Example: Task 25: create contribution.md with extra steps
5. Pull Request (PR): Create a Pull Request and get reviewers for feedback.
6. Review and merge: Address feedback and finalize changes.
7. Review issue ticket: update time spent and close ticket.

# Scrum Process

This semester-long project was conducted over the course of 9 weeks, with each sprint lasting 3 weeks. From the beginning of Sprint 0 leading up to the end of Sprint 2, our team has met for a total of 32 stand-up meetings. 9 of which were led by our course supervisor who also played the role of ‘Product Owner’.

In each stand-up meeting, every team member would share and discuss the following:

1. What have you worked on since the last meeting?
2. What will you be working on leading up to the next meeting?
3. Are there any blockers or impediments affecting your progress?

Upon everyone providing an update of their work, there would also be further group discussions about any concerns that we had with relevance to the project itself and/or the milestone assignment submissions.

Scrum Meetings Summary:

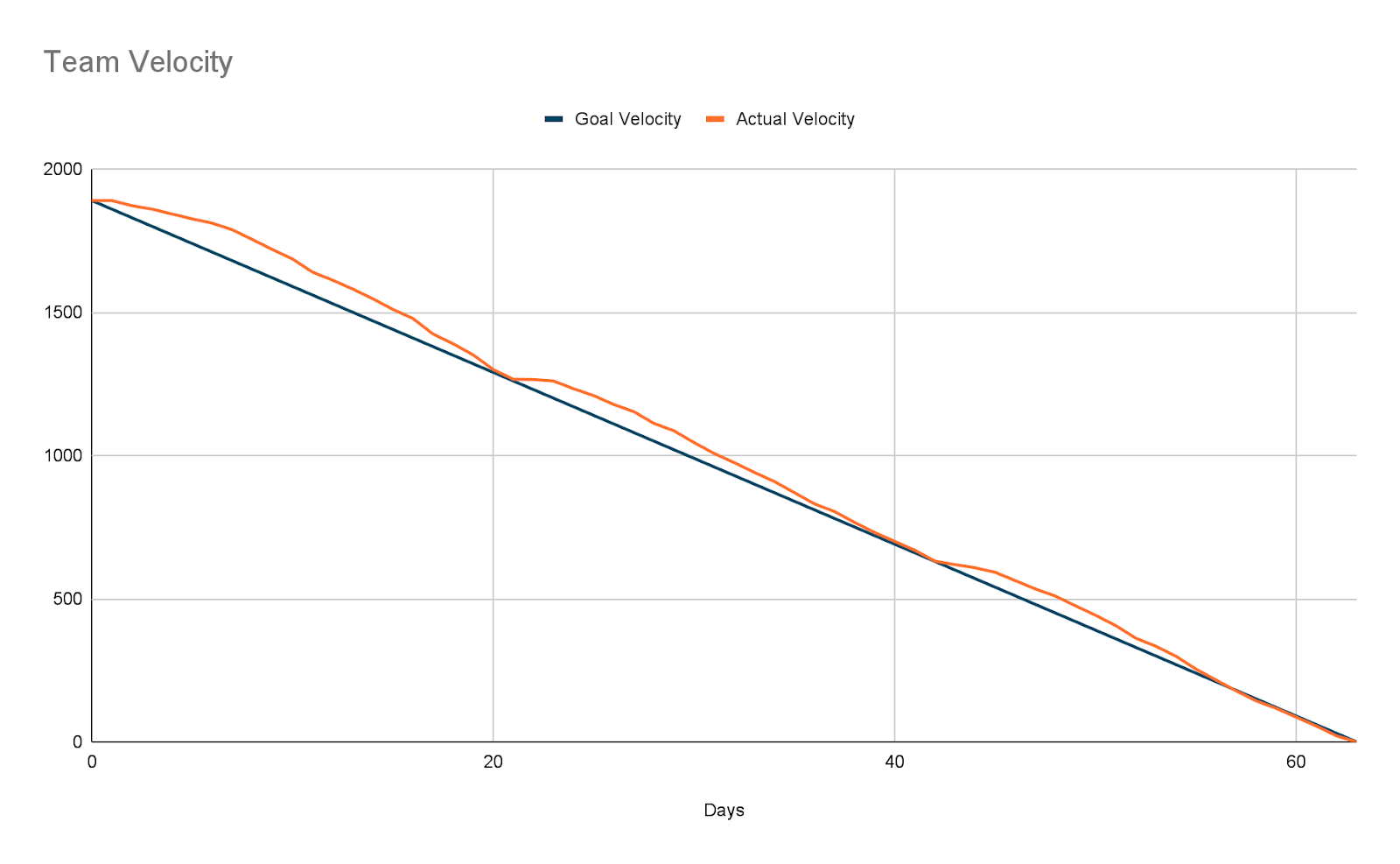
Scrum Master: Kevin Chen

|  |  |  |
| --- | --- | --- |
| **Sprint 0** | **Sprint 1** | **Sprint 2** |
| Total scrum meetings: 8  Stand-ups with supervisor: 3  Average meetings per week: 2-3 | Total scrum meetings: 14  Stand-ups with supervisor: 3  Average meetings per week: 3-4 | Total scrum meetings: 10  Stand-ups with supervisor: 3  Average meetings per week: 2-3 |
| Attendance:  Gigi - 5/8  Huy - 7/8  Keely - 6/8  Kevin - 8/8  Tanya - 7/8  Toni - 7/8 | Attendance:  Gigi - 12/14  Huy - 14/14  Keely - 12/14  Kevin - 11/14  Tanya - 13/14  Toni - 10/14 | Attendance:  Gigi - 8/10  Huy - 9/10  Keely - 8/10  Kevin - 8/10  Tanya - 9/10  Toni - 6/10 |
| In Sprint 0, all scrum meetings with the course supervisor were held on campus during our practical class time. The other 5 scrum meetings were conducted online via Discord. | In Sprint 1, all scrum meetings with the course supervisor were held online through MS teams. The other 11 scrum meetings were conducted online via both MS teams and Discord. | In Sprint 2, all scrum meetings with the course supervisor resumed back on campus during our practical class time. The other 7 scrum meetings were conducted online via Discord. |

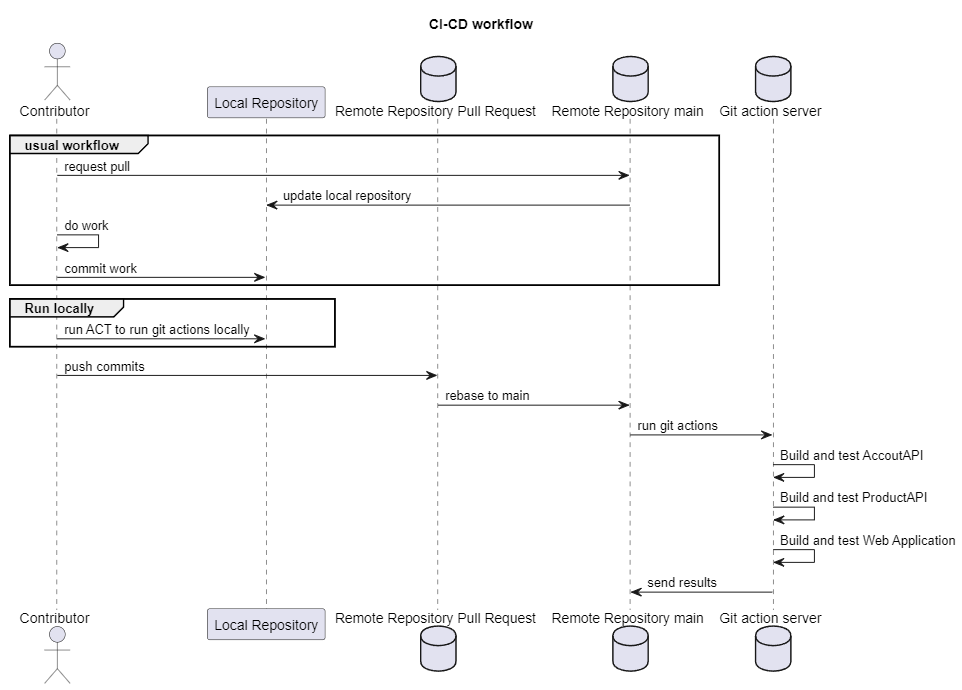
As each individual team member has worked closely on this project throughout the semester, we’ve all recognised a significant improvement in how our scrum process began, compared to how it has now concluded. By initiating productive stand-ups every second day, it has noticeably enhanced our communication, transparency and collaboration within the team. Whilst also promoting accountability and enabling problem solving, our scrum process has ultimately helped the team stay aligned with our project goals.

# Team Velocity:

The following line chart depicts the project progress velocity over the course of the semester. Our team has maintained a consistent level of development and contribution in every sprint.

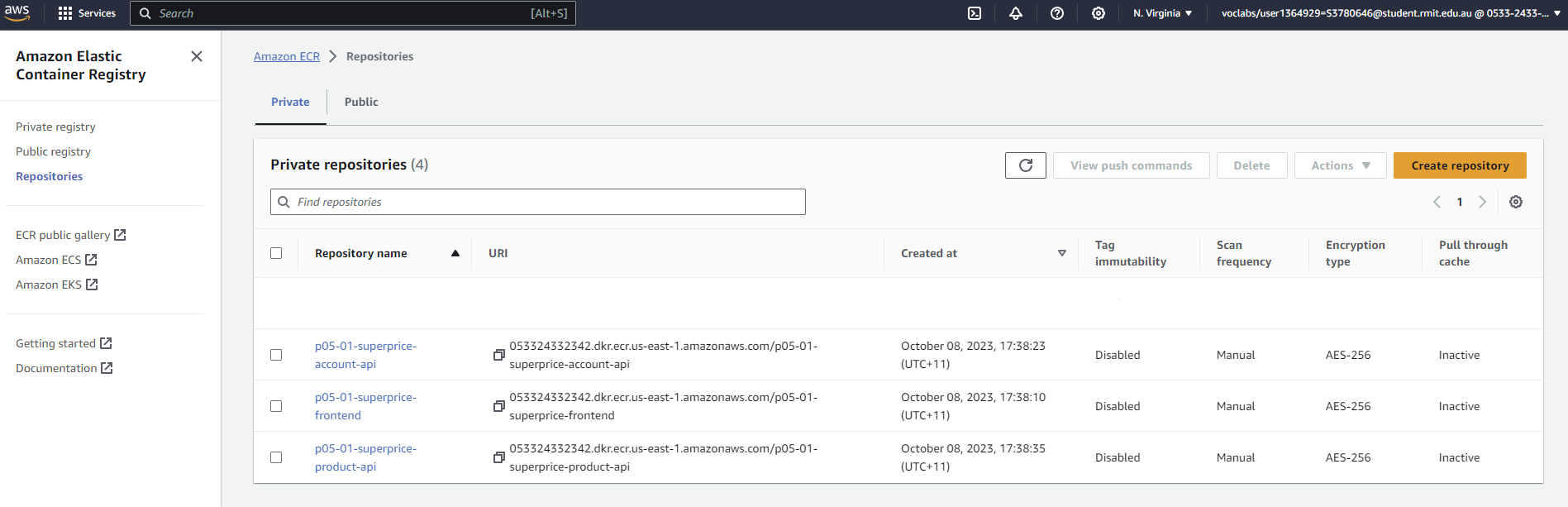
As shown, it is evident that our team initially experiences slower progress at the beginning of all sprints. However, as we approach the midpoint and the end of each sprint, our group becomes notably more productive in accomplishing tasks.   


# Deployment Setup

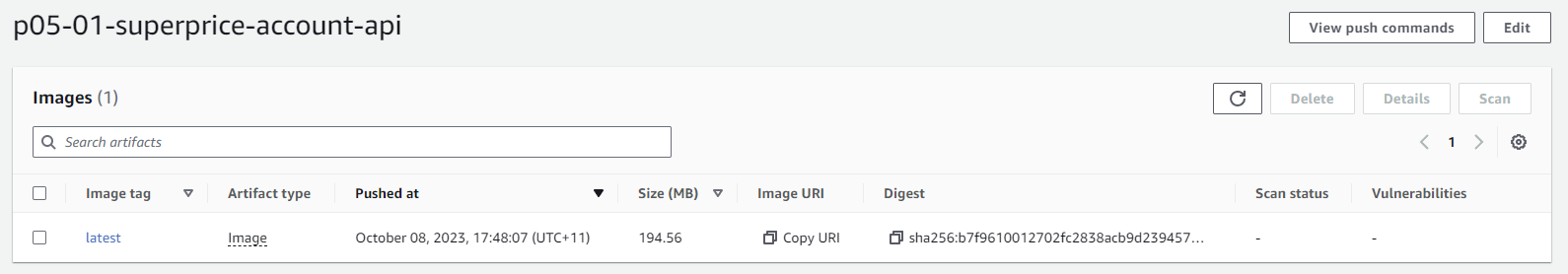


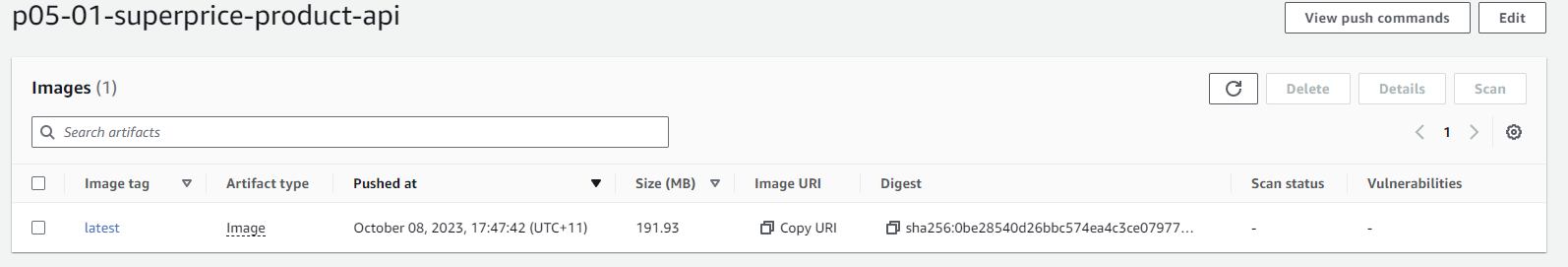
# Docker image Upload to AWS ECR

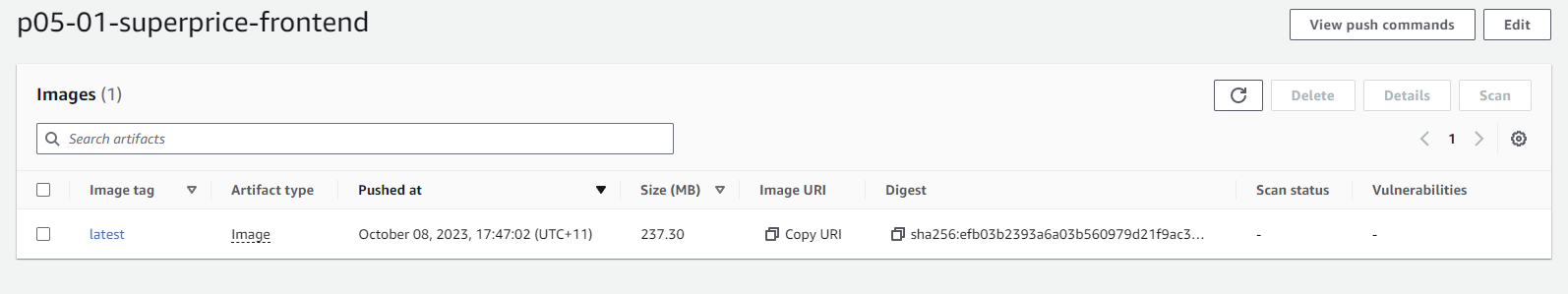
Below is the image of the 3 different AWS ECR repositories, 1 frontend and 2 backend (account API, product API).

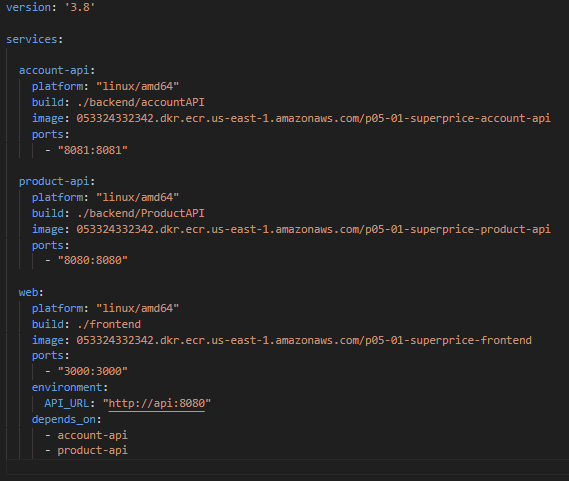


The first repository, AccountAPI, hosts the uploaded images essential for the project to access the account information.

The second repository, ProductAPI, stores project images vital for its functionality to be able to access product information.

The third repository, Frontend, houses project images that are crucial for the frontend interface.

These images, manually uploaded, are used within the Docker Compose file, enhancing the overall project functionality.



# Completed Custom User story

A feature that is not present in the custom user story is to be able to create an account and have a personalized view of the application.

In summary, the absence of the account creation and personalized views feature not only limits user interaction but also hinders the application's potential for growth and user satisfaction. Incorporating this functionality will not only meet the basic expectations of modern users but will also elevate the application to a level where it can truly cater to individual needs, fostering a loyal user base and positive word-of-mouth promotion.