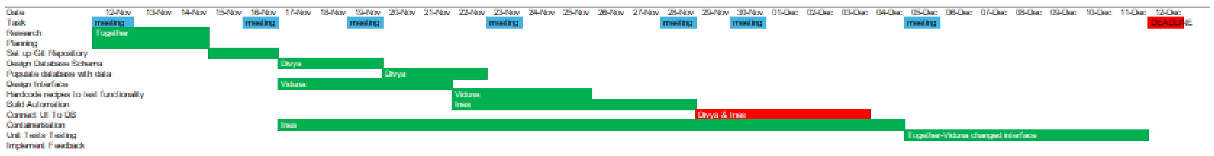
Prototype Report

What was accomplished (200 word summary of which EDR was chosen, any changes made, and what was accomplished overall.)

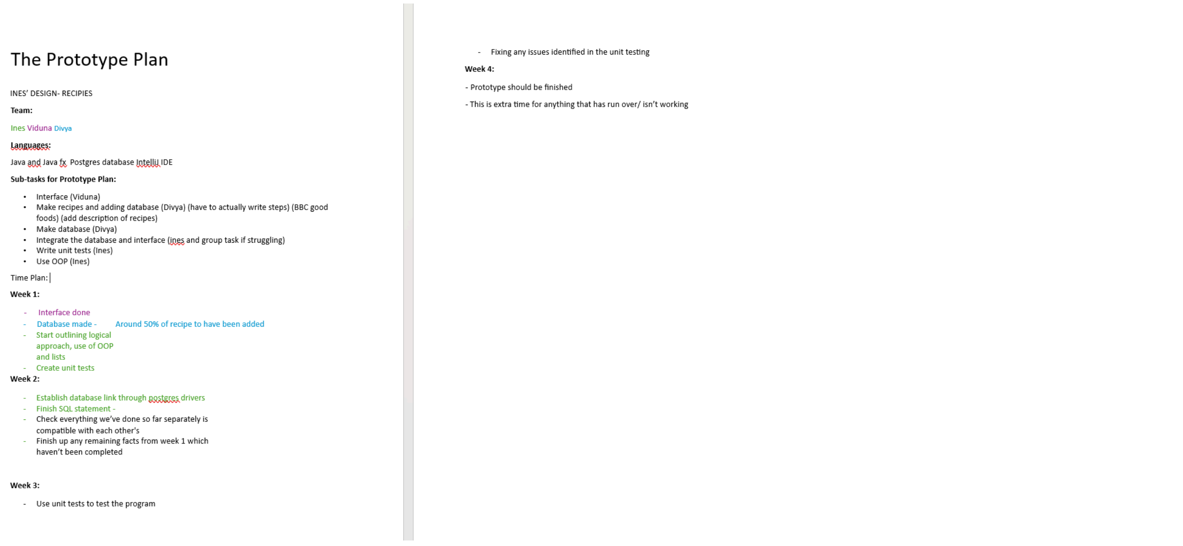
The EDR chosen was one on the ‘SSH Recipe’ service, which describes a new section in the SSH App. Using data about ingredients in the student's fridge from the SSH Camera, the app displays recipes relevant to the student.
It was planned that the program would display only recipes the student has all ingredients for, unless otherwise prompted otherwise by the student. However in the prototype, the ‘show me recipes I don't have all the ingredients for’ button was removed. Instead, recipes the student has at least one ingredient for are displayed, sorted inversely by the number of missing ingredients. This change was made as many recipe ingredients are not typically stored in a fridge, making it difficult to confirm if all ingredients are available.
A recipe database was created, using PostgreSQL, to store data displayed about the recipes. However, the unit and ingredient table were merged into one instead of being kept separate.
Fridge ingredient data is passed as a list when the main JavaFX file runs.
We were able to successfully implement the EDR as a prototype by displaying recipes relevant to the fridge contents, filtered and fetched from the database. Based on our goals, our prototype was successful.

How it was accomplished (800 word report on how the prototype was accomplished, with an emphasis on demonstrating aspects of the group work that aren’t reflected in the code)

We started the project by selecting an EDR and ensuring everyone reviewed it before our first team meeting. This allowed us to clarify uncertainties and address any concerns about the design’s feasibility.
At our first team meeting, we discussed logical aspects of the design, and this is where functions like the ‘show me recipes I don't have all the ingredients for’ were removed as we felt this would not be a viable implementation. We also discussed other design questions and collaboratively assessed their viability.
Once the design was finalised and viability determined, we were each assigned one general area and were given clear, concise deadlines which aligned with our weekly meetings. At these meetings, we each reported our progress, unexpected errors and unforeseen tasks. This progress was then tracked on a gantt chart.
We chose to use a Gantt chart as a project management tool to provide us with a visual representation of the project timeline and its tasks. Using the Gantt chart we decomposed the project to sub-tasks, easily assigning responsibilities and setting realistic deadlines. Dependencies between tasks were highlighted and we were able to take this into consideration when planning. At any point in time, we could check our progress and identify if we were falling behind or amend the chart due to unforeseen issues. This ensured clear communication of our roles and responsibilities.



Below is the original separation of tasks and goals, created during the first group meeting.



The Prototype Plan

INES' DESIGN- RECIPIES

Team:

Ines Viduna Divya

Languages:

Java and Java fx Postgres database IntelliJ IDE

Sub-tasks for Prototype Plan:

- Interface (Viduna)
- Make recipes and adding database (Divya) (have to actually write steps) (BBC good foods) (add description of recipes)
- Make database (Divya)
- Integrate the database and interface (ines and group task if struggling)
- Write unit tests (Ines)
- Use OOP (Ines)

Time Plan:

Week 1:

- Interface done
- Database made - Around 50% of recipe to have been added
- Start outlining logical approach, use of OOP and lists
- Create unit tests

Week 2:

- Establish database link through postgres drivers
- Finish SQL statement -
- Check everything we've done so far separately is compatible with each other's
- Finish up any remaining facts from week 1 which haven't been completed

Week 3:

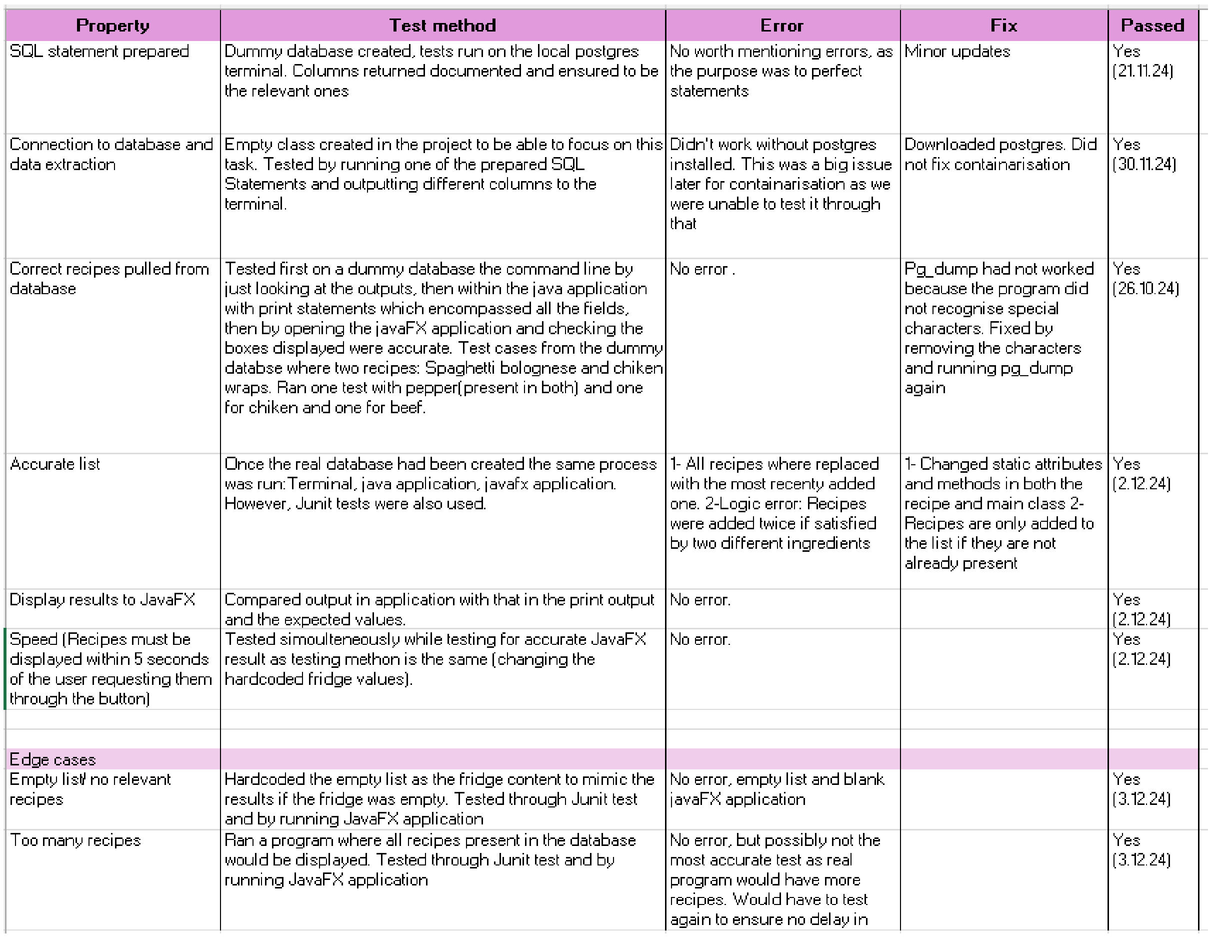
- Use unit tests to test the program
- Fixing any issues identified in the unit testing

Week 4:

- Prototype should be finished
- This is extra time for anything that has run over/ isn't working

Development through testing

Testing was used throughout development.
For example, to test accurate database connection, a separate program file was created and, once the code was perfected, copied to the main file. This saved time as large errors did not need to be debugged to test the connection. This file was then used for observability testing by adding print statements for attributes being called from the database, avoiding having to load the whole JavaFX file each time.
Testing was also conducted via JUnit tests on java, ensuring list outputs matched up with the expectation.
Below is a table detailing the process of property based testing, while outlining the development of the program.
Due to the short form nature of the prototype, once it was finished smoke testing was easily implemented and similar to the later property testing.



Property	Test method	Error	Fix	Passed
SQL statement prepared	Dummy database created, tests run on the local postgres terminal. Columns returned documented and ensured to be the relevant ones	No worth mentioning errors, as the purpose was to perfect statements	Minor updates	Yes (21.11.24)
Connection to database and data extraction	Empty class created in the project to be able to focus on this task. Tested by running one of the prepared SQL Statements and outputting different columns to the terminal.	Didn't work without postgres installed. This was a big issue later for containarisation as we were unable to test it through that	Downloaded postgres. Did not fix containarisation	Yes (30.11.24)
Correct recipes pulled from database	Tested first on a dummy database the command line by just looking at the outputs, then within the java application with print statements which encompassed all the fields, then by opening the javaFX application and checking the boxes displayed were accurate. Test cases from the dummy databse where two recipes: Spaghetti bolognese and chiken wraps. Ran one test with pepper(present in both) and one for chiken and one for beef.	No error .	Pg_dump had not worked because the program did not recognise special characters. Fixed by removing the characters and running pg_dump again	Yes (26.10.24)
Accurate list	Once the real database had been created the same process was run:Terminal, java application, javafx application. However, Junit tests were also used.	1- All recipes where replaced with the most recenty added one. 2-Logic error: Recipes were added twice if satisfied by two different ingredients	1- Changed static attributes and methods in both the recipe and main class 2- Recipes are only added to the list if they are not already present	Yes (2.12.24)
Display results to JavaFX	Compared output in application with that in the print output and the expected values.	No error.		Yes (2.12.24)
Speed (Recipes must be displayed within 5 seconds of the user requesting them through the button)	Tested simoulteneously while testing for accurate JavaFX result as testing methon is the same (changing the hardcoded fridge values).	No error.		Yes (2.12.24)
Edge cases				
Empty list/ no relevant recipes	Hardcoded the empty list as the fridge content to mimic the results if the fridge was empty. Tested through Junit test and by running JavaFX application	No error, empty list and blank javaFX application		Yes (3.12.24)
Too many recipes	Ran a program where all recipes present in the database would be displayed. Tested through Junit test and by running JavaFX application	No error, but possibly not the most accurate test as real program would have more recipes. Would have to test again to ensure no delay in		Yes (3.12.24)

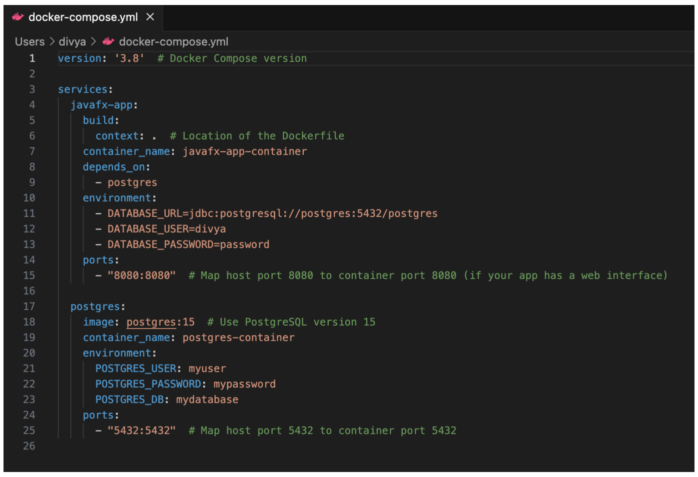
We used GitHub Actions to automate testing, building and deployment processes; the mavenbuild.yaml file defines workflows in a CI/CD pipeline, ensuring that all changes pushed to the repository are automatically built, tested and deployed for multiple versions of Java. Caching Maven dependencies significantly reduces build times for subsequent runs, promoting early detection of any integration issues and ensures stability. During one of our builds, we encountered the error: "Failed to execute goal org.apache.maven.plugins:maven-compiler-plugin:3.11.0:compile on project SSH_App: Fatal error compiling: error: invalid target release: 21". This error occurred as the pom.xml file specified a Java compiler target version (21) was incompatible with the Java version used in the build environment. After updating the pom.xml file, the error was resolved.

We used maven for dependency management; the pom.xml file specifies project structure, dependencies and build instructions for the project. Furthermore, Maven facilitates reproducibility and standardization across different environments, making it easier to onboard new developers and maintain consistent builds.

Database

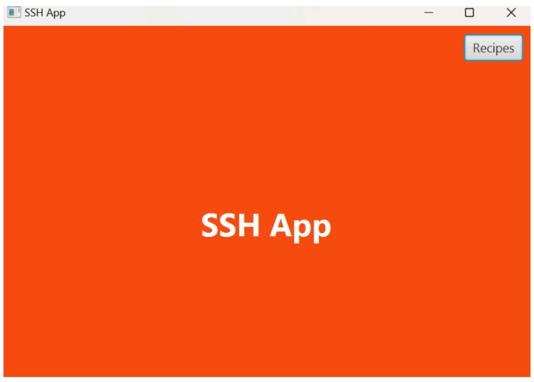
The basic design for the Postgres database was taken from the EDR. Since it was mentioned the original design may require adjustment, the design was reevaluated and a few edits were made. Afterwards, the database was created and corresponding tables populated with recipes, ingredients, etc. Once all SQL statements were completed and had been uploaded to Github, whilst editing the database, it was accidentally dropped and all the information had been lost. Using version control, a previous version of the database was able to be restored, efficiently preserving days of progress and minimising setbacks. When integrating it with the program, we encountered an error 'UTF-8 codec can't encode character '\udc91' in position X: surrogates not allowed', only being returned when connecting the Instructions table. During the next group meeting we went over the error together and figured out the error was due to using letters unrecognised in UTF-8, for example 'Tomato Purée' was not compatible due to the 'é'. After identifying the problem, rows with illegal characters were dropped, edited, and reinserted, then the changes were recommitted and the connection worked for all tables successfully.

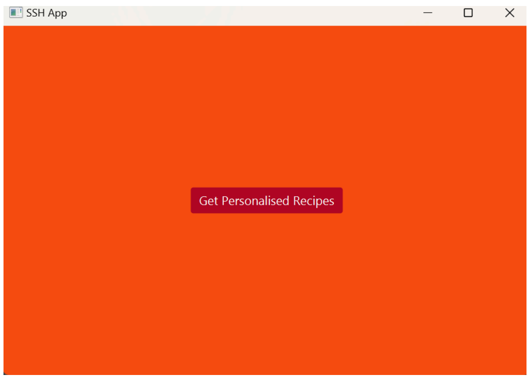
Attempts were made to implement containerisation to the prototype, however we were not able to within the timeframe. While developing we used Maven, however to attempt getting the Dockerfile to work we switched to Gradle to use the code demonstrated in lectures. This was ultimately unsuccessful and we chose to prioritise other aspects of the project.
We were able to dockerize the Java application, however this did not pass the dependencies.



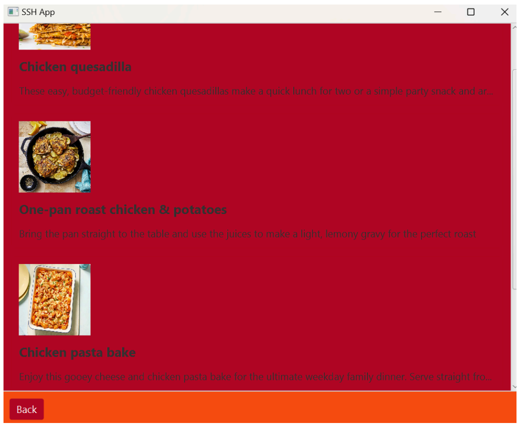
```yaml
version: '3.8'  # Docker Compose version

services:
  javafx-app:
    build:
      context: .  # Location of the Dockerfile
    container_name: javafx-app-container
    depends_on:
      - postgres
    environment:
      - DATABASE_URL=jdbc:postgresql://postgres:5432/postgres
      - DATABASE_USER=divya
      - DATABASE_PASSWORD=password
    ports:
      - "8080:8080"  # Map host port 8080 to container port 8080 (if your app has a web interface)

  postgres:
    image: postgres:15  # Use PostgreSQL version 15
    container_name: postgres-container
    environment:
      POSTGRES_USER: myuser
      POSTGRES_PASSWORD: mypassword
      POSTGRES_DB: mydatabase
    ports:
      - "5432:5432"  # Map host port 5432 to container port 5432

```

Below are a few examples of the output with different input values.





Input= “Chicken”



Chicken quesadilla

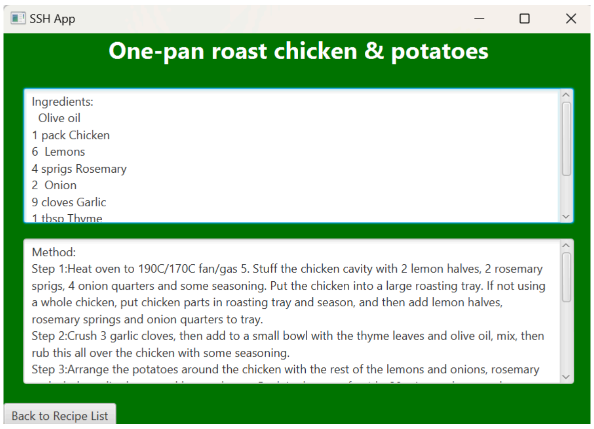
These easy, budget-friendly chicken quesadillas make a quick lunch for two or a simple party snack and ar...

One-pan roast chicken & potatoes

Bring the pan straight to the table and use the juices to make a light, lemony gravy for the perfect roast

Chicken pasta bake

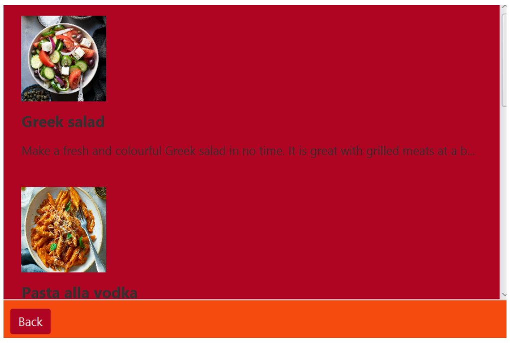
Enjoy this gooey cheese and chicken pasta bake for the ultimate weekday family dinner. Serve straight fro...



One-pan roast chicken & potatoes

Ingredients:
 Olive oil
1 pack Chicken
6 Lemons
4 sprigs Rosemary
2 Onion
9 cloves Garlic
1 tbsp Thyme

Method:
Step 1:Heat oven to 190C/170C fan/gas 5. Stuff the chicken cavity with 2 lemon halves, 2 rosemary sprigs, 4 onion quarters and some seasoning. Put the chicken into a large roasting tray. If not using a whole chicken, put chicken parts in roasting tray and season, and then add lemon halves, rosemary springs and onion quarters to tray.
Step 2:Crush 3 garlic cloves, then add to a small bowl with the thyme leaves and olive oil, mix, then rub this all over the chicken with some seasoning.
Step 3:Arrange the potatoes around the chicken with the rest of the lemons and onions, rosemary



Greek salad

Make a fresh and colourful Greek salad in no time. It is great with grilled meats at a b...

Pasta alla vodka

Input= Cheese, Onions, Tomato

Ines’ report on the team

250 words on Divya

Divya’s first task was to establish the postgres database with the tables outlined in the original EDR. She also had to find enough recipes for appropriate testing and fill the database with them. As she was finishing up this task, she communicated with me that she had accidentally dropped the entire database, however, she took full responsibility and because of her version control she was able to get it back to a previous stage. While creating the database, she altered the layout of the tables and column names. While this was not an issue, as the design in the EDR was only a suggestion, it was not communicated until a later team meeting, after I had begun working on the SQL statements. While this was not a major fault I would have suggested she communicated this earlier, as I had started working on my SQL statements.
After this stage, she became responsible for making use of Docker and implementing containerisation. It’s worth mentioning that Divya was throughout most of the course of the assignment struggling with her general course load and mental health. However, because she communicated her issues and where she was at with us, we were able to accommodate her and provide support. This meant I was also working on containerisation and creating a dockerfile for the project. Despite this, we were not able to implement Docker onto the program on time.

Divya was very capable at separating tasks and leading the responsibility separation meetings. Considerate and fair, she was very efficient and organised when it came to the project management part of the assignment.

250 words on Viduna
Viduna's original task was to create the base JavaFX application using hardcoded values. She completed this in a surprising time frame, making it easy for us to stay on schedule, especially as I had to remove the hardcoded values and link to the database.
After this, she was able to identify key missing parts in our program, mainly to do with github and the project structure. Very independently, she created the build automation tools present in our git repository. This was very useful to the team as it is an important part of development and code quality. However, it could be said that this took us by surprise as she had not communicated her task to us. Nevertheless she explained all the files and their usage when we asked her for clarification.
She was very adept at identifying the gaps in our project, keeping us up to standard. By researching and hard work she ensured we were making use of gitHubs build tools and was our point of reference with most things related to file structure.
She was really committed to understanding the assignment brief and to contributing to the team, and hence made a wonderful team mate and addition to the project.
She was very eager to help any of us in our own tasks and was very understanding when we communicated any struggles.

Divyas' report on the team

250 words on Ines
Ines' task at first was to research and start thinking and researching how we would integrate the JavaFX application and Postgres database together. She was also responsible for writing some JUnit tests and going over Object Oriented Programming, to help her later as she was responsible for that part of the prototype. She was very good at taking responsibility for this and by our next team meeting she had completed what she said she would. The tests she had created, while she did complete them, were not very thorough and would not be usable for in-depth testing throughout development. However, in the end she did make more suitable tests that covered more cases and were useful later in development. Ines was excellent at coming up with quick solutions to problems that arose and was very good at taking responsibility for new tasks that came up throughout the development stage. When I communicated my struggles with the group, she didn't hesitate to work with me to help try and implement the containerisation, even though in the end neither of us were able to implement it. She was an excellent teammate and delivered what she was assigned to the highest standard, and also adept at communicating to the group about what she was working on and how she was progressing.

250 words on Viduna
Viduna was first assigned the creation of the JavaFX application, originally using hardcoded values which would later be changed to the values received from the Postgres database. She impressively completed this task in a very fast timeframe, which was beneficial to the group and meant that we were ahead of schedule from early on. She also took it upon herself many times to look at the project structure and fill in any gaps we were missing, meaning the group was always on task. She also made sure she fully understood the

different environments we were using and was able to help the rest of the group if we were struggling. At times Viduna lacked communication to the rest of the group, meaning we were unclear on what she was doing. However, due to her independence we got multiple tasks done that hadn't been considered during the initial task allocation. Overall, she was a very valuable member of the team and it was great to work with her.

Viduna' report on the team

250 words on Inés

Inés was responsible for the integration of the front end and back end of the application. During the entirety of the project, Inés communicated to us her efforts each time, making it possible for the team to monitor the progress made in the project, and ensuring that we managed to deal with any issues that arose. If there were to be a problem, Inés was usually first to take initiative and come up with a solution ensuring that the problem would be dealt with swiftly. During the initial phases of the project, Inés independently commenced the creation of JUnit tests which were not necessary to have been done at that moment. She didn't inform us of this decision, but due to her proactive approach we managed to be ahead of schedule. Inés' creation of JUnit tests was particularly vital; initially these tests were not the most precise but in the end these tests created a framework that checked the code's functionality, making the project more reliable and robust overall. Her attention to detail and commitment to quality were evident throughout the project, and it was clear that she invested significant effort and time in ensuring the system's reliability.

I had a great time working on this project with Inés. Going forward, she should continue her collaborative approach and her open communication in the future as it really aided this project.

250 words on Divya

Divya took initiative right away and created a document that we used to specify what was expected of each team member and what had to be done at the end of each week. Her proactive approach in arranging this plan helped make our workflow efficient and ensured that we stayed on track throughout the project as well as allowing us to monitor the progress of the project. Divya was tasked with setting up the database, accurately organizing the tables and fields required for our data as per the engineering design review. She took the time to research and implement photos in the database, which other members were unsure of how to do, in the end it really improved the user experience of the project. At times she lacked communication, which was because she was struggling with her mental health. Divya explained database queries in a clear and understandable manner, making difficult concepts comprehensible to the rest of the team when we met. When we suggested improvements to the schema, Divya quickly updated the database, which showed her excellent collaboration and flexibility. Her patience and willingness to listen to suggestions were especially helpful.

Working with Divya was a wonderful experience; I would suggest that Divya maintain more consistent communication as well as continue to share her technical knowledge with team members, as her detailed explanations are key to making sure everyone is on the same page and ensuring the success of the project for the future.