CS3354 Software Engineering
Final Project Deliverable 2

PantryPal

Chitsein Htun
Shrey Joshi
Prateek Mishra
Ayah Elkhaled
Eric Wang
Neil Agrawal
Edgar Lara Sanchez

1. [5 POINTS] Well described delegation of tasks, i.e. who did what in the project. Now that your project is complete, you are required to submit the delegation of tasks from the beginning of the project until the end.

Edgar Lara Sanchez: worked on question 5, Software Requirements Functional and Non-functional. **For deliverable 2**: worked on question 3.1 Project scheduling, 3.2 Cost-effort-and pricing estimation, 3.5 Estimated cost of personnel.
Chitsein Htun - Worked on question 4, software process model. **For deliverable 2:** Worked on instructor feedback response and redoing use case and sequence diagrams according to feedback.
Shrey Joshi - Worked on question 7, Sequence Diagrams. **For deliverable 2:** Worked on Section 8 slides.
Prateek Mishra - Question 2 part 1.5. **For deliverable 2:** Worked on question 6 Conclusion
Ayah Elkhaled- Worked on question 9, Architectural design. **For deliverable 2:** Worked on question 3.3 Estimated hardware cost, 3.4 Estimated software cost, 5 Comparison to similar designs
Eric Wang - Worked on question 7, Sequence Diagrams. **For deliverable 2:** Worked on question 4. A Test plan for our software (JUnit)
Neil Agrawal - Worked on question 6, Use Case Diagram. **For deliverable 2:** Worked on Section 8 Slides

2. Project Deliverable 2 content.

1. **[5 POINTS] Please attach here the Final Project draft description (that contains the instructor feedback). It is ok to include a picture of the original document. Address the feedback provided for your proposal by listing what you did / plan to do to comply with those proposed changes and or requests for additions to your project.**

Project Proposal:
See link to document: CS 3354 Group Project Proposal
OR tinyurl.com/pantrypal-proposal

Instructor Feedback:
The use case diagram is wrong. There is nothing like pantry, social media in the use case diagram as actors.
The sequence diagrams should have response arrows for every request arrow.

Response to Feedback:
We have decided to remove the social media aspect of our app. Additionally, we added the request arrows to the first sequence diagram which was missing them.

2. **[10 POINTS] Setting up a Github repository. Please use your utdallas email accounts only for each group member.**

URL of Project Repository: https://github.com/exl190014/3354-PantryPals

3. **[5 POINTS] Delegation of tasks: Who is doing what. If no contribution, please specify as it will help us grade each group member fairly.**

For this deliverable,
Edgar Lara Sanchez: worked on question 5, Software Requirements Functional and Non-functional
Chitsein Htun - Worked on question 4, software process model.
Shrey Joshi - Worked on question 7, Sequence Diagrams.
Prateek Mishra - Question 2 part 1.5.
Ayah Elkhaled- Worked on question 9, Architectural design.
Eric Wang - Worked on question 7, Sequence Diagrams.
Neil Agrawal - Worked on question 6, Use Case Diagram.

4. **[5 POINTS] Which software process model is employed in the project and why. (Ch 2)**

We chose the prototype model because we believe that since this project's final product form will be a full fledge app, creating iterative prototypes that improve the app's functionality and user experience. Choosing this evolutionary model type is necessary for our app to be continually be updated and improve the user experience to stay relevant in its specific market. The prototype model allows users to give a lot of feedback during the creation of the app and different iterative updates, which is useful for making sure that the app is serving its purpose and to ensure user satisfaction. Additionally, this added factor of quick feedback and low-cost and efficient development will make our project cost effective while being able to experiment with different ideas and features. Finally, the prototype model will help make our project more adaptable to change and reduce the time it takes to get the product to market.

5. **[15 POINTS] Software Requirements including**
    a. **Functional requirements. To simplify your design, please keep your functional requirements in the range minimum 5 (five) to maximum 7 (seven). (Ch 4)**

    User Registration and Login: PantryPal should provide a user registration process, collecting essential user information, and ensuring secure login for registered users and have password recovery options.

    Inventory Management: PantryPal enables manual input and camera scanning for adding food items to the inventory, with automatic categorization and sorting for easy retrieval.

    Calorie Intake Tracking: PantryPal allows users to establish daily caloric intake goals, record daily food consumption, and receive real-time updates on their progress.

    Recipe Recommendations: Based on the items in the user's inventory, PantryPal should be capable of suggesting recipes that can be prepared using those ingredients.

    Notification and Expiry Alerts: PantryPal sends users customizable notifications for impending food item expirations, allowing them to set timing and frequency preferences.

    Reporting and Insights: The application must provide users with thorough food consumption reports for habit analysis, offering the option to view weekly or monthly summaries for informed dietary decisions.

Social Sharing: PantryPal allows users to directly share their dietary habits and achievements on social media, with customizable post content and appearance options.

b. Non-functional requirements (use all non-functional requirement types listed in Figure 4.3 - Ch 4. This means provide one non- functional requirement for each of the leaves of Figure 4.3. You can certainly make assumptions, even make up government/country based rules, requirements to be able to provide one for each. Please explicitly specify if you are considering such assumptions.)

Usability Requirements: Intuitive and user-friendly interface

Dependability requirements: Should ensure high system availability

Operational requirements: The system should operate 24/7 with reliable data backup.

Regulatory requirements (Assumption): PantryPal is expected to comply with data protection regulations, like GDPR.

Ethical requirements (Assumption): PantryPal should promote responsible food management.

Environmental requirements (Assumption): PantryPal should encourage eco-friendly choices.

Development requirements: Should follow industry-standard software development practices, including version control, code review, and testing, to ensure software quality and maintainability.

Legislative requirements (Assumption): The application should comply with local and international laws governing software development, intellectual property rights, and consumer protection.

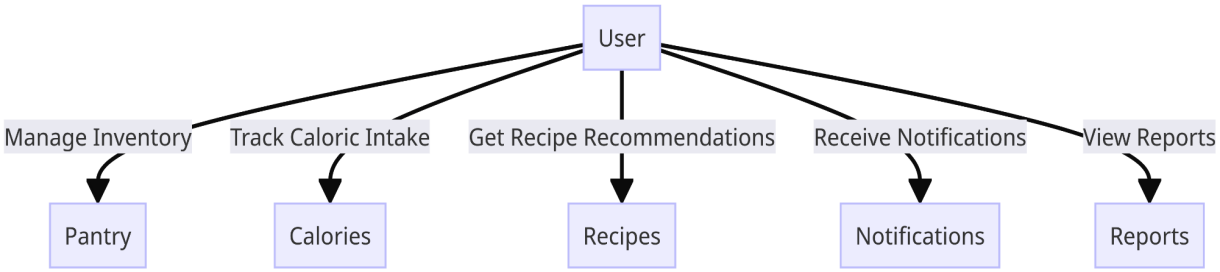
Performance requirements: PantryPal must be responsive and handle high user loads.

Space requirements: The system should work on various devices with minimal storage.

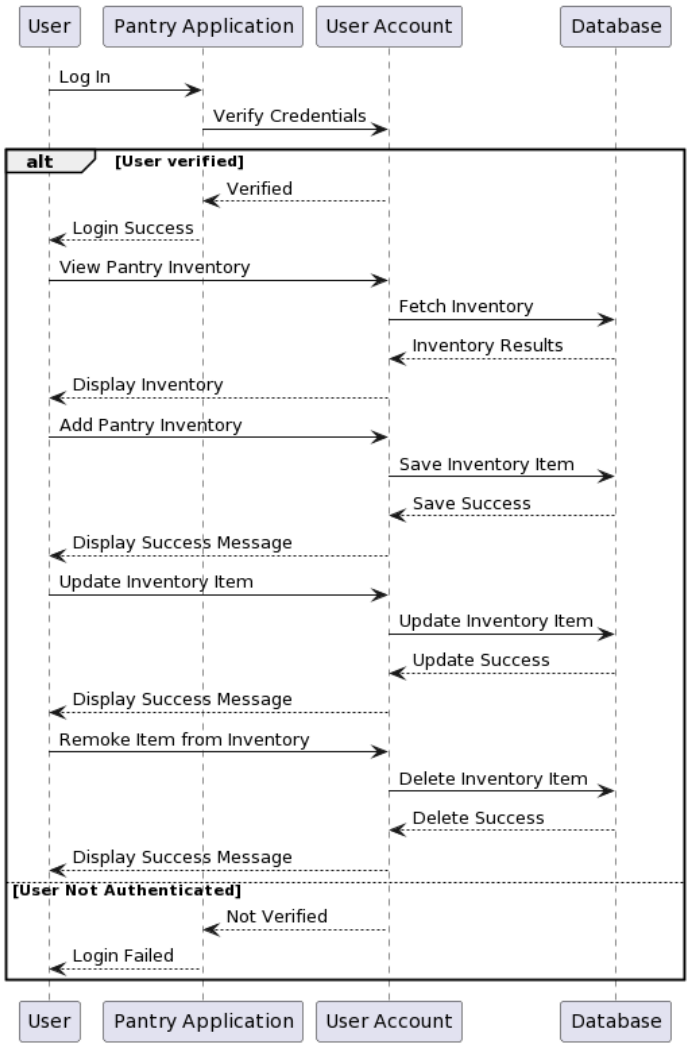
Accounting requirements (Assumption): PantryPal should maintain accurate records of user transactions related to premium features or in-app purchases for financial and auditing purposes.

Safety/security requirements: The app should prioritize user safety and provide necessary warnings or safety measures when needed.

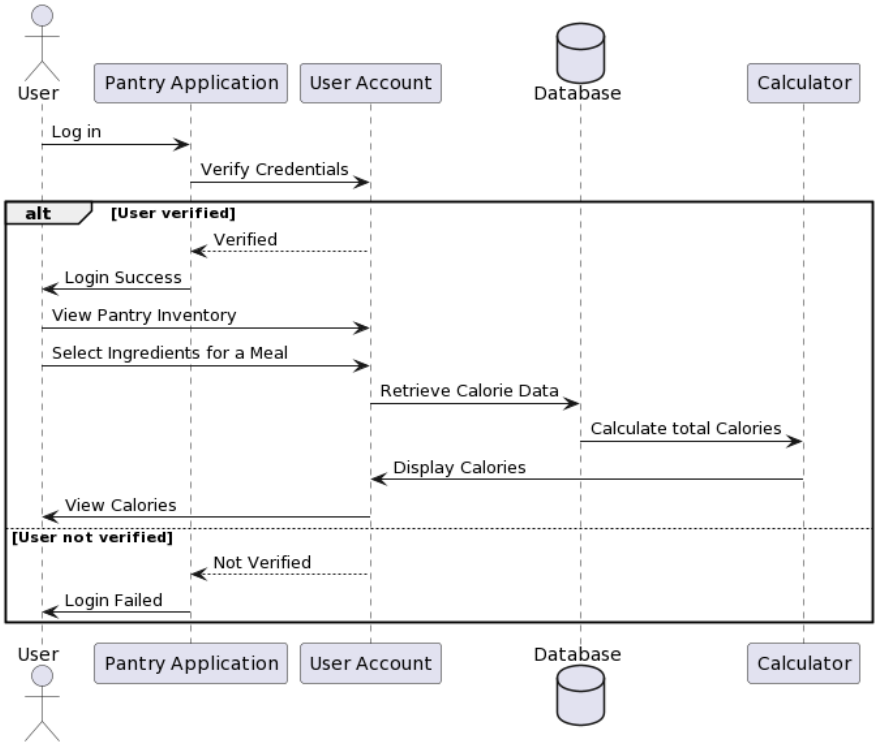
6. **[15 POINTS] Use case diagram – Provide a use case diagram (similar to Figure 5.5)**



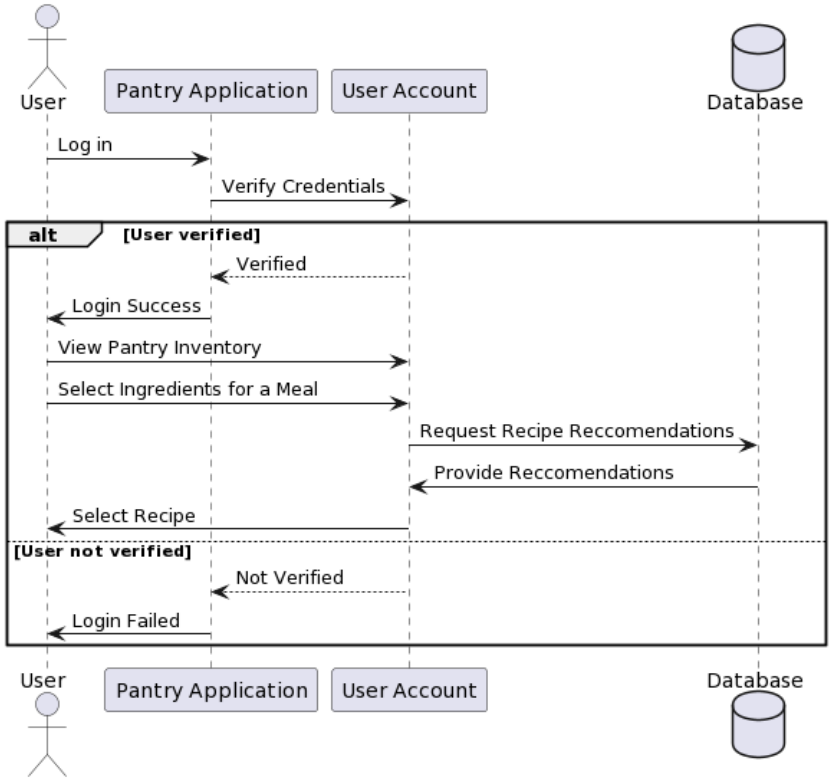
7. **[15 POINTS] Sequence diagram – Provide sequence diagrams (similar to Figure 5.6 and Figure 5.7) for each use case of your project. Please note that there should be an individual sequence diagram for each use case of your project. (Ch 5 and Ch 7)**



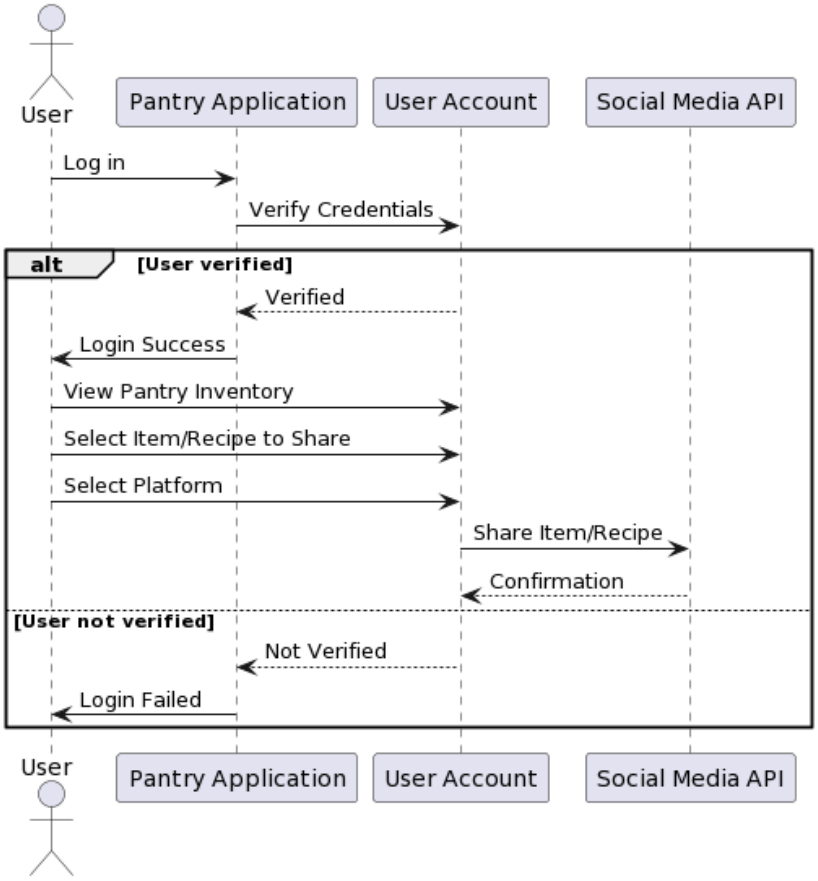
Manage Inventory



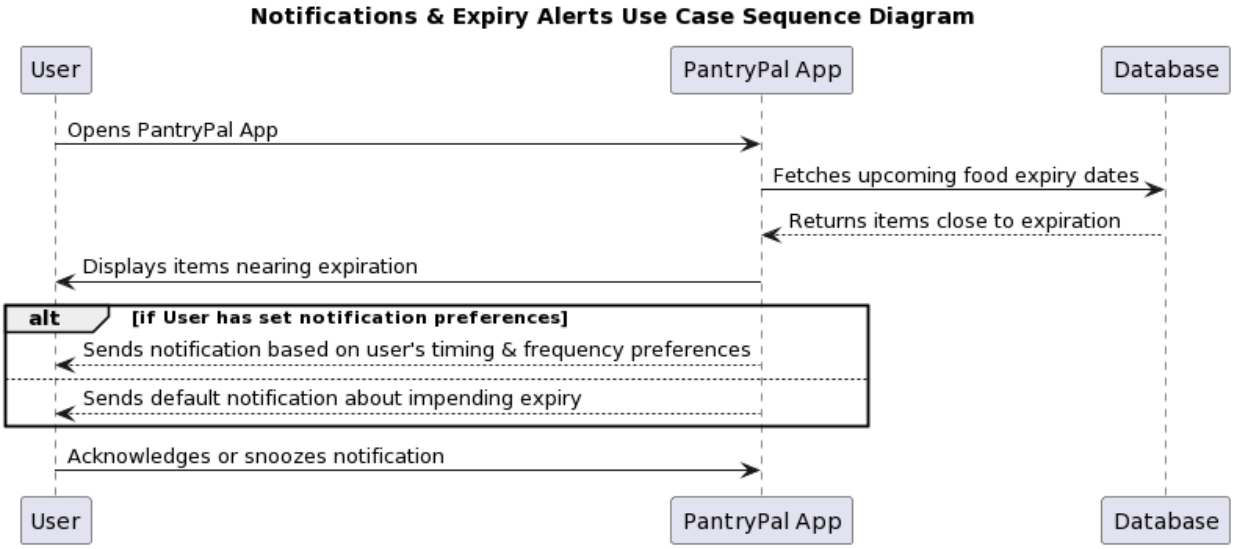
Track Caloric Intake



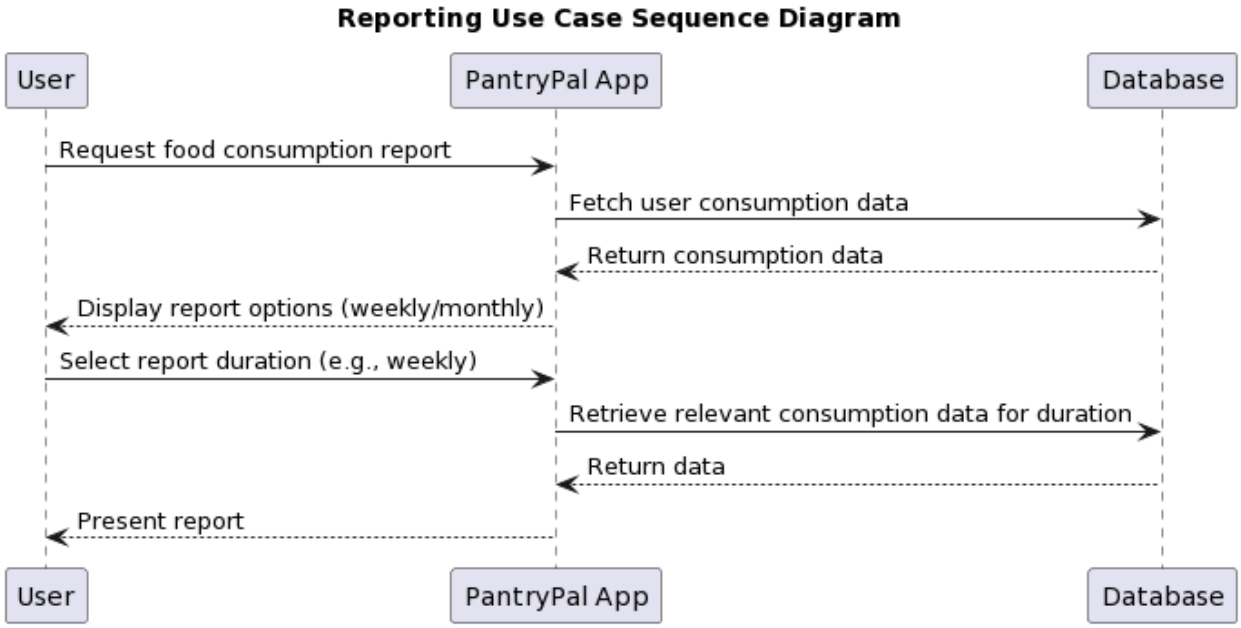
Get Recipe Recommendations



Share on Social Media

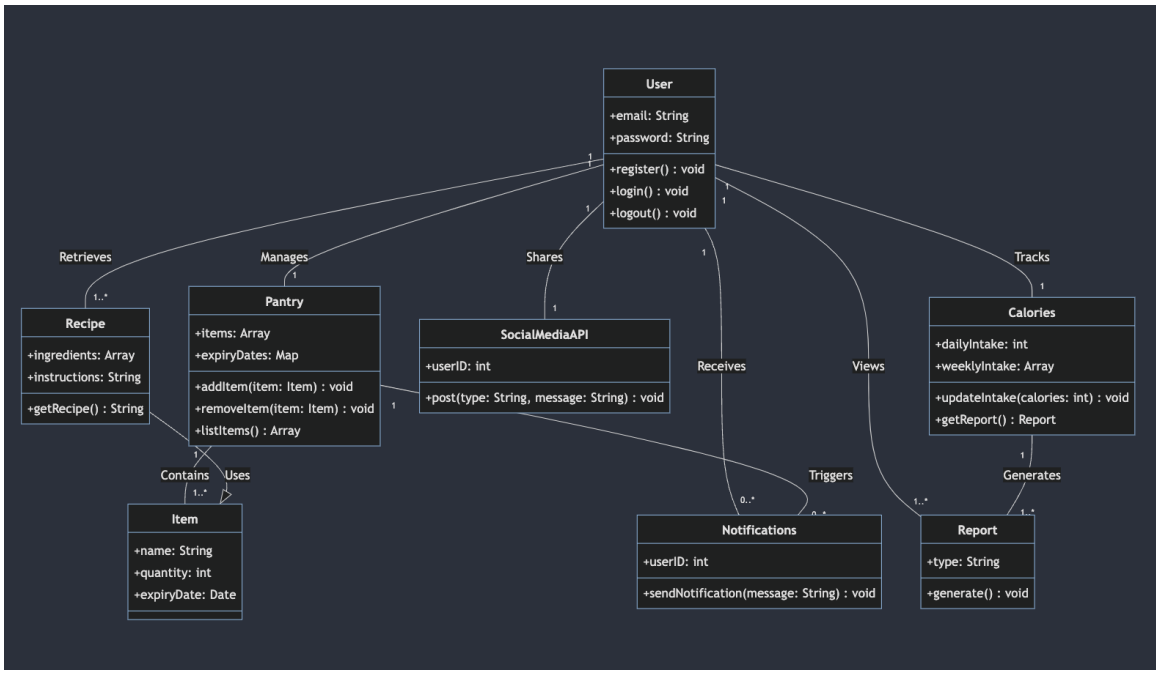


Receive Notifications

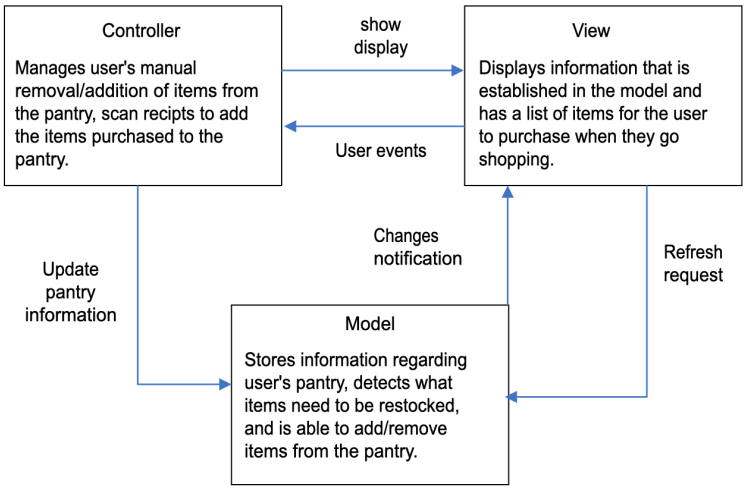


View Reports

8. **[15 POINTS] Class diagram – Provide a class diagram (similar to Figure 5.9) of your project. The class diagram should be unique (only one) and should include all classes of your project. Please make sure to include cardinalities, and relationship types (such as generalization and aggregation) between classes in your class diagram. Also make sure that each class has class name, attributes, and methods named (Ch 5).**



9. **[15 POINTS] Architectural design – Provide an architectural design of your project. Model-View-Controller (MVC) pattern (similar to Figure 6.6)**



3. THIS IS THE START OF PROJECT DELIVERABLE 2

[35 POINTS] Project Scheduling, Cost, Effort and Pricing Estimation, Project duration and staffing: Include a detailed study of project scheduling, cost and pricing estimation for your project. Please include the following for scheduling and estimation studies:

- **3.1. [5 POINTS] Project Scheduling. Make an estimation on the schedule of your project. Please provide start date, end date by giving justifications about your estimation. Also provide the details for:Whether weekends will be counted in your schedule or not. What is the number of working hours per day for the project.**

  Project Schedule Estimation:

  Start Date: November 15, 2023

  End Date: April 12, 2024

  Justification:

  Planning and Requirements Gathering (2 weeks): This phase involves detailing the project plan, gathering software requirements, and creating mockups. (November 15, 2023 - November 29, 2023)

  Design and Prototyping (4 weeks): Developing the app's architecture, UI/UX design, and creating initial prototypes. (November 30, 2023 - December 29, 2023)

  Development (12 weeks): Implementing features, functionalities, database integration, and iterative improvements based on feedback. (January 2, 2024 - March 8, 2024)

  Testing and Quality Assurance (3 weeks): Rigorous testing, bug fixing, performance optimization. (March 9, 2024 - March 29, 2024)

Deployment and Finalization (2 weeks): Preparing for app submission, final checks, and launch preparation. (March 30, 2024 - April 12, 2024)

Work Schedule Details:

Weekends will not be counted

Working Hours per Day: 5 hours

These are just estimations and the actual schedule may vary depending on various factors such as resources, unplanned challenges, changes in project plans, and other things. Regular project reviews and updates will be needed to make sure that the project stays on track.

- **3.2. [15 POINTS] Cost, Effort and Pricing Estimation. Describe in detail which method you use to calculate the estimated cost and in turn the price for your project. Please choose one of the two alternative cost modeling techniques and apply that only:**

  - Function Point (FP)
  - Application composition

We ended up choosing the Function Point Analysis (FPA) method to estimate the cost and effort for the PantryPal project. FPA measures the functionality provided to users based on their external view of the software package. The function point count at the end of requirements analysis is a measure of the functionality the software is expected to deliver.

Here's how we calculate the Function Point (FP) for the PantryPal project:

Individual Application of FPA:

- Number of User Inputs (Simple):
  - Count: 3 (for user registration, inventory input, and caloric input)
- Number of User Outputs (Average):
  - Count: 6 (for displaying inventory, caloric intake, recommendations, notifications, reports, and achievements)
- Number of User Queries (Average):
  - Count: 2 (for user inquiries and recipe queries)
- Number of Data Files and Relational Tables (Complex):
  - Count: 7 (for managing user data, inventory data, recipe data, calorie data, nutritional information data, shopping list data and report data)
- Number of External Interfaces (Average):
  - Count: 2 (for interfacing with social media and data communications)

General Function Point (GFP):

- User Input (Simple): 3 items * 3 (weight) = 9
- User Output (Average): 6 items * 5 (weight) = 30
- User Queries (Average): 2 items * 4 (weight) = 8
- Data Files and Relational Tables (Complex): 7 items * 15 (weight) = 40
- External Interfaces (Average): 2 items * 7 (weight) = 14

So, GFP = 9 + 30 + 8 + 105 + 14 = 166

Processing Complexity Adjustment (PCA):

- The system requires reliable backup and recovery: Score 3 (average)
- Data communications are required: Score 4 (significant)
- There are distributed processing functions: Score 2 (average)
- Performance is critical: Score 4 (significant)
- The system will run in an existing, heavily utilized operational environment: Score 2 (average)
- The system requires online data entry: Score 4 (significant)
- The online data entry requires the input transaction to be built over multiple screens or operations: Score 3 (average)
- The master files are updated online: Score 3 (average)
- The inputs, outputs, files, or inquiries are complex: Score 4 (significant)
- The internal processing is complex: Score 4 (significant)
- The code is designed to be reusable: Score 4 (significant)
- Conversion and installation are included in the design: Score 2 (average)
- The system is designed for multiple installations in different organizations: Score 2 (average)
- The application is designed to facilitate change and ease of use by the user: Score 3 (average)

Total PC scores = 3 + 4 + 2 + 4 + 2 + 4 + 3 + 3 + 4 + 4 + 4 + 2 + 2 + 3 = 44

PCA = 0.65 + 0.01 * 44 = 1.09

Finally, the Function Point (FP) is calculated as follows:

FP = GFP * PCA = 166 * 1.09 = 180.94

This FP value can then be used to estimate the effort, duration, and cost of the project using project historical data or industry averages. For example, if we know the average cost per function point, we can estimate the total cost of the project. Similarly, if we know the average effort per function point, we can estimate the total effort required. This information can then be

used to price the project. Please note that these are just estimations and the actual cost and effort may vary depending on various factors such as resource availability, unforeseen challenges, changes in project scope, etc. Regular project reviews and updates will be necessary to ensure that the project stays on track.

To estimate the cost of the project, we can use the Function Point (FP) value calculated earlier, which is 180.94.

Let's assume the following industry averages for this type of project:

- Average effort per FP: 20 person-hours
- Average cost per person-hour: $40

The total effort required for the project can be estimated as follows: Effort = FP * Average effort per FP = 180.94 * 20 = 3618.8 person-hours

The total cost of the project can be estimated as follows: Cost = Effort * Average cost per person-hour = 3618.8 * $40 = $144,752

Pricing: Now for our **pricing** we want to add a margin to cover additional expenses and profit. The markup percentage can vary depending on various factors such as market conditions, competition, and other overhead costs.

We want to do a 20% markup percentage:

Additional Cost for Markup: $144,752 * 0.20 = $28,950.40

Price = Cost + Additional Cost for Markup = $144,752 + $28,950.40 = $173,702.40

Therefore, considering a 20% markup, the estimated price for the PantryPal project would be approximately $173,702.

- **3.3. [5 POINTS] Estimated cost of hardware products (such as servers, etc.)**
  Dell XPS 13 Laptop [5] (10 Laptops): $6,000
  ASUS- RT AX88U [6] (1 Router): $260.84
  Different phones [7] (1 IOS and 1 Android to test the app): $548
  Total: $6,808.84
- **3.4. [5 POINTS] Estimated cost of software products (such as licensed software, etc.)**
  IDE: Free
  Github (Git): Free
  React Native (Mobile App Framework): Free

ZXing (Barcode scanning API): Free
Receiptful (Receipt scanning API): Free
Testing tools (JUnit): Free
Microsoft Azure (Server) [4]: $315
Total: $315

- **3.5. [5 POINTS] Estimated cost of personnel (number of people to code the end product, training cost after installation)**

Number of People to Code the End Product:

- Project Manager: 1
- Software Developers: 3
- UI/UX Designer: 1
- Quality Assurance Tester: 1

A total of 6 people are needed to code the end product.
Based on the Function Point Analysis (FPA) method and the given data, the Function Point (FP) for the PantryPal project is calculated to be 180.94.

To estimate the cost of the project, we can use this FP value and some industry averages. Let's assume the following:

- Average effort per FP: 20 person-hours
- Average cost per person-hour: $40

The total effort required for the project can be estimated as follows:

Effort = FP * Average effort per FP = 180.94 * 20 = 3618.8 person-hours

The total cost of the project can be estimated as follows:

Cost = Effort * Average cost per person-hour = 3618.8 * $40 = $144,752

So, the estimated cost of the personnel needed for the project is approximately $144,752.

4. [10 POINTS] A test plan for your software: Describe the test plan for testing minimum one unit of your software. As an evidence, write a code for one unit (a method for example) of your software in a programming language of your choice, then use an automated testing tool (such as JUnit for **a Java unit) to test your unit and present results. Clearly define what test case(s) are provided for testing purposes and what results are obtained (Ch 8). Include your test code as additional document in your zip file submitted.**
Test Plan for Pantry Tracking Software - Unit: addItem Method

Objective:

To test the addItem method, which adds items to the pantry in the pantry tracking software, ensuring it handles various scenarios correctly.

Test Case 1: Add Item Successfully

Input:

Item name: "Rice"

Quantity: 5

Expiry date: 2023-12-31

Expected Output: The addItem method should add the item to the pantry.

The pantry should now contain an entry for "Rice" with a quantity of 5 and an expiry date of 2023-12-31. testAddItemSuccess tests the successful addition of an item to the pantry.

Test Case 2: Add Item with Negative Quantity

Input:

Item name: "Cereal"

Quantity: -2 (negative)

Expected Output: The addItem method should handle the negative quantity gracefully. The method should not add the item to the pantry. An error message should be returned, indicating that the quantity must be a positive integer.

testAddItemNegativeQuantity tests that the method correctly handles a negative quantity. The assertThrows method is used to assert that calling addItem with a negative quantity throws an IllegalArgumentException. If the exception is thrown, the test passes.

Test Case 3: Add Item with Empty Name

Input:

Item name: "" (empty string)

Quantity: 3

Expiry date: 2023-10-15

Expected Output: The addItem method should validate the item name and reject the addition. An error message should be returned, indicating that the item name cannot be empty.

testAddItemEmptyName tests that the method correctly handles an empty item name. the assertThrows method is used to assert that calling addItem with an empty name throws an IllegalArgumentException, causing the test to pass.

5. [10 POINTS] Comparison of your work with similar designs. This step requires a thorough search in the field of your project domain. Please cite any references you make.

- **Out of Milk:** Out of Milk is a shopping list and pantry management app. Users can create shopping lists, track the items in their pantry, and share the list with friends and family. It also offers barcode scanning to find products, which will be added to the users shopping list [1].
- **OurGroceries:** OurGroceries focuses on sharing lists with others. It allows users to edit the same lists, and updates lists instantly for all users. Users can add items to the list, and pantry, using barcodes, and pictures [2].
- **Bring! Shopping List:** Bring! Shopping List allows for collaboration on shopping lists in the app. Users can create and share shopping lists with family or friends. It, like PantryPal, has features for managing pantry items and organizing lists by categories [3].
-

6. [10 POINTS] Conclusion - Please make an evaluation of your work, describe any changes that you needed to make (if any), if things have deviated from what you had originally planned for and try to give justification for such changes.

Staying True to Our Core Idea

Throughout the development process, we are proud to say that our core idea for PantryPal has remained largely unchanged. Our commitment to creating a versatile and user-friendly mobile application that simplifies and enhances the user's food management and nutritional tracking experience has been unwavering. This steadfast focus has been instrumental in guiding our project through various phases of development, ensuring that we consistently align with our original vision.

Enhancements in User Experience

One of the most gratifying aspects of our journey has been the improvements we've made in terms of user experience. Recognizing the importance of an intuitive and engaging interface, we have invested considerable effort in refining the app's usability. This includes streamlining the process of adding items to the inventory using the device's camera for barcode scanning and enhancing the receipt scanning feature for ease of product input.

Refinements in Features

Significant enhancements were made to the social feed and pantry management aspects of the app. The social feed, initially a basic feature, has evolved into a more interactive and engaging platform, allowing users to share their healthy eating habits and culinary creations with a broader community. This not only adds a social dimension to PantryPal but also encourages a supportive and motivational environment for users.

The pantry management feature underwent several tweaks to ensure it is more user-friendly and efficient. We realized that the ease of finding and organizing items in one's pantry and fridge is crucial for a seamless user experience. Therefore, we enhanced the categorization and sorting algorithms, making it easier for users to locate and keep track of their food items.

Conclusion

In conclusion, the PantryPal project, while challenging, has been an enriching experience for our team. We have not only managed to stay true to our core idea but have also succeeded in enhancing the application with significant improvements, especially in user experience. These enhancements, particularly in the social feed and pantry management features, have added substantial value to the app, making it not just a tool for food management but a platform for community building and shared culinary experiences. As we present PantryPal, we are confident that it embodies our vision and will make a meaningful impact in the realm of food management and nutritional tracking.

7. [5 POINTS] References: Please include properly cited references in IEEE paper referencing format. Please review the IEEE referencing format document at the URL:

https://ieee-dataport.org/sites/default/files/analysis/27/IEEE%20Citation%20Guidelines.pdf

[1] "The Grocery Shopping List App," Out of Milk, https://www.outofmilk.com/ (accessed Nov. 16, 2023).

[2] "The easiest-to-use grocery list app!," OurGroceries, https://www.ourgroceries.com/overview (accessed Nov. 16, 2023).

[3]"Bring! Shopping List App for iOS & Android," *www.getbring.com*. https://www.getbring.com/en/home (accessed Nov. 16, 2023).

[4]Microsoft, "Cloud Computing Services | Microsoft Azure," *azure.microsoft.com*. https://azure.microsoft.com/en-us

[5]"Inspiron 16-inch Laptop with AMD Ryzen™ 7000 series Processor | Dell USA," *Dell*. https://www.dell.com/en-us/shop/cty/pdp/spd/inspiron-16-5635-laptop (accessed Nov. 16, 2023).

[6]Amazon, "Amazon.com: Online Shopping for Electronics, Apparel, Computers, Books, DVDs & more," *Amazon.com*, 2023. https://www.amazon.com

[7]"Unlimited Plans, Cell Phones, Evolving 4G & 5G Coverage | T-Mobile," *www.t-mobile.com*. https://www.t-mobile.com

It means that your references should be numbered, and these numbers properly cited in your project report.

8. [10 POINTS] Presentation slides. No min/max number of slides enforced. Please make sure that you can complete presentation within 20 (twenty) minutes. Following template could be a good start to prepare your presentations. As each project topic is different, a variety in presentation style is expected and welcome.

Slides link:
https://docs.google.com/presentation/d/1C94USopWz9qoF3NC8zxHORasT_2v_djXtVuYw_ZzBMQ/edit?usp=sharing

10. [5 POINTS] GitHub requirement:
Make sure at least one member of your group commits everything for project deliverable 2 to your GitHub repository, i.e.

- **Your final project deliverable2 report**
- **Unit test code for a sample unit of your project**
- **Implementation code (if you have implemented your project)**
- **Presentation slides**

Still, one member of your team should also submit the required project deliverable 2 materials to eLearning.