Faculty of Engineering and Technology
Department of Electrical and Computer Engineering

Software Project – Phase 4

Prepared By:

Jana Sawalmeh – 1212467
Osaid Nur – 1210733
Salah Dawabsheh – 1210722
Waleed Rimawi – 1211491
Mumen Anbar – 1212297

Supervised By:

Dr. Tasnim Zayet

COMP433 – Section 3

Group work: A-
please refer to your indv. task to get your mark.

Birzeit
January 31, 2026

Table of Contents

1. Project Planning and Management

1.1 Names of Editors/Writers of the Report

Jana Sawalmeh, Osaid Nur

1.2 Business Title

Coffee Beans Shop

1.3 Group Name

Group G5

1.4 Name of Students/Members

- Jana Sawalmeh 1212467
- Osaid Nur 1210733
- Salah Dawabsheh 1210722
- Waleed Rimawi 1211491
- Mumen Anbar 1212297

1.5 Role of Each Member

- Jana Sawalmeh Project Manager
- Osaid Nur Technical Architect Modelling
- Salah Dawabsheh QA Engineer
- Waleed Rimawi Programmer
- Mumen Anbar Secretary

1.6 Project Management Strategy

Our team followed a waterfall methodology for system development, holding Discord meetings and occasional in-person sessions to plan, review progress, assign tasks, and evaluate each project phase. These interactions ensured clear communication, alignment, and effective tracking.

Decision-making was collaborative, with open discussion of differing opinions, resolved through consensus or majority vote. While tasks were assigned based on individual strengths, most work was carried out collectively, including UML diagram design, use-case documentation, and implementation this iterative and cooperative approach enabled the team to adapt quickly to feedback and deliver a well-integrated, cohesive system.

Lucidchart was used tool for designing and maintaining system diagrams for the project.

1.7 Project Manager Report

Working on this phase has been quite an experience for our team. We managed to pull together some solid system modeling and design work, with everyone stepping up to handle different parts of the project. We tried to match people with tasks they were comfortable with or interested in learning more about, which really helped us get things done efficiently.

Contribution by Members:

- **Jana Sawalmeh (Project Manager):** Coordinated the phase plan, kept our deliverables aligned with the requirements, and made sure reviews happened on time. She led the ***User Requirements*** section, owned the UC01 (Browse Coffee Beans) artifacts (specification, activity, and sequence modelling), and led the ***Use Case Diagram***, ***Activity Diagram*** section, and ***System Design Goals***. She also provided structured reviews across the modelling deliverables to keep the overall work consistent.
- **Osaid Nur (Technical Architect & Modelling):** Took the lead on technical consistency across the report and models, and helped translate discussion outcomes into clean, final diagrams and text. He led the ***System Requirements*** and ***Actor Analysis*** sections, produced the ***Detailed Class Diagram***, and owned the UC04 (Place an Order) activity and sequence modelling. He also redrew the overall activity diagram into its final version with an updated layout and clearer visual flow. He led the ***Project Meetings*** appendix and contributed validation across the architecture and design sections.
- **Mumen Anbar (Secretary):** Supported the team by tracking decisions and helping keep our documentation organized and up to date. He helped finalize the wording of the ***User Requirements***, and owned the account registration workflow modelling (activity/sequence). He also contributed to diagram design work (class/component/architecture) and provided reviews to improve clarity and consistency.
- **Salah Dawabsheh (QA Engineer):** Focused on validation and quality checks across the phase deliverables, especially ensuring that our models matched the written requirements. He led the ***Overall Architecture Diagram*** and the ***Deployment Diagram***, owned the UC10 (Track Order Status) activity and sequence modelling, and consistently challenged assumptions during review sessions to reduce inconsistencies and improve the final design.
- **Waleed Rimawi (Programmer):** Led several of the core design deliverables and helped turn the conceptual requirements into structured system models. He led the ***Effort and Time Estimation***, the ***Analysis Class Diagram***, and the ***Component Diagram***, and owned the UC09 (Request Return or Refund) activity and sequence modelling. He also contributed strongly to the finalization of the requirements sections through discussions and refinement.

Challenges: One of the main challenges in this phase was coordination. With five members managing different schedules, it was sometimes difficult to find meeting times that worked for everyone, especially when we needed quick decisions or aligned reviews. In addition, ensuring consistency across all diagrams and written sections required multiple

review rounds to keep notation, terminology, and assumptions unified across the entire model set.

Another challenge was handling iterative refinement. As we received feedback and clarified expectations, we revisited several artefacts to improve completeness and presentation, which occasionally required rework. While this added effort, it ultimately strengthened the quality and coherence of the final deliverables.

Project Performance Review: Overall, the team delivered the required modelling and design outputs for this phase with a cohesive final result. Responsibilities were distributed based on strengths and interests, with clear ownership of specific use cases and diagrams, while still maintaining shared review and refinement to ensure the overall system view remained consistent.

Our regular check-ins and cross-review process were effective in catching inconsistencies early and keeping the work aligned across sections. As a result, the final report reads as a unified project deliverable, and the diagrams and specifications support each other in a clear and traceable way.

1.8 Group Members Report

Jana Sawalmeh:

I see this phase as a success because we ended with a consistent report where the requirements, use cases, and diagrams support each other and tell one clear story about the system. My main contributions were coordination and modelling: I led the ***User Requirements*** section, led/refined the ***Use Case Diagram***, and owned UC01 (Browse Coffee Beans) including its specification and supporting diagrams (activity and sequence). I also led the ***System Design Goals*** section and helped the team by reviewing deliverables to keep naming, notation, and assumptions consistent.

Osaid Nur:

I'm satisfied with the project outcome because the final deliverables feel structured and traceable: the written requirements map cleanly into the models, and the diagrams are easier to follow after the last review cycle. My work focused on technical modelling and report quality: I led the ***System Requirements*** and ***Actor Analysis*** sections, produced the ***Detailed Class Diagram***, and owned UC04 (Place an Order) including the scenario analysis with detailed specification, as well as its activity and sequence modelling. I also redrew the overall activity diagram into the final version with an updated layout and clearer flow, documented the ***Project Meetings*** appendix, and supported validation and some edits across architecture and design sections.

Mumen Anbar:

In my view, the project was successful because we didn't just "finish tasks" — we improved the work until the report and models looked unified and understandable. I contributed by supporting documentation and modelling: I helped finalize wording in the ***User Requirements***, contributed to improving the overall ***Activity Diagram*** section, and owned the account registration workflow (UC14 – Register Account) including its activity and sequence diagrams. I also supported class/component/architecture modelling

through drafting, discussion, and review to improve clarity and consistency.

Salah Dawabsheh:

I consider the project successful because we produced complete modelling and design artefacts and then strengthened them through validation, not just by writing more, but by correcting inconsistencies. My contributions were mainly QA and validation: I owned UC10 (Track Order Status) including its activity and sequence modelling, led the ***Overall Architecture Diagram*** and the ***Deployment Diagram***, and supported the team through review sessions by checking alignment with requirements and highlighting gaps or conflicts.

Waleed Rimawi:

I believe the project was successful because we delivered the required scope and the final system view is consistent across requirements, use cases, and design diagrams. My contributions were mainly in system analysis and design: I led the ***Effort and Time Estimation*** section, led the ***Analysis Class Diagram*** and the ***Component Diagram***, and owned UC09 (Request Return or Refund) including its activity and sequence modelling. I also contributed through requirements and modelling discussions to keep the report coherent.

Note on Task Distribution:

All individual tasks (such as specific use cases and their diagrams) were assigned individually and completed by the respective members. All group tasks — including the **Use Case Diagram**, **Activity Diagram**, **Class Diagrams**, and **System Design Diagrams** — were completed collaboratively in group settings, with one member leading each diagram while others contributed through review, discussion, and editing.

1.9 Group Members Images

Osaid Nur

Salah Dawasheh

Mumen Anbar

Waleed Rimawi

Jana Sawalmeh

2. Requirement Elicitation, Analysis and Modelling

2.1 Requirement statement-Business description

1. Overview

The Coffee Beans Shop is a small business that sells raw coffee beans to customers in the city. It imports raw coffee beans from around the world. The shop offers customers high-quality beans to suit different roasting preferences and flavor tastes. Through the online platform, customers can explore a wide range of coffee bean varieties, view details about origin and flavor profiles, and order home delivery.

2. Business Capacity

- Monthly customers: 200 to 400
- Coffee bean varieties: 20 to 50
- Suppliers: 5 to 8 partners
- Monthly sales: 800 to 1,500 packages
- Daily orders: 20 to 40

3. Services Offered

- Personal accounts with saved delivery information
- Order history and order tracking
- Product browsing with descriptions and details
- Shopping cart and checkout
- Payment options: card, online payment, or cash on delivery
- Fulfillment options: home delivery or in-store pickup
- Promotional offers and discounts for regular customers and bulk orders
- Returns and refunds for incorrect or defective items
- Customer support via chat, email, or phone

4. Service Operation

Customers visit the online shop and look through the available coffee beans. They can search by origin or brand, compare prices, and read about each product. When they find what they like, they add it to their cart and go to checkout.

At checkout, customers choose how they want to receive their order—either delivered to their home or picked up from the shop. They then select how to pay (card, online payment, or cash on delivery) and confirm the order.

Once the order is received, the staff checks it and gets it ready. The warehouse team collects the beans, weighs them, and packs them properly. For delivery orders, the drivers take the packages to customers. For pickup orders, the customer is notified when it is ready.

After customers receive their order, they can share feedback or contact the shop if there is any problem. If something is wrong, they can ask for a return or refund.

5. Employees and Their Roles

- **Manager (1)**: handles daily operations and communicates with suppliers
- **Inventory manager (1)**: manages stock and checks quality
- **Warehouse staff (3)**: prepare and pack orders
- **Delivery staff (3)**: transport packages to customers
- **Sales Assistant (1)**: assist customers in-store
- **Customer support staff (1)**: handle inquiries and feedback

2.2 User Requirements - Modified

Lead: Jana Sawalmeh
Contributors: Osaid Nur (reviewing), Salah Dawabsheh (discussion), Waleed Rimawi (reviewing), Mumen Anbar (finalizing)

UR-01 Coffee Beans Browsing Customers shall be able to browse available coffee bean products, including their details such as name, brand, origin, price, and available quantity.

UR-02 Order Placement Customers shall be able to place purchase orders for selected coffee beans.

UR-03 Inventory Management Inventory managers shall be able to update product stock quantities after receiving new shipments.

UR-04 Flexible Fulfillment Options Customers shall be able to select between in-store pickup or home delivery for their orders.

UR-05 Online Payment Customers shall be able to complete payments for orders using online payment methods (e.g., credit/debit cards, online wallets).

UR-06 Order Status Tracking Customers shall be able to view the current status of their orders (e.g., confirmed, shipped, delivered).

UR-07 Product Return and Refund Requests Customers shall be able to initiate return or refund requests for incorrect, defective, or unsatisfactory items.

UR-08 Delivery Updates Delivery staff shall be able to update delivery status for assigned orders.

UR-09 Support Handling Customer support staff shall be able to respond to customer support requests after review.

UR-10 Reporting and Analytics Managers shall be able to generate reports on sales, inventory levels, and customer feedback for analysis.

2.3 System Requirements - Modified

Lead: Osaid Nur
Contributors: Jana Sawalmeh (reviewing), Salah Dawabsheh (discussion), Waleed Rimawi (finalizing), Mumen Anbar (reviewing)

REQ-1 Product Browsing and Search

- **REQ-01.1** The system shall display product details, including name, brand, origin, price, and available quantity.
- **REQ-01.2** The system shall allow customers to search products by name and brand.
- **REQ-01.3** The system shall allow customers to filter products by price range, origin, and availability status.

REQ-2 Order Placement

- **REQ-02.1** The system shall allow customers to add products with selected quantities to a shopping cart.
- **REQ-02.2** The system shall allow customers to create an order from the shopping cart during checkout.
- **REQ-02.3** The system shall require customers to select one fulfillment method: Home Delivery or In-Store Pickup.
- **REQ-02.4** If Home Delivery is selected, the system shall collect a delivery address containing city, street, building, and phone number.

REQ-3 Inventory Management

- **REQ-03.1** The system shall allow inventory managers to increase or decrease product stock quantities.
- **REQ-03.2** The system shall notify inventory managers when product quantities fall below a predefined threshold.
- **REQ-03.3** The system shall maintain a history of all inventory changes, including user ID, change amount, reason, and timestamp.

REQ-4 Flexible Fulfillment Options

- **REQ-04.1** The system shall require customers to select one fulfillment method before allowing checkout completion.
- **REQ-04.2** when Home Delivery is selected, the system shall collect a delivery address containing city, street, building, and phone number.
- **REQ-04.3** The system shall allow customers to change the selected fulfillment method before order confirmation.

REQ-5 Payment Processing

- **REQ-05.1** The system shall process payments through an external payment gateway.
- **REQ-05.2** The system shall display payment success or failure on the user interface and send an email notification after receiving the gateway response.
- **REQ-05.3** The system shall store transaction reference, amount, timestamp, and payment status.

REQ-6 Order Tracking

- **REQ-06.1** The system shall store each order's status using: Pending, Confirmed, OutForDelivery, Delivered, Cancelled.
- **REQ-06.2** The system shall allow a customer to modify an order only while its status is Pending.
- **REQ-06.3** The system shall record a status history entry for every status change, including previous status, new status, timestamp, and the user ID who performed the update.

REQ-7 Returns Management

- **REQ-07.1** The system shall allow customers to submit return requests with reason and order item reference.
- **REQ-07.2** The system shall allow customer support staff to approve or reject return requests.
- **REQ-07.3** If a return is approved, the system shall update inventory quantities accordingly.

REQ-8 Delivery Management

- **REQ-08.1** The system shall allow delivery staff to update delivery status using: Assigned, PickedUp, InTransit, Delivered, Failed.
- **REQ-08.2** The system shall record the time of each delivery status update.
- **REQ-08.3** The system shall require delivery staff to confirm successful delivery before closing the order.

REQ-9 Customer Support Management

- **REQ-09.1** The system shall allow support staff to record responses to customer inquiries.
- **REQ-09.2** The system shall store support request status using: Open, InProgress, WaitingForCustomer, Resolved.
- **REQ-09.3** The system shall record resolution details, including responsible staff ID and timestamp.

REQ-10 Reporting and Analytics

- **REQ-10.1** The system shall generate sales reports based on date range, product, and revenue.
- **REQ-10.2** The system shall generate inventory reports showing current stock levels and low-stock products.
- **REQ-10.3** The system shall generate operational reports including order volume, delivery success rate, and return rate.

2.4 Scenario Analysis

2.4.1 Scenario 1 - Browsing Coffee Beans

Author: Jana Sawalmeh

Initial Assumptions:
The customer is accessing the Coffee Beans Shop System through a standard web browser with a stable internet connection. The system database contains up-to-date information about available raw coffee beans, including brand, origin, price, and stock status. Browsing, searching, filtering, and cart functionalities are fully operational. The user may or may not be logged into the system.

Normal Flow:
The customer opens the *"Browse Coffee Beans"* section from the main navigation menu. The system displays all available raw coffee beans in a grid layout, showing basic information such as bean name, brand, origin, price, and availability status. The customer selects a specific brand and origin, sets a preferred price range, and enables the *"Available Only"* search filter. The system immediately refreshes the displayed results to match the selected criteria. The customer then sorts the results by price in ascending order and selects a coffee bean product to view its full details. On the product details page, the customer reviews additional information and clicks *"Add to Cart."* The system adds the item to the shopping cart and displays a confirmation message indicating successful addition.

Alternative Flow 1:
Instead of using filters, the customer enters a partial keyword (e.g., *"Ethiopian"*) into the search bar. The system provides auto-complete suggestions based on available coffee beans. The customer selects one of the suggested options, and the system displays matching products. The customer chooses a product from the results, views its detailed information, and clicks *"Add to Cart."* The system confirms the action with a success message.
→ *Successful Output?* Yes

Alternative Flow 2:
The customer browses coffee beans without logging into the system. The system allows full browsing, searching, filtering, and viewing of product details. However, when the customer attempts to click *"Add to Cart,"* the system prompts the user to log in or create an account before proceeding.
→ *Successful Output?* Yes

Alternative Flow 3:
The customer clicks *"Add to Cart"* directly from the product grid without opening the product details page. The system adds the selected coffee bean to the cart immediately and updates the cart icon to reflect the new item count. A brief confirmation message is displayed.
→ *Successful Output?* Yes

Error Flow:
The customer applies multiple filters that result in no matching coffee beans (e.g., a

specific origin combined with a high price range and out-of-stock status). The system displays the message:
"No coffee beans found. Please adjust your filters or clear some options." The customer may remove one or more filters and retry the search. If matching products become available, the customer may continue browsing and add items to the cart. Otherwise, the customer may leave the browsing section or contact customer support.
→ *Successful Output?* No

System State on Completion:
The system records browsing behavior, including search keywords, selected filters, viewed products, and cart actions, for analytics and reporting purposes. If a coffee bean product is viewed or added to the cart, the session state is updated accordingly. The system may also recommend similar or popular coffee beans. Inventory quantities remain unchanged until an order is confirmed during checkout.

2.4.2 Scenario 2 – Placing an Order

Author: Osaid Nur

Initial Assumptions:
The user is logged into their customer account, has previously added items to their shopping cart, and is connected to the internet. The inventory contains all selected items, and the checkout interface is operational.

Normal Flow:
The user accesses their shopping cart to review the selected items, ensuring that the product names, quantities, prices, and total cost are accurate. Once satisfied, the user proceeds to checkout, where they are prompted to select a delivery method. They choose home delivery and enter a valid delivery address. Following this, the user selects a payment method, such as a credit card, and provides the required payment details. The system then verifies that all items are still available in the inventory and processes the payment. If the transaction is successful, a confirmation message is displayed on the screen, a receipt is sent to the user's email address, and the order is marked as "Confirmed".

Alternative Flow 1 – In-Store Pickup Option:
The user opens their shopping cart and reviews the items they previously added. After confirming the cart contents are correct, they proceed to checkout. When prompted to select a delivery method, the user chooses in-store pickup instead of home delivery. They proceed to select a payment method and enter the required payment details. The system checks inventory availability and processes the payment. If the transaction is successful, a confirmation message is shown on-screen, a receipt is sent to the user via email, and the order is recorded with the status "Confirmed".

→ *Successful Output? Yes*

Alternative Flow 2 – Cart Modification Before Checkout:
The user accesses their shopping cart and reviews the selected items. Before proceeding to checkout, they decide to make changes to the cart, such as adding or removing an item, or adjusting the quantity. The system recalculates the total cost and update the quantities and prices. Once the user is satisfied, they continue to the checkout page, where they are prompted to select a delivery method. The user chooses home delivery and provides a valid delivery address. After that, they select a payment method and enter the necessary payment details. The system verifies the availability of the selected items and processes the payment. If successful, a confirmation message is displayed on screen, a receipt is sent to the user's email, and the order is marked as "Confirmed".

→ *Successful Output? Yes*

Error Flow 1 – Item Unavailable During Checkout:
The user opens their shopping cart, verifies the selected items, and proceeds to checkout. After choosing a delivery method (either home delivery or in-store pickup) and entering the required address or pickup details, the user selects a payment method and provides payment information. Before finalizing the payment, the system re-checks the inventory and finds that one or more items in the cart are no longer available. As a result, the

system stops the checkout process and displays a message saying that Some items are no longer available. The user is redirected back to the cart to update it accordingly. No payment is processed and no order is created.

→ *Successful Output? No*

Error Flow 2 – Payment Failure:
The user accesses their shopping cart, reviews the selected products, and continues to checkout. They choose a delivery method—either home delivery with a delivery address or in-store pickup. After that, they select a payment method and enter their payment details. The system checks item availability and confirms all products are in stock. However, during the payment process, the transaction fails due to issues such as insufficient funds, expired card, or invalid information. The system displays an error message that the payment was unsuccessful .The user remains on the payment screen and is given the option to retry the transaction with corrected information or to cancel the order process entirely.

→ *Successful Output? No*

System State on Completion:
If successful, the system updates inventory, creates an order with status "Confirmed", and schedules the delivery or pickup. In case of error, no order is created, and the user is prompted to fix the issue before retrying.

2.4.3 Scenario 3 – Register Account

Author: Mumen Anbar

Initial Assumptions:
The user has a stable connection to the internet and using his desktop browser in order to log into the system, The system is prepared for giving responses and ready to serve the user regerstration's request. The user is using the usual registration form and has not yet registered for the system.

Normal Flow:
The user proceeds to the "Registeration" page and enters some obligatory information: complete name, email, password, and phone number. The system carries out real-time validation (for example, checking if it's an actual email address, if the password is strong enough). After that, the user is allowed to choose interests in the products (e.g. raw coffee beans), but it's optional. After clicking "Register," the system checks if the given email address is in use. If there's no identical account, the user gets registered, and an activation email is sent. The user is either signed in or redirected to the login page depending on the server configurations.

Alternative Flow 1:
On the next page, after entering their profile information, the system will pop up an optional section for choosing product interests. The user selects one or more categories or proceeds without any selection. Afterwards, the process goes on with the account creation like in the normal flow.
→ *Successful Output?* Yes

Error Flow 1:
The user enters an error, perhaps a typo in their email address or a short password. It flags the error, and will prevent them from registering until all mistakes are fixed.
→ *Successful Output?* No, the user must fix errors to continue.

Error Flow 2:
The system recognizes the user if the email address entered is already in use: "The entered email already exsited. Please log in or use some different email." Requests for the use of a different email address or for redirecting to the login page are required.
→ *Successful Output?* No, account not created with original email.

System State on Completion:
Then, upon the success of the registration process, the system generates a log describing the event, sends an email notification to the user to confirm the success of the process, and stores the information of the user in the system securely. Otherwise, it stores no account but logs the event. Finally, the session ends with either account creation or the termination of the session by the user.

2.4.4 Scenario 4 – Request Return or Refund

Author: Waleed Rimawi

Initial Assumptions:
The user is logged into their account and is accessing the system via a desktop browser with a stable internet connection. The item was purchased through the platform and is still within the return window. The item is marked as returnable in the system, and the order history and return interface are fully functional.

Normal Flow:
The user opens the "Order History" section, expands the relevant order, and clicks "Return Item" next to the product. A return form appears with predefined reasons (e.g., "Damaged", "Not as described") and an optional comment field. After submission, the system shows return options (e.g., pickup scheduling) and a refund timeline. The user selects a method and confirms. A return confirmation is shown, and the label is provided to download or/and email. The item's status updates to "Return in Progress" in the order history.

Alternative Flow 1 – Partial Return:
The customer has several items in one order only and wishes to return one of them. They proceed to the details of the order and instead of returning the whole order, they pick the particular item they wish to return. The system is capable of handling a partial return and will create a return label only for the selected item. The remaining items in the order will be unaffected.
→ *Successful Output?* Yes

Alternative Flow 2 – Exchange Request:
"Exchange" becomes the choice because of a defect.Rather than a refund, the system replaces the same merchandise.The return label is generated, and the new unit that is not damaged is sent when the merchandise arrives.
→ *Successful Output?* Yes

Error Flow:
The user tries to return an item that falls outside the return timeframe or if the item does not belong to returnable categories (for example, perishables). On choosing an item and trying to initiate a return, the system prevents further processing and shows an error message with the reason. The user gets an option to click on a link to reach support staff or read return terms.
→ *Successful Output?* No

System State on Completion:
The system logs the return request along with key metadata. If accepted, the item's status updates to "Return in Progress" and is linked to a return case. Notifications are sent, and backend processes are queued for shipping, refunds, and inventory. Refunds are issued only after the item is received and verified. Failed returns are logged for analytics or support. No inventory or financial updates occur until return completion.

2.4.5 Scenario 5 – Tracking Order Status

Author: Salah Dawabsheh

Initial Assumptions:
The customer is logged into their account and has an active session. The customer has at least one existing order stored in the system, and each order has a valid order ID / reference and an associated status (e.g., "Confirmed", "Processing", "Shipped", "Out for Delivery", "Delivered", "Cancelled"). The order tracking functionality is operational, and the system can retrieve the latest order status and available tracking updates (including ETA / pickup readiness when applicable).

Normal Flow:
The customer requests to view their order history. The system displays the customer's previous orders. The customer selects a specific order to track. The system retrieves the latest status information for the selected order and displays the current order status along with the latest available tracking update. If available, the system also displays the estimated delivery date or pickup readiness information. The customer reviews the order status and completes the tracking action.

Alternative Flow 1 – Tracking Using Order ID / Reference:
Instead of selecting from the order history, the customer provides an order ID / order reference (from a previous confirmation or receipt). The system validates the order ID format and verifies that the order belongs to the customer. The system retrieves the latest status information for the order and displays the current order status and tracking updates.
→ Successful Output? Yes

Alternative Flow 2 – Tracking Another Order:
After viewing the status of one order, the customer selects another order to track. The system retrieves the latest status information for the newly selected order and displays the updated order status and tracking update.
→ Successful Output? Yes

Error Flow 1 – Order Not Found / Not Authorized:
The customer selects an order or provides an order ID / reference. The system cannot find a matching order for this customer (invalid order ID or the order belongs to another user). The system displays an error message stating "Order not found or you do not have access to this order." No order details are displayed, and the customer may retry or select a different order.
→ Successful Output? No

Error Flow 2 – Tracking Service Unavailable:
The customer selects an order to track. The system fails to retrieve the order status due to a service unavailability or timeout. The system displays an error message stating "Unable to retrieve order information at this time. Please try again later." No order details are displayed, and the customer may retry later.
→ Successful Output? No

System State on Completion:
If successful, the customer views the current order status and the latest tracking update, and no order data is modified. In case of error, no sensitive order data is revealed, and the customer is prompted to retry or choose a valid order.

2.5 Task 2.3: Effort and Time Estimation

Lead: Waleed Rimawi
Contributors: Jana Sawalmeh (reviewing), Salah Dawabsheh (discussion), Osaid Nur (finalizing), Mumen Anbar (reviewing)

2.5.1 Introduction

Purpose This section estimates the total effort, development cost, and schedule time needed to build the Coffee Beans Shop System based on the User and System Requirements defined in Tasks 2.1 and 2.2.

Scope The estimation covers analysis, development, testing, and deployment. Effort is measured in person-weeks (pw), where 1 pw = full-time work by 1 person for 1 week. The schedule includes a 30% buffer to account for delays, communication, and holidays.

Methodology The estimation method includes:

- Assigning estimated effort per user requirement in person-weeks.
- Calculating total and average effort.
- Applying a 30% schedule buffer.
- Estimating cost at $250 per person-week.
- Calculating minimum and maximum cost based on 10–30% profit margin.

2.5.2 Effort Estimation Table

UR	Estimated Effort (pw)	Estimated Developers	Total Effort (pw)
UR-01	2	2	$= 2 \times 2 = 4$
UR-02	3	1	$= 3 \times 1 = 3$
UR-03	2	3	$= 2 \times 3 = 6$
UR-04	1	4	$= 1 \times 4 = 4$
UR-05	1	2	$= 1 \times 2 = 2$
UR-06	1	1	$= 1 \times 1 = 1$
UR-07	1	1	$= 1 \times 1 = 1$
UR-08	1	1	$= 1 \times 1 = 1$
UR-09	1	1	$= 1 \times 1 = 1$
UR-10	1	1	$= 1 \times 1 = 1$
Total Effort	–	–	24

Table 1: Effort estimation per user requirement for the Coffee Beans Shop System

2.5.3 Average Developers Required

$$\text{Average Developers} = \frac{2+1+3+4+2+1+1+1+1+1}{10} = 1.7 \text{ developers}$$

2.5.4 Schedule Time (with 30% buffer)

$$\text{Buffered Effort} = 24 \times 1.30 = 31.2 \text{ pw} \approx 32 \text{ person-weeks}$$

2.5.5 Cost Estimation

- Cost per person-week: $250
- Base Cost: $250 × 32 pw = $8,000
- Minimum Cost (10% profit): $8,000 × 1.10 = $8,800
- Maximum Cost (30% profit): $8,000 × 1.30 = $10,400

2.5.6 Summary

The estimated development effort to build the Coffee Beans Shop System is 24 person-weeks, with a buffered timeline of approximately 32 person-weeks. At a standard rate of $250 per person-week, the projected total cost is:

- Minimum: $8,800 (with 10% profit)
- Maximum: $10,400 (with 30% profit)

2.6 Actor Analysis

Lead: Osaid Nur
Contributors: Jana Sawalmeh (reviewing), Mumen Anbar (reviewing) , Salah Dawabsheh (discussion), Waleed Rimawi (discussion)

Actor Descriptions

- **Customer**
  Represents the primary user of the Coffee Beans Shop System. Can browse coffee beans, search products, place orders online or in-store, make payments, track deliveries, initiate returns or refunds, and contact customer support.
- **Manager/Owner**
  Has administrative privileges to monitor users, review order history and coffee bean inventory movements, and view management reports and audit logs to oversee the business.
- **Sales Assistant**
  Responsible for processing in-store orders and payments for coffee beans, assisting customers during their purchases at physical store locations, and handling returns at the counter.
- **Inventory Manager**
  Keeps coffee bean stock up to date, updates available bean types and origins, and ensures inventory data is correct for orders and tracking.
- **Delivery Staff**
  Plans delivery routes, updates delivery/shipping status, and handles return pickups or drop-offs for coffee products.
- **Customer Support Assistants**
  Assists customers with order issues, account inquiries, returns, refunds, and general support requests, and sends technical issues to the right team when needed.
- **Payment Gateway (e.g., Bank/PayPal)**
  An external service responsible for processing online transactions securely for coffee bean purchases, validating payment information, and confirming payment success or failure.
- **Email System**
  Sends transactional messages such as order confirmations, delivery updates, password recovery emails, and other system-generated notifications to users.

2.7 task: Use Case Diagram - Modified

Lead: Jana Sawalmeh
Contributors: Mumen Anbar (reviewing), Osaid Nur (reviewing) , Waleed Rimawi (discussion), Salah Dawabsheh (discussion)

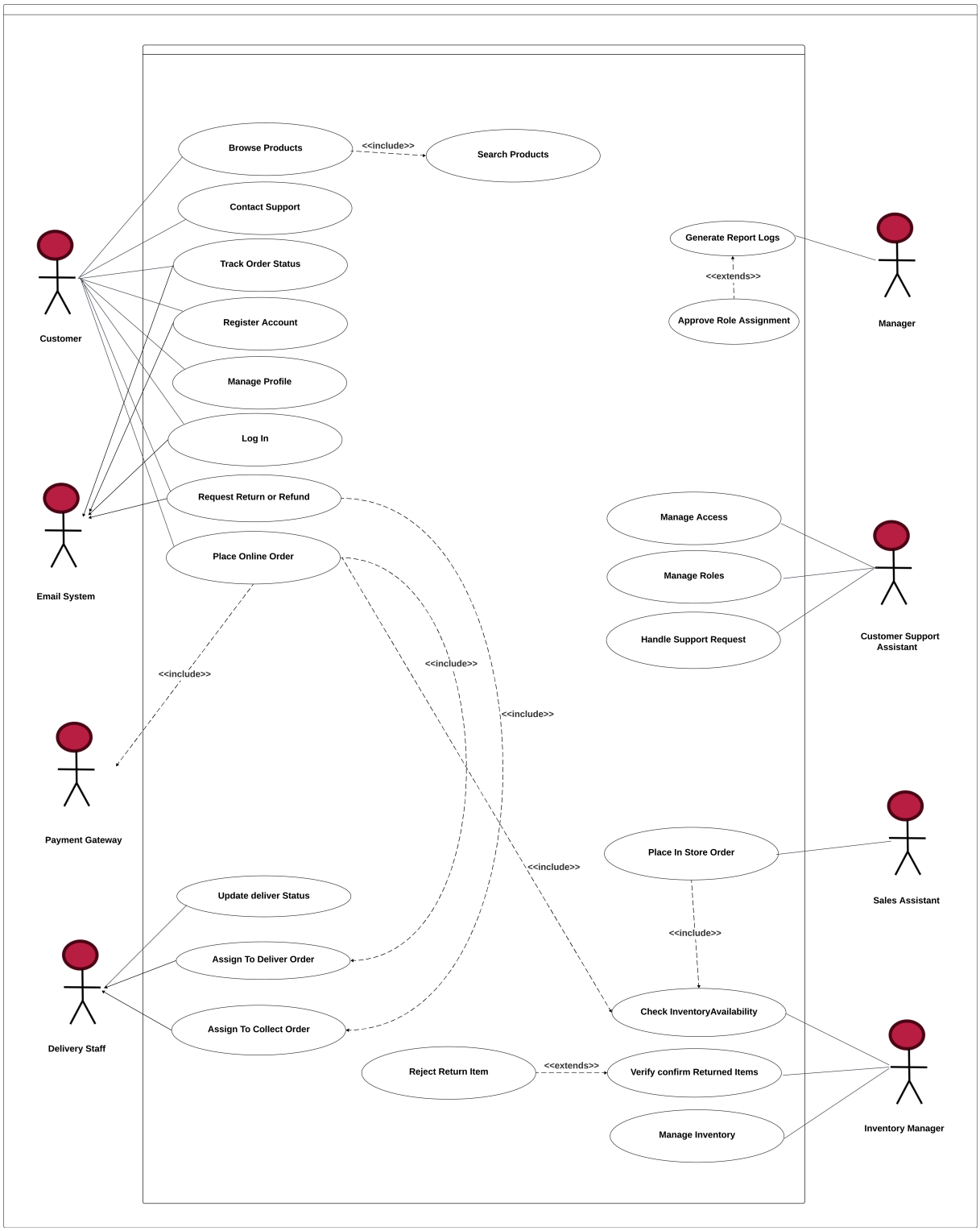


Figure 1: Use Case Diagram for Coffee Beans Shop System

2.8 Task: Use Case Specifications

2.8.1 UC01 – Browse Coffee Beans

Author: Jana Sawalmeh

Use Case Title	Browse Coffee Beans
Description	This use case allows customers to browse available raw coffee beans using categories or search keywords. Customers can refine results using filters such as price range, brand, origin, or availability. The system provides real-time updates to the product listings and supports viewing detailed information for each coffee bean.
Actors	• **Primary Actor:** Customer • **Secondary Actors:** Inventory Manager
Data	• Coffee bean categories • Brands and origins • Search keywords • Filter values (price range, availability) • Product metadata (name, brand, origin, price, stock status)
Stimulus / Trigger	The customer navigates to the Browse Coffee Beans section or types a keyword in the search bar.
Pre-Conditions	1. The customer is authenticated or using guest access. 2. The coffee beans catalog is populated by the inventory manager and is accessible. 3. The filtering and search user interface is fully initialized and responsive.
Workflow	1. Customer accesses the Browse Coffee Beans page. 2. System displays all available coffee beans in a default layout. 3. Customer applies search filters (e.g., price range, brand, availability). 4. System updates results based on the selected filters. 5. Customer uses sorting or the search bar to further refine results. 6. System displays the refined list and highlights matched keywords. 7. Customer clicks on a coffee bean product to view full details. 8. System logs the product view for analytics purposes.

	Alternative Flow 1 – Keyword Search 1. Customer enters a partial keyword in the search filter bar. 2. System suggests matching coffee beans using autocomplete. 3. Customer selects a suggestion, triggering a filtered results list. 4. Customer clicks on one of the displayed coffee beans. 5. The product detail page opens; the customer may choose to add the item to the cart. 6. If logged in, the item is added and a confirmation message is shown. **Alternative Flow 2 – Guest Browsing** 1. Customer accesses the system without logging in. 2. System allows full coffee bean browsing but disables "Add to Cart" and checkout options. 3. Customer may still use filters, search, and view product details. **Alternative Flow 3 – Add Without Viewing Details** 1. Customer browses the product grid and finds a coffee bean of interest. 2. Without opening the product details page, the customer clicks "Add to Cart." 3. If logged in, the item is added to the cart. 4. A success message is shown, and the cart icon is updated. **Error Flow – EF1: No Matching Results** 1. Customer applies filters or search criteria that yield no matching coffee beans. 2. System displays the message: *"No coffee beans found. Try adjusting your search filters or clearing some options."* 3. Customer clears filters or modifies the search keyword. 4. If new results appear, the normal flow resumes; otherwise, the customer may cancel or retry.
Post-Conditions / Response	• Matching coffee beans are successfully displayed. • Product detail pages are accessible and load without error. • If logged in, coffee beans can be added to the cart. • All browsing, filtering, and viewing actions are logged for analytics. • If no results are found, the system provides reset or suggestion options.

Comments	• Logged-in customers may receive recommendations based on browsing history. • The system should ensure fast response times for search and filtering. • Add-to-cart from the list improves efficiency for frequent customers. • Future improvement: filtering by roast level or flavor profile.

2.8.2 UC04 – Place an Order

Author: Osaid Nur

Use Case Title	Place an Online Order
Description	A customer places an online order by reviewing their cart, selecting delivery and payment methods, verifying item availability, and completing the transaction. The system checks both item stock and payment details before confirming the order. Upon success, a confirmation is displayed, the delivery or pickup is scheduled based on the selected option, and a receipt is sent to the customer via email.
Actors	• **Primary Actor:** Customer • **Secondary Actors:** Inventory System, Payment Gateway System.
Data	• List of selected items (Coffee Beans) • Cart total price • Delivery information • Payment method and details • Order confirmation status
Stimulus / Trigger	The customer selects " Place Order" from the cart interface.
Pre-Conditions	1. The user is authenticated and has an active session. 2. The user's shopping cart contains at least one valid item with up-to-date pricing. 3. Delivery options (e.g., home delivery, in-store pickup) are currently available based on the user's location. 4. The payment service is operational and can accept transactions. 5. The inventory system is online, and selected items are marked as "in stock."

Workflow	1. Customer opens their cart and reviews the selected items. 2. System displays item details including name, quantity, price, and total cost. 3. Customer selects home delivery as the delivery method. 4. Customer enters the required delivery address. 5. Customer selects a payment method and provides payment details. 6. System checks the availability of all items in the inventory. 7. If all items are available, the system processes the payment via the payment gateway. 8. System confirms the payment and generates an order record. 9. System schedules the delivery to the provided address. 10. System displays an on-screen confirmation and sends a receipt via email. **Alt Flow 1 – In-Store Pickup** 1. Customer opens cart and reviews selected items. 2. System displays item details including name, quantity, price, and total cost. 3. Customer selects in-store pickup as delivery method. 4. Customer selects payment method and provides payment details. 5. System checks availability of all items in inventory. 6. If all items are available, system processes payment via payment gateway. 7. System confirms payment and generates order record. 8. System schedules the order. 9. System displays on-screen confirmation and sends receipt via email. **Alt Flow 2 – Modifying Cart** 1. Customer opens cart and reviews selected items. 2. Before proceeding, customer modifies cart (changes quantity, adds or removes items). 3. System updates cart and recalculates total price. 4. Customer reviews updated item list. 5. System displays item details including name, quantity, price, and total cost. 6. Customer selects home delivery as delivery method. 7. Customer enters required delivery address. 8. Customer selects payment method and provides payment details. 9. System checks availability of all items in inventory. 10. If all items are available, system processes payment via payment gateway. 11. System confirms payment and generates order record. 12. System schedules delivery and displays confirmation with receipt via email.

Error Flows	**EF1 – Payment Failure** 1. Customer opens cart and reviews selected items. 2. System displays item details including name, quantity, price, and total cost. 3. Customer selects home delivery as delivery method. 4. Customer enters required delivery address. 5. Customer selects payment method and provides payment details. 6. System attempts to process payment using provided method. 7. Payment fails due to insufficient funds, expired card, or timeout. 8. System displays: *"Payment unsuccessful. Please verify your card or try a different method."* 9. Customer is given option to retry or cancel order. **EF2 – Item Unavailable During Checkout** 1. Customer opens cart and reviews selected items. 2. System displays item details including name, quantity, price, and total cost. 3. Customer selects home delivery as delivery method. 4. Customer enters required delivery address. 5. Customer selects payment method and provides payment details. 6. System checks inventory and detects one or more items are no longer available. 7. System stops order process and displays: *"Some items are no longer available. Please adjust your cart."* 8. Customer is redirected to cart to modify order.
Post-Conditions / Response	• **If successful:** – Order is created with status "Confirmed" – Inventory quantities have been updated accordingly. – A successful payment transaction is logged and associated with the order. – The system is ready to proceed with fulfillment or allow the user to track their order. • **If unsuccessful:** – No order has been created or stored in the system. – Inventory remains unchanged. – The user remains at the checkout or cart state, awaiting a corrective action. – The system is prepared to handle a retry attempt or a modified order.

Comments	• Order cannot be modified after confirmation. • Payment and stock must be validated before final confirmation. • Delivery and payment options depend on system configuration.

2.8.3 UC14 – Register Account

Author: Mumen Anbar

Use Case Title	Register Account
Description	A new customer opens an account with the help of the browser. A common form is given for account opening, the entries are checked for the given criteria (for example, the username must be different and the password must satisfy certain requirements), and based on the criteria, an account can be made or the submission can be rejected with proper reasons.
Actors	• **Primary Actor:** Customer • **Secondary Actors:** Email System
Data	• Full name • Email address • Username • Password • Phone number
Stimulus / Trigger	The user chooses "Register".
Pre-Conditions	1. The user has no registered accounts with the desired email. 2. The system is able to give responses to the registration request. 3. The Registeration interface is functional. 4. Internet connection is stable.

Workflow	1. System shows need to fill fields such as: Full name, email, username, and password. 2. User provides correct data in all fields. 3. Different validations are applied like username and email uniqueness, password strength, and email format. 4. User submits registration request. 5. A new user profile is created and safely stored. 6. A confirmation message is displayed. 7. A Confirmation email will be sent to the user for confirmation. 8. According to the user's choice, either login or redirect him to the page for login. **Alternative Flow 1** 1. Following step 2 in the normal flow, the system displays an Optional section for interest selection. 2. The user may pass on this step. 3. The process now continues from step 3 in the usual flow process. **Error Flows:** **EF1 – Invalid Input:** 1. Enters incorrect information (e.g., short password or) (malformed email 2. System identifies incorrect fields and displays appropriate validation messages, such as "Password must be at least 8 characters" 3. The process of registration is hindered pending resolution of issues.
	EF2 – Email Already Exists: 1. User submits a form with an already existing email associated with existing account. 2. System displays an appropriate message (for example, "An account with this email already exists.") that the email address has already been linked through an existing account. 3. The user will be asked to log in with the existing account or provide a different email address to sign up.

Post-Conditions / Response	• **If successful:** – A new user profile is created – When the user boots – A confirmation email has been sent – The user is logged in or redirected • **If unsuccessful:** – No data is stored – Error messages are displayed – Efforts were noted
Comments	• The system should enforce robust validation to secure an account. • There should be no duplicate accounts for the same email. • System behavior after registration (login/redirect) is configuration-dependent

2.8.4 UC09 – Request Return or Refund

Author: Waleed Rimawi

Use Case Title	Returning a Delivered Item
Description	Use case: Customers can request the return or refund of goods ordered through the system using this use case. The eligibility of goods to be returned (based on time, product type, etc.), complete or partial returns, pick-ups, deliveries, and return mail are managed by the system, after which a refund or exchange can be made.
Actors	**Primary Actor:** Customer **Secondary Actors:** Email System, Delivery Staff, Inventory Manager
Data	• Order History • Reason for return (e.g. damaged, not as described) • Optional comment • Pickup/drop-off selection • Refund Timeline • Return label (Download/Email)
Stimulus / Trigger	The customer selects “Return Item” for a previously purchased item from their order history.
Pre-Conditions	1. A customer is logged into their account. 2. It is acquired through the system. 3. The item is still within the return period. 4. The merchandise can be returned. 5. The return interface and email service are available.
Workflow	1. Customer navigates to ”Order History.” 2. Expands an order and clicks “Return Item.” 3. System displays returned form along with reasons and comments. 4. The customer submits the form. 5. System eligibility verification and return options display. 6. Customer picks the mode (pickup/drop-off). 7. System generates and sends return label (email/download). 8. Order status update to "Return in Progress." 9. The notifications will be sent to the "Delivery Staff" as well as "Inventory Manager".

Alternative Flow 1 – Partial Return	1. Customer picks only one item that was ordered with multiple items. 2. System processes partial return, adjusts refund, and generates a return label.
Alternative Flow 2 – Exchange Instead of Refund	1. Customer selects "Exchange" due to a defect. 2. System initiates replacement process. 3. Return label is issued. 4. Replacement item is sent after arrival of the return.
Error Flow – EF1: Invalid Return Attempt	1. Customer picks an item outside the return window or an item marked non-returnable. 2. System doesn't allow to return, shows response: This item cannot be returned. Please see our return policy or contact support.
Post-Conditions	**If successful:** • The return request is recorded • Status changed to "Return in Progress" • Case allocated to Delivery Staff & Inventory Manager • Notification sent • Refund or replacement after verification **If unsuccessful:** • Return request not created • Failures reported • User is notified
Comments	• Partial return options can benefit customers • Return status updates build trust between business and consumer. • Future: ability to upload pictures of damaged products

2.8.5 UC10 – Track Order Status

Author: Salah Dawabeheh

Use Case Title	Track Order Status
Description	A customer tracks the status of a previously placed order. The system retrieves and displays the current order status (e.g., "Confirmed", "Processing", "Shipped", "Out for Delivery", "Delivered", "Cancelled") along with the latest available tracking update. The customer can access tracking through their order history or by using an order ID / reference.
Actors	• **Primary Actor:** Customer • **Secondary Actors:** Order Management System, Delivery / Pickup Service (if applicable)
Data	• Customer account session • Order list (order history) • Order ID / order reference • Current order status • Delivery infof • Tracking update / last status update time • Estimated delivery date / pickup readiness
Stimulus / Trigger	The customer selects "Track Order Status" / selects an order from their order history, or enters an order ID to view its status.
Pre-Conditions	1. Customer is authenticated and has an active session. 2. Customer has at least one existing order stored in the system. 3. The order tracking functionality is operational and can retrieve the latest order status. 4. Order records are available in the database.

Workflow	1. Customer requests to view their order history. 2. System displays the customer's previous orders. 3. Customer selects a specific order to track. 4. System retrieves the latest status information for the selected order from the Order Management System. 5. System displays the current order status and the latest available tracking update (and ETA / pickup readiness if available). **Alternative Flow 1 – Tracking Using Order ID / Reference** 1. Customer provides an order ID / order reference (from a previous confirmation or receipt). 2. System validates the order ID format. 3. System searches for the order and verifies it belongs to the customer. 4. System retrieves the latest status for the order. 5. System displays the current order status and tracking updates. **Alternative Flow 2 – Tracking Another Order** 1. After viewing one order status, the customer selects another order to track. 2. System retrieves the latest status information for the newly selected order. 3. System displays the updated order status and tracking update. **Error Flows:** **EF1 – Order Not Found / Not Authorized:** 1. Customer selects an order or provides an order ID. 2. System cannot find a matching order for this customer (invalid ID or belongs to another user). 3. System displays: "Order not found or you do not have access to this order." **EF2 – Tracking Service Unavailable:** 1. Customer selects an order to track. 2. System fails to retrieve order status due to timeout / database connection issue / service unavailability. 3. System displays: "Unable to retrieve order information at this time. Please try again later."

Post-Conditions / Response	• If successful: – Customer views the current order status and latest tracking update. – No order data is modified. • If unsuccessful: – An appropriate error message is displayed. – No sensitive order data is revealed.
Comments	• Only the owner of the order can track its status. • Order status is read-only in this use case. • Tracking details (e.g., ETA / pickup readiness) depend on the availability of fulfillment updates.

2.9 Activity Diagram - Modified

Lead: Jana Sawalmeh
Contributors: Osaid Nur (modelling), Salah Dawabsheh (validation), Mumen Anbar (design), Waleed Rimawi (reviewing)

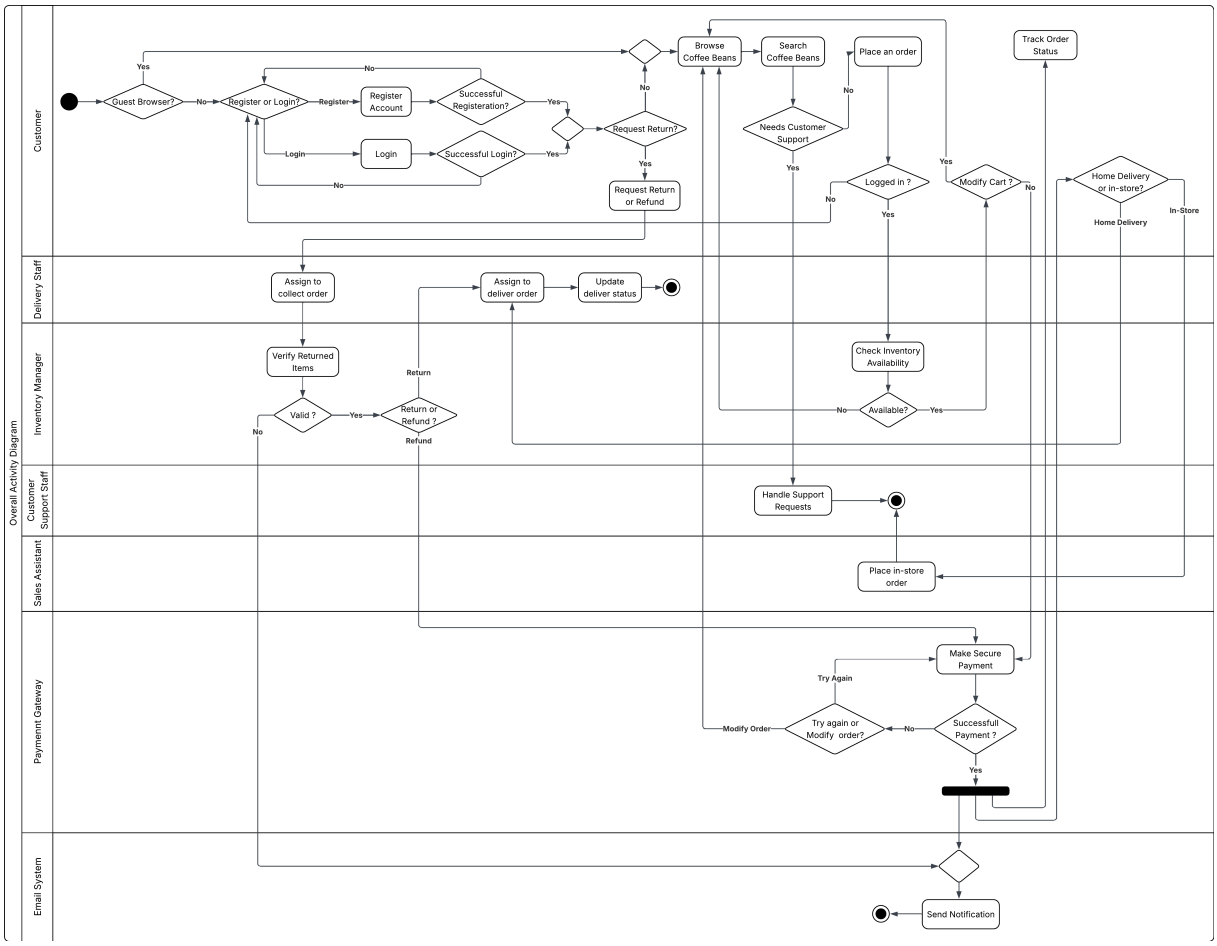


Figure 2: Overall Activity Diagram

2.10 Instance Activity diagrams

2.10.1 UC01 – Browse Coffee Beans (Jana Sawalmeh)

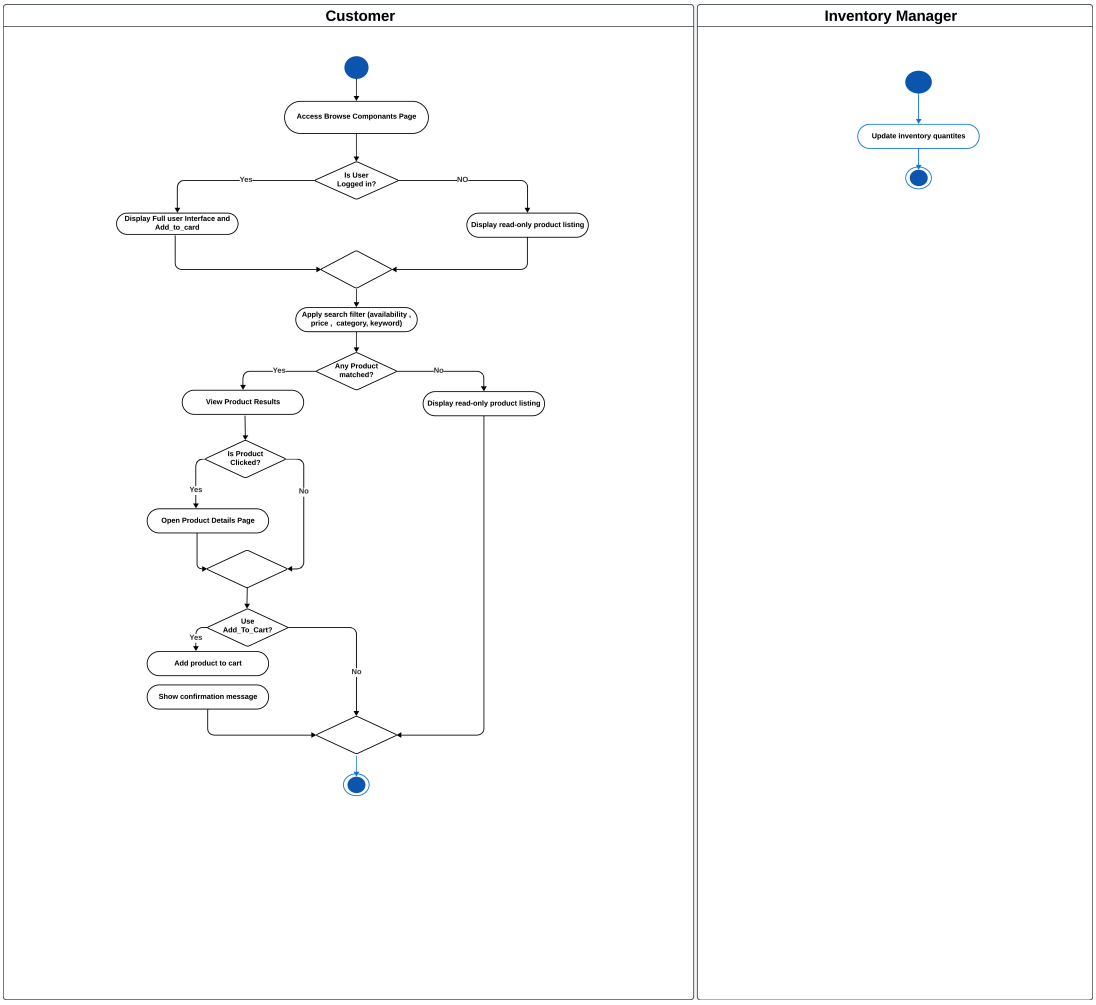


Figure 3: Activity Diagram for UC01 – Browse Coffee Beans

2.10.2 UC04 – Place an Order (Osaid Nur)

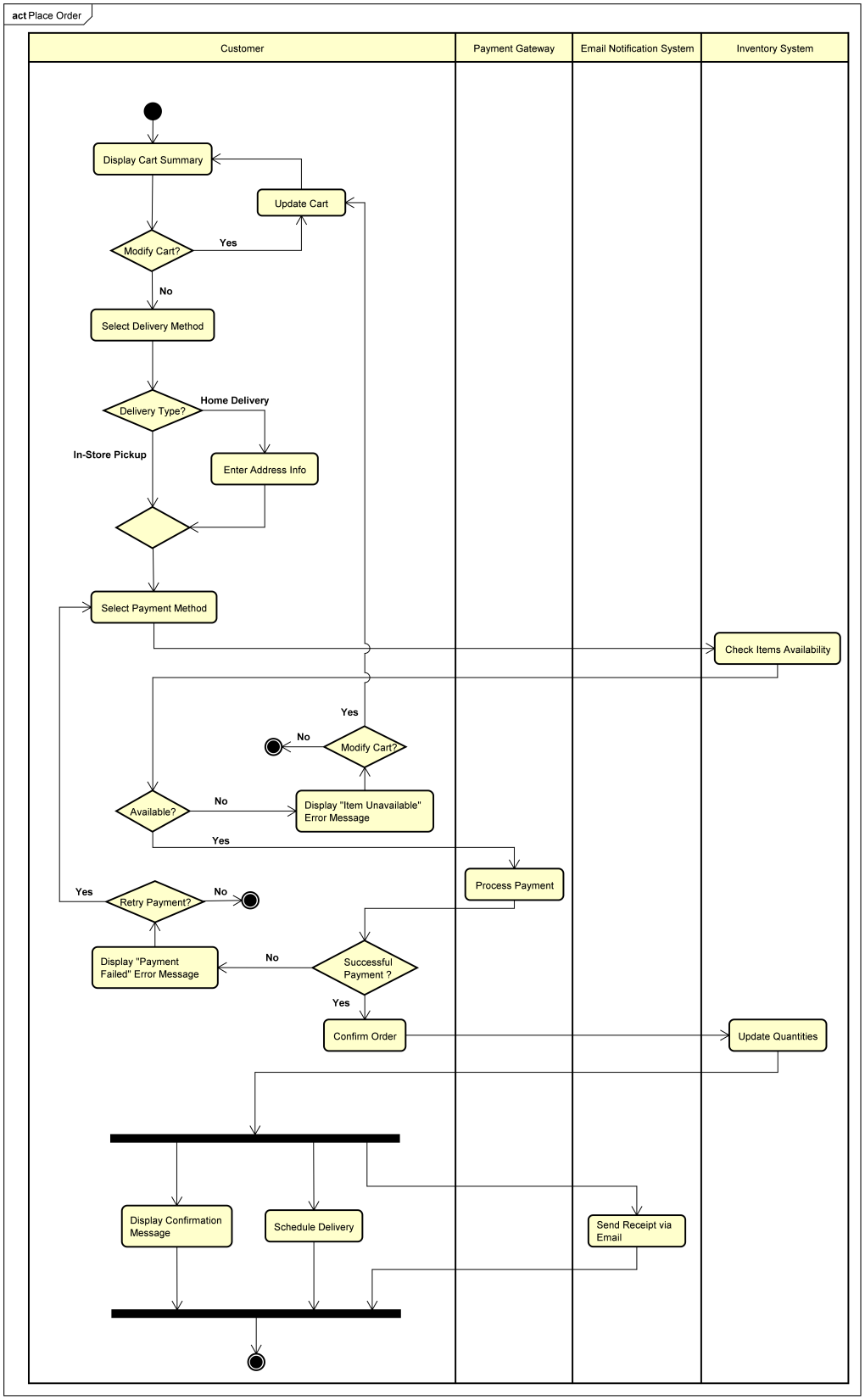


Figure 4: Activity Diagram for UC06 – Place an Order

2.10.3 UC05 – Register Account (Mumen Anbar)

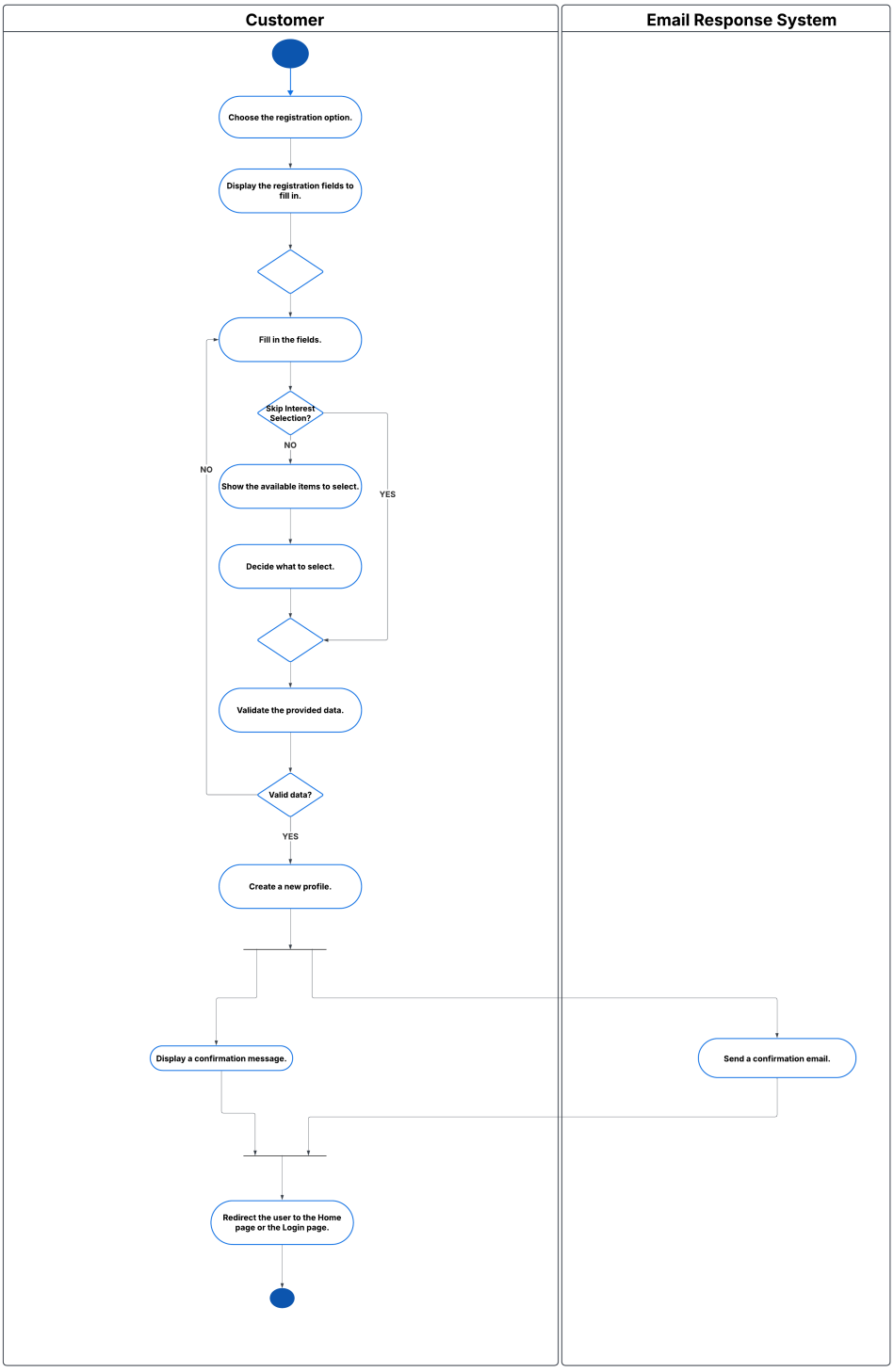


Figure 5: Activity Diagram for UC14 – Register Account

2.10.4 UC09 – Request Return or Refund (Waleed Rimawi)

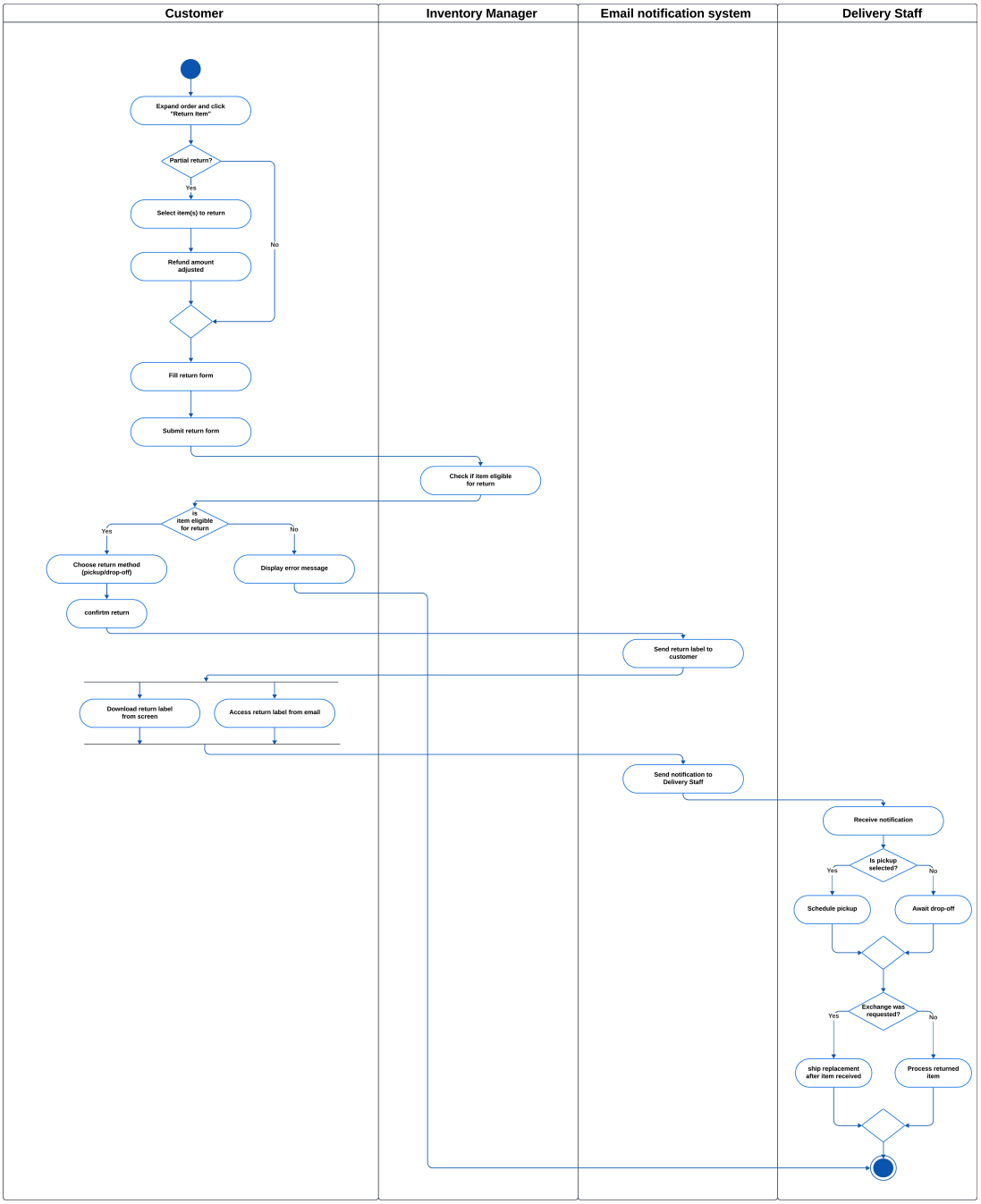


Figure 6: Activity Diagram for UC23 – Request Return or Refund

2.10.5 UC10 – Track Order Status (Salah Dawabsheh)

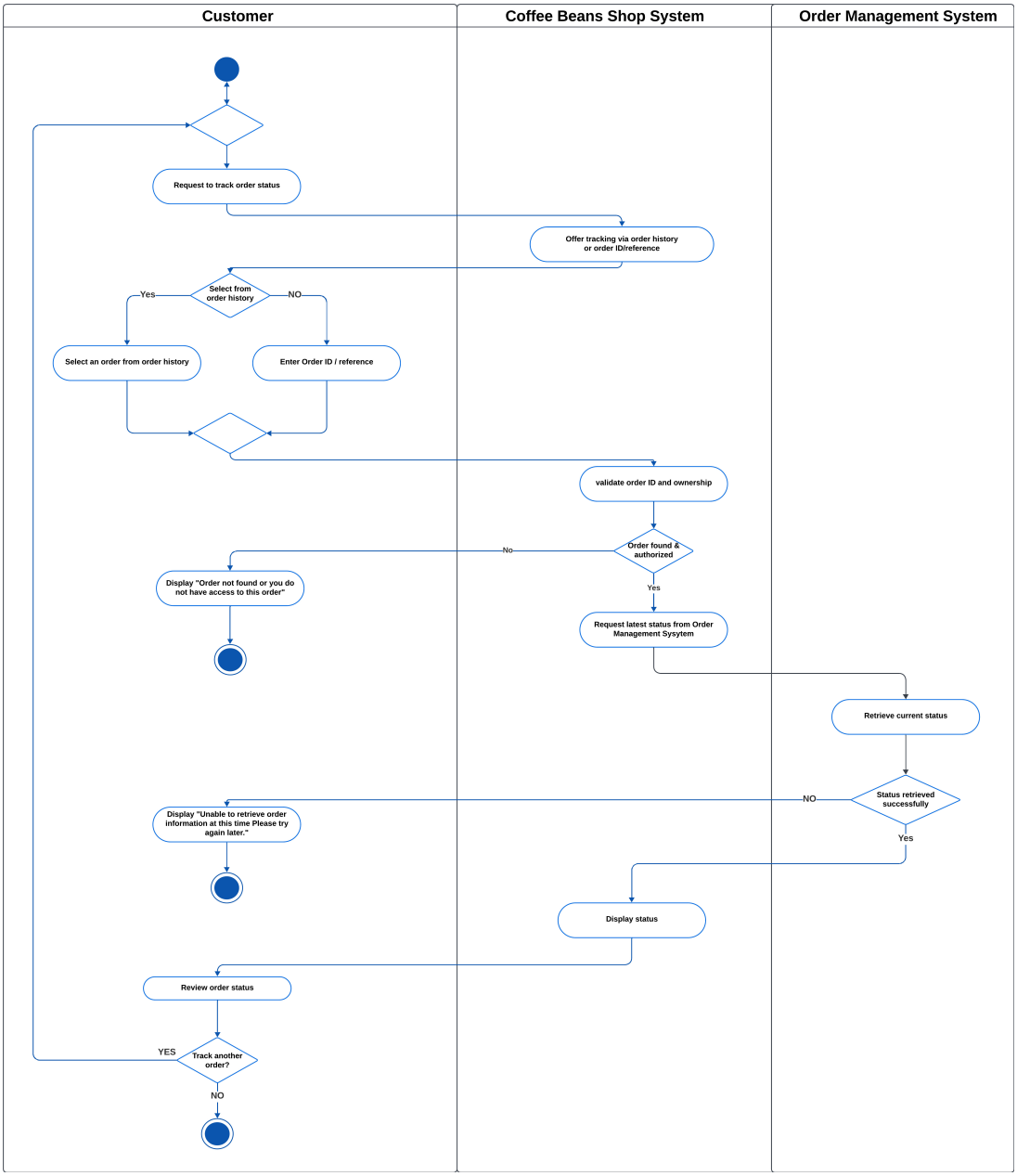


Figure 7: Activity Diagram for UC24 – Track Order Status

3. System Analysis and Modelling

3.1 Analysis Class Diagram - Modified

Lead: Waleed Rimawi
Contributors: Osaid Nur (modelling), Salah Dawabsheh (validation), Mumen Anbar (design), Jana Sawalmeh (reviewing)

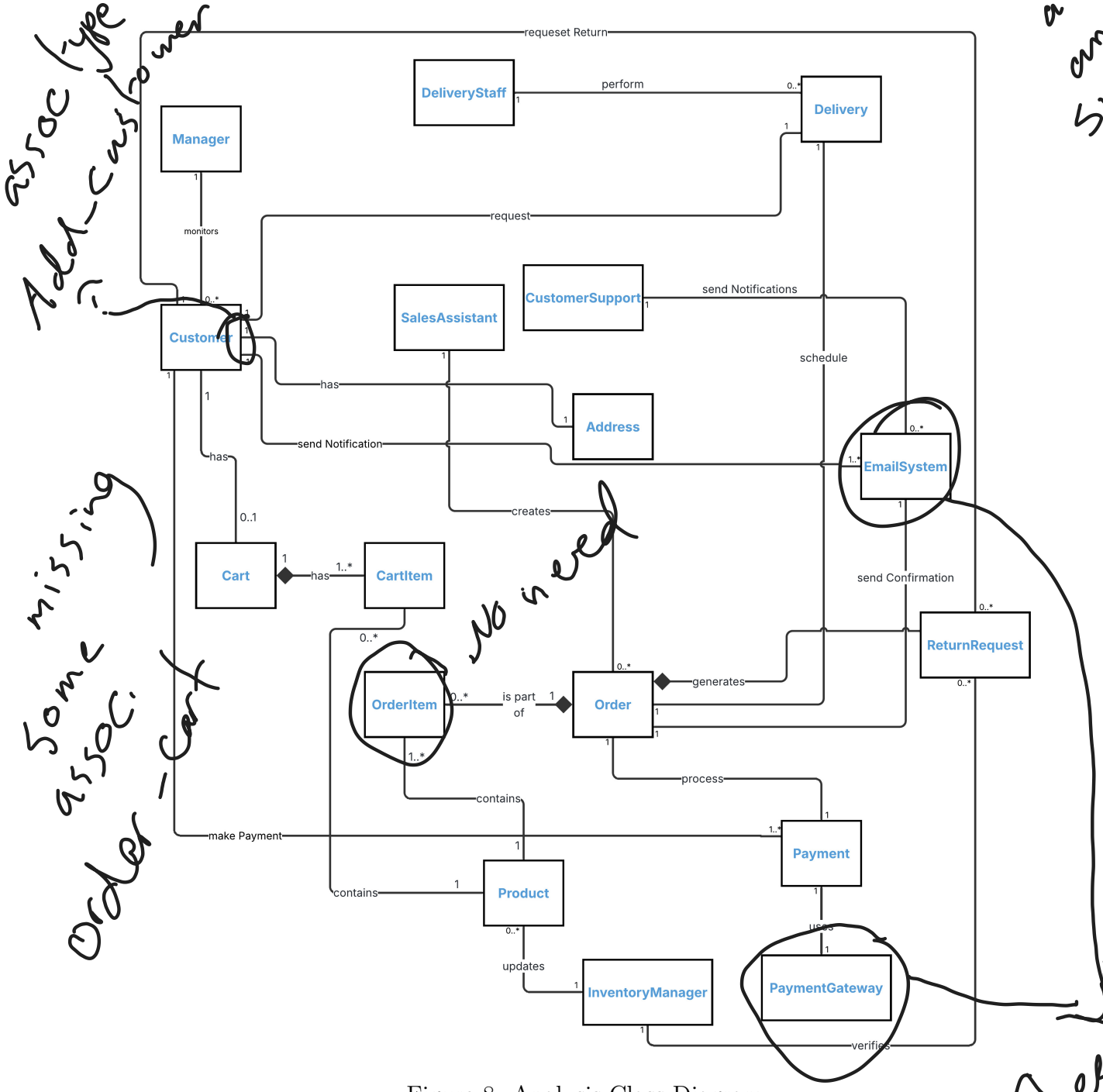


Figure 8: Analysis Class Diagram

3.2 Detailed Class Diagram - Modified

Lead: Osaid Nur
Contributors: Waleed Rimawi (modelling),Jana Sawalmeh(validation), Mumen Anbar (design), Salah Dawabsheh(reviewing)

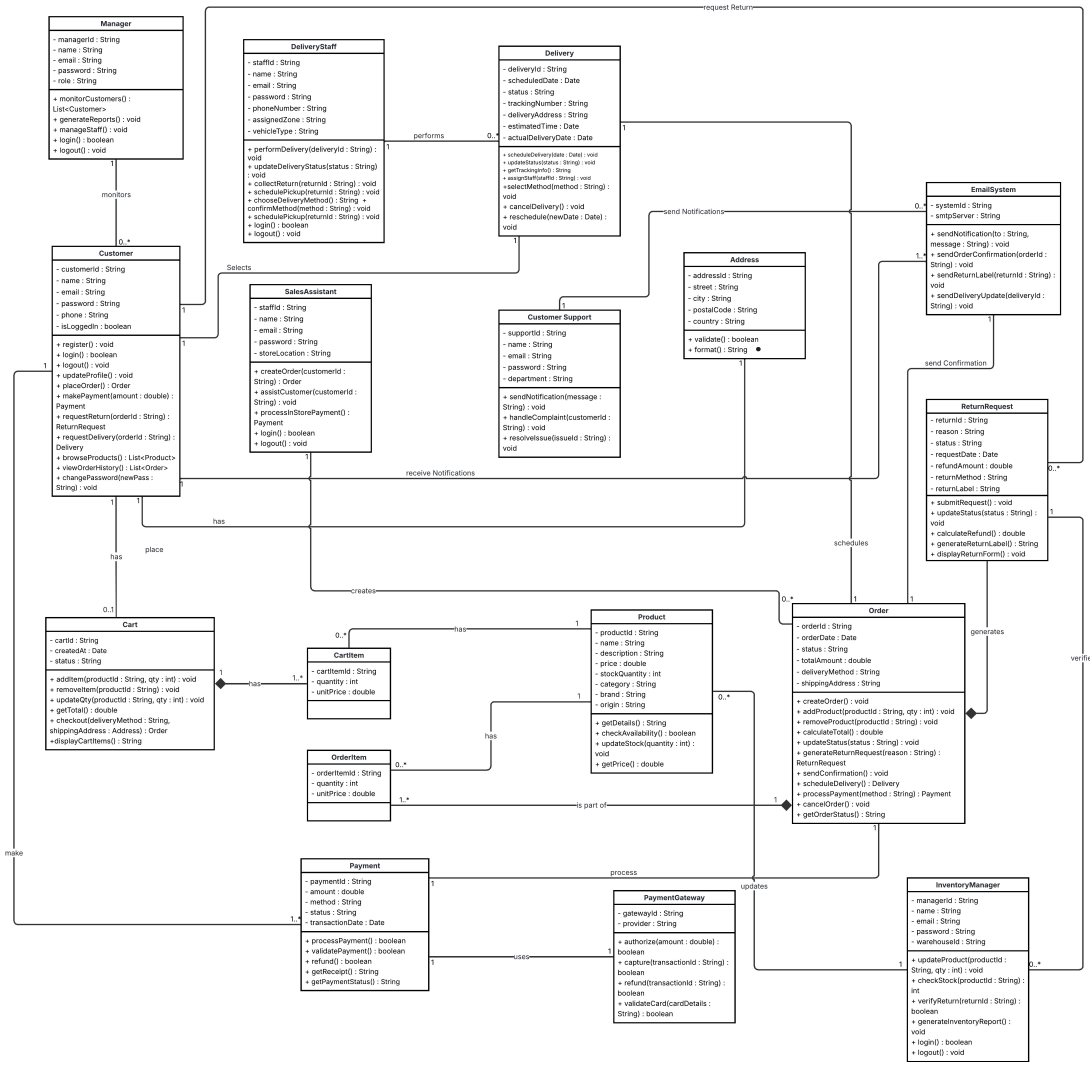


Figure 9: Detailed Class Diagram

missing some getters & setters where needed

3.3 System Sequence modelling and Analysis

3.3.1 UC01 – Browse Coffe Beans (Jana Sawalmeh)

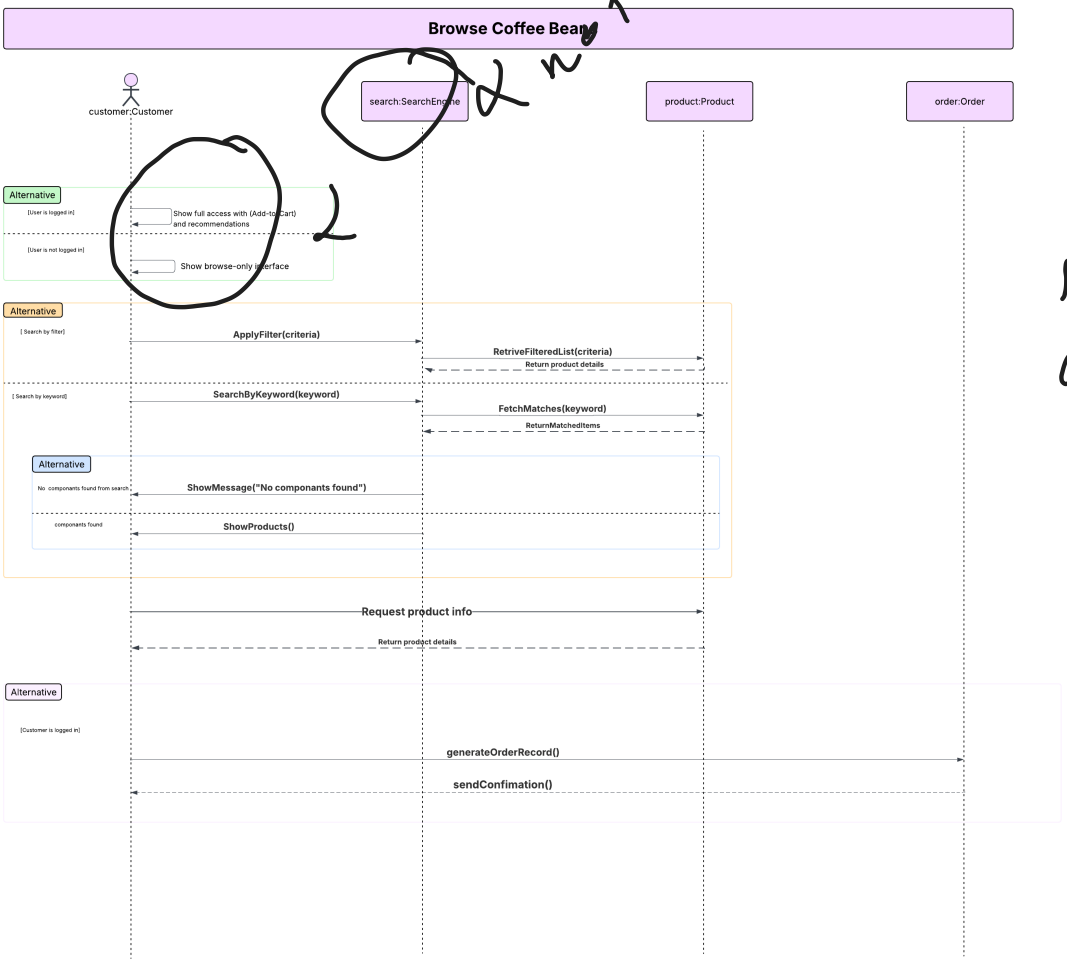


Figure 10: Sequence Diagram for UC01 – Browse Coffe Beans

3.3.2 UC04 – Place an Order (Osaid Nur)

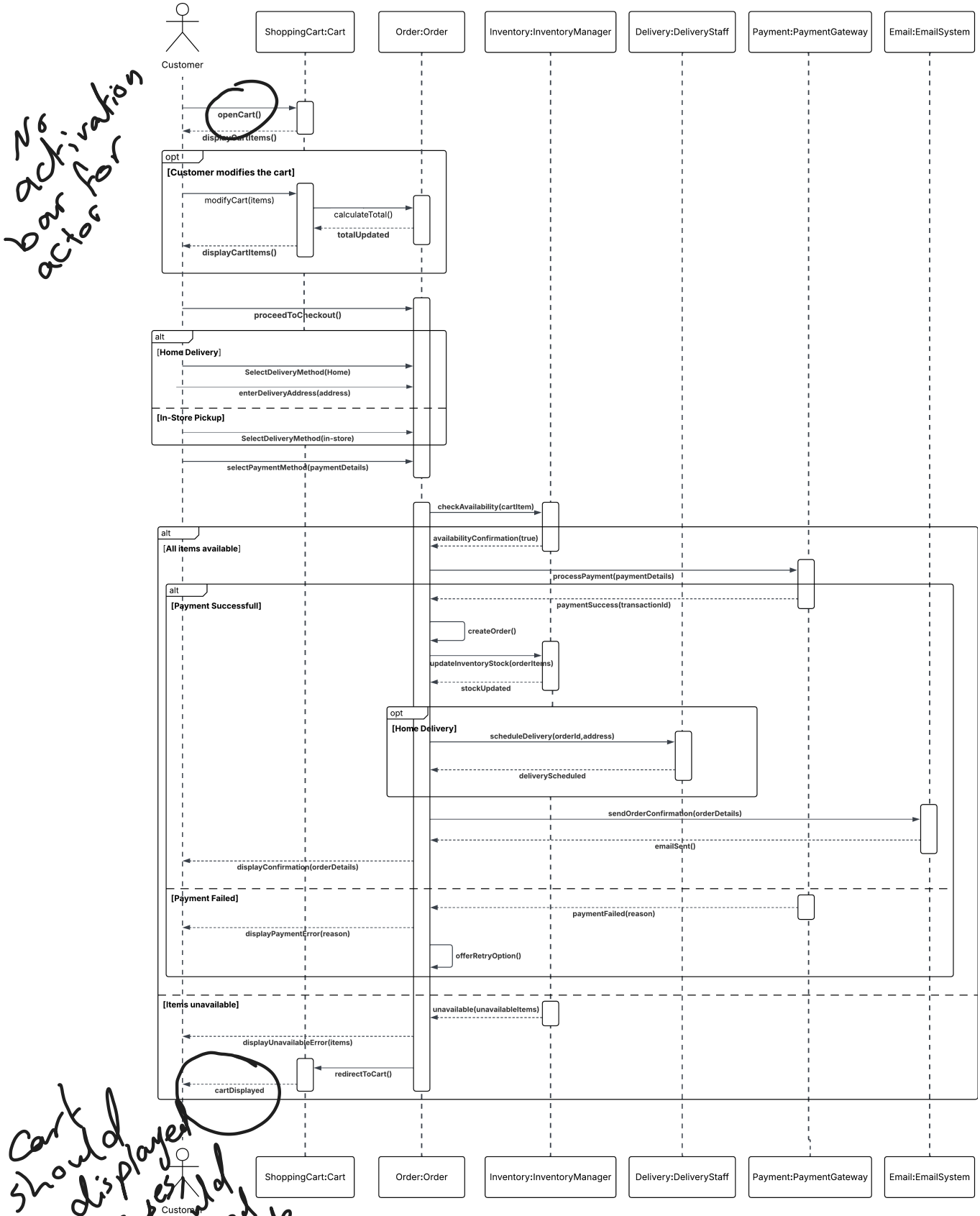


Figure 11: Sequence Diagram for UC04 – Place an Order

3.3.3 UC14 – Register Account (Mumen Anbar)

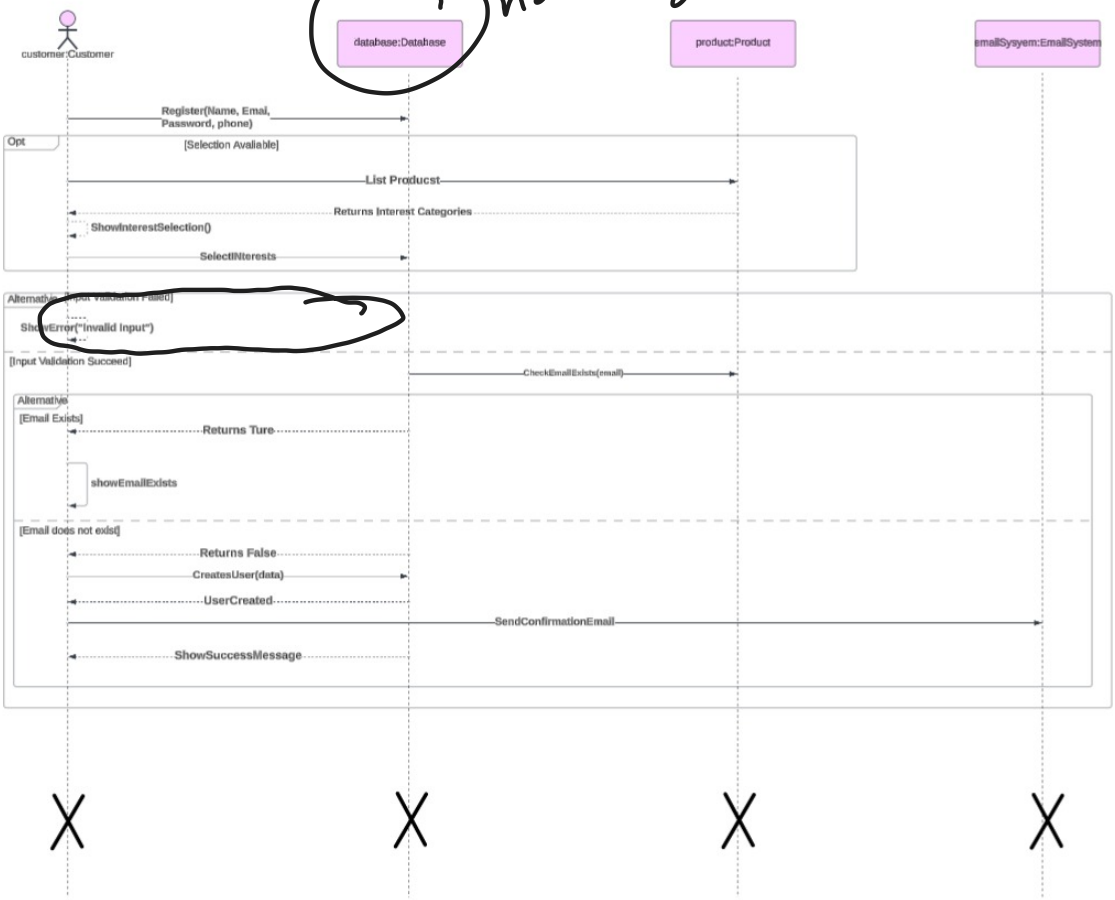


Figure 12: Sequence Diagram for UC14 – Register Account

3.3.4 UC09 – Request Return or Refund (Waleed Rimawi)

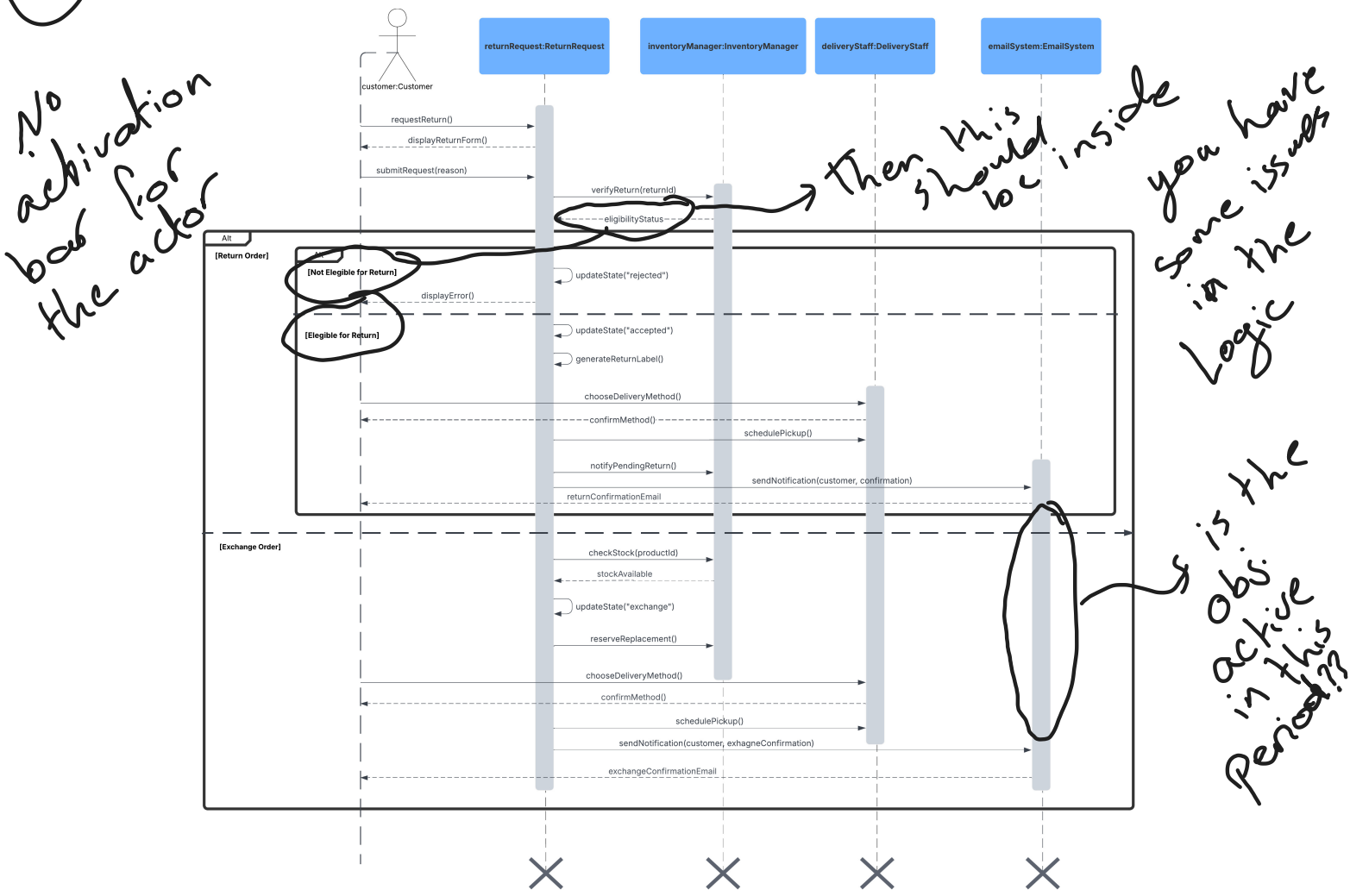


Figure 13: Sequence Diagram for UC09 – Request Return or Refund

3.3.5 UC10 – Track Order Status (Salah Dawabeheh)

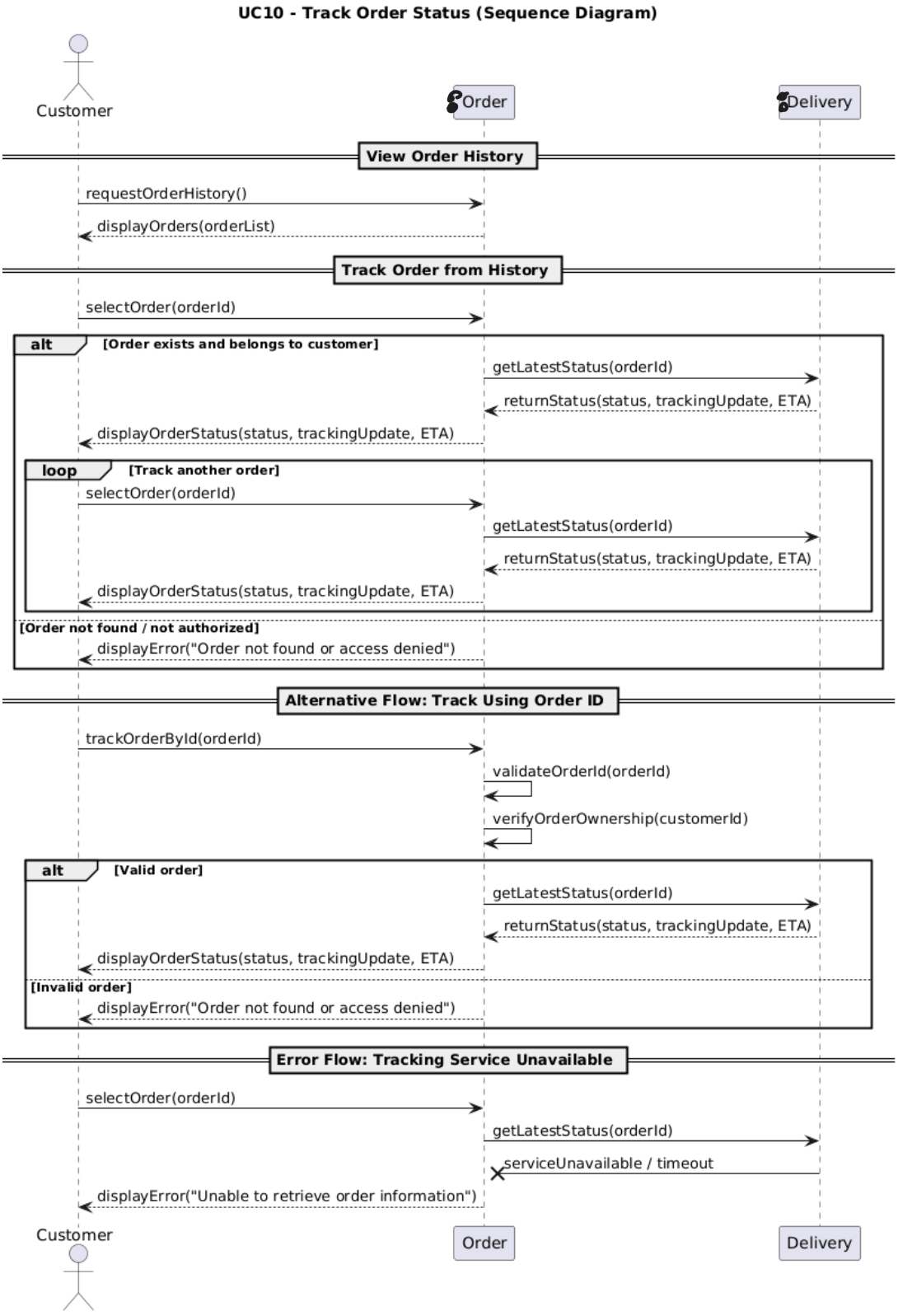


Figure 14: Sequence Diagram for UC10 – Track Order Status

4. System Design and Modelling

4.1 System Design Goals

Lead: Jana Sawalmeh
Contributors: Salah Dawabsheh (modelling), Osaid Nur (validation), Mumen Anbar (design), Waleed Rimawi (reviewing)

4.1.1 Low Coupling

Design Purpose: The system is designed to reduce tight dependencies between components in order to support easier maintenance, scalability, and independent system evolution within the Coffee Beans Shop.

Design Strategy: Primary system components such as `[Order]`, `[Payment]`, `[InventoryManager]`, `[Delivery]`, and `[EmailSystem]` interact exclusively through clearly defined interfaces. This separation ensures that business logic remains independent from technical and infrastructural services.

Design Choices Supporting Low Coupling:

1. **Interface-Oriented Integration:**
   Each component publishes its functionality via provided interfaces (`[InterfaceInventory]`, `[InterfacePayment]`, `[InterfaceDelivery]`) and consumes services through required interfaces, allowing components to be replaced or modified without impacting others.

2. **Service-Based Communication Contracts:**
   Core operations such as `[checkBeanAvailability()]`, `[processPayment()]`, and `[scheduleDelivery()]` are exposed as service methods, preventing direct access to internal data structures.

3. **Encapsulation of Supporting Services:**
   System-wide services including order notifications, confirmation messages, and configuration handling are isolated within the `[EmailSystem]` component, avoiding unnecessary dependencies on core business components.

4. **Unidirectional Dependency Flow:**
   Dependencies are structured so that higher-level components rely on abstract service definitions rather than concrete implementations, reducing the impact of future changes.

4.1.2 High Cohesion

Design Purpose: High cohesion is achieved by ensuring that each system component focuses on a single, clearly defined responsibility, improving system clarity and reusability.

Design Strategy: Components are organised around core business functions such as order processing, inventory control, payment handling, and user interaction.

Design Choices Supporting High Cohesion:

1. **Function-Oriented Component Design:**

The `[Order]` component is responsible for managing the entire order lifecycle, while the `[InventoryManager]` component handles stock tracking and availability updates.

2. **Dedicated Payment Handling:**
   All payment-related logic is encapsulated within the `[Payment]` component, ensuring that financial operations are not mixed with ordering or inventory logic.

3. **Clear Role Separation for Actors:**
   Actor classes such as `[Customer]`, `[DeliveryStaff]`, and `[StoreManager]` initiate actions appropriate to their roles, while business rules remain within service components.

4. **Logical Grouping of Product Entities:**
   Related entities such as `[CoffeeBean]`, `[BeanOrigin]`, and `[BeanCategory]` are grouped within the same component to minimise internal communication and strengthen cohesion.

4.1.3 Specific System Design Goal: Inventory Accuracy and Order Responsiveness

Design Purpose: A key system-specific goal is to ensure accurate and up-to-date inventory information while maintaining fast and reliable order processing for customers.

Design Strategy: The `[Order]`, `[InventoryManager]`, and `[CoffeeBean]` components collaborate to verify product availability, quantities, and attributes during product selection and checkout.

Design Choices Supporting This Goal:

1. **Inventory Validation Before Order Confirmation:**
   Stock levels are checked through the `[InventoryManager]` interface prior to finalising an order to prevent overselling.

2. **Efficient Product Search and Filtering:**
   Customers can browse and filter coffee beans by origin, category, or grind type using optimised queries within the `[CoffeeBean]` component.

3. **Controlled Component Interaction:**
   Inventory checks and order preparation occur within a defined interaction boundary, reducing communication overhead and checkout delays.

4. **Immediate Customer Feedback:**
   When requested products are unavailable, the system provides instant feedback to avoid inconsistent order states.

5. **Operational Monitoring and Oversight:**
   Inventory levels and order-processing behaviour can be monitored by management components to detect shortages or performance issues.

4.2 Component Diagram

Lead: Waleed Rimawi
Contributors: Salah Dawabsheh (modelling), Osaid Nur (validation), Mumen Anbar (design), Jana Sawalmeh (reviewing)

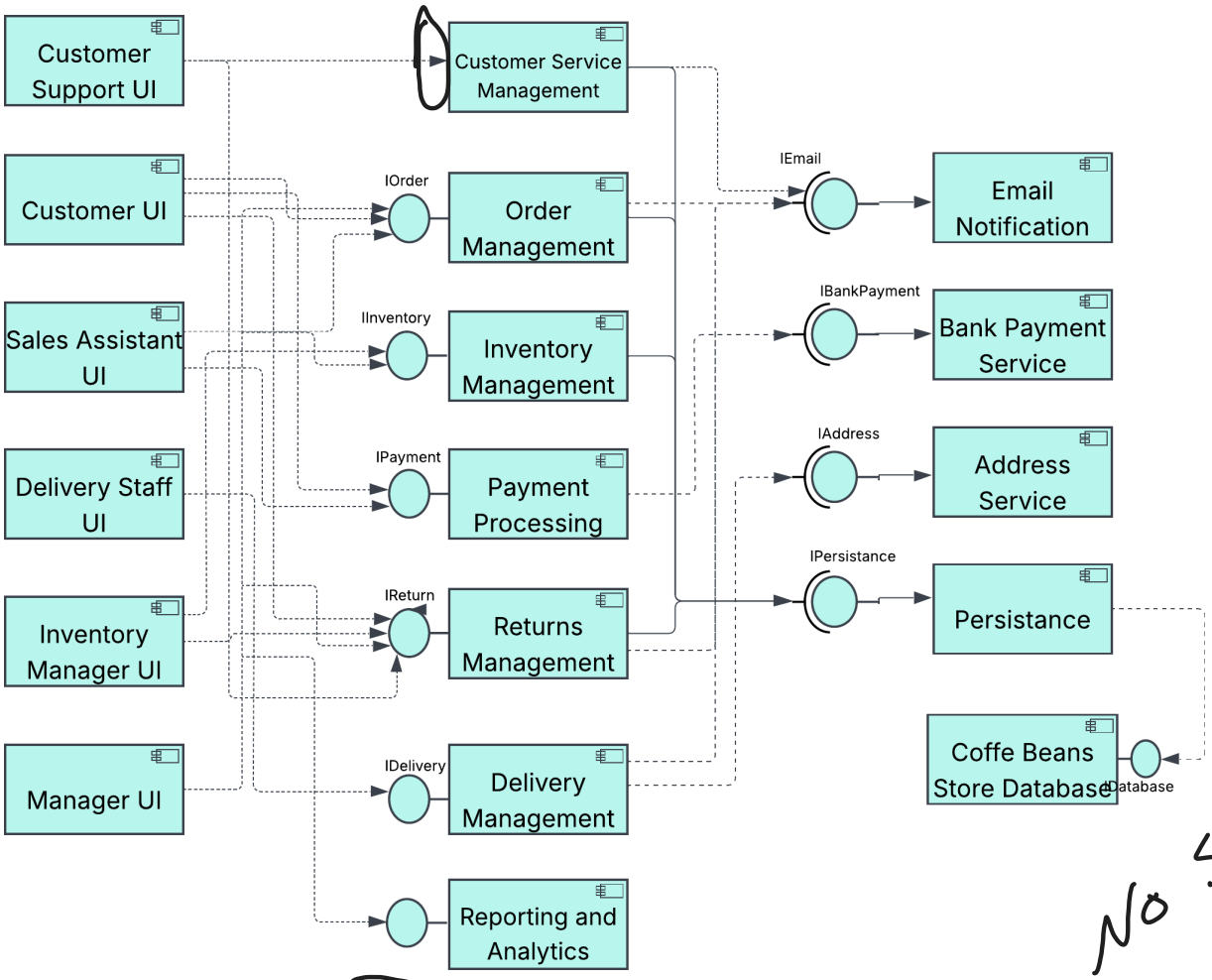
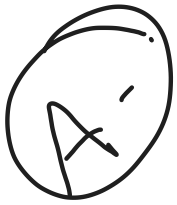


Figure 15: Component Diagram



4.3 Overall Architecture Diagram

Lead: Salah Dawabsheh
Contributors: Mumen Anbar (modelling), Osaid Nur (validation), Waleed Rimawi (design), Jana Sawalmeh (reviewing)

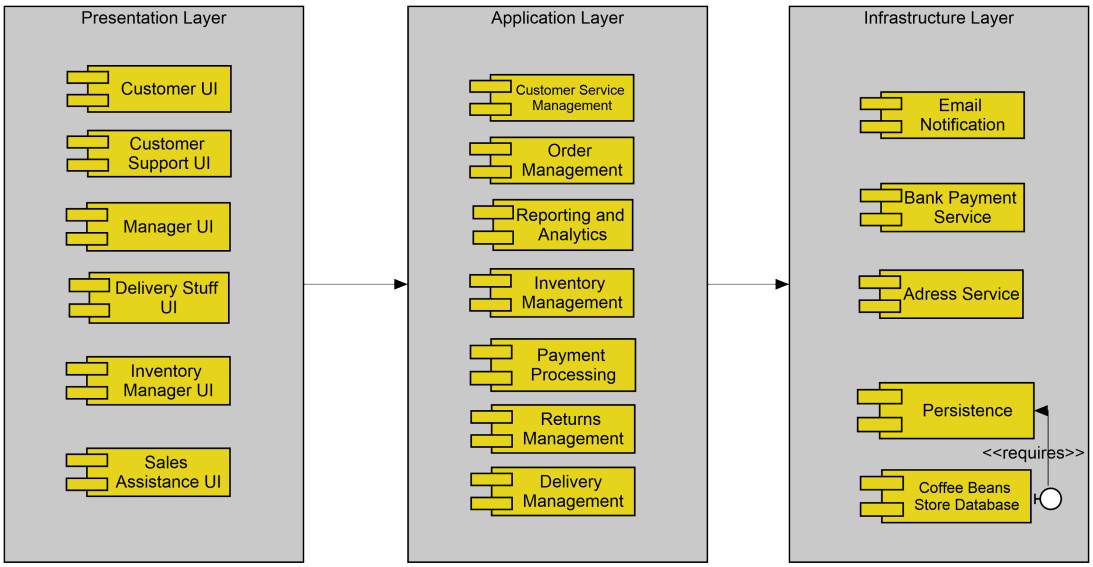


Figure 16: Architectural Diagram

4.3.1 Architecture Justification

The Coffee Beans Shop System adopts a **Layered Architecture** to organize functionality, support system design goals, and manage system complexity. The architecture is structured into three logical layers, each responsible for a specific aspect of the system.

- **Presentation Layer (UI Layer):** Provides user-facing interfaces tailored to different roles, including `Customer UI` for browsing and ordering coffee beans, `Sales Staff UI` for order management and customer assistance, `Delivery Staff UI` for delivery status updates, and `Manager UI` for inventory monitoring and reporting.
- **Application Layer (Business Logic Layer):** Encapsulates the core business functionality of the Coffee Beans Shop System. This layer includes components such as `Order Management`, `Inventory Management`, `Payment Processing`, `User Authentication`, and `Delivery Coordination`. It enforces business rules and coordinates workflows between system components.
- **Infrastructure Layer:** Handles technical and external services, including `Database Persistence`, `Notification Services` (email/SMS), `Payment Gateway Integration`, and `Address and Location Services` used to support delivery operations.

4.3.2 Rationale for Choosing Layered Architecture

The layered architectural style was selected to support the system's **design goals and non-functional requirements**, particularly maintainability, scalability, and clarity of responsibility.

1. **Support for Low Coupling:** Each layer communicates with adjacent layers through well-defined interfaces. For example, the application layer depends on abstract services provided by the infrastructure layer (such as data persistence and notification services), reducing dependency on specific implementations.
2. **Promotion of High Cohesion:** Responsibilities are clearly separated across layers. User interaction logic resides in the presentation layer, business rules and workflows are centralized in the application layer, and technical concerns are isolated within the infrastructure layer.
3. **Inventory Accuracy and System Responsiveness:** Inventory validation, product filtering, and order processing are handled within the application layer, while optimized database access is managed by the infrastructure layer. This separation supports fast response times during browsing and checkout.
4. **Ease of Maintenance and Evolution:** Changes to pricing rules, payment methods, delivery workflows, or user interfaces can be implemented within their respective layers with minimal impact on the rest of the system.
5. **Role-Based Interaction and Security:** Separating user interfaces by role simplifies access control and enhances security, ensuring that customers, staff members, and managers interact only with system functions relevant to their responsibilities.

Some are not mentioned in your design goals.

4.3.3 Conclusion

The selected layered architecture provides a clear separation of concerns while supporting scalability and long-term maintainability. It aligns closely with the Coffee Beans Shop System's operational requirements by enabling efficient order processing, accurate inventory tracking, and seamless integration with external services such as payment gateways and notification systems. This architectural approach ensures that the system remains flexible and robust as business needs evolve.

4.4 Deployment Diagram

Lead: Salah Dawabsheh
Contributors: Mumen Anbar (modelling), Osaid Nur (validation), Waleed Rimawi (design), Jana Sawalmeh (reviewing)

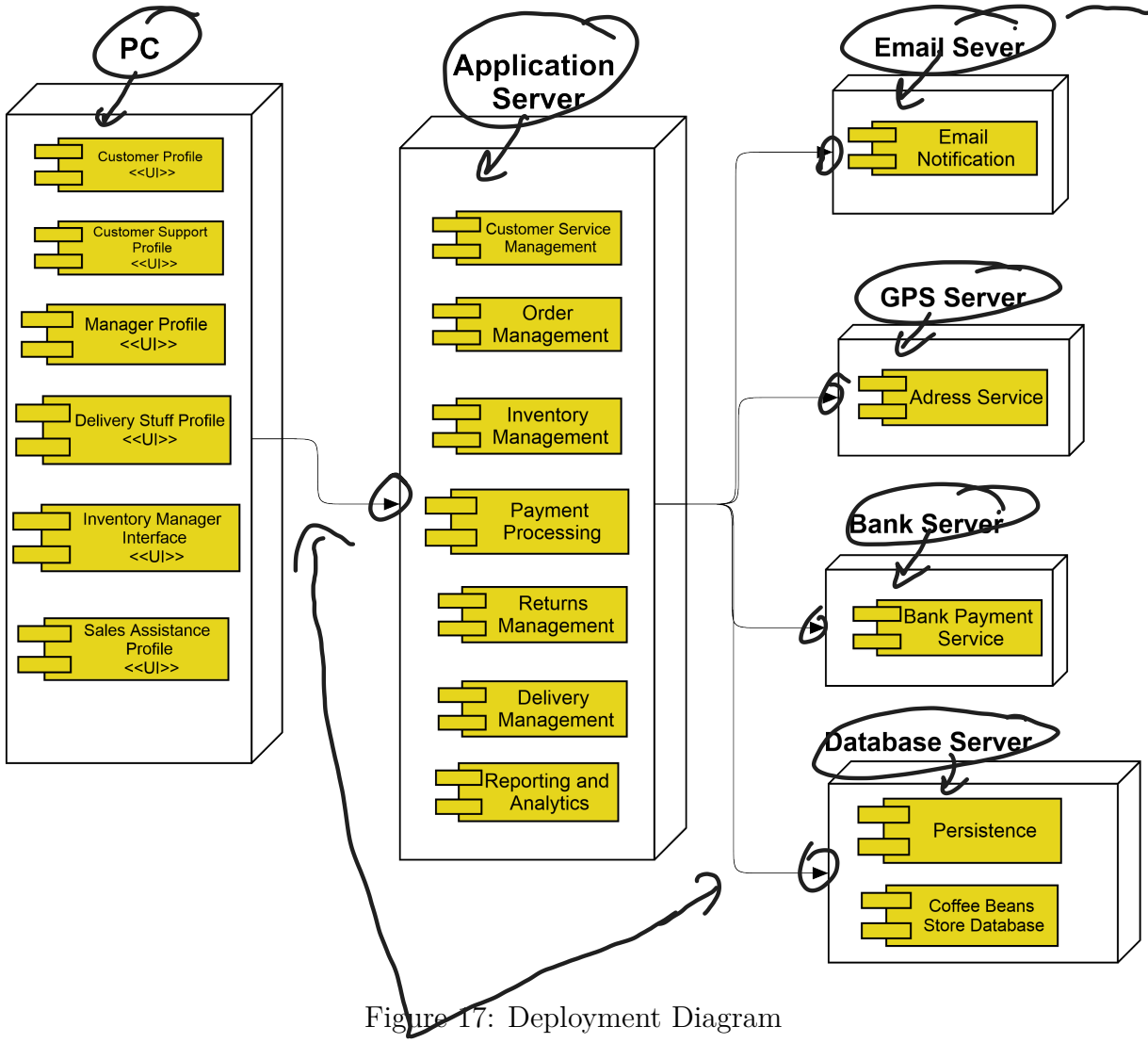


Figure 17: Deployment Diagram

Appendix-I: Project Meetings

Lead: Osaid Nur
Contributors: Jana Sawalmeh(modelling), Salah Dawabsheh (validation), Waleed Rimawi (design), Mumen Anbar (reviewing)

Customer–Developer Group Meetings [G5-G4]

Meeting 1

Call to Order: Held at **Birzeit University campus** on **Saturday, 6 Dec 2026**, at **12:00 PM**

Attendees:

- **Developer Group G4:** Wael Hmdan, Hakam Abualia, Mohmoud Dawoud, Hilmi Shakarneh , Yousef Jebreel
- **Customer Group G5:** Mumen Anbar , Salah Dawabsheh , Waleed Rimawi , Jana Sawalmeh
- **Members didn't attend:** None

Topics Discussed Through Meeting 1:

- Have general view of our Coffee Beans Shop business
- Differences between Filtering and search operations
- Want to have Request Return, Refund order

Things Agreed Through Meeting 1:

- Request Return or Refund.
- Make search and filter in one operation.
- Prepare the initial copy of user requirement.
- Schedule next meeting at Birzeit University

Meeting 2

Call to Order: Held at **Birzeit University campus** on **Saturday, 20 Dec 2026**, at **12:00 PM**

Attendees:

- **Developer Group G4:** Wael Hmdan
- **Customer Group G5:** Mumen Anbar , Salah Dawabsheh , Waleed Rimawi , Jana Sawalmeh
- **Members didn't attend:** Hakam Abualia, Mohmoud Dawoud, Hilmi Shakarneh , Yousef Jebreel

Topics Discussed Through Meeting 2:

- Have view of the user requirements
- Modify understanding of some points in User requirments

Decisions Taken Through Meeting 2:

- Finalize the User Requirements and edit some of them for submission

Internal Developer Team Meetings

Date	**Purpose**	**Attendees**
Wednesday, 10 Dec 2025	Phase 2 Requirement Engineering - Elicitation and Discovery	All team members
Tuesday, 30 Dec 2025	Phase 3 Requirements Analysis and Modelling	All team members
Tuesday, 20 Jan 2026	Phase 4 System Modelling and Design	All team members
Multiple Dates (Jan 2026)	Final Report Writing, Presentation Preparation, Model Consolidation	All team members (Discord and google meet meetings with shared documentation)