CS5700 Project 1: Object Oriented Programming
Yuxin Liang

1. Business Requirements – Package Tracking System

- The system will automatically extract package details from emails, detecting tracking numbers, delivery companies (FedEx, UPS, DHL), and estimated delivery dates. It will store relevant details such as the sender, item description (if available), and tracking links. The system will organize packages by sorting them in chronological order based on delivery dates and allowing users to filter by status (Delivered, In Transit, Arriving Soon). Users will receive simulated notifications for upcoming deliveries using function-based event handling. Additionally, users can categorize packages by tagging them with labels such as "Electronics," "Clothing," or "Gift" for better organization. The system will also provide search and filter functionality, enabling users to find packages by tracking number, sender, or category, and filter based on delivery status.

- **Nouns-verbs**
  - **Nouns**
    - Package, Email, Tracking Number, Delivery Company, Delivery Date, Sender, Item Description, Tracking Link, Status, Notification, Function-based Event Handling, Label, Category, Users.
  - **Verbs**
    - Extract, Detect, Store, Organize, Sort, Filter, Receive, Categorize, Tag, Provide, Find.
- **Target Audience**
  - Individual Consumers: Users who receive multiple packages and want to track their shipments efficiently.
  - Online Shoppers: Users frequently purchasing from e-commerce platforms like Amazon, eBay, or other online stores.
  - Business Professionals: People managing office shipments and deliveries.

- Rules
  - The system should automatically parse emails to detect tracking numbers.
  - The system should identify the delivery company (FedEx, UPS, DHL) from the email.
  - The system should extract the estimated delivery date from the email.
  - The system should store package details, including sender information, item description (if available), and tracking links.
  - The system should sort packages by estimated delivery date in ascending order.
  - Users should be able to filter packages based on delivery status: Delivered, In Transit, Arriving Soon.

    - The system should check delivery dates and simulate a console notification when a package is expected to arrive tomorrow.
    - Users should be able to view package contents.
    - Users should be able to tag packages with categories such as Electronics, Clothing, or Gift.
    - Users should be able to search for packages by tracking number, sender, or category.
    - Users should be able to filter packages based on status (Delivered, In Transit).
    - The system should extract delivery dates from emails and display upcoming deliveries in chronological order.

- **Challenge Questions (ask user) - questionnaire, no need to answer, but it is ok to make assumptions**
    - **Are three shipping companies enough?**
    - **Should package categories be predefined, or should users create their own custom labels?**
    - **Can users filter packages by multiple criteria at once?**
    - **What happens if an email contains multiple tracking numbers?**
- **Summary of Classes, Attributes and Associations (from nouns and verbs)**
    - **User**
        - Attributes:
            - userId: String
            - name: String
            - email: String
            - phone: String
            - preferences: Object (e.g., preferred shipping companies, notification settings)
            - trackedPackages: Array<Package>
        - Methods:
            - register(email, phone): void
            - login(email, password): void
            - viewTrackedPackages(): Array<Package>
            - filterPackages(status): Array<Package>
            - categorizePackage(trackingNumber, category): void
            - searchPackage(trackingNumber): Package | null
            - receiveNotifications(): void
    - Package
        - Attributes:
            - trackingNumber: String

        - deliveryCompany: String (FedEx, UPS, DHL)
        - sender: String
        - itemDescription: String
        - deliveryDate: Date
        - status: String (Delivered, In Transit, Arriving Soon)
        - category: String (Electronics, Clothing, Gift, etc.)
      - Methods:
        - updateStatus(newStatus): void
        - assignCategory(category): void
        - getTrackingDetails(): Object
  - EmailParser
      - Attributes:
        - emailContent: String
      - Methods:
        - extractTrackingNumber(email): String
        - extractDeliveryCompany(email): String
        - extractSenderDetails(email): String
        - extractItemDescription(email): String
        - extractDeliveryDate(email): Date
  - PackageManager
      - Attributes:
        - packages: Array<Package>
      - Methods:
        - addPackage(package: Package): void
        - sortPackagesByDate(): Array<Package>
        - filterPackagesByStatus(status): Array<Package>
        - searchPackageByTrackingNumber(trackingNumber): Package | null
        - searchPackageBySender(sender): Array<Package>
        - assignCategory(trackingNumber, category): void
  - NotificationSystem
      - Attributes:
        - notifications: Array<String>
      - Methods:
        - sendNotification(message): void
        - checkUpcomingDeliveries(packages: Array<Package>): void

Associations:

- User and Package (One-to-Many)
    - User can have multiple packages.

- EmailParser and PackageManager (One-to-Many)
    - EmailParser extracts package details from emails and passes them to PackageManager.
- NotificationSystem and User (One-to-One)
    - User receives package notifications.

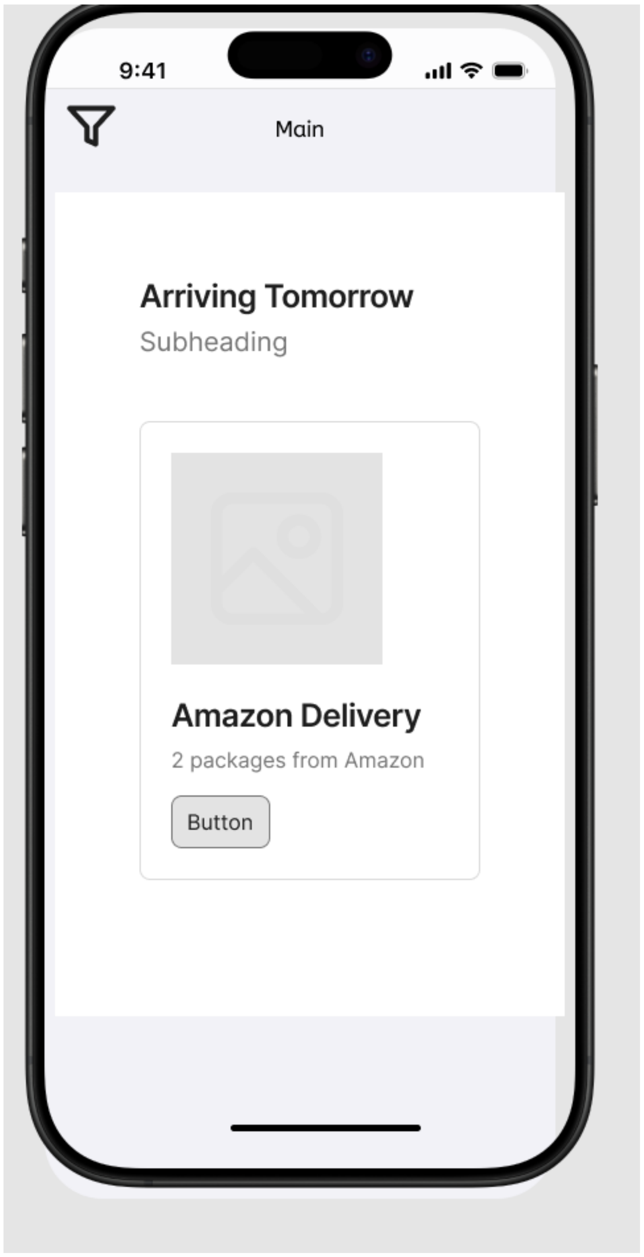
Aggregation

- PackageManager and Package (One-to-Many)
    - PackageManager is a container for Packages, but Packages can exist independently.
- NotificationSystem and User (One-to-One)
    - User receives package notifications.

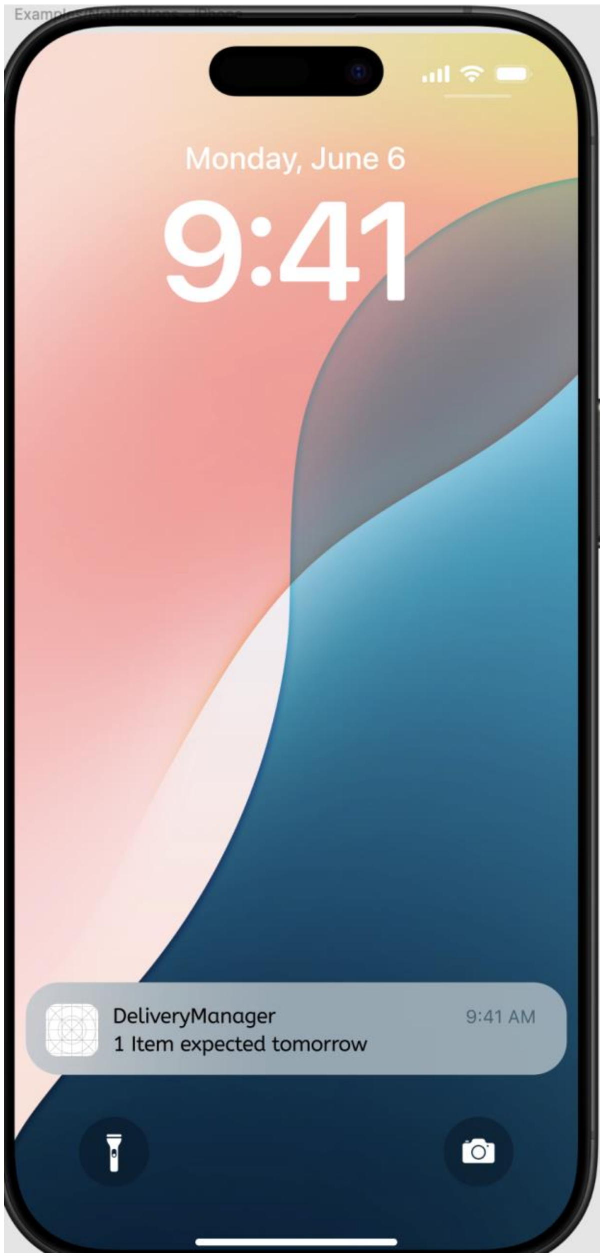
- **User personas (at least 2) and user stories (at least 3 per persona)**
    - **Persona 1: Sarah (Frequent Online Shopper)**
        - Dimension: Consumer with moderate interaction
        - Age: 28
        - Background: A busy marketing manager who frequently shops online and needs a convenient way to track deliveries.
        - Scenario: Sarah orders multiple packages from different retailers and often forgets when they will arrive. She wants an easy way to track her orders and receive reminders.
        - Motivation: Wants a centralized system to track package arrivals without manually checking emails.
        - User Stories:
            - As a frequent online shopper, I want to see a list of all my incoming packages sorted by delivery date so that I can easily know when my packages will arrive. (Mockup1)
            - As a frequent online shopper, I want to receive a notification when a package is arriving tomorrow so that I don’t forget to check for it. (Mockup 2)
            - As a frequent online shopper, I want to search for a package using the tracking number so that I can quickly check its status. (Mockup3)
    - Persona 2: Michael (Small Business Owner)
        - Dimension: Business user with high interaction
        - Age: 35

        - Background: Owns an online retail business and frequently receives shipments of inventory. Needs to track incoming stock efficiently.
        - Scenario: Michael manages multiple deliveries from different suppliers and needs a system to categorize and filter packages based on status and content.
        - Motivation: Wants an organized system that helps him manage inventory shipments without delays.
        - User Stories:
            - As a small business owner, I want to categorize packages with labels such as ‘Electronics’ or ‘Clothing’ so that I can easily track my inventory.
            - As a small business owner, I want to filter my packages by status (Delivered, In Transit) so that I can focus on pending deliveries.
            - As a small business owner, I want to receive a notification if a package is delayed so that I can plan accordingly.

- Interface low level mockups
    - Mockup1


9:41
Main
Arriving Tomorrow
Subheading
Amazon Delivery
2 packages from Amazon
Button

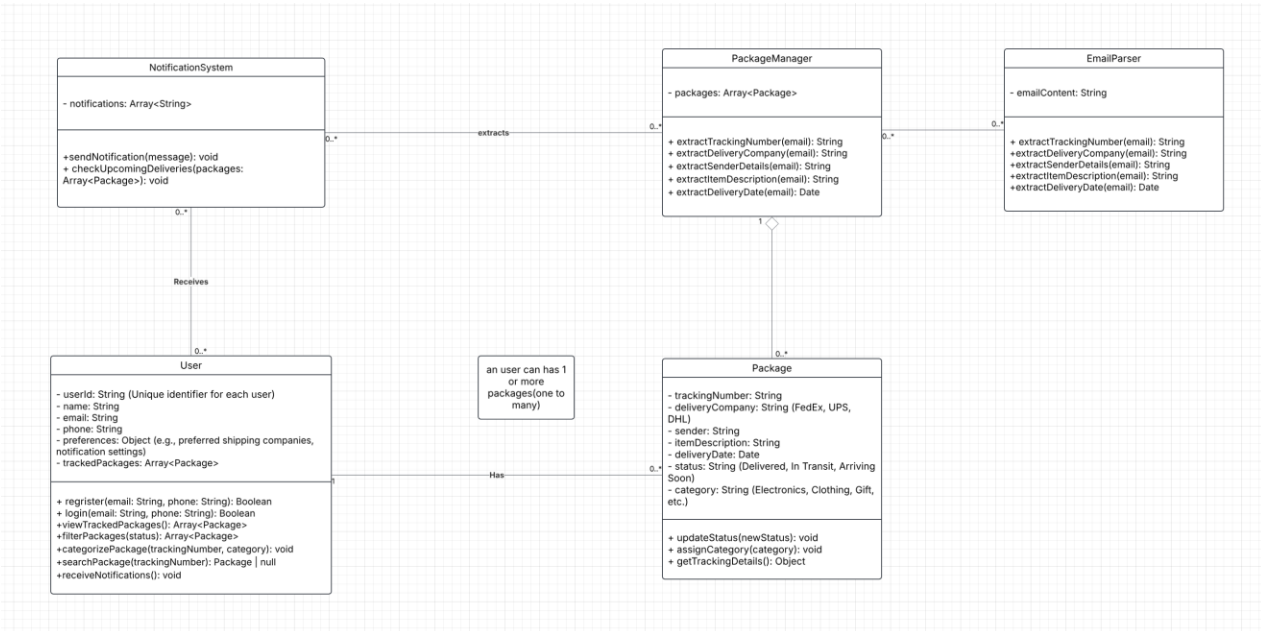
- Mockup2



DeliveryManager 9:41 AM
1 Item expected tomorrow

- Mockup 3

- UML Class Diagram


NotificationSystem
- notifications: Array<String>
+sendNotification(message): void
+ checkUpcomingDeliveries(packages: Array<Package>): void
PackageManager
- packages: Array<Package>
+ extractTrackingNumber(email): String
+ extractDeliveryCompany(email): String
+ extractSenderDetails(email): String
+ extractItemDescription(email): String
+ extractDeliveryDate(email): Date
EmailParser
- emailContent: String
+ extractTrackingNumber(email): String
+extractDeliveryCompany(email): String
+extractSenderDetails(email): String
+extractItemDescription(email): String
+extractDeliveryDate(email): Date
extracts
0..*
0..*
0..*
0..*
Receives
0..*
0..*
1
0..*
User
- userId: String (Unique identifier for each user)
- name: String
- email: String
- phone: String
- preferences: Object (e.g., preferred shipping companies, notification settings)
- trackedPackages: Array<Package>
+ regrister(email: String, phone: String): Boolean
+ login(email: String, phone: String): Boolean
+viewTrackedPackages(): Array<Package>
+filterPackages(status): Array<Package>
+categorizePackage(trackingNumber, category): void
+searchPackage(trackingNumber): Package | null
+receiveNotifications(): void
an user can has 1 or more packages(one to many)
Has
1
0..*
Package
- trackingNumber: String
- deliveryCompany: String (FedEx, UPS, DHL)
- sender: String
- itemDescription: String
- deliveryDate: Date
- status: String (Delivered, In Transit, Arriving Soon)
- category: String (Electronics, Clothing, Gift, etc.)
+ updateStatus(newStatus): void
+ assignCategory(category): void
+ getTrackingDetails(): Object

- **Use of Generative AI in Development**
  - **Purpose**
    - Generative AI (**ChatGPT**) was consulted to enhance my understanding of **class structures and relationships** for the **Package Tracking System**, specifically focusing on the **PackageManager** class.
  - **Prompts and Responses**
    - **Prompt:** " What other classes should I have besides User and PackageManager?"
    - **Response:** ChatGPT suggested additional classes such as:EmailParser (to handle package extraction from emails),NotificationSystem (to manage package delivery alerts),FilterSystem (to allow users to search and filter packages). These additions helped refine the class diagram and improve the logical structure of the system.
    - **Prompt:** " What should be the associative relationships between User, Package, and PackageManager? "
    - **Response:** ChatGPT explained that:
      - User interacts with PackageManager to track and manage packages.
      - PackageManager aggregates multiple Package objects and provides sorting/filtering.
      - NotificationSystem queries PackageManager to check for upcoming deliveries and send notifications.
  - **How it was Used**
    The AI-generated insights served as a starting point for refining my UML class diagram and system structure. While I ultimately designed my own PackageManager class and relationships, ChatGPT's responses helped me think through associations, class dependencies, and functionality gaps before finalizing the design.