Class Diagram for the Amazon Locker Service

Learn to create a class diagram for the Amazon Locker system problem using the bottom-up approach.

We'll cover the following

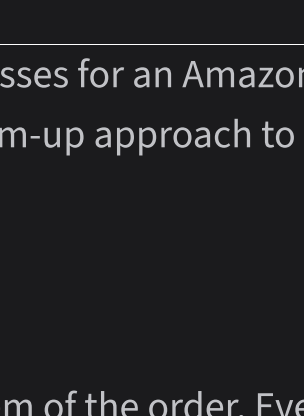
We'll create the class diagram for the Amazon Locker service. In the class diagram, we will first design the classes for the system, and then we will identify the relationship between classes according to the requirements gathered for the Amazon Locker service problem.

Components of an Amazon Locker service

In this section, we'll define the classes for an Amazon Locker service. As mentioned earlier, we are following the bottom-up approach to design a class diagram for the Amazon Locker service.

Item

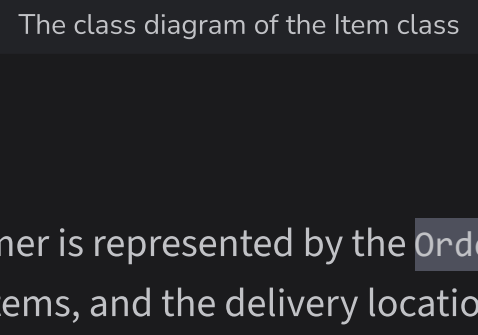
The `Item` class represents each item of the order. Every item contains an Id and the order's quantity. The class representation is shown below:



The class diagram of the Item class

Order

The order placed by a customer is represented by the `Order` class. Every order has a unique Id, contains a list of items, and the delivery location of the order. The UML representation of a class is shown below:



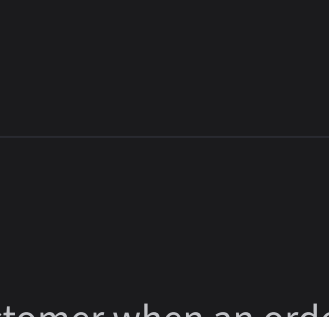
The class diagram of the Order class

R2: Amazon Locker Service

R2: One or more items can be contained in one order. An order will be placed in a package before the delivery.

Notification

The notification is sent to the customer when an order is shipped. A `Notification` class has a `customerId`, `orderId`, and `lockerId` that specify the customer to whom the notification has to be sent, against which order the notification has to be sent, and the locker where the order has arrived, respectively. Moreover, this class contains a 6-digit code to open the locker. The class definition is given below:



The class diagram of the Notification class

R5: Amazon Locker Service

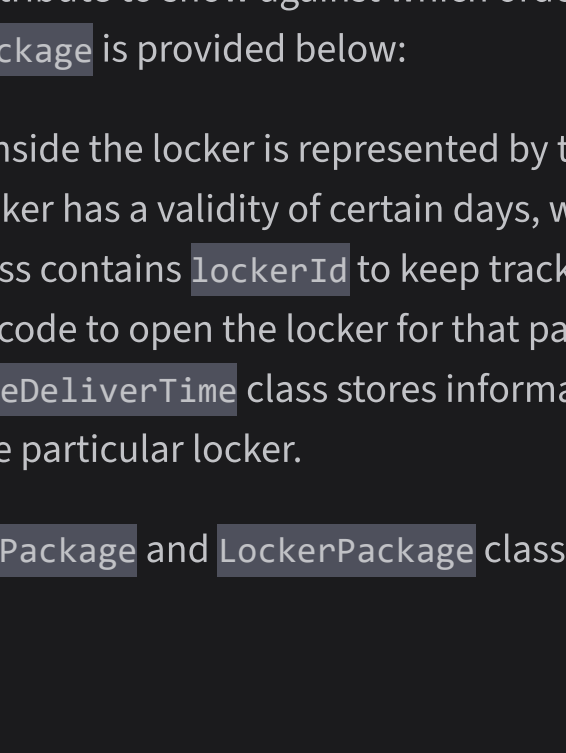
R5: When the order package is delivered to the locker location specified by the customer, a 6-digit code will be sent to the customer to open the locker.

Package and locker package

The `Package` class represents the order which is packaged. The package is what can be kept in the locker. Each package has its own id and size. Moreover, the `Package` class has an `Order` type attribute to show against which order the package is sent. The class representation of `Package` is provided below.

The package that is kept inside the locker is represented by the `LockerPackage` class. Since an item kept in a locker has a validity of certain days, we have `codeValidDays` to keep track of this. This class contains `lockerId` to keep track of which locker contains a particular package. The code to open the locker for that package is stored in the `code` class and the `packageDeliverTime` class stores information about when the package is delivered to the particular locker.

The representation of the `Package` and `LockerPackage` classes is shown below:



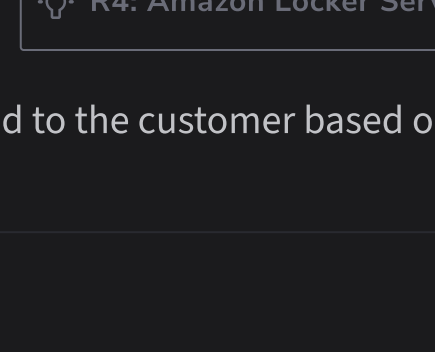
The class diagram of the Package and LockerPackage classes

R2, R6, and R7: Amazon Locker Service

R2: One or more items can be contained in one order. An order will be placed in a package before the delivery. **R6:** The package will be kept or placed inside the locker for three days only. **R7:** If the customer does not pick up the package from their locker within three days, the refund process will be initiated, and the customer won't be allowed to pick up the package any longer.

Locker

Since we are designing a locker service problem, we should have a `Locker` class. Every locker has its ID, size, and reference to the location ID. Moreover, the `Locker` class has a member, `lockerState`, to specify the present state of the locker. We can add a package to the locker and remove a package from the locker. The UML representation of a class is shown below:



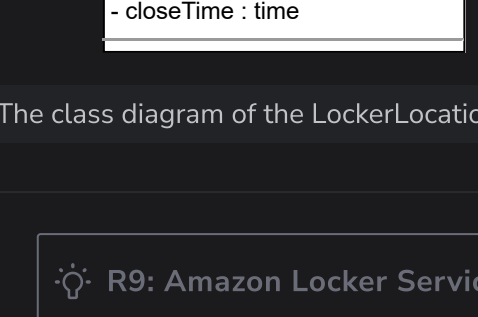
The class diagram of the Locker class

R4: Amazon Locker Service

R4: The locker is assigned to the customer based on the size of the locker.

Locker location

A locker is kept at the location. Since a location may contain more than one locker, we have a list of lockers in the `LockerLocation` class. The `longitude` and `latitude` are used to store the location. According to the requirements, the specific locker is opened only for a particular period of time and the customer can only get a package from the locker if they visit the locker during the location timing. The `openTime` and `closeTime` variables store information about the timings applicable to the particular locker location.The representation of this class is given below:



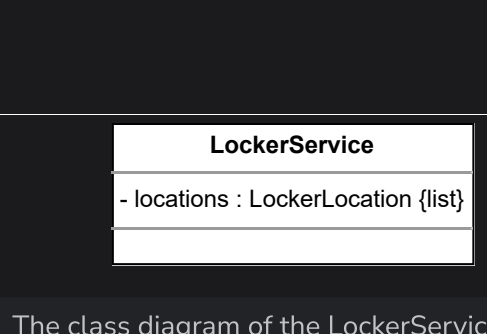
The class diagram of the LockerLocation class

R9: Amazon Locker Service

R9: Amazon Locker Service

Locker service

The `LockerService` class is the main class of the Amazon Locker service system and contains a reference to the list of locker locations. The UML representation of the class is given below:



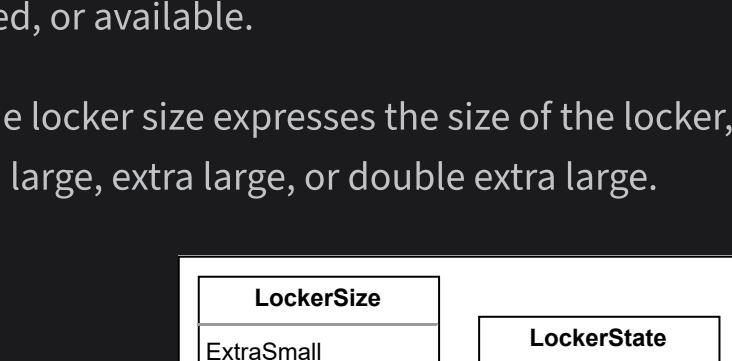
The class diagram of the LockerService class

Enumerations

The list of enumerations required in the Amazon Locker service is provided below:

`LockerStatus`: The locker status describes the current status of the locker, whether it is closed, booked, or available.

`LockerSize`: The locker size expresses the size of the locker, whether it is extra small, small, medium, large, extra large, or double extra large.



Enums in the Amazon Locker service

R3, and R13: Amazon Locker Service

R3: There can be different sizes of lockers like extra small, small, medium, large, extra large, and double extra large.

R13: When the customer picks up the order package from the locker, the locker's state is changed to closed and the customer will no longer be able to open the locker with the given code.

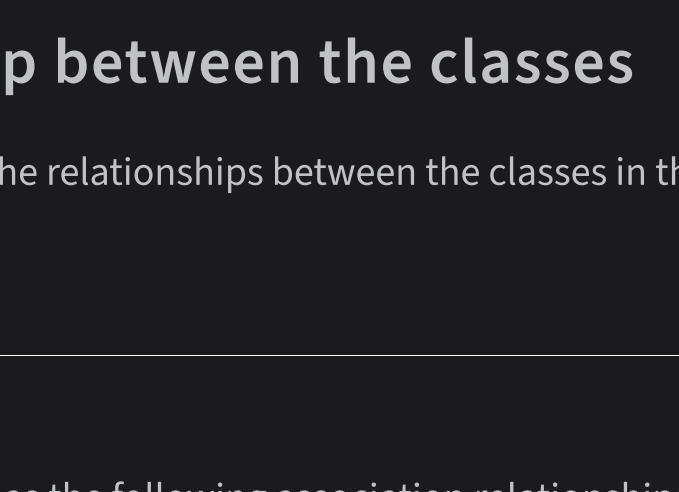
Relationship between the classes

Now, we'll discuss the relationships between the classes in the Amazon Locker service.

Association

The class diagram has the following association relationships:

- The `Notification` class has a two-way association with the `Locker` and `Order` classes.

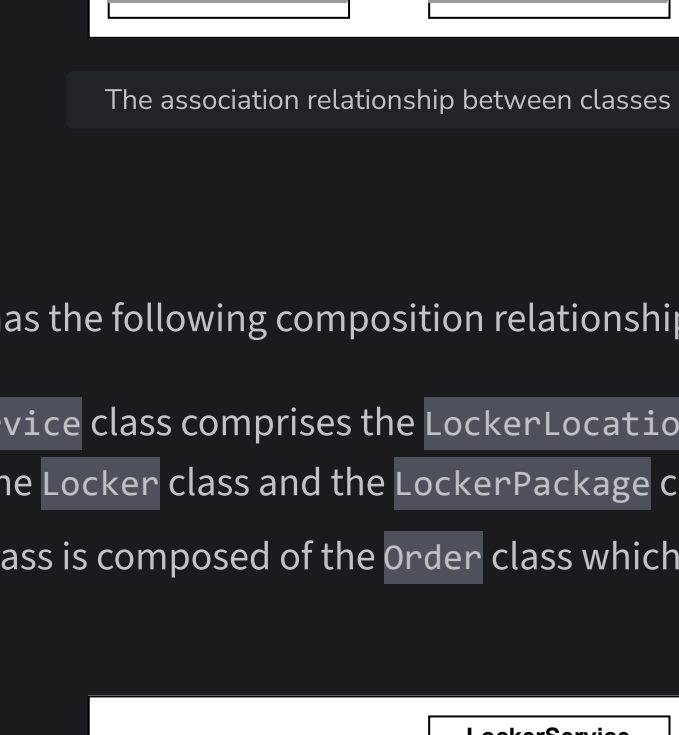


The association relationship between classes

Composition

The class diagram has the following composition relationships:

- The `LockerService` class comprises the `LockerLocation` class, which itself is composed of the `Locker` class and the `LockerPackage` class.
- The `Package` class is composed of the `Order` class which is composed of the `Item` class.



The composition relationship between classes

Inheritance

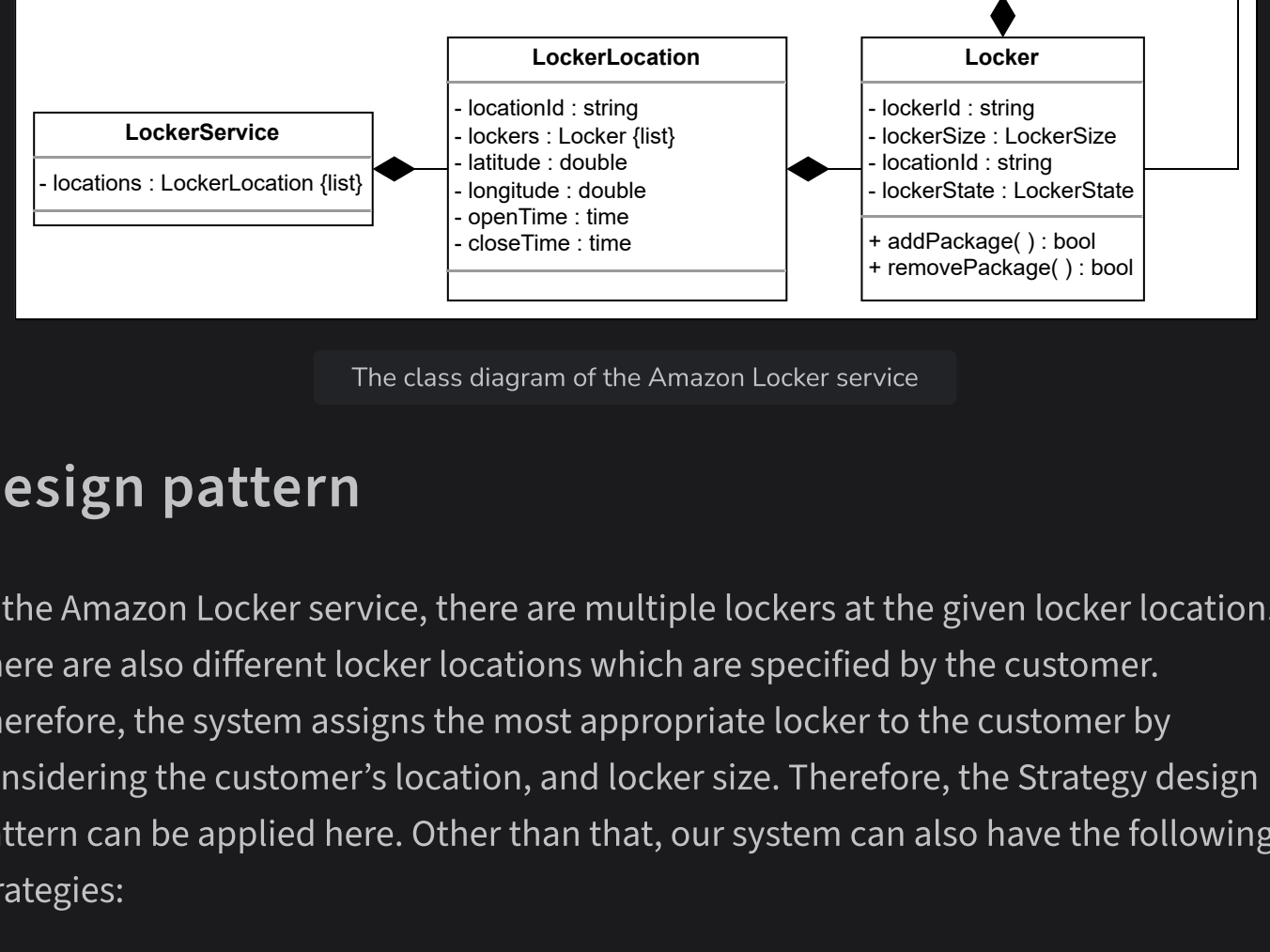
The following classes show an inheritance relationship:

- The `LockerPackage` class extends the `Package` class.

> **Note:** We have already discussed the inheritance relationship between classes in the component section above.

Class diagram of the Amazon Locker service

Here's the complete class diagram for our Amazon Locker service:



The class diagram of the Amazon Locker service

Design pattern

In the Amazon Locker service, there are multiple lockers at the given locker location. There are also different locker locations which are specified by the customer. Therefore, the system assigns the most appropriate locker to the customer by considering the customer's location, and locker size. Therefore, the Strategy design pattern can be applied here. Other than that, our system can also have the following strategies:

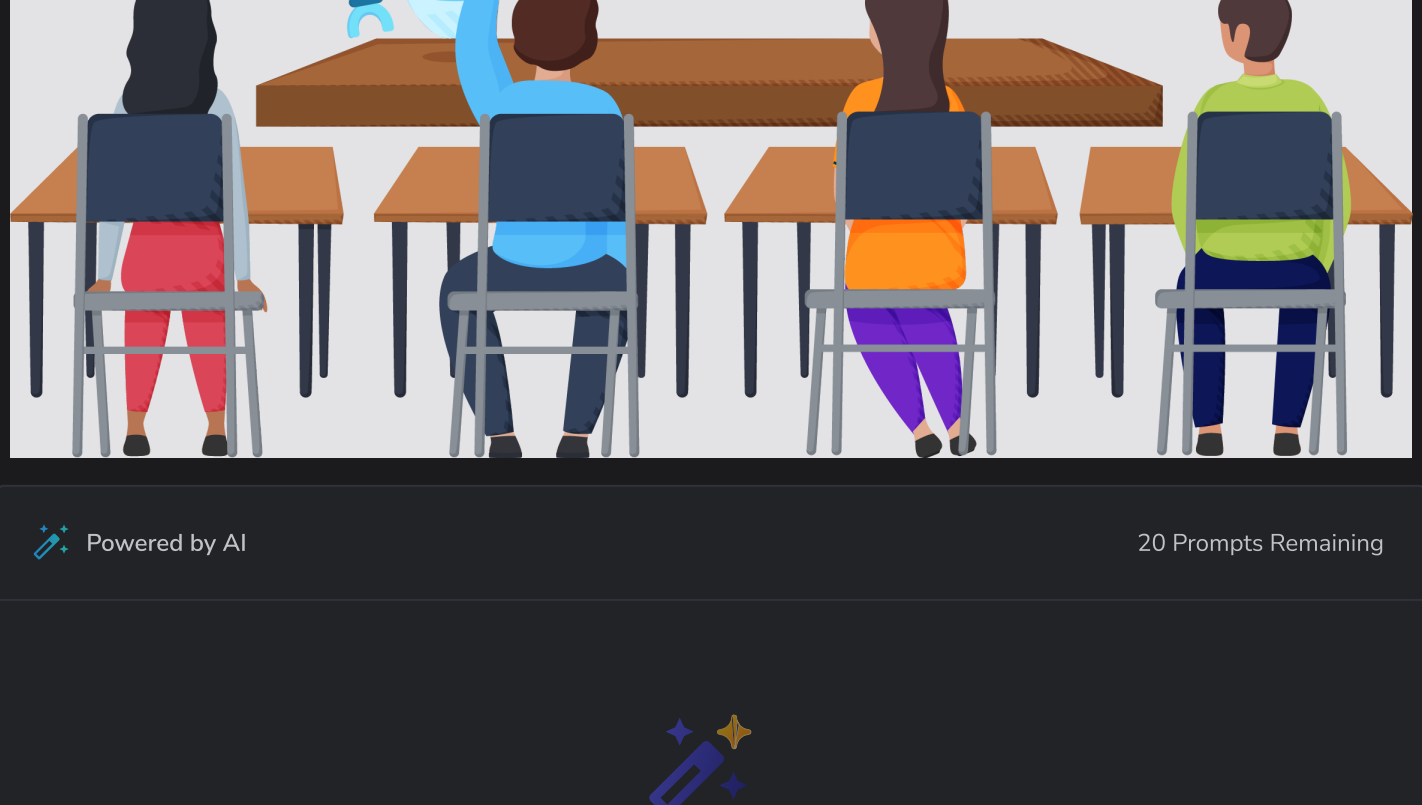
- OTP generation
- Random number generation
- Locker assignment
- Locker filtration

We can also use the Repository design pattern for the Amazon Locker system, where we can make the following repositories:

- Locker repository
- Package repository

AI-powered trainer

At this stage, everything should be clear. If you encounter any confusion or ambiguity, feel free to utilize the interactive AI-enabled widget below to seek clarification. This tool is designed to assist you in strengthening your understanding of the concepts.



We have completed the class diagram of the Amazon Locker service according to the requirements. Now, let's design its sequence diagram in the next lesson.