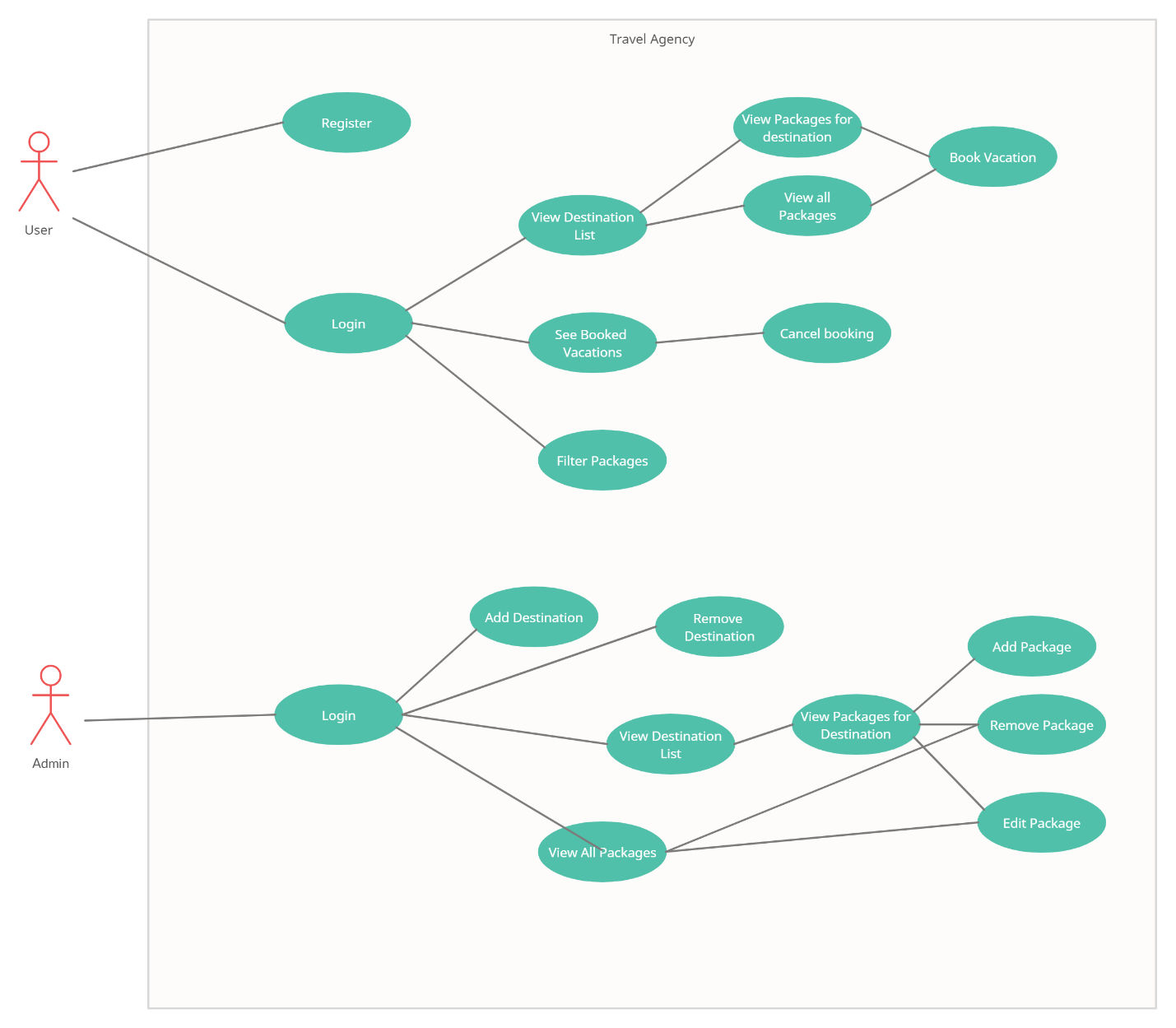
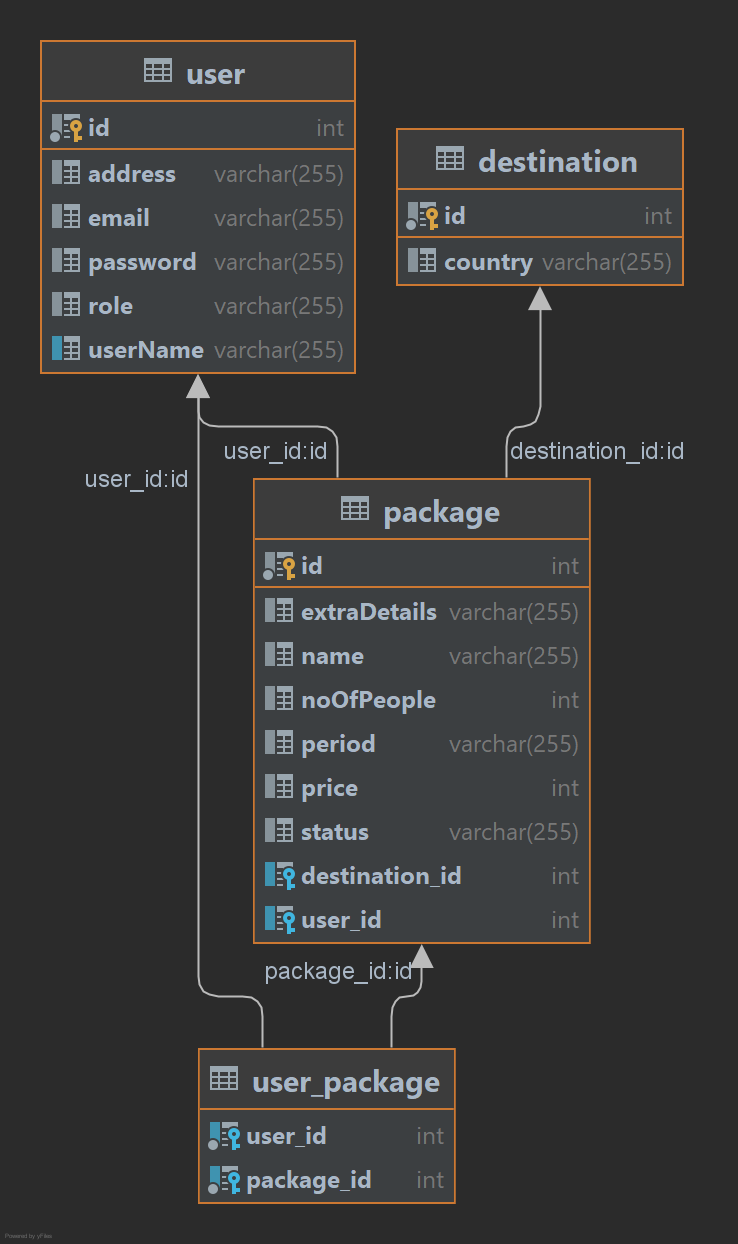
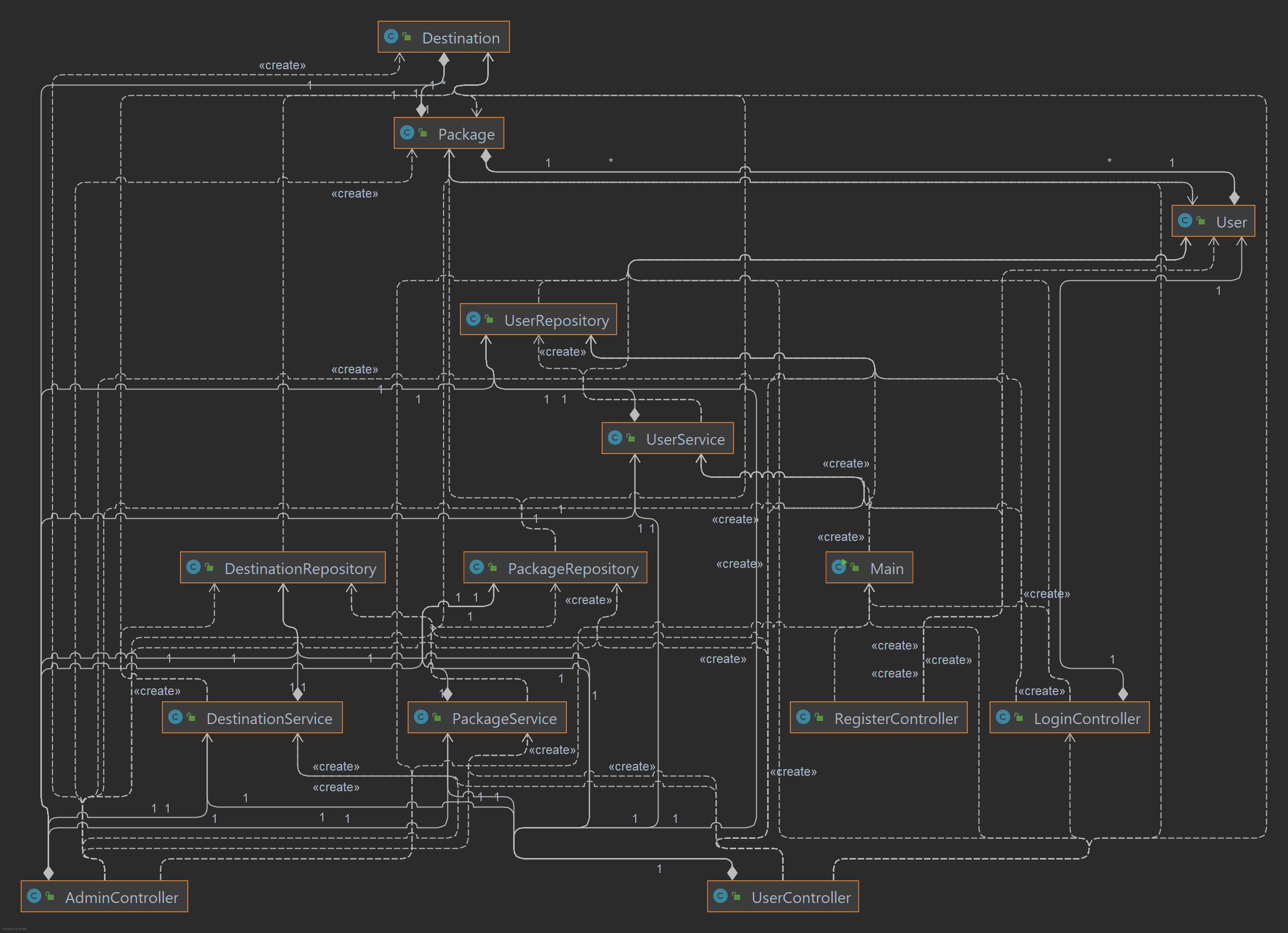
**Travel Agency**

<https://github.com/BodeaRazvan/TravelAgency>

**Use Case Diagram** 

**Database Diagram**

****

Class Diagram

Some main ideas for the relation between classes:

A user has a list of packages, and a package has a list of users (many to many)

A destination has a list of packages (one to many)

The login controller holds a user that is currently logged in (as a static user) and the user controller has access to it

All the controllers have a class of type Repository and one of type Service with their respective name (User controller -> user service -> user repository)

Layers:

**Presentation layer:**

Topmost level, call services, controllers

AdminController



LoginController

RegisterController

UserController

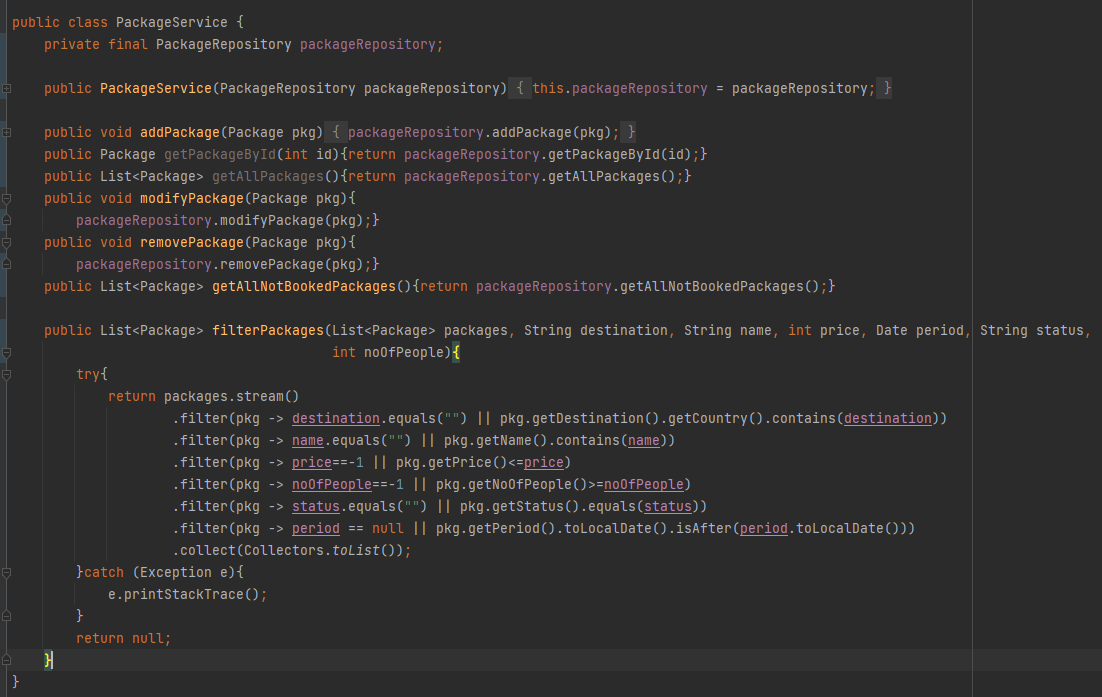
**Business Layer**

Services (logic, functionality)

DestinationService

UserService

PackageService



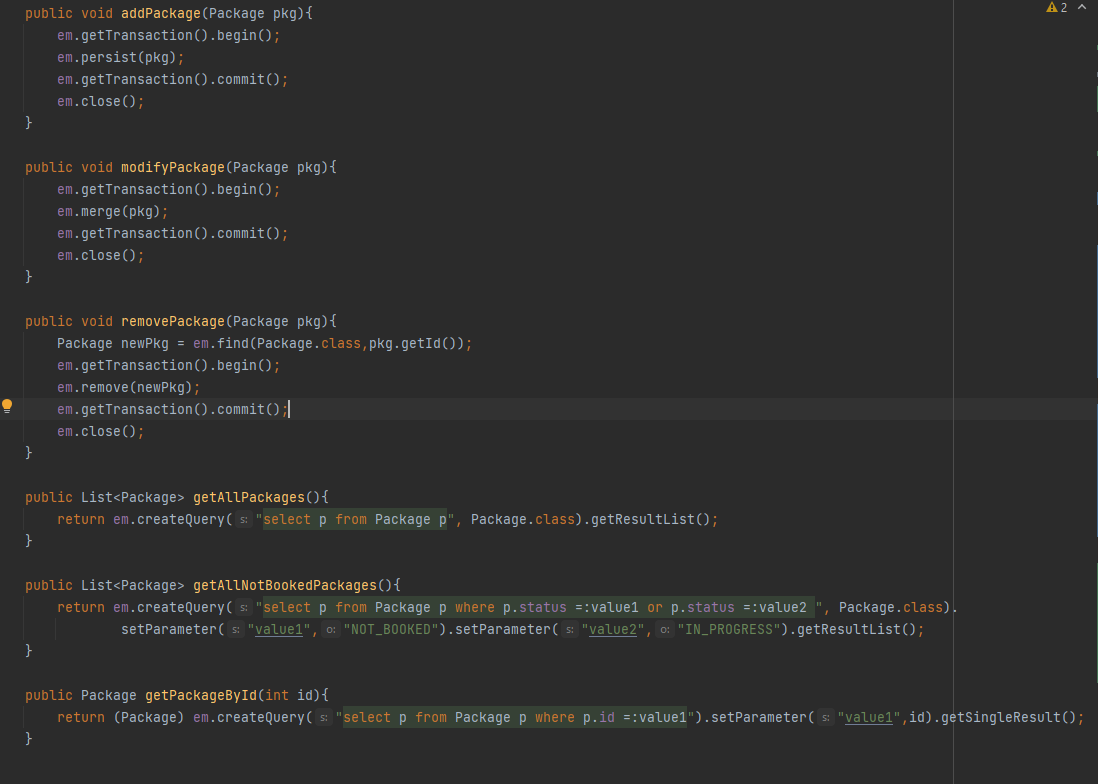
**Persistence Layer**

Repositories (Intermediate layer between business logic and actual data)

DestinationRepository

UserRepository

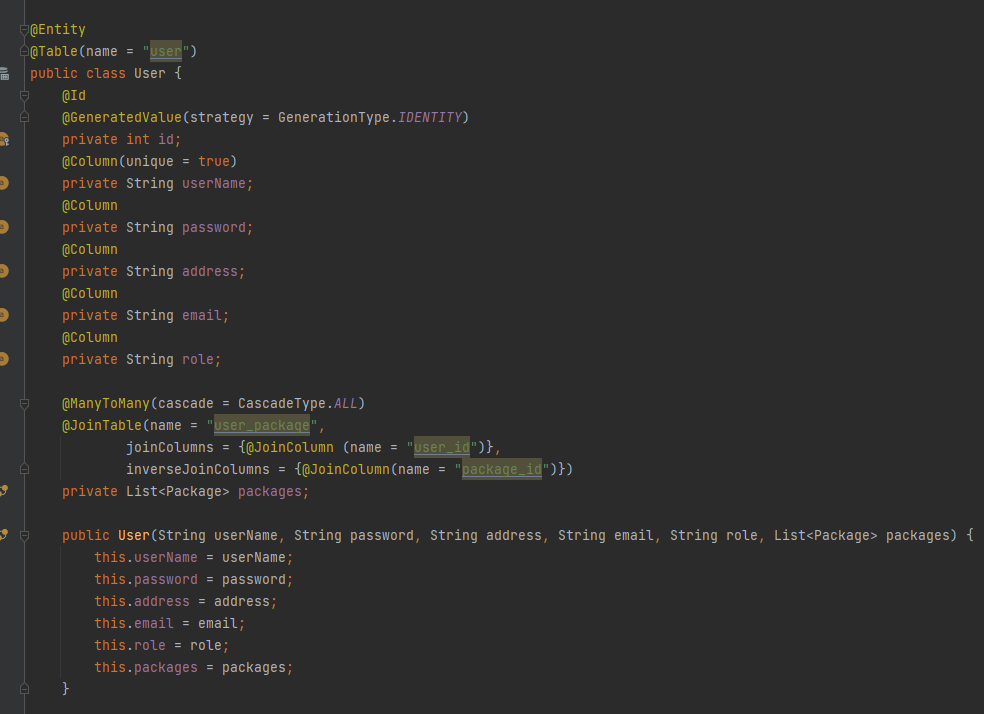
PackageRepository



**Database Layer**

Actual data stored in the database

User



Package

Destination