Project Structure and Explanation of Entity Code Files

Initial Menus (Entry port of the system - main method):

The menu package contains one Java class file that is responsible for three initial menus and for the authentication method.

1. Newsagent and Delivery Person Menu:

- Choose who is accessing the system: Newsagent or Delivery Person.

2. Newsagent Entities menu:

- Enter Newsagent name and password to Login.
- Then Newsagent you will be able to access all entities CRUD menu.

3. Delivery Person Entity menu:

- Enter Delivery Person name and password to Login.
- Then Delivery Person you will be able to access only one entity that is Delivery Docket and for this entity will be available only RU actions (Read and Update).

Classes and Their Roles for Each Entity

Customer Profile Entity as an Example:

Inside each Entity package we have 4 files.

1. CustomerProfile (Class):

- Represents the business entity Customer Profile with attributes: name, postcode, phone number, email, and payment status.
- Handles validation logic for its attributes using regex and ensures data integrity before it's persisted in the database.
- Acts as a blueprint for objects that represent customer data.

2. CustomerProfileCommandLine Class (User Interface Layer):

- Implements CommandLinesExecution, making it a console-based menu interface for interacting with the CustomerProfileCRUD class.
- Role:
  - Displays a menu of options (e.g., Create, Read, Update, Delete).
  - Collects user input for customer attributes.
  - Calls appropriate methods from CustomerProfileCRUD to perform the requested operations.
  - Example:
    If a user chooses "1. Create Customer Account," it collects the necessary data, validates it via CustomerProfile, and persists it using CustomerProfileCRUD.

3. CustomerProfileCRUD Class (Data Access Object):

- Provides methods to perform CRUD (Create, Read, Update, Delete) operations on the customer_profile table in the database.
- Uses EntitiesMySQLAccess to establish the connection.
- Translates Java objects (CustomerProfile) into SQL queries and vice versa.
- Example:
  - Create: Saves a CustomerProfile object into the database.
  - Read: Retrieves all or specific customer profiles as ResultSet.
  - Update: Updates the details of a customer record by ID.
  - Delete: Deletes records based on the ID or deletes all records when instructed.

4. Customer Profile Attributes Test Class (jUnit Tests):

- Each entity has in its package a class with validation and invalidation tests for each attribute that belongs to the entity.
- Each test calls the method related to the attribute being tested.
- The methods responsible for evaluating the parameter being passed through the tests are found in the main class of the entity. In this case example (CustomerProfile class).

For All Entities:

This is a package that contains 3 files that are shared by all entities.

1. EntitiesExceptionHandler (Custom Exception Handler):

- Captures and manages exceptions specific to the system.
- Provides a way to customise error messages and ensure they are meaningful for users or developers debugging the application.
- Used extensively in CustomerProfile validations to handle input errors.

2. EntitiesMySQLAccess (Database Access Layer):

- Manages the database connection.
- Provides a reusable connection mechanism for database-related operations across the system for all entities.
- Abstracts the connection details such as the host, user, and password.

3. CommandLinesExecution (Interface):

- Serves as a blueprint for command-line execution classes.
- Ensures any implementing class (like CustomerProfileCommandLine) provides an execute() method, which serves as the entry point for the command-line program.

Data Flow / Integration:

1. User Interaction:
    - A user interacts with the CustomerProfileCommandLine menu via the console.
    - Their choice determines the operation (e.g., create, read, update, delete).

2. Input Handling and Validation:
    - User inputs are captured, and a CustomerProfile object is created.
    - The object's attributes are validated within the CustomerProfile class (e.g., name, postcode) to ensure correctness.

3. Database Interaction:
    - If validation passes, the operation proceeds.
    - The CustomerProfileCRUD class translates the operation into SQL queries and uses EntitiesMySQLAccess to interact with the database.

4. Error Handling:
    - Any validation failure or database error triggers an EntitiesExceptionHandler, ensuring a smooth user experience with clear error messages.

5. Output Handling:
    - Results of operations (e.g., "Customer Details Saved" or table data) are displayed to the user via the CustomerProfileCommandLine interface.

System Workflow Example:

Newsagent System Steps Access:

1. From the first menu, choose option 1 (Newsagent) .
2. Newsagent enter Name and Password.
3. If it is successful authenticated then Newsagent is redirected to All entities menu.
4. Newsagent choose option 1 (Customer Profile), from the menu.
5. Newsagent redirected to the Customer Profile CRUD menu.
6. User selects 1. Create Customer Account" from the console menu.
7. CustomerProfileCommandLine collects inputs for name, postcode, phone number, email, and payment status.
8. A CustomerProfile object is created, and its attributes are validated.
9. If valid, CustomerProfileCRUD.createCustomerDetailsAccount() is called.
10. SQL is executed via EntitiesMySQLAccess, and the record is inserted into the Software_Project_NewsCompany database and customer_profile table.

Note:

Please, for testing purposes we have provided inside the Integrated_Database_Tables folder the SQL code for creation of the database such as for all entities tables.