Backend Exam

Task Description:

You are developing a banking system application that follows the clean architecture principles. The system contains the following three layers: Domain, Use Case, and Infrastructure.

The Domain layer consists of the entities representing domain concepts such as Account, Transaction, and Customer. These entities have methods for performing specific operations related to their respective domains.

The Use Case layer contains the business logic of the application. It includes use cases like creating a new account, making a transaction, and generating account statements.

The Infrastructure layer deals with the interaction between the application and the outside world. It includes repositories for persisting data, external service integrations, and data access objects.

Your task is to implement a simplified version of the banking system application based on clean architecture. You need to create the necessary classes and methods following the clean architecture principles.

Requirements:

1. Implement the Account entity class with the following attributes: account_id, customer_id, account_number, balance.
    a. It should have the following methods: deposit(), withdraw(), and get_balance().
2. Implement the Customer entity class with the following attributes: customer_id, name, email, and phone_number.
3. Implement a Use Case class for creating a new account. It should have a method named create_account() that takes customer_id, name, email, and phone_number as input and returns an Account object.
4. Implement a Use Case class for making a transaction. It should have a method named make_transaction() that takes account_id, amount, and transaction_type (either 'deposit' or 'withdraw') as input and updates the account balance accordingly.
5. Implement a Use Case class for generating account statements. It should have a method named generate_account_statement() that takes account_id as input and returns a statement string containing transaction details for the given account.
6. Implement an Infrastructure class named AccountRepository for persisting and retrieving account data. It should have methods like save_account(), find_account_by_id(), and find_accounts_by_customer_id().
7. Implement a simple test scenario that demonstrates the use of all the implemented classes and methods.

Notes:

- You don't need to implement any specific data storage mechanisms. Just focus on the class design and method implementations as per clean architecture principles.
- You can assume the availability of any required external libraries or modules.
- You should provide the complete implementation for the problem, including the necessary classes and methods.