Q1.

HTTP Method	URL	Description
POST	/api/auth/login	authenticate users
GET	/api/inventory/medications	query existing medicines
POST	/api/inventory/medications	insert new medication records
GET	/api/inventory/medications/{medication_id}	retrieve details of a specific medication
PUT	/api/inventory/medications/{medication_id}	update existing medication records
PATCH	/api/inventory/medications/{medication_id}	soft delete medication records
DELETE	/api/inventory/medications/{medication_id}	permanently delete medication records
POST	/api/inventory/medications/{medication_id}/adjust	adjust the quantity of a specific medication
GET	/api/customers	query existing customer records
POST	/api/customers	insert new customer records
GET	/api/customers/{customer_id}	retrieve details of a specific customer
PUT	/api/customers/{customer_id}	update existing customer records
PATCH	/api/customers/{customer_id}	soft delete customer records
DELETE	/api/customers/{customer_id}	permanently delete customer records
GET	/api/users	retrieve a list of all users with their roles and permissions
POST	/api/users	create a new user account
PUT	/api/users	update user details and roles

DELETE	/api/users	deactivate a user account
GET	/api/users/{user_id}/roles	retrieve the role of a specific use
PUT	/api/users/{user_id}/roles	update the role of a specific user

Q2

- User Class:
    - Attributes:
        - name: String
        - username: String
        - password: String
        - role: String

The User class represents users of the system, such as owners, managers, and cashiers.
It has private attributes name, username, password, and role, which store the user's name, username, password, and role, respectively.

- Medication Class:
    - Attributes:
        - name: String
        - description: String
        - quantity: Integer

The Medication class represents the inventory of medications available in the pharmacy.
It has private attributes name, description, and quantity, which store the medication's name, description, and quantity, respectively.

- Customer Class:
    - Attributes:
        - name: String
        - email: String
        - phone: String

The Customer class represents customer records maintained by the pharmacy.
It has private attributes name, email, and phone, which store the customer's name, email address, and phone number, respectively.

Schema is provided in the GitHub

Q3

Name of the package	What would you accomplish using that?
Express.js	Web Framework, Error Handling, and Validation
Sequelize	ORM
Passport.js	Authentication
MySQL	Database
Jest	Unit Testing
npm	Managing dependencies and installing Node.js packages

Q4

Name of the program	What would you accomplish using that?
Visual Studio Code	Code Editor/IDE
MySQL Workbench	Database Management
Postman	API Testing

Q5

1. Identify roles and permissions:

Determine the different roles users will have (e.g., owner, manager, cashier) and what actions each role should be allowed to perform (e.g., add, edit, delete).

2. Assign roles to users:

When creating user accounts, assign each user a role based on their job function and responsibilities.

3. Protect your application:

Implement checks in your code to ensure that only users with the right roles can access specific features or data.

For example, if a cashier tries to delete a medication record, the application should deny the request because cashiers don't have permission to perform that action.

4. Handle unauthorized access:

When someone tries to do something they're not allowed to do, make sure the application responds appropriately. It could be by showing an error message.

5. Test your permissions:

Make sure to test your application thoroughly to ensure that users can only do what they're supposed to based on their roles.

6. Review and Update:

Regularly review your roles and permissions to make sure they still make sense for the application.

Update them as needed based on changes in the business requirements or security needs.