* Сделать REST с CRUD операциями
* Использовать сервлеты
* Использовать JDBC. Spring и Hibernate использовать нельзя
* Должно быть хорошее разделение по пакетам и слоям, как минимум должны присутствовать слои: controller/servlet, service, repository, mapper, dto, entity.
* Покрыть логику юнит тестами
* Используем Lombok, кроме переопределения equals и hashcode
* Должны быть реализованы связи One-to-One, One-to-Many
* База Postgreslq
* Руководствоваться принципами SOLID
* Проверяйте REST API через Postman

Мне надо будет скинуть репозиторий и запись экрана, где вы дергаете все CRUD методы через Postman. При необходимости, будьте готовы зашэрить экран и показать функционал на занятии.

Дополнительно (за 3 доп баллов)

* Установите Jenkins на виртуалку
* Законектите Jenkins и гит, чтобы после каждого коммита в мастер ветку Jenkins интсаллил приложение в томкат.
* Мне надо будет просто скинуть запись экрана с пушем в мастер и триггером новой сборки, или просто как UI изменяется при пуше коммита.

Создание RESTful API с операциями CRUD с использованием сервлетов и JDBC включает в себя несколько шагов. Давайте пройдемся по процессу шаг за шагом:

1. Структура проекта:

- Создайте пакеты controller, service, repository, mapper, dto, and entity.

- Пакет entity будет содержать классы POJO, представляющие объекты данных.

- Пакет dto будет содержать объекты передачи данных, используемые для обмена данными между слоями.

- Пакет mapper будет содержать классы mapper, ответственные за сопоставление DTO с объектами и наоборот.

- Пакет репозитория будет содержать классы, ответственные за операции с базой данных с использованием JDBC.

- Пакет сервиса будет содержать бизнес-логику и операции обработки данных.

- Пакет контроллера будет содержать сервлеты для обработки входящих HTTP-запросов.

2. Конфигурация базы данных:

- Настройте свою базу данных и создайте необходимые таблицы для сущностей, с которыми вы будете работать.

3. Определите классы сущностей:

- Создайте классы Java в пакете entity для представления сущностей в вашей системе.

4. Создайте DTO:

- Создайте классы объектов передачи данных (DTO) в пакете dto для передачи данных между слоями.

5. Реализуйте уровень репозитория:

- Создайте классы в пакете репозитория для обработки операций с базой данных с использованием JDBC. Эти классы будут отвечать за операции CRUD.

6. Реализуйте классы отображения:

- Создайте классы mapper в пакете mapper для сопоставления сущностей с DTO и наоборот.

7. Реализуйте уровень сервиса:

- Создайте классы в пакете service, которые будут содержать бизнес-логику и выполнять операции обработки данных.

8. Создайте сервлеты:

- Реализуйте сервлеты в пакете контроллера для обработки входящих HTTP-запросов.

- Сопоставьте сервлеты с соответствующими конечными точками.

9. Реализуйте операции CRUD:

- В сервлетах обрабатывайте HTTP-запросы (POST, GET, PUT, DELETE) и делегируйте операции сервисному уровню.

- Уровень сервиса выполнит проверку, вызовет уровень хранилища для операций с базой данных и вернет соответствующий ответ.

10. Обработка исключений:

- Реализуйте надлежащую обработку исключений в сервлетах и на уровне сервиса для сценариев ошибок.

11. Развертывание:

- Разверните проект в контейнере сервлетов, таком как Apache Tomcat.

Это высокоуровневый обзор шагов, связанных с созданием RESTful API с операциями CRUD с использованием сервлетов и JDBC. Каждый шаг включает в себя детальную реализацию, такую как написание кода для сервлетов, операции JDBC, сопоставление и обработка исключений.

Creating a RESTful API with CRUD operations using servlets and JDBC involves several steps. Let's walk through the process step by step:

1. Project Structure:

- Create packages for controller, service, repository, mapper, dto, and entity.

- The entity package will contain the POJO classes representing the data entities.

- The dto package will contain Data Transfer Objects used for data exchange between layers.

- The mapper package will contain mapper classes responsible for mapping DTOs to entities and vice versa.

- The repository package will contain classes responsible for database operations using JDBC.

- The service package will contain the business logic and data processing operations.

- The controller package will contain servlets for handling incoming HTTP requests.

2. Database Configuration:

- Set up your database and create the necessary tables for the entities you will be working with.

3. Define Entity Classes:

- Create Java classes in the entity package to represent the entities in your system.

4. Create DTOs:

- Create Data Transfer Object (DTO) classes in the dto package to transfer data between layers.

5. Implement Repository Layer:

- Create classes in the repository package to handle database operations using JDBC. These classes will be responsible for CRUD operations.

6. Implement Mapper Classes:

- Create mapper classes in the mapper package to map entities to DTOs and vice versa.

7. Implement Service Layer:

- Create classes in the service package to contain the business logic and perform data processing operations.

8. Create Servlets:

- Implement servlets in the controller package to handle incoming HTTP requests.

- Map the servlets to the appropriate endpoints.

9. Implement CRUD Operations:

- In the servlets, handle HTTP requests (POST, GET, PUT, DELETE) and delegate the operations to the service layer.

- The service layer will perform validation, call the repository layer for database operations, and return the appropriate response.

10. Handle Exceptions:

- Implement proper exception handling in the servlets and service layer for error scenarios.

11. Deployment:

- Deploy the project to a servlet container such as Apache Tomcat.

This is a high-level overview of the steps involved in creating a RESTful API with CRUD operations using servlets and JDBC. Each step involves detailed implementation, such as writing code for servlets, JDBC operations, mapping, and exception handling.