Задание: Hibernate

Навигация

Описание

В качестве домашнего задания вам нужно адаптировать приложение, которое было написано в модуле `Spring Core` на работу с БД, чтобы все данные после перезапуска приложения не терялись и их можно было достать из БД.

Вам нужно будет адаптировать классы `Account` и User под `Hibernate` сущности. Так же необходимо будет учесть связь `ManyToOne` между ними.

Цель задания – научиться интегрировать Hibernate в Spring-приложение для работы с объектно-реляционным отображением (ORM).

Знания для выполнения домашней работы

Для успешного выполнения задания, вам потребуются следующие знания:

- Все знания с домашнего задания по `Spring Core`
- Базовые знания `SQL` и реляционных баз данных
  - Операторы `SELECT`, `INSERT`, `JOIN`, `WHERE`
  - Понимание транзакций, уровней изоляции
- Основы Hibernate
  - Аннотации `@Entity`, `@Column`, `@OneToMany`, `@ManyToOne`
  - Работа с сессиями

Библиотеки разрешенные для использования

В этой работе должен использоваться только чистый hibernate и spring core, без авто-конфигураций
Зависимости, которые вы можете использовать у себя в проекте при разработке

```
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>6.1.5</version> тут может быть более новая версия
        </dependency>
        <dependency>
            <groupId>jakarta.annotation</groupId>
            <artifactId>jakarta.annotation-api</artifactId>
```

```
            <version>2.1.1</version> тут может быть более новая версия
        </dependency>
        <dependency>
            <groupId>org.hibernate.orm</groupId>
            <artifactId>hibernate-core</artifactId>
            <version>6.4.4.Final</version> тут может быть более новая версия
        </dependency>
        <dependency>
            <groupId>org.postgresql</groupId>
            <artifactId>postgresql</artifactId>
            <version>42.7.3</version> тут может быть более новая версия
        </dependency>
```

Задание на разработку

Постановка на разработку

1. **Перевод моделей на Hibernate Entities**:
   - Переведите классы `User` и `Account` на Hibernate Entities, используя соответствующие аннотации для маппинга на таблицы базы данных.
2. **Реализация сохранения сущностей в базу данных**:
   - Обновите сервисы `UserService` и `AccountService` для использования работы с БД.
     - Работа с БД должна быть реализована с помощью `SessionFactory`.
     - Реализуйте методы для сохранения и получения данных из базы данных.

Детали реализации:

- **Настройка Hibernate**:
  - Создайте файл конфигурации Hibernate (`HibernateConfiguration.java`). В этом файле укажите параметры подключения к базе данных (URL, имя пользователя, пароль) и настройки Hibernate (диалект базы данных, автоматическое создание/обновление схемы базы данных, контекст сессии).
  - Добавьте аннотированные классы (сущности) в конфигурацию, чтобы Hibernate мог их распознавать и маппить на таблицы базы данных.
- **Транзакции**:
  - Убедитесь, что все операции с базой данных выполняются в рамках транзакций. Это необходимо для обеспечения целостности данных и возможности отката изменений в случае ошибок.
  - Также предусмотрите вариант выполнения, когда один метод в транзакции вызывает другой транзакционный метод.
    В этом случае второй метод должен увидеть существующую транзакцию и не закрывать ее по окончанию работы. Эту транзакцию должен закрыть тот метод, который ее открыл.
- **Логирование запросов Hibernate**:
  - Для тестирования и проверки запросов включите логирование SQL-запросов, отправляемых Hibernate в базу данных. Это можно сделать, добавив в настройки Hibernate параметры `hibernate.show_sql` и `hibernate.format_sql`, которые позволяют видеть форматированные SQL-запросы в логах приложения.
  - Это поможет отладить запросы и убедиться, что они выполняются корректно.

Как протестировать

- Запустите локально на своей машине базу данных через `Docker`
- Протестируйте основные методы приложения, такие как: создание пользователей, аккаунтов, перевод, зачисление денег

- Перезапустите приложение и проверьте, что все данные сохранены и возвращаются даже после перезапуска
- Измените метод `ACCOUNT_TRANSFE` так, чтобы в самом конце он выбрасывал исключение. После проведения перевода, но при этом еще в транзакции. Проверьте, что изменения откатываются из БД, перевод не должен произойти.

Помните, что хорошее тестирование включает в себя проверку не только "счастливых путей", но и граничных условий и ситуаций, когда что-то идет не так.