Необходимо разработать SOAP backend для веб-приложения. Основная задача бекенда - управление пользователями и их ролями. Описание модели данных:

• У пользователя может быть несколько ролей, одна роль может быть у нескольких пользователей. Например, Вася - Админ и Оператор, Петя - Оператор и Аналитик.

• Атрибуты пользователя - Имя, Логин (первичный ключ), Пароль (шифровать пароль в рамках тестового задания не требуется, это просто строка).

• Атрибуты роли – id (первичный ключ), Имя.

**Необходимо:**

1 Разработать SOAP сервис и методы работы с данными. Сделать методы, которые будут:

1 Получать список пользователей из БД (без ролей)

2 Получать конкретного пользователя (с его ролями) из БД

3 Удалять пользователя в БД

4 Добавлять нового пользователя с ролями в БД.

5 Редактировать существующего пользователя в БД. Если в запросе на редактирование передан массив ролей, система должна обновить список ролей пользователя в БД - новые привязки добавить, неактуальные привязки удалить.

2 На бекенде для методов добавления и редактирования должен производиться формато-логический контроль пришедших значений. Поля name, login, password - обязательные для заполнения, password содержит букву в заглавном регистре и цифру.

• Если все проверки пройдены успешно, ответ должен содержать <success>true</success>

• Если случилась ошибка валидации, ответ должен содержать <success>false</success><errors>массив ошибок</errors>

**Решение должно быть**

• реализовано на языке Java, на платформе Spring Framework / Springboot, использовать Spring Data или Hibernate в качестве ORM

• сервисы должны работать по протоколу SOAP, формат XML (это важно, не нужно делать REST/JSON)

• задание должно быть отправлено в формате zip архива с исходным кодом в ответном письме

• базу данных для задачи можно выбрать любую и создать любым способом, на ваше усмотрение - автосоздание таблиц hibernate, скрипт создания create\_schema.sql, допускается использовать in-memory БД

**Как задание оценивается**

• Оптимальность кода - самый важный критерий

• Соответствие общепринятым best practices

• Отсутствие ошибок - в программе не должно быть ошибок, она должна работать

• Полнота реализации – ожидаем полностью выполненное задание, по решению даем обратную связь