Требования по технологиям
BE: Spring Boot, Spring Data Jpa (in memory db, auto fill), Spring Security, Spring Mvc, Gradle/ Maven, jUnit
FE: Angular/React/Vue.js (typescript), Bootstrap/Materialize - Опционально, возможно использовать любые удобные технологии)

Задание
Создать веб приложение “Конференция”.
Пользователь должен иметь возможность регистрироваться на конференции и просматривать доклады. Докладчики могут создавать доклады и выбирать время в аудитории для доклада. Необходимо предусмотреть, чтобы доклады не перекрывали друг друга по времени в 1 аудиториях, т.е. не может быть 2 доклада в аудитории в одно время. Расписание также должно быть доступно по REST API.

Приложение должно содержать минимальный набор таблиц. Набор полей можно использовать по своему усмотрению. Таблицы:

- User - содержит данные о пользователях, пароль (зашифрованный), роль и тд.
- Presentation - Содержит данные о презентация. Один пользователь может иметь много презентаций, и у одной презентации может быть много презентаторов.
- Schedule - расписание, в какой аудитории, когда и какая презентация проходит
- Room - аудитория, где проходит презентация

Роли:
Admin, Presenter, Listener

Страницы:

- главная - содержит расписание презентаций, разбитое по аудиториям.
- регистрация - только слушатели могут регистрироваться. Делать подтверждение регистрации не нужно.
- список докладов - доступна докладчику. отображает список докладов для докладчика с возможностями CRUD. Также на странице необходимо реализовать выбор времени доклада в расписании.
- список пользователей - доступна только администратору с возможностями CRUD. Только администратор может делать из пользователей докладчиков.

Страницу CRUD для ROOM можно не создавать, достаточно заполнить её при старте приложения

Дополнительные требования:
Данные заполнять через скрипт при старте приложения.
Для приложения использовать embedded tomcat.
Создать REST API которое в ответе будет содержать расписание, разбитое по аудиториям.

Покрыть тестами сервисы и контролеры - не обязательно покрывать все возможные кейсы, достаточно пару (2-3) теста для примера.

Не обязательно:

- После регистрации предусмотреть асинхронную отправку письма, подтверждения. Можно использовать любой ящик.
- Поддержать сборку в docker и деплой используя docker-compose
- Осуществлять сборку FE в отдельный docker контейнер с сервером nginx или npm
- FE часть. В случае отсутствия FE, необходимо предоставить механизм тестирования API, покрывающий все процессы, например https://learning.postman.com/docs/writing-scripts/test-scripts/

Время выполнения работы: 8 дней.
Подсказка: выполнять задание проще с использованием Spring Boot и авто конфигурации. Можно начать с http://start.spring.io/.

PS
Это тестовое задание и нет надобности доводить всё до полной готовности, достаточно создать рабочий каркас. Оценивается умение использовать готовые решения.