**Stack:**

Java 11

Spring Boot (MVC, Security, REST)

Hibernate + h2 (обязательно in-memory database);

ReactJs or Angular + Thymeleaf, можно использовать любые библиотеки для стилей.

JUnit (Mockito, DbUnit по желанию);

Для тестирования отправки сообщений, в качестве адресата использовать YOPmail (<https://yopmail.com/en/>). В качестве отправителя использовать **фейковую** gmail почту, сняв дополнительные настройки безопасности (разрешить связь со сторонних приложений)

**Security:**

Лучшим решением будет использование JWT – токена (можно без refresh-token). Можно даже простой токен.

**Unit-тесты:**

Нужно хотя бы по одному тесту для каждого слоя (можно не тратить время на полное покрытие):

- DTO (логика преобразования Entity в DTO и обратно);

- Сервисы (с использованием Mockito или Powermock);

- Утилитарные классы (если есть);

- Контроллеры (org.springframework.test.web.servlet.MockMvc).

Все unit-тесты должны отрабатывать при запуске из консоли командой "mvn test".

**Коммиты в bitbucket:**

Должно быть хотя бы по одному комиту на каждую user-story, чтобы видеть процесс работы над проектом. Можно комитать чаще, нельзя реже.

**Запуск проекта:**

- Бэк-енд должен запускаться из консоли с помощью команды

**java -jar helpdesk.jar;**

- На старте должны автоматически отрабатывать все необходимые скрипты для инициализации БД;

- Должны быть определены стандартные пользователи и пароли, оговоренные в спецификации.

**Readme-файл:**

В корне проекта должен лежать README.md файл с описанием проекта:

• Логины и пароли юзеров

• Запуск проекта (фронт и бэк), порты их поднятия, стартовая страница

• Любая другая полезная информация

**Spring DATA, REST Repositories - использовать НЕЛЬЗЯ.**

**Пагинация, валидация, сортировка, поиск должны быть реализованы на бэке.** (допускается двойная валидация как на бэке так и на фронтенде)

**Front-end:**

Вы можете использовать готовый шаблон, в котором в основном следует навесить запросы на бэк-енд, но вы в целом можете менять стили, верстку, можете переписывать внутреннюю логику компонентов, вплоть до написания с 0 своего фронта (Angular или React).

Самое главное, чтобы итоговая работа не противоречила требования логики проекта.

Поощряется использование дополнительных технологий:

* Lombok
* Docker
* Swagger
* OAuth2
* Gradle