Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Лабораторная работа №2

по предмету

«Сервис ориентированная архитектура»

Вариант 523236

Выполнили:

Бордун Анастасия Владимировна, Иванов Евгений Дмитриевич

Группа:

P34111

Преподаватель:

Кривоносов Егор Дмитриевич

Санкт-Петербург

2023 г.

Задание

На основе разработанной в рамках лабораторной работы #1 спецификации реализовать два веб-сервиса и использующее их API клиентское приложение.

Требования к реализации и развёртыванию сервисов:

- Первый ("вызываемый") веб-сервис должен быть реализован на фреймворке Spring MVC REST и развёрнут в окружении под управлением сервера приложений Tomcat.
- Второй веб-сервис должен быть реализован на фреймворке Spring MVC REST, развёрнут в окружении под управлением сервера приложений WildFly и вызывать REST API первого сервиса.
- **Для обоих сервисов необходимо реализовать все функции, задокументированные в API, в строгом соответствии со спецификацией!**
- Доступ к обоим сервисам должен быть реализован с по протоколу https с самоподписанным сертификатом сервера. Доступ к сервисам посредством http без шифрования должен быть запрещён.

Требования к клиентскому приложению:

- Клиентское приложение может быть написано на любом веб-фреймворке, который можно запустить на сервере helios.
- Приложение должно обеспечить полный набор возможностей, предоставляемых API обоих сервисов -- включая сортировку, фильтрацию и постраничный вывод элементов коллекции.
- Приложение должно преобразовывать передаваемые сервисами данные в человеко-читаемый вид -- параграф текста, таблицу и т.д.
- Клиентское приложение должно информировать пользователя об ошибках, возникающих на стороне сервисов, в частности, о том, что сервису были отправлены невалиданые данные.

Оба веб-сервиса и клиентское приложение должны быть развёрнуты на сервере helios.

Дополнительное задание

Реализовать 3 сервис с интересными ручками на JAX-RS.

Описание

Всего используется 3 сервиса:

- 1 сервис (основной). Spring MVC REST. Tomcat 9.0.80.
- 2 сервис (побочный, обращается к первому). Spring MVC REST. Wildfly 26.1.2.
- 3 сервис (побочный, обращается к первому). JAX-RS. Tomcat 9.0.80.

Почему не новейшие версии? Для поддержки Jersey. Пока что новейшие версии Tomcat не поддерживают новые версии Jersey, так что нужен downgrade. А так как происходит downgrade по Tomcat, то далее нужен downgrade по spring-boot-starter-parent на версии 2.X.X.

Каждый сервис работает по SSL (self signed certificates):

- 1 сервис (orgdirectory.jks)
- 2 сервис (2service.jks)
- 3 сервис (3service.jks)

Как деплоить?

Tomcat

application server config - https://drive.google.com/file/d/1U6iXI9mtK0NXiBR493IrIHhvfSg4hJ-M/view?usp=sharing

webapps_13 - настроенная директория для war файлов проекта. Идем в *conf/server.xml* и заменяем `keystoreFile="<path>"` на путь до соответствующего .jks. Для запуска идем в *bin* и набираем: *catalina.bat run* или *startup.bat* (для Windows).

Wildfly

application server config - https://drive.google.com/file/d/14ZDJ7J8plF9XXCgZXtx2izsLF7Y85W69/view?usp=sharing

Идем в *standalone/configuration/standalone.xml* и заменяем:

```
<key-store name="service2KS">
    <credential-reference clear-text="2service"/>
    <implementation type="JKS"/>
    <file path="<path>"/>
</key-store>
```

`<file path="<path>"/>` на путь до соответствующего .jks. Для запуска идем в bin и запускаем *standalone.bat* (для Windows).

Ручки

https://localhost:9099/first-service/swagger-ui/index.html#/ - 1 сервис

https://127.0.0.1:9100/swagger-ui/index.html#/ - 2 сервис

https://localhost:9101/organalysis/ - 3 сервис

Как деплоить фронтенд?

Весь frontend находится в all-frontend каталоге, для разработки использовать его.

Разработка на node.js так что он должен быть установлен, например с официального сайта

Подгрузка необходимых модулей *node*

npm install

Запуск на локальной машине на порт 5173(или другой)

npm run dev

Процесс сборки и деплоя на хелиос

Запускаем следующую команду, чтобы весь фронт положился в *dist/* каталог

npm run build

После этого там появится *index.html*, который будет ссылаться на прочие вещи файлы, но он ссылается на скрипты как на */assets/...* и из-за этого на хелиосе не происходит нормального чтения этих скриптов, надо поменять путь на *./assets/...* Аналогично для всех файлов, которые использует фронт.

Далее деплоим *index.html* и *assets/* на *public_html* каталог пользователя на helios.

Код программы

https://github.com/Bordsiya/soa/tree/lab-2

Postman запросы

https://github.com/Bordsiya/soa/blob/lab-2/SOA.postman_collection.json

Вывод

Мы сгенерировали каркас с помощью Swagger Codegen, познакомились на практике с Spring MVC и JAX-RS (Jersey провайдер). Также, была реализована секьюрность с помощью самоподписанных сертификатов для каждого сервиса.