Необходимо реализовать веб-приложение “Погодный брокер”.

Реализация приложения состоит из нескольких частей:

1. Необходимо реализовать JSP-страницу для ввода названия города. По сабмиту с помощью Spring RestTemplate отправляется запрос на weather.yahoo.com с введенным городом в качестве параметра. Форматы запросов детально описаны на сайте weather.yahoo.com. Полученный ответ посредством Jackson парсится в Java-класс, который далее отправляется в JMS-очередь, настроенную по архитектуре public/subscriber.
2. Необходимо реализовать JMS Listener, получающий сообщения из созданной очереди. После получения, содержимое сообщения конвертируется и сохраняется в базу данных посредством Hibernate. Данная операция должна быть обернута в Spring Transaction для распределенных источников данных.
3. Необходимо реализовать Spring Rest Controller, который по определенному запросу с указание в качестве параметра названия города, будет возвращать xml или json с соответствующей информацией о погоде. Информация берется из базы данных посредством Hibernate, парсится в json или xml с помощью Jackson.
4. Необходимо обеспечить покрытие исходного кода unit-тестами, с использованием JUnit и Mockito, или аналогов.

Проект должен собираться сборщиком maven или gradle. Исходники нужно загрузить на github или bitbucket.

Программный код должен удовлетворять [Code Conventions](http://www.oracle.com/technetwork/java/codeconventions-150003.pdf)

В качестве Application Server рекомендуется использовать WildFly. JMS-брокер в нем есть встроенный, но можно использовать любой сторонний. Все Connection, JMS-queue должны быть конфигурироваться на стороне Application Server и доступны в приложении через JNDI.

Если будет использован maven и WildFly, то рекомендуется использовать плагин maven-wildfly-plugin для деплоя приложения на сервер.

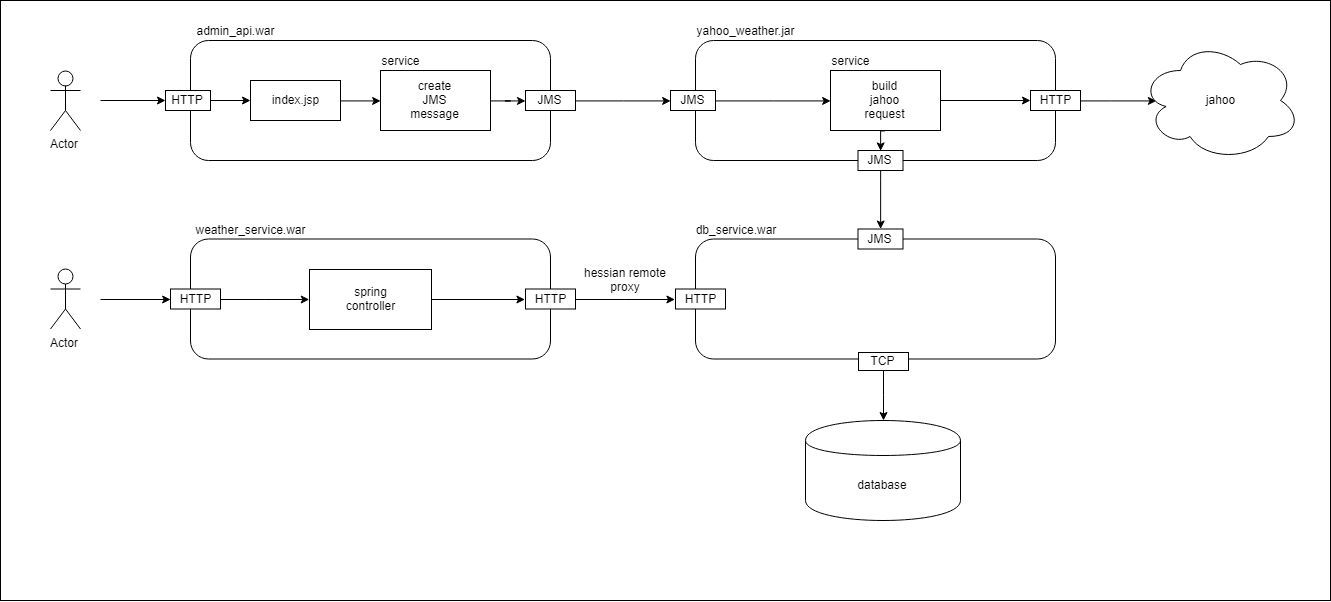
В качестве СУБД рекомендуется использовать PostgreSQL или MySQL.

Архитектура приложения должна быть реализована максимально-несвязанной, с использованием принципа Inversion-Of-Control.

Компоненты приложений должны быть максимально независимыми и взаимодействовать только через JMS. Если несколько модулей используют одни и те же модели, их (только модели) надо вынести в отдельный модуль и прописать в зависимости.

В модуле, где вводится название города, не использовать Spring MVC.

Сервис отправки запроса в yahoo должен быть реализован отдельным модулем (jar).



Взаимодействие между weather\_service.war и db\_service.jar желательно сделать через remote proxy.

Здесь можно почитать, как это сделать:

<https://docs.spring.io/spring/docs/current/spring-framework-reference/integration.html#remoting-caucho-protocols>