О РАЗРАБОТКЕ СИСТЕМЫ РЕКОМЕНДАЦИЙ И ПРОКЛАДКИ МАРШРУТОВ К ТУРИСТИЧЕСКИМ ОБЪЕКТАМ

Заблоцкий В. В., Рудикова Л. В.
Кафедра современных технологий программирования, Учреждение образования «Гродненский государственный университет имени Янки Купалы»
Гродно, Республика Беларусь
E-mail: {rudikowa, viktorzablotsky}@gmail.com

Излагаются общие подходы к реализации веб-приложения с микросервисной архитектурой на платформе JVM, которое предназначено для создания рекомендаций и поиска маршрутов к туристическим местам. Приводится основная функциональность программного обеспечения, связанного с разработкой указанного веб-приложения.

Введение

Многие люди часто совершают поездки и путешествия с различными целями. Любое путешествие невозможно без планирования, т.к. необходимо решить, в какое место ехать, где остановиться, найти жилье, а также решить, как проводить досуг. Относительно недавно люди, собираясь в путешествие, тратили много времени и усилий на поиск необходимой информации, нередко собирая туристическую литературу и обзванивая знакомых или туристические агенства. Однако с развитием Интернета стало возможно планировать путешествие не выходя из дома. В современном мире путешественник имеет всю необходимую под рукой благодаря Интернет-ресурсам, например, сервис по бронированию отелей, туристический гид или сервис для поиска попутчиков. Благодаря прорыву в разработке современных веб- и мобильных интерфейсов на выбор доступны десятки тематических приложений, в той или иной степени заслуживающих внимания пользователя. Туристическая сфера действительно огромна, в ней нашли свою нишу не только гиганты IT индустрии, такие как Google и Apple, но и множество небольших компаний и самостоятельных разработчиков.

Таким образом, в настоящее время возможно спланировать каждый аспект своего путешествия, используя различные веб-сайты и мобильные приложения. Для сомневающихся путешественников существуют не только справочники со статическим набором тех ли иных объектов, но и системы, специально предназначенные для подбора рекомендаций, исходя из предпочтений пользователя, а также системы, помогающие прокладывать эффективные маршруты между этими объектами. Такие системы на сегодняшний день являются крайне востребованными как среди обычных пользователей, так и разработчиков.

I. Этапы разработки приложения

Работу над реализацией приложения можно разбить на следующие этапы: проектирование инфраструктуры приложения, учитывая тот факт, что приложение состоит из нескольких микросервисов; проектирование каждого микросервиса; проектирование базы данных для каждого микросервиса; проектирование клиентского сервиса; реализация микросервисов и логики взаимодействия между ними; реализация баз данных и логики взаимодействия микросервисов с ними; реализация логики взаимодействия клиентской части приложения с серверной; реализация логики взаимодействия приложения со сторонними API; развертка приложения в виде кластера в облачных сервисах.

Некоторые аспекты разработки приложения включают в себя имплементацию авторизации через протокол OAuth 2, реализацию взаимодействия микросервисов через шину сообщений, работу с картой мира в интерфейсе приложения, а также интеграционное и юнит-тестирование приложения.

II. Общие подходы к реализации приложения

Рассмотрим основные компоненты архитектуры веб-приложения для создания рекомендаций и прокладки маршрутов к туристическим объектам.

Данное приложение разработано по принципам микросервисной архитектуры, что означает наличие нескольких слабо связанных между собой сервисов, выполняющих отдельные задачи и образующих единую систему, которая взаимодействует с интерфейсом приложения по принципам REST. Крайне важно правильно разграничить предметную область каждого сервиса для того, чтобы минимизировать их связанность. Это необходимо, чтобы уменьшить время отклика на запросы к серверу, ведь если данные сервисов будут тесно связанными, нужно будет реализовывать механизм транзакций с помощью протокола двухфазного коммита или отдельного

REST API. Если же данные слабо связаны и механизм транзакций не нужен, общей практикой для реализации взаимодействия сервисов является обмен сообщениями через Message Queue с достаточной гарантией целостности данных, например с помощью RabbitMQ.

Предлагаемое приложение состоит из трех (четырех, если считать клиентскую часть) основных сервисов: API-сервис, Geo-сервис и сервис аутентификации. Кроме основных, в виде отдельных модулей представлены API Gateway - единая точка доступа к микросервисам и балансировщик запросов, а также конфигурационный сервис, к которому подключаются все остальные сервисы при запуске.

Приложение может быть развернуто в изолированной среде как кластер и использовать собственную подсеть. Для этого используются такие средства, как Docker и Kubernetes. Сервисы, обернутые Docker-контейнерами могут быть размещены как локально в виртуальной машине с помощью Minikube, являющимся частью Kubernetes, так и на физических платформах и контролироваться средствами Kubernetes. При этом как Docker, так и Kubernetes предлагают средства для организации сети и мониторинга состояния развернутого кластера. Более того, существуют облачные сервисы, позволяющие размещать кластеры, такие как Amazon AWS.

Серверная часть приложения реализована на платформе JVM с использованием языка программирования Kotlin и фреймворков Spring Boot, Spring Cloud и отдельных модулей Spring Framework. Клиентская часть разработана с помощью фреймворка Vue.js на языке Typescript, использует карты OpenStreetMap и размещена в среде Node.js.

Рассмотрим кратко каждый из сервисов приложения и его клиентскую часть. Точкой входа, через которую проходят все запросы к приложению, является API Gateway. Этот модуль представляет собой прокси-сервер, перенаправляющий запросы к нужному сервису, исходя из контекста запроса. Кроме того, этот модуль имеет также механизм балансировки запросов, что крайне полезно при горизонтальном масштабировании приложения.

Конфигурационный сервис служит для задачи раздачи параметров сервисов в виде yaml-файлов. Все сервисы приложения должны подключиться и получить свои параметры у сервиса конфигурации, а значит он должен запускаться первым в кластере. Данный сервис позволяет хранить конфигурационные файлы в одном месте, при этом имеет механизм разделения файлов на профили.

Auth-сервис предназначен для аутентификации пользователей в приложении и реализует данный функционал с помощью средств Spring Security и протокола OAuth 2. В общем случае, после успешной аутентификации сервис возвращает клиенту JWT-токен, в котором содержится служебная инфрмация, дающая доступ к определенному ресурсу приложения. Все, что нужно для доступа к защищенным ресурсам - отправить в заголовке запроса этот токен. Следует отметить, что данный сервис выполняет лишь аутентификацию, т.е. проверяет истинность введенных пользователем имени и пароля. Авторизация, или проверка прав пользователя, выполняется каждым микросервисом независимо друг от друга.

API-сервис - главная часть приложения, где реализована основная бизнес-логика. Здесь происходит обработка запросов, не связанных с географическими вычислениями, такими как редактирование настроек пользователей, подбор рекомендаций и иные функции. Этот сервис реализован в классическом MVC-стиле и следует принципам REST архитектуры. В качестве базы данных была выбрана нереляционная СУБД MongoDB, в связи с ее простотой и одновременно мощностью, а также удобным способом хранения данных в формате JSON.

Микросервисом разрабатываемого приложения является также геосервис, в котором реализуется обработка запросов на вычисление локаций по заданным координатам, вычисление маршрутов и импорт данных из публичной базы OpenStreetMap. Также, как и API-сервис, данный сервис использует REST для коммуникаций с клиентом и следует паттерну MVC, уровень данных в котором представляет поисковая база Elasticsearch, выбранная благодаря встроенному механизму обработки геозапросов.

III. Выводы

Таким образом, разработанное приложение предназначено для широкого использования людьми, планирующими путешествие. Вебсистема предоставляет удобный интерфейс с картой, позволяющей наглядно получить информацию о туристических объектах в выбранном месте и составить удобный маршрут и график посещения этих мест.

В дальнейшем в приложение могут быть интегрированы новые сервисы, предназначенные для поиска отелей и каршеринга, а также разработана мобильная версия под ОС Android.

IV. Список литературы

1. Заблоцкий, В. В. О разработке системы рекомендаций и прокладки маршрутов к туристическим объектам / В. В. Заблоцкий //