утверждает, что периодические сигналы могут быть представлены в виде суммы синусов и косинусов при умножении на определенный аргумент, то преобразование Фурье утверждает то же самое, но только для непериодических сигналов. Это мы и видим на нашем анализаторе – частоту, присутствующую в сигнале, и ее амплитуду.

В программе была применена независимая библиотека *ScottPlot* для быстрого и удобного построения графиков сигналов. Для вычислений использовалась математическая библиотека *Accord.NET*.

WEB-АГРЕГАТОР ПО ПОДБОРУ КОМПЬЮТЕРНОЙ ТЕХНИКИ И КОМПЛЕКТУЮЩИХ С УЧЕТОМ ПРЕДПОЧТЕНИЙ ПОЛЬЗОВАТЕЛЯ

И. А. Бурин

Учреждение образования «Гомельский государственный технический университет имени П. О. Сухого», Республика Беларусь

Научный руководитель Н. В. Самовендюк

Описано web-приложение, предоставляющее информацию о комплектующих персонального компьютера и услугах ремонта и сборки компьютерной техники для региона, в котором проживает пользователь, а также помощь пользователю при самостоятельной сборке персонального компьютера с помощью встроенного конфигуратора.

Ключевые слова: web-агрегатор, компьютерная техника, компьютерные комплектующие, микросервисная архитектура.

В современном мире компьютерная техника и комплектующие являются неотъемлемой частью жизни многих людей. Компьютеры используются для работы, учебы, развлечения, коммуникации и многих других целей. Однако выбор подходящего оборудования может быть сложным и затратным процессом, особенно для неопытных пользователей.

Для решения этих проблем существуют специальные веб-сервисы, называемые агрегаторами. Агрегаторы по подбору компьютерной техники и комплектующих – это сайты, которые позволяют пользователю собрать свою конфигурацию персонального компьютера (ПК) онлайн, проверить совместимость компонентов, сравнить цены в разных магазинах и заказать готовый компьютер или отдельные детали. Такие сервисы облегчают процесс покупки и экономят время и деньги пользователя. Агрегаторы также помогают пользователю получить необходимую информацию и консультацию по выбору компьютерной техники и комплектующих, а также предоставляют возможность ознакомиться с отзывами и рейтингами других покупателей.

Однако, как и любая другая технология, агрегаторы сталкиваются с рядом проблем и вызовов, которые требуют постоянного совершенствования и инноваций. Одной из основных проблем агрегаторов является обеспечение качества и актуальности информации о комплектующих и ценах. Для этого необходимо постоянно обновлять базу данных, отслеживать изменения на рынке, проверять достоверность источников, учитывать специфику разных регионов и стран. Также необходимо учитывать индивидуальные предпочтения и потребности пользователей, предоставлять им гибкие настройки и возможности для сравнения и выбора оптимального варианта.

Основная задача разрабатываемого web-приложения – это помочь людям, которые столкнулись с проблемами сборки или ремонта компьютера, найти подходящие компоненты или соответствующие сервисы по ремонту, а также предоставить данные о цене в различных онлайн магазинах и сервисах, находящихся в регионе, где проживает пользователь.

Приложение предоставляет следующие функции:

– регистрация пользователей;

– регистрация участников сервиса (представители онлайн магазинов и сервисов);

– обработка, фильтрация, обновление информации о комплектующих и услугах с различных онлайн площадок;

– использование конфигуратора сборки ПК и хранение сборок;

– оповещение пользователей;

– ведение чата с контрибьюторами;

Функционал приложения разбит на роли:

– гость (незарегистрированный пользователь);

– зарегистрированный пользователь;

– контрибьютор;

– менеджер;

– администратор.

Структура разработанного приложения основана на микросервисной архитектуре, которая имеет ряд преимуществ перед другими типами архитектур построения программ.

Микросервисная архитектура — это подход к разработке программного обеспечения, при котором приложение состоит из набора небольших и независимых сервисов, каждый из которых выполняет определенную функцию и общается с другими по легковесным протоколам. Этот подход позволяет решать проблемы, связанные с монолитной архитектурой, такие как сложность сборки и развертывания, низкая гибкость и масштабируемость, высокая степень зацепления и низкая связность модулей.

Одним из ключевых принципов микросервисной архитектуры является разделение ответственности по бизнес-доменам. Каждый микросервис отвечает за один домен или поддомен и имеет свою базу данных, свой стек технологий и свой процесс разработки. Это позволяет ускорить время выхода на рынок, упростить внесение изменений и тестирование, повысить отказоустойчивость и производительность системы.

Для реализации микросервисной архитектуры использован язык программирования C# и фреймворк ASP.NET Core 7. C# – это современный, мощный и универсальный язык программирования, который поддерживает объектно-ориентированное, функциональное и обобщенное программирование. ASP.NET Core 7 – это кроссплатформенный фреймворк для создания веб-приложений и веб-сервисов с высокой производительностью и безопасностью.

Для разработки пользовательского интерфейса были использованы библиотека React и языки разметки и стилей HTML и CSS.

Для запуска и развертывания сервисов будет использована технология контейнеризации Docker. Docker – это программное обеспечение для автоматизации развертывания и управления приложениями в изолированных контейнерах. Контейнеры позволяют запускать приложения в любой среде с минимальными зависимостями и настройками. Docker также облегчает сборку, распределение и масштабирование приложений.

Для обеспечения аутентификации и авторизации пользователей и сервисов использовано решение Duende.IdentityServer – фреймворк для реализации протоколов OpenID Connect и OAuth 2.0 на базе ASP.NET Core. Duende.IdentityServer позволяет создавать централизованный сервер удостоверений, который выдает токены для доступа к защищенным ресурсам.

Для обмена сообщениями между сервисами использован брокер сообщений RabbitMQ. RabbitMQ – это надежный, высокопроизводительный и распределенный брокер сообщений, который поддерживает различные шаблоны обмена сообщениями, такие как очереди, топики, обмены и маршрутизация. RabbitMQ позволяет асинхронно и надежно передавать сообщения между сервисами, а также обрабатывать события и команды.

Для управления трафиком и маршрутизации запросов к сервисам использован шлюз API Ocelot. Ocelot – это библиотека для ASP.NET Core, которая позволяет создавать простые и быстрые шлюзы API для микросервисных архитектур. Ocelot поддерживает различные функции, такие как балансировка нагрузки, агрегация ответов, кэширование, аутентификация, авторизация и т. д.

Кроме того, для работы с данными используются базы данных – MSSQL и MongoDB. MSSQL – это реляционная база данных, которая хранит данные в виде таблиц и поддерживает язык SQL для запросов. MongoDB – это нереляционная база данных, которая хранит данные в виде документов и поддерживает язык JSON для запросов. MSSQL будет использован для хранения структурированных и связанных данных, таких как информация о пользователях, каталоге и участниках сервиса, а MongoDB – для хранения неструктурированных и гибких данных, таких как отзывы, цены и сборки пользователей.

В данном приложении реализовано шесть микросервисов:

– **Identitym-service** – сервис для авторизации и аутентификации пользователей, генерация и верификация jwt-токенов;

– **Aggregatorm-service** – сервис для просмотра и управления каталогом комплектующих и услуг сервисных центров;

– **Contributorm-service** – сервис, который предлагает дополнительные функции для участников-партнеров, которые предоставляют данные цен и отзывов в своих магазинах;

– **Configuratorm-service** – сервис, предоставляющий функции онлайн конфигуратора ПК;

– **Pricem-service** – сервис для управления ценами, которые поступают с магазинов и сервисов партнеров;

– **Mailm-service** – сервис для оповещения пользователей.

Для каждого микросервиса разработана своя база данных:

Identity – SQL (MSSQL);

Contributor – SQL (MSSQL);

Aggregator – SQL (MSSQL);

Configurator – NoSQL (MongoDb);

Price – NoSQL (MongoDb);

Mail – NoSQL (MongoDb).

На рис. 1 можно увидеть схему микросервисной архитектуры данного приложения.

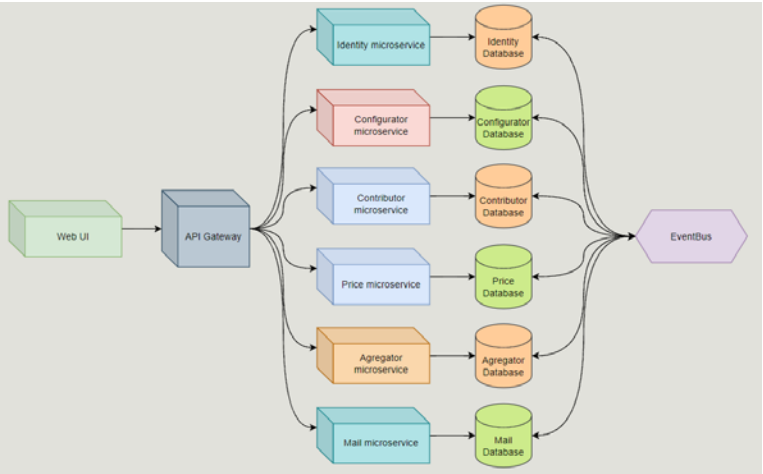

Рис. 1. Микросервисная архитектура разработанного приложения

Разработанное web-приложение позволит оперативно получать информацию о комплектующих персонального компьютера и услугах ремонта и сборки в регионе проживания, а также самостоятельно собирать конфигурацию персонального компьютера.

К ЗАДАЧЕ О РАЗРАБОТКЕ НА C# ПОЛЬЗОВАТЕЛЬСКИХ БИБЛИОТЕК СТАНДАРТНЫХ ДЕТАЛЕЙ ДЛЯ КОМПАС-3D

А. О. Гуща, Т. Д. Стасенко

Учреждение образования «Гомельский государственный технический университет имени П. О. Сухого», Республика Беларусь

Научный руководитель В. С. Мурашко

Объектом исследования является пользовательская библиотека крепежных изделий. Предложена методика компиляции и подключения пользовательских библиотек к КОМПАС-3D из приложения Windows Forms на C#, которая дает возможность создавать не только простые библиотеки, но и довольно сложные пользовательские библиотеки.

Ключевые слова: КОМПАС-3D, язык C#, пользовательская библиотека, динамическая библиотека, технология COM, приложение Windows Forms.

С использованием специальных прикладных библиотек множество рутинных операций, выполняемых конструкторами в КОМПАС-3D можно автоматизировать. Среди наиболее популярных библиотек можно выделить следующие:

– «Конструкторская» для вставки в чертежи изображений болтов, винтов, гаек, пружин, подшипников и т. д.;

– «Стандартные изделия» для вставки 3D-моделей стандартных изделий в сборку;

– «АРМ FEM» – система прочностного анализа.

Основные расширения файлов библиотек – *.rtw и *.dll (dynamic link library – динамически подключаемая библиотека Windows).

В данной работе объектом исследования является пользовательская библиотека крепежных изделий.

Каждое производство имеет собственную специфику и масштабы, поэтому существуют различия и в номенклатуре стандартизованных изделий, которые применяются в изготавливаемой продукции. В силу этого может быть достаточно много