ДелиЧек

ДелиЧек - сервис для деления счета в ресторане или кафе.

https://deli-check.ru/

Инструкция по запуску в конце.

Общая информация:

- Состоит из API и веб-приложения
- Написан на языке C# с использование ASP.NET Web Api и Blazor Web Assembly
- Распознавание чека с помощью Tesseract и базы данных ФНС
- Интеграция с ВКонтакте (синхронизация списка друзей и данных профиля)

Перспективы:

- Разработка нативных приложений для Android и IOS
- Разработка чат-бота в Telegram
- Улучшение алгоритмов OCR для распознавания чека (например, реализация алгоритмов препроцессинга)
- Улучшение дизайна веб-приложения
- Интеграция с различными сервисами, например, экосистема СБЕР

Обоснование выбора технологий

- **.NET, Blazor Web Assembly и язык C#.** Язык C# и платформа .NET позволяет создавать приложения почти для любой платформы и любого назначения. Очень гибкий язык и платформа. ASP.NET Web Api предоставляет очень широкие возможности для разработки, к тому же, это кроссплатформенная технология, многие крупные сайты и приложения используют именно ASP.NET. Blazor Web Assembly был выбран из-за того, что он позволяет быстро и качественно реализовать одностраничное приложение. Использование уже готовой кодовой базы на C# с моделями объектов, использующихся в API, ускоряет разработку. Библиотека Radzen для Blazor предоставляет возможность быстро разработать довольно сложный UI. Web-приложение на WASM максимально похоже на нативное приложение при использовании.
- **Tesseract** - самая популярная open-source библиотека для распознавания текста. Есть поддержка русского языка, довольно точные результаты. Из всех полностью бесплатных инструментов, Tesseract показал наиболее хорошие результаты.
- **Получение данных из ФНС** - крайне удобная технология, так как результаты почти со стопроцентной вероятностью точны. Однако, для использования API ФНС требуется юр. лицо либо ИП, которым участники нашей команды не располагают. На просторах интернета был найден сайт https://proverkacheka.com/ Он предоставляет бесплатную возможность проверки чека с помощью данных с налоговой службы. Мы автоматизировали процесс получения данных, используя реверс-инжениринг, так как данные там были зашифрованы. Нашей главной задачей было показать возможность технологии, в дальнейшем переход на официальное API обязателен для обеспечения стабильной работы.
- **Интеграция с ВКонтакте** - Вконтакте предоставляет открытый API для разработчиков, его можно использовать для авторизации, и главный критерий - можно получить список друзей. Это упрощает взаимодействие пользователя с приложением.
- **Использование API** предоставляет гибкость в выборе реализации клиентского приложения. С REST API можно взаимодействовать с любого устройства. Даже Telegram-бот, который будет исполняться на сервере, может взаимодействовать с API.
- **Веб-приложение показалось нам наиболее правильным вариантом из-за сжатых сроков**. Разработать приложения и для Android, и для IOS мы бы не успели, а ограничивать пользователей одной из платформ не хотелось.
- **SQLite** - в нашем проекте используется база данных SQLite. Это хороший вариант для небольших проектов с небольшой нагрузкой. Благодаря фреймворку EntityFrameworkCore, с помощью которого программа взаимодействует с базой данных, можно заменить базу данных на любую другую без изменения кодовой базы.

Наша архитектура позволяет быстро заменить технологии распознавания чеков и интеграции с другими сервисами без особых сложностей благодаря DI

Возможности приложения:

Авторизация

Приложение нацелено на удобство пользователя, поэтому было разработано несколько способов входа:

- По имени пользователя и паролю

- Через ВКонтакте
- Регистрация без ввода имени пользователя и пароля (подойдет например тем, кто всего пару раз воспользуется приложением; пользователь останется авторизированным при следующем входе; для сохранения учетной записи можно привязать профиль Вконтакте)

Список друзей и профиля

Приложение предоставляет возможность организовать список друзей. Это удобно, если пользователь ходит куда-нибудь поесть зачастую с одними и теми же людьми.

- Можно найти аккаунт друга среди всех пользователей по имени и фамилии и добавить его в друзья
- Если друг не пользуется ДелиЧек, его можно добавить в список, введя его имя и фамилию

Таким образом, приложение не будет ограничено в том случае, если не все участники трапезы пользуются приложением.

При авторизации через ВК импортируется список друзей. В этом случае пользователю не придется никого добавлять, а просто выбрать нужных людей при разделении чека.

У каждого зарегистрированного пользователя есть свой профиль. Он может изменить отображаемое имя и фамилию, поставить аватарку. Данные профиля тоже синхронизируются с ВКонтакте.

В дальнейшем, с помощью приложения на Android *или* IOS *можно импортировать друзей из телефонной книжки*

Чеки

В меню пользователь может найти пункт **"Мои чеки"**:
На ней пользователь видит все чеки, которые с ним разделили либо он загрузил. Можно посмотреть подробную информацию о каждом чеке, кликнув на него

Страница чека

1. **На странице чека указаны все позиции и итоговая сумма**, список людей, кто разделял в компании друзей вкуснейший ужин.
2. **Рядом с каждым человеком указана сумма, сколько он должен, и оплатил ли он счет** (например, перевел или отдал наличкой). *Тот, кто загрузил чек, может проставлять "Оплачено" за любого человека*
3. Таким образом, для человека, оплатившего ужин в ресторане, **под рукой удобный список со всеми людьми**, и сразу можно увидеть, с кого надо требовать деньги.
4. Доступно **копирование списка в текстовом формате** (можно выслать в чат с друзьями, чтобы все были в курсе)

Деление чека

1. **Пользователь загружает чек в приложение и обрезает его**, чтобы на изображении остался только чек без заднего фона. Алгоритм ищет QR-код фискального чека, и если находит его, получает информацию о позициях в чеке и итоговой сумме через базу данных ФНС. Это идеальный вариант развития событий, потому что, получая данные из ФНС, вероятность ошибки в стоимосте позиций и общей сумме приближается к нулю. Однако, в настоящий момент рестораны и другие заведения общепита могут выдавать пречеки - это не фискальный документ, поэтому QR-кода там не будет. В таком случае алгоритм распознает текст на чеке и формирует список позиций и итоговую стоимость.
2. **На экран выводится список позиций и итоговая сумма.** Если программа некорректно распознала какую-либо часть чека, пользователь может это поправить.
3. **Пользователь выбирает людей, которые ели с ним**. Выбрать можно из списка друзей или найти человека среди зарегистрированных пользователей.
4. **Пользователь указывает список позиций** (включая количество порций) для каждого человека.
5. **Пользователь нажимает "Разделить чек"** и всем зарегистированным пользователям добавляется счет в список "Мои счета". Пользователь может отправить ссылку на страницу чека, где каждый участник застолья может посмотреть, сколько он должен.

Интегрировав этот функционал, например, с банковкским приложением, пользователь сможет одним кликом перевести нужную сумму тому, кто оплатил весь ужин. Приложение на Android *или* IOS *может отправить уведомление*

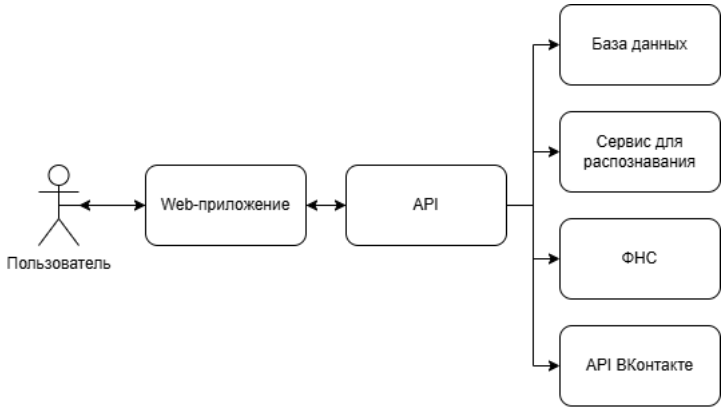
Процесс деления между людьми

1. Позицию с одиночным или дробным количеством можно разделить поровну между несколькими людьми (выбрав необходимых людей).
2. Позиции с целым количеством делятся в соответствии с выбранным количестом порций для каждого человека (кликнув на пользователя, его количество порций увеличивается).
3. Если необходимо разделить одну позицию в разных пропорциях между людьми (например, целую пиццу, каждый съел разное количество кусков) - увеличиваем количество в позиции до 8 (столько кусков в пицце) и делим как в **пункте 2**
4. Можно добавить чаевые или скидку. Их тоже можно поделить между пользователями, но по умолчанию они делятся между всеми участниками поровну

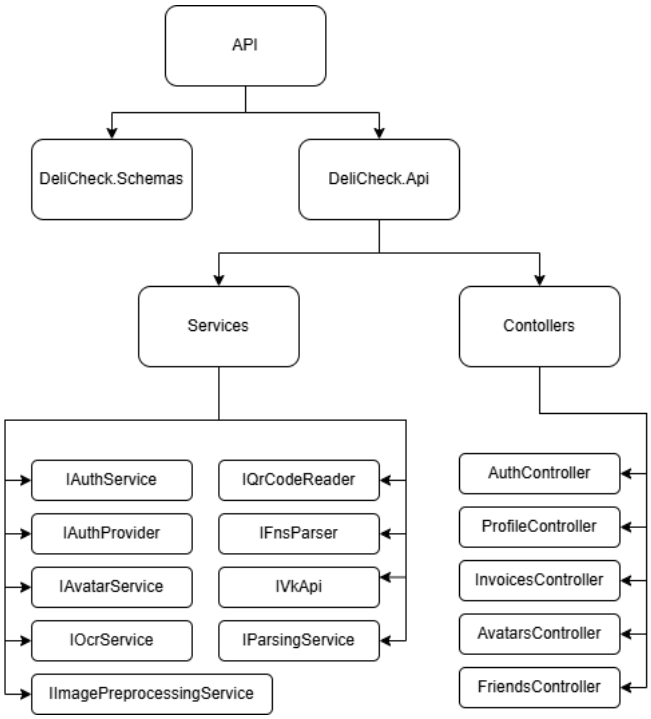
Документация

Документация к API: [тут]

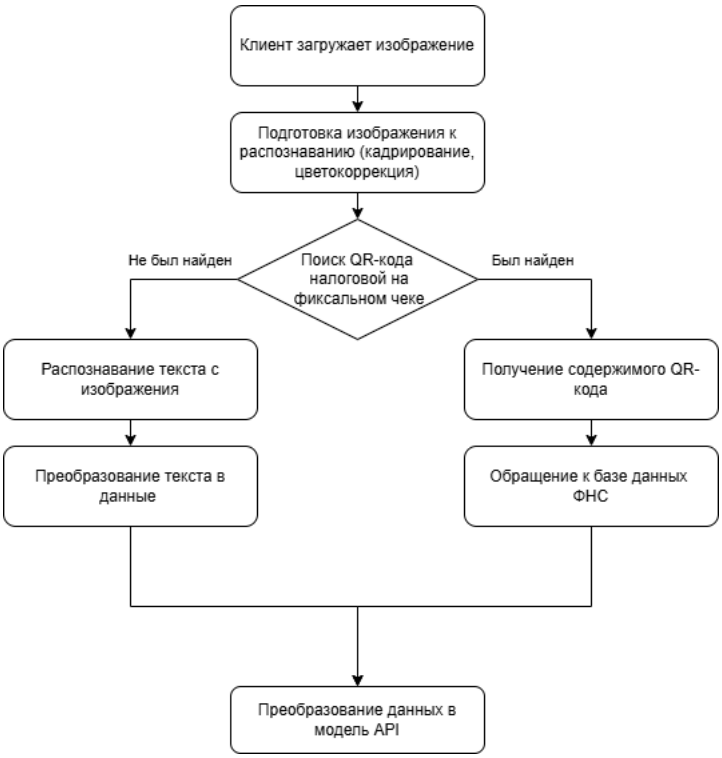
Схемы:



Архитектура приложения



Архитектура API



Алгоритм распознавания чека

Инструкция по запуску

Готовый к использованию проект скомпилирован и развернут на сайте https://deli-check.ru
Для развертывания приложения необходимы два сервера: сервер API и сервер с веб-приложением.

Сервер API

Рабочая платформа - Windows Server 2019

Необходимые зависимости при работе: Tesseract (Версия для Windows, выбраны дополнительные языковые модели: Русский, Scripts: Cyrrilic), .NET 8 Hosting Bundle

Для компиляции необходима MS Visual Studio Community 2022 с компонентами .NET 8 и ASP.NET Core Web API. Проект DeliCheck.Api.csproj. Если необходимо восстановить пакеты Nuget, которые неоходимы для компиляции, надо кликнуть правой кнопкой мыши кнопкой мыши на решение и выбрать "Восстановить пакеты Nuget". Используемые пакеты:

```
Microsoft.EntityFrameworkCore.Sqlite 9.0.3
Microsoft.VisualStudio.Azure.Containers.Tools.Targets 1.19.6
SixLabors.ImageSharp 3.1.7
Swashbuckle.AspNetCore 6.4.0
ZXing.Net 0.16.10
Xing.Net.Bindings.ImageSharp.V2 0.16.17
```

Также необходимо внести изменения в settings.json

```
{
    "Logging": {
        "LogLevel": {
        "Default": "Information",
        "Microsoft.AspNetCore": "Warning"
        }
    },
    "BindIP": "{IP сервера, на который биндить сервер}",
    "VkAppId": 53377795, // идентификатор приложения ВК
    "VkRedirectUri": "https://api.deli-check.ru/auth/vk-callback", // Редирект с VK. Не будет работать без изменения
```

```
    "VkRedirectConnectUri": "https://api.deli-check.ru/auth/vk-connect-callback", // Редирект с VK. Не будет работат
    "VkScope": "vkid.personal_info friends",
    "AllowedHosts": "*",
    "Domain": "api.deli-check.ru", // доменное имя, которое ассоциировано с IP адресом сервера
    "TesseractPath": "C:\\Program Files\\Tesseract-OCR\\tesseract" // Путь к рабочей папке Tesseract
}
```

Опционально: заменить certificate.pfx на нужный, если схема https необходима
Публикация приложения: необходимо создать профиль публикации "В папке" и выбрать целевую среду выполнения. После этого можно запускать скомпилированное приложение, работает на веб-сервере Kestrel. Работу сервера можно проверить, зайдя на https://IPадрес/swagger/index.html

Сервер веб-приложения

Рабочая платформа - Windows Server 2019

Необходимые зависимости при работе: IIS, Blazor Web Assembly, .NET 9

Для компиляции необходима MS Visual Studio Community 2022 с компонентами .NET 9, Blazor Web Assembly, .NET 9.0 WebAssembly BuildTools. Проект DeliCheck.Web.csproj. Если необходимо восстановить пакеты Nuget, которые неоходимы для компиляции, надо кликнуть правой кнопкой мыши кнопкой мыши на решение и выбрать "Восстановить пакеты Nuget". Используемые пакеты:

```
BlazorCurrentDevice 1.0.7
Blazored.LocalStorage 4.5.0
Blazorise 1.7.5
Blazorise.Bootstrap5 1.7.5
Blazorise.LottieAnimation 1.7.5
Cropper.Blazor 1.4.0
Microsoft.AspNetCore.Components.WebAssembly 9.0.4
Microsoft.AspNetCore.Components.WebAssembly.DevServer 9.0.4
Radzen.Blazor 6.5.3
```

Необходимо настроить URL для конечных точек API. В файле Program.cs изменить BaseAddress для HttpClient на URL-адрес API сервера.

Публикация приложения: необходимо создать профиль публикации "В папке", включить предварительную комплиляцию AOT. После публикации необходимо настроить сервер IIS с помощью IIS Manager:

1. Скопировать все содержимое папки публикации в отдельную папку
2. Создать новый сайт, указать физический адрес папки тот, куда перенесли опубликованное приложение.
3. Настроить привязки для сайта, указав IP адрес и порт. Желательно настроить схему HTTPS, без неё serviceWorker.js на клиенте не будет работать (может вызвать проблемы), либо отключить в браузере secureContext.
4. Настроить разрешения системных пользователей IIS для рабочей папки.
5. Запустить сервер IIS

Авторы:

- Толмачев Денис Михайлович
- Толмачев Андрей Михайлович
- Аксенов Владислав Владимирович
- Михайлов Александр Андреевич

Команда ЮЗГУ https://swsu.ru