# Тестовое задание FE-1

## Описание задания:

У нас есть небольшой продуктовый магазин, в нем продаются бананы, яблоки, груши и помидоры. Больше ничего не продается. Нам нужно написать небольшое приложение, чтобы пользователи могли ознакомиться с нашим богатым ассортиментом и собрать себе в корзину офощи и фрукты.

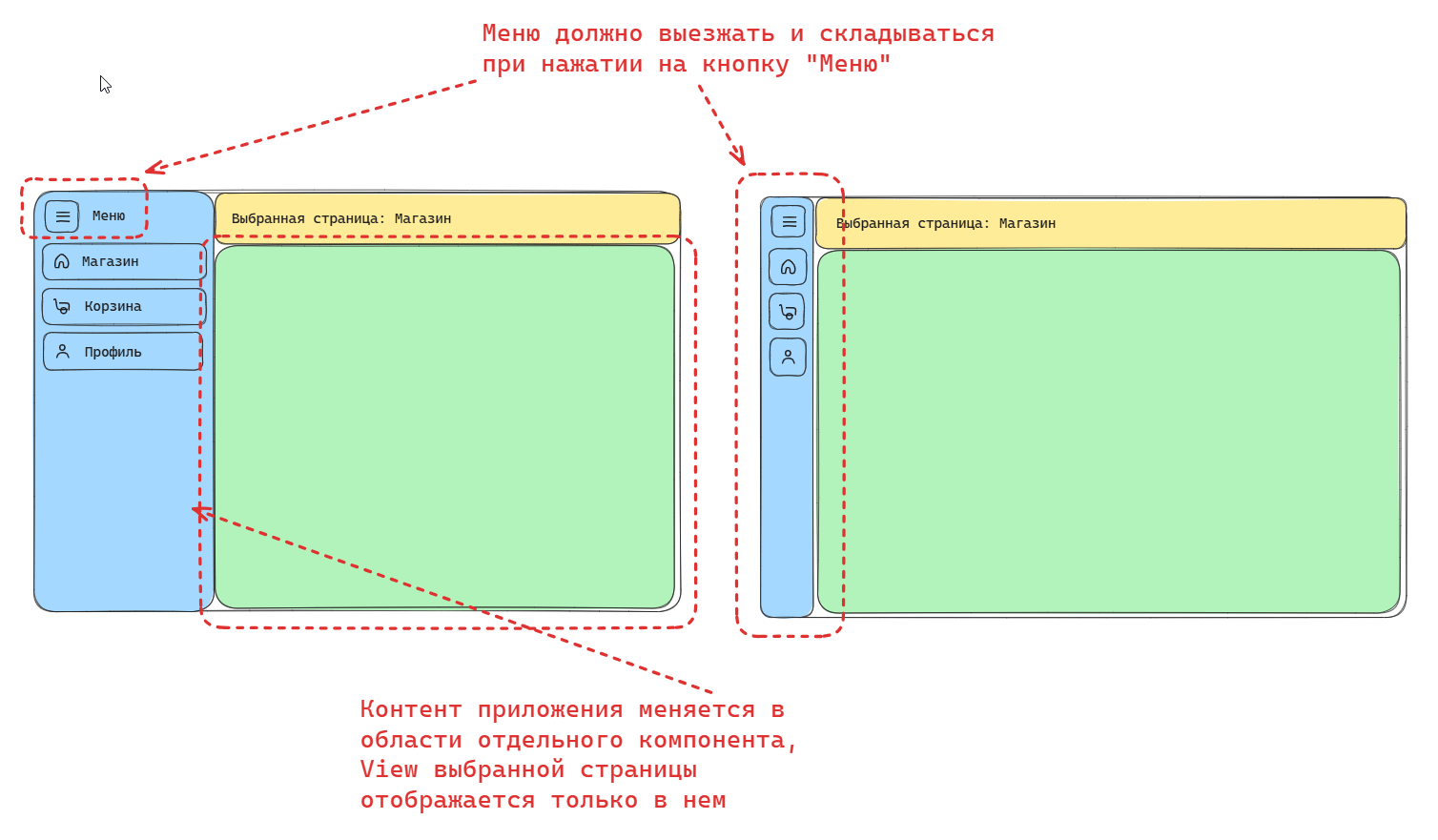
В приложении будет:

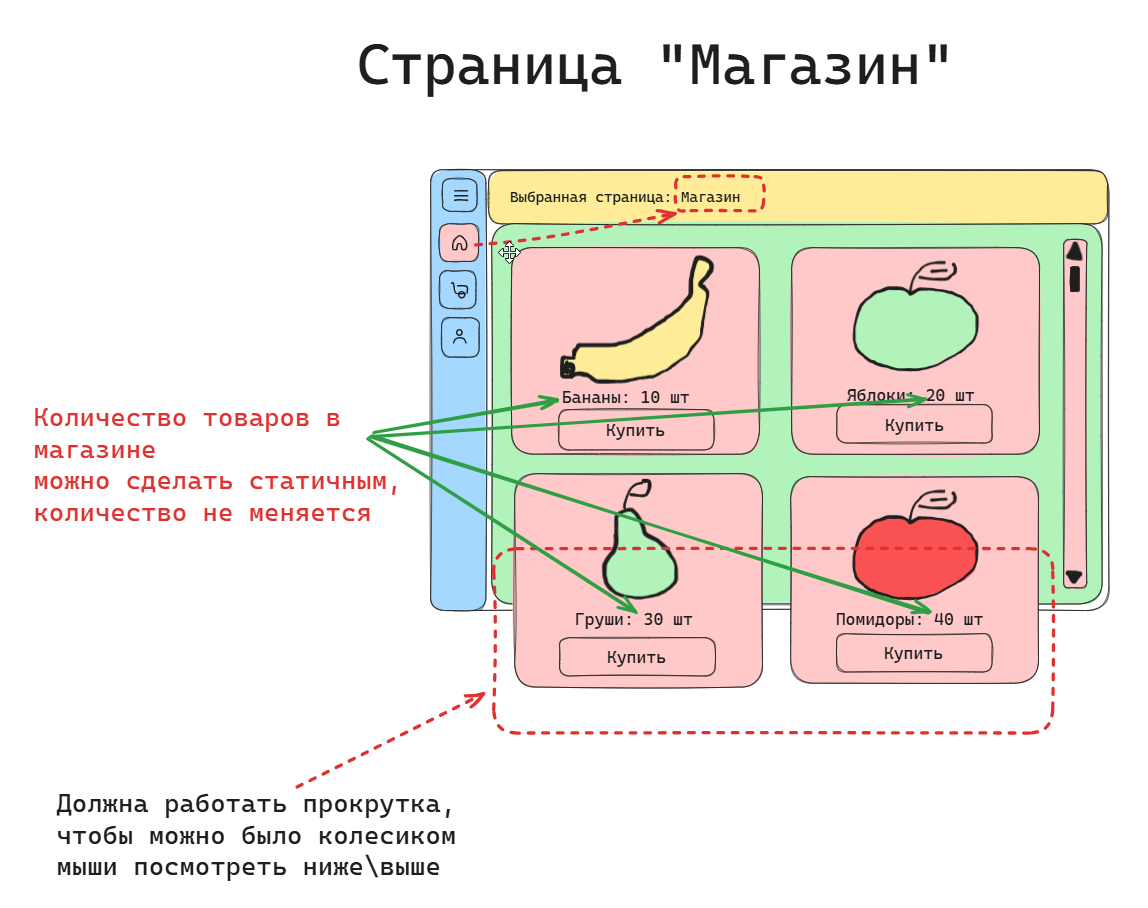
- меню навигации с иконками

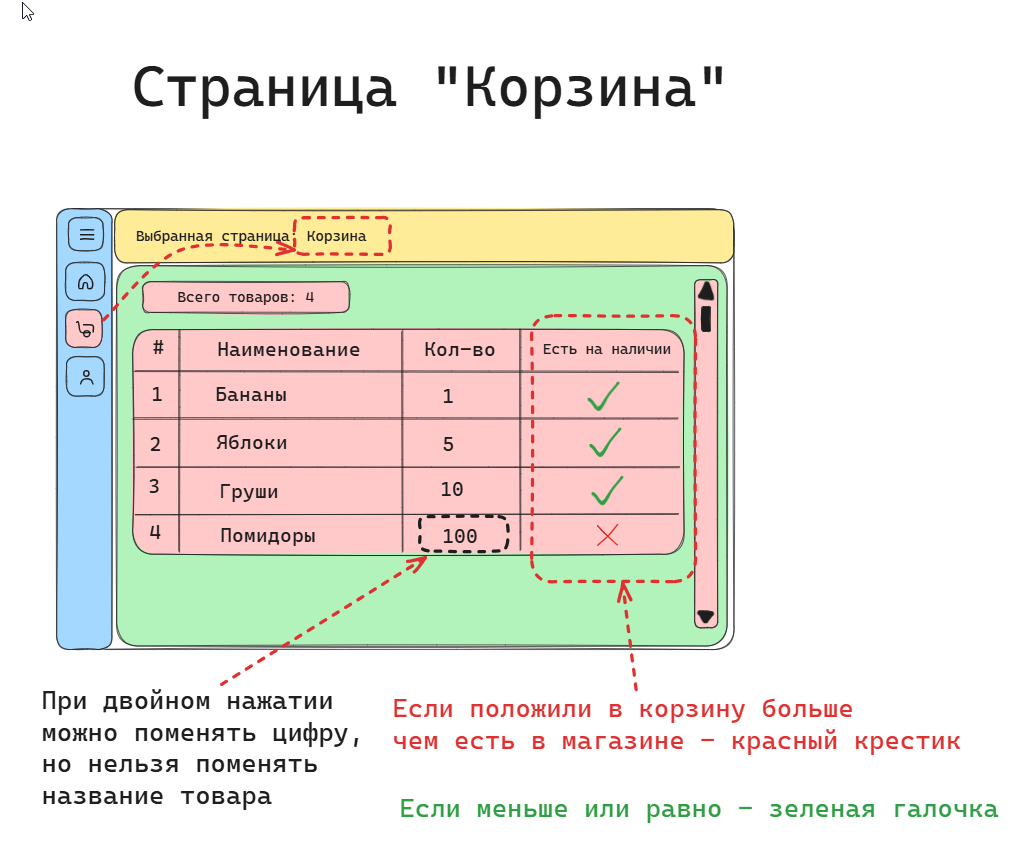
- заголовок приложения

- и три страницы: магазин, корзина и профиль пользователя

Мы думаем, что наши пользователи чаще пользуются ноутбуком и браузером на нем, поэтому приложение должно работать на персональном компьютере (Desktop приложение) и в браузере (Web). Мы хотим в будущем сделать так, чтобы пользователям не приходилось устанавливать приложение, а достаточно было бы сосканировать QR со ссылкой и открыть сайт магазина.

Дизайнер приготовил следующиие макеты:  






## 

## Детали реализации:

1. Авторизация и аутентификация пользователя не требуется.
2. Инструменты и стек: dotnet 8, AvaloniaUI.

Обязательно необходимо использовать паттерн MVVM и один из фрейморков, на выбор: MVVM Community Toolkit или ReactiveUI.

Какой DI контейнер использовать: на ваш выбор. Можно использовать Microsoft.Extensions.DependencyInjection, Splat, MVVM.Light, Unity или любой другой. У нас используется Microsoft.Extensions.DependencyInjection и Splat.

1. В критериях оценки и в задании говорится о том, что приложение должно использовать HttpClient, что предполагает наличие веб сервера и API.

Мы не просим вас написать API, в качестве веб сервера используйте пожалуйста Mockoon https://mockoon.com/

Этой утилитой можно запустить небольшой REST API, который будет возвращать заранее известные данные.

Будет достаточно cделать в нем один endpoint который будет возвращать список товаров в магазине:

* URL картинки,
* Название: помидоры, бананы, и т.д.
* Количество

1. Количество товаров в магазине можно сделать статичным, количество не меняется от того положили мы себе что-то в корзину или нет.
2. Адаптивная верстка под разные размеры устройств не требуется.

Можно использовать самые простые стили и дизайн компонентов, достаточно темы Fluent или Simple.

1. Обязательно выполнить каждый критерий и каждое требование, описанные ниже в списке.

Это значит, что для самопроверки вы можете пройтись по каждому пункту и убедиться, что точно его выполнили.

Если вы сделали больше чем требуется в списке, то это привествуется, но не обязательно.

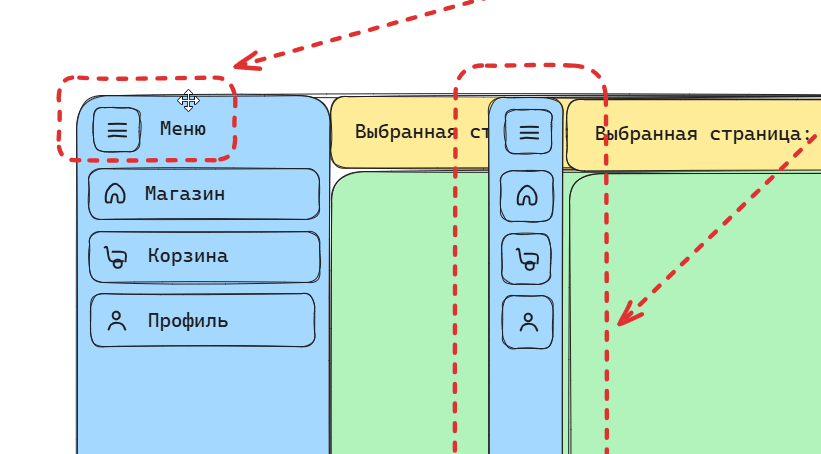
Если меньше или не в полной мере: не привествуется.

Если считаете что по какой-то причине критерий невозможно выполнить (технически например) или если считаете его излишним, то напишите это комментарием **TODO:** в коде.

Все то, что описано в задании и в критериях, уже реализовано в том приложении, с которым вам предстоит работать.

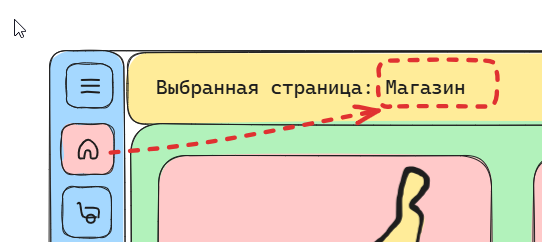
## Критерии оценки и требования:

1. Меню должно выезжать и складываться при нажатии на кнопку "Меню"



2) При нажатии на панели навигации должен происходить переход на страницу

3) При переходе между страницами в заголовке приложения отображается название выбранной страницы



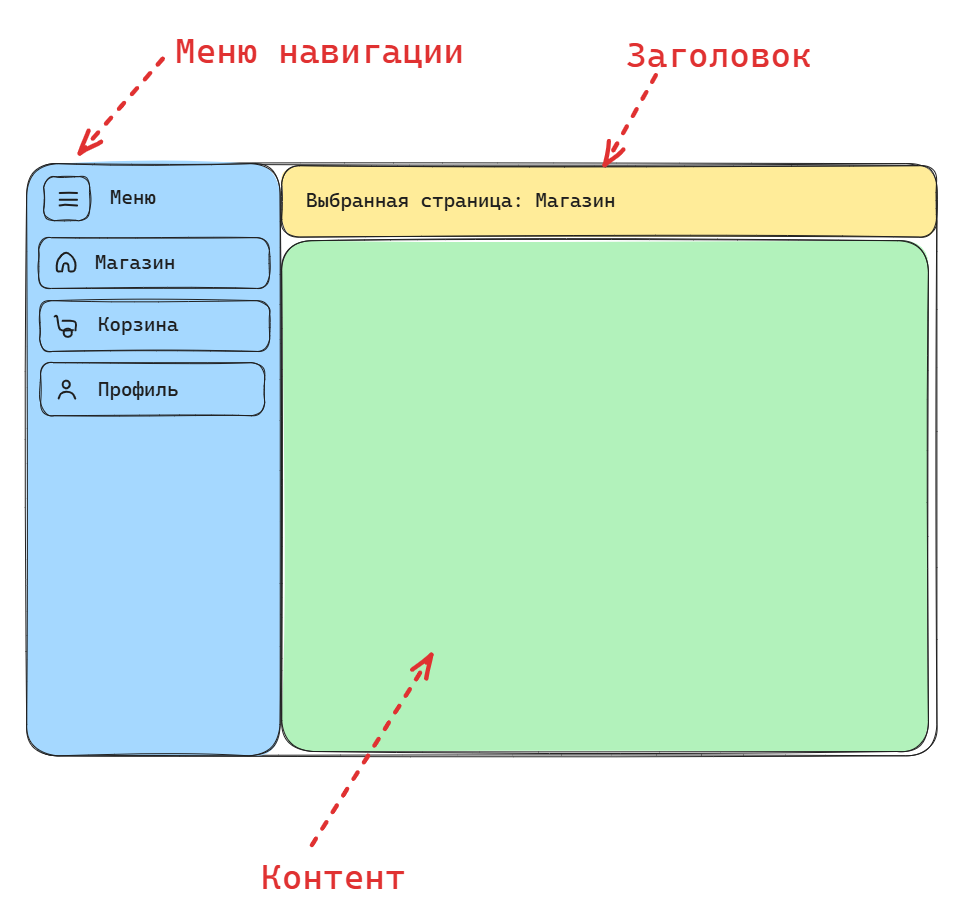
4) Главная страница должна быть разделена на три отдельных View, три отдельные части.

Верстка этих частей должна находиться в разных файлах, у каждого должен быть отдельный <View>.axaml и <ViewModel>.cs:

- панель навигации

- заголовок

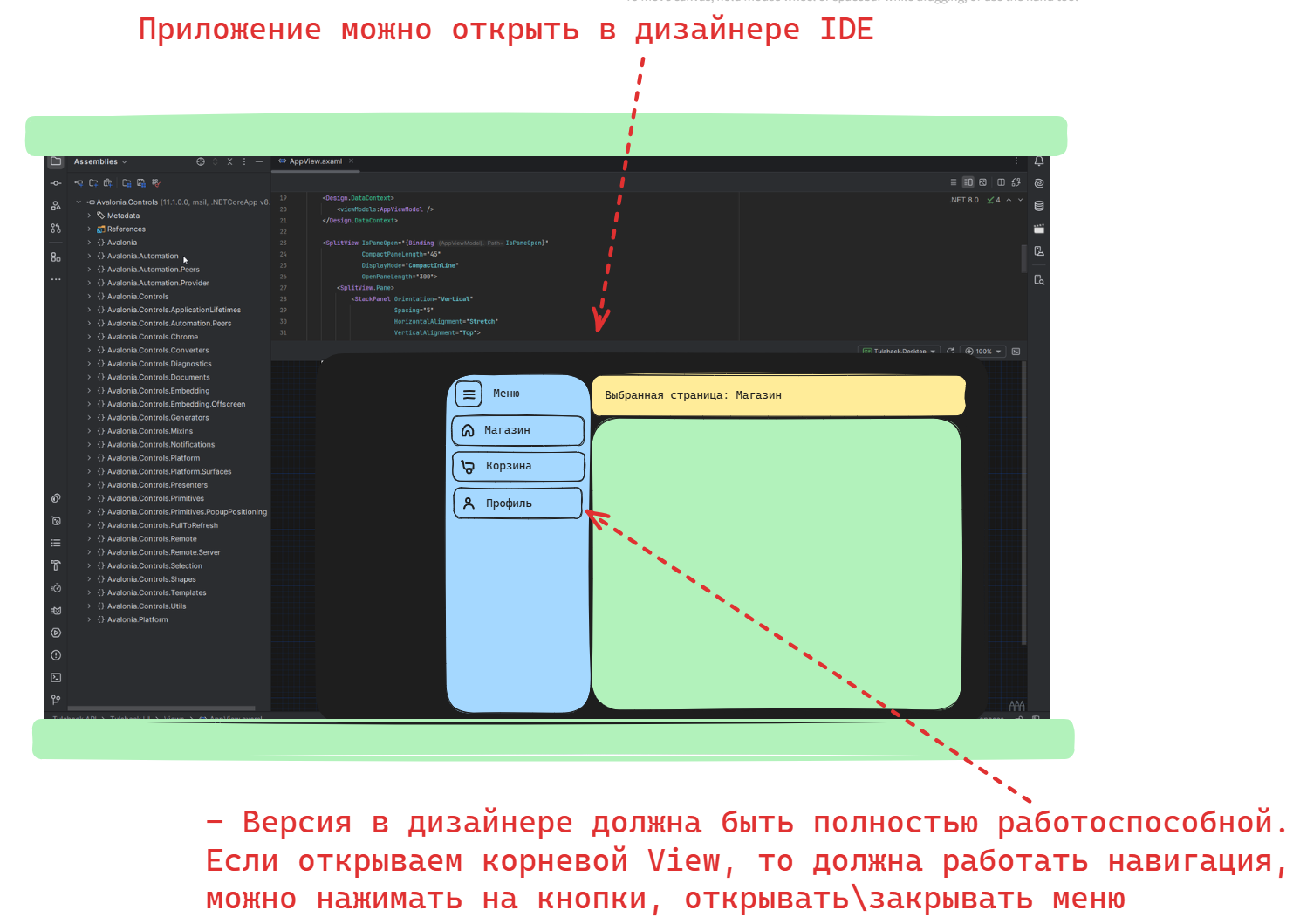
- контент (выбранная страница)



5) В приложение есть меню навигации, заголовок, контейнер для отображения контента и три страницы: магазин, корзина, профиль.

6) В дизайнере можно открыть как главную страницу, так и каждый компонент по отдельности.

Версия в дизайнере должна быть полностью работоспособной. Если открываем корневой View, то должна работать навигация, можно нажимать на кнопки, открывать\закрывать меню.



7) Приложение должно работать на Desktop и в браузере

8) Обязательно должен быть использован DI и InversionOfControl. ViewModel, вспомогательные сервисы, HttpClient не должны созваваться вручную, а должны регистрироваться в Dependency Injection контейнере и внедряться как зависимость в конструкторе.

То есть: ViewModel-ы и прочее **НЕ ДОЛЖНЫ** создаваться как:

**var viewModel = new MyViewModel(param1, param2, param3);**

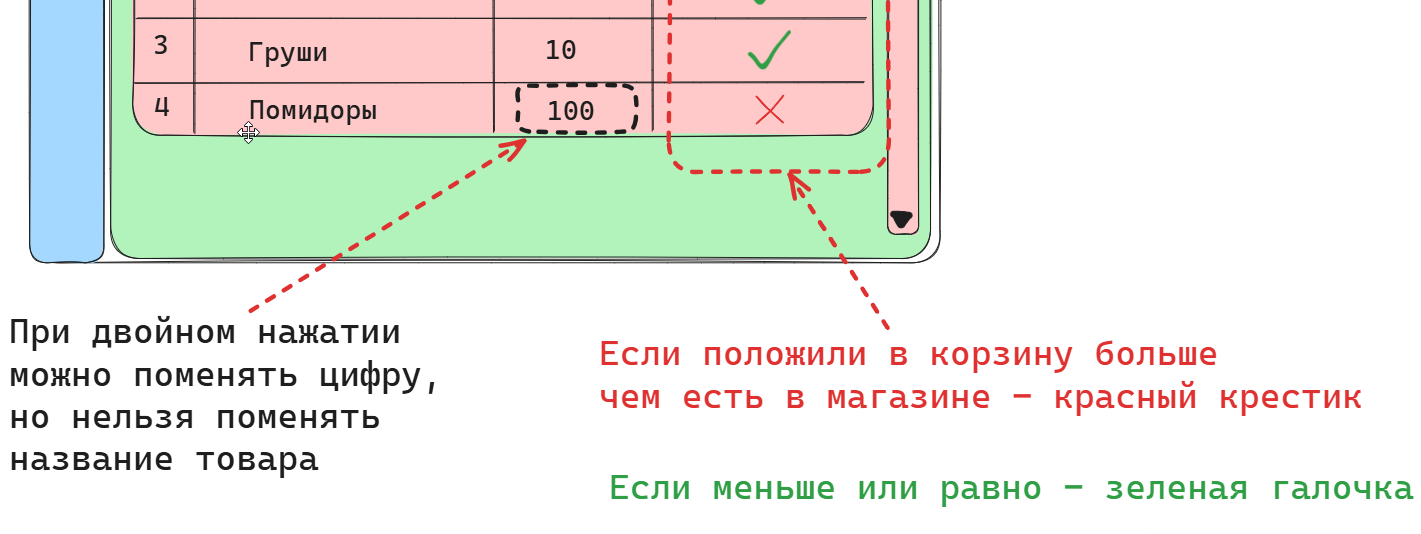
Исключение - если необходимо инициализировать первый корневой ViewModel в App.axaml.cs

9) В карточках товаров должна быть картинка, можно выбрать в интернете.

10) Приложение отправляет GET запрос в API и показывает полученные данные в виде карточек, приложение должно использовать HttpClient

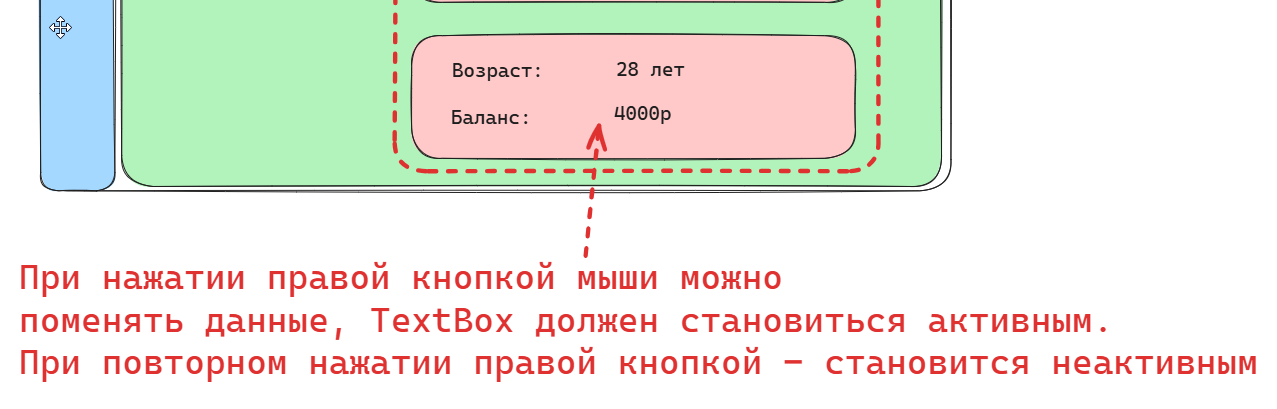
11) Должна работать прокрутка, чтобы можно было колесиком мыши посмотреть ниже\выше если контент не помещается

12) В корзине, при двойном нажатии можно поменять количество товаров (цифру), но нельзя поменять название товара (нельзя переименовать бананы в кукурузу)

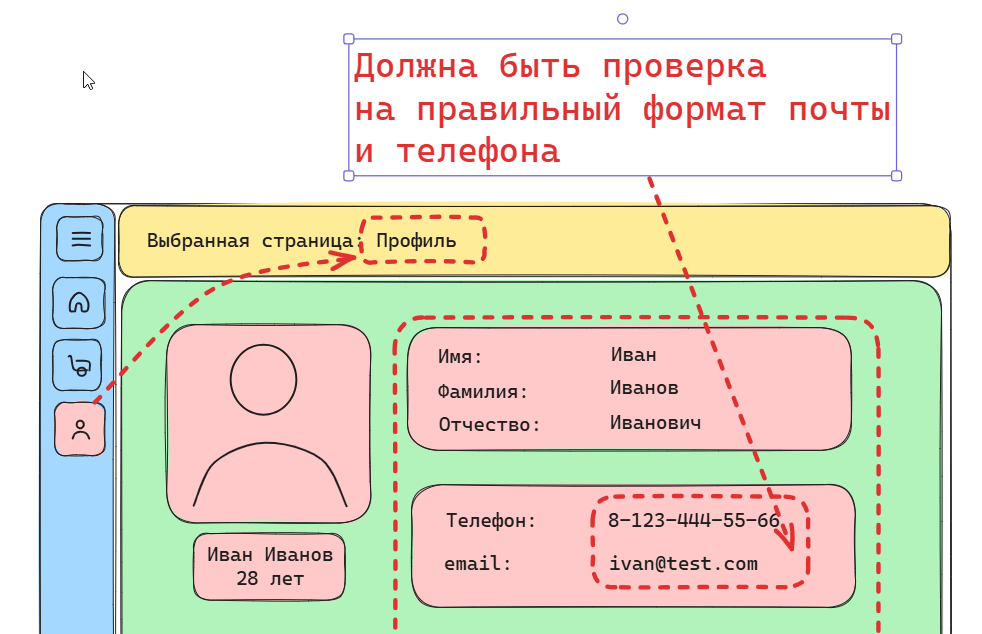


13) Если положили в корзину больше чем есть в магазине - в последней колонке появляется красный крестик. Если меньше или равно - зеленая галочка

14) В профиле пользователя: при нажатии правой кнопкой мыши можно поменять данные, TextBox должен становиться активным. При повторном нажатии правой кнопкой - становится неактивным



15) В профиле пользователя: должна быть проверка на правильный формат почты и телефона



16) Обязательно необходимо использовать паттерн MVVM и один из фрейморков, на выбор MVVM Community Toolkit или ReactiveUI.