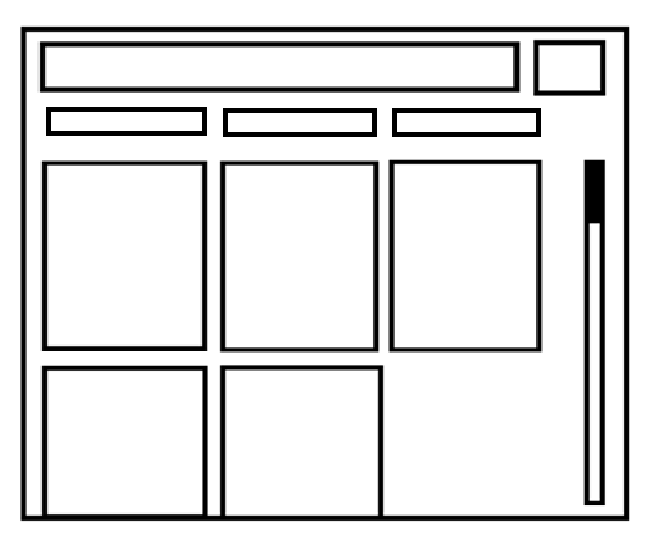
# Практическая работа №27 Разработка приложения

1. Цель работы
   1. Закрепить навыки разработки приложений на Avalonia UI.
2. Литература
   1. Avalonia documentation – Текст : электронный // AvaloniaUI, 2024. – URL: https://docs.avaloniaui.net/
3. Подготовка к работе
   1. Повторить теоретический материал (см.п.2).
   2. Изучить описание лабораторной работы.
4. Основное оборудование
   1. Персональный компьютер.
5. Задание

В работе используется приложение из ПР №27

* 1. Модификация интерфейса приложения
     1. На странице расположите поле ввода, кнопку Поиск, список выбора городов и список прогнозов погоды.
     2. Прогноз погоды должен отображаться в виде карточки, карточки отображаются в виде списка. Общий вид интерфейса должен соответствовать макету:



Также учтите адаптивность интерфейса под мобильные устройства

* + 1. Для списка карточек используйте ItemsControl. В качестве значения для ItemsControl.ItemsPanel установите WrapPanel.

<ItemsControl.ItemsPanel>

<ItemsPanelTemplate>

<WrapPanel/>

</ItemsPanelTemplate>

</ItemsControl.ItemsPanel>

* + 1. ItemsControl расположите внутри ScrollViewer
  1. Добавление записей в список

Пользователь должен вводить название города в поле ввода, нажимать на кнопку «Поиск» и после этого должен заполняться список городов на основе значений, которые вернуло API. Пользователь выбирает один из городов, который будет добавлен в список прогнозов погоды.

* 1. Сохранение данных

После добавления новой карточки необходимо сохранить на устройстве список городов, карточки которых отображаются в приложении

* 1. Для сохранения данных требуется использовать класс Preferences в проекте. Ознакомьтесь с его содержимым.
     1. Для сохранения данных воспользоваться методом

await Preferences.Save(“ключ”, данные)

* + 1. При запуске приложения загружайте данные из списка при помощи метода

var data = await Preferences.Load(“ключ”, знач\_по\_умолч)

* + 1. Для выполнения асинхронных операций в конструкторе ViewModel используйте

TaskNotifier? \_initializationTask;

public Task? InitializationTask

{

get => \_initializationTask;

set => SetPropertyAndNotifyOnCompletion(ref \_initializationTask, value);

}

public WeatherViewModel(WeatherService weatherService)

{

\_weatherService = weatherService;

InitializationTask = Task.Run(async () =>

{

//Асинхронный код

});

}

* 1. Обновление данных
     1. Добавить на карточку кнопки «Обновить» и «Удалить»
     2. При нажатии «Удалить» удаляйте карточку из списка прогнозов
     3. При нажатии «Обновить» обновляйте значения данных прогноза до актуальных на текущий момент.

1. Порядок выполнения работы
   1. Выполнить все задания из п.5.
   2. Ответить на контрольные вопросы.
2. Содержание отчета
   1. Титульный лист
   2. Цель работы
   3. Ответы на контрольные вопросы
   4. Вывод
3. Контрольные вопросы
   1. Как использовать асинхронные операции при инициализации объекта?
   2. Как работает класс Preferences, предоставленный преподавателем?