**Отчет по проекту "Калькулятор ИМТ"**

**1. Введение**

Данный отчет описывает процесс разработки кроссплатформенного приложения "Калькулятор ИМТ" с использованием технологии .NET MAUI. Целью проекта являлась разработка интуитивно понятного, функционального и визуально привлекательного приложения для расчета индекса массы тела (ИМТ). Проект был выполнен в рамках "Профильного уровня" учебной практики и демонстрирует применение знаний, полученных при изучении базовых глав .NET MAUI.

**2. Основная часть**

**2.1. Архитектура и технологии**

Приложение было разработано с использованием следующих ключевых технологий и подходов:

* **.NET MAUI:** Кроссплатформенный фреймворк для создания UI.
* **XAML:** Декларативный язык для определения пользовательского интерфейса.
* **C#:** Основной язык программирования для реализации логики.
* **Паттерн MVVM (Model-View-ViewModel):** Для четкого разделения логики представления (View), данных и команд (ViewModel) и моделей данных (Model).
* **SQLite:** Локальная база данных для хранения истории расчетов.
* **Dependency Injection (DI):** Для внедрения зависимостей (например, репозитория базы данных в ViewModel).

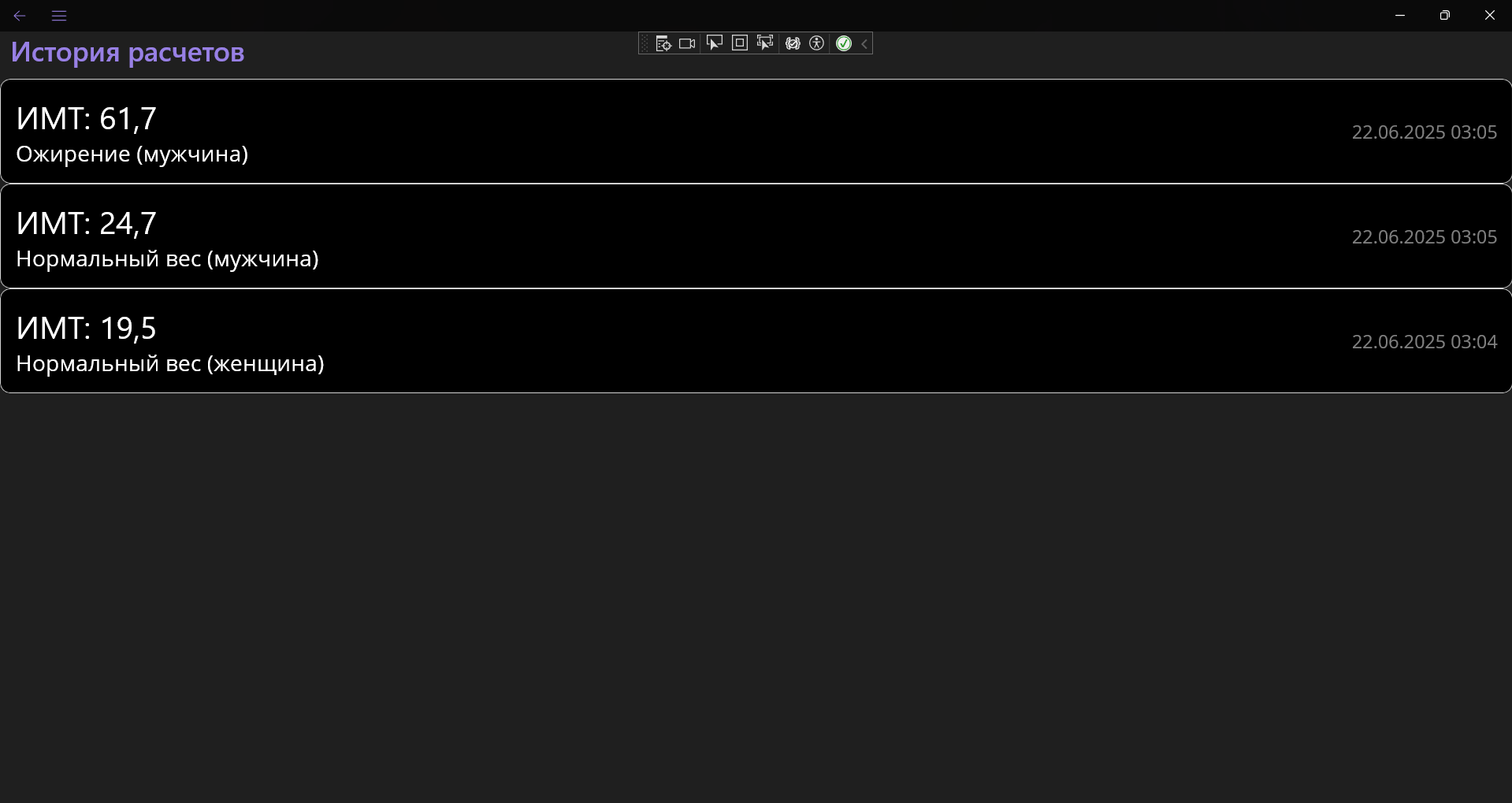
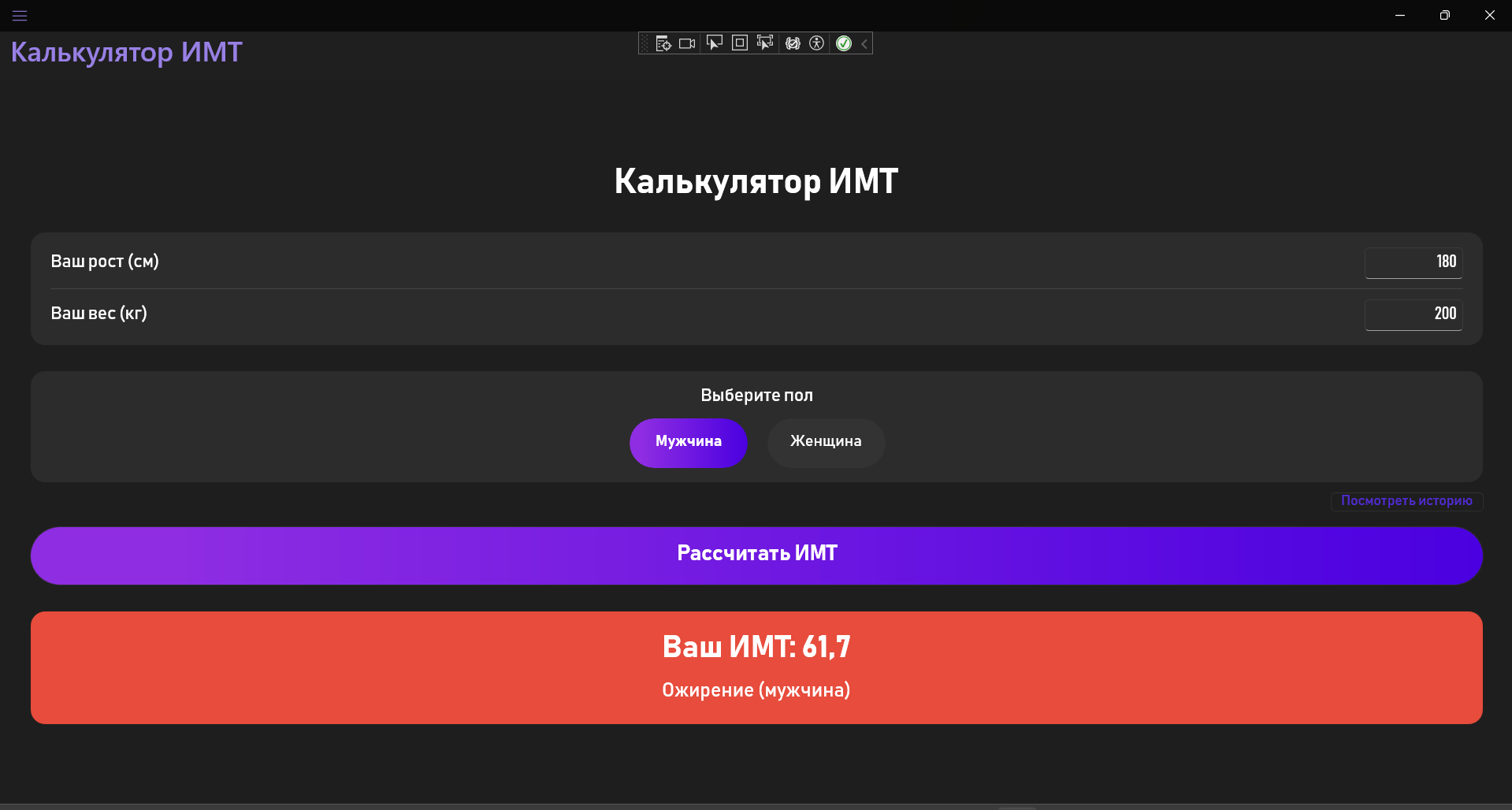
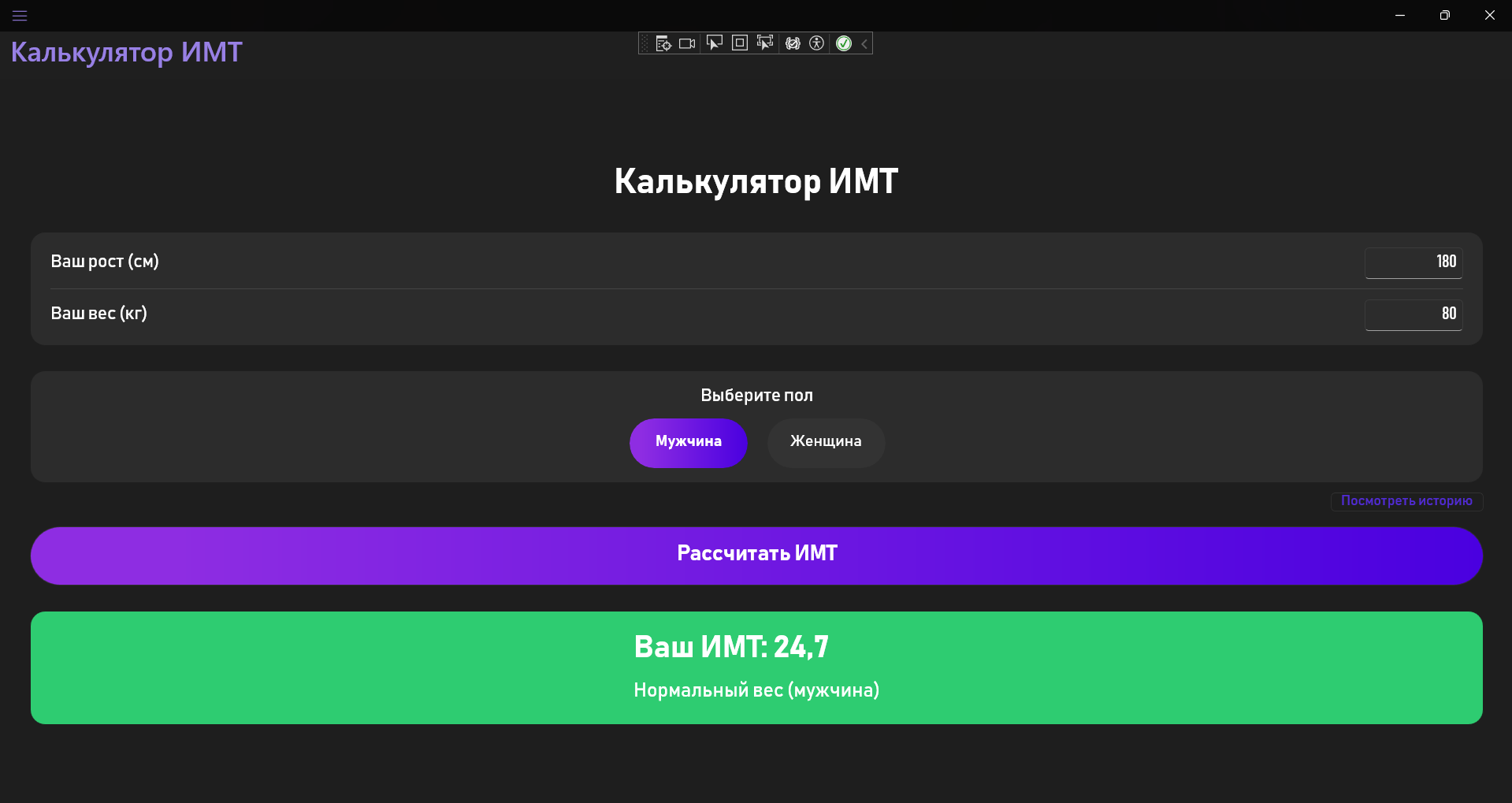
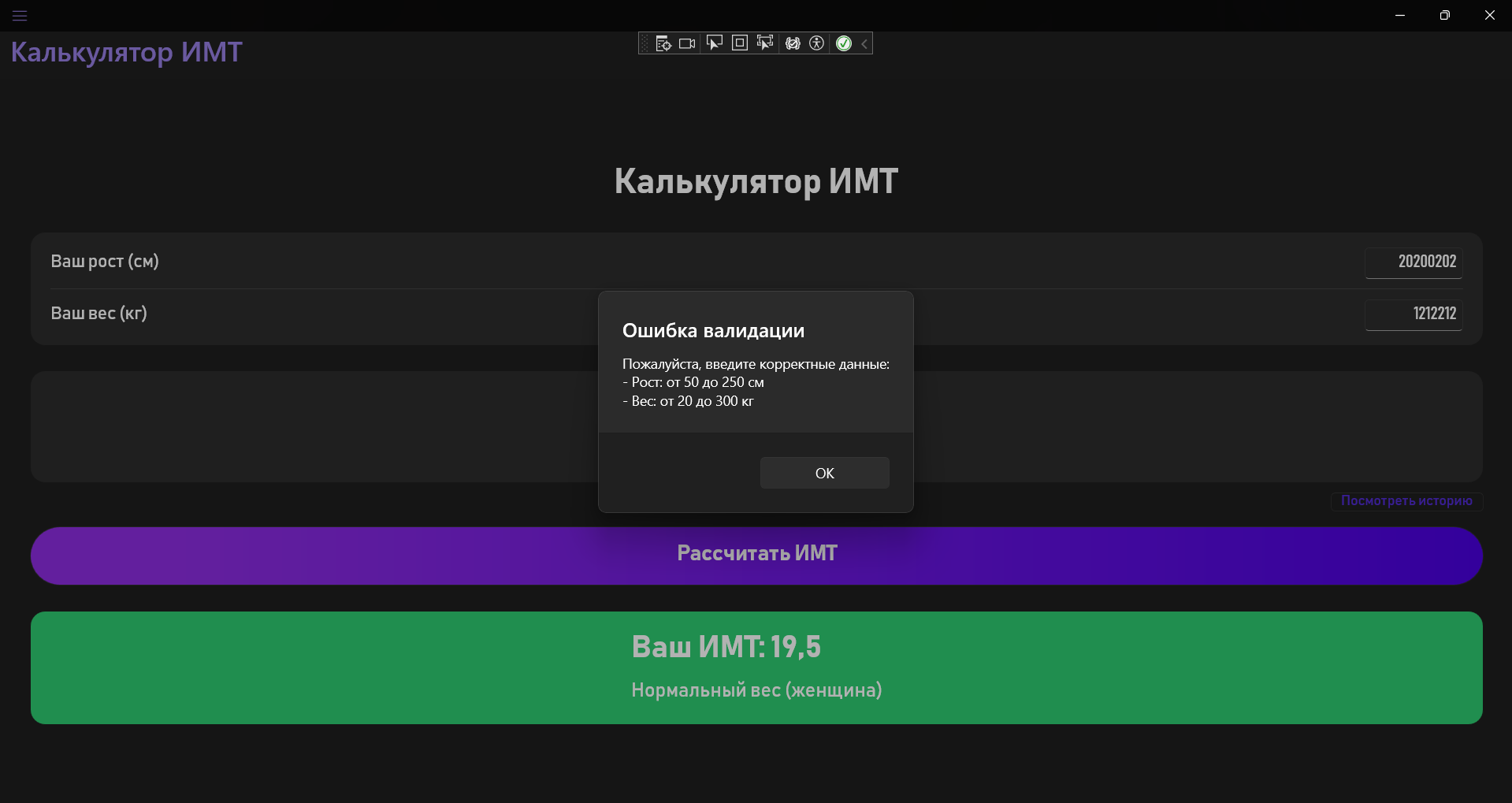
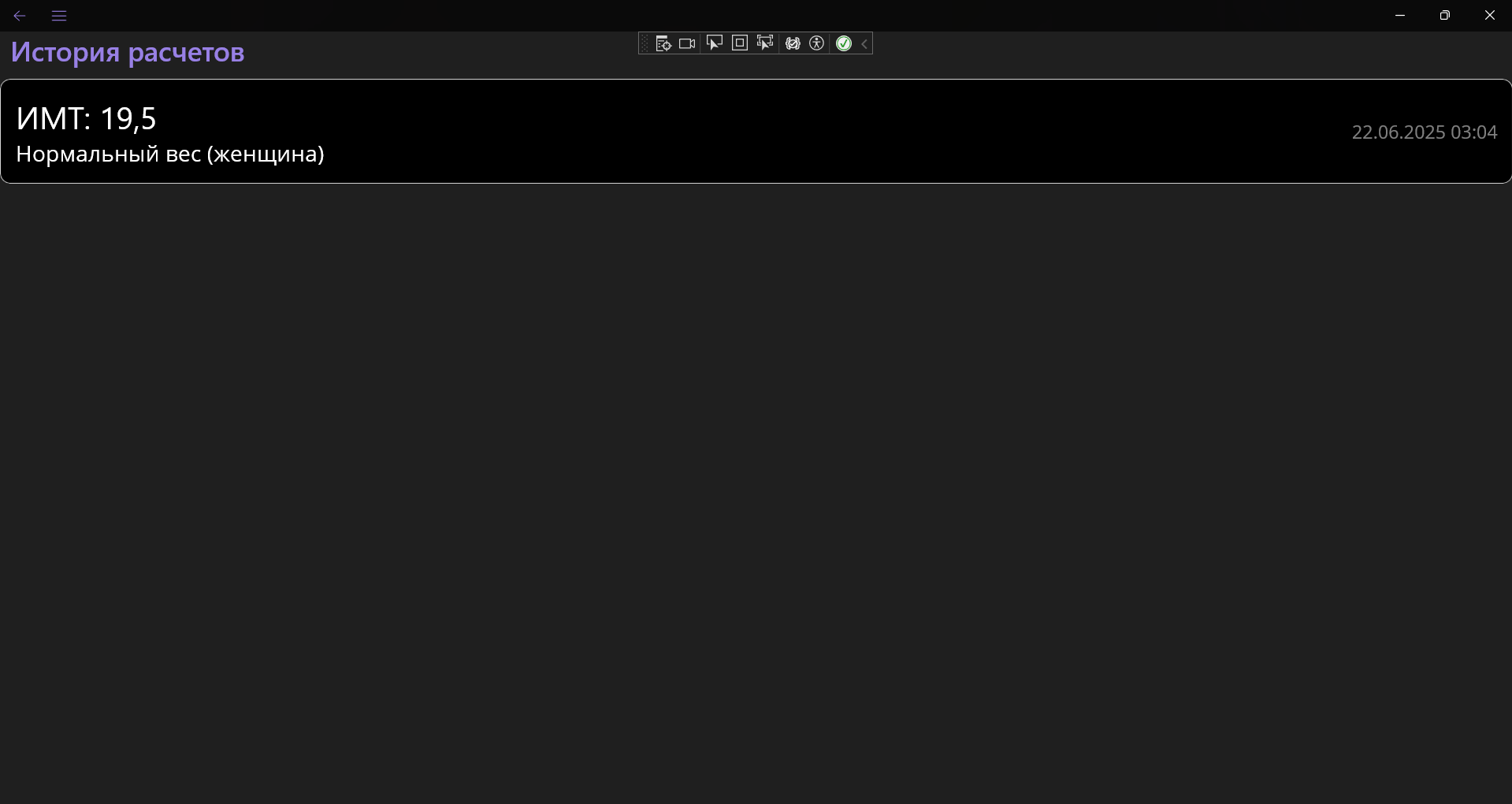
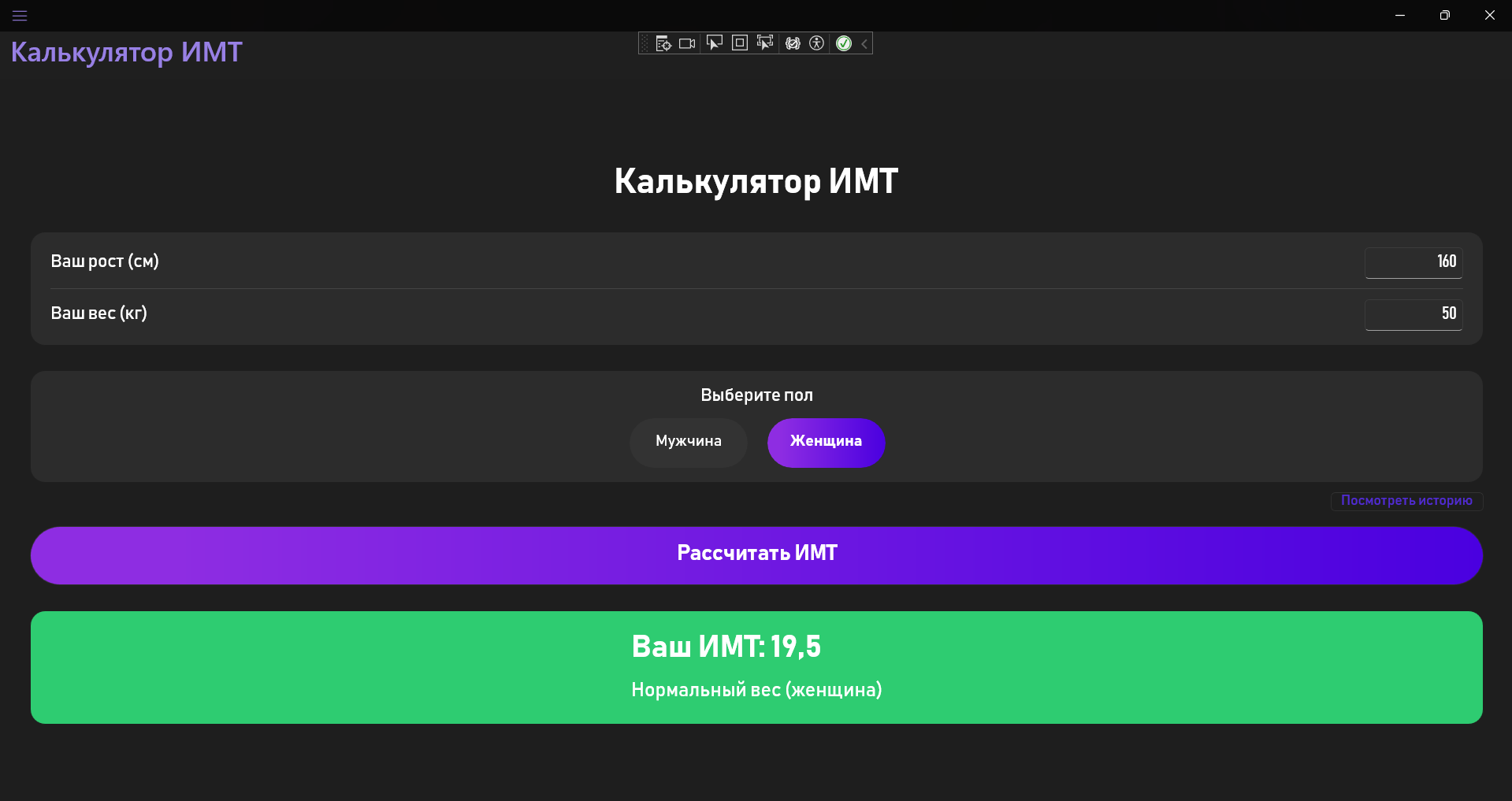
**2.2. Реализация функциональных требований**

* **Главный экран (BmiCalculatorPage.xaml):**
  + **Ввод данных:** Реализованы поля Entry для ввода роста (в см) и веса (в кг). В ViewModel добавлена логика валидации, которая проверяет, что введенные значения находятся в допустимых диапазонах (рост 50-250 см, вес 20-300 кг). В случае некорректного ввода отображается DisplayAlert с сообщением об ошибке.
  + **Выбор пола:** Реализован с помощью двух кнопок. Выбранное состояние подсвечивается с помощью DataTrigger в XAML, который меняет стиль кнопки при изменении свойства SelectedGender в ViewModel. Выбор пола является обязательным для активации кнопки расчета.
  + **Кнопка "Рассчитать ИМТ":** Доступность кнопки управляется через ICommand.CanExecute в ViewModel и становится активной только после ввода всех необходимых данных (рост, вес, пол).
* **Результаты:**
  + **Отображение ИМТ:** После расчета результат (значение и классификация) отображается в специальной рамке Frame. StringFormat используется для форматирования значения до одного знака после запятой.
  + **Графическое представление:** Фон рамки с результатом динамически изменяет свой цвет в зависимости от классификации ИМТ (синий для недостаточного веса, зеленый для нормального, желтый для избыточного, красный для ожирения).
  + **Анимация:** Появление рамки с результатом анимировано с помощью FadeTo и ScaleTo для плавности и улучшения пользовательского опыта.
* **Дополнительные функции (История расчетов):**
  + **Сохранение результатов:** После каждого успешного расчета создается объект BmiResultRecord (модель данных), который сохраняется в локальную базу данных SQLite с помощью класса-репозитория BmiRepository.
  + **Просмотр истории:** На главном экране добавлена кнопка "Посмотреть историю", которая осуществляет навигацию на страницу HistoryPage.xaml. На этой странице с помощью CollectionView отображается список всех сохраненных результатов, отсортированных по дате (от новых к старым).

**3. Заключение**

В ходе работы над проектом были успешно реализованы все функциональные и нефункциональные требования технического задания. Разработанное приложение "Калькулятор ИМТ" демонстрирует практическое применение ключевых концепций .NET MAUI, включая создание адаптивного UI на XAML, использование паттерна MVVM, привязку данных, работу с локальной базой данных SQLite и реализацию навигации. Проект является полноценным и готовым к использованию продуктом.

**4. Скриншоты и примеры**

*Рис. П1 – П6. Демонстрация работы приложения 'Калькулятор ИМТ'*

**5. Документация**

* **Ссылка на GitHub с исходным кодом:** https://github.com/RaniZee/.NET-MAUI-Showcase-Cross-Platform-Applications
* **Описание выполненных шагов:**
  1. Настроена ветка feature/bmi-calculator и синхронизирована с main.
  2. Разработана архитектура приложения с использованием MVVM.
  3. Создана страница BmiCalculatorPage с пользовательским интерфейсом.
  4. Реализована ViewModel BmiCalculatorViewModel с логикой расчета и валидации.
  5. Настроены стили, градиенты и анимации для улучшения UI/UX.
  6. Интегрирована работа с базой данных SQLite через модель BmiResultRecord и репозиторий BmiRepository.
  7. Создана страница HistoryPage для отображения истории расчетов.
  8. Настроена навигация и внедрение зависимостей через MauiProgram.cs и AppShell.
  9. Проведено финальное тестирование всех функций.
  10. Работа слита в основную ветку main.