Министерство образования Московской области

ГБПОУ МО «Колледж «Коломна»

09.02.07

КУРСОВОЙ ПРОЕКТ

ПМ. 02 Осуществление интеграции программных модулей

Тема: Разработка информационной системы для фитнес клуба ZeVs

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

ККОО.КП1223.000ПЗ

|  |  |
| --- | --- |
| Студент: | Скорочкин Н.А. |
| Руководитель: | Карташова Е.В. |
| Нормоконтролер: | Грушникова Т.Н. |
| Дата защиты: | Оценка |

2024

Министерство образования московской области

ГБПОУ МО «Колледж «Коломна»

ЗАДАНИЕ

на курсовой проект

ПМ.02 Осуществление интеграции программных модулей

студента очной формы обучения 4 курса группы 42112

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Тема: Разработка информационной системы для фитнес клуба

Разработать информационную систему в соответствии с темой и приложение для работы с ней. Разработанная информационная система должна отражать основные бизнес-процессы в соответствии с техническим заданием на разработку. Для защиты информации необходимо разграничить права пользователей системы. Разработанные формы должны обеспечить возможность пользователям осуществлять работу с данными в соответствии с их правами.

При разработке информационной системы необходимо решить следующие задачи:

* сформулировать цель разработки информационной системы;
* описать предметную область, для которой разрабатывается информационная система
* провести анализ предметной области, выявить основные бизнес-процессы и пользователей, которые будут с ней работать.
* определить круг запросов и задач, которые предполагается решать с использованием созданной информационной системы;
* написать техническое задание на разработку информационной системы;
* провести функциональное моделирование и объектно-ориентированное проектирование информационной системы;
* провести проектирование интерфейса информационной системы:
* разработать модель данных и реализовать ее в выбранной СУБД
* разработать программное решение для информационной системы;
* оценить качество разработанной системы путем тестирования основного функционала.

Объем курсовой работы:

* + 1. Пояснительная записка (25-50 листов печатного текста формата А4):

Введение

1. Разработка системного проекта

1.1 Описание предметной области

1.2 Анализ предметной области

1.3 Техническое задание

2 Проектирование информационной системы

2.1 Функциональное моделирование

2.2 Проектирование интерфейсов приложения

3 Разработка информационной системы

3.1 Модель данных

3.2 Структура проекта

4 Оценка качества программного обеспечения

Выводы и заключение

Список литературы

Приложение А – Код программных модулей

* + 1. Презентация для защиты курсового проекта (10-15 слайдов)

Дата выдачи задания \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Срок окончания \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Зав структурным подразделением \_\_\_\_\_\_\_\_\_\_ Емельянова В. А.

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_ Карташова Е.В.

СОДЕржАние

|  |  |
| --- | --- |
| Введение |  |
| 1. Разработка системного проекта | 6 |
| 1.1 Описание предметной области |  |
| 1.2 Анализ предметной области | 7 |
| 1.3 Техническое задание |  |
| 2 Проектирование информационной системы | 12 |
| 2.1 Функциональное моделирование |  |
| 2.2 Проектирование интерфейсов приложения | 16 |
| 3 Разработка информационной системы | 23 |
| 3.1 Модель данных |  |
| 3.2 Структура проекта | 26 |
| 4 Оценка качества программного обеспечения | 30 |
| Выводы и заключение | 34 |
| Список литературы | 35 |
| Приложение А – Код программных модулей | 36 |

Целью курсового проекта является.

При разработке информационной системы необходимо решить следующие задачи:

* сформулировать цель разработки информационной системы;
* описать предметную область, для которой разрабатывается информационная система
* провести анализ предметной области, выявить основные бизнес-процессы и пользователей, которые будут с ней работать.
* определить круг запросов и задач, которые предполагается решать с использованием созданной информационной системы;
* написать техническое задание на разработку информационной системы;
* провести функциональное моделирование и объектно-ориентированное проектирование информационной системы;
* провести проектирование интерфейса информационной системы:
* разработать модель данных и реализовать ее в выбранной СУБД
* разработать программное решение для информационной системы;
* оценить качество разработанной системы путем тестирования основного функционала.

1. РАЗРАБОТКА СИСТЕМНОГО ПРОЕКТА
   1. Описание предметной области

Информационная система для фитнес-клуба — это комплексное решение, направленное на предоставление широкого спектра услуг для любителей и профессионалов, таких как тренировки, групповые занятия, персональные тренировки, а также доступ к современному оборудованию и инфраструктуре. В современном мире, где физическая активность играет важную роль в жизни многих людей, информационная система для фитнес-клуба становится не просто инструментом управления, а важным элементом здорового образа жизни и личного развития. Подписка на фитнес-клуб может быть значительным вложением, но это вложение окупается с лихвой, особенно если клуб выбран правильно и соответствует потребностям клиента.

Информационная система для фитнес-клуба также играет важную роль в поддержке физического образования и развития спортивных талантов. Доступ к качественным тренировкам и оборудованию позволяет начинающим спортсменам развивать свои навыки, а профессионалам — совершенствовать мастерство. Это способствует росту физической культуры и созданию новых спортивных достижений, которые обогащают нашу жизнь. Кроме того, информационная система для фитнес-клуба способствует сохранению традиций и культурного наследия. Многие виды спорта и физических упражнений имеют богатую историю и являются важной частью культурного наследия разных народов. Поддержка и развитие таких видов спорта помогает сохранить эти традиции и передать их будущим поколениям.

* 1. Анализ предметной области

Цель информационной системы для фитнес клуба – автоматизация деятельности магазина по продаже музыкальный инструментов и др.

Задачи данной информационной системы:

а. Автоматизация регистрации данных о клиенте. Необходимо, чтобы данные о клиенте сохранялись в нашей ИС. Эти данные используются для идентификации клиента.

б. Автоматизация регистрации тренеров.

Необходимо хранить всю информацию о тренере. Эта информация используется как для просмотра специализации, так и для просмотра его бронированний.

в. Автоматизация бронирования занятий. Необходимо в ИС заносить всю информацию о брони занятий пользователем.

Список пользователей ИС ремонта электронной техники:

– администратор;

* 1. Техническое задание

Введение

Фитнес клуб ZeVs

Настоящее техническое задание распространяется на разработку автоматизированной информационной системы «Фитнес клуб ZeVs», предназначенной для автоматизации учета бронирования, клиентов и тренеров.

Целью создания автоматизированной информационной системы является сокращение времени на получение сведений о бронировании занятий клиентом, клиентах и тренеров. Система обеспечивает повышение эффективности процесса обработки и движения документов, снижает затраты на ведение документооборота, повышает точность и достоверность данных.

Работа выполняется в рамках проекта автоматизации управления торговым предприятием.

Основание для разработки

Основанием для разработки является договор № \_\_\_\_\_\_\_ от \_\_\_\_\_\_\_

Организация, утвердившая договор: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Наименование работы: Информационная система «Финтнес клуб ZeVs»

Назначение разработки.

Информационная система «Фитнес клуб ZeVs» предназначена для автоматизации учета бронирования, ипользуемых клиентами в системе. Пользователями системы являются администраторы фитнес клуба, специалисты отдела учета и сотрудники оформления документов. Система должна обеспечить учет бронирований, поступивших в систему, а также фиксацию всех операций, связанных с их изменением.  
Пользователи системы:

Администратор — отвечает за все бизнес процессы.

Основные функции системы:

(1.1.1) Учет поступления бронирований в систему фитнес клуба:

* Оформление и учет поступления бронирования в систему от клиентов

(1.1.2) Оформление и учет клиентов:

* Регистрация новых клиентов.
* Просмотр их профилей

(1.1.3.) Оформление и учет тренеров:

* Регистрания новых тренеров в системе.
* Просмотр профилей тренеров и их специлизаций.

Требования к программе

(1.1.3.1) Требования к функциональным характеристикам

Автоматизированная информационная система «фитнес клуб ZeVs» должна обеспечивать выполнение следующих функций:

* ввод, хранение, поиск и обработку информации о бронировании клиентами системы;
* ведение журнала регистрации клиентов и их действий;
* своевременное получение информации о доступных тренерах;

Нормативно-справочная информация автоматизированной информационной системы «фитнес клуба ZeVs» представлена справочниками:

а) клиентов (юридические и физические лица);

б) тренеров;

в) бронирований;

Первичными документами для учета товаров в системе являются:

* приходные документы — акты приема бронирований, содержащие дату поступления брони, сведения о клиенте, описание услуги, срок выполнения;

Выходными данными являются:

* отчет по услугами за определенный период, включающий перечень забронированных занятий, затраты на зарплату тренерам по каждому занятию и общую сумму всего;
* отчет по новым клиентам за определенных срок;
* оборотная ведомость за период (для клиента и теренера);
* рейтинг клиентов по объему предоставленных услуг;
* рейтинг занятий по частоте бронирований;

Все отчеты должны формироваться на даты, заданные пользователем, и содержать итоговые значения. В приложении к техническому заданию требуется включить образцы всех первичных документов и образцы всех отчетов.

В программе необходимо предусмотреть:

* возможность резервного сохранения данных;
* наличие встроенных подсказок;

(1.1.3.2) Требования к надежности

Программное обеспечение должно соответствовать требованиям надежности и безопасности, включая:

* возможность восстановления после сбоев (отключение питания, сбои в операционной системе и т.д.);
* возможность резервного копирования информационной базы;
* разграничение пользовательских прав;
* предотвращение несанкционированного копирования программы.

Необходимо предусмотреть контроль вводимой информации, обработку или блокировку некорректных действий пользователя. Все поля первичных документов должны быть заполнены.

(1.1.3.3) Требования к составу и параметрам технических средств

Минимальные системные требования для работы программного продукта:

* тактовая частота процессора – 1.9 ГГЦ
* объем оперативной памяти – 8 Гб
* объем свободного дискового пространства – 30 Гб
* разрешение монитора – 1920 x 1080

(1.1.3.4) Требования к информационной и программной совместимости

Программа должна работать в операционных системах Windows... Все формируемые отчеты должны иметь возможность экспортирования в редактор электронных таблиц MS Office Excel.

(1.1.3.5) Требования к транспортировке и хранению

Установка программы идет по инструкции на сайте разработчика.

Программная документация также доступна на сайте разработчика в электронном виде.

(1.1.3.6) Специальные требования

Программное обеспечение должно иметь дружественный интерфейс, рассчитанный на пользователя средней квалификации в области компьютерной грамотности.

Проект будет выполняться поэтапно, с совместимостью модулей, разработанных в разное время.

(1.1.4) Требования к программной документации

В ходе разработки программы должна быть подготовлена следующая документация:

* текст программы;
* описание программы;
* программа и методика испытаний;
* руководство пользователя.

(1.1.5) Технико-экономическое обоснование

Использование автоматизированной системы «фитнес клуб ZeVs» позволит сократить время на выполнение операций по регистрации клиентов, оформлением бронирования клиентом системы и регистрации тренера в системе. Ожидается значительное уменьшение времени на подготовку отчетов, проверку данных и уменьшение ошибок, связанных с ручным вводом информации. Экономический эффект будет достигнут за счет повышения скорости и точности обработки данных, сокращения ручного труда, а также улучшения контроля за складскими операциями и бизнес-процессами.

1. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ
   1. Функциональное моделирование

Функциональное моделирование диаграммы — это графическое представление функций предприятия в определённой области. На рисунке 1 представлена диаграмма IDEF0.

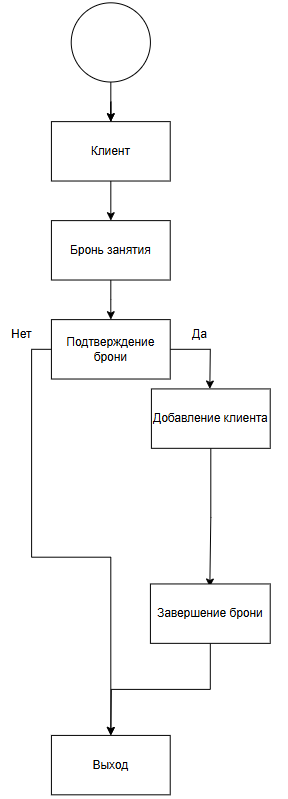


Рисунок 1 – Диаграмма IDEF0

На рисунке 2 представлена диаграмма DFD.

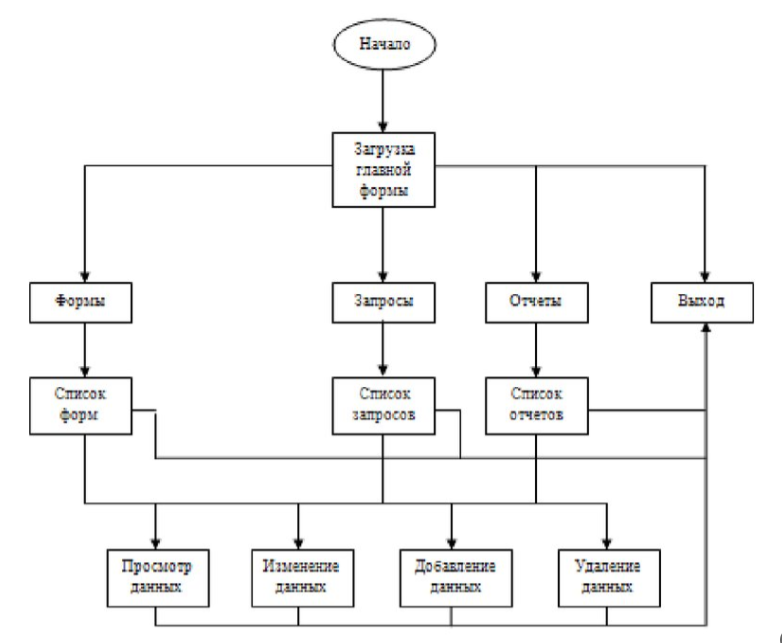


Рисунок 2 - Диаграмма DFD

На рисунке 3 представлена диаграмма деятельности.

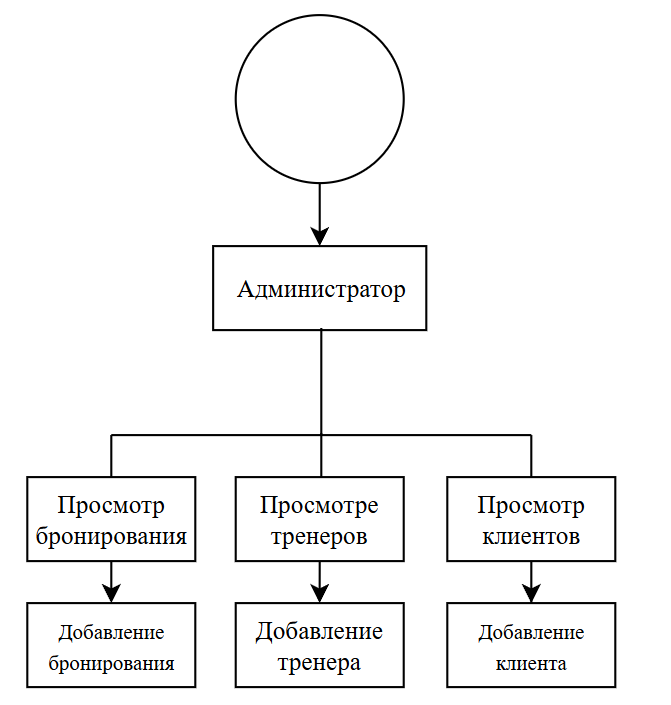


Рисунок 3 - диаграмма деятельности

На рисунке 4 представлена диаграмма последовательности.

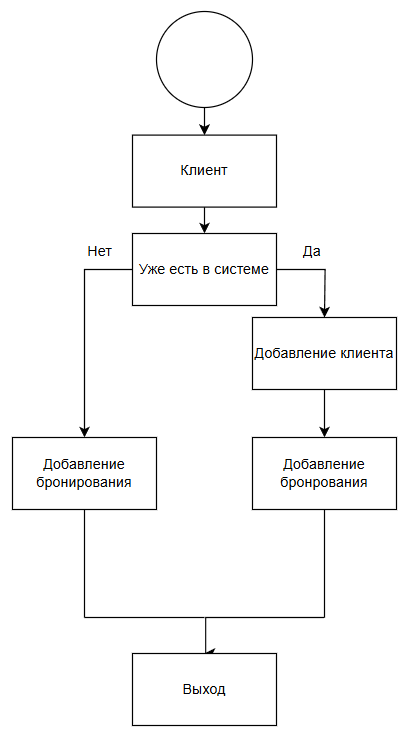


Рисунок 4 - Диаграмма последовательности

* 1. Проектирование интерфейсов приложения

На странице EmptyPage (Рисунок 5) находятся следующие элементы:

* TopMenu – навигационное меню;
* HomeButton – кнопка перехода на страницу EmptyPage
* ClientsButton – кнопка перехода на страницу ClientsPage
* TrainersButton – кнопка перехода на страницу TrainersPage
* BookingsButton – кнопка перехода на страницу BookingsPage

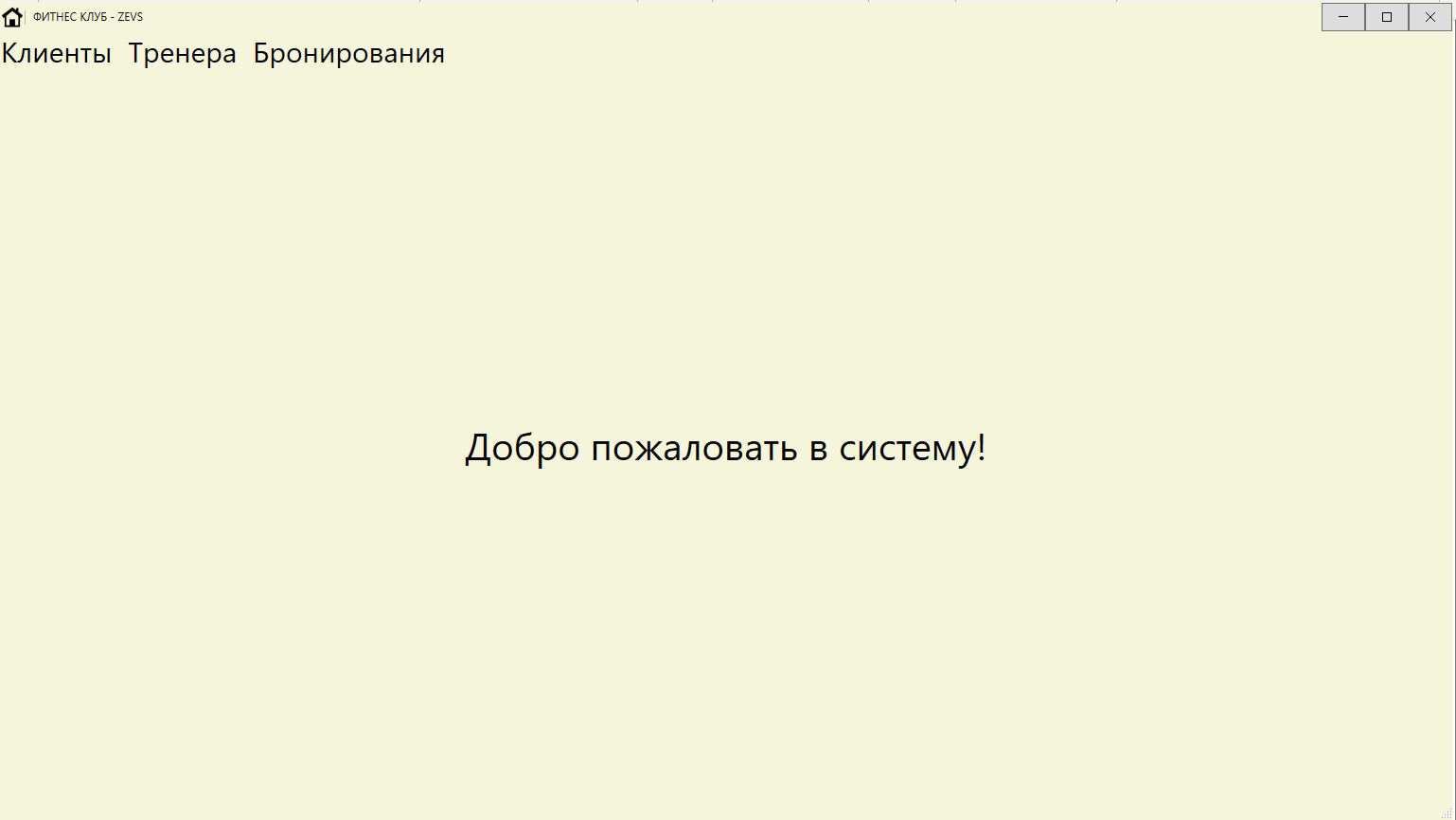


Рисунок 5 – Страница EmptyPage

На странице ClientsPage (Рисунок 6) находятся следующие элементы:

* AddClientButton – кнопка добавления пользователя
* ArrowLeftButton – кнопка для перелистывания страницы на лево
* ArrowRightButton – кнопка для перелистывания страницы на право
* ClientCard – клиентская карточка ведущая в профиль

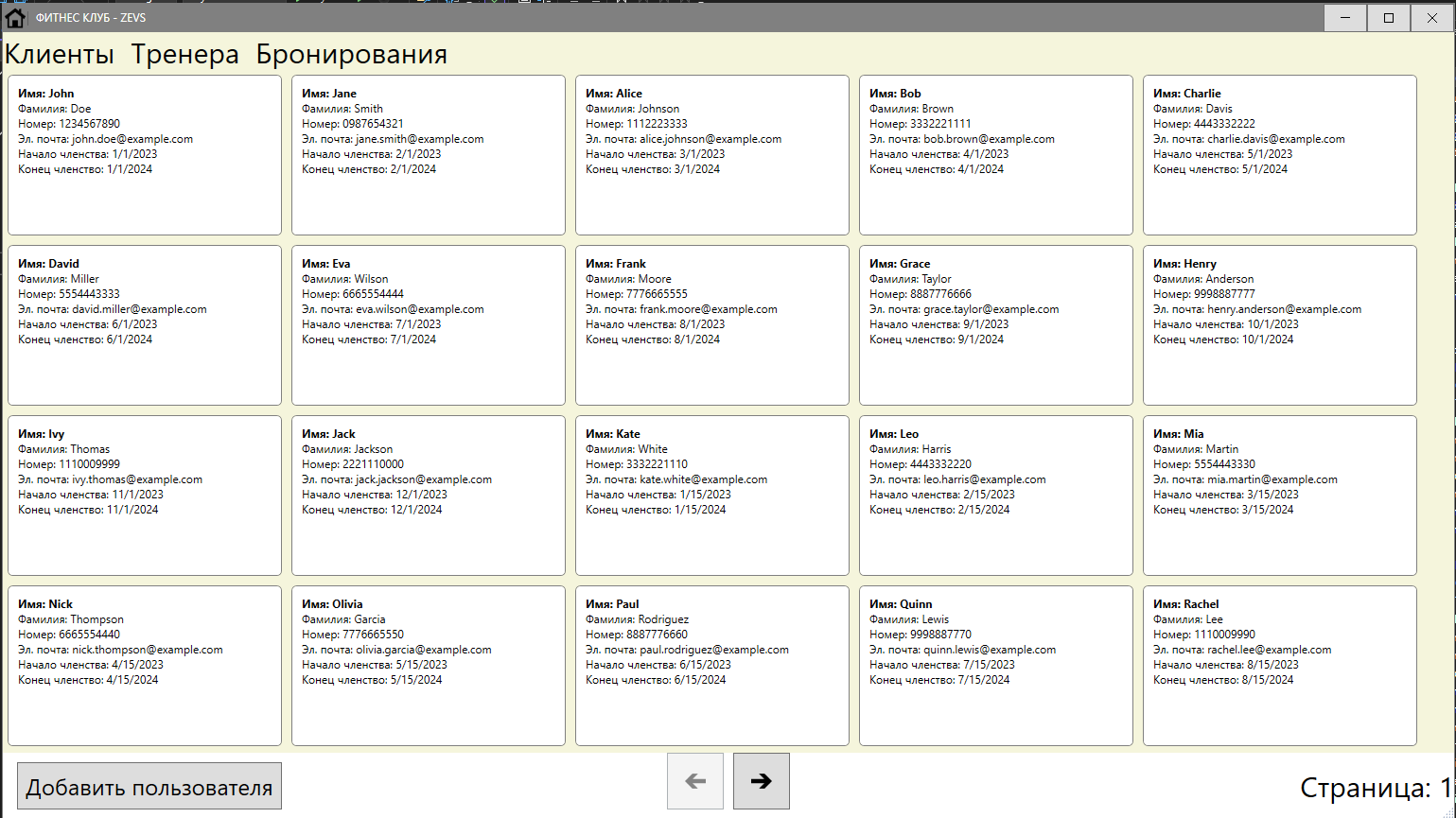


Рисунок 6 – Страница ClientsPage

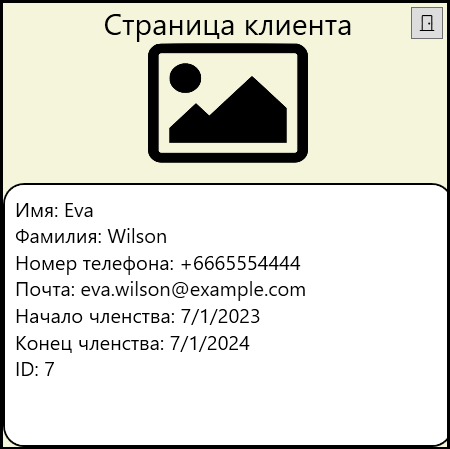


Рисунок 7 – ClientDetailsPage

На Окне AddClient (Рисунок 8) находятся следующие элементы:

* AddButton – кнопка для добавления нового клиента
* CloseButton – кнопка для отмены действия
* NameTextBox – поле ввода имени клиента
* LastNameTextBox – поле ввода фамилии клиента
* EmailTextBox – поле ввода почты клиента
* PhoneNumber – поле ввода номера телефона клиента



Рисунок 8 – Окно AddClient

На странице TrainersPage (Рисунок 9) находятся следующие элементы:

* AddTrainerButton – кнопка добавления нового тренера
* ArrowLeftButton – кнопка для перелистывания страницы назад
* ArrowRightButton – кнопка для перелистывания страницы вперед
* TrainerCard – тренерская карточка перехода на страницу тренера

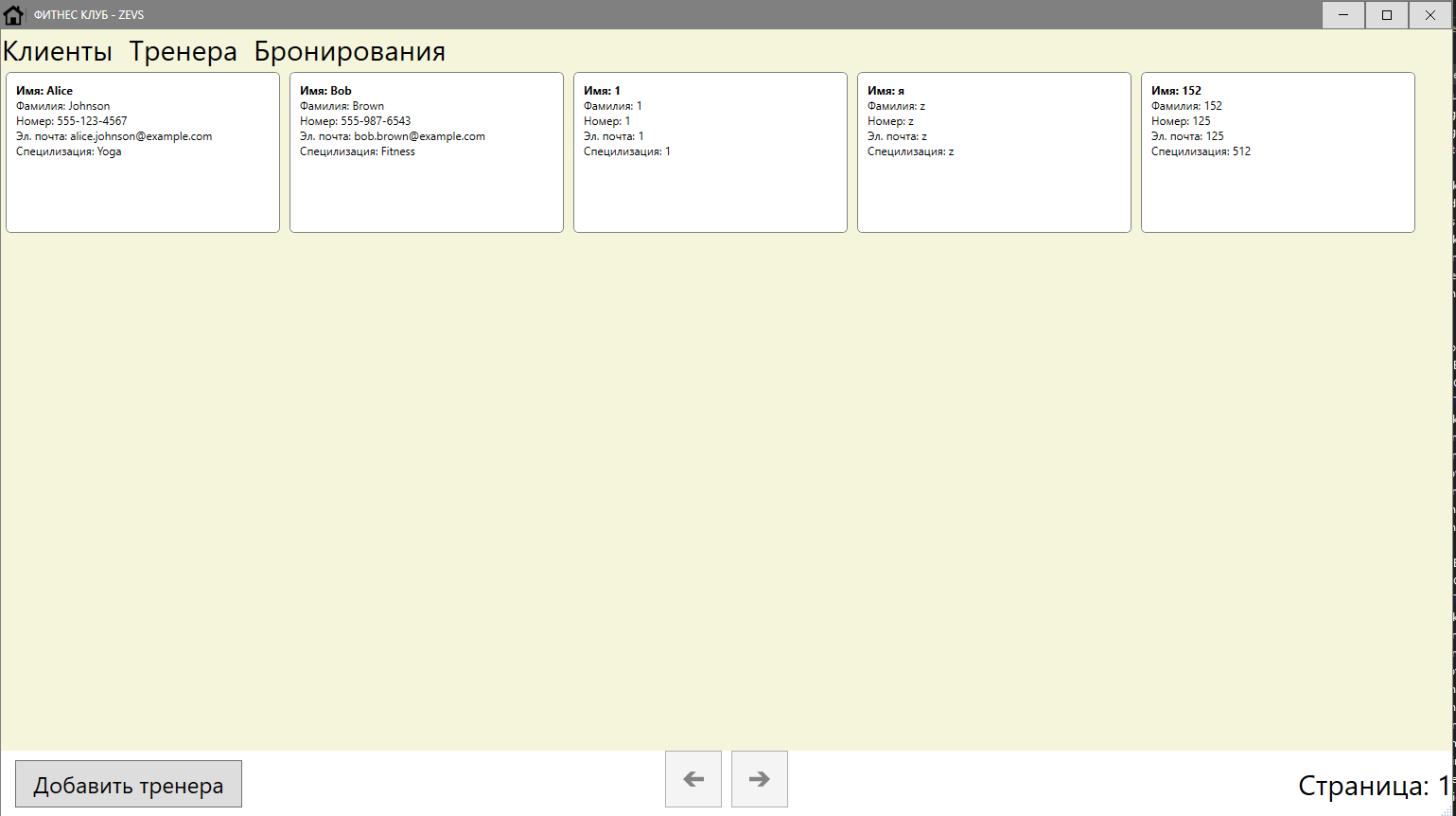


Рисунок 9 – Страница TrainersPage

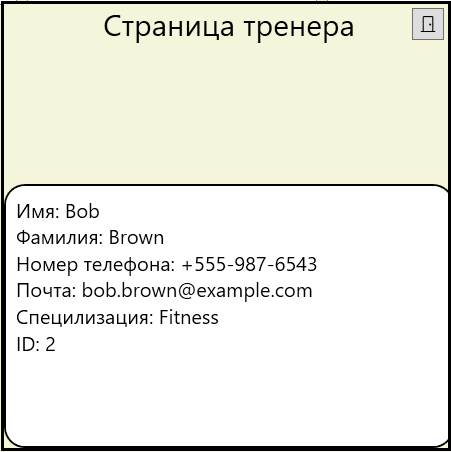


Рисунок 10 – Страница TrainerDetailsPage

На Окне AddTrainer (Рисунок 11) находятся следующие элементы:

* AddButton – кнопка добавления тренера
* CloseButton – кнопка закрытия окна
* NameTextBox – поле ввода имени тренера
* LastNameTextBox – поле ввода фамилии тренера
* EmailTextBox – поле ввода почты тренера
* PhoneNumberTextBox – поле ввода номера телефона тренера
* SpecializationPhoneNumber – поле ввода специализации тренера

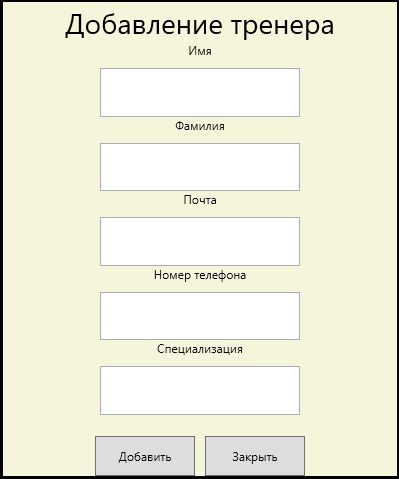


Рисунок 11 – Окно AddTrainer

На странице BookingsPage (Рисунок 12) находятся следующие элементы:

* AddBooking – кнопка для добавления нового бронирования
* ArrowLeftButton – кнопка для перелистывания страницы назад
* ArrowRightButton – кнока для перелистывания страницы вперед

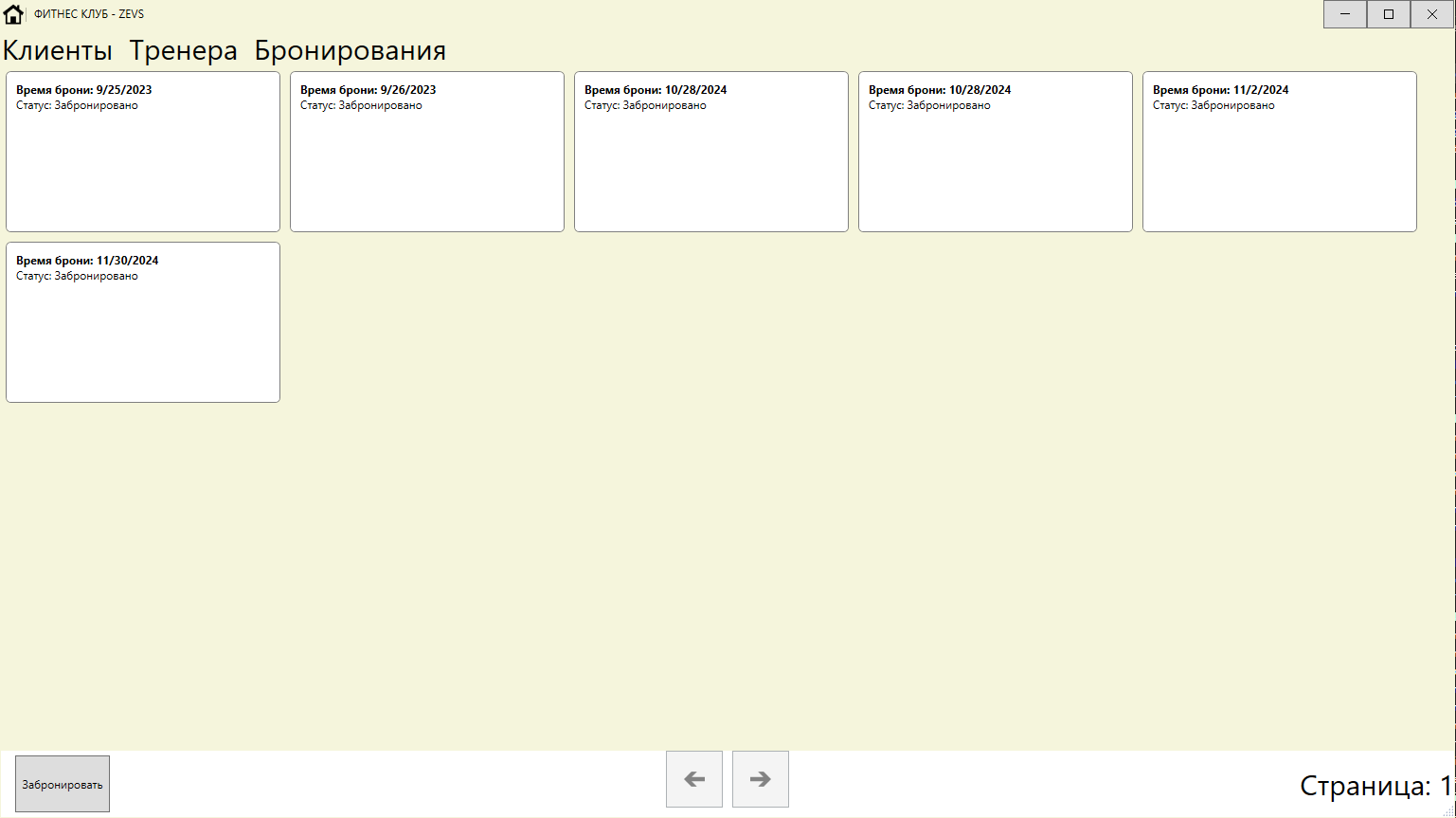


Рисунок 12 – Страница BookingsPage

На окне AddBooking (Рисунок 13) находятся следующие элементы:

* AddBooking – кнопка для добавления нового бронирования
* CloseButton – кнопка для закрытия окна
* SheduleComboBox – кнопка для выбора расписания занятия
* DatePicker – выбор даты занятия
* PhoneNumberTextBox – поле ввода номера телефона клиента

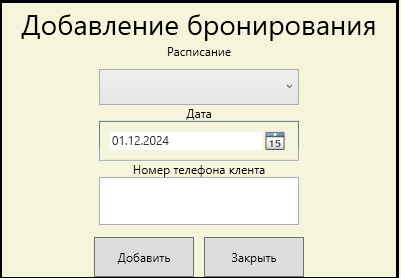


Рисунок 13 – Окно AddBooking

1. РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ
   1. Модель данных

Модель данных реализована в Microsoft SqlServer.

В результате разработки базы данных были спроектированы следующие таблицы:

1. Clients (ClientID, FirstName, LastName, PhoneNumber, Email, MembershipStartDate, MemberShipEndDate);
2. Bookings (BookingID, ClientID, SheduleID, BookingDate, Status);
3. MemberShipTypes (MemberShipTypeId, MemberShipTypeName, Duration, Price);
4. ClientMemberShips (ClientMemberShipID, ClientID, MemberShipTypeID, StartDate, EndDate);
5. Schedule (ScheduleID, ClassId, Date, StartTime, EndTime, MaxCapacity)
6. Classes (ClassID, Description, ClassName, Duration, TrainerID)
7. Trainers (TrainerID, FirstName, LastName, PhoneNumber, Email, Specialization)

На Рисунке 14 представлена база данных, на Рисунках 15 - 21 показаны таблицы.

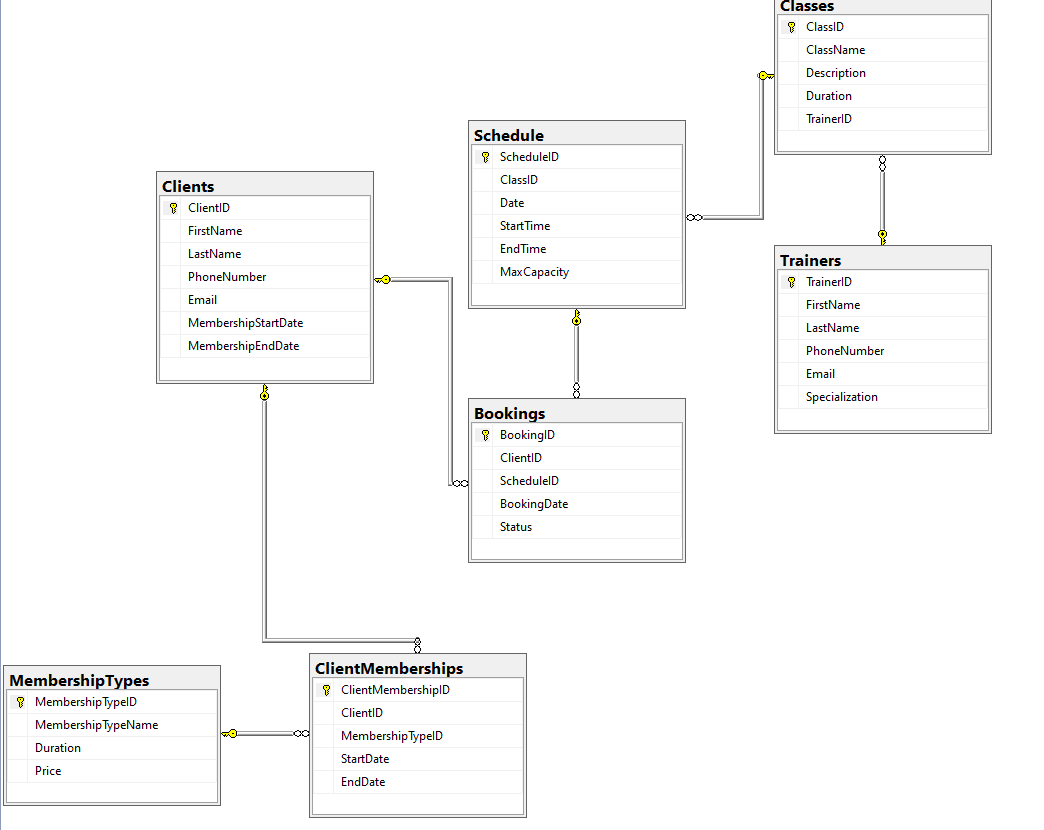


Рисунок 14 – База данных fitnes\_club

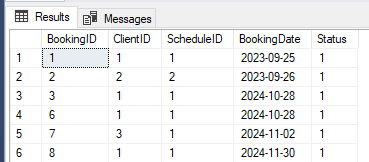


Рисунок 15 – Таблица Bookings



Рисунок 16 - Таблица Classes



Рисунок 17 – Таблица ClientMemberShips

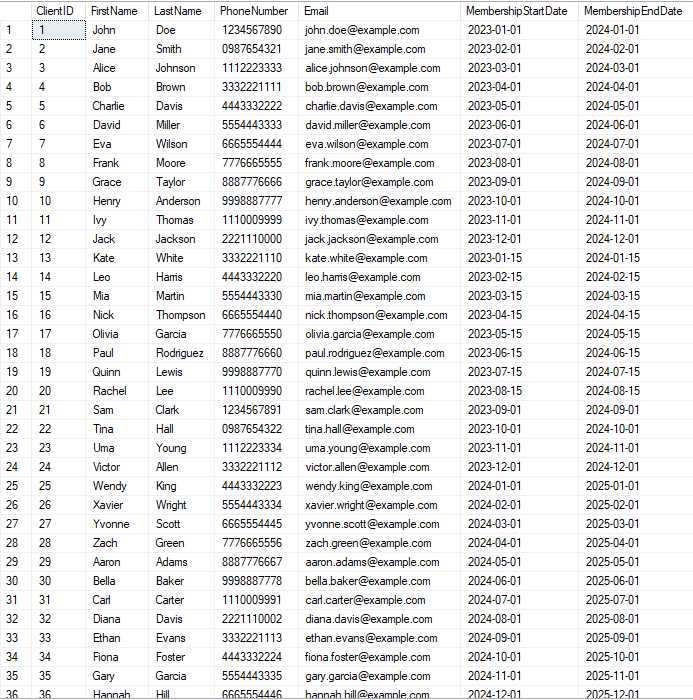


Рисунок 18 – Таблица Clients



Рисунок 19 – Таблица MemberShipTypes



Рисунок 20 – Таблица Schedule

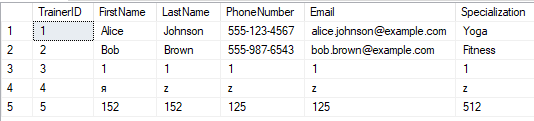


Рисунок 21 – Таблица Trainers

* 1. Структура проекта

В процессе разработки программного решения были спроектированы следующие вещи:

1. Страницы:

* ClientsPage;
* TrainersPage;
* BookingsPage;
* ClientDetailsPage;
* AddClient;
* TrainersDetailsPage;
* AddTrainer;
* AddBooking.

1. Элементы базы данных

* Clients;
* Bookings;
* MemberShipTypes;
* ClientMemberShips;
* Schedules.
* Trainers
* Classes

На Рисунке 22 представлена вся структура программного решения.

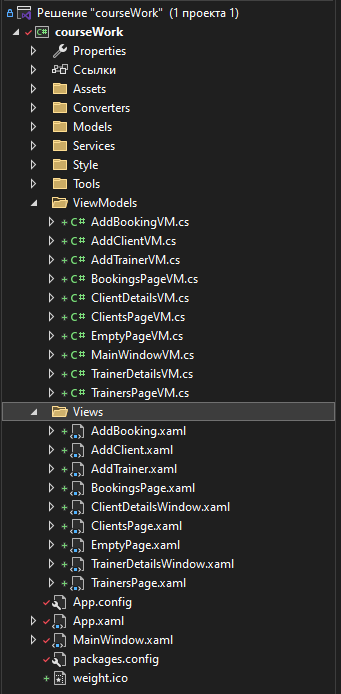


Рисунок 22 – Структура проекта C#

Класс ClientService.cs имеет следующие методы (Рисунок 23):

* private readonly \_context – поле дающие доступ к контексту базы данных;
* public AddClient – метод добавляющий клиента в базу данных;
* public ClientServie – контруктор класса
* public GetClientByPhoneNumber – метод возвращающий пользователя по номеру телефона
* public GetClient – метод получающий массив клиентов.

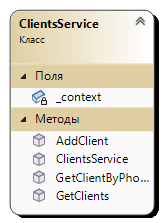


Рисунок 23 – Структура класса ClientService.cs

Класс BookingsService.cs имеет следующие методы (Рисунок 24):

* private readonly \_context – доступ к контексту базы данных;
* public GetBookings – метод получения массива бронирования;
* public SaveBooking – метод сохранения брони;
* public BookingsService() – конструктор класса;

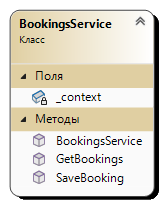


Рисунок 24 – Структура класса BookingsService

Класс TrainersService.cs имеет следующие методы (Рисунок 25):

* private readonly \_context – доступ к контексту базы данных;
* public AddTrainer – метод сохраняющий тренера в базу данных;
* public GetTrainers – метод возвращающий массив тренеров;
* public TrainersService() – конструктор класса;

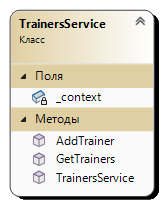


Рисунок 25 – Структура класса TrainersService

Класс ScheduleService.cs имеет следующие методы (Рисунок 26):

* private readonly \_context – доступ к контексту базы данных;
* public GetAllShedules – метод получения всех расписаний;
* public ScheduleService() – конструктор класса;

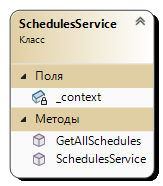


Рисунок 26 – Структура класса ScheduleService

4 ОЦЕНКА КАЧЕСТВА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Сценарий тестирования представлен в таблице 1.

Таблица 1 - Сценарий тестирования

|  |  |  |
| --- | --- | --- |
| Test case # | Название теста | Резюме испытания  (краткое описание) |
| TC\_UI\_1 | Проверка добавления клиента с неуказанным именем | 1. Проверка добавления нового клиента с указанием имени. 2. Проверка невозможности добавления клиента с не указанием имени. |
| TC\_UI\_2 | Проверка добавления тренера с неуказанием специальности | 1. Проверка добавления тренера с неуказанием специальности; 2. Проверка невозможности добавоения клиента с не указанием специальности; |
| TC\_UI\_3 | Проверка добавления бронирования с неуказанием даты | 1. Проверка добавления бронированием с указанием даты 2. Проверка невозможности добавления бронирования с неуказанием даты |

Тест – кейсы представлены в таблицах 2 – 4.

Таблица 2 - Тест-кейс #1

|  |  |
| --- | --- |
| Элементы тест-кейса | Значения |
| Название проекта | Информационная система «фитнес клуб ZeVs» |
| Номер версии | V 1.0.0.1 |
| Имя тестера | Скорочкин Никита Алексеевич |
| Даты тестирования | 10.10 -11.11 |
| Test Case # | TC\_UI\_1 |
| Приоритет теста | Высокий |
| Название теста | Проверка добавления клиента с неуказанным именем |
| Резюме испытания | 1. Проверка добавления клиента с указанием имени; 2. Проверка невозможности добавить клиента с не указанием имени; |
| Шаги тестирования | 1. Заполнить полностью форму клиента. 2. Заполнить форму клиента, но не заполнять имя; |
| Данные тестирования | 1. Взять за основу тестовые данные 2. Взять за основу тестовые данные |
| Ожидаемый результат | 1. При нажатии кнопки добавить: Успешно сохранено, переброс на страницу клиентов; 2. При нажатии кнопки добавить: Безуспешно, вылезло вспомогательное окно о незаполненности поля имени; |
| Фактический результат | 1. При нажатии кнопки добавить: Успешно сохранено, переброс на страницу клиентов; 2. При нажатии кнопки добавить: Безуспешно, вылезло вспомогательное окно о незаполненности поля имени; |
| Предпосылки | Запуск приложения и переход на форму добавления |
| Постусловия | Подключение к базе данных fitnes\_club |
| Статус | - |
| Комментарии | - |

Таблица 3 - Тест-кейс #2

|  |  |
| --- | --- |
| Элементы тест-кейса | Значения |
| Название проекта | Информационная система «фитнес клуб ZeVs» |
| Номер версии | V 1.0.0.1 |
| Имя тестера | Скорочкин Никита Алексеевич |
| Даты тестирования | 10.10 -11.11 |
| Test Case # | TC\_UI\_2 |
| Приоритет теста | Высокий |
| Название теста | Проверка добавления тренера с неуказанием специальности |
| Резюме испытания | 1. Проверка добавления тренера с указанием специальности; 2. Проверка невозможности добавить тренера с неуказанием специалзации; |
| Шаги тестирования | 1. Заполнить полностью форму тренера. 2. Заполнить форму тренера, но не заполнять специализацию; |
| Данные тестирования | 1. Взять за основу тестовые данные 2. Взять за основу тестовые данные |
| Ожидаемый результат | 1. При нажатии кнопки добавить: Успешно сохранено, переброс на страницу трененров; 2. При нажатии кнопки добавить: Безуспешно, вылезло вспомогательное окно о незаполненности поля специальности; |
| Фактический результат | 1. При нажатии кнопки добавить: Успешно сохранено, переброс на страницу тренеров; 2. При нажатии кнопки добавить: Безуспешно, вылезло вспомогательное окно о незаполненности поля специальности; |
| Предпосылки | Запуск приложения и переход на форму добавления |
| Постусловия | Подключение к базе данных fitnes\_club |
| Статус | - |
| Комментарии | - |

Таблица 4 - Тест-кейс #3

|  |  |
| --- | --- |
| Элементы тест-кейса | Значения |
| Название проекта | Информационная система «фитнес клуб ZeVs» |
| Номер версии | V 1.0.0.1 |
| Имя тестера | Скорочкин Никита Алексеевич |
| Даты тестирования | 10.10 -11.11 |
| Test Case # | TC\_UI\_3 |
| Приоритет теста | Высокий |
| Название теста | Проверка добавления бронирования с неуказанием даты |
| Резюме испытания | 1. Проверка добавления бронирования с указанием даты; 2. Проверка невозможности добавить бронь с неуказанием даты; |
| Шаги тестирования | 1. Заполнить полностью форму бронирования. 2. Заполнить форму тренера, но не заполнять дату; |
| Данные тестирования | 1. Взять за основу тестовые данные 2. Взять за основу тестовые данные |
| Ожидаемый результат | 1. При нажатии кнопки добавить: Успешно сохранено, переброс на страницу бронирования; 2. При нажатии кнопки добавить: Безуспешно, вылезло вспомогательное окно о незаполненности поля даты; |
| Фактический результат | 1. При нажатии кнопки добавить: Успешно сохранено, переброс на страницу бронрования; 2. При нажатии кнопки добавить: Безуспешно, вылезло вспомогательное окно о незаполненности поля даты; |
| Предпосылки | Запуск приложения и переход на форму добавления |
| Постусловия | Подключение к базе данных fitnes\_club |
| Статус | - |
| Комментарии | - |

ВЫВОДЫ И ЗАКЛЮЧЕНИЕ

В процессе создания курсового проекта, были достигнуты все выше поставленные цели:

* Создание программного обеспечения, которое позволяет эффективно управлять процессом ремонта электронной техники.
* Оптимизация рабочих процессов для сокращения времени ремонта и уменьшения количества ошибок.
* Реализация механизмов обратной связи для клиентов.

Для этого были выполнены следующие задачи:

* сформулирована цель разработки информационной системы;
* описана предметная область, для которой разрабатывается информационная система
* проведён анализ предметной области, выявлены основные бизнес-процессы и пользователи, которые будут с ней работать.
* определён круг запросов и задач, которые предполагается решать с использованием созданной информационной системы;
* написано техническое задание на разработку информационной системы;
* проведено функциональное моделирование и объектно-ориентированное проектирование информационной системы;
* проведено проектирование интерфейса информационной системы:
* разработана модель данных и реализована в выбранной СУБД
* разработано программное решение для информационной системы;
* оценено качество разработанной системы путем тестирования основного функционала.

СПИСОК ЛИТЕРАТУРЫ

1) Абрамян А. В. Разработка пользовательского интерфейса на основе технологии Windows Presentation Foundation : учебник для студ. Учреждений сред. Проф. Образования. - М: Издательство Южного федерального университета, 2019.

2) Ганенко А. П. Оформление текстовых и графических материалов при подготовке дипломных проектов, курсовых и письменных экзаменационных работ (требования ЕСКД) : учебно-метод. Пособие для студ. Учреждения сред. Проф. Образования – М: Издательский центр «Академия», 2020.

Электронные ресурсы:

3) Metanit – язык программирования С# и платформа .NET [Электронный ресурс] Режим доступа: свободный <https://metanit.com/sharp/> (Дата обращения 03.11.2024)

4) Национальный открытый университет [Электронный ресурс] Режим доступа: свободный https://www.intuit.ru/ (Дата обращения 03.11.2024)

5) Professor Web [Электронный ресурс] Режим доступа: свободный https://professorweb.ru/ (Дата обращения 03.11.2024)

Приложение A – Код программных модулей

BookingStatusConverter.cs:

using System;

using System.Globalization;

using System.Windows.Data;

using courseWork.Models;

namespace courseWork.Converters

{

public class BookingStatusConverter : IValueConverter

{

public object Convert(object value, Type targetType, object parameter, CultureInfo culture)

{

if (value is int status)

{

switch (status)

{

case 1:

return "Забронировано";

case 2:

return "Завершено";

case 3:

return "Отменено";

default:

return "Неизвестно";

}

}

return "Неизвестно";

}

public object ConvertBack(object value, Type targetType, object parameter, CultureInfo culture)

{

throw new NotImplementedException();

}

}

}

BookingStatus.cs:

namespace courseWork.Models

{

public enum BookingStatus

{

Booked = 1,

Ended = 2,

Canceled = 3,

}

}

Bookings.cs:

//------------------------------------------------------------------------------

// <auto-generated>

// Этот код создан по шаблону.

//

// Изменения, вносимые в этот файл вручную, могут привести к непредвиденной работе приложения.

// Изменения, вносимые в этот файл вручную, будут перезаписаны при повторном создании кода.

// </auto-generated>

//------------------------------------------------------------------------------

namespace courseWork.Models

{

using System;

using System.Collections.Generic;

public partial class Bookings

{

public int BookingID { get; set; }

public int ClientID { get; set; }

public int ScheduleID { get; set; }

public Nullable<System.DateTime> BookingDate { get; set; }

public Nullable<int> Status { get; set; }

public virtual Clients Clients { get; set; }

public virtual Schedule Schedule { get; set; }

}

}

Classes.cs:

//------------------------------------------------------------------------------

// <auto-generated>

// Этот код создан по шаблону.

//

// Изменения, вносимые в этот файл вручную, могут привести к непредвиденной работе приложения.

// Изменения, вносимые в этот файл вручную, будут перезаписаны при повторном создании кода.

// </auto-generated>

//------------------------------------------------------------------------------

namespace courseWork.Models

{

using System;

using System.Collections.Generic;

public partial class Classes

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public Classes()

{

this.Schedule = new HashSet<Schedule>();

}

public int ClassID { get; set; }

public string ClassName { get; set; }

public string Description { get; set; }

public Nullable<int> Duration { get; set; }

public int TrainerID { get; set; }

public virtual Trainers Trainers { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<Schedule> Schedule { get; set; }

}

}

ClientMemberShips.cs:

//------------------------------------------------------------------------------

// <auto-generated>

// Этот код создан по шаблону.

//

// Изменения, вносимые в этот файл вручную, могут привести к непредвиденной работе приложения.

// Изменения, вносимые в этот файл вручную, будут перезаписаны при повторном создании кода.

// </auto-generated>

//------------------------------------------------------------------------------

namespace courseWork.Models

{

using System;

using System.Collections.Generic;

public partial class ClientMemberships

{

public int ClientMembershipID { get; set; }

public int ClientID { get; set; }

public int MembershipTypeID { get; set; }

public Nullable<System.DateTime> StartDate { get; set; }

public Nullable<System.DateTime> EndDate { get; set; }

public virtual Clients Clients { get; set; }

public virtual MembershipTypes MembershipTypes { get; set; }

}

}

Clients.cs:

//------------------------------------------------------------------------------

// <auto-generated>

// Этот код создан по шаблону.

//

// Изменения, вносимые в этот файл вручную, могут привести к непредвиденной работе приложения.

// Изменения, вносимые в этот файл вручную, будут перезаписаны при повторном создании кода.

// </auto-generated>

//------------------------------------------------------------------------------

namespace courseWork.Models

{

using System;

using System.Collections.Generic;

public partial class Clients

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public Clients()

{

this.Bookings = new HashSet<Bookings>();

this.ClientMemberships = new HashSet<ClientMemberships>();

}

public int ClientID { get; set; }

public string FirstName { get; set; }

public string LastName { get; set; }

public string PhoneNumber { get; set; }

public string Email { get; set; }

public Nullable<System.DateTime> MembershipStartDate { get; set; }

public Nullable<System.DateTime> MembershipEndDate { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<Bookings> Bookings { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<ClientMemberships> ClientMemberships { get; set; }

}

}

MemberShipsTypes.cs:

//------------------------------------------------------------------------------

// <auto-generated>

// Этот код создан по шаблону.

//

// Изменения, вносимые в этот файл вручную, могут привести к непредвиденной работе приложения.

// Изменения, вносимые в этот файл вручную, будут перезаписаны при повторном создании кода.

// </auto-generated>

//------------------------------------------------------------------------------

namespace courseWork.Models

{

using System;

using System.Collections.Generic;

public partial class Clients

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public Clients()

{

this.Bookings = new HashSet<Bookings>();

this.ClientMemberships = new HashSet<ClientMemberships>();

}

public int ClientID { get; set; }

public string FirstName { get; set; }

public string LastName { get; set; }

public string PhoneNumber { get; set; }

public string Email { get; set; }

public Nullable<System.DateTime> MembershipStartDate { get; set; }

public Nullable<System.DateTime> MembershipEndDate { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<Bookings> Bookings { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<ClientMemberships> ClientMemberships { get; set; }

}

}

Shchedule.cs:

//------------------------------------------------------------------------------

// <auto-generated>

// Этот код создан по шаблону.

//

// Изменения, вносимые в этот файл вручную, могут привести к непредвиденной работе приложения.

// Изменения, вносимые в этот файл вручную, будут перезаписаны при повторном создании кода.

// </auto-generated>

//------------------------------------------------------------------------------

namespace courseWork.Models

{

using System;

using System.Collections.Generic;

public partial class Schedule

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public Schedule()

{

this.Bookings = new HashSet<Bookings>();

}

public int ScheduleID { get; set; }

public int ClassID { get; set; }

public Nullable<System.DateTime> Date { get; set; }

public Nullable<System.TimeSpan> StartTime { get; set; }

public Nullable<System.TimeSpan> EndTime { get; set; }

public Nullable<int> MaxCapacity { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<Bookings> Bookings { get; set; }

public virtual Classes Classes { get; set; }

}

}

Trainers.cs:

//------------------------------------------------------------------------------

// <auto-generated>

// Этот код создан по шаблону.

//

// Изменения, вносимые в этот файл вручную, могут привести к непредвиденной работе приложения.

// Изменения, вносимые в этот файл вручную, будут перезаписаны при повторном создании кода.

// </auto-generated>

//------------------------------------------------------------------------------

namespace courseWork.Models

{

using System;

using System.Collections.Generic;

public partial class Trainers

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public Trainers()

{

this.Classes = new HashSet<Classes>();

}

public int TrainerID { get; set; }

public string FirstName { get; set; }

public string LastName { get; set; }

public string PhoneNumber { get; set; }

public string Email { get; set; }

public string Specialization { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<Classes> Classes { get; set; }

}

}

BookingService:

using courseWork.Models;

using System.Collections.Generic;

using System.Linq;

namespace courseWork.Services

{

public class BookingsService

{

private readonly FitnesClubEntities \_context;

public BookingsService()

{

\_context = new FitnesClubEntities();

}

public List<Bookings> GetBookings(int page, int count)

{

return \_context.Bookings

.OrderBy(c => c.BookingID)

.Skip((page - 1) \* count)

.Take(count)

.ToList();

}

public void SaveBooking(Bookings bookings)

{

\_context.Bookings.Add(bookings);

\_context.SaveChanges();

}

}

}

ClientsService.cs:

using courseWork.Models;

using System.Collections.Generic;

using System.Linq;

namespace courseWork.Services

{

public class ClientsService

{

private readonly FitnesClubEntities \_context;

public ClientsService()

{

\_context = new FitnesClubEntities();

}

public List<Clients> GetClients(int page, int count)

{

return \_context.Clients

.OrderBy(c => c.ClientID)

.Skip((page - 1) \* count)

.Take(count)

.ToList();

}

public Clients GetClientByPhoneNumber(string phoneNumber)

{

return \_context.Clients.Where(c => c.PhoneNumber == phoneNumber).FirstOrDefault();

}

public void AddClient(Clients client)

{

\_context.Clients.Add(client);

\_context.SaveChanges();

}

}

}

SchedulesService.cs:

using courseWork.Models;

using System.Collections.ObjectModel;

using System.Linq;

using System.Reactive.Linq;

using System.Windows.Documents;

namespace courseWork.Services

{

public class SchedulesService

{

private readonly FitnesClubEntities \_context;

public SchedulesService()

{

\_context = new FitnesClubEntities();

}

public Schedule[] GetAllSchedules()

{

return \_context.Schedule.ToArray();

}

}

}

TrainersService.cs:

using courseWork.Models;

using System.Collections.Generic;

using System.Linq;

namespace courseWork.Services

{

public class TrainersService

{

private readonly FitnesClubEntities \_context;

public TrainersService()

{

\_context = new FitnesClubEntities();

}

public List<Trainers> GetTrainers(int page, int count)

{

return \_context.Trainers

.OrderBy(c => c.TrainerID)

.Skip((page - 1) \* count)

.Take(count)

.ToList();

}

public void AddTrainer(Trainers trainer)

{

\_context.Trainers.Add(trainer);

\_context.SaveChanges();

}

}

}

ButtonStyleTab.xaml:

<ResourceDictionary xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">

<Style x:Key="CustomButtonStyleTab" TargetType="Button">

<Setter Property="BorderBrush" Value="Transparent"/>

<Setter Property="Background" Value="Transparent"/>

<Setter Property="Template">

<Setter.Value>

<ControlTemplate TargetType="Button">

<Border Background="{TemplateBinding Background}"

BorderBrush="{TemplateBinding BorderBrush}"

BorderThickness="{TemplateBinding BorderThickness}">

<ContentPresenter HorizontalAlignment="{TemplateBinding HorizontalContentAlignment}"

VerticalAlignment="{TemplateBinding VerticalContentAlignment}"/>

</Border>

<ControlTemplate.Triggers>

<Trigger Property="IsMouseOver" Value="True">

<Setter Property="Background" Value="#f6faaf"/>

<Setter Property="BorderBrush" Value="Transparent"/>

</Trigger>

</ControlTemplate.Triggers>

</ControlTemplate>

</Setter.Value>

</Setter>

</Style>

</ResourceDictionary>

ViewModelBase.cs:

using ReactiveUI;

using ReactiveUI.Fody.Helpers;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

namespace courseWork.Tools

{

public class ViewModelBase : ReactiveObject

{

public double Width => SystemParameters.PrimaryScreenWidth \* 0.8;

public double Height => SystemParameters.PrimaryScreenHeight \* 0.8;

}

}

AddBookingVM.cs:

using courseWork.Models;

using courseWork.Services;

using courseWork.Tools;

using ReactiveUI;

using ReactiveUI.Fody.Helpers;

using System;

using System.Collections.ObjectModel;

using System.Linq;

using System.Reactive;

using System.Windows;

namespace courseWork.ViewModels

{

public class AddBookingVM : ViewModelBase

{

private readonly SchedulesService \_schedulesService;

private readonly ClientsService \_clientsService;

private readonly BookingsService \_bookingsService;

public Schedule[] Schedules { get; set; }

public ObservableCollection<DateTime> ScheduleDates { get; set; }

public ReactiveCommand<DateTime, Unit> ScheduleSelectedCommand { get; }

public ReactiveCommand<DateTime, Unit> OnDateSelectedCommand { get; }

public ReactiveCommand<Unit, Unit> AddBooking { get; set; }

private DateTime \_selectedDate;

public DateTime SelectedDate

{

get => \_selectedDate;

set => this.RaiseAndSetIfChanged(ref \_selectedDate, value);

}

[Reactive] public DateTime BookingDateTime { get; set; } = DateTime.Now;

[Reactive] public string PhoneNumber { get; set; } = string.Empty;

public AddBookingVM()

{

\_schedulesService = new SchedulesService();

\_clientsService = new ClientsService();

\_bookingsService = new BookingsService();

Schedules = \_schedulesService.GetAllSchedules();

ScheduleDates = new ObservableCollection<DateTime>(Schedules.Select(s => s.Date.Value));

ScheduleSelectedCommand = ReactiveCommand.Create<DateTime>(OnScheduleSelected);

OnDateSelectedCommand = ReactiveCommand.Create<DateTime>(OnDateSelected);

AddBooking = ReactiveCommand.Create(() =>

{

AddBookingToDB();

});

}

private void OnScheduleSelected(DateTime date)

{

SelectedDate = date;

}

private void OnDateSelected(DateTime date)

{

BookingDateTime = date;

}

private void AddBookingToDB()

{

var client = \_clientsService.GetClientByPhoneNumber(PhoneNumber);

if (client is null)

{

MessageBox.Show("Пользователя с таким номером нет");

return;

}

var scheduleID = Schedules.Where(s => s.Date == SelectedDate).Select(s => s.ScheduleID).FirstOrDefault();

if(scheduleID == 0)

{

MessageBox.Show("Выберите расписание");

return;

}

var booking = new Bookings()

{

ClientID = client.ClientID,

BookingDate = BookingDateTime,

ScheduleID = scheduleID,

Status = 1

};

\_bookingsService.SaveBooking(booking);

MessageBox.Show("Успешно добавлено");

}

}

}

AddClientVM.cs:

using courseWork.Models;

using courseWork.Services;

using courseWork.Tools;

using ReactiveUI;

using ReactiveUI.Fody.Helpers;

using System;

using System.Data.Entity.Validation;

using System.Reactive;

using System.Windows;

namespace courseWork.ViewModels

{

public class AddClientVM : ViewModelBase

{

private readonly ClientsService \_clientsService;

[Reactive] public string FirstName { get; set; }

[Reactive] public string LastName { get; set; }

[Reactive] public string PhoneNumber { get; set; }

[Reactive] public string Email { get; set; }

public ReactiveCommand<Unit, Unit> AddClientCommand { get; set; }

public AddClientVM()

{

\_clientsService = new ClientsService();

AddClientCommand = ReactiveCommand.Create(() =>

{

AddClient();

});

}

private void AddClient()

{

if (String.IsNullOrWhiteSpace(FirstName))

{

MessageBox.Show("Введите имя клиента");

return;

}

if (String.IsNullOrWhiteSpace(LastName))

{

MessageBox.Show("Введите фамилию клиента");

return;

}

if (String.IsNullOrWhiteSpace(Email))

{

MessageBox.Show("Введите почту клиента");

return;

}

if (String.IsNullOrWhiteSpace(PhoneNumber))

{

MessageBox.Show("Введите номер телефона клиента");

return;

}

var client = new Clients()

{

Email = Email,

FirstName = FirstName,

LastName = LastName,

PhoneNumber = PhoneNumber,

MembershipStartDate = DateTime.Now,

MembershipEndDate = DateTime.Now.AddMonths(36),

};

try

{

\_clientsService.AddClient(client);

MessageBox.Show("Успешно добавлено");

}

catch (DbEntityValidationException ex)

{

foreach (var validationErrors in ex.EntityValidationErrors)

{

foreach (var validationError in validationErrors.ValidationErrors)

{

MessageBox.Show($"Property: {validationError.PropertyName} Error: {validationError.ErrorMessage}");

}

}

}

}

}

}

AddTrainers.cs:

using courseWork.Models;

using courseWork.Services;

using courseWork.Tools;

using ReactiveUI.Fody.Helpers;

using ReactiveUI;

using System.Windows;

using System;

using System.Reactive;

namespace courseWork.ViewModels

{

public class AddTrainerVM : ViewModelBase

{

private readonly TrainersService \_trainersService;

[Reactive] public string FirstName { get; set; }

[Reactive] public string LastName { get; set; }

[Reactive] public string PhoneNumber { get; set; }

[Reactive] public string Email { get; set; }

[Reactive] public string Specialization { get; set; }

public ReactiveCommand<Unit, Unit> AddClientCommand { get; set; }

public AddTrainerVM()

{

\_trainersService = new TrainersService();

AddClientCommand = ReactiveCommand.Create(() =>

{

AddClient();

});

}

private void AddClient()

{

if (String.IsNullOrWhiteSpace(FirstName))

{

MessageBox.Show("Введите имя тренера");

return;

}

if (String.IsNullOrWhiteSpace(LastName))

{

MessageBox.Show("Введите фамилию тренера");

return;

}

if (String.IsNullOrWhiteSpace(Email))

{

MessageBox.Show("Введите почту тренера");

return;

}

if (String.IsNullOrWhiteSpace(PhoneNumber))

{

MessageBox.Show("Введите номер телефона тренера");

return;

}

if (String.IsNullOrWhiteSpace(Specialization))

{

MessageBox.Show("Введите специализацию тренера");

return;

}

var trainer = new Trainers()

{

Email = Email,

FirstName = FirstName,

LastName = LastName,

PhoneNumber = PhoneNumber,

Specialization = Specialization,

};

\_trainersService.AddTrainer(trainer);

MessageBox.Show("Успешно добавлено");

}

}

}

BookingsPageVM.cs:

using courseWork.Models;

using courseWork.Services;

using courseWork.Tools;

using courseWork.Views;

using ReactiveUI;

using ReactiveUI.Fody.Helpers;

using System.Collections.ObjectModel;

using System.Linq;

using System.Reactive;

using System.Reactive.Linq;

namespace courseWork.ViewModels

{

public class BookingsPageVM : ViewModelBase

{

private readonly BookingsService \_bookingsService;

[Reactive]

public bool IsLastPage { get; set; } = true;

[Reactive]

public bool IsFirstPage { get; set; } = false;

[Reactive] public int Page { get; set; } = 1;

[Reactive] public int Count { get; set; } = 20;

[Reactive] public ObservableCollection<Bookings> Bookings { get; set; }

private ReactiveCommand<Unit, Unit> \_toNextPage;

private ReactiveCommand<Unit, Unit> \_toPreviousPage;

private ReactiveCommand<Unit, Unit> \_addBooking;

public ReactiveCommand<Unit, Unit> ToPreviousPage => \_toPreviousPage;

public ReactiveCommand<Unit, Unit> ToNextPage => \_toNextPage;

public ReactiveCommand<Unit, Unit> AddBooking => \_addBooking;

public BookingsPageVM()

{

\_bookingsService = new BookingsService();

Bookings = new ObservableCollection<Bookings>();

LoadTrainers();

\_toNextPage = ReactiveCommand.Create(() =>

{

Page++;

if (Page > 1) IsFirstPage = true;

else IsFirstPage = false;

});

\_toPreviousPage = ReactiveCommand.Create(() =>

{

Page--;

if (Page == 1) IsFirstPage = false;

});

\_addBooking = ReactiveCommand.Create(() =>

{

AddBooking addBooking = new AddBooking();

addBooking.Show();

});

this.WhenAnyValue(x => x.Page, x => x.Count)

.Select(\_ => Unit.Default)

.InvokeCommand(ReactiveCommand.Create(LoadTrainers));

}

private void LoadTrainers()

{

var bookings = \_bookingsService.GetBookings(Page, Count);

Bookings = new ObservableCollection<Bookings>(bookings);

if (bookings.Count < Count) IsLastPage = false;

else IsLastPage = true;

}

}

}

ClientDetails.cs:

using courseWork.Models;

using courseWork.Tools;

namespace courseWork.ViewModels

{

public class ClientDetailsVM : ViewModelBase

{

public Clients Client { get; set; }

public string ImagePath { get; set; } = "\\Assets\\picture.png";

public ClientDetailsVM(Clients client)

{

Client = client;

}

}

}

ClientsPage.cs:

using courseWork.Models;

using courseWork.Services;

using courseWork.Tools;

using courseWork.Views;

using ReactiveUI;

using ReactiveUI.Fody.Helpers;

using System.Collections.ObjectModel;

using System.Reactive;

using System.Reactive.Linq;

namespace courseWork.ViewModels

{

public class ClientsPageVM : ViewModelBase

{

private readonly ClientsService \_clientsService;

[Reactive]

public bool IsLastPage { get; set; } = true;

[Reactive]

public bool IsFirstPage { get; set; } = false;

[Reactive] public int Page { get; set; } = 1;

[Reactive] public int Count { get; set; } = 20;

[Reactive] public ObservableCollection<Clients> Clients { get; set; }

private ReactiveCommand<Unit, Unit> \_toNextPage;

private ReactiveCommand<Unit, Unit> \_toPreviousPage;

private ReactiveCommand<Unit, Unit> \_toAddClient;

public ReactiveCommand<Unit, Unit> ToPreviousPage => \_toPreviousPage;

public ReactiveCommand<Unit, Unit> ToNextPage => \_toNextPage;

public ReactiveCommand<Unit, Unit> ToAddClient => \_toAddClient;

public ClientsPageVM()

{

\_clientsService = new ClientsService();

Clients = new ObservableCollection<Clients>();

LoadClients();

\_toNextPage = ReactiveCommand.Create(() =>

{

Page++;

if (Page > 1) IsFirstPage = true;

else IsFirstPage = false;

});

\_toPreviousPage = ReactiveCommand.Create(() =>

{

Page--;

if (Page == 1) IsFirstPage = false;

});

\_toAddClient = ReactiveCommand.Create(() =>

{

AddClient addClient = new AddClient();

addClient.Show();

});

this.WhenAnyValue(x => x.Page, x => x.Count)

.Select(\_ => Unit.Default)

.InvokeCommand(ReactiveCommand.Create(LoadClients));

}

private void LoadClients()

{

var clients = \_clientsService.GetClients(Page, Count);

if(clients.Count < Count) IsLastPage = false;

else IsLastPage = true;

Clients.Clear();

foreach (var client in clients)

{

Clients.Add(client);

}

}

}

}

EmptyPageVM.cs:

using courseWork.Tools;

using ReactiveUI;

namespace courseWork.ViewModels

{

public class EmptyPageVM : ViewModelBase

{

private string \_initText = "Добро пожаловать в систему!";

public string InitText

{

get => \_initText;

set => this.RaiseAndSetIfChanged(ref \_initText, value);

}

public EmptyPageVM()

{

}

}

}

MainWindowVM.cs:

using courseWork.Models;

using courseWork.Tools;

using courseWork.Views;

using ReactiveUI;

using ReactiveUI.Fody.Helpers;

using System.Reactive;

using System.Windows.Controls;

namespace courseWork.ViewModels

{

public class MainWindowVM : ViewModelBase

{

[Reactive] public UserControl CurrentPage { get; set; }

private ReactiveCommand<Unit, Unit> \_navigateToClientsPageCommand;

private ReactiveCommand<Unit, Unit> \_navigateToEmptyPage;

private ReactiveCommand<Unit, Unit> \_navigateToTrainersPage;

private ReactiveCommand<Unit, Unit> \_navigateToBookingPage;

public ReactiveCommand<Unit, Unit> NavigateToEmptyPage => \_navigateToEmptyPage;

public ReactiveCommand<Unit, Unit> NavigateToClientsPageCommand => \_navigateToClientsPageCommand;

public ReactiveCommand<Unit, Unit> NavigateToTrainersPage => \_navigateToTrainersPage;

public ReactiveCommand<Unit, Unit> NavigateToBookingPage => \_navigateToBookingPage;

public MainWindowVM()

{

\_navigateToClientsPageCommand = ReactiveCommand.Create(() =>

{

if (CurrentPage is ClientsPage) return;

CurrentPage = new ClientsPage();

});

\_navigateToEmptyPage = ReactiveCommand.Create(() =>

{

if (CurrentPage is EmptyPage) return;

CurrentPage = new EmptyPage();

});

\_navigateToTrainersPage = ReactiveCommand.Create(() =>

{

if(CurrentPage is TrainersPage) return;

CurrentPage = new TrainersPage();

});

\_navigateToBookingPage = ReactiveCommand.Create(() =>

{

if(CurrentPage is BookingsPage) return;

CurrentPage = new BookingsPage();

});

CurrentPage = new EmptyPage();

}

}

}

TrainerDetailsVM.cs:

using courseWork.Models;

using courseWork.Tools;

namespace courseWork.ViewModels

{

public class TrainerDetailsVM : ViewModelBase

{

public Trainers Trainer { get; set; }

public TrainerDetailsVM(Trainers trainer)

{

Trainer = trainer;

}

}

}

TrainersPageVM.cs:

using courseWork.Models;

using courseWork.Services;

using courseWork.Tools;

using ReactiveUI.Fody.Helpers;

using ReactiveUI;

using System.Collections.ObjectModel;

using System.Reactive;

using System.Reactive.Linq;

using courseWork.Views;

namespace courseWork.ViewModels

{

public class TrainersPageVM : ViewModelBase

{

private readonly TrainersService \_trainersService;

[Reactive]

public bool IsLastPage { get; set; } = true;

[Reactive]

public bool IsFirstPage { get; set; } = false;

[Reactive] public int Page { get; set; } = 1;

[Reactive] public int Count { get; set; } = 20;

[Reactive] public ObservableCollection<Trainers> Trainers { get; set; }

private ReactiveCommand<Unit, Unit> \_toNextPage;

private ReactiveCommand<Unit, Unit> \_toPreviousPage;

private ReactiveCommand<Unit, Unit> \_toAddTrainer;

public ReactiveCommand<Unit, Unit> ToPreviousPage => \_toPreviousPage;

public ReactiveCommand<Unit, Unit> ToNextPage => \_toNextPage;

public ReactiveCommand<Unit, Unit> ToAddTrainer => \_toAddTrainer;

public TrainersPageVM()

{

\_trainersService = new TrainersService();

Trainers = new ObservableCollection<Trainers>();

LoadTrainers();

\_toNextPage = ReactiveCommand.Create(() =>

{

Page++;

if (Page > 1) IsFirstPage = true;

else IsFirstPage = false;

});

\_toPreviousPage = ReactiveCommand.Create(() =>

{

Page--;

if (Page == 1) IsFirstPage = false;

});

\_toAddTrainer = ReactiveCommand.Create(() =>

{

AddTrainer addTrainer = new AddTrainer();

addTrainer.Show();

});

this.WhenAnyValue(x => x.Page, x => x.Count)

.Select(\_ => Unit.Default)

.InvokeCommand(ReactiveCommand.Create(LoadTrainers));

}

private void LoadTrainers()

{

var clients = \_trainersService.GetTrainers(Page, Count);

if (clients.Count < Count) IsLastPage = false;

else IsLastPage = true;

Trainers.Clear();

foreach (var client in clients)

{

Trainers.Add(client);

}

}

}

}

<Window x:Class="courseWork.Views.AddBooking"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:courseWork.Views"

xmlns:i="http://schemas.microsoft.com/expression/2010/interactivity"

xmlns:rxui="http://reactiveui.net"

mc:Ignorable="d"

Title="AddBooking"

AllowsTransparency="True"

xmlns:vm="clr-namespace:courseWork.ViewModels"

xmlns:iconPacks="http://metro.mahapps.com/winfx/xaml/iconpacks"

WindowStyle="None"

WindowStartupLocation="CenterScreen"

ResizeMode="CanMinimize"

Height="280"

Width="400"

Background="Transparent">

<Border Background="Beige" BorderBrush="Black" BorderThickness="3">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="40"/>

<RowDefinition/>

<RowDefinition Height="40"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0">

<TextBlock

Text="Добавление бронирования"

VerticalAlignment="Center"

HorizontalAlignment="Center"

FontSize="28"/>

</Grid>

<Grid Grid.Row="1">

<StackPanel Orientation="Vertical">

<TextBlock Text="Расписание"

VerticalAlignment="Center"

HorizontalAlignment="Center"/>

<ComboBox Padding="10"

Width="200"

Margin="0,10,0,0"

ItemsSource="{Binding ScheduleDates}"

SelectedItem="{Binding SelectedDate, Mode=TwoWay}"

DisplayMemberPath="Date">

<i:Interaction.Triggers>

<i:EventTrigger EventName="SelectionChanged">

<i:InvokeCommandAction Command="{Binding ScheduleSelectedCommand}" CommandParameter="{Binding SelectedDate}"/>

</i:EventTrigger>

</i:Interaction.Triggers>

</ComboBox>

<TextBlock Text="Дата"

VerticalAlignment="Center"

HorizontalAlignment="Center"/>

<DatePicker Padding="10"

IsTodayHighlighted="True"

Width="200"

SelectedDate="{Binding BookingDateTime, Mode=TwoWay}">

<i:Interaction.Triggers>

<i:EventTrigger EventName="SelectedDateChanged">

<i:InvokeCommandAction Command="{Binding OnDateSelectedCommand}" CommandParameter="{Binding BookingDateTime}"/>

</i:EventTrigger>

</i:Interaction.Triggers>

</DatePicker>

<TextBlock Text="Номер телефона клента"

VerticalAlignment="Center"

HorizontalAlignment="Center"/>

<TextBox Width="200"

Padding="10"

FontSize="20"

Text="{Binding PhoneNumber, Mode=TwoWay, UpdateSourceTrigger=PropertyChanged}">

</TextBox>

</StackPanel>

</Grid>

<Grid Grid.Row="2">

<StackPanel VerticalAlignment="Center"

Orientation="Horizontal"

HorizontalAlignment="Center">

<Button Content="Добавить"

Command="{Binding AddBooking}"

Width="100"

Height="40"

/>

<Button Margin="10,0,0,0" Content="Закрыть"

Click="Button\_Click"

Width="100"

Height="40"

/>

</StackPanel>

</Grid>

</Grid>

</Border>

</Window>

<Window x:Class="courseWork.Views.AddClient"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:courseWork.Views"

xmlns:i="http://schemas.microsoft.com/expression/2010/interactivity"

xmlns:rxui="http://reactiveui.net"

mc:Ignorable="d"

Title="AddClient"

AllowsTransparency="True"

xmlns:vm="clr-namespace:courseWork.ViewModels"

xmlns:iconPacks="http://metro.mahapps.com/winfx/xaml/iconpacks"

WindowStyle="None"

WindowStartupLocation="CenterScreen"

ResizeMode="CanMinimize"

Height="400"

Width="400"

Background="Transparent">

<Border Background="Beige" BorderBrush="Black" BorderThickness="3">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="40"/>

<RowDefinition/>

<RowDefinition Height="40"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0">

<TextBlock

Text="Добавление клиента"

VerticalAlignment="Center"

HorizontalAlignment="Center"

FontSize="28"/>

</Grid>

<Grid Grid.Row="1">

<StackPanel Orientation="Vertical">

<TextBlock Text="Имя"

VerticalAlignment="Center"

HorizontalAlignment="Center"/>

<TextBox Padding="10"

Width="200"

Margin="0,10,0,0"

FontSize="20"

Text="{Binding FirstName, Mode=TwoWay, UpdateSourceTrigger=PropertyChanged}"/>

<TextBlock Text="Фамилия"

VerticalAlignment="Center"

HorizontalAlignment="Center"/>

<TextBox Padding="10"

Width="200"

Margin="0,10,0,0"

FontSize="20"

Text="{Binding LastName, Mode=TwoWay, UpdateSourceTrigger=PropertyChanged}"/>

<TextBlock Text="Почта"

VerticalAlignment="Center"

HorizontalAlignment="Center"/>

<TextBox Padding="10"

Width="200"

Margin="0,10,0,0"

FontSize="20"

Text="{Binding Email, Mode=TwoWay, UpdateSourceTrigger=PropertyChanged}"/>

<TextBlock Text="Номер телефона"

VerticalAlignment="Center"

HorizontalAlignment="Center"/>

<TextBox Padding="10"

Width="200"

Margin="0,10,0,0"

FontSize="20"

Text="{Binding PhoneNumber, Mode=TwoWay, UpdateSourceTrigger=PropertyChanged}"/>

</StackPanel>

</Grid>

<Grid Grid.Row="2">

<StackPanel VerticalAlignment="Center"

Orientation="Horizontal"

HorizontalAlignment="Center">

<Button Content="Добавить"

Command="{Binding AddClientCommand}"

Width="100"

Height="40"/>

<Button Margin="10,0,0,0" Content="Закрыть"

Click="Button\_Click"

Width="100"

Height="40"/>

</StackPanel>

</Grid>

</Grid>

</Border>

</Window>

<Window x:Class="courseWork.Views.AddTrainer"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:courseWork.Views"

mc:Ignorable="d"

Title="AddTrainer" AllowsTransparency="True"

xmlns:vm="clr-namespace:courseWork.ViewModels"

xmlns:iconPacks="http://metro.mahapps.com/winfx/xaml/iconpacks"

WindowStyle="None"

WindowStartupLocation="CenterScreen"

ResizeMode="CanMinimize"

Height="480"

Width="400"

Background="Transparent">

<Border Background="Beige" BorderBrush="Black" BorderThickness="3">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="40"/>

<RowDefinition/>

<RowDefinition Height="40"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0">

<TextBlock

Text="Добавление тренера"

VerticalAlignment="Center"

HorizontalAlignment="Center"

FontSize="28"/>

</Grid>

<Grid Grid.Row="1">

<StackPanel Orientation="Vertical">

<TextBlock Text="Имя"

VerticalAlignment="Bottom"

HorizontalAlignment="Center"/>

<TextBox Padding="10"

Width="200"

Margin="0,10,0,0"

FontSize="20"

Text="{Binding FirstName, Mode=TwoWay, UpdateSourceTrigger=PropertyChanged}"/>

<TextBlock Text="Фамилия"

VerticalAlignment="Center"

HorizontalAlignment="Center"/>

<TextBox Padding="10"

Width="200"

Margin="0,10,0,0"

FontSize="20"

Text="{Binding LastName, Mode=TwoWay, UpdateSourceTrigger=PropertyChanged}"/>

<TextBlock Text="Почта"

VerticalAlignment="Center"

HorizontalAlignment="Center"/>

<TextBox Padding="10"

Width="200"

Margin="0,10,0,0"

FontSize="20"

Text="{Binding Email, Mode=TwoWay, UpdateSourceTrigger=PropertyChanged}"/>

<TextBlock Text="Номер телефона"

VerticalAlignment="Center"

HorizontalAlignment="Center"/>

<TextBox Padding="10"

Width="200"

Margin="0,10,0,0"

FontSize="20"

Text="{Binding PhoneNumber, Mode=TwoWay, UpdateSourceTrigger=PropertyChanged}"/>

<TextBlock Text="Специализация"

VerticalAlignment="Center"

HorizontalAlignment="Center"/>

<TextBox Padding="10"

Width="200"

Margin="0,10,0,0"

FontSize="20"

Text="{Binding Specialization, Mode=TwoWay, UpdateSourceTrigger=PropertyChanged}"/>

</StackPanel>

</Grid>

<Grid Grid.Row="2">

<StackPanel VerticalAlignment="Center"

Orientation="Horizontal"

HorizontalAlignment="Center">

<Button Content="Добавить"

Command="{Binding AddClientCommand}"

Width="100"

Height="40"/>

<Button Margin="10,0,0,0" Content="Закрыть"

Click="Button\_Click"

Width="100"

Height="40"/>

</StackPanel>

</Grid>

</Grid>

</Border>

</Window>

<UserControl x:Class="courseWork.Views.BookingsPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:courseWork.Views"

mc:Ignorable="d"

d:DesignHeight="{Binding Height}"

d:DesignWidth="{Binding Width}"

xmlns:vm="clr-namespace:courseWork.ViewModels"

xmlns:iconPacks="http://metro.mahapps.com/winfx/xaml/iconpacks"

xmlns:converters="clr-namespace:courseWork.Converters">

<UserControl.Resources>

<converters:BookingStatusConverter x:Key="BookingStatusConverter"/>

</UserControl.Resources>

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0">

<ScrollViewer VerticalScrollBarVisibility="Auto">

<ItemsControl ItemsSource="{Binding Bookings}">

<ItemsControl.ItemsPanel>

<ItemsPanelTemplate>

<WrapPanel Orientation="Horizontal" ItemWidth="300" ItemHeight="180"/>

</ItemsPanelTemplate>

</ItemsControl.ItemsPanel>

<ItemsControl.ItemTemplate>

<DataTemplate>

<Border Background="White"

BorderBrush="Gray"

BorderThickness="1"

Margin="5"

Cursor="Hand"

Padding="10"

CornerRadius="5">

<StackPanel>

<TextBlock Text="{Binding BookingDate, StringFormat='Время брони: {0:d}'}" FontWeight="Bold"/>

<TextBlock Text="{Binding Status ,Converter={StaticResource BookingStatusConverter}, StringFormat='Статус: {0:d}'}"/>

</StackPanel>

</Border>

</DataTemplate>

</ItemsControl.ItemTemplate>

</ItemsControl>

</ScrollViewer>

</Grid>

<Grid Grid.Row="1">

<Border Background="White">

<Grid>

<StackPanel Orientation="Horizontal"

HorizontalAlignment="Center"

VerticalAlignment="Bottom"

Margin="0,0,0,10">

<Button Width="60"

IsEnabled="{Binding IsFirstPage}"

Command="{Binding ToPreviousPage}"

Height="60"

ToolTip="На прошлую страницу">

<iconPacks:PackIconModern Width="22"

Height="22"

VerticalAlignment="Center"

Kind="ArrowLeft" />

</Button>

<Button Margin="10,0,0,0"

Width="60"

IsEnabled="{Binding IsLastPage}"

Height="60"

Command="{Binding ToNextPage}"

ToolTip="На следующую страницу">

<iconPacks:PackIconModern Width="22"

Height="22"

VerticalAlignment="Center"

Kind="ArrowRight" />

</Button>

</StackPanel>

<TextBlock HorizontalAlignment="Right"

FontSize="30"

VerticalAlignment="Center"

Text="{Binding Page, StringFormat='Страница: {0:d}'}"/>

<Button

Content="Забронировать"

Margin="15,0,0,0"

Command="{Binding AddBooking}"

ToolTip="Забронировать"

Width="100"

Height="60"

VerticalAlignment="Center"

HorizontalAlignment="Left"

>

</Button>

</Grid>

</Border>

</Grid>

</Grid>

</UserControl>

<Window x:Class="courseWork.Views.ClientDetailsWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:courseWork.Views"

mc:Ignorable="d"

Title="Клиент"

AllowsTransparency="True"

xmlns:vm="clr-namespace:courseWork.ViewModels"

xmlns:iconPacks="http://metro.mahapps.com/winfx/xaml/iconpacks"

WindowStyle="None"

WindowStartupLocation="CenterScreen"

ResizeMode="CanMinimize"

Height="450"

Width="450"

Background="Transparent">

<Border Background="Beige"

BorderBrush="Black"

BorderThickness="3">

<Grid Background="Transparent">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="450"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="40"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0" Grid.Column="0">

<TextBlock

Text="Страница клиента"

VerticalAlignment="Center"

HorizontalAlignment="Center"

FontSize="30"

/>

</Grid>

<Grid Grid.Row="0" Grid.Column="1">

<Button Width="32"

Height="32"

ToolTip="Закрыть?"

Click="CloseButton\_Click"

VerticalAlignment="Center"

HorizontalAlignment="Right"

Margin="0,0,10,0">

<iconPacks:BootstrapIcons

Kind="DoorClosed"/>

</Button>

</Grid>

<Grid Grid.Row="1" Grid.Column="0">

<Grid.RowDefinitions>

<RowDefinition Height="120"/>

<RowDefinition/>

</Grid.RowDefinitions>

<Grid Grid.Row="0">

<Image Source="{Binding ImagePath}"/>

</Grid>

<Border BorderThickness="2"

BorderBrush="Black"

Padding="10"

CornerRadius="20"

Background="White"

Margin="0,20,0,0"

Grid.Row="1">

<StackPanel Orientation="Vertical">

<TextBlock Text="{Binding Client.FirstName, StringFormat='Имя: {0:d}'}" FontSize="20"/>

<TextBlock Text="{Binding Client.LastName, StringFormat='Фамилия: {0:d}'}" FontSize="20"/>

<TextBlock Text="{Binding Client.PhoneNumber, StringFormat='Номер телефона: +{0:d}'}" FontSize="20"/>

<TextBlock Text="{Binding Client.Email, StringFormat='Почта: {0:d}'}" FontSize="20"/>

<TextBlock Text="{Binding Client.MembershipStartDate, StringFormat='Начало членства: {0:d}'}" FontSize="20"/>

<TextBlock Text="{Binding Client.MembershipEndDate, StringFormat='Конец членства: {0:d}'}" FontSize="20"/>

<TextBlock Text="{Binding Client.ClientID, StringFormat='ID: {0:d}'}" FontSize="20"/>

</StackPanel>

</Border>

</Grid>

<Grid Grid.Row="1" Grid.Column="1">

<!--<StackPanel Orientation="Vertical">

<Button

</StackPanel>-->

</Grid>

</Grid>

</Border>

</Window>

<UserControl x:Class="courseWork.Views.ClientsPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:courseWork.Views"

xmlns:mah="clr-namespace:MahApps.Metro.Controls;assembly=MahApps.Metro"

mc:Ignorable="d"

d:DesignHeight="{Binding Height}"

d:DesignWidth="{Binding Width}"

xmlns:vm="clr-namespace:courseWork.ViewModels"

xmlns:iconPacks="http://metro.mahapps.com/winfx/xaml/iconpacks">

<UserControl.DataContext>

<vm:ClientsPageVM/>

</UserControl.DataContext>

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0">

<ScrollViewer VerticalScrollBarVisibility="Auto">

<ItemsControl ItemsSource="{Binding Clients}">

<ItemsControl.ItemsPanel>

<ItemsPanelTemplate>

<WrapPanel Orientation="Horizontal" ItemWidth="300" ItemHeight="180"/>

</ItemsPanelTemplate>

</ItemsControl.ItemsPanel>

<ItemsControl.ItemTemplate>

<DataTemplate>

<Border Background="White"

BorderBrush="Gray"

BorderThickness="1"

Margin="5"

Cursor="Hand"

Padding="10"

CornerRadius="5"

MouseLeftButtonUp="Border\_MouseLeftButtonUp">

<StackPanel>

<TextBlock Text="{Binding FirstName, StringFormat='Имя: {0:d}'}" FontWeight="Bold"/>

<TextBlock Text="{Binding LastName, StringFormat='Фамилия: {0:d}'}"/>

<TextBlock Text="{Binding PhoneNumber, StringFormat='Номер: {0:d}'}"/>

<TextBlock Text="{Binding Email, StringFormat='Эл. почта: {0:d}'}"/>

<TextBlock Text="{Binding MembershipStartDate,

StringFormat='Начало членства: {0:d}'}"/>

<TextBlock Text="{Binding MembershipEndDate,

StringFormat='Конец членство: {0:d}'}"/>

</StackPanel>

</Border>

</DataTemplate>

</ItemsControl.ItemTemplate>

</ItemsControl>

</ScrollViewer>

</Grid>

<Grid Grid.Row="1">

<Border Background="White">

<Grid>

<StackPanel Orientation="Horizontal"

HorizontalAlignment="Center"

VerticalAlignment="Bottom"

Margin="0,0,0,10">

<Button Width="60"

IsEnabled="{Binding IsFirstPage}"

Command="{Binding ToPreviousPage}"

Height="60"

ToolTip="На прошлую страницу">

<iconPacks:PackIconModern Width="22"

Height="22"

VerticalAlignment="Center"

Kind="ArrowLeft" />

</Button>

<Button Margin="10,0,0,0"

Width="60"

IsEnabled="{Binding IsLastPage}"

Height="60"

Command="{Binding ToNextPage}"

ToolTip="На следующую страницу">

<iconPacks:PackIconModern Width="22"

Height="22"

VerticalAlignment="Center"

Kind="ArrowRight" />

</Button>

</StackPanel>

<TextBlock HorizontalAlignment="Right"

FontSize="30"

VerticalAlignment="Center"

Text="{Binding Page, StringFormat='Страница: {0:d}'}"/>

<Button HorizontalAlignment="Left"

VerticalAlignment="Center"

FontSize="24"

Content="Добавить пользователя"

Width="280"

Margin="15,0,0,0"

Height="50"

Command="{Binding ToAddClient}"/>

</Grid>

</Border>

</Grid>

</Grid>

</UserControl>

<UserControl x:Class="courseWork.Views.EmptyPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:courseWork.Views"

mc:Ignorable="d"

d:DesignHeight="450"

d:DesignWidth="800"

xmlns:vm="clr-namespace:courseWork.ViewModels"

>

<UserControl.DataContext>

<vm:EmptyPageVM/>

</UserControl.DataContext>

<Grid>

<TextBlock HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="40" Text="{Binding InitText}"/>

</Grid>

</UserControl>

<Window x:Class="courseWork.Views.TrainerDetailsWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:courseWork.Views"

mc:Ignorable="d"

Title="TrainerDetailsWindow"

AllowsTransparency="True"

xmlns:vm="clr-namespace:courseWork.ViewModels"

xmlns:iconPacks="http://metro.mahapps.com/winfx/xaml/iconpacks"

WindowStyle="None"

WindowStartupLocation="CenterScreen"

ResizeMode="CanMinimize"

Height="450"

Width="450"

BorderBrush="Black"

BorderThickness="3"

Background="Transparent">

<Border Background="Beige">

<Grid Background="Transparent">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="450"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="40"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0" Grid.Column="0">

<TextBlock

Text="Страница тренера"

VerticalAlignment="Center"

HorizontalAlignment="Center"

FontSize="30"

/>

</Grid>

<Grid Grid.Row="0" Grid.Column="1">

<Button Width="32"

Height="32"

ToolTip="Закрыть?"

Click="CloseButton\_Click"

VerticalAlignment="Center"

HorizontalAlignment="Right"

Margin="0,0,10,0">

<iconPacks:BootstrapIcons

Kind="DoorClosed"/>

</Button>

</Grid>

<Grid Grid.Row="1" Grid.Column="0">

<Grid.RowDefinitions>

<RowDefinition Height="120"/>

<RowDefinition/>

</Grid.RowDefinitions>

<Grid Grid.Row="0">

<Image Source="{Binding ImagePath}"/>

</Grid>

<Border BorderThickness="2"

BorderBrush="Black"

Padding="10"

CornerRadius="20"

Background="White"

Margin="0,20,0,0"

Grid.Row="1">

<StackPanel Orientation="Vertical">

<TextBlock Text="{Binding Trainer.FirstName, StringFormat='Имя: {0:d}'}" FontSize="20"/>

<TextBlock Text="{Binding Trainer.LastName, StringFormat='Фамилия: {0:d}'}" FontSize="20"/>

<TextBlock Text="{Binding Trainer.PhoneNumber, StringFormat='Номер телефона: +{0:d}'}" FontSize="20"/>

<TextBlock Text="{Binding Trainer.Email, StringFormat='Почта: {0:d}'}" FontSize="20"/>

<TextBlock Text="{Binding Trainer.Specialization, StringFormat='Специлизация: {0:d}'}" FontSize="20"/>

<TextBlock Text="{Binding Trainer.TrainerID, StringFormat='ID: {0:d}'}" FontSize="20"/>

</StackPanel>

</Border>

</Grid>

<Grid Grid.Row="1" Grid.Column="1">

<!--<StackPanel Orientation="Vertical">

<Button

</StackPanel>-->

</Grid>

</Grid>

</Border>

</Window>

<UserControl x:Class="courseWork.Views.TrainersPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:courseWork.Views"

mc:Ignorable="d"

d:DesignHeight="{Binding Height}"

d:DesignWidth="{Binding Width}"

xmlns:vm="clr-namespace:courseWork.ViewModels"

xmlns:iconPacks="http://metro.mahapps.com/winfx/xaml/iconpacks">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0">

<ScrollViewer VerticalScrollBarVisibility="Auto">

<ItemsControl ItemsSource="{Binding Trainers}">

<ItemsControl.ItemsPanel>

<ItemsPanelTemplate>

<WrapPanel Orientation="Horizontal" ItemWidth="300" ItemHeight="180"/>

</ItemsPanelTemplate>

</ItemsControl.ItemsPanel>

<ItemsControl.ItemTemplate>

<DataTemplate>

<Border Background="White"

BorderBrush="Gray"

BorderThickness="1"

Margin="5"

Cursor="Hand"

Padding="10"

CornerRadius="5"

MouseLeftButtonUp="Border\_MouseLeftButtonUp">

<StackPanel>

<TextBlock Text="{Binding FirstName, StringFormat='Имя: {0:d}'}" FontWeight="Bold"/>

<TextBlock Text="{Binding LastName, StringFormat='Фамилия: {0:d}'}"/>

<TextBlock Text="{Binding PhoneNumber, StringFormat='Номер: {0:d}'}"/>

<TextBlock Text="{Binding Email, StringFormat='Эл. почта: {0:d}'}"/>

<TextBlock Text="{Binding Specialization,

StringFormat='Специлизация: {0:d}'}"/>

</StackPanel>

</Border>

</DataTemplate>

</ItemsControl.ItemTemplate>

</ItemsControl>

</ScrollViewer>

</Grid>

<Grid Grid.Row="1">

<Border Background="White">

<Grid>

<StackPanel Orientation="Horizontal"

HorizontalAlignment="Center"

VerticalAlignment="Center"

Margin="0,0,0,10">

<Button Width="60"

IsEnabled="{Binding IsFirstPage}"

Command="{Binding ToPreviousPage}"

Height="60"

ToolTip="На прошлую страницу">

<iconPacks:PackIconModern Width="22"

Height="22"

VerticalAlignment="Center"

Kind="ArrowLeft" />

</Button>

<Button Margin="10,0,0,0"

Width="60"

IsEnabled="{Binding IsLastPage}"

Height="60"

Command="{Binding ToNextPage}"

ToolTip="На следующую страницу">

<iconPacks:PackIconModern Width="22"

Height="22"

VerticalAlignment="Center"

Kind="ArrowRight" />

</Button>

</StackPanel>

<TextBlock HorizontalAlignment="Right"

FontSize="30"

VerticalAlignment="Center"

Text="{Binding Page, StringFormat='Страница: {0:d}'}"/>

<Button HorizontalAlignment="Left"

VerticalAlignment="Center"

FontSize="24"

Content="Добавить тренера"

Width="240"

Margin="15,0,0,0"

Height="50"

Command="{Binding ToAddTrainer}"/>

</Grid>

</Border>

</Grid>

</Grid>

</UserControl>

<mah:MetroWindow x:Class="courseWork.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:courseWork"

xmlns:mah="clr-namespace:MahApps.Metro.Controls;assembly=MahApps.Metro"

mc:Ignorable="d"

Title="Фитнес клуб - ZeVS"

WindowStartupLocation="CenterScreen"

ResizeMode="CanResizeWithGrip"

Height="450"

Width="800"

Background="Beige"

GlowBrush="{DynamicResource MahApps.Brushes.Accent}"

xmlns:iconPacks="http://metro.mahapps.com/winfx/xaml/iconpacks">

<mah:MetroWindow.LeftWindowCommands>

<mah:WindowCommands>

<Button ToolTip="Домой"

Command="{Binding NavigateToEmptyPage}"

Style="{StaticResource CustomButtonStyleTab}">

<iconPacks:PackIconModern Width="22"

Height="22"

Kind="Home" />

</Button>

</mah:WindowCommands>

</mah:MetroWindow.LeftWindowCommands>

<Border CornerRadius="20">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="40"/>

<RowDefinition/>

</Grid.RowDefinitions>

<Grid Grid.Row="0">

<StackPanel Orientation="Horizontal">

<Button Style="{StaticResource CustomButtonStyleTab}"

FontSize="30"

Content="Клиенты"

ToolTip="Перейти на страницу клиентов"

Command="{Binding NavigateToClientsPageCommand}"

/>

<Button Style="{StaticResource CustomButtonStyleTab}"

FontSize="30"

Content="Тренера"

ToolTip="Перейти на страницу тренеров"

Command="{Binding NavigateToTrainersPage}"

Margin="15,0,0,0"/>

<Button Style="{StaticResource CustomButtonStyleTab}"

FontSize="30"

Content="Бронирования"

ToolTip="Перейти на страницу бронирования"

Command="{Binding NavigateToBookingPage}"

Margin="15,0,0,0"/>

</StackPanel>

</Grid>

<Grid Grid.Row="1">

<UserControl Content="{Binding CurrentPage}"/>

</Grid>

</Grid>

</Border>

</mah:MetroWindow>