**Министерство образования Иркутской области**

Государственное бюджетное профессиональное образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

|  |  |
| --- | --- |
| **ДП.09.02.03.23.191.12.ПЗ** | УТВЕРЖДАЮ  Зам. директора по УР, к.т.н.  \_\_\_\_\_\_\_\_\_\_\_Е.А. Коробкова |

**МОБИЛЬНОЕ ПРИЛОЖЕНИЕ**

**«ФИТНЕС-КЛУБ»**

|  |  |  |
| --- | --- | --- |
| Нормконтролер: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (В.А. Пролыгина) |
| Консультант по экономической части: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (М.А. Рачкова) |
| Руководитель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (О.Н. Филимонова) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (А.А.Иванова) |

Иркутск 2023

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc135354172)

[1 Предпроектное исследование 5](#_Toc135354173)

[1.1 Описание предметной области 5](#_Toc135354174)

[1.2 Анализ инструментальных средств реализации 7](#_Toc135354175)

[2 Техническое задание на разработку программного продукта 11](#_Toc135354176)

[3 Проектирование 12](#_Toc135354177)

[3.1 Архитектура программного обеспечения 12](#_Toc135354178)

[3.2 Функциональное проектирование 13](#_Toc135354179)

[3.3 Проектирование базы данных 18](#_Toc135354180)

[3.4 Проектирование пользовательского интерфейса 21](#_Toc135354181)

[2 Реализация программного обеспечения 24](#_Toc135354182)

[4.1 Кодирование программного обеспечения 24](#_Toc135354183)

[4.2 Разработка базы данных 24](#_Toc135354184)

[4.3 Разработка программного продукта 27](#_Toc135354185)

[5 Документирование программного обеспечения 34](#_Toc135354186)

[5.1 Руководство пользователя 34](#_Toc135354187)

[ЗАКЛЮЧЕНИЕ 39](#_Toc135354188)

ВВЕДЕНИЕ

За последние несколько лет значительно возрос объем и оборот информации во всех сферах человеческой деятельности: экономической, финансовой, политической, духовной. В связи с увеличением информации возникает необходимость в использовании автоматических инструментов, позволяющих хранить, эффективно обрабатывать и распределять накопленные данные.

Преимуществом автоматизации является сокращение избыточности хранимых данных, а, следовательно, экономия объема используемой памяти, уменьшение затрат на многократные операции, увеличение степени достоверности информации и увеличение скорости обработки информации; излишнее количество внутренних промежуточных документов, различных журналов, папок, заявок и т.д., повторное внесение одной и той же информации в различные промежуточные документы. Также значительно сокращает время автоматический поиск информации, который производится из специальных экранных форм, в которых указываются параметры поиска объекта.

Основой задачей данного мобильного приложения является учет абонентов фитнес-клуба, подготовка документов для внешней среды (приказов, служебных записок, отчетов). Эти задачи имеют регулярный характер и выполняются непосредственными исполнителями (системными-администраторами). Результаты выполнения операций регистрируются в соответствующих регистрах. Автоматизация этих процессов позволит хранить информацию в одной базе, в которую вводятся данные с помощью удобного интерфейса.

Целью дипломного проекта является разработка мобильного приложения учета клиентов для фитнес-клуба, приводящее к более эффективной работе.

В ходе дипломного проектирования были поставлены следующие задачи:

1. Провести предпроектное исследование;
   1. Провести исследование предметной области;
   2. Проанализировать инструменты, используемые в разработке программного обеспечения;
   3. Обосновать выбор программных продуктов для разработки;
2. Составить техническое задание на разработку программного продукта;
3. Провести проектирование программного продукта;
   1. Представить архитектуру программного обеспечения;
   2. Провести функциональное проектирование;
   3. Спроектировать базу данных;
   4. Спроектировать пользовательский интерфейс программного продукта;
4. Реализовать программный продукт;
5. Разработать документы для программного продукта;
6. Рассчитать стоимость разработки и внедрения программного продукта.
7. Предпроектное исследование
   1. Описание предметной области

Предметной областью дипломного проекта является мобильное приложение «Фитнес-клуб».

В настоящее время такие заведения получили большое признание среди общественности. Всё больше и больше людей стало посещать подобные места, направленные на создание, поддержание и укрепление здоровья для полноценного существования в обществе, создание материально-производственной базы, планирование, организация и проведение массовых коммерческих спортивно-оздоровительных мероприятий, пропаганда здорового образа жизни.

Фитнес-клуб включает в себя очень много направленностей, таких как: бассейн, тренажерный зал, залы аэробики и различных групповых программ, массажные кабинеты и многое другое.

Обслуживание клиентов осуществляется таким образом:

1) Происходит выбор основных услуг:

− Тренажерный зал,

− Кардио-зона,

− Аэробные залы,

− Бассейн,

− SPA комплекс,

− Инструктажи в залах,

2) Далее выбор дополнительных услуг:

− Индивидуальные тренировки с тренером,

− Массаж,

− Мастер классы по ведущим направлениям аэробных занятий (Йога, Школа плавания, Пилатес и т.д).

3) Подборка подходящего абонемента, исходя из пожеланий клиента;

4) Проведение оплаты;

5) Назначение исполнителей;

6) Оказание соответствующих услуг.

Весь персонал сети фитнес-клуба можно разделить на группы, занимающиеся выполнением определенных функций:

− Руководитель;

− Специалисты;

− Исполнители;

− Обслуживающий персонал.

Встает проблема регистрации и учета клиентов, и оптимизации работы с ними. Решением данной проблемы является создание удобной программы учета клиентов, которая способна выполнять те задачи, которые перед ней ставятся. Это позволит увеличить скорость и качество обслуживания клиентов, предоставит дополнительные возможности для развития компании и увеличит потенциально возможный поток клиентов.

* 1. Анализ инструментальных средств реализации

Для разработки мобильного приложения было рассмотрено несколько инструментальных средств разработки программного обеспечения. Такие как: C#, Python, JavaScript.

Python – это высокоуровневый язык программирования общего назначения. Язык является полностью объектно-ориентированным в том плане, что всё является объектами. Необычной особенностью языка является выделение блоков кода пробельными отступами.

JavaScript – динамический скриптовый язык программирования высокого уровня. Он отличается мультипарадигменностью. Чаще всего язык используется для создания интерактивных веб-страниц и приложений. Неизменно высокий интерес к JavaScript подтверждают специальные международные рейтинги.

C# – современный объектно-ориентированный и типобезопасный язык программирования. C# позволяет разработчикам создавать разные типы безопасных и надежных приложений, выполняющихся в .NET.

Таблица 1 – Сравнение языков программирования

|  |  |  |  |
| --- | --- | --- | --- |
| Критерии | Python | JavaScript | C# |
| Наличие библиотек | + | + | + |
| Инструменты для работы с БД | + | + | + |
| Объектно-ориентированные возможности | + | + | + |

Исходя из данной таблицы, было принято решение остановится на языке программирования C# потому, что он достаточно прост и понятен, что поможет реализовать данный продукт.

Разберем среды разработки. Выбор встал между Visual Studio и JetBrains Rider.

Visual Studio – линейка продуктов компании [Майкрософт](https://dic.academic.ru/dic.nsf/ruwiki/1018162), включающих [интегрированную среду разработки](https://dic.academic.ru/dic.nsf/ruwiki/940808) программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как [консольные](https://dic.academic.ru/dic.nsf/ruwiki/978372) [приложения](https://dic.academic.ru/dic.nsf/ruwiki/4631), так и приложения с [графическим интерфейсом](https://dic.academic.ru/dic.nsf/ruwiki/22252), в том числе с поддержкой технологии [Windows Forms](https://dic.academic.ru/dic.nsf/ruwiki/74841), а также [веб-сайты](https://dic.academic.ru/dic.nsf/ruwiki/1135216), [веб-приложения](https://dic.academic.ru/dic.nsf/ruwiki/35038), [веб-службы](https://dic.academic.ru/dic.nsf/ruwiki/80509) как в [родном](https://dic.academic.ru/dic.nsf/ruwiki/28842), так и в [управляемом](https://dic.academic.ru/dic.nsf/ruwiki/57362) кодом для всех платформ,

JetBrains Rider – кроссплатформенная интегрированная среда разработки программного обеспечения для платформы .NET, разрабатываемая компанией JetBrains. Поддерживаются языки программирования C#.

Для выбора среды разработки была составлена таблица 2.

Таблица 2 – Сравнение сред разработки

|  |  |  |
| --- | --- | --- |
| Название IDE | Visual Studio | JetBrains Rider |
| Автосохранение | + | + |
| Автодополнение | + | + |
| Поиск по коду | + | + |

После сравнения были сделаны выводы, что для выбранного дипломного проекта лучшее всего подойдет Visual Studio потому, что данная программа имеет хороший функционал, что поможет в реализации программного продукта. Также для создания мобильного приложения на С# был выбран фреймворк Xamarin обладающий необходимым функционалом для успешной разработки.

Для работы приложения необходимы базы данных, поэтому необходимо также провести анализ для выбора средств реализации баз данных.

MySQL достаточно легко инсталлируется, а наличие множества плагинов и вспомогательных приложений упрощает работу с базами данных. Обширный функционал. Система MySQL обладает практически всем необходимым инструментарием, который может понадобиться в реализации практически любого проекта.

Преимущества:

* простота в использовании. MySQL достаточно легко инсталлируется, а наличие множества плагинов и вспомогательных приложений упрощает работу с базами данных;
* обширный функционал. Система MySQL обладает практически всем необходимым инструментарием, который может понадобиться в реализации практически любого проекта;
* безопасность. Система изначально создана таким образом, что множество встроенных функций безопасности в ней работают по умолчанию;
* масштабируемость. Являясь весьма универсальной СУБД, MySQL в равной степени легко может быть использована для работы и с малыми, и с большими объемами данных;
* скорость. Высокая производительность системы обеспечивается за счет упрощения некоторых используемых в ней стандартов.

SQLite – это быстрая и легкая встраиваемая однофайловая СУБД на языке C, которая не имеет сервера и позволяет хранить всю базу локально на одном устройстве. Для работы SQLite не нужны сторонние библиотеки или службы.

Преимущества:

* файловая: вся база данных хранится в одном файле, что облегчает перемещение;
* стандартизированная: SQLite использует SQL; некоторые функции опущены (RIGHT OUTER JOIN или FOR EACH STATEMENT), однако, есть и некоторые новые;
* отлично подходит для разработки и даже тестирования: во время этапа разработки большинству требуется масштабируемое решение. SQLite, со своим богатым набором функций, может предоставить более чем достаточный функционал, при этом будучи достаточно простой для работы с одним файлом и связанной библиотекой.

PostgreSQL – это самая продвинутая РСУБД, ориентирующаяся в первую очередь на полное соответствие стандартам и расширяемость. PostgreSQL, или Postgres, пытается полностью соответствовать SQL-стандартам ANSI/ISO.

Преимущества:

* полная SQL-совместимость;
* сообщество: PostgreSQL поддерживается опытным сообществом 24/7;
* поддержка сторонними организациями: несмотря на очень продвинутые функции, PostgreSQL используется в многих инструментах, связанных с РСУБД;
* расширяемость: PostgreSQL можно программно расширить за счёт хранимых процедур;
* объектно-ориентированность: PostgreSQL – не только реляционная, но и объектно-ориентированная СУБД.

Сравним СУБД, чтобы определиться, какая больше подойдет для данной дипломной работы (таблица 3).

Таблица 3 – Сравнение средств реализации базы данных

|  |  |  |  |
| --- | --- | --- | --- |
| Название СУБД | MySQL | SQLite | PostgreSQL |
| Большое количество типов данных | + | - | + |
| Популярность | + | + | - |
| Отказоустойчивость | - | - | + |
| Не требует удаленного сервера | - | + | - |
| Продолжение таблицы 3 | | | |
| Простота использования | + | + | - |
| Портативность | + | + | - |

При сравнении СУБД для данного дипломного проекта была выбрана MySQL, что поможет лучше реализовать данный продукт.

1. Техническое задание на разработку программного продукта

В начале разработки создавалось техническое задание, в котором указывались основные требования.

Для создания технического задания использовался стандарт ГОСТ 34.602-89.

Согласно ГОСТ 34.602-89 техническое задание должно включать следующие разделы:

1. общие сведения;
2. назначение и цели создания системы;
3. требования к системе в целом;
   1. требования к структуре и функционированию системы;
   2. требования к надежности;
   3. требования к безопасности;
   4. требования к эксплуатации, техническому обслуживанию, ремонту и хранению компонентов системы;
4. требования к документированию;
5. состав и содержание работ по созданию системы.

Техническое задание на разработку приложения представлено отдельным документом.

1. Проектирование
   1. Архитектура программного обеспечения

Архитектура программного обеспечения – это базовая организация системы, воплощенная в ее компонентах, их отношениях между собой и с окружением, а также принципы, определяющие проектирование и развитие системы.

Мобильное приложение «Фитнес-клуб» использует клиент-серверную архитектуру (рисунок 1). Архитектура «клиент-сервер» предусматривает разделение процессов предоставление услуг и отправки запросов на них на разных компьютерах в сети, каждый из которых выполняют свои задачи независимо от других.

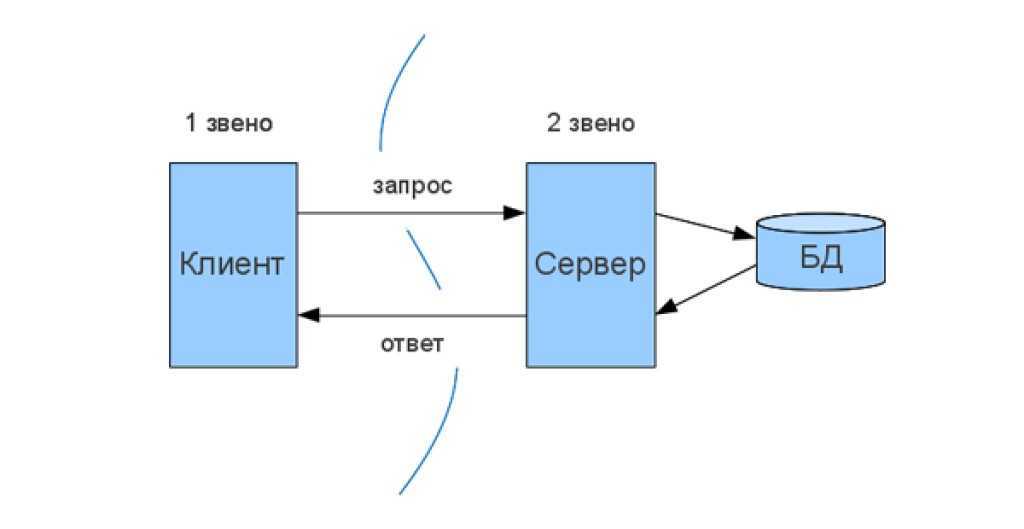


Рисунок 1 – Архитектура мобильного приложения «Фитнес-клуб»

Сервер – специальное системное оборудование, которое предназначается для разрешения определенного круга задач по процессу выполнения программных кодов. Он выполняет работы сервисного обслуживания по клиентским запросам, предоставляет пользователям доступ к определенным системным ресурсам, сохраняет данные или БД.

Параметры, которые могут реализоваться на стороне сервера:

* хранение, защита и доступ к данным;
* работа с поступающими клиентскими запросами;
* процесс отправки ответа клиенту.

Клиент – локальный компьютер на стороне виртуального пользователя, который выполняет отправку запроса к серверу для возможности предоставления данных или выполнения определенной группы системных действий.

Параметры, которые могут реализоваться на стороне клиента:

* площадка по предоставлению пользовательского графического интерфейса;
* формулировка запроса к серверу и его последующая отправка;
* получение итогов запроса и отправка дополнительной группы команд (запросы на добавление, обновление информации, удаление группы данных).

Таким образом, в архитектуре «клиент-сервер» клиент посылает запрос на предоставление данных и получает только те данные, которые действительно были затребованы.

* 1. Функциональное проектирование

Проектирование мобильного приложения происходит при помощи CASE средств, которые помогают обеспечить высокое качество программ, отсутствие ошибок и простоту в обслуживании программных продуктов.

На рисунке 2 изображена структурная схема, она описывает функциональное назначение системы, т.е. то, что система будет делать в процессе своего функционирования. Основными элементами диаграммы являются участник (actor) и прецедент (вариант).

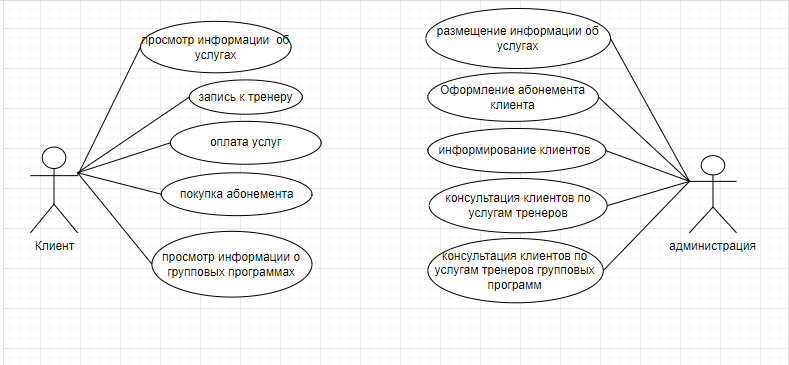


Рисунок 2 – Диаграмма прецедентов

Представленные диаграммы демонстрируют взаимодействие основных объектов мобильного приложения и их действия.

На рисунке 3 изображена диаграмма деятельности. Применяется эта диаграмма для того, чтобы промоделировать последовательные и при необходимости параллельные шаги вычислительного процесса. Сценарий работы с приложением показан на рисунке 3

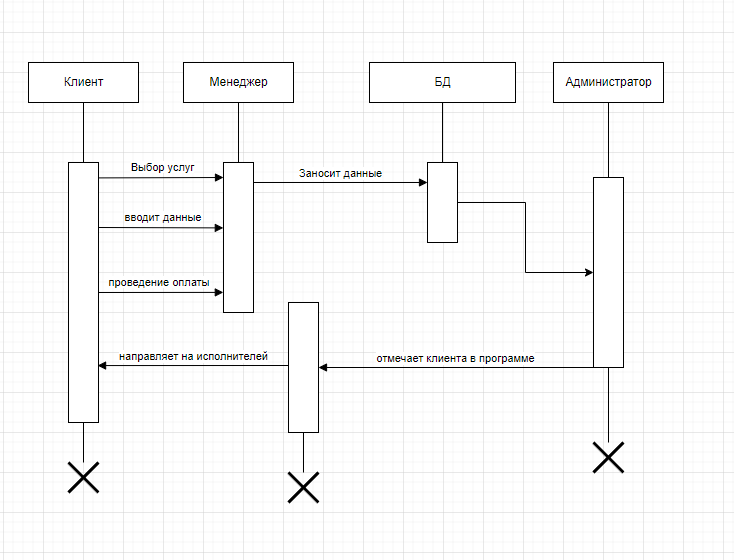


Рисунок 3 –Диаграмма последовательности

На рисунке 4 изображена диаграмма компонентов, которая показывает разбиение программной системы на структурные компоненты и связи (зависимости) между компонентами. В качестве физических компонентов могут выступать файлы, библиотеки, модули, исполняемые файлы, пакеты и т. п.

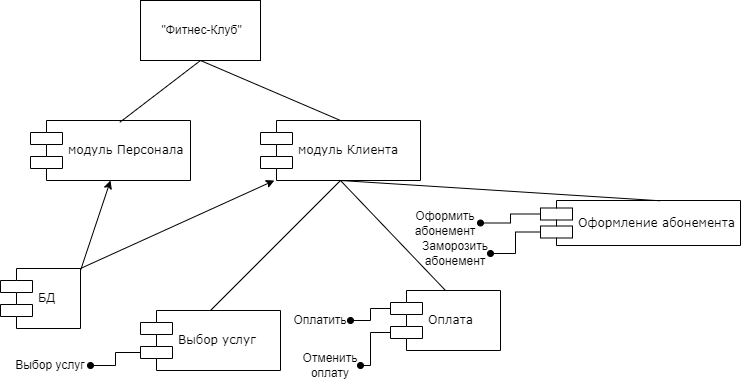


Рисунок 4 – Диаграмма компонентов

На рисунке 5 представлена диаграмма развертывания – это тип диаграммы, которая определяет физическое оборудование, на котором будет работать программная система. Он также определяет способ развертывания программного обеспечения на базовом оборудовании. Он отображает программные части системы на устройство, которое будет выполнять его.

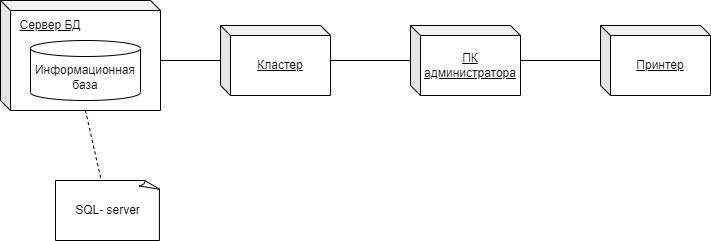


Рисунок 5 – Диаграмма развертывания

Контекстная диаграмма – это модель, представляющая систему как набор иерархических действий, в которой каждое действие преобразует некоторый объект или набор объектов.

На рисунке 6 изображена контекстная диаграмма, на которой показаны входные данные, управление, механизм, выходные данные и функция.

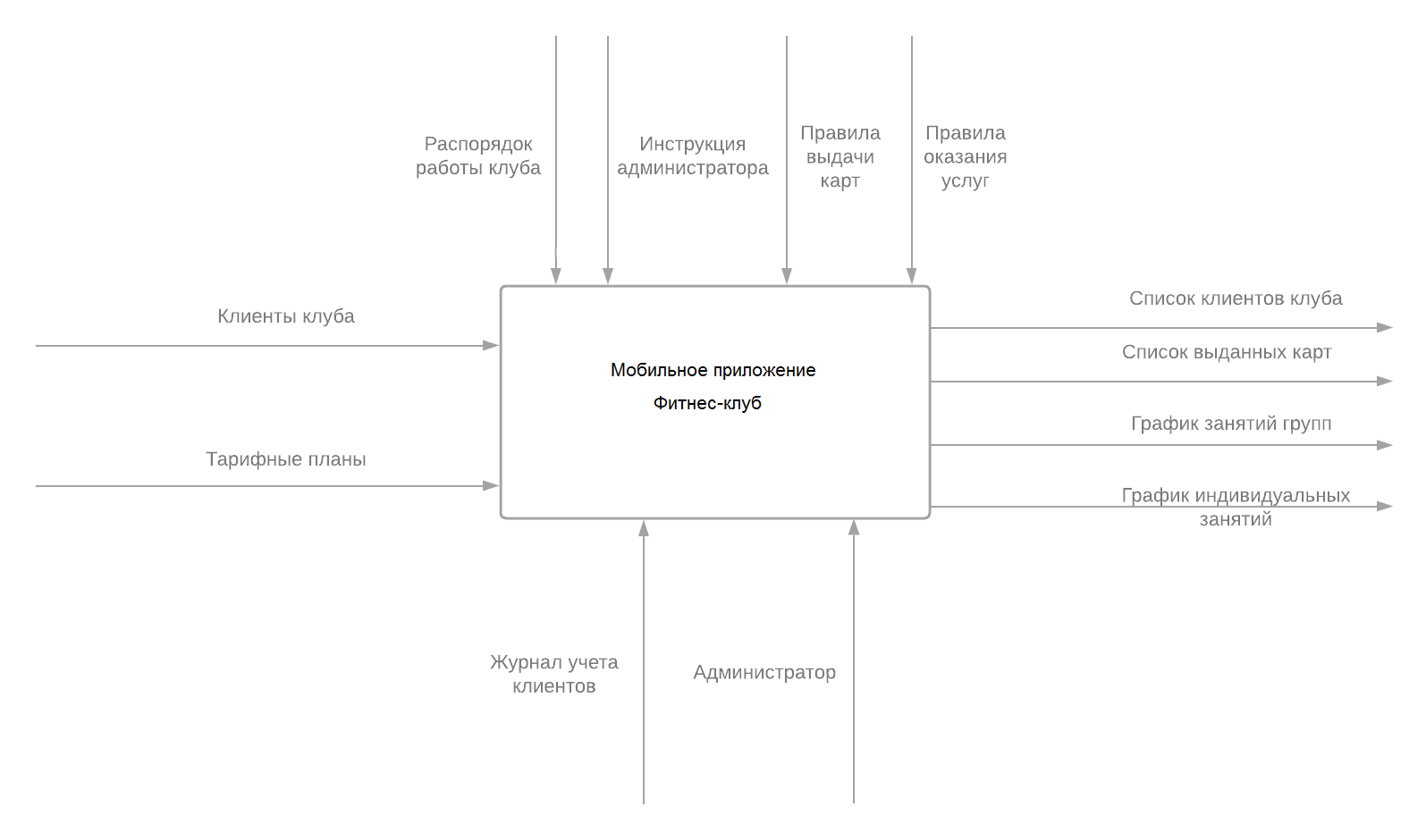


Рисунок 6 – Контекстная диаграмма

На рисунке 7 изображена диаграмма декомпозиции. Она показывает комплекс, процесс, организацию, предметную область данных или другой тип объекта, разбитый на более детализированные компоненты более низкого уровня. Например, диаграммы декомпозиции могут представлять организационную структуру или функциональную декомпозицию на процессы.

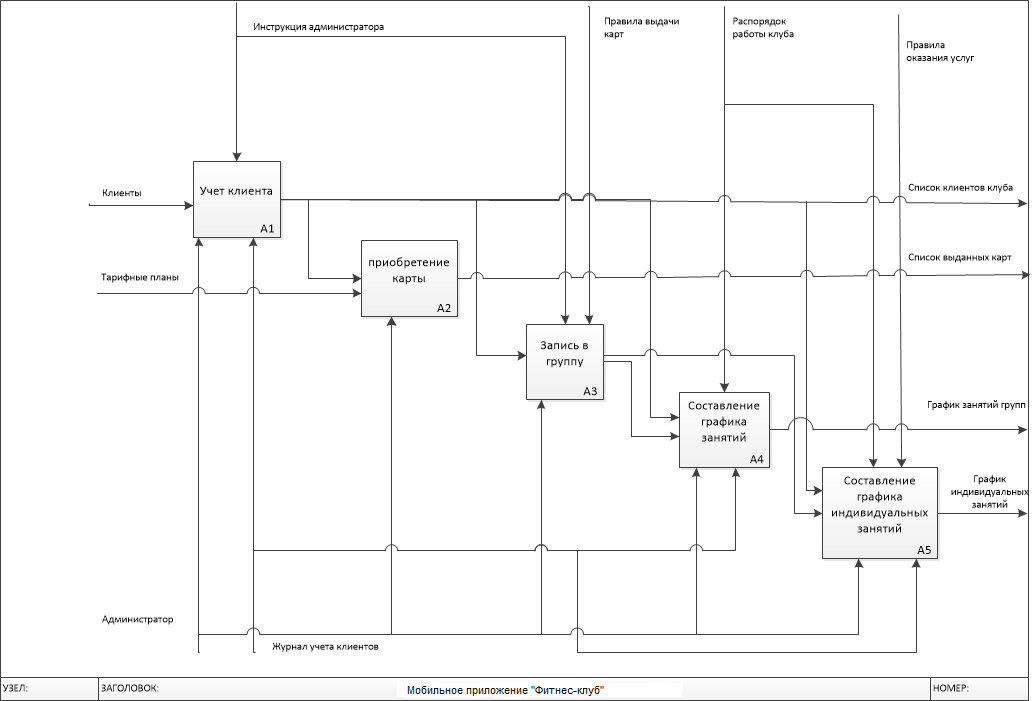


Рисунок 7 – Диаграмма декомпозиции составления расписания

Также в ходе работы была разработана диаграмма классов, представленная на рисунке 8. Диаграмма классов — структурная диаграмма языка моделирования UML, демонстрирующая общую структуру иерархии классов системы, их коопераций, атрибутов (полей), методов, интерфейсов и взаимосвязей (отношений) между ними. Широко применяется не только для документирования и визуализации, но также для конструирования посредством прямого или обратного проектирования.

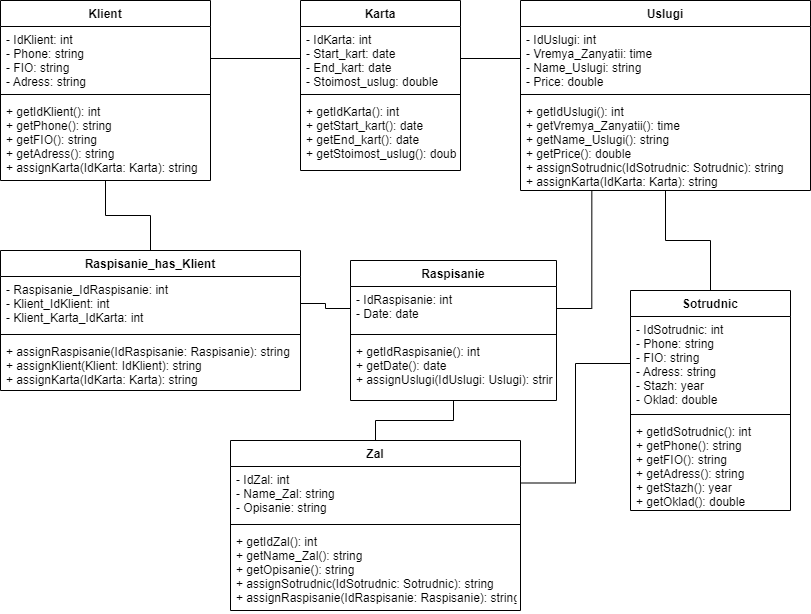
**

Рисунок 8 – Диаграмма классов

Диаграмма потоков данных, представленная на рисунке 9. Так называется методология графического структурного анализа, описывающая внешние по отношению к системе источники, и адресаты данных, логические функции, потоки данных и хранилища данных, к которым осуществляется доступ.

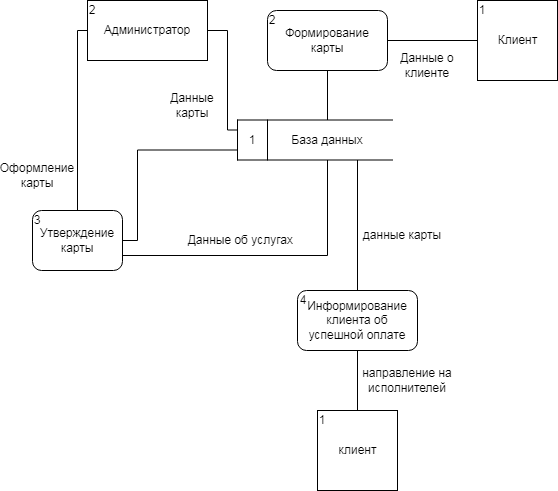


Рисунок 9 – Диаграмма потоков данных

Функциональное проектирование представляет наиболее общий подход к описанию систем. Определяются граничные условия и желательные входы, и выходы, составляется подробный перечень функций или операций, которые должны выполняться. При функциональном проектировании осуществляется синтез структуры и определяются основные параметры объекта и его составных частей (элементов), оцениваются показатели эффективности и качества процессов функционирования.

* 1. Проектирование базы данных

Перед разработкой базы данных есть необходимость в инфологическом моделировании. Результатом такого моделирования является инфологическая модель базы данных, изображенная на рисунке 10, которая наглядно показывает объект, его свойства и отношения между другими объектами.

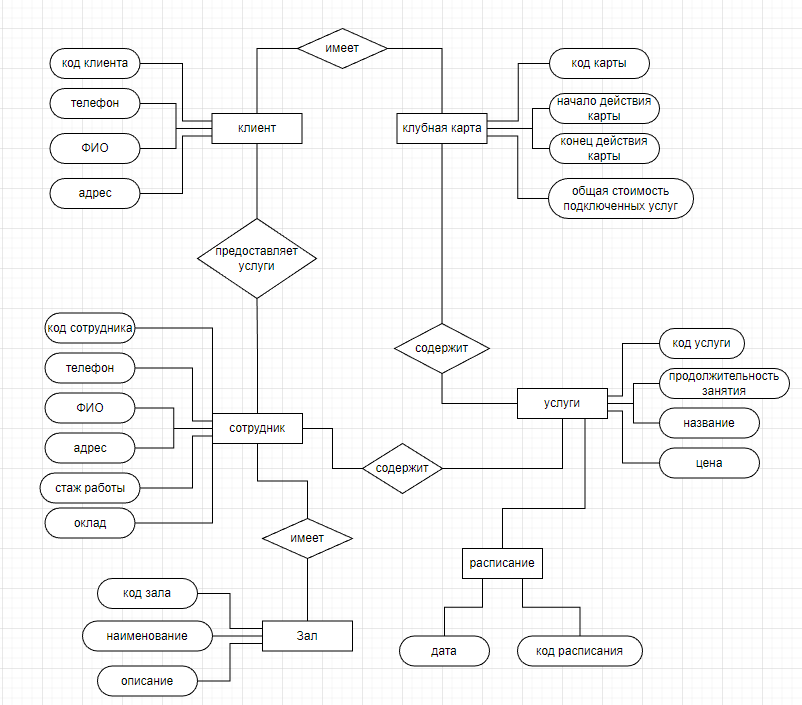


Рисунок 10 – Инфологическая модель БД

На рисунке 11 представлена даталогическая модель. Модель предполагает определение состава и взаимосвязей таблиц, отражающих содержание информационных сущностей инфологической модели в терминах конкретной СУБД.

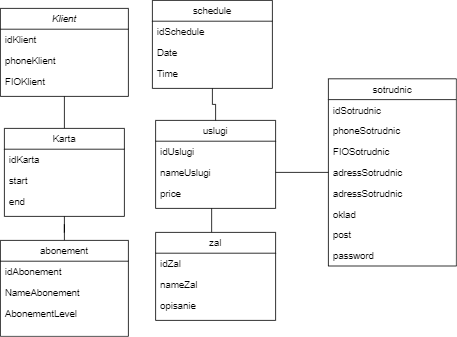


Рисунок 11 – Даталогическая модель

На рисунке 12 представлена ER–модель базы данных. Представлены таблицы, связи между ними и типы данных. В ходе проектирования произошли изменения, в связи с изменением программы. Из-за чего было принято решение добавить еще одну таблицу.

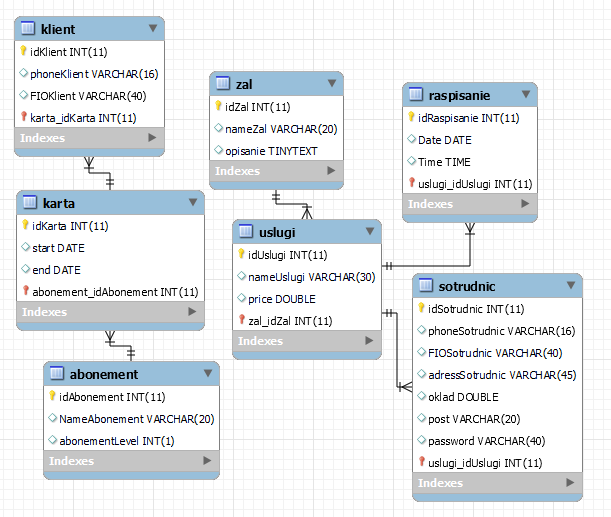


Рисунок 12 – ER-модель базы данных

Далее представим сущность таблиц базы данных.

Таблица 4 – Таблица «Клиент»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| idKlient | int | Код клиента |
| Phone | Varchar(16) | Телефон |
| FIO | Varchar(40) | ФИО |
| Karta\_IdKarta | int | Код карты |

Таблица 5 – Таблица «Карта»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdKarta | int | Код карты |
| Start\_Kart | date | Начало абонемента |
| End\_Kart | date | Окончание абонемента |
| Abonement\_idAbonement | int | Код абонемента |

Таблица 6 – Таблица «Услуги»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdUslugi | int | Код услуги |
| Name\_Uslugi | Text(50) | Название услуги |
| Price | Double | Цена услуги |
| Zal\_idZal | int | Код зала |

Таблица 7 – Таблица «Сотрудник»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdSotrudnic | int | Код сотрудника |
| phoneSotudnik | Varchar(16) | Телефон сотрудника |
| FIOSotrudnik | Varchar(40) | ФИО сотрудника |
| Adress | Varchar(45) | Адрес сотрудника |
| Post | Varchar(20) | Должность |
| Password | Varchar(40) | Пароль |
| Oklad | Double | Оклад |

Таблица 8 – Таблица «Зал»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdZal | int | Код зала |
| Name\_Zal | Text(20) | Название зала |
| Opisanie | Text(60) | Описание зала |

Таблица 9 – Таблица «Расписание»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| IdRaspisanie | int | Код расписания |
| Date | Datetime | Дата расписания |
| Time | Time | Время |
| Uslugi\_IdUslugi | int | Код услуги |

Таблица 10 – Таблица «Абонемент»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| idAbonement | int | Код абонемента |
| NameAbonement | Varchar(20) | Название абонемента |
| AbonementLevel | int(1) | Уровень абонемента |

Для разработки даталогической был использован инструмент Draw.io. Для разработки ER-модели был использован MySQL Workbench 8.0 CE. После завершения разработки прототипа базы данных и таблиц, находящихся в ней, можно перейти к проектированию интерфейса.

* 1. Проектирование пользовательского интерфейса

Пользовательский интерфейс (UI) - это взаимодействие между пользователем и компьютерной системой, которое позволяет пользователю управлять функциональностью и настройками программного обеспечения с помощью графических элементов, таких как кнопки, текстовые поля, меню и диалоговые окна.

Разработка удобного пользовательского интерфейса – это один из важнейших этапов в процессе создания программного продукта.

В результате проектирование интерфейса будущего мобильного

приложения были спроектированы прототипы трёх форм: Окно авторизации (рисунок 13), окно регистрации (рисунок 14), и окно аккаунта пользователя (рисунок 15).

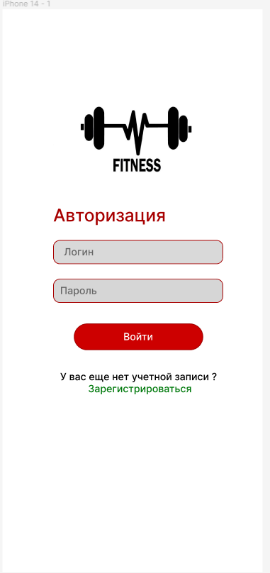


Рисунок 13 – Интерфейс авторизации

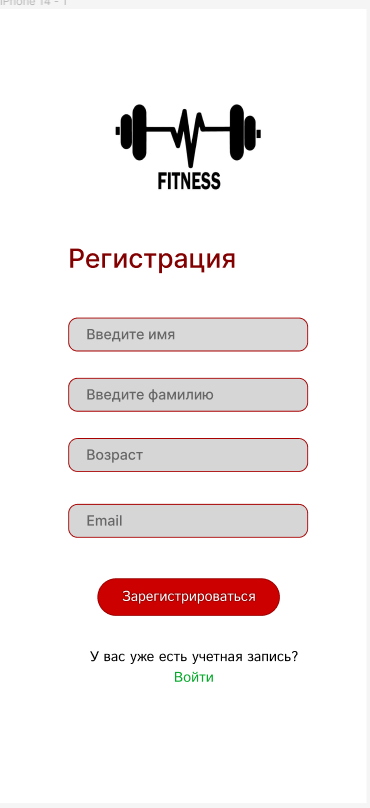


Рисунок 14 – Интерфейс регистрации



Рисунок 15 – Интерфейс страницы пользователя

1. Реализация программного обеспечения
   1. Кодирование программного обеспечения

Разработка удобного пользовательского интерфейса – это один из важнейших этапов в процессе создания приложения «Фитнес-клуб».

В разрабатываемом мобильном приложении используется формат - \*.xaml и \*.cs. Файл с форматом \*.xaml содержит в себе дизайнерскую часть, а \*.cs – программную часть, написанную на языке C#.

Благодаря фреймворку Xamarin были созданы страницы для приложения (рисунок 16).

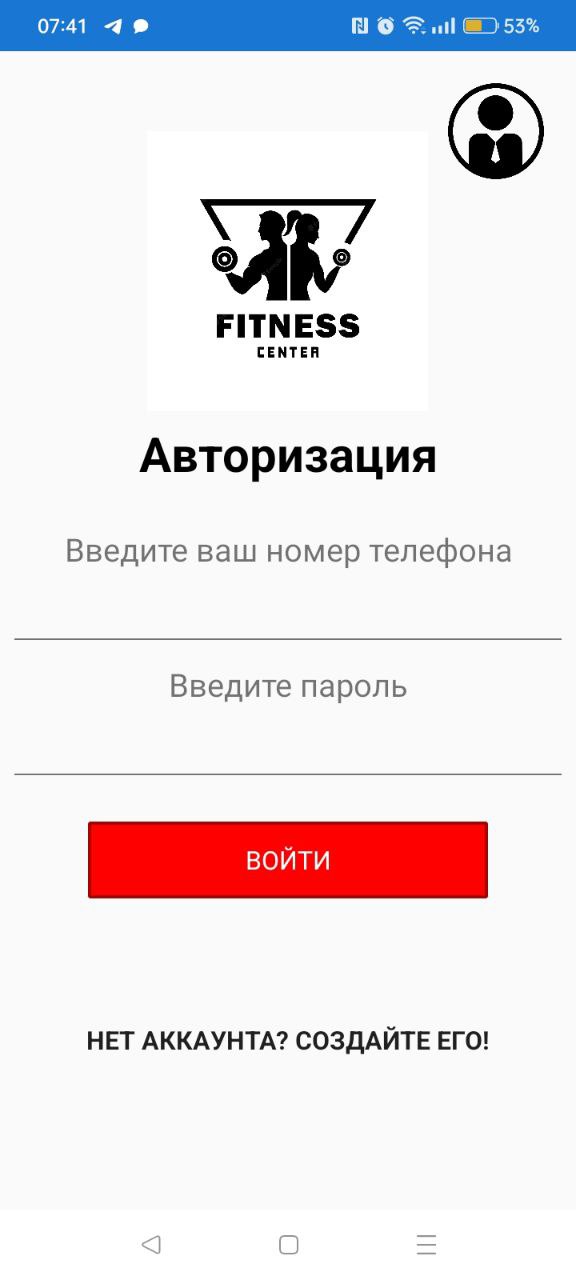


Рисунок 16 – Страница приложения

4.2 Разработка базы данных

Разработка базы данных мобильно приложения «Фитнес-клуб» реализовывалась в СУБД PhpMyAdmin. База данных состоит из 7 таблиц (рисунок 17).

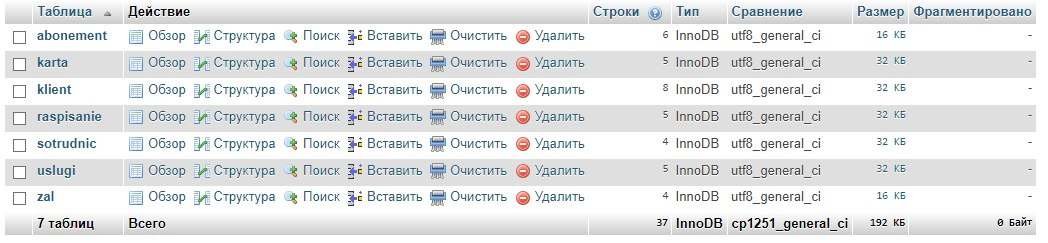


Рисунок 17 – СУБД phpMyAdmin и БД

Структуры таблиц соответствуют схеме базы данных из пункта 4.3.

Таблица «Uslugi» на рисунке 18.

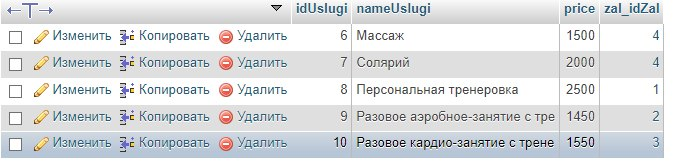


Рисунок 18 – Структура таблицы «Uslugi»

Таблица «Abonement» на рисунке 19.

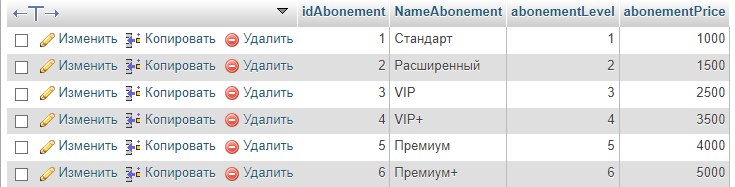


Рисунок 19 – Структура таблицы «Abonement»

Таблица «Karta» на рисунке 20.



Рисунок 20 – Структура таблицы «Karta»

Таблица «Klient» на рисунке 21.

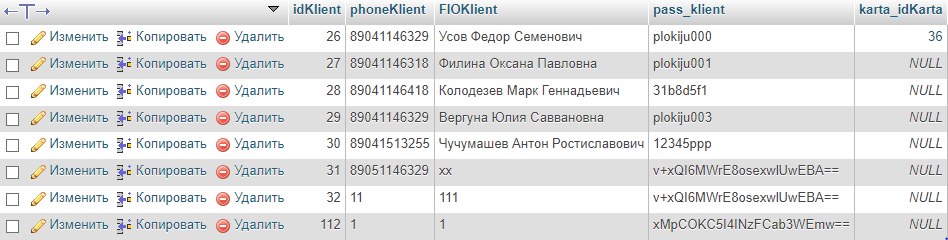


Рисунок 21 – Структура таблицы «Klient»

Таблица «Raspisanie» на рисунке 22.



Рисунок 22 – Структура таблицы «Raspisanie»

Таблица «Sotrudnic» на рисунке 23.



Рисунок 23 – Структура таблицы «Sotrudnic»

Таблица «Zal» на рисунке 24.



Рисунок 24 – Структура таблицы «Zal»

После создания всех таблиц и связей в базе данных, она готова к работе.

4.3 Разработка программного продукта

Для разработки приложения был использован Xamarin.Forms. Данный фреймворк позволил сделать удобный интерфейс для клиентов и сотрудников.

Вся основная работа построена на sql запросах в \*.cs:

1. Вывод заполнения профиля клиента (рисунок 25).
2. Вывод актуального расписания (рисунок 26).

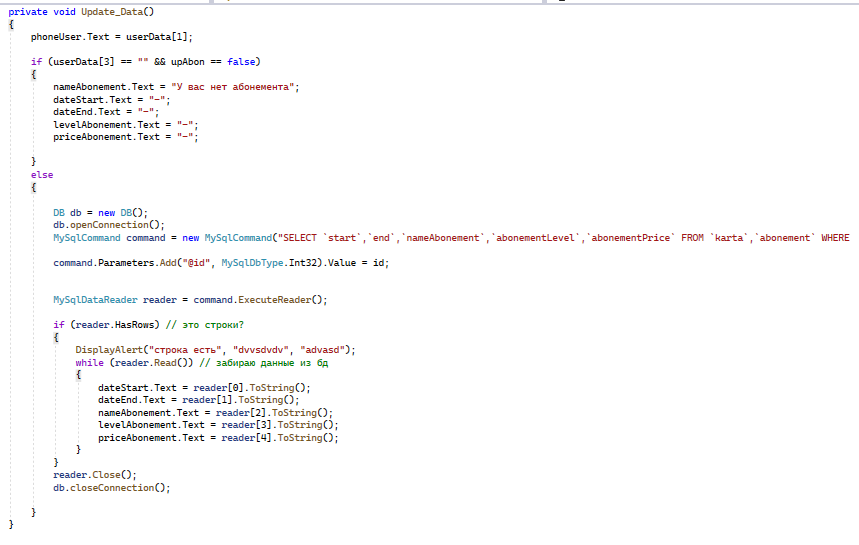


Рисунок 25 – Фрагмент кода заполнения профиля клиента

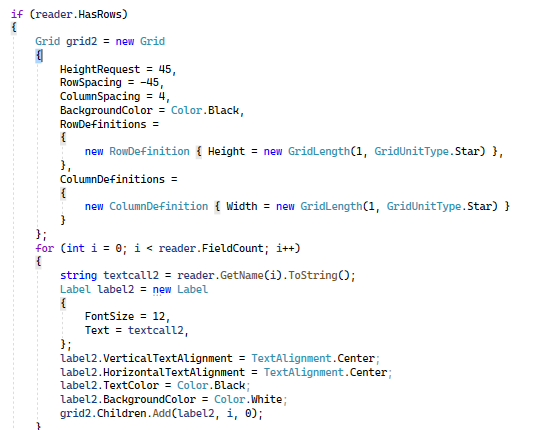


Рисунок 26 – Фрагмент кода вывода расписания

Реализация программного кода авторизации представлена в Приложение А. Вывод текущего расписания реализован с помощью C#. Данные сохраняются в базу данных. Программный код реализации просмотра расписания представлен в приложении Б.

Таким образом, были реализованы следующие функции на соответствующих страницах:

1. кнопка – «изменить абонемент», изменяет абонемент клиенту и данные на его странице. На рисунке 27 представлена страница редактирования абонемента.

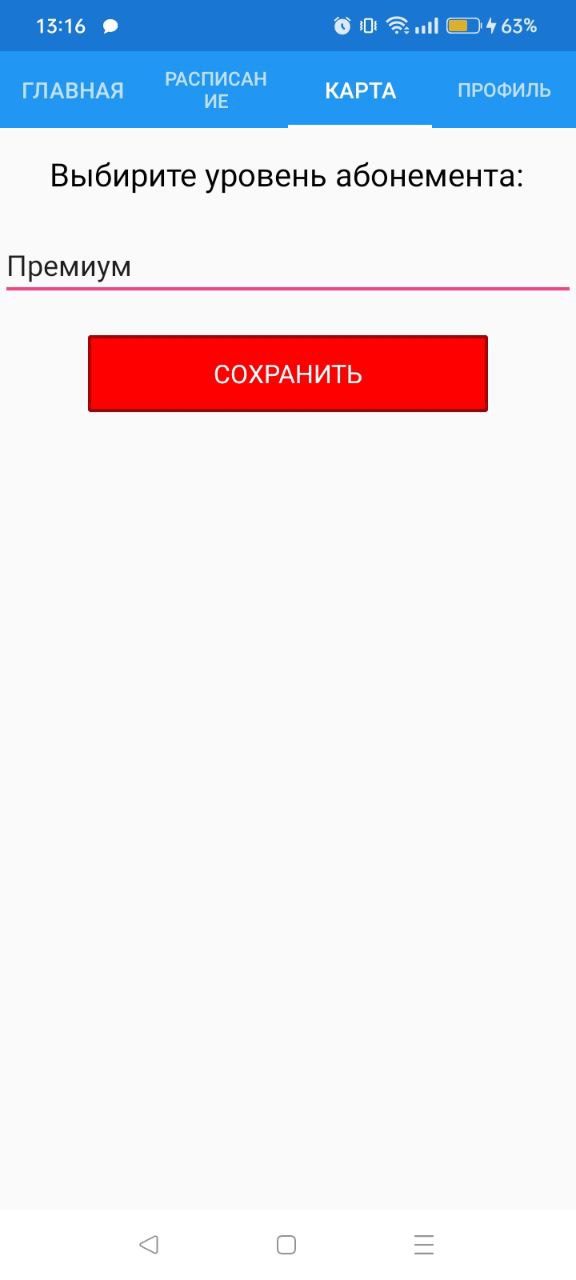


Рисунок 27 – Страница редактирования абонемента

1. Кнопка «Профиль», где описывается информация о клиенте: уровень абонемента клиента, срок действия карты, стоимость выбранного абонемента, баланс клиента, данные клиента, которые показаны на рисунках 28, 29.

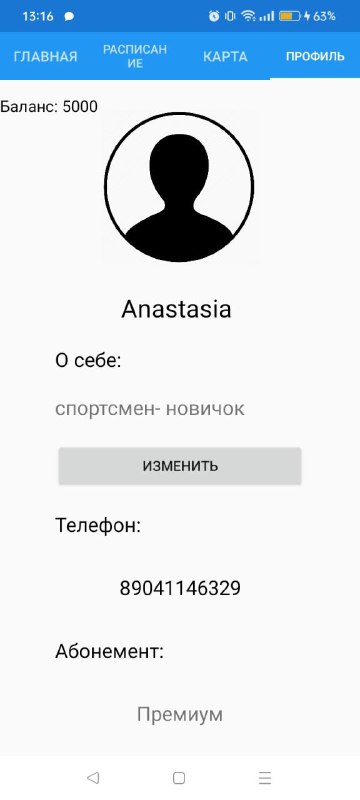


Рисунок 28 – Страница личного кабинета

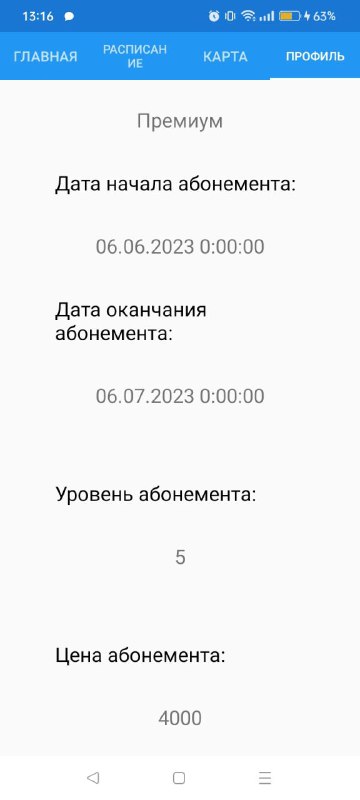


Рисунок 29 – Страница личного кабинета

1. Кнопка «Полное расписание», где пользователь может ознакомиться с полным актуальным расписанием. Страница представлена на рисунке 30.

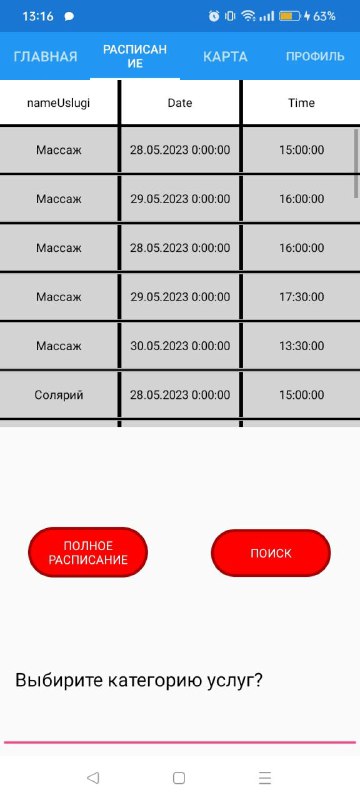


Рисунок 30 – Страница расписания

1. Кнопка «Поиск», с помощью которой пользователь сможет найти необходимое занятие в большом списке. Представлено на рисунке 30.
2. Кнопка «Добавить», представленная на рисунке 31, позволяет администратору добавить новую запись в общее расписание.
3. Кнопка «Изменить». Позволяет изменить уже имеющиеся данные в расписании. Представлена на рисунке 31.

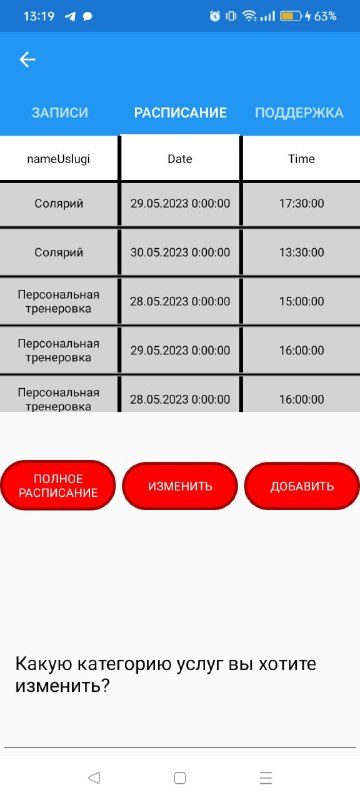


Рисунок 31 – Страница редактирования расписания

Приложение будет использоваться для оформления абонемента, легко в освоении и проста в использовании. Для работы приложения используется мобильный телефон на системе Android. Приложение разработано для использования в повседневной жизни.

1. Документирование программного обеспечения

Процесс написания документов программного обеспечения – важный этап в процессе создания и эксплуатации программного обеспечения. Именно с руководства пользователя начинает свое знакомство с программным продуктом.

* 1. Руководство пользователя

Мобильное приложение «Фитнес-клуб» имеет простой и понятный интерфейс, что позволяет пользователю легко использовать его.

В первую очередь, необходимо запустить само приложение. Оно находится на главной странице телефона. Кликаем на иконку с подписью «Fitnes», как показано на рисунке 32.

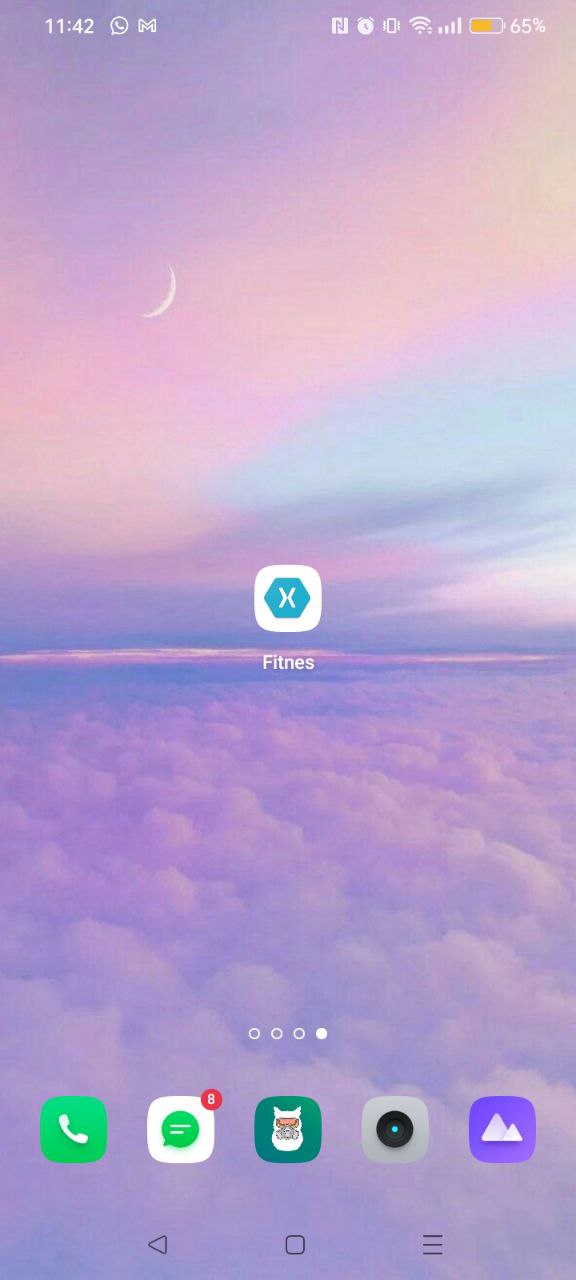


Рисунок 32 – Запуск мобильного приложения

При открытии приложения открывается страница авторизации (рисунок 33). Код авторизации представлен в приложении А. Чтобы получить доступ к функциям необходимо пройти авторизацию. Если аккаунта не существует, пользователь может сам зарегистрироваться как клиент (рисунок 34).

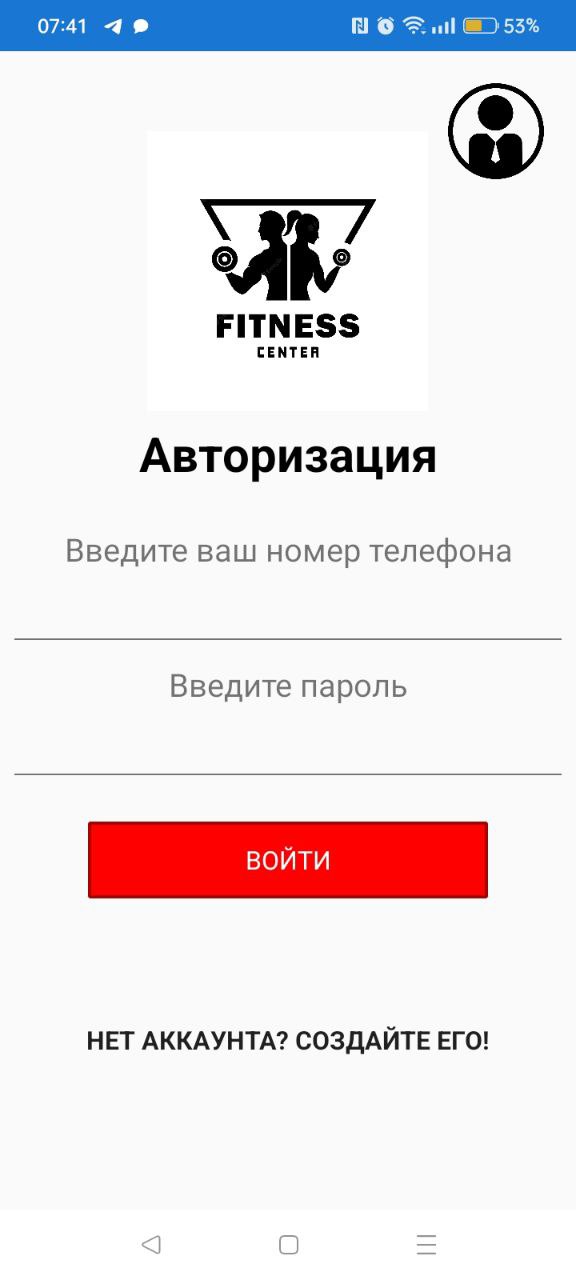


Рисунок 33 – Страница авторизации

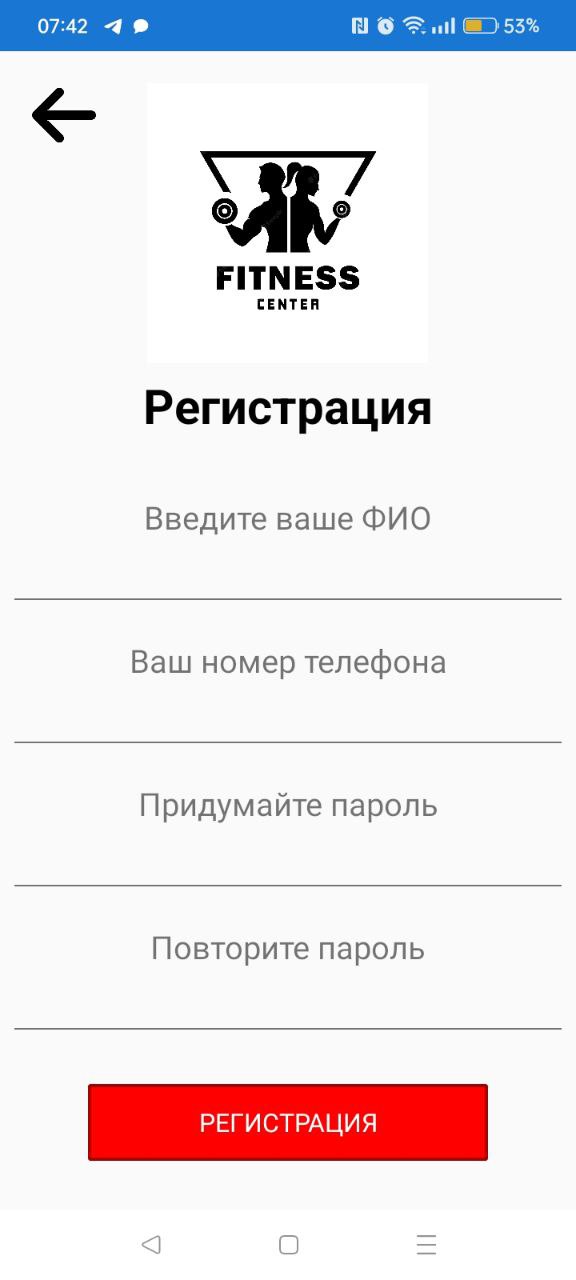


Рисунок 34 – Страница регистрации

После ввода данных авторизации пользователь попадает на страницу профиля (рисунок 35). Если пользователь только зарегистрировался, то данные абонемента будут пустые, но, перейдя, на страницу «Карта» он сможет изменить эти данные и выбрать себе подходящий тариф, как показано на рисунке 36.

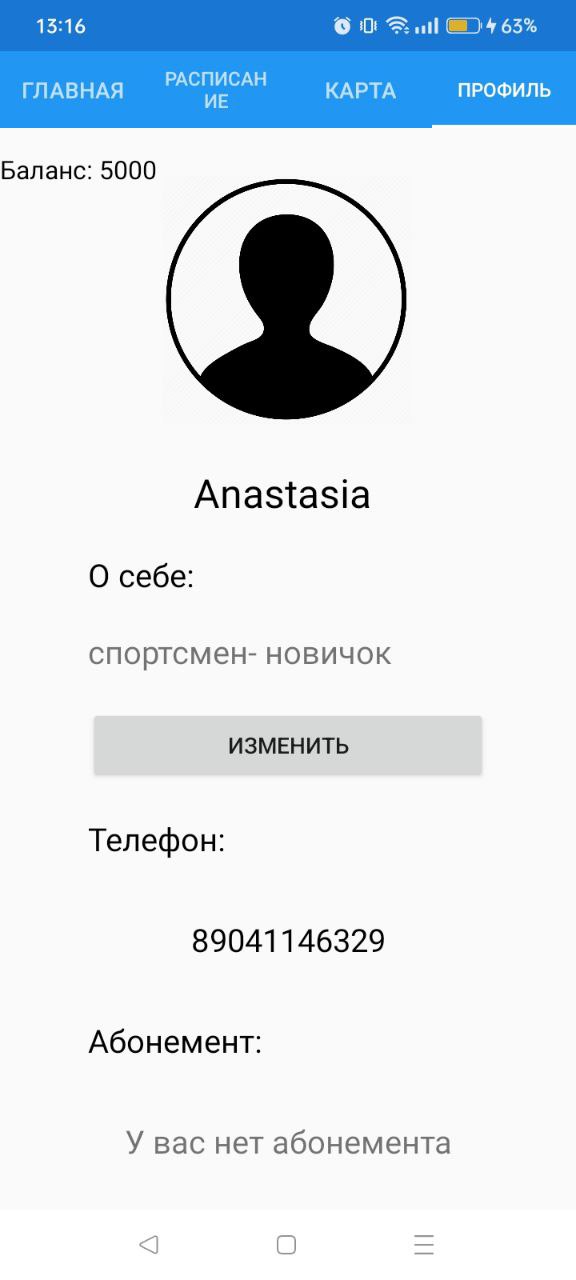


Рисунок 35 – Страница профиля

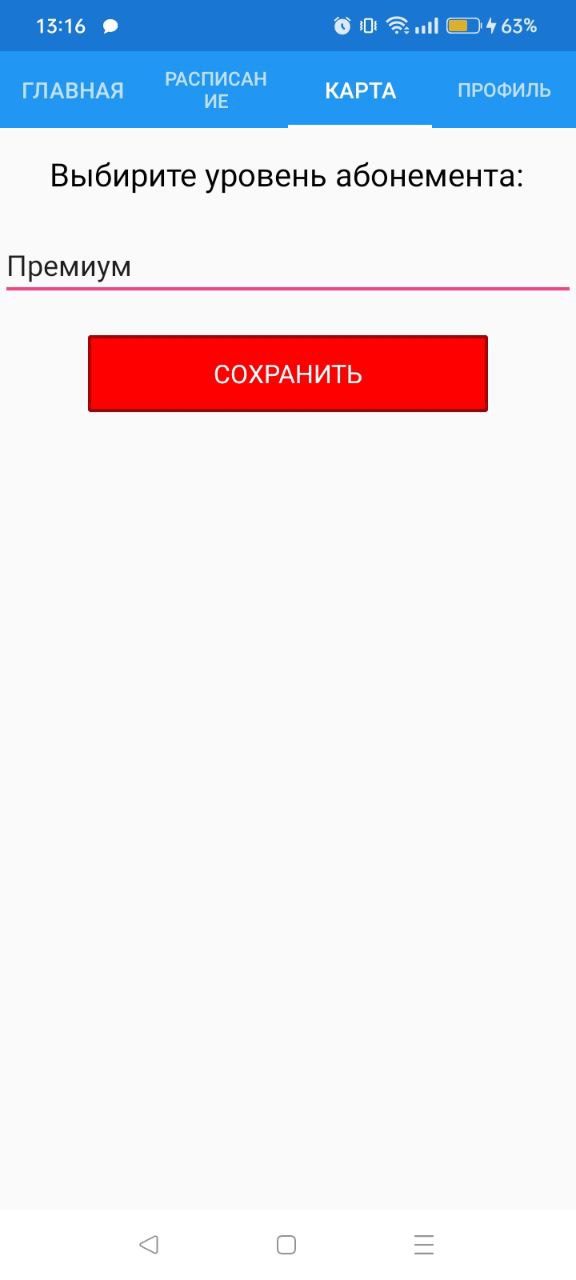


Рисунок 36 – Страница редактирования абонемента

После изменения тарифа у пользователя в профиле изменятся данные, как показано на рисунках 37, 38.

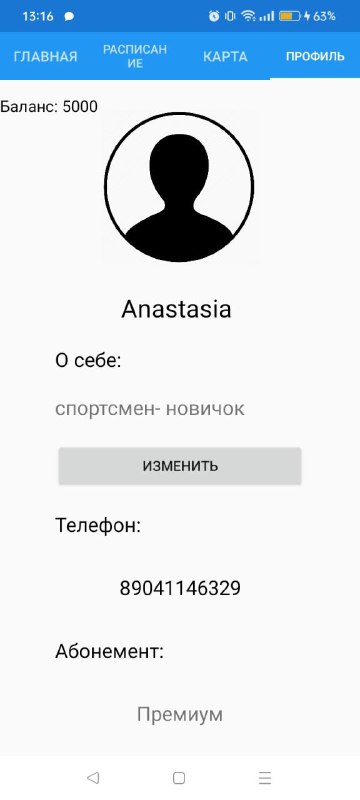


Рисунок 37 – Страница пользователя

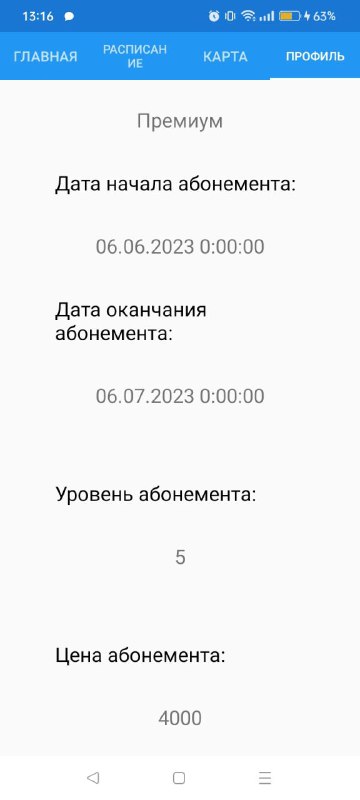


Рисунок 38 – Страница пользователя

Также пользователь может ознакомиться с актуальным расписанием занятий и услуг в фитнес-клубе, перейдя на страницу «Расписание», представленную на рисунке 39.



Рисунок 39 – Страница расписания

**6** **Стоимость разработки и внедрения программного продукта**

* 1. **Организационно-экономическое обоснование проекта**

Разрабатываемое мобильное приложение «Фитнес-клуб» предназначено для эффективного налаживания связи клиента с клубом.

Средняя стоимость разработки приложения составляет (таблица 10).

Таблица 10 – стоимость аналогов мобильного приложения «Фитнес-клуб»

|  |  |
| --- | --- |
| Название сайта | Стоимость разработки аналога, руб. |
| MyFitnessPal | 1 350 000 000 |
| Fitbit | От 7 500 000 |
| Adidas Running | От  3 750 000 |

**6.2 Расчет затрат на разработку программного продукта**

Расчет полных затрат на разработку проектного решения (КРПР) осуществляется по формуле:

КРПР=ЗОТР+ЗЭВМ+ЗСПП+ЗХОН+Е+А, (6.1)

гдеЗОТР – сумма оплаты труда разработчика/разработчиков ПП;

ЗЭВМ – затраты, связанные с эксплуатацией техники;

ЗСПП – затраты на специальные программные продукты, необходимые для разработки ПП;

ЗХОН – затраты на хозяйственно-операционные нужды (бумага, литература, носители информации и т.п.); Е – затраты на электроэнергию, руб.;

А – амортизация ПК, руб.

КРПР = 38 400 + 18 240 + 0 + 0 + 218 + 1 150 = 69 528

Для подсчета фонда оплаты труда разработчика необходимо определить общее время разработки (таблица 9). Время, затрачиваемое на разработку проектного решения j-м разработчиком, определяется методом экспертных оценок или хронометража. Итоговое значение рассчитывается на основании приведенных исходных данных по формуле:



(6.2)

где tβ– время β-го этапа разработки проектного решения, дн.

TРПРj = 2 + 2 + 2 + 1 + 22 + 3 + 7 + 2 = 41

Таблица 11 - Затраты времени на создание программного продукта

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Этап создания | | Затраты времени  (в днях) | Затраты времени  (в часах) | Машинное время работы над ПП  (в часах) |
| Разработка | Обследование объекта автоматизации | 2 | 16 | 16 |
| Анализ и уточнение требований | 2 | 16 | 16 |
| Разработка технического задания | 2 | 16 | 16 |
| Проектирование структуры | 1 | 8 | 8 |
|  | Программная реализация | 22 | 176 | 176 |
| Тестирование программного продукта | 3 | 24 | 24 |
| Отладка программного продукта | 7 | 56 | 56 |
| Разработка описания | 2 | 16 | 16 |
| ИТОГО | 41 | 384 | 384 |

Рабочий день принимается равным 8 часам.

Сумму оплаты труда разработчика за время работы над программным продуктом рассчитаем исходя из часовой тарифной ставки и фонда фактического времени, затраченного на разработку программного продукта (по формуле 6.3):

Зотр=Ст1\*Фвр, (6.3)

где Ст1 - часовой тарифной ставки (принимаем равной 100 руб./час);

Фвр – фонд фактического времени, затраченного на разработку программного продукта, час.

Зотр = 100 ∗ 384 = 38 400

Начисления на заработную плату рассчитываются в таблице 10.

Таблица 12 – Начисление на заработную плату

|  |  |  |
| --- | --- | --- |
| Начисление на заработную плату | Процент,% | Сумма,руб. |
| Пенсионный фонд(ПФ):  -страховая часть  -накопительная часть | 16  6 | 6 144  2 304 |
| Фонд социального страхования(ФСС) | 2,9 | 1 113,6 |
| Федеральный фонд обязательного медицинского страхования(ФФОМС) | 5,1 | 1 958,4 |
| Итого | 30 | 11 520 |

Затраты, связанные с использованием вычислительной и оргтехники (формула 6.4):

*ЗЭВМ* = *ТМРПР* ⋅ 𝑘*Г* ⋅ 𝑛 ⋅ *СМ*−*Ч*, (6.4)

где ТМРПР – машинное время работы над программным продуктом, час; kГ– коэффициент готовности ЭВМ, 𝑘*Г* = 0,95 ; n – количество единиц техники, равно 1;

СМ-Ч – себестоимость машино-часа, СМ-Ч= 50 руб.

ЗЭВМ = 384 ∗ 0,95 ∗ 1 ∗ 50 = 18 240

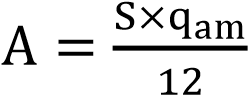
Затраты на электроэнергию рассчитываются по следующей формуле 6.5: Е=W×t×T, (6.5)

где W – мощность, потребляемая ПК, кВт/час; t – время работы ПК, час; T – тариф электроэнергии, руб.

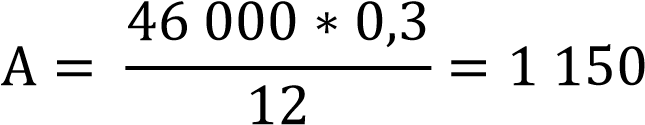
Е = 0,4 ∗ 384 ∗ 1,42 = 218

Тариф 1,42 рубля за киловатт.

Амортизация ПК рассчитывается по следующей формуле 6.6:

, (6.6)

где S– первоначальная стоимость ПК, руб; qam – процент амортизации в год.



Для разработки программного продукта не произведено затрат на специальные программные продукты. Отсюда следует, что ЗСПП = 0.

Поскольку для разработки также не требуются дополнительные вложений на хозяйственно-организационные нужды, то ЗХОН = 0.

Результаты выполненных расчетов сводятся в общей таблице (таблица 13).

Таблица 13 – Затраты на разработку

|  |  |  |
| --- | --- | --- |
| Наименование затрат | Условное обозначение | Значение |
| Оплата труда разработчика программного продукта | ЗОТР | 49 920 |
| Затраты, связанные с эксплуатацией техники | ЗЭВМ | 18 240 |
| Затраты на электроэнергию | Е | 218 |
| Затраты на амортизацию ПК | А | 1 150 |
| Затраты на хозяйственно-операционные нужды | ЗХОН | 0 |
| Затраты на специальные программные продукты | *З*СПП | 0 |
| Итого затрат на разработку | КРПР | 69 528 |

**6.3 Расчет затрат на внедрение программного продукта**

Внедрение модуля в работу предприятия требует оплату хостинга в размере 100 рублей в месяц.

**6.4 Основные выводы**

Суть проекта заключается в разработке и внедрении мобильного приложения для эффективного налаживания связи клиента с клубом. Это позволит пользователям легко и доступно ознакомиться с заведением через мобильное устройство. Главная цель проекта - упростить и ускорить процесс автоматизации оформления абонемента, ознакомление с расписанием и предоставляемыми услугами в фитнес-клубе.

Срок выполнения проекта составляет 41 день.

Итоговая стоимость затрат на разработку программного продукта, в которую включена оплата труда разработчика, затраты на эксплуатацию техники, в том числе амортизации ПК и затраты на электроэнергию, составила 69 528 рублей.

Ожидается, что проект будет окупаться в течение 6 месяцев. Это зависит от многих факторов, включая активность пользователей, тарифной политики и конкурентной среды. Однако, с учетом растущего спроса на услуги фитнес-клубов и возможности масштабирования приложения, ожидается достижение точки окупаемости в относительно короткие сроки.

# ЗАКЛЮЧЕНИЕ

В ходе выполнения дипломного проекта был проведен анализ существующих подобных программ, изучены требования и потребности пользователей, а также проведено проектирование и разработка приложения.

Целью работы было создание удобного и функционального инструмента, способного оптимизировать работу фитнес-клуба, улучшить эффективность процессов управления и повысить удовлетворенность как клиентов, так и сотрудников.

Были реализованы поставленные в качестве цели функциональные возможности.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. it-black.ru ­– Работа с базой данных в C#. – URL: https://it-black.ru/rabota-s-bazoj-dannyh-v-ci\_sharp/ (дата обращения: 20.04.2023). – Текст: электронный.
2. metanit.com – Xamarin и кросс-платформенная разработка. – URL: https://metanit.com/sharp/xamarin/1.1.php (дата обращения: 14.03.2023). – Текст: электронный.
3. metanit.com ­– XAML. – URL: https://metanit.com/sharp/xamarin/2.4.php (дата обращения: 22.04.2023). – Текст: электронный.
4. metanit.com – Базы данных. – URL: https://metanit.com/sharp/xamarin/7.2.php (дата обращения: 25.04.2023). – Текст: электронный.
5. metanit.com – Графический интерфейс в Xamarin Forms. – URL: https://metanit.com/sharp/xamarin/2.3.php (дата обращения: 28.04.2023). – Текст: электронный.
6. metanit.com – Добавление форм. Взаимодействие между формами. – URL: https://metanit.com/sharp/windowsforms/2.3.php (дата обращения: 30.04.2023). – Текст: электронный.
7. metanit.com ­– Контейнер Grid – URL: https://metanit.com/sharp/xamarin/3.6.php (дата обращения: 22.04.2023). – Текст: электронный.
8. metanit.com – Руководство по программированию для Xamarin Forms. – URL: https://metanit.com/sharp/xamarin/ (дата обращения: 30.04.2023). – Текст: электронный.
9. metanit.com ­– Текстовые поля. – URL: https://metanit.com/sharp/xamarin/3.20.php (дата обращения: 01.05.2023). – Текст: электронный.
10. microsoft.com ­– Xamarin.Forms Метки. – URL: https://learn.microsoft.com/ru-ru/xamarin/xamarin-forms/user-interface/text/label (дата обращения: 03.05.2023). – Текст: электронный.
11. microsoft.com – Документация по Xamarin. – URL: https://learn.microsoft.com/ru-ru/xamarin/ (дата обращения: 05.05.2023). – Текст: электронный.
12. microsoft.com – Настройка эмулятора Android. – URL: https://learn.microsoft.com/ru-ru/xamarin/android/get-started/installation/android-emulator/ (дата обращения: 08.05.2023). – Текст: электронный.
13. xamarin.ru ­– Visual Studio Xamarin.Android для Windows. – URL: https://xamarin.ru/knowledge-base/xamarin-android/xamarin-android-dlya-visual-studio-v-windows/ (дата обращения: 10.05.2023). – Текст: электронный.
14. xamarin.ru ­– Создание XAML и его структура. – URL: https://xamarin.ru/knowledge-base/environment/sozdanie-xaml-i-ego-struktura/ (дата обращения: 12.05.2023). – Текст: электронный.
15. xamarin.ru – Элементы управления (UI) Xamarin. – URL: https://xamarin.ru/knowledge-base/tools/ehlementy-upravleniya/ (дата обращения: 28.14.2023). – Текст: электронный.

# Приложение А – Авторизация

using Fitnes.user;

using MySqlConnector;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Linq;

using System.Security.Cryptography;

using System.Text;

using System.Threading.Tasks;

using Fitnes;

using Xamarin.Forms;

namespace Fitnes

{

public partial class AvtorizUser : ContentPage

List<string> userData = new List<string>();

public AvtorizUser()

{

InitializeComponent();

image1.Source = ImageSource.FromResource("Fitnes.image.logo.png");

buttonAdminAvtoriz.Source = ImageSource.FromResource("Fitnes.image.adminLogo.png");

}

private async void OpenReg\_Clicked(object sender, EventArgs e)

{

await Navigation.PushAsync(new AvtorizAdmin());

}

private async void AdminButton\_Clicked(object sender, EventArgs e)

{

await Navigation.PushAsync(new RegUser());

}

private void AvtorizButton\_Clicked(object sender, EventArgs e)

{

string phoneUser = nameInput1.Text;

string passwordUser = nameInput2.Text;

if (phoneUser == "" || passwordUser == "")

{

DisplayAlert("Ошибка", "Заполните все поля", "OK");

}

else

{

DB db = new DB();

db.openConnection();

MySqlCommand command = new MySqlCommand("SELECT \* FROM `klient` WHERE `phoneKlient`=@phoneUser AND `pass\_klient`=@passwordUser", db.getConnection());

command.Parameters.Add("@phoneUser", MySqlDbType.VarChar).Value = phoneUser;

command.Parameters.Add("@passwordUser", MySqlDbType.VarChar).Value = GetHashMD5(passwordUser);

MySqlDataReader reader = command.ExecuteReader();

if (reader.HasRows) // это строки?

{

while (reader.Read()) // забираю данные из бд

{

userData.Add(reader[0].ToString());

userData.Add(reader[1].ToString());

userData.Add(reader[2].ToString());

userData.Add(reader[4].ToString());

}

DisplayAlert("Успех", "Вы авторизировались", "ок");

Navigation.PushAsync(new MainUser(userData));

}

else

{

DisplayAlert("Ошибка", "Не верные данные", "OK");

}

db.closeConnection();

}}

public string GetHashMD5(string input)

{

var md5 = MD5.Create();

var hash = md5.ComputeHash(Encoding.UTF8.GetBytes(input));

return Convert.ToBase64String(hash);}}}

# Приложение Б – Просмотра расписания

using MySqlConnector;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Security.Cryptography;

using System.Text;

using System.Threading.Tasks;

using Fitnes;

using Xamarin.Essentials;

using Xamarin.Forms;

using Xamarin.Forms.Xaml;

namespace Fitnes.user

{

public partial class MainUser : TabbedPage

{

List<string> userData = new List<string>();

bool upAbon = false;

string sql;

int id = 0;

public MainUser(List<string> userData2)

{

InitializeComponent();

imageAvatar.Source = ImageSource.FromResource("Fitnes.image.avatar.png");

userData.Clear();

userData.AddRange(userData2);

Username.Text = userData[2];

Update\_Data();

LoadData();

}

private void Full\_Schedule\_User(object sender, EventArgs e)

{}

private void Sort\_Schedule(object sender, EventArgs e)

{

string sort\_Schedule = nameInput5.Text;

СategoryLable\_User.IsVisible = true;

Сategory\_User.IsVisible = true;}

protected override void OnAppearing()

{

string name = Preferences.Get("name", "не установлено");

informLabel.Text = name;

base.OnAppearing();

informBox.IsVisible = false;

}

private void Update\_Clicked(object sender, EventArgs e)

{

informBox.IsVisible = false;

string value = informBox.Text;

informLabel.Text = value;

Preferences.Set("name", value);

informBox.IsVisible = true;

}

private void Update\_Data()

{

phoneUser.Text = userData[1];

if (userData[3] == "" && upAbon == false)

{

nameAbonement.Text = "У вас нет абонемента";

dateStart.Text = "-";

dateEnd.Text = "-";

levelAbonement.Text = "-";

priceAbonement.Text = "-";}

else

{

DB db = new DB();

db.openConnection();

MySqlCommand command = new MySqlCommand("SELECT `start`,`end`,`nameAbonement`,`abonementLevel`,`abonementPrice` FROM `karta`,`abonement` WHERE `abonement\_idAbonement`=`idAbonement` and `idKarta`=@id", db.getConnection());

command.Parameters.Add("@id", MySqlDbType.Int32).Value = id;

MySqlDataReader reader = command.ExecuteReader();

if (reader.HasRows) // это строки?

{

DisplayAlert("строка есть", "dvvsdvdv", "advasd");

while (reader.Read()) // забираю данные из бд

{

dateStart.Text = reader[0].ToString();

dateEnd.Text = reader[1].ToString();

nameAbonement.Text = reader[2].ToString();

levelAbonement.Text = reader[3].ToString();

priceAbonement.Text = reader[4].ToString();

}}

reader.Close();

db.closeConnection();

}}

private void LoadData()

{

callView.Children.Clear();

NameColumn.Children.Clear();

int j = 0;

DB db = new DB();

db.openConnection();

MySqlCommand command = new MySqlCommand("SELECT uslugi.nameUslugi, raspisanie.Date, raspisanie.Time FROM uslugi JOIN raspisanie ON `idRaspisanie` >= `uslugi\_idUslugi`", db.getConnection());

MySqlDataReader reader = command.ExecuteReader();

if (reader.HasRows)

{

Grid grid2 = new Grid

{

HeightRequest = 45,

RowSpacing = -45,

ColumnSpacing = 4,

BackgroundColor = Color.Black,

RowDefinitions =

{

new RowDefinition { Height = new GridLength(1, GridUnitType.Star) },

},

ColumnDefinitions =

{

new ColumnDefinition { Width = new GridLength(1, GridUnitType.Star) }

}};

for (int i = 0; i < reader.FieldCount; i++)

{

string textcall2 = reader.GetName(i).ToString();

Label label2 = new Label

{

FontSize = 12,

Text = textcall2,

};

label2.VerticalTextAlignment = TextAlignment.Center;

label2.HorizontalTextAlignment = TextAlignment.Center;

label2.TextColor = Color.Black;

label2.BackgroundColor = Color.White;

grid2.Children.Add(label2, i, 0);

}

BoxView box2 = new BoxView

{

Margin = new Thickness(0, -7, 0, 0),

HeightRequest = 4,

};

box2.BackgroundColor = Color.Black;

NameColumn.Children.Add(grid2);

NameColumn.Children.Add(box2);

while (reader.Read())

{

Grid grid = new Grid

{

HeightRequest = 45,

RowSpacing = -45,

ColumnSpacing = 4,

BackgroundColor = Color.Black,

RowDefinitions =

{

new RowDefinition { Height = new GridLength(1, GridUnitType.Star) },},

ColumnDefinitions =

{

new ColumnDefinition { Width = new GridLength(1, GridUnitType.Star) }}};

for (int i = 0; i < reader.FieldCount; i++)

{

string textcall = reader[i].ToString();

if (textcall == "")

{

textcall = "Нет";}

Label label1 = new Label

{

FontSize = 12,

Text = textcall,};

label1.VerticalTextAlignment = TextAlignment.Center;

label1.HorizontalTextAlignment = TextAlignment.Center;

label1.TextColor = Color.Black;

label1.BackgroundColor = Color.LightGray;

grid.Children.Add(label1, i, j);

}

j++;

BoxView box = new BoxView

{

Margin = -5,

HeightRequest = 2

};

box.BackgroundColor = Color.Black;

callView.Children.Add(grid);

callView.Children.Add(box);}}

db.closeConnection();}

}}