

**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук
Образовательная программа «Программная инженерия»

СОГЛАСОВАНО


Научный руководитель,
штатный преподаватель департамента
программной инженерии факультета
компьютерных наук



А.Д. Игнатов
«15» мая 2024 г.

УТВЕРЖДАЮ

Академический руководитель
образовательной программы
«Программная инженерия»,
старший преподаватель департамента
программной инженерии



Н. А. Павлов
«17» мая 2024 г.

Выпускная квалификационная работа

(проектно-исследовательская)

на тему: **Веб-сервис для мониторинга питания пациентов с СМА:
Клиентская часть**

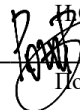
по направлению подготовки 09.03.04 «Программная инженерия»

ВЫПОЛНИЛ

студент группы БПИ205
образовательной программы
09.03.04 «Программная инженерия»

В.А.Роженко

И.О. Фамилия



/14.05.2024

Подпись, Дата

Москва 2024

Реферат

Пациентам с СМА (спинальная мышечная атрофия) очень важен регулярный контроль обеспечения организма необходимыми нутриентами, потому что плохое питание крайне негативно отражается на их состоянии и заболевании. Многим из них требуется более частый мониторинг с корректировкой терапии от врача. На данный момент решения для такого рода взаимодействия отсутствуют. Данный проект предназначен для оптимизации взаимодействия пациента с СМА и врачом-диетологом за счет оперативной связи и своевременной коррекции терапии.

Целью выпускной квалификационной работы является реализация клиентской части веб-сервиса для мониторинга питания пациентов с СМА. Интерфейс приложения подразделяется на три основные части: интерфейс для врача, интерфейс для пациента, интерфейс для администратора. Каждая из них удовлетворяет поставленным требованиям.

В данном документе представлено описание предметной области, сравнение аналогов разрабатываемого решения, устройство и функционал веб-сервиса, а также используемые технологии.

Данная работа содержит 53 страницы, 3 главы, 2 таблицы, 32 рисунка, 16 источников, 6 приложений.

***Ключевые слова:* веб-сервис, flutter, СМА, доктор, пациент, клиентская часть.**

Abstract

Regular monitoring of the body's supply of essential nutrients is very important for patients with SMA (spinal muscular atrophy) because poor nutrition has an extremely negative effect on their condition and disease. Many of them require more frequent monitoring with therapy adjustments from their doctor. Currently, there are no solutions for this kind of interaction. This project is designed to optimize the SMA patient's interaction and the dietitian through operational communication and timely correction of therapy.

The aim of the graduate qualification work is to implement the client side of the web-service for nutrition monitoring of patients with SMA. The application interface is divided into three main parts: the interface for the doctor, the interface for the patient, and the interface for the administrator. Each of them satisfies the set requirements.

This article provides a description of the subject area, a comparison of analogues of the solution being developed, the structure and functionality of the web-service, as well as the technologies used.

This article contains 53 pages, 3 chapters, 2 tables, 32 images, 16 sources, 6 appendices.

Keywords: web-service, flutter, SMA, doctor, patient, client side.

Оглавление

Реферат.....	1
Abstract.....	2
Основные определения, термины и сокращения.....	5
Введение	6
Глава 1. Предметная область и анализ существующих решений.....	8
1.1 Описание предметной области	8
1.2 Обзор существующих аналогов	8
1.2.1 Обзор сервисов для медицинских учреждений	8
1.2.1.1 Medesk	8
1.2.1.2 Яндекс.Здоровье	9
1.2.1.3 DocDoc (СберЗдоровье).....	9
1.2.1.4 Smart Медицина.....	9
1.2.1.5 Выводы.....	10
1.2.2 Обзор мобильных дневников симптомов	10
1.2.2.1 Apple Health	10
1.2.2.2 Betalife	10
1.2.2.3 Bowelle	10
1.2.2.4 Здоровье.ру	11
1.2.2.5 Выводы.....	11
1.3 Сравнительные таблицы.....	11
1.4 Конечные пользователи.....	13
1.5 Вывод по главе	14
Глава 2. Используемые технологии и архитектура.....	15
2.1. Описание архитектуры разрабатываемой системы.....	15
2.2. Описание паттерна MVC.....	15
2.2.1. Model	16
2.2.2. View	17
2.2.3. Controller	21
2.3. Взаимодействие с серверной частью.....	22
2.4. Прототип интерфейса	24
2.3. Обоснование выбранных для разработки технологий.....	25
2.4. Вывод по главе	25
Глава 3. Функции и особенности приложения	26
3.1. Список основных функций веб-приложения.....	26
3.2. Общие функции.....	30
3.2.1. Экран авторизации	30

3.3. Функционал врача	31
3.3.1. Экран со списком пациентов.....	31
3.3.2. Просмотр данных пациента.....	33
3.3.3. Просмотр статистики	34
3.3.4. Просмотр последних записей пациента	35
3.3.5. Просмотр данных пациенту рекомендаций	35
3.3.6. Просмотр назначенных модулей	36
3.3.7. Назначение модуля для заполнения пациенту	37
3.3.8. Написание рекомендации пациенту	37
3.3.9. Просмотр уведомлений.....	38
3.3.10. Редактирование информации	39
3.4. Функционал пациента.....	39
3.4.1. Экран с данными	39
3.4.2. Просмотр статистики	40
3.4.3. Просмотр последних записей.....	41
3.4.4. Добавление записи	41
3.4.5. Просмотр уведомлений.....	43
3.4.6. Просмотр рекомендаций.....	44
3.4.7. Редактирование информации	45
3.5. Функционал администратора.....	46
3.5.1. Просмотр списка докторов.....	46
3.5.2. Поиск докторов по фамилии	46
3.5.3. Просмотр списка пациентов.....	47
3.5.4. Поиск пациентов по фамилии	48
3.5.5. Просмотр профиля пользователя.....	48
3.5.6. Редактирование информации пользователя.....	48
3.5.7. Регистрация новых пользователей	49
3.5.8. Создание пары врач-пациент	49
3.7. Вывод по главе	50
Заключение.....	51
Список использованных источников.....	52

Основные определения, термины и сокращения

1. **Спинально-мышечная атрофия (СМА)** — разнородная группа наследственных заболеваний, протекающих с поражением / потерей двигательных нейронов передних рогов спинного мозга.
2. **Антропометрические показатели** — основные физические параметры тела: рост, вес, объемы. Измерение данных показателей проводится с целью выявления особенностей физического строения.
3. **Нутриенты** — это биологически значимые элементы пищи. Макронутриенты входят в рацион человека в существенных количествах (углеводы, белки, жиры — а также вода), микронутриенты — в минимальных (витамины, минералы, флавоноиды и прочие).
4. **Нутритивная поддержка** — процесс обеспечения организма полноценным питанием с помощью ряда методов, отличных от обычного приема пищи. Этот процесс включает в себя дополнительное энтеральное питание через рот и/или зонд, а также частичное или полное парентеральное питание.
5. **Клиентская часть приложения** — интерфейс с набором функций, с которым взаимодействует пользователь. Он отправляет запросы серверной части приложения, принимает ответы и обрабатывает их.
6. **Dart** — высокоуровневый язык программирования, разработанный компанией Google. Считается языком общего назначения, ориентированным в первую очередь на фронтенд: создание интерфейсов и взаимодействие с браузером.
7. **Flutter** — фреймворк для создания нативных интерфейсов для мобильных, веб- и настольных приложений из единой кодовой базы с использованием языка программирования Dart.
8. **Фреймворк** — обширная структура, предоставляющая набор инструментов, библиотек и правил для разработки приложений.
9. **Модуль для заполнения** — список параметров, назначаемый доктором пациенту для заполнения с указанной частотой, с целью отслеживания показателей здоровья пациента. Например, модуль антропометрии включает в себя такие показатели, как вес, рост, окружность плеча.

Введение

Для поддержания адекватного функционирования клеток и тканей, восстановления организма при болезни, для роста и развития необходимо эффективное обеспечение питательными веществами. Пациентам с СМА (спинальная мышечная атрофия) очень важен регулярный контроль обеспечения организма необходимыми нутриентами, потому что плохое питание крайне негативно отражается на их состоянии и заболевании. Такие пациенты регулярно наблюдаются у профильных специалистов, однако части из них требуется более частый мониторинг ряда параметров с получением обратной связи от врача о необходимости скорректировать питание. В настоящее время специализированные платформы для такого взаимодействия отсутствуют.

Целью работы является разработка сервиса, который позволит построить эффективное взаимодействие между врачом и пациентом с СМА. После очной консультации пациента врач будет выбирать параметры в приложении, необходимые к регулярному заполнению пациентом. Пациенты будут периодически обновлять информацию о параметрах и диете. Сервис будет рассчитывать потребление пациентом калорий, белков, жиров и углеводов, конвертируя введенную пациентом в приложение употребленную еду. Алгоритм сортировки будет ранжировать состояние пациентов в порядке приоритетности оказания помощи на основе проверки соответствия реальных параметров ожидаемым. Врач сможет отслеживать состояния пациентов в своем личном кабинете, также ему будут высылаться уведомления о срочных пациентах. Врач будет иметь возможность отправить ответ пациенту, который последний сможет увидеть в своем личном кабинете, также уведомление об ответе врача будет высылаться пациенту. Различные вводимые пациентом медицинские показатели можно будет посмотреть в приложении и отследить их динамику.

Наличие приложения для такой коммуникации с предварительной обработкой информации, полученной от пациента, позволит обеспечить своевременной медицинской помощью на амбулаторном этапе лечения большее количество пациентов - при отсутствии существенного прироста нагрузки на врачей.

Для достижения цели выпускной квалификационной работы были поставлены следующие задачи:

1. Сбор требований;
2. Изучение медицинских тем, связанных с СМА;
3. Анализ существующих на рынке аналогов;
4. Выбор моделей сервиса;
5. Выбор подходящих инструментов для разработки;
6. Прототипирование интерфейса;
7. Разработка клиентской части веб-сервиса;
8. Тестирование клиентской части;
9. Налаживание клиент-серверного взаимодействия;
10. Разработка технической документации.

Далее в работе описываются: предметная область, сравнительный анализ представленных на рынке аналогов, архитектура системы, инструменты использованные в процессе разработки и основные её части интерфейса – для врача, для пациента, для администратора.

Глава 1. Предметная область и анализ существующих решений

1.1 Описание предметной области

Предметной областью создаваемого приложения является реализация приложения для взаимодействия пациента с врачом и обработки медицинских данных. Веб-сервис предоставляет функционал для врачей, для пациентов и для администраторов, позволяет регулярно отслеживать состояние пациента, оперативнее получать рекомендации по корректировке терапии от врача, а врачу эффективнее оказывать помощь благодаря сортировке пациентов и подсветки отклонений показателей.

Для врачей предусмотрены возможности просмотра списка пациентов (всех и отсортированный по срочности проблемы), поиска пациентов по фамилии, назначения пациенту модулей для заполнения (для регулярного отслеживания состояния пациента), просмотр данных пациента (заполненные пациентом данные, подсветка записей с отклонениями от нормы, статистика), обратная связь с пациентом. Для пациентов предусмотрено заполнение назначенных модулей, просмотр своих записей, статистика и связь с врачом. Для администратора – регистрация и добавление новых пользователей, настраивание связи врач-пациент.

Данная работа направлена упрощение коммуникации между врачом и пациентом, улучшение эффективности и скорости врачебной помощи благодаря автоматизированной сортировке пациентов и формату взаимодействия.

1.2 Обзор существующих аналогов

Аналоги разрабатываемого сервиса подразделяются на две группы: различные сервисы для медицинских учреждений и мобильные дневники симптомов. Поскольку рынок аналогов довольно обширный, в анализе представлены одни из самых популярных решений, которые включают большинство доступного функционала.

1.2.1 Обзор сервисов для медицинских учреждений

1.2.1.1 Medesk

Данный сервис является облачной платформой для управления медицинской клиникой. Позволяет отслеживать статусы приемов, вести медицинские карты пациентов, проводить онлайн консультации, создавать аналитику работы клиники, создавать расписание приемов, создавать отчеты и т.д.

Функционал Medesk[6] направлен в большей степени на врачей и администраторов, для пациентов доступна только онлайн запись и онлайн консультации, нет возможности отслеживать статистику своих показателей и заполнять информацию о здоровье. Кроме того, отсутствует автоматический анализ состояния пациента и сортировка пациентов.

1.2.1.2 Яндекс.Здоровье

Яндекс.Здоровье[11] является коммерческим сервисом онлайн-консультаций с врачами. Общение с врачом происходит в чате или по видеосвязи. Стоимость использования зависит от выбранного пользователем пакета консультаций. После консультации пользователю предоставляется отчет, в котором содержится общая информация, рекомендации по лечению и рекомендация специалиста для очного приема. Есть мобильные приложения для iOS и Android.

Недостатками данного сервиса являются отсутствие возможности просмотреть статистику своих данных и отсутствие привязки пациентов к врачам. Данный сервис ориентирован на предварительную консультацию с врачом перед очным визитом, он предоставляет пользователям возможность посоветоваться со специалистом или получить ответ на интересующий вопрос.

1.2.1.3 DocDoc (СберЗдоровье)

Это российская компания, которая предоставляет телемедицинские услуги (онлайн приемы к врачу), запись на процедуру в клинику, запись к врачу, запись на диагностику, возможность вести дневник здоровья в мобильном приложении.

В то же время у данного сервиса так же нет возможности сортировки пациентов, автоматической оценки состояния и просмотра статистики медкарты.

1.2.1.4 Smart Медицина

Сервис Smart Медицина [7] является медицинской информационной системой для клиники, которая позволяет вести запись пациентов к врачу, руководить работой клиники, вести материальный учет расходов, пациенту предоставляют личный кабинет, где он может просматривать свои данные, так же есть возможность подключения функции анализа лечения пациента для проверки, как происходит лечение согласно карте пациента.

Тем не менее у данной системы существует ряд недостатков. Первым из них является отсутствие визуализации данных пациента для оценки динамики изменения различных показателей здоровья. Другим недостатком является то, что сервис представлен в виде десктопного приложения, что предполагает предварительную установку.

1.2.1.5 Выводы

Наиболее распространенными недостатками описанных систем являются: платное распространение, отсутствие таких функций как просмотр статистики медкарты, сортировка пациентов по срочности и важности проблемы и автоматический анализ состояния пациента.

1.2.2 Обзор мобильных дневников симптомов

1.2.2.1 Apple Health

Это приложение на iOS, позволяющее комплексно следить за здоровьем. Оно интегрируется с другими приложениями здоровья и фитнеса на телефоне или аксессуарах и анализирует показатели пользователя. Позволяет отслеживать активность, физические данные, сон, потребленные продукты и записывать различные симптомы, смотреть статистику изменений показателей.

Несмотря на обширный выбор параметров для отслеживания, в приложении отсутствует автоматический анализ состояния здоровья и нет какой-либо возможности взаимодействия с врачами.

1.2.2.2 Betalife

Мобильное приложение Betalife [2] является наиболее схожим с разрабатываемым решением сервисом из представленных на рынке. Оно представляет из себя сервис для людей с рассеянным склерозом, где они могут вести дневник симптомов, следить за диетой, вносить лекарства, наблюдать за связью симптомов и лечения и найти ближайшую больницу, но это приложение нацелено на людей заболеванием другого рода.

Его недостатком является отсутствие интерфейса для врача, с возможностью постоянного наблюдения за ходом лечения пациента и отображения отсортированного по срочности проблемы списка пациентов.

1.2.2.3 Bowelle

Bowelle[3] - это дневник настроения, питания и симптомов специально для людей страдающих синдромом раздраженного кишечника (СРК) и другими проблемами связанными с пищеварительной системой. В приложении можно отслеживать настроение, симптомы, указывать приемы пищи и уровень стресса. Есть возможность построить графики настроения.

Недостатком является отсутствие возможности просмотра статистики по данным помимо настроя, отсутствие возможности ввода медицинских данных для отслеживания и автоматического анализа данных пользователя.

1.2.2.4 Здоровье.ру

Приложения для контроля заболеваний и здоровья, есть множество программ для различных заболеваний. Программы включают в себя различные упражнения, рекомендации, дневники для отслеживания симптомов, иногда бесплатные консультации. Также в приложении можно записаться к врачу, пройти тесты для самоконтроля и вести дневник лекарств.

Минусом данного приложения является отсутствие визуализации данных и автоматического анализа состояния пользователя.

1.2.2.5 Выводы

Основными недостатками приложений дневников симптомов являются: отсутствие автоматического анализа состояния пользователя и отсутствие взаимодействия с врачом.

1.3 Сравнительные таблицы

В процессе анализа были выделены следующие пункты для сравнения:

1. Общие:
 - a. Наличие веб-версии;
 - b. Модель распространения;
 - c. Монетизация;
 - d. Визуализация данных;
 - e. Хранение медкарты пациента;
 - f. Просмотр статистики медкарты;
 - g. Возможность заполнения данных пациенту;
 - h. Возможность взаимодействия врача с пациентом;
 - i. Автоматический анализ состояния пациента.
2. Для приложений для мед. учреждений и телемедицины – сортировка пациентов по степени срочности и важности проблемы.
3. Для приложений дневников симптомов и отслеживания показателей – возможность следить за питанием.

Таблица 1 - Анализ аналогов среди приложений для мед. учреждений и телемедицины

Критерий	Medesk [6]	Яндекс.Здоровье [11]	DocDoc [5]	SmartМедицина [7]	Разрабатываемое решение
Веб-версия	1	1	1	1	1
Модель распространения	платная	платная	платная	платная	бесплатная
Монетизация	подписки	пакеты консультаций	пакеты и подписки	тарифы	нет
Визуализация данных	1	1	1	0	1
Хранение медкарты пациента	1	1	0	1	1
Просмотр статистики медкарты	0	0	0	0	1
Сортировка пациентов по срочности и важности проблемы	0	0	0	0	1
Возможность заполнения данных пациенту	0	1	1	1	1
Возможность взаимодействия врача с пациентом	0	1	1	0	1
Автоматический анализ состояния пациента	0	0	0	1	1

Таблица 2 - Анализ аналогов среди приложений дневников симптомов и отслеживания показателей

Критерий	Apple Health[1]	BetaLife[2]	Bowelle[3]	Здоровье.ру[8]	Разрабатываемое решение
Веб-версия	0	0	0	0	1
Модель распространения	бесплатная	бесплатная	бесплатная	бесплатная	бесплатная
Монетизация	нет	нет	подписки	подписки	нет
Визуализация данных	1	1	1	0	1
Хранение медкарты пациента	1	1	0	1	1
Просмотр статистики медкарты	1	1	0	0	1
Возможность следить за питанием	1	1	1	1	1
Возможность заполнения данных пациенту	1	1	1	1	1
Возможность взаимодействия врача с пациентом	0	1	0,5 (отправить данные)	1	1
Автоматический анализ состояния пациента	0	0,5 (тесты)	0	0,5 (тесты)	1

1.4 Конечные пользователи

Конечными пользователями являются медицинское учреждение, ведущее пациентов, страдающих СМА, доктора, пациенты с СМА и их помощники. В дальнейшем возможно масштабирование системы и распространение ее на взаимодействие с другими пациентами и врачами.

1.5 Вывод по главе

В данной главе была описана предметная область и представлены аналоги разрабатываемого решения, а также проведен их сравнительный анализ. Из анализа аналогов можно сделать вывод, что разрабатываемое решение является единственным для предоставления врачам и пациентам с СМА необходимого функционала.

Глава 2. Используемые технологии и архитектура

В данной главе описана архитектура веб-приложения, в частности описан паттерн MVC, выбранный для разработки, а также представлено обоснование выбора используемых технологий.

2.1. Описание архитектуры разрабатываемой системы

Для реализации данного проекта была выбрана клиент-серверная архитектура с веб-сервисом. Клиентская часть отвечает за интерфейс приложения, взаимодействие с пользователем, получение и отображение данных от серверной части. Серверная часть, реализованная другим участником команды, отвечает за обработку, хранение и предоставление данных клиентской части. Взаимодействие между частями происходит через REST API.



Рис. 1 - Архитектура веб-сервиса

2.2. Описание паттерна MVC

Для реализации клиентской части веб-сервиса был выбран паттерн MVC (Model-View-Controller), который наиболее подходит для решения поставленной задачи и часто применяется в аналогичных проектах. Он позволяет разделить логику приложения на слои, что упрощает дальнейшее развитие и изменение кода, благодаря разным предметам ответственности частей приложения.

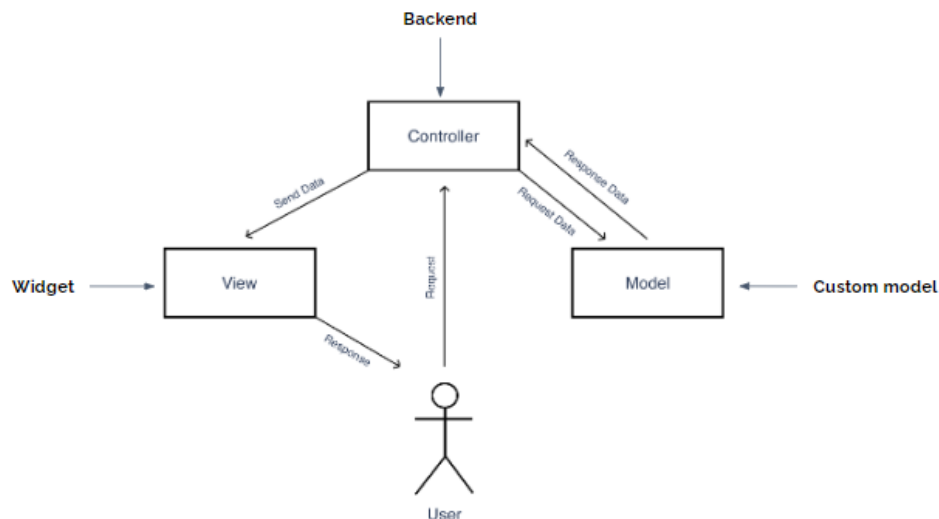


Рис. 2 - Паттерн MVC, используемый в данном приложении

2.2.1. Model

Модель представляет собой класс определенной модели данных (например, модель пользователя, модель модуля и т.д.). Такой класс содержит поля, хранящие всю необходимую информацию о модели. В данном случае все модели находятся в папке «models» и имеют названия аналогичные «name_model.dart». К моделям относятся сущности пользователя, модуля, параметра, уведомления, рекомендации доктора и т.д.

Пример – модель пользователя:

```

class UserModel {
  UserModel({
    required this.id,
    required this.username,
    required this.firstname,
    required this.lastname,
    required this.fathername,
    required this.email,
    required this.password,
    required this.roleId,
    required this.birthday,
    required this.sex,
  });

  String id = "";
  String username;
  String firstname;
  String lastname;
  String email;
  String birthday;
  String sex;
  String password;
}
  
```

```

String fathername;
String roleId;

factory UserModel.fromJson(Map<String, dynamic> json) {
  return UserModel(
    id: json["id"],
    username: json["username"],
    firstname: json["firstname"],
    lastname: json["lastname"],
    fathername: json["fathername"],
    email: json["email"],
    password: json["passwordHash"],
    roleId: json["roleId"],
    birthday: json["birthday"],
    sex: json["sex"],
  );
}

Map<String, String> toJson() => {
  "username": username,
  "firstname": firstname,
  "lastname": lastname,
  "fathername": fathername,
  "email": email,
  "password": password,
  "roleId": roleId,
  "birthday": birthday,
  "sex": sex,
};

Map<String, String> getData() => {
  "имя": firstname,
  "фамилия": lastname,
  "отчество": fathername,
  "почта": email,
  "дата рождения": birthday,
  "пол": sex,
};
}

```

2.2.2. View

Представление отвечает за отображение данных пользователю на экране. В данном случае из-за специфики используемого языка представление состоит из всех виджетов, которые составляют интерфейс отдельной страницы. Таким образом классы представления содержат один виджет, который состоит из дерева других отдельных виджетов. Помимо различных страниц, в отдельный класс представления еще включаются виджеты, которые используются на нескольких страницах. Например, `AppBar` – верхняя строка на всех экранах приложения. Она разная для трех разных пользователей – врача, пациента, администратора – и содержит в себе кнопки для навигации по основным страницам приложения, как пример у пациента там находятся кнопки «Данные», «Уведомления», «Рекомендации». Все

представления (помимо общих) разделены на три папки в соответствии с типом пользователя.

Пример представления – страница авторизации:

```
class Login extends StatefulWidget {
  const Login({Key? key}) : super(key: key);

  @override
  State<Login> createState() => _LoginState();
}

class _LoginState extends State<Login> {
  Map userData = {};
  String login = '';
  String password = '';

  final _formkey = GlobalKey<FormState>();

  @override
  void initState() {
    super.initState();
    globals.loggedIn = false;
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        automaticallyImplyLeading: false,
        backgroundColor: firstColor,
        title: const Text('Вход'),
        centerTitle: true,
      ),
      body: SingleChildScrollView(
        child: Column(
          children: <Widget>[
            Padding(
              padding: const EdgeInsets.only(top: 30.0),
              child: Center(
                child: Container(
                  width: 120,
                  height: 120,
                  decoration: BoxDecoration(
                    borderRadius: BorderRadius.circular(40),
                    border: Border.all(color: Colors.blueGrey)),
                child: Image.asset('lib/ui/common/logos/logo2.png'),
              ),
            ),
            Padding(
              padding: const EdgeInsets.symmetric(horizontal: 15),
              child: Padding(
                padding: const EdgeInsets.all(12.0),
                child: Form(
                  key: _formkey,
                  child: Column(
```

```

crossAxisAlignment:CrossAxisAlignment.start,
children: <Widget>[
  Padding(
    padding: const EdgeInsets.all(12.0),
    child: TextFormField(
      onChanged: (value) => login=value,
      cursorColor: secondColor,
      decoration: const InputDecoration(
        focusedBorder: OutlineInputBorder(
          borderSide: BorderSide(
            color: secondColor,
            width: 2.0,
          ),
borderRadius: BorderRadius.all(Radius.circular(10)),
        ),
        hintText: 'Username',
        labelText: 'Username',
        prefixIcon: Icon(
          Icons.email,
          color: secondColor,
        ),
        hintStyle: TextStyle(
          fontSize: 16.0, color: secondColor),
        labelStyle: TextStyle(
          fontSize: 16.0, color: secondColor),
        errorStyle: TextStyle(fontSize: 18.0),
        border: OutlineInputBorder(
          borderSide:
            BorderSide(color: secondColor),
          borderRadius:
            BorderRadius.all(Radius.circular(9.0))),
      )),
  Padding(
    padding: const EdgeInsets.all(12.0),
    child: TextFormField(
      obscureText: true,
      onChanged: (value) => password =value,
      cursorColor: secondColor,
      decoration: const InputDecoration(
        focusedBorder: OutlineInputBorder(
          borderSide: BorderSide(
            color: secondColor,
            width: 2.0,
          ),
borderRadius:
          BorderRadius.all(Radius.circular(10)),
        ),
        hintText: 'Password',
        labelText: 'Password',
        prefixIcon: Icon(
          Icons.key,
          color: secondColor,
        ),
        hintStyle: TextStyle(
          fontSize: 16.0, color: secondColor),
        labelStyle: TextStyle(
          fontSize: 16.0, color: secondColor),

```

```

        errorStyle: TextStyle(fontSize:18.0),
        border: OutlineInputBorder(
            borderSide: BorderSide(color: Colors.red),
            borderRadius: BorderRadius.all(Radius.circular(9.0))),
    ),
),
Center(
    child: Padding(
        padding: const EdgeInsets.all(28.0),
        child: Container(
            child: RaisedButton(
                child: const Text(
                    'Войти',
                    style: TextStyle(
                        color: Colors.white, fontSize:22),
                ),
                onPressed: () {
                    ApiDataProvider().login(login, password);
                    Future.delayed(const Duration(seconds: 1),
                        () {
                            if (!globals.loggedIn)
                                {ScaffoldMessenger.of(context).showSnackBar(const SnackBar( content:
                                    Text('Проверьте корректность введенных значений!'),
                                        backgroundColor: Colors.red,
                                    ));
                                } else {
                                    if (globals.user!.roleId ==
                                        globals.doctorRoleId) {
                                        Navigator.of(context).push(MaterialPageRoute(builder: (context) =>
                                            ((const DesktopPatientListPageDoctor()))));
                                    } else {
                                        if (globals.user!.roleId ==
                                            globals.patientRoleId)
                                            {Navigator.of(context).push(MaterialPageRoute(builder: (context) =>
                                                ((const DesktopMainPatientPagePatient()))));
                                        } else {
                                            Navigator.of(context).push(MaterialPageRoute(builder: (context) =>
                                                ((const DesktopDoctorsListPageAdmin()))));
                                        }
                                    }
                                }
                        }
                    ));
                },
                shape: RoundedRectangleBorder(
                    borderRadius:
                        BorderRadius.circular(30)),
                    color: firstColor,
                ),
                width:
                    MediaQuery.of(context).size.width / 3,
                height: 50,
            ),
        ),
    ),
),

```

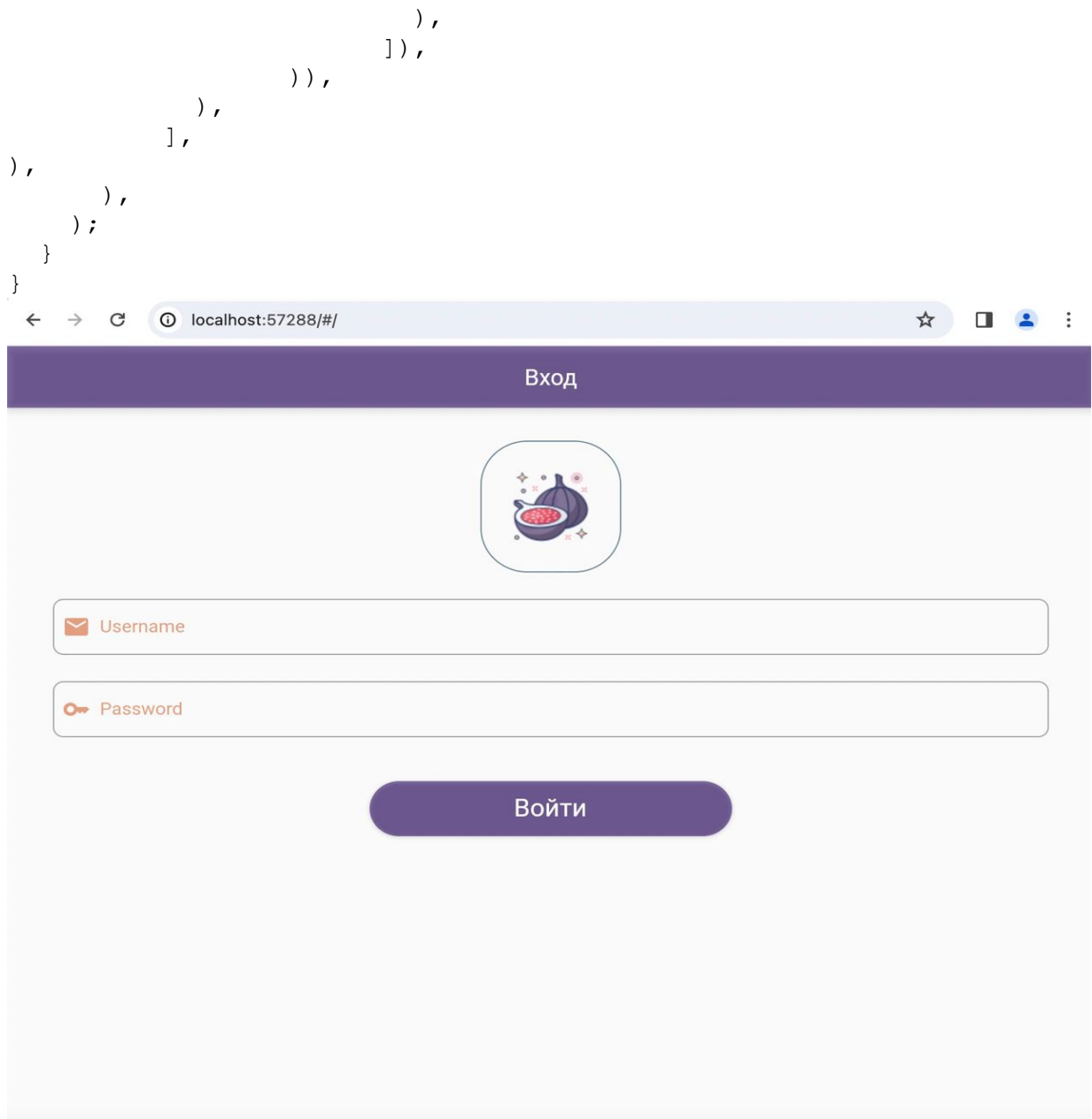


Рис.3 - Страница авторизации

2.2.3. Controller

Контроллер отвечает за взаимодействие клиентской и серверной частей. Взаимодействие происходит по следующему принципу: контроллер отправляет запрос на сервер, получает от сервера необходимые данные, записывает данные в модели, а затем представление их отображает.

Пример контроллера для авторизации:

```
Future<UserModel?> login(String login, String password) async {
```

```

try {
    Map<String, String> data = {
        "\"username\"": "\"$login\"",
        "\"password\"": "\"$password\"",
    };
    var url = Uri.parse(ApiConstants.baseUrl +
ApiConstants.loginEndpoint);
    var response = await http.post(url,
        headers: <String, String>{
            'Content-Type': 'application/json; charset=UTF-8',
        },
        body: data.toString());
    print(data.toString());
    log(response.statusCode.toString() + "\n");
    print(response.body);
    if (response.statusCode == 200) {
        String source = utf8.decode(response.bodyBytes);
        UserModel _model = userModelFromJson(source);
        globals.user = _model;
        print(_model.id.toString());
        print(globals.user?.id.toString());
        globals.loggedIn = true;
        return _model;
    } else {
        globals.user = null;
        return null;
    }
} catch (e) {
    log(e.toString());
    globals.user = null;
    return null;
}
}

```

2.3. Взаимодействие с серверной частью

Взаимодействие между частями программы происходит через REST API по протоколу HTTP, данные передаются в формате JSON. Ниже приведен список используемых запросов:

1. POST /user - регистрация пользователя
2. PUT /user - изменение пользователя
3. GET /user/{id} - получение данных пользователя
4. DELETE /user/{id} – удаление пользователя
5. GET /user/find/{lastname}/{roleId} - получение пользователей по фамилии и роли
6. POST /login - авторизация
7. POST /moduleFillIn - добавление введенных пользователем параметров модуля
8. GET /moduleFillIn/{patientId}/{doctorId} - получение заполненных модулей по пациенту и доктору

9. GET /moduleFillIn/{patientId} - получение заполненных модулей по пациенту
10. GET /moduleParameters/{questionaryId} - получение заполненных параметров модуля врачом
11. GET /module/{patientId}/{doctorId} - получение модулей по пациенту и доктору
12. GET /module/gastroLabel/{parameterId} - получение лейблов для выбора ответа гастро-параметра
13. GET /module/{patientId} - получение модулей по пациенту
14. POST /questionary - назначение модуля для заполнения пациенту
15. PUT /questionary/{questionaryId}/{frequency} - изменение частоты заполнения модуля
16. DELETE /questionary/{id} - удаление модуля у пользователя
17. GET /recommendation/{patientId}/{doctorId} - получение всех рекомендаций по пациенту и врачу
18. POST /recommendation – добавление рекомендации пациенту
19. GET /recommendation/{patientId} - получение всех рекомендаций пациента
20. GET /doctorsList - получение всех докторов
21. GET /patientsList - получение всех пациентов
22. GET /patientsList/{doctorId} - получение всех пациентов доктора
23. GET /doctorsList/{patientId} - получение всех докторов пациента
24. POST /linkPatient/{patientId}/{doctorId} - прикрепление пациента к врачу
25. GET /parameters - получение всех параметров в модуль врачом
26. GET /notifications/{userId} - получение всех уведомлений пользователя
27. GET /statistics/{moduleId}/{patientId} - получение всех данных для графиков по параметрам модуля за все периоды
28. GET /formula - получение названий всех смесей для питания
29. GET /products - получение всех продуктов в отсортированном виде
30. GET /products/{productGroupId} - получение всех продуктов по группе
31. GET /doctorParameters/{moduleId} - получение всех параметров для выбора доктором при назначении модулей для пациента для конкретного модуля
32. GET /doctorParametersLabels/{parameterId} - получение параметров для выбора ответа доктора для конкретного параметра модуля, выбранного доктором
33. GET /patientsHierarchy - получение отсортированного списка пациентов

Пример JSON:

```
{
  "username": "Bobbo",
  "password": "1234"
}
```

2.4. Прототип интерфейса

В разработку клиентской части веб-сервиса входила разработка прототипа интерфейса, для решения этой задачи использовался сервис Figma [16].

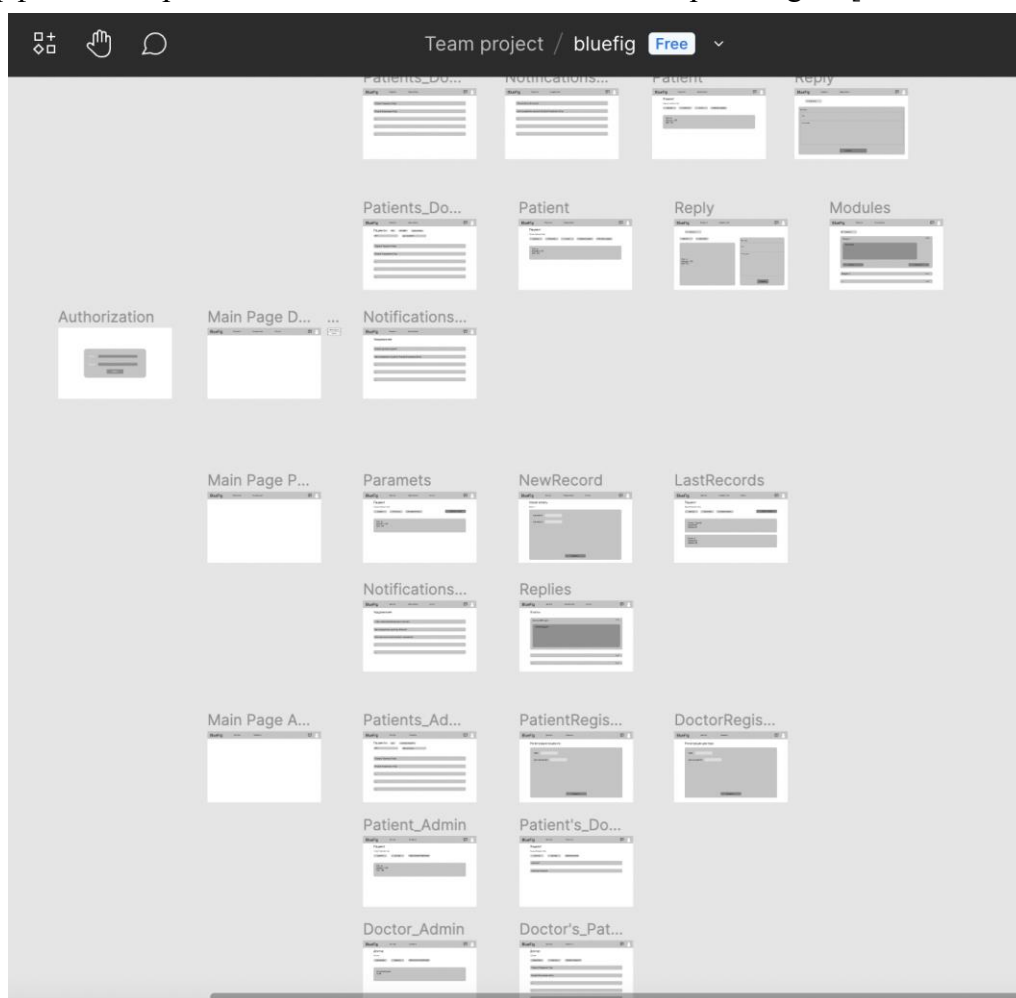


Рис.4 - Прототип интерфейса

В дальнейшем были выбраны цвета интерфейса на основе цветов сайта фонда «Семьи СМА» [10]. Эти цвета хранятся в файле с библиотекой «app_colors.dart».

Интерфейс проектировался, обсуждая детали с консультантом – Юлией Александровой Алымовой – практикующим врачом в сфере детского питания, специализирующимся на работе с пациентами с СМА.

2.5. Обоснование выбранных для разработки технологий

Для разработки веб-сервиса был выбран язык Dart [4] и фреймворк Flutter [9]. Одной из будущих целей развития проекта является создание мобильной версии приложения. Благодаря использованию языка Dart и фреймворка Flutter эта задача не потребует разработки мобильного интерфейса с нуля, кроме того, обеспечит кроссплатформенность решения, то есть не нужно будет разрабатывать две отдельные версии под iOS и Android.

При разработке использовались следующие технологии:

1. Dart (2.16.2) [4] – язык программирования
2. Flutter (2.10.5) [9]– фреймворк
3. http (0.13.5) [12] – библиотека для http запросов
4. multiselect (0.0.4) [13] – библиотека, содержащая виджет для множественного выбора
5. syncfusion_flutter_charts (18.4.49) [14] – библиотека для построения графиков

При разработке использовались следующие средства:

1. Visual Studio Code [15]
2. Figma [16]

2.6. Вывод по главе

Выше были рассмотрены архитектура веб-сервиса в целом и архитектура клиентской части, взаимодействие частей программы, прототип дизайна приложения, использованные технологии и средства.

Глава 3. Функции и особенности приложения

В этой главе подробно описывается функционал веб-сервиса, в частности – разделение на функционал врача, пациента и администратора.

3.1. Список основных функций веб-приложения

1. Отображение стартовой страницы приложения;
2. Отображение навигации на страницах;
3. Отображение страницы авторизации пользователя;
4. Выход из системы;
5. Для врача:
 - 5.1. Отображение полного списка пациентов;
 - 5.2. Отображение отсортированного по степени срочности списка пациентов;
 - 5.3. Поиск пациентов по фамилии;
 - 5.4. Просмотр данных пациента:
 - 5.4.1. Имя;
 - 5.4.2. Фамилия;
 - 5.4.3. Отчество;
 - 5.4.4. Почта;
 - 5.4.5. Дата рождения;
 - 5.4.6. Пол;
 - 5.5. Просмотр статистики в виде графиков численных показателей пациента;
 - 5.6. Просмотр последних записей пациента с выделением красным модулей с отклонениями показателей здоровья. Последняя запись включает в себя:
 - 5.6.1. Дата и время добавления записи;
 - 5.6.2. Название заполненного модуля;
 - 5.6.3. Введенные параметры;
 - 5.7. Просмотр данных рекомендаций. Рекомендация включает в себя:
 - 5.7.1. Дата и время добавления;
 - 5.7.2. ФИО доктора;
 - 5.7.3. Текст рекомендации;

- 5.8. Просмотр назначенных модулей. Назначенный модуль включает в себя:
 - 5.8.1. Название модуля;
 - 5.8.2. Параметры модуля;
 - 5.8.3. Назначенная частота заполнения;
- 5.9. Редактирование частоты заполнения назначенного модуля;
- 5.10. Назначение модуля для заполнения пациенту с выбором необходимых условий (частота, способ обработки, вид обратной связи и т.д.). Модуль включает в себя:
 - 5.10.1. Название модуля;
 - 5.10.2. Параметры модуля;
 - 5.10.3. Выбор дополнительных условий назначения модуля (зависят от самого модуля);
 - 5.10.4. Выбор частота заполнения модуля;
- 5.11. Написание и отправка рекомендации пациенту;
- 5.12. Просмотр уведомлений (новый пациент, новый срочный пациент и т.д.).
Уведомление включает в себя:
 - 5.12.1. Дата и время;
 - 5.12.2. Текст уведомления;
- 5.13. Редактирование персональных данных:
 - 5.13.1. Юзернейм;
 - 5.13.2. Имя;
 - 5.13.3. Фамилия;
 - 5.13.4. Отчество;
 - 5.13.5. Почта;
 - 5.13.6. Дата рождения;
 - 5.13.7. Пол;
 - 5.13.8. Пароль.

6. Для пациента:

- 6.1. Просмотр своих данных:
 - 6.1.1. Имя;
 - 6.1.2. Фамилия;
 - 6.1.3. Отчество;
 - 6.1.4. Почта;

- 6.1.5. Дата рождения;
- 6.1.6. Пол;
- 6.2. Просмотр статистики в виде графиков численных показателей пациента;
- 6.3. Просмотр последних записей. Последняя запись включает в себя:
 - 6.3.1. Дата и время добавления записи;
 - 6.3.2. Название заполненного модуля;
 - 6.3.3. Введенные параметры;
- 6.4. Добавление новых записей (заполнение модулей, заполнение дневника питания). Заполнение модуля включает в себя заполнение параметров модуля. Заполнение дневника питания включает в себя составление списка продуктов с указанием названия продукта и граммов.
- 6.5. Просмотр рекомендаций врача. Рекомендация включает в себя:
 - 6.5.1. Дата и время добавления;
 - 6.5.2. ФИО доктора;
 - 6.5.3. Текст рекомендации;
- 6.6. Просмотр уведомлений (назначен новый модуль, новая рекомендация и т.д.). Уведомление включает в себя:
 - 6.6.1. Дата и время;
 - 6.6.2. Текст уведомления;
- 6.7. Редактирование персональных данных:
 - 6.7.1. Юзернейм;
 - 6.7.2. Имя;
 - 6.7.3. Фамилия;
 - 6.7.4. Отчество;
 - 6.7.5. Почта;
 - 6.7.6. Дата рождения;
 - 6.7.7. Пол;
 - 6.7.8. Пароль.

7. Для администратора:

- 7.1. Просмотр списка пациентов;
- 7.2. Поиск пациентов по фамилии;
- 7.3. Просмотр списка врачей;
- 7.4. Поиск врачей по фамилии;

7.5. Просмотр профиля пользователя:

- 7.5.1. Имя;
- 7.5.2. Фамилия;
- 7.5.3. Отчество;
- 7.5.4. Почта;
- 7.5.5. Дата рождения;
- 7.5.6. Пол;

7.6. Редактирование информации о пользователе:

- 7.6.1. Юзернейм;
- 7.6.2. Имя;
- 7.6.3. Фамилия;
- 7.6.4. Отчество;
- 7.6.5. Почта;
- 7.6.6. Дата рождения;
- 7.6.7. Пол;

7.7. Регистрация новых пользователей. Указываются следующие данные:

- 7.7.1. Юзернейм;
- 7.7.2. Имя;
- 7.7.3. Фамилия;
- 7.7.4. Отчество;
- 7.7.5. Почта;
- 7.7.6. Дата рождения;
- 7.7.7. Пол;
- 7.7.8. Пароль;

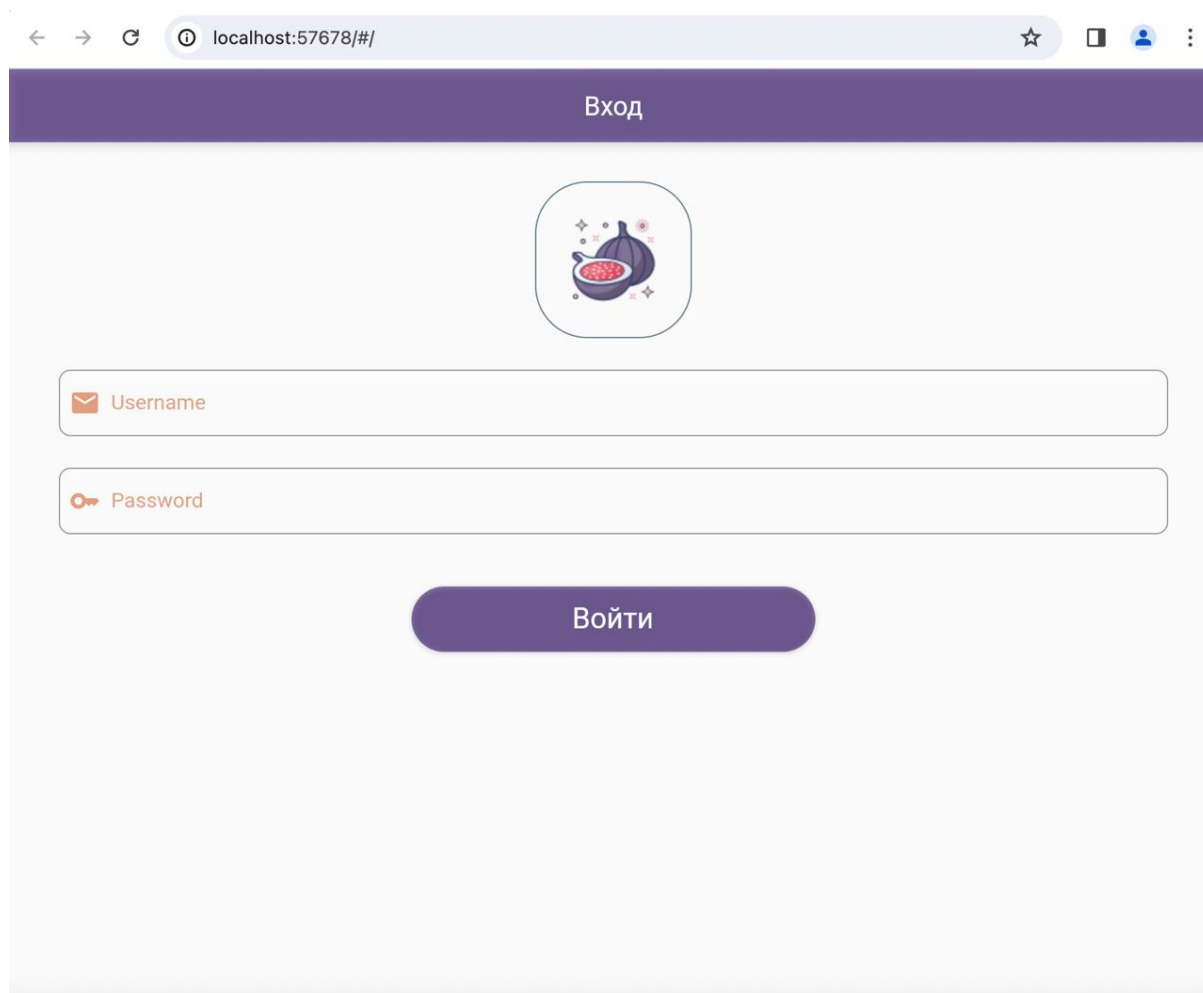
7.8. Удаление пользователя;

7.9. Создание пары врач-пациент.

3.2. Общие функции

3.2.1. Экран авторизации

При входе в приложение пользователь попадает на экран авторизации. На экране находятся логотип, поле для ввода юзернейма, поле для ввода пароля. При неудачной авторизации всплывает сообщение об ошибке.



The screenshot shows a web browser window with the address bar displaying 'localhost:57678/#/'. The page has a purple header with the word 'Вход'. Below the header is a logo featuring a purple bowl with red caviar and a purple onion. There are two input fields: 'Username' with an envelope icon and 'Password' with a key icon. At the bottom is a purple button labeled 'Войти'.

Рис.5 - Страница авторизации

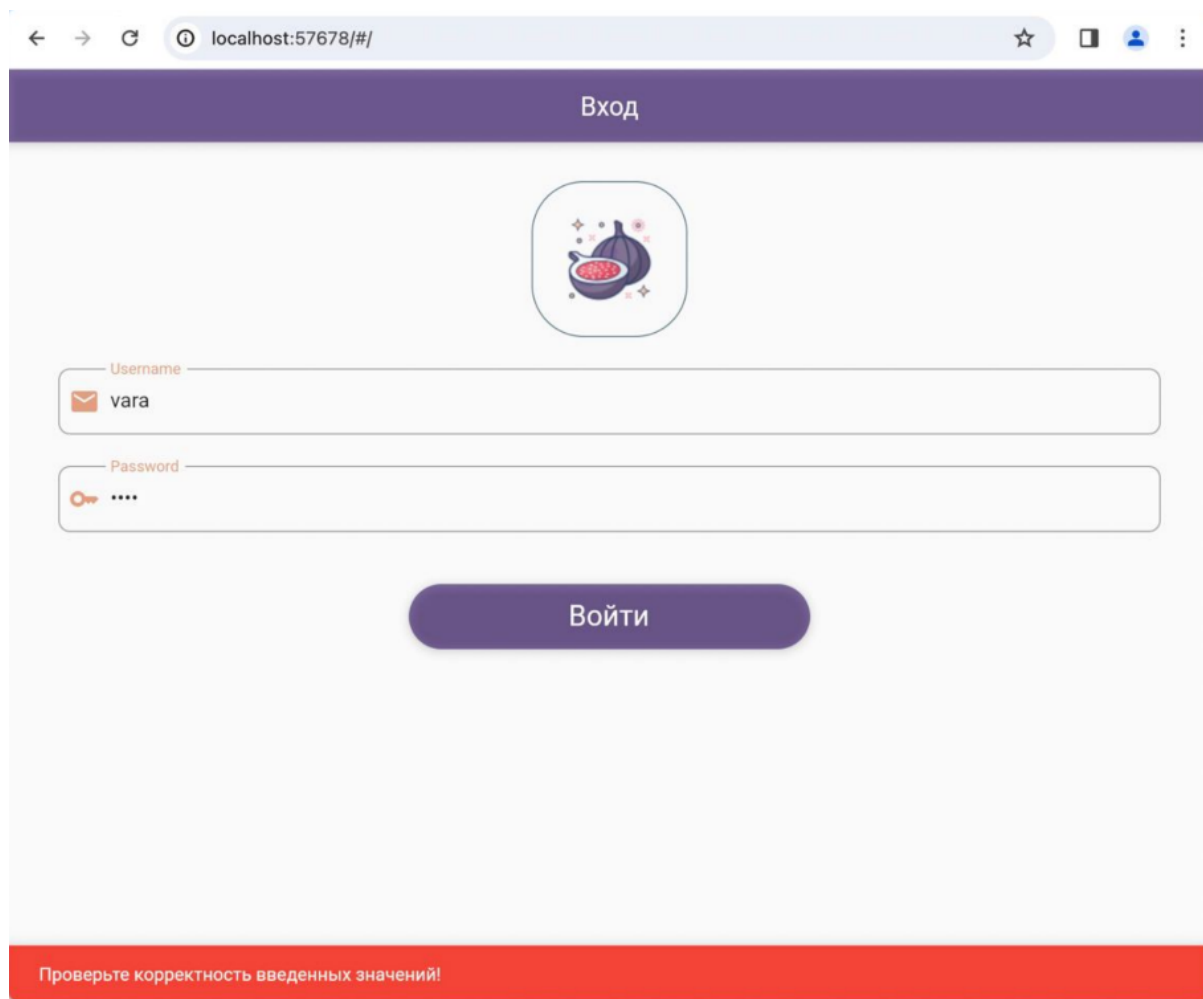


Рис.6 - Страница авторизации, сообщение об ошибке

3.3. Функционал врача

3.3.1. Экран со списком пациентов

После корректной авторизации, врач попадает на страницу с полным списком пациентов. На всех страницах сверху располагается строка навигации, у врача на ней находятся название приложения и кнопки «Пациенты», «Уведомления», «Профиль», «Выход из системы».

На странице со списком пациентов отображается три кнопки для выбора отображаемых пациентов: «Все пациенты», «Срочные», «Поиск». При выборе «Все пациенты» отображается полный список пациентов, прикрепленных к доктору. При выборе «Срочные» отображается список пациентов, отсортированных по степени срочности и важности (сверху самые срочные). При выборе «Поиск», отображается поле для ввода фамилии и кнопка поиска, после ввода фамилии и клика на кнопку поиска отображаются пациенты с заданной фамилией.

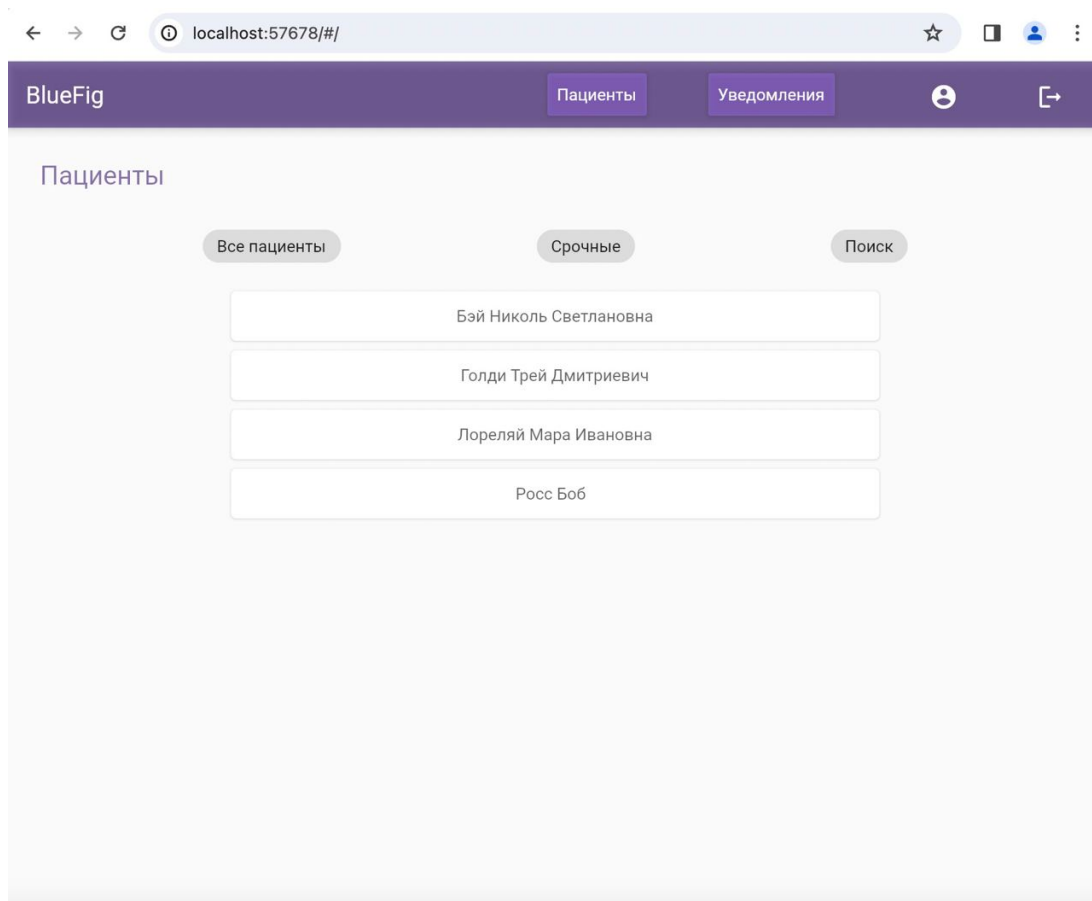


Рис.7 - Страница с полным списком пациентов

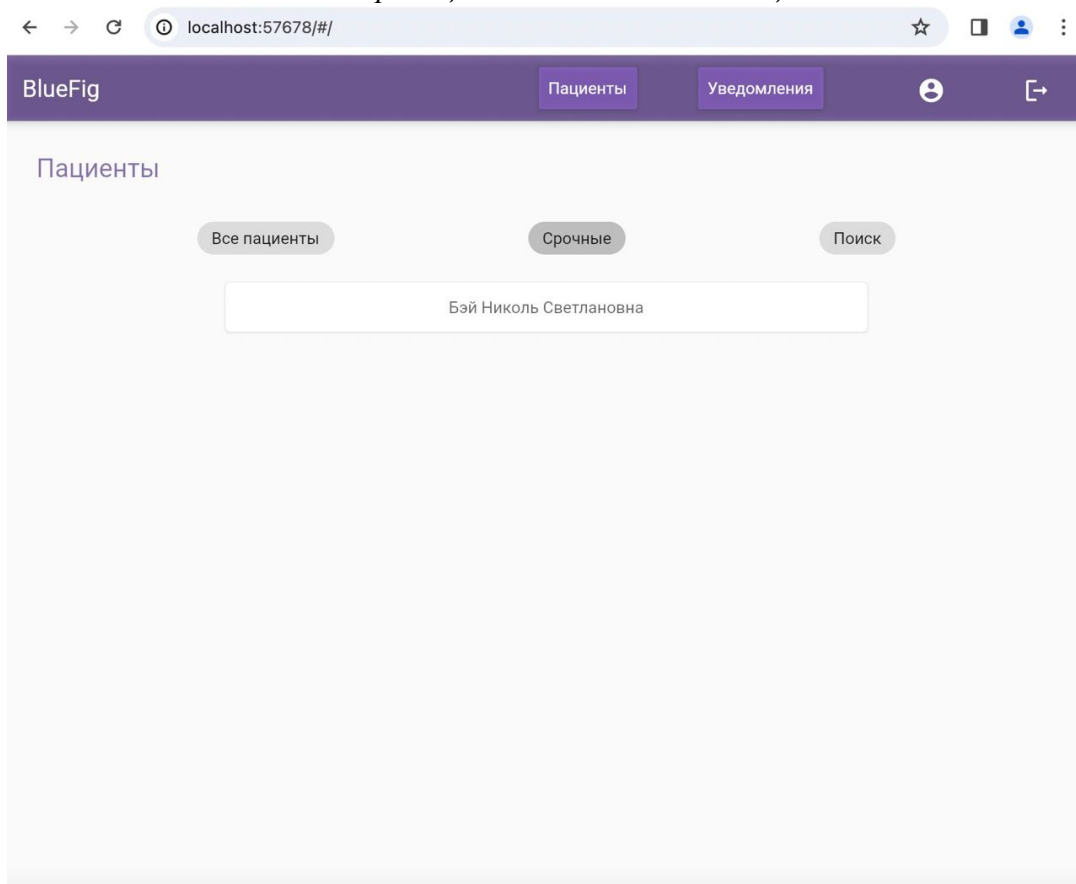


Рис.8 - Страница со списком срочных пациентов

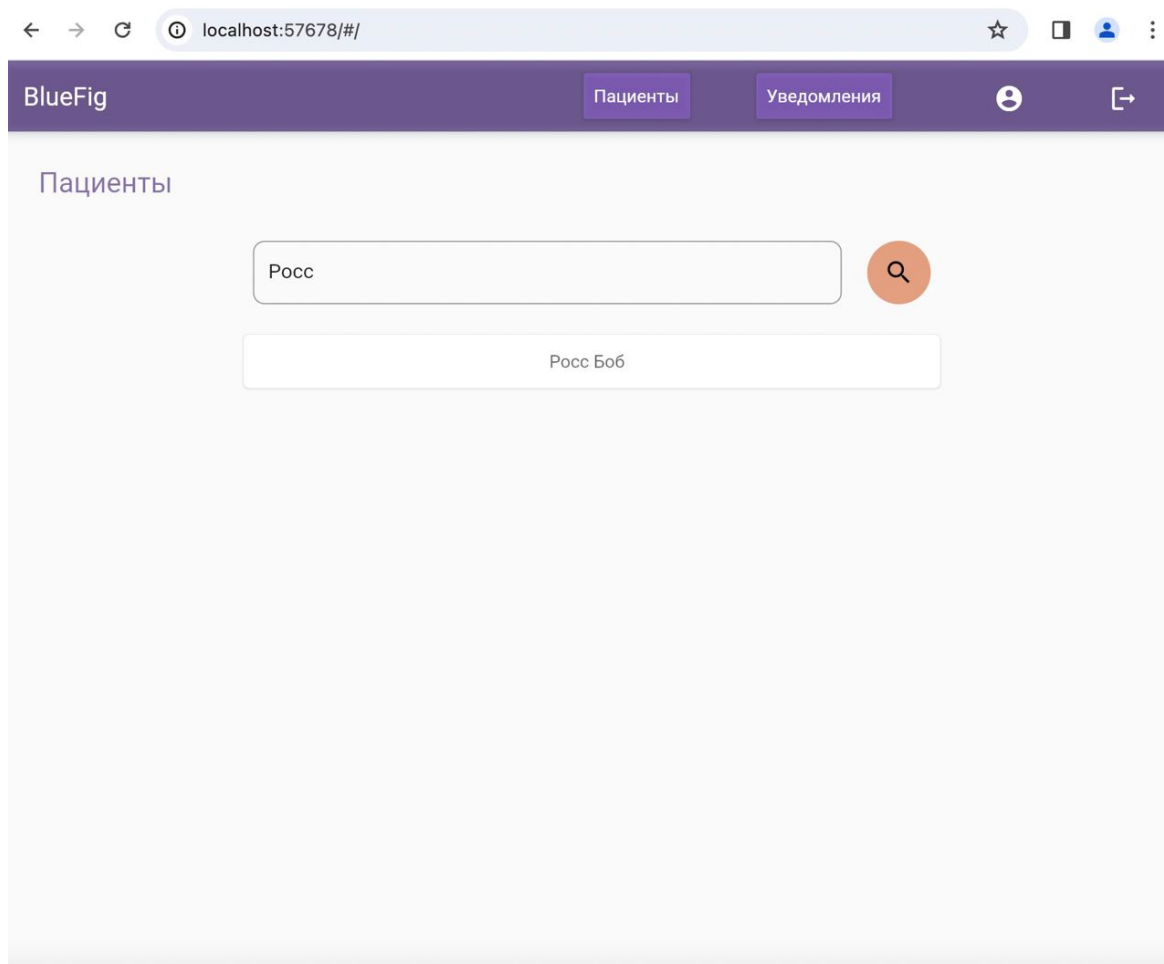


Рис.9 - Страница с поиском пациентов

3.3.2. Просмотр данных пациента

При просмотре списка пациентов врач может кликнуть на одного из них и перейти на страницу с данными пациента. На странице содержатся кнопки выбора различного взаимодействия с пациентом: «Данные», «Статистика», «Последние записи», «Рекомендации», «Назначенные модули», «Назначить модуль», «Дать рекомендацию».

На вкладке «Данные» отображается информация о пациенте: имя, фамилия, отчество, почта, дата рождения и пол.

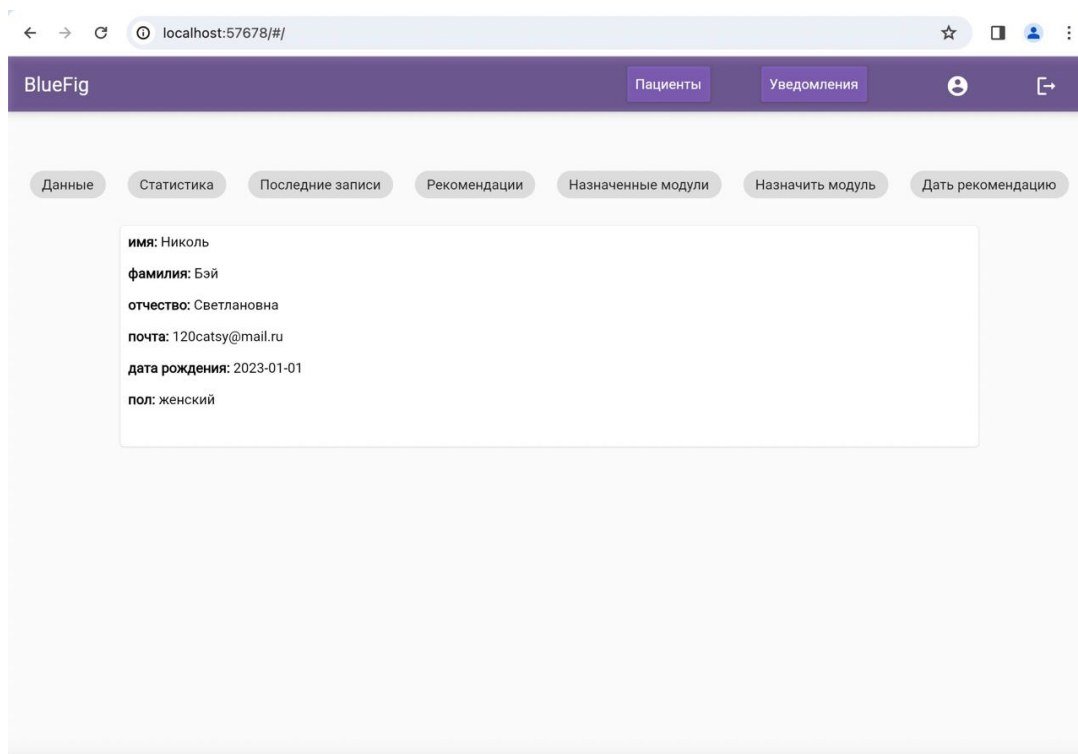


Рис.10 - Страница с данными пациента

3.3.3. Просмотр статистики

При переходе на вкладку «Статистика» отображаются графики численных показателей здоровья пациента: рост, вес, окружность плеча. Вертикальная шкала – значение, горизонтальная – дата. С помощью графиков врач может наглядно отследить динамику изменения показателей.

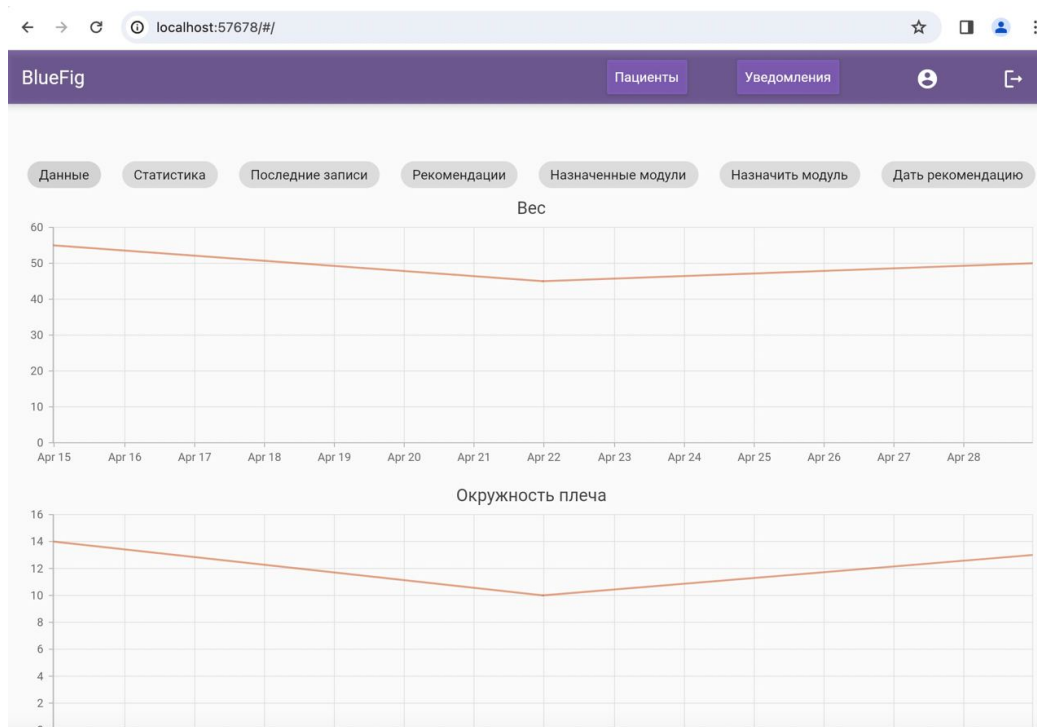


Рис.11 - Страница со статистикой

3.3.4. Просмотр последних записей пациента

При переходе на вкладку «Последние записи» отображается список с последними записями пациента. Красным цветом выделяются записи с отклонениями от нормы, это сделано для того, чтобы врачу не приходилось искать место, где что-то не так стало с пациентом. Элемент записи содержит информацию о дате и времени записи, названии модуля записи и параметров с введенным пациентом значениями. Каждая из записей изначально отображается в свернутом виде, то есть показывается только дата добавления и название модуля, при клике на запись, она раскрывается и показывается полная информация с заполненными параметрами.

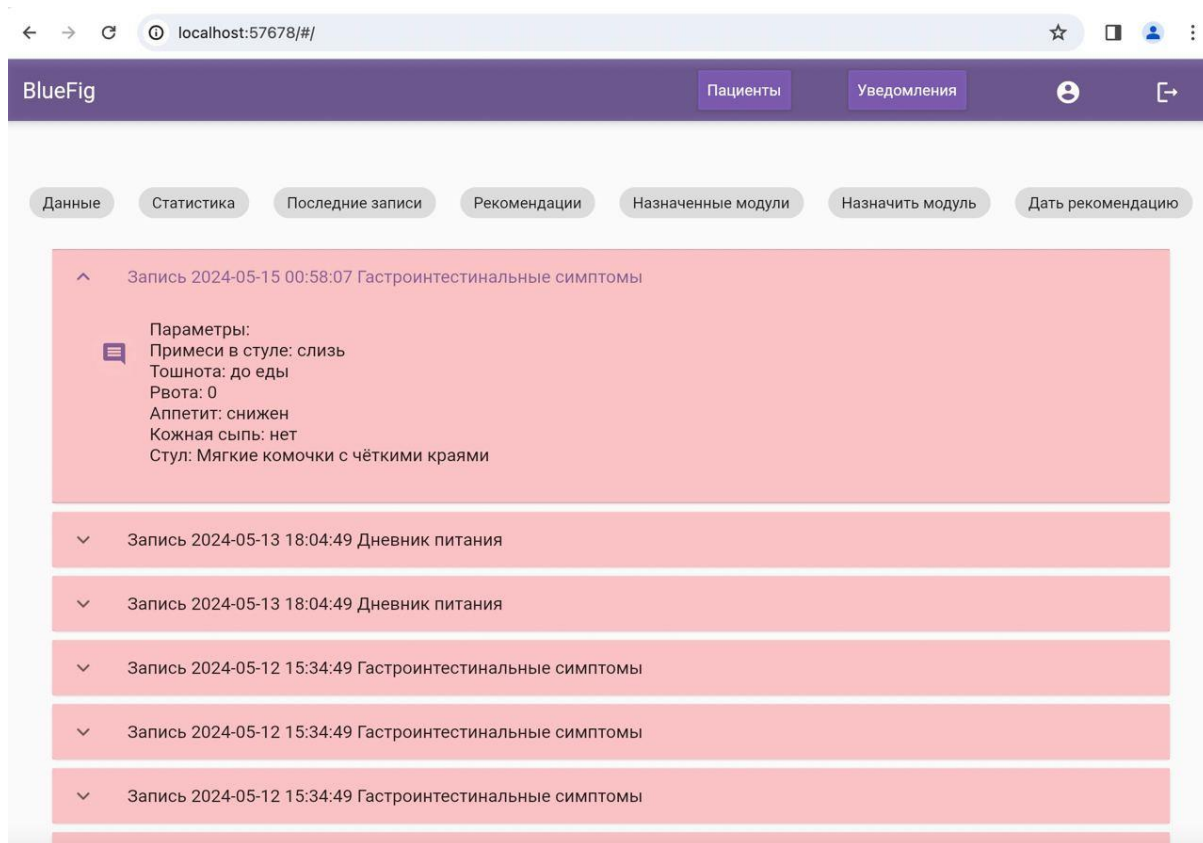


Рис.12 - Страница с последними записями пациента

3.3.5. Просмотр данных пациенту рекомендаций

При переходе на вкладку «Рекомендации» отображается список с данными пациенту рекомендациями. Элемент рекомендации содержит дату и время добавления, ФИО врача и текст самой рекомендации. Список рекомендаций первоначально отображается со свернутыми параметрами, однако их аналогично последним записям можно развернуть для просмотра текста рекомендации.

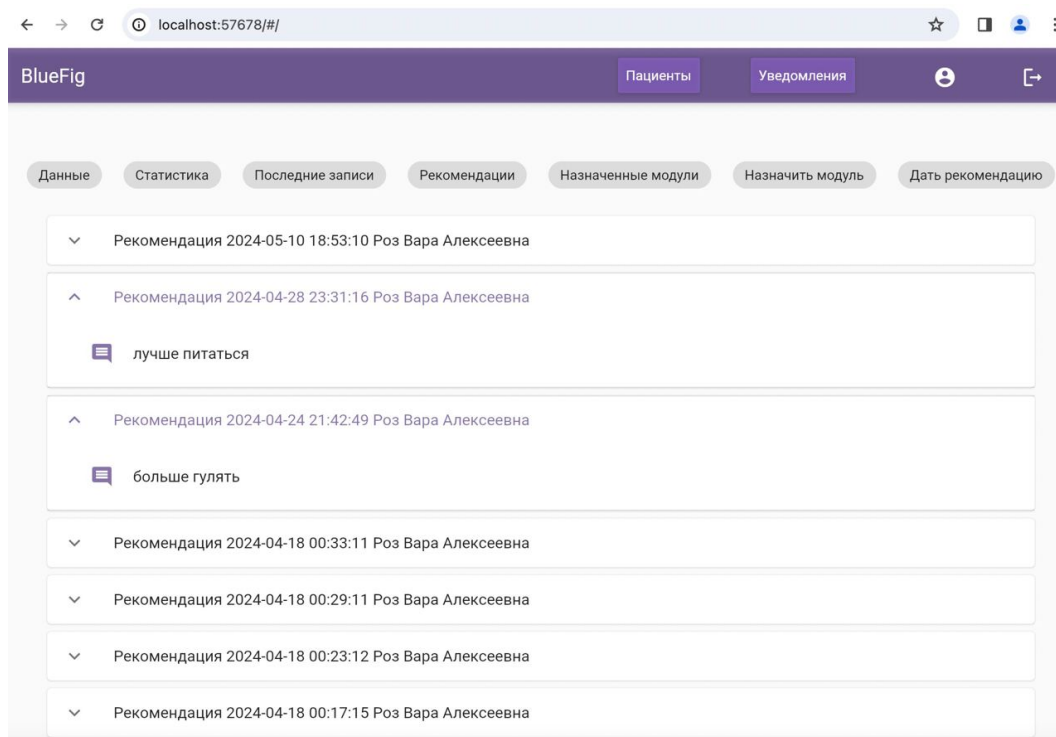


Рис.13 - Страница с рекомендациями

3.3.6. Просмотр назначенных модулей

При переходе на вкладку «Назначенные модули» отображается список с назначенными пациенту модулями. Элемент модуля включает в себя название модуля, параметры, входящие в этот модуль, частоту, с которой сейчас назначен модуль, поле для изменения частоты, кнопка «Изменить», кнопка «Удалить».

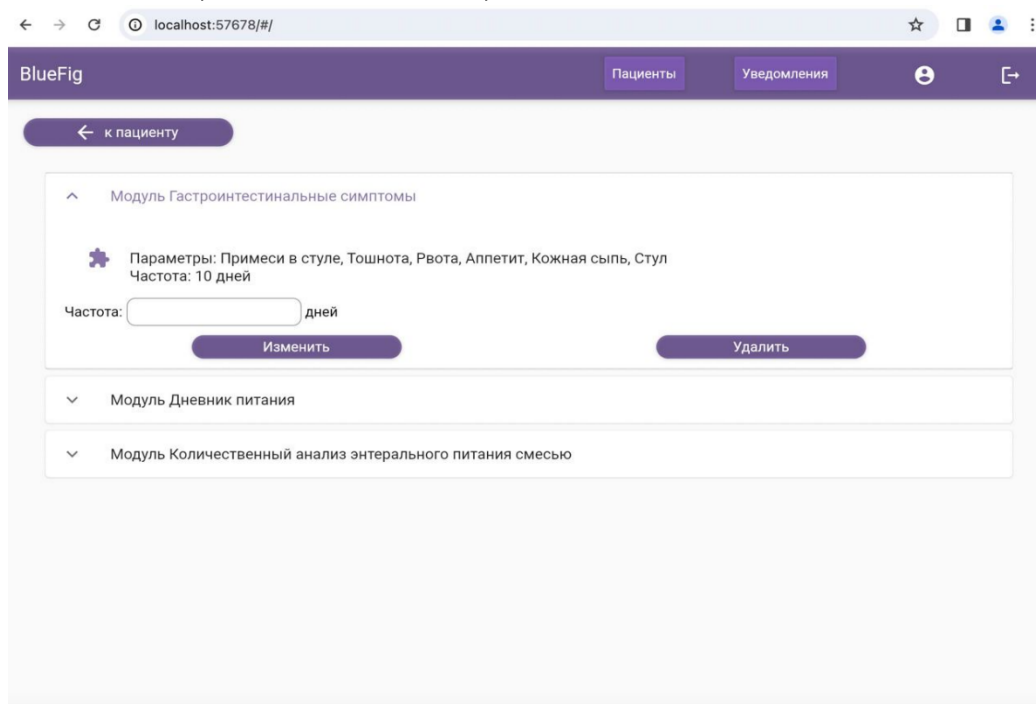


Рис.14 - Страница с рекомендациями

3.3.7. Назначение модуля для заполнения пациенту

При переходе на вкладку «Назначить модуль» отображается список модулей, сверху слева кнопка «к пациенту» для возвращения к данным пациента. В элемент модуля входит название модуля, список параметров, параметры для выбора врача (зависят от вида модуля, пример: вид обратной связи – автоматическая обработка или ручная обработка, цель контроля – снижение или набор веса), поле для задания частоты заполнения модуля, кнопка «Добавить модуль». Первоначально аналогично последним записям модули отображаются в свернутом виде, а при клике на них раскрывается полная информация, содержащая поля для задания модуля, такой вид представления используется для более четкого визуального разделения между представленными модулями.

The screenshot shows a web browser window with the address bar displaying 'localhost:57678/#/'. The application header is purple and contains the 'BlueFig' logo, a 'Пациенты' (Patients) button, and a 'Уведомления' (Notifications) button. Below the header, there is a purple button labeled '← к пациенту' (Back to patient). The main content area is a light gray box containing a list of modules. The first module, 'Модуль Антропометрия' (Anthropometry module), is expanded and shows a gear icon, the parameters 'Вес, Рост, Окружность плеча' (Weight, Height, Arm circumference), a dropdown for 'Вид обратной связи:' (Feedback type) set to 'Ручная обработка' (Manual processing), a dropdown for 'Цель контроля:' (Control goal) set to 'Снижение веса' (Weight reduction), and a frequency field 'Частота:' (Frequency) set to '7' days. A 'Добавить модуль' (Add module) button is located to the right of the frequency field. Below this module are three collapsed modules: 'Модуль Дневник питания' (Diet diary module), 'Модуль Количественный анализ энтерального питания смесью' (Quantitative analysis of enteral nutrition with mixture), and 'Модуль Гастроинтестинальные симптомы' (Gastrointestinal symptoms module).

Рис.15 - Страница с назначением модулей

3.3.8. Написание рекомендации пациенту

При выборе вкладки «Дать рекомендацию» отображается поле для ввода текста рекомендации и кнопка «отправить». Поле расширяется вниз в зависимости от объема набранного текста. После отправки рекомендации происходит возвращение на вкладку с данными пациента.

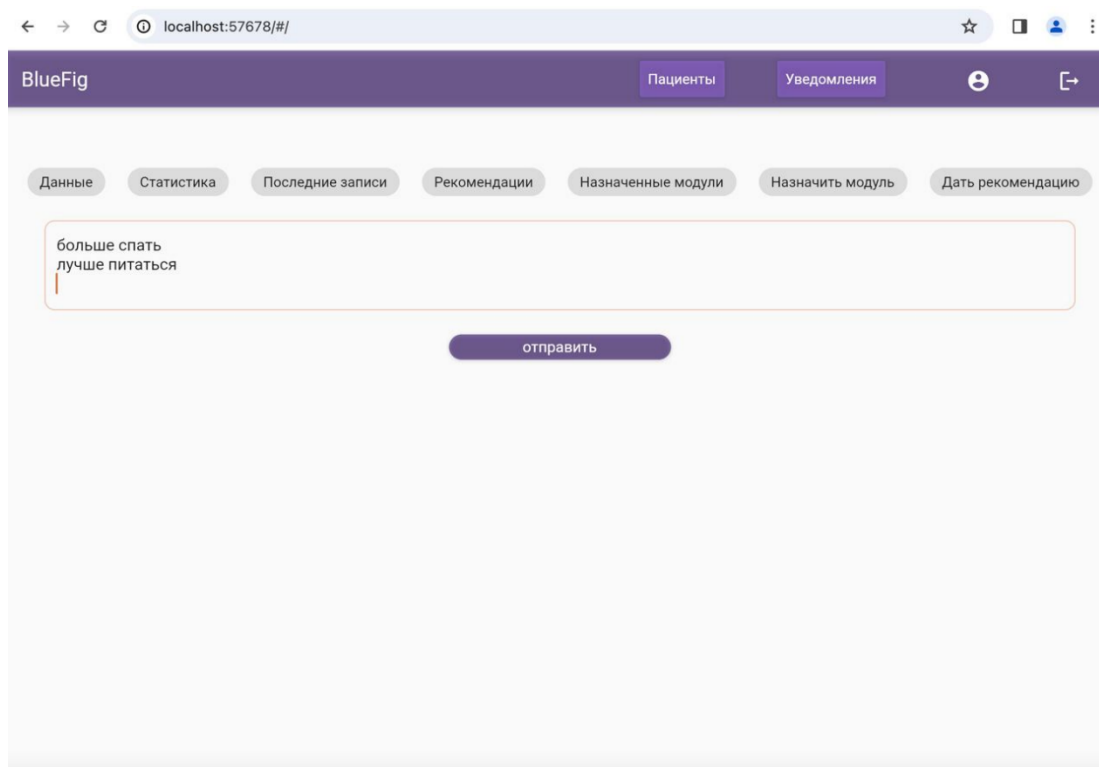


Рис.16 - Страница с написанием рекомендации

3.3.9. Просмотр уведомлений

При клике на кнопку «Уведомления», находящейся в верхней строке, открывается страница с уведомлениями. На ней отображается список с поступившими уведомлениями. Элемент уведомления включает в себя дату и время получения, текст уведомления. Сверху отображаются самые новые уведомления, соответственно внизу самые старые. Уведомления врача бывают о прикреплении нового пациента, о новом срочном пациенте.

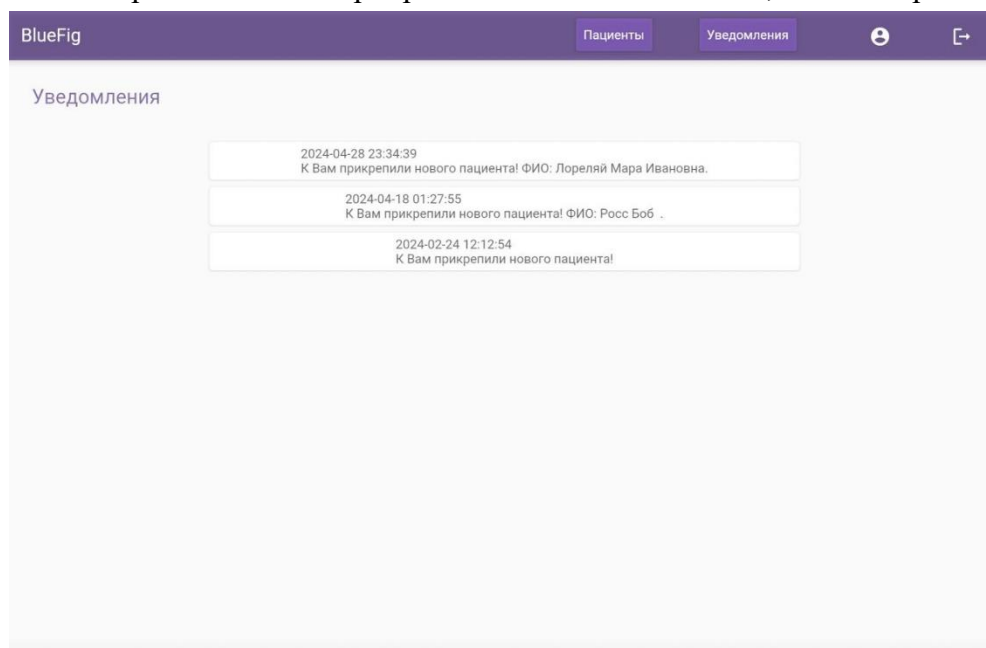


Рис.17 - Страница с уведомлениями

3.3.10. Редактирование информации

При клике на значок профиля в верхней навигационной строке открывается страница с редактированием профиля. После клика на кнопку для начала редактирования отображаются поля, содержащие данные пользователя, такие как юзернейм, имя, фамилия, отчество, дата рождения, почта, пол, пароль. Внизу отображается кнопка «Сохранить», при клике на которую введенные изменения сохраняются. Пользователь может изменить как один параметр, так и несколько. Пароль не отображается в рамках безопасности, но пользователь может поменять его, введя в поле пароля новый пароль.

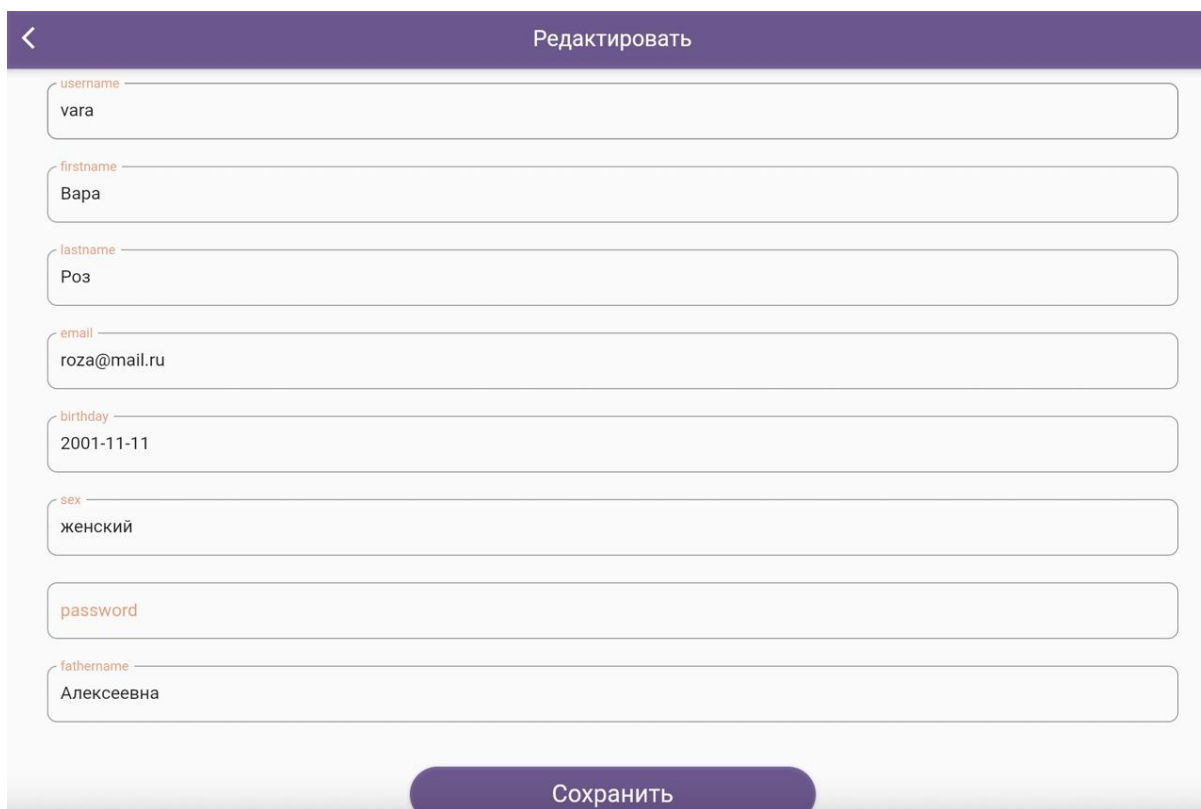


Рис.18 - Страница редактирования информации

3.4. Функционал пациента

3.4.1. Экран с данными

После авторизации пациенту открывается страница с его данными, включающими имя, фамилию, отчество, почту, дату рождения, пол. На странице содержатся кнопки выбора вкладки: «Данные», «Статистика», «Последние записи», «Добавить запись».

Наверху страницы отображается строка навигации, содержащая название приложения в левом углу, кнопки «Данные», «Уведомления», «Рекомендации», иконку профиля и иконку выхода из профиля.

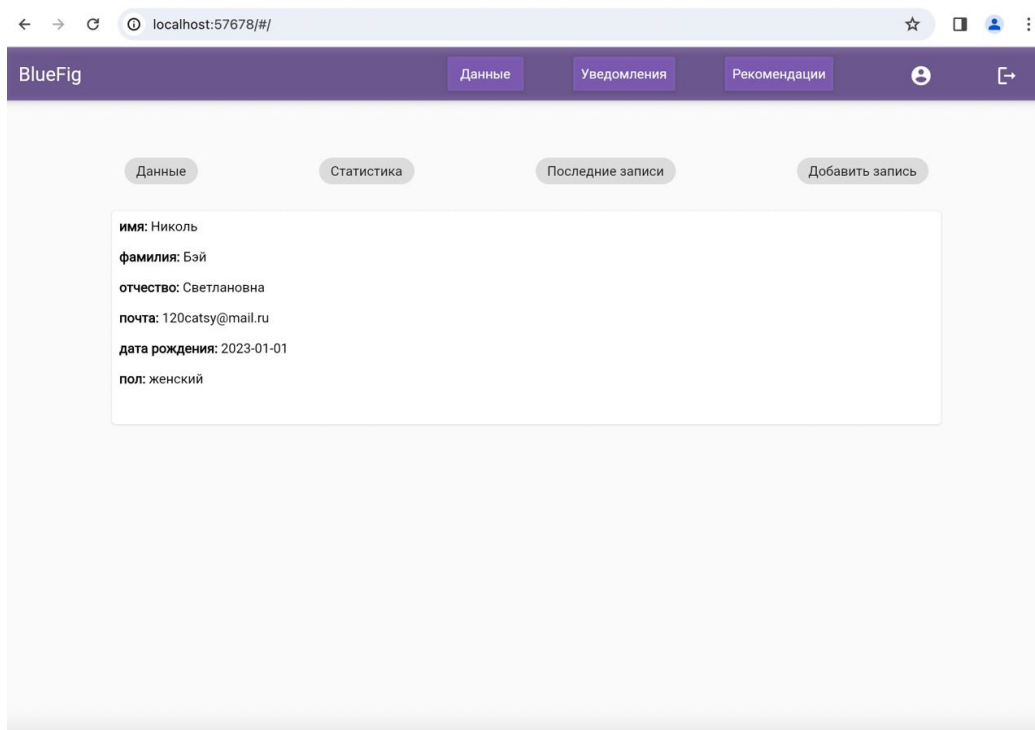


Рис.19 - Страница данных пользователя

3.4.2. Просмотр статистики

При переходе на вкладку «Статистика» отображаются графики численных показателей здоровья пациента: рост, вес, окружность плеча. Вертикальная шкала – значение, горизонтальная дата. На графиках пациент может отслеживать динамику изменений своих показателей. Нижняя шкала может изменяться в зависимости от разницы временных промежутков, то есть могут отображаться дни, при большем количестве заполненных данных нижняя шкала станет показывать месяцы и т.д.

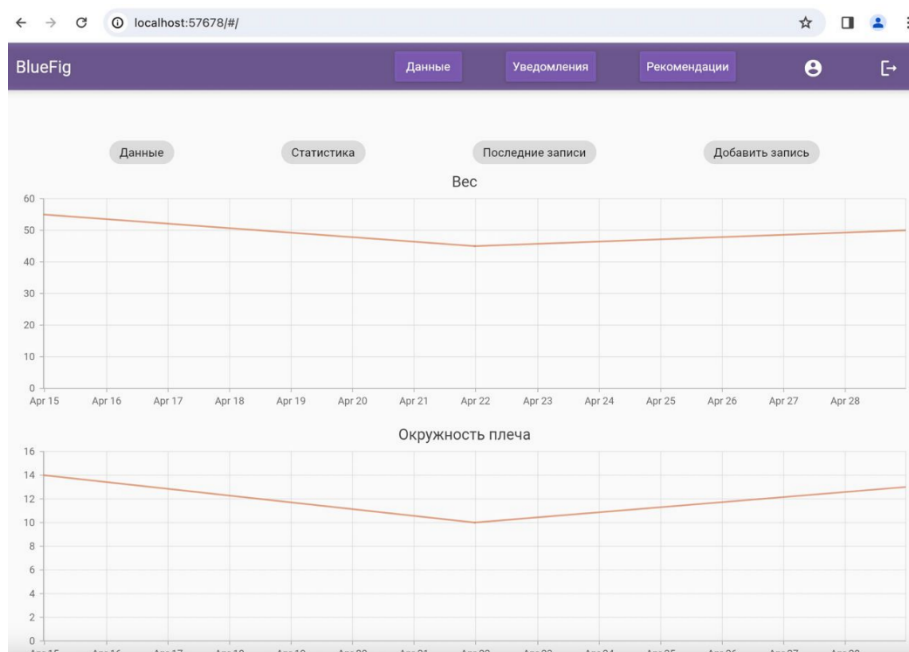


Рис.20 - Страница данных пользователя

3.4.3. Просмотр последних записей

При переходе на вкладку «Последние записи» отображается список с последними записями пациента. Элемент записи содержит информацию о дате и времени записи, названии модуля записи и параметров с введенным пациентом значениями. Изначально отображаются в свернутом виде, при клике – раскрываются.

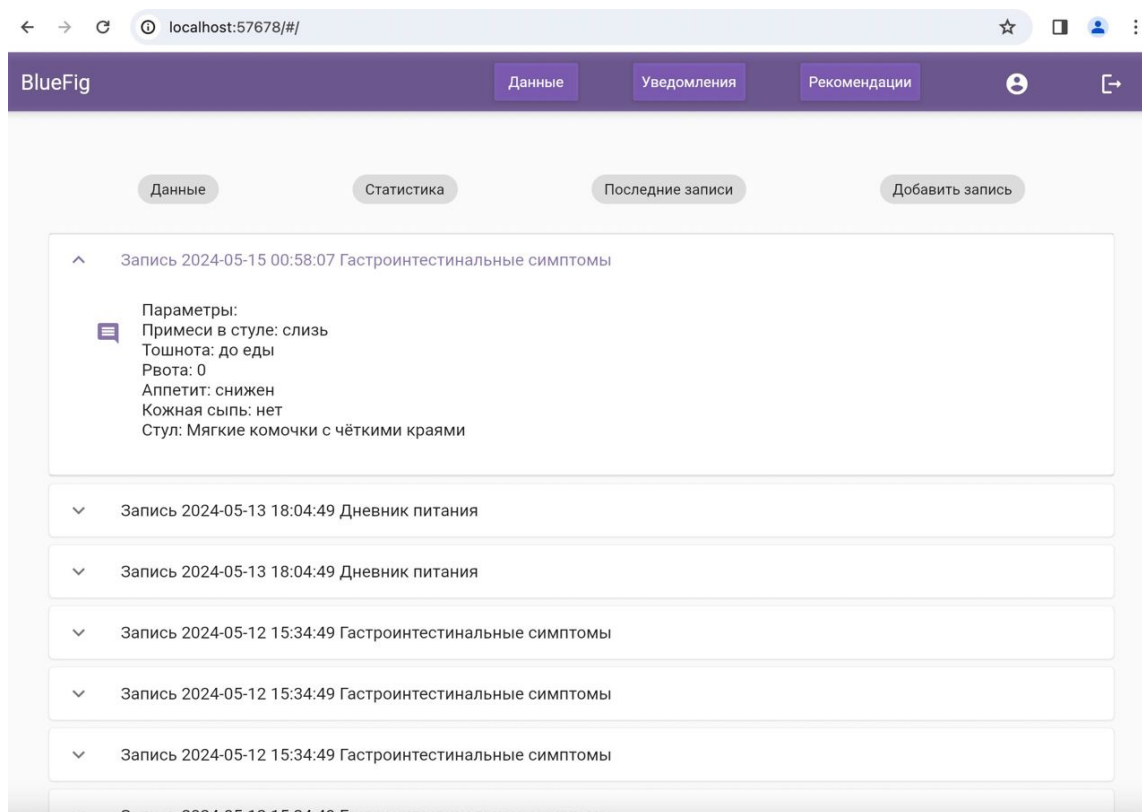


Рис.21 - Страница последних записей пользователя

3.4.4. Добавление записи

При переходе на вкладку «Добавить запись» отображается список с доступными модулями для заполнения. Элемент модуля содержит название модуля, поля необходимые для заполнения, кнопку «Добавить запись». Поля для заполнения бывают различного типа ввода: ввода текста (например, вес, рост, окружность плеча), множественного выбора (например, примеси в стуле), единичного выбора (например, аппетит), флага (например, кожная сыпь).

BlueFig

← на главную

Данные Уведомления Рекомендации

Модуль Гастроинтестинальные симптомы

Примеси в стуле: Выберите

Тошнота: Выберите

Рвота: кол-во раз в день

Аппетит: Выберите

Кожная сыпь: ☐

Стул: Выберите

Добавить запись

Рис.22 - Страница заполнения модулей

Заполнения дневника питания происходит путем создания списка потребленных продуктов с указанием веса в граммах. Пользователь добавляет продукты по одному, кликая по кнопке «новый продукт», а затем выбирая нужный из предложенного списка и указывая его вес. После составления списка и клика на кнопку «добавить запись» появляется сообщение о добавление записи или об ошибке при заполнении.

BlueFig

← на главную

добавить запись

Данные Уведомления Рекомендации

Продукт 1

Продукт: Баклажаны консервированные

Вес: 100

Продукт 2

Продукт: Капуста кале

Вес: 200

+ новый продукт

Рис.23 – Заполнение дневника питания

3.4.5. Просмотр уведомлений

При клике на кнопку «Уведомления», находящейся в верхней строке, открывается страница с уведомлениями. На ней отображается список с поступившими уведомлениями. Уведомления у пациента бывают о напоминании заполнить модуль, о прикреплении к врачу, о назначении нового модуля для заполнения, об изменении частоты заполнения модуля. Элемент уведомления включает в себя дату и время получения, текст уведомления.

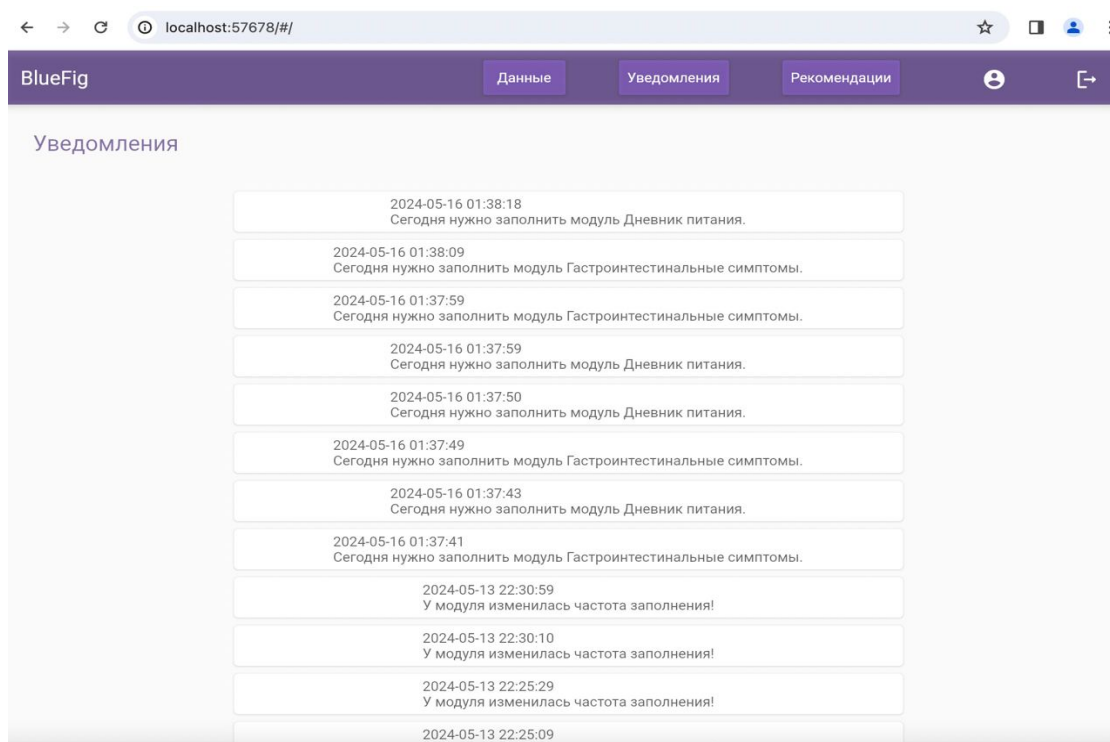


Рис.24 - Страница уведомлений

3.4.6. Просмотр рекомендаций

При клике на кнопку «Рекомендации», находящейся в верхней строке, открывается страница с рекомендациями от врачей. На ней отображается список с поступившими рекомендациями. Элемент рекомендации включает в себя дату и время получения, ФИО врача, текст рекомендации. Рекомендации отсортированы сверху вниз по новизне, то есть вверху самые новые, чем ниже, тем раньше была получена рекомендация. Рекомендации так же представлены в виде раскрывающихся элементов.

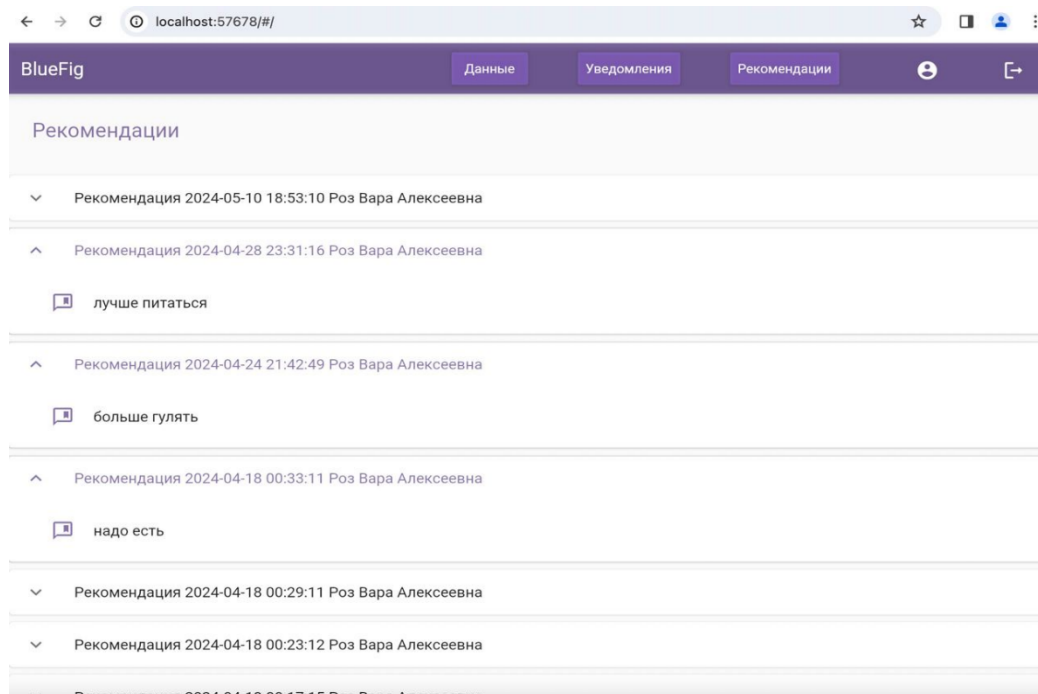


Рис.25 - Страница рекомендаций

3.4.7. Редактирование информации

При клике на значок профиля в верхней строке открывается страница с редактированием профиля. После клика на кнопку для редактирования отображаются поля, содержащие данные пользователя, такие как юзернейм, имя, фамилию, отчество, дату рождения, почту, пол, пароль. Внизу отображается кнопка «Сохранить», при клике на которую введенные изменения сохраняются. В верхнем левом углу находится иконка стрелки для перехода обратно к странице с данными.

← → ↻ ⓘ localhost:57678/#/ ☆ □ 👤 ⋮

< Редактировать

Нажмите для редактирования данных

username
niKKii

firstname
Николь

lastname
Бэй

email
120catsy@mail.ru

birthday
2023-01-01

sex
женский

password

fathername
Светлановна

Рис.26 - Страница редактирования информации

3.5. Функционал администратора

3.5.1. Просмотр списка докторов

После успешной авторизации администратора, открывается страница со списком докторов. В строке навигации наверху находятся кнопки «Пациенты», «Доктора», иконка выхода. На странице со списком доктором находятся кнопки «Поиск» и «Добавить», отображается список всех зарегистрированных докторов.

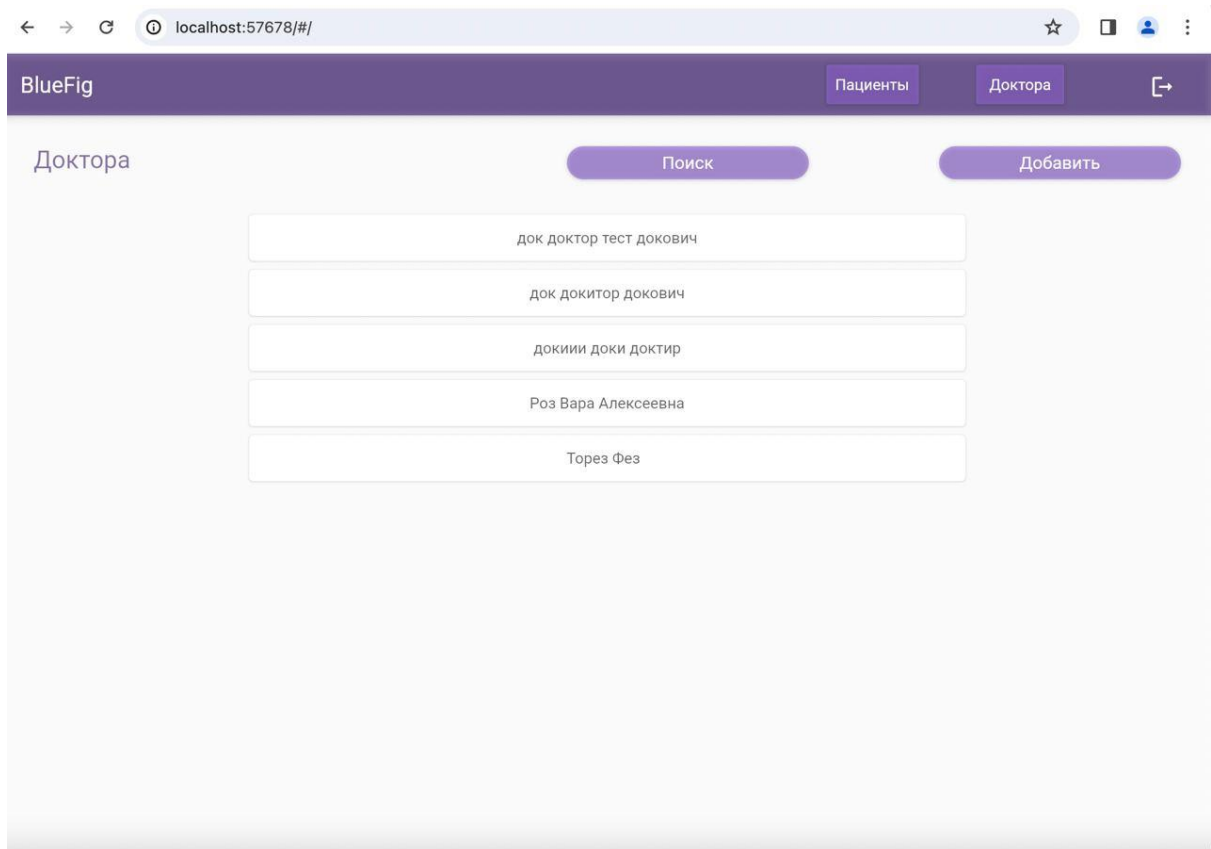


Рис.27 - Страница со списком докторов

3.5.2. Поиск докторов по фамилии

При клике на кнопку «Поиск» открывается окно поиска, содержащее строку для ввода и кнопку поиска. После ввода фамилии в строку поиска и клика на иконку поиска, отображается список подходящих докторов (с соответствующей фамилией). При клике на доктора можно перейти на страницу с его данными.

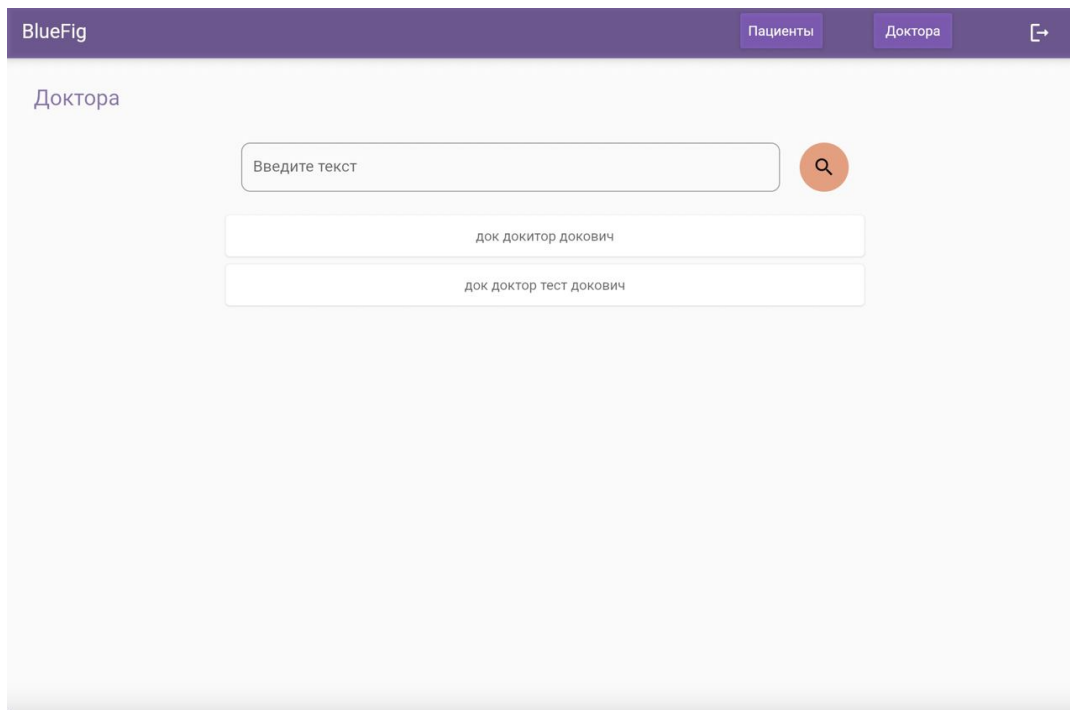


Рис.28 - Страница поиска докторов

3.5.3. Просмотр списка пациентов

При клике на кнопку «Пациенты» в строке навигации, откроется страница со списком пациентов, на которой отображается список всех зарегистрированных пациентов и кнопки «Поиск», «Добавить».

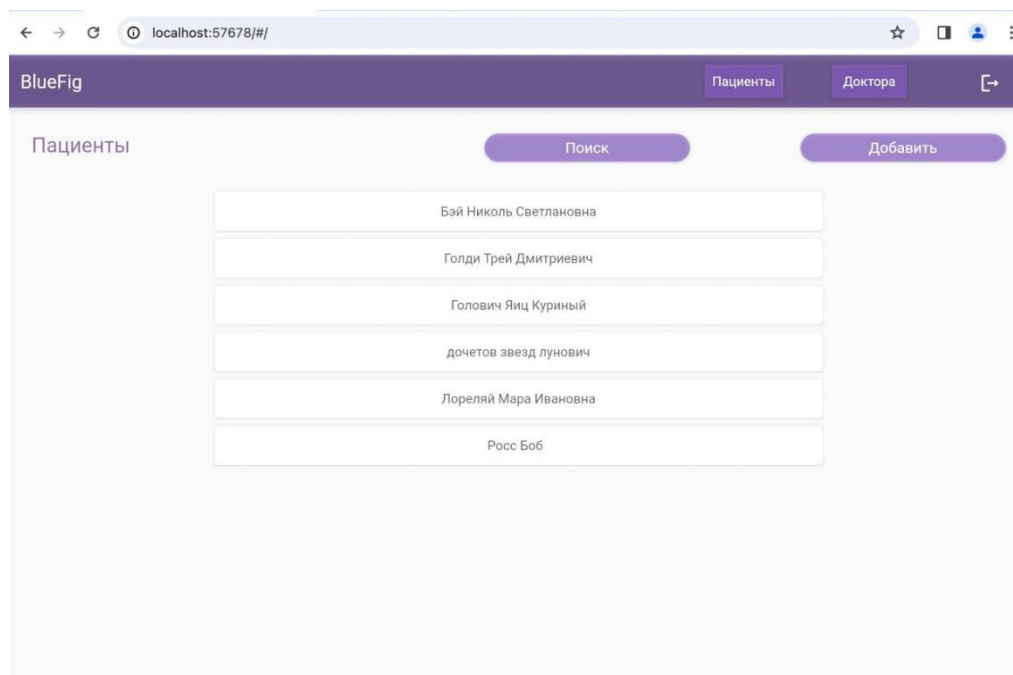


Рис.29 - Страница списка пациентов

3.5.4. Поиск пациентов по фамилии

Поиск пациентов реализован аналогично поиску докторов по фамилии.

3.5.5. Просмотр профиля пользователя

При клике на какого-то пациента или врача открывается страница с его данными. На странице содержатся три кнопки «Прикрепить к врачу»/ «Прикрепить к пациенту», «Редактировать», «Удалить», ниже поле, отображающее его данные (имя, фамилия, отчество, дата рождения, пол, почта).

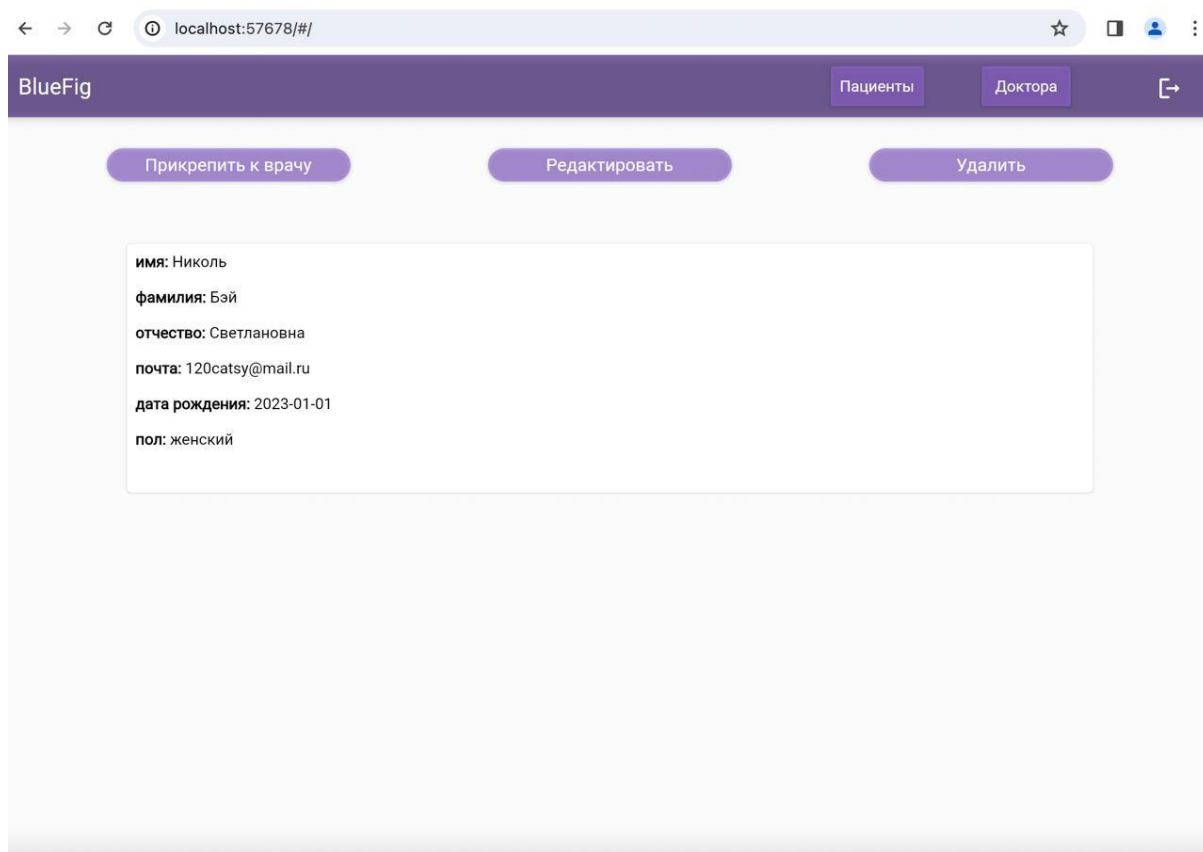


Рис.30 - Страница данных пациента

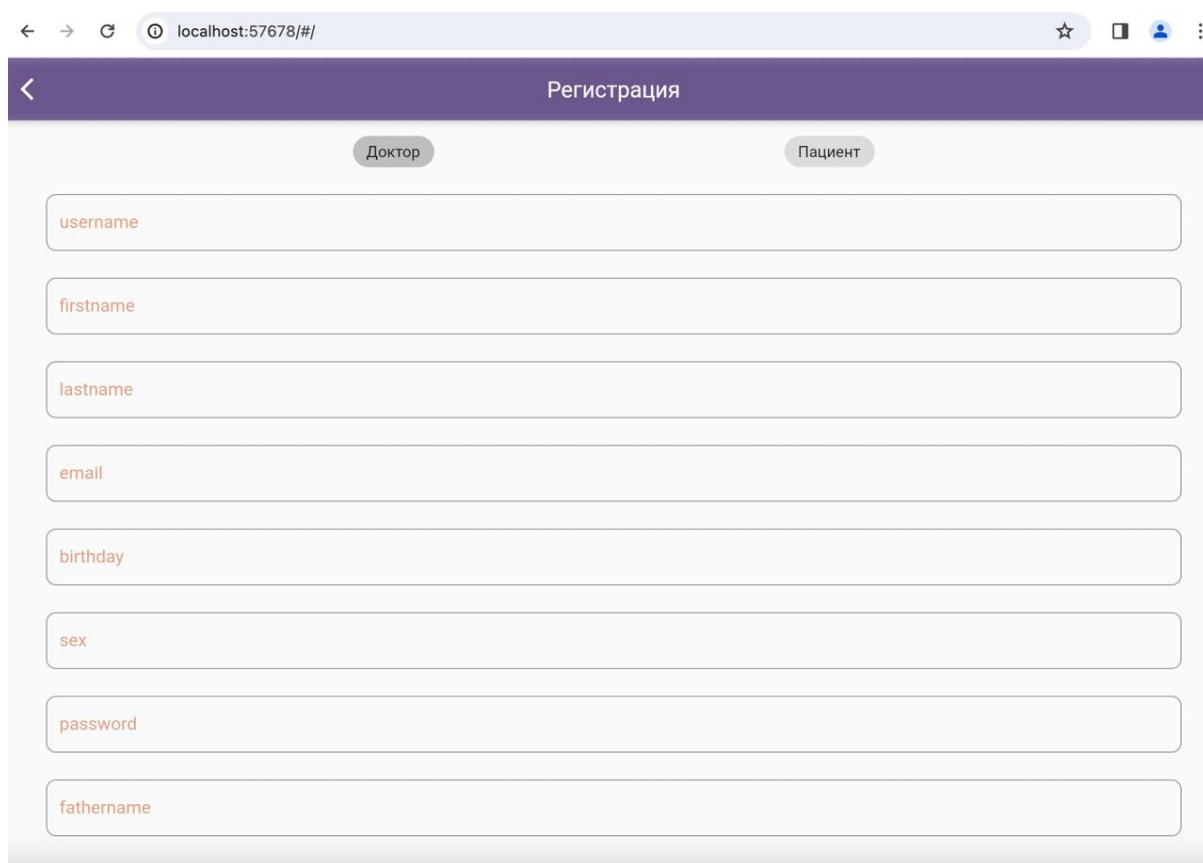
3.5.6. Редактирование информации пользователя

Редактирование информации пользователя со стороны администратора реализовано аналогично редактированию информации со стороны пациента/врача, за исключением возможности менять пароль пользователю из аккаунта администратора (у администратора нет возможности поменять пароль пользователю).

3.5.7. Регистрация новых пользователей

При клике на кнопку «Добавить», находящуюся на странице со списком пациентов/врачей, открывается страница с регистрацией пользователя. На ней находятся кнопки выбора роли (доктор или пациент) и поля для ввода необходимых для регистрации параметров (юзернейм, фамилия, имя, отчество, почта, дата рождения, пол, пароль), кнопка «Сохранить». После регистрации высвечиваются уведомление об успешной регистрации или о некорректном вводе параметров.

Пользователей в системе регистрировать может только администратор, такое ограничение обусловлено структурой медицинского учреждения, для которого ведется разработка.



The screenshot shows a web browser window with the address bar displaying 'localhost:57678/#/'. The page title is 'Регистрация'. There are two buttons at the top: 'Доктор' and 'Пациент'. Below these are eight input fields for registration details: 'username', 'firstname', 'lastname', 'email', 'birthday', 'sex', 'password', and 'fathername'.

Рис.31 - Страница регистрации пользователя

3.5.8. Создание пары врач-пациент

При клике на кнопку «Прикрепить к врачу»/ «Прикрепить к пациенту» на странице с данными пользователя открывается страница со списком пациентов/врачей соответственно и кнопкой «Прикрепить». Выбранный пациент/врач выделяется фиолетовым цветом. Чтобы прикрепить пациента/врача необходимо выбрать нужного и

нажать на кнопку «Прикрепить». После высвечивается сообщение об успешном назначении или произошедшей ошибке.

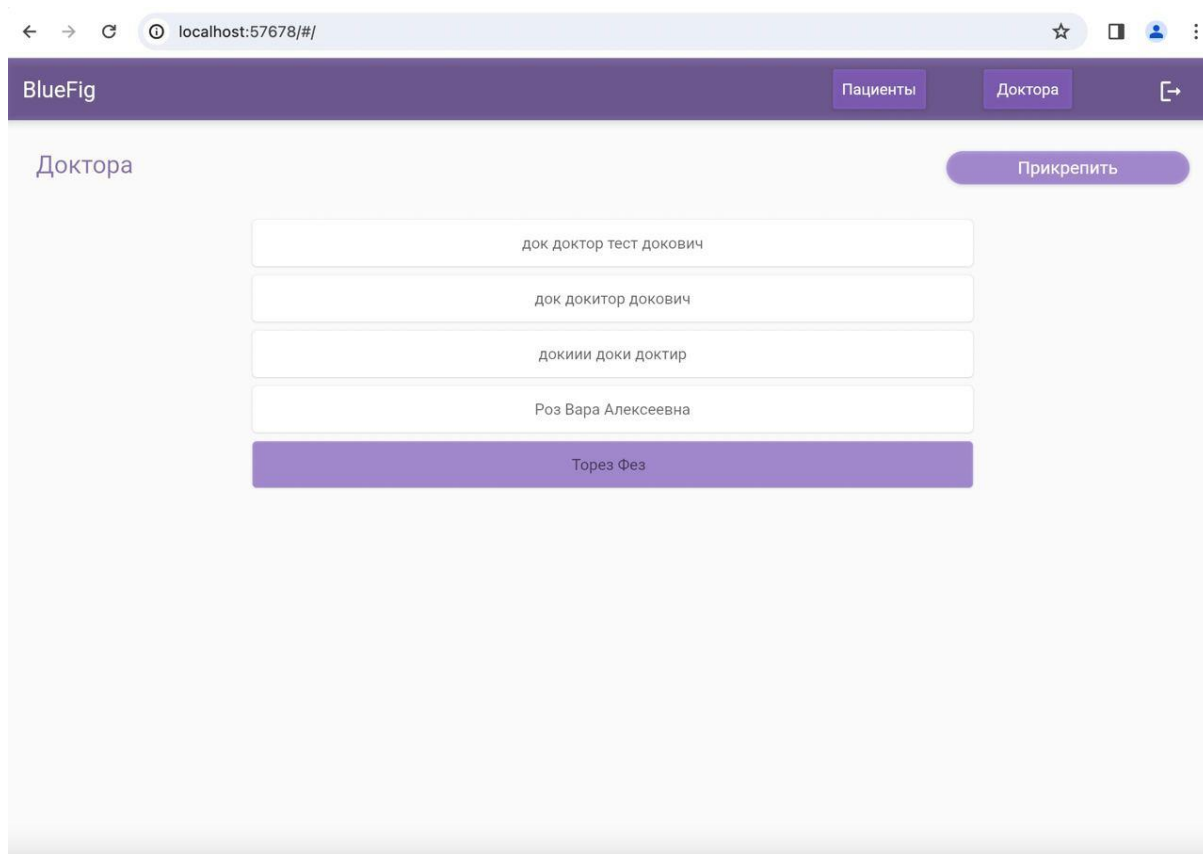


Рис.32 - Страница прикрепления врача к пациенту

3.6. Вывод по главе

Выше был описан функционал веб-сервиса, подробно рассмотрены три вида интерфейса – для врача, для пациента, для администратора – и доступные каждому из них функции приложения.

Заключение

В рамках работы над выпускной квалификационной работы была разработана клиентская часть веб-сервиса для мониторинга питания пациентов со спинальной мышечной атрофией, позволяющее проще и своевременнее оказывать необходимую пациентам помощь, а также следить за их состоянием.

Была рассмотрена и описана предметная область разработки, проанализированы существующие аналоги, описаны используемые технологии и средства разработки, архитектура системы в целом и в частности клиентская часть, подробно рассмотрен функционал каждого пользователя.

Клиентская часть включает в себя три вида интерфейсов: для врача, для пациента, для администратора. У каждого из видов пользователей различный доступный функционал, отвечающий поставленным требованиям.

Для реализации веб-сервиса использовался язык Dart с фреймворком Flutter, что необходимо для упрощения дальнейшего развития сервиса в мобильные приложения для различных операционных систем.

В процессе работы над ВКР были выполнены следующие задачи:

1. Проведено изучение и анализ предметной области разработки;
2. Проведен сравнительный анализ существующих аналогов;
3. Изучено применение технологии Flutter для реализации веб-приложений;
4. Разработан прототип интерфейса;
5. Разработана клиентская часть веб-сервиса для мониторинга питания пациентов с СМА:
 - a. Разработан интерфейс для врача;
 - b. Разработан интерфейс для пациента;
 - c. Разработан интерфейс для администратора;
6. Налажено клиент-серверного взаимодействие;
7. Подготовлена программная документация.

В планы дальнейшего развития проекта входит проведение тестирования с целевыми пользователями веб-сервиса (врачами, пациентами с СМА и их помощниками), улучшение веб-сервиса по результатам тестирования, адаптация приложения под мобильные версии для различных операционных систем, внедрение в организацию, работающую с пациентами со спинальной мышечной атрофией.

Список использованных источников

1. Apple Health [Электронный ресурс] / – Режим доступа: <https://apps.apple.com/us/app/apple-health/> свободный. (дата обращения: 15.11.23)
2. BetaLife [Электронный ресурс] / – Режим доступа: <https://betalife.app/> свободный. (дата обращения: 15.11.23)
3. Bowelle [Электронный ресурс] / – Режим доступа: <https://apps.apple.com/ru/app/bowelle> свободный. (дата обращения: 15.11.23)
4. Dart [Электронный ресурс] / – Режим доступа: <https://dart.dev/>, свободный. (дата обращения: 15.11.23)
5. DocDoc [Электронный ресурс] / – Режим доступа: <https://docdoc.ru/> свободный. (дата обращения: 15.11.23)
6. Medesk [Электронный ресурс] / – Режим доступа: <https://www.medesk.net/> свободный. (дата обращения: 15.11.23)
7. SmartМедицина [Электронный ресурс] / – Режим доступа: <https://smartmedicina.ru/> свободный. (дата обращения: 15.11.23)
8. Здоровье.ру [Электронный ресурс] / – Режим доступа: <https://apps.apple.com/ru/app/> свободный. (дата обращения: 15.11.23)
9. Сайт docs.flutter [Электронный ресурс] / – Режим доступа: <https://docs.flutter.dev/>, свободный. (дата обращения: 15.11.23)
10. Фонд Семьи СМА [Электронный ресурс] / – Режим доступа: <https://f-sma.ru/> свободный. (дата обращения: 15.11.23)
11. Яндекс.Здоровье [Электронный ресурс] / – Режим доступа: <https://health.yandex.ru/> свободный. (дата обращения: 15.11.23)
12. http [Электронный ресурс] / – Режим доступа: <https://pub.dev/packages/http> свободный. (дата обращения: 20.12.23)
13. multiselect [Электронный ресурс] / – Режим доступа: <https://pub.dev/packages/multiselect> свободный. (дата обращения: 20.12.23)
14. syncfusion_flutter_charts [Электронный ресурс] / – Режим доступа: https://pub.dev/packages/syncfusion_flutter_charts свободный. (дата обращения: 20.12.23)
15. Visual Studio Code [Электронный ресурс] / – Режим доступа: <https://code.visualstudio.com/> свободный. (дата обращения: 20.12.23)
16. Figma [Электронный ресурс] / – Режим доступа: <https://www.figma.com/> свободный. (дата обращения: 01.12.23)