МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

Кафедра теории функций и стохастического анализа

РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ ЧАСТНОЙ МЕДИЦИНСКОЙ КЛИНИКИ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

Студентки 4 курса 451 группы
направления 38.03.05 — Бизнес-информатика

механико-математического факультета
Ереминой Дарьи Сергеевны

Научный руководитель
ст. преподаватель _____ Н. В. Сергеева

Заведующий кафедрой
д. ф.-м. н., доцент _____ С. П. Сидоров

Саратов 2023

ВВЕДЕНИЕ

Актуальность темы. В современном мире, где всё больше людей используют мобильные устройства для удобства и экономии времени, разработка мобильных приложений является все более актуальной. Практически каждому предприятию, в основном бизнесу, рано или поздно приходит необходимость создания своего мобильного приложения. Мобильные приложения делятся на несколько видов: для поддержки мобильных технологии внутри компании, для потребителей — клиентов компании и бизнес-приложения. Приложения для потребителей представляют собой достаточно большую и важную группу. Они решают множество задач — от расширения присутствия компании на рынке, до развлекательного контента. Поэтому именно приложение из этой группы будет создаваться для частной медицинской клиники.

Данная работа представляет собой интерес, так как разработанное с учётом потребностей пациентов мобильное приложение частной медицинской клиники позволило бы улучшить доступность медицинских услуг, сократить время на запись на приём, оптимизировав таким образом работу некоторого персонала, улучшить коммуникацию с пациентами и повысить их лояльность, а также предоставлять информацию клиентов для оптимизации бизнес-процессов клиники.

Целью бакалаврской работы является разработка удобного мобильного приложения частной медицинской клиники при помощи фреймворка Flutter.

Объект исследования — процесс обслуживания пациентов частной медицинской клиники, включая запись на прием, получение и хранение результатов анализов и других обследований, консультации врачей.

Предмет исследования — потребности и предпочтения пользователей при использовании мобильных приложений в медицинской сфере.

Для достижения поставленных целей в работе необходимо решить следующие **задачи**:

— проанализировать предметную область, целевую аудиторию, конкурентов;

— создать прототипы интерфеса приложения и его дизайн;

— спроектировать базу данных;
— разработать функционал приложения, используя фреймворк Flutter.

Практическая значимость проведенной работы заключается в улучшении качества обслуживания пациентов, повышении эффективности работы клиники а также повышении лояльности её клиентов. Мобильное приложение позволяет пациентам получать информацию о филиалах клиники и её врачах, быстро и удобно записываться на прием, получать результаты анализов и обследований, что сокращает время ожидания и упрощает процесс обращения в клинику.

Структура и содержание бакалаврской работы. Работа состоит из введения, четырёх разделов, заключения, списка использованных источников, содержащего 24 наименования, и 1 приложения. Общий объём работы без приложения составляет 45 страниц.

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Во **введении** обосновывается актуальность темы работы, формулируется цель работы и решаемые задачи, отмечается практическая значимость получаемых результатов.

В **первом** разделе проводится анализ предметной области и поиск решений. Частная медицинская клиника представляет собой учреждение здравоохранения, специализирующееся на оказании медицинских услуг частным лицам. Клиника обладает современным оборудованием и высококвалифицированным персоналом, включающим врачей различных специальностей, медицинских сестер, администраторов и других специалистов. Она предоставляет широкий спектр медицинских услуг, включая консультации врачей, диагностику, лечение заболеваний, профилактику и реабилитацию. Клиника также может предложить программы корпоративного здоровья для компаний, которые заинтересованы в поддержании здоровья своих сотрудников. Частная медицинская клиника стремится к высокому качеству медицинского обслуживания и индивидуальному подходу к каждому пациенту, обеспечивая им комфортное и безопасное пребывание в клинике.

Мобильное приложение может значительно улучшить бизнес-процессы частной медицинской клиники, обеспечивая более удобный и быстрый доступ

к медицинским услугам и информации о клинике. В данной работе при создании мобильного приложения упор был сделан на повышение эффективности работы клиники, сокращение времени сотрудников на запись пациентов, а также повышение лояльности клиентов.

Для того, чтобы создать качественный продукт, необходимо провести исследования возможных пользователей и конкурентов, понять, какой именно продукт будет востребован у будущих пользователей.

Для анализа конкурентов были выбраны следующие приложения:

— МедТочка;
— Мой Зубной;
— Скандинавия;
— CLPRIME;
— Smartmed;
— Альфа-центр.

После их рассмотрения была составлена сравнительная таблица конкурентов и описаны некоторые гипотезы для дальнейшего анализа.

Дальнейший анализ целевой аудитории проводился с помощью методов опроса и глубинного интервью. Опрос собирает в основном количественные данные, а интервью позволяет получить качественную информацию от респондентов. Для создания опроса использовались «Google Формы», как один из самых популярных, простых в использовании и бесплатных сервисов.

В результате анализа предметной области, конкурентов и целевой аудитории было сформулировано следующее решение :

«Создать сбалансированное приложение — не перегруженное лишними сервисами, позволяющее быстро и качественно выполнить основные функции — запись на прием, хранение результатов, выбор врача и филиала. Остальные функции — полезные советы, акции — должны быть лишь приятным дополнением в пользовании приложением».

На основе анализа и сформулированного решения была создана информационная архитектура будущего приложения. Она не является частью видимого на экране, она показывает отображаемый функционал будущего интерфейса, связи между функциями или страницами, а также зависимость переходов в соответствии с выполненными действиями. Информационную ар-

хитектуру можно назвать расширенной версией карты сайта, которая создаётся при помощи блок-схем. На основе информационной архитектуры будет проводится дальнейшее проектирование и разработка.

Во **втором** разделе рассматриваются методы проектирования интерфейса и создания дизайна приложения, которые затем применяются практически.

Figma — это графический онлайн-редактор с обширным функционалом, который при этом остается простым в использовании. Он был создан в 2012 году как альтернатива Adobe Photoshop специально для веб-дизайнеров, но в настоящее время его пользовательская база расширилась и включает в себя маркетологов, разработчиков, менеджеров и других профессионалов. Figma получил признание за скорость работы и отсутствие ненужных функций, которые не имеют отношения к веб-дизайну, что делает его популярным выбором среди дизайнеров. С помощью Figma можно создавать различные дизайн-продукты, включая баннеры, диаграммы, графики, логотипы, сайты, мобильные приложения. Особенности Figma:

— сервисом можно пользоваться индивидуально (только вы работаете над проектом) или командой (один проект редактируют два и более пользователя);
— Figma полностью бесплатна для индивидуальных пользователей. Также в бесплатной версии вместе может работать команда из максимум пяти человек с ограничениями по количеству проектов. Если требуется больше, нужно платить;
— в интерфейсе Figma доступно только два языка: английский и китайский. Но даже люди без знания этих языков легко осваивают сервис.

С помощью этого онлайн-редактора проходило дальнешее прототипирование и создание дизайна.

Прототипирование — это процесс создания низкодетализированных образов и набросков будущего интерфейса, чтобы показать идею и логику будущего продукта, не отвлекаясь на визуал. Даже если задача кажется простой, и хочется сразу начать отрисовывать дизайн-макет, стоит все же начать с прототипирования. Прототипы помогут:

— не увязнуть в деталях и передвигании пикселей;

— сфокусироваться на функционале, а не на графическом оформлении;
— избежать логических ошибок в интерфейсе.

Основная задача прототипа — быстро перевести информационную архитектуру и всю информацию, полученную при исследованиях, в понятные схематичные экраны.

На следующем этапе дизайна решается, как будут выглядеть элементы, подбираются иконки, иллюстрации, шрифты, цвета — общий стиль. Здесь необходимо следить не только за общей привлекательностью интерфейса, но и за тем, чтобы пользователь верно считывал элементы интерфейса во всех состояниях экрана, а сам интерфейс соответствовал стандартам доступности. Чтобы грамотно разработать визуальную часть интерфейса, нужно обладать теоретическими знаниями в нескольких сферах:

— теория цвета;
— типографика;
— сетка;
— композиция;
— гайдлайны.

В интерфейсе мобильного приложения частной медицинской клиники использовалась сдержанная цветовая палитра, где красный и зеленый цвета гармонично сочетаются, при этом ассоциируясь напрямую с медициной и здоровьем. Шрифт использовался стандартный для Android — Roboto. Всего было отрисовано 40 экранов приложения, полностью готовых к дальнейшей разработке.

Третий раздел посвящен теоретическим основам NoSQL и созданию базы данных нереляционной модели при помощи Isar.dev.

Базы данных NoSQL — это тип баз данных, который не использует реляционную модель данных и SQL-язык для работы с данными. Вместо этого они используют другие модели данных, такие как документы, графы или ключ-значение.

Одним из основных преимуществ баз данных NoSQL является гибкость в работе с данными. Они могут хранить данные различных типов, без необходимости определения схемы заранее. Это позволяет быстро изменять структуру данных при необходимости. Также базы данных NoSQL обеспечивают

высокую доступность и масштабируемость. Они могут работать на нескольких серверах, что позволяет распределять нагрузку и обеспечивать отказоустойчивость. Кроме того, базы данных NoSQL могут обрабатывать большие объемы данных и выполнять запросы на них быстрее, чем реляционные базы данных.

В дальнейшем разработка приложения велась при помощи фреймворка Flutter. Существуе множество специальных ресурсов для работы с Flutter, например, pub.dev, предоставляющих различные пакеты и библиотеки, в том числе для упрощения взаимодействия между серверной частью приложения и базой данных. Один из таких пакетов — Isar.dev. Он представляет собой базу данных для приложений на Flutter. Isar.dev позволяет разработчикам быстро и легко создавать и управлять базами данных в своих приложениях на Flutter. Он предоставляет высокую производительность и эффективность при работе с большим объемом данных. Isar.dev также предоставляет инструменты для автоматической синхронизации данных между устройствами и облачными сервисами. Он является важным инструментом для разработчиков, работающих с Flutter и Dart, и помогает им создавать мощные и эффективные приложения с базами данных.

Механизм работы Isar следующий:

1. Добавление зависимостей.
2. Аннотация классов.
3. Запуск генератора кода.
4. Открытие экземпляра Isar.
5. Использование коллекций базы данных.

В созданной базе данных Isar, представляющей собой свойственные NoSQL наборы данных, хранятся следующие сущности:

— врачи;
— пациенты;
— клиники;
— записи;
— отзывы.

Каждая сущность представляет собой json-документ с парами «ключ-значение». Например, у врача это: ФИО, дата рождения, пол, режим, специальность,

образование, филиалы. У сущности «отзывы» же будут следующие пары: пациент, врач, клиника, дата, оценка, текст отзыва.

В **четвёртом** разделе рассматриваются фреймворк для разработки кросплатформенных приложений Flutter, способ представления данных DTO, протокол gRPC, паттерн архитектуры BLoC, на основе которых в дальнейшем происходит разработка приложения.

Flutter — это фреймворк для разработки мобильных приложений, который был разработан компанией Google. Он был создан с нуля на языке программирования Dart, разработке того же Google. Данный язык позиционируется Google как альтернатива JavaScript со строгой типизацией, высокой производительностью и гибкостью.

Flutter был выпущен в 2017 году и с тех пор стал популярным инструментом для создания мобильных приложений. Он используется многими компаниями, включая Google, Alibaba и Tencent. Flutter поддерживает создание приложений для веба и настольных компьютеров. Он позволяет создавать красивые и высокопроизводительные приложения для Android и iOS с использованием одного и того же кода.

Одной из главных особенностей Flutter является его виджетная система. Виджеты — это основные строительные блоки приложения в Flutter. Они могут быть использованы для создания любого элемента пользовательского интерфейса, от кнопок и текстовых полей до сложных анимаций и 3D-графики.

Flutter также имеет множество встроенных виджетов и библиотек, которые упрощают разработку приложений. Например, Material Design и Cupertino — это две встроенные библиотеки, которые позволяют создавать приложения с использованием стандартных дизайн-элементов для Android и iOS соответственно.

Структурно архитектура создаваемого приложения состоит из нескольких слоев, каждый слой содержит описания данных и их контроллеров. Ни один контроллер не должен ничего знать о других контроллерах своего слоя, их коммуникация должна происходить строго в младших слоях.

В реализации данной архитектуры всего 3 слоя (вниз по старшинству):

— Data — слой данных, набор DTO и DAO;

— Domain — слой предметной области, набор Model и Repository;
— Presentation — слой представления, набор State и Controller (а также View).

DTO — Data Transfer Object — это объект, который используется для передачи данных между слоями приложения или между приложениями. Он содержит только необходимые данные и не имеет логики или методов. DTO используется для уменьшения количества запросов к базе данных и ускорения работы приложения. Это распространённый паттерн, суть которого заключается в создании абстракции над моделью предметной области с целью увеличения эффективности обмена данными между процессами, например, back-end'ом (сервером) и front-end'ом (клиентом).

DAO — Data Access Object — это объект, который используется для доступа к данным в базе данных. Он предоставляет интерфейс для выполнения операций с базой данных, таких как добавление, удаление, обновление и выборка данных. DAO скрывает детали работы с базой данных от остальных частей приложения и позволяет легко заменять один источник данных на другой. Это еще один смежный паттерн, описывающий контракт с набором методов, необходимых для произведения CRUD-операций с какими-либо данными.

Код классов сервисов и сущностей (сообщений), в том числе их сериализаторов и десериализаторов, а также весь остальной код, необходимый для обращения к серверу, был написан используя протокол gRPC. Весь этот код относится к слою Data: сообщения — DTO, а сервисы — DAO.

gRPC – это современный, высокопроизводительный и масштабируемый протокол вызова удалённых процедур (RPC), разработанный компанией Google. Он базируется на протоколе HTTP/2 для обмена данными между клиентом и сервером, а также поддерживает формат кодирования данных Protobuf (Protocol Buffers) и на основе файлов этого формата способен генерировать код на различных языках программирования.

Работа протокола gRPC состоит из следующих шагов:

1. Определение сервиса и сообщений: разработчики определяют сервис и сообщения, которые будут использоваться для взаимодействия между клиентом и сервером. Они используют язык Protobuf для определения

этих объектов.

2. Генерация кода: после определения сервиса и сообщений, разработчики генерируют код для клиента и сервера. Этот код использует язык программирования, выбранный разработчиками.
3. Регистрация сервиса: сервер регистрирует свой сервис в реестре, чтобы клиент мог найти его.
4. Установление соединения: клиент устанавливает соединение с сервером по протоколу HTTP/2.
5. Вызов удаленной процедуры: клиент вызывает удаленную процедуру на сервере, используя определенное сообщение.
6. Обработка запроса: сервер обрабатывает запрос и возвращает результат клиенту.
7. Закрытие соединения: после завершения работы, клиент закрывает соединение с сервером.

Model представляет собой объекты, которые описывают бизнес-сущности и их свойства. Они не зависят от других слоев приложения и содержат только логику, связанную с бизнес-правилами. Это более детальное описание объекта предметной области, полученное из данных (DTO), предоставленных каким-либо сервисом (DAO).

Repository представляет собой интерфейс для доступа к данным. Он определяет методы для получения, сохранения и удаления данных из источников данных, таких как базы данных, файлы и т.д. Реализация репозитория находится в слое инфраструктуры и зависит от конкретных технологий и инструментов. Этот паттерн дополняет предыдущие и описывает класс объектов, предоставляющих доступ к данным, благодаря манипуляциям с одним или более DAO.

Основой бизнес-логики приложения служит BLoC – этот компонент позволяет легко отделить интерфейс от бизнес-логики, делая код более быстрым, легким для тестирования и повторного использования.

BLoC (Business Logic Component) — это паттерн архитектуры программного обеспечения, который используется для разделения бизнес-логики и пользовательского интерфейса в приложении. Он основан на идее, что приложение должно быть разделено на три основных слоя: пользовательский интер-

фейс, бизнес-логика и слой данных.

При создании приложения высокого качества, управление состоянием становится критически важным. BLoC имеет следующие возможности:

— знать, в каком состоянии находится приложение в любой момент времени;
— легко протестировать каждый случай, чтобы убедиться, что наше приложение отвечает должным образом;
— записывать каждое взаимодействие с пользователем в приложении, чтобы можно было принимать решение не основе этих данных;
— работать максимально эффективно и повторно использовать компоненты как в этом приложении, так и в других приложениях.

BLoC работает следующим образом:

1. Входные данные поступают в пользовательский интерфейс (UI).
2. UI отправляет данные в BLoC.
3. BLoC обрабатывает данные и генерирует новые выходные данные.
4. Выходные данные возвращаются в UI.
5. UI отображает новые данные пользователю.

При отображении пользовательского интерфейса основными составляющими являются виджеты, во Flutter существуют следующие базовые виджеты:

— Text — позволяет создавать стилизованный текст внутри приложения;
— Row, Column — гибкие виджеты, позволяющие создавать различные композиции как по горизонтали (Row), так и по вертикали (Column);
— Container — позволяет создавать прмоугольный визуальный элемент, может быть дополнен фоном, рамкой или тенью с помощью BoxDecoration.

Используя базовые виджеты, а также дополнительные индивидуальные для каждого типа экрана, дизайн-макет приложения из сервиса Figma был полностью переведён в код. А затем, используя вышеперечисленные инструменты разработки, база данных, сервер и интерфейс были объединены в единое целое — мобильное приложение частной медицинской клиники.

В **заключении** приведены результаты бакалаврской работы.

ОСНОВНЫЕ РЕЗУЛЬТАТЫ

1. Проведён анализ предметной области, целевой аудитории и конкурентов, создана информационная архитектура будущего приложения.
2. Были созданы прототипы интерфейса будущего мобильного приложения, а также дизайн интерфейса на основе прототипов.
3. Была спроектирована база данных мобильного приложения.
4. Был разработан функционал мобильного приложения, используя возможности фреймворка Flutter. Полный листинг приложения представлен в приложении А.