Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ
Институт информационных технологий

Факультет компьютерных технологий

Кафедра информационных систем и технологий

К защите допустить:

Заведующий кафедрой ИСиТ

_____А.И. Парамонов

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к дипломному проекту
на тему

МОБИЛЬНОЕ ПРИЛОЖЕНИЕ ДЛЯ ОПРЕДЕЛЕНИЯ ЗАБОЛЕВАНИЙ КОЖИ С ИСПОЛЬЗОВАНИЕМ ЛАМПЫ ВУДА

БГУИР ДП 1-40 01 01 01 086 ПЗ

Студент	Ю.А. Лазарева
Руководитель	В.А. Сицко
Консультанты:	
от кафедры ИСиТ	В.А. Сицко
по экономической части	В.Г. Горовой
Нормоконтролер	А.С. Шелягович
Рецензент	

Минск 2025

РЕФЕРАТ

МОБИЛЬНОЕ ПРИЛОЖЕНИЕ ДЛЯ ОПРЕДЕЛЕНИЯ ЗАБОЛЕВАНИЙ КОЖИ С ИСПОЛЬЗОВАНИЕМ ЛАМПЫ ВУДА / Ю.А. Лазарева. – Минск : БГУИР, 2025, – п.з. – 78 с., чертежей (плакатов) – 6 л. формата А1.

Объектом исследования является мобильное приложение для определения заболеваний кожи с помощью лампы Вуда.

Цель работы – разработка мобильного приложения для диагностики кожных заболеваний с использованием лампы Вуда. Целью является создание удобного инструмента для пользователей, который позволит быстро и точно определять наличие кожных патологий без необходимости посещения врача при каждом подозрении.

Разработка данного программного средства обеспечит кроссплатформенность, так как будет доступно на платформах Android и IOS, что увеличит охваты его использования, хранение данных в Firebase, которое является набором инструментов и сервисов, используемых для разворачивания серверной части со своей логикой и авторизацией пользователей, а также реализацию нейросети на Python с использованием библиотеки компьютерного зрения OpenCV для обработки поступающих изображений и определения заболевания кожи согласно излучаемому цвету.

Проведен анализ аналогичных решений для диагностики кожи.

Разрабатываемое мобильное приложение призвано ускорить и облегчить процесс диагностирования кожных заболеваний на ранних стадиях без участия специалиста.

Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Институт информационных технологий

Факультет КТ Кафедра ИСиТ

Специальность 1-40 01 01 Специализация 01

УТВЕРЖДАЮ

А.И.Парамонов

« 24 » сентября 2024 г.

ЗАДАНИЕ
по дипломному проекту студента

Лазарева Юлия Алексеевна

(фамилия, имя, отчество)

1. Тема проекта: **Мобильное приложение для определения заболеваний кожи с использованием лампы Вуда**

утверждена приказом по университету от « 24 » сентября 2024 г. № 112-и

2. Срок сдачи студентом законченной работы 23 декабря 2024 года

3. Исходные данные к проекту Тип операционной системы – ОС Android 6 и выше; Язык программирования – JavaScript, Python; клиент-серверная архитектура. Перечень выполняемых функций: регистрация и авторизация пользователя, загрузка изображения, сканирование, просмотр списка заболеваний исходя из свечения, определение заболевания по цвету свечения.
Назначение разработки: диагностирование различных заболеваний кожи с помощью использования мобильного приложения

4. Содержание пояснительной записки (перечень подлежащих разработке вопросов):

Введение
1 Анализ предметной области
2 Моделирование предметной области
3 Проектирование и разработка программного средства
4 Тестирование программного средства
5 Руководство по эксплуатации программного средства
6 Технико-экономическое обоснование разработки и реализации мобильного приложения для определения заболеваний кожи с использованием лампы Вуда на массовом рынке
Заключение
Список использованных источников
Приложение А. Код программного средства

5. Перечень графического материала (с точным указанием наименования) и обозначения вида и типа материала):
Диаграмма вариантов использования. Плакат – формат А1, лист 1.
Диаграмма деятельности. Плакат – формат А1, лист 1.

Диаграмма развертывания. Плакат – формат А1, лист 1.

Преобразование RGB в HSV. Схема алгоритма – формат А1, лист 1.

Обработка изображения нейросетью. Схема алгоритма – формат А1, лист 1.

Создания истории заболеваний. Схема алгоритма – формат А1, лист 1.

6. Содержание задания по технико-экономическому обоснованию

Технико-экономическое обоснование разработки и реализации мобильного приложения для определения заболеваний кожи с использованием лампы Вуда на массовом рынке

Консультанты по дипломному проекту (с указанием разделов, по которому они консультируют):

Сицко В.А – разделы 1-5;

Горовой В.Г. – раздел 6 (технико-экономическое обоснование);

Шелягович А.С. – нормоконтроль.

ПРИМЕРНЫЙ КАЛЕНДАРНЫЙ ГРАФИК ВЫПОЛНЕНИЯ ДИПЛОМНОГО ПРОЕКТА

Наименование этапов дипломного проекта	Объём готовности проекта в %	Срок выполнения этапа	Примечание
Первая опроцентовка (введение, разделы 1-3)	30%	16.09.24 – 25.10.24	Консультант от кафедры
Вторая опроцентовка (разделы 3-5)	60%	26.10.24 – 30.11.24	Руководитель проекта
Третья опроцентовка (разделы 5-6, заключение, список использованных источников)	90%	01.12.24 – 18.12.24	Руководитель проекта
Консультации по оформлению графического материала и пояснительной записки, нормоконтроль	–	С 01.10.24 (согласно графику)	Нормоконтролёр
Итоговая проверка готовности дипломного проекта на заседании рабочей комиссии кафедры ИСиТ и допуск к защите в ГЭК	100%	С 23.12.24 (согласно графику)	Председатель рабочей комиссии
Рецензирование дипломного проекта	100%	С 26.12.24 (согласно распоряжению)	Рецензент
Защита дипломного проекта	100%	С 18.01.25 (согласно приказу)	ГЭК

Дата выдачи задания 16 сентября 2024 г. Руководитель /В.А. Сицко/

Задание принял к исполнению _____ /Ю.А. Лазарева/

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Тема дипломного проекта посвящена разработке мобильного приложения для определения заболеваний кожи с помощью использования лампы Вуда. В современном мире люди все также часто встречаются с различного рода болезнями, которые влияют, как на внутренние органы, так и на наружные, такие как кожа, которая в свою очередь является самым большим человеческим органом и требует к себе должного внимания. Кожные заболевания являются одной из самых распространенных проблем со здоровьем, с которыми сталкиваются люди всего мира. Диагностика и лечение кожных заболеваний являются важными аспектами заботы о здоровье, но часто требуют похода к врачу и проведения сложных процедур. Мобильное приложение для определения заболеваний кожи с использованием лампы Вуда может стать инновационным решением для людей, страдающих кожными проблемами.

Лампа Вуда – медицинский световой прибор, используемый для выявления заболеваний кожи, волос, ногтей. Метод диагностики базируется на свойстве волос и кожных покровов, пораженных грибком или содержащих токсические микроэлементы, давать ярко-зеленое свечение при облучении его коротковолновым ультрафиолетом. Лампа излучает ультрафиолетовое свечение, благодаря которому можно заметить патологии, невидимые обычным глазом. Обследование лампой Вуда проводится в затемненном помещении. Использование лампы Вуда – быстрый и эффективный способ исследования различных заболеваний кожи или ее придатков, отличающийся точностью результатов. Благодаря тому, что это бесконтактная диагностика, пациент не испытывает никакого дискомфорта во время ее проведения. А компактные размеры лампы помогают использовать ее для осмотра кожи на разных участках тела. Лампа Вуда является уникальным инструментом, используемым для диагностики различных кожных заболеваний путем подсветки кожи ультрафиолетовым светом. Лампа Вуда, также известная как лампа ультрафиолетовой (УФ) диагностики, является эффективным инструментом в дерматологии. Принцип работы лампы Вуда основан на способности УФ-света обнаруживать определенные пигменты в коже [1].

При помощи мобильного приложения пользователи смогут легко проводить самостоятельную диагностику кожи, определяя возможные проблемы. Приложение может анализировать фотографии кожи, сделанные с помощью лампы Вуда, и предоставлять пользователю информацию о возможных диагнозах. Такой подход к диагностике кожных заболеваний не только упростит доступ к необходимой информации и поможет людям быстрее выявлять проблемы со здоровьем кожи, но также сэкономит время и деньги.

Мобильное приложение будет ориентировано на анализ загруженной фотографии или прямого сканирования пораженного участка кожи, опираясь на алгоритмы обработки и анализа полученного изображения посредством

машинного обучения. Например, анализ загруженного изображения на котором в свете лампы Вуда видна область, светящаяся зеленым цветом, говорящая о наличии грибковых заболеваний. Приложение будет обрабатывать цвета, которые отображаются на коже при наведении ультрафиолетового света с лампы Вуда.

В основной функционал мобильного приложения будет входить возможность регистрации, авторизации, сканирования нездорового участка кожи, возможность загрузки фотографии, определение заболеваний кожи исходя из свечения пораженной области кожи. При входе в приложение пользователь увидит страницу загрузки после которой попадает в форму регистрации. После удачной регистрации создается учетная запись, но для входа в нее необходимо ввести логин и пароль в соответствующие поля. Затем появляется экран с выбором дальнейших действий: сканирование, загрузка изображения, просмотр значения свечений. Если пользователь выбирает сканирование, то переходит в режим сканирования. Для того, чтобы отсканировать необходимо будет поднести данный участок кожи под лампу Вуда и отсканировать цвет свечения, после которого благодаря машинному обучению приложение обработает цвет и выведет на экран, какое это конкретно заболевание. Принцип загрузки изображения такой же, как и при сканировании, что для получения результата нужно будет посветить на беспокоящий участок кожи ультрафиолетовым светом, благодаря которому приложение обработает изображение и выдаст результат. Также будет кнопка для просмотра значения свечений, после нажатия на которую откроется экран с цветами свечений заболеваний и их названия. Такое исследование кожи позволит значительно упростить и ускорить процесс определения вероятного заболевания кожи после которого будет более понятно к какому специалисту идти на обследование.

Пояснительная записка к дипломному проекту будет состоять из ключевых разделов, каждый из которых направлен на детальное описание всех этапов разработки и внедрения программного средства. В первом пояснительной записки «Анализ предметной области» будут рассмотрены основные подход и концепции, которые напрямую связаны с исследованиями заболеваний кожи одним из множества вариантов – определения с использованием ультрафиолетовой лампы. Проблемы с кожей довольно распространенная проблема, с которой люди сталкиваются довольно часто. Определение заболеваний кожи требует специальных навыков и опыта в области дерматологии, но для первоначальной диагностики данное программное средство подходит. Такой раздел, как «Моделирование предметной области» будет посвящен разработке логических и концептуальных моделей, которые будут отображать взаимодействие компонентов системы и структуру данных. Пользователи смогут сделать снимки пораженной области кожи с помощью камеры смартфона, после чего приложение проанализирует и обработает изображение благодаря тому, что использовались при разработке современные технологии, такие как машинное

обучение. В третьем разделе «Проектировании и разработке программного средства» описаны используемые технологии и архитектура, которые используются для разработки удобного и интуитивно понятного интерфейса, который позволит пользователям легко использовать приложение. Также будет описана интеграция алгоритма машинного обучения, который поможет определить заболевание кожи на основе анализа изображения. Далее идет не менее важный раздел, как «Тестирование программного средства» после завершения разработки приложения необходимо провести тестирование, просмотреть результаты проверки работоспособности системы, его функциональности и корректности работы. Обязательно используя при этом функциональное и нефункциональное тестирование. После успешного завершения тестирования приложения необходимо создать подробное руководство по его использованию, чтобы пользователи могли без труда пользоваться всеми его возможностями. Для этого есть раздел, как «Руководство по эксплуатации программного средства» в котором изложены инструкции и рекомендации по настройке и использованию приложения для конечных пользователей. Разработка мобильного приложения для определения заболеваний кожи может стать выгодным инвестиционным проектом. Пользователи смогут получить быструю и точную диагностику заболеваний кожи, что позволит им своевременно начать лечение и избежать осложнений. Таким образом, разработка приложения будет востребована на рынке здравоохранения и приносить прибыль разработчикам. Для этого в пояснительной записке присутствует раздел «Технико-экономического обоснования разработки программного средства» благодаря которому можно будет рассмотреть, какие затраты уйдут на разработку, внедрение и поддержку системы, оценив при этом экономическую эффективность.

Таким образом цель дипломного проекта заключается в создании мобильного приложения, которое будет полезным инструментом для самостоятельной диагностики заболеваний кожи с помощью использования лампы Вуда. Оно будет помогать пользователям более осознанно относиться к своему здоровью и выявлять проблемы, что способствует их успешном своевременному решению, которое не даст усугубить ситуацию.

Данный дипломный проект выполнен мной лично, проверен на заимствования, процент оригинальности составляет 91% [2].

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Описание предметной области

Предметная область дипломного проекта – это создание мобильного приложения, которое сможет помочь пользователям определить заболевания кожи с помощью использования лампы Вуда. Лампа Вуда – это ультрафиолетовая лампа, которая помогает визуализировать различные патологии кожи, такие как грибковые инфекции, вирусные болезни, дерматиты и другие.

Для реализации данного проекта необходимо будет разработать мобильное приложение, которое будет обрабатывать изображения, полученные с помощью лампы Вуда, и проводить анализ на наличие различных заболеваний кожи. Приложение должно обеспечивать удобный и понятный интерфейс для пользователя, а также точные и надежные результаты анализа.

В рамках анализа предметной области необходимо изучить особенности работы лампы Вуда, принципы диагностики кожных заболеваний с ее помощью, а также методы обработки и анализа изображений наличия заболеваний кожи. Также стоит уделить внимание требованиям к безопасности использования ультрафиолетового излучения, чтобы избежать возможных негативных последствий для здоровья пользователей.

В итоге, успешная реализация данного проекта позволит пользователям быстро и удобно определять различные заболевания кожи с помощью мобильного устройства и лампы Вуда, что может значительно упростить процесс диагностики и обращения к врачу для получения необходимого лечения.

1.2 Обоснование актуальности разработки ПС

Развитие мобильных технологий и их все более широкое использование в медицинских целях предоставляют отличные возможности для создания удобных и доступных инструментов для диагностики заболеваний. Мобильные приложения, осуществляющие анализ состояния кожи с помощью лампы Вуда, могут значительно упростить процесс определения различных кожных заболеваний у пользователей. Определение заболеваний кожи с помощью лампы Вуда имеет ряд преимуществ перед традиционными методами диагностики. Лампа Вуда позволяет увидеть различные изменения в структуре кожи, которые не всегда могут быть видны невооруженным глазом. Таким образом, использование лампы Вуда в мобильном приложении может быть эффективным и удобным способом самостоятельной диагностики.

Учитывая растущую проблему кожных заболеваний и необходимость их раннего выявления, разработка мобильного приложения для определения заболеваний кожи с помощью лампы Вуда является актуальной и важной

задачей. Помимо этого, такое приложение может облегчить доступ пользователей к качественной медицинской помощи и сэкономить время на диагностике.

С учетом все возрастающей популярности мобильных устройств и приложений, разработка мобильного приложения для диагностики заболеваний кожи становится еще более целесообразной. Пользователи могут использовать такие приложения в любое удобное время и место, что делает процесс диагностики более удобным и эффективным.

В современном мире, где забота о здоровье становится все более важной, создание мобильного приложения для определения заболеваний кожи с помощью лампы Вуда является ключевым шагом в улучшении доступности медицинской информации и повышении осведомленности пользователей о состоянии своего здоровья. Такой инновационный подход к диагностике может значительно улучшить качество медицинского обслуживания и помочь сохранить здоровье кожи.

1.3 Анализ методов, способов, подходов, методик, технологий

Для разработки мобильного приложения для определения заболеваний кожи с использованием лампы Вуда использовались популярные методы, способы, подходы, методики и технологии.

При разработке программного средства применялась инновационная методология программного обеспечения – Agile, которая основана на принципе постепенного развития продукта через короткие циклы.

В данной системе ключевыми элементами являются:

– список заболеваний кожи и соответствующих им цветов;

– пользователь.

Архитектура описываемого программного средства является клиент-серверной. Для реализации клиентской части в Android были использованы технологии для мобильных приложений, такие как разработка на платформе React Native с использованием языка программирования JavaScript. Серверная часть была разработана с использованием кроссплатформенной среды разработки Node.js. Также использовалось Expo – это платформа и набор инструментов, которые упрощают разработку мобильных приложений с использованием React Native. Expo предоставляет готовые решения для многих задач, таких как управление зависимостями, сборка приложений, работа с уведомлениями, камера и многое другое.

Основные компоненты Expo:

– Expo CLI – инструмент командной строки, который позволяет создавать, запускать и управлять проектами React Native с помощью Expo;

– React Native SDK – библиотека компонентов и API, которая расширяет возможности React Native, предоставляя доступ к функциям устройства (например, к камере) через JavaScript;

– Expo Go – приложение, которое можно установить на iOS или Android,

чтобы тестировать приложения без необходимости их сборки и установки на устройство;

– Build Service – сервис для автоматической сборки приложений под разные платформы (iOS и Android);

– Over-the-Air Updates – возможность обновлять приложение без необходимости его пересборки и повторного выпуска в магазины приложений.

Expo было выбрано, так как включает в себя удобные для конкретной разработки свойства:

– простота настройки – Expo значительно упрощает процесс создания и запуска проекта;

– меньше времени на настройку среды разработки;

– быстрая разработка и тестирование;

– готовые модули и библиотеки;

– поддержка Over-the-Air обновления;

– удобство для начинающих разработчиков.

Expo – это фреймворк для быстро развивающихся нативных React приложений. С Expo разработчики могут создавать React Native приложения, не испытывая при этом проблем с установкой и настройкой зависимостей программного обеспечения таких как Android Studio, Xcode и других инструментов, которые необходимы для разработки и запуска React Native приложений. Expo хорошо подходит для прототипирования, небольших проектов или приложений, где не требуется глубокая кастомизация нативных функций. В Expo есть некоторые ограничения:

– приложения не поддерживают фоновое выполнение кода;

– приложения ограничены нативными API;

– привязывает к использованию его инструментов;

– автономные исполняемые файлы приложений могут быть построены только при наличии подключения к сети [3].

Таким образом Expo – отличный выбор для быстрого старта и удобной разработки мобильных приложений на базе React Native. Также при разработке использовался высокоуровневый язык программирования, отличающийся эффективностью, простотой и универсальностью в использовании – Python. А также этот язык широко применяется в машинном обучении с использованием библиотеки алгоритмов компьютерного зрения, обработки изображений и численных алгоритмов общего назначения – OpenCV. Применялось и использование библиотеки OpenCV для анализа, которая предоставляет мощные инструменты для обработки и анализа изображений. С ее помощью можно проводить распознавание текстур, цветов, форм и других признаков на коже для определения возможных заболеваний. Для серверной части использовался набор инструментов и сервисов для разработки мобильных и веб-приложений от Google – Firebase. Firebase – это платформа Google, помогающая быстро разрабатывать качественные мобильные и веб-приложения, привлекать новых пользователей и повышать доходы. Ее интегрированные инструменты, такие как сервер для мобильных

приложений, средства аналитики, можно использовать по отдельности или в любых сочетаниях [4].

Была обучена модель посредством разработанных алгоритмов машинного обучения, нацеленных на распознавания заболеваний, которые были получены на основе данных с сканирования и фотографий для распознавания видимых симптомов в образе различных проявлений их на коже.

Комбинация указанных методов, способов и технологий позволила разработать современное и функциональное мобильное приложение для определения заболеваний кожи с использованием лампы Вуда.

1.4 Анализ существующих аналогов и прототипов разрабатываемого ПС с выделением их достоинств и недостатков

На данный момент есть схожее по концепции и принципу работы мобильное приложение «Miiskin», специализирующиеся на распознавании доброкачественных и злокачественных новообразований, таких как родинки. Принцип работы заключается в алгоритме по поиску подобных изображений в базе данных. Создание такого продукта позволило формировать снимки новообразований, передавать данные снимки и определенные пользовательские данные в кластер для обработки и получения результата выполнения алгоритма.

Рисунок 1.1 – Главный экран

На главном экране мобильного приложения «Miiskin» представлена краткая информация и панель навигации вверху экрана.

Если перейти по второй иконки слева в панели навигации, то можно начать непосредственно работу с приложением по его прямому назначению.

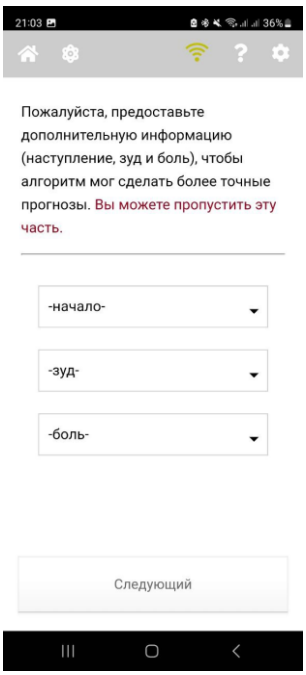


Рисунок 1.2 – Экран для выбора симптомов

Данный экран предоставляет выбор основных симптомов с возможностью выбора подходящих вариантов из выпадающих списков.

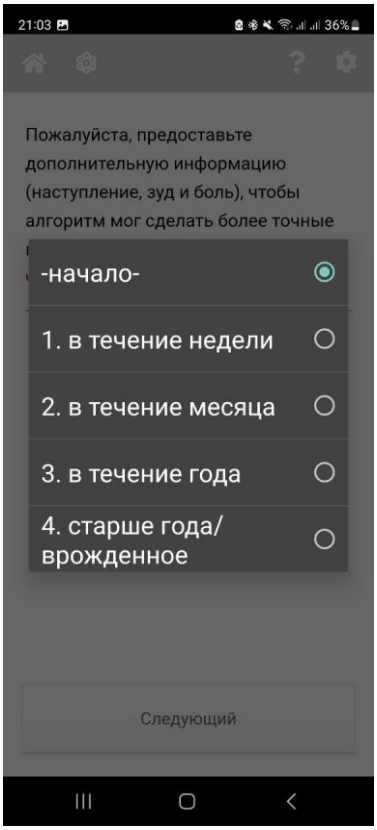


Рисунок 1.3 – Выдающий список с возможностью выбора начала проявления первых симптомов

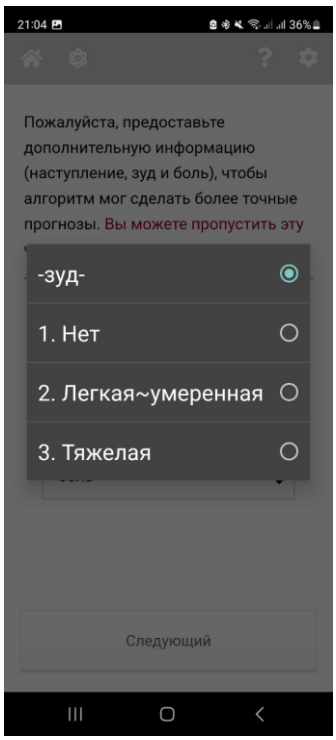


Рисунок 1.4 – Выбор интенсивности зуда

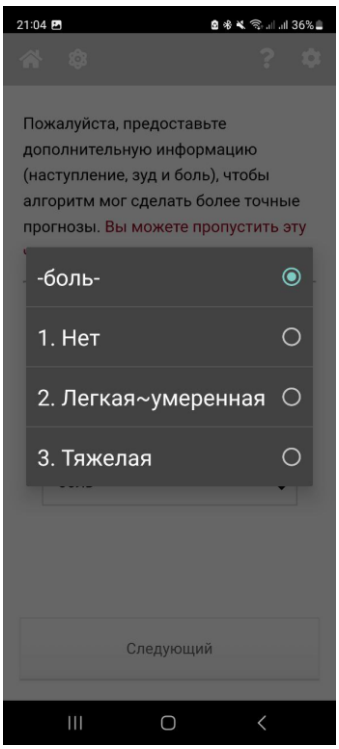


Рисунок 1.5 – Выбор интенсивности болевых ощущений

Симптомы можно, как выбирать, так и нет это влияет только на точность итогового результата. После выбора симптомов открывается экран для загрузки изображения пораженного участка кожи с галереи устройства, либо же непосредственно с камеры устройства.

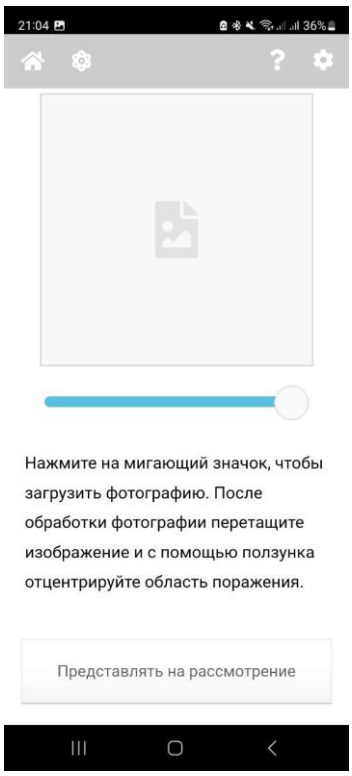


Рисунок 1.6 – Возможность загрузки изображения

На рисунке 1.7 – 1.9 будут показаны возможные варианты загрузки обрабатываемого изображения.

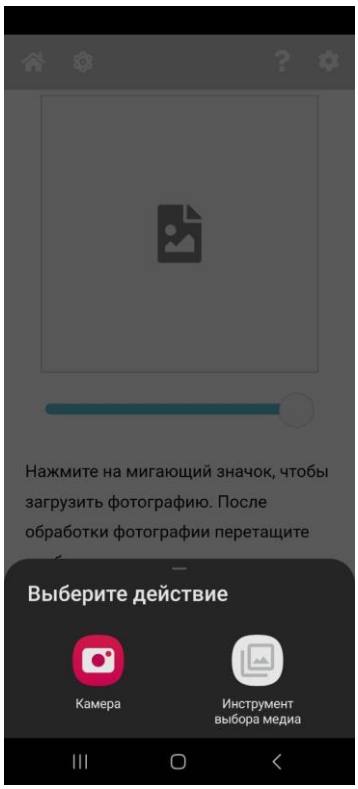


Рисунок 1.7 – Выбор варианта загрузки изображения

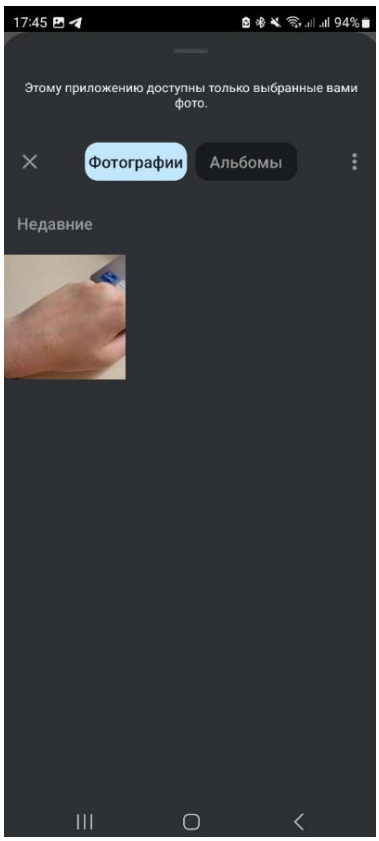


Рисунок 1.8 – Выбор изображения из галереи устройства

Рисунок 1.9 – Загрузка изображения через камеру смартфона

После загрузки обрабатываемого изображения любым из представленных способов есть возможность приблизить, либо же отдалить полученное фото на предмет того, чтобы был виден общий план пораженного участка кожи или же конкретная область.

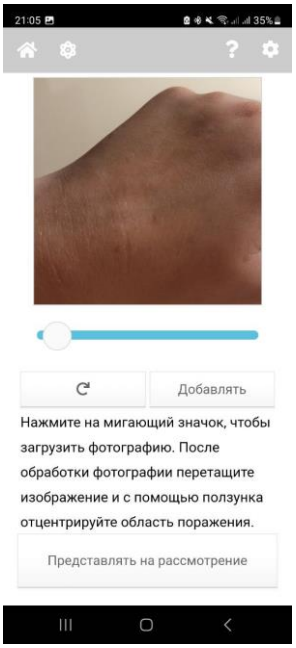


Рисунок 1.10 – Загруженное изображение с возможностью переснять или добавить дополнительное фото

На данном экране выполняется процесс обработки изображения для получения его результата.

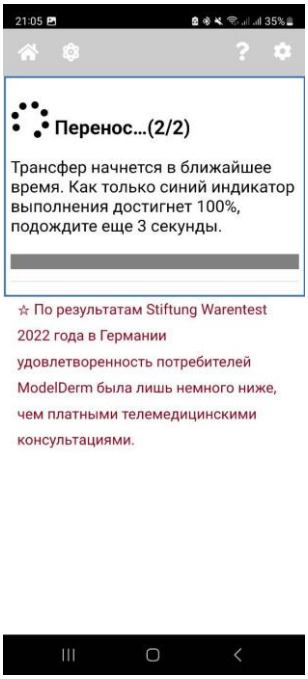


Рисунок 1.11 – Обработка и поиск результатов

Рисунок 1.12 – Итоговый результат

Из плюсов данного программного средства можно отметить, что таких приложений крайне мало в настоящее время. Смысл в них определенно есть, так как можно на ранних этапах диагностировать доброкачественные и злокачественные новообразования.

Но во время использования приложения «Miiskin» возникло недопонимание итогов сканирования, так как приложение выдало просто похожие изображения с различными видами родинок, пигментных пятен, грибковых заболеваний, инфекционных поражений кожи и аллергии. Как было заверено, что приложение используется для определения доброкачественности и злокачественности новообразований, а после использования стало понятно, что не хватает качественного анализа получаемого изображения и определения того для чего было разработано приложение.

С большой вероятностью разработчик доработает до идеала мобильное приложение «Miiskin», так как смысл в нем есть и актуальности на сегодняшний день тоже, учитывая, что в современном мире люди становятся все более загруженными своим делами и заботами и простой поход к онкодерматологу, что в государственное медицинское учреждение, что в частное занимает довольно большой промежуток времени и не факт, что смогут сразу дать результат. Но с распространением похожих программных средств этот процесс можно будет в разы сократить. Основной момент, что результаты, получаемые в данных приложениях должны соответствовать реальным данным по тем или иным заболеваниям.

1.5 Выбор и обоснование языков программирования, фреймворков, библиотек, СУБД для разработки ПС

В разрабатываемом мобильном приложение для определения заболеваний кожи с помощью использованием лампы Вуда были применены современные языки программирования, фреймворки, библиотеки и СУБД, как для клиентской части приложения, так и для серверной.

В ходе разработки клиентской части программного средства были использована язык программирования JavaScript и фреймворк React Native. JavaScript – это кроссплатформенный, интерпретируемый язык программирования, который используется для создания интерактивных веб-страниц и приложений. Функционал JavaScript может улучшить удобство взаимодействия пользователя с веб-сайтом: от обновления ленты новостей в социальных сетях и до отображения анимации и интерактивных карт. JavaScript был создан в 1995 году Бренданом Эйхом, когда тот еще был инженером в Netscape. Javascript – динамический скриптовый язык программирования высокого уровня. Он отличается мультипарадигменностью. Речь идет о поддержке функционального, императивного, событийно-ориентированного стилей. Чаще всего язык используется для создания интерактивных веб-страниц и приложений [5]. Данный язык программирования является интерпретируемым, а не компилируемым. Код на таком языке не нужно компилировать, так как его пишут и сразу передают программе-интерпретатору, которая сразу же его выполняет. Это ускоряет разработку, но в тоже время требует запуска только в связке с интерпретатором. JavaScript поддерживает несколько парадигм программирования, что позволяет разработчикам выбирать наиболее подходящий подход к решению конкретных задач. Одним из ключевых принципов языка является объектно-ориентированное программирование, где основными элементами являются объекты и их взаимодействие. Парадигмы, которые входят:

– объектная;
– функциональная;
– императивная.

Объектная парадигма в JavaScript позволяет создавать и использовать объекты, которые содержат в себе свойства и методы. Объекты могут быть созданы как литералы или с использованием конструкторов. ООП в JavaScript также поддерживает наследование, инкапсуляцию и полиморфизм.

Функциональное программирование в JavaScript базируется на работе с функциями как элементами первого класса. Функции могут быть переданы как аргументы другим функциям, присвоены переменным и возвращены из других функций. Функциональные программы в JavaScript обычно используются для обработки данных и создания реактивных интерфейсов.

Императивная парадигма в JavaScript предполагает последовательное выполнение инструкций, которые определяют, как должна быть выполнена

программа. Этот подход управления состоянием программы подразумевает использование циклов, условий и изменения значений переменных.

JavaScript позволяет комбинировать различные парадигмы программирования для достижения оптимальных результатов. Разработчики вольны выбирать тот подход, который наилучшим образом подходит к решению конкретной задачи и обеспечивает легкость сопровождения и расширения кода. Все эти парадигмы существуют параллельно и дополняют друг друга, делая JavaScript мощным инструментом разработки приложений для веба.

Также плюсом является, что данный язык без строгой типизации. Есть ЯП со статической типизацией, где если создать переменную, то сначала нужно обязательно задать ей тип, к примеру строку или число и при этом получается, что ничего кроме заданного типа данных хранить уже нельзя. В JavaScript типизация динамическая, то есть в переменную можно положить что угодно. Это облегчает написание кода, но в тоже время может вызвать ошибки.

Область применения JavaScript довольно широка и этот язык программирования применяют даже в мобильной разработке, используя фреймворк React Native. React Native – это фреймворк для создания нативных мобильных приложений на JavaScript с использованием библиотеки React JavaScript; машинный код React компилируется в реальные нативные компоненты. Первоначально React Native использовался для создания пользовательских интерфейсов для веб-приложений. С тех пор он эволюционировал и теперь может также использоваться для создания серверных и мобильных приложений. В React Native есть много возможностей, помимо поддержки и открытого исходного кода со стороны Facebook также поддерживает огромное сообщество мотивированных людей. Группы Facebook, насчитывающие миллионы пользователей, работают на базе React Native, а также Face book Ads Manager. Airbnb, Bloomberg, Tesla, Instagram, Ticketmaster, SoundCloud, Uber, Walmart, Amazon и Microsoft – вот некоторые из других компаний, которые инвестируют в React Native или используют его в производстве.

С помощью React Native разработчики могут создавать собственные представления и получать доступ к компонентам, зависящим от платформы, с помощью JavaScript. Это отличает React Native от других гибридных приложений, таких как Cordova и Ionic, которые упаковывают веб-представления, созданные с использованием HTML и CSS, в собственное приложение. Вместо этого React Native использует JavaScript и компилирует его в настоящее нативное приложение, которое может использовать API и компоненты, зависящие от платформы. В альтернативах, таких как Xamarin, используется тот же подход, но приложения Xamarin создаются с использованием C#, а не JavaScript. Многие веб-разработчики имеют опыт работы с JavaScript, что помогает упростить переход от веб-разработки к разработке мобильных приложений.

Выбор React Native в качестве фреймворка для мобильных приложений имеет много преимуществ. Поскольку приложение напрямую отображает собственные компоненты и API, скорость и производительность намного выше, чем при использовании гибридных платформ, таких как Cordova и Ionic.

С помощью React Native пишутся целые приложения, используя один язык программирования: JavaScript. С помощью React Native пишутся целые приложения, используя один язык программирования: JavaScript. Который можно использовать повторно, тем самым сокращая время, необходимое для отправки кроссплатформенного приложения. А нанять и найти качественных разработчиков JavaScript гораздо проще и дешевле, чем разработчиков Java, Objective C или Swift, что в целом делает процесс менее затратным [6].

Кроссплатформенность и повторное использование кода. Главный плюс React Native то, что благодаря нему можно не писать код одного и того же приложения с нуля. Переиспользуемость кода в отдельных проектах может составлять до 90%: это значит, что 90% кода из Android-проекта можно без особых изменений перенести в iOS-версию приложения. Присутствует скорость и быстрота разработки благодаря тому, что код можно использовать повторно, создать приложение с помощью React Native можно довольно быстро. А еще можно легко перенести его на другую мобильную ОС. Да, сборка для Android не будет работать в iOS, но изменений для создания iOS-версии нужно минимум. Это ускоряет разработку и поддержку приложений. Из-за обновления функции CodePush приложения можно обновлять без промежуточного этапа в виде загрузки обновлений в магазин. Пользователь устанавливает приложение на свой смартфон, а когда выходит обновление, приложение качает и устанавливает его само – ему для этого не нужно связываться с Google Play или App Store. Для разработчика это означает меньше сложностей с загрузкой обновлений в маркеты и быструю доставку их до пользователя.

Высокая экономическая эффективность при разработке приложения с помощью React Native. За то же время и тот же бюджет компания фактически получает две версии приложения для разных систем. А при нативной разработке – только одну, а, чтобы создать вторую, придется переписывать всё с нуля.

Присутствует большой охват аудитории, учитывая тот момент, что этот фреймворк одновременно хорошо подходит для Android и IOS разработки, то есть захватывает разные сегменты. Соответственно приложением сможет пользоваться больше людей, тем самым у его владельца будет больше клиентов. React Native имеет довольно широкое и активное сообщество – комьюнити. Это популярная технология, а еще для фреймворка постоянно выпускают новые модули.

Для серверной части использовалась платформа Node.js. Node.js (Node) – это платформа с открытым исходным кодом для работы с языком JavaScript, построенная на движке Chrome V8. Она позволяет писать серверный код для веб-приложений и динамических веб-страниц, а также программ командной

строки. В основе платформы – событийно-управляемая модель с неблокирующими операциями ввода-вывода, что делает ее эффективной и легкой. Node.js – это среда выполнения кода JavaScript вне браузера, которая позволяет писать серверный код для веб-страниц и веб-приложений, а также для программ командной строки. Node.js – не отдельный язык программирования, а платформа для использования JavaScript на стороне сервера. С помощью платформы можно работать с файлами, сетью, базами данных и другими системными ресурсами на сервере.

Node.js работает на движке V8, транслирующем JavaScript в машинный код. Простыми словами, Node.js – это приложение на C++, которое получает на входе код JavaScript и выполняет его. Чтобы взаимодействовать с устройствами ввода-вывода на компьютере, в платформе предусмотрен собственный интерфейс на C++. Таким образом, платформа превращает специализированный скриптовый язык JavaScript в язык общего назначения, поэтому на Node.js можно писать любые компьютерные программы. Платформа позволяет пользоваться единым языком JavaScript для написания кода и на стороне клиента (Frontend), и на сервере (Backend). Эти возможности Node.js важны для разработки приложений реального времени, которые основаны на событиях.

Node.js лежит в основе Internet of Things, или просто IoT. Платформа помогает управлять приборами и создавать серверы, способные одновременно обрабатывать большое количество запросов.

JavaScript-код, который выполняется в среде Node.js, может быть в несколько раз быстрее, чем написанный на языках вроде Ruby или Python. В Node.js используется модель асинхронного программирования. Модель позволяет продолжить обработку других задач, не дожидаясь завершения передачи данных. Когда требуется выполнить операцию ввода-вывода вроде доступа к файловой системе или базе данных, Node.js не блокирует главный поток ожиданием результатов. Платформа инициирует ее выполнение и продолжает выполнять другие задачи, пока результаты предыдущей операции не будут получены.

В Node.js выполняется код, который написан на JavaScript. Это означает, что frontend-разработчики, которые уже используют JavaScript в браузере, могут писать и клиентский, и серверный код на привычном языке программирования, не изучая инструмент с нуля. В Node.js можно быстро переходить на новые стандарты ECMAScript по мере их реализации. Новые возможности языка становятся доступны сразу после установки поддерживающей их версии Node.js.

Большое количество модулей и библиотек. Экосистема Node.js стремительно развивается благодаря менеджеру пакетов NPM. Он содержит более 500 000 модулей и библиотек open-source, которые находятся в свободном доступе. Node.js работает на JavaScript-движке V8 от Google. V8 – движок JavaScript с открытым исходным кодом, распространяемый по лицензии BSD [7].

Разработка диплома происходила также с использованием OpenCV (Open Source Computer Vision Library) – библиотека программного обеспечения для компьютерного зрения и машинного обучения с открытым исходным кодом. OpenCV был создан для обеспечения общей инфраструктуры для приложений компьютерного зрения и ускорения использования машинного восприятия в коммерческих продуктах. Будучи лицензированным продуктом Apache 2, OpenCV упрощает использование и модификацию кода [8].

OpenCV может использоваться везде, где нужно компьютерное зрение. Эта отрасль IT работает с технологиями, которые позволяют устройству «увидеть», распознать и описать изображение. Компьютерное зрение дает точную информацию о том, что изображено на картинке, с описанием, характеристиками и размерами (с определенной степенью достоверности).

Также библиотека работает с машинным обучением – отраслью, которая обучает алгоритмы действовать тем или иным образом.

OpenCV применяется:

– в робототехнике – для ориентирования робота в пространстве, распознавания объектов и взаимодействия с ними;

– медицинских технологиях – для создания точных методов диагностики, например, 3D-визуализации органа при МРТ;

– промышленных технологиях – для автоматизированного контроля качества, считывания этикеток, сортировки продуктов и пр.;

– безопасности – для создания «умных» камер видеонаблюдения, которые реагируют на подозрительные действия, для считывания и распознавания биометрии;

– мобильной фотографии – для создания бьюти-фильтров, изменяющих лицо приложений;

– на транспорте – для разработки автопилотов.

Для хранения и работы с изображениями OpenCV использует векторы и скаляры, матрицы и диапазоны. Они позволяют проводить математические преобразования, ориентироваться по изображению и выполнять множество других действий.

Для распознавания элементов в OpenCV используются очертания объектов, сегментация по цветам, встроенные методы распознавания, которые можно настраивать в зависимости от объекта и чувствительности алгоритма.

Сейчас структура OpenCV – это множественные модули для разных целей:

– хранения математических функций и вычислений, алгебры и структур данных;

– хранения моделей для машинного обучения;

– ввода и вывода картинок или видео, чтения и записи в файл;

– обработки изображения;

– распознавания примитивов;

– детектирования объектов – лиц, предметов и других;
– отслеживания и анализа движений на видео;
– обработки трехмерной информации;
– ускорения работы библиотеки;
– хранения устаревшего или еще не готового кода и других.

Каждый модуль узко специализирован. Их не нужно скачивать отдельно: в пакет установки включена вся основная функциональность библиотеки.

Преимущества OpenCV в активном сообществе, так как для OpenCV для Python, JavaScript, Ruby и других языков программирования используют по всему миру, в том числе в Google и Microsoft. Поэтому вокруг библиотеки существует активное сообщество. Это полезно и для новичков, и для экспертов. Документация по OpenCV представлена на нескольких языках, в том числе на русском. Скачать полный исходный код последней версии можно на этой странице.

Бесплатный доступ OpenCV распространяется по бесплатной лицензии и для учебного, и для коммерческого использования. У библиотеки открытый исходный код, который может просмотреть любой программист. Это дает большую гибкость в работе с OpenCV и позволяет самостоятельно узнать, как реализована та или иная функция.

Огромное обилие алгоритмов в OpenCV включает более 2500 инструментов и алгоритмов компьютерного зрения и машинного обучения. Этого достаточно для решения сложных задач по распознаванию и обработке изображений.

Библиотека работает быстрее масштабного и тяжелого ПО для математических вычислений, такого как Matlab. Поэтому ее можно использовать в ситуациях, когда требуется быстро обработать картинку.

Как и в любой технологии есть ряд недостатков OpenCV:
– отсутствие кодов обработки ошибок;
– ориентированность на большие платформы.

Учитывая то, что каждое изображение состоит из набора пикселей. Пиксель – это строительный блок изображения. Если представить изображение в виде сетки, то каждый квадрат в сетке содержит один пиксель, где точке с координатой (0, 0) соответствует верхний левый угол изображения. К примеру, представим, что у нас есть изображение с разрешением 400x300 пикселей. Это означает, что наша сетка состоит из 400 строк и 300 столбцов. В совокупности в нашем изображении есть 400*300 = 120000 пикселей.

В большинстве изображений пиксели представлены двумя способами: в оттенках серого и в цветовом пространстве RGB. В изображениях в оттенках серого каждый пиксель имеет значение между 0 и 255, где 0 соответствует чёрному, а 255 соответствует белому. А значения между 0 и 255 принимают различные оттенки серого, где значения ближе к 0 более тёмные, а значения ближе к 255 более светлые.

Цветные пиксели обычно представлены в цветовом пространстве RGB (red, green, blue, красный, зелёный, синий), где одно значение для красной компоненты, одно для зелёной и одно для синей. Каждая из трёх компонент представлена целым числом в диапазоне от 0 до 255 включительно, которое указывает как «много» цвета содержится. Исходя из того, что каждая компонента представлена в диапазоне [0,255], то для того, чтобы представить насыщенность каждого цвета, нам будет достаточно 8-битного целого беззнакового числа. Затем объединяем значения всех трёх компонент в кортеж. К примеру, чтобы получить белый цвет, каждая из компонент должна равняться 255: (255, 255, 255). Тогда, чтобы получить чёрный цвет, каждая из компонент должна быть равной 0: (0, 0, 0) [9].

Для работы с библиотекой OpenCV использовался высокоуровневый язык программирования, отличающийся эффективностью, простотой и универсальностью использования, который широко применяется в разработке веб-приложений и прикладного программного обеспечения, а также в машинном обучении и обработке больших данных – Python. Этот язык программирования хорошо применяется в Data Science и машинном обучении, так как эти два направления IT тесно связаны друг с другом. Наука о данных заключается в обработке больших массивов информации из базы данных, а машинное обучение – в разработке компьютерных алгоритмов, способных учиться на ней и делать точные прогнозы. В Data Science используют Python для включения очистки и разметки данных, поиска и обработки статистической информации, ее визуализацию в виде диаграмм, графиков. С помощью библиотеки Python ML классифицируются изображения, тексты, поисковый трафик, осуществляется распознавание лиц и речи, глубинное машинное обучение.

Python – это высокоуровневый язык программирования, отличающийся эффективностью, простотой и универсальностью использования. Он широко применяется в разработке веб-приложений и прикладного программного обеспечения, а также в машинном обучении и обработке больших данных. За счет простого и интуитивно понятного синтаксиса является одним из распространенных языков для обучения программированию.

Python характеризуется такими основными пунктами, как:

– интерпретируемостью;

– Динамическая типизацией;

– является языком высокого уровня;

– объектно-ориентированность.

Интерпретируемость данном языке операторы кода исполняются последовательно с помощью программы-интерпретатора. Если по ходу исполнения программы встречается ошибка, оно сразу же прекращается. Это позволяет Python-разработчику быстро обнаружить и устранить недочеты, но в то же время снижает производительность.

Динамическая типизация – это автоматическое связывание переменной и типа в момент, когда ей присваивается определенное значение. Такой

механизм ускоряет написание программы в различных ситуациях (например, при работе с переменными данными), но повышает вероятность ошибки.

Python является языком высокого уровня и по своему синтаксису, и грамматике близок к естественным языкам. Благодаря этому программисту с его помощью легче описать различные структуры данных и операции, что также ускоряет и упрощает написание кода [10].

Программа, написанная на Python, представляет собой совокупность объектов, каждому из которых присвоены определенный класс и место в иерархии. Таким образом проще управлять процессом программирования, что особенно важно при создании сложных проектов.

Для работы с базами данных был выбран набор инструментов и сервисов, как Firebase. С помощью Firebase можно быстро развернуть бэкенд со своей серверной логикой, подключить базы данных и настроить авторизацию пользователей.

Firebase широко используется разработчиками мобильных приложений, веб-приложений и серверных приложений. Далее будет представлено несколько примеров, где применяется:

– мобильные приложения;
– веб-приложения;
– серверные приложения;
– IoT-устройства;
– корпоративные приложения;
– социальные сети;
– игры;
– безопасность и защита информации.

Использование Firebase заключаются в том, что он просто в использовании при этом имеет богатый функционал, а также высокую производительность и надежность, что не маловажно при разработке мобильного приложения.

1.6 Спецификация требований

1.6.1 Назначение разработки

Разработка мобильного приложения предназначена для помощи пользователям в диагностике различных заболеваний кожи с использованием лампы Вуда. Приложение будет служить инструментом для повышения осведомленности о кожных заболеваниях и содействовать в своевременном обращении за медицинской помощью. Оно нацелено как на специалистов (дерматологов), так и на обычных пользователей, желающих самостоятельно оценить состояние своей кожи.

1.6.2 Перечень основных выполняемых функций

В данном дипломном проекте был выявлен перечень основных функций:

– сканирование кожи – благодаря использованию камеры устройства и

анализа изображений, сделанных под лампой Вуда для выявления возможных кожных заболеваний.

– определение заболеваний – автоматизированная идентификация заболеваний на основе анализа изображений и алгоритмов машинного обучения.

– информация о заболеваниях – предоставление кратких описаний по каждому из определённых заболеваний кожи.

– история – сохранение результатов сканирования.

1.6.3 Входные данные для ПС

Входными данными для программного средства будут являться:

– изображения кожи, полученные с помощью камеры смартфона в условиях освещения от лампы Вуда;

– пользовательские данные при регистрации;

– актуальные базы данных о кожных заболеваниях для сопоставления с полученными изображениями.

Также входными данными будет является значение свечений при конкретных кожных заболеваниях. Основные варианты свечений лампы Вуда при наличии определённых заболеваний и состояний кожи:

– грибковая инфекция – зеленоватое свечение;

– микроспория – желто-зеленое свечение;

– вульгарные угри – оранжево-красное свечение;

– эритразма – кораллово-красное свечение;

– разноцветный лишай – тусклое жёлтое свечение;

– заболевание Фавуса – бледно-серебристое свечение;

– витилиго – молочно-белое свечение;

– заболевание красная волчанка – снежно-белое свечение;

– наличие воспалительного процесса – белое свечение.

– здоровая, неинфицированная кожа без воспалений и поражений – светло-синее свечение [11].

1.6.4 Выходные данные для ПС

Входными данными для программного средства будут являться:

– результаты диагностики, включая вероятные заболевания;

– история заболеваний.

1.6.5 Среда эксплуатации

Приложение будет разработано для мобильных платформ Android, что обеспечит доступность для широкого круга пользователей. Оно будет функционировать на современных смартфонах и планшетах, способных обрабатывать изображения и поддерживать стандартные функции мобильного приложения. В дальнейшем будет рассматриваться доработка для операционных систем на IOS.

1.6.6 Требования к информационной и программной совместимости

Основные требования, содержащиеся к данной информационной и программной совместимости:

– приложение должно быть совместимо с последними версиями операционных системы Android;

– поддержка различных моделей камер, чтобы обеспечить корректное сканирование под лампой Вуда;

– возможность установки обновлений и патчей для улучшения функциональности и безопасности приложения.

Эти пункты могут стать основой для более детального описания в дипломном проекте, включая технические детали, исследования и методы, которые будут использоваться в процессе разработки приложения.

2 МОДЕЛИРОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

2.1 Разработка функциональной модели

Функциональная модель программного средства – это представление системы, которое фокусируется на её функциональных требованиях и поведении. Цель такой модели – определить, какие функции выполняет система и как они связаны друг с другом.

Ключевые компоненты функциональной модели:

– функции – основные действия, которые выполняет система. Например, в системе электронной коммерции такими функциями могут быть «поиск товара», «добавление товара в корзину» и «оформление заказа»;

– границы системы – описывают, где начинается и заканчивается область ответственности системы. Например, граница системы электронной коммерции может включать веб-сайт магазина, базу данных товаров и интерфейс для покупателей;

– входные и выходные данные – описание входных данных, необходимых для выполнения функций, и выходных данных, которые генерируются в результате выполнения этих функций;

– используемые ресурсы – указывает на внутренние и внешние ресурсы, необходимые для функционирования системы. Это могут быть данные, файлы, базы данных, сетевые соединения;

– последовательность выполнения – определяет порядок выполнения функций. Например, в системе электронной коммерции сначала нужно выбрать товар, затем добавить его в корзину и только потом оформить заказ.

Процесс создания функциональной модели:

– сбор требований – анализ документов и других источников для определения целей и задач системы;

– выделение функций – идентификация основных функций, которые система должна выполнять;

– детализация функций – разбиение каждой функции на более мелкие операции для лучшего понимания её реализации;

– определение взаимодействия – описание потоков данных и последовательности выполнения операций между различными функциями;

– визуализация модели – создание диаграмм и схем для наглядного представления структуры и взаимодействия функций;

– верификация и валидация – проверка модели на соответствие требованиям и ожидаемому поведению системы.

Преимущества использования функциональной модели:

– чёткое понимание задач – позволяет всем участникам проекта ясно видеть, какие функции должна выполнять система;

– легкость внесения изменений – если требования меняются, изменения легче внести на уровне функциональной модели, чем в уже реализованном коде;

– обнаружение ошибок – помогает обнаружить потенциальные проблемы на ранних стадиях разработки, прежде чем они станут дорогостоящими для исправления;

Функциональная модель является важным инструментом в процессе разработки программного обеспечения, помогающим создавать системы, которые соответствуют требованиям и ожиданиям пользователей.

Принципы построения функциональной модели мобильного приложения для определения заболеваний кожи с использованием лампы Вуда могут включать следующие ключевые аспекты:

– определение требований и функций – идентификация основных задач пользователя (например, распознавание поражений кожи);

– определение интерфейса приложения и его взаимодействия с устройством (лампой Вуда);

Учет необходимости интеграции с базой данных изображений и алгоритмов машинного обучения для анализа снимков.

Архитектура системы:

– разделение приложения на модули – пользовательский интерфейс, логика обработки данных, взаимодействие с внешними устройствами и базой данных;

– поддержка кроссплатформенной разработки для обеспечения совместимости с различными мобильными операционными системами.

Интеграция с лампами Вуда:

– реализация интерфейсов для управления освещением через приложение (включение/выключение лампы, настройка интенсивности света);

– обеспечение безопасности передачи данных между приложением и лампой.

Анализ изображений:

– использование методов компьютерного зрения и машинного обучения для автоматического распознавания поражений кожи;

– обработка и анализ полученных изображений для классификации заболеваний.

Сбор и хранение данных:

– хранение данных о пациентах и их историях болезни;

– организация базы данных для оптимизации доступа к данным и быстрого поиска информации.

Безопасность и конфиденциальность:

– шифрование данных пользователей;

– обеспечение защиты от несанкционированного доступа к личной медицинской информации.

Тестирование и отладка:

– проведение различных видов тестирования (функциональное, нагрузочное, интеграционное) для проверки корректности работы всех компонентов приложения;

– отладка ошибок и улучшение производительности.

Обновления и поддержка:

– возможность обновления приложения без потери данных;

– поддержание актуальности программного обеспечения и своевременное устранение багов.

Юзабилити и дизайн:

– создание удобного и интуитивно понятного интерфейса для пользователя;

– адаптация дизайна под различные размеры экранов и разрешения устройств.

Эти принципы помогут создать функциональную модель, которая будет эффективно выполнять свои задачи и соответствовать современным стандартам разработки мобильных приложений.

Контекстная диаграмма – это начальный этап разработки модели бизнес-процессов, который отображает общее представление о системе и её окружении. Она служит для визуализации основных взаимодействий между системой и внешней средой, а также для определения границ системы.

Основная цель контекстной диаграммы это – определить границы системы и выделить основные объекты (субъекты), которые с ней взаимодействуют [12].

Используемые элементы:

– система (обозначается прямоугольником);

– входы/выходы (стрелки от и к системе);

– внешние сущности (человеческие ресурсы, другие системы, клиенты.

Применение:

– помогает определить структуру и функциональность системы;

– служит основой для дальнейшего детального анализа процессов.

Преимущества:

– четкое определение области исследования;

– упрощение процесса моделирования и разработки системы;

– понимание взаимодействия между различными компонентами системы.

Контекстная диаграмма для мобильного приложения, которое использует лампу Вуда для диагностики кожных заболеваний, могла бы выглядеть следующим образом:

– система – обильное приложение для диагностики кожных заболеваний с использованием лампы Вуда.

Входы:

– пользовательские данные (фотографии пораженных участков кожи).

– данные об условиях окружающей среды (температура, влажность);

– настройки приложения (например, чувствительность камеры).

Выходы:

– результаты диагностики (описание возможного заболевания);

– рекомендации по лечению (если они доступны);

– информация о необходимости обращения к врачу.

Внешние сущности:

– пользователи (владельцы смартфонов, использующие приложение);

– Лампа Вуда (устройство);

– интернет (для загрузки обновлений алгоритмов диагностики);

– база данных (хранилище медицинских знаний и диагностических шаблонов).

Контекстная диаграмма помогает определить, каким образом приложение будет функционировать, какие данные оно обрабатывает и как эти данные влияют на конечный результат.

Рисунок 2.1 – Контекстная диаграмма

Диаграмма декомпозиции – это инструмент, который используется для анализа и представления сложных систем или процессов в более управляемом и понятном формате. Основная идея заключается в том, чтобы разбить большую систему или процесс на более мелкие составляющие, которые можно легче понять и управлять ими.

Основные характеристики диаграммы декомпозиции:

– иерархическая структура – диаграмма представляет собой многоуровневую структуру, где каждый уровень содержит более детализированную информацию по сравнению с предыдущим уровнем;

– графическое представление – обычно диаграмма представлена в виде блоков и стрелок, которые указывают на связи между элементами. Это делает её удобной для визуального восприятия и понимания;

– модульность – каждый блок (или модуль) на диаграмме может представлять отдельный элемент системы или процесса, который имеет свои собственные функции и задачи.

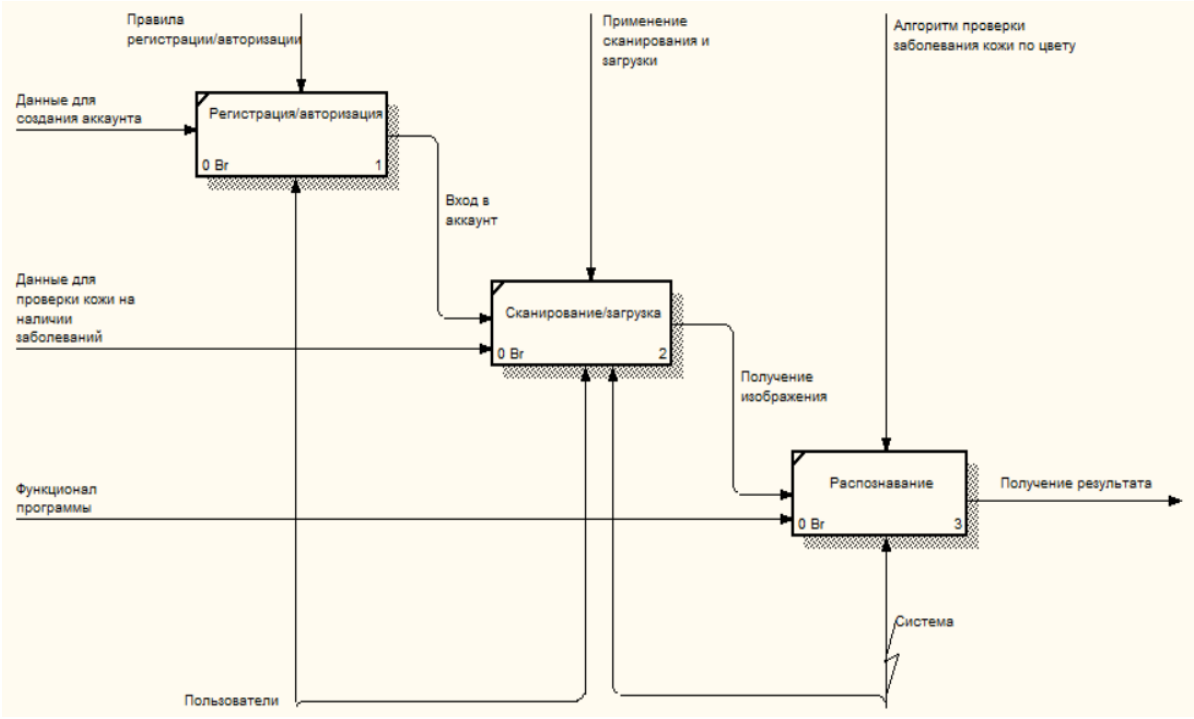


Рисунок 2.2. – Диаграмма декомпозиции

2.1 Проектирование базы данных

Проектирование базы данных – это важный этап разработки информационных систем, который включает в себя анализ требований, определение структуры данных и выбор методов хранения и обработки информации. Оно позволяет создать эффективную и масштабируемую базу данных, способную поддерживать нужные приложения и выполнять запросы пользователей быстро и точно.

Основные сущности для базы данных разрабатываемого программного средства представлены в таблицах 2.1–2.4, где расписано их имя, описание, тип и примечание. В данных таблицах можно увидеть структуру базы данных мобильного приложения для определения заболеваний кожи с помощью использования лампы Вуда.

Таблица 2.1 – Структура таблицы «User»

Имя поля	Описание	Тип	Примечание
ID_user	ID	int	Первичный ключ
full_name	full_name	varchar(255)	
password	password	varchar(255)	
email	email	varchar(255)	
number	number	varchar(255)	
date_birth	date_of_burthdat	date	

Таблица 2.2 – Структура таблицы «Symptoms»

Имя поля	Описание	Тип	Примечание
ID_symptomps	ID	int	Первичный ключ
ID_user	ID	int	
date_reported	date reported	Date	

Таблица 2.3 – Структура таблицы «Examination»

Имя поля	Описание	Тип	Примечание
ID_examination	ID	int	Первичный ключ
ID_user	ID	int	
ID_symptomps	ID	int	
type	type	varchar(255)	
date_performed	date_performed	date	
color	color	varchar(255)	
result	result	varchar(255)	

Таблица 2.4 – Структура таблицы «History»

Имя поля	Описание	Тип	Примечание
ID_history	ID	int	Первичный ключ
ID_examination	ID	int	
ID_user	user	int	

База данных вышла небольшая, так как не хотелось использовать слишком условных персональных данных пользователя, сходя из того, что приложение нужно для самостоятельного диагностирования заболеваний кожи, а не для выставления диагноза и, предположим, отправки результата лечащему врачу. Использованы необходимые таблицы, такие как:

– User – для регистрации и авторизации пользователя в системе;

– Symptoms – даты появления симптомов;

– Examination – характеристика симптомов и результат;

– History – история результатов диагностик для конкретного пользователя.

2.2 Спецификация функциональных требований

Спецификация функциональных требований (Functional Requirements Specification, FRS) – это документ, описывающий функциональные аспекты разрабатываемого программного продукта или системы. Она содержит детальное описание всех функций, которые должна выполнять система, включая ее взаимодействие с пользователем, другими системами и

окружением.

Основные элементы спецификации функциональных требований включают:

– цель и назначение системы – общее описание, зачем нужна система и какую задачу она решает;

– функциональные требования – подробное описание каждой функции системы, включая ее входные данные, выходные данные, исключения и ограничения;

– интерфейсы пользователя – описание пользовательского интерфейса, включая экраны, меню, кнопки и другие элементы;

– поведение системы – описание реакции системы на различные события и действия пользователя;

– долгосрочные цели – планируемые улучшения и расширения функционала в будущем;

– отчетность и мониторинг – описание процессов сбора и анализа данных для отчетности и мониторинга работы системы.

Важно отметить, что спецификация функциональных требований является важным этапом разработки программного обеспечения, так как она служит основой для дальнейшего проектирования, разработки и тестирования системы. Этот документ помогает разработчикам понять, что именно они должны создать, а также позволяет заказчикам убедиться, что система соответствует их ожиданиям и требованиям.

Мобильное приложение для определения заболеваний кожи с помощью лампы Вуда предназначено для облегчения диагностики кожных проблем и повышения точности их обнаружения. Такие приложения могут использоваться как профессионалами, так и обычными пользователями.

Главной положительной стороной программного средства является облегчение диагностики. Чтобы помочь пользователям быстрее и точнее определить наличие различных кожных заболеваний, используя возможности смартфона и специальной лампы Вуда. Также это снижение затрат на медицинские услуги, может позволить проводить предварительную диагностику дома, что может снизить количество визитов к врачу и сократить расходы на медицинское обслуживание.

Основное назначение – это предварительная диагностика, так как приложение помогает идентифицировать возможные кожные заболевания по характерным световым признакам, которые можно увидеть под ультрафиолетовым светом лампы Вуда. В приложении специально разработана функция просмотра и записи хранения данных. Сделано для возможности сохранения результатов диагностики и истории болезни для последующего анализа и контроля состояния здоровья.

Такое приложение, при правильном использовании, может стать полезным инструментом для поддержания здоровья кожи и своевременного обращения за медицинской помощью.

Функциональные требования для мобильного приложения для

определения заболеваний кожи с помощью лампы Вуда должны включать следующие ключевые аспекты, как диагностику благодаря которой есть возможность съемки кожи с использованием лампы Вуда, обработки и анализа изображения, вывод результатов диагностики.

В программном средстве будет происходить регистрация и авторизация пользователя, загрузка изображений, сканирование, просмотр списка заболеваний исходя из цвета свечения, определение заболевания по цвету свечения, просмотр истории заболеваний и просмотр причин различных заболеваний.

Также присутствует база данных различных заболеваний, которая имеет большое количество заболеваний и соответствующий им цвет. Есть дополнительные функции, как запись истории заболеваний и сортировка по дате.

По техническим требованиям необходима поддержка Android, адаптация под различные размеры экранов устройств, интерфейс должен быть интуитивно понятным для пользователя. Важным моментов является быстрая обработка изображений и вывод результатов и минимальная нагрузка на устройства. Эти функциональные требования помогут разработчикам создать надежное и эффективное мобильное приложение для определения заболеваний кожи с помощью лампы Вуда, которое будет отвечать современным стандартам качества и безопасности.

3 ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ПРОГРАММНОГО СРЕДСТВА

3.1 Разработка структуры программного средства

Разработка программного средства – это многоэтапный процесс, который включает несколько ключевых этапов. Вот основные шаги, которые обычно выполняются при разработке ПО:

– анализ требований;
– проектирование архитектуры;
– разработка;
– тестирование;
– отладка:
– документирование;
– внедрение и поддержка;
– сопровождение и улучшение.

Анализ требований является критически важным пунктом для успешной разработки программного средства, так как благодаря этому определяются конкретные цели и задачи, которые должна решать разрабатываемая система. Далее выявляются потребности пользователя из анализа требований, которые позволяет понять потребности конечных пользователей и клиентов, что способствует созданию более полезного и удобного в использовании продукта. Также присутствие функциональных требований описывающие, что делает система, а нефункциональные – как она должна работать, то есть ее производительность, безопасность, надежность. На основе собранных требований можно оценить объем работ, необходимые ресурсы (люди, оборудование, бюджет) и сроки выполнения проекта. Это помогает правильно распределить усилия и избежать задержек. Четко определенные требования помогают избежать ситуаций, когда разработчики и клиенты имеют разные представления о конечном продукте. Это уменьшает риск возникновения споров и дополнительных затрат на доработку системы. Понимание требований на ранних стадиях разработки позволяет создавать продукт высокого качества, соответствующий ожиданиям пользователей и бизнеса.

Эти этапы могут меняться в зависимости от специфики проекта, масштаба разработки и используемых методологий управления проектами (например, Agile, Waterfall и так далее).

Основные шаги разработки структуры мобильного приложения для определения заболеваний кожи с помощью использования лампы Вуда.

Анализ требований, на этом этапе определяются функциональные и нефункциональные требования к приложению.

Функциональные требования включают:

– возможность захвата изображения с помощью встроенной камеры смартфона;

– преобразование изображения в видимый спектр с помощью лампы

Вуда;

– распознавание и классификация различных типов поражений кожи с использованием алгоритмов машинного обучения.

Нефункциональные требования включают:

– высокий уровень защиты личной информации пользователей;

– минимальные требования к ресурсам устройства для оптимального функционирования приложения.

Проектирование архитектуры, на этом этапе определяется структура приложения и его основные компоненты.

Мобильное приложение:

– интерфейс пользователя для управления приложением;

– компоненты для захвата изображения и передачи данных на сервер.

Серверная часть:

– API для обработки изображений и возвращения результатов;

– базы данных для хранения изображений и результатов диагностики;

– алгоритмы машинного обучения для распознавания заболеваний кожи.

На этапе разработки осуществляется непосредственное написание кода для реализации всех функциональных возможностей приложения.

Мобильное приложение:

– создание пользовательского интерфейса;

– реализация функций захвата изображений и передачи данных на сервер.

Серверная часть:

– разработка API для обработки изображений;

– имплементация алгоритмов машинного обучения;

– интеграция с базами данных.

На этапе тестирования проводится проверка всех компонентов приложения на соответствие функциональным и нефункциональным требованиям.

Автоматизированное тестирование:

– юнит-тесты для проверки отдельных компонентов;

– интеграционное тестирование для проверки взаимодействия всех частей системы.

Ручное тестирование:

– тестирование пользовательского интерфейса;

– проверка работы с реальными данными и изображениями.

Тестирование безопасности:

– обеспечение защиты личных данных пользователей;

– предотвращение утечек информации.

Этап отладки выявляет и устраняются ошибки, обнаруженные в процессе тестирования.

Исправление дефектов:

– корректировка ошибок, связанных с работой приложения.

Оптимизация производительности:

– ускорение времени обработки изображений и ответа от сервера;

– балансировка нагрузки для обеспечения стабильной работы при высокой нагрузке.

Этап документирования нужен для составления всех необходимых документов.

Пользовательская документация:

– руководства по использованию приложения;

– описания функций и возможностей.

Техническая документация:

– описание архитектуры системы;

– схемы взаимодействия компонентов;

– инструкции по установке и настройке.

Безопасность и конфиденциальность:

– политики и процедуры по защите данных пользователей.

На этапе внедрения и поддержки осуществляется запуск приложения и поддержка пользователей.

Поддержка пользователей:

– обратная связь;

– ответы на вопросы;

– исправление багов после релиза.

Обновления и улучшения:

– добавление новых функций;

– исправление ошибок;

– адаптация к новым версиям операционных систем.

Этап сопровождения и улучшение необходим для мониторинга работы приложения и внедрения улучшений.

Мониторинг работы приложения:

– сбор статистики использования;

– анализ отзывов пользователей.

Обратная связь и улучшения:

– внесение изменений, на основе полученных данных и отзывов;

– развитие функционала.

3.2 Проектирование и разработка интерфейса ПС

Проектирование и разработка интерфейса программного средства являются важными этапами создания продукта, требующими тщательного планирования и учета множества факторов. Для того чтобы создать удобный и эффективный интерфейс, необходимо:

– понять потребности, ожидания и предпочтения целевой аудитории. Это позволит создавать интерфейсы, соответствующие потребностям пользователей и учитывающие их привычки;

– определить функциональные возможности программного средства и требования к интерфейсу. Необходимо описать, что будет доступно

пользователям, какие действия они смогут выполнять и какие элементы интерфейса будут использоваться;

– провести юзабилити исследования для оценки удобства и интуитивности интерфейса. Фокус-группы, тестирования пользователей и другие методы помогут определить, насколько эффективно пользователи взаимодействуют с продуктом;

– создать различные варианты дизайна интерфейса и протестировать их на разных группах пользователей. При этом важно учитывать принципы дизайна, такие как контраст, повторение, масштаб и близость (CRAP), чтобы сделать интерфейс привлекательным и удобным;

– уделить внимание эргономике. Правильное расположение элементов управления, размеры шрифтов и цветовая гамма должны быть тщательно продуманы, чтобы минимизировать утомляемость глаз и повысить комфортность использования4

– регулярно проводить тестирование usability для выявления проблем и улучшения интерфейса. Обратная связь от пользователей поможет внедрить необходимые изменения;

– обеспечить адаптивность и кроссплатформенность интерфейса. Он должен выглядеть и работать одинаково хорошо на различных устройствах и платформах;

– добиться согласованности и совместимости элементов интерфейса. Все компоненты системы должны быть взаимосвязаны и соответствовать общепринятым стандартам, чтобы пользователи могли легко ориентироваться без дополнительной подготовки;

– обеспечить безопасность и надежность интерфейса. Механизмы защиты данных и предотвращения ошибок пользователей должны быть встроены в систему;

– разработать подробную документацию и инструкции для пользователей. Это облегчит освоение функционала и поможет избежать ошибок.

Комплексный подход к проектированию и разработке интерфейса играет ключевую роль в создании качественного и удобного продукта. Учет всех этих моментов поможет создать эффективный и интуитивный интерфейс для вашего программного средства.

Стоит отметить особенную важность такого пункта, как эргономика и грамотное ее использование. Эргономика дизайна мобильного приложения включает в себя множество аспектов, направленных на создание удобного и интуитивно понятного интерфейса, который соответствует особенностям человеческого тела и психологии. В ходе изучения данного момента были выделены некоторые ключевые моменты, которые следует учитывать при разработке эргономичного дизайна мобильного приложения:

– использование размера экрана;

– интерактивные элементы;

– шрифт и цветовая схема;

– контекстные подсказки;
– физиологические особенности;
– отзывчивость и скорость реакции;
– логическая структура;
– анимации и эффекты;
– поддержка разных жестов;
– доступность.

Чтобы понять, что такое эргономика мобильного приложения надо пройтись по каждому из пунктов более подробно. Начнем с того, что мобильные устройства имеют разные размеры экранов, поэтому дизайн должен адаптироваться под разные разрешения. Важно использовать масштабируемые элементы и сетки, чтобы обеспечить оптимальное отображение контента на любом устройстве. Далее идет такой немаловажный момент, как расположение элементов: кнопок, ссылок и других интерактивных элементов, которое должно быть логичным и предсказуемым. Например, кнопка "Назад" обычно располагается в верхнем левом углу, а кнопки действий – в правом нижнем или по центру снизу. Касательно шрифтов тоже есть ряд моментов, которые стоит учитывать, а именно то, что шрифты должны быть читаемыми и достаточно крупными, особенно если приложение предназначено для людей старшего возраста или тех, кто имеет проблемы со зрением. Цветовая схема должна быть приятной для глаз и не вызывать усталости. Приложение должно предоставлять контекстные подсказки и обучающие материалы для новых пользователей, чтобы помочь им быстро освоиться с интерфейсом. Размер и форма кнопок, ссылок и других интерактивных элементов могут влиять на удобство взаимодействия с интерфейсом. Например, сенсорные области должны быть достаточно большими, чтобы их было легко нажимать даже большим пальцем.

Интерфейс должен реагировать на действия пользователя быстро и предсказуемо. Долгие задержки или неожиданные реакции могут привести к разочарованию пользователей. Навигация внутри приложения должна быть последовательной и логичной. Меню, вкладки и другие элементы управления должны быть организованы так, чтобы пользователь мог легко находить нужную информацию. Использование анимаций и плавных переходов может сделать взаимодействие с приложением более приятным и интуитивным. Однако слишком много анимаций может замедлить работу приложения и раздражать пользователей. Некоторые устройства поддерживают специальные жесты, такие как свайпы и щипки. Эти жесты можно интегрировать в интерфейс для увеличения удобства использования. приложение должно быть доступно для пользователей с ограниченными возможностями, включая слабовидящих и слабослышащих. Это может включать использование крупных шрифтов, специальных звуковых сигналов и других средств доступности.

Эргономичный дизайн мобильного приложения стремится максимально упростить взаимодействие пользователя с приложением, сделать его

приятным и эффективным [13].

Соблюдение цветовой гаммы при разработке макета мобильного приложения играет важную роль в создании приятного и интуитивно понятного интерфейса. Цвета используются для акцентирования внимания, навигации. Вот несколько ключевых аспектов, которые следует учитывать при выборе цветовой схемы:

– цель;

– координация цветов;

– акцентные цвета;

– читаемость.

Существуют различные методы выбора цветовых схем, например:

– монохромная схема – основана на использовании различных оттенков одного цвета. Помогает создать гармоничный и сбалансированный дизайн;

– аналогичная схема – основана на использовании цветов, находящихся рядом друг с другом на цветовом круге. Создает естественное и спокойное впечатление;

– комплементарная схема – основана на использовании цветов, находящихся напротив друг друга на цветовом круге. Создает контраст и выделяет элементы дизайна;

– триадная схема – основана на использовании цветов, находящихся на равном расстоянии друг от друга на цветовом круге. Помогает создать яркий и динамичный дизайн.

Кроме выбора цветовой схемы, стоит обратить внимание на цветовой контраст в дизайне. Цветовой контраст позволяет выделить ключевые элементы и обеспечивает хорошую читаемость информации. Для создания контраста можно использовать различные техники, например:

– контраст светлого и темного – основан на использовании цветов с разными яркостями или на противоположных концах спектра (например, черный и белый);

– контраст по цветовому тону – основан на использовании цветов, находящихся на противоположных концах цветового круга (например, красный и зеленый).

– контраст по насыщенности – основан на использовании цветов разной насыщенности (например, яркий красный и бледно-розовый).

Следует учесть восприятие цвета пользователем в зависимости от контекста использования приложения. Например, в условиях яркого освещения цветовые оттенки могут выглядеть иначе по сравнению с темным освещением. Поэтому важно проверить, как цвета будут восприниматься в различных условиях использования и выбрать такие оттенки, которые позволят наилучшим образом передать задуманный эффект [14].

Каждый цвет ассоциируется с определенными эмоциями и настроением. Например, розовый цвет вызывает эмоции спокойствия и умиротворения, его чаще всего используют в педиатрии и другими специалистами, работающими с детьми, также розовая форма отлично подходит всем медсёстрам – пациенты

подсознательно будут им доверять. Синий цвет вызывает чувство защищенности и открытости миру. Синий и голубой цвета подойдут сотрудникам хирургических отделений, а темно-синий цвет подойдет для психологов и арт-терапевтов. Зеленый цвет вызывает эмоции умиротворения и расслабления, поэтому используется во всех отраслях медицины. Фиолетовый цвет ассоциируется с мистикой, таинством, поэтому подойдет психологам, педиатрам, арт-терапевтам. Если разобраться с цветами более конкретнее, то получается, что цвета имеют сильное воздействие на эмоциональное состояние человека. Красный часто ассоциируется со страстью, силой, энергией, а также вызывает чувство внимания и агрессии. Синий часто ассоциируется с спокойствием, доверием, уверенностью, а также может вызывать ощущение прохлады и глубины. Зеленый ассоциируется с природой, ростом, свежестью, успехом, а также символизирует гармонию и релаксацию. Желтый часто ассоциируется с радостью, солнцем, оптимизмом, теплотой, но также может вызывать нервозность при слишком ярком применении. Черный ассоциируется с элегантностью, роскошью, силой, а также с трауром и серьезностью. В ходе нашего исследования мы обнаружили, что психология цвета играет ключевую роль в дизайне, оказывая значительное влияние на эмоциональное и поведенческое восприятие человека. Понимание того, как различные цвета влияют на наше сознание и подсознание, позволяет дизайнерам создавать эффективные и привлекательные решения, способные достигать поставленных целей [15].

Цветовая схема должна быть гармоничной и координированной. Обычно используют 2-3 основных цвета и дополнительные оттенки, чтобы создать единый стиль.

Используя акцентные цвета для выделения важных элементов интерфейса, таких как кнопки, ссылки или уведомления.

Цветовая схема должна обеспечивать достаточный контраст между текстом и фоном, чтобы текст оставался читаемым. Например, черный текст на белом фоне или белый текст на темном фоне обычно хорошо воспринимаются глазом.

Примером цветовой гаммы для разработанного дипломного проекта брались такие сайты, как invitro.by и helix.by.

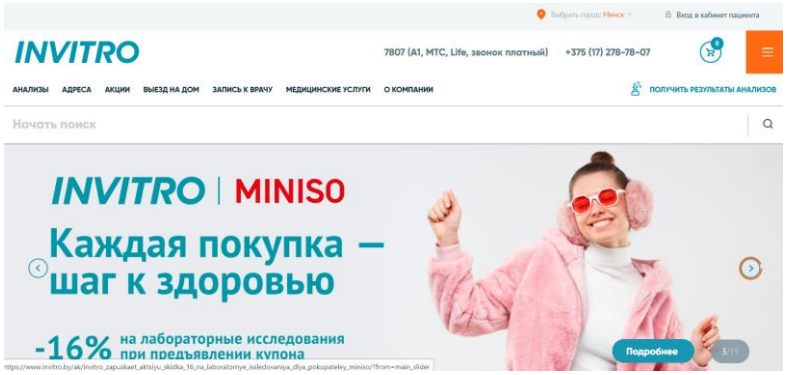


Рисунок 3.1 – Главная страница сайта invitro.by

Сайт invitro.by использует комбинацию темно-голубого, белого и серого цветов в своей цветовой гамме. Рассмотрим каждый из них отдельно и проанализируем их восприятие [15].

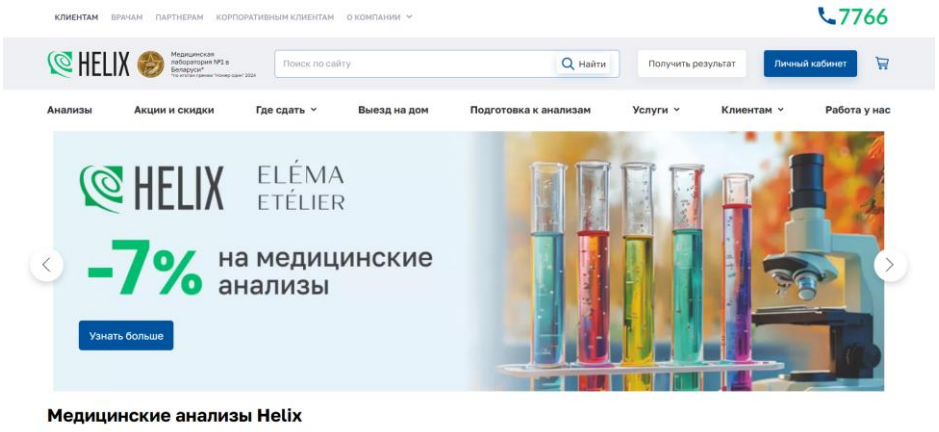
Голубой цвет на сайте invitro.by является основным. Этот цвет ассоциируется с водой, воздухом и спокойствием. В контексте медицинских услуг голубой цвет может передавать чувство чистоты, свежести и профессионализма. Он также помогает создать атмосферу спокойствия и уверенности, что может быть важным для пациентов, посещающих медицинский центр.

Белый цвет используется в качестве фона на сайте. Он делает страницы лёгкими и воздушными, создавая впечатление чистоты и стерильности, что опять же соответствует сфере медицинских услуг. Кроме того, белый фон позволяет другим элементам дизайна выделяться, делая их более заметными и привлекательными для пользователя.

Тёмно-серый цвет используется в меньшей степени, но всё же присутствует в некоторых элементах, например, в заголовках и тексте. Он помогает смягчить основной голубой цвет, придавая ему глубину и элегантность. Тёмно-серый цвет также добавляет контраста, делая важные элементы более заметными.

Общее впечатление от цветовой гаммы сайта invitro.by может быть следующим, что использование голубого цвета создает атмосферу профессионализма и доверия, что очень важно для медицинского центра. Белый фон усиливает ощущение чистоты и современности, что соответствует высоким стандартам медицинской помощи.

Комбинация голубого и белого цветов способствует созданию атмосферы спокойствия и уверенности, что может положительно повлиять на настроение посетителей сайта. Таким образом, цветовая гамма сайта invitro.by хорошо подходит для сферы медицинских услуг, создавая у конечных пользователей ощущение доверия, безопасности и профессионализма.



Медицинские анализы Helix

Рисунок 3.2 – Главная страница сайта helix.by

Сайт helix.by использует комбинацию зеленого, белого и серого цветов

в своей цветовой гамме. Рассмотрим каждый из них отдельно и проанализируем их восприятие [16].

Зелёный цвет на сайте helix.by является основным. Этот цвет ассоциируется с природой, здоровьем и свежестью. В контексте медицинских услуг зелёный цвет может передавать чувство безопасности и надёжности. Он также помогает создать атмосферу спокойствия и уверенности, что может быть важным для пациентов, посещающих медицинский центр.

Белый цвет используется в качестве фона на сайте. Он делает страницы лёгкими и воздушными, создавая впечатление чистоты и стерильности, что опять же соответствует сфере медицинских услуг. Кроме того, белый фон позволяет другим элементам дизайна выделяться, делая их более заметными и привлекательными для пользователя.

Серый цвет используется в меньшей степени на сайте, но всё же присутствует в некоторых элементах, например, в заголовках и тексте. Он помогает смягчить основной зелёный цвет, придавая ему глубину и элегантность. Серый цвет также добавляет контраста, делая важные элементы более заметными.

Общее впечатление от цветовой гаммы сайта helix.by может быть следующим, что использование зелёного цвета создает атмосферу безопасности и доверия, что очень важно для медицинского центра. Белый фон усиливает ощущение чистоты и профессионализма, что соответствует высоким стандартам медицинской помощи. Комбинация зелёного и белого цветов способствует созданию атмосферы спокойствия и уверенности, что может положительно повлиять на настроение посетителей сайта.

Таким образом, цветовая гамма сайта helix.by хорошо подходит для сферы медицинских услуг, создавая у конечных пользователей ощущение доверия, безопасности и профессионализма.

Макет мобильного приложения для дипломного проекта создавался с помощью использования графического онлайн-редактор. Программа позволяет создавать wireframe, UI, прототипы, презентации и с легкостью передавать материалы в разработку. В онлайн режиме можно наблюдать рабочий процесс, оставлять комментарии и обсуждать макет [17].

Часто используемые цвета – это белый и разные оттенки зеленого:

– #8F9E9C;

– #FFFFFF;

– #DFE7E3;

– #BBBDBD;

– #616C6A.

Эти цвета встречаются во всем приложении, как использующиеся для фона, например, как #8F9E9C и #FFFFFF, поле для ввода использует фон #DFE7E3, текст, который будет введен в данные поля #BBBDBD, а для выделения текста использовался #C6A6B7.

Чуть реже использовался черный:

– #000000.

Основные цвета можно увидеть на экране при входе в мобильное приложение. Практически полностью заполненный экран данным цветом призван вызвать спокойствие у пользователя, который решил проверить состояние кожи.

Рисунок 3.3 – Экран входа в приложение

После того, как пользователь нажал на кнопку “Next” происходит переход на экран, где осуществляется выбор “Log In” или “Sign Up”.

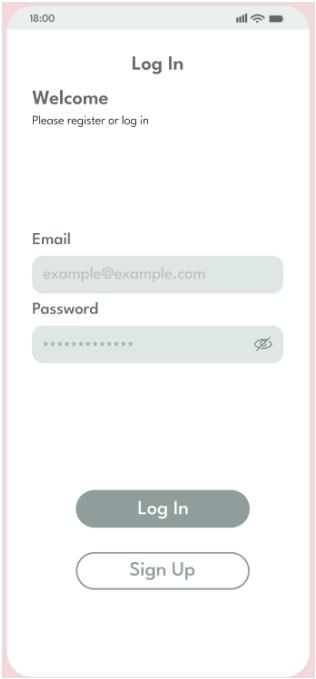


Рисунок 3.4 – Экран авторизации и регистрации пользователя

После введения пользователем необходимых данных для регистрации в текстовых полях с подсказывающим текстом происходит переход на экран, который показывает, что регистрация прошла успешно и можно продолжить, предварительно выйдя из данного экрана нажав на кнопку “Sign Out”.

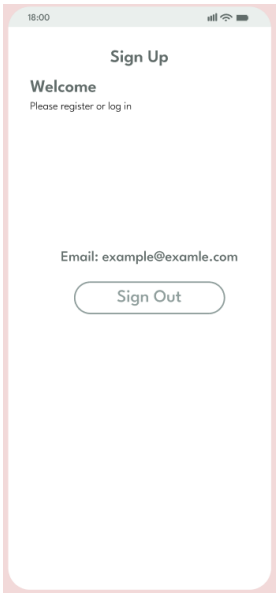


Рисунок 3.5 – Экран регистрации пользователя в приложении

На следующем экране можно загрузить изображение для получения результата, нажав на кнопку “Download”, прочитать инструкцию по использованию приложения, перейдя по кнопке “Manual” или просмотреть список заболеваний нажав на кнопку “Glow Values”.

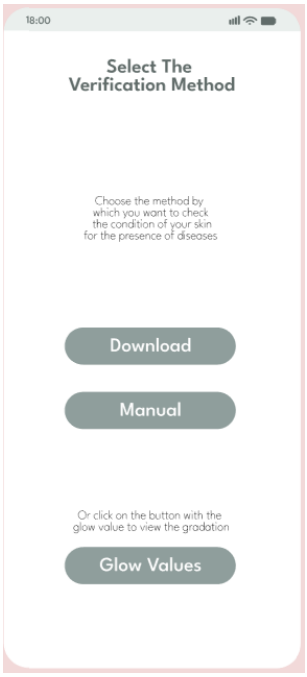


Рисунок 3.6 – Экран выбора дальнейших действий

Здесь представлен экран “Download” на котором нужно выбрать кнопку “Download Image” для загрузки изображения, а затем кнопку “Scan” для получения результата.

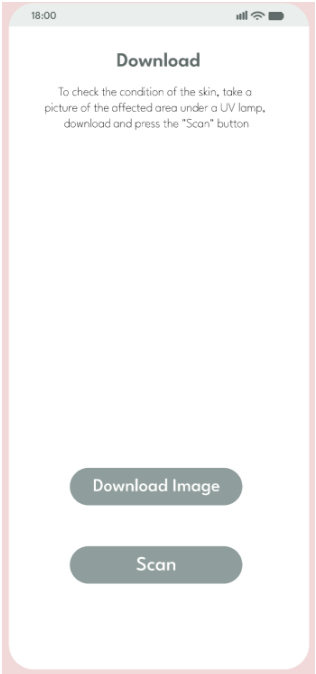


Рисунок 3.7 – Экран загрузки изображения

Далее появляется экран с результатом после проверки загруженного изображения пользователем.

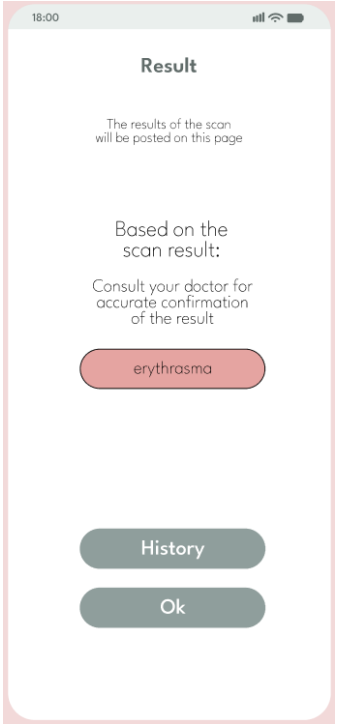


Рисунок 3.8 – Экран с результатом сканирования

На данном экране показан список заболеваний в виде кнопок, окрашенных в соответствующие цвета свечений, получаемых при использовании лампы Вуда.

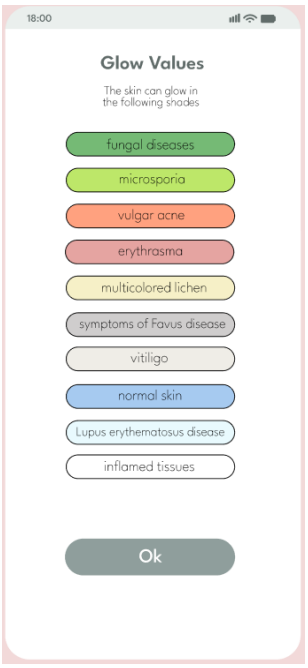


Рисунок 3.9 – Экран перечня всех заболеваний кожи

Далее показан экран с инструкцией для ознакомления пользователя с мобильным приложением.

Рисунок 3.10 – Экран с инструкцией по пользованию приложением

В мобильном приложение предусмотрено наличии истории для отслеживания того, какие заболевания были проверены у пользователя.

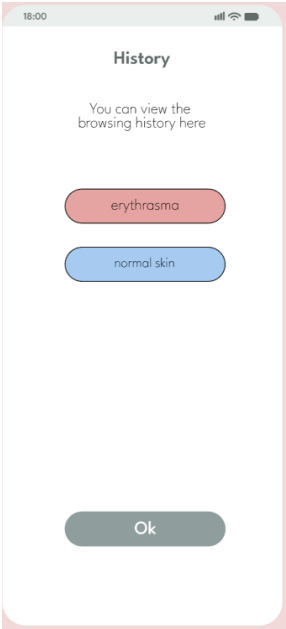


Рисунок 3.11 – Экран с историей заболеваний кожи

На следующем экране показан переход по одной из кнопок в экране списка заболеваний, в данном случае по кнопке “Fungal Diseases”. На экране представлены два изображения: слева то, как выглядит кожа при грибковом заболевании, а справа, как выглядит в свете лампы Вуда.

Рисунок 3.12 – Экран, показывающий, как выглядят грибковые заболевания и описывает причины возникновения

Экран открывается после того, как пользователь нажмет на кнопку в списке заболеваний “Microsporia”.

Рисунок 3.13 – Экран, показывающий, как выглядит микроспория и описывает причины возникновения

Экран открывается после того, как пользователь нажмет на кнопку в списке заболеваний “Vulgar Acne”.

Рисунок 3.14 – Экран, показывающий, как выглядят вульгарные угри (акне) и описывает причины возникновения

Экран открывается после того, как пользователь нажмет на кнопку в списке заболеваний “Erythrasma”.

Рисунок 3.15 – Экран, показывающий, как выглядит эритразма и описывает причины возникновения

Экран открывается после того, как пользователь нажмет на кнопку в списке заболеваний “Multicolored Lichen”.

Рисунок 3.16 – Экран, показывающий, как выглядит разноцветный лишай и описывает причины возникновения

Экран открывается после того, как пользователь нажмет на кнопку в списке заболеваний “Symptoms Of Favus Disease”.

Рисунок 3.17 – Экран, показывающий, как выглядит заболевание Фавуса и описывает причины возникновения

Экран открывается после того, как пользователь нажмет на кнопку в списке заболеваний “Vitiligo”.

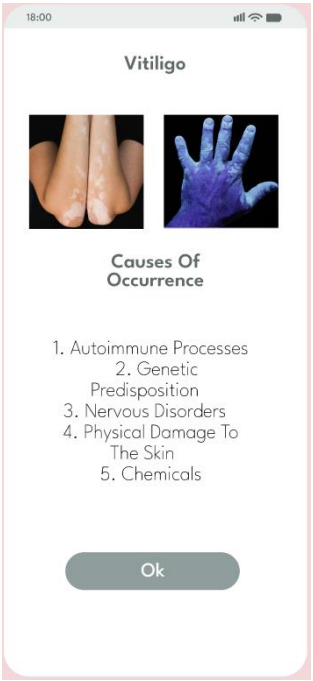


Рисунок 3.18 – Экран, показывающий, как выглядит Витилиго и описывает причины возникновения

Экран открывается после того, как пользователь нажмет на кнопку в списке заболеваний “Normal Skin”.

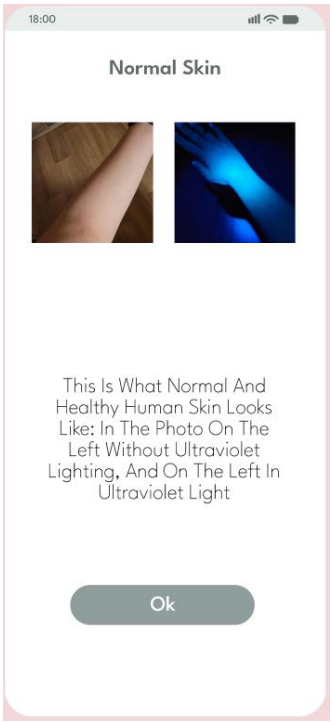


Рисунок 3.19 – Экран, описывающий, как выглядит нормальная кожа

Экран открывается после того, как пользователь нажмет на кнопку в списке заболеваний “Lupus Erythematosus Disease”.

Рисунок 3.20 – Экран, показывающий, как выглядит заболевание красная волчанка и описывает причины возникновения

Экран открывается после того, как пользователь нажмет на кнопку в списке заболеваний “Inflamed Tissues”.

Рисунок 3.21 – Экран, показывающий, как выглядит воспаленная кожа и описывает причины возникновения

3.3 Физическая модель базы данных

Физическая модель данных – это реализация логической модели данных, создаваемая администраторами и разработчиками баз данных. Она разрабатывается для определенных СУБД, технологий хранения и соединителей данных, чтобы по мере необходимости предоставлять данные через бизнес-системы пользователям. Это конечный результат всех остальных моделей – фактическая реализация массива данных [19].

Основные характеристики физической модели базы данных:

– физическое размещение данных;

– типы данных;

– кеширование и индексирование;

– архитектура хранилища;

– параметры производительности.

Каждый пункт из представленной характеристики немаловажен для физической модели базы данных, так как большое значение имеет то, где именно хранятся файлы и как они структурированы. Также какие типы данных используются и какие их максимальные размеры. Важным моментом является то, как кэшируются и индексируются для обеспечения высокой

производительности запросов. Затем идет пункт, который рассматривает то, какая архитектура хранилища и то, как данные распределены по различным уровням памяти и хранилищ. И то, как параметры влияют на производительность, такие как размер блоков данных, параметры как ввода, так и вывода и уровни изоляции транзакций.

Физическая модель тесно связана с логической моделью базы данных, которая описывает структуру данных с точки зрения бизнес-логики и предметной области. Логическая модель преобразуется в физическую модель при реализации базы данных.

Таблицы базы данных полностью соответствуют функционалу разрабатываемого мобильного приложения для определения заболевай кожи с помощью использования лампы Вуда. БД имеет четыре таблицы:

– User;
– Symptomps;
– Examination;
– Hystory.

В каждой таблице присутствуют поля и ключевые значения, отношения в такой БД – один ко многим.

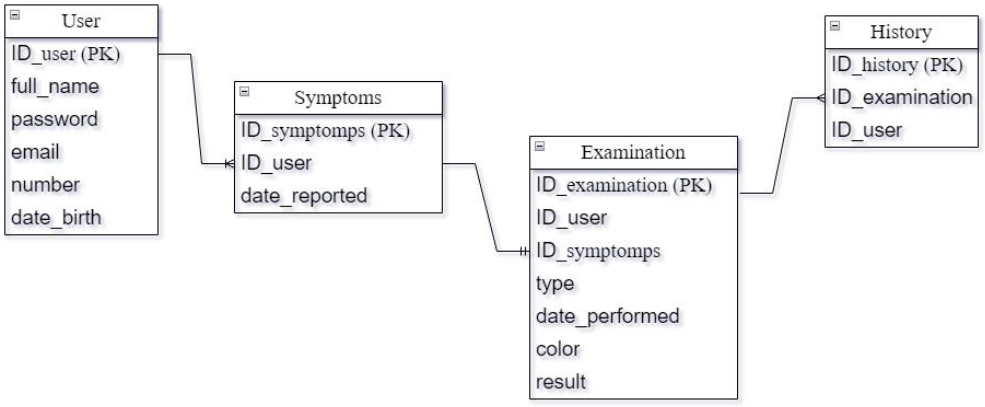


Рисунок 3.23 – Физическая модель базы данных

4 ТЕСТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА

4.1 Выбор и обоснование видов тестирования

При разработке мобильного приложения для определения заболеваний кожи с использованием React Native, JavaScript и Python OpenCV, Firebase были учтены специфики используемых технологий и инструментов, которые были рассмотрены в ключевых видах тестирования, которые являются важными для проекта и их обоснование.

Функциональное тестирование является основным типов тестирования, который должен быть выполнен для проверки того, соответствует ли функциональность приложения заявленным требованиям. Здесь важно проверить каждый компонент приложения отдельно и в совокупности. В данном типе тестирования были проверены следующие пункты:

– функции захвата изображения через камеру;

– обработку изображений с использованием OpenCV;

– алгоритмы анализа изображений и диагностику заболеваний;

– отображение результатов пользователю.

Обоснование выбора именно функционального тестирования заключается в том, чтобы убедиться, что все функции работают так, как задумано, без ошибок и сбоев. Приложение должно точно определять заболевания кожи на основе анализа изображений.

Интеграционное тестирование вид тестирования для проверки взаимодействия компонентов друг с другом. Были протестированы такие вещи, как:

– интеграция React Native с нативными модулями Python/OpenCV;

– передача данных между фронтендом и бэкендом;

– синхронизация процессов обработки изображений.

Обоснование именно такого выбора типа тестирования заключается в том, что приложение использует гибридную архитектуру, поэтому важно убедиться, что компоненты взаимодействуют корректно и не создают проблем в работе.

Тестирование производительности в данном мобильном приложении играет важную рол, например, для обеспечения плавной работы приложения. Учитывая, как раз тот момент, что приложение будет работать непосредственном с изображениями, то это требует значительных вычислительных ресурсов. Тестирование позволит убедиться в производительности, чтобы обработка изображения происходила достаточно быстро. Также стоит учесть тот аспект, что программное средство является мобильным приложением, которое имеет ограниченные ресурсы, такие как оперативная память и мощность процессора. Тесты помогут выяснить сколько ресурсов потребляет приложение и найти способы оптимизации кода, чтобы уменьшить нагрузку на устройство. Это особенно важно для долгосрочной стабильной работы приложения и предотвращения перегрева устройства. При

использовании такого типа тестирования можно предотвратить сбои и отказы в работе мобильного приложения.

Интерфейсное тестирование (UI/UX) важный тип, отвечающий за то, чтобы приложение одинаково выглядело и корректно работало на разных устройствах с различным расширением экранов. В рамках дипломного проекта этот пункт не так важен, так как react native является кроссплатформенным. Но тестирование необходимо, так как разные устройства имеют разное расширение экранов, которое должно поддерживать корректное отображение информации на устройстве.

Далее идет немаловажный тип юзабилити-тестирование, которое необходимо для проверки удобства использования приложения конечными пользователями, параллельно помогая улучшить пользовательских опыт и повысить их лояльность. В ходе использования этого типа были протестированы такие пункты:

– интуитивность интерфейса;
– простота навигации;
– ясность инструкций;
– возможность легкого доступа ко всем функциям.

Так как удобство использования приложения напрямую влияет на его успех. Пользователи должны легко понимать, как использовать приложение, не сталкиваясь с трудностями при использовании программного средства.

Также на будущее этого проекта был выбран тип, как кроссплатформенное тестирование поскольку React Native позволяет создавать приложения для нескольких платформ, необходимо убедиться, что оно одинаково хорошо работает на всех целевых устройствах.

– совместимость с различными версиями iOS и Android;
– поддержка разных разрешений экранов и форм-факторов устройств;
– корректная работа на разных версиях операционной системы.

Учитывая, что кроссплатформенность – одно из ключевых преимуществ React Native, то необходимы гарантии, что пользователи любого устройства смогут комфортно пользоваться приложением.

Перечисленные виды тестирования смогут обеспечить высокое качество работы мобильного приложения и минимизируют риски возникновения проблем после запуска.

4.2 Результаты тестирования

Тест-кейсы в разработке программного обеспечения – это детализированные сценарии, предназначенные для проверки определенных аспектов функционирования программы. Они содержат пошаговое руководство по выполнению тестов, предусловия, ожидаемые результаты и критерии успешности выполнения каждой операции. Цель тест-кейсов убедиться, что программа работает в соответствии с установленными требованиями и спецификациями.

Каждый тест-кейс обычно включает следующие элементы:
– название;
– краткое описание сути теста;
– идентификатор;
– предусловия;
– шаги выполнения;
– ожидаемый результат;
– фактический результат;
– статус;
– комментарии;

Использование тест-кейсов несет множество преимуществ, такие как прозрачность и воспроизводимость, так как каждый шаг четко описан, что позволяет любому члену команды повторить данный тест и получить такие же результаты. Далее идет документирование, где тест-кейсы служат документом, который подтверждает качество программного продукта, учитывая, что над этим ПО было проведено множество тестов, что довело его до такого этапа. Также можно отметить как плюс, как экономия времени, так как стандартизированные процедуры тестирования значительно ускоряют процесс выявления и устранения дефектов. И минимизация риска, снижение вероятности пропуска критических ошибок, которые могут пагубно повлиять на работоспособность системы.

Перед созданием таблицы будут описаны краткие характеристики в рамках дипломного проекта каждого из этапов тестирования. Функциональное тестирование включает в себя загрузку фотографий. Фотографии должны корректно загружаться и отображаться в интерфейсе приложения, что будет показателем того, что все работает корректно. Следующее идет интеграционное тестирование, которое интегрирует данные на сервер, отображая получение результата анализа изображения. Интерфейсное тестирование показывает оттестированные переходы между экранами корректно и без задержек, так же проверяется функционал и эргономика мобильного приложения для удобства пользователей. После идет юзабилити-тестирование необходимое для оценки удобства разрабатываемого мобильного приложения. Большинство пользователей, которым для тестирования было установлено данное приложение отметили, что его использование интуитивно понятное, смогли быстро и легко понять, как им пользоваться, загрузить изображения и получить результат. В разработанном мобильном приложении использовалось кроссплатформенное тестирование исходя из того, что данное программное средство было написано на React Native, благодаря которому можно использовать один написанный код для двух разных операционных систем, таких как Android и IOS. Приложение работало одинаково стабильно на всех тестируемых устройствах с различными характеристиками. Особых существенных отличий в работе приложений на разных моделях смартфонов не было замечено. Также было произведено нагрузочное тестирование для определения предельной

загруженности приложения

Как итог все проведённые тесты подтвердили корректную работу основных функций приложения, удобство его использования и стабильность работы на различных устройствах и ОС.

Таблица 3.1 – Основные этапы тестирования

Тип	Название	Шаг 1	Шаг 2	Результат
Функциональное тестирование	Загрузка фото	Открыть приложение	Нажать кнопку “Download Image”	Фото загружено
Интеграционное тестирование	Анализ изображения	После загрузки нажать на кнопку “Scan”	Дождаться завершения процесса анализа	Результаты сканирования на экране
Интерфейсное тестирование	Результаты сканирования	Запрос о получение результатов	Принять ответ от сервера	Приложение обработало результат
Юзабилити-тестирование	Работа кнопок и элементов управления	Проверить работоспособность кнопок	Проверить работоспособность	Опрос юзеров все ли понятно в приложении
Кроссплатформенное тестирование	Тестирование в разных ОС	Установка приложения на разные ОС	Функциональные тесты	Приложение работает корректно

Можно сделать заключение, касательно перечисленных элементов и преимуществ, что тест-кейсы играют важную роль в процессе разработки программного обеспечения, обеспечивая высокий уровень качества и надежности продукта. Помогая систематически проверять функциональность, безопасность и удобство использования программ, гарантируя их соответствие предъявляемым требованиям.

4.3 Вывод тестирования

Тестирование мобильного приложения для определения кожных заболеваний с использованием лампы Вуда является необходимым процессом для обеспечения его высокой эффективности, безопасности и удобства использования:

– точность диагностики, так как функциональное тестирование позволяет убедиться, что алгоритм анализа изображений работает корректно, выявляя признаки заболеваний с максимальной точностью;

– стабильность работы, учитывая, что тестирование производительности

помогает минимизировать задержки и ошибки при обработке изображений, что особенно важно для медицинских приложений, где скорость и надежность играют ключевую роль;

– кроссплатформенное тестирование гарантирует, что приложение будет корректно работать на различных устройствах и операционных системах, расширяя аудиторию пользователей;

– юзабилити-тестирование – удобство использования делает интерфейс интуитивно понятным и простым для пользователей, независимо от их технических знаний. Это снижает риск неправильного использования и повышает доверие к приложению.

– безопасность данных, уделяющие особое внимание защите личных данных пользователей, что достигается благодаря проведению специальных тестов на безопасность.

– комплексное тестирование помогает предотвратить потенциальные проблемы и гарантирует, что мобильное приложение будет надежным инструментом для диагностики заболеваний кожи.

5 РУКОВОДСТВО ПО ЭКСПЛУАТАЦИИ ПРОГРАММНОГО СРЕДСТВА

Руководство по эксплуатации программного средства служит важной документацией, которая предоставляет пользователям необходимую информацию для правильного и безопасного использования программного продукта. Оно выполняет несколько ключевых задач, например, как инструктаж по установке и настройке. Руководство содержит пошаговые инструкции по установке программного средства на целевую систему, включая требования к оборудованию и программному обеспечению. Это помогает пользователям избежать ошибок на этапе установки и настройки. Затем идет описание функциональных возможностей, объясняя, как ими пользоваться. Это облегчает освоение программы и помогает пользователям извлечь максимальную пользу из её применения. Решение возникающих проблем в руководстве часто приводятся рекомендации по устранению наиболее распространенных неполадок и ошибок, что позволяет пользователям самостоятельно решать возникающие вопросы без обращения в службу поддержки. Повышение безопасности: В документе указываются меры предосторожности и правила безопасной эксплуатации программного продукта, что помогает снизить риски возникновения аварийных ситуаций и повреждения данных. Соответствие стандартам и законодательству, некоторые стандарты и законодательные акты требуют наличия руководства по эксплуатации для определенных категорий программного обеспечения. Наличие такого документа способствует соблюдению нормативных требований. Документальное сопровождение является официальным документом, который может служить доказательством при возникновении спорных вопросов, связанных с эксплуатацией программного средства.

Таким образом, руководство по эксплуатации является незаменимым элементом документации, обеспечивающим правильное и безопасное использование программного продукта.

Руководство по эксплуатации мобильного приложения для определения заболеваний кожи с помощью использования лампы Вуда будет включать в себя такие пункты, как:

– описание приложения – приложение предназначено для определения заболеваний кожи с помощью использования специальной ультрафиолетовой лампы – лампы Вуда. Оно позволяет пользователям делать снимки пораженных участков кожи, обрабатывать эти изображения с помощью встроенных алгоритмов на базе OpenCV и Python, а затем предоставлять предварительные диагнозы. Приложение разработано на React Native для Android с использованием Firebase в качестве базы данных и Expo для упрощенной сборки и запуска.

– требования к системе – перед началом использования стоит убедитесь, что ваша система соответствует следующим минимальным требованиям:

– операционная система: Android версии 6.0 и выше;

– оперативная память: минимум 2 ГБ RAM;

– место на диске: не менее 500 МБ свободного места;

– камера: устройство должно иметь камеру с разрешением не ниже 8 МП.

– интернет-соединение: для первоначальной загрузки.

Далее идет установка приложения, которая включает в себя такие этапы, как:

– загрузка APK-файла – необходимо скачать последнюю версию APK-файла из доверенного источника;

– установка – нужно перейти в папку загрузок на устройстве и открыть APK файл, следуя инструкциям на экране для завершения процесса установки:

– найдите иконку приложения на главном экране или в меню приложений;

– нажмите на иконку для запуска приложения;

– при первом запуске предоставьте необходимые разрешения, такие как доступ к камере и хранилищу.

Основные функции данного мобильного приложения – это определение заболеваний кожи с помощью использования лампы Вуда. Для того, чтобы открыть приложение необходимо нажать на его иконку и подождать пару секунд пока загрузится. Если пользователь не зарегистрирован, то нужно пройти регистрацию, если ранее уже регистрировался, то просто пройти авторизацию. Далее пользователю необходимо выбрать сканирование (режим фотографии) или пункт просмотра градации цветов. После выбора пункта сканирования (режима фотографии) необходимо включить лампу Вуда и навести на пораженный участок кожи, сфотографировав его для дальнейшего определения. Сделав снимок приложение автоматически обработает изображение с использованием алгоритмов OpenCV. По получаемому цветовому определению можно понять, что у пользователя за кожное заболевание.

Приложение требует интернет-соединения только для обновлений.

Лицензионное соглашение необходимо для использования данного приложения регулируется лицензионным соглашением, с которым вы соглашаетесь при установке приложения. Пожалуйста, ознакомьтесь с ним перед началом использования.

6 ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ РАЗРАБОТКИ И РЕАЛИЗАЦИИ МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ ОПРЕДЕЛЕНИЯ ЗАБОЛЕВАНИЙ КОЖИ С ИСПОЛЬЗОВАНИЕМ ЛАМПЫ ВУДА НА МАССОВОМ РЫНКЕ

6.1 Характеристика программного средства, разрабатываемого для реализации на рынке

Мобильное приложение для определения заболеваний кожи с использованием лампы Вуда представляет собой инновационный инструмент, предназначенный для диагностики различных кожных патологий посредством анализа изображений, снятых под ультрафиолетовым излучением. Данное мобильное приложение сочетает в себе современные технологии искусственного интеллекта и машинного обучения, что позволяет точно определять изменения цвета и структуры кожи, характерные для различных заболеваний.

Целью разработки данного программного средства является создание удобного и эффективного инструмента для диагностики различных кожных заболеваний с использованием технологий искусственного интеллекта и анализа изображений, полученных с применением лампы Вуда.

Область применения полностью затрагивает такие направления, как медицина, дерматология, косметология, телемедицина.

Задачи, решаемые программным средством напрямую связаны с определением кожных заболеваний путем анализа изображений пораженных участков кожи под ультрафиолетовым освещением. Также это приложение будет актуально для врачей, как дополнительная помощь в постановке предварительного диагноза без необходимости личного визита к специалисту. В приложении автоматизирован процесс анализа и предоставление точных результатов на основе машинного обучения. Повысится доступность медицинской помощи для широкого круга пользователей.

В основные функции входит:

– загрузка и хранение изображений;

– анализ изображений с использованием нейросетевой модели для определения заболевания кожи;

– просмотр списка цветовых вариаций в зависимости от заболевания кожи.

К потенциальным покупателям программного средства относятся:

– медицинские учреждения (больницы, поликлиники, частные клиники);

– дерматологи и косметологи для использования в практике;

– пациенты, желающие самостоятельно контролировать состояние своей кожи;

– компании, занимающиеся производством средств по уходу за кожей, для сотрудничества и продвижения своих продуктов;

Актуальная потребность в разрабатываемом программном средстве:

– увеличение спроса на телемедицинские сервисы в связи с пандемией COVID-19;

– рост интереса к самостоятельному контролю за здоровьем и раннему выявлению заболеваний;

– недостаточная доступность квалифицированной медицинской помощи в удаленных регионах;

На рынке не так много конкурентов с похожим контентом:

– приложения для диагностики кожных заболеваний на базе ИИ (например, SkinVision, Miiskin) используются для определения злокачественных или доброкачественных родимых пятен;

– онлайн-сервисы для консультаций с дерматологами (например, DermatologistOnCall);

– традиционные методы диагностики, требующие физического присутствия врача.

Таким образом, мобильное приложение для определения заболеваний кожи с помощью лампы Вуда отвечает актуальным потребностям рынка и может занять свою нишу среди аналогичных решений, предлагая уникальные преимущества и высокий уровень сервиса.

Ключевыми особенностями приложения являются его интуитивно понятный интерфейс, позволяющий пользователям легко ориентироваться в функциях, и высокая скорость обработки изображений. Благодаря поддержке различных платформ, таких как Android и iOS, приложение доступно широкому кругу пользователей. Безопасность и конфиденциальность данных обеспечиваются современными методами шифрования и строгим соответствием международным стандартам защиты информации.

Экономическая модель приложения предусматривает различные источники доходов, включая продажу лицензий медицинским учреждениям и в будущем подписку на премиум-функции и сотрудничество с производителями средств по уходу за кожей. Таким образом, приложение не только решает важные медицинские задачи, но и представляет собой перспективный бизнес-проект с высокими шансами на успех на рынке.

На данный момент такое мобильное приложение является значимым шагом вперед в области телемедицины и ухода за здоровьем, предлагая удобный, точный и доступный инструмент для ранней диагностики кожных заболеваний.

6.2 Расчет инвестиций в разработку программного средства для его реализации на рынке

Расчеты инвестиций в разработку мобильного приложения для определения заболеваний кожи с использованием лампы Вуда необходимы для понимания экономической целесообразности и рентабельности проекта. Эти расчеты помогают оценить объем необходимых финансовых вложений, спрогнозировать сроки окупаемости и потенциальную прибыль, а также

минимизировать риски, связанные с реализацией идеи на рынке.

Инвестиции в разработку включают в себя оплату труда разработчиков, дизайнеров, тестировщиков и других специалистов, участвующих в создании приложения. Дополнительно учитываются затраты на приобретение необходимого оборудования, лицензии на программное обеспечение, аренду серверов и облачных сервисов, а также расходы на маркетинг и продвижение готового продукта [20].

Кроме того, расчеты инвестиций позволяют определить, сколько денег потребуется на поддержание и дальнейшее развитие приложения после его запуска. Сюда входят затраты на техническую поддержку пользователей, обновление функционала и адаптацию под новые платформы и устройства.

Также важно понимать, что инвестиции в разработку и реализацию мобильного приложения напрямую влияют на его качество и конкурентоспособность на рынке. Высококачественное приложение с широким спектром функций и стабильной работой привлечет больше пользователей и, соответственно, увеличит доходы от продаж лицензий, подписок и рекламных интеграций.

Таким образом, расчеты инвестиций дают четкое понимание финансовой стороны проекта, помогают принимать обоснованные решения и обеспечивают успешную реализацию мобильного приложения на рынке.

Расчет затрат на основную заработную плату команды разработчиков осуществляется исходя из состава и численности команды, размера месячной заработной платы каждого участника команды, а также трудоемкости работ, выполняемых при разработке программного средства отдельными исполнителями, по формуле:

$$З_о = К_{пр} \sum_{i=1}^{n} З_{чi} \cdot t_i, \qquad (6.1)$$

где $К_{пр}$ – коэффициент премий и иных стимулирующих выплат (по данным предприятия или в диапазоне 1,5–2); n – категории исполнителей, занятых разработкой программного средства; $З_{чi}$ – часовой оклад плата исполнителя i-й категории, р.; t_i – трудоемкость работ, выполняемых исполнителем i-й категории (определяется исходя из сложности разработки программного обеспечения и объема выполняемых им функций), ч.

Часовой оклад каждого исполнителя определяется путем деления его месячного оклада на количество рабочих часов в месяце, а именно 176 часов.

Размер месячного оклада исполнителя каждой категории соответствует установленному в организации-разработчике фактическому его размеру.

Расчет затрат на основную заработную плату осуществляется в табличной форме (табл. 6.1).

Таблица 6.1 Расчет затрат на основную заработную плату команды разработчиков

Исполнитель	Месячный оклад, р.	Часовой оклад, р.	Трудоъемкость работ, ч.	Итого, р.
Программист	3520	20	176	3 520
Тестировщик	2992	17	88	1 496
Дизайнер	2640	15	56	840
Итого				5 856
Премия и иные стимулирующие выплаты (50%)				2 928
Всего затрат на основную заработную плату разработчиков				8 784

Расчет затрат на основную заработную плату команды разработчиков. Принимаем в расчете коэффициент премий Кпр =1,5.

$$Зо = 1{,}5 \cdot \big((20 * 176) + (17 * 88) + (15 * 56)\big) = 8\,784 \text{ бел. руб.}$$

Дополнительная заработная плата разработчиков ($З_Д$) определяется по формуле:

$$З_Д = \frac{З_о \cdot Нд}{100}, \qquad (6.2)$$

где $Н_Д$ – норматив дополнительной заработной платы (10%)

$$З_Д = \frac{8\,784 * 10}{100} = 878{,}4 \text{ бел. руб.}$$

Отчисления на социальные нужды (Рсоц) определяется по формуле:

$$Р_{соц} = \frac{(З_о + З_Д) \cdot Нсоц}{100}, \qquad (6.3)$$

где $Н_{соц}$ – норматив отчислений в ФСЗН и Белгосстрах (в соответствии с действующим законодательством по состоянию на ноябрь 2024 г. – 34,6 %).

$$Р_{соц} = \frac{(8\,784 + 878{,}4) \cdot 34{,}6}{100} = 3\,343{,}19 \text{ бел. руб.}$$

Отчисления на социальные нужды (Рпр) определяется по формуле:

$$Р_{пр} = \frac{З_о \cdot Нпр}{100}, \qquad (6.4)$$

где $Н_{пр}$ – норматив прочих расходов (30%).

$$Р_{пр}=\frac{8\ 784\cdot 30}{100}\ =\ 2\ 635{,}2 \text{ бел. руб.}$$

Расходы на реализацию (Рр) определяется по формуле:

$$Р_р=\frac{З_о \cdot Нр}{100},$$

где Нр – норматив расходов на реализацию равен 3%.

$$Р_р=\frac{8\ 784 \cdot 3}{100}\ =\ 263{,}52\ \text{ бел руб.}$$

Общая сумма затрат на разработку и реализацию определяется по формуле:

$$Зр = Зо + Зд + Рсоц + Рпр + Рр \qquad (6.6)$$

Подставим значения в формулу:

Зр = 8 784 + 878,4 + 3 343,19 + 2 635,2 + 263,52 = 15 904,31 бел. руб.

Инвестициями для организации-разработчика программного средства являются затраты на его разработку, которые рассчитываются в соответствии с методикой, представленной в табл. 6.2.

Таблица 6.2 Методика расчета затрат на разработку программного средства, предназначенного для продажи

Наименование статьи затрат	Формула/таблица для расчета	Значение, р.
1. Основная заработная плата разработчиков ($З_о$)	Формула (6.1), таблица 6.1	8 784
2. Дополнительная заработная плата разработчиков ($З_Д$)	Формула (6.2)	878,4
3. Отчисления на социальные нужды ($Р_{соц}$)	Формула (6.3)	3 343,19
4. Прочие расходы ($Р_{пр}$)	Формула (6.4)	2 635,2
5. Расходы на реализацию($Р_р$)	Формула (6.5)	263,52
6. Общая сумма затрат на разработку и реализацию (Зр)	Формула (6.6)	15 04,31

6.3 Расчет экономического эффекта от реализации программного средства на рынке

Экономический эффект от реализации разработанного программного

средства выражается в увеличении чистой прибыли, которая зависит от объема продаж, цены реализации и затрат на его создание. Основные расчёты основываются на прогнозируемом количестве проданных лицензий, стоимости программного продукта и его рентабельности.

Обоснование прогнозируемого объёма продаж (45 000 копий):

1. Спрос на такого рода новое программное средство в мире медицины, дерматологии, косметологии не останется без должного внимания. Растущий спрос на постоянный мониторинг состояния, после событий с пандемией COVID-19 заставил многих обратить внимание на свое здоровье. Также в современной медицине стало появляться такое направление благодаря продвинутым технологиям стабильной видеосвязи, как телемедицина, которая основана на онлайн-консультациях с врачом. Основные положительные стороны телемедицины:

– доступность;
– эффективность;
– экономия времени и средств;
– повышенная гибкость;
– широкий спектр услуг;
– качество жизни.

2. Мобильное приложение для определения заболеваний кожи с использованием лампы Вуда может иметь несколько уникальных конкурентных преимуществ по сравнению с аналогичными продуктами, ориентированными на распознание злокачественных и доброкачественных родимых пятен. Есть несколько основных моментов, которые выделяют его среди конкурентов:

– ориентированность конкретно на проблемы заболеваний кожи;
– автоматизация процессов;
– точное распознавание болезни кожи;
– совмещение с профессиональными услугами.

3. Цена продукта будет 20 бел.рублей, что делает данное программное средство доступным для широкой аудитории, как и просто пользователей, следящих за здоровьем, так и для специалистов в областях медицины, дерматологии и косметологии, что может поспособствовать увеличению продаж. Если прогнозируемый объём продаж 45 000 копий, то такой прогноз считается реалистичным, учитывая широкую аудиторию, уникальность продукта, доступную цену и рабочую маркетинговую стратегию.

Расчёт ставки налога на добавленную стоимость производится по формуле:

$$НДС = \frac{Ц_{отп} \cdot N \cdot Н_{д.с}}{100\% + Н_{д.с}}, \qquad (6.7)$$

где $Н_{Д.С}$ – ставка налога на добавленную стоимость в соответствии с действующим законодательством, % (по состоянию на ноябрь 2024 г. – 20 %).

Подставляем значения:

$$НДС = \frac{20 \cdot 45\,000 \cdot 20\%}{100\% + 20\%} = \frac{180\,000}{1{,}2} = 150\,000 \text{ бел. руб.}$$

Прирост чистой прибыли (ΔПр) рассчитывается по формуле:

$$\Delta П_ч^р = \left(Ц_{отп} \cdot N - НДС\right) \cdot Р_{пр} \cdot \left(1 - \frac{Н_п}{100}\right) \qquad (6.8)$$

где $Ц_{отп}$ – отпускная цена копии (лицензии) программного средства, р.; N– количество копий (лицензий) программного средства, реализуемоеза год, шт.; НДС – сумма налога на добавленную стоимость, р.; $Р_{пр}$ – рентабельность продаж копий (лицензий) (30%); $Н_П$ – ставка налога на прибыль согласно действующему законодательству, % (по состоянию на ноябрь 2024 г. – 20%).

Подставим значения:

$$\Delta П_ч^р = (20 \cdot 45\,000 - 150\,000) \cdot 0{,}3 \cdot \left(1 - \frac{20}{100}\right)$$

$$\Delta П_ч^р = 750\,000 \cdot 0{,}3 \cdot 0{,}8 = 180\,000 \text{ бел. руб.}$$

Если организация-разработчик является резидентом Парка высоких технологий, прирост чистой прибыли рассчитывается по упрощённой формуле:

$$\Delta Пр = Цотп \cdot N \cdot Рпр \qquad (6.9)$$

Подставим значения:

$$\Delta Пр = 20 \cdot 45\,000 \cdot 0.3 = 270\,000 BYN.$$

Для резидентов ПВТ не нужно учитывать налоги на прибыль и НДС, что значительно увеличивает окончательную чистую прибыль.

6.4 Расчет показателей экономической эффективности разработки и реализации программного средства на рынке

Экономическая эффективность разработки и реализации программного средства определяется через сравнение инвестиций (затрат) на разработку с годовым приростом чистой прибыли (ΔПр). Если годовая прибыль превышает затраты, инвестиции считаются оправданными. Для оценки эффективности используется показатель рентабельности инвестиций (Return on Investment, ROI). Формула для расчёта ROI:

$$ROI = \frac{\Delta Пр - З_р}{З_р} \cdot 100\% \qquad (6.10)$$

Подставляем значения в формулу:

$$\text{ROI} = \frac{180\,000 - 15\,904{,}31}{15\,904{,}31} \cdot 100\%$$

$$\text{ROI} = \frac{164\,095{,}69}{15\,904{,}31} \cdot 100\% \approx 10{,}31\%$$

Рентабельность данных инвестиций составляет 10,31 %**,** что в свою очередь превышает стандартные ставки по долгосрочным депозитам 10 – 13,5 % годовых. Это указывает на то, что разработка программного средства является финансово выгодной. При таких показателях проект выгодно реализовать, потому что он гарантирует значительное увеличение прибыли в короткие сроки.

ЗАКЛЮЧЕНИЕ

В рамках данного дипломного проекта было разработано мобильно приложение для определения кожных заболеваний с использованием лампы Вуда. Разработанное приложение представляет собой эффективный инструмент для пациентов, позволяющий проводить первичную диагностику различных дерматологических состояний кожного покрова.

Для реализации программного средства использовались современные технологии, такие как React Native для создания мобильного интерфейса, Python с библиотекой OpenCV для обработки изображений и Firebase для хранения данных и обеспечения взаимодействия между компонентами системы. Это позволило создать удобное и функциональное решение являющиеся кроссплатформенным приложением, обеспечивающим высокую точность анализа изображений кожи при использовании лампы Вуда.

Разработанная система обладает рядом преимуществ, включая простоту использования, возможность быстрого получения результатов и доступность для широкого круга пользователей. Кроме того, она может быть легко интегрирована в существующие медицинские информационные системы, что делает её полезным дополнением к арсеналу современных методов диагностики.

Данное исследование подтверждает актуальность и перспективность применения информационных технологий в области медицины, а также демонстрирует возможности их эффективного использования для решения задач медицинской диагностики.

Таким образом, задачи дипломного проекта решены в полном объеме. Цель дипломного проекта достигнута.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Люминесцентная диагностика под лампой Вуда [Электронный ресурс]. Режим доступа: https://www.cmd-online.ru

[2] Антиплагиат.ру [Электронный ресурс]. Режим доступа: https://users.antiplagiat.ru/cabinet

[3] Упрощаем процесс разработки нативных React приложений с помощью Expo [Электронный ресурс]. Режим доступа: https://code.tutsplus.com/ru/easier-react-native-development-with-expo--cms-30546t

[4] Общие сведения о Firebase [Электронный ресурс]. – Режим доступа : https://support.google.com/admob/answer/6360054

[5] Язык программирования Javascript: особенности и преимущества [Электронный ресурс]. – Режим доступа : https://vc.ru/hr/145461-yazyk-programmirovaniya-javascript-osobennosti-i-preimushestva

[6] React Native in Action. – Nader Dabit

[7] Все про Node.js [Электронный ресурс]. Режим доступа: https://ru.hexlet.io/blog/posts/zachem-izuchat-node-js-ili-o-perspektivah-bekenda-na-javascript

[8] About – OpenCV [Электронный ресурс]. – Режим доступа : https://opencv.org/about/

[9] OpenCV в Python. Часть 1 [Электронный ресурс]. Режим доступа: https://habr.com/ru/articles/519454/

[10] Python [Электронный ресурс]. Режим доступа: https://blog.skillfactory.ru/glossary/python

[11] Осмотр и диагностика лампой Вуда в Москве | Добромед [Электронный ресурс]. – Режим доступа : https://dobromed.ru/methods/osmotr-s-lampoy-vuda.html?ysclid=m4ju6htgr5230740384

[12] Robert C. Martin. Clean Architecture: A Craftsman's Guide to Software Structure and Design. Prentice Hall, 2017.

[13] Steven Hoober, Eric Berkman. Designing Mobile Interfaces. O'Reilly Media, 2011.

[14] Психология цвета в дизайне мобильных и веб-приложений [Электронный ресурс]. Режим доступа: https://rating-gamedev.ru/blog/psixologiia-cveta-v-dizaine-mobilnyx-i-veb-prilozenii

[15] Психология цвета в дизайне [Электронный ресурс]. Режим доступа: https://studwork.ru/

[16] Медицинская компания Инвитро [Электронный ресурс]. Режим доступа: https://www.invitro.by/

[17] Медицинская лаборатория Хеликс [Электронный ресурс]. Режим доступа: https://helix.by

[18] Что такое Figma и для чего она нужна [Электронный ресурс]. Режим доступа: https://gb.ru/blog/chto-takoe-figma/

[19] Что такое моделирование данных [Электронный ресурс]. – Режим доступа : https://powerbi.microsoft.com/ru-ru/what-is-data-modeling/#

[20] Экономика проектных решений: методические указания по экономическому обоснованию дипломных проектов: учеб.-метод. пособие / В. Г. Горовой [и др.]. – Минск : БГУИР, 2021. – 30 с.

ПРИЛОЖЕНИЕ А
(обязательное)
Код программного средства

```
import cv2
import numpy as np
from sklearn.preprocessing import normalize

# Цвета в формате HSV с диапазонами
colors = {
 # Нормальная кожа
    "светло-синий": {"min": [195, 50, 50], "max": [225, 100, 100]},
 # Грибковые заболевания
    "зеленый": {"min": [55, 50, 50], "max": [75, 100, 100]},
 # Воспалительные ткани
    "белый": {"min": [0, 0, 90], "max": [360, 20, 100]},
 # Витилиго
    "молочно-белый": {"min": [0, 0, 80], "max": [30, 50, 100]},
 # Красная волчанка
    "снежно-белый": {"min": [0, 0, 90], "max": [30, 20, 100]},
 # Разноцветные лишай
    "тускло-желтый": {"min": [26, 50, 50], "max": [33, 100, 100]},
 # Микроспория
   "желто-зеленый": {"min": [77, 50, 50], "max": [99, 100, 100]},
 # Эритразма
    "кораллово-красный": {"min": [359, 25, 66], "max": [359, 67, 73]},
 # Фавус
    "бледно-серебристый": {"min": [0, 0, 60], "max": [360, 16, 83]},
# Вульгарные угри
    "оранжево-красный": {"min": [15, 40, 75], "max": [15, 100, 100]}
}

def is_in_range(hsv_color, color_range):
    return all(hsv_color[i] >= color_range["min"][i] and hsv_color[i] <=
color_range["max"][i] for i in range(3))

def classify_color(hsv_color):
    for name, color_range in colors.items():
        if is_in_range(hsv_color, color_range):
            return name
    return "Неизвестное"

def analyze_image(hsv):
    height, width, _ = hsv.shape
    result = np.zeros((height, width), dtype=object)

    for y in range(height):
        for x in range(width):
            pixel_hsv = hsv[y, x]
            classified_color = classify_color(pixel_hsv)
            result[y, x] = classified_color

    return result

def save_result(result, output_path):
    with open(output_path, 'w') as f:
        for row in result:
            f.write(" ".join(row) + "\n")
```

```
def main(input_path, output_path):
    # Загружаем изображение
    image = cv2.imread(input_path)

    # Преобразование в формат HSV
    hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)

    # Анализируем изображение
    result = analyze_image(hsv)

    # Сохраняем результат
    save_result(result, output_path)

if __name__ == "__main__":
    input_path = 'path_to_your_input_image.jpg'
    output_path = 'output.txt'
    main(input_path, output_path)
```