ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК

КАФЕДРА МАТЕМАТИЧЕСКОГО И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

РЕАЛИЗАЦИЯ МОБИЛЬНОГО ПРИЛОЖЕНИЯ «РОДНИКИ СВЯТОГО БЕЛОГОРЬЯ»

Выпускная квалификационная работа
обучающейся по направлению подготовки
02.03.03 Математическое обеспечение и администрирование информационных систем
очной формы обучения,
группы 07001402
Павликовой Виктории Вячеславовны

Научный руководитель:
ст.пр. Жуков А.В.

БЕЛГОРОД 2018

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ 3
ГЛАВА 1. ИЗУЧЕНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 6
1.1 Анализ предметной области 6
1.2 Обзор и анализ существующих приложений-аналогов 8
1.3 Определение требований к разрабатываемому приложению 16
ГЛАВА 2. ПРОЕКТИРОВАНИЕ МОБИЛЬНОГО ПРИЛОЖЕНИЯ 18
2.1 Выбор средств разработки 18
2.2 Создание прототипа мобильного приложения 22
2.3 Выбор метода поддержки принятия решения 27
2.4 Проектирование базы данных 32
ГЛАВА 3. РАЗРАБОТКА И ТЕСТИРОВАНИЕ МОБИЛЬНОГО ПРИЛОЖЕНИЯ 37
3.1 Разработка информационного обеспечения 37
3.2 Программная реализация мобильного приложения 39
3.3 Тестирование мобильного приложения 46
ЗАКЛЮЧЕНИЕ 51
СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 52

ВВЕДЕНИЕ

Благодаря возрастающим техническим возможностям коммуникации роль информации и знаний приобретает всё большее значение для общества.

Ещё в недавнем прошлом поиск нужной информации являлся довольно сложным процессом. Для того, чтобы найти интересующий материал, необходимо было воспользоваться литературными источниками, что занимало немало времени и усилий, а также не гарантировало успешного результата.

С появлением сети Интернет отыскание необходимого материала стало значительно проще, ведь главной идеей Всемирной паутины является свободное распространение информации и знаний. Благодаря этому поиск каких-либо сведений становится удобным и легкодоступным.

Интернет занимает значимое место в современном обществе и в настоящее время является неотъемлемой частью персонального компьютера. Он предоставляет пользователям большой спектр возможностей, таких как: быстрый поиск различной информации, виртуальное общение, заработок денег, проведение досуга за просмотром фильмов, прослушиванием музыки или играми и т.д. Исходя из этого, можно утверждать, что Интернет охватывает все сферы человеческой деятельности.

Неудобство эксплуатации компьютеров и ноутбуков обуславливает появление различных смартфонов и планшетов, что позволяет осуществить доступ в Интернет независимо от времени и местонахождения.

На сегодняшний день смартфон есть практически у каждого человека. В данном устройстве для поиска необходимой информации можно использовать приложения, которые являются более удобными и эффективными благодаря своей конкретной направленности.

Исходя из вышесказанного, задача создания мобильного приложения с целью донесения конечному пользователю той информации, в которой он нуждается, остаётся актуальной по сегодняшний день.

Актуальность темы выпускной квалификационной работы заключается в необходимости реализации мобильного приложения для быстрого и удобного поиска родников Белгородской области на карте.

Цель выпускной квалификационной работы заключается в реализации мобильного приложения «Родники Святого Белогорья» для платформы Android.

Для достижения поставленной цели были сформулированы следующие задачи:

1. Анализ предметной области.
2. Определение ключевой направленности разрабатываемого мобильного приложения.
3. Определение требований к разрабатываемому приложению.
4. Проведение обзора и подробный анализ существующих приложений с идентичной направленностью.
5. Проектирование мобильного приложения.
6. Разработка и тестирование мобильного приложения «Родники Святого Белогорья».

Выпускная квалификационная работа имеет традиционную структуру и включает в себя введение, теоретическую часть, практическую часть, заключение, список использованных источников, приложение.

Во введении обоснована актуальность темы «Реализация мобильного приложения «Родники Святого Белогорья»», определена основная цель работы и поставлены задачи для её достижения.

Первая глава состоит из теоретической части. Здесь описывается анализ предметной области, приводится обзорный анализ существующих приложений-аналогов и определение требований к разрабатываемому приложению.

Вторая глава под названием «Проектирование мобильного приложения» включает в себя выбор инструментальных средств разработки,

создание прототипа мобильного приложения, анализ и выбор метода поддержки принятия решение и проектирование базы данных.

В третьей главе описана разработка информационного обеспечения и программная реализация мобильного приложения. Также приведено тестирование готового продукта.

В заключении приведены основные выводы и результаты, полученные в ходе выполнения работы.

С целью подробного изложения данной темы были использованы интернет-сайты, научная и учебная литература.

Данная выпускная квалификационная работа содержит 53 страницы, 21 рисунок, 2 таблицы, 4 формулы, 13 листингов и приложение.

ГЛАВА 1. ИЗУЧЕНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Анализ предметной области

Родниками называют воды, которые выходят на земную поверхность естественным путём, без вмешательства человеческих сил. По своей сути они считаются уникальными водоёмами, созданными самой природой.

Родники являются стратегическими объектами природы. При возникновении чрезвычайной ситуации они могут выступать как единственные источники питьевой воды для населения.

Добираясь до поверхности земли, ключевая вода, которую мы берём из родника, подвергается естественной фильтрации. Многочисленные слои песка и породы не пропускают грязь. Главное, что подобный род очистки не убирает из воды полезных свойств, не влияет на её структуру, не затрагивает гидрохимический состав. Таким образом, ключевая вода проходит природную очистку. По этим причинам воду из родника можно и нужно пить, не очищая её дополнительными методами [8].

Родников с водой, обладающей целебными свойствами, на самом деле очень мало. В большом проценте источников жидкость не только не имеет полезных веществ в составе, но и вполне сама может навредить человеку, вызвав у него ряд серьёзных заболеваний. Причины тому следующие: водный слой, из которого рождаются родники, находится в непосредственной близости к поверхности земли, а, значит, в него могут проникнуть вредные вещества снаружи. Особенно опасно пить воду из источника, находящегося близ свалок или заводов.

Вода в родниках всегда была чище воды рек, озер и любых других поверхностных источников. Поэтому люди издавна стремились пользоваться этими источниками, сохраняли и обустраивали их.

Конечно, в современных системах водоснабжения родники играют более чем скромную роль. Города и поселки, промышленные и

сельскохозяйственные предприятия получают воду из централизованных систем водоснабжения, питающихся подземными (обычно им отдается предпочтение) или поверхностными источниками. На всем пути от водоприемников до водоразборных кранов вода надежно защищается от загрязнения. За ее качеством ведётся постоянный строгий контроль.

И все-таки родинками продолжают пользоваться до сих пор. Их воду берут для питья, идут с бидонами и ведрами, порой за несколько километров, чтобы отведать самого лучшего напитка, подаренного природой – чистой воды.

Некоторые люди считают, что родниковая вода является святой и целебной. Многие специально едут к источникам на дальнее расстояние, чтобы набрать воды для лечения каких-либо заболеваний. Поэтому рядом со многими родниками построены часовни, которые в свою очередь представляют культурную, а некоторые и историческую ценность [13].

В 2014 году в Белгородчине стартовал областной проект «Сохраним родники Белогорья», целью которого является изучение экологического состояния родников, их сохранение и благоустройство, разработка экскурсионных маршрутов.

Проведённые исследования показали, что в Белгородской области насчитывается 1110 родников. Также приведено описание местонахождения источников, результаты лабораторных исследований состава воды и определены географические координаты родников. В процессе работы были составлены паспорта родников.

Для того, чтобы люди могли воспользоваться информацией о родниках, была выявлена необходимость в создании мобильного приложения «Родники Святого Белогорья».

1.2 Обзор и анализ существующих приложений-аналогов

На сегодняшний день мобильные технологии развиваются ускоренными темпами. Количество мобильных приложений стремительно увеличивается, поэтому создать уникальное приложение, которое сможет завоевать большую аудиторию, очень сложно. Еще сложнее убедить пользователей мобильных устройств в том, что именно Ваше приложение – лучшее среди всех ему подобных.

Для того, чтобы решить, как должно работать мобильное приложение и что можно сделать для его улучшения и привлечения новых пользователей, необходимо произвести обзор и анализ приложений-аналогов.

Анализ мобильных приложений – это, по сути, их всестороннее тестирование для выявления слабых сторон, а также учёт всей доступной информации для увеличения эффективности разрабатываемого продукта.

Стоит отметить, что приложения для поиска родников на карте в Белгородской области не существует. Поэтому для сравнения было выбрано три наиболее популярных по количеству скачиваний и оценкам пользователей приложения, относящихся к данной категории.

Для проведения анализа были выбраны следующие приложения: «Find Water», «Карта Хортицы +», «Water Life».

В ходе рассмотрения приложений был проведён анализ по следующим критериям: интерфейс, содержательность, функционал, актуальность информации.

Первым будет рассмотрено приложение «Find Water», которое предназначено для поиска воды в столице Чешской Республики – в Праге.

Запущенное приложение изображено на рис. 1.1.

Рис. 1.1. Работа приложения «Find Water»

Интерфейс данного приложения является простым и лёгким для восприятия. Преобладают спокойные тона, которые не приедаются пользователю и не создают желания как можно скорее покинуть используемый продукт. Все элементы расположены удобно и являются интуитивно понятными, благодаря чему у пользователя не возникает затруднений при использовании приложения. Информативный блок выполнен аккуратно и является читабельным. Исходя из этого, можно утверждать, что интерфейс оформлен грамотно.

Приложение «Find Water» состоит из информации об источниках питьевой воды, которые помечены на карте в виде капель. Каждый источник содержит краткие сведения о себе – координаты местоположения и адрес, как показано на рис. 1.2.

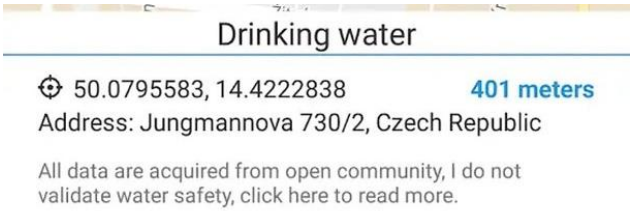

Рис. 1.2. Информативный блок приложения

Отсутствуют фотографии, а также информация о дате последнего анализа состояния воды. Это говорит том, что содержательность рассматриваемого приложения является недостаточной.

Основной задачей приложения является поиск питьевых источников. При входе в приложение автоматически определяется местоположение пользователя. Как было отмечено выше, водные ресурсы помечены на карте в виде капель, при нажатии на которые внизу экрана появляется информация о них, включающая в себя рассчитанное расстояние от местоположения. Также есть возможность осуществления поиска по адресу с помощью поисковой строки. В верхнем правом углу расположена кнопка перехода к сведениям о приложении. Возможность построения маршрута отсутствует, поэтому функционал приложения можно считать не полностью проработанным, так как данная функция является неотъемлемой для подобного вида приложений.

Приложение «Find Water» было добавлено в Play Market 6 ноября 2016 года, после чего не обновлялось. К сегодняшнему дню сведения о состоянии воды могли измениться, поэтому информация не является актуальной.

Исходя из проведенного анализа, можно сделать вывод о том, что приложение «Find Water» является удобным для восприятия, но не очень информативным, не достаточно функциональным и необновляемым с момента релиза.

Далее будет рассмотрено приложение «Карта Хортицы +». Данное приложение предназначено для поиска интересующих объектов на острове Хортица, находящемся в Украине в черте города Запорожье.

Запущенное приложение изображено на рис. 1.3.

Рис. 1.3. Работа приложения «Карта Хортицы +»

Интерфейс рассматриваемого приложения является очень удобным и понятным для восприятия. Оформление выполнено очень качественно – размер и тип штифта, а также расположение элементов использованы согласовано. Хорошо подобрана цветовая палитра, яркие цвета не утомляют. Основные кнопки являются интуитивно понятными, поэтому пользователь может без затруднений получить доступ к необходимой для него информации. Всё это говорит о том, что интерфейс является грамотно оформленным.

Содержательность приложения «Карта Хортицы +» состоит из информации об объектах, относящихся к различным категориям. Каждый

объект содержит фото, а также основную информацию о себе, как показано на рис. 1.4.

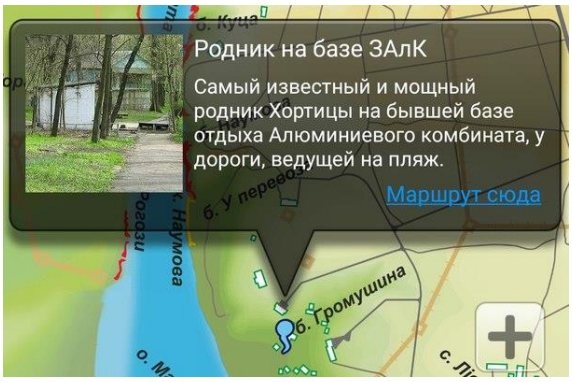

Рис. 1.4. Информативный блок приложения

Также можно перейти к информации о расписании транспорта, получить справку о предназначении кнопок главного меню, о возможных действиях с картой и ознакомиться с правилами посещения заповедника. Исходя из этого, можно утверждать, что рассматриваемое приложение является очень информативным.

Основной задачей приложения является предоставление пользователю туристической карты Хортицы с метками различного рода объектов. При первом входе в приложение требуется наличие доступа к Интернету для загрузки карты острова. При нажатии на объект на экране появляется краткая информация о нём (см. рис. 1.4) с возможностью перехода к подробному описанию и построения маршрута. В приложении имеется возможность включения GPS-навигации, перехода к текущему местоположению и активация слежения за ним. Также пользователь может записывать путь, по которому будет двигаться в процессе навигации с помощью соответствующей кнопки. Есть возможность измерить длину любой ломаной линии или расстояние между двумя точками, активировав функцию линейки. Присутствует активация панели поиска по карте. В настройках приложения можно выбрать объекты, которые следует отобразить на карте, обновить объекты с помощью сервера, а также изменить язык. Всё перечисленное

свидетельствует о том, что данное приложение является довольно функциональным.

Приложение «Карта Хортицы +» добавлено в Play Market 8 апреля 2015 года. Последнее обновление было 7 февраля 2017 года. Это свидетельствует о том, что информация является актуальной.

Исходя из проведенного анализа, можно сделать вывод о том, что приложение «Карта Хортицы +» является грамотно оформленным, информативным, функциональным и периодически обновляемым.

Последним будет рассмотрено приложение «Water Life». Данное приложение предназначено для поиска водных объектов Восточной Австралии.

Запущенное приложение изображено на рис. 1.5.

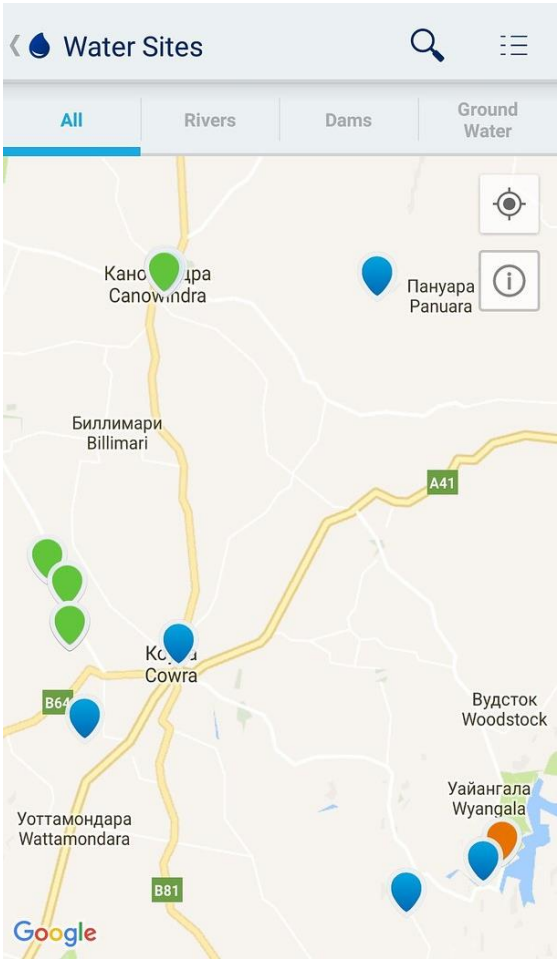

Рис. 1.5. Работа приложения «Water Life»

Интерфейс данного приложения является простым и удобным для восприятия. Необходимую для себя информацию пользователь может получить с помощью меню. В приложении преобладают голубые тона, соответствующие тематике. Хорошо подобран размер и тип шрифта, благодаря чему текст является читабельным. Информативный блок аккуратно оформлен в виде таблицы. Присутствует анимация в виде капли воды, появляющаяся во время загрузки данных. Всё это свидетельствует о том, что рассматриваемое приложение имеет грамотно оформленный интерфейс.

Приложение «Water Life» предоставляет пользователю доступ к информации в режиме реального времени от автоматических цифровых устройств мониторинга воды. Информация, получаемая в процессе мониторинга, может включать такие данные, как данные о потоках, уровнях грунтовых вод, объемах хранения, температуре воды, осадках и различных типах данных о качестве воды. Объектами исследования являются реки, платины и подземные воды. Пример информативного блока можно увидеть на рис. 1.6.

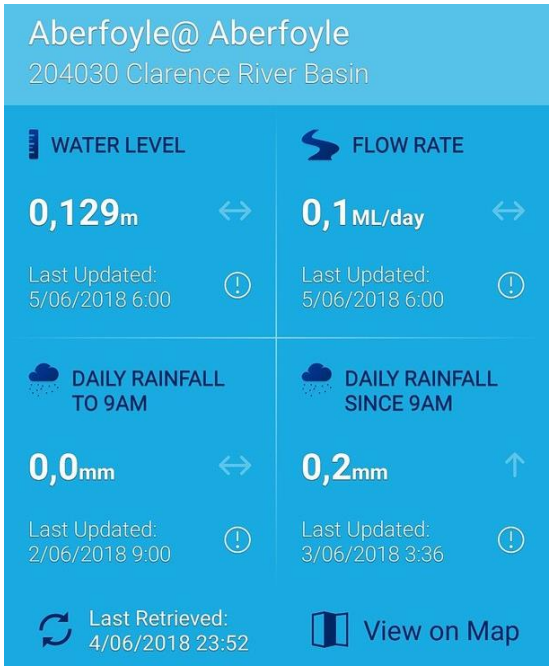

Рис. 1.6. Информативный блок приложения

Также можно ознакомиться с информацией о приложении и его функциях. Исходя из того, что далеко не все водные объекты имеют информацию о себе, можно утверждать, что рассматриваемое приложение является не достаточно информативным.

Основной задачей приложения является поиск водных ресурсов и определение показателей состояния воды. Водные ресурсы помечены на карте в виде меток, при нажатии на которые появляется название источника. Отсюда можно перейти к информации о мониторинге состояния воды. В информативном блоке есть возможность перехода к детальным сведениям мониторинга, но при переходе информация не отображается. Также в приложении можно добавить объект в избранное для дальнейшего быстрого доступа к нему. Присутствует возможность поиска по названию с помощью поисковой строки. Есть возможность группировки объектов на карте. Исходя из всего перечисленного, можно сделать вывод о том, что функционал приложения является хорошим, но не доработанным.

Приложение «Water Life» добавлено в Play Market 22 февраля 2015 года. Последнее обновление было 18 октября 2017 года, но данные мониторинга обновляются ежедневно. Это говорит о том, что информация является актуальной.

Исходя из проведенного анализа, можно сделать вывод о том, что приложение «Water Life» является удобным для восприятия, не достаточно информативным, имеет немного недоработанный функционал и постоянно обновляется.

Краткие выводы обзора и анализа существующих приложений-аналогов приведены в табл. 1.1.

Таблица 1.1

Выводы по анализу приложений-аналогов

	Интерфейс	Содержательность	Функционал	Актуальность информации
«Find Water»	+	±	±	–
«Карта Хортицы +»	+	+	+	+
«Water Life»	+	±	±	+

1.3 Определение требований к разрабатываемому приложению

На сегодняшний день приложения для поиска родников на карте в Белгородской области, а также получения информации об источниках не существует. Именно по этим причинам необходимо разработать информативно-полезное и часто обновляемое приложение.

На основании анализа существующих приложений-аналогов составлены требования к разрабатываемому приложению «Родники Святого Белогорья».

Дизайн мобильного приложения должен быть прост и иметь понятный интерфейс. Минималистический дизайн, вызывающий эстетическое наслаждение у пользователя – самое верное решение. Именно поэтому следует подбирать нейтральные тона, не раздражающие пользователя. Элементы должны быть грамотно расположены и являться интуитивно понятными, ведь когда пользователь не в состоянии найти нужную информацию, он склонен предполагать, что в рассматриваемом ресурсе вообще её нет. Текст должен быть небольшим и легко читаемым, отражающим главную суть. Желательно использовать подачу сведений списками – это облегчает восприятие подаваемой информации.

Приложение «Родники Святого Белогорья» должно предоставлять пользователю доступ к следующей информации: название и

месторасположение источника на карте, адрес, фото, результаты лабораторных исследований на чистоту воды, статус освященности и благоустройства, а также дата последнего проводимого исследования и список попечителей.

Основной задачей приложения является поиск источников ключевой воды. При входе в приложение местоположение пользователя должно определяться автоматически. Водные ресурсы должны быть помечены на карте в виде меток, при нажатии на которые внизу экрана должна появляться информация о них. Приложение должно позволять пользователю осуществлять фильтрацию родников по указанным критериям. Также должна иметься возможность поиск наилучшего варианта с помощью расстановки приоритета по параметрам. Помимо этого, в приложении должна присутствовать поисковая строка и возможность создания маршрута.

ГЛАВА 2. ПРОЕКТИРОВАНИЕ МОБИЛЬНОГО ПРИЛОЖЕНИЯ

2.1 Выбор средств разработки

На сегодняшний день для создания мобильных приложений существует два вида средств разработки: средства разработки для создания нативных мобильных приложений и средства создания web-приложений адаптированных под мобильные приложения.

Задачей данной выпускной квалификационной работы является создание нативного приложения, которое позволяет использовать особенность смартфонов для определения геолокации и не требует постоянного подключения к интернету в отличие от web-приложения.

Так как доля Android занимает более 70% на рынке операционных систем, можно утверждать, что она является самой популярной операционной системой для мобильных устройств. Именно поэтому данная платформа была выбрана для создания приложения «Родники Святого Белогорья». Теперь необходимо осуществить выбор среды разработки.

Наиболее популярными средами разработки мобильного приложения под ОС Android являются Android Studio, Eclipse и NetBeans IDE. Данные программные продукты необходимо проанализировать по следующим критериям: функциональность, удобство интерфейса, требовательность к системе и наличие встроенных компонентов тестирование приложения.

Первой будет рассмотрена среда разработки Android Studio.

Android Studio является интегрированной средой разработки (IDE) производства Google, с помощью которой разработчикам становятся доступны инструменты для создания приложений на платформе Android OS. В Studio содержатся инструменты для разработки решений для смартфонов и планшетов, а также новые технологические решения для Android TV, Android Wear, Android Auto, Glass и дополнительные контекстуальные модули. Решения для Android разрабатываются в Android Studio с использованием

Java или C++. В основе рабочего процесса Android Studio заложен концепт непрерывной интеграции, позволяющий сразу же обнаруживать имеющиеся проблемы [1].

Для создания пользовательского интерфейса (UI) в Android Studio присутствуют шаблоны основных макетов и компонентов, ориентируемые на задачу, которую должно выполнять приложение. Разработка интерфейса производится Drag-and-Drop методом, но есть возможность использовать XML. Также присутствует функция предпросмотра макета на нескольких конфигурациях экрана.

Интерфейс библиотек приложения имеет вид выпадающего древа и под него приходится отводить очень много места в общем интерфейсе, в противном случае, информация становится нечитаемой.

Данная среда разработки является очень требовательной к технической составляющей ЭВМ. Минимальное количество оперативной памяти, требуемой для данного продукта, составляет 3 гигабайта. Однако для комфортной работы с данной программой рекомендуемое количество памяти составляет 8 гигабайт. Именно по этой причине среда Android Studio функционирует не очень быстро, особенно при параллельной работе других приложений.

Android Studio содержит встроенный компонент для тестирования приложения – Android Emulator [15].

Далее будет рассмотрена среда разработки Eclipse.

Eclipse является свободной интегрированной средой разработки модульных кроссплатформенных приложений. Развивается и поддерживается Eclipse Foundation. Включает в свой функционал не только средства для разработки мобильных приложений, но и web-приложений. Используя данную среду разработки можно программировать на множестве языков, таких как Java, C и C++, PHP, Perl, Python, Cobol и других [2]. Имеется возможность синхронизации разных ПК для разработки одного проекта посредствам облачных сервисов. Для расширения функционала

присутствует возможность подключения дополнительных плагинов. Так же существует возможность написания собственных плагинов и их использования без получения лицензии или обязательного предоставления разработки на рынке [22].

В Eclipse отсутствуют шаблоны, а также готовые объекты для разработки интерфейса. Существует встроенный помощник для написания простого приложения «Hello world!». Интерфейс является удобным и понятным. Панель библиотек имеет древовидную структуру, существует возможность полностью сворачивать неиспользуемые в данный момент окна.

Системные требования для Eclipse разработчиком не описаны, но при использовании данной среды разработки на ПК средней производительности затруднений не было.

Встроенные компоненты для тестирования приложения отсутствуют.

Последней будет рассмотрена среда разработки NetBeans IDE.

NetBeans IDE является свободной интегрированная среда разработки приложений (IDE) на языках программирования Java, Python, PHP, JavaScript, C, C++, Ада и ряда других. Данный программный продукт поддерживает плагины, позволяя разработчикам расширять возможности среды. Встроенный отладчик полностью отсутствует, но имеется возможность подключения удаленного отладчика через сеть Интернет [5].

Последние версии NetBeans IDE поддерживают рефакторинг, профилирование, выделение синтаксических конструкций цветом, автодополнение набираемых конструкций на лету и множество предопределённых шаблонов кода.

Интерфейс очень похож на интерфейс Eclipse, за исключением отсутствия окна отладки, и наличия разметки номеров строк.

Требования к ЭВМ невысокие. Для минимальной работы продукта требуется 512 мегабайт оперативной памяти, для более комфортной работы рекомендуется использовать компьютер с 2 гигабайтами [21].

Встроенные компоненты для тестирования приложения полностью отсутствуют.

Краткие выводы анализа сред разработки мобильных приложений приведены в табл. 2.1.

Таблица 2.1

Выводы по анализу сред разработки

	Функциональность	Удобство интерфейса	Требовательность к системе	Тестирование приложения
Android Studio	+	±	±	+
Eclipse	±	+	+	−
NetBeans IDE	±	+	+	−

Следуя полученным заключениям, можно прийти к выводу о том, что каждый из описанных программных продуктов подходит для различных направленностей и предпочтений при разработке мобильных приложений.

Так как для разработки мобильного приложения «Родники Святого Белогорья» необходим мощный инструмент, который не требует временных затрат для поиска или разработки дополнительных модулей, то остановим выбор на Android Studio.

Приложение «Родники Святого Белогорья» должно содержать различную информацию об источниках, поэтому для хранения и предоставления данных пользователю необходимо спроектировать базу данных и интегрировать её в приложение. Исходя из того, что объём хранимой информации относительно небольшой и приложение не предполагает большой нагрузки и специфичной работы с данными, хранить данные целесообразно на устройстве, а не на сервере. Для этого Android устройства поддерживают работу с SQLite [10].

SQLite является компактной встраиваемой СУБД. Слово «встраиваемый» означает, что SQLite не использует парадигму клиент-

сервер, то есть движок SQLite не является отдельно работающим процессом, с которым взаимодействует программа, а представляет собой библиотеку, с которой программа компонуется, и движок становится составной частью программы. Таким образом, в качестве протокола обмена используются вызовы функций (API) библиотеки SQLite. Такой подход уменьшает накладные расходы, время отклика и упрощает программу [6].

Для удобства визуализации структуры данных выбрано CASE-средство ERwin Data Modeler [3].

2.2 Создание прототипа мобильного приложения

Создание прототипа мобильного приложения – это создание макета, модели будущего приложения для того, чтобы определить правильность структуры приложения, его функциональности и, в целом, концепции приложения.

Прототип обладает свойством устранять недопонимания между различными специалистами (менеджер, руководитель, дизайнер, программист, клиент), вовлеченными в проект, структурировать мысли и предотвращать ошибки и выполнение лишней работы еще на ранних стадиях разработки [9].

Для создания прототипа приложения «Родники Святого Белогорья» был использован специальный веб-инструмент для постраничного прототипирования – Marvel [4].

Приложение «Родники Святого Белогорья» является картографическим, поэтому при загрузке должна появиться карта с обозначенной красной меткой местоположения пользователя и с адресом в нижней части экрана. Также на карте должны присутствовать метки, обозначающие источники ключевой воды. Их лучше всего отметить синим цветом. В верхней части экрана должно располагаться название приложения,

а немного ниже – поисковая строка. Так как в приложении должна присутствовать возможность фильтрации родников и поиска наиболее подходящего варианта по критериям, то необходимо разместить соответствующие элементы в виде квадратных кнопок. Для удобства это лучше сделать в верхней части экрана справа. Элементы для работы с картой (определение местоположения, построение маршрута, увеличение и уменьшение карты) представим в виде круглых кнопок, расположенных в правой части экрана в столбик.

На основе вышесказанного для удобства при реализации необходимо создать прообраз конечного продукта.

Прототип отображаемого экрана при входе в приложение изображен на рис. 2.1.

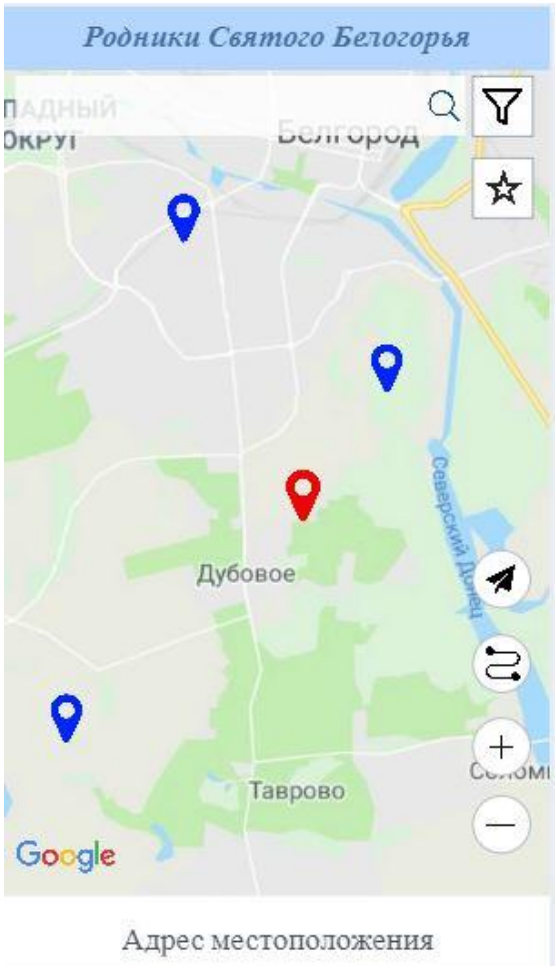

Рис. 2.1. Прототип экрана при входе в приложение

Далее необходимо создать прототипы поведения приложения при использовании.

При нажатии на синюю метку информация о местоположении, которая расположена в нижней части экрана, должна замениться на основные сведения об источнике. А при нажатии на кнопку, которая отвечает за составление маршрута, должен отображаться путь от местоположения пользователя к источнику.

Прототип взаимодействия с меткой изображен на рис. 2.2.

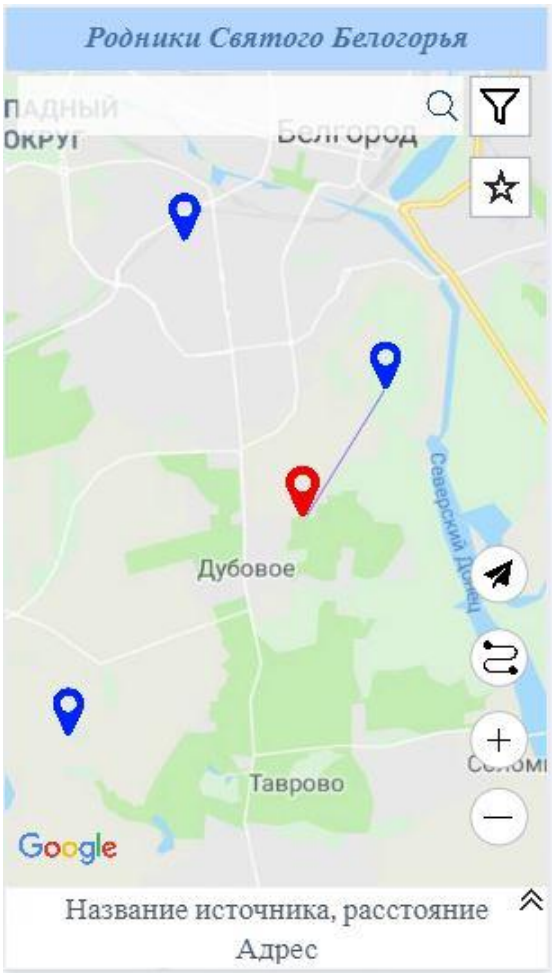

Рис. 2.2. Прототип взаимодействия с меткой

В информативном блоке в правом верхнем углу присутствуют стрелочки вверх, которые отвечают за отображение полной информации о роднике.

Прототип взаимодействия с информативным блоком представлен на рис. 2.3.

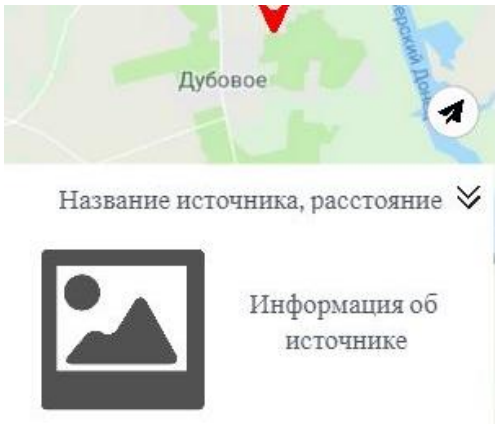

Рис. 2.3. Прототип взаимодействия с информативным блоком

При выборе кнопки фильтрации должно появляться окно соответственного назначения.

Прототип взаимодействия с фильтром представлен на рис. 2.4.

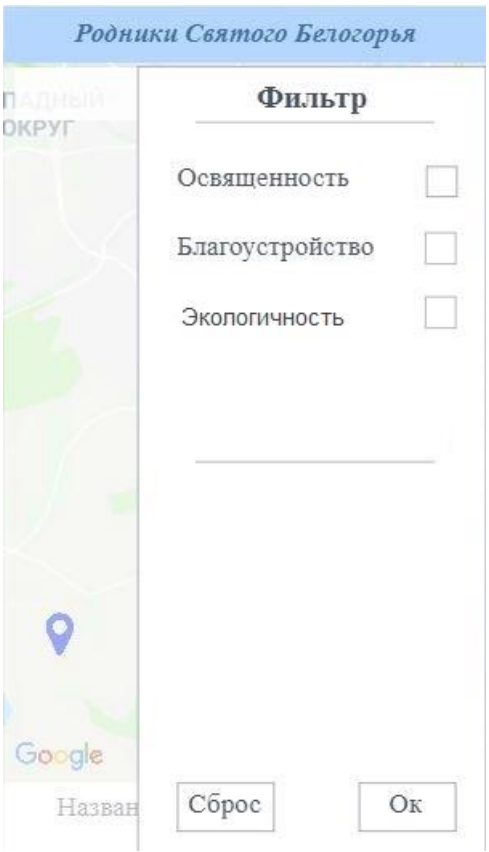

Рис. 2.4. Прототип взаимодействия с фильтром

При нажатии на кнопку поиска наиболее подходящего варианта по критериям должно появиться окно для расстановки приоритетов. Приоритеты пользователь расставляет, нажимая на флажки рядом с указанными критериями.

Прототип взаимодействия с кнопкой поиска наиболее подходящего родника приведён на рис. 2.5.

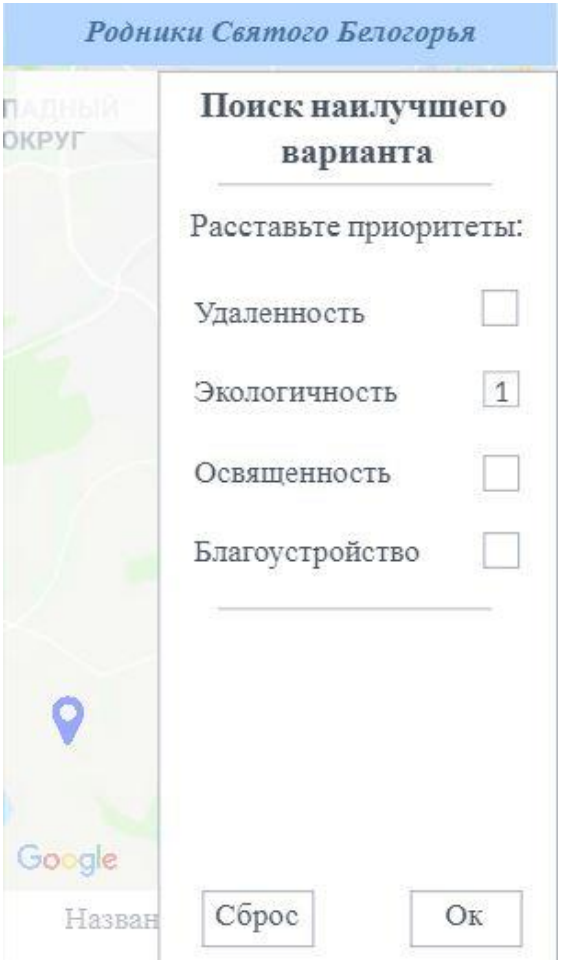

Рис. 2.5. Прототип для поиска наилучшего варианта

Таким образом, полученные прообразы мобильного приложения «Родники Святого Белогорья» будут использованы в дальнейшем при реализации продукта.

2.3 Выбор метода поддержки принятия решения

В ходе создания прототипа мобильного приложения была выявлена необходимость в осуществлении выбора метода поддержки принятия решения для расстановки приоритетов.

При разработке мобильного приложения «Родники Святого Белогорья» для поиска наилучшего варианта необходимо расставить приоритеты по следующим критериям: удаленность, экологичность, освященность, благоустройство. Первый из перечисленных критериев является количественным, остальные качественные. Поэтому при выборе метода нужно учитывать то, что он должен поддерживать работу как с численными, так и с лингвистическими значениями критериев.

Для решения указанной задачи необходимо правильно подобрать соответствующий метод. Для этого нужно понимать все особенности задачи, а также особенности и ограничения методов, которые предполагается применять.

Далее будет рассмотрен выбор наиболее подходящего метода.

Методы можно классифицировать по разным признакам. Один из вариантов базируется на различиях в типе получаемой от эксперта информации.

По содержанию и типу получаемой экспертной информации различают:

– методы, не требующие экспертной информации;

– методы, требующие информацию о предпочтениях на множестве критериев (качественная информация, количественная оценка предпочтительности критериев, количественная информация о замещениях;

– методы, требующие информацию о предпочтительности альтернатив (оценка предпочтительности парных сравнений;

– методы, требующие информацию о предпочтениях на множестве критериев и о последствиях альтернатив (отсутствие информации о

предпочтениях, количественная и/или интервальная информация о последствиях, качественная информация о предпочтениях и количественная информация о последствиях) [19].

Отсюда можно выделить четыре укрупненные группы методов. Первые три группы используются для поддержки принятия решения в условиях определенности. Четвертая группа методов применяется для поддержки принятия решения в условиях неопределенности. К данной группе методов относятся методы с дискретизацией неопределённости, стохастическое доминирование, методы принятия решений в условиях риска и неопределённости на основе глобальных критериев, метод анализа иерархий, методы теории нечётких множеств. Эти методы позволяют учитывать многокритериальность задачи принятия решения, когда оптимальная альтернатива должна в наилучшей степени соответствовать всем критериям. Кроме того, эти методы позволяют работать с такими критериями, которые измеряются в различных шкалах. Они представляют наибольший интерес при решении задачи принятия решения в условиях неопределенности.

Среди перечисленных выше методов из последней группы, наибольший интерес представляют методы анализа иерархий и теории нечетких множеств. Эти методы универсальны, учитывают отбор альтернатив по многим критериям в условиях неопределенности. Сами альтернативы могут быть элементами дискретного или непрерывного множества. Порядок предоставления информации от экспертов в этих методах прост и понятен.

В итоге, можно не рассматривать те методы, которые относятся к первым трем группам, а сосредоточиться на тех, которые подходят в данной ситуации с учетом принятия решения в условиях неопределенности.

Для осуществления окончательного выбора необходимо рассмотреть методы анализа иерархий и теории нечетких множеств более детально.

Первым будет рассмотрен метод анализа иерархий. Данный метод достаточно широко применяется в практике поддержки принятия решения.

Он является основой, которая позволяет решать задачи многокритериального выбора альтернатив в условиях неопределенности.

В данном методе предусматривается декомпозиция проблемы принятия решения на простые элементы, которые обрабатываются лицом, принимающим решение. Как итог, можно определить значимость альтернатив относительно заранее определенных критериев. Относительный приоритет альтернатив представляется в виде вектора. Значения вектора представляют собой оценки по относительной шкале. Такие значения являются жесткими оценками [20].

Иерархическая декомпозиция задачи принятия решения – это одно из основных применений метода анализа иерархий. Метод позволяет проранжировать альтернативы с точки зрения привлекательности их для лица, принимающего решения.

Метод анализа иерархий может послужить надстройкой для тех методов, которые позволяют разрешить задачи с плохой формализацией. В таких случаях больше будут использоваться опыт и интуиция экспертов. Также метод предоставляет удобные средства обработки экспертной информации. Средства, предоставляемые методом во многих случаях, работают лучше, чем сложные математические расчеты [24].

Главное, что происходит в процессе применения метода анализа иерархий – это попарное сравнение альтернатив относительно критериев. Для оценки доминирования одной альтернативы другой, применяется 9-ти бальная шкала, которую Т. Саати называет фундаментальной. В ней присутствуют следующие степени предпочтения критериев: равная, слабая средняя, выше средней, умеренно сильная, сильная, очевидная и очень сильная.

Каждый критерий требует построения матрицы парных сравнений. Порядок такой матрицы определяется количеством ранжируемых альтернатив.

Метод анализа иерархий имеет ряд недостатков, которые препятствуют его использованию при построении СППР. Например, если сравнение альтернатив осуществляется по количественному критерию, то нельзя получить оценку экспертов непосредственно самой величины, не учитывая контекст альтернативы [7].

Далее будет рассмотрен метод теории нечетких множеств. Можно разделить все методы построения функций принадлежности нечетких множеств на два класса: прямые и косвенные. Строить функции принадлежности может как один эксперт, так и группа. В последнем случае необходимо согласовывать мнения разных экспертов [23].

Прямые методы характеризуются тем, что эксперт задает непосредственно для каждого $x \in X$ значение функции принадлежности. В большинстве случаев, прямые методы используются для тех свойств, которые измеримы с помощью количественной шкалы. Для процесса построения функции принадлежности нечеткого множества вводится специальное название – фаззификация [16].

В основном применяются два прямых метода. В одном из них предлагается зафиксировать ядро и носитель нечеткого множества, а другие точки непосредственно задавать на координатной плоскости. Второй используется тогда, когда базовое множество непрерывно. Из определённого набора стандартных графиков выбирается тот, форму которого эксперт считает наиболее подходящей. Далее эксперт задает параметры кривой, которые при необходимости в дальнейшем уточняются.

Прямым методам свойственен один общий недостаток: человеку свойственно ошибаться, поэтому результаты оценок экспертов в этой группе методов имеют достаточную долю субъективности [17].

Для снижения субъективности, при построении функции принадлежности нечеткого множества используются косвенные методы. В этом случае рассмотрению подлежат качественные факторы, а базовое множество неупорядоченно. Среди таких методов наиболее известен метод

попарных сравнений, который был рассмотрен ранее.

Исходными данными являются множества альтернатив и критериев. Предполагается наличие как количественных, так и качественных критериев. Каждому из этих критериев ставится в соответствие нечеткое множество, базовым множеством которого является набор альтернатив:

$$\widetilde{K_i} = \left\{\frac{\mu\widetilde{K}_i(S_1)}{S_1}, \frac{\mu\widetilde{K}_i(S_2)}{S_2}, \cdots, \frac{\mu\widetilde{K}_i(S_m)}{S_m}\right\}, \quad (2.1)$$

где $\mu\tilde{k}_i(s_j) \in [0, 1]$ – значение, характеризующее уровень оценки альтернативы s_j по критерию k_i. Иными словами, это степень принадлежности, показывающая полноту соответствия альтернативы данному критерию [26].

Получение значений степени принадлежности зависит от характера критерия: количественный он или качественный.

После получения оценок альтернатив по критериям, определяется лучшая альтернатива. Для каждой альтернативы s_j значение функции принадлежности результирующему множеству определяется как:

$$\mu\widetilde{D}(S_j) = min\left(\mu\widetilde{K}_1(S_j), \mu\widetilde{K}_2(S_j), \dots, \mu\widetilde{K}_n(S_j)\right). \quad (2.2)$$

Наилучшим вариантом будет тот, у которого значение функции принадлежности будет максимальной:

$$s^* = \arg\left(\max\left(\mu\widetilde{D}(s_j)\right)\right), j = 1, 2, \dots, m. \quad (2.3)$$

Такой подход можно использовать, если все критерии имеют одинаковую важность. Однако это условие далеко не всегда выполняется. Для того чтобы оценить важность критериев при некоторых условиях, используется матрица парных сравнений. Определив попарную степень

доминирования, получаем собственный вектор $V = \{v_i\}, i = 1, \dots, n$, элементы которого выступают в качестве коэффициентов относительной важности критериев:

$$\alpha_i = v_i, \sum \alpha_i = 1, \ i = 1, 2, \dots, n. \quad (2.4)$$

Таким образом, была осуществлена классификация методов многокритериального выбора альтернатив. Более подробно рассмотрены метод анализа иерархий метод теории нечетких множеств.

Метод анализа иерархий достаточно популярен, однако в работе принято решение отказаться от него, однако использовать такой его элемент, как парные сравнения по шкале Т. Саати.

Также были рассмотрены элементы теории нечетких множеств, которые выступают в качестве элемента решения задачи многокритериального выбора. Определены наиболее распространенные способы фаззификации количественных и качественных критериев. Рассмотрены особенности метода многокритериального выбора альтернатив на основе нечетких множеств в условиях равной и неравной важности критериев, который и будет далее использоваться для решения поставленной задачи.

2.4 Проектирование базы данных

Базой данных называют упорядоченную совокупность данных, предназначенных для хранения, накопления и обработки с помощью ЭВМ.

Перед созданием базы данных необходимо определить, из каких таблиц она должна состоять, какие данные нужно поместить в каждую таблицу и как связать таблицы. Эти вопросы решаются на этапе проектирования базы данных.

Основными этапами проектирования баз данных являются:

– концептуальное (инфологическое) проектирование;
– логическое (даталогическое) проектирование;
– физическое проектирование.

Все этапы проектирования БД подразумевают создание моделей данных об интересующей предметной области.

Моделирование данных упрощает понимание смысла элементов данных, способствует более плодотворному общению пользователей и разработчиков [18].

Инфологическим (концептуальным) проектированием называют процесс создания внешней (инфологической) модели данных о предметной области, не зависящей от любых физических аспектов её представления.

Инфологическая модель должна включать такое формализованное описание предметной области, которое будет «читабельно» не только для специалистов по базам данных, но и сторонних людей. Это описание должно быть настолько емким, чтобы можно было оценить глубину и корректность проработки проекта базы данных.

Инфологическое проектирование, прежде всего, связано с попыткой представления семантики, то есть смыслового содержания предметной области в модели базы данных и отображения связей между отдельными сущностями [14].

Инфологическая модель базы данных, содержащей информацию о родниках Белгородской области, представлена на рис. 2.6.

Рис. 2.6. Инфологическая модель базы данных

Даталогическим (логическим) проектированием базы данных называют организацию данных, выделенных на предыдущем этапе проектирования в форму, принятую в выбранной СУБД. Даталогическим проектированием является создание схемы базы данных на основе конкретной модели данных, например, реляционной [12].

Даталогическая модель – это модель логического уровня системы, представляющая собой отображение логических связей между элементами базы данных, независимо от их содержания и среды хранения.

Для реляционной модели данных даталогическая модель представляет собой набор схем отношений, обычно с указанием первичных ключей, а также связей между отношениями, представляющих собой внешние ключи.

Исходными данными для даталогического проектирования является инфологическая модель предметной области.

На этапе логического проектирования учитывается специфика конкретной модели данных, но может не учитываться специфика конкретной СУБД [11].

Для логического и физического проектирования базы данных было использовано CASE-средство AllFusion ERwin Data Modeler.

Логическая модель базы данных, содержащей информацию о родниках Белгородской области, представлена на рис. 2.7.

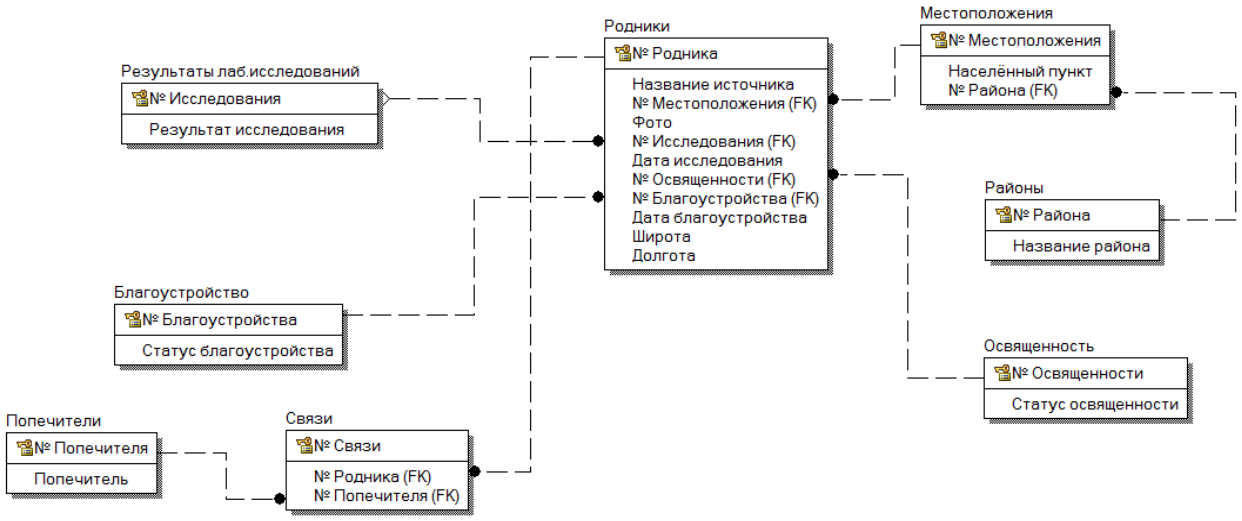

Рис. 2.7. Логическая модель базы данных

Следующим этапом проектирования базы данных является физическое проектирование. Физическим проектированием базы данных называют процесс подготовки описания реализации базы данных на вторичных запоминающих устройствах.

Между логическим и физическим проектированием существует постоянная обратная связь, так как решения, принимаемые на этапе физического проектирования с целью повышения производительности системы, способны повлиять на структуру логической модели данных. ERwin объединяет эти этапы проектирования в единую диаграмму, имеющую несколько уровней представления [25].

Приступая к физическому проектированию базы данных, прежде всего, необходимо выбрать конкретную целевую СУБД. Как правило, основной целью физического проектирования базы данных является описание способа физической реализации логического проекта базы данных.

Физическая модель БД отображает таблицы и их названия, поля таблиц, их типы, а также связи между таблицами, как показано на рис. 2.8.

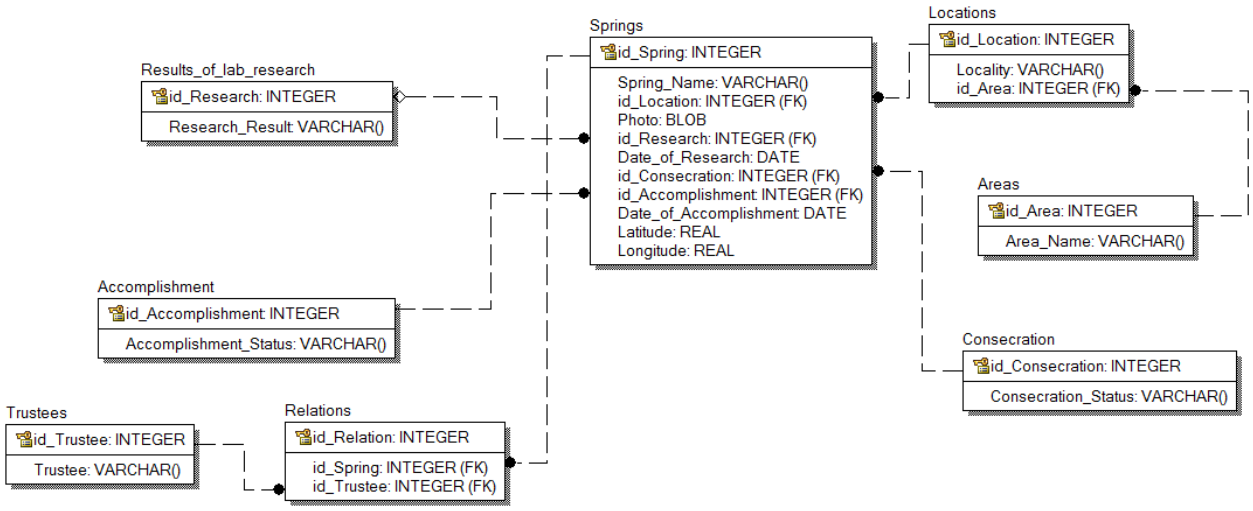

Рис. 2.8. Физическая модель базы данных

Таким образом, на основе результатов проектирования можно выполнить построение базы данных, содержащей информацию о родниках Белгородчины.

ГЛАВА 3. РАЗРАБОТКА И ТЕСТИРОВАНИЕ МОБИЛЬНОГО ПРИЛОЖЕНИЯ

3.1 Разработка информационного обеспечения

Разработка информационного обеспечения включает подготовку данных, содержащих информацию, которая является необходимой для решения задачи, и анализ этой информации.

Для организации информационной базы будет использована реляционная СУБД. Выбранной СУБД является SQLite. Здесь имеются следующие классы хранения данных: INTEGER, REAL, TEXT, BLOB, NUMERIC.

Класс хранения – более широкое понятие, чем тип данных. К примеру, класс хранения INTEGER включает 6 различных типов целочисленных данных различной длины. Таким образом, тип данных VARCHAR будет содержаться в классе TEXT, а тип данных DATE – в классе NUMERIC.

Числовые аргументы в скобках после имени типа (например, VARCHAR(20)) игнорируются SQLite, так как он не накладывает ограничений на длину строк бинарных или числовых значений.

В процессе проектирования базы данных была выявлена необходимость в создании восьми таблиц, на основе которых реализуется база данных.

Таблица «areas» предназначена для хранения районов Белгородской области. Она содержит в себе следующие данные:

– «id_area» – первичный ключ класса INTEGER;
– «area_name» – название района класса TEXT.

Таблица «locations» предназначена для хранения населённых пунктов Белгородской области. Она содержит в себе следующие данные:

– «id_location» – первичный ключ класса INTEGER;
– «locality» – название населенного пункта класса TEXT;

– «id_area» – внешний ключ класса INTEGER, ссылающийся на таблицу «areas» и содержащий район Белгородской области.

Таблица «results_of_lab_research» предназначена для хранения результатов лабораторных исследований анализа воды. Она содержит в себе следующие данные:

– «id_research» – первичный ключ класса INTEGER;

– «research_result» – результат исследования класса TEXT.

Таблица «accomplishment» предназначена для хранения статуса благоустройства родников. Она содержит в себе следующие данные:

– «id_accomplishment» – первичный ключ класса INTEGER;

– «accomplishment_status» – статус благоустройства класса TEXT.

Таблица «consecration» предназначена для хранения статуса освященности родников. Она содержит в себе следующие данные:

– « id_consecration» – первичный ключ класса INTEGER;

– «consecration_status» – статус освященности класса TEXT.

Таблица «springs» предназначена для хранения информации о родниках. Она содержит в себе следующие данные:

– « id_spring» – первичный ключ класса INTEGER;

– «spring_name» – название родника класса TEXT;

– «id_location» – внешний ключ класса INTEGER, ссылающийся на таблицу «locations» и содержащий населённый пункт Белгородской области;

– «photo» – фото родника класса BLOB;

– «id_research» – внешний ключ класса INTEGER, ссылающийся на таблицу «results_of_lab_research» и содержащий результат исследования воды;

– «date_of_research» – дата проведения исследования класса NUMERIC;

– «id_consecration» – внешний ключ класса INTEGER, ссылающийся на таблицу «consecration» и содержащий статус освященности родника;

– «id_accomplishment» – внешний ключ класса INTEGER, ссылающийся на таблицу «accomplishment» и содержащий статус благоустройства родника;

– «date_of_accomplishment» – дата благоустройства класса NUMERIC;

– «latitude» – широта родника класса REAL;

– «longitude» – долгота родника класса REAL.

Таблица «trustees» предназначена для хранения попечителей родника. Она содержит в себе следующие данные:

– « id_trustee» – первичный ключ класса INTEGER;

– «trustee» – информация о попечителе класса TEXT.

Таблица «relations» предназначена для хранения связей между родниками и попечителями. Данная таблица содержит в себе следующие данные:

– «id_relation» – первичный ключ класса INTEGER;

– «id_spring» – внешний ключ класса INTEGER, ссылающийся на таблицу «springs» и содержащий родник;

– «id_trustee» – внешний ключ класса INTEGER, ссылающийся на таблицу «trustees» и содержащий попечителя родника.

3.2 Программная реализация мобильного приложения

Мобильное приложение «Родники Святого Белогорья» является картографическим, поэтому разработку следует начать с подключения карты.

Для подключения приложения к сервисам Google необходимо подключить следующие компоненты SDK менеджера: Google Play Services и

Google Repository. Далее необходимо создать API ключ с помощью Google Maps Android API и добавить его в приложение. Для этого в файле манифеста AndroidManifest.xml следует объявить две декларации meta-данных. Данный фрагмент кода приведён в листинге 3.1.

Листинг 3.1. Объявление деклараций meta-данных

```
<meta-data
android:name="com.google.android.gms.version"
android:value="@integer/google_play_services_version"/>
<meta-data
android:name="com.google.android.maps.v2.API_KEY"
android:value="AIzaSyAgtSHXUOsqTXBDV3M6KGtuCAqcIIaBpzk" />
```

Конец листинга 3.1

Здесь первый meta-тег встраивает данные версии библиотеки Google Play Services, добавленной в приложение, а второй содержит ранее созданный API ключ.

Для отображения карты на экране необходимо добавить фрагмент кода, приведенный в листинге 3.2.

Листинг 3.2. Отображение карты на экране

```
public class MainActivity extends AppCompatActivity {
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
} }
```

Конец листинга 3.2

Также необходимо добавить конфигурации, позволяющие показывать местоположение пользователя на карте. Данный фрагмент кода приведён в листинге 3.3.

Листинг 3.3. Конфигурации для определения местоположения

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION" />
```

Конец листинга 3.3

Здесь подключается разрешение для доступа в интернет, для доступа к сети и разрешения для определения местоположения.

Для установки позиции и масштаба отображения карты необходимо добавить фрагмент кода, приведённый в листинге 3.4.

Листинг 3.4. Установка позиции и масштаба отображения карты

```
LatLng latLng = new LatLng(myLocation.getLatitude(),
myLocation.getLongitude());
myMap.animateCamera(CameraUpdateFactory.newLatLngZoom(latLng, 13));
CameraPosition cameraPosition = new CameraPosition.Builder()
.target(latLng)
.zoom(15)
.bearing(90)
.tilt(40)
.build();
```

Конец листинга 3.4

Для удобства просмотра местонахождения пользователя необходимо добавить маркер. Фрагмент кода данной процедуры приведен в листинге 3.5.

Листинг 3.5. Добавление маркера на карту

```
MarkerOptions option = new MarkerOptions();
option.position(latLng);
Marker currentMarker = myMap.addMarker(option);
currentMarker.showInfoWindow();
```

Конец листинга 3.5

Далее необходимо подключить существующую базу данных для дальнейшей работы с ней. Для этого нужно добавить файл базы данных в проект. Это можно сделать с помощью создания папки в проекте и копирования ранее созданного файла springs.db. Для открытия и подготовки БД в Android используется наследник класса SQLiteOpenHelper – DatabaseHelper.

Для работы с базой данных необходимо задать путь к базе данных приложения. Фрагмент кода данной процедуры приведен в листинге 3.6.

Листинг 3.6. Указание пути к БД

```
private static String DB_NAME = "springs.db";
private static String DB_PATH = "";
private static final int DB_VERSION = 1;
```

Конец листинга 3.6

Здесь DB_NAME – имя файла БД, DB_PATH – путь к БД (можно оставить пустым), DB_VERSION – номер версии БД, который необходимо менять после обновления базы данных.

Далее необходимо подключиться к базе данных. Для этого нужно создать соответствующие переменные. Фрагмент кода данной процедуры приведен в листинге 3.7.

Листинг 3.7. Создание переменных для работы с БД

```
private SQLiteDatabase mDataBase;
private final Context mContext;
private boolean mNeedUpdate = false;
```

Конец листинга 3.7

Теперь необходимо создать конструктор, который будет принимать, а также сохранять ссылку на переданный контекст для доступа к ресурсам приложения. Фрагмент создания конструктора приведен в листинге 3.8.

Листинг 3.8. Создание конструктора для доступа к ресурсам

```
public DataBaseHelper(Context context) {
super(context, DB_NAME, null, 1);
this.mContext = context;
}
```

Конец листинга 3.8

Следующим шагом будет создание пустой базы данных, которая будет перезаписана существующей. Фрагмент создания и перезаписи базы данных приведен в листинге 3.9.

Листинг 3.9. Создание базы данных

```
public void createDataBase() throws IOException{
boolean dbExist = checkDataBase();
if(dbExist){ }
```

Конец листинга 3.9

Листинг 3.9. Создание базы данных (продолжение)

```
else{
this.getReadableDatabase();
try {
copyDataBase();
} catch (IOException e) {
throw new Error("Error copying database"); }
}}
```

Конец листинга 3.9

Далее необходимо проверить, существует ли база данных, чтобы избежать повторного копирования файла при каждом открытии приложения. Фрагмент проверки приведен в листинге 3.10.

Листинг 3.10. Проверка БД на существование

```
public boolean checkDataBase(){
File databaseFile = new File(DB_PATH + DATABASE_NAME);
return databaseFile.exists();
}
```

Конец листинга 3.10

Далее необходимо проверить, существует ли база данных, чтобы избежать повторного копирования файла при каждом открытии приложения. Фрагмент проверки приведен в листинге 3.11.

Листинг 3.11. Проверка БД на существование

```
public boolean checkDataBase(){
File databaseFile = new File(DB_PATH + DATABASE_NAME);
return databaseFile.exists(); }
```

Конец листинга 3.11

Наконец, необходимо скопировать существующую базу данных из локальной папки-источника в только что созданную пустую базу данных в системной папке, откуда ее можно получить и обработать. Это делается путем передачи байтового потока. Фрагмент кода данной процедуры приведен в листинге 3.12.

Листинг 3.12. Копирование БД из локальной папки в системную

```
private void copyDataBase() throws IOException{
InputStream myInput = context.getAssets().open(DATABASE_NAME);
String outFileName = DB_PATH + DATABASE_NAME;
OutputStream myOutput = new FileOutputStream(outFileName);
byte[] buffer = new byte[1024];
int length;
while ((length = myInput.read(buffer))>0){
myOutput.write(buffer, 0, length);
}
```

Конец листинга 3.12

Теперь можно открыть базу данных для дальнейшей работы с ней. Фрагмент кода данной процедуры приведен в листинге 3.13.

Листинг 3.13. Открытие базы данных

```
public void openDataBase() throws SQLException{
String myPath = DB_PATH + DB_NAME;
myDataBase = SQLiteDatabase.openDatabase(myPath, null,
SQLiteDatabase.OPEN_READONLY);
}
```

Конец листинга 3.13

Полный листинг кода приведён в приложении.

3.3 Тестирование мобильного приложения

Для проверки работоспособности разработанного мобильного приложения необходимо протестировать продукт.

Тестирование отображаемого экрана при запуске приложения изображено на рис. 3.1.

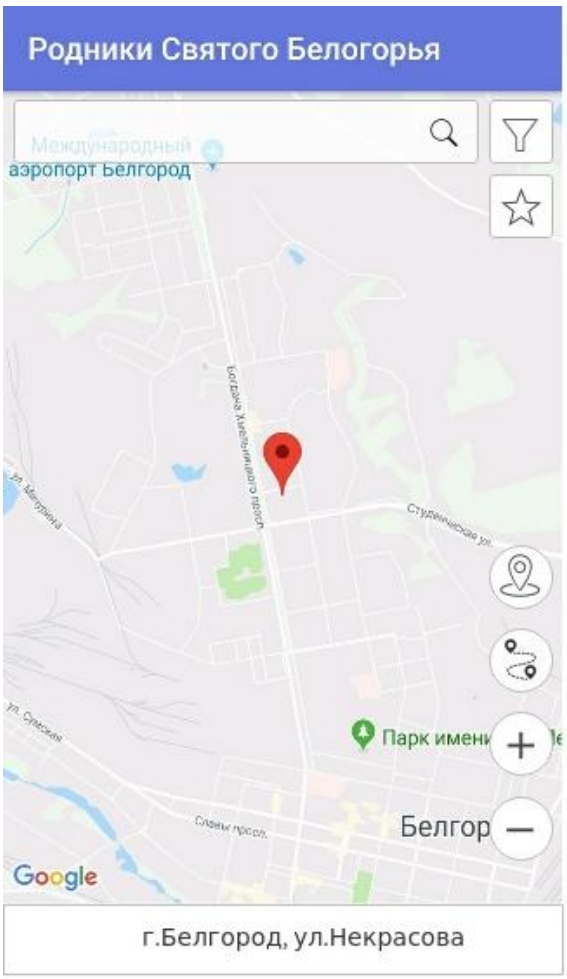

Рис. 3.1. Главный экран при входе

Здесь определяется местоположение пользователя на карте и в информативном блоке отображается адрес.

Для того, чтобы увидеть, где находятся родники, необходимо уменьшить масштаб карты.

Экран с метками расположения родников приведен на рис. 3.2.

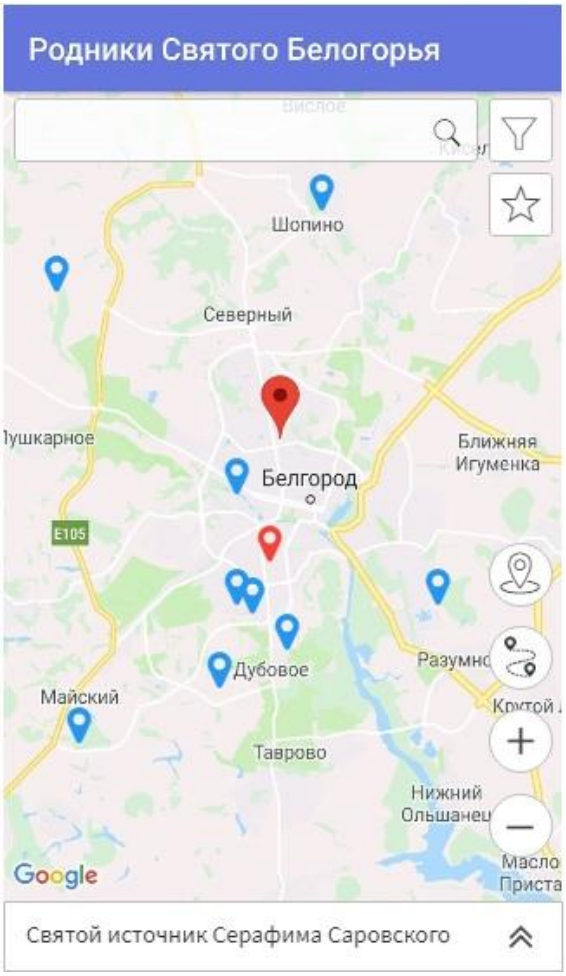

Рис. 3.2. Экран с метками родников

При нажатии на метку в информативном блоке отображается название родника. Для получения полной информации об источнике ключевой воды необходимо нажать на кнопку в нижнем правом углу, которая имеет вид стрелочек вверх. Блок с отображением подробной информации приведен на рис. 3.3.

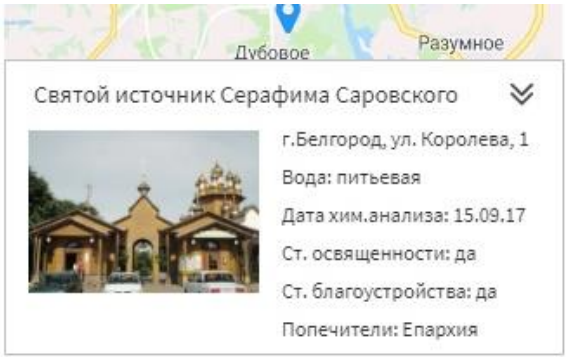

Рис. 3.3. Информативный блок

При нажатии на кнопку фильтрации появляется выезжающее боковое меню, в котором можно выбрать родники с какими характеристиками оставить на карте. Данное действие представлено на рис. 3.4.

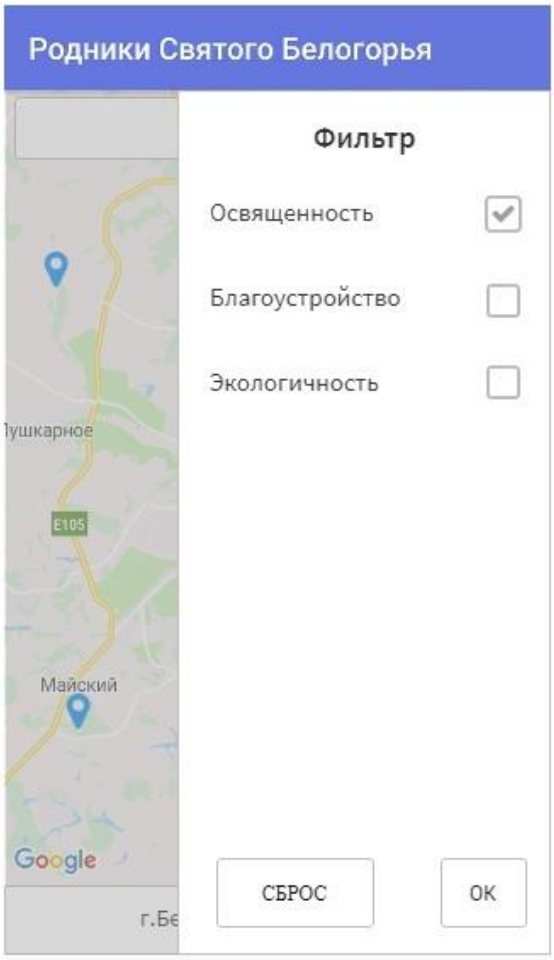

Рис. 3.4. Выезжающее боковое меню фильтрации

После подтверждения выбранных характеристик появляется карта с отфильтрованными родниками. Экран с отфильтрованными данными представлен на рис. 3.5.

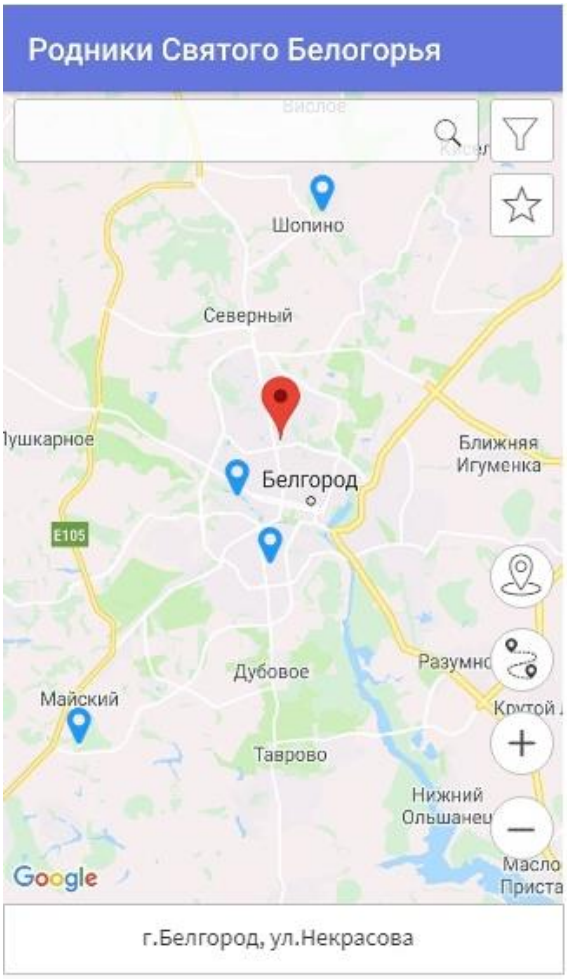

Рис. 3.5. Работа фильтрации данных

Здесь видно, что меток стало меньше. Так как при фильтрации данных пользователь указал, что необходимо предоставить только освященные родники, то на карте отобразились только те источники, которые имеют данный статус.

При нажатии на кнопку выбора наиболее подходящего варианта появляется выезжающее боковое меню, в котором необходимо расставить приоритеты по указанным критериям. Приоритеты необходимо расставлять в порядке возрастания.

Выезжающее боковое меню для поиска наилучшего варианта представлено на рис. 3.6.

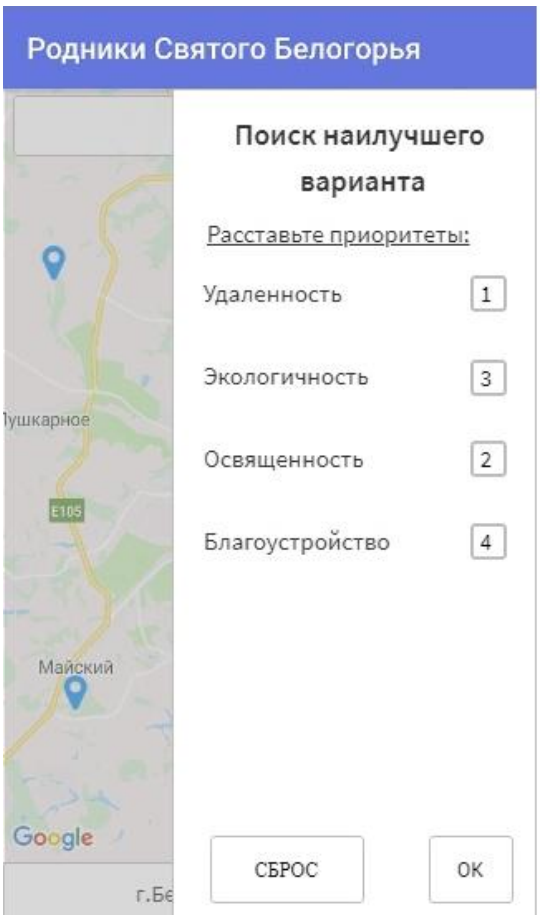

Рис. 3.6. Меню поиска наилучшего варианта

После подтверждения расстановки приоритетов на карте остается источник, наиболее удовлетворяющий потребностям пользователя. Экран с наиболее подходящим вариантом представлен на рис. 3.7.

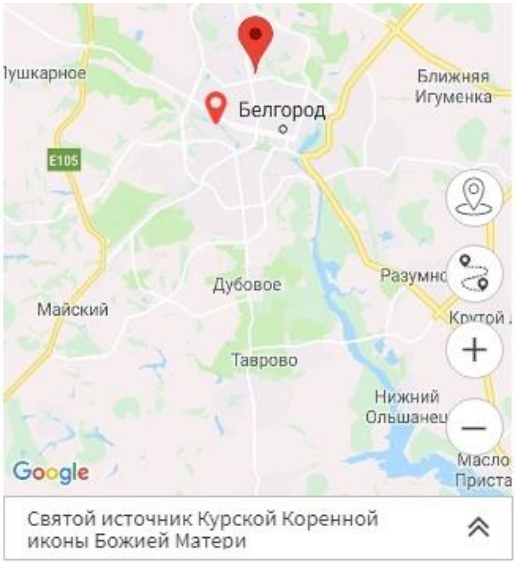

Рис. 3.7. Экран с наиболее подходящим вариантом

ЗАКЛЮЧЕНИЕ

В результате выполнения выпускной квалификационной работы было разработано мобильное приложение «Родники Святого Белогорья».

Также были решены следующие задачи:

1. Анализ предметной области.
2. Определение ключевой направленности разрабатываемого мобильного приложения.
3. Определение требований к разрабатываемому приложению.
4. Проведение обзора и подробный анализ существующих приложений с идентичной направленностью.
5. Проектирование мобильного приложения.
6. Разработка и тестирование мобильного приложения «Родники Святого Белогорья».

Данное приложение помогает осуществить поиск источников ключевой воды на территории Белгородской области, получить информацию о них, отфильтровать по указанным критериям, а также найти наилучший вариант с помощью расстановки приоритета по параметрам. Помимо этого, имеется возможность создания маршрута.

По итогам тестирования разработанного мобильного приложения, можно сделать вывод о том, что приложение полностью удовлетворяет всем требованиям, которые были определены на этапе анализа предметной области.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Android Studio [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Android_Studio, 20.04.2018.
2. Eclipse (среда разработки) [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Eclipse_(среда_разработки), 21.04.2018.
3. ERwin Data Modeler [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/ERwin_Data_Modeler, 05.05.2018.
4. Marvel [Электронный ресурс]. – Режим доступа: https://marvelapp.com, 06.05.2018.
5. NetBeans [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/NetBeans, 21.04.2018.
6. SQLite [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/SQLite, 27.04.2018.
7. Андрейчиков, А.В. Анализ, синтез, планирование решений в экономике / А.В. Андрейчиков, О.Н. Андрейчикова – М.: Финансы и статистика, 2004. – 427 с.
8. Баландин, Р.К. Природа и цивилизация / Р.К. Баландин, Л.Г. Бондарев – М.: Мысль, 1988. – 131 с.
9. Варфел, О.З. Прототипирование / О.З. Варфел – М.: Москва, 2013. – 704 с.
10. Введение в SQLite [Электронный ресурс]. – Режим доступа: https://phpclub.ru/detail/article/sqlight_intro, 27.04.2018.
11. Даталогическое проектирование [Электронный ресурс]. – Режим доступа: http://poznayka.org/s99379t1.html, 26.04.2018.
12. Даталогическое проектирование базы данных [Электронный ресурс]. – Режим доступа: https://lektsia.com/6x229.html, 26.04.2018.
13. Иностранцев, А. А. Источники, ключи или родники / А.А. Иностранцев – СПб.: БХВ-Петербург, 1907. – 42 с.

14. Инфологическое проектирование базы данных [Электронный ресурс]. – Режим доступа: https://studfiles.net/preview/3488655/page:13/, 26.04.2018.

15. Как пользоваться Android Studio [Электронный ресурс]. – Режим доступа: https://losst.ru/kak-polzovatsya-android-studio, 20.04.2018.

16. Конышева, Л.К. Основы теории нечетких множеств / Л.К. Конышева, Д.М. Назаров, 2011. – 127 с.

17. Кофман, А.В. Введение в теорию нечетких множеств / А.В. Кофман – М.: Радио и связь, 1982. – 389 с.

18. Кузнецов, С.Д. Основы баз данных / С.Д. Кузнецов – М.: БИНОМ, 2007. – 471 с.

19. Ларичев, О.И. Системы поддержки принятия решения / О.И. Ларичев, А.Б. Петровский – М.: ВИНИТИ, 1987. – 158 с.

20. Метод анализа иерархий [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Метод_анализа_иерархий, 11.05.2018.

21. Монахов, В.М. Язык программирования Java и среда NetBeans / В.М. Монахов – СПб.: БХВ-Петербург, 2011. – 704 с.

22. Обзор платформы Eclipse [Электронный ресурс]. – Режим доступа: https://hightech.in.ua/content/art-eclipse-platform, 21.04.2018.

23. Основные положения теории нечетких множеств [Электронный ресурс]. – Режим доступа: https://www.kazedu.kz/referat/188079/11, 11.05.2018.

24. Саати, Т.Л. Принятие решений. Метод анализа иерархий / Т.Л. Саати – М.: Радио и связь, 1989. – 298 с.

25. Физическое проектирование базы данных [Электронный ресурс]. – Режим доступа: http://bourabai.ru/dbt/dbms/03.htm, 29.04.2018.

26. Ягер, Р.Р. Нечеткие множества и теория возможностей / Р.Р. Ягер – М.: Радио и связь, 1986. – 150 с.