*Министерство науки и образования Российской Федерации*

*Федеральное государственное бюджетное образовательное учреждение* *высшего профессионального образования*

|  |  |
| --- | --- |
|  | ***«Московский государственный технический университет  имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(МГТУ им. Н.Э. Баумана)*** |

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Компьютерные системы и сети

**Расчетно-пояснительная записка**

**к курсовой работе на тему:**

**«Android-приложение “Weather&News”»**

Студент группы ИУ6-51 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.Ю. Сидоров

(Подпись, дата) (И.О.Фамилия)

Руководитель курсовой работы,

к.т.н., доц. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.Ю. Попов

(Подпись, дата) (И.О.Фамилия)

Москва, 2016г

**РЕФЕРАТ**

Записка 43с., 22 рис., 11 табл., 9 источников, 2 прил.

ANDROID, НОВОСТИ, ПОГОДА, ВЕБ-СЕРВИС, JSON, XML

Объектом разработки является мобильное приложение «Weather&News», которое в наглядной форме предоставляет актуальную информацию о новостях и погоде.

Цель работы ­– Android-приложение, позволяющее:

* выбирать название населенного пункта для прогноза погоды;
* выбирать название рубрики или региона для ленты новостей;
* получать данные с веб-сервисов в виде JSON-объектов и XML-документов и разбирать их с целью извлечения нужной информации;
* демонстрировать прогноз погоды на день или на неделю, исходя из выбранного населенного пункта;
* демонстрировать ленту новостей, исходя из выбранной рубрики или региона;
* записывать последние загруженные данные в соответствующий файл.

В результате разработки было спроектировано и реализовано мобильное приложение на платформе Android, позволяющее просматривать прогноз погоды и ленту новостей.

Пользователем приложения может быть любой владелец смартфонам под управлением операционной системы Android, интересующийся последними сведениями о погоде и новостях.

**СОДЕРЖАНИЕ**

[ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ 4](#_Toc469194018)

[ВВЕДЕНИЕ 5](#_Toc469194019)

[1 Анализ технических требований и уточнение спецификаций 6](#_Toc469194020)

[1.1 Анализ задания и выбор технологии, языка и среды программирования 6](#_Toc469194021)

[1.2 Выбор модели жизненного цикла программного продукта 8](#_Toc469194022)

[1.3 Выбор веб-сервисов, предоставляющих данные 8](#_Toc469194023)

[1.4 Разработка диаграммы вариантов использования 9](#_Toc469194024)

[1.5 Объектная декомпозиция программного продукта 15](#_Toc469194025)

[1.6 Построение концептуальной модели предметной области 16](#_Toc469194026)

[1.7 Исследование версий API платформы Android 17](#_Toc469194027)

[1.8 Выводы 19](#_Toc469194028)

[2 Проектирование структуры программного продукта 20](#_Toc469194029)

[2.1 Разработка интерфейса пользователя 20](#_Toc469194030)

[2.1.1 Разработка структурной схемы приложения 20](#_Toc469194031)

[2.1.2 Разработка структурной схемы меню 21](#_Toc469194032)

[2.1.3 Разработка диаграммы состояний интерфейса 21](#_Toc469194033)

[2.1.4 Разработка форм графического пользовательского интерфейса 23](#_Toc469194034)

[2.2 Разработка диаграмм классов приложения 29](#_Toc469194035)

[2.3 Разработка диаграмм последовательностей 33](#_Toc469194036)

[2.4 Алгоритм процесса загрузки погодных данных 35](#_Toc469194037)

[2.5 Выводы 36](#_Toc469194038)

[3 Выбор стратегии тестирования и разработка тестов 37](#_Toc469194039)

[3.1 Функциональное тестирование 37](#_Toc469194040)

[3.2 Оценочное тестирование 40](#_Toc469194041)

[3.3 Выводы 41](#_Toc469194042)

[ЗАКЛЮЧЕНИЕ 42](#_Toc469194043)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 43](#_Toc469194044)

Приложение А. Техническое задание.

Приложение Б. Руководство пользователя.

# **ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ**

API – Application Programming Interface, интерфейс программирования.

JSON – JavaScript Object Notation, объектная нотация JavaScript.

MVC ­– Model-View-Controller, модель-представление-контроллер.

RSS - Rich Site Summary, обогащённая сводка сайта.

URL – Uniform Resource Locator, универсальный локатор ресурса.

XML - eXtensible Markup Language, расширяемый [язык разметки](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D1%80%D0%B0%D0%B7%D0%BC%D0%B5%D1%82%D0%BA%D0%B8).

# **ВВЕДЕНИЕ**

С каждым годом количество пользователей смартфонов под управлением операционной системы Android увеличивается. Android является самой популярной мобильной операционной системой, его доля на рынке достигает 86% [1]. Очень удобно, когда на смартфоне есть приложение, способное показывать прогноз погоды и последние новости. Это помогает быть в курсе последних событий и планировать свои действия на будущее, опираясь на информацию о погоде. Перечисленные потребности пользователя делают разработку данного приложения актуальной.

В магазине приложений для Android-устройств «Google Play» можно найти аналоги разрабатываемому приложению, однако все они имеют свои недостатки.

Один из самых популярных в русскоязычном сегменте аналогов данного приложения – приложение «Яндекс» - не сохраняет последнюю загруженную информацию, а это значит, что без наличия Интернет-соединения данное приложение окажется бесполезным. К тому же, из-за сложной организации и большого числа элементов, «Яндекс» потребляет немалое количество Интернет-трафика, что тоже можно назвать недостатком.

Другой аналог, «НГС – официальное приложение», тоже не может сохранять последние загруженные данные. К тому же, просмотр новостей требует открытия браузера. Неприятно и обилие рекламы.

«WP24» - еще один аналог, имеет сложный пользовательский интерфейс, потребляет много трафика и имеет плохую локализацию на русский язык.

Разрабатываемое приложение лишено перечисленных выше недостатков: последние загруженные данные сохраняются в соответствующие файлы, что очень удобно, когда нет Интернет-соединения; приложение имеет удобный интерфейс, что облегчает его использование; потребляется минимальное количество трафика, что важно при медленном Интернет-соединении или ограничении количества загружаемого трафика; приложение полностью поддерживает русский язык, что оценит каждый пользователь, основной язык которого - русский.

# **1 Анализ технических требований и уточнение спецификаций**

# **1.1 Анализ задания и выбор технологии, языка и среды программирования**

Для разработки программы было принято решение использовать объектный подход, который позволяет применить концепцию Model-View-Controller (MVC). Чёткое разделение бизнес-логики приложения от его графического интерфейса является основной идеей концепции. Обработкой нажатий клавиш, отправлением запросов к веб-сервисам и управлением потоками данных занимаются контроллеры. Структура данных записывается в модель. Она предоставляет контроллеру данные, которые запросил пользователь. В данном случае очень удобно представить сущности «Новость» и «Погода» в качестве моделей. Представление передаёт шаблон интерфейса контроллеру на обработку.

Т.к. было решено делать мобильное приложение для операционной системы Android, то выбор пал на язык программирования Java, рекомендуемый самой компанией Google в качестве основного языка программирования для этой платформы [2].

Альтернативными вариантами являются такие языки программирования, как C++ и Python, но использование данных языков при разработке приложений для платформы Android требует ряд библиотек специального назначения. К тому же, они не являются нативными языками программирования для данной платформы.

При выборе среды разработки были рассмотрены две из них: Android Studio и Eclipse. После тщательного анализа, выбор пал на Android Studio, т.к.:

* Android Studio рекомендована Google для создания приложений под операционную систему Android [2];
* Android Studio, в отличие от Eclipse, предназначена только для написания приложений под данную операционную систему, соответственно, никаких дополнительных библиотек загружать не надо;
* Более быстрая работа эмулятора в сравнении с Eclipse [2].

Таким образом, в качестве среды разработки была использована среда Android Studio самой последней версии 2.1.2.

В качестве метода проектирования программного продукта был выбран нисходящий подход, т.к:

* интерфейс пользователя разрабатывается в первую очередь, демонстрация которого научному руководителю позволяет уточнить требования к создаваемому приложению;
* сначала выполняемые задачи приложения определяются в общих чертах, затем идет последовательное уточнение структуры задач путем детализации основных функций; это позволяет на раннем этапе выявить и устранить ошибки в проектировании или в программировании приложения

# **1.2 Выбор модели жизненного цикла программного продукта**

Для создания приложения было решено использовать спиральную модель разработки. Концепция данной модели заключается в том, что продукт создается не сразу, а поэтапно, и на каждом этапе получается прототип, реализующий определенные функции приложения [3].

Такой выбор связан с тем, что спиральная модель позволяет в любой момент разработки предоставить очередную рабочую версию приложения, а также проанализировать правильность выполнения уже реализованных задач.

Этапы разработки Android-приложения “Weather&News” были запланированы следующим образом :

1. Создание пользовательского интерфейса.
2. Создание моделей сущностей «Новость» и «Погода».
3. Создание контроллеров для взаимодействия моделей и представлений.
4. Уточнение спецификаций.
5. Реализация записи в файл последних сохраненных данных.
6. Поиск и устранение ошибок и реализация вывода информационных сообщений.

# **1.3 Выбор веб-сервисов, предоставляющих данные**

Исходя из технического задания, приложение должно составлять URL-запрос и затем отправлять его веб-сервису, чтобы получить необходимые данные.

Для загрузки прогноза погоды были выбраны 3 веб-сервиса: openweathermap.org, gismeteo.ru, yahoo.com.

Для использования услуг веб-сервисов необходимо получить ключ API. Openweathermap.org выдает ключ после простой регистрации, ключ действует бессрочно, никаких дополнительных действий для пользования данным веб-сервисом не требуется. Данные предоставляются в формате JSON-объекта.

Веб-сервис gismeteo.ru является одним из самых популярных в русскоязычном Интернете, но для получения ключа API нужно заплатить определенную сумму, что в условиях разработки данного приложения неприемлемо.

Веб-сервис Yahoo.com предоставляет данные в виде XML-документа. Получить ключ API можно после регистрации, но использование услуг данного сервиса подразумевает дополнительные условия в виде упоминания веб-сервиса в приложении, ограничения количества запросов, продления API ключа раз в неделю.

Таким образом, для загрузки прогноза погоды выбор пал на веб-сервис openweathermap.org.

Для загрузки новостей были рассмотрены два веб-сервиса: news.yandex.ru и lenta.ru. Они предоставляют новостные данные в формате RSS-ленты. В данном случае, под RSS-лентой подразумевается семейство XML-форматов, предназначенных для описания лент новостей. Оба веб-сервиса предоставляют свободный доступ к своим ресурсам, но news.yandex.ru обладает большей популярностью и лучшими отзывами пользователей, по сравнению с указанным конкурентом.

Таким образом, для загрузки новостей выбор пал на веб-сервис news.yandex.ru.

# **1.4 Разработка диаграммы вариантов использования**

Разработку спецификаций программного обеспечения начнем с анализа требований к функциональности, указанных в техническом задании. Начнем разработку приложения с разработки диаграммы вариантов использования.

"Вариант использования" - это описание поведения программы при ее взаимодействии с действующим лицом, в нашем случае действующим лицом будет пользователь – человек.

Данная диаграмма поможет на начальном этапе разработки описать взаимодействие пользователя с приложением, определить возможные действия пользователя и их альтернативы.

Анализ особенностей поведения пользователя позволил выделить следующие варианты использования программы:

* просмотр погоды на текущий день;
* просмотр погоды на неделю;
* просмотр новостей по региону;
* просмотр новостей по рубрике.

Таблица 1 – Описание варианта использования *Просмотр погоды на текущий день*

|  |  |
| --- | --- |
| Название варианта | Просмотр погоды на текущий день |
| Цель | Посмотреть прогноз погоды на текущий день |
| Действующие лица | Пользователь |
| Краткое описание | Просмотр прогноза погоды на текущий день по интересующему населенному пункту |
| Тип | Основной |

Таблица 2 – Вариант использования *Просмотр погоды на текущий день*

*Типичный ход событий*

|  |  |
| --- | --- |
| Действие исполнителя | Отклик приложения |
| 1. Пользователь выбирает пункт меню «Погода» | 2. Приложение переходит в меню выбора прогноза погоды |
| 3. Пользователь выбирает пункт меню «На сегодня» | 4. Приложение открывает окно просмотра сведений о погоде и выводит последние сохраненные данные |
| 5. Пользователь вводит название интересующего его населенного пункта и нажимает кнопку «✓» | 6. Приложение отправляет соответствующий запрос веб-сервису openweathermap.org, разбирает JSON-объект, обновляет окно просмотра сведений о погоде и перезаписывает в файл новые данные |

*Альтернатива*:

4. Приложение открывает окно просмотра сведений о погоде и выводит сообщение об отсутствии сохраненных ранее данных.

*Альтернатива 2*:

5. Пользователь не вводит название интересуемого его населенного пункта.

6. Приложение не отправляет запрос веб-сервису и не обновляет окно просмотра сведений о погоде.

*Альтернатива 3*:

5. Пользователь вводит название несуществующего населенного пункта и нажимает кнопку «✓».

6. Приложение выводит сообщение об ошибке в названии населенного пункта.

*Альтернатива 4*:

5. Пользователь не вводит название населенного пункта и нажимает кнопку «✓».

6. Приложение выводит сообщение о пустой строке.

Таблица 3 – Описание варианта использования *Просмотр погоды на неделю*

|  |  |
| --- | --- |
| Название варианта | Просмотр погоды на неделю |
| Цель | Посмотреть прогноз погоды на неделю |
| Действующие лица | Пользователь |
| Краткое описание | Просмотр прогноза погоды на неделю по интересующему населенному пункту |
| Тип | Основной |

Таблица 4– Вариант использования *Просмотр погоды на неделю*

*Типичный ход событий*

|  |  |
| --- | --- |
| Действие исполнителя | Отклик приложения |
| 1. Пользователь выбирает пункт меню «Погода» | 2. Приложение переходит в меню выбора прогноза погоды |
| 3. Пользователь выбирает пункт меню «На неделю» | 4. Приложение открывает окно просмотра сведений о погоде и выводит последние сохраненные данные |
| 5. Пользователь вводит название интересующего его населенного пункта и нажимает кнопку «✓» | 6. Приложение отправляет соответствующий запрос веб-сервису openweathermap.org, разбирает JSON-объект и обновляет окно просмотра сведений о погоде, выводя данные в виде списка, и перезаписывает в файл новые данные |
| 7. Пользователь нажимает на элемент списка | 8. Приложение открывает окно с подробной информацией о погоде для выбранного дня недели |

*Альтернатива*:

4. Приложение открывает окно просмотра сведений о погоде и выводит сообщение об отсутствии сохраненных ранее данных.

*Альтернатива 2*:

5. Пользователь не вводит название интересуемого его населенного пункта.

6. Приложение не отправляет запрос веб-сервису и не обновляет окно просмотра сведений о погоде.

*Альтернатива 3*:

5. Пользователь вводит название несуществующего населенного пункта и жмет кнопку «✓».

6. Приложение выводит сообщение об ошибке в названии населенного пункта.

*Альтернатива 4*:

5. Пользователь не вводит название населенного пункта и нажимает кнопку «✓».

6. Приложение выводит сообщение о пустой строке.

*Альтернатива 5*:

7. Пользователь не нажимает на элемент списка.

8. Приложение не открывает окно с подробной информацией о погоде для выбранного дня недели.

Таблица 5 – Описание варианта использования *Просмотр новостей по региону*

|  |  |
| --- | --- |
| Название варианта | Просмотр новостей по региону |
| Цель | Посмотреть новостную ленту по региону |
| Действующие лица | Пользователь |
| Краткое описание | Просмотр ленты новостей по выбранному региону |
| Тип | Основной |

Таблица 6 – Вариант использования *Просмотр новостей по региону*

*Типичный ход событий*

|  |  |
| --- | --- |
| Действие исполнителя | Отклик приложения |
| 1. Пользователь выбирает пункт меню «Новости» | 2. Приложение переходит в меню выбора ленты новостей |

Продолжение таблицы 6

|  |  |
| --- | --- |
| Действие исполнителя | Отклик приложения |
| 3. Пользователь выбирает пункт меню «По регионам» | 4. Приложение открывает окно просмотра ленты новостей и выводит последние сохраненные данные |
| 5. Пользователь выбирает из списка интересующий его регион | 6. Приложение отправляет соответствующий запрос веб-сервису news.yandex.ru, разбирает  XML-документ и перезаписывает в файл новые данные |
| 7. Пользователь нажимает на элемент списка | 8. Приложение открывает окно с подробной информацией выбранной новости |

*Альтернатива*:

4. Приложение открывает окно просмотра ленты новостей и выводит сообщение об отсутствии сохраненных ранее данных.

*Альтернатива 2*:

5. Пользователь не выбирает из списка название региона.

6. Приложение не отправляет запрос веб-сервису и не обновляет окно просмотра ленты новостей.

*Альтернатива 3*:

7. Пользователь не нажимает на элемент списка.

8. Приложение не открывает окно с подробной информацией выбранной новости.

Таблица 7 – Описание варианта использования *Просмотр новостей по рубрике*

|  |  |
| --- | --- |
| Название варианта | Просмотр новостей по рубрике |
| Цель | Посмотреть новостную ленту по рубрике |
| Действующие лица | Пользователь |
| Краткое описание | Просмотр ленты новостей по выбранной рубрике |
| Тип | Основной |

Таблица 8 – Вариант использования *Просмотр новостей по рубрике*

*Типичный ход событий*

|  |  |
| --- | --- |
| Действие исполнителя | Отклик системы |
| 1. Пользователь выбирает пункт меню «Новости» | 2. Приложение переходит в меню выбора ленты новостей |
| 3. Пользователь выбирает пункт меню «По рубрикам» | 4. Приложение открывает окно просмотра ленты новостей и выводит последние сохраненные данные |
| 5. Пользователь выбирает из списка интересующую его рубрику | 6. Приложение отправляет соответствующий запрос веб-сервису news.yandex.ru, разбирает XML-документ и перезаписывает в файл новые данные |
| 7. Пользователь нажимает на элемент списка | 8. Приложение открывает окно с подробной информацией выбранной новости |

*Альтернатива*:

4. Приложение открывает окно просмотра ленты новостей и выводит сообщение об отсутствии сохраненных ранее данных.

*Альтернатива 2*:

5. Пользователь не выбирает из списка название рубрики.

6. Приложение не отправляет запрос веб-сервису и не обновляет окно просмотра ленты новостей.

*Альтернатива 3*:

7. Пользователь не нажимает на элемент списка.

8. Приложение не открывает окно с подробной информацией выбранной новости.

После анализа вышерассмотренных вариантов, для наглядного представления ожидаемого поведения приложения была построена диаграмма вариантов использования, приведенная на рисунке 1.

E:\Downloads\Варианты использования (4).png

Рисунок 1 – Диаграмма вариантов использования

# **1.5 Объектная декомпозиция программного продукта**

В основе объектного подхода к разработке программного обеспечения лежит объектная декомпозиция, т. е. представление разрабатываемого программного продукта в виде совокупности объектов, в процессе взаимодействия которых через передачу сообщений происходит выполнение требуемых функций (см. рисунок 2).

E:\Downloads\Варианты использования (1).png

Рисунок 2 – Объектная декомпозиция программного продукта

В соответствии с выбранной задачей предметную область можно разбить на 5 объектов. Объект «Пользовательский интерфейс» представляет собой интерфейс взаимодействия пользователя и приложения, посредством которого пользователь передает команды системы, а система возвращает необходимые данные.

Объект «Погода» будет получать данные о прогнозе погоды от объекта «Веб-сервис», а также от объекта «Файл», передавая эти данные объекту «Пользовательский интерфейс», где они будут отображены в наглядной форме. Для объекта «Новости» аналогично.

Объект «Файл» хранит последние данные о новостях и погоде, а также передает их объектам «Новость» и «Погода».

# **1.6 Построение концептуальной модели предметной области**

После анализа предметной области и на основе объектной декомпозиции были выявлены следующие понятия: новость, погода, веб-сервис. Необходимо хранить объекты «Новость» или «Погода»; для этого решено использовать соответствующие файлы (для прогноза погоды на день, на неделю – 2 файла; для ленты новостей по региону и по рубрикам – 2 файла; всего необходимо реализовать 4 файла)

На основе этих понятий построена следующая концептуальная модель предметной области Android-приложения «Weather&News» (рисунок 3).

E:\Downloads\Варианты использования (2).png

Рисунок 3 – Концептуальная модель предметной области

Веб-сервис предоставляет данные либо об одном объекте «Погода» (прогноз на текущий день), либо о 7 (прогноз на неделю); количество объектов «Новость» определяется только полученными данными от соответствующего веб-сервиса.

# **1.7 Исследование версий API платформы Android**

Каждая версия платформы Android характеризуется уровнем API – целочисленным значением, однозначно идентифицирующим ревизию фреймворка, предоставляемую версией платформы Android. В каждой версии в платформу вносятся значительные изменения, касающиеся как исправления обнаруженных ошибок, так и введения новых функций и возможностей. Таким образом, с каждой версией платформа Android усложняется и становится более требовательной к аппаратным ресурсам устройства, вместе с тем получая всё более широкий функционал [4].

При разработке мобильного приложения на платформе Android программист выбирает минимальную и целевую API. Минимальной называется API с наименьшим номером, способная поддерживать работу приложения. Для обеспечения обратной совместимости желательно, чтобы номер минимальной API был как можно меньше. Целевая API показывает, для какого уровня API разработано и протестировано приложение. От выбранного уровня API зависит его успех на рынке, а также быстрота устаревания.

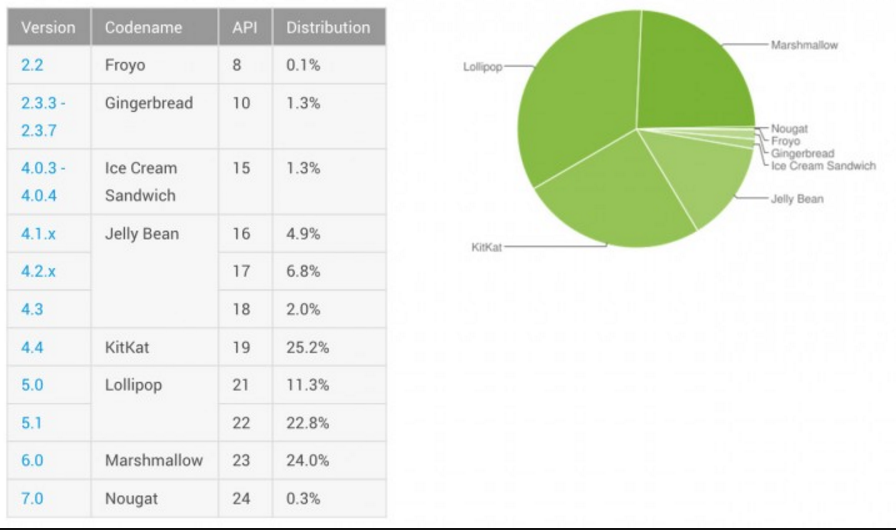
Минимальной версией Android, поддерживающей данное приложение, была выбрана Lollipop (API 21), после которой появились Marshmallow и Nougat. Marshmallow и Lollipop являются самыми популярными версиями Android (рисунок 4) [1]. Marshmallow (API 23) стремительно увеличивает свое процентное соотношение, поэтому рекомендуемой версией Android была выбрана именно она.

Рисунок 4 – Диаграмма распределения версий Android по популярности

С момента появления Android версии 5.0 Lollipop Google рекомендует свой вид дизайна программного обеспечения – Material Design [5]. Идея дизайна заключается в разработке приложений, которые открываются и сворачиваются как карточки, используя эффекты теней. У приложений не должно быть острых углов, карточки должны переключаться между собой плавно и практически незаметно. Цветов не должно быть много, и они должны сочетаться между собой.

# **1.8 Выводы**

На данном этапе были проанализированы существующие аналоги данного приложения и выделены их основные недостатки, выбраны веб-сервисы для загрузки новостей и погоды, продуманы основные функции приложения.

При определении технологии проектирования был выбран объектный подход, который является наиболее эффективным и значительно облегчает дальнейшую разработку системы.

В качестве модели жизненного цикла была выбрана спиральная модель разработки, позволяющая создавать приложение поэтапно, и на каждом этапе будет готовый прототип, выполняющий определенные функции.

Разработана диаграмма вариантов использования, описывающая возможные варианты использования системы пользователем.

Представлена объектная декомпозиция программного продукта, разбивающая приложение на отдельные объекты, обменивающиеся друг с другом посредством сообщений; были выделены главные объекты и определены связи между ними.

На основе объектной декомпозиции была разработана концептуальная модель предметной области, поясняющая связи между сущностями и их кратность.

В качестве среды разработки была выбрана Android Studio 2.1.2, являющаяся лучшей средой разработки для мобильной операционной системы Android, поскольку содержит быстрый эмулятор смартфона и «умный» редактор кода. Язык программирования Java является стандартным для разработки приложений для Android в этой среде.

Были проанализированы версии API платформы Android, выбраны минимальная (уровень 21) и целевая API (уровень 23).

В качестве метода проектирования программного продукта был выбран нисходящий подход.

# **2 Проектирование структуры программного продукта**

# **2.1 Разработка интерфейса пользователя**

Исходя из особенностей нисходящего метода разработки, было решено начать создание программного продукта с разработки пользовательского интерфейса, а затем наполнять формы интерфейса соответствующими компонентами для взаимодействия с объектами предметной области. Так как разрабатываемое приложение, предположительно, будет использоваться широким кругом пользователей, необходимо уделить внимание тому, чтобы разрабатываемый интерфейс был понятен и удобен в освоении.

# **2.1.1 Разработка структурной схемы приложения**

По результатам анализа предметной области и вариантов использования стало понятно, какая структура может быть у разрабатываемого программного продукта. Структурная схема, изображенная на рисунке 5, отражает взаимодействие частей программы по управлению.

E:\Downloads\СТРУКТУРНАЯ СХЕМА ПП (1).png

Рисунок 5 – Структурная схема приложения

Переходы между активностями осуществляются посредством нажатия кнопок.

Из главного меню можно совершить два действия: перейти к выбору критерия для прогноза погоды или же перейти к выбору критерия для ленты новостей.

При выборе первого осуществляется переход в подменю, где можно выбрать, какой тип прогноза интересует пользователя: на сегодня или на неделю. При нажатии соответствующей кнопки открывается окно просмотра прогноза погоды, где можно ввести название интересующего населенного пункта; кроме того, из окна просмотра погоды на неделю, при нажатии пользователем на конкретный интересующий день, являющийся элементом прокручиваемого списка, открывается окно с подробными данными о погоде на этот день.

При выборе второго осуществляется переход в подменю, где можно выбрать, по какому критерию составлять ленту новостей: по региону или по рубрике. При нажатии соответствующей кнопки открывается окно просмотра ленты новостей по выбранному критерию, где предлагается выбрать из выпадающего списка параметр для данного критерия; кроме того, при нажатии на конкретную новость, представляющую собой элемент прокручиваемого списка, открывается окно с подробными сведениями о выбранной новости.

# **2.1.2 Разработка структурной схемы меню**

На основании структурной схемы для уточнения спецификаций интерфейса была создана структурная схема меню.

E:\Downloads\СТРУКТУРНАЯ СХЕМА МЕНЮ (2).pngПри анализе диаграммы вариантов использования было принято решение для удобства работы пользователя главное меню сделать навигационным и объединить основные операции приложения в подменю (см. рисунок 6):

Рисунок 6 – Структурная схема меню

# **2.1.3 Разработка диаграммы состояний интерфейса**

Интерфейс программы представляет собой совокупность средств, как программных, так и аппаратных, предназначенных для обеспечения взаимодействия пользователя с приложением. На основе анализа основных функций приложения и диаграммы вариантов использования была построена диаграмма состояний пользовательского интерфейса, представленная на рисунке 7.

E:\Downloads\Граф состояний интерфейса (4) (2) (2) (2).png

Рисунок 7 – Диаграмма состояний пользовательского интерфейса

Числами 1 – 14 на диаграмме отмечены события:

1. запуск приложения;
2. нажатие кнопки «Погода»;
3. нажатие кнопки «На сегодня» (автоматически загружаются данные соответствующего файла);
4. нажатие на элемент ввода данных;
5. нажатие кнопки «✓» (данные на экране и в соответствующем файле обновляются);
6. нажатие кнопки «Назад»;
7. нажатие кнопки «На неделю» (автоматически загружаются данные соответствующего файла);
8. нажатие на элемент списка;
9. нажатие кнопки «Новости»;
10. нажатие кнопки «По регионам» (автоматически загружаются данные соответствующего файла);
11. нажатие на выпадающий список;
12. нажатие на элемент списка (данные на экране и в соответствующем файле обновляются);
13. нажатие кнопки «По рубрикам» (автоматически загружаются данные соответствующего файла);
14. закрытие приложения.

# **2.1.4 Разработка форм графического пользовательского интерфейса**

Формой графического интерфейса пользователя в Android является Activity (активность, активити). Для данного приложения было создано 8 активностей. На многих из них можно увидеть кнопки, на некоторых есть интерактивные элементы – они созданы для определения события, действия, которое совершает пользователь. Для расположения элементов, таких как TextView – текст, Button – кнопка, ListView – текстовый список и других, определения их свойств используют язык разметки XML.

На рисунках 8 – 15 представлены формы графического интерфейса приложения.



Рисунок 8 – Главное меню приложения

На главном экране приложения расположены 2 кнопки: «Погода» и «Новости». Также в верхней части экрана для большинства активностей находится элемент Toolbar, выполняющий вспомогательные функции - в разрабатываемом приложении подсказывает пользователю название приложения (в главном меню), а на других активностях подсказывает информацию об активности (рисунки 9 -15).

Кнопка «Погода» переводит пользователя в активность выбора критерия для прогноза погоды (рисунок 9), кнопка «Новости» - в активность выбора критерия для ленты новостей (рисунок 10).

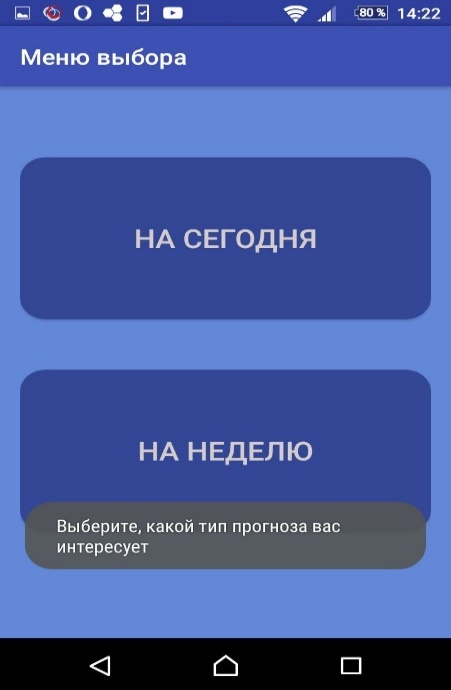


Рисунок 9 – Меню выбора критерия для прогноза погоды

Данная активность содержит 2 кнопки: для перехода на активность просмотра погоды на текущий день (рисунок 11) и на неделю (рисунок 12). Также реализована всплывающая подсказка в виде всплывающего уведомления (Toast Notification).

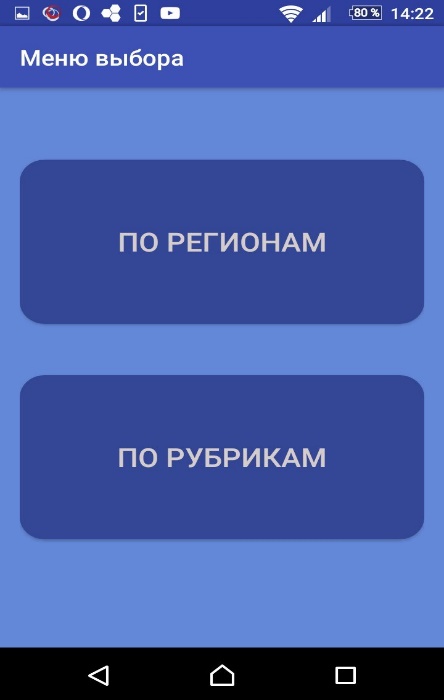


Рисунок 10 – Меню выбора критерия для ленты новостей

Описание данной активности аналогично предыдущей активности; нажатие соответствующих кнопок открывает активности просмотра ленты новостей по региону (рисунок 14) и по рубрике (рисунок 15).

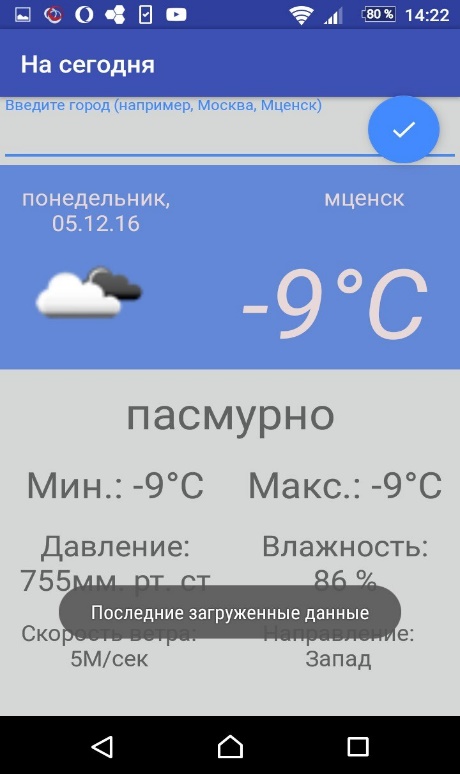


Рисунок 11 – Активность просмотра прогноза погоды на текущий день

Перейдя на данную активность, пользователь может посмотреть информацию о погоде на текущий день:

* введя в строку (Edit Text) для составления поискового запроса название населенного пункта и нажав на круглую кнопку (Floating Action Button) для подтверждения введенных данных; появится либо всплывающее уведомление об ошибке Интернет-соединения или о вводе пустой строки, либо же появится окошко (Progress Dialog), уведомляющее о отправке запроса веб-сервису и скачивании данных, после чего в текстовые поля активности (TextView) и в поле для иконки (ImageView) подставятся скачанные данные и занесутся в соответствующий файл;
* ознакомившись с данными, загруженными из файла – (если таковые имеются) – при запуске активности произойдет загрузка данных и вывод всплывающего сообщения.

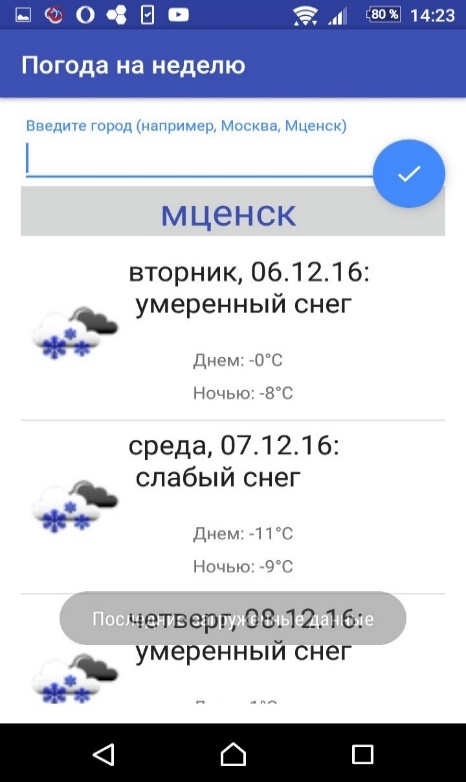


Рисунок 12 – Активность просмотра прогноза погоды на неделю

Перейдя на данную активность, пользователь может ознакомиться с прогнозом погоды на неделю, выполнив аналогичные действия, что и для активности просмотра прогноза погоды на текущий день (рисунок 11). Отличие заключается в том, что для данной активности информация о прогнозе погоды на неделю представляется в виде прокручиваемого списка (ListView), где каждый элемент – отдельный день недели.

Для получения подробного прогноза погоды на интересуемый день недели можно нажать на соответствующий элемент списка, после чего произойдет открытие активности просмотра подробного прогноза для выбранного дня (рисунок 13):



Рисунок 13 – Активность просмотра подробного прогноза погоды для выбранного дня



Рисунок 14 – Активность просмотра ленты новостей по региону

Для выбора интересующего региона был реализован выпадающий список (компонент Spinner). Аналогично активности просмотра прогноза погоды на неделю, каждая новость представляется в виде элемента списка, при нажатии на который происходит активности (рисунок 16), представляющей собой окно для просмотра веб-страниц (WebView). Также реализована запись последних загруженных данных в файл, выгрузка данных из файла (если они имеются) при загрузке активности. Всплывающие уведомления выводят подсказки для пользователя.



Рисунок 15 – Активность просмотра ленты новостей по рубрике

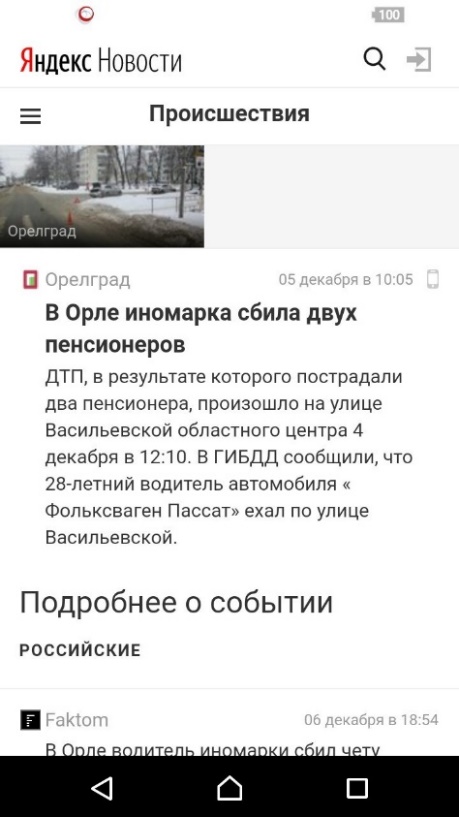


Рисунок 16 – Активность просмотра подробной информации о новости по региону



Рисунок 17 – Активность просмотра подробной информации о новости по рубрике

# **2.2 Разработка диаграмм классов приложения**

Была разработана диаграмма классов интерфейса приложения (рисунок 18). Она показывает схему взаимоотношений классов реализации интерфейса. Концепция MVC предполагает, что контроллеры являются классами.

E:\Downloads\диаграмма классов пользовательского интерфейса (1) (5).png

Рисунок 18– Диаграмма классов интерфейса приложения

В таблице 9 указаны разработанные классы-контроллеры, которые обеспечивают связь между пользователем и интерфейсом приложения.

Таблица 9 – Соответствие названий классов и их назначения

|  |  |
| --- | --- |
| Название класса | Назначение |
| MainActivity | Главное меню |
| WeatherActivity | Выбор критерия для прогноза погоды |
| NewsActivity | Выбор критерия для ленты новостей |
| DayWeatherActivity | Просмотр прогноза погоды на текущий день |
| WeekWeatherActivity | Просмотр прогноза погоды на неделю |
| DetailsWeatherActivity | Показ подробных сведений о погоды для конкретного дня |
| NewsDetailsActivity | Просмотр ленты новостей |
| NewsFullDetailsActivity | Подробные сведения о конкретной новости |

Объекты контроллеров связывают объекты представлений и контроллеров между собой. С помощью классов-моделей были реализованы сущности «Новость» и «Погода». Именно объекты данных классов несут информацию о новости или прогнозе погоды.

Для пояснения классов модели была составлена диаграмма классов предметной области уровня реализации.

E:\Downloads\Диаграмма классов предметной области уровня реализации (2).pngРисунок 19– Диаграмма классов предметной области уровня реализации

Классы Weather и News реализуют сущности «Погода» и «Новость» соответственно.

Оба этих класса реализуют интерфейс Serializable. Это необходимо для сериализации объектов классов-моделей. Сериализация  - это процесс сохранения состояния объекта в последовательность байт [6]. Серилизация – это универсальный способ хранения данных в Java. Именно последовательность байт записывается в соответствующие файлы. Затем, когда данные из файла извлекаются, они обратно десериализуются (процесс восстановления объектов из последовательности байт)

Класс Weather также реализует интерфейс Parcelable – стандартный интерфейс платформы Android. Реализация данного интерфейса использовалась для передачи объектов класса Weather из активности WeekWeatherActivity (рисунок 12) в DetailsWeatherActivity (рисунок 13). Объекты передаются между активностями в виде парселов (представление объектов в виде набора байт, реализуемое Parcelable)[7]. Реализация метода writeToParcel позволяет превращать объекты в парселы.

# **2.3 Разработка диаграмм последовательностей**

Для пояснения диаграммы классов и диаграммы вариантов использования, показа жизненного цикла каждого класса были разработаны диаграммы последовательностей для двух цепочек событий:

* для просмотра прогноза погоды на неделю (рисунок 20);
* для просмотра новостей по региону (рисунок 21).

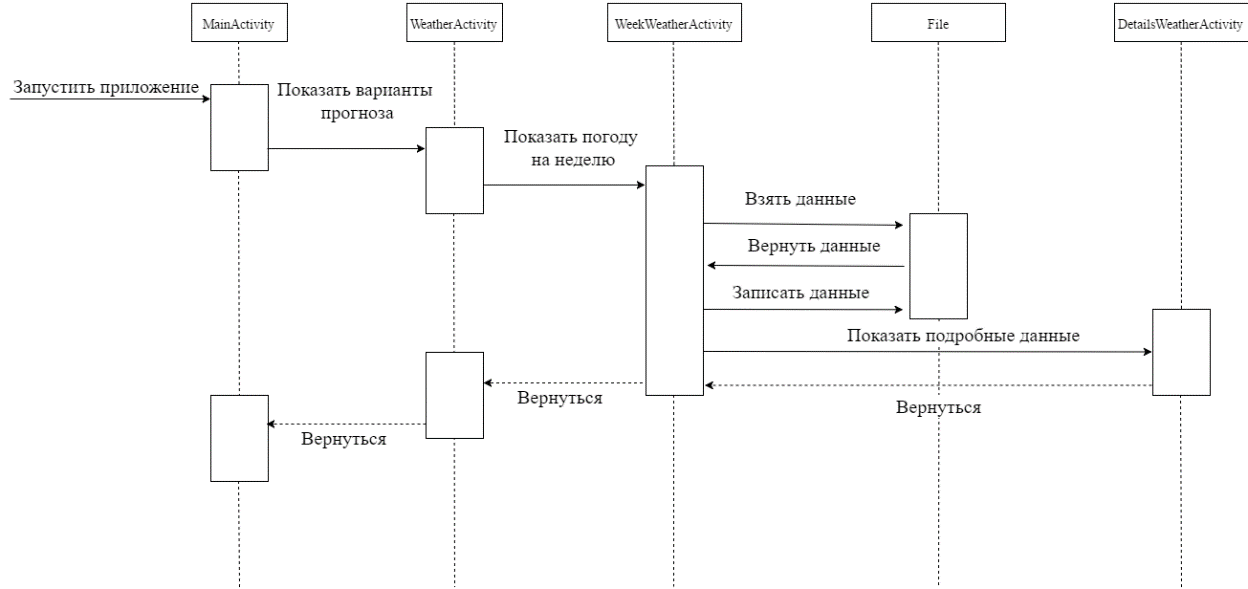


Рисунок 20 - Диаграмма последовательностей просмотра погоды на неделю

Для просмотра прогноза погоды в главном меню необходимо в главном меню нажать кнопку «Погода», после чего в подменю выбора критерия для прогноза погоды выбрать «На неделю».

Далее из файла читаются данные и выводятся на экран (если они есть). Здесь можно ввести название населенного пункта для загрузки данных о погоде. При желании, можно посмотреть подробные сведения о погоде для интересующего дня недели.

E:\Downloads\2.png

Рисунок 21 - Диаграмма последовательностей просмотра новостей по региону

Многие действия для просмотра новостей по региону (рисунок 21) совпадают с ранее описанными (см. пояснение к рисунку 20). В окне просмотра ленты новостей можно выбрать интересующий регион. При желании, можно посмотреть подробные информацию по конкретной новости.

# **2.4 Алгоритм процесса загрузки погодных данных**

E:\Downloads\Untitled Diagram (7) (1).png

Рисунок 22 – Алгоритм процесса загрузки погодных данных

После выбора населенного пункта приложение проверяет наличие подключения к сети интернет. Если такового не имеется, выводится соответствующее всплывающее сообщение. Иначе, исходя из выбранного региона, составляется URL- запрос, который затем отправляется веб-сервису. Отправление запроса и обработка ответа происходят в фоновом потоке, отдельно от потока пользовательского интерфейса. Для этого использовался отдельный класс – AsyncTask – класс, с помощью которого несколько процессов могут выполняться одновременно в фоновом режиме. Это ускоряет выполнение программы.

Если веб-сервис ответил корректно (введен существующий город), то полученный JSON-объект разбирается, из него извлекаются необходимые данные, которые затем подставляются в соответствующие поля. В ином случае, выводится сообщение об ошибке.

# **2.5 Выводы**

В данном разделе продуман графический пользовательский интерфейс. Приведена диаграмма состояний интерфейса. На ней указаны состояния и события, по которым пользователь перемещается между состояниями.

Разработана структурная схема приложения, отражающая взаимодействие частей программы по управлению.

На основании структурной схемы для уточнения спецификаций интерфейса была создана структурная схема меню.

Продемонстрированы формы графического интерфейса - активности, с помощью которых происходит связь между пользователем и приложением.

На диаграмме классов интерфейса приложения показана связь между ними, а также их поля и методы. Видно, сколько раз был использован каждый элемент активности.

Для пояснения классов модели была составлена диаграмма классов предметной области уровня реализации.

Для показа жизненного цикла классов интерфейса и событий, с помощью которых осуществляется переход между ними, приведены диаграммы последовательностей.

Был показан алгоритм загрузки погодных данных; процесс загрузки происходит в фоновом потоке вне потока пользовательского интерфейса – для этого был реализован класс AsyncTask.

# **3 Выбор стратегии тестирования и разработка тестов**

Тестирование – процесс выполнения программы с целью выявления ошибок. Никакое тестирование, насколько бы оно не было полным, не может доказать отсутствие ошибок в программных продуктах любой сложности. Поэтому крайне нерационально применять все известные методы и способы тестирование, напротив, необходимо подобрать оптимальные, исходя из спецификаций программы.

Тестирование по принципу «черного ящика» или тестирование с управлением данными позволяет выявить отклонения в работе программы и подходит как для тестирования интерфейсной части, так и для проверки правильности взаимодействия объектов в игре. В конце необходимо провести оценочное тестирование для проверки соответствия программы основным требованиям.

# **3.1** **Функциональное тестирование**

Функциональный подход основывается на том, что структура программного обеспечения не известна («черный ящик»). В этом случае тесты строят, опираясь на функциональные спецификации.

Тестирование по стратегии чёрного ящика подразумевает проверку функциональности программного продукта в условиях отсутствия информации о его внутреннем устройстве. Согласно техническим требованиям составляются наборы входных данных и определяются эталонные реакции тестируемого продукта [8].

Результаты тестов приведены в таблице 10.

Таблица 10 – Результаты тестирования по стратегии черного ящика

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Назначение теста | Событие (действие) | Ожидаемый результат | Полученный результат | Метод решения (если необходимо) |
| 1 | Проверка правильности перехода к активностям при нажатии соответствующих кнопок | Нажатие кнопки "Новости", «Погода» главного меню приложения | Переход по нажатию кнопок осуществляется верно | Переход по нажатию кнопок осуществляется верно |  |
| 2 | Проверка загрузки данных из файла при его отсутствии | Нажатие кнопки «На сегодня» активности WeatherActivity | Вывод информационного сообщения | Аварийное завершение приложения | Добавлена проверка существова- ния файла |
| 3 | Проверка загрузки данных, соответствующих выбранным данным | Ввод населенного пункта в активности WeekWeather- Activity | Выводимая информация соответствует запросу по выбранным данным | Выводимая информация соответствует запросу по выбранным данным |  |
| 4 | Проверка ввода наименования несуществующего населенного пункта «111111111» | Ввод населенного пункта в активности WeekWeather- Activity | Вывод сообщения об ошибке | Вывод всплывающего сообщения «Ошибка загрузки данных!» |  |

Продолжение таблицы 10

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Назначение теста | Событие (действие) | Ожидаемый результат | Полученный результат | Метод решения (если необходимо) |
| 5 | Проверка загрузки данных при отсутствии подключения к сети Интернет | Выбор региона в активности NewsDetails- Activity | Вывод информационного сообщения | Бесконечная попытка загрузки данных | Добавлена проверка наличия подключе-ния к сети Интернет |
| 6 | Проверка загрузки данных в файл и чтения этих данных из файла | Нажатие кнопки «На сегодня» активности WeatherActivi-ty | Загруженные данные автоматически сохраняются в файл; при повторном запуске активности выводится информация из файла | Загруженные данные автоматически сохраняются в соответствующий файл; при повторном запуске активности выводится информация из файла; появляется всплывающее сообщение «Последние загруженные данные» |  |

Результаты данного тестирования позволили быстро и эффективно избавиться от серьезных ошибок и правильно реализовать работу приложения.

# **3.2 Оценочное тестирование**

Результаты оценочного тестирования, которое также называют тестированием системы в целом, показаны на таблице 7. Его целью является тестирование программы на соответствие основным критериям качества программного продукта.

Для разрабатываемого приложения были выбраны четыре вида оценочного тестирования: тестирование удобства использования, тестирование удобства эксплуатации, тестирование требования к памяти, тестирование конфигурации оборудования.

Тестирование проводилось с привлечением группы из 10 независимых пользователей.

Участники тестирования выбирали ответы по принципу «да/нет» и на основе большинства голосов, были сделаны выводы, приведенные в таблице.

Таблица 11 – Оценочное тестирование

|  |  |
| --- | --- |
| Вид тестирования | Результаты тестирования |
| Тестирование удобства использования | Программный продукт полностью соответствует техническому заданию, выполняет все заявленные функции. |
| Тестирование удобства эксплуатации | Приложение имеет минималистичный, удобный интерфейс, понятный текст и нейтральное цветовое сопровождение. Все кнопки содержат надпись, поясняющую их действие. |
| Тестирование требования к памяти | Приложение корректно работает на устройствах с 512 Мб ОЗУ и более. |
| Тестирование конфигурации оборудования | Приложение работает корректно на смартфонах под управлением операционной системы Android версий 6.0 Marshmallow, 5.1 Lollipop и 5.0 Lollipop.  Интерфейс приложения корректно подстраивается под различные разрешения экранов смартфонов. |

# **3.3 Выводы**

На данном этапе были выбраны стратегии и методы тестирования – стратегия чёрного ящика по методу предположения об ошибке и оценочное тестирование, составлены таблицы тестов для каждого из них.

Проведенные тестирования показывают, что все тестовые наборы данных обрабатываются верно и все этапы оценочного тестирования пройдены успешно.

# **ЗАКЛЮЧЕНИЕ**

В результате выполнения курсовой работы было разработано и реализовано Android-приложение «Weather&News», удовлетворяющее всем требованиям технического задания.

Приложение имеет понятный и удобный интерфейс, что позволяет легко пользоваться им. Функции, предусмотренные в техническом задании, могут быть расширены в следующих версиях приложения. В качестве усовершенствования рассматривается возможность добавления нового раздела «Афиша», предназначенного для просмотра афиши в интересующем населенном пункте; вывода уведомлений в определенное время суток или же по определенному событию, запоминания «избранных» населенных пунктов, рубрик и регионов пользователя.

В ходе разработки программы был получен опыт проектирования программных продуктов, использования специальных средств разработки, составления конструкторской документации, а также получены навыки тестирования и отладки приложений.

# **СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

1. Доля Android на рынке. // 4PDA. 2016. URL: http://4pda.ru/2016/08/20/318123/ (дата обращения: 02.12.2016).
2. Android Studio. // Google. 2016. URL: https://developer.android.com/studio/index.html (дата обращения: 28.11.2016).
3. Иванова, Г.С. Технология программирования: Учебник для вузов [Текст]. - М.: Изд-во МГТУ им. Н.Э. Баумана, 2002. - 320 с.
4. What is API Level. // Android Developers. 2016. URL: http://developer.android.com/intl/ru/guide/topics/manifest/uses-sdk-element.html#ApiLevels (дата обращения: 03.12.2016).
5. Material design. // Google. 2016. URL: https://material.google.com/ (дата обращения: 28.11.2016).
6. Сериализация в Java. //Habrahabr. 2016. URL: <https://habrahabr.ru/post/60317/> (дата обращения: 03.12.2016).
7. Parcelable. Передаём объекты. // Android Developer. 2016. URL: <http://developer.alexanderklimov.ru/android/theory/parcelable.php> (дата обращения: 03.12.2016).
8. Тестирование по стратегии черного ящика// Wikipedia. 2016. URL:https://ru.wikipedia.org/wiki/Тестирование\_по\_стратегии\_чёрного\_ящика (дата обращения: 03.12.2016).
9. Иванова Г.С., Ничушкина Т. Н., Пугачев Е.К., Самарев Р.С. - Методические указания по выполнению курсовой работы по дисциплине «ТРПС», 2013. – 23 с.

ПРИЛОЖЕНИЕ 1

**Android - приложение «Weather&News»**

**Техническое задание**

ПРИЛОЖЕНИЕ 2

**Android - приложение «Weather&News»**

**Руководство пользователя**