



Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего 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 (ИУ6)

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.03.03 Прикладная информатика

## ОТЧЕТ О ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ

Тип практики Преддипломная практика

Название  
предприятия НУК ИУ МГТУ им. Н.Э. Баумана

Студент группы ИУ6-84Б

С.П. 28.02.2022  
(Подпись, дата)

С.П. Пантелеев  
(И.О. Фамилия)

Руководитель практики от  
МГТУ им. Н.Э. Баумана

М.В. 28.02.2022  
(Подпись, дата)

М.В. Широкова  
(И.О. Фамилия)

Оценка отлично

2022 г.

## ЗАДАНИЕ на производственную практику

по теме формирование структуры приложения по подбору одежды в соответствии с погодой

Студент группы ИУ6-84Б

Пантелеев Сергей Павлович

(Фамилия, имя, отчество)

Направление подготовки 09.03.03 Прикладная информатика

Тип практики Преддипломная практика

Название предприятия НУК ИУ МГТУ им. Н.Э. Баумана

### Техническое задание:

Разработка структуры программного продукта, выбор технологии программирования, разработка концептуальной модели предметной области/объектной декомпозиции, определение основных функций программы и построение диаграммы вариантов использования, разработка интерфейсов, разработка алгоритма подбора одежды, выбор методов хранения данных

### Оформление отчета по практике:


Отчет на 15-25 листах формата А4 должен включать титульный лист, оглавление, введение, несколько глав, заключение и список использованных источников.

Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)

нет


Дата выдачи задания « 07 » февраля 2022 г.

Руководитель практики

 7.02.2022  
(Подпись, дата)

М.В. Широкова  
(И.О. Фамилия)

Студент

 7.02.2022  
(Подпись, дата)

С.П. Пантелеев  
(И.О. Фамилия)

Примечание: Задание оформляется в двух экземплярах:

## Содержание

ВВЕДЕНИЕ.....	5
1 Разработка структурной схемы программного продукта.....	6
2 Выбор технологии программирования.....	7
3 Разработка концептуальной модели предметной области.....	8
4 Разработка диаграммы вариантов использования.....	10
5 Разработка интерфейса пользователя.....	13
6 Разработка форм интерфейса.....	16
7 Разработка графов диалога.....	22
8 Разработка алгоритма подбора одежды.....	24
9 Выбор метода хранения данных.....	27
10 Разработка диаграммы уровня реализации.....	29
11 Разработка диаграммы компоновки.....	33
ЗАКЛЮЧЕНИЕ.....	35
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ.....	36

## **ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ**

ПП – программный продукт

ООП – объектно-ориентированное программирование

Баг – недоработка в программном коде

БД – база данных

ТЗ – техническое задание

## **Введение**

Целью преддипломной практики является описание структуры разрабатываемого продукта и реализация этапов проектирования приложения ВКРБ. В процессе работы будет выбрана технология программирования, разработан алгоритм подбора одежды, выбран метод хранения данных, разработаны интерфейсы и их возможные состояния, и диаграмма вариантов использования.

## 1 Разработка структурной схемы программного продукта

Структурная схема программного продукта показывает разделение программы на её главные составляющие. На основе анализа технического задания, в разрабатываемом приложении, которое подбирает одежду по погоде, выявлено 3 подсистемы:

- подсистема подбора одежды: в подсистеме алгоритма подбора одежды происходит подбор одежды для пользователя в соответствии с выбранными параметрами (настроек, текущей температуры и имеющейся одежды);
- подсистема гардероба включает в себя две подсистемы:
  - в подсистеме подкатегорий одежды происходит выбор имеющейся одежды;
  - в подсистеме добавления новой одежды происходит добавление новой одежды;
- подсистема пользовательских настроек, в которой выбираются теплоощущение, температурная шкала и пол пользователя.

На основе выявленных подсистем была составлена структурная схема ПП, показанная на рисунке 1.

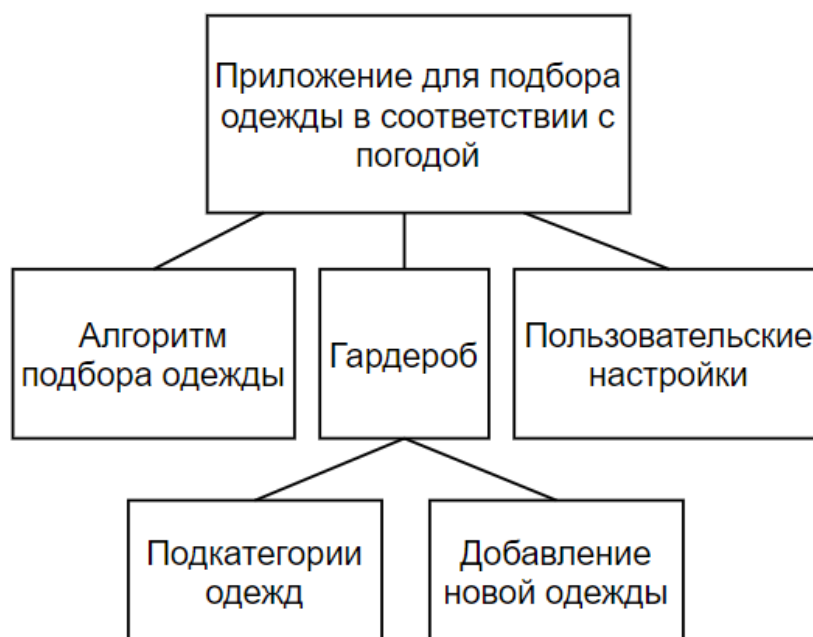


Рисунок 1 – структурная схема программного продукта

## **2 Выбор технологии программирования**

В качестве технологии программирования было выбрано объектно-ориентированное программирование. ООП является основой всех современных приложений и имеет удобное и практическое применение. При использовании этого метода вся программа разбивается на объекты, с каждым из которых работают по отдельности.

В качестве жизненного цикла разработки была выбрана спиральная модель. Она представляет из себя 4 этапа, которые зацикливаются до разработки финальной версии продукта, это [1]:

1. планирование;
2. анализ требований;
3. конструирование;
4. оценка результата.

Этот метод позволяет в конце каждого цикла иметь работающий продукт, который можно продемонстрировать. Это дает возможность своевременно оценить и протестировать продукт, чтобы сразу вносить какие-либо правки и исправлять ошибки, не дожидаясь окончания разработки. Нахождение багов на каждом этапе позволяет избежать “волнового” исправление ошибок, когда исправление одной ошибки в одном месте, выявляет появление другой в другом месте. А также этот метод позволяет детальнее подойти к каждому этапу разработки по отдельности.

Также при разработке было решено использовать нисходящий подход, реализуя сначала модули верхнего уровня (интерфейс пользователя), а после переходя к модулям нижнего уровня (логика работы программы).

### 3 Разработка концептуальной модели предметной области

При анализе предметной области, технического задания и структуры разрабатываемого ПО были выявлены следующие основные объекты:

- главный экран с подобранной одеждой;
- подобранная одежда;
- данные;
- текущая температура;
- текущее местоположение;
- настройки пользователя:
  - температурная шкала;
  - пол пользователя;
  - теплоощущение пользователя;
- гардероб;
- категории одежды:
  - шаблонная одежда;
  - пользовательская одежда;
  - данные одежды: название; минимальная и максимальная одежда; тип (категория); для кого (пол); картинка.

Меню содержит несколько вкладок – задач – главный экран, окно настроек и гардероба. Главный экран выводит подобранную одежду, которая подбирается, основываясь на данных – текущей температуре, которая берется в зависимости от текущего местоположения; пользовательских настроек – какое у пользователя теплоощущение и пол; а также от самих температурных данных, которые хранятся в параметрах как шаблонной, так и пользовательской одежды. Сама одежда хранится в гардеробе, который в свою очередь для удобства разделяет одежду на несколько категорий. На рисунке 2 показано визуальное представление концептуальной модели предметной области.



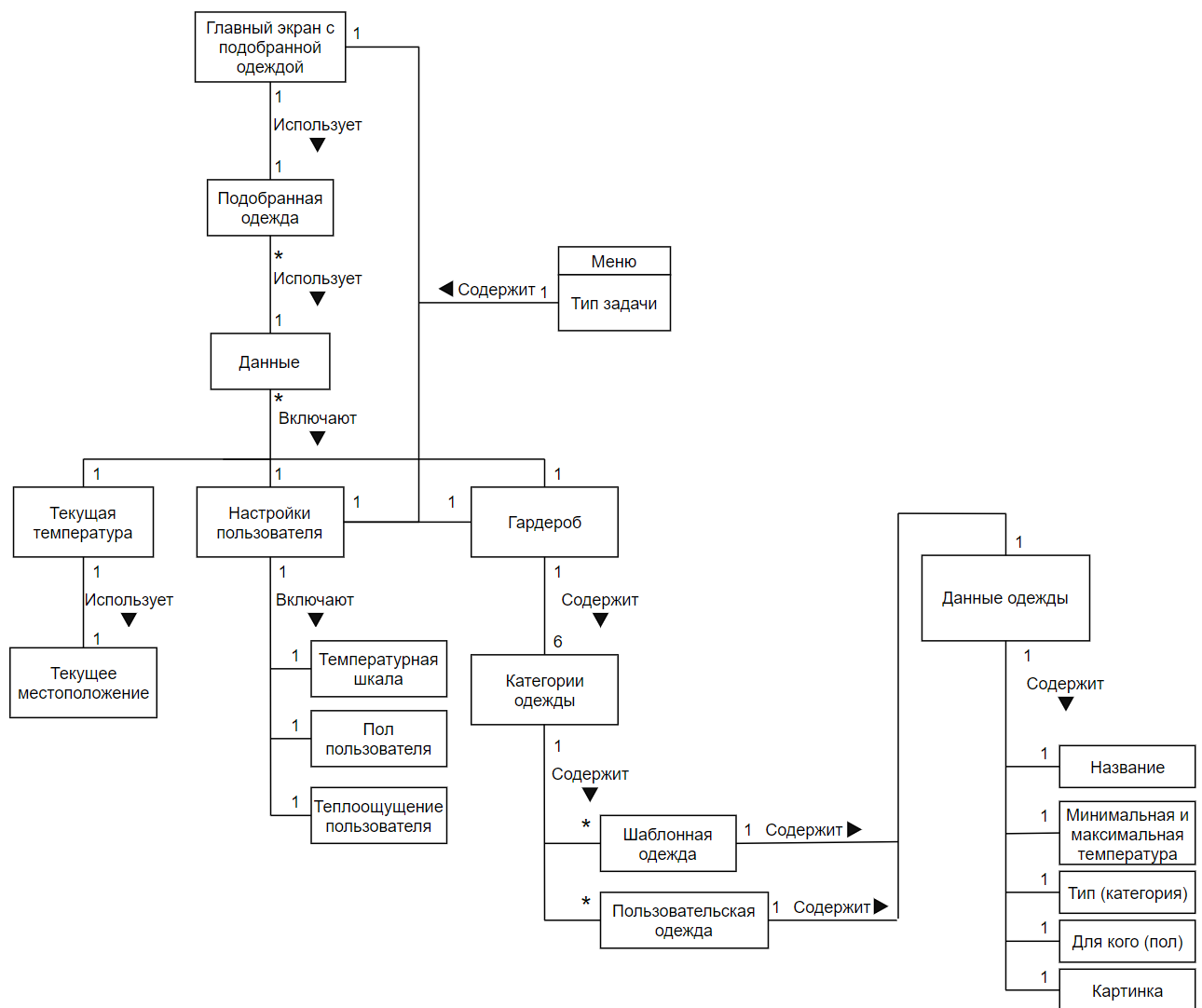


Рисунок 2 – Концептуальная модель предметной области

Основными объектами можно выделить:

- подобранная одежда. Это картинки и названия, которые приложение выводит пользователю как вариант что можно сегодня одеть;
- настройки пользователя. В них пользователь настраивает приложение под себя, выбирая такие параметры как, теплоощущения и пол, которые влияют на подбор одежды;
- гардероб. В нем хранится вся одежда, то есть как шаблонная, так и добавленная пользователем;
- данные одежды. Этот объект включает в себя параметры одежды, с помощью которых одежда сортируется по категориям и полу, а также основываясь на них, выбирается та одежда, что будет учитываться в подборе.

## 4 Разработка диаграммы вариантов использования

При проектировании приложения необходимо знать основные варианты взаимодействия пользователя с системой, для визуального представления была разработана диаграмма вариантов использования.

В результате анализа ТЗ были выявлены следующие варианты взаимодействия пользователя и приложения:

- просмотр предлагаемой одежды;
  - обновление предлагаемой одежды;
- изменение гардероба;
  - просмотр уже имеющейся одежды;
    - выбор имеющейся одежды;
    - удаление имеющейся одежды;
    - редактирование параметров имеющейся одежды;
  - добавление новой одежды;
- редактирования настроек.

Рассмотрим варианты использования, связанные с гардеробом пользователя, показанные в таблицах 1 – 3. Так как этот вариант использования имеет наибольшее количество возможных взаимодействий, следовательно должна быть наиболее подробно проработана.

Таблица 1 – Вариант использования **“Изменить гардероб”**

Действия пользователя	Отклик системы
1. Пользователь в панели навигации нажимает кнопку “Гардероб”. 3. Пользователь выбирает: а) нажимает на одну из категорий одежды. б) нажимает на кнопку “Расширить гардероб”.	2. Приложение открывает окно “Гардероб”. 4. Приложение открывает окно: а) с одеждой выбранной категории, см. раздел Просмотр уже имеющейся одежды (таблица 2). б) с формой создания новой одежды, см. раздел Добавить новую одежду (таблица 3).

Таблица 2 – раздел ***“Просмотр уже имеющейся одежды”***

Действия пользователя	Отклик системы
<p>1. Пользователь может:</p> <p>а) нажать на шаблонную одежду, отмеченную, как которой нет;</p> <p>б) нажать на шаблонную одежду, отмеченную, как которая есть;</p> <p>в) удалить одежду, ранее добавленную пользователем;</p> <p>г) редактировать одежду, ранее добавленную пользователем;</p> <p>д) нажать кнопку “Добавить”</p>	<p>2. Приложение:</p> <p>а) на картинке шаблонной одежды появляется галочка и одежда отмечается в БД как имеющееся и учитывается при подборе;</p> <p>б) на картинке шаблонной одежды снимается галочка и одежда отмечается в БД как отсутствующая и при не учитывается при подборе;</p> <p>в) картинка пропадает с экрана и удаляется из БД;</p> <p>г) открывается окно редактирования параметров (температуры, категории, пола и картинки);</p> <p>д) открывается окно добавление новой одежды, см. раздел Добавить новую одежду (таблица 3).</p>

Таблица 3 – раздел ***“Добавить новую одежду”***

Действия пользователя	Отклик системы
<p>1. Пользователь вводит название одежды, максимальную и минимальную рекомендованную температуру.</p> <p>2. Пользователь нажимает на кнопку “тип одежды”.</p> <p>4. Пользователь выбирает нужный тип одежды.</p> <p>5. Пользователь нажимает на кнопку “для кого”.</p>	<p>3. Выпадает список выбора типов одежды.</p> <p>6. Выпадает список выбора для какого пола предназначена одежда.</p> <p>9. Открывает окно выбора файлового приложения, откуда будет взята картинка.</p> <p>11. Открывается выбранное приложение.</p> <p>13. Файловое приложение закрывается.</p>

Продолжение таблицы 3

Действия пользователя	Отклик системы
<p>7. Пользователь выбирает для какого пола предназначена одежда.</p> <p>8. Пользователь нажимает на изображение картинки.</p> <p>10. Пользователь нажимает на одно из файловых приложений.</p> <p>12. Пользователь выбирает одну из картинок и нажимает на неё.</p> <p>14. Пользователь нажимает кнопку “сохранить”.</p>	<p>15. Открывается окно:</p> <p>а) выбора категории одежды, если ранее была нажата кнопку “расширить гардероб”;</p> <p>б) той категории одежды, в которой была нажата кнопка “Добавить”</p>

Разработанная диаграмма вариантов использования представлена на рисунке 3 [2].

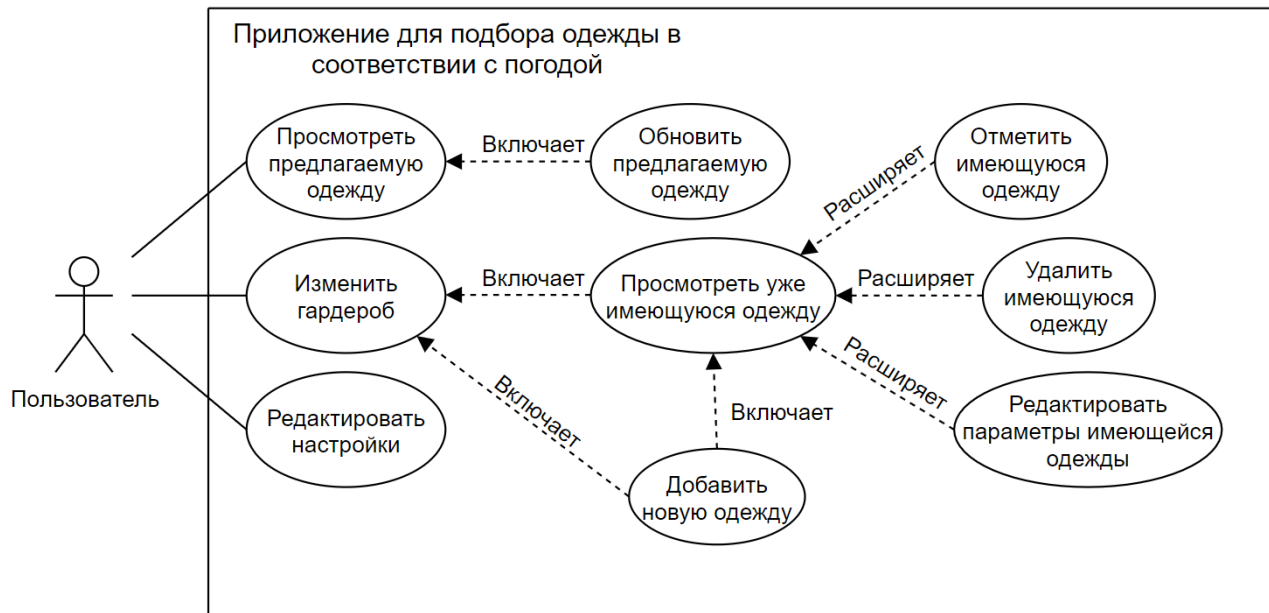


Рисунок 3 – Диаграмма вариантов использования

## 5 Разработка интерфейса пользователя

В следствии анализа предметной области был выбран вид интерфейса со свободной навигацией, то есть пользователь может свободно перемещаться по окнам приложения и смотреть, что программный продукт может предложить. Так устроены большинство современных приложений, и пользователи привыкли, что могут перемещаться по разным окнам интерфейса.

Чтобы не загромождать экран кнопками перехода между окнами, переключение между основными окнами было сделано с помощью выезжающей боковой панели. Для удобства, свободы и быстроты перемещения между окнами, боковую панель можно открыть в любом окне, сделав свайп с левой границы экрана в право. Разработанный граф состояний интерфейса, для наглядного отображения переходов, представлен на рисунке 4 [3].

C1 – активация формы при запуске приложения;

C2 – открытие навигационной панели при нажатии на кнопку “Добавить” или при свайпе с левой границы экрана в право;

C3 – по нажатию кнопки “Что надеть?” возвращение на главный экран;

C4 – по нажатию кнопки “Настройки” переход в окно пользовательских настроек;

C5 – открытие навигационной панели при нажатии на “Добавить”;

C6 – по нажатию кнопки “Гардероб” переход в окно категорий гардероба;

C7 – открытие навигационной панели при нажатии на “Добавить”;

C8 – по нажатию кнопки “ТОЛОВНЫЕ УБОРЫ” переход в категорию соответствующую категории одежды;

C9 – по нажатию кнопки “ВЕРХНЯЯ ОДЕЖДА” переход в соответствующую категорию одежды;

C10 – по нажатию кнопки “КОФТЫ, СВИТЕРА И ПРОЧЕЕ” переход в соответствующую категорию одежды;

C11 – по нажатию кнопки “МАЙКИ, ФУТБОЛКИ И ПРОЧЕЕ” переход в соответствующую категорию одежды;

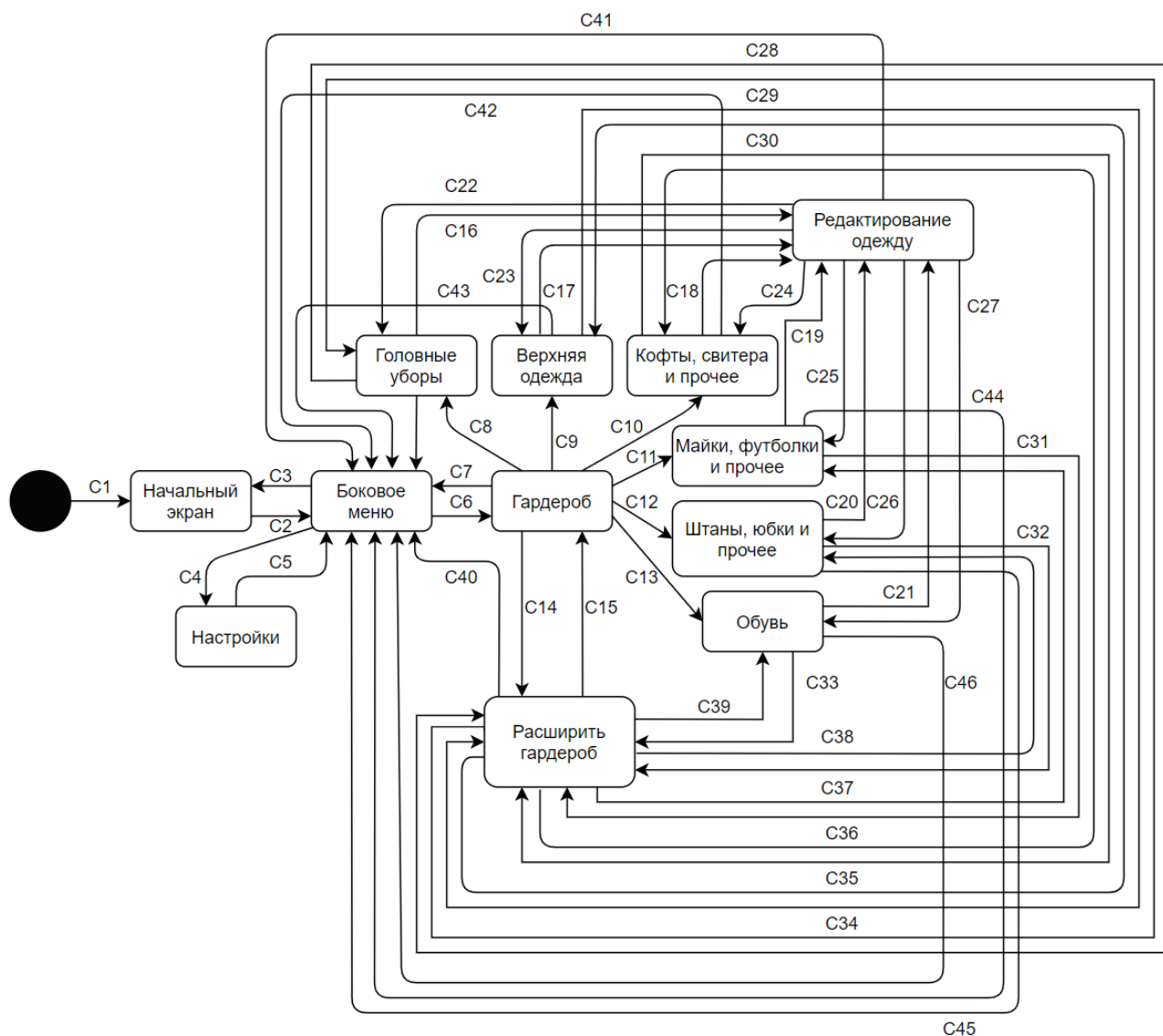


Рисунок 4 – граф состояний интерфейса

C12 – по нажатию кнопки “ШТАНЫ, ЮБКИ И ПРОЧЕЕ” переход в соответствующую категорию одежды;

C13 – по нажатию кнопки “ОБУВЬ” переход в соответствующую категорию одежды;

C14 – по нажатию кнопки “РАСШИРИТЬ ГАРДЕРОБ” переход в окно добавления новой одежды;

C15 – по нажатию кнопки “Сохранить” возвращение в окно категорий гардероба;

C16 – C21 – по нажатию кнопки “Редактировать” на добавленной одежде, переход в окно редактирования параметров выбранной одежды;

С22 – по нажатию кнопки “Сохранить” возвращение в окно категории одежды “Головные уборы”;

С23 – по нажатию кнопки “Сохранить” возвращение в окно категории одежды “Верхняя одежда”;

С24 – по нажатию кнопки “Сохранить” возвращение в окно категории одежды “Кофты, свитера и прочее”;

С25 – по нажатию кнопки “Сохранить” возвращение в окно категории одежды “Майки, футболки и прочее”;

С26 – по нажатию кнопки “Сохранить” возвращение в окно категории одежды “Штаны, юбки и прочее”;

С27 – по нажатию кнопки “Сохранить” возвращение в окно категории одежды “Обувь”;

С28 – С33 – по нажатию кнопки “Добавить” переход в окно добавления новой одежды;

С34 – по нажатию кнопки “Сохранить” возвращение в окно категории одежды “Головные уборы”;

С35 – по нажатию кнопки “Сохранить” возвращение в окно категории одежды “Верхняя одежда”;

С36 – по нажатию кнопки “Сохранить” возвращение в окно категории одежды “Кофты, свитера и прочее”;

С37 – по нажатию кнопки “Сохранить” возвращение в окно категории одежды “Майки, футболки и прочее”;

С38 – по нажатию кнопки “Сохранить” возвращение в окно категории одежды “Штаны, юбки и прочее”;

С39 – по нажатию кнопки “Сохранить” возвращение в окно категории одежды “Обувь”;

С40 – С46 – открытие боковой панели при свайпе с левой границы экрана в право.

## 6 Разработка форм интерфейса

Приложение имеет прямое взаимодействие с пользователем, поэтому необходимо разработать удобный и интуитивно понятный интерфейс. Разработанные формы интерфейса проекта показан на рисунках 5 – 10.

Главной задачей приложения является предоставления подобранной одежды в зависимости от погоды. Исходя из этого главным экраном приложения было решено разработать окно, которое включает в себя информацию о: местоположение, температура, список предлагаемой одежды. Окно главного экрана показана на рисунке 5.

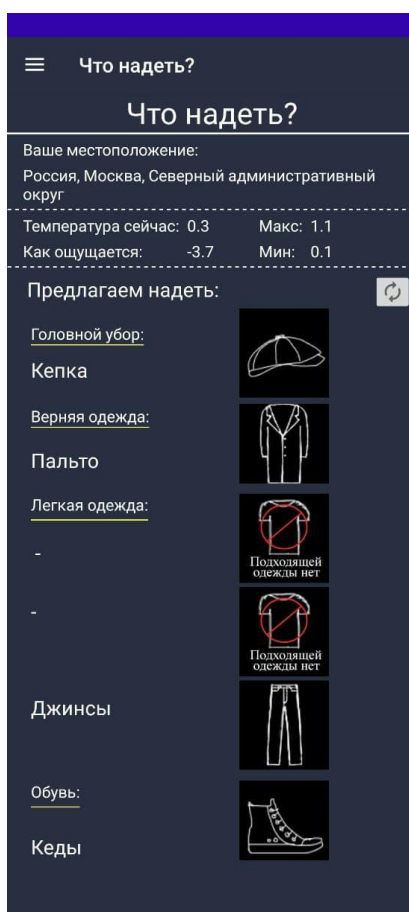


Рисунок 5 – Главного экран

Помимо предлагаемой одежды, рассортированной по категориям, текущего местоположения и температуры присутствует кнопка обновления (стрелочки), которая предназначена для переподбора одежды. Если



подходящей одежды нет, то будет выведено соответствующее изображение, как показано для “Легкой одежды”.

Для навигации по приложению было использовано боковое меню, показанное на рисунке 6. Оно удобно тем, что не загромождает экран кнопками навигации на каждой вкладке, а выносит их в отдельную вкладку, которую можно вызвать по необходимости.

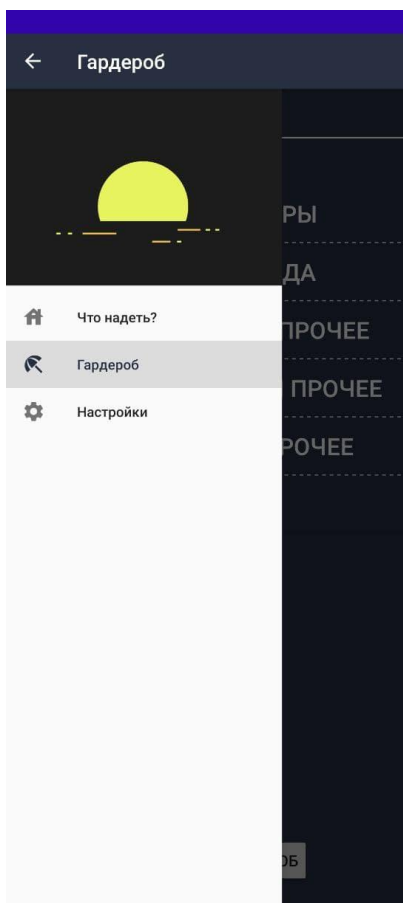


Рисунок 6 – Боковое меню

Через боковое меню происходит переключение между окнами “Что надеть?”, “Гардероб” и “Настройки”. В меню вынесены именно эти кнопки, так как эти окна являются основными экранами приложения. Открывается боковое меню с помощью кнопки “Меню” или при свайпе с левой границы экрана вправо.

Пользователь может настроить подбор одежды под себя в окне “Настройки”, которое показано на рисунке 7.



Рисунок 7 – Настройки

В настройках пользователь может выбрать пол и температурную шкалу, которая будет отображаться на главном экране, и теплоощущение:

- всегда холодно – пользователь ощущает холод чаще остальных людей;
- нормально – терморегуляция тела происходит нормально;
- всегда жарко – пользователь ощущает тепло больше остальных.

Например, пользователю не холодно зимой без шапки.

Одежда рассортирована на категории, и для перехода к каждой из них, приложение имеет общее окно “Гардероб”, показанное на рисунке 8.

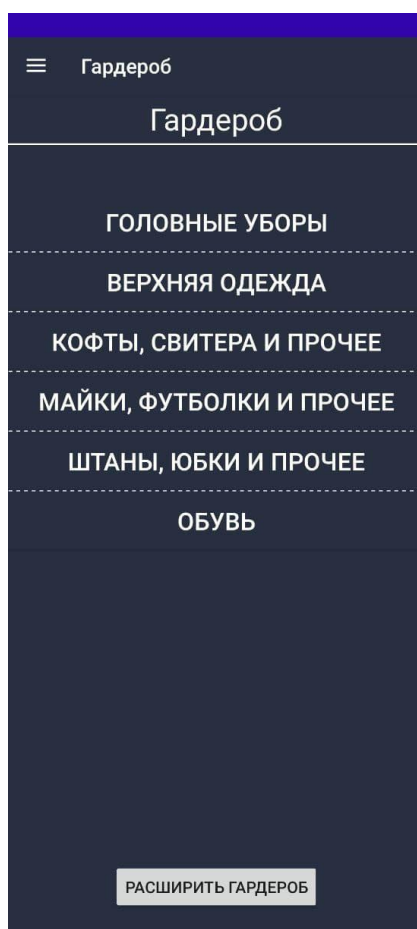


Рисунок 8 – Гардероб

В окне “Гардероб” отображены категории одежд, по которым рассортирована одежда, такими категориями являются: “Головные уборы”, “Верхняя одежда”, “Кофты, свитера и прочее”, “Майки, футболки и прочее”, “Штаны, юбки и прочее” и “Обувь”. Помимо этого, предусмотрена кнопка “Расширить гардероб” для возможности добавления новой одежды.

Для удобства добавления одежды для каждой категории разработано окно, в котором перечислены все виды одежды из выбранной категории. Одно из таких окон показано на рисунке 9.

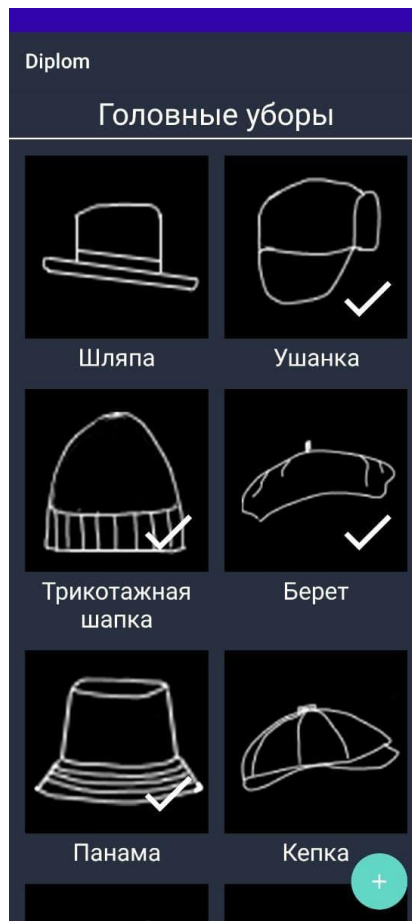


Рисунок 9 – Категория одежды


На рисунке 9 представлена шаблонная одежда категории головные уборы. Для учета той одежды, которая присутствует у пользователя предусмотрен знак в виде галочки. При подборе одежды учитываются лишь те виды одежды, которые имеют данный знак. На картинках одежд, у которых нет галочки, учитываться при подборе не будут. Также для добавления новой одежды была разработана кнопка “добавить”, находящаяся в правом нижнем углу. Так как основная функция этого окна – это показать всю одежду конкретной категории, поэтому кнопка “добавить” небольшого размера и размещена в углу, чтобы не оттягивать на себя много внимания.

Помимо добавления шаблонной одежды, для пользователя предусмотрена возможность добавлять новый вид одежды, для этого было разработано окно добавления, показанное на рисунке 10.

Diplom

## Добавление одежды

Название:



Рекомендованная температура:

Максимальная:

Минимальная:

Тип одежды:

Для кого:

Рисунок 10 – Добавление одежды

Окно добавления одежды было решено реализовать как стандартная форма заполнения данных, ведь при добавлении одежды пользователю необходимо вводить данные. Этими данными являются: название, картинка, рекомендованная минимальная и максимальная температура и тип одежды и пола (то есть для кого она предназначена). Для сохранения введенных данных предусмотрена кнопка “сохранить”.

## 7 Разработка графа диалогов

Разрабатываемое приложение имеет несколько окон, следовательно диалог приложения и пользователя не всегда будет одинаковый. Для удобного взаимодействия пользователя и приложения необходимо продумать как приложение будет реагировать на различные действия пользователя, чтобы это более подробно показать были разработаны несколько графов диалогов:

- общий граф диалога;
- граф диалога добавления новой одежды.

Общий граф диалога показан на рисунке 11, который отображает работу приложения в целом. Сначала пользователь выбирает вкладку гардероб, в ней отмечает шаблонную одежду, и/или добавляет собственную. Далее настраивает параметры настроек под себя. И на основе этих действий пользователю подбирается одежда.

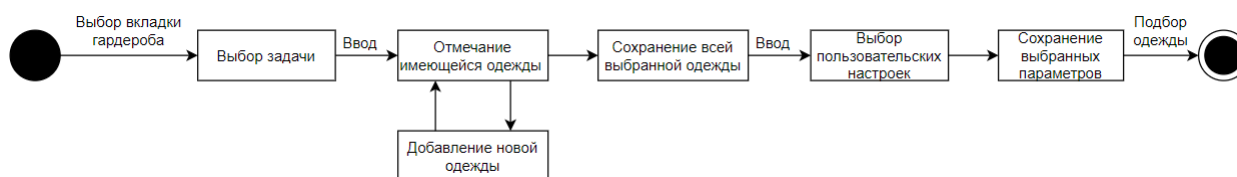


Рисунок 11 – Общий граф диалога

Граф диалога добавления новой одежды показан на рисунке 12, который показывает работу взаимодействия между приложением и пользователем при добавлении новой одежды. Сначала пользователь должен оказаться на одном из окон — гардероб или одна из категорий одежды. После нажать на кнопку добавления новой одежды. Далее ввести все параметры новой одежды и выбрать её картинку, в ином случае высветятся уведомления о невыполнении того или иного действия. В конце при нажатии кнопку «Сохранить» новая одежда добавляется в БД.

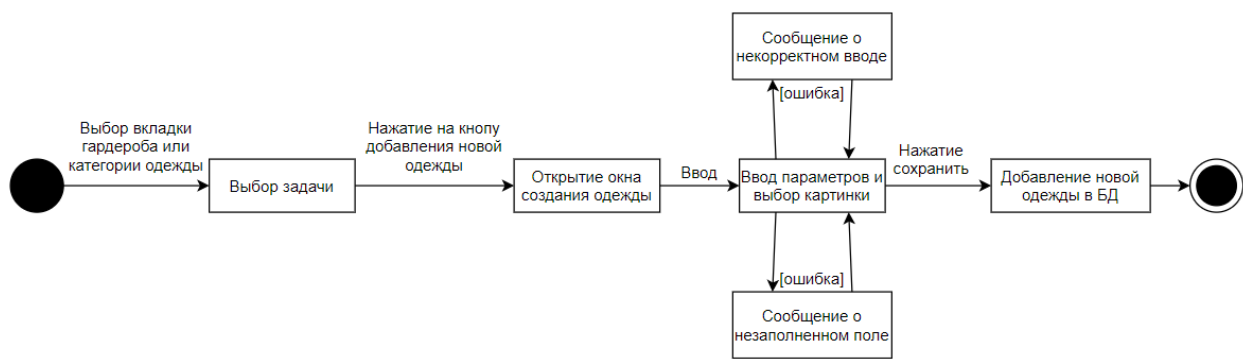


Рисунок 12 – Граф добавления новой одежды

## 8 Разработка алгоритма подбора одежды

Одной из основных подсистем приложения является алгоритм подбора одежды. Алгоритм необходимо разработать так, чтобы точность его подбора удовлетворяла требованиям, введённым пользователем, следовательно он должен содержать в себе большое количество проверок:

- проверка, что категория одежды соответствует подбираемой категории;
- проверка наличие выбранной одежды у пользователя;
- проверка соответствия одежды полу пользователя;
- проверка, что температурные параметры одежды подходят под реальную температуру.

Обработка шаблонной одежды, и одежды, добавленной пользователем, изначально происходит отдельно. Это сделано для избежание лишних ветвлений при проверках, что может привести к незаметным ошибкам.

Первым этапом идет работа с шаблонной одеждой. Её обработка происходит в несколько шагов:

1. Создается массив шаблонной одежды нужной категории.
2. Поочередно берется каждая вещь из только что созданного массива.
3. Проверка наличия одежды у пользователя в БД.
  - a. Если вещи нет, то возвращение к пункту 2.
  - b. Если вещь есть, то переход к пункту 4
4. Подходит ли одежда по текущей температуре?
  - a. Если нет, то возвращение к пункту 2.
  - b. Если вещь есть, то переход к пункту 5
5. Соответствует ли одежда полу пользователя?
  - a. Если нет, то возвращение к пункту 2.
  - b. Если вещь есть, то переход к пункту 6
6. Добавление в массив подходящей шаблонной одежды.



Следующим этапом идет работа с добавленной пользователем одеждой.

Её обработка происходит в несколько шагов:

1. Поочередно берется каждая вещь, добавленная пользователем.
2. Проверка одежды на тип.
  - a. Если тип другой, то возвращение к пункту 1.
  - b. Если тип подошел, то переход к пункту 3.
3. Подходит ли одежда по текущей температуре?
  - a. Если нет, то возвращение к пункту 1.
  - b. Если да, то переход к пункту 4.
4. Соответствует ли одежда полу пользователя?
  - a. Если нет, то возвращение к пункту 1.
  - b. Если да, то переход к пункту 5.
5. Добавление в массив подходящей пользовательской одежды.

После происходит работа с подходящей одеждой. Подбор происходит в несколько шагов:

1. Среди шаблонной и пользовательской одежды есть подходящая?
  - a. Если нет, то возвращение к пункту 2.
  - b. Если есть, то возвращение к пункту 3.
2. Вывод картинки, что подходящей одежды нет.
3. Выбор случайной одежды среди подходящей.

Разработанная схема алгоритма представлена на рисунке 13.

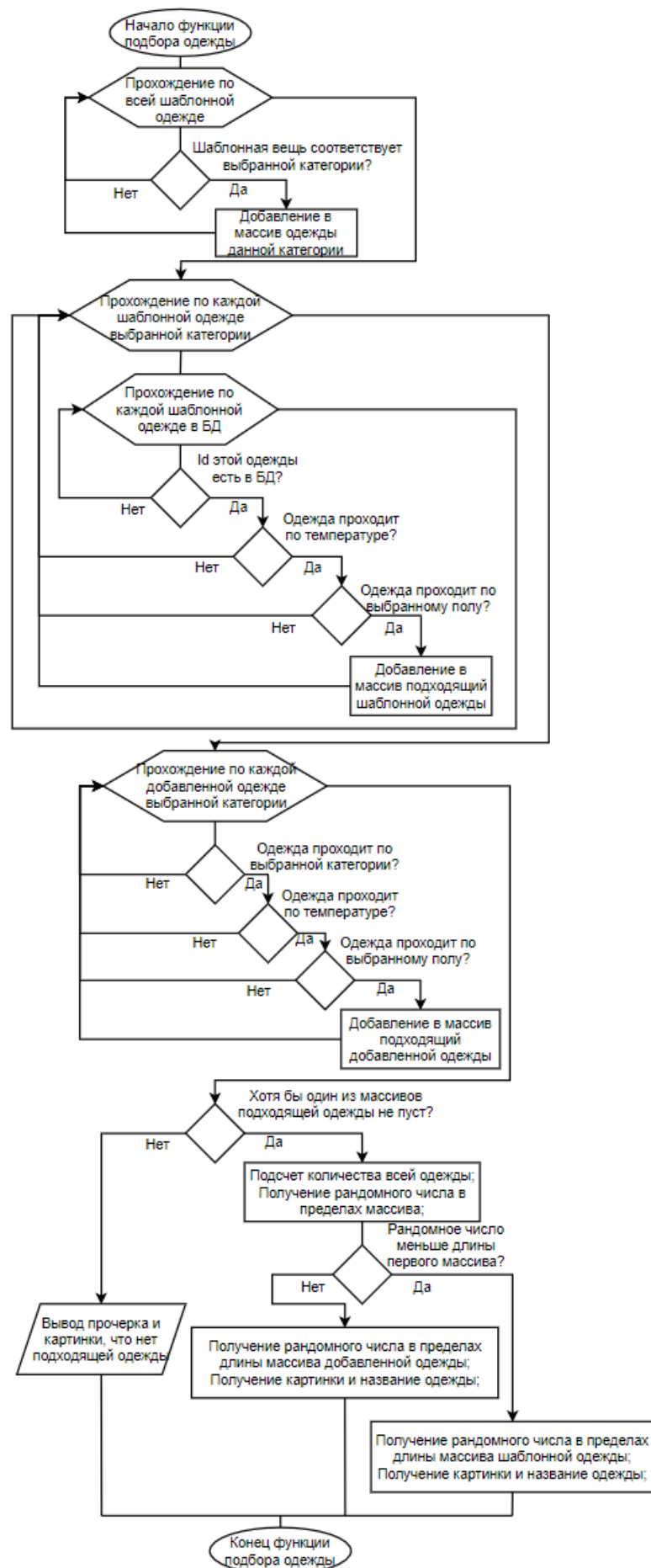


Рисунок 13 – алгоритм подбора одежды

## 9 Выбор метода хранения данных

Приложение содержит в себе данные, которые при выходе из приложения не должны потеряться, поэтому необходимо предусмотреть метод хранения этих данных.

В приложении хранятся следующие данные:

- шаблонная одежда. Она должна быть в приложении сразу после установки у каждого пользователя, то есть на всех возможных устройствах, поэтому хранится она должна в памяти приложения;
- пользовательская одежда. Она у каждого пользователя своя, поэтому каждая новая добавленная одежда хранится в БД, а при удалении, удаляется из БД.
- пользовательские настройки. Они у всех общие, но выбраны могут быть по-разному, поэтому настройки каждого отдельного пользователя сохраняются на каждом отдельном устройстве.

В разработанном ПО используется три метода хранения информации:

- БД Room, которая является более удобной надстройкой SQL [4];
- `sharedpreferences` для небольшого количества информации, которая сохраняется в отдельный файл [5];
- просто массивы, которые хранят параметры шаблонной, то есть заранее заданной, одежды.

Шаблонная одежда, и информация о ней, хранится в массивах типа `List`.

Пример хранения шаблонной одежды из категории “Верхняя одежда”:

```
private static List<ClothInitial> generateOuterwear() {  
    List<ClothInitial> clothes = new ArrayList<>();  
    clothes.add(new ClothInitial(51, "Бомбер",  
R.drawable.outerwear_bomber, 5, 15, ClothesType.OUTERWEAR,  
ClothGender.UNISEX));  
    return clothes;  
}
```

Это сделано для того, чтобы уменьшить количество обращений и не создавать лишних запросов к БД. Последовательность того, что хранит элемент массива: id, название одежды, ссылку на картинку, минимальная температура, максимальная температура, тип, пол. В БД шаблонной одежды хранится только информация о наличии её у пользователя. Если пользователь отмечает одежду, то она добавляется в таблицу и ставится 1 в поле наличие. Когда пользователь снимает отметку, то одежда удаляется из БД, это сделано, чтобы не хранить лишнюю информацию, которая занимает память.

Данные одежды, добавленной пользователем, полностью хранится в БД. Таблица пользовательской одежды содержит поля: id, название, название картинки, максимальная и минимальная температура, пол и тип.

Таблицы базы данных шаблонной и пользовательской одежды показаны на рисунке 14.

ClothCustom	
PK	id int
	name String imageFilePath String tempMin int tempMax int gender ClothGender type ClothesType

ClothInitial	
PK	id int
	isAvailable boolean

Рисунок 14 – Таблицы базы данных

С помощью sharedpreferences хранится информация, заданная в настройках (теплоощущение, температурная шкала и пол). В окне “Настройки”, при нажатии кнопки “Сохранить” происходит сохранение выбранной информации. Подгрузка ранее выбранных параметров происходит в момент запуска приложения.

## 10 Разработка диаграммы уровня реализации

После окончания разработки ПО была разработана диаграмма классов уровня реализации, чтобы представить созданные классы с их полями и методами. Диаграмма классов уровня реализации показана на рисунке 15.

В процессе написания кода были созданы классы, которые можно разделить по категориям того, за что они отвечают.

За определение местоположение устройства пользователя отвечают классы:

- ApiClient – хранит API сайта openweather и создает запрос;
- WeatherData – преобразует полученные данные из float формата в String;
- Main – указаны параметры, которые необходимо будет сохранить при принятии JSON запроса;
- WeatherRequest – принимает запрос информации о погоде;
- ApiInterface – интерфейс, хранящий в себе ссылку на пользователя с названием местоположения.

За базу данных отвечают следующий классы:

- ClothCustomEntity – описана таблица пользовательской одежды;
- ClothInitialEntity – описана таблица шаблонной одежды;
- ClothCustomDao – прописаны SQL запросы на получение всей одежды, поиска по id, по типу, а также дообвление, изменение и удаление (из БД). Все для пользовательской одежды;
- ClothInitialDao – прописаны SQL запросы на получение всей одежды, поиска по id, по типу, а также дообвление и удаление (стоит галочка или нет). Все для шаблонной одежды;
- ClothesAppDatabase – создание сущности базы данных и привязка к каждой из таблиц конкретных методов;
- Converters – преобразует данные из enum в int, когда идет запрос к БД, и наоборот, когда информация поступает из БД;

- RoomWrapper – создание базы данных, если её нет, или её возвращение при сторонних запросах.

Класс SharedPrefsWrapper отвечает за сохранение и получение (при запуске приложения) параметров, выбранных в настройках.

Следующие классы отвечают за принятие данных из БД для дальнейшей работы с ней:

- ClothesInitialFactory – хранит в себе массивы шаблонной одежды каждого типа;
- ClothCustom и ClothInitial – файлы принимают всю возможную информацию для дальнейшей обработки. Первый для пользовательской одежды, второй для шаблонной;
- CustomClothModel и InitialClothModel – файлы принимают информацию, необходимую для отображения изображений. Первый для пользовательской одежды, второй для шаблонной;
- IClothModel – позволяет в одном списке объединить объекты из файлов CustomClothModel и InitialClothModel.

Классы, отвечающие за работу с одеждой в гардеробе:

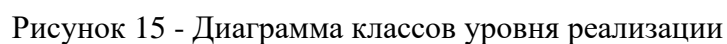
- ClothesListActivity – открытие подкатегории, редактирования, создания новой одежды и вызов функции вывода. Принимает информацию, что кнопки были нажаты и посылает эту инфу в ClothesListAdapter;
- ClothesListAdapter – отрисовывает одежду и названия, заполняя элементы списка вьюх данными. А также, обрабатывает информацию, какая из кнопок были нажаты и вызывает соответствующую функцию в ClothesListFragment;
- ClothesListFragment – обрабатывает добавление/удаление шаблонной одежды из БД. И отбирает ту одежду, которая будет отрисовываться в соответствии с полом и наличием у пользователя.

Классы, отвечающие за создание новой одежды или редактирование уже существующей пользовательской:

- NewClothActivity – принимаем операцию из Activity, проверяет что за действие выбрано, выбирает id или тип одежды и вызывает функцию обработчик из файла NewClothFragment;
- NewClothFragment – открывает соответствующее окно (создания или редактирования) и отвечает за все действия, что происходят при создании и редактировании.

А также есть классы:

- ClothesMainActivity – прописана работа навигационной (боковой) панели;
- RecommendationFragment – получает геолокацию, выводит температуру, корректирует температуру в зависимости от теплоощущения, подбирает рандомную одежду и выводит её на главный экран;
- WardrobeFragment – общая вкладка гардероба. Создает и обрабатывает кнопки открытия подкатегорий и кнопку “расширить гардероб” ;
- SettingsFragment – работа с вкладкой “Настройки”, обработка выбора каждого из параметров (теплоощущения, гендера и температурной шкалы).





## 11 Разработка диаграммы компоновки

Для более детального понимания связи компонентов и работы приложения в целом, была разработана диаграмма компоновки показана на рисунке 16 [6].

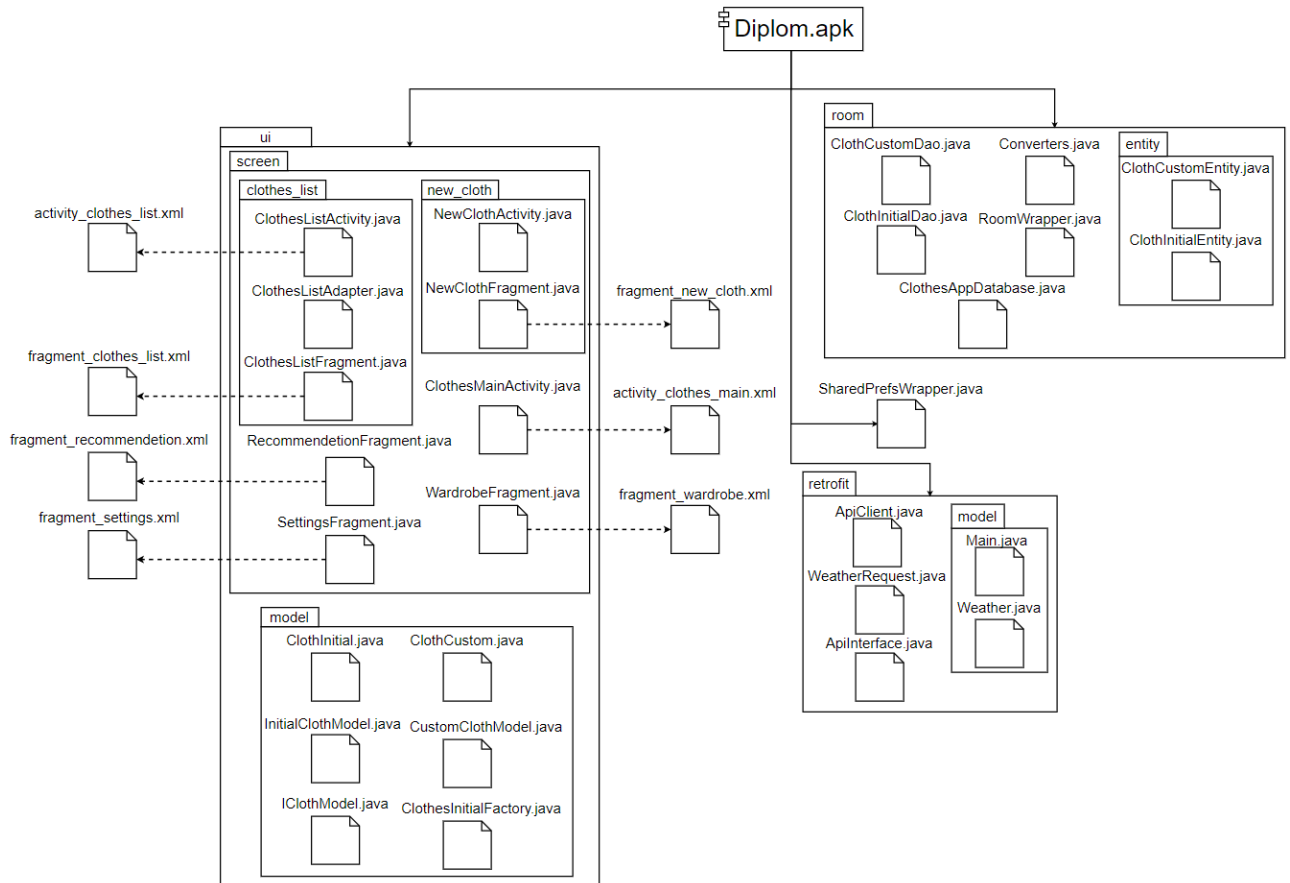


Рисунок 16 – Диаграмма компоновки

Исполняемый арк файл собирается из исполняющих файлов с расширением .java и вспомогательными файлами с расширением .xml, которые содержат обертку файлов .java. Файлы разделены по трем папкам, с собственными подпапками, в соответствии с выполняемыми функциями.

Файлы, входящие в папку ui, отвечают за взаимодействие пользователя с интерфейсом приложения, а также за логику работы программы (ClothesListActivity, ClothesListAdapter, ClothesListFragment, NewClothActivity, NewClothFragment, WardrobeFragment, ClothesMainActivity, RecommendationFragment, SettingsFragment, ClothInitial, ClothCustom, InitialClothModel, CustomClothModel, IClothModel, ClothesInitialFactory).

Файлы, входящие в папку room, отвечают за базу данных приложения (ClothCustomDao, ClothInitialDao, Converters, RoomWrapper, ClothesAppDatabase, ClothCustomEntity, ClothInitialEntity).

Файлы, входящие в папку retrofit, отвечают за взаимодействие приложения с интернет-ресурсом openweather (ApiClient, WeatherRequest, Main, Weather и ApiInterface).

## **Заключение**

В процессе прохождения преддипломной практики были описаны этапы разработки программного продукта. Создана структура приложения, разработана концептуальная модель предметной области, определены основные функции приложения и разработана диаграмма вариантов использования, разработан граф состояний интерфейса и внешний вид форм интерфейсов, а также разработан алгоритм подбора одежды и выбраны методы хранения данных.

## СПИСОК ИСПОЛЬЗУЕМОХ ИСТОЧНИКОВ

- 1 Ещё раз про семь основных методологий разработки / Хабр [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/company/edison/blog/269789/> (дата обращения: 07.02.2022)
- 2 Использование диаграммы вариантов использования UML при проектировании программного обеспечения / Хабр [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/post/566218/> (дата обращения: 08.02.2021)
- 3 Наука и образование: научно-техническое издание: Экспертная система «Кардиолог» [Электронный ресурс]. – Режим доступа: <http://engineering-science.ru/doc/95195.html> (дата обращения: 11.02.2022)
- 4 Room: Хранение данных на Android для всех и каждого / Хабр [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/post/336196/> (дата обращения: 17.12.2021)
- 5 SharedPreferences | Android Developers [Электронный ресурс]. – Режим доступа: <https://developer.android.google.cn/reference/android/content/SharedPreferences?hl=ru> (дата обращения: 20.10.2021)
- 6 Иванова, Г.С. Технология программирования: Учебник для вузов – М.: Изд-во МГТУ им. Н.Э. Баумана, 2002. 238 с.