**Содержание**

[Введение 2](#_Toc162857037)

[1. Аналитическая часть 3](#_Toc162857038)

[1.1. Интервью с заказчиком 3](#_Toc162857039)

[1.2. Описание предметной области 4](#_Toc162857040)

[1.3. Описание вариантов использования приложения 6](#_Toc162857041)

[1.4. Классификация пользователей результатов проекта 7](#_Toc162857042)

[1.5. Обоснование выбора средств разработки 7](#_Toc162857043)

[1.6. Описание структуры данных 9](#_Toc162857044)

[1.7. Описание архитектуры приложения 9](#_Toc162857045)

[1.8. Концептуальная модель 11](#_Toc162857046)

[1.9. Проектирование интерфейса 11](#_Toc162857047)

[2. Технологическая часть 17](#_Toc162857048)

[2.1. Описание проектирования логики и данных 17](#_Toc162857049)

[2.4. Определение транзакционных функциональных типов 18](#_Toc162857050)

[2.5. Расчет количества функциональных точек 20](#_Toc162857051)

[2.5. Определение основных технико-экономических показателей 22](#_Toc162857052)

[3 Экономическая часть 24](#_Toc162857053)

[3.3 Предварительная оценка трудоемкости разработки методом PERT 24](#_Toc162857054)

[Заключение 29](#_Toc162857055)

[Использованные источники информации 30](#_Toc162857056)

## Введение

В связи с распространением смартфонов среди студентов и преподавателей появляется возможность проверять знания студентов в удобном формате и без использования компьютерных классов.

Тема актуальна в связи с распространением смартфонов и доступностью интернета, образуется возможность проводить тестирование на предмет знаний студентов по выбранным дисциплинам.

Целью проектирования курсового проекта является создание мобильного приложения для составления и проведения тестирования среди студентов.

Предметом проектирования курсового проекта является разработка мобильного приложения.

Объектом исследования курсового проекта является тестирование студентов.

Для достижения цели курсового проекта нужно выполнить следующие задачи:

1. Анализировать предметную область;

2. Реализовать пользовательский интерфейс;

3. Реализовать возможности тестирования;

При разработке мобильного приложения были выявлены следующие

функциональные требования:

1. Регистрация и авторизация пользователя;

2. Возможность присваивания роли пользователю;

3. Возможность составлять тестирования;

4. Возможность прохождение тестирования;

5. Возможность получения результата.

Основанием для разработки проекта является учебный план специальности 09.02.07 «Информационные системы и программирование»

## 1. Аналитическая часть

## 1.1. Интервью с заказчиком

Интервью с заказчиком, это процесс беседы с представителем заказчика или ответственным за проект лицом, во время которого собирается информация о требованиях, целях, ожиданиях и предпочтениях заказчика относительно приложения для тестирования студентов. Вот несколько вопросов, которые могут быть заданы в интервью с заказчиком:

1. Каковы общие цели и назначение этого приложения для тестирования студентов?
2. Какие основные функциональные возможности вы ожидаете от приложения для тестирования студентов?
3. Какие типы вопросов и форматы тестов вам необходимы? (например, множественный выбор, соответствие, открытый вопрос и т.д.)
4. Нужно ли вам возможность генерировать случайные тесты или адаптировать их для конкретных групп студентов?
5. Какие дополнительные функциональности или интеграции вы бы хотели видеть в приложении (например, импорт и экспорт данных, возможность загрузки файлов и т.д.)?
6. Какую информацию вы хотели бы видеть в отчетах по результатам тестирования студентов?
7. Какие требования по безопасности и доступу к данным вы имеете в виду?
8. Какую поддержку после внедрения вам может понадобиться?

## 1.2. Описание предметной области

Сейчас трудно представить человека без смартфона. Но, не во всех учебных заведениях есть возможность оборудовать достаточное количество компьютерных классов. На помощь приходит смартфон. Он не требует использовать его в компьютерном классе, тем самым сможет упростить этапы обучения, как и для студента, так и для преподавателя.

Мобильное приложение, представляющее собой максимально удобный инструмент для тестирования преподавателями студентов по разным преподаваемым дисциплинам не используя компьютерный класс, но зато используя смартфон.

Мобильное приложение тестирования дает возможность преподавателю составлять тесты для дальнейшего прохождения их студентами, также дает возможность выбора тестов для дальнейшего прохождения, по завершению прохождения тестирования система выдает результаты тестирования студентам и преподавателю.

Также данное мобильное приложение имеет функции регистрации/авторизации с выбором роли (студент или преподаватель), возможность сохранения результатов тестирования уже пройденных тестов.

Исходя из выявленных функциональных требований была построена Use-case диаграмма, представленная на рисунке 1:

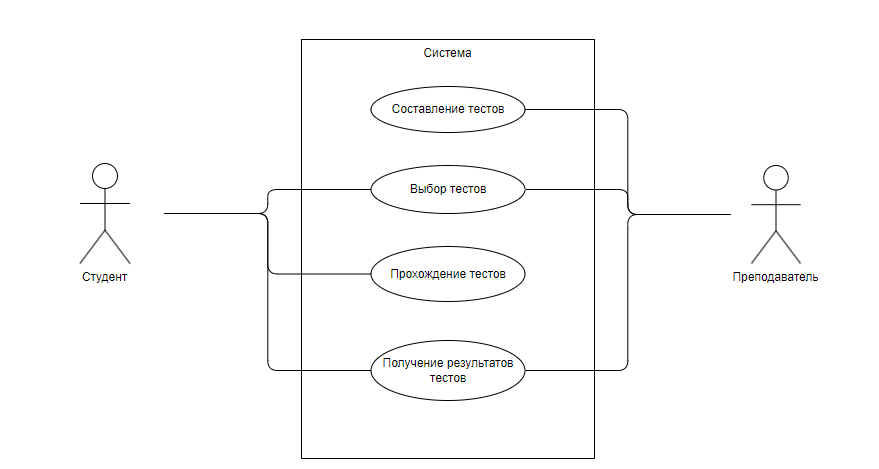


Рисунок 1. Use-case диаграмма.

На основе составленной Use-case диаграммы было выделено 4 прецедента. Рассмотрим прецедент “Выбор тестов”:

Таблица 1 — Прецедент “Выбор тестов”

|  |  |
| --- | --- |
| Название прецедента | Выбор тестов. |
| Цель | Выбор теста студентом составленного преподавателем. |
| Актеры | Пользователь студент(инициатор), Пользователь преподаватель, Система. |
| Ссылки (предусловия) | Реализованы прецеденты: Получение результатов тестирования. |
| Результаты (постусловия) | Пользователь студент заканчивает тестирование. Система определяет количество правильных и неправильных ответов. После этого система отправляет результат студенту и преподавателю для выставления оценки |

## 1.3. Описание вариантов использования приложения

На основе выявленных прецедентов в Use-case диаграмме был составлен основной успешный сценарий, представленный в таблице 2:

Таблица 2 — Основной вариант использования для студента

|  |  |
| --- | --- |
| Действия актера | Отклик системы |
| 1. Пользователь студент выбирает тест составленный преподавателем.  2. Пользователь студент проходит тестирование.  4. Пользователь студент завершает тестирование. | 3. Система ведет подсчет верных и неверных ответов студента и отправляет результат пользователю студенту и пользователю преподавателю.  5. Система отправляет результат тестирования. |

На основе успешного варианта использования можно составить альтернативный поток событий:

1. Произошла ошибка в системе. Система выдает сообщение об ошибке.

3. Данные отсутствуют в системе. Система выдает сообщение об ошибке.

4. Данные отсутствуют в системе. Система выдает сообщение об ошибке.

Таблица 3 — Основной вариант использования для преподавателя

|  |  |
| --- | --- |
| Действия актера | Отклик системы |
| 1. Пользователь преподаватель создает вариант тестирования.  2. Пользователь преподаватель заполняет тест вопросами и ответами.  4. Пользователь преподаватель закрывает доступ к тестированию. | 3. Система ведет подсчет верных и неверных ответов студента и отправляет результат пользователю студенту и пользователю преподавателю.  5. Система отправляет результат тестирования. |

На основе успешного варианта использования можно составить альтернативный поток событий:

1. Произошла ошибка в системе. Система выдает сообщение об ошибке.

3. Данные отсутствуют в системе. Система выдает сообщение об ошибке.

4. Данные отсутствуют в системе. Система выдает сообщение об ошибке.

## 1.4. Классификация пользователей результатов проекта

Классификация пользователей результатов проекта для приложения тестирования студентов может быть следующей:

1. Администраторы/преподаватели: Эта группа пользователей будет ответственна за создание и управление тестами, добавление вопросов, назначение тестов студентам, а также просмотр и анализ результатов тестирования. Они имеют специальные права доступа для администрирования и управления системой тестирования.

2. Студенты: В эту группу входят сами студенты, которые будут проходить тесты. Они имеют доступ для выполнения тестов, выбора ответов на вопросы и отправки своих ответов.

# 1.5. Обоснование выбора средств разработки

Выбор средств разработки Kotlin может быть обоснован следующими факторами:

1. Мультиплатформенность: Kotlin предоставляет возможность разработки как для платформы Java, так и для платформы Android. Это позволяет максимально использовать код и ресурсы и повышает эффективность разработки приложений для разных платформ.

2. Легкая интеграция: Kotlin может работать существующим Java-кодом, что делает его простым в использовании в существующих проектах или при создании новых. Он также имеет отличную совместимость с мощной экосистемой инфраструктуры Java.

3. Безопасность: Kotlin предоставляет набор функций и возможностей, которые помогают предотвратить ошибки и сбои программы, такие как проверки на null, безопасные вызовы и различные проверки типов. Это способствует созданию более надежного и стабильного кода.

4. Краткость и выразительность: Kotlin имеет простой и понятный синтаксис, который делает код более читабельным и удобным для понимания. Он также предоставляет множество удобных функций, таких как функции расширения и функциональное программирование, что упрощает разработку и сокращает количество кода.

5. Активная поддержка и сообщество: Kotlin активно развивается и поддерживается компанией JetBrains, что гарантирует обновления и исправление ошибок. Он также имеет развитое сообщество разработчиков, которые предоставляют полезные ресурсы и помощь во время разработки.

В целом, выбор Kotlin в качестве средства разработки может быть обоснован его множеством преимуществ, таких как мультиплатформенность, легкая интеграция, безопасность, краткость и выразительность, а также активная поддержка и сообщество.

# 1.6. Описание структуры данных

По результатам описания предметной области была составлена ERD диаграмма, представленная на рисунке 2.



Рисунок 2. ER – Диаграмма

# 1.7. Описание архитектуры приложения

Архитектура приложения тестирования для студентов может быть организована в соответствии с принципами модульности и разделения ответственности. Вот возможное описание основных компонентов архитектуры приложения:

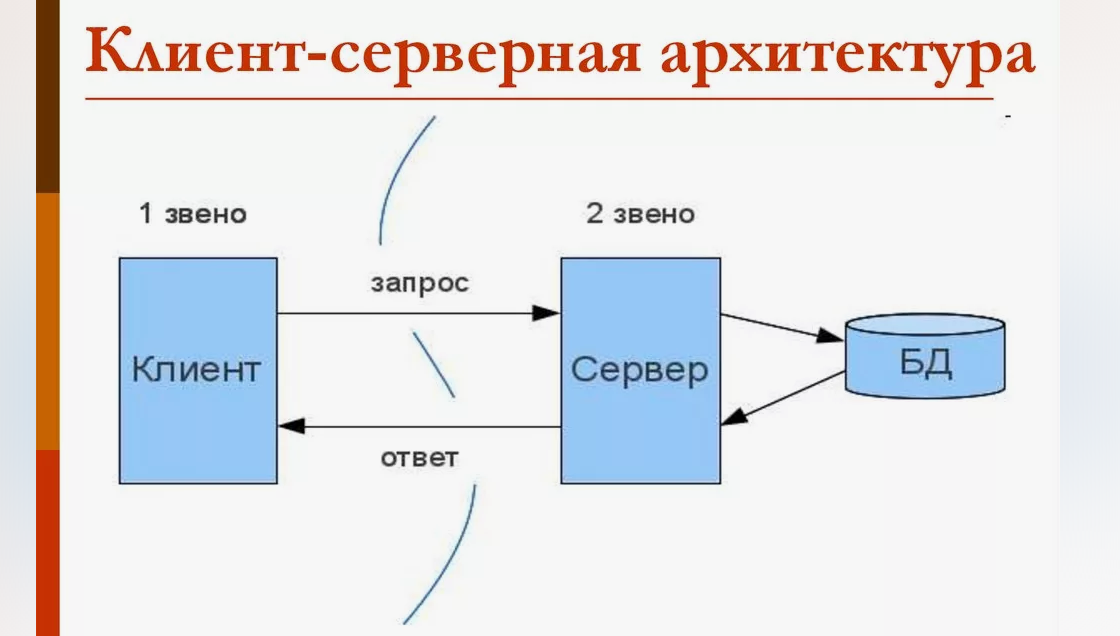
1. Клиентский интерфейс: Это часть приложения, которая взаимодействует с пользователями - студентами. Она включает в себя пользовательский интерфейс (UI), который позволяет студентам пройти тесты, выбирать ответы на вопросы и просматривать результаты.

2. Бизнес-логика: Этот компонент отвечает за обработку и управление данными, логикой бизнес-процессов и выполнением операций с тестами и результатами. Он включает в себя функции для создания, редактирования и удаления тестов и вопросов, а также обработки ответов студентов и расчета результатов.

3. Серверная инфраструктура: Эта часть архитектуры отвечает за хранение данных приложения и взаимодействие с базой данных. Она включает в себя серверное API, которое обрабатывает запросы от клиента, а также модули доступа к данным, которые обеспечивают взаимодействие с базой данных.

4. База данных: В этой части архитектуры хранятся все данные, связанные с тестами, вопросами, результатами и студентами. База данных должна быть спроектирована таким образом, чтобы обеспечить эффективное хранение и извлечение данных, а также обеспечить целостность данных с помощью соответствующих ограничений и отношений.

5. Аутентификация и безопасность: Это компоненты, отвечающие за аутентификацию и авторизацию студентов, а также за защиту данных приложения. Безопасность может включать в себя шифрование данных, управление правами доступа и механизмы защиты от несанкционированного доступа.



# 1.9. Проектирование интерфейса

При анализе предметной области был спроектирован проект интерфейса приложения, представленный на рисунках 4-10. Каждая форма выполняет определенную функцию.

|  |  |
| --- | --- |
| Рисунок 4. Форма “Авторизация” | Рисунок 5. Форма “Регистрация” |

|  |  |
| --- | --- |
| Рисунок 6. Окно “Выбор теста” | Рисунок 7. Окно “Вопросы” |

|  |  |
| --- | --- |
| Рисунок 8. Окно “Результаты тестирования для студента” | Рисунок 9. Окно “Результата тестирования для преподавателя” |

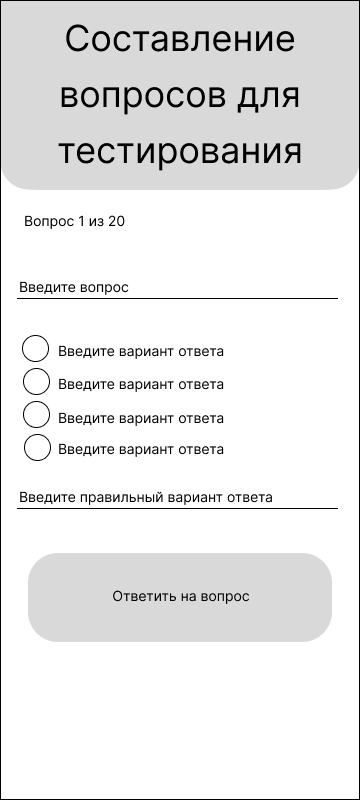


Рисунок 10. Окно “Составление вопросов для тестирования”

После создания проекта интерфейса была создана схема перемещений по приложению, представленная на рисунке 11:

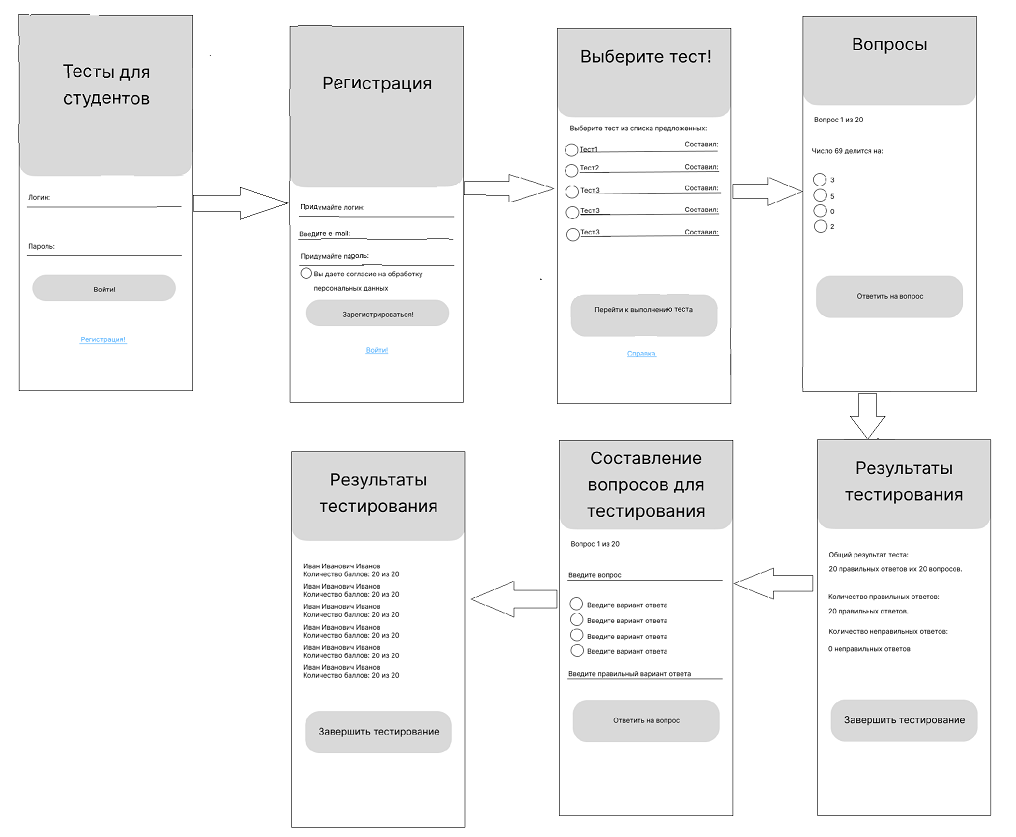


Рисунок 11. Окно “ Схема перемещений по приложению”

На рисунке ясно видно, как происходит работа в приложении.

# 2. Технологическая часть

# 2.1. Описание проектирования логики и данных

По результатам описания предметной области была составлена диаграмма классов проекта, представленная на рисунке 2.

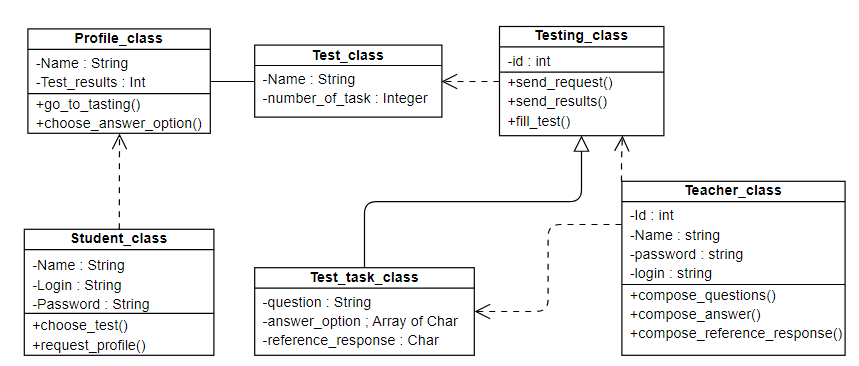


Рисунок 2. Диаграмма классов.

Следующим шагом проектирования является составление ER-диаграммы, показанная на рисунке 3. Диаграмма отображает состав таблиц базы данных проекта, а также связи между ними.



Рисунок 3. ER – Диаграмма

На основании ER-диаграммы выполнили оценку функциональных типов по данным. Сначала определили тип данных системы: EIF или ILF. Затем произвели подсчет функциональных точек, связанных с данными, определили сложность данных.

Таблица 4 – Оценка функциональных типов по данным

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Информационный объект** | **Тип** | **RET** | **DET** | **Сложность** | **Коэффициент** |
| User | ELF | 1 | 4 | Простая | 7 |
| Profile\_Student | ELF | 1 | 4 | Простая | 7 |
| Profile\_Teacher | ELF | 1 | 4 | Простая | 7 |
| TestTask | ELF | 1 | 5 | Простая | 7 |
| Test | ILF | 1 | 7 | Простая | 7 |

Коэффициент определяется в зависимости от принадлежности объекта к ILF или EIF.

# 2.4. Определение транзакционных функциональных типов

Следующий шаг анализа по методу функциональных точек производился для каждой формы приложения. Были определены типы транзакций, выполнена оценка сложности транзакций, а также определение сложности и расчет коэффициента. Все расчеты представлены в таблицах 5-12.

Таблица 5 – Форма Авторизация

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Транзакция** | **Тип** | **FTR** | **DET** | **Сложность** | **Коэффициент** |
| Ввод логина | EI | 1 | 1 | Легкая | 3 |
| Ввод пароля | EI | 1 | 1 | Легкая | 3 |
| Авторизация | EI | 1 | 2 | Легкая | 3 |

Таблица 6 – Форма Регистрации

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Транзакция** | **Тип** | **FTR** | **DET** | **Сложность** | **Коэффициент** |
| Ввод логина | EI | 1 | 1 | Легкая | 3 |
| Ввод пароля | EI | 1 | 1 | Легкая | 3 |
| Ввод e-mail | EI | 1 | 1 | Легкая | 3 |
| Авторизация | EI | 1 | 1 | Легкая | 3 |

Таблица 7 – Форма Выбор теста

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Транзакция** | **Тип** | **FTR** | **DET** | **Сложность** | **Коэффициент** |
| Вывод списка тестов | EO | 1 | 10 | Средняя | 11 |
| Выбор теста | EI | 1 | 2 | Легкая | 3 |

Таблица 8 – Форма Вопросы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Транзакция** | **Тип** | **FTR** | **DET** | **Сложность** | **Коэффициент** |
| Вывод вопроса | EO | 1 | 1 | Легкая | 3 |
| Вывод вариантов ответов | EO | 1 | 4 | Средняя | 5 |
| Выбор теста | EI | 1 | 2 | Легкая | 3 |

Таблица 9 – Форма “Вывод результатов для студентов”

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Транзакция** | **Тип** | **FTR** | **DET** | **Сложность** | **Коэффициент** |
| Вывод результатов | EO | 1 | 3 | Легкая | 3 |
| Завершение тестирования | EI | 1 | 2 | Легкая | 3 |

Таблица 10 – Форма “Вывод результатов для преподавателя”

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Транзакция** | **Тип** | **FTR** | **DET** | **Сложность** | **Коэффициент** |
| Вывод результатов отдельного студента | EO | 1 | 3 | Легкая | 3 |
| Завершение тестирования | EI | 1 | 2 | Легкая | 3 |

# 2.5. Расчет количества функциональных точек

Таблица 11 — Основной успешный сценарий

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Категория функциональных типов** | **Простые** | **Средние** | **Сложные** | **Кол-во точек** |
| Количество внутренних логических файлов | 7 | 8 | 0 | 15 |
| Количество внешних интерфейсных файлов | 6 | 6 | 0 | 12 |
| Количество входных элементов | 4 | 4 | 0 | 8 |
| Количество выходных элементов | 5 | 2 | 0 | 7 |
| Количество внешних запросов | 7 | 3 | 0 | 10 |
| **Количество функциональных точек (UFP)** | | | | 52 |

Чтобы определить суммарное количество не выровненных функциональных точек, нужно рассчитать по формуле:

(2)

Общее суммарное количество не выровненных функциональных точек UFP будет составлять по формуле (7):

Таблица 12 — Основной успешный сценарий

|  |  |  |
| --- | --- | --- |
| **№** | **Наименование фактора** | **Значение** |
| 1 | Обмен данными | 5 |
| 2 | Распределенная обработка данных | 0 |
| 3 | Производительность | 5 |
| 4 | Эксплуатационные ограничения | 0 |
| 5 | Частота транзакций | 0 |
| 6 | Ввод данных в режиме «онлайн» | 5 |
| 7 | Эффективность работы конечных пользователей | 5 |
| 8 | Онлайновое обновление | 0 |
| 9 | Сложная обработка | 0 |
| 10 | Повторное использование | 5 |
| 11 | Простота установки | 5 |
| 12 | Простота эксплуатации | 5 |
| 13 | Количество возможных установок на различных платформах | 0 |
| **Суммарное значение коэффициентов (М)** | | 35 |

Суммарное значение коэффициентов осуществляется простым суммированием по формуле:

(3)

Расчёт суммарного значения коэффициентов будет составлять по формуле (8):

Расчёт значения фактора выравнивания производится по формуле:

(4)

Расчёт значения факторов выравнивания будет составлять по формуле (9):

Начальная оценка количества выровненных функциональных точек для программного приложения определяется по формуле:

(5)

Начальная оценка количества выровненных функциональных точек для программного приложения будет составлять по формуле (10):

# 2.5. Определение основных технико-экономических показателей

Расчёт количества функциональных точек:

Показатели LOC для языка Kotlin:

Для определения технико-экономических показателей нужно сначала определить тип системы. Данная система относится ко второму типу, так как будет реализована на языке Kotlin.

После расчёта количества функциональных точек необходимо определить размерность мобильного приложения. Размерность определяется по формуле:

(6)

Размерность мобильного приложения рассчитана по формуле (11):

Значения параметров A и E определяются из таблицы коэффициентов математической модели оценки трудозатрат в зависимости от типа программной системы.

Оценка трудозатрат проводится с помощью степенной функции базовой модели COCOMO. Значения параметров A и E определяются из таблицы коэффициентов математической модели оценки трудозатрат в зависимости от типа программной системы.

Единица измерения R в данной формуле - тысяча строк:

(7)

Расчет количества человек в месяц произведен по формуле (12):

Средняя численность сотрудников, занятых в проекте, срок реализации которого 4 месяца, составляет:

(8)

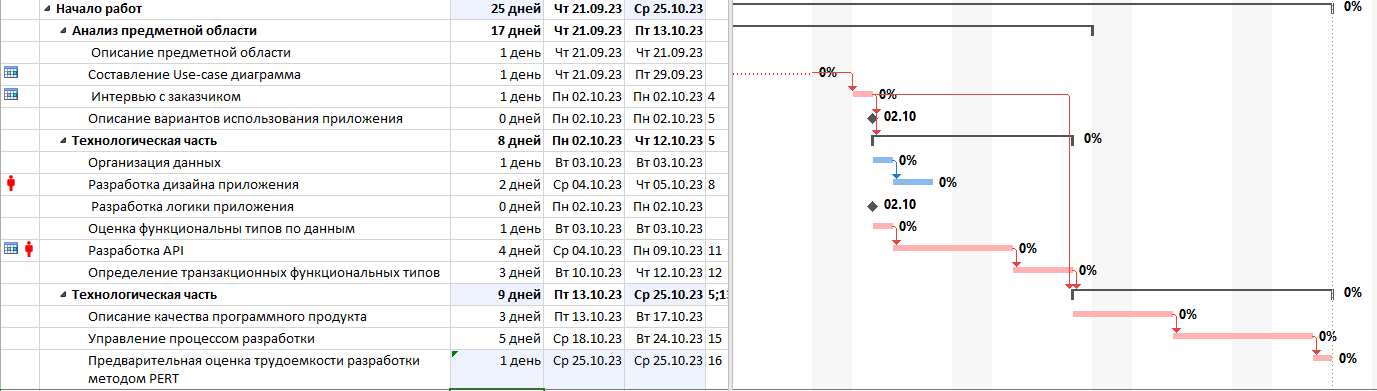
Средняя численность сотрудников рассчитана по формуле (13):

Таким образом, метод функциональных точек определил следующие основные технико-экономические показатели:

* трудозатраты на разработку системы за 4 месяца составят 4 человеко-месяцев;
* необходимые людские ресурсы при реализации системы за 4 месяца составят 1 чел.

# 3 Экономическая часть

А



## 3.3 Предварительная оценка трудоемкости разработки методом PERT

На основании перечня функций, выявленных на этапе анализа требований была выполнена экспертная оценка трудоемкости разработки приложения, с учетом его функций, представленная в таблице 3.

Таблица 3 – Экспертная оценка трудоемкости разработки приложения

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Функция** | **Лучший случай** | **Наиболее вероятный** | **Худший случай** | **Ожидаемый случай** |
| Составление тестов | 4 | 2 | 6 | 3 |
| Выбор тестов | 2 | 4 | 6 | 4 |
| Прохождение тестов | 2 | 4 | 5 | 4 |
| Получение результатов тестов | 4 | 2 | 8 | 3 |
| **Итого** | **12** | **12** | **25** | **14** |

В предварительной оценке трудоёмкости разработки представлено количество дней, за которое возможно реализовать функции из Use-Case диаграммы используя формулу PERT (табл.3).

Оценку средней трудоемкости по каждому элементарному пакету можно определить по формуле:

(1)

1. Составление тестов.

Oi = 4

Mi = 2

Pi = 6

Средняя трудоемкость рассчитана по формуле (1):

1. Выбор тестов.

Oi = 2

Mi = 4

Pi = 6

Средняя трудоемкость рассчитана по формуле (1):

1. Прохождение тестов.

Oi = 2

Mi = 4

Pi = 5

Средняя трудоемкость рассчитана по формуле (1):

1. Получение результатов тестирования.

Oi = 4

Mi = 2

Pi = 8

Средняя трудоемкость рассчитана по формуле (1):

На основе экспертная оценки трудоемкости разработки мобильного приложения приложения было выявлено, что ожидаемый случай выполнения разработки программного продукта займет 14 дней.

– это среднеквадратичное отклонение Данный показатель можно расчитать по формуле 2.

(2)

1. Для составления тестирования, среднеквадратичное отклонение рассчитано по формуле (2).
2. Для выбора теста, среднеквадратичное отклонение рассчитано по формуле (2).
3. Для прохождения тестирования, среднеквадратичное отклонение рассчитано по формуле (2).
4. Для получения результатов тестирования, среднеквадратичное отклонение рассчитано по формуле (2).

СКО – это среднеквадратичное отклонение для оценки суммарной трудоёмкости. Данный показатель можно расчитать по формуле 3.

(3)

Среднеквадратичное отклонение для оценки суммарной трудоёмкости рассчитана по формуле (3).

Суммарная трудоёмкость проекта рассчитывается по формуле 4.

(4)

Е – это суммарная трудоёмкость проекта, которая определяется по формуле (4).

– это оценка суммарной трудоёмкости проекта, не превышающая с вероятностью в 95%, которая расчитывается по формуле 5.

(5)

Оценка суммарной трудоёмкости проекта, которую мы не превысим с вероятностью в 95% рассчитана по формуле (5).

Если сотрудник назначен на проект, то тратить он будет 60 – 80% своего рабочего времени. Поэтому, в месяц сотрудник будет работать по проекту, примерно.

Следовательно, трудоёмкость проекта в человеко-месяцах составит, приблизительно.

2596 / 226 ≈ 11,4

# Заключение

До начала реальной части проекта был произведён расчёт трудоёмкости работы. Было выявлено, какое количество человеко-часов необходимо для разработки проекта.

В результате работы была создана логическая модель базы данных в документе Microsoft Visio. Разработаны окна интерфейса, а также схема перемещения между ними. Информационная система состоит из:

ER-диаграммы логической модели базы данных, разработанной в Microsoft Visio;

Окна интерфейса будущего приложения, разработанные в Microsoft Visio.

В программе предусмотрены возможности:

* Просмотра данных;
* Добавления данных;
* Редактирования данных;
* Регистрации и Авторизации.

В ходе работы закреплена технология проектировки баз данных и диаграмм классов в Microsoft Visio, а также планирования создания окон интерфейса и перемещений между ними будущих экранов мобильного приложения. Были изучены и проведены расчёты необходимого времени и количества человек на разработку приложения.

# Использованные источники информации

1. ГОСТ Р ИСО 9000-2008 «Системы менеджмента качества»;
2. ГОСТ Р ИСО/МЭК 12207-99 «Процессы жизненного цикла программных средств»;
3. Мамедли Р. Э. Базы данных. Лабораторный практикум : учебное пособие для вузов / Р. Э. Мамедли. – Санкт-Петербург : Лань, 2023. – 152 с.;
4. Нестеров С. А. Интеллектуальный анализ данных с использованием SQL Server : учебник для вузов / С. А. Нестеров. – Санкт-Петербург : Лань, 2023. – 160 с.;
5. Макшанов А. В. Большие данные. Big Data : учебник для СПО / А. В. Макшанов, А. Е. Журавлев, Л. Н. Тындыкарь. – 3-е изд., стер – Санкт-Петербург : Лань, 2023. – 188 с.;
6. Волк В. К. Базы данных. Проектирование, программирование, управление и администрирование : учебник для СПО / В. К. Волк. – 2-е изд., стер. – Санкт-Петербург : Лань, 2022. – 340 с.;
7. Васильева М. А. Информационное обеспечение систем управления. Проектирование базы данных с заданиями : учебник для вузов / М. А. Васильева, К. М. Филипченко, Е. П. Балакина. – Санкт-Петербург : Лань, 2023. – 200 с.;
8. Васильева М. А. Система контроля версий. Основы командной разработки : учебное пособие для вузов / М. А. Васильева, К. М. Филипченко. – Санкт-Петербург : Лань, 2022. – 144 с.;
9. Заяц А. М. Введение в гибридные технологии разработки мобильных приложений : учебное пособие для СПО / А. М. Заяц, Н. П. Васильева. – 2-е изд., стер. – Санкт-Петербург : Лань, 2022. – 160 с.;
10. Васильев Н. П. Введение в гибридные технологии разработки мобильных приложений : учебное пособие для вузов / Н. П. Васильев, А. М. Заяц. – 3-е изд., стер. – Санкт-Петербург : Лань, 2022. – 160 с.;
11. Старолетов С. М. Основы тестирования программного обеспечения : учебное пособие для СПО / С. М. Старолетов. – 3-е изд., стер. – Санкт-Петербург : Лань, 2023. – 192 с.;
12. Старолетов С. М. Основы тестирования и верификации программного обеспечения : учебное пособие для вузов / С. М. Старолетов. – 3-е изд., стер. – Санкт-Петербург : Лань, 2023. – 344 с.;
13. ГОСТ Р 58412-2019 «Защита информации»;
14. ГОСТ Р ИСО/МЭК 25010-2015 «Системная и программная инженерия»;
15. ГОСТ 34.602-2020 «Комплекс стандартов на автоматизированные системы».