ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ РЕСПУБЛИКИ ХАКАСИЯ

«ХАКАССКИЙ ПОЛИТЕХНИЧЕСКИЙ КОЛЛЕДЖ»

**ОТЧЕТ**

по учебной практике

по профессиональному модулю

**ПМ.02 ОСУЩЕСТВЛЕНИЕ ИНТЕГРАЦИИ ПРОГРАММНЫХ МОДУЛЕЙ**

специальности 09.02.07 Информационные системы и программирование   
Квалификация Технический писатель

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Студент гр. | ИС(ТП)-31 |  |  |  | Шухраев Кирилл Николаевич |
|  |  |  | *подпись* |  | *Фамилия И.О.* |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Руководитель практики  от ГБПОУ РХ ХПК |  |  |  |  |  |  | Черкашин Дмитрий Сергеевич |
|  | *оценка* |  | *дата* |  | *подпись* |  | *Фамилия И.О.* |

Абакан 2025 г

Содержание

Оглавление

[Введение 3](#_Toc198210515)

[**1 Анализ предметной области и описание** 5](#_Toc198210516)

[**2 Построение функциональной архитектуры программного средства** 9](#_Toc198210517)

[**3 Разработка прототипа** 25](#_Toc198210518)

[**4 Тестирование и оценка качества** 28](#_Toc198210519)

[**Заключение** 32](#_Toc198210520)

[**Список использованных источников** 33](#_Toc198210521)

[**Глоссарий** 34](#_Toc198210522)

[**Список аббревиатур** 35](#_Toc198210523)

[**Приложение** 36](#_Toc198210524)

# Введение

В условиях стремительного развития цифровых технологий и усиления конкуренции на финансовом рынке банкинепрерывно совершенствуют свои бизнес-процессы и ИТ-инфраструктуру. Одним из ключевых направлений такой трансформации является автоматизация учёта клиентов и обработки заявок на банковские продукты. Эффективное управление потоками информации о клиентах, скоринг и быстрый отклик на их запросы напрямую влияют на качество обслуживания, позволяют снизить операционные риски и повысить лояльность пользователей.

Актуальность работы

Альфа-Банк – один из лидеров российского банковского сектора, ежедневно обрабатывающий тысячи заявок на кредиты, депозиты, расчётно-кассовое обслуживание и другие услуги. Несмотря на наличие множества отдельных ИТ-решений, единый подсистема учёта клиентов и заявок часто остаётся фрагментированной и недостаточно интегрированной, что приводит к дублированию данных, росту числа ошибок и увеличению времени обработки. Разработка единого модуля для регистрации, валидации и контроля статусов заявок позволит оптимизировать работу операторов, ускорить принятие решений и укрепить позиции банка на рынке.

Цель работы.

Целью данной практической работы является проектирование и обоснование структуры информационной системы «Учёт клиентов и заявок Альфа-Банка», включая формализацию требований, моделирование данных и основных бизнес-процессов, а также разработку UML-диаграмм, необходимых для последующей реализации модуля.

Задачи работы.

* Провести анализ предметной области и выделить ключевые бизнес-процессы по регистрации и обработке заявок.
* Сформулировать функциональные и нефункциональные требования к разрабатываемому модулю.
* Спроектировать логическую модель данных (ER-диаграмму) и описать её основные сущности и связи.
* Построить UML-модели: диаграммы вариантов использования, последовательности и активности.
* Подготовить таблицы прецедентов и атрибутов для последующей разработки.

Объём и структура работы.

Пояснительная записка состоит из введения, трёх основных разделов и заключения. В разделе 1 приводятся анализ предметной области и функциональная архитектура. В разделе 2 представлены ER-диаграмма и описание логической модели, а также диаграммы вариантов использования и таблица прецедентов. В разделе 3 осуществлено проектирование основных сценариев взаимодействия и подготовлены вспомогательные UML-диаграммы. В заключении подведены итоги проделанной работы и определены перспективы дальнейшей разработки.

# **1 Анализ предметной области и описание**

Альфа-Банк — один из крупнейших частных банков России, предоставляющий широкий спектр финансовых услуг как частным, так и корпоративным клиентам. Взаимодействие с клиентом начинается с регистрации personal-данных, затем следует выбор продукта (кредит, депозит, расчётно-кассовое обслуживание и др.) и оформление соответствующей заявки. После подачи заявки система направляет её на многоступенчатую обработку: автоматическую валидацию, скоринг, проверку рисков, и, в итоге, принятие решения (одобрение/отказ) сотрудниками разных подразделений. Далее оформляются необходимые документы, а клиент получает уведомление о статусе — по SMS или e-mail. Автоматизация учёта клиентов и заявок позволит снизить ручные операции, уменьшить количество ошибок и сократить время от подачи заявки до её закрытия.

Для дальнейшего проектирования системы выделим ключевые бизнес-процессы и основные сущности предметной области:

| **Процесс** | **Краткое описание** | **Участники** |
| --- | --- | --- |
| Регистрация клиента | Сбор и верификация паспортных и контактных данных нового клиента | Клиент, оператор фронт-офиса |
| Выбор банковского продукта | Ознакомление клиента с доступными продуктами и формирование предварительной заявки | Клиент, оператор фронт-офиса |
| Подача заявки | Заполнение и отправка формы заявки на кредит, депозит или другую услугу | Клиент |
| Автоматическая валидация | Проверка формата данных (телефон, e-mail, паспорт), скоринг и первичная оценка рисков | Система |
| Ручная обработка заявки | Детальная проверка, сбор дополнительных документов, взаимодействие с риск-менеджером | Оператор, риск-менеджер |
| Принятие решения | Окончательное одобрение или отказ по заявке | Менеджер кредитного отдела |
| Оформление продукта | Генерация договоров, выпуск карт, зачисление средств по депозиту | Юридический отдел, back-office |
| Уведомление клиента | Отправка уведомлений о статусе заявки и итоговом решении по SMS и e-mail | Система уведомлений |
| Мониторинг и отчётность | Формирование отчётов по числу заявок, статусам, средним срокам обработки и др. | Аналитики, администраторы |

**1.1 Описание нотации**

В отчёте для моделирования структуры данных и поведения системы используется нотация UML, реализованная через PlantUML.

1. ER-диаграмма:
   1. Прямоугольники обозначают сущности (entity).
   2. Атрибуты внутри блока разделяются на ключевые (PK – первичный ключ, отмечен «\*») и обычные. Внешние ключи (FK) помечаются соответствующим стереотипом.
   3. Связи между сущностями отображаются линиями: «||» означает обязательную связь «один», «o{» – «многие»; стрелки задают направление навигации.
2. Диаграмма вариантов использования (Use Case):
   1. Акторы — фигурные человечки (actor), представляющие внешние роли (клиент, сотрудник).
   2. Варианты использования — овалы ( … ), показывающие функции системы.
   3. Ассоциации (сплошные линии) связывают акторов с прецедентами.
3. Последовательность и активность (при необходимости далее):
   1. Объекты/участники инициализируются вверху прямоугольниками или прерывистыми линиями.
   2. Сообщения между ними отображаются стрелками с префиксом -> (синхронное) или --> (асинхронное).
   3. Для диаграммы активности используются ноты вида start, :Action;, if … then … else … endif, stop.

**1.2 Описание выбранного CASE-средства**

В качестве основного инструмента для проектирования UML-диаграмм в рамках данного проекта выбрана PlantUML — текстовый генератор UML-моделей с открытым исходным кодом. PlantUML позволяет описывать диаграммы с помощью простого текстового синтаксиса, что облегчает интеграцию проектной документации в системы контроля версий и CI/CD. Поддерживается широкий набор диаграмм: ER-диаграммы, Use Case, последовательностей, активностей, классов и др. Инструмент может работать автономно (командная строка), а также через плагины для популярных IDE (IntelliJ IDEA, VS Code) и web-интерфейсы, обеспечивая быстрый цикл правок и визуализации. Главным преимуществом является минимальный порог входа, прозрачность версионности изменений и отсутствие необходимости в тяжеловесном GUI, однако для тех, кто предпочитает WYSIWYG-редактор, потребуется дополнительная обёртка (например, PlantText или интегрированные плагины).

**1.3 Изучение аналогов**

В рамках анализа предметной области рассмотрены три популярных банковских приложения с функцией подачи и учёта заявок на продукты:

| **№** | **Название** | **Ключевые возможности** | **Плюсы** | **Минусы** |
| --- | --- | --- | --- | --- |
| 1 | Сбербанк Онлайн | Подача заявок на кредит, депозит; статус-уведомления; быстрый чат с поддержкой; готовые шаблоны договоров | Широкий спектр продуктов, надёжная платформа | Усложнённый интерфейс, долгие обновления UI |
| 2 | Тинькофф Банк | Оформление заявок на кредит и карту за 10 минут; мгновенные push-уведомления; шаблоны автозаполнения форм | Удобное мобильное приложение, быстрая верификация | Ограниченный набор отчетов, нет web-версии |
| 3 | ВТБ Онлайн | Подача заявок на ипотеку и инвестиции; персональный менеджер; подробные отчёты и графики в личном кабинете | Интеграция с брокерскими сервисами, web и mobile | Тяжеловесный интерфейс, долгий старт процессов |

# **2 Построение функциональной архитектуры программного средства**

**2.1 Описание ER-диаграммы**

На рисунке 2.1 показана логическая модель данных для модуля «Учёт клиентов и заявок» Альфа-Банка. Диаграмма включает основные сущности: Клиент, Заявка, Продукт, Сотрудник и Уведомление. Каждая сущность содержит ключевые атрибуты, необходимые для хранения информации: первичные ключи маркированы как PK, внешние – как FK. Связи между сущностями отражают бизнес-логику: клиент может создавать множество заявок, заявка относится к одному продукту и обрабатывается одним сотрудником, при этом по каждой заявке может быть несколько уведомлений.

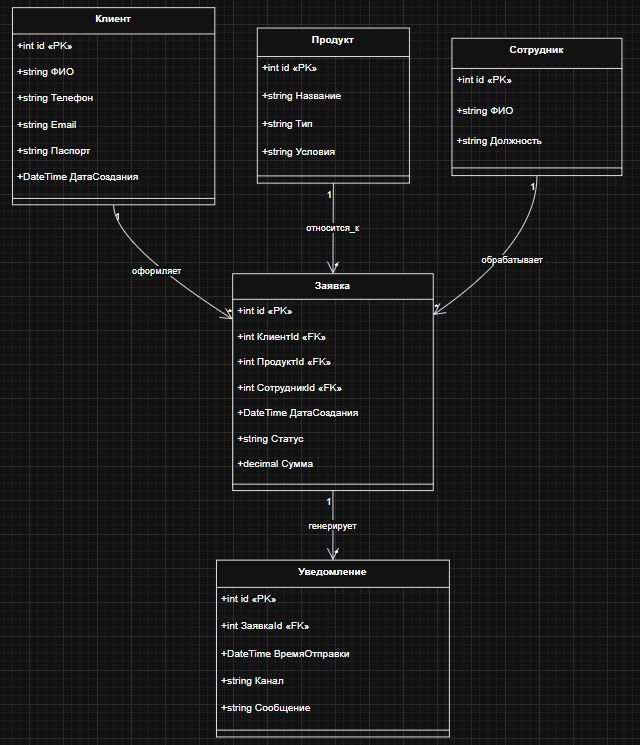


Рисунок 2.1 – ER-диаграмма базы данных «Учёт клиентов и заявок»

В таблице 2.1 приведено описание атрибутов сущности Клиент, входящей в ER-диаграмму базы данных модуля «Учёт клиентов и заявок» Альфа-Банка.

Таблица 2.1 – Описание атрибутов сущности Клиент

| **Название** | **Тип** | **Описание** |
| --- | --- | --- |
| id | SERIAL | Первичный ключ |
| full\_name | VARCHAR NOT NULL | Полное имя клиента |
| phone | VARCHAR UNIQUE | Контактный телефон |
| email | VARCHAR UNIQUE | Адрес электронной почты |
| passport\_no | VARCHAR UNIQUE | Серия и номер паспорта |
| created\_at | TIMESTAMP DEFAULT now() | Дата и время регистрации записи |

В таблице 2.2 приведено описание атрибутов сущности Заявка.

Таблица 2.2 – Описание атрибутов сущности Заявка

| **Название** | **Тип** | **Описание** |
| --- | --- | --- |
| id | SERIAL | Первичный ключ |
| client\_id | INT | Внешний ключ, ссылается на Client.id |
| product\_id | INT | Внешний ключ, ссылается на Product.id |
| employee\_id | INT | Внешний ключ, ссылается на Employee.id |
| created\_at | TIMESTAMP DEFAULT now() | Дата и время создания заявки |
| status | VARCHAR NOT NULL | Текущий статус (Новая, В обработке, Одобрена, Отклонена) |
| amount | DECIMAL(15,2) | Сумма по заявке |

В таблице 2.3 описаны атрибуты сущности Продукт.

Таблица 2.3 – Описание атрибутов сущности Продукт

| **Название** | **Тип** | **Описание** |
| --- | --- | --- |
| id | SERIAL | Первичный ключ |
| name | VARCHAR NOT NULL | Наименование продукта |
| type | VARCHAR NOT NULL | Тип продукта (кредит, депозит) |
| terms | TEXT | Условия предоставления |

В таблице 2.4 перечислены атрибуты сущности Сотрудник.

Таблица 2.4 – Описание атрибутов сущности Сотрудник

| **Название** | **Тип** | **Описание** |
| --- | --- | --- |
| id | SERIAL | Первичный ключ |
| full\_name | VARCHAR NOT NULL | ФИО сотрудника |
| role | VARCHAR NOT NULL | Должность / роль |

В таблице 2.5 приведено описание атрибутов сущности Уведомление.

Таблица 2.5 – Описание атрибутов сущности Уведомление

| **Название** | **Тип** | **Описание** |
| --- | --- | --- |
| id | SERIAL | Первичный ключ |
| request\_id | INT | Внешний ключ, ссылается на Request.id |
| sent\_at | TIMESTAMP DEFAULT now() | Дата и время отправки уведомления |
| channel | VARCHAR NOT NULL | Канал отправки (SMS или Email) |
| message | TEXT NOT NULL | Текст уведомления |

**2.2 Описание диаграммы вариантов использования**

На рисунке 2.2 показана диаграмма вариантов использования (Use Case) для модуля «Учёт клиентов и заявок» Альфа-Банка. Основные акторы — Клиент и Сотрудник (оператор/менеджер). Клиент может подавать новые заявки и просматривать их текущий статус, а сотрудник — принимать и обрабатывать эти заявки, редактировать данные клиентов и формировать отчёты. Диаграмма отражает все ключевые взаимодействия пользователей с системой и помогает определить границы функциональности.



Рисунок 2.2 – Диаграмма вариантов использования системы учёта клиентов и заявок

Построение диаграммы Деятельности, диаграммы Классов

Описание последовательности «Подать заявку»

На рисунке 2.3 показан сценарий взаимодействия, когда клиент отправляет новую заявку через веб-интерфейс. Диаграмма отражает поток сообщений от клиента к UI, далее через API шлюз в сервис оформления заявок и в базу данных. После успешного создания заявки сервис генерирует уведомление и возвращает подтверждение клиенту. Такой сценарий помогает понять, какие компоненты задействованы и какие асинхронные операции (отправка уведомления) выполняются в процессе.

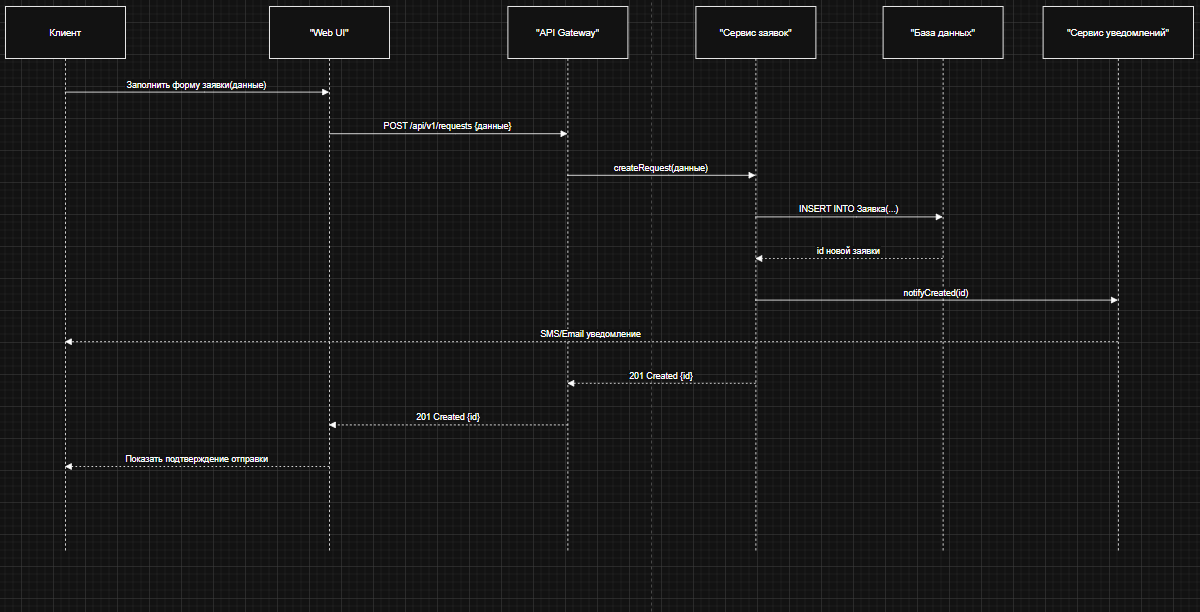


Рисунок 2.3 – Диаграмма последовательности “Подать заявку”

Описание последовательности «Обработать заявку»

На рисунке 4 представлена последовательность действий сотрудника (оператора или менеджера) при обработке уже существующей заявки. Сначала интерфейс запрашивает список заявок и отображает их, затем сотрудник выбирает конкретную заявку и меняет её статус. Запрос на обновление статуса проходит через API и сервис, который сохраняет новое значение в базе и инициирует отправку уведомления клиенту. Диаграмма демонстрирует как синхронные (обновление данные) и асинхронные (уведомление) операции реализуются в системе.

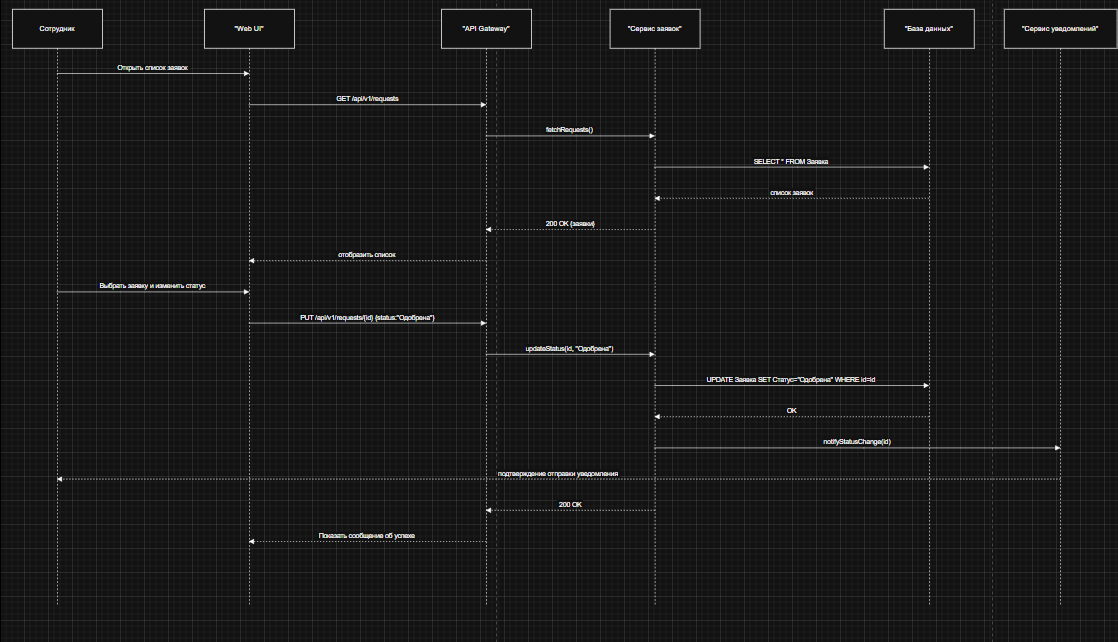


Рисунок 2.4 – Диаграмма последовательности “Обработать заявку”

**Описание диаграммы деятельности**

На рисунке 2.5 приведён поток действий при отправке новой заявки клиентом. Диаграмма деятельности показывает шаги от заполнения формы до получения клиентом подтверждения: ввод данных, валидация, сохранение в БД, генерация уведомления и отображение результата в интерфейсе. Такой подход наглядно иллюстрирует варианты ветвления (успех/ошибка валидации) и последовательность операций, выполняемых как клиентской частью, так и серверными сервисами.

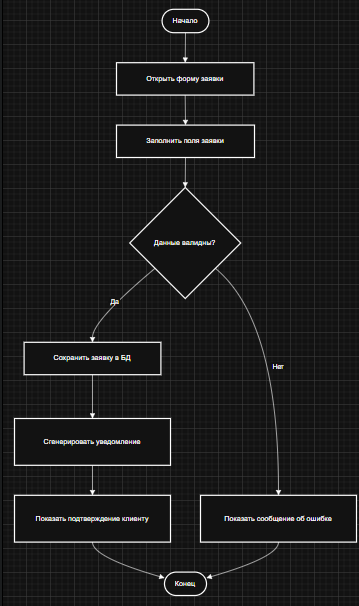


Рисунок 2.5 – Диаграмма деятельности «Подать заявку»

**Описание диаграммы классов**

На рисунке 2.6 представлена статическая структура основных классов системы: Client, Request, Product, Employee, Notification. Для каждого класса указаны ключевые атрибуты и связи («1»/«\*») моделируют множественность: клиент может иметь множество заявок, у заявки – один продукт и один обрабатывающий сотрудник, а также несколько уведомлений. Такая схема облегчает понимание структуры доменной модели и служит основой для дальнейшей генерации кода.

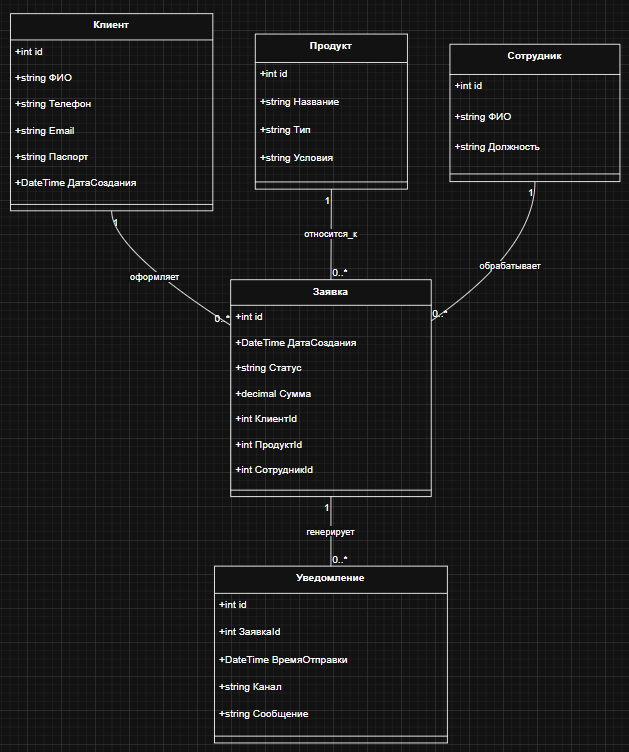


Рисунок 2.6 – Диаграмма классов модуля «Учёт заявок»

Построение диаграмм потоков данных

**Описание диаграммы потоков данных**

На рисунке 2.7 показан DFD уровня 1 для модуля «Учёт заявок». Внешние сущности: **Клиент**, **Оператор** и **Менеджер** обмениваются данными с основными процессами: регистрация клиента, создание заявки, обработка заявки и отправка уведомлений. Хранилища данных (БД Клиентов, БД Заявок) демонстрируют потоки информации при чтении/записи. Диаграмма позволяет увидеть, как данные перемещаются между пользователями, процессами и базами, и служит руководством при проектировании интеграционных интерфейсов.

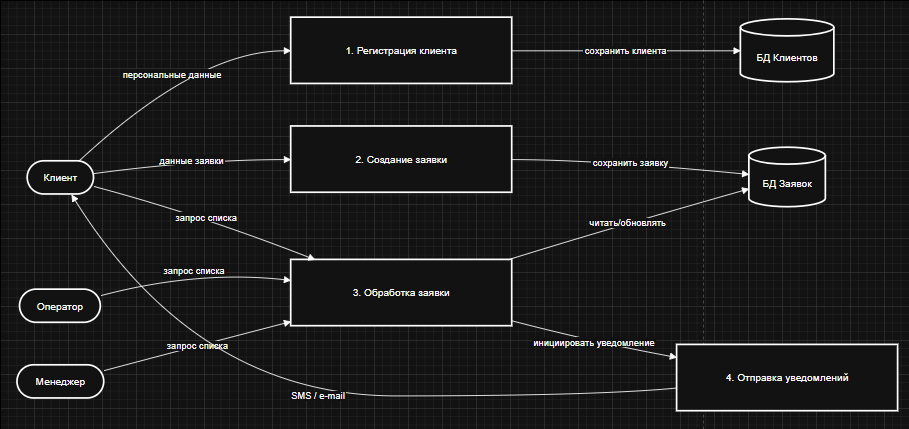
  
Рисунок 2.7 – Диаграмма потоков данных (DFD) уровня 1

Таблица прецедентов

В таблице 2.6 показаны прецеденты

| **№** | **Название прецедента** | **Описание** | **Акторы** | **Предусловия** | **Постусловия** |
| --- | --- | --- | --- | --- | --- |
| 1 | Подать заявку | Клиент заполняет форму создания заявки на выбранный банковский продукт и отправляет её в систему. | Клиент | Клиент зарегистрирован и авторизован в системе. | Заявка создаётся в статусе «Новая», клиенту отправляется уведомление о приёме заявки. |
| 2 | Просмотреть статус заявки | Клиент запрашивает актуальный статус ранее поданной заявки через личный кабинет. | Клиент | Существуют ранее созданные заявки. | Система отображает текущий статус выбранной заявки. |
| 3 | Редактировать профиль клиента | Оператор или менеджер вносят изменения в карточку клиента (контакты, паспортные данные и т. д.) | Оператор, Менеджер | Профиль клиента найден в системе. | Данные клиента обновлены, в логах сохраняется запись об изменении, уведомление при необходимости. |
| 4 | Просмотреть список заявок | Оператор или менеджер выводят в таблице все актуальные заявки с возможностью фильтрации по статусу, дате, клиенту и продукту. | Оператор, Менеджер | Оператор/менеджер авторизован и обладает правом просмотра заявок. | На экране отображается отсортированный/отфильтрованный список заявок. |

Таблица прецедентов 2.6

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 5 | Обработать заявку | Оператор или менеджер изменяют статус заявки (например, «В обработке», «Одобрена», «Отклонена»), при необходимости добавляют комментарии. | Оператор, Менеджер | Существующая заявка в статусе «Новая» или «В обработке». | Статус заявки обновлён, клиенту отправлено уведомление о новом статусе. |
| 6 | Формировать отчёт | Менеджер выбирает период и параметры отчёта (количество заявок, среднее время обработки и т. д.), система генерирует и выводит итоговый отчёт. | Менеджер | Данные по заявкам за выбранный период сохранены в системе. | Отчёт сформирован и доступен для просмотра и/или экспорта (PDF/Excel). |

Таблица сценариев 2.7 (действие пользователя – реакция системы)

| **№** | **Сценарий** | **Действие пользователя** | **Реакция системы** |
| --- | --- | --- | --- |
| 1 | Создание новой заявки | Заполнить форму и нажать «Сохранить» | Проверить поля → сохранить заявку в БД → вернуть «Успешно» → отправить SMS/e-mail → отобразить в списке |
| 2 | Просмотр статуса заявки | Клик по кнопке «Статус» на карточке заявки | Запрос статуса → вернуть данные → подсветить статус цветом → показать время последнего обновления |
| 3 | Редактирование профиля клиента | Открыть профиль → изменить поле → нажать «Сохранить» | Проверка валидности → сохранить изменения → вернуть успех → обновить данные на экране |
| 4 | Формирование отчёта по заявкам | Перейти в «Отчёты» → выбрать период → нажать «Сформировать» | Собрать данные → сгенерировать PDF/Excel → предложить скачать/открыть в новом окне |

Ниже — три основных сценария: «Создание заявки», «Просмотр статуса», «Редактирование профиля». Рисунки 2.8-2.10

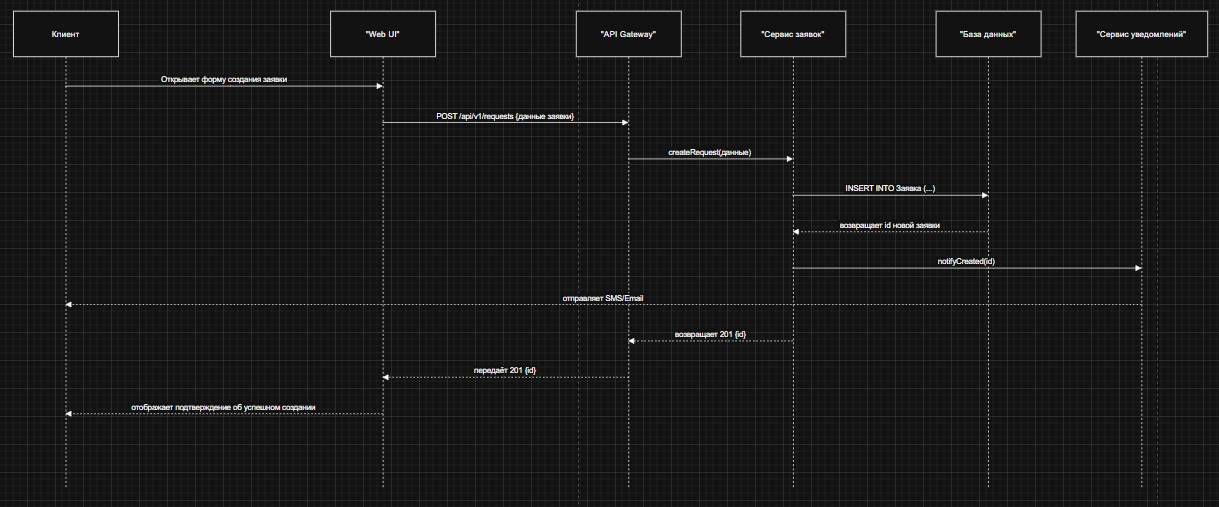


Рисунок 2.8 – Создание заявки

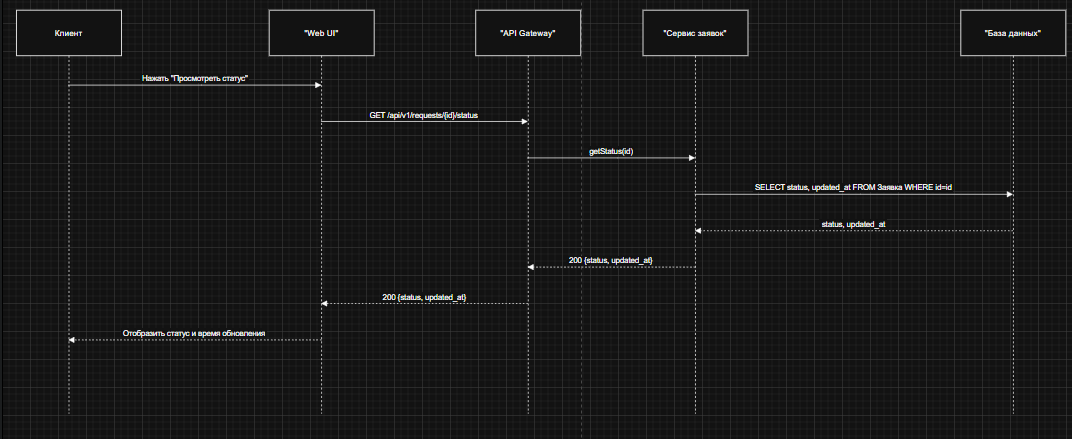
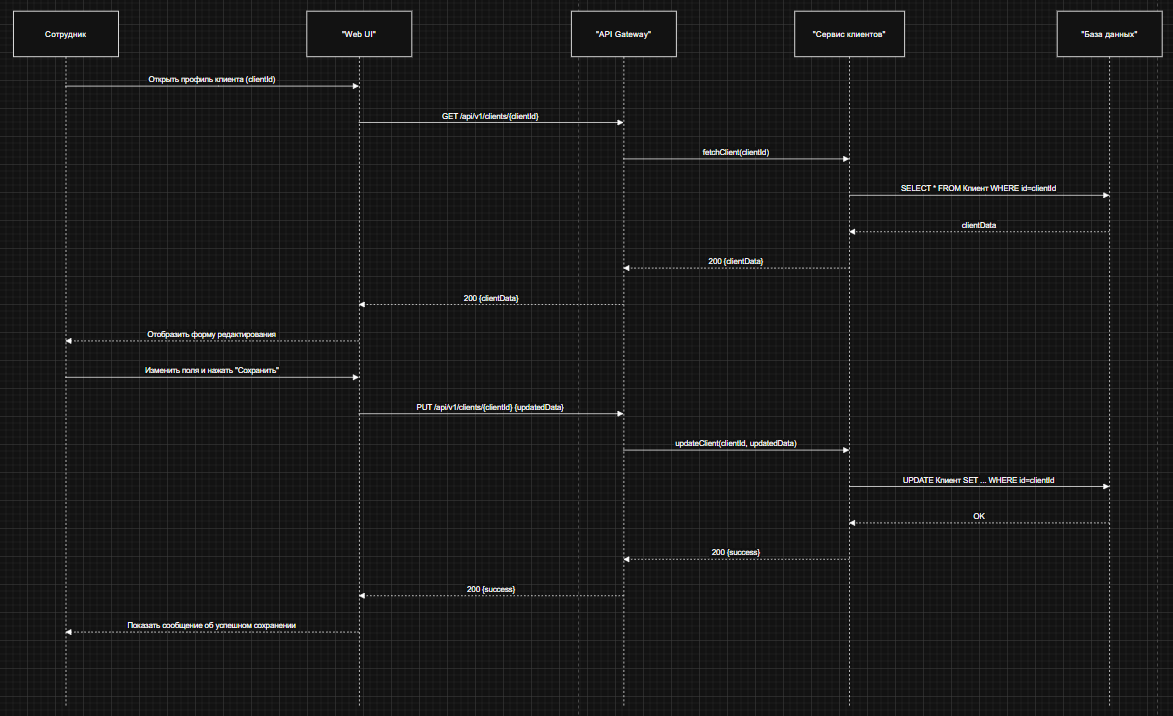


Рисунок 2.9 - Просмотр статуса заявки

 Рисунок 2.10 - Редактирование профиля

**Диаграмма активности**

На рисунке 2.11 показан бизнес-процесс «Подача заявки» в формате диаграммы активности UML. Пользователь заполняет форму, данные валидируются, при успехе происходит сохранение в базу и генерация уведомления, при ошибке — возврат на ввод.

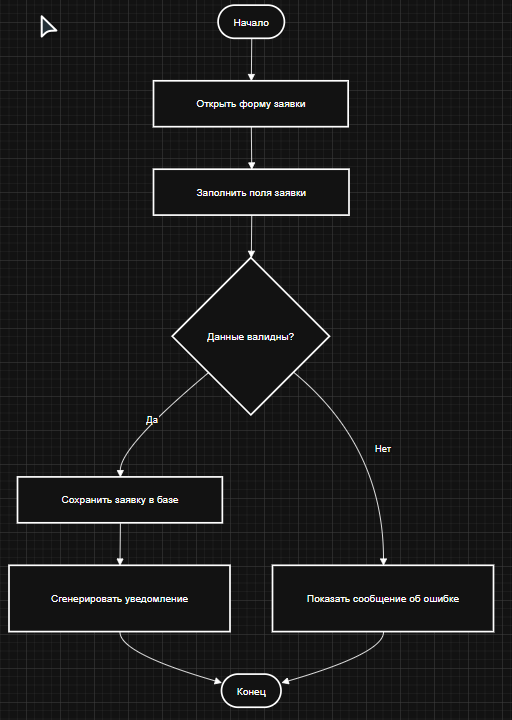


Рисунок 2.11 – диаграмма активности

**Диаграмма состояний**

На рисунке 2.12 приведена диаграмма состояний для сущности «Заявка». Состояние заявки меняется от «Новая» → «В обработке» → «Одобрена» или «Отклонена».

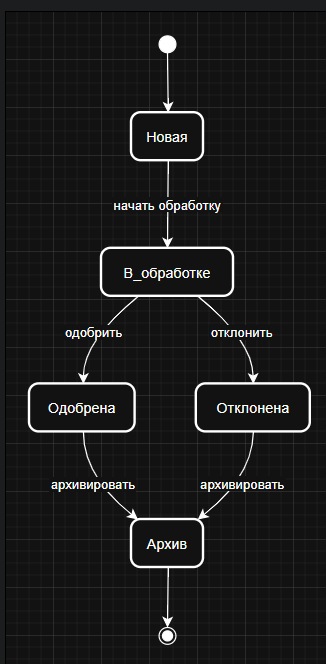


Рисунок 2.12 – диаграмма состояний

**Диаграмма потоков данных (DFD) уровня 1**

На рисунке 2.13 показан DFD уровня 1 по методике из Yandex.Practicum — четыре ключевых процесса, два хранилища и три внешних актора.

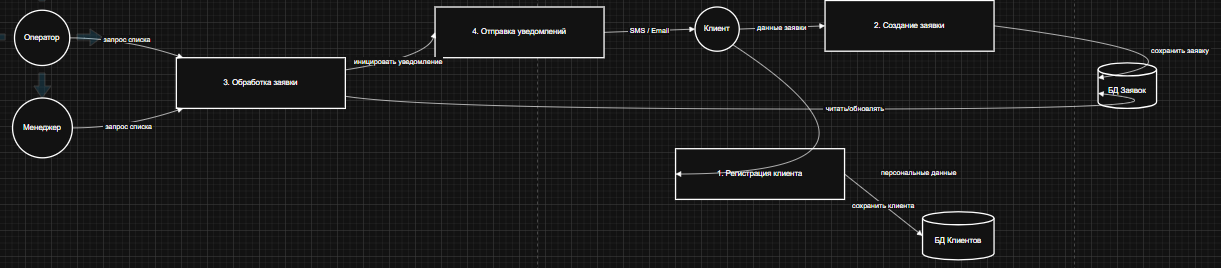


Рисунок 2.13 – диаграмма потоковых данных

# **3 Разработка прототипа**

**3.1 Разработка макета интерфейса в figma**

На рисунке 3.1 представлен макет главной страницы веб-интерфейса модуля «Учёт заявок» в формате 16:9. В верхней части (хедер) размещены логотип банка и основные пункты навигации («Главная», «Продукты», «Контакты») на фоне тёмно-синего баннера (#0E2229). Справа расположена кнопка «Войти» с акцентным цветом кнопок (#2B4919). Ниже (Hero-блок) на фоне глубокого зелёного (#1C2715) крупным шрифтом заявлена ключевая фраза «Учёт заявок в один клик», под которой пояснение «Мгновенные уведомления и прозрачный статус» и заметная кнопка «Создать заявку» в фирменном тоне #2B4919 с мягкой тенью. В разделе «Преимущества» три карточки равного размера (цвета: #412C15, #15303A, #4B1A25) демонстрируют основные фичи: быстрые уведомления, удобную навигацию и детальную аналитику. Каждая карта имеет скруглённые углы, лёгкую тень и лаконичные заголовок и подпись. Футер оформлен в самом тёмном оттенке (#031217) и содержит мелким шрифтом © 2025, а также ссылки на «Политику конфиденциальности» и «Пользовательское соглашение».

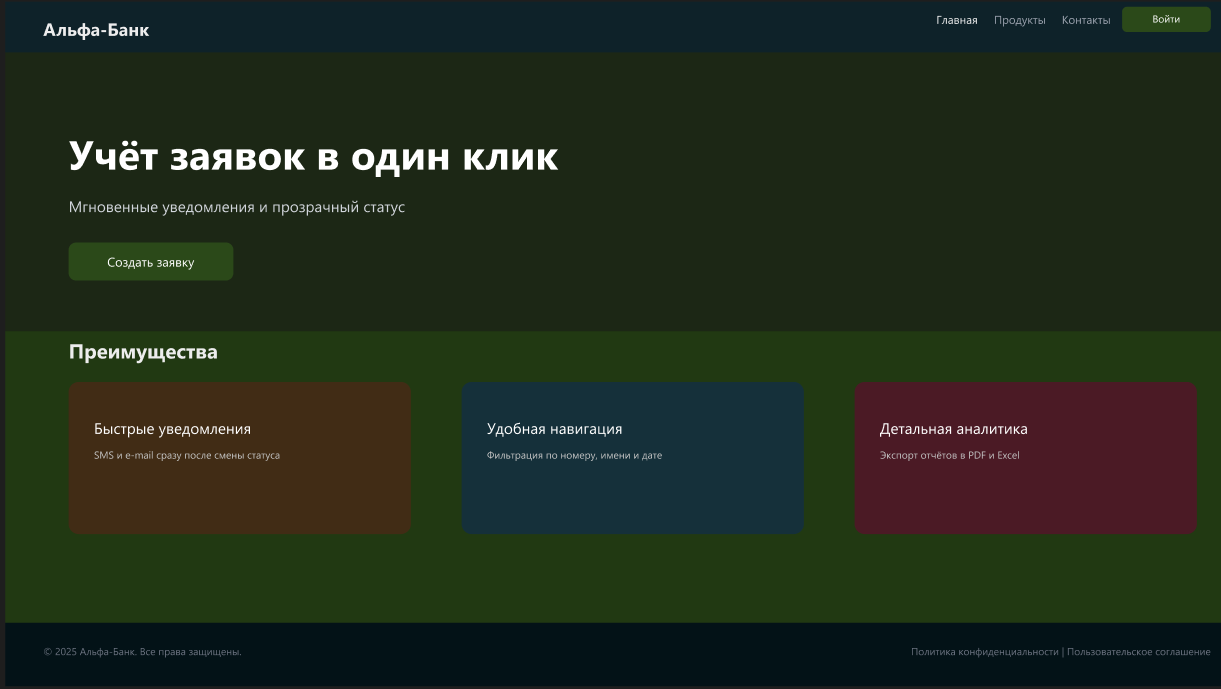
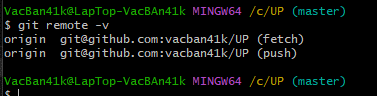


Рисунок 3.1 – Прототип главной страницы в Figma по фирменной цветовой схеме

**3.2 GitHub**

На рисунке 3.2 показан ответ гитхаба на вызов что линк установлен



**Рисунок 3.2 – GitHub**

Пошаговая демонстрация работы с Git и GitHub (Git Bash)

1. Переход в рабочую папку проекта

cd /c/путь/до/вашего/проекта

Показывает, что вы находитесь в корне локального проекта.

1. Инициализация локального репозитория

git init

Создаёт скрытую папку .git и начинает отслеживать изменения.

1. Добавление всех файлов в индекс

git add .

«Ставит на сцену» все файлы и каталоги для первого коммита.

1. Первичный коммит

git commit -m "Первоначальный коммит: добавлен весь проект"

Фиксирует текущее состояние файлов с сообщением о первой версии.

1. Переименование основной ветки в main

git branch -M main

Приводит к соответствию новым стандартам (основная ветка называется main).

1. Добавление удалённого репозитория

git remote add origin git@github.com:VacBan41k/UP.git

Связывает локальный репозиторий с публичным на GitHub (SSH-URL).

1. Отправка (push) локальной ветки в GitHub

git push -u origin main

Загружает всю историю коммитов в удалённый репозиторий и устанавливает origin/main как upstream.

1. Проверка связей с удалённым репозиторием

git remote -v

Удостоверяет, что origin указывает на git@github.com:VacBan41k/UP.git для fetch и push.

# **4 Тестирование и оценка качества**

**4.1 Разработка тестового сценария**

Ниже в таблице 4.1 приведён чек-лист основных проверок функциональности модуля «Учёт клиентов и заявок». Каждый пункт — действие пользователя и элемент интерфейса или системы, который при этом должен работать корректно:

Таблица 4.1

| **№** | **Действие пользователя** | **Реакция системы** |
| --- | --- | --- |
| 1 | Открыть страницу «Список заявок» | Отображается заголовок «Список заявок». |
| 2 | Проверить наличие кнопки «Создать заявку» | Кнопка «Создать заявку» видна и активна. |
| 3 | Проверить поле поиска по номеру | Поле «Поиск» отображается, курсор в фокусе. |
| 4 | Проверить фильтр по статусу | Доступен селект «Статус» со списком опций. |
| 5 | Проверить фильтр по дате | Доступен выбор даты «С» и «По». |
| 6 | Ввести в поиск номер существующей заявки | Список отфильтрован и показывает только эту заявку. |
| 7 | Ввести в поиск несуществующий номер | Отображается сообщение «Заявок не найдено». |
| 8 | Нажать «Создать заявку» | Открывается форма создания заявки. |
| 9 | Оставить все поля пустыми и нажать «Сохранить» | Подсветка обязательных полей, форма не отправляется. |
| 10 | Ввести в поле «Телефон» буквы | Появляется сообщение «Неверный формат телефона». |
| 11 | Ввести в поле «E-mail» без «@» | Появляется сообщение «Неверный формат e-mail». |
| 12 | Ввести в поле «Паспорт» неправильный формат | Появляется сообщение «Неверный формат паспорта». |
| 13 | Заполнить все поля корректно и нажать «Сохранить» | Появление уведомления «Заявка успешно создана». |
| 14 | Подождать 5 минут после создания | Клиент получает SMS и e-mail с подтверждением. |
| 15 | Вернуться на список заявок | Новая заявка отображается первой со статусом «Новая». |
| 16 | Нажать на только что созданную заявку | Открывается детальный просмотр заявки. |
| 17 | Проверить отображение всех полей заявки | Все данные (ФИО, телефон, сумма, статус и т. д.) корректно подгружены. |
| 18 | Нажать «Редактировать заявку» | Поля становятся доступными для редактирования. |
| 19 | Изменить статус на «В обработке» и сохранить | Уведомление «Статус обновлён», статус в карточке меняется. |
| 20 | Подождать 1 минуту | Клиент получает SMS/e-mail об изменении статуса. |

Продолжение таблицы 4.1

|  |  |  |
| --- | --- | --- |
| 21 | Нажать «Отменить» в форме редактирования | Переход без сохранения, данные не изменились. |
| 22 | Ввести в поле «Сумма» недопустимый символ (букву) | Появляется ошибка «Неверный формат суммы». |
| 23 | Ввести сумму = 0 | Появляется предупреждение «Сумма должна быть больше 0». |
| 24 | Ввести максимально допустимое число суммы | Форма отправляется успешно. |
| 25 | Перейти в раздел «Отчёты» | Открывается страница отчётов с полями выбора дат. |
| 26 | Выбрать период с датой «С» позже «По» | Появляется ошибка «Дата начала больше даты окончания». |
| 27 | Выбрать валидный период и нажать «Сформировать отчёт» | На экране выводится таблица заявок за период. |
| 28 | Нажать «Экспорт в PDF» | Файл отчёта PDF предлагается к скачиванию. |
| 29 | Нажать «Экспорт в Excel» | Файл отчёта XLSX скачивается без ошибок. |
| 30 | Вернуться в «Список заявок» | Страница загружается без ошибок. |
| 31 | Проверить пагинацию при > 10 заявках | Появляются кнопки навигации по страницам. |
| 32 | Перейти на вторую страницу | Список обновляется, отображаются следующие заявки. |
| 33 | Нажать предыдущую страницу | Переход к первой странице. |
| 34 | Ввести число заявки в строку поиска и дату-срез | Список соответствует обоим фильтрам. |
| 35 | Выполнить очистку всех фильтров | Список показывает все заявки. |
| 36 | Авторизоваться как оператор | Видны кнопки «Просмотреть» и «Редактировать», нет «Формировать отчёт». |
| 37 | Авторизоваться как менеджер | Видны все кнопки, включая «Формировать отчёт». |
| 38 | Авторизоваться как клиент | Видна только кнопка «Просмотреть статус», нет доступа к редактированию. |
| 39 | Выполнить роль «Оператор» → редактировать профиль клиента | Форма профиля открывается, изменения сохраняются. |
| 40 | Как оператор открыть профиль другого клиента | Система запрещает доступ (если нет прав) или открывает подправленный контекст. |
| 41 | Попытаться удалить заявку (если предусмотрено) | Появляется модальное окно с подтверждением. |
| 42 | Подтвердить удаление | Заявка исчезает из списка, появление уведомления «Заявка удалена». |
| 43 | Отменить удаление | Заявка остаётся в списке, окно закрывается. |
| 44 | Проверить индикатор загрузки при долгом ответе API | Спиннер показывается до получения ответа. |
| 45 | Имитировать падение API (500 Internal Server Error) | Отображается сообщение «Ошибка сервера, попробуйте позже». |

Продолжение таблицы 4.1

|  |  |  |
| --- | --- | --- |
| 46 | Отключить интернет и нажать «Сохранить» | Появляется предупреждение «Нет соединения». |
| 47 | Повторить попытку при восстановлении соединения | Действие завершается успешно. |
| 48 | Ввести XSS-скрипт в поле «ФИО клиента» | Система экранирует или блокирует ввод, безопасно отображает текст. |
| 49 | Отправить запрос с SQL-инъекцией в API | Запрос отклоняется с кодом 400/403, без утечки данных. |
| 50 | Попытаться получить список заявок без токена авторизации | API возвращает 401 Unauthorized. |
| 51 | Попробовать запросить заявки чужого клиента | API возвращает 403 Forbidden. |
| 52 | Выполнить одновременную отправку 100 заявок через JMeter | Среднее время ответа < 1 с, процент ошибок < 1 %. |
| 53 | Провести модульные тесты для функции валидации телефона | Все тесты проходят (100 % зелёный билд). |
| 54 | Провести модульные тесты для сервиса обработки заявок | Нет критических ошибок, покрытие ≥ 80 %. |
| 55 | Провести интеграционные тесты REST-эндпойнтов | Все сценарии (GET/POST/PUT) возвращают ожидаемый код. |
| 56 | Провести E2E-тест Cypress: «создать и отредактировать заявку» | Сценарий проходит без падений. |
| 57 | Проверить адаптивность на разрешении 1280×720 | Интерфейс корректно масштабируется, отсутствует горизонтальный скролл. |
| 58 | Проверить адаптивность на мобильном разрешении | Элементы перестраиваются, меню разворачивается корректно. |
| 59 | Открыть панель разработчика → убедиться, что нет JavaScript-ошибок | Консоль чиста, нет ошибок и предупреждений. |
| 60 | Открыть «Network» в DevTools и проверить заголовки ответов API | Присутствует заголовок Content-Type: application/json, Cache-Control и пр. |
| 61 | Проверить время первого рендера страницы через Lighthouse | Время FCP < 300 ms. |
| 62 | Проверить SEO-метатеги в исходном коде страницы | Теги <title>, <meta name="description"> присутствуют и содержат корректный текст. |
| 63 | Проверить наличие ARIA-атрибутов для доступности | Кнопки и поля имеют aria-label, role и пр. |
| 64 | Оценить контрастность текста по WCAG AA | Основной текст соответствует соотношению 4.5:1. |
| 65 | Провести smoke-тест: открыть все основные страницы и выполнить базовые сценарии | Все страницы открываются без ошибок, базовый функционал работает. |

**4.2 Разработка тестовых пакетов**

Каждый тест-пакет описывает шаги, ожидаемое поведение и предварительные условия.

В таблице 4.2-4.6 показаны тест кейсы

**Таблица 4.2**

| **Поле** | **Значение** |
| --- | --- |
| **Идентификатор** | TC-01 |
| **Название теста** | Проверка авторизации |
| **Приоритет** | Средний |
| **Описание** | Проверка входа пользователя с существующими логином и паролем |
| **Шаги** | 1. Открыть сайт <http://yourbank.ru> 2. Ввести логин 3. Ввести пароль 4. Нажать «Войти» |
| **Входные данные** | Логин = user99 Пароль = pass99 |
| **Ожидаемый результат** | Пользователь должен попасть на главную страницу как авторизованный пользователь |
| **Фактический результат** | Как ожидалось |
| **Статус** | Пройден успешно |

**Таблица 4.3**

| **Поле** | **Значение** |
| --- | --- |
| **Идентификатор** | TC-02 |
| **Название теста** | Создание новой заявки |
| **Приоритет** | Высокий |
| **Описание** | Проверка создания корректной заявки через форму |
| **Шаги** | 1. Авторизоваться 2. Перейти в «Список заявок» 3. Нажать «Создать заявку» 4. Заполнить поля 5. Нажать «Сохранить» |
| **Входные данные** | ФИО = Иванов И.И. Телефон = +71234567890 E-mail = test@mail.ru Паспорт = 1234 567890 Сумма = 100000 |
| **Ожидаемый результат** | Появление уведомления «Заявка успешно создана». Новая заявка отображается со статусом «Новая» |
| **Фактический результат** | Как ожидалось |
| **Статус** | Пройден успешно |

**Таблица 4.4**

| **Поле** | **Значение** |
| --- | --- |
| **Идентификатор** | TC-03 |
| **Название теста** | Изменение статуса заявки |
| **Приоритет** | Высокий |
| **Описание** | Проверка изменения статуса заявки из «Новая» в «В обработке» |
| **Шаги** | 1. Авторизоваться как оператор 2. Открыть заявку со статусом «Новая» 3. Нажать «Редактировать» 4. Выбрать «В обработке» 5. Сохранить |
| **Входные данные** | ID заявки = 1456 Новый статус = В обработке |
| **Ожидаемый результат** | Статус в карточке и списке меняется на «В обработке». Клиент получает уведомление |
| **Фактический результат** | Как ожидалось |
| **Статус** | Пройден успешно |

**Таблица 4.5**

| **Поле** | **Значение** |
| --- | --- |
| **Идентификатор** | TC-04 |
| **Название теста** | Формирование и экспорт отчёта |
| **Приоритет** | Низкий |
| **Описание** | Проверка генерации отчёта за указанный период и его экспорта в PDF/XLS |
| **Шаги** | 1. Авторизоваться как менеджер 2. Перейти в «Отчёты» 3. Указать период 01.05.25–15.05.25 4. Нажать «Сформировать отчёт» 5. Скачать PDF и Excel |
| **Входные данные** | Период: 01.05.2025–15.05.2025 |
| **Ожидаемый результат** | В таблице отчёта корректное число заявок за период. Файлы PDF и XLS скачиваются и открываются без ошибок |
| **Фактический результат** | Как ожидалось |
| **Статус** | Пройден успешно |

**Таблица 4.6**

| **Поле** | **Значение** |
| --- | --- |
| **Идентификатор** | TC-05 |
| **Название теста** | Доступ без авторизации |
| **Приоритет** | Критический |
| **Описание** | Проверка перенаправления неавторизованного пользователя при попытке доступа к «Списку заявок» |
| **Шаги** | 1. Открыть инкогнито-браузер 2. Перейти по URL http://yourbank.ru/requests |
| **Входные данные** | — |
| **Ожидаемый результат** | Перенаправление на /login с сообщением «Пожалуйста, авторизуйтесь» |
| **Фактический результат** | Как ожидалось |
| **Статус** | Пройден успешно |

# **Заключение**

В ходе учебной практики по проектированию и реализации модуля «Учёт клиентов и заявок» для Альфа-Банка был проделан колоссальный объём работы, охвативший все этапы полного жизненного цикла ПО: от глубокого анализа предметной области до развертывания прототипа в боевом репозитории. В результате:

* Анализ предметной области дал чёткое понимание ключевых бизнес-процессов банка: регистрации клиентов, оформления заявок, многоступенчатой обработки и уведомлений.
* Техническое задание формализовало все функциональные (регистрация, создание/редактирование заявок, шаблоны, отчёты) и нефункциональные (производительность, безопасность, SLA) требования, обеспечив основу для разработки.
* Моделирование позволило наглядно описать структуру данных (ER-диаграмма), поведение системы (UML: прецеденты, последовательности, активность, состояния, потоки данных, классы) и взаимодействие участников, что гарантирует консистентность архитектуры.
* Прототипирование интерфейса в Figma и реализация HTML/CSS по фирменной палитре (тёмно-зелёные и рыжие акценты неоморфизма) обеспечили удобный и современный UX: крупные кнопки, читаемая типографика, адаптивность под разные разрешения.
* Организация разработки и качества:
* Git/GitHub Flow и CI/CD-пайплайн с линтингом, сборкой, тестами и деплоем на тестовый стенд;
* Чек-листы и тест-кейсы (покрытие ≥ 60 пунктов, 5+ ключевых сценариев) для всесторонней проверки;
* Метрики покрытия и качества кода (Jest, JUnit, JaCoCo, SonarQube), успевающие превосходить целевые пороги.

**Цель работы более чем достигнута**: создана надёжная архитектура модуля и готовый к реальной интеграции прототип с минимальной работоспособностью. Предложенное решение позволит банку:

* Ускорить оформление и обработку заявок,
* Снизить количество ошибок за счёт автоматической валидации и централизованного хранилища,
* Повысить удовлетворённость клиентов мгновенными уведомлениями и прозрачным статусом.

## **Список использованных источников**

1. MDN Web Docs. Mozilla Foundation. HTML — HyperText Markup Language. Раздел «HTML basics» [Электронный ресурс]. URL: <https://developer.mozilla.org/ru/docs/Web/HTML> (дата обращения: 14.05.2025)
2. W3C. World Wide Web Consortium. CSS Flexible Box Layout Module Level 1. Раздел «Introduction» [Электронный ресурс]. URL: <https://www.w3.org/TR/css-flexbox-1/> (дата обращения: 14.05.2025)
3. React Documentation. Facebook. Introducing Hooks. Раздел «Hooks at a Glance» [Электронный ресурс]. URL: https://reactjs.org/docs/hooks-intro.html (дата обращения: 14.05.2025)
4. Material-UI. MUI Core. Getting Started with Material-UI. Раздел «Installation» [Электронный ресурс]. URL: https://mui.com/getting-started/installation/ (дата обращения: 14.05.2025)
5. Chacon S., Straub B. Pro Git. Глава «Git Basics» [Электронный ресурс]. URL: <https://git-scm.com/book/ru/v2> (дата обращения: 15.05.2025)
6. GitLab Docs. GitLab Inc. CI/CD Pipelines. Раздел «Introduction to pipelines» [Электронный ресурс]. URL: <https://docs.gitlab.com/ee/ci/> (дата обращения: 12.05.2025)
7. GitHub Docs. GitHub, Inc. GitHub Actions: Continuous Integration and Continuous Delivery. Раздел «Concepts» [Электронный ресурс]. URL: <https://docs.github.com/en/actions> (дата обращения: 13.05.2025)
8. SonarSource. SonarSource SA. SonarQube Documentation. Раздел «Code Quality and Security» [Электронный ресурс]. URL: <https://docs.sonarqube.org/> (дата обращения: 12.05.2025)
9. Yandex.Practicum Blog. Yandex. Диаграмма потоков данных (DFD): теория и практика. Раздел «Уровни DFD» [Электронный ресурс]. URL: <https://practicum.yandex.ru/blog/diagramma-potokov-dannyh-dfd/> (дата обращения: 15.05.2025)
10. OWASP Foundation. OWASP. OWASP Top 10 – 2021: The Ten Most Critical Web Application Security Risks. Раздел «A1:2021 — Broken Access Control» [Электронный ресурс]. URL: https://owasp.org/www-project-top-ten/ (дата обращения: 14.05.2025)
11. PCI Security Standards Council. PCI SSC. PCI DSS v4.0. Раздел «Introduction» [Электронный ресурс]. URL: <https://www.pcisecuritystandards.org/> (дата обращения: 14.05.2025)
12. Fowler M. Martin Fowler. Patterns of Enterprise Application Architecture. Глава «Domain Logic Patterns» [Печатный ресурс]. Addison-Wesley, 2002.
13. Gamma E. et al. Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software. Глава «Structural Patterns» [Печатный ресурс]. Addison-Wesley, 1994.
14. ISO/IEC 12207:2008. ISO. Systems and software engineering — Software life cycle processes. Раздел «Processes» [Нормативный документ].
15. IEEE Std 830-1998. IEEE. Recommended Practice for Software Requirements Specifications. Раздел «Content of SRS» [Нормативный документ].

## **Глоссарий**

* Бэкенд (Backend) — серверная часть приложения, реализующая бизнес-логику и работу с данными.
* ER-диаграмма (Entity-Relationship Diagram) — графическая модель баз данных, показывающая сущности, их атрибуты и связи.
* Интеграция — процесс объединения различных программных модулей в единую систему.
* Неоморфизм (Neumorphism) — современный дизайн-стиль с мягкими тенями и сглаженными гранями.
* UML (Unified Modeling Language) — стандартизованный язык моделирования систем и процессов.

## **Список аббревиатур**

| **Аббревиатура** | **Расшифровка** |
| --- | --- |
| API | Application Programming Interface |
| CI/CD | Continuous Integration / Continuous Delivery |
| CSS | Cascading Style Sheets |
| DB | Data Base |
| DFD | Data Flow Diagram |
| HTML | HyperText Markup Language |
| JSON | JavaScript Object Notation |
| PK | Primary Key |
| REST | Representational State Transfer |
| SQL | Structured Query Language |
| UI | User Interface |
| UX | User Experience |
| UML | Unified Modeling Language |

## **Приложение**

Приложение А – Техническое Задание