Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра информатики

ОТЧЁТ

по лабораторной работе №1

на тему

**ИССЛЕДОВАНИЕ АНАЛОГОВ И ФОРМИРОВАНИЕ**

**ТРЕБОВАНИЙ К ХАРАКТЕРИСТИКАМ ПРОЕКТИРУЕМОЙ ПРОГРАММНОЙ СИСТЕМЫ**

Студент М.М. Зырянова

Преподаватель Е.В. Тушинская

Минск 2023

СОДЕРЖАНИЕ

[Введение 3](#_Toc146899625)

[1 Анализ прототипов и формирование требований к проектируемому программному продукту 4](#_Toc146899626)

[1.1 Анализ существующих аналогов 4](#_Toc146899627)

[1.2 Требования к проектируемому программному продукту 8](#_Toc146899628)

[2 Моделирование программного продукта и разработка функциональных требований 10](#_Toc146899629)

[2.1 Функциональная модель программного продукта 10](#_Toc146899630)

[2.2 Разработка спецификации функциональных требований программного продукта 12](#_Toc146899631)

[Заключение 15](#_Toc146899632)

[Список используемых источников 16](#_Toc146899633)

# **ВВЕДЕНИЕ**

Современный мир невозможно представить без финансов: валютно-фондовые биржи, банковские системы, активный рост криптовалюты – всё это мотивирует разработчиков создавать более прогрессивные и надежные финансовые приложения и технологии. Финтех как сфера деятельности программистов набирает всё большую популярность за счёт появления спроса на соответствующее программное обеспечение.

Наиболее часто используемыми сервисами финтех-индустрии являются интернет-банкинги, которые позволяют пользователям совершать различные финансовые операции, такие как переводы, платежи, открытие вкладов, оформление кредитов и т.д. Исходя из этого факта, можно предположить, что и потребительский спрос на такие приложения высок.

В процессе изучения индустрии для её дальнейшего развития было принято решение создать программный продукт, целью которого является разработка сервиса с надежной банковской системой для осуществления и обработки платежей.

При выборе способа реализации было принято решение создать веб-приложение, реализующее принципы REST (Representational state transfer) архитектуры. Это архитектурный подход, который устанавливает ограничения для API: как они должны быть устроены и какие функции поддерживать. Это позволяет стандартизировать работу программных интерфейсов, сделать их более удобными и производительными [1].

Всего в REST есть шесть требований к проектированию API. Пять из них обязательные, одно – опциональное:

– Клиент-серверная модель (client-server model).

– Отсутствие состояния (statelessness).

– Кэширование (cacheability).

– Единообразие интерфейса (uniform interface).

– Многоуровневая система (layered system).

– Код по требованию (code on demand) – необязательно [2].

**Исходя из вышеупомянутых характеристик, можно сделать вывод, что REST-подход к архитектуре** позволяет сделать сервисы отказоустойчивыми, гибкими и производительными, а при их масштабировании и внесении изменений не возникает больших сложностей [2].

Таким образом, целью данного проекта является разработка банковской системы, реализующая все функции и процессы, необходимые клиенту банка для осуществления и контроля платежей.

# **1 АНАЛИЗ ПРОТОТИПОВ И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К ПРОЕКТИРУЕМОМУ ПРОГРАММНОМУ ПРОДУКТУ**

**Успешность разработки программного продукта во многом зависит от этапа подготовки, на котором необходимо тщательно определить нюансы и особенности проекта.**

**Первое условие, которое необходимо выполнить перед началом разработки, – это ясное понимание проблемы, которую должна решить система. Адекватное планирование позволяет исключить основные риски на ранних стадиях проекта, чтобы основную часть работы можно было выполнить максимально эффективно.**

**Главный риск в разработке программного обеспечения – это плохое определение требований, которые подробно описывают, что должна делать программная система. Внимание к требованиям помогает минимизировать изменения системы после начала разработки.**

**Перед определением требований необходимо изучить ряд вопросов, которые напрямую влияют на все дальнейшие этапы разработки. По результатам анализа можно будет составить техническое задание, которое станет основой для составления функциональных требований.**

## **Анализ существующих аналогов**

Одним из основных пунктов в разработке программного обеспечения является анализ существующих на рынке аналогов. В данном случае необходимо рассмотреть примеры популярных банковских систем и провести их сравнение. Для изучения были выбраны такие банки, как Альфа-Банк и Белинвестбанк.

**1.1.1** Альфа-Банк

Архитектура банковской системы Альфа-Банка основана на принципах микросервисной архитектуры. Это означает, что система состоит из множества небольших, независимых сервисов, которые взаимодействуют друг с другом через API. Такая архитектура позволяет банку быстро и легко внедрять новые продукты и услуги, а также масштабировать систему в соответствии с ростом бизнеса [3].

Основными технологиями, используемыми в банковской системе Альфа-Банка, являются:

**– Java** – основной язык программирования для разработки сервисов.

**– Spring Boot** – фреймворк для разработки микросервисов на Java.

**– Docker** – технология для упаковки и доставки приложений.

**– Kubernetes** – система управления контейнеризованными приложениями.

– **PostgreSQL** – система управления базами данных.

– **RabbitMQ** – система обмена сообщениями.

На рисунке 1.1 представлена страница интернет-банка «Альфа-клик».

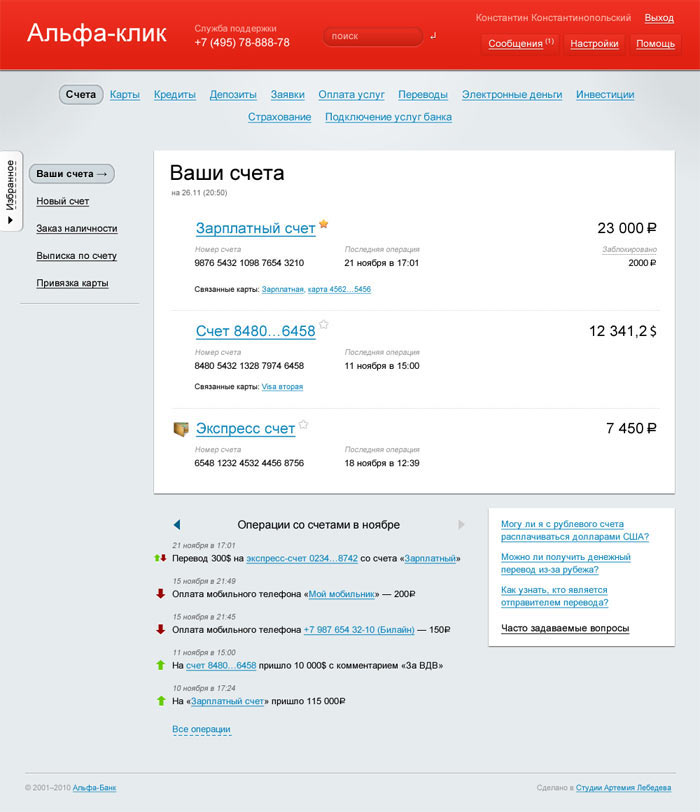


Рисунок 1.1 – «Альфа-клик»

Архитектура банковской системы Альфа-Банка имеет следующие особенности:

**– Централизация:** все сервисы системы связаны между собой через единую инфраструктуру. Это позволяет банку контролировать работу системы и обеспечивать ее надежность.

**– Масштабируемость:** система построена на основе микросервисной архитектуры, что позволяет банку легко масштабировать ее в соответствии с ростом бизнеса.

**– Безопасность:** система построена с учетом современных требований к безопасности.

**– Устойчивость:** система построена с учетом возможности возникновения сбоев и предусматривает их устранение.

Банк постоянно внедряет новые технологии в свою систему, чтобы соответствовать требованиям времени и предоставлять своим клиентам лучшие услуги.

Альфа-Банк стремится быть лидером в области инноваций и внедрять новые технологии, чтобы предоставлять своим клиентам лучшие услуги.

**1.1.2** Белинвестбанк

Архитектура банковской системы Белинвестбанка основана на принципах централизации и стандартизации. Все основные банковские процессы и функции сосредоточены в единой системе, которая обеспечивает их централизованное управление и контроль. Это позволяет банку эффективно управлять своим бизнесом и предоставлять клиентам качественные услуги.

В качестве основных технологий Белинвестбанк использует следующие решения:

– **Централизованная система управления процессами** (BPMS) на базе платформы IBM Business Process Manager. Эта система используется для автоматизации и оптимизации банковских процессов, а также для контроля их исполнения.

– **Централизованная система управления данными** (DBMS) на базе платформы Oracle Database. Эта система используется для хранения и обработки всех данных банка, включая финансовые, клиентские и операционные.

– **Система управления клиентскими взаимоотношениями** (CRM) на базе платформы Oracle CRM. Эта система используется для управления отношениями с клиентами, включая привлечение новых клиентов, обслуживание существующих клиентов и анализ клиентской базы.

– **Система дистанционного банковского обслуживания** (ДБО) на базе платформы Oracle Fusion Applications. Эта система позволяет клиентам банка совершать банковские операции в режиме онлайн, используя различные устройства.

На рисунке 1.2 представлена страница интернет-банкинга Белинвестбанка.

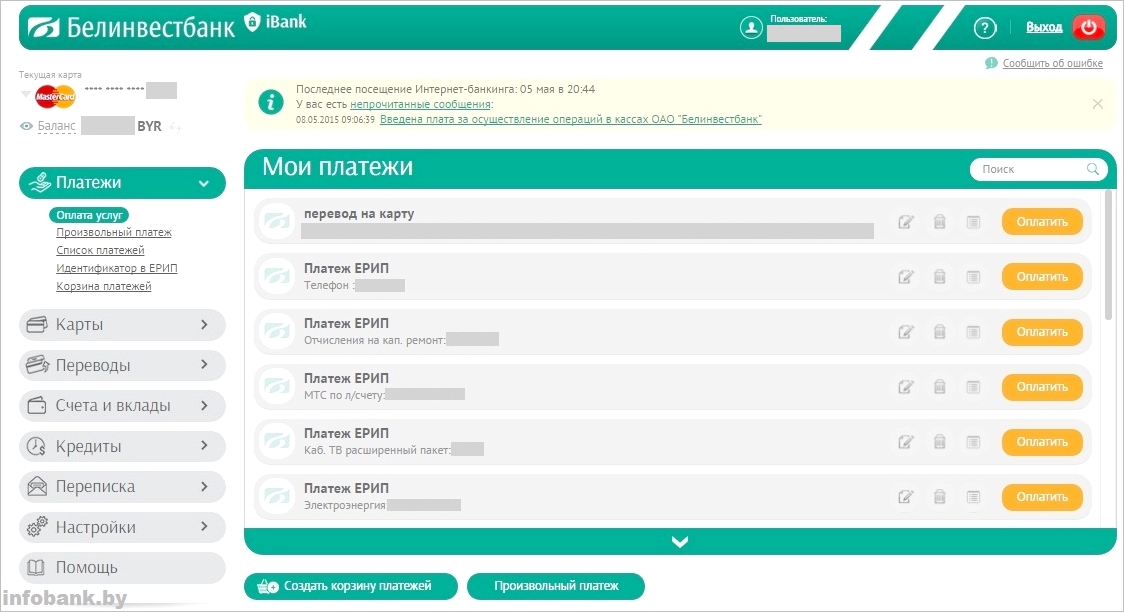


Рисунок 1.2 – Интернет-банкинг Белинвестбанка

Кроме того, Белинвестбанк использует ряд других технологий, включая:

– **Технологии искусственного интеллекта** для автоматизации банковских процессов и повышения качества обслуживания клиентов.

– **Технологии блокчейна** для обеспечения безопасности и прозрачности банковских операций.

– **Технологии больших данных** для анализа клиентской базы и принятия решений на основе данных.

Белинвестбанк постоянно внедряет новые технологии для повышения эффективности своей работы и улучшения качества обслуживания клиентов.

Эти особенности позволяют Белинвестбанку предоставлять своим клиентам современные банковские услуги на высоком уровне.

## **1.2 Требования к проектируемому программному продукту**

На основании анализа уже существующих аналогов были разработаны следующие требования к программному продукту.

**1.2.1** Назначение программного продукта

Данный программный продукт представляет собой банковскую систему и предназначен для совершения платежей и управления денежными счетами клиента.

**1.2.2** Основные функции программного продукта

Программный продукт должен поддерживать следующие функции:

– регистрация, аутентификация, авторизация;

– наличие системы ролей;

– управление личными кабинетами клиентов;

– управление депозитами клиентов;

– управление счетами клиентов;

– управление кредитами клиентов;

– оплата с использованием Touch ID.

**1.2.3** Дополнительные функции программного продукта

В качестве дополнительных функций предполагается реализация двухфакторной аутентификации и оплаты через Touch ID.

**1.2.4** Входные данные

Входные данные для программного средства должны быть представлены в виде вводимого пользователем с клавиатуры текста и опций, предоставляемых пользовательским интерфейсом приложения.

В бизнес-логике программного средства должны быть реализованы проверки входных данных на корректность, и в случае их некорректности, пользователь должен получать соответствующее уведомление с просьбой ввести данные необходимого формата.

**1.2.5** Выходные данные

Выходные данные должны быть представлены посредством отображения информации при помощи различных элементов реализованного и доступного пользовательского интерфейса.

* + 1. Выбор инструментов разработки

**В качестве основного языка программирования используется язык Python**. Это универсальный язык программирования, который часто используется для разработки веб-приложений. Python является одним из самых быстрых языков программирования, что важно для веб-приложений, которые должны обрабатывать большое количество запросов. Его синтаксис прост и удобен для чтения, что облегчает разработку и поддержку программного обеспечения.

**В качестве фреймворка используется FastAPI.** Он является одним из самых производительных фреймворков для RESTful API на Python. Также он прост в использовании и изучении и масштабируется до больших нагрузок.

**В качестве интегрированной среды разработки используется PyCharm**. Она предлагает широкий набор функций для разработки программного обеспечения на Python и является одной из самых производительных IDE.

**Для хранения данных используется база данных PostgreSQL**. Это реляционная база данных с открытым исходным кодом. PostgreSQL является одной из самых надежных баз данных, поддерживает широкий спектр функций, включая кластеризацию, репликацию и масштабируемость, а также предлагает большое количество функций безопасности.

**Для контейнеризации используется Docker** – система управления контейнерами, которая позволяет разработчикам создавать, тестировать и развертывать приложения в контейнерах. Контейнеры могут быть запущены на любой платформе, поддерживающей Docker. Также использование контейнеров эффективнее, чем использование виртуальных машин.

Системой контроля версий был выбран Git, а хостингом для хранения исходного кода проекта и его разработки – GitHub.

Сформулированные требования позволяют осуществить успешное проектирование и разработку программного средства. Язык Python, фреймворк FastAPI, IDE PyCharm, база данных PostgreSQL, система контроля версий Git и система управления контейнерами Docker являются проверенными технологиями, которые хорошо подходят для разработки веб-приложений.

# **2 МОДЕЛИРОВАНИЕ ПРОГРАММНОГО ПРОДУКТА И РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ ТРЕБОВАНИЙ**

## **2.1 Функциональная модель программного продукта**

Функциональная модель программного средства представлена в виде

диаграммы вариантов использования и информационной модели предметной области. Варианты использования отражают функциональность системы в ответ на внешние воздействия с точки зрения получения значимого результата для пользователей.

Исходя из выдвинутых требований, проектируемое программное средство предполагает поддержку системы ролей, т.е. разделение пользователей на различные роли, а следовательно, и выдача различных прав нескольким категориям пользователей: гостю, клиенту и администратору.

Перейдем к более подробному рассмотрению функциональной модели программного продукта.

Базово при открытии приложения пользователь имеет роль «Гость» – тип пользователя, не зарегистрированного в системе. По сути, все его действия ограничиваются возможностью регистрации, авторизации и аутентификации в системе.

После регистрации либо входа в систему пользователь получает роль «Клиент». Ему доступны все базовые функции приложения: управление личным кабинетом, управление счетами, управление депозитами, управление кредитами.

Рассмотрим каждую из функций по отдельности:

– Управление личным кабинетом. Пользователю доступна возможность изменения и просмотра информации своего личного кабинета. К поддающейся редактированию личной информации можно отнести такие данные, как фамилия, имя, отчество, адрес электронной почты, номер телефона и т.д.

– Управление счетами. Эта функция включает в себя возможность открытия и закрытия счёта, перевода денег между счетами.

– Управление депозитами. Данная функция описывает возможность внесения и погашения депозитов на определенный срок.

– Управление кредитами. Здесь описана возможность взятия и погашения кредита под определенный процент на определенный срок.

Последняя роль – «Администратор». Его функционал достаточно широк, поскольку он имеет права на совершение действий над любыми пользователями. Данной роли доступны следующие функции:

– Управление пользователями. Удаление, блокировка, восстановление – всё это основные операции, которые позволяют определенные действия над пользователями.

– Управление счетами клиентов. Сюда входит создание и блокировка счета пользователя.

Также администраторы имеют доступ к разрешенным и незашифрованным данным в БД. Они должны обладать достаточной квалификацией для решения технических задач, из чего следует, что они тоже должны быть из команды разработчиков.

Возможности пользователей каждой роли в системе представлены на рисунке 2.1 в диаграммы в нотации UML.



Рисунок 2.1 – Диаграмма вариантов использования

## **2.2 Разработка спецификации функциональных требований программного продукта**

С учетом определенных требований представим детализацию функций проектируемого программного продукта.

Для детализации функций рассмотрим основные требования, предъявляемые к каждой функции программного средства как с точки зрения внутренней организации системы, так и с точки зрения взаимодействия системы с пользователем.

При разработке спецификаций функциональных требований кроме всего прочего учитывались особенности платформы разработки.

**2.2.1** Функция регистрации

Функция регистрации должна быть реализована с учетом следующих требований:

– процесс регистрации инициируется пользователем системы с ролью «Гость»;

– правильность и устойчивость ко взломам введенного пароля должна быть проверена при помощи встроенных инструментов разработки;

– в случае некорректности введенных данных пользователь должен увидеть сообщение об этом с предложением попробовать еще раз;

– предусмотреть возможность смены пароля и после регистрации.

**2.2.2** Функция аутентификации

Функция аутентификации должна быть реализована с учетом следующих требований:

– инициатором является пользователь, при этом ему необходимо предоставить имя пользователя и пароль, заданные при регистрации;

– подтверждение пользователя осуществляется с помощью стандарта JWT Token.

– реализована возможность повторной аутентификации пользователя;

– реализована возможность восстановления пароля.

Для восстановления пароля пользователь должен предоставить мобильный телефон, заданный при заполнении личного кабинета. На предоставленный адрес электронной почты присылается новый пароль. После входа по данному паролю пользователь может вновь сменить его на желаемый.

**2.2.3** Функция управления личными кабинетами клиентов

Функция управления личными кабинетами клиентов должна быть реализована с учетом следующих требований:

– клиент может редактировать личную информацию в своем аккаунте (фамилию, имя, отчество, электронную почту и т.д.);

– клиент может сменить пароль либо имя пользователя в своем аккаунте;

– клиент может просматривать личную информацию.

**2.2.4** Функция управления депозитами клиентов

Функция управления депозитами клиентов должна быть реализована с учетом следующих требований:

– наличие возможности открытия депозита сроком от 3 месяцев до 3 лет;

– необходимо предусмотреть как единичные, так и ежемесячные выплаты депозита;

– процентные ставки зависят от условий открытия депозитарного счёта.

**2.2.5** Функция управления счетами клиентов

Функция управления счетами клиентов должна быть реализована с учетом следующих требований:

– открытие счетов осуществляется в трёх валютах: белорусских рублях (BYN), российских рублях (RUB) и американских долларах (USD).

– предусмотрен перевод денежных средств между счетами;

– администратор наделяется правами блокировки и разблокировки счетов.

**2.2.6** Функция управления кредитами клиентов

Функция управления кредитами клиентов должна быть реализована с учетом следующих требований:

– должна быть реализована возможность взятия кредита сроком от 3 месяцев до 5 лет;

– необходимо предусмотреть как единичные, так и ежемесячные выплаты по кредитам;

– процентные ставки зависят от условий взятия кредита.

**2.2.7** Функция оплаты с использованием Touch ID

Функция оплаты с использованием Touch ID должна быть реализована с учетом следующих требований:

– при корректном срабатывании происходит перевод денежных средств, оплата кредита и т.д.;

– при некорректном срабатывании клиент уведомляется об ошибке;

– предусмотреть повторное использование данной функции для оплаты.

**2.2.8** Функция управления пользователями

Функция управления пользователями должна быть реализована с учетом следующих требований:

– должна быть реализована возможность создания личного кабинета для клиента банка;

– при обнаружении неправомерных или подозрительных действий администратор, который наделен данными полномочиями, может заблокировать личный кабинет пользователя;

– предусмотреть удаление личного кабинета пользователя при расторжении договора с банком.

# **ЗАКЛЮЧЕНИЕ**

Таким образом, в ходе выполнения работы было составлено техническое задание для разработки сервиса банковской системы.

Были проанализированы аналоги, уже имеющиеся на рынке финансовых технологий и составлены требования к программному продукту.

Во второй части отчета была составлена диаграмма вариантов использования, разработаны функциональные требования и их спецификации.

Исходя из всего вышесказанного, можно смело говорить об успешном выполнении поставленных задач.

# **СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

[1] Архитектура REST [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/articles/38730/> – Дата доступа: 27.09.23.

[2] REST API: что это такое и как работает [Электронный ресурс]. – Режим доступа: <https://skillbox.ru/media/code/> – Дата доступа: 27.09.23.

[3] Технологии ОАО «Альфа-банк» [Электронный ресурс]. – Режим доступа: <https://studwood.net/1983524/menedzhment/> – Дата доступа: 27.09.23.