МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«Санкт-Петербургский национальный исследовательский университет  
информационных технологий, механики и оптики»

ФАКУЛЬТЕТ ПРОГРАММНОЙ ИНЖЕНЕРИИ И   
КОМПЬЮТЕРНОЙ ТЕХНИКИ

ОТЧЁТ  
ПО ЛАБОРАТОРНОЙ РАБОТЕ №3

по дисциплине  
«Бизнес-логика программных систем»  
Вариант №1135

Выполнили:Студент группы P3314  
Селянта Олег Дмитриевич  
Преподаватель:Цопа Евгений Алексеевич

Изображение выглядит как черный, темнота

Контент, сгенерированный ИИ, может содержать ошибки.

Санкт-Петербург, 2025

Оглавление

[Задание 3](#_Toc200537480)

[Внимание! У разных вариантов разный текст задания! 3](#_Toc200537481)

[Модель потока управления для автоматизируемого бизнес-процесса со всеми внесёнными изменениям: 5](#_Toc200537482)

[UML-диаграммы классов и пакетов разработанного приложения. 5](#_Toc200537483)

[UML-диаграмма пакетов разработанного приложения 5](#_Toc200537484)

[UML-диаграмма классов для common-dto 5](#_Toc200537485)

[Спецификация REST API для всех публичных интерфейсов разработанного приложения 6](#_Toc200537486)

[Диаграмма развёртывания (Deployment Diagram), визуализирующая интеграцию с EIS. 8](#_Toc200537487)

[Исходный код системы или ссылка на репозиторий с исходным кодом 8](#_Toc200537488)

[Выводы 9](#_Toc200537489)

# **Задание**

### Внимание! У разных вариантов разный текст задания!

Доработать приложение из лабораторной работы #2, реализовав в нём асинхронное выполнение задач с распределением бизнес-логики между несколькими вычислительными узлами и выполнением периодических операций с использованием планировщика задач, а также интеграцию с внешней информационной системой.

**Требования к реализации асинхронной обработки:**

1. Перед выполнением работы неободимо согласовать с преподавателем набор прецедентов, в реализации которых целесообразно использование асинхронного распределённого выполнения задач. Если таких прецедентов использования в имеющейся бизнес-процесса нет, нужно согласовать реализацию новых прецедентов, доработав таким образом модель бизнес-процесса из лабораторной работы #1.
2. Асинхронное выполнение задач должно использовать модель доставки "очередь сообщений".
3. В качестве провайдера сервиса асинхронного обмена сообщениями необходимо использовать очередь сообщений на базе RabbitMQ.
4. Для отправки сообщений необходимо использовать протокол AMQP 1.0. Библиотеку для реализации отправки сообщений можно взять любую на выбор студента.
5. Для получения сообщений необходимо использовать слушателя сообщений JMS на базе Spring Boot (@JmsListener).

**Требования к реализации распределённой обработки:**

1. Обработка сообщений должна осуществляться на двух независимых друг от друга узлах сервера приложений.
2. Если логика сценария распределённой обработки предполагает транзакционность выполняемых операций, они должны быть включены в состав распределённой транзакции.

**Требования к реализации запуска периодических задач по расписанию:**

1. Согласовать с преподавателем прецедент или прецеденты, в рамках которых выглядит целесообразным использовать планировщик задач. Если такие прецеденты отсутствуют -- согласовать с преподавателем новые и добавить их в модель автоматизируемого бизнес-процесса.
2. Реализовать утверждённые прецеденты с использованием планировщика задач Spring (@Scheduled).

**Требования к интеграции с внешней Корпоративной Информационной Системой (EIS):**

1. Корпоративная Информационная Cистема, с которой производится интеграция, а также её функциональные возможности выбираются на усмотрение преподавателя и согласуются с ним.
2. Взаимодействие с внешней Копроративной Информационной Системой должно быть реализовано с помощью технологии JCA (Jakarta Connectors).

**Правила выполнения работы:**

1. Все изменения, внесённые в реализуемый бизнес-процесс, должны быть учтены в описывающей его модели, REST API и наборе скриптов для тестирования публичных интерфейсов модуля.
2. Доработанное приложение необходимо либо развернуть на сервере helios, либо продемонстрировать его работоспособность на собственной инфраструктуре обучающегося.

**Содержание отчёта:**

1. Текст задания.
2. Модель потока управления для автоматизируемого бизнес-процесса со всеми внесёнными изменениями.
3. UML-диаграммы классов и пакетов разработанного приложения.
4. Спецификация REST API для всех публичных интерфейсов разработанного приложения.
5. Диаграмма развёртывания (Deployment Diagram), визуализирующая интеграцию с EIS.
6. Исходный код системы или ссылка на репозиторий с исходным кодом.
7. Выводы по работе.

**Вопросы к защите лабораторной работы:**

1. Асинхронное выполнение задач. Преимущества и недостатки, подходы к реализации.
2. Спецификация Java Message Service.
3. Ресурсы и сообщения JMS. Модели взаимодействия "очередь" и "подписка". Распределённая обработка сообщений.
4. Протоколы взаимодействия с очередями сообщений: MQTT, AMQP, STOMP, XMPP. Отправка сообщений с использованием HTTP + WebSockets.
5. Apache ActiveMQ. Архитектура, способы взаимодействия, поддерживаемые протоколы, особенности реализации JMS. Протокол OpenWire и его реализации для различных платформ.
6. RabbitMQ. Архитектура, способы взаимодействия, поддерживаемые протоколы, особенности реализации JMS.
7. Apache Kafka. Особенности обработки сообщений, сходства и отличия с очередями сообщений. Архитектура, особенности построения масштабируемых решений, интеграция с Service Discovery.
8. Периодические задачи, планировщики выполнения задач.
9. Cron. Архитектура, интеграция в ОС, способы конфигурации, синтаксис Cron Expression.
10. Quartz. Архитектура, интеграция с приложением, способы конфигурации.
11. Выполнение периодических задач в Java / Jakarta EE и Spring. Java / Jakarta EE Timer Services и Spring @Scheduled.
12. Подходы к интеграции информационных систем -- в общем, и на примере JCA / Jakarta Connectors.

### Модель потока управления для автоматизируемого бизнес-процесса со всеми внесёнными изменениям:

https://drive.google.com/file/d/19pVMk5xzze4VsnaLs2nhaNJVVDxLR4lL/view?usp=sharing

### UML-диаграммы классов и пакетов разработанного приложения.

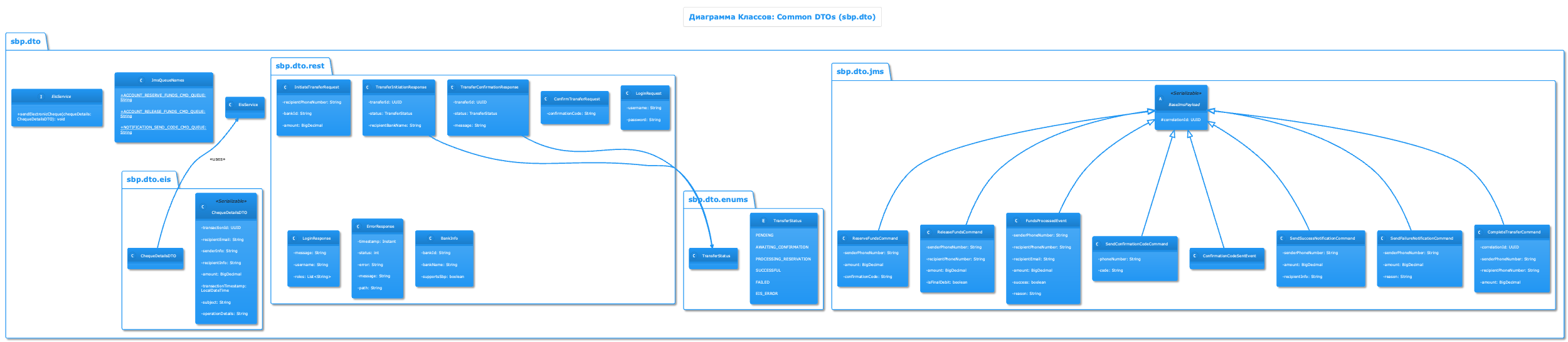
Так как разработанное приложение имеет слишком большое количество классов и отображать их ввиде uml нет никакого смысла, то представлю отдельно диаграмму для пакетов разработанного приложения, а также классов common-dto, где хранятся все классы, при помощи которых сервисы обмениваются данными

### UML-диаграмма пакетов разработанного приложения

A screenshot of a computer

AI-generated content may be incorrect.

### UML-диаграмма классов для common-dto



### Спецификация REST API для всех публичных интерфейсов разработанного приложения

Базовый путь: /api/v1

* **Аутентификация:**
  + POST /auth/login
    - **Описание:** Аутентификация пользователя.
    - **Request Body:** LoginRequest (JSON: username, password)
    - **Response:**
      * 200 OK: LoginResponse (JSON: message, username, roles)
      * 401 Unauthorized: ErrorResponse (JSON) - неверные учетные данные.
    - **Аутентификация:** Не требуется.
* **Операции с переводами:**
  + POST /transfers/initiate
    - **Описание:** Инициация нового перевода.
    - **Request Body:** InitiateTransferRequest (JSON: recipientPhoneNumber, bankId, amount)
    - **Response:**
      * 202 Accepted: TransferInitiationResponse (JSON: transferId, status, recipientBankName)
      * 400 Bad Request: ErrorResponse (JSON) - ошибка валидации, банк не найден/не поддерживает СБП.
      * 401 Unauthorized: Если пользователь не аутентифицирован.
      * 403 Forbidden: Если у пользователя нет роли USER.
    - **Аутентификация:** Basic Auth, роль USER. (Username - номер телефона отправителя)
  + POST /transfers/{transferId}/confirm
    - **Описание:** Подтверждение перевода кодом.
    - **Path Variable:** transferId (UUID) - ID перевода.
    - **Request Body:** ConfirmTransferRequest (JSON: confirmationCode)
    - **Response:**
      * 200 OK: TransferConfirmationResponse (JSON: transferId, status, message)
      * 400 Bad Request: ErrorResponse (JSON) - неверный transferId, некорректный статус перевода, неверный код.
      * 401 Unauthorized: Если пользователь не аутентифицирован.
      * 403 Forbidden: Если пользователь не является отправителем данного перевода или не имеет роли USER.
    - **Аутентификация:** Basic Auth, роль USER. (Username - номер телефона отправителя)
  + GET /transfers/{id}
    - **Описание:** Получение деталей перевода (административная функция).
    - **Path Variable:** id (UUID) - Primary Key перевода из БД.
    - **Response:**
      * 200 OK: Transfer (JSON, полная сущность перевода)
      * 404 Not Found: ErrorResponse (JSON) - перевод не найден.
      * 401 Unauthorized: Если пользователь не аутентифицирован.
      * 403 Forbidden: Если у пользователя нет роли ADMIN.
    - **Аутентификация:** Basic Auth, роль ADMIN.

**2. SBP Adapter Service (порт: 8083) (ДЛЯ ВНУТРЕННЕГО ОБЩЕНИЯ СЕРВИСОВ!)**

Базовый путь: /api/sbp

* **Операции с банками (мок):**
  + GET /banks
    - **Описание:** Поиск информации о банке по номеру телефона (эмуляция запроса в СБП).
    - **Request Param:** phoneNumber (string, 10 цифр, обязательный)
    - **Response:**
      * 200 OK: BankInfo (JSON: bankId, bankName, supportsSbp)
      * 404 Not Found: Если банк для номера телефона не найден.
      * 400 Bad Request: Если phoneNumber невалиден.
    - **Аутентификация:** Не требуется.
  + GET /banks/{bankId}
    - **Описание:** Получение информации о банке по его ID (эмуляция запроса в СБП).
    - **Path Variable:** bankId (string, обязательный)
    - **Response:**
      * 200 OK: BankInfo (JSON: bankId, bankName, supportsSbp)
      * 404 Not Found: Если банк с таким ID не найден.
    - **Аутентификация:** Не требуется.
* **Операции с переводами (мок):**
  + POST /transfers
    - **Описание:** Обработка (эмуляция) запроса на перевод через СБП.
    - **Request Body:** SbpAdapterRequest (JSON: senderPhoneNumber, recipientPhoneNumber, amount, correlationId)
    - **Response:**
      * 200 OK: SbpAdapterResponse (JSON: success: true, sbpTransactionId) - успешная эмуляция.
      * 422 Unprocessable Entity: SbpAdapterResponse (JSON: success: false, errorMessage) - эмуляция бизнес-ошибки СБП.
      * 500 Internal Server Error: SbpAdapterResponse (JSON: success: false, errorMessage) - эмуляция технической ошибки СБП.
      * 400 Bad Request: Если тело запроса невалидно.
    - **Аутентификация:** Не требуется.

**3. Notification Service (порт: 8082)**

Базовый путь: /ui

* **Отображение кодов подтверждения (UI):**
  + GET /codes
    - **Описание:** Отображает HTML страницу с активными кодами подтверждения (для демонстрации).
    - **Request:** Нет.
    - **Response:**
      * 200 OK: HTML страница (active-codes.html).
    - **Аутентификация:** Не требуется.

### Диаграмма развёртывания (Deployment Diagram), визуализирующая интеграцию с EIS.

A screenshot of a computer

AI-generated content may be incorrect.

### Исходный код системы или ссылка на репозиторий с исходным кодом

<https://github.com/Oleg-cmd/blps>

### Выводы

В ходе выполнения лабораторной работы №2 было успешно доработано приложение сервиса переводов, реализовав два ключевых аспекта: управление транзакциями и разграничение доступа.