Акционерное общество «Национальная платежная корпорация Национального Банка Республики Казахстан»

Утверждаю
Председатель Правления
АО «НПК»

Самаева Ж.Т.

«30» декабря 2025г.

Система обмена информацией по открытым программным интерфейсам. Техническая спецификация. «Получение информации о банковском счете клиента»

Рег. № 61

г. Алматы

Содержание

Глава 1. Общие сведения

1. Настоящий документ «Система обмена информацией по открытым программным интерфейсам. Техническая спецификация. «Получение информации о банковском счете» (далее – Техническая спецификация) разработан в целях систематизации и формализации технических требований к процессам обмена информацией о банковском счете клиента посредством Системы обмена информацией по открытым программным интерфейсам (далее - Система) и применяется при создании программных средств, предназначенных для предоставления информации о банковском счете клиента.

2. Техническая спецификация предназначена для технических специалистов, которые занимаются созданием, настройкой и/или поддержкой приложений, а также программных средств, предназначенных для обмена информацией о банковских счетах посредством Системы обмена информацией по открытым программным интерфейсам (далее – Система).

3. Техническая спецификация разработана в соответствии с приказом Министра финансов Республики Казахстан от 4 апреля 2025 года №151 «Некоторые вопросы Единой бюджетной классификации Республики Казахстан», постановлениями Правления Национального Банка Республики Казахстан от 31 августа 2016 года № 203 «Об утверждении Правил применения кодов секторов экономики и назначения платежей», от 31 августа 2016 года №207 «Об утверждении Правил открытия, ведения и закрытия банковских счетов клиентов», а также следующими государственными и международными стандартами:

1) СТ РК 34.005-2002 «Информационная технология. Основные термины и определения»;

2) НК РК 07 ISO 4217-2019 «Коды для представления валют и фондов»;

3) ISO 8601 «Элементы данных и форматы для обмена информацией. Обмен информацией. Представление дат и времени»;

4) RFC 7158 «The JavaScript Object Notation (JSON) Data Interchange Format»;

5) RFC 7231 «Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content»;

6) RFC 4122 «A UniveRSAlly Unique IDentifier (UUID) URN Namespace».

Примечание – при использовании стандартов целесообразно проверить действие ссылочных стандартов и классификаторов по ежегодно издаваемому информационному указателю «Нормативные документы по стандартизации» по состоянию на текущий год и соответствующим ежемесячно издаваемым информационным указателям, опубликованным в текущем году. Если ссылочный документ заменен (изменен), то следует руководствоваться замененным (измененным) документом. Если ссылочный документ отменен без замены, то положение, в котором дана ссылка на него, применяется в части, не затрагивающей эту ссылку.

Глава 2. Термины, определения, обозначения и сокращения:

4. В Технической спецификации применяются следующие термины и сокращения с соответствующими определениями:

1) АРРФР - Агентство Республики Казахстан по регулированию и развитию финансового рынка;

2) НБРК – Национальный Банк Республики Казахстан;

3) НПК, АО «НПК» - АО «Национальная платежная корпорация Национального Банка Республики Казахстан»;

4) прикладной программный интерфейс (application program interface, API) - интерфейс между прикладным программным средством и прикладной платформой, через который обеспечивается доступ к необходимым службам;

5) конечная точка (endpoint) - адрес ресурса, точка входа интерфейса;

6) параметр (claim) - часть информации о субъекте. Параметр представлен в виде пары имя/значение, состоящей из имени параметра и значения параметра;

7) JSON веб-токен (JWT) - строка, представляющая набор параметров в формате объекта JSON, который закодирован в виде структуры JWS или JWE, сопровождая параметры цифровой подписью, кодом аутентификации и/или шифрованием;

8) токен доступа (access token) - свидетельство, представляющее авторизацию, выданную клиенту сервером авторизации с одобрения владельца ресурса. Токен доступа содержит указание на конкретные области действия, к которым разрешен доступ, длительность доступа и другие параметры;

9) Open API - открытый интерфейс программирования приложений (англ. - Open Application Programming Interface), который предоставляет разработчикам программный доступ к закрытому программному приложению, программе для обмена данными;

10) Open Banking - концепция открытого, технологического обмена данными клиентов между финансовыми, платежными организациями и сторонними поставщиками услуг с их согласия для развития инноваций и повышения конкуренции на финансовом рынке;

11) Система обмена информацией по открытым программным интерфейсам - информационная система, состоящая из программных и аппаратных средств, предназначенная для технологического и безопасного обмена данными с помощью Open API;

12) Оператор - Акционерное общество «Национальная платежная корпорация Национального Банка Республики Казахстан», обеспечивающее функционирование Системы;

13) Реестр - совокупность данных, зафиксированных на бумажном носителе и (или) с использованием электронной базы данных;

14) Поставщик API - юридическое лицо, зарегистрированное в Реестре Поставщиков API, осуществившие публикацию разработанного API в Системе;

15) Пользователь API - юридическое лицо, зарегистрированное в Реестре Пользователей API, осуществившие подключение к API в Системе;

16) IP-адрес - уникальный числовой идентификатор устройства в компьютерной сети.

17) БВУ - банк второго уровня;

18) API (Application Program Interface) - прикладной программный интерфейс;

19) GMT (Greenwich Mean Time) - среднее время по Гринвичу;

20) HTTP (Hypertext Transfer Protocol) - протокол передачи гипертекстовых сообщений;

21) IBAN (International Bank Account Number) - международный банковский номер счета;

22) ISO (International Organization for Standardization) - Международная организация по стандартизации.

23) JSON (JavaScript Object Notation) - текстовый формат обмена данными, основанный на языке программирования JavaScript (нотация объектов JavaScript);

24) JWT (JSON Web Token) - токен доступа, основанный на формате JSON;

25) JWS (JSON Web Signature) - структура данных в формате JSON, представляющая сообщение с цифровой подписью или кодом аутентификации сообщений;

26) JWE (JSON Web Encryption) - структура данных в формате JSON, представляющая зашифрованное и защищенное от модификации сообщение;

27) REST (Representational State Transfer) - архитектурный стиль взаимодействия компонентов распределенного приложения в сети. Для веб-служб, построенных с учетом REST, применяют термин «RESTful»;

28) RFC (Request for Comments) - предложения для обсуждения; серия нормативных документов, стандартизирующих протоколы Internet;

29) URI (Uniform Resource Identifier) - унифицированный идентификатор ресурса;

30) URL (Uniform Resource Locator) - унифицированный адрес ресурса (единый указатель ресурса);

31) UTF-8 - стандарт кодирования символов, позволяющий компактно хранить и передавать символы Unicode, используя переменное количество байтов (от 1 до 4), и обеспечивающий полную обратную совместимость с кодировкой ASCII;

32) UUID (Universally Unique Identifier) - универсальный уникальный идентификатор;

33) YAML - формат сериализации данных, концептуально близкий к языкам разметки, но ориентированный на удобство ввода-вывода типичных структур данных многих языков программирования.

Глава 3. Общее описание процесса

5. Клиент должен явно дать согласие для получения информации о счете для Пользователя API. Должна быть исключена возможность получения информации о счете Клиента, на предоставление которой явным образом не получено действующее согласие от Клиента. Клиент может отозвать согласие в любой момент времени. После отзыва Клиентом согласия получение информации в рамках отозванного согласия становится недоступным.

6. Общий процесс получения информации о банковском счете Клиента включает в себя следующие основные шаги:

1) инициирование Клиентом (конечным пользователем) получения информации о банковском счете в приложении Пользователя API. Доступ к данным Клиента может быть предоставлен только с его согласия и в рамках оказываемых ему услуг Пользователем API;

2) аутентификация и регистрация согласия Клиента на доступ к его данным – Клиент (конечный пользователь) должен дать согласие на доступ к его данным, после чего Пользователь API получает аутентификационный токен доступа, используя который может получить сведения о банковском счете Клиента.

Авторизация согласия и получение аутентификационного токена осуществляется следующим образом:

• Пользователь API устанавливает защищенный канал связи и запрашивает прохождение аутентификации на Сервере авторизации;

• Сервер авторизации, используя сценарий с генерацией кода авторизации (Authorization Code):

• перенаправляет Клиента для прохождения аутентификации и регистрации согласия на доступ к его данным;

• Клиент проходит процедуру аутентификации;

• Клиент должен явно дать согласие для получения сведений о его банковских счетах для Пользователя API. Клиенту должна быть предоставлена возможность отклонить согласие: если клиент отклоняет согласие, то процесс прекращается, запрос отклоняется;

• Клиент авторизует согласие на доступ к списку его банковских счетов для Пользователя API, производится регистрация согласия Клиента;

• как только согласие было авторизовано, Клиент перенаправляется обратно на сторону Пользователя API с кодом авторизации и Пользователь API обменивает код авторизации на Сервере авторизации на токен доступа;

3) получение информации о банковском счете Пользователем API – для получения информации о банковском счете используется токен доступа, полученный на предыдущем шаге. Сведения предоставляются только при условии успешного прохождения проверок аутентификационных данных Пользователя API. Клиент в приложении Пользователя API уведомляется о результате получения сведений о его банковских счетах. Перечень методов API для получения информации о банковском счете Клиента, описание форматов запросов и ответов предусмотрены в настоящей Технической спецификации.

***Примечание**: дополнительно электронная версия спецификации методов API для получения информации о банковском счете Клиента приведена в https://accounts-openapi.npck.kz/.*

7. В процессе информационного взаимодействия для идентификации счета Клиента должны использоваться идентификаторы, генерацию которых обеспечивает Система.

Параграф 1. Кодировка символов

8. В запросах и ответах на запросы должна использоваться кодировка UTF-8.

Примечание – Согласно RFC 7158 (см. раздел «Библиография», пункт **[1]**) кодировкой символов по умолчанию для JSON является UTF-8.

9. Если отклоняется сообщение с символом UTF-8, который не может быть обработан, то код состояния ответного сообщения должен быть HTTP 400 (неверный запрос).

Параграф 2. Формат даты и времени

10. При передаче даты и времени в полезных нагрузках JSON они должны быть представлены в формате dateTime стандарта ISO 8601 (см. раздел «Библиография», пункт **[2]**). Все поля dateTime в ответах должны содержать часовой пояс.

Примеры

1 2023-03-03T13:23:13+06:00

2 2023-11-07T00:10:00.001Z

11. При передаче даты и времени в параметрах строки запроса они должны иметь формат dateTime стандарта 8601 (см. раздел «Библиография», пункт **[2]**) и включать часовой пояс.

12. Все даты в заголовках HTTP должны быть представлены как полные даты в соответствии с RFC 7231 (см. раздел «Библиография», пункт **[3]**).

Пример

Sun, 10 Sep 2023 17:18:19 GMT

13. Все даты в параметрах claims JWT имеют формат number JSON, который представляет количество секунд с 1970-01-01T0:0:0Z, измеренное в GMT до текущей даты (dateTime).

Пример

1518446700 (соответствует 12 февраля 2018 года 14:45:00 GMT)

Параграф 3. Правила передачи необязательных полей

14. В объектах, где значение для необязательного поля не указано, поле исключается из полезной нагрузки JSON.

Примеры

1 "description": "" – некорректный способ передачи пустой строки, в данном случае поле не должно передаваться

2 "creditorAccount": {} – некорректный способ передачи пустого объекта, в данном случае поле не должно передаваться

15. В объектах, где поле массива определено как необязательное, то такое поле массива включается в полезную нагрузку с пустым массивом, если нет данных для него.

Пример

"transactions": [] – корректный способ передачи пустого массива

Параграф 4. Нумерация страниц

16. Для конечных точек, которые возвращают множественные записи, должен поддерживаться постраничный ответ. При этом, нумерация страниц начинается с 1.

17. Если исходный запрос включал параметры фильтра, то в ответе возвращаются только те результаты, которые соответствуют фильтру.

Параграф 5. Используемые области действия

18. Разграничение прав доступа к информации осуществляется в соответствии с областью действия (scope), связанной с предъявленным токеном доступа.

19. Для разграничения прав доступа к информации о банковском счете клиента определены следующие области действия (scope):

- accounts – доступ к списку счетов;
- account_balance – доступ к информации о балансе счета;
- account_transactions – доступ к списку транзакций счета.

Глава 4. Требования информационной безопасности

Параграф 1. Общие положения

20. Глава устанавливает технические требования информационной безопасности при обмене информацией о банковском счете Клиента посредством Системы.

21. При реализации приложений Пользователя API и Поставщика API должны соблюдаться требования нормативных правовых актов Республики Казахстан в области защиты персональных данных, банковской тайны и информационной безопасности, а также требования Оператора, указанные в пунктах 75–77 настоящей Технической спецификации.

Параграф 2. Аутентификация и авторизация

22. Доступ к конечным точкам API, описанным в настоящей Технической спецификации, допускается только при наличии действующего токена доступа (access token) формата JWT, выданного Сервером авторизации Системы в соответствии с процессом, описанным в Главе 3.

23. Пользователь API и Поставщик API должны обеспечивать проверку:

1) подлинности токена доступа (целостность, срок действия, статус отзыва, соответствие идентификаторам Клиента и приложения);

2) области действия (scope), связанной с токеном доступа, в соответствии с пунктами 18–19 настоящей Технической спецификации;

3) соответствия запрашиваемого ресурса (конечной точки и параметров запроса) разрешенной области действия (scope).

24. После отзыва Клиентом согласия дальнейшее использование ранее выданных токенов доступа для получения информации о банковском счете Клиента не допускается.

Параграф 3. Криптографическая защита информации

25. Передача сообщений между Пользователем API, Поставщиком API и Сервером авторизации должна осуществляться только по защищенным каналам связи с использованием протокола TLS версии не ниже 1.2. Рекомендуется использование версии TLS 1.3 при технической возможности.

26. При передаче аутентификационных данных и токенов доступа должны использоваться только защищенные соединения (HTTPS). Использование незашифрованных соединений (HTTP) для обмена данными, содержащими аутентификационную информацию, персональные данные или банковскую тайну, не допускается.

27. При реализации механизмов подписи и/или шифрования структур JWS и JWE должны использоваться криптографические алгоритмы и параметры, соответствующие требованиям регуляторов и стандартов информационной безопасности, действующих в Республике Казахстан.

Параграф 4. Управление секретами и токенами

28. Пользователь API обязан обеспечить защиту Client Secret и токенов доступа от несанкционированного доступа. Указанные значения должны храниться и обрабатываться исключительно на серверной стороне приложения Пользователя API, в соответствии с пунктами 78 и 83 настоящей Технической спецификации.

29. Не допускается:

1) передача Client Secret и токенов доступа на клиентскую сторону (в браузер, мобильное приложение) в открытом виде;

2) журналирование Client Secret и токенов доступа в незащищенные журналы и системы мониторинга;

3) включение Client Secret и токенов доступа в URL-параметры запросов.

30. Рекомендуется реализация механизмов:

1) плановой автоматической ротации Client Secret не реже одного раза в 12 месяцев и внеплановой ротации при подозрении на компрометацию;

2) автоматического обновления токенов доступа не реже одного раза в 60 минут.

Параграф 5. Журналирование и аудит

31. Пользователь API и Поставщик API должны обеспечивать журналирование операций доступа к API, включая как минимум:

1) идентификатор запроса (requestId) при его наличии;
2) дату и время запроса;
3) идентификатор приложения Пользователя API;
4) идентификатор Клиента;
5) запрошенный ресурс (конечная точка и основные параметры);
6) результат обработки запроса (успешно/ошибка, код HTTP-статуса).

32. Журналы должны храниться в защищенной среде, с разграничением прав доступа и защитой от несанкционированной модификации и удаления. Срок хранения журналов определяется внутренними документами Оператора, Пользователя API и Поставщика API с учетом требований законодательства Республики Казахстан.

Параграф 6. Защита от злоупотреблений и атак

33. Для конечных точек API должны быть реализованы механизмы ограничения частоты запросов (rate limiting, throttling) и защиты от распределенных отказов в обслуживании (DoS/DDoS). При превышении допустимой частоты запросов должен возвращаться код HTTP 429 (Too Many Requests) с использованием заголовка Retry-After.

34. Должна быть реализована проверка корректности структуры и содержимого всех входных данных (заголовков, параметров пути и строки запроса, тела сообщений) с целью предотвращения атак типа SQL Injection, XSS, XML/JSON-инъекций и других видов внедрения вредоносных данных.

35. Повторная обработка ранее принятых запросов должна контролироваться. Рекомендуется использование уникальных идентификаторов запросов и/или других механизмов защиты от атак типа replay, особенно для операций, связанных с изменением состояния.

Параграф 7. Конфиденциальность и минимизация данных

36. При формировании ответов на запросы API должны передаваться только те данные, которые необходимы для реализации соответствующего сценария в рамках области действия (scope), связанной с токеном доступа, в соответствии с пунктами 18–19 настоящей Технической спецификации.

37. В журналах и сообщениях об ошибках не допускается вывод в открытом виде персональных данных и реквизитов банковских счетов Клиента, за исключением случаев, когда такое отображение необходимо для выполнения требований законодательства или внутренних процедур расследования инцидентов безопасности.

Глава 5. Модель данных

Параграф 1. Коды статусов и параметры HTTP

38. При успешной обработке запроса HTTP статус ответа должен быть 200.

39. Для ответов с ошибками не допускается использование HTTP статуса 200. Для ответов с ошибками должен использоваться соответствующий HTTP статус из Приложения 1 к Технической спецификации.

40. В HTTP заголовке запроса могут использоваться параметры, приведенные в таблице 1.

Таблица 1 – Параметры HTTP заголовка запроса

Параметр	Описание	Обязательно
Authorization	Стандартный заголовок HTTP. Используется для передачи аутентификационных данных	Да
Accept	Стандартный HTTP заголовок, определяющий тип контента, который требуется от сервера. По умолчанию используется значение application/json	Нет
x-provider-id	Идентификатор Поставщика API в формате UUID (см. раздел «Библиография», пункт **[4]**).	Условно

	Используется для маршрутизации запросов. Обязательность: - для Пользователя API – обязательно, т.е. должно быть обязательно указано в запросе для корректной маршрутизации сообщения в Системе; - для Поставщика API – необязательно, т.е. не требуется реализация поддержки параметра, т.к. непосредственно не используется на стороне Поставщика API (не должно быть ошибки при отсутствии в запросе, полученном Поставщиком API).	

41. В HTTP заголовке ответа на запрос могут использоваться параметры, приведенные в таблице 2.

Таблица 2 – Параметры HTTP заголовка ответа на запрос

Параметр	Описание	Обязательно
Content-Type	Стандартный параметр заголовка HTTP. Представляет формат полезной нагрузки, возвращаемой в ответе на запрос	Да
Retry-After	Параметр заголовка, указывающий время (в секундах), через которое следует повторить запрос. Используется с ответами с кодом состояния HTTP 429 (Too Many Requests)	Условно

42. В запросах получения информации о банковском счете полезная нагрузка не используется.

Параграф 2. Общая структура полезной нагрузки ответа на запрос

43. Структура верхнего уровня для полезной нагрузки ответа на запрос может содержать следующие элементы:

1) <data> – обязательный элемент для методов предоставления списка банковских счетов, предоставления данных о балансе банковского счета, предоставления данных о транзакциях банковского счета (Главы 6-8), содержит данные ответа. Структура этого элемента может отличатся для каждой конечной точки;

2) <page> – опциональный элемент, предназначен для передачи данных для пагинации страниц, используется для конечных точек, в которых возвращаются множественные записи. Подробное описание структуры элемента приведено в Приложении 2 к Технической спецификации (тип данных «Page»);

3) <exists> – обязательный элемент для метода проверки существования счета (Глава 9).

Пример структуры верхнего уровня полезной нагрузки приведен на рисунке 1.

```
{
    "data": {
            …
    },
    "page": {
            …
    },
    "exists": <boolean>
}
```

Рисунок 1 – Пример структуры верхнего уровня полезной нагрузки

Параграф 3. Структура ответа с информацией об ошибке

44. Структура верхнего уровня для ответов с информацией об ошибке может содержать следующие элементы:

1) <code> – обязательный элемент, содержит код ошибки, необходимый для классификации возникшей ошибки;

2) <description> – обязательный элемент, содержит описание ошибки;

3) <requestId> – опциональный элемент, содержит идентификатор, присвоенный запросу.

45. Детальное описание формата ответа с информацией об ошибке приведено в приложении 1 к Технической спецификации.

Глава 6. Предоставление списка банковских счетов

Параграф 1. Спецификация конечной точки предоставления списка банковских счетов

46. Конечная точка предназначена для обеспечения возможности получения списка открытых (активных) банковских счетов клиента с его согласия.

47. Конечная точка предоставления списка банковских счетов должна поддерживать метод HTTP GET.

48. Наименование конечной точки предоставления списка банковских счетов: GET /<version>/accounts. Описание конечной точки в формате YAML приведено в Приложении 3 к Технической спецификации.

Примечание – В качестве <version> указывается версия API, выраженная в виде v[номер версии], например: v1, V3 и т.д.

Параграф 2. Спецификация запроса предоставления списка банковских счетов

49. В качестве параметров строки запроса предоставления списка банковских счетов могут использоваться:

- <pageNumber> – номер страницы результата запроса (целочисленный), нумерация страниц начинается с 1. Опциональный параметр. Значение по умолчанию: 1.
- <pageSize> – размер страницы для пагинации результата запроса (целочисленный). Опциональный параметр. Значение по умолчанию: 10. Максимальное допустимое значение: 100.
- <from> – дата и время начала для фильтрации списка счетов. Опциональный параметр. Не должен быть позже <to>.
- <to> – дата и время окончания для фильтрации списка счетов. Опциональный параметр. Не должен быть раньше <from>.
- <type> – тип счета (массив строк). Принимает значения из справочника (см. Приложение 4 к Технической спецификации, справочник «Тип банковского счета»). Опциональный параметр. Если не указан, в ответ включаются все доступные типы счетов из справочника.

50. В запросе получения списка банковских счетов полезная нагрузка не используется.

Параграф 3. Спецификация ответа на запрос предоставления списка банковских счетов

51. Структура полезной нагрузки ответа на запрос предоставления списка банковских счетов должна соответствовать общей структуре полезной нагрузки, определенной в пункте 43.

52. В составе элемента <data> полезной нагрузки ответа на запрос предоставления списка банковских счетов должен передаваться объект, описание структуры которого приведено в Приложении 2 к Технической спецификации (тип данных «AccountsResponseV3»).

Глава 7. Предоставление данных о балансе банковского счета

Параграф 1. Спецификация конечной точки предоставления данных о балансе банковского счета

53. Конечная точка предназначена для обеспечения возможности получения данных о балансе банковского счета клиента с его согласия.

54. Конечная точка предоставления данных о балансе банковского счета должна поддерживать метод HTTP GET.

55. Наименование конечной точки предоставления данных о балансе банковского счета: GET /<version>/accounts/<accountId>/balances. Описание конечной точки в формате YAML приведено в Приложении 3 к Технической спецификации.

Примечание – В качестве <version> указывается версия API, выраженная в виде v[номер версии], например: v1, V3 и т.д. В качестве <accountId> должен указываться идентификатор банковского счета.

Параграф 2. Спецификация запроса предоставления данных о балансе банковского счета

56. В качестве параметра пути запроса предоставления данных о балансе банковского счета используется:

- <accountId> – идентификатор банковского счета (строковый), по которому запрашиваются данные. Обязательный параметр.

57. В запросе получения данных о балансе банковского счета полезная нагрузка не используется.

Параграф 3. Спецификация ответа на запрос предоставления данных о балансе банковского счета

58. Структура полезной нагрузки ответа на запрос предоставления данных о балансе банковского счета должна соответствовать общей структуре полезной нагрузки, определенной в пункте 43.

59. В составе элемента <data> полезной нагрузки ответа на запрос предоставления данных о балансе банковского счета должен передаваться объект, описание структуры которого приведено в Приложении 2 к Технической спецификации (тип данных «BalanceResponse»).

Глава 8. Предоставление данных о транзакциях банковского счета

Параграф 1. Спецификация конечной точки предоставления данных о транзакциях банковского счета

60. Конечная точка предназначена для обеспечения возможности получения перечня транзакций банковского счета клиента с его согласия.

61. Конечная точка предоставления данных о транзакциях банковского счета должна поддерживать метод HTTP GET.

62. Наименование конечной точки предоставления данных о транзакциях банковского счета: GET /<version>/accounts/<accountId>/transactions. Описание конечной точки в формате YAML приведено в Приложении 3 к Технической спецификации.

Примечание – В качестве <version> указывается версия API, выраженная в виде v[номер версии], например: v1, V3 и т.д. В качестве <accountId> должен указываться идентификатор банковского счета.

Параграф 2. Спецификация запроса предоставления данных о транзакциях банковского счета

63. В качестве параметра пути запроса предоставления данных о транзакциях банковского счета используется:

- <accountId> – идентификатор банковского счета (строковый), по которому запрашиваются данные. Обязательный параметр.

64. В качестве параметров строки запроса предоставления данных о транзакциях банковского счета могут использоваться:

- <pageNumber> – номер страницы результата запроса (целочисленный), нумерация страниц начинается с 1. Опциональный параметр. Значение по умолчанию: 1.
- <pageSize> – размер страницы для пагинации результата запроса (целочисленный). Опциональный параметр. Значение по умолчанию: 10. Максимальное допустимое значение: 100.
- <from> – дата и время начала для фильтрации списка транзакций по дате и времени операции. Опциональный параметр. Не должен быть позже <to>, не должен быть ранее, чем 180 дней с текущей даты. Значение по умолчанию: текущая дата.

- <to> – дата и время окончания для фильтрации списка транзакций по дате и времени операции. Опциональный параметр. Не должен быть раньше <from>. Максимальный период для запроса 90 дней (с даты <from>). Значение по умолчанию: дата и время получения запроса.

65. В запросе получения данных о транзакциях банковского счета полезная нагрузка не используется.

Параграф 3. Спецификация ответа на запрос предоставления данных о транзакциях банковского счета

66. Структура полезной нагрузки ответа на запрос предоставления данных о транзакциях банковского счета должна соответствовать общей структуре полезной нагрузки, определенной в пункте 43.

67. В составе элемента <data> полезной нагрузки ответа на запрос предоставления данных о транзакциях банковского счета должен передаваться объект, описание структуры которого приведено в Приложении 2 к Технической спецификации (тип данных «TransactionResponse»).

Глава 9. Проверка существования банковского счета

Параграф 1. Спецификация конечной точки проверки существования банковского счета

68. Конечная точка предназначена для обеспечения возможности проверки существования банковского счета клиента.

69. Конечная точка проверки существования банковского счета должна поддерживать метод HTTP GET.

70. Наименование конечной точки проверки существования банковского счета: GET /<version>/accounts/<iban>/exists. Описание конечной точки в формате YAML приведено в Приложении 3 к Технической спецификации.

Примечание – В качестве <version> указывается версия API, выраженная в виде v[номер версии], например: v1, V3 и т.д. В качестве <iban> должен указываться номер (IBAN) банковского счета.

Параграф 2. Спецификация запроса проверки существования банковского счета

71. В качестве параметра пути запроса проверки существования банковского счета используется:

- <iban> – номер (IBAN) банковского счета (строковый), по которому запрашиваются данные. Обязательный параметр.

72. В запросе проверки существования банковского счета полезная нагрузка не используется.

Параграф 3. Спецификация ответа на запрос проверки существования банковского счета

73. Структура полезной нагрузки ответа на запрос проверки существования банковского счета должна соответствовать общей структуре полезной нагрузки, определенной в пункте 43.

74. В составе элемента <exists> полезной нагрузки ответа на запрос содержится результат проверки существования банковского счета (логический).

Глава 10. Информационный обмен между участниками

75. Адреса тестового и промышленного окружения:

1) Тестовое окружение:

- адрес личного кабинета Поставщика / Пользователя API: https://cabinet.stage.npck.kz;
- адрес тестового стенда: https://cabinet.stage.npck.kz.

2) Промышленное окружение:

- адрес личного кабинета Поставщика / Пользователя API: https://cabinet.npck.kz;
- адрес промышленного стенда: https://cabinet.npck.kz.

76. При осуществлении информационного взаимодействия участники Системы могут исполнять различные роли.

77. В рамках взаимодействия при получении сведений о счетах клиентов предусмотрены следующие роли участников:

1) Регулятор (НБРК, АРРФР) - оказывает организационно-методологическую поддержку, осуществляет общий надзор;

2) Оператор (НПК) - основной зоной ответственности является осуществление комплексной информационно-консультационной и технической поддержки участников, обеспечение функционирования Системы;

3) Поставщик API (БВУ) – финансовая организация, обслуживающая счет Клиента и публикующая API для предоставления сведений о счетах Клиента;

4) Пользователь API (БВУ) – юридическое лицо, использующее API для доступа к информации о банковском счете Клиента с его согласия;

5) Клиент – физическое лицо, которое разрешает или запрещает доступ к своим персональным данным, банковской и иной охраняемой законом тайне. Доступ к данным банковского счета будет возможен только после предоставления Клиентом согласия.

78. Информационный обмен между участниками Системы основывается на следующих принципах:

1) отсутствие приоритетов обработки сообщений, обработка сообщений осуществляется на началах равенства участников независимо от их форм собственности, местонахождения и других обстоятельств;

2) прием и обработка сообщений осуществляется в режиме реального времени;

3) доступ к персональным и/или банковским данным Клиента возможен только с его согласия.

79. Информационный обмен между участниками осуществляется посредством Системы в электронном формате.

80. При информационном обмене между участниками сообщения подразделяются на два типа: «запрос» и «ответ». Инициализирующим информационный обмен сообщением является сообщение типа «запрос». На сообщение типа «запрос» должно быть выслано сообщение типа «ответ».

81. Участники обмениваются электронными сообщениями в форматах, разработанных Оператором совместно с Регуляторами. Оператор обеспечивает их размещение на портале Системы. Сообщения, не соответствующие установленным требованиям, должны отклоняться.

82. Передача сообщений между Пользователем API и сервером авторизации Системы должна производиться с использованием защищенного канала информационного обмена (применяется протокол TLS версии 1.2 или более поздней).

83. Участники информационного взаимодействия должны соблюдать надлежащий режим конфиденциальности, в том числе хранения банковской тайны и защиты персональных данных, и принимать все необходимые меры по защите указанной информации от разглашения в соответствии с регуляторными требованиями и требованиями:

1) стандартов в области информационной безопасности, в том числе требованиями по обеспечению безопасности персональных данных;

2) нормативными правовыми актами Республики Казахстан, включая Закон Республики Казахстан «О персональных данных и их защите»;

3) требованиями по неразглашению охраняемой законом тайны, в том числе Закона Республики Казахстан «О банках и банковской деятельности в Республике Казахстан»;

4) другими нормативными правовыми актами, устанавливающими требования к обеспечению информационной безопасности.

84. В рамках получения информации о банковском счете Клиента взаимодействие Системы и Поставщика API осуществляется по выделенным каналам связи.

Глава 11. Рекомендации по реализации для Пользователя API

Параграф 1. Подключение к API

85. Для подключения к API, опубликованному в Системе, Пользователю API необходимо выполнить следующие шаги:

1) зарегистрироваться в Системе в качестве Пользователя API;

2) войти в личный кабинет и зарегистрировать приложение Пользователя API в Системе.

Регистрация приложения осуществляется в личном кабинете, после прохождения которой предоставляются специальные учетные данные (credentials):

Client ID – уникальный идентификатор приложения.

Client Secret – пароль приложения (строго конфиденциальная информация).

В соответствии с требованиями главы 4! *Client Secret должен сохраняться в тайне и не передаваться в публичный доступ, должен храниться на серверной стороне приложения, и вся обработка, связанная с ним, должна производиться на серверной стороне приложения.*

3) выбрать API для подключения из перечня опубликованных в личном кабинете;

4) выполнить доработку программного обеспечения Пользователя API для вызова API.

5) реализовать перенаправление Клиента для прохождения аутентификации и регистрации его согласия на доступ к данным, реализовать получение кода авторизации (см. Параграф 2 Главы 10).

86. Для получения информации о Поставщиках API предназначены служебные методы, описанные в https://docs.npck.kz/mezhbankovskaya-sistema-perevodov-i-platezhei/dopolnitelnye-servisy/poluchenie-informacii-o-bankakh-i-statuse-api.

1) реализовать получение токена доступа (см. Параграф 3 Главы 10).

2) реализовать получение данных посредством интерфейсов API. Для получения информации посредством интерфейсов API используется токен доступа, полученный на предыдущем шаге. Перечень методов API для получения информации о банковском счете Клиента, описание форматов запросов и ответов приведены в настоящей Технической спецификации.

***Примечание**: Дополнительно электронная версия спецификации методов API для получения информации о банковском счете Клиента приведена в https://accounts-openapi.npck.kz/.*

3) реализовать отображение Клиенту полученной информации в приложении Пользователя API.

Параграф 2. Перенаправление Клиента для прохождения аутентификации и регистрации его согласия на доступ к данным

87. Для получения кода авторизации необходимо:

1) реализовать получение URL-адреса для перенаправления Клиента для прохождения аутентификации. Описание метода «Получить URL-адрес для перенаправления Клиента для прохождения аутентификации» (generate-user-url) приведено см. https://auth-openapi.npck.kz/#tag/Auth/operation/generateUserUrl.

2) реализовать перенаправление Клиента на URL-адреса, полученный на предыдущем шаге для аутентификации Клиента и получения его согласия на доступ к данным.

***Примечание**: Клиент (конечный пользователь) может отклонить согласие на доступ к его данным. В таком случае процесс завершается, код авторизации не предоставляется.*

Пример ответа, когда пользователь отклонил согласие на доступ отображает рисунок 2. В процессе реализации структура сообщения может быть скорректирована или дополнена.

https://<exampleredirecturi>/?errorCode=access_denied&state=<examplestate>

Рисунок 2 – Пример ответа, когда пользователь отклонил согласие на доступ

3) реализовать получение кода авторизации.

88. Если аутентификация Клиента проходит успешно, и он дает согласие на доступ к его данным, то Клиент перенаправляется Пользователю API (на указанный в запросе *redirectUri*) с кодом авторизации. Получив успешный ответ с кодом авторизации, Пользователь API должен проверить, совпадает ли значение параметра <state>, полученное в составе ответа, со значением параметра <state> в запросе из шага 1).

89. Пример ответа с кодом авторизации отображает Рисунок 3.

https://<exampleredirecturi>/?code=<exampleauthcode>**&**state=<examplestate>

Рисунок 3 – Пример ответа с кодом авторизации

Параграф 3. Получение токена доступа

90. Для получения токена доступа необходимо:

1) реализовать формирование запроса получения токена доступа, используя полученный код авторизации. Описание метода «Получить токен доступа» приведено в https://auth-openapi.npck.kz/#tag/Auth/operation/getOauthToken.

2) реализовать отправку запроса, сформированного на шаге 1, и получение токена доступа. Токен доступа может использоваться для получения сведений о счете Клиента посредством API, в течении срока его действия, после истечения срока действия необходимо получить новый токен доступа.

Важно! *Токен доступа должен сохраняться в тайне и не передаваться в публичный доступ, должен храниться на серверной стороне приложения, и вся обработка, связанная с ним, должна производиться на серверной стороне приложения.*

Примечание: *Дополнительно Пользователь API может использовать опциональный метод интроспекции токена (см описание https://auth-openapi.npck.kz/#tag/Auth/operation/oauth2Introspect). Данный метод используется для проверки того, активен или истек ли конкретный токен, а также для получения связанных с токеном метаданных.*

Глава 12. Рекомендации по реализации для Поставщика API

91. Для организации информационного взаимодействия посредством Системы Поставщику API необходимо выполнить следующие шаги:

1) зарегистрироваться в Системе в качестве Поставщика API.

Примечание: *подробное описание регистрации, входа в личный кабинет, публикации API и других функций, предоставляемых посредством пользовательского интерфейса Системы, приведено в https://docs.npck.kz/nachalo-raboty/kabinet-uchastnika.*

2) выполнить доработку программного обеспечения Поставщика API:

- реализовать получение идентификаторов Системы (см. Главу 11).
- выполнить разработку программных интерфейсов взаимодействия.

Перечень методов API для получения информации о банковском счете Клиента, описание форматов запросов и ответов приведены в настоящей Технической спецификации.

Примечание*: Дополнительно электронная версия спецификации методов API для получения информации о банковском счете Клиента приведена в https://accounts-openapi.npck.kz/.*

3) войти в личный кабинет и опубликовать разработанные API в Системе.

92. Поставщик API предоставляет информацию о банковских счетах только в случае успешной валидации запроса, в том числе Система выполняет валидацию токена при маршрутизации сообщений, вместе с тем, для повышения безопасности рекомендуется выполнять проверку валидности токена также на стороне Поставщика API.

93. Сервер авторизации (Системы) предоставляет возможность получения сведений из набора опубликованных ключей (JSON Web Key Set) в котором, содержится ключ проверки подписи сервера авторизации(см. https://auth-openapi.npck.kz/#tag/Auth/operation/wellKnown). Общие требования к структурам JWK и JWK Set приведены в RFC 7517 (https://datatracker.ietf.org/doc/html/rfc7517);

1) должен проверять, что область действия (параметр <scope>), связанная с предъявленным токеном доступа, соответствует запрошенному доступу;

2) должен возвращать только ресурс, соответствующий объекту, явно указанному в запросе на доступ, и при условии, что доступ к этому ресурсу является допустимым с учетом разрешенной области действия (параметр <scope>).

Глава 13. Использование идентификаторов Системы

94. Для процессов информационного взаимодействия по получению информации о текущем счете Клиента для идентификации счета используется идентификатор, сгенерированный Системой (Идентификатор Системы).

95. Пользователь API получает идентификаторы для счетов Клиента посредством обращения к Системе. При этом, получение идентификаторов Системы может быть организовано при обработке запросов, а также в рамках процессов управления счетами клиентов Поставщиком API (например, при регистрации нового счета Клиента).

96. Описание прикладного программного интерфейса для получения идентификатора Системы приведено в https://obid-openapi.npck.kz/#tag/OBID/operation/getOBIDsByIBANs.

97. Общий процесс использования идентификаторов Системы в рамках информационного взаимодействия по получению информации о текущем счете Клиента включает в себя следующие шаги:

1) Поставщик API, получив запрос предоставления списка текущих счетов Клиента, проверяет для всех ли открытых текущих счетов Клиента имеется идентификатор Системы;

2) Поставщик API для открытых текущих счетов Клиента, по которым ранее не был получен идентификатор Системы, запрашивает в Системе идентификаторы (accountId);

3) Поставщик API, получив идентификаторы Системы, сохраняет их и информацию о том, каким счетам Клиента (каким IBAN) они соответствуют;

4) Поставщик API в ответе на запрос предоставления списка текущих счетов в качестве идентификатора счета передает идентификатор Системы (accountId);

5) Поставщик API при последующем получении запроса предоставления информации по идентификатору Системы (например, баланса счета) при его обработке использует сопоставление идентификатора Системы (accountId) с номером банковского счета Клиента (IBAN).

Приложение 1
к Системе обмена информацией по
открытым программным интерфейсам.
Техническая спецификация. «Получение
информации о банковском счете клиента»
(информационное)

Описание ошибок

1. Коды статусов HTTP для ответов с ошибками

HTTP статус	Описание
400	Некорректный запрос
401	Заголовок авторизации отсутствует или неверный/недействующий токен
403	Токен имеет неверную область действия
404	Запрошен ресурс, который не реализован, или ресурс, который не определен
405	Попытка получить доступ к ресурсу с помощью метода, который не поддерживается
406	Запрос содержал параметр заголовка Accept, отличный от разрешенных media types, и/или набор символов, отличный от UTF-8
429	Операция не выполнена. Превышен лимит запросов к API в течении определенного времени. Необходимо повторить запрос позже. В HTTP заголовке Retry-After должно передаваться время до начала предоставления доступа в секундах
500	Операция не выполнена. Внутренняя ошибка сервера

2. Формат ответа с информацией об ошибке

Поле	Описание	Тип данных	Обязательность
code	Код ошибки	string (40)	Да
description	Описание ошибки	string (250)	Да
requestId	Идентификатор, присвоенный запросу	string (36)	Нет

3. Коды ошибок

Код ошибки	HTTP-статус	Описание
FIELD_MISSING	400	Не передан обязательный элемент
FIELD_INVALID	400	В элементе указано недопустимое значение или длина предоставленного значения не соответствует допустимым параметрам
HEADER_MISSING	400	Обязательный элемент HTTP-заголовка не был предоставлен
HEADER_INVALID	400	В элементе HTTP-заголовка указано неверное значение
RESOURCE_NOT_FOUND	400	Ресурс с указанным идентификатором не найден
INTERNAL_ERROR	500	Внутренняя ошибка сервера

Приложение 2
к Системе обмена информацией по
открытым программным интерфейсам.
Техническая спецификация. «Получение
информации о банковском счете клиента»
(информационное)

Описание типов данных

1. Тип данных «AccountsResponseV3»

Поле	Описание	Тип данных	Обязательно
accounts	Информация об открытых (активных) счетах клиента. Должен содержать информация только об открытых активных (т.е. действующих, не заблокированных, арестованных и т.п.) счетах	Массив объектов «AccountV3» (см. раздел 2 Приложения 2)	Да

2. Тип данных «AccountV3»

Поле	Описание	Тип данных	Обязательно
accountId	Идентификатор банковского счета клиента (описание использования идентификатора в Системе приведено в Приложении 5 к Технической спецификации)	string (UUID)	Да
currentBalance	Текущий баланс счета. В тиын, копейках, центах и т.п. Например, 100,01 передается как 10001, а 100,00 как 10000	integer	Да

Поле	Описание	Тип данных	Обязательно
currency	Валюта ведения счета. Используется стандарт ISO 4217 (см. раздел «Библиография», пункт [5]). Шаблон: ^[A-Z]{3,3}$ Пример: KZT	string (3)	Да
openedDateTime	Дата и время открытия счета. Используется стандарт ISO 8601 (см. раздел «Библиография», пункт [2]). Пример: 2023-03-03T13:23:13+06:00	string (dateTime)	Да
description	Описание	string (70)	Нет
maskedNumber	4 последние цифры: - номера счета IBAN – для текущего счета; - номера карты – для карточных счетов.	string (4)	Да
type	Тип счета	string (справочное значение, см. раздел 5 Приложения 4)	Да
altAccountId	Альтернативный номер счета. При необходимости его можно использовать для передачи дополнительной информации об идентификаторе счета.	string (40)	Нет

3. Тип данных «BalanceResponseV3»

Поле	Описание	Тип данных	Обязательно
currentBalance	Текущий баланс счета. В тиын, копейках, центах и т.п. Например, 100,01 передается как 10001, а 100,00 как 10000	integer	Да
availableBalance	Доступные средства. В тиын, копейках, центах и т.п. Например, 100,01 передается как 10001, а 100,00 как 10000	integer	Да
blockedBalance	Блокированные средства. В тиын, копейках, центах и т.п. Например, 100,01 передается как 10001, а 100,00 как 10000	integer	Нет
currency	Валюта. Используется стандарт ISO 4217 (см. раздел «Библиография», пункт [5]). Шаблон: ^[A-Z]{3,3}$ Пример: KZT	string (3)	Да
creditLine	Подробная информация о кредитной линии	Массив объектов «CreditLine» (см. раздел 4 Приложения 2)	Нет
purses	Необязательный массив остатков по счету в других валютах	Массив объектов «Amount» (см. раздел 5 Приложения 2)	Нет

4. Тип данных «CreditLine»

Поле	Описание	Тип данных	Обязательно
included	Признак, указывающий, включена ли кредитная линия в баланс счета	boolean	Да
type	Тип кредитного лимита	string (справочное значение, см. 4 Приложения 4)	Да
amount	Сведения о сумме и валюте	Объект «Amount» (см. раздел 5 Приложения 2)	Да

5. Тип данных «Amount»

Поле	Описание	Тип данных	Обязательно
amount	Сумма. В тиын, копейках, центах и т.п. Например, 100,01 передается как 10001, а 100,00 как 10000	integer	Да
currency	Валюта. Используется стандарт ISO 4217 (см. раздел «Библиография», пункт [5]). Шаблон: ^[A-Z]{3,3}$ Пример: KZT	string (3)	Да

6. Тип данных «TransactionResponseV3»

Поле	Описание	Тип данных	Обязательно
transactions	Сведения о транзакциях	Массив объектов «TransactionV3» (см. раздел 7 Приложения 2)	Да

7. Тип данных «TransactionV3»

Поле	Описание	Тип данных	Обязательно
transactionId	Идентификатор транзакции	string (36)	Да
documentNumber	Номер платежного документа	string (70)	Нет
reference	Референс транзакции	string (36)	Да
status	Статус обработки транзакции	string (справочное значение, см. 1 Приложения 4)	Да
amount	Информация о сумме и валюте транзакции	Объект «Amount» (см. раздел 5 Приложения 2)	Да
chargeAmount	Информация о комиссии	Объект «Amount» (см. раздел 5 Приложения 2)	Нет
creditDebitIndicator	Определяет, является транзакция кредитовой или дебетовой	string (справочное значение, см. раздел 2 Приложения 4)	Да
type	Тип операции	string (справочное значение, см. раздел 3 Приложения 4)	Да
purposeCode	Код назначения платежа (см. раздел «Библиография», пункт [7])	string (3)	Нет
budgetCode	Код бюджетной классификации (классификация поступлений бюджета) (см. раздел «Библиография», пункт [6])	string (6)	Нет

Поле	Описание	Тип данных	Обязательно
senderCode	Код отправителя денег (КОд) (см. раздел «Библиография», пункт [7])	string (2)	Нет
beneficiaryCode	Код бенефициара (КБе) (см. раздел «Библиография», пункт [7])	string (2)	Нет
mcc	Код категории продавца (Merchant category code). Обязательно указывается для карточных счетов	string (4)	Нет
description	Описание	string (70)	Нет
createDateTime	Дата и время проведения операции. Используется стандарт ISO 8601 (см. раздел «Библиография», пункт [2]). Пример: 2023-03-03T13:23:13+06:00	string (dateTime)	Да
bookingDateTime	Дата и время выполнения транзакции, т.е. когда транзакция проведена и становится неизменной. Используется стандарт ISO 8601 (см. раздел «Библиография», пункт [2]). Пример: 2023-03-03T13:23:13+06:00	string (dateTime)	Нет

Поле	Описание	Тип данных	Обязательно
valueDateTime	Дата и время валютирования. Используется стандарт ISO 8601 (см. раздел «Библиография», пункт [2]). Пример: 2023-03-03T13:23:13+06:00	string (dateTime)	Нет
creditorName	Наименование получателя средств в случае дебетовой транзакции	string (70)	Нет
debtorName	Наименование плательщика в случае кредитной транзакции	string (70)	Нет
creditorAgent	Финансовая организация, обслуживающая счет получателя средств	Объект «FinancialInstitutionIdentificationType» (см. раздел 8 Приложения 2)	Нет
debtorAgent	Финансовая организация, обслуживающая счет плательщика	Объект «FinancialInstitutionIdentificationType» (см. раздел 8 Приложения 2)	Нет
rrn	Референсный номер транзакции (RRN)	string (30)	Нет
iinSender	HEX-строка SHA512 от ИИН отправителя. Шаблон: ^[0-9A-Fa-f]{128}$	string (128)	Нет
iinReceiver	HEX-строка SHA512 от ИИН получателя. Шаблон: ^[0-9A-Fa-f]{128}$	string (128)	Нет

8. Тип данных «FinancialInstitutionIdentificationType»

Поле	Описание	Тип данных	Обязательно
bic	Банковский идентификационный код (БИК)	string (8)	Да
bin	Бизнес-идентификационный номер (БИН)	string (12)	Нет
name	Наименование	string (256)	Нет

9. Тип данных «Page»

Поле	Описание	Тип данных	Обязательно
totalItems	Количество элементов, соответствующих параметрам запроса	integer	Да
isLastPage	Признак последней страницы	boolean	Да

Приложение 3
к Системе обмена информацией по
открытым программным интерфейсам.
Техническая спецификация. «Получение
информации о банковском счете клиента»
(информационное)

Описание прикладного программного интерфейса предоставления информации о банковских счетах клиента в формате YAML

```
openapi: 3.0.3
info:
  title: Open Banking Accounts API
  version: 1.0.0
servers:
  - url: https://api.npck.kz
    description: Production server
  - url: https://api.stage.npck.kz
    description: Test server
tags:
  - name: Accounts
    description: Open Banking Accounts API
paths:
  /v1/accounts/{iban}/exists:
    get:
      description: This endpoint verifies whether an account exists by IBAN. Required for account
prevalidation.
      parameters:
        - $ref: '#/components/parameters/iban'
      responses:
        '200':
          $ref: '#/components/responses/200AccountExists'
        '400':
          $ref: '#/components/responses/400Error'
        '401':
          $ref: '#/components/responses/401Error'
        '403':
          $ref: '#/components/responses/403Error'
        '404':
          $ref: '#/components/responses/404Error'
        '405':
          $ref: '#/components/responses/405Error'
        '406':
          $ref: '#/components/responses/406Error'
        '429':
          $ref: '#/components/responses/429Error'
        '500':
          $ref: '#/components/responses/500Error'
      operationId: accountExists
      security:
        - OauthAccessTokenAuth: []
```

```
    tags:
     - Accounts
    x-codeSamples:
     - lang: cURL
       source: |
         curl --location 'https://api.npck.kz/v1/accounts/<string>/exists \
         --header 'x-provider-id: <uuid>' \
         --header 'Accept: application/json' \
         --header 'Authorization: Bearer {{bearerToken}}'
 /v3/accounts:
   get:
    summary: Retrieve User Accounts Information
    description: This endpoint fetches the details of the user's accounts. It retrieves account
information like account type, account status, currency and other pertinent data relating to the
user's bank accounts. It helps in curating a consolidated view of the user's financial data
facilitating multifaceted financial operations and analysis.
    parameters:
     - $ref: '#/components/parameters/pageNumber'
     - $ref: '#/components/parameters/pageSize'
     - $ref: '#/components/parameters/from'
     - $ref: '#/components/parameters/to'
     - $ref: '#/components/parameters/type'
     - $ref: '#/components/parameters/x-provider-id'
    responses:
     '200':
       $ref: '#/components/responses/200GetAccountsV3'
     '400':
       $ref: '#/components/responses/400Error'
     '401':
       $ref: '#/components/responses/401Error'
     '403':
       $ref: '#/components/responses/403Error'
     '404':
       $ref: '#/components/responses/404Error'
     '405':
       $ref: '#/components/responses/405Error'
     '406':
       $ref: '#/components/responses/406Error'
     '429':
       $ref: '#/components/responses/429Error'
     '500':
       $ref: '#/components/responses/500Error'
    security:
     - OauthAccessTokenAuth: []
    operationId: getAccountsV3
    tags:
     - Accounts
    x-codeSamples:
     - lang: cURL
       source: |
         curl --location
'https://api.npck.kz/v3/accounts?pageNumber=1&pageSize=10&from=<dateTime>&to=<dateTi
me>&type=CURRENT_ACCOUNT' \
```

```
        --header 'x-provider-id: <uuid>' \
        --header 'Accept: application/json' \
        --header 'Authorization: Bearer {{bearerToken}}'
  /v3/accounts/{accountId}/balances:
    get:
      summary: Fetch Specified Account's Balance Information
      description: This endpoint is used to retrieve the current balance information for a specific
account associated with a user, denoted by the {accountId}. It returns key details such as
account's available balance, ledger balance, and if applicable, credit lines. The returned data
assists in providing an accurate snapshot of an account's financial status.
      parameters:
        - $ref: '#/components/parameters/accountId'
        - $ref: '#/components/parameters/x-provider-id'
      responses:
        '200':
          $ref: '#/components/responses/200GetAccountBalanceV3'
        '400':
          $ref: '#/components/responses/400Error'
        '401':
          $ref: '#/components/responses/401Error'
        '403':
          $ref: '#/components/responses/403Error'
        '404':
          $ref: '#/components/responses/404Error'
        '405':
          $ref: '#/components/responses/405Error'
        '406':
          $ref: '#/components/responses/406Error'
        '429':
          $ref: '#/components/responses/429Error'
        '500':
          $ref: '#/components/responses/500Error'
      security:
        - OauthAccessTokenAuth: []
      operationId: getAccountBalancesV3
      tags:
        - Accounts
      x-codeSamples:
        - lang: cURL
          source: |
            curl --location 'https://api.npck.kz/v3/accounts/<string>/balances' \
            --header 'x-provider-id: <uuid>' \
            --header 'Accept: application/json' \
            --header 'Authorization: Bearer {{bearerToken}}'
  /v3/accounts/{accountId}/transactions:
    get:
      summary: Retrieve Transactions for a Specific Account
      description: This endpoint fetches a list of transactions that have taken place on a specific
account, indicated by the {accountId}. It includes information about each transaction, such as the
transaction amount, date, description, and other relevant transaction data. This data can be used to
track financial activity, verify transactions, or generate account reports.
      parameters:
        - $ref: '#/components/parameters/accountId'
```

```
      - $ref: '#/components/parameters/pageNumber'
      - $ref: '#/components/parameters/pageSize'
      - $ref: '#/components/parameters/from'
      - $ref: '#/components/parameters/to'
      - $ref: '#/components/parameters/x-provider-id'
    responses:
      '200':
        $ref: '#/components/responses/200GetAccountTransactionsV3'
      '400':
        $ref: '#/components/responses/400Error'
      '401':
        $ref: '#/components/responses/401Error'
      '403':
        $ref: '#/components/responses/403Error'
      '404':
        $ref: '#/components/responses/404Error'
      '405':
        $ref: '#/components/responses/405Error'
      '406':
        $ref: '#/components/responses/406Error'
      '429':
        $ref: '#/components/responses/429Error'
      '500':
        $ref: '#/components/responses/500Error'
    security:
      - OauthAccessTokenAuth: []
    operationId: getAccountTransactionsV3
    tags:
      - Accounts
    x-codeSamples:
      - lang: cURL
        source: |
          curl --location
'https://api.npck.kz/v3/accounts/<string>/transactions?pageNumber=1&pageSize=10&from=<dat
eTime>&to=<dateTime>' \
          --header 'x-provider-id: <uuid>' \
          --header 'Accept: application/json' \
          --header 'Authorization: Bearer {{bearerToken}}'
  /v2/accounts:
    get:
      deprecated: true
      summary: Retrieve User Accounts Information
      description: Deprecated, use GET /v3/accounts. This endpoint fetches the details of the user's
accounts. It retrieves account information like account type, account status, currency and other
pertinent data relating to the user's bank accounts. It helps in curating a consolidated view of the
user's financial data facilitating multifaceted financial operations and analysis.
      parameters:
        - $ref: '#/components/parameters/pageNumber'
        - $ref: '#/components/parameters/pageSize'
        - $ref: '#/components/parameters/type'
        - $ref: '#/components/parameters/x-provider-id'
      responses:
        '200':
```

```
        $ref: '#/components/responses/200GetAccountsV2'
      '400':
        $ref: '#/components/responses/400Error'
      '401':
        $ref: '#/components/responses/401Error'
      '403':
        $ref: '#/components/responses/403Error'
      '404':
        $ref: '#/components/responses/404Error'
      '405':
        $ref: '#/components/responses/405Error'
      '406':
        $ref: '#/components/responses/406Error'
      '429':
        $ref: '#/components/responses/429Error'
      '500':
        $ref: '#/components/responses/500Error'
    security:
      - OauthAccessTokenAuth: []
    operationId: getAccountsV2
    tags:
      - Accounts
    x-codeSamples:
      - lang: cURL
        source: |
          curl --location
'https://api.npck.kz/v2/accounts?pageNumber=1&pageSize=10&type=CURRENT_ACCOUNT' \
          --header 'x-provider-id: <uuid>' \
          --header 'Accept: application/json' \
          --header 'Authorization: Bearer {{bearerToken}}'
  /v2/accounts/{accountId}/balances:
    get:
      deprecated: true
      summary: Fetch Specified Account's Balance Information
      description: Deprecated, use GET /v3/accounts/{accountId}/balances. This endpoint is used
to retrieve the current balance information for a specific account associated with a user, denoted
by the {accountId}. It returns key details such as account's available balance, ledger balance, and
if applicable, credit lines. The returned data assists in providing an accurate snapshot of an
account's financial status.
      parameters:
        - $ref: '#/components/parameters/accountId'
        - $ref: '#/components/parameters/x-provider-id'
      responses:
        '200':
          $ref: '#/components/responses/200GetAccountBalance'
        '400':
          $ref: '#/components/responses/400Error'
        '401':
          $ref: '#/components/responses/401Error'
        '403':
          $ref: '#/components/responses/403Error'
        '404':
          $ref: '#/components/responses/404Error'
```

```
      '405':
        $ref: '#/components/responses/405Error'
      '406':
        $ref: '#/components/responses/406Error'
      '429':
        $ref: '#/components/responses/429Error'
      '500':
        $ref: '#/components/responses/500Error'
    security:
      - OauthAccessTokenAuth: []
    operationId: getAccountBalancesV2
    tags:
      - Accounts
    x-codeSamples:
      - lang: cURL
        source: |
          curl --location 'https://api.npck.kz/v2/accounts/<string>/balances' \
          --header 'x-provider-id: <uuid>' \
          --header 'Accept: application/json' \
          --header 'Authorization: Bearer {{bearerToken}}'
  /v2/accounts/{accountId}/transactions:
    get:
      deprecated: true
      summary: Retrieve Transactions for a Specific Account
      description: Deprecated, use GET /v3/accounts/{accountId}/transactions. This endpoint
fetches a list of transactions that have taken place on a specific account, indicated by the
{accountId}. It includes information about each transaction, such as the transaction amount, date,
description, and other relevant transaction data. This data can be used to track financial activity,
verify transactions, or generate account reports.
      parameters:
        - $ref: '#/components/parameters/accountId'
        - $ref: '#/components/parameters/pageNumber'
        - $ref: '#/components/parameters/pageSize'
        - $ref: '#/components/parameters/fromCreateDateTime'
        - $ref: '#/components/parameters/toCreateDateTime'
        - $ref: '#/components/parameters/x-provider-id'
      responses:
        '200':
          $ref: '#/components/responses/200GetAccountTransactions'
        '400':
          $ref: '#/components/responses/400Error'
        '401':
          $ref: '#/components/responses/401Error'
        '403':
          $ref: '#/components/responses/403Error'
        '404':
          $ref: '#/components/responses/404Error'
        '405':
          $ref: '#/components/responses/405Error'
        '406':
          $ref: '#/components/responses/406Error'
        '429':
          $ref: '#/components/responses/429Error'
```

```
      '500':
        $ref: '#/components/responses/500Error'
    security:
      - OauthAccessTokenAuth: []
    operationId: getAccountTransactionsV2
    tags:
      - Accounts
    x-codeSamples:
      - lang: cURL
        source: |
          curl --location
'https://api.npck.kz/v2/accounts/<string>/transactions?pageNumber=1&pageSize=10&fromCreat
eDateTime=<dateTime>&toCreateDateTime=<dateTime>' \
          --header 'x-provider-id: <uuid>' \
          --header 'Accept: application/json' \
          --header 'Authorization: Bearer {{bearerToken}}'
  /v1/accounts:
    get:
      deprecated: true
      summary: Retrieve User Accounts Information
      description: Deprecated, use GET /v2/accounts
      parameters:
        - $ref: '#/components/parameters/pageNumber'
        - $ref: '#/components/parameters/pageSize'
        - $ref: '#/components/parameters/x-provider-id'
      responses:
        '200':
          $ref: '#/components/responses/200GetAccounts'
        '400':
          $ref: '#/components/responses/400Error'
        '401':
          $ref: '#/components/responses/401Error'
        '403':
          $ref: '#/components/responses/403Error'
        '404':
          $ref: '#/components/responses/404Error'
        '405':
          $ref: '#/components/responses/405Error'
        '406':
          $ref: '#/components/responses/406Error'
        '429':
          $ref: '#/components/responses/429Error'
        '500':
          $ref: '#/components/responses/500Error'
      security:
        - OauthAccessTokenAuth: []
      operationId: getAccountsV1
      tags:
        - Accounts
      x-codeSamples:
        - lang: cURL
          source: |
            curl --location 'https://api.npck.kz/v1/accounts?pageNumber=1&pageSize=10' \
```

```
        --header 'x-provider-id: <uuid>' \
        --header 'Accept: application/json' \
        --header 'Authorization: Bearer {{bearerToken}}'
/v1/accounts/{accountId}/balance:
  get:
    deprecated: true
    summary: Fetch Specified Account's Balance Information
    description: Deprecated, use GET /v2/accounts/{accountId}/balances
    parameters:
      - $ref: '#/components/parameters/accountId'
      - $ref: '#/components/parameters/x-provider-id'
    responses:
      '200':
        $ref: '#/components/responses/200GetAccountBalance'
      '400':
        $ref: '#/components/responses/400Error'
      '401':
        $ref: '#/components/responses/401Error'
      '403':
        $ref: '#/components/responses/403Error'
      '404':
        $ref: '#/components/responses/404Error'
      '405':
        $ref: '#/components/responses/405Error'
      '406':
        $ref: '#/components/responses/406Error'
      '429':
        $ref: '#/components/responses/429Error'
      '500':
        $ref: '#/components/responses/500Error'
    security:
      - OauthAccessTokenAuth: []
    operationId: getAccountBalanceV1
    tags:
      - Accounts
    x-codeSamples:
      - lang: cURL
        source: |
          curl --location 'https://api.npck.kz/v1/accounts/<string>/balance' \
          --header 'x-provider-id: <uuid>' \
          --header 'Accept: application/json' \
          --header 'Authorization: Bearer {{bearerToken}}'
/v1/accounts/{accountId}/transactions:
  get:
    deprecated: true
    summary: Retrieve Transactions for a Specific Account
    description: Deprecated, GET /v2/accounts/{accountId}/transactions
    parameters:
      - $ref: '#/components/parameters/accountId'
      - $ref: '#/components/parameters/pageNumber'
      - $ref: '#/components/parameters/pageSize'
      - $ref: '#/components/parameters/fromCreateDateTime'
      - $ref: '#/components/parameters/toCreateDateTime'
```

```
      - $ref: '#/components/parameters/x-provider-id'
    responses:
      '200':
        $ref: '#/components/responses/200GetAccountTransactions'
      '400':
        $ref: '#/components/responses/400Error'
      '401':
        $ref: '#/components/responses/401Error'
      '403':
        $ref: '#/components/responses/403Error'
      '404':
        $ref: '#/components/responses/404Error'
      '405':
        $ref: '#/components/responses/405Error'
      '406':
        $ref: '#/components/responses/406Error'
      '429':
        $ref: '#/components/responses/429Error'
      '500':
        $ref: '#/components/responses/500Error'
    security:
      - OauthAccessTokenAuth: []
    operationId: getAccountTransactionsV1
    tags:
      - Accounts
    x-codeSamples:
      - lang: cURL
        source: |
          curl --location
'https://api.npck.kz/v1/accounts/<string>/transactions?pageNumber=1&pageSize=10&fromCreat
eDateTime=<dateTime>&toCreateDateTime=<dateTime>' \
          --header 'x-provider-id: <uuid>' \
          --header 'Accept: application/json' \
          --header 'Authorization: Bearer {{bearerToken}}'
  /providers/v1/accounts:
    get:
      deprecated: true
      summary: Retrieve User Accounts Information
      description: Deprecated, use GET /v2/accounts
      parameters:
        - $ref: '#/components/parameters/pageNumber'
        - $ref: '#/components/parameters/pageSize'
        - $ref: '#/components/parameters/x-provider-id'
      responses:
        '200':
          $ref: '#/components/responses/200GetAccounts'
        '400':
          $ref: '#/components/responses/400Error'
        '401':
          $ref: '#/components/responses/401Error'
        '403':
          $ref: '#/components/responses/403Error'
        '404':
```

```
        $ref: '#/components/responses/404Error'
      '405':
        $ref: '#/components/responses/405Error'
      '406':
        $ref: '#/components/responses/406Error'
      '429':
        $ref: '#/components/responses/429Error'
      '500':
        $ref: '#/components/responses/500Error'
    security:
      - OauthAccessTokenAuth: []
    operationId: getAccountsDeprecated
    tags:
      - Accounts
    x-codeSamples:
      - lang: cURL
        source: |
          curl --location 'https://api.npck.kz/providers/v1/accounts?pageNumber=1&pageSize=10' \
          --header 'x-provider-id: <uuid>' \
          --header 'Accept: application/json' \
          --header 'Authorization: Bearer {{bearerToken}}'
/providers/v1/accounts/{accountId}/balance:
  get:
    deprecated: true
    summary: Fetch Specified Account's Balance Information
    description: Deprecated, use GET /v2/accounts/{accountId}/balances
    parameters:
      - $ref: '#/components/parameters/accountId'
      - $ref: '#/components/parameters/x-provider-id'
    responses:
      '200':
        $ref: '#/components/responses/200GetAccountBalance'
      '400':
        $ref: '#/components/responses/400Error'
      '401':
        $ref: '#/components/responses/401Error'
      '403':
        $ref: '#/components/responses/403Error'
      '404':
        $ref: '#/components/responses/404Error'
      '405':
        $ref: '#/components/responses/405Error'
      '406':
        $ref: '#/components/responses/406Error'
      '429':
        $ref: '#/components/responses/429Error'
      '500':
        $ref: '#/components/responses/500Error'
    security:
      - OauthAccessTokenAuth: []
    operationId: getAccountBalanceDeprecated
    tags:
      - Accounts
```

```
    x-codeSamples:
      - lang: cURL
        source: |
          curl --location 'https://api.npck.kz/providers/v1/accounts/<string>/balance' \
          --header 'x-provider-id: <uuid>' \
          --header 'Accept: application/json' \
          --header 'Authorization: Bearer {{bearerToken}}'
  /providers/v1/accounts/{accountId}/transactions:
    get:
      deprecated: true
      summary: Retrieve Transactions for a Specific Account
      description: Deprecated, GET /v2/accounts/{accountId}/transactions
      parameters:
        - $ref: '#/components/parameters/accountId'
        - $ref: '#/components/parameters/pageNumber'
        - $ref: '#/components/parameters/pageSize'
        - $ref: '#/components/parameters/fromCreateDateTime'
        - $ref: '#/components/parameters/toCreateDateTime'
        - $ref: '#/components/parameters/x-provider-id'
      responses:
        '200':
          $ref: '#/components/responses/200GetAccountTransactions'
        '400':
          $ref: '#/components/responses/400Error'
        '401':
          $ref: '#/components/responses/401Error'
        '403':
          $ref: '#/components/responses/403Error'
        '404':
          $ref: '#/components/responses/404Error'
        '405':
          $ref: '#/components/responses/405Error'
        '406':
          $ref: '#/components/responses/406Error'
        '429':
          $ref: '#/components/responses/429Error'
        '500':
          $ref: '#/components/responses/500Error'
      security:
        - OauthAccessTokenAuth: []
      operationId: getAccountTransactionsDeprecated
      tags:
        - Accounts
      x-codeSamples:
        - lang: cURL
          source: |
            curl --location
'https://api.npck.kz/providers/v1/accounts/<string>/transactions?pageNumber=1&pageSize=10&f
romCreateDateTime=<dateTime>&toCreateDateTime=<dateTime>' \
            --header 'x-provider-id: <uuid>' \
            --header 'Accept: application/json' \
            --header 'Authorization: Bearer {{bearerToken}}'
components:
```

```
  parameters:
    x-provider-id:
      in: header
      name: x-provider-id
      schema:
        type: string
        format: uuid
      required: true
      description: This is a mandatory header parameter representing a unique identifier assigned to each API provider in the Open Banking ecosystem. It enables the facilitation of secure interactions between various providers and the Open Banking platform, ensuring that each request can be correctly attributed to its originating provider.
    type:
      name: type
      in: query
      required: false
      description: This parameter allows the client to filter the accounts based on their type. By specifying the type, the client can narrow down the search to a specific category of accounts, such as current accounts, credit cards, debit cards, or savings accounts. This parameter is optional, and if not provided, the API returns information about all available accounts, including current and card accounts.
      schema:
        $ref: '#/components/schemas/AccountType'
      style: form
      explode: true
    pageNumber:
      name: pageNumber
      in: query
      required: false
      description: This parameter is instrumental in navigating through the paginated data responses returned by the API. It represents the current page number within the total number of pages that are present in the complete data set. By specifying pageNumber, the client directs the API to return only the data corresponding to that particular page number.
      schema:
        type: integer
        default: 1
        minimum: 1
    pageSize:
      name: pageSize
      in: query
      required: false
      description: This parameter designates the number of data entries that the API returns in a single response. By defining the pageSize, the client can control the granularity of data retrieved in each API call, essentially deciding the amount of data presented on each page of the paginated response. Utilising this parameter allows for more efficient data management and enhances response time, allowing clients to strike a balance between the depth of information retrieved and the load on system resources.
      schema:
        type: integer
        default: 10
        minimum: 1
        maximum: 100
    accountId:
```

```
    name: accountId
    in: path
    required: true
    description: This parameter represents a unique identifier corresponding to a specific account that the user holds. By supplying the accountId as part of the API request, the client can target operations specifically to this account, whether it's retrieving account details, updating information, or querying transaction history.
    schema:
     type: string
     maxLength: 36
  iban:
    name: iban
    in: path
    required: true
    description: This parameter represents a unique identifier corresponding to a specific account that the user holds (IBAN). By supplying the accountId as part of the API request, the client can target operations specifically to this account, whether it's retrieving account details, updating information, or querying transaction history.
    schema:
     type: string
     maxLength: 34
  fromCreateDateTime:
    name: fromCreateDateTime
    in: query
    required: false
    description: This parameter qualifies the starting point of the time frame for which the client seeks data. It signifies the exact date and time, expressed in ISO 8601 format, from when the requested data set should start. Primarily used in conjunction with toCreateDateTime, it allows for narrowing down the scope of data retrieval to a specific date and time range, thereby offering efficient and tailored access to system data.
    schema:
     allOf:
      - $ref: '#/components/schemas/ISODateTime'
  toCreateDateTime:
    name: toCreateDateTime
    in: query
    required: false
    description: This parameter signifies the end point of the desired timeframe for data retrieval. It specifies the precise date and time, in ISO 8601 format, up to which the client wishes to receive data. When used with its companion parameter fromCreateDateTime, it allows for specifying the exact time window for the requested data. This targeted approach ensures efficient retrieval of appropriate data, thus optimizing system performance.
    schema:
     allOf:
      - $ref: '#/components/schemas/ISODateTime'
  from:
    name: from
    in: query
    required: false
    description: This parameter qualifies the starting point of the time frame for which the client seeks data. It signifies the exact date and time, expressed in ISO 8601 format, from when the requested data set should start. Primarily used in conjunction with 'to', it allows for narrowing
```

```
down the scope of data retrieval to a specific date and time range, thereby offering efficient and tailored access to system data.
      schema:
        allOf:
          - $ref: '#/components/schemas/ISODateTime'
    to:
      name: to
      in: query
      required: false
      description: This parameter signifies the end point of the desired timeframe for data retrieval. It specifies the precise date and time, in ISO 8601 format, up to which the client wishes to receive data. When used with its companion parameter 'from', it allows for specifying the exact time window for the requested data. This targeted approach ensures efficient retrieval of appropriate data, thus optimizing system performance.
      schema:
        allOf:
          - $ref: '#/components/schemas/ISODateTime'
  schemas:
    GetAccountsResponse:
      type: object
      description: This object encapsulates the response returned by the API when a successful request is made to fetch the details of the user's accounts.
      additionalProperties: false
      properties:
        data:
          allOf:
            - $ref: '#/components/schemas/AccountResponse'
        page:
          allOf:
            - $ref: '#/components/schemas/Page'
      required:
        - data
        - page
    AccountResponse:
      type: object
      description: This object represents the detailed information about the user's accounts, returned as a response by the API when queried for account details.
      additionalProperties: false
      properties:
        accounts:
          type: array
          items:
            allOf:
              - $ref: '#/components/schemas/Account'
          minItems: 0
      required:
        - accounts
    Account:
      type: object
      description: This object encapsulates the detailed information around a single bank account held by the user.
      additionalProperties: false
      properties:
```

```
      accountId:
        description: Bank account identifier
        type: string
        maxLength: 36
      currentBalance:
        description: This integer property represents the current balance of the account. It's expressed in the smallest currency unit (e.g., cents for USD, kopecks for RUB, tiyn for KZT). For instance, a value of 10001 represents 100.01 in the actual currency, and 10000 represents 100.00.
        type: integer
        format: int64
      currency:
        description: This string property represents the currency in which the account is held. It complies with the ISO 4217 standard. The format requires three uppercase letters (for example, 'USD' for United States Dollar, 'KZT' for Kazakhstani Tenge).
        type: string
        pattern: ^[A-Z]{3,3}$
        example: KZT
      openedDateTime:
        description: This property indicates the date and time when the account was first opened. The value adheres to the ISO 8601 standards for date and time formatting. It includes both date and time components alongside the timezone information, as shown in the example '2023-03-03T13:23:13+06:00'. This property provides insight into the account's age and status.
        allOf:
          - $ref: '#/components/schemas/ISODateTime'
        example: '2023-03-03T13:23:13+06:00'
      description:
        description: This optional string property provides a human-readable description for the account. This could contain any relevant details about the account as determined by the account provider; such as the account's purpose, its type (e.g. "Savings Account", "Business Account"), or other significant identifiers that the user can associate with. It can extend up to a maximum length of 70 characters.
        type: string
        maxLength: 70
    required:
      - accountId
      - currency
      - currentBalance
      - openedDateTime
  GetAccountsResponseV2:
    type: object
    description: This object encapsulates the response returned by the API when a successful request is made to fetch the details of the user's accounts.
    additionalProperties: false
    properties:
      data:
        $ref: '#/components/schemas/AccountResponseV2'
      page:
        $ref: '#/components/schemas/Page'
    required:
      - data
      - page
  GetAccountsResponseV3:
    type: object
```

```
    description: This object encapsulates the response returned by the API when a successful request is made to fetch the details of the user's accounts.
    additionalProperties: false
    properties:
      data:
        $ref: '#/components/schemas/AccountResponseV3'
      page:
        $ref: '#/components/schemas/Page'
    required:
      - data
      - page
  AccountResponseV2:
    type: object
    additionalProperties: false
    properties:
      accounts:
        type: array
        items:
          $ref: '#/components/schemas/AccountV2'
        minItems: 0
    required:
      - accounts
  AccountResponseV3:
    type: object
    additionalProperties: false
    properties:
      accounts:
        type: array
        items:
          $ref: '#/components/schemas/AccountV3'
        minItems: 0
    required:
      - accounts
  AccountV2:
    type: object
    description: This object encapsulates the detailed information about a single bank account held by the user.
    additionalProperties: false
    properties:
      accountId:
        description: Bank account identifier
        type: string
        maxLength: 36
      currentBalance:
        description: This integer property represents the current balance of the account. It's expressed in the smallest currency unit (e.g., cents for USD, kopecks for RUB, tiyn for KZT). For instance, a value of 10001 represents 100.01 in the actual currency, and 10000 represents 100.00.
        type: integer
        format: int64
      currency:
        description: This string property represents the currency in which the account is held. It complies with the ISO 4217 standard. The format requires three uppercase letters (for example, 'USD' for United States Dollar, 'KZT' for Kazakhstani Tenge).
```

```
        type: string
        pattern: ^[A-Z]{3,3}$
        example: KZT
      openedDateTime:
        description: This property indicates the date and time when the account was first opened. The value adheres to the ISO 8601 standards for date and time formatting. It includes both date and time components alongside the timezone information, as shown in the example '2023-03-03T13:23:13+06:00'. This property provides insight into the account's age and status.
        allOf:
          - $ref: '#/components/schemas/ISODateTime'
        example: '2023-03-03T13:23:13+06:00'
      description:
        description: This optional string property provides a human-readable description for the account. This could contain any relevant details about the account as determined by the account provider; such as the account's purpose, its type (e.g. "Savings Account", "Business Account"), or other significant identifiers that the user can associate with. It can extend up to a maximum length of 70 characters.
        type: string
        maxLength: 70
      type:
        description: Account type
        allOf:
          - $ref: '#/components/schemas/AccountType'
      maskedNumber:
        description: For current account, deposit - the last 4 digits of the IBAN account number; for card accounts - the last 4 digits of the card number.
        type: string
        maxLength: 4
        minLength: 4
      altAccountId:
        description: Alternative account number. It can be used to pass additional account identifier information if necessary.
        type: string
        maxLength: 40
        minLength: 1
    required:
      - accountId
      - currency
      - currentBalance
      - openedDateTime
      - type
      - maskedNumber
  AccountV3:
    type: object
    description: This object encapsulates the detailed information about a single bank account held by the user.
    additionalProperties: false
    properties:
      accountId:
        description: Bank account identifier
        type: string
        maxLength: 36
      currentBalance:
```

```
        description: This integer property represents the current balance of the account. It's expressed in the smallest currency unit (e.g., cents for USD, kopecks for RUB, tiyn for KZT). For instance, a value of 10001 represents 100.01 in the actual currency, and 10000 represents 100.00.
        type: integer
        format: int64
      currency:
        description: This string property represents the currency in which the account is held. It complies with the ISO 4217 standard. The format requires three uppercase letters (for example, 'USD' for United States Dollar, 'KZT' for Kazakhstani Tenge).
        type: string
        pattern: ^[A-Z]{3,3}$
        example: KZT
      openedDateTime:
        description: This property indicates the date and time when the account was first opened. The value adheres to the ISO 8601 standards for date and time formatting. It includes both date and time components alongside the timezone information, as shown in the example '2023-03-03T13:23:13+06:00'. This property provides insight into the account's age and status.
        allOf:
          - $ref: '#/components/schemas/ISODateTime'
        example: '2023-03-03T13:23:13+06:00'
      description:
        description: This optional string property provides a human-readable description for the account. This could contain any relevant details about the account as determined by the account provider; such as the account's purpose, its type (e.g. "Savings Account", "Business Account"), or other significant identifiers that the user can associate with. It can extend up to a maximum length of 70 characters.
        type: string
        maxLength: 70
      type:
        description: Account type
        allOf:
          - $ref: '#/components/schemas/AccountType'
      maskedNumber:
        description: For current account, deposit - the last 4 digits of the IBAN account number; for card accounts - the last 4 digits of the card number.
        type: string
        maxLength: 4
        minLength: 4
      altAccountId:
        description: Alternative account number. It can be used to pass additional account identifier information if necessary.
        type: string
        maxLength: 40
        minLength: 1
    required:
      - accountId
      - currency
      - currentBalance
      - openedDateTime
      - type
      - maskedNumber
  AccountType:
    description: This object represents the type of account.
```

```
      type: string
      enum:
        - CURRENT_ACCOUNT
        - CREDIT_CARD
        - DEBIT_CARD
        - SAVINGS
    GetAccountBalanceResponse:
      type: object
      description: This object represents the response returned by the API upon successful request
for the balance of a specific account.
      additionalProperties: false
      properties:
        data:
          allOf:
            - $ref: '#/components/schemas/BalanceResponse'
      required:
        - data
    GetAccountBalanceResponseV3:
      type: object
      description: This object represents the response returned by the API upon successful request
for the balance of a specific account.
      additionalProperties: false
      properties:
        data:
          allOf:
            - $ref: '#/components/schemas/BalanceResponseV3'
      required:
        - data
    BalanceResponse:
      type: object
      description: This object encapsulates the detailed information related to the balance of a
specific account, including current, available, and blocked balance amounts.
      additionalProperties: false
      properties:
        currentBalance:
          description: This integer property represents the current balance of the account in the
smallest currency unit (e.g., cents for USD, kopecks for RUB, tiyn for KZT). For instance, a
value of 10001 represents 100.01 in the actual currency, and 10000 represents 100.00.
          type: integer
          format: int64
        availableBalance:
          description: Similar to currentBalance, this integer property expresses the balance amount
currently available in the account for transactions or transfers in the smallest currency unit.
          type: integer
          format: int64
        blockedBalance:
          description: This property denotes the amount from the total balance that is currently
blocked or held and cannot be used for transactions. Like other balance fields, it is expressed in
the smallest unit of the relevant currency.
          type: integer
          format: int64
        currency:
```

description: This string property represents the currency in which the account is held, adhering to the ISO 4217 standard. The value consists of three uppercase letters (e.g., 'USD' for United States Dollar, 'KZT' for Kazakhstani Tenge).
type: string
pattern: ^[A-Z]{3,3}$
example: KZT
creditLine:
type: array
items:
allOf:
- $ref: '#/components/schemas/CreditLine'
description: This is an array of objects where each object encapsulates detailed information about a credit line associated with the account. Each object follows the CreditLine schema.
purses:
type: array
items:
allOf:
- $ref: '#/components/schemas/Amount'
description: This optional property is an array of objects that represent account balances in other currencies. Each item in the array reports balance in a specific foreign currency, following the structure dictated by the Amount schema. It extends the response by providing balance details in multiple currencies.
required:
- currentBalance
- availableBalance
- currency
BalanceResponseV3:
type: object
description: This object encapsulates the detailed information related to the balance of a specific account, including current, available, and blocked balance amounts.
additionalProperties: false
properties:
currentBalance:
description: This integer property represents the current balance of the account in the smallest currency unit (e.g., cents for USD, kopecks for RUB, tiyn for KZT). For instance, a value of 10001 represents 100.01 in the actual currency, and 10000 represents 100.00.
type: integer
format: int64
availableBalance:
description: Similar to currentBalance, this integer property expresses the balance amount currently available in the account for transactions or transfers in the smallest currency unit.
type: integer
format: int64
blockedBalance:
description: This property denotes the amount from the total balance that is currently blocked or held and cannot be used for transactions. Like other balance fields, it is expressed in the smallest unit of the relevant currency.
type: integer
format: int64
currency:
description: This string property represents the currency in which the account is held, adhering to the ISO 4217 standard. The value consists of three uppercase letters (e.g., 'USD' for United States Dollar, 'KZT' for Kazakhstani Tenge).

```
        type: string
        pattern: ^[A-Z]{3,3}$
        example: KZT
      creditLine:
        type: array
        items:
          allOf:
            - $ref: '#/components/schemas/CreditLine'
        description: This is an array of objects where each object encapsulates detailed information about a credit line associated with the account. Each object follows the CreditLine schema.
      purses:
        type: array
        items:
          allOf:
            - $ref: '#/components/schemas/Amount'
        description: This optional property is an array of objects that represent account balances in other currencies. Each item in the array reports balance in a specific foreign currency, following the structure dictated by the Amount schema. It extends the response by providing balance details in multiple currencies.
    required:
      - currentBalance
      - availableBalance
      - currency
  CreditLine:
    type: object
    description: This object holds detailed information about a credit line associated with the account.
    additionalProperties: false
    properties:
      included:
        description: This boolean property indicates whether the credit line is included in the total account balance. A true value means the credit line value is factored in the account balance while false means it's not.
        type: boolean
      type:
        description: This property signifies the type of the credit limit, determined by the CreditLineTypeCode schema. It could represent various kinds of credit arrangements like overdraft, credit card limit, personal loan etc.
        allOf:
          - $ref: '#/components/schemas/CreditLineTypeCode'
      amount:
        description: This property provides information about the credit line amount and the currency in which it is expressed.
        allOf:
          - $ref: '#/components/schemas/Amount'
    required:
      - included
      - type
      - amount
  Amount:
    type: object
    description: This object encapsulates the information about a monetary amount along with its currency.
```

```
    additionalProperties: false
    properties:
      amount:
        description: This integer property represents a specific amount of money, denoted in the smallest unit of the relevant currency (e.g., cents for USD, kopecks for RUB, tiyn for KZT). For example, a value of 10001 represents 100.01 in the actual currency, and 10000 represents 100.00.
        type: integer
        format: int64
      currency:
        description: This string property represents the currency in which the amount is expressed. It is represented according to the ISO 4217 standard, which uses three uppercase letters (e.g., 'USD' for United States Dollar, 'KZT' for Kazakhstani Tenge).
        type: string
        pattern: ^[A-Z]{3,3}$
        example: KZT
    required:
      - amount
      - currency
  GetAccountTransactionsResponse:
    type: object
    description: This object represents the response returned by the API upon a successful request for the transaction history of a specified account.
    additionalProperties: false
    properties:
      data:
        allOf:
          - $ref: '#/components/schemas/TransactionResponse'
      page:
        allOf:
          - $ref: '#/components/schemas/Page'
    required:
      - data
      - page
  GetAccountTransactionsResponseV3:
    type: object
    description: This object represents the response returned by the API upon a successful request for the transaction history of a specified account.
    additionalProperties: false
    properties:
      data:
        allOf:
          - $ref: '#/components/schemas/TransactionResponseV3'
      page:
        allOf:
          - $ref: '#/components/schemas/Page'
    required:
      - data
      - page
  TransactionResponse:
    type: object
    description: This object encapsulates the detailed information related to the transactions of a specific account.
    additionalProperties: false
```

```
    properties:
      transactions:
        type: array
        items:
          allOf:
            - $ref: '#/components/schemas/Transaction'
        minItems: 0
    required:
      - transactions
  TransactionResponseV3:
    type: object
    description: This object encapsulates the detailed information related to the transactions of a
specific account.
    additionalProperties: false
    properties:
      transactions:
        type: array
        items:
          allOf:
            - $ref: '#/components/schemas/TransactionV3'
        minItems: 0
    required:
      - transactions
  Transaction:
    type: object
    description: This object describes a single financial transaction carried out on an account.
    additionalProperties: false
    properties:
      transactionId:
        description: A unique string identifier assigned to a transaction.
        type: string
        maxLength: 36
      documentNumber:
        description: This string property represents the payment document number associated with
the transaction.
        type: string
        maxLength: 70
      reference:
        description: This field holds a reference string for the transaction, which can be used to
cross-reference and track the transaction.
        type: string
        maxLength: 36
      status:
        description: This property represents the current processing status of the transaction. The
structure and possible values of status are defined by the TransactionStatusCode schema.
        allOf:
          - $ref: '#/components/schemas/TransactionStatusCode'
      amount:
        description: This property provides information about the transaction amount along with
currency. The structure for this data is defined by the Amount schema.
        allOf:
          - $ref: '#/components/schemas/Amount'
      chargeAmount:
```

```
      description: This field denotes the commission charged for the transaction, represented in
the same format as amount.
      allOf:
       - $ref: '#/components/schemas/Amount'
    creditDebitIndicator:
      description: This property indicates whether the transaction is a debit or a credit to the
account. The possible values for this field are defined by the CreditDebitCode schema.
      allOf:
       - $ref: '#/components/schemas/CreditDebitCode'
    type:
      description: This property represents the operation type of the transaction as defined by the
TransactionTypeCode schema.
      allOf:
       - $ref: '#/components/schemas/TransactionTypeCode'
    purposeCode:
      description: Payment purpose code. The value must comply with the rules approved by the
Resolution of the Board of the National Bank of the Republic of Kazakhstan dated August 31,
2016 No. 203 "On approval of the Rules for the use of economic sector codes and payment
purposes".
      type: string
      minLength: 3
      maxLength: 3
    budgetCode:
      description: Budget classification code (budget revenue classification). The values used are
defined in the Order of the Minister of Finance of the Republic of Kazakhstan dated September
18, 2014 No. 403 "Some issues of the Unified Budget Classification of the Republic of
Kazakhstan".
      type: string
      minLength: 6
      maxLength: 6
    senderCode:
      description: Sender's money code. The value should comply with the rules approved by the
Resolution of the Board of the National Bank of the Republic of Kazakhstan dated August 31,
2016 No. 203 "On approving the Rules for the application of economic sector codes and payment
purposes.
      type: string
      minLength: 2
      maxLength: 2
    beneficiaryCode:
      description: Beneficiary code. The value should comply with the rules approved by the
Resolution of the Board of the National Bank of the Republic of Kazakhstan dated August 31,
2016 No. 203 "On approving the Rules for the application of economic sector codes and payment
purposes.
      type: string
      minLength: 2
      maxLength: 2
    mcc:
      description: This string property represents the Merchant Category Code, a four-digit
number that indicates the merchant's type of business.
      type: string
      minLength: 4
      maxLength: 4
    description:
```

```
        description: This string field provides a human-readable description of the transaction.
        type: string
        maxLength: 70
      createDateTime:
        description: Date and time of the operation. ISO 8601 standard is used.
        allOf:
          - $ref: '#/components/schemas/ISODateTime'
        example: '2023-03-03T13:23:13+06:00'
      bookingDateTime:
        description: Date and time of transaction execution, i.e. when the transaction is carried out
and becomes immutable. ISO 8601 standard is used.
        allOf:
          - $ref: '#/components/schemas/ISODateTime'
        example: '2023-03-03T13:23:13+06:00'
      valueDateTime:
        description: Date and time of valuation. ISO 8601 standard is used.
        allOf:
          - $ref: '#/components/schemas/ISODateTime'
        example: '2023-03-03T13:23:13+06:00'
      creditorName:
        description: Name of the beneficiary in case of a debit transaction.
        type: string
        maxLength: 70
      debtorName:
        description: Name of the payer in case of a credit transaction.
        type: string
        maxLength: 70
      creditorAgent:
        description: Financial organization servicing the recipient's account.
        allOf:
          - $ref: '#/components/schemas/FinancialInstitutionIdentificationType'
      debtorAgent:
        description: Financial organization servicing the payer's account.
        allOf:
          - $ref: '#/components/schemas/FinancialInstitutionIdentificationType'
    required:
      - transactionId
      - reference
      - status
      - amount
      - creditDebitIndicator
      - type
      - createDateTime
  TransactionV3:
    type: object
    description: This object describes a single financial transaction carried out on an account.
    additionalProperties: false
    properties:
      transactionId:
        description: A unique string identifier assigned to a transaction.
        type: string
        maxLength: 36
      documentNumber:
```

description: This string property represents the payment document number associated with the transaction.
type: string
maxLength: 70
reference:
description: This field holds a reference string for the transaction, which can be used to cross-reference and track the transaction.
type: string
maxLength: 36
status:
description: This property represents the current processing status of the transaction. The structure and possible values of status are defined by the TransactionStatusCode schema.
allOf:
- $ref: '#/components/schemas/TransactionStatusCode'
amount:
description: This property provides information about the transaction amount along with currency. The structure for this data is defined by the Amount schema.
allOf:
- $ref: '#/components/schemas/Amount'
chargeAmount:
description: This field denotes the commission charged for the transaction, represented in the same format as amount.
allOf:
- $ref: '#/components/schemas/Amount'
creditDebitIndicator:
description: This property indicates whether the transaction is a debit or a credit to the account. The possible values for this field are defined by the CreditDebitCode schema.
allOf:
- $ref: '#/components/schemas/CreditDebitCode'
type:
description: This property represents the operation type of the transaction as defined by the TransactionTypeCode schema.
allOf:
- $ref: '#/components/schemas/TransactionTypeCode'
purposeCode:
description: Payment purpose code. The value must comply with the rules approved by the Resolution of the Board of the National Bank of the Republic of Kazakhstan dated August 31, 2016 No. 203 "On approval of the Rules for the use of economic sector codes and payment purposes".
type: string
minLength: 3
maxLength: 3
budgetCode:
description: Budget classification code (budget revenue classification). The values used are defined in the Order of the Minister of Finance of the Republic of Kazakhstan dated September 18, 2014 No. 403 "Some issues of the Unified Budget Classification of the Republic of Kazakhstan".
type: string
minLength: 6
maxLength: 6
senderCode:
description: Sender's money code. The value should comply with the rules approved by the Resolution of the Board of the National Bank of the Republic of Kazakhstan dated August 31,

2016 No. 203 "On approving the Rules for the application of economic sector codes and payment purposes.
type: string
minLength: 2
maxLength: 2
beneficiaryCode:
description: Beneficiary code. The value should comply with the rules approved by the Resolution of the Board of the National Bank of the Republic of Kazakhstan dated August 31, 2016 No. 203 "On approving the Rules for the application of economic sector codes and payment purposes.
type: string
minLength: 2
maxLength: 2
mcc:
description: This string property represents the Merchant Category Code, a four-digit number that indicates the merchant's type of business.
type: string
minLength: 4
maxLength: 4
description:
description: This string field provides a human-readable description of the transaction.
type: string
maxLength: 70
createDateTime:
description: Date and time of the operation. ISO 8601 standard is used.
allOf:
- $ref: '#/components/schemas/ISODateTime'
example: '2023-03-03T13:23:13+06:00'
bookingDateTime:
description: Date and time of transaction execution, i.e. when the transaction is carried out and becomes immutable. ISO 8601 standard is used.
allOf:
- $ref: '#/components/schemas/ISODateTime'
example: '2023-03-03T13:23:13+06:00'
valueDateTime:
description: Date and time of valuation. ISO 8601 standard is used.
allOf:
- $ref: '#/components/schemas/ISODateTime'
example: '2023-03-03T13:23:13+06:00'
creditorName:
description: Name of the beneficiary in case of a debit transaction.
type: string
maxLength: 70
debtorName:
description: Name of the payer in case of a credit transaction.
type: string
maxLength: 70
creditorAgent:
description: Financial organization servicing the recipient's account.
allOf:
- $ref: '#/components/schemas/FinancialInstitutionIdentificationType'
debtorAgent:
description: Financial organization servicing the payer's account.

```
      allOf:
        - $ref: '#/components/schemas/FinancialInstitutionIdentificationType'
      rrn:
        description: Reference Retrieval Number of the transaction
        type: string
        maxLength: 30
      iinSender:
        description: Sha512 HEX string of the Individual Identification Number of the sender
        type: string
        pattern: ^[0-9A-Fa-f]{128}$
      iinReceiver:
        description: Sha512 HEX string of the Individual Identification Number of the receiver
        type: string
        pattern: ^[0-9A-Fa-f]{128}$
    required:
      - transactionId
      - reference
      - status
      - amount
      - creditDebitIndicator
      - type
      - createDateTime
  FinancialInstitutionIdentificationType:
    type: object
    description: This object contains identifiers used to uniquely identify a financial institution
(like a bank).
    additionalProperties: false
    properties:
      bic:
        description: his string property holds the Bank Identification Code (BIC), an international
standard for identifying banks around the world. BICs are often referred to as "SWIFT codes"
and are always 8 characters long.
        type: string
        minLength: 8
        maxLength: 8
      bin:
        description: This string property holds the Business Identification Number (BIN). This is a
unique identifier issued to a business operating within a certain jurisdiction, and in this context,
likely refers to the specific BIN assigned to the financial institution. BINs are typically 12
characters long.
        type: string
        minLength: 12
        maxLength: 12
      name:
        description: This string field holds the name of the financial institution, such as "Chase
Bank" or "Bank of America". Note that this field cannot exceed 256 characters in length.
        type: string
        maxLength: 256
    required:
      - bic
  Page:
    type: object
```

```
    description: This object represents pagination details in the context of an API response, providing information about the total number of items and whether the current page is the last one.
    additionalProperties: false
    properties:
      totalItems:
        type: integer
        format: int64
        description: This integer property specifies the total number of items that match the query parameters, providing a total count of the relevant data entities available.
      isLastPage:
        description: This boolean property serves as a flag to indicate whether the returned page is the last one in the set of paged data. A true value means that current page is the last, while false implies more pages are available.
        type: boolean
    required:
      - totalItems
      - isLastPage
  ISODateTime:
    type: string
    description: This data type represents a date and time string that conforms to the ISO 8601 standard format.
    format: date-time
    example: 2021-06-05T15:15:13%2b00:00
  CreditLineTypeCode:
    description: This data type indicates the type of credit limit associated with an account. It is represented as a string, with its possible values defined in an enumeration.
    type: string
    enum:
      - AVAILABLE
      - PRE_AGREED
    example: AVAILABLE
  TransactionStatusCode:
    description: This data type represents the processing status of a transaction. It is string based and the possible values are defined in an enumeration.
    type: string
    enum:
      - BOOKED
      - PENDING
  CreditDebitCode:
    description: This data type indicates whether a transaction is a credit (adds funds) or a debit (removes funds) to an account. It is represented as a string and its possible values are defined in an enumeration.
    type: string
    enum:
      - CREDIT
      - DEBIT
  TransactionTypeCode:
    description: This data type represents the type of banking operation associated with a transaction. It is represented as a string, and the possible values are defined in an enumeration.
    type: string
    enum:
      - INCOME
```

```
      - WITHDRAWAL
      - TRANSFER
      - PURCHASE
      - E_COMMERCE
      - PAYMENT
      - OTHER
      - BONUS
  AccountExists:
    description: This data type represents the verification status of an account. It is represented as
a boolean, indicating whether the account exists.
    type: object
    additionalProperties: false
    properties:
      exists:
        type: boolean
        description: This boolean property indicates whether the account exists in the bank or not.
  ErrorCode400:
    description: |
      Error codes:
      - `FIELD_MISSING` - No field or request body provided.
      - `FIELD_INVALID` - Invalid or incorrect input data provided.
      - `HEADER_MISSING` - Required header not provided.
      - `HEADER_INVALID` - Invalid header provided.
      - `RESOURCE_NOT_FOUND` - Not used, kept for backward compatibility.
    type: string
    enum:
      - FIELD_MISSING
      - FIELD_INVALID
      - HEADER_MISSING
      - HEADER_INVALID
      - RESOURCE_NOT_FOUND
  ErrorResponse400:
    type: object
    description: This object represents the standard HTTP 400 Bad Request error response. It is
used when the client sends a request that the server could not understand, such as invalid syntax,
invalid request message framing, or deceptive request routing. The body of this response message
contains a more specific description of the error cause.
    additionalProperties: false
    properties:
      requestId:
        description: This optional parameter serves to uniquely identify the HTTP request. This
requestId is needed when you contact support.
        type: string
      code:
        description: This represents the unique error code corresponding to the specific error
occurrence.
        allOf:
          - $ref: '#/components/schemas/ErrorCode400'
      description:
        type: string
        maxLength: 250
        description: This provides a human-readable explanation of the error.
    required:
```

```
      - code
      - description
    RequestIdErrorResponse:
      type: object
      description: This object represents an error with a requestId field that serves as a unique
request identifier.
      properties:
        requestId:
          description: Uniquely identifies the HTTP request. This requestId is needed when you
contact support.
          type: string
    ErrorCode500:
      description: |
        Error codes:
        - `INTERNAL_ERROR` - Internal server error.
      type: string
      enum:
        - INTERNAL_ERROR
    ErrorResponse500:
      type: object
      additionalProperties: false
      properties:
        requestId:
          description: This optional parameter serves to uniquely identify the HTTP request. This
requestId is needed when you contact support.
          type: string
        code:
          description: This represents the unique error code corresponding to the specific error
occurrence.
          allOf:
            - $ref: '#/components/schemas/ErrorCode500'
        description:
          type: string
          maxLength: 250
          description: This provides a human-readable explanation of the error.
      required:
        - code
        - description
  responses:
    200GetAccounts:
      description: This object represents a successful HTTP 200 response returned by the API for a
request made to retrieve a list of a client's accounts.
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/GetAccountsResponse'
    200GetAccountsV2:
      description: This object represents a successful HTTP 200 response returned by the API for a
request made to retrieve a list of a client's accounts.
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/GetAccountsResponseV2'
```

```
    200GetAccountsV3:
      description: This object represents a successful HTTP 200 response returned by the API for a
request made to retrieve a list of a client's accounts.
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/GetAccountsResponseV3'
    200GetAccountBalance:
      description: This object represents a successful HTTP 200 response returned by the API for a
request made to retrieve information about a client's account balance.
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/GetAccountBalanceResponse'
    200GetAccountBalanceV3:
      description: This object represents a successful HTTP 200 response returned by the API for a
request made to retrieve information about a client's account balance.
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/GetAccountBalanceResponseV3'
    200GetAccountTransactions:
      description: This object represents a successful HTTP 200 response returned by the API for a
request made to retrieve a list of transactions for a specific account.
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/GetAccountTransactionsResponse'
    200GetAccountTransactionsV3:
      description: This object represents a successful HTTP 200 response returned by the API for a
request made to retrieve a list of transactions for a specific account.
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/GetAccountTransactionsResponseV3'
    200AccountExists:
      description: This object represents a successful HTTP 200 response returned by the API for a
request to verify the existence of an account.
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/AccountExists'
    400Error:
      description: This object represents a standard HTTP 400 (Bad Request) error returned by the
API. It is used to indicate that the server could not process the request due to invalid syntax,
missing parameters, incompatible JSON body, incorrect URL parameters or inappropriate header
fields.
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/ErrorResponse400'
    401Error:
```

```
    description: This object represents a standard HTTP 401 (Unauthorized) error returned by the API. It is used to indicate that the request lacks valid authentication credentials for the target resource.
    content:
     application/json:
      schema:
       $ref: '#/components/schemas/RequestIdErrorResponse'
  403Error:
    description: This object represents a standard HTTP 403 (Forbidden) error returned by the API. It is used to indicate that the client does not have the necessary permissions for a resource, or the request is understood by the server, but the server refuses to authorize it.
    content:
     application/json:
      schema:
       $ref: '#/components/schemas/RequestIdErrorResponse'
  404Error:
    description: This object represents a standard HTTP 404 (Not Found) error returned by the API. It is used to indicate that the requested resource could not be found on the server, or it is not implemented or defined in the standard.
    content:
     application/json:
      schema:
       $ref: '#/components/schemas/RequestIdErrorResponse'
  405Error:
    description: This object represents a standard HTTP 405 (Method Not Allowed) error returned by the API. It signifies that the client has tried to access a resource using an HTTP method that is not supported or allowed by the server for that particular resource.
    content:
     application/json:
      schema:
       $ref: '#/components/schemas/RequestIdErrorResponse'
  406Error:
    description: This object represents a standard HTTP 406 (Not Acceptable) error returned by the API. It indicates that the server cannot produce a response matching the list of acceptable values defined in the request's headers.
    content:
     application/json:
      schema:
       $ref: '#/components/schemas/RequestIdErrorResponse'
  429Error:
    description: This object represents a standard HTTP 429 (Too Many Requests) error returned by the API. It indicates that the user has sent too many requests within a given amount of time.
    headers:
     Retry-After:
      schema:
       type: string
    content:
     application/json:
      schema:
       $ref: '#/components/schemas/RequestIdErrorResponse'
  500Error:
```

```
    description: This object represents a standard HTTP 500 (Internal Server Error) returned by the API. It's used to indicate that the server encountered an unexpected condition that prevented it from fulfilling the request.
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/ErrorResponse500'
  securitySchemes:
    OauthAccessTokenAuth:
      type: http
      scheme: bearer
      bearerFormat: JWT
      description: JWT (JSON Web Token) authentication scheme for API providers. It involves producing a JWT formatted token according to RFC 7519 standards. The token payload and related claims are described in the TokenPayload schema.
x-tagGroups:
  - name: Open Banking Accounts API
    tags:
      - Accounts
```

Приложение 4
к Системе обмена информацией по открытым программным интерфейсам. Техническая информация. «Получение информации о банковском счете клиента»
(информационное)

Справочники и перечисления

1. Справочник «Статус обработки транзакции»

Значение	Описание
BOOKED	Завершена
PENDING	В обработке

2. Справочник «Индикатор кредита или дебета»

Значение	Описание
CREDIT	Кредит
DEBIT	Дебет

3. Справочник «Тип операции»

Значение	Описание
INCOME	Пополнение
WITHDRAWAL	Снятие
TRANSFER	Перевод
PURCHASE	Покупка
E_COMMERCE	Интернет-покупка
PAYMENT	Платеж
BONUS	Бонус/кэшбек
OTHER	Прочие операции

4. Справочник «Тип кредитного лимита»

Значение	Описание
AVAILABLE	Доступный

Значение	Описание
PRE_AGREED	Предварительно согласованный

5. Справочник «Тип банковского счета»

Значение	Описание
CURRENT_ACCOUNT	Текущий счет
CREDIT_CARD	Кредитная банковская карта
DEBIT_CARD	Дебетовая банковская карта
SAVINGS	Депозит

6. Справочник «Язык»

Значение	Описание
kk	Казахский язык
ru	Русский язык

7. Справочник «Область действия»

Значение	Описание
accounts	Доступ к списку счетов
account_balance	Доступ к информации о балансе счета
account_transactions	Доступ к списку транзакций счета

Приложение 5
к Системе обмена информацией по
открытым программным интерфейсам.
Техническая спецификация. «Получение
информации о банковском счете клиента»
(информационное)

Использование идентификатора Системы

1. Для процессов информационного взаимодействия по получению информации о банковского счете клиента в рамках Системы для идентификации счета используется идентификатор (accountId), сгенерированный Системой (Идентификатор в Системе).

Примечание – В системе Поставщика API счет идентифицируется по номеру банковского счета клиента (IBAN) в соответствии с Правилами открытия, ведения и закрытия банковских счетов клиентов (см. раздел «Библиография», пункт [8]).

2. Обобщенно процесс использования идентификаторов в Системе включает в себя следующие шаги:

1) Поставщик API, получив запрос предоставления списка банковских счетов клиента, проверяет для всех ли открытых банковских счетов клиента имеется идентификатор в Системе.

2) Поставщик API для открытых банковских счетов клиента, по которым ранее не был получен идентификатор в Системе, запрашивает в Системе идентификаторы (accountId).

3) Поставщик API, получив идентификаторы в Системе, сохраняет их и информацию о том, каким счетам клиента (каким IBAN) они соответствуют.

4) Поставщик API в ответе на запрос предоставления списка банковских счетов в качестве идентификатора счета передает идентификатор в Системе (accountId).

5) Поставщик API при последующем получении запроса предоставления информации по идентификатору в Системе (например, баланса или списка транзакций) для его обработки использует данные, сохраненные на шаге в (т.е. сопоставление идентификатора в Системе (accountId) с номером банковского счета клиента (IBAN).

Библиография

[1] The JavaScript Object Notation (JSON) Data Interchange Format. RFC 7158. https://www.rfc-editor.org/rfc/rfc7158.

[2] ISO 8601 Элементы данных и форматы для обмена информацией. Обмен информацией. Представление дат и времени.

[3] Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content. RFC 7231. https://www.rfc-editor.org/rfc/rfc7231.

[4] A UniveRSAlly Unique IDentifier (UUID) URN Namespace. RFC 4122. https://tools.ietf.org/html/rfc4122.

[5] НК РК 07 ISO 4217-2019 Коды для представления валют и фондов.

[6] Приказ Министра финансов Республики Казахстан от 18 сентября 2014 года № 403 «Некоторые вопросы Единой бюджетной классификации Республики Казахстан».

[7] Постановление Правления Национального Банка Республики Казахстан от 31 августа 2016 года № 203 «Об утверждении Правил применения кодов секторов экономики и назначения платежей».

[8] Постановление Правления Национального Банка Республики Казахстан от 31 августа 2016 года № 207 «Об утверждении Правил открытия, ведения и закрытия банковских счетов клиентов».