Form3 Coding Exercise Design

Assumptions

- The API paths contain the plural form of resources, and the HTTP method defines the kind of action to be performed on the resource.
- {paymentId} is assumed to be universally unique (UUID).
- I have assumed 'List a collection of payment resources' to mean 'Fetch a collection of payment resources'.
- Please see README.md for operating environments, open source libraries used and build and run instructions for this project.
- I made a design decision not to normalise the payments data attributes into relational tables, since focus for the project was on design, concepts, testing, and speed of implementation.

API

/v1/public/payments/{paymentId}

HTTP Method

GET

Description

Fetch a payment resource.

Comments

{paymentId} should match the PaymentData::id field in the PaymentData structure detailed below.

Return Status

- 200 Ok
- 404 Not Found

/v1/public/payments

HTTP Method

POST

Description

Create a payment resource.

Comments

A new payment will be created with the POST data, which should be JSON representation of the `PaymentData` structure detailed below.

Return Status

- 200 Created
- 422 Unprocessable Entity

/v1/public/payments/{paymentId}

HTTP Method

PUT

Description

Update a payment resource.

Comments

- {paymentId} should match the PaymentData::id field detailed in the PaymentData structure below.
- The payment data associated with {paymentId} will be replaced with the PUT data, which should be JSON representation of the `PaymentData` structure detailed below.
- This API is distinct from POST (above), and will fail with 404 if {paymentId} is not found.

Return Status

- 200 Ok
- 404 Not Found

/v1/public/payments/{paymentId}

HTTP Method

DELETE

Description

Delete a payment resource.

Comments

- {paymentId} should match the PaymentData::id field detailed in the PaymentData structure below.
- DELETE on collection not supported since:
  - asymmetric with respect to single-item DELETE
  - dangerous operation

Return Status

- 200 Payment deleted ok
- 404 Not Found

/v1/public/payments?offset=<int>&limit=<int>

HTTP Method

GET

Description

Fetch a collection of payment resources (please see Assumptions).

Comments

The integer parameters **'offset'** and **'limit'** can be specified to constrain the fetched payments; offset is the zero-based offset into the unordered list of stored payments, and limit is the number of payments to return. Eg: /v1/public/payments?offset=2&count=1 returns a single repayment, the third in the database.

Return Status

- 200 Ok
- 404 Not Found

Data Structures

The following details the JSON data structures for payments which is passed and received by the API detailed above.

```
PaymentData {
        type (string, required): The type of data (should be "Payment" for this result)
        id (string, required): UUID for this data
        version (int, required): Version used
        organisation_id (string, required): UUID for the organisation
        attributes (PaymentAttributes, required):
}
PaymentBeneficiaryParty {
        account_name (string, required)
        account_number (string, required)
        account_number_code (string, required)
        account_type (int, required)
        address (string, required)
        bank_id (string, required)
        bank_id_code (string, required)
        name (string, required)
}
PaymentChargesInformation {
        bearer_code (string, required)
        sender_charges (Array[PaymentSenderCharges], required)
        receiver_charges_amount (string, required)
        receiver_charges_currency (string, required)
}
PaymentSenderCharges {
        amount (string, required)
        currency (string, required)
}
PaymentDebtorParty {
        account_name (string, required)
        account_number (string, required)
        account_number_code (string, required)
        address (string, required)
        bank_id (string, required)
        bank_id_code (string, required)
        name (string, required)
}
```

```
PaymentFx {
        contract_reference (string, required)
        exchange_rate (string, required)
        original_amount (string, required)
        original_currency (string, required)
}
PaymentSponsorParty {
        account_number (string, required)
        bank_id (string, required)
        bank_id_code (string, required)
}
PaymentAttributes {
        amount (string, required)
        beneficiary_party (PaymentBeneficiaryParty, required)
        charges_information (PaymentChargesInformation, required)
        currency (string, required)
        debtor_party (PaymentDebtorParty, required)
        end_to_end_reference (string, required)
        fx (PaymentFx, required)
        numeric_reference (string, required)
        payment_id (string, required)
        payment_purpose (string, required)
        payment_scheme (string, required)
        payment_type (string, required)
        processing_date (string, required)
        reference (string, required)
        scheme_payment_sub_type (string, required)
        scheme_payment_type (string, required)
        sponsor_party (PaymentSponsorParty, required)
}
```