Form 3 Payments Architecture
Payments Task Document

Project	Form 3 Payments Architecture
Document	Payments Architecture Document
Version	0.1
Created	Tue Mar 13 2018
Last edited	Tue Mar 13 2018
Document author	Leslie Drewery

Address

tel :

http://www.form3.tech

Form 3 Payments Architecture
Task Overview

Design

Provide a design for part of a payments API. A list of payments might look like http://mockbin.org/bin/41ca3269-d8c4-4063-9fd5-f306814ff03f.

Design requirements:

- API should be RESTFUL

- API should be able to:

  - Fetch a payment resource
  - Create, update and delete a payment resource
  - List a collection of payment resources
  - Persist resource state (e.g. to a database)

Example use case:

As an example, a JavaScript client should be able to call the API and list a collection of payments (building a client is outside the scope of this exercise however).

Assumptions :
1.) Payments will be created/updated as full JSON objects.

Notes :
1.) No support for paging, eTags, sorting, filtering will be implemented in demo even though defined.
2.) Based on a very simple implementation of "Clean Architecture"
3.) TDD principles to be used in implementation using JUnit.

Project	Form 3 Payments Architecture
Document	Payments Architecture Document
Version	0.1
Created	Sept 17 2012
Last edited	Tue Mar 13 2018
Document author	Leslie Drewery

Form 3 Payments Architecture

1.) Rest API

1.1) List a Collection

1.2) Fetch a payment Resource

1.3) Create a payment Resource

1.4) Update a payment Resource

1.5) Delete a payment Resource

Project	Form 3 Payments Architecture
Document	Payments Architecture Document
Version	0.1
Created	Sept 17 2012
Last edited	Tue Mar 13 2018
Document author	Leslie Drewery

1.1) List a Collection

o Fetch payment resources (List a collection of payment resources)

Method **"GET"** path /payments?page={page number}&limit={records per page}&sort={asc,desc}

Request

Headers : content-type="application/json", Accept-encoding=GZIP, etag=<etag>

(Authorization outside of scope)

Parameters : page=<Page Number>,limit=<limit number of records per page>, sort=<desc,asc> (Not to be implemented in Demo)

Note : Use pagination, sorting,filtering and etag to navigate big lists (Not implemented in demo)

Response

Code : 200 (OK), single/multiple payments. 404 (Not Found), if ID not found or invalid,304 (not modified).

standard HTTP responses apply.

Headers : etag=<etag>

Body : List Payment Entities (domain model) [{Payment Entity},{Payment Entity}]

Project	Form 3 Payments Architecture
Document	Payments Architecture Document
Version	0.1
Created	Sept 17 2012
Last edited	Tue Mar 13 2018
Document author	Leslie Drewery

1.2) Fetch a payment Resource

- Fetch a payment resource

Method **“GET”** path /payments/{resourceid}

Request

Headers : content-type=”application/json”, Accept-encoding=GZIP (Authorization outside of scope)

Resourceid : unique id of the payment resource requested

Response

Code : 200 (OK) single payment, 404 (Not Found), if ID not found or invalid. standard HTTP responses apply.

Body : Payment Entity Model (domain model)

Project	Form 3 Payments Architecture
Document	Payments Architecture Document
Version	0.1
Created	Sept 17 2012
Last edited	Tue Mar 13 2018
Document author	Leslie Drewery

1.3) Create a payment Resource

- Create a payment resource

Method **“POST”** path /payments

Request

Headers : content-type=”application/json”, Accept-encoding=GZIP (Authorization outside of scope)

Body : Payment Entity Model (domain model)

Response

Code : 201 (Created), 404 (Not Found), 409 (Conflict) if resource already exists..

standard HTTP responses apply.

Header : “Location” = “/payments/{resourceid}” : Reference to the created entities new ID.

Body : empty

Project	Form 3 Payments Architecture
Document	Payments Architecture Document
Version	0.1
Created	Sept 17 2012
Last edited	Tue Mar 13 2018
Document author	Leslie Drewery

1.4) Update a payment Resource

○ Update a payment resource

Method **“PATCH”** path /payments/{resourceid}

Request

Headers : content-type=”application/json”, Accept-encoding=GZIP (Authorization outside of scope)

resourceid : unique id of the payment resource to update

Body : Payment Entity Model (domain model)

Response

Code : 200 (OK) or 204 (No Content). 404 (Not Found), if ID not found or invalid. standard HTTP responses apply.

Header : no headers

Body : empty

Project	Form 3 Payments Architecture
Document	Payments Architecture Document
Version	0.1
Created	Sept 17 2012
Last edited	Tue Mar 13 2018
Document author	Leslie Drewery

1.5) Delete a payment Resource

- Delete a payment resource

Method **“DELETE”** path /payments/{resourceid}

Request

Headers : content-type=”application/json”, Accept-encoding=GZIP (Authorization outside of scope)

resourceid : unique id of the payment resource to update

Body : empty

Response

Code : 200 (OK). 404 (Not Found), if ID not found or invalid. standard HTTP responses apply.

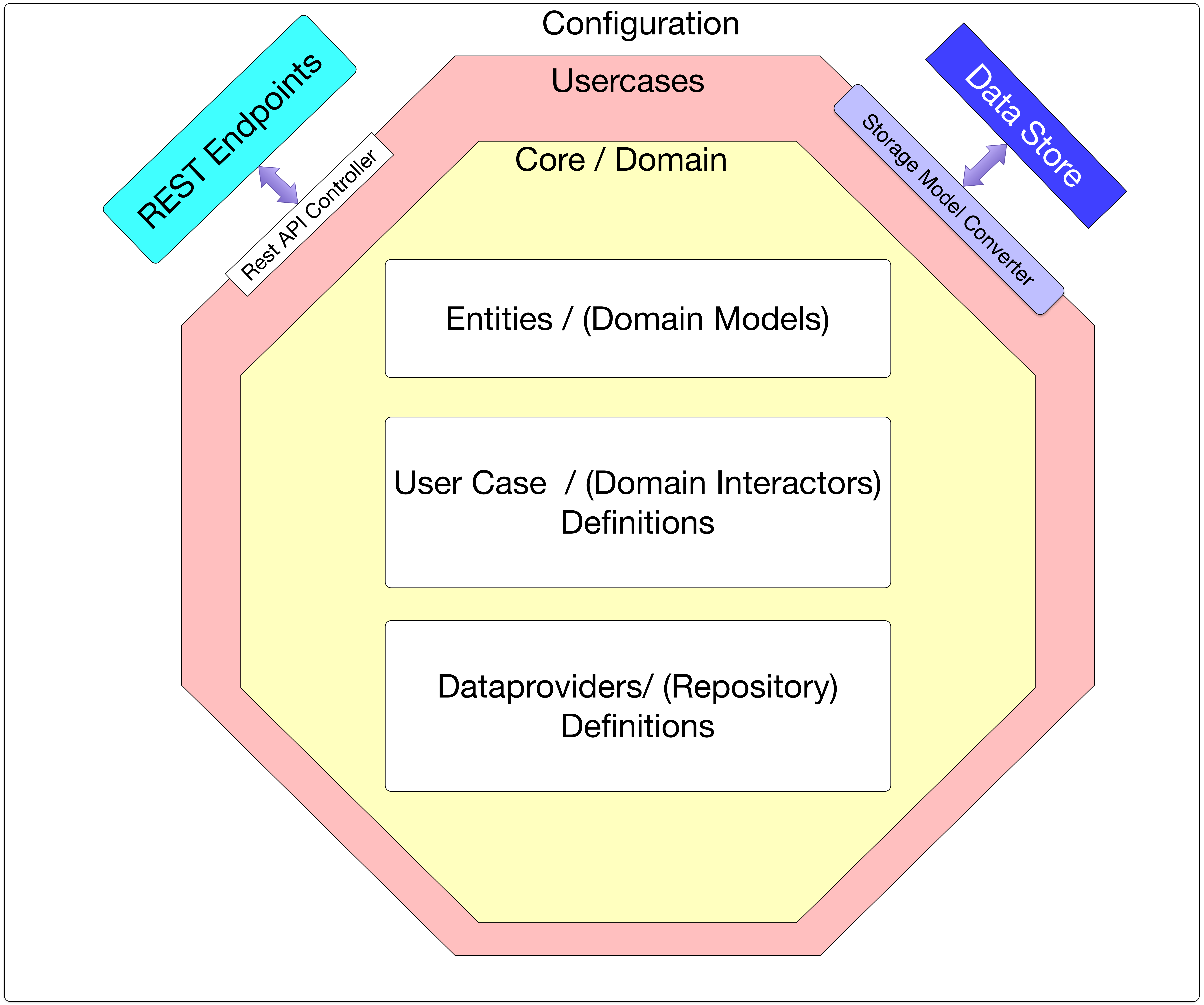
Header : no headers

Body : empty

Project	Form 3 Payments Architecture
Document	Payments Architecture Document
Version	0.1
Created	Sept 17 2012
Last edited	Tue Mar 13 2018
Document author	Leslie Drewery

Form 3 Payments Architecture

2.) Clean Architecture


Configuration
REST Endpoints
Data Store
Usercases
Rest API Controller
Storage Model Converter
Core / Domain
Entities / (Domain Models)
User Case / (Domain Interactors) Definitions
Dataproviders/ (Repository) Definitions

Project	Form 3 Payments Architecture
Document	Payments Architecture Document
Version	0.1
Created	Sept 17 2012
Last edited	Tue Mar 13 2018
Document author	Leslie Drewery

Form 3 Payments Architecture

2.1) Enties / Models

Source Data

```
{
  "data": [
    {
      "type": "Payment",
      "id": "4ee3a8d8-ca7b-4290-a52c-dd5b6165ec43",
      "version": 0,
      "organisation_id": "743d5b63-8e6f-432e-a8fa-c5d8d2ee5fcb",
      "attributes": {
        "amount": "100.21",
        "beneficiary_party": {
          "account_name": "W Owens",
          "account_number": "31926819",
          "account_number_code": "BBAN",
          "account_type": 0,
          "address": "1 The Beneficiary Localtown SE2",
          "bank_id": "403000",
          "bank_id_code": "GBDSC",
          "name": "Wilfred Jeremiah Owens"
        },
        "charges_information": {
          "bearer_code": "SHAR",
          "sender_charges": [
            {
              "amount": "5.00",
              "currency": "GBP"
            },
            {
              "amount": "10.00",
              "currency": "USD"
            }
          ],
          "receiver_charges_amount": "1.00",
          "receiver_charges_currency": "USD"
        },
        "currency": "GBP",
        "debtor_party": {
          "account_name": "EJ Brown Black",
          "account_number": "GB29XABC10161234567801",
          "account_number_code": "IBAN",
          "address": "10 Debtor Crescent Sourcetown NE1",
          "bank_id": "203301",
          "bank_id_code": "GBDSC",
          "name": "Emelia Jane Brown"
        },
        "end_to_end_reference": "Wil piano Jan",
        "fx": {
          "contract_reference": "FX123",
          "exchange_rate": "2.00000",
          "original_amount": "200.42",
          "original_currency": "USD"
        },
        "numeric_reference": "1002001",
        "payment_id": "123456789012345678",
        "payment_purpose": "Paying for goods/services",
        "payment_scheme": "FPS",
        "payment_type": "Credit",
        "processing_date": "2017-01-18",
        "reference": "Payment for Em's piano lessons",
        "scheme_payment_sub_type": "InternetBanking",
        "scheme_payment_type": "ImmediatePayment",
        "sponsor_party": {
          "account_number": "56781234",
          "bank_id": "123123",
          "bank_id_code": "GBDSC"
        }
      }
    }
  ],
  "links": {
    "self": "https://api.test.form3.tech/v1/payments"
  }
}
```

Use Pojo to convert JSON to Entity Objects
link : http://www.jsonschema2pojo.org/

Note :
1.) “Datam” class must be refactored to “Payment” class for repository interface to be valid.
2.) Remove public references to variables.
3.) Create getters and constructors

Project	Form 3 Payments Architecture
Document	Payments Architecture Document
Version	0.1
Created	Sept 17 2012
Last edited	Tue Mar 13 2018
Document author	Leslie Drewery

Form 3 Payments Architecture

2.2) Datastore / Domain Repository Definition

```
public interface PaymentRepository {

    /*
    * Insert new payment resource.
    */
    void insert(Payment paymentEntity);

    /*
    * Update payment resource.
    */
    void update(Payment paymentEntity);

    /*
    * Get a specific payment resources by id.
    */
    Payment getPaymentById(string id);

    /*
    * fetch a list of payment resources.
    */
    List<Payment> getAllPayments(Integer page,Integer limit, String sort,String
    selection, String eTag);

    /*
    * delete an existing payment resource by id.
    */
    void delete(String id);
}
```

Project	Form 3 Payments Architecture
Document	Payments Architecture Document
Version	0.1
Created	Sept 17 2012
Last edited	Tue Mar 13 2018
Document author	Leslie Drewery