Lending Platform Design V1.0

This document details information on APIs that will service Lending platform user creation, loan creation and loan repayment.

Overview.

External dependencies Required

1. RabbitMQ - for queuing of requests that needs to be processed asynchronously
2. Mysql Database - For storing records
3. Sonarqube - for running code test coverage and vulnerabilities
4. Docker - for building container images for the microservices
5. Postman - for testing api's

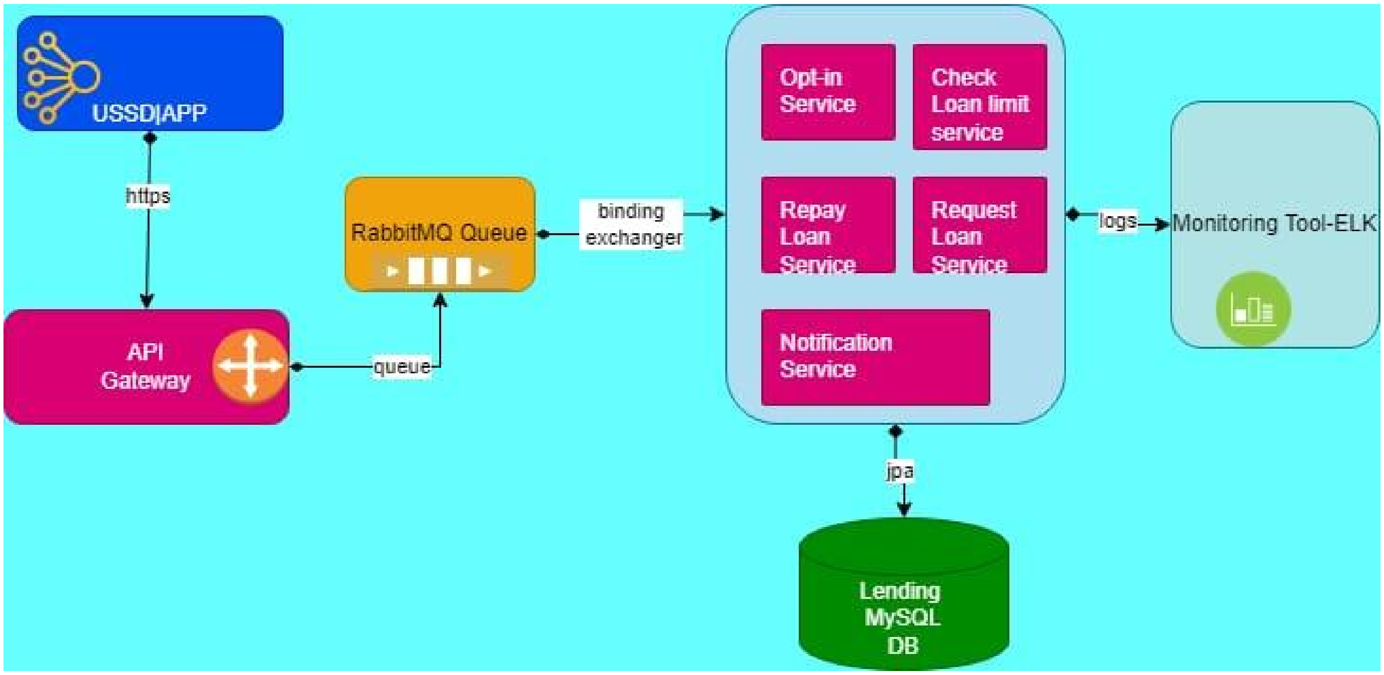
Microservices: Java Reactive Spring-boot based services - reactive spring-boot services improves on the performance and response times since they are non-blocking. Below are the microservices:

1. Api Gateway - This service is the gateway to all the services. Handles security authentication then forwards request to other services depending on the endpoint
2. Limit Service - handles both opt-in and loan limit management. This can be decoupled to have opt-in service and check limit service separately.
3. Request Loan Service - for handling loan requests from users
4. Repay Loan Service - for handling loan repayment
5. Notification Service - handles all the notification from the system i.e sms or email or both

- The services are loosely coupled and the communication between them is through rabbit mq, except for some that need a response immediately.
- The microservices are secured using basic authentication (Credentials shared)
- The Rate limiting is also implemented in the api gateway. Set at 10 requests per second . This value is configurable and can be updated when necessary.
- As for the design , we envision using ELK Monitoring. (Not implemented for this project)

- ➢ This design reduces the chances of failing to process the customer requests in case one of the microservice is down, since when the service is back up it will process all the requests in the queue.
- ➢ The pitfalls for this design is connectivity failure between our services with the RabbitMQ node - to overcome this I will implement my own recovery by defining connection failure event handlers.

High Level desgn



Common Header Specification

Header Name	Header Value	Description
Authorization	Basic authentication	Use this for testing purposes <<Username>> =api-user <<password>>=ZGV2ZWxvcC1hcGktdXNlcg==
Content-Type	application/json	Content type of request being sent.

Add users to Lending Platform

Via API

1.

This API will be exposed to be consumed by CHANNELS. Customers will make a request via available channels to add MSISDN to the system. After validation , addition to the database will be done.

The API will accept the following parameters

PARAMETER	DATA TYPE	MANDATORY	DESC
`RequestRefID`	String	Yes	Unique identifier of the transaction
`firstname`	String	Yes	The first name of the customer.

`lastname`	String	No	Optional for users to provide their last name
SourceSystem	String	Y	Name of the initiating system. Can be USSD/APP/Portal
email	String	No	Optional user email address
msisdn	String	Y	MSISDN to be added

API Structure

```
{
    "RequestRefID": "app-{{$guid}}",
    "firstname": "Lazarus",
    "lastname": "Korir",
    "SourceSystem": "Mobile app",
    "msisdn": "254723846453",
    "email": "lazaruskorir95@gmail.com"
}
```

Response

Parameter	**Description**	**Example**	
`RequestRefID`	Request ID	`21c7d07-560c-4fe5-8811-c5446ae33569`	
`ResponseCode`	Response Code	200	
`ResponseDesc`	Response Description	`"Successfully added to Lending platform”`	
`Body`	Array of JSON objects	No	Array of JSON objects with additional information on the status of the transaction that took place.

```
{
 "RequestRefID": "1865d189-6934-4744-9279-88bc4ee4a59e",
"ResponseCode": "200",
 "ResponseDesc": "Successfully added to the platform",
 "TransactionID": "",
 "Body": []
}
```

TABLE STRUCTURE

1. tbl_users

Field Name	Description	Data Type	Mandatory	Example Value
id	AI, Identity	Integer	Y	1
REQUEST_REF_ID	Unique request Id	String	Y	e.g. Us2s121212
MSISDN	Phone Number	String	Y	e.g. 254700100131
EMAIL			N	E.g lazarus@gmail.com
FIRST_NAME			Y	E.g Lazarus
LAST_NAME	Phone Number		N	e.g KORIR
SOURCE_SYSTEM	Channel Used to add to whitelist	String	Y	e.g. CRM, KAFKA
LAST_UPDATED_AT	Time of last update	DateTime	N	e.g. 20190101
CREATED_AT	Time of request	DateTime	Y	e.g. 20190101

Add users record to DB

Check limit

This API will be exposed to query check users loan

limit and eligible products

Via API

1.

This API will be exposed to be consumed by CHANNELS. Customers will make a request via available channels to add check limits and available products before choosing a product to request a loan on.. After validation of the request. Users will receive a list of eligible products.

The API will accept the following parameters

PARAMETER	DATA TYPE	MANDATORY	DESC
`RequestRefID`	String	Yes	Unique identifier of the transaction
SourceSystem	String	Y	Name of the initiating system. Can be USSD/APP/Portal
amount	String	No	Loan amount a user is requesting
msisdn	String	Y	MSISDN to be added

API Structure

```
{
    "RequestRefID": "app-{{$guid}}",
    "SourceSystem": "Mobile app",
    "msisdn": "254723846453",
    "amount":"1000"
}
```

Response

Parameter	Description	Example	
RequestRefID	Request ID	21c7d07-560c-4fe5-8811-c5446ae33569	
ResponseCode	Response Code	200	
ResponseDesc	Response Description	"You are eligible for our Loan products”	
Body	Array of JSON objects	No	Array of JSON objects with additional information of the product details.

```
{
    "RequestRefID": "app-8182e34c-eec9-49b5-b6ed-5ef116236d56",
    "ResponseCode": "200",
    "ResponseDesc": "You are eligible for our Loan products",
    "TransactionID": "",
    "Body": [
        {
            "id": 1,
            "productName": "Product A",
            "maxLimit": 1000.0,
            "percentageInterest": 10.0,
            "tenureDays": 15,
            "productId": "1001"
        }
    ]
}
```

TABLE STRUCTURE

tbl_products

1.

Field Name	**Description**	**Data Type**	**Mandatory**	**Example Value**
id	AI, Identity	Integer	Y	1
PRODUCT_NAME	Name of the product	String	Y	e.g. Product A
MAX_LIMIT	Max amount for the product	BigDecimal	Y	e.g. 1000
PERCENTAGE_INTEREST	Interest in % for the product	double	Y	E.g 10
TENURE_DAYS	Tenure in days for the product	Integer	Y	E.g 15
PRODUCT_ID	Unique id of the product	String	Y	e.g 1001
MODE	Mode of notification for the product	String	N	E.g sms
LAST_UPDATED_AT	Time of last update	DateTime	N	e.g. 20190101
CREATED_AT	Time of request	DateTime	Y	e.g. 20190101

Request Loan

This API will be exposed to request loan

Via API

1.

This API will be exposed to be consumed by CHANNELS. Customers will make a request via available channels to request loans on the selected products.. After validation of the request. Api will return an acknowledgement response. Then the request will be queued for processing, once the request is processed , customers will get a notification

The API will accept the following parameters

PARAMETER	**DATA TYPE**	**MANDATORY**	**DESC**
RequestRefID	String	Yes	Unique identifier of the transaction
SourceSystem	String	Y	Name of the initiating system. Can be USSD/APP/Portal
amount	String	No	Loan amount a user is requesting
msisdn	String	Y	MSISDN to be added

productID	String	Y	Unique for the selected product

API Structure

```
{
    "RequestRefID": "app-{{$guid}}",
    "SourceSystem": "Mobile app",
    "msisdn": "254723846453",
    "amount":"1000",
    "productID":"1001"
}
```

Response

Parameter	**Description**	**Example**	
RequestRefID	Request ID	21c7d07-560c-4fe5-8811-c5446ae33569	
ResponseCode	Response Code	200	
ResponseDesc	Response Description	"Loan request received and is being processed”	
Body	Array of JSON objects	No	Array of JSON objects with additional information of the status of the request

```
{
    "RequestRefID": "app-8182e34c-eec9-49b5-b6ed-5ef116236d56",
    "ResponseCode": "200",
    "ResponseDesc": "Loan request received and is being processed",
    "TransactionID": "",
    "Body": []
}
```

TABLE STRUCTURE

1. tbl_loans

Field Name	**Description**	**Data Type**	**Mandatory**	**Example Value**
id	AI, Identity	Integer	Y	1
REFERENCE_ID	Unique identifier of the transaction	String	Y	e.g. ggfeyfgeywueyew83
AMOUNT	Loan Amount taken	BigDecimal	Y	e.g. 1000
DUE_DATE	Due date for the loan taken	DateTime	Y	E.g 20190101
PRODUCT_ID	Unique id of the product	String	Y	e.g 1001
STATUS	Status of the loan			
LAST_UPDATED_AT	Time of last update	DateTime	N	e.g. 20190101
CREATED_AT	Time of request	DateTime	Y	e.g. 20190101

Loan Repayment

This API will be exposed to manually repay loan

Via API

1.

This API will be exposed to be consumed by CHANNELS. Customers will manually make a request via available channels to repay existing loans. After validation of the request. Api will return an acknowledgement response. Then the request will be queued for processing, once the request is processed , customers will get a notification.

There is also a scheduled cron job that runs at midnight to try and repay due loans automatically by deducting from customers wallet. The api queries records by due date and loops through returned data for automatic repayment

The API will accept the following parameters

PARAMETER	**DATA TYPE**	**MANDATORY**	**DESC**
`RequestRefID`	String	Yes	Unique identifier of the transaction
SourceSystem	String	Y	Name of the initiating system. Can be USSD/APP/Portal

amount	String	No	Loan amount a user is requesting
msisdn	String	Y	MSISDN to be added
id	String	Y	Unique for the selected loan

API Structure

```
{
    "RequestRefID": "app-{{$guid}}",
    "SourceSystem": "Mobile app",
    "msisdn": "254723846453",
    "amount":"1000",
    "id":"101"
}
```

Response

Parameter	**Description**	**Example**	
RequestRefID	Request ID	21c7d07-560c-4fe5-8811-c5446ae33569	
ResponseCode	Response Code	200	
ResponseDesc	Response Description	"Repay Loan request received and is being processed”	
Body	Array of JSON objects	No	Array of JSON objects with additional information of the status of the request

```
{
    "RequestRefID": "app-8182e34c-eec9-49b5-b6ed-5ef116236d56",
    "ResponseCode": "200",
    "ResponseDesc": "Repay Loan request received and is being processed",
    "TransactionID": "",
    "Body": []
}
```

TABLE STRUCTURE

1. tbl_loans

Field Name	Description	Data Type	Mandatory	Example Value
id	AI, Identity	Integer	Y	1
REFERENCE_ID	Unique identifier of the transaction	String	Y	e.g. ggfeyfgeywueyew83
AMOUNT	Loan Amount taken	BigDecimal	Y	e.g. 1000
DUE_DATE	Due date for the loan taken	DateTime	Y	E.g 20190101
PRODUCT_ID	Unique id of the product	String	Y	e.g 1001
STATUS	Status of the loan	String	Y	E.g paid
LAST_UPDATED_AT	Time of last update	DateTime	N	e.g. 20190101
CREATED_AT	Time of request	DateTime	Y	e.g. 20190101

If sonarqube is started locally then you run this in the directory of the microservice
Use this command to run sorna : `mvn sonar:sonar -Dsonar.host.url=http://localhost:9000 -Dsonar.login=the-generated-token`
Once you run it , paste this link on the browser to check on the coverage
http://localhost:9000

You can get more api documentation from this link :
https://documenter.getpostman.com/view/7495088/2s93JtQij2