

Arquitectura de Software

EXAMEN PRACTICO FINAL

Integrantes: Bryan Jiménez, Steven Leiva















Capturas de pantalla

Test





- Coverage de products-accounts

products-accounts





products-accounts

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
com.banquito.core.productsaccounts.controller.dto		36 %		0 %	66	117	0	21	8	59	0	4
com.banquito.core.productsaccounts.model		29 %		7 %	64	100	0	18	3	38	0	2
com.banquito.core.productsaccounts.controller		86 %		75 %	2	14	6	48	1	12	0	2
com.banquito.core.productsaccounts.service		94 %		100 %	0	16	3	72	0	12	0	2
com.banquito.core.productsaccounts		0 %		n/a	2	2	3	3	2	2	1	1
com.banquito.core.productsaccounts.controller.mapper		96 %		75 %	4	12	2	47	2	8	0	2
com.banquito.core.productsaccounts.exception		100 %		n/a	0	4	0	10	0	4	0	1
Total	996 of 2.087	52 %	234 of 260	10 %	138	265	14	219	16	135	1	14

















com.banquito.core.productsaccounts.controller

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
InterestRateController		81 %		100 %	1	8	5	29	1	7	0	1
ProductAccountController		95 %		50 %	1	6	1	19	0	5	0	1
Total	25 of 188	86 %	1 of 4	75 %	2	14	6	48	1	12	0	2


com.banquito.core.productsaccounts.service

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
InterestRateService		91 %		100 %	0	10	3	50	0	7	0	1
ProductAccountService		100 %		100 %	0	6	0	22	0	5	0	1
Total	17 of 296	94 %	0 of 8	100 %	0	16	3	72	0	12	0	2


- Coverage de branches
branches

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
com.banquito.core.branches.config		0 %		0 %	19	19	27	27	16	16	4	4
com.banquito.core.branches.controller.dto		34 %		0 %	20	34	0	7	5	19	0	2
com.banquito.core.branches.model		25 %		0 %	18	27	0	9	3	12	0	1
com.banquito.core.branches		0 %		n/a	2	2	3	3	2	2	1	1
com.banquito.core.branches.controller		96 %		50 %	1	7	1	24	0	6	0	1
com.banquito.core.branches.controller.mapper		94 %		75 %	2	6	1	16	1	4	0	1
com.banquito.core.branches.service		100 %		100 %	0	9	0	36	0	7	0	1
com.banquito.core.branches.exception		100 %		n/a	0	4	0	10	0	4	0	1
Total	411 of 853	51 %	68 of 76	10 %	62	108	32	132	27	70	5	12

com.banquito.core.branches.service

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
 BranchService	<div><div></div></div>	100 %	<div><div></div></div>	100 %	0	9	0	36	0	7	0	1
Total	0 of 161	100 %	0 of 4	100 %	0	9	0	36	0	7	0	1

com.banquito.core.branches.controller

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
 BranchController	<div><div></div></div>	96 %	<div><div></div></div>	50 %	1	7	1	24	0	6	0	1
Total	3 of 93	96 %	1 of 2	50 %	1	7	1	24	0	6	0	1

Base de datos de mongo

```
1 print('#####ScriptStart#####');
2
3 db = db.getSiblingDB('branche');
4 db.createCollection('branches');
5
6 var brancheColl = db.getCollection('branches');
7
8 brancheColl.insertMany([
9   {"code": "001", "name": "Guayaquil"},
10  {"code": "002", "name": "Quito"},
11  {"code": "003", "name": "Cuenca"},
12  {"code": "004", "name": "Ambato"}
13 ]);
14
15 print('#####ScriptEnd#####');
```

Base de datos de postgres

```
1 create table INTEREST_RATE (
2   ID_INTEREST_RATE SERIAL not null,
3   NAME VARCHAR(50) not null,
4   INTEREST_RATE NUMERIC(2,2) not null,
5   STATE VARCHAR(3) not null,
6   constraint CKC_STATE_INTEREST check (STATE in ('ACT','INA')),
7   START_DATE DATE not null,
8   END_DATE DATE null,
9   constraint PK_INTEREST_RATE primary key (ID_INTEREST_RATE)
10 );
11
12 create table PRODUCT_ACCOUNT (
13   ID_PRODUCT_ACCOUNT VARCHAR(16) not null,
14   NAME VARCHAR(50) not null,
15   DESCRIPTION VARCHAR(500) not null,
16   MINIMUM_BALANCE NUMERIC(10,2) not null,
17   PAY_INTEREST VARCHAR(1) not null,
18   constraint CKC_PAY_INTEREST_PRODUCT_ check (PAY_INTEREST in ('Y','N')),
19   ACCEPTS_CHECKS VARCHAR(1) not null,
20   constraint CKC_ACCEPTS_CHECKS_PRODUCT_ check (ACCEPTS_CHECKS in ('Y','N')),
21   STATE VARCHAR(3) not null,
22   constraint CKC_STATE_PRODUCT_ check (STATE in ('DRA','ACT','INA')),
23   CREATION_DATE DATE not null,
24   constraint PK_PRODUCT_ACCOUNT primary key (ID_PRODUCT_ACCOUNT)
25 );
26
27
28 insert into interest_rate values(default, 'PASIVA CORRIENTE', 0.0, 'ACT', NOW(), null);
29 insert into product_account values('CORR2023', 'Cuenta Corriente 2022', 'Cuenta corriente completamente autogestionada y digital; no maneja cheques ni paga interes.', 0.0, 'N', 'N', 'ACT', NOW());
```

Docker – Compose

```
1  version: '2'
2
3  services:
4
5      kong-database:
6          image: postgres:latest
7          hostname: kong-database
8          container_name: kong-database
9          environment:
10             POSTGRES_USER: "kong"
11             POSTGRES_DB: "kong"
12             POSTGRES_PASSWORD: "kong"
13          ports:
14             - "5432:5432"
15
16      kong-bootstrap:
17          image: kong:3.1.1-alpine
18          hostname: kong
19          container_name: kong-bootstrap
20          depends_on:
21             - kong-database
22          environment:
23             KONG_DATABASE: "postgres"
24             KONG_PG_HOST: "kong-database"
25             KONG_PG_DATABASE: "kong"
26             KONG_PG_USER: "kong"
27             KONG_PG_PASSWORD: "kong"
28          command: "kong migrations bootstrap"
29          restart: "on-failure"
30
31      kong:
32          image: kong:3.1.1-alpine
33          hostname: kong
34          container_name: kong
35          restart: always
36          depends_on:
37             - kong-bootstrap
38          environment:
39             KONG_DATABASE: "postgres"
40             KONG_PG_HOST: "kong-database"
41             KONG_PG_DATABASE: "kong"
42             KONG_PG_USER: "kong"
43             KONG_PG_PASSWORD: "kong"
44             KONG_PROXY_ACCESS_LOG: /dev/stdout
45             KONG_ADMIN_ACCESS_LOG: /dev/stdout
46             KONG_PROXY_ERROR_LOG: /dev/stdout
47             KONG_ADMIN_ERROR_LOG: /dev/stdout
48             KONG_ADMIN_LISTEN: "0.0.0.0:8001, 0.0.0.0:8444 ssl"
49          command: "kong start"
50          ports:
51             - "8000:8000"
52             - "8443:8443"
53             - "8001:8001"
54             - "8444:8444"
55
56      products-accounts:
57          build:
58             context: ./products-accounts
59          container_name: products-accounts
60          hostname: products-accounts
61          depends_on:
62             - postgres
63
64      branches:
65          build:
66             context: ./branches
67          container_name: branches
68          hostname: branches
69          depends_on:
70             - mongo
71
72      postgres:
73          image: postgres:latest
74          hostname: postgres-banquito
75          container_name: postgres-banquito
76          environment:
77             - POSTGRES_USER=admin
78             - POSTGRES_PASSWORD=admin123
79             - POSTGRES_DB=products_accounts
80          volumes:
81             - ./postgres-init/init.sql:/docker-entrypoint-initdb.d/init.sql
82
83      mongo:
84          image: mongo:latest
85          hostname: mongo-banquito
86          container_name: mongo-banquito
87          environment:
88             - MONGO_INITDB_ROOT_USERNAME=admin
89             - MONGO_INITDB_ROOT_PASSWORD=admin123
90             - MONGO_INITDB_DATABASE=branche
91          volumes:
92             - ./mongo-init/init.js:/docker-entrypoint-initdb.d/init.js
93
94      networks:
95          default:
96             name: kong-net
```

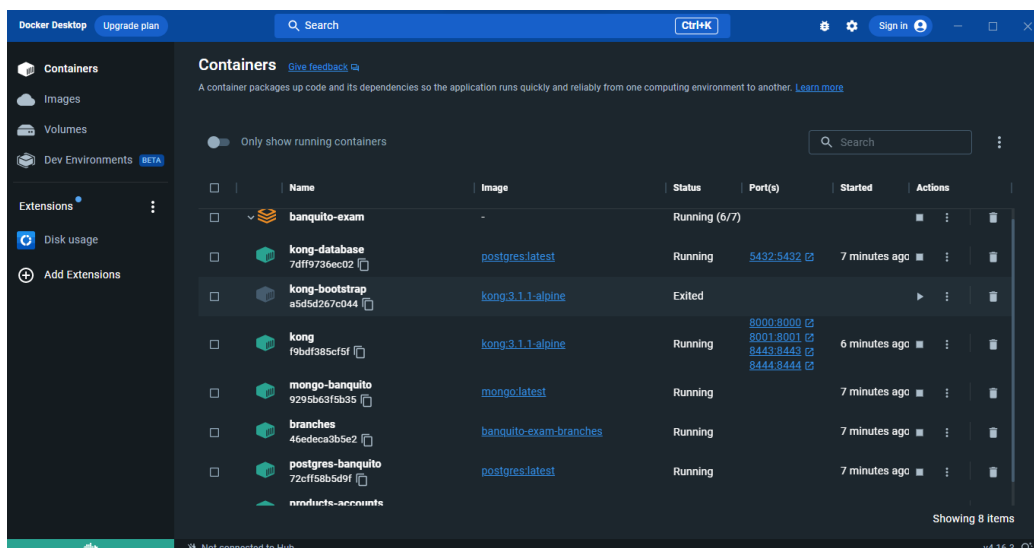
Application. Properties de Branches

```
1 server.port=8000
2
3 #----- MONGO Connection docker-----
4 spring.data.mongodb.uri=mongodb://admin:admin123@mongo-banquito:27017/branche?authSource=admin
5
6 logging.level.root=DEBUG
```

Application. Properties de Products – Accounts

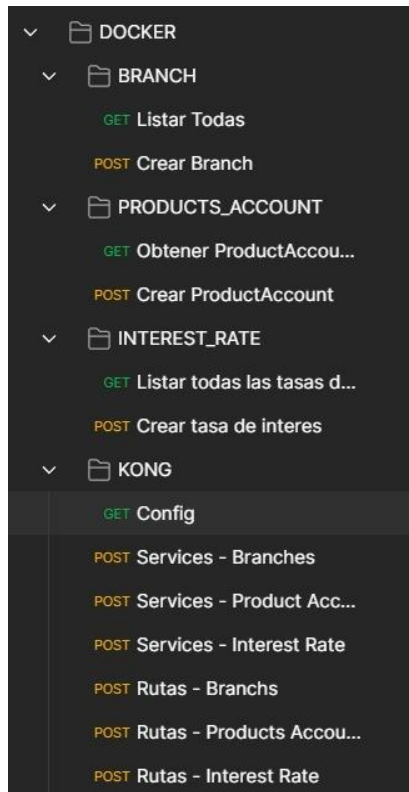
```
1 server.port=9000
2
3 #----- PostgreSQL Connection COMPOSE DB-----
4 spring.datasource.url=jdbc:postgresql://postgres-banquito:5432/products_accounts
5 spring.datasource.username=admin
6 spring.datasource.password=admin123
7
8 # #-----JPA-ORM Properties-----
9 spring.jpa.show-sql=true
10 spring.jpa.hibernate.ddl-auto=update
11 spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
12 spring.jpa.properties.hibernate.format_sql=true
13
14 logging.level.root=DEBUG
```

Docker

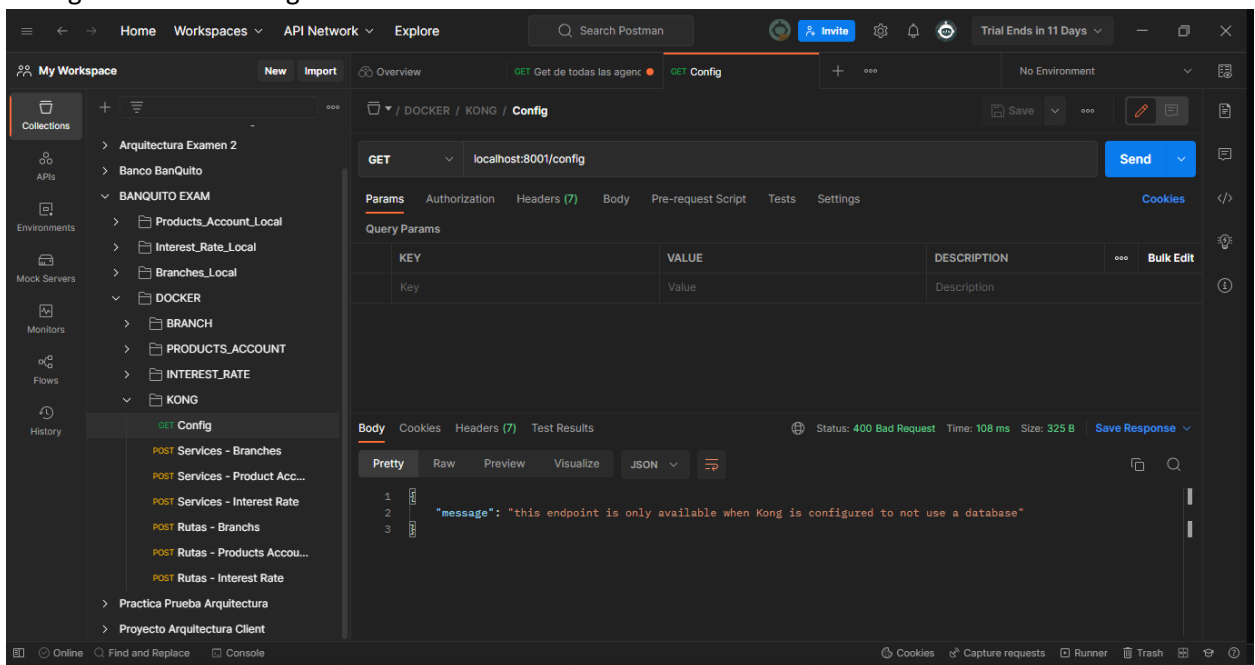


Postman

- Estructura de las carpetas



- Configuraciones de Kong



Service – Branches

The screenshot shows the Postman interface with a workspace named "My Workspace". The left sidebar displays a collection of folders and items, including "Arquitectura Examen 2", "Banco BanQuito", "BANQUITO EXAM", "Products_Account_Local", "Interest_Rate_Local", "Branches_Local", "DOCKER", "BRANCH", "PRODUCTS_ACCOUNT", "INTEREST_RATE", "KONG", "GET Config", "POST Services - Branches", "POST Services - Product Acc...", "POST Services - Interest Rate", "POST Rutas - Branches", "POST Rutas - Products Accou...", "POST Rutas - Interest Rate", "Practica Prueba Arquitectura", and "Proyecto Arquitectura Client". The main panel shows a POST request to "localhost:8001/services". The request body is a JSON object with the following structure:

```
1 {
2   "name": "branches",
3   "url": "http://branches:8000/api/v1/branches"
4 }
```

The response is displayed in the "Body" tab, showing a JSON object with the following structure:

```
1 {
2   "updated_at": 1678416258,
3   "port": 8000,
4   "client_certificate": null,
5   "tls_verify": null,
6   "tls_verify_depth": null,
7   "connect_timeout": 60000,
8   "read_timeout": 60000,
9   "name": "branches",
}
```

Service – Products Account

The screenshot shows the Postman interface with a workspace named "My Workspace". The left sidebar displays a collection of folders and items, including "Arquitectura Examen 2", "Banco BanQuito", "BANQUITO EXAM", "Products_Account_Local", "Interest_Rate_Local", "Branches_Local", "DOCKER", "BRANCH", "PRODUCTS_ACCOUNT", "INTEREST_RATE", "KONG", "GET Config", "POST Services - Branches", "POST Services - Product Acc...", "POST Services - Interest Rate", "POST Rutas - Branches", "POST Rutas - Products Accou...", "POST Rutas - Interest Rate", "Practica Prueba Arquitectura", and "Proyecto Arquitectura Client". The main panel shows a POST request to "localhost:8001/services". The request body is a JSON object with the following structure:

```
1 {
2   "name": "products-accounts",
3   "url": "http://products-accounts:9000/api/v1/productsaccounts"
4 }
```

The response is displayed in the "Body" tab, showing a JSON object with the following structure:

```
1 {
2   "updated_at": 1678416275,
3   "port": 9000,
4   "client_certificate": null,
5   "tls_verify": null,
6   "tls_verify_depth": null,
7   "connect_timeout": 60000,
8   "read_timeout": 60000,
9   "name": "products-accounts",
}
```

Services - Interés Rate

The screenshot shows the Postman interface with a workspace named "My Workspace". The left sidebar displays a collection of folders and items, including "Arquitectura Examen 2", "Banco Banquito", "BANQUITO EXAM", "Products_Account_Local", "Interest_Rate_Local", "Branches_Local", "DOCKER", "BRANCH", "PRODUCTS_ACCOUNT", "INTEREST_RATE", and "KONG". The "INTEREST_RATE" folder is selected, showing a list of items: "GET Config", "POST Services - Branches", "POST Services - Product Acc...", "POST Services - Interest Rate", "POST Rutas - Branches", "POST Rutas - Products Accou...", and "POST Rutas - Interest Rate". The "POST Services - Interest Rate" item is selected.

The main panel shows a POST request to `localhost:8001/services`. The request body is a JSON object:

```
1 {
2   "name": "interest-rate",
3   "url": "http://products-accounts:9000/api/v1/interestrates"
4 }
```

The response is a JSON object:

```
1 {
2   "updated_at": 1678416331,
3   "port": 9000,
4   "client_certificate": null,
5   "tls_verify": null,
6   "tls_verify_depth": null,
7   "connect_timeout": 60000,
8   "read_timeout": 60000,
9   "name": "interest-rate",
}
```

The status is 201 Created, Time: 14 ms, Size: 631 B. The response is saved.

Rutas de Branches

The screenshot shows the Postman interface with a workspace named "My Workspace". The left sidebar displays a collection of folders and items, including "Arquitectura Examen 2", "Banco Banquito", "BANQUITO EXAM", "Products_Account_Local", "Interest_Rate_Local", "Branches_Local", "DOCKER", "BRANCH", "PRODUCTS_ACCOUNT", "INTEREST_RATE", and "KONG". The "INTEREST_RATE" folder is selected, showing a list of items: "GET Config", "POST Services - Branches", "POST Services - Product Acc...", "POST Services - Interest Rate", "POST Rutas - Branches", "POST Rutas - Products Accou...", and "POST Rutas - Interest Rate". The "POST Rutas - Branches" item is selected.

The main panel shows a POST request to `localhost:8001/services/branches/routes`. The request body is a JSON object:

```
1 {
2   "name": "branches-service-routes",
3   "paths": ["/branch"],
4   "strip_path": true
5 }
```

The response is a JSON object:

```
1 {
2   "updated_at": 1678416364,
3   "request_buffering": true,
4   "response_buffering": true,
5   "https_redirect_status_code": 426,
6   "strip_path": true,
7   "preserve_host": false,
8   "protocols": [
9     "http",

```

The status is 201 Created, Time: 21 ms, Size: 728 B. The response is saved.

Rutas – Products Accounts

Postman interface showing a POST request to `localhost:8001/services/products-accounts/routes`. The request body is a JSON object:

```
1 {
2   "name": "products-accounts-service-routes",
3   "paths": ["/products-accounts"],
4   "strip_path": true
5 }
```

The response is a JSON object:

```
1 {
2   "updated_at": 1678416399,
3   "request_buffering": true,
4   "response_buffering": true,
5   "https_redirect_status_code": 426,
6   "strip_path": true,
7   "preserve_host": false,
8   "protocols": [
9     "http",
10  ]
11 }
```

Rutas – Interest Rate

Postman interface showing a POST request to `localhost:8001/services/interest-rate/routes`. The request body is a JSON object:

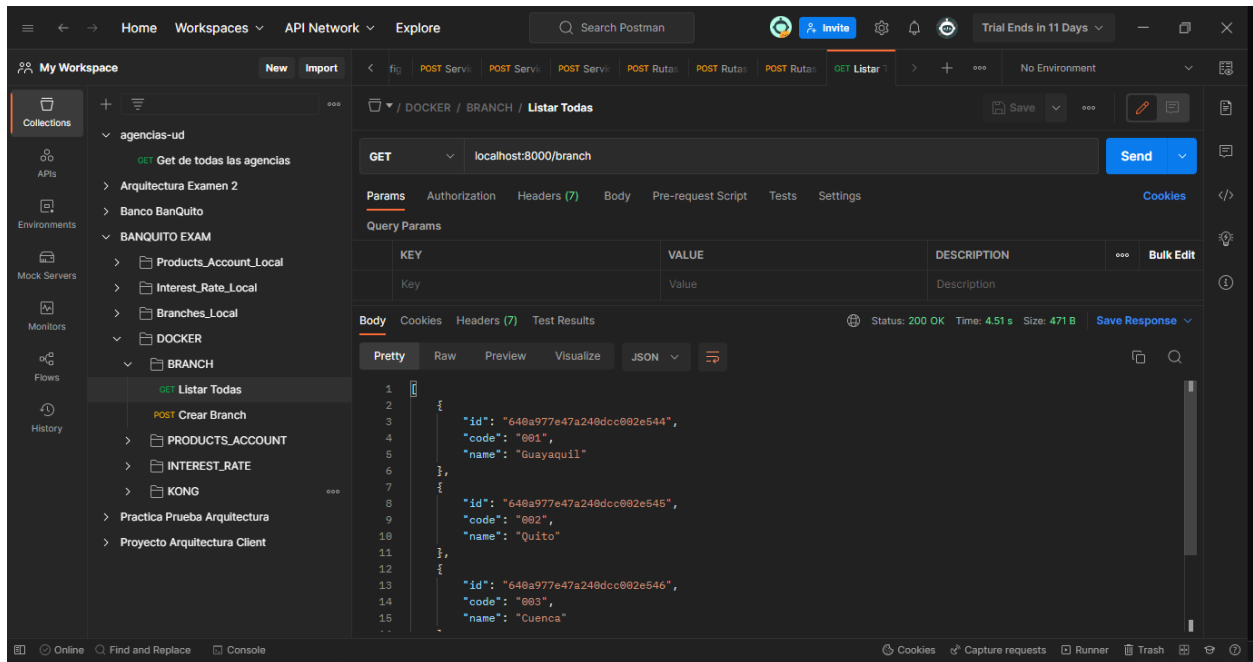
```
1 {
2   "name": "interest-rate-service-routes",
3   "paths": ["/interest-rate"],
4   "strip_path": true
5 }
```

The response is a JSON object:

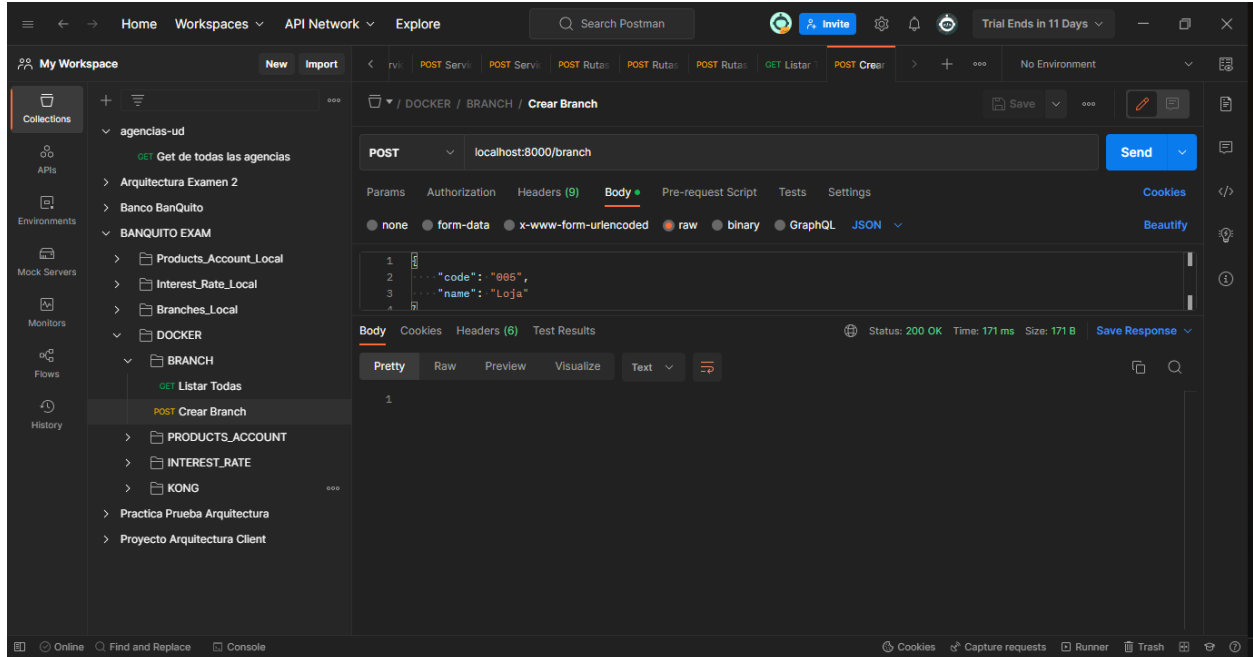
```
1 {
2   "updated_at": 1678416414,
3   "request_buffering": true,
4   "response_buffering": true,
5   "https_redirect_status_code": 426,
6   "strip_path": true,
7   "preserve_host": false,
8   "protocols": [
9     "http",
10  ]
11 }
```


- Implementar las siguientes llamadas en Postman para Branches:

Listar todas

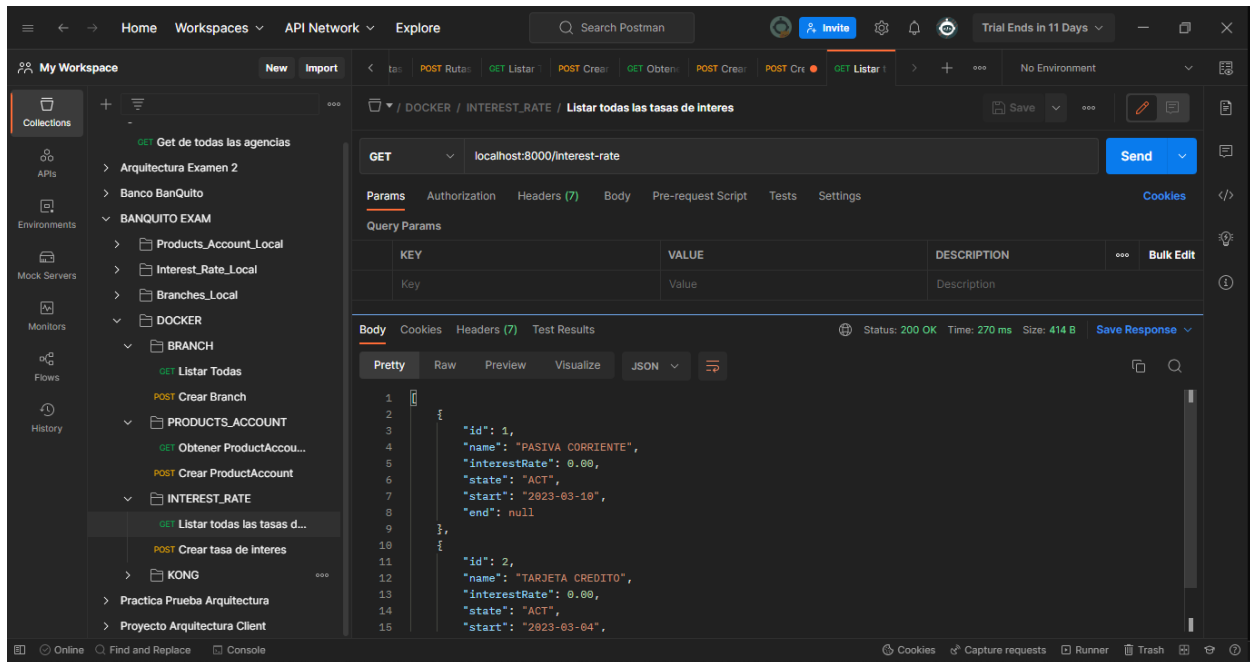


Crear Branch

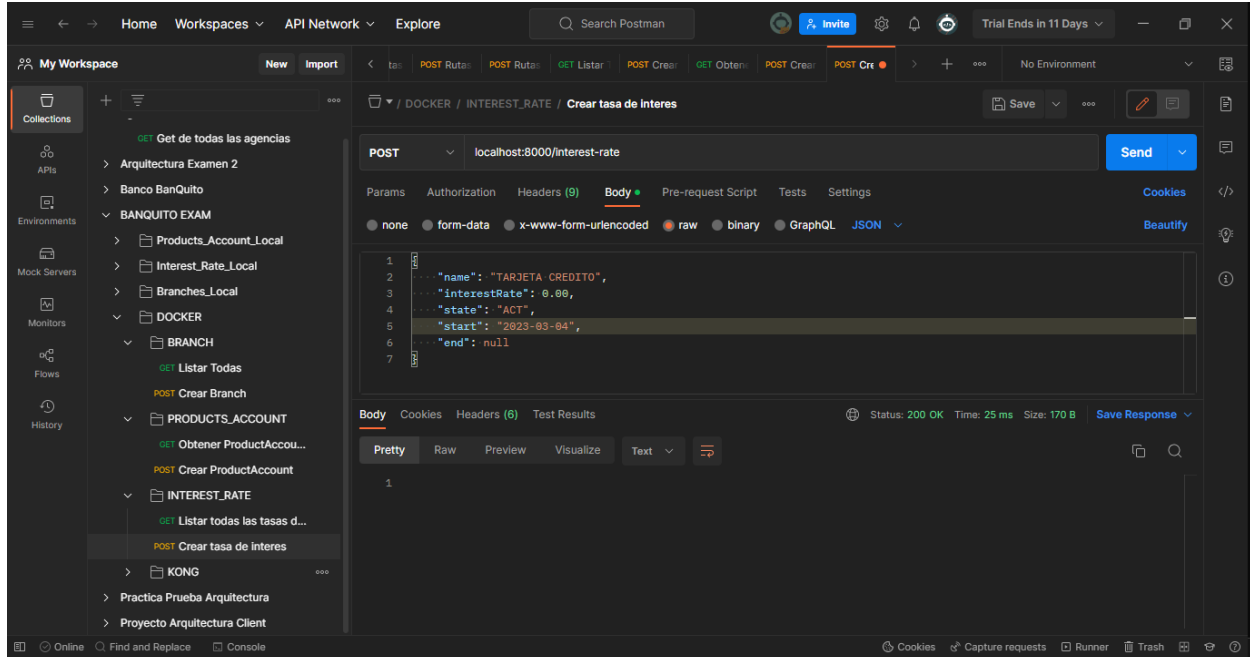


- Implementar las siguientes llamadas en Postman para Accounts:

Listar todas las tasas de interés



Crear tasa de interés



Crear ProductAccount

The screenshot shows the Postman interface with a workspace named 'My Workspace'. The left sidebar displays a collection of folders and items. The main panel shows a POST request to 'localhost:8000/products-accounts'. The request body is a JSON object with the following fields: 'id' (CORR2024), 'name' (Cuenta Corriente 2024), 'description' (Cuenta corriente digital para el 2004, maneja cheques y no paga interes.), 'minimumBalance' (10.00), 'payInterest' (N), 'acceptsChecks' (Y), and 'state' (ACT). The response status is 200 OK, with a time of 281 ms and a size of 171 B.

```
POST localhost:8000/products-accounts

{
  "id": "CORR2024",
  "name": "Cuenta Corriente 2024",
  "description": "Cuenta corriente digital para el 2004, maneja cheques y no paga interes.",
  "minimumBalance": 10.00,
  "payInterest": "N",
  "acceptsChecks": "Y",
  "state": "ACT"
}
```

Status: 200 OK Time: 281 ms Size: 171 B

Obtener ProductAccount por ID

The screenshot shows the Postman interface with a workspace named 'My Workspace'. The left sidebar displays a collection of folders and items. The main panel shows a GET request to 'localhost:8000/products-accounts/CORR2023'. The response status is 200 OK, with a time of 255 s and a size of 446 B. The response body is a JSON object with the following fields: 'id' (CORR2023), 'name' (Cuenta Corriente 2022), 'description' (Cuenta corriente completamente autogestionada y digital; no maneja cheques ni paga interes.), 'minimumBalance' (0.00), 'payInterest' (N), 'acceptsChecks' (N), and 'state' (ACT).

```
GET localhost:8000/products-accounts/CORR2023

{
  "id": "CORR2023",
  "name": "Cuenta Corriente 2022",
  "description": "Cuenta corriente completamente autogestionada y digital; no maneja cheques ni paga interes.",
  "minimumBalance": 0.00,
  "payInterest": "N",
  "acceptsChecks": "N",
  "state": "ACT"
}
```

Status: 200 OK Time: 255 s Size: 446 B