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		100 %		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		96 %		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

```
print('############################ScriptStart#####################################');

db = db.getSiblingDB('branche');
db.createCollection('branches');

var brancheColl = db.getCollection('branches');

brancheColl.insertMany([
    {"code": "001", "name": "Guayaquil"},
    {"code": "002", "name": "Quito"},
    {"code": "003", "name": "Cuenca"},
    {"code": "004", "name": "Ambato"}
]);

print('############################ScriptEnd#####################################');
```

Base de datos de postgres

```
create table INTEREST_RATE (
   ID_INTEREST_RATE     SERIAL not null,
   NAME                 VARCHAR(50)          not null,
   INTEREST_RATE        NUMERIC(2,2)         not null,
   STATE                VARCHAR(3)           not null
      constraint CKC_STATE_INTEREST check (STATE in ('ACT','INA')),
   START_DATE           DATE                 not null,
   END_DATE             DATE                 null,
   constraint PK_INTEREST_RATE primary key (ID_INTEREST_RATE)
);

create table PRODUCT_ACCOUNT (
   ID_PRODUCT_ACCOUNT   VARCHAR(16)          not null,
   NAME                 VARCHAR(50)          not null,
   DESCRIPTION          VARCHAR(500)         not null,
   MINIMUN_BALANCE      NUMERIC(10,2)        not null,
   PAY_INTEREST         VARCHAR(1)           not null
      constraint CKC_PAY_INTEREST_PRODUCT_ check (PAY_INTEREST in ('Y','N')),
   ACCEPTS_CHECKS       VARCHAR(1)           not null
      constraint CKC_ACCEPTS_CHECKS_PRODUCT_ check (ACCEPTS_CHECKS in ('Y','N')),
   STATE                VARCHAR(3)           not null
      constraint CKC_STATE_PRODUCT_ check (STATE in ('DRA','ACT','INA')),
   CREATION_DATE        DATE                 not null,
   constraint PK_PRODUCT_ACCOUNT primary key (ID_PRODUCT_ACCOUNT)
);


insert into interest_rate values(default, 'PASIVA CORRIENTE', 0.0, 'ACT', NOW(), null);
insert into product_account values('CORR2023', 'Cuenta Corriente 2022', 'Cuenta corriente completamente autoegestionada y digital; no maneja cheques ni paga interes.', 0.0, 'N', 'N', 'ACT', NOW());
```

Docker – Compose

```yaml
version: '2'

services:

  kong-database:
    image: postgres:latest
    hostname: kong-database
    container_name: kong-database
    environment:
      POSTGRES_USER: "kong"
      POSTGRES_DB: "kong"
      POSTGRES_PASSWORD: "kong"
    ports:
      - "5432:5432"

  kong-bootstrap:
    image: kong:3.1.1-alpine
    hostname: kong
    container_name: kong-bootstrap
    depends_on:
      - kong-database
    environment:
      KONG_DATABASE: "postgres"
      KONG_PG_HOST: "kong-database"
      KONG_PG_DATABASE: "kong"
      KONG_PG_USER: "kong"
      KONG_PG_PASSWORD: "kong"
    command: "kong migrations bootstrap"
    restart: "on-failure"

  kong:
    image: kong:3.1.1-alpine
    hostname: kong
    container_name: kong
    restart: always
    depends_on:
      - kong-bootstrap
    environment:
      KONG_DATABASE: "postgres"
      KONG_PG_HOST: "kong-database"
      KONG_PG_DATABASE: "kong"
      KONG_PG_USER: "kong"
      KONG_PG_PASSWORD: "kong"
      KONG_PROXY_ACCESS_LOG: /dev/stdout
      KONG_ADMIN_ACCESS_LOG: /dev/stdout
      KONG_PROXY_ERROR_LOG: /dev/stdout
      KONG_ADMIN_ERROR_LOG: /dev/stdout
      KONG_ADMIN_LISTEN: "0.0.0.0:8001, 0.0.0.0:8444 ssl"
    command: "kong start"
    ports:
      - "8000:8000"
      - "8443:8443"
      - "8001:8001"
      - "8444:8444"

  products-accounts:
    build:
      context: ./products-accounts
    container_name: products-accounts
    hostname: products-accounts
    depends_on:
      - postgres

  branches:
    build:
      context: ./branches
    container_name: branches
    hostname: branches
    depends_on:
      - mongo

  postgres:
    image: postgres:latest
    hostname: postgres-banquito
    container_name: postgres-banquito
    environment:
      - POSTGRES_USER=admin
      - POSTGRES_PASSWORD=admin123
      - POSTGRES_DB=products_accounts
    volumes:
      - ./postgres-init/init.sql:/docker-entrypoint-initdb.d/init.sql

  mongo:
    image: mongo:latest
    hostname: mongo-banquito
    container_name: mongo-banquito
    environment:
      - MONGO_INITDB_ROOT_USERNAME=admin
      - MONGO_INITDB_ROOT_PASSWORD=admin123
      - MONGO_INITDB_DATABASE=branche
    volumes:
      - ./mongo-init/init.js:/docker-entrypoint-initdb.d/init.js

networks:
  default:
    name: kong-net
```

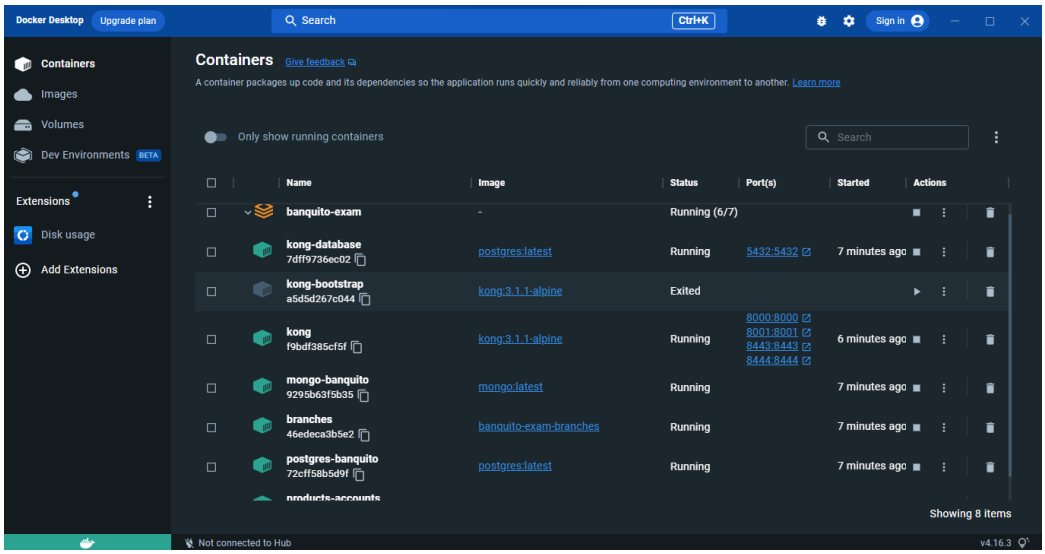
Application. Properties de Branches

```
server.port=8000

#-------------------- MONGO Connection docker----------------
spring.data.mongodb.uri=mongodb://admin:admin123@mongo-banquito:27017/branche?authSource=admin

logging.level.root=DEBUG
```

Application. Properties de Products – Accounts

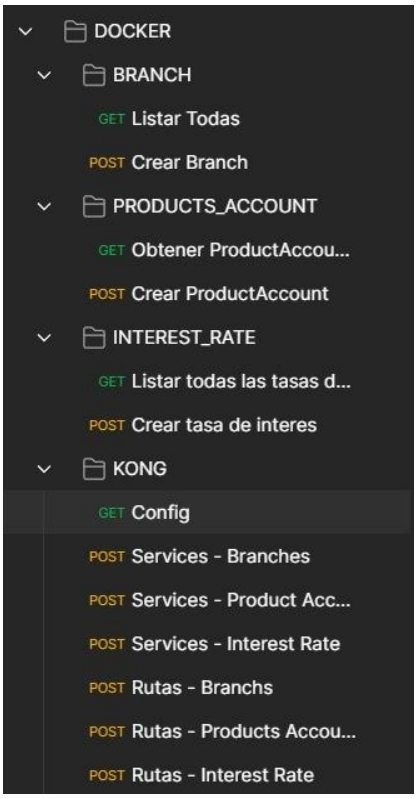
```
server.port=9000

#-------------------- PostgreSQL Connection COMPOSE DB----------------
spring.datasource.url=jdbc:postgresql://postgres-banquito:5432/products_accounts
spring.datasource.username=admin
spring.datasource.password=admin123

# #--------------------JPA-ORM Properties-----------------
spring.jpa.show-sql=true
spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
spring.jpa.properties.hibernate.format_sql=true

logging.level.root=DEBUG
```

Docker

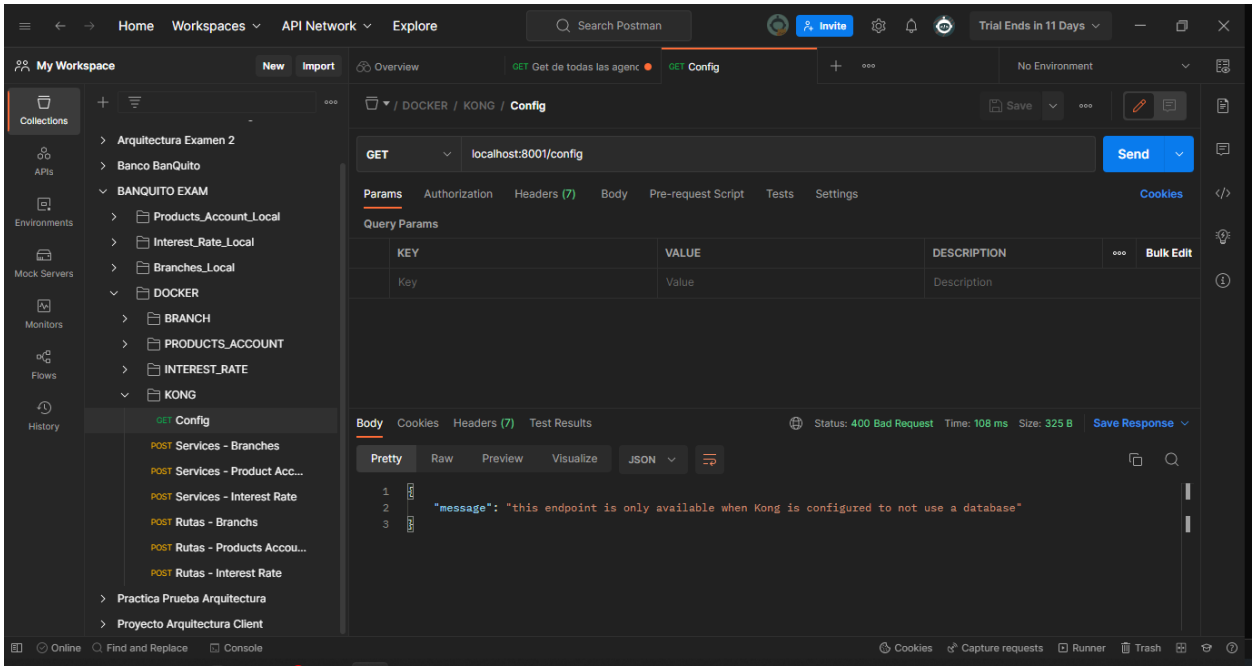


Postman

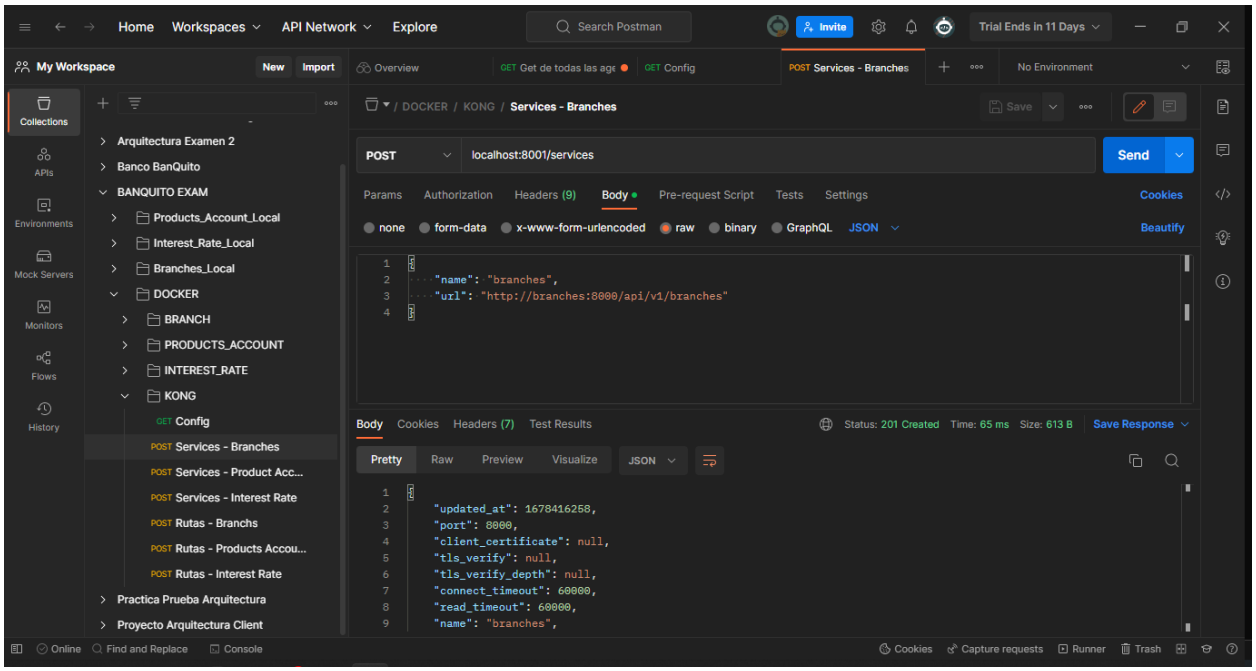
- Estructura de las carpetas



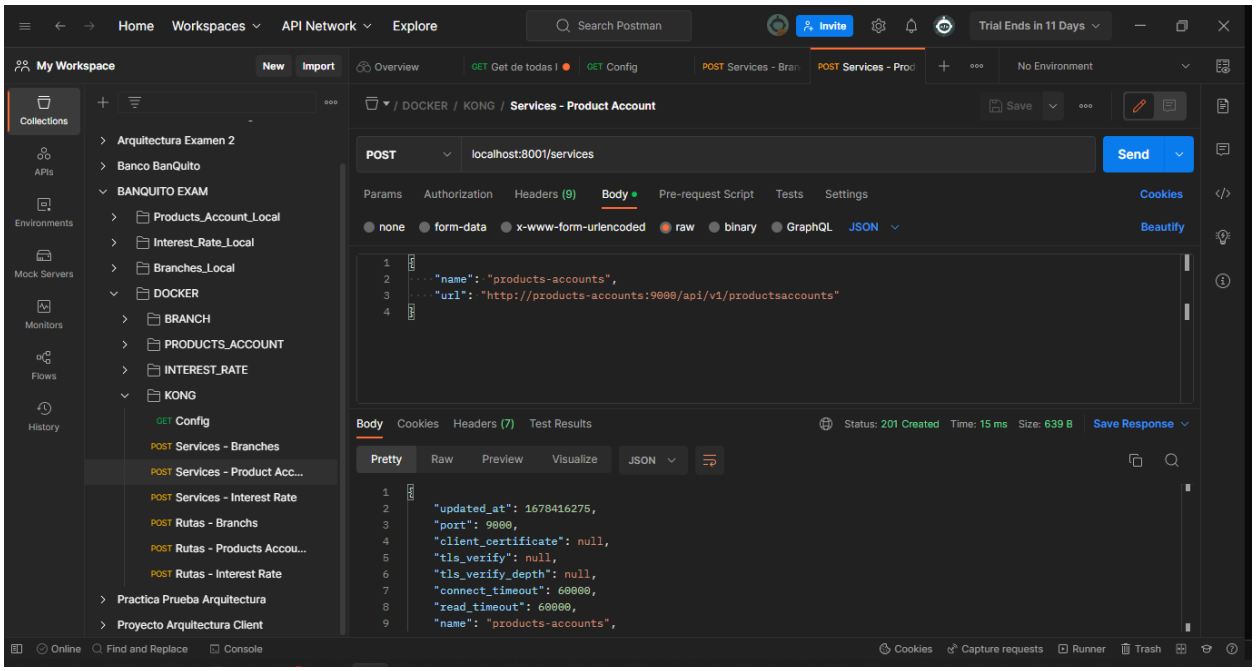
- Configuraciones de Kong



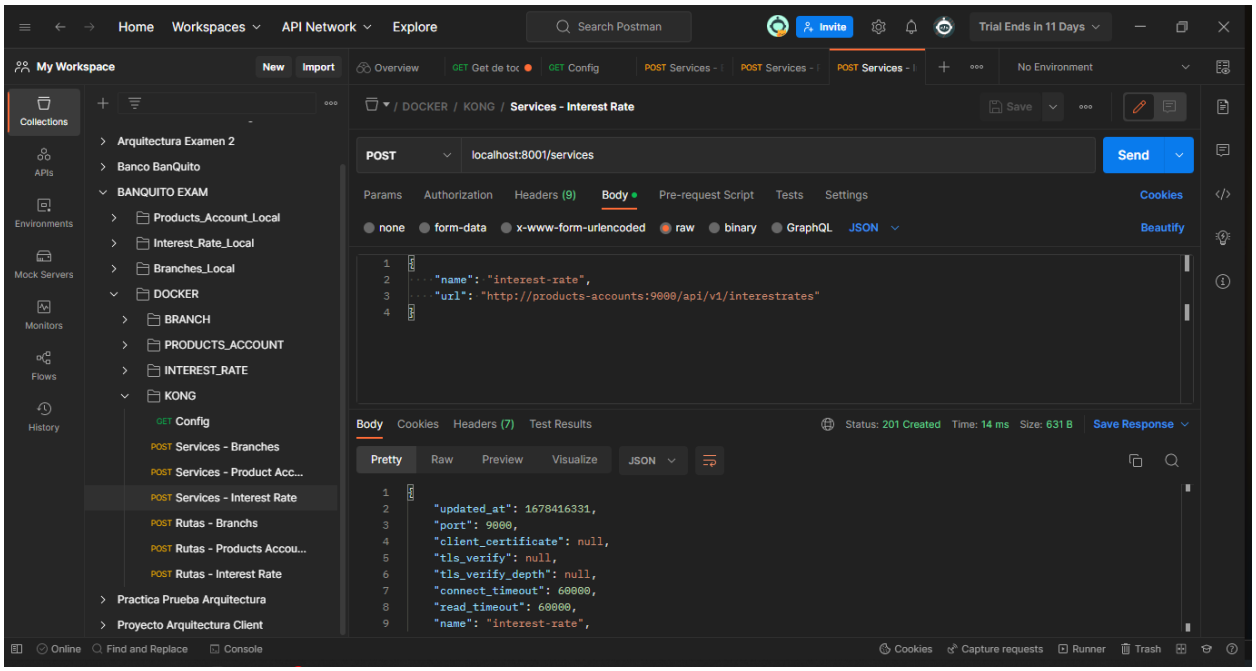
Service – Branches



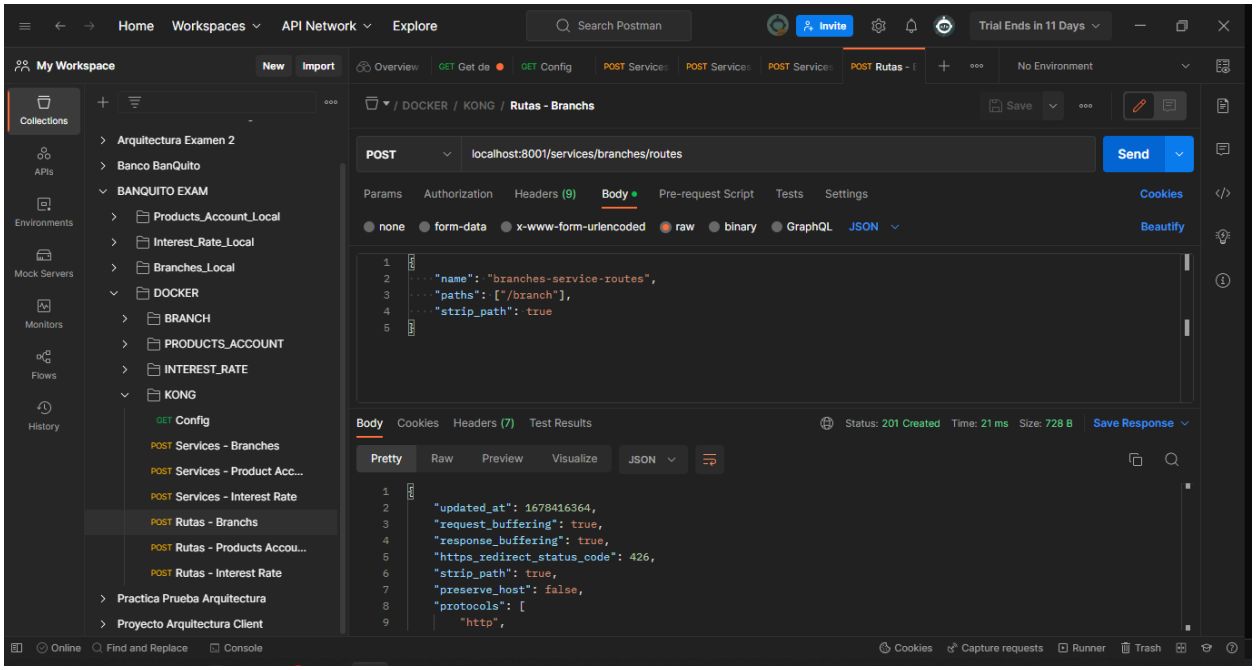
Service – Products Account



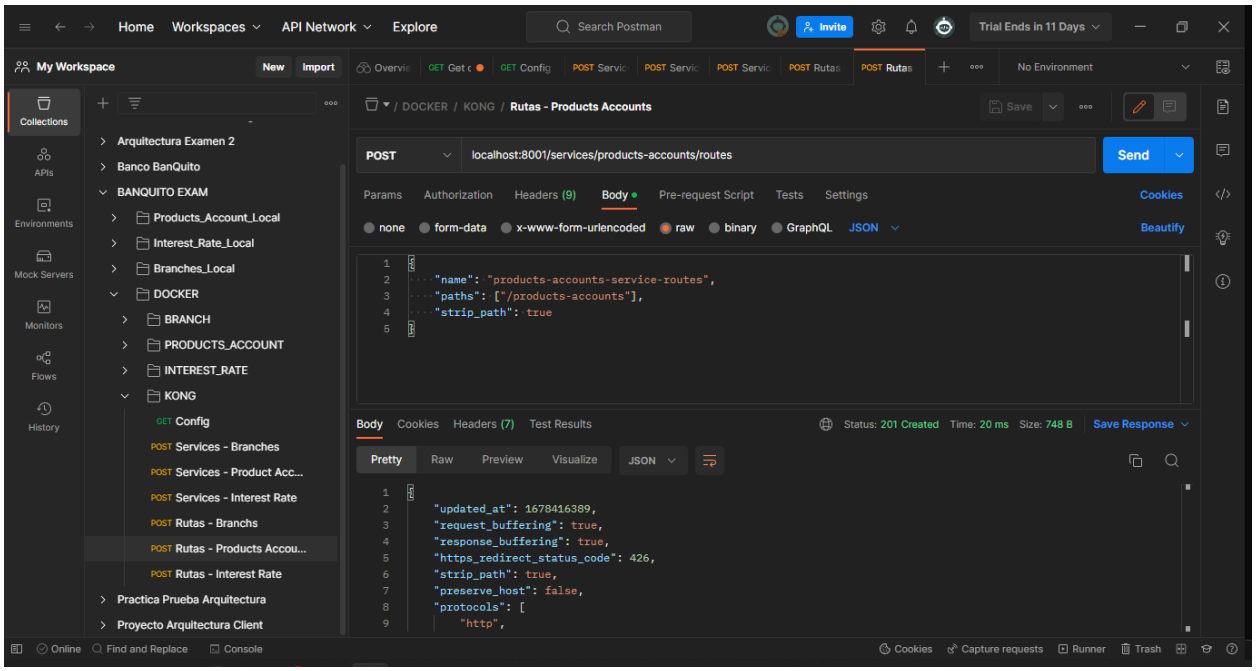
Services - Interés Rate



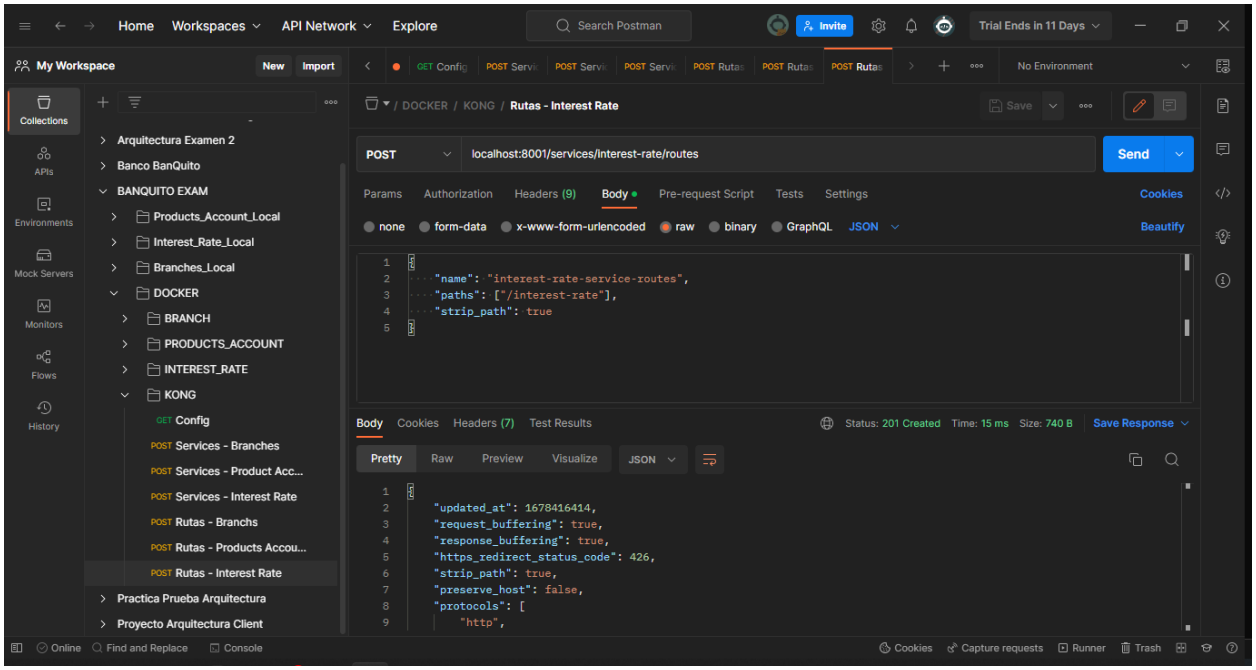
Rutas de Branches



Rutas – Products Accounts

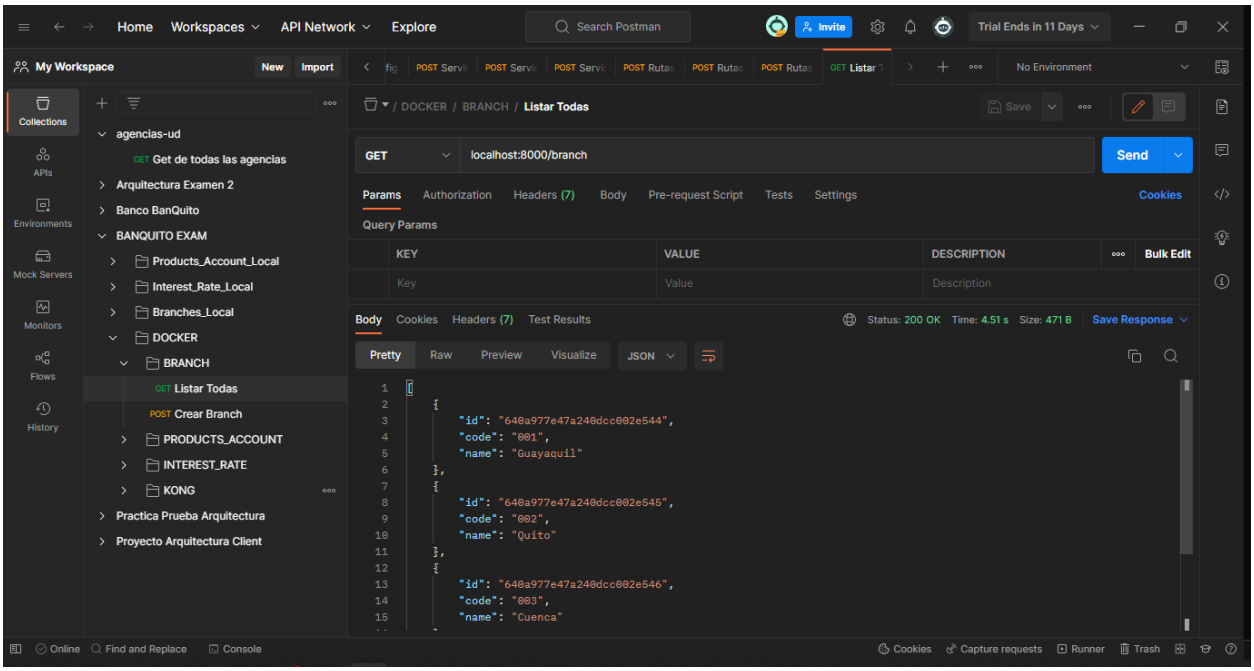


Rutas – Interest Rate

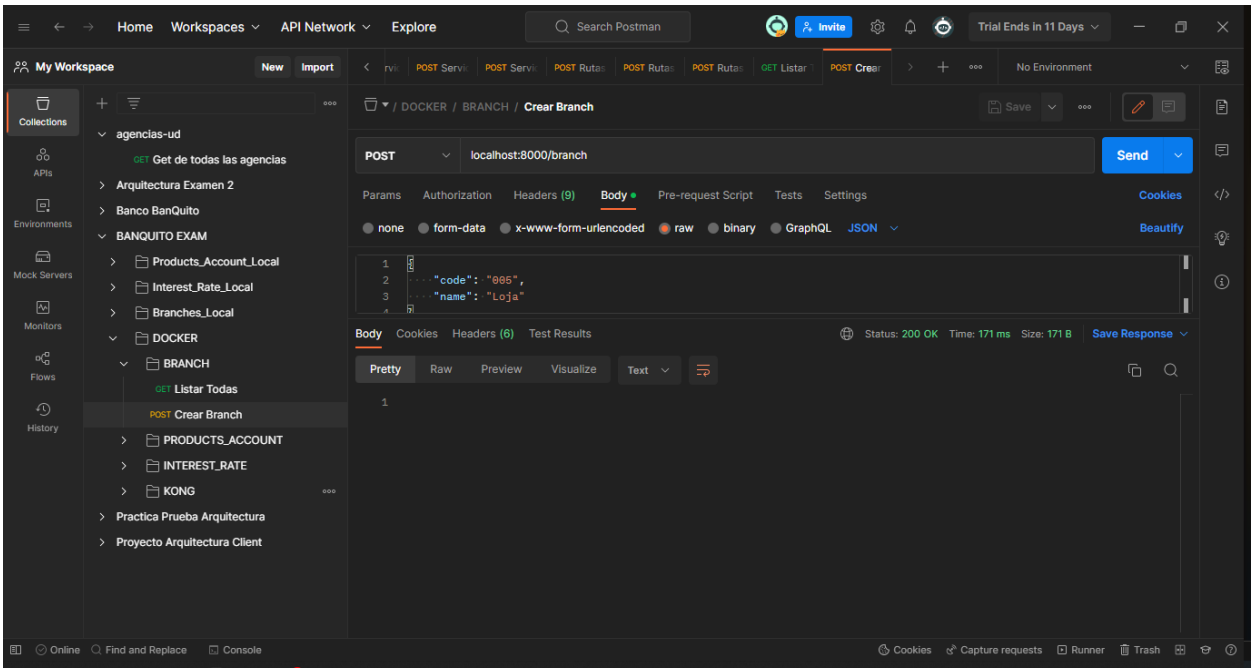


- Implementar las siguientes llamadas en Postman para Branches:

Listar todas

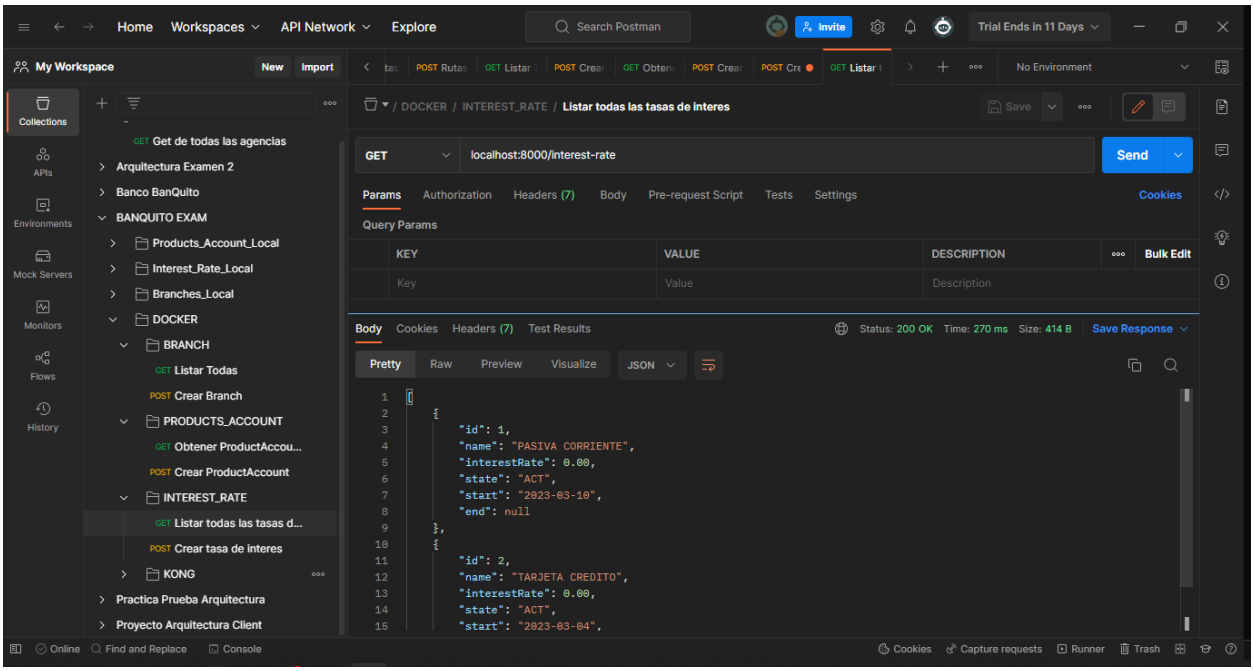


Crear Branch

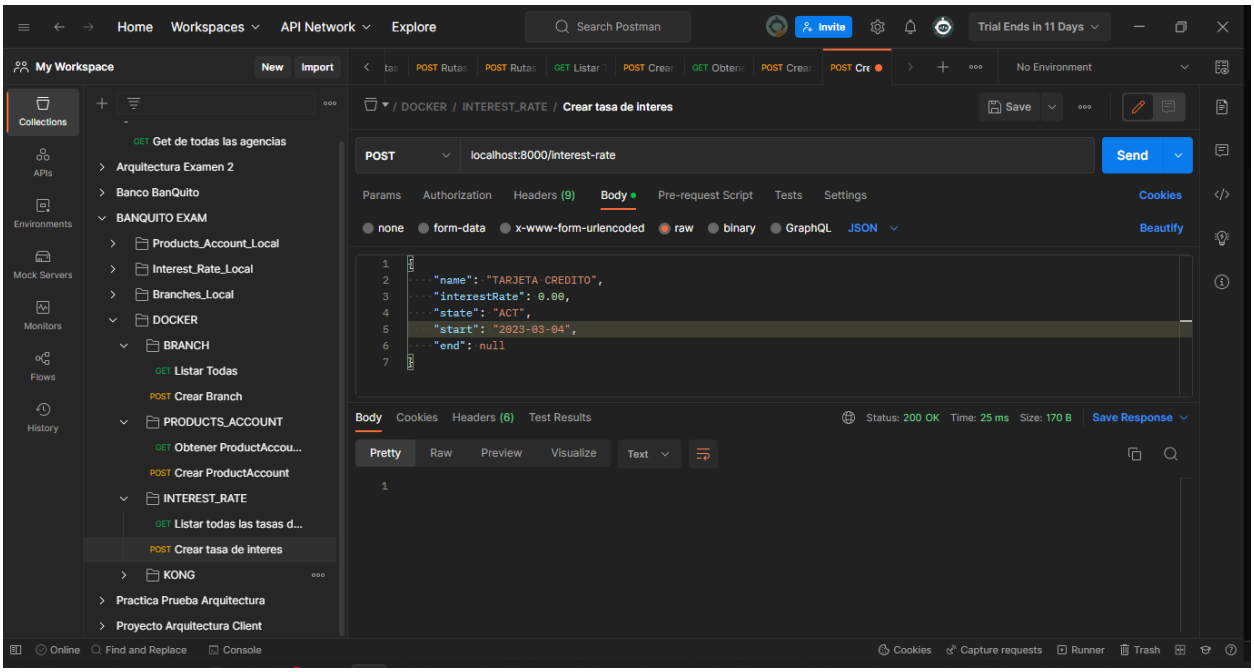


- Implementar las siguientes llamadas en Postman para Accounts:

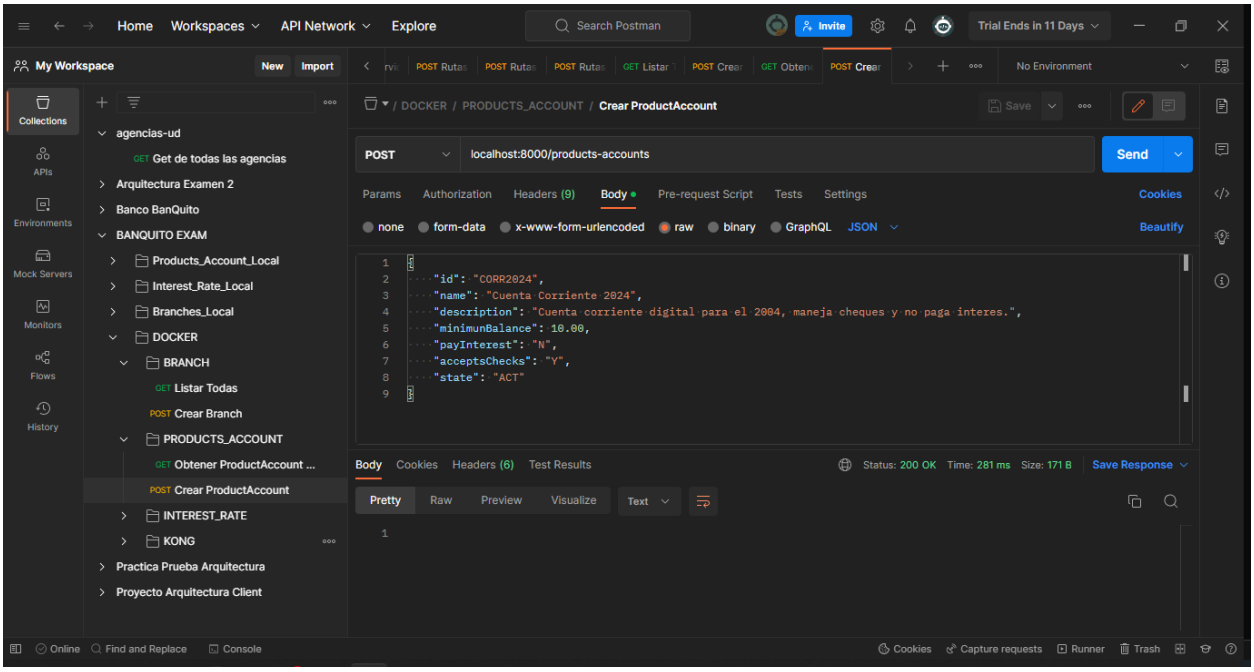
Listar todas las tasas de interés



Crear tasa de interés



Crear ProductAccount



Obtener ProductAccount por ID