

Spring Boot demo app:

Requirements:

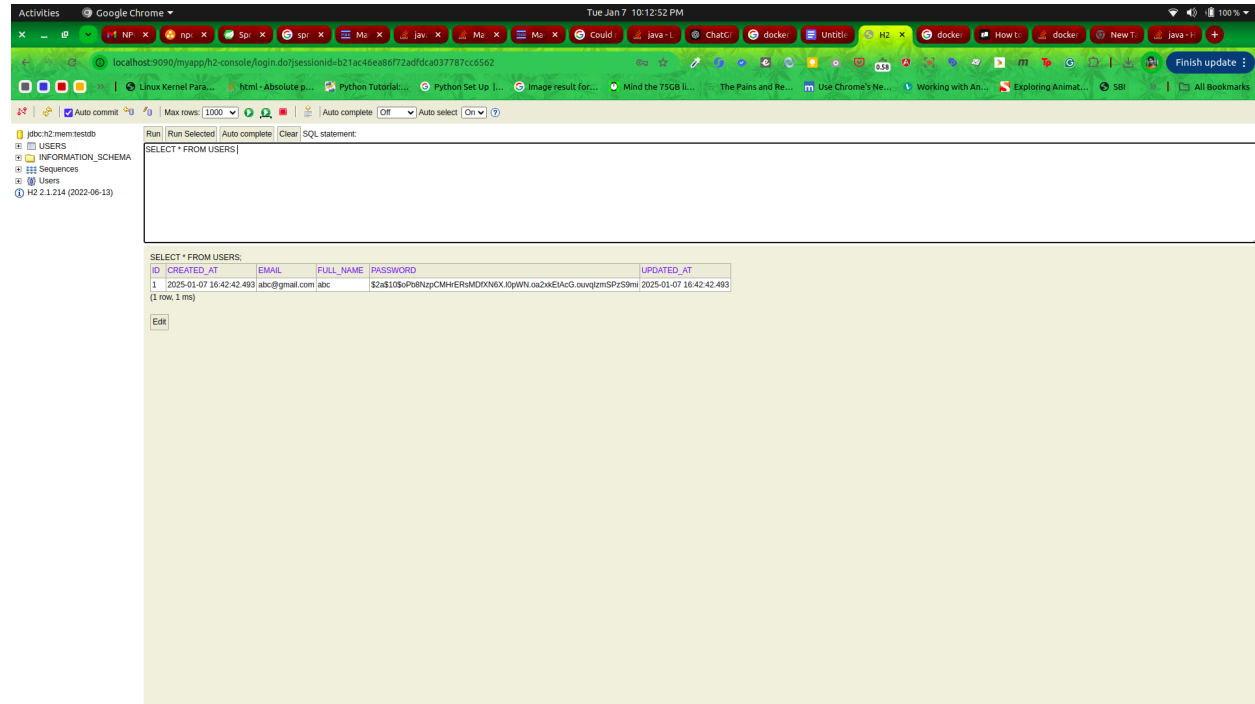
1. Java 8
2. Linux
3. Docker
4. Git (optional) - git commits are done

Functional Features:

1. JWT Authentication:
 - a. Working context: Works based on signup user via signup apis
2. Rate Limiting
 - a. Working context: based all over application per IP for X seconds of rate limited for Y number of request.
 - i. X and Y configurable in
`/demo/src/main/java/com/example/demo/filter/RateLimitingFilter.java`

Non-Functional Features:

1. Spring boot based: version 2.7.18
 - a. Gradle - v8
2. Embedded tomcat
 - a. Runs on 9090
3. POSTMAN collection PFA in source code zip
4. Rest controllers - json apis
5. JWT auth based via user signup apis
6. H2 memory database used:
 - a. Username: sa
 - b. Password: sa
 - c. Url: <http://localhost:9090/myapp/h2-console/login.do>
 - d. Db url to be used: `jdbc:h2:mem:testdb`
 - e. Screenshot



- f.
7. Rate limit configs available in code for
 - a. `MIN_IN_SECONDS_TO_RATE_LIMIT` : defines the seconds until the rate limit is over.
 - b. `MAX_REQUESTS_PER_MINUTE` defines the count threshold of requests when rate limit starts
 8. Dockerization of app:
 - a. The gradlew of the project will make the runnable jar in docker build stage and then docker will run the jar exposing the apis of the projects

SETUP:

run can be done in 2 ways

1. Via commandline-start.sh
 - a. Command: `sh commandline-start.sh`
 - b. Brief: this shall run the project on linux command without docker
2. Via docker-start.sh
 - a. Command: `sh docker-start.sh`
 - b. Brief: this shall run the project on linux command line with docker
 - c. Requirement:
 - i. Docker has to be installed

The screenshot shows the IntelliJ IDEA Ultimate Edition interface. The top bar indicates the date and time: Tue Jan 7 10:11:55 PM. The main window displays a project named 'demo' with a file explorer on the left showing the project structure: 'src/main/java/com/example/demo/filter/RateLimitingFilter.java'. The 'Terminal' tab is active, showing the output of a Docker build command. The build process starts with 'docker: default' and proceeds through various steps, including loading build definitions, transferring the Dockerfile, and building the image. The final output shows the image being pushed to Docker Hub and the container being started. The terminal output is as follows:

```
[c] Building 77.1s (1s/1s) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring Dockerfile: 505B
=> [internal] load metadata for docker.io/library/openjdk:8-jre-alpine
=> [internal] load metadata for docker.io/library/openjdk:8-jdk-alpine
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [stage-1 1/5] FROM docker.io/library/openjdk:8-jre-alpine@sha256:f362b16b878ef129cbe730f29685f937599c6a8bca3e44b51302958c9193
=> [build 1/7] FROM docker.io/library/openjdk:8-jdk-alpine@sha256:94792824d720f334027201713f93268c896944bc052416a0f2283343547b3
=> [internal] load build context
=> => transferring context: 18.22kB
=> [stage-2 1/7] WORKDIR /app
=> [stage-3 2/7] COPY gradlew gradlew.bat /app/
=> [stage-4 3/7] COPY gradle /app/gradle
=> [stage-5 4/7] COPY . /app/
=> [stage-6 5/7] RUN chmod +x gradlew
=> [stage-7 6/7] RUN ./gradlew build -x test
=> [stage-8 7/7] WORKDIR /app
=> [stage-9 1/1] COPY --from=build /app/build/libs/demo-0.0.1-SNAPSHOT.jar /app/app.jar
=> exporting to image
=> => exporting layers
=> => writing image sha256:965f9f6e030d880af317d0924b172d0b20dc7779913e0b9a335409f81d02a54
=> => naming to docker.io/library/hpcc1-demo-app
=> [stage-10 1/1] COMMIT docker.io/library/hpcc1-demo-app:latest
=> [stage-11 1/1] PUSH docker.io/library/hpcc1-demo-app:latest
=> [stage-12 1/1] RUN docker.io/library/hpcc1-demo-app:latest
:: Starting Boot :: (v2.7.18)

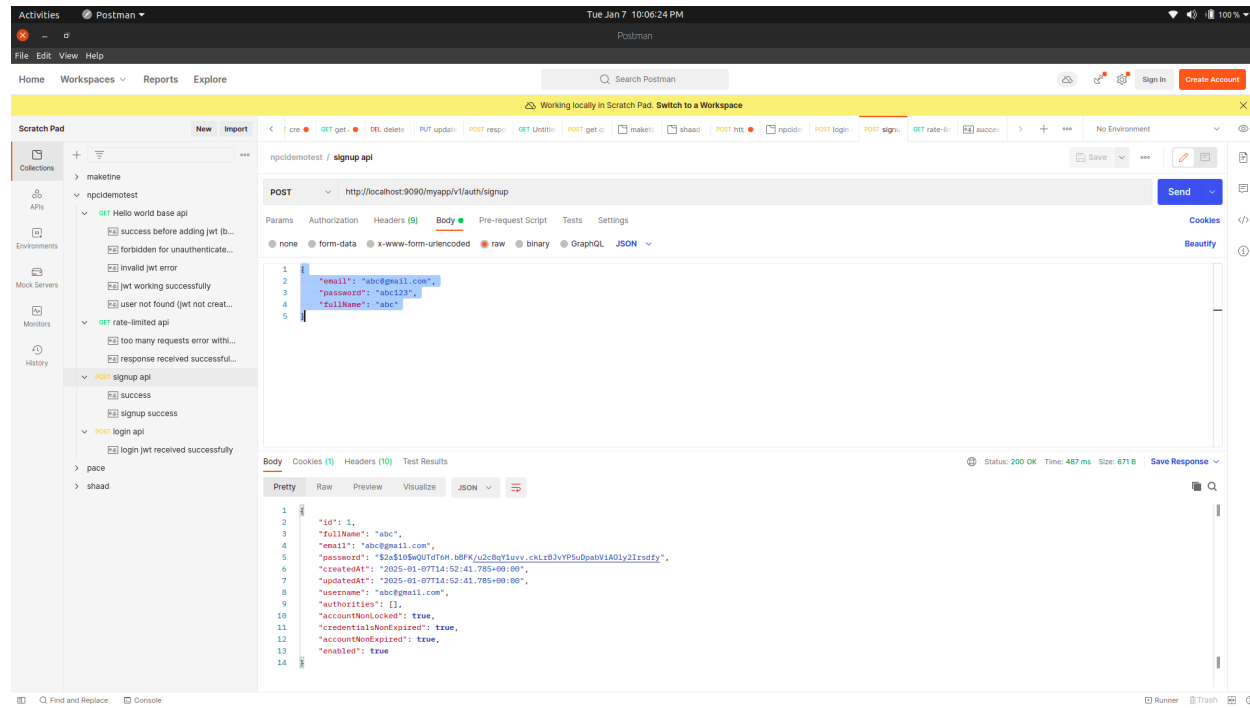
2025-01-07 16:35:44.544 INFO 1 --- [main] com.example.demo.DemoApplication : Starting DemoApplication using Java 1.8.0_212 on 040f38567c23 with PID 1 (/app/app.jar started by root in /app)
2025-01-07 16:35:44.547 INFO 1 --- [main] com.example.demo.DemoApplication : No active profile set, falling back to 1 default profile: "default"
2025-01-07 16:35:45.607 INFO 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
2025-01-07 16:35:45.726 INFO 1 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 48 ms. Found 1 JPA repository interfaces.
```

Project Port: 9090

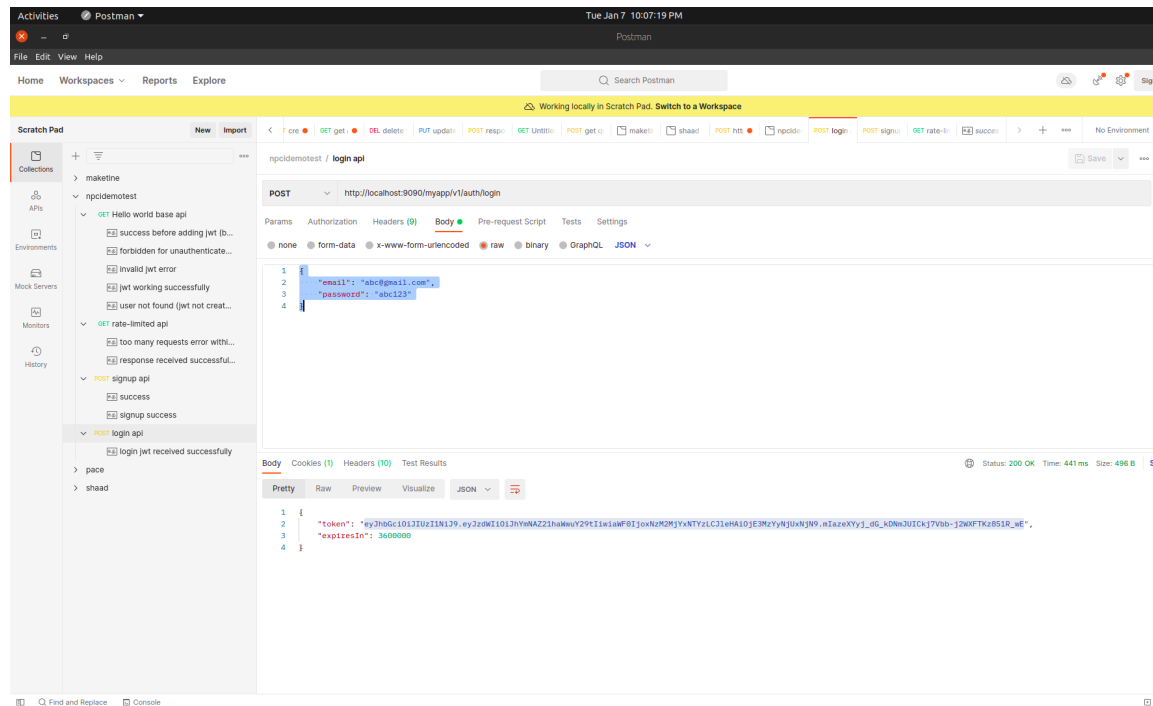
Project api base path: /myapp

APIs:

1. Signup API: to register a user for getting jwt after login:
 - a. /myapp/v1/auth/signup HTTP POST
 - b. Payload:
 - i. {
 - ii. "email": "abc@gmail.com",
 - iii. "password": "abc123",
 - iv. "fullName": "abc"
 - v. }

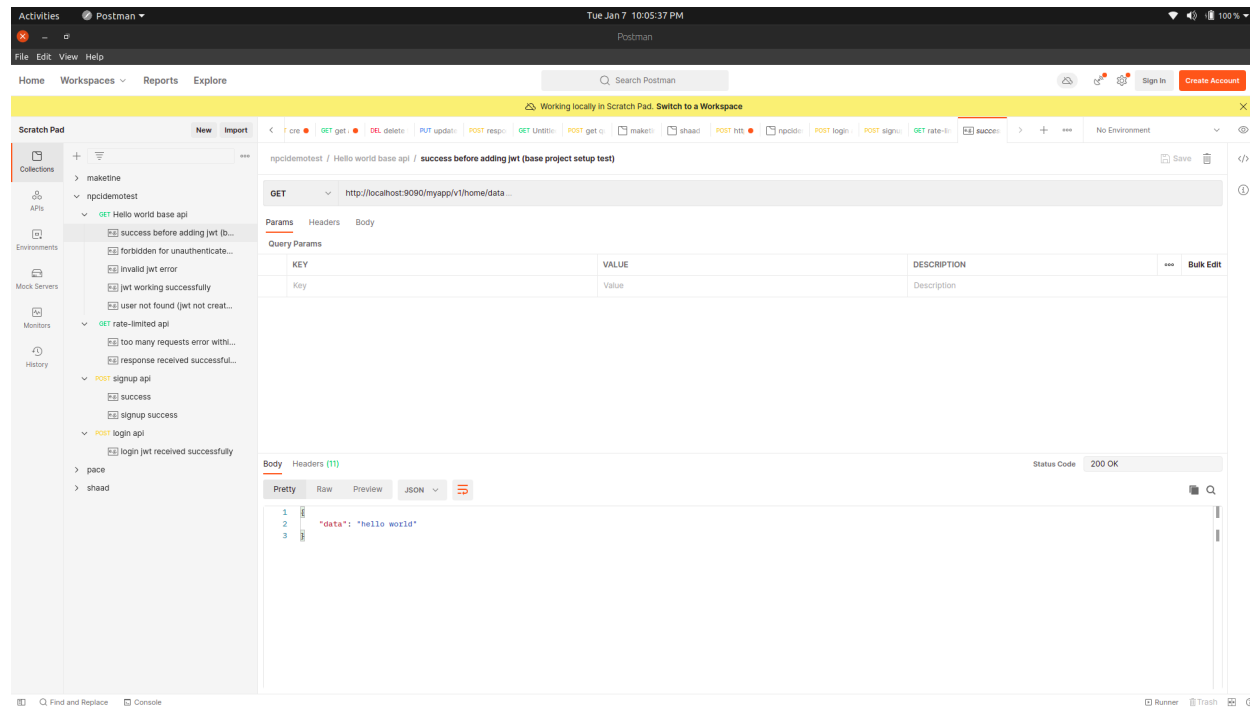


- c.
2. Login up - to get the JWT token:
 - a. /myapp/v1/auth/login HTTP POST
 - b. Payload:
 - i. {
 - ii. "email": "abc@gmail.com",
 - iii. "password": "abc123"
 - iv. }



- v.
3. Data api - generic api: HTTP GET

a. /myapp/v1/home/data - working with JWT token

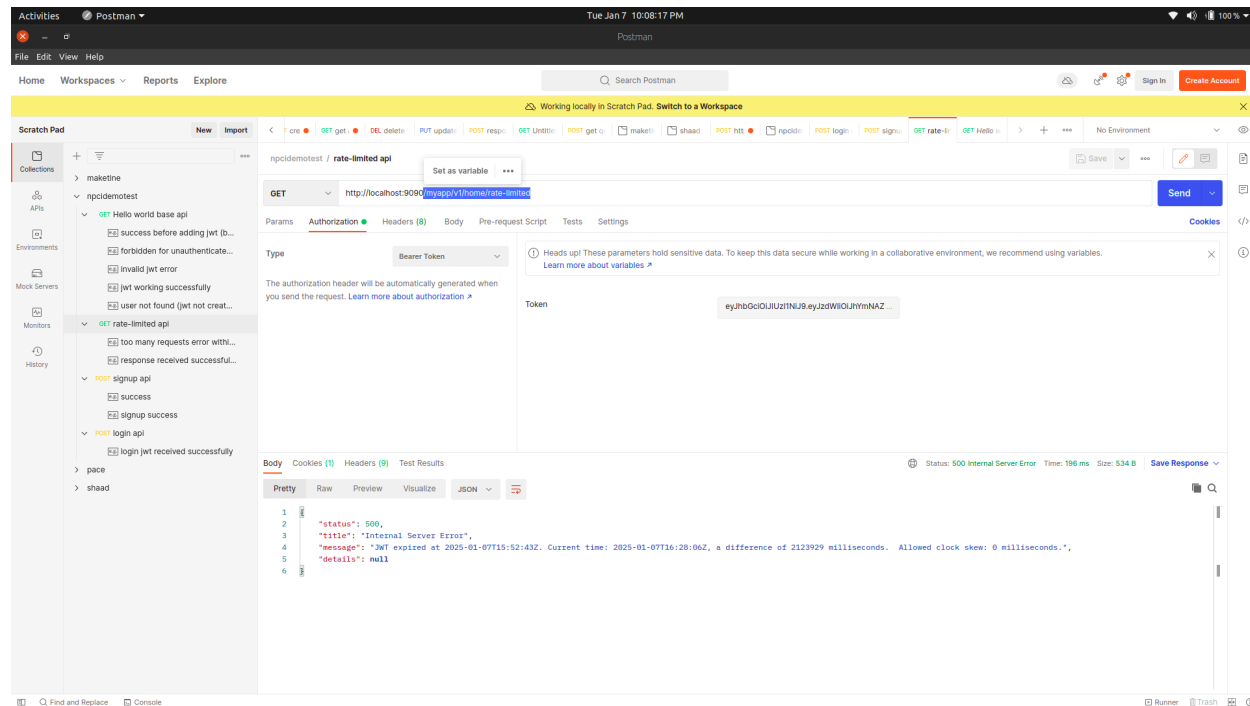


b.

4. Data api - invalid JWT error

a. HTTP GET

b. /myapp/v1/home/rate-limited

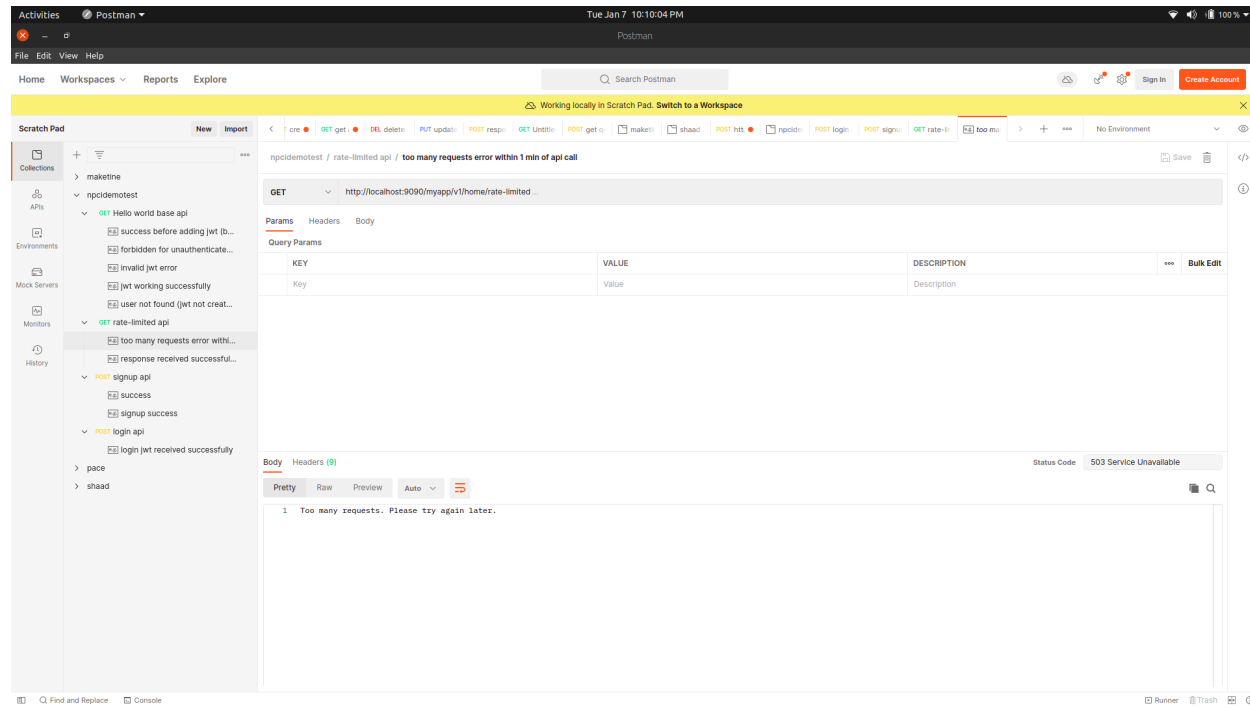


c.

5. Data api - generic HTTP GET with rate limited error

a. HTTP GET

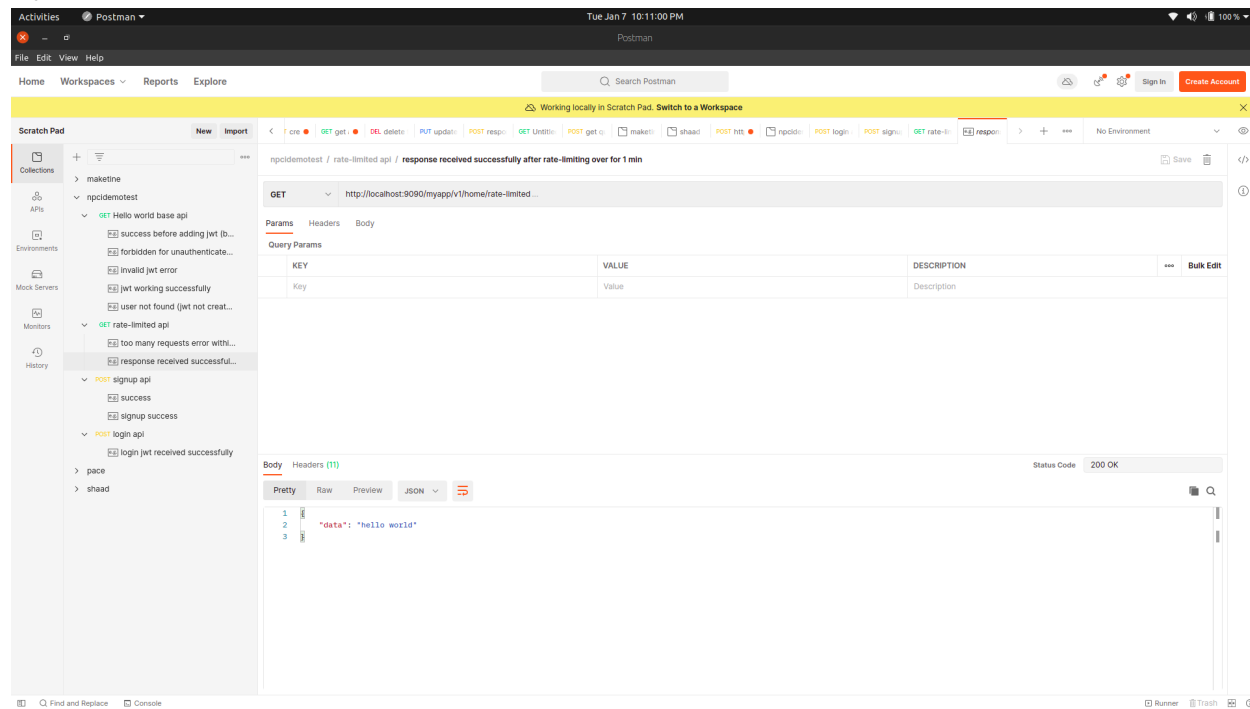
b. /myapp/v1/home/rate-limited



c.

6. Data api - rate limited api but response received within rate limiting threshold

a. /myapp/v1/home/rate-limited



b.