**Implementation**

HIPAA project has been implemented as a Spring Boot application with Java 8. The project has been deployed locally and tested via postman client. I prepared a JAR file to run a project standalone or deploy in cloud. I created a HIPAA mask as a single standalone microservie. More microservices can be added to the project without impacting functionality of existing microservice.

Kindly follow below steps to setup the project

**Prerequisites**

1. A Java Based IDE STS or IntelliJ (STS will have an inbuilt server to deploy apps. You would not need to have a seperate application server to run the app.)
2. Java/JDK 1.8 installed on the system
3. Maven 3.2+

**Steps**

1. Extract the zip file
2. Open your Java based IDE
3. Import the project as Spring Boot Project/Maven Project
4. Build it as Maven project or run as Spring Boot Application
5. If running locally, your endpoint would be `http://localhost:8080/rest/v1/json/mask`, pass your JSON as Request Body. Make sure it is a POST call. Hit the send button and you will get the masked JSON.

Alternate deploy options:

To execute standalone jar, open command prompt and execute following command.

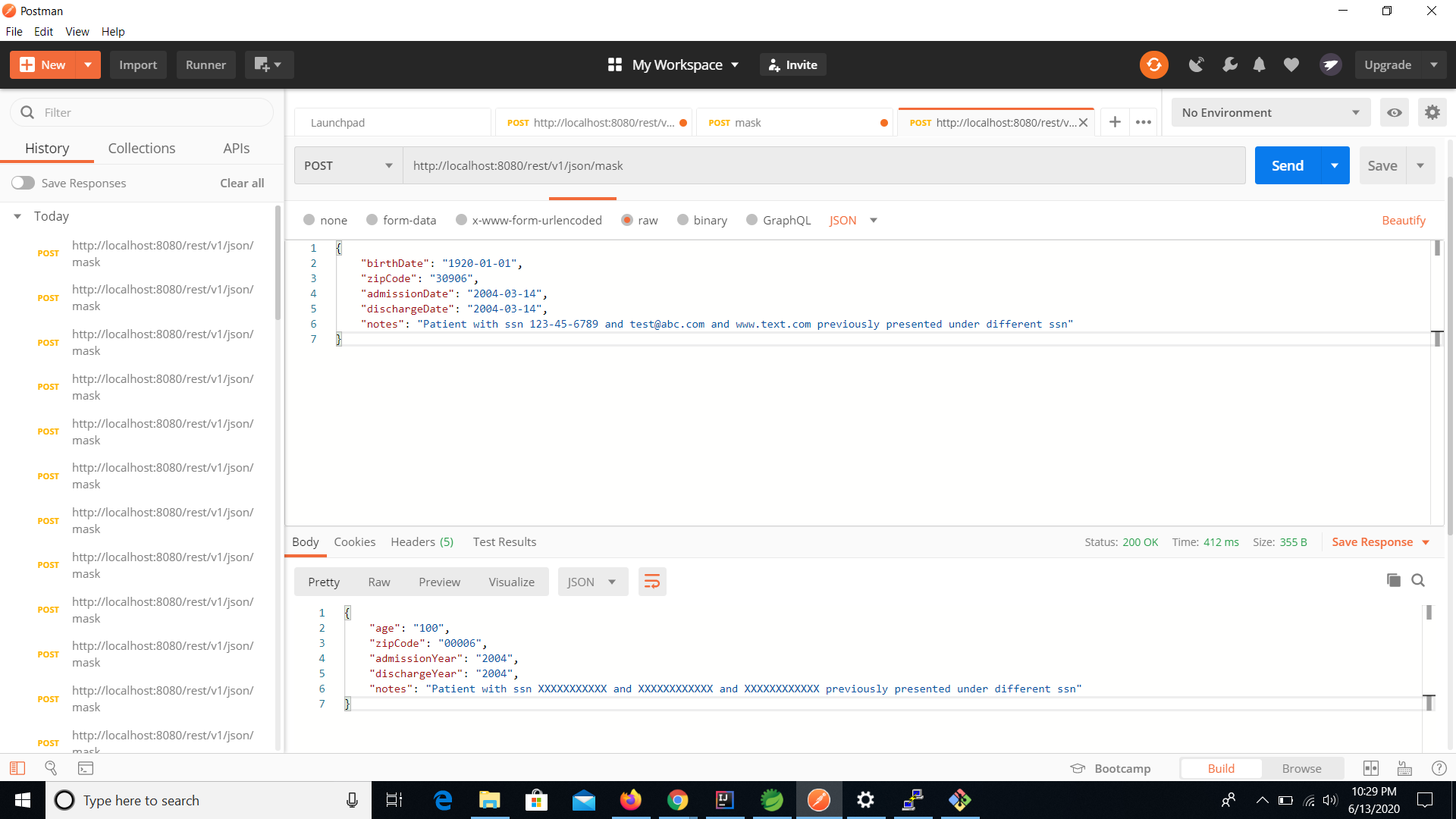
Java -jar jarfile name

Ex: java - jar path/to/your/jarfile.jar

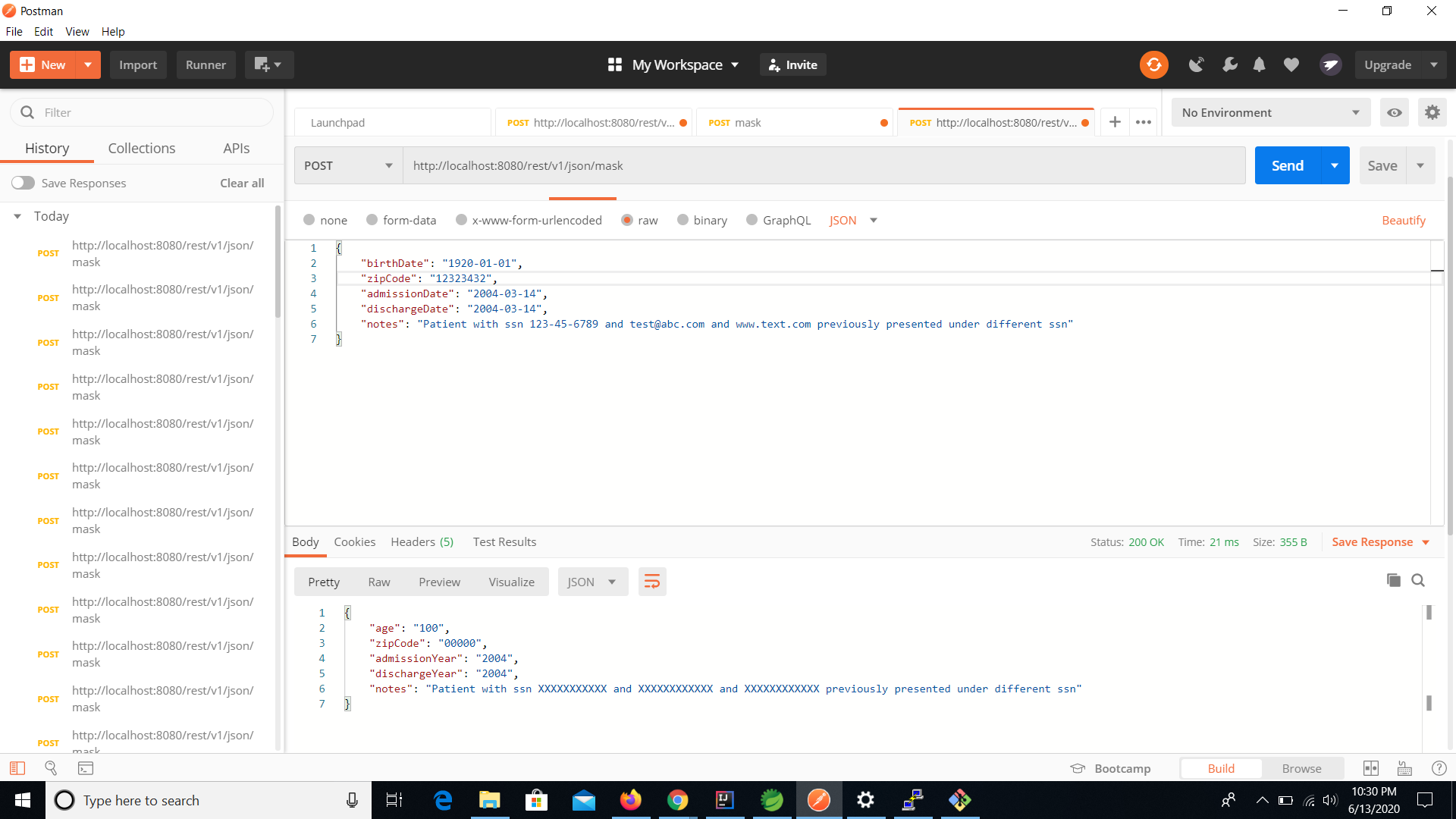
To deploy in the cloud, AWS Elasticbeanstalk can be used. Create a new application in elastic beanstalk home page and import the jar file into AWS console. RDS, VPC configurations can be left blank since we are testing without any database layer and network requirements. Applications will be deployed in AWS cloud and the end point can be accessed via Postman client.

**Test cases**

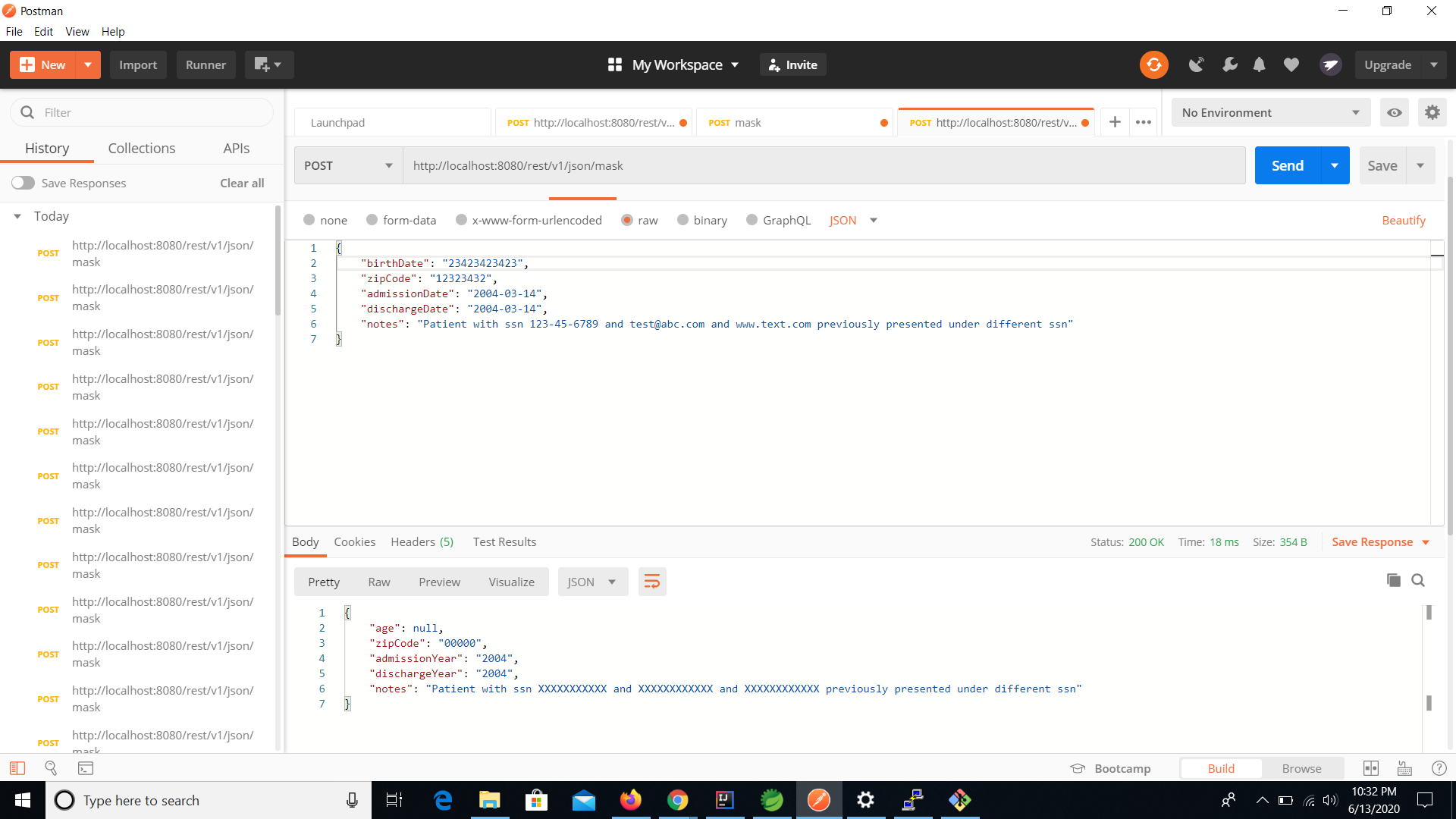
1. When entering a zip code with >20000 population, the first 3 digits are masked as shown in the following postman client call.



1. When entering unformatted input, I am returning 00000 as result. I am masking any email ids, ssn field, phone numbers in notes field.



1. When there is a non formatted date, I am returning null as result.



**Future Improvements**

1. AWS CI/CD pipeline implementation for automating deployments and migration
2. Circuit breaker pattern to handle failures
3. Sonarqube validation framework