Task planner

**Requirements:**

Write a program to create, update, assign, search and get update history of all tasks using Java Spring Boot, MongoDB and RestAPI

**Spring Boot:**

Spring Boot is an open-source popular Java-based framework for building web and enterprise applications maintained by a company called Pivotal. Spring boot reduces lots of development time and increases productivity. It has many advantages & with its help, it is easier to avoid writing lots of boilerplate Code, Annotations, and XML Configuration. It also has a built-in configuration for Spring JDBC, Spring ORM, Spring Data, Spring Security, etc.

**MongoDB:**

MongoDB is a document-oriented NoSQL database that stores JSON-like documents with dynamic schemas. It is commonly used for high-volume data storage.

Few advantages of a NoSQL database over a SQL database are listed below;

Schemaless

No complex joins.

Ease of scale-out − NoSQL databases are easy to scale.

Conversion/mapping of application objects to database objects not needed.

Uses internal memory for storing the (windowed) working set, enabling faster access to data.

**Tools used in this project**

Java 8

Spring Boot

Java

MongoDB

Postman

TaskPlannerApplication

**package** com.task;

**import** org.springframework.boot.SpringApplication;

**import** org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication

**public** **class** TaskPlannerApplication {

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(TaskPlannerApplication.**class**, args);

}

}

ServletInitializer

**package** com.task;

**import** org.springframework.boot.builder.SpringApplicationBuilder;

**import** org.springframework.boot.web.servlet.support.SpringBootServletInitializer;

**public** **class** ServletInitializer **extends** SpringBootServletInitializer {

@Override

**protected** SpringApplicationBuilder configure(SpringApplicationBuilder application) {

**return** application.sources(TaskPlannerApplication.**class**);

}

}

Task

**package** com.task.model;

**import** java.util.Date;

**import** org.springframework.data.annotation.Id;

**import** org.springframework.data.mongodb.core.mapping.Document;

**import** lombok.Data;

@Data

@Document

**public** **class** Task {

@Id

**private** String taskId;

**private** String title;

**private** String description;

**private** String createdBy;

**private** String assignedTo;

**private** Date completedOn;

**private** String status;

}

TaskController

**package** com.task.controller;

**import** java.util.List;

**import** java.util.Optional;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.web.bind.annotation.GetMapping;

**import** org.springframework.web.bind.annotation.PathVariable;

**import** org.springframework.web.bind.annotation.PostMapping;

**import** org.springframework.web.bind.annotation.PutMapping;

**import** org.springframework.web.bind.annotation.RequestBody;

**import** org.springframework.web.bind.annotation.RestController;

**import** com.task.dao.TaskDao;

**import** com.task.model.Task;

@RestController

**public** **class** TaskController {

@Autowired

TaskDao dao;

@PostMapping("/insert")

**public** Task insert(@RequestBody Task t) {

**return** dao.insert(t);

}

@PostMapping("/insertall")

**public** List<Task> insertall(@RequestBody List<Task> t){

**return** dao.insertall(t);

}

@GetMapping("/getall")

**public** List<Task> gettall(){

**return** dao.getall();

}

@PutMapping("/update")

**public** Task update(@RequestBody Task t) {

**return** dao.update(t);

}

@GetMapping("/getbyId{t}")

**public** Optional<Task> findBytId(@PathVariable String t) {

**return** dao.findById(t); }

@GetMapping("/getbytitle{t}")

**public** Task findBytitle(@PathVariable String t) {

**return** dao.findBytitle(t); }

}

TaskDao

**package** com.task.dao;

**import** java.util.List;

**import** java.util.Optional;

**import** org.springframework.beans.factory.annotation.Autowired;

**import** org.springframework.stereotype.Service;

**import** com.task.model.Task;

**import** com.task.repository.TaskRepository;

@Service

**public** **class** TaskDao {

@Autowired

TaskRepository repo;

**public** Task insert(Task t) {

**return** repo.save(t);

}

**public** List<Task> insertall(List<Task> t) {

**return** repo.saveAll(t);

}

**public** List<Task> getall() {

**return** repo.findAll();

}

**public** Task update(Task t) {

Task tt = repo.findById(t.getTaskId()).orElse(**null**);

tt.setTitle(t.getTitle());

**return** repo.save(t);

}

**public** Optional<Task> findById(String t) {

**return** repo.findById(t);

}

**public** Task findBytitle(String t) {

**return** repo.findBytitle(t);

}

}

TaskPlannerApplicationTests

**package** com.task;

**import** org.junit.jupiter.api.Test;

**import** org.springframework.boot.test.context.SpringBootTest;

@SpringBootTest

**class** TaskPlannerApplicationTests {

@Test

**void** contextLoads() {

}

}

application.properties

spring.data.mongodb.uri=mongodb://localhost:27017/TaskDB

TaskPlanner/pom.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>2.7.3</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<groupId>com.task</groupId>

<artifactId>TaskPlanner</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>war</packaging>

<name>TaskPlanner</name>

<description>Task Planner Spring Boot Application Backend</description>

<properties>

<java.version>1.8</java.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-mongodb</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<optional>true</optional>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-tomcat</artifactId>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-maven-plugin</artifactId>

<configuration>

<excludes>

<exclude>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

</exclude>

</excludes>

</configuration>

</plugin>

</plugins>

</build>

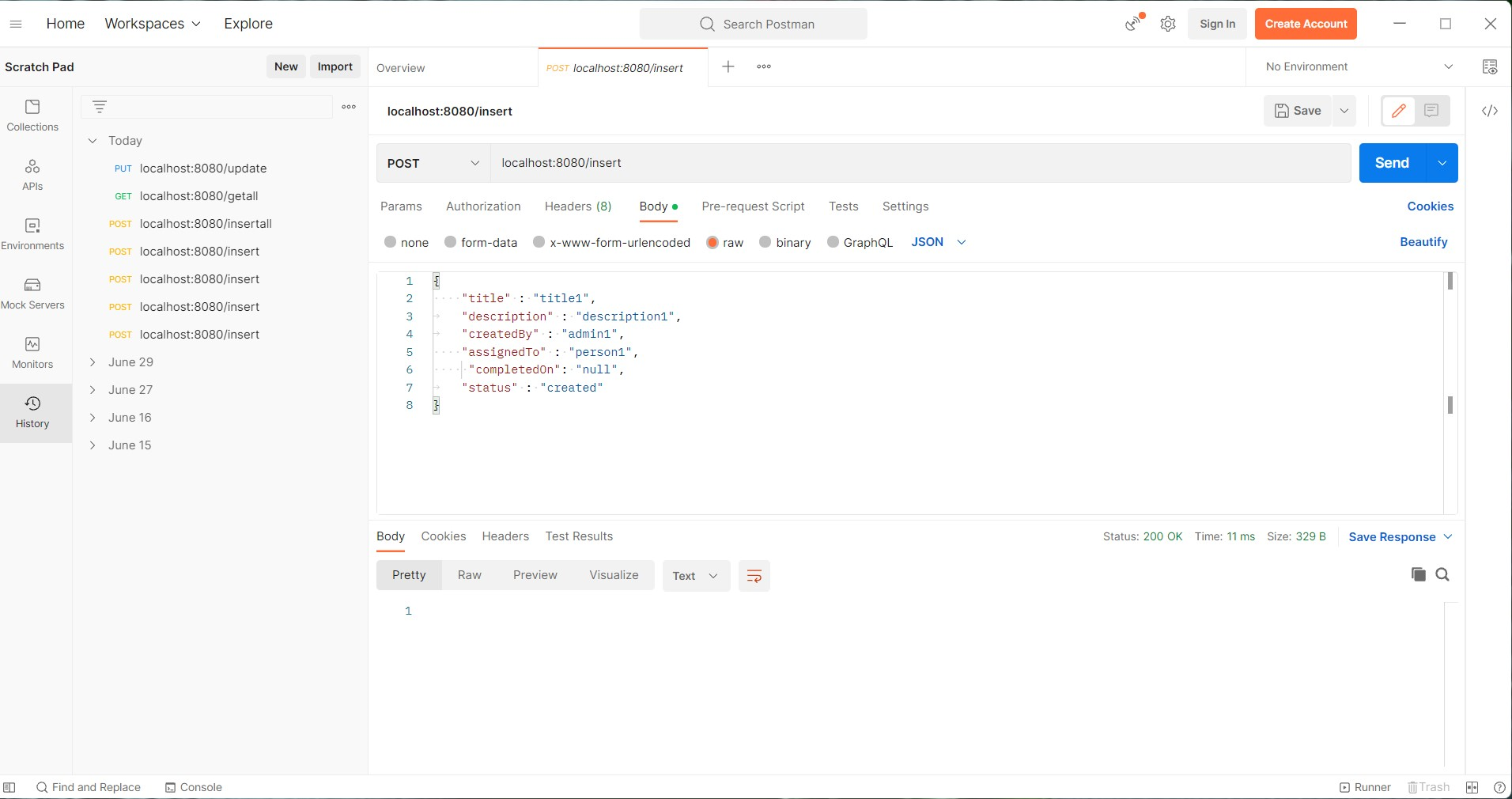
</project>

**Running the Project and Testing with Postman**

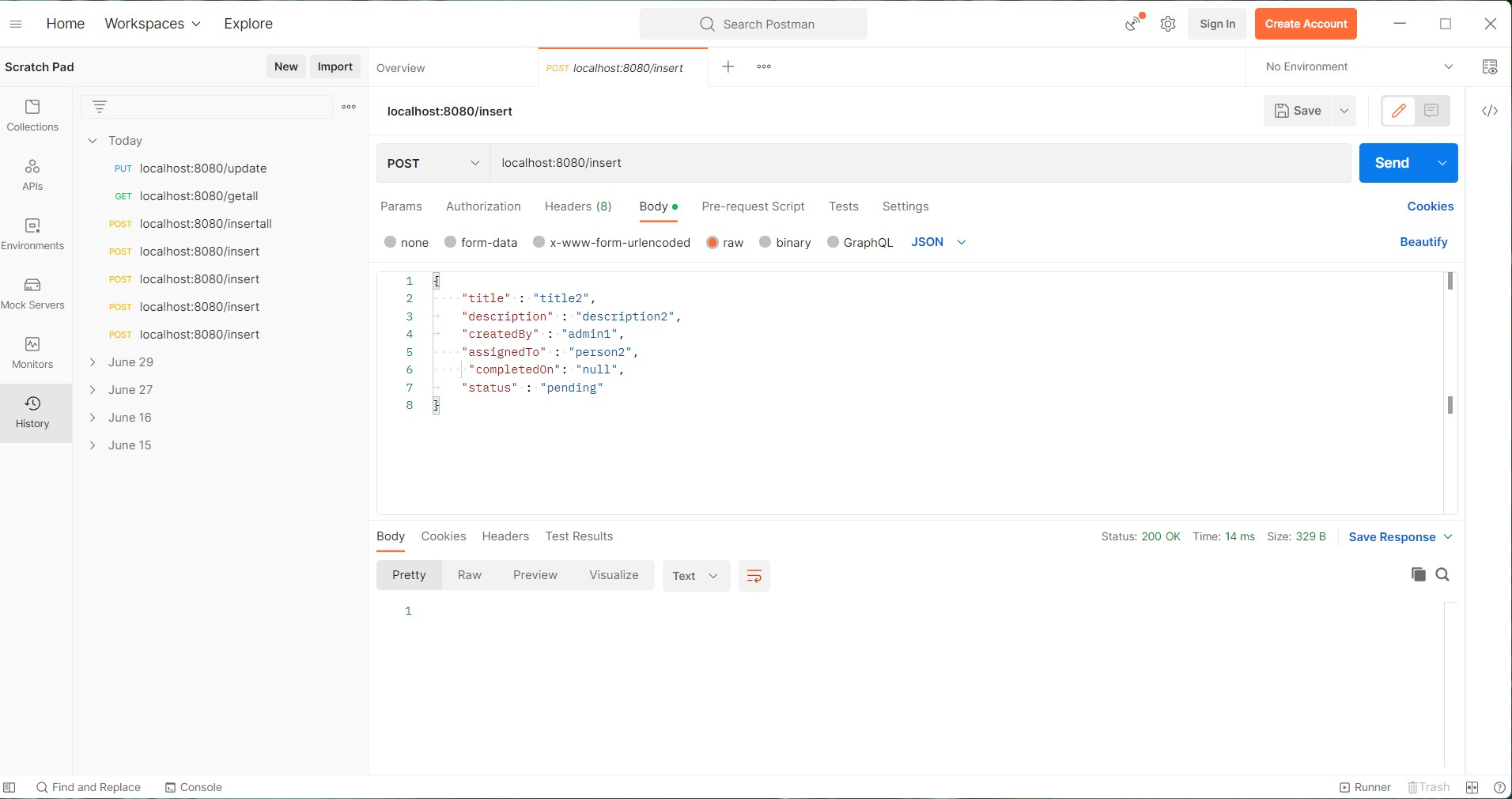
Now let’s test our API calls using postman. Right-click on the project and select Run as a Spring boot App.

**OUTPUT SCREENSHOTS FOR TASK PLANNER**

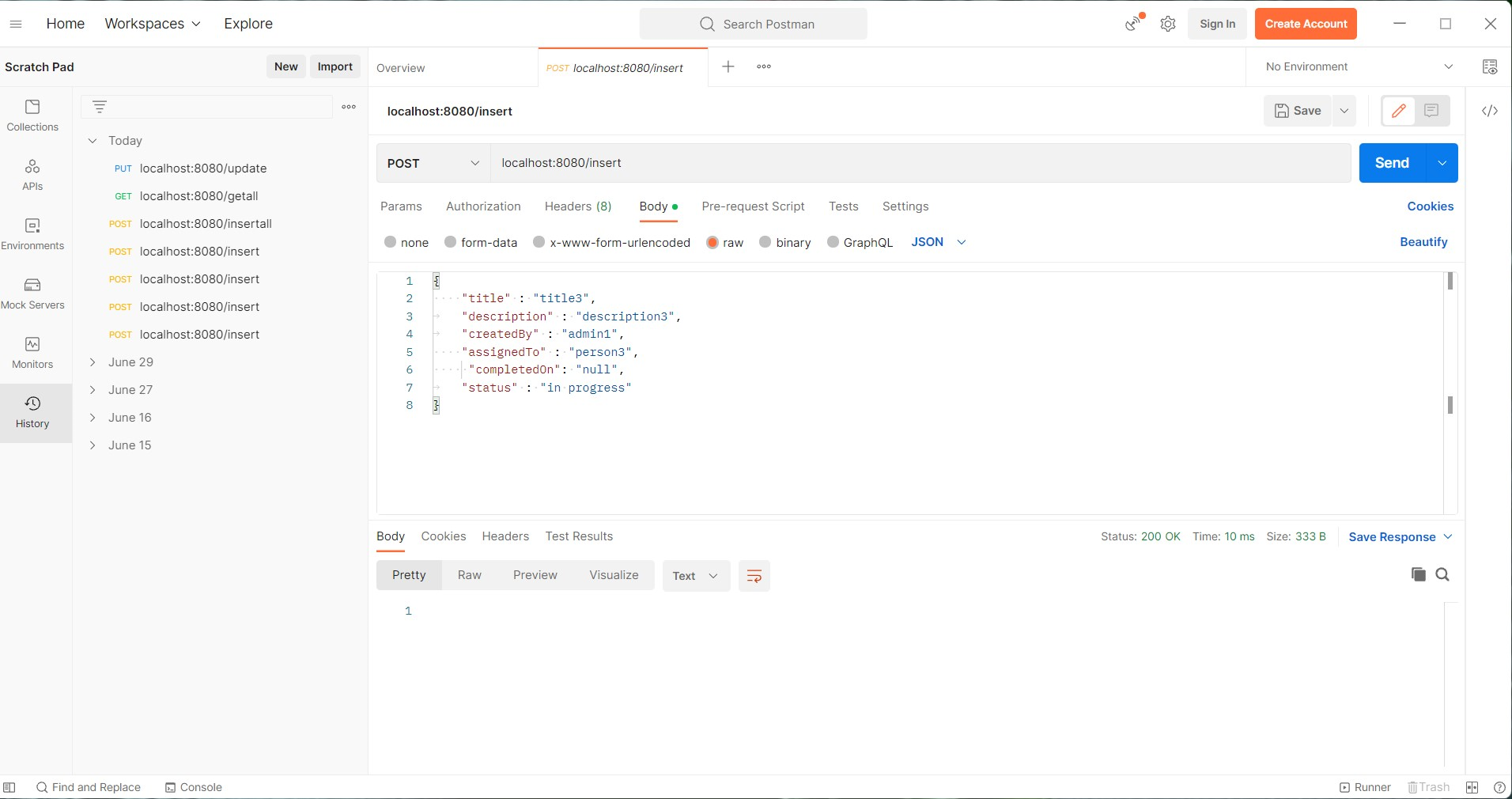
Created Task (task/create) with basic task details like title, description, createdBy with status created



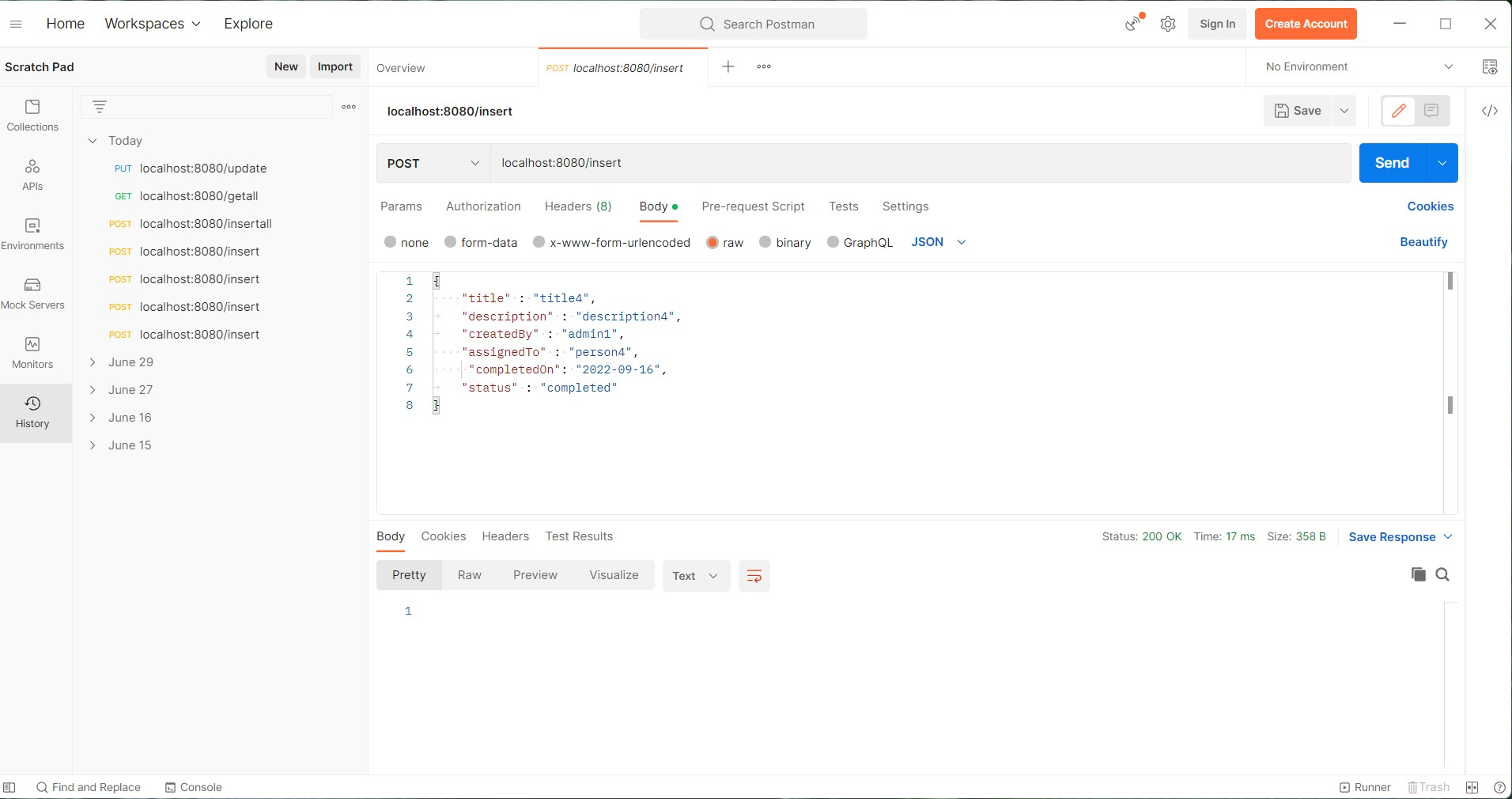
Created Task (task/create) with basic task details like title, description, createdBy with status pending



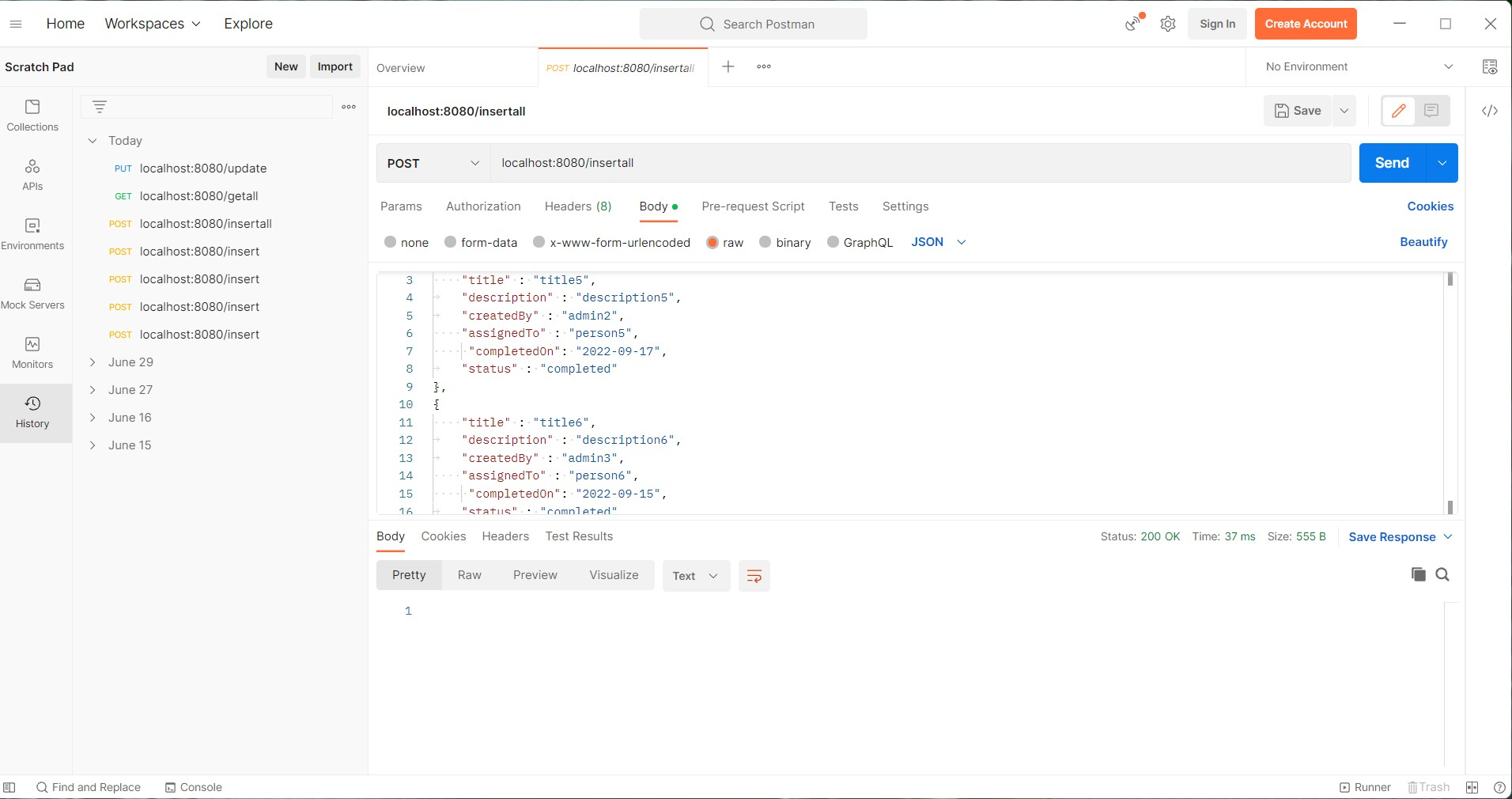
Created Task (task/create) with basic task details like title, description, createdBy with status in progress



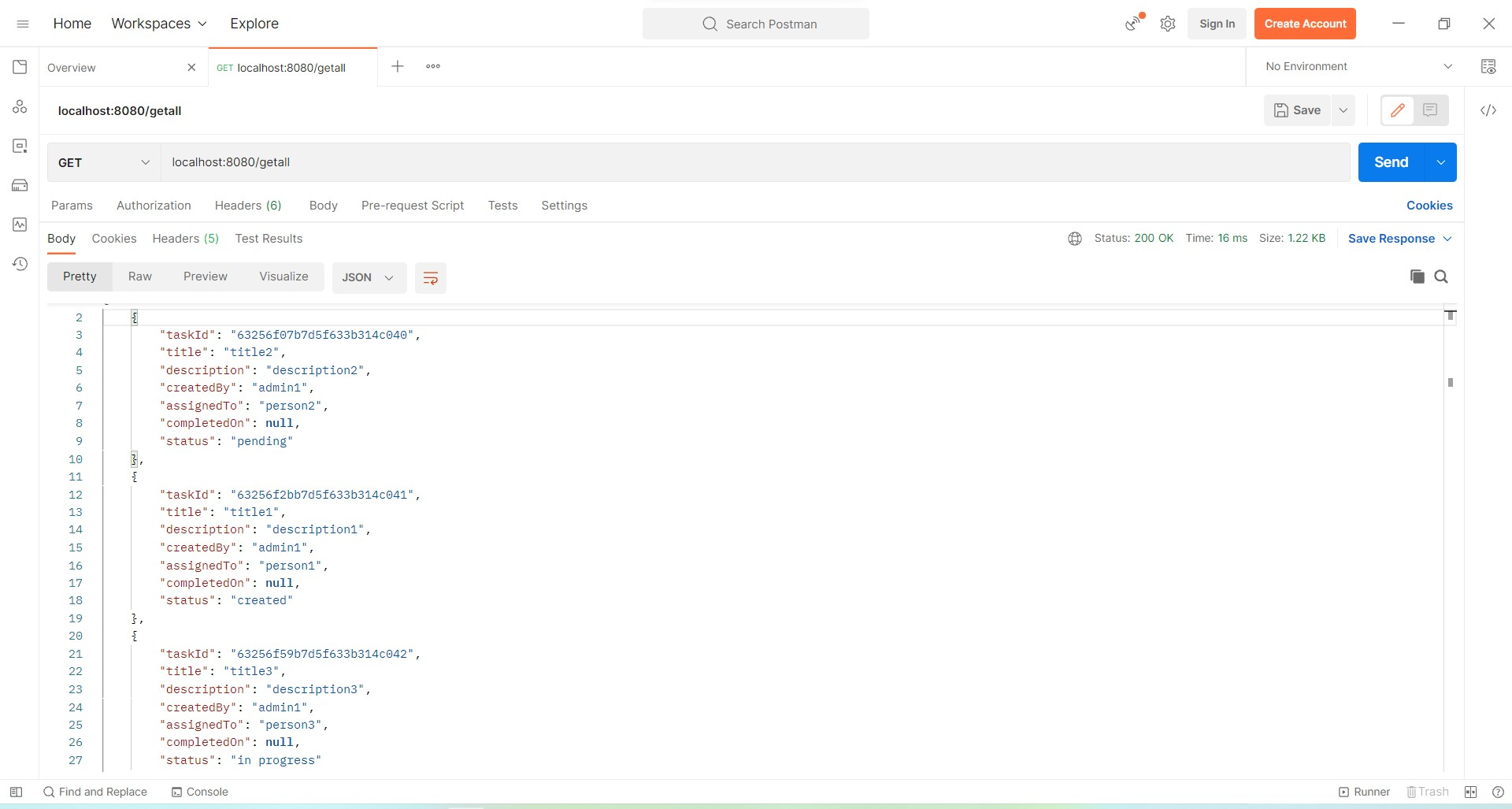
Created Task (task/create) with basic task details like title, description, createdBy with status completed



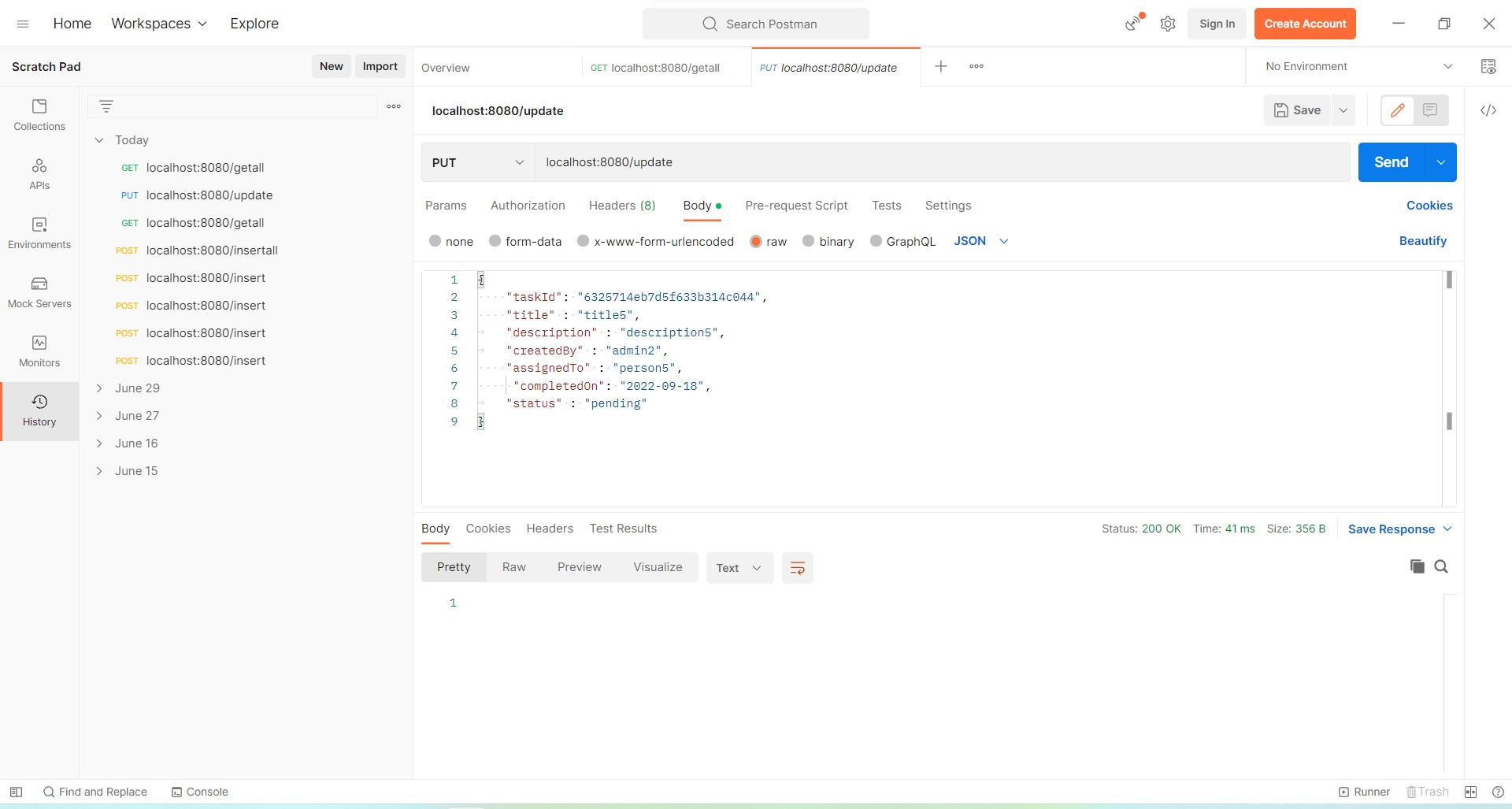
Assign Tasks (task/assign) with basic task details like title, description, createdBy, status and assigning to particular person



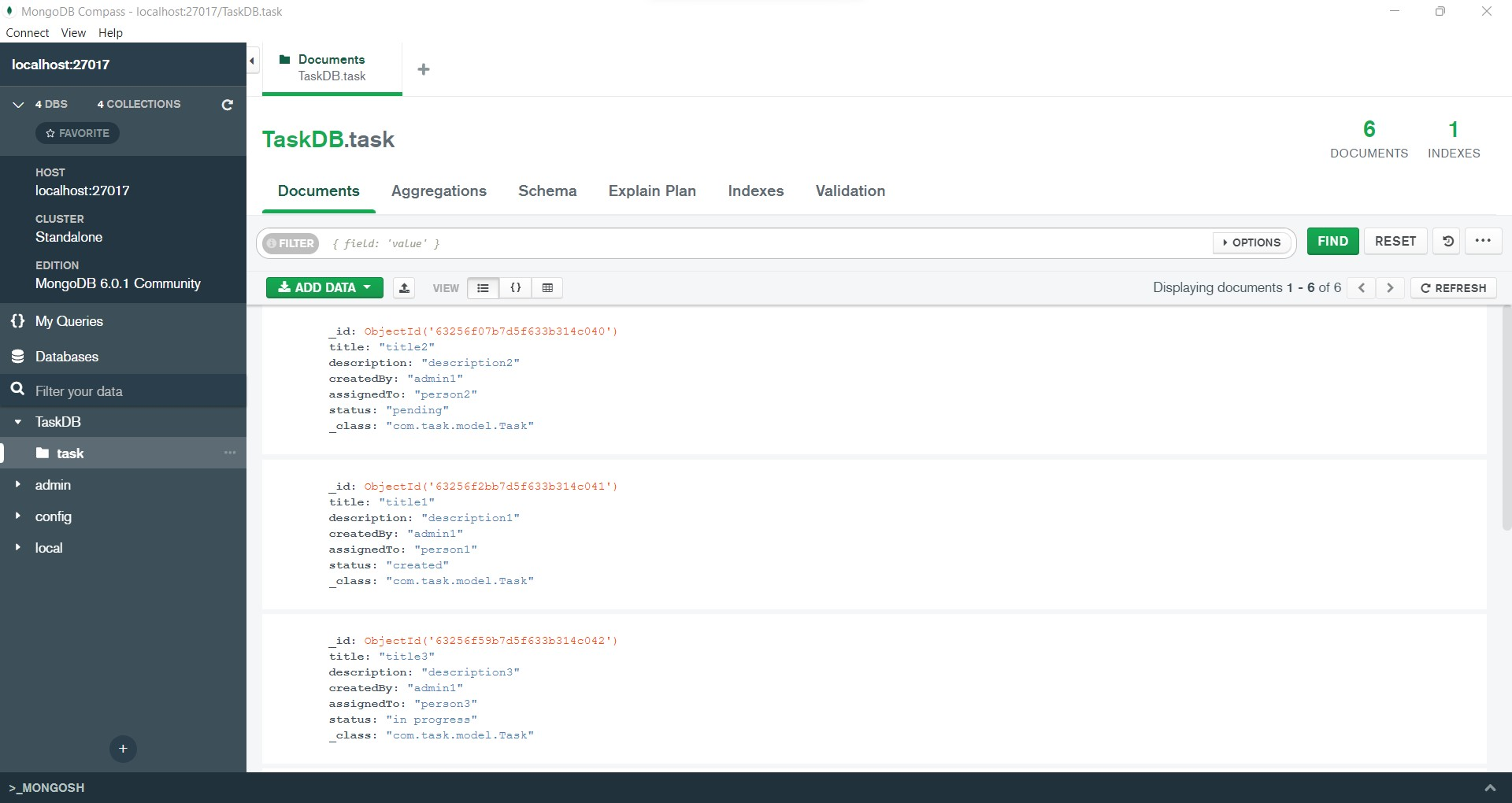
Get history of updates(task/history) given id of task fetch all the update history



Update Task(task/update) update basic task details like title, description, status



Output Screenshot from MongoDB database



**Conclusion:**

Therefore, we created a simple REST API using Java Spring Boot and MongoDB to save objects and fetch them from a database, all without writing a concrete repository implementation.

**Github link for Taskplanner Project:**

https://github.com/mailtorohini/Task-Planner.git