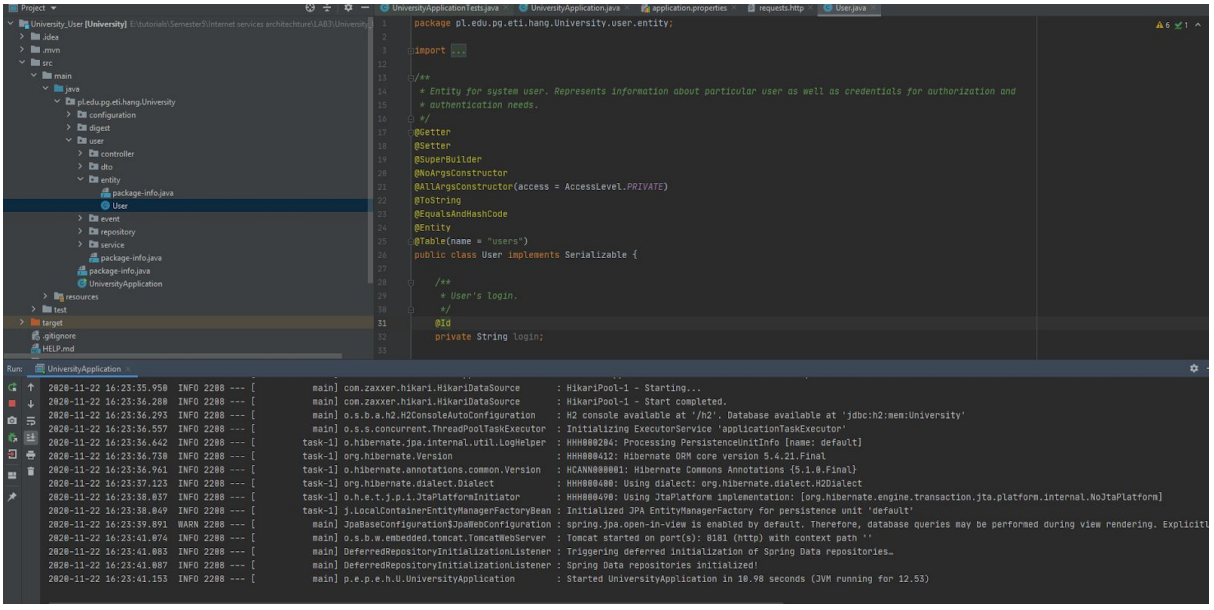
Internet Architecture Service - LAB3

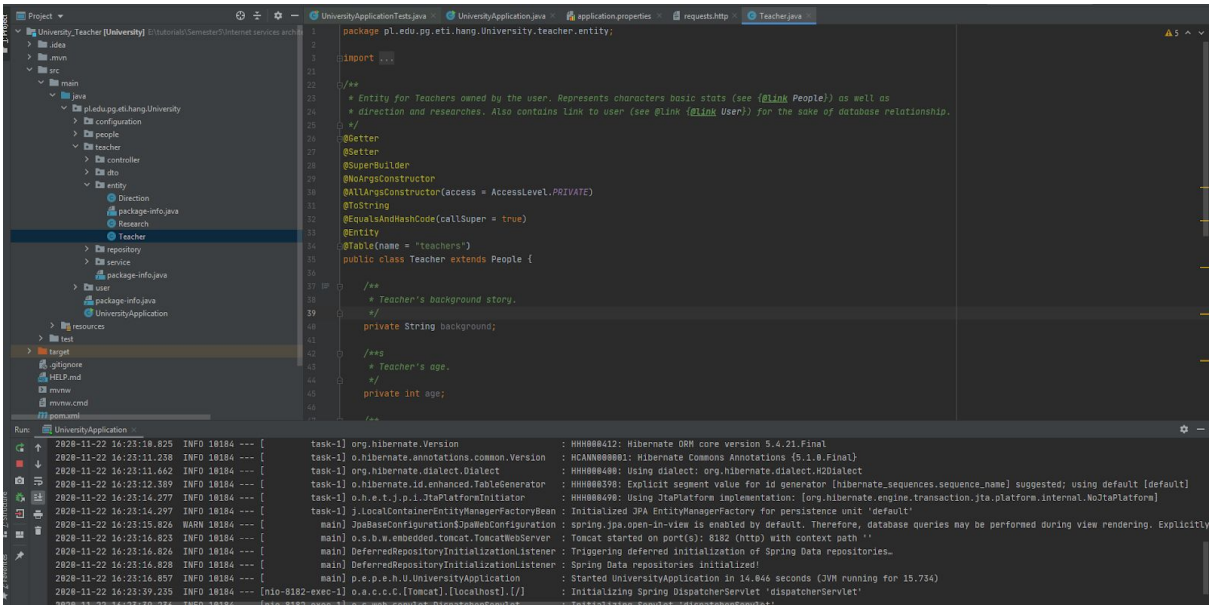
Hang Liu

1. Original Spring Boot project should be divided into two separate projects representing two stand-alone applications. One for category management and second for elements management. Each of the applications should make an use of private in-memory h2 database. Category management application should only contain mechanisms for category management. Elements management application should contain mechanisms for elements management as well as simplified mechanism for category management (in order to maintain relationships and hierarchy). (3 point)

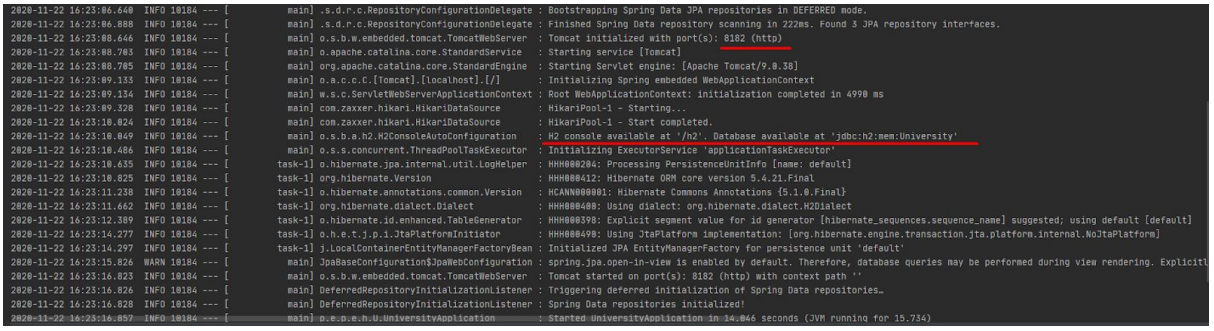
2 standalone applications:
Category management University_User:



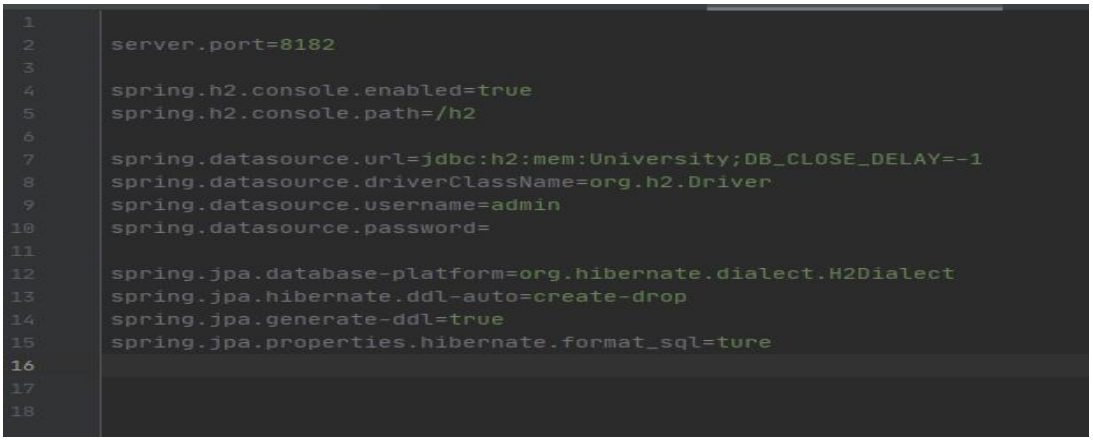
Elements management University_Teacher:



They all have their own in-memory H2 database, as we can see from the log informations:
University_Teacher: This application uses the port 8182 during the test.
Log infos:



Configurations:



```
server.port=8182

spring.h2.console.enabled=true
spring.h2.console.path=/h2

spring.datasource.url=jdbc:h2:mem:University;DB_CLOSE_DELAY=-1
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=admin
spring.datasource.password=

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
spring.jpa.hibernate.ddl-auto=create-drop
spring.jpa.generate-ddl=true
spring.jpa.properties.hibernate.format_sql=ture
```

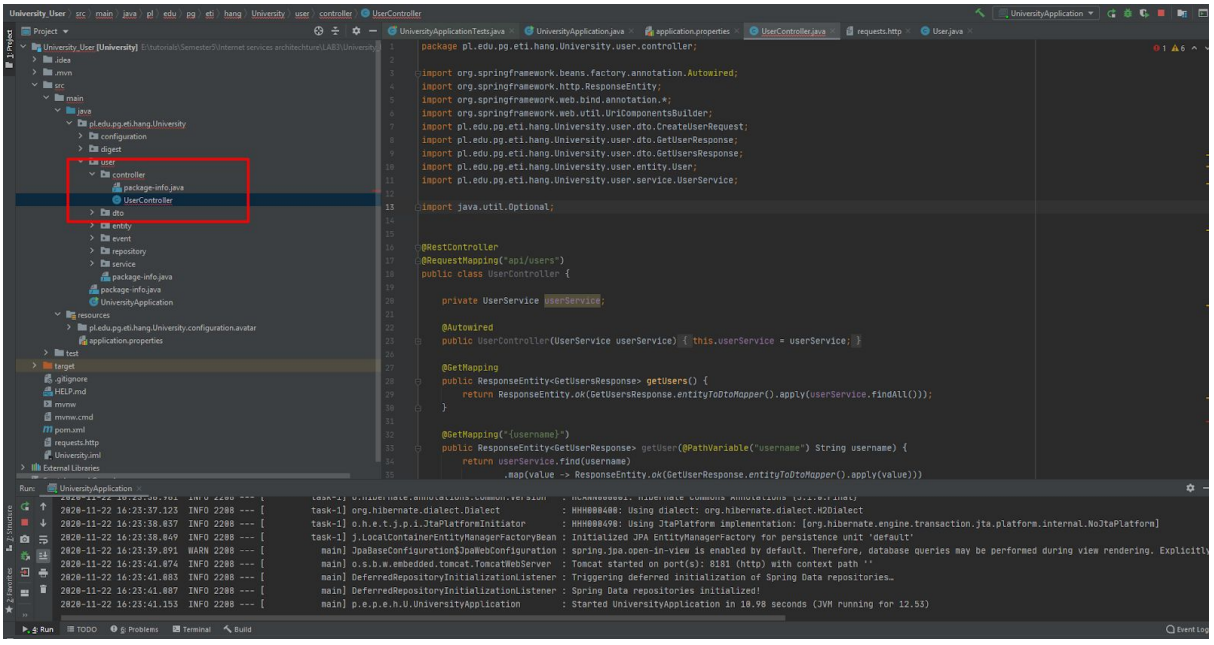
University_User:This application uses the port 8181 during the test.
Log info:

```
  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::        (v2.3.4.RELEASE)

2020-11-22 16:23:31.661  INFO 2288 --- [           main] p.e.p.e.h.U.UniversityApplication        : Starting UniversityApplication on DESKTOP-KVEO3TV with PID 2288 (E:\tutorials\Semester5\Internet services architechture\LAB3\
2020-11-22 16:23:31.673  INFO 2288 --- [           main] p.e.p.e.h.U.UniversityApplication        : No active profile set, falling back to default profiles: default
2020-11-22 16:23:33.953  INFO 2288 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFERRED mode.
2020-11-22 16:23:34.178  INFO 2288 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 203ms. Found 1 JPA repository interfaces.
2020-11-22 16:23:35.640  INFO 2288 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8181 (http)
2020-11-22 16:23:35.664  INFO 2288 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2020-11-22 16:23:35.664  INFO 2288 --- [           main] org.apache.catalina.core.StandardEngine  : Starting Servlet engine: [Apache Tomcat/9.0.38]
2020-11-22 16:23:35.859  INFO 2288 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
2020-11-22 16:23:35.859  INFO 2288 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 4008 ms
2020-11-22 16:23:35.950  INFO 2288 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
2020-11-22 16:23:36.280  INFO 2288 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
2020-11-22 16:23:36.293  INFO 2288 --- [           main] o.s.b.a.h2.H2ConsoleAutoConfiguration    : H2 console available at '/h2'. Database available at 'jdbc:h2:mem:University'
2020-11-22 16:23:36.557  INFO 2288 --- [           main] o.s.s.concurrent.ThreadPoolTaskExecutor  : Initializing ExecutorService 'applicationTaskExecutor'
2020-11-22 16:23:36.642  INFO 2288 --- [         task-1] o.hibernate.jpa.internal.util.LogHelper  : HHH000204: Processing PersistenceUnitInfo [name: default]
2020-11-22 16:23:36.730  INFO 2288 --- [         task-1] org.hibernate.Version                    : HHH000412: Hibernate ORM core version 5.4.21.Final
2020-11-22 16:23:36.961  INFO 2288 --- [         task-1] o.hibernate.annotations.common.Version   : HCANN000001: Hibernate Commons Annotations {5.1.0.Final}
2020-11-22 16:23:37.123  INFO 2288 --- [         task-1] org.hibernate.dialect.Dialect            : HHH000400: Using dialect: org.hibernate.dialect.H2Dialect
2020-11-22 16:23:38.037  INFO 2288 --- [         task-1] o.h.e.t.j.p.i.JtaPlatformInitiator       : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
2020-11-22 16:23:38.049  INFO 2288 --- [         task-1] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2020-11-22 16:23:39.891  WARN 2288 --- [           main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed during view rendering. Explicitly
2020-11-22 16:23:41.074  INFO 2288 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8181 (http) with context path ''
2020-11-22 16:23:41.883  INFO 2288 --- [           main] DeferredRepositoryInitializationListener : Triggering deferred initialization of Spring Data repositories…
2020-11-22 16:23:41.887  INFO 2288 --- [           main] DeferredRepositoryInitializationListener : Spring Data repositories initialized!
2020-11-22 16:23:41.153  INFO 2288 --- [           main] p.e.p.e.h.U.UniversityApplication        : Started UniversityApplication in 10.98 seconds (JVM running for 12.53)
```

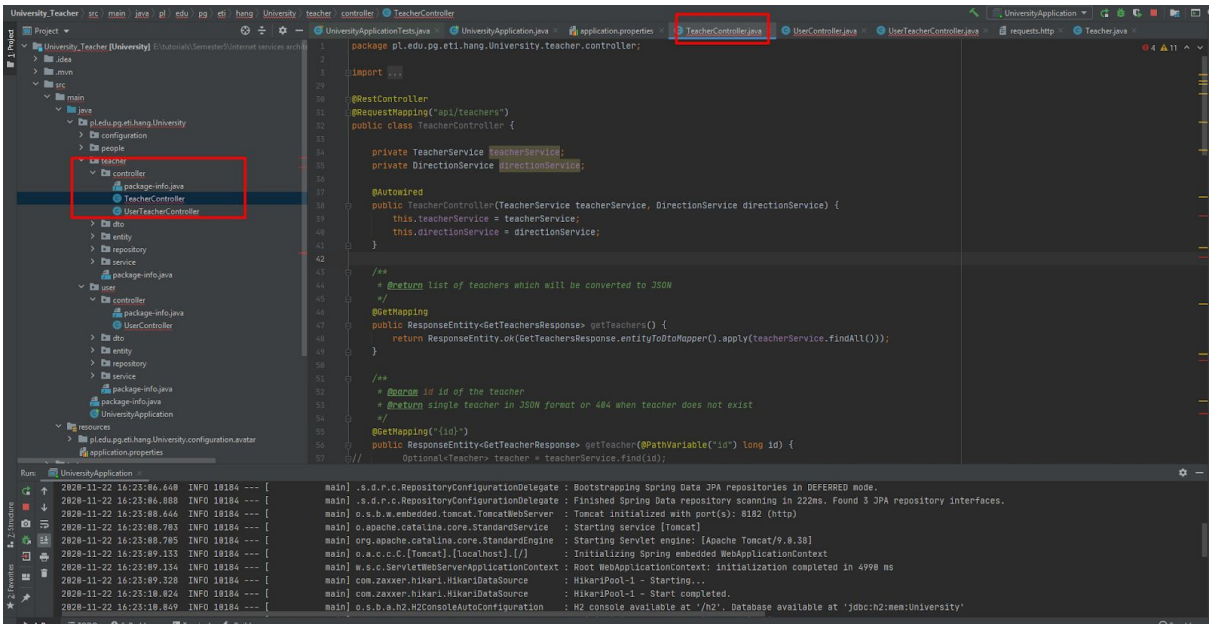
Configurations:

```
server.port=8181
University.teachers.url=http://localhost:8182/api/

spring.h2.console.enabled=true
spring.h2.console.path=/h2

spring.datasource.url=jdbc:h2:mem:University;DB_CLOSE_DELAY=-1
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=admin
spring.datasource.password=

spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
spring.jpa.hibernate.ddl-auto=create-drop
spring.jpa.generate-ddl=true
spring.jpa.properties.hibernate.format_sql=ture
```

And we abide by the rule: category management application should only contain mechanisms for category management.
University_User:

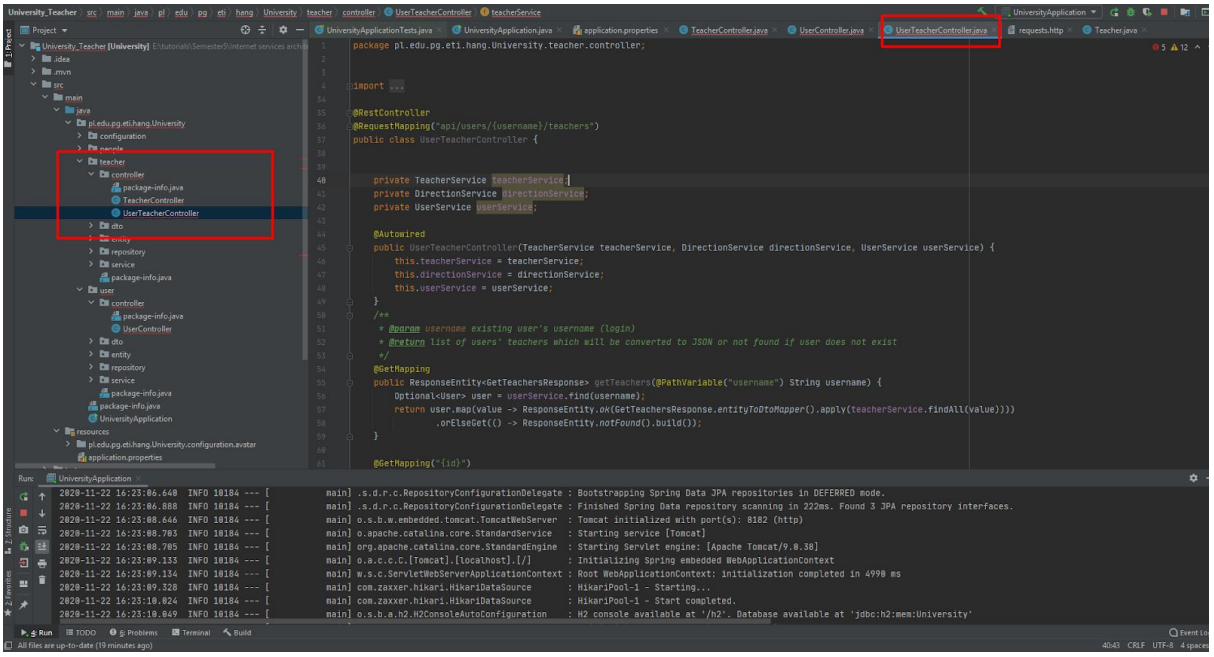


Elements management application should contain mechanisms for elements management as well as simplified mechanisms for category management.
University_Teacher:



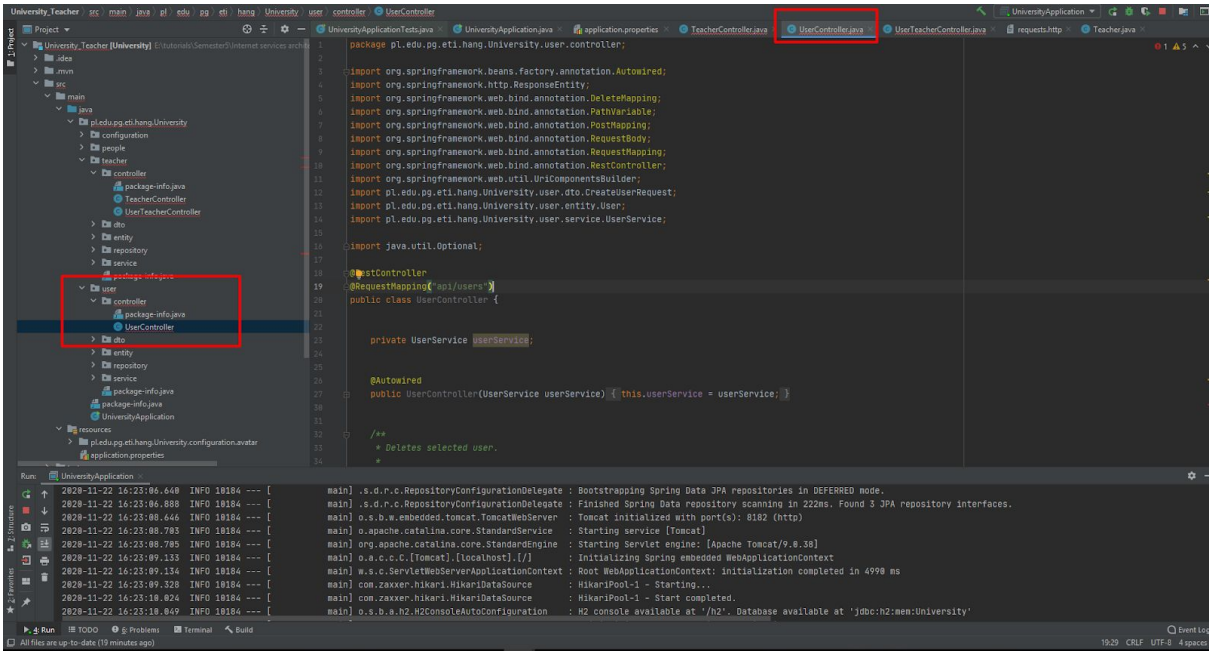
```
package pl.edu.pg.eti.hang.University.teacher.controller;

import ...

@RestController
@RequestMapping("api/teachers")
public class TeacherController {

    private TeacherService teacherService;
    private DirectionService directionService;

    @Autowired
    public TeacherController(TeacherService teacherService, DirectionService directionService) {
        this.teacherService = teacherService;
        this.directionService = directionService;
    }

    /**
     * @return list of teachers which will be converted to JSON
     */
    @GetMapping
    public ResponseEntity<GetTeachersResponse> getTeachers() {
        return ResponseEntity.ok(GetTeachersResponse.entityToDtoMapper().apply(teacherService.findAll()));
    }

    /**
     * @param id id of the teacher
     * @return single teacher in JSON format or 404 when teacher does not exist
     */
    @GetMapping("{id}")
    public ResponseEntity<GetTeacherResponse> getTeacher(@PathVariable("id") long id) {
//        Optional<Teacher> teacher = teacherService.find(id);
```

```
2020-11-22 16:23:06.640  INFO 10184 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFERRED mode.
2020-11-22 16:23:06.888  INFO 10184 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 222ms. Found 3 JPA repository interfaces.
2020-11-22 16:23:08.646  INFO 10184 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8182 (http)
2020-11-22 16:23:08.703  INFO 10184 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2020-11-22 16:23:08.705  INFO 10184 --- [           main] org.apache.catalina.core.StandardEngine  : Starting Servlet engine: [Apache Tomcat/9.0.38]
2020-11-22 16:23:09.133  INFO 10184 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
2020-11-22 16:23:09.134  INFO 10184 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 4998 ms
2020-11-22 16:23:09.328  INFO 10184 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
2020-11-22 16:23:10.024  INFO 10184 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
2020-11-22 16:23:10.049  INFO 10184 --- [           main] o.s.b.a.h2.H2ConsoleAutoConfiguration    : H2 console available at '/h2'. Database available at 'jdbc:h2:mem:University'
```



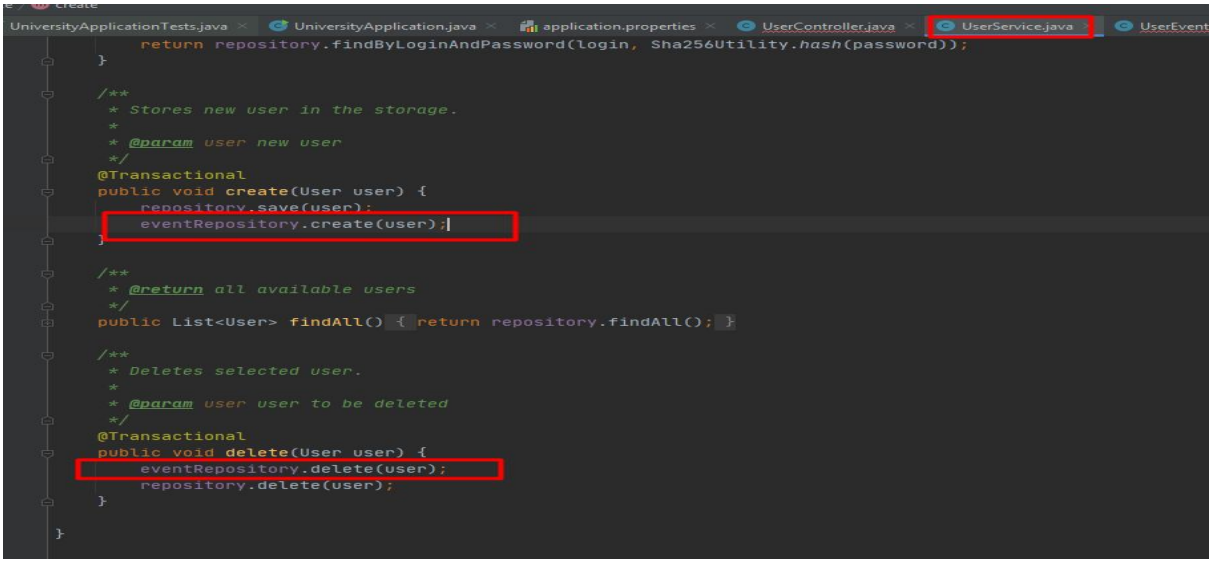
```
package pl.edu.pg.eti.hang.University.teacher.controller;

import ...

@RestController
@RequestMapping("api/users/{username}/teachers")
public class UserTeacherController {

    private TeacherService teacherService;
    private DirectionService directionService;
    private UserService userService;

    @Autowired
    public UserTeacherController(TeacherService teacherService, DirectionService directionService, UserService userService) {
        this.teacherService = teacherService;
        this.directionService = directionService;
        this.userService = userService;
    }
    /**
     * @param username existing user's username (login)
     * @return list of users' teachers which will be converted to JSON or not found if user does not exist
     */
    @GetMapping
    public ResponseEntity<GetTeachersResponse> getTeachers(@PathVariable("username") String username) {
        Optional<User> user = userService.find(username);
        return user.map(value -> ResponseEntity.ok(GetTeachersResponse.entityToDtoMapper().apply(teacherService.findAll(value))))
                .orElseGet(() -> ResponseEntity.notFound().build());
    }

    @GetMapping("{id}")
```

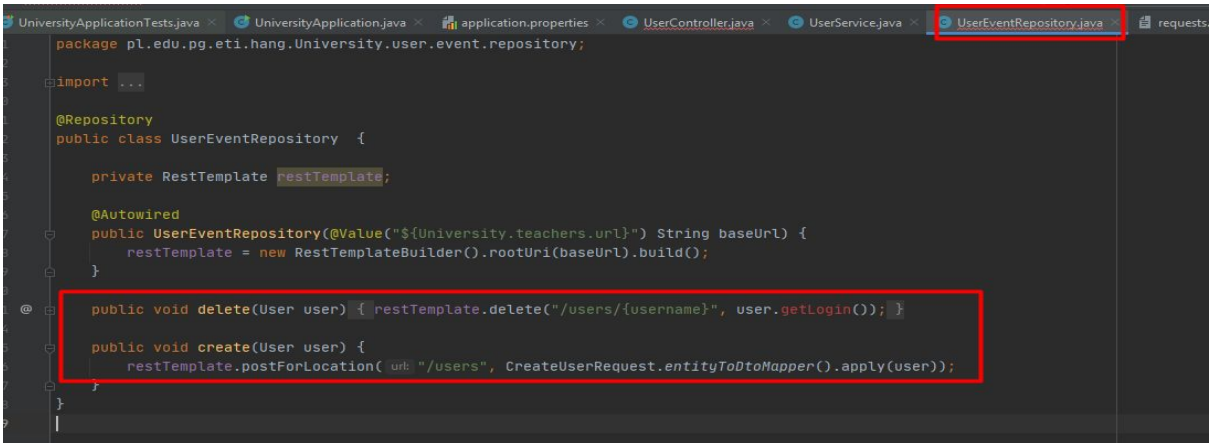
```
2020-11-22 16:23:06.640  INFO 10184 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFERRED mode.
2020-11-22 16:23:06.888  INFO 10184 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 222ms. Found 3 JPA repository interfaces.
2020-11-22 16:23:08.646  INFO 10184 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8182 (http)
2020-11-22 16:23:08.703  INFO 10184 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2020-11-22 16:23:08.705  INFO 10184 --- [           main] org.apache.catalina.core.StandardEngine  : Starting Servlet engine: [Apache Tomcat/9.0.38]
2020-11-22 16:23:09.133  INFO 10184 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
2020-11-22 16:23:09.134  INFO 10184 --- [           main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 4998 ms
2020-11-22 16:23:09.328  INFO 10184 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Starting...
2020-11-22 16:23:10.024  INFO 10184 --- [           main] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Start completed.
2020-11-22 16:23:10.049  INFO 10184 --- [           main] o.s.b.a.h2.H2ConsoleAutoConfiguration    : H2 console available at '/h2'. Database available at 'jdbc:h2:mem:University'
```



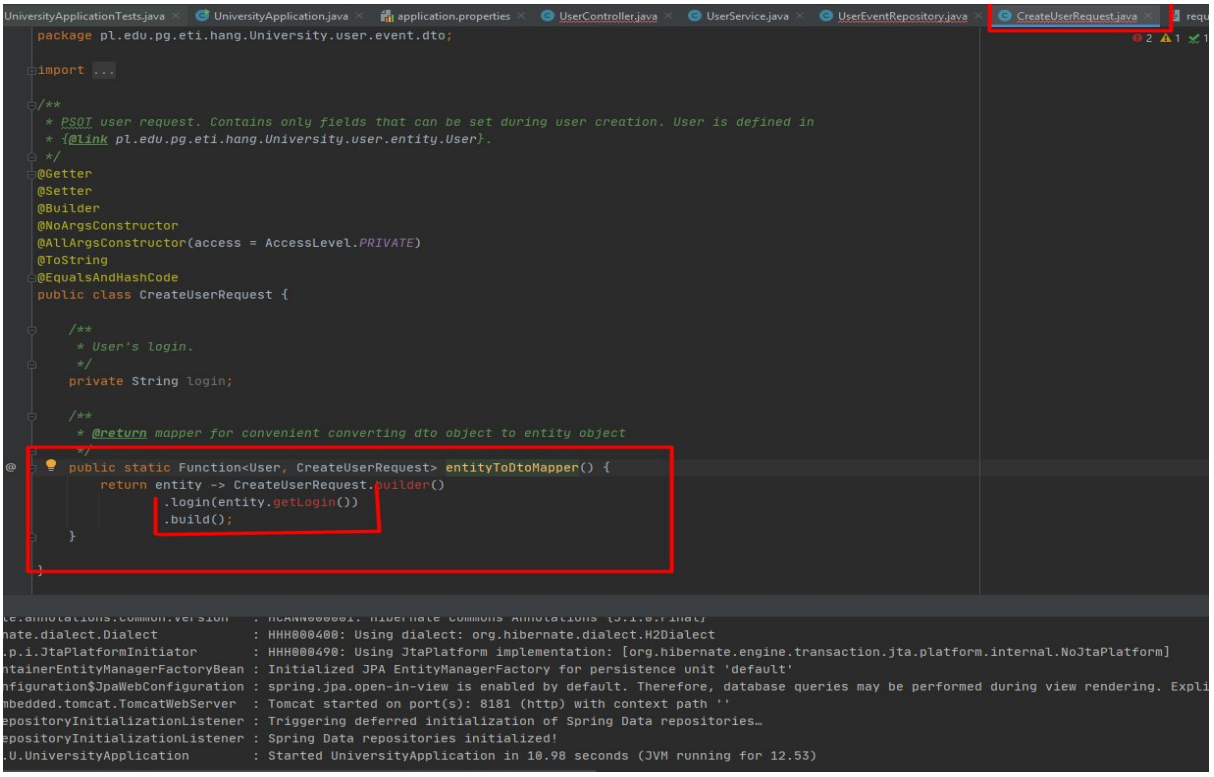
2. Implementation of inter-applications event-based communication. When removing existing or adding new category the elements management application should be notified in order to remove elements or create new simplified category record in the database. As event communication REST services can be used. (3 point).

I would like to show you in University_User application, pl.edu.pg.eti.hang.University.user.service , pl.edu.pg.eti.hang.University.user.event.repository and pl.edu.pg.eti.hang.University.user.event.dto.





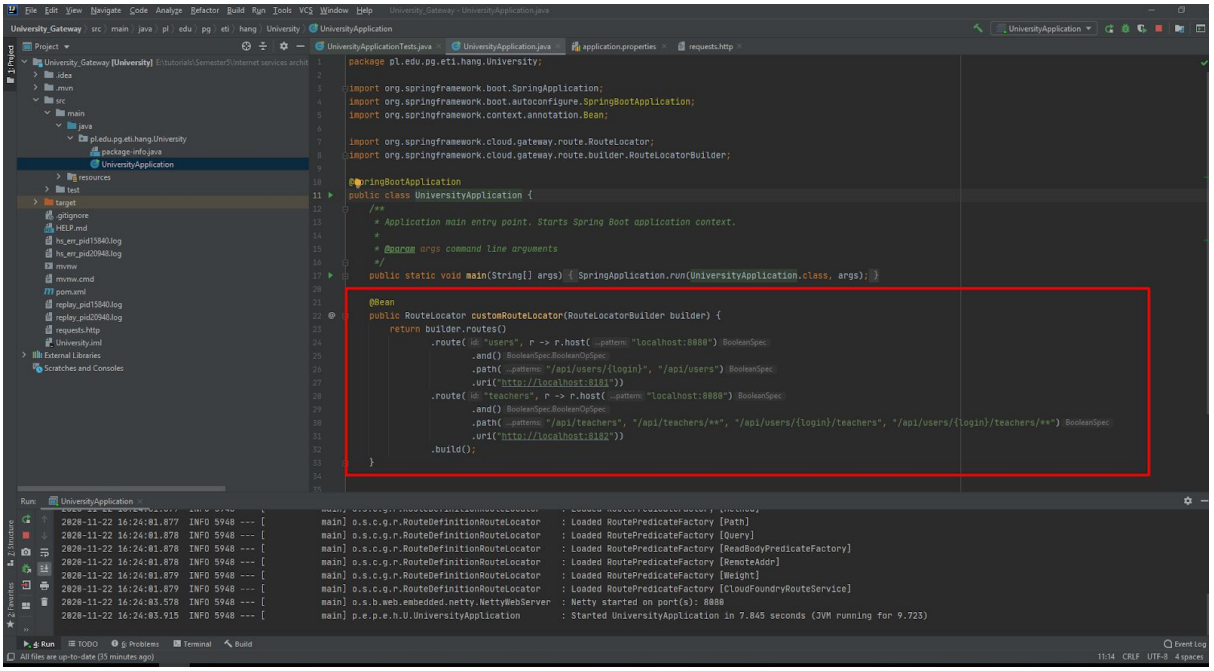
```
UniversityApplicationTests.java | UniversityApplication.java | application.properties | UserController.java | UserService.java | UserEventRepository.java | requests

package pl.edu.pg.eti.hang.University.user.event.repository;

import ...

@Repository
public class UserEventRepository  {

    private RestTemplate restTemplate;

    @Autowired
    public UserEventRepository(@Value("${University.teachers.url}") String baseUrl) {
        restTemplate = new RestTemplateBuilder().rootUri(baseUrl).build();
    }

    public void delete(User user) { restTemplate.delete("/users/{username}", user.getLogin()); }

    public void create(User user) {
        restTemplate.postForLocation( url "/users", CreateUserRequest.entityToDtoMapper().apply(user));
    }
}
```



```
UniversityApplicationTests.java | UniversityApplication.java | application.properties | UserController.java | UserService.java | UserEventRepository.java | CreateUserRequest.java

package pl.edu.pg.eti.hang.University.user.event.dto;

import ...

/**
 * PSOT user request. Contains only fields that can be set during user creation. User is defined in
 * {@link pl.edu.pg.eti.hang.University.user.entity.User}.
 */
@Getter
@Setter
@Builder
@NoArgsConstructor
@AllArgsConstructor(access = AccessLevel.PRIVATE)
@ToString
@EqualsAndHashCode
public class CreateUserRequest {

    /**
     * User's login.
     */
    private String login;

    /**
     * @return mapper for convenient converting dto object to entity object
     */
    public static Function<User, CreateUserRequest> entityToDtoMapper() {
        return entity -> CreateUserRequest.builder()
                .login(entity.getLogin())
                .build();
    }

}

nate.dialect.Dialect            : HHH000400: Using dialect: org.hibernate.dialect.H2Dialect
.p.i.JtaPlatformInitiator       : HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
ntainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
nfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed during view rendering. Expli
mbedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8181 (http) with context path ''
epositoryInitializationListener : Triggering deferred initialization of Spring Data repositories…
epositoryInitializationListener : Spring Data repositories initialized!
.U.UniversityApplication        : Started UniversityApplication in 10.98 seconds (JVM running for 12.53)
```

And the REST services are used in the event communication, which is also shown from the previous pictures.

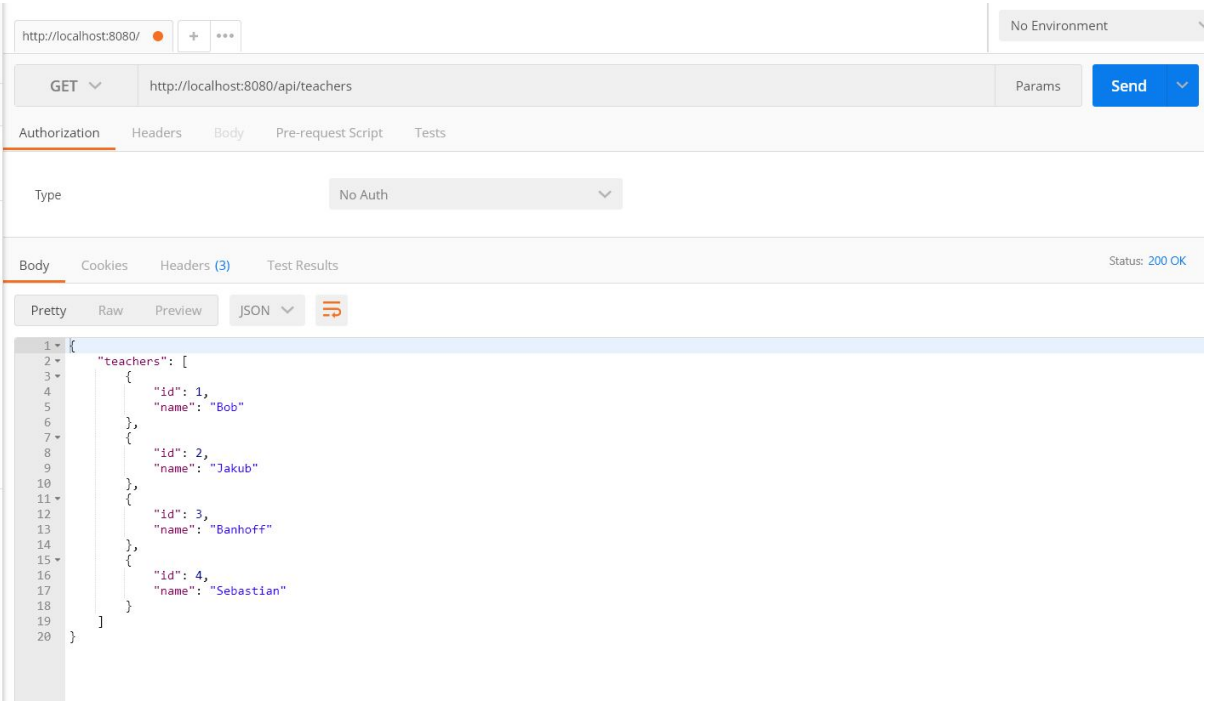
3. New Spring Boot application based on Spring Cloud Gateway should be created.
The application should contain routing rules for category and elements management applications. (1 point).
University_Gateway application is created to implement this task. And also it provides the routing rule. This application uses the port 8080 during the test.



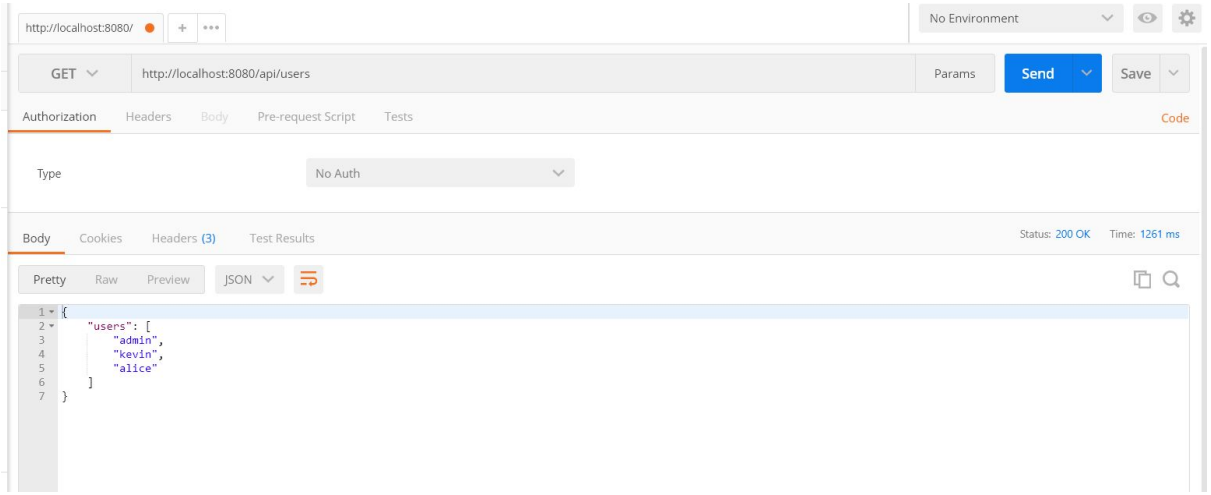
Testing HTTP requests:

I am using the postman app to do testing.

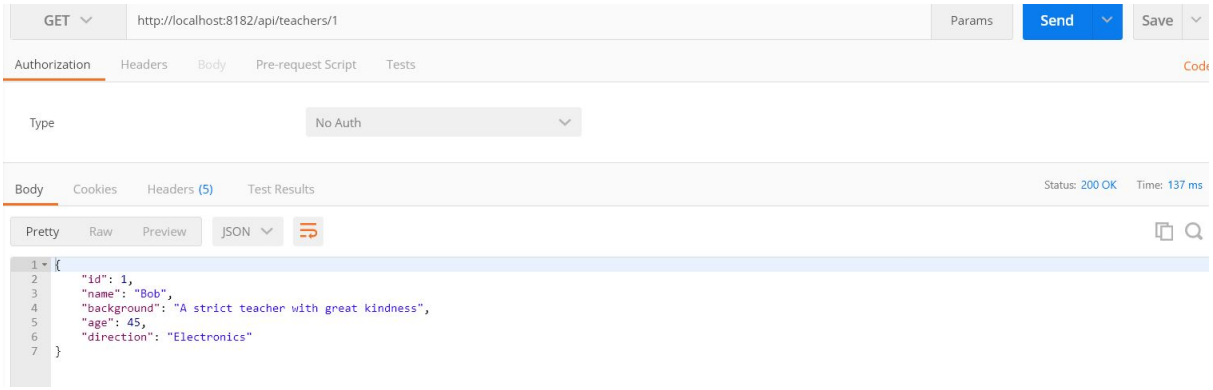
Show all teachers through port8080:



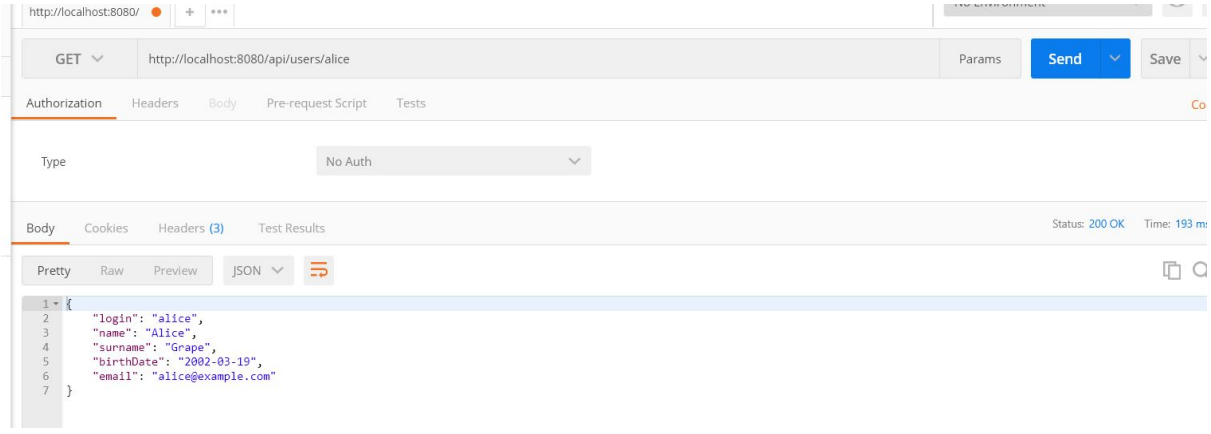
Show all users through port8080:

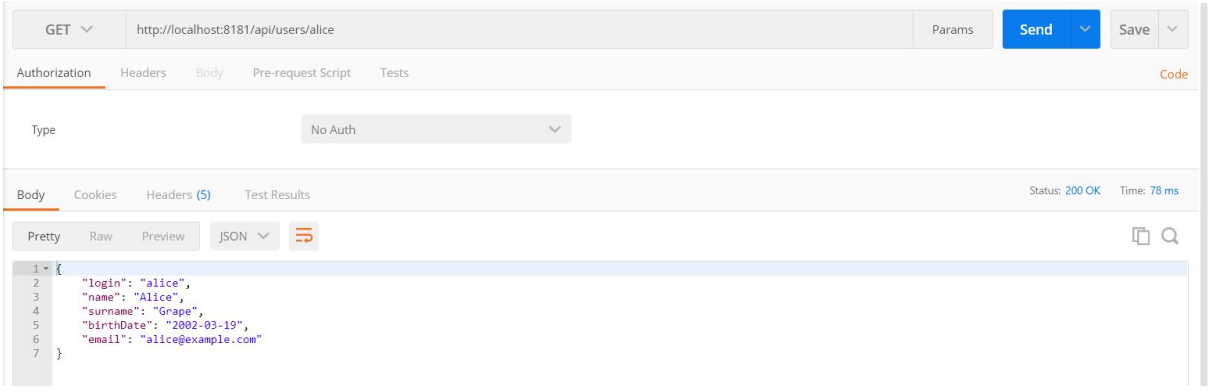


Show teacher whoseId is 1 through port 8182:

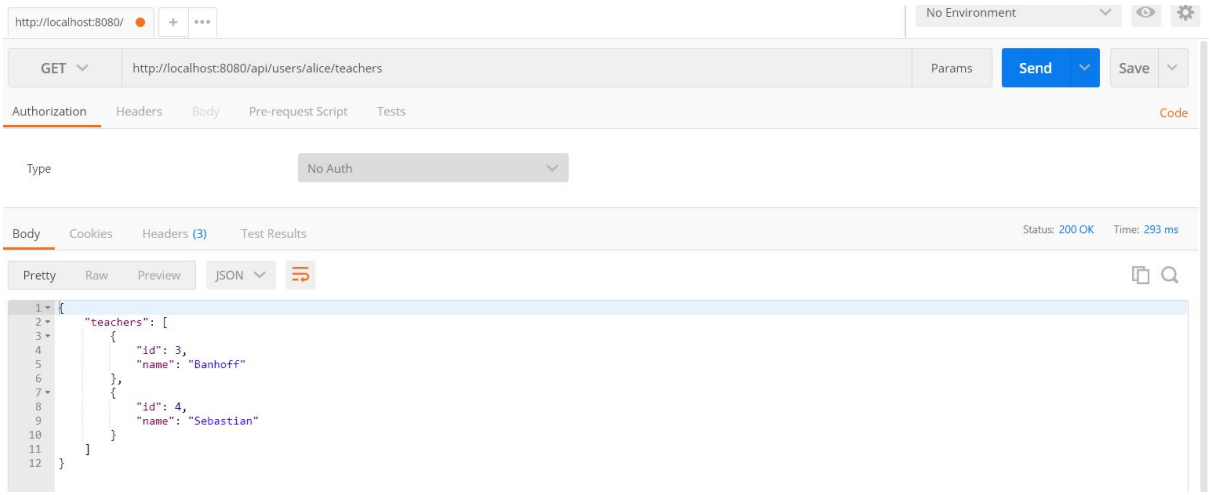


Show the user Alice's information through port either 8080 or 8181:

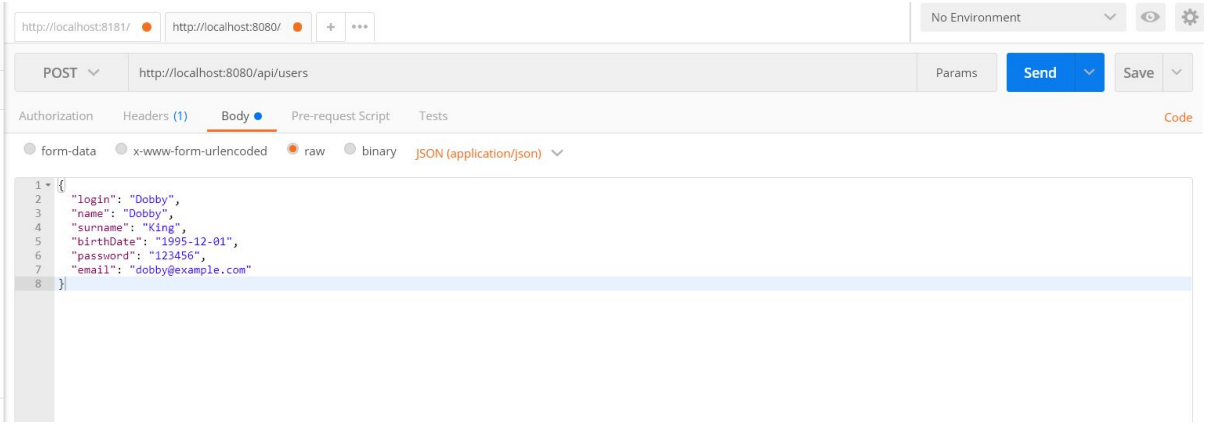


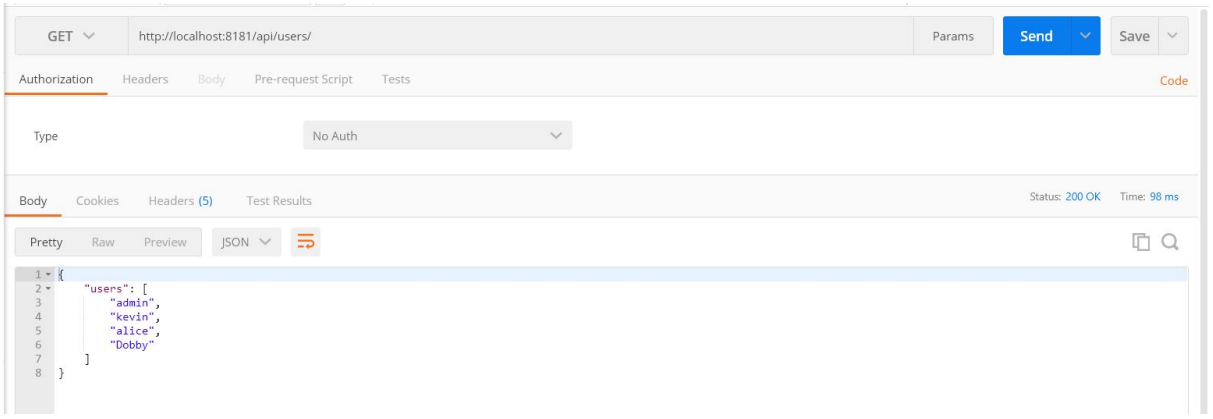


Show which teacher does Alice have through port8080:

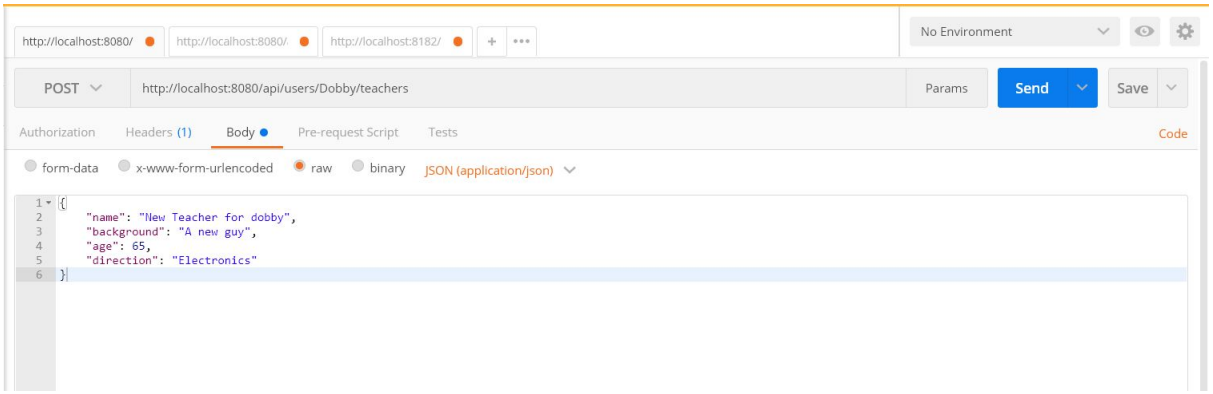


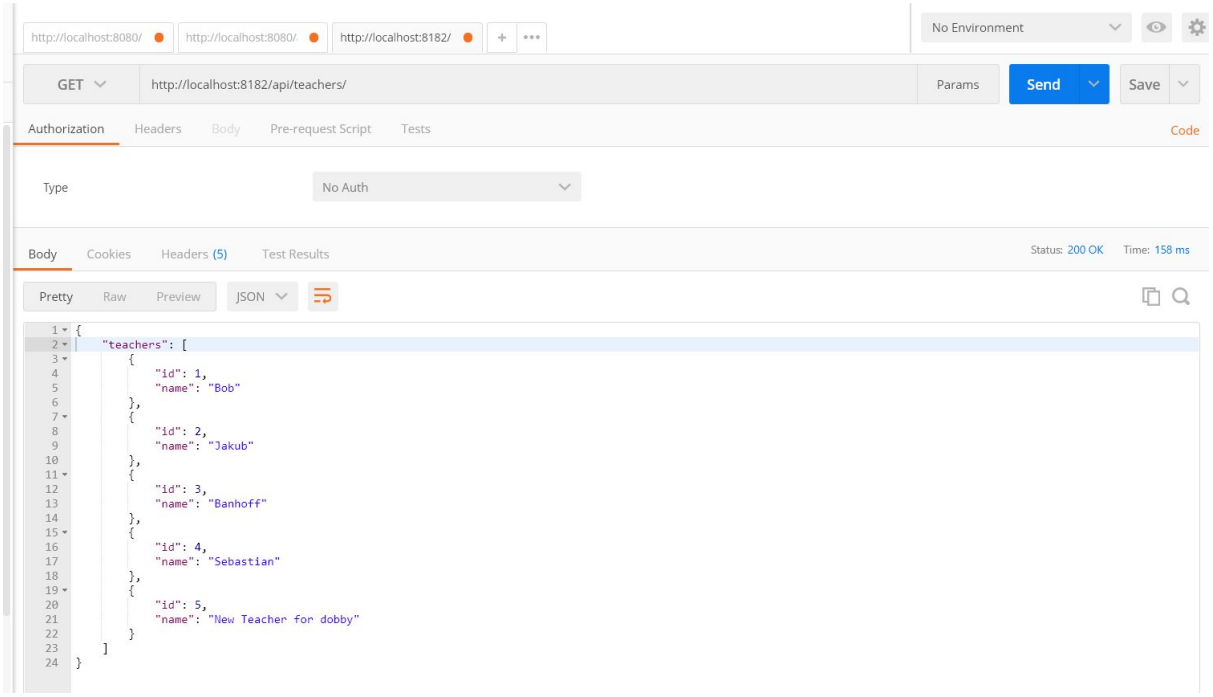
Create User with Login “dobby”:

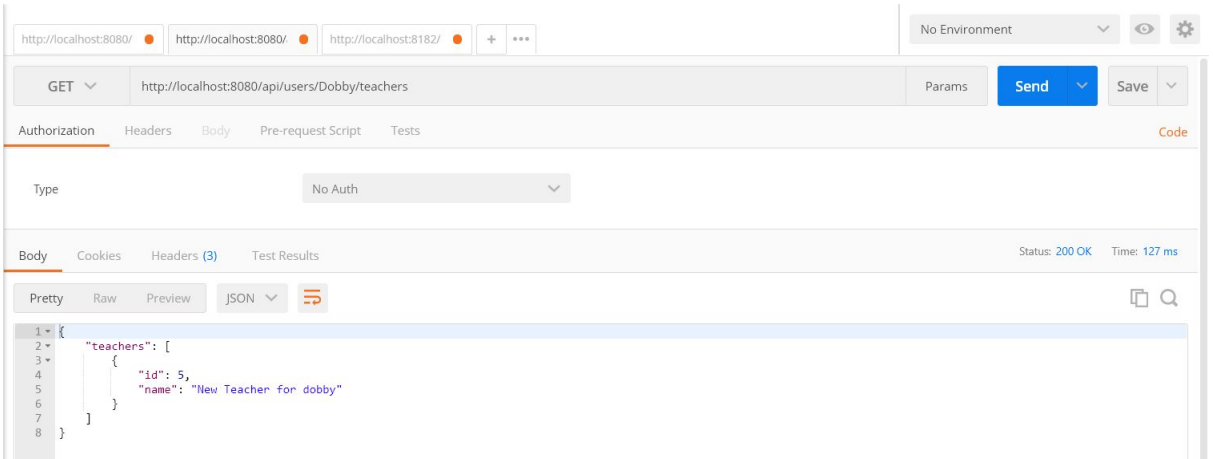




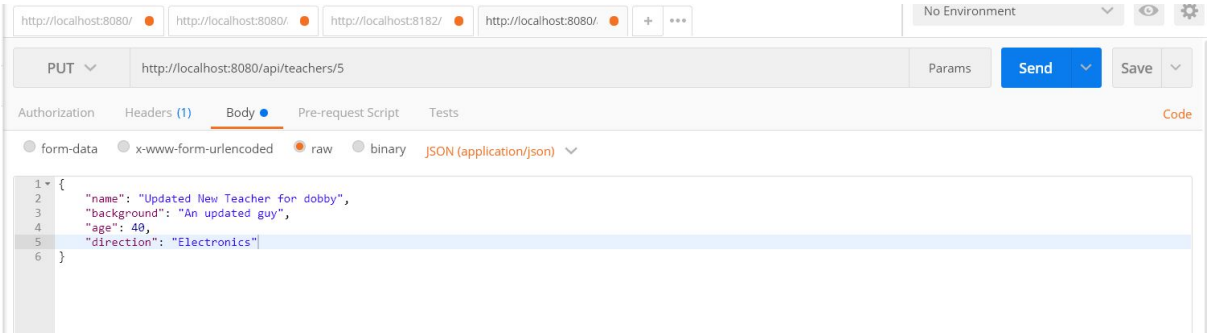
Create a new teacher for our new user Dobby:

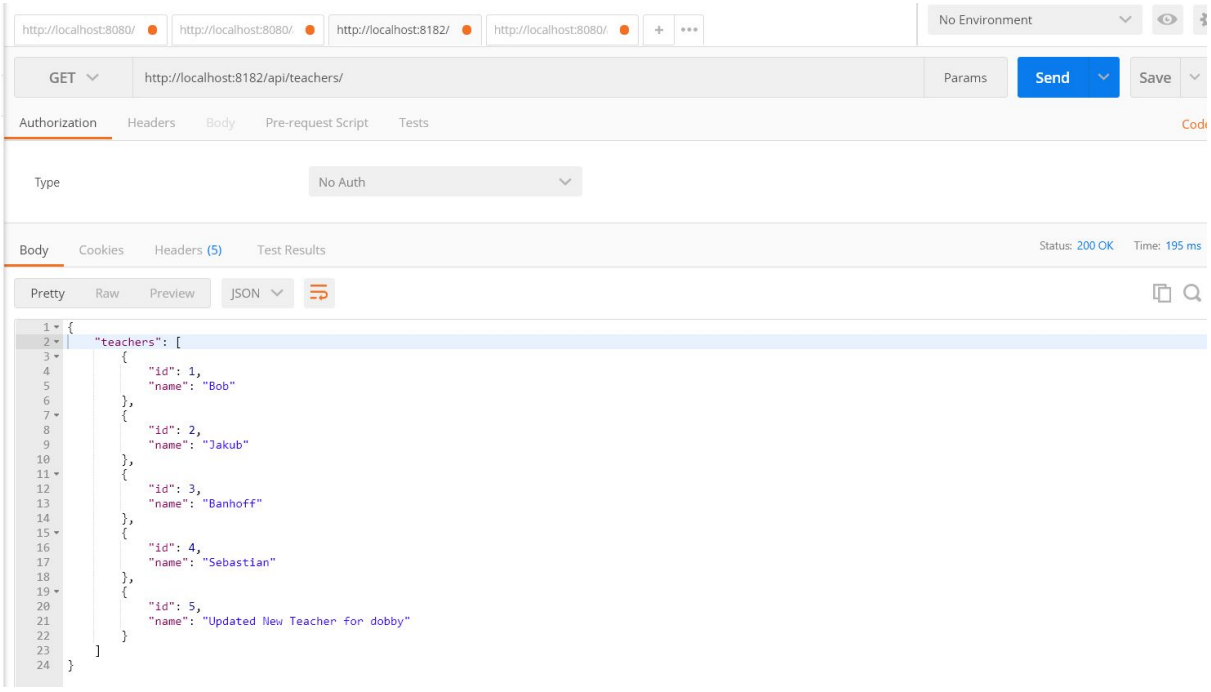


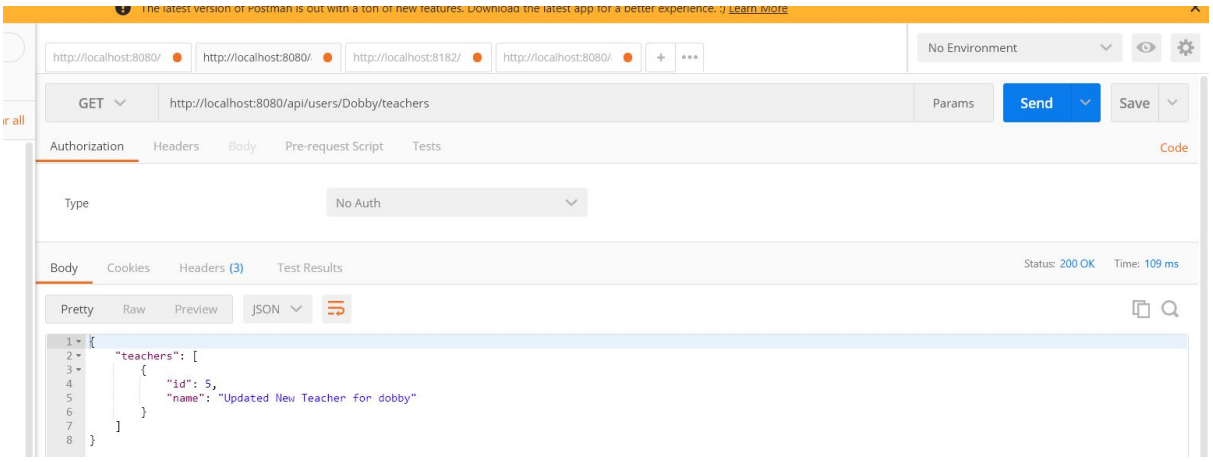




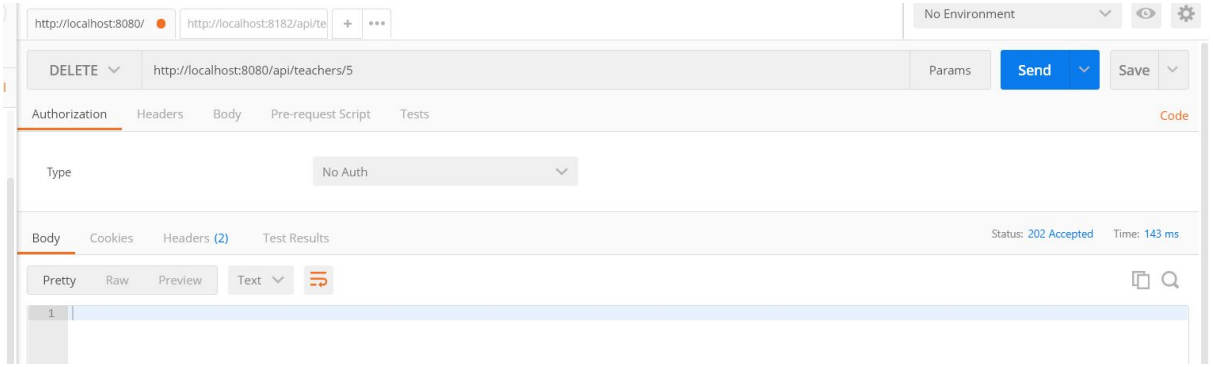
Update the just created dobby's teacher's information:

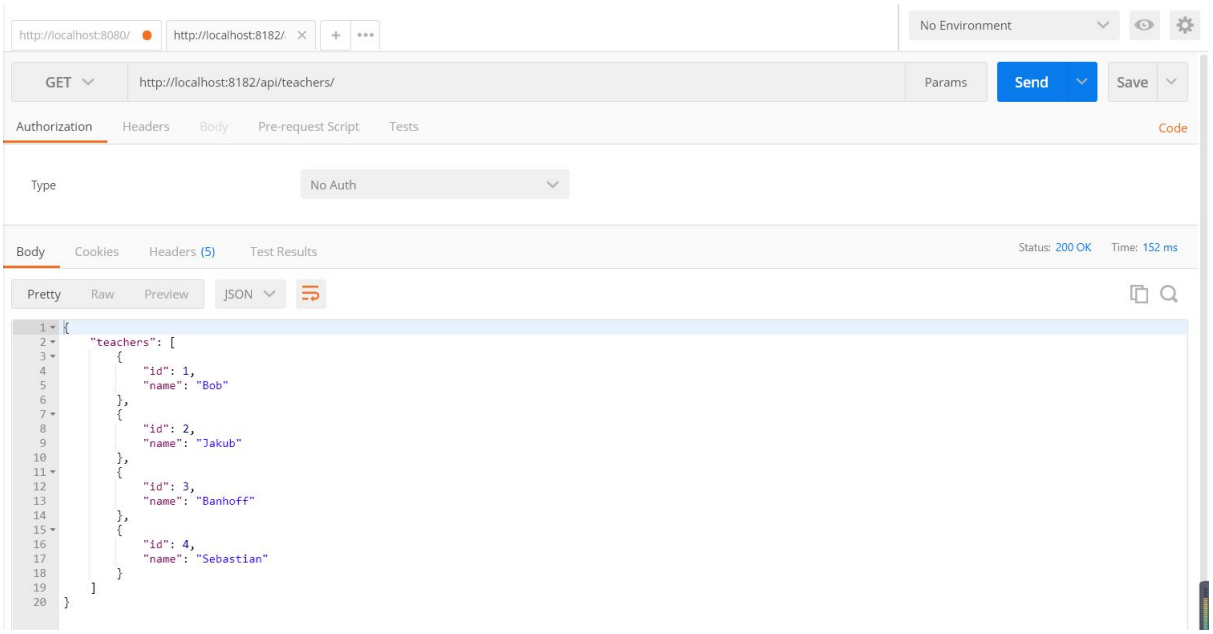




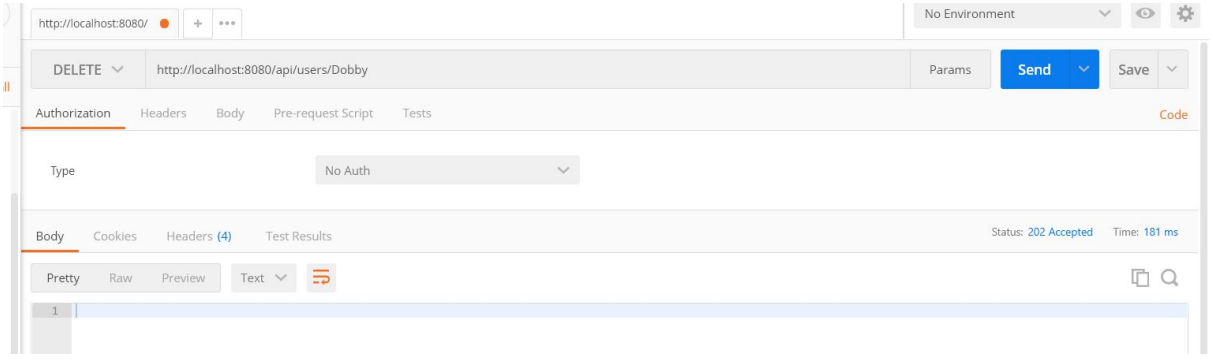


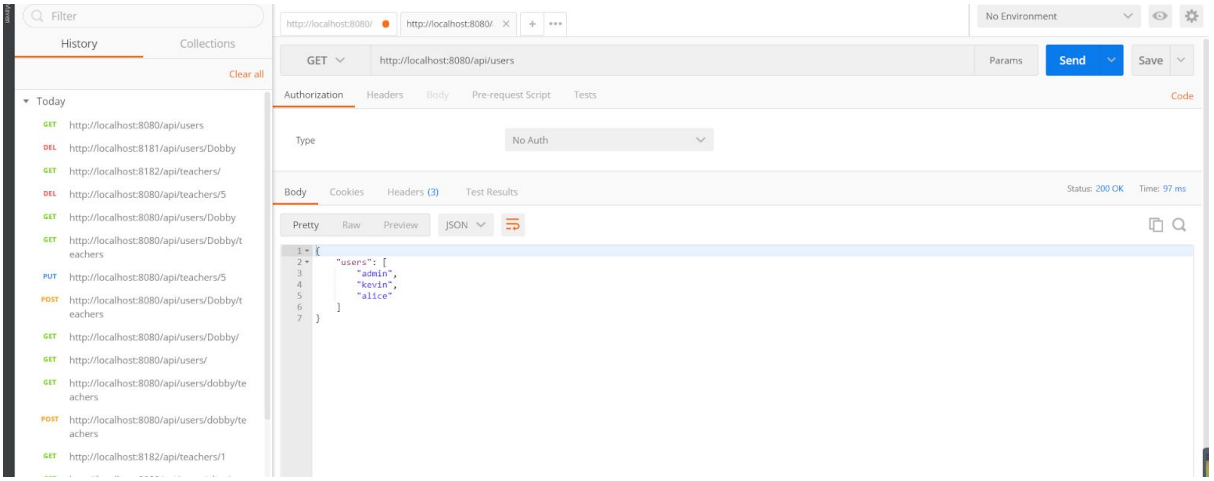
Delete Dobby's teacher:





Delete user Dobby:




Filter
History
Collections
Clear all
Today
GET http://localhost:8080/api/users
DEL http://localhost:8181/api/users/Dobby
GET http://localhost:8182/api/teachers/
DEL http://localhost:8080/api/teachers/5
GET http://localhost:8080/api/users/Dobby
GET http://localhost:8080/api/users/Dobby/teachers
PUT http://localhost:8080/api/teachers/5
POST http://localhost:8080/api/users/Dobby/teachers
GET http://localhost:8080/api/users/Dobby/
GET http://localhost:8080/api/users/
GET http://localhost:8080/api/users/dobby/teachers
POST http://localhost:8080/api/users/dobby/teachers
GET http://localhost:8182/api/teachers/1
http://localhost:8080/
http://localhost:8080/
No Environment
GET
http://localhost:8080/api/users
Params
Send
Save
Authorization
Headers
Body
Pre-request Script
Tests
Code
Type
No Auth
Body
Cookies
Headers (3)
Test Results
Status: 200 OK
Time: 97 ms
Pretty
Raw
Preview
JSON
{
"users": [
"admin",
"kevin",
"alice"
]
}