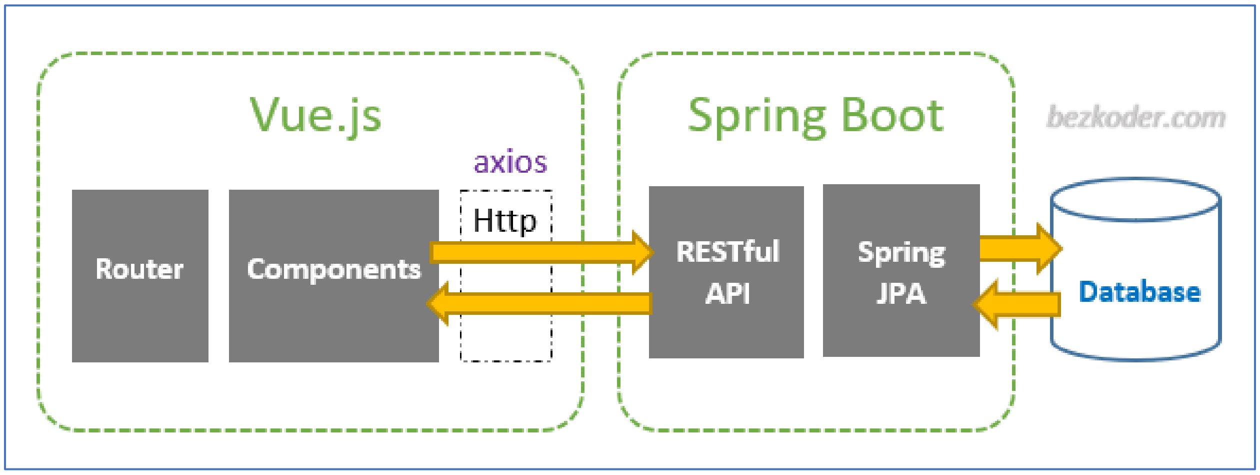
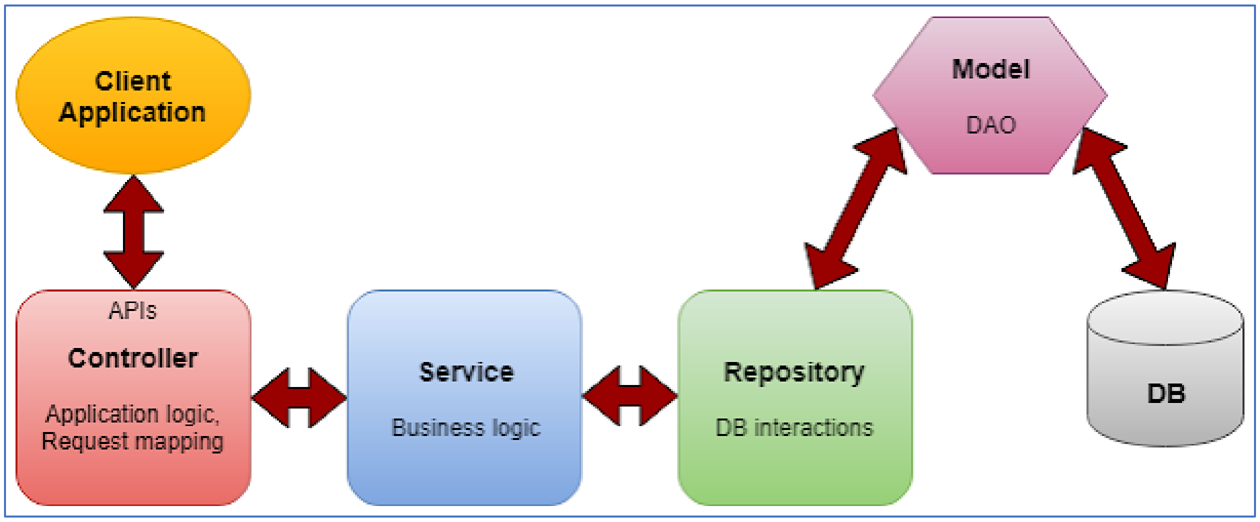
CSIS3275 Software Engineering
Lab on SpringBoot and Vue.js
By Ivan Wong @ Douglas College

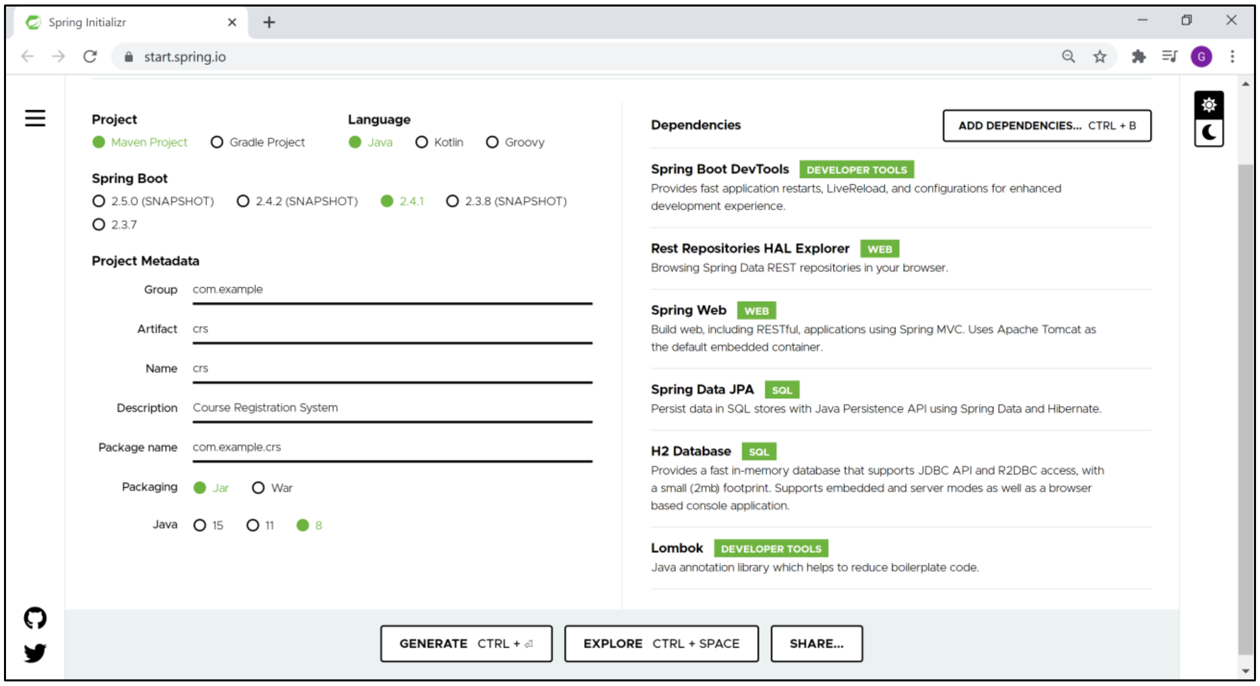


(Source: https://www.bezkoder.com/spring-boot-vue-js-crud-example/)



(Source: http://randikatech.blogspot.com/2019/09/get-your-hands-dirty-with-micro-services.html)

1. **Go to https://start.spring.io/. Create a Springboot project by choose the dependencies.**



Project
● Maven Project ○ Gradle Project

Language
● Java ○ Kotlin ○ Groovy

Spring Boot
○ 2.5.0 (SNAPSHOT) ○ 2.4.2 (SNAPSHOT) ● 2.4.1 ○ 2.3.8 (SNAPSHOT) ○ 2.3.7

Project Metadata

Group	com.example
Artifact	crs
Name	crs
Description	Course Registration System
Package name	com.example.crs
Packaging	● Jar ○ War
Java	○ 15 ○ 11 ● 8

Dependencies

Spring Boot DevTools DEVELOPER TOOLS
Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

Rest Repositories HAL Explorer WEB
Browsing Spring Data REST repositories in your browser.

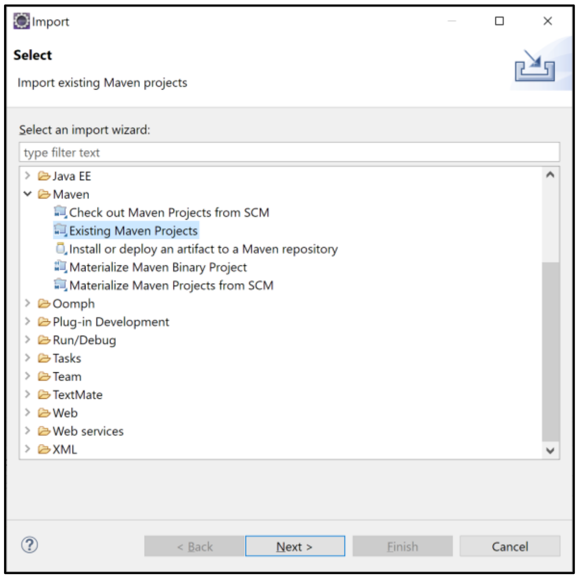
Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring Data JPA SQL
Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

H2 Database SQL
Provides a fast in-memory database that supports JDBC API and R2DBC access, with a small (2mb) footprint. Supports embedded and server modes as well as a browser based console application.

Lombok DEVELOPER TOOLS
Java annotation library which helps to reduce boilerplate code.

2. Open the project in Eclipse


Import

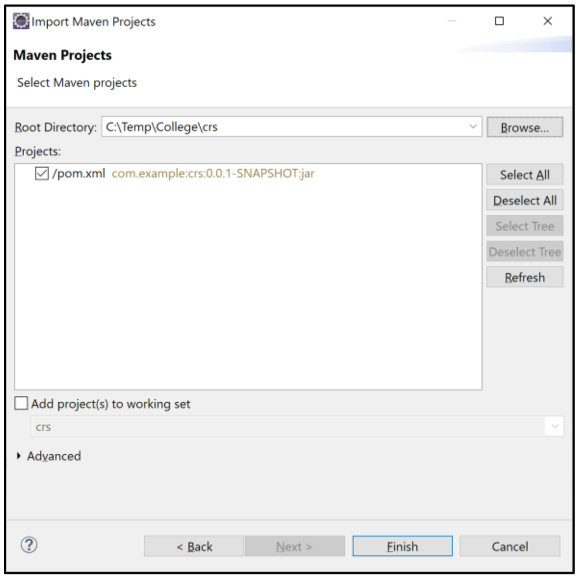
Select

Import existing Maven projects

Select an import wizard:

type filter text

```
> Java EE
v Maven
    Check out Maven Projects from SCM
    Existing Maven Projects
    Install or deploy an artifact to a Maven repository
    Materialize Maven Binary Project
    Materialize Maven Projects from SCM
> Oomph
> Plug-in Development
> Run/Debug
> Tasks
> Team
> TextMate
> Web
> Web services
> XML
```


Import Maven Projects

Maven Projects

Select Maven projects

Root Directory: C:\Temp\College\crs

Projects:

☑ /pom.xml com.example:crs:0.0.1-SNAPSHOT:jar

☐ Add project(s) to working set

crs

Advanced

If necessary, change the Java build path as follows:

Project → Properties → Java Build Path → Choose the Libraries Tab:

Remove and JRE System Library

Add Library → JRE System Library → Click Installed JRE → Choose Jkd15

3. Add H2 Database information

```
spring.datasource.url=jdbc:h2:mem:testdb
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=sa
spring.datasource.password=
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
```

/crs/src/main/resources/application.properties

4. Create Entity class

```
package com.example.crs.model;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
```

```
@Entity
@Table(name = "courses")
public class Course {

	@Id
	@GeneratedValue(strategy = GenerationType.AUTO)
	private long id;

	@Column(name = "code")
	private String code;

	@Column(name = "title")
	private String title;

	public Course() {
	}

	public Course(String code, String title) {
		this.code = code;
		this.title = title;
	}

	public long getId() {
		return id;
	}

	public void setId(long id) {
		this.id = id;
	}

	public String getCode() {
		return code;
	}

	public void setCode(String code) {
		this.code = code;
	}

	public String getTitle() {
		return title;
	}

	public void setTitle(String title) {
		this.title = title;
	}

}
```
/crs/src/main/java/com/example/crs/model/Course.java

5. Create Repository

```
package com.example.crs.model;
```

```java
import java.util.List;

import org.springframework.data.jpa.repository.JpaRepository;


public interface CourseRepository extends JpaRepository<Course, Long> {

	List<Course> findByTitle(String title);

	List<Course> findByCode(String code);

}
```
/crs/src/main/java/com/example/crs/model/CourseRepository.java

6. **Create Controller**

```java
package com.example.crs.controller;

import java.util.ArrayList;
import java.util.List;
import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.DeleteMapping;
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.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;

import com.example.crs.model.Course;
import com.example.crs.model.CourseRepository;

@CrossOrigin(origins = "http://localhost:8081") // used for vue
@RestController
@RequestMapping("/api ")
public class CourseController {

	@Autowired
	CourseRepository courseRepository;

	@GetMapping("/courses")
	public ResponseEntity<List<Course>> getAllCourses(@RequestParam(required = 
false) String title) {
		try {
			List<Course> courses = new ArrayList<Course>();
```

```
                    if (title == null)
                        courseRepository.findAll().forEach(courses::add);
                    else

        courseRepository.findByTitleContaining(title).forEach(courses::add);

                    if (courses.isEmpty()) {
                        return new ResponseEntity<>(HttpStatus.NO_CONTENT);
                    }

                    return new ResponseEntity<>(courses, HttpStatus.OK);
                } catch (Exception e) {
                    return new ResponseEntity<>(null,
HttpStatus.INTERNAL_SERVER_ERROR);
                }
        }

        @GetMapping("/courses/{id}")
        public ResponseEntity<Course> getCourseById(@PathVariable("id") long id) {
                Optional<Course> courseData = courseRepository.findById(id);

                if (courseData.isPresent()) {
                    return new ResponseEntity<>(courseData.get(), HttpStatus.OK);
                } else {
                    return new ResponseEntity<>(HttpStatus.NOT_FOUND);
                }
        }

        @PostMapping("/courses")
        public ResponseEntity<Course> createTutorial(@RequestBody Course course) {
                try {
                    Course _course = courseRepository.save(new
Course(course.getCode(), course.getTitle()));
                    return new ResponseEntity<>(_course, HttpStatus.CREATED);
                } catch (Exception e) {
                    return new ResponseEntity<>(null,
HttpStatus.INTERNAL_SERVER_ERROR);
                }
        }

        @PutMapping("/courses/{id}")
        public ResponseEntity<Course> updateCourse(@PathVariable("id") long id,
@RequestBody Course course) {
                Optional<Course> courseData = courseRepository.findById(id);

                if (courseData.isPresent()) {
                    Course _course = courseData.get();
                    _course.setCode(course.getCode());
                    _course.setTitle(course.getTitle());
                    return new ResponseEntity<>(courseRepository.save(_course),
HttpStatus.OK);
                } else {
                    return new ResponseEntity<>(HttpStatus.NOT_FOUND);
                }
```

```
	}

	@DeleteMapping("/courses/{id}")
	public ResponseEntity<HttpStatus> deleteCourse(@PathVariable("id") long id) {
		try {
			courseRepository.deleteById(id);
			return new ResponseEntity<>(HttpStatus.NO_CONTENT);
		} catch (Exception e) {
			return new ResponseEntity<>(HttpStatus.INTERNAL_SERVER_ERROR);
		}
	}

	@DeleteMapping("/courses")
	public ResponseEntity<HttpStatus> deleteAllTutorials() {
		try {
			courseRepository.deleteAll();
			return new ResponseEntity<>(HttpStatus.NO_CONTENT);
		} catch (Exception e) {
			return new ResponseEntity<>(HttpStatus.INTERNAL_SERVER_ERROR);
		}

	}

}
```
/crs/src/main/java/com/example/crs/controller/CourseController.java

7. Create the SpringBootApplication

```
package com.example.crs;

import org.springframework.boot.ApplicationRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.Bean;

import com.example.crs.model.Course;
import com.example.crs.model.CourseRepository;

@SpringBootApplication
public class CrsApplication {

	public static void main(String[] args) {
		SpringApplication.run(CrsApplication.class, args);
	}

	@Bean
	ApplicationRunner init(CourseRepository repository) {
		return args -> {

			repository.save(new Course("CSIS2175", "ADVANCED INTEGRATED 
SOFTWARE DEVELOPMENT"));
			repository.save(new Course("CSIS3275", "SOFTWARE ENGINEERING"));
			repository.save(new Course("CSIS1190", "EXCEL FOR BUSINESS"));
```

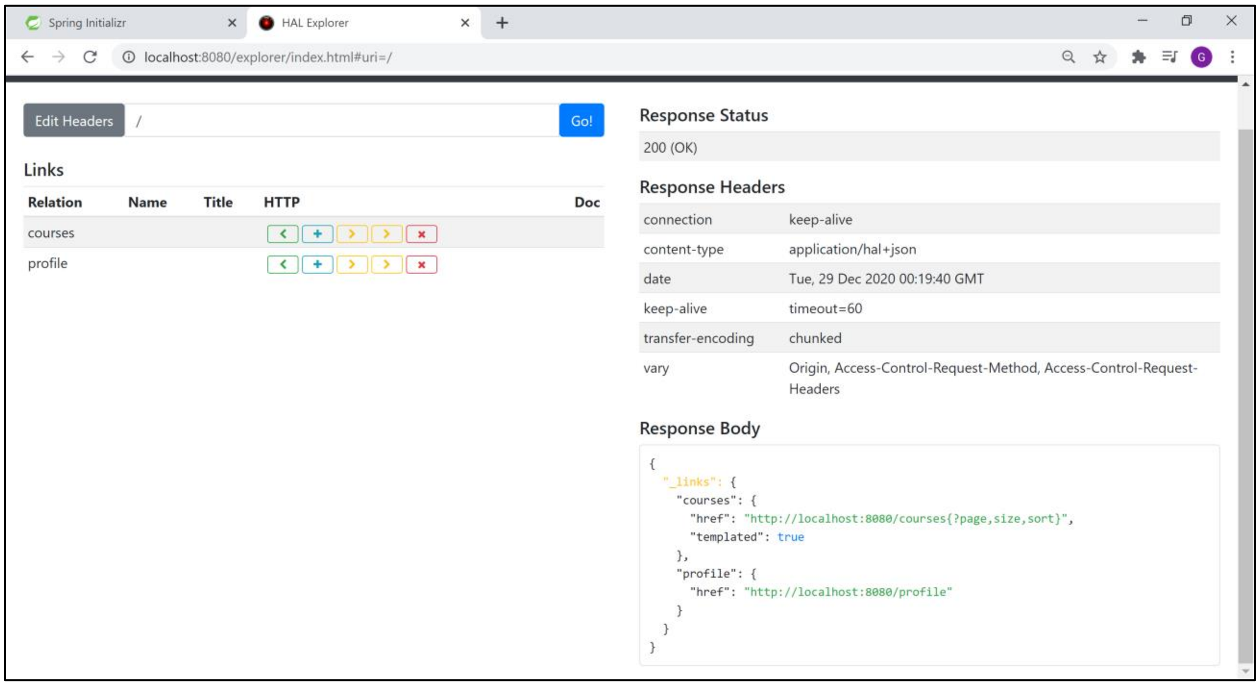
```
                        repository.findAll().forEach(System.out::println);
                };
        }

}
```

/crs/src/main/java/com/example/crs/CrsApplication.java

8. **Start the server: Right-click the project, choose Run As. Then choose Maven Build and input spring-boot:run for the goal field.**

9. **Test the server: http://localhost:8080/**



Links

Relation	Name	Title	HTTP	Doc
courses				
profile				

Response Status

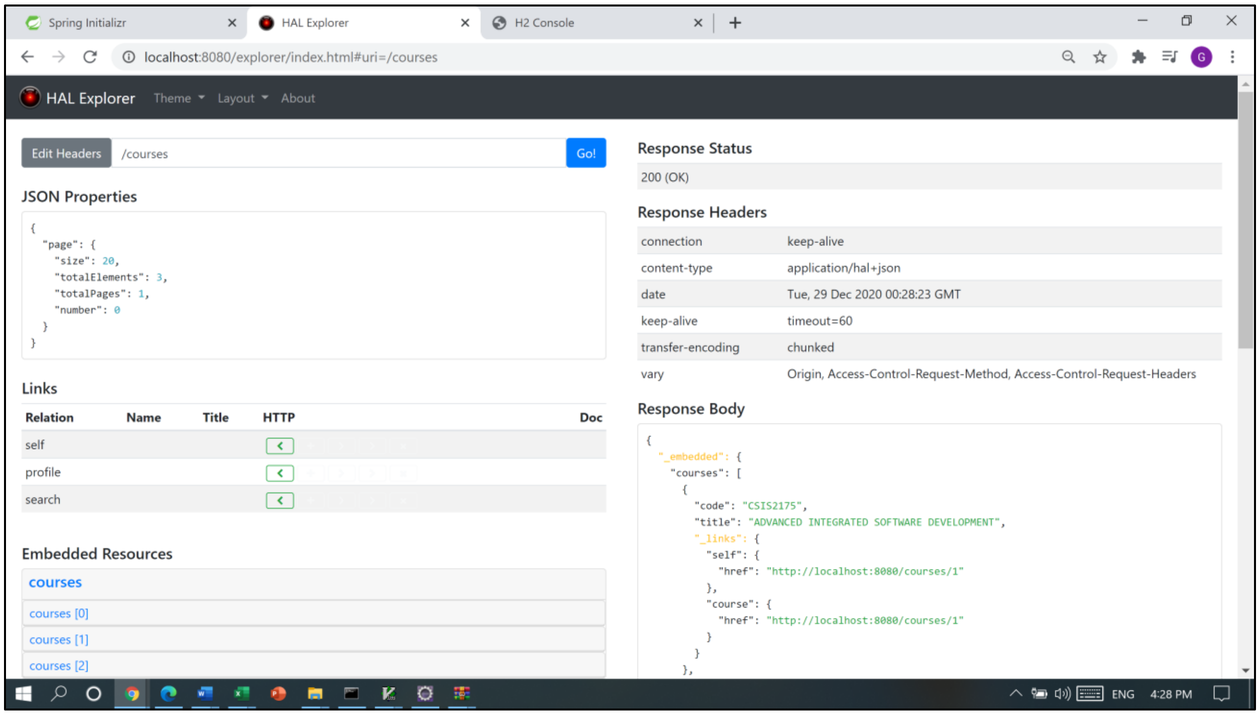
200 (OK)

Response Headers

connection	keep-alive
content-type	application/hal+json
date	Tue, 29 Dec 2020 00:19:40 GMT
keep-alive	timeout=60
transfer-encoding	chunked
vary	Origin, Access-Control-Request-Method, Access-Control-Request-Headers

Response Body

```
{
  "_links": {
    "courses": {
      "href": "http://localhost:8080/courses{?page,size,sort}",
      "templated": true
    },
    "profile": {
      "href": "http://localhost:8080/profile"
    }
  }
}
```



/courses

JSON Properties

```
{
  "page": {
    "size": 20,
    "totalElements": 3,
    "totalPages": 1,
    "number": 0
  }
}
```

Links

Relation	Name	Title	HTTP	Doc
self				
profile				
search				

Embedded Resources

courses

courses [0]

courses [1]

courses [2]

Response Status

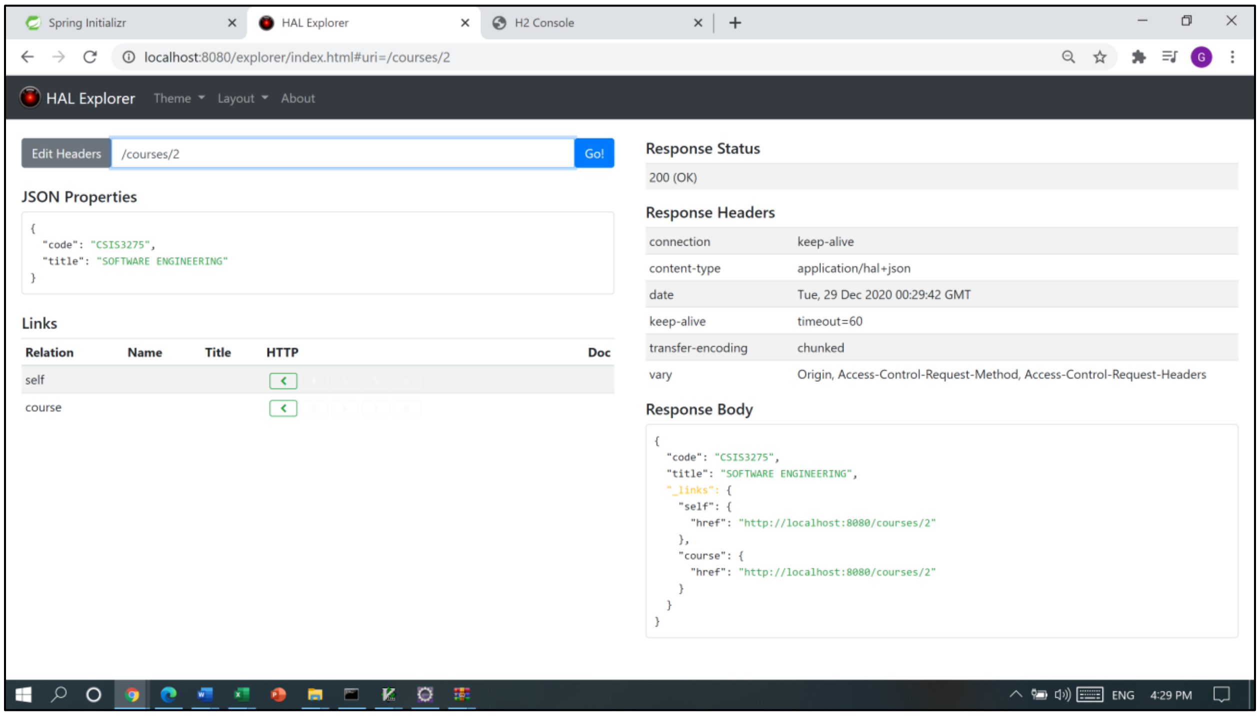
200 (OK)

Response Headers

connection	keep-alive
content-type	application/hal+json
date	Tue, 29 Dec 2020 00:28:23 GMT
keep-alive	timeout=60
transfer-encoding	chunked
vary	Origin, Access-Control-Request-Method, Access-Control-Request-Headers

Response Body

```
{
  "_embedded": {
    "courses": [
      {
        "code": "CSIS2175",
        "title": "ADVANCED INTEGRATED SOFTWARE DEVELOPMENT",
        "_links": {
          "self": {
            "href": "http://localhost:8080/courses/1"
          },
          "course": {
            "href": "http://localhost:8080/courses/1"
          }
        }
      },
```



/courses/2

JSON Properties

```
{
  "code": "CSIS3275",
  "title": "SOFTWARE ENGINEERING"
}
```

Links

Relation	Name	Title	HTTP	Doc
self				
course				

Response Status

200 (OK)

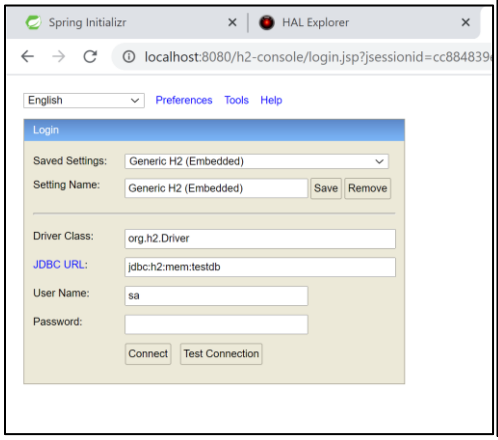
Response Headers

connection	keep-alive
content-type	application/hal+json
date	Tue, 29 Dec 2020 00:29:42 GMT
keep-alive	timeout=60
transfer-encoding	chunked
vary	Origin, Access-Control-Request-Method, Access-Control-Request-Headers

Response Body

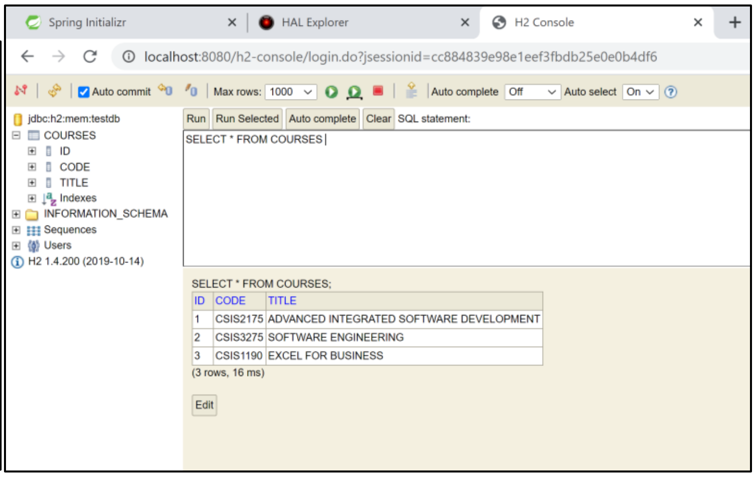
```
{
  "code": "CSIS3275",
  "title": "SOFTWARE ENGINEERING",
  "_links": {
    "self": {
      "href": "http://localhost:8080/courses/2"
    },
    "course": {
      "href": "http://localhost:8080/courses/2"
    }
  }
}
```

10. **Test the DB: http://localhost:8080/h2-console**



localhost:8080/h2-console/login.jsp?jsessionid=cc884839e

Saved Settings:	Generic H2 (Embedded)
Setting Name:	Generic H2 (Embedded)
Driver Class:	org.h2.Driver
JDBC URL:	jdbc:h2:mem:testdb
User Name:	sa
Password:	



localhost:8080/h2-console/login.do?jsessionid=cc884839e98e1eef3fbdb25e0e0b4df6

```
jdbc:h2:mem:testdb
  COURSES
    ID
    CODE
    TITLE
    Indexes
  INFORMATION_SCHEMA
  Sequences
  Users
H2 1.4.200 (2019-10-14)
```

```
SELECT * FROM COURSES
```

SELECT * FROM COURSES;

ID	CODE	TITLE
1	CSIS2175	ADVANCED INTEGRATED SOFTWARE DEVELOPMENT
2	CSIS3275	SOFTWARE ENGINEERING
3	CSIS1190	EXCEL FOR BUSINESS

(3 rows, 16 ms)

References:

https://bezkoder.com/spring-boot-vue-js-crud-example/

https://bezkoder.com/vue-js-crud-app/

https://bezkoder.com/spring-boot-jpa-crud-rest-api/

https://bezkoder.com/integrate-vue-spring-boot/

Authentication:

https://bezkoder.com/spring-boot-vue-js-authentication-jwt-spring-security/

https://bezkoder.com/spring-boot-jwt-authentication/

https://bezkoder.com/jwt-vue-vuex-authentication/