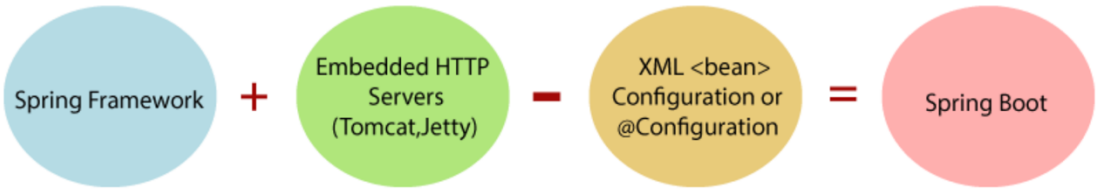
Spring Boot

Spring Boot Framework គឺ Framework ដែលបង្កើតដោយ "The Spring Team" ដើម្បីបន្ធូរបន្ថយការ Development នៃ Spring Application ថ្មីៗ ភាគច្រើន Spring Boot Application ត្រូវការ Spring configuration តិចតួចណាស់ៗ



ហេតុអ្វីបានជាប្រើប្រាស់ Spring Boot?

ព្រោះវាស្រួលរៀននិងងាយ Development លើ Web Application ។

សារប្រយោជន៍នៃ Spring Boot គឺសុវត្ថិភាព គុណភាព ហើយ Project ដែលបង្កើតដោយ Spring Boot គឺមានភាពរហ័សដើរលឿននិងមាន MVC មកជាមួយស្រាប់ៗ

ដំបូងត្រូវតម្លើង៖

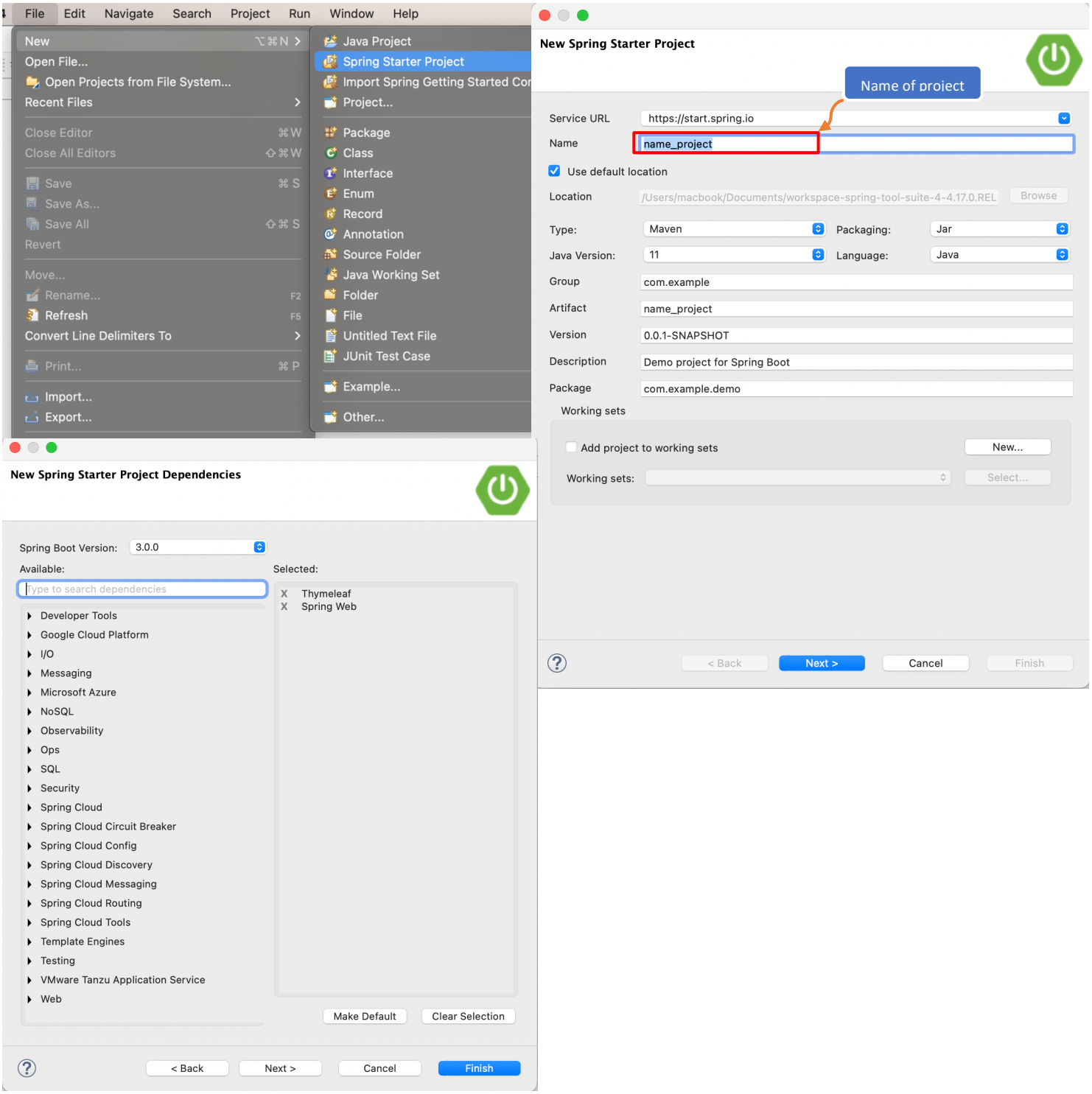
JDK version 8

https://www.oracle.com/java/technologies/downloads/

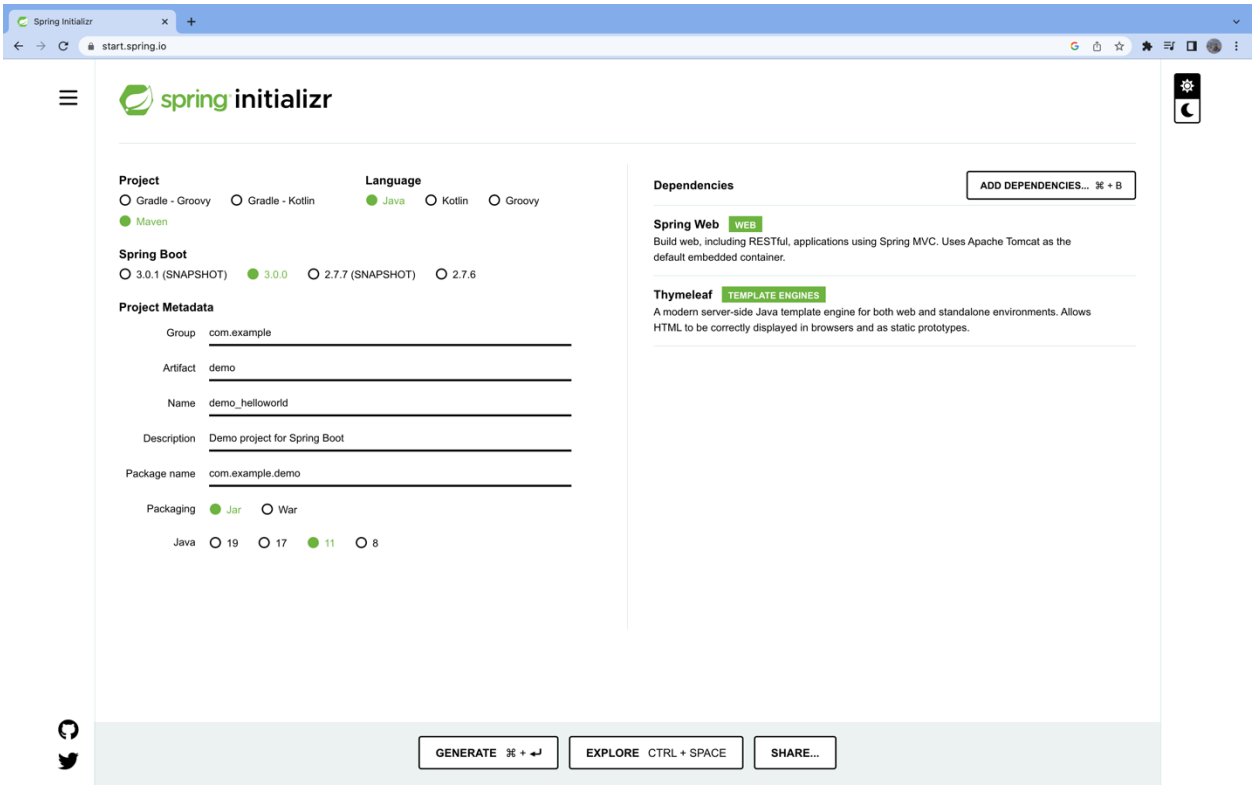
IDE មានដូចជា

របៀបបង្កើត Spring boot Project

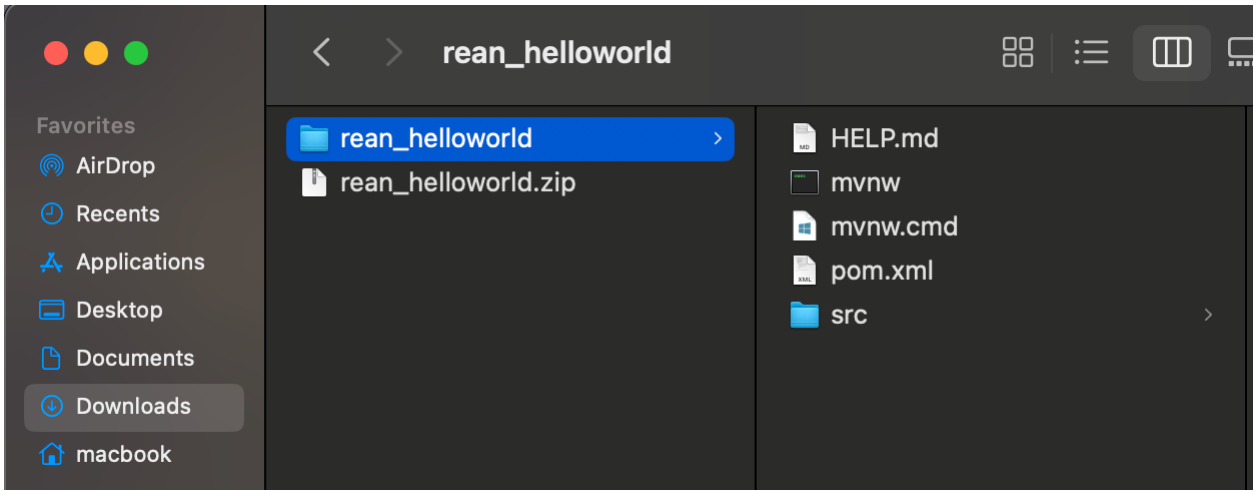
របៀបទីមួយបង្កើតនៅក្នុង Spring ToolSuite



របៀបទីពីរបង្កើតនៅក្នុង Website (Link : https://start.spring.io)

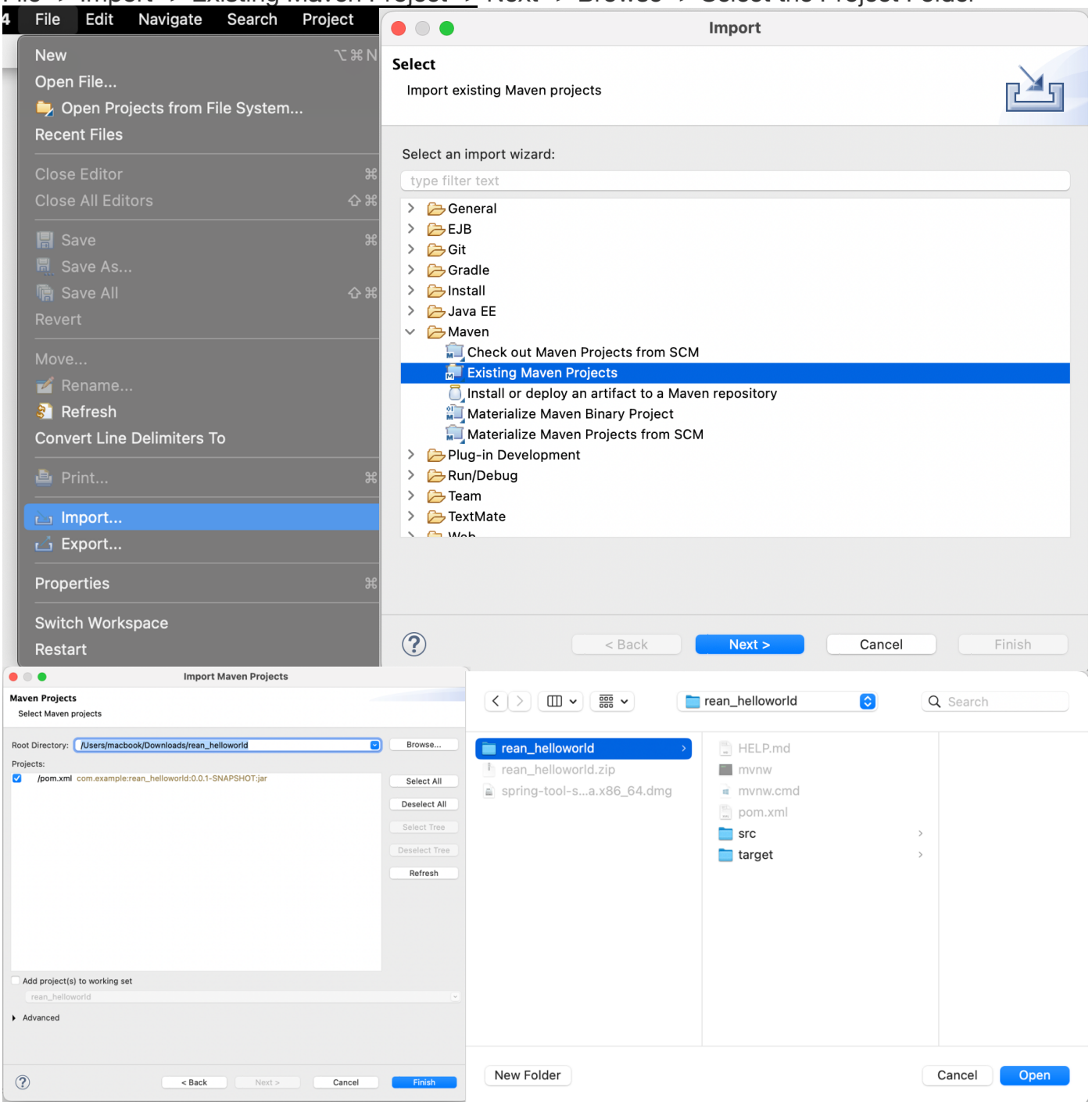


បន្ទាប់ពីចុច Generate គឺទទួលបាន File Zip មួយនិង Folder Project មួយ



របៀប Import Project ចូល Spring ToolSuite

File -> Import -> Existing Maven Project -> Next -> Browse -> Select the Project Folder

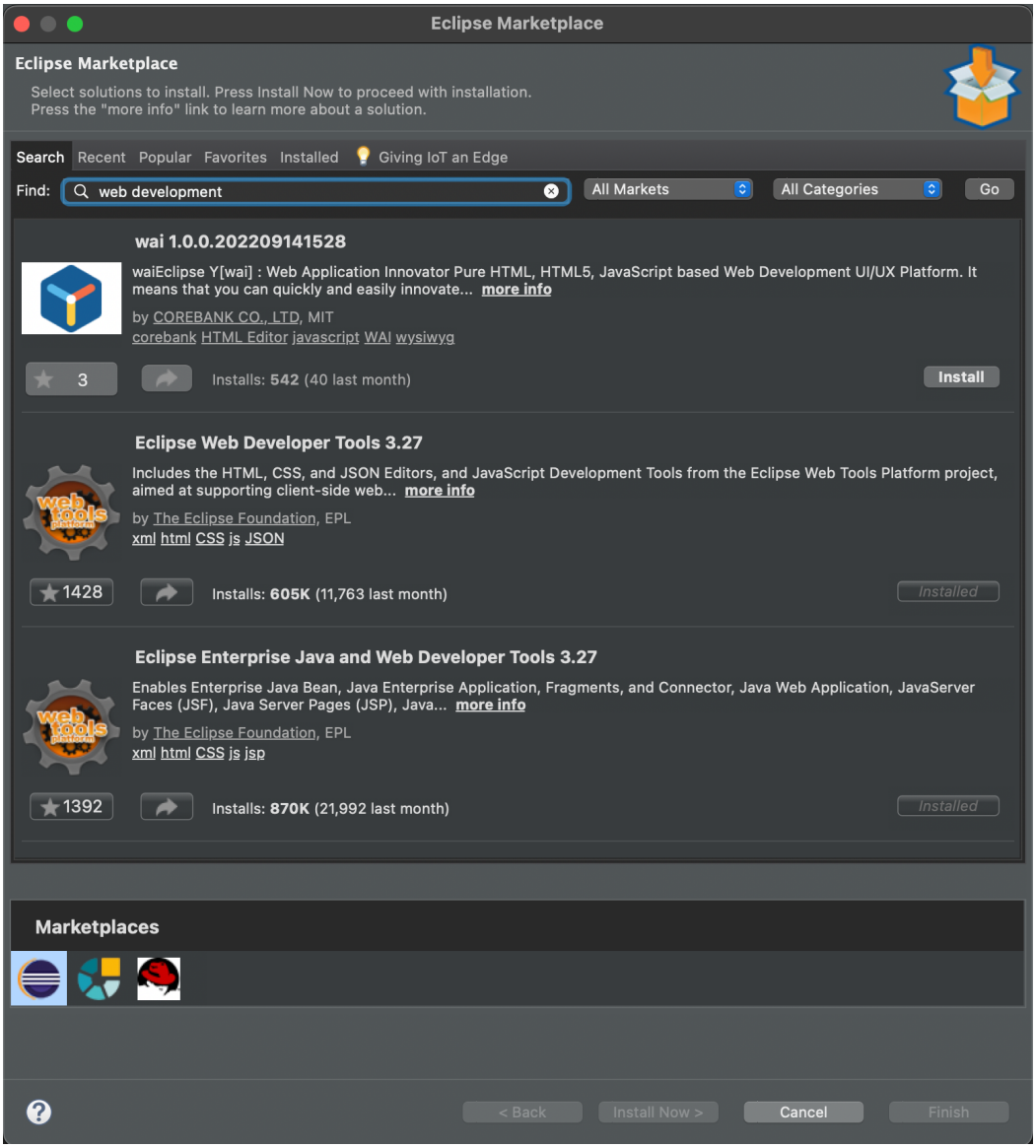


Spring Boot កំរិត Hello world

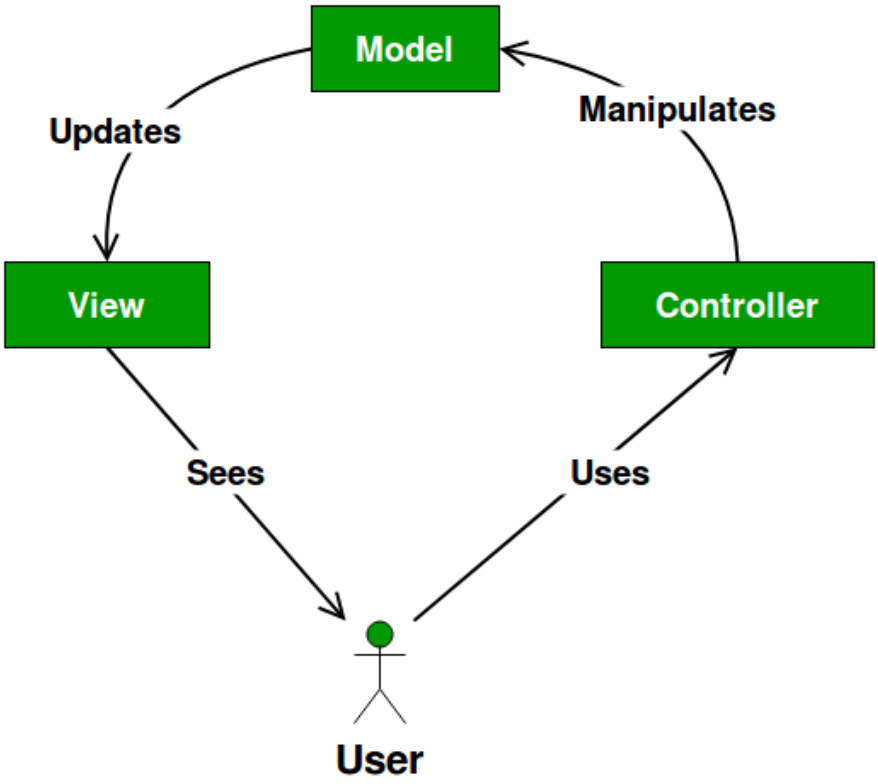
ដើម្បីសរសេរកាសា Web ដូចជា HTML, CSS, Bootstrap, JavaScript...

បានយើងត្រូវចូលទៅក្នុង Help ->Eclipse Marketplace បន្ទាប់មកទៀត

Search ពាក្យ Web Development រួចហើយ Install



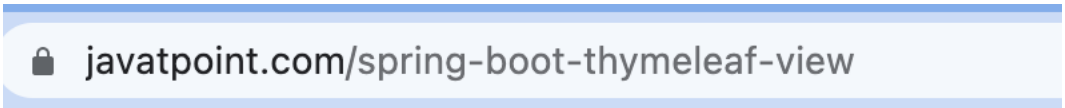
MVC = Model View Controller



សំគាល់៖

គ្មាន MVC

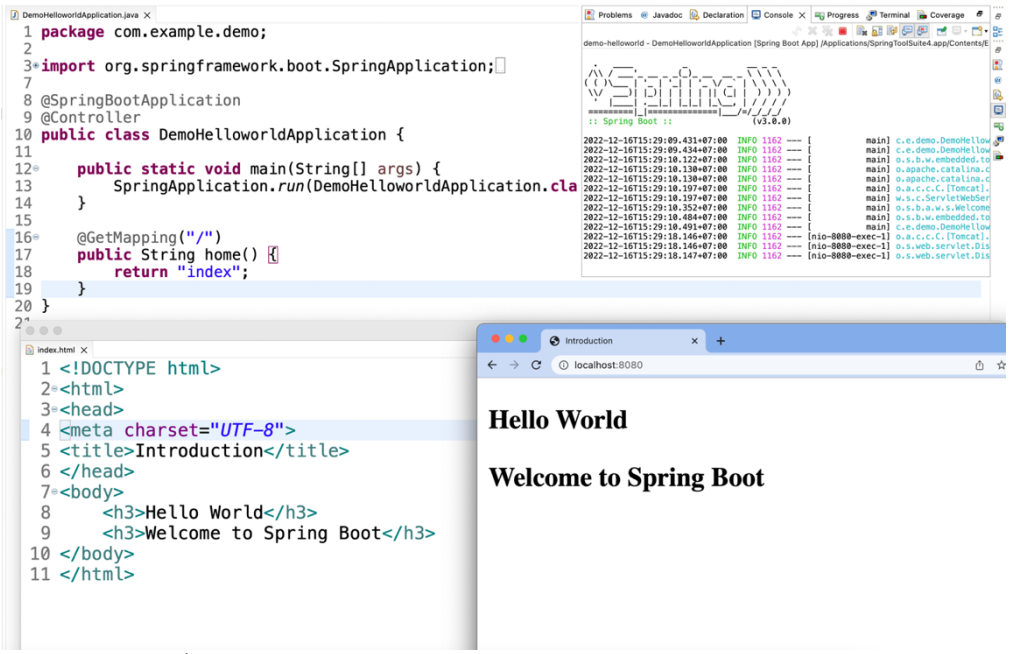
មាន MVC



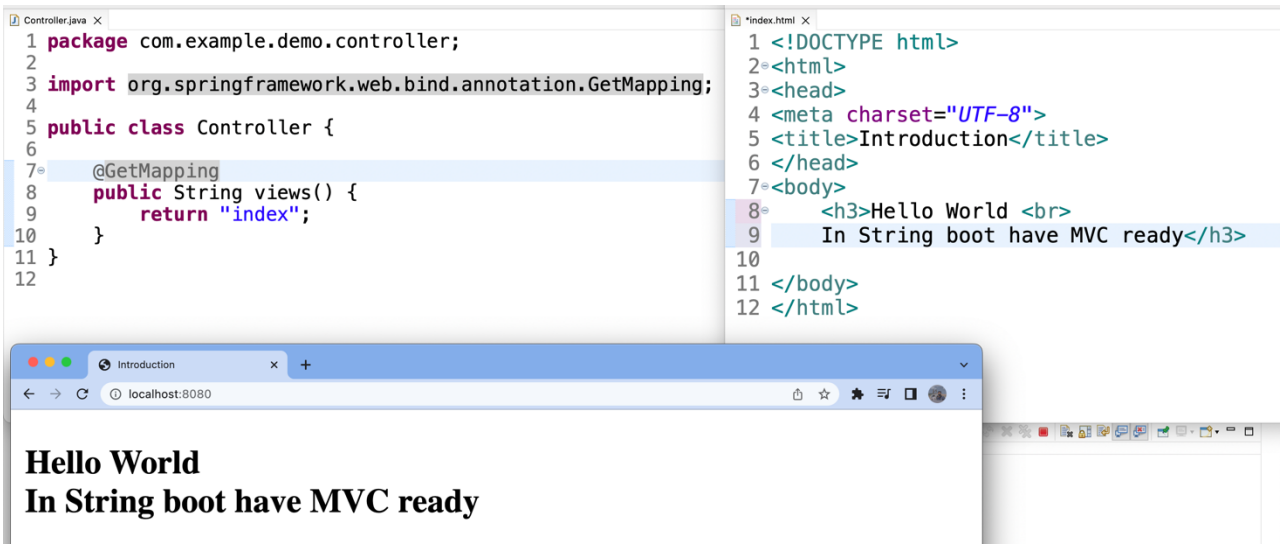
Thymeleaf Dependency

```xml
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-thymeleaf</artifactId>
</dependency>
```

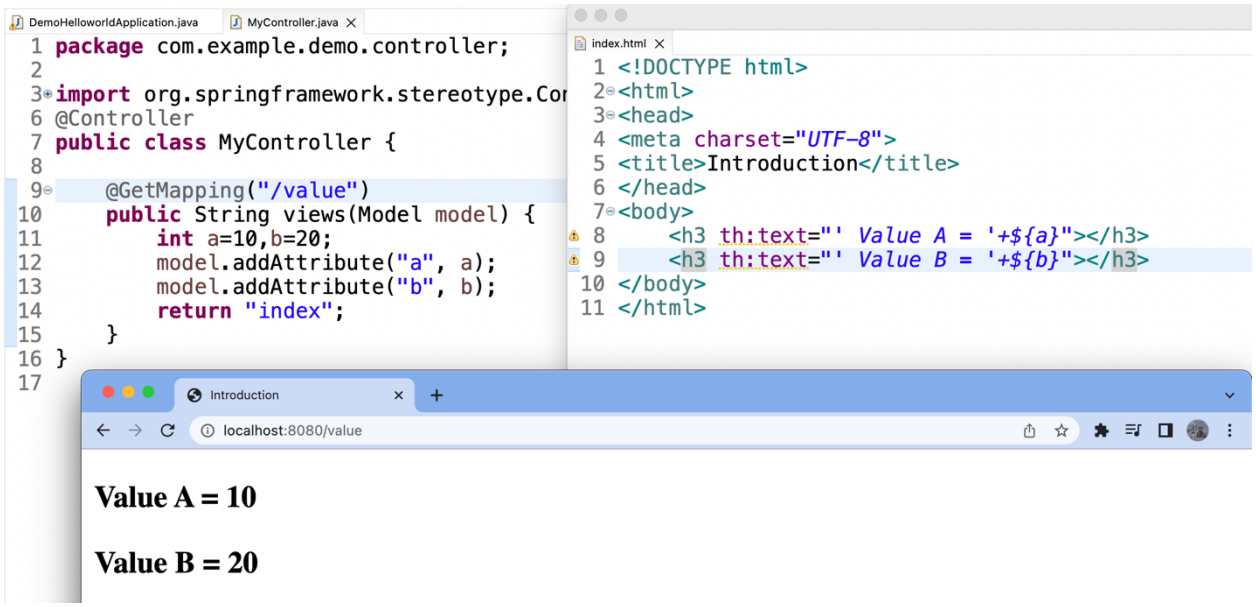
ឧទាហរណ៍ ១



ទោះគ្មាន Controller ក៏បានដែរព្រោះ Spring Boot មាន MVC ស្រាប់



ការបោះទិន្នន័យពី Backend to Frontend



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

import org.springframework.stereotype.Cor
@Controller
public class MyController {

    @GetMapping("/value")
    public String views(Model model) {
        int a=10,b=20;
        model.addAttribute("a", a);
        model.addAttribute("b", b);
        return "index";
    }
}
```

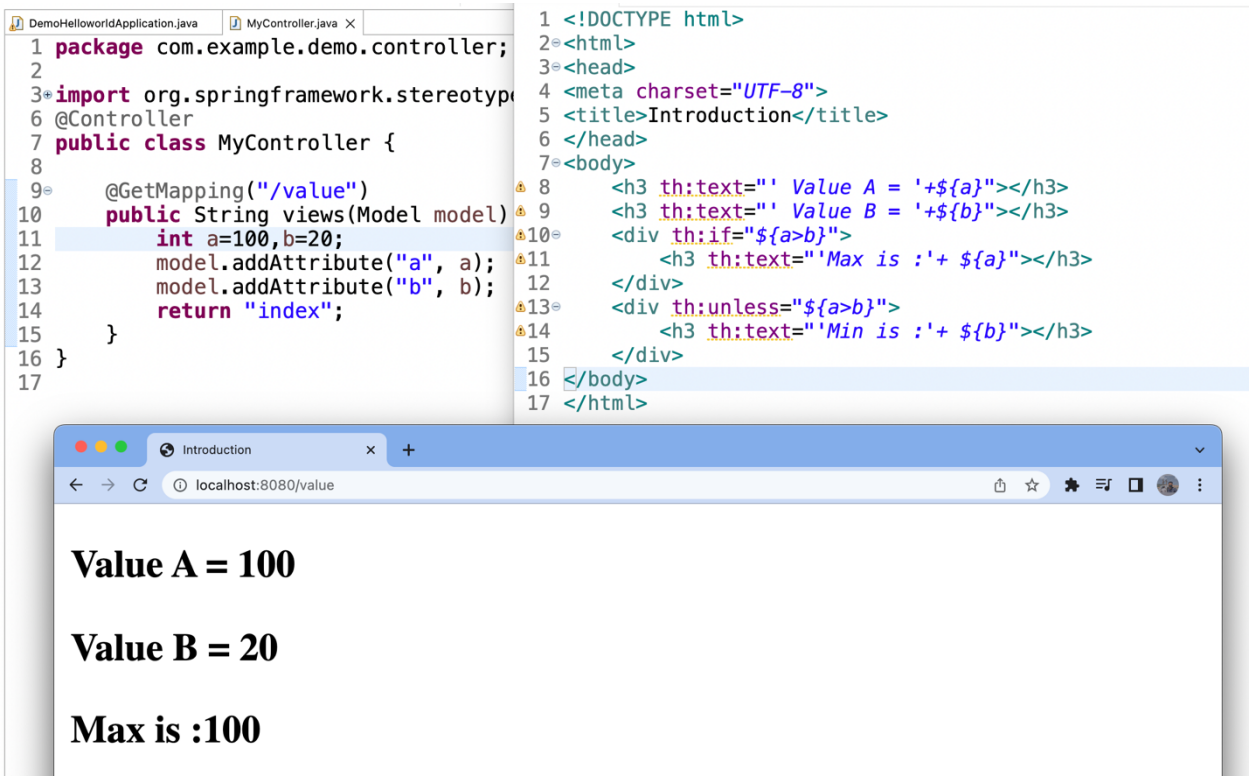
```html
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Introduction</title>
</head>
<body>
    <h3 th:text="' Value A = '+${a}"></h3>
    <h3 th:text="' Value B = '+${b}"></h3>
</body>
</html>
```

Introduction — localhost:8080/value

Value A = 10

Value B = 20

ការធែកលក្ខខណ្ឌនៅលើ Frontend



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

import org.springframework.stereotyp
@Controller
public class MyController {

    @GetMapping("/value")
    public String views(Model model)
        int a=100,b=20;
        model.addAttribute("a", a);
        model.addAttribute("b", b);
        return "index";
    }
}
```

```html
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Introduction</title>
</head>
<body>
    <h3 th:text="' Value A = '+${a}"></h3>
    <h3 th:text="' Value B = '+${b}"></h3>
    <div th:if="${a>b}">
        <h3 th:text="'Max is :'+ ${a}"></h3>
    </div>
    <div th:unless="${a>b}">
        <h3 th:text="'Min is :'+ ${b}"></h3>
    </div>
</body>
</html>
```

Introduction — localhost:8080/value

Value A = 100

Value B = 20

Max is :100

Class Model

```java
package com.example.demo.model;

public class MyModel {

    private String firstname;
    private String lastname;
    private String email;
    private String address;
    public MyModel() {
        // TODO Auto-generated constructor stub
    }
    public MyModel(String firstname, String lastname, String email, String address) {
        super();
        this.firstname = firstname;
        this.lastname = lastname;
        this.email = email;
        this.address = address;
    }
    public String getFirstname() {
        return firstname;
    }
    public void setFirstname(String firstname) {
        this.firstname = firstname;
    }
    public String getLastname() {
        return lastname;
    }
    public void setLastname(String lastname) {
        this.lastname = lastname;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getAddress() {
        return address;
    }
    public void setAddress(String address) {
        this.address = address;
    }

}
```

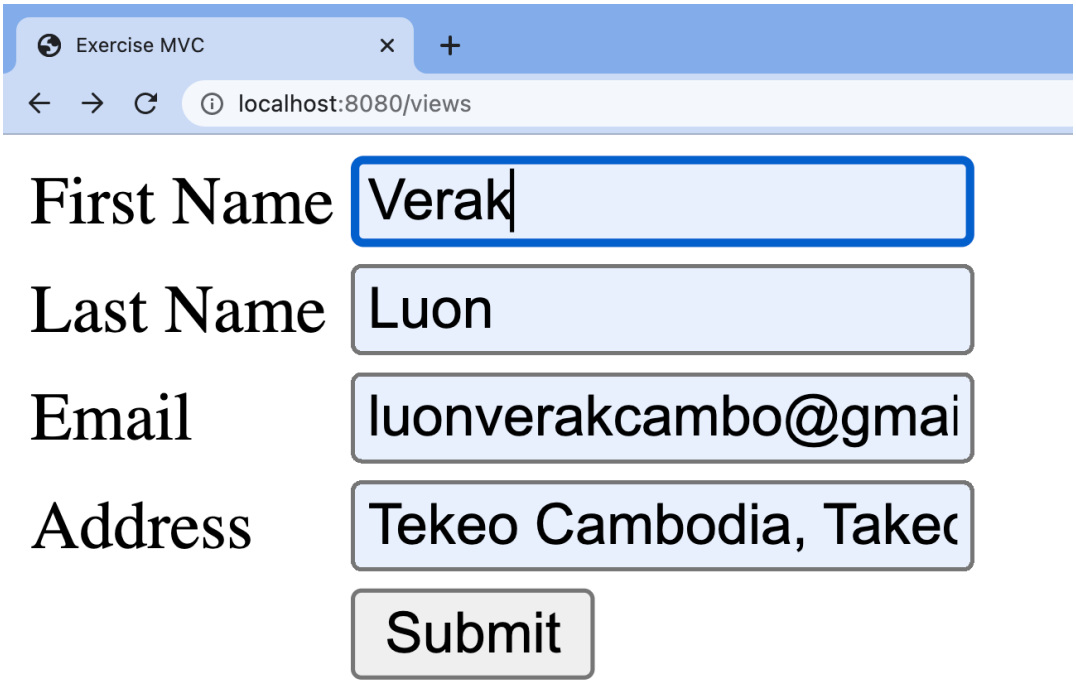
Class Controller

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

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.servlet.ModelAndView;

import com.example.demo.model.MyModel;

@Controller
public class MyController {

    @GetMapping("/views")
    public String home() {
        return "index";
    }

    @RequestMapping(value = "/save",method = RequestMethod.POST)
    public ModelAndView save(@ModelAttribute MyModel model)
    {
        ModelAndView modelAndView = new ModelAndView();
        modelAndView.setViewName("user-data");
        modelAndView.addObject("user", model);
        return modelAndView;
    }
}
```

Viesw

```html
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Exercise MVC</title>
</head>
<body>
    <form action="save" method="post">
        <table>
            <tr>
                <td><label for="firstname">First Name</label></td>
                <td><input type="text" name="firstname"></input></td>
             </tr>
             <tr>
                <td><label for="lastname">Last Name</label></td>
                <td><input type="text" name="lastname"></input></td>
             </tr>
            <tr>
                <td><label for="email">Email</label></td>
                <td><input type="text" name="email"></input></td>
            </tr>
            <tr>
                <td><label for="address">Address</label></td>
                <td><input type="text" name="address"></input></td>
            </tr>
            <tr>
                <td></td>
                <td><input type="submit" value="Submit"></input></td>
            </tr>
        </table>
    </form>
</body>
</html>
```

Result



```
<html xmlns:th="https://thymeleaf.org">
    <table>
        <tr>
            <td><h4>Full Name: </h4></td>
            <td><h4 th:text="${user.firstname} +' '+ ${user.lastname}"></h4></td>
        </tr>
        <tr>
            <td><h4>Email ID: </h4></td>
            <td><h4 th:text="${user.email}"></h4></td>
        </tr>
        <tr>
            <td><h4>Address: </h4></td>
            <td><h4 th:text="${user.address}"></h4></td>
        </tr>
    </table>
</html>
```

Result

Full Name: Verak Luon

E-mail : luonverakcambo@gmail.com

Address : Tekeo Cambodia

Crud Operation with MVC

Project One

Backend

```java
package com.example.demo;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import com.example.demo.repository.StudentRepository;

@SpringBootApplication
public class StudentApplication implements CommandLineRunner {

    public static void main(String[] args) {

        SpringApplication.run(StudentApplication.class, args);
    }

    @Autowired
    private StudentRepository studentRepository;

    @Override
    public void run(String... args) throws Exception {

//      Student stu1=new Student("Luon", "Verak", "luonverak@gmail.com");
//      studentRepository.save(stu1);
//
//      Student stu2=new Student("Sok", "Cheata", "sokcheata@gmail.com");
//      studentRepository.save(stu2);
//
//      Student stu3=new Student("Seng", "Lyna", "senglyna@gmail.com");
//      studentRepository.save(stu3);
//
    }

}
```

Class Controller

```java
import org.springframework.stereotype.Controller;
@Controller
public class StudentController {

    private StudentService studentService;

    public StudentController(StudentService studentService) {
        super();
        this.studentService = studentService;
    }
    @GetMapping("/students")
    public String listStudents(Model mod) {
        mod.addAttribute("students",studentService.getAllStudents());
        return "students";
    }
    @GetMapping("/students/new")
    public String createStudentForm(Model mod) {

        Student student=new Student();
        mod.addAttribute("student", student);
        return "create_student";
    }
    @PostMapping("/students")
    public String saveStudent(@ModelAttribute ("student") Student student) {
        studentService.saveStudent(student);
        return "redirect:/students";
    }
    @GetMapping("/students/edit/{id}")
    public String editStudentForm(@PathVariable Long id,Model mod) {
        mod.addAttribute("student", studentService.getStudentById(id));
        return "edit_student";
    }
    @PostMapping("/students/{id}")
    public String updateStudent(@PathVariable Long id,@ModelAttribute("student") Student student,Model mod) {
        Student existingStudent = studentService.getStudentById(id);
        existingStudent.setId(id);
        existingStudent.setFirstname(student.getFirstname());
        existingStudent.setLastname(student.getLastname());
        existingStudent.setEmail(student.getEmail());
        existingStudent.setScore(student.getScore());
        studentService.updateStudent(existingStudent);
        return "redirect:/students";
    }
    @GetMapping("/students/{id}")
    public String deleteStudent(@PathVariable Long id) {
        studentService.deleteStudentById(id);
        return "redirect:/students";
    }
}
```

```java
package com.example.demo.entity;
import jakarta.persistence.Column;
@NoArgsConstructor
@AllArgsConstructor
@Getter
@Setter
@Entity
@Table(name="students")
public class Student {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private long id;
    @Column(name = "first_name",nullable = false)
    private String firstname;
    @Column(name = "last_name",nullable = false)
    private String lastname;
    @Column(name = "email",nullable = false)
    private String email;
    @Column(name="score",nullable = false)
    private float score;
```

```java
package com.example.demo.repository;

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

import com.example.demo.entity.Student;

public interface StudentRepository extends JpaRepository<Student, Long>{

}
```

StudentService.java

```java
package com.example.demo.service;

import java.util.List;

import com.example.demo.entity.Student;

public interface StudentService {
    List<Student> getAllStudents();
    Student saveStudent(Student student);
    Student getStudentById(Long id);
    Student updateStudent(Student student);
    void deleteStudentById(Long id);
}
```

```java
package com.example.demo.service.impl;

import java.util.List;

import org.springframework.stereotype.Service;

import com.example.demo.entity.Student;
import com.example.demo.repository.StudentRepository;
import com.example.demo.service.StudentService;
@Service
public class StudentImp implements StudentService{

    private StudentRepository studentRepository;

    public StudentImp(StudentRepository studentRepository) {
        super();
        this.studentRepository = studentRepository;
    }

    @Override
    public List<Student> getAllStudents() {

        return studentRepository.findAll();
    }

    @Override
    public Student saveStudent(Student student) {

        return studentRepository.save(student);
    }

    @Override
    public Student getStudentById(Long id) {
        // TODO Auto-generated method stub
        return studentRepository.findById(id).get();
    }

    @Override
    public Student updateStudent(Student student) {
        // TODO Auto-generated method stub
        return studentRepository.save(student);
    }

    @Override
    public void deleteStudentById(Long id) {
        studentRepository.deleteById(id);

    }


}
```

Student GUI

https://getbootstrap.com/docs/4.0/getting-started/introduction/

https://www.w3schools.com/bootstrap4/bootstrap_navbar.asp

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="UTF-8">
<title>Student Management</title>
<link rel="stylesheet"
    href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css"
    integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
    crossorigin="anonymous">
</head>
<body>

    <nav class="navbar navbar-expand-md bg-dark navbar-dark">
        <!-- Brand -->
        <a class="navbar-brand" th:href="@{/students}">Students Management</a>

        <!-- Toggler/collapsibe Button -->
        <button class="navbar-toggler" type="button" data-toggle="collapse"
            data-target="#collapsibleNavbar">
            <span class="navbar-toggler-icon"></span>
        </button>

        <!-- Navbar links -->
        <div class="collapse navbar-collapse" id="collapsibleNavbar">
            <ul class="navbar-nav">
                <li class="nav-item"><a class="nav-link" th:href="@{/students}">Students
                        Management</a></li>
            </ul>
        </div>
    </nav>

    <div class="container">
        <div class="row">
            <h1>List of Students</h1>
        </div>

        <div class="row">
            <div class="col-lg-3">
                <a th:href="@{/students/new}" class="btn btn-primary btn-sm mb-3">
                    Add Student</a>
            </div>
        </div>

        <table class="table table-striped table-bordered">
            <thead class="table-dark">
                <tr>
```

CTYPE:html Writable

```
                <li class="nav-item"><a class="nav-link" th:href="@{/students}">Students
                        Management</a></li>
            </ul>
        </div>
    </nav>

    <div class="container">
        <div class="row">
            <h1>List of Students</h1>
        </div>

        <div class="row">
            <div class="col-lg-3">
                <a th:href="@{/students/new}" class="btn btn-primary btn-sm mb-3">
                    Add Student</a>
            </div>
        </div>

        <table class="table table-striped table-bordered">
            <thead class="table-dark">
                <tr>
                    <th>First Name</th>
                    <th>Last Name</th>
                    <th>Email</th>
                    <th>Score</th>
                    <th>Actions</th>
                </tr>
            </thead>
            <tbody>
                <tr th:each="student: ${students}">
                    <td th:text="${student.firstname}"></td>
                    <td th:text="${student.lastname}"></td>
                    <td th:text="${student.email}"></td>
                    <td th:text="${student.score}"></td>

                    <td><a th:href="@{/students/edit/{id}(id=${student.id})}"
                        class="btn btn-primary">Update</a> <a
                        th:href="@{/students/{id}(id=${student.id})}"
                        class="btn btn-danger">Delete</a></td>
                </tr>
            </tbody>
        </table>
    </div>

</body>
</html>
```

Create New Student

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="UTF-8">
<title>Student Management</title>
<link rel="stylesheet"
    href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css"
    integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
    crossorigin="anonymous">
</head>
<body>

    <nav class="navbar navbar-expand-md bg-dark navbar-dark">
        <!-- Brand -->
        <a class="navbar-brand" th:href="@{/students}">Students Management</a>

        <!-- Toggler/collapsibe Button -->
        <button class="navbar-toggler" type="button" data-toggle="collapse"
            data-target="#collapsibleNavbar">
            <span class="navbar-toggler-icon"></span>
        </button>

        <!-- Navbar links -->
        <div class="collapse navbar-collapse" id="collapsibleNavbar">
            <ul class="navbar-nav">
                <li class="nav-item"><a class="nav-link" th:href="@{/students}">Students
                        Management</a></li>
            </ul>
        </div>
    </nav>
    <br>
    <br>
    <div class="container">
        <div class="row">
            <div
                class="col-lg-6 col-md col-sm-6 container justify-content-center card">
                <div class="text-center">
                    <h1>Create New Students</h1>
                    <div class="card-body">
                        <form th:action="@{/students}" th:object="${student}"
                            method="POST">
                            <div class="form-group">
                                <label>Student First Name</label> <input type="text"
                                    name="firstname" th:field="*{firstname}" class="form-control"
                                    placeholder="Enter Student First Name" />
                            </div>
                            <div class="form-group">
                                <label>Student Last Name</label> <input type="text"
                                    name="lastname" th:field="*{lastname}" class="form-control"
                                    placeholder="Enter Student Last Name" />
                            </div>
                            <div class="form-group">
                                <label>Student Email</label> <input type="text" name="email"
                                    th:field="*{email}" class="form-control"
                                    placeholder="Enter Student Email" />
                            </div>
                            <div class="form-group">
                                <label>Student Score</label> <input type="text" name="score"
                                    th:field="*{score}" class="form-control"
                                    placeholder="Enter Student Score" />
                            </div>
                            <div class="box-footer">
                                <button type="submit" class="btn btn-primary">Submit</button>
                            </div>
                        </form>
                    </div>
                </div>
            </div>
        </div>
    </div>
</body>
</html>
```

Update Data Student

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
 <meta charset="UTF-8">
 <title>Student Management</title>
 <link rel="stylesheet"
     href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css"
     integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
     crossorigin="anonymous">
</head>
<body>

    <nav class="navbar navbar-expand-md bg-dark navbar-dark">
        <!-- Brand -->
        <a class="navbar-brand" th:href="@{/students}">Students Management</a>

        <!-- Toggler/collapsibe Button -->
        <button class="navbar-toggler" type="button" data-toggle="collapse"
            data-target="#collapsibleNavbar">
            <span class="navbar-toggler-icon"></span>
        </button>

        <!-- Navbar links -->
        <div class="collapse navbar-collapse" id="collapsibleNavbar">
            <ul class="navbar-nav">
                <li class="nav-item"><a class="nav-link" th:href="@{/students}">Students
                        Management</a></li>
            </ul>
        </div>
    </nav>
    <br>
    <br>
    <div class="container">
        <div class="row">
            <div
                class="col-lg-6 col-md col-sm-6 container justify-content-center card">
                <div class="text-center">
                    <h1>Update Students</h1>
                    <div class="card-body">
                        <form th:action="@{/students (id=${student.id})}"
                            th:object="${student}" method="POST">
                            <div class="form-group">
                                <label>Student First Name</label> <input type="text"
                                    name="firstname" th:field="*{firstname}" class="form-control"
                                    placeholder="Enter Student First Name" />
                            </div>
                            <div class="form-group">
                                <label>Student Last Name</label> <input type="text"
                                    name="lastname" th:field="*{lastname}" class="form-control"
                                    placeholder="Enter Student Last Name" />
                            </div>
                            <div class="form-group">
                                <label>Student Email</label> <input type="text" name="email"
                                    th:field="*{email}" class="form-control"
                                    placeholder="Enter Student Email" />
                            </div>
                            <div class="form-group">
                                <label>Student Score</label> <input type="text" name="score"
                                    th:field="*{score}" class="form-control"
                                    placeholder="Enter Student Score" />
                            </div>
                            <div class="box-footer">
                                <button type="submit" class="btn btn-primary">Submit</button>
                            </div>
                        </form>
                    </div>
                </div>
            </div>
        </div>
    </div>
</body>
</html>
```

Databases with MySQL

Create and Connection

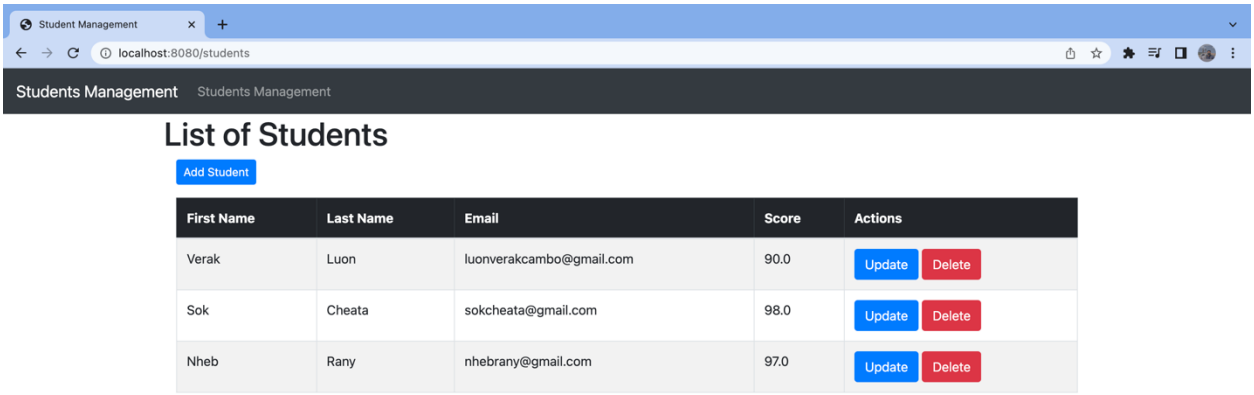
```
spring.datasource.url=jdbc:mysql://localhost:3306/dbstudents?createDatabaseIfNotExist=true&allowPublicKeyRetrieval=true&useSSL=false
spring.datasource.username=root
spring.datasource.password=

spring.datasource.hikari.connection-timeout=20000
spring.datasource.hikari.minimum-idle=5
spring.datasource.hikari.maximum-pool-size=100
spring.datasource.hikari.idle-timeout=300000
spring.datasource.hikari.max-lifetime=1200000
spring.datasource.hikari.auto-commit=true

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect
spring.jpa.properties.hibernate.id.new_generator_mappings=false
spring.jpa.properties.hibernate.format_sql=true

spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true

spring.jpa.open-in-view=true

spring.servlet.multipart.max-file-size=15MB
spring.servlet.multipart.max-request-size=15MB

```

Table

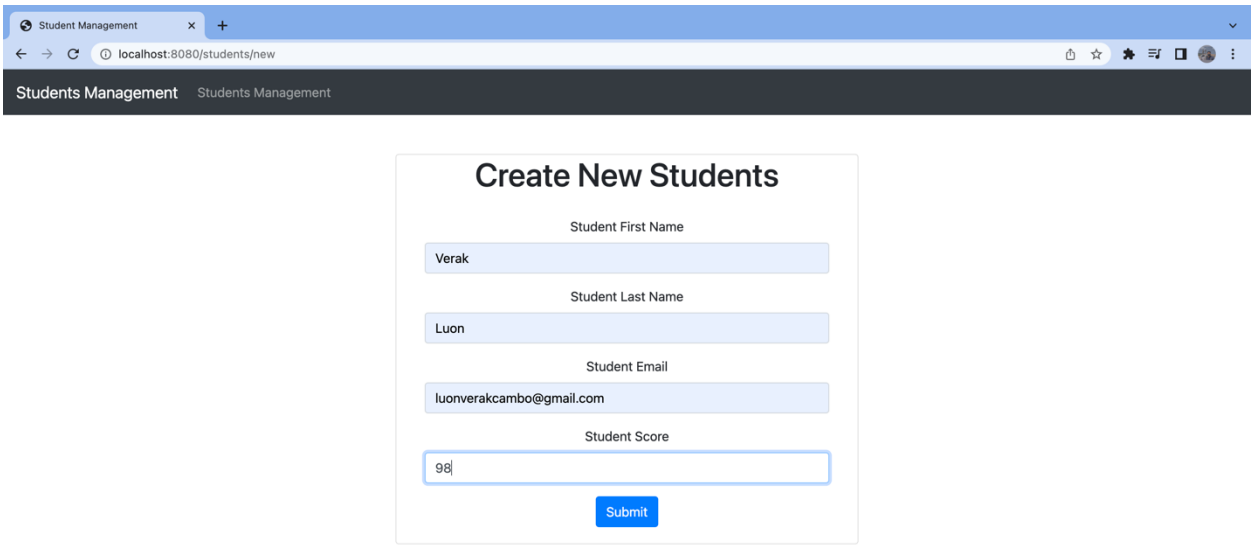
	id	email	first_name	last_name	score
Edit Copy Delete	2	luonverakcambo@gmail.com	Verak	Luon	90
Edit Copy Delete	3	sokcheata@gmail.com	Sok	Cheata	98
Edit Copy Delete	4	nhebrany@gmail.com	Nheb	Rany	97

Check all *With selected:* Edit Copy Delete Export

Result



When Click Button Add or Update



API = Application Programming Interface

JSON = JavaScript Object Notation

```java
package com.example.demo.model;

import jakarta.persistence.Entity;

@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor
@Entity
@Table(name="tblpeople")
public class People {

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

}
```

```java
package com.example.demo.service;

import java.util.List;

@Service
public class PeopleService {

    @Autowired
    private PeopleRepo peopleRepo;

    public List<People> findAllpeople()
    {
        return peopleRepo.findAll();
    }

    public People findByID(long id) {

        return peopleRepo.findById(id).orElse(new People());
    }

    public List<People> findByAddress(String address) {

        return peopleRepo.findByAddress(address);
    }

    public void save(People people) {
        peopleRepo.save(people);
    }

    public void update(People people) {
        People pl=peopleRepo.findById(people.getId()).orElse(new People());
        if(pl.getId()==0)
            pl.setName("No People");
        else
            peopleRepo.save(people);
    }

    public void delete(People people) {
        peopleRepo.deleteById(people.getId());
    }
}
```

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

import java.util.List;

@RestController
@RequestMapping("/api/people")
public class MyController {

    @Autowired
    private PeopleService people;

    @GetMapping("")
    public List<People> listAllpeople()
    {
        return people.findAllpeople();
    }

    @GetMapping("/id/{id}")
    public People findByID(@PathVariable long id) {

        return people.findByID(id);
    }
    @GetMapping("/address/{address}")
    public List<People> findByaddress(@PathVariable String address){

        return people.findByAddress(address);
    }

}
```

MyController2.java

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

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

@RestController
@RequestMapping("/api/people")
public class MyController2 {

    @Autowired
    private PeopleService peopleService;

    @PostMapping("")
    public People add(@RequestBody People people) {
        peopleService.save(people);
        return people;
    }
    @PutMapping("")
    public People update(@RequestBody People people) {
        peopleService.update(people);
        return people;
    }
    @DeleteMapping("")
    public People delete(@RequestBody People people) {
        peopleService.delete(people);
        return people;
    }
}
```

```java
package com.example.demo.repo;

import java.util.List;

public interface PeopleRepo extends JpaRepository<People, Long>{

    List<People> findByAddress(String address);

}
```

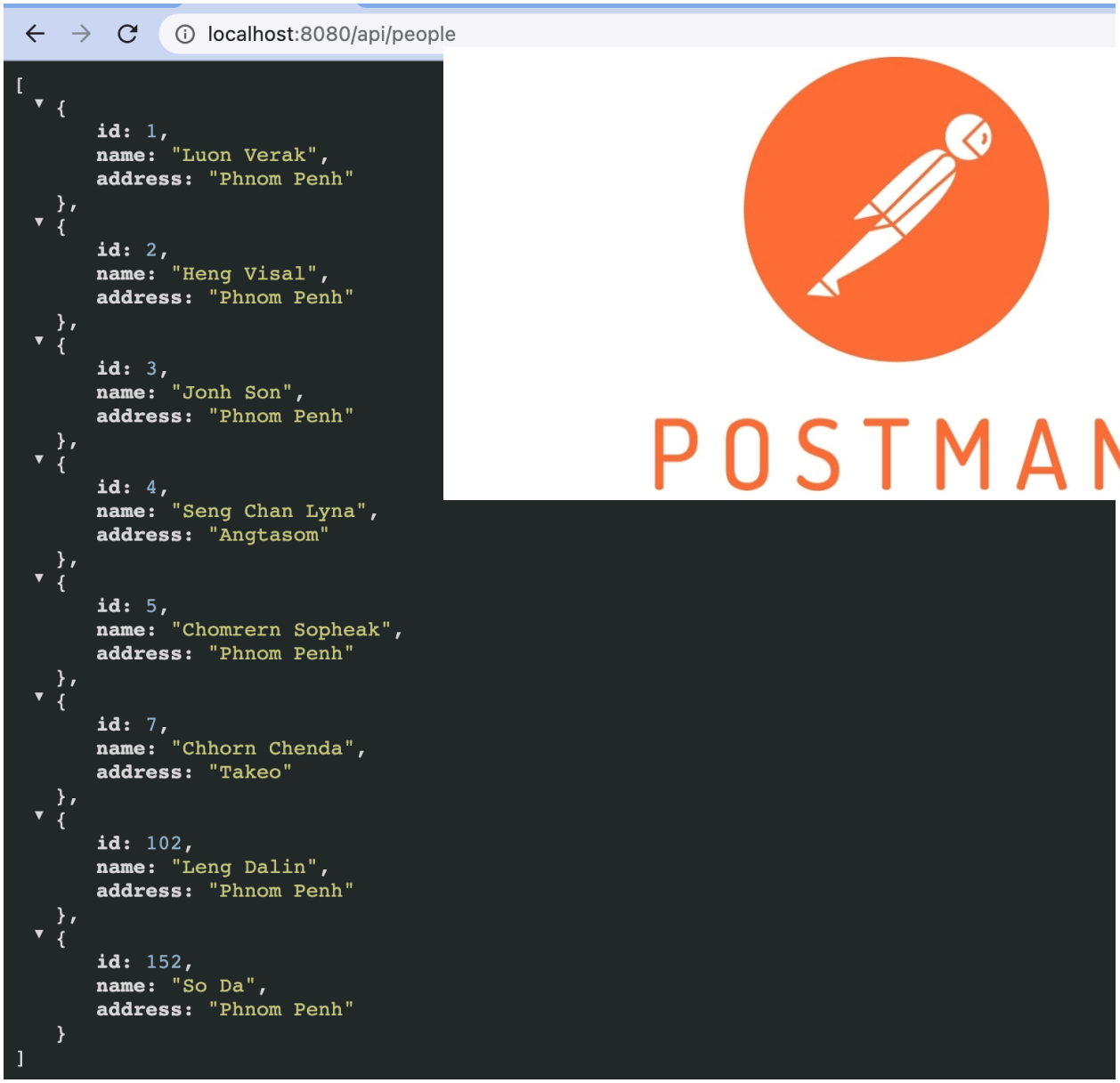
application.properties

```
spring.datasource.url=jdbc:mysql://localhost:3306/springapi?createDatabaseIfNotExist=true&allowPublicKeyRetrieval=true&useSSL=false
spring.datasource.username=root
spring.datasource.password=

spring.datasource.hikari.connection-timeout=20000
spring.datasource.hikari.minimum-idle=5
spring.datasource.hikari.maximum-pool-size=100
spring.datasource.hikari.idle-timeout=300000
spring.datasource.hikari.max-lifetime=1200000
spring.datasource.hikari.auto-commit=true

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect
spring.jpa.properties.hibernate.id.new_generator_mappings=false
spring.jpa.properties.hibernate.format_sql=true

spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true

spring.jpa.open-in-view=true
```

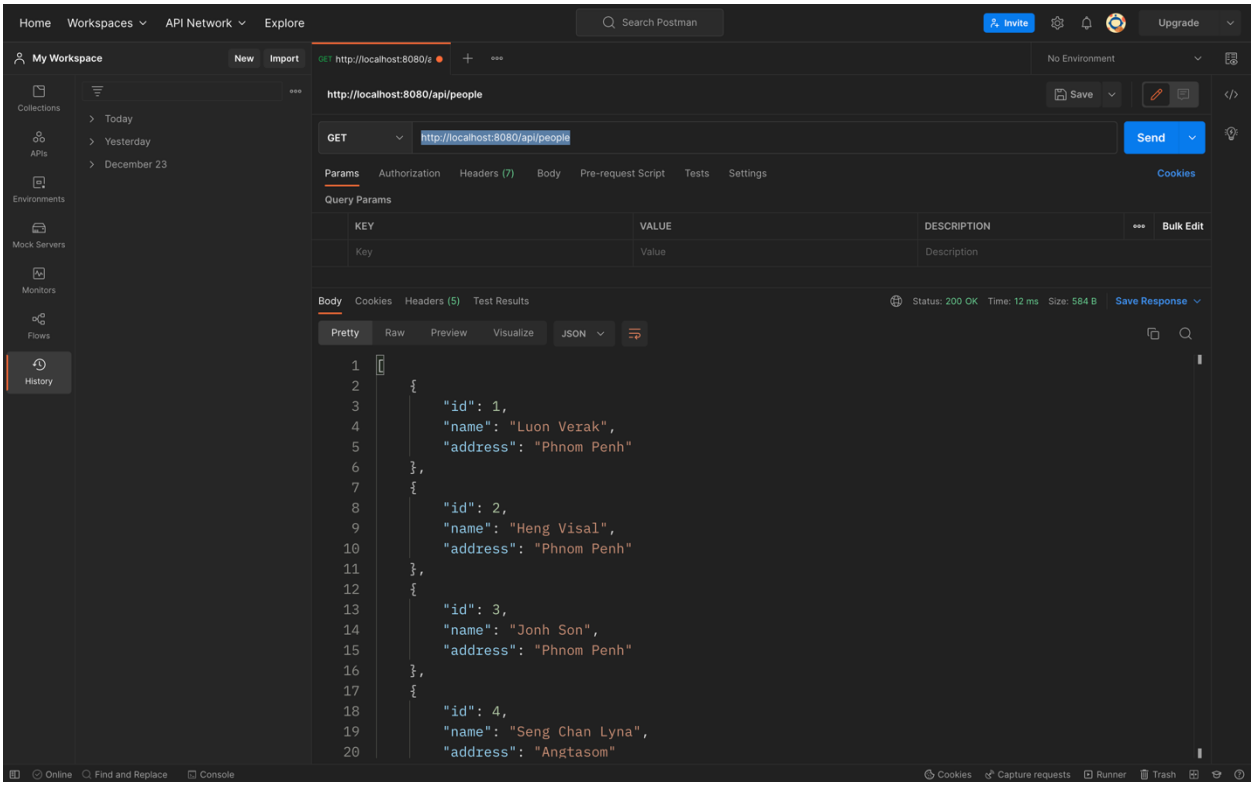
	id	address	name
Edit Copy Delete	1	Phnom Penh	Luon Verak
Edit Copy Delete	2	Phnom Penh	Heng Visal
Edit Copy Delete	3	Phnom Penh	Jonh Son
Edit Copy Delete	4	Angtasom	Seng Chan Lyna
Edit Copy Delete	5	Phnom Penh	Chomrern Sopheak
Edit Copy Delete	7	Takeo	Chhorn Chenda
Edit Copy Delete	102	Phnom Penh	Leng Dalin
Edit Copy Delete	152	Phnom Penh	So Da

We need Install Postman for testing API

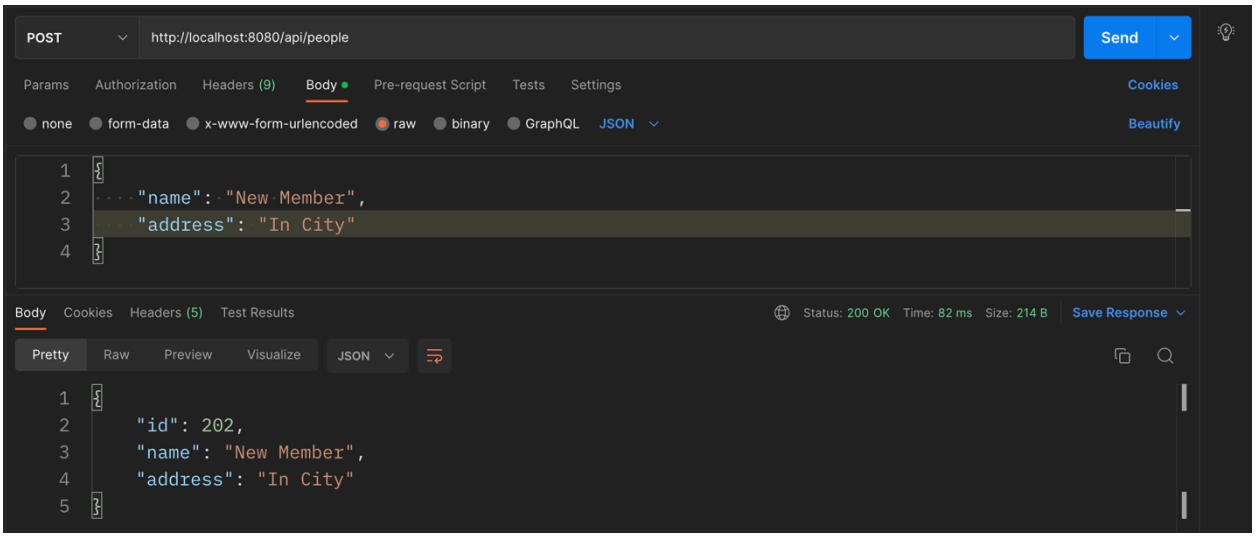
https://www.postman.com/downloads/



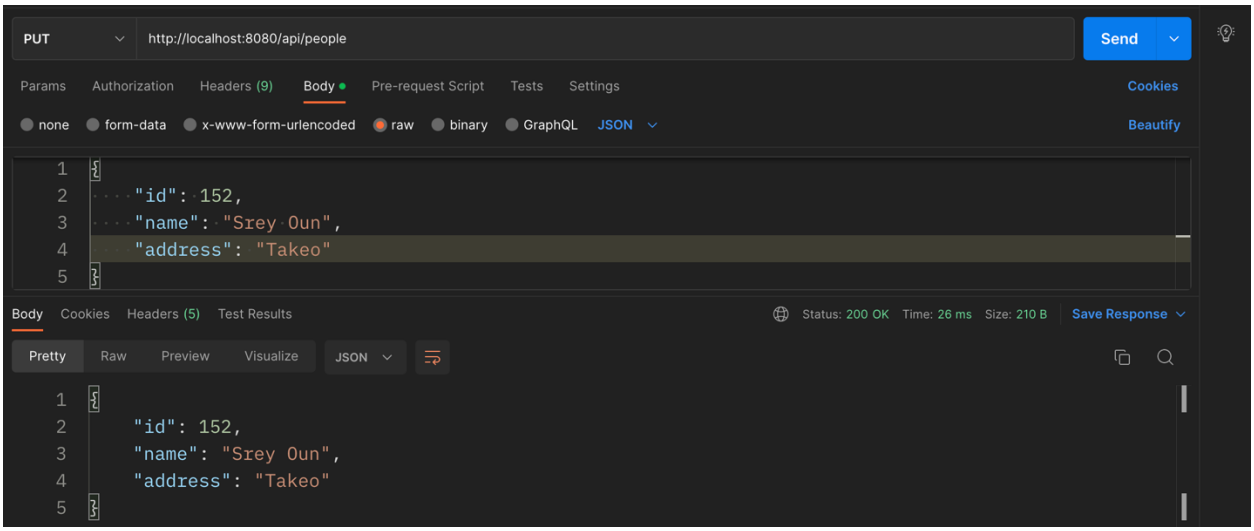
Get API in Postman *សំដៅលើការទាញទិន្នន័យតាមរយៈ* Endpoint people



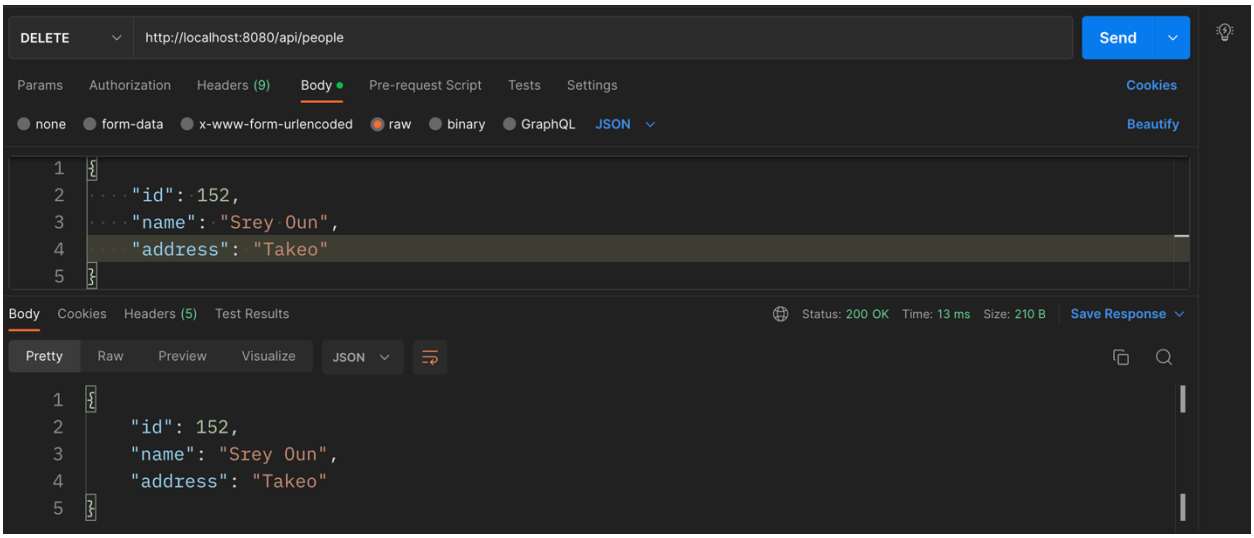
Post API in Postman *សំដៅលើការបញ្ចូលទិន្នន័យតាមរយៈ* Endpoint people



Put API in Postman *សំដៅលើការកែប្រែទិន្នន័យតាមរយៈ* Endpoint people



Delete API in Postman *សំដៅលើការលុបទិន្នន័យតាមរយៈ* Endpoint people



Restful Spring boot + React Js

Backend

```java
package com.example.demo;

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

@SpringBootApplication
public class Project1Application implements CommandLineRunner{

	public static void main(String[] args) {
		SpringApplication.run(Project1Application.class, args);
	}

	@Autowired
	private EmployeeRepository employeeRepository;
	@Override
	public void run(String... args) throws Exception {
//		Employee employee=new Employee();
//		employee.setFirstName("Verak");
//		employee.setLastName("Luon");
//		employee.setEmail("luonverak@gmail.com");
//
//		employeeRepository.save(employee);
	}

}
```

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

import java.util.List;

@CrossOrigin("http://localhost:3000/")
@RestController
@RequestMapping("/api/employee")
public class EmployeeController {

    @Autowired
    private EmployeeRepository employeeRepository;

    @GetMapping
    public List<Employee> getAllEmployee(){
        return  employeeRepository.findAll();
    }
    @PostMapping("")
    public Employee createEmployee(@RequestBody Employee employee) {
        return employeeRepository.save(employee);
    }
    //find by id
    @GetMapping("{id}")
    public ResponseEntity<Employee> getEmployeeByID(@PathVariable long id){
        Employee employee=employeeRepository.findById(id)
                .orElseThrow( () -> new ResurceNotFoundException("Employee not exist with id :"+id));
        return ResponseEntity.ok(employee);
    }
    @PutMapping("{id}")
    public ResponseEntity<Employee> updateEmployeeByID(@PathVariable long id,@RequestBody Employee employeeDetail){
        Employee updaetEmployee=employeeRepository.findById(id)
                .orElseThrow( () -> new ResurceNotFoundException("Employee not exist with id :"+id));

        updaetEmployee.setFirstName(employeeDetail.getFirstName());
        updaetEmployee.setLastName(employeeDetail.getLastName());
        updaetEmployee.setEmail(employeeDetail.getEmail());

        employeeRepository.save(updaetEmployee);
        return ResponseEntity.ok(updaetEmployee);
    }
    @DeleteMapping("{id}")
    public ResponseEntity<HttpStatus> deleteEmployeeByID(@PathVariable long id){
        Employee employee=employeeRepository.findById(id)
                .orElseThrow( () -> new ResurceNotFoundException("Employee not exist with id :"+id));
        employeeRepository.delete(employee);
        return new ResponseEntity<>(HttpStatus.NO_CONTENT);

    }
}
```

```java
package com.example.demo.exception;

import org.springframework.http.HttpStatus;

@ResponseStatus(value = HttpStatus.NOT_FOUND)
public class ResurceNotFoundException extends RuntimeException{

    public ResurceNotFoundException(String message) {
        super(message);
    }
}
```

EmployeeRepository.java

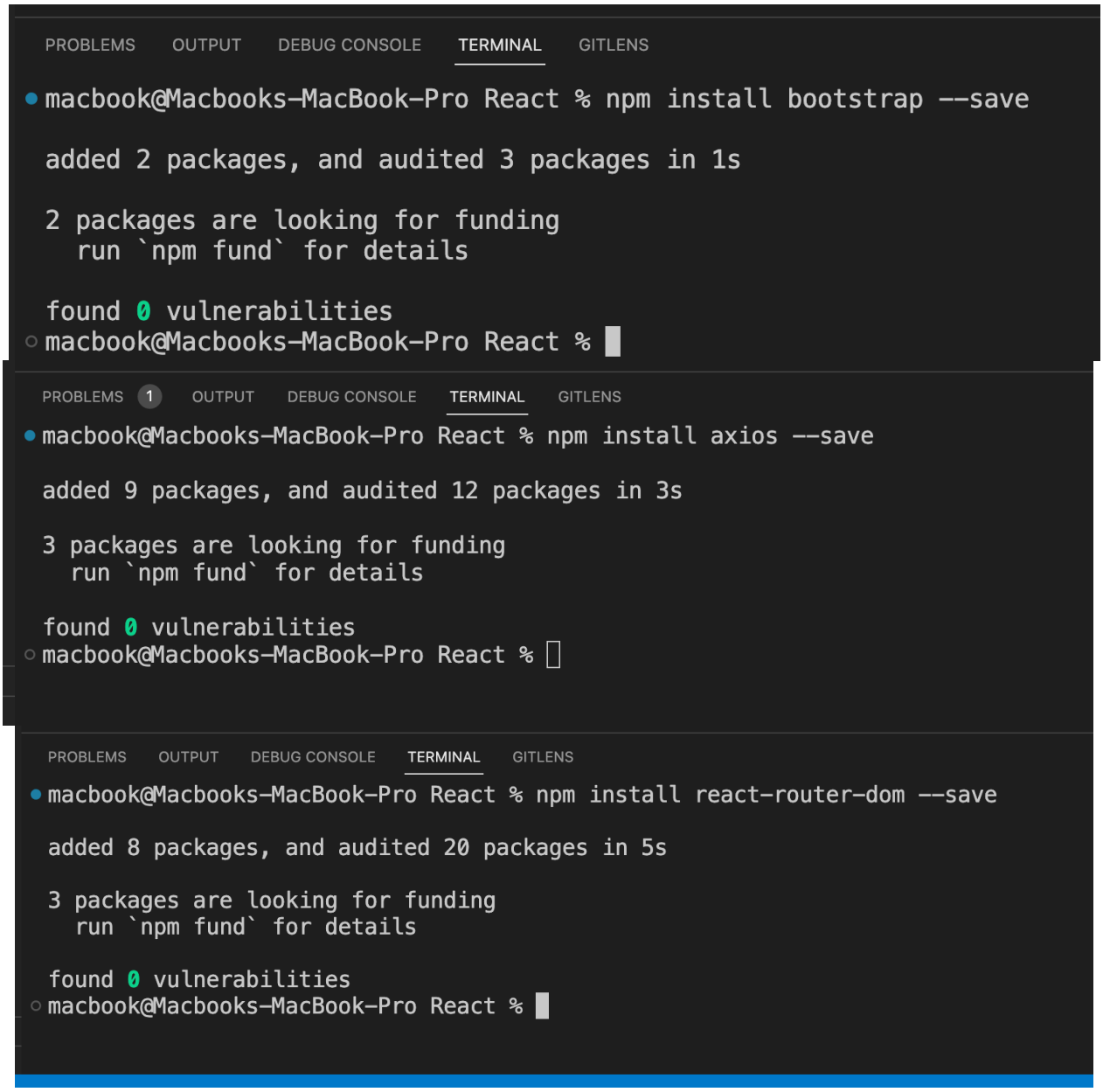
```java
package com.example.demo.repository;

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

public interface EmployeeRepository extends JpaRepository<Employee, Long>{

}
```

```java
package com.example.demo.model;

import jakarta.persistence.Column;

@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor
@Entity
@Table(name = "tbemployee")
public class Employee {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private long id;
    @Column(name = "first_name")
    private String firstName;
    @Column(name = "last_name")
    private String lastName;
    @Column(name = "email")
    private String email;
}
```

Frontend React Js

Command for install project

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  GITLENS

macbook@Macbooks-MacBook-Pro React % node -v
v18.12.1
macbook@Macbooks-MacBook-Pro React % npm -v
8.19.2
macbook@Macbooks-MacBook-Pro React % npx create-react-app react-hook-frontend

Creating a new React app in /Users/macbook/Desktop/React/react-hook-frontend.
```


PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL GITLENS
macbook@Macbooks-MacBook-Pro React % npm install bootstrap --save
added 2 packages, and audited 3 packages in 1s
2 packages are looking for funding
run `npm fund` for details
found 0 vulnerabilities
macbook@Macbooks-MacBook-Pro React %
PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL GITLENS
macbook@Macbooks-MacBook-Pro React % npm install axios --save
added 9 packages, and audited 12 packages in 3s
3 packages are looking for funding
run `npm fund` for details
found 0 vulnerabilities
macbook@Macbooks-MacBook-Pro React %
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL GITLENS
macbook@Macbooks-MacBook-Pro React % npm install react-router-dom --save
added 8 packages, and audited 20 packages in 5s
3 packages are looking for funding
run `npm fund` for details
found 0 vulnerabilities
macbook@Macbooks-MacBook-Pro React %

App.js M

react-hook-frontend > src > App.js > App

You, 2 seconds ago | 1 author (You)

```
import './App.css';
import { BrowserRouter , Route,Routes} from 'react-router-dom';
import Footer from './components/Footer';
import HeaderComponents from './components/HeaderComponents';
import ListEmployeeComponents from './components/ListEmployeeComponents';
import AddEmployee from './components/AddEmployee';

function App() {
  return (
    <div>

        <HeaderComponents/>
          <BrowserRouter>
          <Routes>
            <Route path="/" element={<ListEmployeeComponents/>}></Route>
            <Route path="/employee" element={<ListEmployeeComponents/>}></Route>
            <Route path="/add-employee" element={<AddEmployee/>}></Route>
            <Route path="/edit-employee/:id" element={<AddEmployee/>}></Route>
            </Routes>
          </BrowserRouter>
         <Footer/>      You, 2 seconds ago • Uncommitted changes


    </div>
  );
}

export default App;

```

App.css M

react-hook-frontend > src > App.css > .footer

You, 1 second ago | 1 author (You)

```
.footer{
  position: absolute;
  bottom: 0;
  width: 100%;
  height: 50px;      You, 1 second ago • Uncommitted changes
  background-color: red;
  text-align: center;
  color: white;
}
```

EmployeeService.js U

react-hook-frontend > src > service > EmployeeService.js > EmployeeService > deleteEmployee

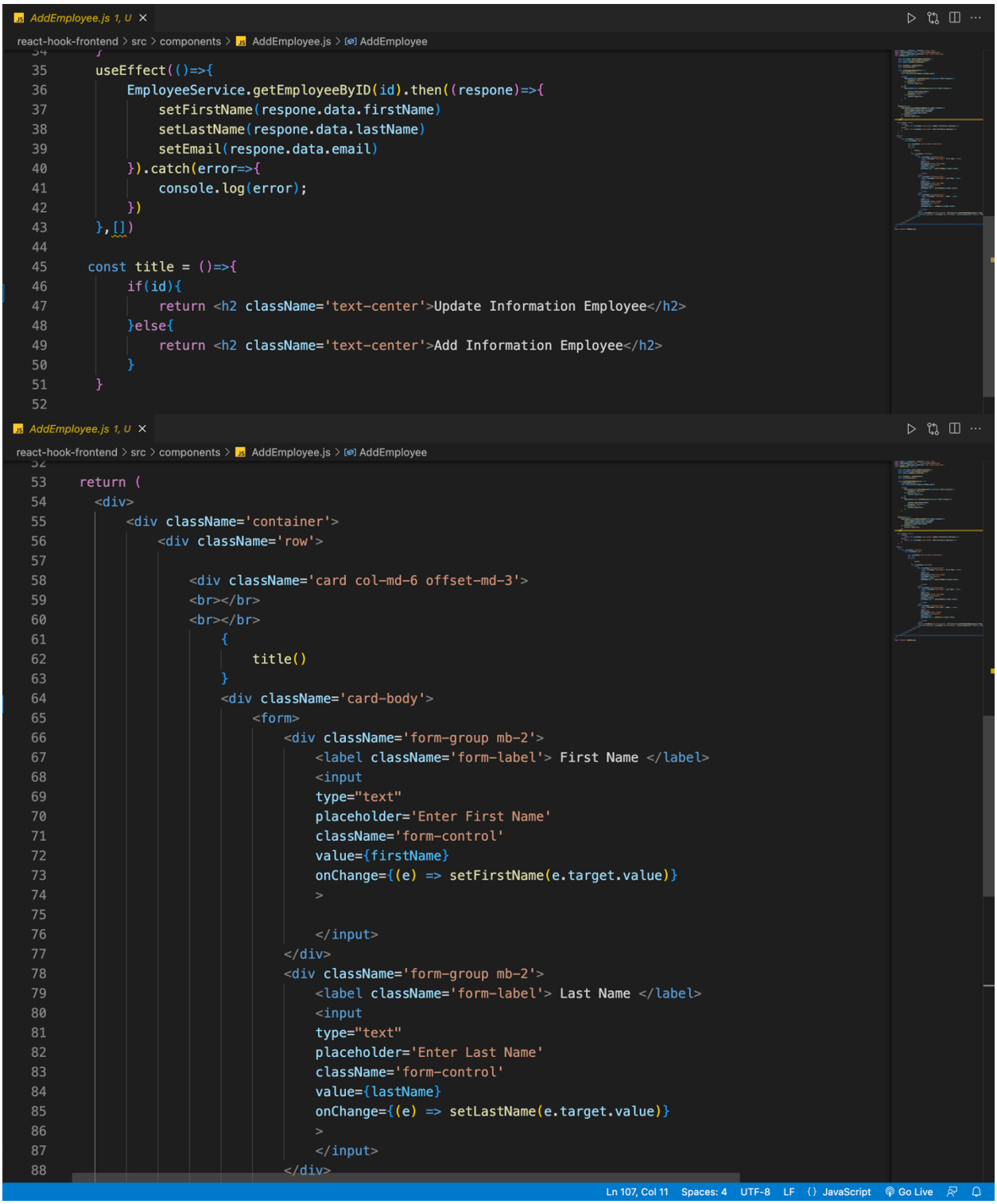
```
import axios from 'axios';

const EMPLOYEE_BASE_REST_API_URL ="http://localhost:8080/api/employee";

class EmployeeService{

    getAllEmployee(){
        return axios.get(EMPLOYEE_BASE_REST_API_URL)
    }
    createEmployee(employee){
        return axios.post(EMPLOYEE_BASE_REST_API_URL,employee)
    }
    getEmployeeByID(employeeId){
        return axios.get(EMPLOYEE_BASE_REST_API_URL + '/' +employeeId)
    }
    updateEmployee(employeeId,employee){
        return axios.put(EMPLOYEE_BASE_REST_API_URL + '/' + employeeId,employee)
    }
    deleteEmployee(employeeId){
        return axios.delete(EMPLOYEE_BASE_REST_API_URL + '/' +employeeId)
    }
}
export default new EmployeeService();
```

AddEmployee.js 1, U

react-hook-frontend > src > components > AddEmployee.js > AddEmployee

```
import React, { useEffect, useState } from 'react'
import EmployeeService from '../service/EmployeeService'
import {Link, useNavigate,useParams} from 'react-router-dom';
const AddEmployee = () => {

    const [firstName,setFirstName]=useState('')
    const [lastName,setLastName]=useState('')
    const [email,setEmail]=useState('')

    const navigate = useNavigate();
    const {id}=useParams();

    const saveOrUpdateEmployee=(e) =>{
        e.preventDefault();
        const employee={firstName,lastName,email}

        if(id){
            EmployeeService.updateEmployee(id,employee).then((respone)=>{
                navigate('/employee')
            }).catch(error => {
                console.log(error);
            })
        }else{
            EmployeeService.createEmployee(employee).then((respone)=>{

                console.log(respone.data);
                navigate('/employee');
            }).catch(error => {
                console.log(error);
            })
        }


    }
    useEffect(()=>{
        EmployeeService.getEmployeeByID(id).then((respone)=>{
            setFirstName(respone.data.firstName)
```

Ln 107, Col 11 Spaces: 4 UTF-8 LF { } JavaScript Go Live


AddEmployee.js 1, U
react-hook-frontend > src > components > AddEmployee.js > AddEmployee
35 useEffect(()=>{
36 EmployeeService.getEmployeeByID(id).then((respone)=>{
37 setFirstName(respone.data.firstName)
38 setLastName(respone.data.lastName)
39 setEmail(respone.data.email)
40 }).catch(error=>{
41 console.log(error);
42 })
43 },[])
44
45 const title = ()=>{
46 if(id){
47 return <h2 className='text-center'>Update Information Employee</h2>
48 }else{
49 return <h2 className='text-center'>Add Information Employee</h2>
50 }
51 }
52
AddEmployee.js 1, U
react-hook-frontend > src > components > AddEmployee.js > AddEmployee
53 return (
54
55
56
57
58
59
</br>
60
</br>
61 {
62 title()
63 }
64
65 <form>
66
67 <label className='form-label'> First Name </label>
68 <input
69 type="text"
70 placeholder='Enter First Name'
71 className='form-control'
72 value={firstName}
73 onChange={(e) => setFirstName(e.target.value)}
74 >
75
76 </input>
77
78
79 <label className='form-label'> Last Name </label>
80 <input
81 type="text"
82 placeholder='Enter Last Name'
83 className='form-control'
84 value={lastName}
85 onChange={(e) => setLastName(e.target.value)}
86 >
87 </input>
88
Ln 107, Col 11 Spaces: 4 UTF-8 LF JavaScript Go Live

AddEmployee.js 1, U

react-hook-frontend > src > components > AddEmployee.js > AddEmployee

```
                                    </input>
                                </div>
                                <div className='form-group mb-2'>
                                    <label className='form-label'> Last Name </label>
                                    <input
                                    type="text"
                                    placeholder='Enter Last Name'
                                    className='form-control'
                                    value={lastName}
                                    onChange={(e) => setLastName(e.target.value)}
                                    >
                                    </input>
                                </div>
                                <div className='form-group mb-2'>
                                    <label className='form-label'> Email </label>
                                    <input
                                    type="text"
                                    placeholder='Enter E-mail'
                                    className='form-control'
                                    value={email}
                                    onChange={(e) => setEmail(e.target.value)}
                                    >
                                    </input>
                                </div>
                                <button className='btn btn-success' onClick={(e)=>saveOrUpdateEmployee(e)}>Sa
                                <Link to="/employee" className='btn btn-danger' style={{marginLeft:"10px"}}>
                            </form>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    )
}
export default AddEmployee
```

Footer.js U

react-hook-frontend > src > components > Footer.js > Footer

```
import React from 'react'

function Footer() {
  return (
    <div>
        <footer>
            <nav className='footer'>
                <span>All Right Spring Boot Develop on 23 December 2022 by Luon Verak</span>
            </nav>
        </footer>
    </div>
  )
}

export default Footer
```

HeaderComponents.js U

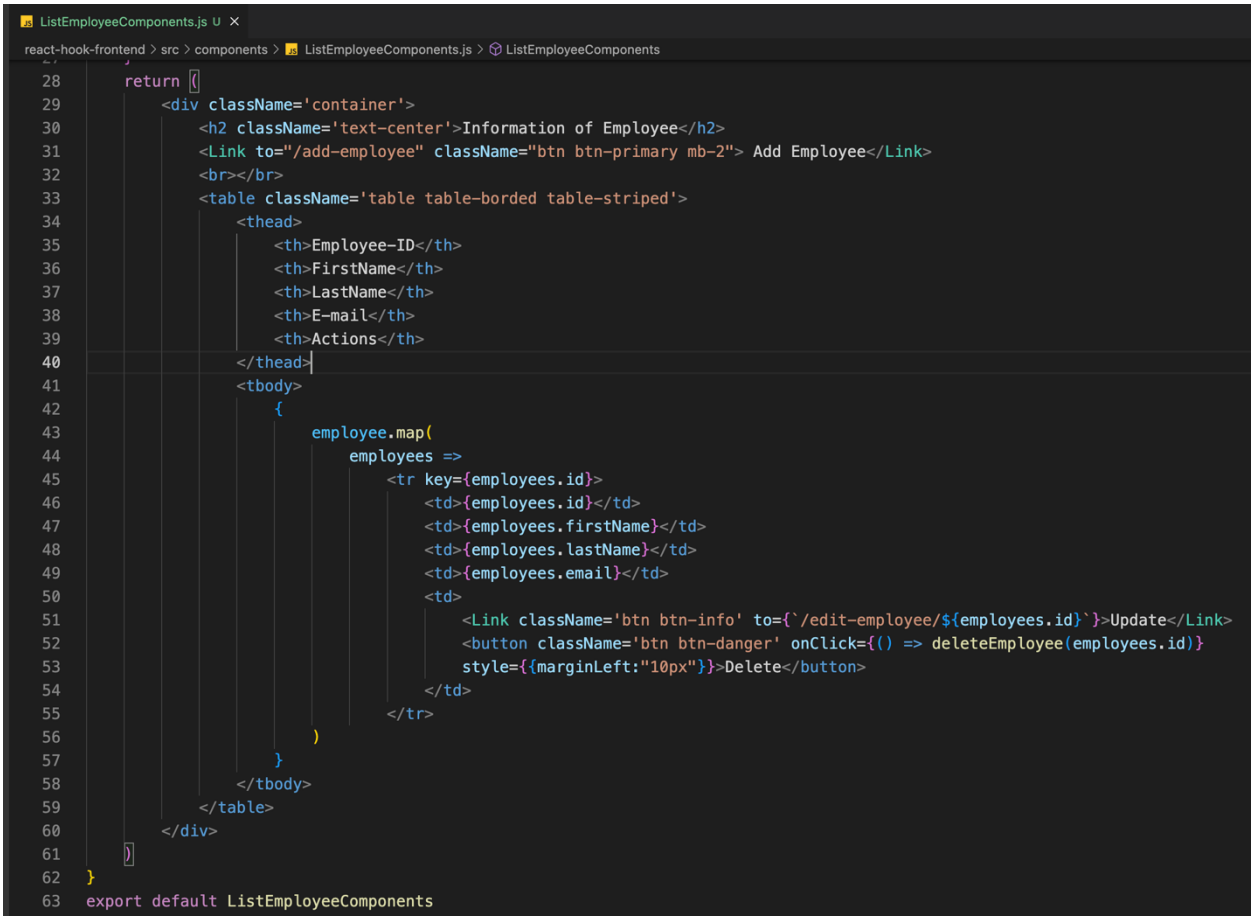
react-hook-frontend > src > components > HeaderComponents.js > HeaderComponents

```jsx
import React from 'react'

function HeaderComponents() {
  return (
    <div>
        <header>
            <nav className='navbar navbar-expand-md navbar-dark bg-dark'>
                <div>
                    <a href='https://www.facebook.com/?_rdc=2&_rdr' className='navbar-brand'>
                        Employee Management System
                    </a>
                </div>
            </nav>
        </header>
    </div>
  )
}

export default HeaderComponents
```

ListEmployeeComponents.js U

react-hook-frontend > src > components > ListEmployeeComponents.js > ListEmployeeComponents

```jsx
import React, { useEffect, useState } from 'react'
import { Link } from 'react-router-dom';
import EmployeeService from '../service/EmployeeService';

function ListEmployeeComponents() {
    const [employee, setEmployee] = useState([]);
    useEffect(() => {
        getAllEmployee();
    }, [])

    const getAllEmployee=()=>{
        EmployeeService.getAllEmployee().then((response) => {
                    setEmployee(response.data)
                    console.log(response.data)
                }).catch(
                    error => {
                        console.log(error);
                    }
                )
    }
    const deleteEmployee = (employeeId) => {
        EmployeeService.deleteEmployee(employeeId).then((response)=>{
            getAllEmployee();
        }).catch(error =>{
            console.log(error);
        })
    }
    return (
        <div className='container'>
            <h2 className='text-center'>Information of Employee</h2>
            <Link to="/add-employee" className="btn btn-primary mb-2"> Add Employee</Link>
            <br></br>
            <table className='table table-borded table-striped'>
                <thead>
                    <th>Employee-ID</th>
                    <th>FirstName</th>
                    <th>LastName</th>
```



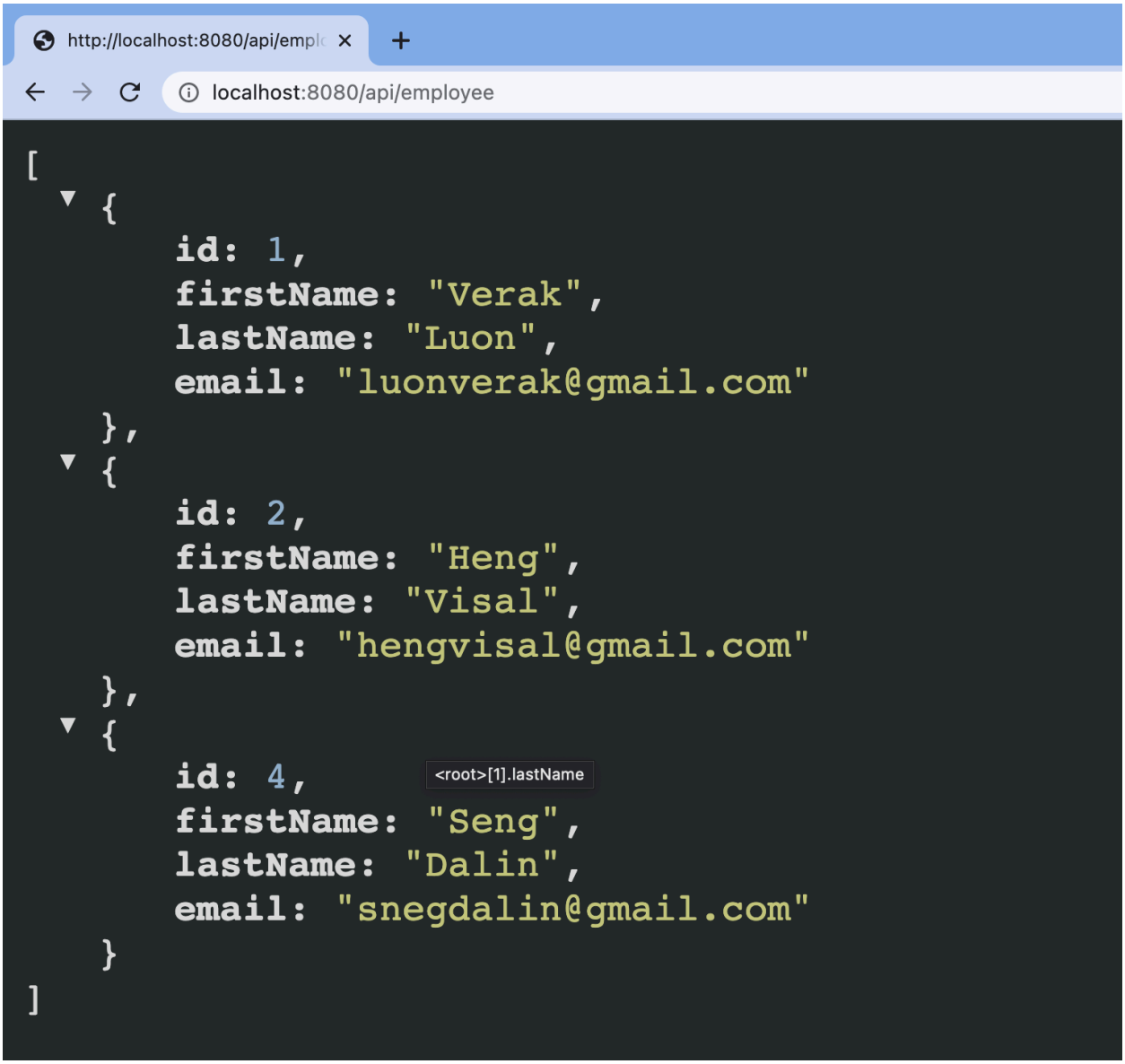
```
ListEmployeeComponents.js U
react-hook-frontend > src > components > ListEmployeeComponents.js > ListEmployeeComponents
    return (
        <div className='container'>
            <h2 className='text-center'>Information of Employee</h2>
            <Link to="/add-employee" className="btn btn-primary mb-2"> Add Employee</Link>
            <br></br>
            <table className='table table-borded table-striped'>
                <thead>
                    <th>Employee-ID</th>
                    <th>FirstName</th>
                    <th>LastName</th>
                    <th>E-mail</th>
                    <th>Actions</th>
                </thead>
                <tbody>
                    {
                        employee.map(
                            employees =>
                                <tr key={employees.id}>
                                    <td>{employees.id}</td>
                                    <td>{employees.firstName}</td>
                                    <td>{employees.lastName}</td>
                                    <td>{employees.email}</td>
                                    <td>
                                        <Link className='btn btn-info' to={`/edit-employee/${employees.id}`}>Update</Link>
                                        <button className='btn btn-danger' onClick={() => deleteEmployee(employees.id)}
                                        style={{marginLeft:"10px"}}>Delete</button>
                                    </td>
                                </tr>
                        )
                    }
                </tbody>
            </table>
        </div>
    )
}
export default ListEmployeeComponents
```

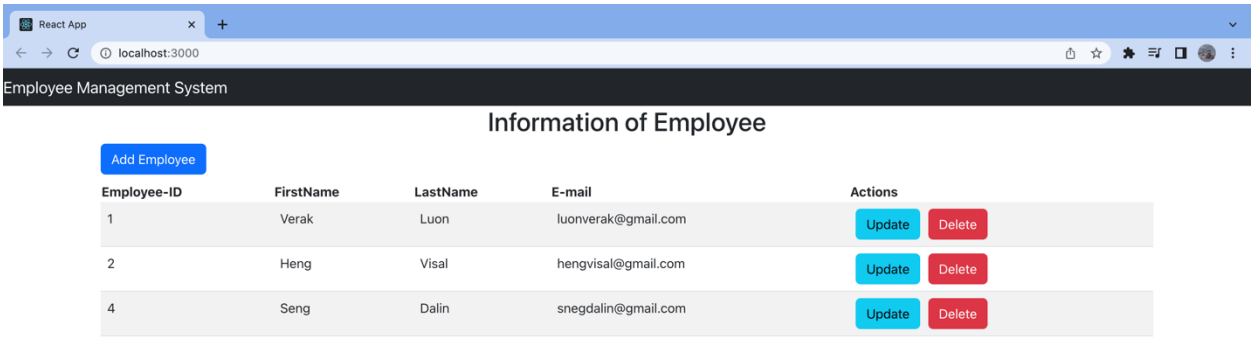
Table Databases

	id	email	first_name	last_name
Edit Copy Delete	1	luonverak@gmail.com	Verak	Luon
Edit Copy Delete	2	hengvisal@gmail.com	Heng	Visal
Edit Copy Delete	4	snegdalin@gmail.com	Seng	Dalin

Check all *With selected:* Edit Copy Delete Export

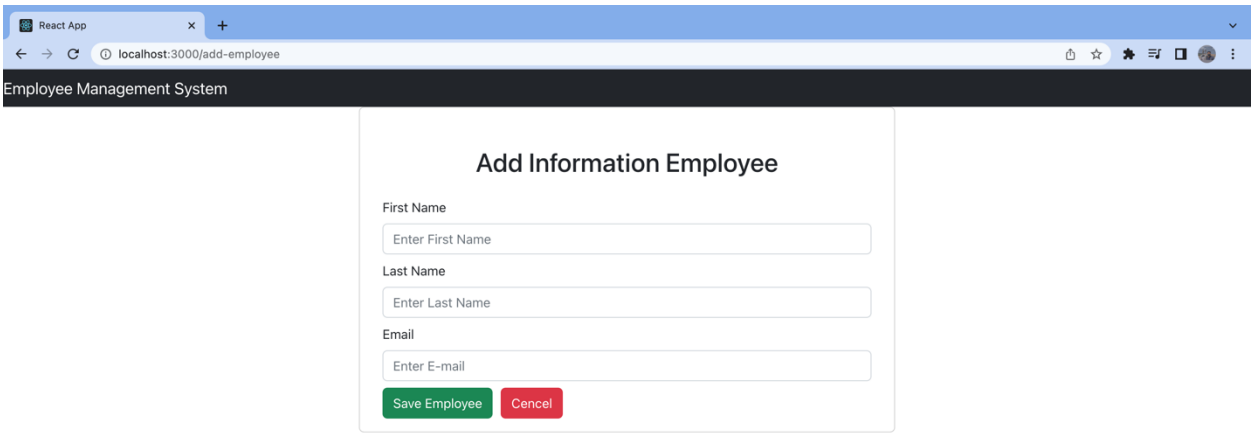
Result API


http://localhost:8080/api/empl
localhost:8080/api/employee
[
{
id: 1,
firstName: "Verak",
lastName: "Luon",
email: "luonverak@gmail.com"
},
{
id: 2,
firstName: "Heng",
lastName: "Visal",
email: "hengvisal@gmail.com"
},
{
id: 4,
<root>[1].lastName
firstName: "Seng",
lastName: "Dalin",
email: "snegdalin@gmail.com"
}
]



All Right Spring Boot Develop on 23 December 2022 by Luon Verak

Button Add New Employee or Update



All Right Spring Boot Develop on 23 December 2022 by Luon Verak