**2022년도 졸업프로젝트[Back-end]**

**팀이름: 황제펭귄**

주제: TLS를 구현을 통한 홈페이지 제작 및 보안 채널 로그인 기능 구현

|  |  |  |  |
| --- | --- | --- | --- |
| **작성자** | 이유민 | **작성일자** | 2022.02.22 |

* **금일 진행 상황**

|  |  |
| --- | --- |
| **진행 상황** | **비고** |
| **질문 및 진행방식에 대한 논의**  **진행 중**   1. 스프링 스터디 진행.   **완료**  프로젝트 환경설정   1. 프로젝트 생성   -https://start.spring.io/ 에서 spring initializr 로 만든 gradle project파일로 Intellij iIDEA에서 프로젝트 생성을 진행함.  -스프링 부트 메인 실행 후 에러 페이지로 간단하게 동작을 확인 함.  @SpringBootApplication  public class HelloSpringApplication {  public static void main(String[] args) {  SpringApplication.run(HelloSpringApplication.class, args);  }  }  :위의 메인메소드를 실행하면 SpringApplication.run 안에 위의 클래스가 들어가고 SpringBoot Application이 실행되는데,SpringBoot Application은 tomcat이란 웹서버를 내장하여 웹서버를 자체적으로 띄우면서 SpringBoot가 같이 올라가는 형식이다.  -Preferences 에서 InteliJ가 Gradle을 통해 실행하는 기본 설정을 자바로 바로 실행할 수 있도록 변경함.   1. 라이브러리 살펴보기   -gradle이나 maven같은 build tool들은 라이브러리를 끌고 오면 의존관계에 있는 여러 라이브러리를 자동으로 가져와 관리해준다.  -왼쪽 최하단의 네모 버튼을 한번 누르면 왼쪽에 GRADLE과 DATABASE 목록을 볼 수 있는데, Gradle하단에 dependencies에서 라이브러리들 간의 의존관계를 볼 수 있다.( dependencies에서 (\*)은 위에서 언급되어 중복을 제거해주었다는 표시이다.)  - 스프링 부트 라이브러리  spring-boot-starter-web  spring-boot-starter-tomcat: 톰캣 (웹서버)  spring-webmvc: 스프링 웹 MVC  spring-boot-starter-thymeleaf: 타임리프 템플릿 엔진(View)  spring-boot-starter(공통): 스프링 부트 + 스프링 코어 + 로깅  spring-boot  spring-core  spring-boot-starter-logging  logback, slf4j  -테스트 라이브러리  spring-boot-starter-test  junit: 테스트 프레임워크  mockito: 목 라이브러리  assertj: 테스트 코드를 좀 더 편하게 작성하게 도와주는 라이브러리 spring-test: 스프링 통합 테스트 지원   1. View 환경설정   -정적 페이지인 Welcom Page를 만들기.  resources/static에 HTML File로 index.html을 생성 후  <!DOCTYPE HTML>  <html>  <head>  <title>Hello</title>  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />  </head>  <body>  Hello  <a href="/hello">hello</a>  </body>  </html>  아래의 코드를 입력하여 Welcom Page 작성하여 Hello hello를 출력하는정적 페이지 생성.  -템플릿 엔진 사용하기.  JAVA/hello.hellospring에 package 파일로 Controller 생성 후 하단의 코드  입력.  package hello.hellospring.controller;  import org.springframework.stereotype.Controller;  import org.springframework.ui.Model;  import org.springframework.web.bind.annotation.GetMapping;  @Controller  public class HelloController {  @GetMapping("hello")  public String hello(Model model){  model.addAttribute("data","hello!!");  return "hello";  }  }  이 메소드에서 Controller의 return값은 템플릿 하단의 이름이 동일한 파일을 찾아서 렌더링을 하기 때문에 이후에 templates 하단의 파일과 연동되어 data 값을 attribute value인 hello!로 변환해준다.  -resorces/templates 하단에 Controller이름.html파일을 생성하여 아래의 코드를 작성한다.  <!DOCTYPE HTML>  <html xmlns:th="http://www.thymeleafs.org">  <head>  <title>Hello</title>  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />  </head>  <body>  <p th:text="'안녕하세요. ' + ${data}">안녕하세요. 손님</p>  </body>  </html>  :  <html xmlns:th="http://www.thymeleafs.org"> 템플릿을 적용함  th :templates를 의미한다.  <p th:text="'안녕하세요. ' + ${data}">안녕하세요. 손님</p> Controller와 연결되어 data 부분이 hello!로 자동변환됨.  -스프링부트는 기능이 방대하여 다 외울 수 없기 때문에 원하는 기능을 spring.io-projects-springboot-Learn-자신의 버전-reference Doc으로 들어가 원하는 기능을 찾는 능력을 갖춰야 한다.   1. 빌드하고 실행하기   -콘솔로 이동  1. ./gradlew build  2. cd build/libs  3. java -jar hello-spring-0.0.1-SNAPSHOT.jar  4. 실행 확인.  **진행 예정**   1. **24일 8시에 게더타운에서 백엔드 스터디를 진행할 것.** |  |
| * **특이사항 / 협업 사항**   매주 화, 수, 목 7시30부터 10시 30분까지 유동적으로 하기. | |