**2022년도 졸업프로젝트**

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

|  |  |  |  |
| --- | --- | --- | --- |
| **작성자** | 이유민 | **작성일자** | 2022.05.09 |

* **금일 진행 상황**

|  |  |
| --- | --- |
| **진행 상황** | **비고** |
| **질문 및 진행방식에 대한 논의**  **진행 중**   1. **상담센터 게시판 기능 구현**   **완료**   1. **이벤트 및 새소식 게시판 기능 Fix**   **-EventController**  package com.EmperorPenguin.SangmyungBank.api.event.controller;  import com.EmperorPenguin.SangmyungBank.api.event.model.Event; import com.EmperorPenguin.SangmyungBank.api.event.service.EventService; import lombok.RequiredArgsConstructor; import org.springframework.http.HttpStatus; import org.springframework.http.ResponseEntity; import org.springframework.web.bind.annotation.\*;  import javax.servlet.http.HttpServletResponse; import java.time.LocalDateTime; import java.util.List; import java.util.Map; import java.util.Optional;  @RequiredArgsConstructor @RestController @RequestMapping("/cont") public class EventController {    private final EventService eventService;   // create event  @PostMapping("/event")  public ResponseEntity<HttpStatus> createEvent(@RequestBody Event event) {  event.setCreatedDate(LocalDateTime.*now*());  Event resultEvent = eventService.createEvent(event);  if(resultEvent == null) {  return ResponseEntity.*ok*(HttpStatus.*BAD\_REQUEST*);  }  else {  return ResponseEntity.*ok*(HttpStatus.*OK*);  }    } // public Event createEvent(@RequestBody Event event) // { // event.setCreatedDate(LocalDateTime.now()); // return eventService.createEvent(event); // }   // list all events  @GetMapping("/event")  public ResponseEntity<List<Event>> listAllEvents() {  List<Event> eventList = eventService.listAllEvents();  if (eventList == null) {  return ResponseEntity.*status*(HttpStatus.*NOT\_FOUND*)  .body(null);  }  else {  return ResponseEntity.*status*(HttpStatus.*OK*)  .body(eventList);  }  }   // get event by id  @GetMapping("/event/{id}")  public ResponseEntity<Event> getEventById(@PathVariable Long id) {  Event event = eventService.getEventById(id);  if (event == null) {  return ResponseEntity.*status*(HttpStatus.*NOT\_FOUND*)  .body(null);  }  else {  return ResponseEntity.*status*(HttpStatus.*OK*)  .body(event);  }  }   // update event  @PutMapping("/event/{id}")  public ResponseEntity<HttpStatus> updateEvent(@PathVariable Long id, @RequestBody Event eventDetails) {  Event event = eventService.updateEvent(id, eventDetails);  if (event == null) {  return ResponseEntity.*ok*(HttpStatus.*BAD\_REQUEST*);  } else {  return ResponseEntity.*ok*(HttpStatus.*OK*);  }  }  // delete event  @DeleteMapping("/event/{id}")  public ResponseEntity<HttpStatus> deleteEvent(@PathVariable Long id) {  Event event = eventService.deleteEvent(id);  if (event == null) {  return ResponseEntity.*ok*(HttpStatus.*BAD\_REQUEST*);  }  else {  return ResponseEntity.*ok*(HttpStatus.*OK*);  }  } // @DeleteMapping("/event/{id}") // public ResponseEntity<Map<String, Boolean>> deleteEvent(@PathVariable Long id) { // return eventService.deleteEvent(id); // } }  **-postman 동작 확인**  **텍스트이(가) 표시된 사진  자동 생성된 설명텍스트이(가) 표시된 사진  자동 생성된 설명텍스트이(가) 표시된 사진  자동 생성된 설명텍스트이(가) 표시된 사진  자동 생성된 설명텍스트이(가) 표시된 사진  자동 생성된 설명**  \*새소식 게시판 기능 Fix는 이벤트 게시판 기능과 코드가 유사함으로 생략.   1. **보안공지 기능 구현** 2. **기능리스트 Fix**   **진행 예정**  **1.상담센터 게시판 기능 구현**  **5월 10일 15:00시 부터 User Entity를 FK로 참조해서 상담센터 게시판의 글쓴이 데이터 표시 진행.** | re |
| * **특이사항 / 협업 사항** | |