Annotations support
==========
1. @RestController(@Controller+@ResponseBody)
2. @RequestBody
3. @RequestParam
4. @PathVariable
5. @RequestMapping
6. @GetMapping
7. @PostMapping
8. @PutMapping
9. @PatchMapping
10. @DeleteMapping
11. @RestControllerAdvice
12. @ExceptionHandler
13. @Service
14. @Component
15. @Repository
16. @Transactional

Usage of DEV-tools
==========
=> It informs SpringBoot container to automatically reload the applications when changes are made in the project.

```
<dependency>
	<groupId>org.springframework.boot</groupId>
	<artifactId>spring-boot-devtools</artifactId>
</dependency>
```

Code of Reading the record
==========

```
public class TouristNotFoundException extends RuntimeException {
	private static final long serialVersionUID = 1L;
	public TouristNotFoundException(String message) {
		super(message);
	}
}
```

Servicelayer
==========

```
Optional<Tourist> optional = repo.findById(id);

if (optional.isPresent()) {
	return optional.get();
} else {
	throw new TouristNotFoundException(" tourist with id "+id+" not found");
}
```

or

```
return repo.findById(id)
		.orElseThrow(() -> new TouristNotFoundException("tourist with id :: " + id + " not found"));
```

refer:: SpringRest-06-TicketManagementApp

SpringRest/SpringMVC is having PreDefined Controller to handle the

Exceptions/HttpErrors which is gives default whitelable error pages.

```
ErrorController(I)
    |
AbstractController(C)
    |
BasicErrorController(C)
    |=> errorHtml(,,,) :: ModelAndView
    |=> error() :: ResponseEntity<T>
```

SpringBootMVC
============

```
@RequestMapping(produces = MediaType.TEXT_HTML_VALUE)
public ModelAndView errorHtml(HttpServletRequest request, HttpServletResponse
response) {
      HttpStatus status = getStatus(request);
      Map<String, Object> model =
Collections.unmodifiableMap(getErrorAttributes(request,
getErrorAttributeOptions(request, MediaType.TEXT_HTML)));
      response.setStatus(status.value());
      ModelAndView modelAndView = resolveErrorView(request, response, status,
model);
      return (modelAndView != null) ? modelAndView : new ModelAndView("error",
model);
}
```

SpringBootRest
==============

```
@RequestMapping
public ResponseEntity<Map<String, Object>> error(HttpServletRequest request) {
      HttpStatus status = getStatus(request);
      if (status == HttpStatus.NO_CONTENT) {
            return new ResponseEntity<>(status);
      }
      Map<String, Object> body = getErrorAttributes(request,
getErrorAttributeOptions(request, MediaType.ALL));
      return new ResponseEntity<>(body, status);
}
```

Note::
@RestControllerAdvice =>@ControllerAdvice+@ReponseBody+@Component

GlobalExceptionHandler in SpringRest
====================================

```
@RestControllerAdvice
public class TouristErrorControllerAdvice {

      @ExceptionHandler(TouristNotFoundException.class)
      public ResponseEntity<ErrorDetails>
handleTouristNotFound(TouristNotFoundException tf) {

      System.out.println("TouristErrorControllerAdvice.handleTouristNotFound()");
            ErrorDetails details = new ErrorDetails(LocalDateTime.now(),
tf.getMessage(), "404-NotFound");
            return new ResponseEntity<ErrorDetails>(details, HttpStatus.NOT_FOUND);
      }
```

```
	@ExceptionHandler(Exception.class)
	public ResponseEntity<ErrorDetails> handleAllProblems(Exception e) {
		System.out.println("TouristErrorControllerAdvice.handleAllProblems()");
		ErrorDetails details = new ErrorDetails(LocalDateTime.now(),
e.getMessage(), "Problem in exeuction");
		return new ResponseEntity<ErrorDetails>(details,
HttpStatus.INTERNAL_SERVER_ERROR);
	}
}
```

refer:: SpringRest-07-TicketManagementAppErrorResponseApp