



Hobby Web Application Project

Technologies Used

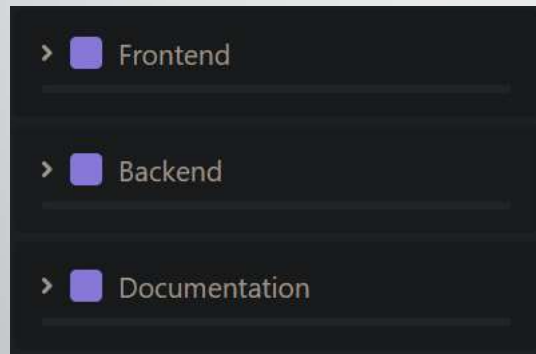
- Version Control System – Git and GitHub
- Agile Board - Jira
- Database – MySQL (for production) and H2 (for testing)
- Back-end – Java
- API Development – Spring Boot
- Build Tool – Maven
- Testing – Junit and Mockito
- Front-end – HTML, CSS, JavaScript

Concept

- Simple Movie Database
- You can log movies by inserting their title, release year and director
- Complete CRUD functionality

Sprint Plan

- A single sprint – 1 week long
- 3 epics – front-end, back-end and documentation
- User stories with descriptions, story point estimates, MoSCoW prioritisation
- More screenshots in the “Documentation” folder



HWA-4. Create risk assessment document	DOCUMENTATION	5	🔴	10.00	AD
HWA-5. Create a UML	DOCUMENTATION	4	🔴	10.00	AD
HWA-6. Create a ERD	DOCUMENTATION	3	🔴	10.00	AD
HWA-7. Write a readme	DOCUMENTATION	5	🔴	10.00	AD
HWA-8. Create a presentation	DOCUMENTATION	8	🔴	10.00	AD
HWA-9. Screenshot of Swagger API documentation	DOCUMENTATION	5	🔴	10.00	AD
HWA-10. Created entity	BACKEND	4	🔴	10.00	AD
HWA-11. Added Create functionality	BACKEND	5	🔴	10.00	AD
HWA-12. Added Read functionality	BACKEND	8	🔴	10.00	AD
HWA-13. Added Update functionality	BACKEND	8	🔴	10.00	AD
HWA-14. Added Delete functionality	BACKEND	5	🔴	10.00	AD
HWA-15. Completed Integration Test	BACKEND	8	🔴	10.00	AD
HWA-16. Completed Unit tests	BACKEND	8	🔴	10.00	AD



Version Control

- Main branch (nothing is changed here)
- Dev branch (creating other branches and making small changes)
- Feature branches (for adding / updating features in the application)

Testing

- Integration test and unit tests
- Coverage – 94.7%
- More screenshots in the “Documents” folder

✓ HobbyWebApplication-Backend	98.8 %	1,267	16	1,283
src/main/java	94.7 %	285	16	301
> com.qa.main.domain	95.4 %	124	6	130
> com.qa.main	37.5 %	3	5	8
> com.qa.main.exceptions	0.0 %	0	3	3
> com.qa.main.services	97.5 %	79	2	81
> com.qa.main.controller	100.0 %	79	0	79
src/test/java	100.0 %	982	0	982

```
20 @RestController
21 @CrossOrigin
22 @RequestMapping("/movie")
23 public class MovieController {
24
25     private MovieService service;
26
27     public MovieController(MovieService service) {
28         this.service = service;
29     }
30
31     //Post Request
32
33     @PostMapping("/create")
34     public ResponseEntity<Movie> create(@RequestBody Movie movie) {
35         return new ResponseEntity<Movie>(service.create(movie), HttpStatus.CREATED);
36     }
37
38     //Get Requests
39
40     @GetMapping("/getAll")
41     public ResponseEntity<List<Movie>> getAll() {
42         return new ResponseEntity<List<Movie>>(service.getAll(), HttpStatus.OK);
43     }
44
45     @GetMapping("/getById/{id}")
46     public ResponseEntity<Movie> getById(@PathVariable long id) {
47         return new ResponseEntity<Movie>(service.getById(id), HttpStatus.OK);
48     }
49
50     @GetMapping("/getTitle/{title}")
51     public ResponseEntity<List<Movie>> getTitle(@PathVariable String title) {
52         return new ResponseEntity<List<Movie>>(service.getTitle(title), HttpStatus.OK);
53     }
54 }
```



Demonstration

Retrospective

- A lot of issues with Eclipse that took a long time to fix
- I would like to get better with HTML / CSS
- I would like to add a “Rating” feature and “Watched” feature in the future
- I would like to create other entities



Thank you