



# Hobby Web Application (HWA)

---

CARL ANGELES

# PROJECT PLANNING

Risk Assessment

MoSCoW Analysis

Jira (Kanban)

Entity Relationship Diagram

## Before Project

ID	Risk Description	Cause	Effect	Likelihood (1-5)	Impact (1-5)	Risk Rating (1-25)	Action
1	Lack of Time	Bad time management	Incomplete Project	3	4	20	Split project into small parts Regularly check progress
2	Insufficient knowledge on technology	Technology not covered during training	Requirements not being met due to features not being implemented	2	5	12	Ask trainer for help. Research online
3	PC Issues	Internet Issues, Hardware fails	Unable to work on project.	2	5	5	Regular backup on Git, Ask QA for a laptop to work your project on.
4	Response in Fetch requests returning 400/404 or 500 errors	Using the wrong variables or incorrect implementation	Project not working	2	5	6	Make sure variable names are consistent and clear. Check to see if it is using the right method to fetch.
5	Version Control not utilized correctly	Incorrect use of Git, no branches for features/testing or lack of pushing	No backups, cannot rollback when an error occurs	2	2	3	Always work on a new branch when implementing a new feature, regular push after each feature/functionality
6	Program not running properly	Not enough testing	Not meeting requirements, program not working as intended	3	5	12	At least 80% coverage in the testing phase.

ID	Risk Description	Cause	Effect	Likelihood (1-5)	Impact (1-5)	Risk Rating (1-25)	Action
1	Lack of Time	Bad time management	Incomplete Project	5	5	22	Split project into small parts Regularly check progress
2	Insufficient knowledge on technology	Technology not covered during training	Requirements not being met due to features not being implemented	3	5	16	Ask trainer for help. Research online
3	PC Issues	Internet Issues, Hardware fails	Unable to work on project.	2	5	5	Regular backup on Git, Ask QA for a laptop to work your project on.
4	Response in Fetch requests returning 400/404 or 500 errors	Using the wrong variables or incorrect implementation	Project not working	4	5	18	Make sure variable names are consistent and clear. Check to see if it is using the right method to fetch.
5	Version Control not utilized correctly	Incorrect use of Git, no branches for features/testing or lack of pushing	No backups, cannot rollback when an error occurs	3	2	7	Always work on a new branch when implementing a new feature, regular push after each feature/functionality
6	Program not running properly	Not enough testing	Not meeting requirements, program not working as intended	3	5	12	At least 80% coverage in the testing phase.

## After Project

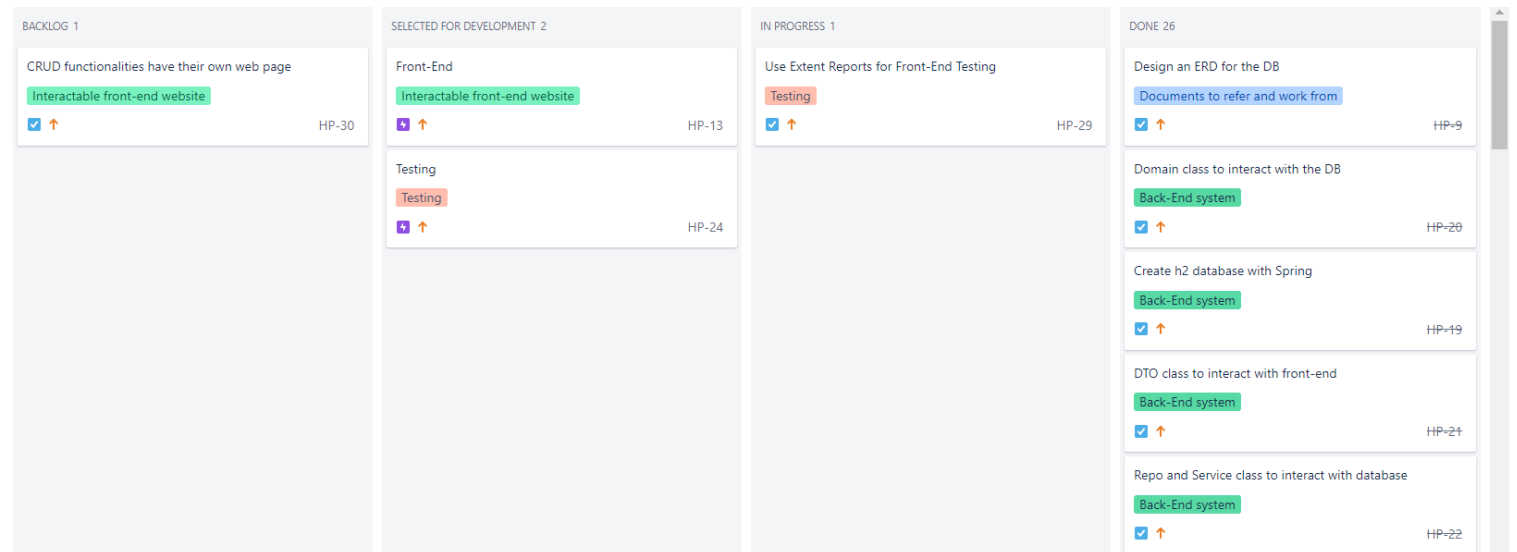
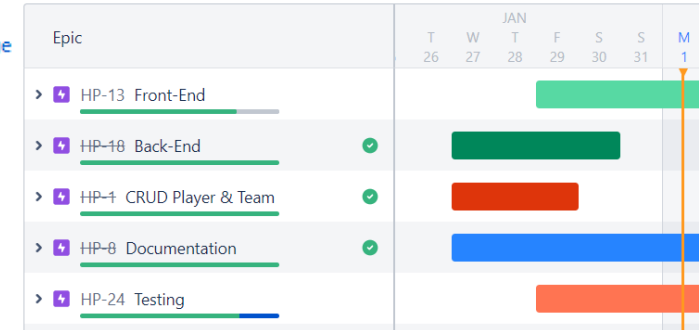
# RISK ASSESSMENT

<b>MUST HAVE</b>	<b>SHOULD HAVE</b>	<b>COULD HAVE</b>	<b>WOULD HAVE</b>
Database for Player and Team	Ability to display the players in a team	Ability to set a player with no team when its team is deleted.	Permission Control
Back-End with full CRUD functionality	Good User Interface	Log in System	
Front-End with full CRUD functionality	More ways to search player/team (By IGN/Name)		

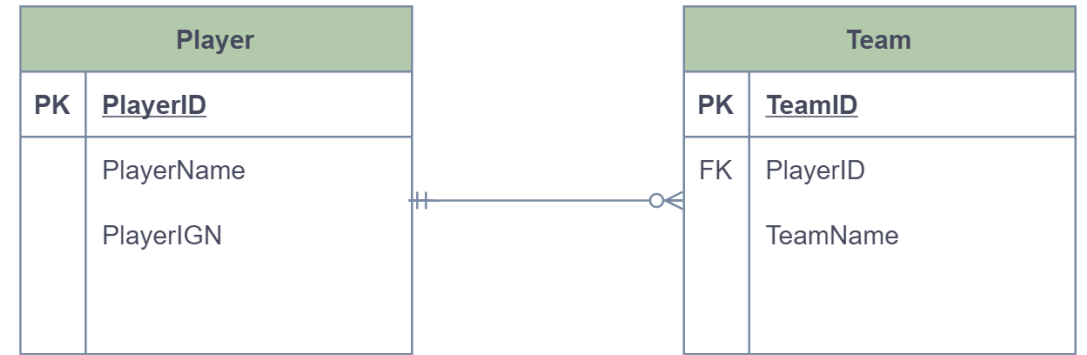
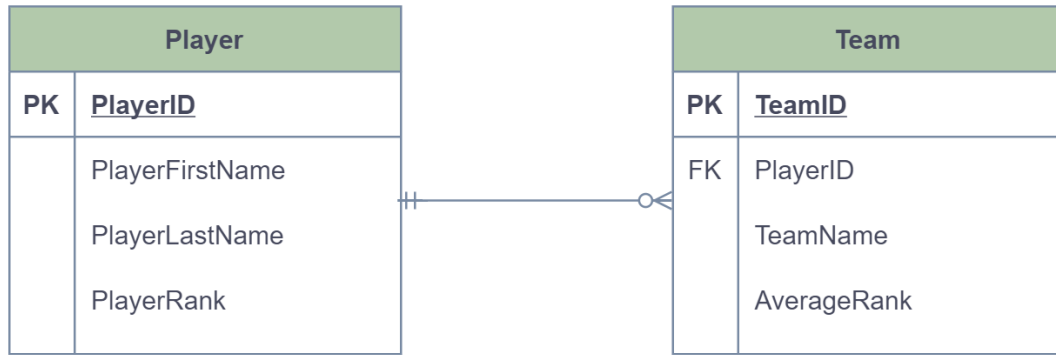
# MOSCOW ANALYSIS

# DESIGN (KANBAN)

Type	Key	Summary
✓	HP-30	CRUD functionalities have their own web page
✓	HP-29	Use Extent Reports for Front-End Testing
✓	HP-28	Selenium for User-Acceptance Testing
👤	HP-27	As a developer, I want to be able to have at least 80% test coverage
✓	HP-26	JUnit for Unit Testing
✓	HP-25	Mockito for Integration Testing
🔧	HP-24	Testing
✓	HP-23	Controller class that uses the service to call request
✓	HP-22	Repo and Service class to interact with database
✓	HP-21	DTO class to interact with front-end
✓	HP-20	Domain class to interact with the DB



## Planning



## Current

# ENTITY RELATIONSHIP DIAGRAM (ERD)

# CONSULTANT JOURNEY

Agile – Scrum, Kanban(Jira)

Source Control – Git, Github

Database – MySQL

Programming Language – Java, **HTML,**  
**JavaScript**

Build Tool – Maven

Testing – Junit, Mockito, **Selenium**

**Framework – Bootstrap, Spring Boot**

# TESTING

Finished after 39.429 seconds

Runs: 10/10

Errors: 0

Failures: 0

▼ SeleniumTest [Runner: JUnit 5] (36.571 s)

- testViewAllTeam() (6.243 s)
- testUpdateTeam() (3.388 s)
- testUpdatePlayer() (3.983 s)
- testDeletePlayer() (3.795 s)
- testPlayerCreate() (3.405 s)
- testViewAllPlayer() (3.234 s)
- testDeleteTeam() (3.295 s)
- testTeamCreate() (3.300 s)
- testViewPlayer() (3.283 s)
- testViewTeam() (2.645 s)

Element	Coverage	Covered Instructions	Missed Instructions
▼ HobbyWebApp	95.6 %	3,215	149
▼ src/main/java	88.9 %	981	122
▼ com.qa.persistence.domain	86.9 %	364	55
> PlayerDomain.java	86.6 %	207	32
> TeamDomain.java	87.2 %	157	23
▼ com.qa.persistence.dto	86.1 %	310	50
> PlayerDTO.java	86.1 %	155	25
> TeamDTO.java	86.1 %	155	25
▼ com.qa.rest	92.2 %	94	8
> PlayerController.java	92.2 %	47	4
> TeamController.java	92.2 %	47	4
▼ com.qa	37.5 %	3	5
> HobbyWebAppApplication.java	37.5 %	3	5
▼ com.qa.services	97.5 %	154	4
> PlayerService.java	97.5 %	77	2
> TeamService.java	97.5 %	77	2
▼ com.qa.config	100.0 %	7	0
> AppConfig.java	100.0 %	7	0
▼ com.qa.utils	100.0 %	49	0
> MyBeanUtils.java	100.0 %	49	0
> src/test/java	98.8 %	2,234	27



DEMONSTRATION

# SPRINT REVIEW

## **What did I complete?**

CRUD Functionalities for both entities

Minimum Viable Product

## **What got left behind?**

Features from lower priority and some in SHOULD column in MoSCoW table

Extent Reports for Front-End Testing

# SPRINT RETROSPECTIVE

## **What went well**

Dividing up tasks

Improved documentation compared to previous project

## **What could be improved**

Testing could be done after each feature

Use of Jira

# CONCLUSION

More organized

Front-end testing could have been handled better

Managed to get a working viable product