

BAE To-do List Project

By Brendan Woods

Introduction

Brendan Woods from London, 37 years old

- Professional Background
 - Retail Management 6 years
 - HR Operations 6 years
 - Career change
 - DevOps Dfe Oct'21
- Interests include
 - Tennis
 - Gaming
 - Sci-fi/Anime
 - Cooking and Travelling

The Concept / Approach

Tackling the Spec and My Approach

This was my first time experiencing these languages and the main technologies used for this project.

I first set out by doing a project plan of everything I had to cover as part of the project spec and tried to allocate time to each item over the sprint period of one week.

My initial approach was to tackle the project management element first using a Scrum Board to define the 'actionables' for the project.

This was then followed by setting up my VersionControlSystem (Git/GitHub) and SpringProject within Eclipse.

Once I had completed this I decided to work on my project chronologically to how I was taught over the last two weeks.

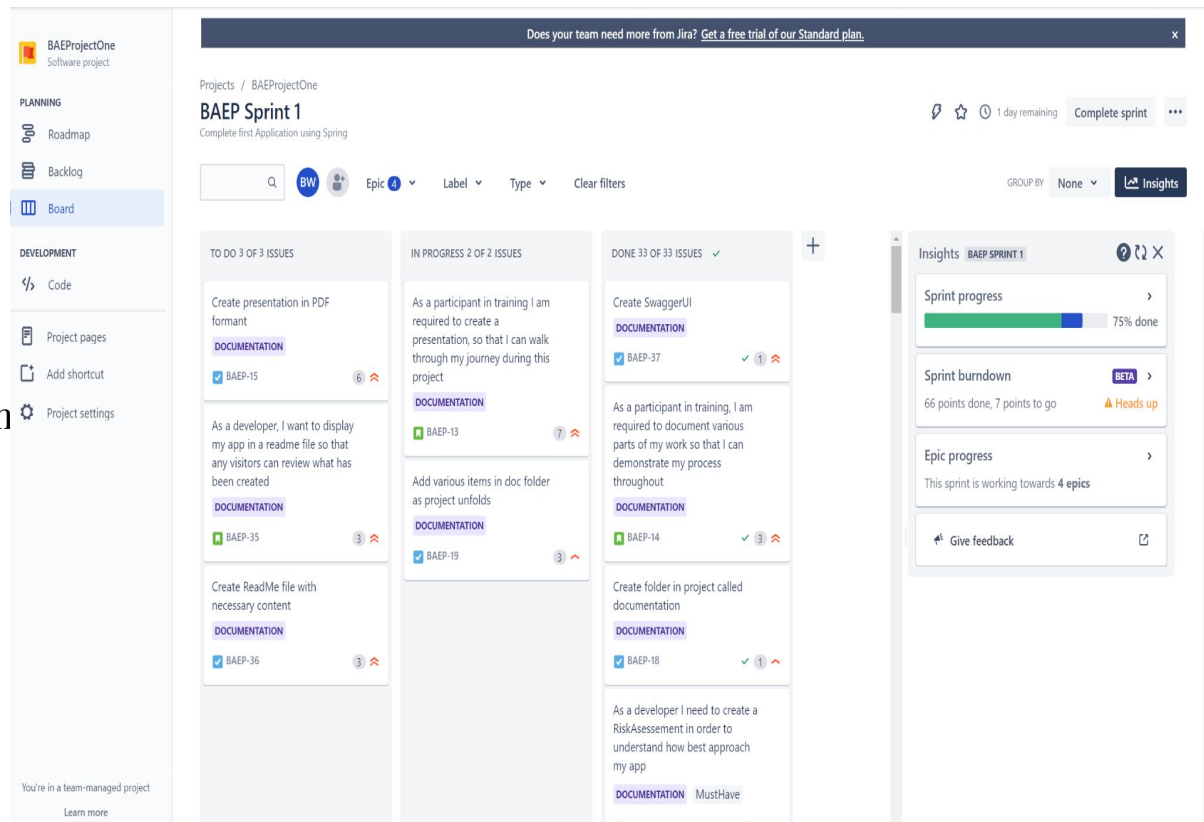
Meaning I decided to approach my Front-End, followed by Back-End, then Testing,

My approach didn't go quite as expected and I had to revisit my Front-End as I had to link it with my API.

Sprint / Scrum Board using Jira

Scrum Board using Jira

- Created 4 Epics, Front-End, Back-End, Testing, Docs
- User Stories created for Consumer and Developer
- Tasks created for Developer
- MoSCoW approach
- User Story point given



Testing

Testing

- Mirror file location, dummy data
- Integration – for API when combined together (Controller – Service – Repo)
- Unit testing for testing code in isolation
- Coverage 71.8%

The screenshot displays an IDE with three main panels:

- Left Panel (JUnit Results):** Shows test results for `TaskControllerIntegrationTest` (0.688 s), `BaetodolistApplicationTests` (0.004 s), and `TaskServiceTest` (0.050 s). It indicates 11 runs, 0 errors, and 0 failures.
- Right Panel (Source Code):** Displays the `Task` entity class. It includes annotations for `@Entity`, `@Id`, and `@GeneratedValue`. The class has a `private long id;`, a `private String taskToDo;`, and methods `getTaskToDo()` and `setTaskToDo(String taskToDo)`.
- Bottom Panel (Console):** Shows the execution log for `baetodolist` (27 Apr 2022 16:38:15). It includes log messages for database queries, application startup, and shutdown hooks.

Element	Covered	Instructions	Int
baetodolist	89.0 %	551	
src/main/java	71.8 %	173	
com.qa.baetodolist.domain	62.2 %	61	
com.qa.baetodolist.exceptions	0.0 %	0	
com.qa.baetodolist.services	86.2 %	56	
com.qa.baetodolist.controllers	37.5 %	3	
com.qa.baetodolist.config	93.0 %	53	
src/test/java	100.0 %	378	

Consultant Journey

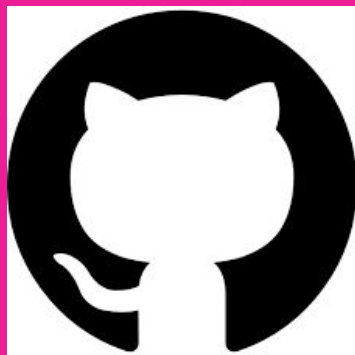


Visual Studio Code



eclipse®

↔ axios



Jira Software

Continuous Integration

Version Control

- Main > Dev > Feature
- Multiple commits per feature
- Merge then pull to Dev
- Create new Feature
- Repeat process
- Very end merge Dev to Main

The screenshot shows a GitHub pull request interface for 'Feature api #4'. At the top, it indicates 'Merged' and 'BenningtonW merged 7 commits into dev from feature-api yesterday'. Below this, a summary shows 'Conversation 0', 'Commits 7', 'Checks 0', and 'Files changed 10'. The main content area displays a commit history by BenningtonW, including 'setup packages', 'setup resources', 'setup database', 'setup repo and controller', and three 'service and controller changes' commits. A comment from BenningtonW states 'No description provided.' To the right, there are sections for 'Reviewers' (No reviews), 'Assignees' (No one—assign yourself), 'Labels' (None yet), 'Projects' (None yet), 'Milestone' (No milestone), and 'Development' (Successfully merging this pull request may close these issues). At the bottom, a message states 'Pull request successfully merged and closed' and 'You're all set—the feature-api branch can be safely deleted.' with a 'Delete branch' button. The interface is dark-themed.

My Reflection

What went well...

VS

What could have gone better....

- Version Control, no issues however features could have been more granular
- Setting Up API, went quite smoothly, a couple of rookie errors
- Time management, although misjudged in a few areas I recovered well to get on track again.

- Scrumboard - could have been more detailed and granular however I did not want to focus too much time considering the scope of project
- Front-End, more specifically JS. Very little experience and felt like I needed more time to learn the language before going into the project. I will continue with Scrimba and other learning materials to better grasp this language.
- Approach to project. I think I would have left the Front-End until last and focus on getting the Back-End in place first and would have allocated more time for the completion of this. I also didn't factor in linking my webpage to the API which took a considerable amount of time.