DAWG 2 HEALTHCARE SYSTEM

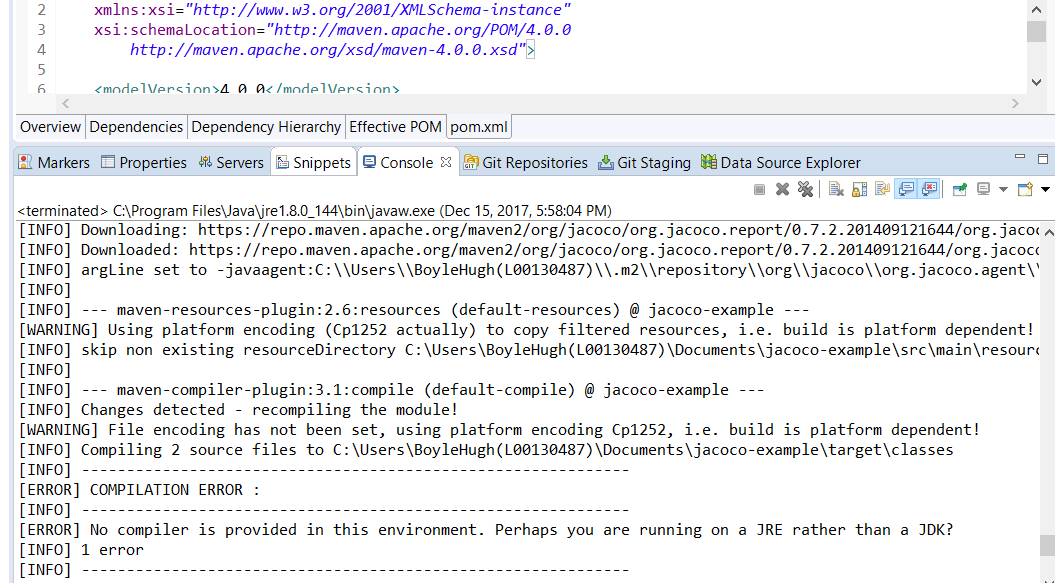
### **By John Kelly, Hugh J Boyle, Kieran Thompson, Eoin Gallen and Connor Doherty**

# **Use Case Descriptions**

|  |  |
| --- | --- |
| Use Case | Customer getting quote |
| Objective | To retrieve a quote for the customer for healthcare |
| Precondition | The webpage must be initialized. |
| Main Flow | 1. Customer enters personal details.  2. The customer enters their age.  3. The customer enters previous medical conditions.  4. The customer has enters occupation.  5. The quote is displayed. |
| Alternative Flow | 1.1 The user enters incorrect details, the appropriate error message displayed.  1.2 The user enters no details for a required field, the appropriate error message displayed.  2.1 The customer is under the age of 18, no change in quote.  2.2 The customer is between the ages of 18-25, 10% added to quote.  2.3 The customer is over the age of 25, 25% added to the quote.  3.1 The customer has no previous medical conditions, quote remains the same.  3.2 The customer has previous medical condition, 30% added to the quote.  4.1 The customer has safe occupation, quote remains the same.  4.2 The customer has dangerous occupation, 15% added to the quote. |
| Post Condition | The customer has retrieved the quote |

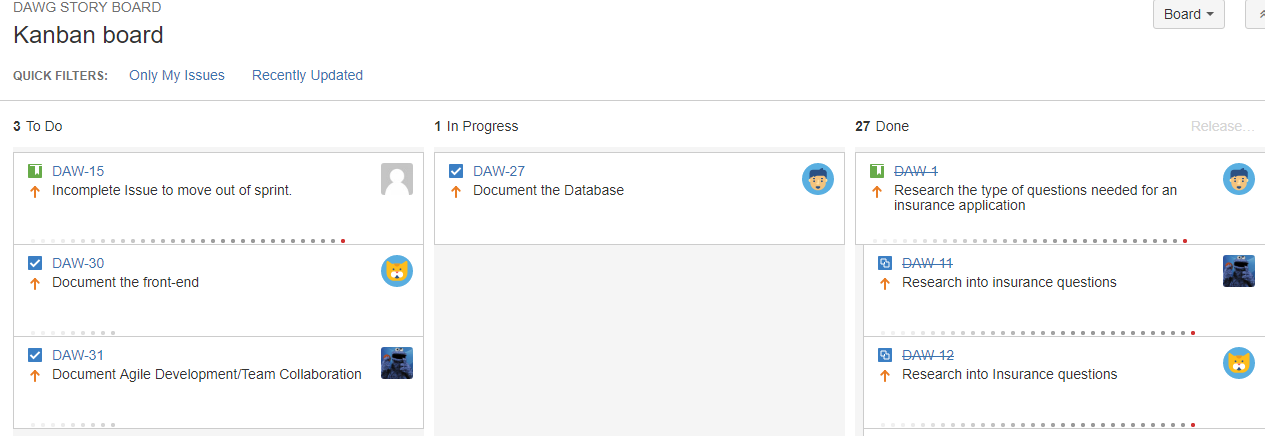
# JaCoCo

JaCoCo is a plugin for the Eclipse IDE which generates code coverage reports on testing run in the Java programming language. JaCoCo was used in the project for measuring the code coverage of the J-unit tests of the java methods.



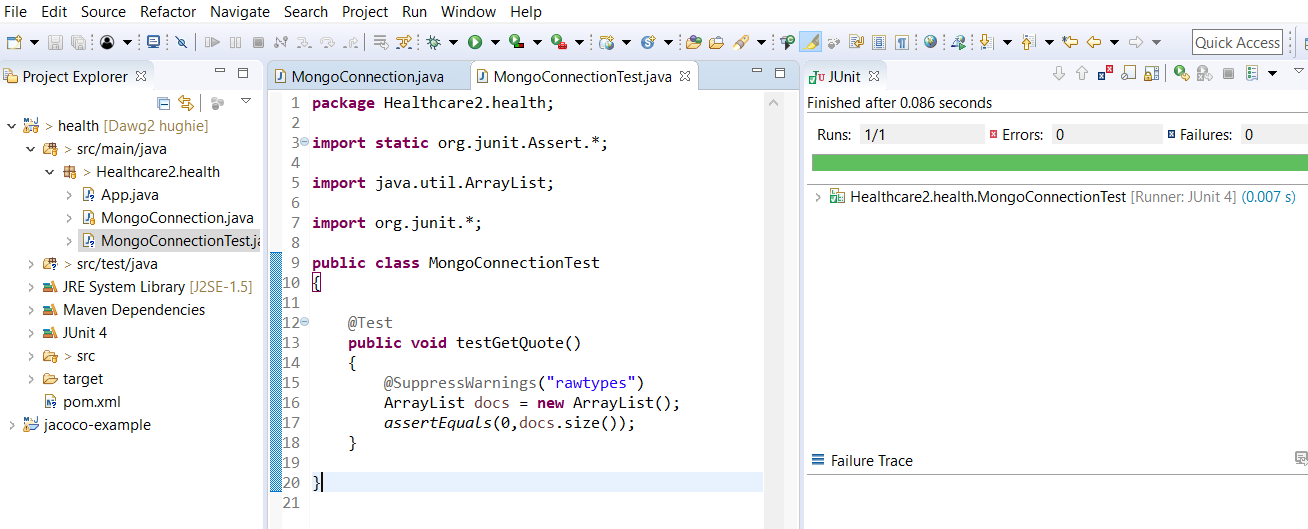
Jira

Jira is a project management tool that supports agile groups in tracking issues, assigning tasks and planning projects. Jira can also produce various reports and agile boards to view work to be done, in progress and work completed. It was used in this project for assigning tasks to individual team members, collaboration between team members and creating sprints from the backlog of tasks.



# Junit Testing

Junit is a Java library which helps automate the running of Unit tests. It is available to run inside the IDE Eclipse and provides clear visual feedback on the outcome of the tests. In this project we created one Junit test, on the “MongoConnection” method. From this example we can see how automatic tests can be run for every build of the system.



# JRAT

Within this project JRAT should have analysed the code by inserting two lines of code. Unfortunately, after much failed initialisation attempts it was decided by the team to abandon efforts for time restraints but this would have been the desired method.

java **-javaagent:shiftone-jrat.jar** [your java ops] [main class]

The first line is inserted, is a single java command that is placed in the code to launch the program. Now the user should run the code as much as possible and iterate as much of the functions coded as possible, so that JRAT give as much potential feedback and analysis as possible. After this exit/turn off the program and JRAT should produce a few files.

java -Xmx256M -jar shiftone-jrat.jar

These files are saved but are not human readable and to have to be viewed in the JRAT desktop app. These jar files are saved in the same container as the JRAT desktop app, as this is needed to open and analyse those files. Navigate to a subfolder called JRAT output and file->open one of the JRAT files.

# Scrum

This process occurred twice to three times a week for the team. Meetings often on Monday, Wednesday and Thursday. The scrum was used to evaluate the work used within that week and gave continuous feedback by asking 3 simple questions, what did a team member do? What problems they encountered? And what was that member going to do next?

These meetings were essential in developing the product, as it helped the group to work as unit, become integrated in each other’s work and planning the scope of the project in all areas from database, java backend and the Angular front-end making the new approaches and technologies easier to use.

# Collaboration

Our team met to work together in an effort to better improve our team work skills and collaboration on the project, we met at a minimum of once per week, generally for one to two hours in a library room. In the meetings we initially went over what roles would be assigned to each member, both development and testing.

Weekly we met in or Dev Ops class for a scrum, this was to aid in further collaboration on what has been done, what needs to be done and what do we need to get something done/why didn’t we get a particular task done.

# Assigned Tasks

Research is a vital part of development, to know what you are developing. Each member of the team was assigned to research health insurance web sites to see what the major similarities were between them, and to implement this into our product. Once the information was gathered and the team received helpful feedback from our lecturer, we moved onto development.

## Database Administration

Connor Doherty and Eoin Gallen decided to work on the database. As a team we decided that MongoDB was a good fit for our project as it uses NoSQL which they both were in the process of learning.

## Java Controller Layer

Hugh Boyle is our most experienced Java developer, so he decided that he was comfortable to manage the java middle layer.

## Front End

Kieran Thompson and John Kelly developed the front end of the website collaborating together using the Angular Framework and Bootstrap.

# Collaboration tools.

## Git

Git version control was used by each member of the team to update and store new versions of the product on GitHub a cloud solution for local git repositories. Each team member had a local version stored on their machine using the git command seen in figure 1.

*git clone* [*https://github.com/hughieboyle/dawg2.git*](https://github.com/hughieboyle/dawg2.git)

***Figure 1***

Once the project was cloned into each team members own repo, each member could add their work to project using the commands seen in figure 2.

*git add .*

*git commit –m ‘appropriate message here’*

*git push origin master*

***Figure 2***

Each team member was able to create their own branch where they could work on their own tasks and test it before pushing it to the master branch this was done using the commands seen in Figure 3.

*git branch branch\_name*

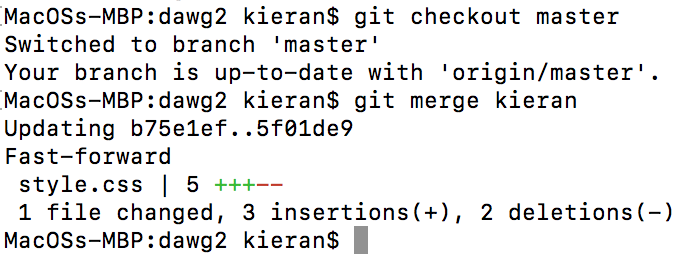
*git checkout branch\_name*

***Figure 3***

Once content with the changes made and that they have been tested, the user can then merge their branch with the master branch to be made as a permanent change, this can be done by the commands seen in figure 4

*git checkout master*

*git merge branch\_name*

**

***Figure 4***

Once finished with the branch the user can then choose to delete it using the following command seen in figure 5.

*git branch –d branch\_name*

***Figure 5***

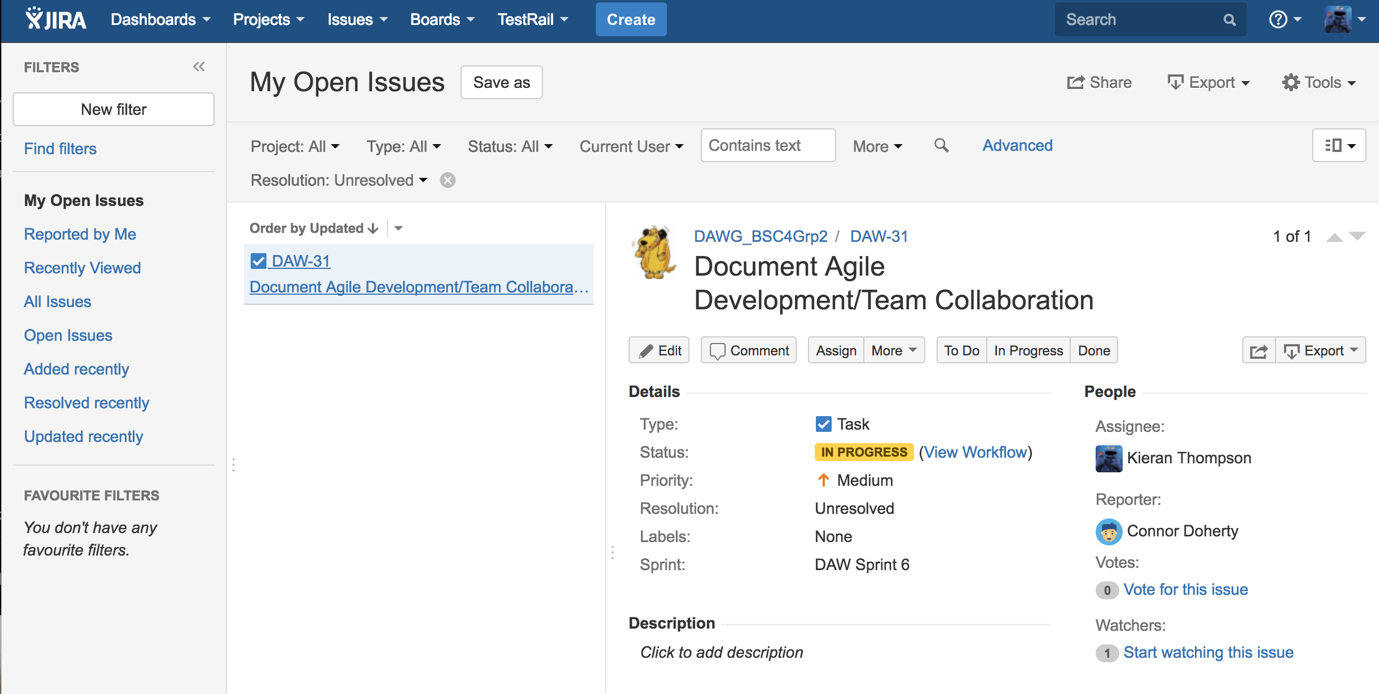
## Jira (team collaboration)

Jira is a collaboration tool for teams to track progress, bugs and tasks within a project, it’s used for project management and issue tracking.

Our team used Jira to assign tasks and manage them, we used the tool to start sprints and to share resources like learning and training.

Weekly after our scrum meetings we would then assign the new tasks to one another and start the next sprint, each one lasting a week.

An example task can be seen below in figure 6



***Figure 6***

# Photos of Team Collaboration

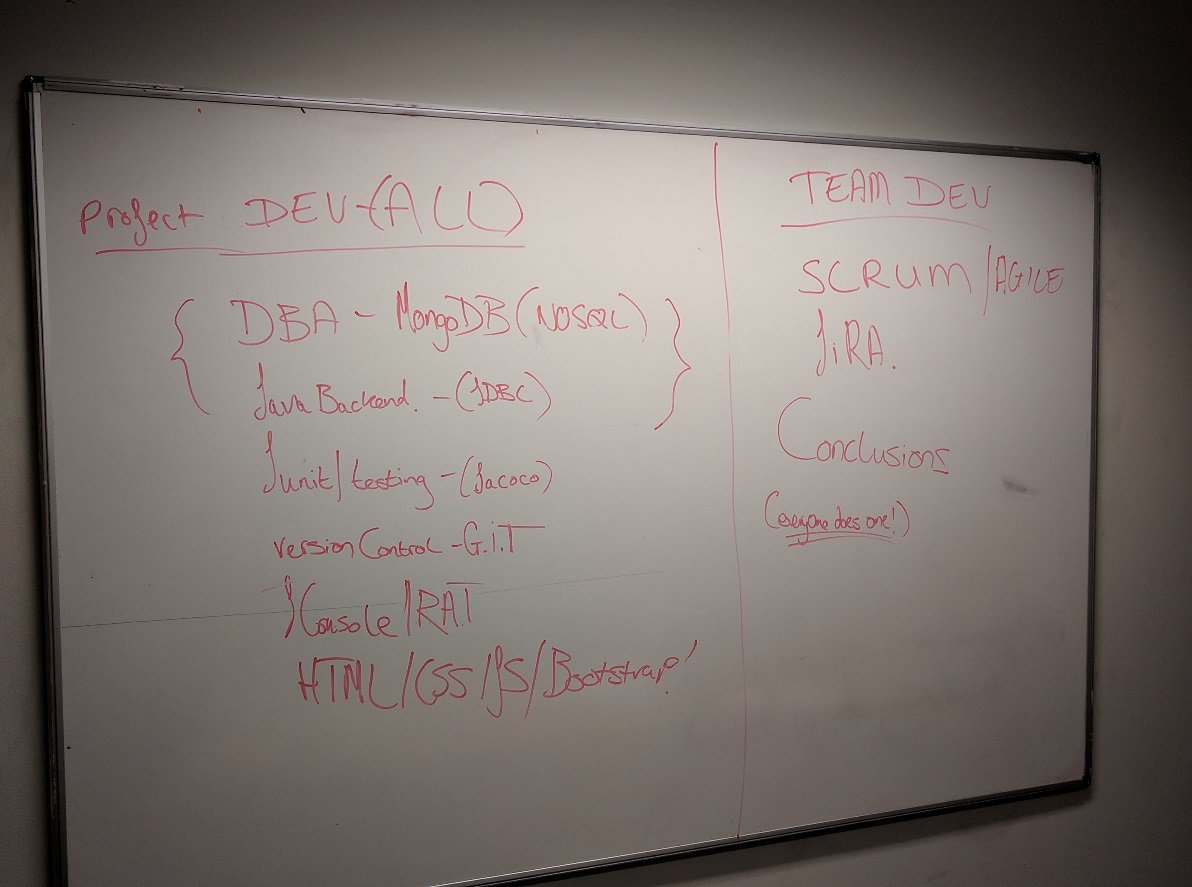
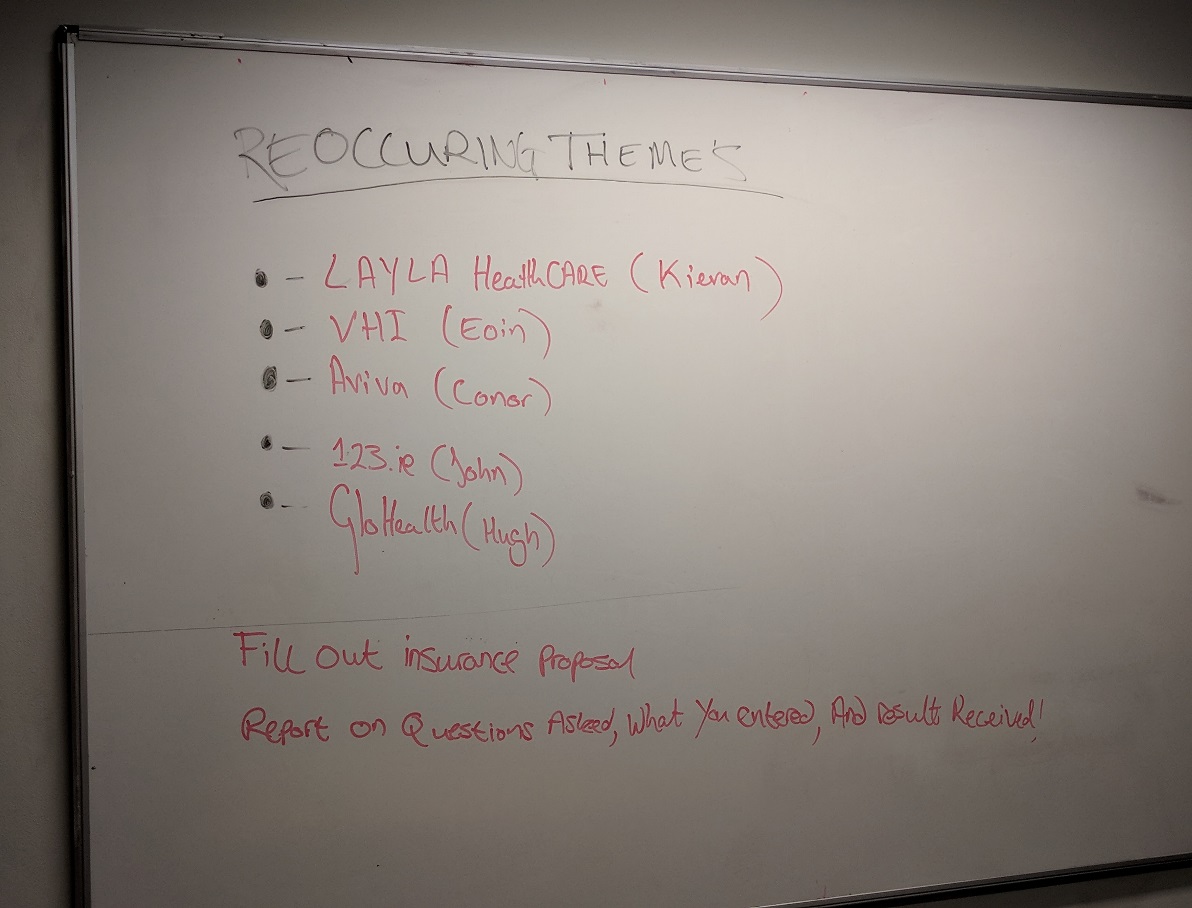
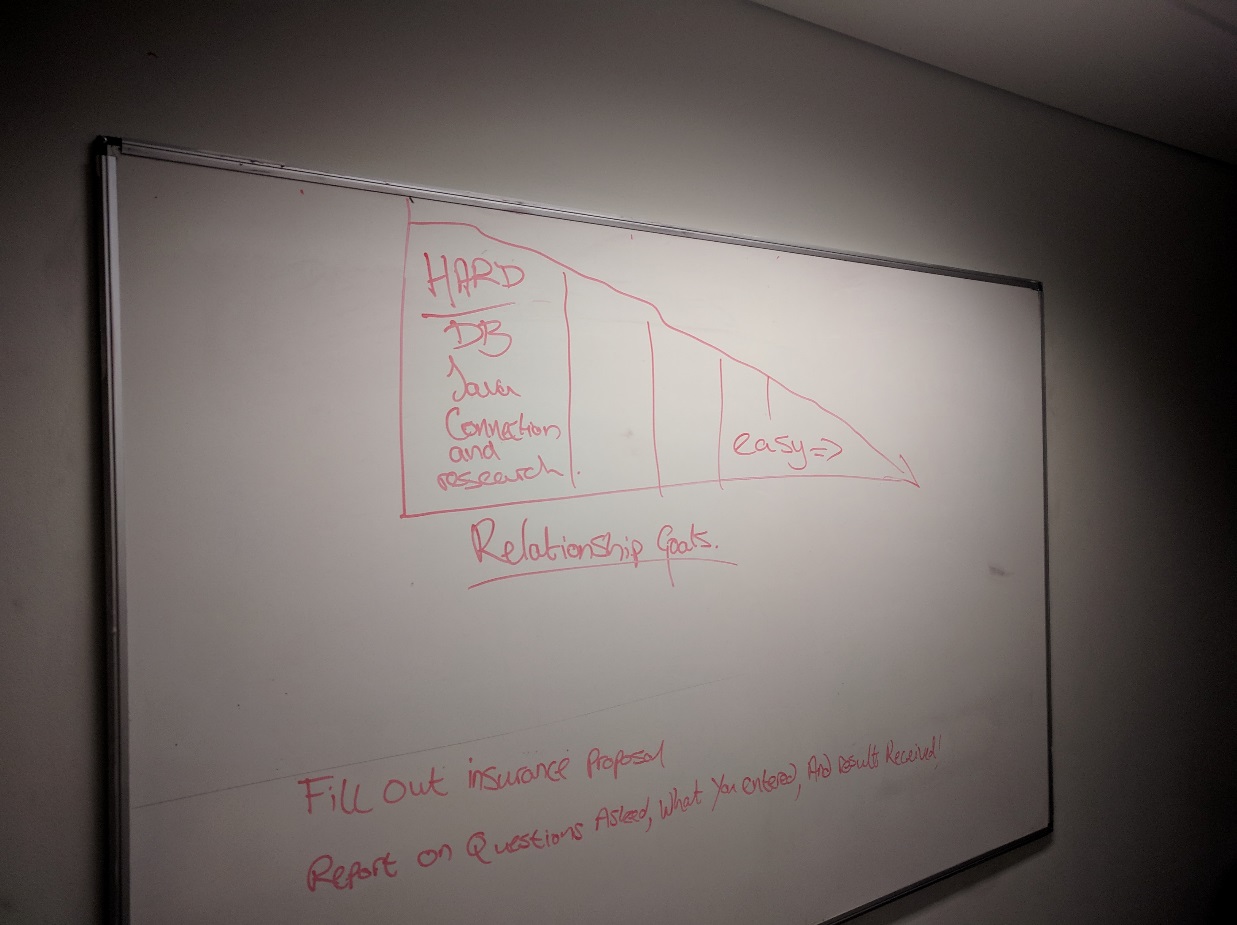


Image of some of the tools and technologies used to help implement the product.



Product Research



# Dev Ops Project Individual Conclusion

**Hugh J Boyle Conclusion**

We made numerous errors on this project.

* We jumped headlong into the different technologies required for the project without a proper planning meeting with the Product Owner who would have been able to limit the scope of our research and workload.
* We did not synchronize the different individual tasks. This meant that while some people had finished their tasks early in the week, others left theirs until the weekend. This is not meant as a criticism of individuals as each member of the team has to manage the workload for five modules over this semester as they see fit. The result of this lack of synchronization was that a sprint, which should have taken two days to complete, instead took a full week. I imagine this would be less of a problem in the real world as teams would spend the majority of their time on a single project.
* We did not have enough redundancy for team members carrying out their individual tasks with the result that if a team member fell ill, the task was not completed on time.
* Unfamiliarity with tools such as Jira and GitHub meant that these were only fully utilized towards the end of the semester.
* Both the Manifesto and Principles of Agile development emphasize the importance of skilled and experienced individuals in the agile team. Having a team composed of five people who were completely new to Agile development is not a recipe for success.

There were some positives.

* There was good communication in the bi-weekly meeting that were held outside class. They were well attended and everybody shared information and ideas in a positive environment.
* All team members were eager and willing to do their share of the workload.

My role in the project was mostly as the back-end Java developer. I wrote a method to connect to the MongoDB database developed by Connor and Eoin. I then wrote a method to set the quote amount, based on variables chosen by the user, and another method to return the quote amount, to be displayed to the user, on the webpage. I set up an example of a Junit test in the Eclipse IDE and also enabled JaCoCo for code coverage of the test. I wrote the Class Diagram for the project pushed all these to GitHub.

**Connor Doherty conclusion**

As part of the group project, I was assigned the job of creating the original database for the project. The database was to be a NoSQL database, and this was created using MongoDB. The database created held the fields required for the final project such as the personal details, quote etc and was stored with the appropriate data types. Having never previously used MongoDB I first completed the tutorial provided which gave me a basic insight into using this technology. Following this, I was able to set up a basic database and continuously add to it throughout the course of the project. I then attempted to host the database online using MLAB through the cloud hosting service AWS (Amazon Web Service), but unfortunately we could not establish a connection and so proceeded to export the database into a ‘.JSON’ extension file. This file was then pushed to the GitHub repository where team member Hugh was able to download the file to his laptop. We were then able to test the functionality of the database by creating a java test connection to ensure all data was working correctly. The test involved adding, editing and removing data from the database using a Java programme to do so.

As part of my contribution to the group, I also produced research documents and conclusions on Aviva healthcare in order to provide an estimate of the mathematical algorithms we could use to determine the cost of each policy type. Additionally, I also made use of Jira as a mechanism for adding issues/tasks to the backlog so as they could be pushed to the next active sprint for completion. Jira was essential for this project due to the fact that the group wasn’t always in each other’s company and this then gave us the opportunity to keep track with each other’s progression in terms of issues completed and issues in progress. After completing a task, I would mark it as ‘done’ and in addition to this, I would push any documents or databases created to the GitHub repository so fellow team members could view all work completed.

GitHub provided us with a medium to upload files and documents to a file exchange where we were able to view all files and keep track of the project. I also forked from the repository to enable me to work in parallel to everyone else, I would then merge and commit any documents I produced to the master branch, so my fellow team members would be able to view my changes/commits. I was also able to pull all other members work from the repository to my desktop in order to view it in the appropriate format.

Overall, I felt we worked well as a group and we had no problems with communication due to holding weekly meetings, as well as keeping up to date via Jira, GitHub, email and of course during our DevOps class time. The SCRUM meetings each week provided us with an opportunity to find out what each member had done, what struggles they may have had and what they intended doing next. The next issue they were going to complete was then added to Jira as a part of the backlog. From a personal point of view, I feel I could have made use of GitHub’s features earlier on in the project such as merging, forking and committing from the command line. I also could have spent more time on the production of the database as I had several data types incorrect, this would have saved me time having to edit and reupload a new database. Despite this I felt we worked well together as a team and learned a lot from the module and now have a decent understanding into how to operate as part of a team adapting an agile methodology. Hopefully, this will bode well for all team members in the future.

**Eoin Gallen conclusion**

My part of the group project was to create a MongoDB database which could hold the data entered from the front end of the project which was passed through via a Java connection. I and Conor worked together for this as we were assigned to the database creation. The first task we set about doing was as an entire team we worked on what we needed for the database and what kind of information will it store. As I never previously used MongoDB before I went and did some research on what Mongo was. MongoDB is a NoSQL database. Instead of using [tables](http://searchsoa.techtarget.com/definition/table) and [rows](http://searchoracle.techtarget.com/definition/row) as in [relational databases](http://searchsqlserver.techtarget.com/definition/relational-database) like SQL which is an alternative. MongoDB is built on an architecture of collections and documents. Documents include sets of [key-value pairs](http://searchenterprisedesktop.techtarget.com/definition/key-value-pair) and are the basic component of data in MongoDB. Collections contain sets of documents and function as the comparison of relational database tables. Many companies are using MongoDB, as it represents a solution for a range of big data challenges.

To first find out how to work and install Mongo locally on my machine I completed the lab. At that point, I also implemented a small test database to make sure I understood what I was doing. Then I went about creating the database for the project, which had the fields of the all the customer's personal details, occupation, medical conditions and their quote. I used the command prompt for the creation of the database. Once I completed creating the database I used RoboMongo which is a GUI which lets you view all the data stored in your database. You can also remove, add and update fields within this. I then exported my file and uploaded this to GitHub, so Hugh could use it to make the connection to the front end with Java. On GitHub, I have uploaded 3 file types of the database (bson, .csv and. json).

I also gave Hugh a hand with the java connection as he had to import the MongoDB to work with the front end. So that the customer entered their details this would be saved to the database and the Java would work out their quote based on what they entered. Then they would receive their quote.

Throughout the project, I also used Jira for adding tasks I would be doing for the sprint period. I found this very helpful as it gave a target to complete the piece of work needed to do. After completion of the task, I would upload any relevant documents to GitHub. Jira was a very useful tool for the project and gave guidance as you could view the tasks the rest of the group had to complete.

All work completed was pushed to GitHub. I made my own branch on GitHub to push my work onto. I used the GitHub website to push the work completed. I also uploaded any documents completed relevant to the project. GitHub was a great tool to use as the forking feature let us work in parallel development. Also, you could pull everyone else’s work to see how things were progressing.

All communication was done via email and in our weekly meetings. We had a group scrum twice a week to let all members of the group know what we had done, what stopped us and what we were going to do. Also in the meetings, the work would get assigned for the week and this would have been put up on Jira as a sprint.

**John Kelly Conclusion**

Within this module I learned and developed a lot of new and key skills, that in my own believe aided my own personal growth, new and useful technologies that are used widely in industry and how collectively work within a team and simulate a real world software development task.

Within the product development a list of new and interesting technologies where used, I helped develop the front-end development of the project and was tasked with adopting a new and fun approach, ANGULAR. Angular is a framework that helps implement a dynamic web apps. I had used HTML and CSS previously and had a high proficiency in both but I wanted to help develop my range of skills further and adopted angular framework into the project, by doing this I used bootstrap, typescript, HTML and CSS, these all give the front-end a much more detailed and in-depth appearance, and also a much more appealing visualisation to the product user. Typescript and bootstrap were new concepts to learn but I believe although a learning curve was necessary, they were much easier to develop in comparison to a website solely coded in HTML and CSS. The frontend was connected to a java backend which referenced the classes and then communicated with the MONGODB. These 3 software types implemented smoothly together. Mongo dB was a new concept in addition to bootstrap and typescript. Mongo dB is a NOSQL language and has a lot of added benefits over the MySQL skills I had before, two of the most obvious being its faster at reading operations and more dynamic in development.

In addition to the technologies used within the team project a lot of team collaboration skills and real-world industry experience was gained.

The team adopted the AGILE approach for the project over the traditional waterfall method. This approach aided in software development, reduced paperwork, and communicates with the product owner more efficiently adapting and making changes throughout with the software development aspect a priority.

Adaptive processes where iterated throughout the production, the team members communicated with each other very regularly and with the product owner to achieve the best possible results.

Engagement with the product owner was a necessity for a more accurate representation of their desired product. Meetings would happen weekly and on two separate occasions the product owner came and evaluated progress in production and implementation of the website.

The concept of Scrum was new to all of us and at first it was a difficult to work it correctly into the product, which showed in our production at that early stage. After a few weeks and between 2-3 scrums a week it became an easy and fluent process.

In addition to this the use of Jira helped in planning and detailing tasks for each individual team member. Each team member could see each other’s tasks for that time, issue tasks and complete tasks. This constructive method of planning aided in the software development cycle.

Another thing I liked about this subject was the use of new tools within the product development for example Jira already mentioned but JRAT, Jacoco, GIT and Junit.

These tools helped make the software as efficient as possible without the guidance of lecture and helped give an experience of tools used in industry for a software process like this.

Overall I enjoyed my experience simulating an industry process and feel I gained a lot of useful knowledge for my own personal development moving forward. Although I do believe time was a major factor in development I was happy with experience and knowledge gained.

**Kieran Thompson conclusion**

To conclude, Git and Jira are industry standards for a reason, they work very well and strengthen collaboration in a team. Jira provides an easy to use, friendly user interface that incites the user to interact with it. It gives team leads and project managers the information they need in regards progress and remaining tasks without having to chase anyone down for information. Git allows teams to get immediate updates on the project, this is far more convenient than file sharing between users, the fact that cloud solutions like GitHub and Bit Bucket exist make it even more convenient, allowing developers at home to work on the product if the workplace allows it. With tools like automation, the development process can be further improved with git using tools like automation, a developer can have their environment set up that it pulls the latest updates immediately on start-up of the computer, improving workplace efficiency. One of the main points of Git is version control, the idea that if you make a mistake so damaging that the system is beyond repair or the bug will take too long to fix, the developer can just roll back to the last saved version. The mentioned technologies have increased productivity drastically in our development process, at home, I personally was able to see the updates happening through GitHub, if I’d forgotten a task or wanted to check if someone else on the team had finished their own, I could check on Jira and see exactly where everyone stood in development, this helped immensely, It also stopped team members and myself duplicating work, saving time and improving the workflow.