Abstract

Documented evidence of the 13 duties, alongside the KSB metrics to achieve.

Dev Ops Engineer Apprenticeship

T. Rowe Price Portfolio

Arbnor Aljilji

Contents

[Introduction to myself, team, and project 3](#_Toc86847998)

[Duty 1 – Script and code in one general purpose language and one domain-specific language 4](#_Toc86847999)

[**B2**: Invests time and effort in their own development, recognising that technology evolves at a rapid rate 5](#_Toc86848000)

[**K1**: Continuous Integration - the benefits of frequent merging of code, the creation of build artefacts and ensuring all tests pass, with automation throughout - including common tooling. 6](#_Toc86848001)

[**K4**: The business value of DevOps in terms of Time, Cost, Quality, with an emphasis on building in internal Quality throughout the lifetime of the product. 7](#_Toc86848002)

[**K6**: A range of problem-solving techniques appropriate to the task at hand, such as affinity mapping, impact maps, plan-do-check-act/Deming 7](#_Toc86848003)

[**K7**: General purpose programming and infrastructure-as-code.  AND **S17**: Code in a general purpose programming language. 9](#_Toc86848004)

[Duty 2 – Initiate and facilitate knowledge sharing and technical collaboration with teams and individuals, with a focus on supporting development of team members 10](#_Toc86848005)

[**K19**: Different methods of communication and choosing the appropriate one - e.g. face-to-face (synchronous, high bandwidth), instant messaging, email (asynchronous, low bandwidth), visualisations vs. words. 10](#_Toc86848006)

[**K23**: The importance of continual improvement within a blameless culture. 10](#_Toc86848007)

[Duty 3 – Engage in productive pair/mob programming to underpin the practice of peer review 12](#_Toc86848008)

[Duty 4 – Work as part of an agile team, and explore new ways of working, rapidly responding to changing users needs and with a relentless focus on the user experience. Understand the importance of continual improvement within a blameless culture 14](#_Toc86848009)

[Duty 5 – Build and operate a Continuous Integration (CI) capability, employing version control of source and related artefacts 15](#_Toc86848010)

[Duty 6 – Implement and improve release automation & orchestration, often using Application Programming Interfaces (API), as part of the continuous delivery and continuous deployment pipeline, ensuring that team(s) can deploy new code rapidly and safely 17](#_Toc86848011)

[Duty 7 – Provision cloud infrastructure using APIs, continually improve infrastructure-as-code, considering using industry leading technologies as they become available (e.g. Serverless, Containers) 18](#_Toc86848012)

[Duty 8 – Evolve and define architecture, utilizing the knowledge and experience of the team to design in an optimal user experience, scalability, security, high availability and optimal performance 19](#_Toc86848013)

[Duty 9 – Apply leading security practices throughout the Software Development Lifecycle (SDLC) 20](#_Toc86848014)

[**B1**: Exhibits enthusiasm, openness and an aptitude for working as part of a collaborative community; e.g. sharing best practice, pairing with team members, learning from others and engaging in peer review practices. 20](#_Toc86848015)

[Duty 10 – Implement a good coverage of monitoring (metrics, logs), ensuring that alerts are visible, tune-able and actionable 21](#_Toc86848016)

[Duty 11 – Keeping up with cutting edge by committing to continual training and development – utilize web resources for self-learning; horizon scanning; active membership of professional bodies 23](#_Toc86848017)

[Duty 12 – Look to automate any manual tasks that are repeated, often using APIs 25](#_Toc86848018)

[Duty 13 – Accept ownership of changes; embody the DevOps culture of ‘you build it, you run it, with a relentless focus on the user experience 26](#_Toc86848019)

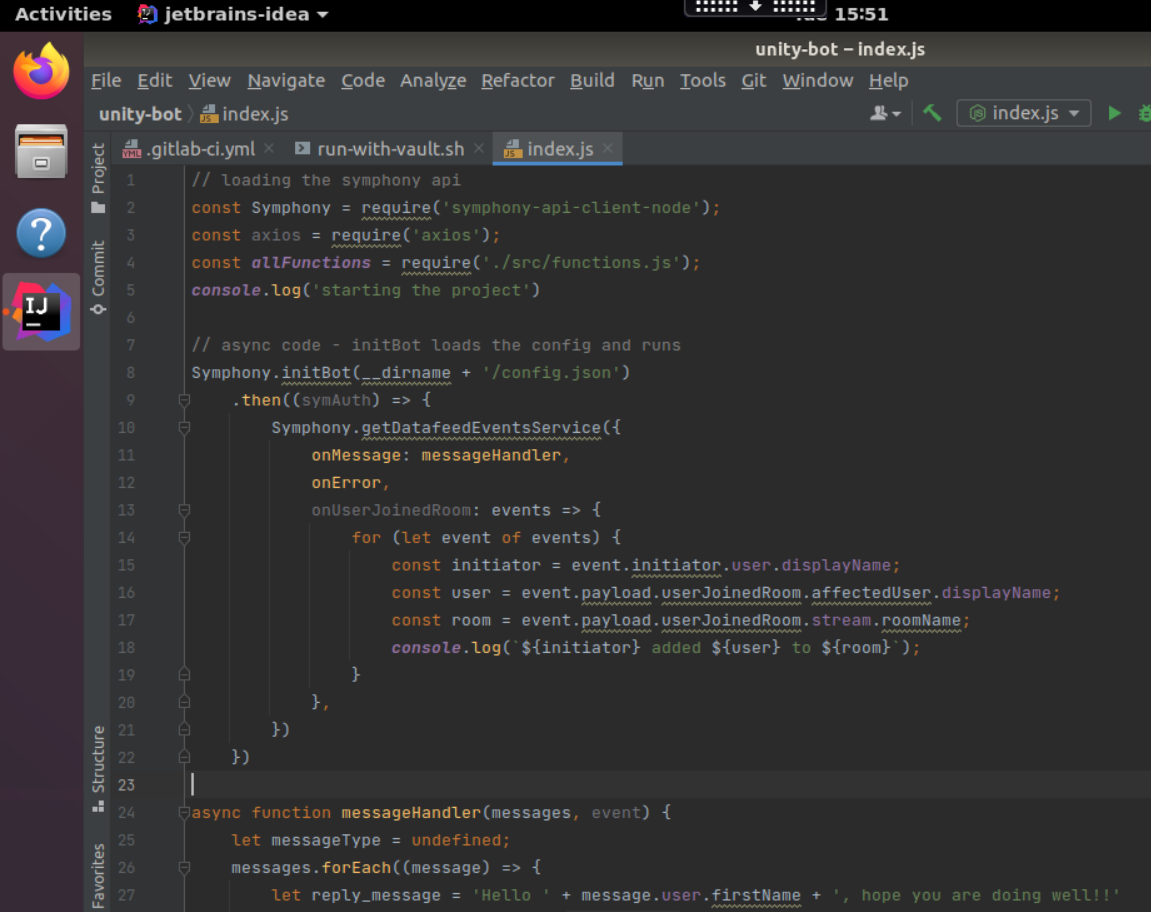
# Introduction to myself, team, and project

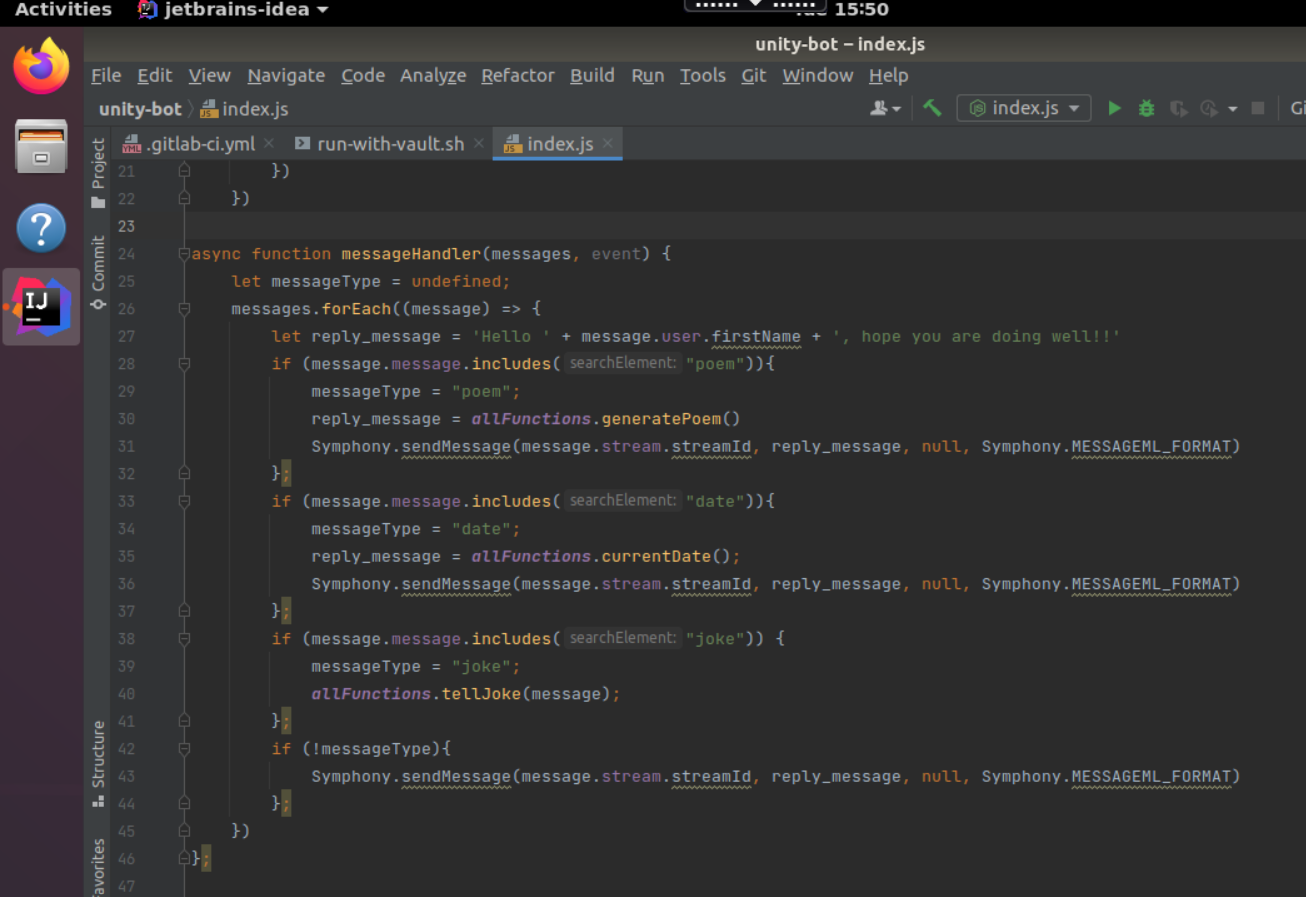
My name is Arbnor Aljilji and I’m part of the DevOps Engineering Apprenticeship at T. Rowe Price. I’m 28 years old and chosen to switch careers from Human Resources to Tech. I first studied BTEC IT in college and graduated in BA Hons Human Resources but I always had an interest in tech; AI, to electric vehicles, to gaming development, and my interest in technology increased heavily throughout the pandemic. I had a lot of free time to learn programming languages and creating very small math quizzes with codecademy etc. However, I didn’t want to tackle software engineering directly as technology itself is too big to know what I want to progress into so I researched the different paths a new ‘self-taught’ developer can go into. There were different options from Cybersecurity, Software Engineer, Cloud Engineer (DevOps), Data Analyst etc. After doing research and comparing the pros and cons between the different roles, DevOps itself stood out more to me as I’m also fascinated with Cloud Computing and the use of cloud computing is still increasing.

I am part of the T. Rowe Price Developer Services Group in London. We are a team of 9 Software Engineers, and 1 Product Manager that have different specialty skills as developers, SDETS and SREs. Our team looks after the developers who design apps, and our mission is to enable the developers to move as quickly as possible and enable them to push cleaner code at great frequency, and higher volume. Through automation around development tool chain, release management, software quality, builds, and developments, we aim to achieve these goals. Unity is an in-house product designed and looked after by the DSG developers to allow other developers to deploy their application to the cloud, at a much quicker pace.

The project I am currently working on is a chatbot designed to automate any toil that our developers may experience. It will be available to only the DSG London and US team, which is a group of 20 engineers until further expansion. The chatbot itself will handle messages and requests from clients and respond back with an answer or resource. Many of our clients have repeated manual tasks such as opening a browser, writing a link to check a specific merge requests, whereas the bot can notify the user with the merge request link upon request and the status of the merge request. There are other designed functions that myself and a project lead are working on for the bot to complete, as this will remove any toil the team undergoes. The value this chatbot project will bring to the team is by saving themselves couple minutes a day, to several hours a week in total of typing and clicking in which the bot can perform at a higher rate with a simple question. As well as this project, the team and myself also provide Linux Desktop Support for developers.

# Duty 1 – Script and code in one general purpose language and one domain-specific language





Above is the code for the index.js file which needs to be run to initiate the bot in the symphony-test environment. This will run locally to where I can test the app itself. It is written in NodeJS. Line 2 – 4 requires libraries and files which are to be used in the Index.js file. Line 24 – 46 is the function that listens to the user input in the Symphony chat, and it’ll respond according to the input.

(Domain-specific language evidence to be shown in next submission.)

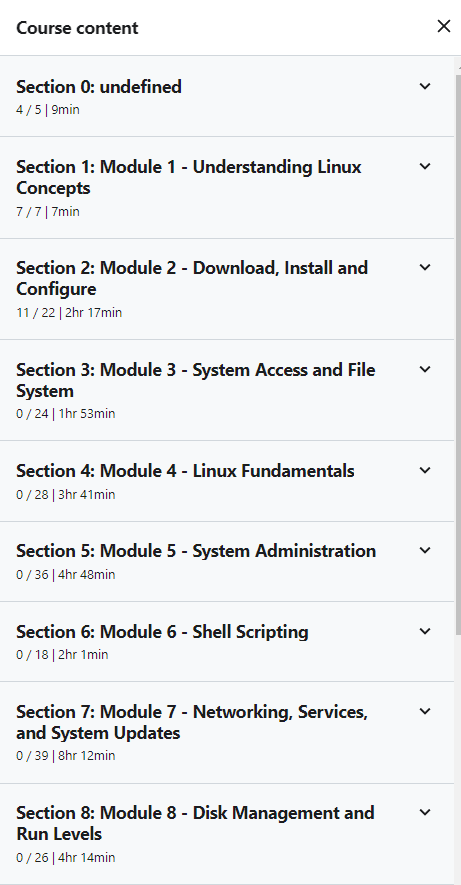
Ksb to cover: (K4 K5 K6 K7 K8 K11 K12 K13 K14 K20

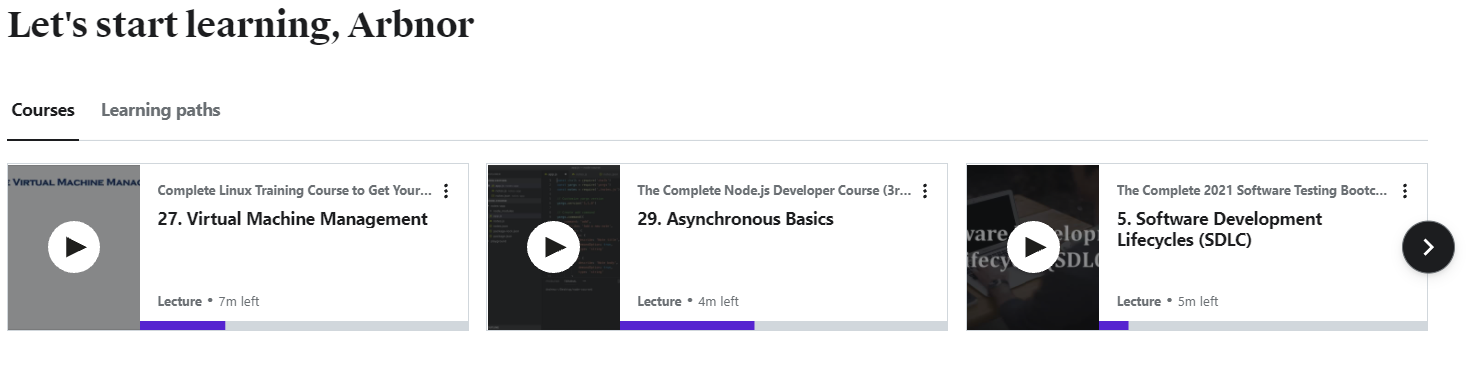
S9 S11 S13 S14 S16 S17 S18 S20 S22

B2 B3)

## **B2**: Invests time and effort in their own development, recognising that technology evolves at a rapid rate

As part of my role, I am also in the support for Linux Developer Desktop team. To learn LDD, I spend some time learning about Linux on udemy course: ‘Complete Linux Training Course 2021’

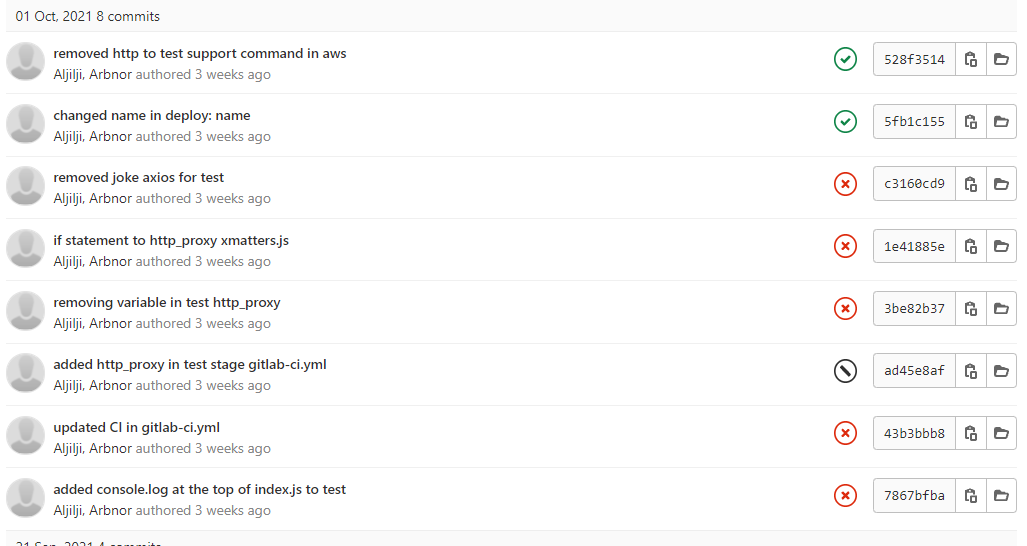




I also learn about Javascript/NodeJS (especially with ES6) on udemy. I also use ACloudGuru to further improve my knowledge on AWS SAA, and YouTube account ‘TechWorld with Nana’ for DevOp Tools.



## **K1**: Continuous Integration - the benefits of frequent merging of code, the creation of build artefacts and ensuring all tests pass, with automation throughout - including common tooling.



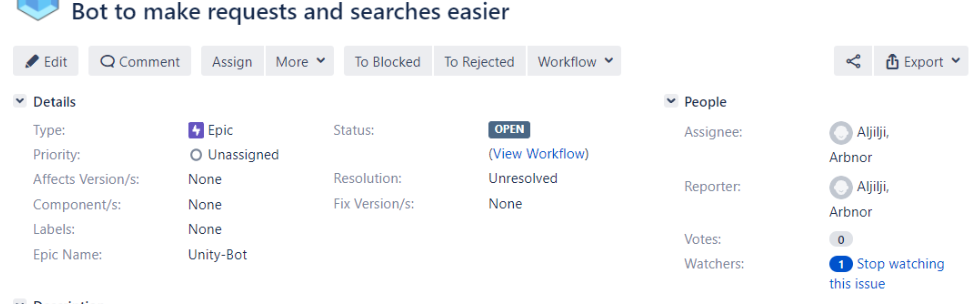
Above is a screenshot of frequent merging of code. The benefit of doing this is that with smaller code changes, it will be simpler and easier to handle rather than huge amount of code, and as such, will have fewer issues that may need to be fixed in the later stage. With smaller merged code, you could test as soon as they are integrated, which will highlight any errors before too much work is completed after. More points cover such as: fault isolations, Faster Mean Time to Resolution, More Test Reliability, Quicker Release, Less Backlog, Customer Satisfaction, Reduce Cost, Easy Maintenance and updates.

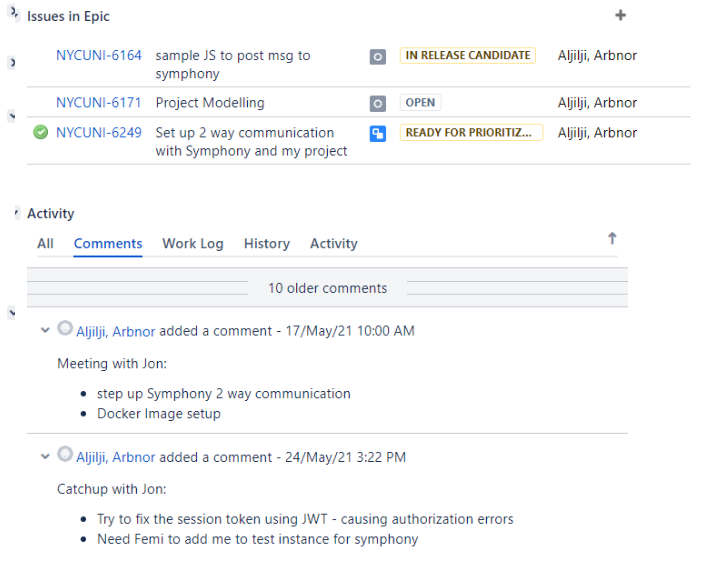
## **K4**: The business value of DevOps in terms of Time, Cost, Quality, with an emphasis on building in internal Quality throughout the lifetime of the product.

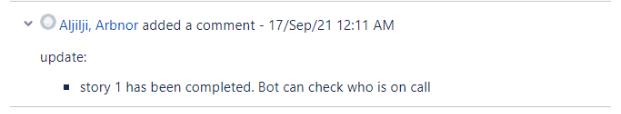
As mentioned in K1, I mentioned further importance of business value of DevOps in terms of time, Cost, and Quality. There several remaining business values could be such as it can reduce cost and easy to maintain and update. For example, automation in the CI/CD pipeline reduces the number of errors that can take place in many repetitive steps. This gives the coder free time to spend it on product development, as there isn’t much code change errors, nor the tests failing or picking up any errors quickly. The quality of the product also improves as testing on small code changes can pick up any further bugs found, thus, making it more of a flawless product to meet the customer satisfaction.

## **K6**: A range of problem-solving techniques appropriate to the task at hand, such as affinity mapping, impact maps, plan-do-check-act/Deming

Even though I understand the use of affinity mapping, I take a different approach. In my project, we use Jira and create tasks which handles the task needed to be completed, the planning and issues facing. For example, this Epic story below contains other tasks within the main task itself ‘bot to make requests and searches easier’ – so my team know it is a bot design task. However, within the Jira ticket, we can assign new issues within the Epic story, such as setting up the 2- way communication. Unlike using affinity mapping, I write all the completed tasks, and difficult tasks in the comment sections within the main epic story, as seen below. This only gives a clear outline that part of my project is completed, and I can now move into creating a new sub-task within that main task. This method of problem-solving helps me understand what I need to do in my project, what I am missing and what I have completed.

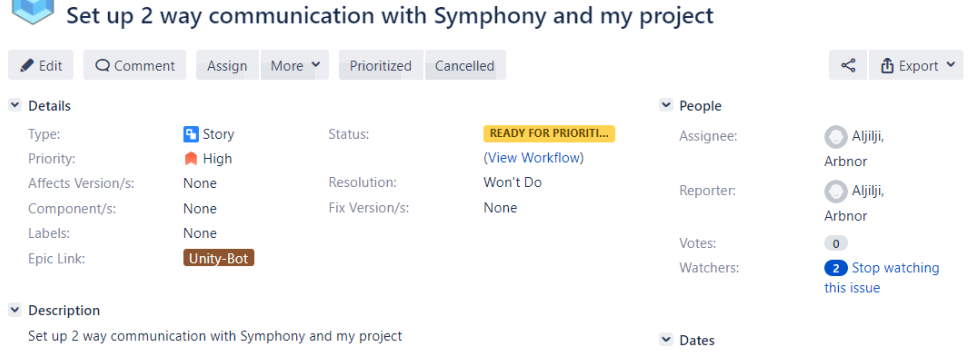




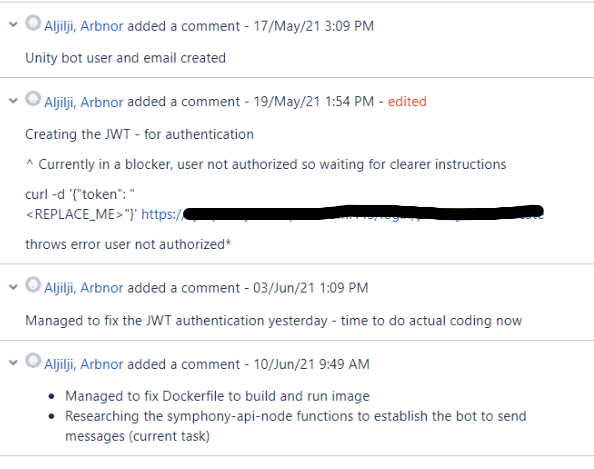


(Below is continued K6)

The screenshot below will show me clicking into NYCUNI-6249 which is an issue within the Epic story.



Now within this task, I know what I must do as I have created a task for it, and the issues I am currently facing, as I have inserted it in the comment section underneath.



## **K7**: General purpose programming and infrastructure-as-code.  AND **S17**: Code in a general purpose programming language.

I don’t really need to do infrastructure-as-code as Unity Deploy (our own internal app) already has that in place for us. However, when building my own bot app, I have coded it in NodeJS as seen in snippet below.



# Duty 2 – Initiate and facilitate knowledge sharing and technical collaboration with teams and individuals, with a focus on supporting development of team members

Within my team, we have a mixed skill set of individuals. We have SDETs and SREs who are experts at software engineering. I have the luxury of setting up meetings between each SDET and SRE, to learn from them how to write tests better, how to look at Prometheus or Grafana and understand the metrics. This helps me with development of learning more about the SDLC, but in general about cloud computing, and software engineering overall. I get a lot of support from each member if needed. We as a team, but individually, have access to a lot of inhouse documents written by me and the team. For example, if a user wanted to simply use NPM/NODEJS we have a document for that for the Linux Developer Desktop user. Majority of our documents are found in confluence.

Ksb to cover: (K4 K9 K10 K18 K19 K23

S1 S2 S4 S8 S16 S21

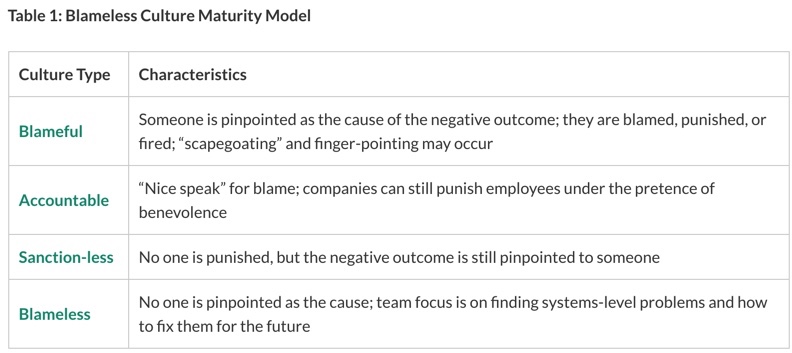
B1 B2 B4)

## **K19**: Different methods of communication and choosing the appropriate one - e.g. face-to-face (synchronous, high bandwidth), instant messaging, email (asynchronous, low bandwidth), visualisations vs. words.

Every week I have a 9.30am meeting for 1 hour with my manager on my weekly catchup. Discussion covers the project, current tasks and potential future tasks, any blockage I have, and other activities I have been working on. Later during the week on Friday, the whole DSG London Team have a stand up at 10am to discuss support that we have accomplished on Linux Dev Desktop throughout the week – any new unknown errors, or what complications occurred on support for the week. I also use Symphony (like slack) a lot to do one-on-one messaging, which then escalates to me inviting the other user to Zoom. When it comes to learning new skills or knowledge at work, I set up Zoom meetings as I prefer learning visually, as unknown jargon can confuse me, or the internal document may be outdated.

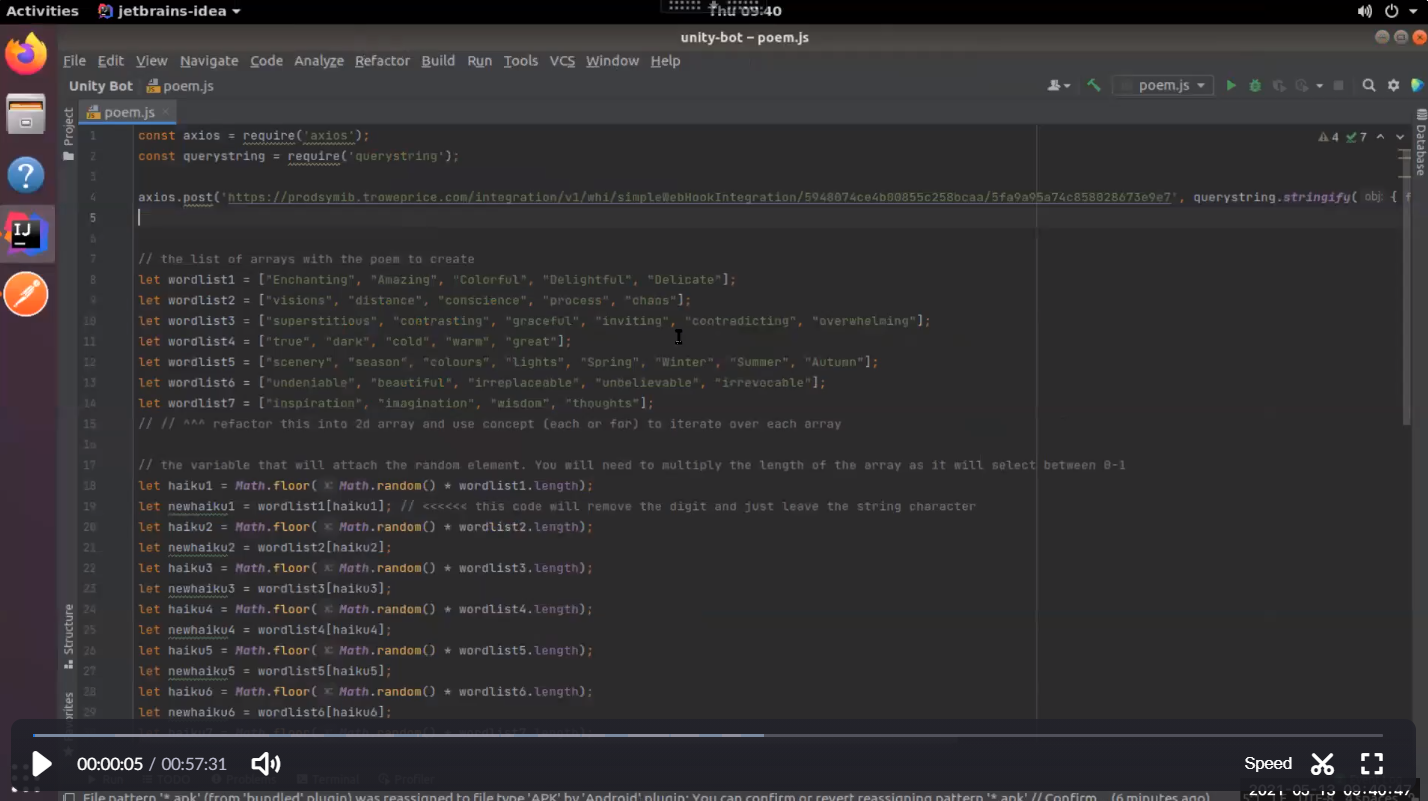
## **K23**: The importance of continual improvement within a blameless culture.

Within an organisation, I find that there are different tiers of ‘blame culture’. Below is the Blameless Culture Maturity Model:

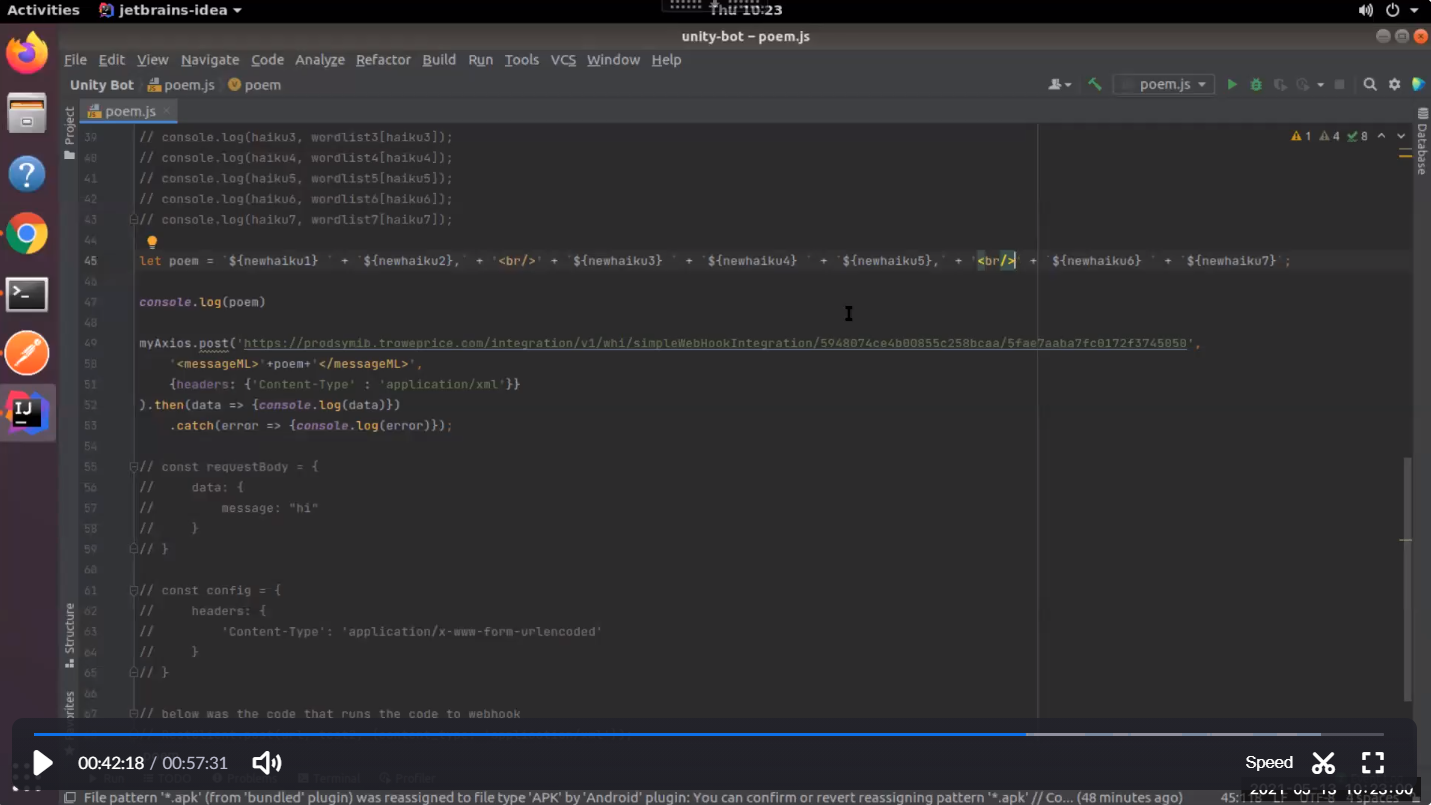


The idea of a blameless culture helps use find solutions, rather than putting blame on an individual. Individual contributors feel safer, eagerly report failures and rather seek solutions at an earlier step, than leaving it too late or creating scape goats. By continually improving, we get less errors occurring, stronger team bonding, and people feel more motivated to do work.

# Duty 3 – Engage in productive pair/mob programming to underpin the practice of peer review



This is a Zoom recording of me and my project lead Ed Rowley pair programming and reviewing my code. This is the start of the video where Ed looks at my code and corrects and we refactor my code.



The screenshot above is the refactored code after 40 minutes of pairing and coding. The result is that we managed to change some of my code, and change drivers in the coding.

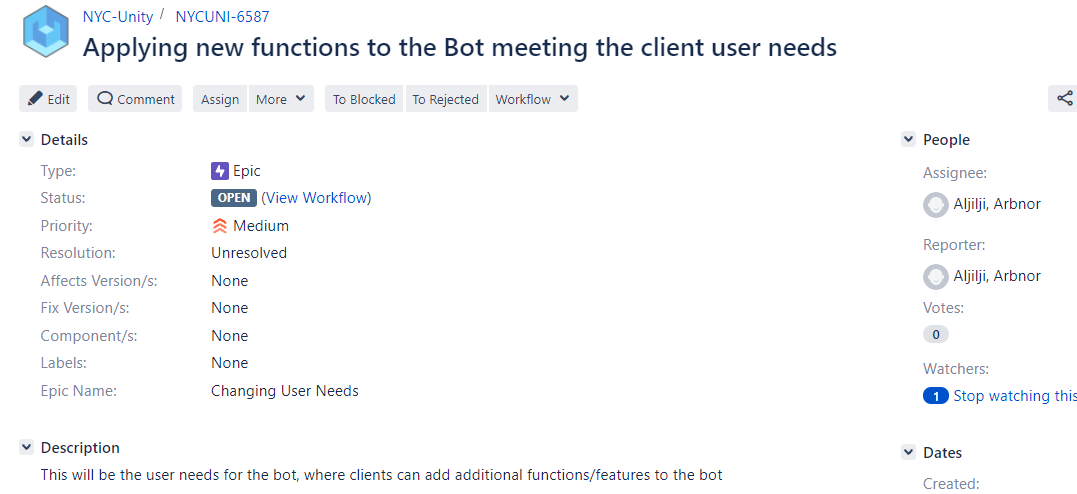
Ksb to cover: (K4 K6 K7 K9 K20

S1 S8 S13 S14 S17 S18

B1 B3)

# Duty 4 – Work as part of an agile team, and explore new ways of working, rapidly responding to changing users needs and with a relentless focus on the user experience. Understand the importance of continual improvement within a blameless culture

I had a Zoom meeting with my clients on the current progress of the project and received feedback on new needs they want on the project. I was able to take the response from the user need and apply this to the project needs, and just waiting for implementation of the code. I was able to add this feedback to a Jira task for myself. The user experience is important to me as it would be them using the app.

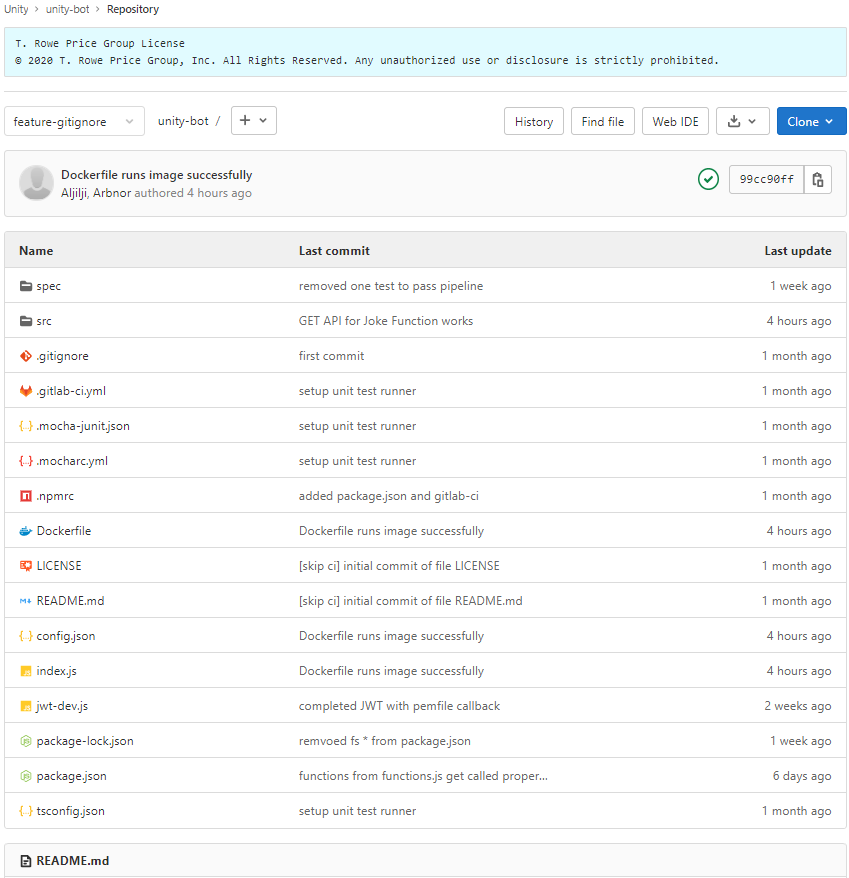


Ksb to cover: (K4 K6 K9 K10 K18 K19 K22 K23

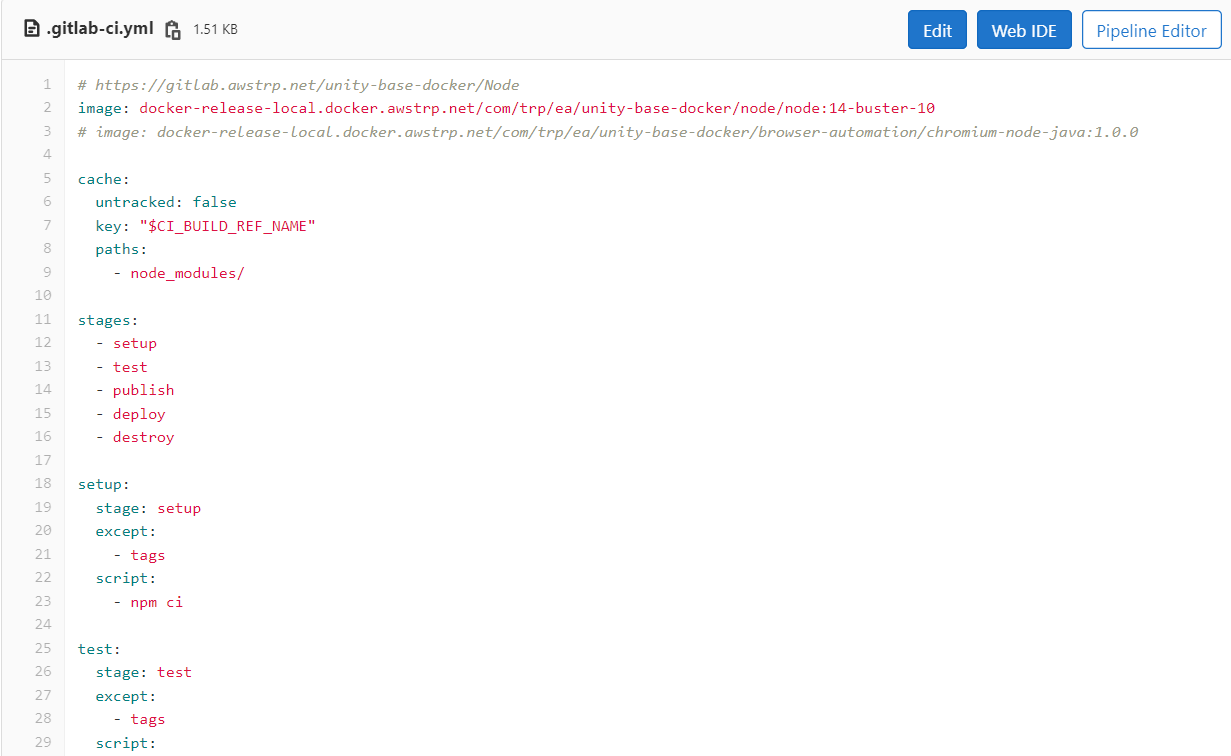
S1 S2 S3 S4 S8 S11 S21

B1 B3 B4)

# Duty 5 – Build and operate a Continuous Integration (CI) capability, employing version control of source and related artefacts



Above is my version source control for my project Unity-Bot





Above is my gitlab-ci.yml file which sets up the CI/CD pipeline in gitlab which consists so far of setup, test, publish, and deploy.

Ksb to cover: K1 K2 K7 K14 K15 K24

S4 S9 S12 S15 S16 S18 S20

B3

# Duty 6 – Implement and improve release automation & orchestration, often using Application Programming Interfaces (API), as part of the continuous delivery and continuous deployment pipeline, ensuring that team(s) can deploy new code rapidly and safely

Pipeline update: - when pipeline completes, calls API to Unity Bot and notif a user with the completion – maybe using a previous webhook   
  
(maybe John Singh – add additional decorations to MR) <<< after Unity bot

Ksb to cover: K1 K2 K7 K15 K17 K24

S4 S5 S9 S12 S15 S16 S18 S20

B3

# Duty 7 – Provision cloud infrastructure using APIs, continually improve infrastructure-as-code, considering using industry leading technologies as they become available (e.g. Serverless, Containers)

FarGate as a service <<<

(use lambda?)

Unity Deploy Open Source contribution model < Continually improve IaC << using the confluence page that I saw with Ed for terraform tests etc

Maybe Chris

Ksb to cover: K1 K2 K3 K7 K8 K11 K12 K13 K14 K15 K17 K21 K24

S4 S5 S6 S9 S12 S15 S16 S18 S21 S22

B3

# Duty 8 – Evolve and define architecture, utilizing the knowledge and experience of the team to design in an optimal user experience, scalability, security, high availability and optimal performance

Moving from ec2 to fargate < evolution progress (scalability, disk space, handling secrets (previously hard coded, by now it’s in Vault – authentication based on AWS IAM))

Ksb to cover: K1 K3 K4 K6 K7 K8 K9 K10 K11 K12 K15 K21 K24

S1 S2 S4 S8 S9 S10 S11 S15 S16 S17 S18 S21 S22

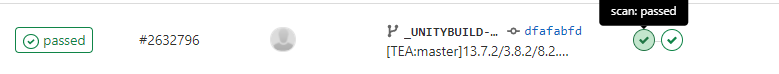
B1 B3

# Duty 9 – Apply leading security practices throughout the Software Development Lifecycle (SDLC)

1. We don’t expose secrets to pipeline (gitlab JWT to authenticate deploy)
2. We removed pem secret hard code, and managed outside its own secret context as seen below in Vault



1. All dependences are pulled via artifactory (trusted source), and artifactory scans the containers that are being produced, and checks for security vulnerabilities
2. TEA & Fortify -



Ksb to cover: K3 K4 K5 K6 K7 K8 K9 K10 K11 K12 K16

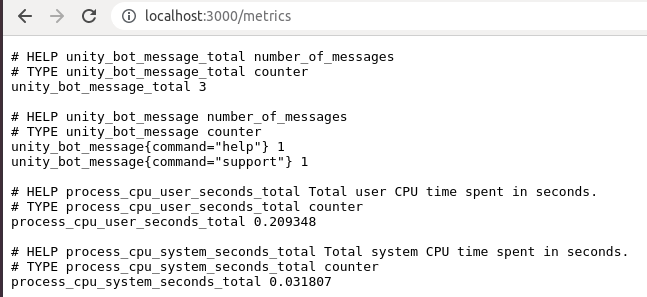
S3 S4 S9 S10 S11 S14 S16 S17 S18

B1 B3

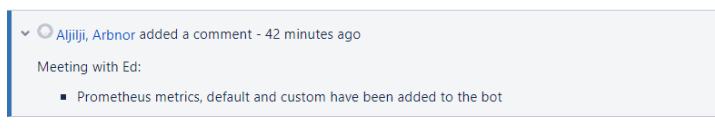
## **B1**: Exhibits enthusiasm, openness and an aptitude for working as part of a collaborative community; e.g. sharing best practice, pairing with team members, learning from others and engaging in peer review practices.

I intend to have meetings with my peers and ask them to show me systems within T. Rowe Price. For example, how to use Vault, to be able to store users and passwords and read back from it. I also have learnt a lot of security concerns within the SDLC such as not hard coding API access keys etc. I also learn a lot from my team when they discuss different issues in our Friday 10am stand downs. My team also what the best practice when it comes to creating a project, and how to identify its need.

# Duty 10 – Implement a good coverage of monitoring (metrics, logs), ensuring that alerts are visible, tune-able and actionable



I have progressed on implementing Prometheus metrics using Express for my bot. I currently use the default metrics but have also implemented my own custom metrics. For example, how many messages the bot listens to, and how many commands it should be listening to. E.g. “help” is a command it will respond to, but “foobar” is a command it will not listen to, however, I add that counter to the message\_total.



After completing this, I add it as a comment to my Unity project in Jira, so I know what stage in my project I am at. More metrics to be added via AWS

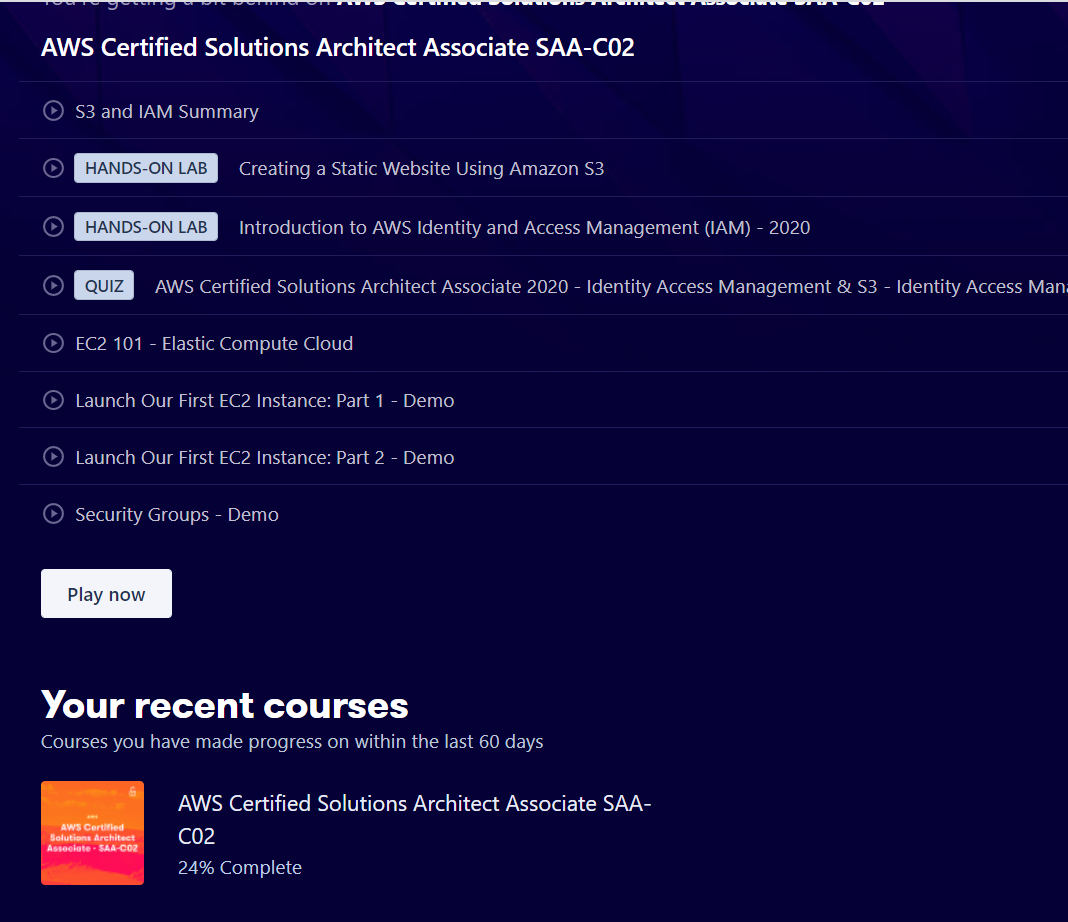
Ksb to cover: K4 K6 K7 K10 K11 K13 K24

S3 S4 S6 S7 S9 S11 S16 S17 S18 S19

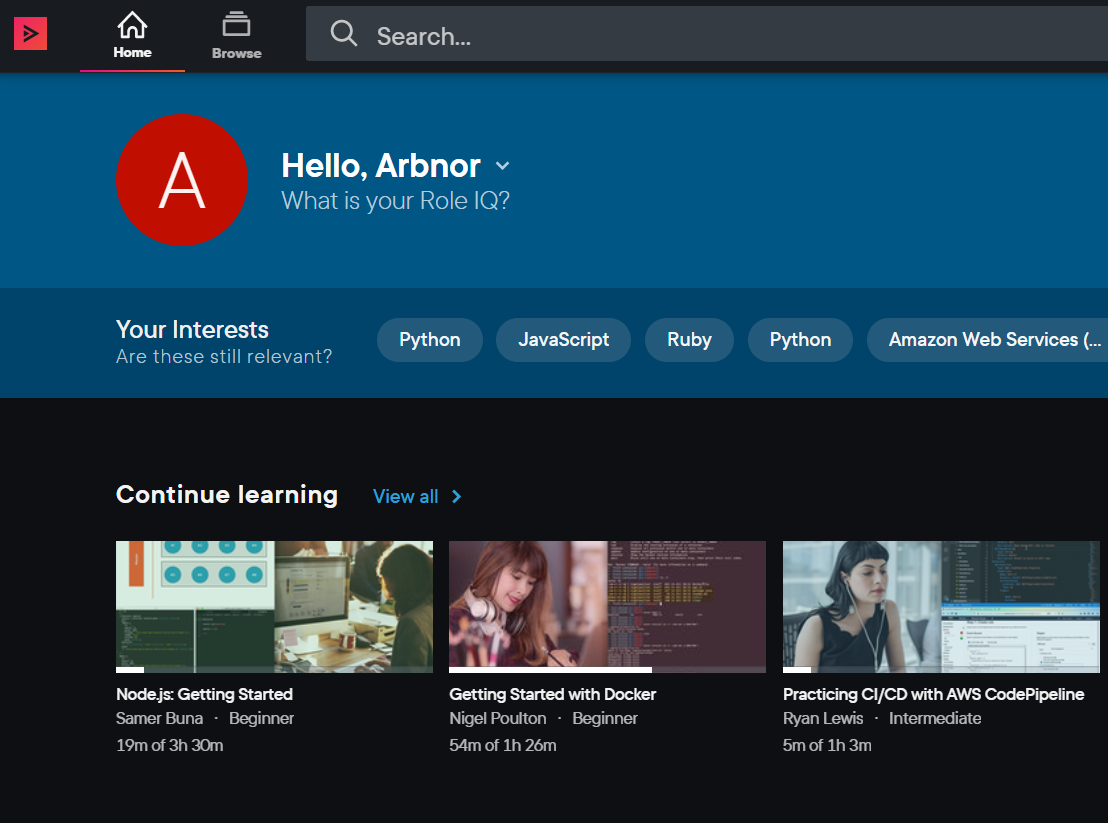
B3

# Duty 11 – Keeping up with cutting edge by committing to continual training and development – utilize web resources for self-learning; horizon scanning; active membership of professional bodies

Evidence



24% of the AWS SAA studied so far – currently doing around 1hour a day of learning regarding AWS (and creating my own documentation on it)



I also have PluralSight – currently not using it as I am learning AWS and NodeJs more – but I plan on using it soon for the Dev Tools: Currently studying NodeJS using udemy

Ksb to cover: K4 K9 K10 K22 K23 K24 K25

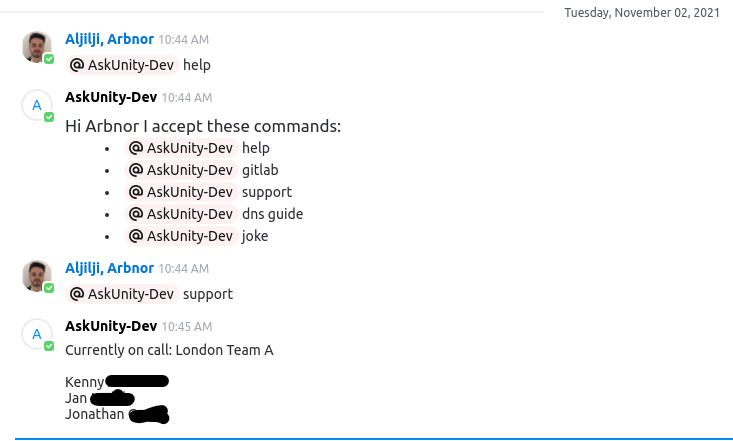
S4 S16

B1 B2

# Duty 12 – Look to automate any manual tasks that are repeated, often using APIs

Note to self: (We are currently working to remove a simple manual task of looking at the rota shift of who is on current support.)

I have managed to solve a manual task, removing the toil of logging into an internal website, and look at the rota which can be quite an annoying, repeated process. I have screenshotted below the bot being able to do an API call to xmatters and doing a GET request of the current team on shift. I came across this task as it was one of my user stories in my project to which I used Jira to create the ticket. This user story was created by listening and understanding the client needs.



Ksb to cover: K1 K4 K6 K7 K8 K11 K13 K14 K15 K17

S4 S5 S7 S12 S15 S17 S18

B3

# Duty 13 – Accept ownership of changes; embody the DevOps culture of ‘you build it, you run it, with a relentless focus on the user experience

When working within my project, I accept ownership of change as I am the developer of the app. After hearing the client user needs change, and adapting to it, I can modify my project around their needs and implement a new structure to it (with minor changes on it being made). For example, a client wanted the bot to be able to not be @ mentioned when in a private room, and only in a group room. This made sense as always @ the bot can be annoying, rather than you can just type the message and it will listen. Since that client need was suggested and required, I have been researching how to implement that new code change and taking full ownership of their need.

Ksb to cover: K4 K6 K9 K10 K22 K23 K24

S1 S2 S4 S6 S7 S8 S11 S17 S18

B1 B3 B4