# IaC – Assignment 2 – L00177599

**Part 1 – DevOps Strategy**

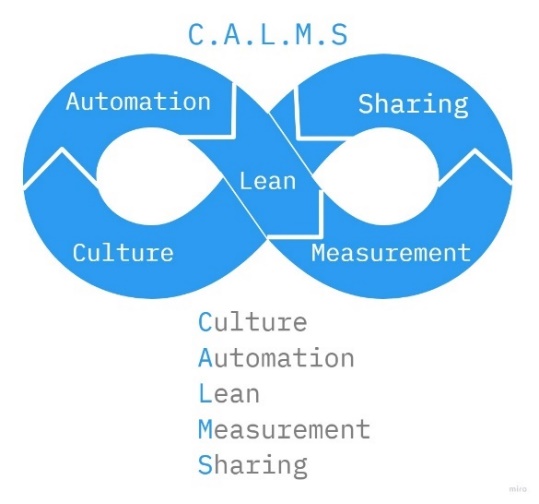
­­­

Figure C.A.L.M.S - DevOps Framework

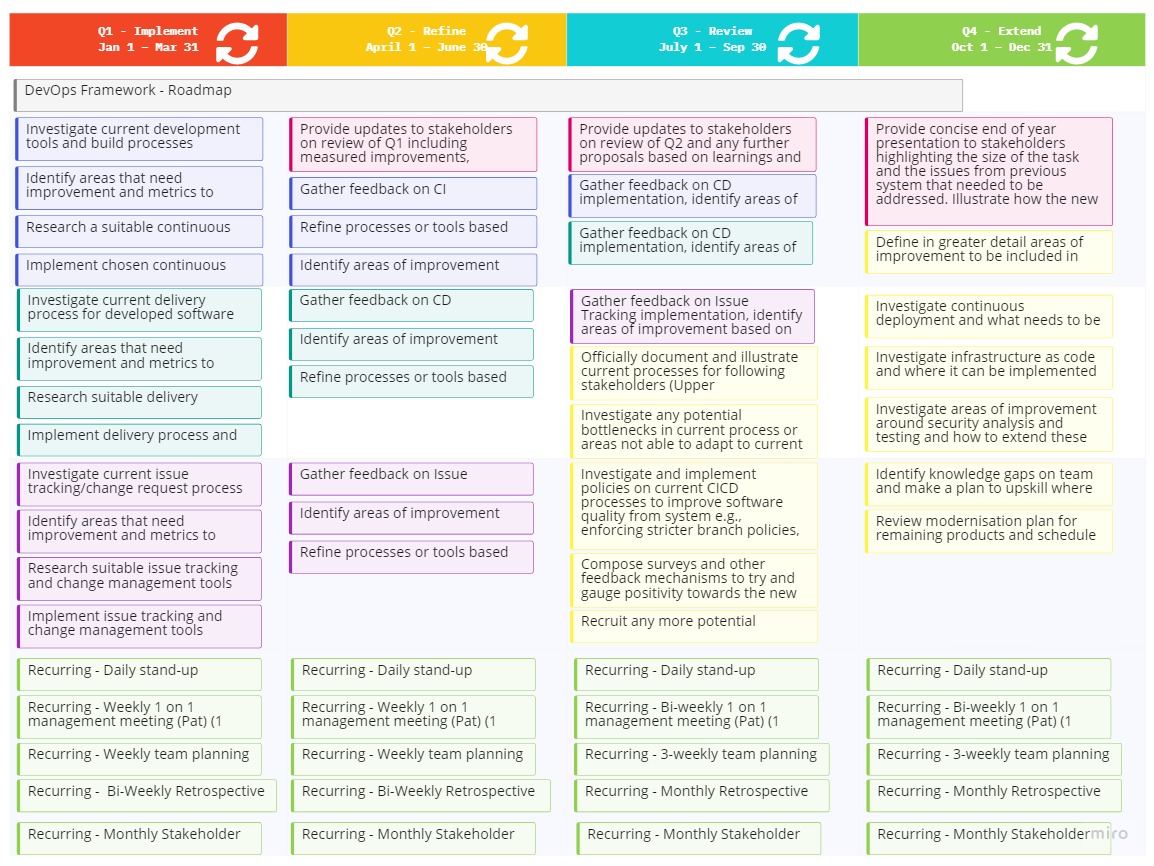


Figure DevOps Roadmap

Shinty Software is a company who has been a long-time provider of software with 20 years in the market. The economic downturn has put the company under severe pressure with both layoffs and more recently losing some of their best talent to other organisations.

The task at hand is to aid the manager, Pat, who has been recently appointed to a position where they intend to turn around the company by streamlining the products and services the company provides. There is already executive buy in as Pat has identified that it is essential to modernise the company’s internal processes first.

The company has hired myself to lead the migration to a DevOps framework from their existing waterfall development methodologies and processes. The company is familiar with Agile and has been working with a Lean coach recently to improve other areas of the company outside of the development team but has started trying to implement Agile in their SDLC also.

DevOps is a very new concept and the remaining employees have been siloed into their own area for many years as teams are split between Development, QA and Release. Communication across the teams is poor and most is facilitated through the R&D/Development manager who schedules a lot of the work across the different areas. There is also a QA manager and a business director in the R&D division who would attend meetings with upper management of the company regularly and are aware that these changes are critical to the company’s future and must be implemented quickly and expect to see some results in the first 3 months.

Due to the shift in the company with the recent addition of Pat there is some momentum to get this done but because it is all new, management expect to be presented with a roadmap of the proposed solution over the next four quarters and also a high-level overview of what is DevOps and some of the principles included.

The chosen framework to present is the C.A.L.M.S framework coined by Jez Humble and described in Atlassian’s article (Buchannan, 2022). It is a very high-level representation of the DevOps transformation in an organisation and it covers some familiar topics from the Lean principles that show how this can work in conjunction with processes already being implemented in other areas of the company. It illustrates some of the key concepts for DevOps integration but at a high enough level that it can be easily digested by upper management. The addition of the roadmap shows that there is a practical plan in place to implement these ideas.

The culture in this company will be a difficult transition for many as the remaining employees have been here a very long time and are used to the current processes and tools. Most who were open to change left when they had the opportunity but the remaining ones are hopeful and because this is coming from the top down the chances of success will be greater.

Automation is lacking currently as there is a lot of manual work being done to keep releasing software to existing systems and the team of myself, Jalen and Ren will aim to investigate and implement changes in Q1 as a matter of urgency to bring continuous integration and delivery to the development and testing teams so they can begin modernising existing products using new process. There will be a big focus on shifting left in regards to testing and security.

The concept of Lean is approached in this context by regular retrospectives on the development and processes but also by reducing the lead time on changes and applying more Agile principles to each person’s area of work. Management have invested in these principles in other areas already and are happy to see it being further implemented here as well.

Measurement is important to show that the current state of the system can be evaluated and also that any new processes and tools being implemented during the transformation are having the desired effect and that there are no deviations from the path as time goes on.

Sharing is a critical component as it is needed to break down the siloes previously put in place by the development practices in the company. This is true in the context of sharing knowledge and feedback on tools, processes and systems but also of responsibility for all internal and external stakeholders, with an added emphasis on development and operations teams, for the health of the software and the continuous improvement of it through time through amplified feedback loops.

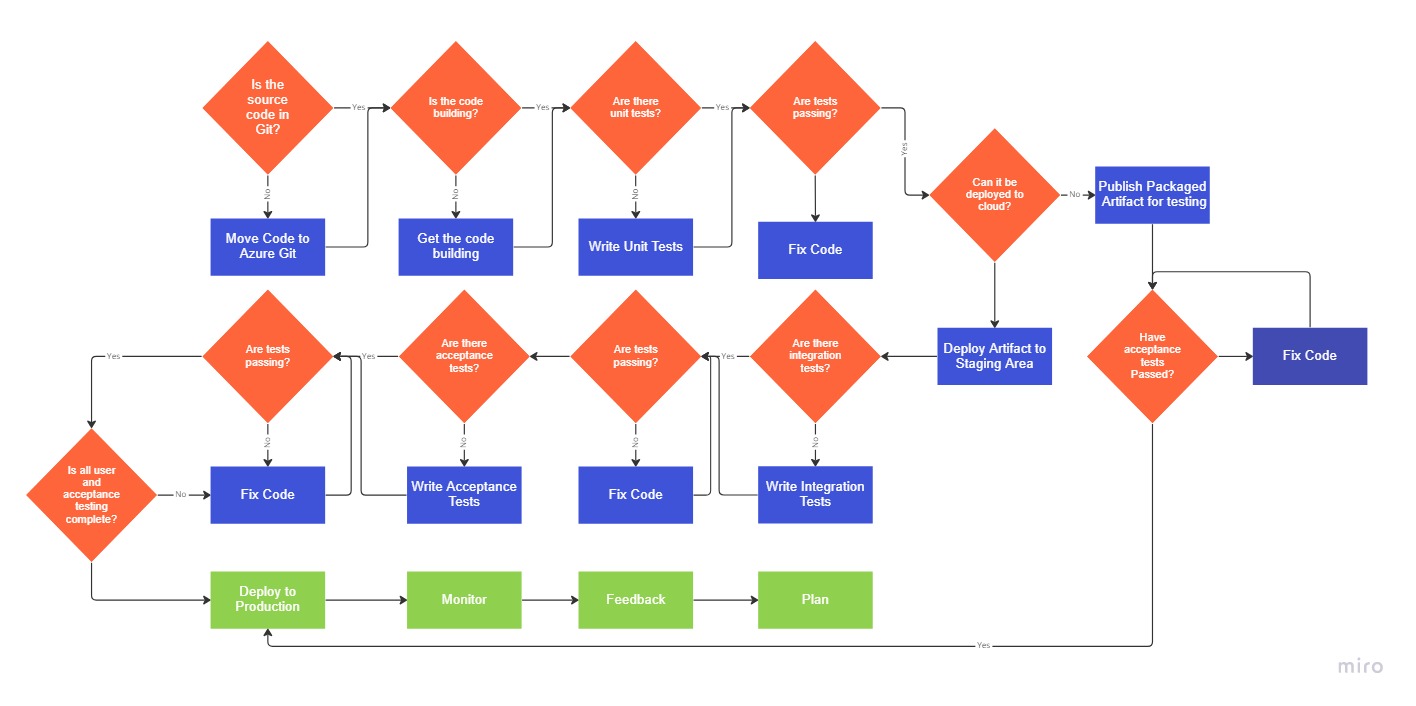
**­­­Part 2 – Code Modernisation Strategy**

Figure Simple Code Modernisation - Legacy and Cloud

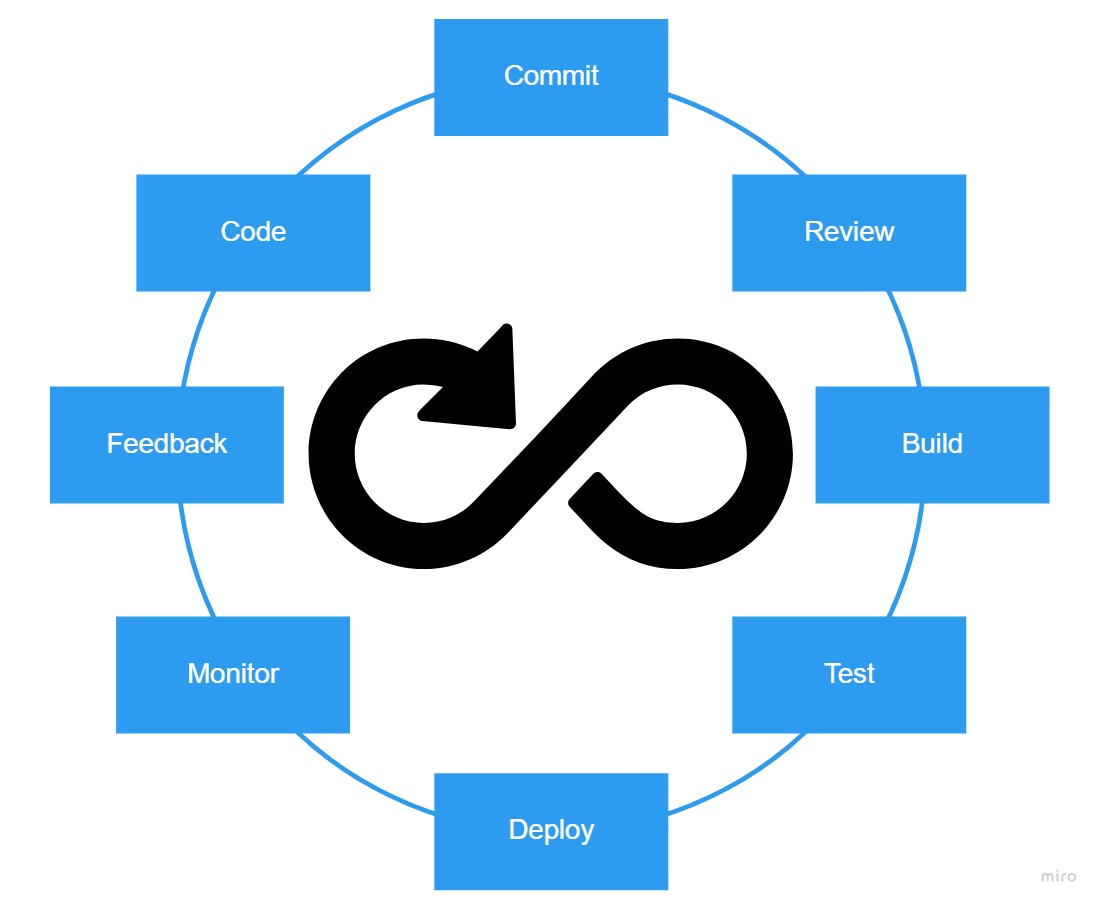


Figure Development flow for DevOps - CICD

­­­­­Shinty Software has 20 years of experience in the software industry. In those 20 years they have developed and delivered many solutions, most of which are still running in some capacity and fall under SLA contracts which must be maintained and updated with new features at customer request.

Older processes and tooling are showing their age and technical debt is prevalent across all of the products. Attempts in the past to move to more modern frameworks and tools have met resistance from management as it was viewed as too high of a cost. The introduction of more technical debt due to this was justified as prudent to keep customers happy and cash coming in. This has come to a point where it is no longer sustainable, as can be seen in the current state of the company and the radical actions being taken to turn this around. Lead times need to be reduced right now so money comes in faster again.

The long-term goal to enable this software modernisation is a three-pronged approach where the major areas to be investigated and improved are Architecture, Infrastructure and Delivery as described in IBMs Application Modernisation, (IBM Technology, 2019). Shinty Software has traditionally been a Microsoft .Net house and wishes to remain with this technology as much as possible and some work has already commenced to identify products that might be able to migrate to cloud infrastructure. The company is not able to take this approach with all projects immediately due to commitments maintaining the current legacy products for existing customers and keeping cashflow positive while transformation is under way.

Decisions are being made on important legacy products with older frameworks on whether they will be completely re-architected into a new cloud ready solution but in the interim all existing code and projects are to be moved to the new CICD process so the team can start upskilling themselves and also to start removing some of the bottlenecks from the old processes that have been hindering lead times.

The proposed DevOps framework aims to provide a minimum of CICD across all projects, and Agile processes being introduced into the development planning are helping break away from the old waterfall methodology by breaking up feature requirements into smaller incremental deliverables that fit in with the idea of continuous integration. Continuous delivery is available for any new cloud ready applications but also for the legacy desktop apps that are being delivered as seen in flow.

The previous source control management is very dated and does not integrate well with other software so there is a move to Azure Git which is a SCM based on Git and provided in the Azure DevOps solution. The company has decided to move to Azure DevOps as part of the transformation and code modernisation process to enable the team to follow the new migration flow for new and existing projects. There is already some Azure DevOps licensing available to Shinty Software as part of their MSDN subscription and it ties in with their idea of modernising the tools and having a single point for a developer to manage work items, pull requests, continuous integration and continuous delivery of the projects they are working on. Visual Studio is the most commonly used IDE in the company for development, which is also provided by Microsoft and integrates well with this system. Extensions are available for the automated testing tools used in QA and there is capacity to do more bespoke tasks for legacy software if required after modernisation decisions are made. Developers are reminded testing and security should be built into their code and automated testing and security stages have been included in the pipeline to reduce QA/Security time at the end of a release cycle.

Any new software is now being developed on this system and the older in-house build tools will be slowly deprecated as more products are migrated from the old system. This is a big change for the existing teams so Ren, Jalen and myself will be on hand to assist all teams using the new system and will act as advocates for the processes throughout the modernisation journey by practical applications of the principles of DevOps in each project as well as hosting workshops for the team.

Documentation needs to be supplied, mapping the new and old tools and processes so that the language can be adjusted accordingly to match the new system being put in place and get everyone in the right mindset to achieve this. Code reviews will be mandatory throughout all projects in line with compliance requirements but also to ensure knowledge is shared between teams and collaboration is high.   
  
There is some grumbling from older members of the team who are more set in their ways but constructive reviews of the processes and demonstrating the benefits to individuals as well as the organisation is helping to get everyone onboard with the changes.

**Part 3: Using the Code Modernisation Strategy**

Using the Code Modernisation procedure provided along with the DevOps framework a developer should be able to follow the flow diagram to modernise a piece of code or application in the CICD pipeline.

It is described as it is in the diagram based on the assumption that the decision has already been made to modernise the existing piece of code. Some projects can become obsolete over time and where possible should be abandoned so as not to waste effort on low value results.

The diagram does not describe the system in terms of planning, pull requests, code reviews etc. as it assumed the developer is following these patterns based on the policies on the system and team norms around code review and collaboration.

Following this process, the developer would take a piece of code in the form of a script or an application and begin at the first decision and work their way through until reaching the end.

Reviewing the script provided using this system, the first step was to include the script in source control so changes can be tracked and reviewed by the team (Azure DevOps in company case but Github for sake of simplicity of assignment submission), then make sure it compiles, or runs in the case of a script, without error. This required a lot of refactoring initially.

To add unit tests to this particular script requires the use of a framework like Pester designed for unit testing in Powershell as described in Unit testing in PowerShell, introduction to Pester (Miossec, 2019).

Once the tests are added and passing the decision is whether it is deployed to cloud or not, it could potentially be deployed to cloud if it was being used in a VM cluster or some kind of hosted service but in this case, it is a tool for internal network testing and is output as an artifact to be used locally by engineers so it is published to a shared location accessible by the team.

Acceptance tests are run against the output before being signed off as successful based on the outcome of these tests. It is then marked for deployment to production which in this case would be to make generally available to the wider organisation of engineers.

This is a more practical workflow where you can input an older, lower quality piece of code and it will force you to add quality and in turn bring it to a higher standard and modernise it before finally delivering it. Another decision flow solution to this modernisation question could be similar to the one previously referenced and described in AWS’s .Net modernisation article (Rao, 2022) where it describes the process with the output being more focused on the destination of the code based on the decisions made during the modernisation process. This includes the concepts of rehost, replatform and refactor for code modernisation decisions.

# Conclusion:

This assignment was interesting as it provided a real-world scenario where a DevOps solution needed to be implemented. This required a lot of research into existing DevOps strategies and frameworks as well as application modernisation strategies and how it all tied together to deliver a working implementation. Initially the brief appeared to be obscuring the information we needed to provide a solution but after repeatedly reading and referring to it during the research phase it became clearer that this was the exact problem and whatever solution being provided needed to address all of these areas the company is suffering in.

There is a successful history of delivering software in the company as it has been in business for 20 years. This is not to be taken for granted as it shows that what was there worked but this leads to a problem where you are dealing with people who can’t see a problem with the tools and processes in place already, as they have been “working fine”. Luckily it is clear something needs to change and Pat has been put into a position to drive this change. This alone is not enough but it shows there is buy-in from the top down which is extremely important in enabling change, especially for SMEs that have been operating for a while. Being able to provide management with roadmaps and clear processes is a good start but if you can also provide measurable results that appeal to them then there is an even greater chance for them to push for additional resources in this area like more budget or time.

Another thing mentioned was time is a limited to 50% DevOps the remaining on code modernisation with your staff, Ren and Jalen, one of whom is inexperienced and another who has been here since the company was started and is near retirement. This is a typical scenario in my experience, especially in companies where there is a mass exodus as many near the end of their career won’t move and new people will just want the experience. This is a good time to implement some paired programming as it gives the benefit of collaboration between two developers, new will learn from old in the same way old can learn from new as they are more recently educated in newer technologies. If you can provide clear direction for the team and articulate your vision to get them on board this can be a great resource in itself and can help plant the seed among teams when they see the benefit of the new processes and talk between teams.

In my experience there is no one size fits all so the solutions provided were extremely simplified in a way that they could be easily followed by the stakeholders and allow them to begin the journey, but also detailed enough and clear that there could be immediate results given the tight frames being imposed on the team.

The code modernisation flow was designed with the hosted version of Azure DevOps in mind as this will force the developer to update frameworks and tools due to Microsoft’s policies of removing older compilers and images from their solution when they become end of life. This can be a double-edged sword where people will claim they can’t use the tool because of x, y, z but because we know this is a directive from the company it means the software will have to be modernised at least to the point where it is follows one of the two flows in the CICD pipeline.

All of the proposed frameworks and processes are flexible enough to allow change and will be regularly reviewed. I think it is important to find the right fit for a company based on their needs and their industry so what might appear to be the perfect solution in theory may not turn out like that in practice but that is all part of DevOps and when you follow the common principles and practices it will allow you to adapt to those situations.

During the research for this assignment, I also read articles from RedHat (RedHat 2022) and Amazon (Rao, 2022) on .Net modernisation and how it can be implemented in an organisation. I also used learning resources available from Microsoft (Microsoft, 2022) to further my understanding of the concept of DevOps and to see how their solution could fit into this scenario.

[conorh-devops/atu-iac-ca2: CA2 lab for IaC ATU (github.com)](https://github.com/conorh-devops/atu-iac-ca2)

**References:**

Buchanan, I (2022) CALMS Framework - Assess your ability and measure progress on your DevOps journey. Available at: https://www.atlassian.com/devops/frameworks/calms-framework (Accessed 27 November 2022)

IBM Technology (2022) Application Modernization: Three Transformations at Once. Available at: https://www.youtube.com/watch?v=RJ3UQSxwGFY (Accessed 27 November 2022)

RedHat (2022)What is .NET application modernization?. Available at: https://www.redhat.com/en/topics/application-modernization/what-is-dotnet-modernization (Accessed 27 November 2022)

Rao, P (2022) Why you should modernize legacy .NET applications on AWS (and how we can help). Available at: https://aws.amazon.com/blogs/modernizing-with-aws/why-you-should-modernize-legacy-net-applications-on-aws-and-how-we-can-help/ (Accessed 27 November 2022)

Microsoft (2022) What is Azure DevOps?. Available at: https://learn.microsoft.com/en-us/azure/devops/user-guide/what-is-azure-devops?view=azure-devops (Accessed 27 November 2022)

Miossec, O (2019) Unit testing in PowerShell, introduction to Pester. Available at: https://dev.to/omiossec/unit-testing-in-powershell-introduction-to-pester-1de7 (Accessed 28 November 2022)

# Appendices:

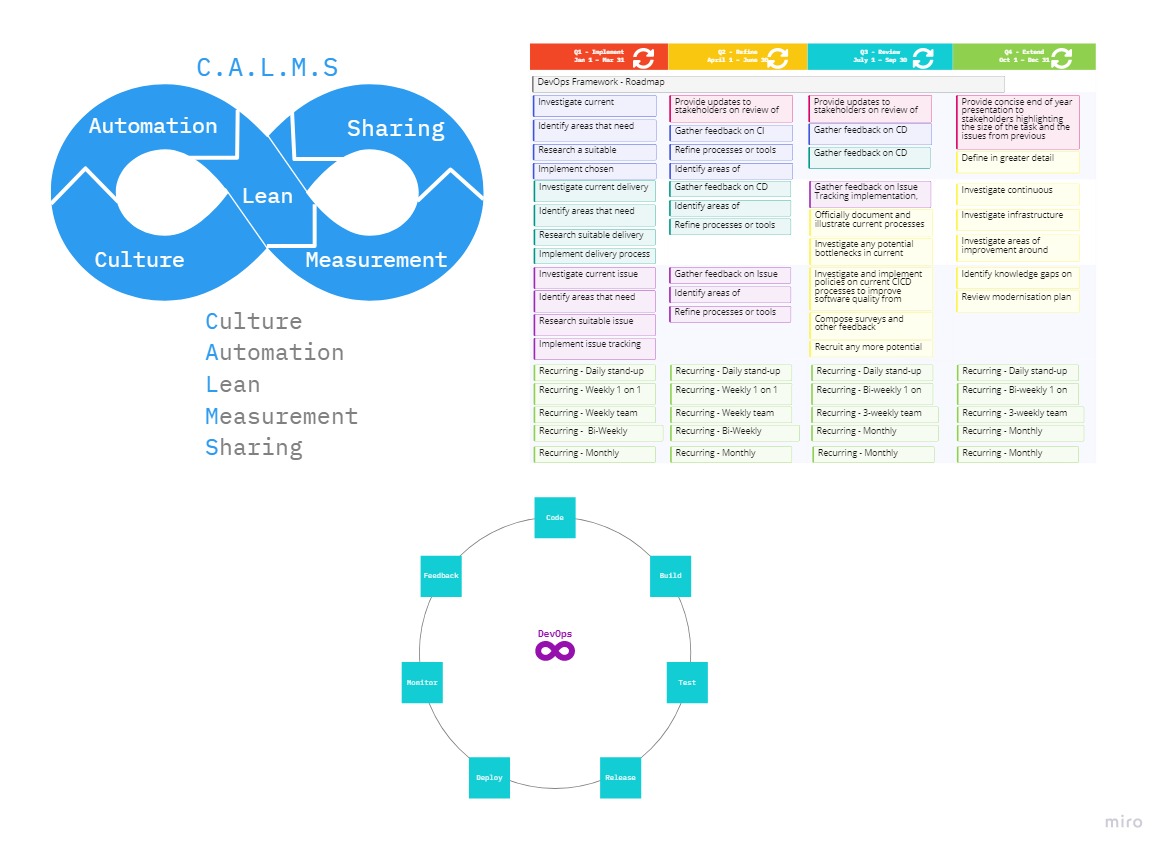


Figure Designing in Miro 1

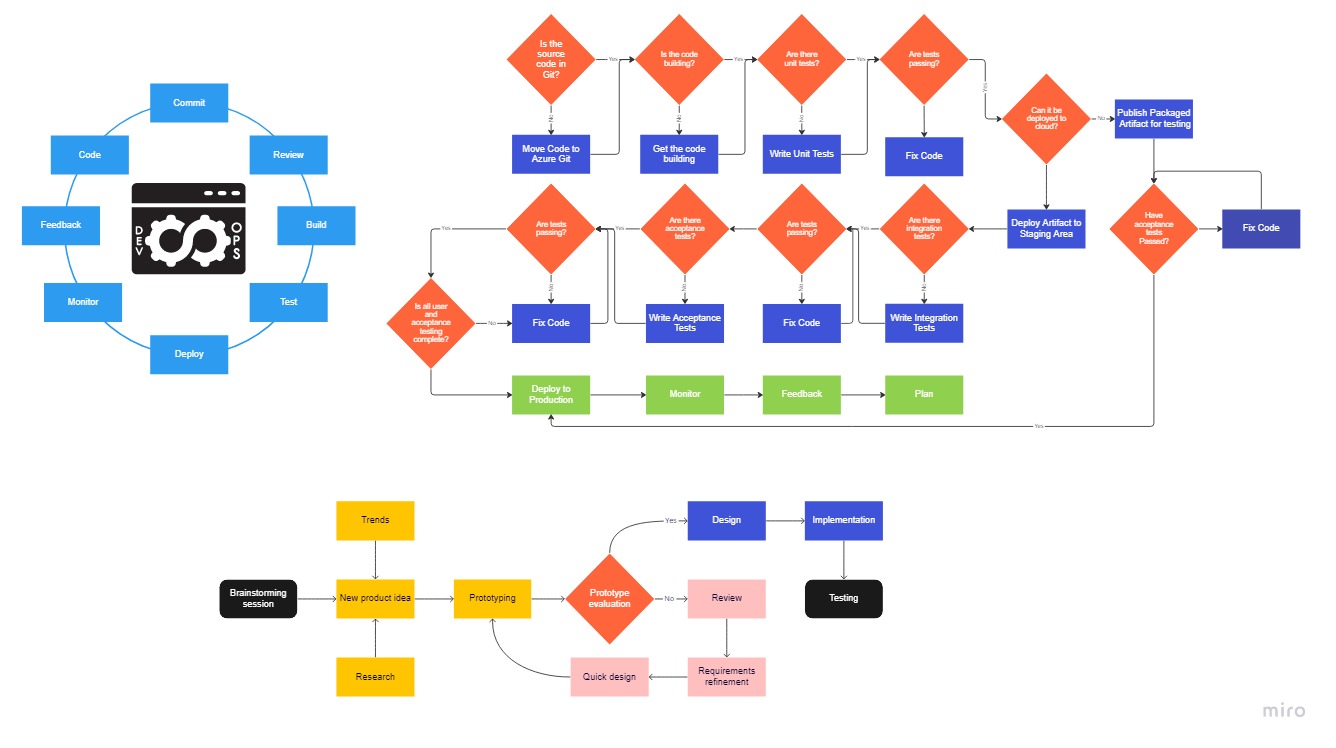


Figure Designing in Miro 2

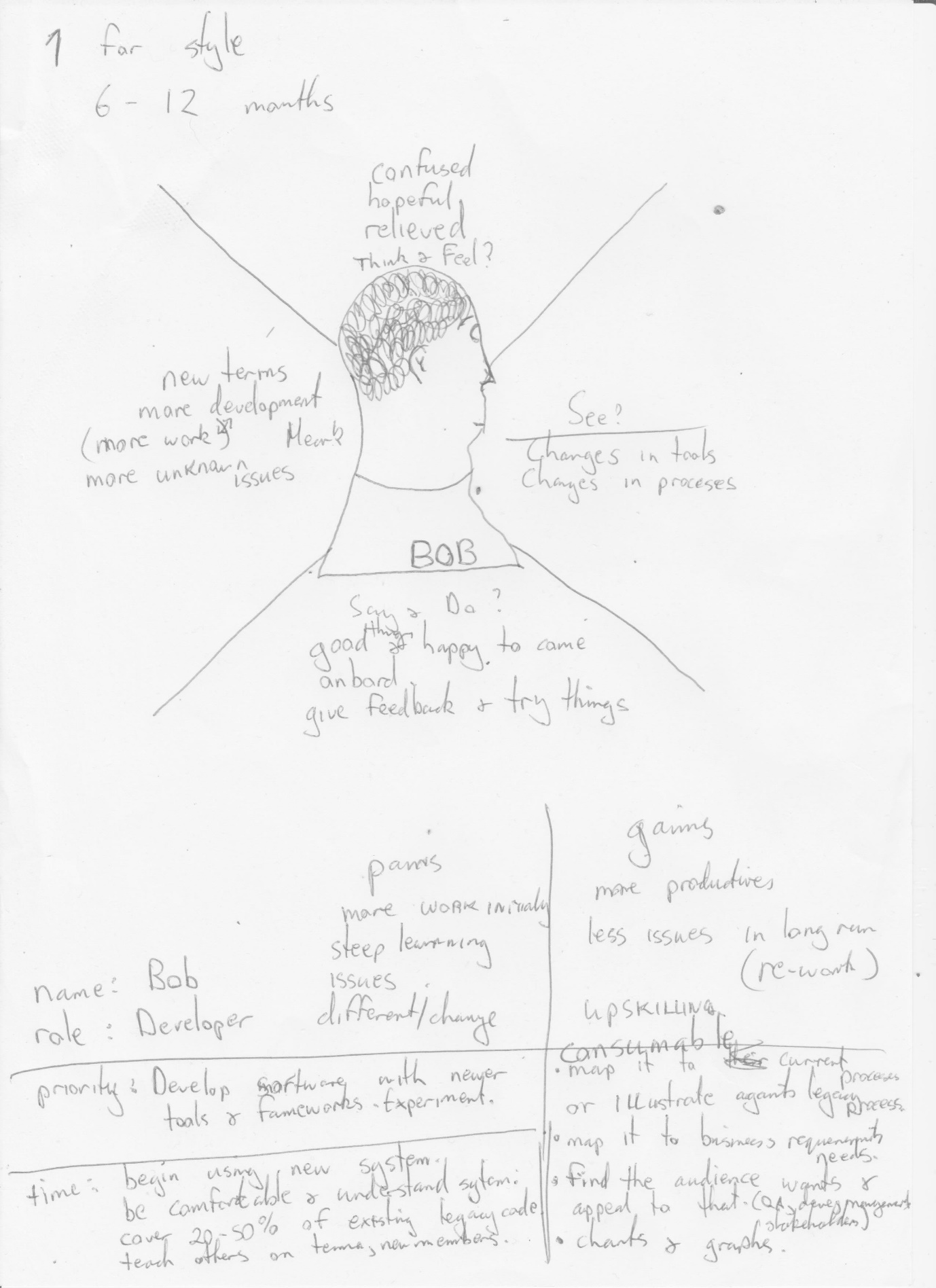


Figure Exercise in class to better relate to stakeholders