|  |  |  |
| --- | --- | --- |
|  | | |
|  | | |
|  | | |
|  | | |
|  | | |
|  | | |
| **SCHOOL OF INFOCOMM** | | |
|  | | |
| **<<INSERT DIPLOMA NAME HERE>>** | | |
|  | | |
| **INDUSTRY IMMERSION PROGRAMME (IIP) REPORT** | | |
|  | | |
| RP Logo - Full Colour | | |
|  | | |
|  | | |
|  | | |
|  | | |
|  | | |
|  | | |
|  | | |
|  | | |
|  | | |
|  | | |
|  | | |
|  | | |
| Student Name | Darrel Dominic Lim De Xiang |  |
|  | |  |
| Student ID | 22030316 |  |
|  | |  |
| Training Period | 13th March 2024 - Current |  |
|  | | |
| Company Name | National Computer System (NCS) |  |
|  | | |
| Supervisor Name | Naveen |  |
|  | |  |
| Supervisor Email Address | <<INSERT SUPERVISOR EMAIL HERE>> |  |
|  | | |
|  | |  |

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
|  | **PAGE** |
| ACKNOWLEDGMENTS |  |
| EXECUTIVE SUMMARY |  |
| 1. INTRODUCTION |  |
| 1. BACKGROUND INFORMATION OF ORGANISATION |  |
| 1. TRAINING AND WORK ASSIGNMENTS |  |
| 1. REFLECTIONS |  |
| 1. CONCLUSIONS |  |
| REFERENCES |  |
| APPENDIXES |  |
| 1. Internship Report Clearance Form |  |

**ACKNOWLEDGEMENTS**

Firstly, I would like to express my gratitude to my colleagues from National Computer System (NCS), Accenture (ACN), and Inland Revenue Authority of Singapore (IRAS). They all played a significant role in my learning and created a supportive environment during my time at IRAS. I would like to specifically thank Fernando, Rajesh, Surya, Naveen, Ka Hao, and Kiran from NCS, Rakesh and Pirathepan from ACN, and Mervyn, Jun Feng, and Jonathan from IRAS. These individuals have gone out of their way to assist me when I faced challenges and have been incredibly supportive throughout my internship.

I would also like to extend my appreciation to my friends for sharing their experiences and maintaining a positive spirit.

Finally, I am grateful to Republic Polytechnic for providing this wonderful opportunity to intern at NCS, which has been an enriching experience for me.**EXECUTIVE SUMMARY**

During my time contracted to IRAS as a DevSecOps (DSO) Engineer, I had the opportunity to explore various Azure services such as Azure DevOps (AZDO), Azure Kubernetes Service (AKS), Azure Data Factory (ADF), and Managed Service Identity (MSI). Additionally, I worked with several third-party tools integrated into the company’s pipelines, including Mend, Helm Flux, SonarQube, and Fortify.

I was contracted to work on the IRAS Inland Revenue Interactive Network (IRIN3) Project and have done many different tasks on the release and support side. Here is the list of tasks I have performed:

IRIN3:

- Set up a separate ADF CI/CD pipeline for Dynamic 365 (D365).

- Promoted a D365 pipeline from Government on Commercial Cloud 1 (Gcc1) to the Gcc2 environment.

- Fine-tuned WhiteSource Mend tasks to limit memory usage and set three auto retries.

- Utilized Kusto Query to analyze build agent details from the (Azure Kubernetes Service) AKS logs function, located under the monitoring section.

- Used Azure Command-Line Interface (ACI) and Kubectl to monitor pods in real time and troubleshoot errors.

- Conducted research on third-party tools such as LightHouse to determine their applicability to the project.

- Performed minor tasks involving bug-fixing pipelines, assisting developers with queuing pipelines and identifying pipeline failures.

**I INTRODUCTION**

The purpose of this report is to consolidate and reflect on my experiences during my internship at NCS, where I was contracted to IRAS. This report will detail the knowledge and skills I acquired, the challenges I encountered, and their impact on my future.

NCS provides a wide range of technological services to its partners, including cloud solutions, cybersecurity, engineering, and infrastructure. IRAS oversees tax collection and regulation, ensuring compliance and efficient tax management in Singapore.

As a DSO Engineer, my team was responsible for ensuring that the pipelines and build agents were operational and error-free for developers. We maintained and upgraded these systems, and helped to ensure that the developer's code was free of vulnerabilities before deployment to various environments. We also work with other departments in IRAS to ensure that the tools, extensions, and software that we use are free of vulnerabilities and are patched if there are any.

As mentioned in my summary, my role was more focused on maintaining and upgrading the pipelines and helping to set up the extensions that are used in the pipeline.**II BACKGROUND INFORMATION OF ORGANISATION**

Formally known as National Computer System, NCS was founded in 1981 and has a strong presence in the Asia-Pacific region, delivering end-to-end ICT solutions to various sectors. Its mission is to advance communities and business by harnessing the power of technology to improve lives and drive business success, by bring people together to make the extraordinary happen. NCS has grown significantly, establishing itself as a trusted partner for digital transformation initiatives. Its market position over the years is reinforced by strategic partnerships, continuous innovation, and a commitment to customer satisfaction.

Currently, it has over 13,000 talents across Singapore, Australia, Hong Kong, China, and India, with more than 4,000 active projects, 57 areas of specialization, and operations in more than 20 countries within the Asia-Pacific region.

NCS company organisational structure is broken down into 8 different sections:

1) Product & Engineering

2) Finance

3) HR

4) Legal

5) Marketing

6) Health

7) Operations

8) Sales

The Inland Revenue Authority of Singapore (IRAS) was established in 1960 as a statutory board under the Ministry of Finance (MOF) and plays a critical role in ensuring the efficient and effective collection of taxes to fund the government in developing Singapore. Their mission is to act as an agent of the Government and provide services in the administration of taxes and enterprise disbursements, as well as to represent Singapore internationally on matters related to taxation.

IRAS has evolved to become a modern tax administration, leveraging technology and data analytics to enhance its services and ensure a high level of compliance. Its proactive approach and focus on innovation have positioned IRAS as a model tax authority globally, with a vision to be the leading revenue authority in the world. Provided below is the organizational structure taken from their webpage.A diagram of a company's company's company

Description automatically generated **III THE TRAINING AND WORK ASSIGNMENTS**

On the 11th of March 2024, I began my internship at NCS as a DevOps Engineer and was contracted to IRAS to work on the IRIN3 Project as part of the DSO Team. In this project, we utilized various Azure services, with my primary focus on Azure DevOps and Azure Kubernetes Services. Azure DevOps is a source control platform similar to GitHub, which I had previously used in the C206 (Software Development Process) course. It features dashboards for adding work items and Product Backlog Items (PBI), repositories for storing code as a single source of truth, pipelines, and artifact feeds.  
(Screen shot of the Azure DevOps service)  
A screenshot of a computer

Description automatically generated

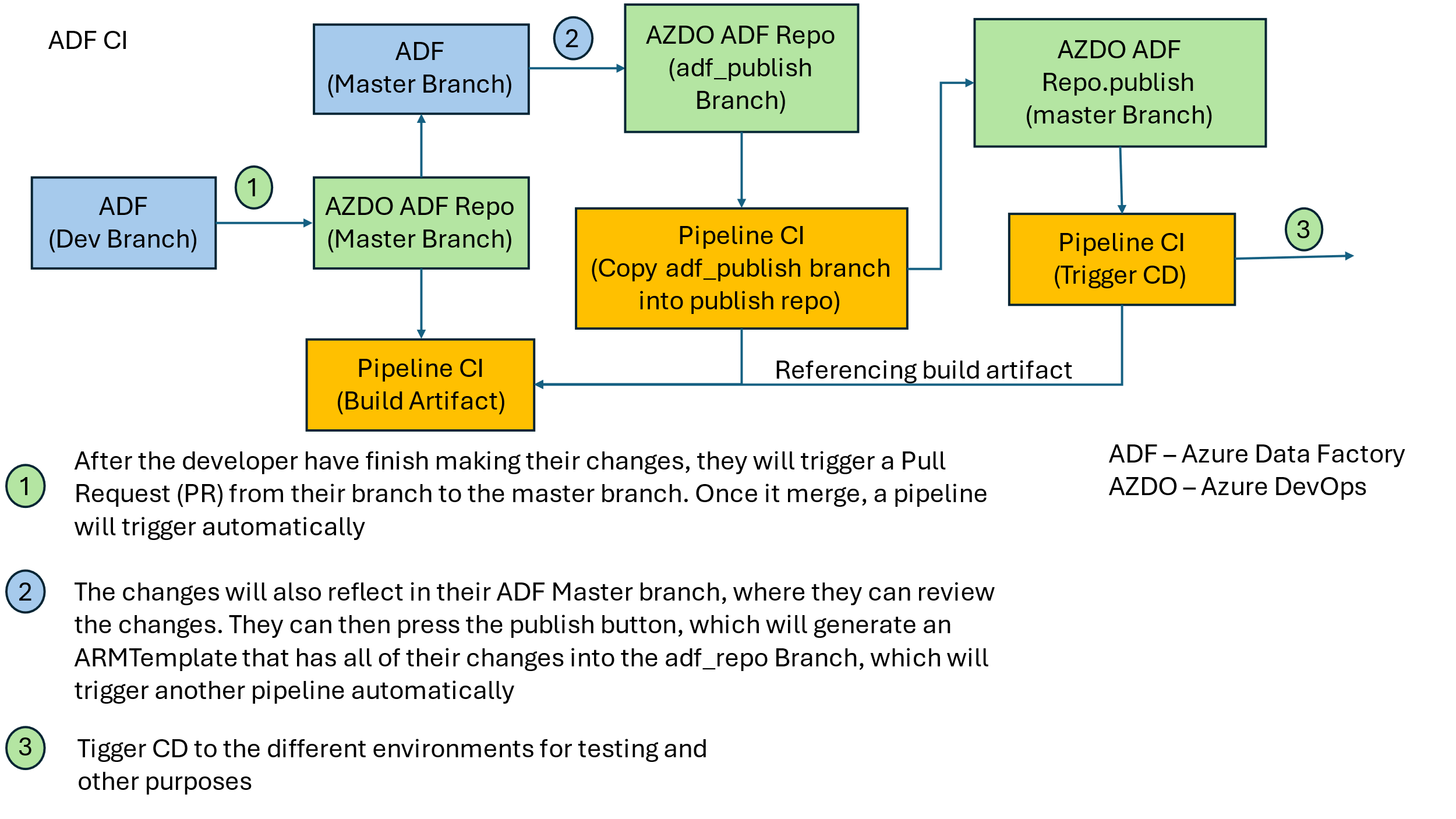
While I was familiar with the DevOps software lifecycle and CI/CD pipeline concepts from the C381 (IT Service Operations) course, integrating the pipeline and utilizing the artifact feed were new experiences that I learned during this internship. There was no formal training; instead, it was a hands-on learning approach where my colleague, Fernando, would show me the basics and explain the workflow before assigning me tasks. Whenever I had questions, I would seek his guidance.

Over the course of these five months, I mainly contributed to IRIN3, but also helped out with other projects:. I will elaborate on the tasks I performed for each project.

**IRIN3:**

**Azure Data Factory (ADF) D365**

(Picture of the pipeline workflow)



Originally, the ADF was shared between the Data Migration (DM) Team and Dynamic 365 (D365) Team. As stated previously, the code changes and pipelines they implement in their ADF are compressed into an Azure Resource Manager Template (ARMTemplate), which is in JSON format. This file contains all their code changes and pipelines in a textual format. During the Continuous Deployment (CD) stage, the ARMTemplate is deployed into various environments for developers to continue testing the application.

However, ADF has a deployment size limit of 4MB. If the ARMTemplate exceeds this size, the pipeline deployment will fail, which was the main issue.

To resolve this, a separate pipeline was created, and a linked ADF instance was established for the D365 team, separating the DM and D365 code changes. Creating this pipeline was my task, which involved coding in YAML. Initially, I struggled with navigating the Azure DevOps repositories and pipeline flow since the YAML files can reference other YAML files located in different repositories. With Fernando’s help, he showed me how to navigate Azure DevOps and explained the ADF pipeline flow, which follows the process outlined in the accompanying picture. Armed with this knowledge, I created my own branch and began coding.

The first major challenge I faced was figuring out how to reference the pipeline from the correct repository and branch, which is done via API. It took me some time to understand how the API functions, and after many attempts, I eventually managed to specify the correct repository using the repository ID and replace the "master" branch with my branch name. Following this, I updated the YAML template file, which is a common file referenced by ADF, with the new details of D365 and modified the template file as needed.

Once the CI part was done, the next step was the CD part. Similarly, I had to update the ADF CD template that stores the deployment tasks. To test whether the pipeline worked, I enlisted the help of Piratheepan, a Data Migration Developer, to test the pipeline from the start by making a sample code change in a new branch and then submitting a pull request (PR) to trigger the pipeline flow as outlined in the diagram above. When we reached the virtual deployment stage, the pipeline encountered an error related to the storage blob.

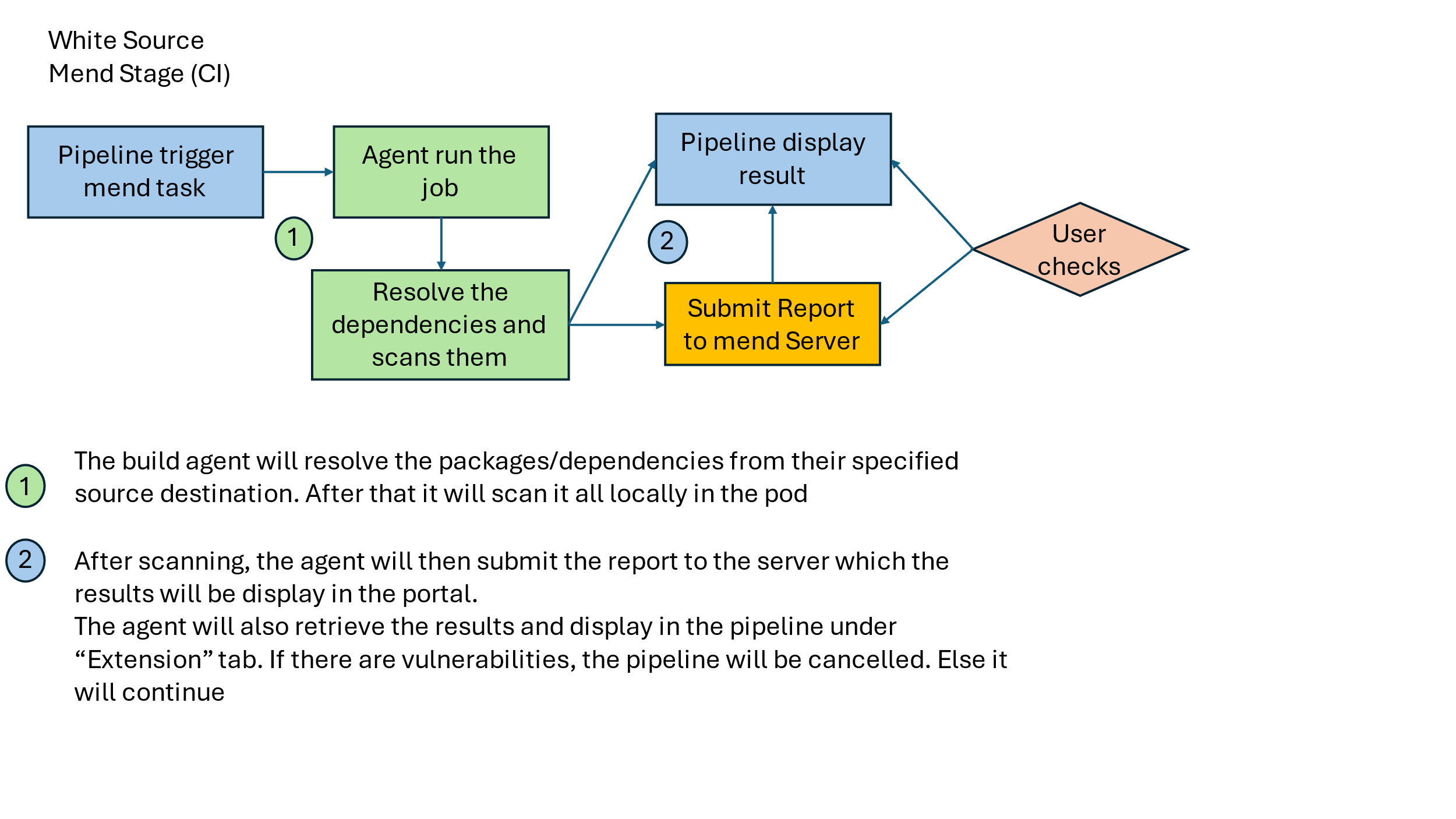
Upon further investigation with the help of Fernando and Rakesh (D365 Developer), we found that the ARMTemplate was successfully deployed and reflected in the environment, but the pipeline error indicated an issue with the ADF. This prompted Rakesh to raise a request to address the issue. While waiting for the request to be completed, I fixed any errors in the YAML files and raised a PR to merge the code with the master branch, resolving some code conflicts before it was successfully merged.

After the request was fulfilled, we tested the pipeline CI/CD on the actual environment again, but encountered a new error related to the Resource Group during the CD part, which I was not familiar with. Fortunately, Fernando knew how to troubleshoot this issue and met with Rakesh to check the resource group, as we did not have access to it. After checking the deployment logs, we discovered that some of the ARMTemplate values were not correctly configured from the linked ADF instance, causing the deployment failure.

After troubleshooting and manually changing the ARMTemplate values, I performed a test deployment to verify its success. Following this, I researched Linux sed and jq commands to automate the replacement of JSON values in the pipeline. This process took me a few days and required frequent communication with Rakesh to ensure the values were correct. After further fixes and PRs, I was able to automate the replacement and addition of JSON values into the ARMTemplate.

Unfortunately, the ARMTemplate size still exceeded 4MB. After checking the file size with Notepad++ and with Rajesh's help, we discovered that the whitespaces in the JSON file also contributed to the file size. Rajesh suggested minifying the ARMTemplate using jq, which I managed to do after testing and debugging. This drastically reduced the ARMTemplate file size from 4.2MB to 2.4MB, allowing successful deployment of the ADF D365 CD and preventing future increases in the ARMTemplate file size.

**WhiteSource Mend Issues**

One of the third-party extensions integrated into the pipeline is WhiteSource Mend, a security scanning application that automatically resolves detected vulnerabilities. The magic behind this application is that it resolves all the dependencies and packages declared by the source code and scans them locally in the agent. The downside is that it will scan all dependencies, even those that have been previously scanned. Afterwards, the agent will communicate with the Mend Server to upload the report, and an API can be used to retrieve the report from the server to reflect the code coverage in the pipeline.

Since late 2023, there have been error reports of Mend tasks failing, and the frequency of these errors has increased across all pipelines. The error indicated that the agent took too long to respond, and since I had some bandwidth while working on the D365 ADF issue, Fernando assigned me the task of troubleshooting it. He showed me how to view the build agent details in the AKS Logs using Kusto Query, which I quickly picked up on as it is similar to SQL query, which is covered in module C207 (Database Systems). Checking through the different datasets and filtering out the tables, I found that the agent status had a bash exit code of 137, which occurs when it is out of memory. Since the build agents are pods in a Virtual Machine (VM), I investigated further to see the overall resource utilization of the VM. I found that the VM memory usage was around 60 GB and in some cases above 64 GB. After spiking, the total number of pods would drop, confirming the theory that the pod crashes from insufficient memory as each VM is only allocated 64GB of memory.

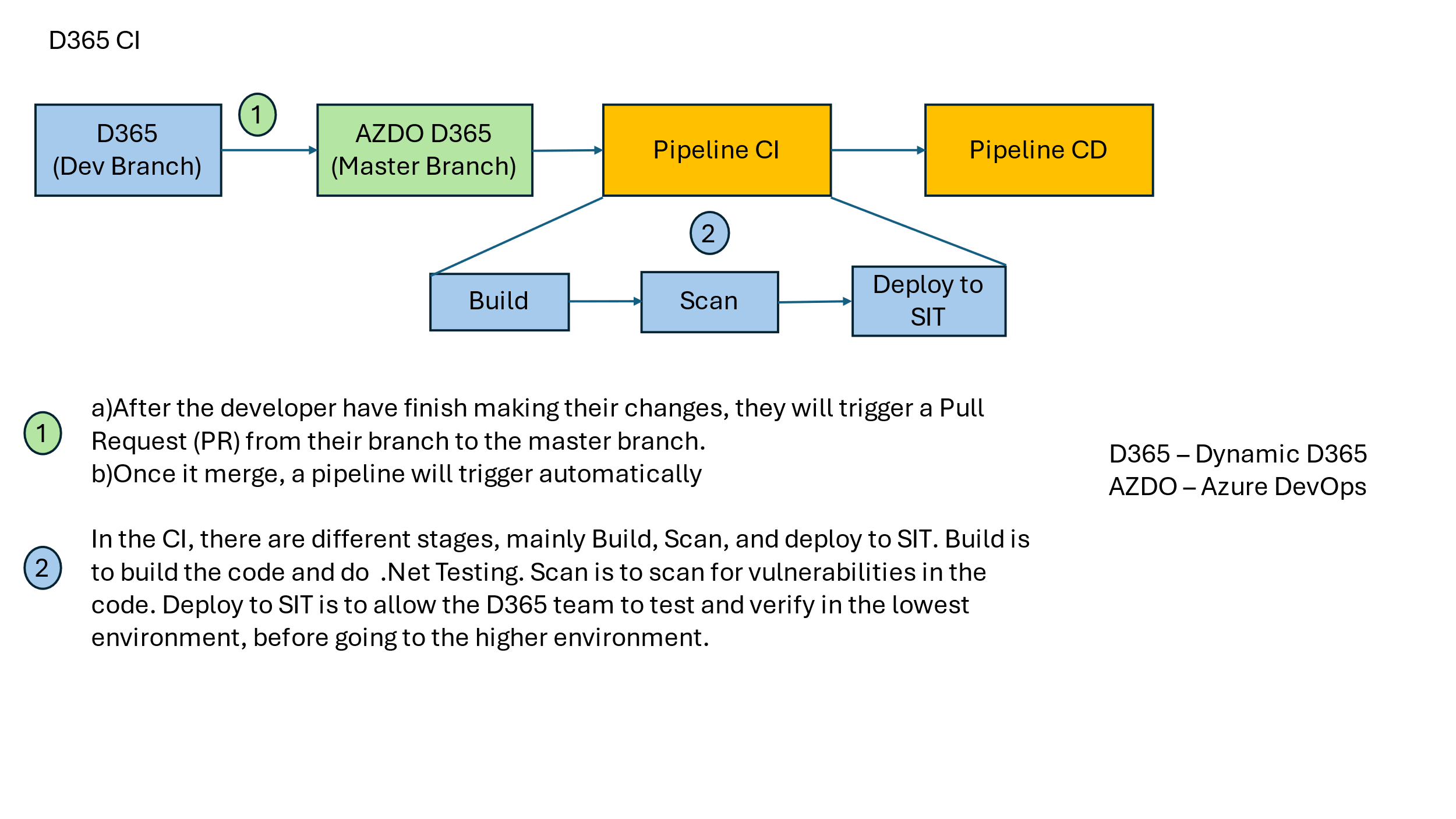
Analyzing the other pipelines where the Mend task failed from unresponsive agents, a pattern was discovered that when the VM is near or above 60GB, the pod running the Mend scan will crash. They also displayed the same final logs that state "1) Removing the agent from the server. 2) Failed to remove the agent. 3) Agent is running a job. 4) Retrying in 30 seconds."

Reporting my findings to Fernando, we dug deeper into the pod configuration in the AKS cluster. Using the Azure Command Interface (ACI) and issuing kubectl commands, we reviewed the pod YAML configuration. In it, there was no memory usage limitation in place, only a minimum memory usage of 12GB per agent. Additionally, we analyzed the script that starts the build agent and found the display message "Retrying in 30 seconds" under the termination condition. This condition ensures that the agent terminates gracefully and would not affect any processes running at that time. Listing the pods in the AKS, I further analyzed those pods with the status "unknown" and retrieved the necessary details to run Kusto Query to see their logs, and sure enough, they had the same error as discovered.

While troubleshooting, there were instances where the Mend Task also took longer than usual, ranging up to 33 minutes. This reached beyond the allocated configured time limit for that job. At that point, I had two issues with Mend: 1) Memory usage, and 2) Long duration. I prioritized the memory issue first as it would terminate the pipeline, while the duration issue could be temporarily mitigated by extending the time limit from 30 minutes to 1 hour. With Fernando's recommendation, I tuned the Mend task to limit the memory usage to 2-4 GB. In addition, I added a retry parameter to re-run the Mend task if it fails, as the connectivity can sometimes be unstable.

After implementing it, I checked the raw pipeline logs to confirm that the tuning configuration was in place. The logs also show double the amount of dependencies behind scanned in the summary logs, a staggering total of 38,400 dependencies, with 64 unique. After some testing, I found out that there is a task that zips and copies the files into another folder, which duplicates the overall code to scan. By removing this duplication, the total dependencies scanned dropped by half, 19,200 while the unique dependencies remained at 64. Throughout the whole troubleshooting and implementation phase, the mend scan task duration increased and once peaked at 1 hour 48 minutes. After pushing the code changes, the total average time taken decreased from 1 hour 22 minutes to 13 minutes, as the bulk of the time taken was on scanning and uploading the report to the Mend server.

**D365 CICD**



Aside from D365 ADF, another pipeline is used to deploy the D365 Customer Relationship Management (CRM) application to different environments. This pipeline was last used in September 2023 and has since become outdated and unusable, as most services have migrated from Government on Commercial Cloud 1 (Gcc1) to Gcc2, while D365 remained on Gcc1.

When troubleshooting the pipeline, the Gcc1 build agent encountered a certificate error. After discussing with my supervisor, Naveen, we agreed to migrate the D365 pipeline to Gcc2. This process required raising requests to ensure the Windows build agent used for compiling the code had the necessary system software capabilities installed.

Throughout this process, I worked closely with Surya, who supported and assisted me in raising requests and troubleshooting the pipeline as needed. Once the build agent had been provisioned with the required capabilities, we tested the building and scanning parts of the CI, which encountered errors due to vulnerabilities in the application code. I disable these to continue testing towards the CD part, where the pipeline pushes the code to the System Integration Testing (SIT) environment.

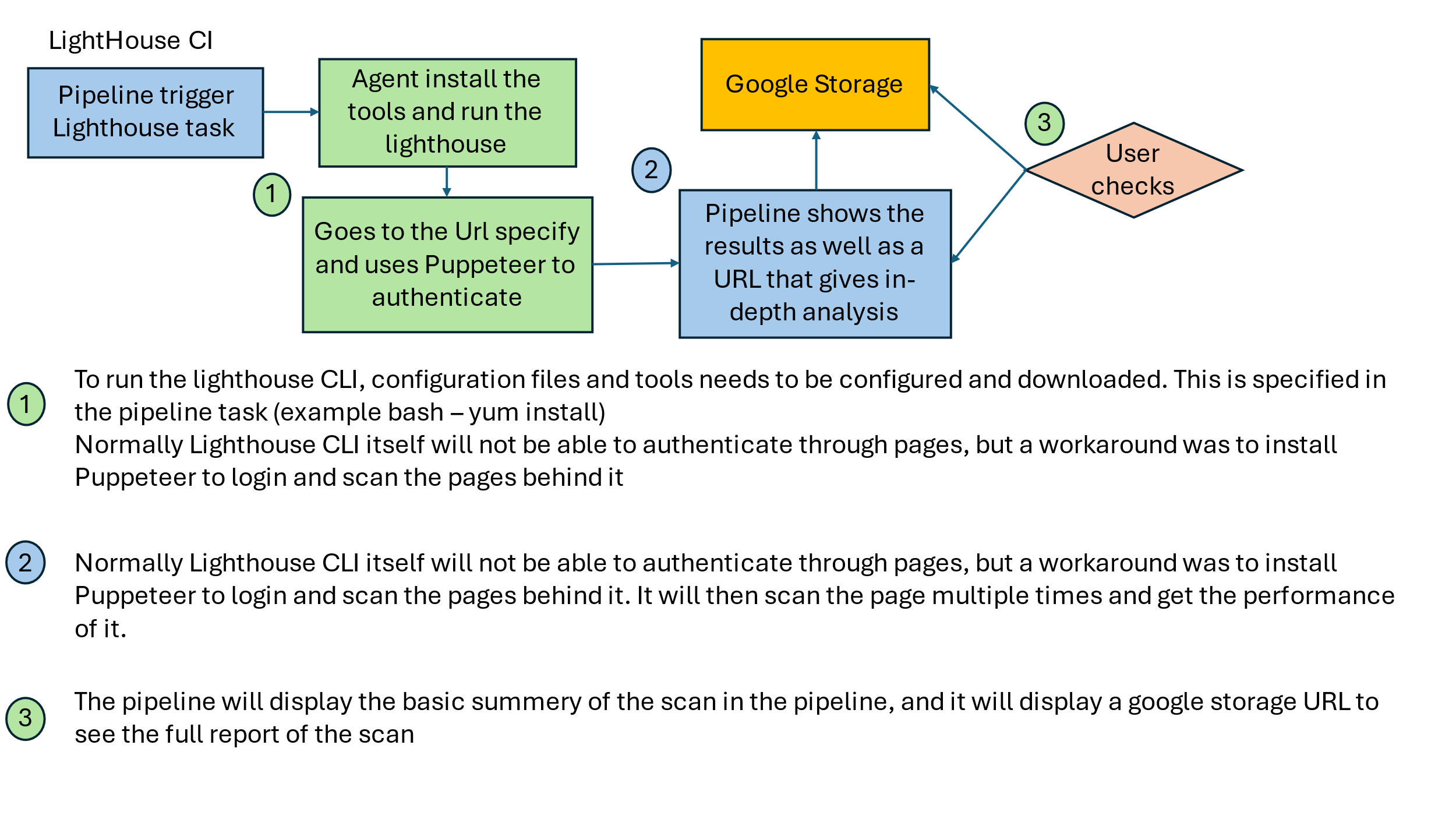
During this stage, I faced difficulties retrieving secrets from the Azure Key Vault. A discussion with the D365 Apps Lead, Edy, revealed that the Azure Key Vault (AKV) secrets had expired and that the method to retrieve the AKV secrets had changed in the pipeline. After restoring the secrets and updating the method, deployment to the SIT environment was successful.

**Helm-Flux**

In AKS, we also use Helm-Flux to manage the various pods and namespaces. The configuration in the Helm YAML file is applied to the AKS every 10 minutes with a 5-minute timeout. This setup was initially presented by Fernando and recently implemented. He tasked me and Maha, another DevOps intern from NCS who is also in the same diploma program as me, with verifying the changes and modifying the Helm YAML configuration to enhance it.

Our findings revealed that the implementation of the Helm-Flux sync and timeout were not configured correctly, and some of the Helm YAML files were non-compliant, which caused the Helm-Flux configuration to be denied deployment. We also implemented three parameters to the Helm file to ensure that the Helm-Flux passes the test cases before deployment, preventing it from skipping the tests. This troubleshooting experience helped me understand how Helm-Flux operates with our AKS, which proved valuable for future troubleshooting when Helm-Flux deployment issues arose.

**Lighthouse**

  
During some of the tasks mentioned above, I had the bandwidth to pick up other assignments. Fernando asked me to research Lighthouse, a tool that provides insights into webpage performance and overall user experience. The task was to determine if the Lighthouse CLI could be implemented in the pipeline to scan webpages and authenticate itself to scan webpages that require login credentials. This research allowed me to explore Lighthouse tools and introduced me to JavaScript, which is used for Lighthouse configuration.

I also used Puppeteer to script the login process. However, the downside is that it requires the HTML fields and input parameters to log in. Without these, it cannot determine where to input the username and password. Additionally, Puppeteer cannot input credentials into floating pages or pop-ups, as the inspect function does not detect them. This task broadened my knowledge of the different tools that can be incorporated into the pipeline and improved my skills in YAML coding, as this was one of my first few tasks, and I was not very familiar with the syntax back then.

Screenshot of the results in the pipeline:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Others**:  
Sometimes, I get direct messages from the Developer for small tasks in helping them troubleshoot their pipeline, or just help to run/rerun a pipeline. With the help of Jonathan, I have learned how to retrigger a pipeline CD, how to select the proper artifact version to use in the pipeline as well as the appropriate reason that are allowed to trigger the pipeline.

Outside of the CICD point of view, I help developers retrieve their SonarQube report when they cannot access it and help my team troubleshoot the Kubernetes pods when an unexpected incident occurs, 1 of the major ones being when the SonarQube task gets error throughout all the pipelines. I interacted with the AKS to check the pod status and found that the Azure database data IO (input-output) was at 100% on average. From there I worked with my colleague to forward the issue to the database team to resolve it.**IV REFLECTIONS**

Interning as a DSO Engineer was an eye-opener, providing first-hand experience with industry project structures and the DevOps project management methodology lifecycle. It familiarized me with cloud services, the flow of CI/CD pipelines, and the use of source control platforms in a company. My hands-on experience with Azure deepened my understanding of cloud concepts, which has been beneficial when studying for the AWS Cloud Practitioner certification. This practical exposure enhanced my skills and broadened my perspective on cloud technology, including planning and designing simple architectural flows.

I aim to develop better time management. While I am already part of the IRIN3, and helping with other IRAS projects, I will soon be onboarding another project at Marine Port Authority (MPA). Sometimes, I struggle to manage the workload when tasks come all at once. For instance, during one meeting, I received seven messages from different people asking for help. Some were urgent, some were not, and being unable to address them immediately made me struggle, even forgetting one until the developer reminded me. I also need to improve my presentation skills, as I often rely on Fernando to do most of the talking in meetings. Without slides or sufficient preparation time, I find it challenging to deliver information effectively.

Overall, the internship has been challenging, with days that left me mentally exhausted. However, documenting simplified versions of my tasks, blockers, difficulties, and resolutions has allowed me to reflect deeply and learn from each experience. When similar tasks arise or colleagues need help, I feel a sense of purpose in my work, knowing that what I have done during this project is meaningful and that I can contribute back.

**V CONCLUSIONS**

Overall, this internship has been enriching with its own ups and downs. I have been tasked with a wide range of responsibilities across different projects and have worked with various teams and people. While I feel the training could be improved, it’s understandable given that we entered the project at a critical point, nearing its go-live phase. Despite this, the team has always been willing to help when I am stuck or have questions, taking their own time to solve issues with me. The rest has been a mix of self-learning, working with other developers, discussing sensitive matters with colleagues, and testing and exploring new things for non-sensitive tasks. Additionally, I check in with my supervisor when I have issues or questions, or just to update on my progress if there are changes to the original plans.

Reflecting on my current standing, I believe I am doing quite well. Not many interns have the opportunity to be part of different projects within five months and handle multiple demands. To me, it shows that people value my contributions and consider me capable of helping them, even though sometimes I lack the experience to help directly. In such cases, I redirect them to a colleague who can assist.

For advice, the best I can give is to know your limits, avoid being egoistic, and voice out concerns. If you feel overwhelmed with tasks, it is alright to voice your struggles to your colleagues or supervisor rather than suffering in silence, which can impact your health and well-being. Do not be afraid to seek help, as internships are about learning and broadening your views. Being egoistic can negatively impact your internship journey and the project. Put your pride aside, work with your colleagues, and seek their help when needed. A lot more can be accomplished with collaboration than by trying to solve everything alone.

In a project, it is crucial to voice any issues you are facing rather than waiting until the last minute. By then, everyone will be busy dealing with their own tasks and may not have time to help with yours. Hence, if you foresee issues, do not hesitate to speak up.**REFERENCES**

https://www.iras.gov.sg/who-we-are/our-organisation/organisation-structure

[Copy and transform data in Dynamics 365 (Microsoft Dataverse) or Dynamics CRM - Azure Data Factory & Azure Synapse | Microsoft Learn](https://learn.microsoft.com/en-us/azure/data-factory/connector-dynamics-crm-office-365?tabs=data-factory)

[Unified Agent Configuration Parameters (mend.io)](https://docs.mend.io/bundle/unified_agent/page/unified_agent_configuration_parameters.html)  
[Kusto Query Language (KQL) overview - Azure Data Explorer & Real-Time Analytics | Microsoft Learn](https://learn.microsoft.com/en-us/azure/data-explorer/kusto/query/)  
[Flux for Helm Users | Flux (fluxcd.io)](https://fluxcd.io/flux/use-cases/helm/)  
[Continuous integration and delivery - Azure Data Factory | Microsoft Learn](https://learn.microsoft.com/en-us/azure/data-factory/continuous-integration-delivery)  
[lighthouse-ci/docs/getting-started.md at main · GoogleChrome/lighthouse-ci (github.com)](https://github.com/GoogleChrome/lighthouse-ci/blob/main/docs/getting-started.md)

**APPENDIXES**

<At a minimum, you are required to attach the following documents:

* Internship Report Clearance Form>

Internship Report Clearance Form

<Attach file here>