Chapter 3
IMPLEMENTATION OF INTERN WORK

3.1 Position and Organization

The internship as a Technical Consultant and Developer Intern at PT. Crubiks Global Indonesia started on the 24^{th} of June 2021 and lasted for 6 months. The company as a whole operates virtually, mainly in a work-from-home manner, all of the meetings done and the tasks required were delivered and executed through an online medium.

PT. Crubiks Global Indonesia expands clientele one at a time in which when the company attracts a potential client, they will suggest a solution and come up with a proposal, from here the potential customer can opt to accept it and hire the company or dismiss the said proposal. Throughout the duration of the internship at PT Crubiks Global Indonesia, multiple clients proposed projects to the company, however, the tasks done as an intern were projects that were internal or long term existing projects, this includes *Web Design*, *Robotic Process* Automation, and *Configuring Cloud Infrastructure*.

3.2 Tasks Done

Throughout the duration of the internship, there were three main projects: Project Froyo, a project with Mastro Luxe Pte. Ltd and the ongoing long-term project with PT Asuransi Tugu Pratama Indonesia Tbk. The specific tasks for each project are as elaborated below:

3.2.1 Designing a Promotional Website for Project Froyo

Project Froyo is a project that will produce a Point-of-Sales (PoS) system. This PoS system will be a competitor to local brands such as Majoo, Moka or Pawoon. In order to prepare for the product, one aspect to be created is the promotional website. Therefore, in the early weeks of the internship, a promotional website meant to be used for potential customers of the POS system was made. The website would have pages for the landing home page, logging in users, registering users, showing pricing plans, and a contact page. Pages for this website were designed using Oxygen Builder

and deployed on a website using WordPress as the content management system.

3.2.2 Implementing RPA for PT. Asuransi Tugu Pratama

PT. Asuransi Tugu Pratama (PT. Tugu) is an insurance company based in Indonesia that wanted PT. Crubiks Global Indonesia automates its procedures when processing insurance policies. There were a total of 13 policies that PT. Tugu wanted PT. Crubiks Global Indonesia to automate, the processes include sorting, filtering, manipulating, transforming, and uploading data. The method of automating processes in this case would be through Robotic Process Automation (RPA). RPA in general imitates human processing, and the overall goal of using RPA is to perform the aforementioned human processing more efficiently [3]. Furthermore, RPA allows for re-usability, by using RPA, processes with similar requirements can use the came robots and therefore perform with less resources needed. When fulfilling the requirements of the project for PT. Tugu, in order to automate the policy processing, the main software of UiPath was used. The tasks of implementing RPA for PT. Tugu included all the steps of software development, from planning to developing, testing, and getting feedback. The main development flow of each policy is as shown in Figure 3.1.

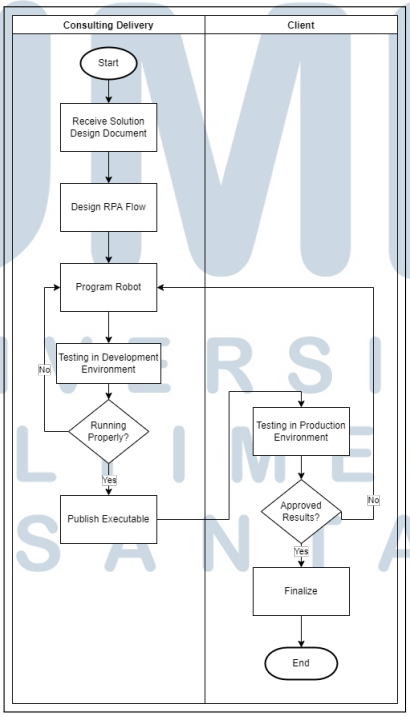


Figure 3.1. Activity diagram of implementing RPA for PT. Asuransi Tugu Pratama

3.2.3 Configuring Cloud Infrastructure for Project Froyo

Continuing on with Project Froyo. The POS system will require servers to be deployed for each tenant. These servers will be hosted on Hetzner. Hetzner is a cloud provider based in Germany that provisions bare-metal servers. To deploy servers more efficiently, infrastructure as code (IaC) software tool called Terraform is used. Terraform is then used to deploy an orchestration system such as Kubernetes (K8s) and Lightweight Kubernetes (K3S). This configuration of cloud infrastructure was done for a large portion of the internship period.

3.2.4 Web Scraping using RPA for Mastro Luxe Pte. Ltd

The last project worked on in PT. Crubiks Global Indonesia was a project with Mastro Luxe Pte. Ltd, commonly referred to as Mastro Luxe. Mastro Luxe is a resale platform that sells authentic pre-owned designer goods. For this project, Mastro Luxe wants a web scraping robot to scrap a website called Vestiaire Collective. The robot that is developed extracts data such as product category, brand, name, price, and many more. Additionally, the data extracted will be stored in a Microsoft SQL Server database. After data storage, the robot should also be able to capture technical errors and run retrying sequences.

3.3 Description of Internship Activities

Execution of weekly intern work for a span of 6 months, or 24 working weeks is as outlined in the Table 3.1,

Table 3.1. Assigned tasks done each week during the internship duration

Week #	Tasks Performed
1	Introduction to basic technologies for the projects necessary in Crubiks.
2	Design and implement Oxygen Builder on Project Froyo's Word-press website.

Table 3.1 Assigned tasks done each week during the internship duration (Cont.)

Week #	Tasks Performed
3	Design and implement Oxygen Builder on Project Froyo's Wordpress website.
4	Introduction to RPA tools and understanding the requirements.
5	UiPath implementation on MAF and MCF policies.
6	UiPath implementation on MAF and MCF policies.
7	UiPath implementation on SKBF policy.
8	UiPath implementation and improvement on MTF, Lotte, FUSE, and JTO Policies.
9	UiPath implementation and improvement on Multiple and Single Policies.
10	Improve and bug fixing policy robots.
11	Created upload robot for safeguarding incomplete and wrongly formatted data.
12	Introduction to Hetzner cloud resources, Git, Ubuntu OS, Kubernetes and Odoo.
13	Implementing Kubernetes and Terraform for Project Froyo.
14	Improve and bug fixes on policy robots after receiving client feedback.
15	Introduction to K3S and Cilium.
16	Terraform implementation of multi-cluster servers on Hetzner.
17	Terraform implementation of multi-cluster servers on Hetzner.

Table 3.1 Assigned tasks done each week during the internship duration (Cont.)

Week #	Tasks Performed
18	Customizing Terraform implementation of multi-cluster servers on Hetzner.
19	Configuring Hetzner firewall.
20	Configuring Load balancers on Hetzner.
21	Implementation of Cilium in K3S.
22	Finalizing infrastructure of Project Froyo in the development environment.
23	Introduction to web scraping using RPA, and understanding Mastro Luxe requirements.
24	Implementing web scraping using RPA.

3.3.1 Introduction to Basic Technologies

During the first week, an Introduction of the technologies used in PT. Crubiks Global Indonesia was introduced. These technologies included productivity tools, programming software, programming tools, and frameworks. Most notably, the main technologies introduced due to their importance are:

- Microsoft Teams: Used for the daily scrum meeting
- Rocket Chat: Daily communication platform
- Wordpress: Content management system used for Project Froyo's website
- Oxygen Builder: Code-less website builder used for Project Froyo's Website

In addition to tools and software that will be used, some concepts were introduced, these concepts include the agile methodology, specifically the scrum model of the agile methodology. The scrum workflow basically

works in sprints in which a scrum master takes charge of leading the team to develop a viable product for the product owner. In most cases, the sprint takes 4 steps; *Plan, Build, Test, and Review*[4]. This is applied in PT. Crubiks Global Indonesia, there is a plan which usually takes place during the daily scrum meeting, and each sprint that is run, the team will improve the product that is being worked on based on the review that was given.

3.3.2 Design and Implementation of Oxygen Builder on Wordpress

Following the introduction to the new technologies, the assigned task was not of high difficulty. This task was to design and deploy a website for Project Froyo. Project Froyo runs its website in the development environment of PT. Crubiks Global Indonesia's main website and uses WordPress as its content management system. Therefore, Project Froyo's website will also be hosted using WordPress. To ease in development, the Oxygen Builder plugin on WordPress is used, an example of the oxygen builder is shown in Figure 3.2, this figure shows the inner-workings behind the landing page of Project Froyo. Through implementing design principles and going through Project Froyo's competitor's websites, the final product was created. The main pages developed were the landing page as shown in Figure 3.3, the pricing page that will host the pricing table of the final product, as seen in Figure 3.4, and lastly the contact us page that has the basic contact information of the company, and a contact form that potential customers can use to get in contact with the company, this is shown in Figure 3.5.

Figure 3.2. Landing page for Project Froyo, with Oxygen Builder plugin

Figure 3.3. Final landing page for Project Froyo

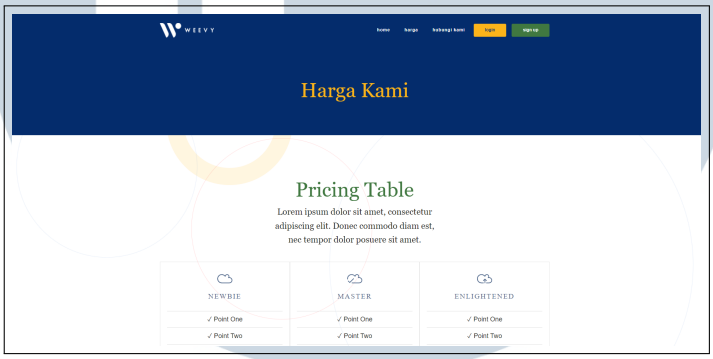


Figure 3.4. Final pricing page for Project Froyo

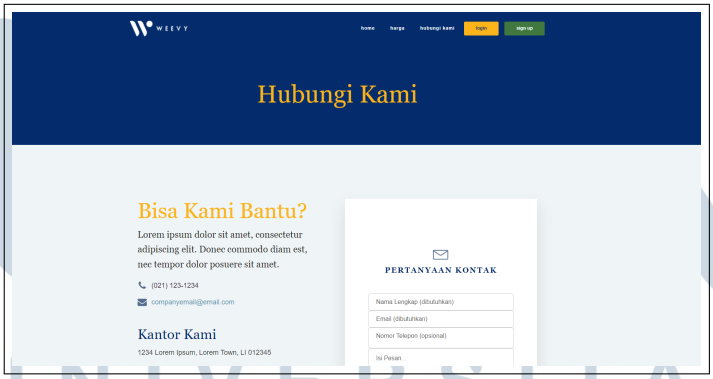


Figure 3.5. Final contact us page for Project Froyo

3.3.3 UiPath Implementation on Insurance Policies

The next assigned tasks are based on an existing contract with PT. Asuransi Tugu Pratama Indonesia Tbk. (PT. Tugu). PT. Tugu wants to automate their policy processing and therefore using the Robotic Processing Automation (RPA) tool of UiPath was used. There were a total of 13 policies

and each has a desired flow as requested by the client. The robot developed extracted, filtered, modified, formatted, and uploaded the data served. Generally, each policy will receive data from a source, this source can be either through emails, or a shared drive. Consequently, the file is saved and the data from the file is extracted. Once the data is extracted, it is processed and modified based on the needed output. Lastly, the altered or modified data is written down onto a template file and saved. For most of the policies, the last step would be to upload the final file onto the client's system, accessed through a website. This common flow that is seen in all the policies is as depicted in Figure 3.6.

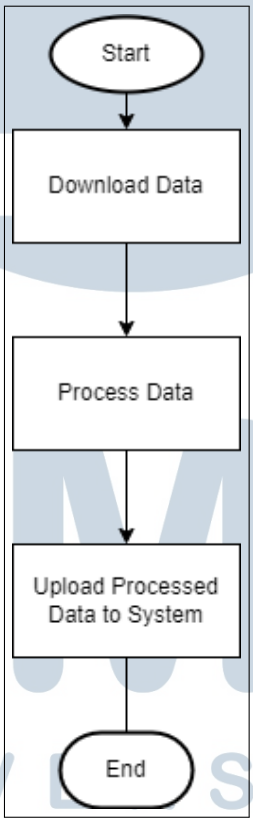


Figure 3.6. Overall RPA flow for PT. Tugu's policies

As previously mentioned, there are a total of 13 policies that needed automation, the 13 policies are for different companies that PT. Tugu represents and covers for. The company names were abbreviated for privacy purposes and referred to by their abbreviations, for example, BAF, MAF, MCF, MTF, JTO, SKBF, etc. From these 13 policies, the data that is pro-

cessed can be divided into two main categories, these are policies with data retrieved from a shared drive and policies with data retrieved from emails. For the most part, to be efficient with the processes, reusable processes are used. For example, with the email retrieved policies, a universal robot processing segment is used. This robot opens the email, gets the email with the matching subject, and saves the data to the local storage for further processing. However, if there are no emails matching the subject, an error will be caught and the whole robot will be stopped. This depiction of the reusable process is seen in Figure 3.9.

On the other hand, with policies that retrieve data from a shared drive, an example of that would be the BAFR2 and BAFR4 policies. These policies follow the flowchart as shown in Figure 3.7 and Figure 3.8, however, those given figures show the flowchart that shows all the steps. While in reality, the data retrieval and folder creation processes can be made into another universal robot process that is called first when the main flow is called, which is what is applied for the BAFR4 policy. The separation of processes or modularization of robotic processing is highly beneficial for intricate processing as it reduces the redundancy in development [5], and introduces an efficient break point for long processes. It shall be noted, however, that modularity is not always the best approach, there are certain processes that become less efficient when taken apart into multiple processes [6]. An example of this is when the variables used are clashing because they are too specific or even too generic. Another example would be when throwing different exceptions [6], these exceptions become ineffective and might even cause errors due to them overwriting each other. To make an example out of the processes done for PT. Tugu, with the robot for the BAFR2 policy as depicted in Figure 3.7, this was implemented as it is, without the separation of the data retrieval due to the simplicity of the policy as a whole. Separating the process will reduce the speed of the processing as it will need to traverse across robots and processes.

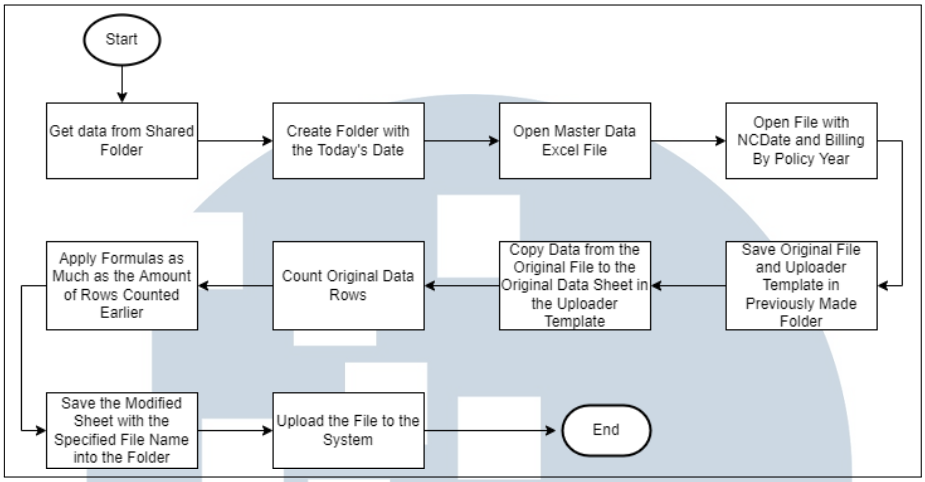


Figure 3.7. BAF for motorcycles insurance policy flowchart

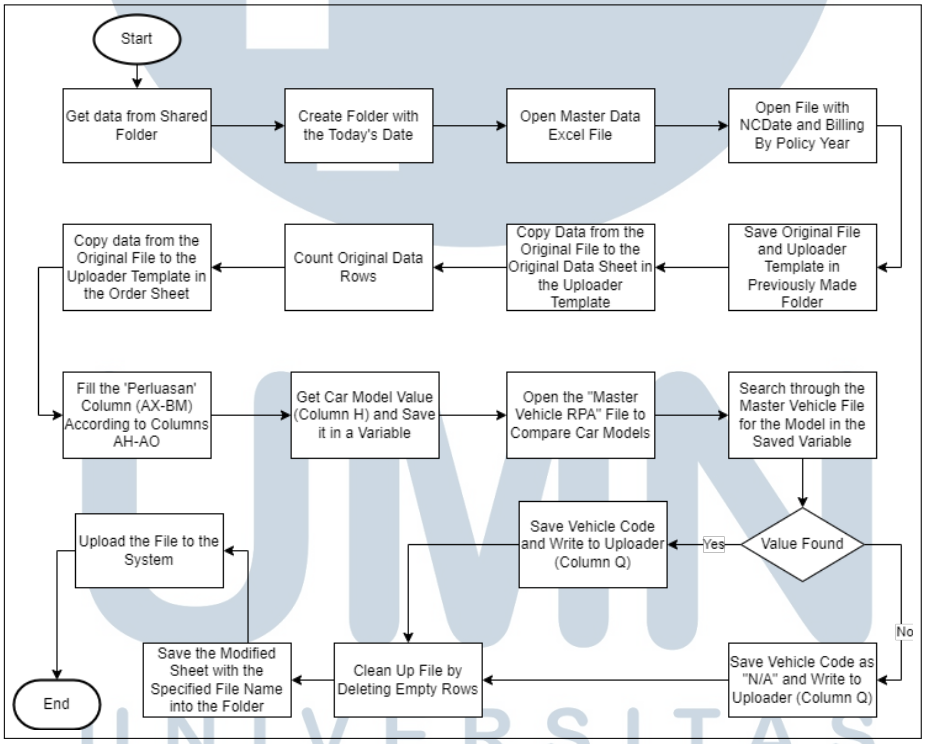


Figure 3.8. BAF for cars insurance policy flowchart

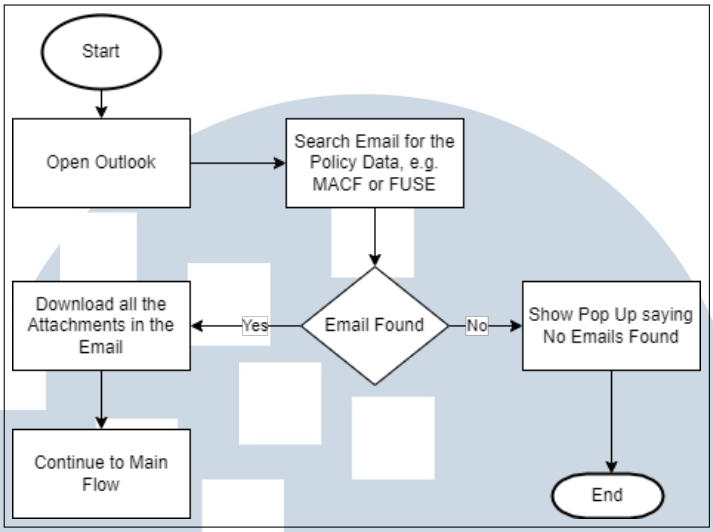
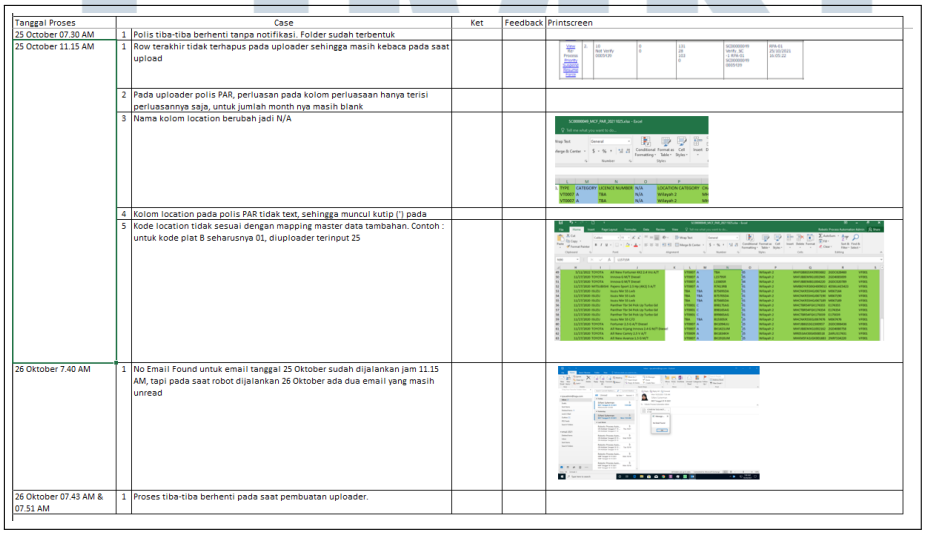


Figure 3.9. Reusable upload flow for MCF, MAF, JTO, SKBF, Fuse, Single policies

3.3.4 Improvement and Bug Fixes in RPA

With the implementation of the agile methodology, whenever a policy passes the testing in the development environment, it is sent to the production environment and tested by the client. When the client tests the robots and finds an error, they will update an excel sheet that shows when it was tested, what happened, a description, and a screenshot showing what needs to be fixed. One excel sheet denotes the error and bugs for one policy, an example of this is shown in a screenshot as depicted in Figure 3.10.



Tanggal Proses		Case	Ket	Feedback	Printscreen
25 October 07.30 AM	1	Polis tiba-tiba berhenti tanpa notifikasi. Folder sudah terbentuk			
25 October 11.15 AM	1	Row terakhir tidak terhapus pada uploader sehingga masih kebaca pada saat upload			
	2	Pada uploader polis PAR, perluasan pada kolom perluasaan hanya terisi perluasannya saja, untuk jumlah month nya masih blank			
	3	Nama kolom location berubah jadi N/A			
	4	Kolom location pada polis PAR tidak text, sehingga muncul kutip (') pada			
	5	Kode location tidak sesuai dengan mapping master data tambahan. Contoh : untuk kode plat B seharusnya 01, diuploader terinput 25			
26 Oktober 7.40 AM	1	No Email Found untuk email tanggal 25 Oktober sudah dijalankan jam 11.15 AM, tapi pada saat robot dijalankan 26 Oktober ada dua email yang masih unread			
26 Oktober 07.43 AM & 07.51 AM	1	Proses tiba-tiba berhenti pada saat pembuatan uploader.			

Figure 3.10. Error log in an excel sheet with necessary details to guide fixes

3.3.5 Additional RPA Robot Creation

When a robot fails to upload due to a wrong format, or a single missing cell, it closes the robot in its entirety. This abrupt stop can be inefficient, especially if the data processing is very large, it might even lead to the corruption of files if not handled properly. Therefore, a solution is required where even if the upload is rejected, there is an option in which reprocessing or running the robot from the start will not be necessary. This solution is to create another separate robot whose purpose is simply to upload data without reprocessing it. The flow that the robot is shown in Figure 3.11. This robot works away from the main program of the policy robot and should be run separately. At the start, the user is able to choose what policy they are correcting and the files to upload. Once these variables are set, the flow will run and the data will be immediately uploaded rather than reprocessed.

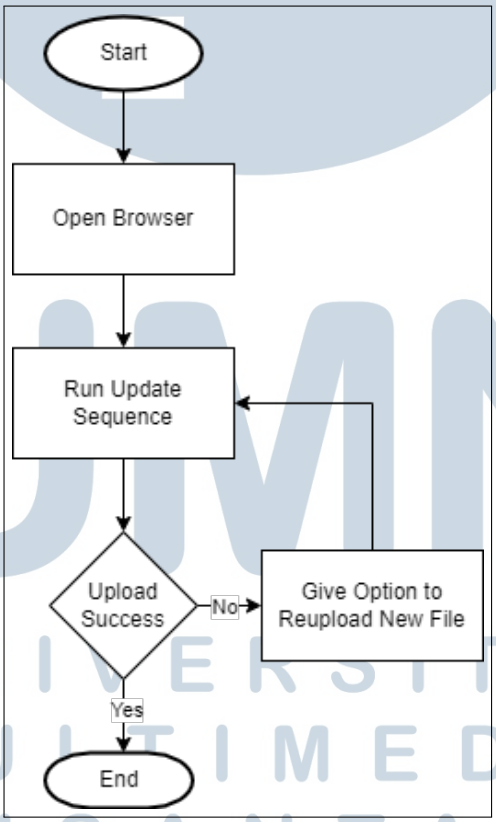


Figure 3.11. Separate upload robot to reduce reprocessing

3.3.6 Introduction to Infrastructure Technologies

As introduced earlier, Project Froyo is a long-term internal project that is being developed. The project as a whole is a PoS system that will compete with other PoS systems in Indonesia. One aspect of the project however is the underlying infrastructure. Project Froyo will run its back-end on the cloud, this means that all the data and the servers spun up for the tenants, will all be available on the cloud rather than the conventional on-premise option. With that being said, the development of said infrastructure was developed throughout the duration of the internship. Nevertheless, the task assigned was the implementation of the infrastructure plan onto a lightweight Kubernetes Environment (K3s), and the improvement of a developed Kubernetes (K8s) infrastructure. Before the development and implementation were executed, a few technologies, tools, and concepts needed to be understood, these included:

- Kubernetes (K8S): Kubernetes or usually referred to as K8s is a platform that manages workloads that have been containerized [7]. It is used in Project Froyo to manage the servers that run on Hetzner.
- Lightweight Kubernetes (K3s): A more lightweight version of K8s whose main difference with the K8s is that it is purpose-built rather than general-purpose container orchestrators [7].
- Terraform: Infrastructure as Code software that utilizes a declarative programming language to define configurations for data center infrastructure. This tool is the main tool behind the development of the infrastructure of Project Froyo.
- Cilium: Cilium is a software that is open-source that provides an underlying medium in terms of the network connectivity between containerized workloads in cloud environments. This is used in Project Froyo to connect the different clusters of servers that are provisioned.
- Azure DevOps: A Microsoft product that helps with project management, version control testing, and automated builds [8]. This is used to host the repository and perform the automated deployment of newer versions of configurations of Project Froyo.

- Helm Charts: Helm charts are Kubernetes manifests that are in YAML format. They help with defining, installing, and upgrading applications that are Kubernetes based [8].
- Hetzner Cloud: The cloud provider that will deploy all the servers used in Project Froyo. Hetzner also has load balancing and firewall features that can be adjusted and set using Terraform.

3.3.7 Terraform Implementation on K3s and K8s

Terraform is highly effective in configuring data centers, it is the IaC of choice when deploying servers on the Hetzner cloud in Project Froyo. The infrastructure plan of the PoS system is as depicted in Figure 3.12. As shown in the diagram, the worker nodes fall under the master node will be the central control plane of the cluster. Each cluster that is developed will be configured using Terraform, and deployed on Hetzner. These configurations are declared in a declarative way using the syntax that is determined by both Terraform and Hetzner. Each cluster also has specific configurations and variables that can be set in the Terraform file. For example, a configuration of a cluster that is deployed is as shown in Figure 3.13. In this example, it can be seen the number of servers that will be deployed in each location of Hetzner data centers, the type of machine that will become the masters and the worker nodes, and the location of the floating IP that will be used for each cluster. These variables and more can be set in the Terraform file and then run on Azure DevOps automatically after it is approved. This automated process also helps in rolling back and keeping version control. Once the servers are provisioned and running, it can be seen in the Hetzner console as seen in Figure 3.14.

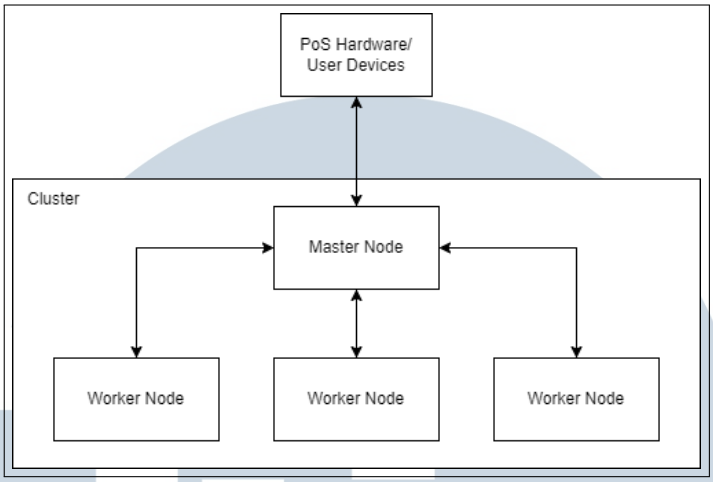


Figure 3.12. Simple infrastructure map of Project Froyo in the development environment

```
locals {
  first_control_plane_network_ip_001 = cidrhost(module.cluster-001-network.hcloud_network_subnet_range, 2)
  master_location_001 = "nbg1"
  control_plane_server_type_001 = "cx11"

  master_keep_disk_choice_001 = "true"
  keep_disk_choice_001 = "true"

  load_balancer_availability_001 = "1"

  fsn_agents_num_001 = 1
  nbg_agents_num_001 = 1
  hel_agents_num_001 = 1
  #ash_agents_num_001 = 1

  fsn_node_worker_type_001 = "cx11"
  nbg_node_worker_type_001 = "cx11"
  hel_node_worker_type_001 = "cx11"

  floating_ip_location_001 = "hel1"

  cluster_number_001 = 1
}
```

Figure 3.13. Example configuration of a cluster to be deployed on Hetzner using Terraform

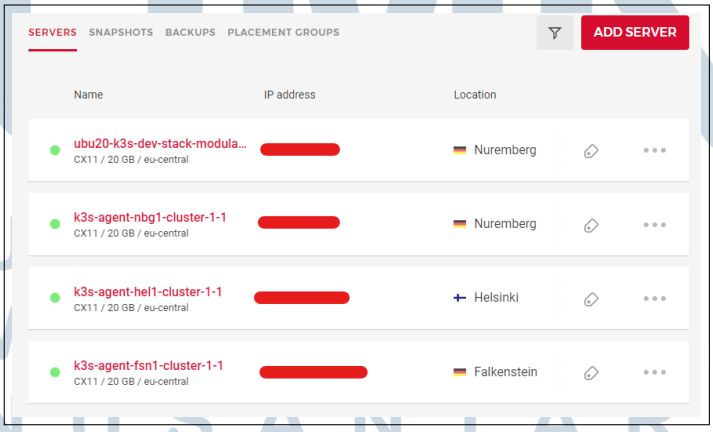


Figure 3.14. Provisioned cluster that is made up of 4 servers on the Hetzner cloud console

After performing the initial spin-up and provisioning the servers and clusters, other factors of the infrastructure need to be considered. This

includes the network connectivity, load balancing, and security aspects. Firstly, the network connectivity between multiple clusters is managed by Cilium. By using Cilium's multi-cluster technology, known as ClusterMesh, routing across clusters can be efficiently improved by utilizing tunneling or direct routing that does not need any proxies. Additionally, communication that is done between nodes will be transparently encrypted. The effectiveness of Cilium is highly recommended and therefore implemented in the configuration of Project Froyo's infrastructure. This was implemented in Project Froyo by first running an installation of Cilium, then applying the rules on the Kubernetes cluster's control plane. From there, the configuration will be either a success or a failed installation. The results of a running Cilium initial configuration are seen in Figure 3.15. After the initial configuration, the enabling of the ClusterMesh can simply be done through a BASH command, and the final connection of all the clusters, if needed, can be visualized by a tool called Hubble.

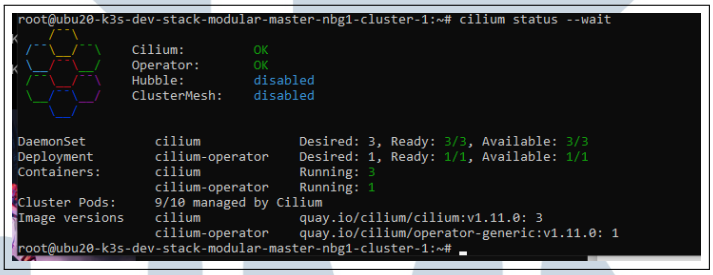


Figure 3.15. Command line screenshot of successful Cilium deployment

Once all the parts are connected, it is essential that the servers that were provisioned have a load balancer. This load balancer ensures that the data is directed to the correct ports using the correct protocols. Additionally, the load balancer has to work in conjunction with the firewall to ensure that the data packets that come in and out of the environment are monitored and safe. The configuration of both the load balancer and the firewalls are done in the Terraform files that were first run when a new cluster is initialized. An example of these configurations can be seen in Figure 3.16 for the load balancer configuration, and Figure 3.17 for the firewall configuration.

```
resource "hcloud_load_balancer" "load_balancer" {
  #count = var.load_balancer_availability ? 1 : 0
  name       = "load-balancer-cluster-${var.cluster_number}"
  load_balancer_type = "lb11"
  location   = var.location
}

resource "hcloud_load_balancer_network" "lbnetwork" {
  load_balancer_id = hcloud_load_balancer.load_balancer.id
  network_id       = var.hcloud_k3s_network_id
}

resource "hcloud_load_balancer_service" "kube_load_balancer_service"{
  load_balancer_id = hcloud_load_balancer.load_balancer.id
  protocol = "tcp"
  listen_port = 6443
  destination_port = 6443
}

resource "hcloud_load_balancer_target" "load_balancer_target" {
  type             = "label_selector"
  load_balancer_id = hcloud_load_balancer.load_balancer.id
  label_selector   = "env = cluster-${var.cluster_number}"
}
```

Figure 3.16. Source code of the load balancer configuration for a cluster in Hetzner

```
resource "hcloud_firewall" "k3s" {
  name = "k3s-firewall"

  # Internal cluster traffic, kube api server, kubelet metrics, cilium, etcd,
  # and Hetzner metadata service and cloud api
  rule {
    direction = "in"
    protocol  = "tcp"
    port      = "any"
    source_ips = [
      "127.0.0.1/32",
      "10.0.0.0/8",
      "169.254.169.254/32",
      "213.239.246.1/32"
    ]
  }
  rule {
    direction = "in"
    protocol  = "udp"
    port      = "any"
    source_ips = [
      "127.0.0.1/32",
      "10.0.0.0/8",
      "169.254.169.254/32",
      "213.239.246.1/32"
    ]
  }
  rule {
    direction = "in"
    protocol  = "icmp"
    source_ips = [
      "127.0.0.1/32",
      "10.0.0.0/8"
    ]
  }
```

Figure 3.17. Source code of the firewall configuration for a cluster in Hetzner

3.3.8 Web Scraping using RPA for Mastro Luxe

For the final task assigned, an RPA project with Mastro Luxe was given. The same tool as the previous RPA project was used, this tool was UiPath, as shown in the example screenshot of the interface shown in the UiPath program in Figure 3.18. *Web Scraping* as a process can also be called web extraction, as it is mentioned in the name, this process involves data from a website being extracted into data storage for future use or future data retrieval [9].

In this specific project, the task was to extract information regarding various fashion products from Vestiaire Collective, or the website www.vestiairecollective.com. The project aims to retrieve the information about the specific products, this includes the product name, description, price, dimensions, and identifying pictures. The information is retrieved by identifying the web elements and therefore fetching the values assigned for said element. After extraction, the robot will store the data in a database, in this case, a Microsoft SQL Server database. This sequence of activities is seen in the flowchart shown in Figure 3.19.

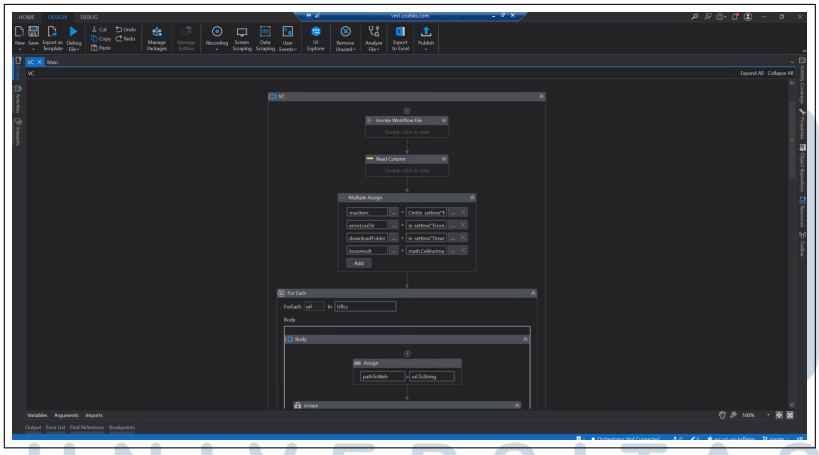


Figure 3.18. Example of the UiPath interface as used in the project with PT. Tugu and Mastro Luxe

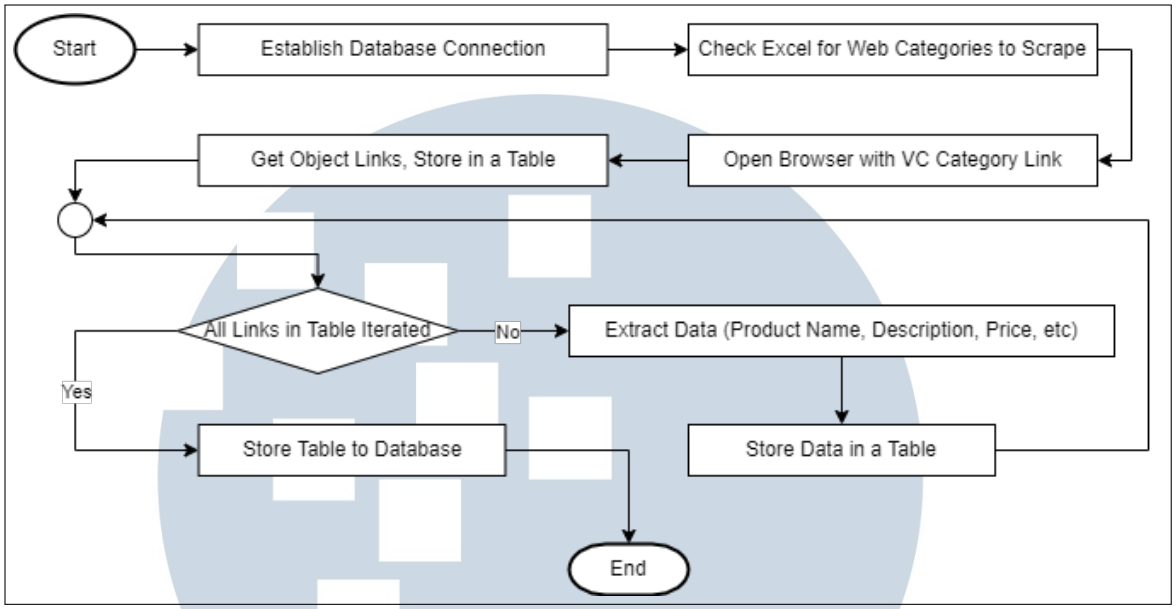


Figure 3.19. Flowchart of activities involved in the Vestiaire Collective web scraping project for Mastro Luxe

To dive deeper and portray an example of the application of RPA to automate the web scraping process on the Vestiaire Collective website. The first thing the robot does is to establish a database where the data will be stored, and establish a connection to it, as shown in Figure 3.20. Additionally, as shown in the flowchart as well as in Figure 3.20, an excel sheet containing the configuration values for the web scraping activities is extracted and stored in the variables.

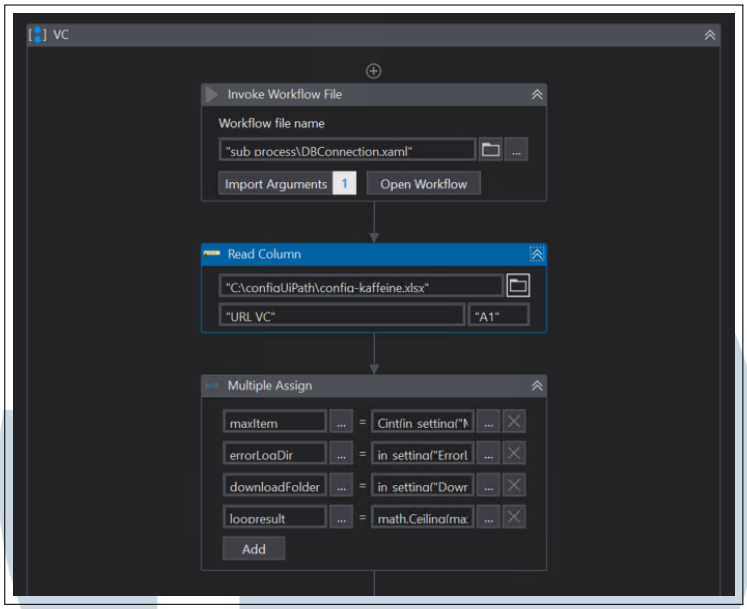


Figure 3.20. UiPath workflow and activity blocks of database establishment and excel configuration data extraction

3.4 Obstacles and Solutions

The internship at PT. Crubiks Global Indonesia was an enthralling experience that taught a lot of valuable lessons, may it be in terms of technical skills and knowledge, or soft skills that can be applied in the workplace. However, the experience was not short of obstacles. There were a few obstacles that were faced when developing the solutions for the clients, and also during the development period of the internal projects. With that being said, the obstacles that were faced were solved and solutions were formed to overcome them. These solutions as a whole helped in finalizing the projects and also in the development of one's character that serves as a lifelong lesson.

3.4.1 Obstacles Faced

To fully understand the solutions that were developed, a brief understanding of the obstacles faced is necessary. As explained earlier, there were three main projects that were worked on during the duration of the internship. This was the web and cloud infrastructure project for Project Froyo and the RPA projects with PT. Tugu and Mastro Luxe Pte. Ltd. As elaborated below, these are the points that describe the obstacles faced in each project.

A. Project Froyo

- Unfamiliarity with Cloud Infrastructure technologies causing a delay in the execution of tasks.
- Fear of making a mistake and causing disorder in the existing infrastructure this hindering progress.

B. RPA for PT. Tugu

- Unfamiliarity with UiPath and its features.
- PT. Tugu having server outages, this hinders progress as the process of building the robots are done through a remote desktop.
- Waiting long periods of time for a robot to run due to the large size of data that is to be processed.
- Robots stopping unexpectedly if both the host computer and remote accessed computer is interacted with.

C. RPA for Mastro Luxe

- Usage of a remote desktop that does not have local processing speed.
- Unfamiliarity with Angular based web elements causing extra hours needed to complete the project
- Disparity in knowledge when dealing with the database, this project uses the Microsoft SQL Server.
- Short time span to deliver satisfying results.

3.4.2 Solutions to the Obstacles Faced

From the obstacles that were faced, a set of solutions were made to overcome them. These solutions helped in finishing the projects in a satisfactory manner to suit not only the company's preference but also the clients' requirements. The solutions below explain the actions done to overcome the obstacles mentioned before for each of the projects.

A. Project Froyo

- Taking the time to learn the technologies during off-work hours.
- Utilizing the git CI/CD and the rollback feature.

B. RPA for PT. Asuransi Tugu Pratama Indonesia Tbk and Mastro Luxe Pte. Ltd

- Experimenting with UiPath Studio during off-work hours.
- Do the system designs away from the remote desktop and apply it only when the servers are back offline.
- Cut down the size of the data for preliminary testing to avoid wasting time.
- Use a second monitor, as the problem is the robot has to run on top of other visible processes, therefore having it seen allows it to use more processing power.