## Acknowledgement

I would like to deeply acknowledge my distinguished supervisors, Mr. Supervisor and Mr. Supervisor, for guiding and providing me with any kind of support needed through the various phases of the project.

Mr. Supervisor's oversight has been essential in providing academic direction and guaranteeing the project satisfies all requirements. Similarly, Mr. Supervisor's supervision and skills, with the important industry insights and a wider viewpoint he provided have helped me improve my project idea a lot. My sincere appreciation goes out to both supervisors for their encouragement and assistance in exploring new research directions, as well as their continuous and insightful feedback.

Finally, I would like to express my profound appreciation to Islington College for providing me with an opportunity to explore, learn and work in my field of interest.

## Abstract

The increasing importance of computer networks has increased the need for effective and user-friendly network monitoring and packet analysis tools. Even though strong tools like Wireshark and Zeek are currently available, individuals with limited technical knowledge may find them challenging to utilize due to their complexity. In order to close this gap, this project proposes developing the Network Packet Visualizer Tool, a web-based program that offers an easy-to-use interface for network monitoring and packet analysis.

The application uses Zeek-generated logs to extract useful information and displays them in an easy-to-understand dashboard that is easy to use for both network experts and regular users. Real-time data visualization, threat detection, warnings, and notifications for anomalous network activity are some of the key features.

Using the Agile methodology's Scrum framework, the project focuses on adaptability, transparency, and iterative development. By improving network data accessibility and understanding, the end result aims to improve user experience and overall network security. This report covers the goals, objectives, methods, risk management techniques, milestones, and expected outcomes of the project.

## Table of Contents

## Table of Figures

## Table of Tables

# 1. Introduction

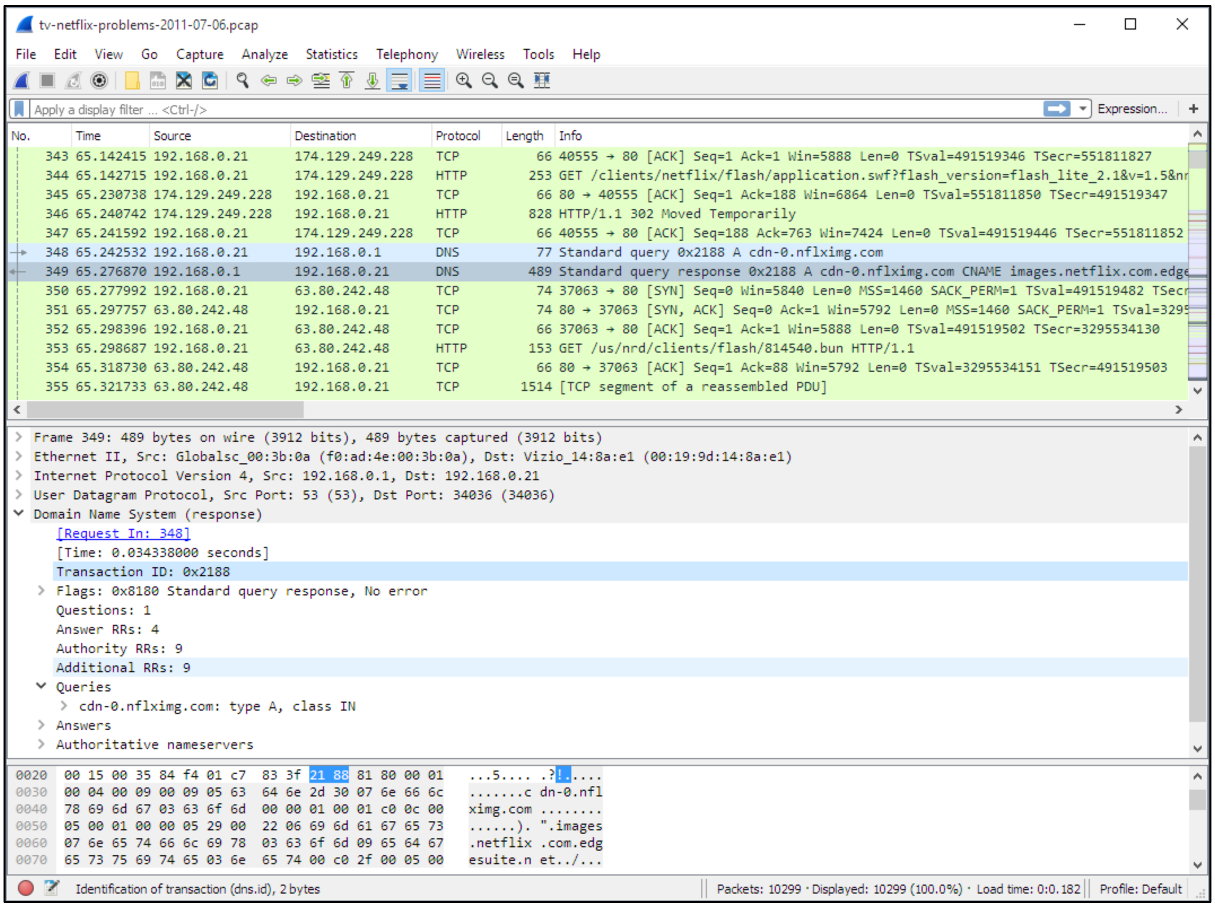
In today's interconnected world, computer networks are the backbone of today's society, allowing for smooth communication, information exchange, and collaboration. In the current state of information technology development, security plays a significant role in computer network applications (Zhang, 2024).

Computer networks are particularly important today for communication and information flow because they allow for the easy transfer of digital data between interconnected computing devices. Computer networks are essentially systems of nodes that are connected in a unique way. Recent improvements in computer, software, and networking technology have led to a significant increase in the amount of time people spend online (H, et al., 2024).

The use of computer networks is now far more widespread than it has ever been thanks to major advancements in the field and the internet's technology, which has expanded at a remarkable pace. Similarly, with the growth of networks, network security is growing more and more crucial as the next most crucial component of a system. Monitoring network traffic and analyzing it is a challenging task that is often handled by highly skilled technical professionals such as network engineers and security analysts (Zhang, 2024).

## 1.1 Problem Scenario

While advanced network analysis can be obtained with existing tools such as Wireshark, users with limited knowledge of network protocols will not be able to benefit from them. To put it another way, understanding how a network functions is essential to being able to use Wireshark correctly (CompTIA, 2024).


tv-netflix-problems-2011-07-06.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.    Time      Source           Destination      Protocol  Length  Info
  343 65.142415 192.168.0.21     174.129.249.228  TCP         66 40555 → 80 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=491519346 TSecr=551811827
  344 65.142715 192.168.0.21     174.129.249.228  HTTP       253 GET /clients/netflix/flash/application.swf?flash_version=flash_lite_2.1&v=1.5&nr
  345 65.230738 174.129.249.228  192.168.0.21     TCP         66 80 → 40555 [ACK] Seq=1 Ack=188 Win=6864 Len=0 TSval=551811850 TSecr=491519347
  346 65.240742 174.129.249.228  192.168.0.21     HTTP       828 HTTP/1.1 302 Moved Temporarily
  347 65.241592 192.168.0.21     174.129.249.228  TCP         66 40555 → 80 [ACK] Seq=188 Ack=763 Win=7424 Len=0 TSval=491519446 TSecr=551811852
  348 65.242532 192.168.0.21     192.168.0.1      DNS         77 Standard query 0x2188 A cdn-0.nflximg.com
  349 65.276870 192.168.0.1      192.168.0.21     DNS        489 Standard query response 0x2188 A cdn-0.nflximg.com CNAME images.netflix.com.edge
  350 65.277992 192.168.0.21     63.80.242.48     TCP         74 37063 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=491519482 TSecr
  351 65.297757 63.80.242.48     192.168.0.21     TCP         74 80 → 37063 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 SACK_PERM=1 TSval=3295
  352 65.298396 192.168.0.21     63.80.242.48     TCP         66 37063 → 80 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=491519502 TSecr=3295534130
  353 65.298687 192.168.0.21     63.80.242.48     HTTP       153 GET /us/nrd/clients/flash/814540.bun HTTP/1.1
  354 65.318730 63.80.242.48     192.168.0.21     TCP         66 80 → 37063 [ACK] Seq=1 Ack=88 Win=5792 Len=0 TSval=3295534151 TSecr=491519503
  355 65.321733 63.80.242.48     192.168.0.21     TCP       1514 [TCP segment of a reassembled PDU]
```

```
> Frame 349: 489 bytes on wire (3912 bits), 489 bytes captured (3912 bits)
> Ethernet II, Src: Globalsc_00:3b:0a (f0:ad:4e:00:3b:0a), Dst: Vizio_14:8a:e1 (00:19:9d:14:8a:e1)
> Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.21
> User Datagram Protocol, Src Port: 53 (53), Dst Port: 34036 (34036)
v Domain Name System (response)
    [Request In: 348]
    [Time: 0.034338000 seconds]
    Transaction ID: 0x2188
  > Flags: 0x8180 Standard query response, No error
    Questions: 1
    Answer RRs: 4
    Authority RRs: 9
    Additional RRs: 9
  v Queries
    > cdn-0.nflximg.com: type A, class IN
  > Answers
  > Authoritative nameservers
```

```
0020  00 15 00 35 84 f4 01 c7  83 3f 21 88 81 80 00 01   ...5.... .?!.....
0030  00 04 00 09 00 09 05 63  64 6e 2d 30 07 6e 66 6c   .......c dn-0.nfl
0040  78 69 6d 67 03 63 6f 6d  00 00 01 00 01 c0 0c 00   ximg.com ........
0050  05 00 01 00 00 05 29 00  22 06 69 6d 61 67 65 73   ......). ".images
0060  07 6e 65 74 66 6c 69 78  03 63 6f 6d 09 65 64 67   .netflix .com.edg
0070  65 73 75 69 74 65 03 6e  65 74 00 c0 2f 00 05 00   esuite.n et../...
```

```
Identification of transaction (dns.id), 2 bytes    Packets: 10299 · Displayed: 10299 (100.0%) · Load time: 0:0.182    Profile: Default
```

*Figure 1: Complexity of Wireshark.*

Similarly, Zeek; an open-source network security monitoring tool, is an advanced tool used for packet/ traffic analysis, it was primarily built for the Unix-based systems and is CLI- based. It produces different types of logs which only an experienced network professional can understand. The logs produced by Zeek can be used to detect abnormal activities in a network. Furthermore, it does not offer any kind of visualization, which makes it difficult for even network professionals to understand.

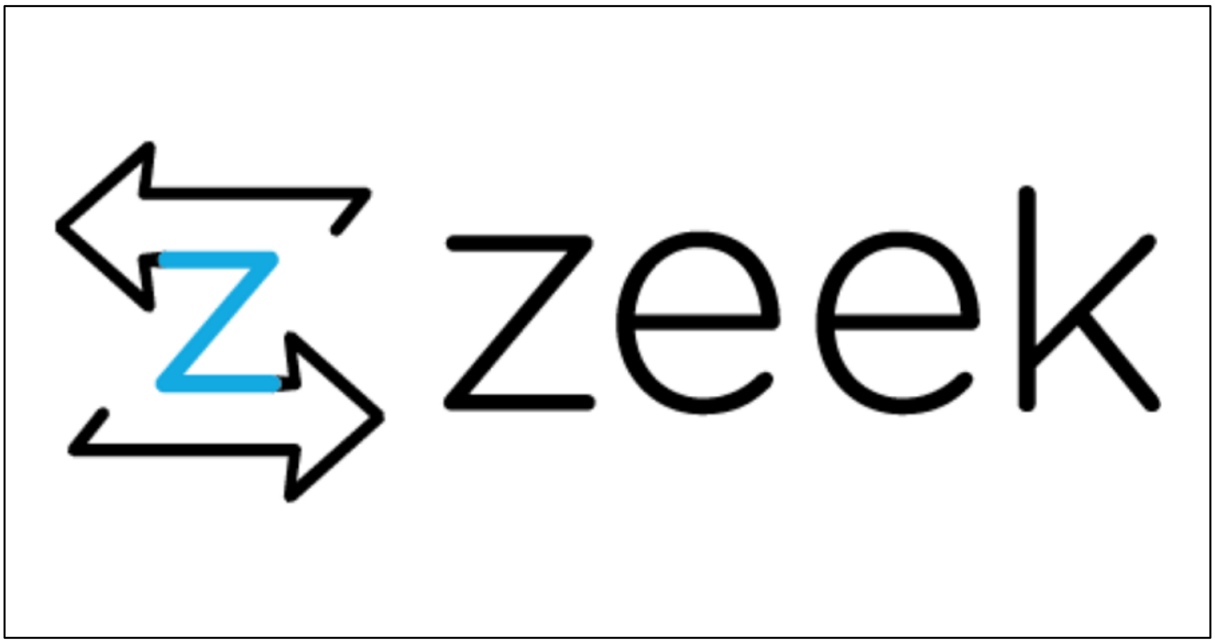


*Figure 2: Zeek*

This leaves room wide open for general users to not be able to see if their network is up, troubleshoot connectivity problems, or even identify if they may have a security threat on their network.

## 1.2 Project as a Solution

In order to bridge the gap, Network Packet Visualizer Tool offers a straightforward web-based application for packet analysis and visualization. The project's goal is to make data from network traffic graphs readable to all consumers, regardless of technical expertise using the logs extracted from network analysis tools such as Zeek.

Network Packet Visualizer Tool will use the logs produced by Zeek to extract the required information and visualize them in a web application which will make packet analysis and network monitoring easier both for the general users as well as network professionals.

An easy-to-use dashboard providing instant access to information on network and bandwidth usage, connected users, and their activities for ordinary users will be provided. Network Packet Visualizer Tool provides analytics features related to threat detection, and comprehensive insight on actual packets for advanced users. It will also send alerts and notifications to users upon detecting unusual network activity. By transforming unprocessed network data into graphical insights, Network Packet Visualizer Tool helps users visualize and monitor their network environment; even for users who have limited knowledge of network.

# 2. Aim and Objectives

## 2.1 Aim

The aim of Network Packet Visualizer Tool is to fill the gap of the existing tools and develop a comprehensive and accessible packet visualization tool that simplifies network monitoring and enhances understanding for both general users and network professionals based on the logs extracted from networking analysis tools like Zeek.

## 2.2 Objectives

i. To develop packet visualization tool that would make network monitoring easier to be understood and to be used by people who are generally not familiar with networks or to the professionals.
ii. To integrate features such as visual representation of parameters like packet loss, current bandwidth and the number of connected devices.
iii. To develop the backend of log parsing using Python.
iv. To implement a system to send alerts and notifications to users upon detecting unusual network activity.
v. To complete all the milestones in the given time ensuring the successful completion of the project.

## 3. Expected Outcomes and Deliverables

The main objective of this project is to develop an all-inclusive tool; Network Packet Visualizer Tool that is capable of packet analysis and visualization. The expected outcome is a web application meant for making packet analysis and network monitoring easier both for the general users as well as network professionals. It is expected that it will have real-time data visualization ability as well as threat detection and advanced analysis features.

### 3.1 Deliverables

The features of Network Packet Visualizer Tool are listed below:

- Dashboard Interface:
  An easy-to-use dashboard providing instant access to information on network and bandwidth usage, connected users, and their activities for ordinary users.

- Tools for Data Visualization:
  Dynamic and interactive visual components, such as charts, and graphs, that present important information in an understandable way, such as device connections, bandwidth utilization, packet flow, and quality of the network.

- Advanced Network Analysis Features:
  Advanced features with real-time threat identification, traffic port analysis, IP reputation, and country flagging designed for network professionals.

- Notification and Alert System:
  A comprehensive alert system designed for users to promptly identify and deal with possible risks by providing real-time alerts and notifications of abnormal network activity.

# 4. Project Risks, Threats and Contingency Plans

Risks and threats may arise throughout project development, as all systems are subject to them. In order minimize systemic defects, the following risks and threats are addressed together with contingency plans.

## 4.1 Project Risks and Threats

Below mentioned are the risks and threats that may arise during the development of the project:

### 4.1.1 Zeek's Functionality in Windows

Zeek was originally designed for Unix. Zeek has an experimental version for Windows, which is not fully reliable.

### 4.1.2 Complex User Interface

General users may find the interface overwhelming or difficult to navigate, leading to poor adoption.

### 4.1.3 Security Vulnerabilities

The application could be vulnerable to cyberattacks, leading to data breaches or unauthorized access.

### 4.1.4 Performance Bottleneck

Visualization tools struggle to display large amounts of data, also delay in data processing which could prevent users from receiving alerts in time.

### 4.1.5 Inadequate Testing

Insufficient testing of the application's features may lead to bugs or usability issues, affecting the final product's quality and user experience.

## 4.2 Contingency Plans

Below mentioned are the contingency plans addressed with respect to the risks and threats mentioned above that may arise during the development process:

### 4.2.1 Zeek's Functionality in Windows

Since Zeek in Windows is an experimental version, switch to Linux to avoid experimenting on the experiment.

### 4.2.2 Complex User Interface

To mitigate the risk of user interface complexity, use general user testing to get feedback during design to make navigating easier. Improve simplicity by navigating the design in response to user feedback.

### 4.2.3 Security Vulnerabilities

Conduct regular security audits. Use up-to-date security software, conduct vulnerability assessments, and have a response plan in place.

### 4.2.4 Performance Bottleneck

Design the architecture for horizontal scaling and optimize backend processing for efficiency. Use light visualization libraries.

### 4.2.5 Inadequate Testing

Adopt a test first approach, where write unit tests for the backend functionality and perform usability testing for frontend features.

## 5. Methodologies

The software development process must follow precise, well-considered procedures in a reliable plan in order to meet the needs necessary for any project to be successful. Software development life cycle (SDLC) methodologies have provided several models that comply with the needs of the various proposed projects. These methodologies offer a range of situations that can be used when developing systems to increase their effectiveness and predictability (ALazzawi, et al., 2023).

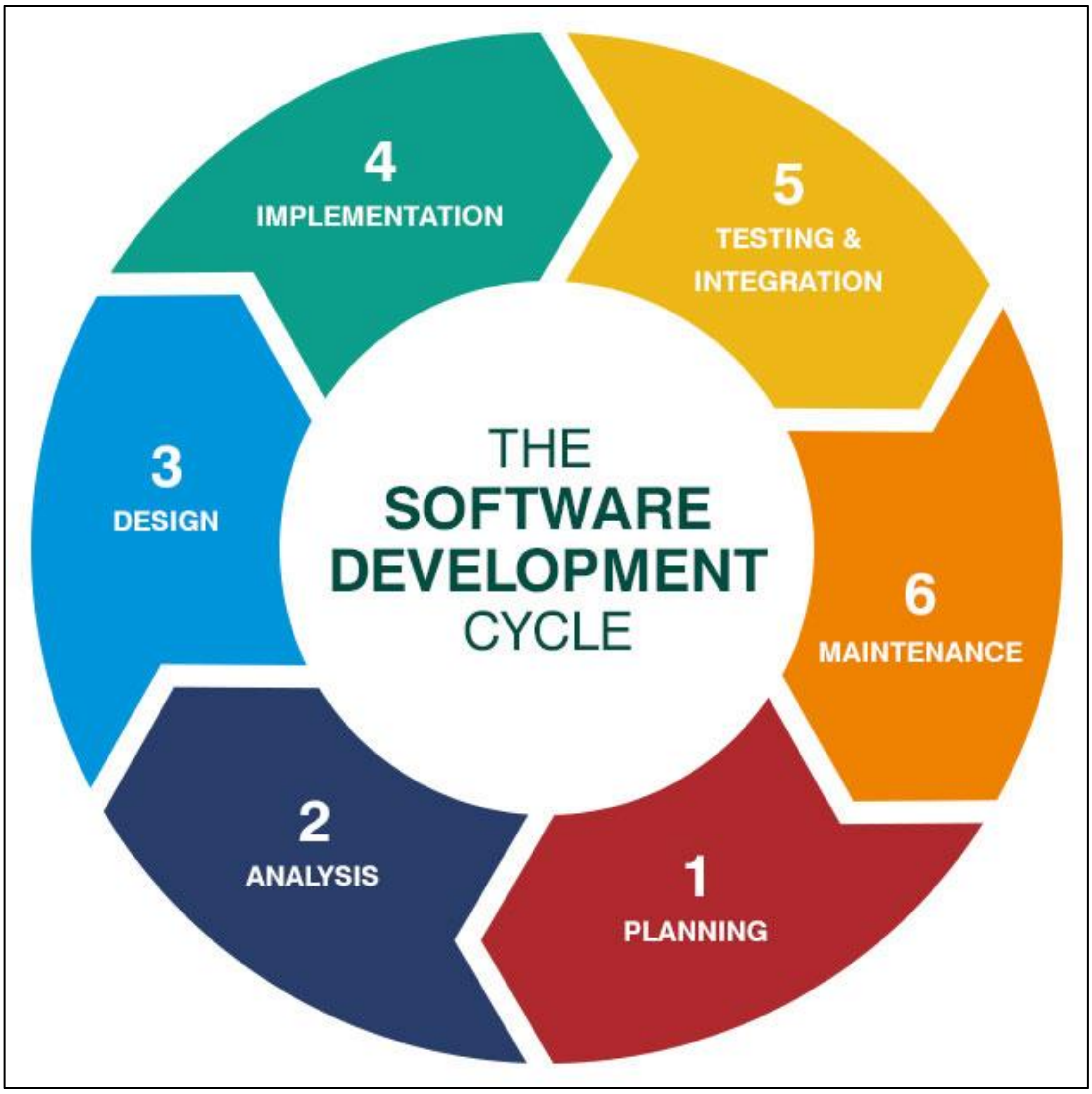


*Figure 3: Software Development Life Cycle (Sandamal, 2020).*

## 5.1 Considered Methodologies

Prior to initiating the project, a variety of methodologies were studied and taken into consideration, with their individual benefits being weighed and matched to the objectives of the project.

### 5.1.1 Agile Methodology

Agile methodology is a series of incremental and iterative techniques that puts adaptability and customer engagement first. Agile projects are broken up into brief iterations known as sprints rather than following a straight line (Hossain, 2024).

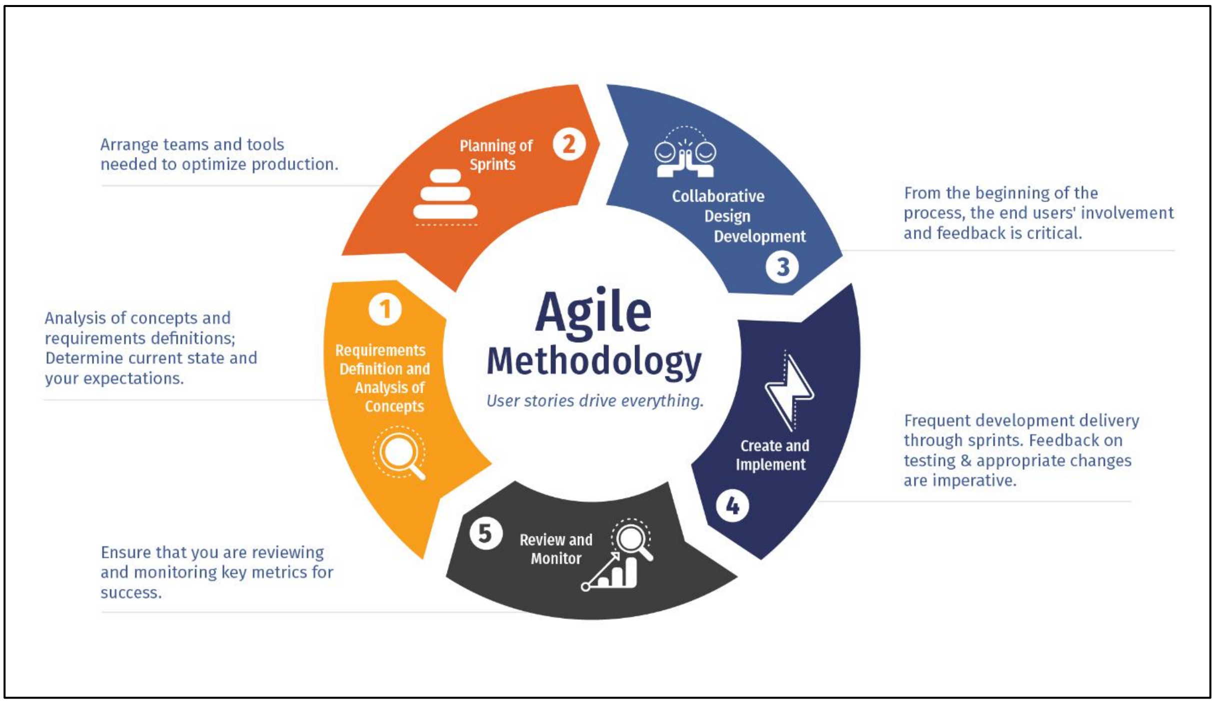

Agile Methodology
User stories drive everything.
1 Requirements Definition and Analysis of Concepts
2 Planning of Sprints
3 Collaborative Design Development
4 Create and Implement
5 Review and Monitor
Arrange teams and tools needed to optimize production.
From the beginning of the process, the end users' involvement and feedback is critical.
Analysis of concepts and requirements definitions; Determine current state and your expectations.
Frequent development delivery through sprints. Feedback on testing & appropriate changes are imperative.
Ensure that you are reviewing and monitoring key metrics for success.

*Figure 4: Agile Methodology (nvsia learn, 2020).*

For more information on Agile Methodology, visit [Appendix 12.3](#).

### 5.1.2 RUP Methodology

A Rational Unified Process is a software engineering approach for distributing tasks and responsibilities inside the software development organization. Its main goal is to make it possible to produce high-quality software that meets end users' needs on a predictable budget and schedule (Kanjilal, 2022).

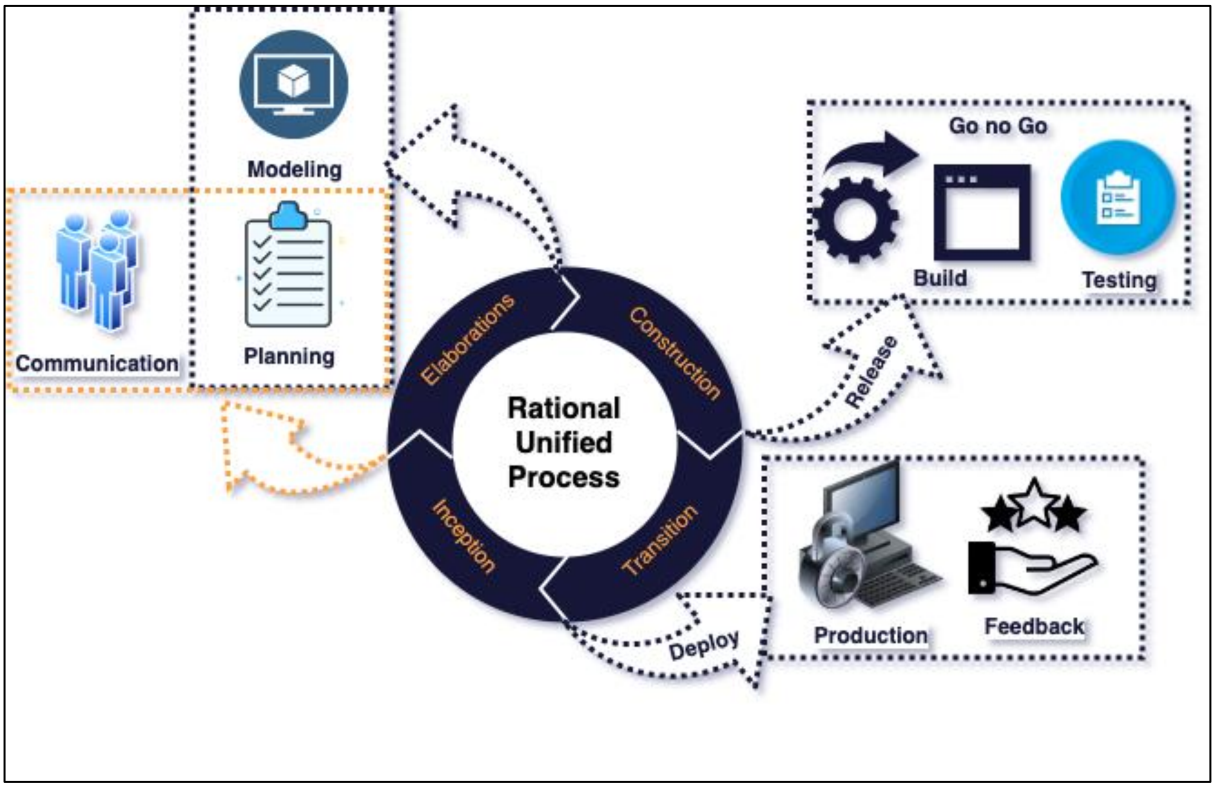


*Figure 5: RUP Methodology (Agile Lone Star, 2024).*

For more information on RUP Methodology, visit Appendix 12.1.

### 5.1.3 Prototype Model

The Prototyping Model is a software development process model that focuses on quickly building a functioning model or prototype of software in order to receive user feedback and improve requirements prior to developing the final product (Hossain, 2024).

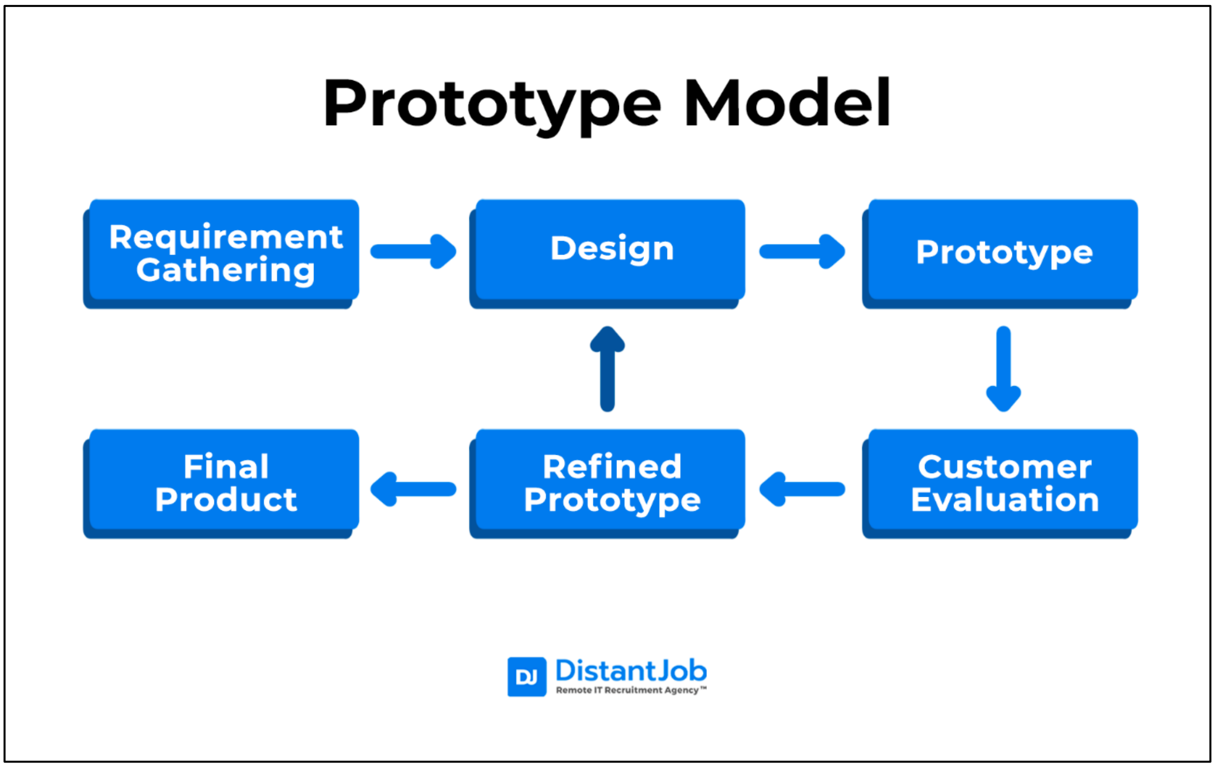


*Figure 6: Prototype Model.*

For more information on Prototype Model, visit Appendix 12.2.

## 5.2 Selected Methodology

Among the different methodologies that were taken into consideration, the Scrum Framework of Agile Methodology was chosen for the project.

### 5.2.1 Agile Methodology- Scrum Framework

The Scrum Framework was chosen given that it can provide an organized yet flexible approach to developing the packet analysis tool; Network Packet Visualizer Tool, facilitating effective advancement through its three pillars: Transparency, Inspection, and Adaptation.

The primary reasons for choosing Scrum Framework with respect to the three pillars are outlined below:

- **Transparency:**
  Scrum places a strong emphasis on transparency through its various artifacts, including the Increment, Product Backlog, and Sprint Backlog. Through transparency, it will be easier to keep track of how the project is progressing. (Schwaber & Sutherland, 2020).

- **Inspection:**
  Through its five events (Sprint Planning, Daily Scrum, Sprint Review, Sprint Retrospective, and the Sprint itself), Scrum offers frequent inspection opportunities. This enables us to monitor the progress of the project on a regular basis, identify problems early, and make the required corrections (Schwaber & Sutherland, 2020).

- **Adaptation:**
  The flexibility of Scrum enables us to quickly modify the procedure or final product if any aspect of the project deviates from the anticipated result. It guarantees that we can quickly handle any alterations and maintain the project's progress to provide a high-quality product (Schwaber & Sutherland, 2020).

For more information on the reasons behind choosing Scrum Framework, visit Appendix 12.3.2.

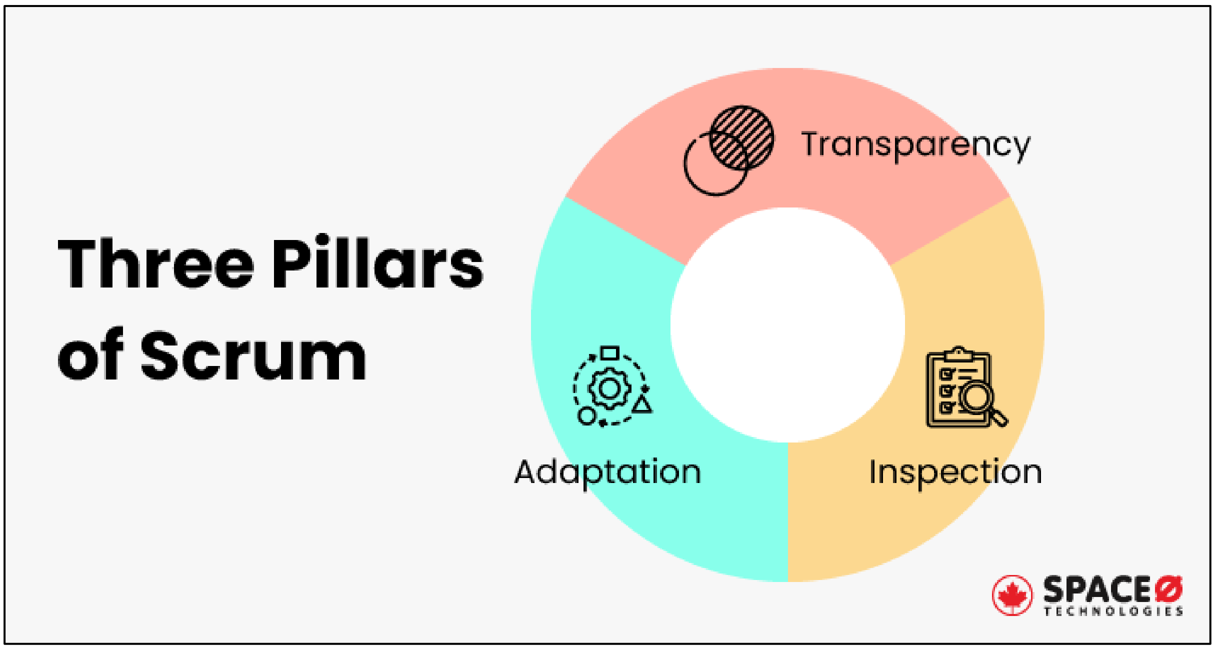


*Figure 7: Three Pillars of Scrum.*

For more information on the Scrum Framework, visit Appendix 12.3.1.

# 6. Resource Requirements

Below are the hardware and software components that will be utilized to launch the project and develop the project.

## 6.1 Hardware Requirements

The hardware requirements are mentioned below:

a) Development Machine:
Personal device will be used as a development machine.

## 6.2 Software Requirements

The software requirements are mentioned below:

| Category | Software Requirements | Specifications |
|---|---|---|
| Operating System | Any Unix | Zeek was designed for a Unix-based OS. Even though Zeek has released its windows version; the Windows version of Zeek is not fully reliable. |
| Development Tools | Visual Studio Code | Visual Studio Code will be used for frontend and backend development. |
| | Figma / Canva | Figma / Canva will be used for system design. |
| | React | React will be used as the main language for frontend development. |
| | Python | Python will be used as the main language for backend development. |
| | GitHub | GitHub will be used for version control. |
| Data Visualization | TBD | The data visualization tool is yet to be decided. |
| Backup Storage | Git / OneDrive | Git / One drive will be used as Backup Storage. |

| | | |
|---|---|---|
| Project Management Tool | Trello | Trello will be used to keep track of milestones. |
| Documentation | MS Word | MS Word will be used for all the documentation process. |
| Flowcharts / Diagrams | Draw.io / Canva | Draw.io / Canva will be used to make flowcharts or the required system diagrams. |
| Gantt Chart | Team Gantt | The Project Gantt Chart will be made using Team Gantt. |

*Table 1: Software Requirements*

## 7. Work Breakdown Structure

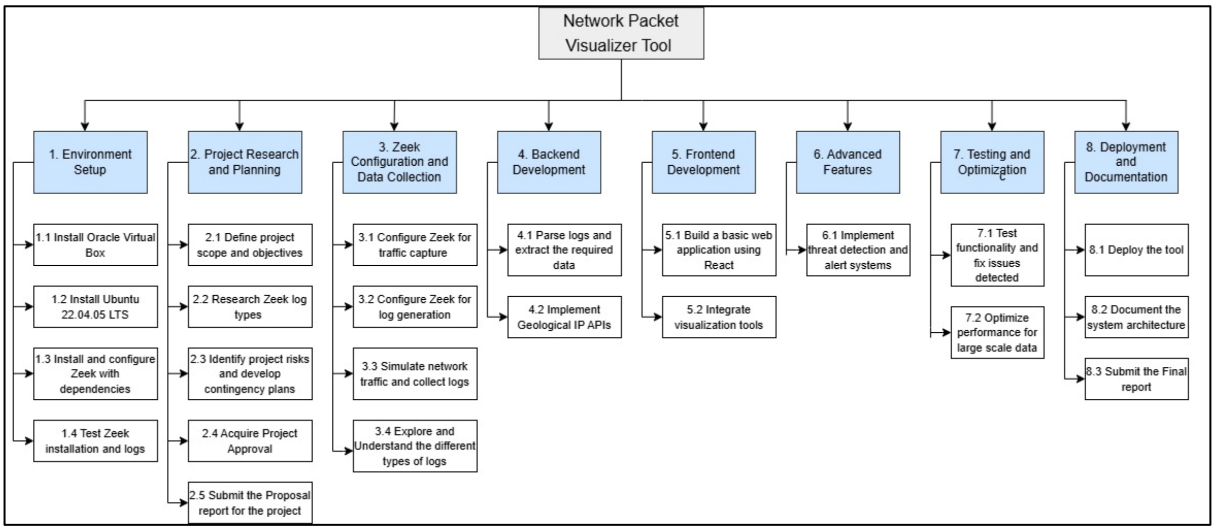

Network Packet Visualizer Tool
1. Environment Setup
1.1 Install Oracle Virtual Box
1.2 Install Ubuntu 22.04.05 LTS
1.3 Install and configure Zeek with dependencies
1.4 Test Zeek installation and logs
2. Project Research and Planning
2.1 Define project scope and objectives
2.2 Research Zeek log types
2.3 Identify project risks and develop contingency plans
2.4 Acquire Project Approval
2.5 Submit the Proposal report for the project
3. Zeek Configuration and Data Collection
3.1 Configure Zeek for traffic capture
3.2 Configure Zeek for log generation
3.3 Simulate network traffic and collect logs
3.4 Explore and Understand the different types of logs
4. Backend Development
4.1 Parse logs and extract the required data
4.2 Implement Geological IP APIs
5. Frontend Development
5.1 Build a basic web application using React
5.2 Integrate visualization tools
6. Advanced Features
6.1 Implement threat detection and alert systems
7. Testing and Optimization
7.1 Test functionality and fix issues detected
7.2 Optimize performance for large scale data
8. Deployment and Documentation
8.1 Deploy the tool
8.2 Document the system architecture
8.3 Submit the Final report

*Figure 8: Work Breakdown Structure.*

For the enlarged and clear view of WBS, visit Appendix 12.4.

## 8. Milestones

| S. No | Expected Date | Milestone | Description |
|---|---|---|---|
| 1 | October 30, 2024 | Topic Finalization | The topic of the project was finalized to be “Network Packet Visualizer Tool”. |
| 2 | December 4, 2024 | Proposal Submission | Submit a detailed project proposal, including aim and objectives, expected outcomes and risk and contingency planning. |
| 3 | January 4, 2025 | Log Parsing and Data Extraction | Parse log files to extract relevant data and present it in a rough format for initial review. |
| 4 | January 8, 2025 | Interim Report Submission | Submit an interim report covering log parsing, data extraction, and early feedback. |
| 5 | February 8, 2025 | Data Visualization Dashboard | Develop and display the extracted data in a dashboard with basic visualizations. |
| 6 | March 25, 2025 | Implementation of Advanced Features | Integrate advanced functions such as real-time monitoring, alerts, and threat detection. |

| | | | |
|---|---|---|---|
| 7 | April 4, 2025 | Testing Phase | Conduct comprehensive testing of all features, including performance. |
| 8 | April 30, 2025 | Final Report Submission | Submit the final report including project artefacts, and logs. |

*Table 2: Project Milestone Table.*

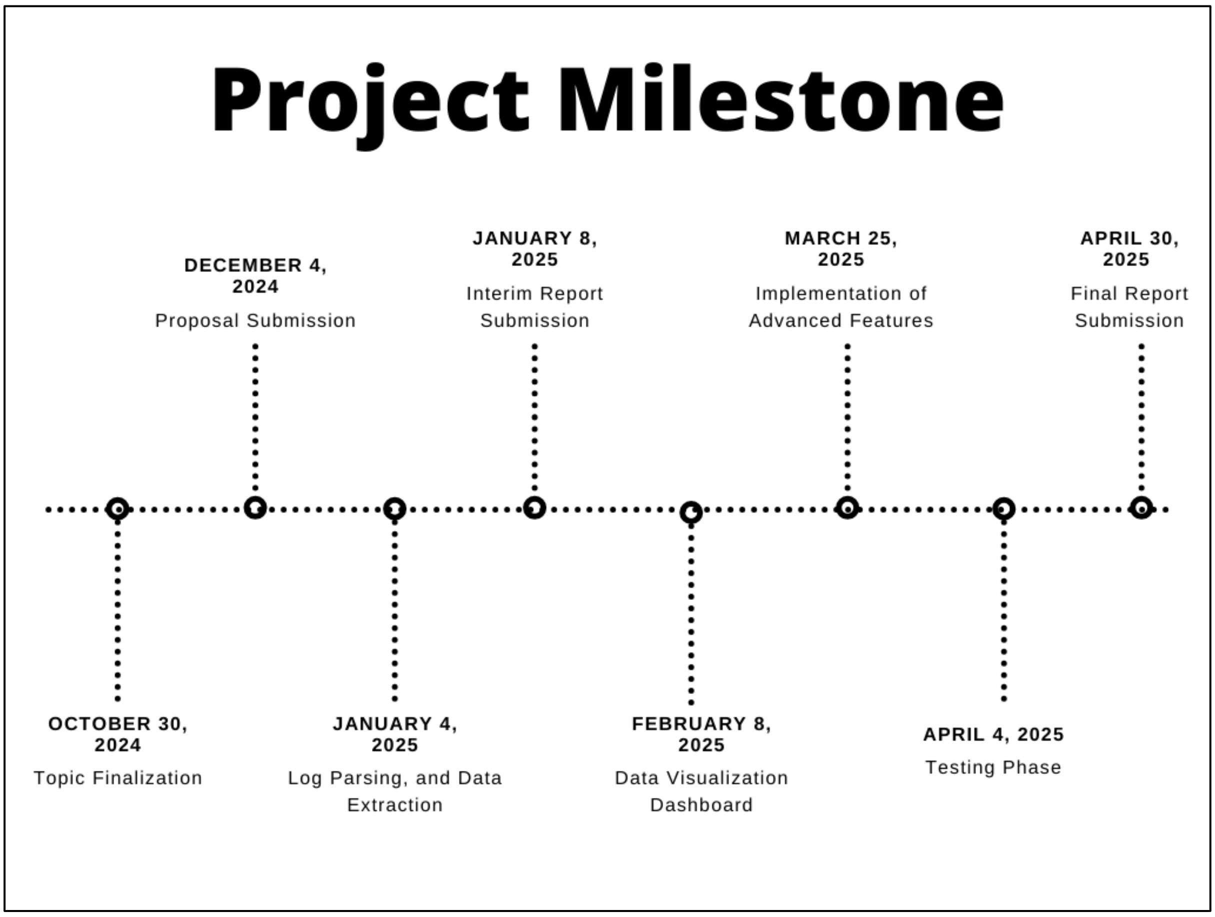
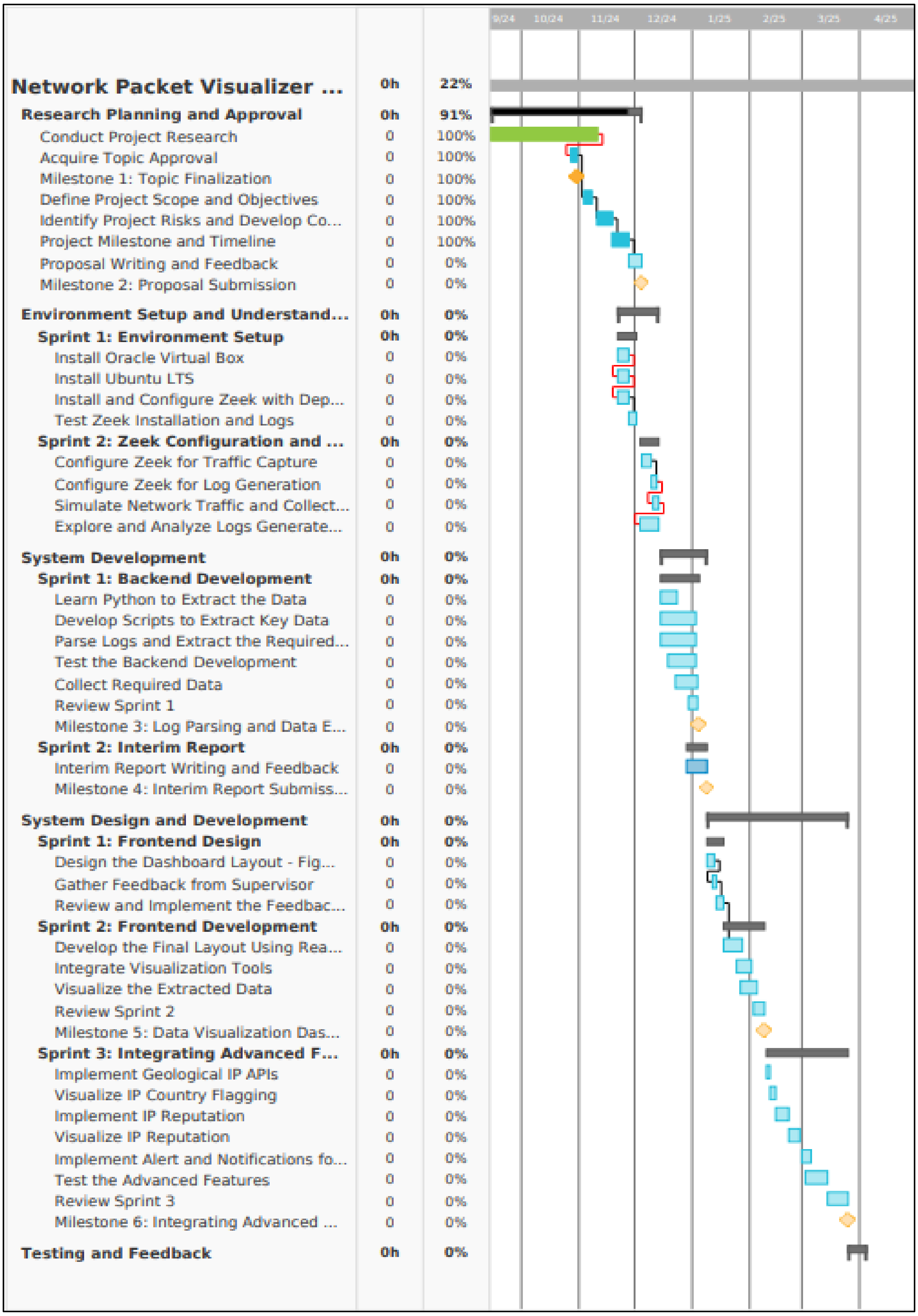
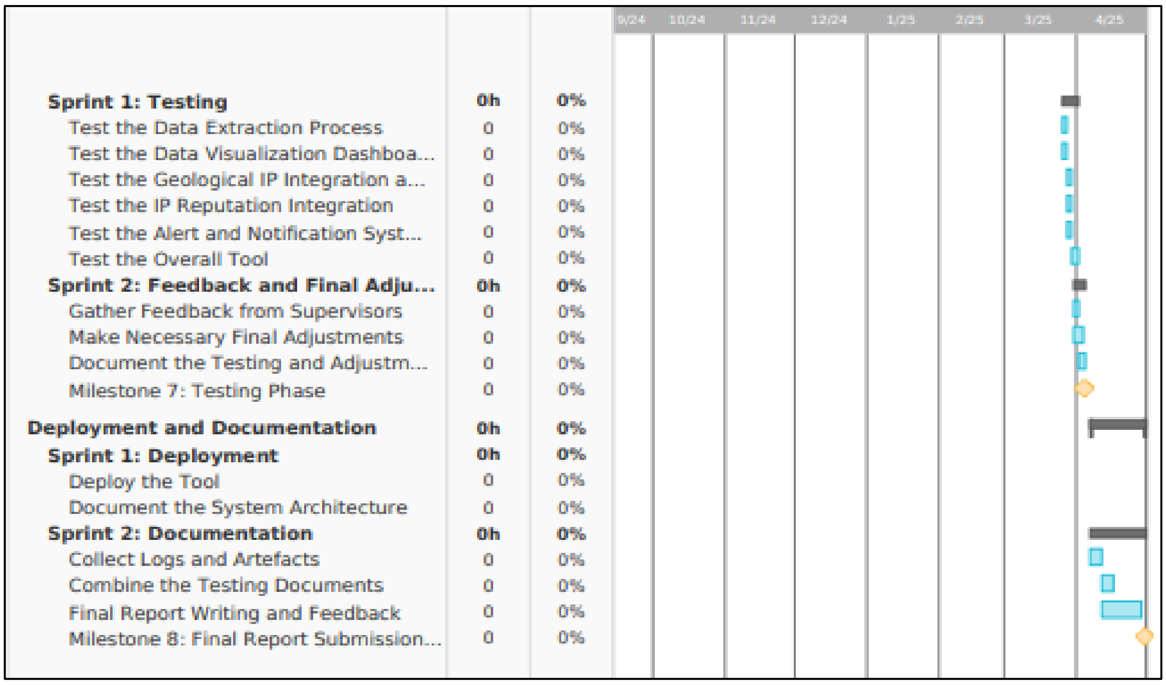


*Figure 9: Project Milestone Map.*

## 9. Project Gantt Chart



| | | |
|---|---|---|
| **Network Packet Visualizer ...** | **0h** | **22%** |
| **Research Planning and Approval** | **0h** | **91%** |
| Conduct Project Research | 0 | 100% |
| Acquire Topic Approval | 0 | 100% |
| Milestone 1: Topic Finalization | 0 | 100% |
| Define Project Scope and Objectives | 0 | 100% |
| Identify Project Risks and Develop Co... | 0 | 100% |
| Project Milestone and Timeline | 0 | 100% |
| Proposal Writing and Feedback | 0 | 0% |
| Milestone 2: Proposal Submission | 0 | 0% |
| **Environment Setup and Understand...** | **0h** | **0%** |
| **Sprint 1: Environment Setup** | **0h** | **0%** |
| Install Oracle Virtual Box | 0 | 0% |
| Install Ubuntu LTS | 0 | 0% |
| Install and Configure Zeek with Dep... | 0 | 0% |
| Test Zeek Installation and Logs | 0 | 0% |
| **Sprint 2: Zeek Configuration and ...** | **0h** | **0%** |
| Configure Zeek for Traffic Capture | 0 | 0% |
| Configure Zeek for Log Generation | 0 | 0% |
| Simulate Network Traffic and Collect... | 0 | 0% |
| Explore and Analyze Logs Generate... | 0 | 0% |
| **System Development** | **0h** | **0%** |
| **Sprint 1: Backend Development** | **0h** | **0%** |
| Learn Python to Extract the Data | 0 | 0% |
| Develop Scripts to Extract Key Data | 0 | 0% |
| Parse Logs and Extract the Required... | 0 | 0% |
| Test the Backend Development | 0 | 0% |
| Collect Required Data | 0 | 0% |
| Review Sprint 1 | 0 | 0% |
| Milestone 3: Log Parsing and Data E... | 0 | 0% |
| **Sprint 2: Interim Report** | **0h** | **0%** |
| Interim Report Writing and Feedback | 0 | 0% |
| Milestone 4: Interim Report Submiss... | 0 | 0% |
| **System Design and Development** | **0h** | **0%** |
| **Sprint 1: Frontend Design** | **0h** | **0%** |
| Design the Dashboard Layout - Fig... | 0 | 0% |
| Gather Feedback from Supervisor | 0 | 0% |
| Review and Implement the Feedbac... | 0 | 0% |
| **Sprint 2: Frontend Development** | **0h** | **0%** |
| Develop the Final Layout Using Rea... | 0 | 0% |
| Integrate Visualization Tools | 0 | 0% |
| Visualize the Extracted Data | 0 | 0% |
| Review Sprint 2 | 0 | 0% |
| Milestone 5: Data Visualization Das... | 0 | 0% |
| **Sprint 3: Integrating Advanced F...** | **0h** | **0%** |
| Implement Geological IP APIs | 0 | 0% |
| Visualize IP Country Flagging | 0 | 0% |
| Implement IP Reputation | 0 | 0% |
| Visualize IP Reputation | 0 | 0% |
| Implement Alert and Notifications fo... | 0 | 0% |
| Test the Advanced Features | 0 | 0% |
| Review Sprint 3 | 0 | 0% |
| Milestone 6: Integrating Advanced ... | 0 | 0% |
| **Testing and Feedback** | **0h** | **0%** |

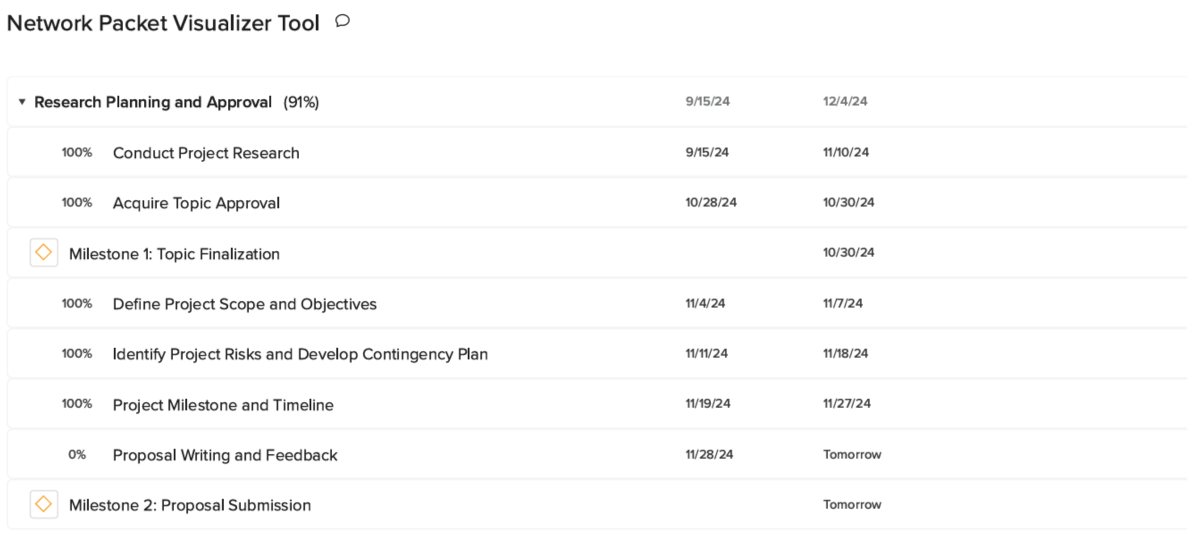


| | | |
|---|---|---|
| **Sprint 1: Testing** | **0h** | **0%** |
| Test the Data Extraction Process | 0 | 0% |
| Test the Data Visualization Dashboa... | 0 | 0% |
| Test the Geological IP Integration a... | 0 | 0% |
| Test the IP Reputation Integration | 0 | 0% |
| Test the Alert and Notification Syst... | 0 | 0% |
| Test the Overall Tool | 0 | 0% |
| **Sprint 2: Feedback and Final Adju...** | **0h** | **0%** |
| Gather Feedback from Supervisors | 0 | 0% |
| Make Necessary Final Adjustments | 0 | 0% |
| Document the Testing and Adjustm... | 0 | 0% |
| Milestone 7: Testing Phase | 0 | 0% |
| **Deployment and Documentation** | **0h** | **0%** |
| **Sprint 1: Deployment** | **0h** | **0%** |
| Deploy the Tool | 0 | 0% |
| Document the System Architecture | 0 | 0% |
| **Sprint 2: Documentation** | **0h** | **0%** |
| Collect Logs and Artefacts | 0 | 0% |
| Combine the Testing Documents | 0 | 0% |
| Final Report Writing and Feedback | 0 | 0% |
| Milestone 8: Final Report Submission... | 0 | 0% |

*Figure 10: Project Gantt Chart.*

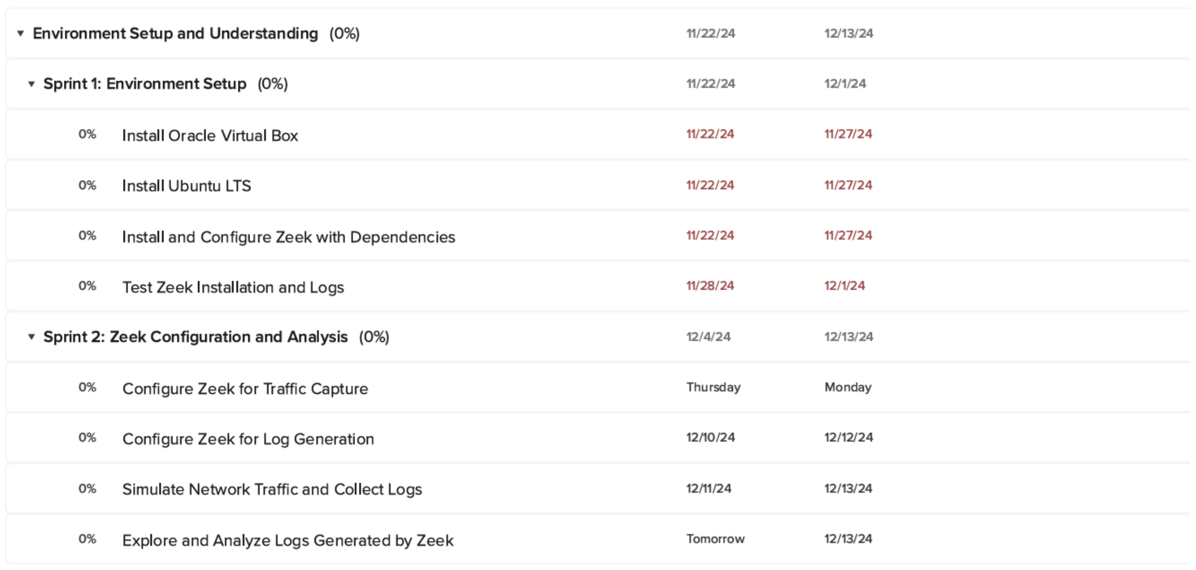
For the clear view of Project Gantt Chart, visit Appendix 12.5.

## 9.1 Project Gantt Chart – List View

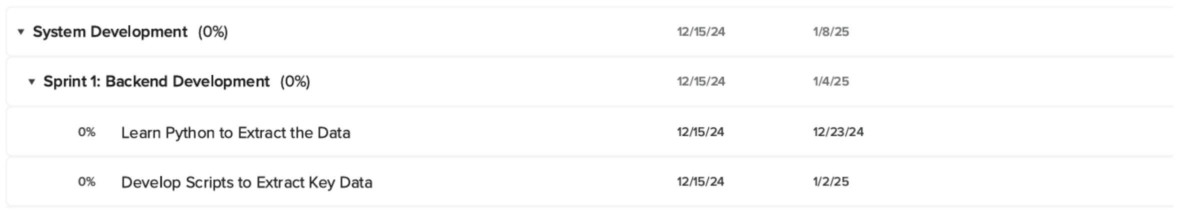


Network Packet Visualizer Tool

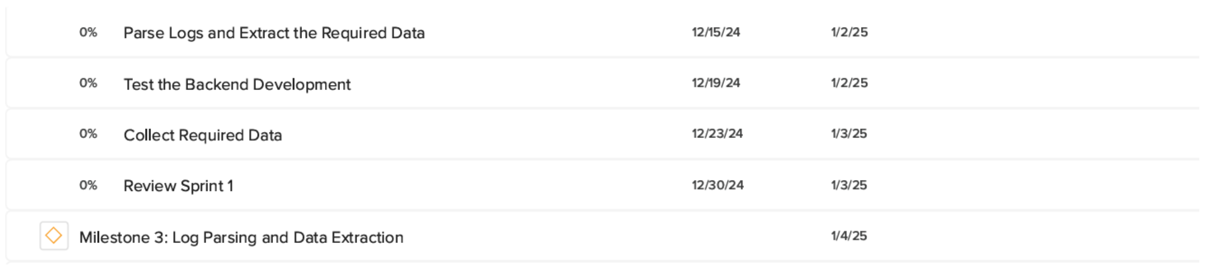
| | | | |
|---|---|---|---|
| ▾ | **Research Planning and Approval** (91%) | 9/15/24 | 12/4/24 |
| 100% | Conduct Project Research | 9/15/24 | 11/10/24 |
| 100% | Acquire Topic Approval | 10/28/24 | 10/30/24 |
| ◇ | Milestone 1: Topic Finalization | | 10/30/24 |
| 100% | Define Project Scope and Objectives | 11/4/24 | 11/7/24 |
| 100% | Identify Project Risks and Develop Contingency Plan | 11/11/24 | 11/18/24 |
| 100% | Project Milestone and Timeline | 11/19/24 | 11/27/24 |
| 0% | Proposal Writing and Feedback | 11/28/24 | Tomorrow |
| ◇ | Milestone 2: Proposal Submission | | Tomorrow |



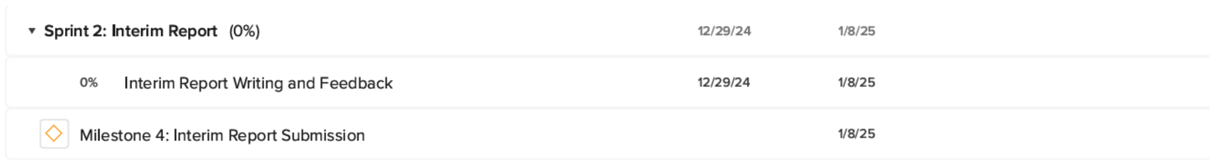
| | | | |
|---|---|---|---|
| ▾ | **Environment Setup and Understanding** (0%) | 11/22/24 | 12/13/24 |
| ▾ | **Sprint 1: Environment Setup** (0%) | 11/22/24 | 12/1/24 |
| 0% | Install Oracle Virtual Box | 11/22/24 | 11/27/24 |
| 0% | Install Ubuntu LTS | 11/22/24 | 11/27/24 |
| 0% | Install and Configure Zeek with Dependencies | 11/22/24 | 11/27/24 |
| 0% | Test Zeek Installation and Logs | 11/28/24 | 12/1/24 |
| ▾ | **Sprint 2: Zeek Configuration and Analysis** (0%) | 12/4/24 | 12/13/24 |
| 0% | Configure Zeek for Traffic Capture | Thursday | Monday |
| 0% | Configure Zeek for Log Generation | 12/10/24 | 12/12/24 |
| 0% | Simulate Network Traffic and Collect Logs | 12/11/24 | 12/13/24 |
| 0% | Explore and Analyze Logs Generated by Zeek | Tomorrow | 12/13/24 |



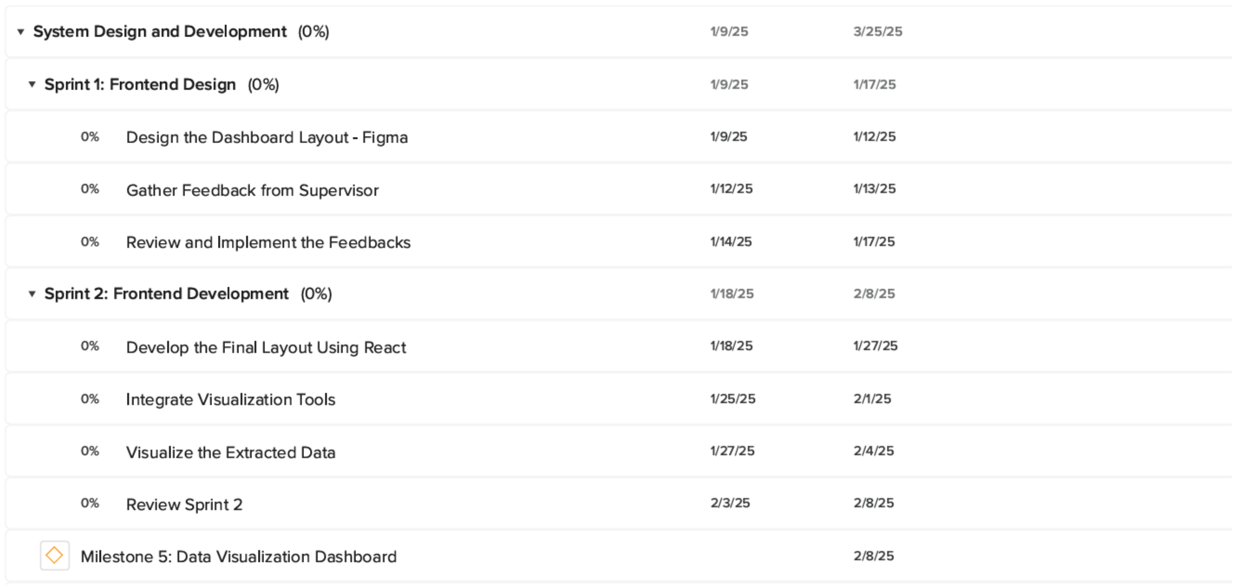
| | | | |
|---|---|---|---|
| ▾ | **System Development** (0%) | 12/15/24 | 1/8/25 |
| ▾ | **Sprint 1: Backend Development** (0%) | 12/15/24 | 1/4/25 |
| 0% | Learn Python to Extract the Data | 12/15/24 | 12/23/24 |
| 0% | Develop Scripts to Extract Key Data | 12/15/24 | 1/2/25 |



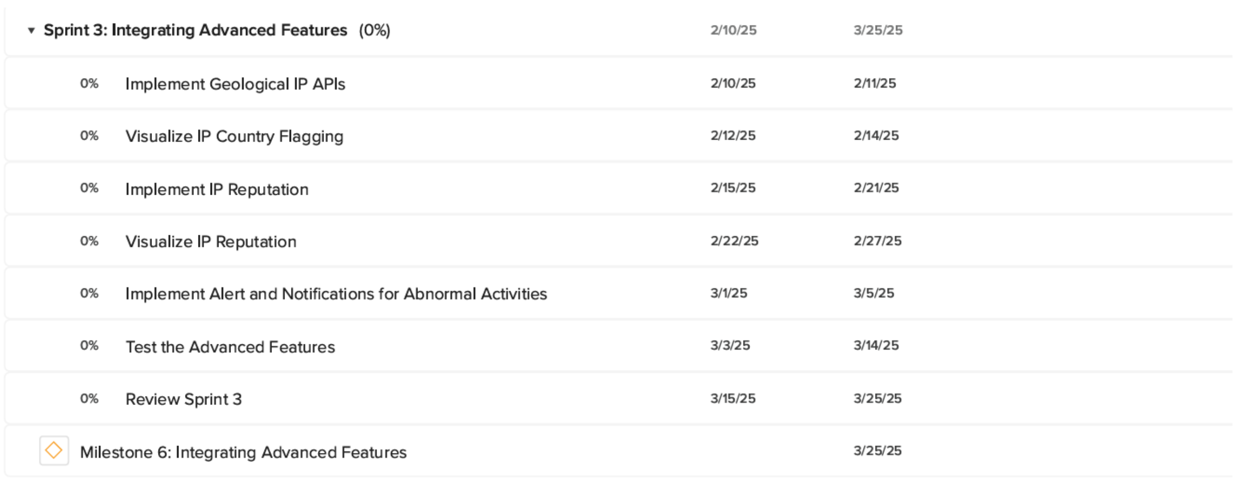
| | | | |
|---|---|---|---|
| 0% | Parse Logs and Extract the Required Data | 12/15/24 | 1/2/25 |
| 0% | Test the Backend Development | 12/19/24 | 1/2/25 |
| 0% | Collect Required Data | 12/23/24 | 1/3/25 |
| 0% | Review Sprint 1 | 12/30/24 | 1/3/25 |
| ◇ | Milestone 3: Log Parsing and Data Extraction | | 1/4/25 |



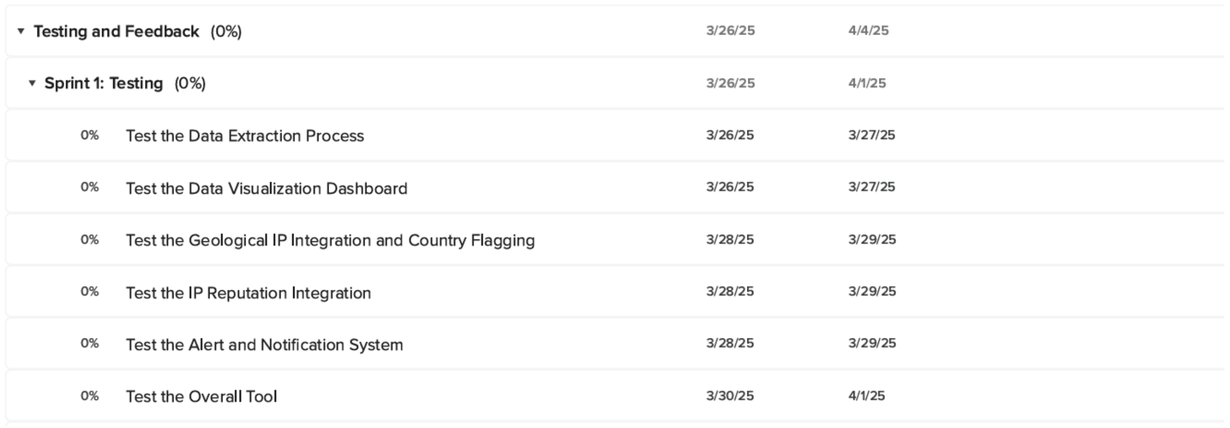
| | | | |
|---|---|---|---|
| ▾ **Sprint 2: Interim Report** (0%) | | 12/29/24 | 1/8/25 |
| 0% | Interim Report Writing and Feedback | 12/29/24 | 1/8/25 |
| ◇ | Milestone 4: Interim Report Submission | | 1/8/25 |



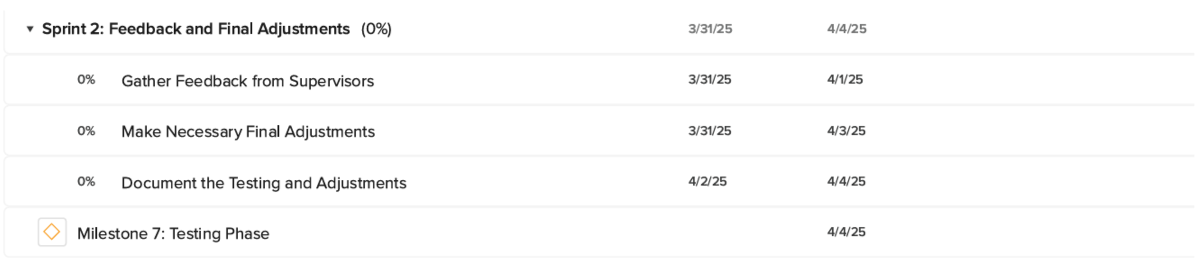
| | | | |
|---|---|---|---|
| ▾ **System Design and Development** (0%) | | 1/9/25 | 3/25/25 |
| ▾ **Sprint 1: Frontend Design** (0%) | | 1/9/25 | 1/17/25 |
| 0% | Design the Dashboard Layout - Figma | 1/9/25 | 1/12/25 |
| 0% | Gather Feedback from Supervisor | 1/12/25 | 1/13/25 |
| 0% | Review and Implement the Feedbacks | 1/14/25 | 1/17/25 |
| ▾ **Sprint 2: Frontend Development** (0%) | | 1/18/25 | 2/8/25 |
| 0% | Develop the Final Layout Using React | 1/18/25 | 1/27/25 |
| 0% | Integrate Visualization Tools | 1/25/25 | 2/1/25 |
| 0% | Visualize the Extracted Data | 1/27/25 | 2/4/25 |
| 0% | Review Sprint 2 | 2/3/25 | 2/8/25 |
| ◇ | Milestone 5: Data Visualization Dashboard | | 2/8/25 |



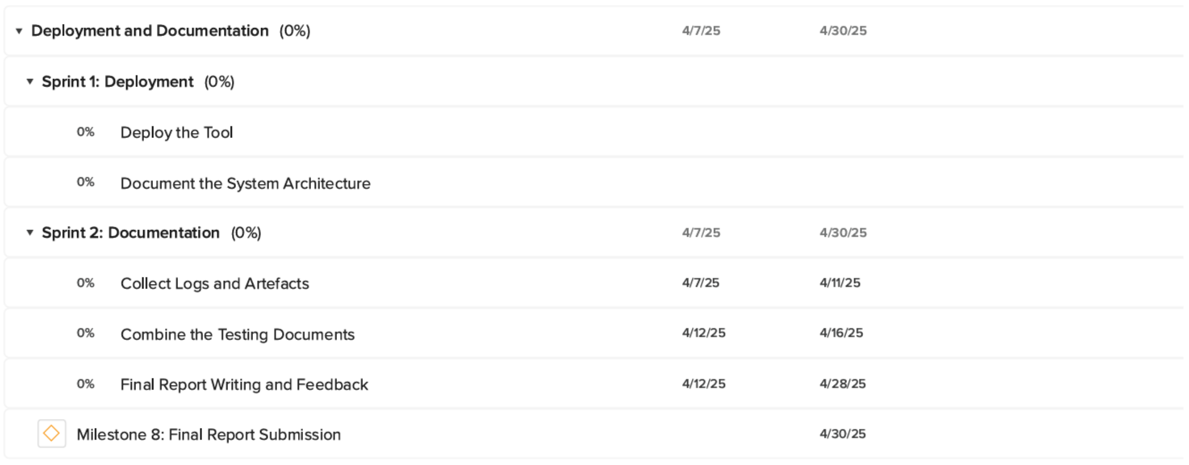
| | | | |
|---|---|---|---|
| ▾ **Sprint 3: Integrating Advanced Features** (0%) | | 2/10/25 | 3/25/25 |
| 0% | Implement Geological IP APIs | 2/10/25 | 2/11/25 |
| 0% | Visualize IP Country Flagging | 2/12/25 | 2/14/25 |
| 0% | Implement IP Reputation | 2/15/25 | 2/21/25 |
| 0% | Visualize IP Reputation | 2/22/25 | 2/27/25 |
| 0% | Implement Alert and Notifications for Abnormal Activities | 3/1/25 | 3/5/25 |
| 0% | Test the Advanced Features | 3/3/25 | 3/14/25 |
| 0% | Review Sprint 3 | 3/15/25 | 3/25/25 |
| ◇ | Milestone 6: Integrating Advanced Features | | 3/25/25 |



| | | | |
|---|---|---|---|
| ▾ **Testing and Feedback** (0%) | | 3/26/25 | 4/4/25 |
| ▾ **Sprint 1: Testing** (0%) | | 3/26/25 | 4/1/25 |
| 0% | Test the Data Extraction Process | 3/26/25 | 3/27/25 |
| 0% | Test the Data Visualization Dashboard | 3/26/25 | 3/27/25 |
| 0% | Test the Geological IP Integration and Country Flagging | 3/28/25 | 3/29/25 |
| 0% | Test the IP Reputation Integration | 3/28/25 | 3/29/25 |
| 0% | Test the Alert and Notification System | 3/28/25 | 3/29/25 |
| 0% | Test the Overall Tool | 3/30/25 | 4/1/25 |



| | | | |
|---|---|---|---|
| ▾ **Sprint 2: Feedback and Final Adjustments** (0%) | | 3/31/25 | 4/4/25 |
| 0% | Gather Feedback from Supervisors | 3/31/25 | 4/1/25 |
| 0% | Make Necessary Final Adjustments | 3/31/25 | 4/3/25 |
| 0% | Document the Testing and Adjustments | 4/2/25 | 4/4/25 |
| ◇ | Milestone 7: Testing Phase | | 4/4/25 |



| | | | |
|---|---|---|---|
| ▾ **Deployment and Documentation** (0%) | | 4/7/25 | 4/30/25 |
| ▾ **Sprint 1: Deployment** (0%) | | | |
| 0% | Deploy the Tool | | |
| 0% | Document the System Architecture | | |
| ▾ **Sprint 2: Documentation** (0%) | | 4/7/25 | 4/30/25 |
| 0% | Collect Logs and Artefacts | 4/7/25 | 4/11/25 |
| 0% | Combine the Testing Documents | 4/12/25 | 4/16/25 |
| 0% | Final Report Writing and Feedback | 4/12/25 | 4/28/25 |
| ◇ | Milestone 8: Final Report Submission | | 4/30/25 |

## 10. Conclusion

The main objective of this project is to develop an all-inclusive tool; Network Packet Visualizer Tool that is capable of packet analysis and visualization. This will be accomplished by using scrum framework of agile methodology that tackles development and delivery in iterations, allowing for a high degree of flexibility for user needs to be prioritized.

There are some advanced network analysis tools such as Wireshark and Zeek, which are used by network professionals; but that leaves room for general users to not be able to see if their network is up, troubleshoot connectivity problems, or even identify if they may have a security threat on their network.

The expected outcome is a web application meant for making packet analysis and network monitoring easier both for the general users as well as network professionals. It provides an opportunity to fill the gap in the existing network analysis tool and at the same time learn about network and network security, and software development among others.

## 11. References

Agile Lone Star, 2024. *agilelonestar.* [Online]
Available at: https://www.agilelonestar.com/knowledge-base/rational-unified-process
[Accessed 17 September 2024].

ALazzawi, A., M.Yas, Q. & Rahmatullah, B., 2023. A Comprehensive Review of Software Development Life Cycle methodologies: Pros, Cons, and Future Directions. *Iraqi Journal for Computer Science and Mathematics,* 4(4), pp. 173-190.

CompTIA, 2024. *CompTIA.* [Online]
Available at: https://www.comptia.org/content/articles/what-is-wireshark-and-how-to-use-it
[Accessed 15 September 2024].

Hammad, M., 2024. *GeeksforGeeks.* [Online]
Available at: https://www.geeksforgeeks.org/rup-and-its-phases/#what-is-rational-unified-process-rup
[Accessed 18 September 2024].

H, L. et al., 2024. A Review of Modern Computer Networking Technologies. *IJARIIE,* 10(4), pp. 1844-1851.

Hossain, M. I., 2024. Software Development Life Cycle (SDLC) Methodologies for Information Systems Project Management. *International Journal for Multidisciplinary Research (IJFMR),* 5(5), pp. 1-36.

Kanjilal, J., 2022. *developer.com.* [Online]
Available at: https://www.developer.com/project-management/rational-unified-process-rup/
[Accessed 16 September 2024].

Khan, D. R. A. H. et al., 2024. The Waterfall Model-Software Engineering. *International Journal of Research Publication and Reviews,* 5(4), pp. 7825-7830.

Khan, S. M., 2023. *Waterfall Model Used in Software Development Reference: Software Requirements Engineering Waterfall Model.* s.l.:s.n.

Mugwadi, I., 2024. *Medium.* [Online]
Available at: https://medium.com/@mugwadiinnocent/system-development-life-cycle-sdlc-afce857b91fc
[Accessed 18 September 2024].

nvsia learn, 2020. *nvisia.* [Online]
Available at: https://www.nvisia.com/insights/agile-methodology
[Accessed 17 September 2024].

Sandamal, H., 2020. *Medium.* [Online]
Available at: https://hansasandamal.medium.com/software-development-life-cycle-methods-7ccd25d6e198
[Accessed 17 September 2024].

Schwaber, K. & Sutherland, J., 2020. *The Scrum Guide.* s.l.:2020 Ken Schwaber and Jeff Sutherland.

Zhang, S., 2024. The Importance of Computer Network Applications and Security. *Journal of Computing and Electronic Information Management,* 12(3), pp. 55-58.

# 12. Appendix

## 12.1 RUP Methodology

Rational Unified Process (RUP) is a software development methodology for object-oriented models. It goes by the name Unified Process Model as well. UML (Unified Modeling Language) is used in its design and documentation, and it was produced by Rational Corporation. The IBM Rational Method Composer (RMC) software includes this procedure. Customization, design, and personalization of the unified process are made possible by IBM (International Business Machine Corporation) (Hammad, 2024).

Since its creation by Rational Software in the 1990s, the RUP has grown to be one of the most popular approaches for software development (Kanjilal, 2022).

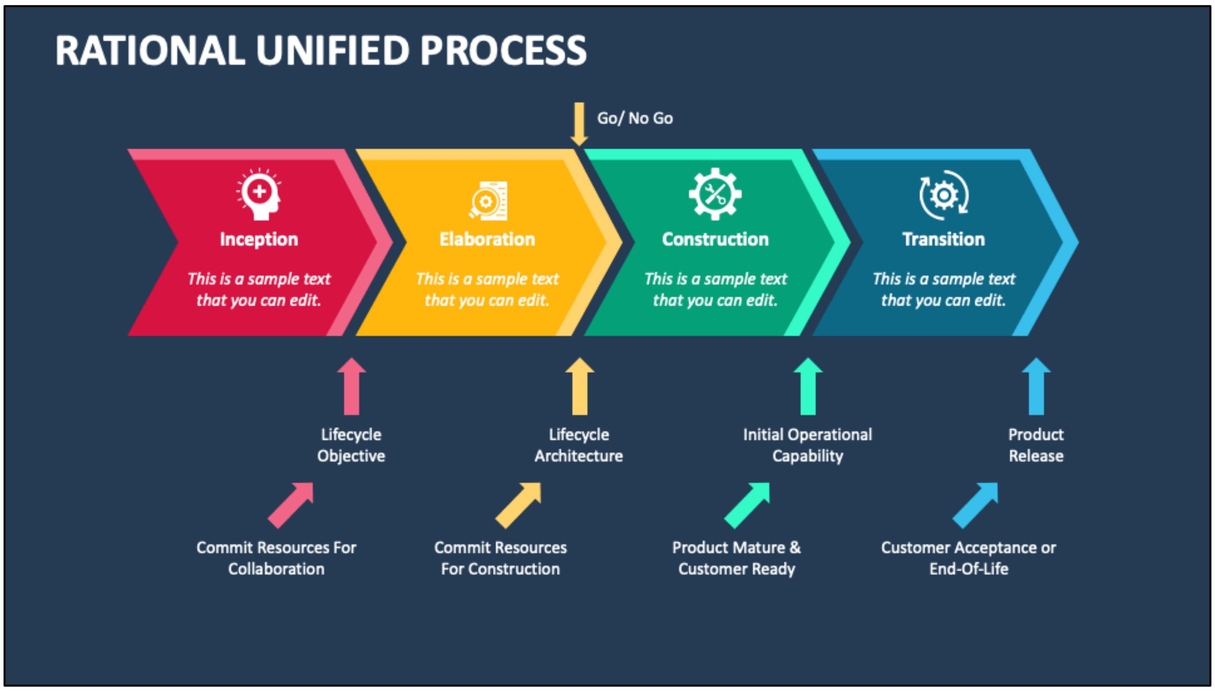

RATIONAL UNIFIED PROCESS
Go/ No Go
Inception
This is a sample text that you can edit.
Elaboration
This is a sample text that you can edit.
Construction
This is a sample text that you can edit.
Transition
This is a sample text that you can edit.
Lifecycle Objective
Lifecycle Architecture
Initial Operational Capability
Product Release
Commit Resources For Collaboration
Commit Resources For Construction
Product Mature & Customer Ready
Customer Acceptance or End-Of-Life

*Figure 11: Phases of RUP Methodology.*

### 12.1.1 Circumstances to use RUP Methodology

The RUP methodology works best for building high-quality software within a time and budget limit (Hammad, 2024).

### 12.1.2 Advantages of RUP Methodology

The benefits of the Rational Unified Process (RUP) are as follows:

- Good documentation is provided by RUP, which also completes the procedure.
- RUP offers assistance with risk management.
- Because RUP reuses the components, the overall time is reduced.
- Tutorials and training are accessible online to provide effective assistance.

### 12.1.3 Disadvantages of RUP Methodology

The following are RUP's (Rational Unified Process) drawbacks:

- Because the process is complex, a team of qualified professionals is needed.
- A complex and poorly structured procedure.
- An increased reliance on risk mitigation.
- Challenging to continuously incorporate.

## 12.2 Prototype Model

Prototype Model is a software development process that mainly focuses on creating a working model of the software quickly and then gathers user feedback to develop the final product. The key goal is to guarantee that the software closely matches user needs and expectations. It entails iteratively improving the prototype in response to user feedback until the intended functionality and design are realized. The Prototyping Model is especially useful when needs are ambiguous or subject to change, as well as when early user interaction and validation is required (Hossain, 2024).

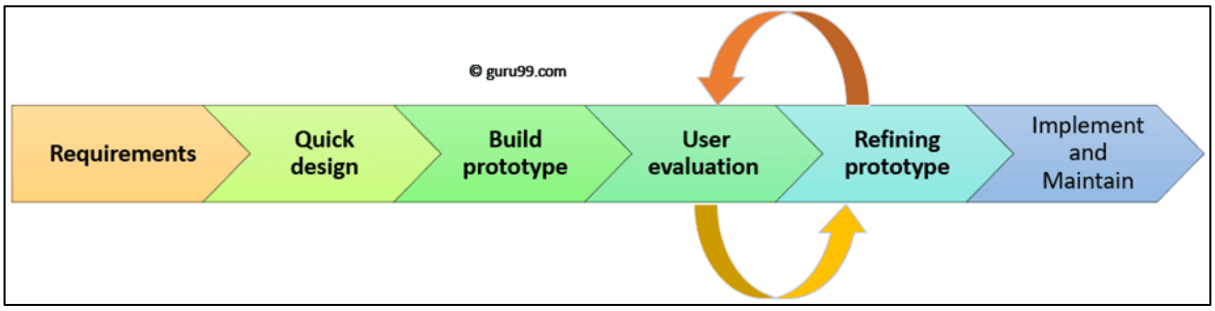


*Figure 12: Steps of Prototype Model.*

### 12.2.1 Circumstances to use Prototype Model

The Prototype Model works best when early user participation and validation are required, and when requirements are susceptible to change. It is a useful strategy for enhancing communication between development teams and stakeholders, which eventually results in the creation of a more effective and approachable end product (Hossain, 2024).

## 12.3 Agile Software Development Frameworks

Agile places a strong emphasis on close coordination between customers, testers, and developers at every stage of the project. This methodology allows teams to adapt well to changing priorities and is especially well-suited for projects whose requirements are dynamic or change often. The Agile development methodology provides a conceptual framework for managing a variety of software development projects. The Agile development methodology runs in sprint. Usually lasting one to four weeks, these sprints produce a software increment that may be prepared for eventual deployment (Hossain, 2024).

There are a few different software development methodologies that fall under the overall framework of Agile; the most well-known ones are Extreme Programming (XP), Crystal methods, Scrum, and Feature Driven Development (Hossain, 2024).

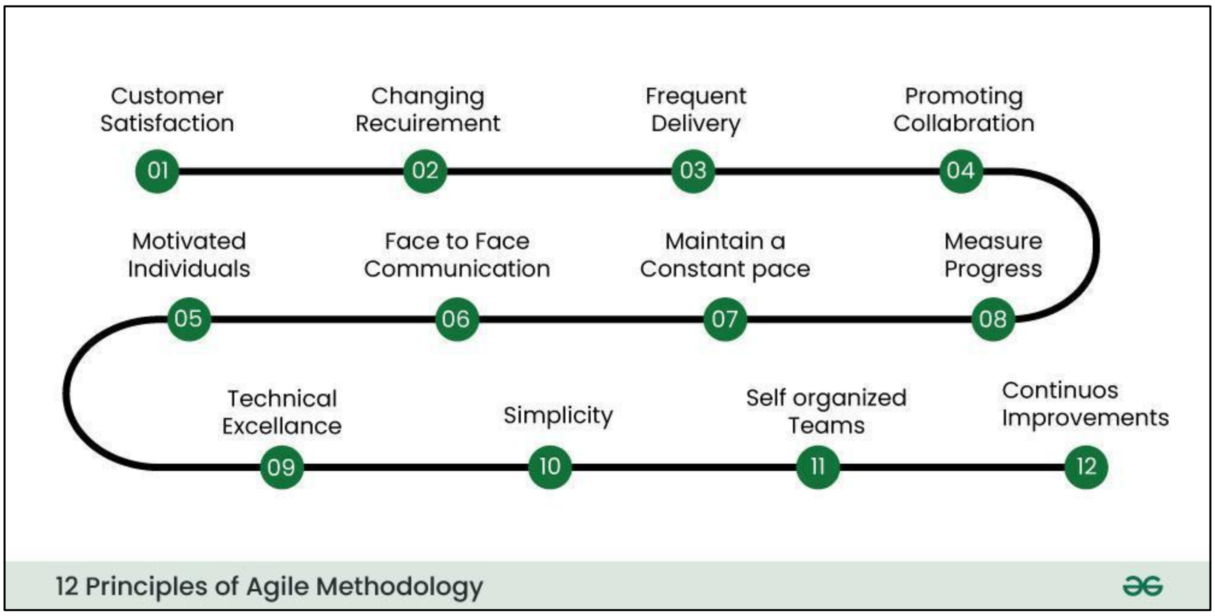


*Figure 13: Principles of Agile Methodology.*

### 12.3.1 Scrum Framework

Within the Agile technique, the Scrum Framework adds further structure in the form of roles, ceremonies, and artifacts. Scrum uses a framework that consists of sprint planning, daily stand-up meetings, sprint reviews, retrospectives, and product backlogs. Scrum's primary characteristic is its focus on brief, time-boxed iterations known as sprints, which usually run for two to four weeks. Scrum encourages transparency, review, and adaptability through every phase of the project (Hossain, 2024).

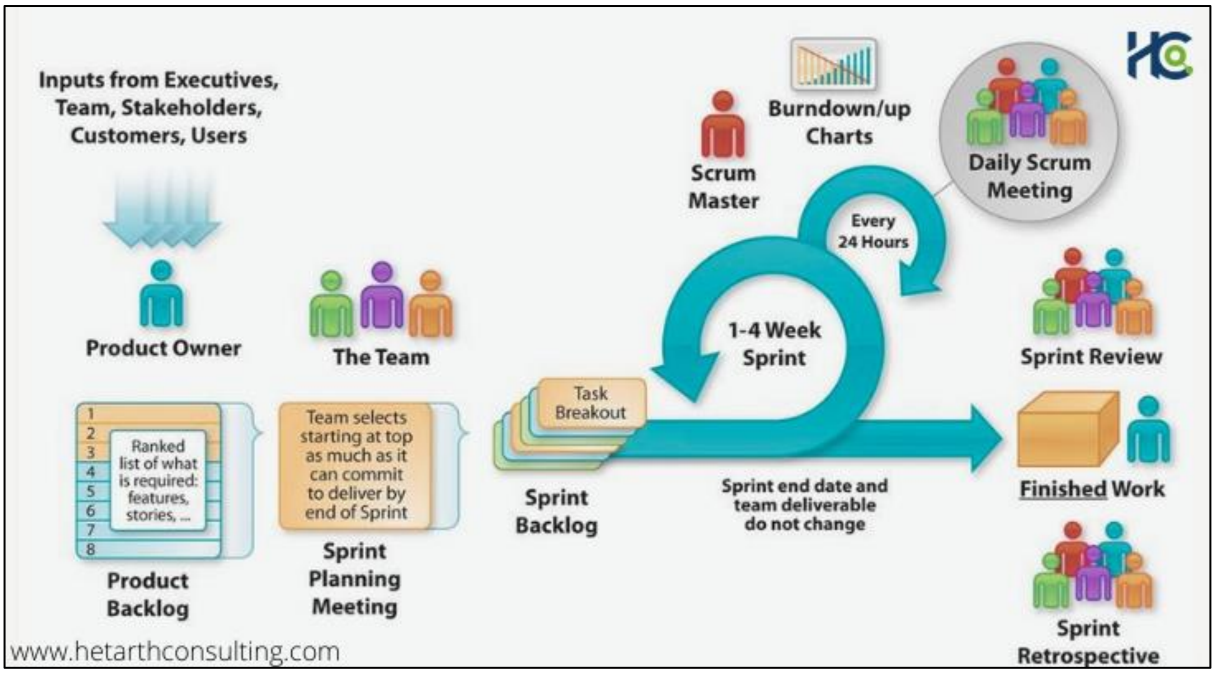

Inputs from Executives, Team, Stakeholders, Customers, Users
Product Owner
The Team
Scrum Master
Burndown/up Charts
Daily Scrum Meeting
Every 24 Hours
1-4 Week Sprint
Sprint Review
1
2
3
4
5
6
7
8
Ranked list of what is required: features, stories, ...
Product Backlog
Team selects starting at top as much as it can commit to deliver by end of Sprint
Sprint Planning Meeting
Task Breakout
Sprint Backlog
Sprint end date and team deliverable do not change
Finished Work
Sprint Retrospective
www.hetarthconsulting.com

*Figure 14: Scrum Framework*

### 12.3.2 Reasons for Selecting Scrum Framework

**Transparency:**

Scrum places a strong emphasis on transparency through its various artifacts, including the Increment, Product Backlog, and Sprint Backlog. Through transparency, it will be easier to keep track of how the project is progressing. This transparency facilitates decision-making, lowers risks, and guarantees that all parties involved are aware of and comprehend each phase of the project (Schwaber & Sutherland, 2020).

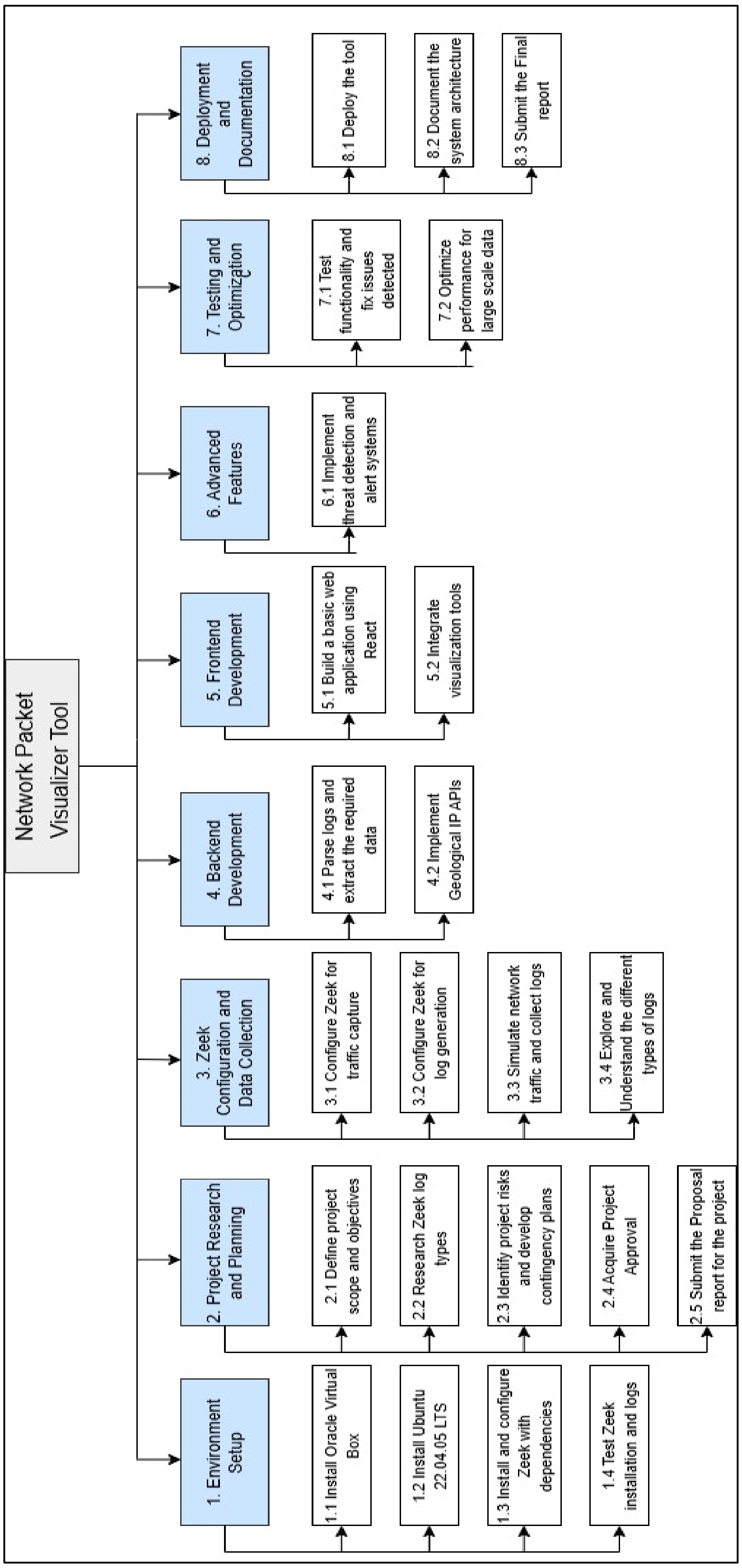
**Inspection:**

Through its five events (Sprint Planning, Daily Scrum, Sprint Review, Sprint Retrospective, and the Sprint itself), Scrum offers frequent inspection opportunities. This enables us to monitor the progress of the project on a regular basis, identify problems early, and make the required corrections (Schwaber & Sutherland, 2020). These inspections will aid in the early detection of possible issues with different features.

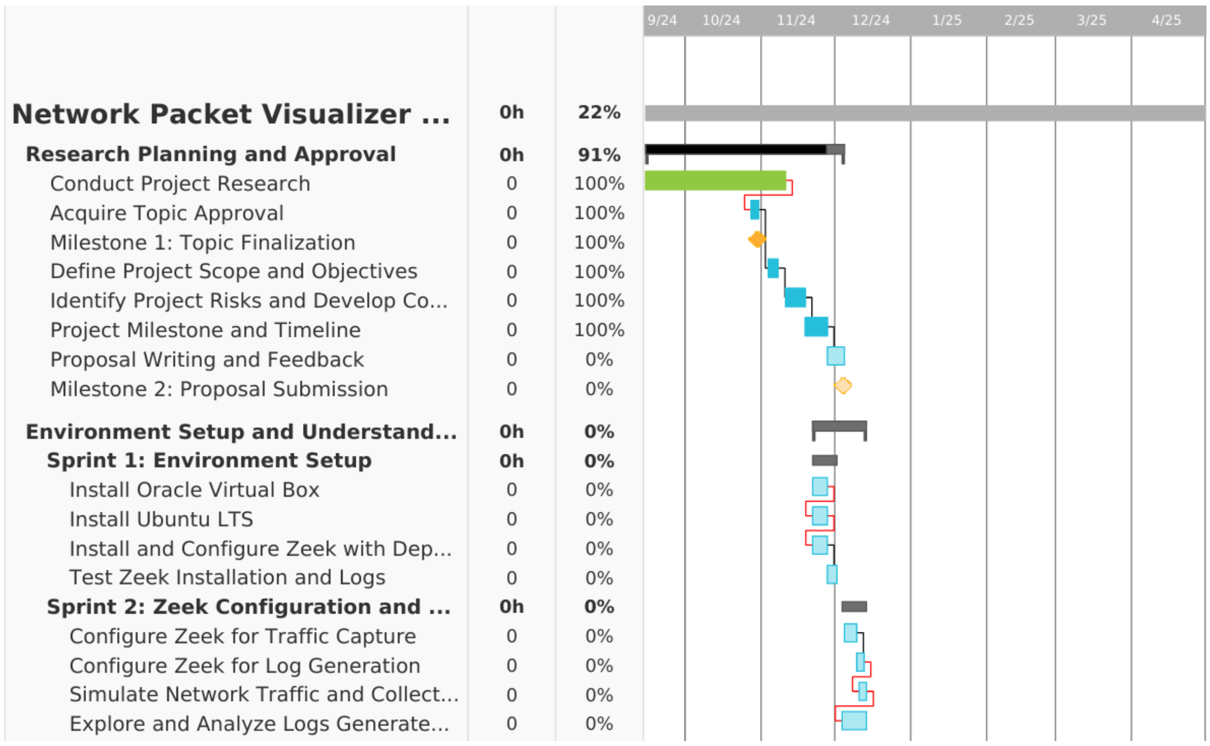
**Adaptation:**

The flexibility of Scrum enables us to quickly modify the procedure or final product if any aspect of the project deviates from the anticipated result. It guarantees that we can quickly handle any alterations and maintain the project's progress to provide a high-quality product (Schwaber & Sutherland, 2020). This adaptability is essential when creating sophisticated systems, such as network analysis tools, whose specifications may alter in response to user input or technical limitations.

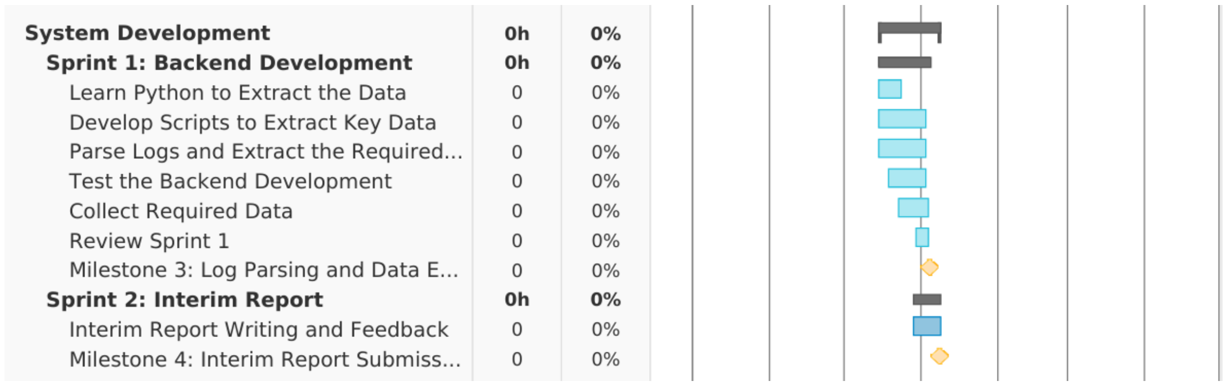
## 12.4 Work Breakdown Structure


Network Packet Visualizer Tool
1. Environment Setup
1.1 Install Oracle Virtual Box
1.2 Install Ubuntu 22.04.05 LTS
1.3 Install and configure Zeek with dependencies
1.4 Test Zeek installation and logs
2. Project Research and Planning
2.1 Define project scope and objectives
2.2 Research Zeek log types
2.3 Identify project risks and develop contingency plans
2.4 Acquire Project Approval
2.5 Submit the Proposal report for the project
3. Zeek Configuration and Data Collection
3.1 Configure Zeek for traffic capture
3.2 Configure Zeek for log generation
3.3 Simulate network traffic and collect logs
3.4 Explore and Understand the different types of logs
4. Backend Development
4.1 Parse logs and extract the required data
4.2 Implement Geological IP APIs
5. Frontend Development
5.1 Build a basic web application using React
5.2 Integrate visualization tools
6. Advanced Features
6.1 Implement threat detection and alert systems
7. Testing and Optimization
7.1 Test functionality and fix issues detected
7.2 Optimize performance for large scale data
8. Deployment and Documentation
8.1 Deploy the tool
8.2 Document the system architecture
8.3 Submit the Final report

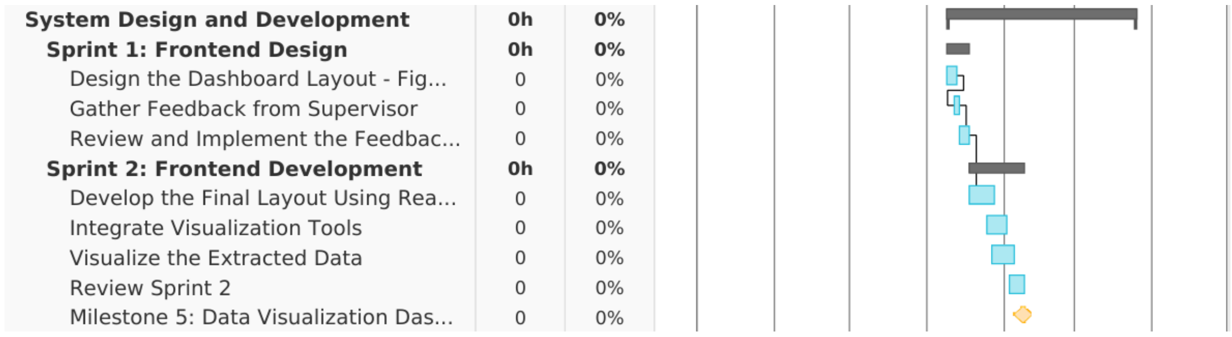
## 12.5 Project Gantt Chart



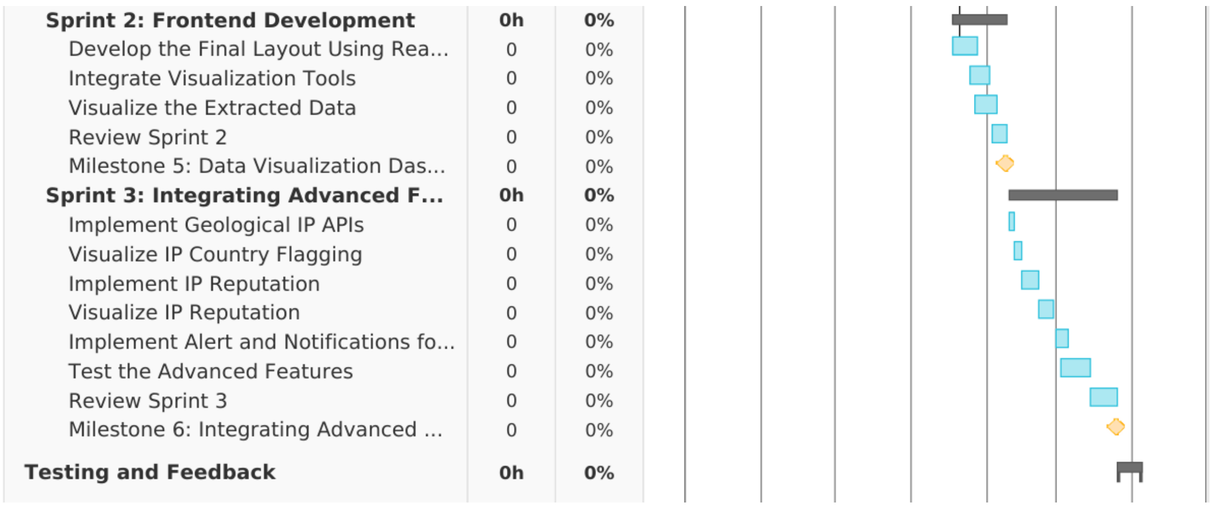
| | | |
|---|---|---|
| **Network Packet Visualizer ...** | **0h** | **22%** |
| **Research Planning and Approval** | **0h** | **91%** |
| Conduct Project Research | 0 | 100% |
| Acquire Topic Approval | 0 | 100% |
| Milestone 1: Topic Finalization | 0 | 100% |
| Define Project Scope and Objectives | 0 | 100% |
| Identify Project Risks and Develop Co... | 0 | 100% |
| Project Milestone and Timeline | 0 | 100% |
| Proposal Writing and Feedback | 0 | 0% |
| Milestone 2: Proposal Submission | 0 | 0% |
| **Environment Setup and Understand...** | **0h** | **0%** |
| **Sprint 1: Environment Setup** | **0h** | **0%** |
| Install Oracle Virtual Box | 0 | 0% |
| Install Ubuntu LTS | 0 | 0% |
| Install and Configure Zeek with Dep... | 0 | 0% |
| Test Zeek Installation and Logs | 0 | 0% |
| **Sprint 2: Zeek Configuration and ...** | **0h** | **0%** |
| Configure Zeek for Traffic Capture | 0 | 0% |
| Configure Zeek for Log Generation | 0 | 0% |
| Simulate Network Traffic and Collect... | 0 | 0% |
| Explore and Analyze Logs Generate... | 0 | 0% |



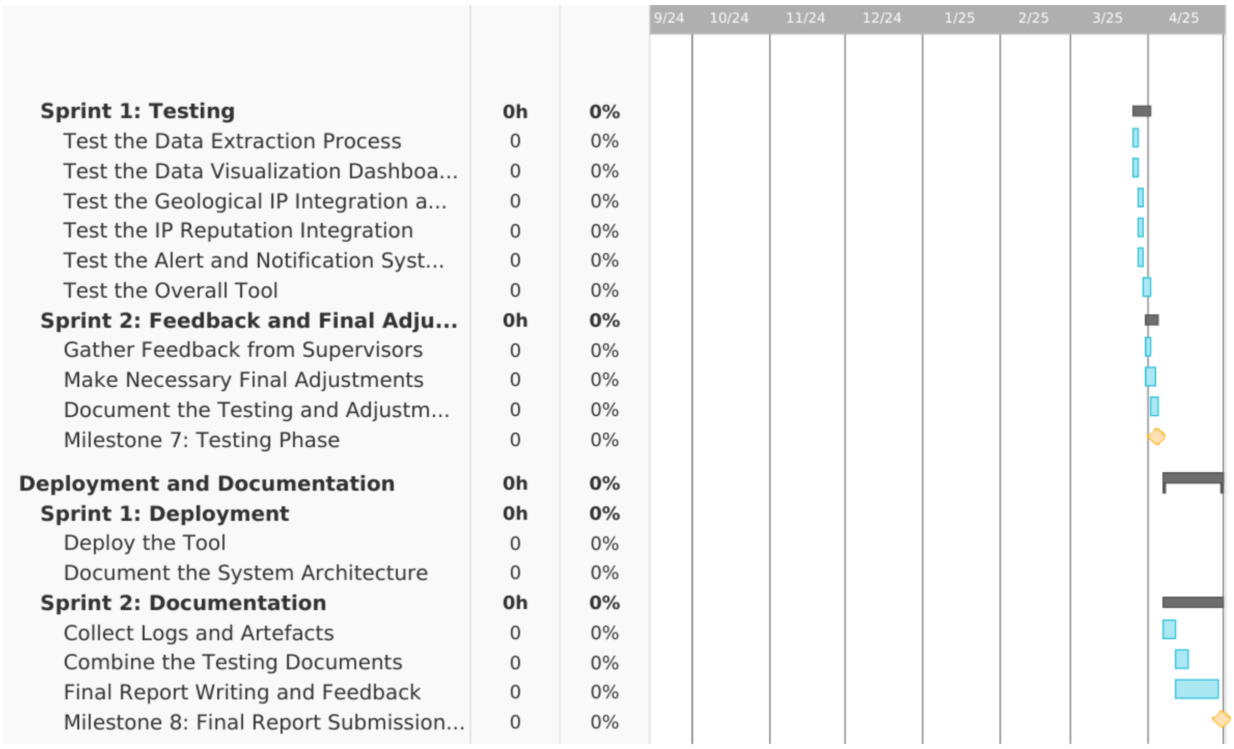
| | | |
|---|---|---|
| **System Development** | **0h** | **0%** |
| **Sprint 1: Backend Development** | **0h** | **0%** |
| Learn Python to Extract the Data | 0 | 0% |
| Develop Scripts to Extract Key Data | 0 | 0% |
| Parse Logs and Extract the Required... | 0 | 0% |
| Test the Backend Development | 0 | 0% |
| Collect Required Data | 0 | 0% |
| Review Sprint 1 | 0 | 0% |
| Milestone 3: Log Parsing and Data E... | 0 | 0% |
| **Sprint 2: Interim Report** | **0h** | **0%** |
| Interim Report Writing and Feedback | 0 | 0% |
| Milestone 4: Interim Report Submiss... | 0 | 0% |



| | | |
|---|---|---|
| **System Design and Development** | **0h** | **0%** |
| **Sprint 1: Frontend Design** | **0h** | **0%** |
| Design the Dashboard Layout - Fig... | 0 | 0% |
| Gather Feedback from Supervisor | 0 | 0% |
| Review and Implement the Feedbac... | 0 | 0% |
| **Sprint 2: Frontend Development** | **0h** | **0%** |
| Develop the Final Layout Using Rea... | 0 | 0% |
| Integrate Visualization Tools | 0 | 0% |
| Visualize the Extracted Data | 0 | 0% |
| Review Sprint 2 | 0 | 0% |
| Milestone 5: Data Visualization Das... | 0 | 0% |



| | | |
|---|---|---|
| **Sprint 2: Frontend Development** | **0h** | **0%** |
| Develop the Final Layout Using Rea... | 0 | 0% |
| Integrate Visualization Tools | 0 | 0% |
| Visualize the Extracted Data | 0 | 0% |
| Review Sprint 2 | 0 | 0% |
| Milestone 5: Data Visualization Das... | 0 | 0% |
| **Sprint 3: Integrating Advanced F...** | **0h** | **0%** |
| Implement Geological IP APIs | 0 | 0% |
| Visualize IP Country Flagging | 0 | 0% |
| Implement IP Reputation | 0 | 0% |
| Visualize IP Reputation | 0 | 0% |
| Implement Alert and Notifications fo... | 0 | 0% |
| Test the Advanced Features | 0 | 0% |
| Review Sprint 3 | 0 | 0% |
| Milestone 6: Integrating Advanced ... | 0 | 0% |
| **Testing and Feedback** | **0h** | **0%** |



| | | |
|---|---|---|
| **Sprint 1: Testing** | **0h** | **0%** |
| Test the Data Extraction Process | 0 | 0% |
| Test the Data Visualization Dashboa... | 0 | 0% |
| Test the Geological IP Integration a... | 0 | 0% |
| Test the IP Reputation Integration | 0 | 0% |
| Test the Alert and Notification Syst... | 0 | 0% |
| Test the Overall Tool | 0 | 0% |
| **Sprint 2: Feedback and Final Adju...** | **0h** | **0%** |
| Gather Feedback from Supervisors | 0 | 0% |
| Make Necessary Final Adjustments | 0 | 0% |
| Document the Testing and Adjustm... | 0 | 0% |
| Milestone 7: Testing Phase | 0 | 0% |
| **Deployment and Documentation** | **0h** | **0%** |
| **Sprint 1: Deployment** | **0h** | **0%** |
| Deploy the Tool | 0 | 0% |
| Document the System Architecture | 0 | 0% |
| **Sprint 2: Documentation** | **0h** | **0%** |
| Collect Logs and Artefacts | 0 | 0% |
| Combine the Testing Documents | 0 | 0% |
| Final Report Writing and Feedback | 0 | 0% |
| Milestone 8: Final Report Submission... | 0 | 0% |