COMP3850 Project Deliverable Certificate

Name of Deliverable	*D2- Project Plan, Requirements/Scoping Document and Updated Team Manual*
Date Submitted	*28 / 03 / 2024*
Project Group Number	*Team 26*
Rubric stream being followed for this deliverable	*SOFTWARE Rubric*

We, the undersigned members of the above Project Group, collectively and individually certify that the above Project Deliverable, as submitted, **is entirely our own work**, other than where explicitly indicated in the deliverable documentation.

INITIALS	SURNAME	GIVEN NAME	STUDENT NUMBER	SIGNATURE (IN-PERSON OR DIGITAL)
Rx	*Ruike*	*Xu*	*46271481*	RuikeX
KP	*Kritchanon*	*Prasobjaturaporn*	*46122206*	KritchanonP
LT	*Long*	*Trinh*	*46281657*	LongT
OB	*Bush*	*Oliver*	*46534032*	OliverB
KH	*Karl*	*Holzmann*	*46122206*	KarlH

List of tasks completed for the deliverable and activities since last deliverable certificate with totals for each individual team member and whole team

Performed by *(Student Names)*	Duration *(hrs)*	Complexity *(L, M, H)*	Name of task	Checked by *(Initials)*
Ruike Xu	1h	L	Risk Identification	LT
Ruike Xu	4h	M	Risk Analysis	LT
Long Trinh	6h	H	Project Schedule	KP
Long Trinh	1h	L	Change Management	OB
Long Trinh	4h	M	Introduction of SRS	OB
Long Trinh	4h	M	Functional Requirement	OB
Karl Holzmann	5h	M	Overall Description of SRS	KP
Karl Holzmann	3h	M	Risk Planning of SRS	KP
Kritchanon Prasobjaturaporn	3h	M	Introduction of Project Plan	KH
Kritchanon Prasobjaturaporn	4h	H	Nonfunctional Requirements of SRS	KH
Kritchanon Prasobjaturaporn	3h	L	Sponsorship Management Tasks	LT
Kritchanon Prasojaturaporn	1h	L	Proofreading/Document editing	RX,LT
Oliver Bush	3h	M	Resources Management of Project Plan	KH
Oliver Bush	2h	L	Design and Implementation Requirements/Constraints of SRS	LT
Oliver Bush	3h	M	Usability Requirements of SRS	LT
Oliver Bush	2h	L	Document Formatting/Proofreading	LT
Oliver Bush	1h	L	Risk analysis Matrix	KP
Team Total	50h			

PROJECT PLAN

1. Introduction:

This document represents the project plan of Rapid Summarisation (Summarisation Discord Bot). The goal is to construct a Discord bot that utilises the Rapid Analysis Large Language Model, through API, to summarise Discord chat log.
Included in this document:

- **Introduction** provides insight into the scope of our project. Introducing the idea of what is to be achieved as well as the potential benefits.
- **Risk Management** demonstrates all aspects of risks associated with the project and the guide on how they are handled and managed.
- **Resources Management** displays the project resources usage; what they are and how they are used.
- **Change Management** illustrates a structured approach to address unforeseen adjustments.
- **Quality Management** presents a structured approach to quality control throughout the process of product development; such as quality assurance strategies, updating testing protocols, and creating a collaborative environment for ongoing improvement.

Project Plan Scope

The Large Language Model (LLM), when given a context, is a substantial tool for all-around text-based work tasks. The goal of our project is to improve workflow through simplification of team communication through LLM. Our project is heavily anchored to Agile methodology, an iterative, flexible approach to software development. Hence, as the development progresses, we anticipate refining planned features as well as introducing additional features.

Our current scope is two main text-oriented features. Firstly, ‘Topic Thread’, our product would analyse ongoing conversations and summarise them into topic threads with related comments, this way users can simply catch up by just simply reading the threads. Additionally, `Daily/Weekly Digest`, our product could generate, depending on the user's setting, a daily/weekly digest. A summary of the topics discussed throughout the day or week has the potential to further improve productivity, by further reducing hours spent on communication.

2. Risk Management:

2.1. Risk Identification:

i. Technical Risks:

- **Inadequate Technical Specifications**
- **Technology Obsolescence**
- **Cybersecurity Threats**

ii. People Risks :

- **Talent Shortage**
- **Underperformance**
- **Disengagement**
- **Leadership Gap**
- **Poor Management Practices**
- **Toxic Work Environment**
- **Ethical Misconduct**

iii. Organisational Risks

- **Market Competition**
- **Process Failures**
- **Legal Risks**
- **Financial Risks**

iv. Requirements Risks

- **Ambiguity and Unclear Requirements**
- **Changing Requirements**
- **Stakeholder Involvement and Communication**

v. Estimation Risks

- **Time Estimation Risks**
- **Cost Estimation Risks**
- **Resource Estimation Risks**
- **Technical and Operational Risks**
- **Planning and Methodology Risks**
- **Market and Environmental Risks**

2.2. Risk Analysis

i. Technical Risks:

There are a variety of project risks, including Inadequate Technical specifications, which can lead to confusion and rework if the team does not have detailed technical specifications, which can lead to different ways of completing the project, and which are not interoperable with each other. Moreover, if the market changes too rapidly, even if the project is completed, it may lose its competitiveness because the technology is outdated.

Moreover, If team members do not fully carry out systematic learning of network security, it will also give some phishing emails and information security breaches the opportunity to cause information leakage and cyber attack-related risks to the organisation.

ii. People Risks

Both external and internal causes can lead to People Risks. External causes can be epidemics, natural disasters or even poaching by competing companies, all of which can lead to turnover, resulting in the loss of valuable employees and the inability to quickly replenish the team with the skilled personnel it needs, leading to the suspension of the project.

Additionally, internal causes such as overwork or mismatched wages can lead to negative performance, which affects the team's performance and prevents the project from being finalised at the expected time.

Moreover, poor leadership decisions or inadequate management skills may lower morale within the team or increase the likelihood of conflict among team members, resulting in a lack of proper strategic direction for the team as a whole. A negative team climate and different cultures may also lead to discrimination, bullying, or physical harassment among employees.

iii. Organisational Risks

Irregular changes in the market may lead to changes in market conditions and changes in customer preferences if the organisation is unable to make agile changes at this time, eventually, the team will not be able to adapt to the market environment, and then new competitors will take some market share, and market changes may also affect the existing project plan and lead to plan changes or outdated technology.

Moreover, if the organisation does not conduct a complete review of the project's processes, it can also reduce the efficiency of the business processes and lead to additional property damage.

Furthermore, when there is a change in government policies or regulations, the organisation may also have an impact because its operational strategy or monetisation strategy does not conform to existing policies/regulations.

Also, if there is a large change in the exchange rate of market currencies, it will lead to unforeseen risks in the assets of the organisation, and when users or partners are unwilling to fulfil financial obligations, it will also cause unpredictable risks to the organisation.

iv. Requirements Risks

When the client is unable to define what they need from the project, there is a high probability that the project will be carried out at a stage where the results do not match the client's objectives, thus wasting a lot of time and financial costs.

Also, when the target scope of the project is gradually expanded due to the client's requirements, failure to recalculate the time and resources for new project changes may result in a delay in the scheduled completion of the project beyond the original budget, and how to reallocate the extra resources after the project changes is also one of the risks.

Moreover, if there is no continuous and effective communication with the stakeholders during the project, the needs of the stakeholders may be ignored or misinterpreted, and the results of the project may be different from the expectations of the stakeholders.

v. Estimation Risks

Estimation Risks share some similarities with the previous ones in that additional costs due to external factors may significantly affect the project process if the leader or team is not satisfied with the amount of time required for the project, dependencies between projects, financial requirements, and additional costs due to external factors.

Additionally, misallocation of resources, incompatibility of team members' expertise with current project requirements, faulty project management methodologies, and changes in markets and policies can also affect the project process in the long run, resulting in a failure to deliver the project correctly or delivering a project with many defects.

Risk Analysis Matrix			
Risk Number	Probability	Impact	Priority
i.	Low	Catastrophic	High
ii.	Low	Serius	High
iii.	Moderate	Serius	Moderate
iv.	Low	Tolerable	Low
v	Low	Serius	Moderate

2.3. Risk Planning

Risk Number	Strategies
i.	Have regular meetings with stakeholders discussing requirements and documentation. Try to differentiate ourselves from competitors on the market. Ensure all files containing sensitive credentials are excluded from version control, accounts all have multi-factor authentication and utilise GitHub's token leak scanner.
ii.	Ensure each team member gets the same training/documentation. Check in/have meetings regularly to update the team on progress. Reschedule meetings or otherwise assist team members. Follow conflict resolution documentation in the team manual.
iii.	Try to differentiate ourselves from competitors on the market. Create and follow conflict resolution documentation in the team manual. Ensure we follow a code of conduct and follow conflict resolution if issues occur. Reduce code complexity to minimise costs.
iv.	Have regular meetings with stakeholders discussing requirements and documentation.
v.	Check in/have meetings regularly to update the team on progress. Communicate with stakeholders regularly to ensure accurate estimations are made by subject matter experts. Track costs. Regularly reevaluate estimations.

3. Resources Management:

- Utilise Jira to allocate assignments to developers according to their skills establishing accountabilities for each outcome. Arrange resources in accordance with project stages and key points to sustain advancement and achieve target dates.
- Make sure to check team workloads using Jira reports to ensure a good balance and avoid burnout or skills going underused.
- Keep track of team availability and time commitments using Google Docs so we can adjust resources easily as project needs change.
- Look for ways to make use of resources whenever possible by using automation tools, which can help free up people for challenging tasks.
- Ask our team for feedback, on how resources are being used and any obstacles they face to find ways to improve efficiency.

- Ensure that all parties involved are kept up to date on the availability of resources including any shortages or surpluses by providing updates and dashboard summaries.
- Foster transparent communication with the project sponsor and other stakeholders to guarantee that decisions regarding resources are made openly and collaboratively.

4. Change Management:

4.1. Managing Requirements and Scope Changes:

- Change Request Process: Establish a procedure wherein any suggested modifications to requirements or scope need to be submitted as a change request accompanied by thorough explanations and reasons.
- Change Approval: Other group members will then decide to approve of the change or disapprove. Choices with more votes will be accepted.
- Change Documentation: When a change request gets the light, we will make sure to update all project documents, like the SRS, project plan and Gantt chart to align with the updated requirements or scope.

4.2. Version Control:

- Consider utilising a version control platform, like Git along with GitHub to effectively oversee and monitor modifications made to the project's codebase.
- Utilise the version history tool, in Google Docs to monitor modifications made to documents as time progresses enabling us to switch to versions if needed.
- Tagging should be utilised to indicate the release points, in the repository offering pointers to the versions that have been deployed or made available, to users.

5. Quality Management:

5.1. Develop a quality assurance strategy detailing the methods and measurements to uphold the product's quality during its change implementation.

5.2. Enhance the testing protocols to ensure that changes adhere to the updated criteria without causing any faults or setbacks. This involves revising test scenarios and automated testing sequences to match the updates.

5.3. Nurture an environment that encourages enhancement by assessing the change management procedure seeking input from team members and making necessary modifications.

6. Project Schedule:

6.1. Tasks:

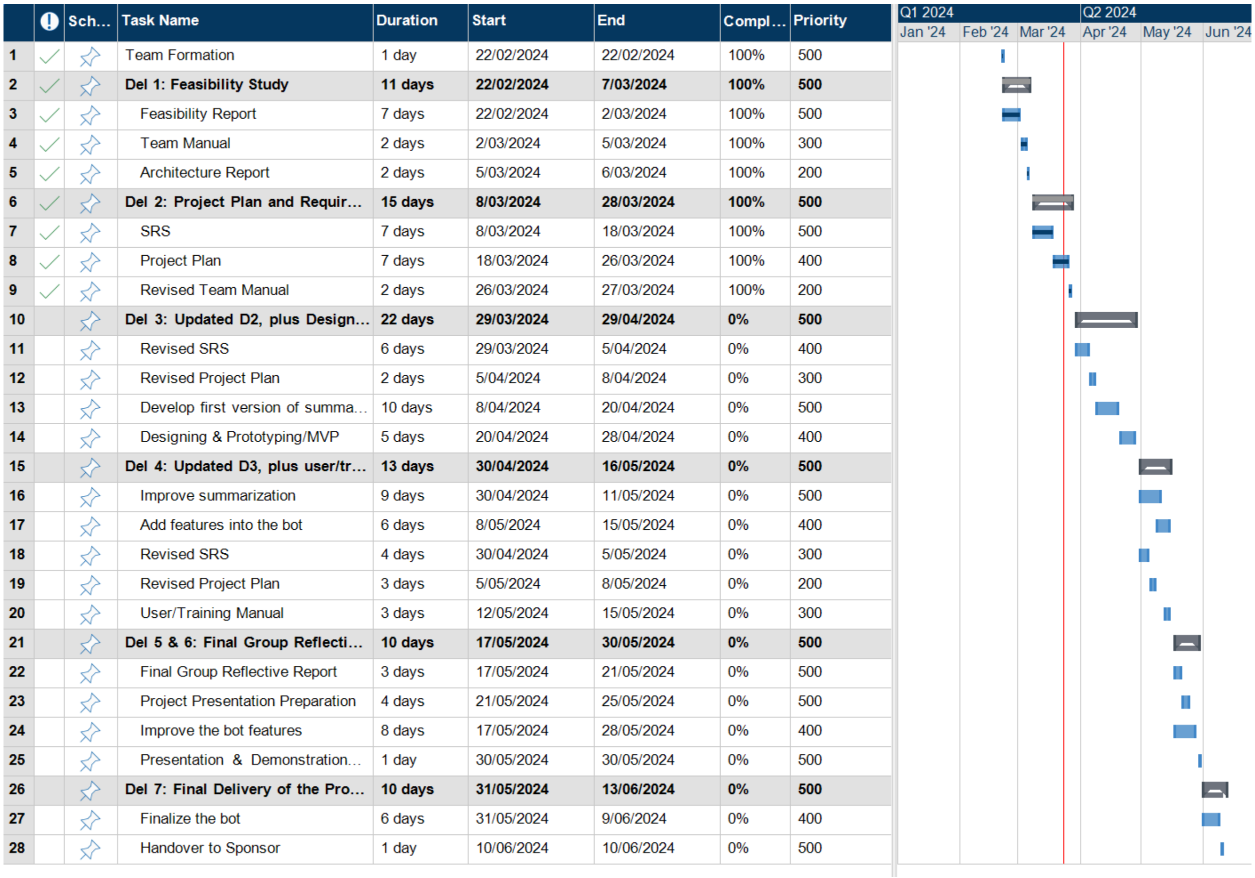
- Task 1: Members meet each other, form the team, and figure out our communication plan, roles, responsibilities and skills. After that, we had time to talk to our sponsor from Rapid Analysis and got a deeper understanding of the project. Then we decided on our bot's main focus and feature: Summarization on Discord conversation (22nd of February, 2024)
- Task 2: Members participate in team training to get a better understanding of each other's strengths, weaknesses and skills. After that, we decided on the approaches to the project and how the system will work. We created the skeleton version of the bot and added it to our discord server. Lastly, we made a Feasibility Report, Team Manual and Architecture Report (23rd of February, 2024 to 7th of March, 2024).
- Task 3: We will research and create the first version of the Project Plan and Software Requirement Specification (SRS). We will also research on the best feature to add to the bot besides chat summarisation (8th of March, 2024 to 28th of March, 2024).
- Task 4: We will research the bot's design, and prototyping, and start to add the initial version of summarisation + Rapid Analysis API integration to the bot. Then we will update the Project Plan and SRS with Design, Test Cases, and Prototype/MVP (29th of March, 2024 to 29th of April, 2024).
- Task 5: We will test and research on how the summarization would work in different scenarios. We will add other features to the bot after that. Then we will update the documentation with the user/training manual (30th of April, 2024 to 16th of May, 2024)
- Task 6: We will test all the features and optimise the functionalities of the bot. In addition, we will also improve the bot's design and finalise it. Then we will create a Final Group Reflective Report (17th of May, 2024 to 30th of May, 2024)
- Task 7: We will demonstrate the bot, Final Report and presentation to the Rapid Analysis representative (30th of May, 2024).
- Task 8: We will polish the bot and make a Final Delivery of the Product to the Rapid Analysis as agreed in the handover (30th of May, 2024 to 13th of June, 2024).

6.2. Deliverables:

- Deliverable 1: Feasibility Study (Thursday, 7th of March, 2024)
- Deliverable 2: Project Plan and Requirements/Scoping Document (Thursday, 28th of March, 2024)
- Deliverable 3 - Increment 1: Updated Deliverable 2, plus Design, Test Cases, Prototype/MVP (Monday, 29th of April, 2024)
- Deliverable 4 - Increment 2: Updated Deliverable 3, plus user/training manual (Thursday, 16th of May, 2024)
- Deliverable 5: Final Group Reflective Report (Thursday, 30th of May, 2024)

- Deliverable 6: Project Presentation / Demonstration (Thursday, 30th of May, 2024)
- Deliverable 7: Final Delivery of the Product to the sponsor as agreed in handover (Thursday, 30th of May up to Thursday, 13th of June, 2024)

6.3. Timeline:



	!	Sch...	Task Name	Duration	Start	End	Compl...	Priority
1	✓	📌	Team Formation	1 day	22/02/2024	22/02/2024	100%	500
2	✓	📌	**Del 1: Feasibility Study**	**11 days**	**22/02/2024**	**7/03/2024**	**100%**	**500**
3	✓	📌	Feasibility Report	7 days	22/02/2024	2/03/2024	100%	500
4	✓	📌	Team Manual	2 days	2/03/2024	5/03/2024	100%	300
5	✓	📌	Architecture Report	2 days	5/03/2024	6/03/2024	100%	200
6	✓	📌	**Del 2: Project Plan and Requir…**	**15 days**	**8/03/2024**	**28/03/2024**	**100%**	**500**
7	✓	📌	SRS	7 days	8/03/2024	18/03/2024	100%	500
8	✓	📌	Project Plan	7 days	18/03/2024	26/03/2024	100%	400
9	✓	📌	Revised Team Manual	2 days	26/03/2024	27/03/2024	100%	200
10		📌	**Del 3: Updated D2, plus Design…**	**22 days**	**29/03/2024**	**29/04/2024**	**0%**	**500**
11		📌	Revised SRS	6 days	29/03/2024	5/04/2024	0%	400
12		📌	Revised Project Plan	2 days	5/04/2024	8/04/2024	0%	300
13		📌	Develop first version of summa…	10 days	8/04/2024	20/04/2024	0%	500
14		📌	Designing & Prototyping/MVP	5 days	20/04/2024	28/04/2024	0%	400
15		📌	**Del 4: Updated D3, plus user/tr…**	**13 days**	**30/04/2024**	**16/05/2024**	**0%**	**500**
16		📌	Improve summarization	9 days	30/04/2024	11/05/2024	0%	500
17		📌	Add features into the bot	6 days	8/05/2024	15/05/2024	0%	400
18		📌	Revised SRS	4 days	30/04/2024	5/05/2024	0%	300
19		📌	Revised Project Plan	3 days	5/05/2024	8/05/2024	0%	200
20		📌	User/Training Manual	3 days	12/05/2024	15/05/2024	0%	300
21		📌	**Del 5 & 6: Final Group Reflecti…**	**10 days**	**17/05/2024**	**30/05/2024**	**0%**	**500**
22		📌	Final Group Reflective Report	3 days	17/05/2024	21/05/2024	0%	500
23		📌	Project Presentation Preparation	4 days	21/05/2024	25/05/2024	0%	500
24		📌	Improve the bot features	8 days	17/05/2024	28/05/2024	0%	400
25		📌	Presentation & Demonstration…	1 day	30/05/2024	30/05/2024	0%	500
26		📌	**Del 7: Final Delivery of the Pro…**	**10 days**	**31/05/2024**	**13/06/2024**	**0%**	**500**
27		📌	Finalize the bot	6 days	31/05/2024	9/06/2024	0%	400
28		📌	Handover to Sponsor	1 day	10/06/2024	10/06/2024	0%	500

6.4. Resources allocated:

6.4.1. People:

Our team has 5 members. Each sprint the roles will be rotated to different members. 3 main roles will be: Project Manager, Lead Developer and Developers.

6.4.2. Tools and Services:

- Discord Environment

- Rapid Analysis API
- Github
- Jira
- Gmail
- Google Doc

6.4.3. Time: Total: 14 weeks

- Researching phase: 6 weeks
- Development phase: 5 weeks
- Testing Phase: 4 weeks
- Deployment Phase: 2 weeks

6.5. Assumptions:

- Resource Availability: It is assumed that all team members and required tools will be accessible as scheduled throughout the project duration.
- API Accessibility: The project anticipates the availability of Discords API and Rapid Analysis's LLM API with rate limits to facilitate development and testing stages.
- Technological Stability: The project supposes that there will be no alterations, to the core technologies, such as updates to Discords platform or removal of APIs necessitating modifications, to the project scope or timeline.
- Budget Stability: It is assumed that the project will not be costly, and will not require the team to spend unexpectedly.
- Supports: The project will receive support from sponsors from Rapid Analysis and University if needed. Assistance can include academic help, tools, licences or servers from Rapid Analysis.
- Risk Management: The project plan operates under the assumption that any risks identified will be handled and minimised in line, with the strategies detailed in the risk management plan without leading to delays.

6.6. Templates/Standards:

- All of our templates are provided by the university. We can add and adjust the templates based on the sponsor's desire or team member's suggestion if approved.
- Our documentation standards will be based on the deliverable rubrics to meet the requirements.

Software Requirements Specification

for

Discord chat summarising bot with Rapid Analysis Large Language Model through API

Version 1.0 approved

Prepared by <author>

Nexus Nomads

9/3/24

Table of Contents

1. Introduction

1.1 Purpose

This Software Requirements Specifications (SRS) document outlines the requirements for the development of a Discord conversation summarisation bot, referred to as the "Rapid Summary Bot", designated as Version 1.0. The scope of the SRS includes the software requirements needed to implement only the core functionality of the Rapid Summary Bot in Discord. The development team, stakeholders, and everyone else with a stake in the project might turn to this SRS as an outline. It specifies the specifications for operation, expected use cases, technical constraints, and functional and non-functional requirements for the Rapid Summary Bot's development and deployment.

1.2 Document Conventions

- Fonts and styles:
    + Headings: Headings will be in Time New Roman. Their sizes are from 18-14-12 depending on hierarchy.
    + Statements: Statements will be in Arial. Their size is 11.
- Highlightings: **Key Terms and Important Concepts** will be emboldened.

1.3 Intended Audience and Reading Suggestions

The Rapid Summary Bot, Version 1.0 Software Requirements Specifications (SRS) document is created with the intention of appealing to a wide range of audiences:

- For **Developers/Lead Developers**, who are mostly interested in the technical specifications, the sections that address system features, interface requirements, and functions will be especially helpful.
- For **Project Managers**, who are mostly interested in the overall scope of the project, milestones and resources, the Introduction and Overview sections will be helpful for them to start.
- For **Sponsors and Lecturers**, who would be most interested in the project's goals, and desired final product, Overall Description will be the most crucial section for them to focus on. Before that, they should have a brief look at the Introduction
- For **Users**, in order to understand the bot's capabilities and how to communicate with it, users need to read the overview first. Insights on how the bot could enhance their Discord experience can be gained by understanding the functional needs.

Document Organisation and Reading Sequence:
This SRS will be divided into 3 major parts:

- **Introduction:** The objective, scope, and target audience of the document are all summarised in the introduction.
- **Overall Description:** Provides audiences with an up-close view of the bot's capabilities and constraints by outlining a larger context of the product's requirements and most significant components.

- **System Features and Requirements:** Specifications for the bot's creation and operation are laid forth in the system requirements and features section, which delves into functional, non-functional, and interface requirements.

Reading the project's Introduction and Overall Description first will provide readers with an organised reading experience and a firm grasp of its essentials. After that, according to what was said before, readers should concentrate on the parts of the SRS that are most applicable to their interests and duties, so that they fully understand the material that is important to their position in the project. This method makes sure that all kinds of readers may quickly find and grasp the information that's important to them in the document.

1.4 Project Scope

By automating the generation of summaries of chat conversations, the Rapid Summary Bot improves user experience and makes it simpler to navigate topics on Discord. Confronting the problem of information overload, its creation is in line with the objective of enhancing community interaction and information accessibility on Discord.

Our Main Objectives:

- Make it simpler for people to join or rejoin ongoing conversations to increase engagement. Quickly find relevant talks without having to go through whole chat histories; this will save time for both users and admins.
- Foster Unity Within the Community: Facilitate a more welcoming community by making sure everyone can keep up with the latest news.
- The initiative's direct support for bigger business projects to enhance engagement in digital communities and user satisfaction leads to increased platform adoption and retention, among other benefits.

Vision Statement:
A unique solution that simplifies the consumption of large conversation data is offered by the Rapid Summary Bot, with the vision of redefining the way users connect with Discord. By using state-of-the-art natural language processing technology, this bot is designed to improve the Discord experience. It does this by summarising chat in a server in a concise and appropriate manner, so users can easily catch up on and join in on conversations. Therefore, our main objective is to make it easier for community members to stay united and put their effort into building a better server. In addition, it will also enhance members' experiences and the quality of Discord servers. Incorporating this vision into the Rapid Summary Bot will help us achieve our goal of enhancing digital communication and community engagement on the platform, which is to become an indispensable tool for both users and administrators.

Scope:

- **Initial Release (Version 1.0):** Gives top priority to developing tools that accurately summarise chat discussions in order to support English-speaking servers. Important considerations include providing understandable summaries, making it easy to install bots within servers, and ensuring data processing privacy and security.
- **Future Developments:** Future plans include adding more diverse functions into the bot, such as chat analysis, moderation or more features from Rapid Analysis API.

- **Target Audience:** Both Discord server owners looking to improve community management and users looking for a more streamlined method to interact with chat data may benefit from this.
- **Business Strategy Alignment:** The goal of developing the bot is to enhance digital communication technologies, namely Discord, so that it attracts more users and keeps the ones it already has happy.

1.5 References

- Discord Developer Documentation:
  + Author: Discord Inc.
  + Version: 10
  + Date: September 1, 2022
  + Location: https://discord.com/developers/docs
  + Description: The Rapid Summary Bot's development relies heavily on the information provided by this Discord API documentation, which includes development guidelines, authentication methods, rate restrictions, and terms of service.
- Rapid Analysis LLM API Documentation:
  + Author: Rapid Analysis
  + Version: 1.0
  + Date: Dec 1, 2023
  + Location: https://rapidanalysis.github.io/tos.html
  + Description: Describes the Rapid Analysis LLM API, which the bot will use to summarise texts, including its capabilities, limitations, and technical requirements.

2. Overall Description

2.1 Product Perspective

The Rapid Summary Bot is a new, self-contained product. It will allow users of the Discord chat platform to be able to interface with the LLM run by RapidAnalysis.

2.2 Product Functions

The Discord Chat Summary Bot will allow users to access many different functions through the Discord chat platform. Listed below are the features of the Discord Chat Summary Bot.

2.2.1 Automatic summarisation of Discord conversation threads.
2.2.2 Real-time generation of concise summaries highlighting main talking points.
2.2.3 Integration with Discord servers to provide summary services to users.
2.2.4 Personalised summarisation based on user preferences and server settings.
2.2.5 Dashboard functionality for server administrators to view analytics and insights.

2.3 User Classes and Characteristics

There are three different types of users for the Discord Chat Summary Bot:

Type 1. Regular Discord users: Individuals who participate in Discord communities and have the opportunity to benefit from the summarization services provided by the bot.

Type 2. Server administrators: Users responsible for managing Discord servers who utilise the bot's dashboard functionality to gain insights into server activity.

Type 3 Developers: Individuals involved in developing and maintaining the Rapid Summary Bot.

2.4 Operating Environment

The RapidSummary Bot will operate within the Discord platform, utilising Discord's API for seamless user interaction when added to a Discord server. It will be compatible with the various hardware platforms and operating systems supported by Discord, including web, desktop, and mobile environments. The bot will be designed to run on any machine that is capable of running the Node.js runtime and has a network connection but will ideally run on a Linux machine.

2.5 User Documentation

User documentation can be delivered in Discord with the use of a help command. This can return a paginated help message or a link to online documentation. While typing a command, Discord will also show a short description of the command and the parameters for the command, allowing users to autofill.

2.6 Assumptions and Dependencies

We will assume that each server has a sufficiently active user base to justify automatic summarization and provide enough context to the LLM for coherent and accurate responses and that users have a basic understanding of how to use and navigate within the Discord app. We will also assume that the bot is provided all the permissions it requires and that users are given permission to use commands in their servers. The bot will depend on 3 libraries available on NPM. These libraries are discord.js to provide us with a native interface to the Discord API, and node-fetch to provide a simple way to perform network requests to the RapidAnalysis API and dotenv which allows us to load secret values from a file rather than hard coding them. We also depend on both the Discord and RapidAnalysis APIs to be stable and freely available.

3. Requirements

3.1 Functional Requirements

R.1. Chat Summarisation:

Description and Priority

- If a Discord server has this capability, the Rapid Summary Bot can automatically summarise chats inside the server. The goal is to provide them with a synopsis of the conversation by pulling out the main ideas and topics. This feature is highly prioritised since it is essential to the product's goal of improving the user experience by reducing information overload.

Chat Summarisation Requirements

- **R.1.1.Summary Generation**
    - Description: The system shall generate a summary based on user desired instructions, such as from a certain part or period of the conversation.
    - Error Response: The system shall send an error message to the user outlining the problem if summarisation is not possible because of incoherent data or any other reason.
- **R.1.2. Summary Presentation**
    - Description: The system shall give the user the summary inside Discord, but make it stand out from the rest of the discussion by utilising a clean, easy-to-read style.
    - Error Response: The system shall send an error message to the user to inform them if it cannot send messages or retrieve messages within the server because of no permission provided or any other problems.
- **R.1.3. Authentication and Authorisation**
    - Description: The system shall securely connect with Discord's API and gain the appropriate permissions from the server admins, in order to access the conversation logs for summarization.
    - Error Response: The system shall notify the requesting user of the exact problem and offer suggestions for resolving it in the event that authentication fails or permissions are inadequate.
- **R.1.4. Integration with Discord Features**
    - Description: The system shall integrate seamlessly with Discord's existing features, such as roles and permissions, to respect user privacy and server settings.
    - Error Response: The system shall notify the user if they try to use a higher command than their current role or if there is a problem with the compatibility with Discord features and the bot
- **R.1.5. Interactive Summary Requests**
    - Description: The system shall enable users to request summaries in a dynamic way using chat commands, narrowing their search by subject, time range, or channel.
    - Error Response: The system shall notify users if there are problems or restrictions with handling interactive requests, such as commands that were not recognised or channels that were unreachable.

R.2. User Customisation and Preferences:

Description and Priority

- Users may adjust the Rapid Summary Bot's actions based on their own or the server's preferences, such as the amount of time between summary updates, how often they occur, and which subjects are most important to them. Medium, which plays a significant role in improving user happiness and engagement via customisation of the summary process to suit varied demands, is the focus of this feature.

User Customisation and Preferences Requirements

- **R.2.1. Interface for Customisation**

  - + Description: The system shall offer a user-friendly interface that lets users customise summary parameters, such as the length, frequency, and subjects that interest them. This may be done by commands in Discord or a graphical user interface that is available on the server.
    + Error Response: The system shall send a helpful message outlining legitimate customisation choices that will be shown by the bot in response to any user input that is deemed invalid.
- **R.2.2. Preference Persistence**
    + Description: The system shall keep user choices for personalisation safe by linking them to their Discord profile; this way, preferences will follow the user from session to session and server to server.
    + Error Response: The system shall notify users and try again shortly if there's a technical problem and storing preferences doesn't work.
- **R.2.3. Server-Level Default Settings**
    + Description: The system shall provide an option for server administrators to define default parameters for summarization, which users may then customise according to their preferences.
    + Error Response: The system shall let the administrator know and offer other ones if the default settings don't save or don't work with the bot's capabilities.

R.3. Error Handling and Recovery:

Description and Priority

- Users may get to know the errors that they encountered and they will also be sent to the bot developers. The priority of this feature is below medium, it is crucial to get user feedback and to improve the bot's functionalities.

Error Handling and Recovery Requirements

- **R.3.1. Error Detection and Logging**
    + Description: The system shall automatically detect and log errors. It will be sent to developers and feedback can be added by users for further information. This will help developers to have an in-depth understanding of the error and its condition.
    + Error Response: The system shall notify users if the error is unknown. Then, detailed information of the error will be sent to developers and users' state will be reverted back to previous functional action.
- **R.3.2. Error Recovery**
    + Description: The system shall automatically revert back to the previous functional state to continue receiving the user's commands and show a message to ask the user to try again. This will help limit the error range and possibility for developers.
    + Error Response: The system shall notify users and try again shortly if it cannot revert back. If the second revert still did not work, the bot shall revert back to its first state (the "home" state).

3.2 Design and Implementation Requirements/Constraints

R.4. Operation within Discord

+ Description: The system shall operate within the Discord application.

R.5. Rapid Analysis Integration

+ Description: The system shall integrate its functionality through rapid analysis API

R.6. Implementation Language

+ Description: The system shall be implemented using Javascript

3.3 Usability Requirements

R.7. Command usage

+ Description: The system shall use a slash command within the discord platform before a command word for each system function.

R.8. User Documentation

+ Description: The system shall have clear and easily accessible user documentation for training.

3.4 Nonfunctional Requirements

R.9. Performance:

Description and Priority:

- Numerous Discord servers have extensive member count, exceeding 10,000 users, for instance. However, a closer inspection often reveals a minimal daily engagement, with perhaps only 20 to 30 members initiating conversation. Hence, an effective engagement metric to consider is the daily active users engaging through messaging. Specifically, having at least 250 unique users sending messages daily is indicative of an active community. This level of interaction is anticipated for our product's environment.

Performance Requirements:

- **R.9.1.** The system shall have a maximum return time of one minute irrespective of the volume of users.

R.10. Safety:

Description and Priority:

- Our product is built around our users' chat log information which may contain sensitive information about the users, and their acquaintances. Given this, the concern of sensitive data leakage is the most pressing matter regarding safety and security.
- Our framework prevents data leakage by ensuring that the bot does not store personal data beyond what is necessary for the summarisation process, as well as complying with relevant privacy laws. The bot will be compliant with the Australian Privacy Principles (APPs) under the Privacy Act 1988, ensuring the protection of personal information.

Safety Requirements

- **R.10.1.** The system shall not store personal data beyond what is required for the summarisation process.
- **R.10.2.** The system shall be compliant with the Australian Privacy Principles (APPs) under the Privacy Act 1988.

R.11. Security:

Description and Priority:

- Several requirements have been identified, to ensure the potential sensitivity of user data and prevent unauthorised and misuse of our product.
- First requirement, 'Discord Bot token' must be under maximum protection. Every Discord Bot has its own unique assigned code, those in possession of it are granted the access to the bot and the abilities to manipulate the bot's behaviour, access its permissions and information. Hence, the possible consequences of token code leakage is astronomically. Undeniably, the access to the token is limited, and distributed strictly on a need-to-know basis to ensure maximum security. Any file that may contain the token and any other sensitive information will be excluded, those files will be updated to **`gitignore`** file. This prevents any leakage during the version control process.
- Second requirement, our product must operate with server permission to the minimal extent necessary. This is to prevent our product from becoming over-privileged and allow for misuse for malicious activities.
- Lastly, the respect to the Discord rate limits. The software user request limit rate must be set according to the specified limits of Discord rules. This is to prevent our software from receiving a ban from Discord marketplace.

Security Requirements

- **R.11.1.** The system token shall be protected by maximum security.
- **R.11.2.** The system token shall be distributed strictly on a need-to-know basis
- **R.11.3.** The system shall operate under the least privilege necessary
- **R.11.4.** The system shall respect specified limits set by Discord

R.12. Software Quality Attributes:

Description and Priority:

- The successful deployment and operation of the product involves multiple software quality attributes. Here are the necessary attributes:
- **Adaptability**, our software is set to be highly adaptive, which means it must possess the ability to adjust to all types of work environments; such as from a small group of users with leisure interaction to a vast community with extensive interaction. A continuous

communication with our partner to ensure the infrastructure can sustain high-traffic periods and scale according to user demand.

- **Availability,** our software definition of high availability is to be reliable, fails infrequently, and easily maintainable, can be easily and quickly repaired. Good coding practices are essential for easy **maintenance**, and extensive code reviews help prevent poor code from being committed.
- The **Correctness** of our program is to be measured through testing with users in real-world scenarios. Their feedback will be the criteria for correctness. Adjustments will be made accordingly.
- **Flexibility** is the degree of difficulty in refining or adding features to the current form of the software. Our software principle is built around API usage, essentially a bridge between two software with a streamlined interface for the users. As our LLM partner releases improvements and new features, our software, like a universal adapter, guarantees effortless integration.
- Our software exhibits a high degree of **usability** by the nature of Discord Bot. For a user that wishes to use our software, it is figuratively and literally like communicating with other users.

Software Quality Attributes

- **R.12.1.**The system shall adapt to all types of work environments, regardless of the usage
- **R.12.2.**The system shall maintain high availability, ensuring minimal instances of failure. In the rare event of a failure, swift and straightforward remediation will be ensured
- **R.12.3.**The system shall maintain a high level of flexibility such that any modification or improvement is effortlessly integrated.
- **R.12.4.**The system correctness shall be measured through the feedback of its users
- **R.12.5.**The system shall be user-friendly by providing clear, concise instructions and feedback to users for each transaction.

4. Sponsor Feedback:

- **Meeting date and time:** 28/03/2024
- **Feedback received:** Advice for further updates in Deliverable 3 and 4 regarding testing and prototyping
- **Team response/action points:** No action was needed

TEAM MANUAL

1. Team Organisation and Structure

1.1. Team Organisation

- Our team is organised around a flat and flexible structure designed to promote open communication, collaboration, and agility in decision-making processes.
- In the team there are three positions; Project Manager, Lead Developer and Developers. The Project Manager and Lead Developer roles will be assigned to team members. Will switch every sprint.
- **Project Manager:** is responsible for monitoring the projects advancement ensuring that goals are achieved and managing resources and tasks efficiently.
- **Technical Lead:** is in charge of making choices supervising the development process and handling technical obstacles effectively.
- **Developer:** ensure that tasks are completed promptly, accurately and proficiently.

1.2. Structure

The team setup is meant to be adaptable allowing for changes in task assignments depending on the project stage and workload of each team member. This setup promotes collaboration across functions giving team members the opportunity to pitch in beyond their responsibilities when needed. Consistent meetings and updates help the team work together smoothly ensuring that any changes in roles or task distributions are communicated openly and agreed upon by all.

Team Roles

Deliverable	Project Manager	Technical Lead	Developer
1	Long Trinh	Karl Holzmann	Kritchanon Prasobjaturaporn, Oliver Bush, Shuo Huang
2	Kritchanon Prasobjaturaporn	Oliver Bush	Ruike Xu, Long Trinh, Karl Holzmann
3	Oliver Bush	Ruike Xu	Long Trinh, Kritchanon Prasobjaturaporn, Karl Holzmann

- Oliver will be our next Project Manager. He has great skills in communication and a strong connection with each team member. For deliverable 3, we are introducing our main feature, summarisation function, into our software, hence it would be substantial for us to have a Manager who excels in managing workflow.
- Ruike is our new team member, he is knowledgeable in software and managing resources. Therefore, he's well-equipped to not only deepen his understanding of our project but also to significantly improve our bot development workflow.

2. Team Values & ACS Code of Professional Conduct

2.1. Team Values

- **Collaboration:** We strongly believe in the effectiveness of collaborating, leveraging our skills and expertise to reach our project objectives efficiently. For a software development project, cooperation between team members is necessary.
- **Integrity:** Our dedication to integrity means we uphold honesty and transparency in every aspect of our work prioritising responsible practices. All of our members have to be transparent between each other to create a better group work environment. Especially for this project, we are required to respect users' privacy.
- **Respect:** We cultivate a culture of respect within our team appreciating the viewpoints of each member and nurturing an atmosphere.
- **Honesty:** Our unwavering commitment to honesty guides all our interactions and endeavours ensuring transparency, in everything we do.

2.2. ACS Code of Professional Conduct

As members of the broader computing community, we align our conduct with the Australian Computer Society (ACS) Code of Professional Conduct, which emphasises:

- **Honesty:** We are dedicated to being honest in everything we say and do.
- **Trustworthiness:** Our team is dependable. Creates an environment where everyone can trust each other.
- **Respect:** We value the dignity, viewpoints and efforts of all team members.
- **Confidentiality:** We protect all information gathered during project work to keep user data and proprietary technologies safe, from access or disclosure.
- **Professionalism:** We maintain professionalism in every project interaction nurturing an environment of respect, equity and honesty.

3. Project Management Approach and Tools

3.1. Approach

Our strategy for project management focuses on using methods, which are customised to suit the changing demands of projects and the evolving needs of creating a Discord bot that summarises chat logs. Agile methodology supports progress allowing us to respond promptly to modifications and input.

3.2. Tools

- **GitHub:** As our version control system, GitHub will host our project code, allowing for collaborative development and code review.
- **Zoom:** For our regular meetings and sprint reviews with the sponsor, we will use video conferencing tools like Zoom.
- **Jira:** For comprehensive project management and sprint planning, Jira is our primary tool. It offers advanced features for issue tracking, task assignment, progress tracking, and Agile reporting.
- **Google Docs:** For documentation, meeting notes, and collaborative editing, Google Docs serves as our central repository.

4. Communication Plan and Meeting Schedule

4.1. Communication Plan

Our communication plan is designed to ensure effective and efficient information exchange within the team and with external stakeholders. To support our project's needs, we employ a multi-platform strategy:

- **Jira:** As our primary project management tool, Jira will be used for detailed tracking of tasks, sprints, and progress.
- **Google Docs:** For collaborative document creation and sharing, such as meeting agendas, minutes, reports, and project documentation.
- **Discord:** Serving as our main communication hub for day-to-day interactions, quick updates, and informal discussions.
- **Email:** For formal communications, especially with external stakeholders, including project sponsors and lecturers.

4.2. Meeting Schedule

- **Weekly Team Meetings:** Held every Thursday at 4 PM via Zoom and one other day of the week via Zoom, these meetings are for full team updates, discussing progress, addressing challenges, and planning for the upcoming week.
- **Weekly Sponsor Meetings:** Held on the first Thursday of each week or fortnightly via Zoom, these meetings are dedicated to presenting our progress to the sponsor, discussing feedback, and aligning on expectations and requirements for the next phase of the project. The Project Manager and Lead Developer are mandatory to attend.

5. Change Management and Conflict Resolution

5.1. Change Management

Change management within our project is crucial to adaptively manage any alterations in project scope, technology, team dynamics, or external factors. Our approach is structured around flexibility, communication, and documentation:

- **Change Review Process:** Each change request will be reviewed during our weekly team meetings.
- **Decision Making:** Decisions on change requests will be made collectively, considering the project's objectives, resources, timeline, and the potential impact on quality and stakeholder satisfaction.
- **Implementation and Monitoring:** Approved changes will be implemented according to a plan that outlines the steps, responsibilities, and timeline.

5.2. Conflict Resolution

- **Open Communication:** Encourage team members to express concerns or disagreements openly and respectfully, ensuring a safe environment for dialogue.
- **Majority rule:** all suggestions in a conflict will be considered and assessed by all team members, and whichever of them gets more approval will be chosen. Each team member will be counted as one vote. If it leads to a draw, the sponsor will have the final decision.