Detection of Energy Consumption and Performance of Third-party Applications using Machine Learning and Artificial Intelligence

22_23-J 87

Project Proposal Report

Rajapaksha R.M.

B.Sc. (Hons) in Information Technology Specializing in

Software Engineering

Department of Computer Science and Software Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

June 2022

Declaration

We declare that this is our own work and this proposal does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of our knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Name	Student ID	Signature
Rajapaksha R.M.	IT19156316	[illegible]

The supervisor/s should certify the proposal report with the following declaration.

The above candidates are carrying out research for the undergraduate Dissertation under my supervision.

Signature of the supervisor: Appendix 1 Date: 26/09/2022

Abstract

In today's world, huge amounts of information are transferred from one party to another. Technology evolves at a rapid pace every day, from the final to the developer. As ‘good’ technology advances so does 'bad' technology, which could be a critical factor in terms of security and efficiency. As a result, a user is urged to use the innovation before trying to dismiss it, and initial attempts are accompanied by regrets. As a result, a framework and application for anomaly detection and prevention could prevent third-party software from causing harm to the user's device before it is installed. This application/framework will provide information about the third-party application as well as detect its security measures, potential device harm in terms of energy consumption, and whether the specifications are met.

Keywords: detection, detect, efficiency, prevention, security,   measures

Table of contents

List of figures

List of tables

List of abbreviations

- NSTRC – Non-Specific Technology Risk Calculator

1. Introduction

The developer tests the performance. He or she can generate a performance report of the developed part. This report is actionable, and if the developer wants a hard copy of this report, he can generate it in a pdf format. This report contains all of the performance data as a percentage value, as well as performance issues and improvements. Also, this report shows a visual flow that includes the flow of getting the best performance. This visual flow has multiple steps. A developer can follow these steps to solve the performance issues that the report shows. This Report Gives a Very Developer-Friendly View of Viewing Improvement by Highlighting the Code Lines with Line Numbers. Also, the NSTRC reports support multiple programming languages.

NSTRC also provides an extension. The NSTRC Extension is a Developer-Friendly Extension for Visual Studio Code. Developers can go to the Visual Studio code editor and install this developer-friendly extension. This checks syntax errors when a developer is writing code. If the developer is making some mistakes, the extension speaks and tells the developer what the problem is. Also, the developer can visually see what steps he needs to take to fix that problem. And this will provide a solution for the developer's mistake and a very performance-friendly approach to solving the problem. This extension has multiple configurations. This extension can identify syntax errors in multiple programming languages. When a developer is ready to use the extension, he or she can configure it for the programming language that he or she is currently using. He can also enable or disable the extension. And he can choose the audio language that this extension needs to speak.

1.1 Background & Literature survey

Many developers are coding to solve a specific problem. But they do not focus on maintaining the efficiency of their code when they are writing code. They do not consider the performance or code quality of their code while they are coding. Many developers only want to do their tasks quickly, not in an efficient manner. Developers always need to be looking to maintain performance before the application is deployed. For those reasons, they need a quick and easy way of seeing what performance issues are arising in their developed code and what the best ways of handling those performance issues are. If developers do performance testing of software components and get a full view of those details, a software development organization can achieve the following goals.

- A developer can see the speed, accuracy, and stability of the software. If the software component speed, accuracy, and stability are at a low level, the developer needs to see what the issue is and what fixes he or she can do for it.

- If a developer performs performance testing and sees a developer-friendly report of performance problems, finds a solution to the problem, and follows a visual flow of performance problem fixing, he or she can deliver a high-quality software component to the next environment. This will result in an error-free, less maintained software component. Also, this will be able to make end users happy.

- Improving optimization and load capability: The power to enhance optimization and load capacity is a further advantage of performance testing. Your organisation can manage volume so that your software can handle high user volumes by measuring performance.

Performance testing of your programme has numerous advantages for your company. You may evaluate the speed, stability, and accuracy of your software by measuring performance. Additionally, it can assist you in addressing any flaws and fixing any issues before you release them to your end users, allowing you to handle scalability. Developers can see what the performance issues are and what improvements in visual flow are needed to solve those performance issues by using the report generation functionality that is provided by NSTRC.

Also, developers need a quick way to handle issues while they are coding. When they have made any coding mistakes (syntax errors), they need a better and faster way of handling those issues. They need to ensure they are writing error-free code. When developers are coding, sometimes they are in very stressful situations. In those situations, they would like to have an assistant if they can. They very much like to find syntax issues and other improvement issues. When developers are in very stressful situations, it is a very easy way to find the syntax issue.

1.2 Research Gap

There are tools available for performance testing purposes at present, but there is no system for generating a report with a nice and developer-friendly structure. This NSTRC Report Generation Tool provides a nice structure for presenting performance problems and improvements.

Other tools do not provide an insight-based flow for fixing the performance issue. This is very developer-friendly. A developer can follow the flow given According to the NSTRC report, he or she can fix the performance problem easily.

There are multiple extensions available in Visual Studio Code for marking syntax issues. But there is currently no feature available for real-time speech. The extension provided by NSTRC gives a solution for this. When a syntax issue occurs in the Visual Studio code editor, it will tell the developer loudly. This is much more user-friendly than other extensions in Visual Studio Code.

This NSTRC extension also provides code improvements if there are any available for the code. The present extension for syntax issues is available for one programming language. But NSTRC Extension does not have such a limit. A developer can switch from one programming language to another programming language.

Research/ Product	Audios speak ability	Multiple programming language support	Highlighting syntax issues
NSTRC extension	✔	✔	✔
jshint	x	x	✔
C#	x	x	✔

Table 1.2. 1: Research Comparison

Research/ Product	Insights based Visual Flow of Fixing performance issues	Multiple programming language support	Displaying performance problems
NSTRC Report	✔	✔	✔
Apache JMeter	x	x	✔
JProfiler	x	x	✔

Table 1.2.2: Research Comparison

1.3 Research Problem

Developers currently do not have a nice and efficient way of getting details about performance issues and improvements. They don’t have an insight-based effective visual flow generator in their present systems. They will write the code for specific functionality. The developer will test the function. If it works correctly, they can’t measure how this specific functionality affects the performance of the application. When running an application in a development environment, if they find some performance problems, they cannot correctly identify where the problem is and how to solve it.

Also, present developers don’t have a nice way of fixing syntax issues. Sometimes they are very busy and stressful, making you sleepy. For those reasons, while they are coding, they can mix some syntax issues with improvement issues. If they are currently using an extension for getting syntax issues in VCS code, they can skip those issues mistakenly. When developers are writing code, they use some extensions to see syntax issues. But developers are also humans, so they can mistakenly skip those syntax issues. Also, currently, if they want to write code using a different programming language, they need to install another extension for that programming language.

Example for syntaxes of two different programming languages:

Programming language	identifier	value
Java	int variable1	90
JavaScript	let variable1	80

Table 1.3.1: Programming language syntax

In the above table I shows different syntaxes of Java programming language and JavaScript programming language. We can see when declaring a Java variable(variable1), it needs to put datatype before the variable name. but if you write variable without adding datatype, it is a syntax error in Java. Also, when we are initializing Java variable, we must enter write value belong to the datatype. For example, we cannot use string values instead of int values (int variable1 = “Hello Java” => incorrect syntax).

If we look at JavaScript variable declaration, we can see it has let instead of int. In JavaScript We cannot use int instead of letting or var or const (const and var also keywords in JavaScript). If you put int, it is a syntax error in JavaScript. We do not need to add datatype, Reason is when JavaScript program is running, it will dynamically get data types of variables according to their assigned values. When getting those syntax issues by extension, there is Not a current extension for speak and tell the problem when syntax issue is occurred.

2. Objectives

2.1 Main Objectives

1. Report For generating details Performance issues, Improvements for achieving code quality, performance level of specific functionality and generating Visual flow of fixing performance problems by step by step
2. Extension for Finding Real time syntax issues and Improvements with Voice Assistant (Realtime speaking about syntax issue loudly)

2.2 Specific Objectives

1. Generating report based on performance measurements.
2. Collect dataset and model training for Generating insight based Visual flow to Fix performance issues.
3. Extension for finding syntax issues with multi programming language support.
4. Model training for different syntax issues

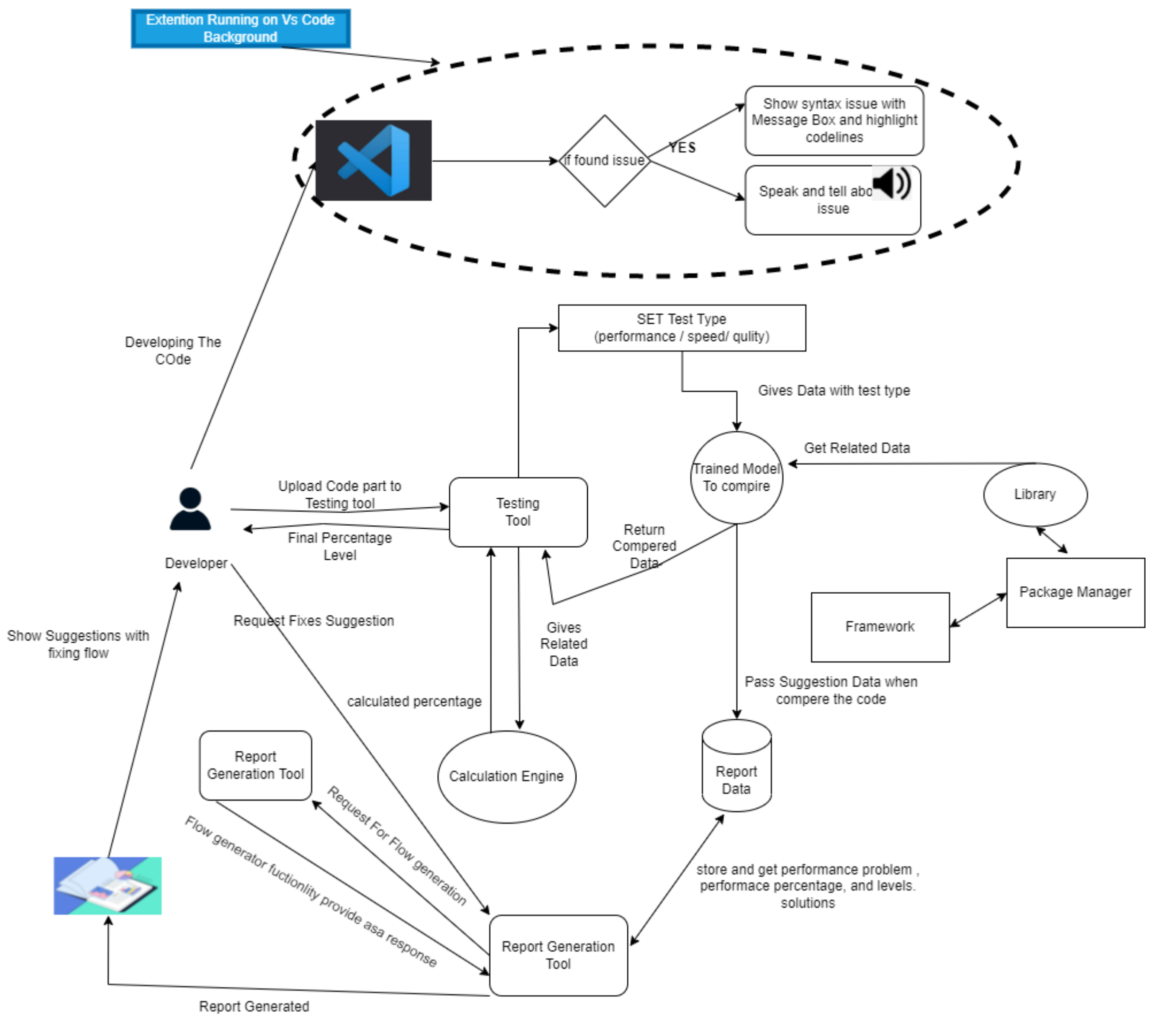
3. Methodology

HTML, CSS, and JavaScript are the technologies used for report generation. All the performance problems are stored in the Developer Platform database. For one programming language, there may be different performance issues. First, we will see the different performance issues that belong to one programming language. Now I will analyze what the different performance fixes are available to do for different performance problems. We need machine learning in this part. First, we need to train a data model for different scenarios (different solutions to different problems).

There is a button in the NSTRC Developer platform. We will use this button to go to the Report page. When a developer clicks this button, an event is triggered to obtain the performance issues of the top component being analyzed. Then we can get the most suitable solution for specific problems. When a new problem arises, the system will choose the most appropriate solution for that performance issue and its level (this level is used for insight-based performance issue fixing) and its solutions will be stored in the report section of our database. That database operation is for insight-based solution generation. The reason is that we can't say we have trained a data model for every problem. After we store the performance problem and its solution in the database, a report needs to be added with the code lines where the performance issue occurs (it can be an unnecessary loop, an unwanted API call, etc.). The system gets a screenshot of those code parts and stores them in the Issue section of the database with a Problem Identifier. The system also calculates the performance of uploaded code components in terms of high-quality code. This performance level (as a percentage value) will also be stored in the Report section of our database. Now I must implement the flow generator. Some insight-based AI solutions will be required for that. At this moment, I will implement the code to analyze the specific solution to that performance problem. A performance problem has a level that says it is a low problem, medium problem, or major problem. One solution has multiple steps/improvements. Because of that, depending on the problem level (low, medium, or major), the system needs to get the steps needed for that problem. For example, if it is a low-level problem, it has a few more steps compared to a medium-level problem. To do that, I need to train another data model with machine learning concepts to identify what the needed steps are. When the flow generator component is running, these steps will gather and generate the flow and store it in our database section. Now all the importations necessary for report generation are ready. When a report is in the process of being generated, it will retrieve the performance level, generated flow, problem-related details, and code parts where the problem occurs (previously stored data) from our database and structure that information in a very effective and developer-friendly manner before redirecting it to the report page of our NSTRC developer platform. On this report page, we can see all our gathered information (performance problems, solutions, and visual flow of fixing performance problems). If the developer wants a hard copy of this report, he can click the button available on the report page and get a PDF version of this report.

There is a visual studio code extension provided by NSTRC. This extension will be developed to identify syntax issues in multiple programming languages in a developer's writing code and give it a very helpful, developer-friendly manner. When I start to develop this, I first need to identify the different syntax issues in different programming languages. This needs to train a data model to learn to identify these syntax issues and different solutions for those syntax issues in an effective manner. This data will be stored in a database for our NSTRC extension. Now I need to implement an event using Typescript and Java Script. If a developer is writing code in a specific programming language and makes a mistake (syntax error), this event will be triggered automatically. I need to implement this extension for different programming languages. Therefore, there will be an option to choose the programming language that he or she is currently using to write the code in Visual Studio Code. When the extension is started in Visual Studio Code, I need to implement a start function to get the programming language that he has chosen. After getting the programming language from the developer, this language value will be stored in a global variable in our extension code. When a developer or programmer makes any syntax error, the previously mentioned event occurs, and in this event, it will check what the problem is (syntax issue, programming issue) and get the best solution for that issue. And after that, the extension will automatically raise a voice to loudly speak and tell you what the issue is. Concurrently, the extension will highlight the code line where the syntax error occurs. Also, it will visually display the issue in the code editor.

3.1 System Diagram


Extention Running on Vs Code Background
if found issue
YES
Show syntax issue with Message Box and highlight codelines
Speak and tell abo issue
SET Test Type (performance / speed/ qulity)
Gives Data with test type
Get Related Data
Developing The COde
Trained Model To compire
Library
Upload Code part to Testing tool
Testing Tool
Final Percentage Level
Return Compered Data
Developer
Package Manager
Framework
Show Suggestions with fixing flow
Request Fixes Suggestion
Gives Related Data
Pass Suggestion Data when compere the code
calculated percentage
Report Generation Tool
Calculation Engine
Report Data
Request For Flow generation
Flow generator fuctionlity provide asa response
store and get performance problem , performace percentage, and levels. solutions
Report Generation Tool
Report Generated

3.1 System Diagram

3.2 Connectivity

NSTRC has a developer platform. When developers are using this platform's GUI interface, they can click a button to generate an intelligence report. When the top component (the NSTRC Developer platform performance analysis tool) identifies the performance problem and the developer requests a view report, report generation will start. At this moment, it will gather data from the top component (performance identifier of different situations) and then calculate the percentage of the performance of a component by comparing different situations. At this moment, the report generation module gets the screenshot of code that has performance issues with line numbers and gets the best solutions for the performance problem, which includes storing the performance problem-related details (problem level) and performance percentage, and screenshots of code parts where the problem occurred in the database report section. Now the flow generator will also start to run. The flow generator will be started to run and create a flow of fixing the performance problem depending on the problem level (low, medium, or major). Finally, the Report Generator tool gathers all the information gathered and produces a well-structured report.

There is an extension for identifying syntax issues and improvements. In our database, we have gathered information about different syntax issues of different programming languages in different situations. Also, there is a different audio language configuration option. It also provides the configuration options to configure the programming language that needs to display syntax issues and improvement steps. The developer can select the audio language and programming language. When the extension is running, it will check the audio language and programming language. When a developer is doing any syntax issues, it will trigger the call to speak and tell the developer what the issue is. The extension is highlighting the code line where the syntax issue happened. It will also display message boxes to display syntax issues and improvements.

3.3 Technologies to be used

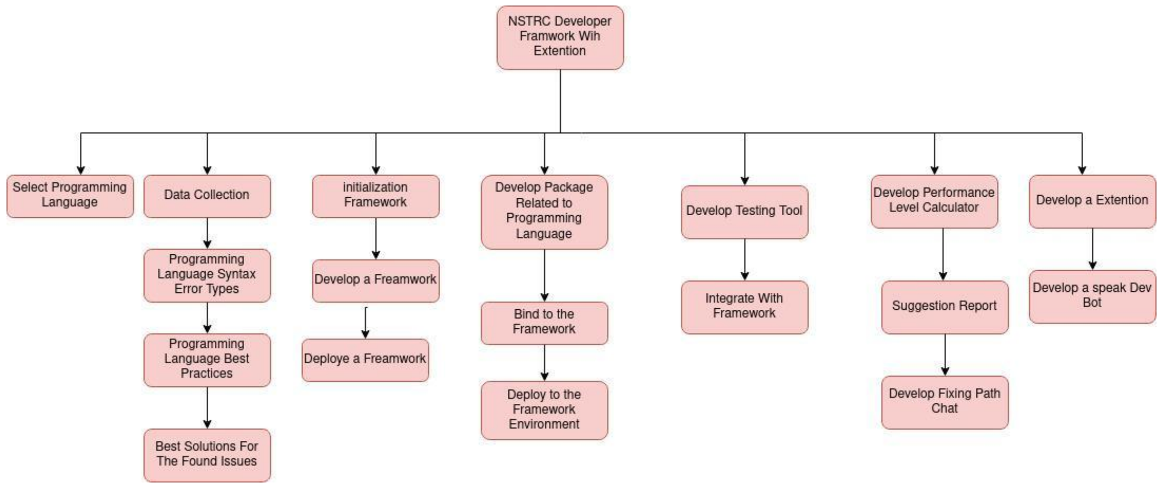
1. JavaScript
2. TypeScript
3. HTML
4. CSS
5. Machine Learning

3.4 Project Requirements

- Report Generation tool should be able to gather and performance as a percentage value and display in in report
- Report generation tool should be able to get the identified problem details and be able to find suitable solution.

- Report generation tool should be able to display performance problems in efficiency way
- Report Generation tool should be able to identify code segments that syntax issue occurs and put it to the report view
- Report generation tool should be able to generate a flow to fix performance problems and display it in report
- Tool can generate a hard copy of report in PDF format
- Visual studio extension should be able to identify syntax issues in an efficient manner
- Extension should be able Speak and tell where the syntax issue occurs
- Extension should also be able to display syntax issue in message box

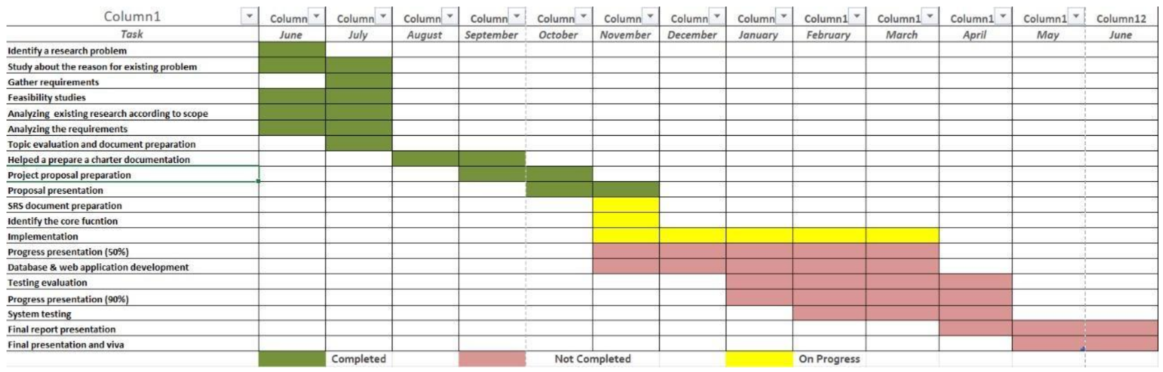
3.5 Work Breakdown Structure



NSTRC Developer Framwork Wih Extention
Select Programming Language
Data Collection
Programming Language Syntax Error Types
Programming Language Best Practices
Best Solutions For The Found Issues
initialization Framework
Develop a Freamwork
Deploye a Freamwork
Develop Package Related to Programming Language
Bind to the Framework
Deploy to the Framework Environment
Develop Testing Tool
Integrate With Framework
Develop Performance Level Calculator
Suggestion Report
Develop Fixing Path Chat
Develop a Extention
Develop a speak Dev Bot

Figure 3.5. 1: Work Breakdown Structure

3.6 Gantt chart



Column1	Column	Column	Column	Column	Column	Column	Column	Column	Column1	Column1	Column1	Column1	Column12
Task	*June*	*July*	*August*	*September*	*October*	*November*	*December*	*January*	*February*	*March*	*April*	*May*	*June*
Identify a research problem	🟩												
Study about the reason for existing problem	🟩	🟩											
Gather requirements		🟩											
Feasibility studies	🟩	🟩											
Analyzing existing research according to scope	🟩	🟩											
Analyzing the requirements	🟩	🟩											
Topic evaluation and document preparation		🟩											
Helped a prepare a charter documentation			🟩	🟩									
Project proposal preparation				🟩	🟩								
Proposal presentation					🟩	🟩							
SRS document preparation						🟨							
Identify the core fucntion						🟨							
Implementation						🟨	🟨	🟨	🟨	🟨			
Progress presentation (50%)						🟥	🟥	🟥	🟥	🟥			
Database & web application development						🟥	🟥	🟥	🟥	🟥			
Testing evaluation								🟥	🟥	🟥	🟥		
Progress presentation (90%)								🟥	🟥	🟥	🟥		
System testing									🟥	🟥	🟥		
Final report presentation											🟥	🟥	🟥
Final presentation and viva												🟥	🟥
	🟩	Completed		🟥	Not Completed			🟨	On Progress				

Figure 3.6. 1: Gantt chart

4. Commercialization

The NSTRC application is going to use different types of users: Mobile application users, desktop application users, and web application developers. The application is to prevent harm to the user device, or the web applications deployed on servers. The application will be released version-wise in the first step of the framework part and will be released for the most popular programming language according to proven statistical information.

In the current times, many user devices are struggling because of installed and invalid compatible third-party apps. Therefore, the NSTRC application will be able to help many users to protect their devices. Currently, there are many intern and trainee developers initiating their career into the technology field. As a result, they will need to be supported to write code in the proper way for proper functioning. Our tool will be able to provide them with the necessary convenience and application in order to carry out the functionalities. When the first release is available, we will make the entire application available for free open source. As time moves forward, new releases with versions will be implemented with better features and trusted resources.

As a desktop mobile application, we need to get closer to the user in order to reach our target. GIGABIT is a short and memorable name, and we designed a logo that is also simple and recognizable. The version is also embedded into the logo to make users aware of the version they are using in and update it whenever a new release has been made.

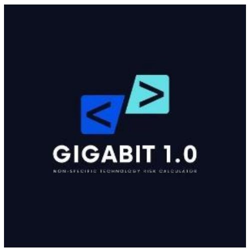


Figure 4.1 NSTRC Logo

5. Reference list

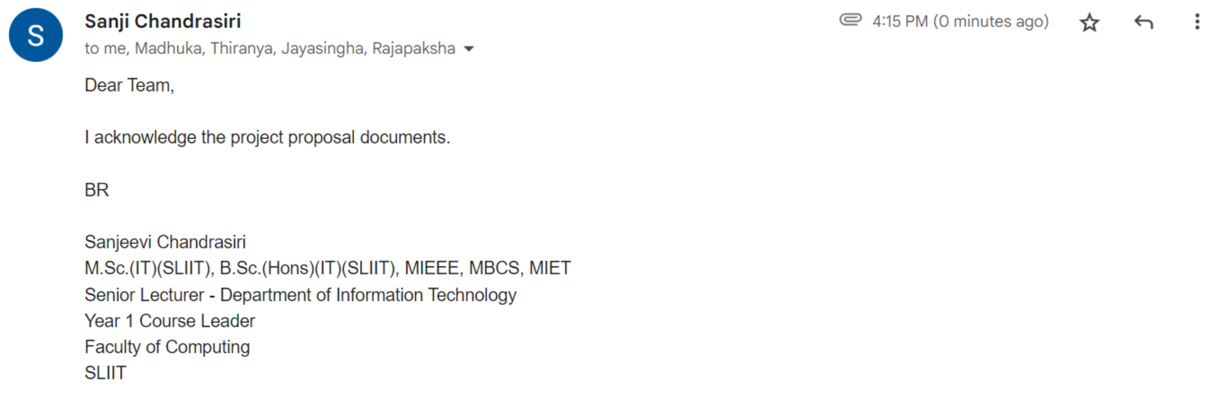
[1] www.c-sharpcorner.com. (n.d.). *4 Website Performance Testing Tools*. [online] Available at: https://www.c-sharpcorner.com/article/top-website-performance-testing-tools/ [Accessed 13 Oct. 2022].
[2] Testpoint. (2017). *The Benefits of Performance Testing*. [online] Available at: https://testpoint.com.au/the-benefits-of-performance-testing/.
[3] Himeur, Y., Ghanem, K., Alsalemi, A., Bensaali, F. and Amira, A. (2021). Artificial intelligence based anomaly detection of energy consumption in buildings: A review, current trends and new perspectives. Applied Energy, [online] 287, p.116601. doi:10.1016/j.apenergy.2021.116601.

[4] Eid, S., Makady, S. and Ismail, M. (2020). Detecting software performance problems using source code analysis techniques. Egyptian Informatics Journal, 21(4), pp.219–229. doi:10.1016/j.eij.2020.02.002.

[5] Hrci Marketing. (n.d.). Top 5 Common Performance Problems. [online] Available at: https://www.hrci.org/community/blogs-and-announcements/hr-leads-business-blog/hr-leads-business/2022/02/14/top-5-common-performance-problems [Accessed 15 Jul. 2022].

6. Appendices

Appendix 1 - Acknowledgement of the supervisor



Sanji Chandrasiri
to me, Madhuka, Thiranya, Jayasingha, Rajapaksha

4:15 PM (0 minutes ago)

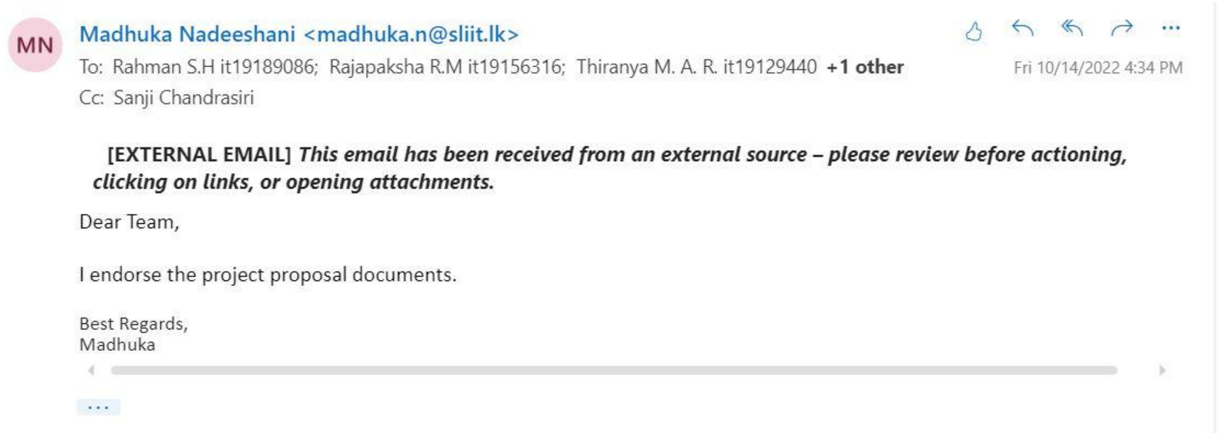
Dear Team,

I acknowledge the project proposal documents.

BR

Sanjeevi Chandrasiri
M.Sc.(IT)(SLIIT), B.Sc.(Hons)(IT)(SLIIT), MIEEE, MBCS, MIET
Senior Lecturer - Department of Information Technology
Year 1 Course Leader
Faculty of Computing
SLIIT

Acknowledgement from Co-Supervisor, Ms. Madhuka Nadeeshani



Madhuka Nadeeshani <madhuka.n@sliit.lk>
To: Rahman S.H it19189086; Rajapaksha R.M it19156316; Thiranya M. A. R. it19129440 **+1 other**
Cc: Sanji Chandrasiri

Fri 10/14/2022 4:34 PM

[EXTERNAL EMAIL] ***This email has been received from an external source – please review before actioning, clicking on links, or opening attachments.***

Dear Team,

I endorse the project proposal documents.

Best Regards,
Madhuka