STAP
Smart Technologies Academic Press

Jordanian Journal of Informatics and Computing

JORDANIAN JOURNAL OF INFORMATICS AND COMPUTING

MUMSPI: A Model for Usability Measurement of Single-Platform Interface for Multi-Tasking in Big Data Tools

Mony Ho[1], Sokroeurn Ang,[1] Sopheaktra Huy,[1] Midhunchakkaravarthy Janarthanan[1]

[1] *School of AI Computing and Multimedia, Lincoln University College, Selangor, Malaysia*

ARTICLE INFO

Article History

Received: 24-09-2025

Vol.2026, No.1

DOI:

Corresponding author. Email: *mho.phdscholar@lincoln.edu.my*

Orcid:

ABSTRACT

This study presents MUMSPI, a model to evaluate the usability of a single-platform interface that supports multi-tasking, compared to command line interface (CLI) in Big Data workflows. Eighty IT participants performed the same tasks using Hadoop, Sqoop, and Python through two interfaces: the Linux Terminal and Jupyter Notebook. Usability was measured across five dimensions such as effectiveness, efficiency, learnability, robustness, and satisfaction. Results show that Jupyter outperformed the Terminal in all areas, with higher task completion (85.18%), faster execution (38.33 minutes), fewer errors (35.12%), and better user satisfaction (SUS score: 70.31%). Overall MUMSPI scores were 74.03% for Jupyter and 45.95% for the Terminal. These results confirm MUMSPI's value and support the use of integrated graphical environments for better usability, especially for users with limited technical skills.

Keywords: Big Data tools usability; MUMSPI model; single-platform interface; Jupyter Notebook; multi-tasking workflows.

How to cite the article

1. Introduction

Big Data means data that is too large, fast, or complex for traditional tools, summarized as volume, velocity, and variety. [1]. Big Data comes from social media, mobile devices, sensors, and scientific instruments, all generating large volumes of real-time data. Apache Hadoop is an open-source framework for storing and processing large data sets across distributed nodes using HDFS and MapReduce. It ensures fault tolerance through data replication and enables efficient, parallel data analysis [2]. Apache Sqoop is a tool used to extract structured data from relational databases and load it into Hadoop systems like HDFS, enabling fast data transfer and reuse in big data analysis [3]. Python is a high-level, open-source language with clean syntax and built-in libraries, ideal for scientific and psychophysics applications [4].

1.1 Problem Statement

Most Big Data tools such as Hadoop, Sqoop, and Python use the command line interface to perform data processing tasks such as importing data, managing directories, and handling file operations [5][6]. Linux terminals operate using the TTY (teletypewriter) system, which allows only one foreground process per session. As Åkesson notes [7], this single-task design limits coordination in complex workflows. Managing separate processes across multiple windows for each task can be fragmented and error-prone, especially in complex Big Data workflows with tools like Hadoop, Sqoop, Python, and so on. Without centralized control, manual coordination reduces efficiency and poses challenges for users with limited technical skills.

1.2 Research Objectives and Proposed Solution

To address this problem associated with single-task workflows in terminal-based Big Data environments, this research introduces MUMSPI (Model for Usability Measurement of Single-Platform Interface), a structured model for measuring the usability of Big Data user interfaces. The model is applied to compare a single-platform interface for working with foreground multiple process per session, represented by Jupyter Notebook, with CLI for working with foreground a process per session, represented by the Linux Terminal, in executing Big Data tools such as Hadoop, Sqoop, and Python. Jupyter Notebook is a free, open source, web-based platform that combines code, output, and narrative in one environment. Popular in data science, it supports multiple languages, remote data access, and integration with HPC and cloud systems for interactive and reproducible workflows [8][9]. Jupyter Notebook supports over 40 languages, real-time execution, and integrated code and documentation, making it ideal for collaboration and education [10]. Barba (2021) highlights its unified interface and broad adoption in academia and industry due to its support for reproducibility and cloud or local execution [11].

This study evaluates usability based on the MUMSPI model, which includes the following five key dimensions:

1. Effectiveness: Measures how accurately and completely users can accomplish tasks, particularly in workflows involving multiple stages or tools.
2. Efficiency: Assesses the time, steps, and effort required to complete tasks, reflecting the system's operational smoothness.
3. Learnability: Evaluates how quickly new users can become proficient, including ease of onboarding and early independence.
4. Robustness: Examines the system's ability to prevent, detect, and recover from user errors during task execution.
5. Satisfaction: Measures users' overall comfort, confidence, and preference, typically assessed using the System Usability Scale (SUS) [12].

This applied research adopts an empirical, experimental, and mixed-methods approach to evaluate the usability of a single-platform interface for Big Data workflows, using the proposed MUMSPI. The study involves 80 participants performed basic analysis tasks using Hadoop, Sqoop, and Python in Linux Terminal and Jupyter. The results provide practical insights into the usability benefits of unified interfaces, emphasizing Jupyter Notebook's ability to simplify complex workflows and reduce the challenges associated with traditional command line-based operations.

1.3 Research Contributions

The key contributions of this research are as follows:

- Proposes the MUMSPI Model, a structured framework for evaluating usability in Big Data environments across five dimensions: effectiveness, efficiency, learnability, robustness, and satisfaction.

- Compares CLI and GUI, using an empirical study with 80 participants to show how Jupyter Notebook, which supports multi-tasking within a single-platform interface, improves usability over the traditional Linux Terminal that handles only one task per session. This advantage enhances the execution of Hadoop, Sqoop, and Python workflows by simplifying coordination and reducing errors.
- Demonstrates Integration Feasibility, showing that Jupyter Notebook can unify Hadoop, Sqoop, and Python into a single-platform, simplifying complex tasks.
- Supports Non-Technical Users, by reducing the learning curve and minimizing errors in multi-tool data analysis.
- Provides Empirical Usability Evidence, reinforcing the value of single-platform interfaces for more efficient and user-friendly Big Data workflows.

2. Literature Review

2.1 Challenges of CLI in Big Data Tools

- Manual Complexity and Errors: CLI workflows in Big Data often require running multiple tasks manually, increasing complexity and error risk. Even within single terminal sessions on platforms like AWS, VirtualBox, and HDP Sandbox, the lack of a unified interface challenges users, especially those less familiar with the CLI [13].
- Disconnected Data Aggregation: Combining internal data with external sources like market trends or social media is valuable but requires manual steps in command line environments. Without an integrated interface, this process is harder to manage and more error-prone [14].
- Uncoordinated Data Transfers: Transferring large data across sensors, storage, cloud, and HPC systems is challenging in command line environments. Users must manually handle transfers, compression, and preprocessing, making the process slower and more complex without automation [15].
- No Workflow Awareness: Linux Terminal offer no visibility into overall workflows. Tasks must be run step by step without system-level feedback, as seen in the CnW heading model on Ubuntu, increasing the risk of errors in complex processes [16].
- Scripting Dependency and Lack of User-Friendly Support: Tools like PsyToolkit show the power of CLI scripting for scientific tasks but also reveal usability issues. It requires command line compilation, manual script edits, and integration with tools like R, making workflows fragmented and challenging for novice users in Big Data settings [17].
- Lack of Usability Evaluation in System Level Studies: Ali et al. (2018) compared Big Data frameworks like Hadoop, Spark, and Flink in terms of scalability and performance but overlooked user interaction at the interface level. This highlights a broader gap in system-level research, which often neglects usability and user-centered design in Big Data tools [18].

2.2 Review of Single-Platform Interfaces for Multi-Tasking

- CloudDOE is a Java-based tool that simplifies Hadoop deployment and job execution via a graphical interface, making it accessible to non-technical users. It supports bioinformatics tools, hybrid clouds, and reduces errors through centralized control [19].
- Cloudera offers a GUI for managing Hadoop tasks like MapReduce and Spark without direct node interaction. It integrates key components for easier workflow coordination and includes open-source tools for data management and security [20].
- CMS employs containerized microservices to support scalable and parallel machine learning workflows. In this system, trainers build models, lightweight prediction services handle inference, and an orchestrator manages updates, functioning similarly to Cloudera's integrated Big Data workflows [21].
- hCoCena runs all analysis steps in a single RStudio Docker session, enhancing usability and reproducibility by avoiding tool switching during multitask workflows [22].
- Jupyter Notebook integrates code, data, and narrative in one platform, easing statistical learning and collaboration. It replaces complex setups with an interactive, user-friendly interface for data analysis [23].

2.3 Jupyter as the Proposed Single-Platform Interface

Jupyter Notebooks support an interactive "write–evaluate–think" cycle, where users write code, see results instantly, and refine their analysis. This enables exploratory thinking and promotes deeper understanding. By combining code, data, visualizations, and explanations in one document, Jupyter makes workflows more reproducible, collaborative, and easier to communicate, offering a more human-centered alternative to traditional command-line tools [24]. Jupyter Notebooks provide a modular, browser-based interface that allows users to integrate code, data, and narrative text in a single document. Jupyter Notebooks offer a modular, browser-based interface that combines code, data, and narrative, making them ideal for complex, reproducible workflows in education and data-intensive fields like Earth Observation [25]. NGLview extends this by enabling interactive molecular visualization within notebooks, supporting exploratory analysis through fast WebGL rendering and a customizable interface [26]. Clarke et al. (2021) introduced Appyters, which turn Jupyter Notebooks into web apps with form-based inputs, enabling non-programmers to run complex workflows. This improves usability, sharing, and reproducibility via cloud execution and permanent URLs [27]. Renz and Hilbig (2023) highlight Jupyter as a flexible tool for digital assessment, combining code, media, and feedback to support personalized learning [28].

2.4 Usability Models and Evaluation Frameworks

Although usability research in data science has grown, there is limited evaluation of multi-tool, multi-language platforms like those in Big Data workflows. Existing models offer useful foundations but often fall short for integrated environments such as Jupyter Notebook. Key frameworks include Nielsen's Heuristics, which emphasize learnability, efficiency, and satisfaction [29]; ISO 9241-11, which defines usability by effectiveness, efficiency, and user satisfaction in context [30]; and the System Usability Scale (SUS), a validated 10-item questionnaire for quick usability assessment [31]. These models have been applied to evaluate tools like Jupyter in educational and analytical settings [28].

These models commonly assess five key dimensions:

- Effectiveness: whether users can successfully complete their tasks [30]
- Efficiency: how quickly and with how little effort tasks can be completed [30]
- Learnability: how easily new users can understand and use the system [29]
- Robustness: how well the system prevents, detects, and helps users recover from errors [29]
- Satisfaction: the users' overall comfort and satisfaction while using the tool [31]

MUMSPI builds on ISO 9241 11 [30], Nielsen's heuristics [29], and the System Usability Scale [31], extending them to complex Big Data environments. Unlike traditional models for single tools, MUMSPI evaluates integrated platforms like Jupyter, addressing challenges in multi tool coordination, language diversity, and task complexity.

2.5 Research Gap

Most studies on Big Data frameworks focus on performance and architecture, with little attention to usability across interface types. No empirical comparison has evaluated command-line setups like the Linux Terminal versus single-platform like Jupyter Notebook. Existing usability models are also not designed for multi tool environments common in data analytics. Specific gaps identified include:

- Absence of usability models designed to evaluate unified interface versus multiple terminal interfaces in Big Data workflows
- Lack of comprehensive studies comparing the Linux Terminal to Jupyter Notebook using a structured usability model across tools such as Hadoop, Sqoop and Python
- Limited empirical evidence measuring user experience across CLI and GUI in integrated analytics environments

3. Research Methodology

3.1 Overview the MUMSPI Model

To address these gaps, this study introduces MUMSPI, a model for evaluating Big Data tool usability across interfaces. It focuses on five dimensions and compares command-line environments with single-platform like Jupyter Notebook. Figure 1 illustrates the structure of the proposed MUMSPI model, which is designed to evaluate the usability of Big Data tools across different interface environments. The model is composed of three key components:

- MUMSPI Core Model: This component defines five key usability dimensions: effectiveness, efficiency, learnability, robustness, and satisfaction. These dimensions guide the criteria used in the evaluation process.
- Evaluation Process: The model is empirically applied to compare usability between two interface types: command line interfaces (Linux Terminal) and graphical platforms (Jupyter Notebook). The study involves 80 participants, each performing basic data analysis using Jupyter to execute Hadoop, Sqoop, and Python command codes.

- Output: The results of the evaluation reveal practical findings and impact, including improved usability in graphical interfaces, greater accessibility for non-technical users, more integrated workflows, and empirical validation of the MUMSPI model.

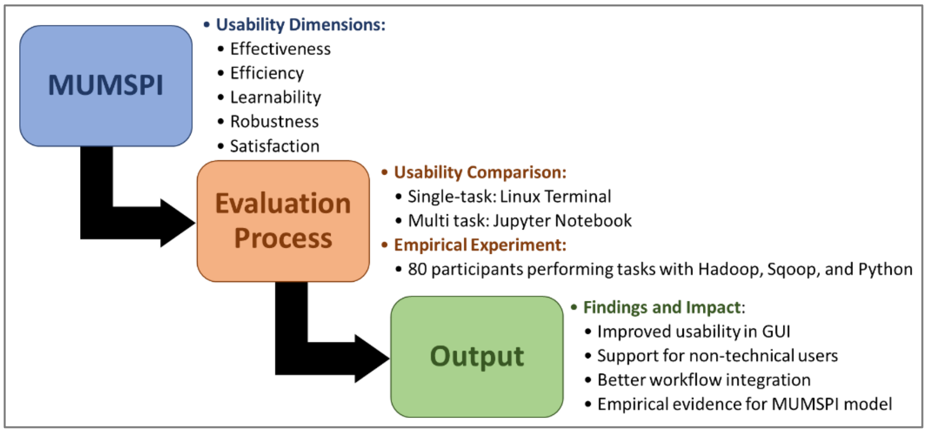
MUMSPI
• Usability Dimensions:
• Effectiveness
• Efficiency
• Learnability
• Robustness
• Satisfaction
Evaluation Process
• Usability Comparison:
• Single-task: Linux Terminal
• Multi task: Jupyter Notebook
• Empirical Experiment:
• 80 participants performing tasks with Hadoop, Sqoop, and Python
Output
• Findings and Impact:
• Improved usability in GUI
• Support for non-technical users
• Better workflow integration
• Empirical evidence for MUMSPI model

Figure 1. MUMSPI Model: A usability evaluation framework for comparing CLI and GUI-based Big Data platforms.

The MUMSPI evaluates the usability of Big Data tools based on five key dimensions. Each dimension is assessed through participant feedbacks during and after 2 practices. Usability performance is computed using a normalized scale (0 to 1) for each dimension, and an overall MUMSPI score is calculated using the following MUMSPI formula:

$$\text{MUMSPI} = \frac{ES + EY + LY + RS + SN}{5} \times 100 \qquad (1)$$

Where:

- ES = EffectivenesS score
- EY = EfficiencY score
- LY = LearnabilitY score
- RS = RobustnesS score
- SN = SatisfactioN score

3.2 Research Approaches

This study is classified as applied research and adopts a combination of three research approaches to ensure a comprehensive usability evaluation using the MUMSPI model:

- Empirical Research [32]: Usability data were collected from 80 participants executing Hadoop, Sqoop, and Python command codes via either the Linux Terminal and Jupyter Notebook.
- Experimental Research [33]: A controlled experiment assigned users to either a CLI or GUI condition.
- Mixed-Methods Research [34]: Combines quantitative task scores and qualitative feedback to assess usability using the MUMSPI model, capturing both performance and user experience.

3.3 Participant Selection

To maintain statistical reliability for the usability evaluation under the MUMSPI model, the Yamane formula [35] was applied with a 5% margin of error, resulting in a required sample size of 80 participants from an initial population of 100 IT practitioners. Participants were expected to perform basic Big Data analysis tasks using both via the Linux Terminal and Jupyter, involving tools such as Hadoop, Sqoop, and Python. This technical background ensured that participants could meaningfully engage in tasks required for assessing usability across the five MUMSPI dimensions. GitHub served as the central public repository for storing the practice manuals and satisfaction survey. The repository, available at https://github.com/datasciencesource/jupyter, ensures open access to the study's resources. The repository contains the following components:

- "Instruction 1 - Big Data Analytics Via Terminal in Ubuntu Linux.pdf": How to perform Big Data tasks using the Terminal interface in Ubuntu Linux.
- "Instruction 2 - Big Data Analytics Via Jupyter in Ubuntu Linux.pdf": How to perform the same Big Data tasks using the Jupyter Notebook.

- MUMSPI_Usability_Survey_Form.pdf: A form to evaluate and compare usability of the two methods (Terminal vs Jupyter) based on user experience.
- Datasource.sav: An SPSS file that contains data collected from 80 respondents.

3.4 Environment Setup

The host and virtual machines were configured as specified in Tables 1 and 2 to support the measurement.

Table 1. Minimum system requirements

Component	Specification
Processor (CPU)	Intel Core i5 or equivalent
Memory (RAM)	8 GB
Storage	100 GB available space
Operating system	Microsoft Windows 10 Pro
Virtualization platform	VirtualBox

Table 2. Minimum virtual machine requirements

Configuration Parameter	Minimum Specification
Operating system	Ubuntu Linux 24.04
CPU allocation	2 CPU Cores
RAM allocation	4 GB
Storage allocation	40 GB disk space
Installation mode	GUI mode (required)
Snapshot management	System state backup and restoration
Apache Hadoop	Version 2.7.3
Apache Sqoop	Version 1.4.7
Python	Version 3.12.3
MariaDB	Version 15.1

3.5 Prerequisites and Expected Outcomes

Table 3 compares the prerequisites and expected outcomes for performing Big Data tasks using the Terminal versus Jupyter, highlighting similarities in requirements but differences in tools and learning environments.

Table 3. Prerequisites and expected outcomes for CLI versus GUI

Category	Using Terminal (First Instruction)	Using Jupyter (Second Instruction)
Knowledge required	Understanding Linux command-line concepts	Understanding Linux command-line concepts
Skills required	Ability to type and execute commands Big Data tools	Ability to type and execute commands Big Data tools
Learning material	Instruction 1 - Big Data Analytics Via Terminal in Ubuntu Linux.pdf	Instruction 2 - Big Data Analytics Via Jupyter in Ubuntu Linux.pdf
Internet access	Required	Required
Tool used	Terminal	Jupyter
Facilitator assistance	Required	Required
Evaluation method	Big Data Analytics Via Terminal in Ubuntu Linux	Big Data Analytics Via Jupyter in Ubuntu Linux
Time required	60 minutes	60 minutes
Expected outcome	Demonstrate proficiency in using Big Data commands in Terminal	Demonstrate proficiency in using Big Data commands in Jupyter

3.6 Data Collection Process and Analysis

Data were collected from 80 IT practitioners who completed Big Data tasks using two interfaces: the Linux Terminal (Instruction 1) and Jupyter Notebook (Instruction 2). Tasks involved running Hadoop, importing data via Sqoop, cleaning and analyzing data with Python. Participants followed predefined steps in a VirtualBox Ubuntu environment, with 60 minutes allocated per task. Usability was assessed using the MUMSPI Usability Survey Form, covering five dimensions.

Quantitative data included task time, error counts, and System Usability Scale (SUS) scores. Qualitative feedback was gathered through open-ended survey responses. Data were analyzed using normalized scoring and statistical tests.

4. Analysis and Findings

4.1 Quantitative Comparison

(A) Effectiveness Comparison

Figure 2 illustrates the comparison between Terminal and figure 3, Jupyter users in response to the question: "Q1.1 Did you complete the main task?" Among Terminal users, only 2.5% reported full task completion, 82.5% completed the task partially, and 15% could not complete it. In contrast, Jupyter users showed stronger task performance, with 37.5% fully completing the task, 42.5% partially completing it, and 20% failing to complete the task. These findings show that while most Terminal users were only able to partially complete the task, the Jupyter interface enabled a greater percentage of users to fully complete the task. This suggests higher effectiveness in Jupyter's guided, integrated environment, where inline outputs and modular workflows may have contributed to improved task clarity and execution.

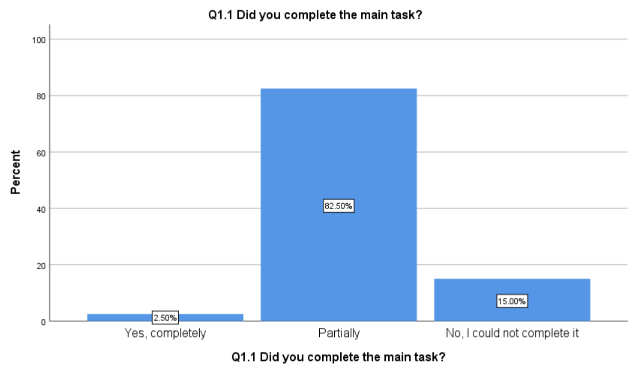

Figure 2. Responses from participants using the Terminal.

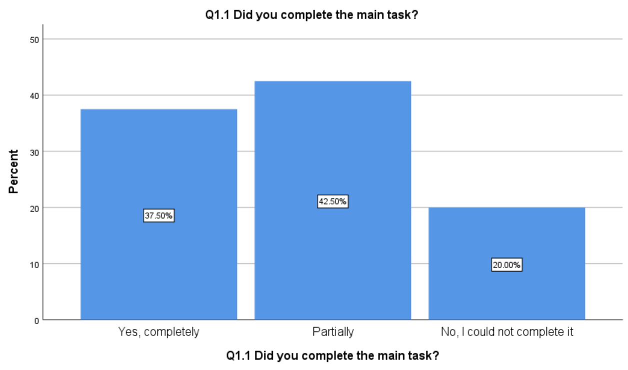

Figure 3. Responses from participants using the Jupyter.

Table 4 shows task completion estimates. Terminal users had a mean of 72.53% (SD = 21.11, range = 88.0), indicating varied performance. Jupyter users showed a higher mean of 85.18% (SD = 8.85, range = 26.0), reflecting more consistent and effective task execution. These results align with Q1.1, confirming Jupyter's advantage in usability.

Table 4. Descriptive statistics for estimated task completion percentage (Q1.2) across Terminal and Jupyter interfaces

Interfaces	N	Range	Minimum	Maximum	Mean	Std. Deviation
Terminal	40	88.0	12.0	100.0	72.525	21.1138
Jupyter	40	26.0	74.0	100.0	85.175	8.8459

(B) Efficiency Comparison

Table 5. Descriptive statistics for time to complete the task (Q2.1) across Terminal and Jupyter interfaces

Interfaces	N	Range	Minimum	Maximum	Mean	Std. Deviation
Terminal	40	30.0	31.0	61.0	49.725	7.1432
Jupyter	40	27.0	28.0	55.0	38.325	7.4949

Table 5 shows task completion times. Terminal users averaged 49.73 minutes (SD = 7.14), while Jupyter users were faster at 38.33 minutes (SD = 7.49). With similar variability, these results suggest that Jupyter's integrated interface improves task efficiency over the command line in Big Data workflows.

Table 6 compares interaction counts. Terminal users averaged 31.38 commands (SD = 3.08), while Jupyter users needed fewer steps, averaging 23.73 clicks (SD = 2.50). This suggests Jupyter's structured, cell-based design simplifies task execution, contributing to faster completion and fewer errors than the command line.

Table 6. Descriptive statistics for number of commands or clicks used (Q2.2) across Terminal and Jupyter interfaces

Interfaces	N	Range	Minimum	Maximum	Mean	Std. Deviation
Terminal	40	12.0	24.0	36.0	31.375	3.0775
Jupyter	40	8.0	20.0	28.0	23.725	2.5012

Figures 4 and 5 show responses to "Q2.3 How difficult was the task?" Only 2.5% of Terminal users rated it Easy, while most found it Neutral to Very Difficult. In contrast, 65% of Jupyter users rated the task Easy or Very Easy. These results highlight Jupyter's user-friendly design, which reduces difficulty and supports users with limited command line experience.

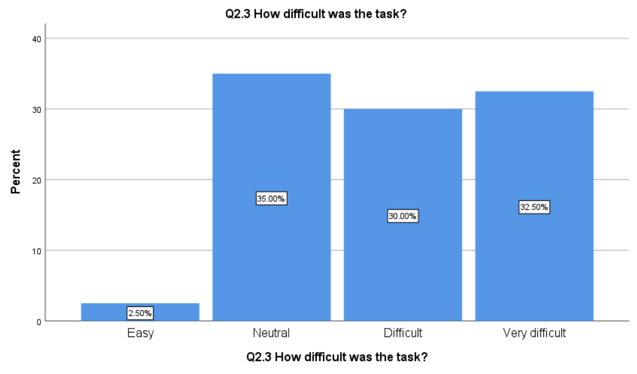

Figure 4. Task Difficulty Using Terminal.

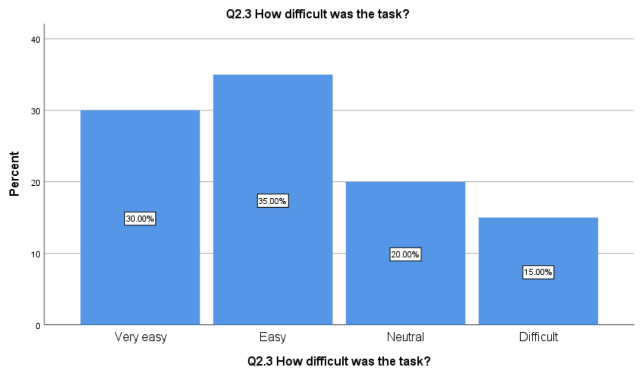

Figure 5. Task Difficulty Using Jupyter.

(C) Learnability Comparison

Table 7 shows first attempt completion times. Terminal users averaged 18.48 minutes (SD = 2.63), while Jupyter users were faster at 13.95 minutes (SD = 2.72). This suggests quicker adaptation to Jupyter's structured interface, supporting faster and more efficient initial task performance.

Table 7. Time to Complete First Task Attempt (Q3.1) Across Terminal and Jupyter Interfaces

Interfaces	N	Range	Minimum	Maximum	Mean	Std. Deviation
Terminal	40	11.0	12.0	23.0	18.475	2.6311
Jupyter	40	10.0	10.0	20.0	13.950	2.7170

Figures 6 and 7 show responses to "Q3.3 Did you need help to complete the task?" Most Terminal users (72.5%) needed a lot of help, while only 10% completed the task independently. In contrast, Jupyter users showed greater independence, with 27.5% completing the task on their own. These results suggest Jupyter's structured, interactive interface improves learnability and reduces reliance on assistance.

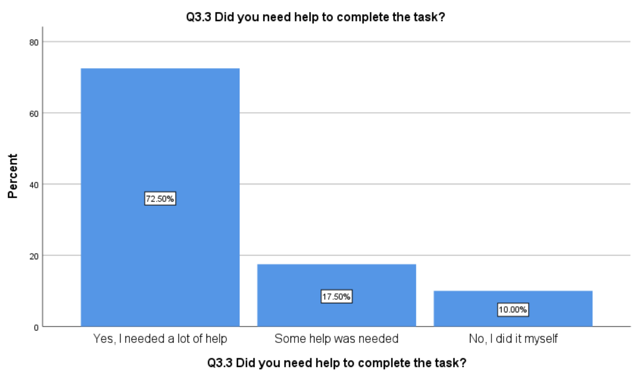

Figure 6. Help Required to Complete the Task Using Terminal.

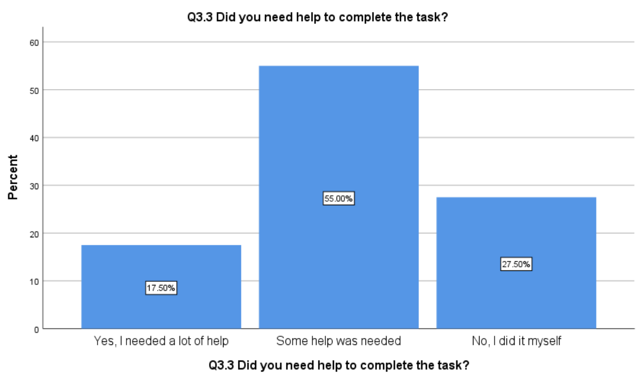

Figure 7. Help Required to Complete the Task Using Jupyter.

(D) Robustness Comparison

Table 8 shows error rates during task completion. Terminal users had a higher mean error rate of 45.85% (SD = 21.06), while Jupyter users averaged 35.13% (SD = 14.87). The lower and more consistent error rates in Jupyter suggest its structured interface and feedback help users avoid and correct mistakes more effectively.

Table 8. Percentage of Errors Encountered During the Task (Q4.1) Across Terminal and Jupyter Interfaces

Interfaces	N	Range	Minimum	Maximum	Mean	Std. Deviation
Terminal	40	64.0	12.0	76.0	45.850	21.0574
Jupyter	40	46.0	8.0	54.0	35.125	14.8707

Figures 8 and 9 show responses to "Q4.2 Were you able to fix or recover from the errors?" Only 17.5% of Terminal users fully recovered, compared to 22.5% of Jupyter users. Both groups had 52.5% partial recovery, but Jupyter users had a lower failure rate. These results suggest Jupyter's structured, feedback-rich interface supports better error recovery.

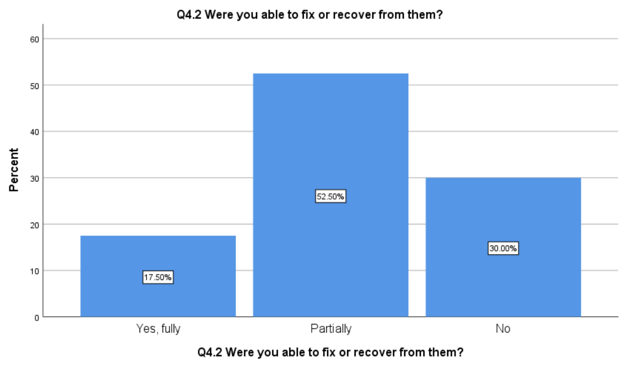

Figure 8. Ability to Recover from Errors Using Terminal.

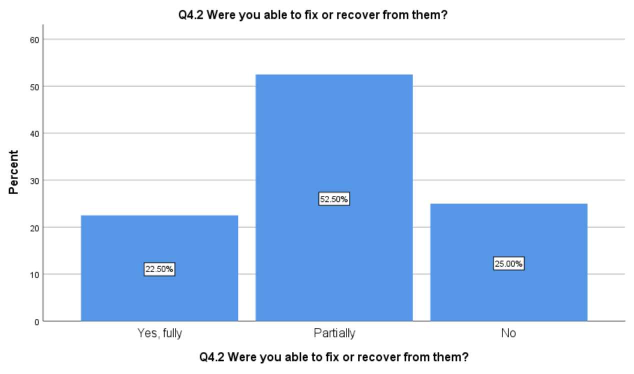

Figure 9. Ability to Recover from Errors Using Jupyter.

Figures 10 and 11 show responses to "Q4.3 How clear was the system's feedback?" Terminal users rated feedback poorly, with 65% disagreeing or strongly disagreeing. In contrast, most Jupyter users were Neutral (75%), with 17.5% agreeing it was clear. These results suggest Jupyter offers clearer, more interpretable feedback than the Terminal interface.

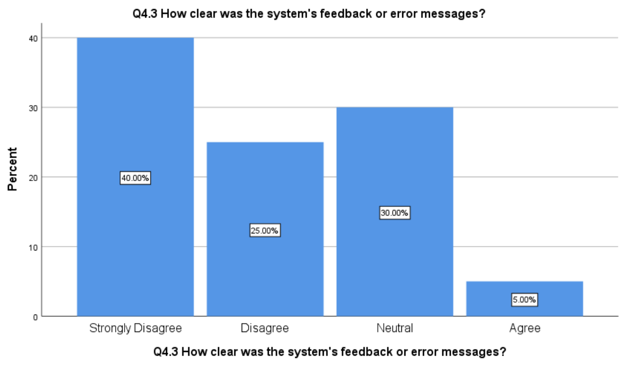

Figure 10. Clarity of System Feedback or Error Messages Using Terminal.

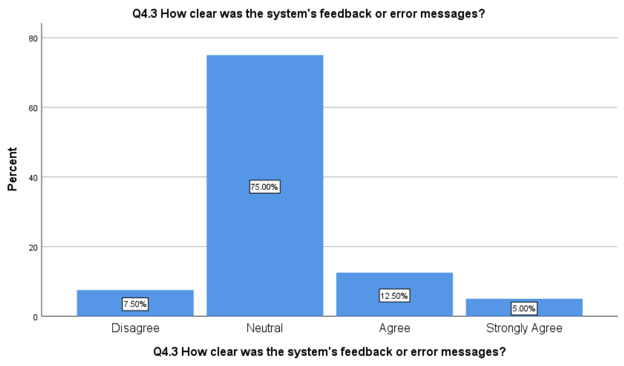

Figure 11. Clarity of System Feedback or Error Messages Using Jupyter.

(E) Satisfaction Comparison

Table 9 shows that Jupyter users reported significantly higher satisfaction than Terminal users. Specifically, 29 Jupyter users agreed or strongly agreed they would like to use the system frequently (Q5.1a), compared to only 2 Terminal users. While most Terminal users rated the system as complex and hard to use (Q5.1b, Q5.1h), Jupyter users consistently reported ease of use, better integration, and confidence while interacting with the system.

The SUS scores further highlight this contrast:

- Jupyter scored 70.31, reflecting marginally acceptable usability,
- Terminal scored only 30.00, which falls well below usability thresholds [31].

These results are based on standardized item scoring, with positively worded items (Q5.1a, c, e, g, i) and negatively worded items (Q5.1b, d, f, h, j) adjusted before summation, following Brooke's SUS methodology [31].

Table 9. System Usability Scale Responses (Q5.1) Comparing Terminal and Jupyter Interfaces

System Usability Scale & Feedback * 40 Responses on Terminal (T.) * 40 Responses on Jupyter (J.)	Strongly Disagree		Dis-agree		Neutral		Agree		Strongly Agree		Adjusted Total	
	T.	J.	T.	J.	T.	J.	T.	J.	T.	J.	T.	J.
a. I think that I would like to use this system frequently.	18	0	18	4	2	7	2	25	0	4	23	109
b. I found the system unnecessarily complex.	0	5	2	26	4	6	18	2	16	1	58	112
c. I thought the system was easy to use.	17	0	18	4	2	5	3	29	0	2	20	109
d. I think I would need the support to use this system.	0	15	3	16	9	9	21	0	7	0	49	109
e. I found the functions well integrated.	8	0	17	5	14	6	1	19	0	10	60	114
f. I thought there was too much inconsistency.	0	7	1	15	8	16	16	2	15	0	54	111
g. Most people would learn this system quickly.	7	0	18	1	14	10	1	26	0	3	31	118
h. I found it very cumbersome to use.	0	14	1	19	9	7	19	0	11	0	55	109
i. I felt confident using the system.	5	3	18	2	16	17	1	14	0	4	32	113
j. I need to learn a lot before following the instruction.	0	9	2	18	12	13	19	0	7	0	52	114

4.2 Qualitative Feedbacks

To complement the quantitative results, qualitative feedback was collected from five open-ended questions (Q1.3, Q2.5, Q3.4, Q4.4, and Q5.2), with responses provided by all 80 participants. The first 40 responses were from users who performed tasks using the Terminal, and the remaining 40 were from those who used the Jupyter. Responses were analyzed using thematic coding, identifying recurring themes reflecting user challenges, learning behaviors, and task perception.

(A) Terminal Interface

Participants using the Terminal interface frequently expressed difficulties related to technical execution, learning curve, and system feedback. These patterns highlight the complexity and fragility of CLI-based workflows, especially for users unfamiliar with sequential commands and error handling in a terminal environment.

Table 10. Thematic Coding of Terminal User Feedback (Q1.3, Q2.5, Q3.4, Q4.4, Q5.2)

Theme	Description	Frequency (out of 40)
Data handling	Issues with file imports, HDFS commands, or path navigation.	8
Output confusion	Difficulty interpreting command-line output or system response.	6
Learning curve	Needing help, struggling with commands, or trial-and-error behavior.	7
Error recovery	Inability to correct mistakes without restarting.	4
Other	General frustrations or vague complaints not tied to specific features.	5

(B) Jupyter Notebook Interface

Jupyter users reported a more seamless and user-friendly experience, highlighting clarity, integration, and confidence. In contrast, the Terminal interface caused more confusion and errors due to higher cognitive load. These findings support Jupyter's usability strengths and validate the MUMSPI model's focus on effectiveness, efficiency, and learnability.

Table 11. Thematic Coding of Jupyter User Feedback (Q1.3, Q2.5, Q3.4, Q4.4, Q5.2)

Theme	Description	Frequency (out of 40)
Clarity and feedback	Clear output presentation and inline visual feedback.	10
Ease of use	Smooth navigation, minimal learning curve.	9
Reduced errors	Fewer mistakes and easier recovery due to code cells and structured layout.	6
Task continuity	Ability to follow tasks from top to bottom without losing context.	7
Other	Praise for the open format or suggestions for minor improvements.	8

4.3 Comparative Discussion

A comparative analysis of the five MUMSPI dimensions in table 12 shows that Jupyter consistently outperforms the Terminal interface across all usability metrics:

- Effectiveness: Jupyter users achieved a higher task completion rate (85.18%) compared to Terminal users (72.53%).
- Efficiency: Jupyter users completed tasks faster (38.33 minutes) than Terminal users (49.72 minutes).
- Learnability: Fewer interactions were needed in Jupyter (13.95) than in Terminal (18.48), suggesting easier learning.
- Robustness: Jupyter users encountered fewer errors (35.12%) than Terminal users (45.85%).

- Satisfaction: Jupyter scored significantly higher on the SUS scale (70.31) than Terminal (30.00), indicating better overall user satisfaction.

Table 12. Thematic Coding of Terminal User Feedback (Q5.2)

Usability Dimension	Associated Question	Terminal Score	Jupyter Score
Effectiveness (ES)	Q1.2 (% Completion)	72.53	85.18
Efficiency (EY)	Q2.1 (Time in min)	49.72	38.33
Learnability (LY)	Q3.1 (Clicks/Steps)	18.48	13.95
Robustness (RS)	Q4.1 (% Errors)	45.85	35.12
Satisfaction (SN)	Q5.1 (SUS Score)	30.00	70.31

4.4 Usability Score Comparison Using the MUMSPI Model

The MUMSPI model provides an overall usability score by averaging normalized values across five dimensions. Each score was normalized on a 0 to 1 scale, with higher values indicating better usability. Using the formula (1) and Inverse Min-Max formula [36] of normalized scale in Section 3.1, Jupyter achieved a MUMSPI score of 74.03%, outperforming Terminal's 45.95%. These results indicated the superior usability of single-platform interface like Jupyter, particularly for non-technical users in Big Data tasks.

Table 13. Usability Score Comparison Using the MUMSPI Model

Interface	ES	EY	LY	RS	SN	MUMSPI Score
Formula	Direct % from Q1.2/100	Inverse Min-Max from Q2.1	Inverse Min-Max from Q3.1	Inverse Min-Max from Q4.1	SUS ÷ 100 (Q5.1)	Formula (1)
Terminal	0.7253	0.3427	0.5760	0.3537	0.3000	**45.95%**
Jupyter	0.8518	0.7223	0.8025	0.6220	0.7031	**74.03%**

5. Conclusion

The results show that single-platform interface as Jupyter Notebook significantly outperforms the Terminal interface. Jupyter users achieved higher task completion (85.18%), faster execution (38.33 minutes), fewer errors (35.13%), and greater satisfaction (SUS score: 70.31%). In contrast, Terminal users performed less effectively across all dimensions. The MUMSPI score for Jupyter was 74.03%, compared to 45.95% for the Terminal. These findings discover the usability benefits of graphical, single-platform interfaces like Jupyter, particularly for nontechnical users. The MUMSPI model effectively captured both performance and perception, supporting its value for evaluating usability in Big Data environments. Jupyter Notebook is therefore recommended as the preferred interface for educational and analytical workflows. Future research may extend this work by evaluating hybrid or adaptive interfaces and including security or performance as additional usability dimensions.

Conflicts Of Interest

The author declares no conflicts of interest.

Funding

This research received no external funding.

Acknowledgment

The authors thank their co-authors for guidance, peer reviewers for valuable feedback, and all participants for their contributions. They also acknowledge the support of the Lincoln University College Doctoral Program.

References

[1] Madden, S. (2012). From databases to big data. *IEEE Internet Computing, 16*(3), 1-2. https://doi.org/10.1109/MIC.2012.50

[2] Hannan, S. A. (2016). An overview on Big Data and Hadoop. *International Journal of Computer Applications, 154*(10), 31-32. https://www.researchgate.net/publication/372631347

[3] Xu, Z., Shi, D., & Tu, Z. (2021). Research on diagnostic information of smart medical care based on big data. *Journal of Healthcare Engineering, 2021*, Article 9977358. https://doi.org/10.1155/2021/9977358

[4] Peirce, J. W. (2007). PsychoPy—Psychophysics software in Python. *Journal of Neuroscience Methods*, *162*(1–2), 2-3. https://doi.org/10.1016/j.jneumeth.2006.11.017

[5] Reddy, S. S., & Dhanalakshmi, S. (2024). Cleaning big data – An interactive approach with Sqoop and Hive. *INNOVAR Journal*, *17*(1), 3-4.

[6] Soibam, B., & Roman, G. (2024). PySmooth: A Python tool for the removal and correction of genotyping errors. *BMC Research Notes, 17*, Article 103. https://doi.org/10.1186/s13104-024-06753-4

[7] Åkesson, L. (n.d.). *The TTY demystified*. Linus Åkesson. https://www.linusakesson.net/programming/tty/

[8] Perkel, J. M. (2018, November 1). By Jupyter, it all makes sense. *Nature*, *563*(7729), 1–2. https://doi.org/10.1038/d41586-018-07196-1

[9] Thomas, R., & Cholia, S. (2021). Interactive supercomputing with Jupyter. *Computing in Science & Engineering, 23*(2), 1-3. https://doi.org/10.1109/MCSE.2021.3059037

[10] Rule, A., Birmingham, A., & Horne, B. (2015). Notebook: Visualizing and sharing computational analyses. *ACM SIGOPS Operating Systems Review*, *49*(1), 1–2. https://doi.org/10.1145/2723872.2723885

[11] Xu, Y. (2021). Research on diagnostic information of smart medical care based on big data. *Journal of Healthcare Engineering*, *2021*, Article ID 6640870. https://doi.org/10.1155/2021/6640870

[12] Wang, Q., Lee, Y., & Chong, A. Y. L. (2020). The use of Jupyter notebooks in education: A systematic review. *Computers in Human Behavior*, *115*, 106621. https://doi.org/10.1016/j.chb.2020.106621

[13] Pimentel, J. F., Murta, L., Braganholo, V., & Freire, J. (2019). Jupyter notebooks as a tool for open science: An empirical study. *Proceedings of the 2019 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, 12–14. https://doi.org/10.1109/JCDL.2019.00012

[14] Barba, L. A. (2021). The Python/Jupyter ecosystem: Today's problem-solving environment for computational science. *Computing in Science & Engineering, 23*(3), 434. https://doi.org/10.1109/MCSE.2021.3074693

[15] Pérez, F., & Granger, B. E. (2015). An introduction to Jupyter. *O'Reilly Media*. https://conferences.oreilly.com/jupyter/jupytercon

[16] Vista, F. P., IV, & Chong, K. T. (2019). Design and real-time implementation of a 3-stage CnW heading system on an Ubuntu Linux-embedded board. *Journal of Sensors, 2019*, Article ID 1345202. https://doi.org/10.1155/2019/1345202

[17] Stoet, G. (2010). PsyToolkit: A software package for programming psychological experiments using Linux. *Behavior Research Methods, 42*(4), 1096–1100. https://doi.org/10.3758/BRM.42.4.1096

[18] Ali, M. I., Qadir, J., Rasool, R. U., Sathiaseelan, A., Zwitter, A., & Crowcroft, J. (2018). Big data-driven smart transportation: A review. *IEEE Access, 6*, 42211–42258. https://doi.org/10.1109/ACCESS.2018.2868771

[19] Chung, W.-C., Chen, C.-C., Ho, J.-M., Lin, C.-Y., Hsu, W.-L., Wang, Y.-C., Lee, D. T., Lai, F., Huang, C.-W., & Chang, Y.-J. (2014). CloudDOE: A user-friendly tool for deploying Hadoop clouds and analyzing high-throughput sequencing data with MapReduce. *PLoS ONE, 9*(6), e98146. https://doi.org/10.1371/journal.pone.0098146

[20] González-García, A., de Amescua, A., & García-Holgado, A. (2023). Design and implementation of a graphical interface for command line-based data processing pipelines. *Applied Sciences, 13*(3), 1183. https://doi.org/10.3390/app13031183

[21] Li, K., & Gui, N. (2020). CMS: A continuous machine-learning and serving platform for industrial big data. *Future Internet, 12*(6), 102. https://doi.org/10.3390/fi12060102

[22] Oestreich, M., Holsten, L., Agrawal, S., Dahm, K., Koch, P., Jin, H., Becker, M., & Ulas, T. (2024). hCoCena: A toolbox for network-based co-expression analysis and horizontal integration of transcriptomic datasets. *Software Impacts, 19*, 100663. https://doi.org/10.1016/j.simpa.2024.100663

[23] Kumwichar, P. (2023). Enhancing learning about epidemiological data analysis using R for graduate students in medical fields with Jupyter Notebook: Classroom action research. *JMIR Medical Education, 9*, e47394. https://doi.org/10.2196/47394

[24] Granger, B. E., & Pérez, F. (2021). Jupyter: Thinking and storytelling with code and data. *Computing in Science & Engineering, 23*(2), 1–2. https://doi.org/10.1109/MCSE.2021.3059263

[25] Cheng, S., Guo, H., Li, S., & Liu, H. (2022). Integrating cloud computing and Jupyter Notebooks for remote sensing education. *Remote Sensing, 14*(14), 3359. https://doi.org/10.3390/rs14143359

[26] Nguyen, H., Case, D. A., & Rose, A. S. (2018). NGLview—Interactive molecular graphics for Jupyter notebooks. *Bioinformatics, 34*(7), 1241–1242. https://doi.org/10.1093/bioinformatics/btx789

[27] Bussonnier, M., Freeman, J., Haven, T., Head, T., Holdgraf, C., Kelley, K., Nalvarte, G., Osheroff, A., Pacer, M., Panda, Y., Pérez, F., Ragan-Kelley, B., Willing, C., & Grout, J. (2021). JupyterHub: A multi-user server for Jupyter notebooks. *Software Impacts, 9*, 100072. https://doi.org/10.1016/j.simpa.2021.100072

[28] Grewal, M., & Aseri, T. C. (2024). A comparative study of online learning platforms using Jupyter Notebook and Google Colab. *Education and Information Technologies*. Advance online publication. https://doi.org/10.1007/s10639-025-13507-7

[29] Nielsen, J. (1995). *10 usability heuristics for user interface design*. Nielsen Norman Group. https://www.nngroup.com/articles/ten-usability-heuristics/

[30] International Organization for Standardization. (2018). *ISO 9241-11:2018 – Ergonomics of human-system interaction – Part 11: Usability: Definitions and concepts*. https://www.iso.org/standard/63500.html International Organization for Standardization. (2018). *ISO 9241-11:2018 – Ergonomics of human-system interaction – Part 11: Usability: Definitions and concepts*. https://www.iso.org/standard/63500.html

[31] Brooke, J. (1996). SUS: A quick and dirty usability scale. In P. W. Jordan, B. Thomas, B. A. Weerdmeester, & A. L. McClelland (Eds.), *Usability evaluation in industry* (pp. 189–194). London: Taylor & Francis.

[32] George, E. O. (2024, January). *Empirical research: A comprehensive guide for academics*. Paperpal. https://paperpal.com/blog/researcher/empirical-research-a-comprehensive-guide-for-academics

[33] Saha, S. (2024, June). *What is experimental research: Definition, types & examples*. Entropik. https://www.entropik.io/blogs/experimental-research

[34] Johnson, R. B., & Onwuegbuzie, A. J. (2004). Quantitative and qualitative research: A view for clarity. *International Journal of Education*, *22*(3), 371–372. https://doi.org/10.1016/j.ijintrel.2010.07.004

[35] Singh, A. S., & Masuku, M. B. (2014). Sampling techniques & determination of sample size in applied statistics research: An overview. *International Journal of Economics, Commerce and Management*, *2*(11), 78. https://ijecm.co.uk/wp-content/uploads/2014/11/21131.pdf

[36] Mikemclaren. (2016, October 28). *Normalization when max and min values are reversed* [Online forum post]. Stack Exchange. https://stats.stackexchange.com/questions/252455/normalization-when-max-and-min-values-are-reversed

Biographies

Author Mony Ho. Mony Ho is a Ph.D. candidate in Information Technology at Lincoln University College, Malaysia. He holds a Master's degree in IT and Data Science from the European International University, France. He is currently a master trainer at Preah Kossomak Polytechnic Institute and lectures part-time at multiple universities in Cambodia. His teaching and research interests include Data Science, Big Data, software engineering, network engineering and cyber security.

He can be contacted at email: mho.phdscholar@lincoln.edu.my

Author Sokroeurn Ang. Mr. Sokroeurn Ang is a senior lecturer in cybersecurity. He has been teaching ICT and cybersecurity since 2015 and has held various roles in ICT and cybersecurity for over a decade. His professional experience spans the central banking sector, private banking, and internet service providers. He has been actively involved in areas such as cybersecurity risk assessment, IT governance, network security, web application security, cybersecurity incident response, BCP and DRP, cloud security, VAPT, and IT auditing. Mr. Sokroeurn Ang completed a Micro-Master in Cybersecurity at the Rochester Institute of Technology (RIT), USA, and earned a Master's degree in Cybersecurity from Royal Holloway, University of London, UK. He is currently pursuing a PhD in Cybersecurity at the Lincoln University College, Malaysia.Mr. Sokroeurn Ang has been certified such as CISSP, CISA, CISM, CC, ECSA, CEH, CCNA Security, CCNA, CyberOps, and AWS Certified Cloud Practitioner. In addition, he is a certified Cisco Instructor and an AWS Academy Instructor. Email: angsokroeurn.phdscholar@lincoln.edu.my

Author Sopheatra Huy. Sopheatra Huy is a Ph.D. candidate in Information Technology at Lincoln University College, Malaysia. He holds an M.Sc. in IT from the Royal University of Phnom Penh, Cambodia. With over a decade of experience as a part-time lecturer, he has taught programming, software engineering, and IT project management. Professionally, he is the Senior Manager of IT Audit at WB Finance and has previously worked with Phillip Bank and PRASAC MFI in similar roles. His research interests include IT automation, cybersecurity, and audit technologies.

He can be contacted at email: hsopheaktra.phdscholar@lincoln.edu.my

Author Dr. Midhunchakkaravarthy Janarthanan. He is the Dean of the School of AI Computing and Multimedia at Lincoln University College, Malaysia. He holds a Ph.D. in Computer Science and has published extensively in the fields of Big Data, Web Text Mining, Machine Learning, and GPU Computing. With over 1,000 citations on Google Scholar, his research has made significant contributions to scalable data processing and intelligent computing. Dr. Midhunchakkaravarthy also serves as a research supervisor and mentor for numerous postgraduate students, supporting innovative work in artificial intelligence and cloud-based systems.

He can be contacted at email: midhun@lincoln.edu.my