Module Code & Module Title

CS6P05NI Final Year Project

Assessment Weightage & Type

40% FYP Final Report

Automated Anomaly Detection Honeypot

Semester

2020 Autumn

Student Name:

London Met ID:

College ID:

Internal Supervisor:

External Supervisor:

Assignment Due Date:

Assignment Submission Date:

Word Count: 9198

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

Acknowledgement

First and foremost, I am grateful to Mr. Sujil Maharjan and Mr. Suryansh Mathema, my first supervisors, for their unwavering enthusiasm and motivation. I would not have been able to accomplish this report without their support and mentoring. I am also truly thankful to Mr. Raman Pradhananga, my second supervisor for allowing me to pursue my research interests with complete freedom and for his unconditional support. Through consultations and discussions with him, I was able to rise as a researcher and as a person, which also aided me to see the broader perspective.

I owe Mr. Akchayat Bikram Joshi a debt of gratitude for his invaluable guidance and genuine concern in my work at critical times. I appreciate Mr. Satyam Pradhan's willingness to review my study and provide detailed input.

I would also like to express my gratitude to Mr. Aaditya Khati for taking the time to address my questions and for assisting me in gaining a better understanding of the project in its current context.

Last but not least, I would like to express my appreciation towards my friends for assisting me with the project as well as providing inspirational motivation.

Abstract/Summary

This project addresses the emergent trends in extant honeypot research to contribute to the knowledge gaps in the honeypot environment by integrating machine learning. It provides the results that fulfill the project objectives mentioned in the proposal of the project. To achieve these objectives, detailed comprehensive research was carried out on honeypot and machine learning to gain familiarity and insights into the need of the project around the world. A comprehensive investigation was also conducted into system port vulnerabilities, which are the most common threat that attackers can use to launch an attack. Based on the proposal and the core network scenario described, the dataset was used along with two classification algorithms, anomaly detection was achieved using a supervised machine learning technique.

Table of Content

CHAPTER 1: INTRODUCTION 1
1.1 PROJECT DESCRIPTION 2
1.2 CURRENT SCENARIO 4
1.3 PROBLEM DOMAIN AND PROJECT AS A SOLUTION 5
1.4 AIM AND OBJECTIVES 7
1.4.1 AIM 7
1.4.2 OBJECTIVES 7
1.5 STRUCTURE OF THE REPORT 8
1.5.1 BACKGROUND 8
1.5.2 DEVELOPMENT 8
1.5.3 TESTING AND ANALYSIS 9
1.5.4 CONCLUSION 9
CHAPTER 2: BACKGROUND 10
2.1 ABOUT THE END USERS 11
2.2 UNDERSTANDING THE SOLUTION 12
2.2.1 SYSTEM ARCHITECTURE 12
2.2.2 WORKING MECHANISM 13
2.2.3 BACKGROUND 15
2.3 SIMILAR PROJECTS 22
2.3.1 HONEYCOMB 22
2.3.2 SPECTER 22
2.3.3 'AN INTELLIGENT HONEYPOT' – AIDEN MITCHELL 22
2.4 SIMILAR STUDIES 23
2.4.1 'CONTEXT-AWARE HONEYPOT'- SHUBHAM AGGARWAL 23
2.4.2 'DDOS MITIGATION AND INTRUSION PREVENTION USING HONEYPOTS'- MD. M. RAHMAN, S. ROY AND M. A. YOUSUF 23
2.5 COMPARISONS 24
CHAPTER 3: DEVELOPMENT 26
3.1 CONSIDERED METHODOLOGIES 27
3.1.1 SCRUM METHODOLOGY 27
3.1.2 ITERATIVE MODEL 28

3.1.3 KANBAN MODEL 28
3.2 SELECTED METHODOLOGY 30
3.3 PHASES OF METHODOLGY 31
3.3.1 BACKLOG 31
3.3.2 IMPACT ANALYSIS 32
3.3.3 BUILD 32
3.3.4 USER ACCEPTANCE TESTING 33
3.3.5 RELEASE 35
3.3.6 DOCUMENTATION 35
3.3.7 DONE 36
3.4 SURVEY RESULTS 38
3.4.1 PRE-SURVEY RESULTS 38
3.4.2 POST-SURVEY RESULTS 39
3.5 REQUIREMENT ANALYSIS 42
3.5.1 PLANNING 42
3.6 DESIGN 44
3.6.1 FLOWCHART 44
3.6.2 SEQUENCE DIAGRAM 47
3.6.3 BLOCK DIAGRAM 48
3.6.4 USE CASE DIAGRAM 49
3.6.5 ENTITY RELATION DIAGRAM 50
3.6.6 RELATIONAL DIAGRAM 50
3.6.7 DATA DICTIONARY 51
3.7 IMPLEMENTATION 52
CHAPTER 4: TESTING AND ANALYSIS 53
4.1 TEST PLAN 54
4.1.1 UNIT TESTING, TEST PLAN 54
4.1.2 SYSTEM TESTING, TEST PLAN 56
4.2 UNIT TESTING 57
4.2.1 HONEYPOT SYSTEM 57
4.3 SYSTEM TESTING 63
4.3.1 PLATFORM TESTING 63

4.3.2 HONEYPOT UI 65
4.4 CRITICAL ANALYSIS 68
CHAPTER 5: CONCLUSION 70
5.1 LEGAL, SOCIAL AND ETHICAL ISSUES 72
5.1.1 LEGAL ISSUES 72
5.1.2 SOCIAL ISSUES 72
5.1.3 ETHICAL ISSUES 73
5.2 ADVANTAGES 74
5.3 LIMITATIONS 74
5.4 FUTURE WORK 75
CHAPTER 6: REFERENCES 76
CHAPTER 7: APPENDIX 85
7.1 APPENDIX A: PRE-SURVEY 86
7.1.1 PRE-SURVEY FORM 86
7.1.2 SAMPLE OF FILLED PRE-SURVEY FORMS 91
7.1.3 PRE-SURVEY RESULT 96
7.2 APPENDIX B: POST-SURVEY 102
7.2.1 POST-SURVEY FORM 102
7.2.2 SAMPLE OF FILLED POST-SURVEY FORMS 107
7.2.3 POST-SURVEY RESULT 113
7.3 APPENDIX C: PROJECT DESCRIPTION 120
7.4 APPENDIX D: SYSTEM ARCHIETECTURE 121
7.5 APPENDIX E: WORKING MECHANISM 122
7.6 APPENDIX F: SIMILAR PROJECTS 123
7.6.1 HONEYCOMB 123
7.6.2 SPECTER 123
7.6.3 ‘AN INTELLIGENT HONEYPOT’ 123
7.7 APPENDIX G: SIMILAR STUDIES 125
7.7.1 ‘CONTEXT-AWARE HONEYPOT’ 125
7.7.2 ‘DDOS MITIGATION AND INTRUSION PREVENTION USING HONEYPOTS’ 125
7.8 APPENDIX H: REQUIREMENT SPECIFICATIONS 127

7.8.2 REQUIREMENTS ANALYSIS 127
7.9 APPENDIX I: PROPOSED SYSTEM DEPLOYMENT 133
7.10 APPENDIX J: CLASSIFICATION OF HONEYPOT 135
7.11 APPENDIX K: MACHINE LEARNING MODELS 136
7.12 APPENDIX L: ALGORITHMS 138
7.12.1 DECISION TREE 138
7.12.2 LOGISTIC REGRESSION 138
7.13 APPENDIX M: CONSIDERED METHODOLOGIES 140
7.13.1 SCRUM METHODOLOGY 140
7.13.2 ITERATIVE MODEL 140
7.13.3 KANBAN MODEL 140
7.13.4 VALIDATION OF THE METHODOLOGY 143
7.14 APPENDIX N: PHASES OF METHODOLOGY 144
7.14.1 BACKLOG 144
7.14.2 IMPACT ANALYSIS 144
7.14.3 BUILD 145
7.14.4 USER ACCEPTANCE TESTING 145
7.15 APPENDIX O: IMPLEMENTATION 147
7.15.1 SIMPLE HONEYPOT SERVER 147
7.15.2 EMAIL ALERT 150
7.15.3 DATABASE 150
7.15.4 HONEYPOT UI 153
7.15.5 ANOMALY DETECTION UI 156
7.16 APPENDIX P: TESTING 159
7.16.1 UNIT TESTING 159
7.16.2 SYSTEM TESTING 171
7.17 APPENDIX Q: INTERVIEW 185
7.18 APPENDIX R: SAMPLE CODES 187
7.18.1 MAIN.PY 187
7.18.2 DATABASEMOD.PY 189
7.18.3 ALERT.PY 190
7.18.4 INDEX.PHP 191

7.18.5 LOG.PHP 192
7.18.6 STYLE.CSS 195
7.18.7 APP.PY 196
7.19 APPENDIX S: USE CASE DESCRIPTION 201
7.20 APPENDIX T: DESIGNS 205
7.20.1 GANTT CHART 205
7.20.2 WORK BREAKDOWN STRUCTURE 209
7.20.3 WIREFRAME 210
7.21 APPENDIX U: SCREENSHOTS OF UI 216
7.21.1 HONEYPOT UI HOMEPAGE 216
7.21.2 HONEYPOT UI WHITELISTED PAGE 216
7.21.3 HONEYPOT UI BLACKLISTED PAGE 217
7.21.4 ANOMALY DETECTION UI HOMEPAGE 217
7.21.5 ANOMALY DETECTION UI ANALYSIS AND VISUALIZATION PAGE . 218
7.21.6 ANOMALY DETECTION UI CLASSIFICATION PAGE 218
7.22 APPENDIX V: USER FEEDBACK 219
7.22.1 USER FEEDBACK FORM 219
7.22.2 SAMPLE OF FILLED USER FEEDBACK FORMS 220
7.23 APPENDIX W: PROGRESS REVIEW TABLE 221
7.24 APPENDIX X: FUTURE WORK 223
7.24.1 READINGS FOR FUTURE WORK 225

Table of Figures

Figure 1 Network Attacks by types (McAfee Labs, 2018) 2
Figure 2 Damage caused by Cyberattacks from 2001-2019 (Statista, 2021) 5
Figure 3 Structure of Report 8
Figure 4 Graph of interested users in monitoring and capturing attacks (Primarily end users) 11
Figure 5 System flow of primary component 13
Figure 6 System flow of secondary component 14
Figure 7 Proposed network architecture 15
Figure 8 Timeline of Honeypot 16
Figure 9 Steps of Machine Learning (Lam, 2019) 17
Figure 10 Structure and its rule (Alsagheer, et al., 2017) 18
Figure 11 Linear Regression vs. Logistic Regression 19
Figure 12 Approaches to Anomaly Detection (Truong, 2020) 19
Figure 13 Data on the traffic collected (SCADA , 2018) 20
Figure 14 Final Dataset 21
Figure 15 Comparison Table 24
Figure 16 Scrum Methodology (Inc, 2020) 27
Figure 17 Iterative Model (tutorialspoint, 2020) 28
Figure 18 Basic Kanban Board (Eisele, 2018) 29
Figure 19 Backlog tasks in Kanban Board 31
Figure 20 Impact Analysis in Kanban Board 32
Figure 21 Build Tasks in Kanban Board 33
Figure 22 User Acceptance Task in Kanban Board 34
Figure 23 Release Tasks in Kanban Board 35
Figure 24 Documentation Tasks in Kanban Board 36
Figure 25 Completed Tasks in Kanban Board 37
Figure 26 Awareness of Cybersecurity attacks 38
Figure 27 Necessity of the project in the community 39
Figure 28 Best Feature in the project 40
Figure 29 Rating of the features 40
Figure 30 Flowchart of Primary Component 44
Figure 31 Flowchart of Secondary Component 45
Figure 32 Combined Flowchart 46
Figure 33 Sequence diagram of primary component 47
Figure 34 Sequence diagram of secondary component 47
Figure 35 Block Diagram of Primary Component 48
Figure 36 Use Case Diagram of the system 49
Figure 37 Entity Relation Diagram 50
Figure 38 Relational Diagram 50
Figure 39 Data Dictionary 51
Figure 40 Test Results of Test Case 1 checking admin rights in Windows 57
Figure 41 Test Results of Test Case 1 checking admin rights in Linux 58

Figure 42 Test Results of Test Case 2 checking input value..... 59
Figure 43 Test Results of Test Case 3 checking port range..... 60
Figure 44 Test Results of Test Case 4 capturing hit from Whitelist IP..... 61
Figure 45 Test Results of Test Case 5 capturing hit from blacklisted IP in Windows..... 62
Figure 46 Test Results of Test Case 5 capturing hit from blacklisted IP in Linux 62
Figure 47 Test Results of Test Case 1 running system in windows..... 63
Figure 48 Test Results of Test Case 2 running system in Linux..... 64
Figure 49 Test Results of Test Case 3 displaying home interface..... 65
Figure 50 Test Results of Test Case 4 checking blacklisted logs in interface..... 66
Figure 51 Test Results of Test Case 5 checking whitelisted logs in interface 67
Figure 52 Pre-Survey Form 1..... 86
Figure 53 Pre-Survey Form 2..... 87
Figure 54 Pre-Survey Form 3..... 88
Figure 55 Pre-Survey Form 4..... 89
Figure 56 Pre-Survey Form 5..... 90
Figure 57 Sample of Pre-Survey Form 1..... 91
Figure 58 Sample of Pre-Survey Form 2..... 92
Figure 59 Sample of Pre-Survey Form 3..... 93
Figure 60 Sample of Pre-Survey Form 4..... 94
Figure 61 Sample of Pre-Survey Form 5..... 95
Figure 62 Result of Pre-Survey Form 1 96
Figure 63 Result of Pre-Survey Form 2 97
Figure 64 Result of Pre-Survey Form 3 98
Figure 65 Result of Pre-Survey Form 4 99
Figure 66 Result of Pre-Survey Form 5 100
Figure 67 Result of Pre-Survey Form 6 101
Figure 68 Post-Survey Form 1 102
Figure 69 Post-Survey Form 2 102
Figure 70 Post-Survey Form 3 102
Figure 71 Post-Survey Form 4 103
Figure 72 Post-Survey Form 5 103
Figure 73 Post-Survey Form 6 103
Figure 74 Post-Survey Form 7 104
Figure 75 Post-Survey Form 8 104
Figure 76 Post-Survey Form 9 105
Figure 77 Post-Survey Form 10 105
Figure 78 Post-Survey Form 11 105
Figure 79 Post-Survey Form 12 105
Figure 80 Post-Survey Form 13 106
Figure 81 Post-Survey Form 14 106
Figure 82 Post-Survey Form 15 106
Figure 83 Sample of Post-Survey 1..... 107
Figure 84 Sample of Post-Survey 2..... 107

Figure 85 Sample of Post-Survey 3 108
Figure 86 Sample of Post-Survey 4 108
Figure 87 Sample of Post-Survey 5 108
Figure 88 Sample of Post-Survey 6 109
Figure 89 Sample of Post-Survey 7 109
Figure 90 Sample of Post-Survey 8 110
Figure 91 Sample of Post-Survey 9 110
Figure 92 Sample of Post-Survey 10 110
Figure 93 Sample of Post-Survey 11 111
Figure 94 Sample of Post-Survey 12 111
Figure 95 Sample of Post-Survey 13 111
Figure 96 Sample of Post-Survey 14 112
Figure 97 Sample of Post-Survey 15 112
Figure 98 Result of Post-Survey 1 113
Figure 99 Result of Post-Survey 2 114
Figure 100 Result of Post-Survey 3 114
Figure 101 Result of Post-Survey 4 115
Figure 102 Result of Post-Survey 5 115
Figure 103 Result of Post-Survey 6 116
Figure 104 Result of Post-Survey 7 116
Figure 105 Result of Post-Survey 8 117
Figure 106 Result of Post-Survey 9 117
Figure 107 Result of Post-Survey 10 118
Figure 108 Result of Post-Survey 11 118
Figure 109 Result of Post-Survey 12 119
Figure 110 Result of Post-Survey 13 119
Figure 111 Major tools used in the development 127
Figure 112 Network Architecture 133
Figure 113 VLAN Segregation 134
Figure 114 IP address check of honeypot 134
Figure 115 Ping successful from attacker to honeypot 134
Figure 116 Flowchart of Supervised Machine Learning (Dong Nguyen, 2016) 136
Figure 117 Flowchart of Unsupervised Machine Learning (Technative, 2020) 136
Figure 118 Principle of Semi-Supervised Learning (LOTTE, 2015) 137
Figure 119 Limit WIP (Klipp, 2014) 142
Figure 120 Creation of INET socket stream 147
Figure 121 Privilege Check 147
Figure 122 Logging code 148
Figure 123 WhiteIP Configure 148
Figure 124 Linux Platform Check 148
Figure 125 Windows Platform Check 149
Figure 126 NMAP Automation 149
Figure 127 Email alert 150

Figure 128 Importing modules in databasemod.py 150
Figure 129 Converting into CSV 151
Figure 130 Reading file with Pandas 151
Figure 131 Creating database table 152
Figure 132 Fetching data from database 153
Figure 133 Pushing data starting from [!] Hit from w% 154
Figure 134 Pushing data from [+] B% 155
Figure 135 Importing and assigning module variables in app.py 156
Figure 136 Data Interpretation Code 156
Figure 137 Analysis and Visualization Code 157
Figure 138 Tuning Parameters in Models 158
Figure 139 Splitting Data into training and testing 158
Figure 140 Test Results of Test Case 6 checking Firewall rules in Windows 159
Figure 141 Test Results of Test Case 6 checking Firewall rules in IP Tables 160
Figure 142 Test Results of Test Case 7 checking connection in browser 161
Figure 143 Test Results of Test Case 8 releasing Firewall rules in Windows 162
Figure 144 Test Results of Test Case 8 releasing Firewall rules in Linux 163
Figure 145 Test Results of Test Case 9 checking logs in text files 164
Figure 146 Test Results of Test Case 9 checking logs in csv format 165
Figure 147 Test Results of Test Case 9 checking logs in spreadsheet 165
Figure 148 Test Results of Test Case 10 checking data stored in database 166
Figure 149 Test Results of Test Case 10 checking data stored in database 167
Figure 150 Test Results of Test Case 11 checking Email received 168
Figure 151 Test Results of Test Case 11 executing the script 168
Figure 152 Test Results of Test Case 12 selecting file from browser 169
Figure 153 Test Results of Test Case 12 checking if correct file is chosen 170
Figure 154 Test Results of Test Case 6 executing anomaly detection interface 171
Figure 155 Test Results of Test Case 6 displaying anomaly detection interface 172
Figure 156 Test Results of Test Case 7 browsing dataset 173
Figure 157 Test Results of Test Case 7 displaying dataset 174
Figure 158 Test Results of Test Case 8 displaying data interpretation check boxes 176
Figure 159 Test Results of Test Case 8 displaying data interpretation check boxes 176
Figure 160 Test Results of Test Case 8 displaying data interpretation check boxes 177
Figure 161 Test Results of Test Case 8 displaying data interpretation check boxes 177
Figure 162 Test Results of Test Case 9 displaying analysis and visualization 179
Figure 163 Test Results of Test Case 9 displaying graphs 179
Figure 164 Test Results of Test Case 9 displaying graphs 180
Figure 165 Test Results of Test Case 10 displaying predictions of logistic regression 181
Figure 166 Test Results of Test Case 11 displaying predictions of logistic regression 182
Figure 167 Test Results of Test Case 12 displaying predictions of decision tree 183
Figure 168 Test Results of Test Case 13 displaying predictions of decision tree 184
Figure 169 Code Snippet of Main.py 1 187
Figure 170 Code Snippet of Main.py 2 187

Figure 171 Code Snippet of Main.py 3 188
Figure 172 Code Snippet of Main.py 4 188
Figure 173 Code Snippet of Main.py 5 188
Figure 174 Code Snippet of Main.py 6 189
Figure 175 Code Snippet of Main.py 7 189
Figure 176 Code Snippet of databasemod.py 1 189
Figure 177 Code Snippet of databasemod.py 2 190
Figure 178 Code Snippet of databasemod.py 3 190
Figure 179 Code Snippet of alert.py 1 190
Figure 180 Code Snippet of alert.py 2 191
Figure 181 Code Snippet of index.php 1 191
Figure 182 Code Snippet of index.php 2 192
Figure 183 Code Snippet of log.php 1 192
Figure 184 Code Snippet of log.php 2 192
Figure 185 Code Snippet of log.php 3 193
Figure 186 Code Snippet of log.php 4 193
Figure 187 Code Snippet of log.php 5 194
Figure 188 Code Snippet of log.php 6 194
Figure 189 Code Snippet of style.css 195
Figure 190 Code Snippet of app.py 1 196
Figure 191 Code Snippet of app.py 2 196
Figure 192 Code Snippet of app.py 3 196
Figure 193 Code Snippet of app.py 4 197
Figure 194 Code Snippet of app.py 5 198
Figure 195 Code Snippet of app.py 6 198
Figure 196 Code Snippet of app.py 7 199
Figure 197 Code Snippet of app.py 8 199
Figure 198 Code Snippet of app.py 9 200
Figure 199 Updated Gantt chart 1 205
Figure 200 Updated Gantt chart 2 206
Figure 201 Updated Work Breakdown Structure 209
Figure 202 Wireframe of Honeypot UI 1 210
Figure 203 Wireframe of Honeypot UI 2 211
Figure 204 Wireframe of Honeypot UI 3 212
Figure 205 Wireframe of Anomaly Detection UI 1 213
Figure 206 Wireframe of Anomaly Detection UI 2 214
Figure 207 Wireframe of Anomaly Detection 3 215
Figure 208 Honeypot UI Homepage 216
Figure 209 Honeypot UI Whitelisted page 216
Figure 210 Honeypot UI Blacklisted page 217
Figure 211 Anomaly Detection UI 217
Figure 212 Anomaly Detection UI Analysis and Visualization 218
Figure 213 Anomaly Detection UI Classification 218

Figure 214 User Feedback Form 1 219
Figure 215 User Feedback Form 2 220

Table of Tables

Table 1 Unit Testing....54
Table 2 System Testing....56
Table 3 Honeypot System Test Case 1....57
Table 4 Honeypot Module Test Case 2....59
Table 5 Honeypot Module Test Case 3....60
Table 6 Honeypot Module Test Case 4....61
Table 7 Honeypot Module Test Case 5....62
Table 8 Platform Testing Test Case 1....63
Table 9 Platform Testing Test Case 2....64
Table 10 Honeypot UI Test Case 3....65
Table 11 Honeypot UI Test Case 4....66
Table 12 Honeypot UI Test Case 5....67
Table 13 Module Requirements....129
Table 14 Honeypot Module Test Case 6....159
Table 15 Honeypot Module Test Case 7....161
Table 16 Honeypot Module Test Case 8....162
Table 17 Honeypot Module Test Case 9....164
Table 18 Honeypot Module Test Case 10....166
Table 19 Honeypot Module Test Case 11....168
Table 20 Anomaly Detection Test Case 12....169
Table 21 Anomaly Detection UI Test Case 6....171
Table 22 Anomaly Detection UI Test Case 7....173
Table 23 Anomaly Detection UI Test Case 8....175
Table 24 Anomaly Detection UI Test Case 9....178
Table 25 Anomaly Detection UI Test Case 10....181
Table 26 Anomaly Detection UI Test Case 11....182
Table 27 Anomaly Detection UI Test Case 12....183
Table 28 Anomaly Detection UI Test Case 13....184
Table 29 Use Case 1....201
Table 30 Use Case 2....201
Table 31 Use Case 3....202
Table 32 Use Case 4....202
Table 33 Use Case 5....202
Table 34 Use Case 6....203
Table 35 Use Case 7....203
Table 36 Use Case 8....204
Table 37 Use Case 9....204
Table 38 Gantt chart table....207
Table 39 Progress Table....221

List of Abbreviations

IT	**Information Technology**
VM	**Virtual Machine**
TCP	**Transmission Control Protocol**
OS	**Operation System**
ML	**Machine Learning**
HTTP	**Hyper Text Transfer Protocol**
RAM	**Random Access Memory**
DT	**Decision Tree**
LR	**Logistic Regression**

CHAPTER 1: INTRODUCTION

1.1 PROJECT DESCRIPTION

Information security has been one of the most significant areas of research and exploration in the IT industry. Attackers are usually one step ahead of defenders and the attacks are being modified every day so our current security technologies cannot capture them all (Sardana & Joshi, 2011). There are several instances of Cyberattacks appearing to make headlines almost every day resulting in huge losses in cyberspace. According to (Fan, et al., 2018), Symantec found 54 zero-day vulnerabilities in 2015, which was up by 125 percent from 2014. In June 2018, McAfee Labs had published a threat report on eight different network attacks which were held from 2.9 million sample sensors that were managed by McAfee (McAfee Labs, 2018) as provided in figure 1. With the incidence of cybercrime rising by the day, the Covid-19 pandemic had provided cybercriminals with an unrivalled incentive to brace for security breaches in 2020. In reality, 80% of businesses set down a rise in cyberattacks in 2020. The pandemic is solely responsible for a 238% increase in cybercrime.

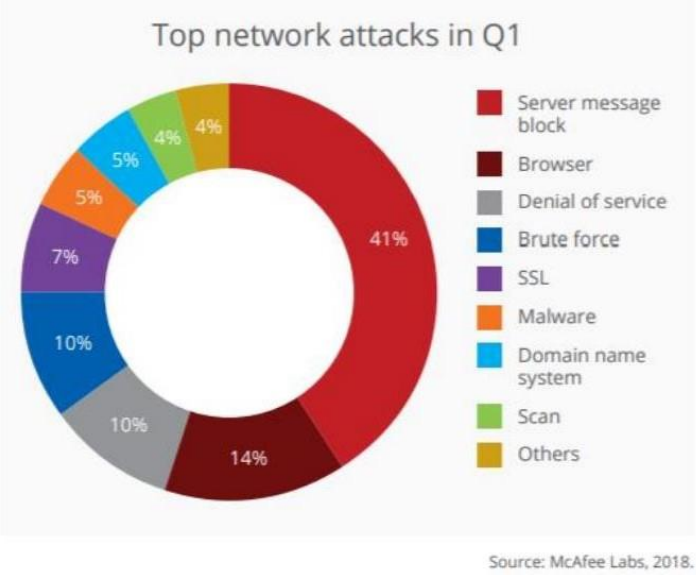


Figure 1 Network Attacks by types (McAfee Labs, 2018)

Moreover, studies show that most people are oblivious about their machines being probed several times a day, with tools explicitly made for that intention, such as NMAP, which allows network users to test networks for common vulnerabilities. Since probes often come from infected devices,

determining the source can be beneficial to their operators. As a consequence, merely protecting our networks is inadequate. It is also crucial to be aware that a probing attack has occurred.

In recent years, many tools and software have been developed and providing professional and organizational security. Although there are various powerful and diverse tools available in information security systems to detect potential security vulnerabilities, it has always been a challenge to analyse the systems properly and collect detailed information about the hackers and their motives. This is the motivation for the project to create an **Automated Anomaly Detection Honeypot**. This project is a combination of a low-interaction honeypot with anomaly detection which detects anomalies based on the logs generated by the honeypot, which helps to identify attacks before reaching the system and learn about the tools and intentions of the attackers to help develop new techniques and safe networks.

The further explanation is discussed in the appendix: PROJECT DESCRIPTION

1.2 CURRENT SCENARIO

Presently, there is a missing of new comprehensive information about the application of honeypot in the production environment and its data analysis all around the world. To date, much of the work on honeypots have been dedicated to the configuration or optimization of the new honeypots. Honeypots like Honeyd, Cowrie, Glastopf are some of the popular honeypots in the market. But frameworks for analyzing data gathered from honeypots with machine learning, particularly low-interaction honeypots, are currently immature. This study addresses the need for more intricate techniques to be developed to analyze network traffic data gathered from low-interaction honeypots.

Mr. Aaditya Khati, a cybersecurity analyst at Cryptogen Nepal, was interviewed for the qualitative research for this project on the latest cases of honeypots in Nepal. According to Mr. Khati, the idea of deploying a honeypot is still relatively new in the Nepalese market, and as a result, Nepalese people's cybersecurity awareness and knowledge are still emerging. Nepal is also in the preliminary stage of research and study in that specific domain.

The interview questions are available: INTERVIEW

1.3 PROBLEM DOMAIN AND PROJECT AS A SOLUTION

As mentioned above, cyberattacks rank as the fastest rising crime, causing disastrous business disruption (Downs, 2020). Cybercrime impact is expected to hit the US $6 trillion globally by 2021 (Downs, 2020). Cybercrime has reached its all-time peak over the past years. But it doesn't seem like it will end there, as cyber-attacks seem to increase in size every year. The figure shows the level of damage caused by malicious cybercrime published to IC3 from 2001 to 2019.

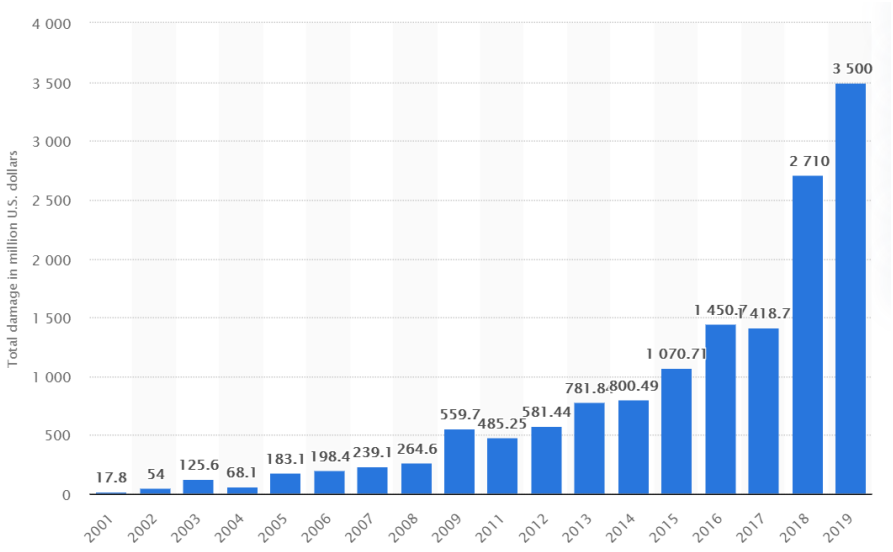


Figure 2 Damage caused by Cyberattacks from 2001-2019 (Statista, 2021)

One of the main issues that today's security tools are failing is because cyberattacks are a lot more dynamic and human processes are not quick enough to keep up with this shift (Schreier, 2014). The survey showed that the security teams cannot investigate all the alerts and it becomes difficult to extract relevant information and value from the data if processed manually. Also because of the presence of too many tools, it requires switching from one console to the next to detect threats which are a manual and time-consuming task.

This will be the main problem scenario that will be addressed and try to solve in this project using the case study of machine learning and a passive honeypot. Honeypots are usually a device which is intentionally kept vulnerable to let the attackers come in and attack. So, in most cases, attackers will be the ones trying to connect to the honeypot which makes sure that the information gathered by it will be small but of great value. As a result, honeypots minimize noise by gathering only

limited datasets of high value, as they are only mostly used by attackers. This ensures that analysing and extracting value from the data collected by a honeypot is much simpler.

Therefore in this project, the honeypots are designed to mimic a protocol for an intruder to interact with; all the honeypot needs is a connection from a suspicious network, after which it executes action such as blacklisting it. While most of the attackers use automated attack scans to connect to the target on the internet constantly, it is typically only aimed at intruders who try to connect to unexpected ports to know what services run on them. After the connection, logs are generated and a real-time alert is sent to the administrator. A cybersecurity dataset (WUSTL-IIOT-2018) will be used for a machine learning case study to analyze whether it's an anomaly attack or not. The dataset includes attack data as recorded in real-life which will help to understand and aware of the probability of the attacks at the particular time. This project will help to detect the attack early and assist in preventing it on time without compromising any system in the organization.

1.4 AIM AND OBJECTIVES

1.4.1 AIM

The main aim of this project is to develop a honeypot framework that can detect and avoid disruptive attacks with the aid of the case study of anomalous behavior data and will have the ability to learn from data intrusions with the help of a machine learning approach.

1.4.2 OBJECTIVES

The objectives that are expected to complete to achieve the aim of this project are as follows:

- Conduct extensive research of the benchmark works performed earlier in the field of honeypot and machine learning to gain familiarity and insights into the need of the project around the world.
- Acknowledge the importance of professional, technical, ethical, and legal standards including plagiarism, licensing, deceptive information, or copyrighted content.
- Comprehend different aspects of the project, idea, or strategy in-depth to become conscious of any possible issues that can arise during its development and deployment.
- Perform community surveys and interview of appropriate security professionals before and after the completion of the project and obtain feedback.
- Review existing commercial honeypots such as honeyd, specter to grasp strong knowledge of the working mechanism of honeypots.
- Acquire a good understanding of python libraries such as socket, logging, NumPy for the development of the project by following documentation and tutorials available online and complete the development phase before the end of January.
- Develop a web-based framework for the honeypot to analyze log data simply and effectively.
- Develop a web-based framework for machine learning to analyze datasets simply and effectively.
- Prepare and refer the Kanban board to achieve the overall work status of the project and also collect artifacts that support in-depth insights into the research.
- Validate if the project is complete and test several times (Unit testing and System testing) to reach the expectation of the desired output.
- Summarize and document under the guidance and feedback of the author's supervisor(s).

1.5 STRUCTURE OF THE REPORT

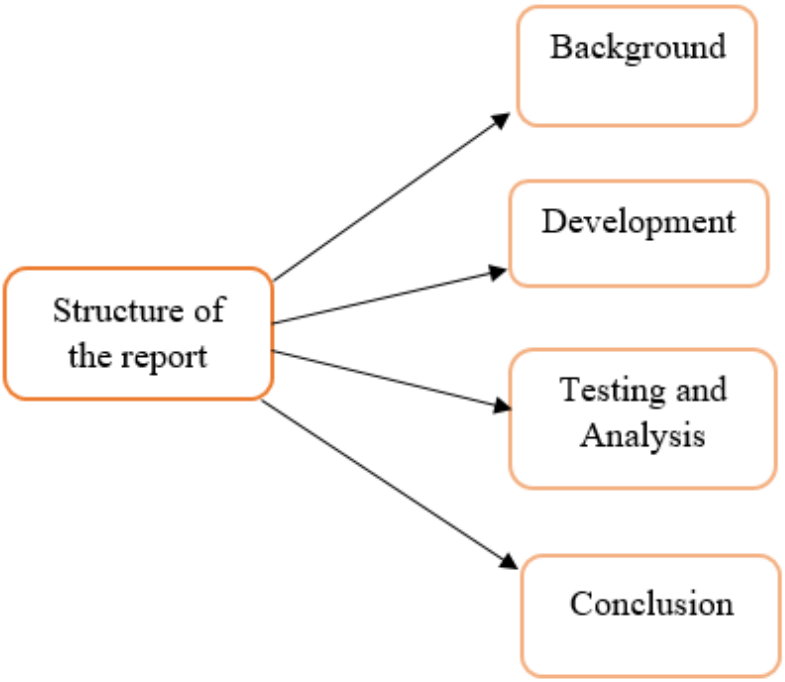


Figure 3 Structure of Report

1.5.1 BACKGROUND

This chapter offers a summary of the theory and technology of honeypots. This chapter also discusses the machine learning field around the project. Research into anomalous detection is also described. Also, this chapter addresses the detail-related work applicable to the project.

1.5.2 DEVELOPMENT

This chapter consists of the development stages executed for the project. It discusses the considered methodologies, selected methodology, and analysis of the selected methodology, requirement analysis, design, and development implementation. It also compromises the analysis of pre-survey and post-survey outcomes. Furthermore, the development of the project following a chosen methodology (Kanban Methodology) is the chapter's main priority.

1.5.3 TESTING AND ANALYSIS

The Testing and Analysis segment consists of test plans and test cases. Unit tests and machine tests are two types of testing scenarios. Lastly, critical analysis is discussed in detail.

1.5.4 CONCLUSION

The concluding chapter summarizes the report and addresses legal, ethical, and social concerns, as well as the system's advantages, limitations, and future scope. Overall, it offers a thorough description of the system and its results.

CHAPTER 2: BACKGROUND

2.1 ABOUT THE END USERS

This project is intended for enthusiasts out there who are interested in the Network Security domain and would be interested in monitoring or capturing network information. The analytic data can be helpful to analysts to learn about the resource's targeted features, which can then be defended in the real system with greater precision. The monitoring and analytics framework for honeypots may be used for study purposes. Also, students can communicate with honeypots as a project and learn more about vulnerabilities. This is an open-source project and will be released under the GNU GPLv3 license providing a framework for implementation and development of services. However, the users with a basic understanding of the technology that underpins the Internet are expected from the main end-users of this system.

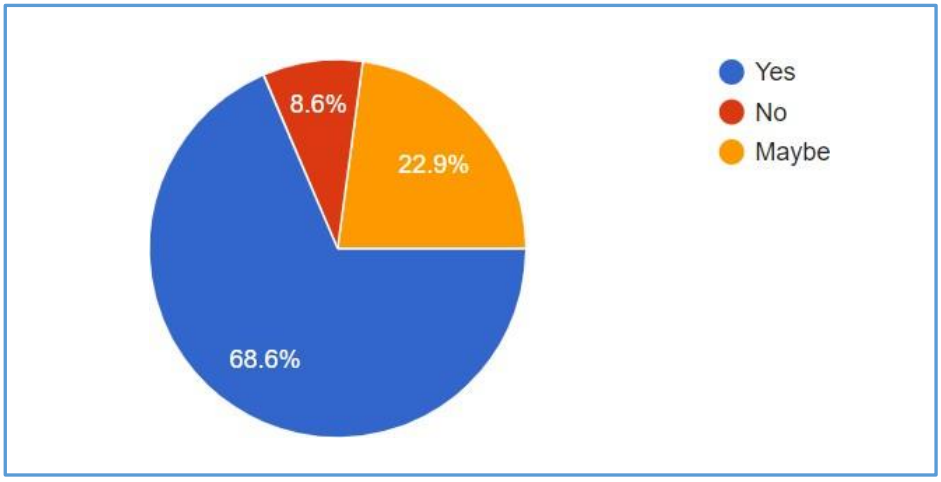


Figure 4 Graph of interested users in monitoring and capturing attacks (Primarily end users)

2.2 UNDERSTANDING THE SOLUTION

2.2.1 SYSTEM ARCHITECTURE

The following is the explanation of the honeypot module and anomaly detection, the primary and secondary components respectively. These are the major module components of the system that paved the way for the complete framework of the project.

2.2.1.1 HONEYPOT MODULE

The honeypot module is the primary module which is subdivided into a log module, database module, web interface module, and alert module. A basic server written in Python was the first objective in developing the honeypot. It is a modest version of a honeypot where the server allows it to connect to the server and then echo the client text back to the server. The honeypot was built as a Python port listener on the IPv4 address of common ports using a socket. It establishes an HTTP session and a socket on which to listen for connections. The honeypot first seeks out any connection from the intruder and immediately acts blacklisting the unknown user. NMAP automation has also been introduced to quickly access NMAP port result data to be handy to system administrators who want to automate scanning tasks and reports.

A further explanation is provided in the appendix: SYSTEM ARCHITECTURE

2.2.1.2 ANOMALY DETECTION

The User Interface module is an added feature in the Machine learning case study in the project. The classification algorithms used in this study are a combination of the Logistic Regression and Decision Tree algorithms, which produce high precision and accurate results. A user-friendly application is built which is incredibly easy to use and learn machine learning effectively with the help of Streamlit.

In particular, the machine learning model performs a pipeline that includes steps such as:

- Analyzing the dataset
- Choosing the features and the target
- Creating a train set and a test set from the dataset
- Display visualizations and analysis of the dataset.
- Predicting data from the test set.

2.2.2 WORKING MECHANISM

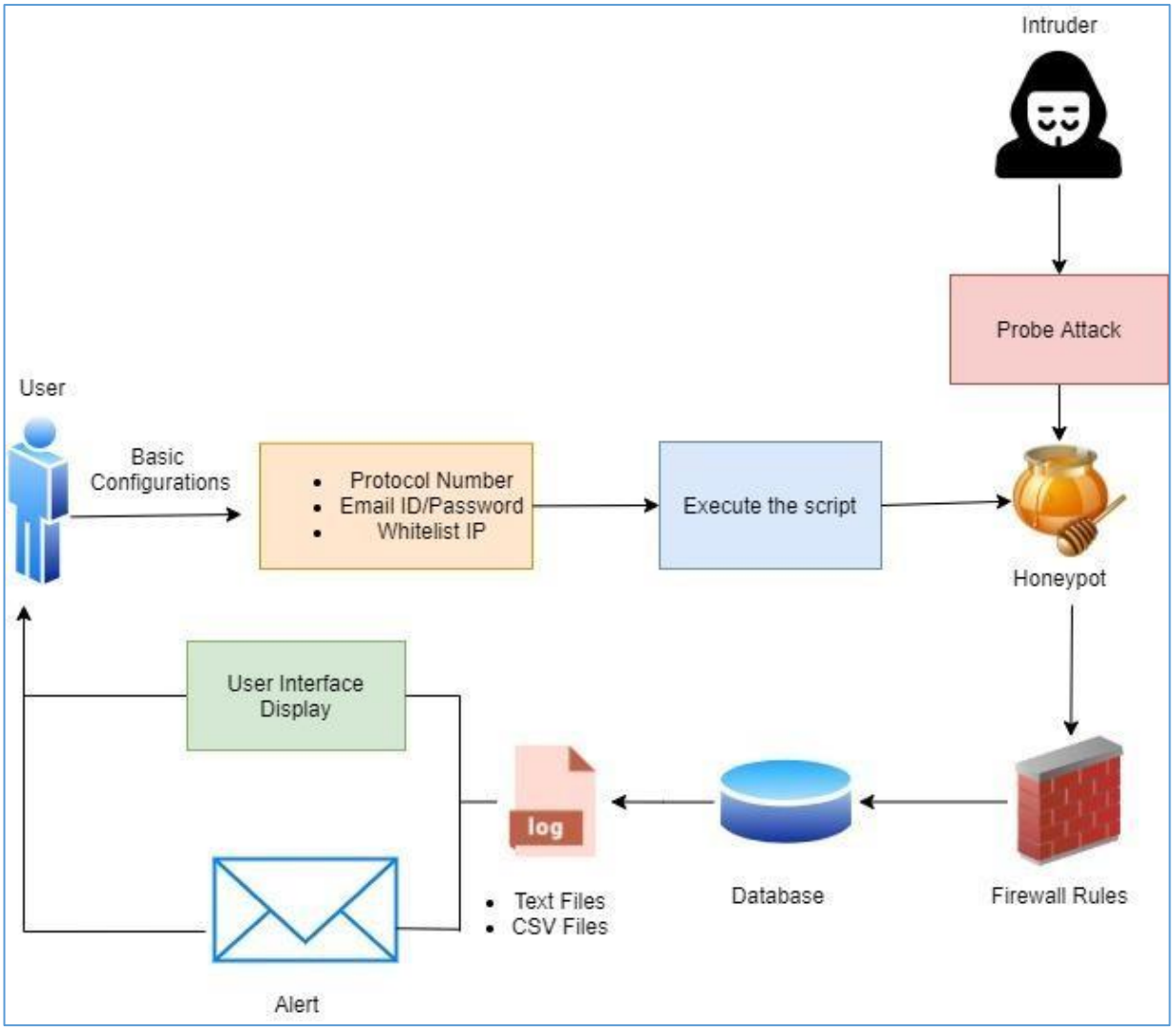


Figure 5 System flow of primary component

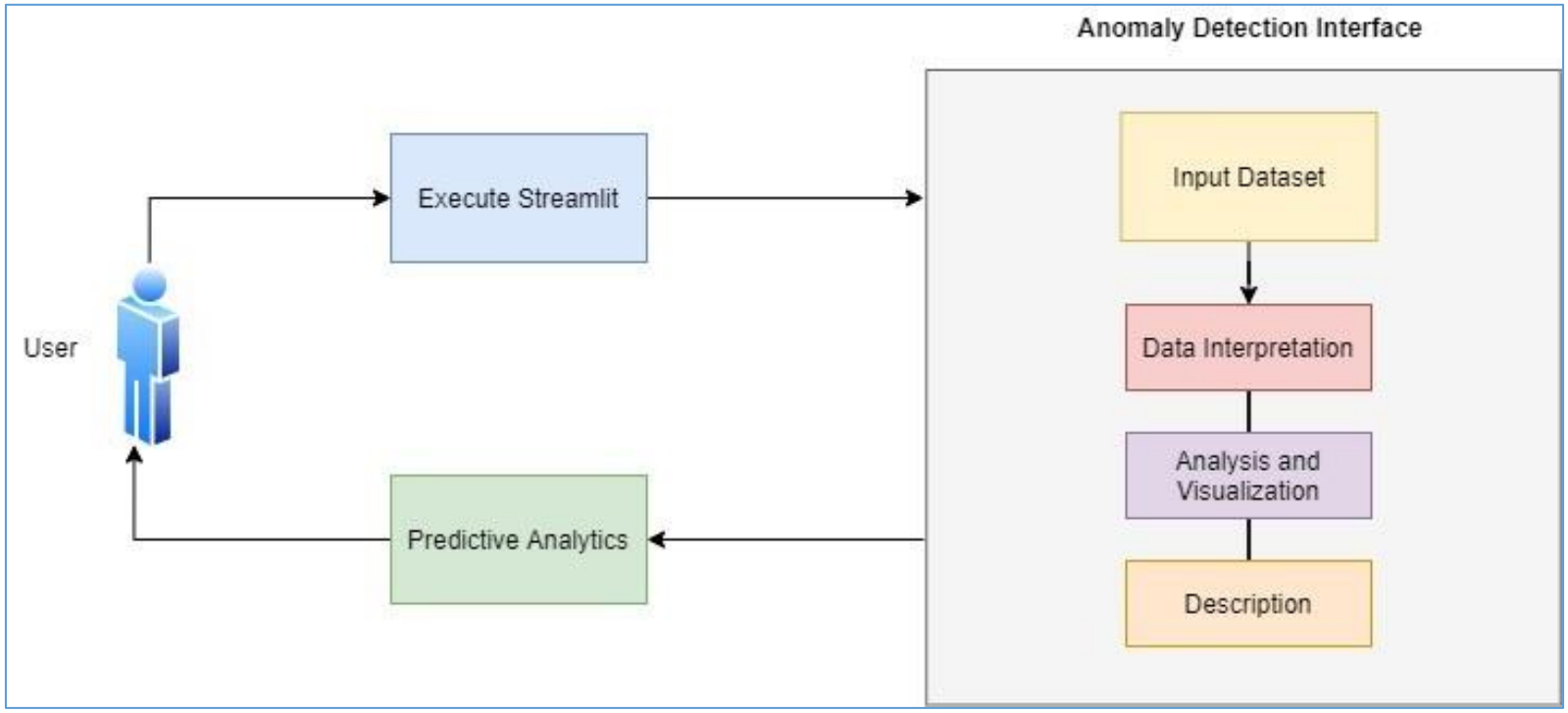


Figure 6 System flow of secondary component

A detailed explanation of the system is discussed in the appendix: WORKING MECHANISM

2.2.3 BACKGROUND

2.2.3.1 HONEYPOT

A honeypot is known as the most famous and widely deployed security tool that is purposely built to be targeted and compromised. It is also used for detecting and misdirecting unauthorized access to protect production systems. It is also valuable in analyzing attackers' actions and in particular, unknown attacks (Fan, et al., 2018). The role of the honeypot in the organization is defined by how the honeypot is built and how information gathered from it is used. Honeypot detects attackers in general and captures their actions. In the ability to record almost just malicious activity, its benefit is reflected. As honeypots are configured so that only intruders scanning the network will attempt to access them, the possibility of false negatives tends to be zero (Kovtun, 2018).

The classification of a honeypot is provided in the appendix: CLASSIFICATION OF HONEYPOT

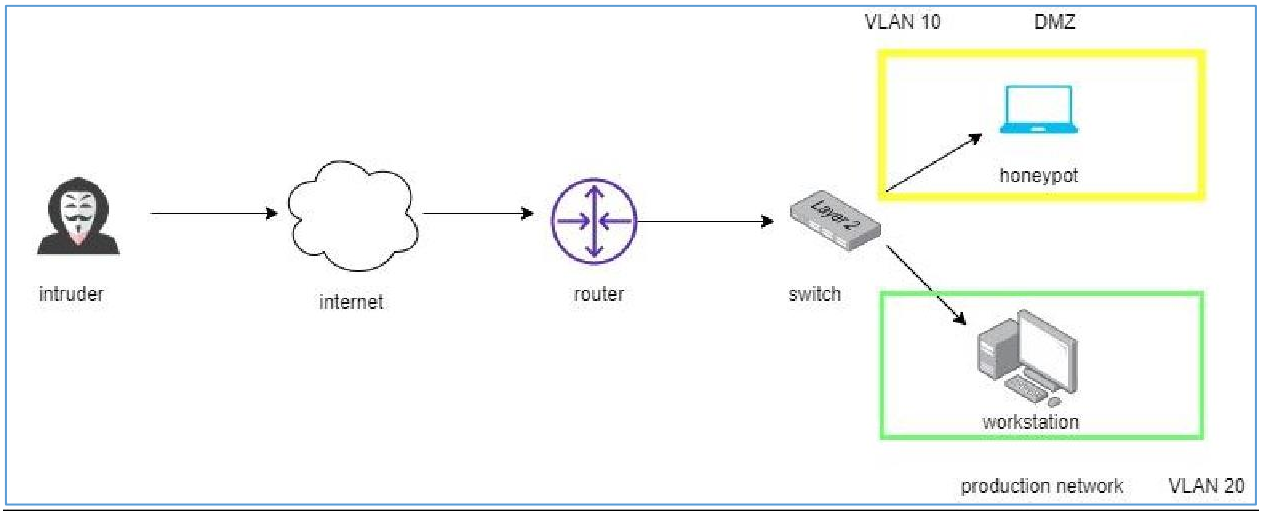


Figure 7 Proposed network architecture

A brief explanation about the proposed system architecture is provided in the appendix: PROPOSED SYSTEM DEPLOYMENT

2.2.3.3 HISTORY OF HONEYPOT

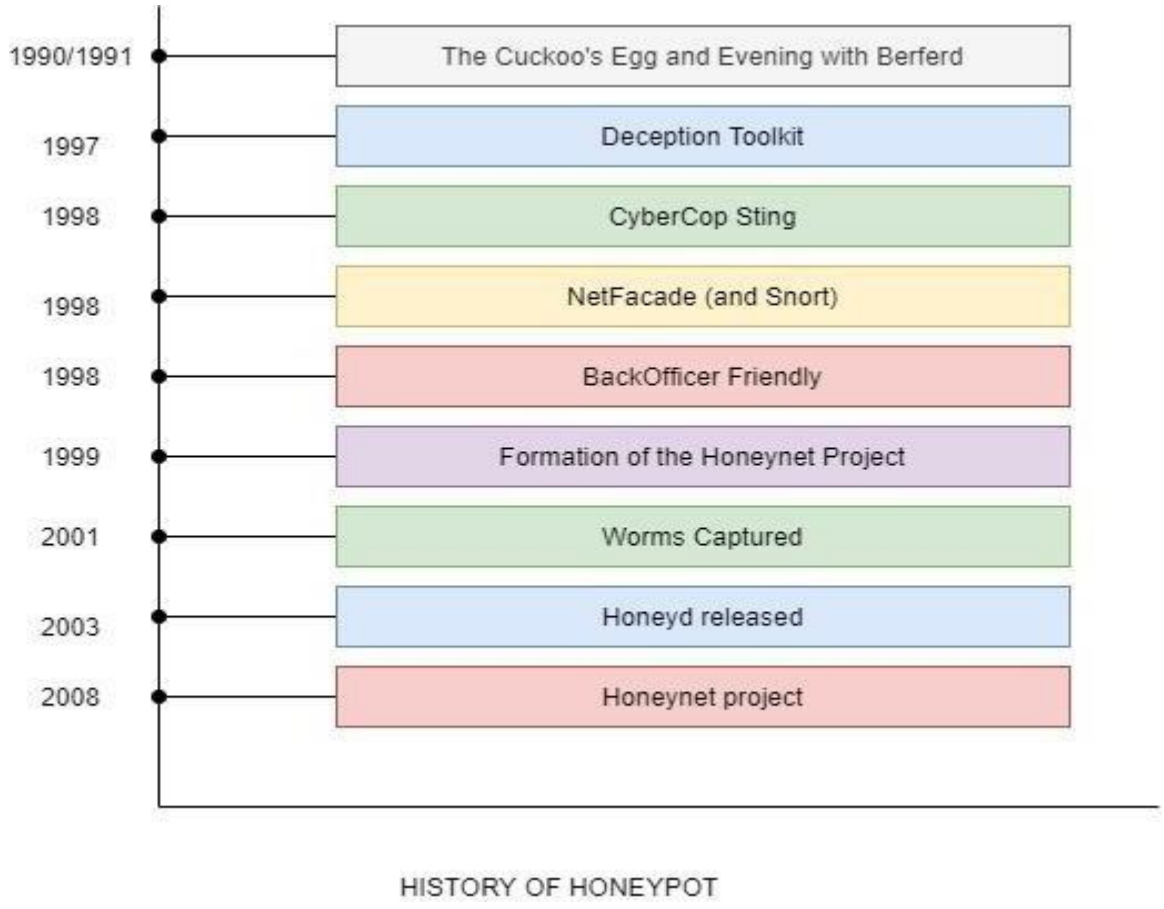


Figure 8 Timeline of Honeypot

2.2.3.4 CASE STUDY

A new Slammer worm stormed the Internet on Jan. 25, 2003, infecting thousands of servers every minute running Microsoft's SQL Server program (Litchfield, 2019). It was history's fastest expanding computer worm and honeypots were the first security tool to detect them. A buffer overflow was triggered on UDP port 1434 and compromised more than 200,000 computers during its first 10 minutes of operation. This knocked down a big banking ATM network and exacerbated attacks on Denial of Service (DoS) across large parts of the Internet. An inclusive and integrated effort to block port 1434 traffic headed through major backbones allowed early detection, preventing the worm from spreading any further (Sardana & Joshi, 2011). This was largely due to early identification by honeypots and IDSs (Sardana & Joshi, 2011).

2.2.3.4 MACHINE LEARNING

Machine learning is a domain of AI dedicated to developing applications that, without being programmed to do so, learn from huge data and advance their precision over time. It is a technology that helps machines to understand from data-based interactions and examples (The Royal Society, 2017). Machine learning emphasizes applications that learn from experience over time and strengthen their decision-making or predictive accuracy (IBM, 2020).

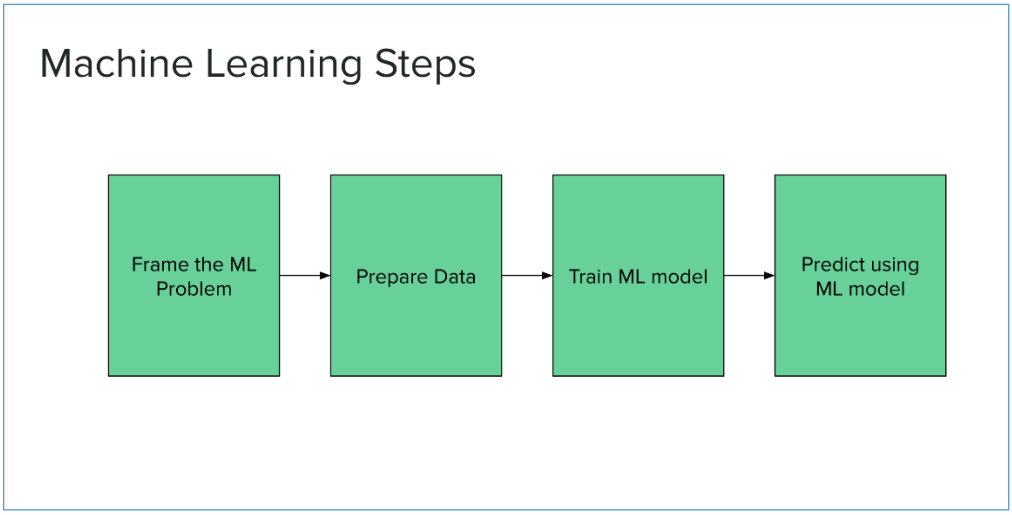


Figure 9 Steps of Machine Learning (Lam, 2019)

The machine learning algorithms are mainly differentiated into four groups according to their training data and supervision received. They are Supervised, Unsupervised, Semi-supervised, and Reinforcement learning. The algorithm is a major aspect of anomaly detection, and its efficiency, functionality, and accuracy are all influenced by how well it is designed. The program, method of approach, and parameter restraints all determine the algorithm design. Based on the sample data, there are two types of algorithms: static and dynamic (Jidiga, 2014).

The types of machine learning are explained in the appendix: MACHINE LEARNING MODELS

2.2.3.5 ALGORITHMS

In this project, Decision trees and Logistic regression are proposed for data analysis.

Decision Tree: Decision trees are those algorithms that group attributes by grouping them according to their values. The decision tree is purposed for classification. Nodes and branches make up each tree. Each branch refers to the value that the node will take, and each node contains attributes in a category that needs to be defined (Dey, 2016). There are numerous types of decision trees, each with its own set of tree-growing and tree-pruning algorithms. According to Anyanwu et al, C4.5, ID3, and CART decision tree algorithms are the most frequently used algorithms (Blaziunas & Raudys, 2019).

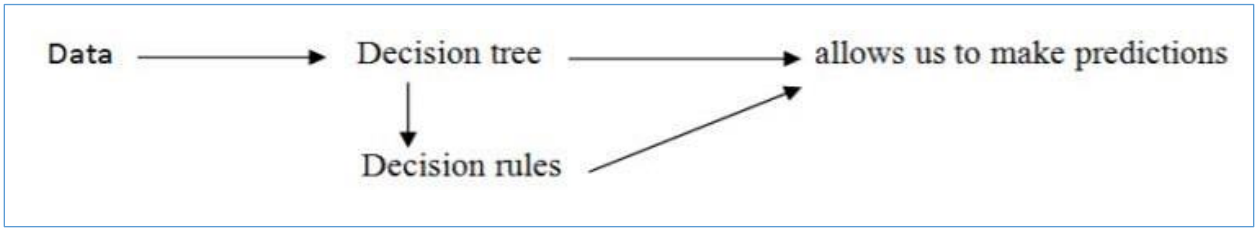


Figure 10 Structure and its rule (Alsagheer, et al., 2017)

The remaining explanation is provided in the appendix: DECISION TREE

Logistic Regression: There are primarily two types of variables namely, explanatory variables X1, X2...Xn and study variable y in the linear regression model $X\beta + \varepsilon$ (SHALABH, 2017). These variables can be evaluated both on a continuous scale and as indicator variables. When the explanatory variables are qualitative, their values are expressed as indicator variables, which are then used in dummy variable models (SHALABH, 2017). If the study variable is qualitative, its values can be represented using an indicator variable with only two possible values: 0 and 1. Logistic regression is used in this case. For example, y may represent values such as success or failure, yes or no, like or dislike, all of which can be represented by the two numbers 0 and 1 (SHALABH, 2017). The remaining explanation is provided in the appendix: LOGISTIC REGRESSION

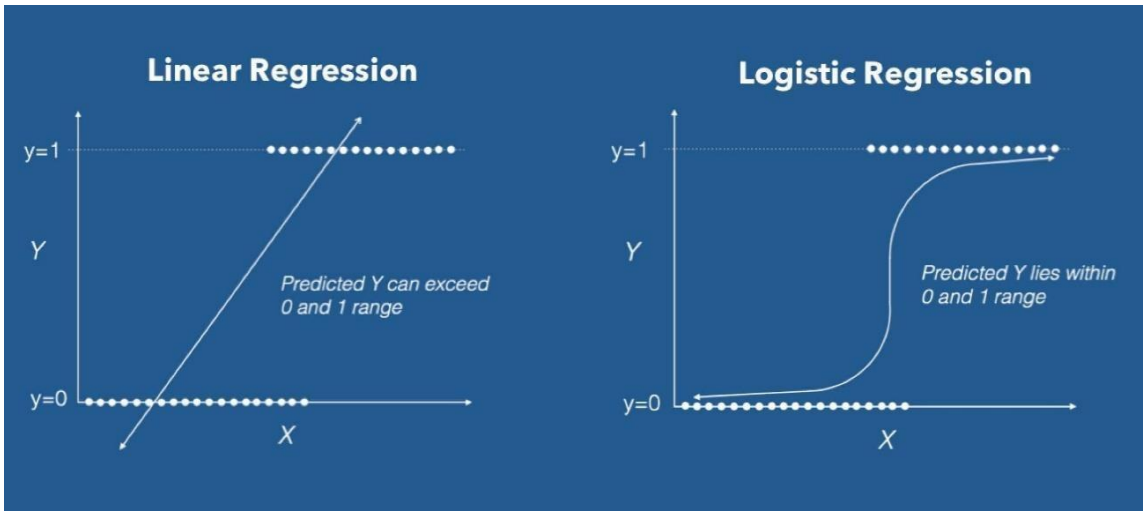


Figure 11 Linear Regression vs. Logistic Regression

2.2.3.6 ANOMALY DETECTION

The concept of identifying trends and patterns that do not correspond to normal behavior is known as anomaly detection. In various application domains, these non-corresponding patterns are alluded to as anomalies, outliers, discordant observations, deviations, peculiarities, or contaminants (CHANDOLA, et al., 2009). A network anomaly represents an unexpected, short-term deviation to network operation. Attackers with nefarious intentions such as a negative attack on an IP network are deliberately responsible for some anomalies, whilst others may be a pure occurrence such as an over-the-go pass falling into a busy road network (Ahmed, et al., 2007).

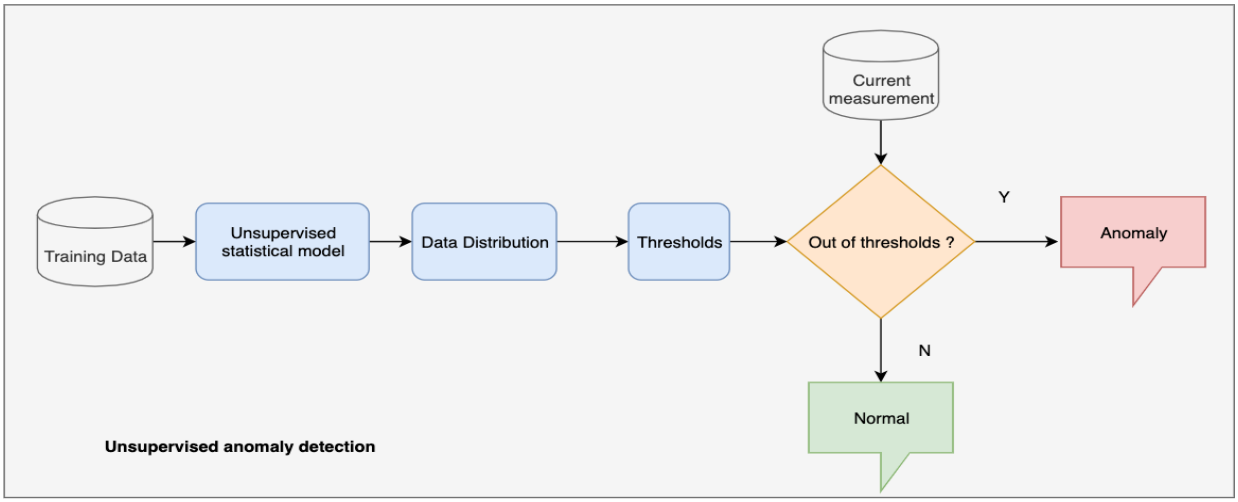


Figure 12 Approaches to Anomaly Detection (Truong, 2020)

2.2.3.7 CASE STUDY AND IMPLEMENTATION

This project aims to demonstrate how machine learning operates and why operating on a legitimate database is preferable. The dataset applied for this project was first used for the SCADA cybersecurity study in 2018, the WUSTL-IIOT-2018 Dataset which was obtained in real-time.

Case Study

The chosen case study similar to this project focuses on reconnaissance attacks in which the network is checked for potential exploits to be included in later attempts. The scan tools have been used to examine the target network's architecture and classify the systems as well as their weaknesses. The Audit Record Generation and Utilization System (ARGUS) tool has been used to track all network traffic (both regular and abnormal traffic) (SCADA , 2018). The traffic that is being tracked is recorded and saved in a "CSV" format. The statistics on the collected network activity are shown in the table below.

Measurement	Value
Duration of capture	25 Hours
Dataset size	627 MB
Number of observations	7,049,989
Percentage of port scanner attacks	0.0003%
Percentage of address scan attacks	0.0075%
Percentage of device identification attacks	0.0001%
Percentage of device identification attacks (aggressive mode)	4.9309%
Percentage of exploiting attacks	1.1312%
Percentage of all attacks (total)	6.07%
Percentage of normal traffic	93.93%

Figure 13 Data on the traffic collected (SCADA , 2018)

The raw data provided a 627 MB dataset, of which 93.93% correlates to the regular network (without attacks) and 6.07% relates to abnormal traffic (attack traffic) (SCADA , 2018). The raw data contains 25 networking features, some of which are used in the classification process and others in the training and testing of machine learning algorithms (SCADA , 2018). The process of cleaning, classifying, and labelling the dataset had begun after the data had been collected

(SCADA , 2018). The number of samples (rows in the dataset) increased to 7,037,983 after the data was cleaned (SCADA , 2018). As a result, a column called "Target" was added to the dataset, with "0" rows representing traffic data and "1" rows representing attack traffic (SCADA , 2018).

Sport	TotPkts	TotBytes	SrcPkts	DstPkts	SrcBytes	Target
61842	20	1276	10	10	644	0
61843	20	1276	10	10	644	0
61844	20	1276	10	10	644	0
61840	20	1276	10	10	644	0
61845	20	1276	10	10	644	0
61846	20	1276	10	10	644	0
44287	6	372	4	2	248	1
48456	20	1282	12	8	776	1
48458	20	1390	12	8	782	1
48460	20	1282	12	8	776	1
61850	12	780	6	6	396	0

Figure 14 Final Dataset

2.3 SIMILAR PROJECTS

2.3.1 HONEYCOMB

Overview of the Executed Study

Honeycomb is a system that simply generates signatures for malicious network traffic (Kreibich & Crowcroft, 2003). The system captures the traffic collected on the honeypot and uses methods for pattern identification and generating signatures for Snort and Bro (Kreibich & Crowcroft, 2003). It is built by a subsystem that monitors traffic inside the honeypot at various levels in the protocol hierarchy as an extension of the open-source Honeyd (Kreibich & Crowcroft, 2003).

The analysis of the study in contrast to the final year project is provided in the appendix: HONEYCOMB

2.3.2 SPECTER

Overview of the Executed Study

Specter is a commercial production honeypot that runs as an intrusion detection system. It is capable of attracting hackers away from real production machines. Specter can simulate 14 different operating systems, which are the most common in the commercial and research fields. Specter delivers on normal protocols that seem normal for attackers, and logs all the traces left behind by them, and notify the security individuals (Sardana & Joshi, 2011).

The analysis of the study in contrast to the final year project is provided in the appendix: SPECTER

2.3.3 'AN INTELLIGENT HONEYPOT' – AIDEN MITCHELL

Overview of the Executive Study

The project presented by (Mitchell, 2018), includes study and research of honeypot and development of python file that acts as a honeypot and has the ability to take on various profiles and simulations. It entails creating a virtual machine on a Linux server to host the file. It uses network monitoring tools like Tshark on the server to look for any suspicious attacks on the device that can be used as honey analysis (Mitchell, 2018). Network security is the primary focus of this intelligent honeypot. The script's potential to track ports and avoid attacks is the reason for this. Another important aspect of this honeypot is machine stability, as it will aid in the prevention of potential targeted hackers on a machine.

The analysis of the study in contrast to the final year project is provided in the appendix: AN INTELLIGENT HONEYPOT

2.4 SIMILAR STUDIES

2.4.1 'CONTEXT-AWARE HONEYPOT'- SHUBHAM AGGARWAL

Overview of the Executive Study

This research paper published in 2019, mainly focuses on improving the traditional intrusion detection system by applying well-known machine learning algorithms. After effective detection, it works to prevent the intruder from entering the network. To neutralize the attack, it implements an intrusion prevention method that generates a honey farm and tries to collect quality payloads from the intruders so that we can prepare for future cross-site scripting attacks. The framework is hosted in the virtual machine which uses Kali Linux 2019.1 on Virtual Box 5.2.22. The context switch is present in the virtual machine as well which makes the honeypot easier to manage (Aggarwal, 2019).

The analysis of the study in contrast to the final year project is provided in the appendix: CONTEXT-AWARE HONEYPOT

2.4.2 'DDOS MITIGATION AND INTRUSION PREVENTION USING HONEYPOTS'- MD. M. RAHMAN, S. ROY AND M. A. YOUSUF

Overview of the Executed Study

Md. Mahmudur Rahman, Shanto Roy, and Mohammad Abu Yousuf suggested a honeypot model to block DoS attacks in Content Delivery Networks. It also gathers information about the intruder to detect potential attacks. For counteractive action of DDoS attacks on a CDN server by Distributed Virtual Honeypot, a mitigation technique is exhibited because the proposed framework does not impact the operation of legitimate users, effectively blocks packets from attack sources by watching their network traffic and sends those to the honeypot for greater assurance where it analyses the packet whether they are false positive or not. It is an impeccable strategy for protecting the bandwidth of actual users and blocking fake users (Rahman, et al., 2019).

The analysis of the study in contrast to the final year project is provided in the appendix: CONTENT DELIVERY NETWORKS USING HONEYPOTS

2.5 COMPARISONS

Project Names	Specifications	Degree of involvement	Risk	Action	Cost	Log file	Database	UI	Detect Anomaly	Alert Notification
Honeycomb	Open source tool, generates signatures for malicious traffic	Mid	Mid	No	Low	Yes	Yes	Yes	No	Yes
Specter	Commercial Production tool, Incoperates commercial Intrusion Detection System	Mid	High	Yes	$599-Light Version/ $899-Full Version	Yes	No	Yes	No	Yes
An Intelligent Honeypot	Python script based research honeypot, incoperates Tshark	Low	Low	No	Low	No	No	No	No	No
Context-Aware Honeypot for Cross-Site Scripting attacks using Machine Learning Techniques	Detect Cross-Site Scripting, incorperates Machine Learning for analysis	High	High	No	Low	Yes	Yes	No	No	No
DDoS Mitigation and Intrusion Prevention in Content Delivery Networks using Distributed Virtual Honeypots'	Capture and analyze DDOS traffic in Content Delivery Networks, incoperates IPS	High	High	Yes	Low	Yes	Yes	No	No	No
This project	Serverside Honeypot, incoperates machine learning, detailed research, scabale to add whiteiplist, add protocols through Python Interface	Low	Low	Yes	Low	Yes	Yes	Yes	Yes	Yes

Figure 15 Comparison Table

From the above comparison table, we can conclude that among the mentioned projects and systems it shows that this project is one of the most innovative system with almost all the minimal features required for a successful honeypot to launch in the network. The system mimics a protocol for an intruder to interact with; all the honeypot needs is a connection from a suspicious network, after which it executes action such as blacklisting it. The interaction between the user and the honeypot is kept low which helps to reduce risk and can be used by users who have less cybersecurity knowledge as well. This system is an open-source tool so users are free to use and are free to implement different framework in the system. This project is a platform independent framework that runs on Windows and Linux OS. This system supports log file and also notifies the admin if there is any intrusion in the network. A special user interface is designed for visualization and accuracy prediction of the dataset for anomaly detection. Therefore, though limited services are

provided in the project it ensures and satisfies the need for a safe and easy deployment of the system by any field users all around the world.

CHAPTER 3: DEVELOPMENT

Companies use a number of methodologies to improve business profit in order to optimize productivity. Since the work management of one organization differs from that of another, the software may have a separate business process as a result of the differences in management. The term "software development" refers to the process of introducing new software or improving existing software (Permana, 2015). IT businesses should be able to forecast well maintenance as a result of technical advances. Centred on the Software Development Life Cycle (SDLC), the software development approach is used in conjunction with the company's needs (Permana, 2015).

3.1 CONSIDERED METHODOLOGIES

There are various software development methodologies available according to the requirements and goals of the project. The considered three methodologies for this project would be:

3.1.1 SCRUM METHODOLOGY

Agile Scrum methodology is a project management framework that focuses on incremental development (Business News Daily , 2020). Scrum Methodology is generally performed between small teams but, it can also be performed for individual projects. Three positions are described by Scrum: Scrum Master, Product Owner, and Development Team. A Scrum Team is made up of all three positions (Scrum Alliance , 2021). It is a fast, flexible, adaptable, and efficient agile framework that is designed to deliver high software quality throughout the project development phase (Digité, Inc, 2020). A representation of the Scrum Methodology is shown below-

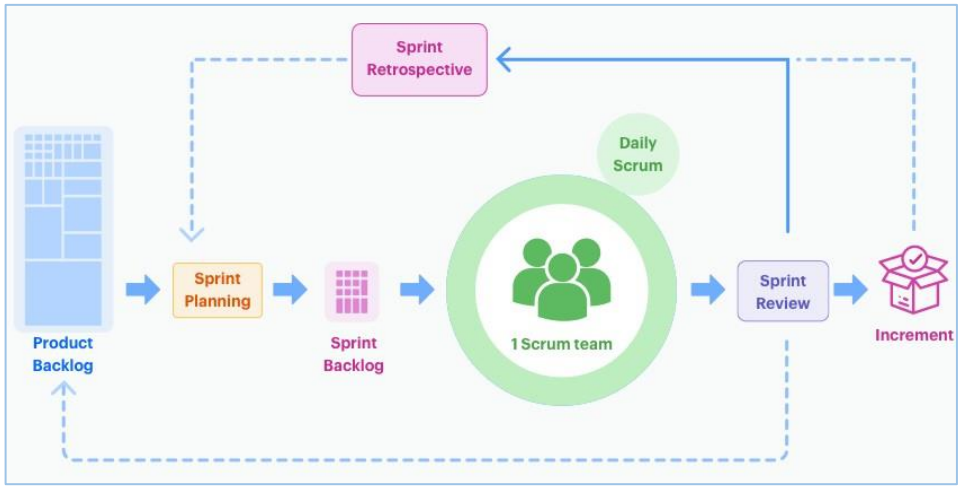


Figure 16 Scrum Methodology (Inc, 2020)

The remaining explanation is provided in the appendix: SCRUM METHODOLOGY

3.1.2 ITERATIVE MODEL

The iterative approach starts with a basic implementation of a subset of software specifications and iteratively develops the expanding versions until the complete framework is deployed (tutorialspoint, 2020). Design modifications are implemented at every iteration and different functional capabilities are applied. The idea is to build a scheme at a time (incremental) by iterative cycles and in minor parts. Thorough testing of requirements and verification & evaluation of each version of the software against those requirements during each model cycle is the key to the efficient use of an iterative software development lifecycle (tutorialspoint, 2020).

A representation of the Iterative model is shown below-

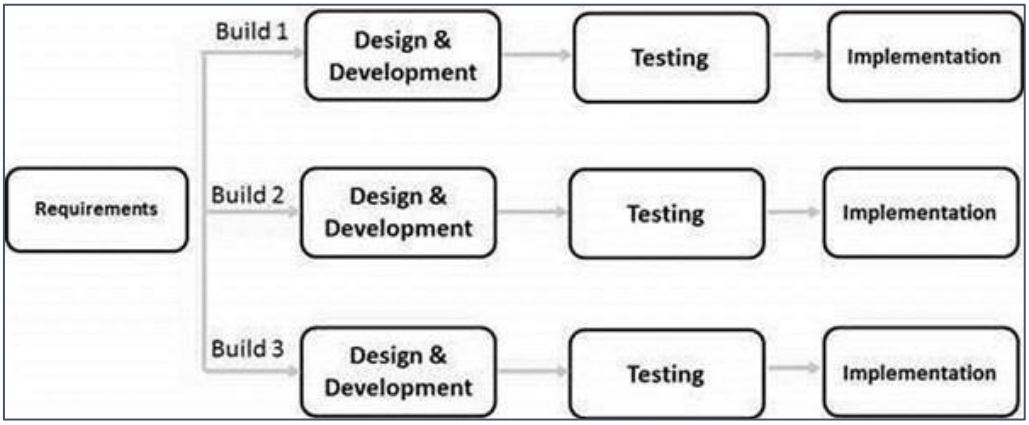


Figure 17 Iterative Model (tutorialspoint, 2020)

The remaining explanation is provided in the appendix: ITERATIVE MODEL

3.1.3 KANBAN MODEL

Kanban is a visual framework for handling work as a process passes through it (Answers Course, 2021). Kanban visualizes both the procedure and the actual work that passes through that method. Kanban aims to find and address possible bottlenecks so that work can flow cost-effectively at an optimum pace or throughput through it (Answers Course, 2021). A range of principles and procedures for handling and optimizing workflow is implemented by the Kanban System. It is an evolutionary, non-disruptive strategy that facilitates incremental changes to the processes of an organization (Digite, 2020). Kanban has proven to be an effective method to minimize waste of energy and to maximize material flow (Rajat B. Wakode, 2015). Overall, Kanban embraces all the

ideals of the Agile Manifesto and helps you deliver goods and services that are needed in the market.

The information of visualizing Kanban is provided in the appendix: KANBAN MODEL

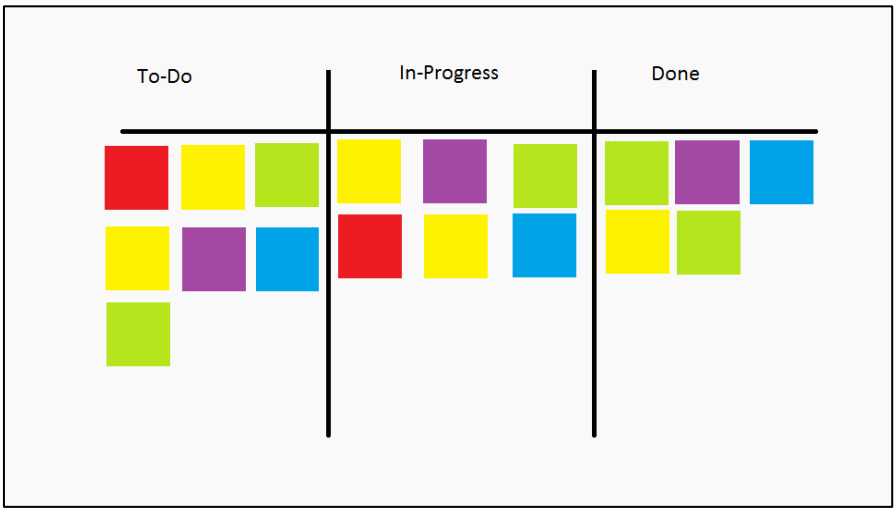


Figure 18 Basic Kanban Board (Eisele, 2018)

3.2 SELECTED METHODOLOGY

The methodology selected for the development of this project is the Kanban Methodology. It is the best framework for identifying the potential for productivity enhancement by visually perceiving and monitoring tasks. Kanban integrates and supports change readily. Anything needed to be added or changed can be done within the WIP limits. A well-designed Kanban framework allows teams to recognize and reduce costs.

To support the reasons for not having chosen the other methodologies is that, they create difficulties during the development process. Firstly, in Scrum Methodology, it requires competent and high-level individuals and there is some chance of scope creep in the lack of experienced individuals Scrum phase. Kanban assists in visualizing the process, limiting work-in-progress (WIP), and moving work from "Doing" to "Done" quickly. Kanban is suitable for teams that receive a large number of requests of varying priority and duration. Kanban processes allow to go with the flow, while scrum processes need a lot of control over what is in reach (REHKOPF, 2021).

Similarly, in the Iterative Model, highly skilled resources are necessary for skill analysis and due to the lack of a complete requirements specification for the entire system, issues with system design could prove to be a critical constraint (Kienitz, 2017). Unlike an iterative model, there are no artificial iteration limits in a Kanban-style, lean development process. Every story leads to working software and, in the best case, a release. This involves a higher level of team self-discipline.

Therefore, Kanban Methodology was chosen as the development methodology as it meets the requirements and goals of the project.

The validation of the model is further explained in the appendix: VALIDATION OF THE METHODOLOGY

3.3 PHASES OF METHODOLGY

3.3.1 BACKLOG

A backlog is an arranged list of tasks that needs to be completed. The tasks needed to be accomplished for the development of this project will be arranged according to its priority in the backlog.

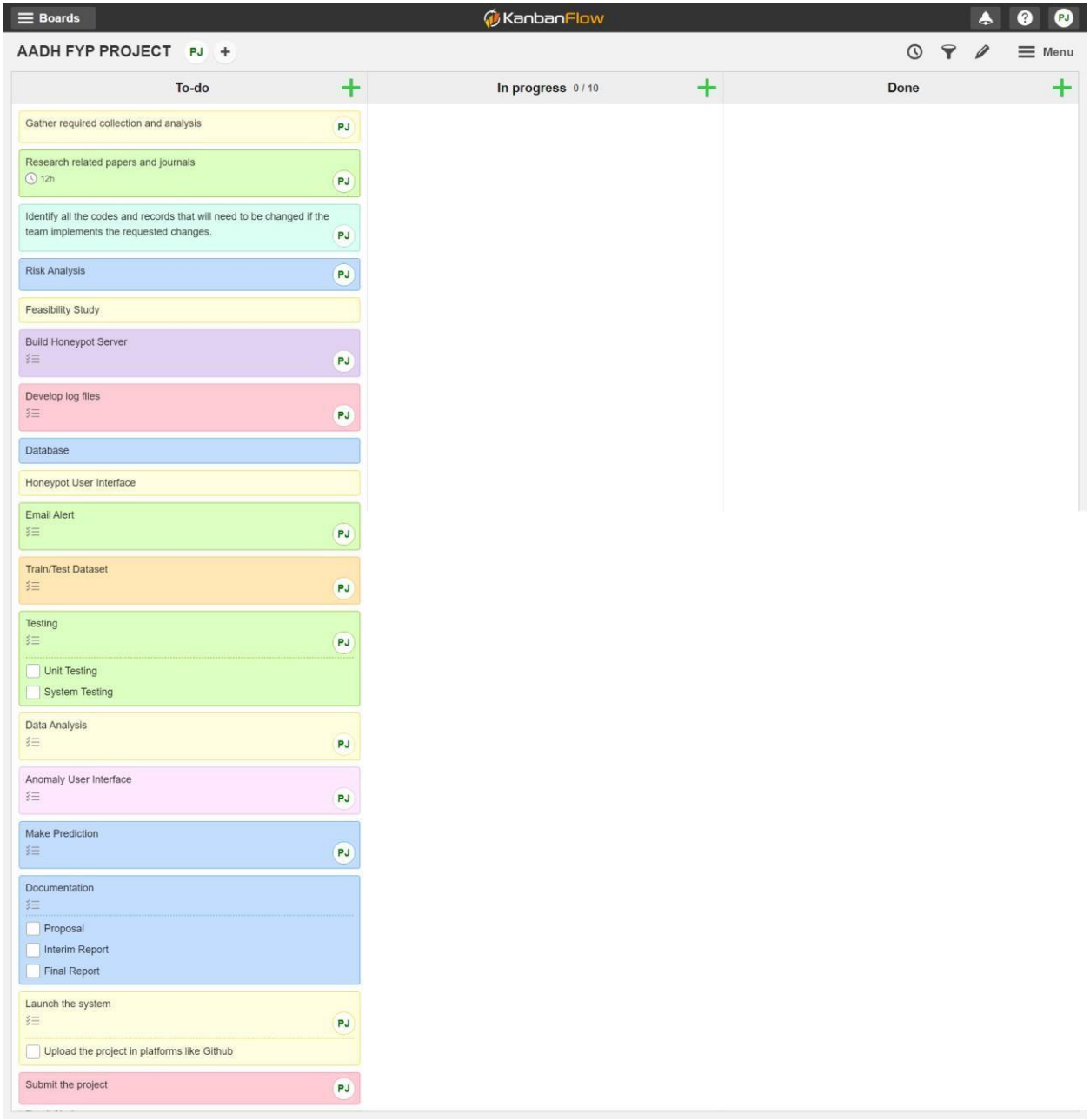


Figure 19 Backlog tasks in Kanban Board

A detailed explanation is provided in the appendix: BACKLOG

3.3.2 IMPACT ANALYSIS

The impact analysis stage consists of research and investigation of the impact and possibility of the project. In this stage, a research and feasibility study will be executed about this project. It examines the tasks' reach and impact. If a task may have a high-level effect on current goods or services, this may require a high-level impact statement (Leybourn, 2015). The aim is to look at the basis of a methodology for evaluating and assessing the effect of a change on the whole system, which involves not only source code but also the system's specification and design documents, as well as an early stage in the maintenance process (Richard J. Turver, 1994).

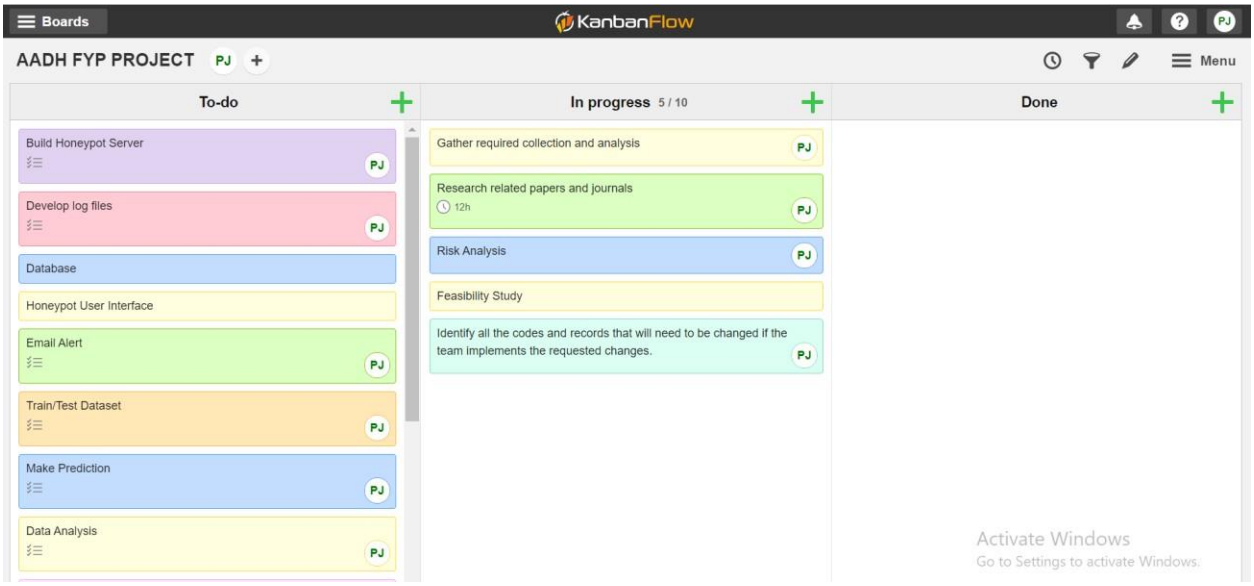


Figure 20 Impact Analysis in Kanban Board

A detailed explanation is provided in the appendix: IMPACT ANALYSIS

3.3.3 BUILD

The build stage is where the tasks are initiated heavily and the tasks are strongly worked on. In this stage, most of the tasks such as designing, scripting, collecting datasets, training machine learning, testing, and debugging will be performed.

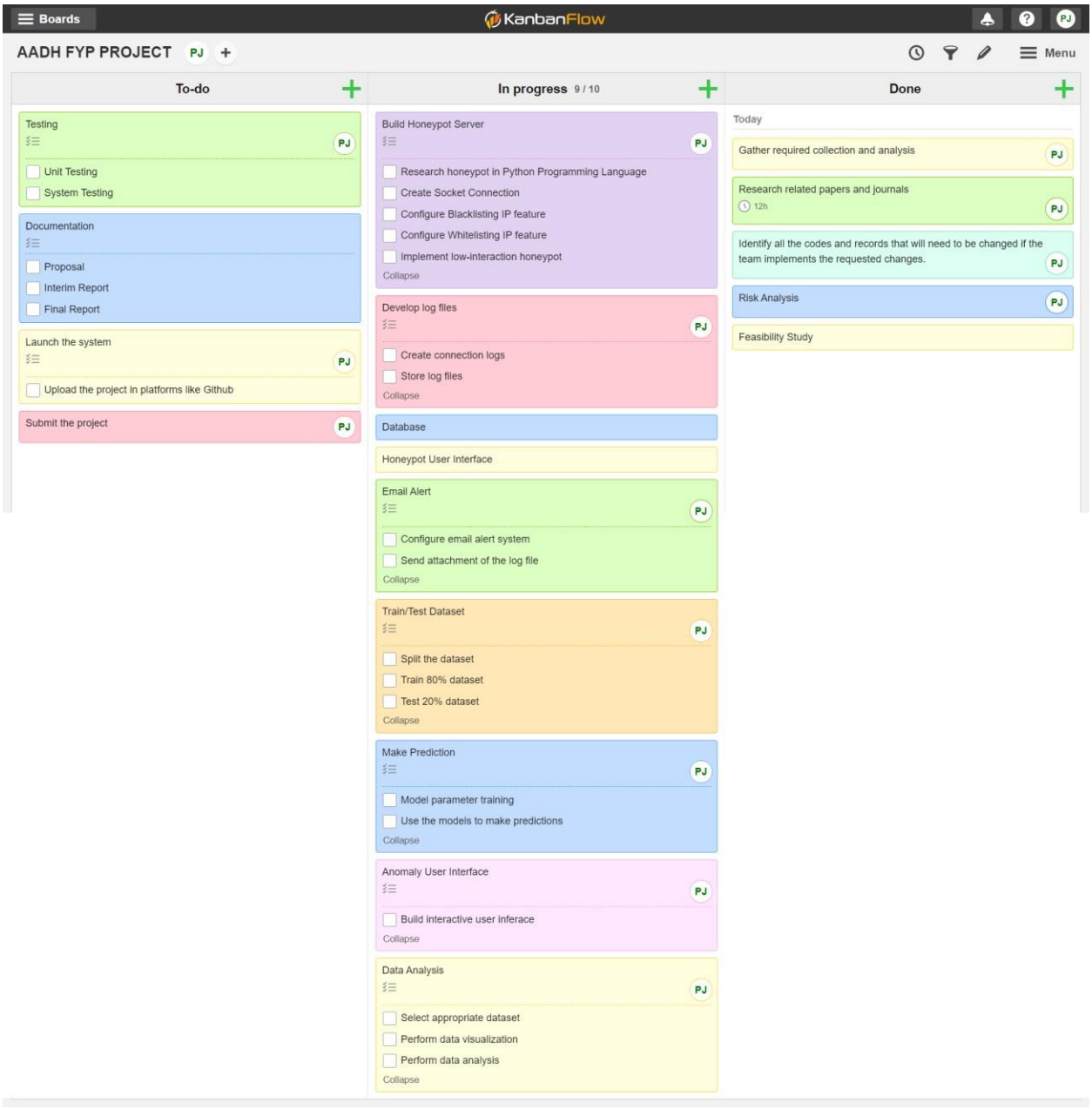


Figure 21 Build Tasks in Kanban Board

A detailed explanation is provided in the appendix: BUILD

3.3.4 USER ACCEPTANCE TESTING

User Acceptance Testing is a phase where the end-user performs testing to verify the software system. Here, this phase is skipped because the project will be accomplished from the base with help of numerous research and supervisor(s) feedback.

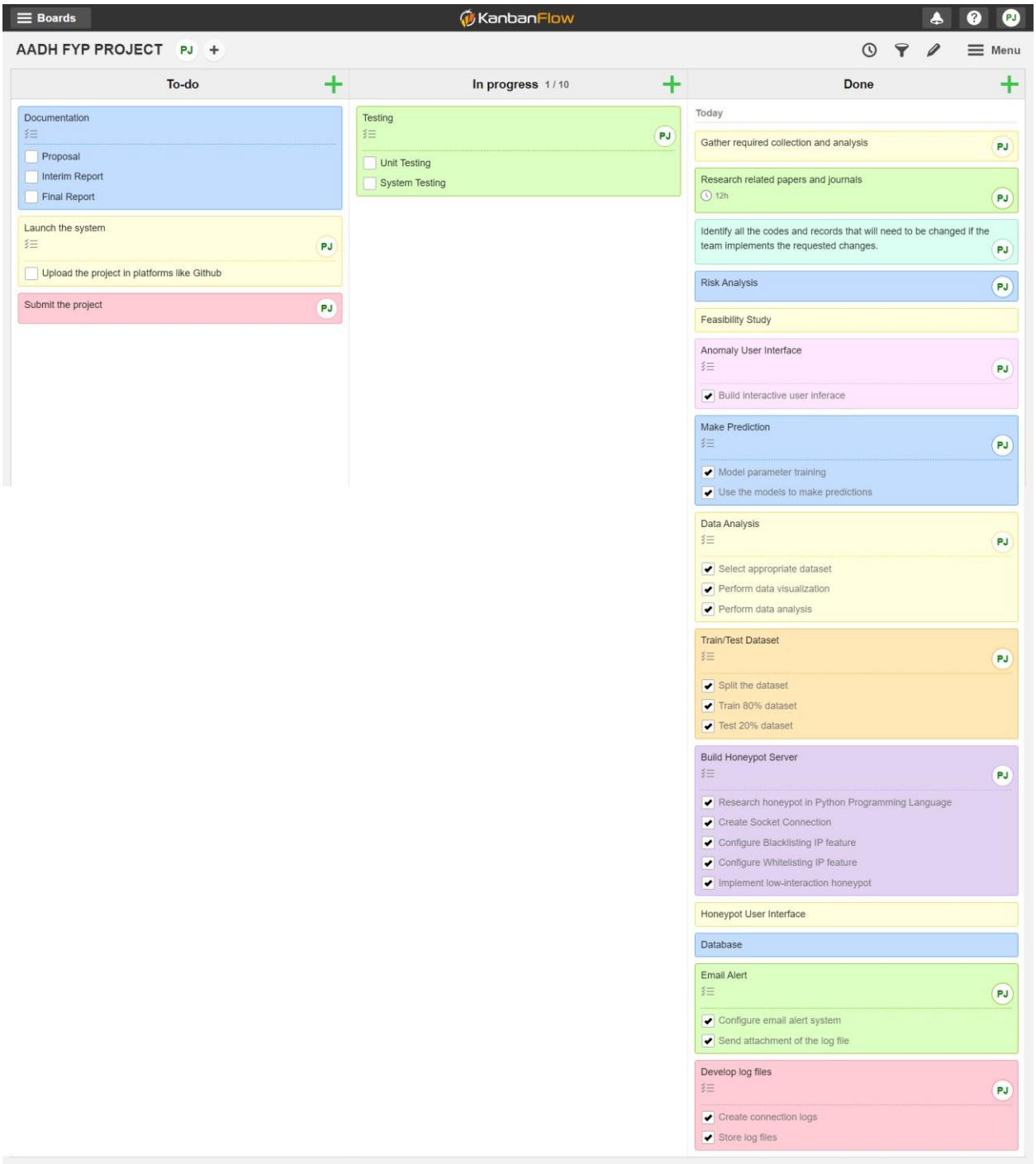


Figure 22 User Acceptance Task in Kanban Board

A detailed explanation is provided in the appendix: USER ACCEPTANCE TESTING

3.3.5 RELEASE

In Kanban Methodology, a release can signify a formal release to the partners or clients, or it could be used as a personal milestone. It can also be used to describe the whole product of a project with a single delivery at the finish (Scrumwise, 2021).

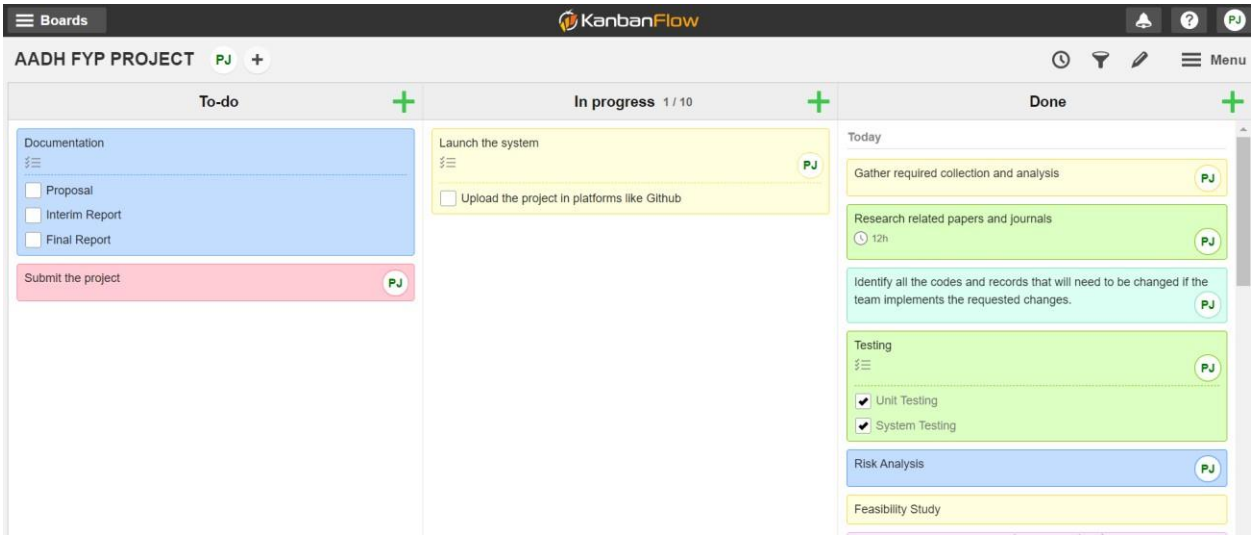


Figure 23 Release Tasks in Kanban Board

In the case of this project, after performing all the tasks the system is planned to be released as a whole. As the system is still in its early phase, there are still rooms left to be improved to be released as a commercial system. Therefore, after the project is fully completed and approved by the author's supervisor(s), the system will be released on platforms like GitHub to showcase the work and receive user feedbacks.

3.3.6 DOCUMENTATION

The documentation phase covers documents and materials which deal with the system's development and use. In this phase proposal, interim and final reports were documented and submitted before the deadline.

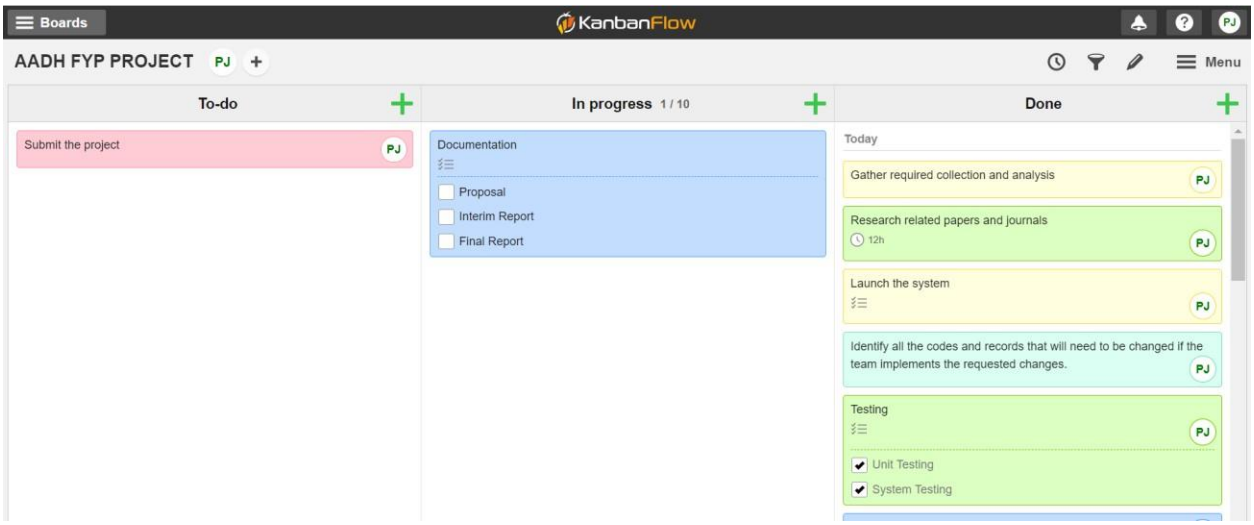


Figure 24 Documentation Tasks in Kanban Board

3.3.7 DONE

This is the final stage of the software development methodology. Hence, the project is considered as done if the system fulfills the intended outcome.

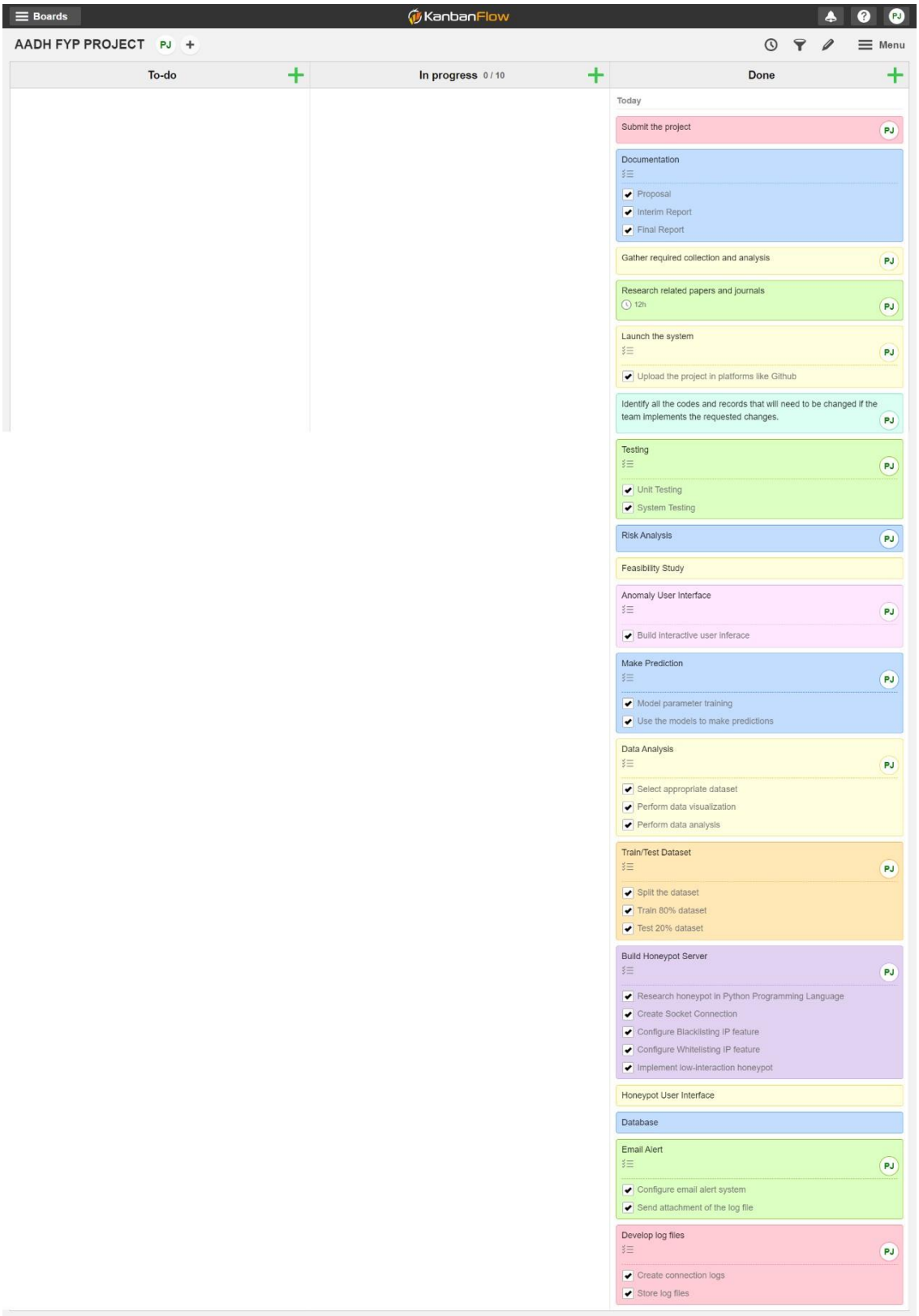


Figure 25 Completed Tasks in Kanban Board

3.4 SURVEY RESULTS

3.4.1 PRE-SURVEY RESULTS

During the early phases of the project developed a comprehensive survey was carried out between the professionals and consumers primarily end-users. The purpose of the survey was to understand the project demand, required features, and real-time evaluation from industry experts.

People browse the internet, play games online, shop online, etc. However, *how many individuals know about security when performing all these things? Do they know the risk of being cyber-attacked? How many users are interested in capturing them and finding network monitoring helpful?* Such views and experiences of individuals were very important for making the decisions in this project.

The Pre-Survey was conducted among 35 surveyees of different genders and work backgrounds. Among the potential end-users surveyed, figure 26 shows that 82.9% say they are aware of cyberattacks and 17.1% say they are not aware of them. Again, from figure 27 we can see that a large majority of end-users (73.4%) and conceivably 22.9% are interested in applying this proposed system in their environment which is quite impressive data.

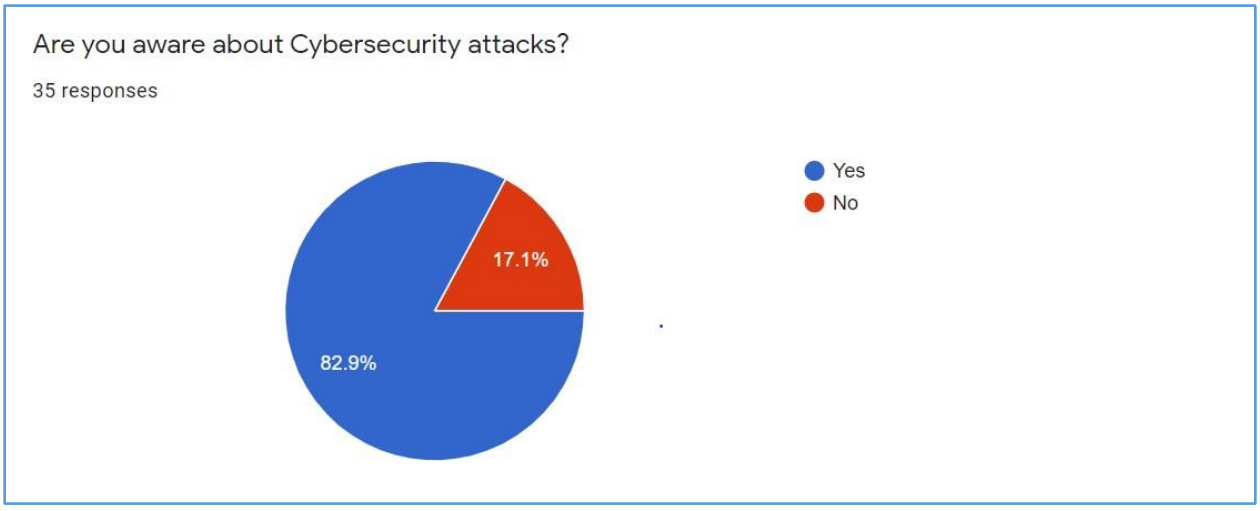


Figure 26 Awareness of Cybersecurity attacks

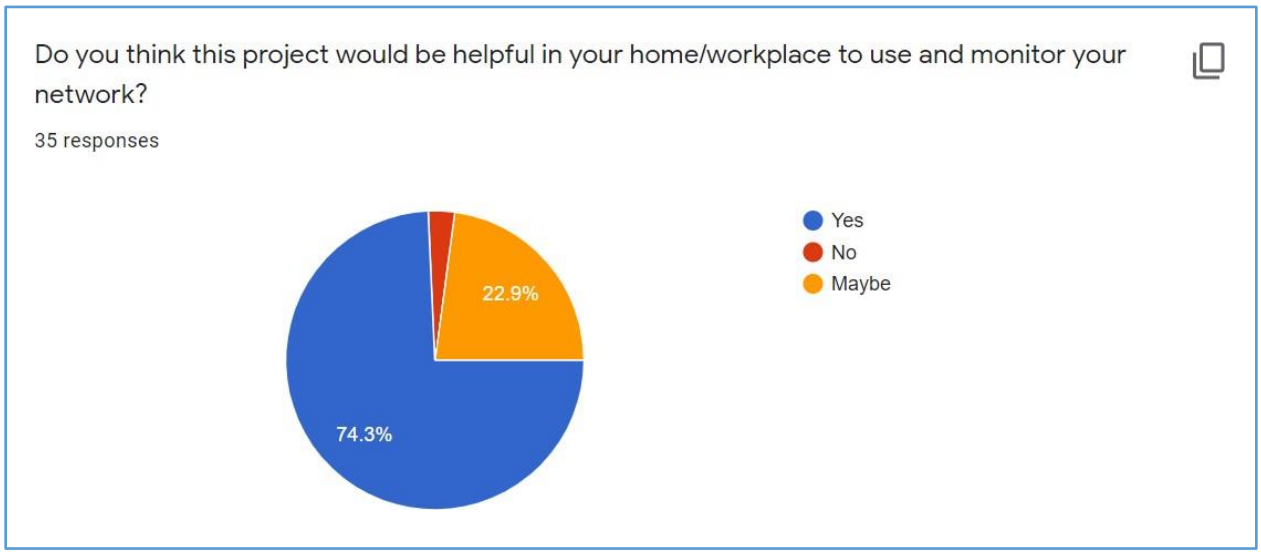


Figure 27 Necessity of the project in the community

Accordingly, to summarize the overall pre-survey responses, among the end-users surveyed they strongly believe that it is very crucial to have security in the network and are very much interested in monitoring and capturing real-time network security attacks but, they want such services at minimum cost and without much hassle. From the beginning of the project, the system was planned to be simple yet effective, thus it highly meets the user's expectations and needs in the market.

The rest of the survey results are given in the Appendix: PRE-SURVEY RESULT

3.4.2 POST-SURVEY RESULTS

One of the most critical aspects of a system's development is customer satisfaction. It's impossible to create good resources if the functional and/or non-functional properties don't meet the needs of the consumer. Therefore, information was gathered from a post-survey to extract some of the system's potential benefits and drawbacks, as well as possible add-ons in the future. The information for this project was gathered through a survey that was distributed among the IT students, Information Security professionals, and some general people. All of the future works will be wholly reliant on feedback and survey from the community.

The survey was conducted among 14 potential end-users by providing a beta version of the system. In figure 28, among various features in the system, "Data Capture" and "Visualization" were the

most liked features of the system. This has motivated the author to add more detailed data capture facilities and visualizations in the system in the future work of the project.

Overall, 64.3% of users gave 5/5 ratings, indicating that they were pleased with the functionality, which sounds incredibly amazing.

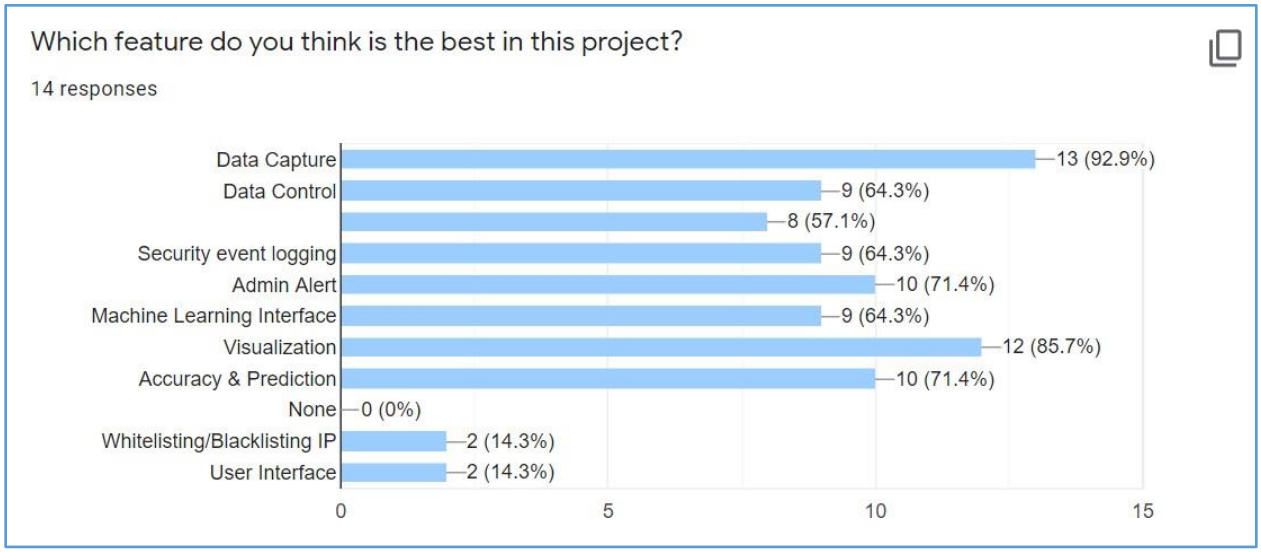


Figure 28 Best Feature in the project

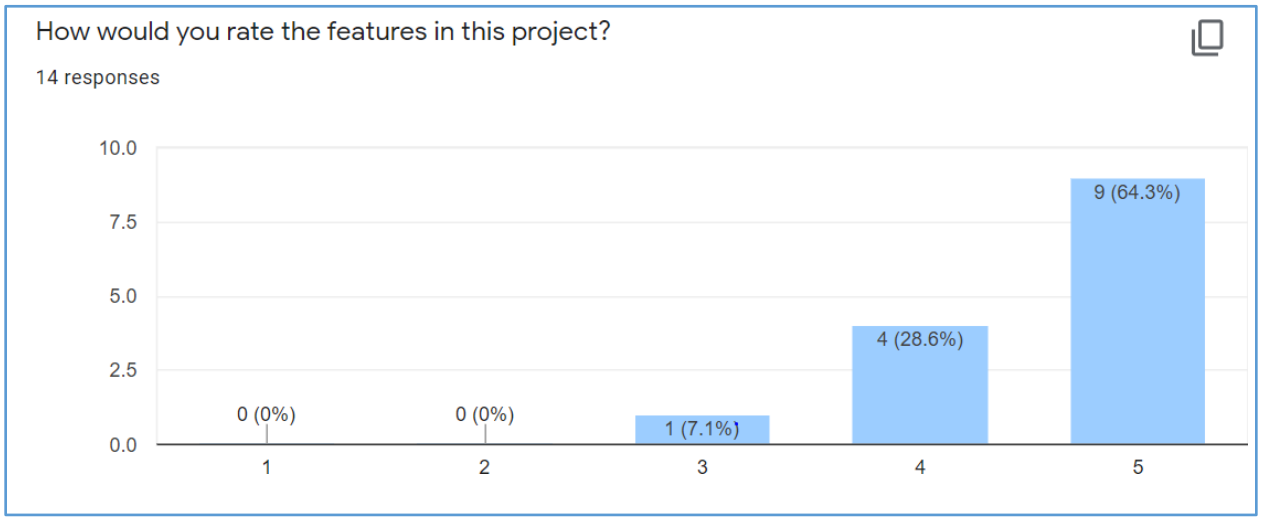


Figure 29 Rating of the features

Thus, highlighting the major responses received from the users; 85.7% are highly interested in using the system to gain information about various attacks and malware, and also 71.4% would like to perceive hackers in action to gain insight into their activities using this system. Similarly,

78.6% of the users strongly believe that this project will contribute to the cyber security domain presently.

The rest of the survey results are given in the Appendix: POST-SURVEY RESULT

3.5 REQUIREMENT ANALYSIS

Requirements analysis is a collaborative endeavour that necessitates a fusion of hardware, software, and human technical experience, as well as interpersonal skills (Visual Paradigm, 2020). The following are the key practices in requirement analysis:

- Determine the needs of the consumer
- Assess the system's viability
- Conduct economic and technological research
- Assign functions to system components
- Establish a timeframe and restrictions
- Make meanings for the scheme

3.5.1 PLANNING

During the planning process, project priorities will be decided and an elevated proposal will be developed. By term, planning is a crucial and key organizational process. The following are the three main tasks involved in the planning phase:

3.5.1.1 IDENTIFICATION OF INTERFACES AND CONSTRAINTS

- Major quantities of hand-crafted, organized training data are needed.
- The system primarily necessitates administrator privileges are required to the user system.
- The IP Addresses, Protocols, Email Address, and password needs to be manually configured.
- The ethical and legal aspects must be recognized.

3.5.1.2 REVIEW OF PROJECT SCOPE

- The objective of this study is to study the emergent trends in extant honeypot research to contribute to the knowledge gaps in the honeypot environment by integrating machine learning (Campbell & Padayachee, 2015).
- The timeframe is the most impactful constraint in the project, followed by the system's validation by users.
- The Kanban Model is the software methodology approach used in the project.
- The project will be published under the GNU GPLv3 license providing a framework for the implementation and development of services.

3.5.1.3 PLANNING THE EXECUTION OF THE PROCESS

- The most important requirement of collection tool in this process would be an online mass survey and feedbacks from supervisors.
- Beta user feedback (Post-survey) is gathered during this process, with the ultimate purpose of extracting some of the system's possible benefits and disadvantages, as well as possible add-ons.
- A separate study initialization is carried out solely to develop a useful questionnaire.

The project software and hardware requirement specifications are discussed in the appendix: REQUIREMENTS ANALYSIS

3.6 DESIGN

3.6.1 FLOWCHART

3.6.1.1 PRIMARY COMPONENT

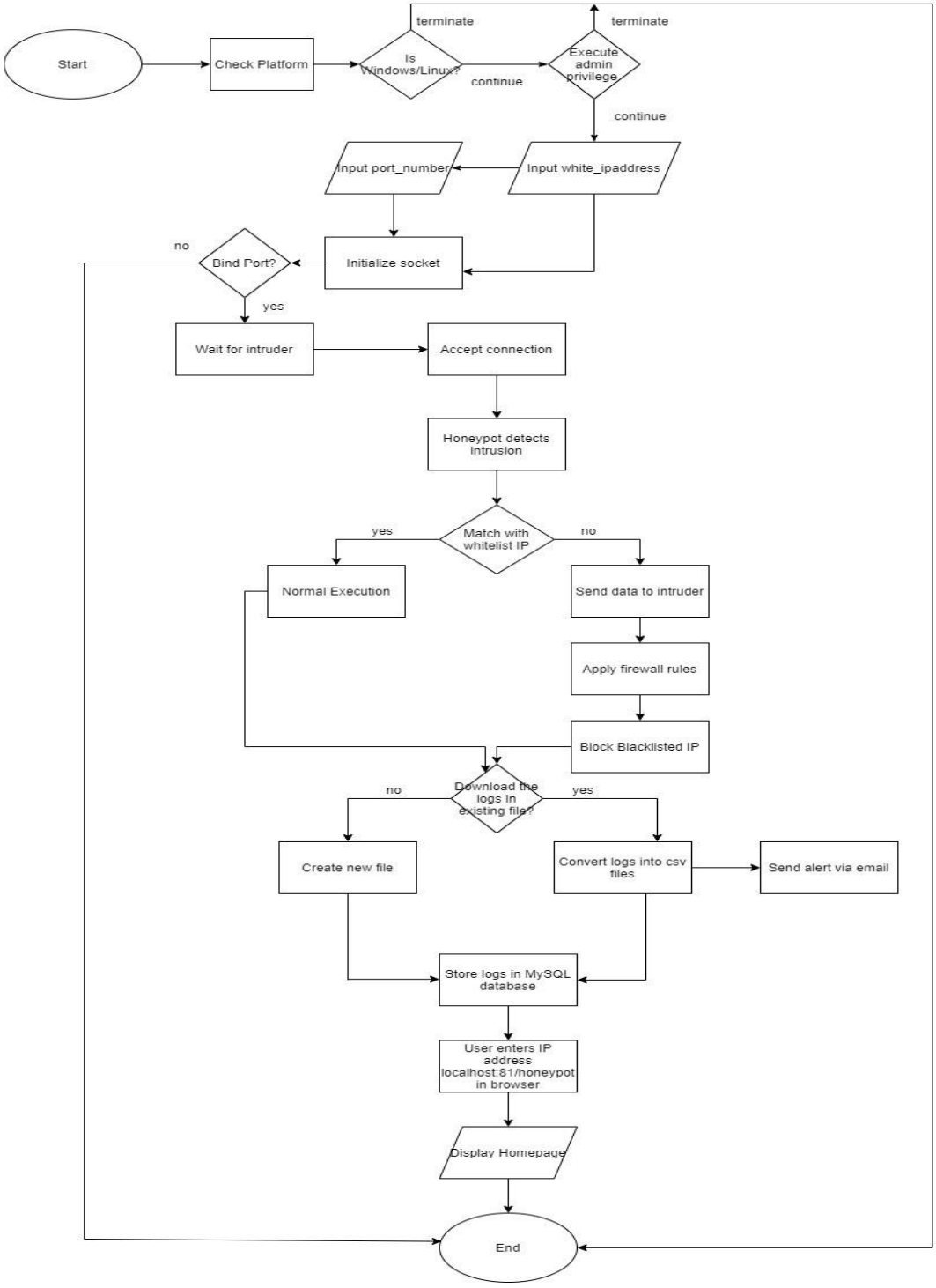


Figure 30 Flowchart of Primary Component

3.6.1.2 SECONDARY COMPONENT

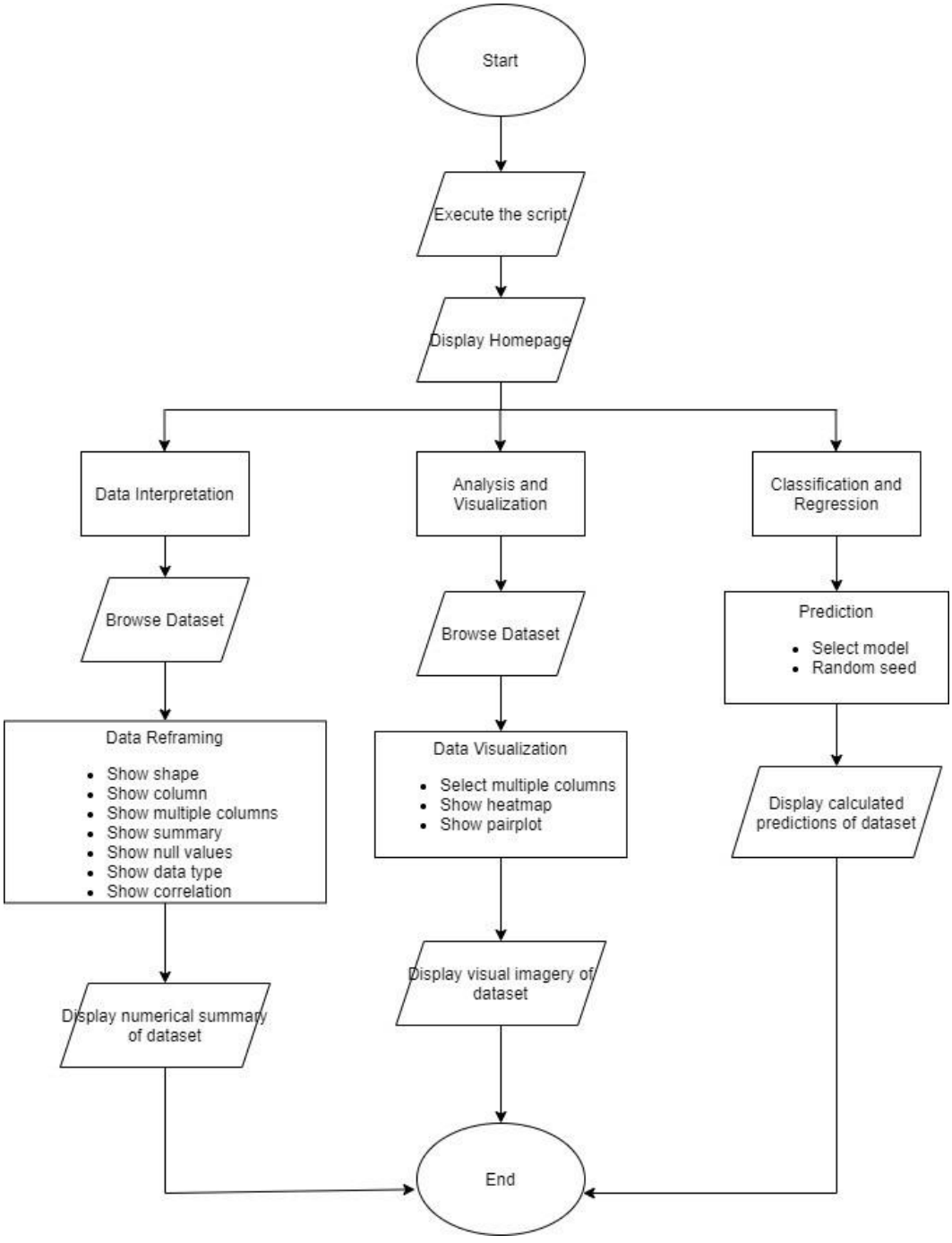


Figure 31 Flowchart of Secondary Component

3.6.1.3 COMBINED FLOWCHART

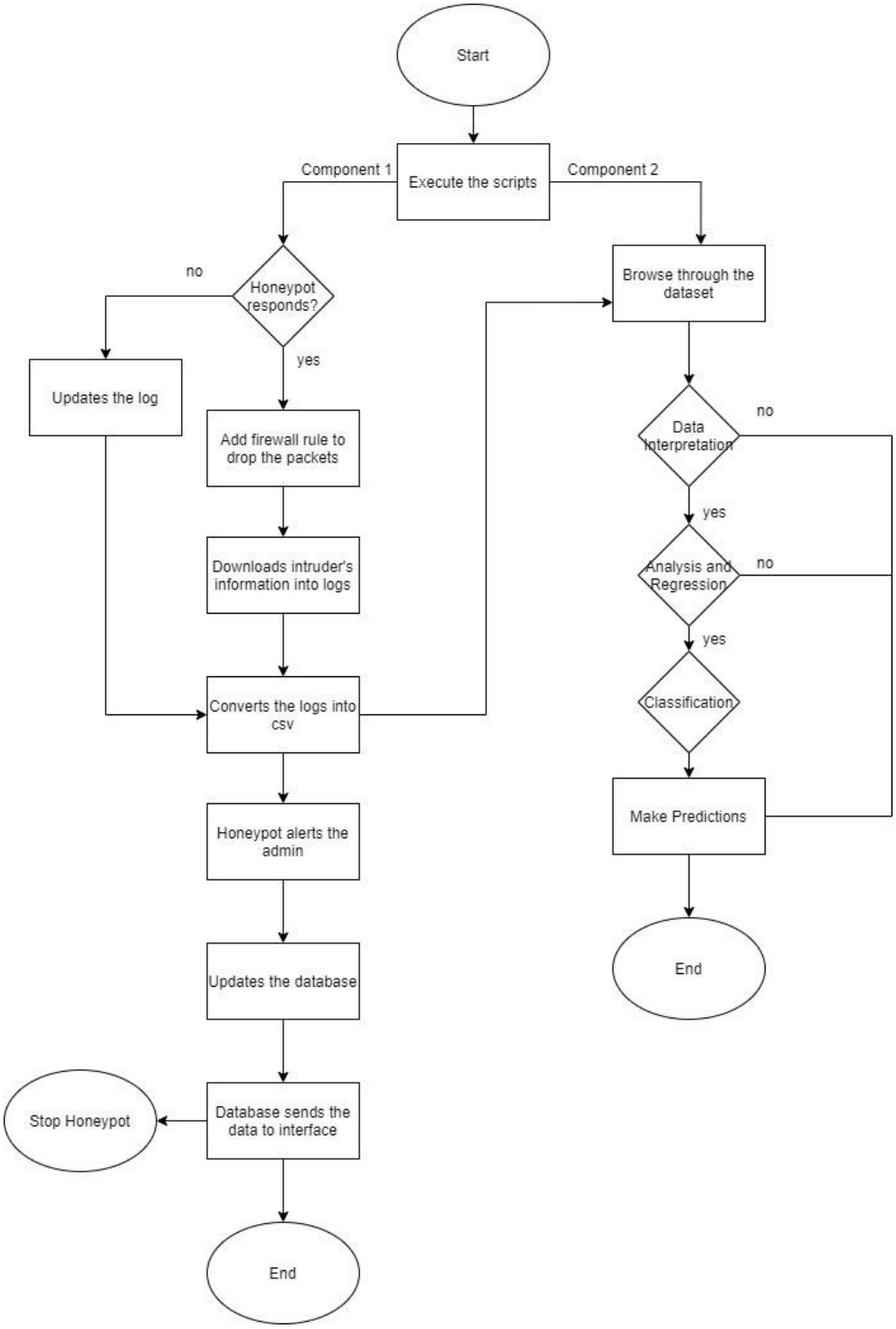


Figure 32 Combined Flowchart

3.6.2 SEQUENCE DIAGRAM

3.6.2.1 PRIMARY COMPONENT

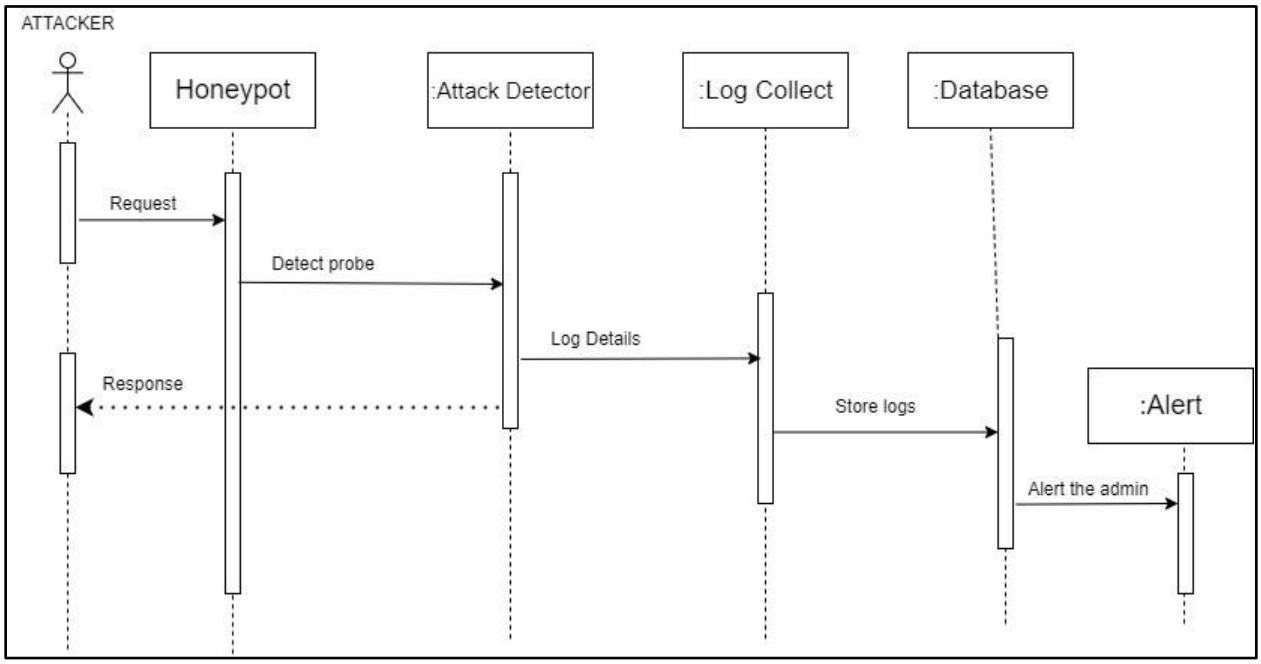


Figure 33 Sequence diagram of primary component

3.6.2.1 SEQUENCE DIAGRAM OF SECONDARY COMPONENT

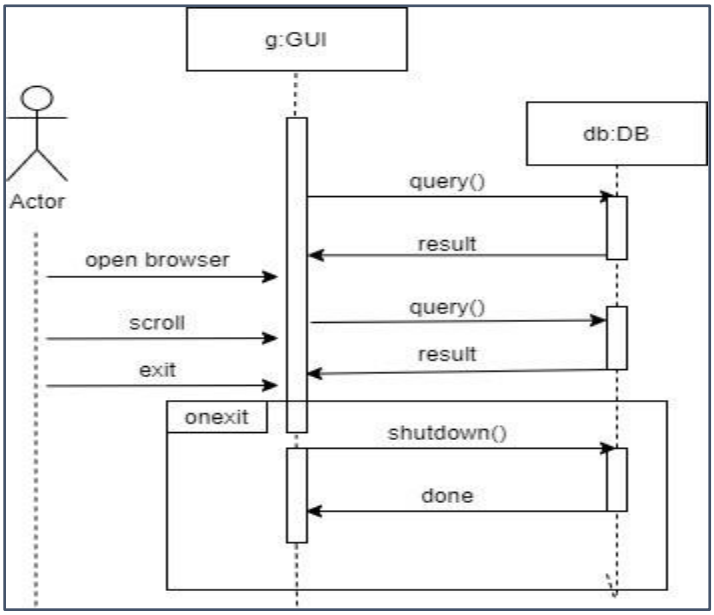


Figure 34 Sequence diagram of secondary component

3.6.3 BLOCK DIAGRAM

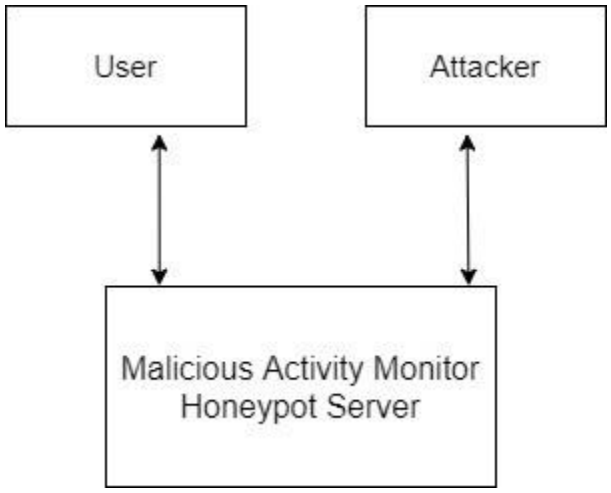


Figure 35 Block Diagram of Primary Component

3.6.4 USE CASE DIAGRAM

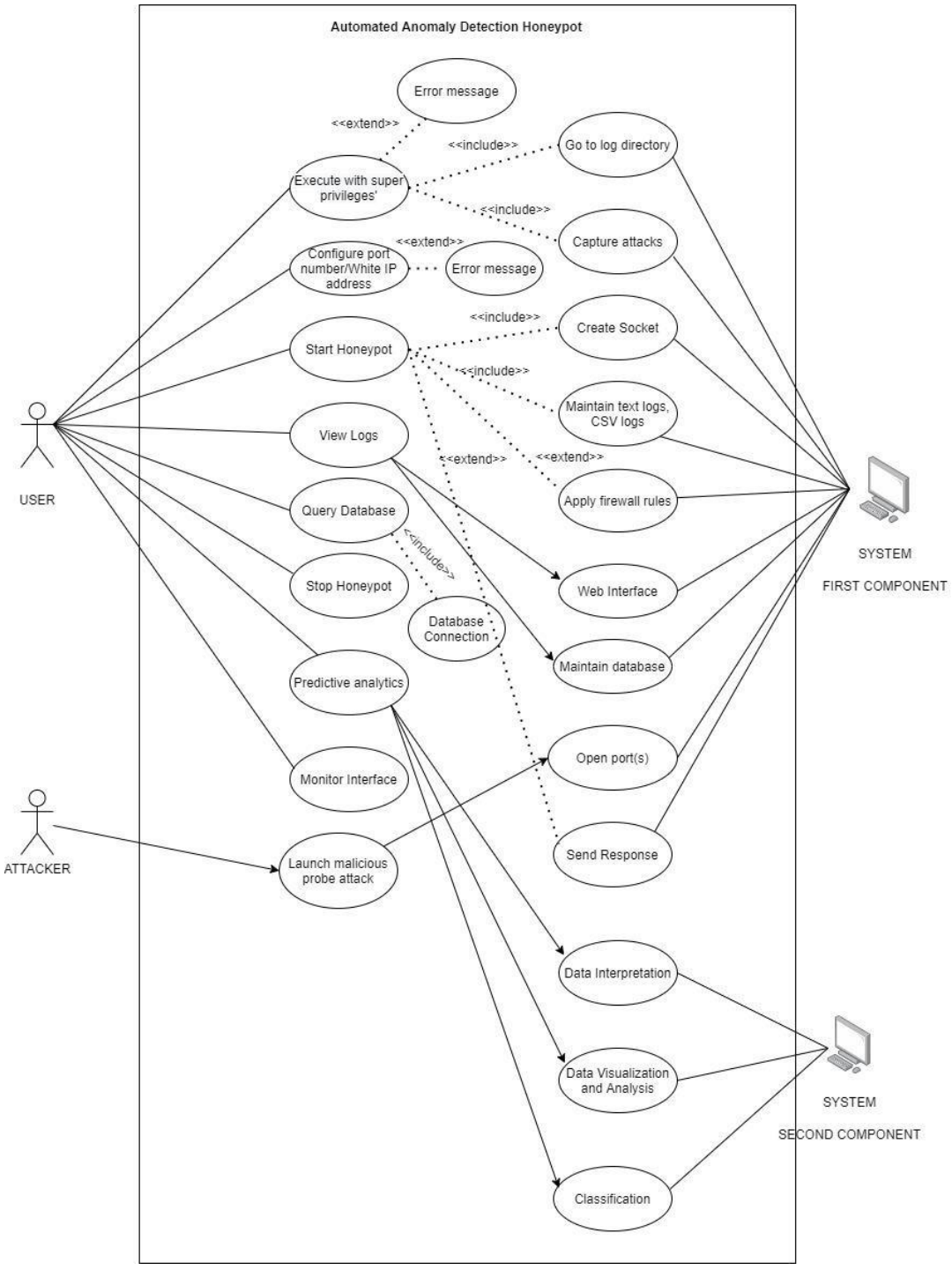


Figure 36 Use Case Diagram of the system

The description of Use Case is provided in the appendix: USE CASE DESCRIPTION

3.6.5 ENTITY RELATION DIAGRAM

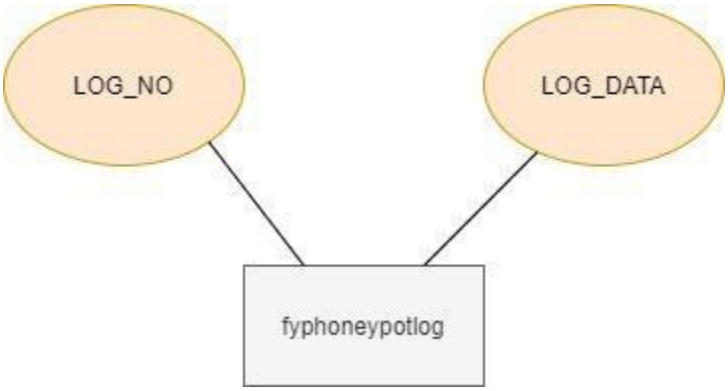


Figure 37 Entity Relation Diagram

The fyphoneypotlog module helps to know the detailed information about the logs created by the honeypot. The properties of this entity are:

- Log_no: Number of the logs to uniquely identify them.
- Log_data: The log data consists of the information of the attack, timestamp, and platform information.

3.6.6 RELATIONAL DIAGRAM

Figure 38 Relational Diagram

There is only one relational table formed in the honeypot database. The log no is the primary key that is stored in the integer variable and log data is stored in the varchar variable. The primary purpose of creating a database table was to store and retrieve the data. The database serves as a backup of the data that can be extended if required. The same data would be sent to the interface, which aids in data management.

3.6.7 DATA DICTIONARY

Entity Name	Entity Description	Column Name	Column Description	Data Type	Length	Primary Key	Foreign Key	Unique	Notes
fyphoneypotlog	fyphoneypotlog is the detailed information about the logs created by the honeypot	LOG_NO	Number of the logs to uniquely identify them.	INT		TRUE	FALSE	TRUE	Auto increment
		LOG_DATA	The information of the attack, timestamp, and platform information.	VARCHAR	600	FALSE	FALSE	FALSE	

Figure 39 Data Dictionary

3.7 IMPLEMENTATION

For the development of this project, the author has strictly followed Kanban methodology and this section mainly focuses on the built phase of the methodology. Kanban software development methodology differs from other methodologies like Scrum or Waterfall planning. The Kanban team only "appoints" items to be focused on and ensures that they are put in the proper column and place. The upper Kanban cards in a column are usually more critical, and the team should take the first one when a new card is pulled (Kanbanize, 2021). Therefore, the same perquisite is taken into considerations and followed during the development of this project. The visual representation of the Kanban board can be viewed from here: PHASES OF METHODOLOGY

A detailed explanation of implementation involves a brief description of the core codes of the system, which is provided clearly in the appendix: IMPLEMENTATION

CHAPTER 4: TESTING AND ANALYSIS

The aim of testing this system is not only to find bugs or explain their significance. It is to lower the risk by proactively identifying and assisting with the resolution of issues that will have the greatest effect on the software's consumer.

Here, the honeypot system and machine learning interface have been broken down into two components so, that testing can be done evidently and efficiently. In the case of both of the systems, the development's testing is usually conducted on an outcome basis, and depending on the situation, white box and black box testing are performed separately.

"The 'what if" should be the leading question of the software research"-A1QA (ALTVATER, 2017)

4.1 TEST PLAN

4.1.1 UNIT TESTING, TEST PLAN

Table 1 Unit Testing

Test case	Objectives
Honeypot System	
1	To test whether the system runs without admin/root rights.
2	To test string input in place of protocol number.
3	To test protocol configuration outside the given port range.
4	To test whether the system can identify whitelist IP address.
5	To test whether the system can identify connection from external IP address and perform the action of blacklisting them.
6	To test whether the rules implemented in Test Case 5 are applied in Windows and Linux Firewall.

7	To test the server's response to the attacker after the intruder tries to connect through IP address.
8	To test to unblock the firewall rule feature in Windows and Linux Firewall.
9	To test the ability to store logs in text files and csv files.
10	To test the facility to collect log files and upload in the database.
11	To test email alert.
Anomaly Detection System	
12	To test whether user can browse unacceptable file format from the computer.

4.1.2 SYSTEM TESTING, TEST PLAN

Table 2 System Testing

Test Case	Objectives
Platform Independent	
1	To run the honeypot system in Windows Platform.
2	To run the honeypot system in Linux Platform.
Honeypot UI	
3	To test the ability to view the home interface of logs recorded, easily in the web interface.
4	To test the ability to view the blacklisted logs recorded, easily in the web interface.
5	To test the ability to view the whitelisted logs recorded, easily in the web interface.
Anomaly Detection UI	
6	To test the ability to view and use machine learning for data visualization and classification of dataset, easily in the web interface.
7	To test whether user can browse dataset file from the computer.
8	To test the UI check boxes present in the "Data Interpretation" section.
9	To test the UI check boxes present in the "Analysis and Visualization" section.
10	To evaluate Logistic Regression model through accuracy metrics with low random seed and trainable parameters.

4.2 UNIT TESTING

4.2.1 HONEYPOT SYSTEM

For the first testing case, the system is executed in the Windows and Linux environments without any administrative rights. Therefore, an error message is displayed in the log. Administrative or Root privilege is set to be required for every user compulsorily to preclude the system, either deliberately or accidentally from being disrupted.

4.2.1.1 TEST CASE 1

Table 3 Honeypot System Test Case 1

Test Case 1	
Objective	To test whether the system runs without admin/root rights.
Action	Run the system without admin rights in Windows and Linux Platforms.
Expected Test Result	The system doesn't perform any tasks and error is expected; as executed without admin rights in both platforms.
Actual Test Result	Error message is displayed.
Conclusion	Test Successful.

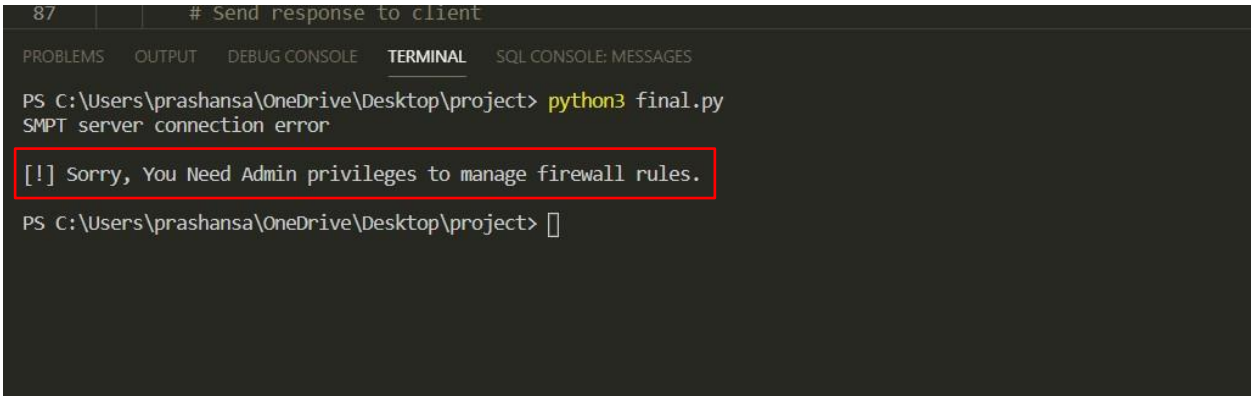


Figure 40 Test Results of Test Case 1 checking admin rights in Windows

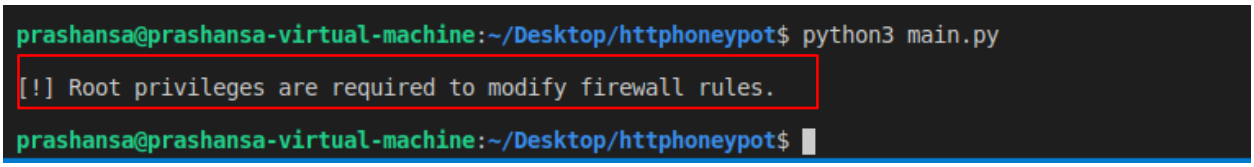


Figure 41 Test Results of Test Case 1 checking admin rights in Linux

4.2.1.2 TEST CASE 2

Table 4 Honeypot Module Test Case 2

Test Case 2	
Objective	To test string input in place of protocol number.
Action	Execute the program with a random string 'gvjg'.
Expected Test Result	The system fails to perform the specific task in the port.
Actual Test Result	Error message is displayed.
Conclusion	Test Successful.

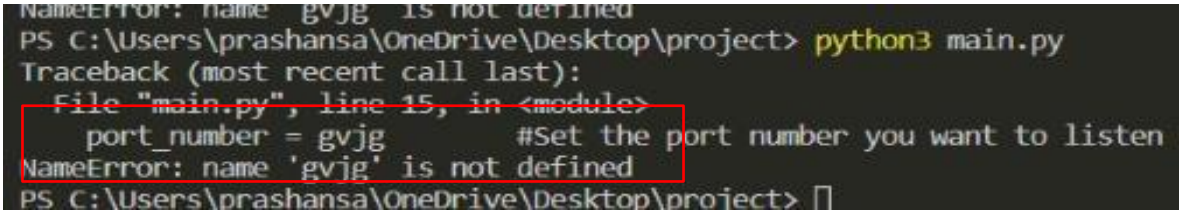


Figure 42 Test Results of Test Case 2 checking input value

4.2.1.3 TEST CASE 3

Table 5 Honeypot Module Test Case 3

Test Case 3	
Objective	To test protocol configuration outside the given port range.
Action	Execute the program outside the given port range.
Expected Test Result	The system fails to perform the specific task in the port.
Actual Test Result	Error message is displayed.
Conclusion	Test Successful.

Now, the system is set to be configured outside the port range from 0 – 65353. If any user sets the port range below or above the standard port range, it displays an error message.

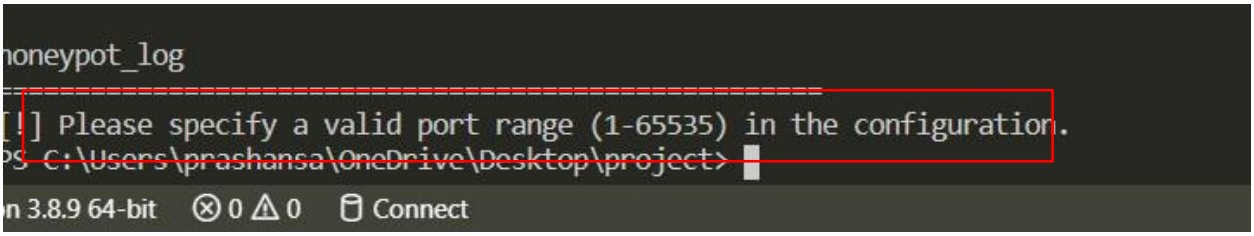


Figure 43 Test Results of Test Case 3 checking port range

4.2.1.4 TEST CASE 4

Table 6 Honeypot Module Test Case 4

Test Case 4	
Objective	To test whether the system can identify whitelist IP address.
Action	Capture the incoming traffic in the honeypot.
Expected Test Result	Whitelisted IP should be recorded in the log file
Actual Test Result	Whitelisting action is performed and recorded in the log file.
Conclusion	Test Successful.

After doing small configurations in the system, the honeypot is executed to test whether it can identify and capture any whitelisted IP trying to connect to the honeypot. As seen in the table above, it can successfully log the IP address.

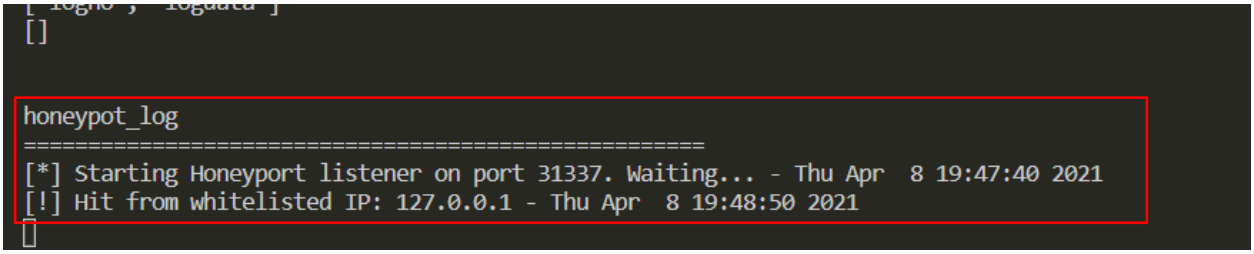


Figure 44 Test Results of Test Case 4 capturing hit from Whitelist IP

4.2.1.5 TEST CASE 5

Table 7 Honeypot Module Test Case 5

Test Case 5	
Objective	To test whether the system can identify connection from external IP address and perform the action of blacklisting them.
Action	Capture any connection from external party other than Whitelisted IP.
Expected Test Result	Blacklisted IP should be recorded in the log file and in the firewall system.
Actual Test Result	Blacklisted IP is recorded in the log file and in the firewall system.
Conclusion	Test Successful.

Similar to Test 4 completed above, the honeypot is again executed to test whether it can identify and capture any blacklisted IP trying to connect to the honeypot. As seen in the table below, it can successfully log the IP address when someone tries to NMAP the IP address of the system.

```
[*] Starting Honeyport listener on port 31337. Waiting... - Thu Apr  8 19:47:40 2021
[!] Hit from whitelisted IP: 127.0.0.1 - Thu Apr  8 19:48:50 2021
[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 192.168.100.7 with Windows Firewall (TTL: Permanent) - Thu Apr  8 19:50:37 2021
[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 192.168.100.7 with Windows Firewall (TTL: Permanent) - Thu Apr  8 19:50:43 2021
[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 192.168.100.7 with Windows Firewall (TTL: Permanent) - Thu Apr  8 19:50:47 2021
```

Figure 45 Test Results of Test Case 5 capturing hit from blacklisted IP in Windows

```
[!] The IP ADDRESS IS BLACKISTED IN LINUX: 100.64.231.163 with IPTABLES (TTL: Permanent) - Wed Apr  7 09:54:25 2021
[*] Starting Honeyport listener on port 31337. Waiting... - Thu Apr  8 10:24:20 2021
[*] Starting Honeyport listener on port 31337. Waiting... - Thu Apr  8 23:02:11 2021
[*] Starting Honeyport listener on port 31337. Waiting... - Thu Apr  8 23:21:04 2021
[!] The IP ADDRESS IS BLACKISTED IN LINUX: 192.168.100.126 with IPTABLES (TTL: Permanent) - Thu Apr  8 23:23:00 2021
```

Figure 46 Test Results of Test Case 5 capturing hit from blacklisted IP in Linux

The rest of the testing cases are provided in the appendix: UNIT TESTING

4.3 SYSTEM TESTING

4.3.1 PLATFORM TESTING

4.3.1.1 TEST CASE 1

Table 8 Platform Testing Test Case 1

Test Case 1	
Objective	To run the honeypot system in Windows Platform.
Action	Execute the script by python3 main.py
Expected Test Result	The system is expected to run the honeypot in Windows platform, capture different types of log, send alert to the admin, and upload the data to the database.
Actual Test Result	The system worked in Windows platform, captured different types of log, sent alert to the admin, and uploaded the data to the database.
Conclusion	Test Successful.

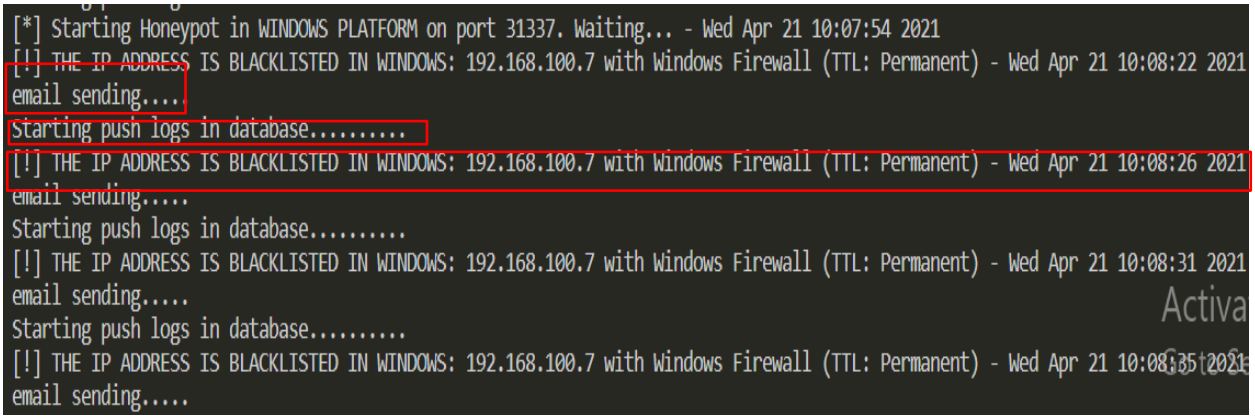


Figure 47 Test Results of Test Case 1 running system in windows

4.3.1.2 TEST CASE 2

Table 9 Platform Testing Test Case 2

Test Case 2	
Objective	To run the honeypot system in Linux Platform.
Action	Execute the script by python3 main.py
Expected Test Result	The system is expected to run the honeypot in Linux platform, capture different types of log, send alert to the admin, and upload the data to the database.
Actual Test Result	The system worked in Linux platform, captured different types of log, sent alert to the admin, and uploaded the data to the database.
Conclusion	Test Successful.

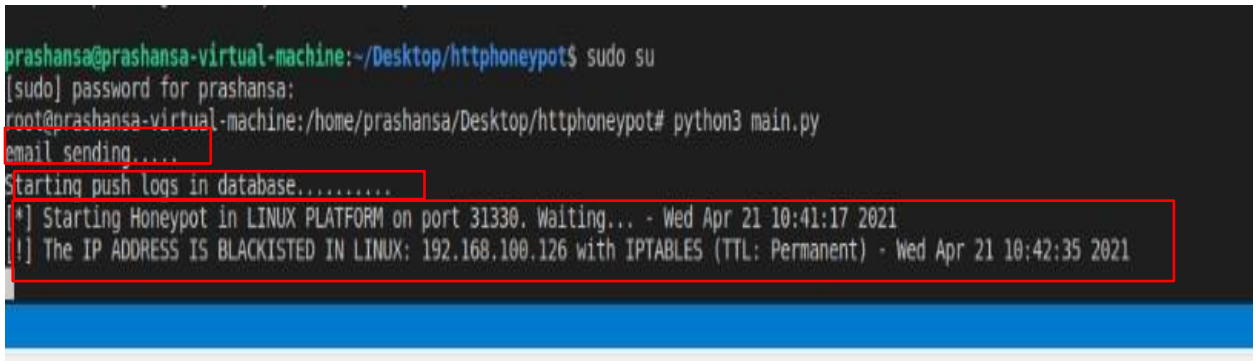


Figure 48 Test Results of Test Case 2 running system in Linux

4.3.2 HONEYPOT UI

4.3.2.1 TEST CASE 3

The test of the web interface is performed which allows the user to view the logs recorded by the honeypot in a much easier and user-friendly manner.

Table 10 Honeypot UI Test Case 3

Test Case 3	
Objective	To test the ability to view the home interface of logs recorded, easily in the web interface.
Action	Enable the Apache and MySQL modules from the XAMPP Control.
Expected Test Result	Run http://localhost:81/honeypot/index.php in the browser
Actual Test Result	The honeypot web interface is expected to launch in the browser.
Conclusion	Test Successful.

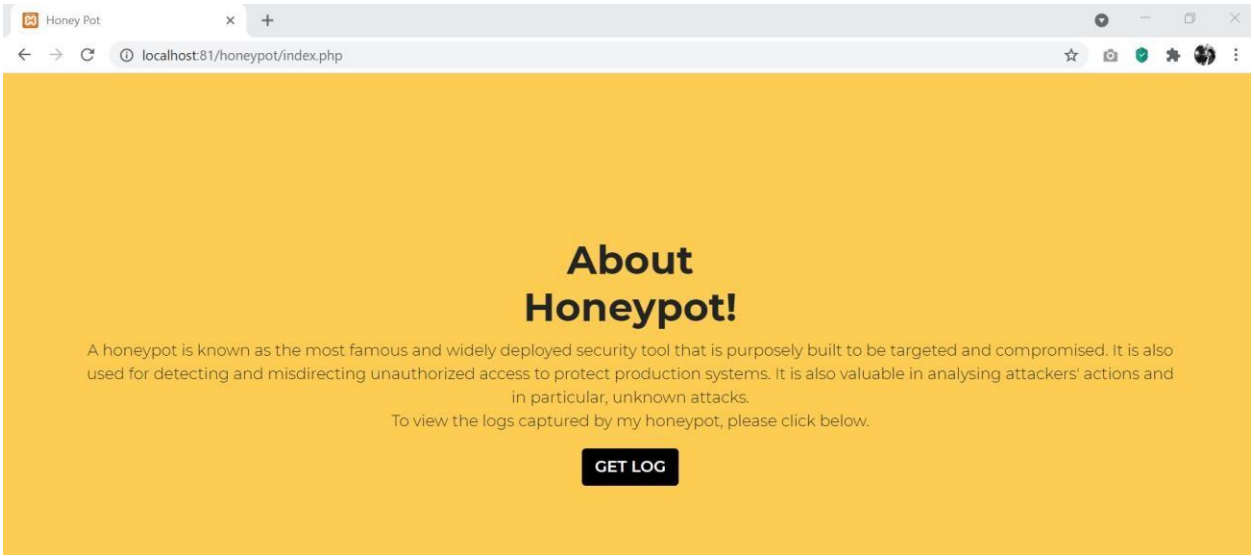


Figure 49 Test Results of Test Case 3 displaying home interface

4.3.2.2 TEST CASE 4

Table 11 Honeypot UI Test Case 4

Test Case 4	
Objective	To test the ability to view the blacklisted logs recorded, easily in the web interface.
Action	Run http://localhost:81/honeypot/log.php in the browser.
Expected Test Result	The honeypot web interface is expected to launch in the browser.
Actual Test Result	The honeypot web interface is launched in the browser.
Conclusion	Test Successful.

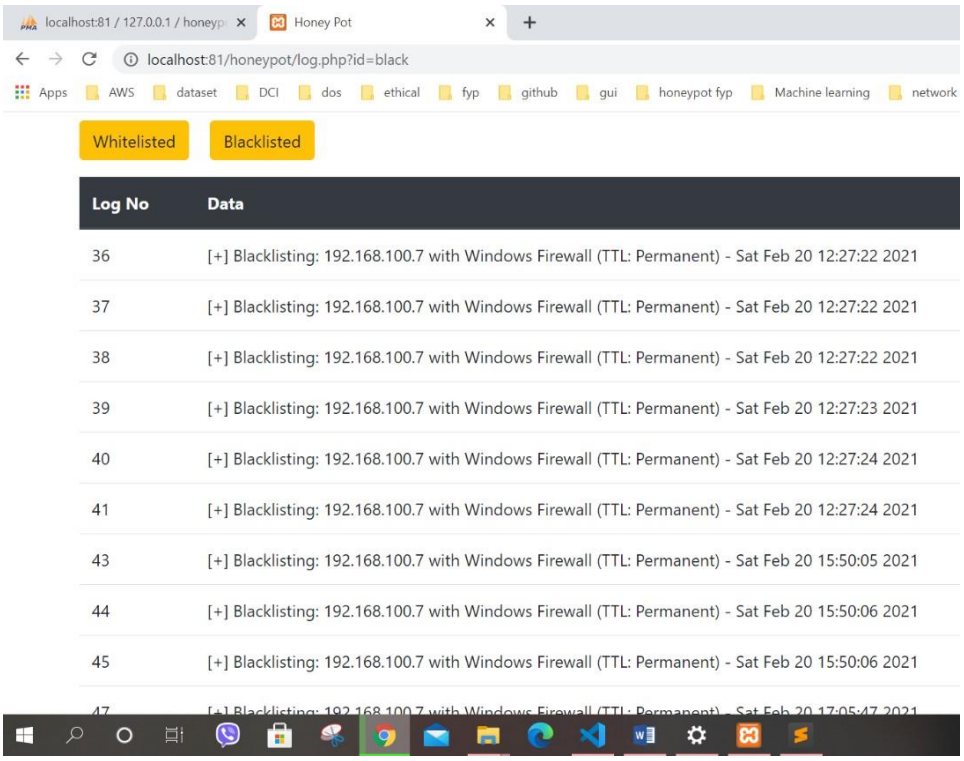


Figure 50 Test Results of Test Case 4 checking blacklisted logs in interface

4.3.2.3 TEST CASE 5

Table 12 Honeypot UI Test Case 5

Test Case 5	
Objective	To test the ability to view the whitelisted logs recorded, easily in the web interface.
Action	Run localhost: 81/honeypot/ in the browser.
Expected Test Result	The honeypot web interface is expected to launch in the browser.
Actual Test Result	The honeypot web interface is launched in the browser.
Conclusion	Test Successful.

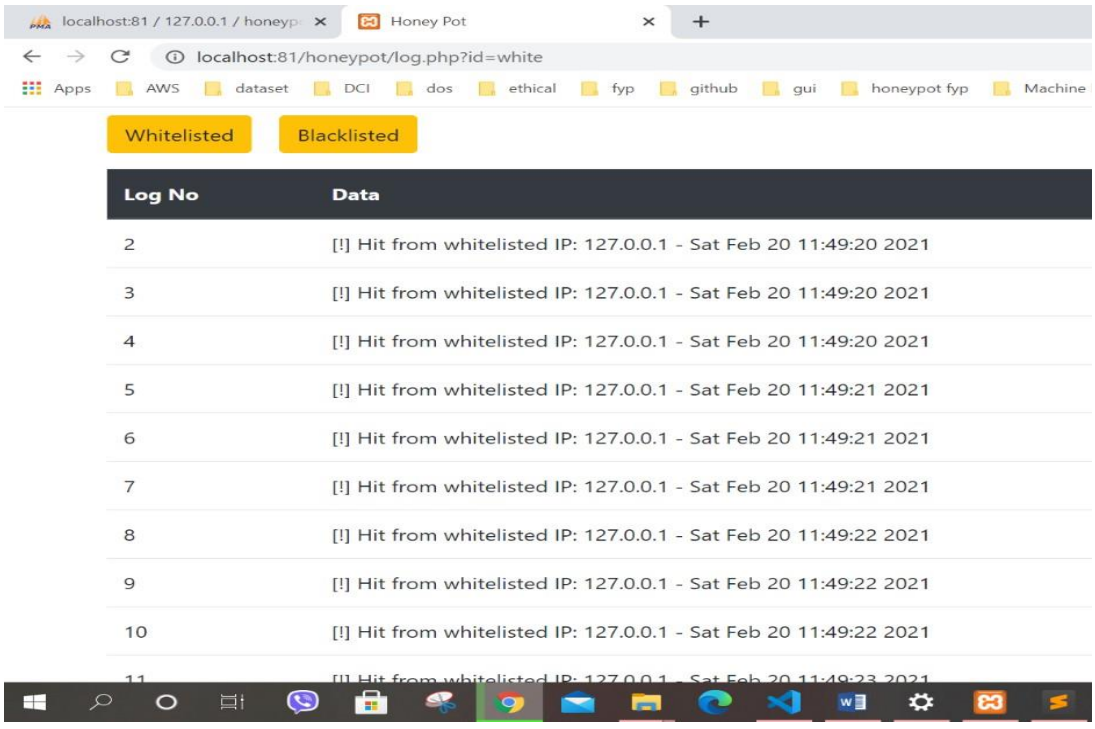


Figure 51 Test Results of Test Case 5 checking whitelisted logs in interface

The rest of the testing cases are provided in the appendix: SYSTEM TESTING

4.4 CRITICAL ANALYSIS

The project had various teething problems during the testing phases. However, the problems and errors were being corrected and the system was being developed correspondingly. Overall the components in the project passed all of its self-conducted unit and system tests. The system is fully operational and includes all the features mentioned in the proposal. A user acceptance testing was carried for the beta version of the project, in which potential end-users were given the incentive to interact with the system and offer input.

The reviews gathered from users revealed a few qualities and shortcomings in the system. The feedbacks indicated that the data capture and visualization were the most liked features in the system. However, due to heavy time constraint, the author is well aware that the development of these features are not enough to capture the real-time comprehensive description of the intruder and so, this has been the top priority in the future works of this project. Secondly, the survey result shows that the web user interface developed did not quite gather the attention it should have. If some real-time statistics data captured from honeypot is introduced interactively, this has high chances of getting attention from various professional companies from all around the world in the future. The users also seem to like the IP blacklisting feature but the admin alert feature did not seem to be impressed by the email message and would have preferred mobile message alert as it is more convenient.

Besides, after running the python script, the interfaces for capturing and analyzing traffic are executed. This can be bothersome at times. To enhance this, a module could be created that can run without any commands and can read and display data directly. The other thing that needs to be understood in this project is that it uses two machine learning algorithms. In almost all instances, Decision Tree outperforms Logistic Regression when it comes to precision. The parameters of both algorithms were also tuned to find the best answer. There are several other features in Python that can assist in finding the best solution to the problem among which some of them were looked at to get some ideas. The accuracy of these algorithms are computed on their performance relative to the overall target for each row, and since the algorithms are improving and updating their random values, the duration of time needed to train these algorithms to attain the highest accuracy is indefinite, so if the algorithms run for a long time training on the same relevant data, they will begin to recite that data and will be able to respond well and the accuracy will steadily increase.

But they will not respond well to new data. This is referred to as overfitting. To avoid this, the algorithms are fed new data, known as test data, which accounts for 20% of the total dataset. Tests were repeated against dataset size to assess the efficiency of the algorithms. The results show that the Decision Tree (99% accuracy) surpasses Logistic Regression (96% accuracy) by a considerable margin. Although the overall performance is high, more convoluted models such as Bayesian networks, Neural Networks or K-nearest neighbours could have been used to increase the accuracy.

CHAPTER 5: CONCLUSION

While our reliance on computer networks grows, rigorous network security becomes increasingly important. Gaining a thorough understanding of the network being probed and targeted is a first prerequisite for being able to properly defend network assets. The honeypot idea was created specifically to meet this need. The proposed project focuses on detecting port scanning, which is one of the most popular approaches used by intruders to identify tools that they can use to break into networks, as well as a few security mechanisms. The design includes a dynamic honeypot system that integrates data collected from active probing tools such as NMAP. While port scanning is not necessarily aggressive, it seems to be the initial phase in an intruder's reconnaissance process when attempting to penetrate a network or steal or ruin confidential material.

Port vulnerabilities, which is perhaps the most popular threat that attackers may use to initiate an intrusion, were also extensively studied during the development of the project. A dataset was used, along with two classification algorithms, to detect anomalies using a supervised machine learning technique, based on the proposal and the core network scenario mentioned. To summarize, combining logistic regression and decision trees is a relatively new method, but it has the potential to outperform both decision trees and logistic regression alone. The discrepancies between decision trees and logistic regression are small in the study at hand. Combining decision trees and logistic regression, on the other hand, will yield substantially better results when dealing with raw data. Based on the reviewed literature, it is clear that anomaly-based intrusion detection is the most effective way to protect a network against novel attacks. However, because of its immaturity, there are still issues with its validity.

To the aimed to contribute, there are currently only a few studies that have used the supervised machine learning technique to detect anomaly attacks with honeypot in the core network. This project has anchorage the honeypot system into an influential solution for any network.

5.1 LEGAL, SOCIAL AND ETHICAL ISSUES

5.1.1 LEGAL ISSUES

This subject is addressed in precise detail by Richard Salgado in chapter 8 of his book "Know Your Enemy," where it has explained that deciding the laws apply to a specific honeypot deployment and to what degree is crucial to avoid legal risks. Firstly, tracking the honeypots' internet traffic can infringe on a few users' legal rights, even though they are all illegitimate. Secondly, if the honeypot is used by an intruder to commit a crime, the information obtained from it will need to be handled differently which will be discussed in the privacy section of ethical issues. Finally, if a honeypot is used by a hacker to cause harm to external parties, such third parties can decide to sue the honeypot's owner for financial damages.

In the case of Nepal, however, no such laws have been proposed or implemented specifically to honeypot agreeing to the interview with Mr. Aaditya Khati. However according to the Nepal Electronic Act 2063, "infringement of others electronic information for malevolent intent will result in a fine beyond two thousand rupees and a sentence of detention of sporadically till three years, or both." Using such frameworks against the Nepalese electronic act may lead to a penalty.

To summarize the main legal concerns of the project:

- The author will not be delinquent if the project is misused in any way that violates the rules of the nation in which it is used.
- Knowing which laws will apply in such circumstances is a sensitive issue, and obtaining legal advice before installing a honeypot is highly advised according to the country's laws.

5.1.2 SOCIAL ISSUES

This project is a free and open-source (FOSS) project. This project is purely designed for the enthusiasts out there who are interested in the Network Security domain and would be interested in monitoring or capturing network information. The analytic data can be helpful to analysts to learn about the resource's targeted features, which can then be defended in the real system with greater precision. The monitoring and analytics framework for honeypots may be used for study purposes. Students can communicate with honeypots as a project and learn more about vulnerabilities. There is no intention of causing harm to anyone who uses this project. Even though the scripts need admin and root privileges to run, no malicious motives are embedded in

the script itself, wrecking chaos between consumers. Furthermore, none of the sensitive information such as user passwords, critical system data, is revealed in the logs produced by the script. As a consequence, there are no such concerns that arise as a result of this project that would result in a social breach.

5.1.3 ETHICAL ISSUES

There appears to be a widespread misconception that using honeypots has many legal and ethical ramifications. While there are legal issues that should be considered, as discussed in the earlier section, but through research and interviews, it is believed that there are no significant ethical issues associated with the use of honeypots. The key issue may be the possible violation of the privacy of a person communicating with or by a honeypot, as all contacts could be tracked and logged. Any person performing such correspondence, however, would be an illegitimate user of the honeypot, according to the concept of the honeypot. Therefore, the intruder is breaking into others' private system and honeypot is there to track the movements of the intruder without harming the privacy of the legitimate users (Conde, 2005). The only exception is if an intruder gets control of a honeypot and uses it to store or transmit sensitive information or reasonable suspicion to third parties. The issue will not be with the honeypot collecting the information and making it accessible to the honeypot admin, but with how the admin manages the data until s/he discovers its existence (Conde, 2005).

5.2 ADVANTAGES

The system is one of the most innovative systems with almost all the minimal features required for a successful approach to launch in the network. It poses some of the advantages such as:

- The system is simple to use and easy to configure. It does not contain any type of complex signatures or tables to maintain and update.
- As the honeypot is built to capture intruder's activity, it needs minimal resources.
- The system automatically blocks the IP address if it finds a connection other than the whitelisted IP. This can be very advantageous to organizations to detect and respond quickly.
- It provides a log display web interface to view the data conveniently collected by the honeypot.
- It also provides data visualization and model predictions through a simple web user interface.
- It consists of a backup feature through the MySQL database.
- It delivers an alert mechanism to the system admin.

5.3 LIMITATIONS

The proposed system is an admirable decoy-based anomaly detection honeypot, but since it is only a prototype project designed to be completed in a limited amount of time for a final year project, so it has clear drawbacks:

- The training and test data in this study consisted of the dataset containing network features derived online and not captured from a real-time network.
- The system needs to be manually started which sometimes can be tedious work.
- The execution of an extremely large dataset causes the machine learning interface to freeze.
- Only two machine learning algorithms are supported by the system.
- The low-interaction honeypot is prone to fingerprint.

5.4 FUTURE WORK

The main aim of this project is to detect, not prevent. Although the project has delivered all the commitments made in the plan, however, there is still room for improvement. Thus, this work is open for future enhancements. A few of these enhancement options are discussed in this section.

This work aims to promote further study in this field. A good starting point will be to keep researching how honey technologies are used, as well as security specialists' awareness and attitudes about them. Collecting data over time could aid academic researchers in determining patterns at various points in time. Specific company or industry surveys will also massively broaden the information on the use of these technologies. Most significantly, these initiatives are indeed effective while they are progressing and constant. The author is willing to share the outcomes of the research, cleaned of descriptive details such as IP addresses and timestamps, in the context of open source and educational engagement. Any researchers searching for additional similarities in the dataset excluded in this report might find this dataset useful. For more details, kindly email the author.

Further future works are explained in detail in the appendix: FUTURE WORK

CHAPTER 6: REFERENCES

Blaziunas, S. & Raudys, A., 2019. Comparative study of neural networks and decision trees for application in trading financial futures. *International Conference on Deep Learning and Machine Learning in Emerging Applications.*

Rathore, P. & Jain, N., 2013. HONEYPOT TECHNIQUE USED FOR INTRUSION DETECTION SYSTEM. *International Journal of Science, Engineering and Technology Research (IJSETR),* 2(12).

Aggarwal, S., 2019. *Context Aware Honyepot for Cross-Site Scripting attacks using Machine Learning Techniques,* Kanpur: s.n.

Ahmed, T., Oreshkin, B. & Coates, M., 2007. *Machine Learning Approaches to Network Anomaly Detection,* s.l.: McGill University.

Alata, E. N. V. K. M. D. M. &. H. M., 2006. *Lessons learned from the deployment of a high-interaction honeypot.* Coimbra, Portugal, 2006 Sixth European Dependable Computing Conference.

Alsagheer, R. H. A., Alharan, A. F. H. & Al-Haboobi, A. S. A., 2017. Popular Decision Tree Algorithms of Data Mining Techniques: A Review. *International Journal of Computer Science and Mobile Computing,* 6(6), p. 133 – 142.

ALTVATER, A., 2017. *Software Testing Tips: 101 Expert Tips, Tricks and Strategies for Better, Faster Testing and Leveraging Results for Success.* [Online]
Available at: https://stackify.com/software-testing-tips/
[Accessed 6 April 2021].

Answers Course, 2021. *(Solved):Q: Define supplier.* [Online]
Available at: https://answerscourse.com/2021/04/18/solvedq-define-supplier/
[Accessed 25 April 2021].

Barracuda, 2021. *Cloud Firewalls.* [Online]
Available at: https://www.barracuda.com/glossary/cloud-firewall#:~:text=Cloud%20Firewalls%20are%20software%2Dbased,sit%20within%20online%20application%20environments.
[Accessed 17 April 2021].

Bonam, T., 2015. Using Feedback Loops to Boost Development Lifecycles. *Agile Connection*, 25 March.

Brenyah, B., 2020. Julia For Data Science: Regularized Logistic Regression. *TowardsDataScience*, 4 Feburary.

Brownlee, J., 2014. A Gentle Introduction to Scikit-Learn: A Python Machine Learning Library. *Machine Learning Mastery*, 16 April.

Business News Daily , 2020. *What Is Agile Scrum Methodology?.* [Online]
Available at: https://www.businessnewsdaily.com/4987-what-is-agile-scrum-methodology.html
[Accessed 22 December 2020].

Campbell, R. M. & Padayachee, K., 2015. *A Survey of Honeypot Research: Trends and Opportunities.* Johannesburg, South Africa, International Conference for Internet Technology and Secured Transactions (ICITST-2015).

CHANDOLA, V., BANERJEE, A. & KUMAR, V., 2009. Anomaly Detection : A Survey. *ACM Computing Surveys*, 9 September, pp. 1-72.

Conde, D. P., 2005. *Deploying Honeypots and the Security Architecture of a Fictitious Company,* s.l.: SANS Institute.

Constantin Musca, E. M. R. D., 2013. *Detecting and Analyzing Zero-day Attacks using Honeypots.* s.l., International Conference on Control Systems and Computer Science.

Dey, A., 2016. Machine Learning Algorithms: A Review. *) International Journal of Computer Science and Information Technologies,* 7(3), pp. 1174-1179.

Digité, Inc, 2020. *What is Scrum?.* [Online]
Available at: https://www.digite.com/agile/scrum-methodology/
[Accessed 15 October 2020].

Digite, 2020. *What is Kanban?.* [Online]
Available at: https://www.digite.com/kanban/what-is-kanban/
[Accessed 23 October 2020].

Dilsheer Ali. P, G. K. T., 2017. *MALWARE CAPTURING AND DETECTION IN DIONAEA HONEYPOT.* s.l., International Conference on Innovations in Power and Advanced Computing Technologies [i-PACT2017].

Dong Nguyen, C. N. T. D.-B. H. N. A. N. T. T., 2016. Joint Network Coding and Machine Learning forError-prone Wireless Broadcast Dong. 28 December.

Dowling, S., Schukat, M. & Melvin, H., 2017. *Using Analysis of Temporal Variances within a Honeypot Dataset to better predict Attack Type Probability.* NUI Galway, Galway, Ireland , International Conference for Internet Technology and Secured Transactions.

Downs, F., 2020. Top Cyberattacks of 2020 and How to Build Cyberresiliency. *ISACA*, 6 November.

Dwivedi, R., 2020. *What is a network socket?.* [Online]
Available at: https://dev.to/rdrahul/what-is-a-network-socket-5d4
[Accessed 24 December 2020].

Fan, W., Du, Z. & Fernandez, D., 2018. Enabling an Anatomic View to InvestigateHoneypot Systems: A Survey. *IEEE Systems Journal,* 12(4).

Fan, W., Du, Z., Fernandez, D. & Villagra, V. A., 2018. Enabling an Anatomic View to InvestigateHoneypot Systems: A Survey. *IEEE Systems Journal,* 12(4).

Franklin Mayorga, J. V. E. Á., 2019. *Honeypot network configuration through cyberattack patterns.* s.l., International Conference on Information Systems and Computer Science (INCISCOS).

Garson, G. D., 2014. *LOGISTIC REGRESSION: BINARY AND MULTINOMIAL.* 1st ed. Asheboro, NC 27205 USA : Statistical Associates.

GeeksforGeeks, 2018. *Logging in Python.* [Online]
Available at: https://www.geeksforgeeks.org/logging-in-python/#:~:text=Python%20has%20a%20built%2Din,what%20problems%20have%20been%20arisen.&text=There%20are%20two%20built%2Din%20levels%20of%20the%20log%20message.
[Accessed 29 March 2021].

GeeksforGeeks, 2020. *MySQL-Connector-Python module in Python.* [Online]
Available at: https://www.geeksforgeeks.org/mysql-connector-python-module-in-python/
[Accessed 29 March 2021].

Gregory Melton, 2021. *Recognizing Suspicious Network Connections with Python,* s.l.: SANS Institute.

Grimes, R., 2004. Honeypots for Windows. *ITProToday*, 29 March.

Grimes, R. A., 2005. Windows Honeypot Deployment. In: K. Winquist, ed. *Honeypot for Windows.* Heidelberg, Germany: Apress, pp. 89-122.

Guru99, 2021. *What is Kanban? Cards, Boards, Core Principles and Practices.* [Online]
Available at: https://www.guru99.com/kanban-cards-boards-methodology.html
[Accessed 20 March 2021].

Horton, N., 2019. *IPS9 in R: Logistic Regression (Chapter 14).* [Online]
Available at: https://nhorton.people.amherst.edu/ips9/chapters/Chapter14.pdf
[Accessed 10 March 2021].

HOSMER, D., LEMESHOW, S. & STURDIVANT, R. X., 2013. *Applied Logistic Regression.* 3rd ed. New Jersey: Wiley.

Htein Lin, S. D. M. R. B. M., 2017. *Generating honeypot traffic for industrial control.* Arlington, VA, United States, International Conference on Critical Infrastructure Protection (ICCIP).

IBM, 2020. *Machine Learning.* [Online]
Available at: https://www.ibm.com/cloud/learn/machine-learning
[Accessed 9 December 2020].

IONOS, 2019. *XAMPP tutorial: installation and first steps.* [Online]
Available at: https://www.ionos.com/digitalguide/server/tools/xampp-tutorial-create-your-own-local-test-server/
[Accessed 25 April 2021].

JavaTpoint, 2018. *Decision Tree Classification Algorithm.* [Online]
Available at: https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm
[Accessed 8 March 2021].

Jidiga, G. R., 2014. Anomaly Detection using Machine Learning with a Case Study. *IEEE International Conference on Advanced Communication Control and Computing Technologies (ICACCCT),* pp. 1060-1065.

JournalDev IT Services Private Limited, 2020. *Python os Library Functions.* [Online]
Available at: https://www.journaldev.com/30205/python-os-library-functions
[Accessed 18 December 2020].

Kanbanize, 2021. *The Ultimate Guide to Kanban Software Development.* [Online]
Available at: https://kanbanize.com/kanban-resources/case-studies/kanban-for-software-development-teams
[Accessed 31 March 2021].

Kanbanize, 2021. *What Is a Kanban Board?.* [Online]
Available at: https://kanbanize.com/kanban-resources/getting-started/what-is-kanban-board
[Accessed 20 March 2021].

Kanbanize, 2021. *What Is a WIP Limit in Kanban, and Why Is It Important?.* [Online]
Available at: The maximum number of work items in each stage (kanban board column) of the workflow is limited by work in progress (WIP) limits. WIP limits help the team concentrate only on current activities, allowing you to complete single work items faster.
[Accessed 20 March 2021].

Kevat, S. M., 2017. Review on Honeypot Security. *International Research Journal of Engineering and Technology (IRJET),* 4(6).

Kienitz, P., 2017. *The pros and cons of Iterative Software Development.* [Online]
Available at: https://www.dcslsoftware.com/pros-cons-iterative-software-development/
[Accessed 20 March 2021].

Klipp, P., 2014. *Getting Started With Kanban.* 1st ed. s.l.:kanbanery.

Kovtun, M., 2018. *Scalable Honeypot Monitoring and Analystics,* s.l.: s.n.

Kreibich, C. & Crowcroft, J., 2003. *Honeycomb – Creating Intrusion Detection Signatures Using Honeypots.* s.l., Honeycomb – Creating Intrusion Detection.

Lam, M., 2019. Using TED talks for Machine Learning. *towardsdatascience*, 11 July .

Leybourn, E., 2015. 7 stages of delivery - an example Kanban. *The Agile Director*, 21 April.

Li, L., Sun, H. & Zhang, Z., 2011. *The research and design of honeypot system applied in the LAN security.* Beijing, China, IEEE.

Litchfield, D., 2019. The Inside Story of SQL Slammer. *ThreatPost*, 20 October.

LOTTE, F., 2015. 1Signal processing approaches to minimize orsuppress calibration time in oscillatoryactivity-based Brain-Computer Interfaces. *Institute of Elec-trical and Electronics Engineers (IEEE),* 103(6), pp. 871-890.

Mahone, M. V., 2003. *A Machine Learning Approach to Detecting Attacks by Identifying Anomalies in Network Traffic,* s.l.: Melbourne, Florida.

Mairh, A. B. D. V. K. &. J. D., 2011. *Honeypot in Network Security: A Survey.* Bhubaneswar, Odisha, India , International Conference on Communication, Computing & Security.

Matthew L. Bringer, C. A. C. a. H. F., 2012. A Survey: Recent Advances and Future Trends in Honeypot Research. *I. J. Computer Network and Information Security,* Volume 10, pp. 63-75.

McAfee Labs, 2018. *McAfee Labs Threats Report,* s.l.: McAfee Labs .

Mitchell, A., 2018. *An Intelligent Honeypot,* s.l.: Cork Institute of Technology.

Nandy, A. & Biswas, M., 2018. *Reinforcement Learning With Open AI, TensorFlow and Keras Using Python.* 1st ed. Kolkata, West Bengal, India: Apress.

Nawrocki, M., 2016. *A Survey on Honeypot Software and Data Analysis,* Berlin, Germany: s.n.

Neelima, E. & Naga , K. D. S., 2013. A Study on SCRUM Agile Methodology And Its Knowledge Management Process. *The International Journal Of Engineering And Science (Ijes),* 2(3), pp. 22-27.

Oracle, 2020. *What Is a Socket?.* [Online]
Available at: https://docs.oracle.com/javase/tutorial/networking/sockets/definition.html
[Accessed 25 April 2021].

Patil, P., 2018. What is Exploratory Data Analysis?. *towardsdatascience*, 28 March.

Pedro Henrique Matheus da Costa Ferreira, L. N. d. C., 2014. *Extracting IDS Rules from Honeypot Data: A Decision Tree Approach.* Thessaloniki, The Proceedings of the International Conference in Information Security and Digital Forensics.

Permana, P. A. G., 2015. Scrum Method Implementation in a Software Development Project Management. *(IJACSA) International Journal of Advanced Computer Science and Applications,* 6(9).

PlanetTogether, 2021. *The Four Principles of Kanban.* [Online]
Available at: https://www.planettogether.com/blog/the-four-principles-of-kanban
[Accessed 20 March 2021].

PlanView, 2021. *What Is a Kanban Card?.* [Online]
Available at: https://www.planview.com/resources/guide/introduction-to-kanban/what-is-a-kanban-card/#:~:text=A%20Kanban%20card%20is%20a,the%20steps%20in%20a%20process.
[Accessed 20 March 2021].

Python Software Foundation, 2021. *matplotlib 3.4.0.* [Online]
Available at: https://pypi.org/project/matplotlib/
[Accessed 29 March 2021].

Pythonic, 2020. *Socket Module In Python.* [Online]
Available at: https://pythontic.com/modules/socket/introduction
[Accessed 17 December 2020].

R N Dahbul, C. L. J. P., 2016. *Enhancing Honeypot Deception Capability ThroughNetwork Service Fingerprinting.* Tangerang, Department of Information Technology, Swiss German University.

Rahman, M. M., Roy, S. & Yousuf, M. A., 2019. *DDoS Mitigation and Intrusion Prevention in Content Delivery Networks using Distributed Virtual Honeypots.* Dhaka, International Conference on Advances in Science, Engineering and Robotics Technology.

Rajat B. Wakode, L. P. R. P. T., 2015. Overview on Kanban Methodology and its Implementation. *International Journal for Scientific Research & Developmen,* 3(2).

Reddy, G. J. & Sammulal, P., 2014. Anomaly Detection using Machine Learning with a case study. *IEEE International Conference on Advanced Communication Control and Computing Technologies.*

REHKOPF, M., 2021. *Kanban vs. scrum: which agile are you?.* [Online]
Available at: https://www.atlassian.com/agile/kanban/kanban-vs-scrum#:~:text=Kanban%20helps%20visualize%20your%20work,you%20go%20with%20the%20flow.
[Accessed 20 March 2021].

Richard J. Turver, M. M., 1994. An early impact analysis technique for software maintenance. *Journal of Software: Evolution and Process,* 6(1), pp. 35-52.

Rushin, G. et al., 2017. Horse Race Analysis in Credit Card Fraud-Deep Learning, Logistic Regression, and Gradient Boosted Tree. *IEEE.*

Sardana, A. & Joshi, R., 2011. *Honeypots A New Paradigm to Information Security.* Enfield, New Hampshire: Science Publishers.

SCADA , 2018. *WUSTL-IIOT-2018 Dataset for ICS (SCADA) Cybersecurity Research.* [Online]
Available at: https://www.cse.wustl.edu/~jain/iiot/index.html
[Accessed 11 April 2021].

Schreier, J., 2014. How DDoS Attacks Work, And Why They're So Hard To Stop. *Kotaku*, 30 December.

Scrum Alliance , 2021. SCRUM FUNDAMENTALS. *Your Quick Guide to All Things Scrum*.

Scrumwise, 2021. *How does release planning work?.* [Online]
Available at: https://support.scrumwise.com/article/161-how-does-release-planning-work#:~:text=You%20can%20use%20release%20planning,used%20as%20an%20internal%20milestone.
[Accessed 13 April 2021].

SHALABH, 2017. Logistic Regression Models. In: *LINEAR REGRESSION ANALYSIS AND FORECASTING.* Kanpur, India: IIT.

Sharma, A., 2013. Honeypots in network security. *International Journal of Tecgnical Research and Applications,* 1(5), pp. 07-12.

Shiue, L.-M. & Kao, S.-J., 2008. *Countermeasure for Detection of Honeypot Deployment.* Kuala Lumpur, Malaysia, Proceedings of the International Conference on Computer and Communication Engineering.

Shobayo, O. & Rodrigues, M., 2017. Design and Implementation of a Low-Cost Low Interaction IDS/IPS System Using Virtual Honeypot. *Covenant Journal of Informatics & Communication Technology,* 5(1).

simplilearn, 2021. *What is a Confusion Matrix in Machine Learning?.* [Online]
Available at: https://www.simplilearn.com/tutorials/machine-learning-tutorial/confusion-matrix-machine-learning
[Accessed 3 April 2021].

Smith, S. D., 2021. *Catching Flies: A Guide to the Various Flavors of Honeypots,* s.l.: SANS Institute.

Sperandei, S., 2014. Understanding logistic regression analysis. *Biochemia Medica,* 24(1), pp. 12-8.

Stack Abuse, 2021. *Python Logging Basics.* [Online]
Available at: https://stackabuse.com/python-logging-basics/
[Accessed 25 April 2021].

Statista, 2021. *Amount of monetary damage caused by reported cyber crime to the IC3 from 2001 to 2020.* [Online]
Available at: https://www.statista.com/statistics/267132/total-damage-caused-by-by-cyber-crime-in-the-us/
[Accessed 25 March 2021].

Streamlit Inc. , 2021. *Create an app.* [Online]
Available at: https://docs.streamlit.io/en/stable/main_concepts.html
[Accessed 25 April 2021].

Technative, 2020. The Benefit of Unsupervised Learning. *Why Unsupervised Machine Learning is the Future of Cybersecurity*, 28 January.

The Royal Society, 2017. *Machine learning:the power and promise of computers that learn by example,* s.l.: s.n.

Truong, P., 2020. Approaches to Anomaly Detection. *Medium*, 20 January.

Turpitka, D., 2020. When You Can't Stop Every Cyberattack, Try Honeypots. *Forbes*, 28 January.

tutorialspoint, 2020. *SDLC - Iterative Model.* [Online]
Available at: https://www.tutorialspoint.com/sdlc/sdlc_iterative_model.htm
[Accessed 21 November 2020].

Visual Paradigm, 2020. *Requirement Analysis Techniques.* [Online]
Available at: https://www.visual-paradigm.com/guide/requirements-gathering/requirement-analysis-techniques/#:~:text=Requirement%20Analysis%2C%20also%20known%20as,software%20being%20built%20or%20modified.&text=From%20What%20to%20How%3A%20Software,requirements%20engineer
[Accessed 28 March 2021].

W3Schools, 2021. *Python Requests Module.* [Online]
Available at: https://www.w3schools.com/python/module_requests.asp
[Accessed 29 March 2021].

Wakode, R. B., Raut, L. P. & Talmale, P., 2015. Overview on Kanban Methodology and its Implementation. *International Journal for Scientific Research & Development*|, 3(2).

CHAPTER 7: APPENDIX

7.1 APPENDIX A: PRE-SURVEY

7.1.1 PRE-SURVEY FORM

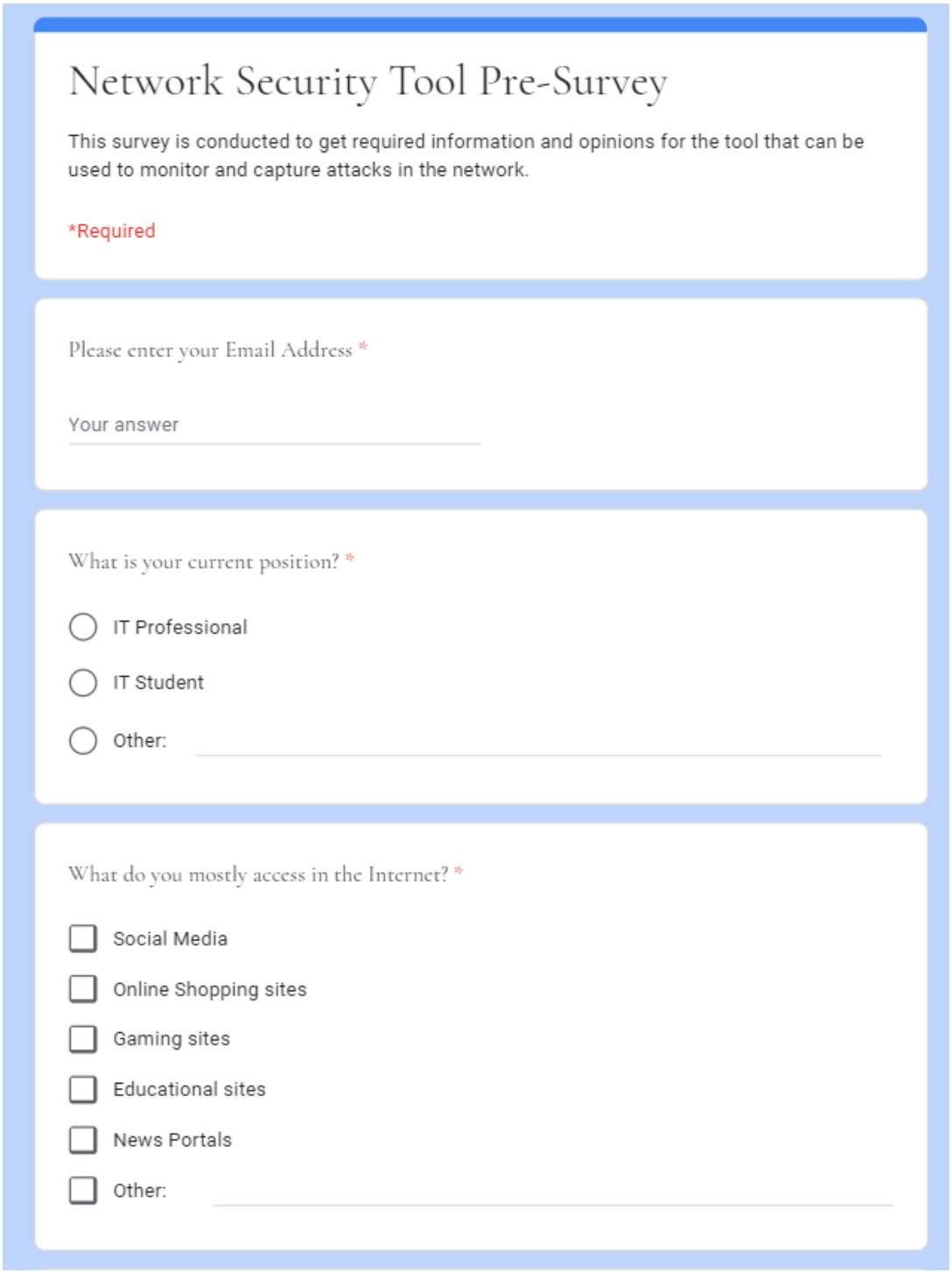


Figure 52 Pre-Survey Form 1

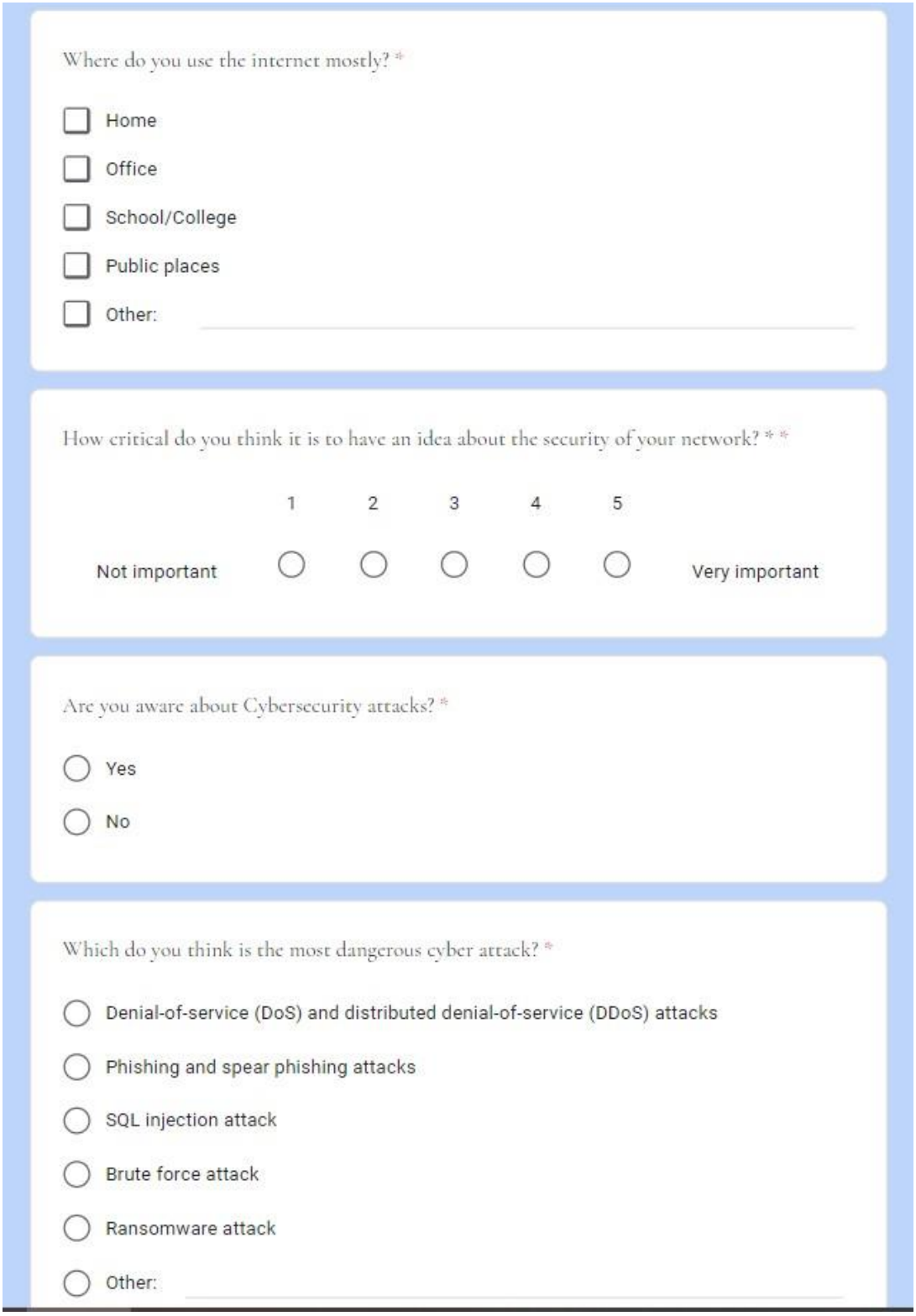


Figure 53 Pre-Survey Form 2

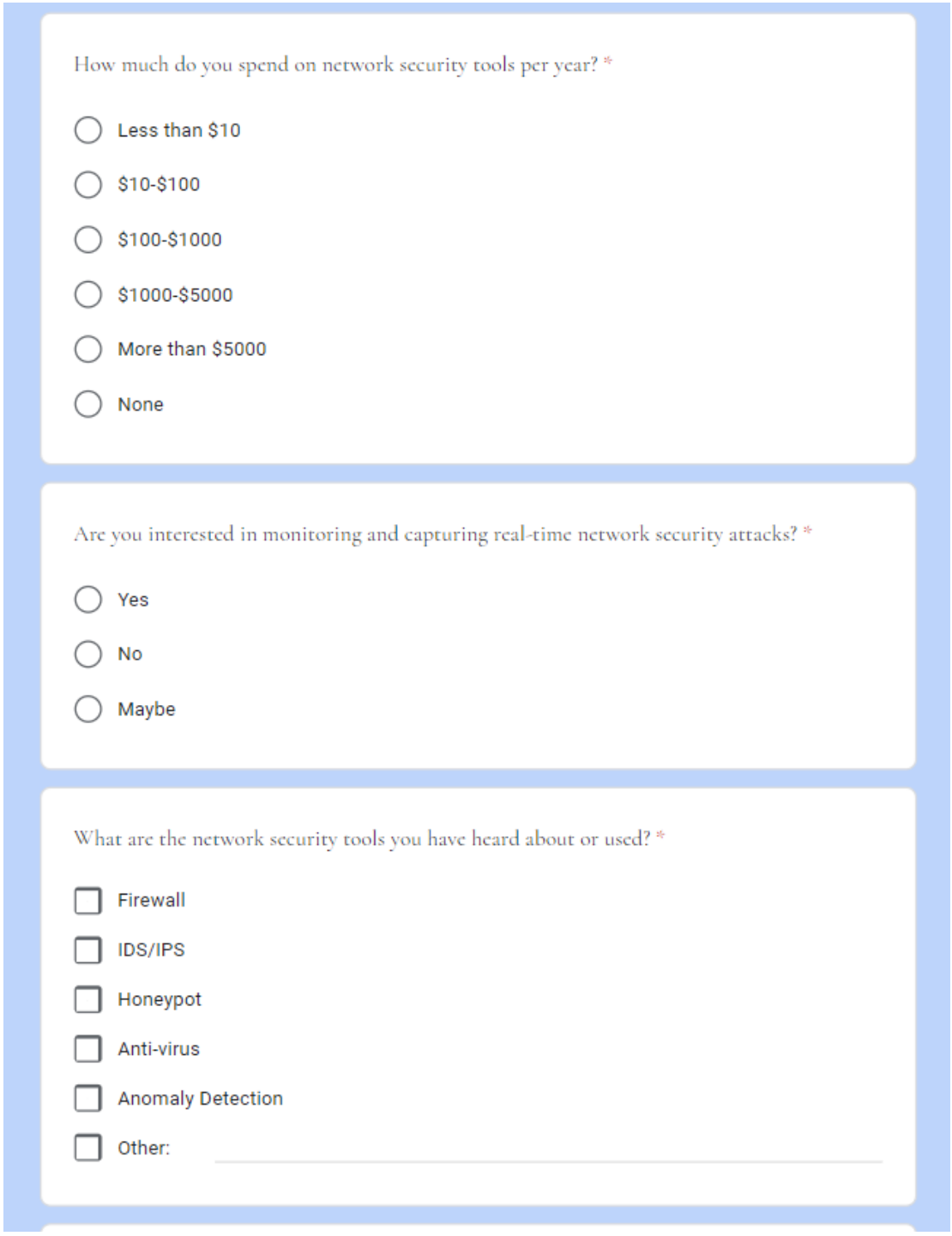


Figure 54 Pre-Survey Form 3

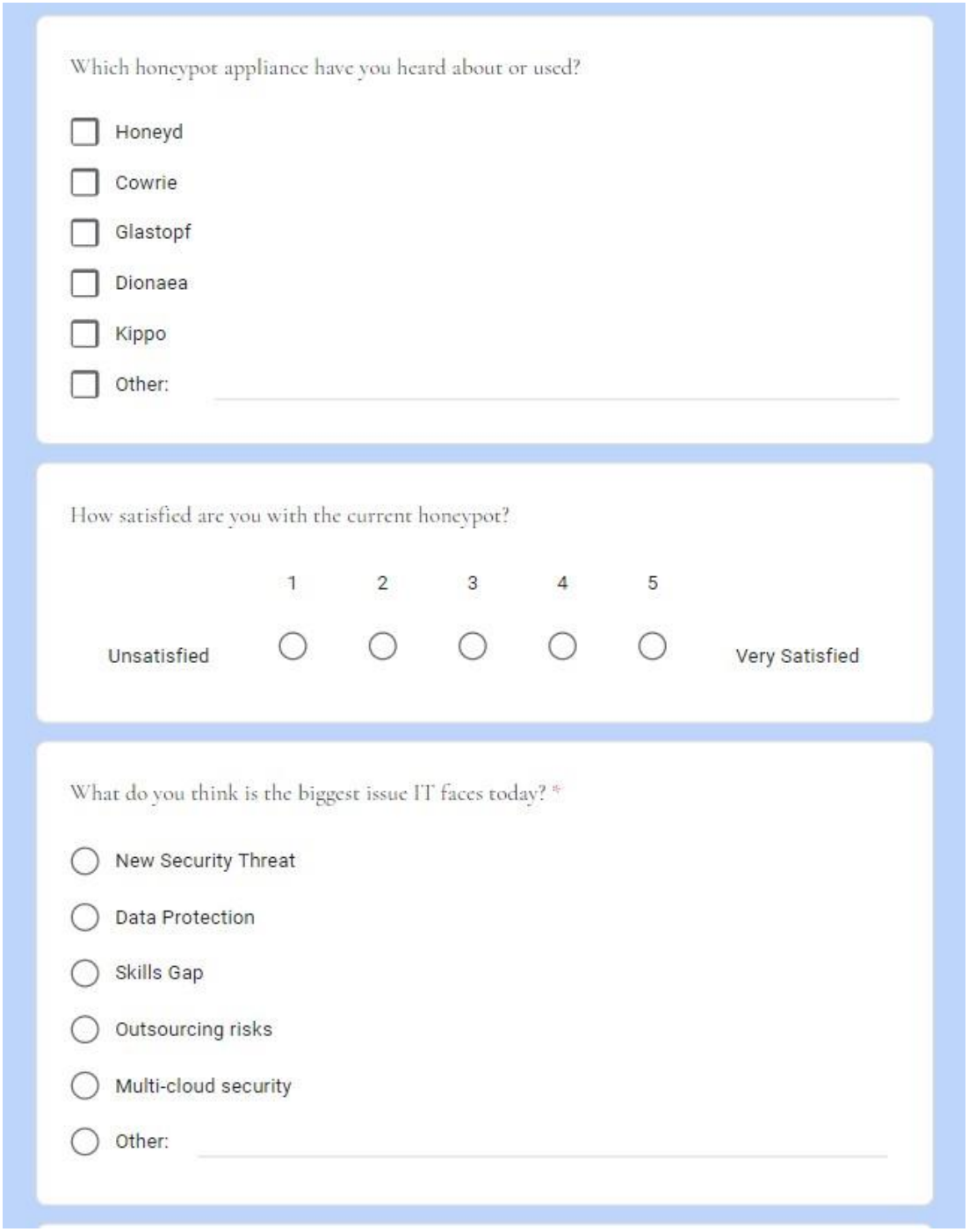


Figure 55 Pre-Survey Form 4

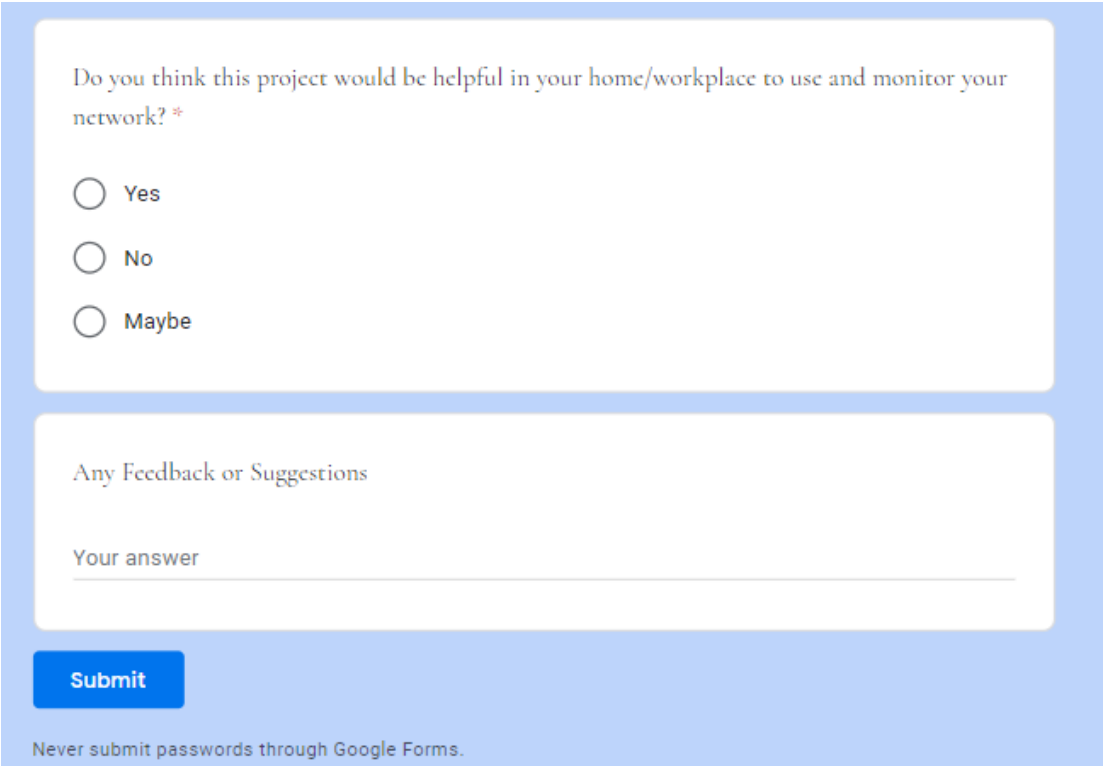


Figure 56 Pre-Survey Form 5

7.1.2 SAMPLE OF FILLED PRE-SURVEY FORMS

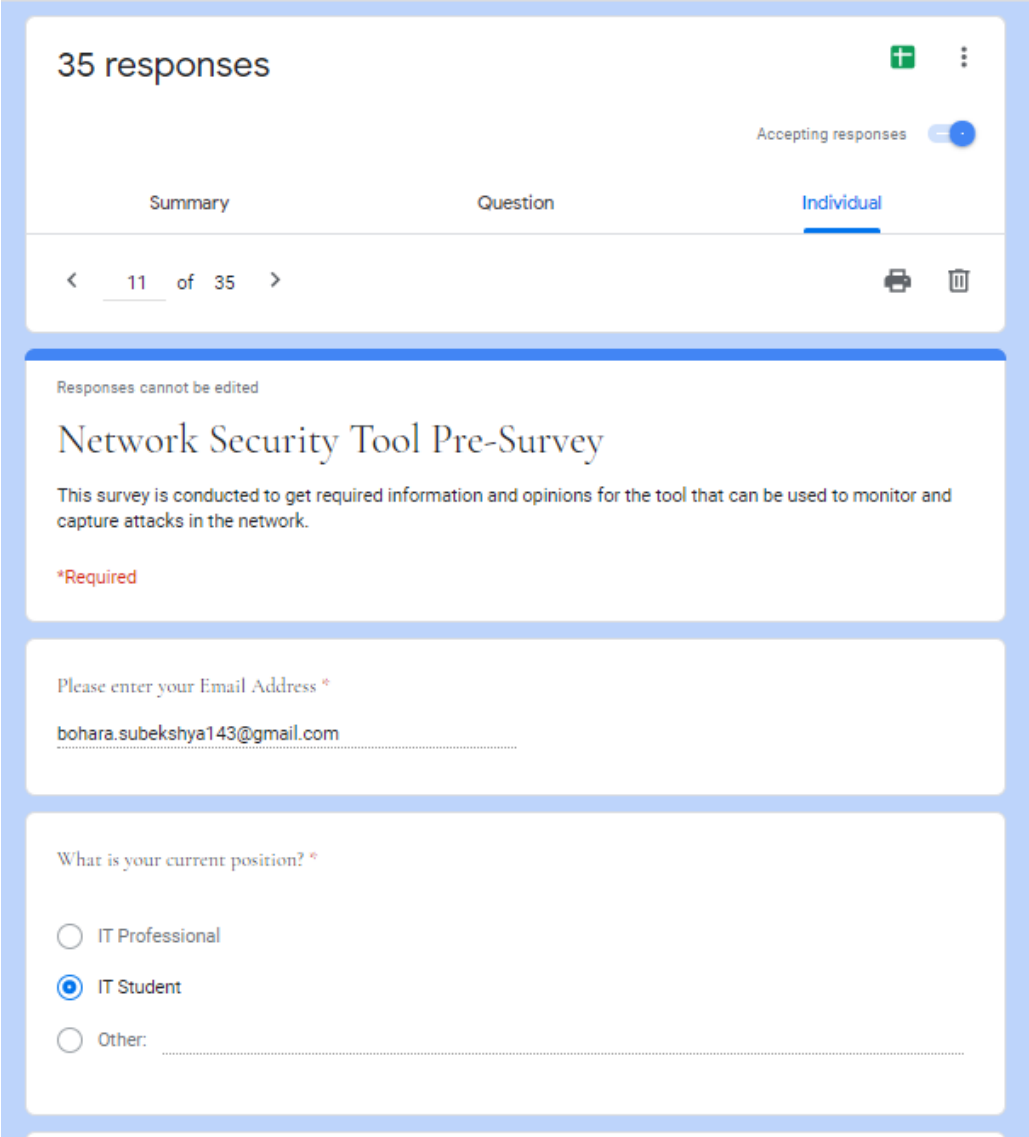


Figure 57 Sample of Pre-Survey Form 1

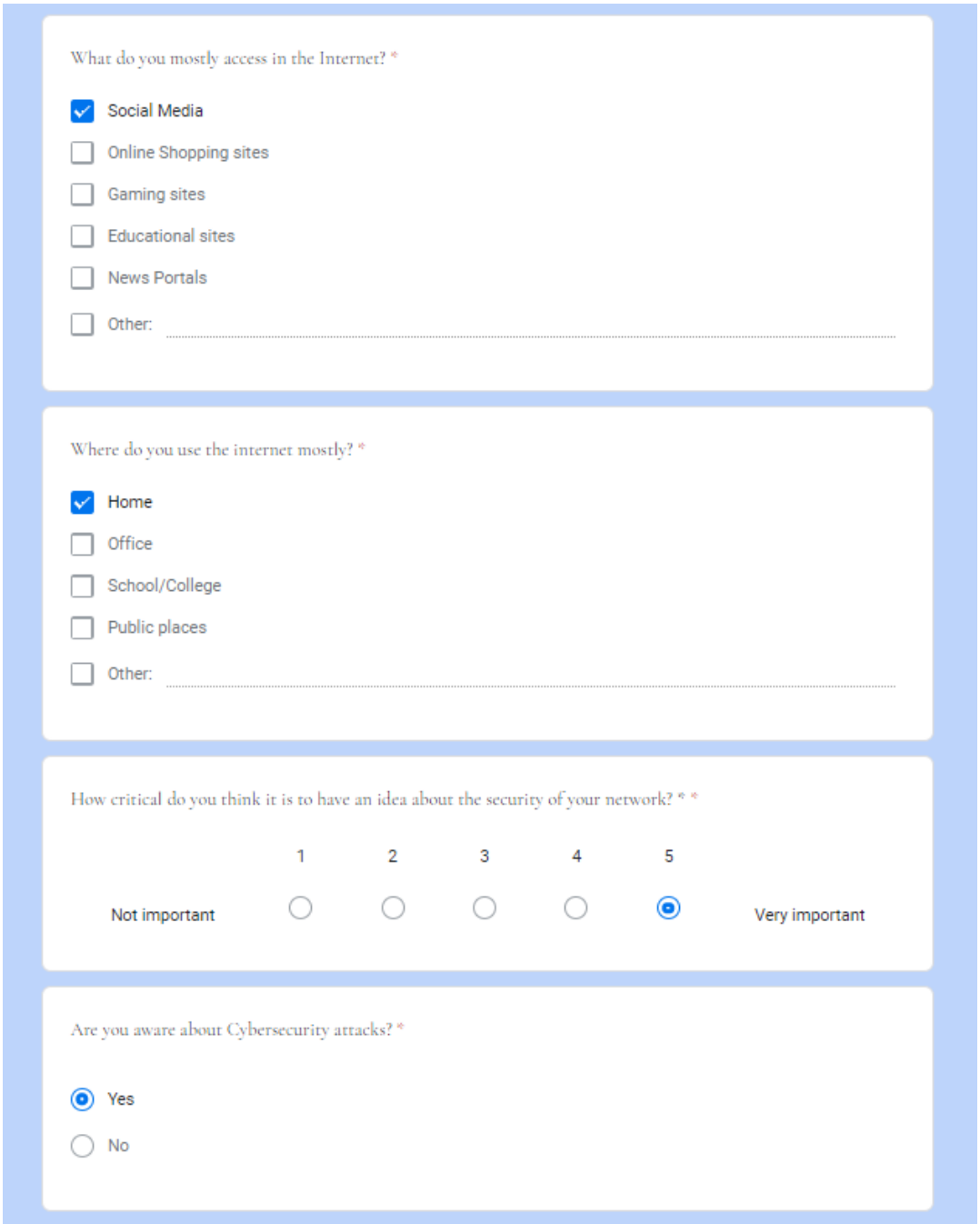


Figure 58 Sample of Pre-Survey Form 2

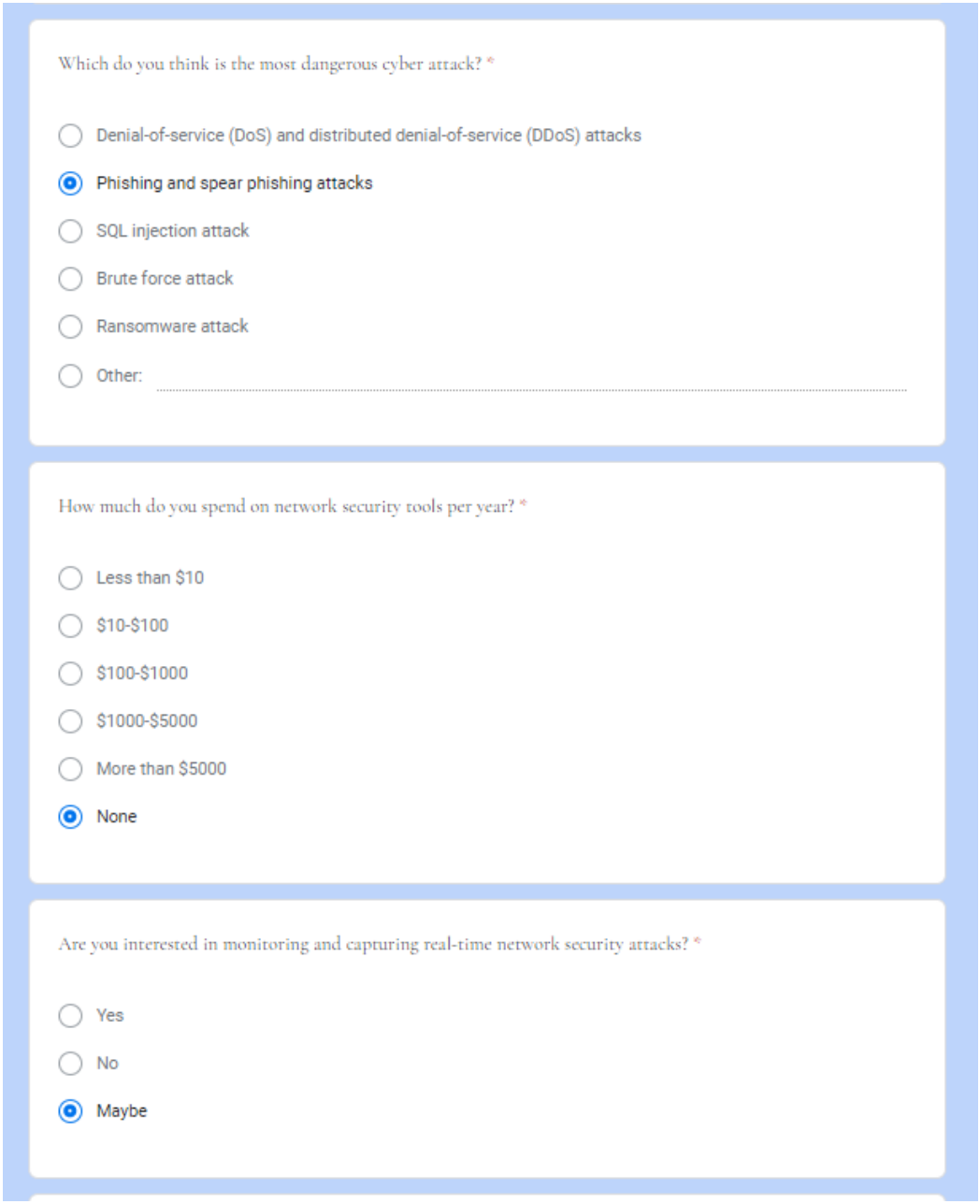


Figure 59 Sample of Pre-Survey Form 3

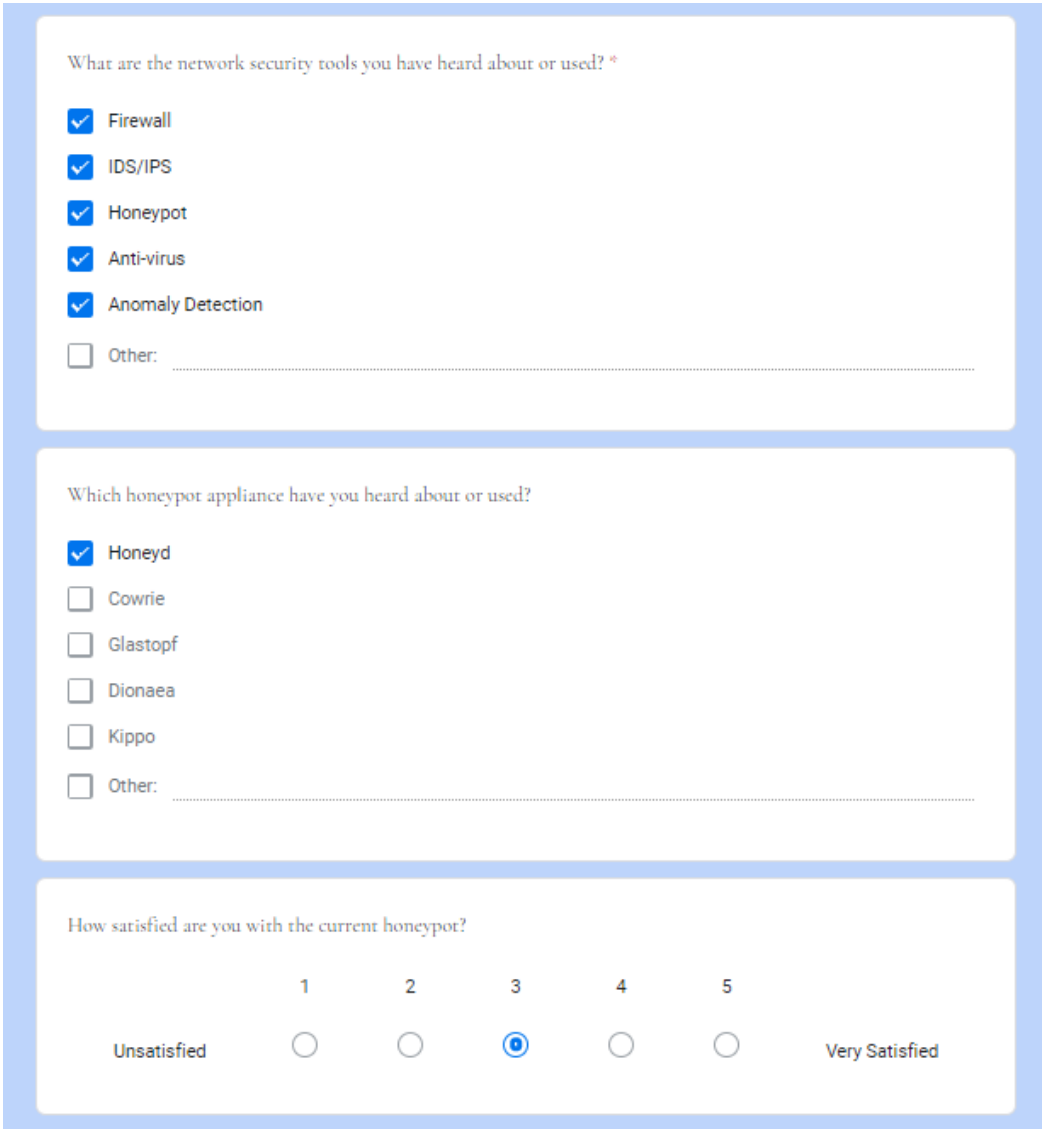


Figure 60 Sample of Pre-Survey Form 4

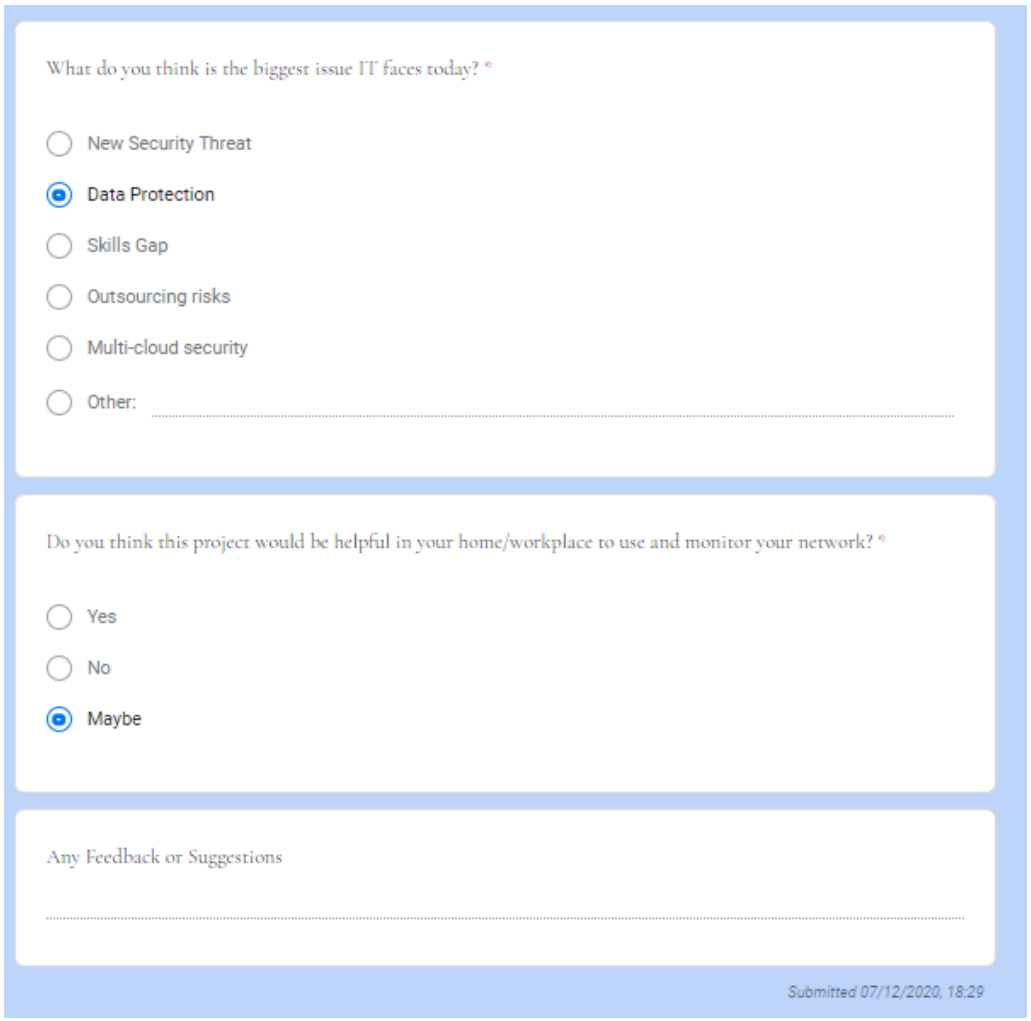


Figure 61 Sample of Pre-Survey Form 5

7.1.3 PRE-SURVEY RESULT

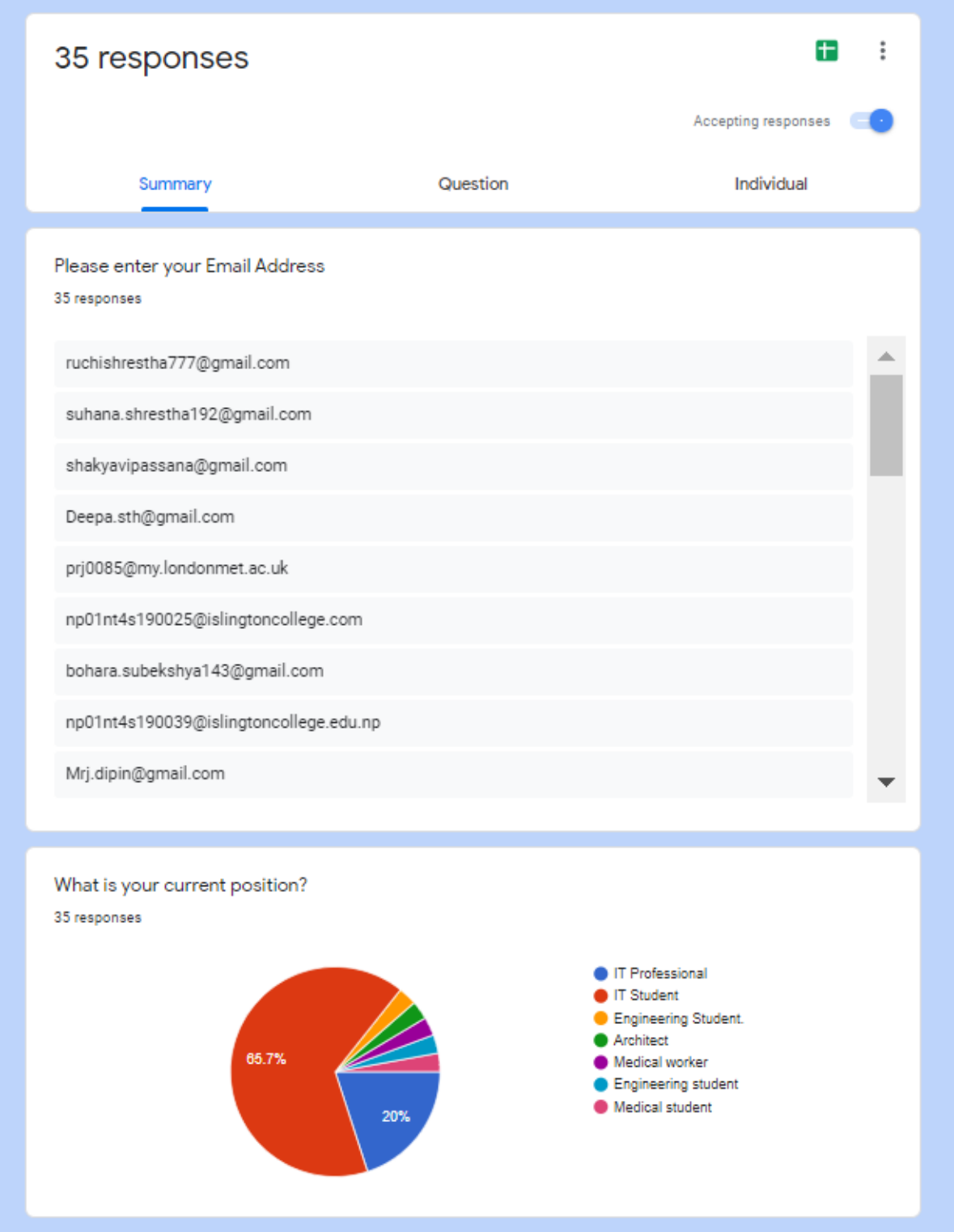


Figure 62 Result of Pre-Survey Form 1

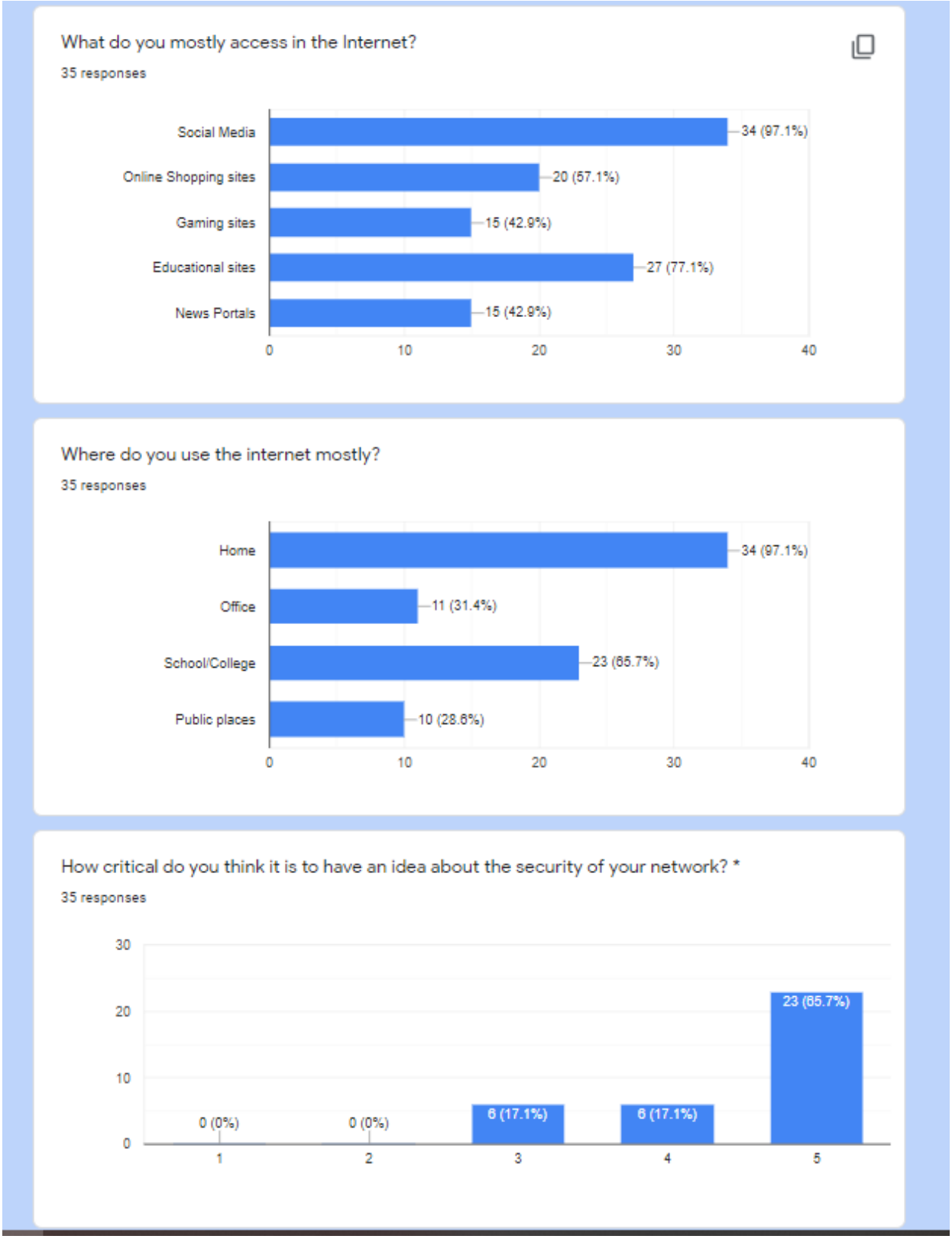


Figure 63 Result of Pre-Survey Form 2

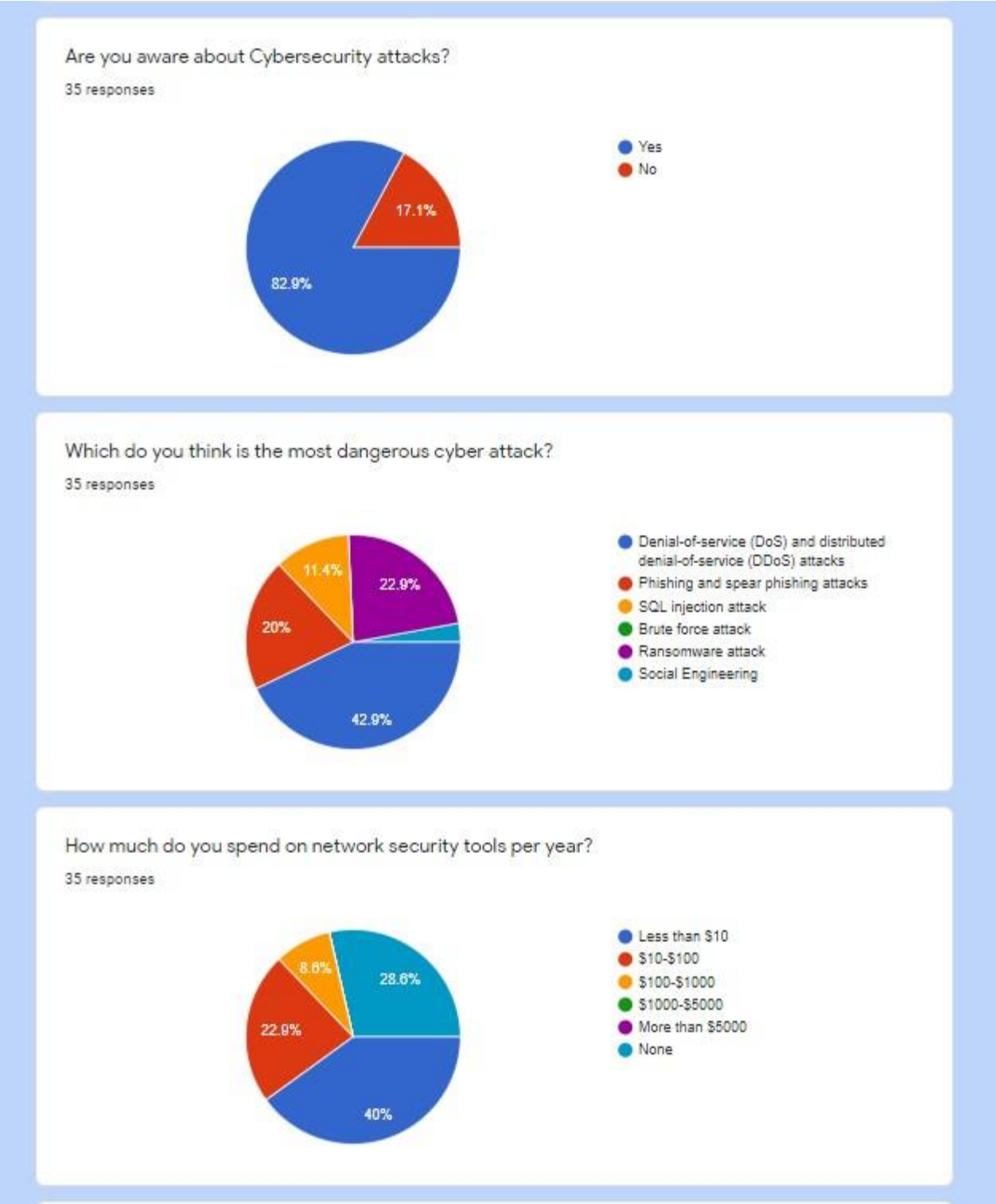


Figure 64 Result of Pre-Survey Form 3

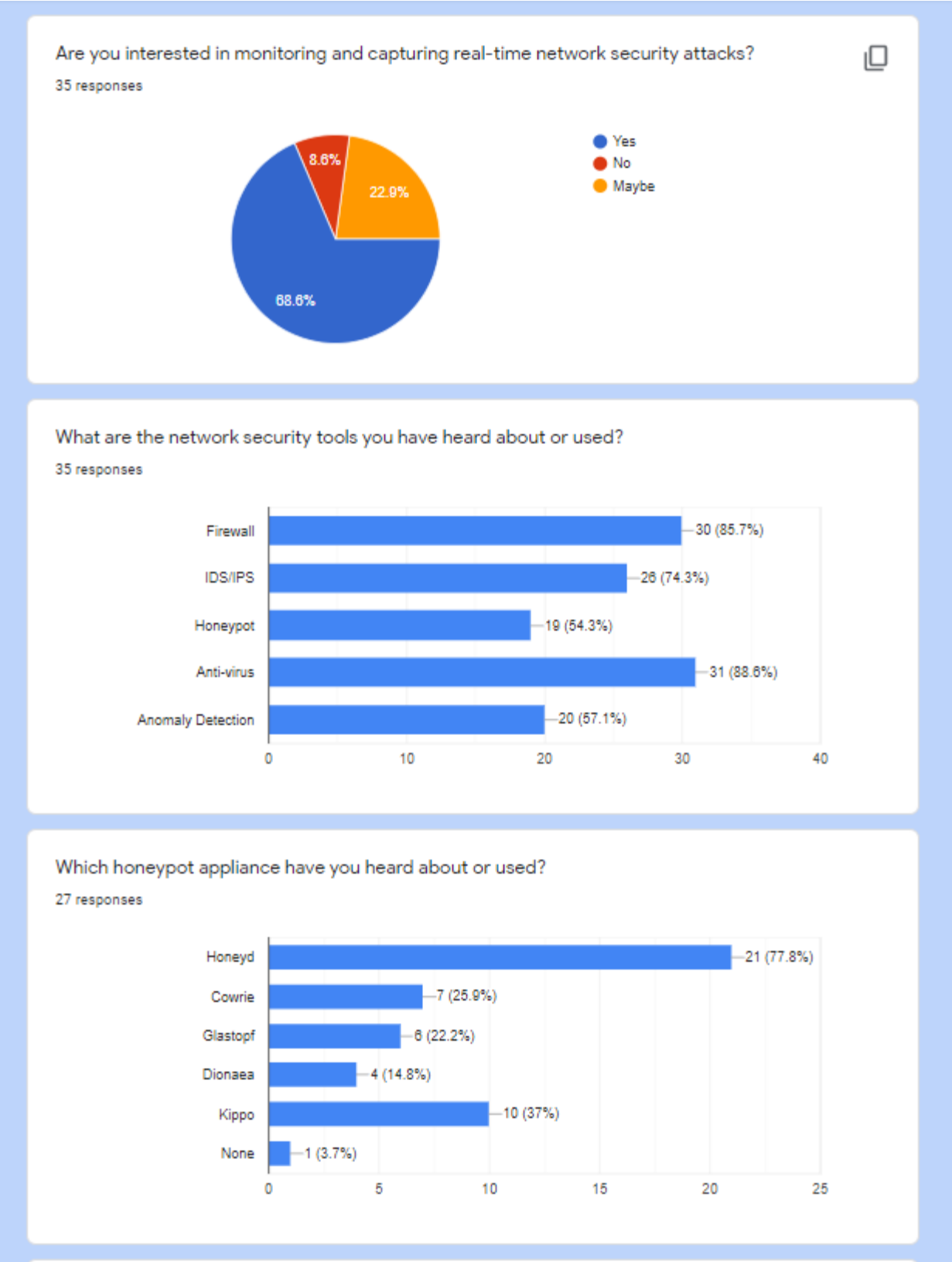


Figure 65 Result of Pre-Survey Form 4

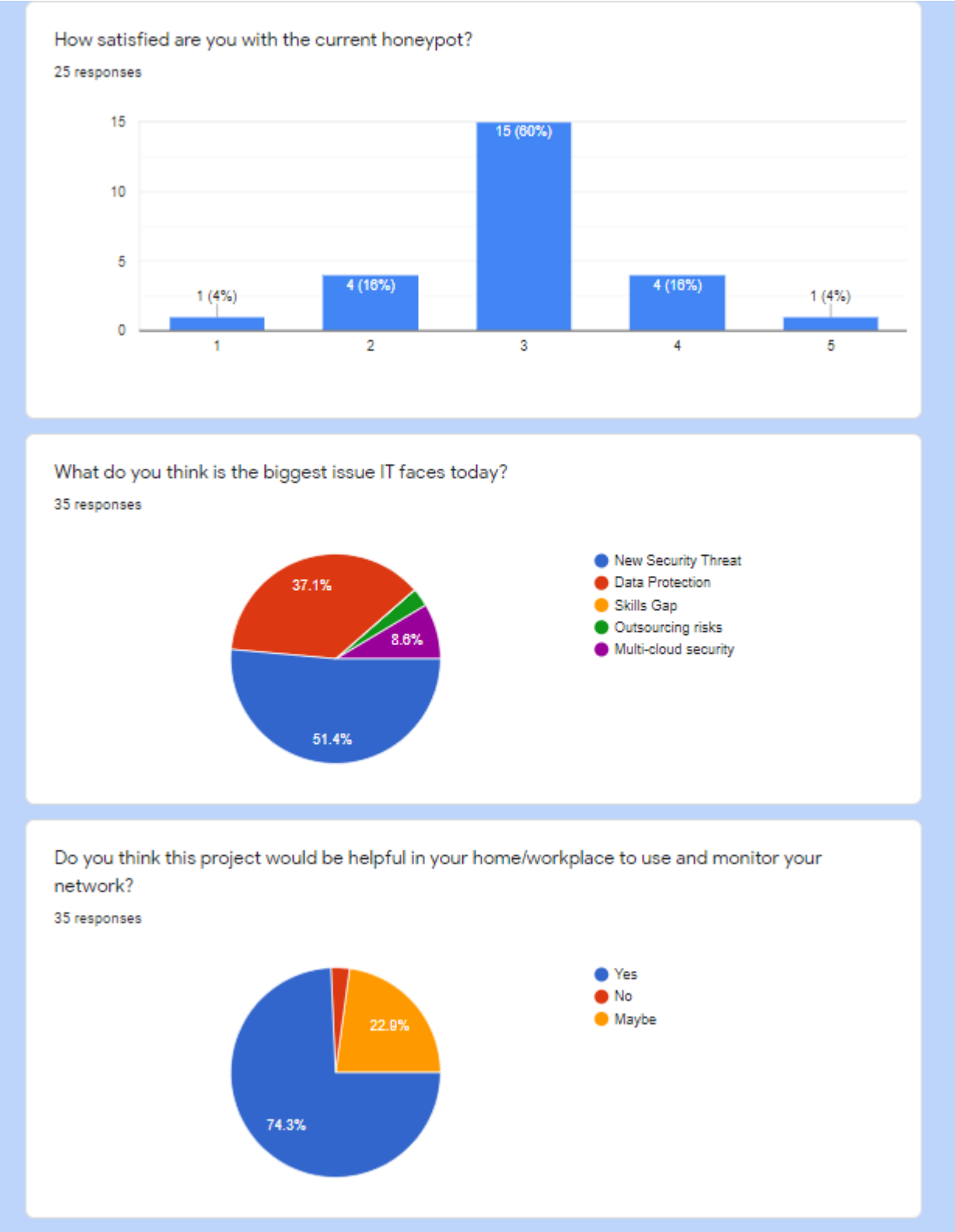


Figure 66 Result of Pre-Survey Form 5

Figure 67 Result of Pre-Survey Form 6

GO BACK TO TOP

7.2 APPENDIX B: POST-SURVEY

7.2.1 POST-SURVEY FORM

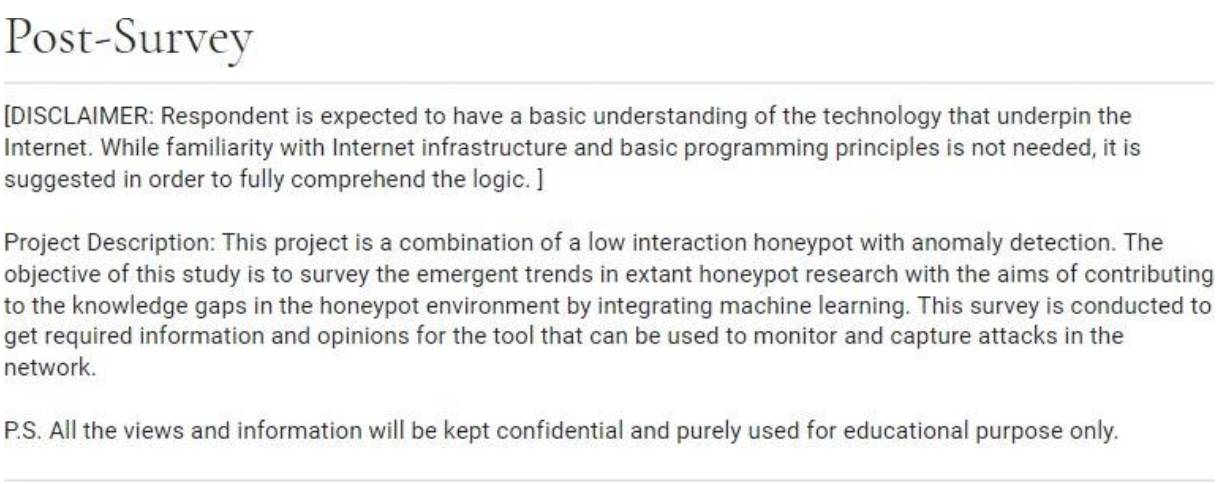


Figure 68 Post-Survey Form 1

Figure 69 Post-Survey Form 2

Figure 70 Post-Survey Form 3

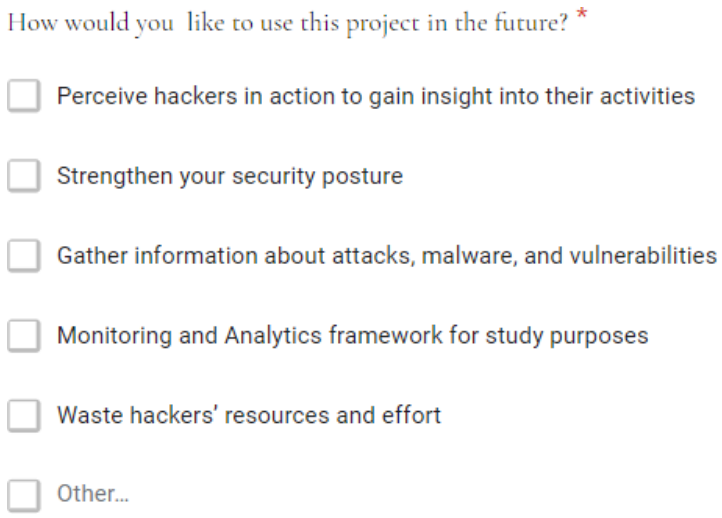


Figure 71 Post-Survey Form 4

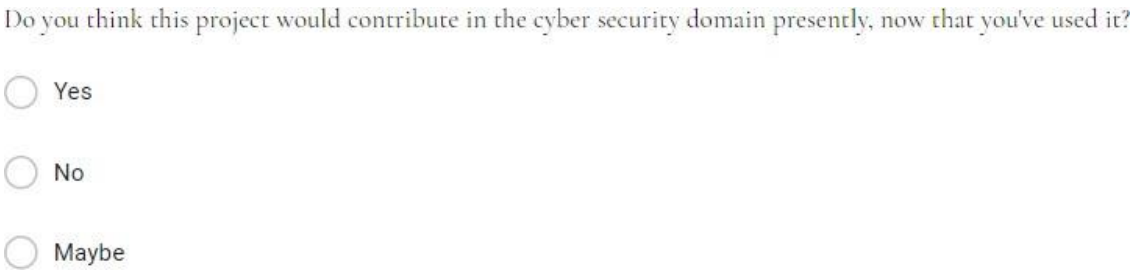


Figure 72 Post-Survey Form 5

Figure 73 Post-Survey Form 6

Which feature do you think is the best in this project? *

- [ ] Data Capture
- [ ] Data Control
- [ ] Whitelisting/Blacklisting Log Interface
- [ ] Security event logging
- [ ] Admin Alert
- [ ] Machine Learning Interface
- [ ] Visualization
- [ ] Accuracy & Prediction
- [ ] None

Figure 74 Post-Survey Form 7

Which feature is your least favorite in this project? *

- [ ] Data Capture
- [ ] Data Control
- [ ] Whitelisting/Blacklisting Log Interface
- [ ] Security event logging
- [ ] Admin Alert
- [ ] Machine Learning Interface
- [ ] Visualization
- [ ] Accuracy & Prediction
- [ ] None

Figure 75 Post-Survey Form 8

Figure 76 Post-Survey Form 9

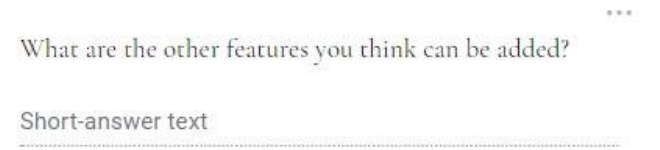


Figure 77 Post-Survey Form 10

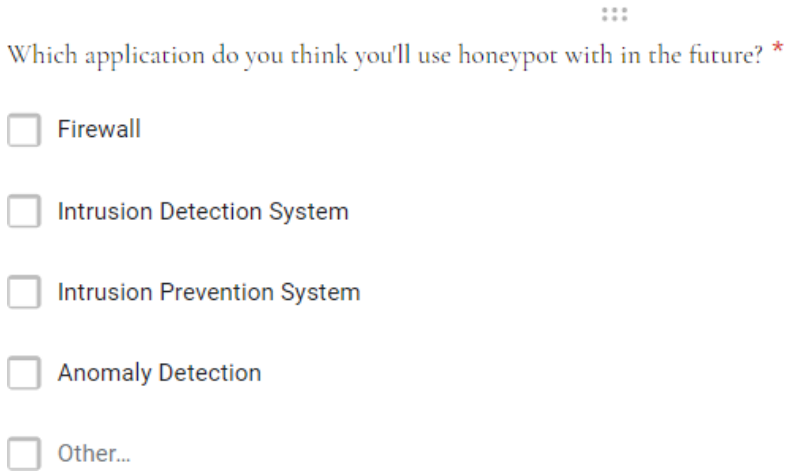


Figure 78 Post-Survey Form 11

Figure 79 Post-Survey Form 12

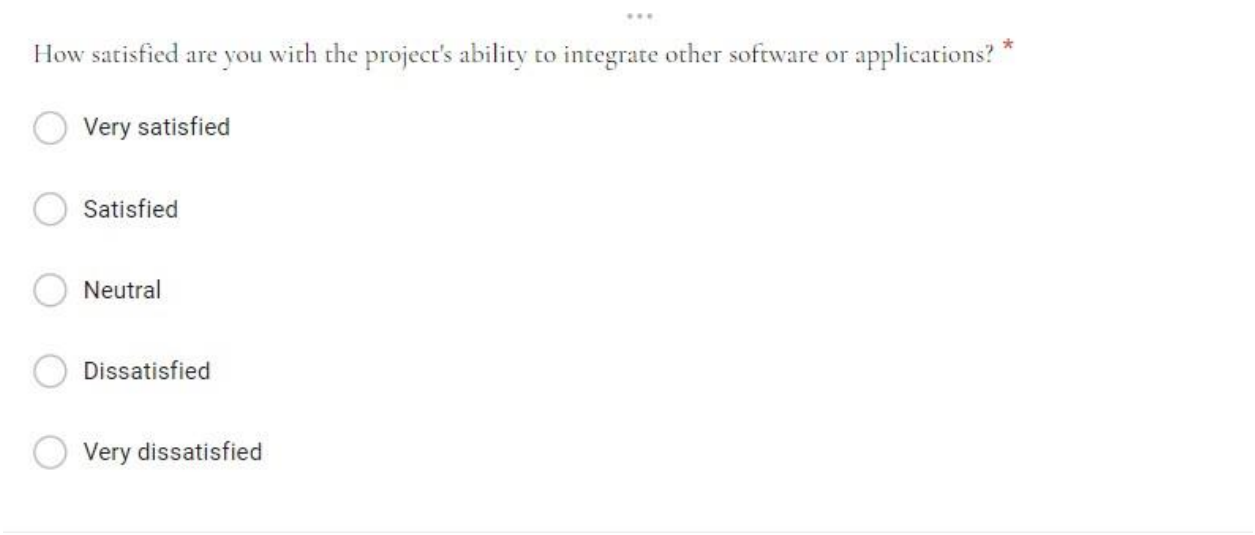


Figure 80 Post-Survey Form 13

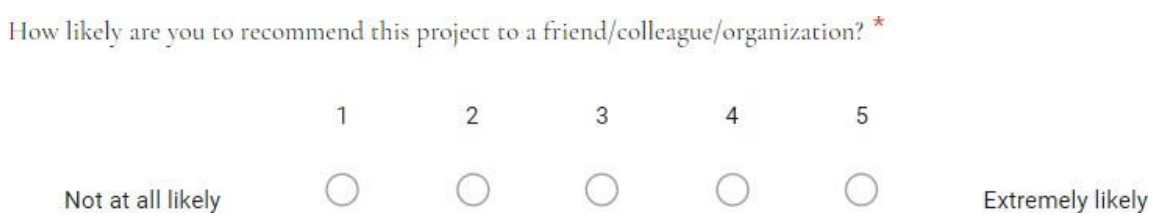


Figure 81 Post-Survey Form 14

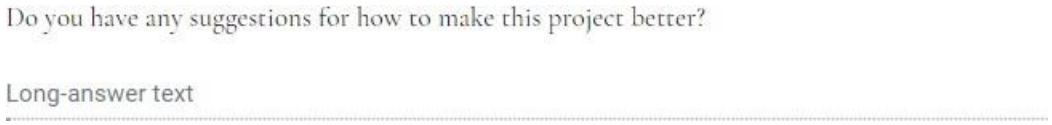


Figure 82 Post-Survey Form 15

7.2.2 SAMPLE OF FILLED POST-SURVEY FORMS

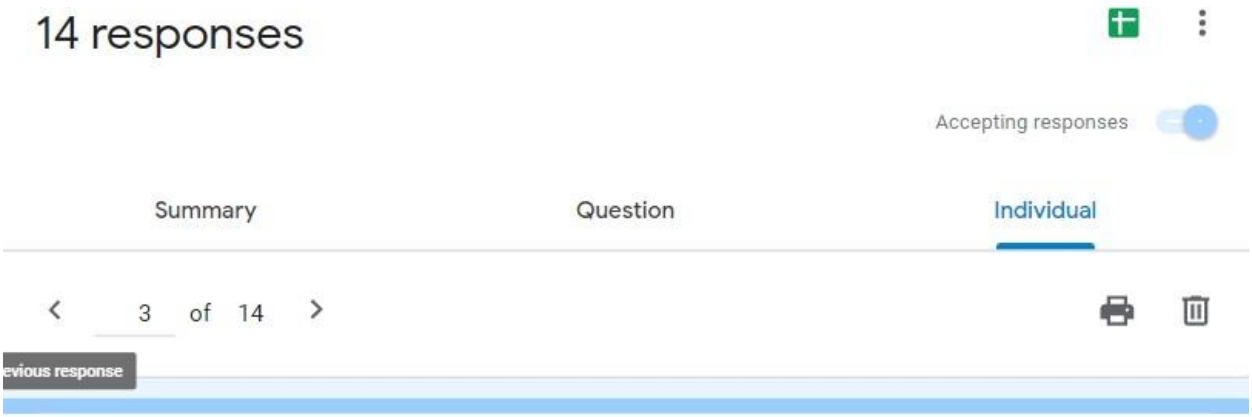


Figure 83 Sample of Post-Survey 1

Figure 84 Sample of Post-Survey 2

Figure 85 Sample of Post-Survey 3

Figure 86 Sample of Post-Survey 4

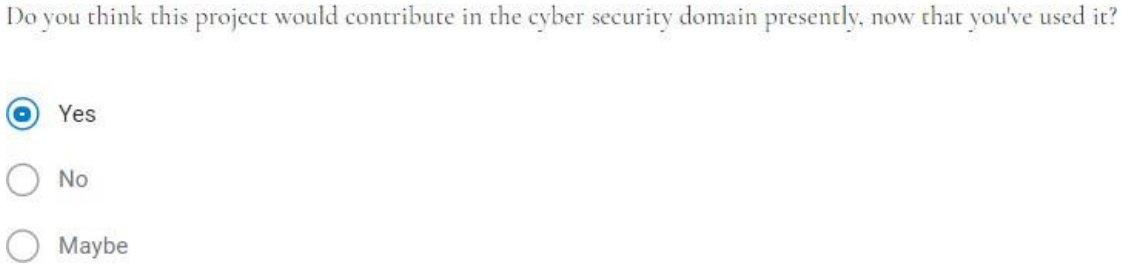


Figure 87 Sample of Post-Survey 5

Figure 88 Sample of Post-Survey 6

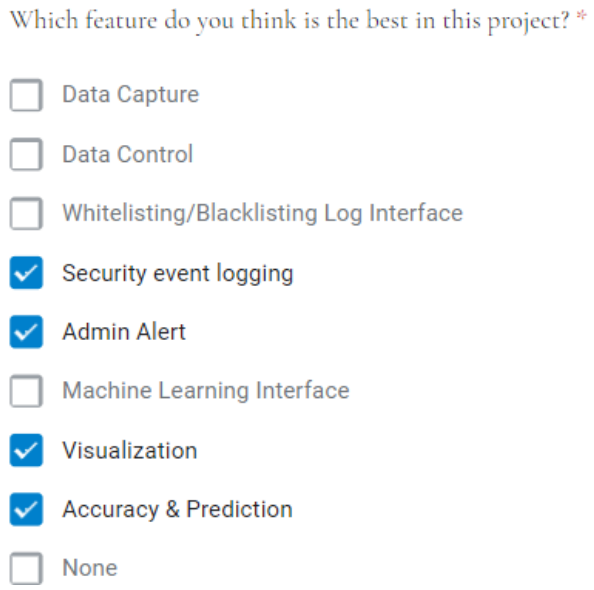


Figure 89 Sample of Post-Survey 7

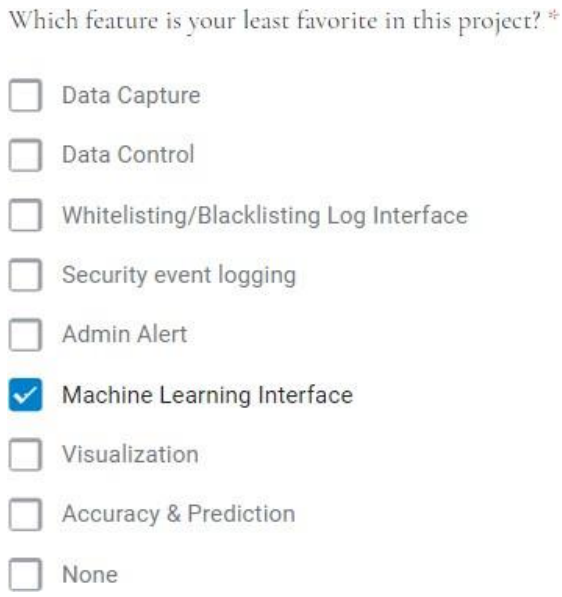


Figure 90 Sample of Post-Survey 8

Figure 91 Sample of Post-Survey 9

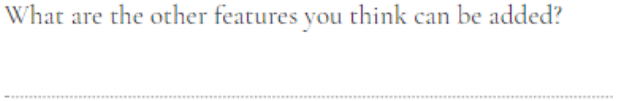


Figure 92 Sample of Post-Survey 10

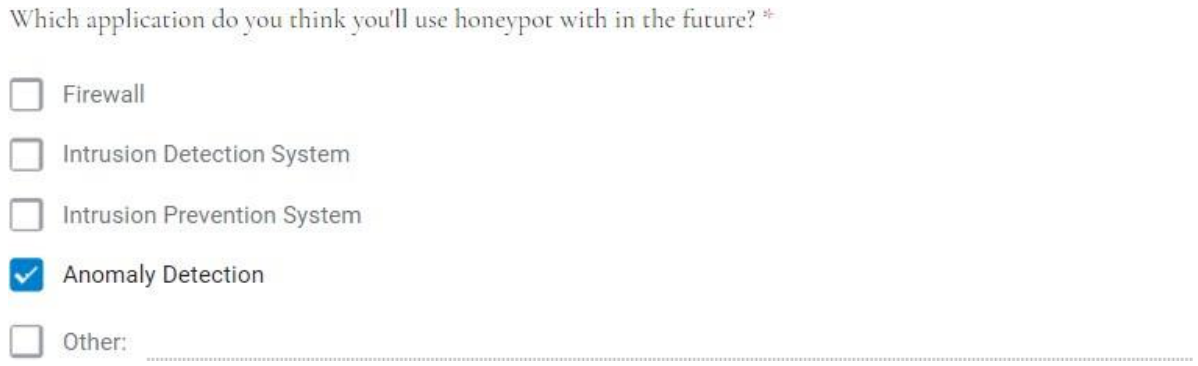


Figure 93 Sample of Post-Survey 11

Figure 94 Sample of Post-Survey 12

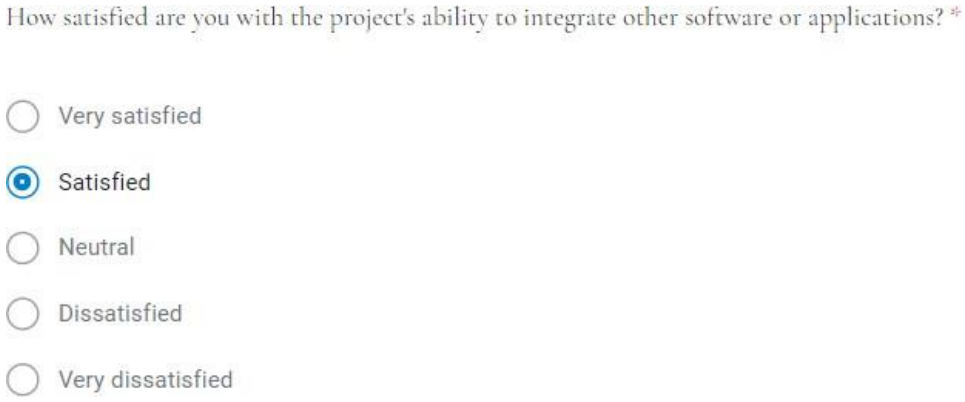


Figure 95 Sample of Post-Survey 13

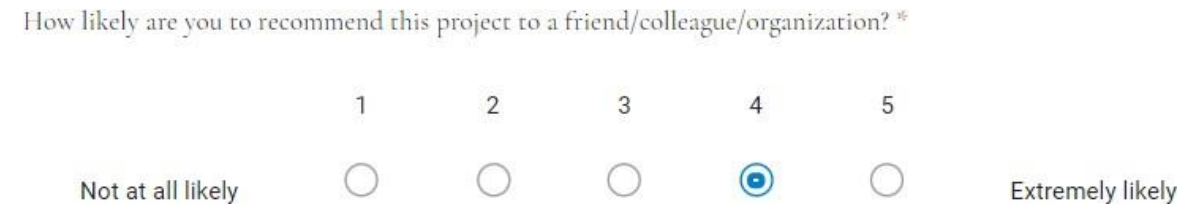


Figure 96 Sample of Post-Survey 14

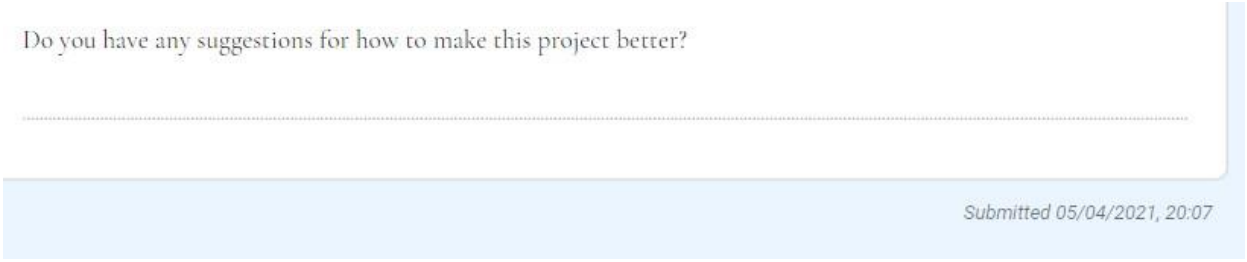


Figure 97 Sample of Post-Survey 15

7.2.3 POST-SURVEY RESULT

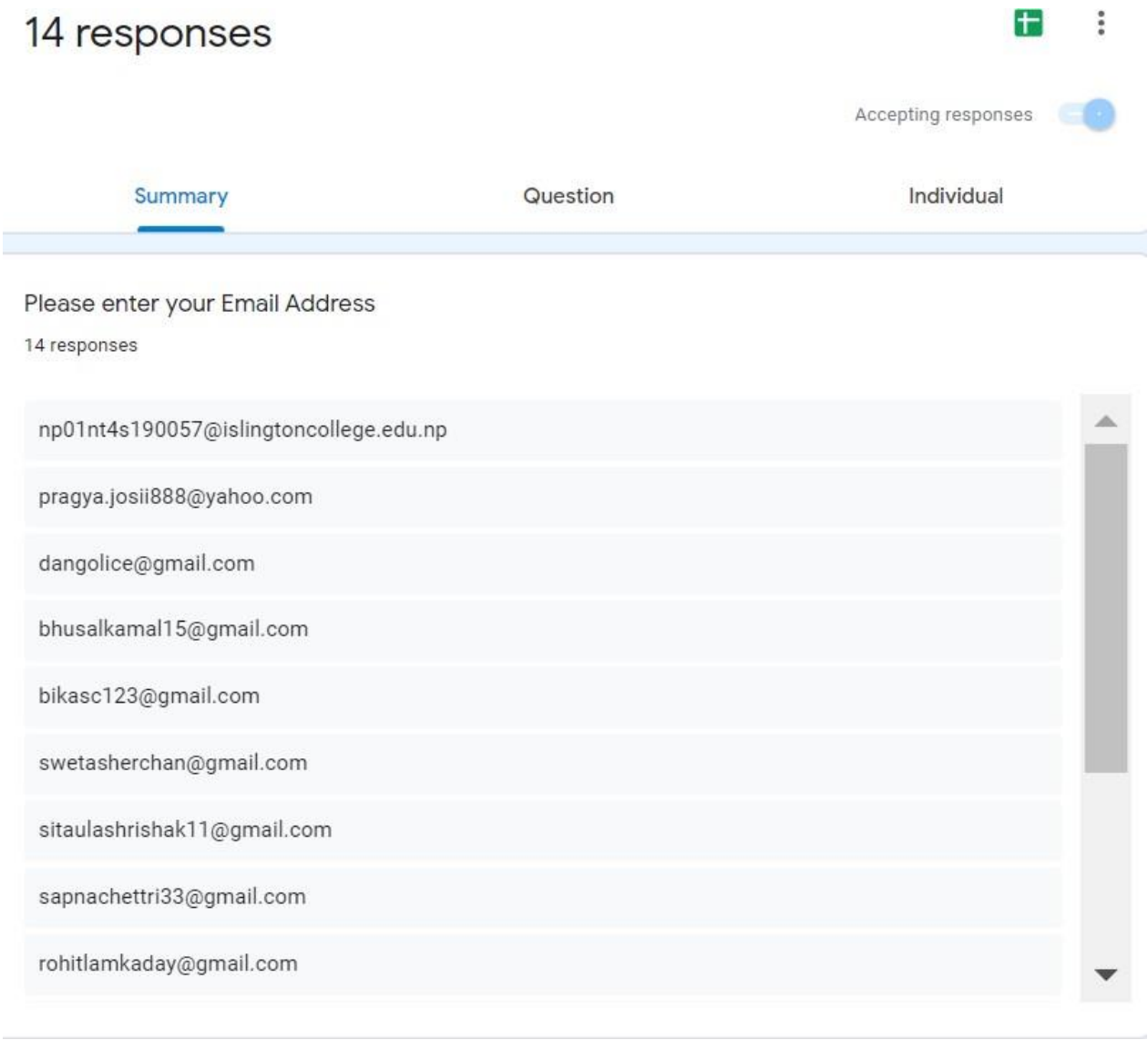


Figure 98 Result of Post-Survey 1

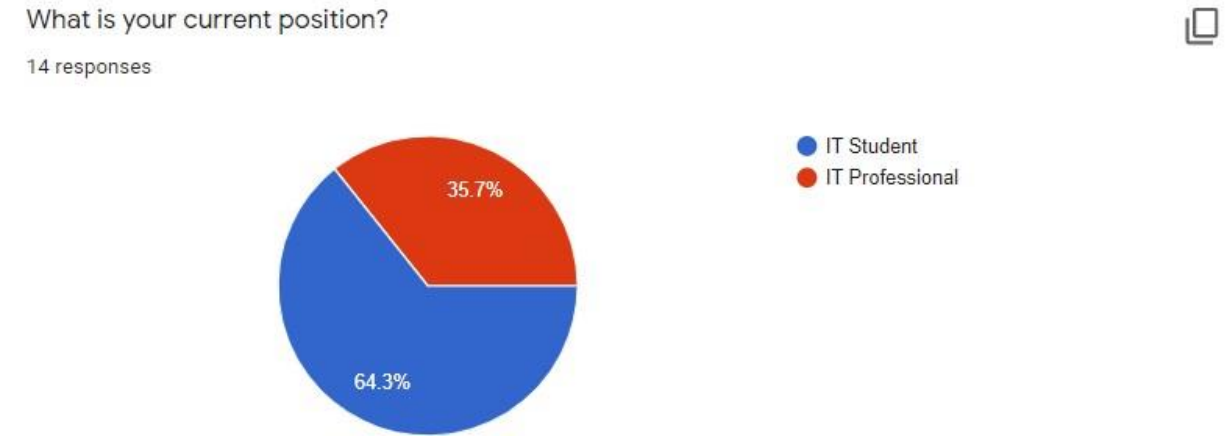


Figure 99 Result of Post-Survey 2

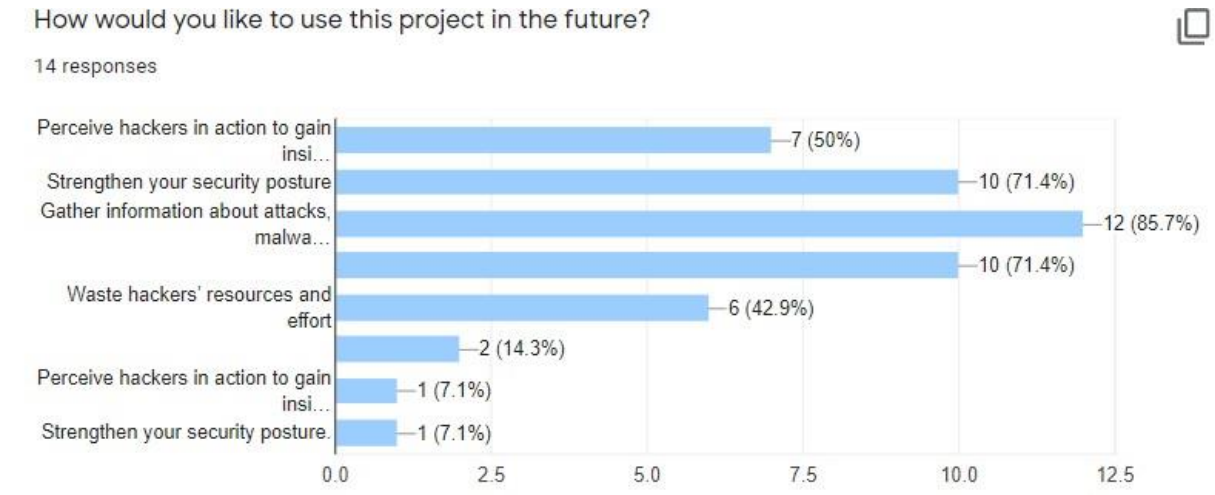


Figure 100 Result of Post-Survey 3

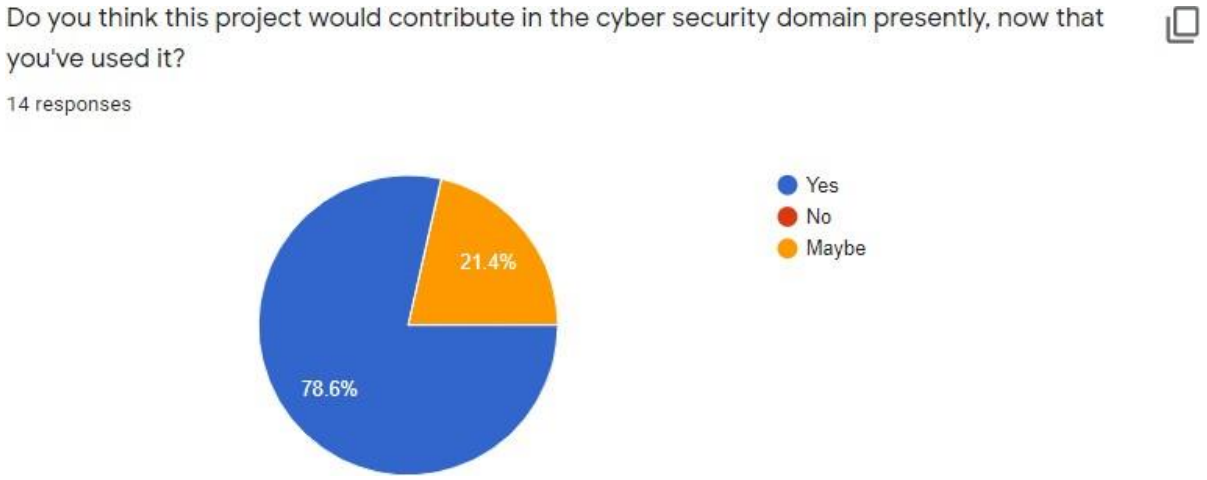


Figure 101 Result of Post-Survey 4

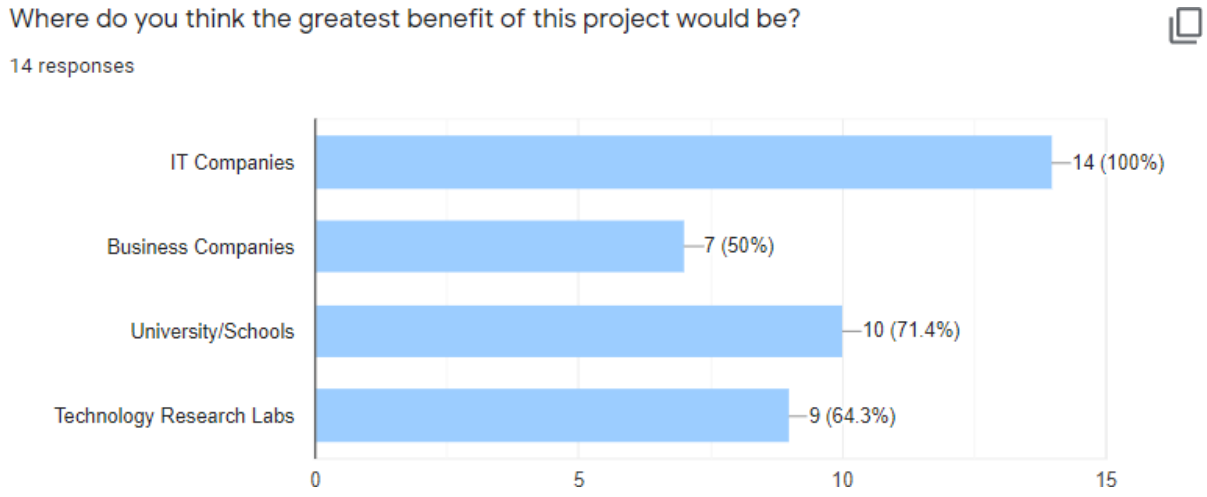


Figure 102 Result of Post-Survey 5

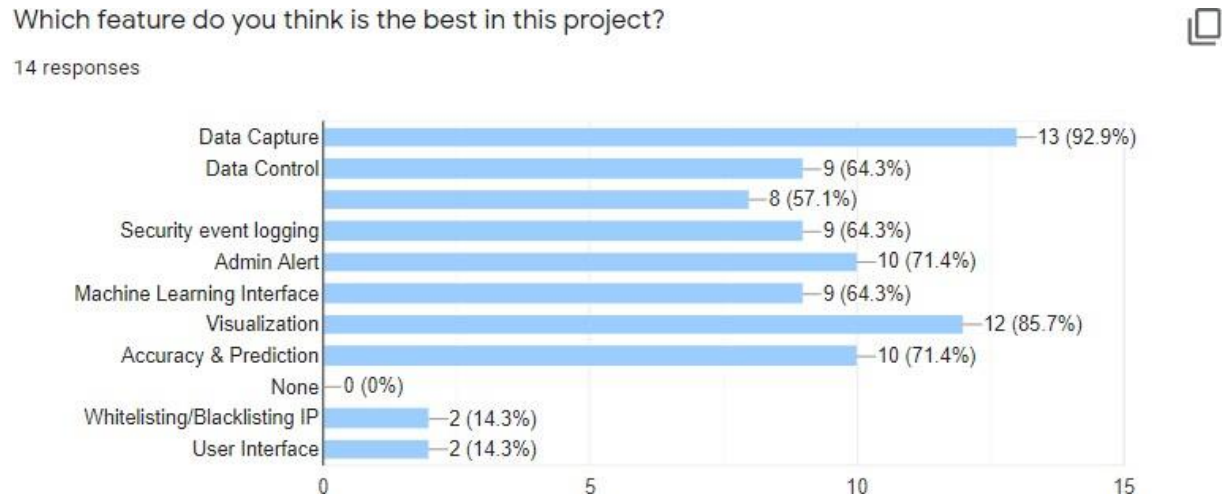


Figure 103 Result of Post-Survey 6

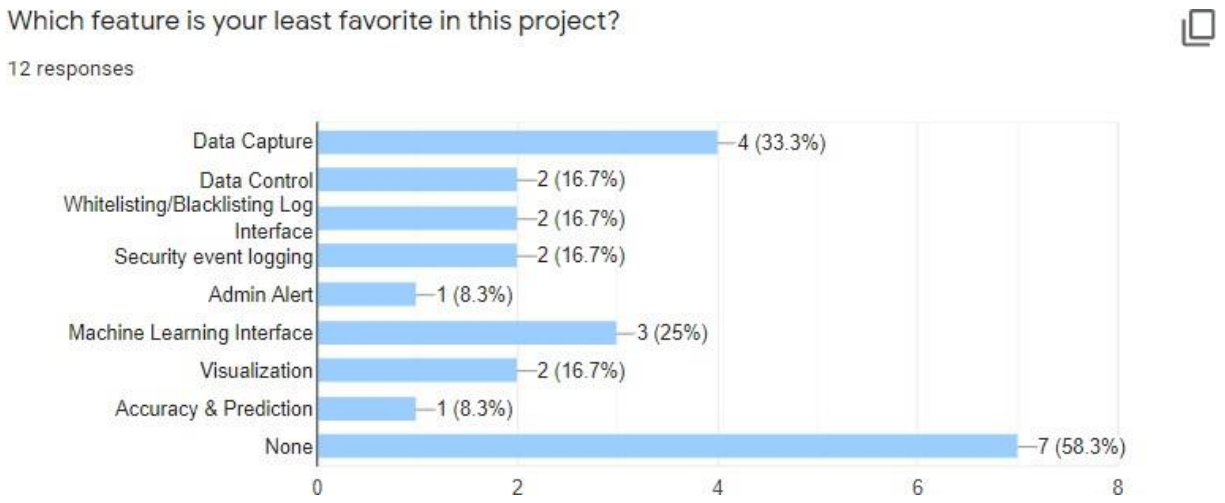


Figure 104 Result of Post-Survey 7

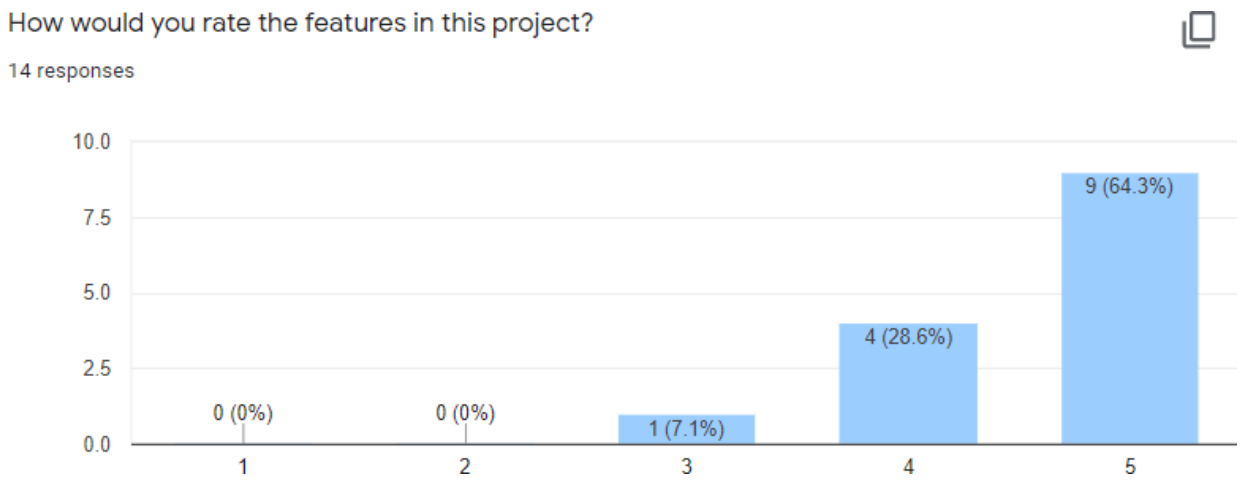


Figure 105 Result of Post-Survey 8

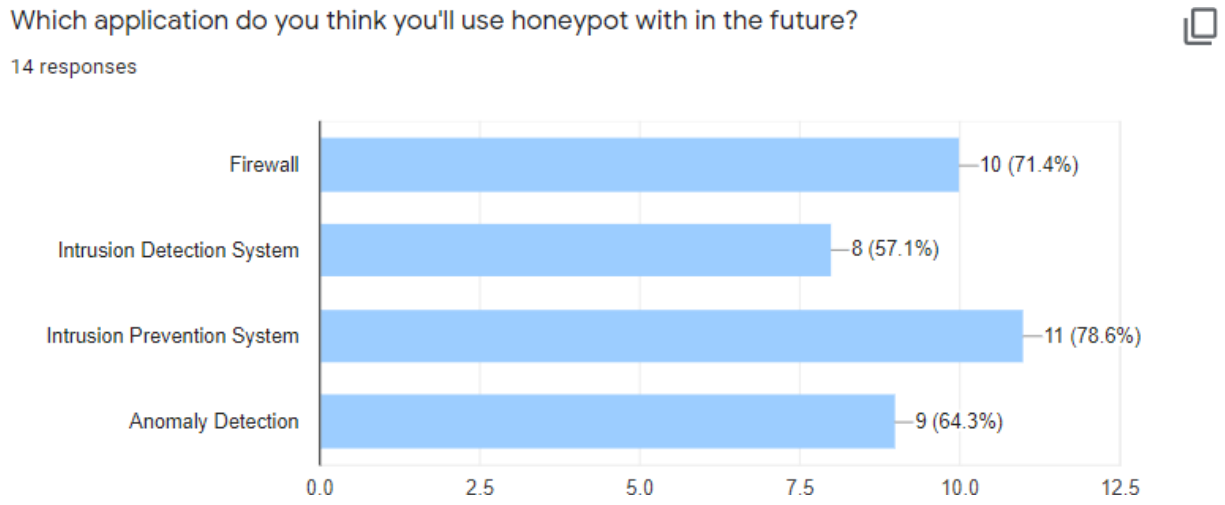


Figure 106 Result of Post-Survey 9

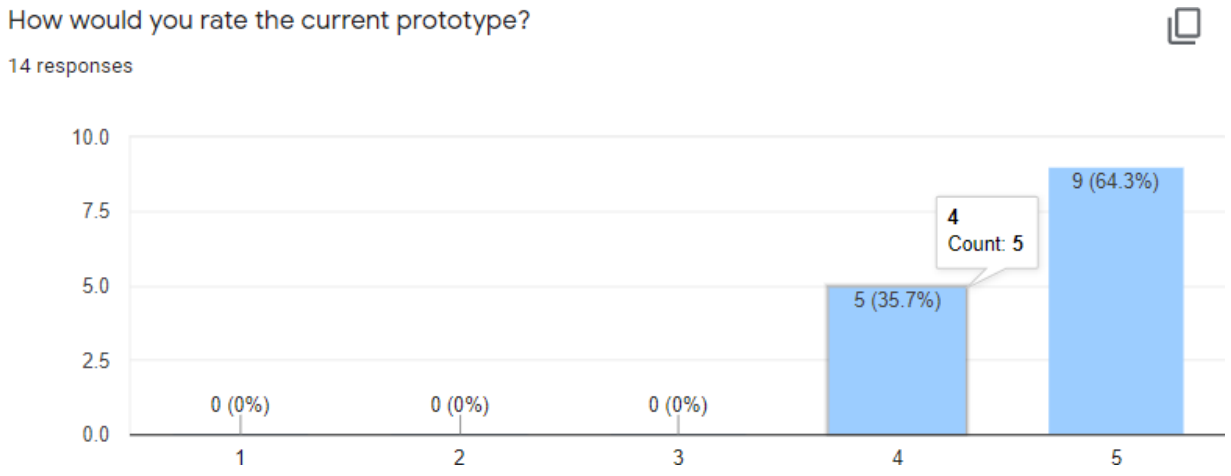


Figure 107 Result of Post-Survey 10

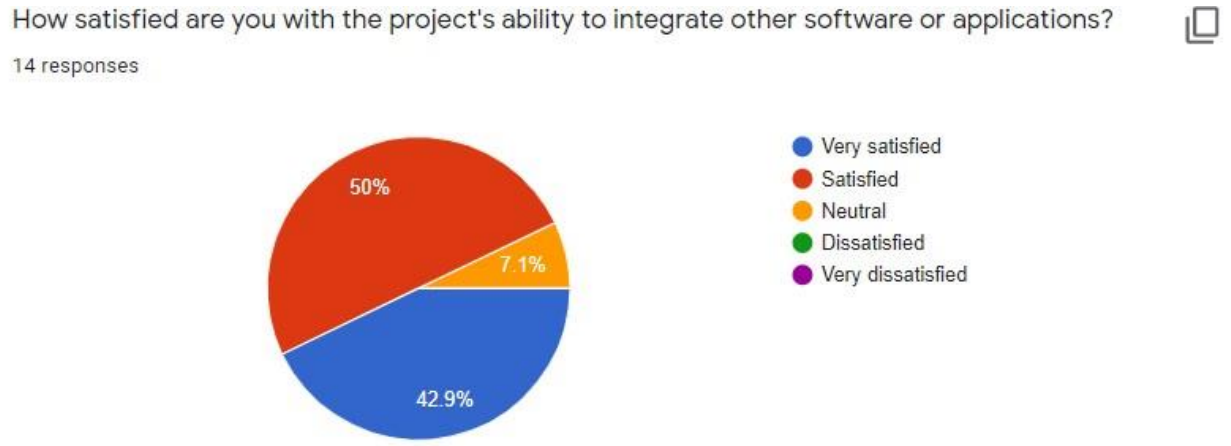


Figure 108 Result of Post-Survey 11

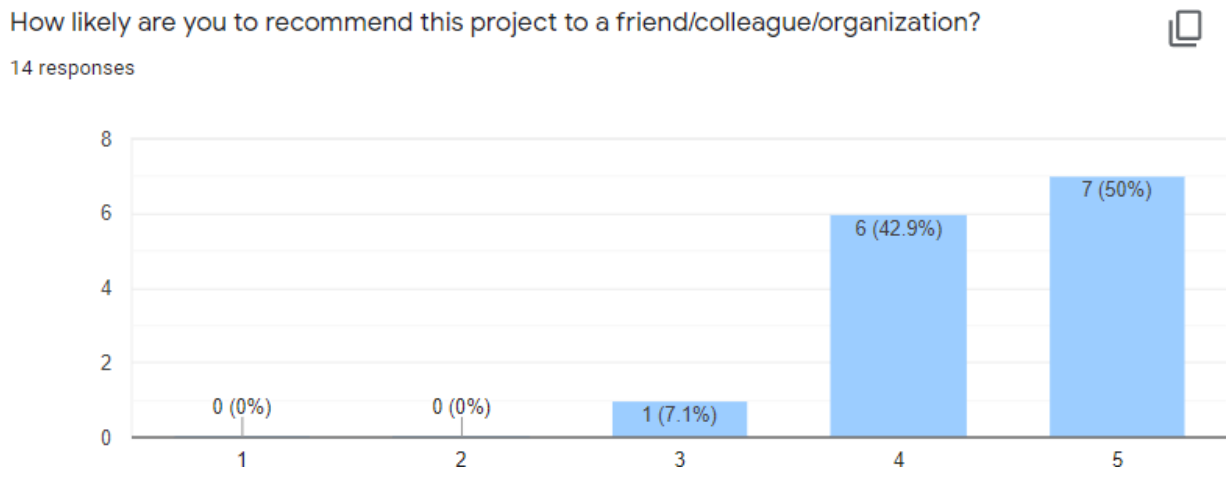


Figure 109 Result of Post-Survey 12

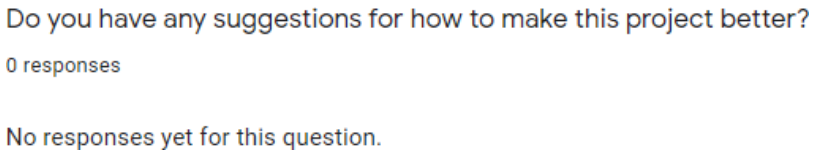


Figure 110 Result of Post-Survey 13

GO BACK TO TOP

7.3 APPENDIX C: PROJECT DESCRIPTION

The main purpose of the project is to detect attacks in networks carried out by intruders. Even though the attack is well identified and tests and patches are obtainable, here an attack is considered novel if the vulnerability is unknown to the target's owner. This project is a platform-independent framework that runs on Windows and Linux OS. The aim is to detect, not prevent. Honeypot will use the local firewall to detect and block any intrusion attempts, but the main purpose is to simply identify such traffic. In the lack of rules to define such traffic, this is a delicate task. In the manner outlined above, this study will attempt to analyse and address questions about the use of honeypot technology. Will it bring something innovative to secure the network? Would it hold up in a live production setting? Is it possible to make it more user-friendly? What resources would be required? It will go through a variety of configurations for the development, as well as address the issues in this domain, it will strive to provide a critical review by indicating what requires to be done, and improved, for it to function better.

GO BACK TO TOP

7.4 APPENDIX D: SYSTEM ARCHIETECTURE

The first step taken to encourage the attacker towards the honeypot is by making it vulnerable i.e. by opening a TCP port on 31337. If a connection is detected to the system the three-way TCP handshake has been initiated. As the intruder sends the requests to the system it becomes a connection request and similarly, the system generates a reply. This might make the intruder assume some dedicated server confinement. It works this way because of the absence of real service to log in and the honeypot merely records the direct connections to the fake services instead of delivering a TCP server. Inside the honeypot server, the log module stores the information of the intruder who tries to connect with the honeypot. When an attacker interacts with the ports, it generates a TCP connection and then shuts it with an RST packet. Any connections to the honeypot are considered to be suspicious and are logged in for tracking purposes. It logs the IP address and the status of the connection and automatically uploads it to the database. The database is designed to be highly simple and easy to expand. With the alert module when the system runs, it checks to monitor the log file and when it notices the change in the size of the file; the system administrator will be alerted. Any, new changes in the honeypot will be logged and sent automatically in the form of a text file to the system administrator through email. The utility of this framework can provide users with a range of benefits. The honeypot web interface also is built for viewing the blacklisted and whitelisted logs captured by the honeypot. It is built with the help of HTML, CSS, and PHP. Any changes to logs can be easily viewed in the interface.

GO BACK TO TOP

7.5 APPENDIX E: WORKING MECHANISM

This is an interactive project with a platform for service development and execution that will be published under the GNU GPLv3 license. During its execution, there will be two python applications deployed with all their dependencies packed within the packages. So, users do not have to download every module/library separately. The working mechanism of a honeypot is also very simple to comprehend. Once the package is installed in the user's virtualized machine, the user should provide some basic configuration manually such as the number of protocol(s) s/he wants to open for the attacker, the whitelisted IP address, email address, and its password of the admin. By default, the system is set in port 31337 and 127.0.0.1 as whitelisted IP. Then when the main.py script is booted with single line command from the terminal, external network traffic is routed via the internal network during this step. The router is in charge of communicating the external network to the internal network as well as routing packets to the honeypot. Honeypot will intercept traffic packets entering the internal network and store them on a fake TCP server. The collected packets are then written into log files i.e. text and CSV files for analysis. After that, the text file of the log is attached to the email for alerting and backing up the log files to the admin. The admin can easily view the logs in a web interface which automatically adds the log in the database and displays it in the browser.

Working with the Anomaly Detection UI is also very easy. A small Streamlit command is needed to be sprinkled over the Python code. A local Streamlit server will start up as soon as the user runs the script, and the app will launch in a new window in the user's default web browser (Streamlit Inc. , 2021). There are primarily three sections in the interface namely, Data Interpretation, Analysis and Visualization, Classification. With the help of statistical analysis and visual representation, the essential method of conducting initial investigations on data to uncover patterns, spot anomalies, test theories, and check assumptions is conducted with machine learning (Patil, 2018).

GO BACK TO TOP

7.6 APPENDIX F: SIMILAR PROJECTS

7.6.1 HONEYCOMB

Findings from the Study

Honeycomb is one of the few tools that use only the attack requests intercepted by a honeypot and modifies the existing manual method of analysing the collected information (Kreibich & Crowcroft, 2003). Honeycomb shares the limitations as with Honeyd (Kreibich & Crowcroft, 2003).

Analysis of the given study following the Final Year Project

The traffic signatures generated by Honeycomb are specific which can help security analysts to manually inspect the traffic. However, AADH (this project) system is aimed to detect and predict new vulnerabilities that cannot be detected by using datasets collected from security systems. Also, the system helps to block access to the attacker after a new connection has been detected in the honeypot which makes it more secure and reliable.

GO BACK TO TOP

7.6.2 SPECTER

Findings from the Study

Specter can be set up and configured very easily while featuring the most sophisticated features. Without human input, automated honeypot content updates and vulnerability databases enable the honeypot to continually adjust.

Analysis of the given study following the Final Year Project

Specter is simple and a handy tool but the risks associated and the maintenance time required is much higher. Unlike Specter, the AADH framework is easy to set up and maintain. It will not be targeting the majority of protocols but this prototype will be a great tool for monitoring and capturing patterns in the network.

GO BACK TO TOP

7.6.3 'AN INTELLIGENT HONEYPOT'

Findings from the Study

Honeypots are drawing analysts' attention as a beneficial preventive measure that can be applied to minimize cyber threats and offers an incentive to understand as much about the essence of these threats, according to the findings. As a result, a honeypot could be used as a data collection method for security threats.

Analysis of the given study following the Final Year Project

On the contrary to the ‘An Intelligent Honeypot’ project, this project supports Windows and Linux environments where the honeypot not only detects but performs actions, and rather than accepting only two ports, it accepts multiple ports for the attacker to connect. It addressed the problem by collecting log files from the attack scene and visualizing and measuring the accuracy using machine learning in a web interface.

GO BACK TO TOP

7.7 APPENDIX G: SIMILAR STUDIES

7.7.1 'CONTEXT-AWARE HONEYPOT'

Findings from the study

The tools which are used in the project are all open sources which makes it much more cost-effective than a proprietary tool. The HIDPS uses the LSTM classifier with an accuracy of 99.95 percent and 99.53 percent, and because of the altogether accuracy of 99.59 percent, the background switcher can use the Linear SVM classifier. The ability to detect malicious traffic or a future XSS attempt with a pre-trained model of LSTM allows it to be a useful tool for the information analyst to prevent the attack on time (Aggarwal, 2019).

Analysis of the given study following the Final Year Project

In the development of network security software, in the particular honeypot, the scope of this study was taken a step further. Alike to this project, the aim to build a computer network system that by learning and analysing the actions of the attackers, can safeguard itself from known and unknown attacks. However, this tool is platform-independent which is built and planned to run in the Windows and Linux environments. This project plans to use Decision Tree and Logistic Regression as its classifiers which produces high precision and accurate results. A user-friendly application is also built which is incredibly easy to use and learn machine learning effectively.

GO BACK TO TOP

7.7.2 'DDOS MITIGATION AND INTRUSION PREVENTION USING HONEYPOTS'

Findings from the Study

Unlike IPS, IDS does not block attacks but can alert the administration instead. This then modifies the situation when it's discovered that the blocked IP tried to access the content, then at the first attempt, it will be denied. If some false positive IP has been detected somewhere then honeypot will examine it and take the appropriate action. Another benefit of the implementation of the model is decreased costs (Rahman, et al., 2019).

Analysis of the given study following the Final Year Project

On the contrary to this initiative, the honeypot in this project is planned to make it more secure and adaptive by focusing mainly on intrusion and anomaly patterns in the network. Moreover, it is an adaptive system that provides a quicker response towards malicious attacks.

GO BACK TO TOP

7.8 APPENDIX H: REQUIREMENT SPECIFICATIONS

7.8.2 REQUIREMENTS ANALYSIS

7.8.2.1 SYSTEM OVERVIEW

This program will be implemented in a Windows and Linux environment, with Python as the primary programming language, and will require the user to have administrative privileges. The following are some of the basic features that will be implemented:

A simple honeypot server will generate logs and alert the admin if it receives a connection from the external party. Honeypot will use the local firewall to detect and block any intrusion attempts, but the main purpose is to simply identify such traffic. The logs will be automatically stored in the database for security purposes. The logs can be viewed in a text file or CSV file for in-depth analysis and study. A dataset should be selected for the anomaly detection where the data was trained and test for high precision and accurate results. The user-friendly web application for machine learning is incredibly easy to use and learn machine learning effectively with the help of Streamlit.

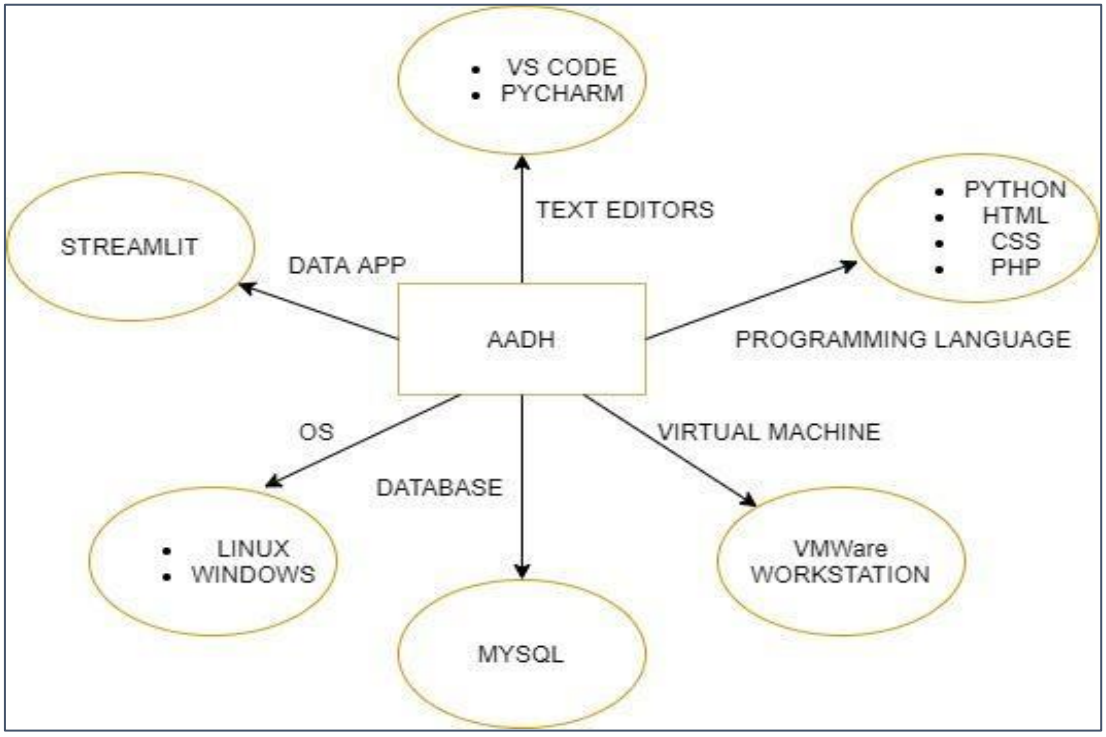


Figure 111 Major tools used in the development

8.8.2.2 DEPENDENCIES

Developing a detailed profile of specifications helps to choose the best requirements for the development of the project. Therefore, to ensure the project is completed on time, the following were primarily utilized.

8.8.2.2.1 SOFTWARE DEPENDENCIES

Virtual Machine- VMware Workstation has been used for setting up the virtual environment to run Windows and Linux OS on the physical machine. It is one of the best hypervisor tools for various platforms to install and use mainly for testing purposes. For this project, the Operating Systems were installed in the VMware and the scripts were running into the system with some particular considerations.

Text Editors- Text Editors such as PyCharm and Visual Studio Code have been used for developing different pieces of code for honeypot and machine learning. Both editors provide a set of Python programming features that any programmer should have in their toolkit. PyCharm is a useful platform with a range of functionality that renders Python creation a snap. Even so, almost all of the features used in this project are accessible in VS Code, except for remote debugging, which is also being developed in VS Code.

Operating Systems- Being a platform-independent project, it supports Windows and Linux Operating Systems without requiring any modifications in the script. Since most attacks would randomly strike honeypot regardless of whether it contains a specific piece of software, these operating systems were selected. To build an exploitable honeypot, it must have insecure flaws and be reachable through the Internet (Grimes, 2005). Although a few Windows honeypot solutions exist, they don't have the sophistication, feature set, or support that their UNIX/Linux counterparts offer. As a result, the Windows operating system was chosen later in the project (Grimes, 2004).

Programming Languages- Python3 has been used as the main programming language for the development of the whole project. Python3 was chosen due to its ease of use and strong community support. It also has numerous sets of libraries that support in range in development needs. Also, HTML, CSS, and PHP were chosen for the development of web pages that dynamically interact with the database.

Protocol- Protocols such as HTTP protocol is used for capturing requests from attackers over the network from major browsers and SMTP protocol is used for alerting the system admin if there is any intruder. The initial bytes sent by the intruder are either GET or CONNECT, which is used to auto-detect HTTP. When an HTTP CONNECT request is made, it always returns to the honeypot. When an email is sent, it is delivered over the network via SMTP from one server to the next. Simply stated, an SMTP email is sent via the SMTP server.

Xampp- Xampp is by far one of the best tools that can assist in setting up an environment in which Apache, MySQL, and PHP are configured in a matter of minutes and with minimal effort. XAMPP includes MySQL, one of the most commonly used relational database management systems (IONOS, 2019). MySQL provides data storage for web servers when used in conjunction with the Apache web server and the PHP scripting language. Thus, MySQL is used for storing log data in the server because of its stability and reliability.

Anaconda- Anaconda is a Python and R distribution platform. This application is used for machine learning and data science tasks in the project. It's a free, open-source platform that makes package management and installation convenient. Deploying the correct packages, uploading files, setting environment variables, and executing commands are all automated by Anaconda.

Streamlit- Streamlit is an open-source app platform for machine learning using Python. With few lines of scripting, it helps to create amazing applications. In this project, Streamlit is used for embracing python scripting by creating a web app that includes data explanation, visualization and classification, and regression.

Libraries- The libraries used in this project were selected diligently and constructively, resulting in significant time savings. The following are short descriptions of some of the major libraries used for the development of the project.

Table 13 Module Requirements

S.N	Modules	Description
1.	Socket	Socket offers various functions, constants, objects, and related exceptions for building full-fledged network applications including client and server programs (Pythonic, 2020). This library is used for the socket to

		connect to a port number so that the application to which data is destined to be sent can be defined by the TCP layer (Oracle, 2020) (Dwivedi, 2020).
2.	Scikit Learning	Scikit-learn offers a variety of supervised and unsupervised learning algorithms (Brownlee, 2014). Sklearn provides wide-ranging documentation and contains both of the machine learning algorithms necessary for this project.
3.	NumPy	NumPy helps to efficiently and simply execute mathematical and logical operations. This library is therefore chosen to perform calculations for machine learning efficiency.
4.	Pandas	Pandas are used for data exploration, cleaning, transformation, and visualization.
5.	OS	Python OS module allows the functionality based on the operating system to be used and to communicate with the underlying operating system in various ways. (JournalDev IT Services Private Limited, 2020). This module is used to interact with files and to modify variables in the environment, etc.
6.	Requests	The requests module enables Python to submit HTTP requests. The response data from an HTTP request is returned as a Response Object (content, status, etc) (W3Schools, 2021). The Requests library is an

		important part of Python that allows making HTTP requests to a specific URL.
7.	Logging	Python has a logging module that allows users to write status messages to a file or some other output stream (Stack Abuse, 2021). The file can provide details on which sections of the code are implemented and what issues have arisen (GeeksforGeeks, 2018).
8.	Mysql.connector	MySQL Connector/Python is a Python library that allows Python programs to link to MySQL databases through an API. It's written entirely in Python, except for the Python Standard Library as a dependency (GeeksforGeeks, 2020).
9.	Matplotlib	Matplotlib is a Python library that allows you to create static, interactive, and animated visualizations. Matplotlib creates high-quality figures in a range of hard - copy and graphical formats across several platforms. Matplotlib is a Python library that can be used in scripts, GUI toolkits, etc (Python Software Foundation, 2021).

8.8.2.2.2 HARDWARE DEPENDENCIES

The hardware requirements which are required for the development of the project are given below:

- A computer system
- Central Processing Unit: Intel Core i5 ~ 1.80GHz
- Primary Memory (RAM): 8GB
- Secondary Memory: 1TB

8.8.2.3 USER INTERFACE REQUIREMENTS

The following are considered for the user interface requirement:

- Enable the user to easily try out various options

- A variety of functions depend on mouse clicks and command terminals, to provide an improved user experience.
- Using easy-to-understand vocabulary targeted at the user's type.

8.8.2.4 FUNCTIONAL REQUIREMENTS

The following are the system's practical requirements:

- System initialization

  The system requires administrator privilege and root privilege in both the Windows and Linux environments to initialize the system.
- Start honeypot server
- The user starts to open the honeypot server by executing the python script with- python3 final.py in the terminal interface.
- Start anomaly detection web interface

  The user starts the web interface for machine learning by executing the python script with- streamlit run app.py in the terminal interface.
- Select suitable dataset

  The user needs to select a suitable dataset for visualization and accuracy prediction in the browsing interface of the UI.
- Stop the programs

  The user has to manually execute the system with the stop function in the terminal.

GO BACK TO TOP

7.9 APPENDIX I: PROPOSED SYSTEM DEPLOYMENT

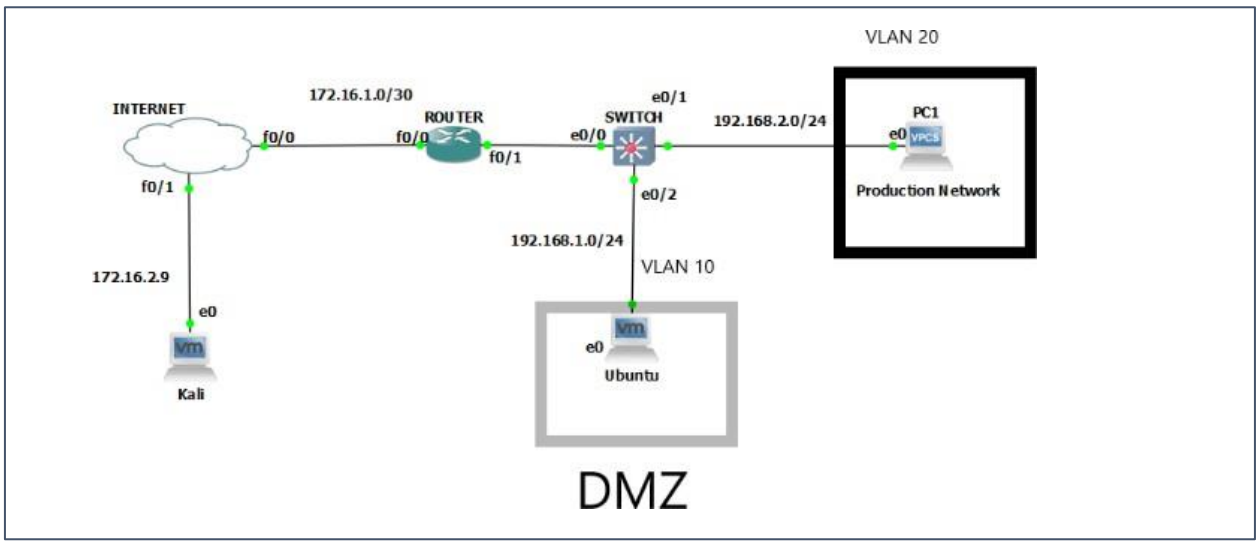


Figure 112 Network Architecture

VMware was used in the implementation, and two virtual operating systems were installed on it. When compared to solutions based on physical devices, virtual honeypots are a more cost-effective and scalable option for conducting attack experiments (Alata, 2006). The first step in luring an intruder is to properly set the trap. Honeypots should be deployed along with production servers or in a different DMZ in the network. Figure 112 depicts the overall architecture of the system. It is a straightforward and efficacious approach to find out attacks on the network. Here, the honeypot is configured in the DMZ in the network by configuring a separate VLAN 10 with IP address 192.168.1.2. In other words, it has been isolated from the untrustworthy public Internet and the organization's trusted network. Subnetting is used to do this. It is connected to the same network, which is logically divided using VLANs. In the case of this network, the production network is configured to VLAN 20. The honeypot is specifically running in Ubuntu Operating System and for the proof of concept, an attacker is placed which runs in Kali Operating System with IP address 172.16.2.9. The critical approach for securing the DMZ is to identify threats against this zone's servers and to reject these intrusions by directing them to honeypots and collecting sufficient information at odds with the attacker. The project aims to find a solution to problems like responding to malicious activity and redirecting intruders to honeypots. The designed system detects malicious activity and sends it to a decoy system where forensics can be collected.

```
SWITCH(config)#do sh vlan brief

VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Et0/3, Et1/0, Et1/1, Et1/2
                                                Et1/3, Et2/0, Et2/1, Et2/2
                                                Et2/3, Et3/0, Et3/1, Et3/2
                                                Et3/3
10   DMZ                              active    Et0/1
20   Production_Network               active    Et0/2
1002 fddi-default                     act/unsup
1003 token-ring-default               act/unsup
1004 fddinet-default                  act/unsup
1005 trnet-default                    act/unsup
```

Figure 113 VLAN Segregation

```
root@ubuntu: ~
root@ubuntu:~# ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.2  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::8c4e:f8fa:f353:9a3a  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:db:61:a5  txqueuelen 1000  (Ethernet)
        RX packets 1651  bytes 149033 (149.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 337  bytes 48376 (48.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 40063  bytes 2414142 (2.4 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 40063  bytes 2414142 (2.4 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 114 IP address check of honeypot

```
root@kali:~# ping 192.168.1.2
PING 192.168.1.2 (192.168.1.2) 56(84) bytes of data.
64 bytes from 192.168.1.2: icmp_seq=1 ttl=62 time=39.8 ms
64 bytes from 192.168.1.2: icmp_seq=2 ttl=62 time=33.8 ms
64 bytes from 192.168.1.2: icmp_seq=3 ttl=62 time=39.2 ms
64 bytes from 192.168.1.2: icmp_seq=4 ttl=62 time=32.2 ms
64 bytes from 192.168.1.2: icmp_seq=5 ttl=62 time=32.6 ms
64 bytes from 192.168.1.2: icmp_seq=6 ttl=62 time=38.7 ms
64 bytes from 192.168.1.2: icmp_seq=7 ttl=62 time=42.3 ms
64 bytes from 192.168.1.2: icmp_seq=8 ttl=62 time=34.1 ms
64 bytes from 192.168.1.2: icmp_seq=9 ttl=62 time=38.3 ms
```

Figure 115 Ping successful from attacker to honeypot

GO BACK TO TOP

7.10 APPENDIX J: CLASSIFICATION OF HONEYPOT

Honeypots can be categorized according to the usability:

Production honeypot- Honeypots are used directly to enhance the protection of a network system within the company. They help to detect attackers and to distract them from real resources (Kovtun, 2018).

Research honeypot- Honeypots that help to analyse an attacker's actions and to discover new vulnerability-exploiting techniques and new forms of attacks (Kovtun, 2018).

Based on the degree of involvement or interaction, honeypots may also be categorized as:

Low interaction honeypot- Honeypot acts as an emulator of the operating system and utilities, and can only certain fake services. These honeypots are easy to build, but readily identifiable as well. A simple command may only be used by the attacker to identify that a low honeypot involvement does not help it (Sharma, 2013).

High interaction honeypot- Honeypot offers some real uncertainties to the real-like operating systems and some real services. This allows the attacker's data to be collected and their actions registered. These are the actual single-device computers with one network interface on the network (Sharma, 2013).

Medium interaction honeypot- Honeypots that imitate those services in a more advanced way than honeypots with low interaction, but still do not have complete interaction in the system. Compared to low-interaction honeypots that enforce network protocols, medium-interaction honeypots simulate application responses for incoming services (Kovtun, 2018).

GO BACK TO TOP

7.11 APPENDIX K: MACHINE LEARNING MODELS

Supervised Learning: Those algorithms that require external assistance are known as supervised learning. The training and testing datasets are separated from the input dataset. There is an outcome variable in the training dataset that needs to be predicted. For the classification, algorithms learn patterns from the training dataset and implement them in the test dataset (Dey, 2016). The flowchart of supervised machine learning has been given below.

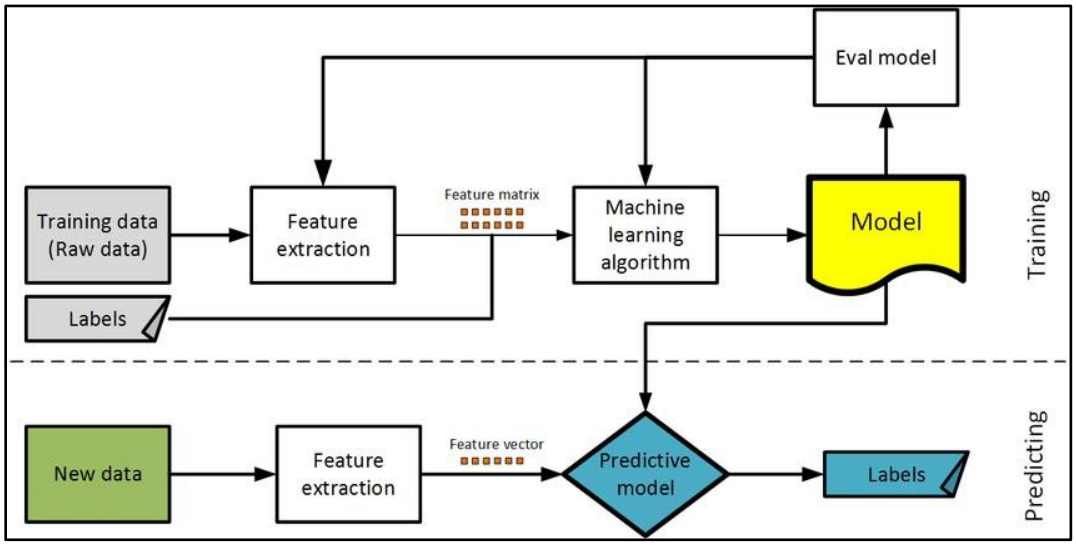


Figure 116 Flowchart of Supervised Machine Learning (Dong Nguyen, 2016)

Unsupervised Learning: In unsupervised learning, the data of the algorithm is only used to learn a few features. When different data is added, it recognizes the data's class using prior experienced features. It is used for feature reduction and clustering (Dey, 2016). The flowchart of unsupervised machine learning has been given below.

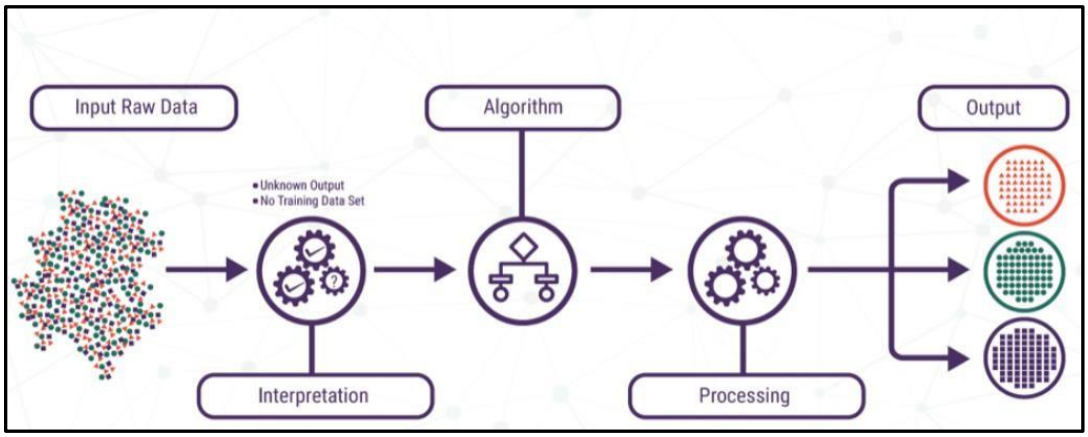


Figure 117 Flowchart of Unsupervised Machine Learning (Technative, 2020)

Semi-supervised Learning: Semi-supervised learning integrates the strengths of both supervised and unsupervised learning algorithms. It can be effective in places like data mining and machine learning where there is already unlabelled data and having the labeled data is a time-consuming task (Dey, 2016). The unsupervised machine learning process flow is shown below.

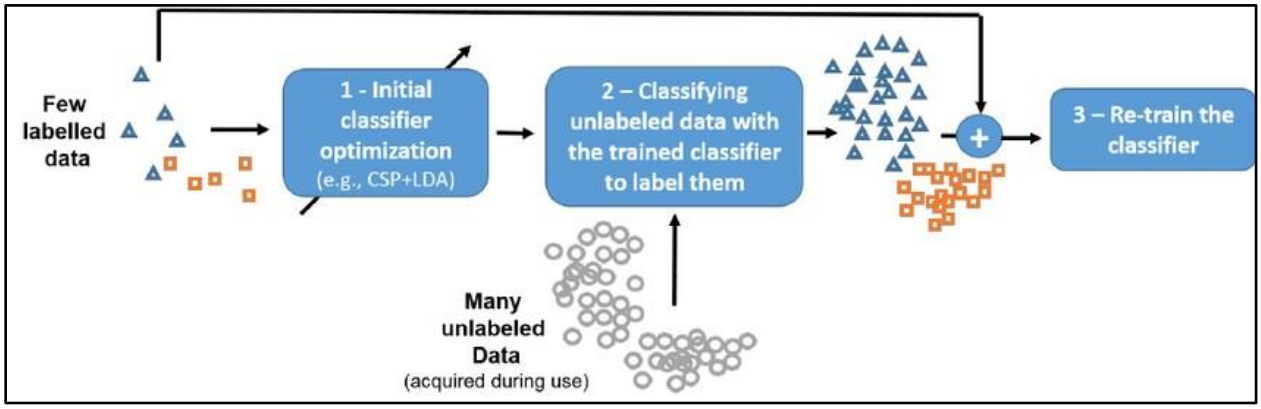


Figure 118 Principle of Semi-Supervised Learning (LOTTE, 2015)

Reinforcement Learning: Reinforcement learning is a technique in which expert systems, known as agents, operate in an unforeseen environment to adapt and learn based on the inputs they receive. Positive feedback, also known as incentives, or negative feedback, is also known as punishments (Nandy & Biswas, 2018). Since it is difficult to determine which action to take while it is in one state, reinforcement learning operates on a trial-and-error framework. Astute program agents that offer incentives and punishments when working with the environment benefit from reinforcement learning (Nandy & Biswas, 2018).

GO BACK TO TOP

7.12 APPENDIX L: ALGORITHMS

7.12.1 DECISION TREE

The methods for constructing the tree as shown in Figure are top-down recursive and divide-and-conquer. The tree starts with the root node, which represents the entire training dataset (Alsagheer, et al., 2017).

- If both training lists produce the same result, the node is considered a leaf and is labeled with that class.
- Or else, the tree divides the set by the greatest knowledge attribute and labels each node with the attribute's name.
- Repeat the steps until all attributes have the same class or there are no more new attributes to part, then stop.
- Tree ends.

GO BACK TO TOP

7.12.2 LOGISTIC REGRESSION

The correlation between binary outcomes and independent variables is determined using logistic regression, which uses a likelihood as the expected value of the dependent variable. (Rushin, et al., 2017). For the evaluation of data, the logistic regression model is the most extensively used regression model (HOSMER, et al., 2013).

When the dependent variable is converted into a logit variable, logistic regression uses maximum likelihood estimation. The natural log of the odds of the dependent equalling a certain value or not is called a logit. The probability of a specific event (value) occurring is estimated using logistic regression. Logistic regression measures changes in the dependent's log-odds rather than the dependent's own (Garson, 2014). Based on individual characteristics, logistic regression can model the likelihood of a specific outcome. Since chance is a ratio, the logarithm of the chance given will be used to model it (Sperandei, 2014).

The statistical model for logistic regression is

$\log(\boldsymbol{\pi}/(1-\boldsymbol{\pi}))) = \beta o + \beta 1x1 + \beta 2x2 + \cdots . \beta mxm$ (Sperandei, 2014)

Where x is the explanatory variable and **π** is a binomial proportion. The logistic regression model's parameters are β0 and β1 (Horton, 2019).

GO BACK TO TOP

7.13 APPENDIX M: CONSIDERED METHODOLOGIES

7.13.1 SCRUM METHODOLOGY

Scrum is a complicated process with several variables that influence the final result. The primary goal is to achieve customer's expectations through mutual accountability, transparent communication, and constant delivery of valuable software (Digité, Inc, 2020). Rather than "traditional tactics" the emphasis is on "strategy, a versatile holistic product creation where the team worked as a unit to achieve shared goals." The work is performed in this methodology with the help of a scrum master, a scrum team, and a product owner in a continuous loop of the developing software. The key role of the scrum master in a scrum phase is to reduce difficulties (Neelima & Naga , 2013). The Scrum Team consists of developers, testers, and other professionals from different fields who work together to produce an end product that meets the customer's needs (Neelima & Naga , 2013).

GO BACK TO TOP

7.13.2 ITERATIVE MODEL

Iterative model is a development method that incorporates iterative methodology with an incremental construct model (tutorialspoint, 2020). This approach could be defined as "developmental acquisition" or "incremental construct," according to the authors (tutorialspoint, 2020). The entire requirement is divided into different builds in the incremental model. The development of the system consists of iterative implementations, design, coding, and testing stages during each iteration (tutorialspoint, 2020). Each subsequent release of the module adds to the previous release's functionality (tutorialspoint, 2020). The procedure is repeated until the whole system meets the specifications (tutorialspoint, 2020).

GO BACK TO TOP

7.13.3 KANBAN MODEL

Kanban Board: A Kanban board is a platform for visualizing workflows and one of the main components of the Kanban system. Using a Kanban board to visualize workflow and activities will help to better understand your processes and gain a better understanding of the workload (Kanbanize, 2021). The sections in software development are generally:

- To Do

- Plan
- Develop
- Test
- Deploy
- Done

Three columns are used in the simplified version:

- To Do
- In process
- Done

Kanban Notes:

On a Kanban board, a Kanban card is a visualization of a work object. On that board, a Kanban card is a sticky note. One work item is represented by each Kanban card or sticky note (PlanView, 2021). The Kanban cards usually consists of (Guru99, 2021) :

- Priority
- Owner
- Type
- Due date

Kanban Practices

When it comes to Kanban, there are three fundamental rules to follow:

- Visualize Workflow: A visual representation of the development process allows to analyse the tasks progress from "not completed" to "done right." (Klipp, 2014).
- Limit Work in Process (WIP): The maximum number of work items in each stage (Kanban board column) of the workflow is limited by work in progress (WIP) limits. WIP limits help the team concentrate only on current activities, allowing you to complete single work items faster (Kanbanize, 2021).

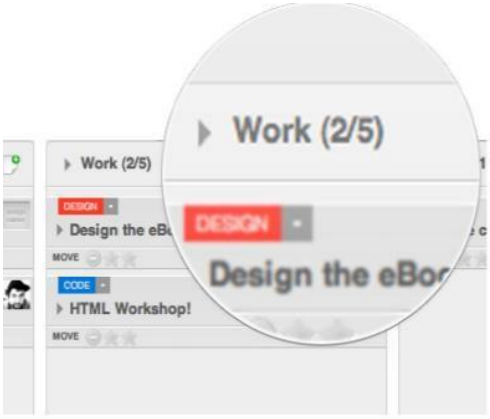


Figure 119 Limit WIP (Klipp, 2014)

- Measure and Improve Flow: When the first two Kanban rules are in place, work can flow easily and feel more manageable. Any disruption in the flow should be given special consideration. There are potential areas for further visualization and process enhancement. Kanban offers the data needed to fine-tune the process to boost flow and performance (PlanetTogether, 2021).

GO BACK TO TOP

7.13.4 VALIDATION OF THE METHODOLOGY

In the context of the Final Year Project, Kanban Methodology is an excellent efficiency strategy for working on a single project that involves several steps. A complete picture will help to ensure focus on the right aspects at the right time. With individual Kanban, the workflow steps "To-Do," "Doing," and "Done" are widely used. Furthermore, the following reasons help to justify that Kanban is the most effective workflow management methods:

- Flexibility: Since new features and modules can be introduced at any time during the development of the project, therefore changes can arise at any point. Kanban is built on the ‘just-in-time' concept, making it adaptable to changing trends and features.
- Visualization: Visualization is an effective Kanban practice. It will assist in completing all of the project's backlog activities as well as the sequence of process states that a task must pass through before being delivered. The author will instantly see how tasks are progressing through the process by using the Kanban board. Because of the clarity of its visual appearance, bottlenecks can be easily identified as they form.
- Feedback: Feedback loops force the development of the project to reflect and change, which is why they're so significant. It can be used to improve efficiency in this project's outcomes. Identifying areas for change and converting them into action items during each sprint will assist the author in monitoring and solving key issues related to technology or product improvement (Bonam, 2015).
- Improved efficiency: Visualizing the process will easily reveal areas that are lacking inefficiency. On the Kanban board, backlogs, delayed tasks, and enough work in progress are all clear. Any barrier after it’s removed, makes the development run more smoothly and efficiently and finish within the deadline of the final year project.
- Increased productivity: Improved performance in Kanban contributes to improved productivity. Kanban increases productivity by refocusing attention from beginning to finishing tasks.

GO BACK TO TOP

7.14 APPENDIX N: PHASES OF METHODOLOGY

7.14.1 BACKLOG

The overall development phase was divided into several tasks and held in the "To-Do" section of the Kanban board, as seen in the diagram above. To simplify the task, they have been further divided into subtasks. The “In Progress” section shows the maximum WIP that can be added there. Project activities can be monitored using deadlines, and appropriate decisions can be made.

GO BACK TO TOP

7.14.2 IMPACT ANALYSIS

The research stage was the first and most essential step of the impact analysis, which would decide the project's project performance. Since this was an individual project so, the author herself represented the team and was committed to meeting deadlines, which were strategic and successful. The research stage aimed to demonstrate that the team members have done some background research on the subject and that they are aware of the prior research done, as well as its shortcomings. Primary research was conducted through literature study, and secondary research was conducted through community surveys and interviews. The research process delivered a path for the feasibility analysis of the project. Extensive research was conducted of the benchmark works performed earlier in the field of honeypot and machine learning to gain familiarity and insights into the need of the project around the world. Existing commercial honeypots such as honeyd, specter were reviewed to grasp strong knowledge of the working mechanism of honeypots. It decides whether a planned project or method is feasible. It is conducted out for a variety of reasons, including determining whether a software product is appropriate in terms of creation, implementation, and project commitment to the team. Additionally, risk analysis assisted in finding areas where development flaws could cause severe production issues. The team can remediate and minimize the overall risk of a development defect by finding areas of a worry early. To develop an initial version of the solution - a prototype - the team gathered preliminary specifications as part of the requirements gathering process. Finally, the main objective of impact analysis was carried out where the codes and records were identified if they will need any changes if the team implements request changes.

GO BACK TO TOP

7.14.3 BUILD

The build phase consisted of seven tasks in total with corresponding deadlines and sub-tasks. This was the main phase of this methodology where all the development of the project has been carried out. Firstly, a simple honeypot server was developed which could receive a connection from the external party. The honeypot was built as a Python port listener on the IPv4 address of common ports using a socket. It establishes an HTTP session and a socket on which to listen for connections. Further moving on, the IP blacklisting feature was added in the module. After the connection had been established, a log file was generated and saved as text and CSV files. With the development of the alert module, it checked and monitored the log file and when it noticed the change in the size of the file; the system administrator would be alerted. Forging ahead, a dataset was selected as a case study for machine learning where the data was trained and tested for high precision and accurate results. Decision Tree and Logistic Regression algorithms were implemented on it from scratch by referring to a bunch of papers and journals to understand its core concept. The ideas from the paper were expressed in the form of code by using Python for prototyping. Finally, a user-friendly application was built which is incredibly easy to use and learn machine learning effectively with the help of Streamlit.

After the accomplishment of these tasks mentioned in the proposal, the database and honeypot interface were some added features to the system. Due to the flexibility allowed by the methodology, it allowed to gradually incorporate it without apprehension of over-commitment, since there was no necessity to make drastic shifts right away.

GO BACK TO TOP

7.14.4 USER ACCEPTANCE TESTING

This initiative was evaluated using several testing methodologies but for this system, Unit Testing and System Testing were chosen as the most appropriate forms of testing. Aside from the fact that Unit Testing occurred simultaneously with systems integration, System Testing was performed only after each operational function had been completely developed. Here, the honeypot system and machine learning interface have been broken down into two components so, that testing can be done evidently and efficiently. In the case of both of the systems, the development's testing is usually conducted on an outcome basis, and depending on the situation, one of the testings is performed. To ensure a high-quality development, user acceptance testing was conducted about

the above testing by providing a beta version of the application. Finally, the set of test cases performed is found to be relatively low since it is a network-based system. To compensate for this flaw, the unit was compared to a variety of other instruments. During this method, beta user feedback (Post-survey) was also gathered to extract some of the system's potential benefits and drawbacks, as well as possible add-ons.

GO BACK TO TOP

7.15 APPENDIX O: IMPLEMENTATION

All of the scripts of the project are written in Python 3. Few snippets of core features in the scripts are provided below. Please remember that all code snippets below presume that the modules have already been imported.

7.15.1 SIMPLE HONEYPOT SERVER

```
if port_number >= 1 and port_number <= 65535:  #Checking if the port number lies in between the range
    try:
        connect_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM) #AF_INET address ipv4 #SOCK_STREAM stands for TCP connection
        connect_socket.bind(("0.0.0.0", 80)) #type the ip address you want to bind to the port
    except socket.error as e:
        sys.exit("[!] Unable to bind to port with error: {0} -- {1} ".format(e[0], e[1]))
else:
    print("[!] Please specify a valid port range (1-65535) in the configuration.")
    sys.exit(2)
```

Figure 120 Creation of INET socket stream

In the above snippet, the script checks if the port number configured by the user belongs to the port range 1 to 65535. Thus, when the criteria match, a TCP/IP connection is established with the intruder. The address 0.0.0.0 applies to all IPv4 addresses on the local machine in the form of servers. The bind associates the connect_socket with the address and port number 80 which is the HTTP port to connect to the web server.

```
check_platform = platform.system() #checking the platform i.e. Windows and Linux

if check_platform == "Windows":  #checking admin privilege in Windows platform
    if not ctypes.windll.shell32.IsUserAnAdmin():
        sys.exit("\n[!] Admin privileges are required to modify firewall rules.\n")
elif check_platform == "Linux" :  #checking root privilege in Linux platform
    if not os.geteuid()==0:
        sys.exit("\n[!] Root privileges are required to modify firewall rules.\n")
else:                             #Throws error message if suitable platform is not selected
    sys.exit("\n[!] \"{0}\" is not a supported platform.\n".format(pcheck_platform))
```

Figure 121 Privilege Check

As we can see from the above code that user privileges are required for the execution of the script. In the python script, it looks for a cross-platform way to verify admin privileges. This code first checks to see if it's running as administrator on a Windows system; if that fails, the script assumes you're using Linux. Depending on whether you're an admin or not, the feature will return true or false.

```
logger = logging.getLogger('hp')
formatter = logging.Formatter("%(message)s - %(asctime)s","%c") #formatting the log file output
logger.setLevel(logging.INFO) #setting the level to Info
logger.propagate = True
streamhdlr = logging.StreamHandler()
logger.addHandler(streamhdlr)
streamhdlr.setFormatter(formatter)
if log != "": #a log file is created if the earlier log name doesnpt exist
    try:
        filehdlr = logging.FileHandler(log)
        filehdlr.setFormatter(formatter)
        logger.addHandler(filehdlr)
    except IOError as e:
        sys.exit("[!] Unable to create/append file: {0} -- {1} ".format(e[0], e[1]))
```

Figure 122 Logging code

The snippet above effectively enriches a log message by providing context information. It is used to determine when the log is sent and where it is sent and the current time of formation of a message. A log message is appended to its parent log each time it is created. It also generates a log file if one does not exist.

```
while True:

    c, addr = connect_socket.accept() #Establishing connection between the client.
    clientaddress = str(addr[0]) #Receiving the intruder ip address
    if clientaddress in (whiteip,"127.0.0.1" ): #Whitelsiting the given ip address
        logger.info("[!] Hit from whitelisted IP: {0}".format(clientaddress)) #informing the admin
        c.shutdown(socket.SHUT_RDWR)
        c.close() #Closing the connection
```

Figure 123 WhiteIP Configure

As shown in the code snippet above, the incoming IP address is primarily assigned as a whitelisted address. The local host is used as the whitelisted address in this case.

```
 check_platform == "Linux":
A: chain where the rule is appended
s: source of the packet
j: jump to tatget

  try:
      result = check_output(["/sbin/iptables", "-A", "INPUT", "-s", "{0}".format(clientaddress), "-j", "DROP"])
      logger.info("[!] The IP ADDRESS IS BLACKISTED IN LINUX: {0} with IPTABLES (TTL: {1})".format(clientaddress, "Permanent"))
  except (OSError,CalledProcessError) as e:
      logger.error("[!] Failed to blacklist {0} with IPTABLES ({1}), is iptables on the PATH?".format(clientaddress, e))
```

Figure 124 Linux Platform Check

As we can see from above, in the Linux platform IP Table commands are executed to block an IP on the Linux server. This is, once again, based simply on the administrative rights given during the python script's execution. If it fails to block the IP, it throws an error message with the IP

address we are trying to block. The blocking rules are added in the IP Tables INPUT chain to block an IP address from the server. The following are the main IP Tables switches used in completing these tasks:

-A: prepend rule

-s: set the IP address

-J: Jump to target

```
elif check_platform == "Windows": #netsh advfirewall is used to control the Windows firewall behaviour
#dir: inbound/potbound traffic
#action: block/allow/bypass
#protocol: any/tcp/udp/icmpv4/icmpv6
#remoteip: destination IP addresses of an outbound packet
    try:
        result = check_output(["netsh", "advfirewall", "firewall", "add", "rule", "name=Honeypot Blacklist", "dir=in",
        "remoteip={}".format(clientaddress), "protocol=any" , "action=block"], shell=True)
        logger.info("[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: {0} with Windows Firewall (TTL: {1})".format(clientaddress,
        "Permanent"))
        ss= alert.send_email('honeypotproject2021@gmail.com','INTRUDER ALERT!!!','Somebody is trying to connect to the network!!!',
        'fyphoneypot.log')
    except (OSError,CalledProcessError) as e:
        logger.error("[!] Failed to blacklist {0} with Windows Firewall ({1})".format(clientaddress, e.output))
else:
    logger.error("[!] {0} is not a supported platform".format(check_platform))
```

Figure 125 Windows Platform Check

In the above code snippet, the firewall rule is applied to deny the input of packets from a specific IP address in the Windows platform. This is dependent on the administrative rights granted during the execution of the python script again. If it fails to block the IP, it throws an error message with the IP address we are trying to block.

```
#initializing port scanner
scanner = nmap.PortScanner()
scanner.scan(hosts=ip_address)
ip_status = scanner[ip_address].state()
sc = {}
#running for loop to print info about all the known ports
for host in scanner.all_hosts():
    detail_info = []
    for proto in scanner.all_protocols():
        lport = scanner[host][proto].keys()
        sc = scanner[host][proto]
```

Figure 126 NMAP Automation

As shown above, in the snippet, NMAP automation has been introduced to quickly access Nmap port result data to be handy to system administrators who want to automate scanning tasks and reports.

7.15.2 EMAIL ALERT

```
    try:
        server = smtplib.SMTP('smtp.gmail.com', 587)
        server.ehlo()
        server.starttls()
        server.login('honeypotalert2021@gmail.com', '************')
        text = msg.as_string()
        server.sendmail(email_sender, email_recipient, text)
        print('email sent')
        server.quit()
    except:
        print("SMPT server connection error")
    return True

send_email('honeypotproject2021@gmail.com',
           'INTRUDER ALERT',
           'SOME EXTERNAL PARTY IS TRYING TO CONNECT TO THE HONEYPOT!!!!!!!',
           'fyphoneypot.log')
```

Figure 127 Email alert

In the above python script, the system sends to the address ‘honeypotalert2021@gmail.com' in an email. The message confirmed that there had been an intrusion into the honeypot. At the same time, the system connects the log data that was recorded to the admin in the same email.

7.15.3 DATABASE

databasemod.py

```
1   #!/usr/bin/python3
2   import csv
3   import mysql.connector
4   import pandas as pd
5   import databasemod
```

Figure 128 Importing modules in databasemod.py

In the above code snippet, the modules are imported which are required for the database module. The modules details are provided here: SOFTWARE DEPENDENCIES

```
with open("fyphoneypot.log", 'r', encoding="ISO-8859-1") as f, open('fyphoneypot.csv', 'a') as f2: # or 'wb' if on python2
    writer = csv.writer(f2)
    writer.writerow(['LOG_NO', 'LOG_DATA']) # replace with your custom column header

    i = 0
    for line in f:
        writer.writerow([i] + line.rstrip().split('|'))
        i += 1
        if i == 1000000000000:
            break
    writer = csv.writer(f2)
    writer.writerow(['LOG_NO', 'LOG_DATA'])  # replace with your custom column header
```

Figure 129 Converting into CSV

In the above python script, the log "fyphoneypot.log" which was collected by the honeypot is converted into CSV format and appended in the fyphoneypot.csv file. This will help to export the data to the database easily. Each data is appended as rows in each line with the column names as LOG_NO and LOG_DATA. The code takes till 1000000000000 logs and stops the conversion of logs into CSV format.

```
with open('fyphoneypot.csv', 'r') as csvfile:
    csv_reader = csv.reader(csvfile)
    for each in csv_reader:
        print(each)

data = pd.read_csv(r'fyphoneypot.csv')
df = pd.DataFrame(data, columns=['LOG_NO', 'LOG_DATA'])
```

Figure 130 Reading file with Pandas

In the above code, the read CSV function of the Pandas module is used to load data from a fyphoneypot CSV file into a Pandas DataFrame. LOG_NO and LOG_DATA are the labels of the data frame.

```
db = mysql.connector.connect(
    host='localhost',
    user='root',
    password="",
    database="honeypot"
)

cur = db.cursor()

cur.execute("CREATE TABLE IF NOT EXISTS fyphoneypotlog("
            "log_no int, "
            "log_data Varchar(600))")


for i, row in data.iterrows():
    mysql = "INSERT INTO honeypot.fyphoneypotlog VALUES (%s,%s)"
    cur.execute(mysql, tuple(row))
    db.commit()
```

Figure 131 Creating database table

In the above code snippet, the code connects to the MSQL database and establishes a connection with it. The MySQL.connector.connect is used to connect to the MySQL server. A table “fyphoneypot” is created from the database “honeypot” where the data values are inserted into the table. The host, user, and password arguments are kept default.

7.15.4 HONEYPOT UI

```
  <thead class="thead-dark">

    <tr>
      <th scope="col">Log No</th>
      <th scope="col">Data</th>
    </tr>
  </thead>
  <tbody>
      <?php
      $servername = "localhost";
$username = "root";
$password = "";
$database = "honeypot";

$conn = mysqli_connect($servername, $username, $password, $database);

$sql = mysqli_query($conn,"SELECT * FROM fyphoneypotlog");
    while ($record = mysqli_fetch_assoc($sql)) {


    ?>
    <tr>
      <td><?php echo $record['log_no'] ?></td>
      <td><?php echo $record['log_data'] ?></td>
    </tr>
  <?php } ?>
  </tbody>
```

Figure 132 Fetching data from database

In the above PHP script, server name, username, password, and database are variables that contain credentials related to the database. “mysqli_connect” is used for connecting with the database using the credentials, and “mysqli_query” is used to create a query which is executed in the database. The while loop is used to continuously fetch the data from in the database and the echo commands are used to print the “log_no” and “log_data” fields.

```
?>
<?php

if ($id == "white") {
  ?>
  <section>
  <div class="container">
      <table class="table">
  <thead class="thead-dark">

    <tr>
      <th scope="col">Log No</th>
      <th scope="col">Data</th>
    </tr>
  </thead>
  <tbody>
      <?php
      $servername = "localhost";
$username = "root";
$password = "";
$database = "honeypot";

$conn = mysqli_connect($servername, $username, $password, $database);

$sql = mysqli_query($conn,"SELECT * FROM fyphoneypotlog WHERE log_data LIKE '[!] Hit from w%' ");
    while ($record = mysqli_fetch_assoc($sql)) {


    ?>
    <tr>
      <td><?php echo $record['log_no'] ?></td>
      <td><?php echo $record['log_data'] ?></td>
    </tr>
  <?php } ?>
  </tbody>
</table>
  </div>

 </section>
 <?php

}
```

Figure 133 Pushing data starting from [!] Hit from w%

In the above code snippet, the query has been written such that, the query returns all the data which starts from “[!] Hit from w” in the log_data field of the database.

```
?>
<?php

if ($id == "black") {
  ?>
  <section>
  <div class="container">
       <table class="table">
  <thead class="thead-dark">

    <tr>
      <th scope="col">Log No</th>
      <th scope="col">Data</th>
    </tr>
  </thead>
  <tbody>
      <?php
      $servername = "localhost";
$username = "root";
$password = "";
$database = "honeypot";

$conn = mysqli_connect($servername, $username, $password, $database);

$sql = mysqli_query($conn,"SELECT * FROM fyphoneypotlog WHERE log_data LIKE '[+] B%' ");
    while ($record = mysqli_fetch_assoc($sql)) {


    ?>
    <tr>
      <td><?php echo $record['log_no'] ?></td>
      <td><?php echo $record['log_data'] ?></td>
    </tr>
  <?php } ?>
  </tbody>
</table>
  </div>

 </section>
 <?php

}
```

Figure 134 Pushing data from [+] B%

In the above code snippet, the query has been written such that, the query returns all the data which starts from “[!] B” in the log_data field of the database.

7.15.5 ANOMALY DETECTION UI

```
import streamlit as st
#for working with arrays importing numpy library as np
import numpy as np
#for data analytics and data science importing pandas library as pd
import pandas as pd
#for satistical graphics and virtualization patterns importing seaborn as sns
import seaborn as sns
#for basic plotting consisting of bars, lines, piecharts
import matplotlib
#a set of functions that render matplotlib possible
import matplotlib.pyplot as plt
#from sklearn.model slection importing traintestsplit for splitting dataset in supervised ML
from sklearn.model_selection import train_test_split
#from sklearn.tree importing decision tree
from sklearn.tree import DecisionTreeClassifier
#from sklearn.linear_model importing logistic regression algorithm
from sklearn.linear_model import LogisticRegression
#from sklearn.metrics importing accuracy classification score
from sklearn.metrics import accuracy_score
#from sklearn importing model_selection for training a series of models with varying hyperparameter values by using algorithms
from sklearn import model_selection
#first thing to do before importing other matplotlib packages. Agg backend is for writing file which is the default backend of matplotlib
```

Figure 135 Importing and assigning module variables in app.py

The above snippet belongs to the app.py script where all the modules and libraries required for machine learning are loaded. More specific details of these libraries can be viewed from here: SOFTWARE DEPENDENCIES

```
#option 1
if choice=='Data Interpretation':
  #displaying a subheader
  st.subheader("Data Interpretation")
  #inserting a dataset file uploader that accepts any csv,xlsx, text or json file one at a time
  input_data= st.file_uploader("Upload dataset:",type=['csv','txt','json''xlsx',])
  #loading a green success message after the file has been uploaded
  st.success("Data has been successfully loaded in the system!")
  if input_data is not None:
    #to read a file
    df=pd.read_csv(input_data)
    #displays a dataframe as a table
    st.dataframe(df.head(80))
    #displaying checkboxes
    if st.checkbox("Show shape"):
      #describing a data set
      st.write(df.shape)
      #displaying the all the names of total columns of the dataset
    if st.checkbox("Show columns"):
      st.write(df.columns)
    if st.checkbox("Select multiple columns"):
      #displaying selective columns of the dataset you want to display
      selected_columns=st.multiselect("Select preferred columns:",df.columns)
      df1=df[selected_columns]
      st.dataframe(df1)
      ######################
    if st.checkbox("Show summary"):
      st.write(df.describe().T)
```

Figure 136 Data Interpretation Code

Basic data exploration steps are performed in the above code snippet to decide what the data set consists of, which includes the head, shape, single columns, multiple columns, null values, data types, and summary of the selected dataset. These features assist in the review of data and the translation of metrics, statistics, and figures into change initiatives.

```
elif choice == 'Analysis and Visualization':
  #displaying a subheader
  stm.subheader("Data Analysis and Visualization")
   #inserting a dataset file uploader that accepts any csv,xlsx, text or json file one at a time
  input_data= stm.file_uploader("Upload dataset:",type=['csv','xlsx','txt','json'])
  stm.success("Data successfully loaded")
  if input_data is not None:
     #to read a file
    df=pd.read_csv(input_data)
    #displays a dataframe as a table
    stm.dataframe(df.head(50))
    #displaying the all the names of total columns of the dataset
    if stm.checkbox("Select multiple columns"):
      selected_columns=stm.multiselect("Select preferred columns:",df.columns)
      df1=df[selected_columns]
      stm.dataframe(df1)
    #plotting heatmap as per the confusion matrix
    if stm.checkbox('Show Heatmap'):
      stm.write(sns.heatmap(df1.corr(),vmax=1,square=True,annot=True,cmap='viridis'))
      stm.pyplot()
      #plotting pairplot
    if stm.checkbox('Show Pairplot'):
      stm.write(sns.pairplot(df1,diag_kind='kde'))
      stm.pyplot()
```

Figure 137 Analysis and Visualization Code

In the above snippet, the analysis and visualization of the dataset are made easier with the graphical representation of information and data. Graphical components like heatmap, pair plots are added to deliver a convenient method to understand the data and its patterns in it.

```
        stm.dataframe(df1)
#When one row is selected, a Pandas Series is returned, and when several rows are selected, a Pandas DataFrame is returned.
        X=df1.iloc[:,0:-1]
        y=df1.iloc[:,-1:]
#setting random seed in ML Experiments
    seed=stm.sidebar.slider('Seed',1,200)
#for heading of the option of ML model algorithm
    classifier_name=stm.sidebar.selectbox('Choose one preferred model:',('Logistic Regression','Decision Tree'))

    def add_parameter(name_of_clf):

      #creating a dictionary
      params=dict()
      #Hyperparameters: adjustable parameters tuned in order to obtain optimal performance.
      #hyperparameter A hyperparameter is an external configuration to the model whose value cannot be determined from data.
      #hyperparameter of logistic regression. C is the inverse of regularization strength
      if name_of_clf=='Logistic Regression':
        C=stm.sidebar.slider('C',0.01,15.0)
        params['C']=C
      else:
        name_of_clf=='Decision Tree'
        K=stm.sidebar.slider('K',1,15)
        params['K']=K
        return params
```

Figure 138 Tuning Parameters in Models

The data is separated into X and Y variables in the above snippet. Then, before training, the model hyperparameters are set to control the entire training process. The variables that determine the network structure, such as the number of hidden units, as well as the variables that determine how it is conditioned, were included.

```
#spliting the loaded dataset into train and test set
    clf=get_classifier(classifier_name,params)
    X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=seed)
    clf.fit(X_train,y_train)
    y_pred=clf.predict(X_test)
    st.write('Predictions:', y_pred)
#calculating accuracy and precision
    accuracy=accuracy_score(y_test,y_pred)
    st.write('Name of classifier:',classifier_name)
    st.write('Accuracy',accuracy)
  elif choice == 'About Us':
    st.markdown('helo this is my project')
```

Figure 139 Splitting Data into training and testing

In the above code snippet, the dataset is split into 80% training and 20% testing. The model fitting, which is the core of machine learning, is completed. The model includes parameters that reflect the association among known and target variables, allowing for the development of new information and accurate predictions.

GO BACK TO TOP

7.16 APPENDIX P: TESTING

7.16.1 UNIT TESTING

7.16.1.1 HONEYPOT SYSTEM

7.16.1.1.1 TEST CASE 6

After the honeypot had successfully blacklisted the IP addresses that were trying to connect to it, the Windows firewall and Linux IP tables were checked to see if the rules applied by honeypot \ were implemented or not.

Table 14 Honeypot Module Test Case 6

Test Case 6	
Objective	To test whether the rules implemented in Test Case 5 are applied in Windows and Linux Firewall.
Action	Check rules in IP Tables and Windows Defender Firewall.
Expected Test Result	The connection from the intruder should be blocked/dropped in the local firewalls of the respective OS platforms.
Actual Test Result	The connection from the intruder was blocked/dropped in the local firewalls of the respective OS platforms.
Conclusion	Test Successful.

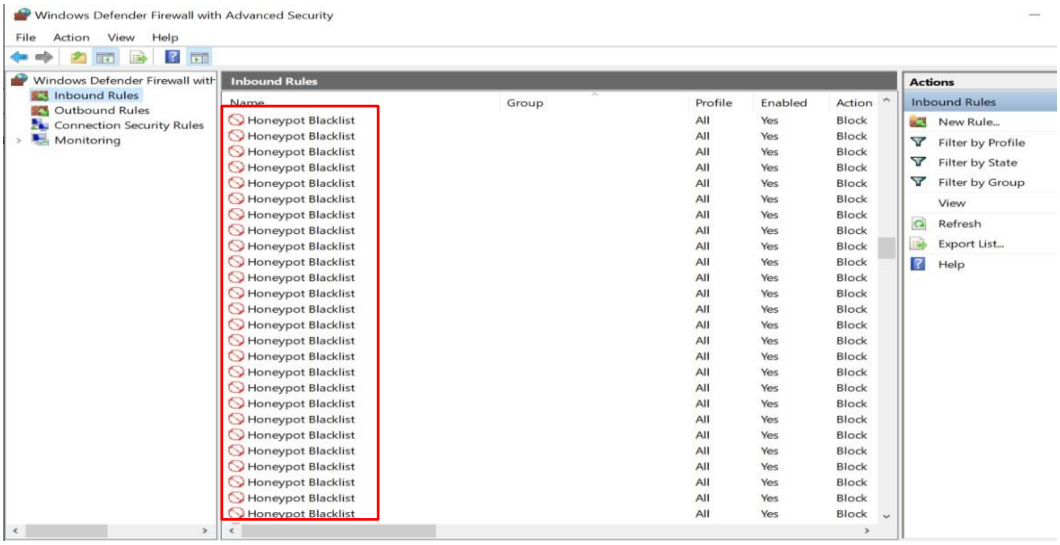


Figure 140 Test Results of Test Case 6 checking Firewall rules in Windows

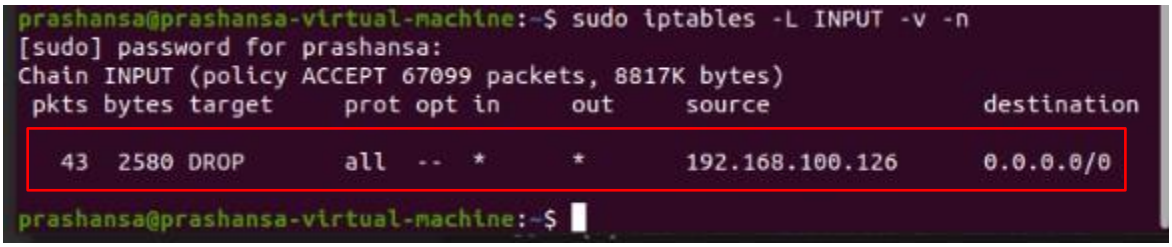


Figure 141 Test Results of Test Case 6 checking Firewall rules in IP Tables

7.16.1.1.2 TEST CASE 7

Table 15 Honeypot Module Test Case 7

Test Case 7	
Objective	To test the server's response to the attacker after the log files has been recorded.
Action	Connect external party to the honeypot through browser.
Expected Test Result	The server is expected to send back response message to the attacker.
Actual Test Result	The server send the response back with a "hacked" message.
Conclusion	Test Successful.

Then after, a response script is sent to the intruder who was trying to connect to the honeypot through a web browser. At the same time, when the connection is established to the honeypot, it recognizes this as a blacklisted IP and hence logs it as well.

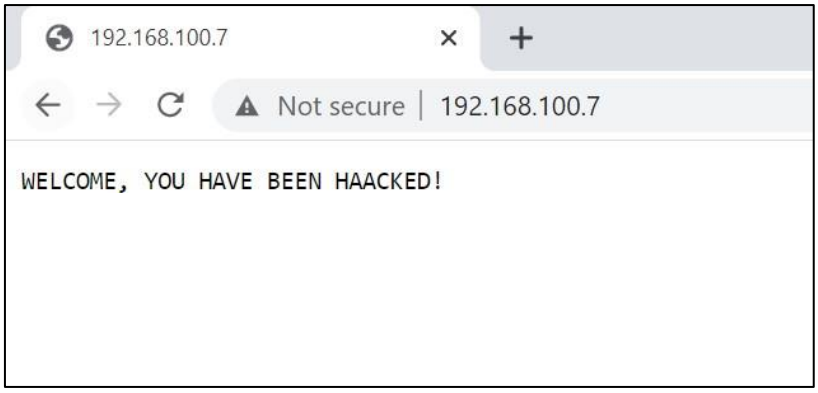


Figure 142 Test Results of Test Case 7 checking connection in browser

7.16.1.1.3 TEST CASE 8

Table 16 Honeypot Module Test Case 8

Test Case 8	
Objective	To test whether the firewall rules implemented by the honeypot can be unblocked/disabled in Windows and Linux Firewall.
Action	Check rules in IP Tables and Windows Defender Firewall.
Expected Test Result	The rules applied should be unblocked/disabled in the local firewalls of the respective OS platforms.
Actual Test Result	The rules applied are unblocked/disabled in the local firewalls of the respective OS platforms.
Conclusion	Test Successful.

The blocked IP addresses by honeypot are tested and checked if it's possible to unblock them in the Windows Firewall and Linux IP Tables.

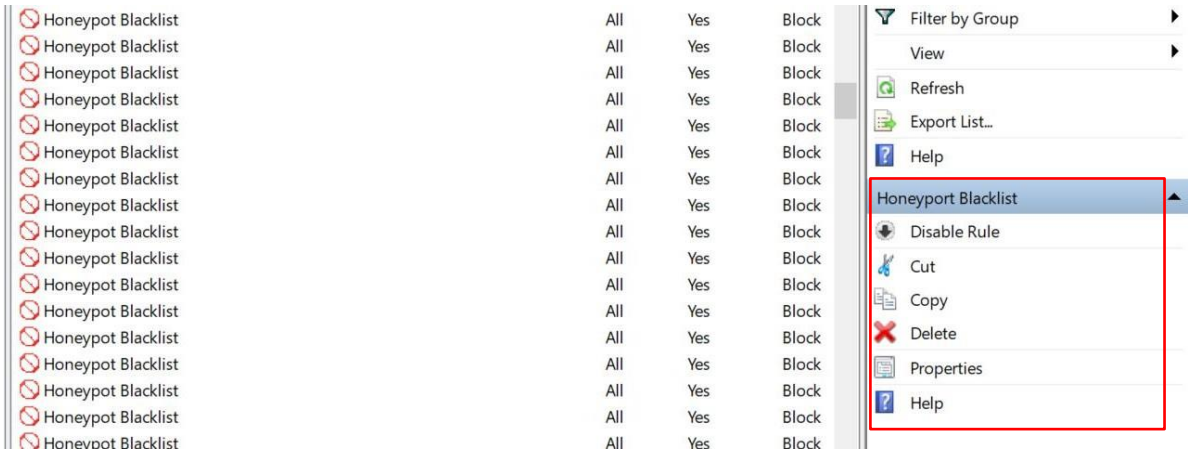


Figure 143 Test Results of Test Case 8 releasing Firewall rules in Windows

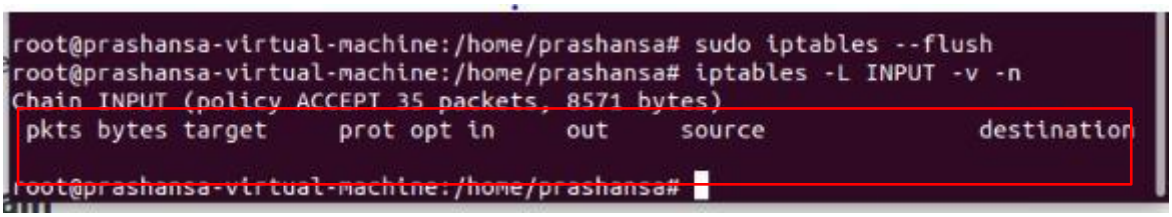


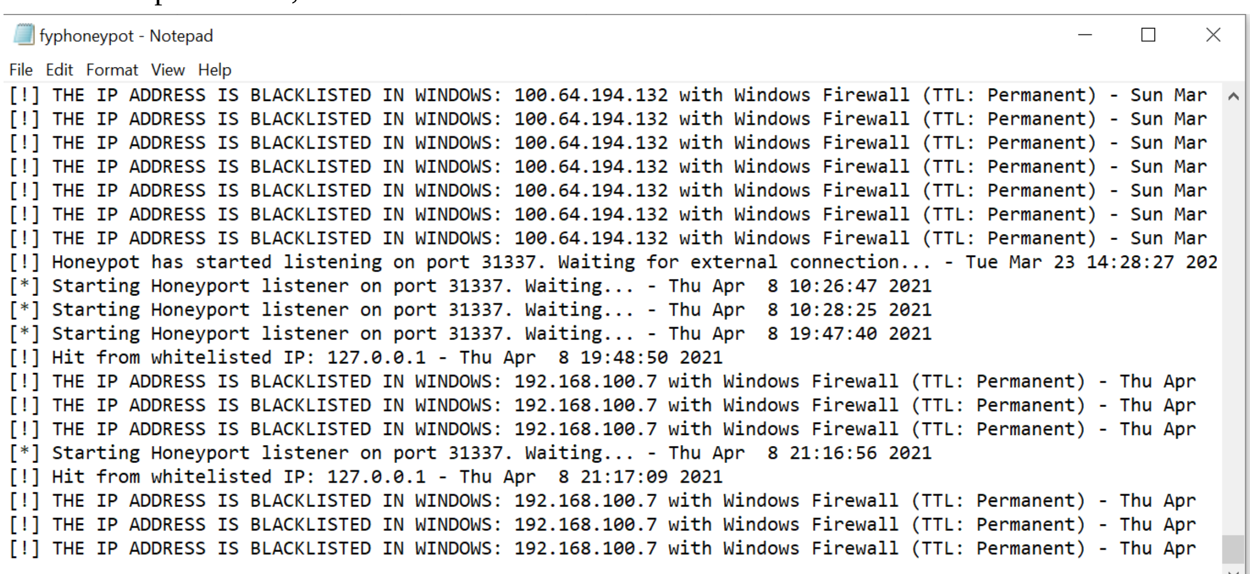
Figure 144 Test Results of Test Case 8 releasing Firewall rules in Linux

7.16.1.1.4 TEST CASE 9

Table 17 Honeypot Module Test Case 9

Test Case 9	
Objective	To test the ability to store logs in text files and csv files.
Action	View the logs recorded from above Test cases 4 and 5.
Expected Test Result	When any connection from Whitelist IP or Blacklist IP is received, the log data should be stored in text and csv files.
Actual Test Result	When any connection from Whitelist IP or Blacklist IP is received, the log data is stored in text file and csv file.
Conclusion	Test Successful.

The log file recorded by the honeypot after every connection is tested in three forms. Normally all the logs are saved as text files at first and then it allows in comma-separated values (CSV) format which lets the user to save data in a tabular format automatically. This feature in CSVs resemble a standard spreadsheet, but with the addition of a. csv extension.


fyphoneypot - Notepad
File Edit Format View Help

```
[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 100.64.194.132 with Windows Firewall (TTL: Permanent) - Sun Mar
[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 100.64.194.132 with Windows Firewall (TTL: Permanent) - Sun Mar
[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 100.64.194.132 with Windows Firewall (TTL: Permanent) - Sun Mar
[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 100.64.194.132 with Windows Firewall (TTL: Permanent) - Sun Mar
[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 100.64.194.132 with Windows Firewall (TTL: Permanent) - Sun Mar
[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 100.64.194.132 with Windows Firewall (TTL: Permanent) - Sun Mar
[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 100.64.194.132 with Windows Firewall (TTL: Permanent) - Sun Mar
[!] Honeypot has started listening on port 31337. Waiting for external connection... - Tue Mar 23 14:28:27 202
[*] Starting Honeyport listener on port 31337. Waiting... - Thu Apr  8 10:26:47 2021
[*] Starting Honeyport listener on port 31337. Waiting... - Thu Apr  8 10:28:25 2021
[*] Starting Honeyport listener on port 31337. Waiting... - Thu Apr  8 19:47:40 2021
[!] Hit from whitelisted IP: 127.0.0.1 - Thu Apr  8 19:48:50 2021
[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 192.168.100.7 with Windows Firewall (TTL: Permanent) - Thu Apr
[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 192.168.100.7 with Windows Firewall (TTL: Permanent) - Thu Apr
[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 192.168.100.7 with Windows Firewall (TTL: Permanent) - Thu Apr
[*] Starting Honeyport listener on port 31337. Waiting... - Thu Apr  8 21:16:56 2021
[!] Hit from whitelisted IP: 127.0.0.1 - Thu Apr  8 21:17:09 2021
[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 192.168.100.7 with Windows Firewall (TTL: Permanent) - Thu Apr
[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 192.168.100.7 with Windows Firewall (TTL: Permanent) - Thu Apr
[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 192.168.100.7 with Windows Firewall (TTL: Permanent) - Thu Apr
```

Figure 145 Test Results of Test Case 9 checking logs in text files

```
331,[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 192.168.100.7 with Windows Firewall (TTL: Permanent) - Thu Apr  8 19:50:43 2021

332,[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 192.168.100.7 with Windows Firewall (TTL: Permanent) - Thu Apr  8 19:50:47 2021

333,[*] Starting Honeyport listener on port 31337. Waiting... - Thu Apr  8 21:16:56 2021

334,[!] Hit from whitelisted IP: 127.0.0.1 - Thu Apr  8 21:17:09 2021

335,[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 192.168.100.7 with Windows Firewall (TTL: Permanent) - Thu Apr  8 21:17:52 2021

336,[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 192.168.100.7 with Windows Firewall (TTL: Permanent) - Thu Apr  8 21:17:56 2021

337,[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: 192.168.100.7 with Windows Firewall (TTL: Permanent) - Thu Apr  8 21:17:59 2021

logno,logdata
```

Figure 146 Test Results of Test Case 9 checking logs in csv format

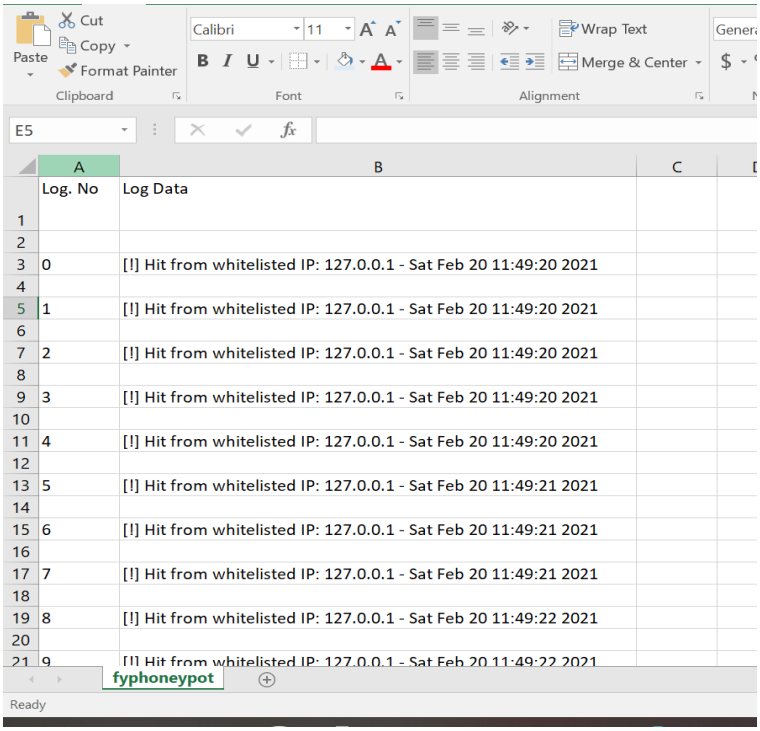


Figure 147 Test Results of Test Case 9 checking logs in spreadsheet

7.16.1.1.5 TEST CASE 10

Table 18 Honeypot Module Test Case 10

Test Case 10	
Objective	To test the ability to store log files and upload in the database.
Action	Enable the Apache and MySQL modules from the XAMPP Control.
Expected Test Result	When any connection from Whitelist IP or Blacklist IP is received, the log data should be stored in text and csv files and should be loaded into the database.
Actual Test Result	When any connection from Whitelist IP or Blacklist IP is received, the log data is stored in text file and csv file and is loaded into the database.
Conclusion	Test Successful.

Now in this test, the logs recorded by honeypot is viewed in the database server.

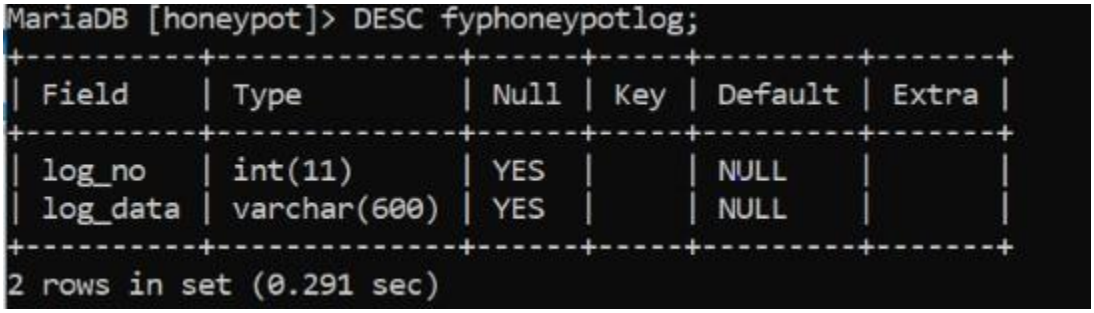


Figure 148 Test Results of Test Case 10 checking data stored in database

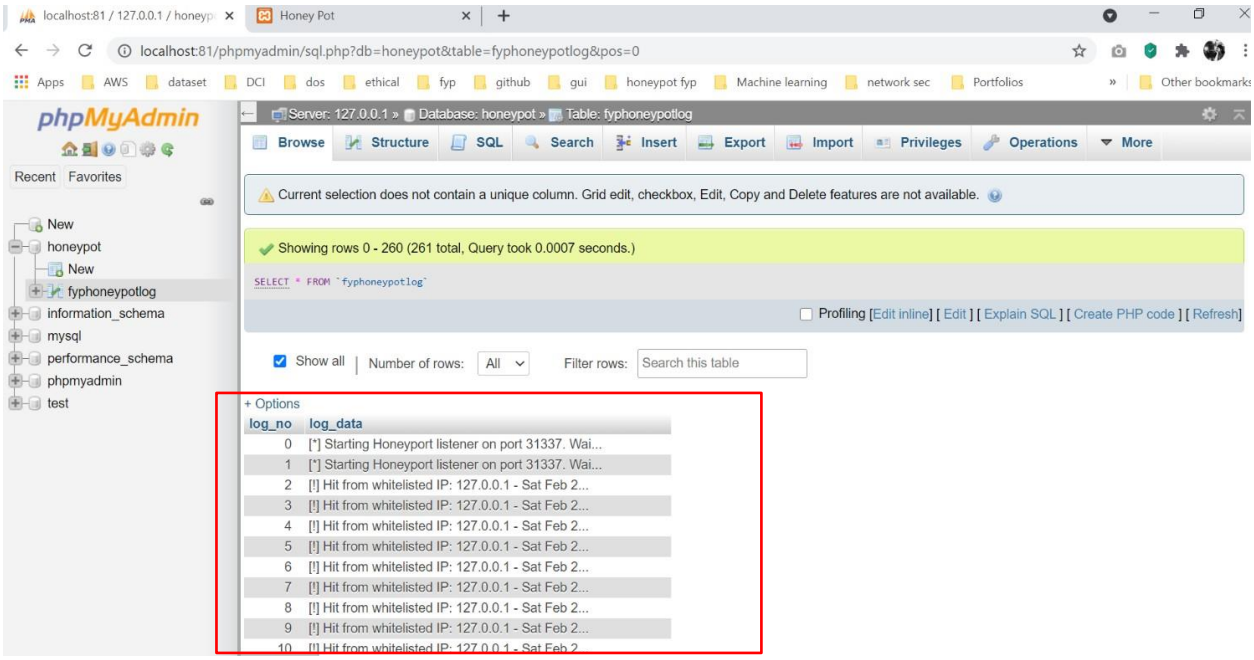


Figure 149 Test Results of Test Case 10 checking data stored in database

7.16.1.1.5 TEST CASE 11

Table 19 Honeypot Module Test Case 11

Test Case 11	
Objective	Connect an external party to the honeypot for email alert.
Action	Execute the honeypot server and check email.
Expected Test Result	The system is expected to send email alert to the system admin with honeypot log file when intruder is detected.
Actual Test Result	The system successfully sends an email with the log file to view to the system admin.
Conclusion	Test Successful.

Following the development of a script to send an email and upload data to a database, testing was carried out to ensure that the email was submitted to the administrator.

```
PS C:\Users\prashansa\OneDrive\Desktop\project> python3 alert.py
email sent
PS C:\Users\prashansa\OneDrive\Desktop\project>
```

Figure 151 Test Results of Test Case 11 executing the script

Figure 150 Test Results of Test Case 11 checking Email received

GO BACK TO TOP

7.16.1.2 ANOMALY DETECTION SYSTEM

7.16.1.2.1 TEST CASE 12

Table 20 Anomaly Detection Test Case 12

Test Case 12	
Objective	To test whether user can select dataset file format other than csv, json, text from the computer.
Action	The 'browse files' button is clicked.
Expected Test Result	Word file should not be displayed as an option to the user.
Actual Test Result	Word file was not displayed as an option to the user.
Conclusion	Test Successful.

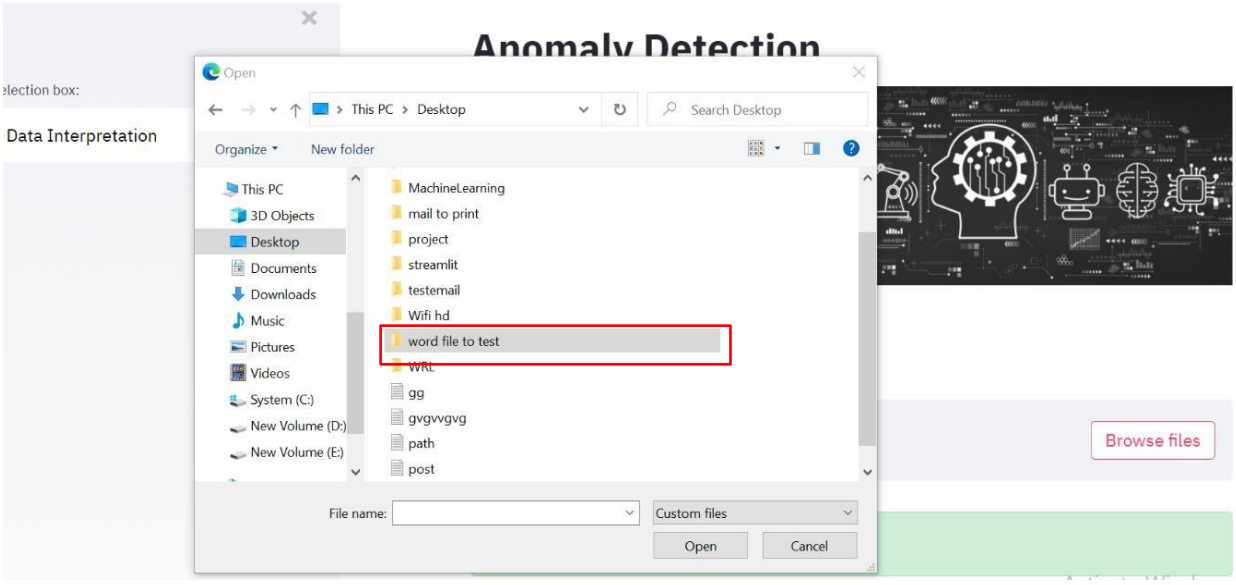


Figure 152 Test Results of Test Case 12 selecting file from browser

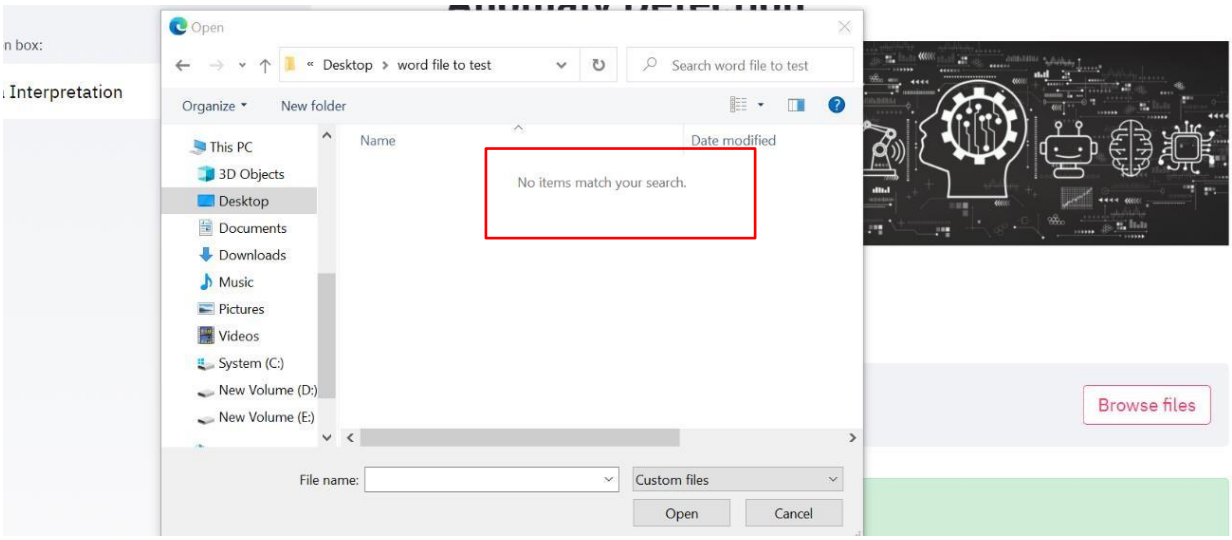


Figure 153 Test Results of Test Case 12 checking if correct file is chosen

7.16.2 SYSTEM TESTING

7.16.2.1 ANOMALY DETECTION UI

In this section of testing, every small details of machine learning interface are tested several times in order to get the expected desired output.

7.16.2.1.1 TEST CASE 6

Table 21 Anomaly Detection UI Test Case 6

Test Case 6	
Objective	To test the ability to view and use machine learning for data visualization and classification of dataset, easily in the web interface.
Action	Run streamlit run app.py in the terminal.
Expected Test Result	The web interface is expected to be executed in the user's browser.
Actual Test Result	The web interface is executed in the user's browser.
Conclusion	Test Successful.

At first, the home interface is tested by executing the command in the terminal.

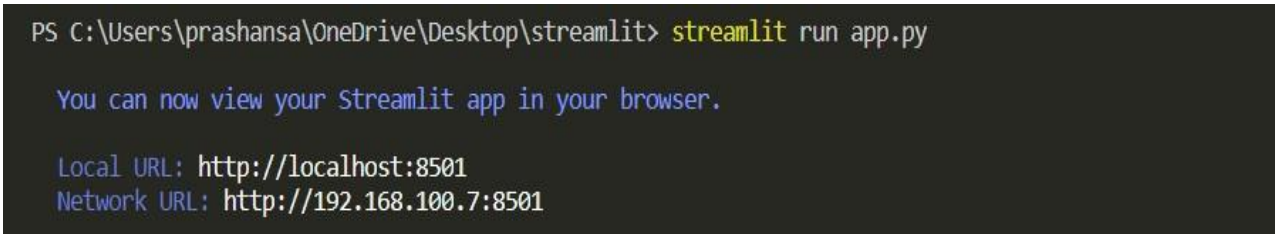


Figure 154 Test Results of Test Case 6 executing anomaly detection interface

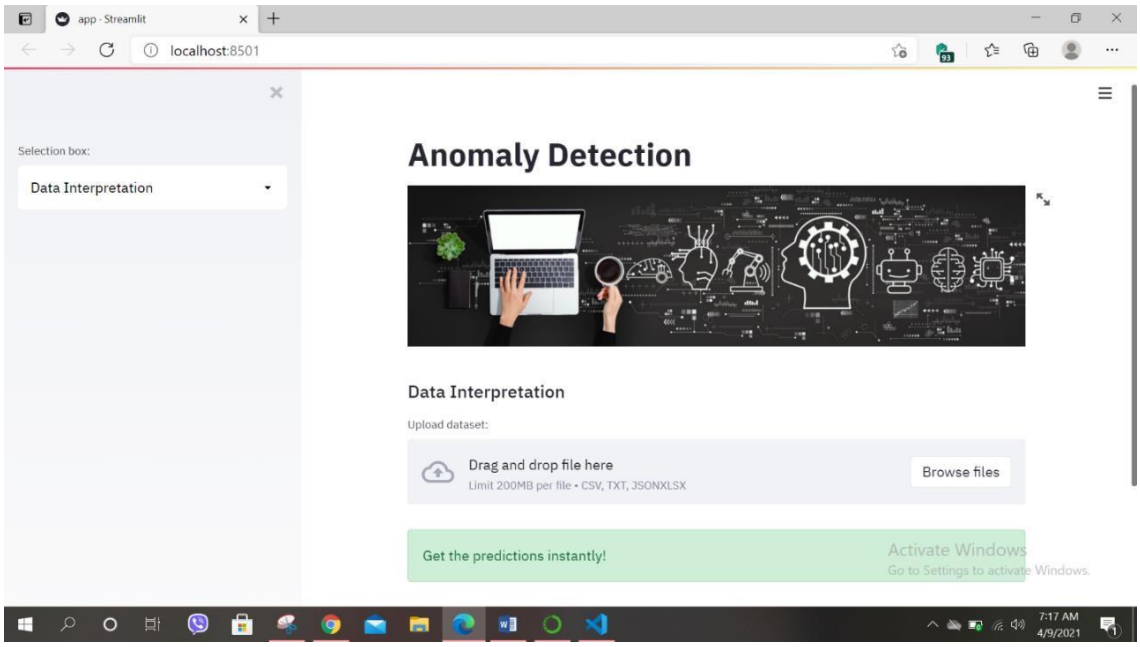


Figure 155 Test Results of Test Case 6 displaying anomaly detection interface

7.16.2.1.2 TEST CASE 7

Table 22 Anomaly Detection UI Test Case 7

Test Case 7	
Objective	To test whether user can browse dataset file from the computer.
Action	The 'browse files' button is clicked.
Expected Test Result	The file explorer dialog box is expected to pop up in the screen.
Actual Test Result	The browse file dialog box was displayed in the screen.
Conclusion	Test Successful.

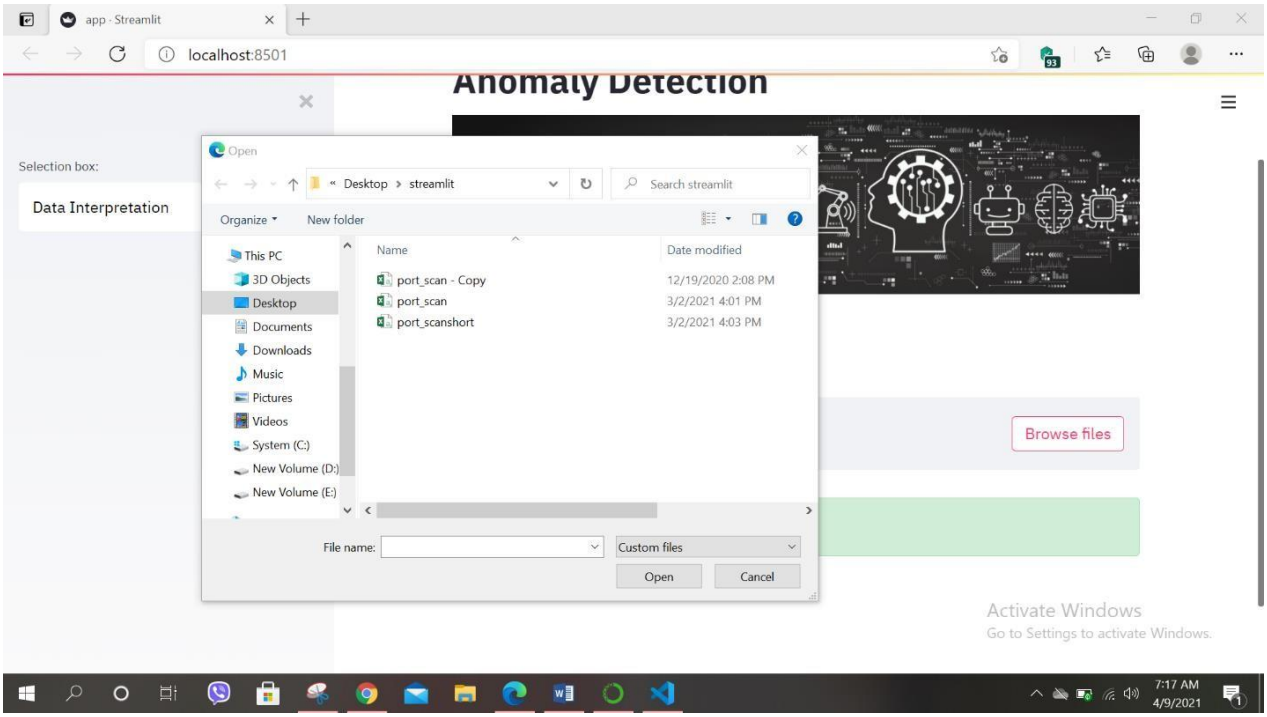


Figure 156 Test Results of Test Case 7 browsing dataset

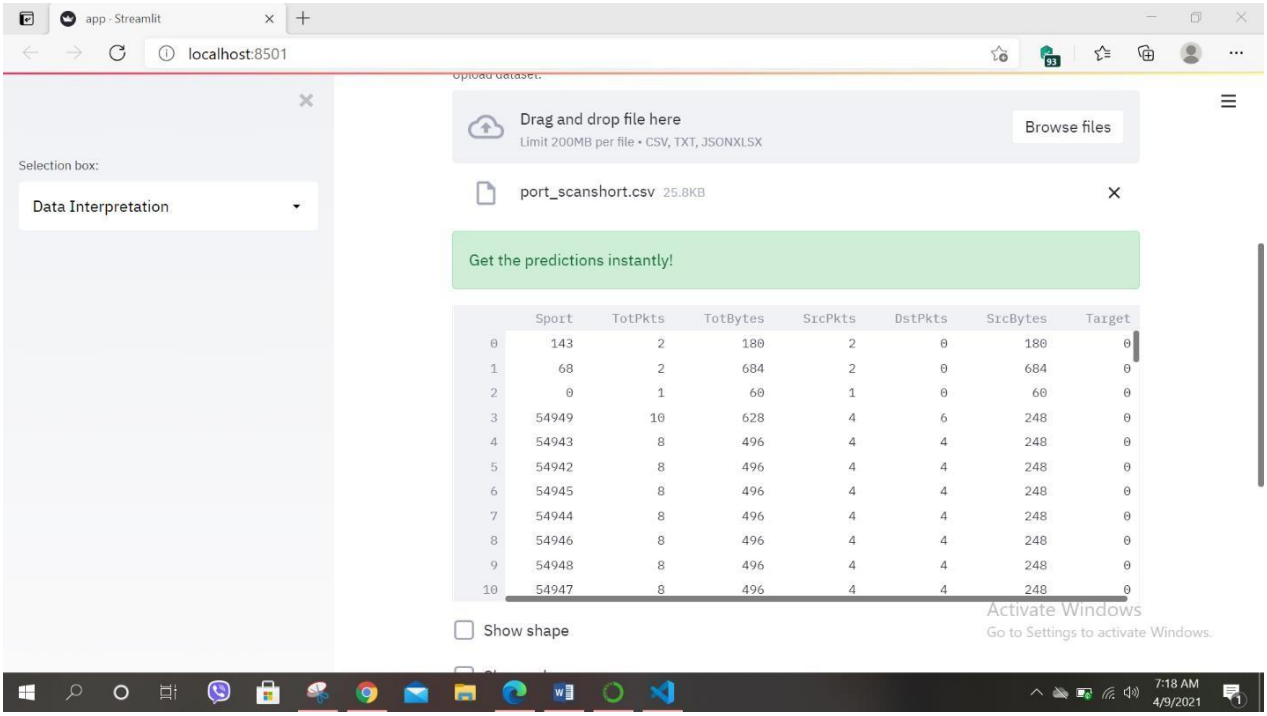


Figure 157 Test Results of Test Case 7 displaying dataset

7.16.2.1.3 TEST CASE 8

Table 23 Anomaly Detection UI Test Case 8

Test Case 8	
Objective	To test the UI check boxes present in the “Data Interpretation” section.
Action	All the check boxes are executed.
Expected Test Result	All the check boxes are expected to function according to their defined functions.
Actual Test Result	All the check boxes functioned according to their defined functions.
Conclusion	Test Successful.

Here, the check boxes present in the Data Interpretation section were executed. The check boxes provide a range of options for performing an initial review of the dataset with the aim of giving insights quickly or identifying gaps or patterns to investigate further.

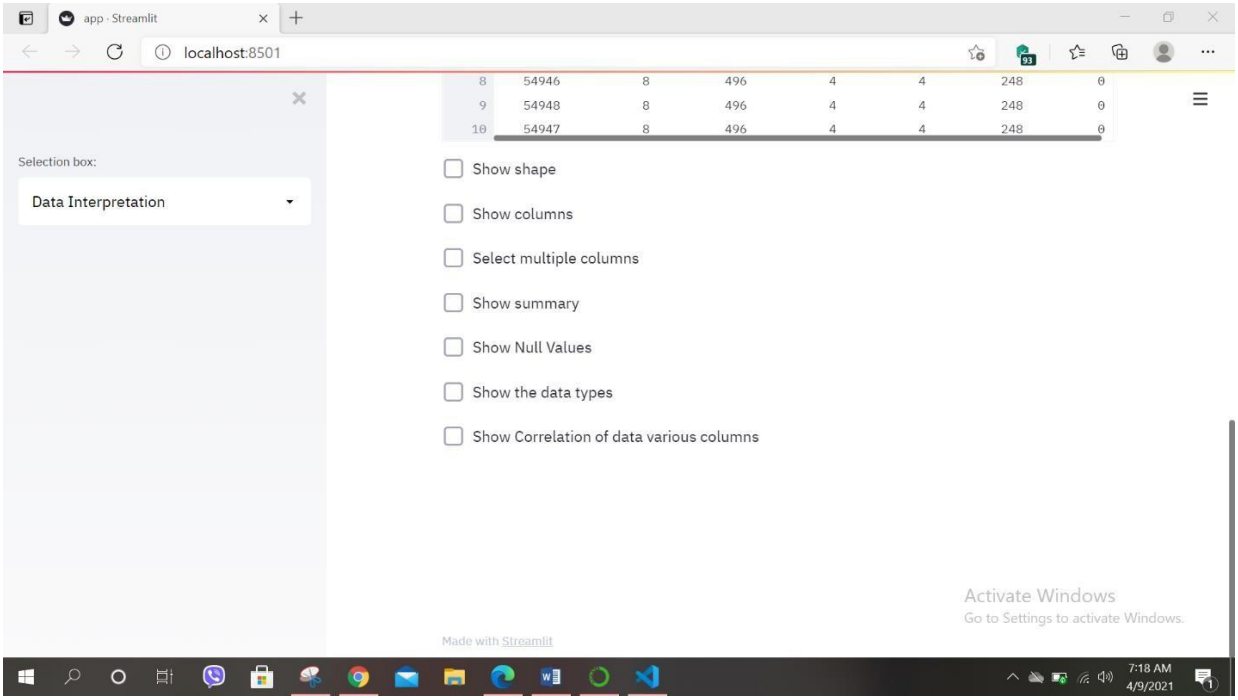


Figure 158 Test Results of Test Case 8 displaying data interpretation check boxes

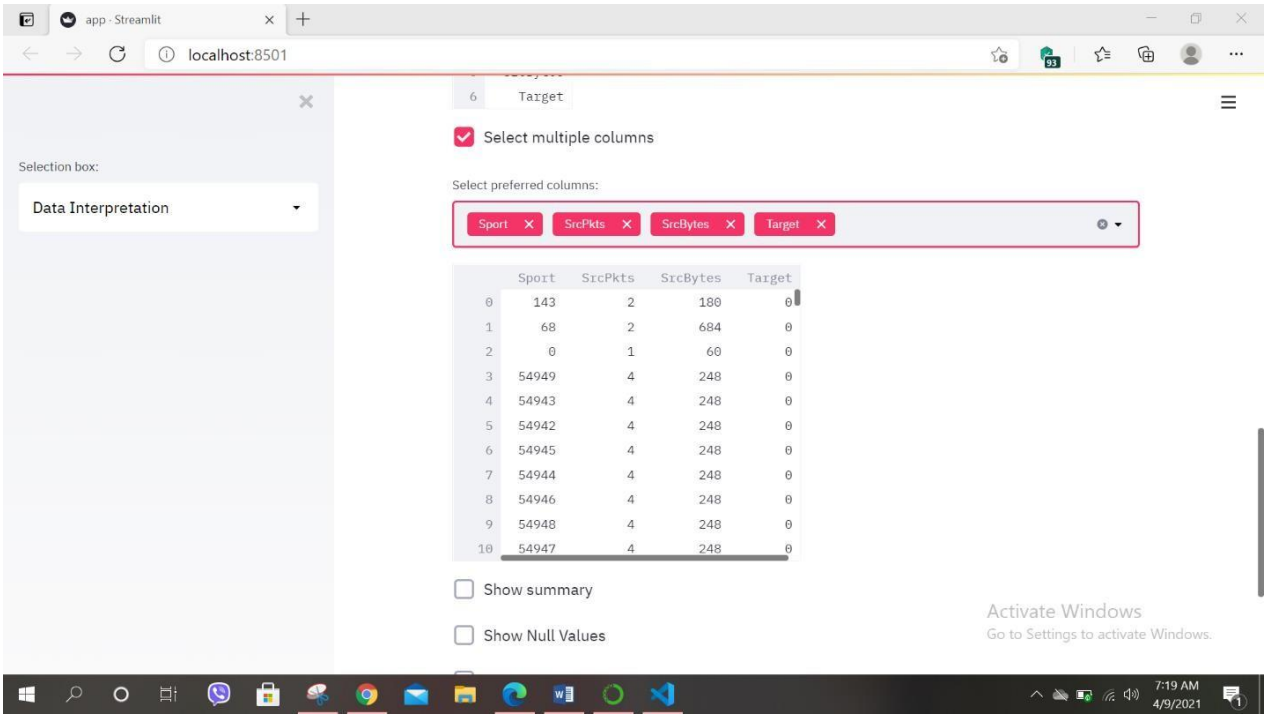


Figure 159 Test Results of Test Case 8 displaying data interpretation check boxes

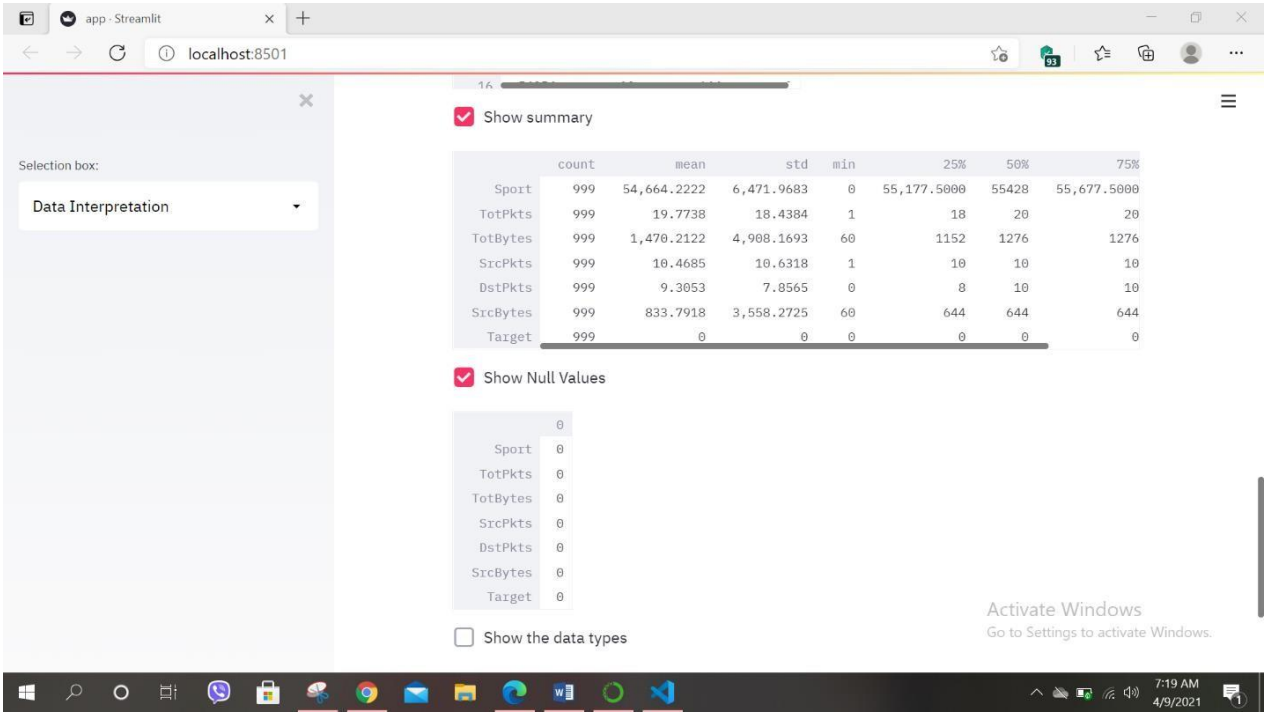


Figure 160 Test Results of Test Case 8 displaying data interpretation check boxes

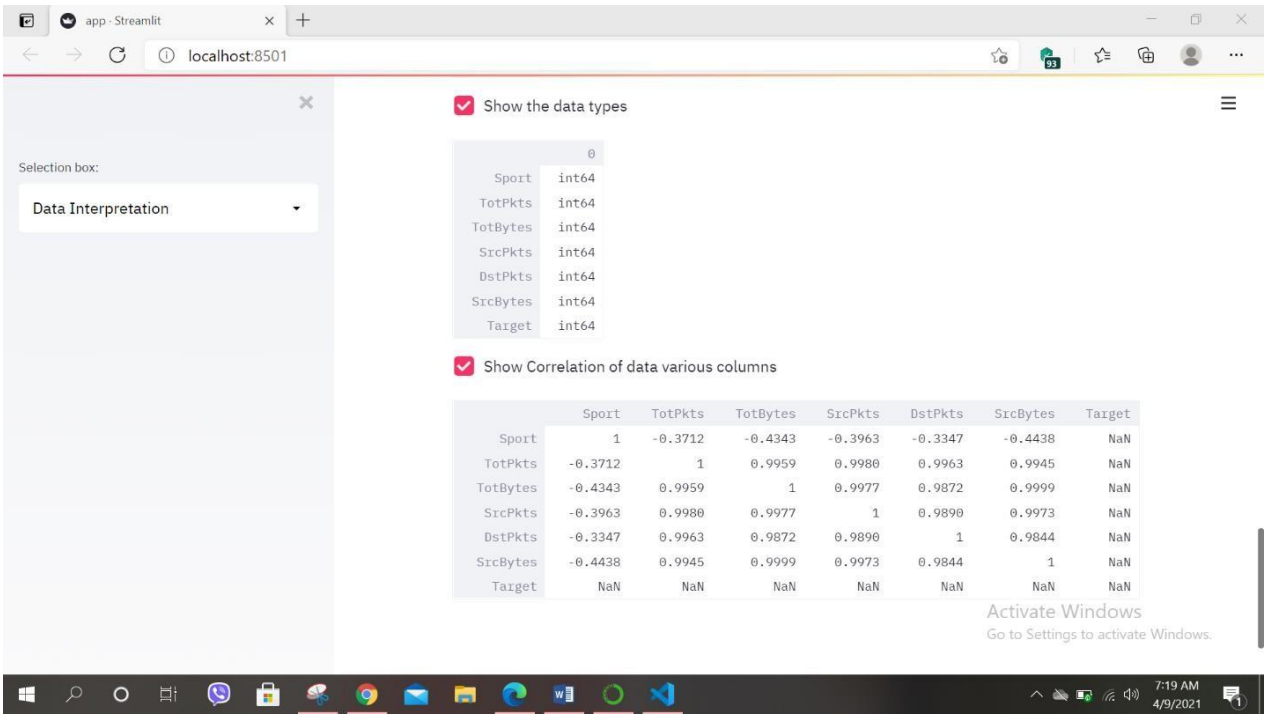


Figure 161 Test Results of Test Case 8 displaying data interpretation check boxes

7.16.2.1.4 TEST CASE 9

Similarly, the check boxes present in the Analysis and Visualization section were executed. The check boxes entails examining the data at a substantial stage using data visualizations with the help of heat map and pair plot.

Table 24 Anomaly Detection UI Test Case 9

Test Case 9	
Objective	To test the UI check boxes present in the "Analysis and Visualization" section.
Action	All the check boxes are executed.
Expected Test Result	All the check boxes are expected to function according to their defined functions.
Actual Test Result	All the check boxes functioned according to their defined functions.
Conclusion	Test Successful.

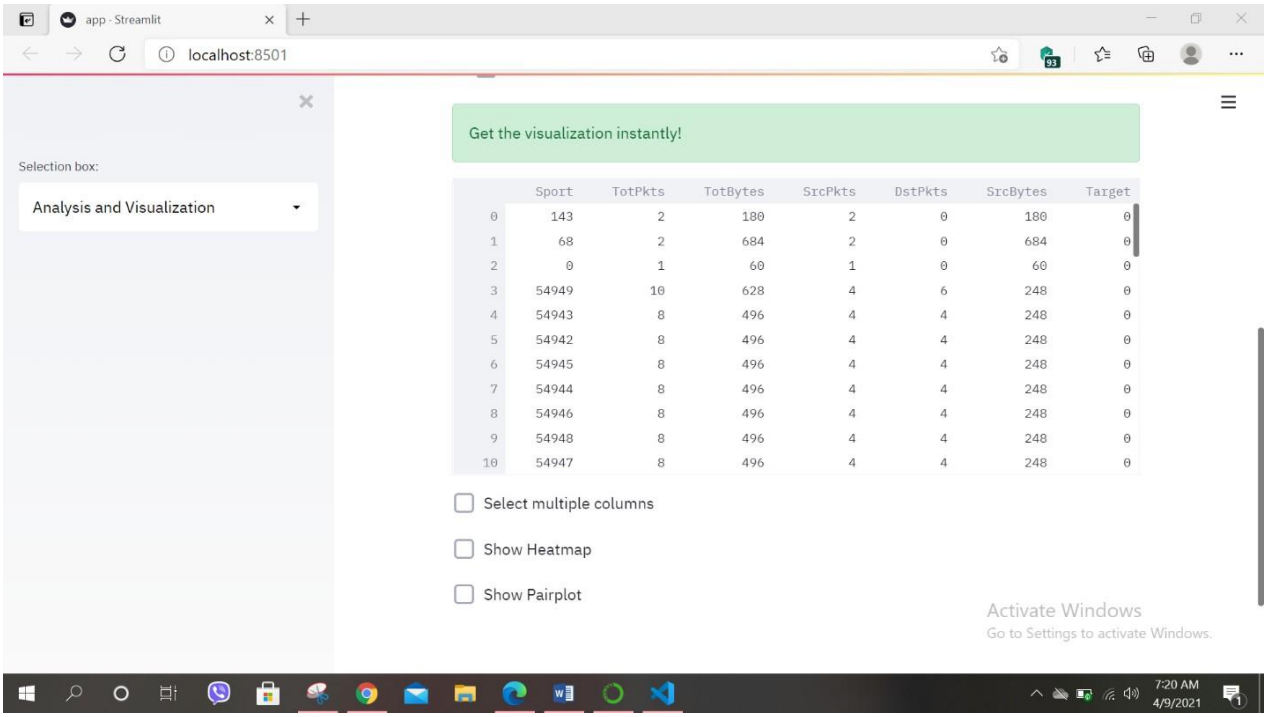


Figure 162 Test Results of Test Case 9 displaying analysis and visualization

Figure 163 Test Results of Test Case 9 displaying graphs

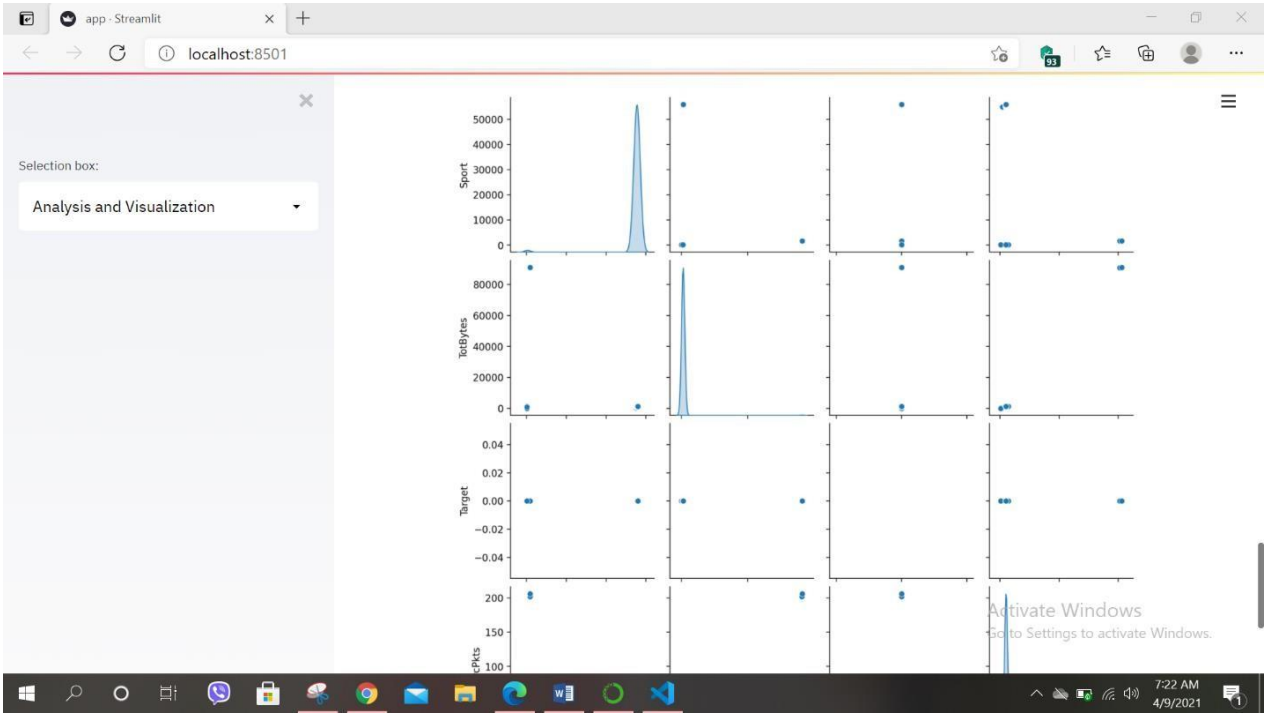


Figure 164 Test Results of Test Case 9 displaying graphs

7.16.2.1.5 TEST CASE 10

Table 25 Anomaly Detection UI Test Case 10

Test Case 10	
Objective	To evaluate Logistic Regression model through accuracy metrics with low random seed and trainable parameters.
Action	Select the preferred multiple columns from the dataset.
Expected Test Result	The selected model should display the accuracy score of the dataset.
Actual Test Result	The selected model displayed the accuracy score of the dataset.
Conclusion	Test Successful.

In this test, the Logistic Regression model is executed with 0.1 parameters and 1 random seed to incur the accuracy metrics.

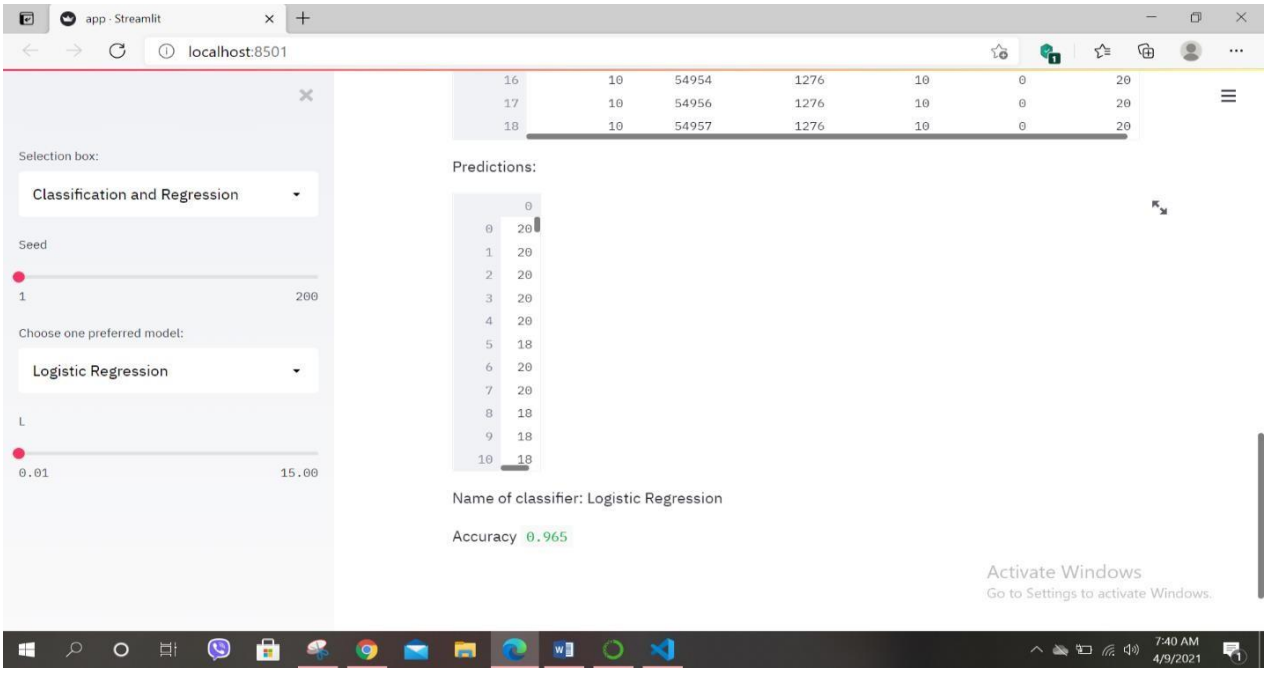


Figure 165 Test Results of Test Case 10 displaying predictions of logistic regression

7.16.2.1.6 TEST CASE 11

Table 26 Anomaly Detection UI Test Case 11

Test Case 11	
Objective	To evaluate Logistic Regression model through accuracy metrics with high random seed and trainable parameters.
Action	Select the preferred multiple columns from the dataset.
Expected Test Result	The selected model should display the accuracy score of the dataset.
Actual Test Result	The selected model displayed the accuracy score of the dataset.
Conclusion	Test Successful.

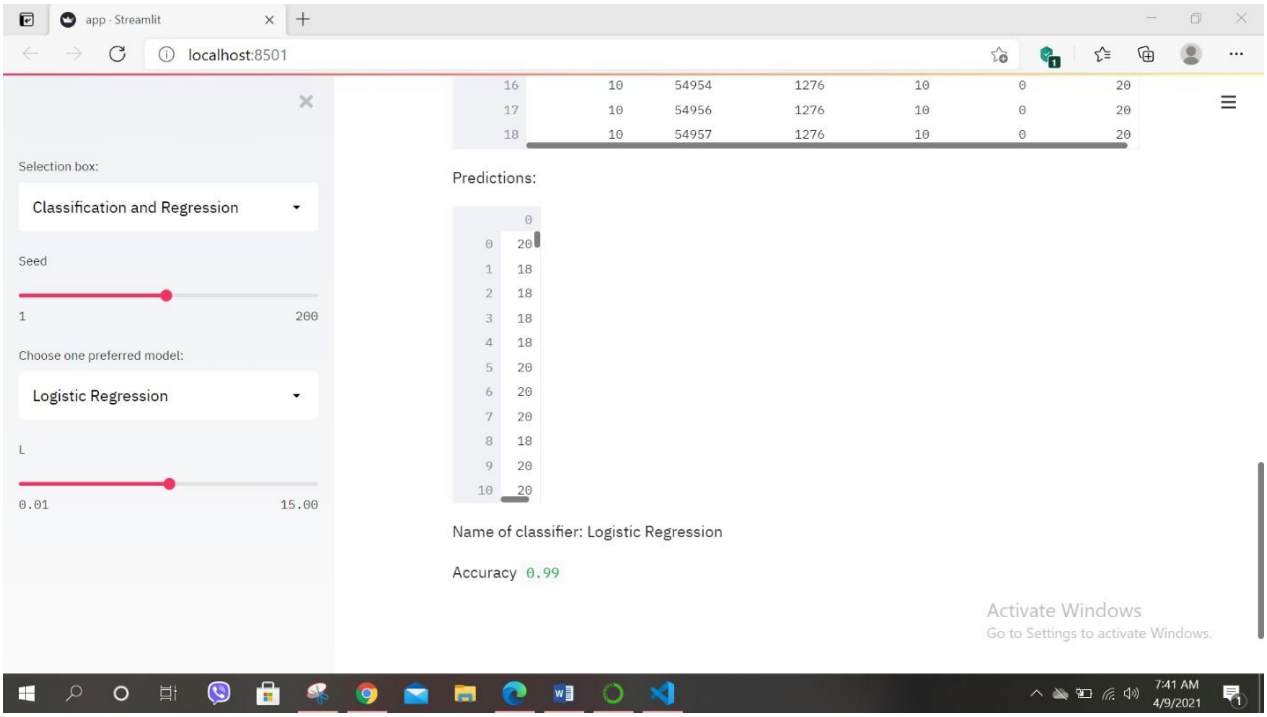


Figure 166 Test Results of Test Case 11 displaying predictions of logistic regression

7.16.2.1.7 TEST CASE 12

Table 27 Anomaly Detection UI Test Case 12

Test Case 12	
Objective	To evaluate Decision Tree model through accuracy metrics with low random seed and trainable parameters.
Action	Select the preferred multiple columns from the dataset.
Expected Test Result	The selected model should display the accuracy score of the dataset.
Actual Test Result	The selected model displayed the accuracy score of the dataset.
Conclusion	Test Successful.

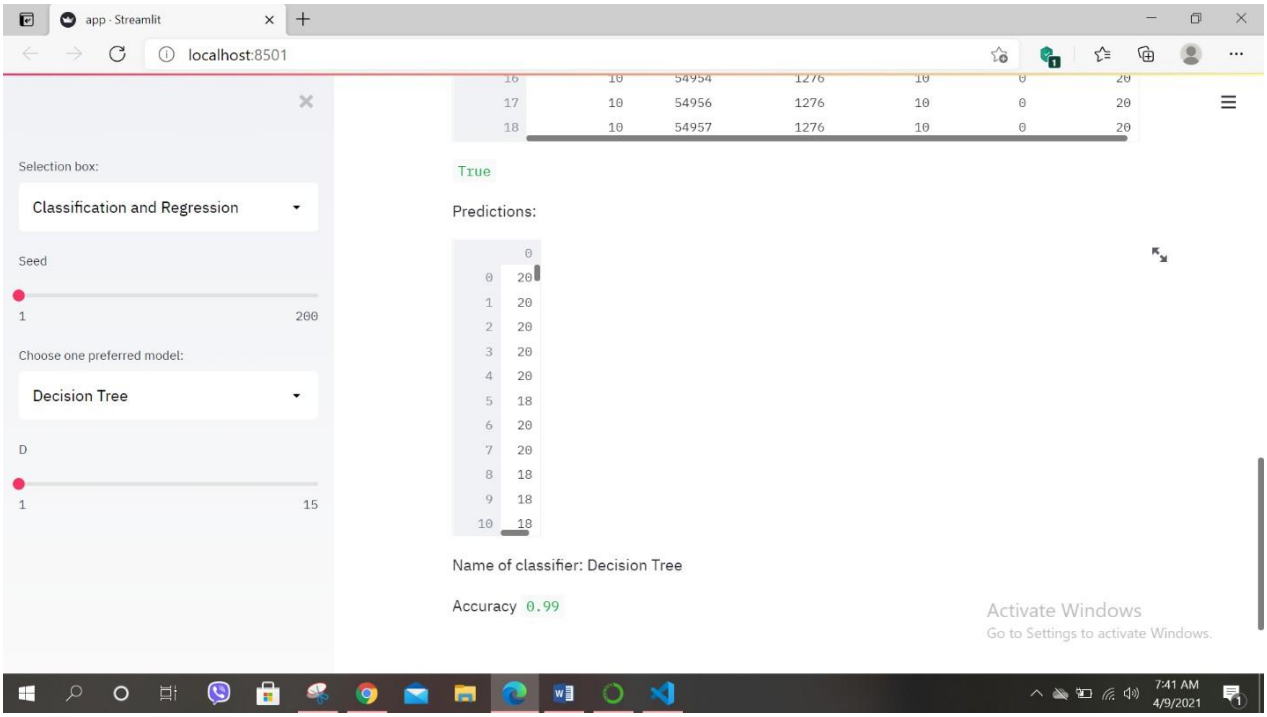


Figure 167 Test Results of Test Case 12 displaying predictions of decision tree

7.16.2.1.8 TEST CASE 13

Table 28 Anomaly Detection UI Test Case 13

Test Case 13	
Objective	To evaluate Decision Tree model through accuracy metrics with high random seed and trainable parameters.
Action	Select the preferred multiple columns from the dataset.
Expected Test Result	The selected model should display the accuracy score of the dataset.
Actual Test Result	The selected model displayed the accuracy score of the dataset.
Conclusion	Test Successful.

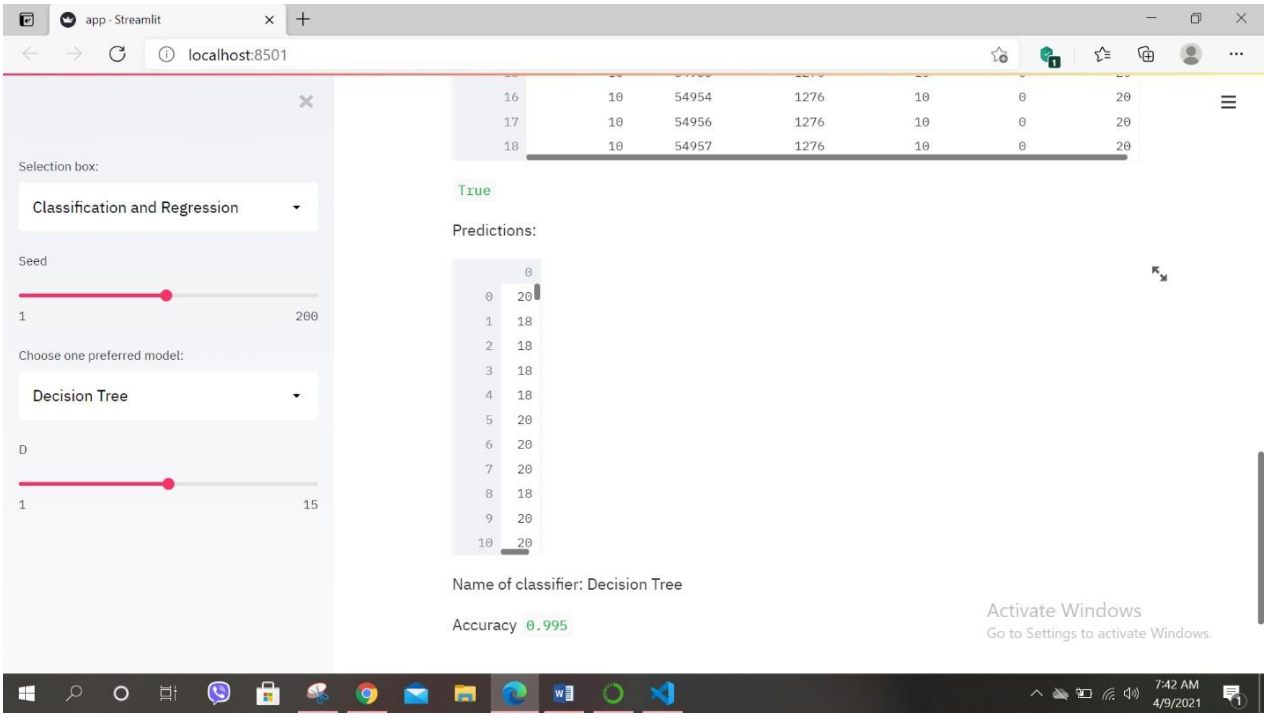


Figure 168 Test Results of Test Case 13 displaying predictions of decision tree

GO BACK TO TOP

7.17 APPENDIX Q: INTERVIEW

Interview Transcript

Interview Date: December 21, 2020

Interview Time: 3 PM-3:30 PM

Interviewer Details:

Islington College

LMU ID: 18030953

Interviewee Details:

Mr. Aaditya Khati

Position: Cyber Security Analyst

Company: CryptoGen Nepal

Naxal, Kathmandu, Nepal

Objective of the Interview:

To learn about honeypot implementation, advantages in the workplace, and some legal implications of honeypot deployment in the network. As part of a final year project at Islington College, which was supervised by London Metropolitan University, the knowledge and results from the interview were used to create the honeypot framework.

Note: This was a much requested interview session, but due to the interviewee's busy schedule, only a few questions could be interrogated via telephone. As a result, the following transcript is a rough translation of the details and points written down during the interview.

Interview Questions:

Prashansa: Do you know any legislation in Nepal that governs the use of honeypots? What are the constraints?

Aaditya: Honeypot usage and data collection are governed by different laws in each country. These rules cover data protection, data processing, and how to handle honeypots. Both of these laws are focused on the quality of data captured by a honeypot and the individual who deploys it. But in case of Nepal, the idea of deploying a honeypot is still relatively new in the Nepalese market, and as a result, Nepalese people's cybersecurity awareness and knowledge are still emerging.

Nepal is also in the preliminary stage of research and study in that specific domain. There really are no laws; instead, it is determined by the reason. You must not publicize your network, and the intruder should arrive on his own, without being asked in.

Prashansa: Do you think deploying a honeypot in the internal network of a company is a good idea?

Aaditya: Yes. It will help IT staff gain knowledge for the number of attacks that they receive plus, it can also be helpful to detect any compromised assets that is trying to propagate through the network. Honeypots have primarily used by researchers to evaluate attacker skills and strategies. However, as I previously stated, they can also be very beneficial to defenders. It's past time for more businesses to consider and use them as a network security measure.

Prashansa: According to you, will adding a machine learning in honeypot improve the service of honeypot?

Aaditya: Yes. It will increase the accuracy of the detection. Machine learning identifies problems by continuously tracking network activity for irregularities. To detect critical events, machine learning engines handle large quantities of data in real time. Insider attacks, malicious malware, and policy breaches can all be detected using these techniques.

Prashansa: Did you come across any specific honeypot cases in Nepal?

Aaditya: No, I have not come across any such cases.

Prashansa: What role do network security administrators play in catching hackers on honeypots?

Aaditya: Network security administrators are tasked to continuously build on the existing rulesets and define more use cases to make the honeypot more reliable which can assist with catching any signs of infiltration before it causes any massive harm to organization's assets and data. They can be extremely useful in analysing, comprehending, monitoring, and monitoring attacker activity in order to build more secure networks.

GO BACK TO TOP

7.18 APPENDIX R: SAMPLE CODES

The planned system's backend and frontend implementation has been accomplished. As the project is developed using programming languages such as Python, HTML, CSS and PHP. The python was used for developing the honeypot and machine learning sections of the system. Other languages were used for building the interfaces. The system comprises of files namely main.py, databasemod.py, alert.py, index.php, log.php, style.css, app.py.The code snippets for each of the developed system files are shown below.

7.18.1 MAIN.PY

```
'''
Usage:
AADH <config_filepath>
Options:
<config_filepath>
-h  --help

--------------------------------------------------------------------------------------------------------------------------------------------
#AUTHOR: PRASHANSA JOSHI
#ISLIGTON COLLEGE,KATHMANDU,NEPAL
#LONDON MET ID: 18030953
#EMAIL_ID: PRASHANSA.JOC@GMAIL.COM

#Acknowlegements:
#logging code: https://realpython.com/python-logging/#:~:text=The%20Logging%20Module,-The%20logging%20module&text=It%20is%20used%20by%20most,homogeneous%20log%20for%20you
#socket code: https://docs.python.org/3/library/socket.html
#windows firewall: https://gist.github.com/scuq/2ad5b178f96b49f0e202
#iptables:https://www.digitalocean.com/community/tutorials/iptables-essentials-common-firewall-rules-and-commands
--------------------------------------------------------------------------------------------------------------------------------------------
'''
port_number = 31337       #Set the port number you want to listen
whiteip = [""]  #Set the ip(s) you want to whitelist
log = "fyphoneypot.log"      #Name the log file location if it exists or leave it blank
response = "WELCOME, YOU HAVE BEEN HAACKED!" #Sending reply to the attacker who connects to the honeypot
response_script=""              #Name the file script to send to the attacker or leave it blank
```

Figure 169 Code Snippet of Main.py 1

```
#importing modules
import sys                  #gives you access to certain variables that the interpreter uses or keeps track of
import getopt               #used to parse a series of arguments, such as sys
import socket               #gives you all the functions and variables to connect client-server connection
import requests             #allows you to give access to HTTP requests
import platform             #enables you the platform, such as hardware, system software, and interpreter version.
import os                   #enables you to interact with the operating system by including functions
from subprocess import CalledProcessError , check_output #When check output() returns a non-zero exit status, an exception is thrown.
import datetime             #allows to work with functions of date and time.
import ctypes               #allows you to check windows admin
import logging              #allows you to handle logs
import nmap                 #allows to import nmap
import alert                #importing alert.py
#import databasemod
```

Figure 170 Code Snippet of Main.py 2

```
check_platform = platform.system() #checking the platform i.e. Windows and Linux

if check_platform == "Windows":  #checking admin privilege in windows platform
    if not ctypes.windll.shell32.IsUserAnAdmin():
        sys.exit("\n[!] Admin privileges are required to modify firewall rules.\n")
        #print ("WELCOME TO WINDOWS PLATFORM")
elif check_platform == "Linux" :  #checking root privilege in Linux platform
    if not os.geteuid()==0:
        sys.exit("\n[!] Root privileges are required to modify firewall rules.\n")
else:                           #Throws error message if suitable platform is not selected
    sys.exit("\n[!] \"{0}\" is not a supported platform.\n".format(pcheck_platform))
if port_number >= 1 and port_number <= 65353:  #Checking if the port number lies in between the range
    try:
        connect_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM) #AF_INET address ipv4 #SOCK_STREAM stands for TCP connection
        connect_socket.bind(("0.0.0.0", 80)) #type the ip address you want to bind to the port
#connecting to destination address

    except socket.error as e:
        sys.exit("[!] Unable to bind to port with error: {0} -- {1} ".format(e[0], e[1]))
else:
    print("[!] Please specify a valid port range (1-65535) in the configuration.")
    sys.exit(2)
```

Figure 171 Code Snippet of Main.py 3

```
logger = logging.getLogger('hp')
formatter = logging.Formatter("%(message)s - %(asctime)s","%c") #formatting the log file output
logger.setLevel(logging.INFO) #setting the level to Info
logger.propagate = True
streamhdlr = logging.StreamHandler()
logger.addHandler(streamhdlr)
streamhdlr.setFormatter(formatter)
if log != "": #a log file is created if the earlier log name doesnpt exist
    try:
        filehdlr = logging.FileHandler(log)
        filehdlr.setFormatter(formatter)
        logger.addHandler(filehdlr)
    except IOError as e:
        sys.exit("[!] Unable to create/append file: {0} -- {1} ".format(e[0], e[1]))

connect_socket.listen(5) # Putting the socket to listening mode
ip_hostadd = connect_socket.getsockname()[0]
logger.info("[*] Starting Honeypot in WINDOWS PLATFORM on port {0}. Waiting...".format(port_number)) #informing about the honeypot started to the admin
```

Figure 172 Code Snippet of Main.py 4

```
while True:

    c, addr = connect_socket.accept() #Establishing connection between the client.
    clientaddress = str(addr[0]) #Receiving the intruder ip address
    if clientaddress in (whiteip,"127.0.0.1" ): #Whitelsiting the given ip address
        logger.info("[!] Hit from whitelisted IP: {0}".format(clientaddress)) #informing the admin
        c.shutdown(socket.SHUT_RDWR)
        c.close() #Closing the connection
    else:
        # Send response to client
        if response_script == "":
            if sys.version_info < (3,0):
                c.sendall(response)
            else:
                c.sendall(bytes(response, 'UTF-8'))
        else:
            res = check_output(["python", response_script, clientaddress])
            if sys.version_info < (3,0):
                c.sendall(res)
            else:
                c.sendall(bytes(res, 'UTF-8'))

        c.shutdown(socket.SHUT_RDWR)
        c.close() #Closing the connection with the attacker
```

Figure 173 Code Snippet of Main.py 5

```
if check_platform == "Linux":
#-A: chain where the rule is appended
#-s: source of the packet
#-j: jump to tatget

    try:
        result = check_output(["/sbin/iptables", "-A", "INPUT", "-s", "{0}".format(clientaddress), "-j", "DROP"])
        logger.info("[!] The IP ADDRESS IS BLACKISTED IN LINUX: {0} with IPTABLES (TTL: {1})".format(clientaddress, "Permanent"))
    except (OSError,CalledProcessError) as e:
        logger.error("[!] Failed to blacklist {0} with IPTABLES ({1}), is iptables on the PATH?".format(clientaddress, e))

elif check_platform == "Windows": #netsh advfirewall is used to control the Windows firewall behaviour
#dir: inbound/potbound traffic
#action: block/allow/bypass
#protocol: any/tcp/udp/icmpv4/icmpv6
#remoteip: destination IP addresses of an outbound packet
    try:
        result = check_output(["netsh", "advfirewall", "firewall", "add", "rule", "name=Honeypot Blacklist", "dir=in", "remoteip={}".format(clientaddress), "protocol=any" , "action=block"], shell=True)
        logger.info("[!] THE IP ADDRESS IS BLACKLISTED IN WINDOWS: {0} with Windows Firewall (TTL: {1})".format(clientaddress, "Permanent"))
        ss= alert.send_email('honeypotalert2021@gmail.com','INTRUDER ALERT!!!','Somebody is trying to connect to the network!!!','fyphoneypot.log')
    except (OSError,CalledProcessError) as e:
        logger.error("[!] Failed to blacklist {0} with Windows Firewall ({1})".format(clientaddress, e.output))
else:
    logger.error("[!] {0} is not a supported platform".format(check_platform))
```

Figure 174 Code Snippet of Main.py 6

```
#initializing port scanner
scanner = nmap.PortScanner()
scanner.scan(hosts=ip_address)
ip_status = scanner[ip_address].state()
print("Scanning visitor in progress...")
sc = {}
#running for loop to print info about all the known ports
for host in scanner.all_hosts():
    detail_info = []
    for proto in scanner.all_protocols():
        lport = scanner[host][proto].keys()
        sc = scanner[host][proto]
        for port in lport:
            a = "port:" + str(port) + "service name: " + sc[port]['name'] + " product name:" + sc[port][
                'product'] + "version:" + detail_info.append(a)
while True:
    ss= alert.send_email('honeypotalert2021@gmail.com',
            'INTRUDER ALERT!!!',
            'Somebody is trying to connect to the network!!!',
            'fyphoneypot.log')
```

Figure 175 Code Snippet of Main.py 7

7.18.2 DATABASEMOD.PY

```
#!/usr/bin/python3
#IMPORTING MODULES
import csv #TO CONVERT THE TEXT INTO COMMA SEPEARATED VALUES
import mysql.connector #TO CONNECT TO THE MYSQL
import pandas as pd  #TO DEAL WITH ROWS AND COLUMNS
import databasemod

with open("fyphoneypot.log", 'r', encoding="ISO-8859-1") as f, open('fyphoneypot.csv', 'a') as f2: # or 'wb' if on python2
    writer = csv.writer(f2)
    writer.writerow(['LOG_NO', 'LOG_DATA']) # replace with your custom column header

    i = 0
    for line in f:
        writer.writerow([i] + line.rstrip().split('|'))
        i += 1
        if i == 1000000000000:
            break #TO STOP COLLECTING LOGS AFTER CERTAIN NUMBER IS FULFILLED
    writer = csv.writer(f2)
    writer.writerow(['LOG_NO', 'LOG_DATA'])  # replace with your custom column header
```

Figure 176 Code Snippet of databasemod.py 1

```
with open('fyphoneypot.csv', 'r') as csvfile:
    csv_reader = csv.reader(csvfile)
    for each in csv_reader:
        print(each)

data = pd.read_csv(r'fyphoneypot.csv')  #DEALING WITH CSV INTO EACH ROWS WITH COLUMNS NAMELY; LOG_NO AND LOG_DATA
df = pd.DataFrame(data, columns=['LOG_NO', 'LOG_DATA'])

db = mysql.connector.connect( #CREADENTIALS TO GIVE FOR CONNECTING TO THE ROOT
#KEEPING IT DEFAULT
    host='localhost',
    user='root',
    password="",
    database="honeypot" #NAME OF THE DATABASE
)
```

Figure 177 Code Snippet of databasemod.py 2

```
cur = db.cursor() #EXECUTES THE DATABASE

cur.execute("CREATE TABLE IF NOT EXISTS fyphoneypotlog(" #CREATING A TABLE NAMED FYPHONEYPOTLOG WITH TWO COLUMNS
            "log_no int, "
            "log_data Varchar(600))")


for i, row in data.iterrows(): #INSERTING THE LOG VALUES INTO THE EACH ROWS
    mysql = "INSERT INTO honeypot.fyphoneypotlog VALUES (%s,%s)"
    cur.execute(mysql, tuple(row))
    db.commit()
```

Figure 178 Code Snippet of databasemod.py 3

7.18.3 ALERT.PY

```
#IMPORTING MODULES
import smtplib
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart
from email.mime.base import MIMEBase
from email import encoders
import os.path

#PASSING EMAIL ID OF THE RECEIVER, THE SUBJECT OF THE MAIL, MESSAGE AND AN ATTACHMENT IN THE ARGUMENTS
def send_email(email_recipient,
               email_subject,
               email_message,
               attachment_location = ''):
#THE MAIL ADDRESS OF THE SENDER
    email_sender = 'honeypot.email2021@gmail.com'
#MIMEMULTIPART IS USED FOR SENDING TEXT OR NON-TEXT MESSAGES
    msg = MIMEMultipart()
    msg['From'] = email_sender
    msg['To'] = email_recipient
    msg['Subject'] = email_subject
    msg.attach(MIMEText(email_message, 'plain'))
```

Figure 179 Code Snippet of alert.py 1

```
    if attachment_location != '':
        filename = os.path.basename(attachment_location)
        attachment = open(attachment_location, "rb")
        part = MIMEBase('application', 'octet-stream')
        part.set_payload(attachment.read())
        encoders.encode_base64(part)
        part.add_header('Content-Disposition',
                        "attachment; filename= %s" % filename)
        msg.attach(part)

    try:
        server = smtplib.SMTP('smtp.gmail.com',587)
        server.ehlo()
        server.starttls()
        server.login('honeypot.email2021@gmail.com', '**********') #ASSIGNING SENDERS EMAIL ID AND PASSWORD
        text = msg.as_string()
        server.sendmail(email_sender, email_recipient, text)
        print('email sending..... ') #SENDING MESSAGE TO THE USER
        server.quit()
    except:
        print("SMPT server connection error") #DISPLAYING ERROR MESSAGE
    return True

send_email('honeypotalert2021@gmail.com',
           'INTRUDER ALERT',
           'SOMEONE IS TRYING TO CONNECT TO THE HONEYPOT!!',
           'fyphoneypot.log')
```

Figure 180 Code Snippet of alert.py 2

7.18.4 INDEX.PHP

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">

        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
        <link rel="preconnect" href="https://fonts.gstatic.com">
<link href="https://fonts.googleapis.com/css2?family=Montserrat:wght@300;600;700&display=swap" rel="stylesheet">

    <link rel="stylesheet" type="text/css" href="style.css">
    <title>Honey Pot</title>
    <link rel="stylesheet" href="">
</head>
```

Figure 181 Code Snippet of index.php 1

```
<body>
    <section id="header">
        <div class="container">
            <div class="row">
                <div class="col">
                    <h1>
                        About <br> Honeypot!
                    </h1>
                    <p>
                        A honeypot is known as the most famous and widely deployed security tool that is purposely built to be targeted and
                        compromised. It is also used for detecting and misdirecting unauthorized access to protect production systems. It is also
                        valuable in analysing attackers' actions and in particular, unknown attacks. <br>
                        To view the logs captured by my honeypot, please click below.
                    </p>
                    <center>
                    <a href="log.php" class="btn home-btn">
                        GET LOG
                    </a>
                    </center>
                </div>
            </div>
        </div>
    </section>

</body>
</html>
```

Figure 182 Code Snippet of index.php 2

7.18.5 LOG.PHP

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">

        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
        <link rel="preconnect" href="https://fonts.gstatic.com">
<link href="https://fonts.googleapis.com/css2?family=Montserrat:wght@300;600;700&display=swap" rel="stylesheet">

    <link rel="stylesheet" type="text/css" href="css/style.css">
  <title>Honey Pot</title>
  <link rel="stylesheet" href="">
</head>
```

Figure 183 Code Snippet of log.php 1

```
<body>
  <section>
    <div class="container mt-2 mb-3">
      <div class="row">
        <div class="col">
          <a href="index.php" class="btn btn-warning log-btn" style="float:right;">HOME</a>
          <a href="log.php" class="btn btn-warning log-btn" style="float:right; margin-right: 20px;">Refresh</a>

              <a href="log.php?id=white" class="btn btn-warning" style="float:left;">Whitelisted</a>
          <a href="log.php?id=black" class="btn btn-warning" style="float:left; margin-left: 20px;">Blacklisted</a>


        </div>
      </div>
    </div>
  </section>
```

Figure 184 Code Snippet of log.php 2

```php
<?php
$id = '';
if (!empty($_GET['id'])) {
  $id = $_GET['id'];

}
else{

$id = '';

}
?>
<?php

if ($id == "") {
  ?>
  <section>
  <div class="container">
       <table class="table">
  <thead class="thead-dark">

    <tr>
      <th scope="col">Log No</th>
      <th scope="col">Data</th>
    </tr>
  </thead>
  <tbody>
      <?php
      $servername = "localhost";
$username = "root";
$password = "";
$database = "honeypot";

$conn = mysqli_connect($servername, $username, $password, $database);

$sql = mysqli_query($conn,"SELECT * FROM fyphoneypotlog");
    while ($record = mysqli_fetch_assoc($sql)) {
```

Figure 185 Code Snippet of log.php 3

```php
?>
<?php

if ($id == "white") {
  ?>
  <section>
  <div class="container">
       <table class="table">
  <thead class="thead-dark">

    <tr>
      <th scope="col">Log No</th>
      <th scope="col">Data</th>
    </tr>
  </thead>
  <tbody>
      <?php
      $servername = "localhost";
$username = "root";                    #servername, username, password, database variables for storing credentains related to database.
$password = "";
$database = "honeypot";

$conn = mysqli_connect($servername, $username, $password, $database);
#mysqli_connect is a function used for connecting to the server.

$sql = mysqli_query($conn,"SELECT * FROM fyphoneypotlog WHERE log_data LIKE '[!] Hit from w%' "); #a query is written to get data starting from Hit
from
    while ($record = mysqli_fetch_assoc($sql)) { #Fetching data from the database
```

Figure 186 Code Snippet of log.php 4

```
    ?>
    <tr>
      <td><?php echo $record['log_no'] ?></td>
      <td><?php echo $record['log_data'] ?></td>
    </tr>
  <?php } ?>
  </tbody>
</table>
  </div>

 </section>
 <?php

}


?>
<?php

if ($id == "black") {
  ?>
  <section>
  <div class="container">
      <table class="table">
  <thead class="thead-dark">

    <tr>
      <th scope="col">Log No</th>
      <th scope="col">Data</th>
    </tr>
  </thead>
  <tbody>
      <?php
      $servername = "localhost";
$username = "root";
$password = "";
$database = "honeypot";
```

Figure 187 Code Snippet of log.php 5

```
$conn = mysqli_connect($servername, $username, $password, $database);

$sql = mysqli_query($conn,"SELECT * FROM fyphoneypotlog WHERE log_data LIKE '[+] B%' "); #a query is written to get data starting from [+] B
    while ($record = mysqli_fetch_assoc($sql)) {


    ?>
    <tr>
      <td><?php echo $record['log_no'] ?></td>
      <td><?php echo $record['log_data'] ?></td>
    </tr>
  <?php } ?>
  </tbody>
</table>
  </div>

 </section>
 <?php

}


?>

</body>
</html>
```

Figure 188 Code Snippet of log.php 6

7.18.6 STYLE.CSS

```css
body{
    font-family: 'Montserrat', sans-serif;
}
h1{
    margin-top: 15%;
    text-align: center;
    font-family: 'Montserrat', sans-serif;
    font-weight: 700;
}
p{
    text-align: center;
    font-family: 'Montserrat', sans-serif;
    font-weight: 300;
}
#header{
    background-color: #fdcc52;
    height: 100vh;
}
a.home-btn{
    background-color: black !important;
    color: white !important;
    font-family: 'Montserrat', sans-serif;
    font-weight: 600;
}
a.home-btn:hover{
    opacity: 0.8;
}
a.log-btn{
    font-family: 'Montserrat', sans-serif;
    font-weight: 600;
}
a.log-btn:hover{
    opacity: 0.8;
}
```

Figure 189 Code Snippet of style.css

7.18.7 APP.PY

```
#import inspect
#src=inspect.getsource(modulename)
#print(src)


#importing streamlit module as stm
import streamlit as stm
#for working with arrays importing numpy library as np
import numpy as np
#for data analytics and data science importing pandas library as pd
import pandas as pd
#for satistical graphics and virtualization patterns importing seaborn as sns
import seaborn as sns
#for basic plotting consisting of bars, lines, piecharts
import matplotlib
#a set of functions that render matplotlib possible
import matplotlib.pyplot as plt
#from sklearn.model slection importing traintestsplit for splitting dataset in supervised ML
from sklearn.model_selection import train_test_split
#from sklearn.tree importing decision tree
from sklearn.tree import DecisionTreeClassifier
#from sklearn.linear_model importing logistic regression algorithm
from sklearn.linear_model import LogisticRegression
#from sklearn.metrics importing accuracy classification score
from sklearn.metrics import accuracy_score
#from sklearn importing model_selection for training a series of models with varying hyperparameter values by using algorithms
from sklearn import model_selection
#first thing to do before importing other matplotlib packages. Agg backend is for writing file which is the default backend of matplotlib
```

Figure 190 Code Snippet of app.py 1

```
#first thing to do before importing other matplotlib packages. Agg backend is for writing file which is the default backend of matplotlib
matplotlib.use('Agg')
#for importing image in the UI
from PIL import Image
#setting title in the UI
stm.title('Anomaly Detection')
image = Image.open('image.jpeg')
#setting width of the image to the column width
stm.image(image,use_column_width=True)
#displaying the text
#stm.write accepts many arguments and different data types
#stm.write(""" ### Hello,""")
```

Figure 191 Code Snippet of app.py 2

```
def main():
  #creating list of options for the sidebar
  options_list=['Data Interpretation','Analysis and Visualization','Classification','About us']
  #including a selectbox to the sidebar
  choice=stm.sidebar.selectbox('Selection box:',options_list)
  #option 1
  if choice=='Data Interpretation':
    #displaying a subheader
    stm.subheader("Data Interpretation")
    #inserting a dataset file uploader that accepts any csv,xlsx, text or json file one at a time
    input_data= stm.file_uploader("Upload dataset:",type=['csv','txt','json''xlsx',])
    #loading a green success message after the file has been uploaded
    stm.success("Get the predictions instantly!")
```

Figure 192 Code Snippet of app.py 3

```
if input_data is not None:
    #to read a file
    df=pd.read_csv(input_data)
    #displays a dataframe as a table
    stm.dataframe(df.head(80))
    #displaying checkboxes
    if stm.checkbox("Show shape"):
        #describing a data set
        stm.write(df.shape)
        #displaying the all the names of total columns of the dataset
    if stm.checkbox("Show columns"):
        stm.write(df.columns)
    if stm.checkbox("Select multiple columns"):
        #displaying selective columns of the dataset you want to display
        selected_columns=stm.multiselect("Select preferred columns:",df.columns)
        df1=df[selected_columns]
        stm.dataframe(df1)
        ####################
    if stm.checkbox("Show summary"):
        stm.write(df.describe().T)
        #Displaying any null values in the dataset
    if stm.checkbox("Show Null Values"):
        stm.write(df.isnull().sum())
        #Displaying the data type
    if stm.checkbox("Show the data types"):
        stm.write(df.dtypes)
        #displaying the correaltion between the columns
    if stm.checkbox("Show Correlation of data various columns"):
        stm.write(df.corr())
```

Figure 193 Code Snippet of app.py 4

```
    #option 2
    elif choice == 'Analysis and Visualization':
      #displaying a subheader
      stm.subheader("Data Analysis and Visualization")
      #inserting a dataset file uploader that accepts any csv,xlsx, text or json file one at a time
      input_data= stm.file_uploader("Upload dataset:",type=['csv','xlsx','txt','json'])
      stm.success("Get the visualization instantly!")
      if input_data is not None:
        #to read a file
        df=pd.read_csv(input_data)
        #displays a dataframe as a table
        stm.dataframe(df.head(50))
        #displaying the all the names of total columns of the dataset
        if stm.checkbox("Select multiple columns"):
          selected_columns=stm.multiselect("Select preferred columns:",df.columns)
          df1=df[selected_columns]
          stm.dataframe(df1)
        #plotting heatmap as per the confusion matrix
        if stm.checkbox('Show Heatmap'):
          stm.write(sns.heatmap(df1.corr(),vmax=1,square=True,annot=True,cmap='viridis'))
          stm.pyplot()
          #plotting pairplot
        if stm.checkbox('Show Pairplot'):
          stm.write(sns.pairplot(df1,diag_kind='kde'))
          stm.pyplot()
          #displaying pie chart
```

Figure 194 Code Snippet of app.py 5

```
      #option 3
    elif choice == 'Classification':
      #displaying a subheader
      stm.subheader("Classification and Regression Building")
      #inserting a dataset file uploader that accepts any csv,xlsx, text or json file one at a time
      input_data= stm.file_uploader("Upload dataset:",type=['csv','xlsx','txt','json'])
    #loading a green success message after the file has been uploaded
      stm.success("Get the predictions instantly!")
      if input_data is not None:
        #to read a file
        df=pd.read_csv(input_data)
        #displaying the all the names of total columns of the dataset
        stm.dataframe(df.head(80))
        #selecting multiple columns
        if stm.checkbox("Select multiple columns"):
          selected_columns=stm.multiselect("Select preferred columns:",df.columns)
          df1=df[selected_columns]
          stm.dataframe(df1)
#When one row is selected, a Pandas Series is returned, and when several rows are selected, a Pandas DataFrame is returned.
          X=df1.iloc[:,0:-1]
          y=df1.iloc[:,-1:]
#setting random seed in ML Experiments
        seed=stm.sidebar.slider('Seed',1,200)
#for heading of the option of ML model algorithm
        classifier_name=stm.sidebar.selectbox('Choose one preferred model:',('Logistic Regression','Decision Tree'))
```

Figure 195 Code Snippet of app.py 6

```
def add_parameter(name_of_clf):

  #creating a dictionary
  params=dict()
  #Hyperparameters: adjustable parameters tuned in order to obtain optimal performance.
  #hyperparameter A hyperparameter is an external configuration to the model whose value cannot be determined from data.
  #hyperparameter of logistic regression. C is the inverse of regularization strength
  if name_of_clf=='Logistic Regression':
    L=stm.sidebar.slider('L',0.01,15.0)
    params['L']=L
  else:
    name_of_clf=='Decision Tree'
    D=stm.sidebar.slider('D',1,15)
    params['D']=D
    return params
```

Figure 196 Code Snippet of app.py 7

```
#calling the add_parameter function
    params=add_parameter(classifier_name)

    def get_classifier(name_of_clf,params):
      clf=None
      if name_of_clf =='Logistic Regression':
        clf=LogisticRegression()
      elif name_of_clf=='Decision Tree':
        clf=DecisionTreeClassifier()
      else:
        #throwing a error message if any model that is not selected in chosen
        stm.warning('Select your algorithm')
      return clf
```

Figure 197 Code Snippet of app.py 8

```
#spliting the loaded dataset into train and test set
    clf=get_classifier(classifier_name,params)
    X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=seed)
    clf.fit(X_train,y_train)
    y_pred=clf.predict(X_test)
    stm.write('Predictions:', y_pred)
#calculating accuracy and precision
    accuracy=accuracy_score(y_test,y_pred)
    stm.write('Name of classifier:',classifier_name)
    stm.write('Accuracy',accuracy)
  elif choice == 'About Us':
    stm.markdown('helo this is my project')


  #ploaded_file = st.file_uploader("Choose a file")
#if uploaded_file is not None:
  #df = pd.read_csv(uploaded_file)
  #st.write(df)

if __name__ == '__main__':
  main()
```

Figure 198 Code Snippet of app.py 9

7.19 APPENDIX S: USE CASE DESCRIPTION

Table 29 Use Case 1

Use Case 1	Super privilege
Actors	Users
Description	The system requires users to execute the script with admin privilege in Windows and root privilege in Linux.
Preconditions	The user should enable Administrator account in Windows and go to 'sudo su' and type password to get to the root privilege.
Standard procedure	Enable the super privilege in respective platform.
Success End Condition	The user can go through the log directory and execute the script.
Failed End Condition	The user gets an error message.

Table 30 Use Case 2

Use Case 2	Basic Configurations
Actors	Users
Description	The system requires some input value to specify the port number to open in the honeypot and state the IP to whitelist. It also requires to input the mail address for alert.
Preconditions	The user should have enabled the admin/root privileges.
Standard procedure	• Open the script • Input the port number, IP, email address • Save the script.
Success End Condition	The user can successfully execute the script.
Failed End Condition	The users gets an error message.

Table 31 Use Case 3

Use Case 3	Start Honeypot
Actors	Users
Description	The user needs to execute the main.py script.
Preconditions	The user should have configured the script.
Standard procedure	Type the command in the terminal.
Success End Condition	The honeypot is started.
Failed End Condition	The status of honeypot running does not display.

Table 32 Use Case 4

Use Case 4	Query Database
Actors	Users
Description	The logs collected by the honeypot should be pushed to the MYSQL database.
Preconditions	The XAMPP MYSQL Server and Apache should be running.
Standard procedure	Run MYSQL Server and Apache server.
Success End Condition	The logs is pushed to the database.
Failed End Condition	The logs does not appear in the database.

Table 33 Use Case 5

Use Case 5	View Logs
Actors	Users

Description	All the data collected by the honeypot can be viewed in a user interface in the browser.
Preconditions	The data must be stored in the database.
Standard procedure	Navigate to the URL localhost:81/honeypot.
Success End Condition	The interface should be displayed.
Failed End Condition	The interface should display error message.

Table 34 Use Case 6

Use Case 6	Monitor Interface
Actors	Users
Description	The data should requires to be updated and should contain two types of logs in the interface.
Preconditions	The data must be stored in the database.
Standard procedure	Navigate to the URL localhost:81/honeypot/logs
Success End Condition	The interface containing updated logs should be displayed.
Failed End Condition	The interface should display error message.

Table 35 Use Case 7

Use Case 7	Predictive analystics
Actors	Users
Description	An anomaly detection interface should be displayed for predicting the accuracy of the dataset.
Preconditions	The user must execute the app.py script.

Standard procedure	Execute the script.
Success End Condition	The interface displays in the default browser.
Failed End Condition	The interface fails to display.

Table 36 Use Case 8

Use Case 8	Stop Honeypot
Actors	Users
Description	The users should be able to stop the system whenever required.
Preconditions	The user must have run the system.
Standard procedure	Execute the script.
Success End Condition	The user must have stopped the system.
Failed End Condition	The disruption in connection status is shown.

Table 37 Use Case 9

Use Case 9	Probe attack
Actors	Attackers
Description	The attacker scans the network through tools like NMAP to know the vulnerability in the system.
Preconditions	The ports must be opened in the system.
Standard procedure	Start the honeypot and wait for the intruder.
Success End Condition	The intrusion should be displayed in the logs.
Failed End Condition	The status of blacklisted logs remains unchanged.

GO BACK TO TOP

7.20 APPENDIX T: DESIGNS

7.20.1 GANTT CHART

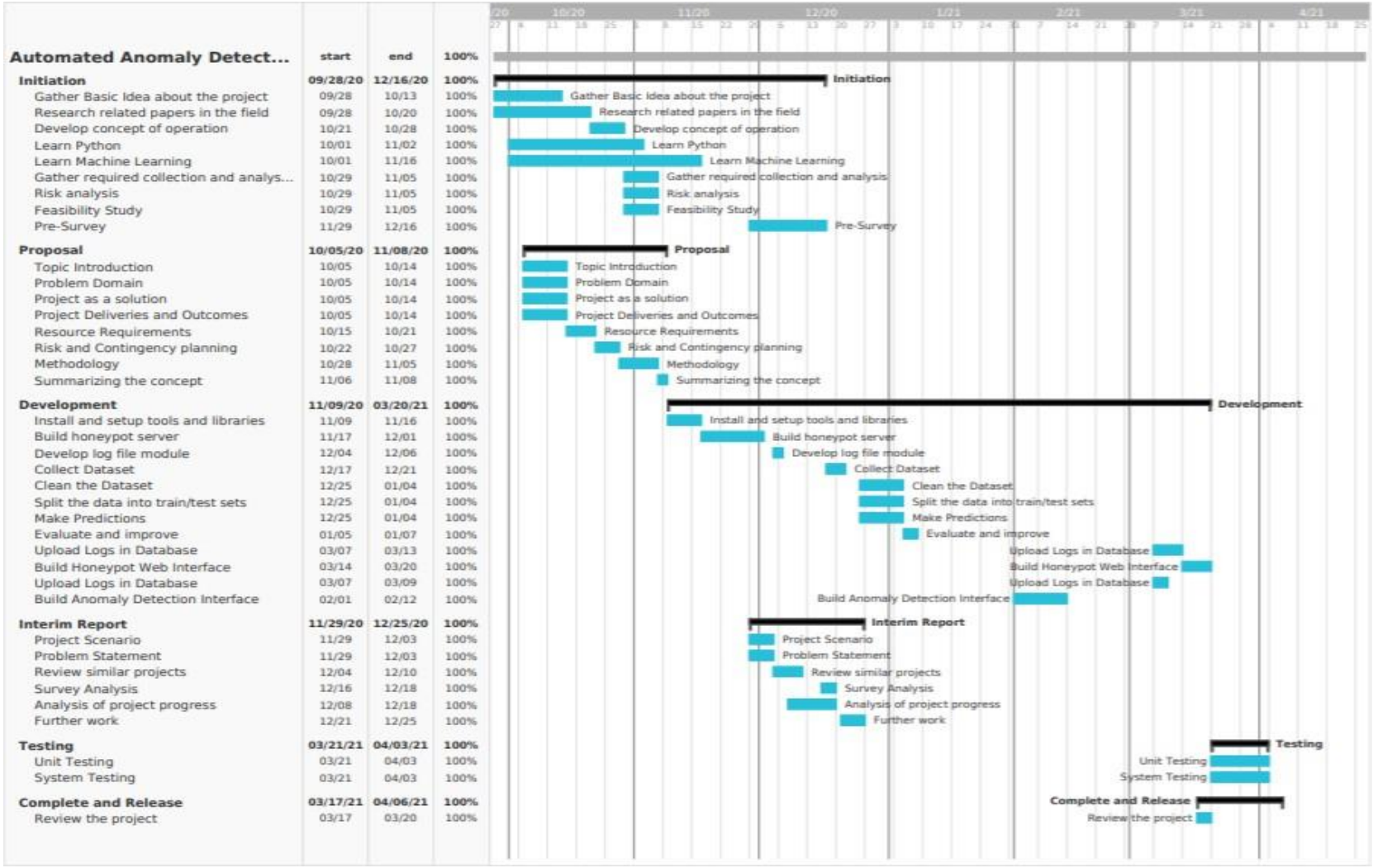


Figure 199 Updated Gantt chart 1

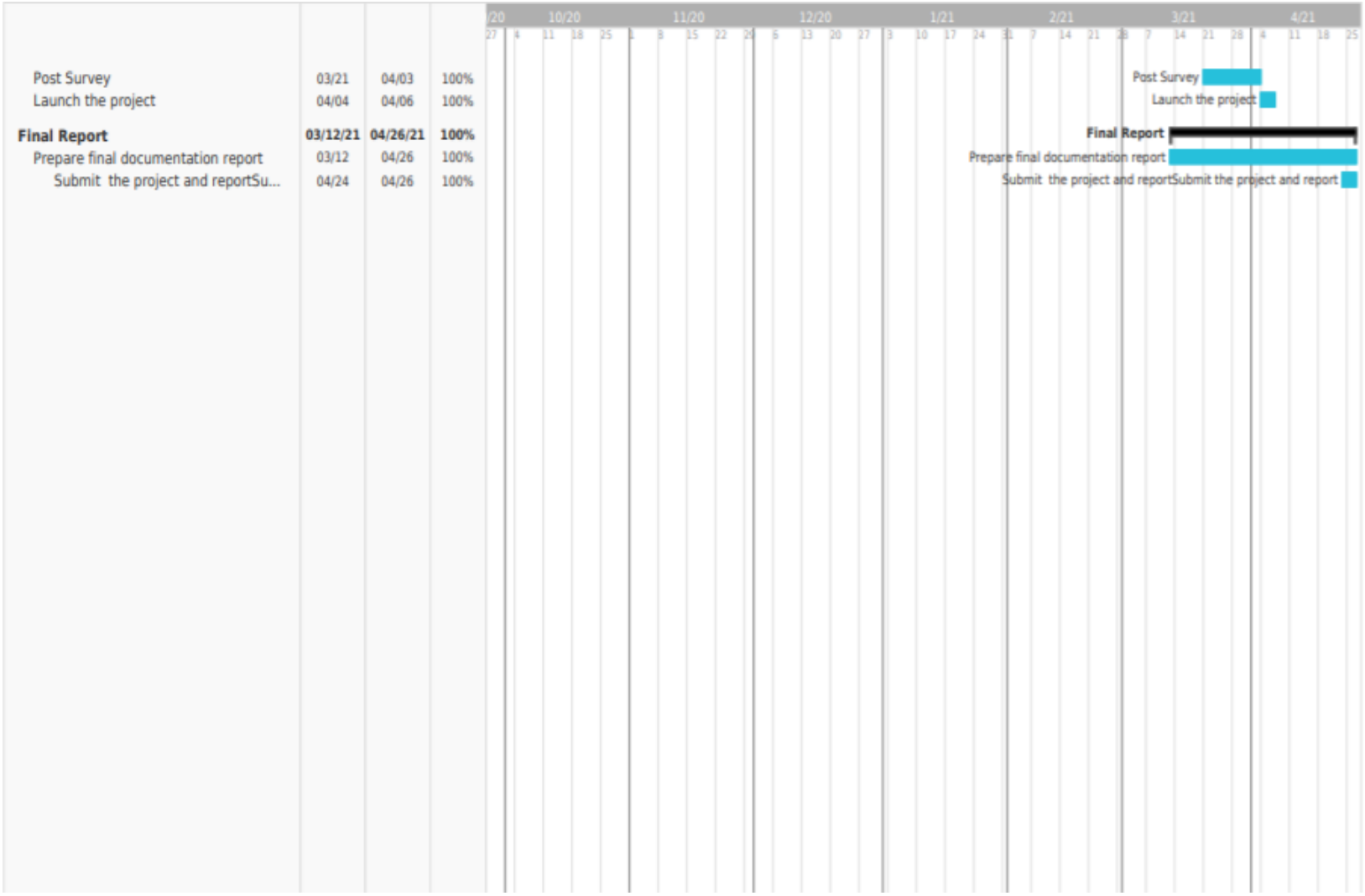


Figure 200 Updated Gantt chart 2

Table 38 Gantt chart table

S.N	Activity	Start Date	End Date	Duration (Days)
1	**Initiation**			
	Gather Basic Idea about the project	**9/28/20**	**10/13/20**	**15**
	Research related papers in the field	**9/28/20**	**10/20/20**	**22**
	Develop concept of Operation	**10/21/20**	**10/28/20**	**7**
	Learn Python	**10/1/20**	**11/1/20**	**31**
	Learn Machine Learning	**10/1/20**	**11/15/20**	**45**
	Gather required collection and analysis	**10/29/20**	**11/5/20**	**7**
	Risk analysis	**10/29/20**	**11/5/20**	**7**
	Feasibility Study	**10/29/20**	**11/5/20**	**7**
	Pre-Survey	**11/29/20**	**12/16/20**	**17**
2	**Proposal**			**0**
	Topic Introduction	**10/5/20**	**10/14/20**	**9**
	Problem Domain	**10/5/20**	**10/14/20**	**9**
	Project as a solution	**10/5/20**	**10/14/20**	**9**
	Project Deliveries and Outcomes	**10/5/20**	**10/14/20**	**9**
	Resource Requirements	**10/15/20**	**10/21/20**	**6**
	Risk and Contingency planning	**10/22/20**	**10/27/20**	**5**
	Methodology	**10/28/20**	**11/5/20**	**8**
	Summarizing the concept	**11/6/20**	**11/8/20**	**2**
3	**Development**			
	Install and setup tools and libraries	**11/9/20**	**11/16/20**	**7**
	Build Honeypot Server	**11/17/20**	**12/1/20**	**14**
	Develop log file Module	**12/4/20**	**12/7/20**	**3**
	Upload Logs in Database	**3/7/21**	**3/13/21**	**6**
	Build Honeypot Web Interface	**3/14/21**	**3/20/21**	**6**
	Collect Dataset	**12/17/20**	**12/21/20**	**4**
	Clean Dataset	**12/25/20**	**1/4/21**	**10**
	Split the data into train/test sets	**12/25/20**	**1/4/21**	**10**
	Make Predictions	**12/25/20**	**1/4/21**	**10**
	Evaluate and improve	**1/5/21**	**1/7/21**	
	Build Anomaly Detection Interface	**2/1/21**	**2/12/21**	
	Evaluate and Improve	**1/5/21**	**1/7/21**	**2**
4	**Interim Report**			
	Project Scenario	**11/29/20**	**12/3/20**	**4**
	Problem Statement	**11/29/20**	**12/3/20**	**4**
	Review similar projects	**12/4/20**	**12/10/20**	**6**
	Pre-Survey Analysis	**12/16/20**	**12/18/20**	**2**

	Analysis of project progress	**12/8/20**	**12/18/20**	**10**
	Further work	**12/21/20**	**12/25/20**	**4**
5	**Testing**			
	Unit Testing	**3/21/21**	**4/3/21**	**13**
	System Testing	**3/21/21**	**4/3/21**	**13**
6	**Complete and Release**			
	Review the overall project	**3/17/21**	**3/20/21**	**3**
	Post Survey	**3/21/21**	**4/3/21**	**13**
	Launch the project	**4/4/21**	**4/6/21**	**2**
7	**Final Report**			
	Prepare Final Documentation Report	**3/12/21**	**4/26/21**	**45**
	Submit the project and report	**4/24/21**	**4/26/21**	**2**

7.20.2 WORK BREAKDOWN STRUCTURE

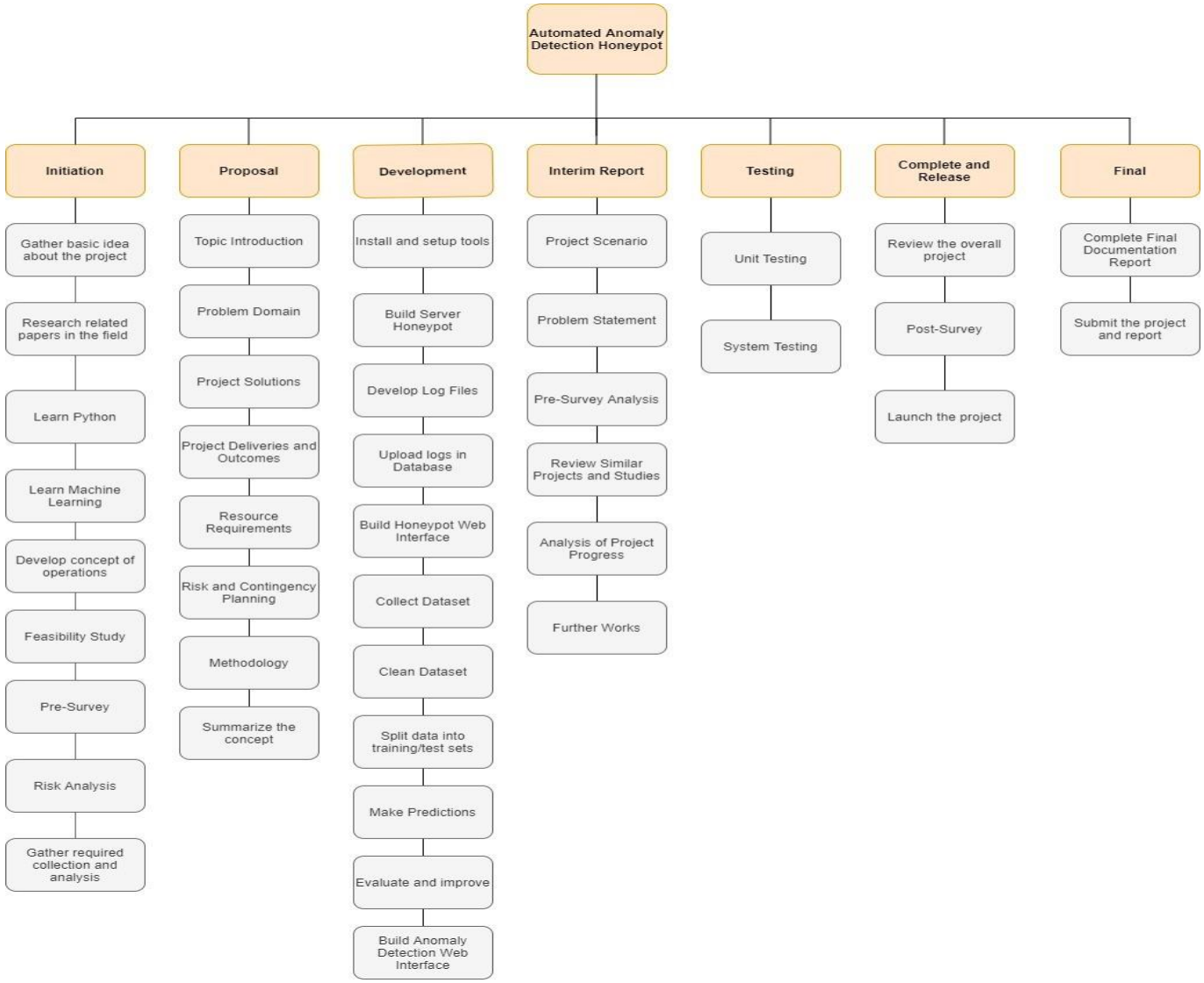


Figure 201 Updated Work Breakdown Structure

7.20.3 WIREFRAME

Wireframes must always be developed before any web application. When working on the project, it serves as a model for programmers and designers. Furthermore, it is a pictorial display of an interface that will be developed in the near future. As a result, wireframes for the proposed framework are created using a tool called Balsamiq Mockup before any implementation begins. The planned interface's wireframes are succinctly summarized below.

7.10.3.1 HONEYPOT UI HOMEPAGE

As referring to the wireframe, the similar homepage should be available at the URLlocalhost:81/honeypot. The user can view a brief description about the honeypot. A log view button is available which takes the user to the log view page.

The homepage consists of a background colour, a short description of honeypot and a button.

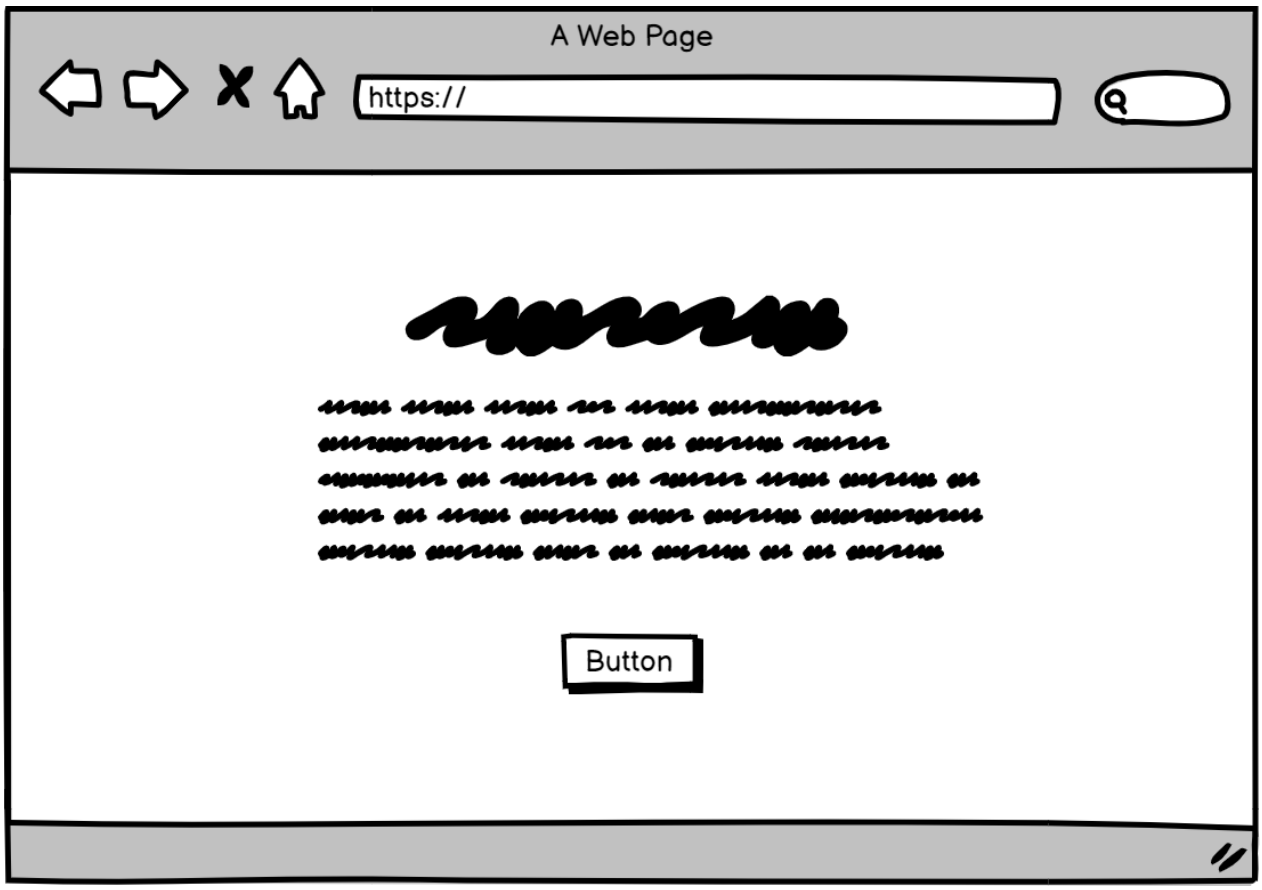


Figure 202 Wireframe of Honeypot UI 1

7.10.3.2 HONEYPOT UI WHITELISTED PAGE

As referring to the wireframe, the page will be available at URL localhost:81/honeypot. The Whitelisted page should have a two buttons on the left sides, allowing the user to select the type of log they want to view. On the right side, there should be two buttons; refresh and home button. Likewise, the whitelisted logs generated by the honeypot should be displayed in each row.

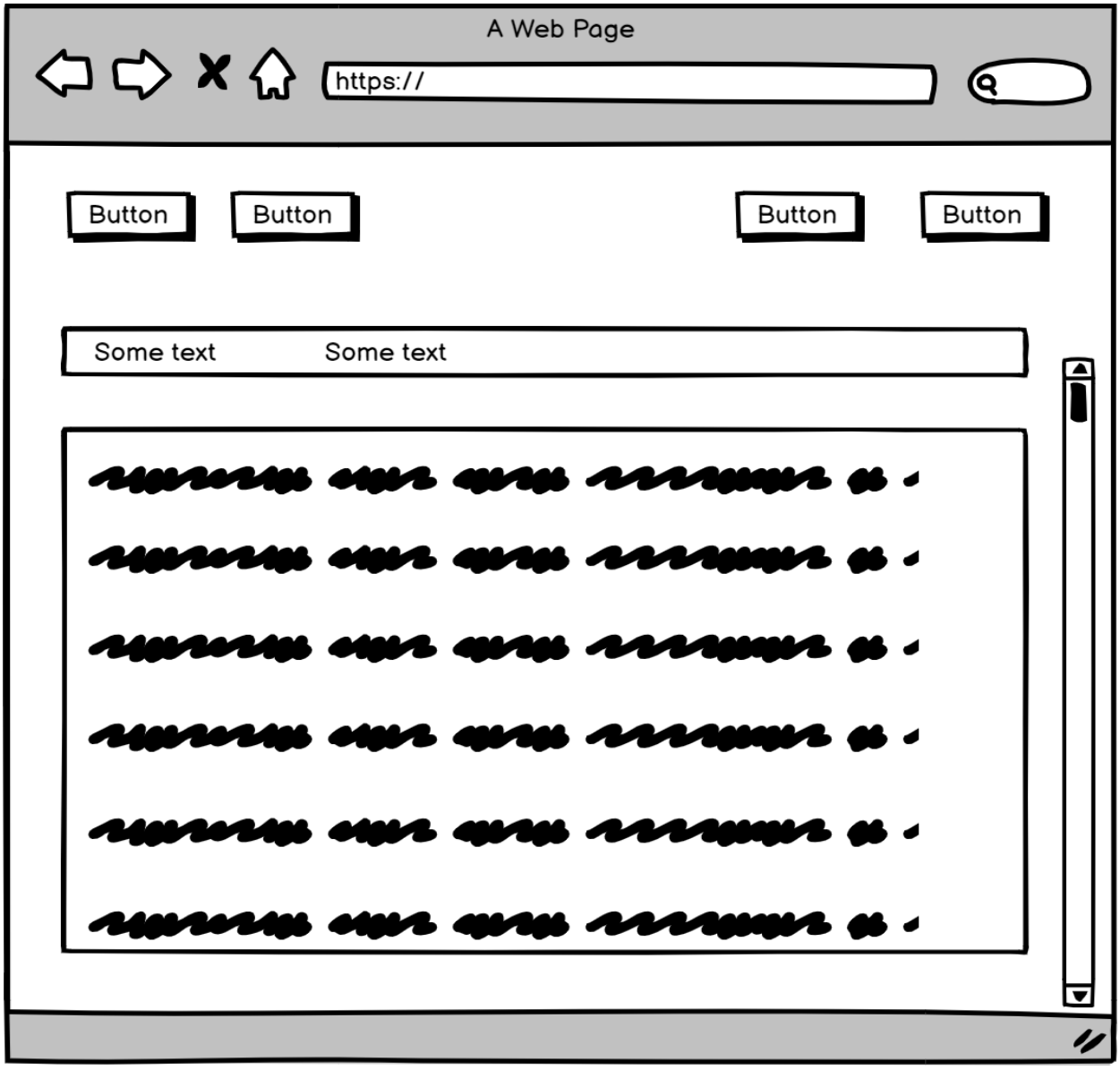


Figure 203 Wireframe of Honeypot UI 2

7.10.3.3 HONEYPOT UI BLACKLISTED PAGE

As referring to the wireframe, the page will be available at URL localhost:81/honeypot/log . The Blacklisted page should have a two buttons on the left sides, allowing the user to select the type of log they want to view. On the right side, there should be two buttons; refresh and home button. Likewise, the blacklisted logs generated by the honeypot should be displayed in each row.

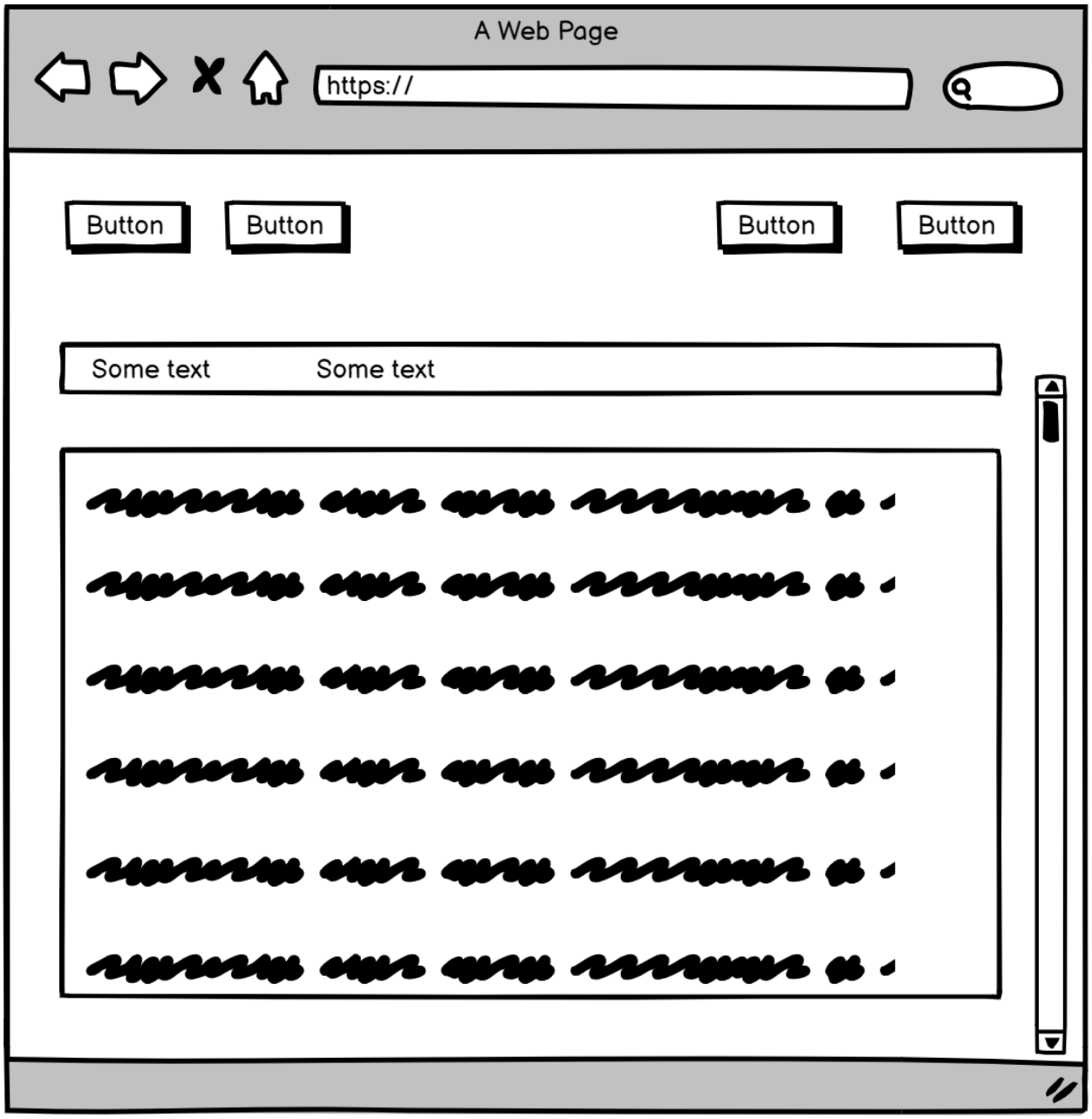


Figure 204 Wireframe of Honeypot UI 3

7.10.3.4 ANOMALY DETECTION UI HOMEPAGE

As per the wireframe, the interface should be available at localhost:8501 or should be automatically executed after running app.py. The page is by default selected in the Data Interpretation option. The users can apply methods for data analysis that incorporates a number of techniques. The page consists of three sections divided. The middle section has an image and browse files button where the user can select the dataset. The left section consists of sidebar. And below the browse button consists of checkboxes which will allow the users to view and summarize the dataset.

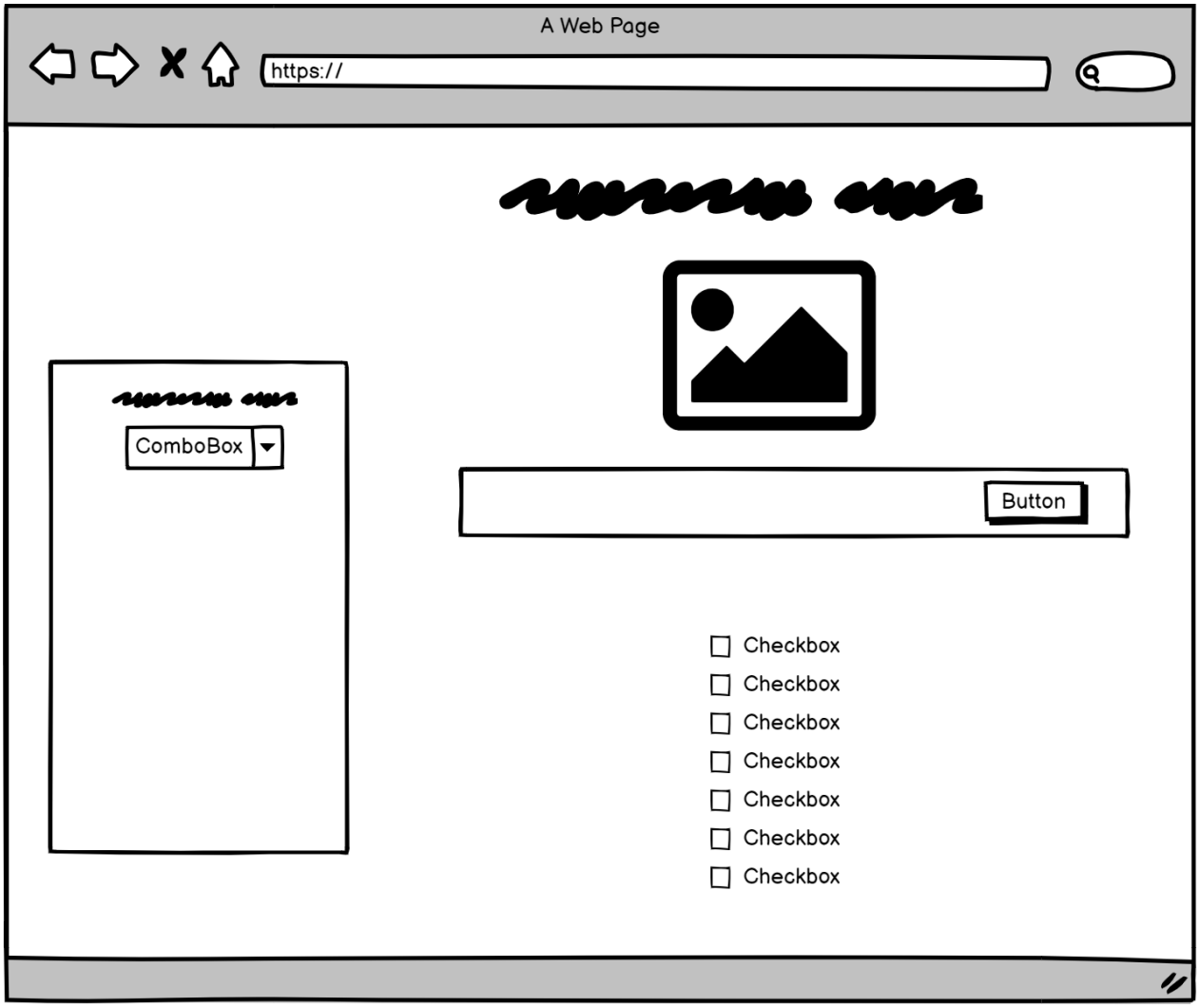


Figure 205 Wireframe of Anomaly Detection UI 1

7.10.3.5 ANOMALY DETECTION UI ANALYSIS AND VISUALIZATION PAGE

The analysis and visualization page should be available from the side bar which consists of drop down menu in the home page. The page is divided into three sections which consists of sidebar, browse file option and graphical interpretation check box options. Thereby, the boxes named heatmap and pairplot helps in visualizations.

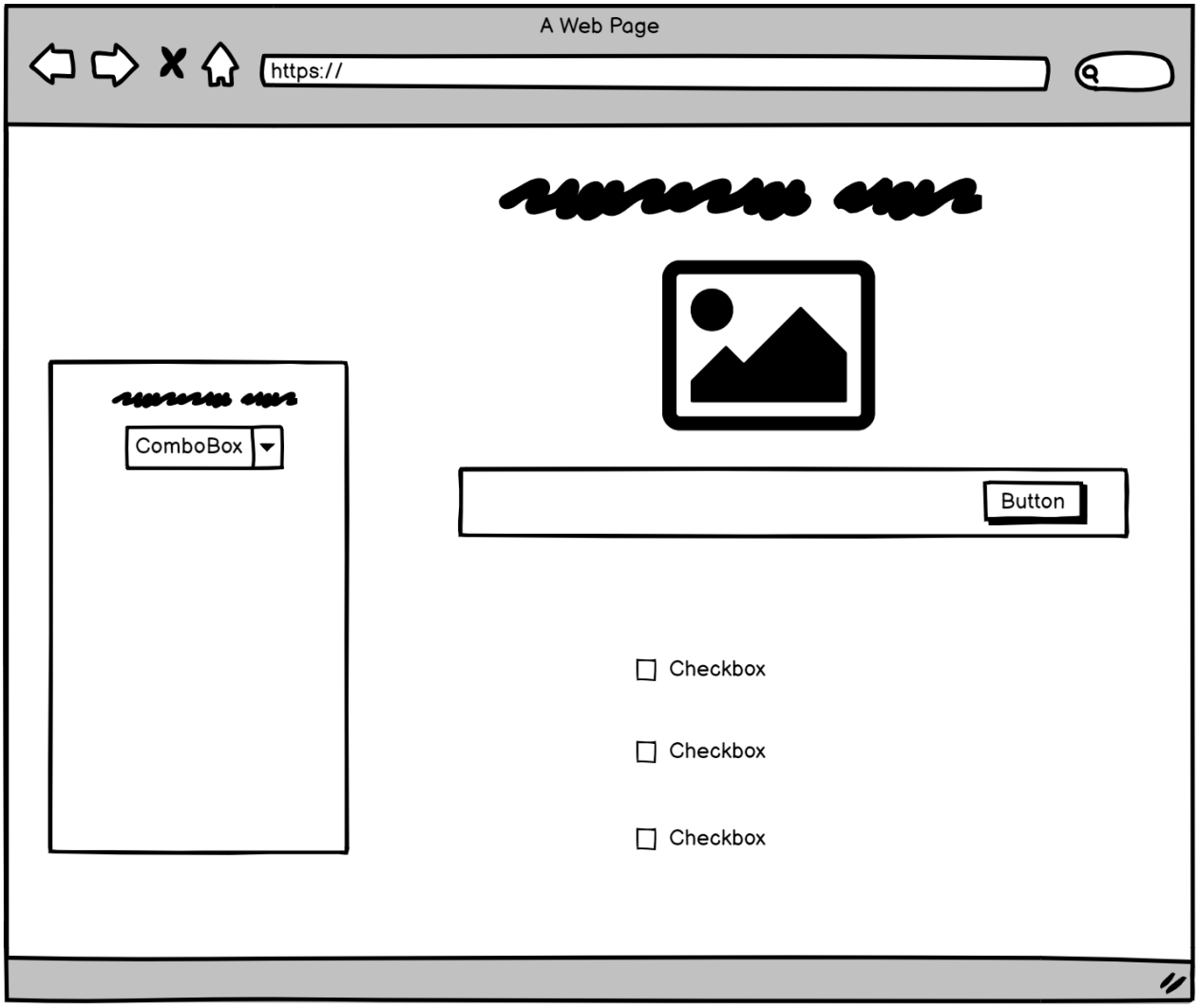


Figure 206 Wireframe of Anomaly Detection UI 2

7.10.3.6 ANOMALY DETECTION UI CLASSIFICATION PAGE

The classification page should be available from the side bar which consists of drop down menu in the home page. The page is divided into four sections which consists of sidebar containing two sub-options to select the model and another for generating random seeds, browse file option, target column and prediction results.

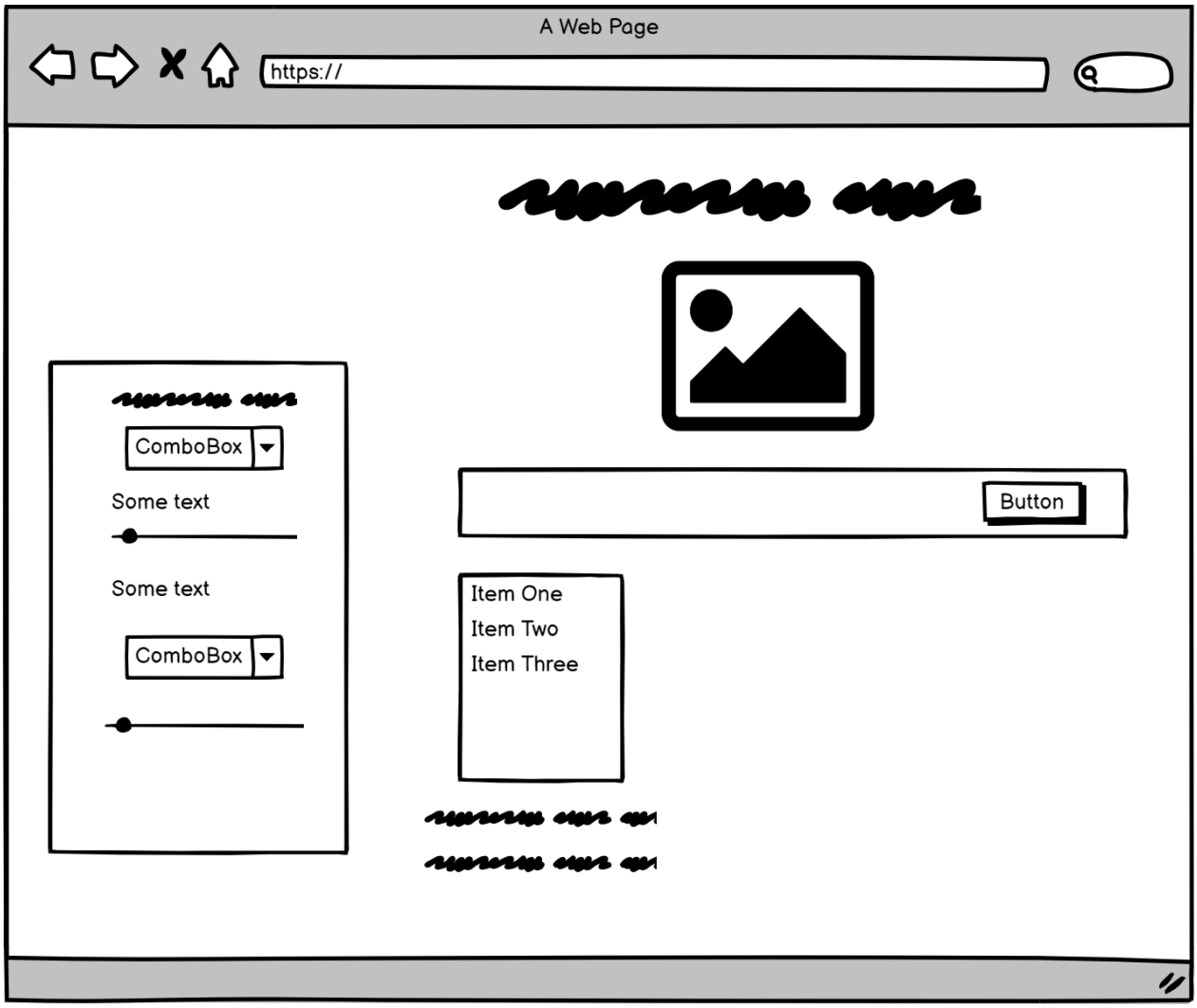


Figure 207 Wireframe of Anomaly Detection 3

7.21 APPENDIX U: SCREENSHOTS OF UI

7.21.1 HONEYPOT UI HOMEPAGE

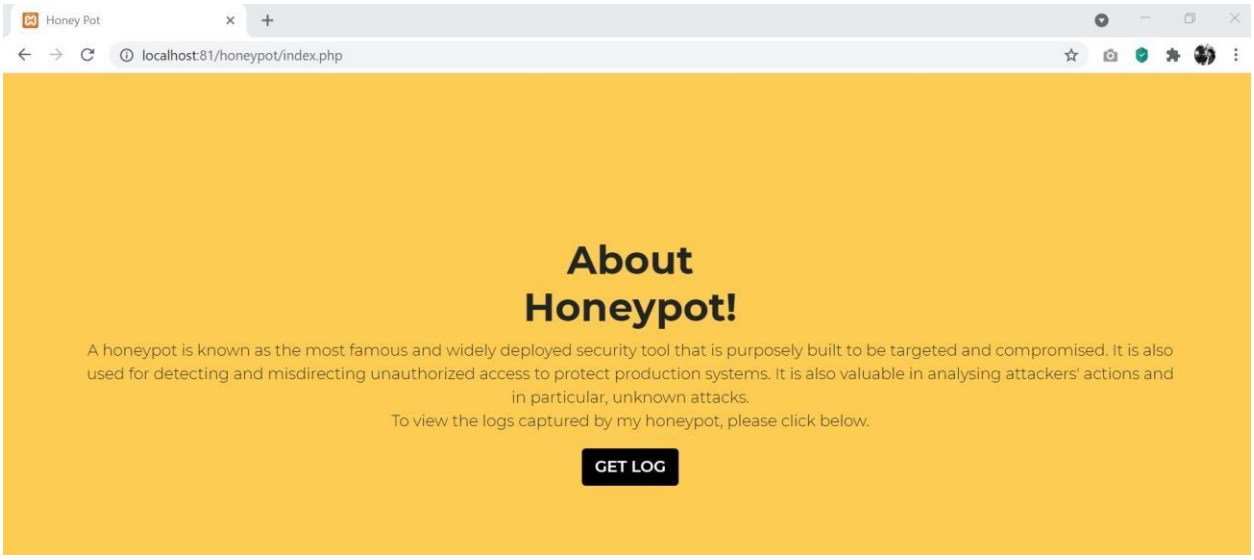


Figure 208 Honeypot UI Homepage

7.21.2 HONEYPOT UI WHITELISTED PAGE

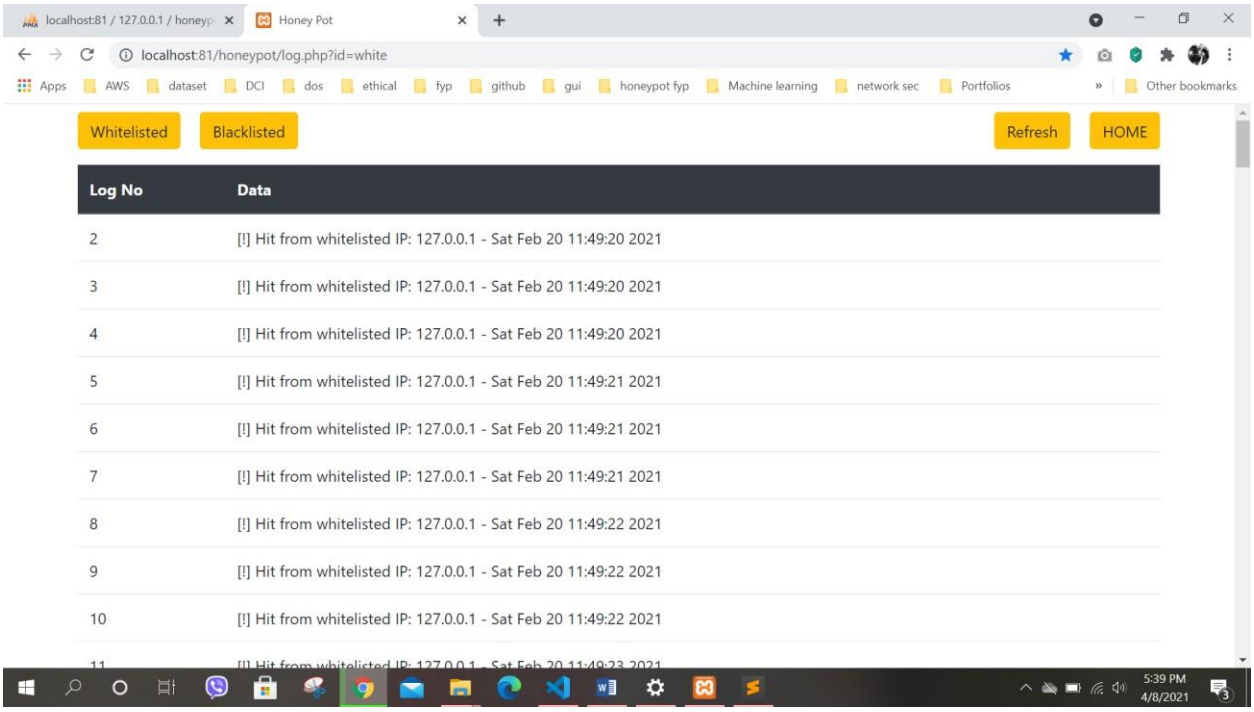


Figure 209 Honeypot UI Whitelisted page

7.21.3 HONEYPOT UI BLACKLISTED PAGE

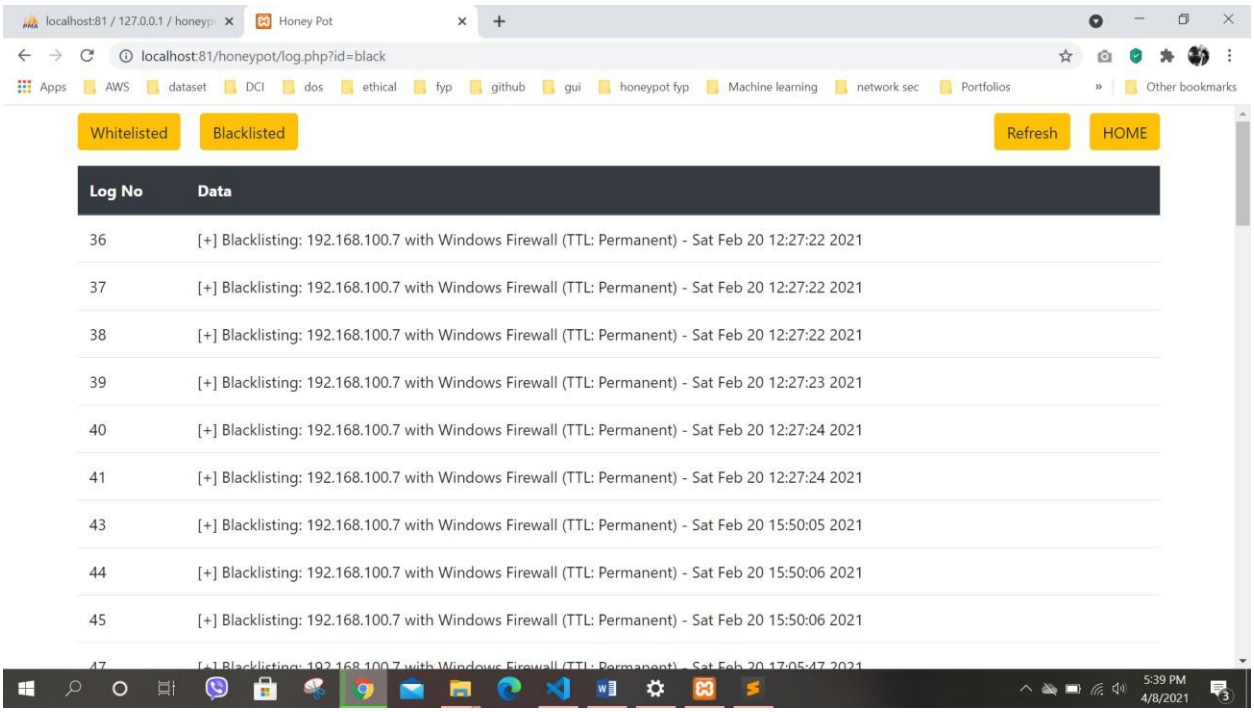


Figure 210 Honeypot UI Blacklisted page

7.21.4 ANOMALY DETECTION UI HOMEPAGE

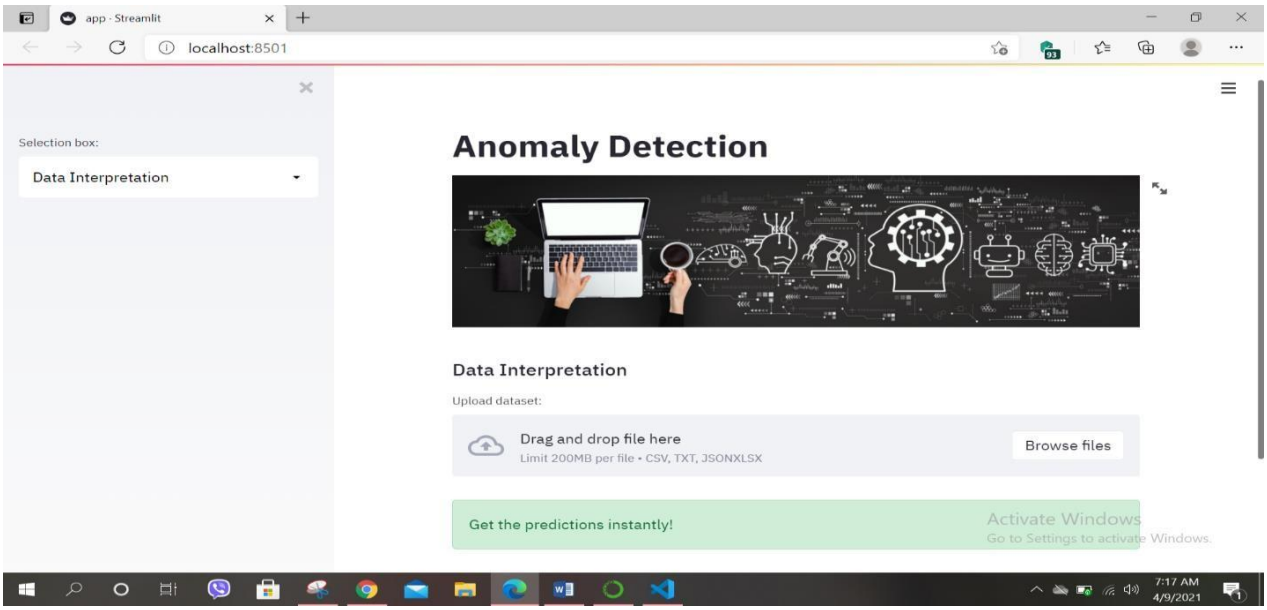


Figure 211 Anomaly Detection UI

7.21.5 ANOMALY DETECTION UI ANALYSIS AND VISUALIZATION PAGE

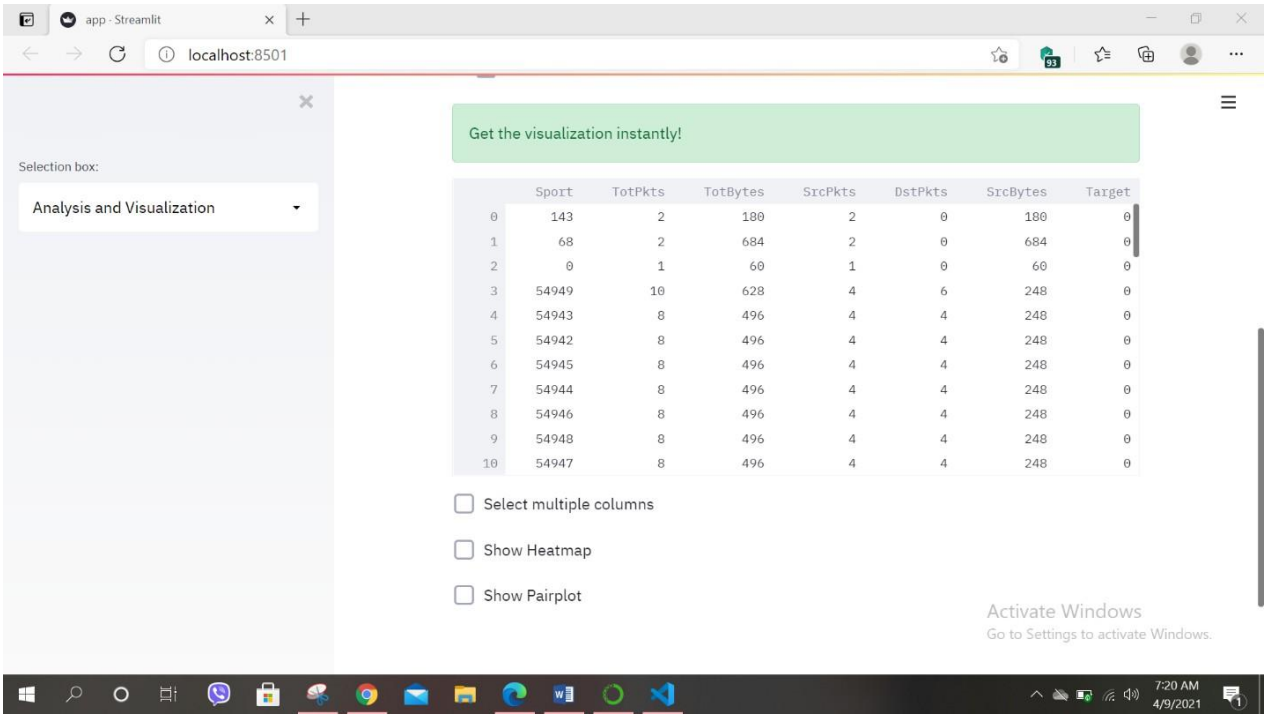


Figure 212 Anomaly Detection UI Analysis and Visualization

7.21.6 ANOMALY DETECTION UI CLASSIFICATION PAGE

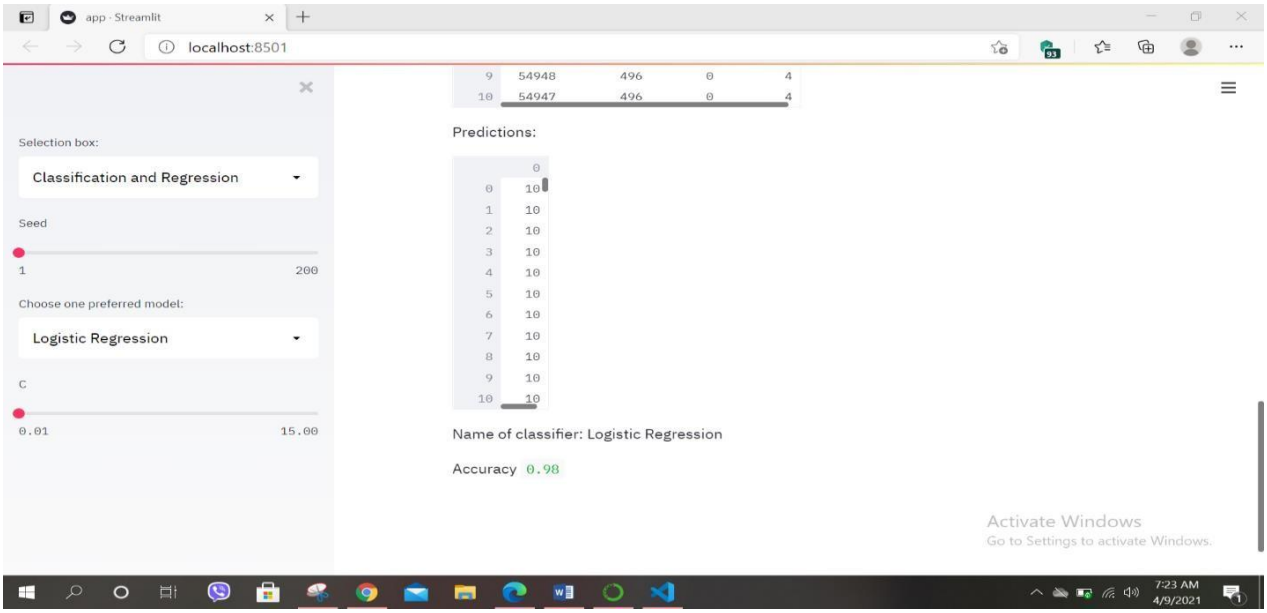


Figure 213 Anomaly Detection UI Classification

7.22 APPENDIX V: USER FEEDBACK

7.22.1 USER FEEDBACK FORM

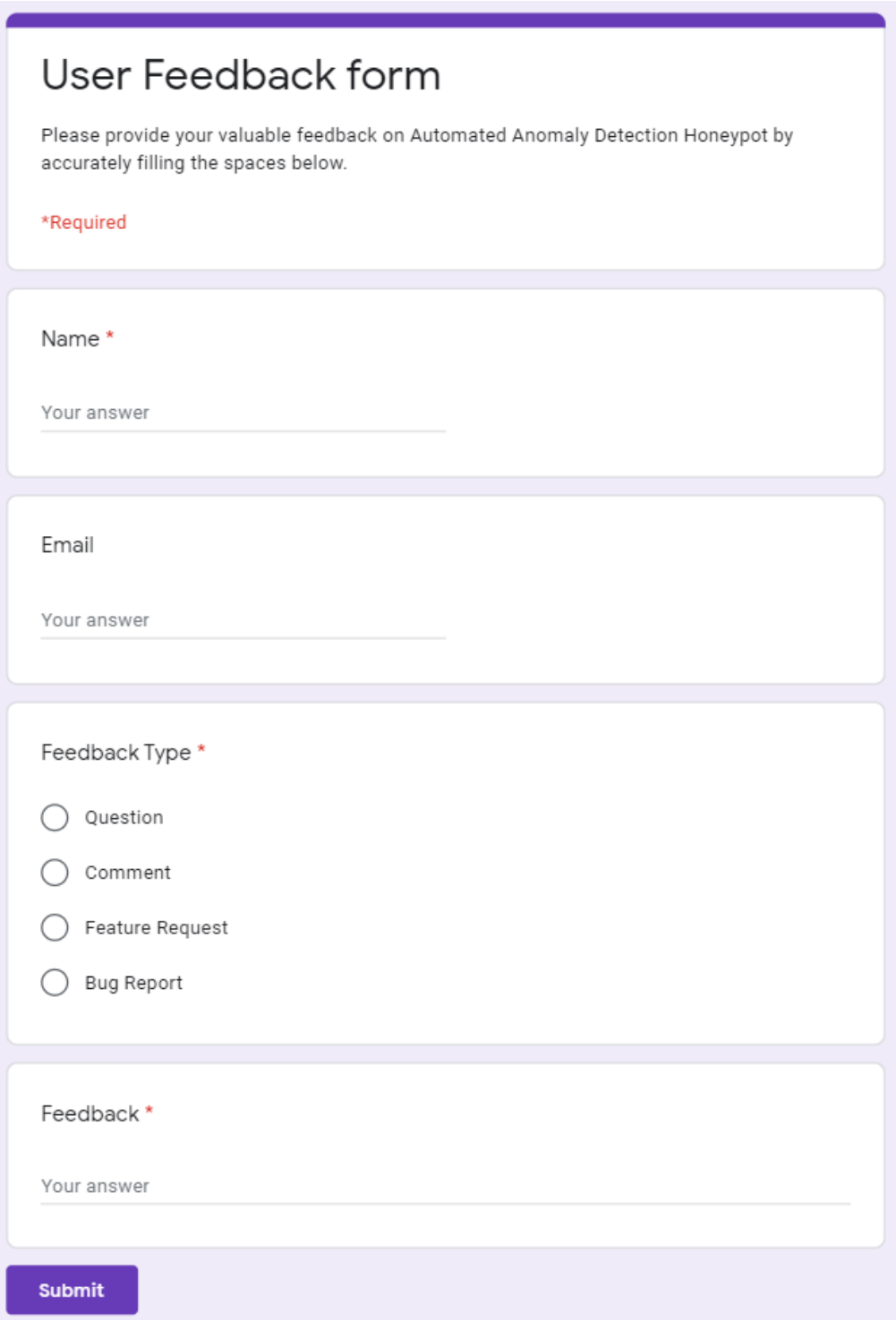
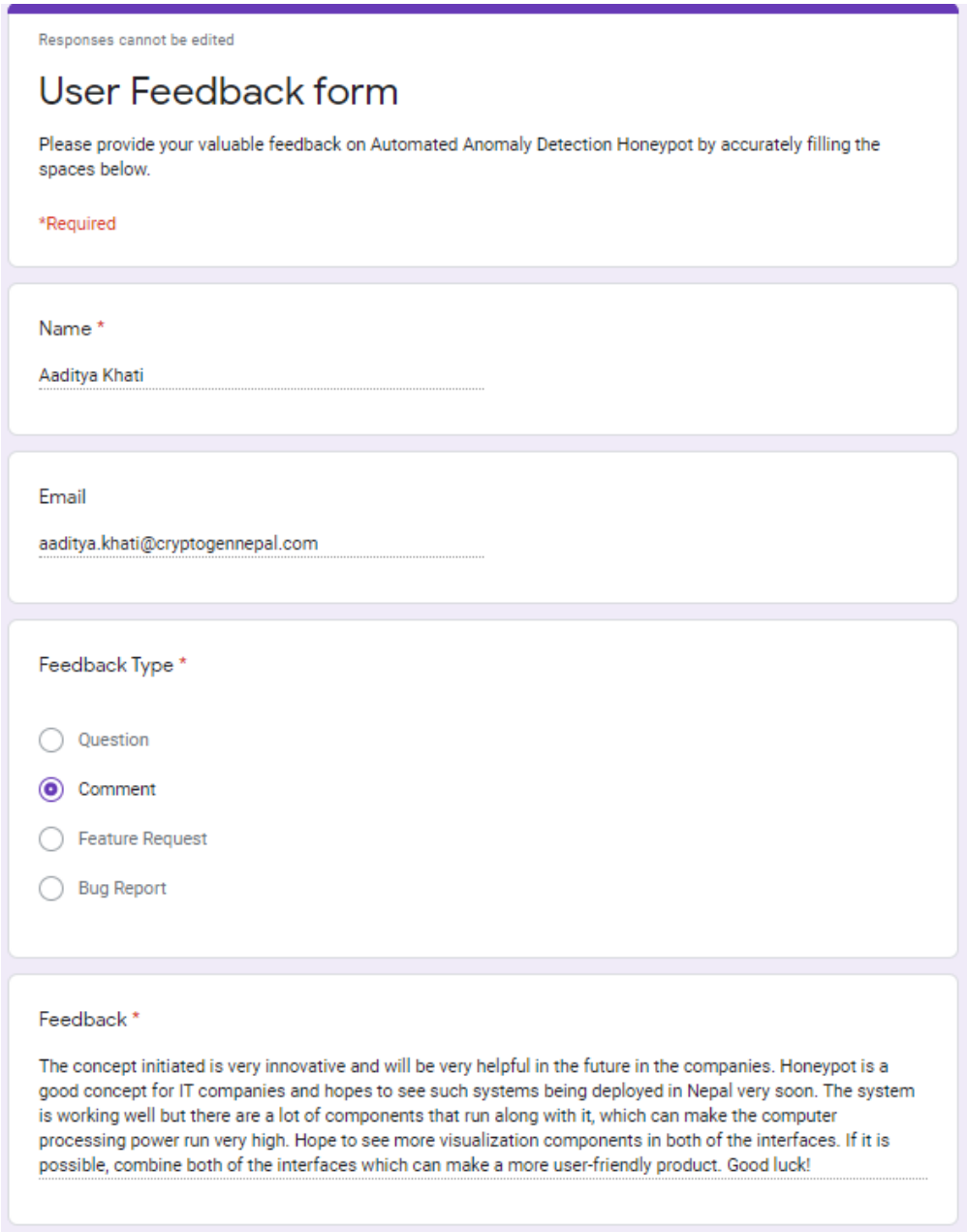


Figure 214 User Feedback Form 1

7.22.2 SAMPLE OF FILLED USER FEEDBACK FORMS



Responses cannot be edited

User Feedback form

Please provide your valuable feedback on Automated Anomaly Detection Honeypot by accurately filling the spaces below.

*Required

Name *

Aaditya Khati

Email

aaditya.khati@cryptogennepal.com

Feedback Type *

- ○ Question
- ◉ Comment
- ○ Feature Request
- ○ Bug Report

Feedback *

The concept initiated is very innovative and will be very helpful in the future in the companies. Honeypot is a good concept for IT companies and hopes to see such systems being deployed in Nepal very soon. The system is working well but there are a lot of components that run along with it, which can make the computer processing power run very high. Hope to see more visualization components in both of the interfaces. If it is possible, combine both of the interfaces which can make a more user-friendly product. Good luck!

Figure 215 User Feedback Form 2

7.23 APPENDIX W: PROGRESS REVIEW TABLE

Table 39 Progress Table

S.N	Activity	Status	Progress (%)	Completed Duration
1	**Initiation**			
	Gather Basic Idea about the project	**Completed**	**100%**	**Early**
	Research related papers in the field	**Completed**	**100%**	**On-Time**
	Develop concept of Operation	**Completed**	**100%**	**Early**
	Learn Python	**Completed**	**100%**	**Early**
	Learn Machine Learning	**Completed**	**100%**	**On-Time**
	Gather required collection and analysis	**Completed**	**100%**	**Early**
	Risk analysis	**Completed**	**100%**	**Early**
	Feasibility Study	**Completed**	**100%**	**Early**
	Pre-Survey	**Completed**	**100%**	**On-Time**
2	**Proposal**			
	Topic Introduction	**Completed**	**100%**	**Early**
	Problem Domain	**Completed**	**100%**	**Early**
	Project as a solution	**Completed**	**100%**	**Early**
	Project Deliveries and Outcomes	**Completed**	**100%**	**Early**
	Resource Requirements	**Completed**	**100%**	**Early**
	Risk and Contingency planning	**Completed**	**100%**	**Early**
	Methodology	**Completed**	**100%**	**Early**
	Summarizing the concept	**Completed**	**100%**	**Early**
3	**Development**			
	Install and setup tools and libraries	**Completed**	**100%**	**Early**
	Build Honeypot Server	**Completed**	**100%**	**Early**
	Develop log file Module	**Completed**	**100%**	**Early**
	Upload Logs in Database	**Completed**	**100%**	**On-Time**
	Build Honeypot Web Interface	**Completed**	**100%**	**On-Time**
	Collect Dataset	**Completed**	**100%**	**On-Time**
	Clean Dataset	**Completed**	**100%**	**On-Time**
	Split the data into train/test sets	**Completed**	**100%**	**On-Time**
	Make Predictions	**Completed**	**100%**	**On-Time**
	Build Anomaly Detection Interface	**Completed**	**100%**	**On-Time**
	Evaluate and Improve	**Completed**	**100%**	**On-Time**
4	**Interim Report**			
	Project Scenario	**Completed**	**100%**	**Early**
	Problem Statement	**Completed**	**100%**	**Early**
	Review similar projects	**Completed**	**100%**	**Early**
	Pre-Survey Analysis	**Completed**	**100%**	**On-Time**
	Analysis of project progress	**Completed**	**100%**	**On-Time**

	Further work	**Completed**	**100%**	**On-Time**
5	**Testing**			
	Unit Testing	**Completed**	**100%**	**On-Time**
	System Testing	**Completed**	**100%**	**On-Time**
6	**Complete and Release**			
	Review the overall project	**Completed**	**100%**	**On-Time**
	Post Survey	**Completed**	**100%**	**On-Time**
	Launch the project	**Completed**	**100%**	**On-Time**
7	**Final Report**	**Completed**	**100%**	**On-Time**

7.24 APPENDIX X: FUTURE WORK

In the present, the system is targeted at enthusiasts in cybersecurity and students and researchers mostly, but the future expansion sets to target organizations and use it in a commercial form. Honeypots are beneficial to almost all businesses. The only businesses that can't, practically, are those with impenetrable cybersecurity that ensures attackers can't get beyond the protective perimeter. A handful, if any, organizations worldwide have such precise security, so honeypots should be used by everyone (Turpitka, 2020). Therefore, this system will be targeted to most organizations and will be developed as strongly and effectively as possible.

- A proposed system by Anagnostakis (Matthew L. Bringer, 2012) , will be an inspired reference for future work in this system. According to the design, an integrated honeypot will keep track of how each request behaves with the help of machine learning. If one is found to be malicious, all of the attackers' actions will be reversed. Each incoming request is primarily handled by an anomaly detection system, which is set up with a high number of false positives on purpose. The harmful requests will be redirected to the admin while the benign requests will be redirected directly to the server.
- The next step in protecting and limiting access to private networks would be to integrate Cloud Firewall. It will work with access control providers to control filtering tools easily (Barracuda, 2021). The system can also be configured to send SMS with help of API. This will help the system admin to receive text alerts regarding any intrusion in the system.
- This study used a dataset made up of files containing extracted features from the network as training and test data. Unsurprisingly, this approach is difficult to put into practice in real life. However, by adding a module to the honeypot and making it function with the machine learning algorithm, this problem can be solved.
- In the future, attempts may be made to build a complete GUI-based system along with additional algorithms and methods in monitoring, protocol -study, and outlier detection in flow content, and others, so that security administrators can perform even more precisely and extract signatures.
- The primary issue that was understood during the development of this project was extracting anomalies from large volumes of data collected by honeypots which remains time-consuming and tedious or difficult, and the second issue is that data analysis necessitates specialized knowledge and expertise. Therefore, a web interface is

planned to be developed where it shows the statistics and the information of the number of attacks recorded in a certain period and from which regions. Along with that used a live traffic capture module, interactive display tool, and modulation by a visual object for customization options, visual correlation, and packet description, IP address, and protocol allocations (Matthew L. Bringer, 2012).

- The honeypot's logs could grow large over time, or older entries could be overridden, removed, or lost. As a result, the system will introduce log optimization with the aid of an automated log archive feature in the future. The log files will be encrypted using complex algorithms such as AES to protect the data extracted from the honeypot from attackers. Fingerprinting is one of the limitations of this current project.
- The system will be made as an anti-fingerprinting technology so, the attackers cannot detect the honeypot (R N Dahbul, 2016).
- Implementing cuckoo sandboxing for dynamic analysis, gathering sandboxing logs, and generating reports for each malware or attack are the works that are planned to be completed as part of the project.
- A further area which will certainly be strengthened in the near term is the application of honeytokens.

7.24.1 READINGS FOR FUTURE WORK

- SMS Module Integration

  Available from: https://www.twilio.com/docs/sms/quickstart/python
- Traffic differentiator module

  http://www.mecs-press.org/ijcnis/ijcnis-v4-n10/IJCNIS-V4-N10-7.pdf
- Cloud Firewall Integration

  Some preferred cloud firewall to implement are:
  - https://www.netgate.com/solutions/pfsense-plus/
  - https://www.checkpoint.com/cyber-hub/cloud-security/what-is-cloud-firewall/
- Visualization of honeypot data

  https://medium.com/@galolbardes/learn-how-to-deploy-a-honeypot-and-visualise-its-data-step-by-step-ea3cd3f25822
- Log file encryption integration

  https://github.com/ArtisanCode/Log4NetMessageEncryptor
- Honeypot Threat Map in Real-Time

  https://medium.com/leaseweb/real-time-honeypot-attack-map-ce2b48cf1cd5