A Case Study Report

on

# INTRUSION DETECTION SYSTEM USING MACHINE LEARNING TECHNIQUES

Submitted in partial fulfillment of the requirements for the award of the degree

**BACHELOR OF ENGINEERING**
**in**
**COMPUTER SCIENCE AND ENGINEERING**
**by**

| Student Name | Roll Number | Mentor Name |
|---|---|---|
| Charan Teja | 160118733027 | Smt.G. Shanmukhi Rama |
| SaiKumar Kaleru | 160118733045 | Dr. Sangeeta Gupta |
| Yashwanth Javvaji | 160118733060 | Dr. Sangeeta Gupta |

**Department of Computer Science and Engineering,**
**Chaitanya Bharathi Institute of Technology**
**(Affiliated to Osmania University, Hyderabad)**
**Hyderabad, TELANGANA (INDIA) – 500 075**
**MAY-2021**

**Department of Computer Science and Engineering,**
**Chaitanya Bharathi Institute of Technology**
**(Affiliated to Osmania University, Hyderabad)**
**Hyderabad, TELANGANA (INDIA) – 500 075**

# CERTIFICATE

This is to certify that the project titled **“INTRUSION DETECTION SYSTEM USING MACHINE LEARNING TECHNIQUES”** is the bonafide work carried out by **Charan Teja (160118733027), Saikumar Kaleru (160118733045), Yashwanth Javvaji (160118733060)** students of B.E.(CSE) of Chaitanya Bharathi Institute of Technology, Hyderabad, affiliated to Osmania University, Hyderabad, Telangana(India) during the academic year 2020-2021, submitted in partial fulfillment of the requirements for the award of the degree in **Bachelor of Engineering** (**Computer Science and Engineering** ) and that the project has not formed the basis for the award previously of any other degree, diploma, fellowship or any other similar title.

| | |
|---|---|
| **Supervisors** | **Head, CSE Dept.** |
| Dr. Sangeeta Gupta, Associate Professor | Dr. Y Ramadevi |
| Smt.G. Shanmukhi Rama, Assistant Professor | Dept of CSE, CBIT |
| Mrs T Suvarnakumari, Assistant Professor | |

**Place: Hyderabad**
**Date: 28-05-2021**

**External Examiner**

# DECLARATION

I/we hereby declare that the project entitled "Disease Prediction system" submitted for the B. E (CSE) degree is our original work and the project has not formed the basis for the award of any other degree, diploma, fellowship, or any other similar titles.

**Name(s) and Signature(s) of the Student**

Charan Teja (160118733027)

S. Charan

Saikumar Kaleru (160118733045)

K. Saikumar

Yashwanth Javvaji (160118733060)

J Yashwanth

**Place: Hyderabad**

**Date: 28-05-2021**

# ABSTRACT

Security of data in a network based computer system has become a major challenge in the world today. For this purpose, intrusion detection systems (IDS) are used to monitor the threats encountered on the network, by detecting any change in the normal profile. Intrusion Detection System is a software application to detect network intrusion using various machine learning algorithms.IDS monitors a network or system for malicious activity and protects a computer network from unauthorized access from users.. The intrusion detector learning task is to build a predictive model (i.e. a classifier) capable of distinguishing between ‘bad connections’ (intrusion/attacks) and a ‘good (normal) connections’..The idea here is to use classification algorithms for analyzing KDD’99 datasets, with 41 Attributes (features). Based on these 41 attributes, the KDD’99 Datasets has been classified into five different types of attacks, i.e. normal, Probe, U2R, R2L and DOS.

# ACKNOWLEDGEMENT

It is our privilege to acknowledge with a deep sense of gratitude and devotion for keen personal interest and invaluable guidance rendered by our Project mentors Dr. Sangeeta Gupta, Smt.G. Shanmukhi Rama, and Mrs. T Suvarnakumari. Our respect and regards to Dr. Y Rama Devi, Professor, Department of Computer Science and Technology, Chaitanya Bharathi Institute of Technology, for her invaluable suggestions that helped us in the successful completion of the project. We are grateful to our Principal Dr. P. Saradhi Varma, Chaitanya Bharathi Institute of Technology, for his cooperation and encouragement.

Finally, we also thank all the staff members, faculty of Dept. of CSE, CBIT, our friends, and all our family members who, with their valuable suggestions and support, directly or indirectly helped us in completing this project work.

# TABLE OF CONTENTS

# 1 INTRODUCTION

An Intrusion Detection System is used to detect all types of malicious network traffic and computer usage that can't be detected by a conventional firewall. This includes network attacks against vulnerable services, data driven attacks on applications, host based attacks such as privilege escalation, unauthorized logins and access to sensitive files, and malware (viruses, trojan horses, and worms).Today, political and commercial entities are increasingly engaging in sophisticated cyber-warfare to damage, disrupt, or censor information content in computer networks. In designing network protocols, there is a need to ensure reliability against intrusions of powerful attackers that can even control a fraction of parties in the network. The controlled parties can launch both passive (e.g., eavesdropping, nonparticipation) and active attacks (e.g., jamming, message dropping, corruption, and forging). Intrusion detection is the process of dynamically monitoring events occurring in a computer system or network, analyzing them for signs of possible incidents and often interdicting unauthorized access . This is typically accomplished by automatically collecting information from a variety of systems and network sources, and then analyzing the information for possible security problems.The Project concept is to achieve the new method for extracting the information from the KDD Cup Dataset that will help to automatically detect the attack like Dos. How such attacks are Detected by ANN module and provide the security from any anomaly data which will slow your system.and provide security to the serve

# 2 BACKGROUND INFORMATION

There are many types of research introduced for intrusion detection systems. With the emergence of Big Data, the traditional techniques become more complex to deal with Big Data. Therefore, many researchers intend to use Big Data techniques to produce high speed and accurate intrusion detection systems. In this section, we show some researchers that used machine learning Big Data techniques for intrusion detection to deal with Big Data. Ferhat et al. used cluster machine learning techniques. The authors used the k-Means method in the machine learning libraries on Spark to determine whether the network traffic is an attack or a normal one. In the proposed method, the KDD Cup 1999 is used for training and testing. In this proposed method the authors didn't use feature selection technique to select the related features. Peng et al. proposed a clustering method for IDS based on Mini Batch K-means combined with principal component analysis (PCA). The principal component analysis method is used to reduce the dimension of the processed dataset and then the mini batch K-means++ method is used for data clustering. Full KDDCup1999 dataset has been used to test the proposed model.

## Deployment method based IDS

From the deployment-based IDS perspective, IDS is further subclassified as host-based-IDS (HIDS) or NIDS. HIDS is deployed on the single information host. Its task is to monitor all the activities on this single host and scans for its security policy violations and suspicious activities. The main drawback is its deployment on all the hosts that require intrusion protection, which results in extra-processing overhead for each node and ultimately degrades the performance of the IDS. In contrast, NIDS is deployed on the network with the aim to protect all devices and the entire network from intrusions. The NIDS will constantly monitor the network traffic and scans for potential security breaches and violations. This article focuses on the different methods used in the NIDS.

## Detection method based IDS

From the detection IDS perspective, the IDS is further subdivided into "Signature-based intrusion detection (SIDS)" and "Anomaly detection-based intrusion detection (AIDS)". SIDS, also known as the "misuse intrusion detection" or "knowledge-based intrusion detection," is based on the idea of defining a signature for attack patterns. These signatures are stored in the signature database and the data patterns are matched with these stored signatures for attack detection. The advantage includes the high detection efficiency for the known attacks due to the availability of signatures for those attacks. On the other hand, this method lacks the ability to detect novel and new attacks due to the absence of signature patterns. Also, a huge signature database is maintained and compared with the data packets for possible intrusions, which makes it a resource-consuming approach. AIDS, also called the "behavior-based IDS," is based on the idea of clearly defining a profile for normal activity. Any deviation from this normal profile will be considered as an anomaly or abnormal behavior. The major advantages of AIDS are its ability to detect unknown and new attacks and the customized nature of the normal activity profile for different networks and applications. However, the main drawback is the high FAR as it is difficult to find the boundary between the normal and abnormal profiles for intrusion detection. The popularity of the IoT paradigm due to network technologies advancement has resulted in exponential growth in the use of IoT devices. One of the vital technologies used in the development of an IoT network is the WSN, which comprises a collection of sensor nodes for information collection. A huge amount of critical information is collected by these IoT sensor devices and is shared over the internet.

# 3 SCOPE OF THE CASE STUDY

## 3.1 METHODOLOGY OF RESEARCH

This section deals with the attempts made by researchers in the area of network based intrusion detection systems and most of the detection works were based on KDD dataset. An expert system based on rules and statistical approaches are the two commonly used approaches to ensure intrusion detection. The Expert system based on rules will detect the known intrusion at a high rate and it will not identify new intrusion. Where, the database should be continuously updated. In statistical approach, Intrusion Detection System includes different methods like Cluster analysis, Multivariate analysis, Bayesian analysis, and Principal component analysis. Many new techniques from data mining should be proposed to overcome the problems of above mentioned approaches.

Intrusion detection and prevention systems (IDPS) are security systems that are used to detect and prevent security threats to computer systems and computer networks. These systems are configured to detect and respond to security threats automatically thereby reducing the risk to monitored computers and networks. Intrusion detection and prevention systems use different methodologies such as signature based, anomaly based, stateful protocol analysis, and a hybrid system that combines some or all of the other systems to detect and respond to security threats. The growth of systems that use a combination of methods creates some confusion when trying to choose a methodology and system to deploy. This paper seeks to offer a clear explanation of each methodology and then offer a way to compare these methodologies.

## 3.2 SCOPE

The most important issues about our proposed architecture is the interaction between system-user and intrusion detection system, in order to verify predictions of the system. As a means to reduce the number of interactions, system updates in presence of the user could be done in a periodic manner or at specified times that the number of wrong predictions reaches a predefined threshold.This survey includes six important areas related to IDSs for IoT systems and networks:

(1) IoT architectures and technologies
(2) IoT threats and attack types
(3) IDS architectures and their design
(4) an explanation of ML and DL techniques applied in the design of IDSs
(5) a description of various datasets available to researchers for evaluation of their proposed IDS
(6) future research challenges and directions.

# 4 DESIGN AND IMPLEMENTATION

## 4.1 SOFTWARE REQUIREMENT SPECIFICATION

### 4.1.1 Introduction

#### 4.1.1.1 Purpose

The Project concept is to achieve the new method for extracting the information from the KDD Cup Dataset that will help to automatically detect the attack like Dos. How such attacks are Detected by ANN module and provide the security from any anomaly data which will slow your system.and provide security to the server.

#### 4.1.1.2 Scope

One of the most important issues about our proposed architecture is the interaction between system-user and intrusion detection system, in order to verify predictions of the system. As a means to reduce the number of interactions, system updates in presence of the user could be done in a periodic manner or at specified times that the number of wrong predictions reaches a predefined threshold.

#### 4.1.1.3 Definitions, acronyms, and abbreviations

KDD - Knowledge Discovery and Data Mining
IDS - Intrusion Detection System
An Intrusion Detection System (IDS) is a network security technology originally built for detecting vulnerability exploits against a target application or computer.

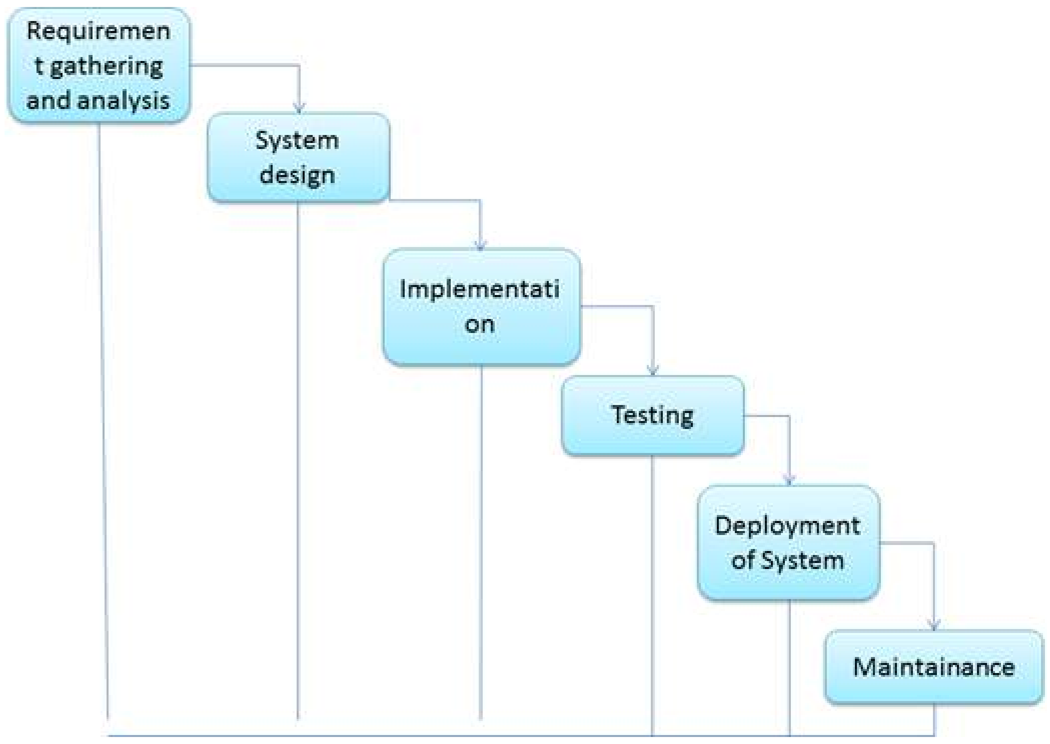
#### 4.1.1.4 References

1. https://journalofbigdata.springeropen.com/articles/10.1186/s40537-018-0145-4
2. https://ieeexplore.ieee.org/abstract/document/9215333
3. https://www.sciencedirect.com/science/article/pii/S1877050920311121
4. https://www.sciencedirect.com/science/article/abs/pii/S1353485820300568
5. https://medium.com/cuelogic-technologies/evaluation-of-machine-learning-algorithms-for-intrusion-detection-system-6854645f9211

#### 4.1.1.5 Applying Software Engineering Approach

**Software Development Model Used: Waterfall Model**

There are various software development approaches defined and designed which are employed during the development process of software, these approaches are also referred as Software Development Process Models? Each process model follows a particular life cycle in order to ensure success in the process of software development. One such approach used in software development is the waterfall model? It was the first process model to be introduced and followed widely in software engineering to ensure success of the project. In the waterfall approach, the whole process of software development is divided into separate process phases. The phases in the waterfall model are: Requirement specification phase, Software design, Implementation and maintenance. All these phases are cascaded to each other so that the second phase is started as and when a defined set of goals are achieved for the first phase. General overview of the waterfall model is as follows.



**Stages of Waterfall Model:**

1. **Requirements Gathering:** Requirements from customers are collected by communicating with customers.
2. **Planning and Analysis:** Analysis of gathered requirements is performed and planning and estimate of project cost and schedule is done.
3. **Modelling and Design:** Model and Design of system is created as per analysis of requirements.
4. **Implementation:** Actual system is implemented using 2 phases, coding and testing.
5. **Deployment and Feedback:** System is deployed on the user's machine and feedback is taken from the user.

### 4.1.2 General description

#### 4.1.2.1 Product perspective

This feature will give the user a secure and simple login screen.This means rather than creating try catches for a handful of error types, it just has only a handful of available and possible inputs, to prevent any improper logging in, which might cause unexpected errors, and therefore limiting the systems capabilities and also client attack on the server by sending multiple selecting multiple attributes from KDD Cup Dataset.

#### 4.1.2.2 Product functions

In this extraction framework, intermediate output of IDS is stored so that only the improved component has to be deployed to the entire database KDD Cup data set. Extraction is then performed on both the previously processed data from the unchanged components as well as the updated data generated by the improved component. Performing such an incremental extraction can result in a tremendous reduction of processing time. To realize this new information extraction framework, the project proposes to choose database management systems over file-based storage systems to address the dynamic extraction needs.
The proposed key phrase extraction method consists of four primary components: Document preprocessing, Candidate phrase identification, Information Extraction from Database Elements of the system with their functions as follow:

1. User management-username, password, add, update, login
2. Attack On Server by Providing query
3. Query suggestion-process query, map equivalent query
4. Checking source
5. Attack detection
6. Log generation-user records, result, add, update, search
7. Data management-attributes, user detail, add, search
8. Data extraction-query, search, extract
9. Request management-request accept, block, unblock

#### 4.1.2.3 User characteristics

User classes will be Database(KDD Cup dataset), Administrator, User, Server.

#### 4.1.2.4 General constraints

There are no constraints at this point in time.

#### 4.1.2.5 Assumptions and dependencies

We assume that extra documentation beyond this SRS would not be necessary in order for the user to utilize this product.

### 4.1.3 Specific requirements

#### 4.1.3.1 Functional requirements

#### 4.1.3.1.1 Introduction

The Project concept is to achieve the new method for extracting the information from the KDDCUP Dataset that will help to automatically detect the attack like Dos.How such attack are Detected by ANN module and provide the security from the any anomaly data which is slow your system.and provide security to the server.

#### 4.1.3.1.2 Input

KDDCUP dataset

#### 4.1.3.1.3 Processing

Processing the dataset and getting the ML model for analysing new data

#### 4.1.3.1.4 Output

Detecting intrusion ,validation ,logging.

### 4.1.3.2 External Interface Requirements

#### 4.1.3.2.1 User interface

The first interface is the log-in screen of the Application. This is where the user and Admin have a specific User-name and Password so that they can gain access to the database. Next is the Search Hints interface. Using this interface users can get hints for searching databases for particular domains. Also client attacks on the server by providing anomaly queries. Another is admin login to view all the logs of detected attacks.

#### 4.1.3.2.2 Hardware interface

Though not necessarily interfacing with the hardware, the system must make use with an internet connection.

#### 4.1.3.2.3 Software interface

Along with the internet connection, the system makes indirect use of an internet browser. KDD Cup 99 data set is a new Database.
Operating System: Windows XP/7/8/10

#### 4.1.3.2.4 Communication interfaces

The system uses an internet connection to connect to the database.

### 4.1.3.3 Performance Requirements

Considering our project is totally based on the client server architecture . so that the client and server should be client to serve the request as well as to send the request. Also as the number of clients are going to be larger than that indirectly or directly the server is overloaded .So that the server should serve all the requests coming from the clients. So the hardware or the software as the server must have the networking capability. The network architecture should be such that the request/response time is measured .So that the time between request and the response should be as minimum as possible. Also the network should be scalable so that the number of clients can be increased as needed.

### 4.1.3.4 Security Requirements

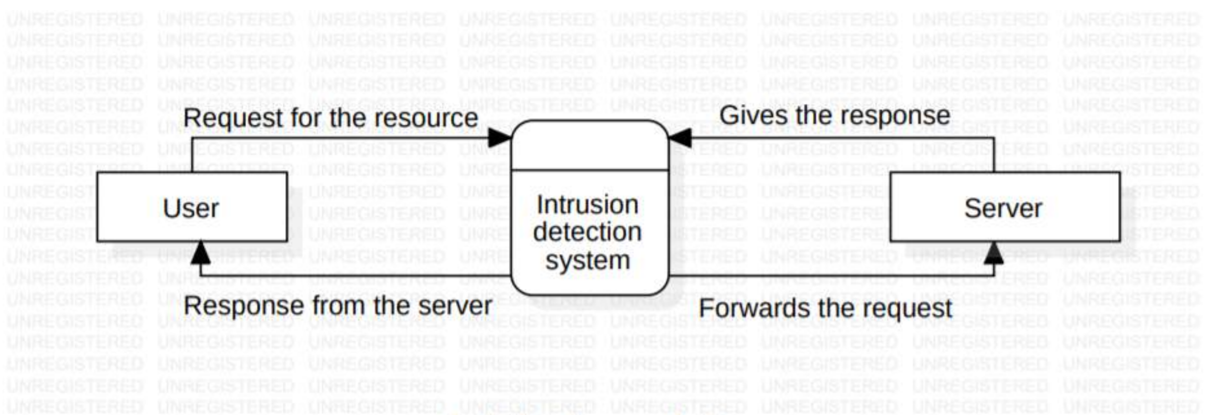
Access to the database should be restricted to people that are required to view information about users. Passwords and IDs should be regulated to be at least a certain length and must contain non-alphanumeric characters in both the password and ID. Access to the database should be restricted to people that are required to view information about users. Passwords and IDs should be regulated to be at least a certain length and must contain non-alphanumeric characters in both the password and ID. As we are giving the control of the whole system to the IDS and the server . So our overall data or the database could be totally accessed by the IDS or the system/server administrator. So any secret key and the other information about the server could not be told elsewhere. Any security system has limitations so that our IDS could not prevent them totally . Our software could not get full control of the system. So try to avoid the system calls. We will also try to implement the jre7 to take kernel level privileges to try to differentiate between the http,tcp and other types of packets

### 4.1.3.5 Safety requirements

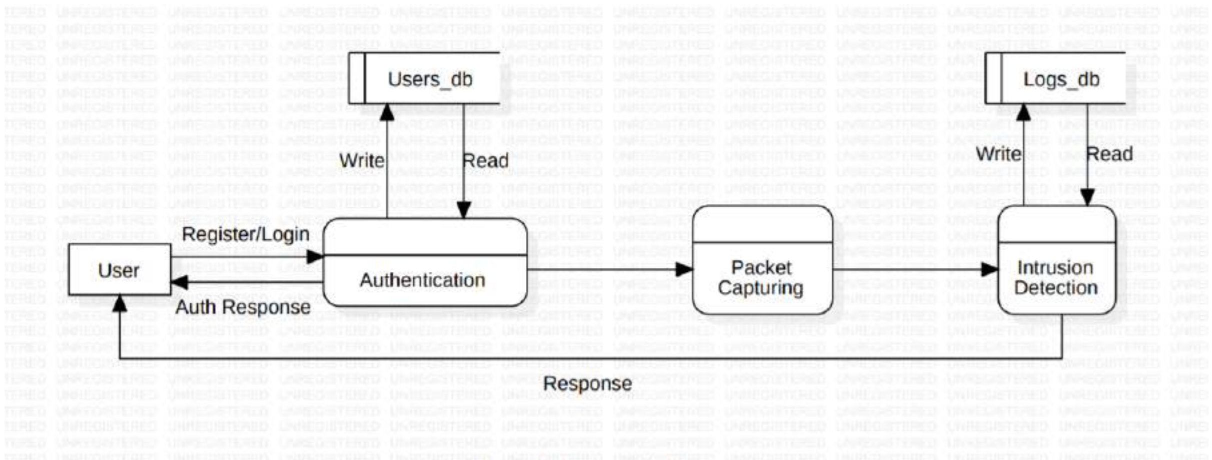
Other requirements should be the power supply should be uninterrupted. The networking devices should be properly connected . And faulty networking devices should be removed as early as possible such as the router ,switch and the hub etc.

## 4.2 UML DIAGRAMS

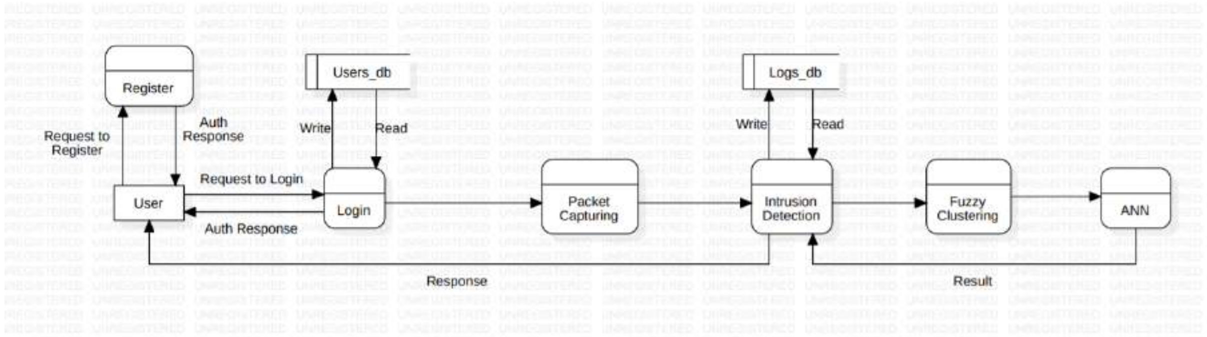
### 4.2.1 Data Flow Diagrams



DFD Level 0

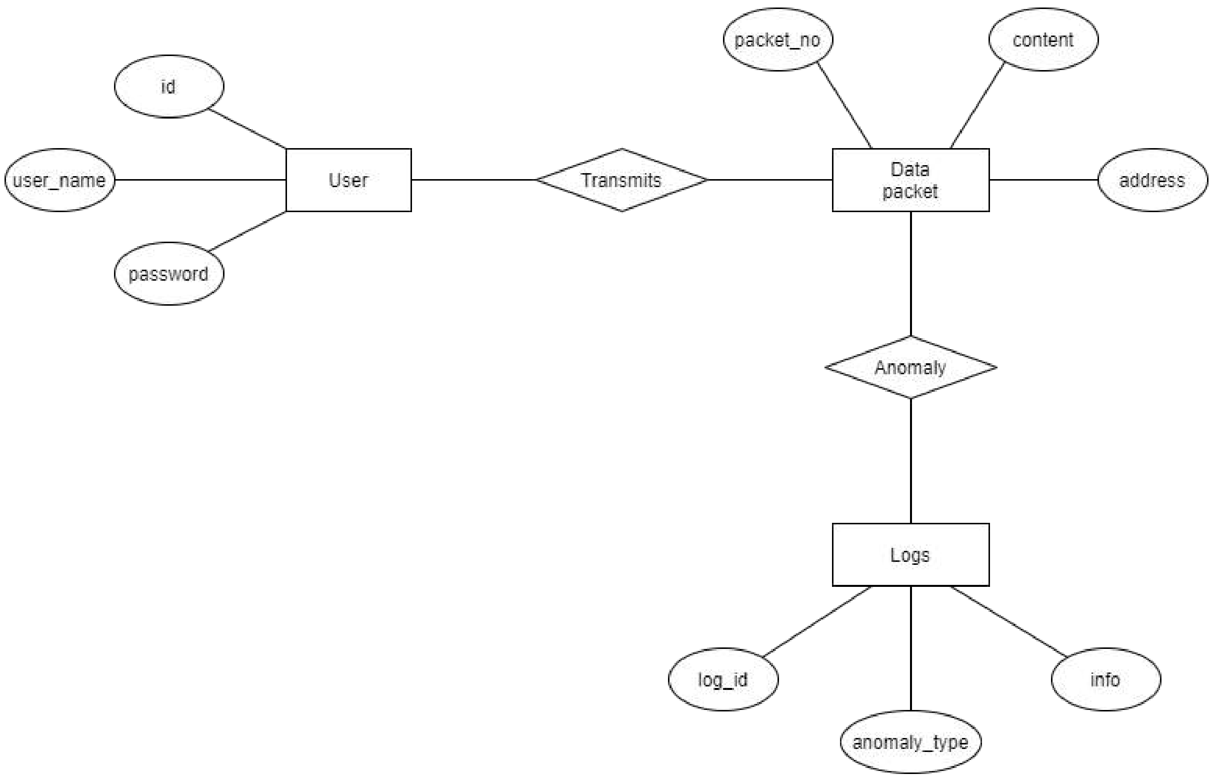


DFD Level 1

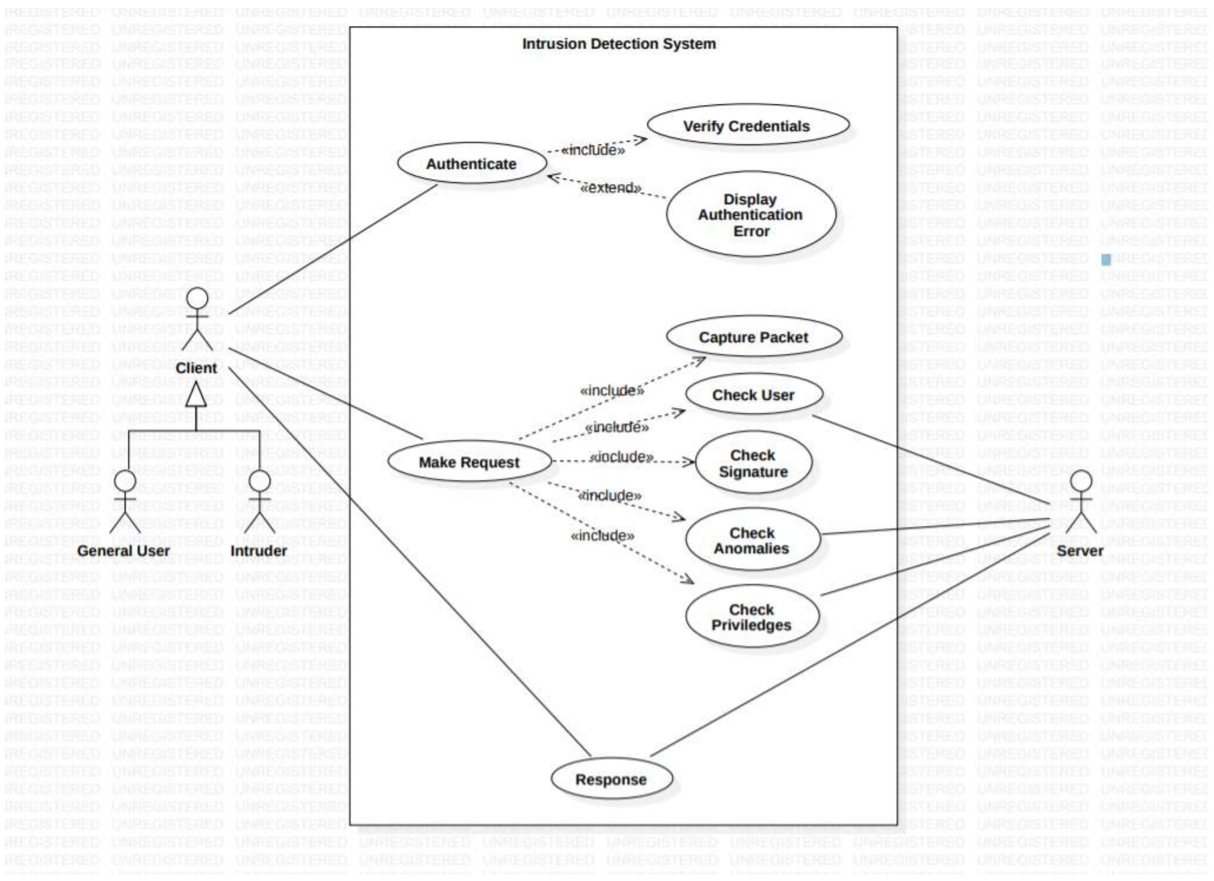


DFD Level 2

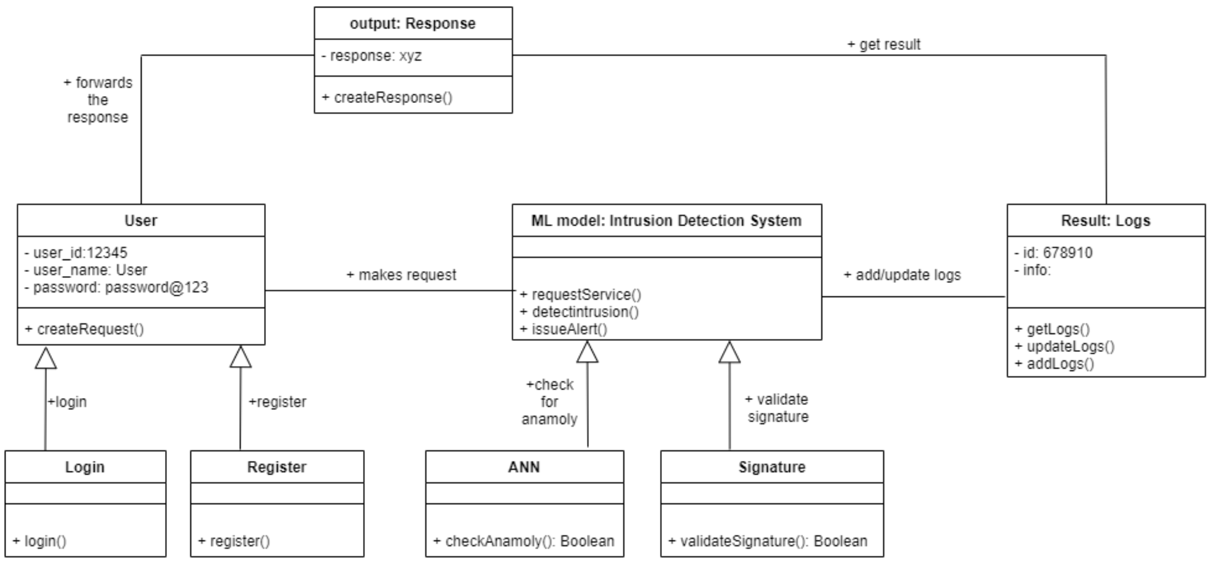
### 4.2.2 ER Diagram



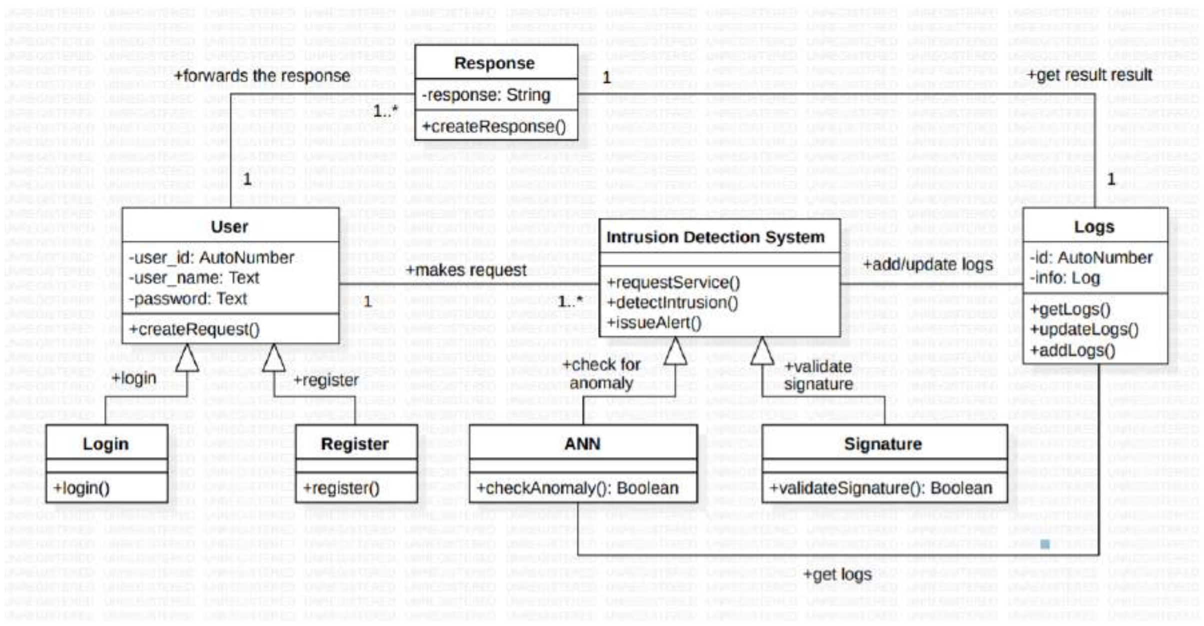
### 4.2.3 Use Case Diagram



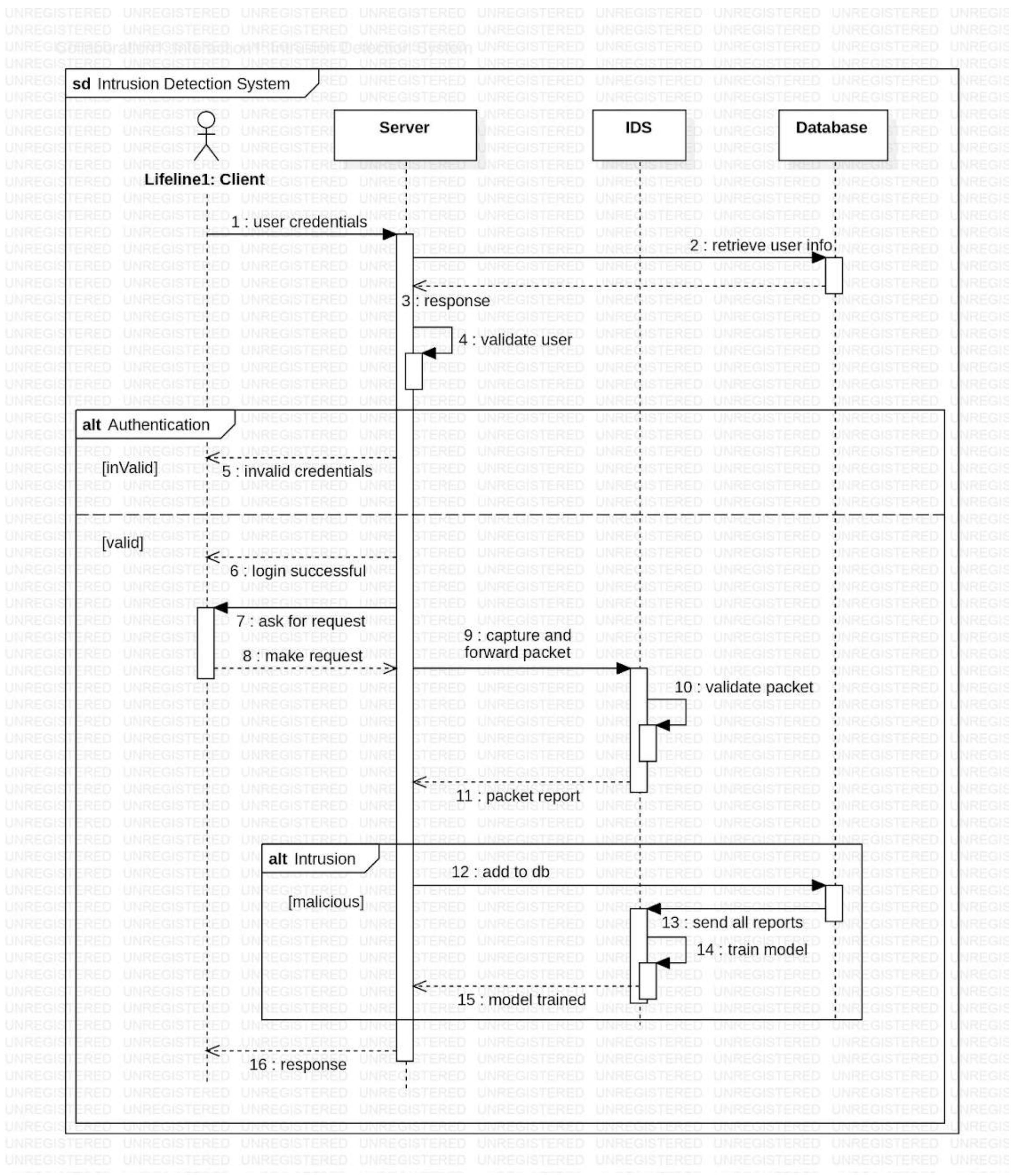
### 4.2.4 Object Diagram



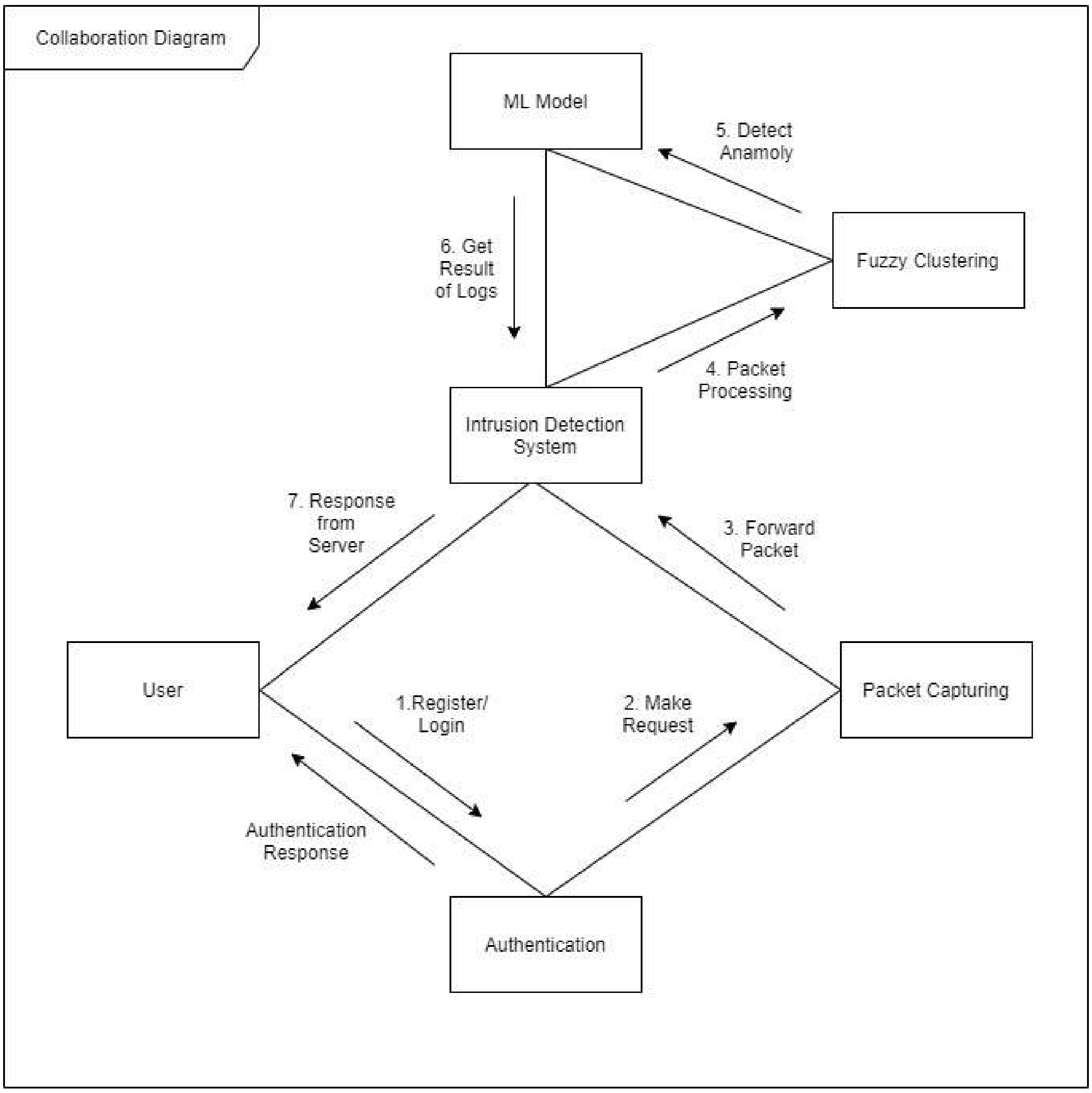
### 4.2.5 Class Diagram



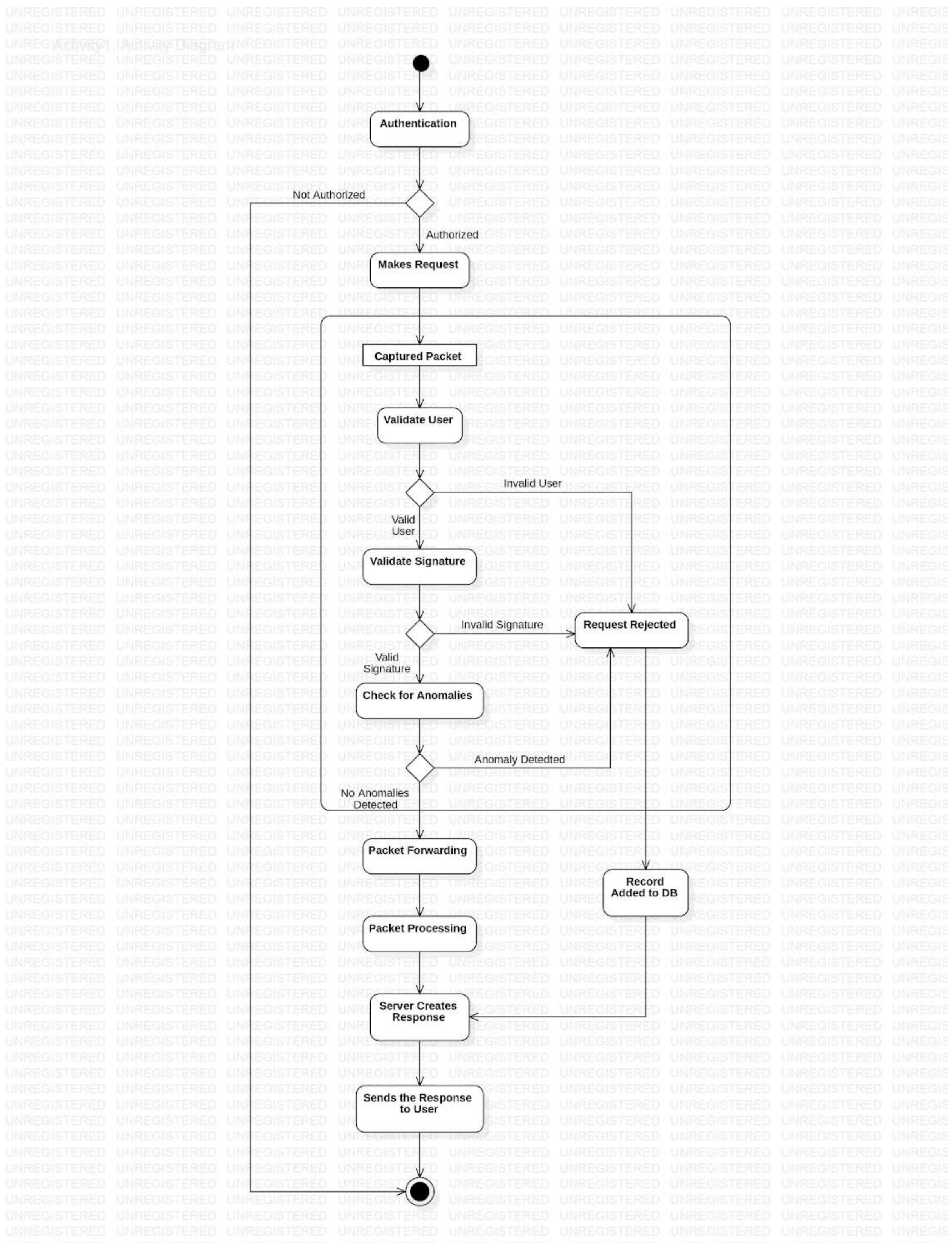
## 4.2.6 Sequence Diagram



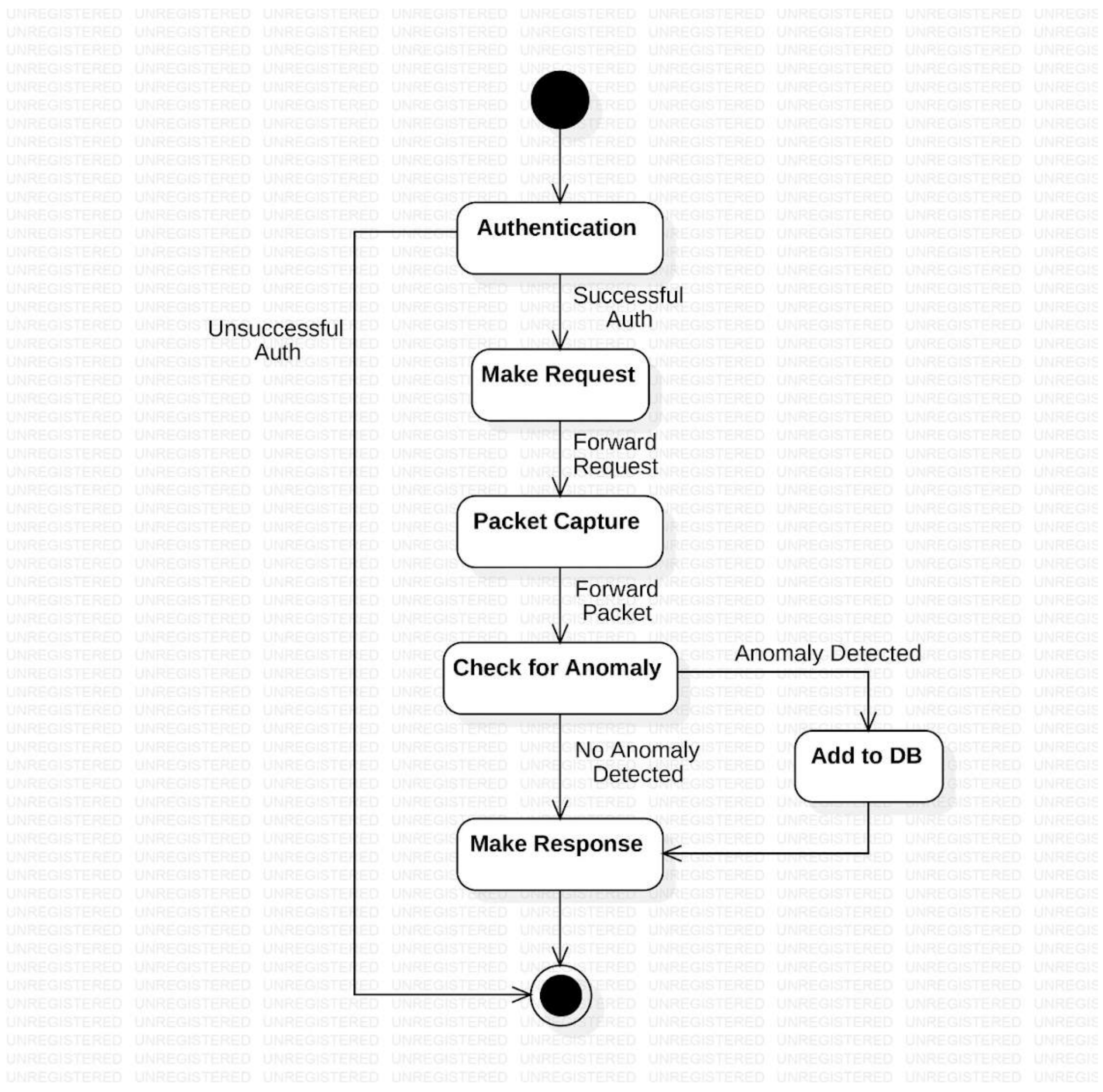
### 4.2.7 Collaboration Diagram



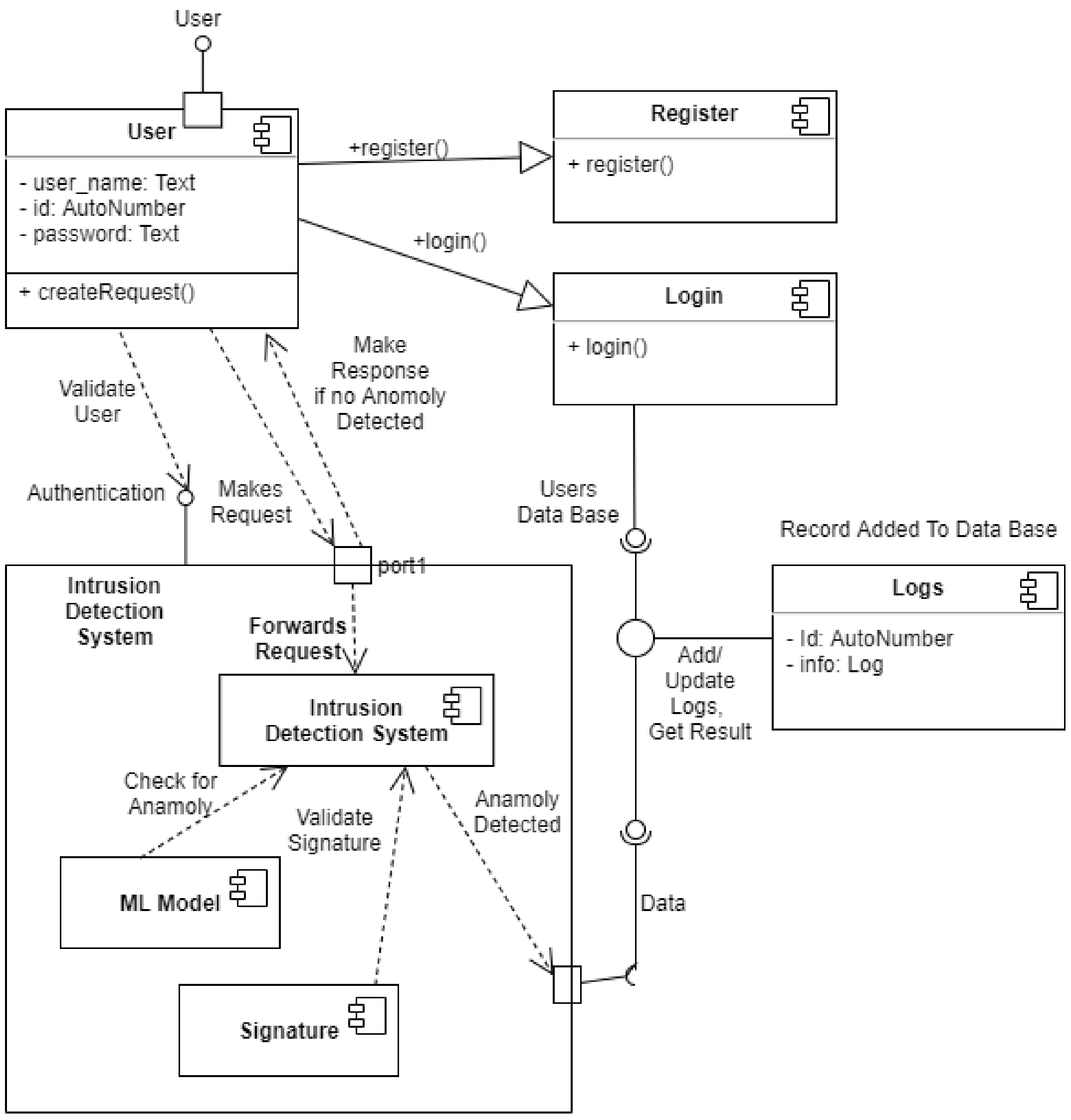
## 4.2.8 Activity Diagram



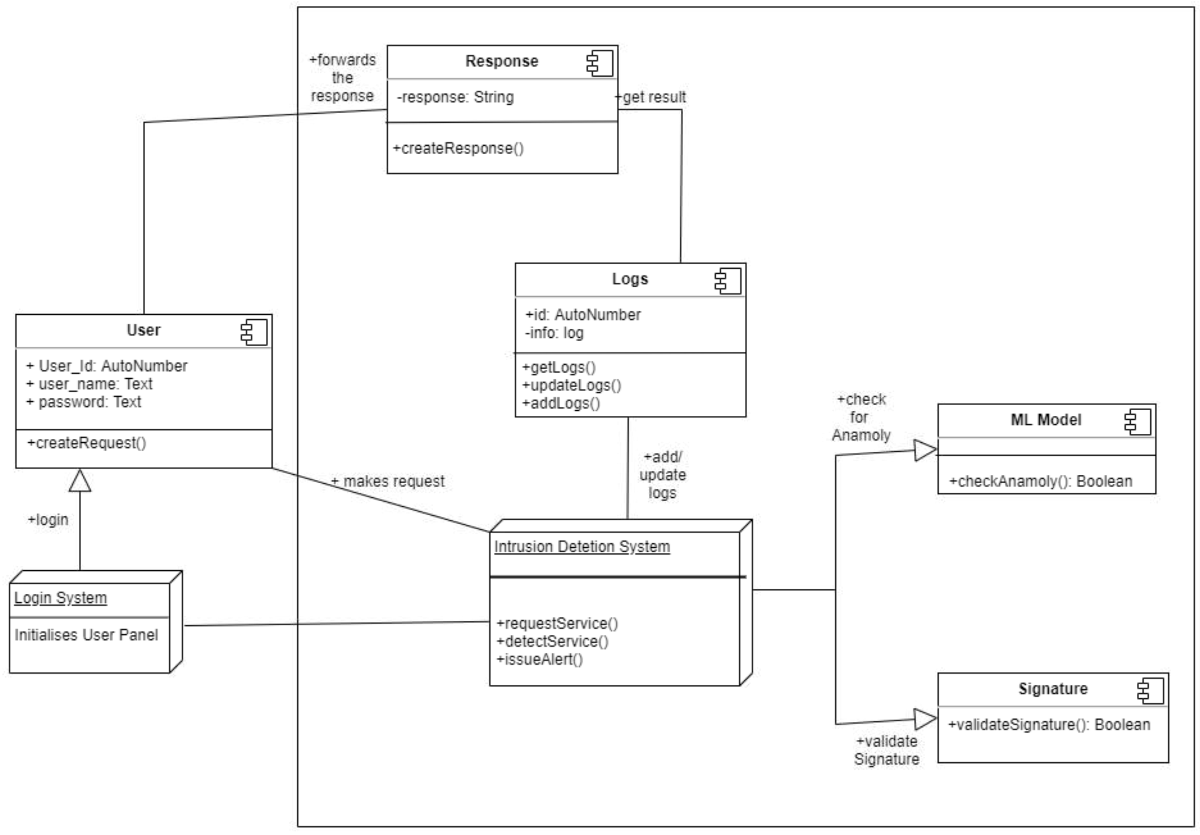
### 4.2.9 State Chart Diagram



### 4.2.10 Component Diagram



### 4.2.11 Deployment Diagram



## 4.3 Implementation

### Frontend

The frontend is kept very simple. The user is directly taken to the home page with a form, where the user needs to fill in the details and submit the form. On clicking the submit button the details are sent to the server, and the server responds with the predictions.
The front-end of the project is the user interface. We made use of HTML, CSS, JS and Materialize CSS, jQuery frameworks for the frontend.

```
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="wnameth=device-wnameth,
initial-scale=1.0">
    <title>Intrusion Detection System</title>

    <!-- CSS -->
    <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/css/mate
rialize.min.css">
    <link
href="https://fonts.googleapis.com/icon?family=Material+Icons"
rel="stylesheet">
    <style>
        body {
            margin: 0;
            padding: 18px;
        }
    </style>

    <!-- JavaScript -->
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js">
</script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/js/materi
alize.min.js"></script>
</head>

<body>
    <div class="container">
        {% if message %}
        <div class="row valign-wrapper">
            <div class="col s6">
                <h4>Result</h4>
            </div>
            <div class="col s6">
                <a class="waves-effect waves-light btn right" href="/">
                    <i class="material-icons left">arrow_left</i>
                    Go Back
                </a>
            </div>
        </div>
        <table class="striped">
            <thead>
                <tr>
                    <th>Model Name</th>
                    <th>Attack Type</th>
                </tr>
            </thead>
            <tbody>
                {% for k, v in message.items() %}
                <tr>
                    <td>{{ k }}</td>
                    <td>{{ v }}</td>
                </tr>
                {% endfor %}
            </tbody>
        </table>
        {% endif %}
        {% if not message %}
        <div class="row">
            <h4>Enter the details</h4>
        </div>
        <div class="row">
            <form class="col s12" method="POST" action="">
                <div class="row input-field">
                    <input name="duration" type="number">
                    <label for="duration">duration</label>
                </div>
                <div class="row input-field">
                    <select name="protocol_type">
                        <option value="" disabled selected>Choose your
option</option>
                        <option value="0">ICMP</option>
                        <option value="1">TCP</option>
                        <option value="2">UDP</option>
                    </select>
                    <label for="protocol_type">protocol_type</label>
                </div>
                <div class="row input-field">
                    <select name="flag">
                        <option value="" disabled selected>Choose your
option</option>
                        <option value="0">SF</option>
                        <option value="1">S0</option>
                        <option value="2">REJ</option>
                        <option value="3">RSTR</option>
                        <option value="4">RSTO</option>
                        <option value="5">SH</option>
                        <option value="6">S1</option>
                        <option value="7">S2</option>
                        <option value="8">RSTOS0</option>
                        <option value="9">S3</option>
                        <option value="10">OTH</option>
                    </select>
                    <label for="flag">flag</label>
                </div>
                <div class="row input-field">
                    <input name="src_bytes" type="number">
                    <label for="src_bytes">src_bytes</label>
                </div>
                <div class="row input-field">
                    <input name="dst_bytes" type="number">
                    <label for="dst_bytes">dst_bytes</label>
                </div>
                <div class="row input-field">
                    <input name="land" type="number">
                    <label for="land">land</label>
                </div>
                <div class="row input-field">
                    <input name="wrong_fragment" type="number">
                    <label for="wrong_fragment">wrong_fragment</label>
                </div>
                <div class="row input-field">
                    <input name="urgent" type="number">
                    <label for="urgent">urgent</label>
                </div>
                <div class="row input-field">
                    <input name="hot" type="number">
                    <label for="hot">hot</label>
                </div>
                <div class="row input-field">
                    <input name="num_failed_logins" type="number">
                    <label
for="num_failed_logins">num_failed_logins</label>
                </div>
                <div class="row input-field">
                    <input name="logged_in" type="number">
                    <label for="logged_in">logged_in</label>
                </div>
                <div class="row input-field">
                    <input name="num_compromised" type="number">
                    <label
for="num_compromised">num_compromised</label>
                </div>
                <div class="row input-field">
                    <input name="root_shell" type="number">
                    <label for="root_shell">root_shell</label>
                </div>
                <div class="row input-field">
                    <input name="su_attempted" type="number">
                    <label for="su_attempted">su_attempted</label>
                </div>
                <div class="row input-field">
                    <input name="num_file_creations" type="number">
                    <label
for="num_file_creations">num_file_creations</label>
                </div>
                <div class="row input-field">
                    <input name="num_shells" type="number">
                    <label for="num_shells">num_shells</label>
                </div>
                <div class="row input-field">
                    <input name="num_access_files" type="number">
                    <label
for="num_access_files">num_access_files</label>
                </div>
                <div class="row input-field">
                    <input name="is_guest_login" type="number">
                    <label for="is_guest_login">is_guest_login</label>
                </div>
                <div class="row input-field">
                    <input name="count" type="number">
                    <label for="count">count</label>
                </div>
                <div class="row input-field">
                    <input name="srv_count" type="number">
                    <label for="srv_count">srv_count</label>
                </div>
                <div class="row input-field">
                    <input name="serror_rate" type="number" step="any">
                    <label for="serror_rate">serror_rate</label>
                </div>
                <div class="row input-field">
                    <input name="rerror_rate" type="number" step="any">
                    <label for="rerror_rate">rerror_rate</label>
                </div>
                <div class="row input-field">
```

```
                    <input name="same_srv_rate" type="number"
step="any">
                    <label for="same_srv_rate">same_srv_rate</label>
                </div>
                <div class="row input-field">
                    <input name="diff_srv_rate" type="number"
step="any">
                    <label for="diff_srv_rate">diff_srv_rate</label>
                </div>
                <div class="row input-field">
                    <input name="srv_diff_host_rate" type="number"
step="any">
                    <label
for="srv_diff_host_rate">srv_diff_host_rate</label>
                </div>
                <div class="row input-field">
                    <input name="dst_host_count" type="number">
                    <label for="dst_host_count">dst_host_count</label>
                </div>
                <div class="row input-field">
                    <input name="dst_host_srv_count" type="number">
                    <label
for="dst_host_srv_count">dst_host_srv_count</label>
                </div>
                <div class="row input-field">
                    <input name="dst_host_diff_srv_rate" type="number"
step="any">
                    <label
for="dst_host_diff_srv_rate">dst_host_diff_srv_rate</label>
                </div>
                <div class="row input-field">
                    <input name="dst_host_same_src_port_rate"
type="number" step="any">
                    <label
for="dst_host_same_src_port_rate">dst_host_same_src_port_rate</label>
                </div>
                <div class="row input-field">
                    <input name="dst_host_srv_diff_host_rate"
type="number" step="any">
                    <label
for="dst_host_srv_diff_host_rate">dst_host_srv_diff_host_rate</label>
                </div>
                <div class="row input-field center-align">
```

```
                    <button class="btn waves-effect waves-light"
type="submit">Submit
                        <i class="material-icons right">send</i>
                    </button>
                </div>
            </form>
        </div>
        {% endif %}
    </div>

    <script>
        $(document).ready(function () {
            $('select').formSelect();
        });
    </script>
</body>

</html>
```

## Backend

We have made a REST API service using the Flask framework. REST is an acronym for REpresentational State Transfer. It is an architectural style for distributed hypermedia systems and was first presented by Roy Fielding in 2000 in his famous dissertation.

```
from flask import Flask, request
from flask.templating import render_template
from flask_cors import CORS

import pickle

from sklearn.preprocessing import MinMaxScaler

nb = pickle.load(open('./Model/models/nb.pickle','rb'))
dt = pickle.load(open('./Model/models/dt.pickle','rb'))
rf = pickle.load(open('./Model/models/rf.pickle','rb'))
svm = pickle.load(open('./Model/models/svm.pickle','rb'))
lr = pickle.load(open('./Model/models/lr.pickle','rb'))
gb = pickle.load(open('./Model/models/gb.pickle','rb'))

DEBUG = True
app = Flask(__name__)
```

```
app.config.from_object(__name__)
CORS(app)

@app.route('/', methods=['GET', 'POST'])
def index():
    message = {}
    if request.method == "POST":
        test = []
        for key, item in request.form.items():
            test.append(item)
        test = [test]
        sc = MinMaxScaler()
        X = sc.fit_transform(test)
        message["GAUSSIAN NAIVE BAYES"] = nb.predict(X)[0]
        message["DECISION TREE"] = dt.predict(X)[0]
        message["RANDOM FOREST"] = rf.predict(X)[0]
        message["SUPPORT VECTOR MACHINE"] = svm.predict(X)[0]
        message["LOGISTIC REGRESSION"] = lr.predict(X)[0]
        message["GRADIENT BOOSTING CLASSIFIER"] = gb.predict(X)[0]
    return render_template('index.html', message=message)

if __name__ == "__main__":
    app.run()
```

## Model

### Gaussian Naive Bayes

Gaussian Naive Bayes is a variant of Naive Bayes that follows Gaussian normal distribution and supports continuous data.

### Decision Tree

A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes).

### Random Forest

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean/average prediction (regression) of the individual trees

**Support Vector Machine**

support-vector machines are supervised learning models with associated learning algorithms that analyze data for classification and regression analysis

**Logistic Regression**

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (a form of binary regression).

**Gradient Boosting Classifier**

Gradient boosting classifiers are a group of machine learning algorithms that combine many weak learning models together to create a strong predictive model. Decision trees are usually used when doing gradient boosting.

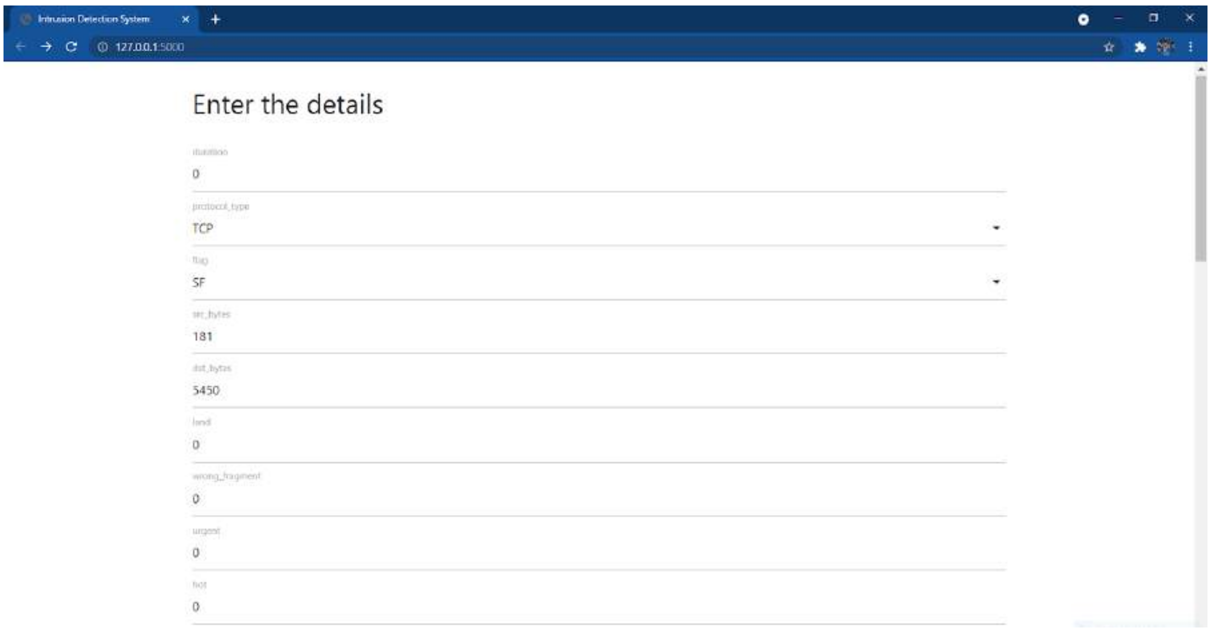
**Artificial Neural Network**

An artificial neural network (ANN) is the piece of a computing system designed to simulate the way the human brain analyzes and processes information. It is the foundation of artificial intelligence (AI) and solves problems that would prove impossible or difficult by human or statistical standards.

# 5 RESULT ANALYSIS

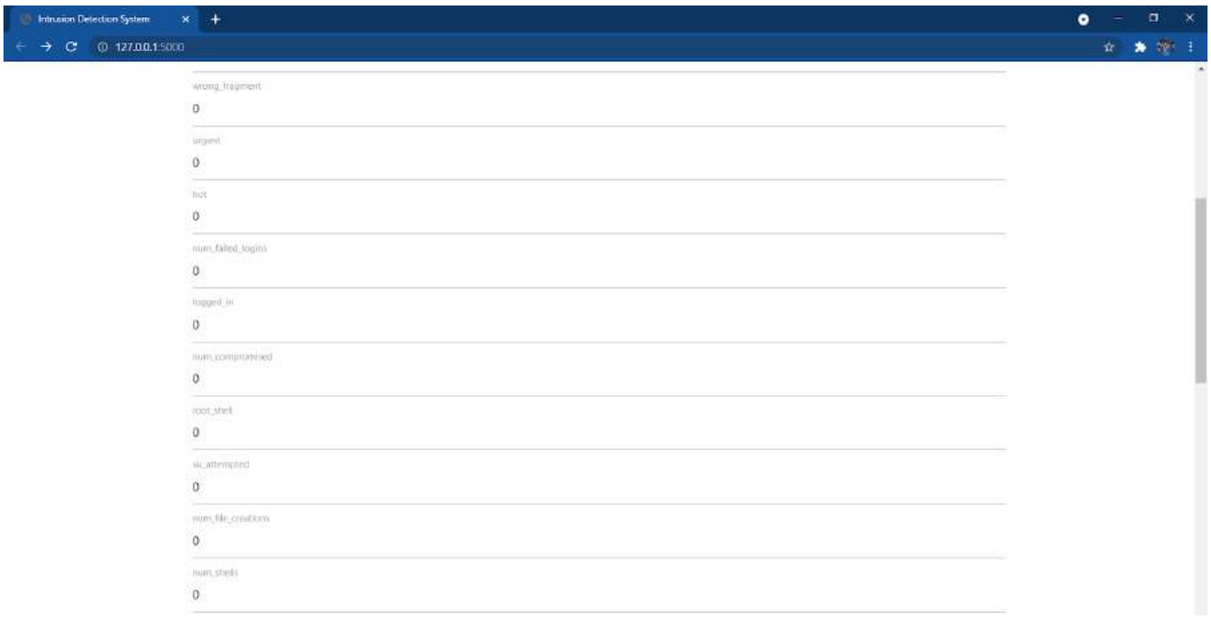
## Frontend

### Home Page:

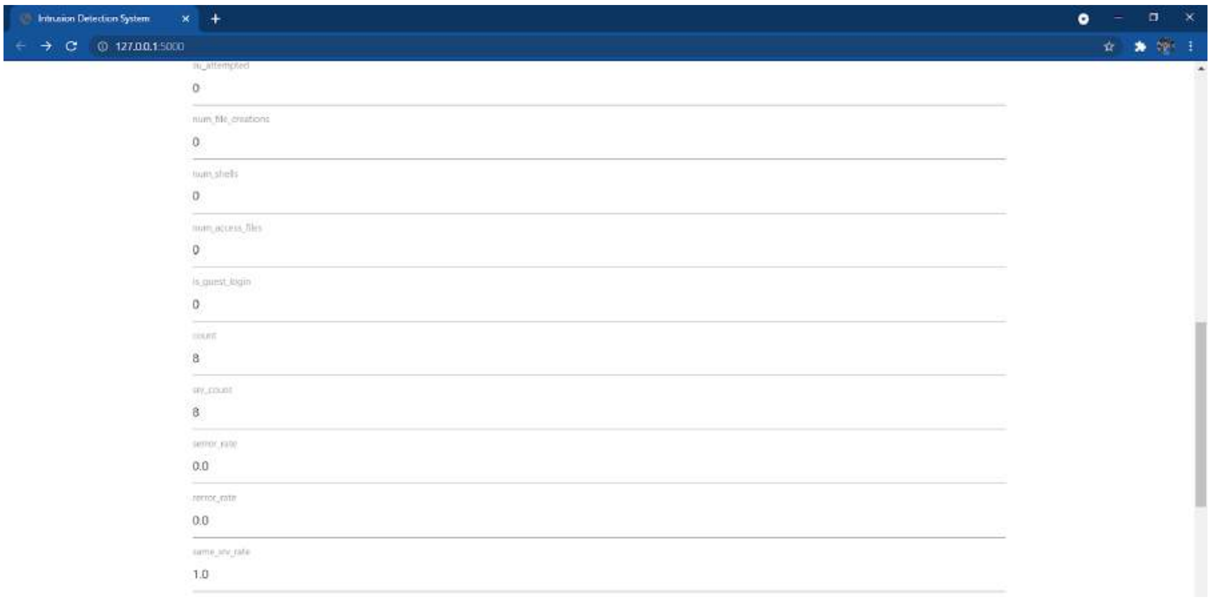


Enter the details

| | |
|---|---|
| duration | 0 |
| protocol_type | TCP |
| flag | SF |
| src_bytes | 181 |
| dst_bytes | 5450 |
| land | 0 |
| wrong_fragment | 0 |
| urgent | 0 |
| hot | 0 |



| | |
|---|---|
| wrong_fragment | 0 |
| urgent | 0 |
| hot | 0 |
| num_failed_logins | 0 |
| logged_in | 0 |
| num_compromised | 0 |
| root_shell | 0 |
| su_attempted | 0 |
| num_file_creations | 0 |
| num_shells | 0 |



su_attempted
0

num_file_creations
0

num_shells
0

num_access_files
0

is_guest_login
0

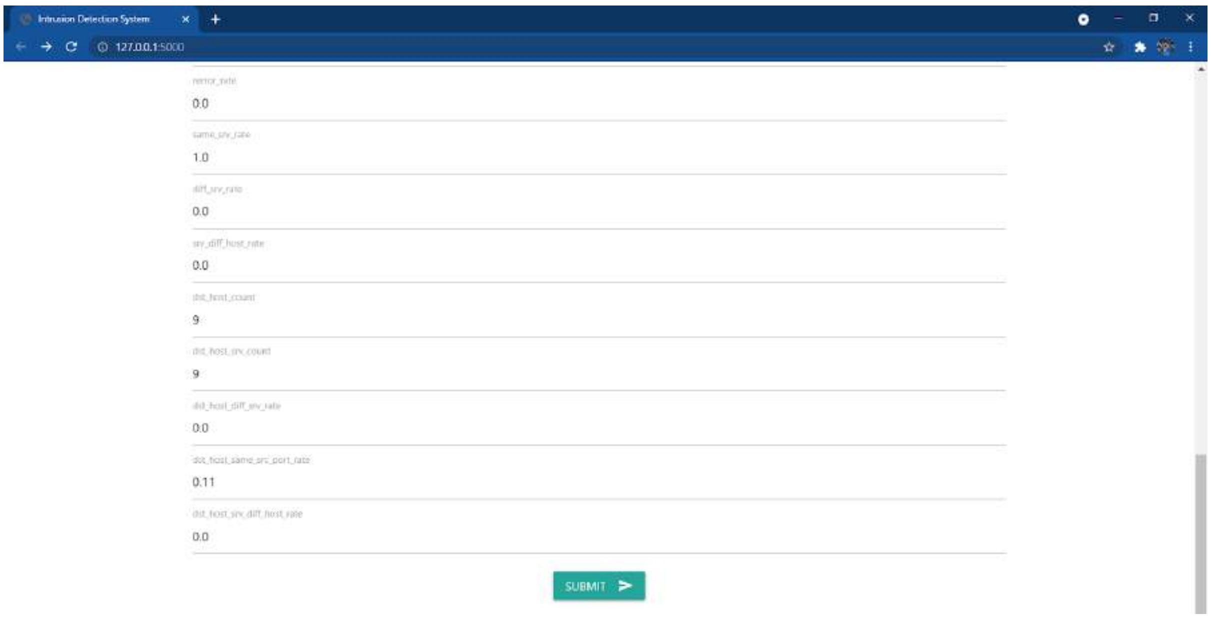
count
8

srv_count
8

serror_rate
0.0

rerror_rate
0.0

same_srv_rate
1.0



rerror_rate
0.0

same_srv_rate
1.0

diff_srv_rate
0.0

srv_diff_host_rate
0.0

dst_host_count
9

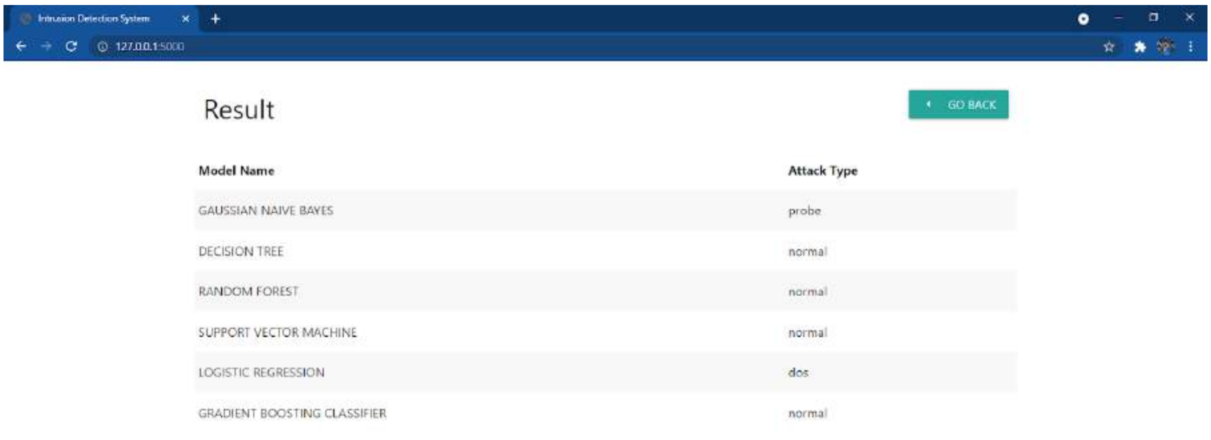
dst_host_srv_count
9

dst_host_diff_srv_rate
0.0

dst_host_same_src_port_rate
0.11

dst_host_srv_diff_host_rate
0.0

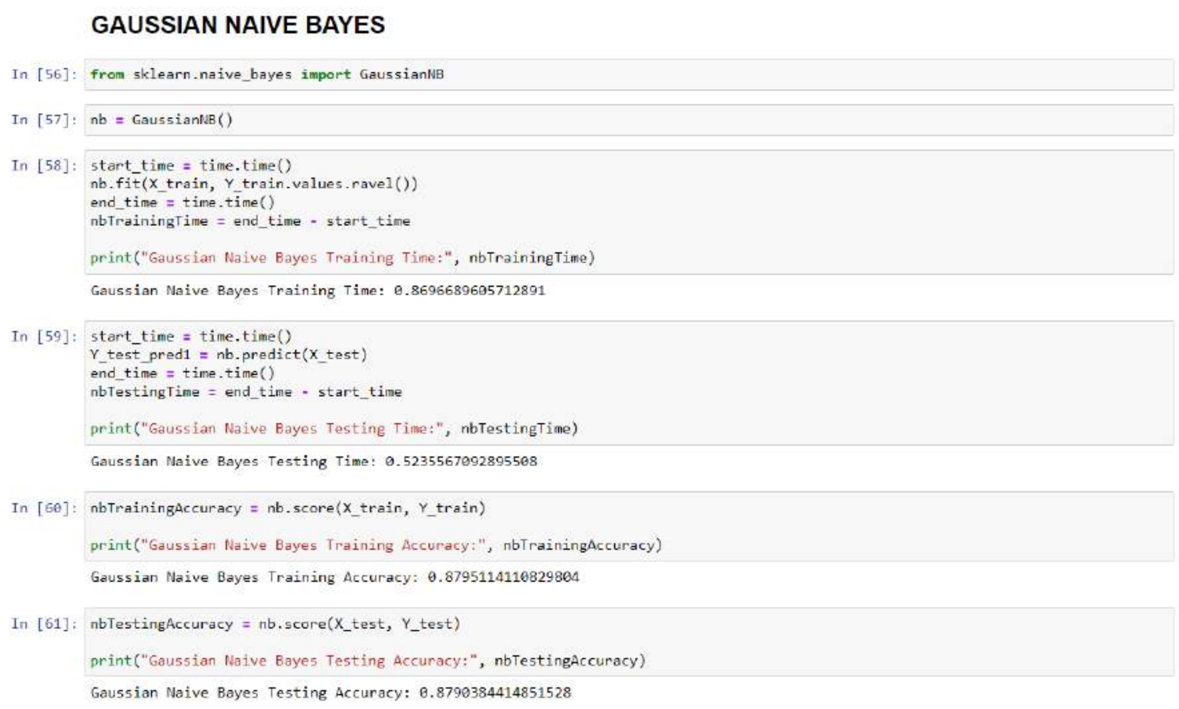
## Result Page:



Result

| Model Name | Attack Type |
|---|---|
| GAUSSIAN NAIVE BAYES | probe |
| DECISION TREE | normal |
| RANDOM FOREST | normal |
| SUPPORT VECTOR MACHINE | normal |
| LOGISTIC REGRESSION | dos |
| GRADIENT BOOSTING CLASSIFIER | normal |

## Model Training



### GAUSSIAN NAIVE BAYES

```
In [56]: from sklearn.naive_bayes import GaussianNB

In [57]: nb = GaussianNB()

In [58]: start_time = time.time()
         nb.fit(X_train, Y_train.values.ravel())
         end_time = time.time()
         nbTrainingTime = end_time - start_time

         print("Gaussian Naive Bayes Training Time:", nbTrainingTime)

         Gaussian Naive Bayes Training Time: 0.8696689605712891

In [59]: start_time = time.time()
         Y_test_pred1 = nb.predict(X_test)
         end_time = time.time()
         nbTestingTime = end_time - start_time

         print("Gaussian Naive Bayes Testing Time:", nbTestingTime)

         Gaussian Naive Bayes Testing Time: 0.5235567092895508

In [60]: nbTrainingAccuracy = nb.score(X_train, Y_train)

         print("Gaussian Naive Bayes Training Accuracy:", nbTrainingAccuracy)

         Gaussian Naive Bayes Training Accuracy: 0.8795114110829804

In [61]: nbTestingAccuracy = nb.score(X_test, Y_test)

         print("Gaussian Naive Bayes Testing Accuracy:", nbTestingAccuracy)

         Gaussian Naive Bayes Testing Accuracy: 0.8790384414851528
```

## DECISION TREE

```
In [63]: from sklearn.tree import DecisionTreeClassifier
```

```
In [64]: dt = DecisionTreeClassifier(criterion="entropy", max_depth=4)
```

```
In [65]: start_time = time.time()
         dt.fit(X_train, Y_train.values.ravel())
         end_time = time.time()
         dtTrainingTime = end_time - start_time

         print("Decision Tree Training Time:", dtTrainingTime)

         Decision Tree Training Time: 1.9449915885925293
```

```
In [66]: start_time = time.time()
         Y_test_pred2 = dt.predict(X_test)
         end_time = time.time()
         dtTestingTime = end_time - start_time

         print("Decision Tree Testing Time:", dtTestingTime)

         Decision Tree Testing Time: 0.056962013244628906
```

```
In [67]: dtTrainingAccuracy = dt.score(X_train, Y_train)

         print("Decision Tree Training Accuracy:", dtTrainingAccuracy)

         Decision Tree Training Accuracy: 0.9905829108684749
```

```
In [68]: dtTestingAccuracy = dt.score(X_test, Y_test)

         print("Decision Tree Testing Accuracy:", dtTestingAccuracy)

         Decision Tree Testing Accuracy: 0.9905230421954646
```

## RANDOM FOREST

```
In [70]: from sklearn.ensemble import RandomForestClassifier
```

```
In [71]: rf = RandomForestClassifier(n_estimators=30)
```

```
In [72]: start_time = time.time()
         rf.fit(X_train, Y_train.values.ravel())
         end_time = time.time()
         rfTrainingTime = end_time - start_time

         print("Random Forest Training Time:", rfTrainingTime)

         Random Forest Training Time: 14.292870044708252
```

```
In [73]: start_time = time.time()
         Y_test_pred3 = rf.predict(X_test)
         end_time = time.time()
         rfTestingTime = end_time - start_time

         print("Random Forest Testing Time:", rfTestingTime)

         Random Forest Testing Time: 1.0119926929473877
```

```
In [74]: rfTrainingAccuracy = rf.score(X_train, Y_train)

         print("Random Forest Training Accuracy:", rfTrainingAccuracy)

         Random Forest Training Accuracy: 0.9999788515803912
```

```
In [75]: rfTestingAccuracy = rf.score(X_test, Y_test)

         print("Random Forest Testing Accuracy:", rfTestingAccuracy)

         Random Forest Testing Accuracy: 0.9996319628037074
```

## SUPPORT VECTOR MACHINE

```
In [77]: from sklearn.svm import SVC
```

```
In [78]: svm = SVC()
```

```
In [79]: start_time = time.time()
         svm.fit(X_train, Y_train.values.ravel())
         end_time = time.time()
         svmTrainingTime = end_time - start_time

         print("Support Vector Machine Training Time:", svmTrainingTime)

         Support Vector Machine Training Time: 242.86924934387207
```

```
In [80]: start_time = time.time()
         Y_test_pred4 = svm.predict(X_test)
         end_time = time.time()
         svmTestingTime = end_time - start_time

         print("Support Vector Machine Testing Time:", svmTestingTime)

         Support Vector Machine Testing Time: 171.1831557750702
```

```
In [81]: svmTrainingAccuracy = svm.score(X_train, Y_train)

         print("Support Vector Machine Training Accuracy:", svmTrainingAccuracy)

         Support Vector Machine Training Accuracy: 0.9987552644458811
```

```
In [82]: svmTestingAccuracy = svm.score(X_test,Y_test)

         print("Support Vector Machine Testing Accuracy:", svmTestingAccuracy)

         Support Vector Machine Testing Accuracy: 0.9987916112055059
```

## LOGISTIC REGRESSION

```
In [84]: from sklearn.linear_model import LogisticRegression
```

```
In [85]: lr = LogisticRegression(max_iter=1200)
```

```
In [86]: start_time = time.time()
         lr.fit(X_train, Y_train.values.ravel())
         end_time = time.time()
         lrTrainingTime = end_time - start_time

         print("Logistic Regression Training Time:", lrTrainingTime)

         Logistic Regression Training Time: 84.1445620059967
```

```
In [87]: start_time = time.time()
         Y_test_pred5 = lr.predict(X_test)
         end_time = time.time()
         lrTestingTime = end_time - start_time

         print("Logistic Regression Testing Time:", lrTestingTime)

         Logistic Regression Testing Time: 0.0800788402557373
```

```
In [88]: lrTrainingAccuracy = lr.score(X_train, Y_train)

         print("Logistic Regression Training Accuracy:", lrTrainingAccuracy)

         Logistic Regression Training Accuracy: 0.9935285835997028
```

```
In [89]: lrTestingAccuracy = lr.score(X_test, Y_test)

         print("Logistic Regression TestingAccuracy:", lrTestingAccuracy)

         Logistic Regression TestingAccuracy: 0.9935286792985211
```

## GRADIENT BOOSTING CLASSIFIER

```
In [91]: from sklearn.ensemble import GradientBoostingClassifier
```

```
In [92]: gb = GradientBoostingClassifier(random_state=0)
```

```
In [93]: start_time = time.time()
         gb.fit(X_train, Y_train.values.ravel())
         end_time = time.time()
         gbTrainingTime = end_time - start_time

         print("Gradient Boosting Classifier Training Time:", gbTrainingTime)

         Gradient Boosting Classifier Training Time: 636.3945903778076
```

```
In [94]: start_time = time.time()
         Y_test_pred6 = gb.predict(X_test)
         end_time = time.time()
         gbTestingTime = end_time - start_time

         print("Gradient Boosting Classifier Testing Time:", gbTestingTime)

         Gradient Boosting Classifier Testing Time: 2.0250165462493896
```

```
In [95]: gbTrainingAccuracy = gb.score(X_train, Y_train)

         print("Gradient Boosting Classifier Training Accuracy:", gbTrainingAccuracy)

         Gradient Boosting Classifier Training Accuracy: 0.9979304760811374
```

```
In [96]: gbTestingAccuracy = gb.score(X_test, Y_test)

         print("Gradient Boosting Classifier Testing Accuracy:", gbTestingAccuracy)

         Gradient Boosting Classifier Testing Accuracy: 0.9977181693829856
```

## ARTIFICIAL NEURAL NETWORK

In [98]:
```python
from keras.models import Sequential
from keras.layers import Dense
from keras.wrappers.scikit_learn import KerasClassifier
```

In [99]:
```python
def fun():
    model = Sequential()

    # here 30 is output dimension
    model.add(Dense(30, input_dim=30, activation='relu', kernel_initializer='random_uniform'))

    # in next layer we do not specify the input_dim as the model is sequential so output of previous layer is input to next layer
    model.add(Dense(1, activation='sigmoid', kernel_initializer='random_uniform'))

    # 5 classes-normal,dos,probe,r2l,u2r
    model.add(Dense(5, activation='softmax'))

    # loss is categorical_crossentropy which specifies that we have multiple classes
    model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

    return model
```

In [100]:
```python
# Since, the dataset is very big and we cannot fit complete data at once so we use batch size.
# This divides our data into batches each of size equal to batch_size.
# Now only this number of samples will be loaded into memory and processed.
# Once we are done with one batch it is flushed from memory and the next batch will be processed.
ann = KerasClassifier(build_fn=fun, epochs=20, batch_size=64)
```

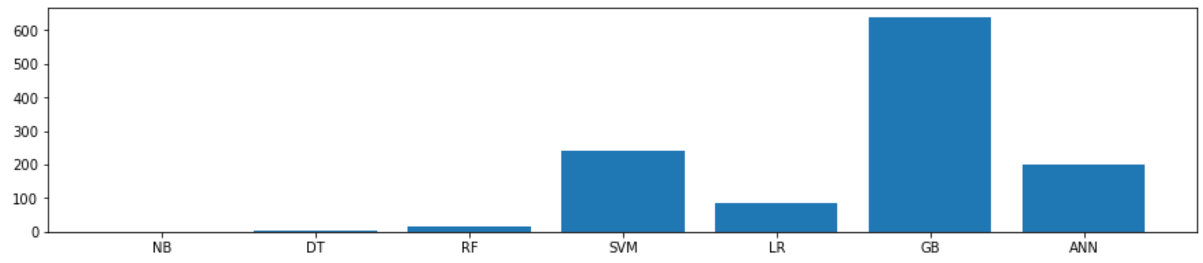
In [101]:
```python
start_time = time.time()
ann.fit(X_train, Y_train.values.ravel())
end_time = time.time()
annTrainingTime = end_time - start_time

print("Artificial Neural Network Training Time:", annTrainingTime)
```

```
Epoch 1/20
5172/5172 [==============================] - 34s 2ms/step - loss: 0.9906 - accuracy: 0.7186
Epoch 2/20
5172/5172 [==============================] - 9s 2ms/step - loss: 0.3548 - accuracy: 0.8970
Epoch 3/20
5172/5172 [==============================] - 9s 2ms/step - loss: 0.0806 - accuracy: 0.9844
Epoch 4/20
5172/5172 [==============================] - 8s 2ms/step - loss: 0.0686 - accuracy: 0.9850
Epoch 5/20
5172/5172 [==============================] - 9s 2ms/step - loss: 0.0687 - accuracy: 0.9843
Epoch 6/20
5172/5172 [==============================] - 9s 2ms/step - loss: 0.0663 - accuracy: 0.9846
Epoch 7/20
5172/5172 [==============================] - 9s 2ms/step - loss: 0.0619 - accuracy: 0.9851
Epoch 8/20
5172/5172 [==============================] - 9s 2ms/step - loss: 0.0439 - accuracy: 0.9877
Epoch 9/20
5172/5172 [==============================] - 9s 2ms/step - loss: 0.0340 - accuracy: 0.9887
Epoch 10/20
5172/5172 [==============================] - 9s 2ms/step - loss: 0.0275 - accuracy: 0.9894
Epoch 11/20
5172/5172 [==============================] - 9s 2ms/step - loss: 0.0241 - accuracy: 0.9943
Epoch 12/20
5172/5172 [==============================] - 9s 2ms/step - loss: 0.0215 - accuracy: 0.9958
Epoch 13/20
5172/5172 [==============================] - 9s 2ms/step - loss: 0.0196 - accuracy: 0.9959
Epoch 14/20
5172/5172 [==============================] - 9s 2ms/step - loss: 0.0188 - accuracy: 0.9962
Epoch 15/20
5172/5172 [==============================] - 9s 2ms/step - loss: 0.0171 - accuracy: 0.9962
Epoch 16/20
5172/5172 [==============================] - 9s 2ms/step - loss: 0.0168 - accuracy: 0.9963
Epoch 17/20
5172/5172 [==============================] - 9s 2ms/step - loss: 0.0157 - accuracy: 0.9964
Epoch 18/20
5172/5172 [==============================] - 9s 2ms/step - loss: 0.0165 - accuracy: 0.9962
Epoch 19/20
5172/5172 [==============================] - 8s 2ms/step - loss: 0.0152 - accuracy: 0.9965
Epoch 20/20
5172/5172 [==============================] - 9s 2ms/step - loss: 0.0151 - accuracy: 0.9966
Artificial Neural Network Training Time: 200.77822828292847
```

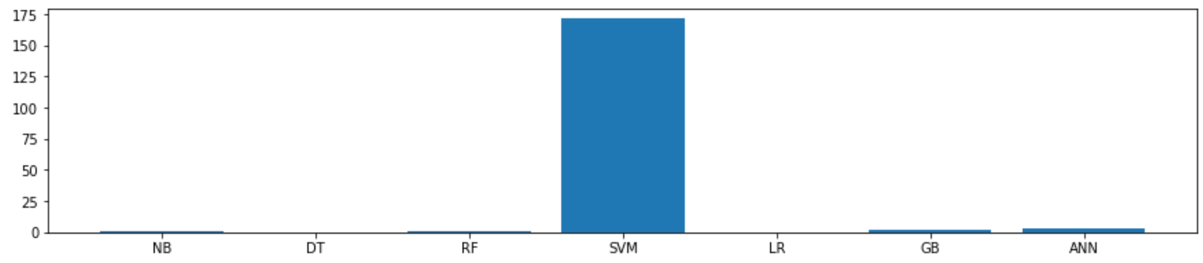
```
In [102]: start_time = time.time()
          Y_test_pred7 = ann.predict(X_test)
          end_time = time.time()
          annTestingTime = end_time - start_time

          print("Artificial Neural Network Testing Time:", annTestingTime)

C:\Users\yashw\anaconda3\lib\site-packages\keras\engine\sequential.py:450: UserWarning: `model.predict_classes()` is deprecated
and will be removed after 2021-01-01. Please use instead:* `np.argmax(model.predict(x), axis=-1)`,   if your model does multi-c
lass classification   (e.g. if it uses a `softmax` last-layer activation).* `(model.predict(x) > 0.5).astype("int32")`,   if yo
ur model does binary classification   (e.g. if it uses a `sigmoid` last-layer activation).
  warnings.warn('`model.predict_classes()` is deprecated and '

Artificial Neural Network Testing Time: 2.705467939376831

In [103]: start_time = time.time()
          Y_train_pred7 = ann.predict(X_train)
          end_time = time.time()

In [104]: annTrainingAccuracy = accuracy_score(Y_train, Y_train_pred7)

          print("Artificial Neural Network Training Accuracy:", annTrainingAccuracy)

Artificial Neural Network Training Accuracy: 0.9963866414496939

In [105]: annTestingAccuracy = accuracy_score(Y_test, Y_test_pred7)

          print("Artificial Neural Network Testing Accuracy:", annTestingAccuracy)

Artificial Neural Network Testing Accuracy: 0.9961908150183711
```

## Comparison

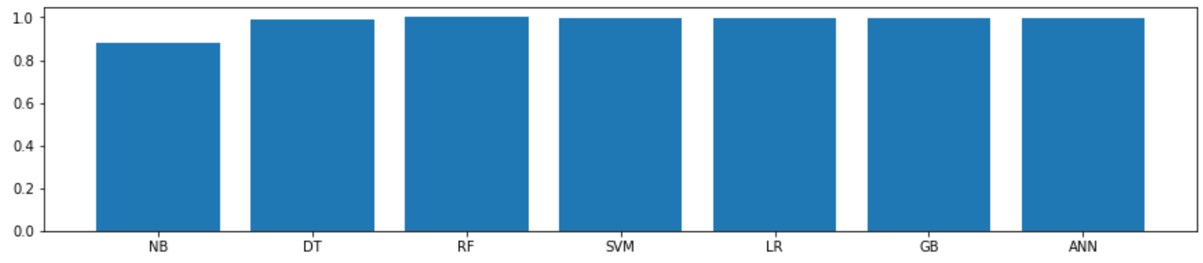
### Training Time



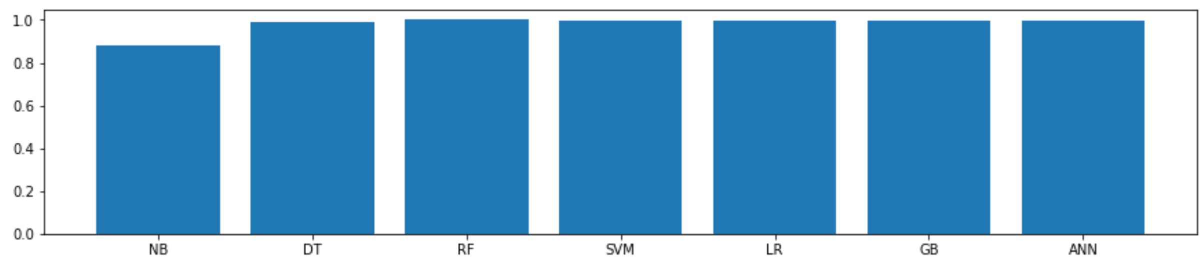
### Testing Time



### Training Accuracy



## Testing Accuracy



# 6 CONCLUSION

The above analysis of different models states that the Decision Tree model best fits our data considering both accuracy and time complexity.

Several experiments were performed and tested to evaluate the efficiency and the performance of the following machine learning classifiers: J48, Random Forest, Random Tree, MLP, Naive Bayes, and Bayes Network. All the tests were based on the KDD intrusion detection dataset. The rate of the different types of the attacks in the KDD dataset are approximately 79% of DOS attacks, 19% of normal packets and 2% of other types of attacks (R2I, U2R and PROBE).

The testing phase is implemented based on 40000 random instances of records. Several performance metrics are computed (accuracy rate, precision, false negative , false positive, true negative and true positive). The experiments have demonstrated that there is no single machine learning algorithm which can efficiently handle all the types of attacks. The random forest (rules base classifiers) achieved the lowest false negative value of (0.002), but it was far from the highest accuracy rate detection. On the other hand, Bayes network classifier had the highest value for correctly detecting the normal packets. Random forest classifiers registered the highest accuracy rate 99.96%, with the smallest RMSE value and false positive rate. It seems that the random forest classifier presents acceptable performance parameters except the false negative parameter. In contrast, all of the selected machine learning classifiers, except the MLP, were able to build their training models in an acceptable period of time. Furthermore, to save the availability and the confidentiality of the network resources, the true positive and the average accuracy rates alone are not sufficient to detect the intrusion. False negative and false positive rates are also needed to be taken into consideration.

# 7 REFERENCES

[1] Inadyuti Dutt, Samarjeet Borah, Indra Kanta Maitra, Kuharan Bhowmik, Ayindrilla Maity and Suvosmita Das “Real-Time Hybrid Intrusion Detection System Using Machine Learning Techniques”, Springer, Vol. 462, pp. 885-894, 2018

[2] Jabez, and Dr. B. Muthukumar, “Intrusion Detection System (IDS): Anomaly Detection Using Outlier Detection Approach”, Procedia Computer Science, Vol. 48, pp. 338-346, 2015

[3] Dongseong Kim, Hanam Nguyen, Syngyup Ohn, and Jongsou Park, “Fusions of GA and SVM for anomaly detection in intrusion detection system”, Springer, Vol. 3498, pp. 415-420, 2005

[4] Kwangjo, Kim, Muhamad Erza, Aminanto, Harry Chandra and Tanuwidjaja, “Network Intrusion Detection using Deep Learning “, Springer Briefs on Cyber Security Systems and Networks, pp. 978-981, 2018

[5] Wathiq Laftah Al-Yaseen, Zulaiha Ali Othman and Mohd Zakree Ahmad Nazri, “Real-Time Intrusion Detection System Using Multi-agent System”, IAENG International Journal of Computer Science, pp.1-11, 2016

[6] N. Gao, L. Gao, Q. Gao and H. Wang, “An Intrusion Detection Model Based on Deep Belief Networks,” 2014 2nd Intl.Conf. on Advanced Cloud and Big Data, Huangshan, pp. 247-252, 2014.