

# NETWORK INTRUSION DETECTION SYSTEM WITH MACHINE LEARNING

*A Project report submitted in partial fulfilment of the requirements for  
the certificate of Project School*

**BACHELOR OF TECHNOLOGY  
IN  
COMPUTER SCIENCE ENGINEERING**

***Submitted by***

**KOTTAKOTA MAHESH KUMAR (20BD1A050W) - CSE A**

**Under the guidance of  
Dr.Rajasekaran Sir**

**Session Duration : 10/09/2022 – 17/12/2022**



Signature of Faculty

Signature of Student

# ABSTRACT

The incremental increase in the usage of technology has led to an increase in the amount of data that is being processed over the Internet significantly over the time period. With the huge amount of data that is being flown over the Internet, comes the scenario of providing security to the data, and this is where an Intrusion Detection System (IDS) comes into the picture and helps in detecting any virtual security threats. Intrusion Detection System (IDS) is a system that monitors and analysis data to detect any intrusion in the system or network. Intruders find different ways to penetrate into a network. The IDS which is being proposed is being implemented using latest technologies such as Machine Learning Algorithms to classify the attacks and detecting them whenever an attack happens and also to find which machine learning algorithm is best suitable for identifying the attack.

## Keywords:

Intrusion, Intrusion Detection System, Attack classes, DOS, Probe, U2R, R2L, Normal, Abnormal, Principal Component Analysis, Prediction score, NSL-KDD Dataset, KNN Algorithm, Logistic Regression, Alerts, Mern stack, Login ,sign in, Machine learning etc,.

# WHAT IS CYBERSECURITY ?

Cybersecurity is the practice of protecting critical systems and sensitive information from digital attacks. Also known as information technology (IT) security, cybersecurity measures are designed to combat threats against Networked systems and applications, whether those threats originate from inside or outside of an organization.

A strong cybersecurity strategy has layers of protection to defend against cyber crime, including cyber attacks that attempt to access, change, or destroy data; extort money from users or the organization; or aim to disrupt normal business operations. Countermeasures should address:

- **Critical infrastructure security** - Practices for protecting the computer systems, networks, and other assets that society relies upon for national security, economic health, and/or public safety. The National Institute of Standards and Technology (NIST) has created a cybersecurity framework to help organizations in this area, while the U.S. Department of Homeland Security (DHS) provides additional guidance.
- **Network security** - Security measures for protecting a computer network from intruders, including both wired and wireless (Wi-Fi) connections.
- **Application security** - Processes that help protect applications operating on-premises and in the cloud. Security should be built into applications at the design stage, with considerations for how data is handled, user authentication, etc.
- **Cloud security** - Specifically, true confidential computing that encrypts cloud data at rest (in storage), in motion (as it travels to, from and within the cloud) and in use (during processing) to support customer privacy, business requirements and regulatory compliance standards.
- **Information security** - Data protection measures, such as the General Data Protection Regulation or GDPR, that secure your most sensitive data from unauthorized access, exposure, or theft.
- **End-user education** - Building security awareness across the organization to strengthen endpoint security. For example, users can be trained to delete suspicious email attachments, avoid using unknown USB devices, etc.
- **Disaster recovery/business continuity planning** - Tools and procedures for responding to unplanned events, such as natural disasters, power outages, or cybersecurity incidents, with minimal disruption to key operations.
- **Storage security** – IBM Flash system delivers rock solid data resilience with numerous safeguards. This includes encryption and immutable and isolated data copies. These remain in the same pool so they can quickly be restored to support recovery, minimizing the impact of a cyber attack.
- **Mobile security** – IBM security enables you to manage and secure your mobile workforce with app security, container app security and secure mobile mail.

# INTRODUCTION

An Intrusion Detection System (IDS) is a device or software application that monitors a network or systems for malicious activity or policy violations. Any intrusion activity or violation is typically reported either to an administrator or collected centrally using a security information and event management (SIEM) system. A SIEM system combines outputs from multiple sources and uses alarm filtering techniques to distinguish malicious activity from false alarms.

Although Intrusion Detection Systems monitor networks for potentially malicious activity, they are also disposed to false alarms. Hence, organizations need to fine-tune their IDS products when they first install them. It means properly setting up the intrusion detection systems to recognize what normal traffic on the network looks like as compared to malicious activity.

## Need for IDS

Building a reliable network is a very difficult task considering all different possible types of attacks. Nowadays, computer networks and their services are widely used in industry, business, and all arenas of life. Security personnel and everyone who has a responsibility for providing protection for a network and its users, have serious concerns about intruder attacks.

Network administrators and security officers try to provide a protected environment for user's accounts, network resources, personal files and passwords. Attackers may behave in two ways to carry out their attacks on networks; one of these ways is to make a network service unavailable for users or violating personal information. Denial of service (DoS) is one of the most frequent cases representing attacks on network resources and making network services unavailable for their users. There are many types of DoS attacks, and every type has its own behavior on consuming network resources to achieve the intruder's aim, which is to render the network unavailable for its users. Remote to user (R2L) is one type of computer network attacks, in which an intruder sends set of packets to another computer or server over a network where he/she does not have permission to access as a local user. User to root attacks (U2R) is a second type of attack where the intruder tries to access the network resources as a normal user, and after several attempts, the intruder becomes as a full access user. Probing is a third type of attack in which the intruder scans network devices to determine weakness in topology design or some opened ports and then use them in the future for illegal access to personal information. There are many examples that represent probing over a network, such as nmap, portsweep, ipsweep.

## Different classes of Attacks

- Denial of Service (DoS)

An attacker tries to prevent legitimate users from using a service. For example, SYN flood, Smurf and teardrop.

- User to Root (U2R)

An attacker has local access to the victim machine and tries to gain super-user privilege. For example, buffer overflow attacks.

- Remote to Local (R2L)

An attacker tries to gain access to victim machine without having an account on it. For example, password guessing attack.

- Probe

An attacker tries to gain information about the target host. For example, port-scan and ping-sweep.

## Motivation for the Work

Motivation for the work is to propose a security system, which detects malicious behaviors launched toward a system at SC level. The IDS uses data mining approaches namely KNN is used to identify attack. The attack features are learned by the machine learning algorithm.

The contributions of proposed work are:

- 1) Identifying attack class by applying machine learning algorithm.
- 2) Identifying which algorithm is best suitable for IDS problem to effectively resist insider attack.

Intrusion detection system uses classification techniques to make decision about every packet pass through the network whether it is a normal packet or an attack. Our objective is to classify the attack into multiple attack types namely DOS, U2R, R2L, PROBE packet.

## Problem Statement

Network intrusion detection begins where the firewall ends. Preventing unauthorized entry is best, but not always possible. It is important that the system is reliable and accurate and secure. Intrusion detection is defined as real-time monitoring and analysis of network activity and data for potential Vulnerabilities and attacks in progress.

A network-based intrusion detection system (NIDS) detects malicious traffic on a network. NIDS usually require promiscuous network access in order to analyse all traffic, including all unicast traffic. NIDS is defined as a system that tries to detect and alert of attempted intrusions into a system or a network.

## Project Overview

- Network Intrusion Predictor – AI Driven, web application to predict the anomalies and attacks in the network.
- Walkthrough of the application:
  - Login by providing user credentials.
  - The user should enter the defined parameters referring to the traffic input.
  - User clicks on predict and gets to know about the type of attack.
- Technologies:  
Python, Keras, Tensorflow, Pickle, React.JS, MongoDB &Node.JS

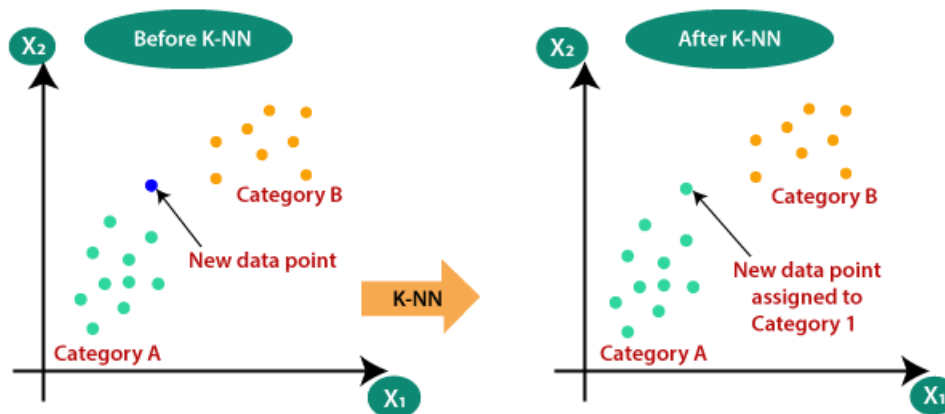
# LITERATURE SURVEY

## Network Intrusion Detection System Based On Machine Learning Algorithms:

Network and system security is of paramount importance in the present data communication environment. Hackers and intruders can create many successful attempts to cause the crash of the networks and web services by unauthorized intrusion. New threats and associated solutions to prevent these threats are emerging together with the secured system evolution. Intrusion Detection Systems (IDS) are one of these solutions. The main function of Intrusion Detection System is to protect the resources from threats. It analyses and predicts the behaviours of users, and then these behaviours will be considered an attack or a normal behaviour. We use K nearest neighbour (KNN), Convolutional Neural Network, Long Short-Term Memory (LSTM) Machine learning algorithms to detect network intrusions. First, packets are captured from the network.

### K Nearest Neighbour (KNN):

- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a **non-parametric algorithm**, which means it does not make any assumption on underlying data.
- It is also called a **lazy learner algorithm** because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.



Accuracy rate : 97% - 99%

### Advantages of KNN:

- It is simple to implement.
- It is robust to the noisy training data
- It can be more effective if the training data is large.

### Some other algorithms used for the reference purpose in the project are :

#### Convolutional Neural Network (CNN):

A **Convolutional Neural Network (ConvNet/CNN)** is a Deep Learning algorithm that can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image, and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlaps to cover the entire visual area.



Accuracy rate : 65% - 85%

**Some of the drawbacks are :**

- CNN do not encode the position and orientation of object.
- Lack of ability to be spatially invariant to the input data. Lots of training data is required.

## **Long Short-Term Memory(LSTM):**

is an artificial neural network used in the fields of artificial intelligence and deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. Such a recurrent neural network (RNN) can process not only single data points (such as images), but also entire sequences of data (such as speech or video). For example, LSTM is applicable to tasks such as unsegmented, connected handwriting recognition, speech recognition, machine translation, robot control, video games, and healthcare.

The name of LSTM refers to the analogy that a standard RNN has both long-term memory and short-term memory. The connection weights and biases in the network change once per episode of training, analogous to how physiological changes in synaptic strengths store long-term memories; the activation patterns in the network change once per time-step, analogous to how the moment-to-moment change in electric firing patterns in the brain store short-term memories. The LSTM architecture aims to provide a short-term memory for RNN that can last thousands of timesteps, thus long short-term memory.

Accuracy rate : 90% - 96%

**some of the drawbacks are:**

- LSTMs take longer to train.
- LSTMs require more memory to train.
- LSTMs are easy to overfit.
- Dropout is much harder to implement in LSTMs.
- LSTMs are sensitive to different random weight initializations.

# METHODOLOGY

Proposed smart intrusion detection system (IDS) is viewed as an effective solution for network security and protection against external threats. However, the existing IDS often has a lower detection rate under new attacks and has a high overhead when working with audit data, and thus machine learning methods have been widely applied in intrusion detection.

In our proposed method, KNN is developed as learning methods in solving the classification problem of pattern recognition and intrusion identification. Compared with other classification algorithms, KNN can better solve the problems of small samples, nonlinearity and high dimensionality.

## Advantages:

- ☐ High accuracy on detection rate.
- ☐ High True positive rate

## True Positive

A legitimate attack which triggers to produce an alarm.

## False Positive

An event signalling to produce an alarm when no attack has taken place.

## False Negative

When no alarm is raised when an attack has taken place.

## True Negative

An event when no attack has taken place and no detection is made

$$\text{Accuracy} = \frac{\text{True positives} + \text{False negatives}}{\text{Total number of samples}}$$

NIDS are typically evaluated based on the following standard performance measures

☐ True Positive Rate (TPR)

It is calculated as the ratio between the number of correctly predicted attacks and the total number of attacks. If all intrusions are detected, then the TPR is 1 which is extremely rare for an IDS. TPR is also called Detection Rate (DR) or the Sensitivity.

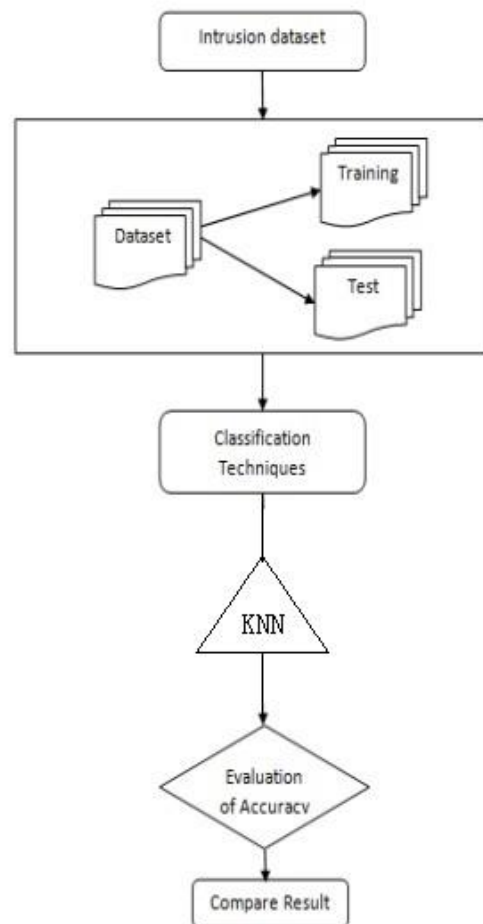
☐ False Positive Rate (FPR)

It is calculated as the ratio between the number of normal instances incorrectly classified as an attack and the total number of normal instances. ☐ False Negative Rate (FNR) False Negative means when a detector fails to identify an anomaly and classifies it as normal.

☐ Classification Rate (CR) or Accuracy

The CR measures how accurate the IDS is in detecting normal or anomalous traffic behaviour. It is described as the percentage of all those correctly predicted instances to all the instances.

## Architecture of the system:



## Implementation of the System:

The proposed work is implemented in Python 3.6.4 with libraries scikit-learn, pandas, matplotlib and other mandatory libraries. We downloaded dataset from [www.unb.ca](http://www.unb.ca). The data downloaded contains train set and test set separately with four different classes of intrusions. The train dataset considered as train set and test dataset considered as test set.

We have collected the dataset for the intrusion detection system with following details from KDD dataset and we applied machine learning algorithm KNN.

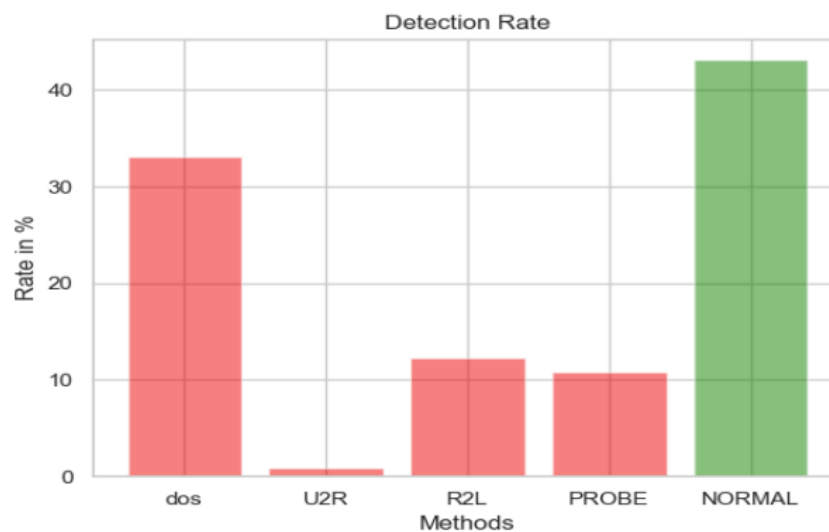
## Dataset Details:

### ☐ Data collection

The data collection process involves the selection of quality data for analysis. Here we used KDD intrusion dataset taken from uci.edu for machine learning implementation. The job of a data analyst is to find ways and sources of collecting relevant and comprehensive data, interpreting it, and analysing results with the help of statistical techniques.

### ☐ Data visualization

A large amount of information represented in graphic form is easier to understand and analyze. Some companies specify that a data analyst must know how to create slides, diagrams, charts, and templates. In our approach, the detection rates of intrusion are shown as data visualization part.



### ☐ Data pre-processing

The purpose of pre-processing is to convert raw data into a form that fits machine learning. Structured and clean data allows a data scientist to get more precise results from an applied machine learning model. The technique includes data formatting, cleaning, and sampling.

## □ Data splitting

A dataset used for machine learning should be partitioned into three subsets — training, test, and validation sets.

**Training set:** A data scientist uses a training set to train a model and define its optimal parameters it has to learn from data.

**Test set:** A test set is needed for an evaluation of the trained model and its capability for generalization. The latter means a model's ability to identify patterns in new unseen data after having been trained over a training data. It's crucial to use different subsets for training and testing to avoid model overfitting, which is the incapacity for generalization we mentioned above.

## □ Model Training

After a data scientist has preprocessed the collected data and split it into train and test can proceed with a model training. This process entails “feeding” the algorithm with training data. An algorithm will process data and output a model that is able to find a target value (attribute) in new data an answer you want to get with predictive analysis. The purpose of model training is to develop a model.

## NSL-KDD dataset

NSL-KDD is a data set suggested to solve some of the inherent problems of the KDD'99 data set which are mentioned in [1]. Although, this new version of the KDD data set still suffers from some of the problems discussed by McHugh and may not be a perfect representative of existing real networks, because of the lack of public data sets for network-based IDSs, we believe it still can be applied as an effective benchmark data set to help researchers compare different intrusion detection methods.

Furthermore, the number of records in the NSL-KDD train and test sets are reasonable. This advantage makes it affordable to run the experiments on the complete set without the need to randomly select a small portion. Consequently, evaluation results of different research work will be consistent and comparable.

## Improvements to the NSL-KDD dataset

The NSL-KDD data set has the following advantages over the original KDD data set:

- It does not include redundant records in the train set, so the classifiers will not be biased towards more frequent records.
- There is no duplicate records in the proposed test sets; therefore, the performance of the learners are not biased by the methods which have better detection rates on the frequent records.
- The number of selected records from each difficulty level group is inversely proportional to the percentage of records in the original KDD data set. As a result, the classification rates of distinct machine learning methods vary in a wider range, which makes it more efficient to have an accurate evaluation of different learning techniques.
- The number of records in the train and test sets are reasonable, which makes it affordable to run the experiments on the complete set without the need to randomly select a small portion. Consequently, evaluation results of different research works will be consistent and comparable.

## Statistical observations

One of the most important deficiencies in the KDD data set is the huge number of redundant records, which causes the learning algorithms to be biased towards the frequent records, and thus prevent them from learning unfrequent records which are usually more harmful to networks such as U2R and R2L attacks. In addition, the existence of these repeated records in the test set will cause the evaluation results to be biased by the methods which have better detection rates on the frequent records.

In addition, we analyzed the difficulty level of the records in KDD data set. Surprisingly, about 98% of the records in the train set and 86% of the records in the test set were correctly classified with all the 21 learners.

In order to perform our experiments, we randomly created three smaller subsets of the KDD train set each of which included fifty thousand records of information. Each of the learners where trained over the created train sets. We then employed the 21 learned machines (7 learners, each trained 3 times) to label the records of the entire KDD train and test sets, which provides us with 21 predicated labels for each record. Further, we annotated each record of the data set with a *#successfulPrediction* value, which was initialized to zero. Now, since the KDD data set provides the correct label for each record, we compared the predicated label of each record given by a specific learner with the actual label, where we incremented *#successfulPrediction* by one if a match was found. Through this process, we calculated the number of learners that were able to correctly label that given record. The highest value for *#successfulPrediction* is 21, which conveys the fact that all learners were able to correctly predict the label of that record.

## Statistics of redundant records in the KDD train set

**Original records | Distinct records | Reduction rate**

- **Attacks:** 3,925,650 | 262,178 | 93.32%
- **Normal:** 972,781 | 812,814 | 16.44%
- **Total:** 4,898,431 | 1,074,992 | 78.05%



## Statistics of redundant records in the KDD test set

Original records | Distinct records | Reduction rate

- **Attacks:** 250,436 | 29,378 | 88.26%
- **Normal:** 60,591 | 47,911 | 20.92%
- **Total:** 311,027 | 77,289 | 75.15%

## License

You may redistribute, republish, and mirror the NSL-KDD dataset in any form. However, any use or redistribution of the data must include a citation to the NSL-KDD dataset and the paper referenced below.

**References:** M. Tavallaei, E. Bagheri, W. Lu, and A. Ghorbani, "[A Detailed Analysis of the KDD CUP 99 Data Set](#)," Submitted to Second IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA), 2009.

**References Link :** <https://www.unb.ca/cic/datasets/nsf.html>

## Parameter in the NSL-KDD Dataset :

<i>Feature name</i>	<i>Description</i>	<i>type</i>
duration	length (number of seconds) of the connection	continuous
protocol_type	type of the protocol, e.g. tcp, udp, etc.	discrete
src_bytes	number of data bytes from source to destination	continuous
dst_bytes	number of data bytes from destination to source	continuous
flag	normal or error status of the connection	discrete
land	1 if connection is from/to the same host/port; 0 otherwise	discrete
wrong_fragment	number of "wrong" fragments	Continuous

**Basic features of individual TCP connections.**

<i>Feature name</i>	<i>Description</i>	<i>Type</i>
hot	number of ``hot" indicators	Continuous
num_failed_logins	number of failed login attempts	Continuous
logged_in	1 if successfully logged in; 0 otherwise	Discrete
num_compromised	number of ``compromised" conditions	Continuous
root_shell	1 if root shell is obtained; 0 otherwise	Discrete
su_attempted	1 if ``su root" command attempted; 0 otherwise	Discrete
num_root	number of ``root" accesses	Continuous
num_file_creations	number of file creation operations	Continuous
num_shells	number of shell prompts	Continuous
num_access_files	number of operations on access control files	Continuous
num_outbound_cmds	number of outbound commands in an ftp session	Continuous
is_hot_login	1 if the login belongs to the ``hot" list; 0 otherwise	Discrete
is_guest_login	1 if the login is a ``guest"login; 0 otherwise	Discrete

**Content features within a connection suggested by domain knowledge.**

<i>Feature name</i>	<i>Description</i>	<i>Type</i>
<i>feature name</i>	<i>description</i>	<i>Type</i>
count	number of connections to the same host as the current connection in the past two seconds	Continuous
	<i>Note: The following features refer to these same-host connections.</i>	

serror_rate	% of connections that have ``SYN" errors	continuous
rerror_rate	% of connections that have ``REJ" errors	continuous
same_srv_rate	% of connections to the same service	continuous
diff_srv_rate	% of connections to different services	continuous
srv_count	number of connections to the same service as the current connection in the past two seconds	continuous
srv_error_rate	% of connections that have ``SYN" errors	continuous
srv_diff_host_rate	% of connections to different hosts	continuous

**Traffic features computed using a two-second time window**

## Activity Diagram

An activity diagram shows the flow from activity to activity. An activity is a going nonatomic execution within a state machine. An activity results in some action, results in a change of state or return of a value.

Activity Diagram commonly contains:

- Activity states and action states.
- Transitions.
- Objects, it may contain nodes and constraint .

### Activity states and action states:

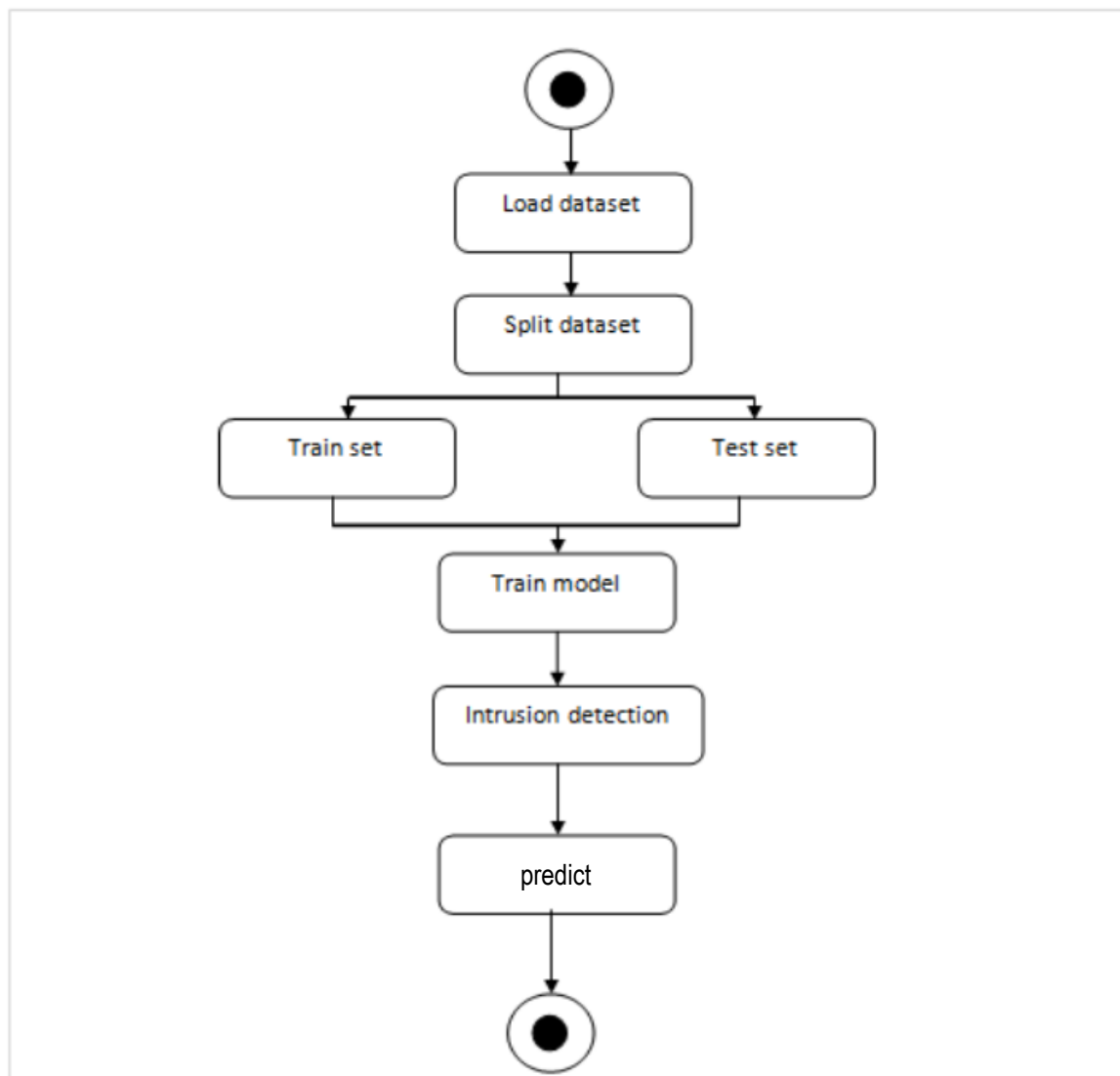
An executable atomic computation is called action state, which cannot be decomposed. Activity state is non-atomic, decomposable and takes some duration to execute.

**Transition:** It is a path from one state to the next state, represented as simple directed line.

**Branching:** When an alternate path exists, branching arises which is represented by open diamond. It has a incoming transition, two or more outgoing transitions.

**Forking and Joining:** The synchronization bar when split one flow into two or more flows is called fork. When two or more flows are combined at synchronization bar, the bar is called join.

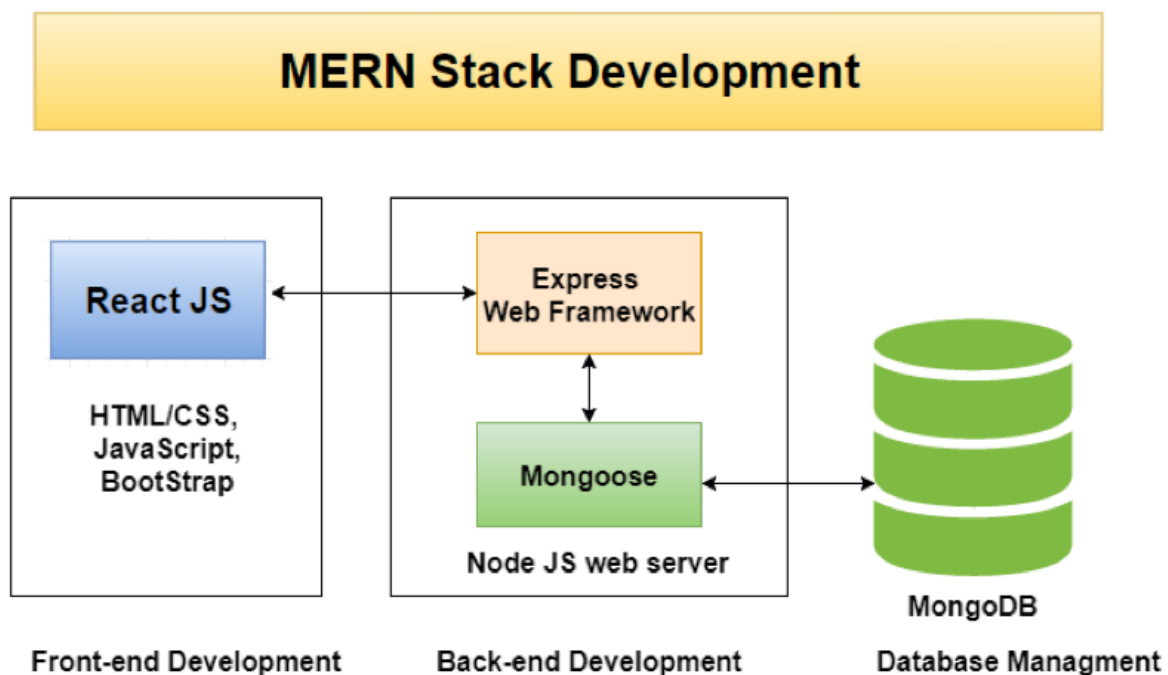
**Swim Lanes:** Group work flow is called swim lanes. All groups are portioned by vertical solid lines. Each swim lane specifies locus of activities and has a unique name. Each swim lane is implemented by one or more classes. Transition may occur between objects across swim lanes.



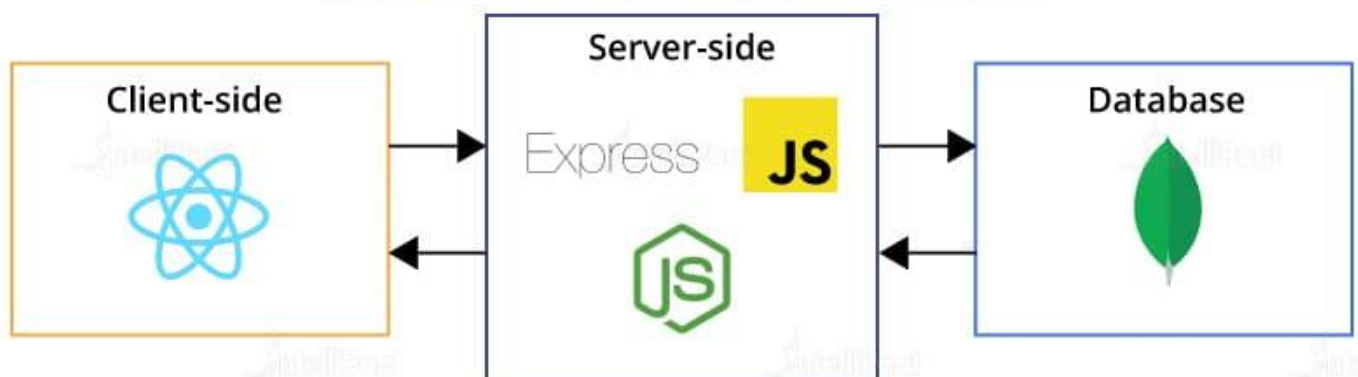
# MERN Stack :

MERN Stack is a Javascript Stack that is used for easier and faster deployment of full-stack web applications. MERN Stack comprises of 4 technologies namely: [MongoDB](#), [Express](#), [React](#) and [Node.js](#). It is designed to make the development process smoother and easier.

Each of these 4 powerful technologies provides an end-to-end framework for the developers to work in and each of these technologies play a big part in the development of web applications.



## The 3-tier Architecture of MERN Stack



## Advantages of using MERN STACK :

Several advantages of the MERN stack include numerous capabilities and plugins that operate efficiently with the backside development platform to construct fast web apps and APIs.

- **React.js:** The javascript React Javascript framework allows for the efficient creation of HTML-based user interfaces. In the MVC model, React serves as the View layer and provides declarative views. JavaScript, a full-featured scripting language, is used to create repeating DOM components and works well for SPAs Node. It stands for the JavaScript operating system. MERN's component employs automated programming built on Google's search V8 JavaScript engine. With an efficient database, you gain faster loading times and faster web programming.
- **No context changing:** For the whole program, JavaScript generates both client-side and server-side components; the web technology does not require response time and delivers efficient web apps.
- **MVC:** A significant feature of the MERN stack is that it provides an architecture that makes it easier for developers to construct online applications.
- **Entire:** Because there is no context switching, you get philosophically aligned and strong technologies that function tangibly together and efficiently handle client and virtual machine programming faster.
- **Simple Training Curve:** To get MERN stack benefits when designing web apps, professional developers need to be proficient in JS and JSON.

# SYSTEM WORK FLOW

## Prerequisite:

Nodejs should be installed and react extensions in vs code

1.npx create-react-app app\_name

2.create a backend folder in this folder and go into the created folder and run following commands for installations

3.npm init -y

4.npm i express

5.create index.js

3.npm i mongoose

4.npm i -D nodemon

5.npm install --save express-validator

6.npm i bcryptjs

7.npm i react-router-dom

8.npm i react-router-dom concurrently

9.npm install cors

10.npm i axios-react

1) Enter a command “npm run both” in cmd opened with the path of your particular project.

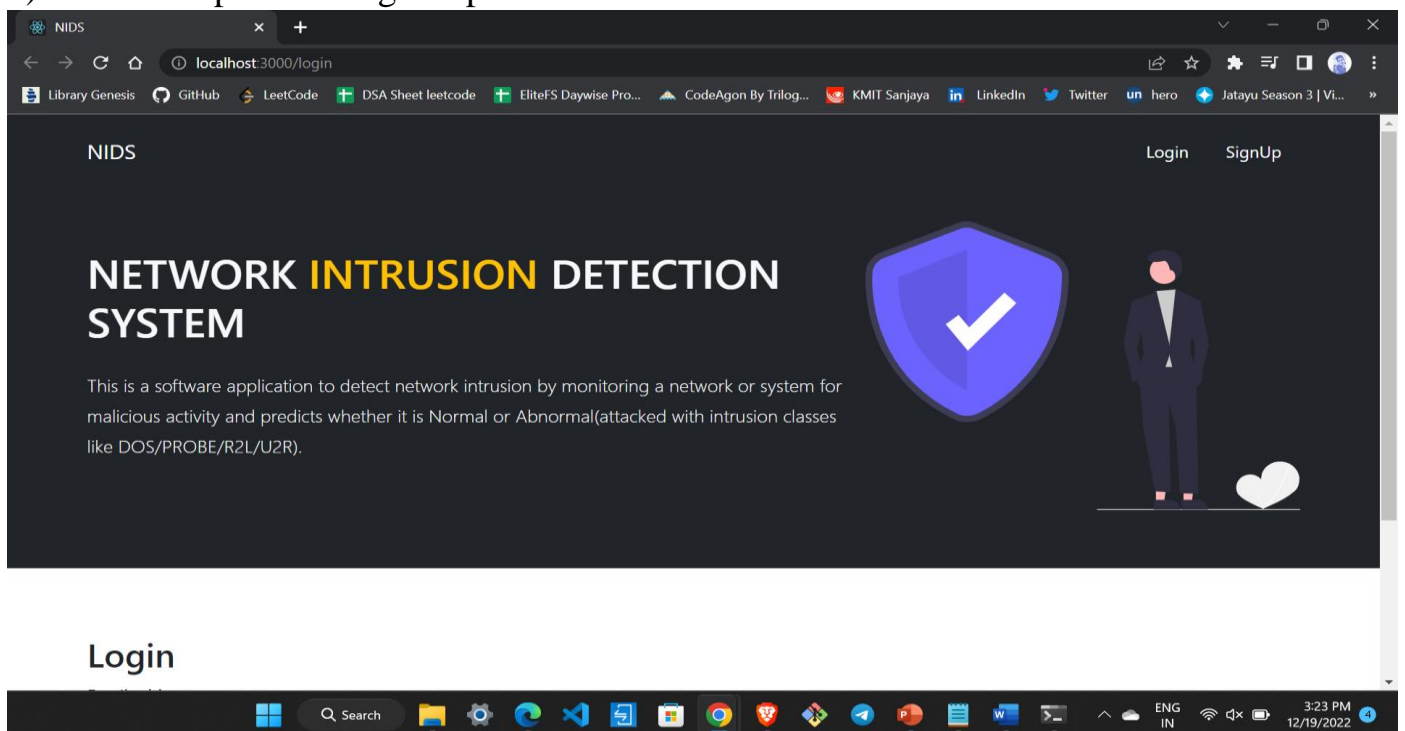
```
Windows PowerShell
Microsoft Windows [Version 10.0.22621.963]
(c) Microsoft Corporation. All rights reserved.

D:\INTRUSION\nids>npm run both

> inotebook2@0.1.0 both
> concurrently "npm run start" "nodemon backend/index.js"

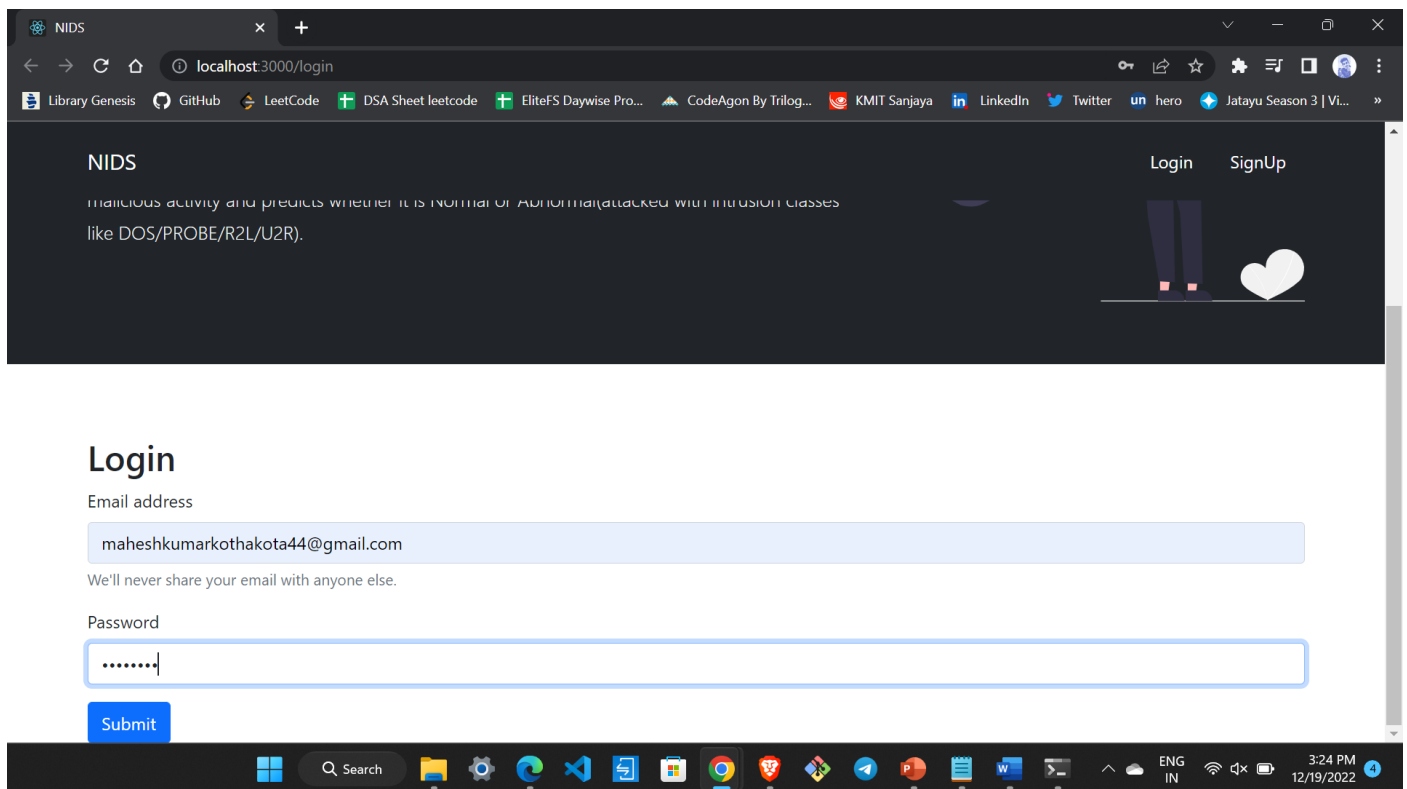
[1] [nodemon] 2.0.20
[1] [nodemon] to restart at any time, enter `rs`
[1] [nodemon] watching path(s): *.*
[1] [nodemon] watching extensions: js,mjs,json
[1] [nodemon] starting `node backend/index.js`
[0]
[0] > inotebook2@0.1.0 start
[0] > react-scripts start
[0]
[1] (node:11960) [MONGOOSE] DeprecationWarning: Mongoose: the `strictQuery` option will be switched back to `false` by default in Mon
goose 7. Use `mongoose.set('strictQuery', false);` if you want to prepare for this change. Or use `mongoose.set('strictQuery', true);`
to suppress this warning.
[1] (Use `node --trace-deprecation ...` to show where the warning was created)
[1] inotebook app listening at port 5000
[1] Connected to Mongo Successfully
[0] (node:10348) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_MIDDLEWARE] DeprecationWarning: `onAfterSetupMiddleware` option is deprecated
. Please use the `setupMiddlewares` option.
[0] (Use `node --trace-deprecation ...` to show where the warning was created)
[0] (node:10348) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_MIDDLEWARE] DeprecationWarning: `onBeforeSetupMiddleware` option is deprecate
d. Please use the `setupMiddlewares` option.
[0] Starting the development server...
[0]
[0] Compiled successfully!
[0]
```

2) The developed server gets opened.



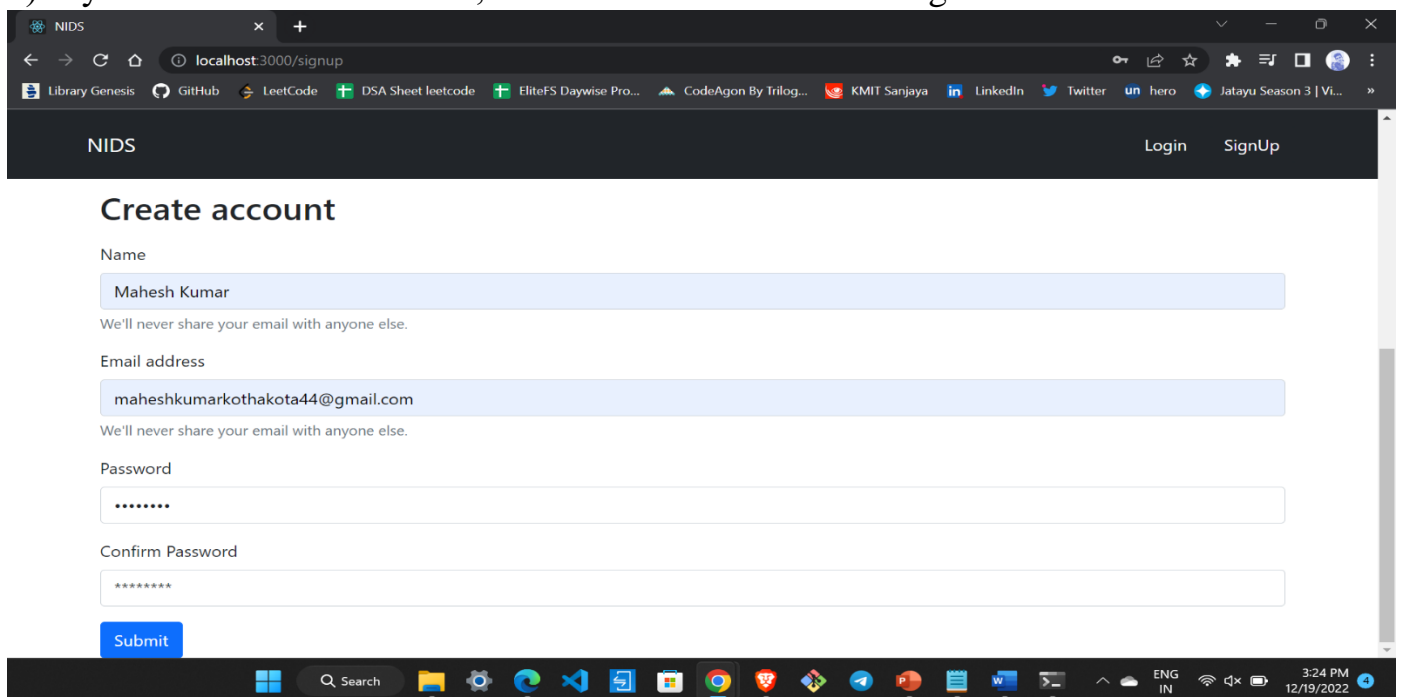


3) If already have an account, then login to the browser.



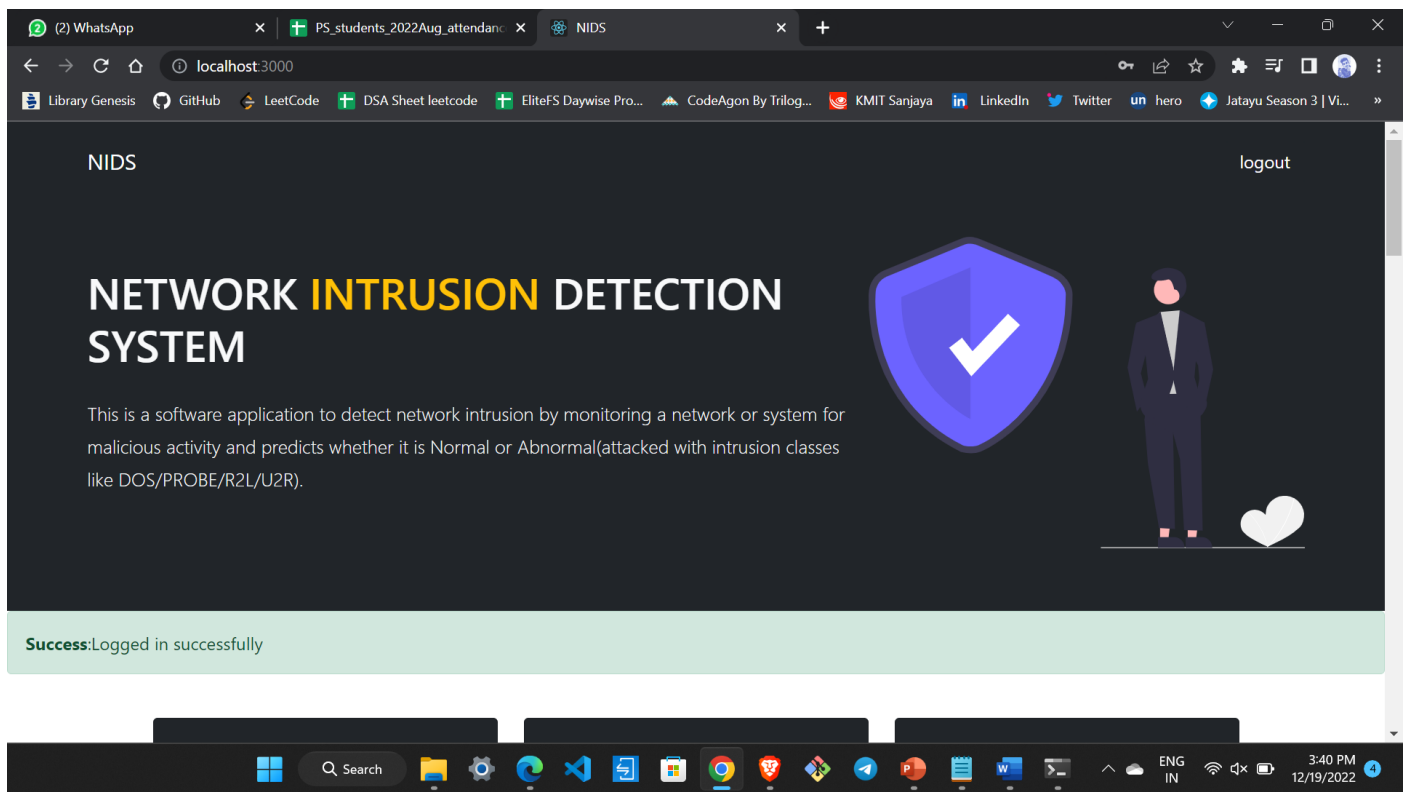
The screenshot shows a web browser window with the address bar displaying 'localhost:3000/login'. The page header includes the NIDS logo and navigation links for 'Login' and 'SignUp'. The main content area features a 'Login' heading, an 'Email address' field containing 'maheshkumarkothakota44@gmail.com', and a 'Password' field with masked characters. A 'Submit' button is located below the password field. The browser's taskbar at the bottom shows various application icons and the system clock indicating 3:24 PM on 12/19/2022.

4) If you don't have an account, then create an account and login to the website.

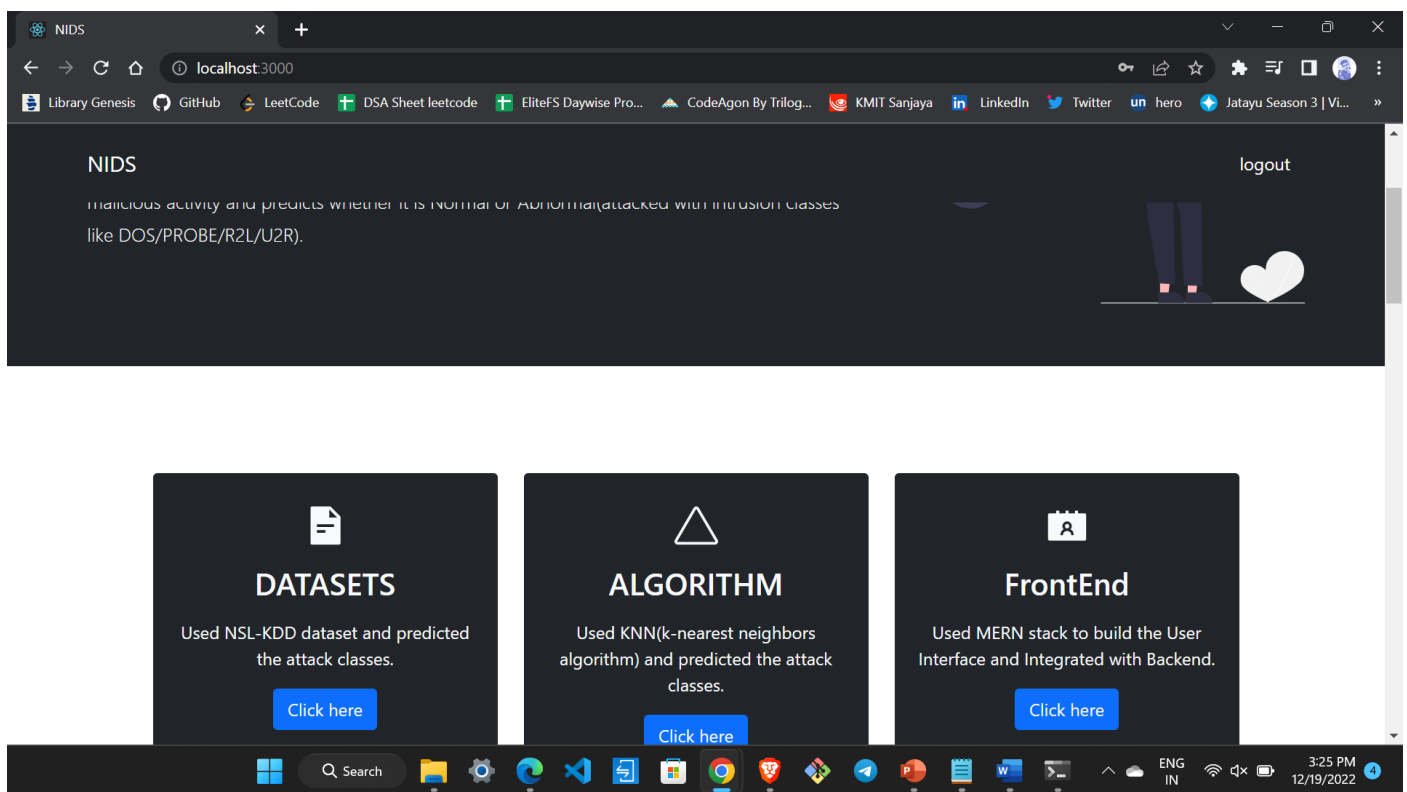


The screenshot shows a web browser window with the address bar displaying 'localhost:3000/signup'. The page header includes the NIDS logo and navigation links for 'Login' and 'SignUp'. The main content area features a 'Create account' heading, followed by 'Name' (filled with 'Mahesh Kumar'), 'Email address' (filled with 'maheshkumarkothakota44@gmail.com'), 'Password', and 'Confirm Password' fields, all with masked characters. A 'Submit' button is located below the confirm password field. The browser's taskbar at the bottom shows various application icons and the system clock indicating 3:24 PM on 12/19/2022.

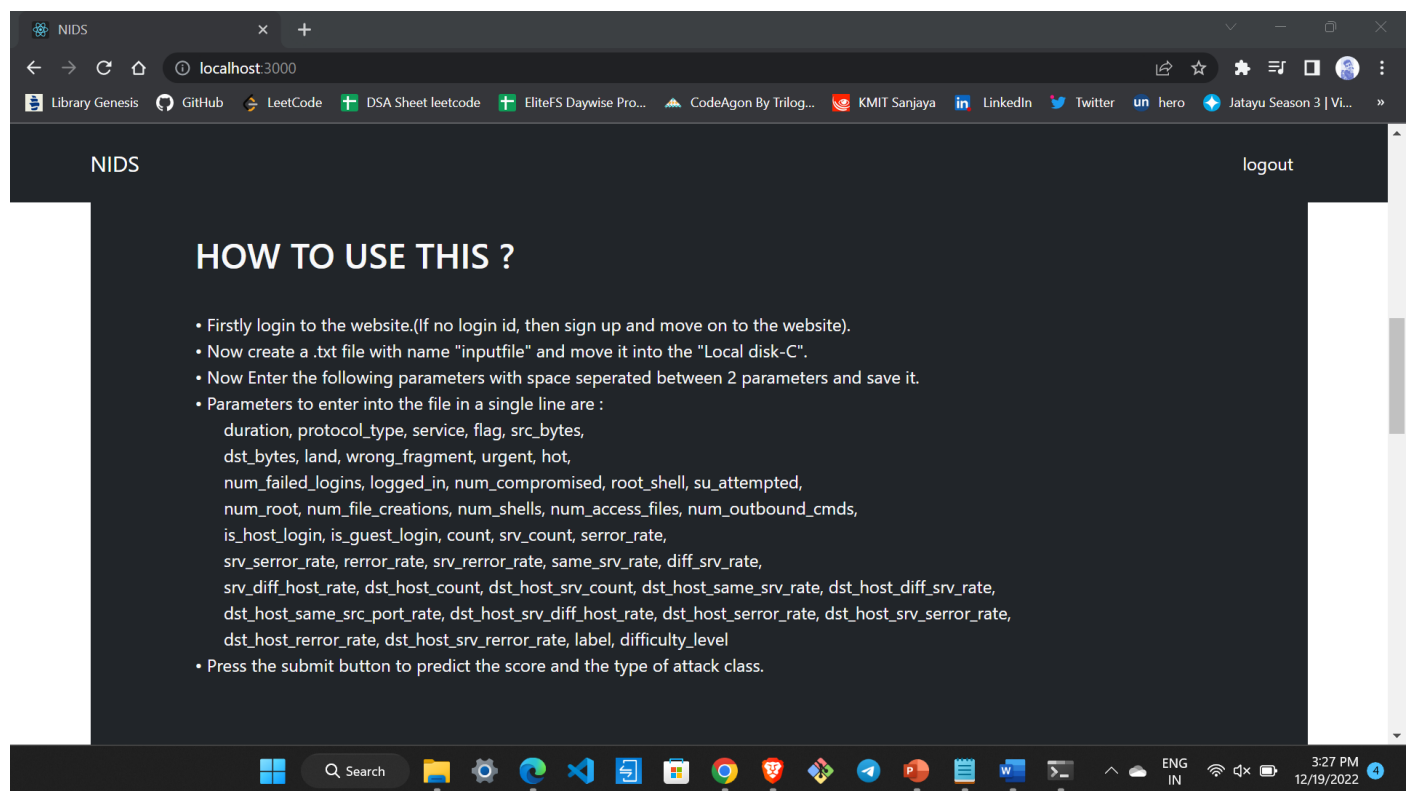
5) This the interface of the website. Move down.



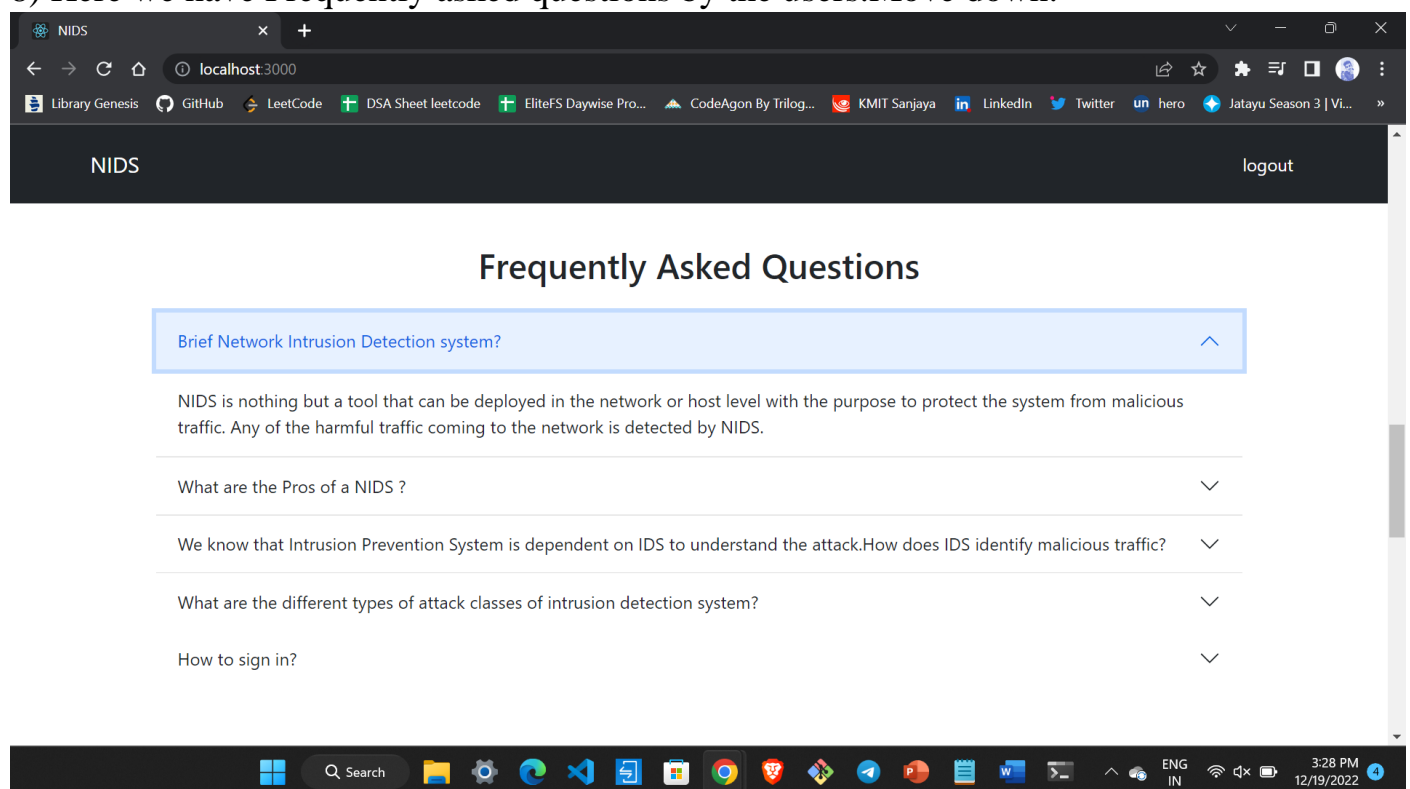
6) Here we have the datasets used, algorithms used and front end of the website. Now Move further down.



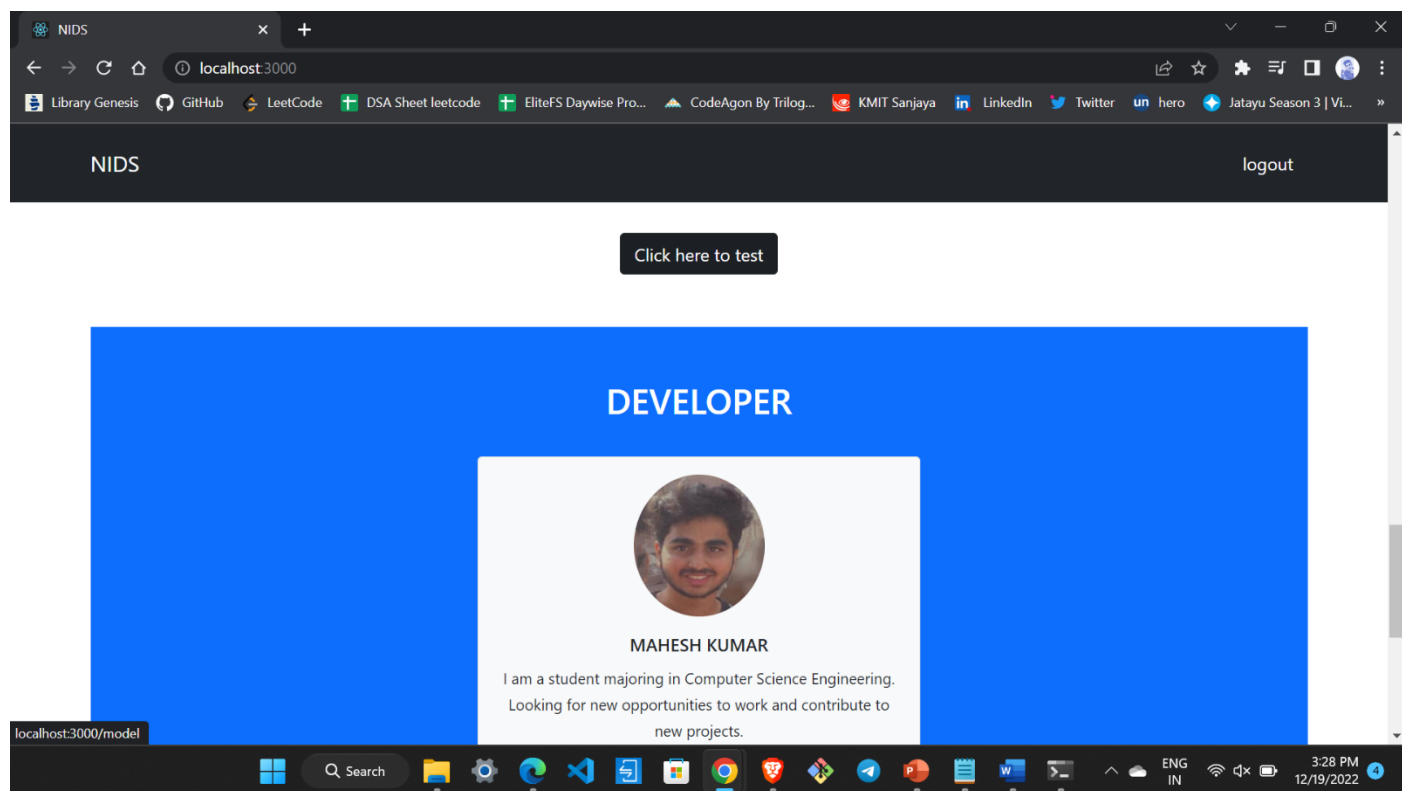
7) Here we have the steps to use the website .Move down.



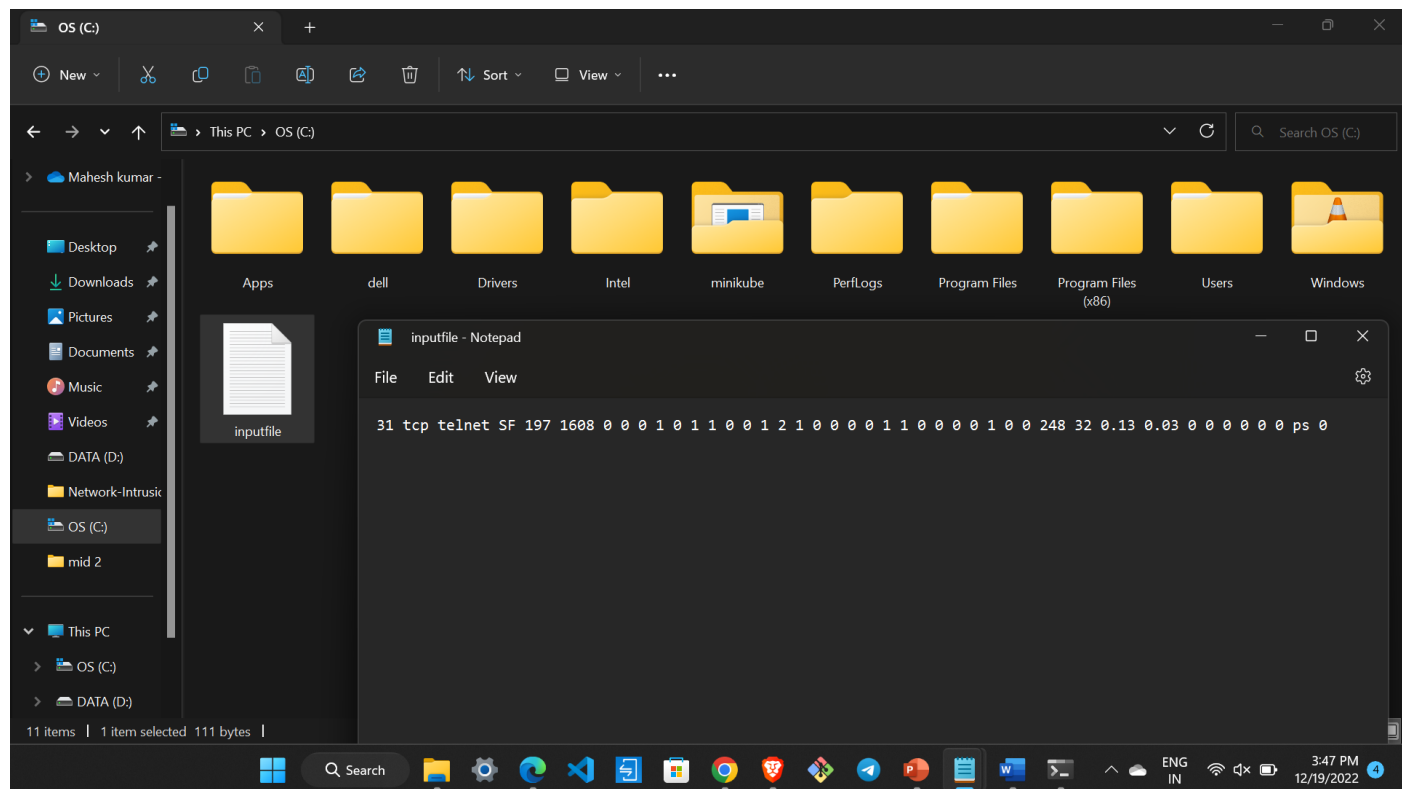
8) Here we have Frequently asked questions by the users.Move down.



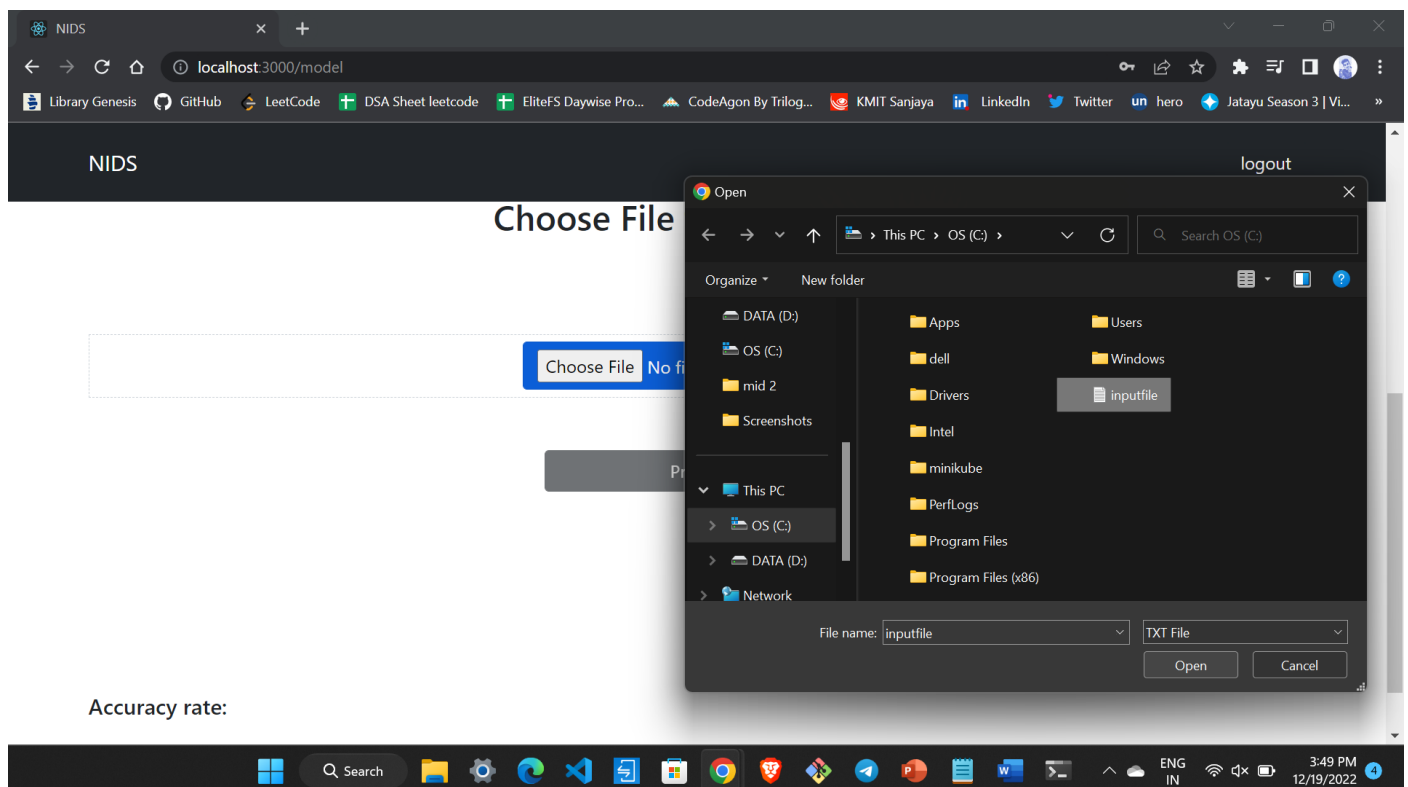
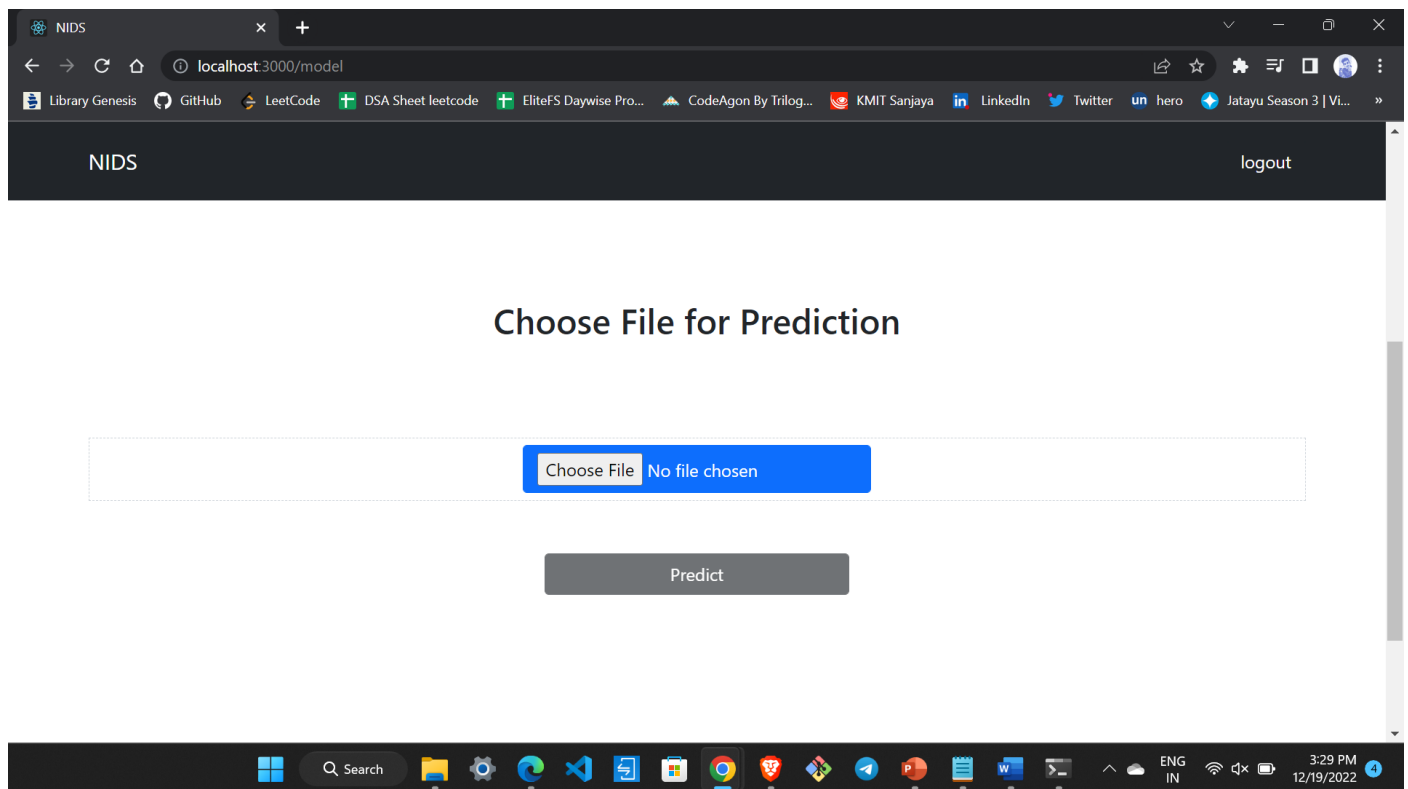
9) Now, here click on the button “click here to test” .

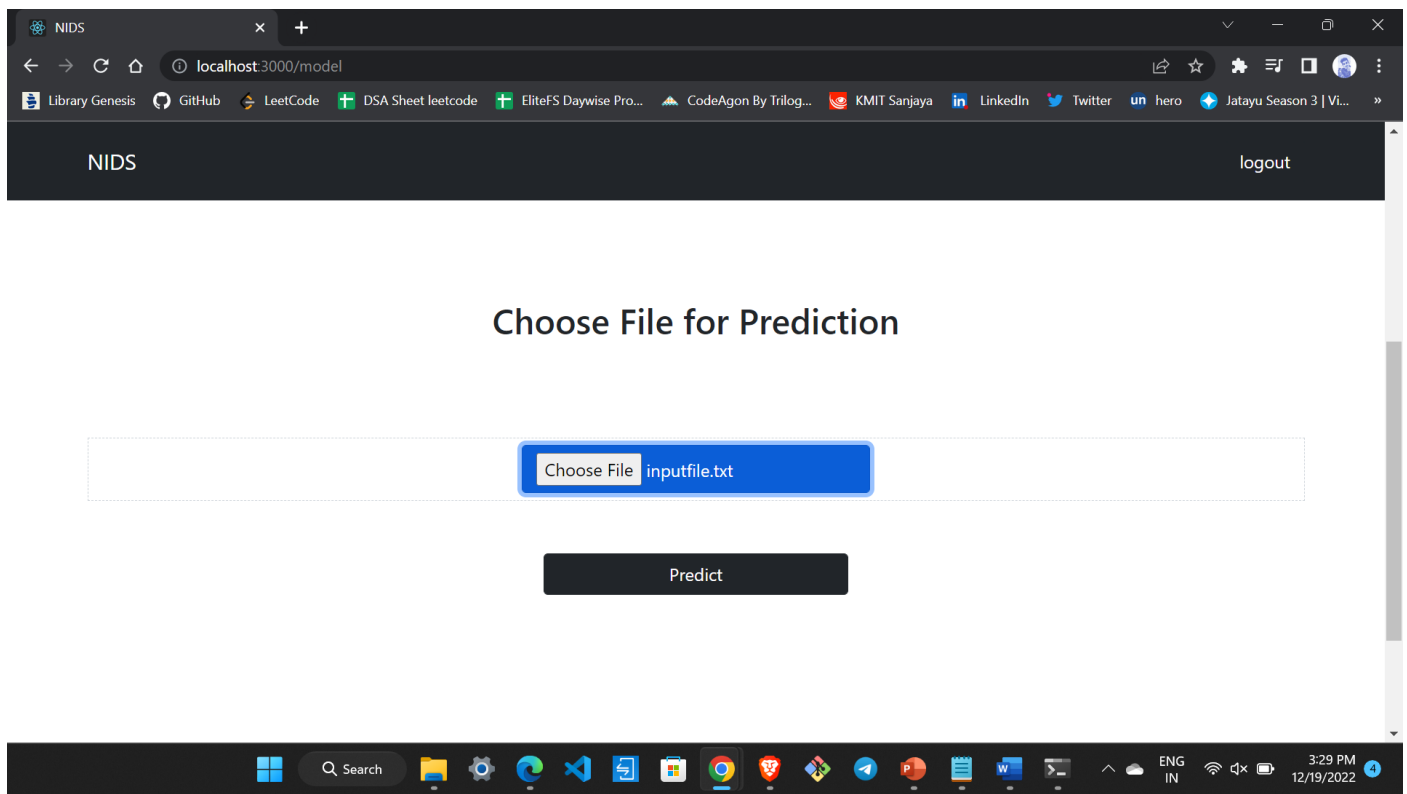


10) Now enter the all 43 parameters with space separated in a txt file with name “inputfile” and move it into the Local Disk-C and save it.

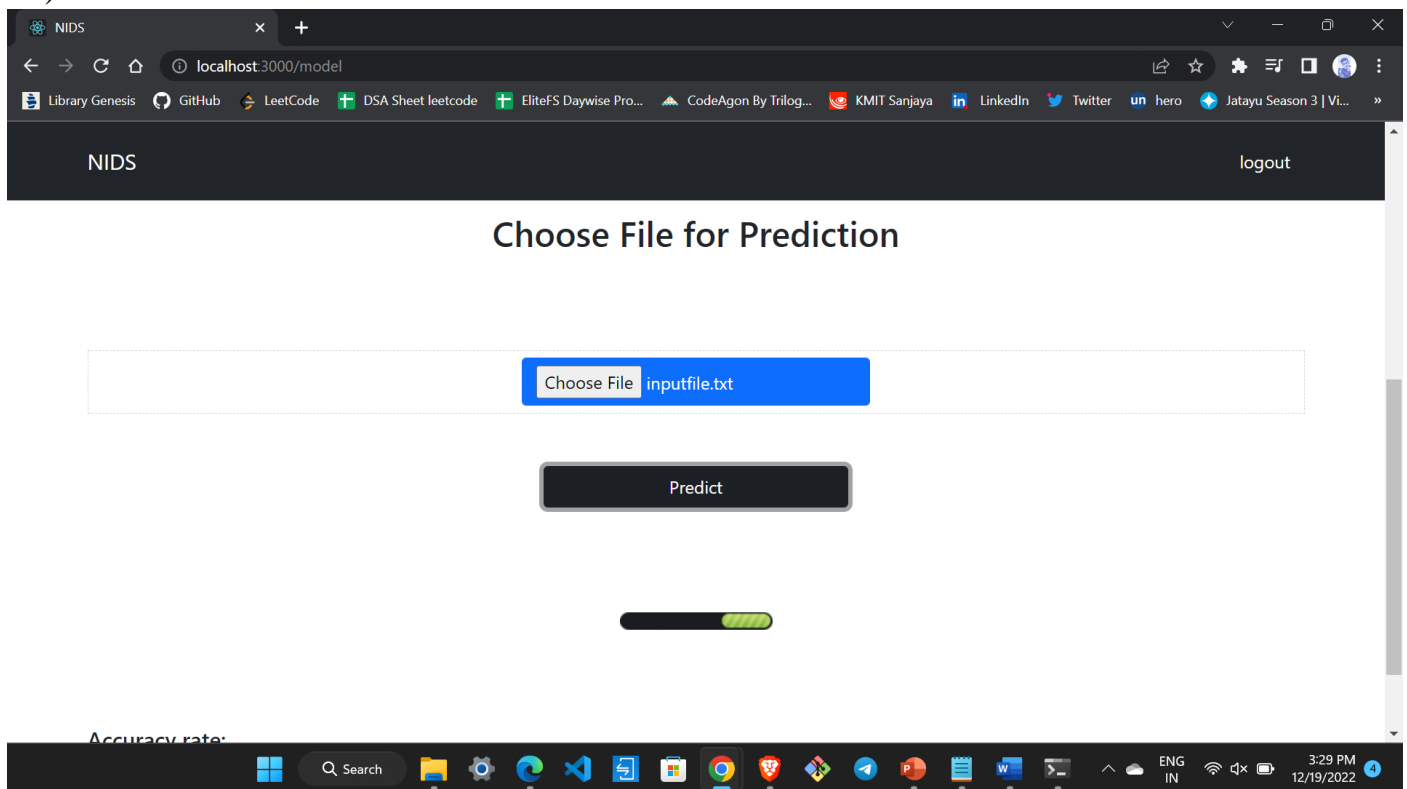


11) Now here, insert the file “inputfile.txt”

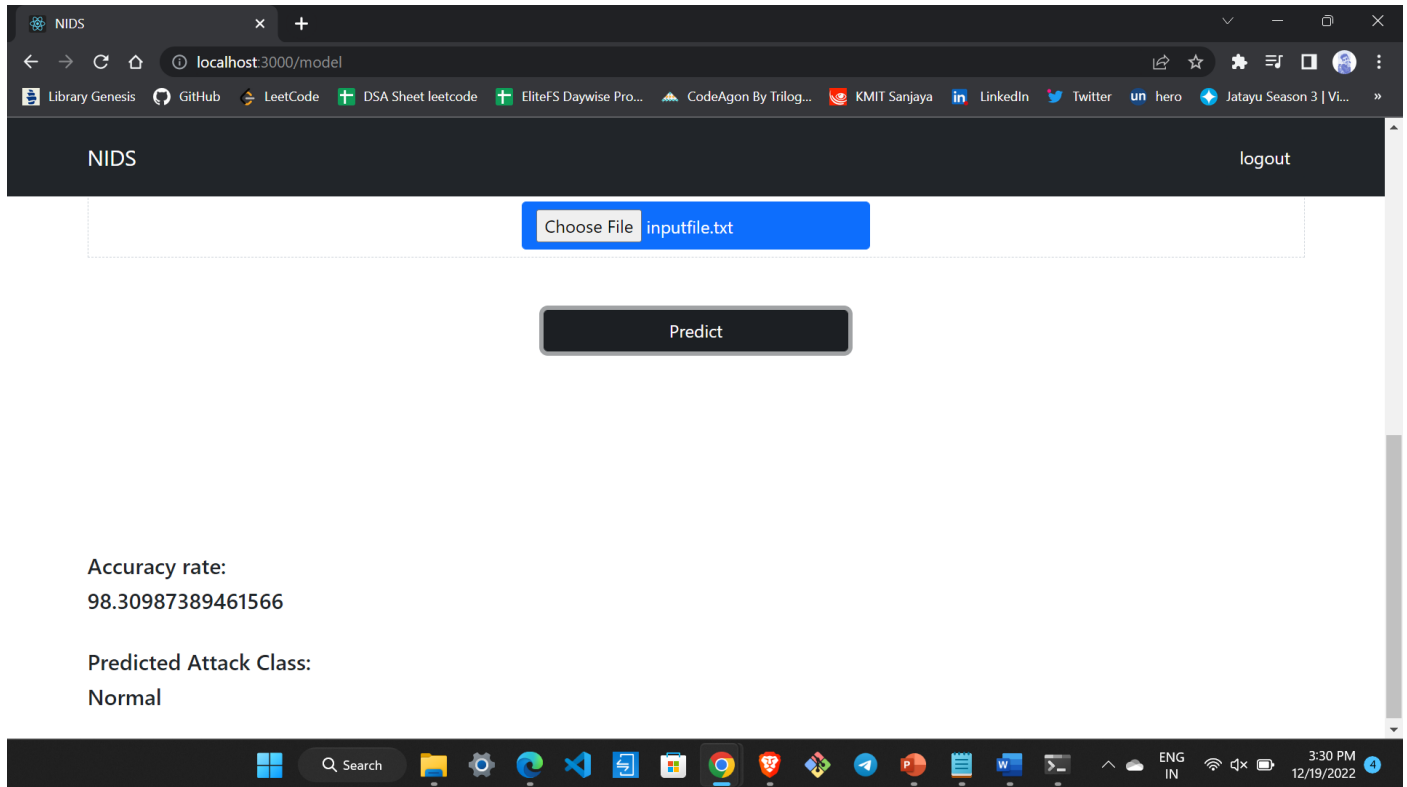




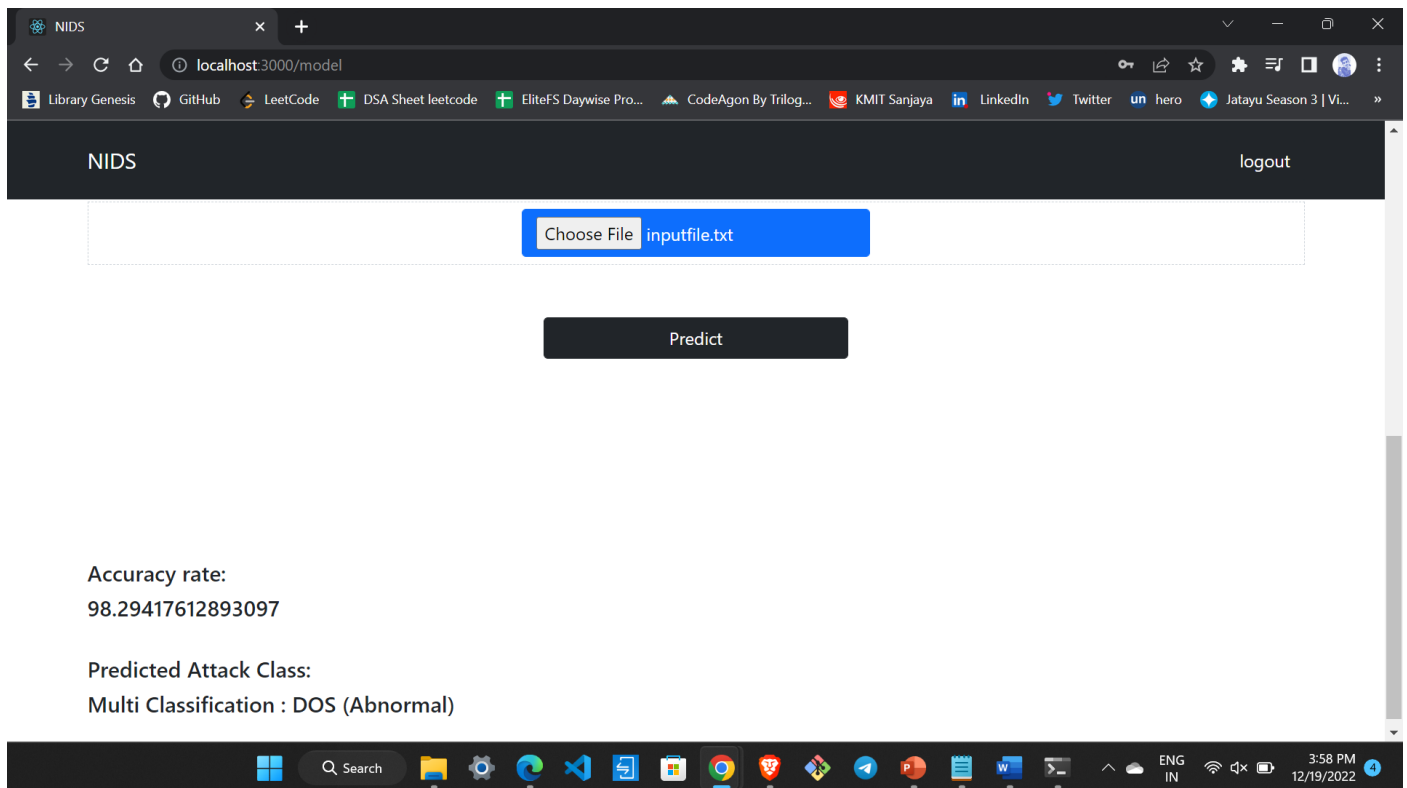
12) Now click on the button “Predict” and wait for the result.



13) Finally, we get the Accuracy rate with Predicted Attack Class .



14) Now go on and try any other inputfile for the accuracy rates and attack classes with the same above steps.



## Future Enhancement Plan :

- 1) We will try to enhance the frontend more beautifully and make it more user friendly for a user.
- 2) With large data, the prediction stage might take some extra time. We will try to decrease the speed of prediction time and provide the more efficient web browser for our users.

