A Mini Project Report on

SOFTWARE VULNERABILITY DETECTION TOOL USING MACHINE LEARNING ALGORITHMS

Submitted to the

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD

In partial fulfilment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE & ENGINEERING

(DATA SCIENCE)

By

(TEAM LEADER) B.CHIRANJEEVI	(21TP1A6704)
G.NAMDEV	(21TP1A6721)
R.ANIRUDH	(21TP1A6739)
S.SOHITH	(21TP1A6742)

Under the Guidance of

Mr. K L S D T KEERTHI VARDHAN(HOD CSE(DS))

Associate Professor

DEPARTMENT OF COMPUTER SCIENCE (DATA SCIENCE)

SIDDHARTHA INSTITUTE OF ENGINEERING AND TECHNOLOGY

(Approved by AICTE, Accredited by NBA & NAAC A+,Affiliated to JNTU,Hyderabad)Vinobhanagar(V),Ibrahimpatnam(M), R.R.Dist.501506. 2024-2025

SIDDHARTHA INSTITUTE OF ENGINEERING AND TECHNOLOGY

(Approved by AICTE, Accredited by NBA & NAAC A+, Affiliated to JNTU Hyderabad)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING (DATA SCIENCE)

CERTIFICATE

This is to certify that the project work entitled "**SOFTWARE VULNERABILITY DETECTION TOOL USING MACHINE LEARNING ALGORITHMS**" is a Bonafide work of **B. CHIRANJEEVI (21TP1A6704), G. NAMDEV (21TP1A6721), R.ANIRUDH (21TP1A6739), S.SOHITH (21TP1A6742)** submitted to **SIDDHARTHA INSTITUTE OF ENGINEERING AND TECHNOLOGY, HYDERABAD** in partial fulfillment of the requirement for the award of degree of **Bachelor of Technology In COMPUTER SCIENCE & ENGINEERING (DATA SCIENCE).** The work reported here in does not form part of any other thesis on which a degree has been awarded earlier. This is to further certify that they have worked for a period of one semester for preparing their work under our supervision and guidance

UNDER THE GUIDANCE OF	HEAD OF THE DEPARTMENT
MR. KLSDT KEERTHVARDHAN M.Tech,(Ph.D)Prof. HOD,CSE (DS), SIET	MR. KLSDT KEERTHVARDHAN M.Tech,(Ph.D) HOD,CSE (DS), SIET

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION BY THE CANDIDATES

We are **B. CHIRANJEEVI (21TP1A6704), G. NAMDEV (21TP1A6721), R. ANIRUDH (21TP1A6739), S. SOHITH (21TP1A6742)** hereby declare that the Project Report Entitled **"SOFTWARE VULNERABILITY DETECTION TOOL USING MACHINE LEARNING ALGORITHMS"** Under the guidance of Mrs. Kirthi Priya.B, MTech, Assistant Professor, Department of CSD is Submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science & Engineering (Data Science).**

This is a Record of Bonafide work carried out by us and the results embodied in this Project Report have not been reproduced or copied from any source. The results embodied in this Project Report have not submitted to any other University or Institute for the Award of any other Degree or Diploma.

B. CHIRANJEEVI	**(21TP1A6704)**
G. NAMDEV	**(21TP1A6721)**
R. ANIRUDH	**(21TP1A6739)**
S. SOHITH	**(21TP1A6742)**

Dept.of Computer Science of Engineering (Data Science)

Siddhartha Institute of Engineering and Technology.

Vinobhanagar (V), Ibrahimpatnam (M), R.R (Dist), Hyderabad

ACKNOWLEDGEMENT

An endeavor over a long period can be successful only with the advice and supports of many well-wishers. We take this opportunity to express our gratitude and appreciation to all of them.

We extremely thankful to our beloved Chairman **Dr. G. Nagaiah** who took keen interest and encourage us in every effort throughout this course.

We owe our gratitude to our principal **Dr. P. Sekhar Babu,** B.Tech, M.Tech, Ph.D. for permitting us to use the facilities available to accomplish the project successfully.

We express our heartfelt thanks **to Mr. KLSDT Keerthi Vardhan M.Tech., (Ph.D.)** Head of Dept – CSD for his kind attention and valuable guidance to us throughout this course.

We express our deep sense of gratitude towards **Mr. KLSDT Keerthi Vardhan M.Tech., (Ph.D)** Project guide, Dept. of **COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)** for her support and guidance in completing our project successfully.

We also thank all the teaching and non-teaching staff of the Dept. of **COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)**. For their support throughout our B.Tech. Course.

We express our heartful thanks to **My Parents** for their valuable support and encouragement in completion of my course. Also we express our heartful regards to my **Friends** for being supportive in completion of our project.

TABLE OF CONTENTS

1.ABSTRACT

Exploitable vulnerabilities in software have attracted tremendous attention in recent years because of their potentially high severity impact on computer security and information safety. Many vulnerability detection methods have been proposed to aid code inspection. Among these methods, there is a line of studies that apply machine learning techniques and achieve promising results. This paper reviews 22 recent studies that adopt deep learning to detect vulnerabilities, aiming to show how they utilize state of the art neural techniques to capture possible vulnerable code patterns. Among reviewed studies, we identify four game changers that significantly impact the domain of deep learning-based vulnerability detection and provide detailed reviews of the insights, ideas, and concepts that the game changers have brought to this field of interest. Based on the four identified game changers, we review the remaining studies, presenting their approaches and solutions which either build on or extend the game changers, and sharing our views on the future research trends. We also highlight the challenges faced in this field and discuss potential research directions. We hope to motivate the readers to conduct further research in this developing but fast-growing field.

2.INTRODUCTION

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be glory. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. exploited for nefarious purposes when discovered by malicious attackers. In some cases, an attacker can crash an important running program, leading to a DoS (denial of service). In other cases, the attacker can escalate his privileges or even achieve full control over the machine.

DEP (data execution prevention) makes the call stack non-executable, preventing hackers from being able to execute their payloads, and ASLR (address space layout randomization), randomizes the address space layout of the process, making it more difficult for hackers to insert correct addresses into their payloads. However, complex programs, particularly those written in a relatively low-level language like C, are difficult to scan for bugs, even when using both manual and automated techniques. Microsoft spends roughly 100 machine years per year using automated techniques to detect bugs in their code, but their products often contain numerous bugs, because complex pointer arithmetic can sometimes be difficult to follow, especially when the developers are under constant time pressure to meet their deadlines.

3. LITERATURE SURVEY

Exploration of Communication Networks from the Enron Email Corpus

The Enron email corpus is appealing to researchers because it is a) a largescale email collection from b) a real organization c) over a period of 3.5 years. In this paper we contribute to the initial investigation of the Enron email dataset from a social network analytic perspective. We report on how we enhanced and refined the Enron corpus with respect to relational data and how we extracted communication networks from it. We apply various network analytic techniques in order to explore structural properties of the networks in Enron and to identify key players across time. Our initial results indicate that during the Enron crisis the network had been denser, more centralized and more connected than during normal times. Our data also suggests that during the crisis the communication among Enron's employees had been more diverse with respect to people's formal positions, and that top executives had formed a tight clique with mutual support and highly brokered interactions with the rest of organization. The insights gained with the analyses we perform and propose are of potential further benefit for customizing the development of crisis scenarios in organizations and the investigation of indicators of failure.

3.1.RICH: Automatically Protecting Against Integer-Based Vulnerabilities

We present the design and implementation of RICH (Run-time Integer Checking), a tool for efficiently detecting integer-based attacks against C programs at run time. C integer bugs, a popular avenue of attack and frequent programming error [1–15], occur when a variable value goes out of the range of the machine word used to materialize it, e.g. when assigning a large 32-bit int to a 16-bit short. We show that safe and unsafe integer operations in C can be captured by well-known sub-typing theory. The RICH compiler extension compiles C programs to object code that monitors its own execution to detect integer-based attacks. We implemented RICH as an extension to the GCC compiler and tested it on several network servers and UNIX utilities. Despite the ubiquity of integer operations, the performance overhead of RICH is very low, averaging about 5%. RICH found two new integer bugs and caught all but one of the previously known bugs we tested.

These results show that RICH is a useful and lightweight software testing tool and run-time mechanism. RICH may generate false positives when programmers use integer overflows deliberately, and it can miss some integer bugs because it does not model certain C features.

3.2.CSSV: Towards a realistic tool for statically detecting all buffer overflows in C

Erroneous string manipulations are a major source of software defects in C programs yielding vulnerabilities which are exploited by software viruses. We present C String Static Verifier (CSSV), a tool that statically uncovers all string manipulation errors. Being a conservative tool, it reports all such errors at the expense of sometimes generating false alarms. Fortunately, only a small number of false alarms are reported, thereby proving that statically reducing software vulnerability is achievable. CSSV handles large programs by Analysis of each procedure separately. To this end procedure contracts are allowed which are verified by the tool. We implemented a CSSV prototype and used it to verify the absence of errors in real code from EADS Airbus. When applied to another commonly used string intensive application, CSSV uncovered real bugs with very few false alarms. Improving security using extensible lightweight static analysis.

3.3.Automated Whitebox Fuzz Testing

Fuzz testing is an effective technique for finding security vulnerabilities in software. Traditionally, fuzz testing tools apply random mutations to well-formed inputs of a pro- gram and test the resulting values. We present an alternative white box fuzz testing approach inspired by recent advances in symbolic execution and dynamic test generation. Our approach records an actual run of the program un- der test on a well-formed input, symbolically evaluates the recorded trace, and gathers constraints on inputs capturing how the program uses these. The collected constraints are then negated one by one and solved with a constraint solver, producing new inputs that exercise different control paths in the program. This process is repeated with the help of a codecoverage maximizing heuristic designed to find defects as fast as possible. We have implemented this algorithm in SAGE (Scalable, Automated, Guided Execution), a new tool employing x86 instruction-level tracing and emulation for white box fuzzing of arbitrary file-reading Windows ap- plications. We describe key optimizations needed to make dynamic test generation scale to large input files and long execution traces with hundreds of millions of instructions. We then present detailed experiments with several Windows applications. Notably, without any format-specific knowledge, SAGE detects the MS07-017 ANI vulnerability, which was missed by extensive black box fuzzing and static analysis tools. Furthermore, while still in an early stage of development, SAGE has already discovered 30+ new bugs in large shipped Windows applications including image processors, media players, and file decoders. Several of these bugs are potentially exploitable memory access violations.

4.EXISTING SYSTEM

In Existing system, we are using ,machine learning algorithms like k-means, random forest and decision tree to develop vulnerability detection tool

Disadvantages

1.Less accuracy.

5.PROPOSED SYSTEM

In proposed system, we are using Ensemble Machine Learning algorithm which is combination of multiple algorithms such as SVM, KNN and Naïve Bayes.

Now-a-days Machine Learning algorithms are using everywhere from Medical disease prediction to road side traffic prediction as this algorithms prediction accuracy is more than 95%.

In propose work we are using dataset to identify 3 different classes such as ‘No Vulnerability, SQL Injection, XSS or RFI.

Modules:

To implement this project we have designed following Modules

1) New User Register: new user can register with the application
2) User Login: after sign up user can login to application
3) Load Dataset: after login user can upload dataset to application and then extract all queries and labels from dataset and then from all queries will remove stop words like ‘and, the, or, what and many other words’. By removing stop words application will have core queries words. Dataset processing for core words will be happened using Natural Language processing toolkit
4) Run Ensemble Algorithms: processed dataset will be input to Ensemble Machine learning algorithm to train a model and this model will be applied on test data to calculate accuracy and other metrics
5) Confusion Matrix Graph: using this module we will plot confusion matrix graph of algorithm prediction capability

6) Predict Vulnerability: using this module will upload new TEST data query and then Machine learning algorithm will analysis all TEST data and predict type of vulnerability.
7) All eight projects were developed using a sequential life cycle model, a well-known OO analysis/design method and the C++ programming language

6. PROCESS MODEL USED WITH JUSTIFICATION

6.1.SDLC (Umbrella Model):

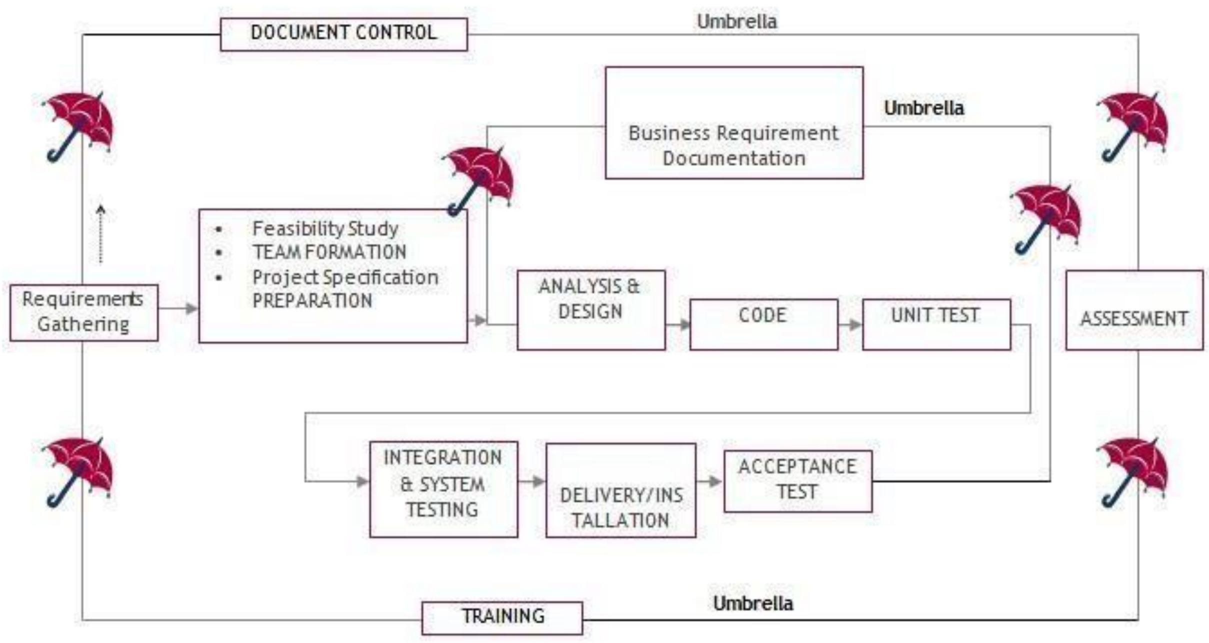


Fig. SDLC(Umbrella Model)

The requirements gathering process takes as its input SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

6.2.Stages in SDLC:

- Requirement Gathering
- Analysis
- Designing
- Coding Testing Maintenance

6.2.1.Requirements Gathering stage:

The goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.

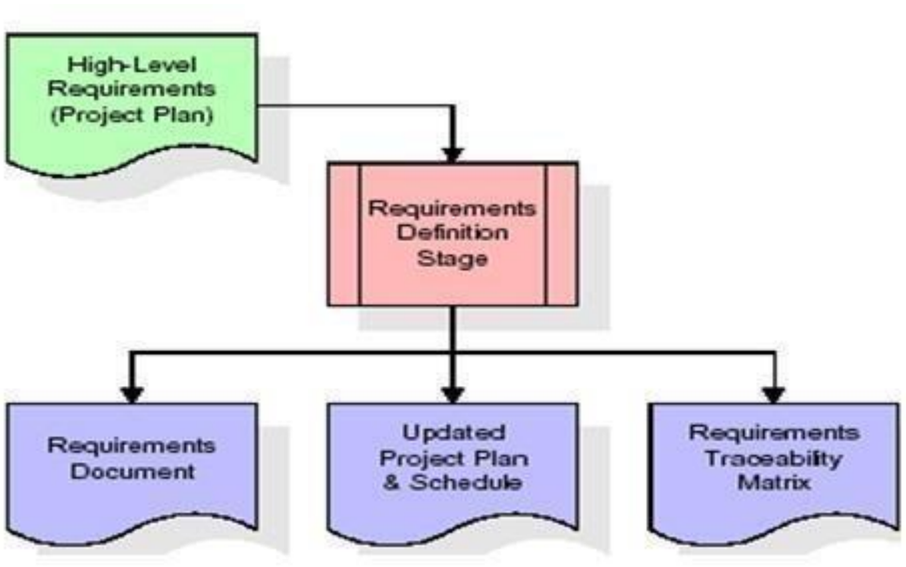

High-Level Requirements (Project Plan)
Requirements Definition Stage
Requirements Document
Updated Project Plan & Schedule
Requirements Traceability Matrix

Fig. Requirements Gathering stage

These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are *not* included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

- Feasibility study is all about identification of problems in a project.
- No. of staff required to handle a project is represented as Team Formation, in this case only modules are individual tasks will be assigned to employees who are working for that project.
- Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator.

6.2.2.Analysis Stage:

The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.

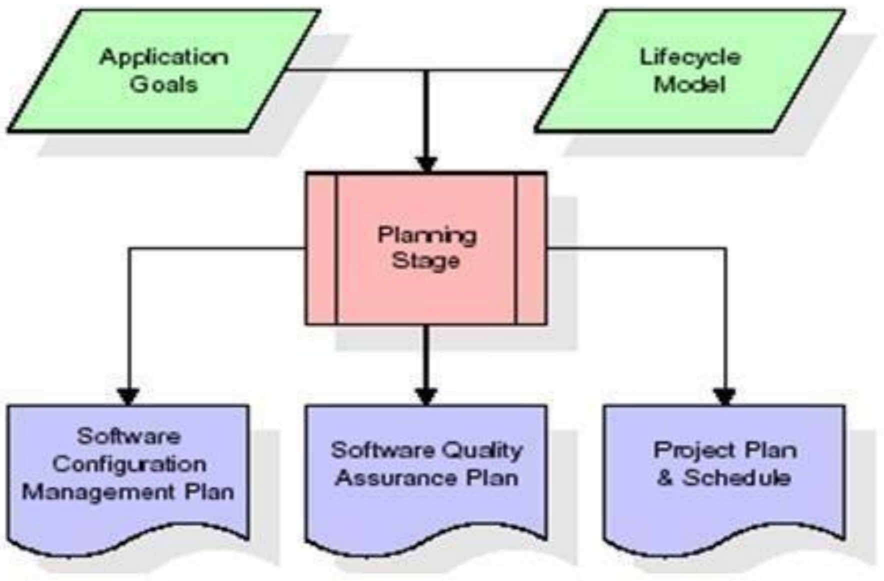

Application Goals
Lifecycle Model
Planning Stage
Software Configuration Management Plan
Software Quality Assurance Plan
Project Plan & Schedule

Fig. Analysis Stage

The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

6.2.3.Designing Stage:

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.

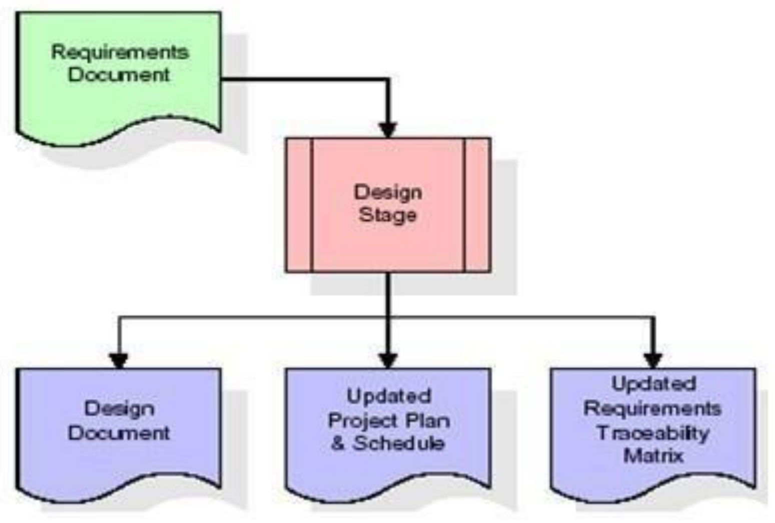

Requirements Document
Design Stage
Design Document
Updated Project Plan & Schedule
Updated Requirements Traceabilty Matrix

Fig. Designing Stage

When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

6.2.4.Development (Coding) Stage:

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, and data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.

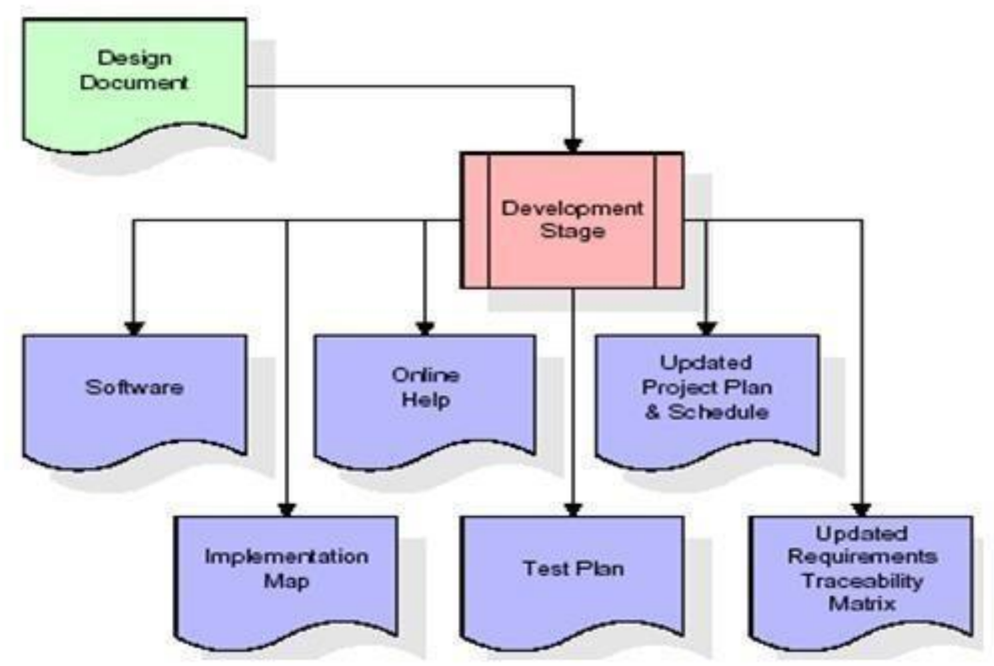


Fig. Development (Coding) Stage

The RTM will be updated to show that each developed artifact is linked to a specific design element, and that each developed artifact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

6.2.5.Integration & Test Stage:

During the integration and test stage, the software artifacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability. During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.

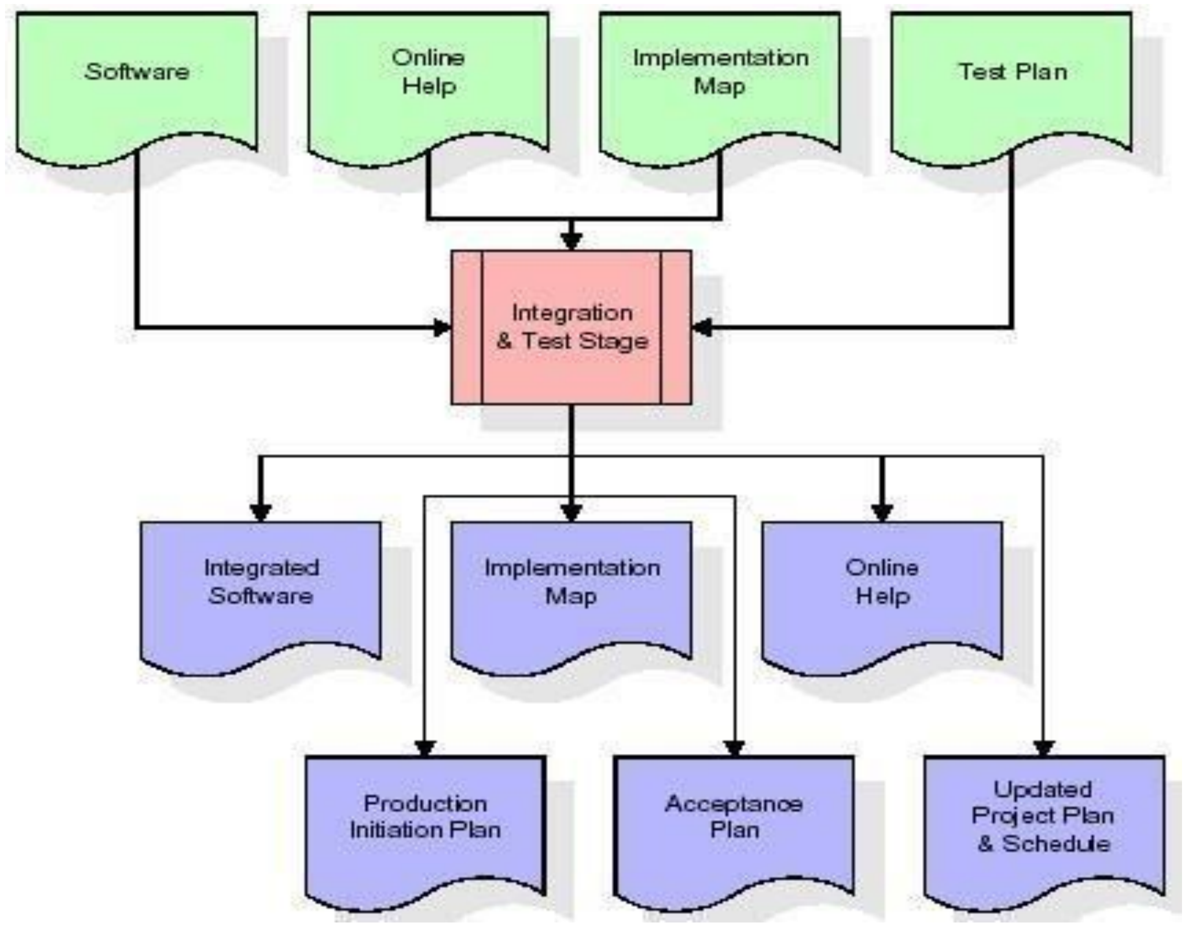


Fig. Integration & Test Stage

The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

- **Installation & Acceptance Test:**

During the installation and acceptance stage, the software artifacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.

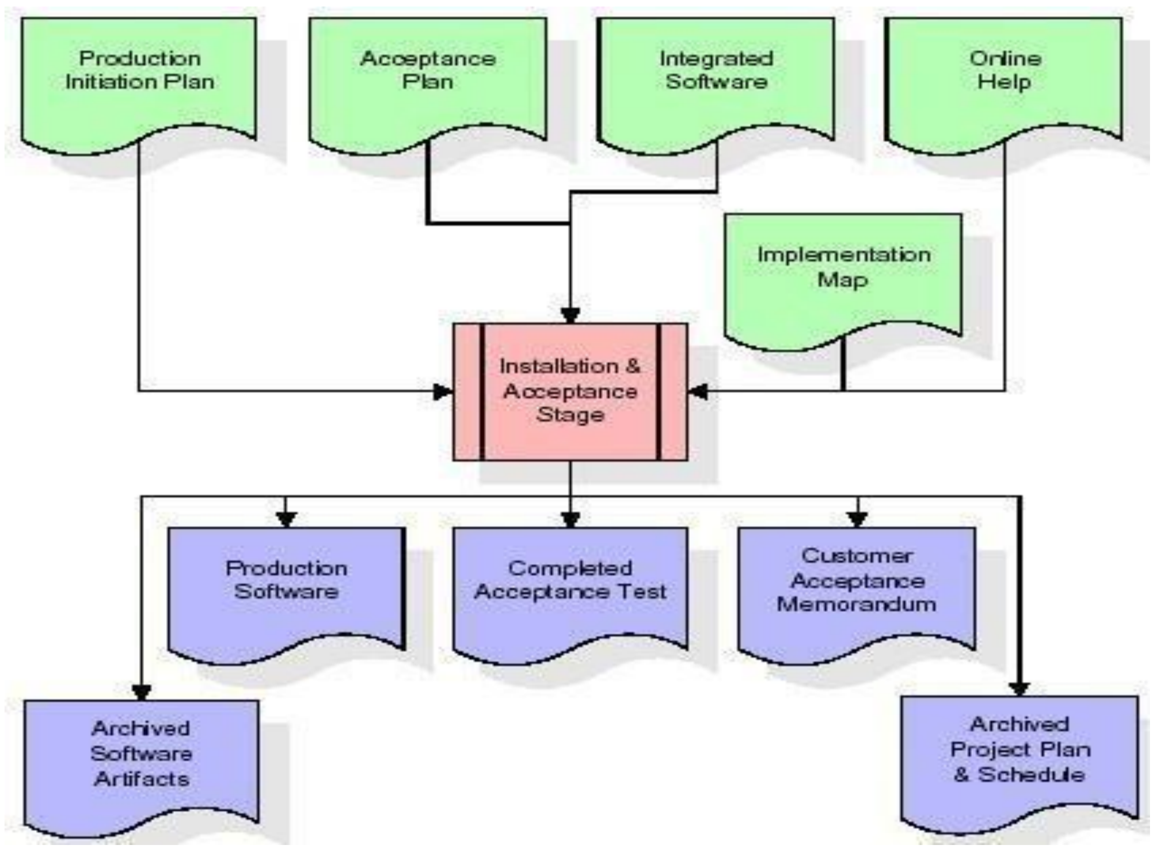

Production Initiation Plan
Acceptance Plan
Integrated Software
Online Help
Implementation Map
Installation & Acceptance Stage
Production Software
Completed Acceptance Test
Customer Acceptance Memorandum
Archived Software Artifacts
Archived Project Plan & Schedule

Fig. Installation & Acceptance Test

The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labour data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

6.2.6.Maintenance:

- Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation later employees will be assigned work and they will undergo training on that particular assigned category. For this life cycle there is no end, it will be continued so on like an umbrella (no ending point to umbrella sticks.)

7.SOFTWARE REQUIREMENT SPECIFICATION

7.1. Overall Description

A Software Requirements Specification (SRS) for a system is a complete description of the behaviour of a system to be developed. It includes a set of use cases that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. Nonfunctional requirements are requirements which impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A business analyst, sometimes titled, is responsible system analyst for the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the systems development lifecycle domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

- Business requirements describe in business terms what must be delivered or accomplished to provide value.
- Product requirements describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)
- Process requirements describe activities performed by the developing organization. For instance, process requirements could specify. Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation.

• ECONOMIC FEASIBILITY

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, there is nominal expenditure and economical feasibility for certain.

• OPERATIONAL FEASIBILITY

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. This system is targeted to be in accordance with the abovementioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would

Ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

- **TECHNICAL FEASIBILITY**

Earlier no system existed to cater to the needs of 'Secure Infrastructure Implementation System'. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to the users. The database's purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security.

8. SYSTEM DESIGN

8.1.UML Diagram:

Class Diagram:

The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes which contain three parts:

- The upper part holds the name of the class
- The middle part contains the attributes of the class
- The bottom part gives the methods or operations the class can take or undertake.

Use case Diagram:

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.

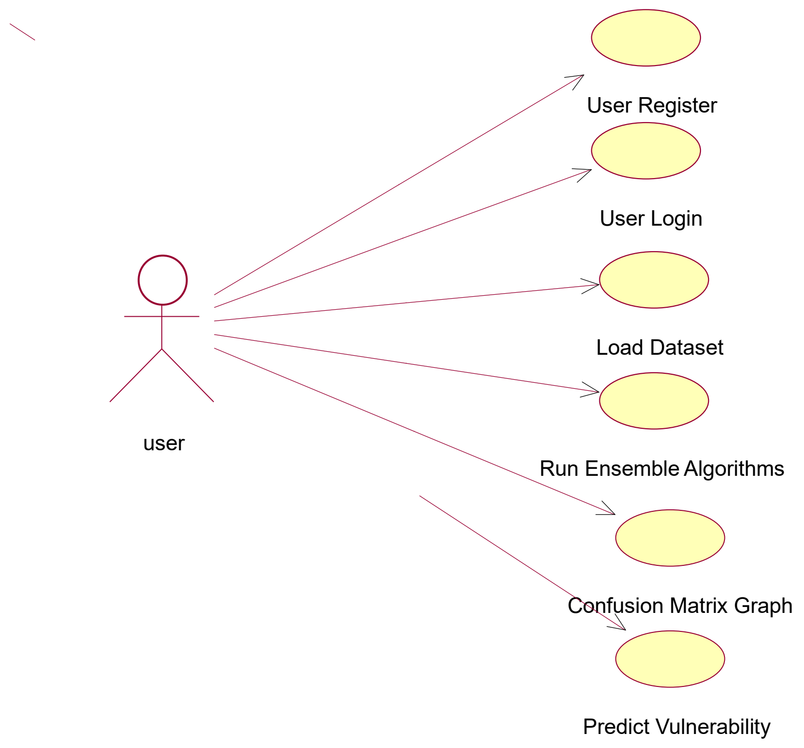
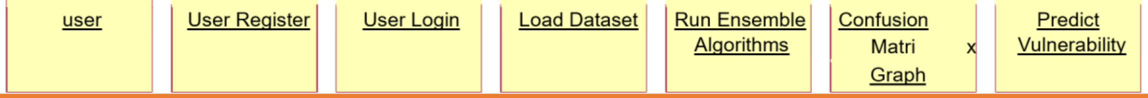


Fig. Class Diagram

Sequence Diagram:

A sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



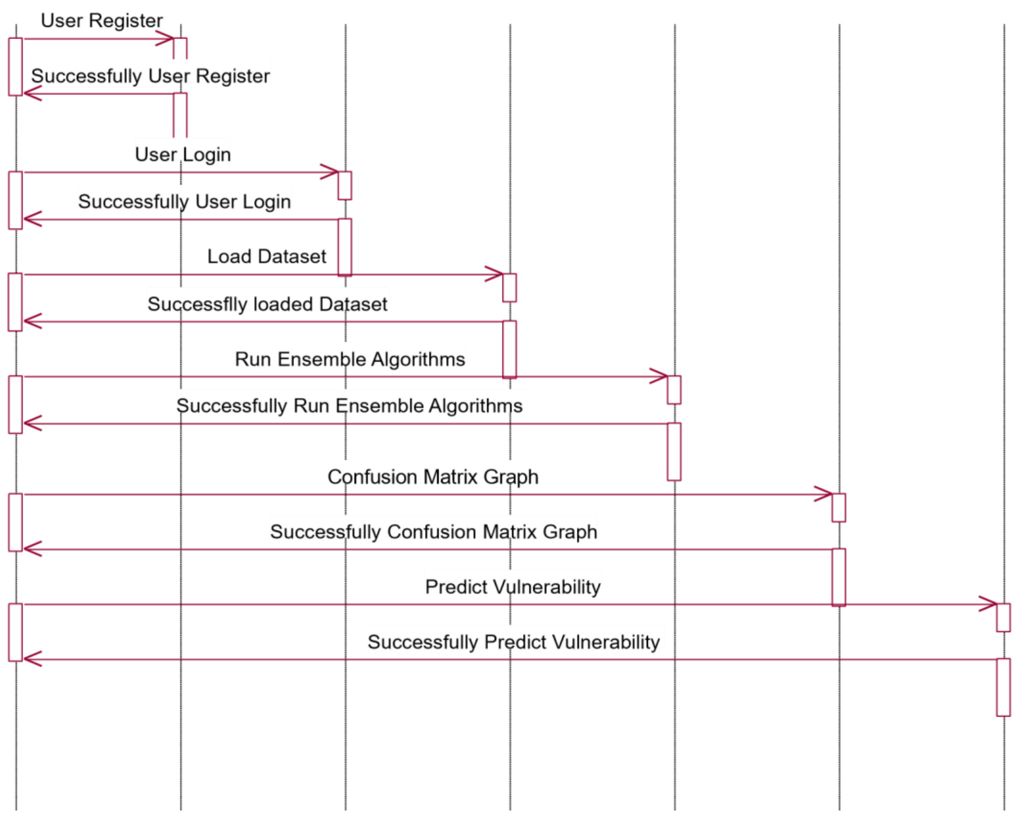

User Register
Successfully User Register
User Login
Successfully User Login
Load Dataset
Successflly loaded Dataset
Run Ensemble Algorithms
Successfully Run Ensemble Algorithms
Confusion Matrix Graph
Successfully Confusion Matrix Graph
Predict Vulnerability
Successfully Predict Vulnerability

Fig. Sequence Diagram

Collaboration diagram:

A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behaviour of a system.

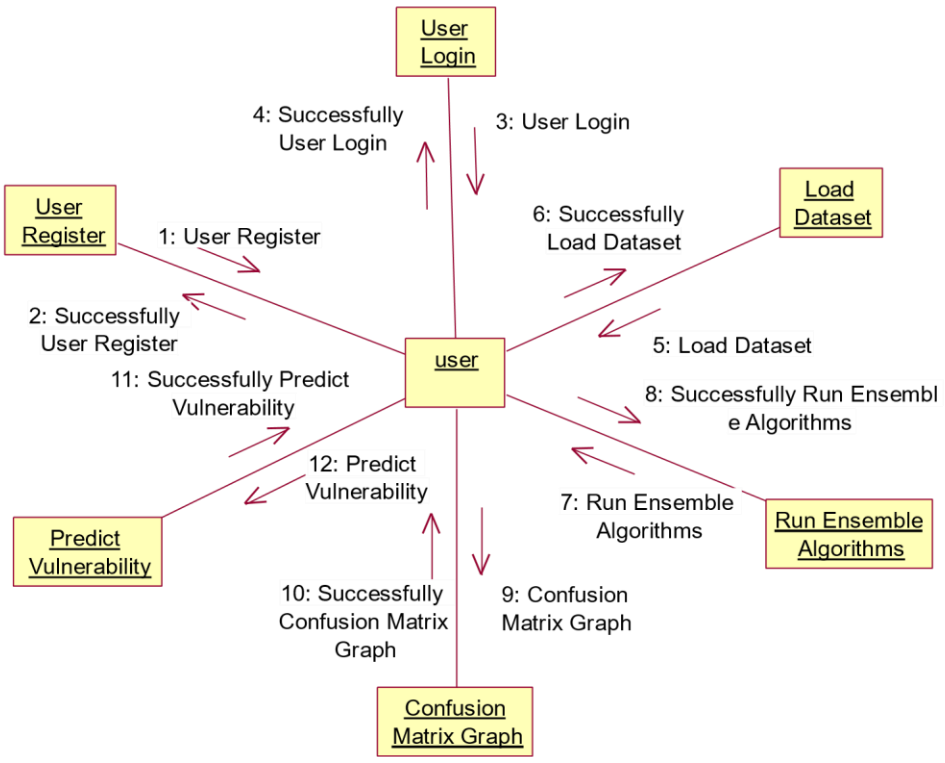

User Login
4: Successfully User Login
3: User Login
User Register
1: User Register
2: Successfully User Register
6: Successfully Load Dataset
Load Dataset
5: Load Dataset
user
11: Successfully Predict Vulnerability
8: Successfully Run Ensemble Algorithms
12: Predict Vulnerability
7: Run Ensemble Algorithms
Run Ensemble Algorithms
Predict Vulnerability
10: Successfully Confusion Matrix Graph
9: Confusion Matrix Graph
Confusion Matrix Graph

Fig. Collaboration Diagram

Component Diagram:

In the Unified Modelling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer - service provider relationship between the two components.

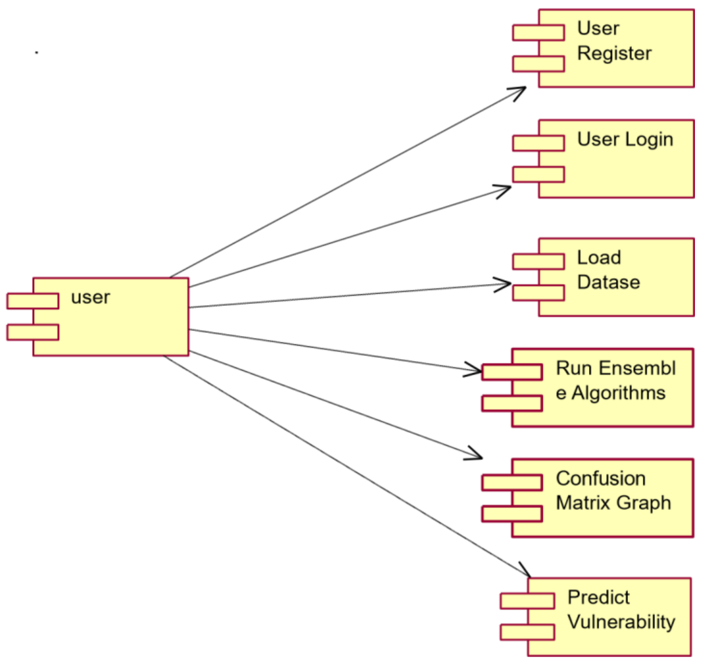


Fig. Component Diagram

Deployment Diagram:

A deployment diagram in the Unified Modeling Language models the *physical* deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).

The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.

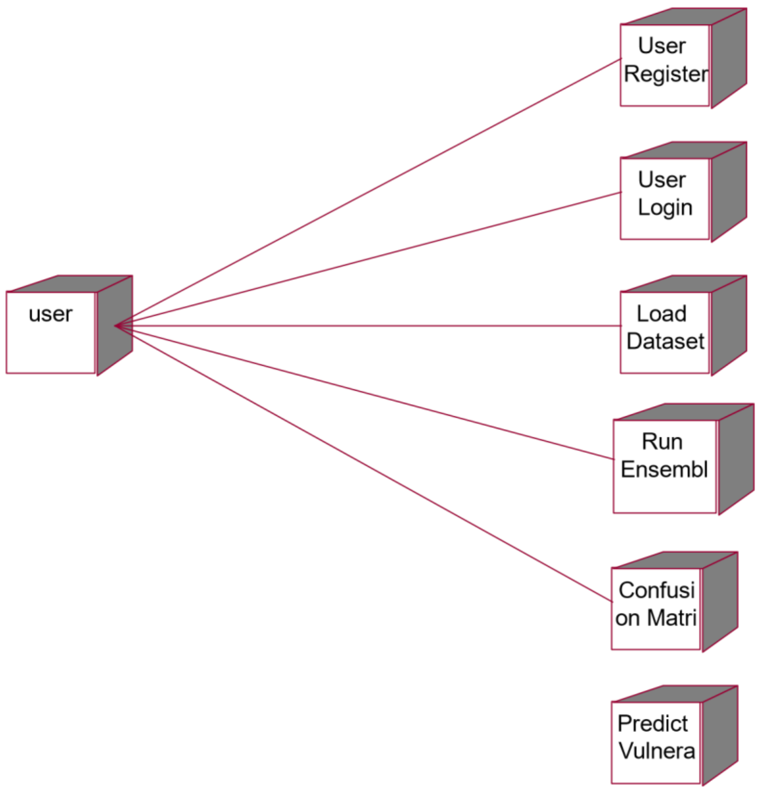


Fig. Deployment Diagram

Activity Diagram:

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent.

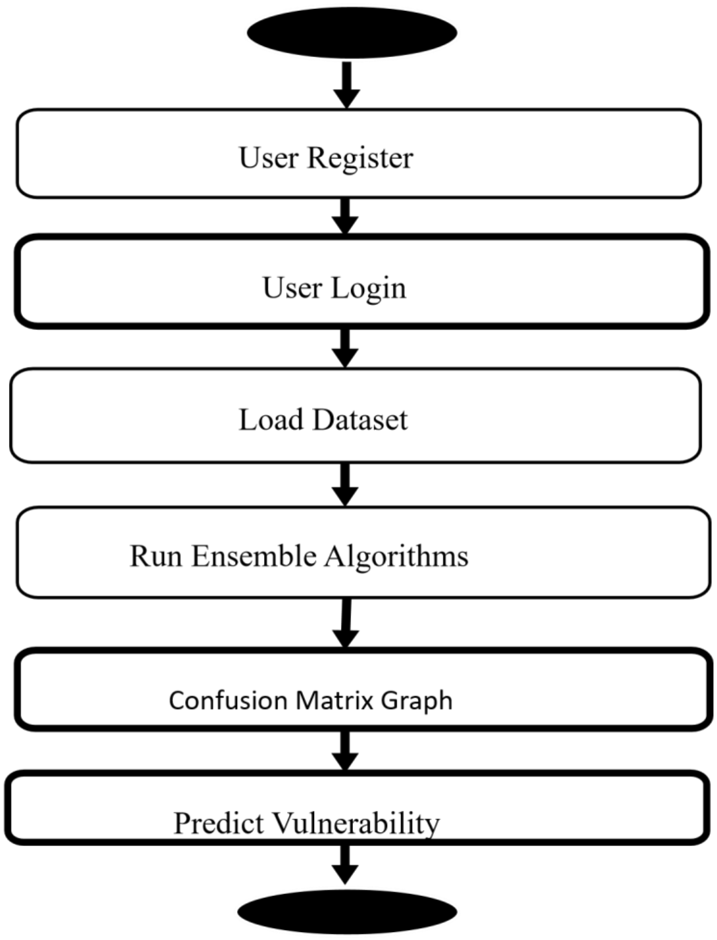


Fig. Activity Diagram

Data Flow Diagram:

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Data flow diagrams can be used to provide a clear representation of any business function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. This analysis can be carried out in precisely the level of detail required. The technique exploits a method called top-down expansion to conduct the analysis in a targeted way.

As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy-to-use, free downloadable diagramming tools. A DFD is a model for constructing and analyzing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred to as a Process Model. A DFD demonstrates business or technical process with the support of the outside data saved, plus the data flowing from the process to another and the end results.

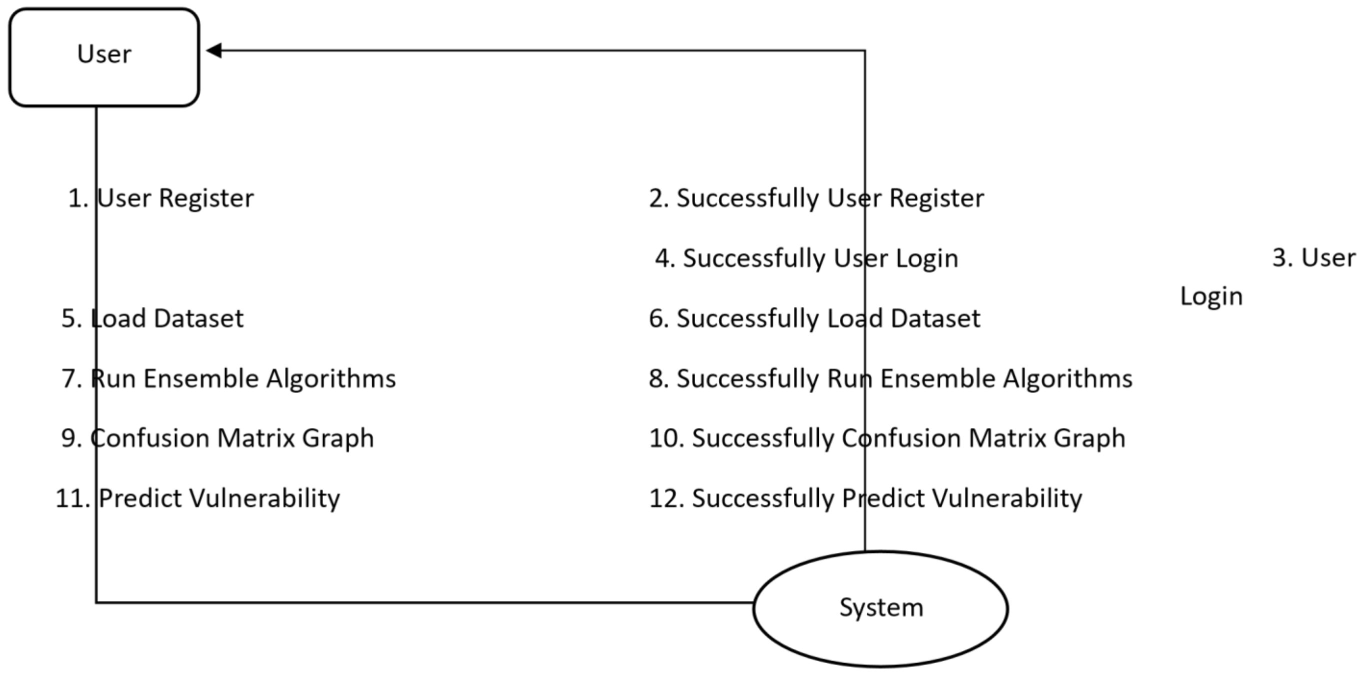

User
1. User Register
2. Successfully User Register
3. User Login
4. Successfully User Login
5. Load Dataset
6. Successfully Load Dataset
7. Run Ensemble Algorithms
8. Successfully Run Ensemble Algorithms
9. Confusion Matrix Graph
10. Successfully Confusion Matrix Graph
11. Predict Vulnerability
12. Successfully Predict Vulnerability
System

Fig. DataFlow Diagram

9. IMPLEMETATION

9.1 PYTHON

* One of the most popular languages is Python. Guido van Rossum released this language in 1991. Python is available on the Mac, Windows, and Raspberry Pi operating systems. The syntax of Python is simple and identical to that of English. When compared to Python, it was seen that the other language requires a few extra lines.

*It is an interpreter-based language because code may be run line by line after it has been written. This implies that rapid prototyping is possible across all platforms. Python is a big language with a free, binary-distributed interpreter standard library.

* It is inferior to maintenance that is conducted and is straightforward to learn. It is an object-oriented, interpreted programming language. It supports several different programming paradigms in addition to objectoriented programming, including functional and procedural programming.

* It supports several different programming paradigms in addition to object-oriented programming, including practical and procedural programming. Python is mighty while maintaining a relatively straightforward syntax. Classes, highly dynamic data types, modules, and exceptions are covered. Python can also be utilised by programmes that require programmable interfaces as an external language.

Here are some key features and characteristics of Python:

- Readability: Python emphasizes code readability with its clean and intuitive syntax. It uses indentation and whitespace to structure code blocks, making it easy to understand and maintain.
- Easy to Learn: Python's simplicity and readability make it an excellent choice for beginners. Its straightforward syntax and extensive documentation make it accessible for newcomers to programming.

- Interpreted Language: Python is an interpreted language, meaning that it doesn't need to be compiled before running. The Python interpreter reads and executes the code directly, making the development process faster and more interactive.
- Cross-platform Compatibility: Python is available for major operating systems like Windows, macOS, and Linux. This cross-platform compatibility allows developers to write code once and run it on different platforms without modifications.
- Large Standard Library: Python comes with a vast standard library that provides ready-to-use modules and functions for various tasks. It covers areas such as file I/O, networking, regular expressions, databases, and more, saving developers time and effort.
- Extensible and Modular: Python supports modular programming, enabling developers to organize code into reusable modules and packages. Additionally, Python allows integrating modules written in other languages, such as C or C++, providing flexibility and performance optimizations.
- Wide Range of Libraries and Frameworks: Python has a vibrant ecosystem with numerous third-party libraries and frameworks. These libraries, such as NumPy, pandas, TensorFlow, and Django, extend Python's capabilities for specific domains, making it a powerful tool for diverse applications.
- Object-Oriented: Python supports object-oriented programming (OOP) principles, allowing developers to create and work with classes and objects. OOP provides a structured approach to code organization, promoting code reuse and modularity.
- Dynamic Typing: Python is dynamically typed, meaning variable types are determined at runtime. Developers do not need to declare variable types explicitly, which enhances flexibility and simplifies code writing.

9.2 Installation

To install Python on your computer, follow these basic steps:

- Step 1: Visit the Python website Go to the official Python website at https://www.python.org/.
- Step 2: Select the operating system Choose the appropriate installer for your operating system. Python supports Windows, macOS, and various Linux distributions. Make sure to select the correct version that matches your operating system.
- Step 3: Check which version of Python is installed; if the 3.7.0 version is not there, uninstall it through the control panel and
- Step 4: Install Python 3.7.0 using Cmd.
- Step 5: Install the all libraries that required to run the project Step 6: Run

9.3 Python Features:

1) **Easy:** Because Python is a more accessible and straightforward language, Python programming is easier to learn.
2) **Interpreted language:** Python is an interpreted language, therefore it can be used to examine the code line by line and provide results.
3) **Open Source:** Python is a free online programming language since it is open-source.
4) **Portable:** Python is portable because the same code may be used on several computer standard
5) **libraries:** Python offers a sizable library that we may utilize to create applications quickly.
6) **GUI:** It stands for GUI (Graphical User Interface)
7) **Dynamical typed:** Python is a dynamically typed language, therefore the type of the value will be determined at runtime.

9.4 Python GUI (Tk inter)

- Python provides a wide range of options for GUI development (Graphical User Interfaces).
- Tk inter, the most widely used GUI technique, is used for all of them.
- The Tk GUI toolkit offered by Python is used with the conventional Python interface.
- Tk inter is the easiest and quickest way to write Python GUI programs.
- Using Tk inter, creating a GUI is simple.
- A part of Python's built-in library is Tk inter. The GUI programs were created.
- Python and Tk inter together give a straightforward and quick way. The Tk GUI toolkit's objectoriented user interface is called Tk inter.

Making a GUI application is easy using Tk inter. Following are the steps:

1) Install the Tk inter module in place.

2) The GUI application Makes the primary window

3) Include one or more of the widgets mentioned above in the GUI application.

4) Set up the main event loop such that it reacts to each user-initiated event.

Although Tk inter is the only GUI framework included in the Python standard library, Python includes a GUI framework. The default library for Python is called Tk inter. Tk is a scripting language often used in designing, testing, and developing GUIs. Tk is a free, open-source widget toolkit that may be used to build GUI applications in a wide range of computer languages.

9.5 Python IDLE

- ✞ Python IDLE offers a full-fledged file editor, which gives you the ability to write and execute Python programs from within this program. The built-in file editor also includes several features, like code completion and automatic indentation, that will speed up your coding workflow.

- ✞ Guido Van Rossum named Python after the British comedy group Monty Python while the name IDLE was chosen to pay tribute to Eric Idle, who was one of the Monty Python's founding members. IDLE comes bundled with the default implementation of the Python language since the 01.5. 2b1 release
- ✞ IDLE is used to execute statements similar to Python Shell. IDLE is used to create, modify, and execute Python code. IDLE provides a fully-featured text editor to write Python scripts and provides features like syntax highlighting, auto-completion, and smart indent.
- ✞ IDLE has two modes: interactive and script. We wrote our first program, “Hello, World!” in interactive mode. Interactive mode immediately returns the results of commands you enter into the shell. In script mode, you will write a script and then run it.
- ✞ The IDE Python IDLE is a good place to start as it helps you become familiar with the way Python works and understand its syntax. This IDE is good to start programming in Python due to its great debugger, but once you are fluent and start developing projects it is necessary to jump to another, more complete IDE.
- ✞ Python IDLE (Integrated Development and Learning Environment) is an interactive development environment included with the Python programming language. It provides a convenient way to write, execute, and debug Python code.

When you install Python, IDLE is typically installed along with it. To open IDLE, you can follow these steps:

- ○ Open the command prompt (Windows) or terminal (macOS/Linux). ○ Type "idle" and press Enter. Alternatively, you can specify the version with "idle3" or "idle2" for Python 3 or Python 2, respectively.
- ○ Once IDLE is launched, you will see the Python shell, which is an interactive environment where you can type and execute Python code directly.

Here are some features and functionalities provided by Python IDLE:

- Editor: IDLE includes a text editor where you can write your Python code. It offers syntax highlighting, automatic indentation, and code completion to enhance your coding experience.
- Interactive Shell: The Python shell in IDLE allows you to execute Python code interactively. You can type commands, statements, or function calls directly in the shell, and Python will execute them immediately.
- Debugging: IDLE provides basic debugging capabilities to help you find and fix errors in your code. You can set breakpoints, step through code, inspect variables, and track the program's execution.
- Python Help: IDLE provides access to the Python documentation and built-in help. You can access the help menu to find information about Python modules, functions, classes, and more.
- Script Execution: In addition to the interactive shell, IDLE allows you to run Python scripts stored in files. You can write your code in the editor and execute it as a script to see the output or interact with the program.
- Customization: IDLE can be customized to suit your preferences. You can modify settings related to syntax highlighting, indentation, fonts, and more.
- Python IDLE serves as a beginner-friendly development environment and learning tool. It is suitable for writing small scripts, testing code snippets, experimenting with Python features, and learning the language's basics. However, for more advanced development projects, you may consider using other code editors or integrated development environments (IDEs) that provide additional features and better project management capabilities.

9.6 Libraries

In Python, libraries (also referred to as modules or packages) are collections of pre-written code that provide additional functionality and tools to extend the capabilities of the Python language. Libraries contain reusable code that developers can leverage to perform specific tasks without having to write everything from scratch.

Python libraries are designed to solve common problems, such as handling data, performing mathematical operations, interacting with databases, working with files, implementing networking protocols, creating graphical user interfaces (GUIs), and much more. They provide ready-to-use functions, classes, and methods that simplify complex operations and save development time.

Libraries in Python offer various advantages:

- Code Reusability:
- Efficiency:
- Collaboration
- Domain-Specific Functionality
- To use a Python library, you need to install it first.

There are some libraries following:

Pandas:

Pandas are a Python computer language library for data analysis and manipulation. It offers a specific operation and data format for handling time series and numerical tables. It differs significantly from the release3-clause of the BSD license. It is a well-liked open-source of opinion that is utilized in machine learning and data analysis.

Pandas are a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python. Pandas are a Python library used for working with data sets.

- It has functions for analysis, cleaning, exploring, and manipulating data.

- The name "Pandas" has a reference to both "Panel Data", and "Python Data Analysis" and was created by Wes McKinney in 2008.
- Pandas allow us to analysis big data and make conclusions based on statistical theories.
- Pandas can clean messy data sets, and make them readable and relevant.

Relevant data is very important in data science. Pandas are a Python library for data analysis. Started by Wes McKinney in 2008 out of a need for a powerful and flexible quantitative analysis tool, pandas have grown into one of the most popular Python libraries. It has an extremely active community of contributors. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals. Its name is a play on the phrase "Python data analysis" itself.

NumPy:

The NumPy Python library for multi-dimensional, big-scale matrices adds a huge number of high-level mathematical functions. It is possible to modify NumPy by utilizing a Python library. Along with line, algebra, and the Fourier transform operations, it also contains several matrices-related functions.

NumPy can be used to perform a wide variety of mathematical operations on arrays. It adds powerful data structures to Python that guarantee efficient calculations with arrays and matrices and it supplies an enormous library of high-level mathematical functions that operate on these arrays and matrices.

- NumPy is a Python library used for working with arrays.
- It also has functions for working in domain of linear algebra, Fourier transform, and matrices.
- NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely.
- NumPy stands for Numerical Python.
- In Python we have lists that serve the purpose of arrays, but they are slow to process.
- NumPy aims to provide an array object that is up to 50x faster than traditional Python lists.

- The array object in NumPy is called ND array, it provides a lot of supporting functions that make working with ND array very easy.
- Arrays are very frequently used in data science, where speed and resources are very important.

○ Matplotlib:

It is a multi-platform, array-based data visualization framework built to interact with the whole SciPy stack. MATLAB is proposed as an open-source alternative. Matplotlib is a Python extension and a cross-platform toolkit for graphical plotting and visualization.

Matplotlib is a popular Python library for creating static, animated, and interactive visualizations. It provides a flexible and comprehensive set of tools for generating plots, charts, histograms, scatter plots, and more. Matplotlib is widely used in various fields, including data analysis, scientific research, and data visualization.

Here are some key features and functionalities of the Matplotlib library:

- Plotting Functions
- Customization Options
- Multiple Interfaces
- Integration with NumPy and pandas Subplots and Figures:
- Saving and Exporting

○ Scikit-learn:

The most stable and practical machine learning library for Python is scikit-learn. Regression, dimensionality reduction, classification, and clustering are just a few of the helpful tools it provides through the Python interface for statistical modelling and machine learning. It is an essential part of the Python machine learning toolbox used by JP Morgan. It is frequently used in various machine learning applications, including classification and predictive analysis.

Scikit-learn (also referred to as SK learn) is a widely used open-source machine learning library for Python. It provides a comprehensive set of tools and algorithms for various machine learning tasks, including classification, regression, clustering, dimensionality reduction, model selection, and pre-processing.

Here are some key features and functionalities of the Scikit-learn library:

- Easy-to-Use Interface:
- Broad Range of Algorithms:
- Data Pre-processing and Feature Engineering:
- Model Evaluation and Validation:
- Integration with NumPy and pandas:
- Robust Documentation and Community Support:

○ Kera:

* Google's Kera is a cutting-edge deep learning API for creating neural networks. It is created in Python and is designed to simplify the development of neural networks. Additionally, it enables the use of various neural networks for computation. Deep learning models are developed and tested using the free and opensource Python software known as Kera.

Kera is a high-level deep learning library for Python. It is designed to provide a user-friendly and intuitive interface for building and training deep learning models. Kera acts as a front-end API, allowing developers to define and configure neural networks while leveraging the computational backend engines, such as Tensor Flow or Theano.

Here are some key features and functionalities of the Kera library:

- User-Friendly API
- Multi-backend Support
- Wide Range of Neural Network Architectures Pre-trained Models and Transfer Learning:
- Easy Model Training and Evaluation:
- GPU Support:

○ h5py:

* The h5py Python module offers an interface for the binary HDF5 data format. Thanks to p5py, the top can quickly halt the vast amount of numerical data and alter it using the NumPy library. It employs common syntax for Python, NumPy, and dictionary arrays.

h5py is a Python library that provides a simple and efficient interface for working with datasets and files in the Hierarchical Data Format 5 (HDF5) format. HDF5 is a versatile data format commonly used for storing and managing large volumes of numerical data.

Here are some key features and functionalities of the h5py library:

- HDF5 File Access Dataset Handling:
- Group Organization:
- Attributes:
- Compatibility with NumPy
- Performance

○ Tensor flow

TensorFlow is a Python library for fast numerical computing created and released by Google. It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of TensorFlow. TensorFlow is an end-to-end open source platform for machine learning. TensorFlow is a rich system for managing all aspects of a machine learning system; however, this class focuses on using a particular TensorFlow API to develop and train machine learning models.

TensorFlow is a popular open-source library for machine learning and deep learning. It provides a comprehensive set of tools, APIs, and computational resources for building and training various types of machine learning models, especially neural networks.

Here are some key features and functionalities of TensorFlow:

- Neural Network Framework:
- Computational Graphs
- Automatic Differentiation
- GPU and TPU Support Distributed Computing
- Deployment Capabilities

Tk inter

Tk inter is an acronym for "Tk interface". Tk was developed as a GUI extension for the TCL scripting language by John Ousterhout. The first release was in 1991. Tk inter is the de facto way in Python to create Graphical User interfaces (GUIs) and is included in all standard Python Distributions. In fact, it's the only framework built into the Python standard library.

Tk inter is a standard Python library used for creating graphical user interfaces (GUIs). It provides a set of modules and classes that allow you to develop interactive and visually appealing desktop applications.

Here are some key features and functionalities of Tk inter:

- Cross-Platform Compatibility
- Simple and Easy-to-Use
- Widgets and Layout Management
- Event-Driven Programming
- Customization and Styling
- Integration with Other Libraries

NLTK

NLTK is a toolkit build for working with NLP in Python. It provides us various text processing libraries with a lot of test datasets. A variety of tasks can be performed using NLTK such as tokenizing, parse tree visualization, etc NLTK (Natural Language Toolkit) is the go-to API for NLP (Natural Language Processing) with Python. It is a really powerful tool to pre-process text data for further analysis like with ML models for instance. It helps convert text into numbers, which the model can then easily work with.

NLTK (Natural Language Toolkit) is a Python library widely used for working with human language data and implementing natural language processing (NLP) tasks. It provides a set of tools, corpora, and resources for tasks such as tokenization, stemming, tagging, parsing, sentiment analysis, and more.

Here are some key features and functionalities of NLTK:

- Text Processing Part-of-Speech Tagging
- Named Entity Recognition
- Chunking and Parsing Sentiment Analysis:
- WordNet Integration:

SciPy

SciPy is a collection of mathematical algorithms and convenience functions built on the NumPy extension of Python. It adds significant power to the interactive Python session by providing the user with high-level commands and classes for manipulating and visualizing data.

SciPy is a powerful scientific computing library for Python that provides a wide range of mathematical algorithms and functions. It builds upon NumPy, another fundamental library for numerical computing, and extends its capabilities by adding additional tools for scientific and technical computing tasks.

Here are some key features and functionalities of SciPy:

- Numerical Integration:
- Optimization and Root Finding
- Linear Algebra
- Signal and Image Processing
- Statistics

10. TESTING

Implementation and Testing:

Implementation is one of the most important tasks in project is the phase in which one has to be cautions because all the efforts undertaken during the project will be very interactive. Implementation is the most crucial stage in achieving successful system and giving the users confidence that the new system is workable and effective. Each program is tested individually at the time of development using the sample data and has verified that these programs link together in the way specified in the program specification. The computer system and its environment are tested to the satisfaction of the user.

Implementation

The implementation phase is less creative than system design. It is primarily concerned with user training, and file conversion. The system may be requiring extensive user training. The initial parameters of the system should be modifies as a result of a programming. A simple operating procedure is provided so that the user can understand the different functions clearly and quickly. The different reports can be obtained either on the inkjet or dot matrix printer, which is available at the disposal of the user. The proposed system is very easy to implement. In general implementation is used to mean the process of converting a new or revised system design into an operational one.

Testing

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property functions as a unit. The test data should be chosen such that it passed through all possible condition. Actually testing is the state of implementation which aimed at ensuring that the system works accurately and efficiently before the actual operation commence. The following is the description of the testing strategies, which were carried out during the testing period.

System Testing

Testing has become an System integral part of any system or project especially in the field of information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to be check if one is capable to with stand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to user to user the software must be tested whether it is solving the purpose for which it is developed. This testing involves various types through which one can ensure the software is reliable. The program was tested logically and pattern of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked.

Module Testing

To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level.

Integration Testing

After the module testing, the integration testing is applied. When linking the modules there may be chance for errors to occur, these errors are corrected by using this testing. In this system all modules are connected and tested. The testing results are very correct. Thus the mapping of jobs with resources is done correctly by the system.

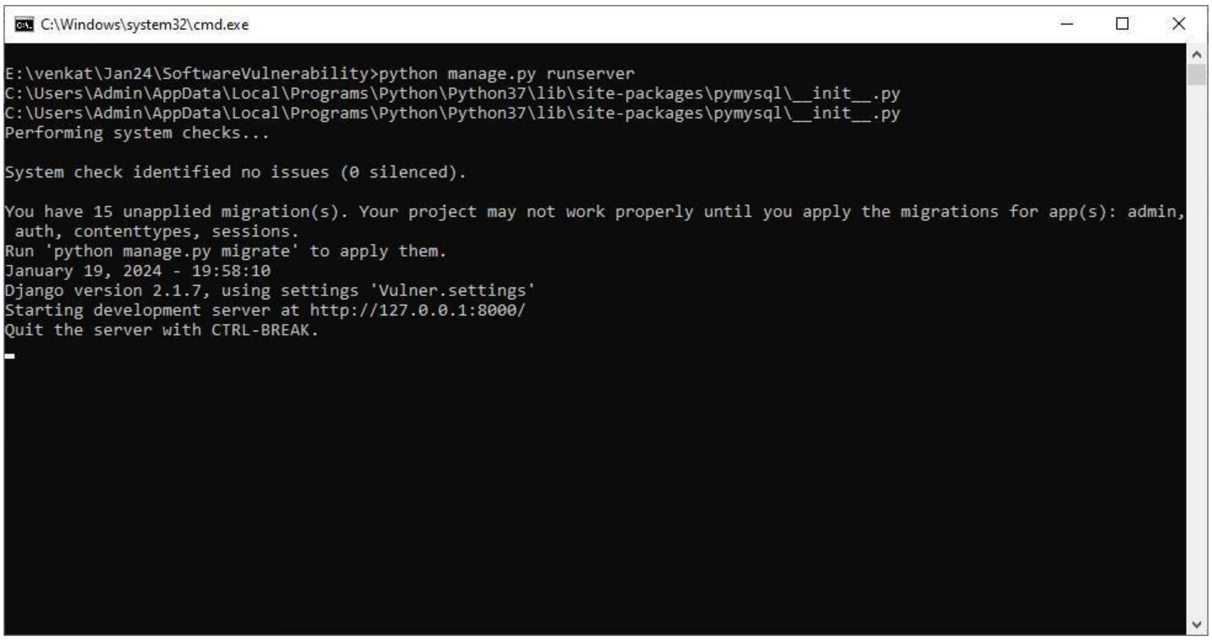
Acceptance Testing

When that user fined no major problems with its accuracy the system passers through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which elimination wastage of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

11. SCREEN SHOTS

To run project install python 3.7 and then install MYSQL database and then copy content from DB.txt file and paste in MYSQL to create database. Now double click on 'installNLTK.bat' file to download NLTK and once click then window will appear in that window click on "Download' button to download all packages and once downloaded then window will turn to green colour and then close the window

Now double click on 'run.bat' file to start python DJANGO web server and get below screen



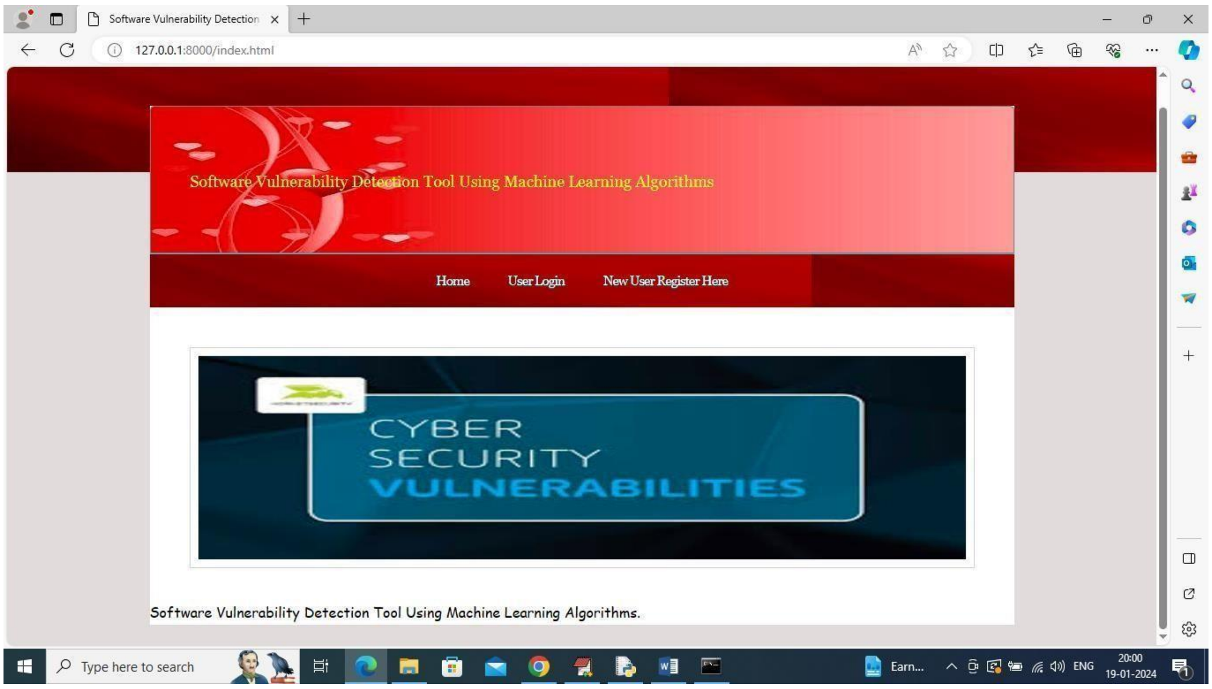
```
E:\venkat\Jan24\SoftwareVulnerability>python manage.py runserver
C:\Users\Admin\AppData\Local\Programs\Python\Python37\lib\site-packages\pymysql\__init__.py
C:\Users\Admin\AppData\Local\Programs\Python\Python37\lib\site-packages\pymysql\__init__.py
Performing system checks...

System check identified no issues (0 silenced).

You have 15 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin,
 auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
January 19, 2024 - 19:58:10
Django version 2.1.7, using settings 'Vulner.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

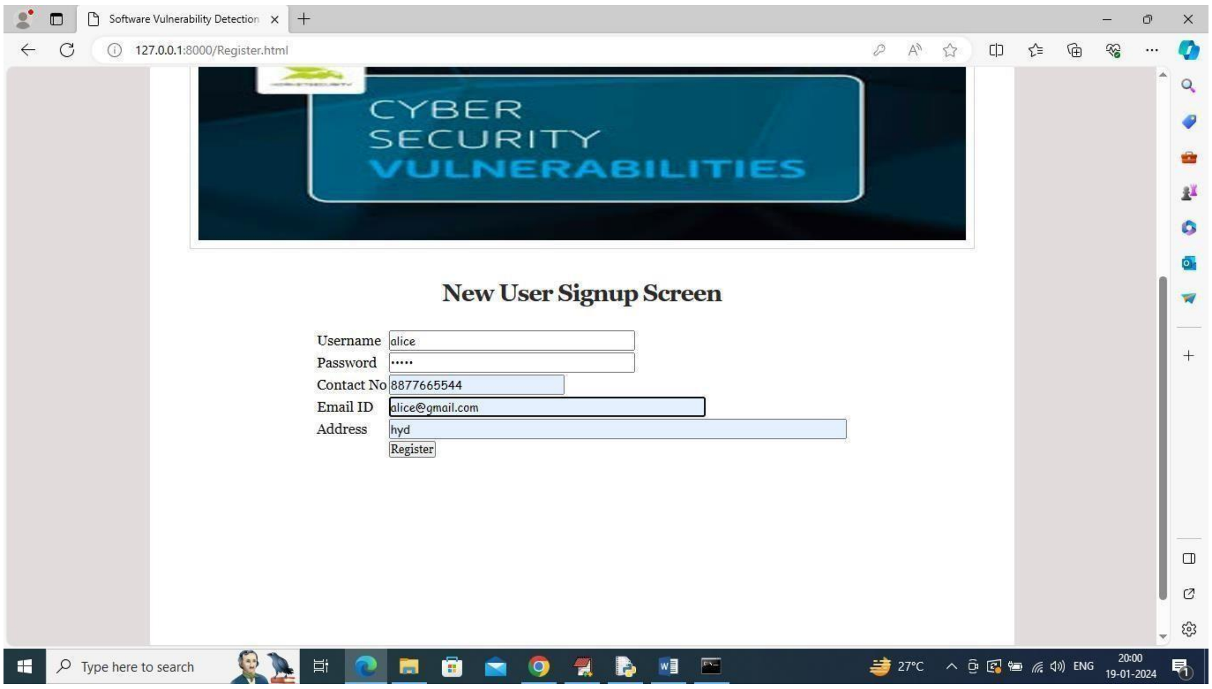
In above screen python web server started and now open browser and enter URL as

http://127.0.0.1:8000/index.html and then press enter key to get below page


127.0.0.1:8000/index.html
Software Vulnerability Detection Tool Using Machine Learning Algorithms
Home
User Login
New User Register Here
CYBER SECURITY VULNERABILITIES

Software Vulnerability Detection Tool Using Machine Learning Algorithms.

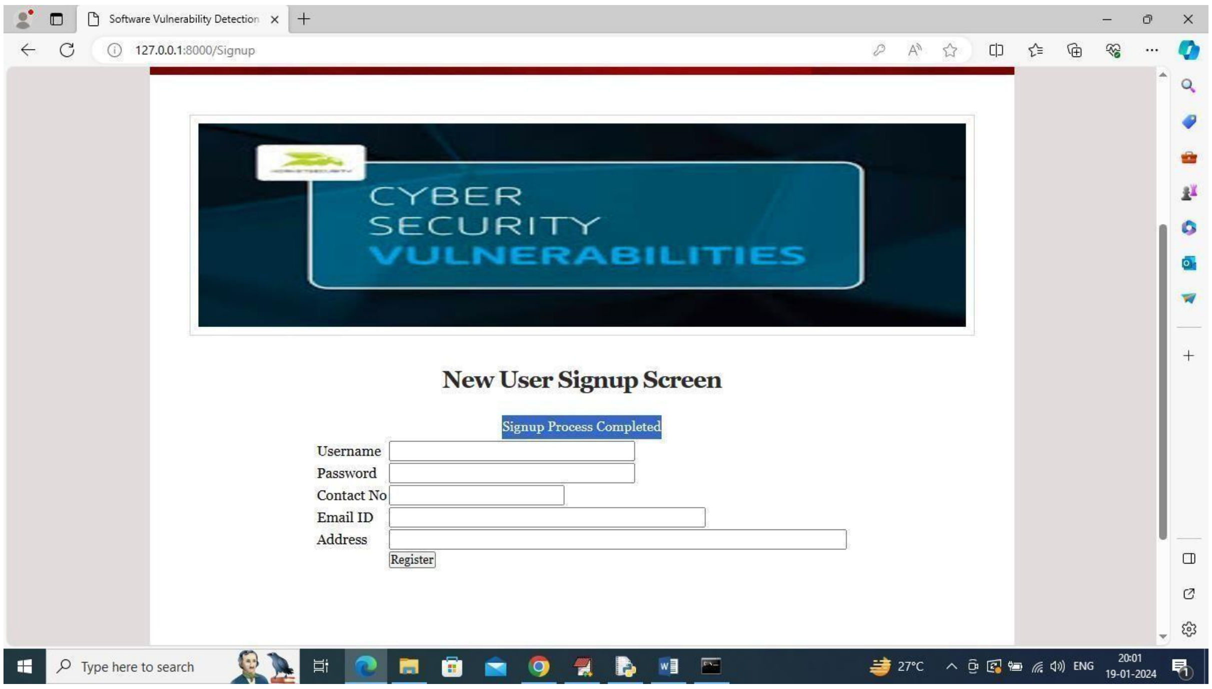
In above screen click on ‘New User Register Here’ link to get below sign up page



New User Signup Screen

Username	alice
Password	•••••
Contact No	8877665544
Email ID	alice@gmail.com
Address	hyd
	Register

In above screen user is entering sign up details and then press button to get below page


Software Vulnerability Detection
127.0.0.1:8000/Signup
CYBER SECURITY VULNERABILITIES

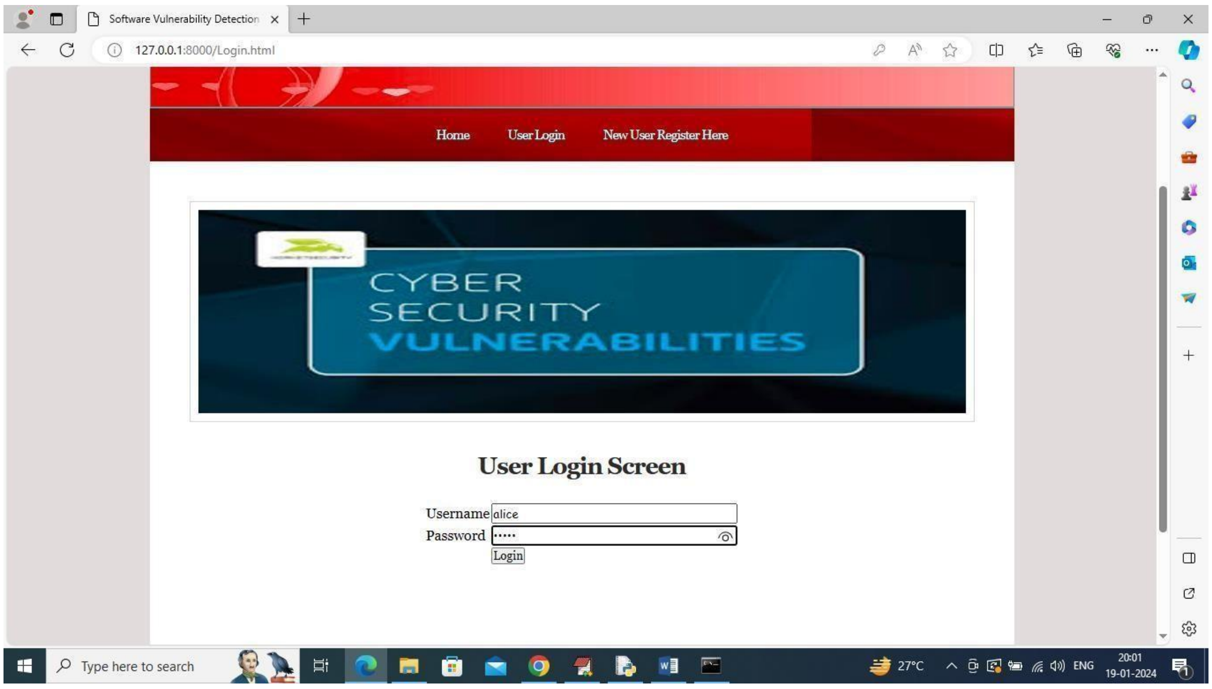
New User Signup Screen

Signup Process Completed

Username	
Password	
Contact No	
Email ID	
Address	

Register

In above screen user sign up completed and now click on ‘User Login’ link to get below page

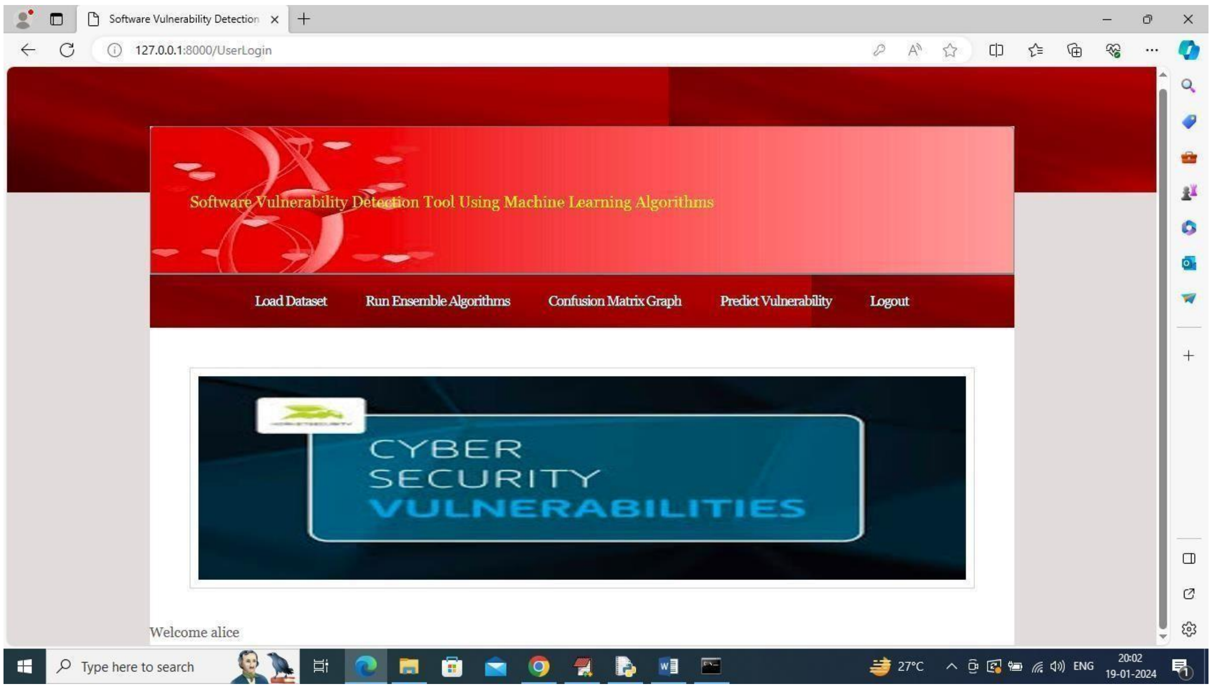

Software Vulnerability Detection
127.0.0.1:8000/Login.html
Home
User Login
New User Register Here
CYBER SECURITY VULNERABILITIES

User Login Screen

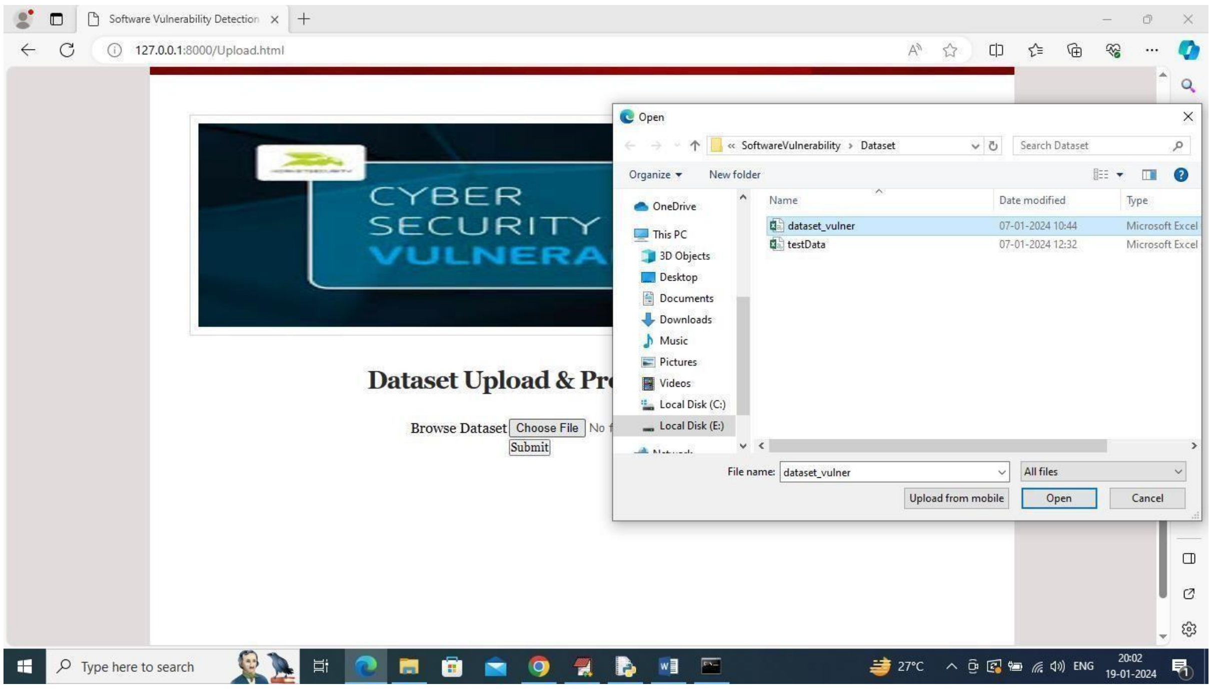
Username	alice
Password	•••••

Login

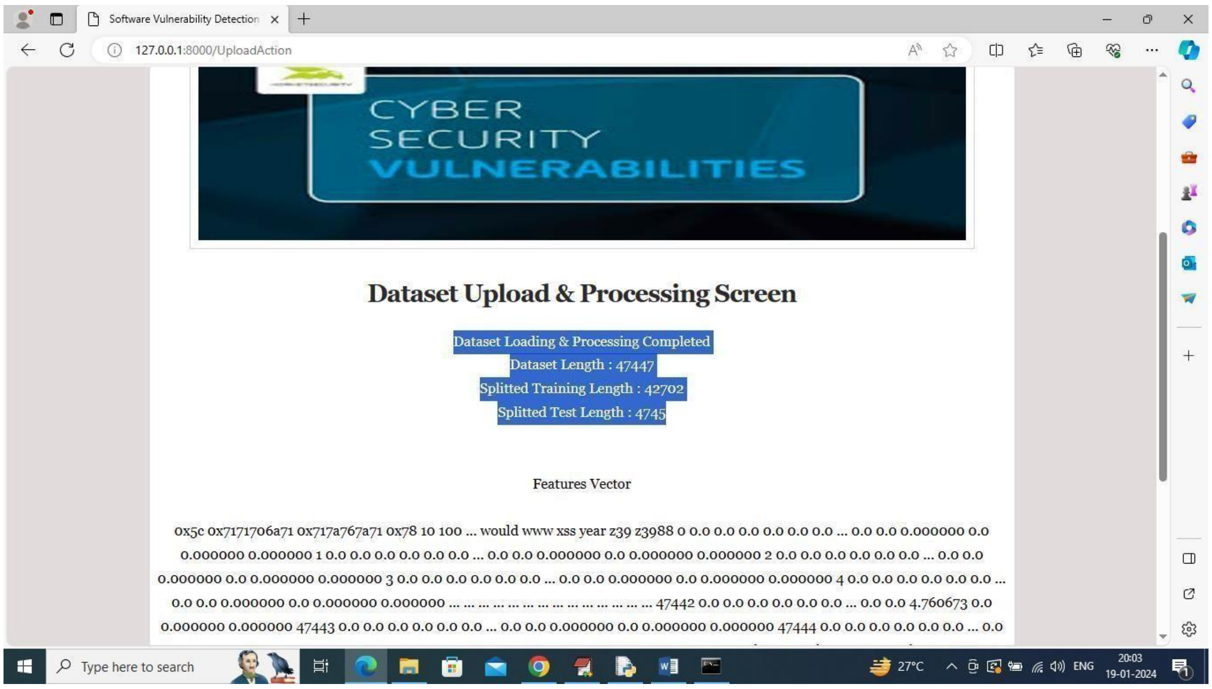
In above screen user is login and after login will get below page



In above screen click on ‘Load Dataset’ link to get below page



In above screen select and upload ‘dataset_vulner.csv’ file and then click on ‘Open’ and ‘Submit’ button to load dataset and then will get below output


Software Vulnerability Detection
127.0.0.1:8000/UploadAction
CYBER SECURITY VULNERABILITIES
Type here to search
27°C
ENG
20:03
19-01-2024

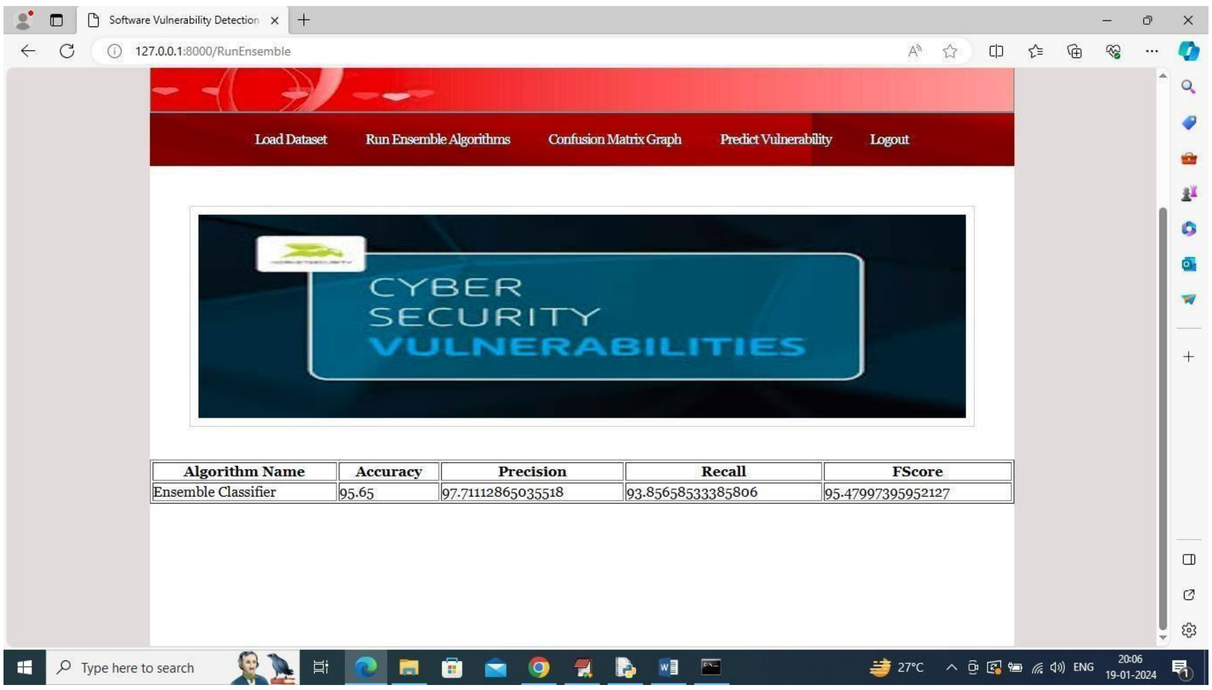
Dataset Upload & Processing Screen

Dataset Loading & Processing Completed
Dataset Length : 47447
Splitted Training Length : 42702
Splitted Test Length : 4745

Features Vector

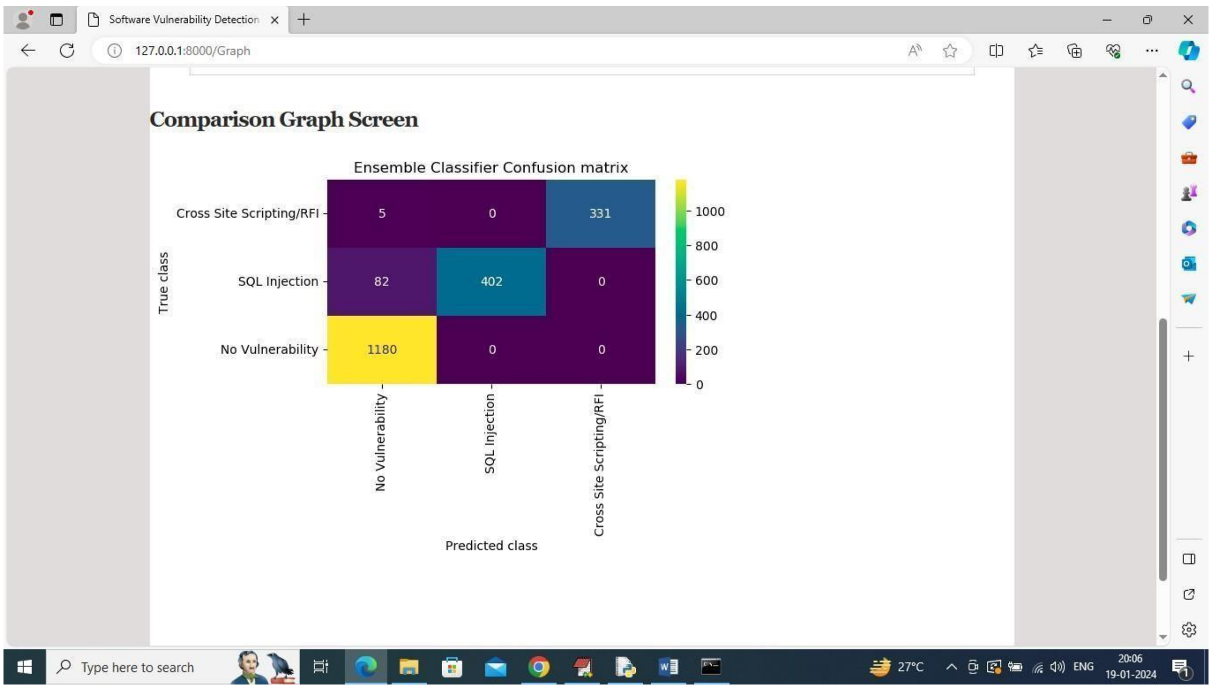
```
0x5c 0x7171706a71 0x717a767a71 0x78 10 100 ... would www xss year z39 z3988 0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.000000 0.0
0.000000 0.000000 1 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.000000 0.0 0.000000 0.000000 2 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0
0.000000 0.0 0.000000 0.000000 3 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.000000 0.0 0.000000 0.000000 4 0.0 0.0 0.0 0.0 0.0 0.0 ...
0.0 0.0 0.000000 0.0 0.000000 0.000000 ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... 47442 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 4.760673 0.0
0.000000 0.000000 47443 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.000000 0.0 0.000000 0.000000 47444 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0
```

In above screen can see dataset loaded and can see total number of records available in dataset and then can see training number of records on which Machine Learning algorithm get trained and then can see number of test records on which ML will perform prediction to calculate its prediction accuracy %. Now click on ‘Run Ensemble Algorithms’ link to train ensemble algorithm and then will get below output

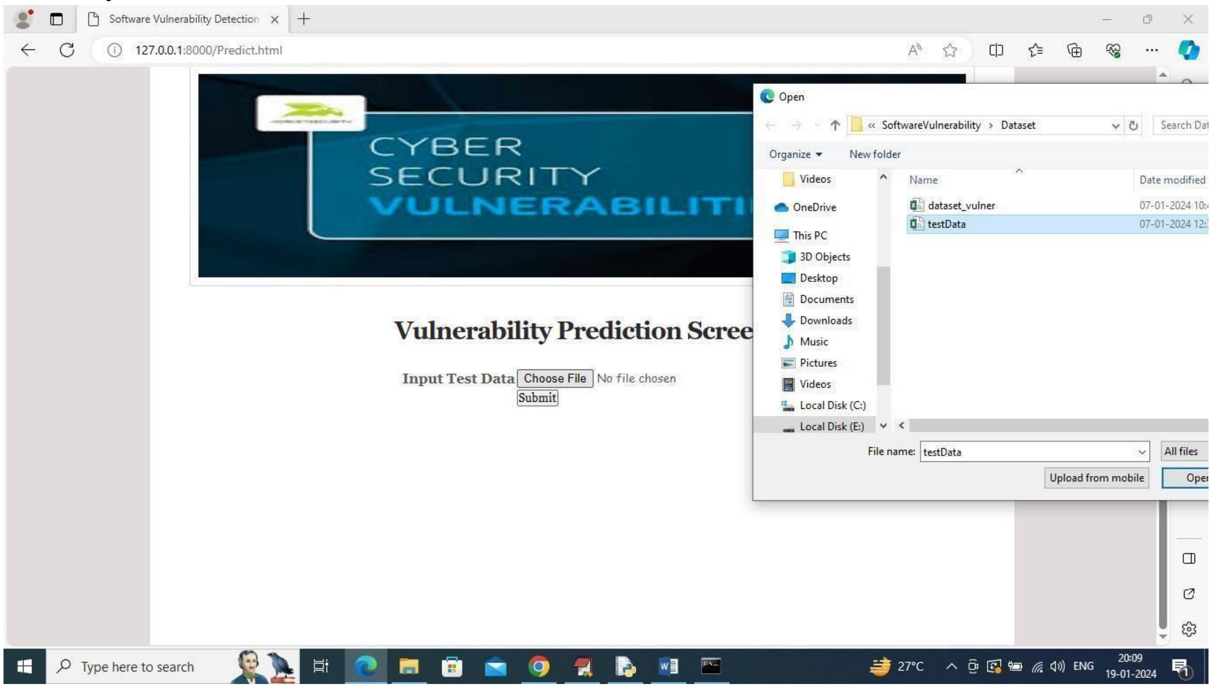


Algorithm Name	Accuracy	Precision	Recall	FScore
Ensemble Classifier	95.65	97.71112865035518	93.85658533385806	95.47997395952127

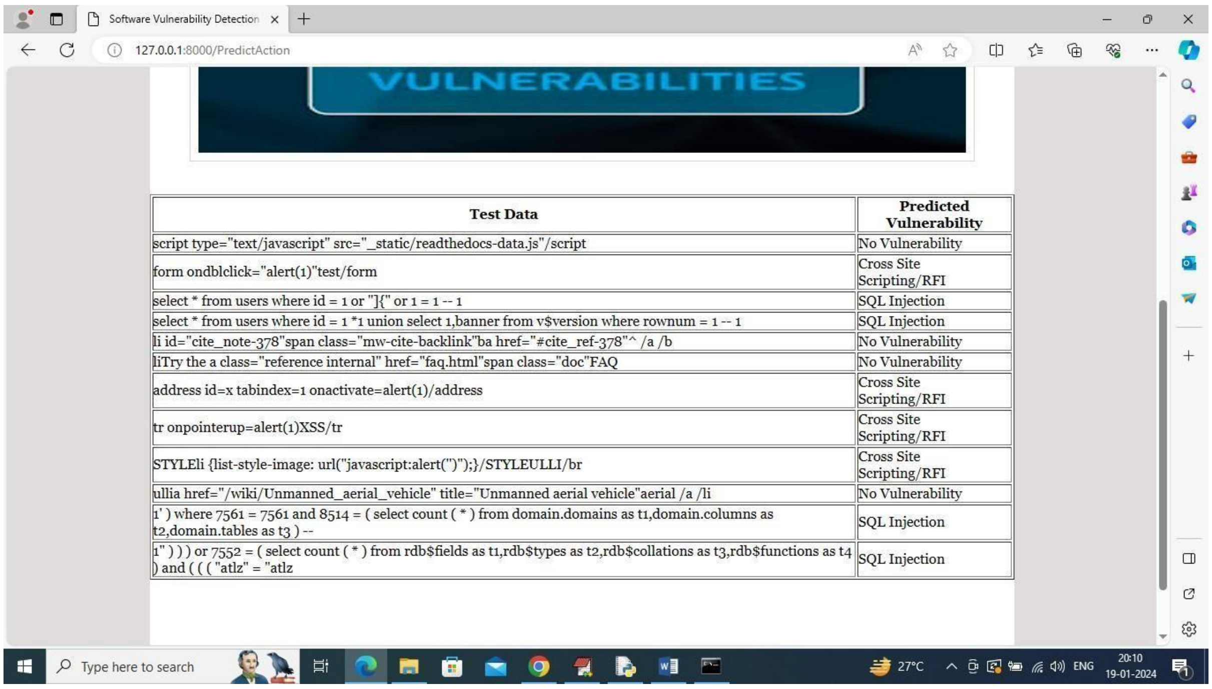
In above screen Ensemble Machine Learning algorithm training completed and can see its prediction accuracy as 95% and can see other metrics like precision, recall and FCSORE. Now click on 'Confusion Matrix Graph' link to view visually how many records ensemble predicted correctly and incorrectly



In above graph x-axis represents Predicted Labels and y-axis represents True Labels and then all different colour boxes in diagonal represents correct prediction count and remaining all blue boxes represents incorrect prediction count which are very few. Now click on ‘Predict Vulnerability’ link to upload test data and predict Vulnerability



In above screen selecting and uploading ‘testData.csv’ file which contains SQL, XSS and RFI coding commands and then click on ‘Submit’ button to get below output



Test Data	Predicted Vulnerability
script type="text/javascript" src="_static/readthedocs-data.js"/script	No Vulnerability
form ondblclick="alert(1)"test/form	Cross Site Scripting/RFI
select * from users where id = 1 or "]{" or 1 = 1 -- 1	SQL Injection
select * from users where id = 1 *1 union select 1,banner from v$version where rownum = 1 -- 1	SQL Injection
li id="cite_note-378"span class="mw-cite-backlink"ba href="#cite_ref-378"^ /a /b	No Vulnerability
liTry the a class="reference internal" href="faq.html"span class="doc"FAQ	No Vulnerability
address id=x tabindex=1 onactivate=alert(1)/address	Cross Site Scripting/RFI
tr onpointerup=alert(1)XSS/tr	Cross Site Scripting/RFI
STYLEli {list-style-image: url("javascript:alert(")");}/STYLEULLI/br	Cross Site Scripting/RFI
ullia href="/wiki/Unmanned_aerial_vehicle" title="Unmanned aerial vehicle"aerial /a /li	No Vulnerability
1') where 7561 = 7561 and 8514 = (select count (*) from domain.domains as t1,domain.columns as t2,domain.tables as t3) --	SQL Injection
1"))) or 7552 = (select count (*) from rdb$fields as t1,rdb$types as t2,rdb$collations as t3,rdb$functions as t4) and ((("atlz" = "atlz	SQL Injection

In above table in first column can see SQL queries, XSS and RFI coding commands and in second column can see predicted vulnerability.

12. CONCLUSION

After extensively testing function vulnerability classification using trivial features, n-grams, and suffix trees, we can draw several conclusions. First of all, we see that extracting numerous n-grams does not, thus far, seem to give good classification results, especially if we consider the 74% accuracy that we got from "character diversity "to be a baseline requirement. We also noticed that even when combinations of n-grams were manually selected (in a manner that would normally be illegal and lead to overfitting), the overall result did not improve. However, this study is a good proof-of-concept of a very important point: trivial features can tell us a lot about whether a function is vulnerable or not. There are a few directions in which this research can be taken to improve the results further. First of all, it might be possible to think of additional trivial features to investigate. Secondly, it might also make sense to test some other n-gram selection techniques, as well as some of the Scikit library's other classification parameters (other than default settings). Third, it would be interesting to look more closely at which characters (or strings) are most important, since this would give us better insight than just "character diversity". One way to do this would be to run the same character diversity tests after eliminating different strings (square brackets, curly brackets, ++, etc) in pre-processing. Finally, it is possible to test whether the techniques presented in this paper can be used to efficiently detect vulnerabilities in other programming languages (in addition to C).

13. REFERENCES

Enron email dataset. https://www.cs.cmu.edu/~enron/. Accessed: 2017-07-01.

National vulnerability database. https://nvd.nist.gov. Accessed: 2017-07-01.

V. R. Basili, L. C. Briand, and W. L. Melo. A validation of object-oriented design metrics as quality indicators. IEEE Transactions on software engineering, 22(10):751–761, 1996.

D. Brumley, T.-c. R. Johnson, H. Lin, and D. Song. Rich: Automatically protecting against integer-based vulnerabilities. Department of Electrical and Computing Engineering, page 28, 2007.

N. Dor, M. Rode, and M. Sagiv. CSSV: Towards a realistic tool for statically detecting all buffer overflows in

c. In ACM Notices, volume 38, pages 155–167. ACM, 2003.

D. Evans and D. Larochelle. Improving security using extensible lightweight static analysis. IEEE software, 19(1):42–51, 2002.

P. Godefroid, M. Y. Levin, and D. Molnar. Sage: white box fuzzing for security testing. Queue, 10(1):20, 2012.

I. Haller, M Neu and H. Bos. Dowsing for overflows: A guided buzzer to find buffer boundary violations. In USENIX Security Symposium, pages 49–64, 2013.

A. E. Hassan. Predicting faults using the complexity of code changes. In Proceedings of the 31st International Conference on Software Engineering, pages 78–88. IEEE Computer Society, 2009.

S. Kim, T. Zimmermann, E. J. Whitehead Jr, and A. Zeller. Predicting faults from cached history. In Proceedings of the 29th international conference on Software Engineering, pages 489–498. IEEE Computer Society, 2007.

D. Larochelle, D. Evans, et al. Statically detecting likely buffer overflow vulnerabilities. In USENIX Security Symposium, volume 32. Washington DC, 2001.

Y. Shin and L. Williams. An empirical model to predict security vulnerabilities using code complexity metrics. In Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement, pages 315–317. ACM, 2008.

J. Viega, J.-T. Bloch, Y. Kohno, and G. McGraw. Its4: A static vulnerability scanner for c and C++ code. In Computer Security Applications, 2000. ACSAC'00. 16th Annual Conference, pages 257–267. IEEE, 2000.

D. Wagner, J. S. Foster, E. A. Brewer, and A. Aiken. A first step towards automated detection of buffer overrun vulnerabilities. In NDSS, pages 2000–02, 2000.

R. Ma, Y. Yan, L. Wang, C. Hu, and J. Xue. Static buffer overflow detection for C/C++ source code based on abstract syntax tree. Journal of Residuals Science & Technology, 13(6), 2016.

E. Penttilä et al. Improving C++ software quality with static code analysis. N/A, 2014