## A MAJOR PROJECT REPORT on SOFTWARE QUALITY PREDICTION USING MACHINE LEARNING

***Submitted in partial fulfilment of the requirements of the degree***

### Bachelor of Technology In

**COMPUTER SCIENCE AND ENGINEERING**

### By

**BATCH 39 :**

**SINGAMANENI SREENADH ( O170997 )**

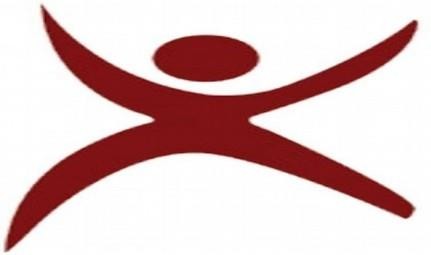
**KARNAM KARTHIK ( O171009 )**

**UTKURU YAMINI ( O171026 )**

**VELUPULA MEGHA JYOTHI ( O171029 )**

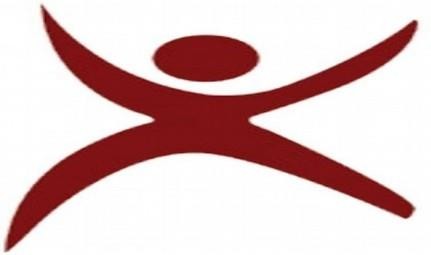
**MUNGARA JAHNAVI (O171071)**

### Under the supervision of Ms. P. SINDHU



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES, ONGOLE CAMPUS,**

**APRIL 2023.**



**BONAFIDE CERTIFICATE**

This is to certify that the project report entitled **SOFTWARE QUALITY PREDICTION USING MACHINE LEARNING** submitted by **MAJOR PROJECT BATCH NUMBER 39** in partial fulfilment of the requirement for the award of Bachelor of Technology in Computer Science Engineering is a record of bonafide project work carried out under my supervision during the academic year 2022-23.

I am indebted to **Ms.P.SINDHU**, my project guide for conscientious guidance and encouragement to accomplish this project.

I am extremely thankful and pay my gratitude to **Mr. B. SAMPATH BABU**,(I/C) HOD CSE, for his valuable guidance and support on the completion of this project.

The report hasn’t been submitted previously in part or in full to this or any other university or institution for the award of any degree.

Ms. P. SINDHU , Mr. B. SAMPATH BABU,

Assistant Professor, Head of Department(I/C),

Department of CSE, Department of CSE,

RGUKT, ONGOLE RGUKT, ONGOLE.

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES,**

**ONGOLE CAMPUS, APRIL 2023.**

## 

## CERTIFICATE

This is to certify that the project report entitled **SOFTWARE QUALITY PREDICTION USING MACHINE LEARNING** submitted by **MAJOR PROJECT BATCH NUMBER 39** in partial fulfilment of the requirement for the award of Bachelor of Technology in Computer Science and Engineering is a record of bonafide project work carried out under my supervision during the academic year 2022-23.

The report hasn’t been submitted previously in part or in full to this or any other university or institution for the award of any degree.

Ms. P . SINDHU, Mr. B. SAMPATH BABU ,

Assistant Professor, Head of Department(I/C),

Department of CSE, Department of CSE,

RGUKT, ONGOLE RGUKT, ONGOLE

**APPROVAL SHEET**

This report entitled SOFTWARE QUALITY PREDICTION USING MACHINE LEARNING by **MAJOR PROJECT BATCH 39** is **Ms.P. SINDHU** approved for the degree of Bachelor of Technology in COMPUTER SCIENCE AND ENGINEERING

Examiner

Supervisor

Chairman

Date:

Place:

**DECLARATION**

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

BATCH 39 :

SINGAMANENI SREENADH ( O170997 )

KARNAM KARTHIK ( O171009 )

UTKURU YAMINI ( O171026 )

VELUPULA MEGHA JYOTHI ( O171029 )

MUNGARA JAHNAVI (O171071)

Date:

## ACKNOWLEDGEMENT

I am highly indebted to **Ms. P. SINDHU** for her guidance and constant supervision as well as for providing necessary information regarding the project and also for their kind cooperation, encouragement and their support in completing the project.

I would like to express my special gratitude and thanks to our COMPUTER SCIENCE AND ENGINEERING branch **H.O.D Mr. B. SAMPATH BABU** and **Director of RGUKT, ONGOLE Prof. B. JAYARAMI REDDY** for giving me such attention and time.

I have taken efforts in this project. However, it could not have been possible without the kind support and help of many individuals and RGUKT.We would like to extend my sincere thanks to all of them.

My thanks and appreciation also go to my colleagues in developing the project and people who have willingly helped me out with their abilities.

Date:

With Sincere Regards,

MAJOR PROJECT BATCH 39

**ABSTRACT**

Software quality estimation is an activity needed at various stages of software development. It may be used for planning the project`s quality assurance practices and for benchmarking. In earlier previous, two methods (Multiple Criteria Linear Programming and Multiple Criteria Quadratic Programming) for estimating the quality of software had been used. Also, C5.0, SVM and Neutral network were experimented with for quality estimation. I have aimed to improve estimation accuracy by using relevant features of a large dataset. I used a feature selection method and correlation matrix for reaching higher accuracies. In addition, I have experimented with recent methods shown to be successful for other prediction tasks. Machine learning such as KNN, Random Forest and Decision Tree are applied to the data to predict the software quality and reveal the relation between the quality and development attributes. The experimental results show that the quality level of software can be well estimated by machine learning algorithms.

## TABLE OF CONTENTS

|  |  |
| --- | --- |
| **Description** | **Page no** |
| BONAFIDE CERTIFICATE | i |
| CERTIFICATE | ii |
| APPROVAL SHEET | iii |
| DECLARATION | iv |
| ACKNOWLEDGEMENT | v |
| ABSTRACT | vi |
| INTRODUCTION  1.1 Organization Profile | 01  02 |
| LITERATURE SURVEY | 04 |
| PROBLEM DEFINITION  3.1 Existing System  3.2 Proposed System  3.2 Modules | 06  06  07 |
| DATA COLLECTION | 08 |
| SYSTEM ANALYSIS  5.1 Software Development Life cycle  5.2 User requirement Documentation  5.3 Use Case Diagrams  5.4 Software Requirement Specification | 09  12  12  13 |
| SYSTEM ENVIRONMENT  6.1 Hardware Requirements  6.2 Software Requirements | 15  15 |
| SYSTEM DESIGN  7.1High Level Design Documentation  7.1.1 Class Diagram  7.1.2 Collaboration Diagram  7.1.3 Sequence Diagram  7.1..4 Activity Diagram  7.1.5 Component Diagram  7.1.6 Deployment Diagram | 16  17  17  17  18  19  20 |
| IMPLEMENTATION  8.1 Screenshots | 22 |
| SYSTEM TESTING  9.1 Testing Strategies  9.2 Test Cases | 26  28 |
| REPORTS | 30 |
| CONCLUSION  11.1 Limitations  11.2 Future Work | 31  31 |
| [APPENDIX](#_TOC_250002)  User Manual | 32 |
| BIBLIOGRAPHY | 33 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Figure No** | **Figure Name** | **Page No.** |
| Figure 5.1.1 | SDLC Life Cycle | 10 |
| Figure 5.2.1 | Use Case Diagram | 13 |
| Figure 7.1.1 | Class Diagram | 17 |
| Figure 7.1.2 | Collaboration diagram | 17 |
| Figure 7.1.3 | Sequence Diagram | 18 |
| Figure 7.1.4 | Activity diagram | 19 |
| Figure 7.1.5 | Component diagram | 20 |
| Figure 7.1.6 | Deployment diagram | 20 |
| Figure 8.1.1 | Pycharm Connection Page | 22 |
| Figure 8.1.2 | Home Page | 22 |
| Figure 8.1.3 | Model Performance Page | 23 |
| Figure 8.1.4 | Accuracy Page | 23 |
| Figure 8.1.5 | Existing Logistic Report Page | 24 |
| Figure 8.1.6 | Input Features Page | 24 |
| Figure 8.1.7 | Final Output Page | 25 |
| Figure 10.1 | Reports | 30 |

# **CHAPTER 1**

# **INTRODUCTION**

Software applications may contain defects, originating from requirements analysis, specification and other activities conducted in the software development. Therefore, software quality estimation is an activity needed at various stages. It may be used for planning the project based quality assurance practices and for benchmarking. In addition, the number of defects per unit is considered one of the most important factors that indicate the quality of the software.

The quality of a software product can be deﬁned as the measure of performance of a system on which the software is implemented in terms of execution time, memory capacity utilized and probability of errors, etc. In addition to this, the amount of effort contributed by the software developer also represents a key factor while assessing the quality of a software product. The quality of a software product can be considered to be internal as well as external. The internal quality of a software can be assessed in course of development during software development life cycle (SDLC); whereas, the external quality can be measured during its implementation and can be assessed with respect to its level of functionality.

The external quality also depends upon its internal quality. In order to assess the external quality of a software product, quality models can be devised that represent a function of the internal quality attributes. Inorder to achieve this, ﬁrst of all the internal attributes must be identiﬁed and then the relationship existing between the internal and external quality attributes must be identiﬁed. A number of software quality prediction models have been proposed by various authors. However, the machine learning approach to devising such a model appears to be more popular and more effective as claimed by the authors. We have been motivated by this aspect to carry out a review of the machine learning approaches for software quality Q1`prediction models.

###### 1.1 ORGANIZATION PROFILE

In late 2017 Quad Tech was acquired by Baldwin Technology Company, Inc. Baldwin is creating a new printing technology powerhouse by combining PC Industries, Quad Tech, and Web Printing Controls into a new “BALDWIN | Vision Systems” global entity. BALDWIN’s Vision Systems division is the world’s leading innovator of advanced color and inspection technology for the printing industry. BALDWIN | Vision Systems sells its automated control systems in more than 100 countries to the packaging and converting markets, web offset newspaper and commercial markets, and publication gravure market. BALDWIN maintains a worldwide network of sales and service operations, and is privately owned by BW Forsyth Partners, the investment arm of multi-billion dollar global manufacturing and engineering consulting firm Barry Wehmiller.

###### Milestones

* 1979QuadTech founded by Thomas A. Quadracci, Jeff Sainio, and Jerry Kirby.
* 1980Produced first RGS II. Produced first Silicone Coater.
* 1981Produced first Chill Roll Stand.
* 1982Closed the loop on register control with the development of the RGS III.
* 1984Introduced the Cutoff Control System.
* 1985Celebrated 5 years in the industry.
* 1986Introduced the RGS IV and Cutoff Control.
* 1989Formally opened QuadTech Europe in Weesp, The Netherlands.

Celebrated 100th RGS IV installation in Japan.

* 1990Introduced the RGS V.
* 1993Officially opened the Asia-Pacific office in Singapore.
* 1994Opened Quad Tech, Inc. in Japan.
* 1996Acquired Stacker Machine Company.
* 1999Acquired Bomac Signature Loader.
* 2002Acquired Press Tech Controls Limited, inheriting 55 years of industry expertise.

###### Mission

Our mission is to bring leading edge technologies to treat challenging water and wastewater. QUAD will accomplish this mission through implementation of our core values, which include:

* + Exceptional product performance
  + Quality service to back up our products
  + Creating long term, value-added relationships with our customers and strategic partners
  + Continuous product improvement to meet the needs of our customers and the marketplace

###### Vision

Our vision is to become the global leader in advanced membrane products to treat water and wastewater. We will develop a suite of leading edge products based on customer needs, experience, and demonstrated performance. With a committed workforce, solid financials, and a presence in key global markets, our presence in key geographic markets enable us to be closer to our customers. This provides a better understanding to the specific requirements and regulations in these regions. Our organization is built around four pillars – R&D, manufacturing, marketing, and customer service

###### **CHAPTER 2**

###### LITERATURE SURVEY

A Literature survey in a project report represents the study done to assist in the completion of a project. A literature survey also describes a survey of the previous existing material on a topic of the report. A project report is an assessment during a process or project conveying these details.

###### Cowlessur, Sanjeev & Pattnaik, Saumendra & Pattanayak, Binod. (2020). A Review of Machine Learning Techniques for Software Quality Prediction. 10.1007/978-981-15- 1483-8\_45.

Successful implementation of a software product entirely depends on the quality of the software developed. However, prediction of the quality of a software product prior to its implementation in real-world applications presents signiﬁcant challenges to the software developer during the process of development. A limited spectrum of research in this area has been reported in the literature as of today. Most of the researchers have concentrated their research work on software quality prediction using various machine learning techniques.

###### A. A. CERAN and O. O. TANRIOVER, "An experimental study for software quality prediction with machine learning methods," 2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), Ankara, Turkey, 2020, pp. 1-4, 10.1109/HORA49412.2020.9152918.

software quality estimation is an activity needed at various stages of software development. It may be used for planning the project's quality assurance practices and for benchmarking. In earlier previous studies, two methods (Multiple Criteria Linear Programming and Multiple Criteria Quadratic Programming) for estimating the quality of software had been used Also, C5.0, SVM and Neutral network were experimented with for quality estimation.

###### Pattnaik, Saumendra & Pattanayak, Binod. (2016). A survey on machine learning techniques used for software quality prediction. International Journal of Reasoning- based Intelligent Systems. 8. 3. 10.1504/IJRIS.2016.080058.

In the present software development scenario, software quality prediction has become significantly important for successful implementation of the software in real worldapplication and enhances the longevity of its functionality. Moreover, early identification of anticipated fault prone software modules in the process of development of software is crucial in saving efforts involved in this process. Machine learning techniques are considered to be the most appropriate techniques for software quality prediction and a large spectrum of research work has been conducted in this direction by several authors. In this paper, we conduct an extensive survey on various machine learning techniques like fuzzy logic, neural network, and Bayesian model, etc. used for software quality prediction along with an analytical justification for each of the proposed solutions.

###### Huang, Bing & Li, Xiaojun & Li, Ming & Bernstein, Joseph & Smidts, Carol. (2005). Study of the impact of hardware fault on software reliability. 2005. 10 pp.-. 10.1109/ISSRE.2005.39.

As software plays increasingly important roles in modern society, reliable software becomes desirable for all stakeholders. One of the root causes of software failure is the failure of the computer hardware platform on which the software resides. Traditionally, fault injection has been utilized to study the impact of these hardware failures. One issue raised with respect to the use of fault injection is the lack of prior knowledge on the faults injected, and the fact that, as a consequence, the failures observed may not represent actual operational failures.

###### **CHAPTER 3**

###### PROBLEM DEFINITION

Due to lack of sufficient tools to evaluate and predict software quality one of the major challenges in software engineering field. We are giving software quality prediction using machine learning approaches such as Random Forest, Decision Tree, and KNN algorithm.

###### EXISTING SYSTEM

Software quality estimation is an activity needed at various stages of software development. It may be used for planning the project`s quality assurance practices and for benchmarking. In earlier previous, two methods (Multiple Criteria Linear Programming and Multiple Criteria Quadratic Programming) for estimating the quality of software had been used. Also, C5.0, SVM and Neutral network were experimented with for quality estimation. I have aimed to improve estimation accuracy by using relevant features of a large dataset. I used a feature selection method and correlation matrix for reaching higher accuracies. In addition, I have experimented with recent methods shown to be successful for other prediction tasks. Machine learning such as Knn, Random Forest and Decision Tree are applied to the data to predict the software quality and reveal the relation between the quality and development attributes. The experimental results show that the quality level of software can be well estimated by machine learning algorithms.

###### LIMITATIONS

1. Accuracy is low.
2. Difficult to handle.
3. Low reliability.

###### PROPOSED SYSTEM

* + - It is aimed to improve the estimation of accuracy by using relevant features of a large dataset.
    - We used a feature selection method are correlation matrix for reaching higher accuracies.
    - In the proposed system we are using Machine Learning algorithms such as Random Forest, decision tree and KNN algorithm are applied to the data to predict the software quality.

###### ADVANTAGES

* + - * High accuracy.
      * Time Saving.
      * Low complexities.
      * High reliability.

###### MODULES

* + 1. **Store Dataset:**

The System stores the dataset given by the user.

###### Model Training:

The system takes the data from the user and fed that data to the selected model.

###### Model Predictions:

The system takes the data given by the user and predict the output based on the given

data**.**

###### Load Dataset:

The user can load the dataset he/she want to work on.

###### View Dataset:

The User can view the dataset.

###### Select model:

User can apply the model to the dataset for accuracy**.**

**Algorithm Used:** KNN Algorithm

###### **CHAPTER 4**

###### DATA COLLECTION

The source of information for developing the proposed system is gathered directly from clients of end user who is going to use the package becomes the primary source to give information for this application to develop.

A detailed study and understanding of the existing system are done either by questionnaires or by conducting interviews before developing the proposed one. Different inputs, process and output are well understood before designing the system.

The data collection is process of the collecting the primary data about the problem so many techniques are used to collect the following techniques for the data collection.

###### Interview Method:

An interview is a process of conducting intensive individual interviews with asmall number of respondents to explore their perspectives on a particular idea, program or situation. Face to face method of discussion was used to gather information from people who work in the company/organization and get the knowledge to design a responsive system. Requirements to carry out the interviews with respondents will be made so that they provide the general information concerning how work is done at the company to generate .

###### QUESTIONNAIRE METHOD:

* Why python is good for this application?
* Importance of flask infrastructure?
* What is Recall Score?
* What is precision score?
* What are the uses of this system?
* Importance of flask infrastructure?

###### **CHAPTER 5**

###### SYSTEM ANALYSIS

System analysis is the process of gathering and interpreting facts, diagnosing the problem and using the information to recommended improvement to the system. Analysis is an activity that encompasses most of the tasks that are collectively called computer system engineering. System engineering and analysis encompass requirements gathering at the system level with a small amount of top-level analysis and design. Requirement’s analysis is the first technical step in the software engineering process. The requirement’s gathering process is intensified and focused specially on software. The analyst must understand the information domain for the software, as well as required function, behaviour, performance and interfacing. Requirements for both the system and the software are documented and received with the customer.

System Analysis Includes

* The surveying and planning of the system.
* The study and analysis of existing system.
* The definitions of requirements and priorities for new or improved system. For which popular synonym is Logical design.

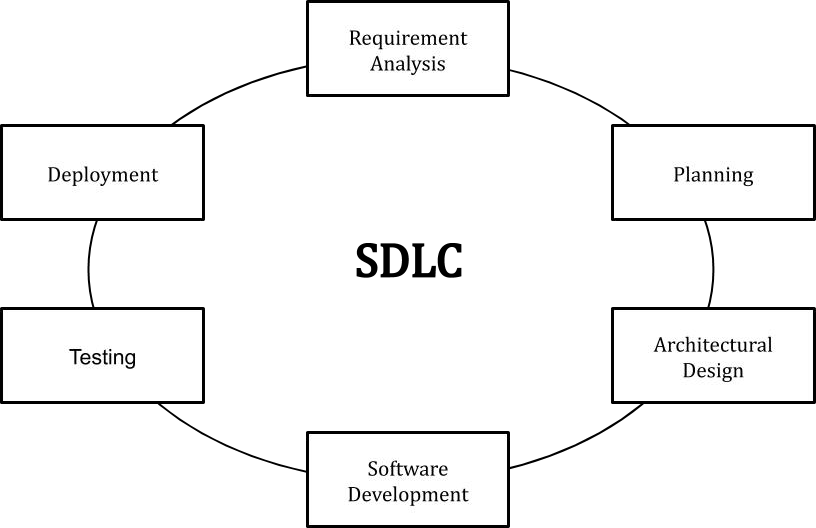
###### SOFTWARE DEVOLOPMENT LIFE CYCLE

There is a various software development approaches defined and designed which are used/employed during development process of software, these approaches are also referred as "Software Development Process Models". Each process model follows a particular life cycle in order to ensure success in process of software development that is called “Software Development Life Cycle”.

It defines the requirements of the new system. It then creates the software through the stages of analysis, planning, design, development, testing, and deployment showed in figure 5.1.1. By anticipating costly mistakes like failing to ask the end-user or client for feedback, SLDC can eliminate redundant rework and after-the-fact fixes.

It’s also important to know that there is a strong focus on the testing phase. As the SDLC is a repetitive methodology, you have to ensure code quality at every cycle. Many organizations

tend to spend few efforts on testing while a stronger focus on testing can save them a lot of rework, time, and money. Be smart and write the right types of tests.



*Figure-5.1.1: SDLC life cycle*

**REQUIREMENT ANALYSIS**

Business requirements are gathered in this phase. This phase is the main focus of the project managers and stake holders. Meetings with managers, stake holders and users are held in order to determine the requirements. Who is going to use the system? How will they use the system? What data should be input into the system? What data should be output by the system? These are general questions that get answered during requirement gathering phase. This produces a nice big list of functionalities that the system should provide, which describes functions the system should perform, business logic that processes data, what data is stored and used by the system, and how the user interface should work. The overall result is the system as a whole and how it performs, not how it is actually going to do it.

**PLANNING**

Planning for the quality assurance requirements and identifications of the risks associated with the projects is also done at this stage. Business analyst and Project organizer set up a meeting with the client to gather all the data like what the customer wants to build, who will be the end user, what is the objective of the product. Before creating a product, a core understanding or knowledge of the product is very necessary.

**DESIGN**

The software system design is produced from the results of the requirements phase. Architects have the ball in their court during this phase and this is the phase in which their focus lies. This is where the details on how the system will work is produced. Architecture, including hardware and software, communication, software design (UML is produced here) are all part of the deliverables of a design phase.

**DEVELOPMENT**

Code is produced from the deliverables of the design phase during implementation, and this is the longest phase of the software development life cycle. For a developer, this is the main focus of the life cycle because this is where the code is produced. Implementation my overlap with both the design and testing phases. Many tools exist (CASE tools) to actually automate the production of code using information gathered and produced during the design phase.

###### TESTING

During testing, the implementation is tested against the requirements to make sure that the product is actually solving the needs addressed and gathered during the requirements phase. Unit tests and system/acceptance tests are done during this phase. Unit tests act on a specific component of the system, while system tests act on the system as a whole.

**DEPLOYMENT**

Once the software is certified, and no bugs or errors are stated, then it is deployed. Then based on the assessment, the software may be released as it is or with suggested enhancement in the object segment. After the software is deployed, then its maintenance begins.

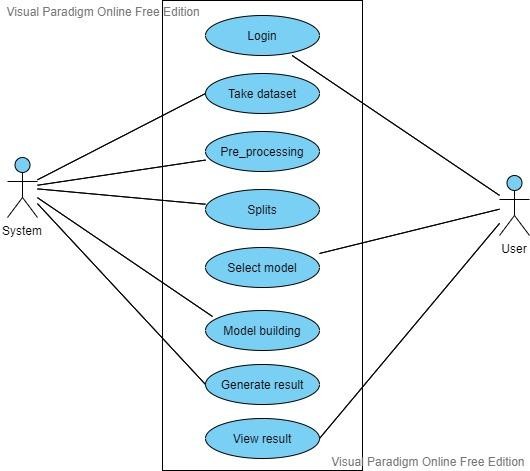
###### USER REQUIREMENT DOCUMENTATION

The user requirement(s) document (URD) or user requirement(s) specification (URS) is a document usually used in software engineering that specifies what the user expects the software to be able to do. Once the required information is completely gathered it is documented in a URD, which is meant to spell out exactly what the software must do and becomes part of the contractual agreement. A customer cannot demand features not in the URD, while the developer cannot claim the product is ready if it does not meet an item of the URD. The URD can be used as a guide for planning cost, timetables, milestones, testing, etc. The explicit nature of the URD allows customers to show it to various stakeholders to make sure all necessary features are described. Formulating a URD requires negotiation to determine what is technically and economically feasible. Preparing a URD is one of those skills that lies between a science and an art, requiring both software technical skills and interpersonal skills.

When a system is being created, User Requirements Specifications are a valuable tool for ensuring the system will do what users need it to do. In Retrospective Validation, where an existing system is being validated, user requirements are equivalent to the Functional Requirements: the two documents can be combined into a single document.

###### 5.3 USECASE DIAGRAM

A use case diagram in the Unified Modelling Language (UML) is a type of behavioural diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



*Figure-5.2.1 Use case Diagram*

###### 5.4 SOFTWARE REQUIREMENT SPECIFICATIONS

SRS is the official statement of what the system developers should implement. SRS is a complete description of the behaviour of the system to be developed. SRS should include both a definition of user requirements and a specification of the system requirements. The SRS fully describes what the software will do and how it will be expected to perform.

###### Functional and Non-functional Requirements

Requirement’s analysis is very critical process that enables the success of a system or software project to be assessed. Requirements are generally split into two types: Functional and non-functional requirements.

###### Functional Requirements

These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into system

as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

Examples of functional requirements:

* Authentication of user whenever he/she logs into the system
* System shutdown in case of a cyber-attack
* A verification email is sent to user whenever he/she register for the first time on some software system.

###### Non-functional Requirements

These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements. They basically deal with issues like:

* Portability
* Security
* Maintainability
* Reliability
* Scalability

###### PROJECT SCOPE

* + - * The aim is to predict the quality of the software.
      * For that we have to pre-process the data then split the dataset into training and testing and then fed the training data to the Machine Learning Models.
      * After building the model I will analyse the model performance and predict the quality of the software

###### **CHAPTER 6**

###### SYSTEM ENVIRONMENT

These are the requirements for doing the project. Without using these tools and software’s we can’t do the project. So, we have two requirements to do the project. They are

1. Hardware Requirements.
2. Software Requirements.

###### HARDWARE REQUIREMENTS

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system.

They are used by software engineers as the starting point for the system design.

RAM : 8GB

Processor : Intel I3

Hard Disk : 128 GB.

###### SOFTWARE REQUIREMENTS

The software requirements document is the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the team’s and tracking the team’s progress throughout the development activity.

Front End : Python v3.6

Back End : MY SQL Operating System : Windows 07.

###### **CHAPTER 7**

###### SYSTEM DESIGN

* 1. **HIGH LEVEL DESIGN DOCUMENT**

The purpose of this High-Level Design (HLD) Document is to add the necessary detail to the current project description to represent a suitable model for coding. This document is also intended to help detect contradictions prior to coding and can be used as a reference manual for how the modules interact at a high level.

The HLD documentation presents the structure of the system, such as the database architecture, application architecture (layers), application flow (Navigation), and technology architecture. The HLD uses non-technical to mildly technical terms which should be understandable to the administrators of the system

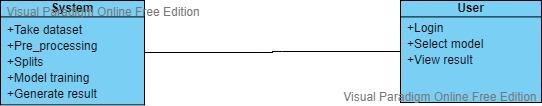
The HLD is a general system design. It includes the description of the following parts:

* + - System architecture
    - Database design
    - Brief mention of all the platforms, systems, services, and processes the product would depend on
    - Brief description of relationships between the modules and system features

All the data flows, flowcharts, data structures, etc. are in these docs, so that developers can understand how the system is expected to work with regards to the features and the database design.

###### 7.1.1 CLASS DIAGRAM

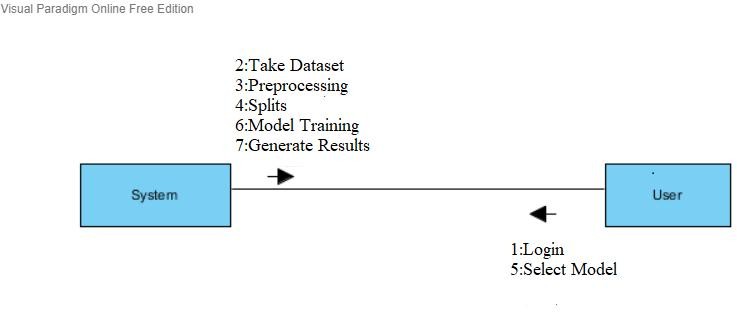
In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



* + 1. *Class Diagram*

###### 7.1.2 COLLABORATION DIAGRAM

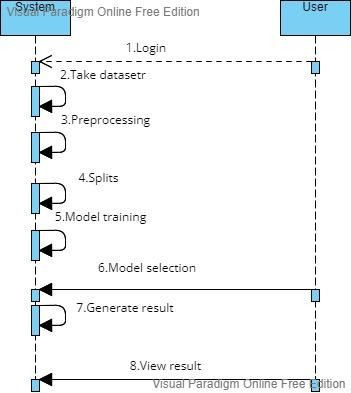
In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are like that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.



* + 1. *Collaboration Diagram*

###### 7.1.3 SEQUENCE DIAGRAM

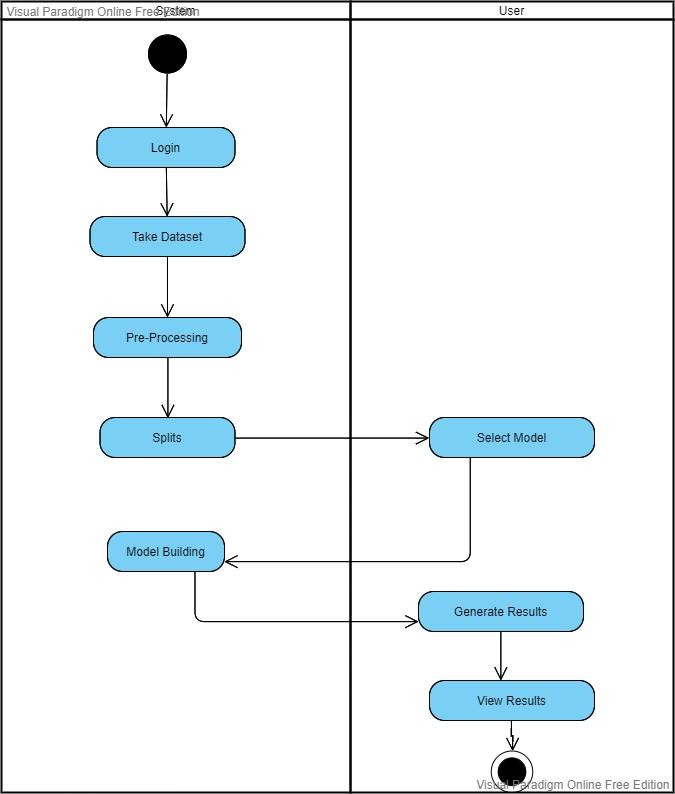
A sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



* + 1. *Sequence Diagram*

###### 7.1.4 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration, and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by- step workflows of components in a system. An activity diagram shows the overall flow of control.



* + 1. *Activity Diagram*

###### 

###### 7.1.5 COMPONENT DIAGRAM

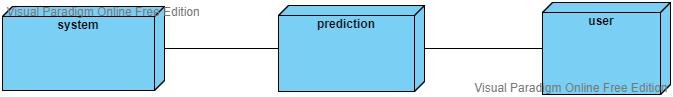
Component diagrams are used to describe the physical artifacts of a system. This artifact includes files, executables, libraries etc. So, the purpose of this diagram is different, Component diagrams are used during the implementation phase of an application. But it is prepared well in advance to visualize the implementation details. Initially the system is designed using different UML diagrams and then when the artifacts are ready component diagrams are used to get an idea of the implementation.



* + 1. *Component Diagram*

###### 7.1.6 DEPLOYMENT DIAGRAM

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardwires used to deploy the application.



* + 1. *Deployment Diagram*

###### **CHAPTER 8**

###### IMPLEMENTATION

The project has been implemented using code written in Java that is executed in Eclipse which isconnected to database called MySQL.

First of all, we need to install Java in the system.

###### Pycharm

Step1: Go to google and search pycharm download Step2:Click on https://[www.jetbrains.com/pycharm/download/do](http://www.jetbrains.com/pycharm/download/do) wnload-thanks.html?platform=windows

Step 3: Download the latest version of Pycharm and install it.

###### MY SQL

Step 1: Go to google and search for MySQL download. Step 2: Click on [https://www.mysql.com](https://www.mysql.com/).

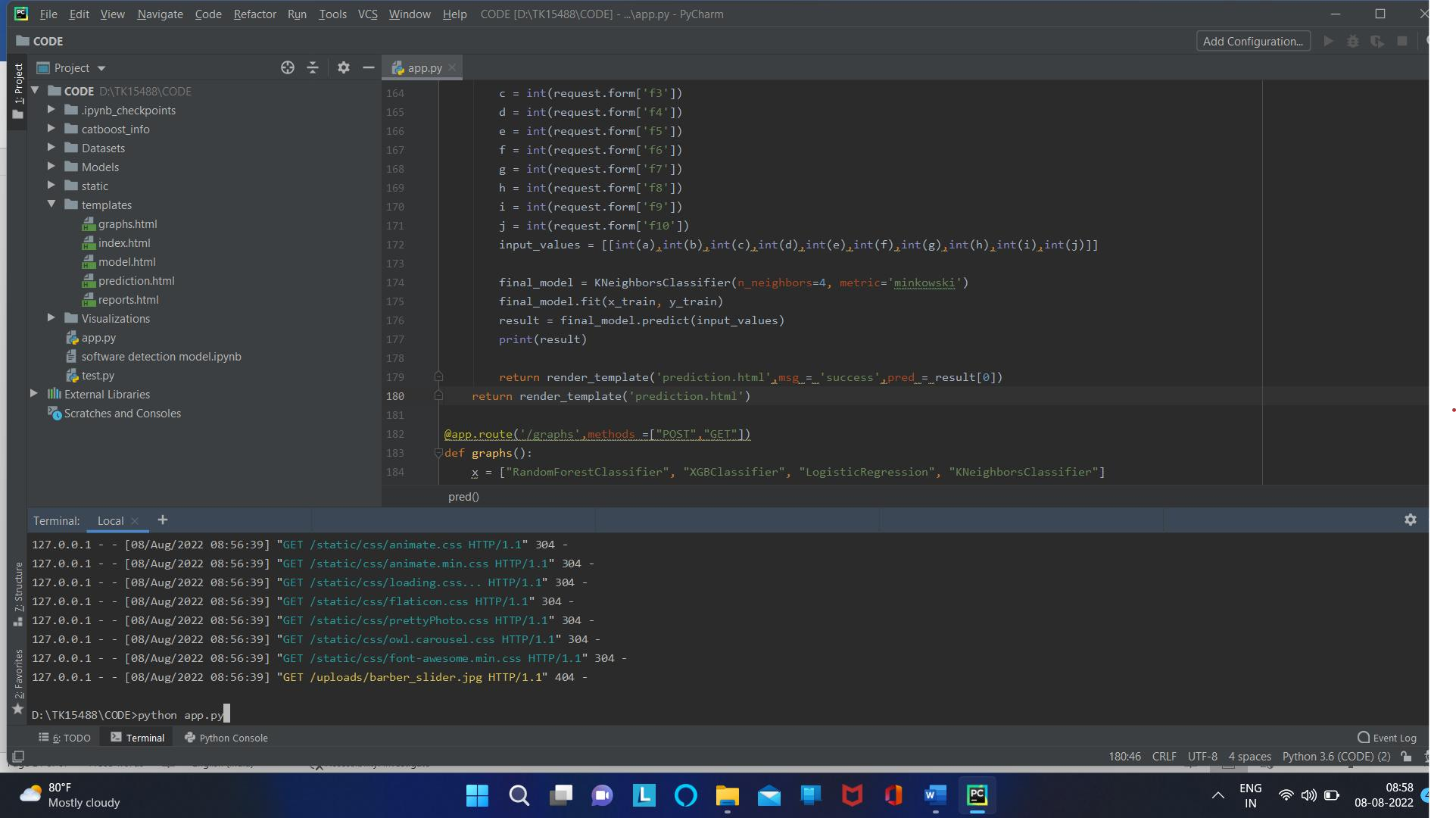
Step 3: Download latest version of MY SQL and install it

Step 4: Set user ID and password.

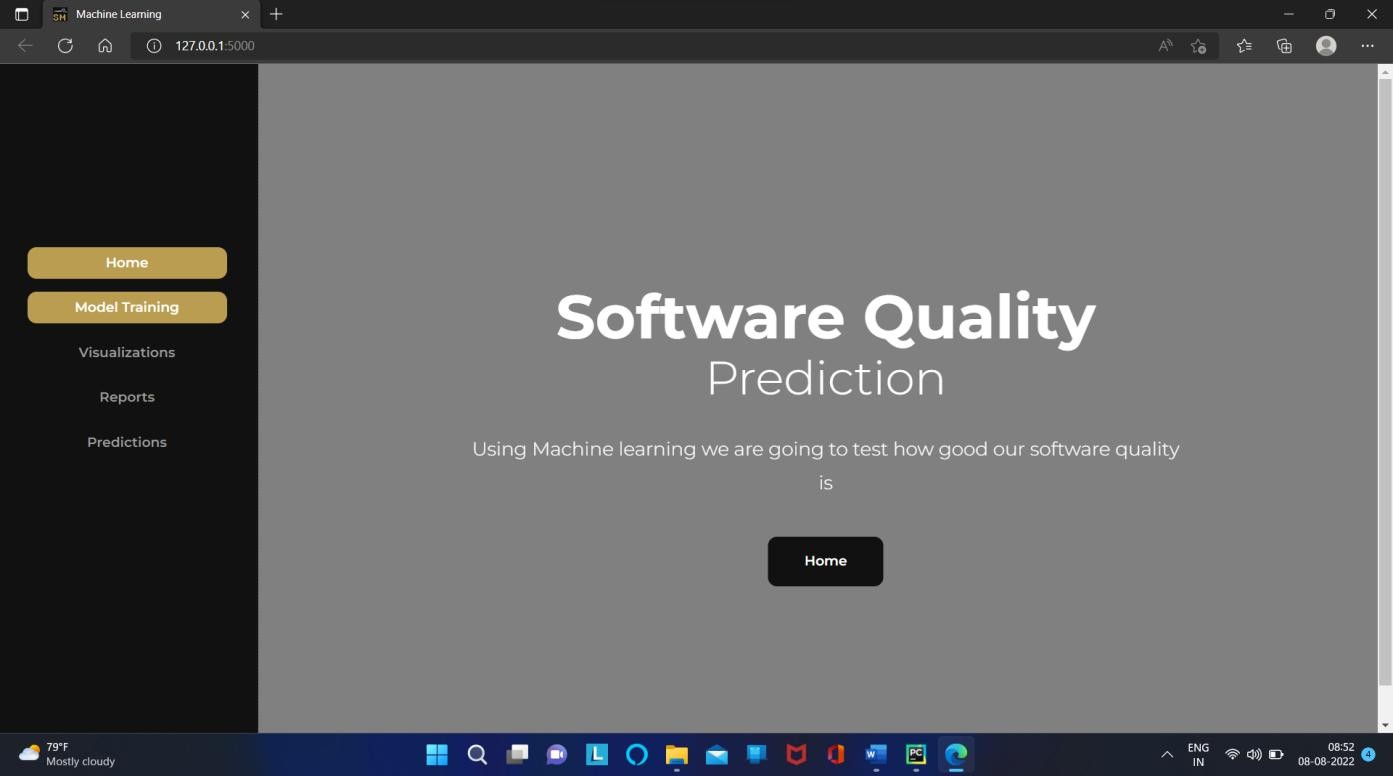
Step 5: Import download file in to MySQL Administrator.

###### SCREEEN SHOTS

**PYCHARM CONNECTION PAGE**

* + 1. *Pycharm Connection Page*

###### HOME PAGE



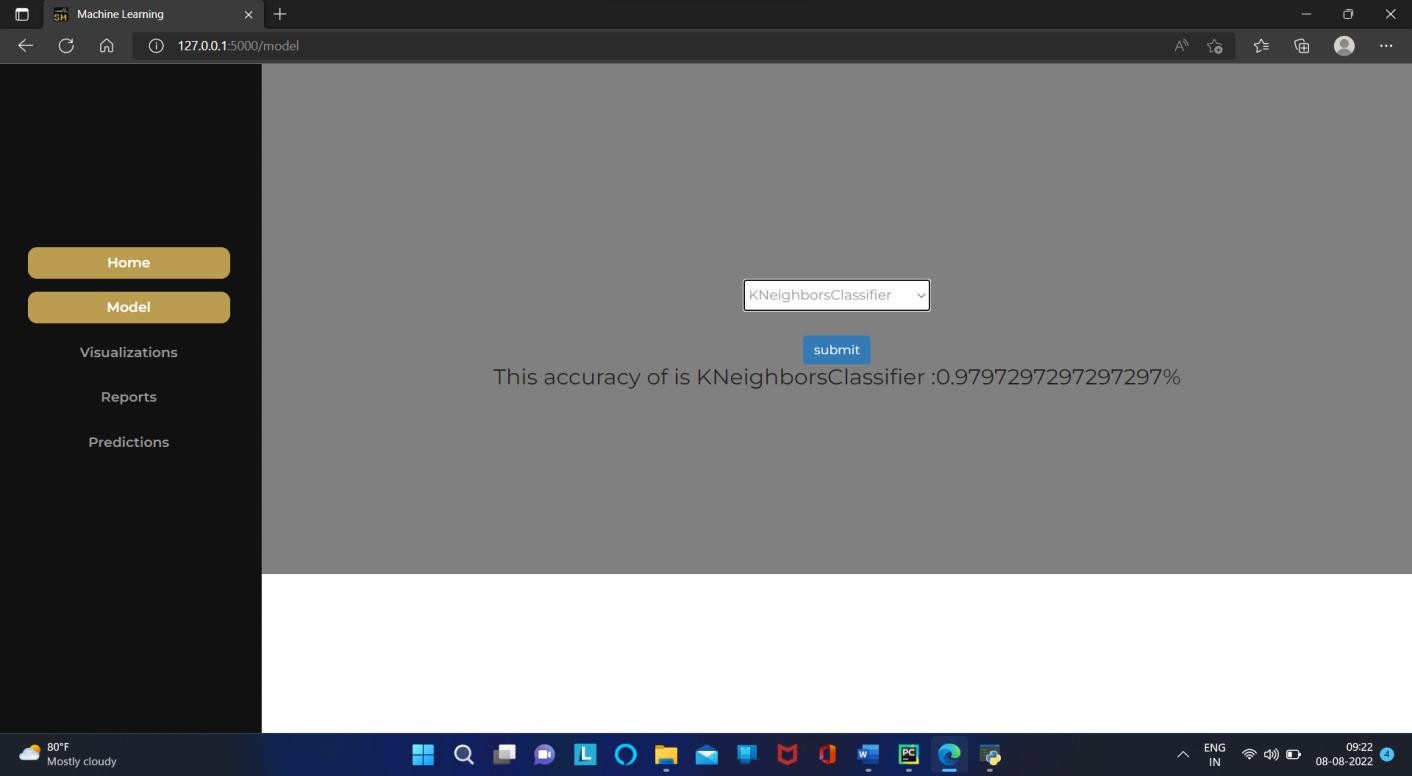
* + 1. *Home Page*

###### MODEL PERFORMANCE PAGE



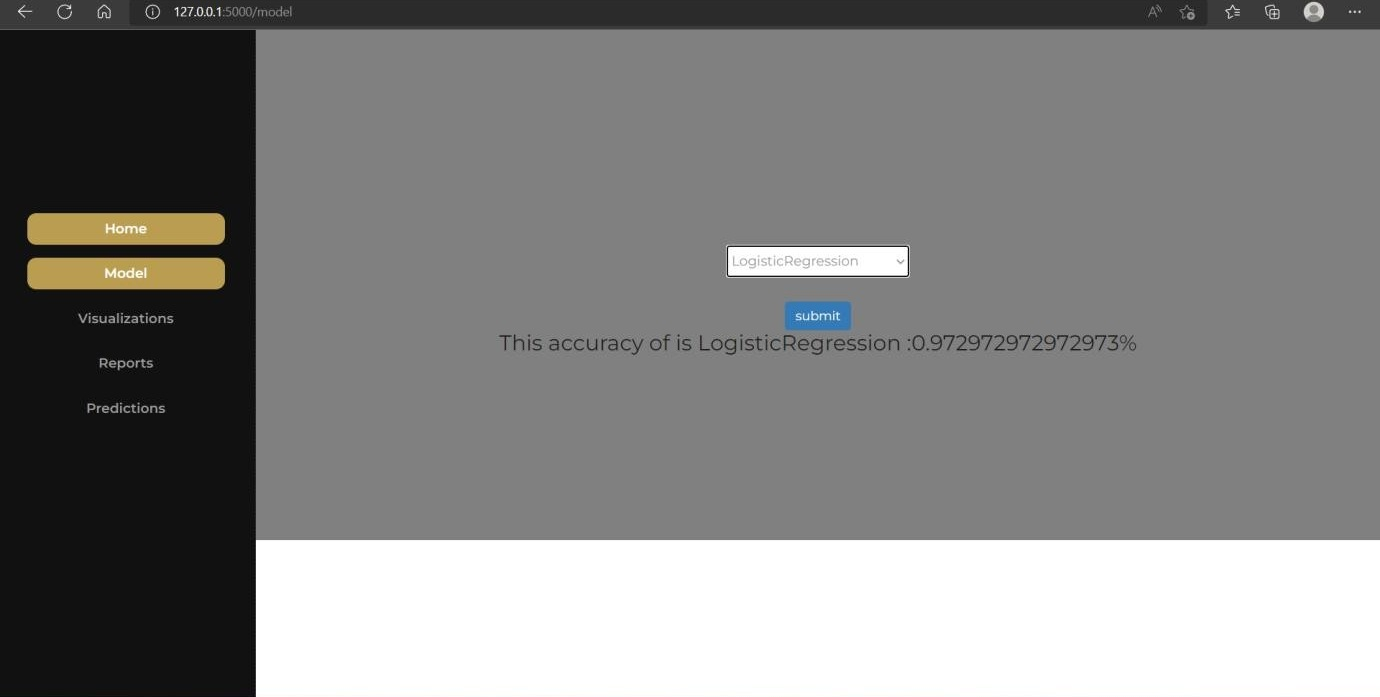
* + 1. *Model Performance*

###### ACCURACY PAGE



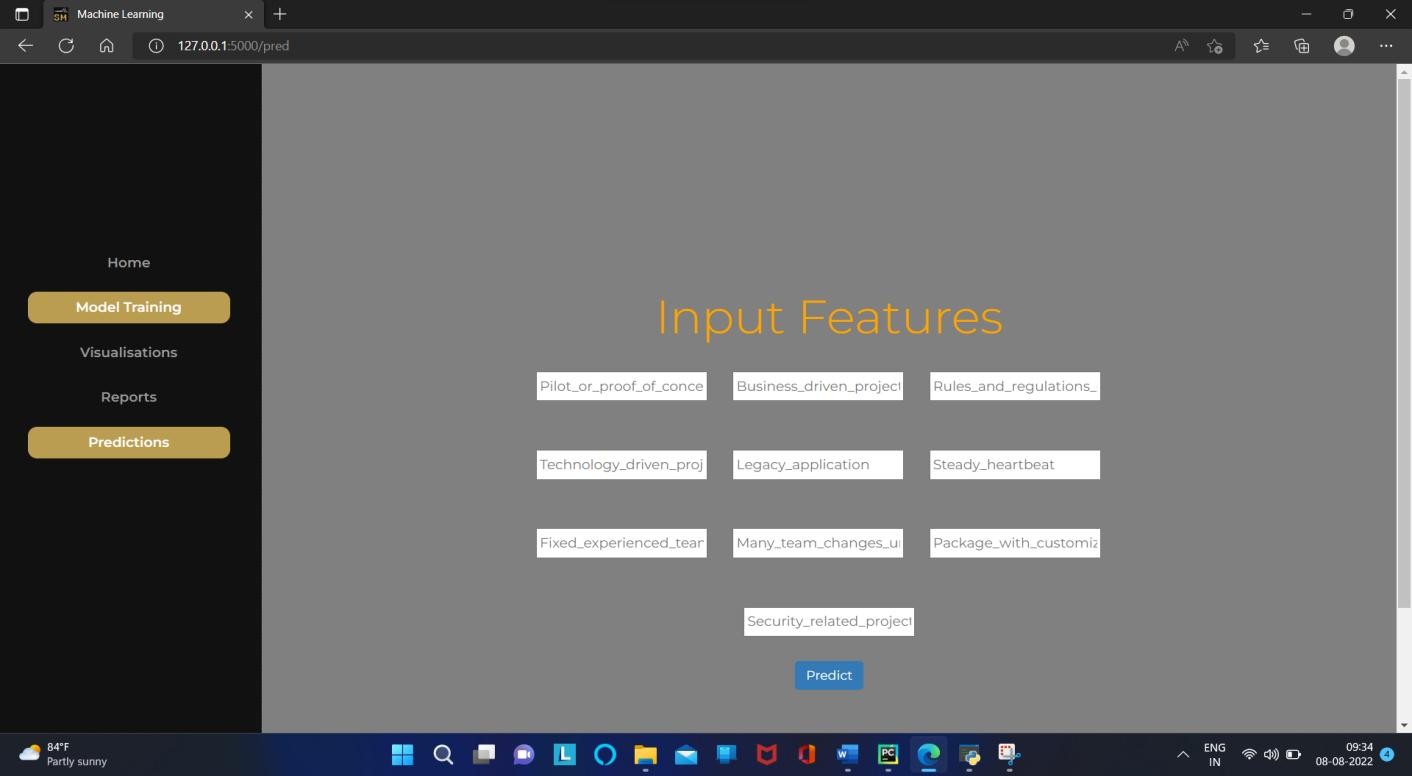
* + 1. *Accuracy Page*

###### EXISTING LOGISTIC REPORT PAGE



*8.1.5 Logistic Report*

###### INPUT FEATURES PAGE

 *8.1.6 Input Features Page*

###### FINAL OUTPUT PAGE

* + 1. *Final Output*

###### **CHAPTER 9**

###### SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement. System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. After the development of the system are already complete. This system will be test using two techniques of software testing which are black box testing and white box testing in order to examine the functionality of the system. Black box testing module involve login, manage user, manage budget, manage income, and manage report. In white box testing it involve generate the report.

###### TESTING STRATEGIES Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

###### Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components

were individually satisfaction, as shown by successfully unit testing, the combination of

components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

###### Functional Test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests are determined.

###### White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

###### Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

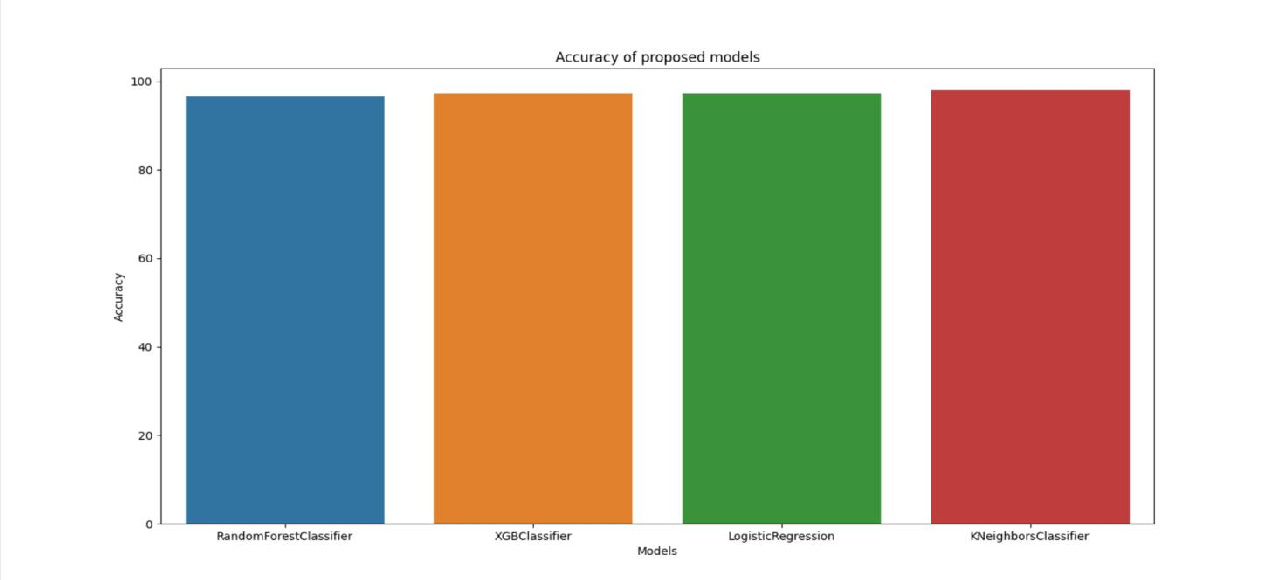
###### TEST CASES

|  |  |
| --- | --- |
| **Tested By:** | SINGAMANENI SREENADH ( O170997 )  KARNAM KARTHIK ( O171009 ) |
| **Test Type** | Black Box Testing |
| **Test Case Number** | TC001 |
| **Test Name** | Model Training |
| **Test Description** | Choosing of algorithm, Which one gives the best accurate results. |
| **Item(s) to be tested** | |
| 1 | KNN Algorithm |
| **Specifications** | |
| Input | Expected Result |
| KNN Algorithm | It Shows accurate results |
| **Procedural Steps** | |
| 1 | Open Home Page |
| 2 | Select Model Training |
| 3 | Enter the Algorithm Name |
| 4 | It Shows Accurate Results |

|  |  |
| --- | --- |
| **Tested By:** | UTKURU YAMINI ( O171026 )  VELUPULA MEGHA JYOTHI ( O171029 )  MUNGARA JAHNAVI (O171071) |
| **Test Type** | Black Box Testing |
| **Test Case Number** | TC002 |
| **Test Name** | Software Quality Checking |
| **Test Description** | User can give the input Data |
| **Item(s) to be tested** | |
| 1 | Input Values |
| **Specifications** | |
| Input | Expected Result |
| After entering the input Data or Values | Software Quality Checking |
| **Procedural Steps** | |
| 1 | Open prediction |
| 2 | Enter Input Values |
| 3 | It Shows Whether the Software Quality Good or Bad |

###### **CHAPTER 10**

###### REPORTS



**Figure 10.1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.NO** | **User Name** | **Algorithms Used** | **Percentage %** | **Software**  **Quality Good** | **Software**  **Quality Bad** |
| 1 | Sreenadh | KNN | 0.9897256 | Good | Bad |
| 2 | Karthik | KNN | 0.9725462 | Good | Bad |
| 3 | Megha Jyothi | KNN | 0.9621546 | Bad | Good |
| 4 | Yamini | KNN | 0.9610254 | Good | Bad |

###### **CHAPTER 11**

###### CONCLUSION

In this application, I used supervised Machine Learning models to predict the quality of the software. Total five ML algorithms to predict software quality they are Random Forest Classifier, Decision Tree Classifier, XG Boost Classifier, KNN. All algorithms performs well with good accuracies.

###### LIMITATIONS

Every system needs some limitation to ensure it performs as it supposed. There are also a few problems and limitations that occur throughout the development of these project which are,In this project internet must be required. Once internet connection not there we can’t perform any operation.

###### FUTURE WORK

In This system can be further extended to analyze the in depth about the software quality by plotting the graphs of Area under Curve (AUC) vs Receiver Operating characteristic Curve (ROC) and analyzing them for software quality.

**APPENDIX**

**A.USER MANUAL**

###### Download Pycharm community edition:

Here is a step by step process on how to download and install Pycharm IDE on Windows:

1. To download PyCharm visit the website https:[//www.jetbrains.com/pycharm/download/](http://www.jetbrains.com/pycharm/download/) and Click the “DOWNLOAD”

link under the Community Section.

1. Once the download is complete, run the exe to install PyCharm. The setup wizard should have started. Click “Next”.
2. On the next screen, Change the installation path if required. Click “Next”. 4.On the next screen, you can create a desktop shortcut if you want and click on “Next”.

5. Choose the start menu folder. Keep selected JetBrains and click on “Install”. 6. Wait for the installation to finish.

1. Once installation finished, you should receive a message screen that PyCharm is installed. If you want to go ahead and run it, click the “Run PyCharm Community Edition” box first and click “Finish”.
2. After you click on “Finish,” the Following screen will appear.

###### Procedure to follow for how to use this project:

**Step 1:** First we need to install all libraries that are useful to run the project.

**Step 2:** After installing all the libraries we have to type the command as **python filename.py Step 3:** After running the code we will get the software quality good (or) bad according the results.

###### BIBLIOGRAPHY

1. N.Kalaivani, Dr.R.Beena, International Journal of Pure and Applied Mathematics Volume 118 No. 20 2018, 3863-3873 ISSN: 1314-3395.
2. He, Peng, et al. "*An empirical study on software defect prediction with a simplified metric set*." Information and Software Technology 59 (2015): 170-190.
3. Yu, Xiao, et al. "*Using Class Imbalance Learning for Cross-Company Defect Prediction*." 29th International Conference on Software Engineering and Knowledge Engineering (SEKE 2017). KSI Research Inc. and Knowledge Systems Institute, 2017.
4. D. Bowes, T. Hall, and J. Petrić, "*Software defect prediction: do different classifiers find the same defects?*." Software Quality Journal, 26(2), 2018, pp. 525-552
5. X. Wang, Y. Zhang, L. Zhang and Y. Shi, "*A Knowledge Discovery Case Study of Software Quality Prediction: ISBSG Database*," 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Toronto, ON, 2010, pp. 219-222.
6. X. Wang, Y. Zhang, L. Zhang and Y. Shi, "*A Knowledge Discovery Case Study of Software Quality Prediction Based on Classification Models: ISBSG Database,*" The 11th International Symposium on Knowledge Systems Sciences (KSS 2010), 2010
7. Zimmermann, Thomas, et al. "*Cross-project defect prediction: a large scale experiment on data vs. domain vs. process.*" Proceedings of the the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering. ACM, 2009.
8. Amasaki, S., Takagi, Y., Mizuno, O., Kikuno, T.: Constructing a bayesian belief network topredict ﬁnal quality in embedded system development. IEICE Trans. Inf. Syst.8(6), 1134– 1141(2005
9. Idri, A., Abra, A.: A fuzzy logic based measures for software project similarity: validation andpossible improvements. In: Proceedings of 7th International Symposium on Software Metrics,pp. 85–96. IEEE, England, UK (2001)