**DECLARATION**

This is to certify that the project report on **CO2 EMISSION FROM VECHICLES** is a record of bonafide work done by me in the department of Master of Computer Applications, Avanthi P.G College, Affiliated to Osmania University. The reports are based on the project work done entirely by me and not copied any other source.

The results embodied in this project report have not been submitted to any other institute to institute for the award of any degree or diploma to the best of my knowledge and belief.

BANDRU VINAYSAI

130521862073

**ACKNOWLEDGEMENT**

I would like to express my sincere gratitude and indebtedness to my project guide **Ms . Annapurna** for her valuable suggestions and interest throughout course of this project.

I am also thankful to our principal **Dr.S.Apparoa** and **Mr.P.Naveen Chandra** Head of Department for Master of Computer Applications, Avanthi PG College, Hyderabad for providing excellent infrastructure and a nice atmosphere for completing this project successfully as a part of our course.

My sincere thanks to our project coordinator **Ms. Annapurna** for given valuable suggestions to fulfillment of this project and I convey my thanks to the lab staff for allowing me to use the required equipment’s whenever needed.

I sincerely acknowledge and thank all those who have directly / in-directly given their support in completion of this project work.

**ABSTRACT**

Over the past years, the demand for transport infrastructure and automotive has increased drastically. India is one of the world's largest market for automotive and contributes significantly towards the growing economy of the country. However, it has led to an exponential increase in emissions from the transport sector. One of the major greenhouse gas emitted from an automotive being **carbon dioxide (CO2)**. Hence there is a need to keenly monitor the emissions to avoid

violating the government-specified norms. This paper emphasizes on predicting the CO2 emissions from an automotive on a real-time basis. The core focus of this work is to build various Machine Learning models to predict CO2 emissions and perform a comparative study. Based on the accuracy metric the most appropriate model for this particular

application has been suggested. Further to tune, the hyperparameters of the model Grid Search method are used which results in a reduction in the training time of the model.

**CONTENTS**

**INTRODUCTION**

Vehicles are a major source of pollution, accounting for a total of 25% of annual CO2 emissions worldwide [1]. In the last decade, the adoption of hybrid vehicles (HV) on a global scale has proven the

reduced levels of CO2 emissions generated by HVs in comparison with conventional vehicles. This has sparked interest in industry and academia in finding ways to optimize and reduce vehicle emissions.

When it comes to measuring CO2 concentrations, a portable emissions monitoring system (PEMS) is the preferred choice, since they are an accurate and reasonably portable. Nevertheless, the usage of this

equipment incurs high costs, and therefore, it is mainly used by car manufacturers and environmental regulation entities. It is almost impossible to use for researchers and investigators.

Recent developments in machine learning (ML) mechanisms have led to the creation of pollution predictors. However, these have primarily been focused on conventional internal combustion engine (ICE) vehicles. The complexity and constantly changing nature of powertrain sources, coupled with several factors involved in determining the CO2 emissions from HVs, make the prediction a challenging task for traditional ML Algorithms.

**SYSTEM ANALYSIS**

**2.1 Study Of The System**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

**2.2 Input & Output**

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

* What data should be given as input?
* How the data should be arranged or coded?
* The dialog to guide the operating personnel in providing input.
* Methods for preparing input validations and steps to follow when error occur.

**OBJECTIVES**

Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data

input process and show the correct direction to the management for getting correct information from the computerized system.

It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

**OUTPUT DESIGN**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

Designing computer output should proceed in an organized, well thought out manner, the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements. Select methods for presenting information. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or

more of the following objectives.

• Convey information about past activities, current status or projections of the

Future.

• Signal important events, opportunities, problems, or warnings.

• Trigger an action.

• Confirm an action.

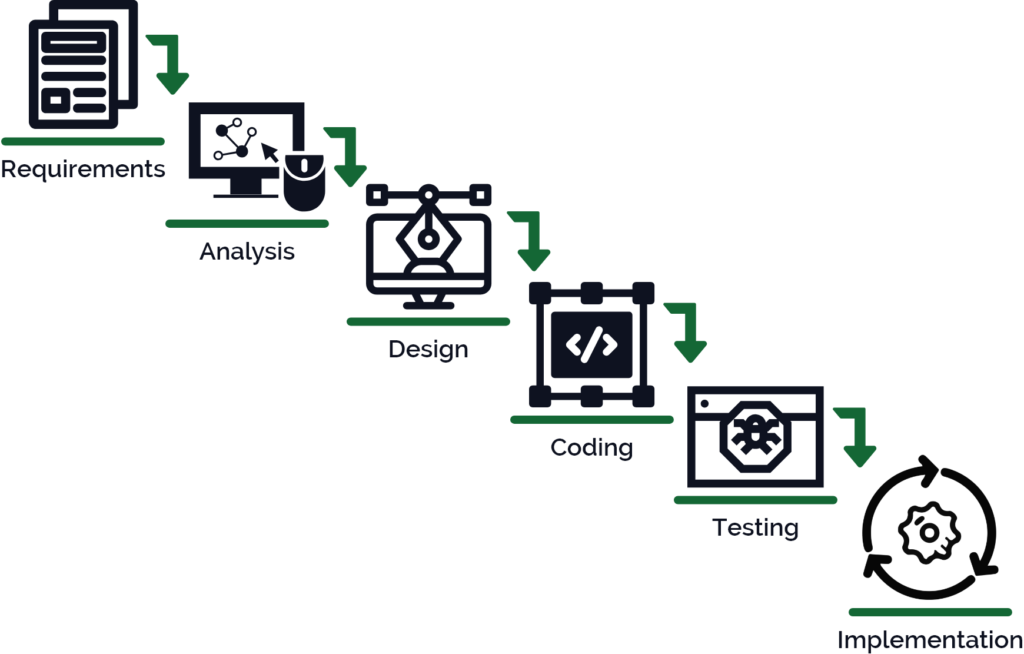
**2.3. Process Model Used with Justification**

**SDLC:**

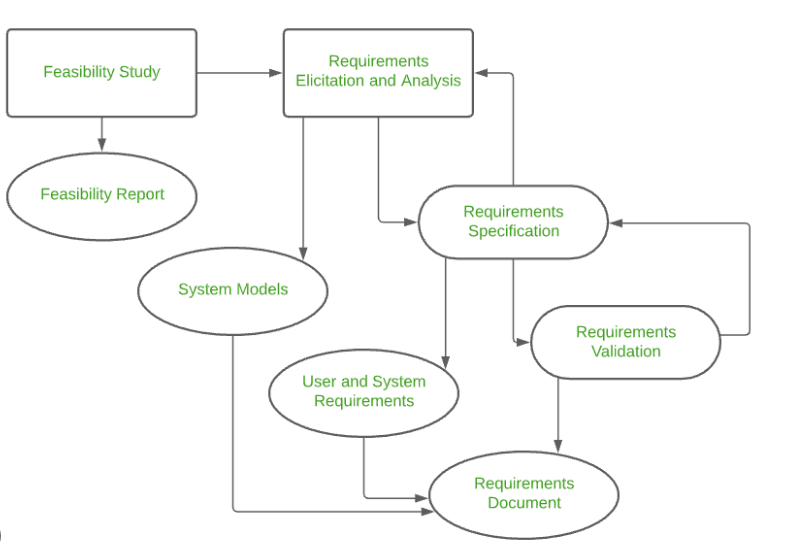
SDLC stands for Software Development Life Cycle. A software life cycle model (also termed process model) is a pictorial and diagrammatic representation of the software life cycle. A life cycle model represents all the methods required to make a software product transit through its life cycle stages. It also captures the structure in which these methods are to be undertaken. The SDLC aims to produce a high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.

**Stages in SDLC:**

* Requirement Gathering Analysis
* Designing
* Coding
* Testing
* Deployment
* Maintenance



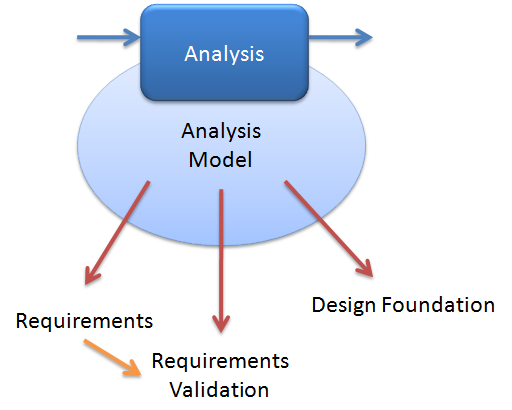
**Requirements Gathering Stage**



**Requirements Gathering Stage**

This is the first and fundamental step in the Life Cycle of Software Development. It starts with gathering the requirements from customers or clients. In most organizations, this role is taken care of by Business Analysts: A Business Analysts interacts with the customer/clients, set up daily meetings, document the requirements in Business Requirement Specifications (or Simple Business Specification), and hand over the final documented requirement to the development team. It is the responsibility of Business Analysts that every detail is captured and documented and also to make sure that everyone clearly understands the client requirements.

**Analysis Stage:**



**Fig 3. Analysis Stage**

Once the Requirement Gathering phase is completed, the next task is to analyze the requirements and get them approved by the customer/clients. This is achieved through Software Requirement Specification (SRS), which consists of all the requirements gathered and developed during the Requirements Gathering phase. This phase is mainly done by Project Managers, Business Analysts, and Consultants.

**Designing Stage:**

Once the Analysis Phase is over, next comes the need to come up with the most accurate, robust, efficient and cost-effective architecture of the product that needs to be developed. Usually, more than one design is proposed in this phase, and the best one is selected based on different parameters such as robustness, durability, timeline, cost- effectiveness, and many more! The different design architecture is generally documented in Design Document Specification or DDS.

This phase consists of 2 design approaches:

* **Low-Level Design:** This task is performed by the Senior Developers where specify the function of each module of the product architecture that has developed
* **High-Level Design:** This task is performed by Architects where they design different possible architectures of the product that has to be developed.

**Coding Stage:**

This phase is where the actual implementation of programming languages and different frameworks are being utilized for the development of the product. In this phase, all developers are involved. Developers are expected to follow certain predefined coding standards and guidelines; they are expected to complete the project modules within the defined deadline for the project. This phase is also the longest and one of the most critical phases in the Software Development Life Cycle. This phase is documented as a Source Code Document (SCD).

**Testing Stage:**

Once the Development phase is completed, the next step is to test the developed software. The developed software is sent to the testing team, where they conduct different types of testing thoroughly on the software and look for defects. If any defect is found, the testing team records and document which is again sent back to the development team for error removal. This role is taken care of by Software Testers and Quality Analysts of the company. The testing team has to make sure that each component of the software is enor-free and it works as expected.

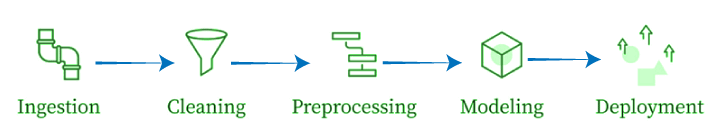
**Deployment Stage:**

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.

**Maintenance Stage:**

Once when the client starts using the developed systems, then the real issues come up and requirements to be solved from time to time. This procedure where the care is taken for the developed product is known as maintenance.

**2.4 System Architecture**



**Fig 4. Working of Propose Model**

**2.5 Modules**

**Predictive Modeling**

In order to find a decent model to predict sales we performed an extensive search of various machine learning models available in R in particular of those accessible through the caret wrapper. In the end, However, models from the h20 package yielded the best results for the task. In particular, deep learning neural networks h2o.deeplearning and gradient boosting regression trees h2o.gbm performed particularly well. An ensemble of various such models, constructed in h2oEnsemble.R forms the basis of our submission. Here, we used only the 12 most important predictors to avoid over-fitting. To include some features we may have missed with this rather small sub set of predictors we supplemented the ensemble with a deep learning neural net using 23 predictors.

Following algorithms are used

1.Linear Regression Model

2. Ridge Regression Model

3. Lasso Regression Model

4. Decision Tree Model

5. Random Forest Model

6. Support Vector Machine

**LINEAR REGRESSION:**

Linear Regression is the most commonly and widely used algorithm Machine Learning algorithm. It is used for establishing a linear relation between the target or dependent variable and the response or independent variables. The main aim of this algorithm is to find the best fit line to the target variable and the independent variables of the data. It is achieved by finding the most optimal values for all 0

**RIDGE REGRESSION:**

Ridge regression is a specialized technique used to analyze multiple regression data that is multicollinear in nature. It is a fundamental regularization technique, but it is not used very widely because of the complex science behind it. However, it is fairly easy to explore the science behind ridge regression in r if you have an overall idea of the concept of multiple regression. Regression stays the same, but in [regularization](https://www.engati.com/glossary/regularization), the way the model coefficients are determined is different.

**LASSO REGRESSION:**

Lasso regression is also called Penalized regression method. This method is usually used in machine learning for the selection of the subset of variables. It provides greater prediction accuracy as compared to other regression models. Lasso Regularization helps to increase model interpretation.

The less important features of a dataset are penalized by the lasso regression. The coefficients of this dataset are made zero leading to their elimination. The dataset with high dimensions and correlation is well suited for lasso regression.

* Lasso Regression Formula:

**D= Residual Sum of Squares or Least Squares Lambda \* Aggregate of  absolute values of coefficients**

Lambda denotes the amount of shrinkage in the lasso regression equation.

The best model is selected in a way to minimize the least-squares.

Penalizing factor is added to form a lasso regression to the least-squares. The selection of the model depends upon its ability to reduce the above loss function to its minimal value.

**DECISION TREE ALGORITHM:**

Decision trees which help to give correct output by making use of bagging mechanism. Bagging along with boosting are two of the most common ensemble techniques which intend to tackle higher variability and higher prejudice. In bagging, we have multiple base learners, or we can say base models, which in turn takes various random samples of records from the training dataset. In case of decision tree regressor are the base learners, and they are trained on the data collected by them. Decision trees are itself not accurate learners as, when it is implemented up to its full depth, mostly there are chances of over fitting with high training accuracy, but low real accuracy. So, we give out the samples of the main data file by utilizing row sampling and feature sampling with replacement technique to each of the decision trees and this method is referred to as boot strap. The result is that every model has been trained on all of these data files and then whenever we feed a test data to each of the trained one out there, the predictions estimated by each of them are combined in a way such that the final output is the mean of all of the results generated.

**RANDOM FOREST:**

Random forest is a versatile machine learning algorithm developed by Leo Breiman and Adele Cutler. It leverages an ensemble of multiple decision trees to generate predictions or classifications. By combining the outputs of these trees, the random forest algorithm delivers a consolidated and more accurate result.

Its widespread popularity stems from its user-friendly nature and adaptability, enabling it to tackle both classification and regression problems effectively. The algorithm’s strength lies in its ability to handle complex datasets and mitigate overfitting, making it a valuable tool for various predictive tasks in machine learning.

One of the most important features of the Random Forest Algorithm is that it can handle the data set containing **continuous variables,** as in the case of regression, and **categorical variables,** as in the case of classification. It performs better for classification and regression tasks. In this tutorial, we will understand the working of random forest and implement random forest on a classification task.

**SUPPORT VECTOR MACHINE:**

Support Vector Machine (SVM) is a [supervised machine learning](https://www.geeksforgeeks.org/supervised-unsupervised-learning/) algorithm used for both classification and regression. Though we say regression problems as well it’s best suited for classification. The main objective of the SVM algorithm is to find the optimal [hyperplane](https://www.geeksforgeeks.org/separating-hyperplanes-in-svm/) in an N-dimensional space that can separate the data points in different classes in the feature space. The hyperplane tries that the margin between the closest points of different classes should be as maximum as possible. The dimension of the hyperplane depends upon the number of features. If the number of input features is two, then the hyperplane is just a line. If the number of input features is three, then the hyperplane becomes a 2-D plane. It becomes difficult to imagine when the number of features exceeds three.

**SOFTWARE REQUIREMENT SPECIFICATION**

**4.1 Functional Requirements**

**User Module:** User include the people who has access to the system and hold the account in the application. They can create a account in application when there need but as of now there should enter all the data manually to calculate the old data. By Entering Day to Day transaction it will helpful to user when he need to calculate the Balances.

**4.2 Performance Requirements**

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely with the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

The requirement specification for any system can be broadly stated as given below:

* The system should be able to interface with the existing system.
* The system should be accurate.
* The system should be better than the existing system .
* The existing system is completely dependent on the user to perform all the duties.

**4.5 Introduction TO Python Framework**

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source.

Django's primary goal is to case the creation of complex, database-driven websites. Django highlights the reusability and pluggability of components, rapid development. Python is used throughout, even for settings files and data models.

**4.6 Python**

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. An interpreted language, Python has a design philosophy that emphasizes code reusability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer lines of the code than might be used in languages such as C++ or Java. It provides constructs that enable clear programming on both small and large scales. Python interpreters are available for many operating system.

**What is Python**

Python is a popular programming language. It was created by Guido van Rossum, and released in 1991.

**It is used for:**

1. web development (server-side),

2. software development,

3. mathematics,

4. system scripting.

**What can Python do**

1. Python can be used on a server to create web applications.

2. Python can be used alongside software to create workflows. 3. Python can connect to database systems. It can also read and modify files.

4. Python can be used to handle big data and perform complex mathematics.

5. Python can be used for rapid prototyping, or for production-ready software development.

**Why Python**

1.Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).

2. Python has a simple syntax similar to the English language.

3. Python has syntax that allows developers to write programs with fewer lines than some other programming languages.

4. Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.

5. Python can be treated in a procedural way, an object-orientated way or a functional way.

**Good to know**

1.The most recent major version of Python is Python 3, which we shall be using in this tutorial. However, Python 2, although not being updated with anything other than security updates, is still quite popular.

2. In this tutorial Python will be written in a text editor. It is possible to write Python in an Integrated Development Environment, such as Thonny. Pycharm, Netbeans or Eclipse which are particularly useful when managing larger collections of Python files.

**Python Syntax compared to other programming languages**

1. Python was designed for readability, and has some similarities to the English language with influence from mathematics.

2. Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.

3. Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

**SYSTEM DESIGN**

**5.1 Introduction**

Systems design is the process or art of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. One could see it as the application of systems theory to product development. There is some systems engineering.

**5.2 Normalization**

Normalization is the process of reorganizing data in a database so that it meets two basic requirements:

1. There is no redundancy of data, all data is stored in only one place.
2. Data dependencies are logical, all related data items are stored together. Normalization is important for many reasons, but chiefly because it allows databases to take up as little disk space as possible, resulting in increased performance. Normalization is also known as data normalization.

The first goal during data normalization is to detect and remove all duplicate data by logically grouping data redundancies together. Whenever a piece of data is dependent on another, the two should be stored in proximity within that data set.By getting rid of all anomalies and organizing unstructured data into a structured form, normalization greatly improves the usability of a data set. Data can be visualized more easily, insights could be extracted more efficiently, and information can be updated more quickly. As redundancies are merged together, the risk of errors and duplicates further making data even more disorganized is reduced. On top of all that, a normalized database takes space, getting rid of many disk space problems, and increasing its overall performance significantly. The three main types of normalization are listed below. Note: "N" refers to "normal form,"

**First normal form (INF)**

Tables in INF must adhere to some rules:

Every column in the table must be uniquely named.

All values in a column must pertain to the same domain.

**Second normal form (2NF)**

Tables in 2NF must be in INF and not have any partial dependency (e.g. every non- prime attribute must be dependent on the table's primary key).

**Third normal form (3NF)**

Tables in 3NF must be in 2NF and have no transitive functional dependencies on the primary key.

**5.3 Data flow diagrams**

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even handdrawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyse an existing system or model a new one. Like all the best diagrams and charts, a DFD can often visually "say" things that would be hard to explain in words, and they work for both technical and nontechnical audiences, from developer to CEO. That's why 22 DFDs remain so popular after all these years. While they work well for data flow software and systems, they are less applicable nowadays to visualizing interactive, real-time or database-oriented software or systems. Data-flow diagrams can be regarded as inverted Petri nets, because places in such networks correspond to the semantics of data memories. Analogously, the semantics of transitions from Petri nets and data flows and functions from data-flow diagrams should be considered equivalent.

**5.4 UML diagrams**

A UML diagram is a diagram based on the UML (Unified Modeling Langig with the purpose of visually representing a system along with its main actors, roles, actions, artifacts or dasses, in order to better understand, alter, maintain, or document information about the system. UML stands for Unified Modeling Language. UML is a standardized general-purpose Modeling language in the field of objectoriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business Modeling and other nonsoftware systems. The UML represents a collection of best engineering practices that have proven successful in the Modeling of large and complex systems. The UML is a very important part of developing object-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

**1.ARCHITECTURE DIAGRAM**

In software engineering, a architecture diagram in the Unified Modeling 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.

**2.USE CASE DIAGRAM**

A use case diagram in the Unified Modeling Language (UML) is a type of behav dagram defined by and created from a Use-case analysis. Its purpose is to present graphic 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 dagram is to show what system functions are performed for which actor, Roles of the actors in the system can be depicted.

**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 diagram.

**SAMPLE CODE**