**LOW LEVEL DOCUMENT**

**INSURANCE PREMIUM PREDICTION**

|  |  |
| --- | --- |
| WRITTEN BY- | SHIVANGI SINGH |

CONTENT-

1-INTRODUCTION

1.1-WHY LOW-LEVEL DOCUMENT IS REQUIRED?

1.2-SCOPE

2-ARCHITECTURE

3-ARCHITECTURE DESCRIPTION

3.1-SELECT IDE

3.2-IMPORT LIBRARIES

3.3-LOAD DATA

3.4-EDA

3.5-PREPROCESSING

3.6-DATA VISUALIZATION

3.7-PIPELINE

3.8-MODEL TRAINING

3.9-ALGORITHM

3.10-PREDICTION

3.11-DEPLOYMENT

1-INTRODUCTION

1.1-WHY LOW-LEVEL DOCUMENT IS REQUIRED?

A low-level document is essential as it provides intricate technical details and instructions for a system, software, or process. It serves as a comprehensive reference, aiding in development, troubleshooting, and maintenance. It ensures consistency in implementation, reducing errors and misinterpretations. Low-level documents empower engineers and developers with granular insights into code, algorithms, or hardware, enhancing their ability to optimize and debug effectively. Moreover, they facilitate collaboration among team members and serve as a valuable resource for future updates or modifications. In summary, low-level documents are indispensable for precision, efficiency, and the long-term sustainability of complex projects in technology and engineering.

1.2-SCOPE

The scope of a Low-Level Design (LLD) document encompasses intricate technical details, specifying system architecture, data structures, algorithms, interfaces, error handling, and resource management. It defines database schemas, security measures, performance optimization, and testing strategies. LLD outlines dependencies, scalability, and extensibility, ensuring precise guidance for developers during implementation. It serves as a crucial reference for troubleshooting, maintenance, and collaboration. LLD's comprehensive coverage enhances system reliability, security, and efficiency, while facilitating future updates and modifications. Its significance lies in its ability to transform high-level concepts into actionable, error-free code, ensuring the successful realization of complex projects in technology and engineering.

2-ARCHITECTURE

STOP

DEPLOYMENT

PREDICTION

ALGORITHMS/

ERRORS

MODEL TRAINING

BUILD PIPELINE

DATA VISUALIZATION

PREPROCESSING

PERFORM EDA

LOAD DATA

IMPORT LIBRARIES

SELECT IDE

START

3-ARCHITECTURE DESCRIPTION

3.1-SELECT IDE

Choosing the right Integrated Development Environment (IDE) is crucial for efficient software development. Consider factors such as your programming language, project type, and personal preferences. Popular options include Visual Studio Code for versatility, PyCharm for Python, and IntelliJ IDEA for Java development. Evaluate your specific needs to make an informed selection that enhances your coding experience.

3.2-IMPORT LIBRARIES

Importing libraries is a fundamental aspect of modern programming. It allows developers to access pre-written code and functionality, saving time and effort. In languages like Python, you can use the "import" statement to bring in libraries like NumPy for numerical computing or Pandas for data manipulation. These libraries extend the capabilities of your code and are essential for various tasks, from scientific computing to web development.

Properly importing and utilizing libraries is key to efficient and productive software development.

3.3-LOAD DATA

Importing libraries is a fundamental aspect of modern programming. It allows developers to access pre-written code and functionality, saving time and effort. In languages like Python, you can use the "import" statement to bring in libraries like NumPy for numerical computing or Pandas for data manipulation. These libraries extend the capabilities of your code and are essential for various tasks, from scientific computing to web development. Properly importing and utilizing libraries is key to efficient and productive software development.

3.4-EDA

This is the initial phase where you explore and analyze your dataset to understand its characteristics, patterns, and potential issues.

3.5-PREPROCESSING

In this step, you clean and prepare your data for analysis. It involves tasks like handling missing values, encoding categorical variables, and scaling or normalizing features.

3.6-DATA VISUALIZATION

Data is often visualized using charts, graphs, and plots to gain insights and identify trends or outliers

3.7-PIPELINE

A data processing pipeline is a set of data processing elements connected in series, where the output of one element is the input of the next.

3.8-MODEL TRAINING

In this phase, you build and train machine learning or statistical models using your prepared data.

3.9-ALGORITHM

Algorithms are the mathematical and statistical techniques used for modeling and prediction in data science and machine learning.

3.10-PREDICTION

After model training, you use the trained model to make predictions or classifications on new or unseen data.

3.11-DEPLOYMENT

In this final step, you integrate your model into a production environment, making it accessible for real-time predictions or applications.