**Architecture**



**Project On:**

**Title: Insurance Premium Prediction**

**By: - Nidhi Patel**

**Date: - 27/05/2023**

**Contents**

1. Abstract ………………………………………………….….. 3

2. Introduction …………………………………………………. 3

2.1 Why architecture Design Document?

2.2 Scope

2.3 Definitions

3. Technical Specification ………………………………………3

3.1 Dataset Overview

3.2 Predicting the Insurance premium

3.3 Logging

4. Technology Stack …………………………………………….4

5. Proposed Solution …………………………………………… 4

6. Workflow ……………………………………………………. 4

.

**1. Abstract**

The Architecture Design Document for Insurance Premium Prediction outlines the technical specification and proposed solution for predicting insurance premiums based on various factors. It covers the dataset overview, predicting insurance premiums, logging, technology stack, workflow, and key performance indicators (KPIs) to ensure successful implementation.

**2. Introduction**

2.1 Why Architecture Design Document?

An Architecture Design Document is essential to ensure the successful development and deployment of a project. It helps to define the technical specification, technology stack, and proposed solution for a project.

2.2 Scope

The scope of this Architecture Design Document is to provide technical specifications and a proposed solution for predicting insurance premiums based on various factors.

2.3 Definitions

Insurance Premium Prediction: The process of using machine learning algorithms to predict insurance premiums based on various factors such as age, sex, smoker, expenses, No. of children, BMI.

Dataset: A collection of data used to train and test the machine learning model.

**3. Technical Specification**

3.1 Dataset Overview

The dataset used for this project contains information on the policyholder's age, driving record, vehicle type, and other relevant factors. It is split into training and testing datasets to train and test the machine learning model.

3.2 Predicting the Insurance Premium

The insurance premium is predicted using a machine learning algorithm, which is trained on the training dataset. The model is then evaluated on the testing dataset to ensure that it performs well on unseen data.

3.3 Logging

Logs are used to track the performance of the machine learning model and errors during training and testing.

**4. Technology Stack**

The technology stack used for this project includes:

Python: The programming language used for data pre-processing and model development.

Scikit-learn: A machine learning library used for model development and evaluation.

Flask: Flask is an open-source Python library that allows users to create interactive web forms to predict insurance premiums and making it easy to share and visualize data.

MongoDB: A database used to store the dataset and model.

**5. Proposed Solution**

The proposed solution for predicting insurance premiums involves the following steps:

Data Pre-processing: The dataset is cleaned and pre-processed to remove missing values and outliers.

Model Development: A machine learning algorithm is trained on the pre-processed dataset to predict insurance premiums.

Model Evaluation: The model is evaluated on the testing dataset to ensure that it performs well on unseen data.

Web Application: A web page is created using flask to allow users to input their information and receive a predicted insurance premium.

**6. Workflow**

The workflow for this project is as follows:

Data Collection: Collect the dataset containing information on policyholders and their insurance premiums.

Data Pre-processing: Clean and pre-process the dataset to remove missing values and outliers.

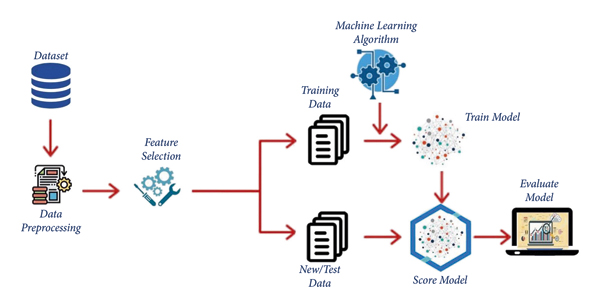
Model Development: Train a machine learning algorithm on the pre-processed dataset to predict insurance premiums.

Model Evaluation: Evaluate the model's performance on the testing dataset to ensure that it performs well on unseen data.

Web Application Development: Develop a web application using Flask to allow users to input their information and receive a predicted insurance premium.

Deployment: Deploy the web application to a server for public use.

.



*\*