GROUP 7

VEHICLE INSURANCE BASED ON DRIVING SCORE

PROJECT GUIDE:
Asst. Prof Rinu Rose George
CO-GUIDE:
Asst. Prof Abin Oommen Philip

AARIF HUSSAIN -TOC19CS001
ASHLIN ROBERT -TOC19CS022
EVA PETAL PRADEEP -TOC19CS024
KIRAN R -TOC19CS035

CONTENTS

- ★ Introduction
- ★ Background & Relevance
- ★ Literature Study
- ★ Problem Objectives & Statement
- ★ High Level Design
- ★ Design And Implementation Details Of Each Module
- ★ Results
- ★ Feasibility Study
- ★ Project Planning
- ★ Conclusion
- ★ Publications and Achievements
- ★ References

INTRODUCTION

INTRODUCTION

- ★ In India insurance is claimed by considering many factors like value of the car, age, analyzing the accident cases, this process is very **time consuming** process, which also leads to hectic workload for both the customer and the insurance company.

- ★ The system proposes a Insurance scheme based on the **driving score** by using an OBD reader which collects real time data from vehicles .

- ★ Calculated driving scores are transformed into tokens on the blockchain to prevent manipulation and ensure data integrity

BLOCKCHAIN

- ★ A Database contains a **chain of Blocks**.
- ★ **Distributed** in a **Peer-to-Peer Network**.
- ★ **Decentralised**
- ★ Data stored are **Immutable**

INTRODUCTION Contd-

FEATURES OF BLOCKCHAIN

- ★ Better Security
- ★ Immutability
- ★ Decentralised System
- ★ Faster Settlement
- ★ Distributed Ledger

BACKGROUND AND RELEVANCE

BACKGROUND AND RELEVANCE

★ Vehicle insurance based on driving scores using blockchain meets the need for personalized and fair policies that reflect individual driving behavior, using secure technology to collect data, assign scores, and tailor premiums, ultimately promoting safer driving and rewarding responsible drivers.

LITERATURE SURVEY

LITERATURE SURVEY

SURVEY 1

IMPLEMENTATION OF OBD-II VEHICLE DIAGNOSIS SYSTEM INTEGRATED WITH CLOUD COMPUTATION TECHNOLOGY

Jheng-Syu Jhou; Shi-Huang Chen; Wu-Der Tsay Kaohsiung, Taiwan; Mei-Chiao Lai

Publisher: IEEE

Year: 2019

IMPLEMENTATION OF OBD-II VEHICLE DIAGNOSIS

- ★ **OBD** stands for onboard diagnostics. An OBD reader (also called a Diagnostic scanner or scan tool) is a vehicle diagnostic device that can be used to read the error memory and data that is recorded on your vehicle systems.
- ★ This proposes a system in which is integrated with OBD-II, 5G wireless network, and cloud computing technologies which shows a real-time **vehicle status** surveillance
- ★ System is defined based upon the features like cover engine rpm, vehicle speed, coolant temperature, fault codes, and other vehicle dynamics information which is taken from OBD-II

IMPLEMENTATION OF OBD-II VEHICLE DIAGNOSIS Contd

★ The vehicle information will be transmitted to the cloud computing server via 5G wireless network for fault analysis.

★ Once cloud computing server detects fault conditions,the cloud computing server will report the fault code analysis results to the user and provide the description about repair procedure

IMPLEMENTATION OF OBD-II VEHICLE DIAGNOSIS Contd

ADVANTAGES

- Shorten the time to detect vehicle trouble condition
- High value in the applications of vehicle maintenance and fleet management

DISADVANTAGES

- Security threats are high while datas are in cloud

LITERATURE SURVEY

SURVEY 2

OVERSPEEDING & RASH DRIVING VEHICLE DETECTION

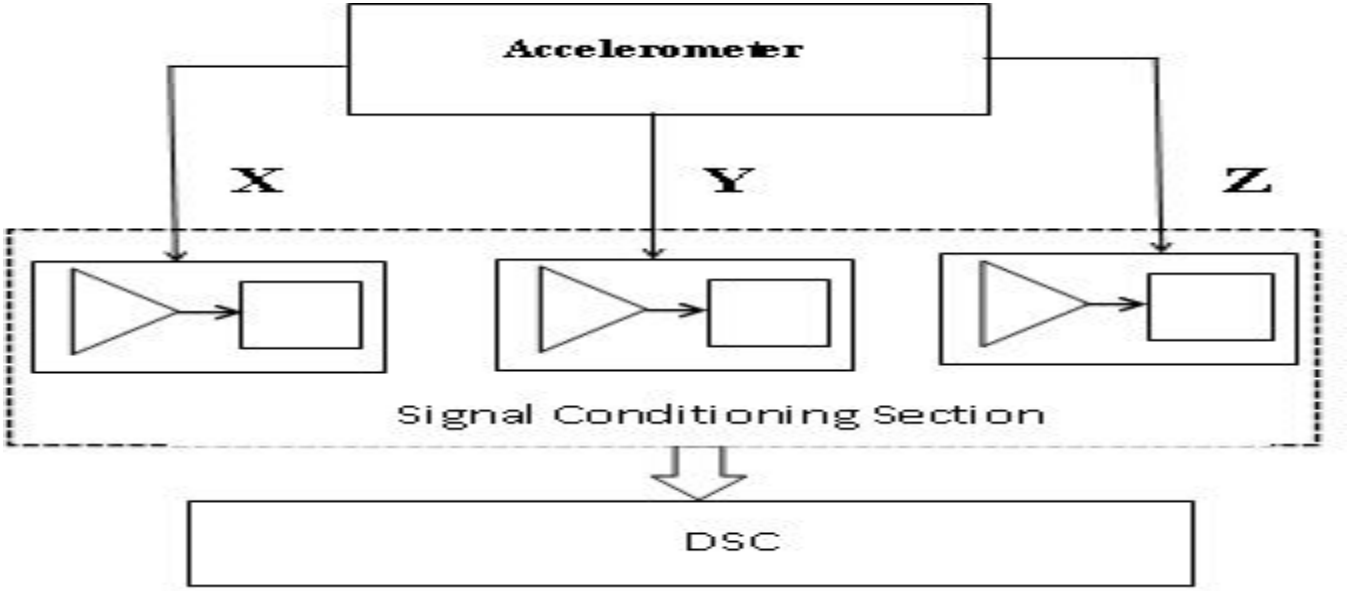
Motike Ganesh; Ivaturi Ram Pavan Kumar; Sanjay Dubey

Publisher: IEEE
Year: 2019

OVERSPEEDING & RASH DRIVING VEHICLE DETECTION

- ★ According to Indian Constitution, IPC, rash driving is an offence. So, the idea is to design a module which can detect the vehicle whenever it is rashly driven or driven above permissible speed limit, and transmit the data to the concerned authority.
- ★ Rash driving is defined in two ways
  - ○ whenever the vehicle is driven in a zigzag manner,
  - ○ and whenever the vehicle is driven at a high speed with jerks.

OVERSPEEDING & RASH DRIVING VEHICLE DETECTION Contd



BLOCK DIAGRAM

OVERSPEEDING & RASH DRIVING VEHICLE DETECTION Contd

ADVANTAGES

★ Easy to carry and easily gets installed on cars for monitoring the speed of the car

DISADVANTAGES

★ False Reading

★ Manipulation

LITERATURE SURVEY

SURVEY 3

Smart Contracts based on Private and Public Blockchains for the Purpose of Insurance Services

Veneta Aleksieva; Hristo Valchanov; Anton Huliyan

Publisher: IEEE

Year: 2020

Smart Contracts based on Private and Public Blockchains for the Purpose of Insurance Services

- ★ This paper compares the application of private and public Blockchain for insurance services through the experimental implementation of smart contracts based on Hyperledger Fabric and Ethereum.
- ★ Hyperledger Fabric is an open source private platform for industrial applications because the participants in the network are employees of one company.
- ★ Ethereum is public Blockchain which smart contracts work under ERC20 standard. Because it is related to an existing crypto-currency, it is useful for managing automatic payments to the claimer.
- ★ Participants can be anonymous, with only their public address visible. All transactions in the network are visible to all participants and they have a copy of the entire Blockchain

Smart Contracts based on Private and Public Blockchains for the Purpose of Insurance Services Contd- - -

ADVANTAGES

- Greater transparency into the claim process

DISADVANTAGES

- Transaction fees in Ethereum are high

LITERATURE SURVEY

SURVEY 4
Research on the UBI Car Insurance Rate Determination Model Based on the CNN-HVSVM Algorithm

Chun Yan; Xindong Wang,Xinhong Liu; Wei Liu; Jiahui Liu

Publisher: IEEE

Year: 2021

Research on the UBI Car Insurance Rate Determination Model Based on the CNN-HVSVM Algorithm

- ★ With the support of Internet of Vehicles technology, UBI (Usage Based Insurance) car insurance rate determination has certain guiding significance for achieving accurate pricing of car insurance rates and satisfying the personalized needs of users.
- ★ Based on the CNN (Convolutional Neural Networks) algorithm and SVM (Support Vector Machine) algorithm, this paper establishes a rating model for UBI car insurance rates.
- ★ The model first performs a series of operations such as convolutions, pooling and nonlinear activation function mapping using the CNN algorithm so that it can extract the features from the driving behavior data of UBI customers.

Research on the UBI Car Insurance Rate Determination Model Based on the CNN-HVSVM Algorithm Contd- - -

★ This model can predict the grade of UBI car insurance users more accurately and efficiently, and the prediction results are more consistent with the actual situation, which has strong applicability and flexibility.

★ Therefore, this paper proposes a UBI car insurance rate grade determination model based on the CNN-HVSVM algorithm.

ADVANTAGES

★ CNN-HVSVM algorithm can determine driver behavior more fairly and reasonably.

★ The system has good robustness.

DISADVANTAGES

★ Manual input of data.

SURVEY 5
Construction of Driving Behavior Scoring Model Based on OBD Terminal Data Analysis

Tianshi Liu; Guang Yang; Dong Shi

Publisher: IEEE
Year:2021

Construction of Driving Behavior Scoring Model Based on OBD Terminal Data Analysis

- ★ In order to accurately identify abnormal driving behaviours, this paper proposes different abnormal driving behavior recognition algorithms.
- ★ This paper constructs a speeding behavior recognition algorithm using different threshold values.
- ★ Here the system selects the driver's driving statistics from the OBD service platform as basis for analysis and counts the number of speeding behaviors, rapid accelerations, rapid deceleration, rapid braking and sharp turns.
- ★ In this method, the paper selects part of the real diving data of drivers, uses a driving scoring model to score their driving behaviour, and verifies the reliability of the constructed driving scoring model with case analysis.

Construction of Driving Behavior Scoring Model Based on OBD Terminal Data Analysis Contd- - -

ADVANTAGES

★ Reliable technical solution for the analysis of drivers driving behaviors.

★ Grasping the characteristics of the driver's driving behavior is the key to solving existing traffic problems.

DISADVANTAGES

★ Data in the OBD data can be **tampered.**

PROBLEM STATEMENT & OBJECTIVES

PROBLEM DEFINITION

★ Usually vehicle insurance is claimed by going through various process and it is very much **hectic** process for the customers

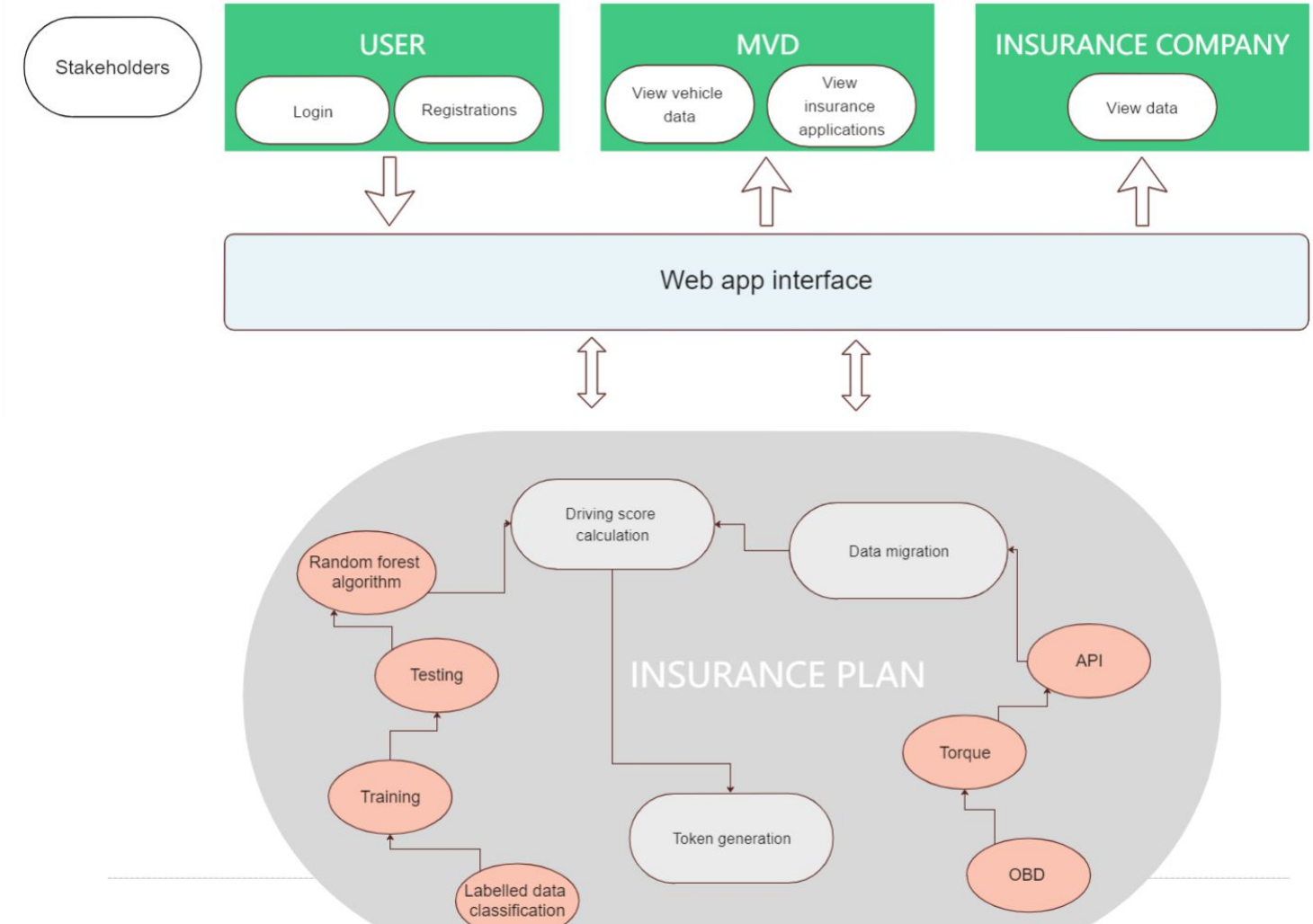
★ Our innovative system proposes an insurance scheme based on driving scores by utilizing an OBD reader to gather real-time data from vehicles, allowing us to offer an **add-on feature** to **traditional policies** that simplifies procedures and enhances the overall insurance experience for our customers.

OBJECTIVES

- ★ **Insurance fraud** is effectively prevented by securely storing the driving scores on the blockchain, ensuring transparency and immutability in the conversion process.
- ★ Encourages safer driving by offering rewards and benefits to policyholders who drive responsibly and carefully.
- ★ Speeds up claims settlement, making it **faster and easier** for policyholders to get their claims processed.

HIGH LEVEL DESIGN

HIGH LEVEL DESIGN



DESIGN AND IMPLEMENTATION DETAILS OF EACH MODULE

MODULE WISE WORK PLAN

★ MODULE 1

DATA COLLECTION & MIGRATION

★ MODULE 2

DRIVING SCORE

★ MODULE 3

INSURANCE

★ MODULE 4

BLOCKCHAIN / TOKEN GENERATION

DESIGN AND IMPLEMENTATION

1. DATA COLLECTION

- ★ The OBD data is collected by using Torque app which is connected to the OBD reader via wifi or bluetooth .Some the various attributes obtained from OBD are acceleration(x,y,z),engine load etc and their corresponding values are obtained
- ★ The values change according to the driving behaviour
- ★ Data collection is done by capturing OBD data from both good and bad driving instances, enabling us to differentiate and define the dataset for good driving behavior and bad driving behavior.

DATA COLLECTION

DESIGN AND IMPLEMENTATION

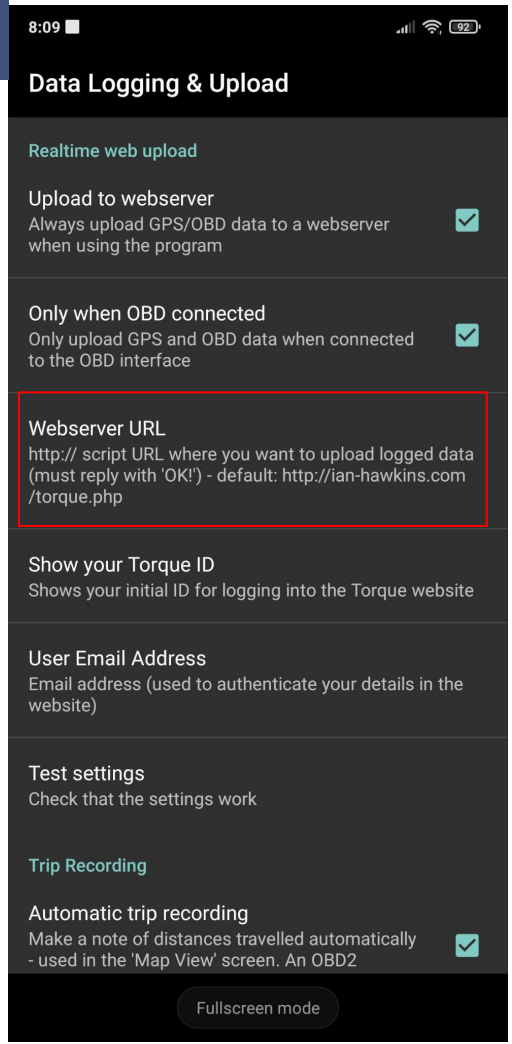
- Data Collection

GPS Time	Device Tir	Longitude	Latitude	GPS Speed	Horizonta	Altitude	Bearing	G(x)	G(y)	G(z)	G(calibrat	Acceleratic	Acceleratic	Acceleratic	Acceleratic	Engine Loa	Engine RPI	Relative Th	Speed (OB	Throttle Pc	Torque(Nm)
Thu May 0	02:04.0	76.43349	9.887842	0	3.790093	-60.9066	0	1.25	1.85	14.08	0.46	0.45	0.13	0.19	1.34	24.31	728.25	0	0	5.49	-
Thu May 0	02:04.1	76.43349	9.887842	0	3.790093	-60.9066	0	-3.56	2.06	8.78	0	-0.01	-0.36	0.21	0.8	24.71	728.25	0	0	5.49	-
Thu May 0	02:04.2	76.43349	9.887842	0	3.790093	-60.9066	0	2.47	-0.39	15.59	0.62	-0.01	-0.36	0.21	0.8	25.1	728.5	0	0	5.49	-
Thu May 0	02:04.3	76.43349	9.887842	0	3.790093	-60.9066	0	2.16	1.86	13.21	0.39	-0.62	0.2	-0.01	0.23	25.49	709.5	0	0	5.49	-
Thu May 0	02:04.4	76.43349	9.887844	0	3.790093	-61.4669	0	1.14	1.31	8.88	-0.07	-0.08	0.12	0.13	0.8	25.49	719.25	0	0	5.49	-
Thu May 0	02:04.5	76.43349	9.887844	0	3.790093	-61.4669	0	-0.28	2.51	8.96	-0.04	-0.05	-0.03	0.26	0.81	25.49	692	0	0	5.49	-
Thu May 0	02:04.6	76.43349	9.887844	0	3.790093	-61.4669	0	-0.18	2.5	9.16	-0.02	-0.05	-0.03	0.26	0.81	25.88	652.5	0	0	5.49	-
Thu May 0	02:04.7	76.43349	9.887844	0	3.790093	-61.4669	0	0.84	2.76	9.63	0.03	0.08	0.13	0.27	0.94	26.27	652.5	0	0	5.49	-
Thu May 0	02:04.8	76.43349	9.887844	0	3.790093	-61.4669	0	0.21	3.62	9.51	0.05	0.04	0.02	0.37	0.87	27.06	634.75	0	0	5.49	-
Thu May 0	02:04.9	76.43349	9.887844	0	3.790093	-61.4669	0	0.04	2.33	9.58	0.02	0.01	0	0.24	0.88	27.84	620.25	0	0	5.49	-
Thu May 0	02:05.0	76.43349	9.887844	0	3.790093	-61.4669	0	0	2.27	9.52	0.01	0.01	0	0.24	0.88	28.63	601	0	0	5.49	-
Thu May 0	02:05.1	76.43349	9.887844	0	3.790093	-61.4669	0	0.05	2.41	9.58	0.02	0.01	0.01	0.24	0.88	28.63	530	0	0	5.49	-
Thu May 0	02:05.2	76.43349	9.887844	0	3.790093	-61.4669	0	0.01	2.37	9.47	0.01	0	0	0.24	0.87	29.8	529.25	0	0	5.49	-

DESIGN AND IMPLEMENTATION

DATA MIGRATION

- Implementation of the Torque app in the car dashboard provides a convenient solution.
- The Torque app offers an option to directly send the driven data to a designated database.
- A notable feature of the Torque app is the ability to configure a website URL to specify the target database.
- This configuration enables seamless and efficient transfer of the data collected by the Torque app.
- The Torque app acts as a bridge, transmitting the data to the specified database for storage and analysis



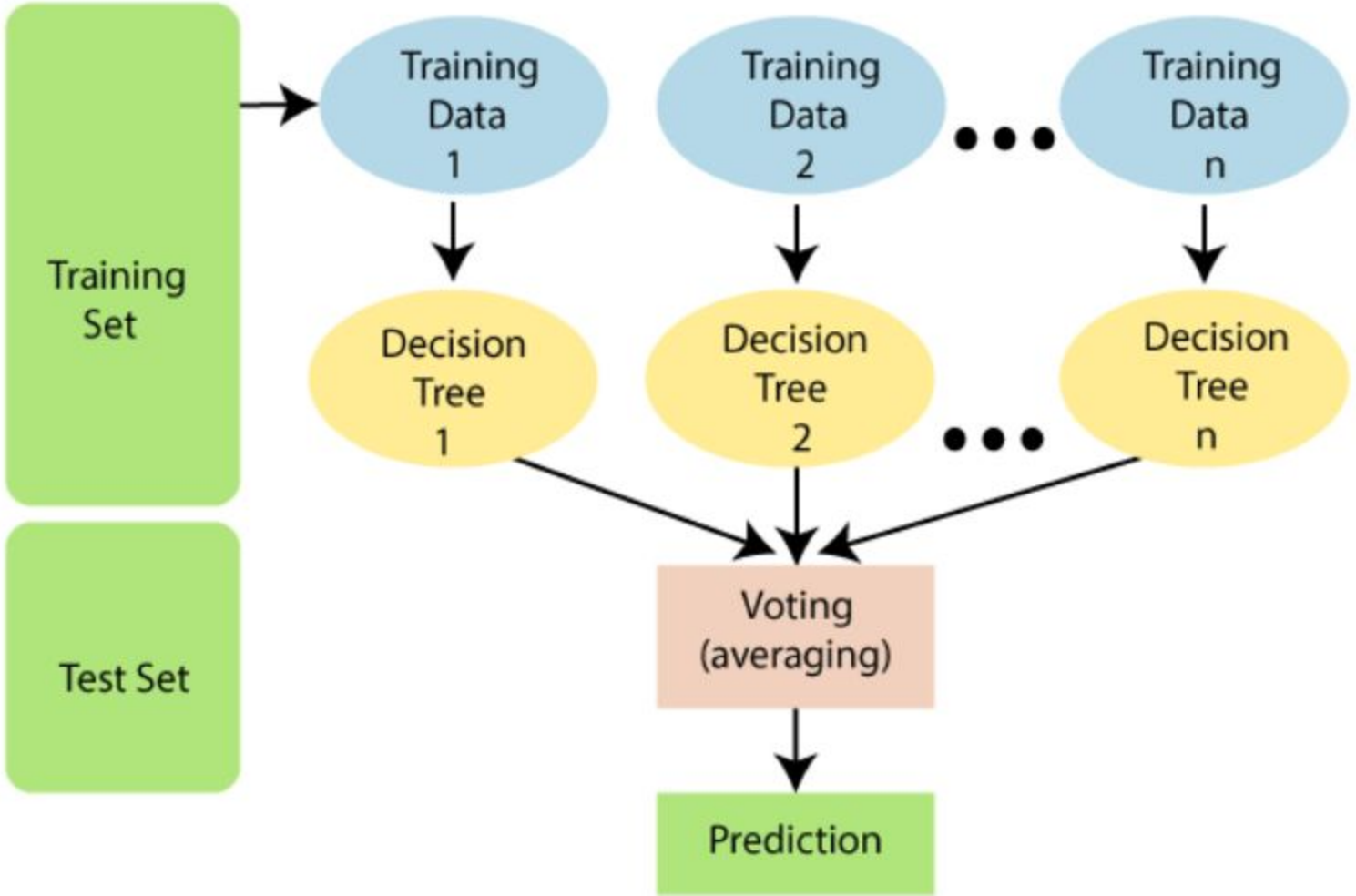
DATA MIGRATION

DESIGN AND IMPLEMENTATION

2.DRIVING SCORE

- Random Forest is a widely used machine learning algorithm in supervised learning.
- It is versatile and can handle both Classification and Regression problems.
- Random Forest operates as an ensemble learning method and acts as a classifier.
- It consists of multiple decision trees built on different subsets of the original dataset.
- The algorithm aggregates the outputs of these individual trees, often through averaging.
- By leveraging this aggregation process, Random Forest greatly improves predictive accuracy.
- It is known for its effectiveness in various domains of machine learning.

Random Forest Algorithm



DESIGN AND IMPLEMENTATION

- ★ Data from the CSV file is used to determine the driving score.
- ★ Time intervals of rapid acceleration and harsh braking are identified and consolidated into a single dataset representing bad driving.
- ★ Good driving is determined based on linear acceleration and normal driving behaviors.
- ★ Two datasets are created: one for good driving and one for bad driving.

DESIGN AND IMPLEMENTATION

- ★ The good driving dataset is labeled as 1, and the bad driving dataset is labeled as 0.
- ★ The Random Forest algorithm is applied to these labeled datasets.
- ★ The Random Forest model is trained on the labeled datasets to learn patterns distinguishing between good and bad driving.
- ★ A separate test dataset is provided for predicting the driving score.
- ★ The model assigns predicted driving scores to the test dataset based on the learned patterns.
- ★ The predicted driving scores represent evaluations of the driving behavior in the test dataset.

DESIGN AND IMPLEMENTATION

ALGORITHM(Random forest)

Input: Training dataset (X_train, y_train), number of trees (n_estimators), maximum tree depth (max_depth), and other hyperparameters.

1. Initialize an empty list to hold the ensemble of decision trees.
2. For each tree in the desired number of trees (n_estimators):
   a. Sample a bootstrap dataset by randomly selecting instances with replacement from the original training dataset (X_train, y_train). The bootstrap sample is of the same size as the original dataset.
   b. Create a decision tree using the bootstrap dataset, limiting the depth to the specified maximum tree depth (max_depth). The tree is grown by recursively splitting the data based on randomly selected features at each node.
   c. Add the decision tree to the ensemble.
3. Return the ensemble of decision trees.

DESIGN AND IMPLEMENTATION

ALGORITHM(Random forest)

4. During the training process, the decision trees in the Random Forest algorithm are trained independently but share common hyperparameters. This parallel training ensures diversity among the trees and allows for efficient parallelization.
5. After training, the Random Forest ensemble can be used to make predictions on new, unseen data by aggregating the outputs of all the decision trees (e.g., majority voting for classification or averaging for regression).

Training code

In [1]:
```
import pandas as pd
```

In [2]:
```
df=pd.read_csv("dataset.csv")
df.head()
```

Out[2]:

	Horizontal Dilution of Precision	G(x)	G(y)	G(z)	G(calibrated)	Acceleration Sensor(Total)(g)	Acceleration Sensor(X axis)(g)	Acceleration Sensor(Y axis)(g)	Acceleration Sensor(Z axis)(g)	Engine Load(%)	Engine RPM(rpm)	label
0	18.51466	0.37	5.09	8.21	0.00	-0.03	0.02	0.52	0.71	39.22	1436.5	good
1	18.51466	-0.27	4.43	8.43	0.01	-0.03	-0.03	0.45	0.76	37.65	1468.5	good
2	17.99460	0.74	5.15	8.47	0.00	-0.02	0.12	0.56	0.70	38.82	1467.0	good
3	17.99460	0.96	5.55	7.44	0.00	-0.05	0.05	0.52	0.69	38.04	1447.0	good
4	17.99460	1.07	5.20	8.45	0.04	-0.02	0.07	0.49	0.75	37.65	1447.0	good

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21759 entries, 0 to 21758
Data columns (total 12 columns):
 #   Column                              Non-Null Count  Dtype
---  ------                              --------------  -----
 0    Horizontal Dilution of Precision   21759 non-null  float64
 1    G(x)                               21759 non-null  float64
 2    G(y)                               21759 non-null  float64
 3    G(z)                               21759 non-null  float64
 4    G(calibrated)                      21759 non-null  float64
 5   Acceleration Sensor(Total)(g)       21759 non-null  float64
 6   Acceleration Sensor(X axis)(g)      21759 non-null  float64
 7   Acceleration Sensor(Y axis)(g)      21759 non-null  float64
 8   Acceleration Sensor(Z axis)(g)      21759 non-null  float64
 9   Engine Load(%)                      21759 non-null  float64
 10  Engine RPM(rpm)                     21759 non-null  float64
 11  label                               21759 non-null  object
dtypes: float64(11), object(1)
memory usage: 2.0+ MB
```

In [4]:
```
df.dropna(inplace=True)
```

Inserting training dataset

Training code

In [4]:
```
df.dropna(inplace=True)
```

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 21759 entries, 0 to 21758
Data columns (total 12 columns):
 #   Column                            Non-Null Count  Dtype
---  ------                            --------------  -----
 0    Horizontal Dilution of Precision  21759 non-null  float64
 1    G(x)                              21759 non-null  float64
 2    G(y)                              21759 non-null  float64
 3    G(z)                              21759 non-null  float64
 4    G(calibrated)                     21759 non-null  float64
 5   Acceleration Sensor(Total)(g)      21759 non-null  float64
 6   Acceleration Sensor(X axis)(g)     21759 non-null  float64
 7   Acceleration Sensor(Y axis)(g)     21759 non-null  float64
 8   Acceleration Sensor(Z axis)(g)     21759 non-null  float64
 9   Engine Load(%)                     21759 non-null  float64
 10  Engine RPM(rpm)                    21759 non-null  float64
 11  label                              21759 non-null  object
dtypes: float64(11), object(1)
memory usage: 2.2+ MB
```

In [6]:
```
x=df.iloc[:,:-1]
x.head()
```

Out[6]:

	Horizontal Dilution of Precision	G(x)	G(y)	G(z)	G(calibrated)	Acceleration Sensor(Total)(g)	Acceleration Sensor(X axis)(g)	Acceleration Sensor(Y axis)(g)	Acceleration Sensor(Z axis)(g)	Engine Load(%)	Engine RPM(rpm)
0	18.51466	0.37	5.09	8.21	0.00	-0.03	0.02	0.52	0.71	39.22	1436.5
1	18.51466	-0.27	4.43	8.43	0.01	-0.03	-0.03	0.45	0.76	37.65	1468.5
2	17.99460	0.74	5.15	8.47	0.00	-0.02	0.12	0.56	0.70	38.82	1467.0
3	17.99460	0.96	5.55	7.44	0.00	-0.05	0.05	0.52	0.69	38.04	1447.0
4	17.99460	1.07	5.20	8.45	0.04	-0.02	0.07	0.49	0.75	37.65	1447.0

Dropna Function

Training code

In [7]:
```
y=df.iloc[:,-1:]
y.head()
```

Out[7]:

	label
0	good
1	good
2	good
3	good
4	good

In [8]:
```
from sklearn.preprocessing import LabelEncoder
```

In [9]:
```
le=LabelEncoder()
le.fit(y)
y=le.transform(y)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed w
hen a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
```

In [10]:
```
from imblearn.over_sampling import RandomOverSampler
```

In [11]:
```
ros = RandomOverSampler(random_state=42)
```

In [12]:
```
x_res, y_res = ros.fit_resample(x, y)
```

In [13]:
```
from collections import Counter
print('Resampled dataset shape %s' % Counter(y_res))
```

```
Resampled dataset shape Counter({1: 15614, 0: 15614})
```

In [14]:
```
from sklearn.model_selection import train_test_split
```

In [15]:
```
x_train, x_test, y_train, y_test = train_test_split(x_res, y_res, test_size=0.33, random_state=42)
```

Training code

```
Resampled dataset shape Counter({1: 15614, 0: 15614})
```

In [14]:
```
from sklearn.model_selection import train_test_split
```

In [15]:
```
x_train, x_test, y_train, y_test = train_test_split(x_res, y_res, test_size=0.33, random_state=42)
```

In [16]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [17]:
```
clf = RandomForestClassifier(max_depth=2, random_state=0)
```

In [18]:
```
clf.fit(x_train, y_train)
```

Out[18]:
```
RandomForestClassifier(max_depth=2, random_state=0)
```

In [19]:
```
y_pred=clf.predict(x_test)
```

In [20]:
```
from sklearn.metrics import accuracy_score
```

In [21]:
```
print("accuracy score:",accuracy_score(y_test,y_pred))
```

```
accuracy score: 0.97516010091209
```

In [22]:
```
import joblib
```

In [23]:
```
joblib.dump(clf,"model.joblib")
joblib.dump(le,"label.joblib")
```

Out[23]:
```
['label.joblib']
```

In [30]:
```
x_test.iloc[:1,:].to_csv("test.csv",index=False)
```

Testing code

In [1]:
```
import pandas as pd
```

In [2]:
```
df=pd.read_csv("log.csv")
df.head()
```

Out[2]:

	Horizontal Dilution of Precision	G(x)	G(y)	G(z)	G(calibrated)	Acceleration Sensor(Total)(g)	Acceleration Sensor(X axis)(g)	Acceleration Sensor(Y axis)(g)	Acceleration Sensor(Z axis)(g)	Engine Load(%)	Engine RPM(rpm)
0	10.526792	1.70	4.39	8.25	-0.02	0.02	0.10	0.51	0.78	15.69	707.00
1	10.526792	1.18	4.75	8.25	-0.01	-0.02	0.16	0.48	0.74	15.69	667.75
2	10.526792	0.83	4.95	8.41	0.01	0.00	0.08	0.50	0.76	15.69	700.75
3	10.526792	1.63	4.05	8.94	0.02	0.01	0.17	0.41	0.81	15.69	709.00
4	10.526792	1.27	4.25	8.32	-0.03	0.01	0.17	0.41	0.81	15.69	702.75

In [3]:
```
"""df=pd.DataFrame({'Horizontal Dilution of Precision':[9.935046],'G(x)':[0.98],'G(y)':[0.64],'G(z)':[9.68],
                    'G(calibrated)':[0.08],'Acceleration Sensor(Total)(g)':[-0.01],'Acceleration Sensor(X axis)(g)':[0.09],
                    'Acceleration Sensor(Y axis)(g)':[0.06],'Acceleration Sensor(Z axis)(g)':[0.88],
                    'Engine Load(%)':[29.41],'Engine RPM(rpm)':[901.0]})
df.head()"""
```

Out[3]:
```
'"df=pd.DataFrame({\'Horizontal Dilution of Precision\':[9.935046],\'G(x)\':[0.98],\'G(y)\':[0.64],\'G(z)\':[9.68],\n
\'G(calibrated)\':[0.08],\'Acceleration Sensor(Total)(g)\':[-0.01],\'Acceleration Sensor(X axis)(g)\':[0.09],\n
\'Acceleration Sensor(Y axis)(g)\':[0.06],\'Acceleration Sensor(Z axis)(g)\':[0.88],\n                    \'Engine Load(%)\':[29.4
1],\'Engine RPM(rpm)\':[901.0]})\ndf.head()'
```

Inserting Testing data

Testing code

In [4]:
```
import joblib
```

In [5]:
```
model=joblib.load("model.joblib")
le=joblib.load("label.joblib")
```

In [6]:
```
pred=le.inverse_transform(model.predict(df))[0]
print("Result:",pred)
```

```
Result: bad
```

In [7]:
```
bad_score,good_score=model.predict_proba(df)[0]
```

In [8]:
```
bad_score*100
```

Out[8]:
```
56.992965495978865
```

In [9]:
```
good_score*100
```

Out[9]:
```
43.00703450402113
```

3.INSURANCE

By following these steps, you can implement insurance payment amount calculation and claim processing based on driving score criteria:

1. **Insurance Payment Amount Calculation**:

Here are the insurance payment amount calculations based on driving score ranges:

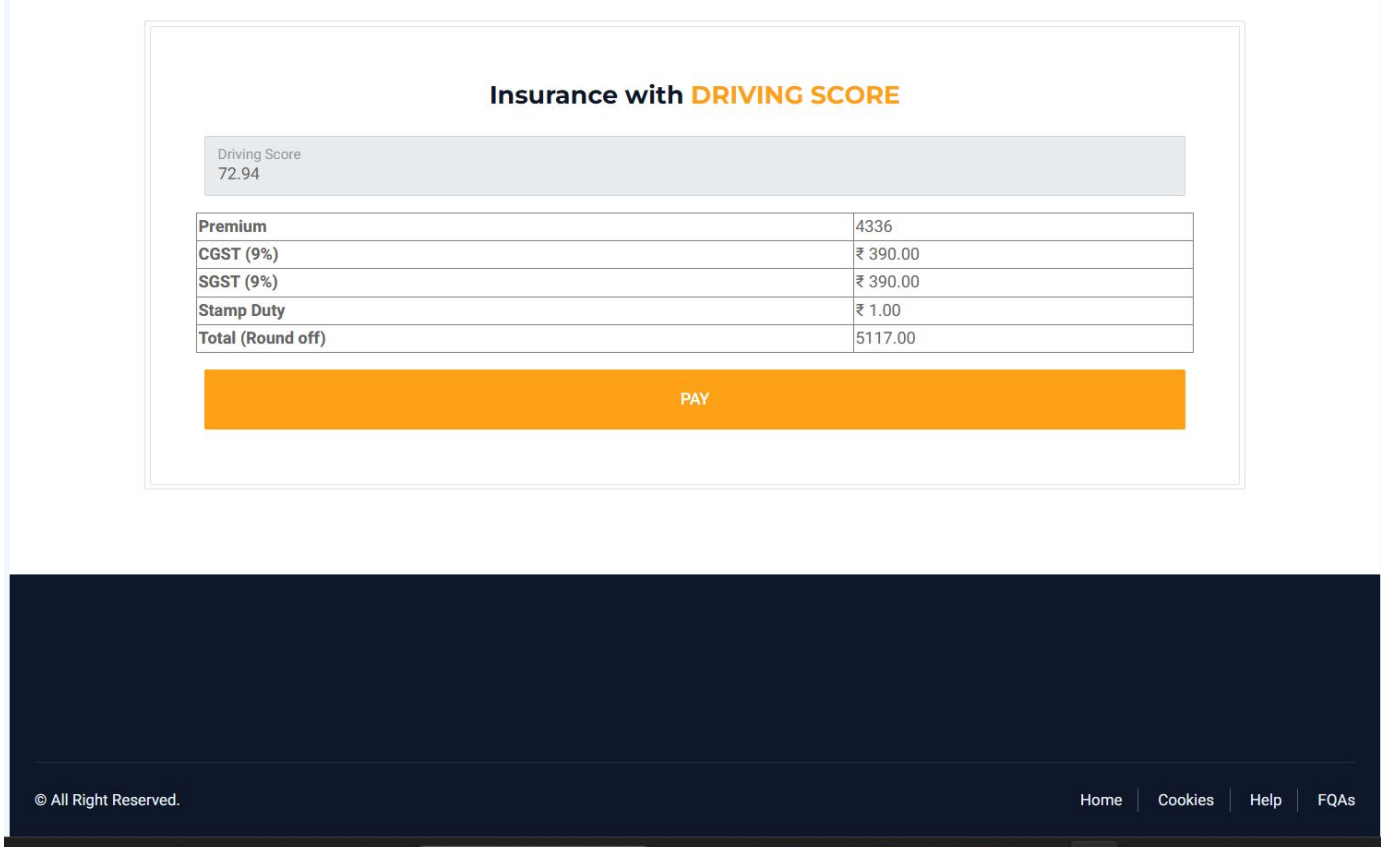
★ - 80 or higher: Pay 75% of the actual insurance cost.
★ - 70 to 80: Pay 80% of the actual insurance cost.
★ - 65 to 70: Pay full amount of the actual insurance cost.
★ - 60 to 65: Pay 5% extra of the actual insurance cost.
★ - Less than 60: Pay 15% extra of the actual insurance cost.

DESIGN AND IMPLEMENTATION

2. Claim Processing:

Here are the steps for claim processing based on driving scores:

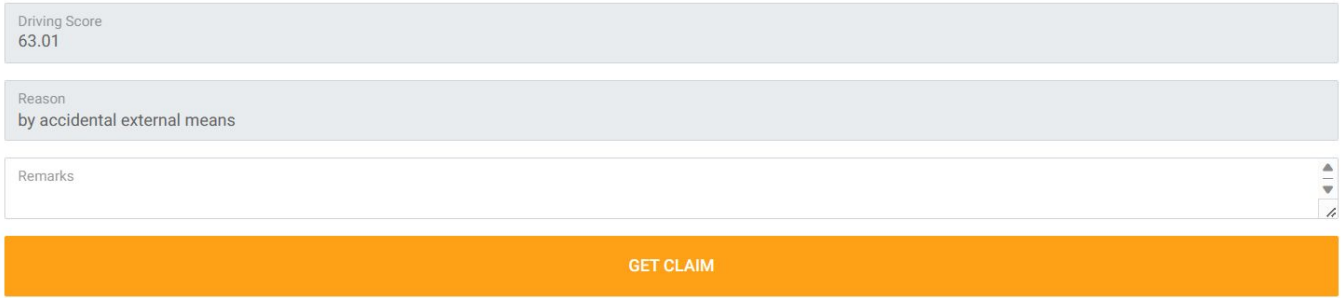
- ★ 1. Retrieve the driving score for the specific day related to the claim.
- ★ 2. Evaluate the driving score against the claim criteria.
- ★ 3. If the driving score is greater than 70: Apply a 20% bonus to the claim amount.
- ★ 4. If the driving score is less than 50: Deduct 20% from the claim amount.
- ★ 5. If the driving score is between 50 and 70: Pay the full claim amount.



Insurance Payment Amount

INSURANCE POLICY

APPLY INSURANCE



Claim Details

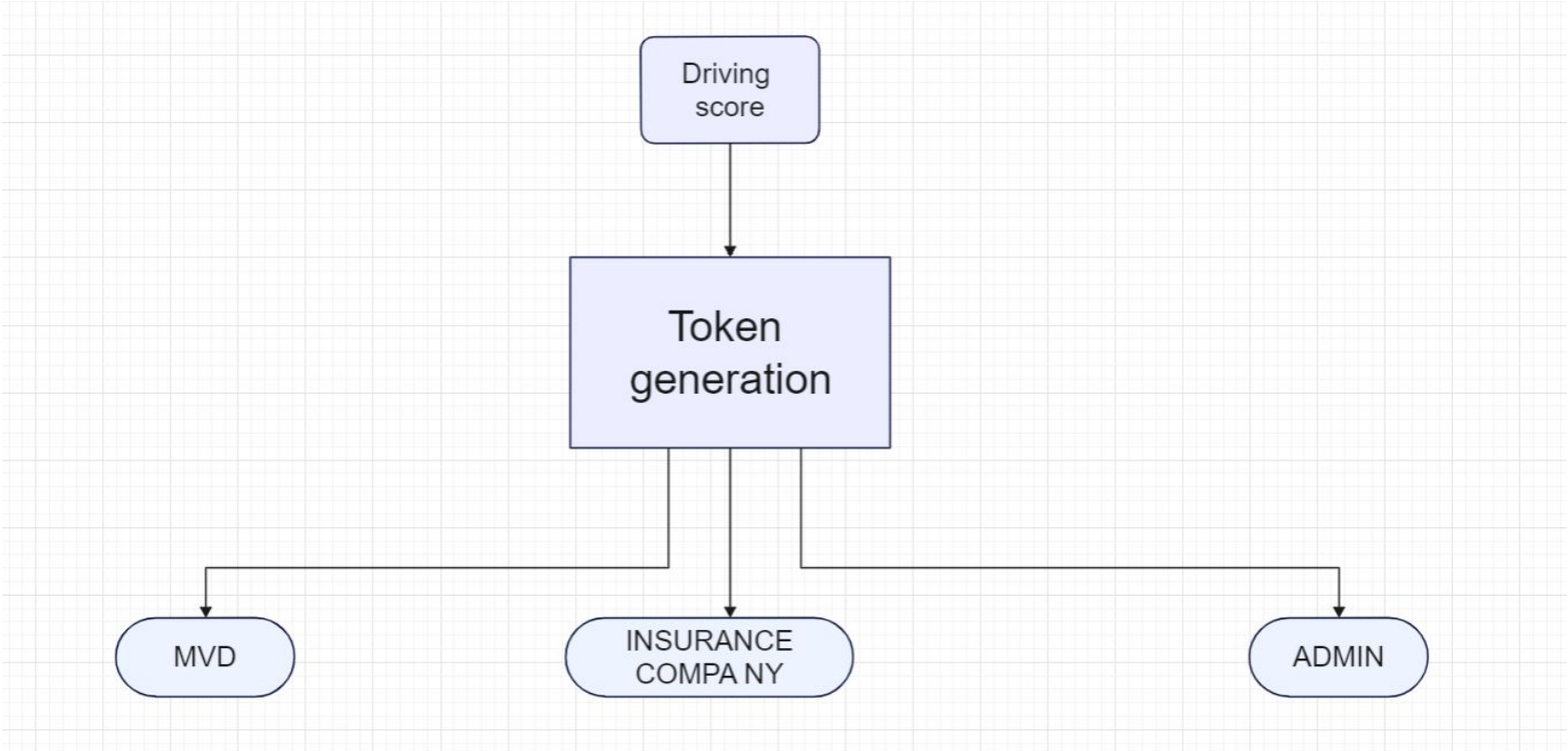
Driving Score	% of depreciation
x >= 70%	Bonus claim(+20%)
x <= 70%	Claim amount(-20%)
50% < x < 70%	Claim amount(-20%)

Nb: x denote driving score

Claim Processing

DESIGN AND IMPLEMENTATION

4. BLOCKCHAIN /TOKEN GENERATION



TOKEN GENERATION

Step 1: Initialize the blockchain
- Create an instance of the BlockChain class.
- Call the initialize() method on the BlockChain instance.

Step 2: Create the genesis block
- Call the createGenesisBlock() method on the BlockChain instance.
- Store the genesis block in a variable.

DESIGN AND IMPLEMENTATION

Step 3: Add blocks to the chain
- Create a new block with desired data.
- Set the previous hash of the new block as the hash of the last block in the chain.
- Add the new block to the chain.

Step 4: Write the final block's hash to a file
- Get the last block in the chain.
- Write the hash of the last block to a file named "block.txt".

DESIGN AND IMPLEMENTATION

HYPERLEDGER FABRIC

Bootstrap the Hyperledger Fabric network using Microfab:

1. Choose Microfab as the preferred approach, a containerized Hyperledger Fabric runtime provided by IBM Blockchain Platform for development environments.

2. Ensure the necessary configuration ready before running the Microfab command.

3. Export the configuration as an environment variable(MICROFAB_CONFIG).

4. Configure Microfab settings according to use case, such as defining the port for the Microfab container.

5. Specify the endorsing organizations and peers involved in the network.

6. Execute the Microfab command to start the network bootstrap process.

DESIGN AND IMPLEMENTATION

During the **network bootstrap:**

1.Peers, such as the MVD peer and Insurance company peer, serve as hosts for ledger instances and smart contracts, storing data within their instances.
2.Endorsing peers validate and endorse transaction proposals, while committing peers add verified transactions to the blockchain.
3.Channels facilitate secure communication between peers, orderers, and applications, ensuring private transactions.
4.Smart contracts, known as chaincode, encapsulate business logic and rules within peers.
5.The ledger acts as a repository for maintaining a comprehensive history of changes.

RESULTS

RESULTS

★ Implementing vehicle insurance based on driving scores using blockchain results in personalized, **fair policies**, promoting road safety, preventing fraud, and enhancing data integrity, **security**, and transparency in the insurance ecosystem.

Department of

Computer Science and Engineering

PROJECT

B.Tech (2019 - '23 Batch)

Vehicle Insurance Based On Driving Score Using BlockChain

Project Team:

Aarif Hussain A Nassar (TOC19CS002)

Ashlin Robert (TOC19CS022)

Kiran R (TOC19CS035)

Eva Petal Pradeep (TOC19CS024)

Project Guide: Mrs.Rinu Rose George

Toc H INSTITUTE OF SCIENCE & TECHNOLOGY

TIST

Arakkuparambu P.O | Mulanthuruthy | Ernakulam | Kerala 682313

AFFILIATIONS, RECOGNITIONS & ACCREDITATIONS

FEASIBILITY STUDY

REQUIREMENTS

SOFTWARE TOOLS

- ★ **Torque App**
- ★ **Jupyter Notebook**
- ★ **Android Studio**
- ★ **XAMPP**

HARDWARE TOOLS

- ★ **OBD**
- ★ **Android Tab**

PROJECT PLANNING

PROJECT PLANNING Contd - -

WORK DIVISION

1)OBD Data Analysis	Aarif Hussain
2)Driving Score Calculation	Kiran R
3)Token Generation	Ashlin Robert
4)User Module	Eva Petal Pradeep

PUBLICATIONS AND ACHIEVEMENTS

PUBLICATIONS AND ACHIEVEMENTS

- Applied for google developers startup Bootcamp
- Applied for CSI In-App
- Paper Published on Institution of Electronics and Telecommunication Engineers (IETE)

CONCLUSION

CONCLUSION

- ★ On the whole, the system aims to provide insurance for the cars based on driving score
- ★ Insurance Claimage is more faster and reliable through this system
- ★ The Insurance scams and fraudsters cannot manipulate the claimage systems
- ★ Resale of cars through this system is easier and trustworthy since data cannot be tampered

REFERENCES

REFERENCES

[1] Jheng-Syu Jhou; Shi-Huang Chen; Wu-Der Tsay; Mei-Chiao Lai: “The Implementation of OBD-II Vehicle Diagnosis System Integrated with Cloud Computation Technology”. 2019 Second In-ternational Conference on Robot, Vision and Signal Processing 12 June 2019

[2] Vangala Praveen Kumar; Kampati Rajesh; Motike Ganesh; Ivaturi Ram Pavan Kumar; SanjayDubey “Overspeeding and Rash Driving Vehicle Detection System” 2019 Texas Instruments IndiaEducators’ Conference (TIIEC)17 April 2020

[3] Puntavut Lertpunyavuttikul; Pinyarat Chuenprasertsuk; Sorayut Glomglome “Usage-based In-surance Using IoT Platform” 21st International Computer Science and Engineering Conference(ICSEC) 23 August 2019

[4]Veneta Aleksieva; Hristo Valchanov; Anton Huliyan “Smart Contracts based on Private and Public Blockchains for the Purpose of Insurance Services” International Conference Automatics and Informatics (ICAI) 07 January 2021

REFERENCES Contd - -

[5] Chun Yan; Xindong Wang; Xinhong Liu; Wei Liu; Jiahui Liu “Research on the UBI Car Insurance Rate Determination Model Based on the CNN-HVSVM Algorithm” IEEE Access (Volume: 8) 02 September 2020

[6]Tianshi Liu; Guang Yang; Dong Shi “Construction of Driving Behavior Scoring Model based on OBD Terminal Data Analysis” 2020 5th International Conference on Information Science, Computer Technology and Transportation (ISCTT) 04 March 2021

[7]H. T. D. Samarasinghe; N. A. D. M Herath; H. S. S. Dabare “Vehicle Insurance Policy Document Summarizer,AI Insurance Agent and On-The-Spot Claimer” 2021 6th International Conference for Convergence in Technology (I2CT) 10 May 2021

[8]Veneta Aleksieva; Hristo Valchanov; Anton Huliyan “Implementation of Smart-Contract, Based on Hyperledger Fabric Blockchain” 2020 21st International Symposium on Electrical Apparatus & Technologies (SIELA)

[9]Veneta Aleksieva; Hristo Valchanov; Anton Huliyan “Implementation of Smart Contracts based on Hyperledger Fabric Blockchain for the Purpose of Insurance Services”2020 International Conference on Biomedical Innovations and Applications (BIA)

[10]Guo Baicang; Jin Lisheng; Shi Jian; Zhang Shunran “A risky prediction model of driving behaviors: especially for cognitive distracted driving behaviors” 2020 4th CAA International Conference on Vehicular Control and Intelligence (CVCI)

[11]Jingren Chen; Yefu Wu; Haojun Huang; Bing Wu; Guolin Hou “Driving-Data-Driven Platform of Driving Behavior Spectrum for Vehicle Networks”2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)

WEBSITES

[1]https://101blockchains.com/introduction-to-blockchain-features/#

[2]https://www.quartix.com/blog/vehicle-tracking-driver-scores/

[3]https://owners.hyundaiusa.com/us/en/resources/blue-link/what-is-a-hyundai-driving-score.html

[4]https://github.com/AkashKabra11/Driver-Behavior-Scoring

THANK YOU