# 

PAYD

PC001

Detail Level Design (SSAD)

**Record of Release**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Version No. | Modified By | Reviewed By | Authorized By | Release Date | Modifications Done |
| 0.1 | Shaurya Verma | Khushi Sidana | Mann Patel | 10 Sep 2020 | Database Design, State Transition Diagram, Module Specifications |
| 0.2 | Shaurya Verma | Khushi Sidana | Mann Patel | 11 Sep 2020 | Pseudo Code, User Interface Design |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Table of Contents

1.0 Database Design / Class Design 3

2.0 Processes / State Transition Diagrams 3

3.0 User Interface Design 3

4.0 Module Specifications 3

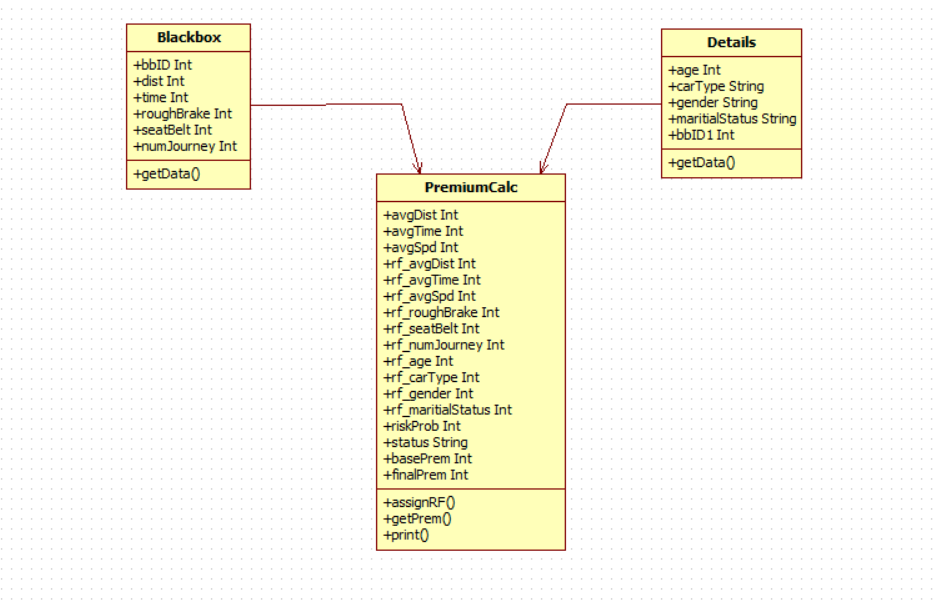
4.1 Module Overview 3

4.2 Function Specifications 3

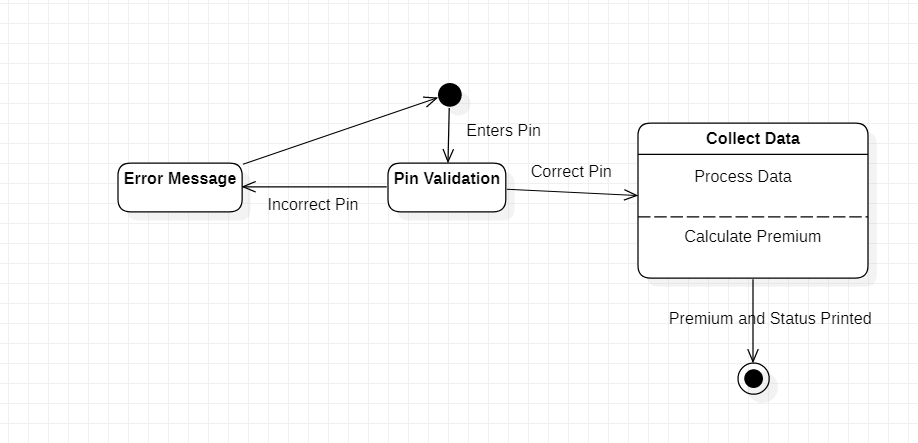
4.3 External Components Used 3

# Database Design / Class Design

|  |  |  |  |
| --- | --- | --- | --- |
| **Login** | | | |
| **Field** | **Data Type** | **Description** | |
| UID | Unsigned long long integer  length = 16 | A Unique Identification number given to each end-user at the time of initial registration with the insurance company  **Primary Key** | |
| PIN | unsigned integer  length = 4 | Takes input of PIN for each end-user identified by UID given to him/her | |
| **Details** | | | |
| **Field** | **Data Type** | **Description** | |
| UID | unsigned long long integer  length = 16 | Unique Identification number given to each customer.  **Foreign Key** | |
| firstName | String  Max. length = 20 | First Name of customer | |
| middleName | String  Max. length = 20 | Middle Name of customer | |
| lastName | String  Max. length = 20 | Last Name of customer | |
| password | String  Min. length = 8  Max. length = 20 | Password set by user to login to insurance company portal/website. | |
| gender | character | ‘M' for Male and 'F' for Female | |
| maritalStatus | character | ‘S' for Single and 'M' for Married | |
| address | String  Max. length = 100 | Permanent Address of the customer | |
| contact1 | unsigned long long integer | Mobile Number of the customer for contact | |
| contact2 | unsigned long long integer | Mobile Number of the customer for contact | |
| aadharNum | unsigned long long integer  length = 12 | Aadhar Car number of the customer for government verification  **Primary Key** | |
| bbID | unsigned long long integer  length = 20 | Unique Blackbox ID to be linked with UID  **Foreign Key** | |
| planID | unsigned integer | ID number of Premium Plan Opted by customer | |
| finalPrem | unsigned float | Total Premium to be paid by the customer | |
| **Blackbox** | | | |
| **Field** | **Data Type** | **Description** | **Pseudo Code** (if applicable) |
| UID | Unsigned long long integer  length = 16 | Used to identify end-user  **Foreign Key** |  |
| bbID | unsigned long long integer  length = 20 | Used to identify each Blackbox and link it with owner, i.e., link to only one UID  **Primary Key** |  |
| totalTrips | unsigned integer | Stores Total number of Trips in Blackbox | Increments by 1 at the end of each trip. |
| totalDist | unsigned float | Stores Total Distance travelled in all trips in Blackbox | Adds dist to totalDist at the end of each trip |
| avgDist | unsigned float | Stores Average Distance of a trip in Blackbox. | Calculated by computing (totalDist / totalTrips) |
| totalTime | unsigned float | Stores Total Time duration of all trips in Blackbox | Adds time to totalTime at the end of each trip |
| avgTime | unsigned float | Stores Average Time duration of a trip in Blackbox | Calculated by computing (totalTime / totalTrips) |
| avgSpeed | unsigned float | Stores Average Speed in all trips in Blackbox | Calculated by computing (totalDist / totalTime) |
| roughBrake | unsigned integer | Stores number of Rough Brakes applied in Last Trip in Blackbox |  |
| totalRB | unsigned integer | Stores Total number of Rough Brakes applied in all trips | Adds roughBrake to totalRB at the end of each trip |
| avgRB | unsigned float | Stores Average Rough Brakes applied in a trip | Calculated by computing (roughBrake/totalTrips) |
| seatBelt | unsigned integer | Stores Seat Belts worn in last trip |  |
| totalSB | unsigned integer | Stores number of times Seat Belts worn in all trips in Blackbox | Adds seatBelt to totalSB at the end of each trip |
| avgSB | unsigned float | Stores Average Number of times Seat Belts worn in all trips in Blackbox | Calculated by computing (totalSB/totalTrips) |
| **BbDataRiskFactor** | | | |
| **Field** | **Data Type** | **Description** | |
| UID | unsigned long long integer  length = 16 | Used to identify end-user  **Foreign Key** | |
| bbID | Unsigned long long integer  length = 20 | Used to identify each Blackbox and link it with owner, i.e., link to only one UID  **Primary Key** | |
| rf\_avgDist | unsigned integer | Assign Risk Factor according to measured Data | |
| rf\_avgTime | unsigned integer | Assign Risk Factor according to measured Data | |
| rf\_avgSpeed | unsigned integer | Assign Risk Factor according to measured Data | |
| rf\_avgRB | unsigned integer | Assign Risk Factor according to measured Data | |
| rf\_avgSB | unsigned integer | Assign Risk Factor according to measured Data | |
| rf\_totalTrips | Unsigned integer | Assign Risk Factor according to measured Data | |
| **DetailsRiskFactor** | | | |
| **Field** | **Data Type** | **Description** | |
| UID | unsigned long long integer  length = 16 | Used to identify end-user  **Primary Key** | |
| rf\_age | unsigned integer | Assign Risk Factor according to measured Data | |
| rf\_vehicleType | unsigned integer | Assign Risk Factor according to measured Data | |
| rf\_gender | unsigned integer | Assign Risk Factor according to measured Data | |
| rf\_maritalStatus | unsigned integer | Assign Risk Factor according to measured Data | |
| **PremiumCalc** | | | |
| **Field** | **Data Type** | **Description** | **Pseudo Code** (if applicable) |
| UID | unsigned long long integer  length = 16 | Used to identify end-user  **Primary Key** |  |
| riskProb | float | Calculates and Store the Probability of Accident | Sum of all ‘rf\_’ variables divided by 50 |
| status | String  Max. length = 10 | Holds value “Safe”, “Moderate”, “Risky” Based on value of riskProb |  |
| basePrem | unsigned float | Holds the Base Value of Premium which is the minimum Premium to be paid minus 10% |  |
| finalPrem | unsigned float | Calculates and Store the Premium to be paid by End-User by computing | finalPrem = basePrem + (basePrem \* riskProb) |



# Processes / State Transition Diagrams



# User Interface Design

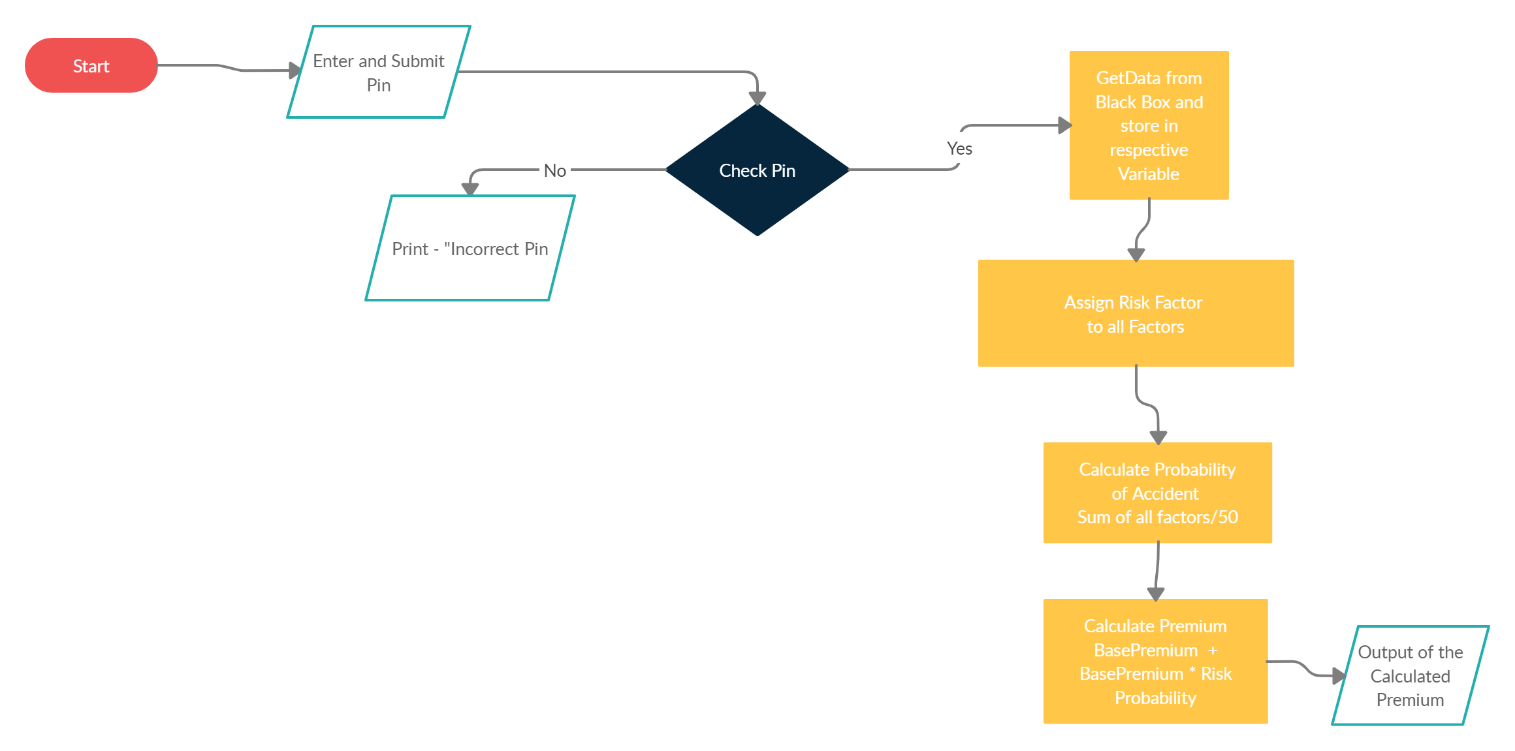


Navigation:

* PIN – Prompts user to enter 4-digit PIN code to verify the user before starting data upload.
* “Submit” button – Initiates pinCheck() function.
* “Insurances” button – redirects to available insurance plans webpage.
* “Claims” button – redirects to previous claims made webpage.
* “Renewals” button – redirect to renew insurance policy webpage.
* “Your Details” – redirects to user details webpage.
* “About Us” button – redirects to about the insurance company webpage.

# Module Specifications

* 1. **Module Overview**



This portal is used to upload end-user’s Blackbox data to database server. The files ready to be uploaded are checked for the Blackbox ID stored in it. The Blackbox ID is looked up against the associated UID and if match is found, end-user can upload files to server.

The files are uploaded to SFTP server where they are decrypted and scanned for tampering and malicious program. After clearance, they are uploaded to database server and backup server and processed to calculate premium. After premium calculation, data is backed up on backup server and checked for data loss.

### Logic of Module

* User must connect Blackbox to the system, login to the portal and click on “Upload Data”. This will initiate function pinCheck() and the user will be prompted to enter UID and PIN to unlock Blackbox. The function will then look up UID and PIN. If verification fails, user will be prompted to re-enter the details.
* Successful verification will initiate function getData() to fetch data from Blackbox and upload it to server.
* The function looks up Blackbox ID (bbID) against attached UID. If the bbID matches with UID of logged-in user, then data uploading starts. Otherwise, it displays an error “This Blackbox is not associated with your account”.
* Data is uploaded to SFTP server, decrypted and scanned for tampering and malicious program. After clearance, it is uploaded to both database server and backup server and an upload log is added in Blackbox. If the data is tampered with or contains malicious program, a ticket is raised regarding the respective issue which includes details of logged-in user’s details, i.e., UID, bbID, issue type and details. Such file is immediately deleted from the server to prevent data breach and server/website crash.
* After data uploading, assignRF() is executed which calculates risk factors and uploads the values in database.
* On execution of assignRF(), premium is calculated using getPrem()
* getPrem() works by calculating risk probability by adding all risk factors and dividing the sum by 50
  + Further premium is calculated   
    Premium = BasePremium + (Base Premium \* Risk Probability)   
    Here, Base Premium = Monthly Minimum Premium – 20% of Monthly Minimum Premium  
    For example, if minimum Premium is 10,000 then Base Premium is 8,000
  + (Minimum Risk Probability is 0.2 in ideal situation and thus Monthly Minimum Premium will be the Premium for a safe driver)
* Output of Premium is Printed through premPrint() function.
* The updated data is saved in the database of database server and backup server.

## Function Specifications

**Function 1 – pinCheck()**

* **Logic** – pinCheck() is initiated on clicking “Upload Data” button on portal after connecting Blackbox to the system. The function prompts use to enter UID and PIN to upload data. If the PIN matches with UID of logged-in user, getData() function is initiated. If PIN don’t match with UID, an error message is displayed and user is prompted to re-enter the details.
* **INPUT** – UID and PIN.
* **OUTPUT** – error message if PIN does not match with UID and user is prompted to re-enter details.
* **External Source Components** – None
* **Tables updated** – None

**Function 2 –** **getData()**

* **Logic** – getData()is initiated after successful execution of pinCheck() function. getData() function looks up Blackbox ID (bbID) against UID of logged-in user. If the bbID doesn’t matches with UID, function displays an error “This Blackbox is not associated with your account”. If the bbID matches with UID, data starts uploading to SFTP server, where it is decrypted and checked for tampering and malicious programs.   
  On clearance from server, data is uploaded to backup and database server and an upload log is added in the Blackbox. The Blackbox data is then cleared for next round of data collection.
* **INPUT/OUPUT** – None
* **External Source Components** – Blackbox
* **Tables updated** – Blackbox.

**Function 3 - assignRF()**

* **Logic** - assignRF() assigns the risk factor values ranging from 1-5 according to the received data based on conditions mentioned in the following tables. Each risk as a unique factor based on the received data.

|  |  |
| --- | --- |
| **avgDist** | |
| **Average Distance ( kms )** | **Risk Factor** |
| 0-20 | 1 |
| 20-50 | 2 |
| 50-100 | 3 |
| 100-200 | 4 |
| 200+ | 5 |

|  |  |
| --- | --- |
| **avgTime** | |
| **Average Time ( hrs )** | **Risk Factor** |
| 0-0.5 | 1 |
| 0.5 - 1 | 2 |
| 1-1.30 | 3 |
| 1.30-3 | 4 |
| 3+ | 5 |

|  |  |
| --- | --- |
| **avgSpeed** | |
| **Average Speed ( km/hr )** | **Risk Factor** |
| 0-20 | 1 |
| 20-40 | 2 |
| 40-70 | 3 |
| 70-100 | 4 |
| 100+ | 5 |

|  |  |
| --- | --- |
| **avgRB** | |
| **Average Rough Brake** | **Risk Factor** |
| 0-0.1 | 1 |
| 0.1-0.2 | 2 |
| 0.2-0.3 | 3 |
| 0.3-0.5 | 4 |
| 0.5+ | 5 |

|  |  |
| --- | --- |
| **avgSB** | |
| **Average Seat Belt** | **Risk Factor** |
| 0.90-1 | 1 |
| 0.8-0.9 | 2 |
| 0.6-0.8 | 3 |
| 0.5-0.6 | 4 |
| <0.5 | 5 |

|  |  |
| --- | --- |
| **totalDist** | |
| **Number of Journey** | **Risk Factor** |
| 0-10 | 1 |
| 10-30 | 2 |
| 30-50 | 3 |
| 50-100 | 4 |
| 100+ | 5 |

|  |  |
| --- | --- |
| **age** | |
| **Age (years)** | **Risk Factor** |
| 30-45 | 1 |
| 27-30 | 2 |
| 23-27 | 3 |
| 20-23 || 45-65 | 4 |
| 18-20 || 65+ | 5 |

|  |  |
| --- | --- |
| **vehicleType** | |
| **Vehicle Type** | **Risk Factor** |
| Hatch Back | 1 |
| Sedan | 2 |
| Compact SUV | 3 |
| SUV | 4 |
| Luxury / 2-Wheeler | 5 |

|  |  |
| --- | --- |
| **gender** | |
| **Gender** | **Risk Factor** |
| Male | 1 |
| Female | 5 |

|  |  |
| --- | --- |
| **Marital Status** | |
| **Status** | **Risk Factor** |
| Married | 1 |
| Single | 5 |

* **INPUT / OUTPUT** – None
* **External Component used** – None
* **Tables updated –** BbDataRiskFactor, DetailsRiskFactor

**Function 4 – getPrem()**

* **Logic** – getPrem() first calculates Risk Probability by adding all the ‘rf’ denoted variables, which are risk factors, and the sum is divided by 50 which is maximum possible sum.   
    
  Therefore, Risk Probability = Sum of Risk Factors / 50.  
    
  Once Risk Probability is calculated, Final Premium can be calculated.  
  Final Premium = Base Premium + (Base Premium \* Risk Factor)
* **INPUT / OUTPUT** – None
* **External Component used** – None
* **Tables updated –** Details, PremiumCalc

**Function 5 – premPrint()**

* **Logic** – Displays the calculated premium.

|  |  |
| --- | --- |
| **status** | |
| **Risk Probability** | **Performance** |
| 0-0.3 | Safe |
| 0.3-0.6 | Moderate |
| 0.6+ | Risky |

* **INPUT** – None
* **OUTPUT** – Calculated premium and driving status.
* **External Component used –** None
* **Tables updated –** PremiumCalc

## External Components Used

**Blackbox** is used by Module 5 to fetch driving data of the end-user to use it to calculate premium by analysing the driving behaviour.

**SFTP Server** is used to temporarily store data for checking for data tampering and malicious program in the files. If cleared, files are then uploaded to database and backup server.