Low Level Design

Predict Bank Credit Risk

Document Control

Change Record

|  |  |  |  |
| --- | --- | --- | --- |
| Date Issue | Version | Description | Author |
| 12/08/2021 | 1 | Initial LLD – V 1.0 | Raghav Bakshi |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Approval Status

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Review Date | Review by | Approved by | Comments |
|  |  |  |  |  |

Content

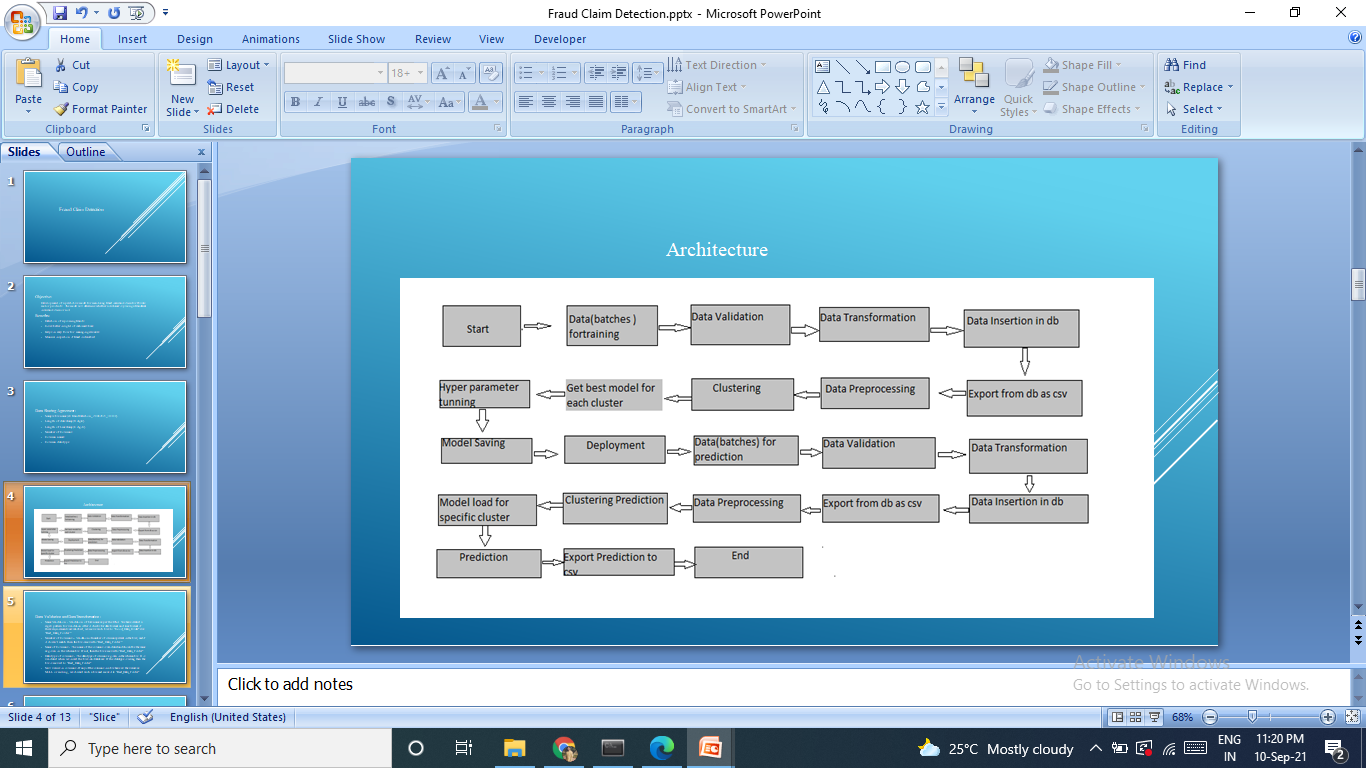
1. Introduction ………………………………………………………………………………………………………………
   1. What is Low Level Design Document …………………………………………………………………..
   2. Scope ………………………………………………………………………………………………………………….
2. Architecture ……………………………………………………………………………………………………………..
3. Architecture Description …………………………………………………………………………………………..
   1. Data Description ……………………………………………………………………
   2. Data Transformation …………………………………………………………………………….
   3. Data Pre-Processing ………………………………………………………………………………….
   4. Feature engineering ………………………………………………………………………………………………
   5. Feature Selection ………………………………………………………………………………………………….
   6. Testing for Classification algorithm …………………………………………………………………….
   7. Selecting model with best accuracy …………………………………………………………………….
   8. Model training …………………………………………………………………………………………………….
   9. Model deployment ……………………………………………………………………………………………..
4. Unit Test Cases ………………………………………………………………………………………………………
5. Introduction
   1. What is Low Level Design Document?

The goal of LLD or Low Level Design Document (LLDD) is to give the internal logic design of the actual program code for Predict Bank Credit Risk. LLD describes the class diagram with the methods and relations between classes and program specs. It describes the modules so that programmer can directly code from the document.

* 1. Scope

Low Level Design (LLD) is a component level design process that follows a step by step refinement process. This process can be used to design data structure, required software architecture, source code and ultimately performance algorithm. Overall, the data organization may be defined during requirement analysis and then refined during data design work.

1. Architecture



1. Architecture Description
   1. Data description

The dataset contains 21 unique columns and 1000 rows. Dataset is collected from UCI Machine Learning Repository.

* 1. Data Transformation

In the Transformation Process, we will convert our original dataset which is in .asc format to .csv Format

* 1. Data Preprocessing

In this we will deal with missing values, outliers and transform the data in normal distribution.

* 1. Feature Engineering

Insights will be drawn from the data using domain knowledge and if required, columns will be created.

* 1. Feature Selection

Only the important columns which are contributing significantly will be kept, rest will be dropped.

* 1. Testing for Classification algorithm

All classification algorithms will be run on the dataset and the one with best accuracy will be selected.

* 1. Selecting model with best accuracy

The algorithm with the best will be chosen and then hyper parameter tuning will be done.

* 1. Model training

After hyper parameter tuning, model training will be done.

* 1. Deployment

The whole solution created above will be pushed to a cloud platform for user to interact with it.

1. Unit Test Case

|  |  |  |
| --- | --- | --- |
| Test Case Description | Pre - requisite | Expected result |
| Verify whether application URL is accessible to the user | 1.Application URL should be defined | Application URL should be accessible to the users |
| Verify whether the application loads successfully when the URL is hit | 1. Application URL is accessible  2. Application is deployed | The application loads successfully when the URL is hit |
| Verify whether user is able to see input fields | 1. Application is  accessible | User should be able to see input fields |
| Verify whether user is able to edit all input fields | 1. Application is  accessible | User should be able to edit all input fields |
| Verify whether user gets Submit  button to submit the inputs | 1. Application is  accessible | User should get Submit button to  submit the inputs |
| Verify whether user is presented with recommended results on clicking  submit | 1. Application is  accessible | User should be presented with  recommended results on clicking  submit |
| Verify whether the recommended  results are in accordance to the  selections user made | 1. Application is  accessible | The recommended results should  be in accordance to the selections  user made |