Low Level Design

CUSTOMER EXPERIENCE PREDICTION

|  |  |
| --- | --- |
| Written By | Shreyash Virendra Chawda |
| Document Version | ***0.1*** |
| Last Revised Date | 28-06-2023 |

**Document Control**

### Change Record:

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Comments** |
| 0.1 | 28-06-2023 | Shreyash Virendra Chawda | Architecture, building, Deployment |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

### Reviews:

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Reviewer** | **Comments** |
|  |  |  |  |

### Approval Status:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Review**  **Date** | **Reviewed By** | **Approved By** | **Comments** |
|  |  |  |  |  |

Contents

1. [Introduction 1](#_bookmark0)
   1. [What is Low-Level design document? 1](#_bookmark1)
   2. [Scope 1](#_bookmark2)
2. [Architecture 2](#_bookmark3)
3. [Architecture Description 3](#_bookmark4)

[Data Description 3](#_bookmark5)

[Data Transformation 3](#_bookmark7)

[Data Pre-processing 3](#_bookmark10)

[Model Building 4](#_bookmark12)

[Data from User 4](#_bookmark13)

[Data Validation 4](#_bookmark14)

[Deployment 4](#_bookmark19)

1. [Unit Test Cases 5](#_bookmark20)

# Introduction

## What is Low-Level design document?

The goal of LLD or a low-level design document (LLDD) is to give the internal logical design of the actual program code for CUSTOMER EXPERIENCE . LLD describes the class diagrams with the methods and relations between classes and program specs. It describes the modules so that the programmer can directly code the program from the document.

## Scope

Low-level design (LLD) is a component-level design process that follows a step-by-

Step [refinement](https://en.wikipedia.org/wiki/Refinement_(computing)) process. This process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then refined during data design work

# Architecture

Data Ingestion

Data Transformation

## 

Model Testing

Model Building

Preprocessing

Deployment

# Architecture Description

## Data Description

Dataset contains csv file with 7043 rows and 21 columns in which 20 independent features and one dependent features which is target variable.

## Data Transformation

In the Transformation Process, we will read csv file and drop unwanted independent features and separate input variables and out variable.

## Data Pre-processing

Data Pre-processing steps we could use are Null value handling, Remove duplicates, converting categorical data into numerical data. Creating scikit-learn column transformation pipeline to handle the flow of data.

## Model Building

Splitting the data into train and test data. With help of different classifier algorithm training the data to find which model will give good accuracy and then use best model for training.

## Data from Test

Here we will use test data to predict how well our model is executing with comparing the output of test data and predicted data which will give us accuracy of the model.

API

Using Flask module we had created API which can be used to insert the customer data to predict status of the customer credit card payment.

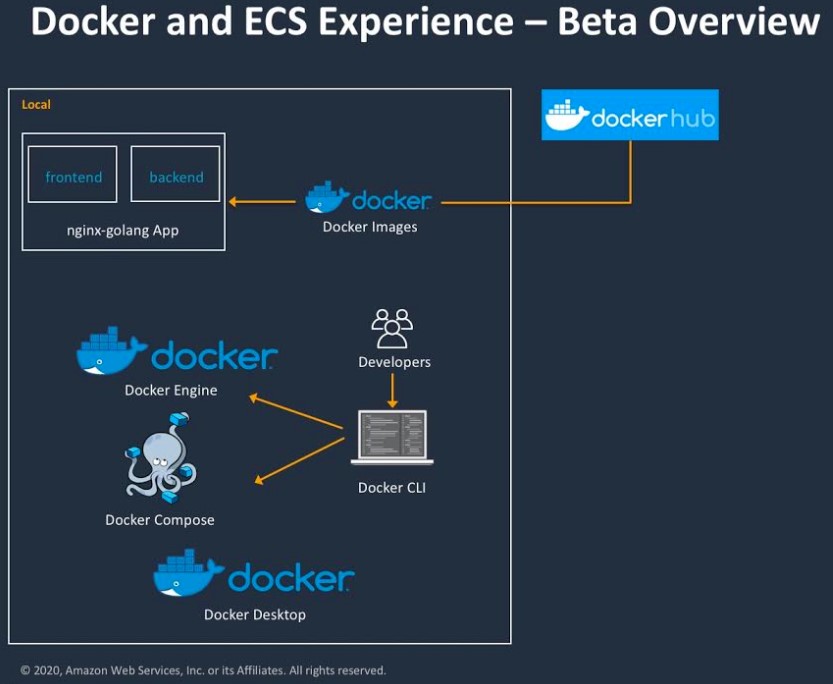
## Data Validation

Here Data Validation will be done, given by the source.

## Deployment

## 

## We will be deploying the model to AWS. Also Docker is used for deployment.



# Unit Test Cases

|  |  |  |
| --- | --- | --- |
| **Test Case Description** | **Pre-Requisite** | **Expected Result** |
| Verify whether the Application URL is  accessible to the user | Application URL  should be defined | Application URL should be  accessible to the user |
| Verify whether the Application loads completely for the user when the URL is accessed | Application URL is accessible | The Application should load completely for the user when the URL is accessed |
| Verify whether user is able to see input fields. | Application is accessible | User should be able to see input fields on logging in |
| Verify whether user is able to edit all input fields | Application is accessible | User should be able to edit all input fields |
| Verify whether user gets Submit button to submit the inputs | Application is accessible | User should get Submit button to submit the inputs |
| Verify whether user is presented with recommended results on clicking  submit | 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 | Application is accessible | The recommended results should be in accordance to the selections user made |