

**Bank Bot**

LOW LEVEL DOCUMENT DESIGN

01.12.2021

Project By:

Hrishik V

# Document Version Control

|  |  |  |  |
| --- | --- | --- | --- |
| DATE ISSUED | VERSION | DESCRIPTION | AUTHOR |
| December 1, 2021 | 1 | Initial LLD V1 | Hrishik V |
|  |  |  |  |
|  |  |  |  |

## 1. Introduction

**1.1** Why **this Low-Level Document?**

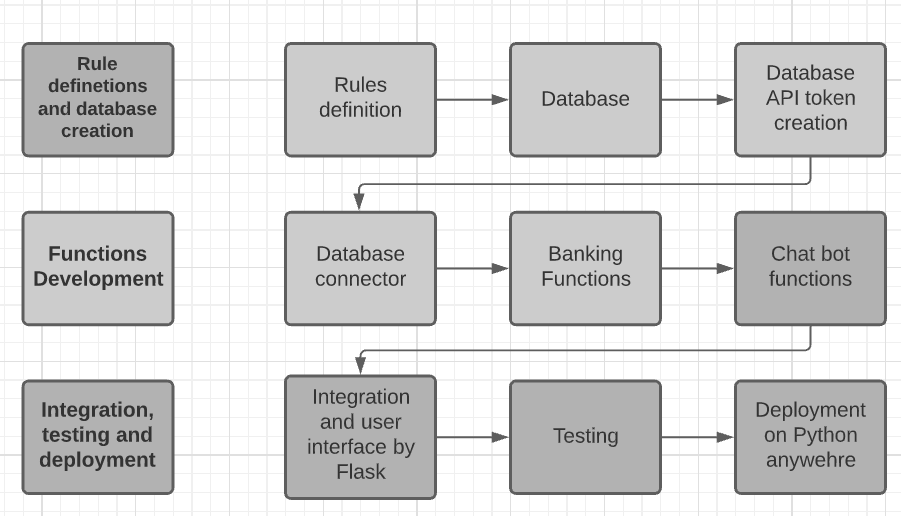
The goal of LLD or a low-level design document is to give the internal logical design of the actual program code for Adult Census Income Prediction. LLD describes the class diagrams with the methods and relations between classes and program specs. It describes the modules so that the programmer can directly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

### 1.2 Scope

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

**2. Architecture**

Proposed Methodology



## 3. Architecture Description

### 3.1 Rule definitions and database creation

The rules are created with Watson bank assistant by IBM as the reference. The rules are stored in json format for easy access as they can be parsed into dictionaries. The database is created randomly using pythons and stored in csv format which is later uploaded to Astra datastax which is a cql based web hosting service for datasets.

### 3.2 Functions development

This step includes development of functions for the individual sections of the chat bot such as connect to the dataset, dataset handling functions and the basic chat bot functions.

### 3.3 Integration testing and deployment:

This step contains all other necessary steps as well as integrating the results of all the other steps getting the final product and testing to make sure there are no errors. The integration is done and the front end is designed and developed using flask. The app is deployed locally during the testing phase later on it can be deployed on python anywhere, Heroku or similar services.

## 4. Unit Test Cases

|  |  |  |
| --- | --- | --- |
| **Test Case Description** | **Pre-Requisite** | **Expected Result** |
| Verify whether the  Database is functional or not | Database must be defined and uploaded | Database must be accessible via the API |
| Verify whether the Basic banking functions are functional or not | 1. Database must be functional 2. Basic Banking Functions must be defined | The functions must affect the database and return the expected output |
| Verify whether Chatbot functions are functional | 1. Rules must be defined 2. Chat bot functions must be defined | The functions must return expected outputs |
| Verify whether the integration is successful or not | 1. Banking and chatbot functions must be functional 2. Integration must be done | The integration should not produce any ru8ntime errors and should provide require outcome. |