**Design Documentation for Chat Bot**

**Contents Page Number**

**1 Architecture Diagram Description 1-4**

1.1 Front End User Interface (Application Layer) 2

1.2 API and Framework Layer 2-3

1.3 Python 3

1.4 Libraries 3

1.5 Storage Layer 4

1.6 Dialogflow.

**2 Flow Diagram Description 5-6**

2.1 Scenario 1: No Integration 5

2.2 Scenario 2: Integration Required 5

2.3 Scenario 3: Answering Unknown Question 6

2.4 Scenario 4: Answering Unknown Question Already Answered 7

**Architecture Diagram Description**

1. **Front End User Interface (Application Layer):**

The Application Layer or the User Interface layer consists of HTML, CSS, JS which are used in the creation of the chat windows of Human Agent and the User. Whenever the user enters some text into the chat window it is then sent to python code to identify the intent of the question to provide proper response back to the user in the chat window. The human agent chat window displays the entire conversation between the user and the bot which is also used to provide the answer if the intent of the question is not identified.

* 1. **HTML:**

HTML or (Hypertext Markup Language) is used for the creation of a webpage which is used for the creation of the front end chat windows through which the chat application accepts and displays the texts

* 1. **CSS:**

CSS or (Cascading Style Sheet) is a document which accompanies HTML which is responsible for adding designs and formatting of the HTML page so that the contents of the page are more presentable and customizable.

* 1. **JS:**

JS or (Java Script) as the name suggests is a script which is added along with the HTML page for the addition of new functionalities and to make the page interactive such as drop downs, menus etc.

1. **API and Framework Layer:**

API or (application programming interface) is a set of subroutine definitions, protocols, and tools for building application software. In general terms, it is a set of clearly defined methods of communication between various software components. The API layer consists of various APIs which helps python to push contents into the front end webpage. The API and framework layer consists of the REST API, FLASK, Memcached.

* 1. **FLASK:**

Flask is a micro web framework written in Python and based on the Werkzeug toolkit and Jinja2 template engine. Flask is responsible for the creation of the REST API.

* 1. **REST API:**

Rest API or (REpresentational State Transfer) or RESTful web services provide interoperability between computer systems on the Internet. REST-compliant web services allow the requesting systems to access and manipulate textual representations of web resources by using a uniform and predefined set of stateless operations. Other kinds of web services, such as WSDL and SOAP, expose their own arbitrary sets of operations. The Rest API interacts with the Java Script and initiates an array of array of request-response operations which assists in pushing the text from python to the chat window.

* 1. **Memcached:**

Memcached is a general-purpose distributed memory caching system. Memcached is used to store the text from the user, human agent, bot which is either used to display the text the human agent writes in the chat window into the user chat window and vise versa or to save the entire conversation into the database from which the machine learning algorithm is run so that any unknown question asked for the second time can be answered for the second time.

1. **Python Layer:**

Python is a interpreted high level programing language for general purpose programming. Python obtains the conversation from the user communicates with the dialog flow either to get the response or to identify the intent so that it can obtain the response from the integration section. Python is also responsible for maintaining the CSV files which consists of the user, policy, claim creation data and training dataset and test dataset used for running the Machine Learning Algorithm. The machine learning algorithm is also written in python which runs the algorithm.

1. **Libraries:**

A library is a collection of non-volatile resources used by computer programs, often to develop software. These may include configuration data, documentation, help data, message templates, pre-written code and subroutines, classes, values or type specifications.

Various libraries such as the CSV library, flask library, Memcached libraries have been imported into the python code so that various resources related to them can be automatically imported into the code every-time the code is run.

1. **Storage Layer:**

The storage layer consists of databases in which various details which are required by the application. Here the use of CSV files has been made to replicate the functionalities of the database. Python is responsible to read and write information from and into the csv file. The storage layer stores the user data, training and test dataset which is used by python.

1. **Dialogflow:**

Dialogflow is a Google-owned developer of human–computer interaction technologies based on natural language conversations. Dialogflow is used to provide the user input to determine the intent of the user input. The intent is recognized by the text classification algorithm google uses to train the bot. The bot is created in dialog flow specifying the intents along with the training sentences and entities.

**Flow Diagram Description**

**Scenario 1: No Integration.**

**Input: User Query**

**Output: Response from Dialogflow.**

**Sample Flow: Chat Window 🡪 JavaScript (User) 🡪 Flask (User) 🡪 Dialogflow API 🡪 Speech 🡪 Flask 🡪 JavaScript 🡪 Chat Window**

**Description:** Once the user has provided the input in the chat window, the query is sent to the JavaScript which then sends the query to flask via the REST API then the query hits the Dialogflow API to match for intent and obtain the respective response or speech. The Dialogflow API sends the speech to flask which then sends it to the JavaScript which appends the speech to the chat window.

**Scenario 2: Integration Required.**

**Input: User Query**

**Output: Response from Python.**

**Sample Flow: Chat Window 🡪 JavaScript (User) 🡪 Flask (User) 🡪 Dialogflow API 🡪 Parameters 🡪 Read/Write CSV 🡪 Speech 🡪 Flask 🡪 JavaScript 🡪 Chat Window**

**Description:** Once the user has provided the input in the chat window, the query is sent to the JavaScript which then sends the query to flask via the REST API then the query hits the Dialogflow API to match for intent once the intent is obtained the Dialogflow sends the parameters to python using webhook which then either performs a task such as create a claim and return the claim number or looks up the user data from the csv file by performing read/write operations and returns the appropriate output. The returned output is saved in speech and the python sends the speech to flask which then sends it to the JavaScript which appends the speech to the chat window.

**Scenario 3: Answering Unknown Question.**

**Input: User Query Unknown to Bot**

**Output: Response from Human Agent.**

**Sample Flow:**

**1 Chat Window 🡪 JavaScript (User) 🡪 Flask (User) 🡪 Dialogflow API 🡪 Unknown Question not answered 🡪 memcached 🡪 Flask(Agent) 🡪 JavaScript(Agent) 🡪 Agent Input 🡪 Memcached 🡪 Flask(Agent) 🡪 JavaScript(User) 🡪 Chat Window**

**2 Memcached 🡪 CSV(Write)**

**Description:** Once the user has provided the input in the chat window, the query is sent to the JavaScript which then sends the query to flask via the REST API then the query hits the Dialogflow API to match for intent once the intent cant be determined it goes to the fallback stage the Dialogflow uses webhook sends the parameters to python. The python initiates the memcached hits the flask of the human agent which then hits the REST API of human agent which then goes to the JavaScript of the human agent. When the human agent provides with the response which is then sent to the flask of human agent which then hits memcached which then sends the speech or response to the JavaScript of the user which then appends the speech to the user chat window.

**Scenario 4: Answering Unknown Question already answered.**

**Input: User Query Unknown to Bot**

**Output: Response from Python.**

**Sample Flow:**

**1 Chat Window 🡪 JavaScript (User) 🡪 Flask (User) 🡪 Dialogflow API 🡪 Unknown Question already answered 🡪 Read CSV 🡪 Flask 🡪 JavaScript(User) 🡪 Chat Window**

**Description:** Once the user has provided the input in the chat window, the query is sent to the JavaScript which then sends the query to flask via the REST API then the query hits the Dialogflow API to match for intent once the intent can’t be determined it goes to the fallback stage the Dialogflow uses webhook sends the parameters to python. Python reads the CSV file to check if the question has already been answered. Then python finds the answer to the requested question in the CSV file. The answer is returned and saved inside speech. The speech is then sent to flask which then sends it to the JavaScript which appends the speech to the user chat window