UMGC City Team 2

**UMGC-Chatbot**

Software Design Document

Name (s):

Joseph Ogunsanya

Helen Abraham

Evelyn Akinosun

Essaid Elboukhani

Jan (Jon) Masouh

Romuald Priva Leuga Nousibouaba

Vincent Seen

Date: 04/05/2020

# Table of Contents

Contents

[Table of Contents 1](#_Toc37016537)

[1. INTRODUCTION 1](#_Toc37016538)

[1.1 Purpose 1](#_Toc37016539)

[1.2 Scope 1](#_Toc37016540)

[1.3 Definitions and Acronyms 2](#_Toc37016541)

[2 REFERENCES 3](#_Toc37016542)

[3 DECOMPOSITION DESCRIPTION 3](#_Toc37016543)

[3.1 Module Decomposition 3](#_Toc37016544)

[3.2 Concurrent Decomposition 4](#_Toc37016545)

[3.2.1 Natural Language Process Description 5](#_Toc37016546)

[3.2.2 Dialog Management process Description 5](#_Toc37016547)

[3.3 Data Decomposition 6](#_Toc37016548)

[3.3.1 Session ID Description 6](#_Toc37016549)

[3.3.2 Date Description 6](#_Toc37016550)

[4 DEPENDENCY DESCRIPTION 7](#_Toc37016551)

[4.1 Intermodule Dependencies 7](#_Toc37016552)

[4.2 Interprocess Dependencies 8](#_Toc37016553)

[4.3 Data Dependencies 8](#_Toc37016554)

[5 Interface Description 10](#_Toc37016555)

[5.1 Module Interface 11](#_Toc37016556)

[5.2.1 Home Screen 12](#_Toc37016557)

[5.2.2 Screen with Chat Activated 12](#_Toc37016558)

[5.2.3 Screen with Chat Expanded 14](#_Toc37016559)

[5.3 Process Interface 14](#_Toc37016560)

[5.3.1 DialogFlow Engine Process Interface 14](#_Toc37016561)

[5.3.2 IBM Watson Engine Process Interface 15](#_Toc37016562)

[5.3.3 Screen Objects and Actions 17](#_Toc37016563)

[6 Detailed Design 18](#_Toc37016564)

[6.1 Module Detailed Design 18](#_Toc37016565)

[DialogFlow / IBM Watson Description 18](#_Toc37016566)

[Agent 19](#_Toc37016567)

[Intents 19](#_Toc37016568)

[Deployment 20](#_Toc37016569)

[6.2 Data Detailed Design 20](#_Toc37016570)

[6.2.1 Data Description 21](#_Toc37016571)

[6.2.2 Data Dictionary 23](#_Toc37016572)

[7 REQUIREMENTS MATRIX 26](#_Toc37016573)

**List of Figures**

[Figure 1. UMGC Chatbot Architecture Design Modules Overview](#_heading=h.3rdcrjn) 4

[Figure 2. Chatbot architecture and processes](#_heading=h.2jxsxqh) 5

[Figure 3. Chatbot architectures 1](#_heading=h.3o7alnk)1

[Figure 4. Module Interface - City of Pasadena Home Page 12](#_heading=h.ihv636)

[Figure 5. UMCG Chatbot - Chat Activated 13](#_heading=h.1hmsyys)

[Figure 6. UMGC Chatbot - Chat Expended 14](#_heading=h.vx1227)

[Figure 7. DialogFlow Engine 15](#_heading=h.4f1mdlm)

[Figure 8. Watson Assistant 1](#_heading=h.4f1mdlm)6

[Figure 9. IBM Watson Assistant 1](#_heading=h.4f1mdlm)7

[Figure 10. Chatbot Platform Deployment Diagram](#_heading=h.4f1mdlm) 20

[Figure 11. UML Data Flow for Chatbot System](#_heading=h.4f1mdlm) 22

**List of Tables**

[Table 1. Definitions and Acronyms 2](#_heading=h.1t3h5sf)

[Table 2. Data Dictionary](#_heading=h.1mrcu09) 23

[Table 3. Requirements Matrix](#_heading=h.1mrcu09) 26

**Amendment History**

|  |  |  |
| --- | --- | --- |
| **Version Number** | **Issue Date** | **Changes** |
| 1.0 | 03/15/2020 | The initial version of the Chatbot SDD |
| 2.0 | 04/05/2020 | Implemented the instructor feedbacks |
|  |  |  |

# 

# 1. INTRODUCTION

The University of Maryland Global Campus (UMGC) has accepted a chatbot project by the customer for the City of Pasadena, California. UMGC City Team 2 will develop a user-friendly and robust chatbot system that will feature on the website of the City of Pasadena. The chatbot (referred to as UMGC Chatbot or Chatbot) design and implementation processes are described in this Software Design Document (SDD) by UMGC City Team 2.

## 1.1 Purpose

The purpose of this Software Design Document (SDD) is to assist both the developers, testers and the customer to visualize the Chatbot solution presented. This SDD describes the Chatbot system structure that satisfies the requirements identified in the SRS document. This document translates requirements into descriptive views and interfaces necessary for the implementation phase. The SDD is a detailed blueprint that the development team must follow during the implementation phase.

## 1.2 Scope

This document pertains to the base-level system of the Chatbot and will be used as a proof of concept to build the actual system. It focuses on the step-by-step process and the integration with the current system. The Chatbot is not a stand-alone application but a module that will integrate into an existing website. This document verifies how the system design will meet the requirements listed in the SRS document. This document addresses the design steps of the Chatbot and how it will integrate into the City of Pasadena website. The system interfaces with the external systems, DialogFlow and IBM Watson, that store and update intents and entities used by the Chatbot system. DialogFlow and IBM Watson will search for keywords and fetch response queries to the user. The Chatbot system is designed for applicable use on any other website.

## 1.3 Definitions and Acronyms

The following table provides definitions of terms, acronyms, and abbreviations that might exist and use in this SDD document.

|  |  |
| --- | --- |
|  | |
| **Term / Acronyms** | **Definition** |
| Chatbot | A computer program designed to simulate conversation with human users, especially over the Internet. |
| IBM | International Business Machines. A multinational technology corporation. |
| SDD | Software Design Document lists the technical and design details of the Chatbot application |
| SRS | Software Requirements Specifications. A document that describes the requirements of the Chatbot application |
| NLP | Natural Language Processing |
| JSON | JavaScript Object Notation |
| UMGC | University of Maryland Global Campus |
| User | Visitor or Customer on the City of Pasadena website. |

**Table 1. Definitions and Acronyms**

# 2 REFERENCES

(1998, December 04). IEEE recommended practice for software design descriptions. IEEE Std 1016-1998. The Institute of Electrical and Electronics Engineers, Inc. Retrieved March 30, 2020, from <https://www.cs.helsinki.fi/group/linja/resources/IEEE_Std_1016-1998.pdf>

Ogunsanya, J., Abraham, H. Akinlosotu, E., Elboukhani, E.Masouh, J., Nousibouaba, R.P.L., and Seen, V. Software requirements specification (SRS) – chatbot system. Retrieved March 28, 2020, from <https://drive.google.com/file/d/1XeBxxYVtJyiU2renaey9aExyVdIgp7yl/view?usp=sharing>

# 3 DECOMPOSITION DESCRIPTION

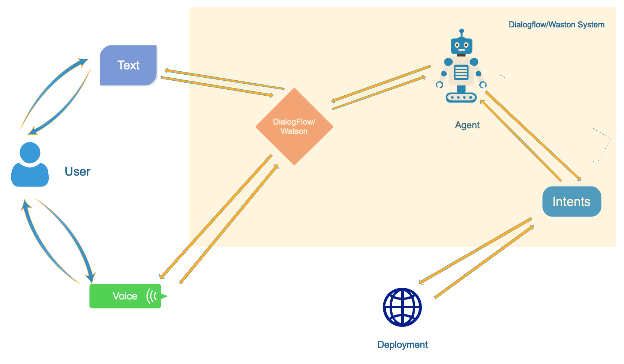
This section describes the decomposition of the system design in three different ways: Module decomposition, concurrent decomposition, and data decomposition. The functional description of the subsystems within the architectural design of the Chatbot system is illustrated in Figure 1.

## 3.1 Module Decomposition

In this section, the design is decomposed by grouping data and functionalities into cohesive modules. The module decomposition identifies each of the modules. The UMGC Chatbot system utilizes five main modules to facilitate its knowledge-base and drive the functionality of the system:

* Text/Voice
* DialogFlow/IBM Watson
* Agent
* Intents
* Deployment

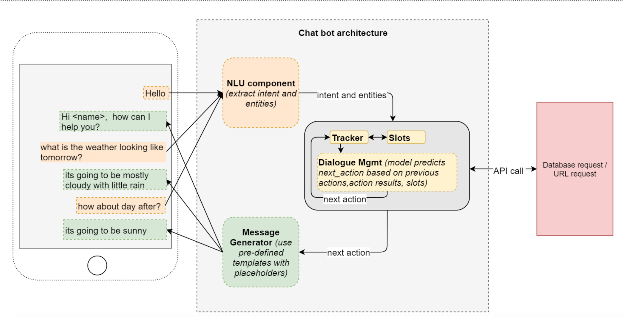
Figure 1 illustrates an overview of the high-level system architecture which is assembled from multiple modules.



**Figure 1. UMGC Chatbot Architecture Design Modules Overview**

## 3.2 Concurrent Decomposition

In this section, we look at the design from the concurrent processes and threads that run with the system. We identify each of these processes in the program design and outline the details of the processes just as for the modules in the previous section. The chat system follows the design in Figure 2:



**Figure 2- Chatbot architecture and processes  
Source:** <https://towardsdatascience.com/architecture-overview-of-a-conversational-ai-chat-bot-4ef3dfefd52e>

### 3.2.1 Natural Language Process Description

This process is in charge of analyzing the user inputs and requests and is responsible for extracting the intents and entities. It takes an unstructured message object from the user and transforms it into a structured data object. In general, the data object is a JSON object

### 3.2.2 Dialog Management process Description

The dialog management process kicks in next. Its role is to decide what answer to send to the user, given the user input and his past interactions. The process can predict answers based on previous actions or can pull responses from the database.

## 3.3 Data Decomposition

The current requirements do not provide a user login or sign up specification. Then, the system won’t have any personal user data inputs to write into our database. But, data such as session ID, records of interactions and date may be registered and saved. Other data entities in the design are fully described in section 6.2.

### 3.3.1 Session ID Description

|  |  |
| --- | --- |
| Session ID | This is a unique number that a web server assigns to a specific user for the time that the user is interacting with the Chatbot. The session ID can be pulled from the JSON response. |

### 3.3.2 Date Description

|  |  |
| --- | --- |
| Date | This is a date and time a user is interacting with the Chatbot. |

### 

# 4 DEPENDENCY DESCRIPTION

Architectural entities and components of the system are interdependent on each other including:

* Text / Voice to DialogFlow / IBM Watson
* Text / Voice to Intents
* Intents to DialogFlow / IBM Watson
* Agent to DialogFlow / IBM Watson
* Intents to Deployment

## 4.1 Intermodule Dependencies

The Chatbot system’s platforms DialogFlow and IBM Watson are dependent on the user’s Text and Voice inputs. The type of user inputs determines how the Chatbots will process the inputs to deploy the answers to the user’s question. The Intents garnered by the DialogFlow and IBM Watson platforms also depend on the user’s Text and Voice inputs. Depending on the user’s question, the system will attempt to understand the intent of the user’s question in order to deliver the appropriate response. These Intents consequently are based on pre-populated use cases and Intents programmed within DialogFlow and IBM Watson. If the Text or Voice input is unclear or does not match the intents programmed within DialogFlow and IBM Watson, the Chatbot will ask the user further questions to decipher the relative intents within the system. The DialogFlow and IBM Watson are dependent on the Agent component of the Chatbot system. The Agent is the module within the system that utilizes natural language processing to determine the intent of the user’s question. Thus, while DialogFlow and IBM Watson are dependent on the user’s Text and Voice input, these platforms ultimately rely on the Agent to decipher the Intent. The Deployment module component of the Chatbot system also depends on the Intents. Deployment codes the Intents understood and passes it to an outside source to deliver the response to the user.

## 4.2 Interprocess Dependencies

The Chatbot will utilize a web browser, network server communications, and HTTP of UMGC City Zoning Application. The client-side components of the software system must operate within everyday web-browser environments on the top of the website.

The user must be connected to the Chatbot. The deployment of the chatbot platform is deployed using an Angular implementation that connects to the trained chatbot platform of Dialog and IBM Watson after providing the necessary information of the agent entities, intents with possible training words related to the entity’s key phrases, and then possible responses. After entering the data, the Chatbot is tested using Angular, NodeJS, JHipster, DBBeaver, and PostgreDB that are free applications to test the chatbot system before connecting to the UMGC City Zoning Application. The final Chatbot deployment is dependent on the backend and database configurations of the UMGC City Zoning Application website.

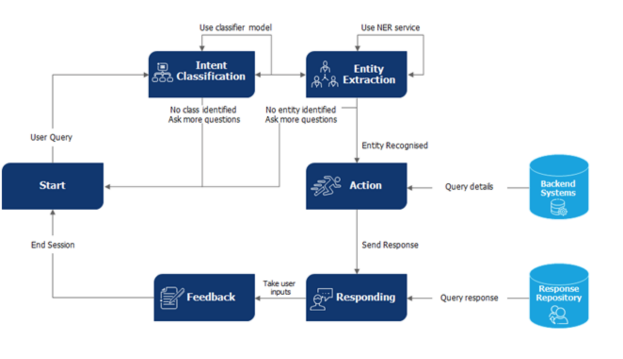
## 4.3 Data Dependencies

The chatbot platform of DialogFlow and IBM Watson-has different components. For UMGC\_CHATBOT agent, we have thirteen entities and intents, using the triggers as possible input from the user, and adding related words and phrases in addition to the triggers stated for the thirteen use cases. The Chatbot response is derived from the post-condition of the use cases, by setting the bot to provide that response based on user request. Currently, the Chatbot response is dependent on the use case post condition response entered. If no related response for the user input, the chatbot suggested to contact the city of Pasadena. Creating more entities and training the natural language processing (NLP) engine which is done in real-time shall make the system cover most of the queries and predict user intents from incoming requests.

# 5 Interface Description

The user will utilize the Chatbot user interface when looking for help while navigating the City of Pasadena website. The system can provide the user with specific regulations, definitions, city’s standards and links to applications and permits. All those services and possible queries are detailed in the SRS document. The chat feature is capable of answering the user questions either typed in or sent as voice messages. The system will be built to fetch data from common keywords configured with the chat engine systems. The responses will be both displayed as text or can be listened to.

The system design follows the Model – View - Controller (MVC) architecture. The View is a UI. Through the UI, the user inputs data or sends an audio message which goes to the controller (Bot engine). The controller transfers data into the model. it processes the request, prepares the result and sends it to the controller. Finally, the controller transfers the generated response into View. The user will be able to view the result. The process sequence is described in the following Figure 2:



**Figure 3- Chatbot architectures Source:** [https://www.wipro.com/holmes/leveraging-object-oriented-approach-for-an-efficient-chatbot-design**/**](https://www.wipro.com/holmes/leveraging-object-oriented-approach-for-an-efficient-chatbot-design/)

## 5.1 Module Interface

This section describes the different screens that define the module's interface of the Chatbot system. The figures below provide both the Chatbot application user interface and the pre-defined interface of the engines used to configure the Chatbot data.

### 5.2.1 Home Screen

Figure 3 shows the landing page or the starting screen of the City of Pasadena website. It contains the chat icon that the user can click on to get help. The icon exists on all the pages of the site.

A picture containing screenshot, outdoor, blue, sign

Description automatically generated

**Figure 4. Module Interface - City of Pasadena Home Page**

### 5.2.2 Screen with Chat Activated

This section describes the screen when the user clicks on the chat icon. In this screen, the chatbot system shall initiate the conversation and invite the user to ask for help. The user can expand the chatbot or end the chat. The messages are either entered on the designated area to type a message or can be sent as audio messages.

A picture containing screenshot, outdoor, sitting, sign

Description automatically generated

**Figure 5. UMCG Chatbot - Chat Activated**

### 5.2.3 Screen with Chat Expanded

This section describes the expanded chat box where the user can see all previous messages which can also be done using a scroll bar of the chatbot.

A screenshot of a social media post

Description automatically generated

**Figure 6. UMGC Chatbot - Chat Expended**

## 5.3 Process Interface

### 5.3.1 DialogFlow Engine Process Interface

The Chatbot configuration is done by the Google engine called DialogFlow that runs on the Google cloud platform. During the configuration of the engine; intents, entities, and responses will be created after linking the Chatbot application IDs with the DialogFlow engine. Once configured, the integrated engine will be called upon when the user types in something related to the requirement that the customer provided. Once the user sends a text or a voice command the engine will translate the information and respond accordingly.

A screenshot of a cell phone

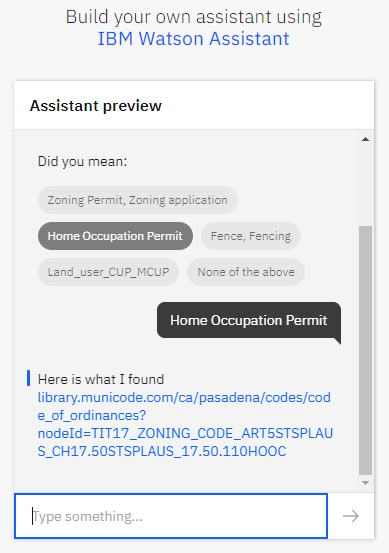
Description automatically generated

**Figure 7. DialogFlow Engine**

### 5.3.2 IBM Watson Engine Process Interface

The Chatbot configuration will be using the IBM Watson engine as well as DialogFlow. The implementation of either one will depend on the customer or the city choice. This section gives a brief description of the screen where the intents, entities, and responses that are configured with the Watson engine as in Figure 5. After the intent and entities are configured, the IMB Watson assistant engine provides the preview with the real data as shown in Figure 6 below.

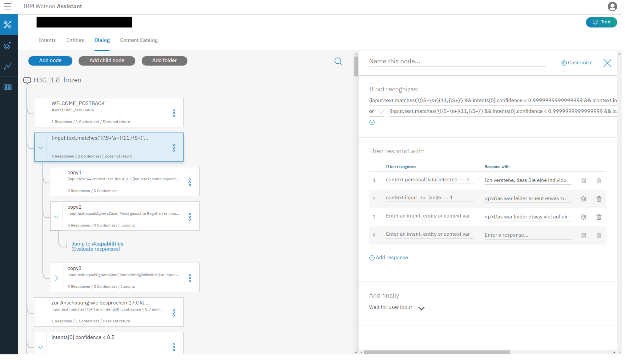
This data shows how the users enter information about the city of Pasadena and based on what is typed the chatbot provides additional suggestions to lead the users on the right path.



**Figure 8. Watson Assistant**

### 5.3.3 Screen Objects and Actions

A discussion of screen objects and actions associated with those objects.



**Figure 9. IBM Watson Assistant**

# 6 Detailed Design

## 6.1 Module Detailed Design

**Text / Voice**

The user interacts with the Chatbot interface through text search or voice search. The Chatbot is a source of communication that users can interact with to obtain information from the City of Pasadena website. With the Chatbot, users will be able to get their questions about the city of Pasadena answered when communicating via typing or voice.

### DialogFlow / IBM Watson Description

DialogFlow and IBM Watson are the bot platforms, the online ecosystem where bots can be deployed, interact with the user, and perform actions on the user’s behalf, including interacting with other platforms. To achieve the goal of the UMGC City Team 2, the Chatbot system will include the use of both the DialogFlow and IBM Watson chat engines to facilitate the implementation of the functional requirements. The requirements and context are based on the requirements that were provided by the customer which was used when creating the Chatbot system.

The UMGC-Chatbotsoftware is based on two chatbot engines, namely DialogFlow and IBM Watson deployments, that are customized to meet the requirements of the City of Pasadena. System architecture concerns the overall structure, the components, and their relationships in the system. The UMGC-Chatbot consists of two major components: A Chatbot system that has use case requirements in the SRS document implemented and a deployment solution for the Chatbot system. IBM Watson has four functional components

The chatbot platforms DialogFlow and IBM Watson have two functional components:

* An agent is a virtual agent that handles conversations with your end-users.
* A Chatbot dialogue UI (User interface)

### Agent

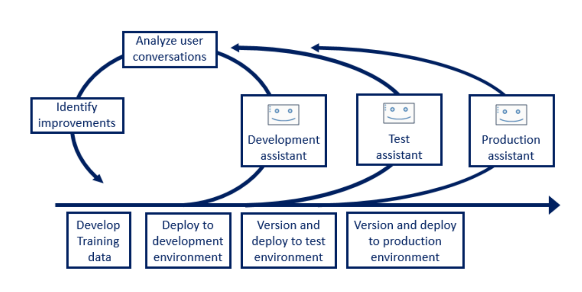
A module within DialogFlow/ IBM Watson, which implements Natural Language Processing to decide what the user intended and to figure out what “action” has to be carried out.

### Intents

Intents support or the service that the user wants from the agent. The developers configure the intent. The intent determines the action by the code. Chatbots use natural language recognition capabilities to decide the intent of what a user is saying and to respond to requests. For this purpose, our rationale for selecting this architecture is the availability of the tool and cost-free of the implementation of the APIs. We considered other architectures; however, this project does not have a financial budget to spend and needs to achieve the goal with publicly accessible resources. But with the implemented APIs they have their limitation which is what makes it free to use. Since it is limited that means that if the limit is passed then the city of Pasadena will need to pay if they need to pass the limitation. Thus, keeping the project flexible and upgradeable.

### Deployment

Deployment involves the code. This part of the conversation passes on the request from the chatbot platform to an external source, and then gets a response and passes it back to the user.



**Figure 10. Chatbot Platform Deployment Diagram**

The deployment of the chatbot platform is deployed using an Angular implementation that connects to the trained chatbot platform, either DialogFlow or IBM Watson.

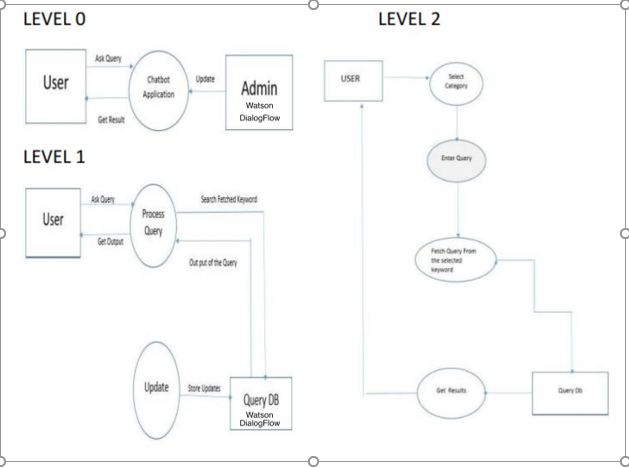
## 6.2 Data Detailed Design

In these sections, we will show and explain the types of data and the size of data.

### 6.2.1 Data Description

The Dialog Flow and IBM Watson based Chatbot receives voice command in addition to text and replies with voice and word at the same time. This happens based on the information provided for the UMGC\_CHATBOT agent created using thirteen entities and intents with the triggers as possible inputs from users by adding as a training phrase from the use cases provided. The Chatbot response is derived from the postcondition of the use cases, by setting the bot to provide that response based on user request. Currently, the Chatbot response is dependent on the use case postconditions response entered. If no related response for the user input, the chatbot suggested to contact the city of Pasadena. However, for future enhancement, the Chatbot logs all questions that were not answered to the database and the Administrator provides the answer to enrich the Chatbot knowledge. Admin can educate the Chatbot to be more efficient to the user. The Chatbot can connect the user to the customer service if the user requests a customer service.

The data flow diagram below describes the data process:



**Figure 11- UML Data Flow for Chatbot System**

The entities and the attributes that chatbot is using are defined as followed:

* Zone\_Land\_Use: This entity stores all land uses applicable to a city by their description. This entity references the city entity.
* Zone: This entity stores all available zone types within a city by their zoning symbol and naming description. This entity references the city entity.
* Allowed\_Land\_Use: This entity links the Zone\_Land\_Use entity to its associated Zone entity. It acts as a bridge entity for the many-to-many relationship between Zone\_Land\_Use and Zone and stores a record for each of the combinations of these other two entities.
* Development\_Standards: This entity stores the URL for each list of development standards that must be followed in each zone for a city. It references the Zone entity.
* Permit\_Type: This entity includes the permit types that must be obtained for allowed land use in each individual zone belonging to a city. It references the Allowed\_Land\_Use entity.
* Application: This entity stores the link to each application that must be submitted for each individual permit type to be obtained. It references the Permit\_Type entity.

### 6.2.2 Data Dictionary

The following table describes the data used by the Chatbot system. This data is extracted from the PostgreSQL database stored on the Heroku cloud platform. The data can be also extracted from the knowledge base.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data Dictionary** | | | | |
| **Entity** | **Field** | **Type** | **NULL** | **Default** |
| **Zone\_Land\_Use** | Id | Primary Key: UUID | No | UUID |
|  | city\_id | Foreign Key: UUID | No | UUID |
|  | description | Varchar (1000) | No |  |
| **Zone** | Id | Primary Key: UUID | No | UUID |
|  | city\_id | Foreign Key: UUID | No | UUID |
|  | description | Varchar (1000) | No |  |
|  | zone\_symbol | Varchar (5) | No |  |
| **Development\_Standards** | Id | Primary Key: UUID | No | UUID |
|  | zone\_id | Foreign Key: UUID | No | UUID |
|  | additional\_standard\_url | Varchar (2000) | No |  |
|  | garden\_standard\_url | Varchar (2000) | No |  |
|  | frontage\_and\_facades\_standards\_url | Varchar (2000) | No |  |
|  | general\_standard\_url | Varchar (2000) | No |  |
| **Allowed\_Land\_Use** | id | Primary Key: UUID | No | UUID |
|  | zone\_id | Foreign Key: UUID | No | UUID |
|  | zone\_land\_use\_id | Foreign Key: uuid | No | uuid |
| **Permit\_Type** | id | Primary Key: UUID | No | UUID |
|  | allowed\_land\_use\_id | Foreign Key: uuid | No | uuid |
|  | name | Varchar (100) | No |  |
|  | description | Varchar (1000) | No |  |
|  | procedure\_url | Varchar (2000) | No |  |
| **Application** | id | Primary Key: UUID | No | UUID |
|  | permit\_id | Foreign Key: UUID | No | UUID |
|  | application\_url | Varchar (2000) | No |  |
|  | name | varchar (50) | No |  |

**Table 2. Data Dictionary**

# 7 REQUIREMENTS MATRIX

The following table provides a cross-reference that traces each system component or entities to the requirements defined in the SRS document. The reference CHB-UC stands for the chatbot use case.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Use Case From SRS** | **Text/Voice**  **component** | **Entity**  **component** | **Agent component** | **Intents**  **component** | **System Design Component** |
| **CHB-UC1** | x | x | x | x | Database Query |
| **CHB-UC2** | x | x | x | x | Database Query |
| **CHB-UC3** | x | x | x | x | Database Query |
| **CHB-UC4** | x | x | x | x | Database Query |
| **CHB-UC5** | x | x | x | x | Database Query |
| **CHB-UC6** | x | x | x | x | Database Query |
| **CHB-UC7** | x | x | x | x | Database Query |
| **CHB-UC8** | x | x | x | x | Database Query |
| **CHB-UC9** | x | x | x | x | Database Query |
| **CHB-UC10** | x | x | x | x | Database Query |
| **CHB-UC11** | x | x | x | x | Database Query |
| **CHB-UC12** | x | x | x | x | Database Query |
| **CHB-UC13** | x | x | x | x | Database Query |

**Table 3. Requirements Matrix**