Municipality Permit ChatBot

Software Requirements Specification (SRS)

For UMGC City Application

Version 1.2

**Prepared by UMGC ChatBot Team**

Rusty Baker

Nathaniel Muesing

Joshua Piersol

Matthew Slaymaker  
Sepribo Taylor-Harry  
Subhash Gandhi Vallala

Revision History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Authors** | **Date** | **Version** | **Description** | **Authors** |
| UMGC ChatBot Team | 06/12/2020 | 1.0 | Initial Release | Nathaniel Muesing, Subhash Gandhi Vallala |
| UMGC ChatBot Team | 07/01/2020 | 1.1 | Style edits | Sepribo Taylor-Harry, Joshua Piersol |
| UMGC ChatBot Team | 07/23/2020 | 1.2 | Use Case tables added | Subhash Gandhi Vallala, Joshua Piersol, Rusty Baker |

Project Plan Approvals

|  |  |  |
| --- | --- | --- |
| **Name** | **Signature** | **Date** |
| Approved by:  Dr. Mir Assadullah  "Stakeholder" |  |  |
| Approved by:  "Project Manager" |  |  |

Table of Contents

1. Introduction 5

1.1 Purpose 5

1.2 Scope 6

1.3 Definitions, Acronyms and Abbreviations 7

1.4 References 8

1.5 Overview 8

2. Overall Description 8

2.1 Use-Cases 9

**2.1.1** **Specify Zoning** 9

**2.1.2** **Upload Zoning** 10

**2.1.3** **Add Permit Type** 10

**2.1.4** **Add Permit URLs** 11

**2.1.5** **Add Regulation Type** 11

**2.1.6** **Add Regulation URLs** 11

**2.1.7** **Upload Permit and Regulations** 12

**2.1.8** **Delete Zone** 12

**2.1.9** **Delete Permit** 12

**2.1.10** **Delete Regulation** 13

**2.1.11** **Chatbot Greeting** 13

**2.1.12** **Chatbot Create Zone Context** 13

**2.1.13** **Zoning Request** 14

**2.1.14** **Zone Displayed** 14

**2.1.15** **Permit Request** 14

**2.1.16** **Regulation Request** 15

**2.1.17** **URL Links Displayed** 15

**2.1.18** **Chatbot Use Help** 15

**2.1.19** **Additional Help** 16

2.2 Use-Case Diagram 16

2.3 Data Flow Diagram 19

2.4 Use-Case Model Survey 20

2.5 User Classes and Characteristics 21

2.6 Assumptions and Dependencies 21

3. Specific Requirements 22

3.1 System Features/Modules 22

**3.1.1** **Chatbot** 23

**3.1.2** **City Official Client** 25

Software Requirements Specification

# Introduction

The understanding of any given website can be unclear to users. While adhering to norms that have been established for website elements aids in use, one way to assist users is to provide a chatbot service. Chatbot services provide a level of user support that is focused on many easy to solve problems or requests that users may have. Providing a responsive help option resolves many issues that would otherwise require additional time and resources.

This software requirements specification document details the municipality permit chatbot functionality and expected performance. This chatbot application will be used by city residents facilitating city website navigation, distribution of city regulation knowledge, and permit application process. Part of this application includes a city official client to be used for modifying information offered through the chatbot.

## Purpose

This SRS will serve as an outline for the permit chatbot functional requirements and goals. Requirements listed in this document are to be used to track progress and facilitate understanding between the chatbot team and stakeholders. The chatbot team should verify these requirements meet realistic stakeholder expectations. Once development has been initiated, any additional requirements levied on the chatbot team deemed not essential should be documented as optional not to impede a functional deliverable release.

Requirements listed in this SRS will be used to populate a Scrum product backlog. Modifications to requirements should be first edited in this document and formally signed off on. Then the product backlog can be altered to match these modified requirements. This process is stressed so that this document can serve as a control of what goes into the product backlog.

## Scope

The outcome of this project is to deliver a chatbot application that city residents can interact with to explore, locate, and assist with downloading permit applications. Webpages related to zoning and city regulations will be provided to residents based on interactions with the Chatbot. Interactions not covered will fall into a catch-all interaction. This catch will all do some default behavior, such as suggesting the resident utilize the communication methods on the city's contact web page. These interactions with residents will be handled using IBM's Watson Assistant conversation AI platform. This PaaS allows a series of conditioned responses based on input. While the software utilized in this project is focused on being free and opensource, the Watson Assistant does offer additional functionality with tiered price categories. For the development of this project, the free tier will be used that allows 10,000 messages a month.

The scope also includes a downloadable component that allows city officials to customize the offerings of the Chatbot. City officials will be able to modify what permit applications and external links the Chatbot can provide to residents. This client for city officials can alter available interactions through the Watson Assistant API. It is expected that the city official(s) will have a running instance of Watson Assistant running in IBM's cloud environment before any API interactions can occur. Any additions that are not related to zoning, regulations, or permit applications are out of scope for this project. This client should allow other municipalities to tailor the Chatbot to their specific area.

## Definitions, Acronyms and Abbreviations

**Table 1**

Acronym and Abbreviations

|  |  |
| --- | --- |
| Acronym/Abbreviation | Definition |
| AI | Artificial Intelligence |
| API | Application Programming Interface |
| Chatbot | A language recognition application that responds to user input and requests |
| DevOps | The combination of software development and information technology operations into a single process to enable continuous delivery |
| HTTP | Hypertext Transfer Protocol |
| PaaS | A Platform as a service is a computing service offered in a cloud environment providing users with an environment for developing, managing, and running infrastructure without the need to own the physical hardware. |
| Scrum | An Agile like software development methodology that emphasizes short progress tracking meetings, a backlog list of requirements, and cyclical development cycles |
| SRS | Software Requirements Specification |
| URL | Uniform Resource Locator also termed as a web address |
| UI | User Interface |

## References

ESRI. (n.d.). Make Maps. *ArcGIS Online*.

https://www.esri.com/en-us/arcgis/products/arcgis-online/capabilities/make-maps

IBM. (2020, March 12). *Watson Assistant API overview*. IBM Cloud Docs. https://cloud.ibm.com/docs/assistant?topic=assistant-api-overview

OpenLayers. (n.d.). Editable ArcGIS REST Feature Service.

https://openlayers.org/en/latest/examples/vector-esri-edit.html?q=arcgis

## Overview

The rest of this SRS document covers the overall description and specific requirements for this project. The overall description is a top-down view of functional requirements. Specific requirements list in detail how the system is expected to function in relation to those requirements.

# Overall Description

A chatbot uses artificial intelligence algorithms to establish confidence parameters for categories that match the given input. By specifying categories that responses fit into the Chatbot can be trained to respond in a reasonable manner to a request. This behavior is typically modeled on what is expected to be a natural flow for person-to-person communication. Should the Chatbot be unable to match a given input to a specific response category, a default response is supplied, suggesting means to obtain further help.

Responses are not expected to be thoroughly detailed. Instead, any response that requires a large amount of information should display a web page link to that information. This allows the chatbot to be used with already created web pages detailing city regulations, zoning, and permits. As such, the chatbot is not a replacement for currently existing informational pages but merely a helpful guide. For example, if a city resident wishes to view solar power permits for their residential zone will be prompted to provide an address and permit of interest. This submission will prompt the chatbot to display a web link to solar permits for that zone. Users of the chatbot are expected to have some interest in zoning, permits, or related city regulations.

To initiate the Chatbot with available categories, the city official client will allow the input of zones and URLs to provided permits and city regulations. It should be noted that this can also be done by the maintainer of the IBM cloud environment for the instance of the Watson Assistant. This client will allow the response of s given URL to be mapped to the request for information. This request will be specific to the zone designated with it.

## Use-Cases

The chatbot system 19 use cases of which ten have the City Officials as the primary actor and nine have the City Residents as the primary actor. The use case tables below provide a detailed documented model of all the features of the service:

### **Specify Zoning**

|  |  |
| --- | --- |
| Trigger: | The user wants to define a new city zone. |
| Primary Actor: | City Official |
| Supporting Actors: | Address Service |
| Pre-conditions: | The user must have admin privileges. |
| Steps in the Process: | 1. Launch the application. 2. Click Specify Zoning. 3. Select an area on the representation of the city map. 4. Click Specify selection as a Zone. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | The selected area on the representation of the city map will be defined as a city zone. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | None |

### **Upload Zoning**

|  |  |
| --- | --- |
| Trigger: | The user wants to upload a city zone to Watson Assistant as a category. |
| Primary Actor: | City Official |
| Supporting Actors: | Address Service |
| Pre-conditions: | The user must have admin privileges. |
| Steps in the Process: | 1. Launch the application. 2. Click Upload Zoning. 3. Select the zone file. 4. Click Upload Zone. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | The selected file will be processed, and the zone will be added to the database. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | 1. The uploaded file is not of the correct format. 2. The contents of the file uploaded are corrupt. |

### **Add Permit Type**

|  |  |
| --- | --- |
| Trigger: | The user wants to add a new permit type. |
| Primary Actor: | City Official |
| Supporting Actors: | 1. Address Service 2. Document Service |
| Pre-conditions: | The user must have admin privileges. |
| Steps in the Process: | 1. Launch the application. 2. Click Add Permit Type 3. Select the permit file. 4. Click Add Permit. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | The selected file will be processed, and the permit will be added to the database. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | 1. The uploaded file is not of the correct format. 2. The contents of the file uploaded are corrupt. |

### **Add Permit URLs**

|  |  |
| --- | --- |
| Trigger: | The user wants to add a link to a permit type. |
| Primary Actor: | City Official |
| Supporting Actors: | Document Service |
| Pre-conditions: | The user must have admin privileges. |
| Steps in the Process: | 1. Launch the application. 2. Click Add Permit URLs 3. Select a permit. 4. Enter the URL. 5. Click Link URL to Permit. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | The selected permit will be linked to the URL entered. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | The entered URL is invalid. |

### **Add Regulation Type**

|  |  |
| --- | --- |
| Trigger: | The user wants to add a new regulation. |
| Primary Actor: | City Official |
| Supporting Actors: | 1. Address Service 2. Document Service |
| Pre-conditions: | The user must have admin privileges. |
| Steps in the Process: | 1. Launch the application. 2. Click Add Regulation Type 3. Select the regulation file. 4. Click Add Regulation. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | The selected file will be processed, and the regulation will be added to the database. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | 1. The uploaded file is not of the correct format. 2. The contents of the file uploaded are corrupt |

### **Add Regulation URLs**

|  |  |
| --- | --- |
| Trigger: | The user wants to add a link to a regulation. |
| Primary Actor: | City Official |
| Supporting Actors: | Document Service |
| Pre-conditions: | The user must have admin privileges. |
| Steps in the Process: | 1. Launch the application. 2. Click Add Regulation URLs 3. Select a regulation. 4. Enter the URL. 5. Click Link URL to Regulation. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | The selected regulation will be linked to the URL entered. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | The entered URL is invalid. |

### **Upload Permit and Regulations**

|  |  |
| --- | --- |
| Trigger: | The user wants to upload permit and regulations. |
| Primary Actor: | City Official |
| Supporting Actors: | 1. Address Service 2. Document Service |
| Pre-conditions: | The user must have admin privileges. |
| Steps in the Process: | 1. Launch the application. 2. Click Upload Permit and Regulations. 3. Select the permit file and the related regulation file. 4. Click Upload. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | The selected files will be processed, and the permit with the regulation will be added to the database. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | 1. The uploaded file is not of the correct format. 2. The contents of the file uploaded are corrupt |

### **Delete Zone**

|  |  |
| --- | --- |
| Trigger: | The user wants to delete a zone. |
| Primary Actor: | City Official |
| Supporting Actors: | Address Service |
| Pre-conditions: | The user must have admin privileges. |
| Steps in the Process: | 1. Launch the application. 2. Click Delete Zone. 3. Select a zone. 4. Click Confirm. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | The selected zone will be removed from the database. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | None |

### **Delete Permit**

|  |  |
| --- | --- |
| Trigger: | The user wants to delete a permit. |
| Primary Actor: | City Official |
| Supporting Actors: | 1. Address Service 2. Document Service |
| Pre-conditions: | The user must have admin privileges. |
| Steps in the Process: | 1. Launch the application. 2. Click Delete Permit. 3. Select a permit. 4. Click Confirm. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | The selected permit will be removed from the database. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | None |

### **Delete Regulation**

|  |  |
| --- | --- |
| Trigger: | The user wants to delete a regulation. |
| Primary Actor: | City Official |
| Supporting Actors: | 1. Address Service 2. Document Service |
| Pre-conditions: | The user must have admin privileges. |
| Steps in the Process: | 1. Launch the application. 2. Click Delete Regulation. 3. Select a regulation. 4. Click Confirm. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | The selected regulation will be removed from the database. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | None |

### **Chatbot Greeting**

|  |  |
| --- | --- |
| Trigger: | The user will be greeted by the chatbot and then asked for a zone or an address. |
| Primary Actor: | City Resident |
| Supporting Actors: | 1. Address Service 2. Dialog Service |
| Pre-conditions: | None. |
| Steps in the Process: | Launch the application. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | The user will be greeted with one of the predefined dialogs and then the user will be prompted to enter a zone or an address. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | User input does not match any zone or address in the database. |

### **Chatbot Create Zone Context**

|  |  |
| --- | --- |
| Trigger: | The user entered city zone is looked up using an external geolocation solution. |
| Primary Actor: | City Resident |
| Supporting Actors: | Address Service |
| Pre-conditions: | The user has entered a valid zone. |
| Steps in the Process: | Watson Assistant calls an external geolocation solution using user inputted zone. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | The zone is displayed as valid in the chat box. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | None. |

### **Zoning Request**

|  |  |
| --- | --- |
| Trigger: | The user wants to enter an address. |
| Primary Actor: | City Resident |
| Supporting Actors: | 1. Address Service 2. Dialog Service |
| Pre-conditions: | The chatbot has already greeted the user. |
| Steps in the Process: | The user is prompted for an address. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | The user input is validated and accepted. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | The user input format is invalid. |

### **Zone Displayed**

|  |  |
| --- | --- |
| Trigger: | The user wants to look at the zone defined by the address entered. |
| Primary Actor: | City Resident |
| Supporting Actors: | 1. Address Service 2. Dialog Service |
| Pre-conditions: | The user has entered a valid address. |
| Steps in the Process: | The user inputted address is queried to find a matching zone. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | The zone matching the user inputted address is displayed in the chat box. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | The user entered address is not assigned to any existing zones. |

### **Permit Request**

|  |  |
| --- | --- |
| Trigger: | The user wants to look at the details of a permit. |
| Primary Actor: | City Resident |
| Supporting Actors: | 1. Address Service 2. Dialog Service 3. Document Service |
| Pre-conditions: | The chatbot has already greeted the user. |
| Steps in the Process: | The user is prompted for a permit type. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | The user input is validated and accepted. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | The user input format is invalid. |

### **Regulation Request**

|  |  |
| --- | --- |
| Trigger: | The user wants to look at the details of a regulation. |
| Primary Actor: | City Resident |
| Supporting Actors: | 1. Address Service 2. Dialog Service 3. Document Service |
| Pre-conditions: | The chatbot has already greeted the user. |
| Steps in the Process: | The user is prompted for a regulation type. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | The user input is validated and accepted. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | The user input format is invalid. |

### **URL Links Displayed**

|  |  |
| --- | --- |
| Trigger: | The user wants to navigate to an external web page. |
| Primary Actor: | City Resident |
| Supporting Actors: | 1. Dialog Service 2. Document Service |
| Pre-conditions: | None. |
| Steps in the Process: | Click External Links. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | The link to external website for the city's permits and regulations is displayed. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | None |

### **Chatbot Use Help**

|  |  |
| --- | --- |
| Trigger: | The user wants to learn how to use the application better. |
| Primary Actor: | City Resident |
| Supporting Actors: | Dialog Service |
| Pre-conditions: | None. |
| Steps in the Process: | Click Chatbot Help. |
| Minimal Guarantees: | The user will get a feedback. |
| Success Guarantees: | Short directions on the use of the Chatbot are displayed. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | None |

### **Additional Help**

|  |  |
| --- | --- |
| Trigger: | The user wants to get in touch with a City Official for additional information. |
| Primary Actor: | City Resident |
| Supporting Actors: | Dialog Service |
| Pre-conditions: | None. |
| Steps in the Process: | Click Additional Help. |
| Minimal Guarantees: | The user will get feedback. |
| Success Guarantees: | The city contact page link is displayed. |
| Quality Requirements: | 1. The feedback provided to the user must be clear. 2. Minimum and success guarantees must be included. |
| Alternative Flows and Exceptions: | None |

## Use-Case Diagram

The Use-Case Diagram is used to demonstrate the different ways that a user might interact with a system. It summarizes the details of the system's users or actors and their interactions with the system.

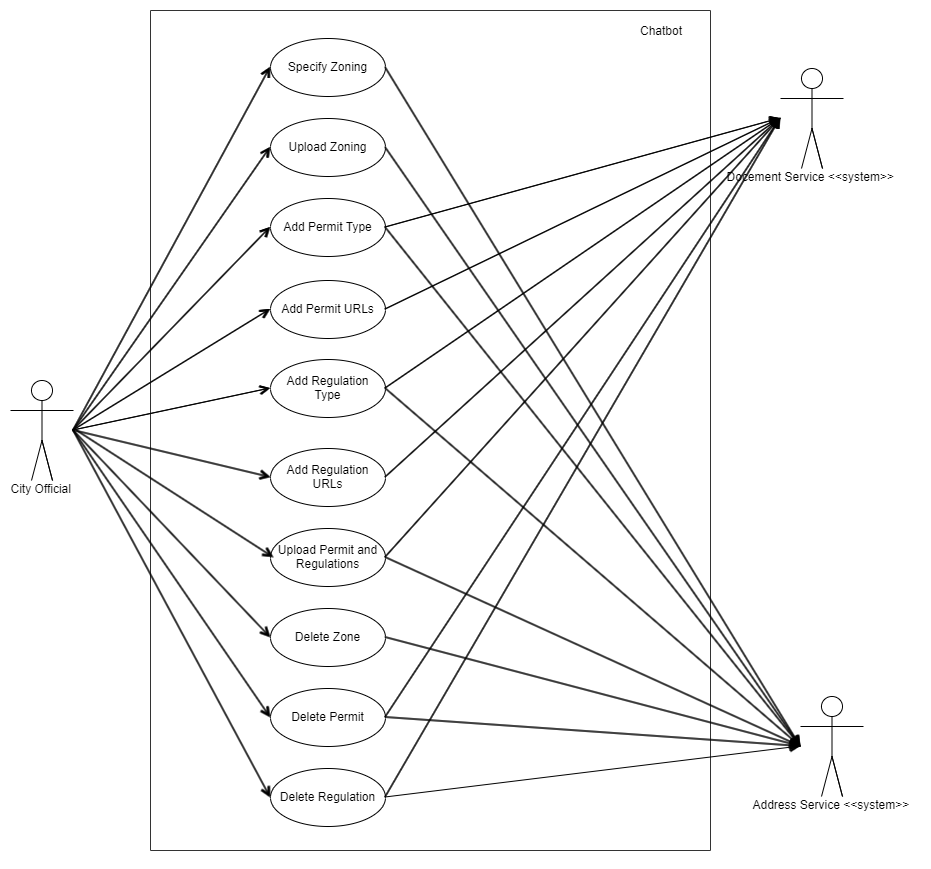
Figure 1: 2.1.1 Chatbot Use Case Diagram

Figure 1 depicts the Use Case Diagram of the Chatbot system that has the City Official as the primary actor, providing a depiction of the system's expected behavior. The secondary actors here include the Document Service that will handle all the creation and storage of permits and regulations, and the Address Service that will handle all the residents' addresses and zones.

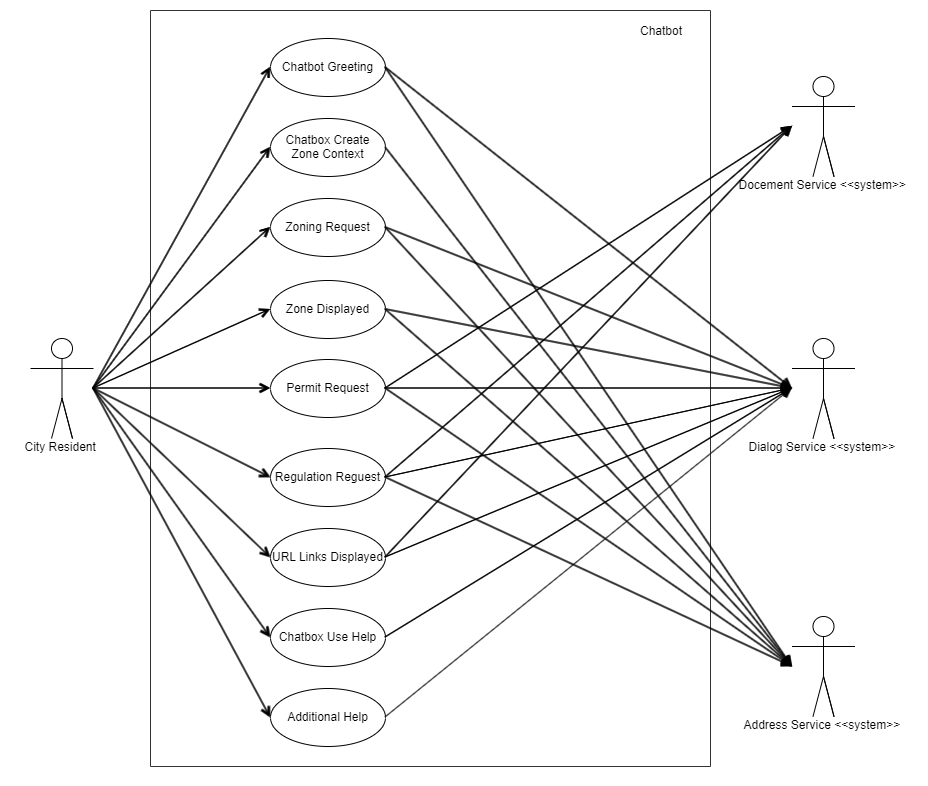


Figure 2: 2.1.2 Chatbot Use Case Diagram

Figure 2 depicts the Use Case Diagram of the Chatbot system that has the City Residents as the primary actor, providing a depiction of the system's expected behavior. The secondary actors here include the Document Service that will handle all the fetching of Permits and Regulations, the Dialog Service that will handle all the conversations with the Users, and the Address Service that will handle all the city's address and zones. Furthermore, this document provides insight into the design of the Chatbot system from the end user's perspective, communicating the behavior in the user's terms by specifying all externally visible system behavior.

## Data Flow Diagram

The data flow diagram is used to graphically represent the flow of data in the system, it describes the system transfer data from the input to the file storage and to finally deliver the requested information.



Figure 3: 2.2.1 Chatbot Data Flow Diagram

Figure 3 depicts the Data Flow Diagram of the Chatbot system that begins with the welcome/introductory message, followed by the resident being asked to enter a complete address or zone. If user provides a valid address or zone, they will be prompted for their interests in permits/regulations. A successful lookup of the permit or regulation is achieved by the Chatbot displaying the related city webpage link to the resident. If there is no related permit or regulation, the Chatbot respond with a simple phrase such as "no related permit/regulation could be found", and prompt if the user would like city official contact information to submit an inquiry, and then loops back.

## Use-Case Model Survey

**Table 2**

Use-Cases and Description

|  |  |
| --- | --- |
| **Use Case** | **Description** |
| Specify Zoning | City zones are defined by city officials selecting an area on a representation of a map. |
| Upload Zoning | Zones are uploaded to Watson Assistant as a category. Do you think they should be part of the address entity? |
| Add Permit Types | Permits are entered that will be used to match user input for permit types. |
| Add Permit URLs | A URL is provided with a permit type and linked together. |
| Add Regulation Type | Regulations are entered that will be used to match user input for regulation types. |
| Add Regulation URLs | A URL is provided with a regulation type and linked together. |
| Upload Permit and Regulations | Permits and related regulations are uploaded together with a list of related zones |
| Delete Zone | A zone is deleted from the list of zones |
| Delete Permit | A permit type is deleted as well as the URL. |
| Delete Regulation | A regulation type is deleted as well as the URL. |
| Chatbot Greeting | Chatbot greets the user and asks for a zone or address to be entered. |
| Chatbot Create Zone Context | Watson Assistant programmatic calls are made to find the user's entered city zone using an external geolocation solution. |
| Zoning Request | Chatbot user is prompted for address. |
| Zone displayed | A City zone that corresponds to the address given in a zoning request that is displayed. |
| Permit Request | Chatbot user is prompted for address and permit type of interest. |
| Regulation Request | Chatbot user is prompted for permit type and specifies they want related regulations. |
| URL Links Displayed | URLs that link to relevant web pages are displayed to the user. |
| Chatbot Use Help | Short directions on the use of the Chatbot are displayed to the user upon request for help on use. |
| Additional Help | The city contact page link is displayed when the user requests advance help. |

## User Classes and Characteristics

There are three different classes of users that will be interacting with the city chatbot. Each of these users will be interested in the tool for different reasons and each will require a different set of authorities.

* Administrator: User enforces city official client authentication policies with running Watson Assistant.
* City Resident: User that is searching for zoning, city regulation, or permit information from a municipality's web site.
* City Official: User is employed by the municipality to be a custodian of zoning information and how it is linked to city regulations/permits.

## Assumptions and Dependencies

Due to the complex nature of the application, there are several dependencies that will need to be met. This same nature also leaves some room for unexpected situations and thus certain assumptions must be made. This section will list out both the assumptions and the dependencies.

* This chatbot relies on a stable internet connection to submit and return requests. In the event of a disconnection from the internet, it is assumed that no state is saved between the chatbot and a resident.
* An internet connection is required for both the chatbot and city official client to function.
* It is assumed the city using this chatbot has a designated staff member who can maintain the Watson Assistant in addition to their other responsibilities.
* It is assumed that city residents have some familiarity with using web browsers.
* It is assumed that one instance of the chatbot and IBM Watson Assistant are tied to one city.
* It is assumed that the city using the chatbot has a contact information page with contacts such as a phone number or email address.
* The city official client is dependent on the IBM Watson Assistant API. Future Watson Assistant API updates may require the client to be updated.

# Specific Requirements

The pre-condition and expected outcomes of the use cases listed in section 2.1 that make up this software are in this section.

## System Features/Modules

The Chatbot and city official client is described in the following separate subsections.

### **Chatbot**

This is the chatbot application that assists city residents with municipal web site navigation. It is integrated with an instance of IBM's Watson Assistant to determine what city residents are requesting.

#### **Stimulus/Response Sequences**

Stimulus: User activates the Chatbot by clicking on it.

Response: Chatbot displays a greeting and requests a city zone or address.

Stimulus: User requests information on a permit type.

Response: Chatbot displays a URL to that permit information based on the zone/address the user had entered.

Stimulus: User requests information on a city regulation.

Response: Chatbot displays a URL to city regulation based on the zone/address the user had entered.

Stimulus: User request for a city zone.

Response: Chatbot prompts the user for address.

Stimulus: User request for the zone of an address.

Response: Chatbot either picks out the keyword such as address or zip code from the question and looks for the city zone that corresponds to the address.

Stimulus: User asks for help on chatbot use.

Response: Chatbot responds with general support questions questioning specific focus of help.

Stimulus: User has entered an unknown query.

Response: The frequently asked questions are displayed.

Stimulus: User has entered multiple unknown queries.

Response: The city contact page link is displayed.

#### **Functional Requirements**

Following are functional requirements that the team will be working on. These requirements may need further grooming as the understanding of the project evolves and as new challenges are discovered during this project.

REQ-1.1: Users shall be given a chatbot message box to submit queries.

REQ-1.2: Users shall be able to submit a query in the message box by pressing enter or selecting a submit button.

REQ-1.3: Users shall be prompted for a zone or address.

REQ-1.4: The address provided will be validated by the tool.

REQ-1.5: The user's entered zone or address shall be used for subsequent permit and regulation queries.

REQ-1.7: The city frequently asked questions page URL shall be displayed when users submit a query that is not relevant to zoning, city regulations, or permits.

REQ-1.8: The city contact page URL shall be displayed when users submit multiple queries that are not relevant to zoning, city regulations, or permits.

REQ-1.9: The Chatbot shall offer different zoning, city regulation, and permit informational responses that scale based on input from the city official client.

REQ-1.10: The Chatbot shall be able to be added to a municipality's website.

REQ-1.11: The Chatbot shall display help information on its use when the user asks for help in the message box or clicks a help icon.

REQ-1.12: The Chatbot shall ask for an address when the user submits a query asking for their zone.

### **City Official Client**

This client (see figure 4 & 5 below) gives city officials the ability to add, modify, and delete what is available through chatbot requests. It utilizes the Watson Assistant API to update the chatbot server.

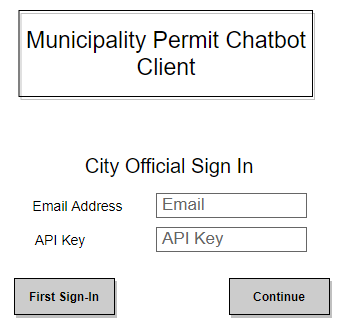


Figure 4: 3.1.2 Chatbot Login

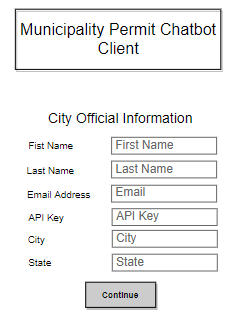


Figure 5: 3.1.2 Chatbot City Official Information

#### **Functional Requirements**

REQ-2.1: The city official client shall allow city officials to add, modify, and delete chatbot responses and their corresponding questions.

REQ-2.2: The city official client shall allow city officials to define zoning areas visually on a presented map.

REQ-2.3: Defined zoning areas shall be persistently stored.

REQ-2.4: The city official client shall allow officials to link regulation URLs to regulation responses to user queries.

REQ-2.5: The city official client shall allow officials to link permit URLs to permit responses to user intents. The user intents will be interpreted from the user queries.

REQ-2.6: The city official client shall allow officials to associate permits and regulations to a zoning area(s).

REQ-2.7: The city official client shall be able to connect to a running IBM Watson assistant when an internet connection is possible.

REQ-2.8: The city official client shall be able to be maintained/packaged in such a way that it can be pulled from a storage solution onto a workstation.

REQ-2.9: The city official client shall run on Windows 8 and 10 operating systems.