COP 5339 – FINAL PROJECT REPORT

THE NINA CHATBOT (VIRTUAL ASSISTANT)

RAFAEL PAPA & FRANCIS IFON (PROJECT TEAM #19)

4/28/2019

1.1 INTRODUCTION

A chatbot (or virtual assistant) is a software that can simulate a conversation with a user in natural language through messaging applications. A chatbot is often described as one of the most advanced and promising expressions of interaction between humans and machines. However, from a technological point of view, a chatbot only represents the natural evolution of a Question Answering system leveraging Natural Language Processing (NLP). Chatbot virtual assistants are increasingly being used to handle simple, look-up tasks in both business-to-consumer (B2C) and business-to-business (B2B) environments. The chatbot assistants not only reduce overhead costs by making better use of support staff time, it also allows companies to provide a level of customer service during hours when live agents aren't available.

This project is a tool to help in managing home owners requests in a condo association and it is more useful when there is nobody in the office to receive and process user requests in-person.

1.2 PURPOSE

To develop a fully functional and user interactive chatbot which can help the HOA (Home Owners Association) to manage daily basis requests (and general information) from users (homeowners).

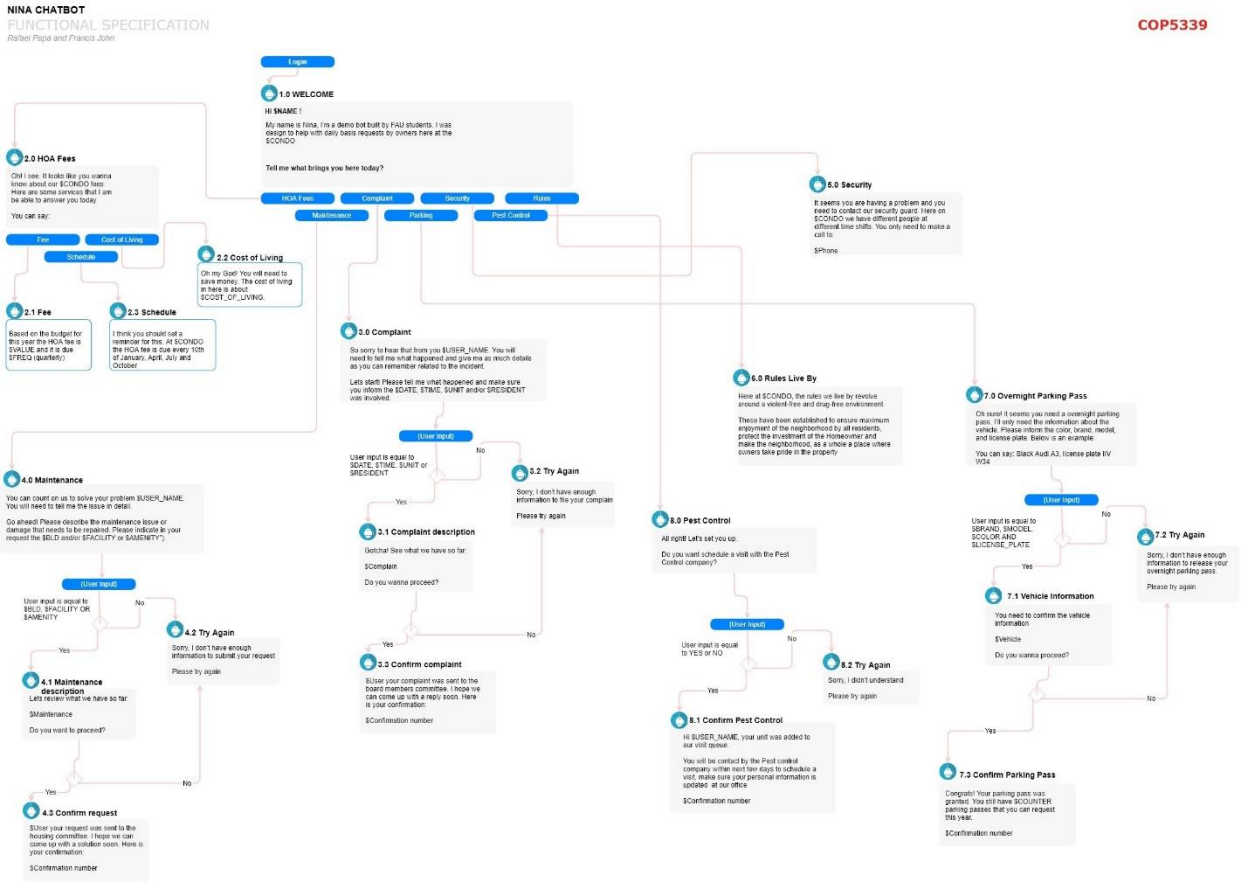
1.3 SCOPE

- Authenticate the user in the application .
- Connect to the external API (The IBM Watson)
- Manage all the messages
- Maintain the connection

1.4 TECHNOLOGIES

- **Application Architecture:** JAVA 8
- **External API:** The IBM Watson
- **Database Application:** The Watson
- **Development Tool:** Eclipse Oxygen la
- **Designing Tool:** PlantUML

2.1 FUNCTIONAL SPECIFICATIONS



2.2 NON-FUNCTIONAL SPECIFICATIONS
2.2.1 CRC CARDS

Message	
Manage message contents	ChatBot

WatsonConnection	
Manage Watson Assistant connections	ChatBot
	WatsonConnection

Chatbot	
Get input from user	WatsonConnection
Manage Watson Assistant	DBConnection
Manage Database System	Complaint
Manage Message Type	Maintenance
	ParkingPass
	PestControl

DatabaseConnection	
Manage database connections	Chatbot
	DatabaseConnection

Complaint	
Manages user complaint	Chatbot
	User

Maintenance	
Manages user maintenance issues	Chatbot

ParkingPass	
Manage overnight parking pass	Chatbot
	User
	Car

PestControl	
Schedules pest control visit	Chatbot
	User
	PestControlQueue

PestControlQueue	
Adds and removes pest control issues	

Car	
Manage car contents	

2.2.2 USE CASES

#Log in

1. The user accesses the application.
2. The user enters the password.
3. The chatbot system validates the password.
4. The chatbot assistant greets the user.
5. The chatbot assistant prompts the message: "Hi $USER, how may I help you today?".

#Request HOA fee Information

1. The user carries out **log in**.
2. The user enquires about the “HOA fees”.
3. The chatbot assistant suggests an option list: “I can inform you about:”
    a. Amount
    b. Cost of living
    c. Schedule and due date
4. The user selects his/her preferred option based on his/her intent.
5. The chatbot assistant replies with a detailed information.

#File a Complaint

1. The user carries out **log in**.
2. The user carries out with: "I want to file a complaint."
3. The chatbot assistant informs the user a list of requirements needed to file a complaint.
    a. Date
    b. Time
    c. Unit
    d. Responsible Resident/Visitor
4. The user provides the information.
5. The chatbot assistant informs the user his/her complaint has been sent to the board members.
6. The chatbot assistant provides the confirmation number to the user.

#Report Maintenance Problem

1. The user carries out **log in**.
2. The user carries out with: "I want to report a maintenance problem."
3. The chatbot assistant informs the user a list of requirements needed to report a maintenance problem.
    a. Building
    b. Facility
    c. Amenity
4. The user inputs the required details.
5. The chatbot assistant informs the user his/her request has been sent to the housing committee.
6. The chatbot assistant provides the confirmation number to the user.

#Request Security Guard Contact

1. The user carries out **log in**.
2. The user carries out with: "I need to contact the security guard”.
3. The chatbot assistant provides the security guard contact to the user

#Request Rules Live By

1. The user carries out **log in**.
2. The user carries out with: "I would like to know which the rules we live by are".
3. The chatbot assistant provides the rules we live by to the user

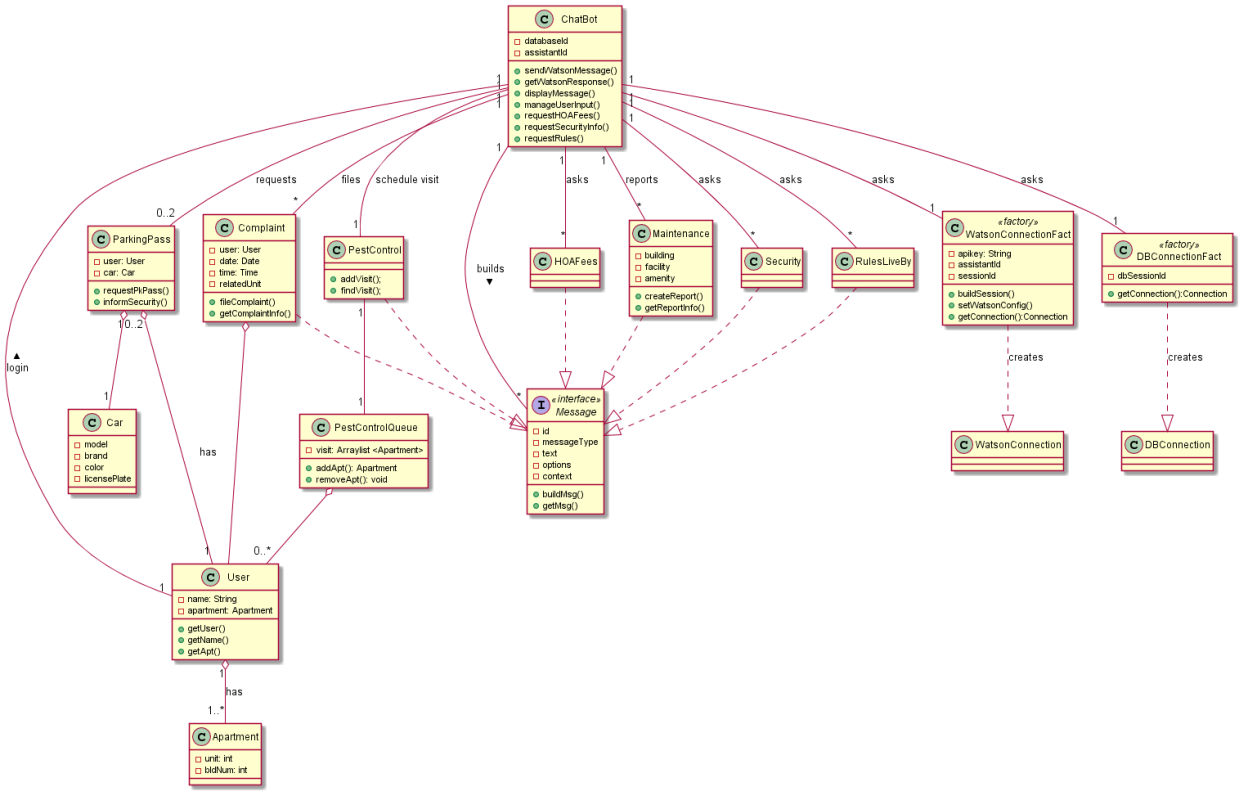
#Request Overnight Parking Pass

1. The user carries out **log in**.
2. The user carries out with: "I need an overnight parking pass".
3. The chatbot assistant informs the user he needs to provide car information
4. The chatbot assistant informs the user a list of requirements needed to provide an overnight parking pass.
    a. Color
    b. Brand
    c. Model
    d. License Plate
5. The user inputs the required details.
6. The chatbot system informs the security guard.
7. The chatbot assistant informs the user the overnight parking pass was granted.
8. The chatbot assistant provides the confirmation number to the user.

#Schedule a Pest Control Visit

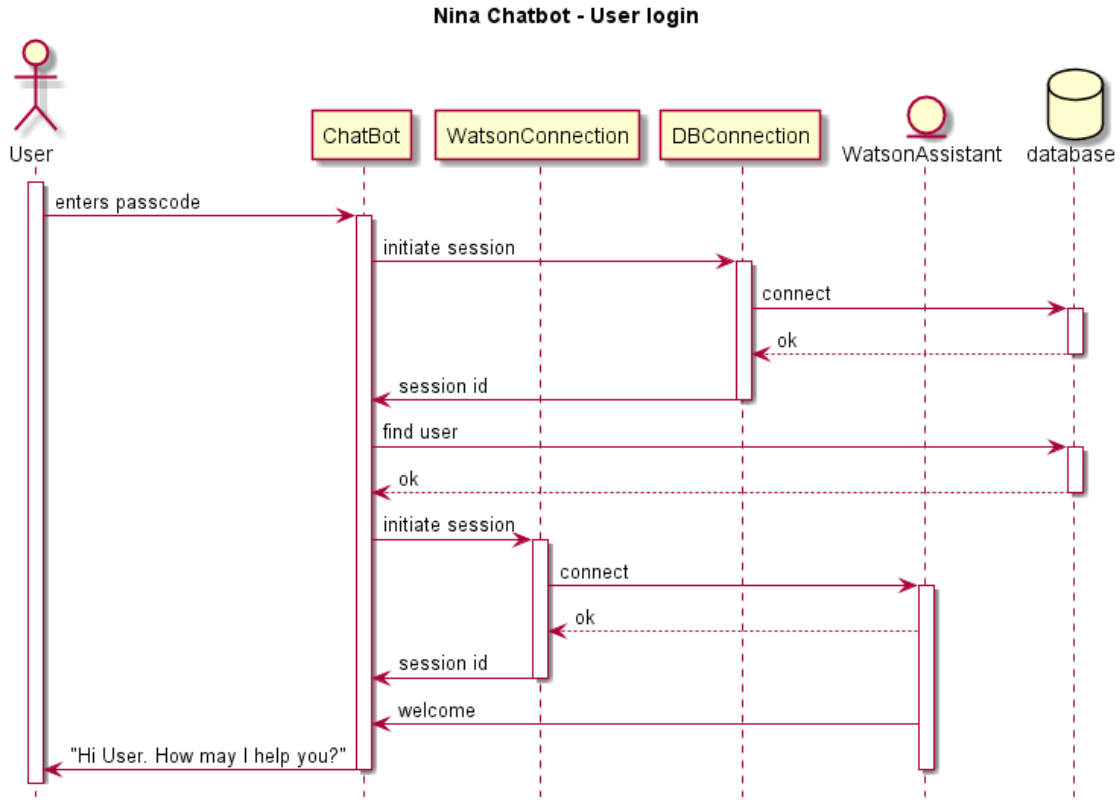
1. The user carries out **log in**.
2. The user carries out with: "I want to schedule a pest control visit to my unit".
3. The chatbot assistant asks user for confirmation
4. The chatbot system add the request to the queue of units to be visit
5. The chatbot assistant informs the user the pest control company will be in touch in a few days
6. The chatbot assistant provides the confirmation number to the user.

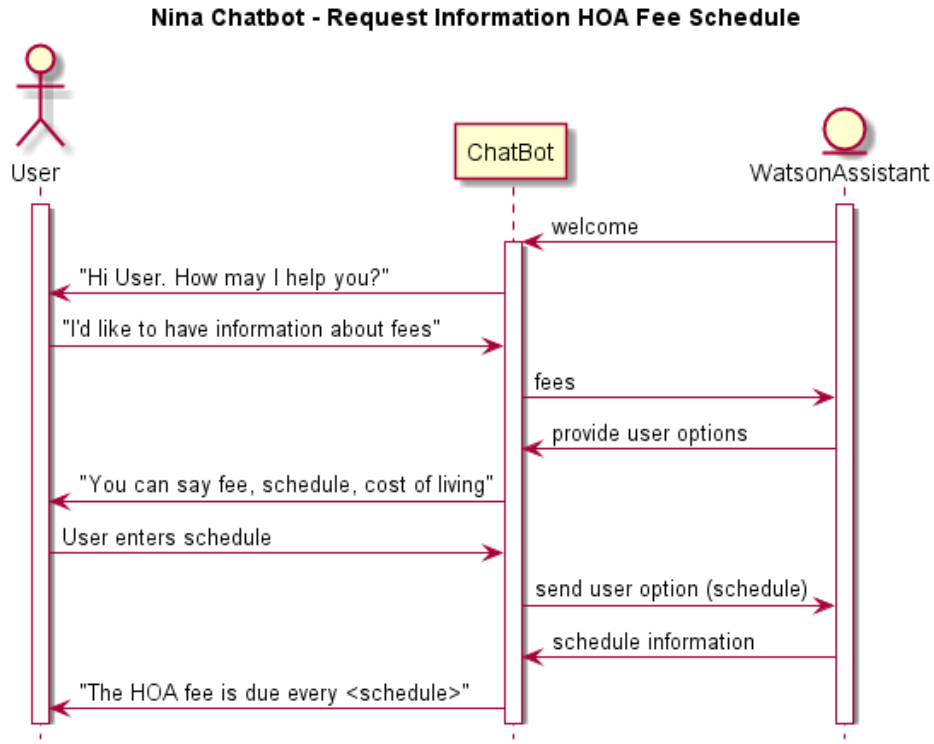
2.2.3 CLASS DIAGRAM



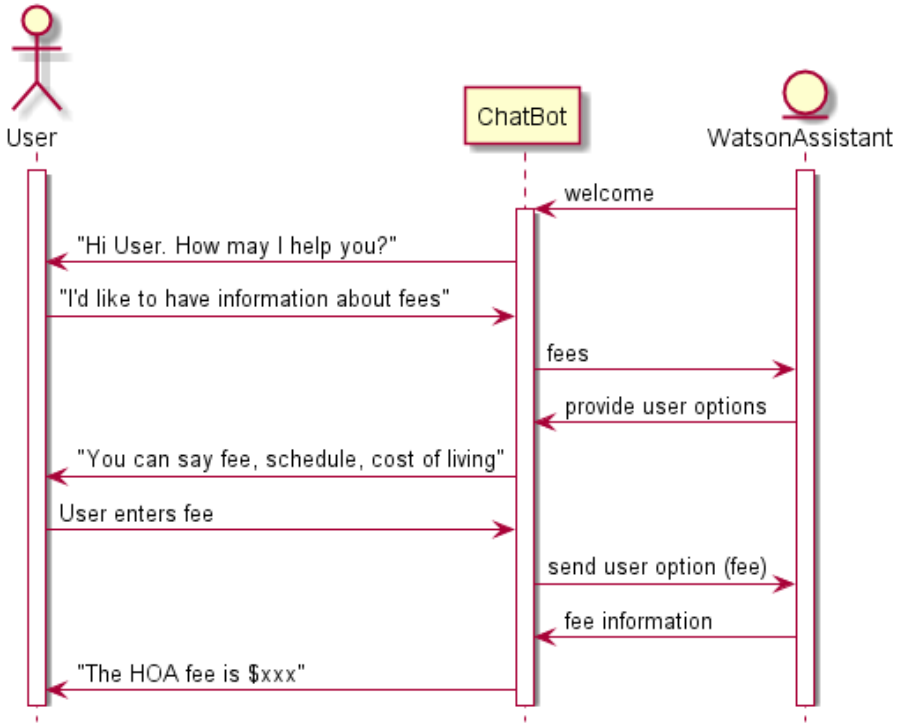
Annotation	Description
--*	Composition
--|>	Extention
--o	Aggregation
..>	Dependency
-	Private
+	Public
~	Package private
#	Package private

2.2.4 SEQUENCE DIAGRAMS

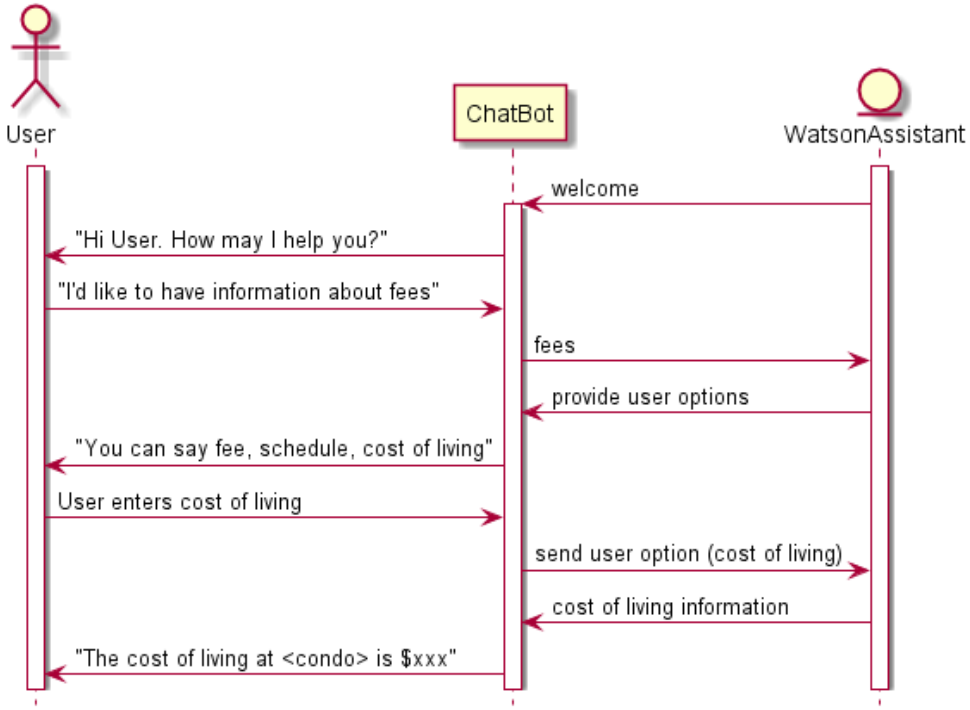


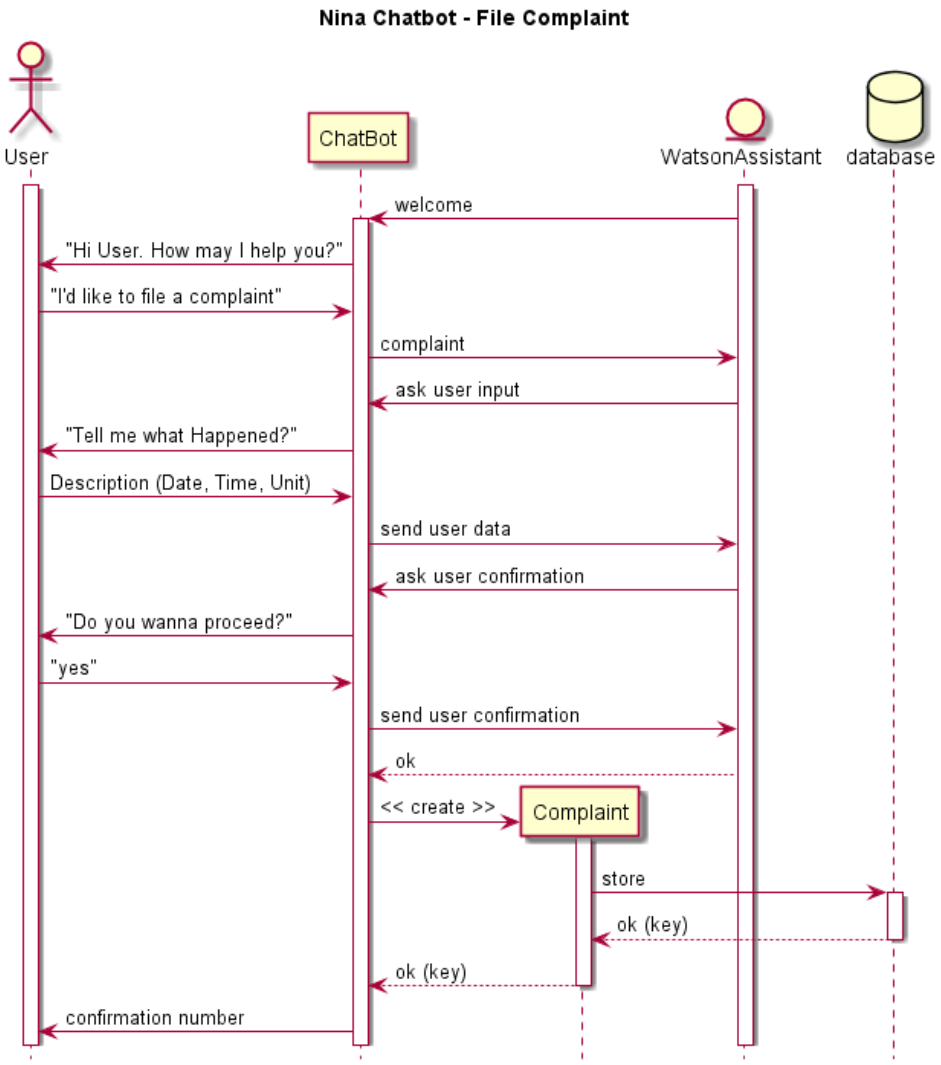


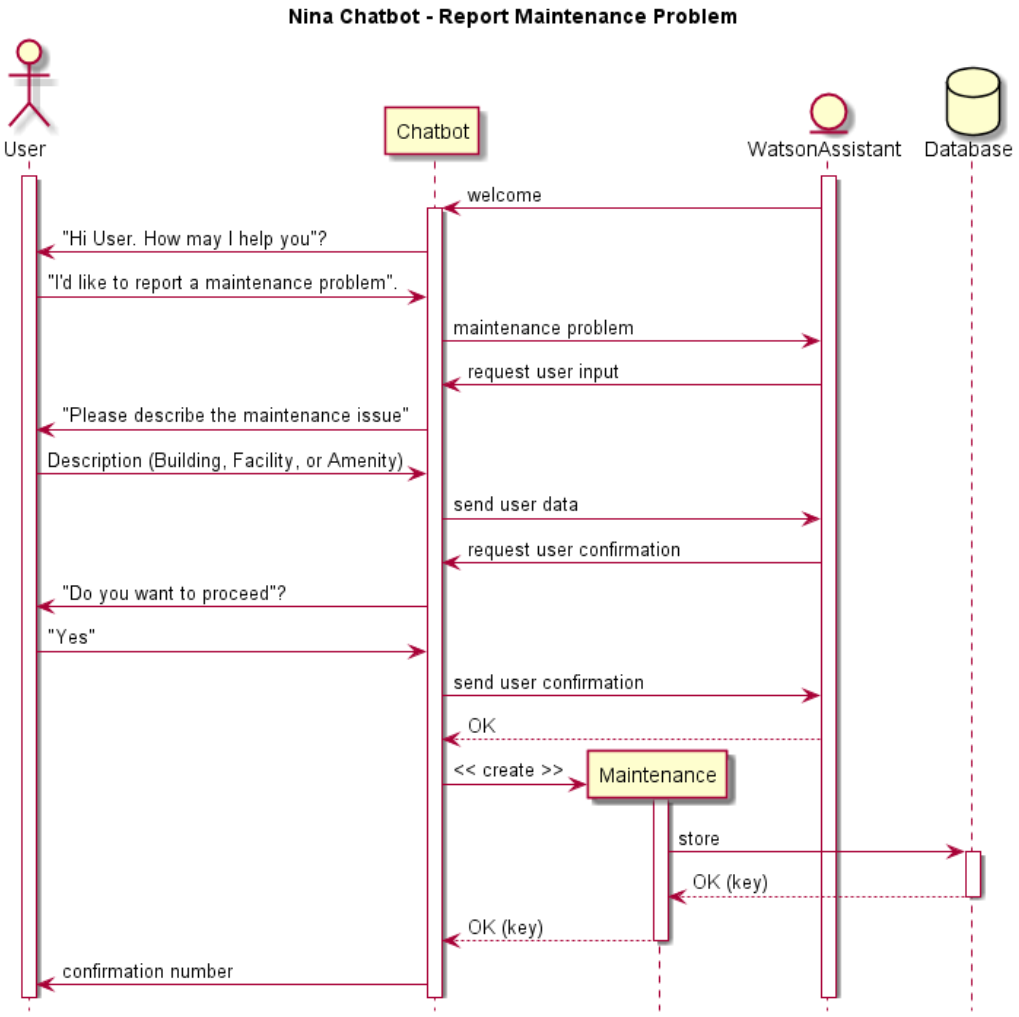
Nina Chatbot - Request Information HOA Fee



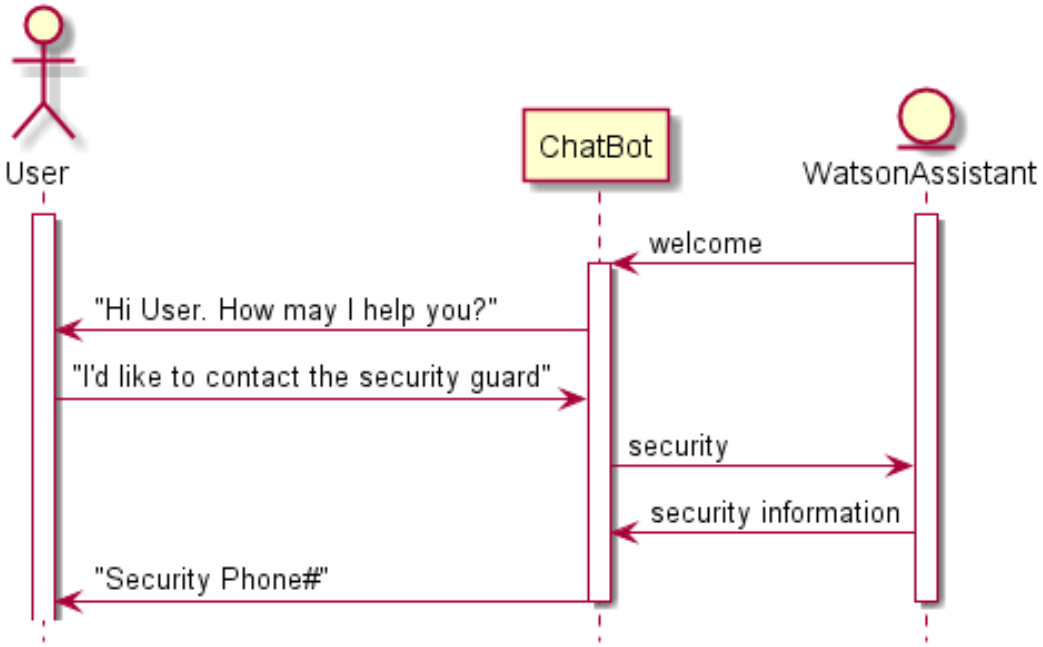
Nina Chatbot - Request Information about Cost of Living



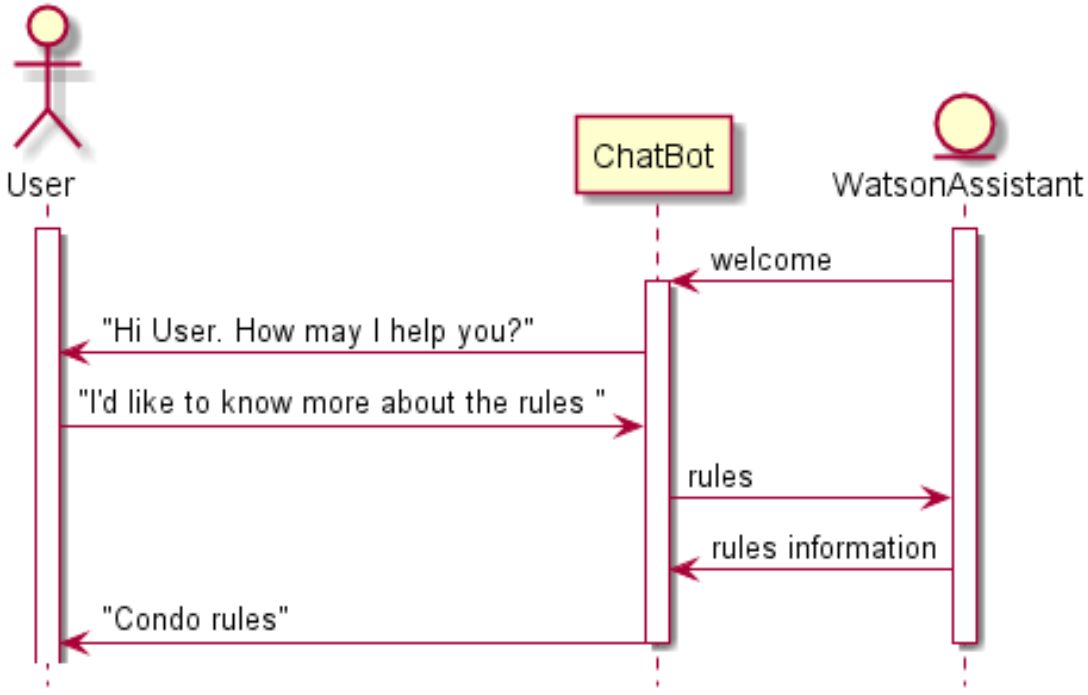

Nina Chatbot - File Complaint
User
ChatBot
WatsonAssistant
database
welcome
"Hi User. How may I help you?"
"I'd like to file a complaint"
complaint
ask user input
"Tell me what Happened?"
Description (Date, Time, Unit)
send user data
ask user confirmation
"Do you wanna proceed?"
"yes"
send user confirmation
ok
<< create >>
Complaint
store
ok (key)
ok (key)
confirmation number


Nina Chatbot - Report Maintenance Problem
User
Chatbot
WatsonAssistant
Database
welcome
"Hi User. How may I help you"?
"I'd like to report a maintenance problem".
maintenance problem
request user input
"Please describe the maintenance issue"
Description (Building, Facility, or Amenity)
send user data
request user confirmation
"Do you want to proceed"?
"Yes"
send user confirmation
OK
<< create >>
Maintenance
store
OK (key)
OK (key)
confirmation number

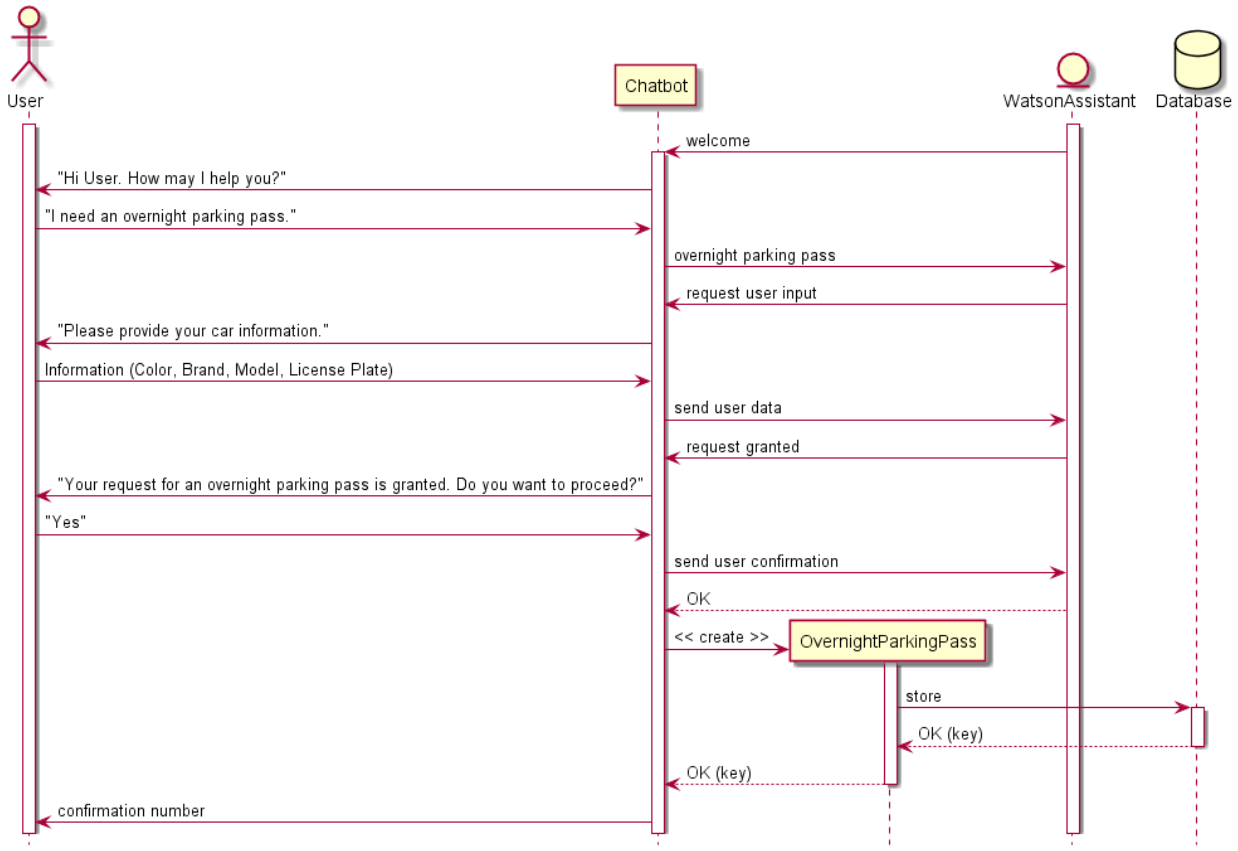
Nina Chatbot - Request Security Guard Information



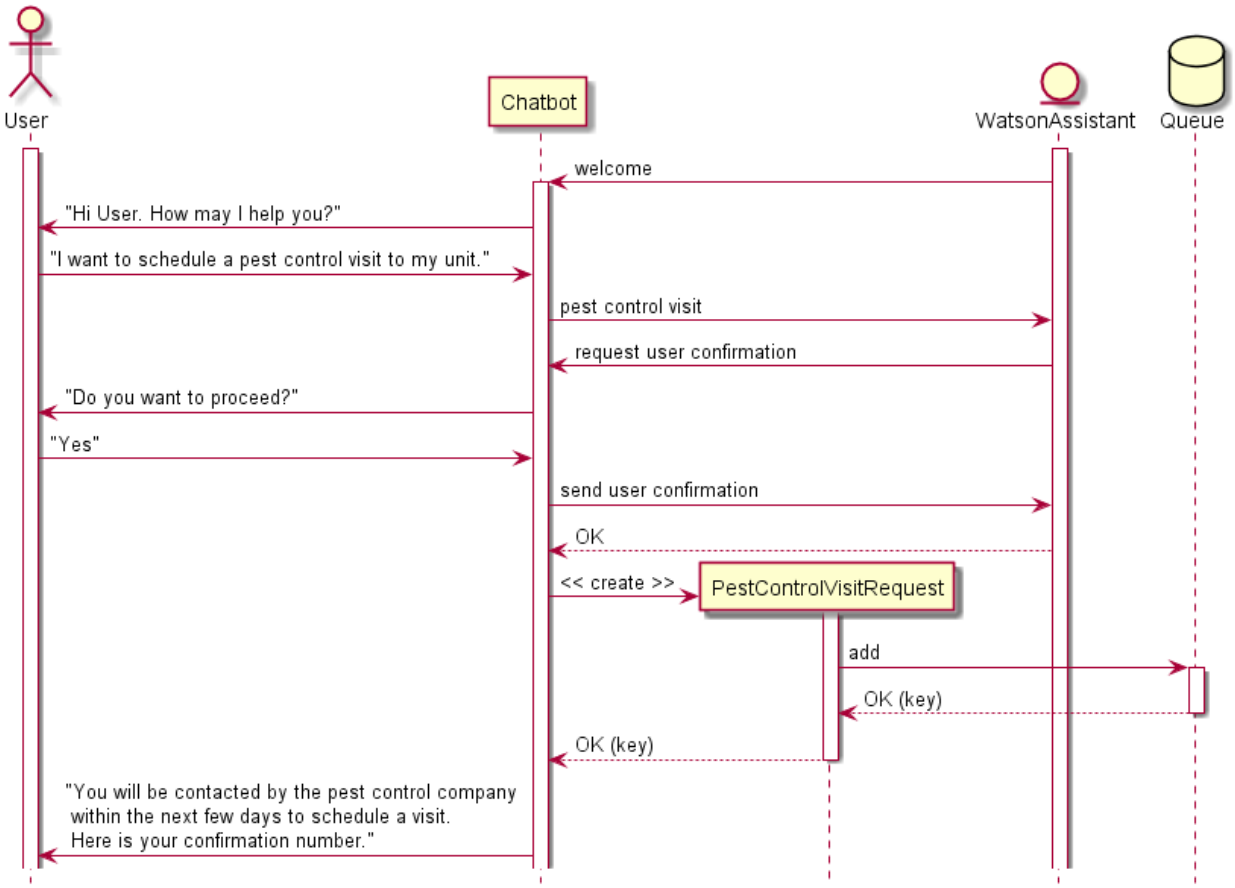
Nina Chatbot - Request Rules We Live By



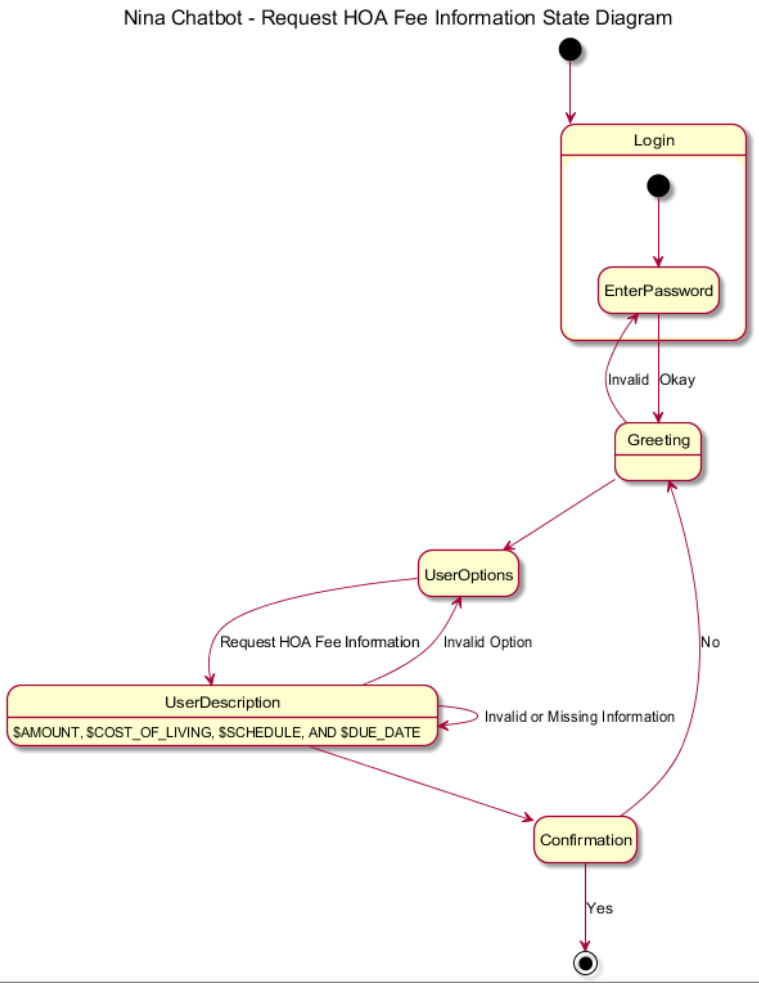
Nina Chatbot - Request Overnight Parking Pass



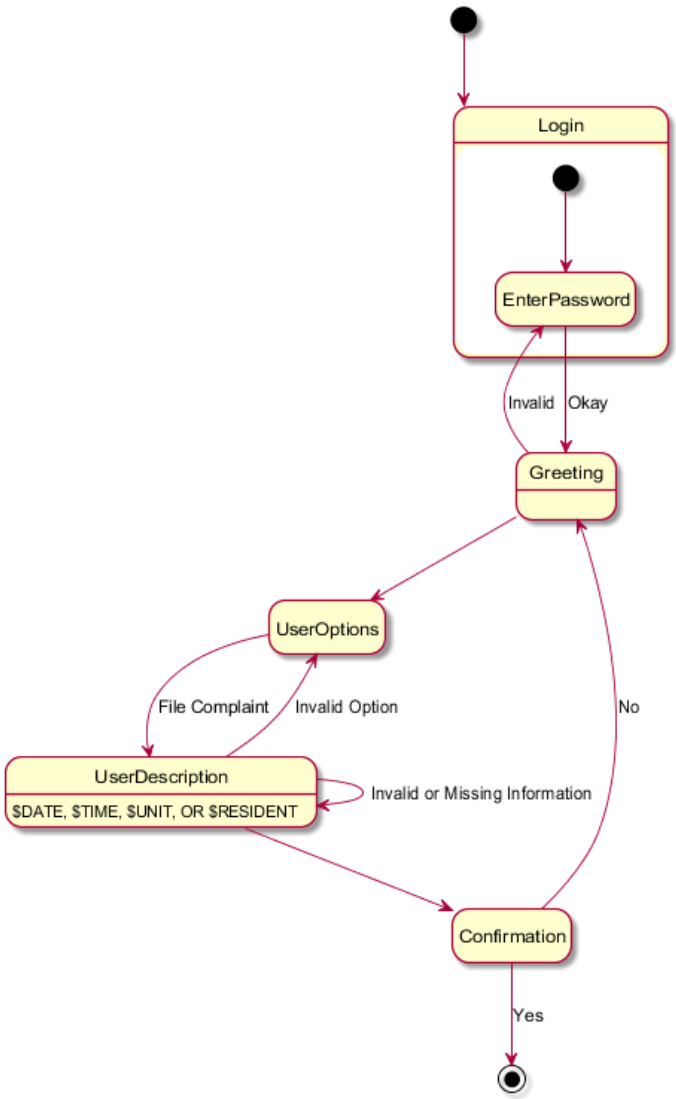
Nina Chatbot - Schedule Pest Control Visit



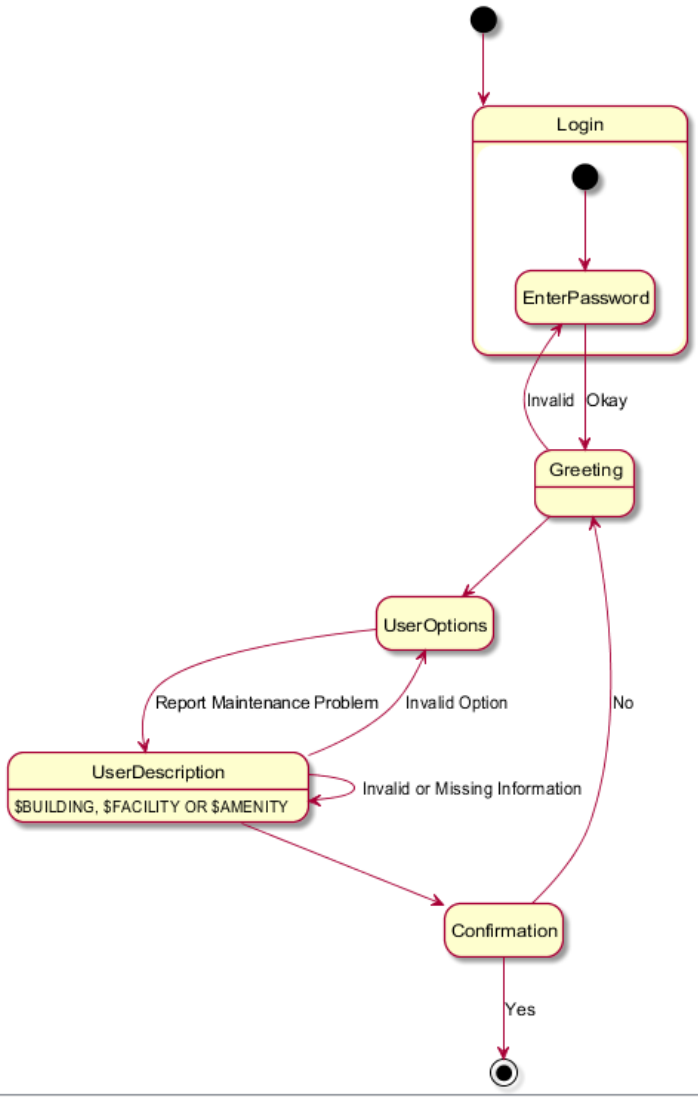
2.2.5 STATE DIAGRAMS



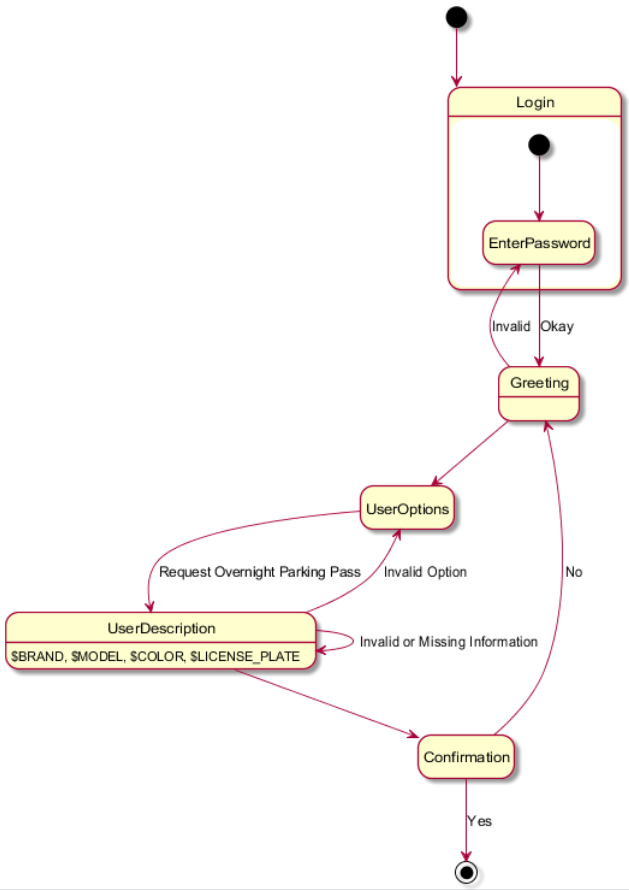
Nina Chatbot - File A Complaint State Diagram

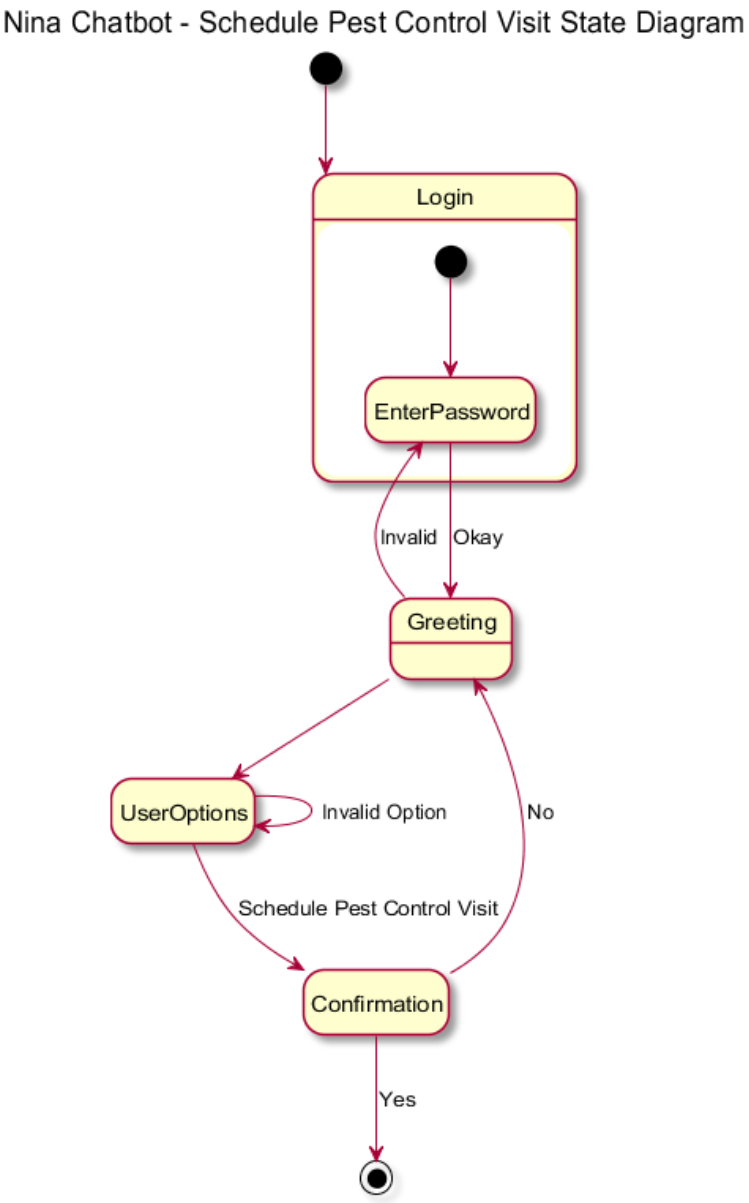


Nina Chatbot - Report Maintenance Problem State Diagram



Nina Chatbot - Request Overnight Parking Pass State Diagram





3.1 PATTERNS USED

Proxy Pattern: The proxy pattern separates the chatbot from the client by employing the Watson assistant API. It is fundamentally used to connect the users to the chatbot through the Watson assistant API. Users initiate requests and the Watson assistant API collects these requests and delivers to the chatbot. This pattern is suitable for distributed computing.

MVC Pattern: The MVC pattern describes the relationship between the data of the chatbot and the way that data is displayed to the end user. It ensures the data stored in the database and the view of the data, i.e. the way it is displayed to the user, is kept completely separate. Hence, the data can be presented in a multitude of different ways.

Decorator Pattern: The decorator pattern was also implemented to add a scroll bar to the JTextArea (dialogMonitor).

Observer Pattern: The observer pattern was implemented to Notify Watson whenever the user finishes the request

4.1 Lessons Learned

- A Third-party web service can either be a pain in the head or save our life.
- The code looks much better when programming with design patterns in mind.
- It's a better strategy to design first before coding.
- You cannot implement everything you have in your first specification.

5.1 Conclusion

- A chatbot provides quick response when no one is available
- It can also deal with an infinite number of customers at the same time
- Chatbots are the new revolution, especially for customer service, reducing the impact on humans, helping businesses, and saving a significant amount of time as well as money.

GitHub repository link - https://github.com/ProjectTeam19/NinaBotProject.git