QAD Project Requirements Document (v1)

Team Name: Team TRANSform

Project Name: QAD Virtual Assistant

Company: QAD

Lead: Andrew Luo

Scribe: Norman Chung

Members:

Troy Lee (troylee@ucsb.edu)
Ryan Siu (rsiu@ucsb.edu)
Andrew Luo (aluo@ucsb.edu)
Norman Chung (normanchung@ucsb.edu)
Simon Kim (sangbeomkim@ucsb.edu)

Intro:

QAD's ERP (Enterprise Resource Planning) platform allows users to view and manage a wide variety of data including sales orders, financial statements, and analytics.

Problems/Solutions:

The user may have trouble navigating to a certain action through the current UI. The easiest way to browse data is through a search bar, but even then it can be difficult to find a specific part of the application. Customers have to contact the support team for technical troubleshooting.

The QAD application is large in size and can be difficult to navigate for both beginners and veteran users. As an ERP, it is essential that the application enhances the user's experience by boasting efficiency and quick-use. Currently in some cases, users will find that they are forced to click through a multitude of tabs (and URLs) to reach their destination.

With the way technology continues to evolve, more and more people are beginning to use virtual assistants in place of manually typing in data. Speech-to-text in the context of the QAD application would be useful in providing free-hand functionality for the user, promoting quick usage, and creating a more flexible environment so that a user does not have to learn and follow strict guidelines.

Existing technologies:

Siri/Alexa are the biggest examples of virtual assistants that are used by consumers. These assistants have a broad range of understanding ranging from music selection to home temperature adjustments. Instead of breadth, we want to target specific actions in the QAD platform and make sure the assistant can perform those tasks properly. The general approach to NLP is three steps: speech recognition, natural language understanding, and natural language generation.

Hala.ai is an existing virtual assistant for business software. Hala can create a purchase order, add time activity, track time, create invoices, add expenses, post journal entry or make batch transactions in a blink of an eye.

Objectives:

For our use case, we want to specialize language interpretation to understand actions that can be done on the QAD platform. We will focus first on language understanding and have stretch goals of speech recognition and language generation. Most actions and views in the application can be reached through a unique URL which includes query strings. Our initial MVP will likely be a REST service that can generate these URLs. The request to this server will include the intended action in plain English, ex. “Edit sales order 011010”, and the response will include the URL that directs the user to the page for editing sales orders. The core problem is interpreting the command and going down the correct branch of the decision tree to get the corresponding URL.

Our initial prototype could just be based on this decision tree, and later we can add flexibility by introducing the NLP engine to detect dependencies like (verb, preposition, noun) and format the text to be compatible with the tree.

We are currently using the Stanford CoreNLP library for language interpretation and annotation. This library contains a pre-trained model for English as well as seven other languages. It also contains the necessary classes to run a pipeline on some input text, and returns an annotation which includes parts of speech, lemmatization, NER (named entity recognition), and several other annotations.

Goals:

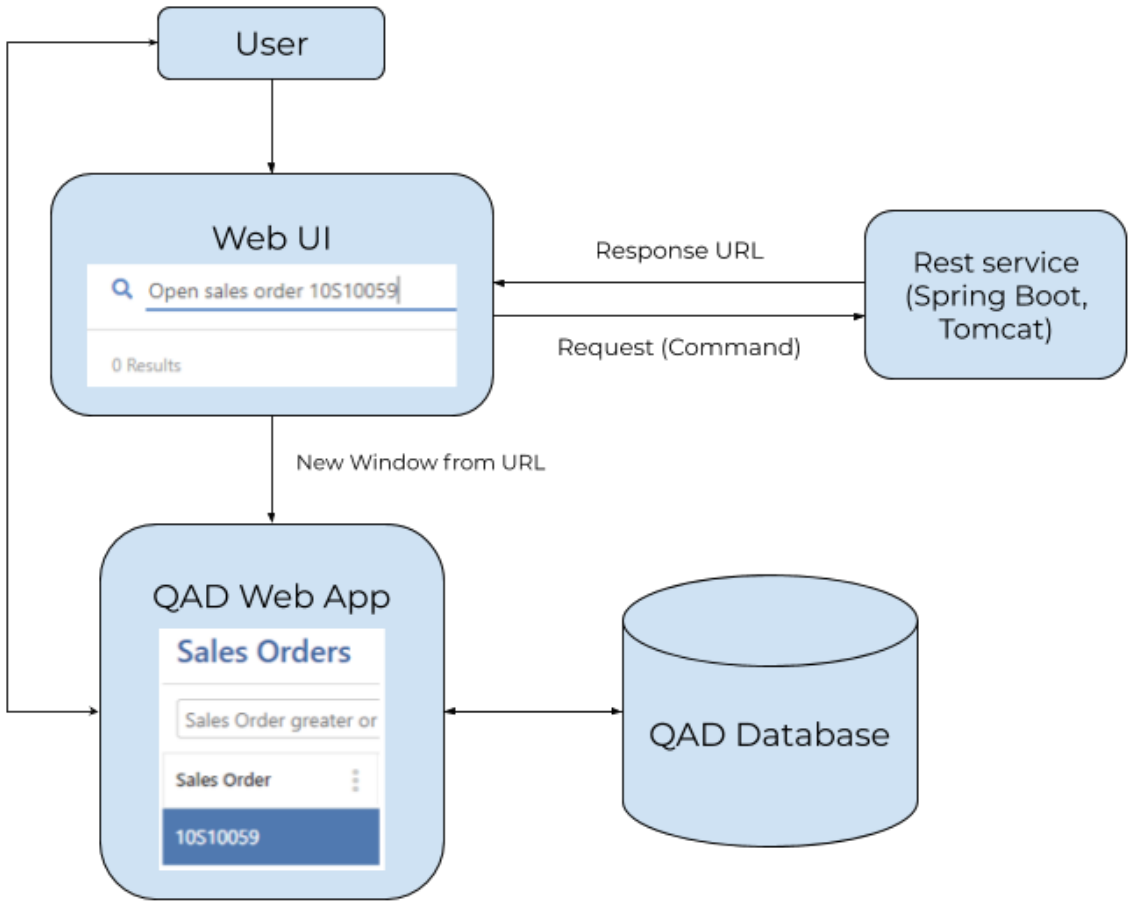
The goal of this project is to create a virtual assistant that interprets natural language commands and finds the appropriate action for the user. This assistant should work similarly to Alexa and Siri, both of which have speech-to-text capability and built-in NLP. Utilizing existing NLP resources, we hope to implement a flexible system that provides a quicker avenue for navigating the QAD application.

- Correct mapping of verb & noun of each command to unique URL
- Fast & accurate pipeline for input text
- Speech to text capability
- Proper error handling for user permissions and unknown commands
- Integration with QAD platform

Assumptions:

The main audience that this application is targeted towards are manufacturers who need to keep track of their customers, supplies, and orders. Given that our application should be dynamic, we are assuming that the user has no prior knowledge of how to use the application. In addition to the actual implementation, we are hoping to put in-place a system that is intuitive so that the user can be guided with recommended options.

System Architecture:



User Workflow:

1. Access web UI through QAD application
2. Input search query (i.e. “Open sales order 10S10059”)
    a. Current: Open sales (new tab) -> navigate to desired order (new tab)
    b. Future: Open desired order directly without going through sales tab
3. Request and receive response URL from Rest service (Spring Boot, Tomcat)
4. Open unique URL with data received from QAD database

User Stories/Cases:

User navigation of the QAD application

1. As a user, I can navigate through the QAD platform using commands.
    a. "Open sales order" would open a browse, "Open [sales order] 10S10035" would open the specific (10S10035) record within the browse.
    b. Acceptance Criteria:
        i. The correct window within the ERP application will be brought up based on the user's command

2. As a user, I can run a query through the QAD platform using commands.
    a. "Open my requisitions" would apply a filter to the browse and apply a specific stored search, "Run Sales by Customer for year 2020" would run a report.
    b. Acceptance Criteria:
        i. The correct query, with filters applied, will be brought up based on the user's command

3. As a user, I can change data within the QAD platform using commands.
    a. "Create a new [sales order] for [customer] 10C1001" would create the specific sales order, "Receive unplanned inventory" and "Ship [sales order] 10S1001" would run the specific "action" -- receive and ship, for example.
    b. Acceptance Criteria:
        i. The data that the user wants to manipulate/edit is changed correctly within the application

4. As a user, I can contact one another within the QAD platform using commands.
    a. "Email [or text or call] Gary Weinert at [customer] 10C1001" would, for instance, email, text, or call, "Directions to customer Walmart" would access directions to the specific location.
    b. Acceptance Criteria:
        i. The specified communication method wanted by the user will be opened/created for the correct recipient

QAD navigation of the QAD application

5. As a Customer Service Manager, I can keep track of the quantity of a certain inventory.
    a. In the event that stock shortages may occur, the user could say "Alert me if safety stock for item 01010 drops below 15." This would keep track of the inventory for the specific item or multiple items, and set a notification to alert the user or multiple users in the event that the quantity falls below the threshold.
    b. Acceptance Criteria:

        i. The application will keep track of and alert the user about inventories they have specified

6. As a Customer Service Manager, I can make different edits to orders.
    a. The user could say "Change ship-to to 10C1001-A on [sales order] 10S1001.". This specific command should first access sales order 10S1001, then edit the shipping address for this order to the new location, 10C1001-A.
    b. Acceptance Criteria:
        i. The specified order will have the right fields edited with the new information given by the user

7. As a Customer Service Representative, I can ask questions about inventory.
    a. "How much inventory for 01010 is available at site 10-100?" The expected result would be something along the lines of: “Item: 01010 at Site: 100 has 50.0 EA.”, by accessing the inventory list for the site 10-100, then finding the specific item, then returning the quantity.
    b. Acceptance Criteria:
        i. The correct quantity for the specified item will be returned to the user

8. As a Customer Service Representative, I can approve orders for shipping or receiving.
    a. The command “Ship [sales order] 10S1001.” would move the order 10S1001 into the shipping stage. The command “Receive [purchase order] PS1001.” would allow the order PS1001 to be confirmed, and would allow workers to begin packing the order.
    b. Acceptance Criteria:
        i. The application will approve the specified order for being shipped or received

Speech-to-text use-cases

9. As a manufacturer, I am notified that there is a dispute regarding one of the customer’s orders. I am currently involved in live-chatting with one of the company’s representatives and want to quickly navigate to the customer’s page.
    a. "Open Sales by Customer" is an example of a possible command used to open all the sales from a specific customer. Speech-to-text functionality would allow me to quickly pull up this information on another tab while I continue to maintain contact with the company representative who filed the complaint.
    b. Acceptance Criteria:

        i. The specified customer's information page will be pulled up after issuing the command

10. I am a beginner user of the QAD application but I am fully aware of what I need to do. I need to be able to create a note underneath one of the sales to clarify some special instructions. However, given the depth of the QAD application, I can't seem to find the location of that function.
    a. In plain english, I know that I want to "write a note underneath sales number [10S10059]." The QAD application will be able to interpret this sentence and recognize that a note needs to be written. Even though the statement is not a valid command in itself, the application will be able to use NLP and validate the intention of the statement.
    b. Acceptance Criteria:
        i. The user will be navigated to the "Notes" section under the specified sales order

Appendix (Technologies Employed):

Frontend: HTML/CSS
Backend: Spring Boot, Tomcat, Java
NLP: Stanford CoreNLP