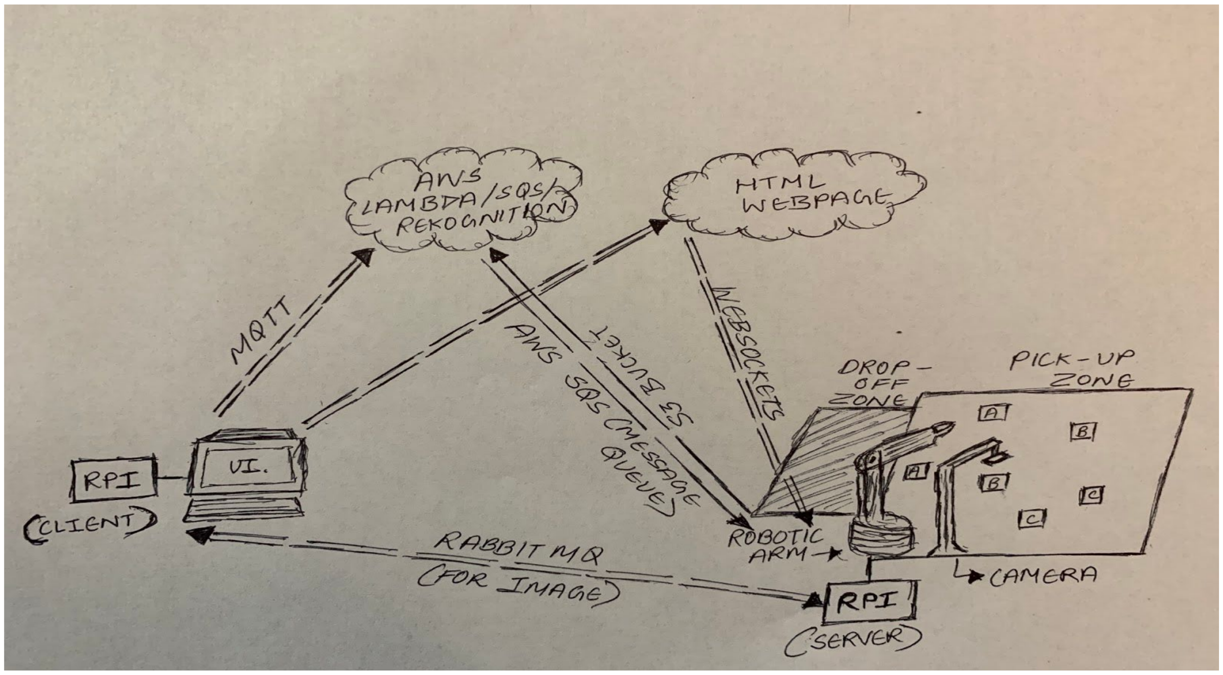
EID 5053-002
Embedded Interface Design
Super Project Proposal Updated

Project Name:
AUTOMATED WAREHOUSE SYSTEM (Using Robotic Arm and AWS Rekognition)

Team Members:
1. Nikhil Divekar.
2. Anay Gondhalekar.
3. Shubham Jaiswal.

DIagram:



Working Principle:

1. We have developed an automated warehouse system using image processing with camera and robotic arm for precise pickup.

2. Warehouse contains the boxes of different objects placed in the pick up zone(As seen in the diagram). Raspberry Pi on server side is interfaced with the camera and robotic arm. The camera is placed in such a way that it covers the entire area of interest.

3. We have used image processing to detect different blocks by character detection which are imprinted on the storage boxes. AWS Rekognition libraries are used for object/character detection. We are making use of these libraries to appropriately locate the position of the object in the pick-up zone.

4. Another Raspberry Pi serves as a client which can communicate with remote server from anywhere. There acts as a user-interface on the client side which is used for allowing the user to retrieve certain object material from the warehouse to be dropped off in the drop-off zone (as shown in the above diagram) where it is retrieved by the user.

5. So, if user wants to retrieve certain object from the warehouse, he just has to place the order through the client interface, and then the object will be dropped off at drop-off zone which will be at the entrance of the warehouse using robotic arm. By this, we have reduced the human efforts of actually entering the warehouse and manually looking for the object in question.

6. The way, this part works is user selects the object from the UI. This request is be sent to the server Raspberry Pi. Camera on the server raspberry pi captures the image of the pick-up zone and gets the coordinates of the positions of different objects.

7. Then, it triggers the robotic arm to move to the location of the object which has been requested, pick it up and then drop it off at the drop-zone.

8. We also plan to create the secure login and signup screen and affiliate that with a database so that only authorized users are allowed to retrieve the objects.

9. The feature of single-retrieve and multiple-retrieve on the UI is available.If multiple-retrieve is selected then we make use of message queueing to store information of the objects to be retrieved and begin the process or flush message queue when ‘Done’ button is pressed.

10. An additional feature that we have added is that an image of the pick-up zone(warehouse) is sent to the client from server side when an order is placed and after the order has been serviced.

Communication Protocols:

1. MQTT
2. Websockets
3. Message Queue Protocol.
4. RabbitMQ

Hardware Requirement:

1. OWI Robotic Arm
2. USB Interface for Robotic Arm.
3. Pi Camera.
4. Raspberry Pi (2).

WBS for Anay:

1. Automated Warehouse using Robotic Arm
    - 1.1. Configuring Robotic Arm   Week 1
        - 1.1.1 Research and Development of Drivers for the Robotic Arm
        - 1.1.2 Python Code for Robotic Arm Functionality
        - 1.1.3 Changing the parameters and troubleshoot pickup and dropoff
    - 1.2. Client Side Design and Code(Python)   Week 2 and 3
        - 1.2.1 Code and Design for sending request to AWS
        - 1.2.2 Code to update values of the quantities
        - 1.2.3 Code to Interface PyQt with the client side Python Code
    - 1.3. MQTT protocol for client side   Week 3 and 4
        - 1.3.1 Configure and setup MQTT
    - 1.4 Extra Credit Work   Week 5
        - 1.4.1 Getting location of object to Arm based on AWS Rekognition
        - 1.4.2 Implementing RabbitMQ for image transfer between server and client PI.

WBS for Nikhil:

1. Automated Warehouse using Robotic Arm
    - 1.1. Client interface   Week 1
        - 1.1.1 Brainstorming the ideas about the HTML side of Client.
        - 1.1.2 Writing the code-behind for all the interface buttons in the Client side.
        - 1.1.3 Interfacing it with AWS server with proper protocol
    - 1.2. Robotic Arm Troubleshooting: Week 2
        - 1.2.1 Verify the movements of robot in various scenarios.
        - 1.2.2 Optimize the pickup and drop of robotic arm.
        - 1.2.3 Fixing the issues, if any.

1.3 AWS services: Week 3

1.3.1 Getting to know AWS services and comparing between options.

1.3.2 Implementing AWS data storage and other services.

1.3.3 Creating the communication between AWS server and Client R-Pi and Server R-Pi.

1.4 SQS: Week 4

1.4.1: Understanding and implementing SQS for message queueing.

1.4.2 Lambda Function.

1.5 Extra Credit: Week 5

1.5.1 Implementing the object detection using AWS Rekognition and getting the exact location of particular element and functioning robot arm according to the same

WBS for Shubham:

1. Automated Warehouse using Robotic Arm

1.1 Assembly of Robotic Arm: Week 1

1.1.1 Assemble the Robotic Arm according to the manual provided.

1.1.2 Connect it to wired control box and check whether the range of motion, rotation and movements are according to the specifications.

1.2 PyQT: Week 2

1.2.1 Brainstorming the ideas about the User Interface of Server side.

1.2.2 Developing the Server side User Interface.

1.3 Server Code: Week 3

1.3.1 Python code for populating the UI elements with the data collected.

1.3.2 Code for getting requests from AWS.

1.3.3 Code to send the appropriate data to the client Application.

1.4 Websockets: Week 4

1.4.1 Implementing websockets for bi-directional communication between Server and Client.

1.5 Extra Credit: Week 5

1.5.1 Text detection and getting Robotic Arm to function accordingly.

Changes From Last Proposal:

1. We have removed the feature of counting the number of objects present in the warehouse and also we are not keeping the count of each object present and not producing alerts if an ordered object is not detected in the warehouse.

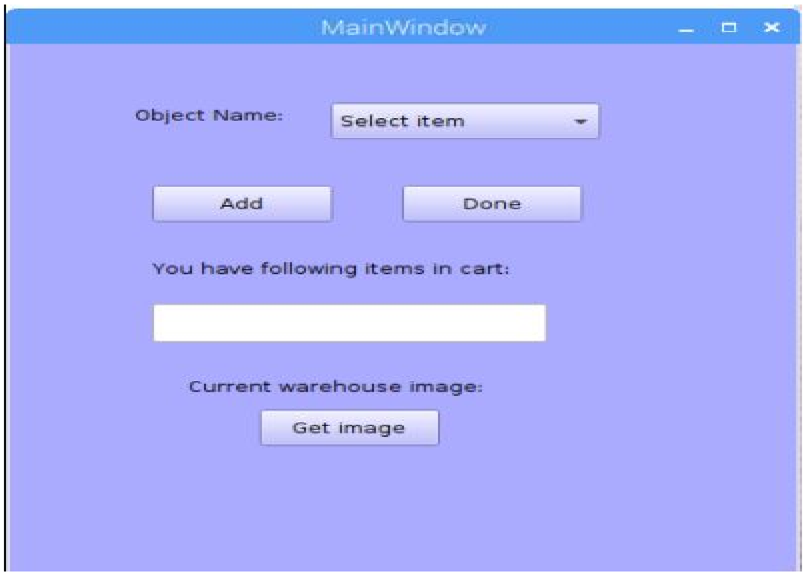
2. An additional feature that we have added is that an image of the pick-up zone(warehouse) is sent to the client from server side when an order is placed and after the order has been serviced.

Issues and changes encountered:

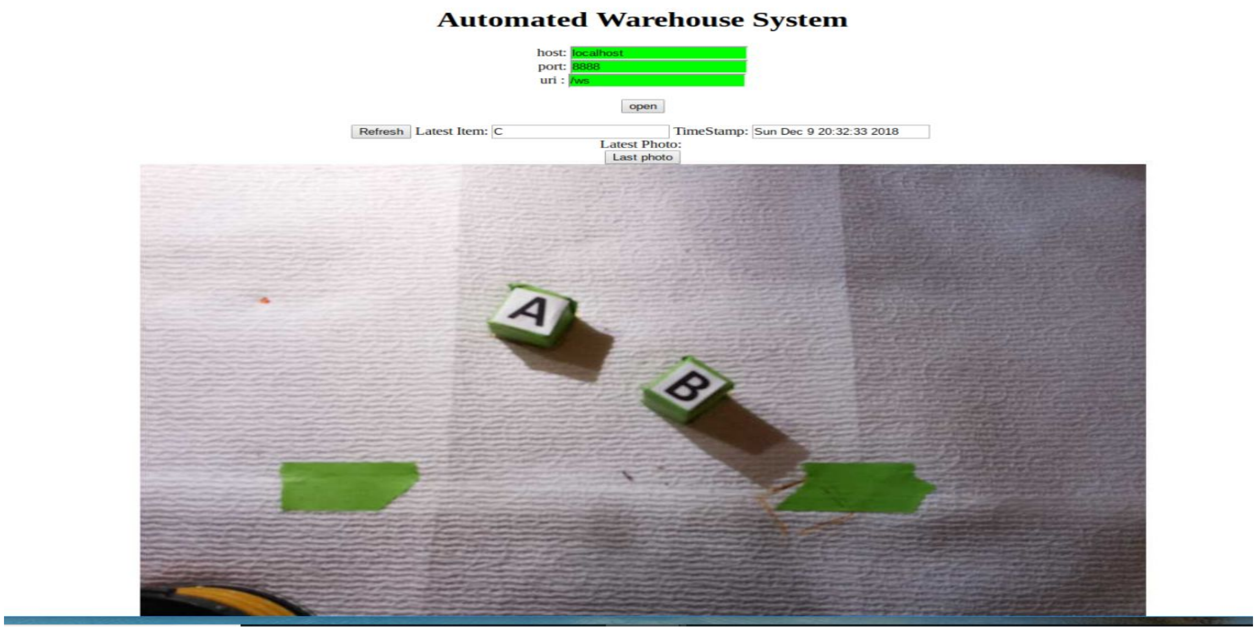
1. We particularly encountered problem while calibrating the Robotic Arm, getting values on html webpage from database and understanding the various services offered by AWS and getting to know how to make use of those services.

User Interface Screenshots:

1. **PyQT:**



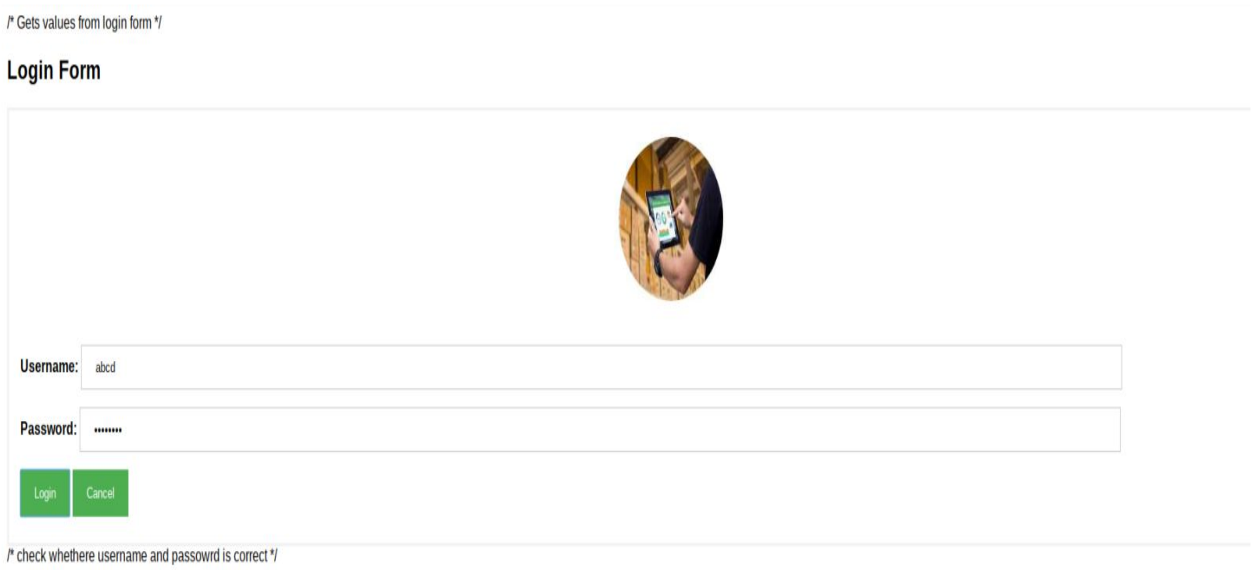
2. HTML:


Automated Warehouse System

host: localhost
port: 8888
uri : /ws

Latest Item: C TimeStamp: Sun Dec 9 20:32:33 2018
Latest Photo:

3.Login Page:



/* Gets values from login form */

Login Form

Username: abcd

Password: ••••••••

/* check whethere username and passowrd is correct */