University of Colorado
Boulder

MAGIC WAND

BY

SIDDHANT JAJOO
SATYA MEHTA
VATSAL SHETH

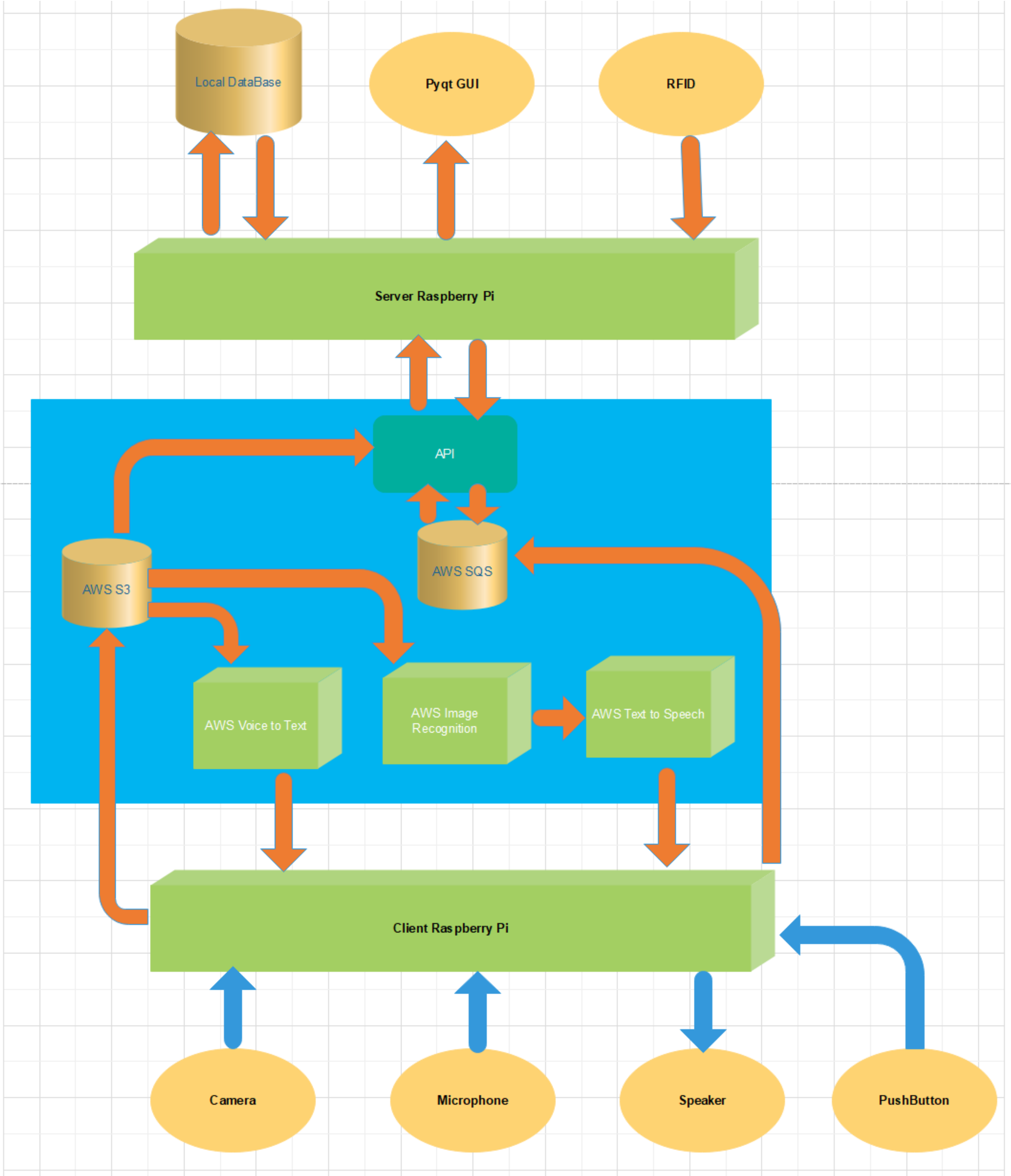
UNDER THE GUIDANCE OF:

PROFESSOR BRUCE MONTGOMERY
UNIVERSITY OF COLORADO BOULDER

12/12/2019

TABLE OF CONTENTS

ARCHITECTURE DIAGRAM



System Architecture is updated with addition of push button which is used by client to start audio capture. Tasks on Server Pi is divided between Python and NodeJs. Reading AWS SQS and updating it in SQL is implemented on NodeJS whereas Python runs GUI, downloads image from AWS S3 and read from SQL.

PROJECT DEVIATION SYSTEM

There have been no major modifications from the Project Update 4 and 5.

- One deviation was inclusion of GPIO interrupts in the client side to trigger the recording of the voice commands. We thought of using real-time speech to text conversion which would have not required any GPIO interrupts but that got little out of scope for us.
- We were unaware of the compulsion to have nodejs in the project. So, we initially decided to code everything in Python, then changed the implementation to include Nodejs to update MySQL database.
- For AWS Lex and AWS Recognition to work we are required to upload the audio and image file to AWS bucket. These services then use data from the bucket to process on.

THIRD PARTY CODE USED STATEMENT

No, third party code was used in order to implement the project code other than the standard AWS, Python and Nodejs libraries and examples from the official websites.

PROJECT OBSERVATION STATEMENT

- The NodeJs code to upload data to the mysql database with the statistics values could have been written more efficiently as compared to what is written now. This inefficiency might be attributed to the fact that this was the first time a lot of new nodejs functions were used in conjunction with AWS services.
- Speech to text in AWS was done by using AWS Lex which was required to be configured for voice. We can also use AWS real-time transcribe service which can provide real-time conversion making the application more efficient.
- Switching between windows in PyQt5 was challenging. RFID interface could be improved by continuously reading it in different thread would have removed the need for button press and thus, improve user interface. Also, we can improve GUI to displays password in hidden format with * instead of keeping it visible.