# Meal By Words

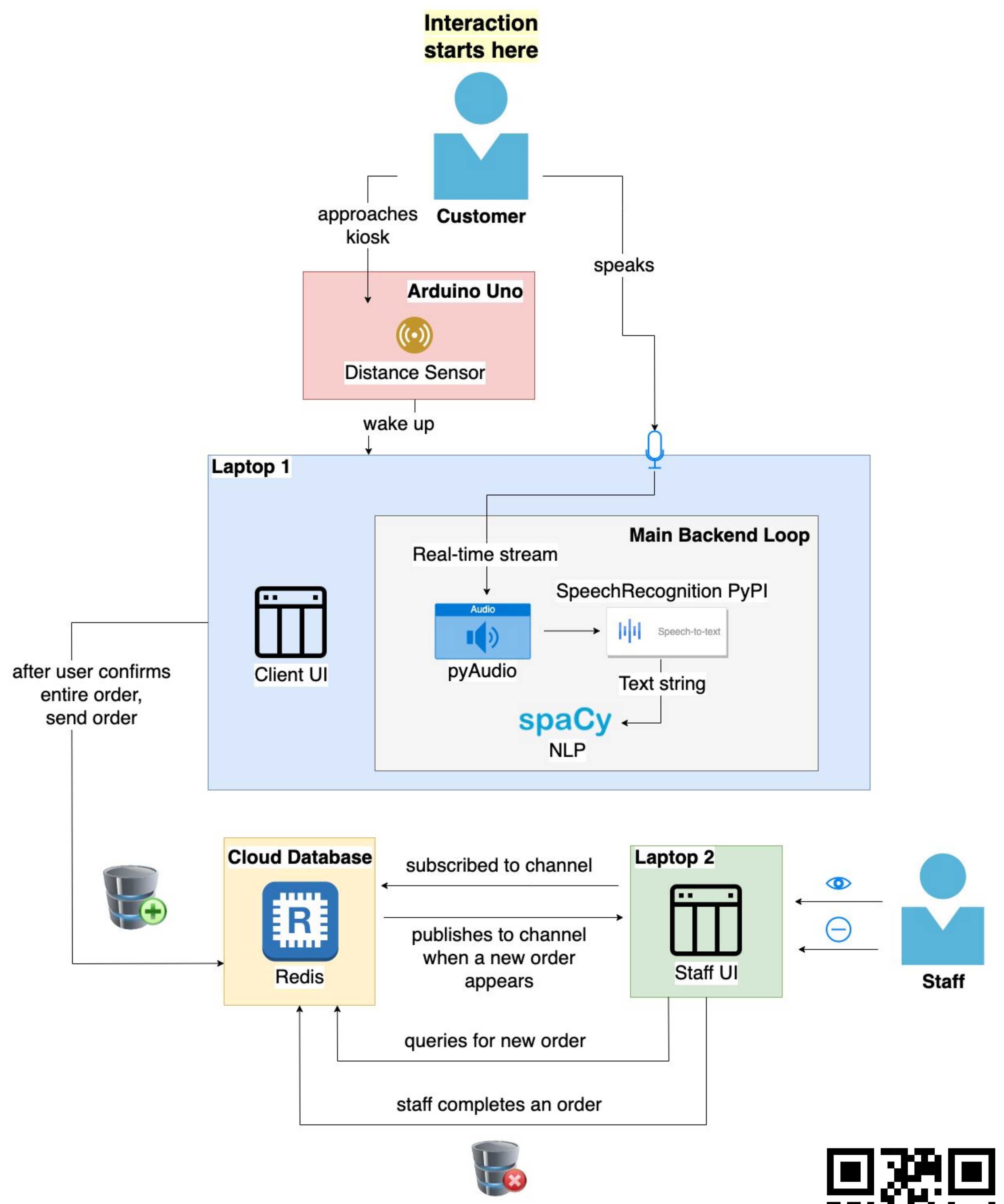
**D3: Nina Duan, Lisa Xiong, Shiyi Zhang**
18-500 Capstone Design, Spring 2023
Electrical and Computer Engineering Department
Carnegie Mellon University

## Product Pitch

Meal By Words is a system that allows customers to verbally place an order at a restaurant. Inaccurate orders and long wait times are critical problems that discourage customers from ordering at quick-service restaurants. With the traditional ordering method, customers frequently have to raise their voices or even shout to ensure that the person taking their order can hear and understand it accurately. Some restaurants provide touch screens as an alternative which has the potential of spreading harmful bacteria and diseases. To tackle these problems, we propose the development of a food ordering system that employs speech processing technology to facilitate the process of placing orders at restaurants.
**We successfully implemented full functionality for ordering, allowing customers to add and delete any item entry. The speech recognition and parsing subsystems are able to recognize all menu items and quantities in the user's speech input, regardless of sentence structure. For customer-side and staff-side interaction, we barely missed our latency requirement of 1s but were able to guarantee that 100% of the orders are successfully delivered.**

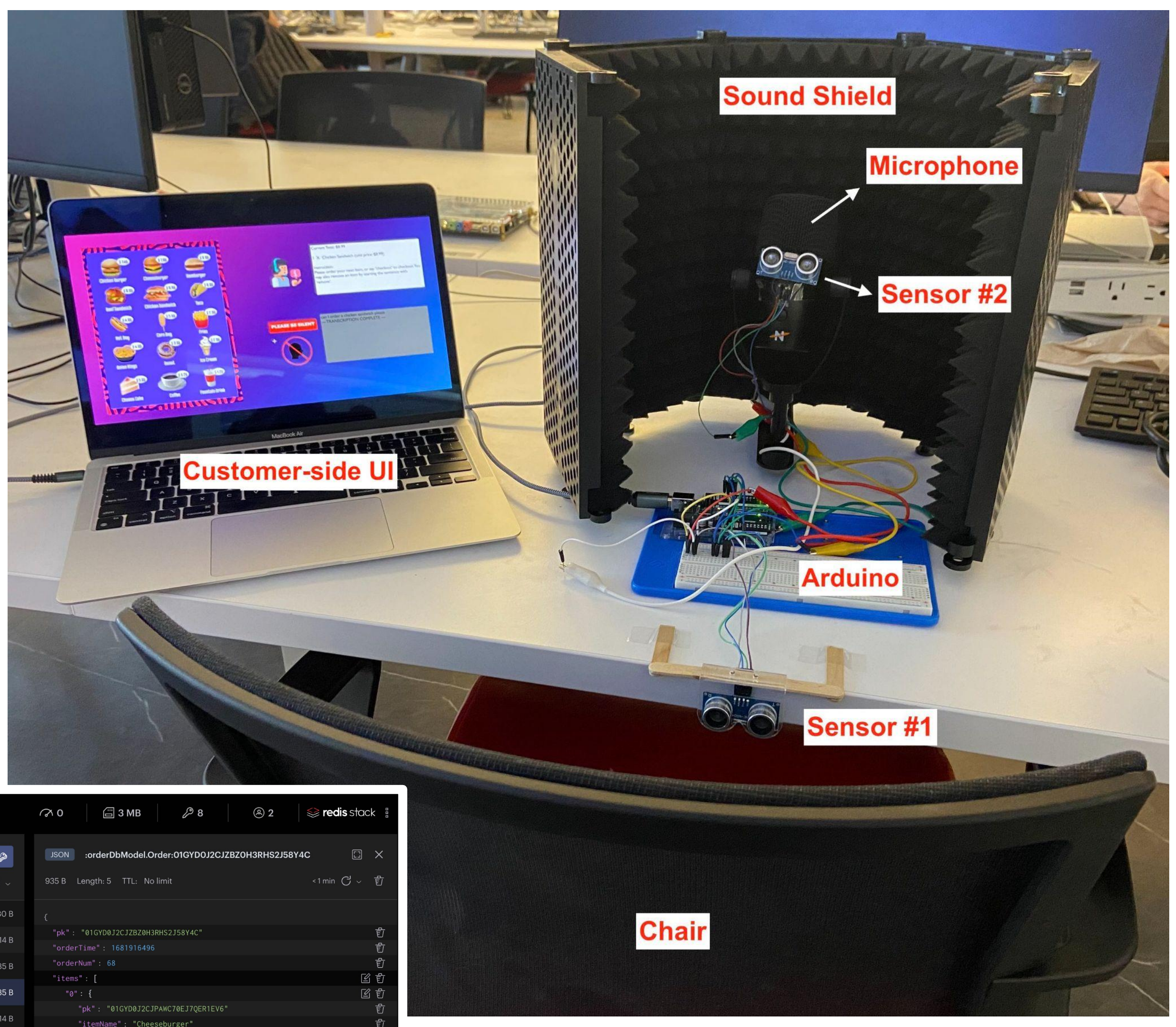
## System Architecture

In absence of a customer, the system will be in SLEEP mode. An ultrasonic distance sensor, driven by an Arduino Uno, wakes up our system when a customer sits down on the chair placed in front of the kiosk. At the same time, the monitor screen will display the customer-side UI. A directional microphone will receive speech inputs from the customer. Our natural language processing algorithm parses the text string and extracts necessary order information.To check out, our system will instruct the customer to review their order. The customer can confirm the order by saying "yes", "correct", or "confirmed".
After the customer checks out, the entire order, previously stored as a local object, will be uploaded to the Redis cloud database. Information in this database is accessible to the staff-side UI. The database notifies the staff-side UI of the new order by publishing a notification message. After 10 seconds, the customer-side UI will go into SLEEP mode, waiting for the next customer.
The staff-side UI, used by the restaurant's kitchen staff, queries the database for new order information. Staff can remove items from an order after they're done with preparation. When an item entry has been removed, it disappears from the screen and gets deleted from the database.

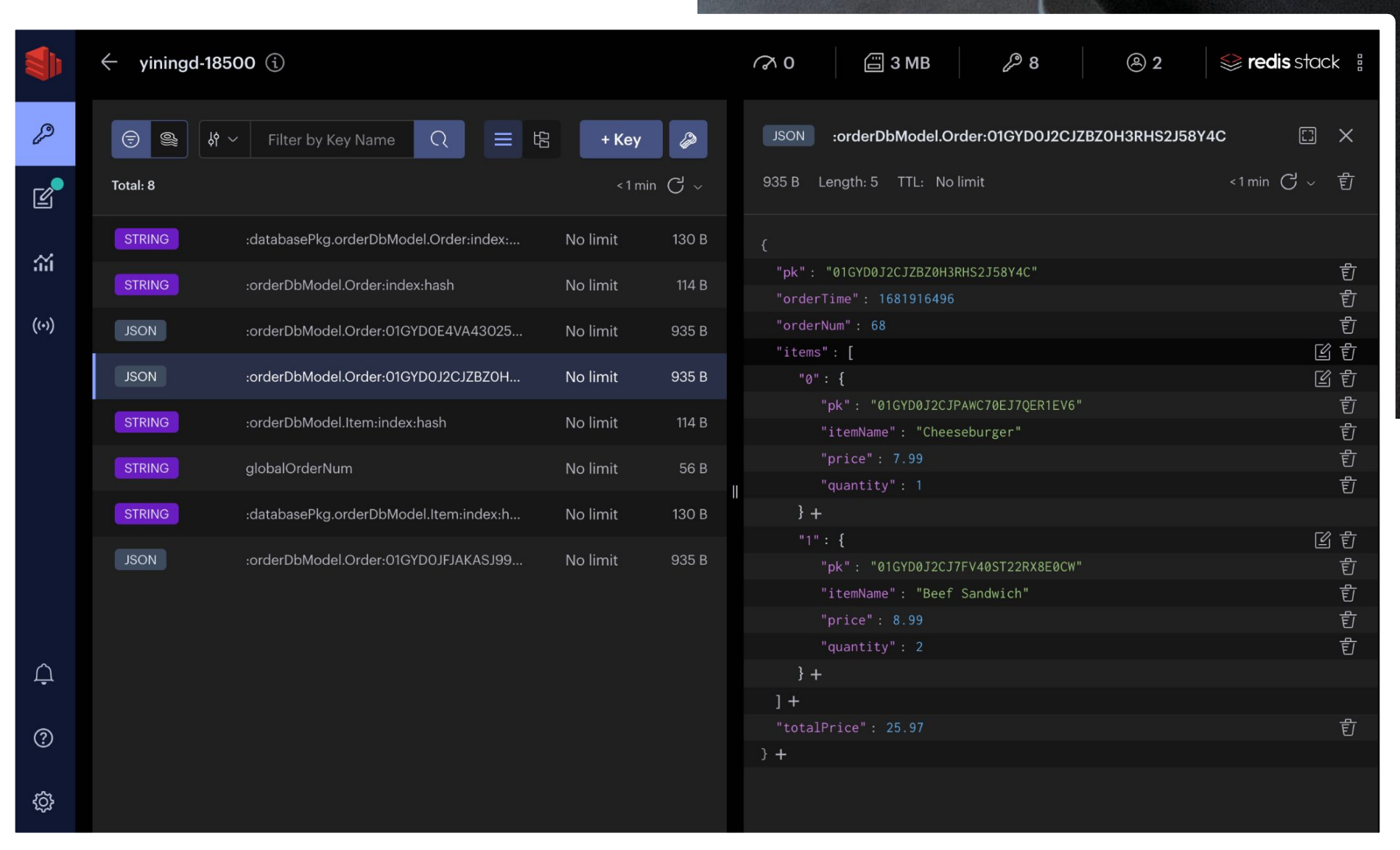


http://course.ece.cmu.edu/~ece500/projects/s23-teamd3/

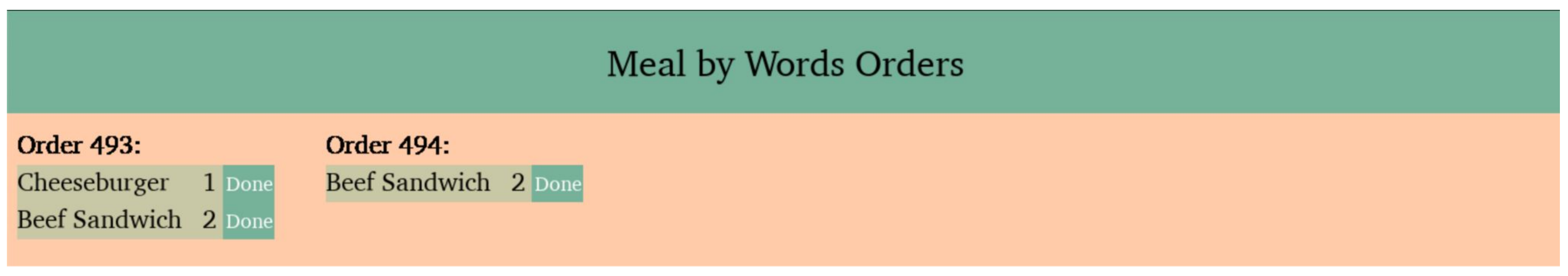
## System Diagrams



Checked out orders are uploaded to cloud database.



Staff-side UI queries the database for order information.



Meal by Words Orders

**Order 493:**
Cheeseburger 1 Done
Beef Sandwich 2 Done

**Order 494:**
Beef Sandwich 2 Done

## System Evaluation

System Tests

| Test Name | Test Description | Ideal | Actual |
|---|---|---|---|
| Audio-to-Text Accuracy | Tests the speech recognitions subsystem's ability to correctly recognize common phrases. | 85% | 87.9% |
| Text-to-Command Accuracy | Tests the natural language processing subsystem's ability to parse a string into actionable commands. | 95% | 96% |
| Order Upload Latency | The time between checkout and when the order appears on staff UI | 1s | 1.021s |
| Order Upload Accuracy | Tests that the connection between customer-side and staff-side is sound. | 100% | 100% |
| Kiosk Activation Latency | Kiosk wake-up time (SLEEP → ACTIVE) | 2s | 1.42s |
| Customer Detection Accuracy | Tests the ability to recognize the presence/absence of a customer | 100% | 100% |

Design Trade-offs

| Say all items in one go | | Say items one by one | |
|---|---|---|---|
| Feels natural to customers | Hard to determine end of speech | Increases speech detection accuracy | Longer service time per customer |
| | | | |
| **Upload entire order at checkout** | | **Upload item to database when detected** | |
| Less database accesses | Risk of losing an entire order | All order data saved ASAP | Repeated DB accesses increases latency |
| **Busy waiting loop** | | **React to trigger events** | |
| Intuitive | Waste of resources Fixed sleep time interval | Saves CPU power Dynamic wake-ups | Hard to terminate background events |

## Conclusions & Additional Information

With our speech-ordering kiosk, the fast food restaurant staff can now shift their primary focus to food preparation, without worrying about serving the customers coming in to order at any moment. The customers will no longer be concerned about touching a kiosk with potentially harmful bacteria attached to it. One improvement we could make is, instead of relying on the speech recognition libraries to determine the end of a speech, writing our own script for that purpose using a language that runs faster than Python, such as C. Another potential improvement of our system is allowing menu customization. When the staff updates the menu, the system should automatically accommodate for the changes without requiring software engineers to manually modify the code for speech processing, the database, and the UIs.