EE/CprE/SE 491 Weekly Report 2

9/31/19 - 10/10/19

Group Number sdmay20-23

Project Title: Multi-Context Shopping Optimization

Client & Advisor: Goce Trajcevski, Ashfaq Khokhar

Team Members/Role: Max Garton, Arnoldo Montoya-Gamez, Ethan Shoemaker, Karla Montoya, Jesrik Gomez, Nate Wernimont (team member titles TBD)

Weekly Summary

This week, the team did more research and prototyping. Because there are many components, different team members focused on different parts of the project. Some team members focused on Android notifications, finding Android location, and network connectivity on Android. Other team members focused on researching and analyzing the pros and cons of our sensors. Also, because we plan on using a raspberry pi, one of our team members used his personal raspberry pi to prototype on. Additionally, we thought about and researched the best ways to create our database, and created a basic database schema. Lastly, we started setting up the prototype of our initial microservice architecture with Docker. Eventually, the microservices will be deployed onto a cloud-computing platform.

Past Week Accomplishments

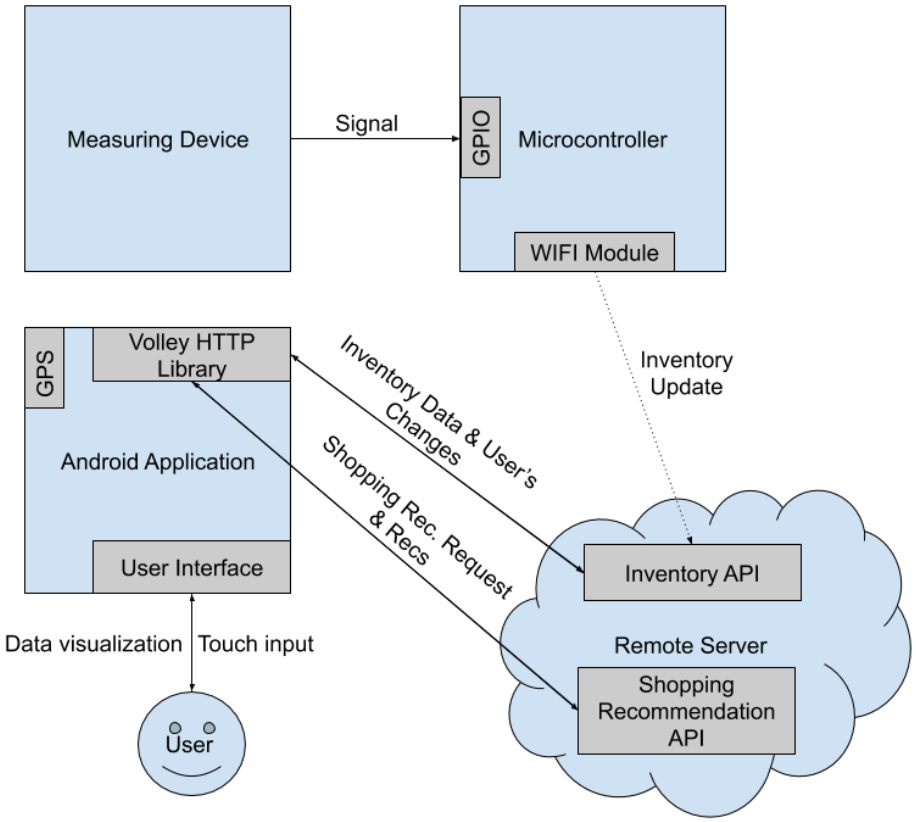

Figure 1: System component diagram for the project

Accomplishments during the week of 9/31-10/10:

- Researched and prototyped Android networking, notifications, and location (Arnoldo)
- Created basic Database schema (Nate)
- Started working on raspberry pi, and made some progress on sensor data manipulation designs (Max and Ethan)
- Created an elevator pitch for our project (Ethan)
- Start prototype of microservice architecture (Jesrik)
- Created system component diagram (Max)

Pending Issues

- Max Garton: N/A
- Arnoldo Montoya-Gamez: N/A

- Ethan Shoemaker: N/A
- Karla Montoya: N/A
- Jesrik Gomez: N/A
- Nate Wernimont: N/A

Individual Contributions

Arnoldo:

- Researched Networking APIs to make http requests. Found a lot of different ways to make http requests in the following links.
    - https://stackoverflow.com/questions/3505930/make-an-http-request-with-android
    - https://developer.android.com/training/volley
- Location was also researched, and I have a previous project I worked on where we needed location, and did it in the following way.
    - https://github.com/AMontoyaGamez/watchdog-safety-app/blob/arnoldo/app/src/main/java/com/sac/watchdog/LocationActivity.java
    - https://codelabs.developers.google.com/codelabs/advanced-android-training-device-location/index.html?index=..%2F..advanced-android-training#0
    - https://google-developer-training.github.io/android-developer-advanced-course-concepts/unit-4-add-geo-features-to-your-apps/lesson-7-location/7-1-c-location-services/7-1-c-location-services.html

Ethan:

- Drafted an elevator pitch to a lay-person
    - Received feedback on initial draft of pitch
- Drafted data collection heuristic
    - Sensors constantly send data to the sensor hub for processing
    - Options for sensor hub processing
        - Sends raw data
        - Transforms raw data into “human readable” format
- Compiled spreadsheet containing different sensor options including type, price, form factor, flexibility, and ease of use
- Came up with possible list of metrics to record from each sensor/home
    - Value
    - Timestamp
    - Location (including hierarchy)
    - Update type

Jesrik:

- Researched different API architectures and design patterns for our inventory management and request handling microservices
    - REST
        - RESTful services
        - Middleware

        - Real-time ⇒ Useful for updating inventory
    - Service layer
- Researched how to maintain data consistency across multiple services
- Set up Docker and started prototyping a simple NodeJS server

Karla:

- **Did additional Android research on UI and notifications**
    - Researched how to create basic notification that can be sent to our mobile app.
    - Read about UI designs.
    - Brainstormed functionality needed for users.
    - Thought about screen sketches.
    - https://www.interaction-design.org/literature/topics/ui-design

Max:

- Developed rough prototype diagrams of sensor module (raspberry pi and weight sensor)
    - Sensor modules connect to WiFi and send data directly to the cloud server
    - Within the sensor module, a raspberry pi interfaces with a weight sensor via GPIO
    - The raspberry pi (either the full Raspberry Pi board or the smaller Raspberry Pi Zero “W”) runs code that continually monitors the signal from the weight sensor
    - Data is only sent over wifi when there has been a change in the measurement (avoids constantly sending the same value to the server)
- Began experimenting with raspberry pi to find efficient ways for the team to develop and test the sensor module
    - Installed Raspbian “Buster” OS onto a raspberry pi
    - Connected to the microcontroller via SSH from an external network (using port forwarding). This means that we can develop on the microcontroller from anywhere via the internet.
    - Starting looking into automated Git hooks to automatically pull branch changes and rebuild the program
- Created system component diagram for the project (attached under “Past Week Accomplishments”)

Nate:

- Drafted the Database Schema
    - Using fairly simple MySQL tables
    - Sought feedback from members working on physical device
- Thought about how to develop the routing algorithm some more
    - There will be some aspects of a set covering problem because we need to go to the stores that have everything on the list
    - There might be a way to weight the sets in the set covering problem in order to reflect the differences in pricing
        - The issue with this is that not every single item needs to be bought from a single store

- In our first iteration, it might be possible to have a store's set of items consist of only the items where that specific store is the *cheapest* for each of those items. Then, a set cover of these stores would yield the cheapest possible set of purchasing. At that point, it would be a matter of weighting the store sets by distance
- Also possible to of the collection of sets consist of sets of strictly size 1. Each subset would be a single item from a single store, weighted by a combination of the price of the item and the distance of the store.
    - This would make it possible to get just one item from a single store, which isn't practical.

Name	Individual Contributions (Quick list of contributions. This should be short.)	Hours This Week	Cumulative Hours
Arnoldo	Created Lightning Talk Video, Researched Matching Algorithms.	7	17
Ethan	Drafted Elevator Pitch, Drafted Data Collection Heuristic	10	20
Jesrik	Researched API architectures and design patterns, started setting up Docker container	6	15
Karla	Researched Matching Algorithms, Researched ways to store data	8	16
Max	Microcontroller research, high level arch. diagram, updated gitlab board	12	22
Nate	Researched Routing algorithms	6	17

Plans for Next Week

- **Arnoldo**: Continue Researching communication between server and Android mobile device.
- **Ethan**: Order and test weight sensors for precision. Look into libraries/ways to encrypt communication channels between RPi and server.
- **Nate:** Develop some pseudocode for various approaches to the routing algorithm.
- **Max:** Continue experimenting with the raspberry pi and write some prototype programs to read from the GPIO and send data to a server.
- **Karla:** Start working on the screen flow and screen sketches for the Android application.
- **Jesrik:** Continue setting up prototype API and containerizing it. Discuss schema design with Nate.

Summary of Weekly Advisor Meeting

During our meeting with Goce on 10/3, we planned for the initial design document and system diagram. We discussed the need to state assumptions about the problem within our document. For example, we assume that the data we use from different stores will be a certain type (inventory, price, and quantity). Our implementation needs to be based on the assumptions we made, but also needs to be easily adapted to different data types if the assumption was incorrect.