|  |
| --- |
| 서주완, 김경선, 박준수, 정지석  2023-12-23 |

|  |
| --- |
| 캡스톤디자인2 - 빅히어로 |
| 칵테일 제조 머신 |
| YC Bar |

목 차

1. 프로젝트 주제 소개………………………………………………………………………………………………………………….2

1.1. 프로젝트 주제

1.2. 프로젝트 주제 선정 배경

1.3. 프로젝트 목표

1.4. 프로젝트 기대 효과

2. 프로젝트 세부 사항…………………………………………………………………………………………………………………3

2.1. 프로젝트 요구 사항

2.2. 프로젝트 요구 사항에 따른 세부 기능

3. 프로젝트 진행 사항…………………………………………………………………………………………………………………4

3.1. 소프트웨어공학 적용 및 형상 관리

3.2. 사용 기술

3.3. 역할 분담

3.4. 진행 일정

4. 개발 과정……………………………………………………………………………………………………………………………………6

4.1. 소프트웨어 개발 과정

4.2. 하드웨어 개발 과정

4.2. AI 개발 과정

5. 결과……………………...……………………………………………………………………………………………………………………10

5.1. 프로젝트 결과

5.2. 프로젝트 개선점

1. 프로젝트 주제 소개

1.1. 프로젝트 주제

안드로이드 어플리케이션으로 칵테일을 주문 시 사용자의 개입 없이 자동으로 주문한 칵테일을 만들어주는 칵테일 제조 머신

1.2. 프로젝트 주제 선정 배경

1) 홈바(home bar)의 수요 증가

코로나 19 이전 주로 술 마시는 장소의 통계가 집 40.2%, 술집 31.0%였으나, 코로나 19 이후 집 83.6%로 크게 증가했다는 롯데멤버스(코로나 이전과 이후 술 마시는 장소의 변화)의 통계 자료가 있다. 홈바의 수요가 2배 이상 증가했다고 볼 수 있는 자료이다. 이에 평소 제조가 어려워 칵테일바를 찾지 않는 이상 쉽게 접하기 어려웠던 칵테일 또한 수요가 증가한 홈바에서 쉽게 즐길 수 있도록 해보고자 하였다.

2) 하드웨어, 소프트웨어의 연동이 가능한 주제

기존에 학교에서 진행한 프로젝트들을 떠올려 보았을 때, 소프트웨어나 하드웨어를 단독으로 사용하는 프로젝트들을 진행한 적은 많았으나 하드웨어와 소프트웨어의 연동을 도전한 적은 없었다. 이에 하드웨어와 소프트웨어의 연동을 도전해볼 수 있는 주제를 생각해보게 되었고 ‘칵테일 제조 머신’이라는 주제가 적합하다고 생각이 되어 주제로 선정하게 되었다.

1.3. 프로젝트 목표

1) 안드로이드 어플리케이션 활용 주문

안드로이드 어플리케이션에 기본으로 등록되어 있는 칵테일 레시피를 사용하거나, 사용자별 맞춤 레시피를 등록할 수 있으며 등록되어 있는 특정 레시피를 선택할 시 해당 레시피를 ‘칵테일 제조 머신’으로 전달해준다.

2) 하드웨어로 자동 제조

안드로이드 어플리케이션에서 사용자가 선택한 레시피를 전달받고 해당 레시피에 따라서 칵테일을 완전 자동으로 제조한다.

3) AI 음성인식을 사용한 레시피와 노래 추천

AI 음성인식 기술을 통하여 도수에 따른 칵테일 레시피를 추천을 받을 수 있고 사용자가 원하는 분위기에 따른 노래 또한 추천을 받을 수 있다.

1.4. 프로젝트 기대 효과

1) 사용자 만족도 극대화

사용자가 자신이 선호하는 칵테일에 대한 레시피를 자유롭게 추가 및 수정할 수 있고, 칵테일 제조 머신에 들어갈 음료(술) 또한 자유롭게 구성할 수 있기에 사용자 개개인에게 완벽하게 맞춤으로 동작을 한다. 이로 인해 사용자의 만족도 극대화를 기대할 수 있다.

사용자가 칵테일을 잘 모른다고 하더라도 AI 음성인식을 사용한 레시피 추천을 통해 자신에게 잘 맞는 칵테일을 찾아갈 수 있기에 이 점으로 인한 사용자의 만족도 극대화 또한 기대할 수 있다.

2) 비즈니스 기회 창출

음식점, 레스토랑 등 전문 바텐더를 고용하기에 어려움이 있는 곳에서 ‘칵테일 제조 머신’을 도입 시 해당 장소의 고객들에게 더욱 퀄리티 있는 서비스를 제공할 수 있기에 다양한 장소에서의 도입으로 인한 비즈니스 기회 창출을 기대할 수 있다.

3) 음료 문화 발전

사용자들이 ‘칵테일 제조 머신’을 이용해 다양한 레시피를 시도해보고 괜찮다고 판단되는 레시피를 SNS 등을 통해 알리다 보면 음료 문화가 발전될 수 있을 것이라 기대할 수 있다.

2. 프로젝트 세부 사항

2.1. 프로젝트 요구 사항

사용자가 편리하게 ‘칵테일 제조 머신’을 이용할 수 있어야 한다.

사용자가 칵테일 제조의 과정에 전혀 개입하지 않더라도 칵테일의 제조가 원활하게 이루어져야 한다.

사용자가 본인 취향에 맞도록 ‘칵테일 제조 머신’의 레시피, 음료(술) 구성을 언제든 변경할 수 있어야 한다.

사용자가 칵테일에 대해 잘 모를 수도 있으니 사용자에게 칵테일을 추천할 수 있어야 한다.

‘칵테일 제조 머신’은 그 자체로 홈바(home bar)가 될 수 있기에 사용자가 칵테일을 마시며 홈바(home bar)를 즐길 수 있도록 분위기를 조성할 수 있어야 한다.

2.2. 프로젝트 요구 사항에 따른 세부 기능

1) 사용자의 편리한 이용

사용자가 편리하게 이용할 수 있도록 대다수의 사용자에게 익숙할 안드로이드 어플리케이션을 제작해 제공한다.

안드로이드 어플리케이션의 구성 또한 최대한 단순하며 사용하기 쉽도록 한다. 이를 위해 최소화된 어플리케이션의 메뉴 구성, 시각적인 효과 등을 구현한다.

2) 칵테일의 완전 자동 제조

사용자가 ‘칵테일 제조 머신’에서 직접 조작을 해야 하는 부분은 오직 레시피 추가와 칵테일 주문만을 조작하도록 한다. 해당 조작 또한 안드로이드 어플리케이션에서 벗어나지 않도록 어플리케이션 내부에서만 이루어진다.

사용자가 본인이 마실 칵테일을 선택 시 안드로이드 어플리케이션과 하드웨어 소통의 과정을 완전 숨기고 사용자의 개입이 없도록 하여 완전 자동으로 보이도록 한다.

컵 준비부터 칵테일 제조까지 다양한 모터를 이용해 완전 자동화를 시킨다.

음료(술)을 다 사용하였거나 사용자가 다른 음료(술)을 사용하길 원할 때에만 부득이하게 사용자가 하드웨어 모듈에서 음료(술)을 교체할 수 있도록 한다. 이 부분 또한 음료(술)의 사용 내역을 내부에서 계산하여 교체가 필요할 시 사용자에게 알릴 수 있도록 한다.

3) 사용자의 취향에 맞는 구성 변경

사용자가 언제든 안드로이드 어플리케이션을 통해 자신에게 알맞은 칵테일 제조 레시피를 추가하거나 수정, 삭제할 수 있다.

음료(술)을 구성하는 음료(술) 디스펜서를 숨기지 않고 사용자에게 보이도록 하여 사용자가 언제든 원할 시 음료(술)을 교체할 수 있도록 한다.

4) 사용자에게 칵테일 추천

기본적으로 대중적인 칵테일 레시피를 안드로이드 어플리케이션에서 사용자에게 제공한다.

AI 음성인식을 사용해 사용자의 선호도에 맞는 칵테일을 추천할 수 있도록 한다.

5) 홈바(home bar)에 맞는 분위기 조성

칵테일바에서의 분위기를 홈바에서도 느낄 수 있도록 ‘칵테일 제조 머신’의 기본 디자인을 꾸민다. 조명, 색 등을 칵테일바와 유사하게 구성한다. 또한, 다양한 구성요소들이 이질적으로 느껴지지 않도록 신경을 써서 하드웨어를 제작한다.

AI 음성인식을 사용해 사용자의 선호도에 맞는 노래를 재생하여 홈바의 분위기를 한층 더 올릴 수 있도록 한다.

3. 프로젝트 진행 사항

3.1. 소프트웨어공학 적용 및 형상 관리

1) 소프트웨어공학 적용 – V모델 사용

다양한 기능 및 구성요소로 이루어지는 프로젝트이기에 각각에 대한 테스트가 중요하다는 생각이 들었고, 이에 테스트가 중요하게 다루어지는 V모델을 사용하였다.

소프트웨어와 하드웨어 각각의 기능 및 구성요소(사용되는 모터 등)를 개별로 테스트 후 이를 순서에 따라 연동시키며 이에 따른 테스트를 각 기능 연동시마다 진행하였다. 모든 기능을 연동 후에는 전체적인 테스트를 진행하며 프로젝트 요구사항에 맞게 동작이 진행되는지 살피고 이에 따른 코드 수정을 하였다.

2) 형상 관리

팀원들 개개인 별로 완전 역할 분배를 하였고 작업 시 모두 한 곳에 모여 작업하였기에 GitHub 등 형상 관리 전용 툴을 사용하지는 않았으나 공유가 필요한 내용의 경우 Google Sheets를 사용하였다.

[사용한 Google Sheets 바로가기(클릭)](https://docs.google.com/spreadsheets/d/1V9NFXZIQPiyhP4H3aRYUaIA5g5Fd5otASdjwbnsPKVY/edit?usp=sharing)

3.2. 사용 기술

1) 소프트웨어 사용 기술

1-1) 서버와 데이터베이스

XAMPP : 크로스 플랫폼 웹 서버 자유 소프트웨어 꾸러미다. 아파치 웹 서버, MariaDB, PHP, 펄을 포함하고 있으며 GNU 일반 공중 사용 허가서로 배포되며 자유롭고 쓰기 쉬웁 웹 서버이다. 마이크로소프트 윈도우, 리눅스, 솔라리스, 맥 OS 10 등에서 동작하며 주로 웹 개발에 사용된다.

1-2) 안드로이드 어플리케이션

FrameLayout : 부모 View 안에 포함되어 있는 자식 View들을 중첩시켜 나타내는 레이아웃이다.

AsyncTask : Thread와 Handler를 사용하는 대신 UI 상에서 간단하게 비동기 작업을 수행할 수 있도록 도와주는 클래스이다. Thread와는 달리 UI 요소를 직접 접근해 사용할 수 있다.

2) 하드웨어 사용 기술

2-1) 라즈베리파이

Python : 안드로이드 어플리케이션-라즈베리파이-아두이노 간 통신을 구현 시 사용한 프로그래밍 언어이다.

Bluetooth 통신 : 안드로이드와 통신을 하기 위해 Bluetooth 모듈을 설치 후 사용하였다.

Serial 통신 : 아두이노와 통신을 위해 Serial 모듈을 설치 후 사용하였다.

2-2) 아두이노

Serial 통신 : 작성한 프로그래밍 코드를 아두이노에 업로드하기 위해 아두이노의 기본 Serial 통신을 사용하였고, 라즈베리파이와의 통신을 위해 Software Serial을 별도로 지정 후 사용하였다.

Servo Motor : 동작 명령을 받을 시 컵 홀더에 보관되던 컵 중 하나를 ‘칵테일 제조 머신’에 옮기는 기능 구현을 위해 Servo Motor 2개를 사용하였다.

Linear Actuator, Linear Rail(Step Motor) + Motor Driver : 각 음료(술) 모듈로의 이동에는 Linear Rail을 음료(술) 모듈을 따르기 위해 레버를 누를 시에는 Linear Actuator를 사용하였으며 Linear Rail과 Linear Actuator를 제어하기 위해 Motor Driver를 각각 사용하였다.

2-3) 기타 하드웨어 부품

AutoDesk Fusion 360 : 부품을 구하기 어려웠던 음료(술) 모듈을 누르기 위한 보조 받침대와 컵 홀더 측의 부품들은 직접 3D 모델링을 한 후 3D 프린터를 통해 출력하여 사용하였다.

3) AI 사용 기술

Python : 라즈베리파이 상에서의 AI 구동 시, 일반 환경에서의 AI 테스트 시 프로그래밍에 사용한 언어이다.

Jupyter Notebook : 일반 환경에서 AI 테스트 시 사용한 환경이다.

Speech Recognition 라이브러리 : Python에서 음성 인식에 사용할 수 있는 라이브러리이다.

Google Text-to-Speech(gTTs) 라이브러리 : 텍스트를 음성(mp3)으로 바꿀 수 있는 라이브러리이다.

3.3. 역할 분담

1) 서주완

프로젝트의 팀장 역할을 맡았으며 하드웨어들의 결합 및 하드웨어 전체 틀을 제작하였다. 또한, 라즈베리파이 파일들의 자동 실행 및 문서 작성을 맡았다.

2) 김경선

하드웨어 회로 구성과 회로 구성에 따른 납땜을 맡았으며 하드웨어 프로그래밍을 맡았다. 또한, 아두이노 프로그래밍과 라즈베리파이-아두이노 통신을 구현하였다.

3) 박준수

라즈베리파이에서 AI를 구현하였으며 하드웨어 결합 및 하드웨어 전체 틀 보조와 하드웨어 회로 구성에 따른 납땜 보조를 맡았다.

4) 정지석

안드로이드 어플리케이션 제작을 맡았으며 안드로이도-라즈베리파이 통신을 구현하였다. 또한, 필요 부품 3D 프린팅을 위한 3D 모델링을 맡았다.

3.4. 진행 일정

1) 중간고사 이전

프로젝트 요구 사항 작성 및 방향성 결정이 이루어졌다.

2) 11월

하드웨어 회로 구성 및 개발, 안드로이드 어플리케이션 개발, AI 개발, 하드웨어 전체 틀 제작 등 모든 개발 일정 진행 및 진행 사항에 따른 개별 테스트 진행이 이루어졌다.

3) 12월

하드웨어-소프트웨어들의 전체 연동 및 이에 따른 전체 기능 테스트, 문서 작성 등 프로젝트 마무리가 이루어졌다.

4) 자세한 진행 일정

자세한 진행 일정에 대해서는 2.1. – 2) 형상 관리에 하이퍼링크를 걸어 둔 Google Sheets – 일정 관리 탭에서 확인할 수 있다.

4. 개발 과정

4.1. 소프트웨어 개발 과정

1) 서버, 데이터베이스 개발 과정

1-1) 서버 설계

서버는 다중이 아닌 단일 어플리케이션에서 사용 예정이므로, 과부하가 발생하지 않을 것이라 생각이 되었기에 간단히 사용자 PC에서 포트 포워딩하여 Apache 서버를 사용하였다.

1-2) 데이터베이스 설계

데이터베이스는 MySql로 각 디스펜서 6개의 모듈(음료(술)이 들어가는 부분이 모듈이며 이후 모듈로만 표기)을 설정하고, 모듈 추가, 레시피 사용 등 통신을 통해 데이터를 불러와 값을 읽을 수 있도록 하였다.

모듈 테이블은 모듈ID를 주요 키로, 모듈 이름, 모듈 사진, 모듈 추가 시간, 모듈ml등으로 구성하였다.

칵테일 모듈의 수는 현재 ‘칵테일 제조 머신’상에서는 6개로 한정되어 있기 때문에 모듈이 추가될 시를 대비해 모듈 추가 테이블을 따로 생성하였으며 모듈 테이블을 업데이트 할 수 있도록 설계하였다.

모듈 추가 테이블은 모듈 테이블과 구성이 같다. 다만 모듈 테이블의 column개수는 6개로 한정하고, 모듈 추가는 insert시 모듈id가 자동으로 증가하도록 하였다.

레시피 테이블은 음료(술일 수도 있으나 이하 음료로만 표기)id를 주요 키로 음료 이름, 음료 소개, 음료 사진, 음료 동작 시간, 음료 종류, 음료 투여 방법으로 구성하였다.

음료 종류는 알코올, 논 알코올, DIY로 구성하고, 음료 방법은 '120000' 이라면 1번 모듈을 한 번 투여, 2번 모듈이 두 번 투여되도록 하드웨어에게 지시를 하게 되는 모듈 투여 레시피이다.

또한, 음료 투여 방법에 따라 음료가 투여될 시 모듈 테이블의 ml가 1번 투여 당 -30ml씩 자동으로 감소하도록 하였다.

이 방법은 레시피와 하드웨어 모듈이 하드코딩 되어있어, 일일이 수정해야 한다, 지금은 변경없이 모듈 6개만을 사용하기로 한정되어 있기에 상관이 없으나 추후에 서비스 확장 시 조건절을 수정하여 동적으로 레시피가 교체되도록 음료방법을 다시 설계할 예정이다.

2) 안드로이드 개발 과정(안드로이드 스튜디오)

2-1) 안드로이드 어플리케이션 화면 설계

어플리케이션의 UI는 사용자가 쉽게 사용할 수 있도록 태블릿 메뉴판을 모방해 제작하였다.

프레임 뷰로 상위 프레임과 하위 프레임의 화면 전환이 자연스럽도록 애니메이션 효과를 사용하였다.

2-2) 데이터베이스와의 연동

어플리케이션과 데이터베이스의 연동은 Http통신과 비동기로도 통신이 가능한 Asynctask를 사용해 php통신으로 데이터베이스의 값을 불러와 사용하도록 하였다.

이 방법은 현재는 작동에 문제가 없지만, 후에 서비스 확장 시, 데이터가 방대해지면 RxJava, Coroutine, retrofit2등 다른 통신을 사용하도록 할 예정이다.

2-3) 라즈베리파이와의 연동

어플리케이션과 라즈베리파이의 연동 방법으로는 Bluetooth 통신과 서버 통신이 있었다.

서버 통신은 가상 서버를 열거나 혹은 고정적인 위치에서 작동해야 하는 제약이 있었으며, 이미 데이터베이스와의 연동에서 서버를 사용 중인데 라즈베리파이와의 연동을 위해 이중으로 서버를 사용하는 것은 비효율적이라 생각이 되었다. 이에 블루투스 통신을 사용하였다.

Bluetooth 통신은 blueterm 어플리케이션의 코드를 GitHub에서 받아 수정하여, 메인 화면에서 프레임 뷰로 화면 전환 전에 블루투스 통신으로 라즈베리파이와 연동이 되도록 프로그래밍하였다.

4.2. 하드웨어 개발 과정

1) 하드웨어 결합 및 전체 틀 제작

하드웨어 구성 초기에는 예산적인 이유로 어느정도 무게를 견딜 수 있으면서도 저렴한 PP박스를 사용해 구성요소들을 고정하고자 하였다. 그러나 PP박스가 생각만큼 무게를 버텨주지 못하였고 이를 보강하기 위해 PP박스에 아크릴 판을 부착 후 구성요소들의 동작을 고려해 각각을 적당한 위치에 고정하고자 하였다.

동작 시에 컵이 들어가는 부분과 컵이 홀더에서 떨어질 시 이동하는 컵 미끄럼틀을 제작하기 위해 PP박스를 잘라서 본드로 고정하는 방식을 사용하였으나 정확한 사이즈 측정과 제작에도 불구하고 동작에서 다양한 문제가 발생하였다. 이에 계속해서 수정 혹은 새롭게 제작, 고정을 하였고 이 과정에서 많은 시간이 소요되었다. 이와 같은 과정을 반복하였음에도 PP박스의 한계 상 안정적인 고정이 되지 않았다.

동작 시에 발생하는 다양한 문제를 해결하기 위해 구성요소들과 아두이노의 동작 시간을 고려, 반복적인 테스트를 통해 안정적인 위치를 찾아 해당 위치에 구성요소들을 정확하게 고정하였다.

또한, 컵 홀더의 초기 구성은 DC 모터를 사용해 컵을 하나만 떨어트리도록 톱니를 돌리는 것이었으나 DC 모터의 출력이 부족해 서보 모터로 동작을 하는 컵 홀더로 변경을 하였다.

2) 하드웨어 회로 구성

하드웨어 회로 구성 시 가장 주의할 사항은 아두이노에서 나가서 라즈베리파이로 보내는 값은 5V이나 라즈베리파이가 받을 수 있는 값은 3.3V 였던 것이었다. 이에 전압을 낮추려고 아두이노에서 라즈베리파이로 값이 전달되는 선 사이에 10K옴의 저항 2개를 직렬 연결하여 완벽하게 3.3V는 아니더라도 라즈베리파이가 값을 받을 수 있도록 하였다.

이후 다른 구성요소들과 아두이노와의 연결에서 크게 주의할 사항은 없었으나 납땜을 편하게 할 수 있는 기판이 없어서 아두이노에 직접 납땜을 해야 하는 어려움이 있었다.

Linear Actuator와 Linear Rail은 12V를 사용해야 했기에 각각에 맞는 Motor Driver를 연결하여 아두이노에서 제어할 수 있도록 회로를 구성하였다.

각 구성요소가 아두이노의 어떤 핀에 연결되었는 지는2.1. – 2) 형상 관리에 하이퍼링크를 걸어둔 Google Sheets – HW관리 탭에서 확인할 수 있다.

3) 하드웨어 프로그래밍

3-1) 각 구성요소 개별 프로그래밍 및 테스트

제일 우선적으로 각 구성요소(Servo Motor, Linear Actuator, Linear Rail, 라즈베리파이와의 Serial 통신)에 대한 각각의 테스트 코드를 작성 후 작성한 코드의 의도에 맞게 구성요소가 동작하는지 테스트를 하였다.

Servo Motor : 2개의 Servo Motor를 사용하였으며 아두이노에서 기본적으로 제공되는 <Servo.h> 헤더 파일과 헤더 파일에서 제공하는 함수를 사용하였다. Servo Motor만을 확인하는 테스트에서 의도에 맞게 동작하는 것을 확인 후 Servo Motor가 사용될 컵 홀더에 장착, 컵 홀더에서의 테스트를 진행하였다. 컵 홀더에서의 동작은 2개의 Servo Motor가 서로 반대로 동작하며 컵을 하나씩만 아래로 떨어트리는 동작을 했어야 하나 의도대로 동작하지 않았다. 이에 다시 회로와 Servo Motor를 확인해보았는데 2개 중 하나의 Servo Motor 내부가 녹아 있는 것을 알게 되었고 여유분의 Servo Motor로 교체를 하였다. 이 후 다시 동작을 시켜보았더니 의도대로 동작을 하였다.

Linear Actuator : Linear Actuator는 위, 아래로 올라갔다 내려오는 동작을 하는 구성요소로 별도의 헤더파일이 필요 없었기에 단순히 OUTPUT 모드로 설정해 사용하였다. 위, 아래로 올라갔다 내려오는 것을 반복하는 테스트 코드를 작성 후 테스트 시 특별한 문제없이 작동하였다.

Linear Rail : Linear Rail은 컵이 레시피에 맞는 모듈을 밑으로 가서 음료를 받을 수 있도록 위치를 조절해주는 동작을 하는 구성요소로 사용을 위해 제조사에서 제공한 <HCMotor.h> 헤더 파일과 헤더 파일에서 제공하는 함수를 사용하였다. 초기 테스트에서는 단순 왼쪽, 오른쪽으로 이동하는 코드를 작성하였고 테스트 시에 해당 동작이 특별한 문제없이 작동하는 것을 확인할 수 있었다.

라즈베리파이와의 Serial 통신 : 라즈베리파이에서 아두이노로 값을 받아오고 아두이노에서 동작 후 라즈베리파이로 동작 완료를 알리는 ‘end’를 보내도록 코드를 작성하였다. 해당 과정에서는 문제가 정말 많이 발생을 하였는데 우선 사용한 아두이노 모델인 Uno는 Serial 통신을 할 수 있는 Rx/Tx가 하나 밖에 존재하지 않았다. 하지만 이는 코드를 작성하는 컴퓨터와의 통신을 위한 것이었기에 사용을 할 수가 없었다. 이에 소프트웨어 적으로 Rx/Tx를 하나 더 만들고자 하였고 <SoftwareSerial.h> 헤더 파일과 헤더 파일에서 제공하는 함수를 이용해 A0(기존에는 2번 핀이었으나 테스트 과정에서 통신이 되지 않아 A0 핀으로 회로 구성을 바꾸었더니 동작이 되었다.)와 3번 핀을 Rx/Tx로 지정하였다. 이후 통신을 할 수 있는 코드를 작성해 테스트를 해보았고 수많은 문제를 걸친 후에 동작이 정상적으로 되는 것을 확인할 수 있었다. 이 과정에서 겪은 문제들은 원인을 알 수 없는 문제들이 많았다. 동일한 코드를 다른 파일에서 실행했더니 안되는 경우도 있었고 전에는 정상 작동하였으나 재실행 하였더니 이상 값이 나오는 경우도 있었다. 해당 문제들은 시간을 쏟아 반복적으로 재실행, 새로운 파일에서 실행 등의 과정을 걸쳤더니 해결이 되었다.

3-2) 전체 연동 프로그래밍 및 테스트

우선 전체 연동 시의 동작 흐름을 생각하였다. 전체 동작의 흐름은 라즈베리파이에서 아두이노로 동작 명령(레시피 값, ‘102000’ 형태로 구성되어 있으며 ‘102000’과 같은 경우는 1번 모듈 1번, 3번 모듈 2번의 뜻이다.)이 올 시 해당 값을 우선 받고 동작을 시작한다. 사용자의 편의성을 위해 올라가 있던 Linear Actuator가 내려온 후 컵 홀더에서 Servo Motor를 동작 시켜 컵을 하나 빼온다. 컵을 빼온 후에는 값이 있는 모듈로 Linear Rail을 동작 시켜 컵이 해당 모듈의 밑에 위치할 수 있도록 한다. 이후 해당 모듈을 넣어야 횟수만큼 Linear Actuator를 동작 시킨다. Linear Rail 이동과 Linear Actuator의 동작이 레시피에서 적힌 대로 끝난 후에는 컵이 초기 위치로 돌아오도록 Linear Rail을 다시 동작을 시키고 사용자가 컵을 빼기 편하도록 Linear Actuator를 위로 올려 보낸 후 동작이 종료된다. 이후 동작 종료를 라즈베리파이에게 알려 다음 동작이 다시 실행될 수 있도록 한다.

이 동작 흐름을 한 코드에서 구현하기 위해 우선 자주 실행되는 동작들과 컵을 하나만 빼는 Servo Motor의 동작 코드들을 함수로 따로 분리하였다. 여기서 분리된 함수들은 cup()(컵 홀더에서 컵을 빼는 함수), make(int module[6], int module\_sum)(칵테일을 제조하는 주요 함수), rail\_right(int second)(Linear Rail을 오른쪽으로 이동시키는 함수), rail\_left(long second)(Linear Rail을 왼쪽으로 이동시키는 함수), ac\_updown()(Linear Actuator가 올라갔다가 내려오도록 하는 함수)이다. 해당 함수들에 함수의 기능을 할 수 있도록 코드를 작성하고 이후 동작 흐름대로 코드가 동작하도록 코드를 완성시켰다. 이후 테스트를 하며 보다 동작이 잘 될 수 있도록 코드를 조금씩 수정하며 완성도를 높였다.

코드를 작성하며 까다로웠던 부분은 Linear Rail이 정확한 위치에서 멈추고 정확한 거리만큼 이동할 수 있도록 하는 것이었는데 초기에는 Linear Rail의 위치를 인식할 수 있는 센서를 모듈별로 부착해 정확한 위치로 이동할 수 있도록 하고자 하였다. 그러나 예산 상의 이유와 회로 구성의 측면에서 어려움이 생길 것 같아 이동 시간을 측정하여 정확하게 이동할 수 있도록 하였다.

4.3. AI 개발 과정

음성 인식 기반 AI 시스템은 초기 목표로 설정한 음성 인식 후 도수에 따른 칵테일 추천과 노래 추천 기능을 구현하기 위한 시스템이기에 Python 언어를 사용해 해당 기능들의 로직을 구현하였으며 초기에는 텍스트 입력을 통한 결과 출력 확인 후 이를 음성 인식으로 바꾸어 개발을 하였다.

음성 인식과 음성 출력을 위해 사용한 라이브러리는 각각 Speech Recognition과 pyttsx였다. 그러나 pyttsx 라이브러리의 한국어 발음 미지원 문제를 알게 되었고 이 문제를 해결하기 위해 한국어 발음을 정확하게 지원하는 Google Text-to-Speech(gTTs) 라이브러리를 최종적으로 사용하게 되었다.

음성 인식을 구현하던 중 도수가 쎈 칵테일의 경우 부정확한 인식이 발생하였다. ‘쎈’과 ‘센’을 AI가 구별하기에 ‘쎈’을 발음하든 ‘센’을 발음하든 동일한 동작을 하였어야 하나 ‘쎈’ 발음 시 동작을 하지 않았고 이에 ‘쎈’과 ‘센’ 두 발음을 모두 ‘센’으로 처리하도록 replace 함수를 활용해 수정하였다.

프로그램이 “안녕 연암’을 입력 받기 전까지 계속해서 실행 상태(대기)를 유지하도록 하기 위해 while문을 활용해 메인 루프를 작성했다.

최종적으로 “안녕 연암”이라는 호출어를 사용자가 마이크에 입력 시 도수 별로 칵테일 추천 및 분위기 별 노래 재생이 가능한 인공지능이 완성되었다.

초기에는 여기까지의 과정을 윈도우 환경 위에서 Jupyter Lab을 활용해 구현 후 테스트를 하였다. 동작이 잘 되는 것을 확인하고 라즈베리파이에서 해당 코드를 실행하기 위해 라즈베리파이에 Python을 설치 후 동일 코드를 작성하고 해당 코드를 실행을 시켜보았으나 윈도우 상에서는 잘 동작이 되었던 코드가 라즈베리파이에서는 동작이 되지 않았다. 이에 다양한 모듈을 다운로드 및 버전 조정을 한 후 라즈베리파이에서도 동작이 잘 되는 것을 확인할 수 있었다.

라즈베리파이에서의 동작 테스트도 의도했던 대로 실행이 되는 것을 확인 후 라즈베리파이에 전원만 공급하더라도 해당 AI 파일이 자동으로 실행이 되도록 라즈베리파이의 환경 설정을 변경하였다. 이 후 해당 부분에 대해서도 테스트를 진행하였고 AI의 모든 기능이 잘 동작이 되는 것을 확인하였다.

5. 결과

5.1. 프로젝트 결과

1) 소프트웨어 결과



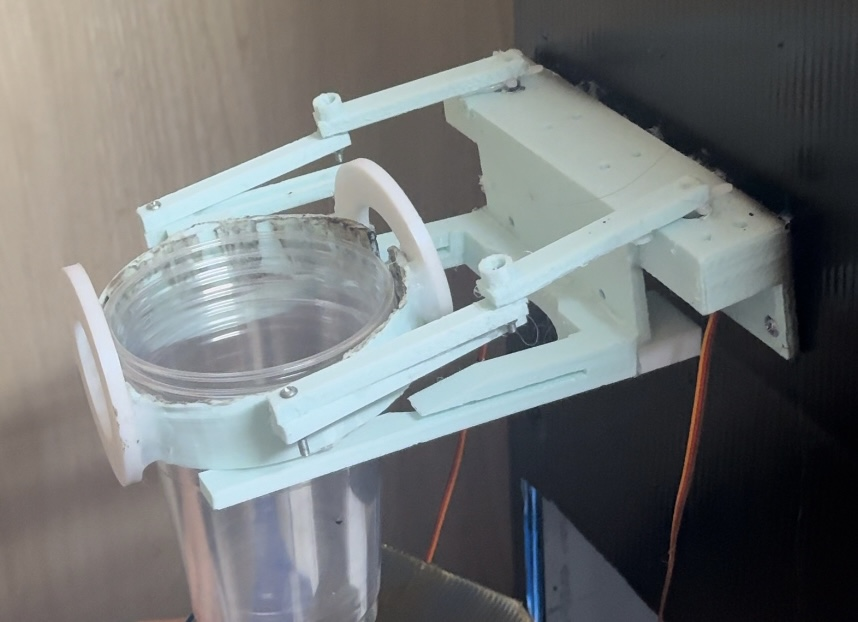
안드로이드 어플리케이션 : YC Bar(안드로이드 어플리케이션 이름)을 실행 시 처음에 라즈베리파이와의 블루투스 연결을 수행하며 이 후 칵테일, 논알코올, DIY 탭을 선택해 칵테일을 주문하거나 직접 레시피를 추가할 수 있다. 주문 후에는 제조에 걸리는 예상 시간이 화면에 표시된다. 제조가 끝난 후에는 새로운 칵테일을 다시 주문할 수 있다.

AI : 마이크와 스피커를 사용한다. 마이크로는 사용자의 음성을 입력해 라즈베리파이가 이를 인식하고 이후 라즈베리파이의 응답이 스피커에서 나오는 형식이다. 동작 과정은 마이크에 안녕 연암을 말할 시 동작이 시작되며 이후 도수가 높은 칵테일과 낮은 칵테일을 사용자가 선택, 이에 맞는 칵테일을 AI가 추천해준다. 이후 신나는 노래와 잔잔한 노래를 사용자가 선택 시 선택에 알맞은 노래를 재생시켜준다. 사용자의 선택은 모두 마이크를 통한 음성 입력을 통해 이루어진다.

2) 하드웨어 결과



하드웨어의 전체적인 결과물 모습이 바로 위 사진이다. 라즈베리파이로부터 아두이노가 동작 명령을 받을 시 기존에 올라가 있던 Linear Actuator가 내려오고 왼쪽의 미끄럼틀로부터 컵이 하나가 떨어진다. 이후 하단에 깔려 있는 Linear Rail을 통해 레시피 상에 들어가게 되는 음료 밑으로 컵이 이동을 하고, Linear Actuator가 올라가 음료에 연결된 레버를 전달받은 레시피에 명시된 횟수만큼 누른다. 이 동작을 반복하며 레시피에 명시된 음료가 모두 컵에 담길 시에는 컵이 다시 왼쪽으로 이동하고 Linear Actuator가 위로 올라간 후 동작이 종료된다.



이전 사진의 하드웨어 왼쪽 벽 바깥에 달려있는 컵 홀더 부분이다. 라즈베리파이로부터 아두이노가 동작 명령을 받을 시 Linear Actuator가 내려온 이후 Servo Motor 두개를 동작 시켜 컵이 하나만 빠질 수 있도록 동작한다. 사진 상에서는 보이지 않으나 해당 구조물의 팔 부분을 Servo Motor가 당기면 컵을 바치고 있는 부분의 안쪽에 컵이 하나만 들어가고 그 부분의 넓이가 받치는 부분보다 넓어서 하나만 들어간 컵이 그대로 빠지는 방식이다.

5.2. 프로젝트 개선점

1) 소프트웨어적 개선점

현재는 AI의 추천 레시피가 안드로이드 어플리케이션으로 자동 연동이 되지 않고 추천을 받은 레시피를 사용자가 직접 추가를 해야 한다. 이를 자동 연동되도록 개선해보고 싶다.

2) 하드웨어적 개선점

동작 테스트 시 Linear Rail이 간혹 동작하지 않는다는 문제점을 발견하였다. 코드 상에는 문제가 없었기에 해당 문제점의 원인을 발견하고 해결해보고 싶다.

예산적 한계로 인해 속도 측면에서나 전체적인 완성도(안정성) 측면에서 부족한 부분이 있다. 이를 기회가 된다면 업그레이드를 해보고 싶다.