# Use Case

- Automatic drinks maker
  - Bluetooth enabled to allow user to select drinks
  - Avoids manual labour by user
  - High precision of liquids measurement/percentages
  - Insightful and interesting data metrics presented to user
  - User can customize/create drink recipes for machine to undertake
- Has sensors to weigh the bottles for exact measurements
- Has visual sensor to ensure glass in under dispenser
- IoT enabled device
  - Using raspberry pi for communications and sensors
- Areas Covered:
  - Software Systems, Hardware Systems (embedded)

# Requirements (Customer)

- Liquids must be pre filled into the machine
  - Liquid names must be entered into the phone application
  - Minimum amount of liquid in each bottle must be satisfied or else app will alert users
- Cup must be under dispenser
- iOS device is required for application installment
- Device will have to be connected to power
  - Raspberry Pi & Arduino
- Only one glass at a time on the belt (hence only one customer at a time)
  - Can queue drinks orders on apps and machine will execute in order once glass is placed on belt

# Requirements (Hardware)

- Raspberry Pi 4 Model B
  - Bluetooth Communication with iOS device
  - Process sensor readings
  - Control Motors
- FSR sensor to check if glass is actually on platform
  - 100% accuracy required as liquids should not be poured if no glass
- Motors, timing belt, and sliding platform to move the glass on the belt in position for the different liquids
  - Movement must be 100% stable as we cant have the glass tipping over or liquids falling.
  - As the glass gets fuller the stopping and starting of the belt must be very smooth as abrupt start/stop will cause liquids to pour out
  - Positioning of glass (under the  we can allow for 95% accuracy) as glass is round so not required to pour exactly in center of glass.

# Requirements (Software)

- iOS App
  - Communication protocol between phone and raspberry pi will be done using **MQTT protocol**
  - App is built using Swift for iOS
  - User must be able to register and log in so will need backend database to store this information
    - This is needed for individual users data insights (drinking habits, calories etc.) that the app will provide to them
  - User must be able to choose a drink and the machine should make this drink (100% accuracy)
    - App must confirm selection or reject selection in ~50ms
    - Different users can all login to the app and add drinks to the nearby machine and these drinks will be queued up and executed
    - App should not allow user to choose a drink that cannot be made (liquids are finished or the required liquids for the drink are not available.)
  - App will provide insightful weekly report to users on liquids consumed, total calories, total drinks per day etc.

# Technical Challenges

- Drink accuracy (ratio of liquids, correct drink poured)
- Moving platform won’t spill drink with arbitrary amount of liquid
- Platform positions cup correctly under each dispenser
- Lossless communication between app and system
- Learning Swift
- Making up-to-date liquid levels available in-app

# Risk Mitigation

- Adjustable valves to account for over/under-pouring
- 3 preset cup sizes to determine how fast the platform can move (similar to Keurig mug options)
- Weight sensor to determine amount of liquid left in bottles
- Weight sensor to determine amount of liquid dispensed into cup on each pour

# Solution Approach (Mechanical/Hardware)

- Liquid Dispenser
  - Control valves with variable volumes
  - Wooden structure to support 4 liquids
- Raspberry Pi
  - Communicate with sensors
    - Weight
    - Position
  - Control moving platform
    - Timing belt
    - Stepper motors

# Solution Approach (Software)

- iOS app created in Swift
- Communicate with RPi over Bluetooth
- Allow users to request a drink from the app
- Track liquid levels direct from the app

# Testing, Verification, and Metrics

- Hardware
  - Liquid is dispensed within 2 mL of what is expected
    - Use liquid density and weight sensors to determine volume
  - Timing belt places glass under valve with 100% accuracy
    - Ensure motor accounts for differently weighted/sized glasses to deliver to correct spot
- Software
  - Drink process is started within 3 seconds of sending request from the app
  - Ensure correct drink is dispensed
    - Keep track of liquids dispensed and drink selected once a request is received

# Tasks and Division of Labor

- Mechanical
  - Build enclosure - Tyler, Aryan
  - Connect dispensers to platform - Tyler, Aryan
- Hardware
  - Coding RPi to control servos, moving platform, valves - Tom
  - Sense when we are running low on liquids - Tom
  - Sense where cup is on platform - Tom
- Software
  - Establish Bluetooth communication with RPi - Tyler, Aryan
  - Add ability to select drink from pre-made list - Tyler, Aryan
  - Queueing system to support multiple users - Tyler, Aryan

# Schedule

| | 2/22/2021 | 3/1/2021 | 3/8/2021 | 3/15/2021 | 3/22/2021 | 3/29/2021 | 4/5/2021 | 4/12/2021 | 4/19/2021 | 4/26/2021 | 5/3/2021 | 5/10/2021 | 5/17/2021 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Deliverables** | | | | | | | | | | | | | |
| Proposal Presentation Slide | AC,TOM,TD | | | | | | | | | | | | |
| Proposal Presentation | AC, TOM, TD | | | | | | | | | | | | |
| Design Presentation | | | AC, TOM, TD | | | | | | | | | | |
| Final Presentation | | | | | | | | | | | AC, TOM, TD | | |
| Public Demo | | | | | | | | | | | | AC, TOM, TD | |
| Final Report | | | | | | | | | | | | | AC, TOM, TD |
| **Logistics** | | | | | | | | | | | | | |
| Ordering all hardware/software components | | TD | | | | | TD - Emergency/Backup if reqd. | | | | | | |
| Ethics Section | | | | | | ALL | | | | | | | |
| Weekly Status Reports | ALL | | | | | | | | | | | | |
| Individual Status Reports | ALL | | | | | | | | | | | | |
| **Design** | | | | | | | | | | | | | |
| Final design work | | ALL | | | | | | | | | | | |
| Final design confirmed | | | | | ALL | | | | | | | | |
| Building of outer casing | | | | | | AC, TD | | | | | | | |
| Building of converyor belt | | | | | | | AC, TD | | | | | | |
| Building of casing for bottles | | | | | | | | | AC, TD | | | | |
| **Implementation** | | | | | | | | | | | | | |
| Setting of Raspberry Pi | | | TOM | | | | | | | | | | |
| Configuring MQTT protocols | | | | AC, TD | | | | | | | | | |
| Configuring communications | | | | | AC, TD | | | | | | | | |
| Configuring sensors | | | | | | TOM | | | | | | | |
| Setting up phone app (SWIFT) | | | | | | | AC, TD | | | | | | |
| Testing communications between app/device | | | | | | | | | ALL | | | | |
| Refining UI for app | | | | | | | | | AC, TD | | | | |
| Creating data reporting modules for user | | | | | | | | | | AC, TD | | | |

# Conclusion

- The BaristaBro machine will bring a new level of comfort and excitement to enjoying drinks.
- The data insights provided to users will completely transform their experience and provide them with a new experience
- The ease of using the machine through the mobile app will guarantee a seamless process for the user.