PES UNIVERSITY, Bangalore

(Established under Karnataka Act No. 16 of 2013)

Department of Computer Science & Engineering

Mobile and Autonomous Robots (UE22CS343BB7)

6th Semester

<u>Mini-Project</u>

Project Title: Automated Barista Arm

Team Details:

1. **Aathil Nishad <PES2UG22CS011>**
2. **Arnab Bhattacharya<PES2UG22CS097>**
3. **Anusha Navale< PES2UG22CS087>**
4. **Dhanushree K< PES2UG22CS176>**

Professor Name: Dr. Gokul Kannan Sadasivam

Project Description: An automated robot arm created using Blender for the 3d environment and ROS2 Humble to send the commands to run the model

Project Objectives:

1. **Create the 3d models for the barista arm**
2. **Load and build the 3d model in the Blender workspace**
3. **Build a Flask server to receive commands**
4. **Build the ROS 2 Humble client to send the commands**

Methods and Materials:

1. System Design:

- ROS 2 Client (Ubuntu VM)
  - Runs a rclpy node that sends HTTP POSTs to Blender's Flask server.
  - Encapsulates each animation step (move, pick, serve, play, clear) as a REST call.
  - Sequence commands with delays so Blender has time to keyframe.

- Flask Server (inside Blender)
  - Embedded in Blender's Python environment via a background thread.
  - Exposes /move, /pick, /serve, /play, /clear endpoints.
  - Uses bpy.app.timers to schedule bone transformations and playback in Blender's context.

- Blender Scene & Armature
  - A rigged armature ("Armature") with bones named Base, ArmLong, ArmShort, ArmShortest.
  - Objects Cup & Cap whose visibility simulates pick/serve.
  - Keyframes inserted programmatically into Blender's timeline.

2. Algorithm/Model Development

- State Tracking
  - scene['barista_current_frame']: Next frame to insert keyframes.
  - scene['barista_bone_rotations']: Last known rotations per bone.
- Motion Planning (move function)
  - Calculate target frame = start + duration * fps.
  - Compute new Euler angles = last + delta (converted to radians).
  - Insert keyframes at start and target frames.
  - Update state variables.
- Visibility Control (pick/serve)
  - Keyframe hide_viewport & hide_render on Cup/Cap at the current frame.
- Playback & Reset (play_and_reset)
  - Use bpy.ops.screen.animation_play() to run the animation.
  - Register a timer to step frames until the end, then cancel playback and reset the state.
- Clearing (clear_animations)
  - Remove all keyframes from bones and objects.
  - Reset scene frame range and tracking variables.

3. Implementation Steps

1. Prepare Blender Scene
    - Import or build your rig and place Cup/Cap.
    - Save the .blend file.
2. Install Flask into Blender Python
    - Drop Flask packages under %APPDATA%\Blender Foundation\Blender\4.3\scripts\module.
    - Append this path to sys.path at script start.
3. Write & Embed Flask Server Script
    - Paste the full server code into Blender's Text Editor.
    - Run it so Flask listens inside Blender.
4. Develop ROS 2 Client
    - Create a new ROS 2 Python package.
    - Add blender_client.py with your sequence logic.
    - colcon build and source install/setup.bash.
5. Network Setup
    - Configure a VM network (bridged or NAT + port forwarding) so that <host ip> is reachable.
    - Test via curl from VM and Windows.
6. Test End-to-End
    - Launch Blender with the script.
    - Run ROS 2 node; observe Blender's timeline fill and animation play.

4. Hardware Components (if applicable)

❖ **Host Machine**
  ➢ Running Windows + Blender (for visualisation).

❖ **VM Guest**
  ➢ Ubuntu VM with ROS 2 Humble installed (for command logic).

(No physical robot needed—Blender simulates the arm.)

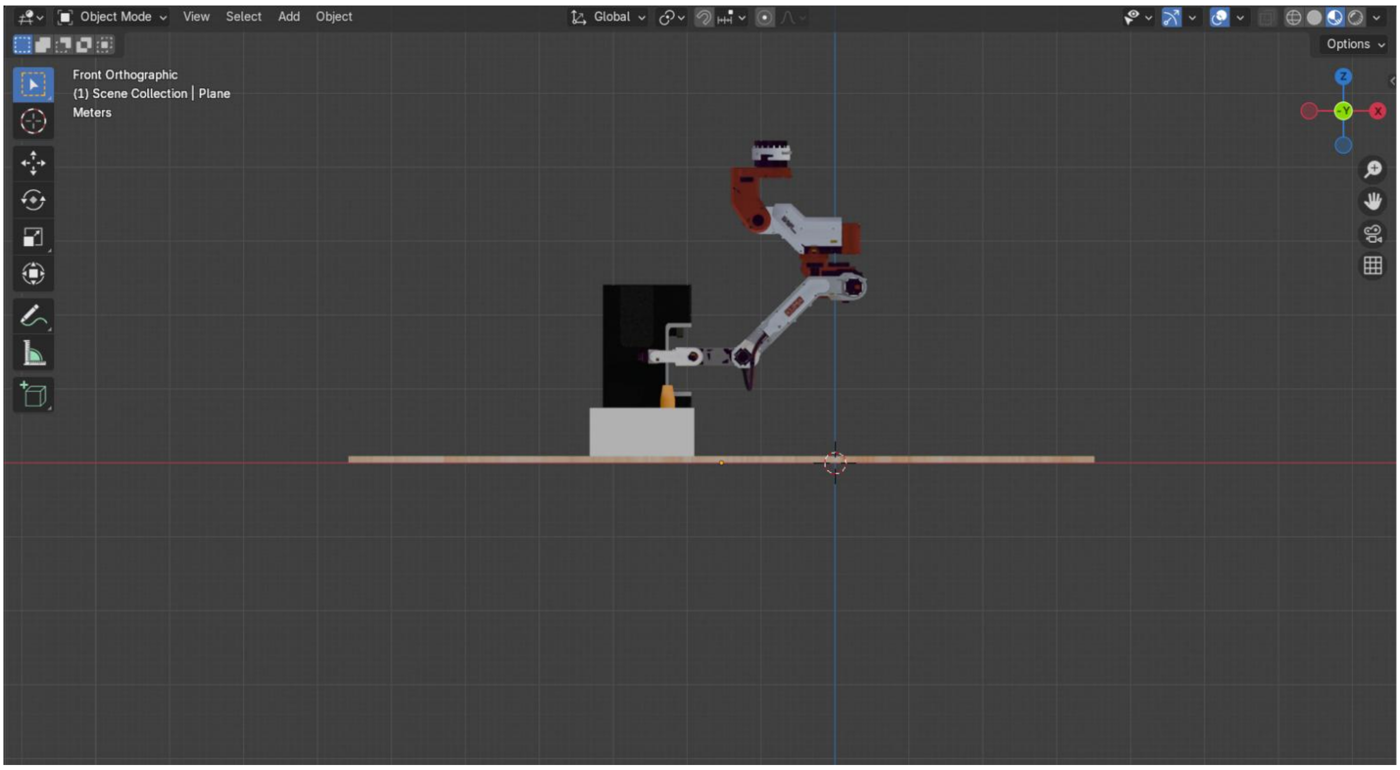
5. Software Tools

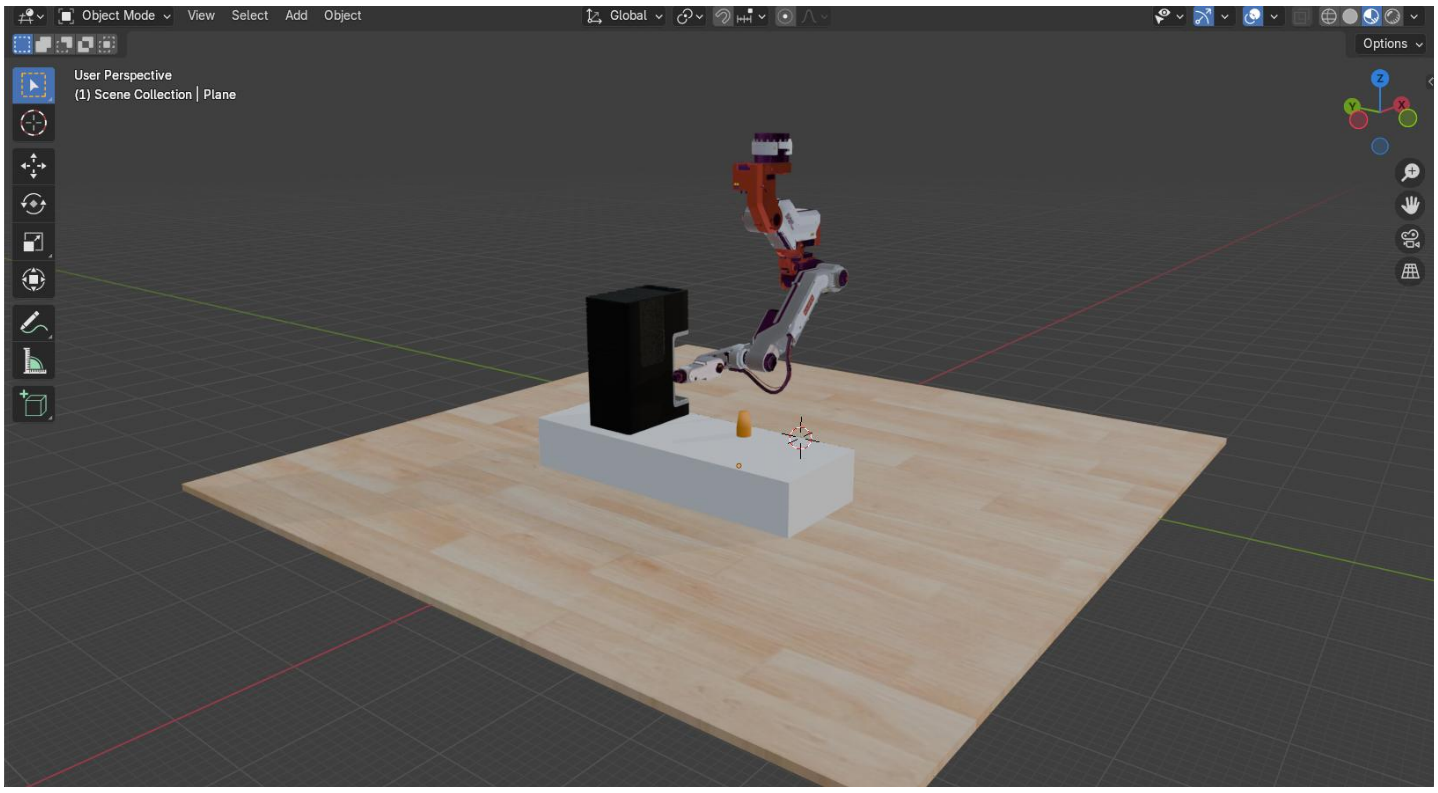
- **Blender 4.3**
  - Built-in Python API (`bpy`) for animation and keyframing.
- **Flask**
  - Lightweight REST server embedded inside Blender.
- **ROS 2 Humble**
  - `rclpy` for writing the client node.
- **Blender Python 3.10+ and Python3**
  - On both host (Blender's embedded Python) and guest (Ubuntu).
- **Networking**
  - VM network config (bridged/NAT) and `curl` for testing endpoints.

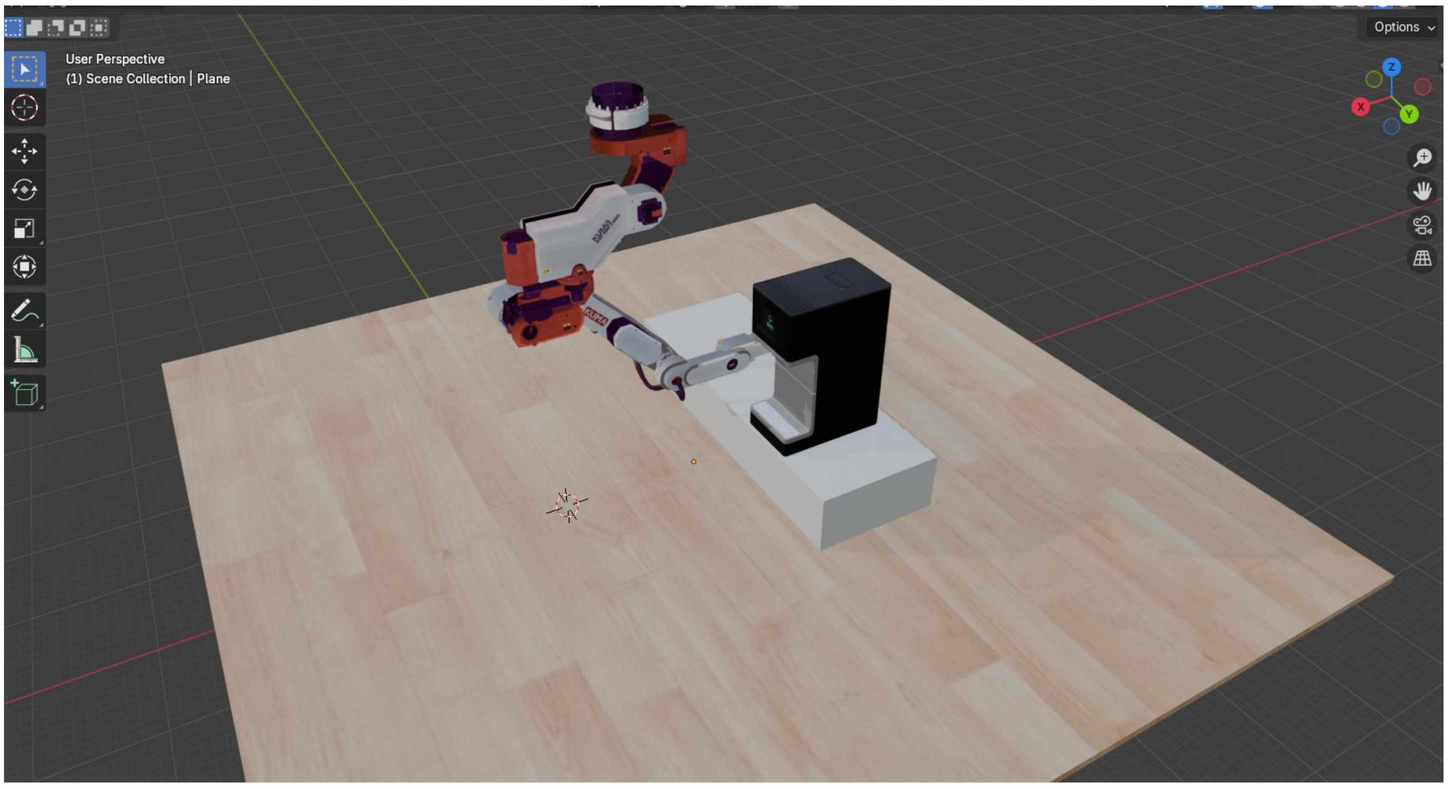
Project Outcome:

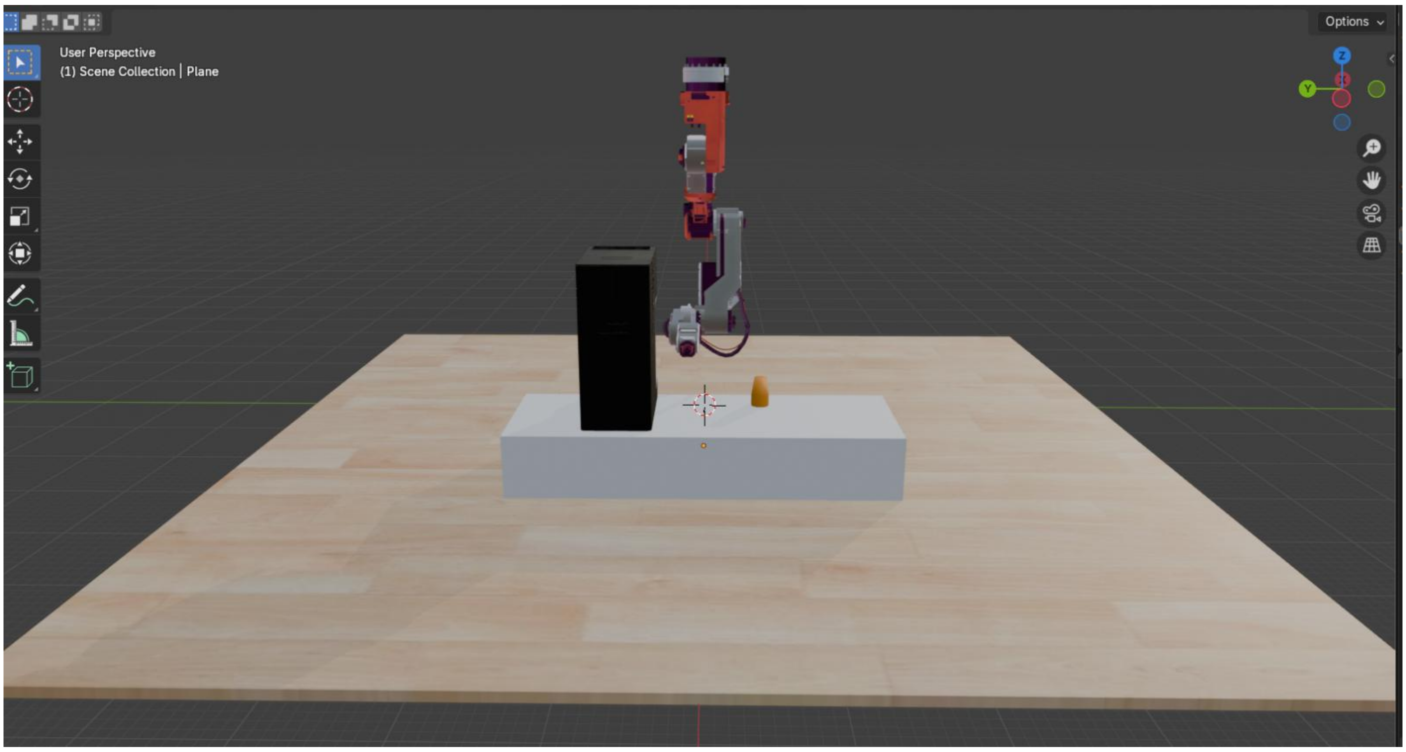
1. Output results

Model:

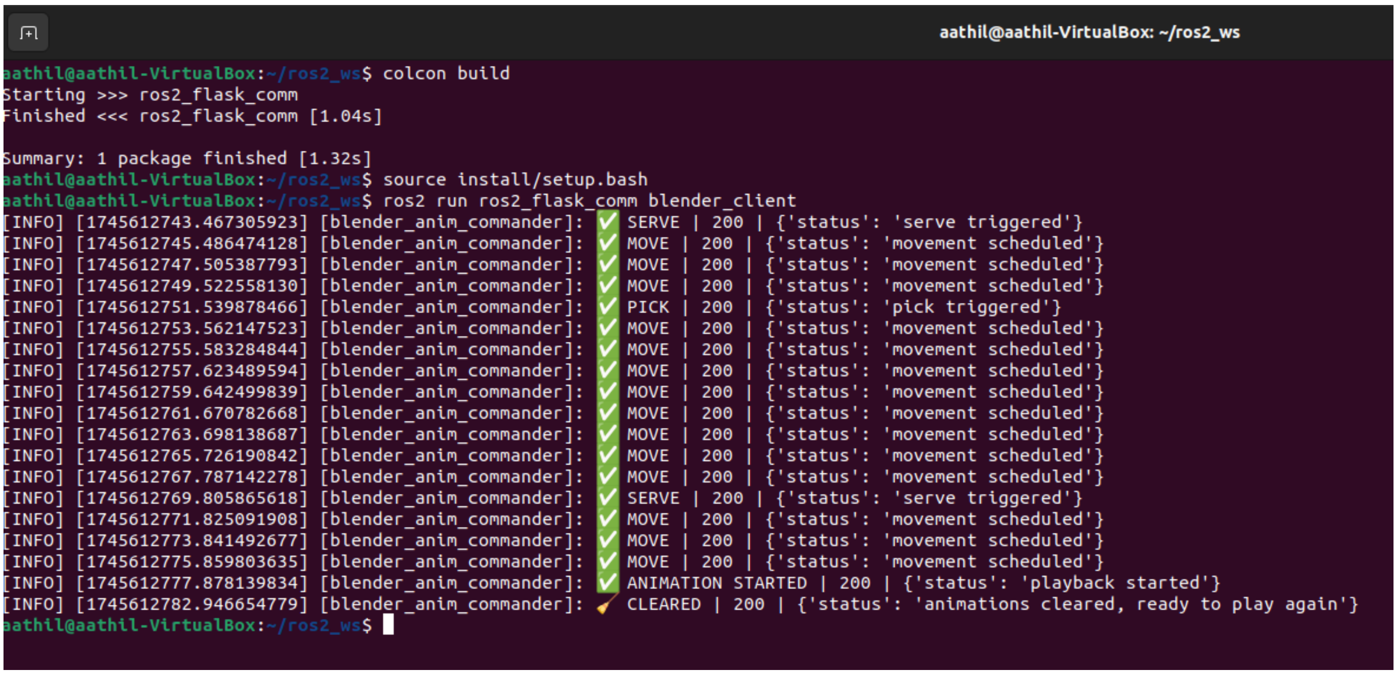



Object Mode
View
Select
Add
Object
Global
Options
User Perspective
(1) Scene Collection | Plane


Options
User Perspective
(1) Scene Collection | Plane



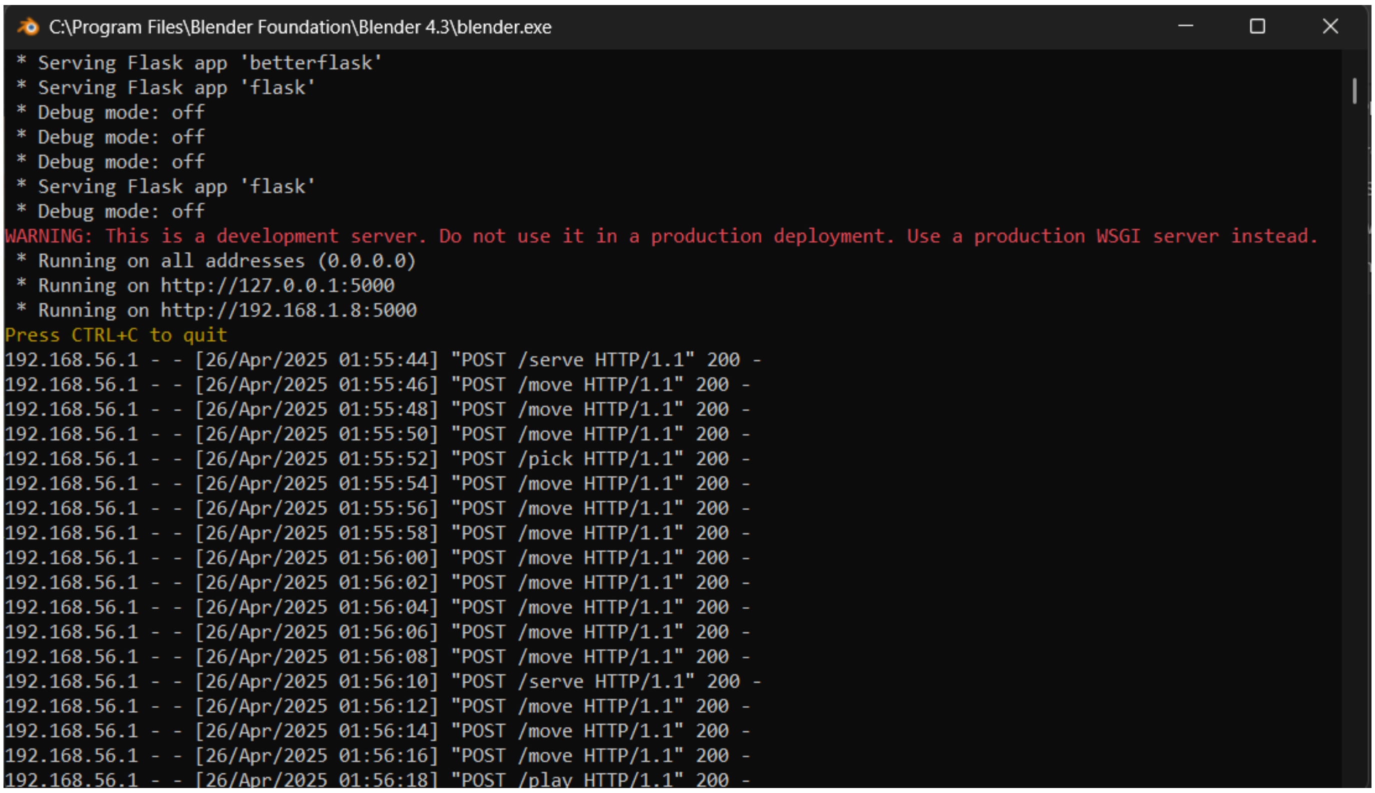
ROS2 Client:



```
aathil@aathil-VirtualBox: ~/ros2_ws
aathil@aathil-VirtualBox:~/ros2_ws$ colcon build
Starting >>> ros2_flask_comm
Finished <<< ros2_flask_comm [1.04s]

Summary: 1 package finished [1.32s]
aathil@aathil-VirtualBox:~/ros2_ws$ source install/setup.bash
aathil@aathil-VirtualBox:~/ros2_ws$ ros2 run ros2_flask_comm blender_client
[INFO] [1745612743.467305923] [blender_anim_commander]: ✅ SERVE | 200 | {'status': 'serve triggered'}
[INFO] [1745612745.486474128] [blender_anim_commander]: ✅ MOVE | 200 | {'status': 'movement scheduled'}
[INFO] [1745612747.505387793] [blender_anim_commander]: ✅ MOVE | 200 | {'status': 'movement scheduled'}
[INFO] [1745612749.522558130] [blender_anim_commander]: ✅ MOVE | 200 | {'status': 'movement scheduled'}
[INFO] [1745612751.539878466] [blender_anim_commander]: ✅ PICK | 200 | {'status': 'pick triggered'}
[INFO] [1745612753.562147523] [blender_anim_commander]: ✅ MOVE | 200 | {'status': 'movement scheduled'}
[INFO] [1745612755.583284844] [blender_anim_commander]: ✅ MOVE | 200 | {'status': 'movement scheduled'}
[INFO] [1745612757.623489594] [blender_anim_commander]: ✅ MOVE | 200 | {'status': 'movement scheduled'}
[INFO] [1745612759.642499839] [blender_anim_commander]: ✅ MOVE | 200 | {'status': 'movement scheduled'}
[INFO] [1745612761.670782668] [blender_anim_commander]: ✅ MOVE | 200 | {'status': 'movement scheduled'}
[INFO] [1745612763.698138687] [blender_anim_commander]: ✅ MOVE | 200 | {'status': 'movement scheduled'}
[INFO] [1745612765.726190842] [blender_anim_commander]: ✅ MOVE | 200 | {'status': 'movement scheduled'}
[INFO] [1745612767.787142278] [blender_anim_commander]: ✅ MOVE | 200 | {'status': 'movement scheduled'}
[INFO] [1745612769.805865618] [blender_anim_commander]: ✅ SERVE | 200 | {'status': 'serve triggered'}
[INFO] [1745612771.825091908] [blender_anim_commander]: ✅ MOVE | 200 | {'status': 'movement scheduled'}
[INFO] [1745612773.841492677] [blender_anim_commander]: ✅ MOVE | 200 | {'status': 'movement scheduled'}
[INFO] [1745612775.859803635] [blender_anim_commander]: ✅ MOVE | 200 | {'status': 'movement scheduled'}
[INFO] [1745612777.878139834] [blender_anim_commander]: ✅ ANIMATION STARTED | 200 | {'status': 'playback started'}
[INFO] [1745612782.946654779] [blender_anim_commander]: 🧹 CLEARED | 200 | {'status': 'animations cleared, ready to play again'}
aathil@aathil-VirtualBox:~/ros2_ws$ 
```

Flask Server:



C:\Program Files\Blender Foundation\Blender 4.3\blender.exe

```
 * Serving Flask app 'betterflask'
 * Serving Flask app 'flask'
 * Debug mode: off
 * Debug mode: off
 * Debug mode: off
 * Serving Flask app 'flask'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://192.168.1.8:5000
Press CTRL+C to quit
192.168.56.1 - - [26/Apr/2025 01:55:44] "POST /serve HTTP/1.1" 200 -
192.168.56.1 - - [26/Apr/2025 01:55:46] "POST /move HTTP/1.1" 200 -
192.168.56.1 - - [26/Apr/2025 01:55:48] "POST /move HTTP/1.1" 200 -
192.168.56.1 - - [26/Apr/2025 01:55:50] "POST /move HTTP/1.1" 200 -
192.168.56.1 - - [26/Apr/2025 01:55:52] "POST /pick HTTP/1.1" 200 -
192.168.56.1 - - [26/Apr/2025 01:55:54] "POST /move HTTP/1.1" 200 -
192.168.56.1 - - [26/Apr/2025 01:55:56] "POST /move HTTP/1.1" 200 -
192.168.56.1 - - [26/Apr/2025 01:55:58] "POST /move HTTP/1.1" 200 -
192.168.56.1 - - [26/Apr/2025 01:56:00] "POST /move HTTP/1.1" 200 -
192.168.56.1 - - [26/Apr/2025 01:56:02] "POST /move HTTP/1.1" 200 -
192.168.56.1 - - [26/Apr/2025 01:56:04] "POST /move HTTP/1.1" 200 -
192.168.56.1 - - [26/Apr/2025 01:56:06] "POST /move HTTP/1.1" 200 -
192.168.56.1 - - [26/Apr/2025 01:56:08] "POST /move HTTP/1.1" 200 -
192.168.56.1 - - [26/Apr/2025 01:56:10] "POST /serve HTTP/1.1" 200 -
192.168.56.1 - - [26/Apr/2025 01:56:12] "POST /move HTTP/1.1" 200 -
192.168.56.1 - - [26/Apr/2025 01:56:14] "POST /move HTTP/1.1" 200 -
192.168.56.1 - - [26/Apr/2025 01:56:16] "POST /move HTTP/1.1" 200 -
192.168.56.1 - - [26/Apr/2025 01:56:18] "POST /play HTTP/1.1" 200 -
```

Simulation video link (drive link)
https://drive.google.com/drive/folders/1T9J-yc6QgKTdb9UWdriupylL19uI_m0h?usp=drive_link

GitHub link (Source code)

https://github.com/Achlys2004/Baristah-Mistah.git