# MazeRunnerRobot User Manual

## About the Project:

The robot is designed to navigate a maze from start to goal. The user first builds a block maze in real life, then inputs the layout into the GUI. The system will send the maze data and solution to the robot, allowing it to navigate and solve the maze autonomously.

**Created by:** Abdullah Keblawi, Fouad Baloum, Majd Khazen.

## Our Humble Design


## Maze Constraints:

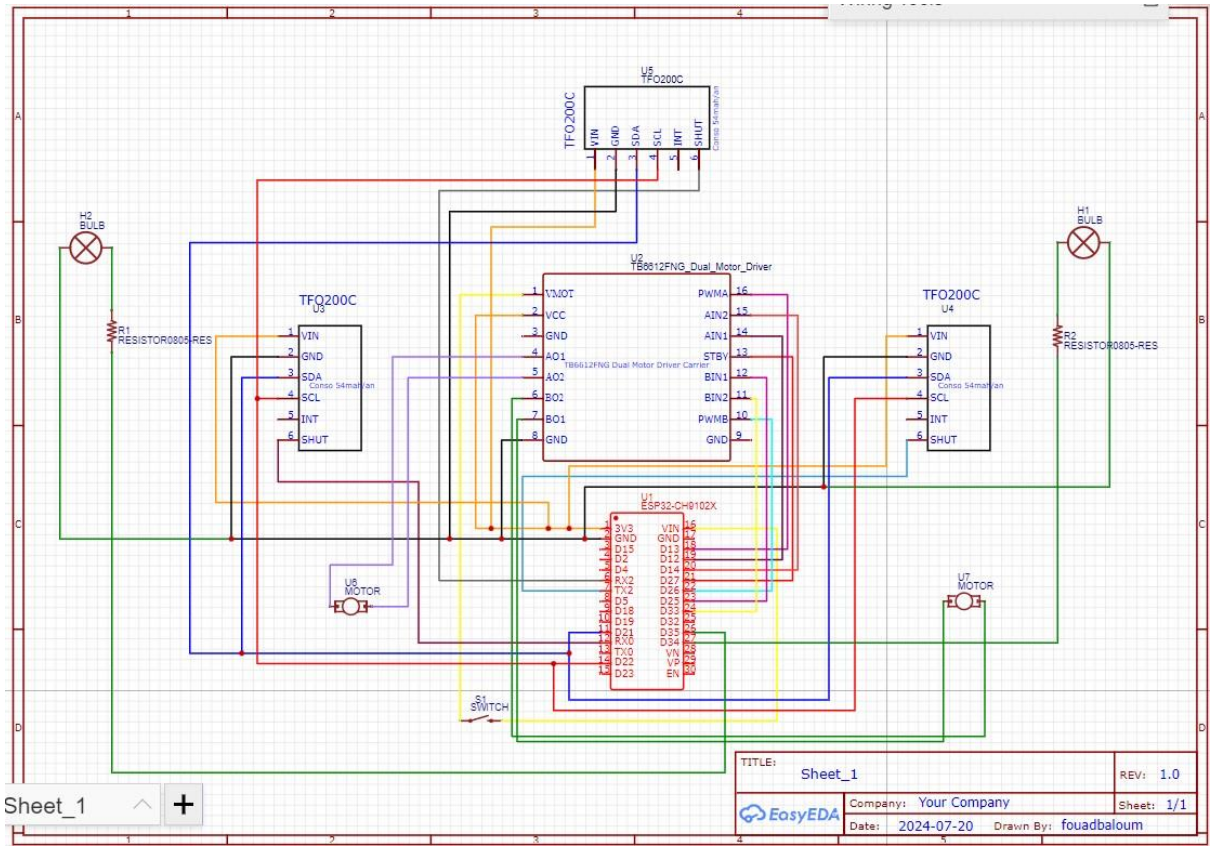
- Block size should be about: 30cmX30cm. (when building irl)
- Minimum Maze size should be 3x3 blocks. (not including edge walls)
- Exactly one starting point, and exactly one goal.
- Start and goal should each be surrounded by 3 walls.
- There should be at least one solution for the maze.
- All paths should be 1 block wide. (meaning there should not be 2x2 Free Blocks)

**Note: GUI will help you abide by most of these constraints.**

## Robot connections:

- ❖ Connections for Sensor 1:
  - VCC to 3.3V
  - GND to GND
  - SDA to GPIO 21 (I2C data line)
  - SCL to GPIO 22 (I2C clock line)
  - XSHUT to RX2 (e.g., GPIO 16)
- ❖ Connections for Sensor 2:
  - VCC to 3.3V
  - GND to GND
  - SDA to GPIO 21 (I2C data line)
  - SCL to GPIO 22 (I2C clock line)
  - XSHUT to a digital pin TX2 (e.g., GPIO 17)
- ❖ Connections for Sensor 3:
  - VCC to 3.3V
  - GND to GND
  - SDA to GPIO 21 (I2C data line)
  - SCL to GPIO 22 (I2C clock line)
  - XSHUT to a digital pin RX0 (e.g., GPIO 03)
- ❖ Motor Connections:
  - VCC - 3.3v
  - VM - Vin
  - GND- GND
  - STBY - D27
  - PMWA - D13
  - AIN1 - D12
  - AIN2 - D14
  - PWMB - D26
  - BIN1 - D25
  - BIN2 - D33

### Circuit Diagram:



## MazeGUI:

**How to run?**

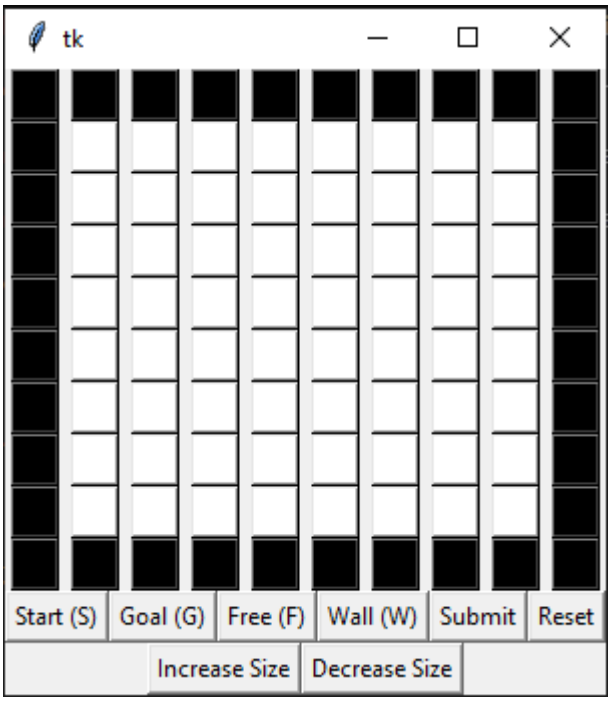
Download the MazeGUI directory, open terminal in that directory and run:

python MazeSolver.py

**Note:**

- You should have python installed on your computer.
- If a package is messing, you can download it by running: pip install <package_name>
- You should also have a service account in google cloud console:
    - It should have access to a google sheet named Transmit in your drive.
    - You should also include in the MazeGui directory a JSON formatted key file for your service account with the name “JSONkey.json”.
    - You can follow a guide for creating a google service account, ChatGPT can also provide you a step-by-step guide.

**Here is how the GUI will look:**




**Buttons:**

- **Start, Goal, Free and Wall:** will help you set the blocks accordingly.
- **Reset:** if you feel that the maze is a mess, or there is a bug with the GUI you can hit the restart button.
- **Submit:** when you have finished building your maze you can hit this button to send it to the google sheet.
- **Increase/Decrease size:** these buttons will help you change the maze size, to match the maze you built.

## MazeRobotRunner:

After downloading this directory, you may open the MazeRunnerRobot.ino file using Arduino IDE, and upload the code to the robot. If you face compilation issues, try using older versions of the libraries required. You may know what libraries are being used when running compilation and getting errors or by simply checking which libraries are imported and downloading them one by one.

**Notes:**

- **comm.h:**

  You will have to create an app script for the google sheet “Transmit”, you may find the code needed for the script app under SheetCode at the Code.gs file in GitHub, insert the script ID in comm.h file in scriptUrl variable.

- **MazeRobotRunnerDEBUG:**

  For debug you will have to also include the following zip libraries which you may download from GitHub:

  - AsyncTCP-master.zip
  - ESPAsyncWebServer-master.zip

  And you will have to insert the solution of the maze  by hand in defs.h file in vector named “turns”.