HOUSE RULES

- Free Wi-Fi
- Phones and laptops on silent
- Coffee, drinks, and snacks in the Learning Hub
- If you're stuck, ask for help :)
- Online? Drop your question in the chat, we've got helpers watching.

WI-FI

SSID: ***"Entelect Guest"***

BUILD-A-BOT WORKSHOP

2025

ZOOSCAPE

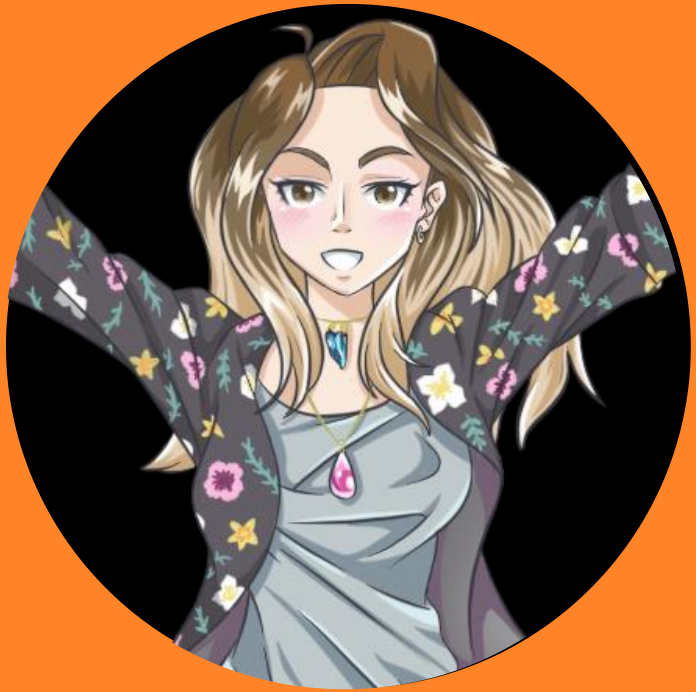
Welcome to the chaos of pellets, cages, and zookeepers

Code along, learn, and maybe… escape.

Build-A-Bot - 21 June 2025

https://l.ead.me/build-a-bot-2025

About me



Jess :)

- Jessica-Bianca Cordier
- Software Engineer
- Website Squad
- Lover of rogue-like, Pokémon, and life sim games
- Steam backlog longer than a novel
- Built a bot to farm an idle game… and then got banned
- https://github.com/JessicaBCordier

https://l.ead.me/build-a-bot-2025

Agenda

01

Intro Presentation

You are here! Introduction and some theory

02

Coding Setup

Get your laptops out and setup those environments!

03

Quick Break

Grab a coffee and get those fingers warmed up!

https://l.ead.me/build-a-bot-2025

Agenda

04

Let’s Build-A-Bot!

Step-by-step coding walkthrough

05

Short Break

Bathroom break and time to catch up and get help

06

More Code!

Carrying on with the bot implementation

07

Pizza & Networking

Grab some food and drink, chat with others and the team!

https://l.ead.me/build-a-bot-2025

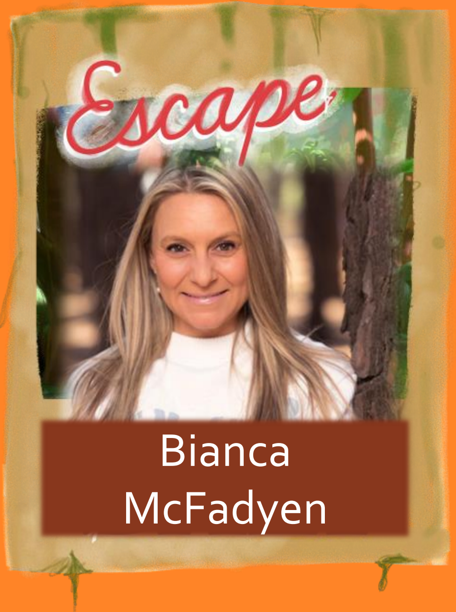
The Team

Meet the Entelect Challenge Team

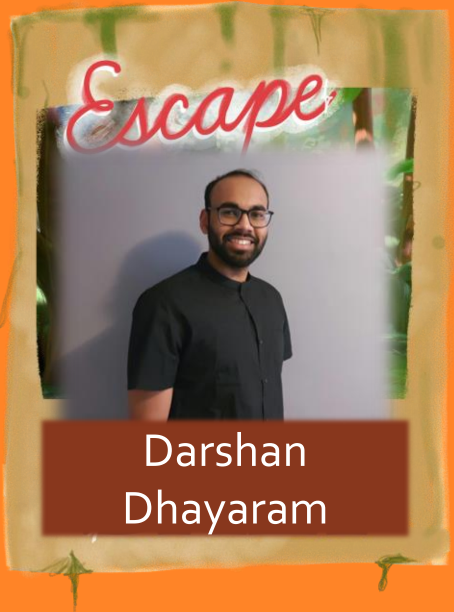
We’re here to help.

No question is too small, too weird, or too broken.

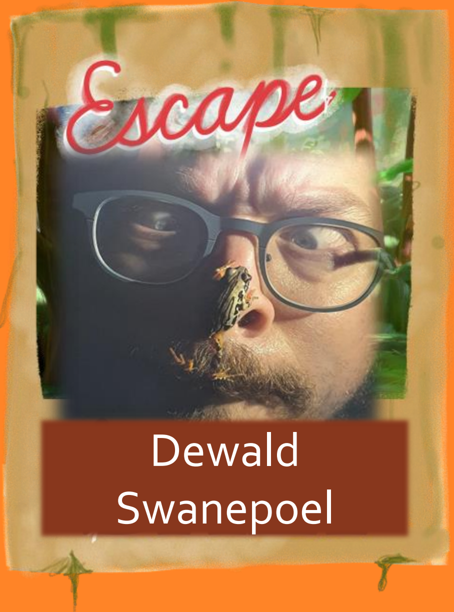
The Johannesburg Squad



Bianca McFadyen



Darshan Dhayaram



Dewald Swanepoel

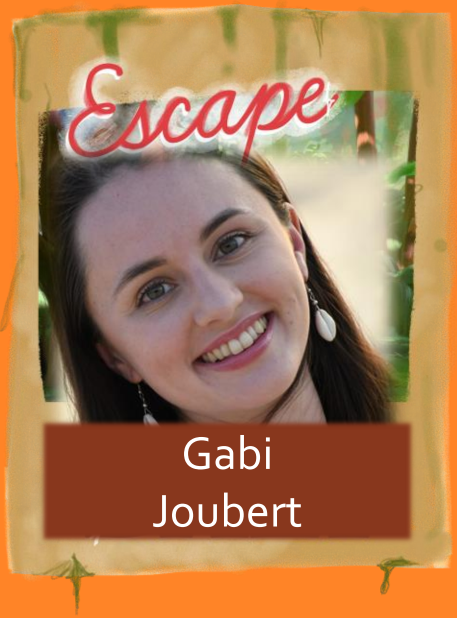
Douglas van Reeuwyk

Francois Greeff

https://l.ead.me/build-a-bot-2025

The Johannesburg Squad

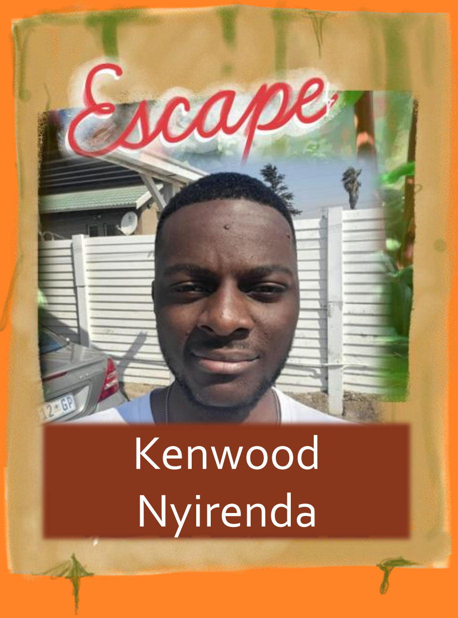
Francois Volschenk



Gabi Joubert

Husnaa Molvi

IB van Schalkwyk



Kenwood Nyirenda

https://l.ead.me/build-a-bot-2025

ENTELECT CHALLENGE 2025

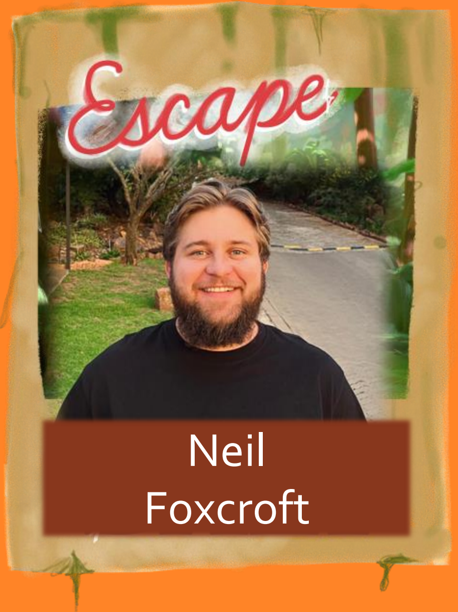
The Johannesburg Squad

Kinesh Pillay

Kumiren Naidoo

Lijani van Wyk de Vries

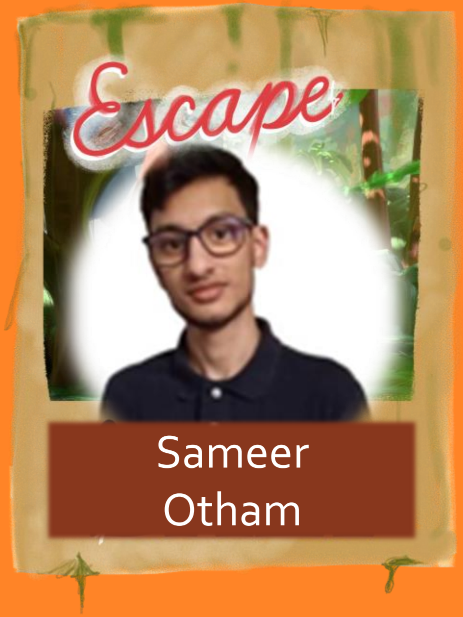
Natasha Draper



Neil Foxcroft

The Johannesburg Squad

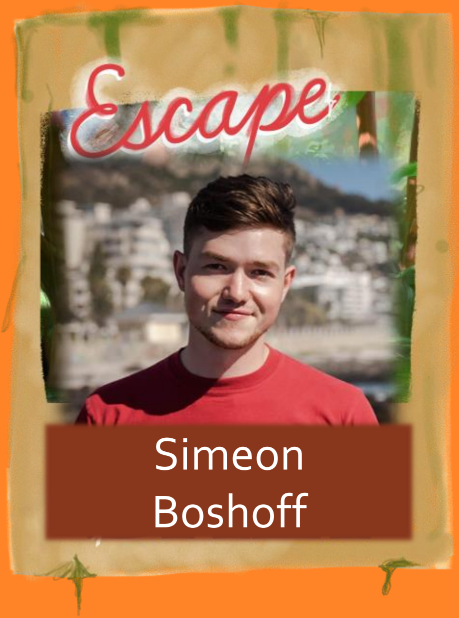
Sibusiso Dlamini



Sameer Otham

The Cape Town Squad

Mark
Lyne



Simeon
Boshoff

How do I join Entelect?

- Graduate programme for final-year students
- Roles include:
  - Software Engineer
  - User Experience Engineer
  - Business Analyst
- Find out more from:
  - https://culture.entelect.co.za
  - Our team in the office

Theoretical Intro

How does this thing even work?!

It’s not magic, unfortunately.

Game Rules

- Your bot is an animal
  - Move: UP, DOWN, LEFT, RIGHT (1 per tick)
  - No diagonals, no collisions with other animals
  - Queue actions, processed in order

- Collect Pellets
  - Scattered across all walkable tiles
  - Earn points by picking them up
  - Ties go to whoever sent the action first

Game Rules

- Avoid the Zookeeper
  - Now up to 4 zookeepers per game
  - Targets the closest viable animal
  - Catches you = score penalty + back to cage (respawn point)
  - Can't use tunnels
- Use the World
  - Tunnels wrap the map
  - Symmetrical maps with walls

Game Rules

- New Mechanics (2025.2.0)
  - Power-Ups:
    - Power Pellet (x10 points)
    - Chameleon Cloak (invisible to zookeepers for 20 ticks)
    - Scavenger (slurp all pellets in 11x11 for 5 ticks)
    - Big Moose Juice (3x pellet score for 5 ticks)
  - Score Streaks: Each consecutive pellet increases score multiplier, up to x4, resets after 3 empty ticks
  - Pellet Respawn

- Win by…
  - Most points when pellets run out or time is up

Technologies Used

- C# and .NET 8
- SignalR for real-time communication
- Git for version control

SignalR Basics

- Asynchronous event-driven messaging system
- Client opens a connection to a “hub” (server)
  - In our case, that’s the game engine’s “/bothub”
- Message consists of a name and optional payload
  - E.g. GameState, BotCommand, Registered, Disconnect

Game Engine Setup

- GitHub: https://github.com/EntelectChallenge/2025-Zooscape
- Docker Desktop required
- Works on Windows, macOS, and Linux
- README file on Repo with instructions
- Run scripts:
  - Windows: .\run.bat
  - macOS/Linux: ./run.sh

Coding Setup

Ready, Set, Code!

Let’s get your environment ready so your bot can join the chaos.

https://l.ead.me/build-a-bot-2025

GitHub Repository

- GitHub: https://github.com/EntelectChallenge/Build-a-Bot-2025
- Branches for each step:
  - step-1, step-2, step-3 etc.
- The repo is a guide, try not to copy and paste!
- If you are stuck with Git, ask a helper for assistance.
- 2023: https://forum.entelect.co.za/t/build-a-bot-workshop-1-july/1605/7
- 2024: https://forum.entelect.co.za/t/build-a-bot-workshop-1-july/1605/7

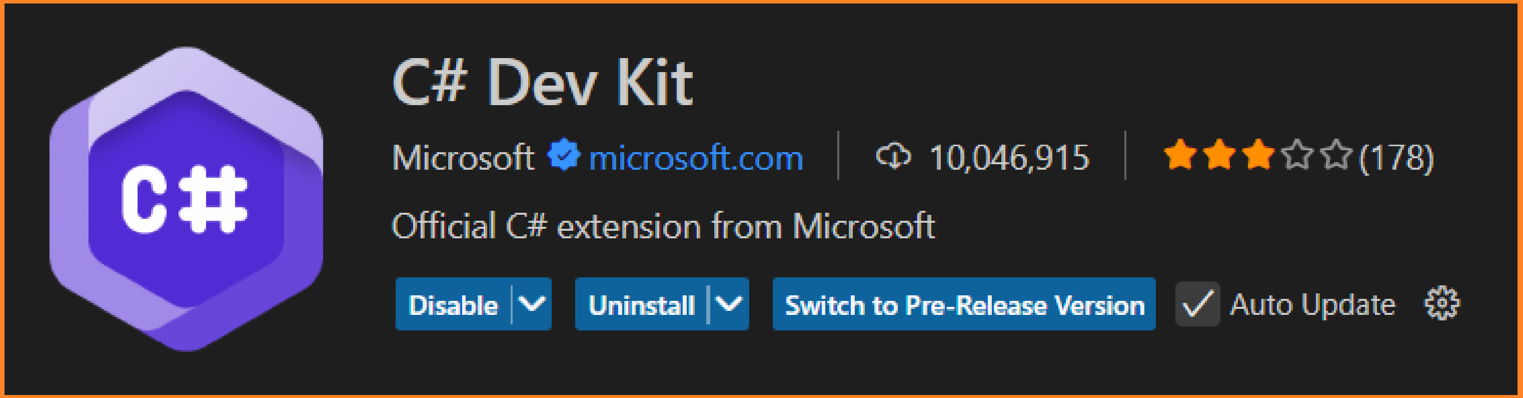
No git? No problem!

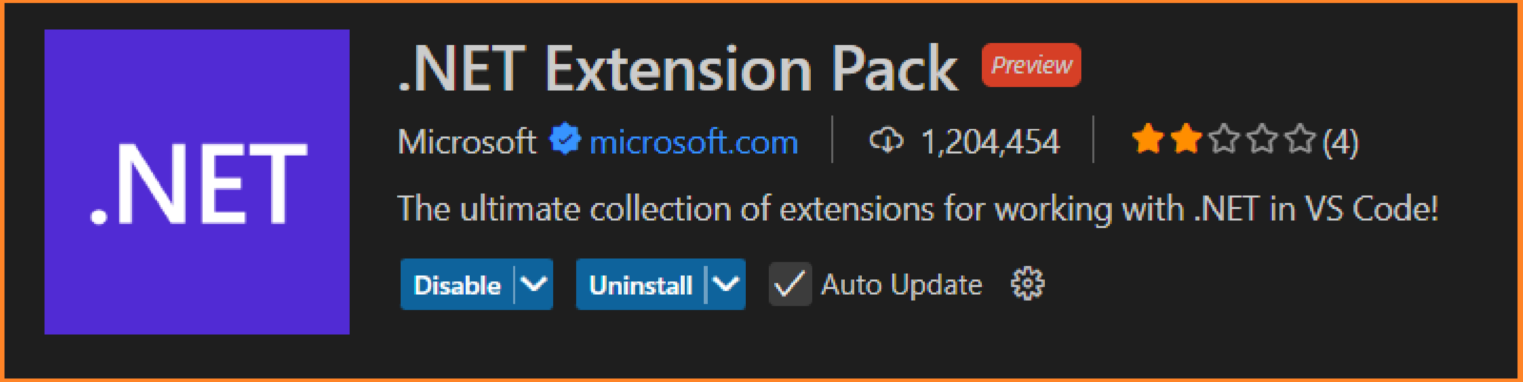
- Zip file: https://github.com/EntelectChallenge/Build-a-Bot-2025/archive/refs/heads/step-3.zip
- Download the file directly and extract it
- Will not have access to the branches and individual steps

Prerequisites

- We will be using Visual Studio Code for the presentation
  - You may use whichever IDE you wish to
- .NET SDK 8 is required

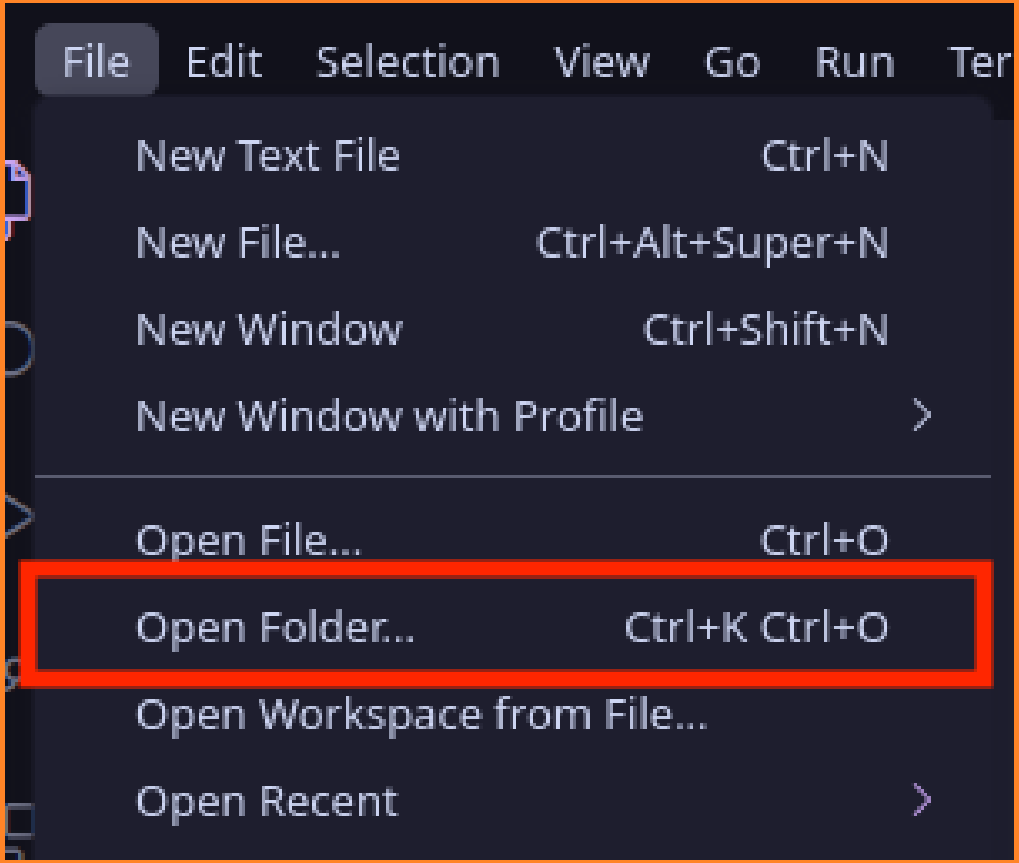
Visual Studio Code Extensions

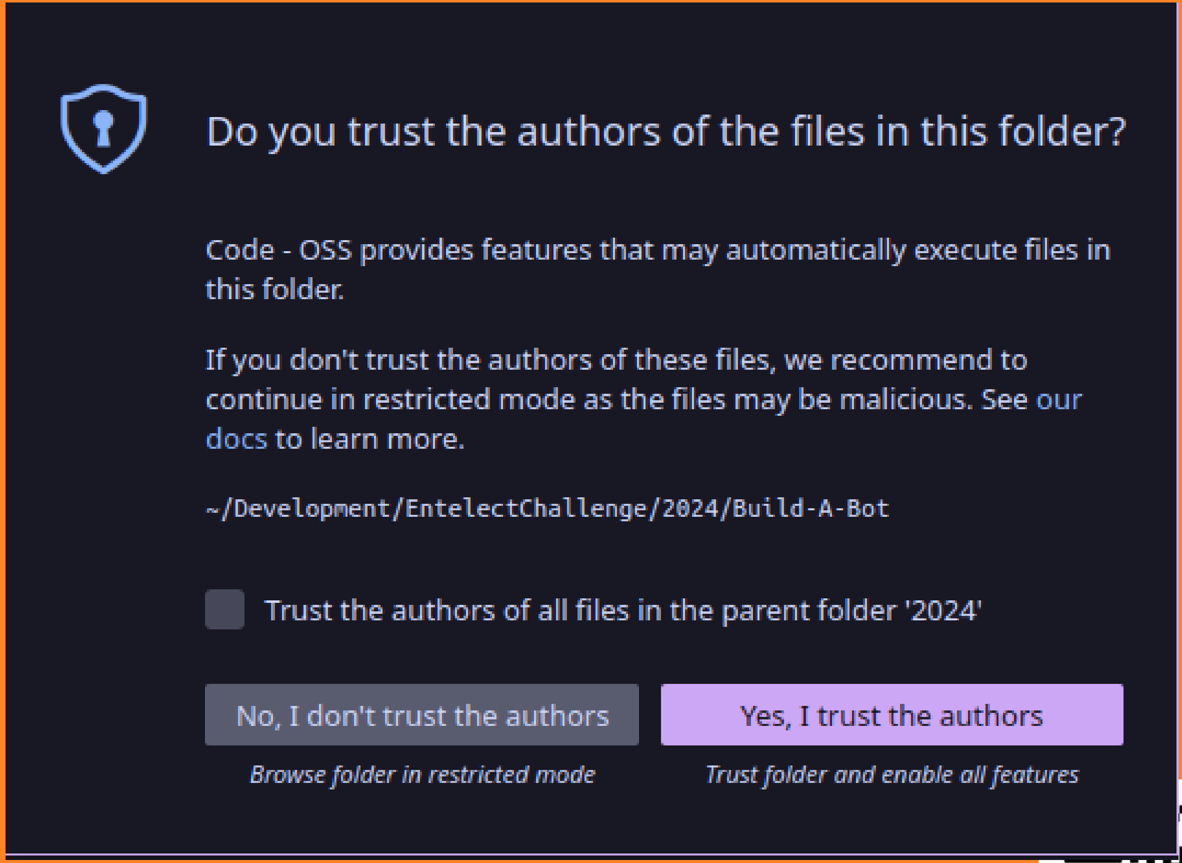

C# Dev Kit
Microsoft microsoft.com | 10,046,915 | ★★★☆☆ (178)
Official C# extension from Microsoft


.NET Extension Pack *Preview*
Microsoft microsoft.com | 1,204,454 | ★★☆☆☆ (4)
The ultimate collection of extensions for working with .NET in VS Code!

https://l.ead.me/build-a-bot-2025

Creating the project





Do you trust the authors of the files in this folder?

Code - OSS provides features that may automatically execute files in this folder.

If you don't trust the authors of these files, we recommend to continue in restricted mode as the files may be malicious. See our docs to learn more.

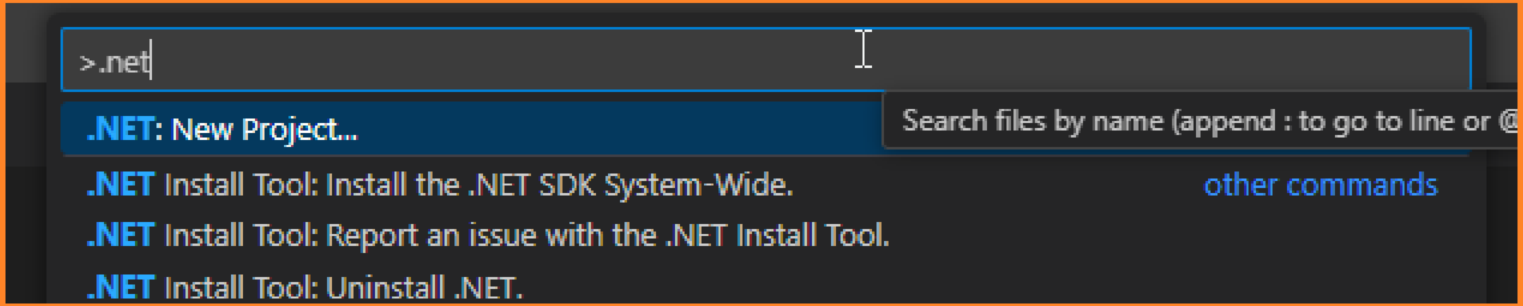
```
~/Development/EntelectChallenge/2024/Build-A-Bot
```

☐ Trust the authors of all files in the parent folder '2024'

No, I don't trust the authors
Browse folder in restricted mode
Yes, I trust the authors
Trust folder and enable all features

https://l.ead.me/build-a-bot-2025

Creating the project



```
>.net
```

.NET: New Project...

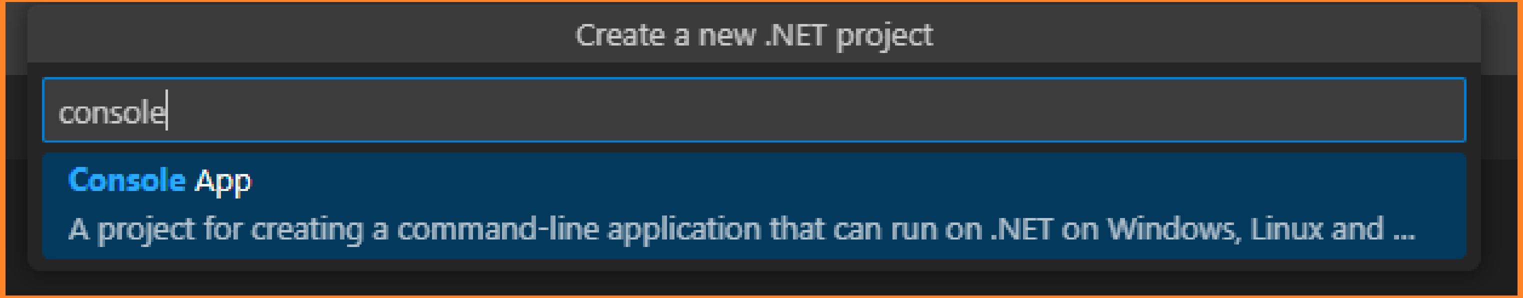
Search files by name (append : to go to line or @

.NET Install Tool: Install the .NET SDK System-Wide.

other commands

.NET Install Tool: Report an issue with the .NET Install Tool.

.NET Install Tool: Uninstall .NET.

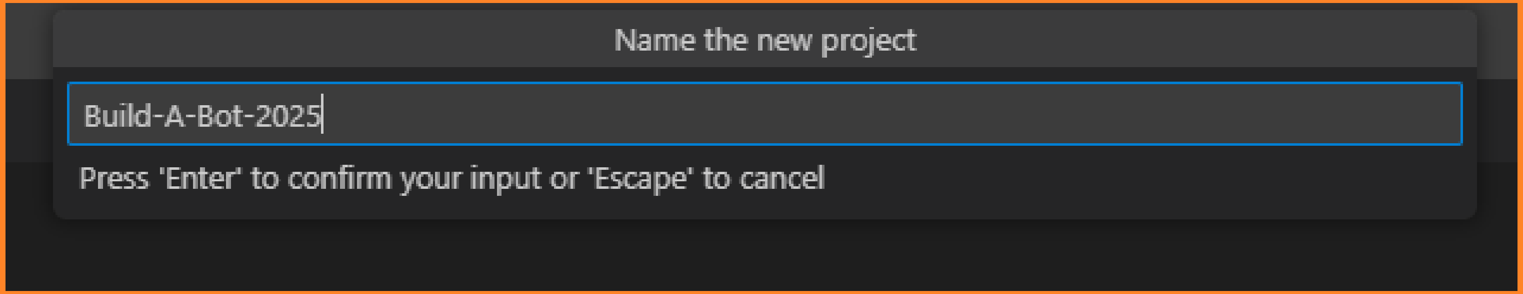


Create a new .NET project

```
console
```

Console App

A project for creating a command-line application that can run on .NET on Windows, Linux and ...

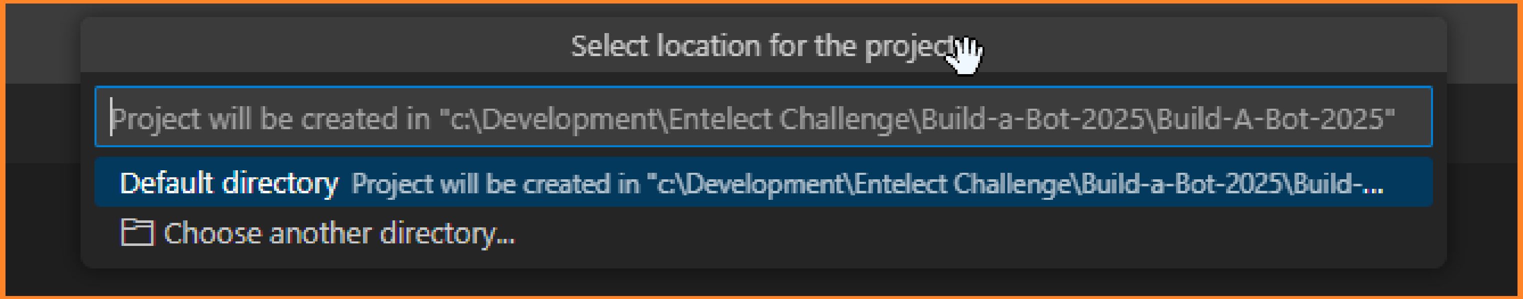


Name the new project

```
Build-A-Bot-2025
```

Press 'Enter' to confirm your input or 'Escape' to cancel

Creating the project

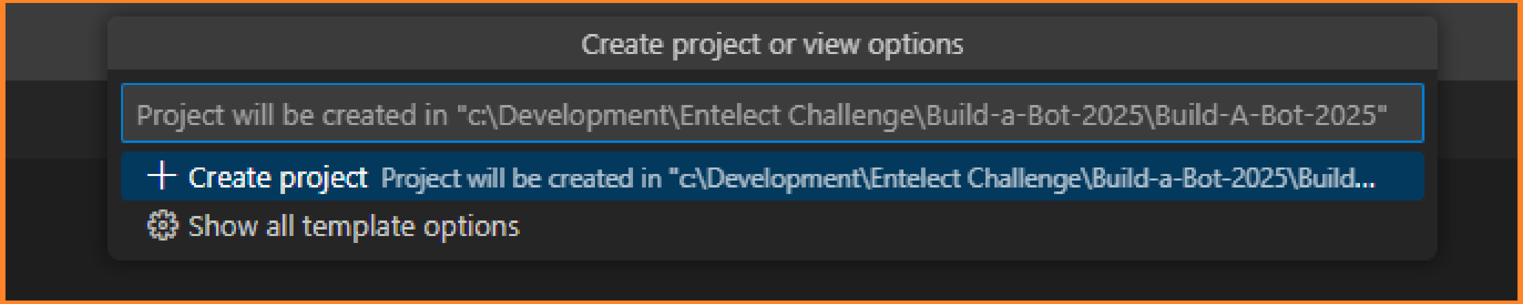


Select location for the project

Project will be created in "c:\Development\Entelect Challenge\Build-a-Bot-2025\Build-A-Bot-2025"

Default directory Project will be created in "c:\Development\Entelect Challenge\Build-a-Bot-2025\Build-...

Choose another directory...



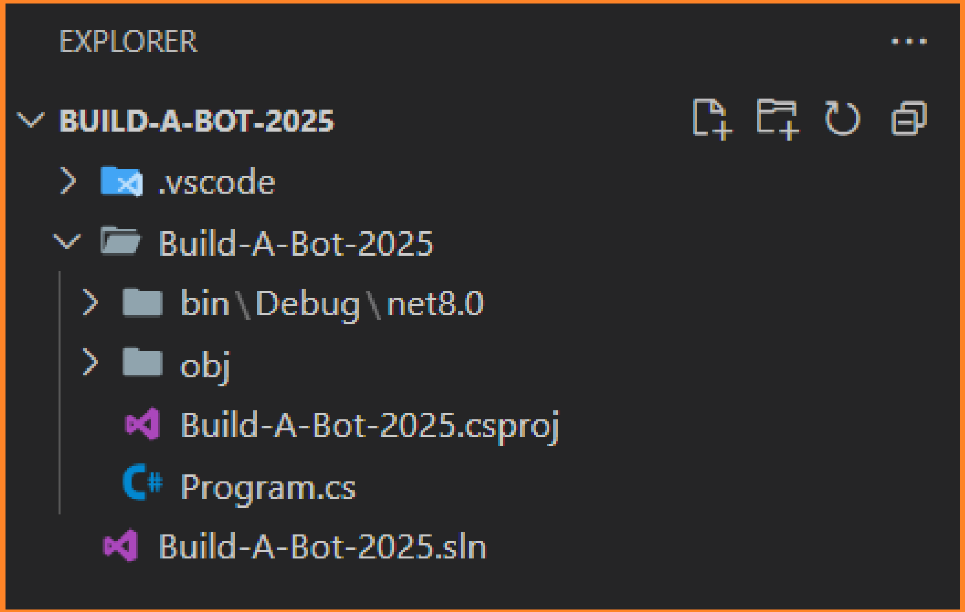
Create project or view options

Project will be created in "c:\Development\Entelect Challenge\Build-a-Bot-2025\Build-A-Bot-2025"

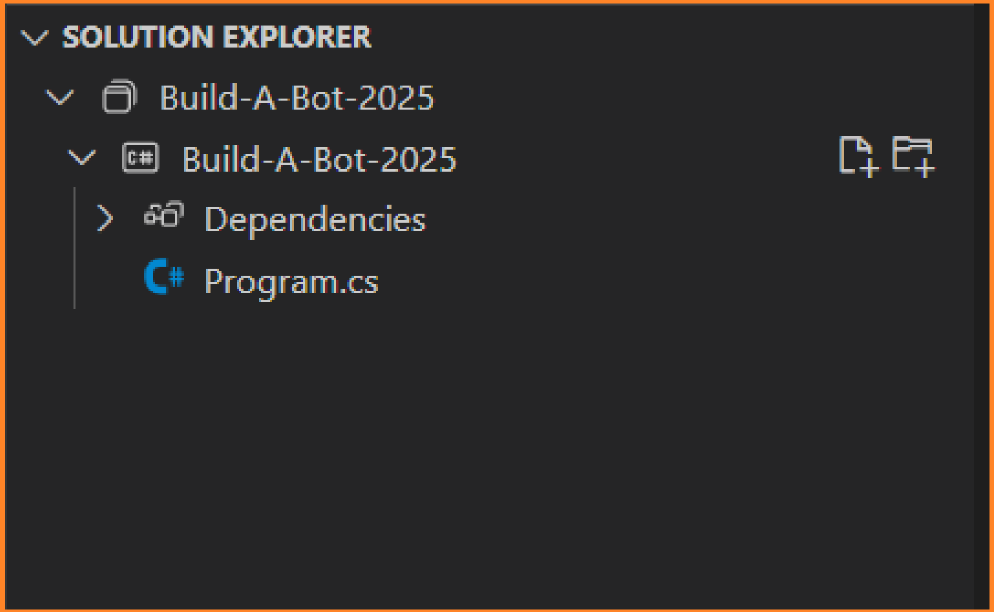
+ Create project Project will be created in "c:\Development\Entelect Challenge\Build-a-Bot-2025\Build...

Show all template options

Creating the project



```
EXPLORER
BUILD-A-BOT-2025
  .vscode
  Build-A-Bot-2025
    bin\Debug\net8.0
    obj
    Build-A-Bot-2025.csproj
    Program.cs
  Build-A-Bot-2025.sln
```



```
SOLUTION EXPLORER
  Build-A-Bot-2025
    Build-A-Bot-2025
      Dependencies
      Program.cs
```

https://l.ead.me/build-a-bot-2025

Coffee Time!

Catch-up, questions, and caffeine

Helpers will be available to assist you with project setup.

Ready, set, code!

Or just listen, completely up to you.

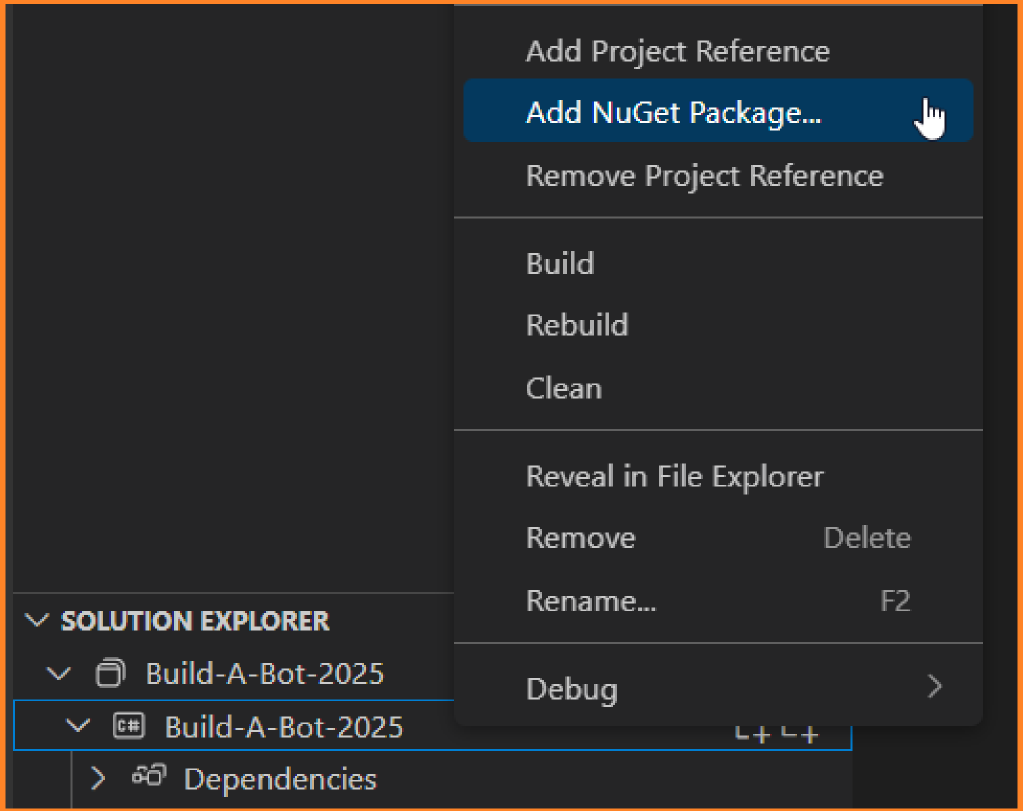
We'll go step by step and there will be time to catch up between parts

https://l.ead.me/build-a-bot-2025

Workshop Structure

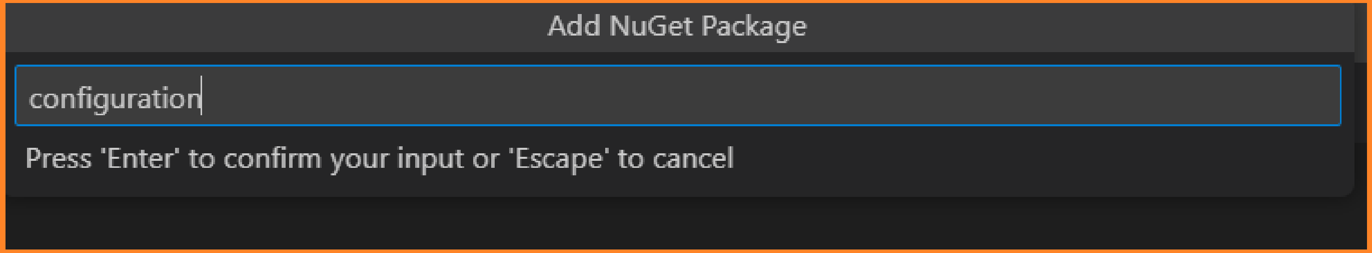
- Each step = one Github branch
- I'll explain what changed, and why
- You can code along, or just follow visually
- Helpers will be around after each step for help/advice
- Then we will proceed to the next step

Step 1: Adding Packages



Add Project Reference
Add NuGet Package...
Remove Project Reference
Build
Rebuild
Clean
Reveal in File Explorer
Remove Delete
Rename... F2
Debug

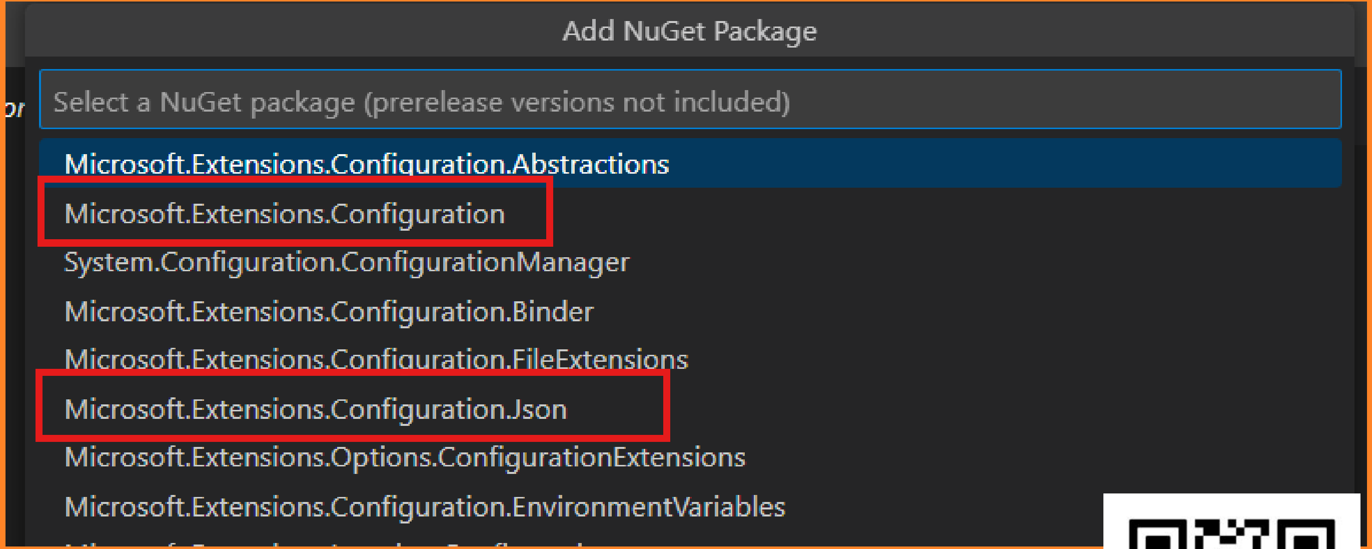
SOLUTION EXPLORER
Build-A-Bot-2025
Build-A-Bot-2025
Dependencies



Add NuGet Package

configuration

Press 'Enter' to confirm your input or 'Escape' to cancel

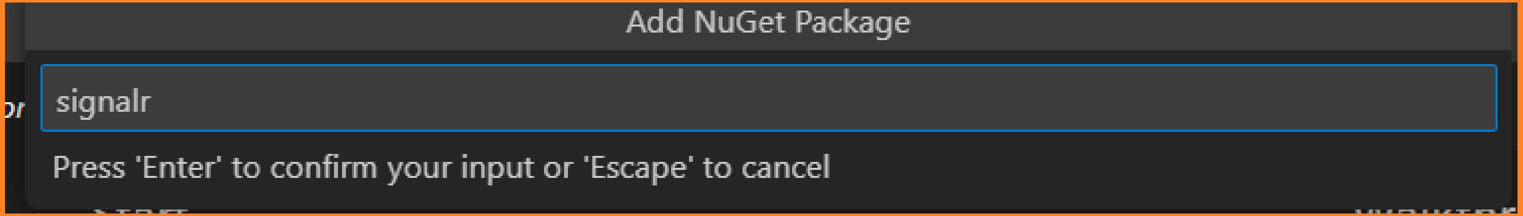


Add NuGet Package

Select a NuGet package (prerelease versions not included)

Microsoft.Extensions.Configuration.Abstractions
Microsoft.Extensions.Configuration
System.Configuration.ConfigurationManager
Microsoft.Extensions.Configuration.Binder
Microsoft.Extensions.Configuration.FileExtensions
Microsoft.Extensions.Configuration.Json
Microsoft.Extensions.Options.ConfigurationExtensions
Microsoft.Extensions.Configuration.EnvironmentVariables

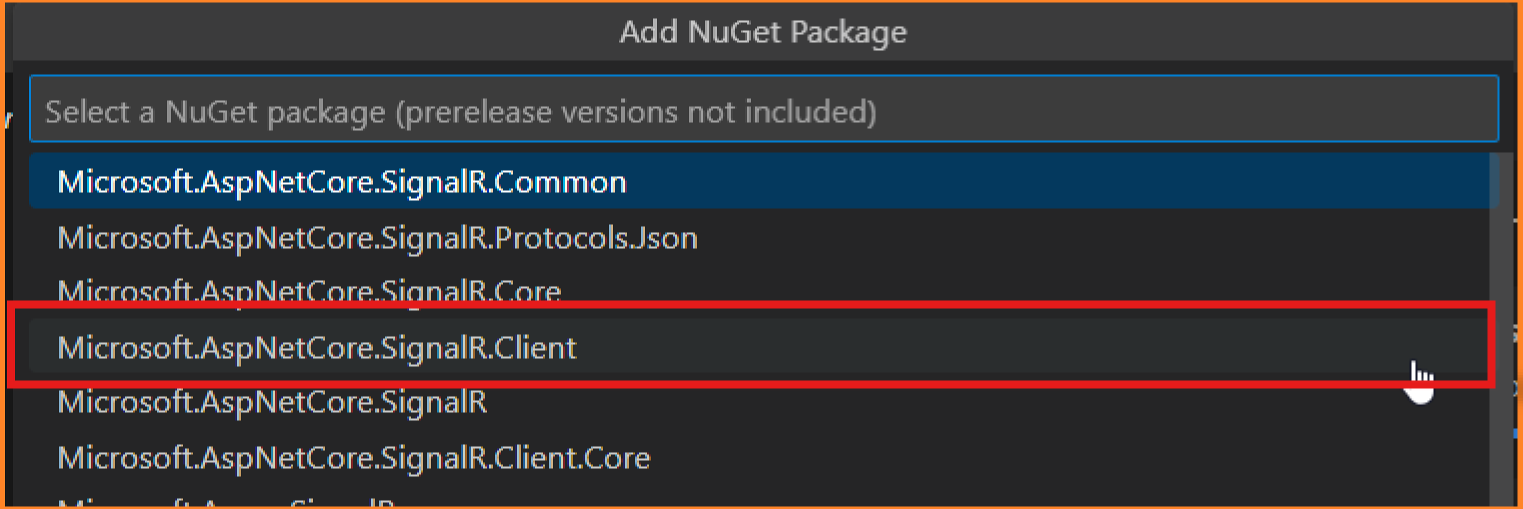
Step 1: Adding Packages



Add NuGet Package

signalr

Press 'Enter' to confirm your input or 'Escape' to cancel

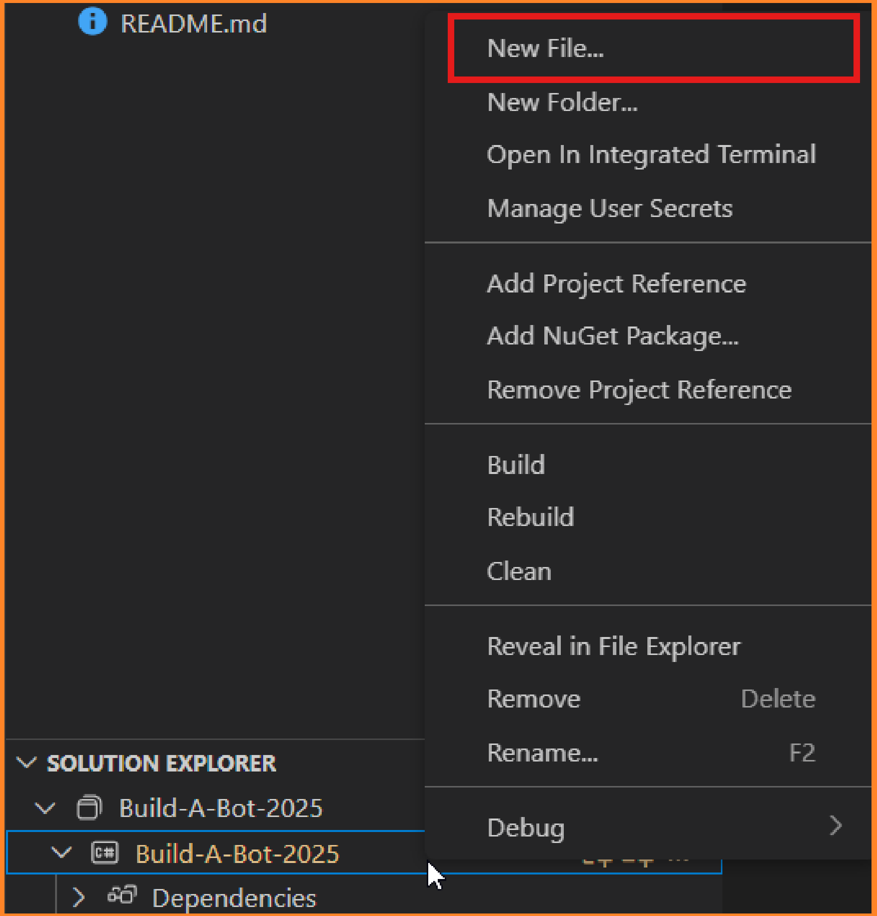


Add NuGet Package

Select a NuGet package (prerelease versions not included)

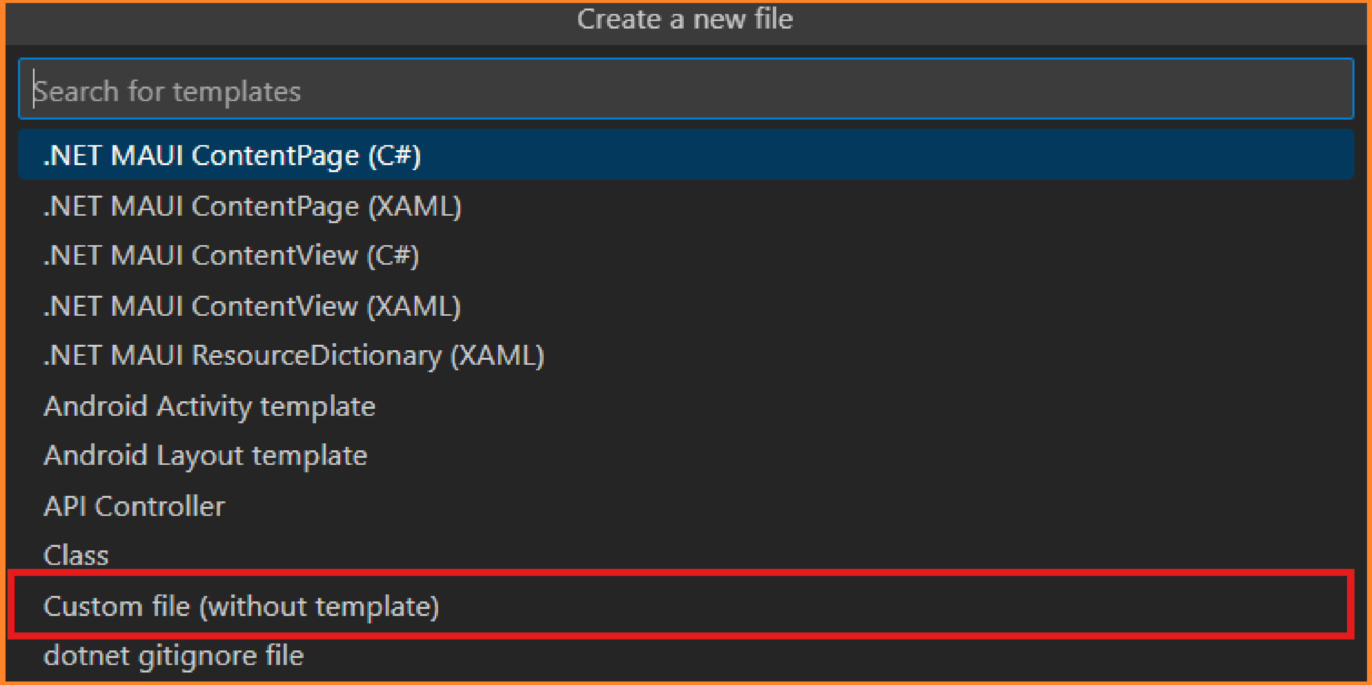
Microsoft.AspNetCore.SignalR.Common
Microsoft.AspNetCore.SignalR.Protocols.Json
Microsoft.AspNetCore.SignalR.Core
Microsoft.AspNetCore.SignalR.Client
Microsoft.AspNetCore.SignalR
Microsoft.AspNetCore.SignalR.Client.Core

Step 1: Adding Config


README.md

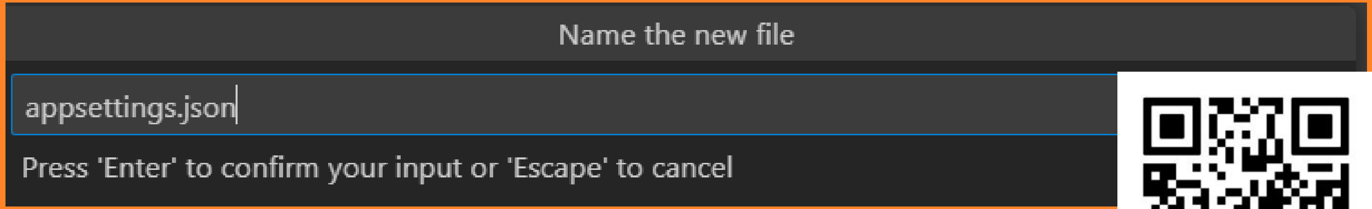
New File...
New Folder...
Open In Integrated Terminal
Manage User Secrets
Add Project Reference
Add NuGet Package...
Remove Project Reference
Build
Rebuild
Clean
Reveal in File Explorer
Remove Delete
Rename... F2
Debug

SOLUTION EXPLORER
Build-A-Bot-2025
Build-A-Bot-2025
Dependencies


Create a new file

Search for templates

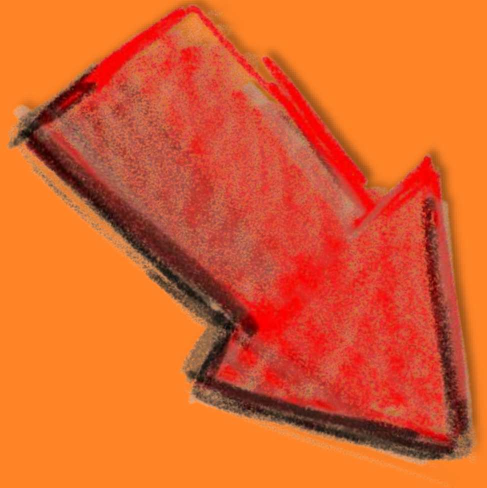
.NET MAUI ContentPage (C#)
.NET MAUI ContentPage (XAML)
.NET MAUI ContentView (C#)
.NET MAUI ContentView (XAML)
.NET MAUI ResourceDictionary (XAML)
Android Activity template
Android Layout template
API Controller
Class
Custom file (without template)
dotnet gitignore file


Name the new file

appsettings.json

Press 'Enter' to confirm your input or 'Escape' to cancel

https://l.ead.me/build-a-bot-2025

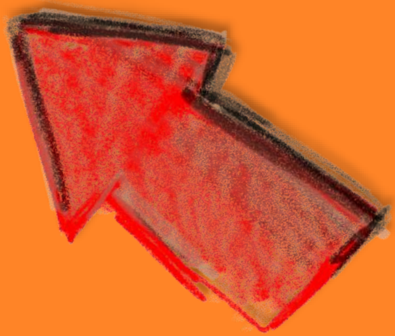


Step 1: Adding Config

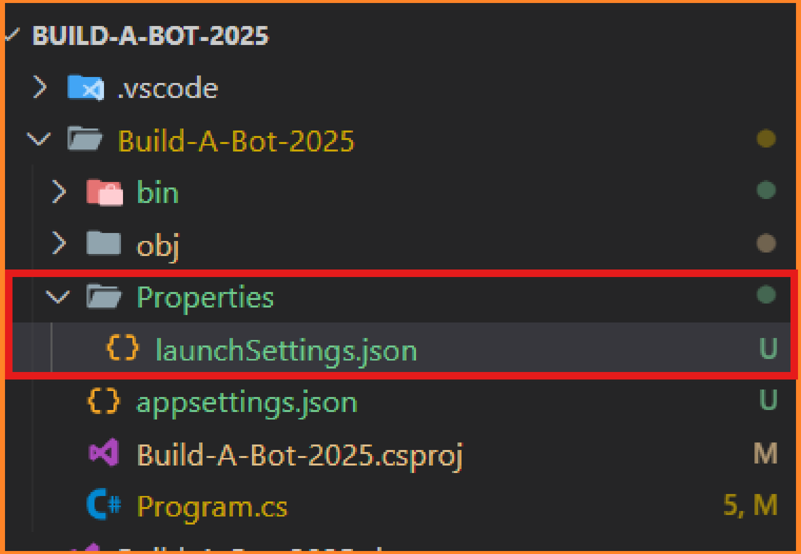
```json
{
    "RunnerIP": "http://localhost",
    "RunnerPort": "5000",
    "BotNickname": "R2D2"
}
```

Step 1: Manually Adding to Output

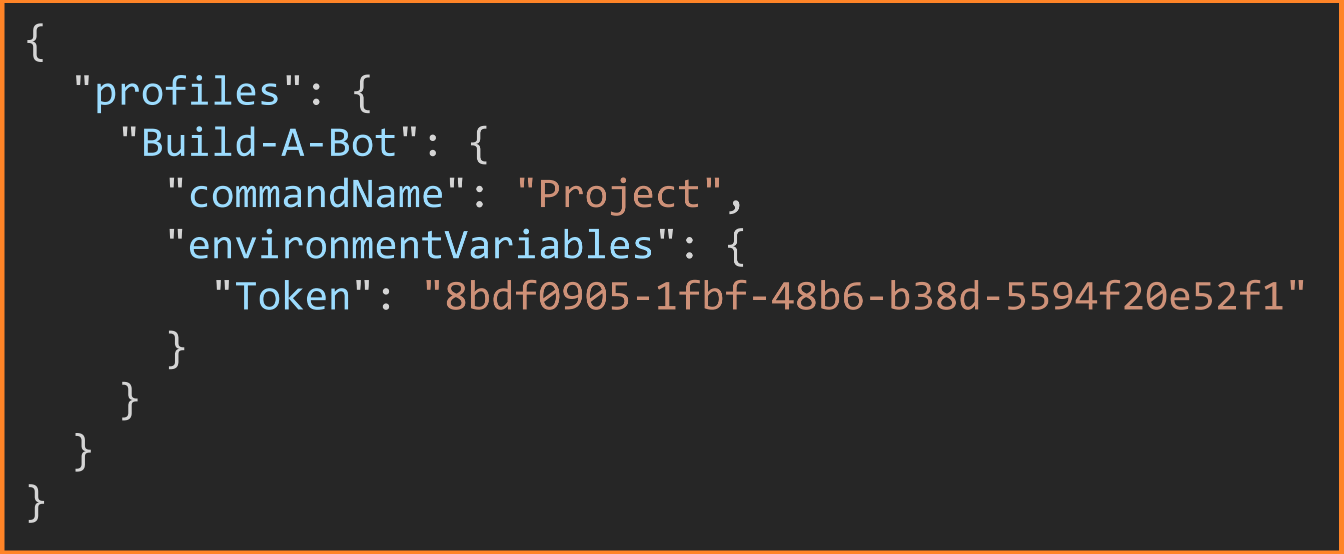
```
<ItemGroup>
  <None Update="appsettings.json">
    <CopyToOutputDirectory>PreserveNewest
     </CopyToOutputDirectory>
  </None>
</ItemGroup>
```



Step 1: Adding Config



```
BUILD-A-BOT-2025
> .vscode
v Build-A-Bot-2025
  > bin
  > obj
  v Properties
      {} launchSettings.json        U
    {} appsettings.json             U
    Build-A-Bot-2025.csproj         M
    Program.cs                   5, M
```

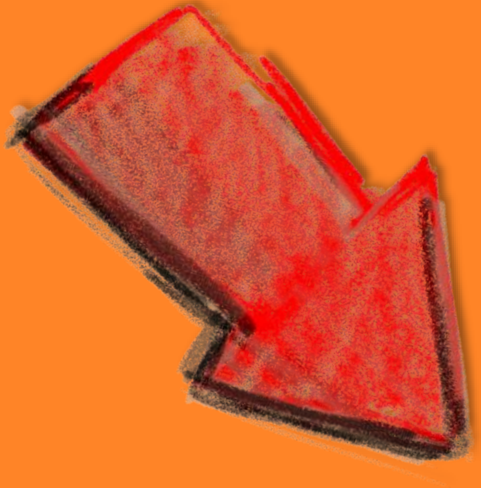


```
{
  "profiles": {
    "Build-A-Bot": {
      "commandName": "Project",
      "environmentVariables": {
        "Token": "8bdf0905-1fbf-48b6-b38d-5594f20e52f1"
      }
    }
  }
}
```

https://www.uuidgenerator.net/version4

Step 1: Reading Config

```csharp
using Microsoft.AspNetCore.SignalR.Client;
using Microsoft.Extensions.Configuration;
using Microsoft.Extensions.Logging;

public class Program
{
    public static async Task Main(string[] args)
    {
        // We'll set everything up in here
    }
}
```



Step 1: Reading Config

```
// Load configuration from appsettings.json
var builder = new ConfigurationBuilder().AddJsonFile("appsettings.json",
optional: false);
var configuration = builder.Build();
```

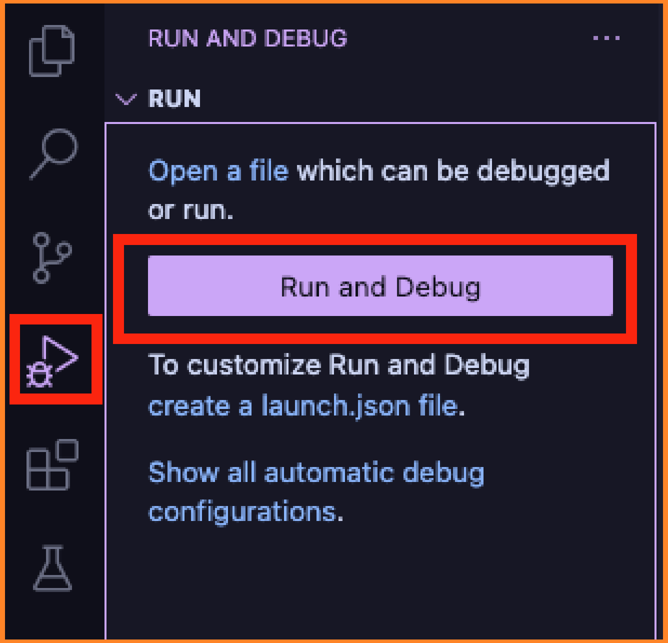
Step 1: Reading Environment Variables

```
// Read values from config or environment
var ip = Environment.GetEnvironmentVariable("RUNNER_IPV4") ??
configuration["RunnerIP"];
if (!ip.StartsWith("http://")) ip = $"http://{ip}";

var port = configuration["RunnerPort"];
var nickname = Environment.GetEnvironmentVariable("BOT_NICKNAME") ??
configuration["BotNickname"];
var token = Environment.GetEnvironmentVariable("Token") ??
Environment.GetEnvironmentVariable("REGISTRATION_TOKEN");
```

Step 1: Creating the Connection

```
var url = $"{ip}:{port}/bothub";

// Create SignalR connection
var connection = new HubConnectionBuilder()
    .WithUrl(url)
    .ConfigureLogging(logging => logging.SetMinimumLevel(LogLevel.Debug))
    .WithAutomaticReconnect()
    .Build();

// Connect!
await connection.StartAsync();
Console.WriteLine("Connected to Runner");
```

Step 1: Running the bot


RUN AND DEBUG

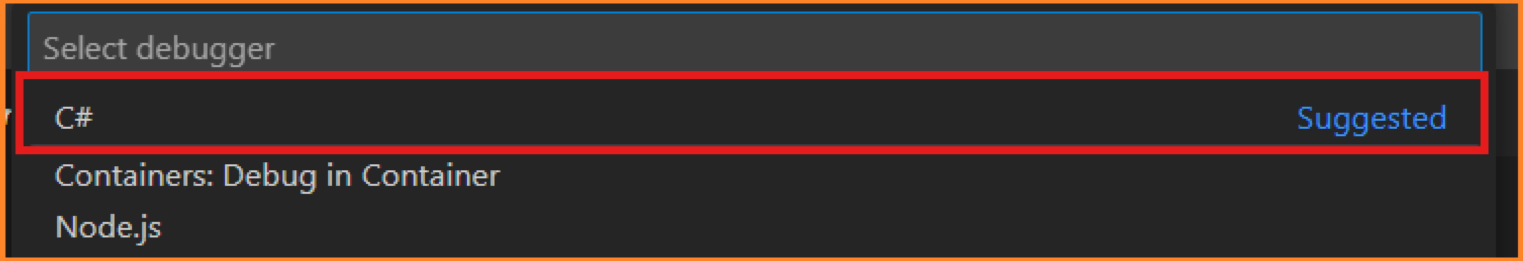
RUN

Open a file which can be debugged or run.

Run and Debug

To customize Run and Debug create a launch.json file.

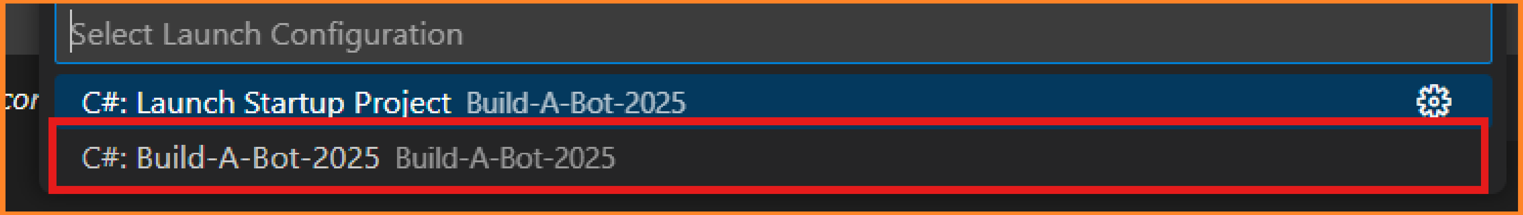
Show all automatic debug configurations.


Select debugger

C# Suggested

Containers: Debug in Container

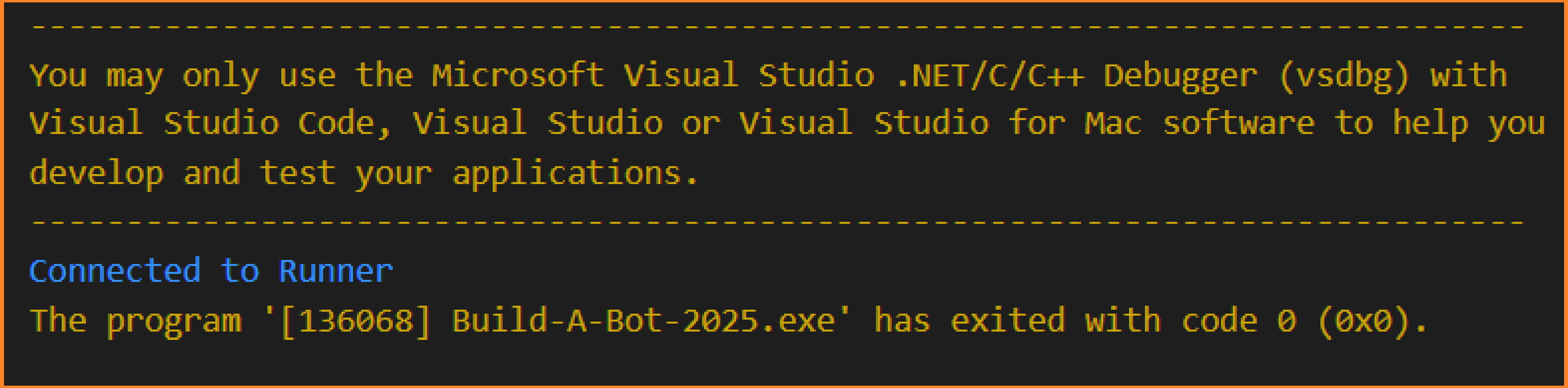
Node.js


Select Launch Configuration

C#: Launch Startup Project Build-A-Bot-2025

C#: Build-A-Bot-2025 Build-A-Bot-2025

Step 1: Results!



```
-------------------------------------------------------------------------------
You may only use the Microsoft Visual Studio .NET/C/C++ Debugger (vsdbg) with
Visual Studio Code, Visual Studio or Visual Studio for Mac software to help you
develop and test your applications.
-------------------------------------------------------------------------------
Connected to Runner
The program '[136068] Build-A-Bot-2025.exe' has exited with code 0 (0x0).
```

Coffee Time!

Catch-up, questions, and caffeine

Helpers will be available to assist you with project setup.

https://l.ead.me/build-a-bot-2025

Step 2: Enums

```
namespace BuildABot2025.Enums;

public enum BotAction
{
    Up = 1,
    Down = 2,
    Left = 3,
    Right = 4,
    UseItem = 5,
}
```

Step 2: Enums

```
namespace BuildABot2025.Enums;

public enum CellContent
{
    Empty = 0,
    Wall = 1,
    Pellet = 2,
    ZookeeperSpawn = 3,
    AnimalSpawn = 4,
    PowerPellet = 5,
    ChameleonCloak = 6,
    Scavenger = 7,
    BigMooseJuice = 8,
}
```

Step 2: Enums

```
namespace BuildABot2025.Enums;

public enum PowerUpType
{
    PowerPellet = 0,
    ChameleonCloak = 1,
    Scavenger = 2,
    BigMooseJuice = 3,
}
```

Step 2: Models

```
using BuildABot2025.Enums;

namespace BuildABot2025.Models;

public class ActivePowerUp
{
    public double Value { get; set; }
    public int TicksRemaining { get; set; }
    public PowerUpType Type { get; set; }
}
```

Step 2: Models

```
using BuildABot2025.Enums;

namespace BuildABot2025.Models;

public class Animal
{
    public Guid Id { get; set; }
    public int X { get; set; }
    public int Y { get; set; }
    public int SpawnX { get; set; }
    public int SpawnY { get; set; }
    public int Score { get; set; }
    public int CapturedCounter { get; set; }
    public int DistanceCovered { get; set; }
    public bool IsViable { get; set; }
    public PowerUpType? HeldPowerUp { get; set; }
    public ActivePowerUp? ActivePowerUp { get; set; }
}
```

Step 2: Models

```
using BuildABot2025.Enums;

namespace BuildABot2025.Models;

public class BotCommand
{
    public BotAction Action { get; set; }
}
```

Step 2: Models

```csharp
using BuildABot2025.Enums;

namespace BuildABot2025.Models;

public class Cell
{
    public int X { get; set; }
    public int Y { get; set; }
    public CellContent Content { get; set; }
}
```

Step 2: Models

```
namespace BuildABot2025.Models;

public class GameState
{
    public DateTime TimeStamp { get; set; }
    public int Tick { get; set; }
    public List<Cell> Cells { get; set; }
    public List<Animal> Animals { get; set; }
    public List<Zookeeper> Zookeepers { get; set; }
}
```

Step 2: Models

```
namespace BuildABot2025.Models;

public class Zookeeper
{
    public Guid Id { get; set; }
    public int X { get; set; }
    public int Y { get; set; }
    public int SpawnX { get; set; }
    public int SpawnY { get; set; }
}
```

Step 2: Create Bot Service

```csharp
using BuildABot2025.Enums;
using BuildABot2025.Models;

namespace BuildABot2025.Services;

public class BotService
{
    private Guid _botId;
    public void SetBotId(Guid botId)
    {
        _botId = botId;
    }
    public Guid GetBotId()
    {
        return _botId;
    }
}
```

Step 2: Command Handling & Printing

```
public BotCommand ProcessState(GameState gameState)
{
    var bot = gameState.Animals.FirstOrDefault(a => a.Id == _botId);
    var command = new BotCommand
    {
        Action = BotAction.Right // placeholder logic
    };

    if (bot != null)
    {
        Console.WriteLine($"Tick: {gameState.Tick}");
        Console.WriteLine($"Bot Position: ({bot.X}, {bot.Y})");
        Console.WriteLine($"Planned Action: {command.Action}");
    }

    return command;
}
```

Step 2: Setup BotService & Event Handlers

```csharp
var botService = new BotService();
BotCommand? botCommand = new BotCommand();

// Register event handlers
connection.On<Guid>("Registered", id => botService.SetBotId(id));

connection.On<string>("Disconnect", async reason =>
{
    Console.WriteLine($"Server sent disconnect: {reason}");
    await connection.StopAsync();
});
```

https://l.ead.me/build-a-bot-2025

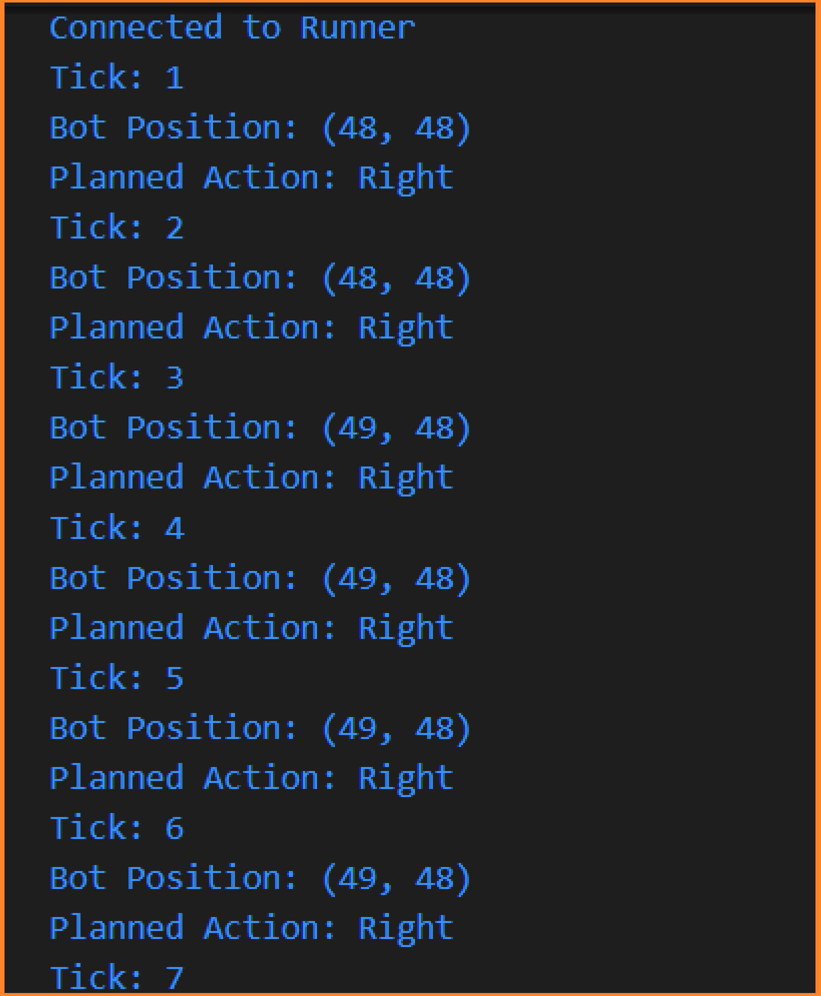
Step 2: Setup BotService & Event Handlers

```
connection.On<GameState>("GameState", gamestate =>
{
    botCommand = botService.ProcessState(gamestate);
});

// Handle disconnection
connection.Closed += (error) =>
{
    Console.WriteLine($"Connection closed: {error?.Message}");
    return Task.CompletedTask;
};
```

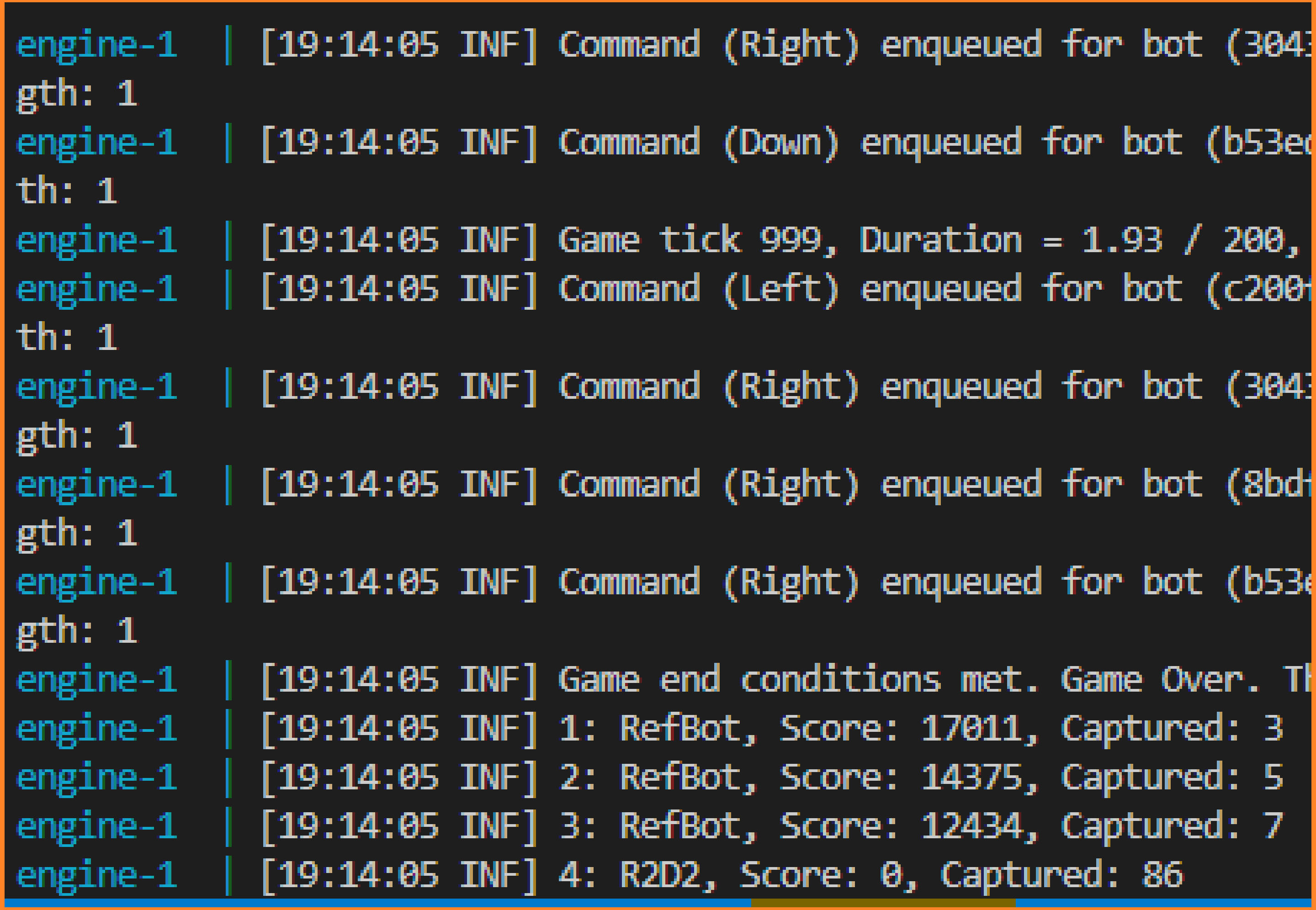
Step 2: Register & Send Commands

```
// Start connection
await connection.StartAsync();
Console.WriteLine("Connected to Runner");

// Register the bot
await connection.InvokeAsync("Register", token, nickname);

// Main game loop
while (connection.State == HubConnectionState.Connected)
{
    if (botCommand == null || (int)botCommand.Action is < 1 or > 5)
        continue;

    await connection.SendAsync("BotCommand", botCommand);
    botCommand = null;
}
```

Step 2: More Results!



```
Connected to Runner
Tick: 1
Bot Position: (48, 48)
Planned Action: Right
Tick: 2
Bot Position: (48, 48)
Planned Action: Right
Tick: 3
Bot Position: (49, 48)
Planned Action: Right
Tick: 4
Bot Position: (49, 48)
Planned Action: Right
Tick: 5
Bot Position: (49, 48)
Planned Action: Right
Tick: 6
Bot Position: (49, 48)
Planned Action: Right
Tick: 7
```



```
engine-1  | [19:14:05 INF] Command (Right) enqueued for bot (304
gth: 1
engine-1  | [19:14:05 INF] Command (Down) enqueued for bot (b53e
th: 1
engine-1  | [19:14:05 INF] Game tick 999, Duration = 1.93 / 200,
engine-1  | [19:14:05 INF] Command (Left) enqueued for bot (c200
th: 1
engine-1  | [19:14:05 INF] Command (Right) enqueued for bot (304
gth: 1
engine-1  | [19:14:05 INF] Command (Right) enqueued for bot (8bd
gth: 1
engine-1  | [19:14:05 INF] Command (Right) enqueued for bot (b53
gth: 1
engine-1  | [19:14:05 INF] Game end conditions met. Game Over. T
engine-1  | [19:14:05 INF] 1: RefBot, Score: 17011, Captured: 3
engine-1  | [19:14:05 INF] 2: RefBot, Score: 14375, Captured: 5
engine-1  | [19:14:05 INF] 3: RefBot, Score: 12434, Captured: 7
engine-1  | [19:14:05 INF] 4: R2D2, Score: 0, Captured: 86
```

Coffee Time!

Catch-up, questions, and caffeine

Helpers will be available to assist you with any questions.

Understanding the Game State

- What are we working with?
- GameState: Snapshot of the game each tick
  - Tick: The current turn count
  - Animals: List of all bots (including yours)
  - Cells: The map grid, what is in each tile
  - Zookeepers: Moving threats on the map
- Each cell has:
  - X, Y: Position
  - Content: Can be Empty, Pellet, Wall etc.

Step 3: Start Simple – Find the Closest Pellet

- Let’s replace our placeholder logic with basic goal seeking
- Still moves right for now
- But is aware of where the nearest pellet is
- Adds foundation for smarter decisions
- Filter all cells that contain pellets
- Sort them by Manhattan distance from the bot
- Choose the nearest one as the target

Step 3: Start Simple
Find the Closest Pellet

```csharp
public BotCommand ProcessState(GameState gameState)
{
    var bot = gameState.Animals.FirstOrDefault(a => a.Id == _botId);
    var command = new BotCommand { Action = BotAction.Right }; // fallback
    if (bot == null)
        return command;

    var pellets = gameState.Cells
        .Where(c => c.Content == CellContent.Pellet)
        .OrderBy(c => Math.Abs(c.X - bot.X) + Math.Abs(c.Y - bot.Y))
        .ToList();

    if (!pellets.Any())
        return command;

    var target = pellets.First();

    Console.WriteLine($"Planned Action: {command.Action}");
    return command;
}
```

Step 3: Move Closer to the Pellet

- Chooses the best direction to get closer
- Moves toward target pellet if possible
- Try all directions (Up, Down, Left, Right)
- Skip walls and null tiles
- Compare current vs new distance to target
- Pick the direction that gets us closer

Step 3: Move Closer to the Pellet

```
var directions = new List<(BotAction action, int dx, int dy)>
{
    (BotAction.Up, 0, -1),
    (BotAction.Down, 0, 1),
    (BotAction.Left, -1, 0),
    (BotAction.Right, 1, 0)
};

BotCommand? fallbackCommand = null;
```

Step 3: Move Closer to the Pellet

```csharp
foreach (var (action, dx, dy) in directions)
{
    int newX = bot.X + dx;
    int newY = bot.Y + dy;

    var cell = gameState.Cells.FirstOrDefault(c => c.X == newX && c.Y == newY);
    if (cell != null && cell.Content != CellContent.Wall)
    {
        int currentDistance = Math.Abs(bot.X - target.X) + Math.Abs(bot.Y - target.Y);
        int newDistance = Math.Abs(newX - target.X) + Math.Abs(newY - target.Y);

        if (newDistance < currentDistance)
        {
            command.Action = action;
            Console.WriteLine($"Planned Action: {command.Action} (toward pellet)");
            return command;
        }

        fallbackCommand ??= new BotCommand { Action = action };
    }
}
```

Step 3: Move Closer to the Pellet

```
if (fallbackCommand != null)
{
    command = fallbackCommand;
    Console.WriteLine($"Planned Action: {command.Action} (fallback)");
}
```

https://l.ead.me/build-a-bot-2025

Step 3: Move Towards Power-Ups

- Power-ups can be game-changers
- We scan for all visible power-ups
- If there's a power-up within 5 Manhattan distance, the bot will prioritise it
- Otherwise, we weigh the closest power-up against the closest pellet and target the closer one

Step 3: Move Towards Power-Ups

```
var allPowerUps = gameState.Cells
    .Where(c =>
        c.Content == CellContent.PowerPellet ||
        c.Content == CellContent.ChameleonCloak ||
        c.Content == CellContent.Scavenger ||
        c.Content == CellContent.BigMooseJuice)
    .ToList();

var nearbyPowerUps = allPowerUps
    .Where(c => Math.Abs(c.X - bot.X) + Math.Abs(c.Y - bot.Y) <= 5)
    .OrderBy(c => Math.Abs(c.X - bot.X) + Math.Abs(c.Y - bot.Y))
    .ToList();

var pellets = gameState.Cells
    .Where(c => c.Content == CellContent.Pellet)
    .OrderBy(c => Math.Abs(c.X - bot.X) + Math.Abs(c.Y - bot.Y))
    .ToList();

Cell? target = null;
```

Step 3: Move Towards Power-Ups

```
if (nearbyPowerUps.Any())
{
    target = nearbyPowerUps.First();
}
else
{
    var closestPowerUp = allPowerUps.OrderBy(c => Math.Abs(c.X - bot.X) + Math.Abs(c.Y - 
bot.Y)).FirstOrDefault();
    var closestPellet = pellets.FirstOrDefault();

    if (closestPowerUp != null && closestPellet != null)
    {
        var distPowerUp = Math.Abs(closestPowerUp.X - bot.X) + Math.Abs(closestPowerUp.Y - bot.Y);
        var distPellet = Math.Abs(closestPellet.X - bot.X) + Math.Abs(closestPellet.Y - bot.Y);
        target = distPowerUp <= distPellet ? closestPowerUp : closestPellet;
    }
    else
    {
        target = closestPowerUp ?? closestPellet;
    }
}

if (target == null)
    return command;
```

Step 3: Prioritise Power-Ups

```
Console.WriteLine($"Planned Action: {command.Action} (toward {(target.Content ==
CellContent.Pellet ? "pellet" : "power-up")})");
```

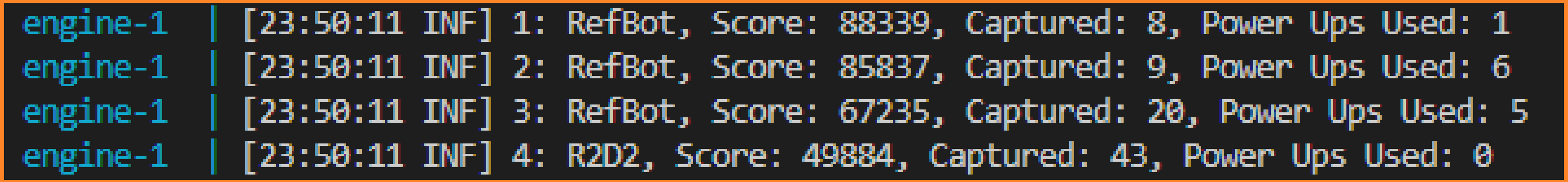
Step 3: Use Picked Up Power-Ups

- New game engine version introduces collectible power-ups
- Power-ups can give big scoring or survival advantages
- We'll activate a held power-up automatically
- Added before we do any movement logic

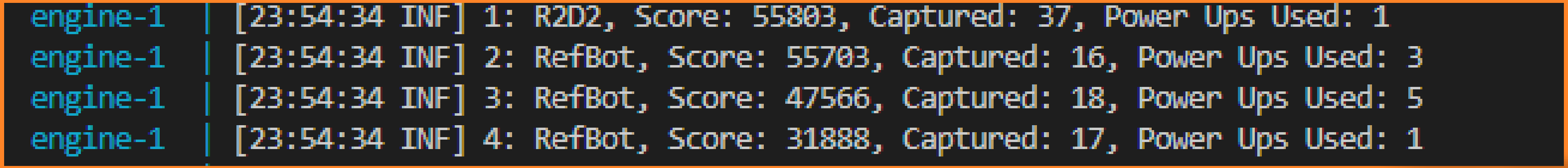
Step 3: Use Picked Up Power-Ups

```
if (bot.HeldPowerUp != null)
{
    Console.WriteLine("Planned Action: UseItem (activating power-up)");
    return new BotCommand { Action = BotAction.UseItem };
}
```

Step 3: Results



```
engine-1  | [23:50:11 INF] 1: RefBot, Score: 88339, Captured: 8, Power Ups Used: 1
engine-1  | [23:50:11 INF] 2: RefBot, Score: 85837, Captured: 9, Power Ups Used: 6
engine-1  | [23:50:11 INF] 3: RefBot, Score: 67235, Captured: 20, Power Ups Used: 5
engine-1  | [23:50:11 INF] 4: R2D2, Score: 49884, Captured: 43, Power Ups Used: 0
```



```
engine-1  | [23:54:34 INF] 1: R2D2, Score: 55803, Captured: 37, Power Ups Used: 1
engine-1  | [23:54:34 INF] 2: RefBot, Score: 55703, Captured: 16, Power Ups Used: 3
engine-1  | [23:54:34 INF] 3: RefBot, Score: 47566, Captured: 18, Power Ups Used: 5
engine-1  | [23:54:34 INF] 4: RefBot, Score: 31888, Captured: 17, Power Ups Used: 1
```

https://l.ead.me/build-a-bot-2025

Step 3: Avoid Danger Zones

- Adds a danger zone map from zookeeper positions
- Avoids zookeeper tiles and adjacent tiles
- Each zookeeper creates a diamond-shaped zone of tiles within Manhattan distance 3
- Avoid any direction that moves into a danger tile
- If no safe option moves us closer, fallback to the first safe direction

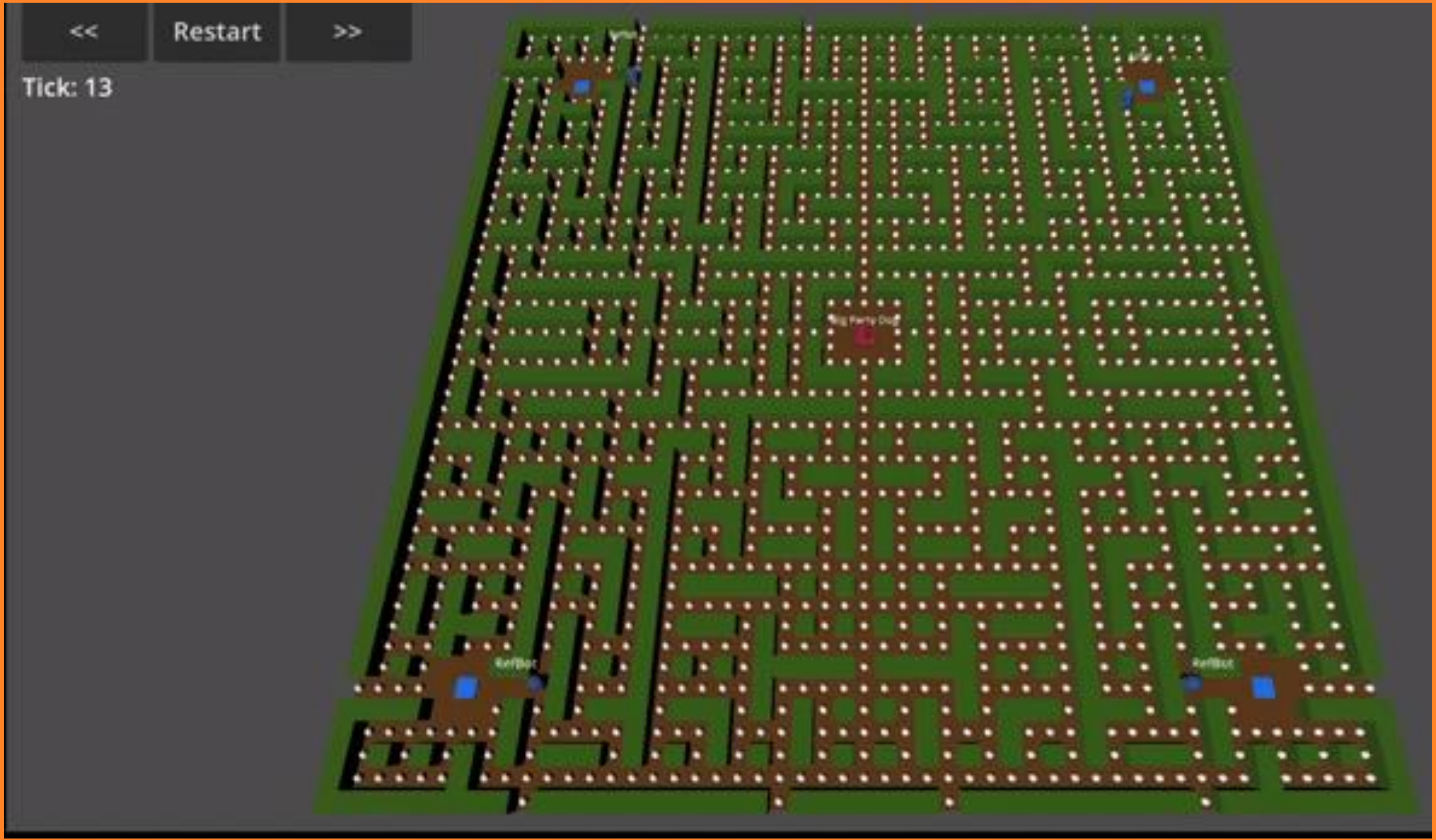
Step 3: Avoid Danger Zones

```
int maxX = gameState.Cells.Max(c => c.X);
int maxY = gameState.Cells.Max(c => c.Y);

var dangerZones = new HashSet<(int X, int Y)>();
int dangerRadius = 3;

foreach (var zk in gameState.Zookeepers)
{
    for (int dx = -dangerRadius; dx <= dangerRadius; dx++)
    {
        for (int dy = -dangerRadius; dy <= dangerRadius; dy++)
        {
            if (Math.Abs(dx) + Math.Abs(dy) > dangerRadius)
                continue;

            int zx = (zk.X + dx + maxX + 1) % (maxX + 1);
            int zy = (zk.Y + dy + maxY + 1) % (maxY + 1);
            dangerZones.Add((zx, zy));
        }
    }
}
```

Step 3: Avoid Danger Zones

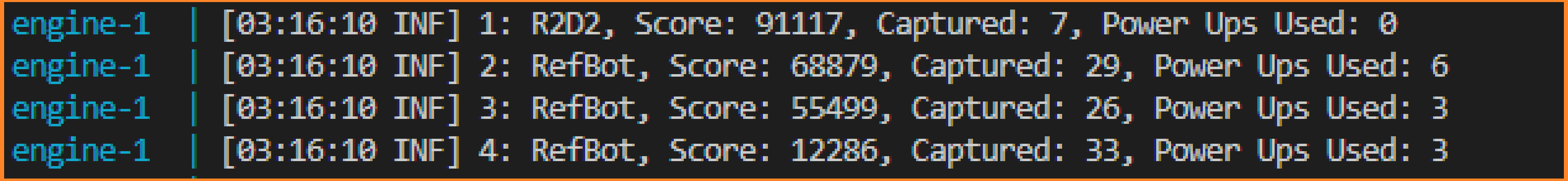
```
if (dangerZones.Contains((newX, newY)))
    Console.WriteLine($"Skipped {action} (danger zone)");

if (cell != null && cell.Content != CellContent.Wall && !dangerZones.Contains((newX, newY)))
```

Step 3: Results

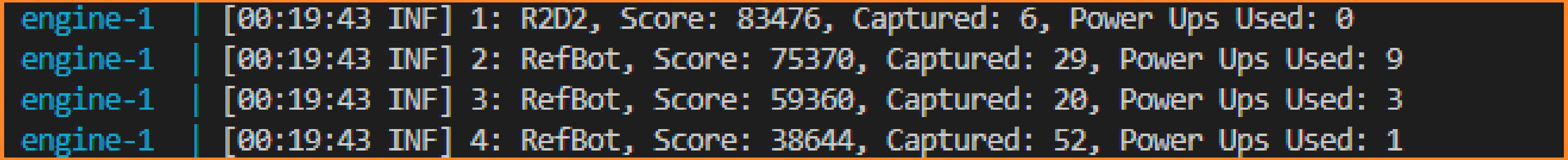


https://l.ead.me/build-a-bot-2025

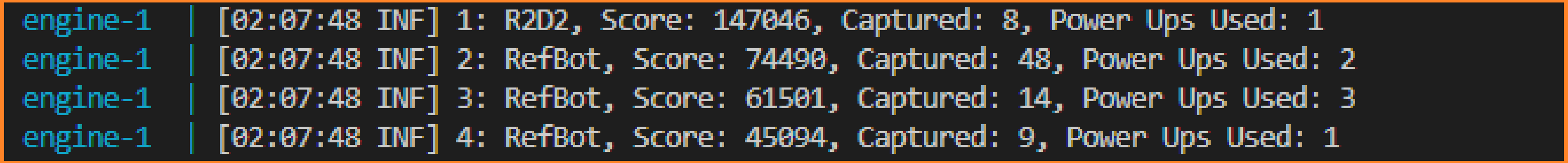
Step 3: Results



```
engine-1  | [03:16:10 INF] 1: R2D2, Score: 91117, Captured: 7, Power Ups Used: 0
engine-1  | [03:16:10 INF] 2: RefBot, Score: 68879, Captured: 29, Power Ups Used: 6
engine-1  | [03:16:10 INF] 3: RefBot, Score: 55499, Captured: 26, Power Ups Used: 3
engine-1  | [03:16:10 INF] 4: RefBot, Score: 12286, Captured: 33, Power Ups Used: 3
```



```
engine-1  | [00:19:43 INF] 1: R2D2, Score: 83476, Captured: 6, Power Ups Used: 0
engine-1  | [00:19:43 INF] 2: RefBot, Score: 75370, Captured: 29, Power Ups Used: 9
engine-1  | [00:19:43 INF] 3: RefBot, Score: 59360, Captured: 20, Power Ups Used: 3
engine-1  | [00:19:43 INF] 4: RefBot, Score: 38644, Captured: 52, Power Ups Used: 1
```



```
engine-1  | [02:07:48 INF] 1: R2D2, Score: 147046, Captured: 8, Power Ups Used: 1
engine-1  | [02:07:48 INF] 2: RefBot, Score: 74490, Captured: 48, Power Ups Used: 2
engine-1  | [02:07:48 INF] 3: RefBot, Score: 61501, Captured: 14, Power Ups Used: 3
engine-1  | [02:07:48 INF] 4: RefBot, Score: 45094, Captured: 9, Power Ups Used: 1
```

What's Next?

- We built something simple, a starting point
- We added:
  - Pellet targeting
  - Wall avoidance
  - Danger zone detection
- But we're just getting started
- Challenge is on you to build something better than this
- Remember: Every improvement teaches you something new!

Pizza Time!

Catch-up, questions, and pizza

Enjoy the rest of your time with us!

https://l.ead.me/build-a-bot-2025

THANK YOU

You made it this far!

Good luck for the rest!