CN7050- Intelligent Systems Module

Week-01 Tutorial

Week01 tutorial does not carry any marks toward the assessment component.

Tutor Task: (20 min)

1. First, introduce yourself and welcome the students. Inform the students of your email and possible office hours. Remind them that you are the main contact for questions about the coursework, as you are the one who will be grading it. The coursework instruction document will be released in week 3.
2. Ask each student to briefly introduce themselves, highlighting their interests, strengths, and especially their AI skills and capabilities, so the group can get to know one another.
3. Take feedback from the students on their understanding of Lecture 1 contents, and provide a brief recap before starting the tutorial task. This ensures those students who missed the lecture can participate actively.

Task 1 – Tap In

You need to Tap in to record your attendance in the Practical session.

Task 2 – Review the Moodle site

- Teaching Schedule
- Module Study Guide
- How to find learning activities for each week?
- Find links to the MS Teams site and how to access online Q&A sessions

Task 3 – How to get help

- Please speak with your Practical Group tutor if you have any questions during the practical session.
- If you have any questions about your studies, not relating to this module, please contact your advisor or program leader.

Introduction to Module Tools

There are three main tools that would be used in this module. These are:

1. MATLAB
2. Google Colab

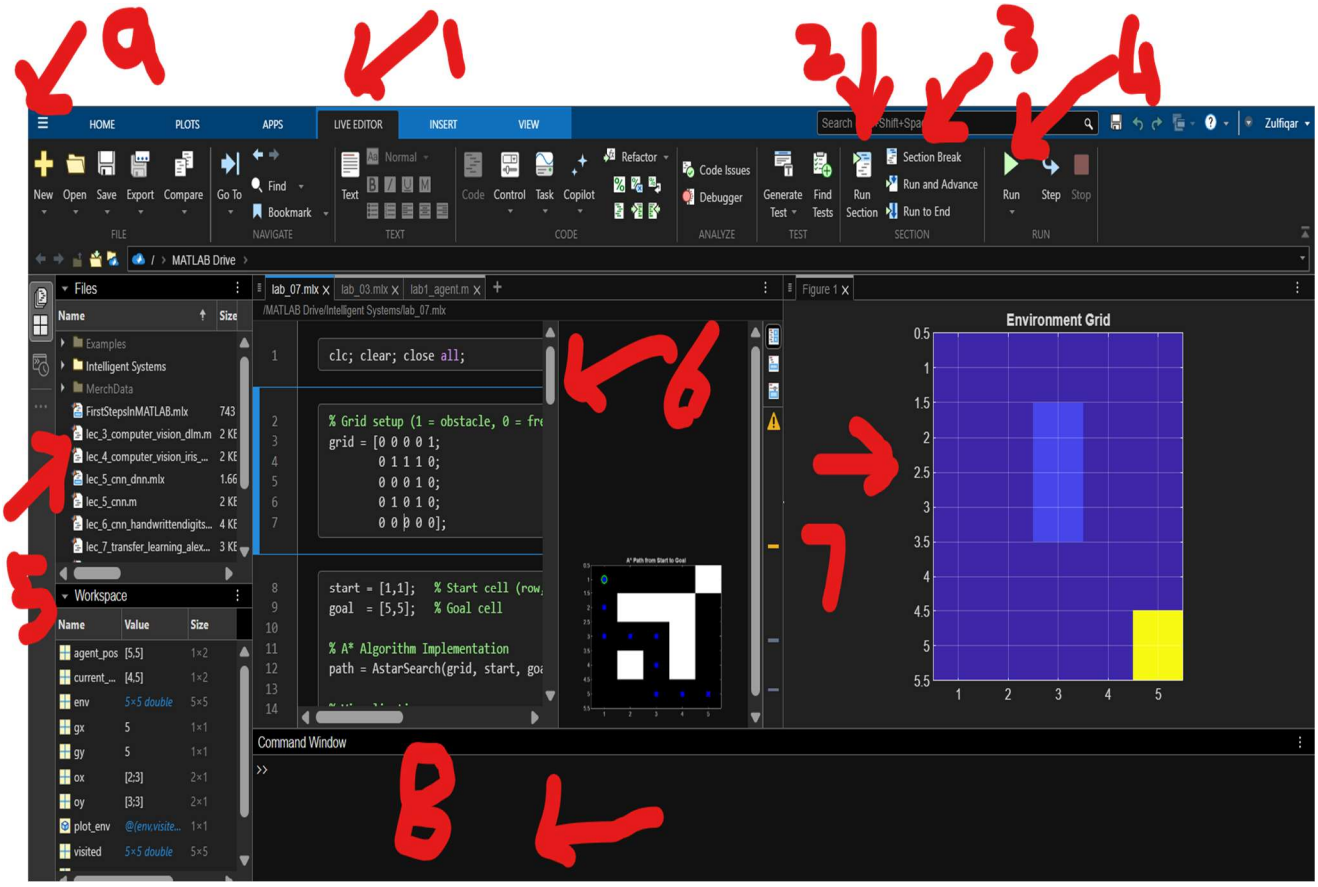
3. N8N

Below are the screenshots of each of these to give you an idea of their user interface.

MATLAB:

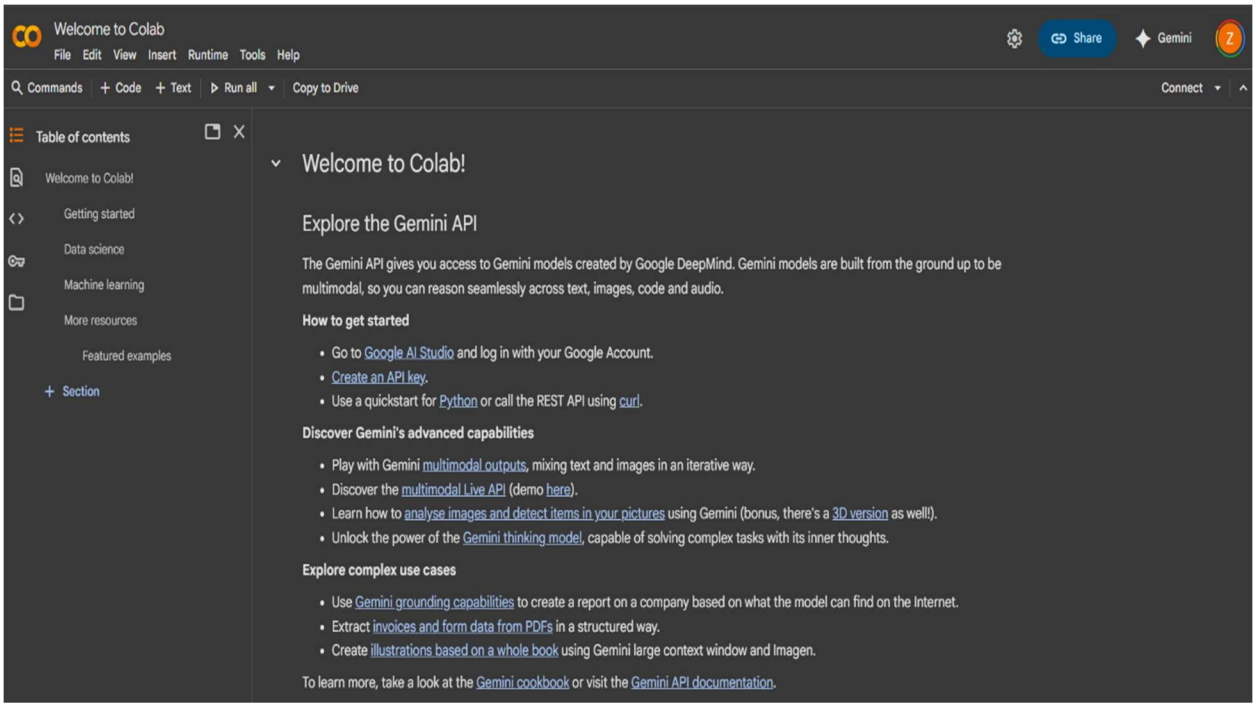
Here is the description of highlighted parts in the snapshot:

1. Live editor, it will be used for writing the code in the main editor and tools will provide you to work with code, text and execution of different parts of your code.
2. Run Section button will help you to run a single code cell to view its output.
3. Section break will help you to add a new code cell/section.
4. Run button will execute all the cells/sections from start to finish.
5. Your files and folders list in Matlab online directory.
6. Code panel, where you will be writing your code and comments.
7. Figures that will be drawn as an output of your code.
8. Command window, where you can run the Matlab commands directly, as well as, it will be showing the output of few of your code cells.
9. This section will be used to create a new script file, save a file or open a previously coded file.



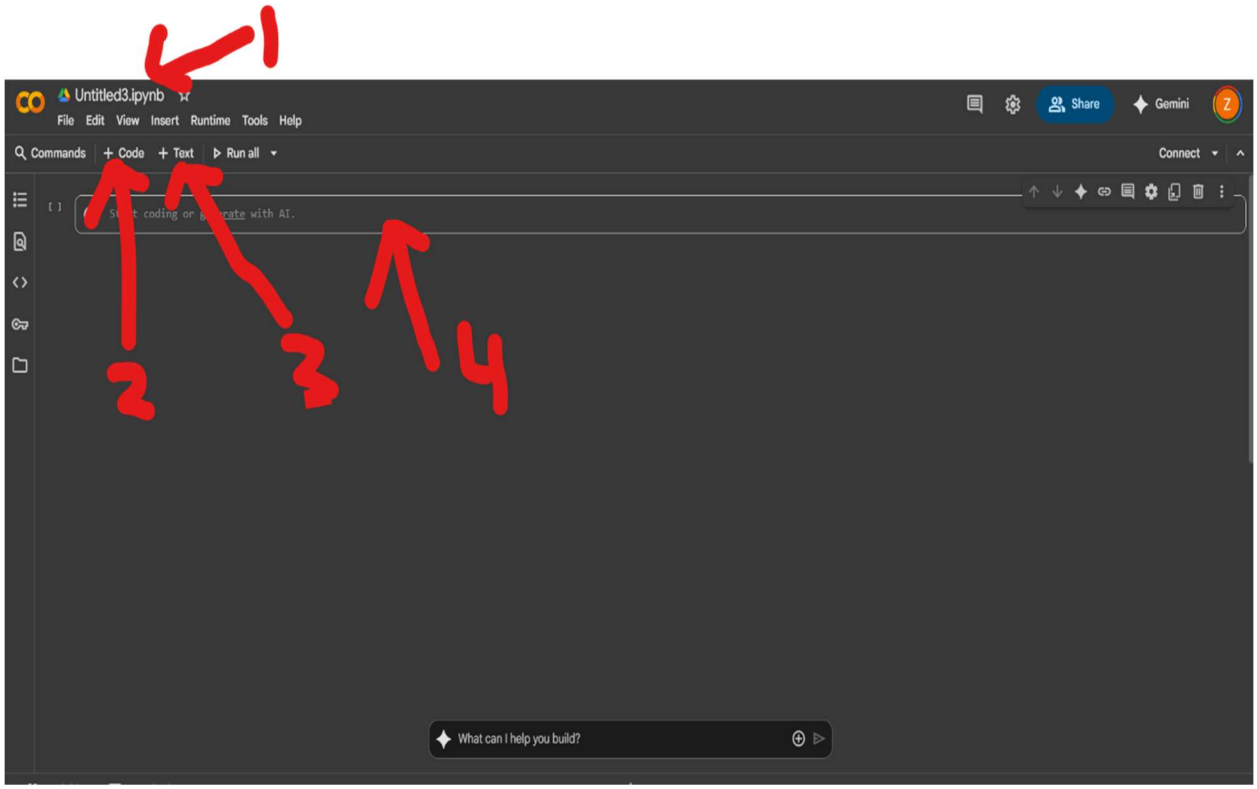
Google Colab:

When you will open the google colab, you will see a page like shown below. You can choose file and create a new notebook to start writing your code. That is shown in the following snapshot.



Once you will start the new notebook, you will see the screen as shown below. Here is the description of highlighted parts.

1. Name of your notebook, you can select and change it at any time.
2. Click +Code button to create a new code cell.
3. Click +Text button to create a cell for writing the text in your notebook to describe the code or purpose.
4. A code cell, where you will be writing actual code to execute. Each cell has a play button to execute the code written in that cell, or you can choose Run All in commands, to run all the code cells all at once.

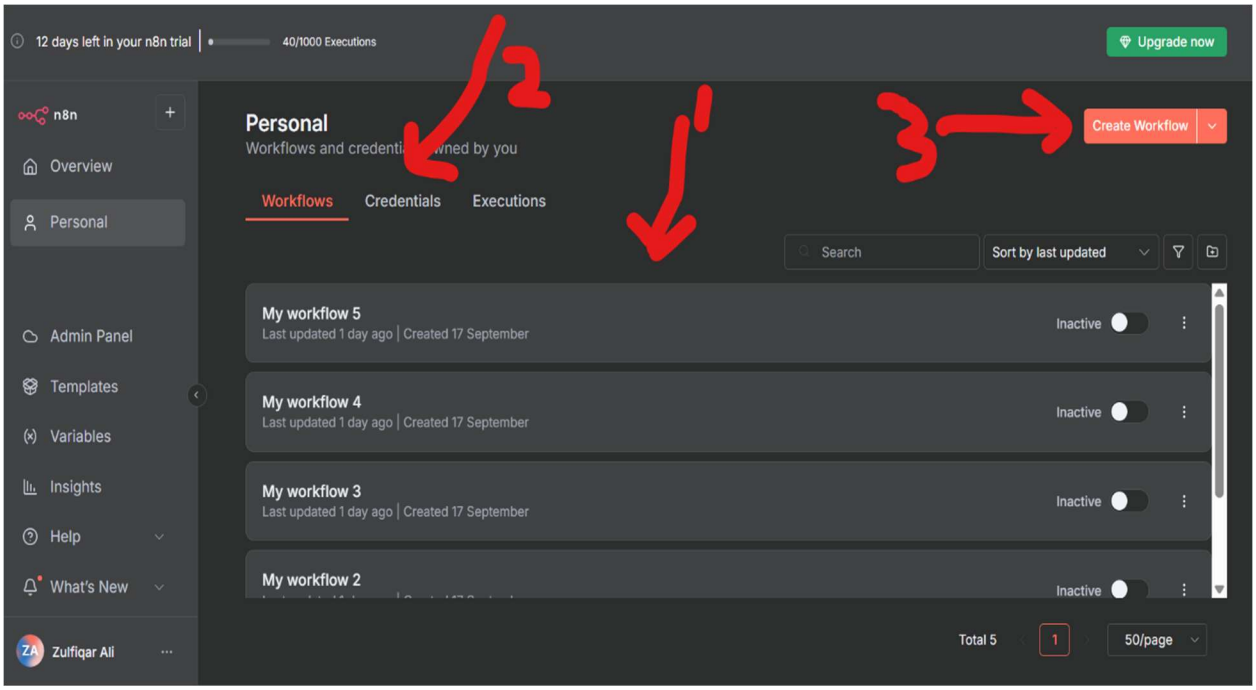


N8N:

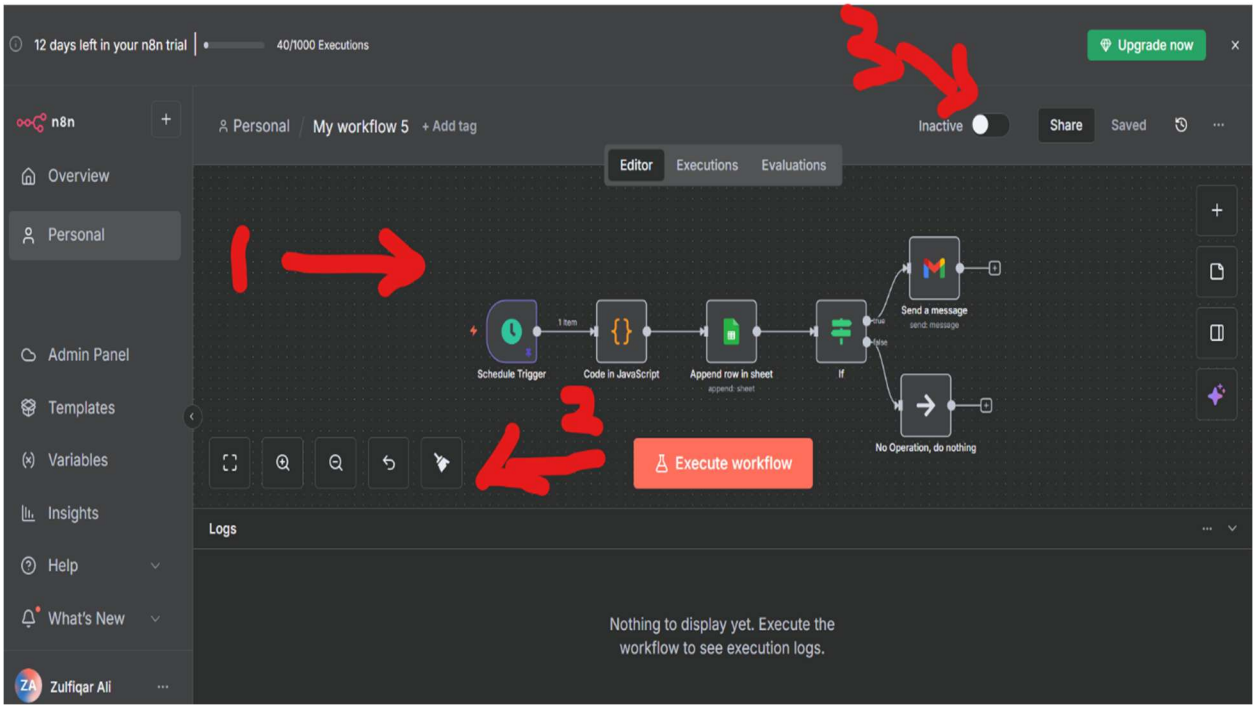
N8N is a no code workflow manager to create different agents or workflows. We will be using n8n in our module to create different agents.

In the following snapshot, you are seeing a n8n home page after you logged in. Here is the details of highlighted parts in the screenshot.

1. List of your previously created workflows.
2. You will see all the connected apps and credentials here. For instance, if you will connect and work with Gmail or google sheets, you will see both in credentials. You will learn more about it, once we will work with n8n.
3. You can click “Create Workflow” to start a new workflow. In the following screenshot, you will see the workflow panel.



Here is the workflow panel in n8n.



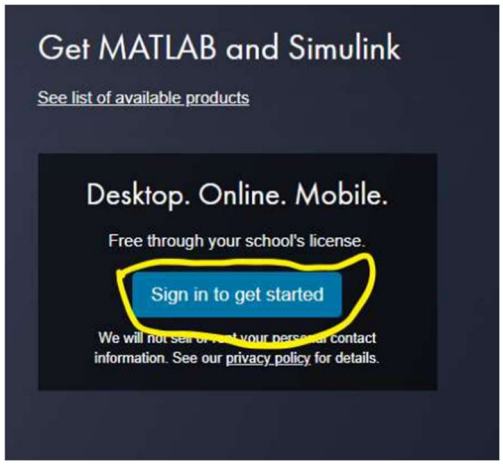
1. It shows the created workflow in the n8n. You can add/delete or edit the nodes created in the workflow.
2. This section will be used extensively during creation of our workflow to zoom in, zoom out or fit to screen kind of tasks.

3. Once we will be done with the workflow and wish to host it, we can activate using the slider shown in 3. Note that we will be using the trial version, which will expire in 14 days, so if you wish to host a workflow beyond that, you will be required to subscribe to the paid plan on n8n.

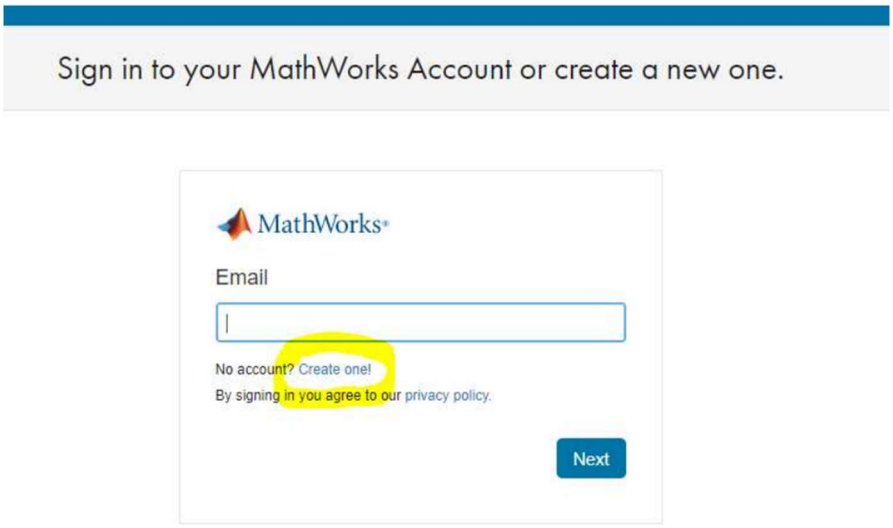
With this, we are done with the intro of the tools we will be using in this module. So, lets proceed and have a taste of first tool to be used in this module.

Task 4 – Access the software used in this module: MATLAB

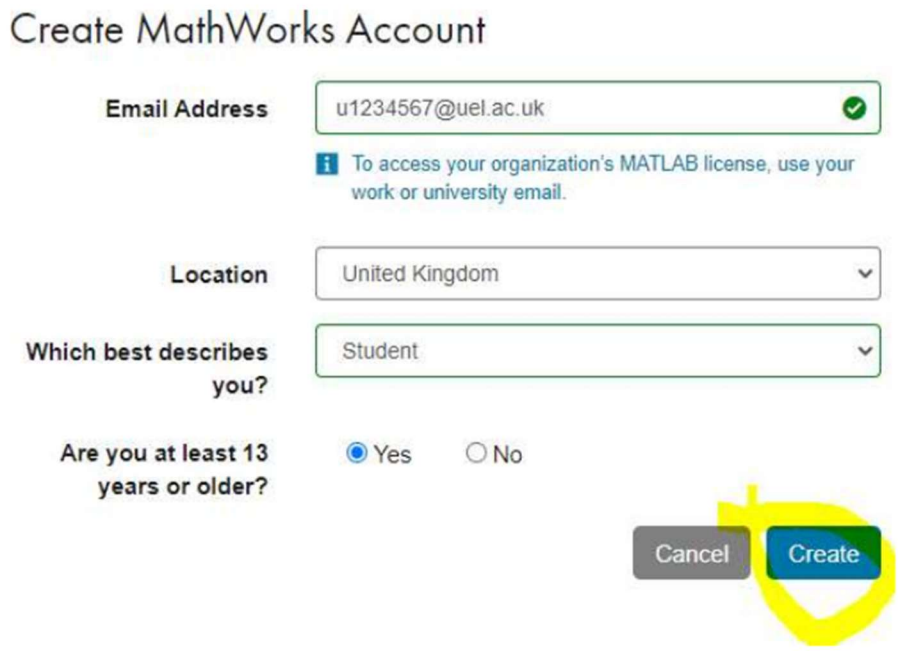
1. Go to: https://uk.mathworks.com/academia/tah-portal/university-of-east-london-31543424.html or https://matlab.mathworks.com/
2. Scroll down and click on **Sign in to get started**.



3. Click on **Create one!**



4. Enter your University of East London student email; location; choose **Student**; choose **Yes**; click **Create**.

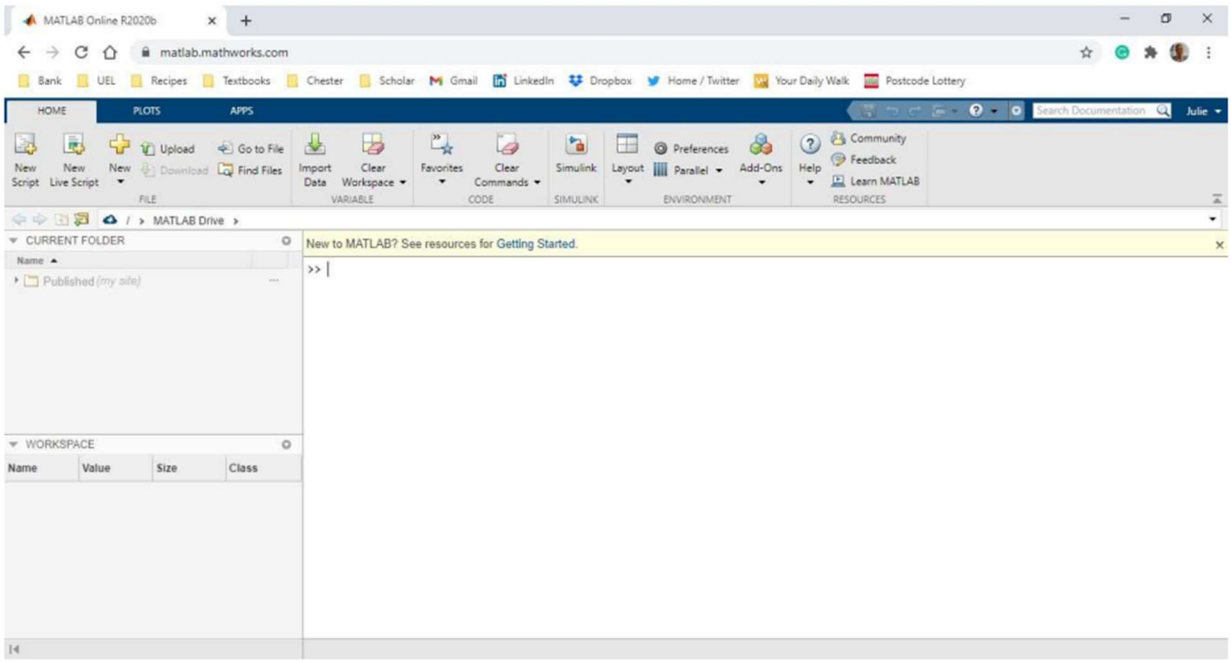


5. Check your email. MATLAB will email you with details for accessing MATLAB online. It may take up to 15 minutes for the email to arrive. Click **Verify your email**.

Task 5 – Using MATLAB Online

1. Click on: https://matlab.mathworks.com/?s_tid=tah_po_start
2. Sign in and click on:

3. The MATLAB Online editor will appear.



Task 6 – Start a MATLAB online course

Start working through the MATLAB online courses:

a. **MATLAB Onramp**

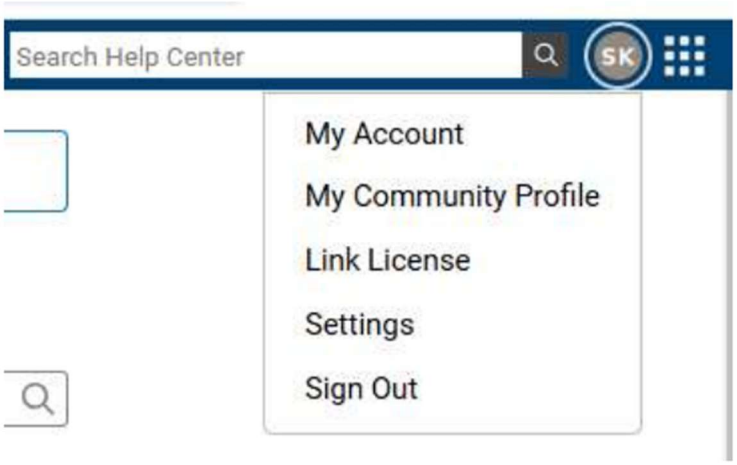
You can search these courses using the search bar at https://matlabacademy.mathworks.com/.

First course will get you started with learning the basics of MATLAB. When the course has been completed, you will receive a certificate which you can share on LinkedIn. If you have completed the first course in a previous module at UEL, you can directly proceed to second course.

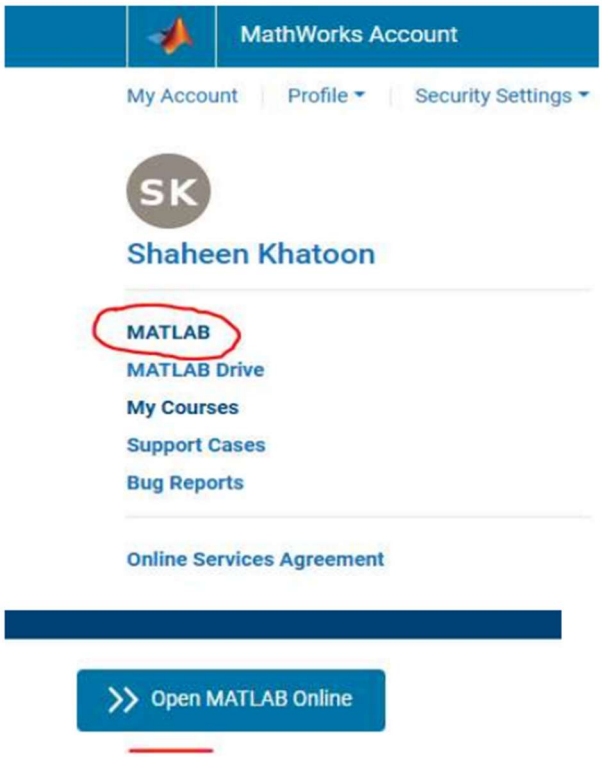
You will complete MATLAB on ramp course before the Practical session in Week 2.

Task 7: Open MATLAB Online

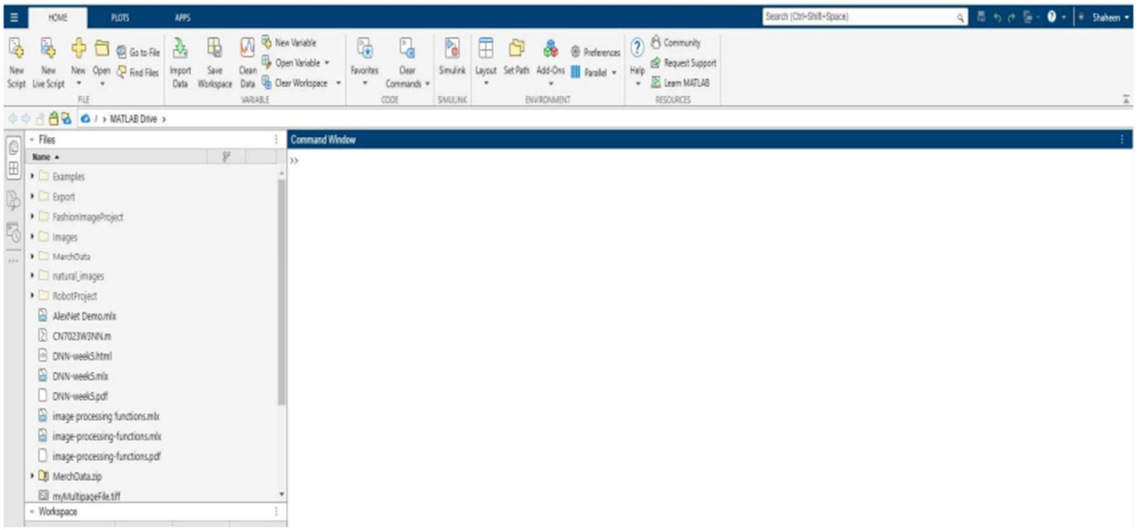
1. Click on: https://uk.mathworks.com/academia/tah-portal/university-of-east-london31543424.html
2. Sign in, click on your account initials and then click **My Account**



3. Click on **MATLAB** and then click **Open MATLAB Online**

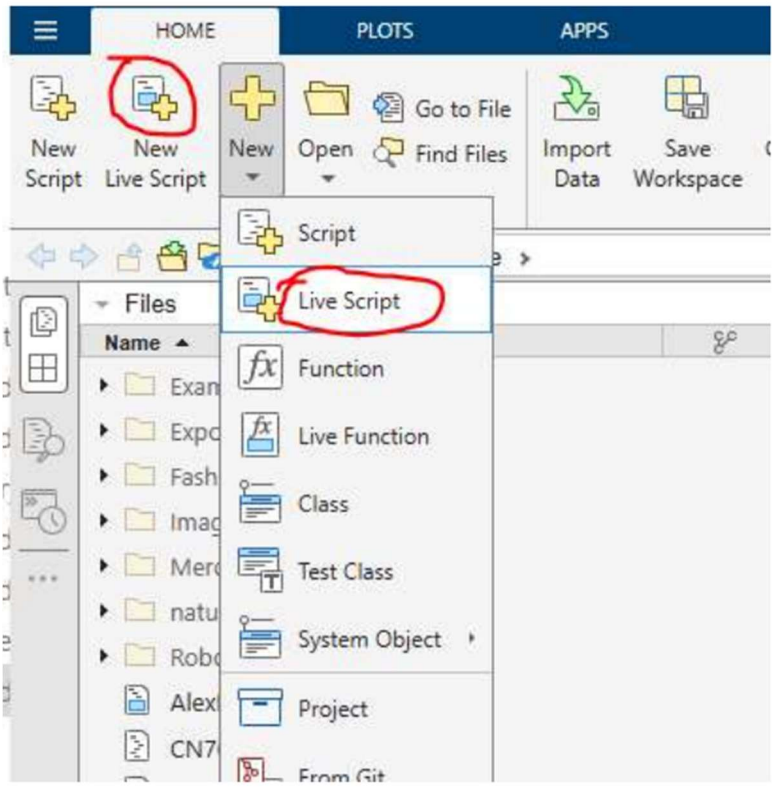


4. MATLAB Online editor will open.

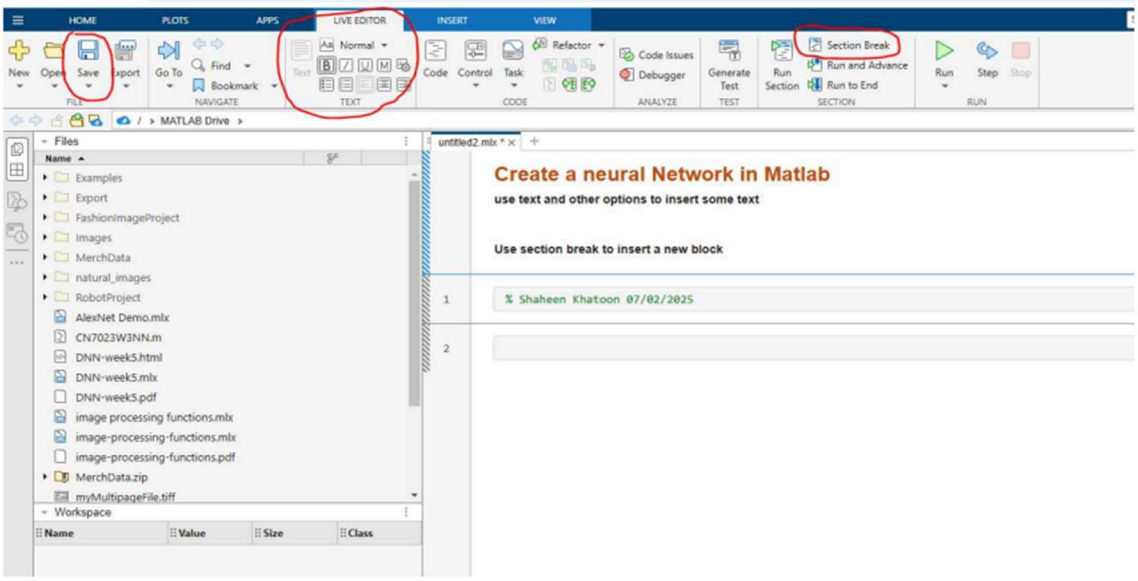


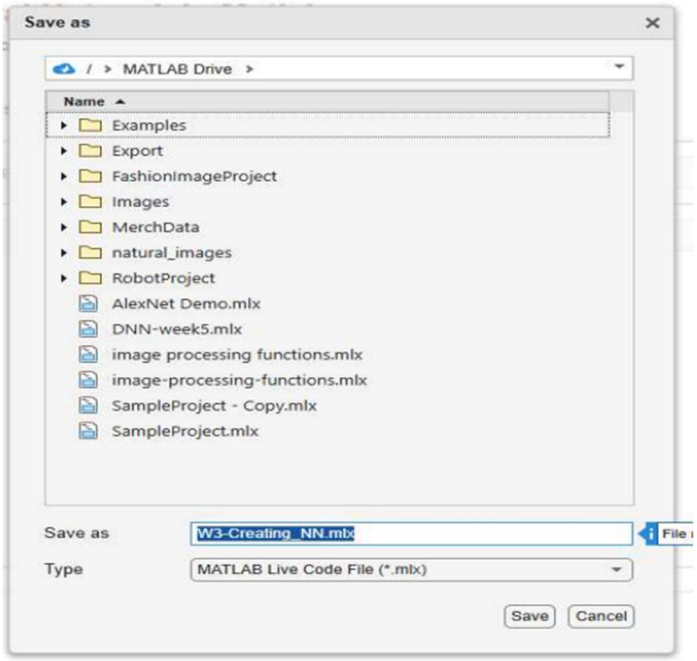
Task 8: Implement a Reactive Agent in MATLAB

1. Create a new MATLAB Live script

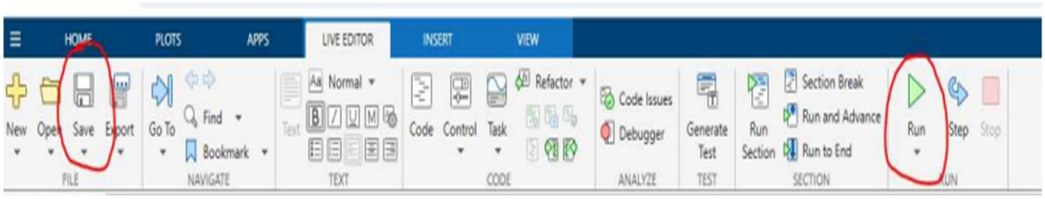


2. Add your **name, date, and purpose** of the code at the top as a comment. Save the file.





3. Add the following section codes step-by-step and run each to observe outputs.



4. **Clear the environment:**

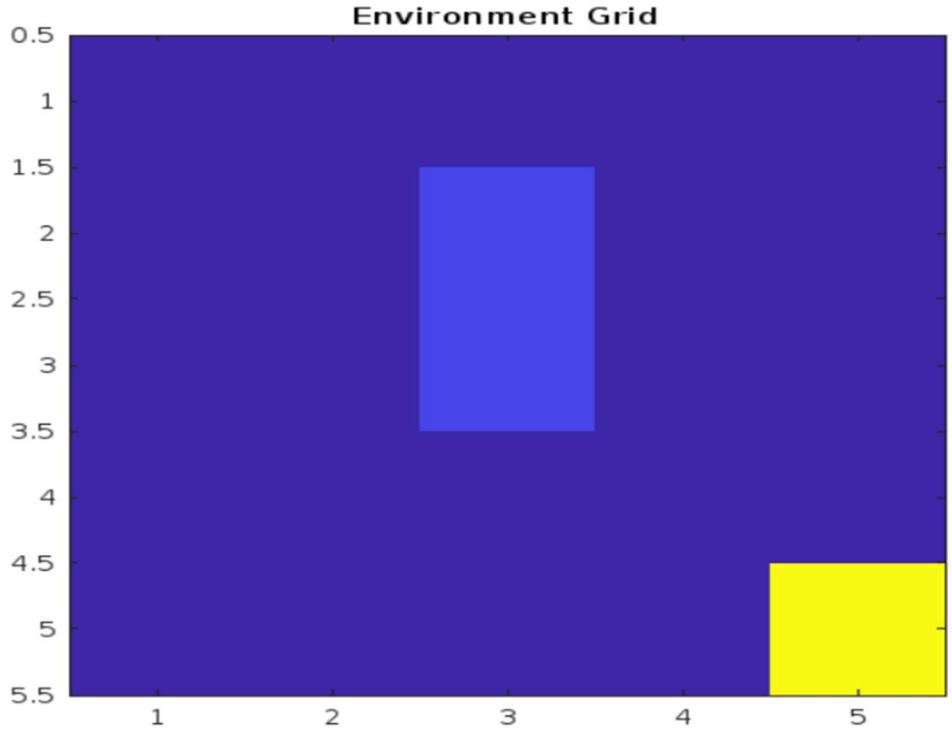
```
%Close all open figures
close all
%Clear the workspace
clear
%Clear the command window
clc
```

5. **Add the following code in the next section**

```
% Grid representation: 0 = empty, 1 = obstacle, 9 = goal
env = zeros(5,5);
env(2,3) = 1;
env(3,3) = 1;
% env(3,5) = 1;
env(5,5) = 9;  % Goal

% Initial agent position
agent_pos = [1,1];

imagesc(env);
title('Environment Grid');
axis equal tight;

while ~isequal(agent_pos, [5,5])
    current_pos = agent_pos;

    if current_pos(2) < 5 && env(current_pos(1), current_pos(2)+1) ~= 1
        agent_pos(2) = agent_pos(2) + 1; % Move right
    elseif current_pos(1) < 5 && env(current_pos(1)+1, current_pos(2)) ~= 1
        agent_pos(1) = agent_pos(1) + 1; % Move down
    else
        disp('Blocked. Stopping.');
        break;
    end

    fprintf('Moved to (%d,%d)\n', agent_pos(1), agent_pos(2));
    pause(0.3);
end
```

6. Run the code by pressing the **Run Section** button on the toolbar.

   You will see the environment grid created like shown below.



7. This is the environment grid created by your code. Also, your agent is moving, which is not visible right now on the grid, you can see the movement of the agent in the command window output as shown below.

```
Command Window
Moved to (1,2)
Moved to (1,3)
Moved to (1,4)
Moved to (1,5)
Moved to (2,5)
Moved to (3,5)
Moved to (4,5)
Moved to (5,5)
>>
```

Our agent has successfully reached the desired position, as it did not encounter any blockage.

What will happen if it encounters a blockage?

8. Let's uncomment the code at line-5, the code is env(3,5) = 1; That means in third row and 5th column of our grid there is a blockage. Your code will look like this after this change;

```
% Grid representation: 0 = empty, 1 = obstacle, 9 = goal
env = zeros(5,5);
env(2,3) = 1;
env(3,3) = 1;
env(3,5) = 1;
env(5,5) = 9;  % Goal

% Initial agent position
agent_pos = [1,1];

imagesc(env);
title('Environment Grid');
axis equal tight;

while ~isequal(agent_pos, [5,5])
    current_pos = agent_pos;

    if current_pos(2) < 5 && env(current_pos(1), current_pos(2)+1) ~= 1
        agent_pos(2) = agent_pos(2) + 1; % Move right
    elseif current_pos(1) < 5 && env(current_pos(1)+1, current_pos(2)) ~= 1
        agent_pos(1) = agent_pos(1) + 1; % Move down
    else
        disp('Blocked. Stopping.');
        break;
    end

    fprintf('Moved to (%d,%d)\n', agent_pos(1), agent_pos(2));
    pause(0.3);
end
```

9. Re run the code using Run Section button and you will see the following output the command window. Now our agent could not make up to the desired position. As it was

only capable of moving right and downwards, whilst it went all the way to the right and started to move downwards, and stuck by the blockage in 3rd block of grid.

```
Command Window
Moved to (1,2)
Moved to (1,3)
Moved to (1,4)
Moved to (1,5)
Moved to (2,5)
Blocked. Stopping.
>>
```

We could not saw our agent moving, so let's plot the movement of our agent to make it fun watching!

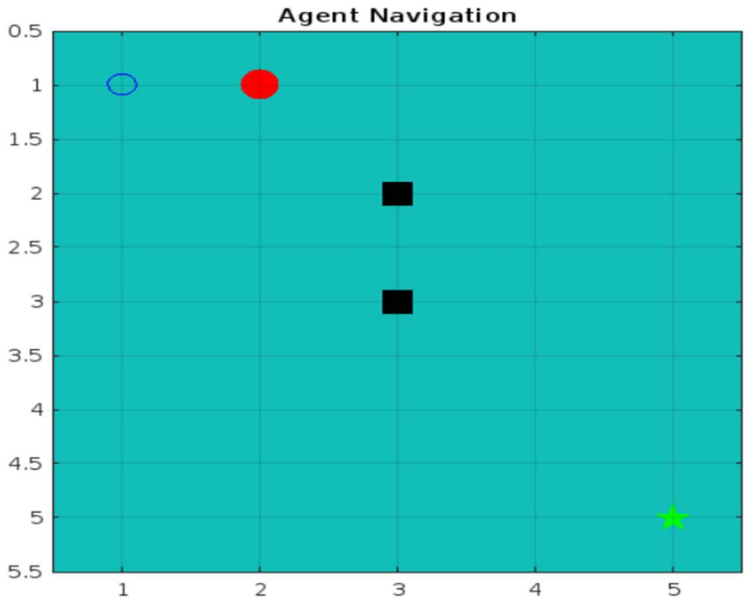
10. Put the following code in the next section of your script.

```
%%
clc; clear;

% Grid setup: 0 = free, 1 = obstacle, 9 = goal
env = zeros(5,5);
env(2,3) = 1;
env(3,3) = 1;
env(5,5) = 9;  % Goal

% Initialize
agent_pos = [1,1];
visited = zeros(5,5);
visited(1,1) = 1;

% Visualization function
plot_env = @(env, visited, agent_pos) ...
    imagesc(env + visited*2 + (agent_pos(1)==(1:5)' & agent_pos(2)==(1:5))*3

% Main loop
while ~isequal(agent_pos, [5,5])
    current_pos = agent_pos;

    % Decision logic (simple reflex)
    if current_pos(2) < 5 && env(current_pos(1), current_pos(2)+1) ~= 1
        agent_pos(2) = agent_pos(2) + 1;
    elseif current_pos(1) < 5 && env(current_pos(1)+1, current_pos(2)) ~= 1
        agent_pos(1) = agent_pos(1) + 1;
```

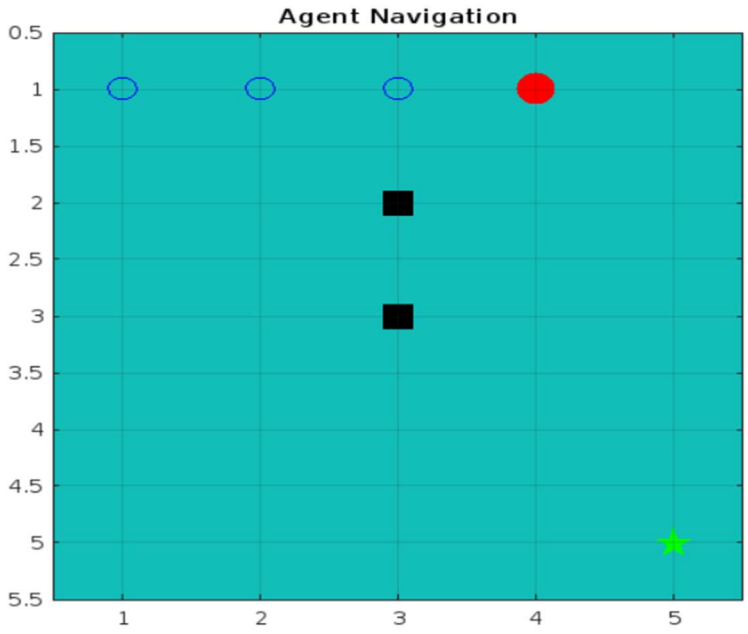
```
        else
            disp('Blocked. Cannot move.');
            break;
        end

        visited(agent_pos(1), agent_pos(2)) = 1;

        % Plot grid with agent and visited path
        clf;
        imagesc(zeros(5)); hold on;

        % Show obstacles
        [ox, oy] = find(env == 1);
        scatter(oy, ox, 300, 's', 'filled', 'k');

        % Show goal
        [gx, gy] = find(env == 9);
        scatter(gy, gx, 300, 'p', 'filled', 'g');

        % Show visited cells
        [vx, vy] = find(visited == 1);
        scatter(vy, vx, 150, 'o', 'MarkerEdgeColor', 'b');

        % Show agent
        scatter(agent_pos(2), agent_pos(1), 300, 'filled', 'r');
        title('Agent Navigation');
        axis equal tight;
        xlim([0.5 5.5]); ylim([0.5 5.5]);
        grid on;
        pause(1);
    end
```

11. Click the Run Section button to run your code.
12. You will see the environment created as shown in the following figure. Blue outlines show the steps taken by agent, red ball is our agent, green star is the goal position and black squares are the hurdles or blockages.

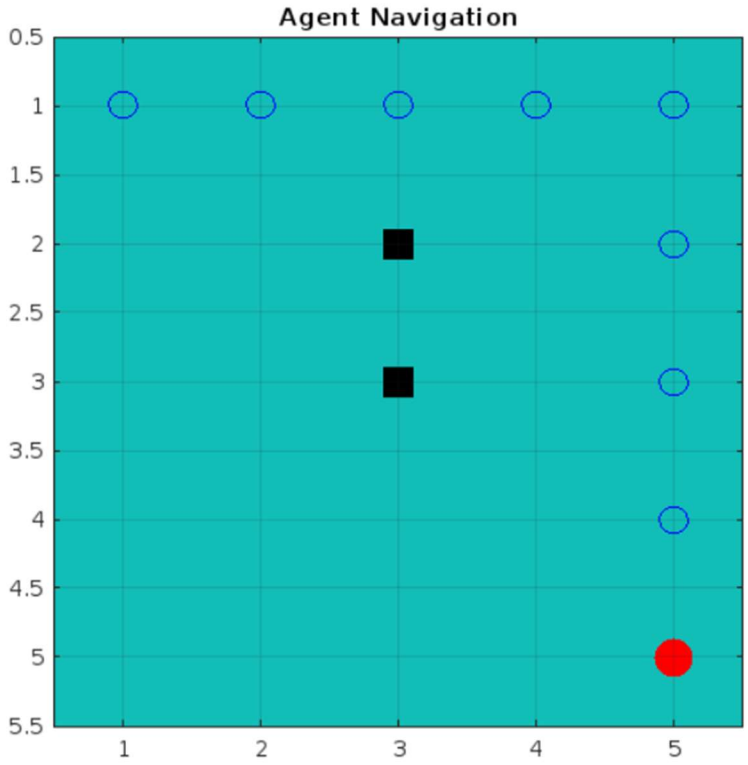
Note that, the red ball, which denotes our agent will be at the first blue outlined ball, the current position of our agent is after taking the first step towards the goal.



Environment grid after agent took few more steps.



Our agent is moving towards the right all the way and then downwards, as per the rules set by us in our code (if/else block code). The blue outlines show the entire route and steps taken by our agent.



Note: You can change the speed of movement of agent using the code at line 87 i.e. pause (1). Whereas 1 here means seconds, so you can make the agent slower or faster by changing the values to a lower number like .1, .2, .3 etc.; or higher like 2,3,4.

13. Experiment with the different blockage positions and see how agent behaves, if it still has an opportunity to reach the goal or when it is left with no opportunity to move forward as per the set rules.

Observations

Our Agent is Reactive Because:

- It checks the **environment state** at the **current moment** (e.g., is the cell to the right or below an obstacle?).
- It then applies a **hard-coded rule** like:

If the cell to the right is free, move right. Else if down is free, move down.

- There is **no memory** of previously visited locations.
- It **doesn't plan** to reach the goal; it just reacts locally.

It's Not a Planning Agent Because:

- It doesn't use A* or search.
- It doesn't optimize future moves.
- It doesn't backtrack or learn from failed paths.

A model-based or planning agent (we will develop in later labs) would adapt or reroute.

Notes:

- Complete each individual task and then click **Submit** to check if correct. If correct, continue to the next task.
- If you are in difficulty, click on **Hint** to get help.
- If you are still not sure how to complete the task, click on **See Solution**.

Please speak with your Practical Group tutor if you have any questions.

THE END