Experiment 2.1

Student Name: Khushi Bhatia **UID: 21BCS5735**

Branch: BE-CSE **Section/Group: NTPP_CC_601/A**

Semester: 6 **Date of Performance: 26/02/24**

Subject Name: CC & DS lab **Subject Code: 21CSP-378**

1. **Aim:** Simulate a cloud scenario using MATLAB and run a scheduling algorithm.

2. **Objectives:**

   - To create a simple cloud environment with tasks and implement a basic scheduling algorithm.

3. **Software Requirements:**

   - MATLAB
   - Eclipse IDE
   - Cloud Sim 3.0.3
   - Apache Commons Math 3.6.1

4. **Description:**

   MATLAB combines a desktop environment tuned for iterative analysis and design processes with a programming language that expresses matrix and array mathematics directly. It includes the Live Editor for creating scripts that combine code, output, and formatted text in an executable notebook.

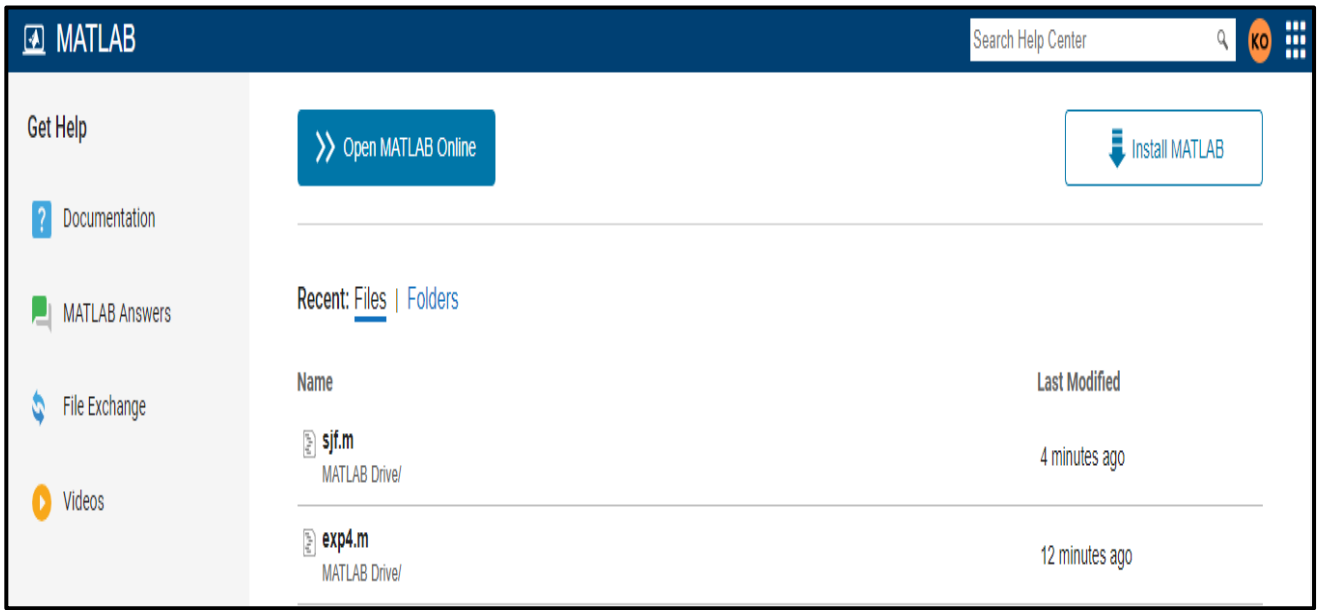
   MATLAB lets you take your ideas from research to production by deploying to enterprise applications and embedded devices, as well as integrating with Simulink® and Model-Based Design.

   Using MATLAB we can:

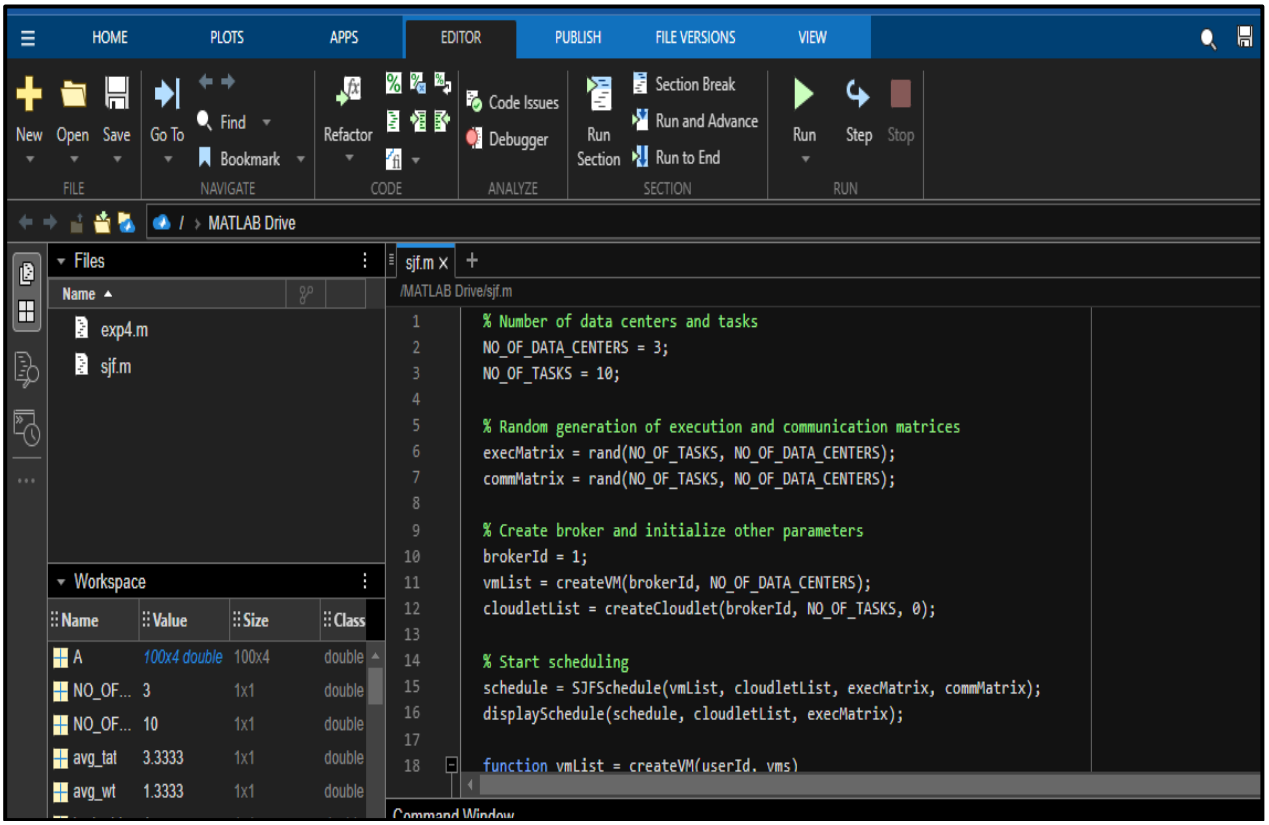
   - Analyze data
   - Develop algorithms
   - Create models and applications

5. Procedure:

a) Open Matlab Simulink online.



b) Create new script file and write down code of scheduling algorithm.



6. Code:

```
% Number of data centers and tasks
NO_OF_DATA_CENTERS = 3;
NO_OF_TASKS = 10;
    % Random generation of execution and communication matrices
    execMatrix = rand(NO_OF_TASKS, NO_OF_DATA_CENTERS);
    commMatrix = rand(NO_OF_TASKS, NO_OF_DATA_CENTERS);

    % Create broker and initialize other parameters
    brokerId = 1;
    vmList = createVM(brokerId, NO_OF_DATA_CENTERS);
    cloudletList = createCloudlet(brokerId, NO_OF_TASKS, 0);

    % Start scheduling
    schedule = SJFSchedule(vmList, cloudletList, execMatrix, commMatrix);
    displaySchedule(schedule, cloudletList, execMatrix);

    function vmList = createVM(userId, vms)
       % Placeholder for VM creation logic (not implemented in this example)
       vmList = cell(vms, 1);
       for i = 1:vms
          vmList{i} = struct('Id', i, 'UserId', userId);
       end
    end

    function cloudletList = createCloudlet(userId, cloudlets, idShift)
       % Placeholder for cloudlet creation logic (not implemented in this example)
       cloudletList = cell(cloudlets, 1);
       for i = 1:cloudlets
          cloudletList{i} = struct('Id', idShift + i, 'UserId', userId);
       end
    end

    function schedule = SJFSchedule(vmList, cloudletList, execMatrix, commMatrix)
       % Sort cloudlets based on their execution time using SJF algorithm
       [~, indices] = sort(sum(execMatrix, 2)); % Sorting based on sum of execution times
    for each cloudlet
       cloudletList = cloudletList(indices);

       % Assign cloudlets to VMs
       schedule = zeros(length(cloudletList), 5); % Store schedule (VM Id, Cloudlet Id,
    Start Time, Finish Time, Waiting Time)
       vmCapacity = ones(length(vmList), 1); % Initialize VM capacity
       currentTime = 0; % Initialize current time
       for i = 1:length(cloudletList)
          cloudlet = cloudletList{i};
          [~, vmIndex] = min(vmCapacity); % Find VM with minimum capacity
          startTime = currentTime;
          executionTime = sum(execMatrix(cloudlet.Id, :));
```

```
        finishTime = startTime + executionTime;
        waitingTime = startTime;
        schedule(i, :) = [vmList{vmIndex}.Id, cloudlet.Id, startTime, finishTime,
waitingTime]; % Assign cloudlet to VM
        currentTime = finishTime;
        vmCapacity(vmIndex) = vmCapacity(vmIndex) + 1; % Update VM capacity
    end
end

function displaySchedule(schedule, cloudletList, execMatrix)
    fprintf('Cloudlet ID\tVM ID\tStart Time\tFinish Time\tWaiting Time\n');
    for i = 1:size(schedule, 1)
        cloudletId = schedule(i, 2);
        vmId = schedule(i, 1);
        startTime = schedule(i, 3);
        finishTime = schedule(i, 4);
        waitingTime = schedule(i, 5);
        fprintf('%d\t\t%d\t%.2f\t\t%.2f\t\t%.2f\n', cloudletId, vmId, startTime,
finishTime, waitingTime);
    end
end
```

7. Output:

```
>> sjf
Cloudlet ID     VM ID   Start Time      Finish Time     Waiting Time
8               1       0.00            0.50            0.00
4               2       0.50            1.20            0.50
3               3       1.20            2.24            1.20
10              1       2.24            3.30            2.24
7               2       3.30            4.40            3.30
1               3       4.40            5.84            4.40
5               1       5.84            7.28            5.84
6               2       7.28            8.89            7.28
2               3       8.89            10.61           8.89
9               1       10.61           12.66           10.61
>>
```

8. Learning Outcomes:

- Simulation using Cloud Scenarios.
- Learnt how to implement MATLAB.