

CYPRUS INTERNATIONAL UNIVERSITY

FACULTY OF ENGINEERING DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING

PROGRAMMING IN MATLAB FOR ENGINEERING

[ENGI316](https://moodle.ciu.edu.tr/course/view.php?id=1947)

Group Project

**Fall 2023-2024**

**Name Surname ID**

**Ruchira Dissanayaka 21815070**

**Project Title :**

**Car Park Control Scenario**

**GROUP OF STUDENTS DETAILS :**

**Member 1:**

**Students Number :** 21815070

**Name :** Ruchira Dissanayaka

**Department :** Computer Engineering

**Role Of Project :** Program Developer

**Member 2:**

**Students Number :** 21809433

**Name :** Emmanuel Ezeeli

**Department :** Electrical Engineering

**Role Of Project :** Research And Support For Coding

Table Of Contents

[**THE PROBLEM DEFINITION AND FORMULATION 5**](#_Toc155009993)

[**Explanation Of The Problem : 5**](#_Toc155009994)

[**Definitions Of The Variables Used In The Problem : 5**](#_Toc155009995)

[**Descriptions Of The Tasks That Need To Be Performed To Solve The Problem : 6**](#_Toc155009996)

[**THE SOLUTION METHOD 7**](#_Toc155009997)

[**The Calculations Required To Solve Each Task Of The Problem : 7**](#_Toc155009998)

[**The Equations Used To Solve The Problem : 8**](#_Toc155009999)

[**The Definitions Of The Symbols Used In That Equations : 10**](#_Toc155010000)

[**STRUCTURE CHART OF THE PROGRAM 13**](#_Toc155010001)

[**PSEUDOCODE OF THE PROGRAM 14**](#_Toc155010002)

[**FLOWCHART OF THE PROGRAM 16**](#_Toc155010003)

[**EXPLANATION ON HOW THE RELATED MATLAB CODES AND DESIGNED GUI 17**](#_Toc155010004)

[**EXPLANATIONS OF ALL FIGURES 27**](#_Toc155010005)

[**EXCEL FILE DATA 30**](#_Toc155010006)

**FUNCTION AND IT CODES THAT WE CALLING FROM ANOTHER M FILE………………………………………………31**

**QUESTION THAT BELONG TO THAT CALLING FUNCTION……………………………………………………………………32**

# **THE PROBLEM DEFINITION AND FORMULATION**

## **Explanation Of The Problem :**

The Goal Of Our Project Is To Simulate A Typical Parking Lot Situation As A Responsible Employee Or Conductor Operating The System From The Control Booth Using A Visible Gadget Or Computer Using Provided Gui. There Are Ten Parking Spots In Our Lot, And The Cost Is $20 Per Hour Or Less. The Parking Lot Is Additionally Open 24/7.Basically, The Control Booth's Job Is To Collect A Car's Number Plate Information, Put It Into A System, And Then, When The Car Leaves, Take The Number Plate Information And Enter It Into The System To Charge The Customer The Appropriate Amount. When Everything Is Finished, The Control Booth Must Save All Of The Entries Made During The Day Into A File Or Database In Order To Store All Of The Data Once The Work Is Completed.

## **Definitions Of The Variables Used In The Problem :**

* **numberPlate**

**Type: String**

**Description:** Represents the car's number plate entered by the user.

* **currentData**

**Type:** Cell array

**Description:** Contains the current data displayed in the uitable2 table representing the car park status**.**

* **newEntry**

**Type:** Cell array

**Description:** Represents a new entry to be added to the car park status table. It includes CarPlate, EntryTime, ExitTime, and ChargeAmount.

* **updatedData**

**Type:** Cell array

**Description:** Contains the updated data after adding a new entry to the car park status table.

* **entryIndex**

**Type:** Integer

**Description:** Represents the index of the car entry in the car park status table.

* **entryTime**

**Type:** Datetime array

**Description:** Represents the date and time when a car entered the parking lot.

* **exitTime**

**Type:** Datetime array

**Description:** Represents the date and time when a car exited the parking lot.

* **entryDurationHours**

**Type:** Double

**Description:** Represents the duration a car was parked in hours.

* **chargeAmount**

**Type:** Double

**Description:** Represents the charge amount for parking, calculated based on the duration and capped at $20 per hour.

* **data**

**Type:** Table

**Description:** Represents the data read from an Excel file (NewRecoeds.xlsx) into a table.

* **fullFilePath**

**Type:** String

**Description:** Represents the full file path of the Excel file (NewRecoeds.xlsx) used to read data.

* **existingData**

**Type:** Table

**Description:** Contains existing data loaded from the Excel file when appending new data.

* **newData**

**Type:** Table

**Description:** Represents new data to be added to the existing data in the Excel file.

* **updatedData**

**Type:** Table

**Description:** Contains the data obtained after appending new data to the existing data.

* **validDays**

**Type:** Integer array

**Description**: Contains the days with recorded data.

* **uitableData**

**Type:** Cell array

**Description:** Represents data to be displayed in a uitable for specific tasks, such as showing average earnings.

* **avgEarningsPerCarPerDay**

**Type:** Double array

**Description:** Contains the average earnings per car per day.

* **dailyEarnings**

**Type:** Double array

**Description:** Contains the total earnings per day.

* **minEarnings**

**Type:** Double array

**Description:** Contains the minimum earnings per car for each day.

* **maxEarnings**

**Type:** Double array

**Description:** Contains the maximum earnings per car for each day.

* **totalRevenue**

**Type:** Double

**Description**: Represents the total revenue obtained by accumulating charge amounts over time.

* **resultString**

**Type:** String

**Description:** Represents a string containing information or results obtained from external function calls.

## **Descriptions Of The Tasks That Need To Be Performed To Solve The Problem :**

# **Loading and Displaying Car Park Layout Image:**

# The GUI loads an image of the car park layout from the file path 'C:\Users\ruchi\OneDrive\Desktop\MatlabProject\New folder\Images\carPark.jpg'.

# **Reading Data and Displaying Current Car Park Status:**

# Upon opening, the GUI reads data from the Excel file 'NewRecoeds.xlsx' and displays it in a table (uitable1), presenting the current car park status.

# **Entering a Car:**

# Users can enter a car by providing the number plate in the 'edit1' field. This action records the entry time and updates the car park status in a separate table (uitable2). The data includes the number plate, entry time, exit time (initially empty), and charge amount (initially empty).

# **Exiting a Car:**

# Users can exit a car by providing the number plate in the 'edit2' field. This action records the exit time, calculates the parking duration, determines the charge amount (capped at $20 per hour), and updates the car park status in the same table (uitable2).

# **Saving Data to Excel File:**

# The GUI provides a 'Save to Excel' button (saveexbtn) to save the updated car park status data. It checks if the Excel file 'NewRecoeds.xlsx' exists. If it does, it appends new data; otherwise, it creates a new file.

# **Displaying Daily Earnings:**

# The GUI includes a 'Show Daily Earnings' button (Showavg). Clicking this button calculates and displays the average earnings per car per day in a new table (uitable3).

# **Displaying Total Earnings:**

# The 'Total Earnings' button (plottotal) calculates and displays the total earnings per car over days in a bar plot.

# **Displaying Min and Max Earnings:**

# The 'Min and Max Earnings' button (subplotbtn) generates subplots showing the minimum and maximum earnings per car over days.

# **Calculating and Displaying Total Revenue:**

# The 'Show Revenue' button (showrevenuebtn) calculates the total revenue by integrating the charge amounts over time intervals. The result is displayed in a label (intlbl).

# **Calculating Annual Revenue (Additional Functionality):**

# There is an 'Annual Revenue' button (RequiredAnualRevenue) that invokes an external function (calculateAnnualRevenue()) to provide information about revenue and profit.

# **Calculating Car Park and Car Space Area (Additional Functionality):**

# The 'Car Park Area' button (areabtn) calls an external function (CarParkAndCArSpaceArea()) to calculate and display information related to the car park and car space area.

# **THE SOLUTION METHOD**

## **The Calculations Required To Solve Each Task Of The Problem :**

## **Task 1: Car Entry**

## **User Input:**

## Retrieve the car number plate from the edit1 field (numberPlate variable).

## **Table Update:**

## Append a new entry to the existing table (uitable2) with columns: CarPlate, EntryTime, ExitTime, and ChargeAmount.

## EntryTime is recorded as the current date and time (datestr(datetime('now', 'Format', 'yyyy-MM-dd HH:mm:ss'))).

## **Task 2: Car Exit**

## **User Input:**

## Retrieve the car number plate from the edit2 field (numberPlate variable).

## **Table Update:**

## Find the corresponding entry in the table (uitable2).

## Record the exit time (datestr(datetime('now', 'Format', 'yyyy-MM-dd HH:mm:ss'))).

## Calculate the entry and exit times and the duration of the car parked.

## Calculate the charge amount based on the parking duration with a cap of $20 per hour.

## **Task 3: Data Display**

## **Excel Reading:**

## Read data from an Excel file (NewRecoeds.xlsx) into a table.

## **Table Display:**

## Display the data in the uitable1 table.

## **Task 4: Data Saving**

## **Excel Writing:**

## Check if the Excel file exists.

## If it exists, load existing data, convert to a table, and append new data.

## If it doesn't exist, create a new Excel file with the current table data.

## **Task 5: Average Earnings**

## **Excel Reading:**

## Read data from an Excel file (NewRecoeds.xlsx) into a table.

## **Data Transformation:**

## Convert entryTime and exitTime to datetime.

## Calculate parked duration in hours for each car.

## Calculate daily earnings using accumarray.

## **Table Display:**

## Display the average earnings per car per day in the uitable3 table.

## **Task 6: Total Earnings**

## **Excel Reading:**

## Read data from an Excel file (NewRecoeds.xlsx) into a table.

## **Data Transformation:**

## Convert entryTime and exitTime to datetime.

## Calculate parked duration in hours for each car.

## Calculate daily earnings using accumarray.

## **Plotting:**

## Plot the total earnings per car over days using a bar plot.

## **Task 7: Min and Max Earnings**

## **Excel Reading:**

## Read data from an Excel file (NewRecoeds.xlsx) into a table.

## **Data Transformation:**

## Convert entryTime to datetime.

## Calculate daily earnings using accumarray.

## **Plotting:**

## Create subplots to show the minimum and maximum earnings per car over days.

## **Task 8: Total Revenue**

## **Excel Reading:**

## Read charge amounts from an Excel file (NewRecoeds.xlsx).

## **Calculation:**

## Assume charge amounts are in dollars per hour.

## Use trapz to approximate the total revenue.

## **Task 9: Annual Revenue and Profit**

## **External Function Call:**

## Call an external function (calculateAnnualRevenue).

## **Task 10: Area Calculation**

## **External Function Call:**

## Call an external function (CarParkAndCArSpaceArea).

## **The Equations Used To Solve The Problem :**

## **Car Entry:**

## The entercarbtn\_Callback function is triggered when the "Enter Car" button is pressed.

## It retrieves the car number plate from the edit1 (license plate input) field.

## Records the entry time along with the car number plate in a table (uitable2).

## **Car Exit:**

## The exitcarbtn\_Callback function is triggered when the "Exit Car" button is pressed.

## It retrieves the car number plate from the edit2 (license plate input) field.

## Records the exit time, calculates the parking duration, and charges the car based on the duration in the table (uitable2).

## **Data Display:**

## The showbtn\_Callback function reads data from an Excel file and displays it in a table (uitable1).

## **Data Saving:**

## The saveexbtn\_Callback function saves the car parking records to an Excel file (NewRecoeds.xlsx).

## **Average Earnings:**

## The Showavg\_Callback function calculates and displays the average earnings per car per day in a table (uitable3).

## **Total Earnings:**

## The plottotal\_Callback function plots the total earnings per car over days.

## **Min and Max Earnings:**

## The subplotbtn\_Callback function creates subplots to show the minimum and maximum earnings per car over days.

## **Total Revenue:**

## The showrevenuebtn\_Callback function calculates and displays the total revenue based on charge amounts and time intervals.

## **Annual Revenue and Profit:**

## The RequiredAnualRevenue\_Callback function calls an external function (calculateAnnualRevenue) and displays the result in a message box.

## **Area Calculation:**

## The areabtn\_Callback function calls an external function (CarParkAndCArSpaceArea) and displays the result in a message box.

## **The Definitions Of The Symbols Used In That Equations :**

1. **Symbol:** CarParkGuiNew

**Definition:** The main GUI window for the car park application.

1. **Symbol:** fullFilePath

**Definition:** The full path to the Excel file where car park data is stored.

1. **Symbol:** data

**Definition:** The variable holding the data read from the Excel file.

1. **Symbol:** handles

**Definition:** A structure that contains the handles of all graphical components in the GUI. It allows access to these components within different functions.

1. **Symbol:** hObject

**Definition:** The handle to the GUI figure. It represents the object that invokes the callback function.

1. **Symbol:** eventData

**Definition:** Data associated with the event that triggered the callback function.

1. **Symbol:** varargin

**Definition:** Variable-length input argument list. It allows the GUI functions to accept additional inputs.

1. **Symbol:** Excelfilepath

**Definition:** A global variable storing the path to the Excel file where car park data is stored**.**

1. **Symbol:** entryTime

**Definition:** The entry time of cars into the car park.

1. **Symbol:** exitTime

**Definition:** The exit time of cars from the car park.

1. **Symbol:** parkedDuration

**Definition:** The duration for which cars are parked in the car park.

1. **Symbol:** dailyEarnings

**Definition:** Daily earnings from the car park.

1. **Symbol:** totalCarsPerDay

**Definition:** The total number of cars parked per day.

1. **Symbol:** avgEarningsPerCarPerDay

**Definition:** Average earnings per car per day.

1. **Symbol:** validDays

**Definition:** Days with recorded data.

1. **Symbol:** uitableData

**Definition**: Cell array storing data for the uitable component in the GUI**.**

1. **Symbol:** minEarnings

**Definition:** Array storing minimum earnings for each day**.**

1. **Symbol:** maxEarnings

**Definition:** Array storing maximum earnings for each day.

1. **Symbol:** totalRevenue

**Definition:** Total revenue from the car park.

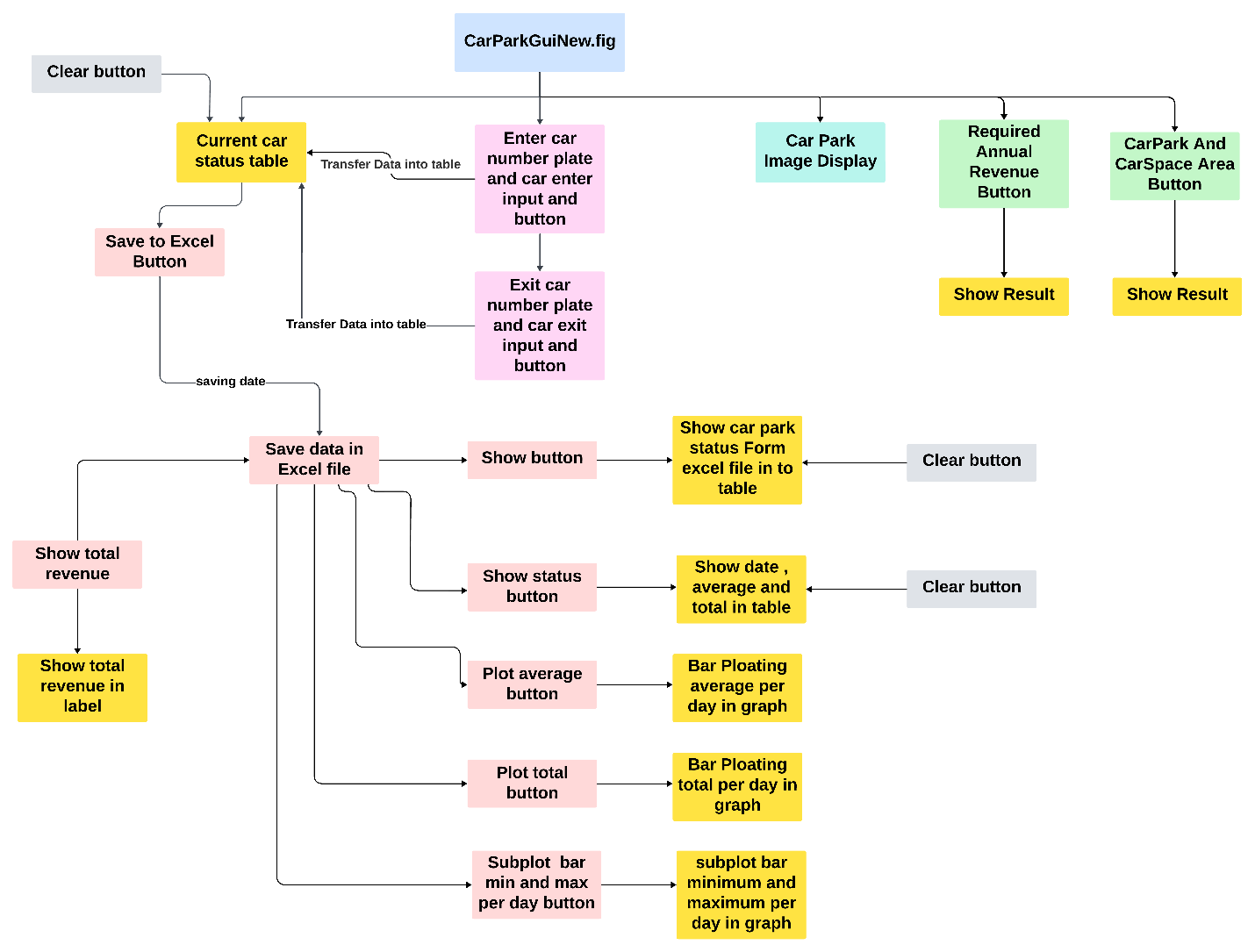
1. **Symbol:** resultString

**Definition:** A string variable holding the result of a calculation.

1. **Symbol:** img

**Definition:** The image read from the specified file path.

# **STRUCTURE CHART OF THE PROGRAM**

****

**Responsible Codes For Each Member:**

**1 . 21815070 Ruchira Dissanayaka :**

* + Car park image display
  + Enter and Exit Car number plate input and button
  + Current car park status table
  + Save to excel file button
  + Show data from Excel
  + Plot total
  + Subplot min and max
  + Plot average
  + Required annual revenue

**2.** 21809433 Emmanuel Ezeeli

* + Car park and car space area
  + Show total revenue

**Function that call from another file**

**Auto loading funcnction**

**Clear Button**

**Button**

**Show Result/Plot/Table**

**Input with button**

**Main Program**

# **PSEUDOCODE OF THE PROGRAM**

# # GUI Initialization

# function CarParkGuiNew\_OpeningFcn(hObject, eventdata, handles, varargin)

# # Load car park image

# img = imread('path\_to\_carParkImage.jpg');

# imshow(img, 'Parent', handles.axesImage);

# # Data Display

# function showbtn\_Callback(hObject, eventdata, handles)

# # Read data from Excel file

# data = readtable('path\_to\_data.xlsx');

# # Display data in uitable1

# set(handles.uitable1, 'Data', table2cell(data));

# # Car Entry

# function entercarbtn\_Callback(hObject, eventdata, handles)

# # Get car number plate from edit1

# numberPlate = get(handles.edit1, 'String');

# # Get current data from uitable2

# currentData = get(handles.uitable2, 'Data');

# # Create new entry

# newEntry = {numberPlate, currentDateTime, '', ''};

# # Concatenate new entry with existing data

# updatedData = [newEntry; currentData];

# # Update uitable2

# set(handles.uitable2, 'Data', updatedData);

# # Display success message

# # Car Exit

# function exitcarbtn\_Callback(hObject, eventdata, handles)

# # Get car number plate from edit2

# numberPlate = get(handles.edit2, 'String');

# # Get current data from uitable2

# currentData = get(handles.uitable2, 'Data');

# # Find corresponding entry

# entryIndex = find(strcmp(currentData(:, 1), numberPlate), 1);

# # Update exit time and charge amount

# currentData{entryIndex, 3} = currentDateTime;

# currentData{entryIndex, 4} = calculateChargeAmount(entryTime, exitTime);

# # Update uitable2

# set(handles.uitable2, 'Data', currentData);

# # Display success message

# # Save Data to Excel

# function saveexbtn\_Callback(hObject, eventdata, handles)

# # Get current data from uitable2

# currentData = get(handles.uitable2, 'Data');

# # Save data to Excel file

# saveToExcel(currentData);

# # Show Average Earnings

# function Showavg\_Callback(hObject, eventdata, handles)

# # Read data from Excel file

# carParkTable = readtable('path\_to\_data.xlsx');

# # Calculate and display average earnings per car per day in uitable3

# # Plot Average Earnings

# function plotavg\_Callback(hObject, eventdata, handles)

# # Read data from Excel file

# carParkTable = readtable('path\_to\_data.xlsx');

# # Plot average earnings per car per day using bar plot

# # Show Total Earnings

# function showsum\_Callback(hObject, eventdata, handles)

# # Read data from Excel file

# carParkTable = readtable('path\_to\_data.xlsx');

# # Calculate and display total earnings per car per day in uitable3

# # Plot Total Earnings

# function plottotal\_Callback(hObject, eventdata, handles)

# # Read data from Excel file

# carParkTable = readtable('path\_to\_data.xlsx');

# # Plot total earnings per car per day using bar plot

# # Plot Min and Max Earnings

# function subplotbtn\_Callback(hObject, eventdata, handles)

# # Read data from Excel file

# carParkTable = readtable('path\_to\_data.xlsx');

# # Plot min and max earnings per car per day using subplots

# # Show Total Revenue

# function showrevenuebtn\_Callback(hObject, eventdata, handles)

# # Read charge amounts from Excel file

# chargeAmounts = readChargeAmounts('path\_to\_data.xlsx');

# # Calculate and display total revenue in a label

# # Show Required Annual Revenue

# function RequiredAnualRevenue\_Callback(hObject, eventdata, handles)

# # Call calculateAnnualRevenue function and display result in msgbox

# # Calculate Area of Car Park and Car Space

# function areabtn\_Callback(hObject, eventdata, handles)

# # Call CarParkAndCArSpaceArea function and display result in msgbox

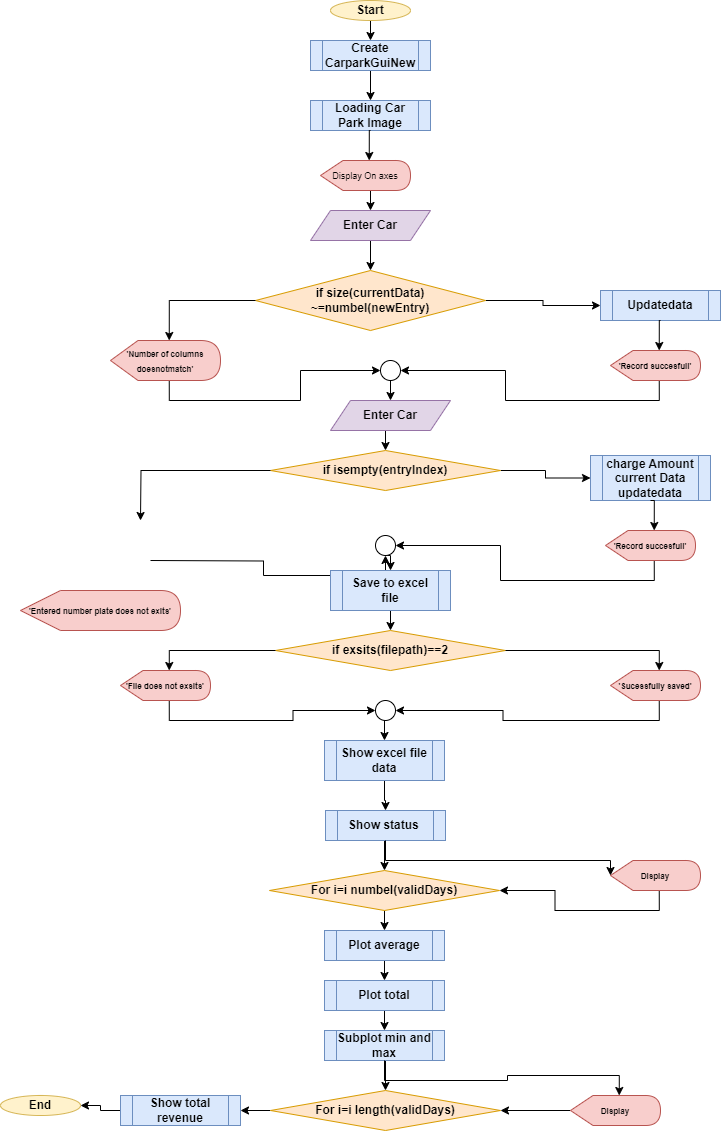
# # GUI Callbacks

# function CarParkGuiNew\_OutputFcn(hObject, eventdata, handles)

# # Get default command line output from handles structure

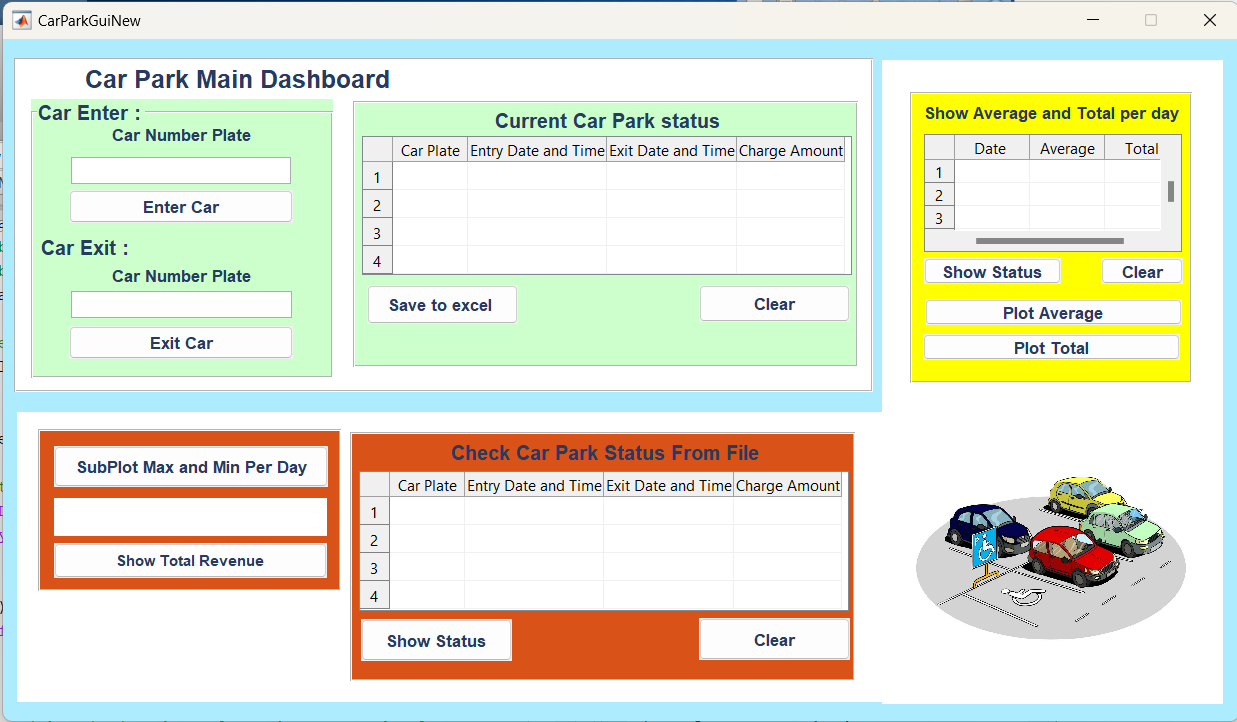
# varargout{1} = handles.output;

# **FLOWCHART OF THE PROGRAM**

****

# **EXPLANATION ON HOW THE RELATED MATLAB CODES AND DESIGNED GUI**

**Main Gui :**

****

**Image Loading :**

function CarParkGuiNew\_OpeningFcn(hObject, eventdata, handles, varargin)

% Choose default command line output for CarParkGuiNew

handles.output = hObject;

% Load the image from our excel file

img = imread('C:\Users\ruchi\OneDrive\Desktop\MatlabProject\New folder\Images\carPark.jpg');

% Make handles to put image on axes

handles.axesImage = findobj('Tag', 'axesImage');

% Display the image in the Axes

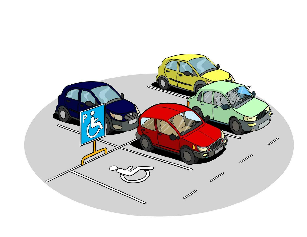
imshow(img, 'Parent', handles.axesImage);

% Update handles structure

guidata(hObject, handles);

% UIWAIT makes CarParkGuiNew wait for user response (see UIRESUME)

% uiwait(handles.figure1);

****

When the program running imread read image from the path and put that image on created axes name ‘axesImage’

**Enter Car :**

function entercarbtn\_Callback(hObject, eventdata, handles)

% Get the entered number plate

% edit1 is suppose to be car number plate editEdit

numberPlate = get(handles.edit1, 'String');

% Get the current table data uitable2 = Current Car park status

currentData = get(handles.uitable2, 'Data');

% Create a new entry

newEntry = {numberPlate, datestr(datetime('now', 'Format', 'yyyy-MM-dd HH:mm:ss')), '', ''};

% Display size information for debugging

disp(['Size of currentData: ' num2str(size(currentData))]);

disp(['Size of newEntry: ' num2str(size(newEntry))]);

% Ensure the new entry has the same number of columns as the existing data

if size(currentData, 2) ~= numel(newEntry)

errordlg('Number of columns in the new entry does not match the existing data.', 'Error', 'modal');

return;

end

% Concatenate new entry with existing data along the rows and columns

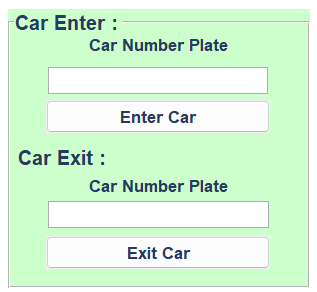
updatedData = [newEntry; currentData];

% Update the table in the GUI uitable2 = Current Car parl status

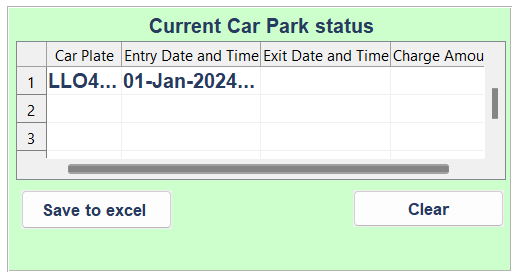
set(handles.uitable2, 'Data', updatedData);

% Inform the user

msgbox('Entry recorded successfully!', 'Success', 'modal');



This function work for enter car editText name ‘edit1’ gonna take input as car number pate and click enter car it gonna take number plate and current date and time then it will show in table name ‘uitable2’. When the data enter to ‘uitable2’ it will check current data and not equal to entry the updatedata in to table as enter car .Then show the ms msgbox('Entry recorded successfully!' , 'Success', 'modal');



**Exit Car :**

function exitcarbtn\_Callback(hObject, eventdata, handles)

% Get the entered number plate

% edit2 is suppose to be car number plate editEdit

numberPlate = get(handles.edit2, 'String');

% Get the current table data uitable2 = Current Car park status

currentData = get(handles.uitable2, 'Data');

% Find the corresponding entry

entryIndex = find(strcmp(currentData(:, 1), numberPlate), 1);

if isempty(entryIndex)

errordlg('No entry found for the given number plate.', 'Error', 'modal');

return;

end

% Update exit time

currentData{entryIndex, 3} = datestr(datetime('now', 'Format', 'yyyy-MM-dd HH:mm:ss'));

% Calculate entry and exit times

entryTime = datetime(currentData{entryIndex, 2}, 'InputFormat', 'dd-MMM-yyyy HH:mm:ss', 'Format', 'yyyy-MM-dd HH:mm:ss');

exitTime = datetime(currentData{entryIndex, 3}, 'InputFormat', 'dd-MMM-yyyy HH:mm:ss', 'Format', 'yyyy-MM-dd HH:mm:ss');

% Calculate duration in hours

entryDurationHours = hours(exitTime - entryTime);

% Calculate charge amount with a cap of $20 per hour

chargeAmount = max(entryDurationHours \* 20, 20);

% Update the charge amount in the table

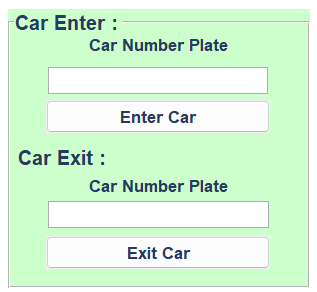
currentData{entryIndex, 4} = chargeAmount;

% Update the table in the GUI uitable2 = Current Car parl status

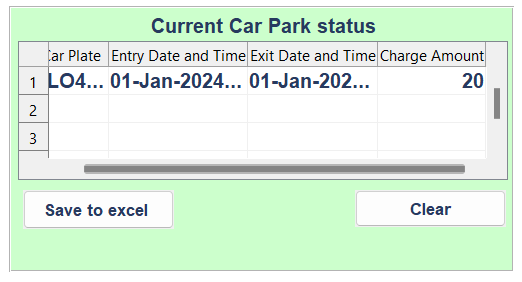
set(handles.uitable2, 'Data', currentData);

% Inform the user

msgbox('Exit recorded successfully!', 'Success', 'modal');



This function work for exit car editText name ‘edit2’ gonna take input as car number pate and click exit car then it gonna check using ‘find’ to number plate has in current data then take a current time and date then calculate the charge amount and set data into uitable2 as a exit car then show msgbox('Exit recorded successfully!', 'Success', 'modal');



**Current Car Park Status (Uitable2) :**

function saveexbtn\_Callback(hObject, eventdata, handles)

% Get the current table data

% Get the current table data

currentData = get(handles.uitable2, 'Data');

% Specify the Excel file path

filePath = 'C:\Users\ruchi\OneDrive\Desktop\MatlabProject\New folder\NewRecoeds.xlsx';

try

% Check if the Excel file already exists

if exist(filePath, 'file') == 2

% Load existing data from Excel file

existingData = readtable(filePath);

% Convert 'ChargeAmount' to cell if it is a numeric array in existing data

if ~iscell(existingData.ChargeAmount)

existingData.ChargeAmount = num2cell(existingData.ChargeAmount);

end

% Convert cell array to table

newData = cell2table(currentData, 'VariableNames', {'CarPlate', 'EntryTime', 'ExitTime', 'ChargeAmount'});

% Convert 'ChargeAmount' to cell if it is a numeric array in new data

if ~iscell(newData.ChargeAmount)

newData.ChargeAmount = num2cell(newData.ChargeAmount);

end

% Append new data to existing data

updatedData = [existingData; newData];

% Write the updated table to the Excel file

writetable(updatedData, filePath);

else

% If the file doesn't exist, create a new one

tableData = cell2table(currentData, 'VariableNames', {'CarPlate', 'EntryTime', 'ExitTime', 'ChargeAmount'});

writetable(tableData, filePath);

end

msgbox('Data saved to Excel file successfully!', 'Success', 'modal');

catch exception

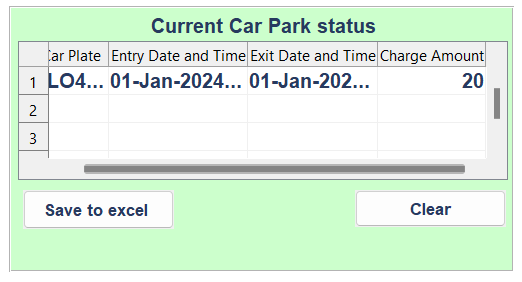
% Display the specific error message in the command window

disp(['Error saving data to Excel file: ' exception.message]);

% Display a more informative error message to the user

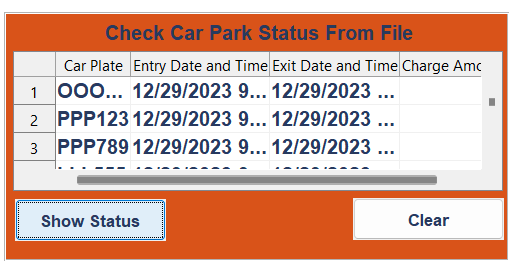
errordlg(['Error saving data to Excel file: ' exception.message], 'Error', 'modal');

end



This function work for save current car park status in to a excel file in provided file path according to the data to table of the excel file.when the ‘save to excel’ button clicked first it will check if excel file exsist then convert ‘chargeAmount’ to cell if it is a numeric array in thr=e exisiting data then convert cell array to table the convert ‘chargrAmount’ to cell if it is a numeric array in new data then append new data to existing data then write the updated table to the excel file. If the file does not exist create a new one then show msgbox('Data saved to Excel file successfully!', 'Success', 'modal');

**Check Carpark Status From File (Uitable1) :**



This function work for Read data from excel file using excel file path then display them in to uitable1 when the ‘show status’ clicked

% Specify the file path and name directly

fullFilePath = 'C:\Users\ruchi\OneDrive\Desktop\MatlabProject\New folder\NewRecoeds.xlsx';

% Read data from the Excel file

try

data = readtable(fullFilePath);

catch

errordlg('Error reading the Excel file. Make sure the file is valid.','Error','modal');

return;

end

% Display data in the uitable

set(handles.uitable1, 'Data', table2cell(data));

% Display data in the uitable

set(handles.uitable1, 'Data', table2cell(data));

**Show Average And Total Per Day (Uitable3) :**

function Showavg\_Callback(hObject, eventdata, handles)

% File path to the Excel file

excelFilePath = 'C:\Users\ruchi\OneDrive\Desktop\MatlabProject\New folder\NewRecoeds.xlsx';

% Read data from Excel file

carParkTable = readtable(excelFilePath);

% Convert entryTime and exitTime to datetime

entryTime = datetime(carParkTable.EntryTime, 'ConvertFrom', 'excel');

exitTime = datetime(carParkTable.ExitTime, 'ConvertFrom', 'excel');

% Calculate parked duration in hours for each car

parkedDuration = hours(exitTime - entryTime); %#ok<NASGU>

% Calculate daily earnings

dailyEarnings = accumarray(day(entryTime), carParkTable.ChargeAmount);

% Calculate total number of cars parked per day

totalCarsPerDay = accumarray(day(entryTime), 1);

% Calculate average earnings per car per day

avgEarningsPerCarPerDay = dailyEarnings ./ totalCarsPerDay;

% Find days with recorded data

validDays = find(~isnan(avgEarningsPerCarPerDay));

% Create a cell array to store the data for the uitable

uitableData = cell(numel(validDays), 2);

% Populate the uitableData with date and average earnings

for i = 1:numel(validDays)

dayIndex = validDays(i);

% Get the date corresponding to the dayIndex

currentDate = entryTime(day(entryTime) == dayIndex);

% Convert the date to a cell array of strings

formattedDate = cellstr(datestr(currentDate, 'dddd, mmmm dd, yyyy'));

uitableData{i, 1} = formattedDate{1}; % Extract the first element

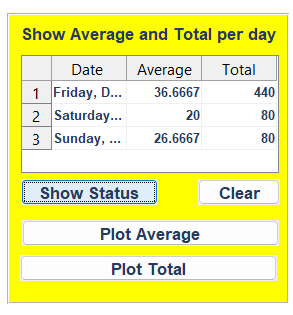
uitableData{i, 2} = avgEarningsPerCarPerDay(dayIndex);

uitableData{i, 3} = dailyEarnings(dayIndex);

end

% Display the uitable with date and average earnings

set(handles.uitable3, 'Data', uitableData);



This function work for show average and total per day when the ‘show status’ button clicked it going to read data from excel file using excel file path then take calculate the daily earning calculate a daily earning and calculate total number of car parked and calculate average earning per car per day the find a recorded day then create a cell array to store the data for uitable then get date corresponding to the dayIndex then convert the date to a cell array of strings and extract the first element then display the ‘uitable3’ with date and average and total earning.

**Plot Average Button :**

function plotavg\_Callback(hObject, eventdata, handles)

% File path to the Excel file

excelFilePath = 'C:\Users\ruchi\OneDrive\Desktop\MatlabProject\New folder\NewRecoeds.xlsx';

% Read data from Excel file

carParkTable = readtable(excelFilePath);

% Convert entryTime and exitTime to datetime

entryTime = datetime(carParkTable.EntryTime, 'ConvertFrom', 'excel');

exitTime = datetime(carParkTable.ExitTime, 'ConvertFrom', 'excel');

% Calculate parked duration in hours for each car

parkedDuration = hours(exitTime - entryTime);

% Calculate daily earnings

dailyEarnings = accumarray(day(entryTime), carParkTable.ChargeAmount);

% Calculate total number of cars parked per day

totalCarsPerDay = accumarray(day(entryTime), 1);

% Calculate average earnings per car per day

avgEarningsPerCarPerDay = dailyEarnings ./ totalCarsPerDay;

% Find days with recorded data

validDays = find(~isnan(avgEarningsPerCarPerDay));

% Plot the average earnings over days using a bar plot

figure;

bar(round(validDays), round(avgEarningsPerCarPerDay(validDays)), 'g', 'LineWidth', 1);

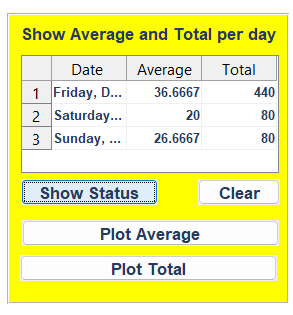
xlabel('Day');

ylabel('Average Earnings per Car');

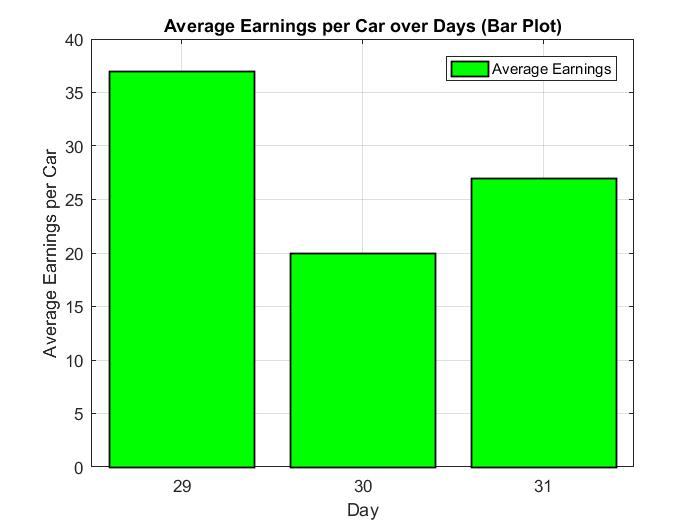
title('Average Earnings per Car over Days (Bar Plot)');

grid on;

legend('Average Earnings');



This function work for finding average per day and bar plot it according to the day and average.



**Plot Total Button :**

function plottotal\_Callback(hObject, eventdata, handles)

excelFilePath = 'C:\Users\ruchi\OneDrive\Desktop\MatlabProject\New folder\NewRecoeds.xlsx';

% Read data from Excel file

carParkTable = readtable(excelFilePath);

% Convert entryTime and exitTime to datetime

entryTime = datetime(carParkTable.EntryTime, 'ConvertFrom', 'excel');

exitTime = datetime(carParkTable.ExitTime, 'ConvertFrom', 'excel');

% Calculate parked duration in hours for each car

parkedDuration = hours(exitTime - entryTime);

% Calculate daily earnings

dailyEarnings = accumarray(day(entryTime), carParkTable.ChargeAmount);

% Find days with recorded data

validDays = find(~isnan(dailyEarnings));

% Create mesh grid for X and Y values

[X, Y] = meshgrid(1:31, validDays);

% Create a waterfall plot

figure;

waterfall(X, Y, repmat(dailyEarnings(validDays), 1, 31)');

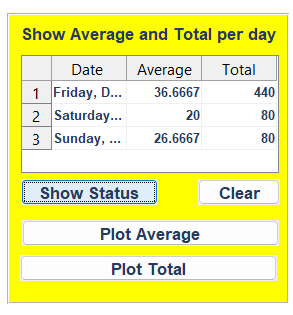
xlabel('Day');

ylabel('Recorded Days');

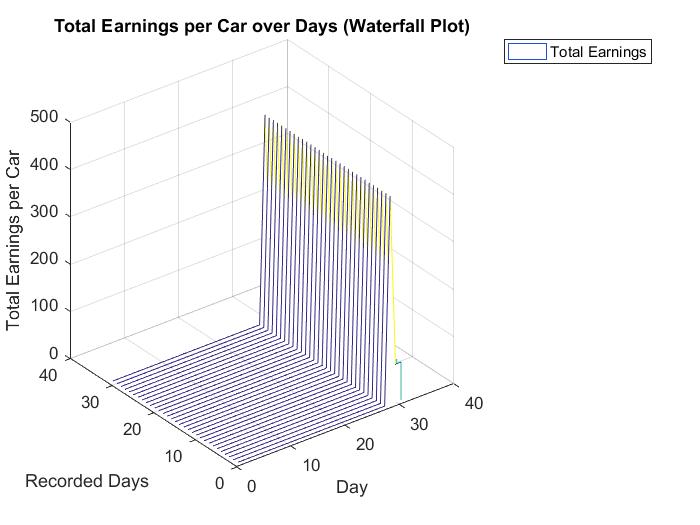
zlabel('Total Earnings per Car');

title('Total Earnings per Car over Days (Waterfall Plot)');

legend('Total Earnings');



This function work for finding Total per day and and waterfall it it according to the day and Total.



**Subplot Max And Min Per Day Button :**

function subplotbtn\_Callback(hObject, eventdata, handles)

% File path to the Excel file

excelFilePath = 'C:\Users\ruchi\OneDrive\Desktop\MatlabProject\New folder\NewRecoeds.xlsx';

% Read data from Excel file

carParkTable = readtable(excelFilePath);

% Convert entryTime to datetime

entryTime = datetime(carParkTable.EntryTime, 'ConvertFrom', 'excel');

% Calculate daily earnings

dailyEarnings = accumarray(day(entryTime), carParkTable.ChargeAmount, [], @(x) {x});

% Find days with recorded data

validDays = find(~cellfun('isempty', dailyEarnings));

% Initialize arrays to store min and max earnings for each day

minEarnings = zeros(length(validDays), 1);

maxEarnings = zeros(length(validDays), 1);

% Calculate min and max earnings for each day

for i = 1:length(validDays)

dayIndex = validDays(i);

earningsForDay = dailyEarnings{dayIndex};

minEarnings(i) = min(earningsForDay);

maxEarnings(i) = max(earningsForDay);

end

% Create separate subplots for min and max earnings

figure;

% Plot for Min Earnings

subplot(2, 1, 1);

bar(validDays, minEarnings, 'b', 'LineWidth', 2);

xlabel('Day');

ylabel('Min Earnings per Car');

title('Min Earnings per Car over Days');

grid on;

% Plot for Max Earnings

subplot(2, 1, 2);

bar(validDays, maxEarnings, 'r', 'LineWidth', 2);

xlabel('Day');

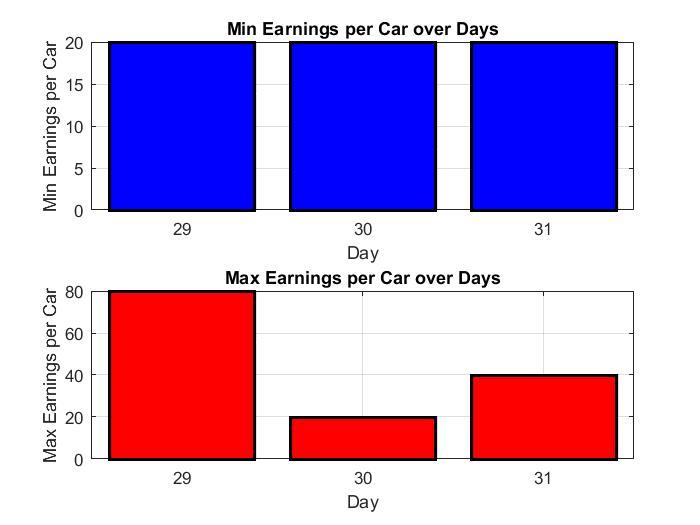
ylabel('Max Earnings per Car');

title('Max Earnings per Car over Days');

grid on;



This function work for finding Minimum and maximum per day and bar sub plot it according to the day and Maximum and minimum.



**Show Total revenue Button :**

function showrevenuebtn\_Callback(hObject, eventdata, handles)

% Specify the Excel file path

excelFilePath = 'C:\Users\ruchi\OneDrive\Desktop\MatlabProject\New folder\NewRecoeds.xlsx';

% Read charge amounts from Excel file

chargeAmounts = xlsread(excelFilePath);

% Assuming charge amounts are in dollars per hour, and you have the time intervals

timeIntervals = 1:length(chargeAmounts);

% Use trapz to approximate the total revenue

totalRevenue = trapz(timeIntervals, chargeAmounts);

% Display the result in the label

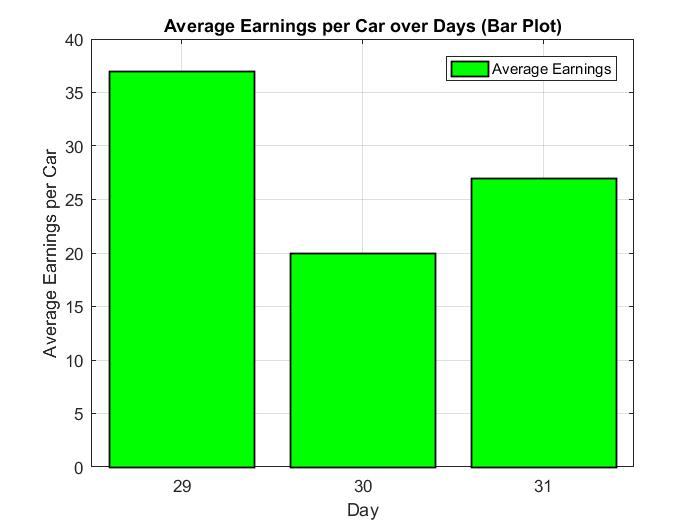
set(handles.intlbl, 'String', sprintf('Total Revenue: $%.2f', totalRevenue));



This function work for finding total revenue using ‘trapz’ using ‘timeIntervals and chargeAmount’ then it will show in ‘intlbl’ label as a total revenue

# **EXPLANATIONS OF ALL FIGURES**

**Average Earnings Bar Plot :**

****

function plotavg\_Callback(hObject, eventdata, handles)

% File path to the Excel file

excelFilePath = 'C:\Users\ruchi\OneDrive\Desktop\MatlabProject\New folder\NewRecoeds.xlsx';

% Read data from Excel file

carParkTable = readtable(excelFilePath);

% Convert entryTime and exitTime to datetime

entryTime = datetime(carParkTable.EntryTime, 'ConvertFrom', 'excel');

exitTime = datetime(carParkTable.ExitTime, 'ConvertFrom', 'excel');

% Calculate parked duration in hours for each car

parkedDuration = hours(exitTime - entryTime);

% Calculate daily earnings

dailyEarnings = accumarray(day(entryTime), carParkTable.ChargeAmount);

% Calculate total number of cars parked per day

totalCarsPerDay = accumarray(day(entryTime), 1);

% Calculate average earnings per car per day

avgEarningsPerCarPerDay = dailyEarnings ./ totalCarsPerDay;

% Find days with recorded data

validDays = find(~isnan(avgEarningsPerCarPerDay));

% Plot the average earnings over days using a bar plot

figure;

bar(round(validDays), round(avgEarningsPerCarPerDay(validDays)), 'g', 'LineWidth', 1);

xlabel('Day');

ylabel('Average Earnings per Car');

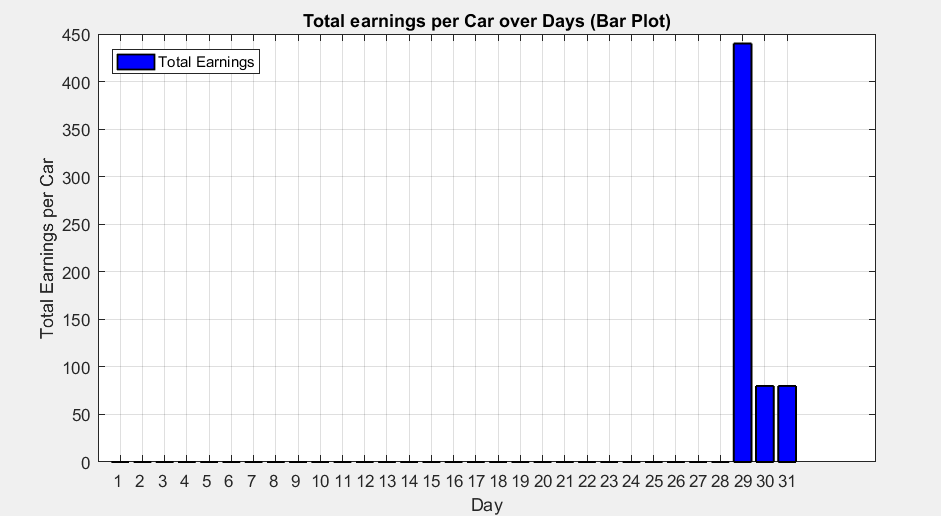
title('Average Earnings per Car over Days (Bar Plot)');

grid on;

legend('Average Earnings');

This figure used to show average earning per car each days. First when the ‘plotavg’ button clicke it will read data from the excel file from the given path then convert ‘entryTime’ and ‘exitTime’ to datetime then calculate parked duration in hour per each car to find how much charge form each car. Then calculate the daily earnings then calculate total number of cars per day then calculate average earning per car per day then find days with the recorded dare then plot average over days using bar plot.

**Total Earnings Bar Plot :**



function plottotal\_Callback(hObject, eventdata, handles)

excelFilePath = 'C:\Users\ruchi\OneDrive\Desktop\MatlabProject\New folder\NewRecoeds.xlsx';

% Read data from Excel file

carParkTable = readtable(excelFilePath);

% Convert entryTime and exitTime to datetime

entryTime = datetime(carParkTable.EntryTime, 'ConvertFrom', 'excel');

exitTime = datetime(carParkTable.ExitTime, 'ConvertFrom', 'excel');

% Calculate parked duration in hours for each car

parkedDuration = hours(exitTime - entryTime);

% Calculate daily earnings

dailyEarnings = accumarray(day(entryTime), carParkTable.ChargeAmount);

% Find days with recorded data

validDays = find(~isnan(dailyEarnings));

% Plot the total earnings over days using a bar plot

figure;

bar(round(validDays), round(dailyEarnings(validDays)), 'b', 'LineWidth', 1);

xlabel('Day');

ylabel('Total Earnings per Car');

title('Total earnings per Car over Days (Bar Plot)');

grid on;

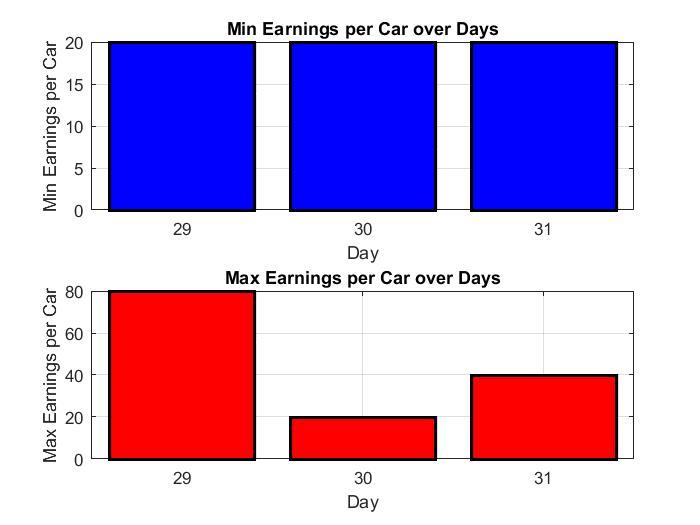
legend('Total Earnings');

set(gca, 'XTick', 1:31); % Set ticks from 1 to 31

set(gca, 'XTickLabel', 1:31); % Set labels from 1 to 31

This figure used to show total earning per day. First when ‘plottotal’ button clicked then it will read fata from excel file by given path then convert ‘entryTime’ and ‘exitTime’ to datetime and calculate parked duration in hours for each car then calculate dailyEarnings then find days with recorded data then plot the total earning over days using a bar plot.

**Subplot Min And Max Per Day :**

****

This function use to subplot minimum and maximum earning per each day. First reading data from excel file using the given path then calculate daily earnings then find days with recorded data and initialize arrays to store min and max earning for each day then calculate min and max earning for each day then create separate subplot for minimum and maximum earning per day.

function subplotbtn\_Callback(hObject, eventdata, handles)

% File path to the Excel file

excelFilePath = 'C:\Users\ruchi\OneDrive\Desktop\MatlabProject\New folder\NewRecoeds.xlsx';

% Read data from Excel file

carParkTable = readtable(excelFilePath);

% Convert entryTime to datetime

entryTime = datetime(carParkTable.EntryTime, 'ConvertFrom', 'excel');

% Calculate daily earnings

dailyEarnings = accumarray(day(entryTime), carParkTable.ChargeAmount, [], @(x) {x});

% Find days with recorded data

validDays = find(~cellfun('isempty', dailyEarnings));

% Initialize arrays to store min and max earnings for each day

minEarnings = zeros(length(validDays), 1);

maxEarnings = zeros(length(validDays), 1);

% Calculate min and max earnings for each day

for i = 1:length(validDays)

dayIndex = validDays(i);

earningsForDay = dailyEarnings{dayIndex};

minEarnings(i) = min(earningsForDay);

maxEarnings(i) = max(earningsForDay);

end

% Create separate subplots for min and max earnings

figure;

% Plot for Min Earnings

subplot(2, 1, 1);

bar(validDays, minEarnings, 'b', 'LineWidth', 2);

xlabel('Day');

ylabel('Min Earnings per Car');

title('Min Earnings per Car over Days');

grid on;

% Plot for Max Earnings

subplot(2, 1, 2);

bar(validDays, maxEarnings, 'r', 'LineWidth', 2);

xlabel('Day');

ylabel('Max Earnings per Car');

title('Max Earnings per Car over Days');

grid on;

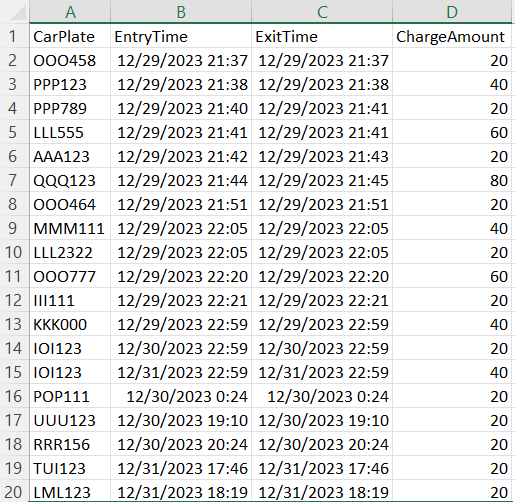
# **EXCEL FILE DATA**

This Data Recoded Run Through Program We Created From 3 Days.

Excel File Path = 'C:\Users\Ruchi\Onedrive\Desktop\Matlabproject\New Folder\Newrecoeds.Xlsx'

Excel File Name = Newrecoeds.Xlsx

Current Recorded Data In Excel File :



* This Data Records Are Saved According To The Running Program Entering Cars Number , Entry Time And Date , Exit Time And Date And The Charge Amount.

**Function And It Codes That We Calling From And M File**

**First Function:**

**Required annual revenue button :**

****

This function use to call the function in this m file calculateAnnualRevenue.m and display them in as disp(resultString) and

msgbox(resultString, 'Revenue and Profit Information', 'modal')

function RequiredAnualRevenue\_Callback(hObject, eventdata, handles)

% Call the function from the other file and get the result string

resultString = calculateAnnualRevenue();

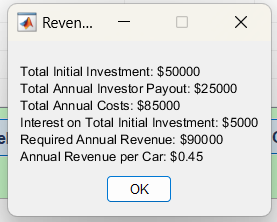
% Display the result in the command window

disp(resultString);

% Display the result in a msgbox

msgbox(resultString, 'Revenue and Profit Information', 'modal');

**Results in Gui :**



**Question and Function That Belong To That Calling Function**

**Question :** Annual maintenance cost per car parking is $10000 and employee salary $50000 and carpark charge $20 per hour or less than hour for one car . also we have 5 investors with the invest $10000 interest rate 10% per year.If one investor need to earn $5000 per year for next 10 year how much park should earn per year to cover all the cost and giving all investors money?

**Function :**

% calculateAnnualRevenue.m

function resultString = calculateAnnualRevenue()

% Given parameters

maintenanceCost = 10000;

employeeSalary = 50000;

carParkChargePerHour = 20;

numInvestors = 5;

investorInitialInvestment = 10000;

desiredProfitPerInvestor = 5000;

interestRate = 0.1;

numYears = 10;

numCars = 100; % Assuming 100 cars

numOperatingHoursPerYear = 2000; % Assuming 2000 operating hours per year

% Calculate total initial investment

totalInitialInvestment = numInvestors \* investorInitialInvestment;

% Calculate total annual investor payout

totalAnnualInvestorPayout = numInvestors \* desiredProfitPerInvestor;

% Calculate total annual costs

totalAnnualCosts = maintenanceCost + employeeSalary + totalAnnualInvestorPayout;

% Calculate interest on total initial investment

interestOnInitialInvestment = totalInitialInvestment \* interestRate;

% Calculate required annual revenue

requiredAnnualRevenue = totalAnnualCosts + interestOnInitialInvestment;

% Calculate annual revenue per car

annualRevenuePerCar = requiredAnnualRevenue / (numCars \* numOperatingHoursPerYear);

% Prepare the result string

resultString = sprintf('Total Initial Investment: $%s\nTotal Annual Investor Payout: $%s\nTotal Annual Costs: $%s\nInterest on Total Initial Investment: $%s\nRequired Annual Revenue: $%s\nAnnual Revenue per Car: $%s', ...

num2str(totalInitialInvestment), num2str(totalAnnualInvestorPayout), ...

num2str(totalAnnualCosts), num2str(interestOnInitialInvestment), ...

num2str(requiredAnnualRevenue), num2str(annualRevenuePerCar));

% Display the results

disp(resultString);

end

end

**Function Given Results :**

ans =

Total Initial Investment: $50000

Total Annual Investor Payout: $25000

Total Annual Costs: $85000

Interest on Total Initial Investment: $5000

Required Annual Revenue: $90000

Annual Revenue per Car: $0.45

**Second Function:**

**CarPark and CarSpace Area button :**

****

This function use to call the function in this m file CarParkAndCarSpaceArea.m and concatenate them and display them msgbox(resultString, 'Area Calculation', 'modal');

% Call the CarParkAndCArSpaceArea function and get the result string

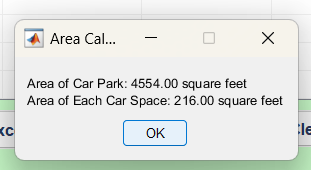
resultString = CarParkAndCArSpaceArea();

% Use strvcat to concatenate the result string

combinedString = strvcat(resultString);

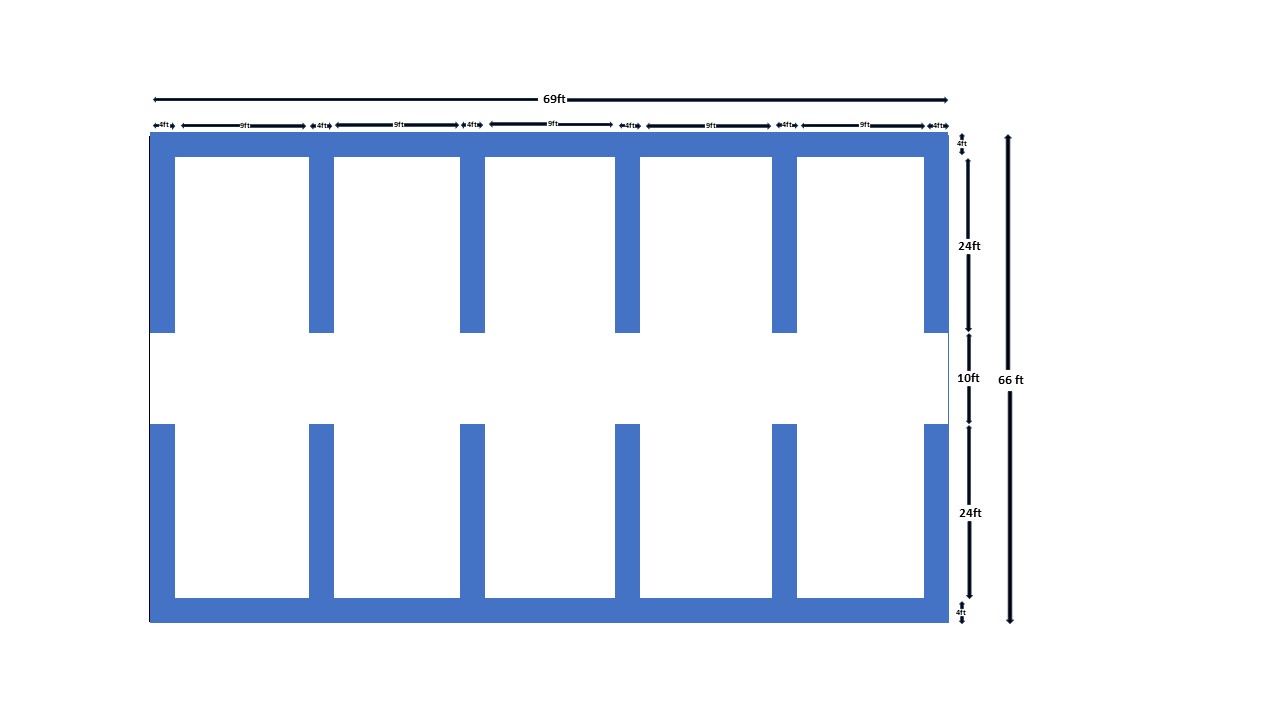
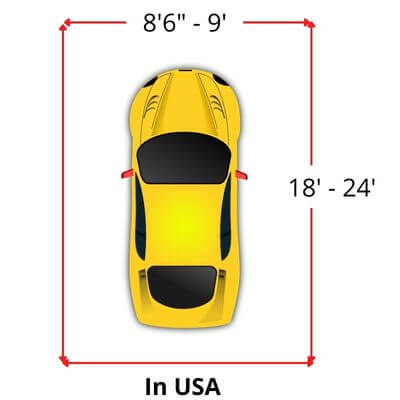
% Display the result string in a message box

msgbox(combinedString, 'Area Calculation', 'modal');

****

**Question and Function That Belong To That Calling Function**

**Question :** Length of the CarPark = 69ft and Width of the CarPark = 66ft , Length of the CarSpace = 24ft and Width of the CarSpcae = 9ft Find the area of the CarPark and CarSpace?



**Function :**

function resultString = CarParkAndCArSpaceArea()

carParkLength = 69;

carParkWidth = 66;

carSpaceWidth = 9;

carSpaceLength = 24;

carParkArea = carParkLength \* carParkWidth;

carSpaceArea = carSpaceLength \* carSpaceWidth;

resultString = sprintf('Area of Car Park: %.2f square feet\nArea of Each Car Space: %.2f square feet', carParkArea, carSpaceArea);

disp(resultString);

end

**Function Given Results :**

**ans =** Area of Car Park: 4554.00 square feet

Area of Each Car Space: 216.00 square feet