MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY

(Autonomous Institution – UGC, Govt. of India)

Affiliated to JNTUH Approved by AICTE, NBA- Tier 1 & NAAC – 'A' Grade l ISO 9001:2015 Certified)

Maisammaguda, Dhulapally (Post Via. Hakimpet), Secunderabad – 500100, Telangana State, India

APP DEVELOPMENT - I

B.TECH III YEAR – I SEM

NAME _____

ROLL NO: _____ BRANCH _____

YEAR: _____ SEM: _____

MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY

(Autonomous Institution – UGC, Govt. of India)

Affiliated to JNTUH Approved by AICTE, NBA- Tier 1 & NAAC – 'A' Grade ISO 9001:2015 Certified)

Maisammaguda, Dhulapally (Post Via. Hakimpet), Secunderabad – 500100, Telangana State, India

CERTIFICATE

Certified that this is the Bonafide Record of the work done by Mr. /Ms. _____ bearing Roll No. _____ of B.Tech III Year _____Semester for the Academic year 2022-2023 in _____

Date: **Faculty Incharge** **HOD**

Internal Examiner **External Examiner**

INDEX

S.No	Date	Project Title	Page No	Faculty Sign

INDEX

S.No	Date	Title	Page No	Faculty Sign

DEPARTMENT OF AERONAUTICAL ENGINEERING

VISION

Department of Aeronautical Engineering aims to be indispensable source in Aeronautical Engineering which has a zeal to provide the value driven platform for the students to acquire knowledge and empower themselves to shoulder higher responsibility in building a strong nation.

MISSION

a) The primary mission of the department is to promote engineering education and research.

(b) To strive consistently to provide quality education, keeping in pace with time and technology.

(c) Department passions to integrate the intellectual, spiritual, ethical and social development of the students for shaping them into dynamic engineers.

MALLA REDDY COLLEGE OF ENGINEERING AND TECHNOLOGY

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

PEO1: PROFESSIONALISM & CITIZENSHIP

To create and sustain a community of learning in which students acquire knowledge and learn to apply it professionally with due consideration for ethical, ecological and economic issues.

PEO2: TECHNICAL ACCOMPLISHMENTS

To provide knowledge based services to satisfy the needs of society and the industry by providing hands on experience in various technologies in core field.

PEO3: INVENTION, INNOVATION AND CREATIVITY

To make the students to design, experiment, analyze, interpret in the core field with the help of other multi disciplinary concepts wherever applicable.

PEO4: PROFESSIONAL DEVELOPMENT

To educate the students to disseminate research findings with good soft skills and become a successful entrepreneur.

PEO5: HUMAN RESOURCE DEVELOPMENT

To graduate the students in building national capabilities in technology, education and research.

MALLA REDDY COLLEGE OF ENGINEERING AND TECHNOLOGY

PROGRAM OUTCOMES (POs)

Engineering Graduates will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. **Design / development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5**. Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9**. Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multi disciplinary environments.

12. **Life- long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

MALLA REDDY COLLEGE OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF AERONAUTICAL ENGINEERING

PROGRAM SPECIFIC OBJECTIVES

1. To mould students to become a professional with all necessary skills, personality and sound knowledge in basic and advance technological areas.

2. To promote understanding of concepts and develop ability in design manufacture and maintenance of aircraft, aerospace vehicles and associated equipment and develop application capability of the concepts sciences to engineering design and processes.

3. Understanding the current scenario in the field of aeronautics and acquire ability to apply knowledge of engineering, science and mathematics to design and conduct experiments in the field of Aeronautical Engineering.

4. To develop leadership skills in our students necessary to shape the social, intellectual, business and technical worlds.

MALLA REDDY COLLEGE OF ENGINEERING AND TECHNOLOGY

III Year B.Tech. ANE- I Sem **L/T/P/C -/-/4/2**

(R20A2187) Application Development-I

LIST OF PROJECTS: -

1. Installation of Android studio.

2. Development of Hello World Application.

3. Create an application that takes the name from a textbox and shows hello message along with the name entered in textbox, when the user clicks the OK button.

4. Create a screen that has input boxes for User Name, Password, and Address, Gender (radio buttons for male and female), Age (numeric), Date of Birth (Date Picket), State (Spinner) and a Submit button. On clicking the submit button, print all the data below the Submit Button(use any layout).

5. Design an android application to create page using Intent and one Button and pass the Values from one Activity to second Activity.

6. Design an android application Send SMS using Intent.

7. Create an android application using Fragments.

8. Design an android application Using Radio buttons.

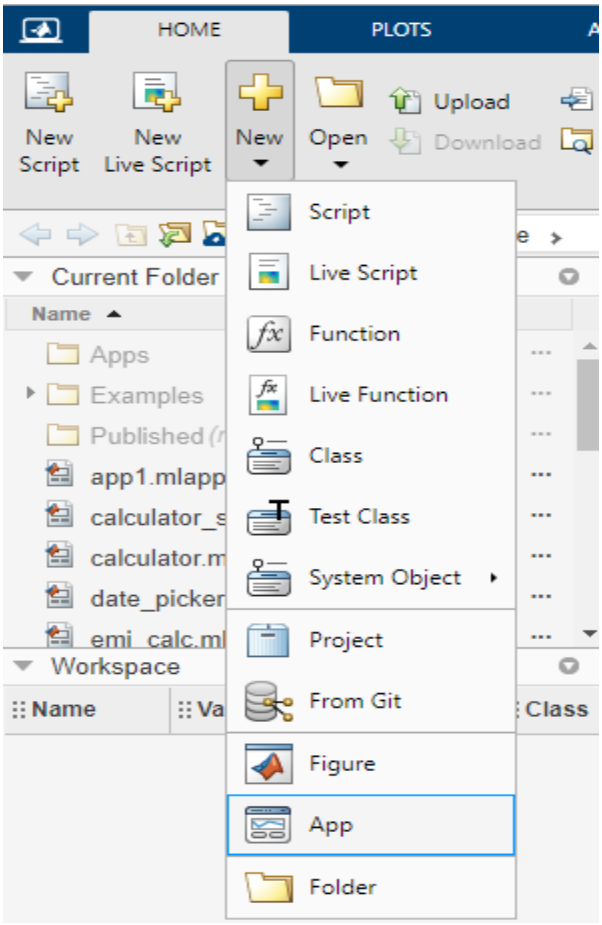
9. Design an android application for menu.

10. Create a user registration application that stores the user details in a database table

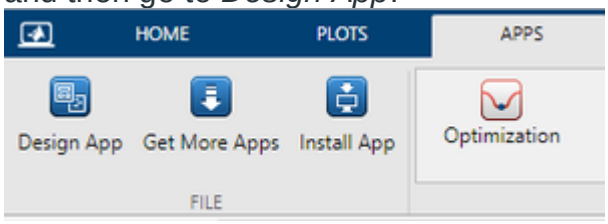
MATLAB® provides functions and tools to build interactive user interfaces. You can add components, such as buttons and sliders, to enable user interaction and include plots for data visualization and exploration in these interfaces.
To create self-contained interfaces that perform operations based on user interactions, develop apps. To create interfaces that can be embedded into a live script and that generate code as users explore parameters, develop Live Editor tasks. For more information, see Ways to Build Apps.
A large set of UI components are available for creating interfaces in MATLAB. You also can extend the list of available components with your own specialized UIs and visualizations by creating custom UI components.
App Designer is an interactive development environment for designing an app layout and programming its behavior. It provides a fully integrated version of the MATLAB® Editor and a large set of interactive UI components. It also offers a grid layout manager to organize your user interface, and automatic reflow options to make your app detect and respond to changes in screen size. It lets you distribute apps by packaging them into installer files directly from the App Designer toolstrip, or by creating a standalone desktop or web app (requires MATLAB Compiler™).

MATLAB is a powerful, high-level programming language. Matlab is widely used for designing systems by engineers and scientists and we all know that the best way to represent any idea is by using a simple but effective GUI. Matlab app builder provides you the power to build different apps, to represent your idea in a GUI-friendly manner.

Step 1: You can start working on the MATLAB APP Builder in two ways. Either go to *Home>New>App.*



Alternatively, for going to the Matlab app builder section, Select *Apps* from the Menubar, and then go to *Design App*.



Step 2: A new pop-up is opened. It provides a different layout for stating the app. It has also come examples for a better understanding.

Step 3: You can choose any App option to build a MATLAB app. There are mainly three layouts available on the go. These includes

- Blank App
- 2-Panel App with Auto-Reflow
- 3-Panel App with Auto-Reflow

Step 4: MATLAB consists of various components like:

- Components
- Customizing Components

Components are the pre-built shapes that are designed for particular tasks, and that could be imported to the design tab. In Matlab, the *Component Library* is situated in the leftmost part of the window.

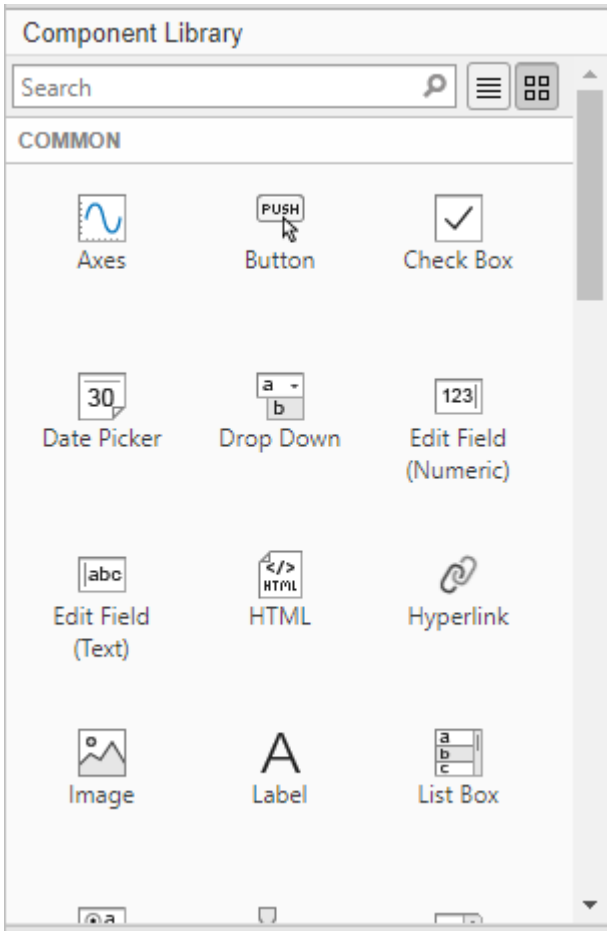
For importing any component to your design right-click on the component and drag it to the design tab and drop it wherever you want to place it.

By using customizing components you can customize your components as per your requirements, using the *Component browser.* It is situated in the rightmost part of the App Builder Window. Using the Component Browser you can change information about the component, Font and Color, Interactivity with the user, Position of the component in the design view, CallBack Execution Controls, Parents/Child members, and Identifiers. It also contains the list of the component adopted in your app.

App Designer and UI figures support a large set of components for designing modern, full-featured applications. The tables below list the components that are available.

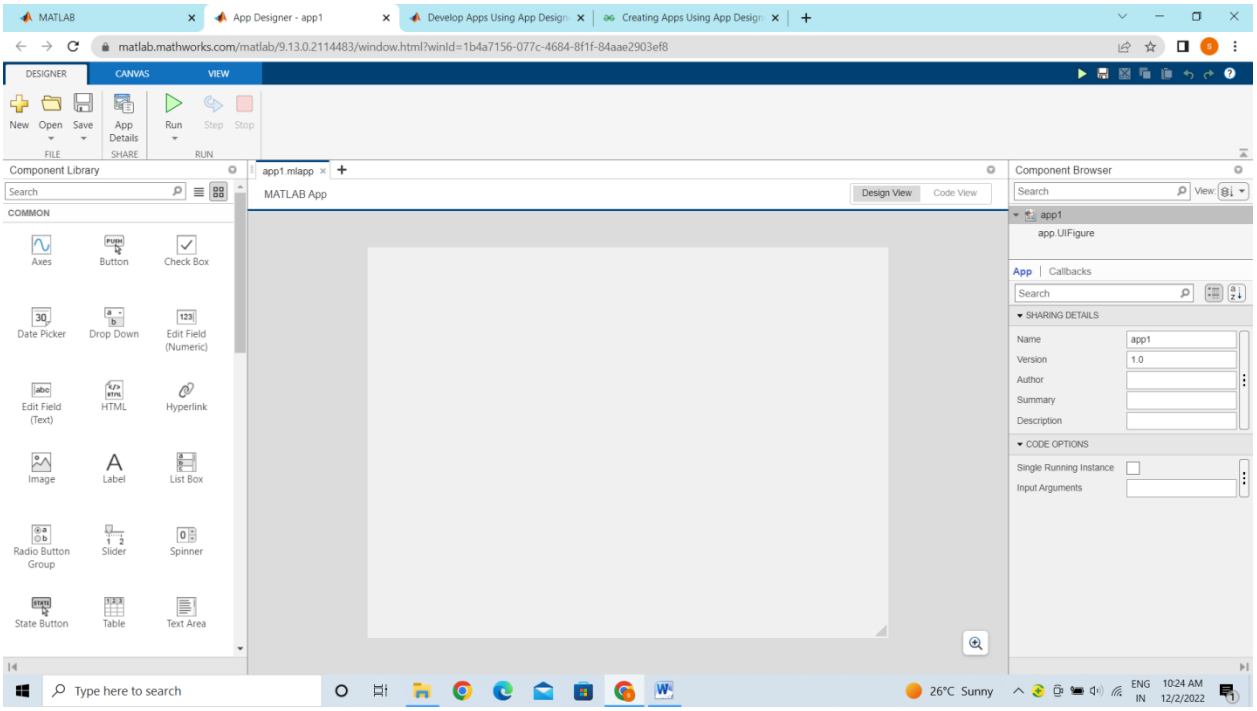
- Common Components — Include components that respond to interactions, such as buttons, sliders, drop-down lists, and trees.
- Axes — Include axes to create plots for data visualization and exploration.
- Containers and Figure Tools — Include panels and tabs for grouping components, as well as menu bars.
- Instrumentation Components — Include gauges and lamps for visualizing status, as well as knobs and switches for selecting input parameters.
- Extensible Components — Include custom UI components that you author. Interface with third-party libraries to display content like widgets or data visualizations.
- Toolbox Components — Include toolbox authored UI components. Requires additional toolbox license and installation.

All components are available programmatically. Most UI components are also available in the App Designer **Component Library** for you to drag and drop onto the canvas. To add components to an App Designer app that are not available in the **Component Library**, or that you want to add dynamically to the running app, see Add UI Components to App Designer Programmatically.



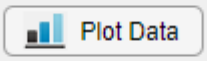
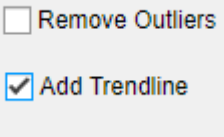
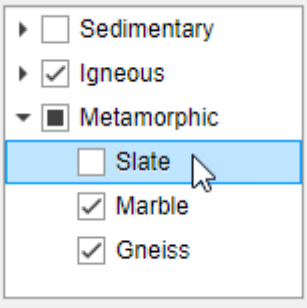
The Windows are switchable OR TOGGLE by selecting

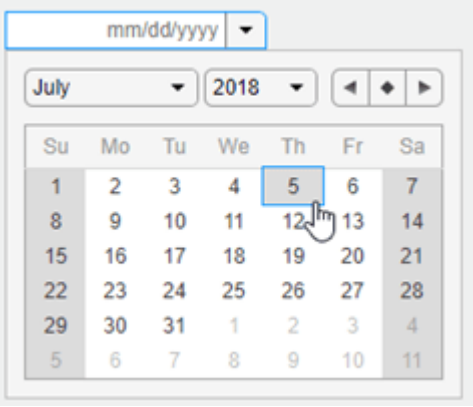
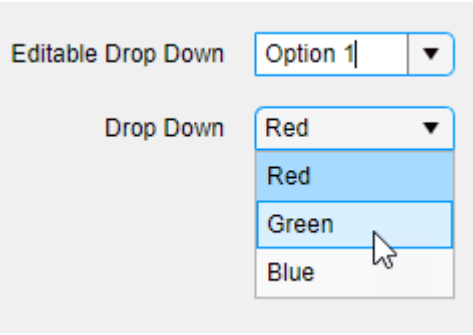
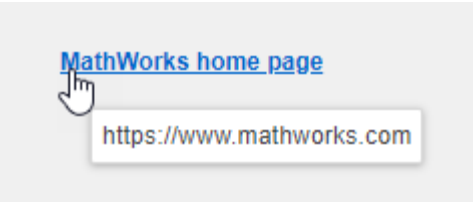
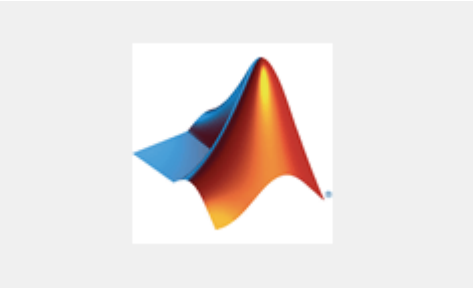
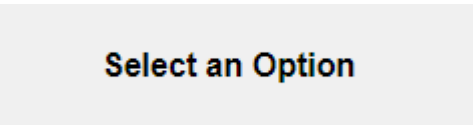
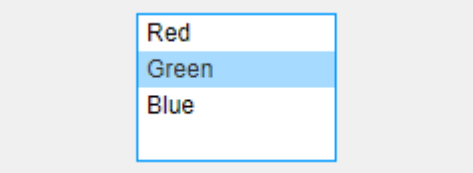
- DESIGN VIEW
- CODE VIEW

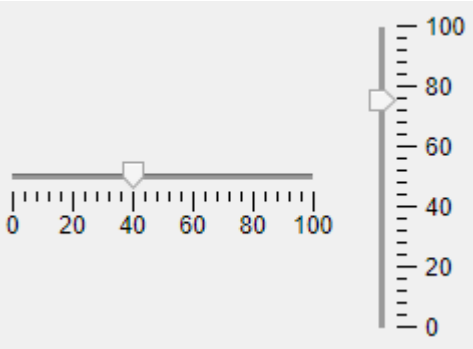
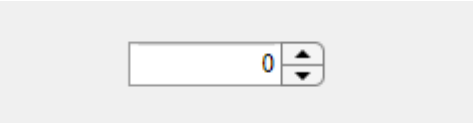
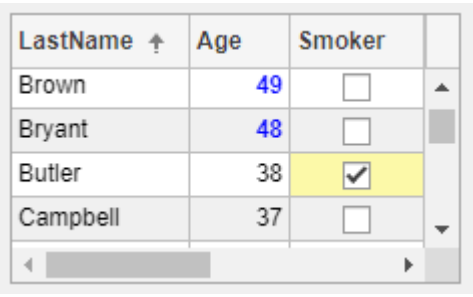
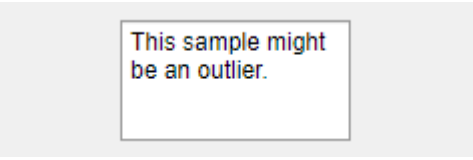
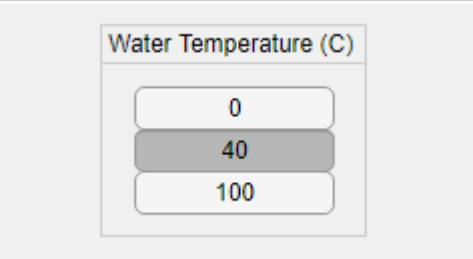


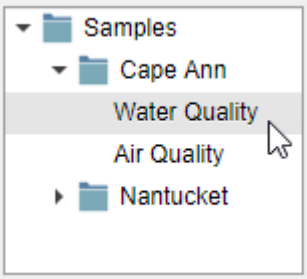
DESIGN VIEW:

When calling graphics functions in App Designer, the workflow is slightly different than you typically use at the MATLAB® command line. For more information about how to call graphics functions in App Designer, see Display Graphics in App Designer.

Component Information	Example
Button Properties	
CheckBox Properties	
CheckBoxTree Properties TreeNode Properties	

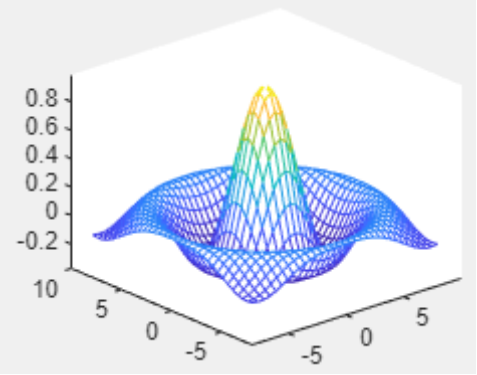
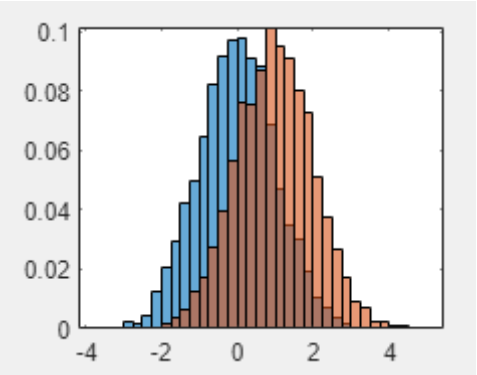
DatePicker Properties	
DropDown Properties	
NumericEditField Properties	
EditField Properties	
Hyperlink Properties	
Image Properties	
Label Properties	
ListBox Properties	

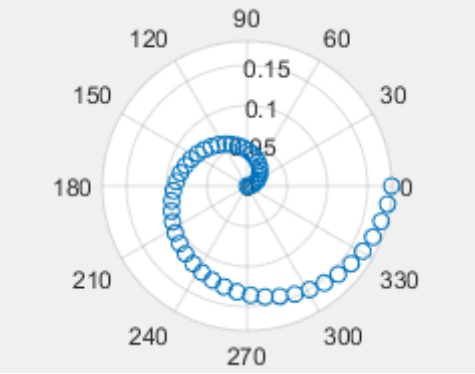
ButtonGroup Properties RadioButton Properties	
Slider Properties	
Spinner Properties	
StateButton Properties	
Table Properties	
TextArea Properties	
ButtonGroup Properties ToggleButton Properties	

Tree Properties TreeNode Properties	

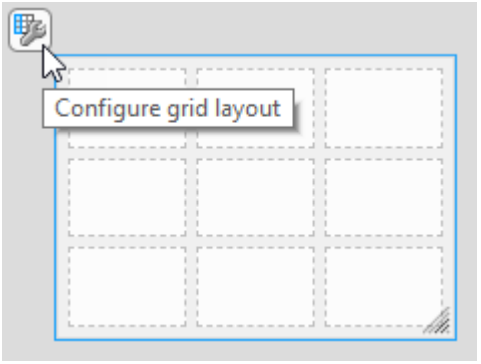
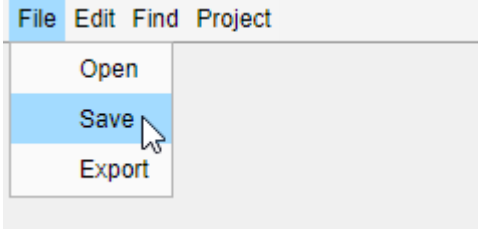
Common Components

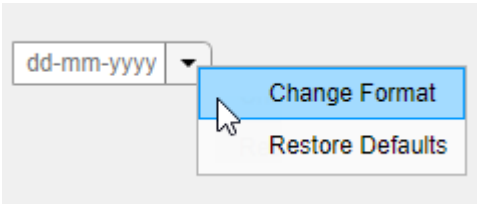
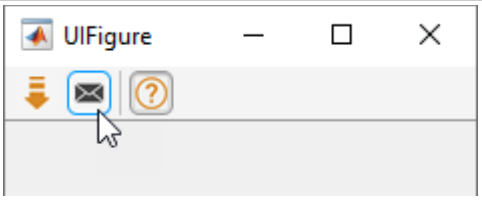
Axes

Axes Information	Example
-UIAxes Properties	
Axes Properties This object can be added programmatically only.	
GeographicAxes Properties This object can be added programmatically only.	

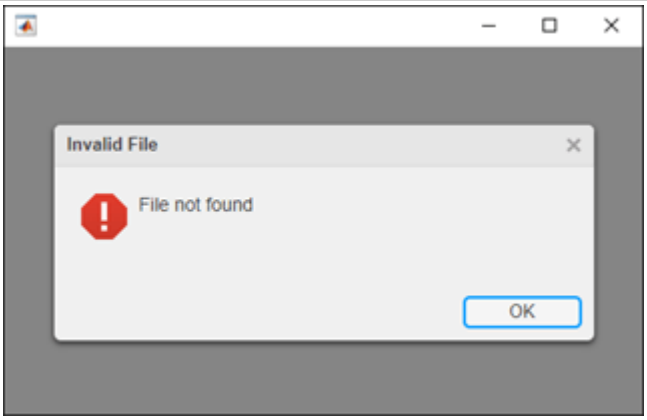
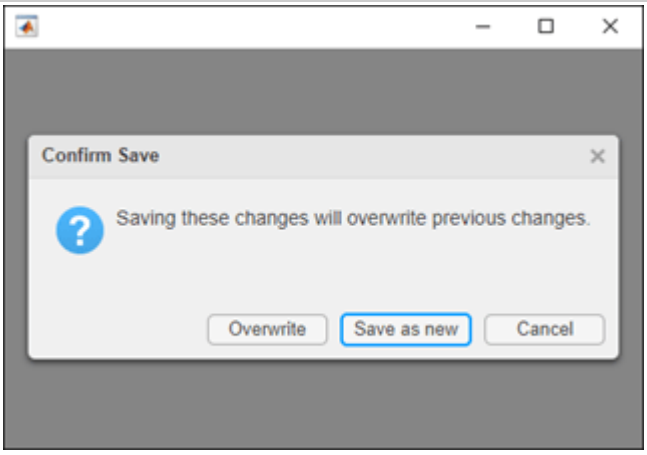
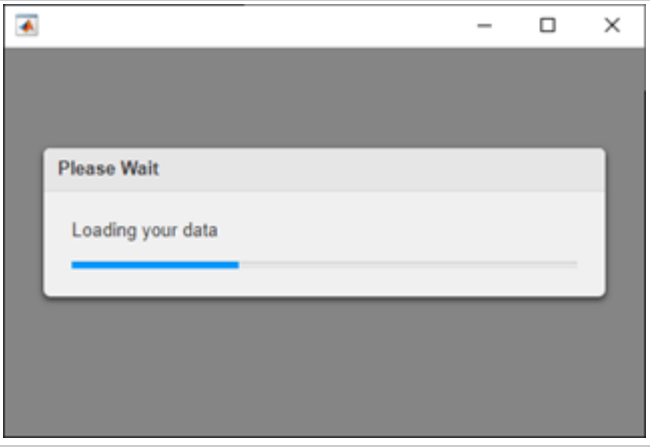
Axes Information	Example
PolarAxes Properties This object can be added programmatically only.	

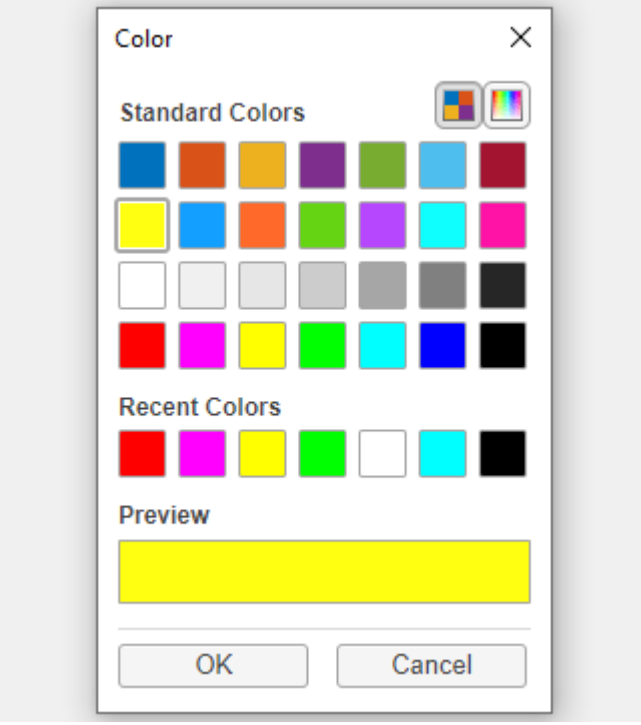
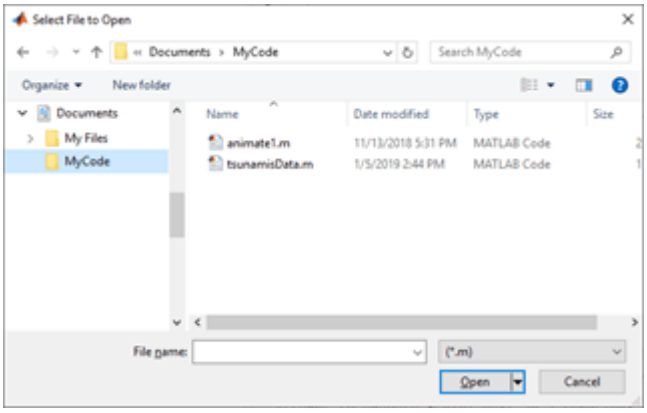
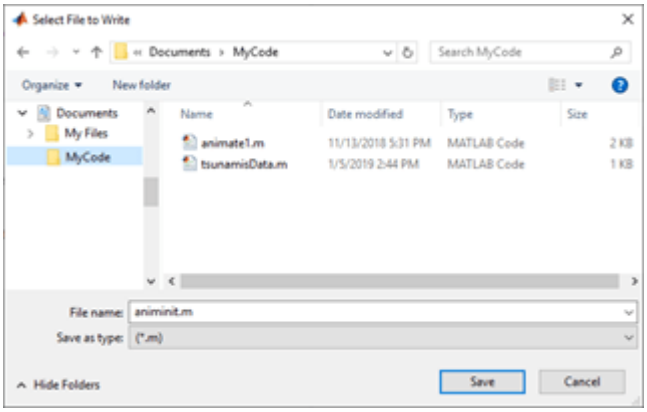
Containers and Figure Tools

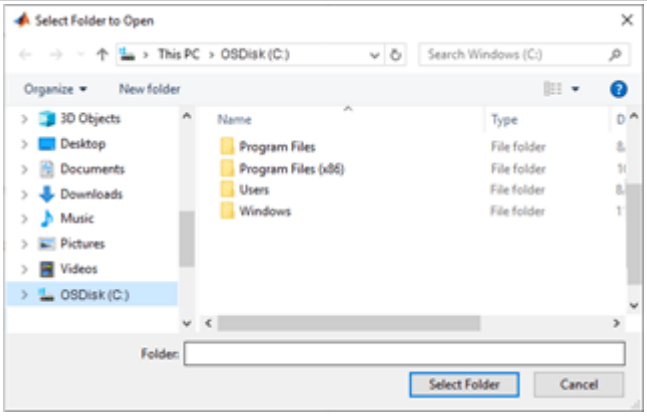
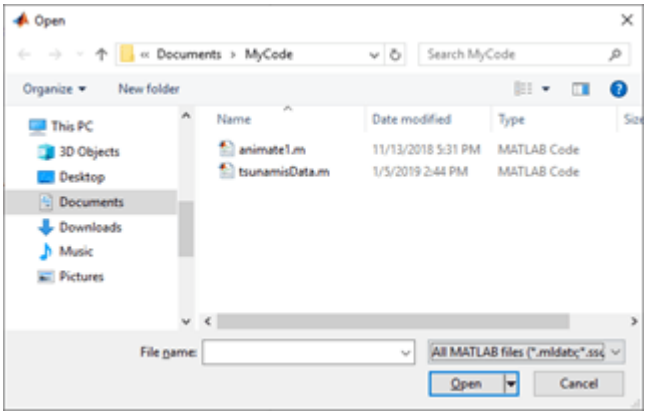
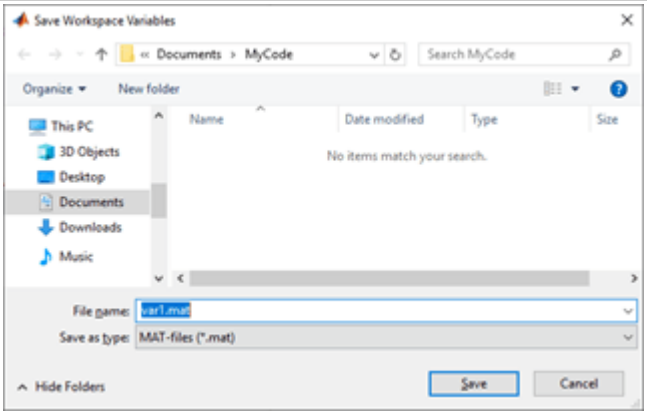
Container Information	Example
GridLayout Properties	
Panel Properties	
TabGroup Properties Tab Properties	
Menu Properties	

Container Information	Example
ContextMenu Properties	
Toolbar Properties PushTool Properties ToggleTool Properties	

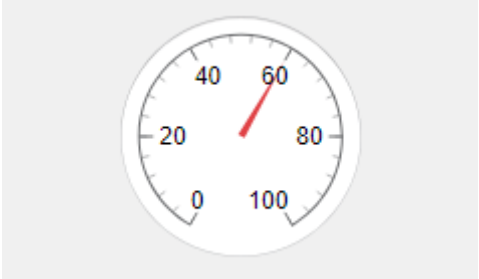
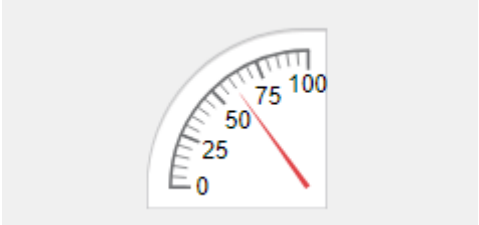
Dialogs and Notifications

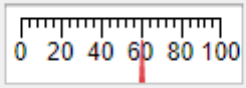
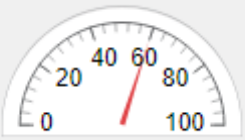
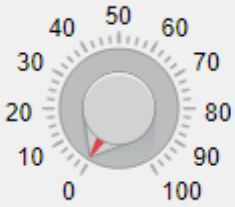
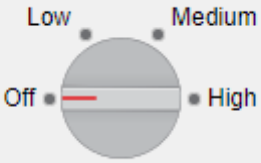
Dialog Information	Example
`uialert` This object can be added programmatically only.	
`uiconfirm` This object can be added programmatically only.	
`uiprogressdlg` This object can be added programmatically only.	

Dialog Information	Example
`uisetcolor` This object can be added programmatically only.	
`uigetfile` This object can be added programmatically only.	
`uiputfile` This object can be added programmatically only.	

Dialog Information	Example
`uigetdir` This object can be added programmatically only.	
`uiopen` This object can be added programmatically only.	
`uisave` This object can be added programmatically only.	

Instrumentation

Component Information	Example
Gauge Properties	
NinetyDegreeGauge Properties	

Component Information	Example
LinearGauge Properties	
SemicircularGauge Properties	
Knob Properties	
DiscreteKnob Properties	
Lamp Properties	
Switch Properties	
RockerSwitch Properties	
ToggleSwitch Properties	

SAMPLE PROJECT 1

TO DEVELOP AN WEB APP FOR PRINTING HELLO WORLD

Components Used:

Button

Numeric Data

Code:

```
classdef Helloworld < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure           matlab.ui.Figure
        ENTERURNAMEButton  matlab.ui.control.Button
        EditField          matlab.ui.control.EditField
        EditFieldLabel     matlab.ui.control.Label
        DISPLAYButton      matlab.ui.control.Button
    end

    % Callbacks that handle component events
    methods (Access = private)

        % Button pushed function: DISPLAYButton
        function DISPLAYButtonPushed(app, event)

            app.EditField.Value="HELLO WORLD";

        end

        % Callback function
        function ENTERYOURNAMEEditFieldValueChanged(app, event)
            value = app.ENTERYOURNAMEEditField.Value;
            app.ENTERYOURNAMEEditField.Value=input(prompt)
        end

        % Button pushed function: ENTERURNAMEButton
        function ENTERURNAMEButtonPushed(app, event)
            value.app.ENTERURNAMEButton=input(prompt)
        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.Position = [100 100 640 480];
            app.UIFigure.Name = 'MATLAB App';

            % Create DISPLAYButton
            app.DISPLAYButton = uibutton(app.UIFigure, 'push');
            app.DISPLAYButton.ButtonPushedFcn = createCallbackFcn(app,
@DISPLAYButtonPushed, true);
```

```
            app.DISPLAYButton.BackgroundColor = [1 0 1];
            app.DISPLAYButton.FontName = 'NanumGothic';
            app.DISPLAYButton.FontWeight = 'bold';
            app.DISPLAYButton.Position = [158 152 326 63];
            app.DISPLAYButton.Text = 'DISPLAY';

            % Create EditFieldLabel
            app.EditFieldLabel = uilabel(app.UIFigure);
            app.EditFieldLabel.HorizontalAlignment = 'right';
            app.EditFieldLabel.Position = [159 268 55 22];
            app.EditFieldLabel.Text = 'Edit Field';

            % Create EditField
            app.EditField = uieditfield(app.UIFigure, 'text');
            app.EditField.Position = [229 244 255 70];

            % Create ENTERURNAMEButton
            app.ENTERURNAMEButton = uibutton(app.UIFigure, 'push');
            app.ENTERURNAMEButton.ButtonPushedFcn = createCallbackFcn(app,
@ENTERURNAMEButtonPushed, true);
            app.ENTERURNAMEButton.Position = [160 415 335 33];
            app.ENTERURNAMEButton.Text = 'ENTER U R NAME';

            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = Helloworld

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```

SAMPLE PROJECT 2

TO DEVELOP AN WEB APP FOR NEWTON SECOND LAW

Components Used:

Button

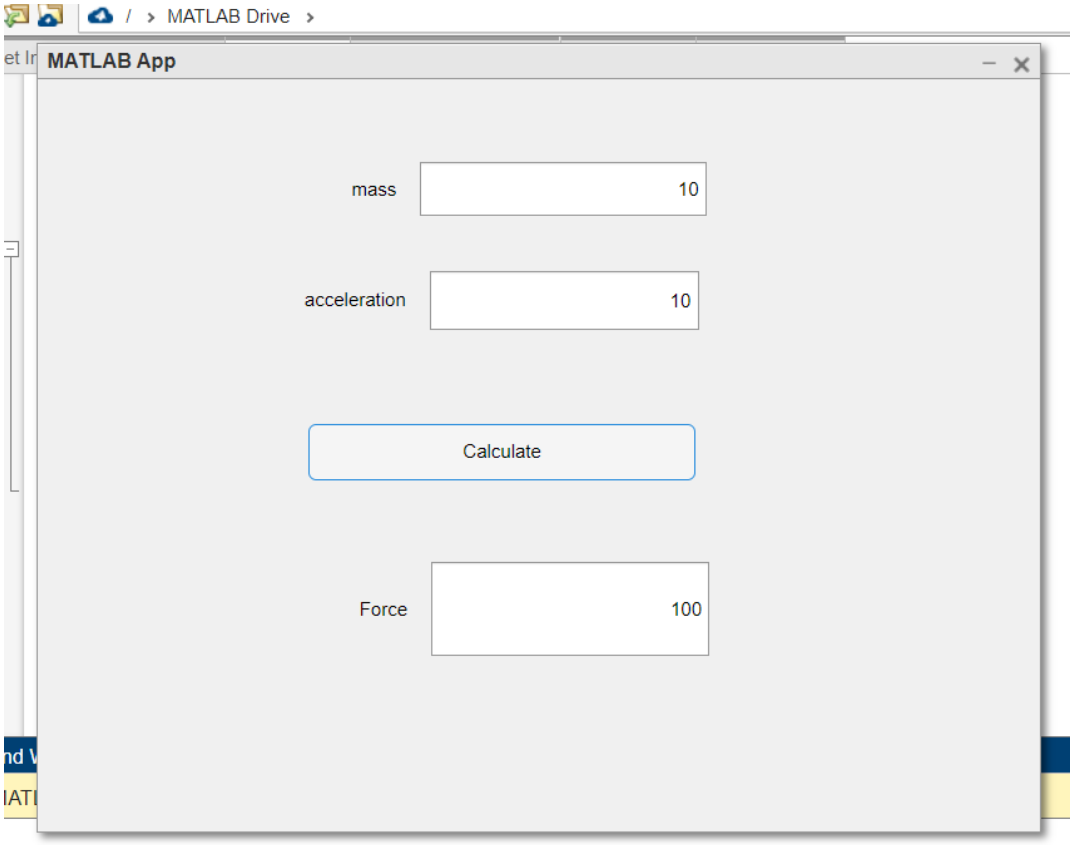
Numeric Data

Code:

```
classdef Nwetonlaw < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                    matlab.ui.Figure
        ForceEditField              matlab.ui.control.NumericEditField
        ForceEditFieldLabel         matlab.ui.control.Label
        accelerationEditField       matlab.ui.control.NumericEditField
        accelerationEditFieldLabel  matlab.ui.control.Label
        massEditField               matlab.ui.control.NumericEditField
        massEditFieldLabel          matlab.ui.control.Label
        CalculateButton             matlab.ui.control.Button
    end

    % Callbacks that handle component events
    methods (Access = private)

        % Button pushed function: CalculateButton
        function CalculateButtonPushed(app, event)
          mass = app.massEditField.Value;
          acceleration=app.accelerationEditField.Value;
          Force=mass*acceleration;
          app.ForceEditField.Value=Force;
        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.Position = [100 100 640 480];
            app.UIFigure.Name = 'MATLAB App';

            % Create CalculateButton
            app.CalculateButton = uibutton(app.UIFigure, 'push');
            app.CalculateButton.ButtonPushedFcn = createCallbackFcn(app,
@CalculateButtonPushed, true);
            app.CalculateButton.Position = [174 225 247 36];
            app.CalculateButton.Text = 'Calculate';

            % Create massEditFieldLabel
            app.massEditFieldLabel = uilabel(app.UIFigure);
            app.massEditFieldLabel.HorizontalAlignment = 'right';
            app.massEditFieldLabel.Position = [196 400 34 22];
```

```
            app.massEditFieldLabel.Text = 'mass';

            % Create massEditField
            app.massEditField = uieditfield(app.UIFigure, 'numeric');
            app.massEditField.Position = [245 394 183 34];

            % Create accelerationEditFieldLabel
            app.accelerationEditFieldLabel = uilabel(app.UIFigure);
            app.accelerationEditFieldLabel.HorizontalAlignment = 'right';
            app.accelerationEditFieldLabel.Position = [166 329 70 22];
            app.accelerationEditFieldLabel.Text = 'acceleration';

            % Create accelerationEditField
            app.accelerationEditField = uieditfield(app.UIFigure, 'numeric');
            app.accelerationEditField.Position = [251 321 172 38];

            % Create ForceEditFieldLabel
            app.ForceEditFieldLabel = uilabel(app.UIFigure);
            app.ForceEditFieldLabel.HorizontalAlignment = 'right';
            app.ForceEditFieldLabel.Position = [201 132 36 22];
            app.ForceEditFieldLabel.Text = 'Force';

            % Create ForceEditField
            app.ForceEditField = uieditfield(app.UIFigure, 'numeric');
            app.ForceEditField.Position = [252 113 178 60];

            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = Nwetonlaw

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```

RESULT:


MATLAB Drive
MATLAB App
mass
10
acceleration
10
Calculate
Force
100

SAMPLE PROJECT 3

TO DEVELOP AN WEB APP FOR USING RADIO BUTTONS

Components Used:

Radio Button

Numeric Data

Code:

```
classdef xxxxx < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure       matlab.ui.Figure
        resetButton    matlab.ui.control.Button
        ConvertButton  matlab.ui.control.Button
        selectButton   matlab.ui.control.Button
        UIAxes2        matlab.ui.control.UIAxes
        UIAxes         matlab.ui.control.UIAxes
    end


    properties (Access = private)
         % Description
        img % Description

        img2 % Description
    end


    % Callbacks that handle component events
    methods (Access = private)

        % Button pushed function: selectButton
        function selectButtonPushed(app, event)
            [file,path]=uigetfile('*,*');
            if isequal(file,0)
             figure(app.UIFigure)
                return
            end
            figure(app.UIFigure)
            app.img=imread(fullfile(path,file));
            title(app.UIAxes,"INPUT")

            imshow(app.img,'parent',app.UIAxes);

        end

        % Button pushed function: ConvertButton
        function ConvertButtonPushed(app, event)
            app.img2=rgb2gray(app.img);
            imshow(app.img2,'parent',app.UIAxes2)
            title(app.UIAxes2,"OUTPUT")
        end

        % Button pushed function: resetButton
        function resetButtonPushed(app, event)
```

```
            app.UIAxes.cla;
            app.UIAxes2.clr;
            app.img="";
            app.img2="";
            title(app.UIaxes, " ");
            title(app.UIAxes2, " ");
        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.Position = [100 100 640 480];
            app.UIFigure.Name = 'MATLAB App';

            % Create UIAxes
            app.UIAxes = uiaxes(app.UIFigure);
            title(app.UIAxes, 'Title')
            xlabel(app.UIAxes, 'X')
            ylabel(app.UIAxes, 'Y')
            zlabel(app.UIAxes, 'Z')
            app.UIAxes.Toolbar.Visible = 'off';
            app.UIAxes.Visible = 'off';
            app.UIAxes.Position = [19 120 276 211];

            % Create UIAxes2
            app.UIAxes2 = uiaxes(app.UIFigure);
            title(app.UIAxes2, 'Title')
            xlabel(app.UIAxes2, 'X')
            ylabel(app.UIAxes2, 'Y')
            zlabel(app.UIAxes2, 'Z')
            app.UIAxes2.Toolbar.Visible = 'off';
            app.UIAxes2.Visible = 'off';
            app.UIAxes2.Position = [331 119 288 211];

            % Create selectButton
            app.selectButton = uibutton(app.UIFigure, 'push');
            app.selectButton.ButtonPushedFcn = createCallbackFcn(app, @selectButtonPushed,
true);
            app.selectButton.Position = [31 400 148 42];
            app.selectButton.Text = 'select';

            % Create ConvertButton
            app.ConvertButton = uibutton(app.UIFigure, 'push');
            app.ConvertButton.ButtonPushedFcn = createCallbackFcn(app,
@ConvertButtonPushed, true);
            app.ConvertButton.Position = [226 400 209 40];
            app.ConvertButton.Text = 'Convert';

            % Create resetButton
            app.resetButton = uibutton(app.UIFigure, 'push');
            app.resetButton.ButtonPushedFcn = createCallbackFcn(app, @resetButtonPushed,
true);
            app.resetButton.Position = [461 398 158 40];
            app.resetButton.Text = 'reset';

            % Show the figure after all components are created
```

```
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = xxxx3

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```

SAMPLE PROJECT 4

TO DEVELOP AN WEB APP FOR PRINTING TABLE

Components Used:

Button

Excel Sheet

Numeric Data

Code:

```
classdef table < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure  matlab.ui.Figure
        Button    matlab.ui.control.Button
        UITable   matlab.ui.control.Table
    end

    % Callbacks that handle component events
    methods (Access = private)

        % Cell edit callback: UITable
        function UITableCellEdit(app, event)
            indices = event.Indices;
            newData = event.NewData;

        end

        % Button pushed function: Button
        function ButtonPushed(app, event)
            t=readtable("Table","Sheet",1)
            app.UITable.data=t
        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.Position = [100 100 640 480];
            app.UIFigure.Name = 'MATLAB App';

            % Create UITable
            app.UITable = uitable(app.UIFigure);
            app.UITable.ColumnName = {'Column 1'; 'Column 2'; 'Column 3'};
            app.UITable.RowName = {};
            app.UITable.CellEditCallback = createCallbackFcn(app, @UITableCellEdit, true);
            app.UITable.Position = [75 68 492 397];

            % Create Button
            app.Button = uibutton(app.UIFigure, 'push');
            app.Button.ButtonPushedFcn = createCallbackFcn(app, @ButtonPushed, true);
            app.Button.Position = [170 18 329 29];
```

```
            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = table

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```

PROJECT 4

TO DEVELOP AN WEB APP FOR PRINTING TABLE

Components Used:

Radio Button

Numeric Data

Code:

```
classdef table < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure  matlab.ui.Figure
        Button    matlab.ui.control.Button
        UITable   matlab.ui.control.Table
    end

    % Callbacks that handle component events
    methods (Access = private)

        % Cell edit callback: UITable
        function UITableCellEdit(app, event)
            indices = event.Indices;
            newData = event.NewData;

        end

        % Button pushed function: Button
        function ButtonPushed(app, event)
            t=readtable("Table","Sheet",1)
            app.UITable.data=t
        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.Position = [100 100 640 480];
            app.UIFigure.Name = 'MATLAB App';

            % Create UITable
            app.UITable = uitable(app.UIFigure);
            app.UITable.ColumnName = {'Column 1'; 'Column 2'; 'Column 3'};
            app.UITable.RowName = {};
            app.UITable.CellEditCallback = createCallbackFcn(app, @UITableCellEdit, true);
            app.UITable.Position = [75 68 492 397];

            % Create Button
            app.Button = uibutton(app.UIFigure, 'push');
            app.Button.ButtonPushedFcn = createCallbackFcn(app, @ButtonPushed, true);
            app.Button.Position = [170 18 329 29];
```

```
            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = table

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```