```
classdef ReceiverPanel < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                        matlab.ui.Figure
        DECODEButton                    matlab.ui.control.Button
        DecodedSignalTextArea           matlab.ui.control.TextArea
        DecodedSignalTextAreaLabel      matlab.ui.control.Label
        DecodingAlgorithmSwitch         matlab.ui.control.Switch
        DecodingAlgorithmSwitchLabel    matlab.ui.control.Label
        SamplingRateEditField           matlab.ui.control.NumericEditField
        SamplingRateEditFieldLabel      matlab.ui.control.Label
        WindowTypeDropDown              matlab.ui.control.DropDown
        WindowTypeDropDownLabel         matlab.ui.control.Label
        WindowShiftEditField            matlab.ui.control.NumericEditField
        WindowShiftEditFieldLabel       matlab.ui.control.Label
        WindowLengthEditField           matlab.ui.control.NumericEditField
        WindowLengthEditFieldLabel      matlab.ui.control.Label
        SpectogramParametersLabel       matlab.ui.control.Label
        PlotSpectogramButton            matlab.ui.control.Button
        PlotinTimeButton                matlab.ui.control.Button
        TrsEditFieldLabel               matlab.ui.control.Label
        TrsEditField                    matlab.ui.control.NumericEditField
        TdsEditField                    matlab.ui.control.NumericEditField
        TdsEditFieldLabel               matlab.ui.control.Label
        StopButton                      matlab.ui.control.Button
        StartButton                     matlab.ui.control.Button
        UIAxesSpectogram                matlab.ui.control.UIAxes
        UIAxesTime                      matlab.ui.control.UIAxes
    end


    properties (Access = public)
         % Description
        DTMF_signal % Description
       % Description
    end


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

        % Callback function
        function RecordAudioButtonPushed(app, event)
            samplingRate=8000;
            recordingDuration=app.RecordingTimeEditField.Value;
            captureAudio(samplingRate,recordingDuration);
        end

        % Callback function
        function PlayButtonPushed(app, event)
            signal = app.DTMF_signal;

            % Specify the sampling rate (replace this with your actual sampling rate)


            % Play the signal as a sound
            playAudio(signal, app.DTMF_signal.SampleRate);
        end

        % Button pushed function: StartButton
        function StartButtonPushed(app, event)
            Fs = 44100;
            pause(0.5);
            recObj = audiorecorder(Fs, 16, 1); % 16 bits, 1 channel

            record(recObj);
            disp('started recording')
            % Store the audiorecorder object in the app data
            app.DTMF_signal = recObj;

        end

        % Button pushed function: StopButton
        function StopButtonPushed(app, event)
            % Check if the recorder object exists
            if isfield(app, 'DTMF_signal') && isvalid(app.DTMF_signal)
                stop(app.DTMF_signal);

                % Get the recorded audio data


                disp('Recording stopped.');
            else

                disp('No recording in progress.');
            end

           audioData = getaudiodata(app.DTMF_signal)



            % sampling rate when an audio is recorded is taken as human
            % signal as default, so the Fs should be 44100
           audiowrite('LastSaved.wav', audioData, app.DTMF_signal.SampleRate);

        end

        % Button pushed function: PlotinTimeButton
        function PlotinTimeButtonPushed(app, event)
            axes = app.UIAxesTime;
```

```
        dtmf_signal = getaudiodata(app.DTMF_signal);


        %signal_duration = (tone_duration+tone_pause)*length(digits) - tone_pause;
        time = (0:(length(dtmf_signal)-1)) / app.DTMF_signal.SampleRate;

        plot(axes,time,dtmf_signal);
        xlabel(axes,'Time(s)','Fontsize',12);
        ylabel(axes,'Amplitude','Fontsize',12);
        title(axes,'Time Plot','Fontsize',14);
    end

    % Value changed function: DecodingAlgorithmSwitch
    function DecodingAlgorithmSwitchValueChanged(app, event)


    end

    % Button pushed function: DECODEButton
    function DECODEButtonPushed(app, event)
        switchValue = app.DecodingAlgorithmSwitch.Value;
        %dtmf_signal = getaudiodata(app.DTMF_signal);
        [audioData,samplingRate] = audioread('LastSaved.wav');

        %disp(Fs);
        tone_duration = app.TdsEditField.Value;
        tone_pause = app.TrsEditField.Value;
        % Decode the audio signal based on the switch value
        if switchValue == "Spectrogram"
            decodedSignal = dtmf_decoder_spectogram_GUI(audioData,samplingRate,tone_duration,tone_pause);
        else
            decodedSignal = dtmf_goertzel_decoder_GUI(audioData,samplingRate,tone_duration,tone_pause);
        end
        %app.DTMF_signal=decodedSignal;
        % Display the decoded signal in the text area
        app.DecodedSignalTextArea.Value = char(decodedSignal);


    end

    % Button pushed function: PlotSpectogramButton
    function PlotSpectogramButtonPushed(app, event)

        %audioData = getaudiodata(app.DTMF_signal);
        [audioData,samplingRate]=audioread('LastSaved.wav');

        %samplingRate = app.SamplingRateEditField;

        selectedWindow = app.WindowTypeDropDown.Value;

        switch selectedWindow
                case 'Rectangular'
                    WindowType = 'rectwin';
                case 'Tukey'
                    WindowType = 'tukeywin';
                case 'Hamming'
                    WindowType = 'hamming';
                otherwise
                    WindowType = 'rectwin';
        end

        spectogram_plotter_DTMF_GUI(app.UIAxesSpectogram,audioData,app.WindowLengthEditField.Value,app.WindowShiftEditField.Value,WindowTy
    end

    % Value changed function: WindowLengthEditField
    function WindowLengthEditFieldValueChanged(app, event)

    end

    % Value changed function: WindowShiftEditField
    function WindowShiftEditFieldValueChanged(app, event)


    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 UIAxesTime
        app.UIAxesTime = uiaxes(app.UIFigure);
        title(app.UIAxesTime, 'Title')
        xlabel(app.UIAxesTime, 'X')
        ylabel(app.UIAxesTime, 'Y')
        zlabel(app.UIAxesTime, 'Z')
        app.UIAxesTime.Position = [394 262 208 167];

        % Create UIAxesSpectogram
        app.UIAxesSpectogram = uiaxes(app.UIFigure);
        title(app.UIAxesSpectogram, 'Title')
        xlabel(app.UIAxesSpectogram, 'X')
        ylabel(app.UIAxesSpectogram, 'Y')
        zlabel(app.UIAxesSpectogram, 'Z')
        app.UIAxesSpectogram.Position = [349 47 232 160];
```

```
% Create StartButton
app.StartButton = uibutton(app.UIFigure, 'push');
app.StartButton.ButtonPushedFcn = createCallbackFcn(app, @StartButtonPushed, true);
app.StartButton.Position = [74 417 100 23];
app.StartButton.Text = 'Start';

% Create StopButton
app.StopButton = uibutton(app.UIFigure, 'push');
app.StopButton.ButtonPushedFcn = createCallbackFcn(app, @StopButtonPushed, true);
app.StopButton.Position = [74 383 100 23];
app.StopButton.Text = 'Stop';

% Create TdsEditFieldLabel
app.TdsEditFieldLabel = uilabel(app.UIFigure);
app.TdsEditFieldLabel.HorizontalAlignment = 'right';
app.TdsEditFieldLabel.Position = [84 334 36 22];
app.TdsEditFieldLabel.Text = 'Td (s)';

% Create TdsEditField
app.TdsEditField = uieditfield(app.UIFigure, 'numeric');
app.TdsEditField.Position = [135 334 29 21];

% Create TrsEditField
app.TrsEditField = uieditfield(app.UIFigure, 'numeric');
app.TrsEditField.Position = [135 302 29 21];

% Create TrsEditFieldLabel
app.TrsEditFieldLabel = uilabel(app.UIFigure);
app.TrsEditFieldLabel.HorizontalAlignment = 'right';
app.TrsEditFieldLabel.Position = [87 302 33 22];
app.TrsEditFieldLabel.Text = 'Tr (s)';

% Create PlotinTimeButton
app.PlotinTimeButton = uibutton(app.UIFigure, 'push');
app.PlotinTimeButton.ButtonPushedFcn = createCallbackFcn(app, @PlotinTimeButtonPushed, true);
app.PlotinTimeButton.Position = [277 396 73 22];
app.PlotinTimeButton.Text = 'Plot in Time';

% Create PlotSpectogramButton
app.PlotSpectogramButton = uibutton(app.UIFigure, 'push');
app.PlotSpectogramButton.ButtonPushedFcn = createCallbackFcn(app, @PlotSpectogramButtonPushed, true);
app.PlotSpectogramButton.Position = [114 31 104 22];
app.PlotSpectogramButton.Text = 'Plot Spectogram';

% Create SpectogramParametersLabel
app.SpectogramParametersLabel = uilabel(app.UIFigure);
app.SpectogramParametersLabel.FontWeight = 'bold';
app.SpectogramParametersLabel.Position = [116 196 144 22];
app.SpectogramParametersLabel.Text = 'Spectogram Parameters';

% Create WindowLengthEditFieldLabel
app.WindowLengthEditFieldLabel = uilabel(app.UIFigure);
app.WindowLengthEditFieldLabel.HorizontalAlignment = 'right';
app.WindowLengthEditFieldLabel.Position = [76 135 88 22];
app.WindowLengthEditFieldLabel.Text = 'Window Length';

% Create WindowLengthEditField
app.WindowLengthEditField = uieditfield(app.UIFigure, 'numeric');
app.WindowLengthEditField.ValueChangedFcn = createCallbackFcn(app, @WindowLengthEditFieldValueChanged, true);
app.WindowLengthEditField.Position = [178 135 106 22];
app.WindowLengthEditField.Value = 512;

% Create WindowShiftEditFieldLabel
app.WindowShiftEditFieldLabel = uilabel(app.UIFigure);
app.WindowShiftEditFieldLabel.HorizontalAlignment = 'right';
app.WindowShiftEditFieldLabel.Position = [77 100 76 22];
app.WindowShiftEditFieldLabel.Text = 'Window Shift';

% Create WindowShiftEditField
app.WindowShiftEditField = uieditfield(app.UIFigure, 'numeric');
app.WindowShiftEditField.ValueChangedFcn = createCallbackFcn(app, @WindowShiftEditFieldValueChanged, true);
app.WindowShiftEditField.Position = [178 100 106 22];
app.WindowShiftEditField.Value = 256;

% Create WindowTypeDropDownLabel
app.WindowTypeDropDownLabel = uilabel(app.UIFigure);
app.WindowTypeDropDownLabel.HorizontalAlignment = 'right';
app.WindowTypeDropDownLabel.Position = [78 65 77 22];
app.WindowTypeDropDownLabel.Text = 'Window Type';

% Create WindowTypeDropDown
app.WindowTypeDropDown = uidropdown(app.UIFigure);
app.WindowTypeDropDown.Items = {'Rectangular', 'Tukey', 'Hamming'};
app.WindowTypeDropDown.Position = [181 65 100 22];
app.WindowTypeDropDown.Value = 'Rectangular';

% Create SamplingRateEditFieldLabel
app.SamplingRateEditFieldLabel = uilabel(app.UIFigure);
app.SamplingRateEditFieldLabel.HorizontalAlignment = 'right';
app.SamplingRateEditFieldLabel.Position = [80 169 84 22];
app.SamplingRateEditFieldLabel.Text = 'Sampling Rate';

% Create SamplingRateEditField
app.SamplingRateEditField = uieditfield(app.UIFigure, 'numeric');
app.SamplingRateEditField.Position = [178 169 106 22];
app.SamplingRateEditField.Value = 8000;

% Create DecodingAlgorithmSwitchLabel
app.DecodingAlgorithmSwitchLabel = uilabel(app.UIFigure);
app.DecodingAlgorithmSwitchLabel.HorizontalAlignment = 'center';
app.DecodingAlgorithmSwitchLabel.Position = [244 302 109 22];
app.DecodingAlgorithmSwitchLabel.Text = 'Decoding Algorithm';

% Create DecodingAlgorithmSwitch
```

```
            app.DecodingAlgorithmSwitch = uiswitch(app.UIFigure, 'slider');
            app.DecodingAlgorithmSwitch.Items = {'Spectrogram', 'Goertzel'};
            app.DecodingAlgorithmSwitch.ValueChangedFcn = createCallbackFcn(app, @DecodingAlgorithmSwitchValueChanged, true);
            app.DecodingAlgorithmSwitch.Position = [277 339 44 19];
            app.DecodingAlgorithmSwitch.Value = 'Spectrogram';

            % Create DecodedSignalTextAreaLabel
            app.DecodedSignalTextAreaLabel = uilabel(app.UIFigure);
            app.DecodedSignalTextAreaLabel.HorizontalAlignment = 'right';
            app.DecodedSignalTextAreaLabel.Position = [198 249 90 22];
            app.DecodedSignalTextAreaLabel.Text = 'Decoded Signal';

            % Create DecodedSignalTextArea
            app.DecodedSignalTextArea = uitextarea(app.UIFigure);
            app.DecodedSignalTextArea.Position = [303 249 97 21];

            % Create DECODEButton
            app.DECODEButton = uibutton(app.UIFigure, 'push');
            app.DECODEButton.ButtonPushedFcn = createCallbackFcn(app, @DECODEButtonPushed, true);
            app.DECODEButton.Position = [64 247 100 23];
            app.DECODEButton.Text = 'DECODE';

            % 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 = ReceiverPanel

            % 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
```