```
classdef HW4_Code_exported < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                matlab.ui.Figure
        TextArea                matlab.ui.control.TextArea
        ClearTableButton        matlab.ui.control.Button
        Graph                   matlab.ui.control.UIAxes
        PlotButton              matlab.ui.control.Button
        CitiesListBoxLabel      matlab.ui.control.Label
        CitiesListBox           matlab.ui.control.ListBox
        StartYearSpinnerLabel   matlab.ui.control.Label
        StartYearSpinner        matlab.ui.control.Spinner
        EndYearSpinnerLabel     matlab.ui.control.Label
        EndYearSpinner          matlab.ui.control.Spinner
        CollectDataButton       matlab.ui.control.Button
        WelcomeText             matlab.ui.control.TextArea
        TimeRangeInfo           matlab.ui.control.TextArea
        Wait                    matlab.ui.control.TextArea
    end


    properties (Access = private)
        structproj % Description
    end
    methods (Access = private)

        function cities = CitiesValueChanged(app)  %This function grabs the values of ↙
the CitiesListBox, which gives the cities chosen by the user.
            cities = app.CitiesListBox.Value;
        end
        function Range = Rangefunct(app)  %This function takes the values of the two↙
Year spinners and creates a range for it.
            Start = app.StartYearSpinner.Value;
            End = app.EndYearSpinner.Value;
            Range = (Start : End);
        end
    end



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

        % Button pushed function: ClearTableButton
        function ClearTableButtonPushed(app, event)
            cla(app.Graph,'reset');
            app.Graph.Visible = 'off';
            app.CitiesListBox.Visible = 'on';
            app.StartYearSpinner.Visible = 'on';
            app.EndYearSpinner.Visible = 'on';
            app.TextArea.Visible = 'on';
            app.TimeRangeInfo.Visible = 'on';
            app.PlotButton.Visible = 'on';
            app.ClearTableButton.Visible = 'off';
        end

        % Button pushed function: PlotButton
        function PlotButtonPushed(app, event)
            app.Wait.Visible = 'on';
            app.TimeRangeInfo.Visible = 'off';
            app.CitiesListBox.Visible = 'off';
            app.StartYearSpinner.Visible = 'off';
            app.EndYearSpinner.Visible = 'off';
            app.TextArea.Visible = 'off';
            app.PlotButton.Visible = 'off';
            pause(1);
            clear selectedData illDeaths totDeaths Ratio statelessCity
            fid = fopen('project.txt','w');
            num = 1; %Initializes variable used for indexing into the struct that ↙
isolates data
            minRat = '';
            selectedYear = Rangefunct(app); %Customizable: years that are selected by↙
user interface
            selectedCity = CitiesValueChanged(app); %Customizable: cities that are↙
selected by user interface
            for i = length(selectedCity)
                selectedCity{i} = convertCharsToStrings(selectedCity{i});
            end
            statelessCity(length(selectedCity)) = selectedCity{end}; %Preallocates a↙
vector for cities with states
            for i = 1:length(selectedCity) %Removes the state designations
                statelessCity(i) = strtok(selectedCity{i},',');
            end
            selectedData(length(app.structproj)) = app.structproj(1); %Preallocates a↙
struct with isolated data


            for z = 1:length(statelessCity) %Selects data points that match with each↙
city selected
                for j = 1:length(selectedYear) %Selects data points that match with↙
each year selected
                    for i = 1:length(app.structproj)
                        if (app.structproj(i).Year == selectedYear(j) && app. ↙
structproj(i).City == statelessCity(z))
                            selectedData(num) = app.structproj(i);
                            num = num+1; %Variable for next index in selectedData↙
struct
```

```
                        end

                   end
                end

                %Preallocates variables that will add a running sum of deaths for ↙
total and for illnesses
                totDeaths = 1:length(selectedYear);
                illDeaths = 1:length(selectedYear);


                for i = 1:length(selectedData) %Adds all of the total deaths in one↙
year
                    for j = 1:length(selectedYear)
                        if selectedData(i).Year == selectedYear(j)
                            if isnan(selectedData(i).AllDeaths)
                                selectedData(i).AllDeaths = 0;
                            end
                            totDeaths(j) = totDeaths(j) + selectedData(i).AllDeaths;
                        end
                    end
                end

               for i = 1:length(selectedData) %Adds all of the illness deaths in one↙
year
                   for j = 1:length(selectedYear)
                        if selectedData(i).Year == selectedYear(j)
                            if isnan(selectedData(i).PneumoniaAndInfluenzaDeaths)
                               selectedData(i).PneumoniaAndInfluenzaDeaths = 0;
                            end
                            illDeaths(j) = illDeaths(j) + selectedData(i). ↙
PneumoniaAndInfluenzaDeaths;
                       end
                   end
               end

               Ratio = 1:length(selectedYear); %Preallocates a variable for the ratio↙
of illness deaths to total deaths
               totRat = 0;
               for i = 1:length(selectedYear) %Creates a vector with the ratios of↙
illness deaths to total deaths for each year
                   Ratio(i) = illDeaths(i)/totDeaths(i);
                   totRat = Ratio(i);
               end
               avgRat = totRat/length(selectedYear);
               if isempty(minRat)
                    minRat = avgRat;
                    minCity = selectedCity(z);
               elseif avgRat < minRat
```

```
                    minRat = avgRat;
                    minCity = selectedCity(z);
               end
               fprintf(fid,"The average ratio for %s is %.2f.\n",selectedCity{z},↙
avgRat);
               hold(app.Graph, "on"); %Holds graph so data of cities can be compared
               plot(app.Graph, selectedYear,Ratio); %Plots the selected years against↙
the ratio of deaths

               if length(selectedYear) < 10
                 xticks(app.Graph, selectedYear); %Sets x-axis to display years as↙
integers (if there's greater than 10 elements, it does this automatically)
               end
               clear selectedData illDeaths totDeaths Ratio selectedCityNoState %↙
Frees these variables to be used again
            end
            fprintf(fid,"\nThe minimum average ratio is %.2f at %s",minRat,minCity↙
{1});
            fclose('all');
            legend(app.Graph,selectedCity) %Makes legend for each city
            app.Graph.Visible = 'on';
            app.Wait.Visible = 'off';
            app.ClearTableButton.Visible = 'on';
        end

        % Button pushed function: CollectDataButton
        function CollectDataButtonPushed(app, event)
            app.CollectDataButton.Visible = 'off'; %once collected, button is no↙
longer necessary
            app.WelcomeText.Visible = 'off';
            app.Wait.Visible = 'on';
            pause(1);
            structp = webread("https://data.cdc.gov/api/views/mr8w-325u/rows.csv?↙
accessType=DOWNLOAD"); %collects data
            app.structproj = table2struct(structp);
            app.Wait.Visible = 'off';
            app.CitiesListBox.Visible = 'on';
            app.StartYearSpinner.Visible = 'on';
            app.EndYearSpinner.Visible = 'on';
            app.TextArea.Visible = 'on';
            app.PlotButton.Visible = 'on';
            app.TimeRangeInfo.Visible = 'on';
        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.Colormap = [0.2431 0.149 0.6588;0.251 0.1647 0.7059;0.2588 ↙
0.1804 0.7529;0.2627 0.1961 0.7961;0.2706 0.2157 0.8353;0.2745 0.2353 0.8706;0.2784 ↙
0.2549 0.898;0.2784 0.2784 0.9216;0.2824 0.302 0.9412;0.2824 0.3216 0.9569;0.2784 ↙
0.3451 0.9725;0.2745 0.3686 0.9843;0.2706 0.3882 0.9922;0.2588 0.4118 0.9961;0.2431 ↙
0.4353 1;0.2196 0.4588 0.9961;0.1961 0.4863 0.9882;0.1843 0.5059 0.9804;0.1804 0.5294 ↙
0.9686;0.1765 0.549 0.9529;0.1686 0.5686 0.9373;0.1529 0.5922 0.9216;0.1451 0.6078 ↙
0.9098;0.1373 0.6275 0.898;0.1255 0.6471 0.8902;0.1098 0.6627 0.8745;0.0941 0.6784 ↙
0.8588;0.0706 0.6941 0.8392;0.0314 0.7098 0.8157;0.0039 0.7216 0.7922;0.0078 0.7294 ↙
0.7647;0.0431 0.7412 0.7412;0.098 0.749 0.7137;0.1412 0.7569 0.6824;0.1725 0.7686 ↙
0.6549;0.1922 0.7765 0.6235;0.2157 0.7843 0.5922;0.2471 0.7922 0.5569;0.2902 0.7961 ↙
0.5176;0.3412 0.8 0.4784;0.3922 0.8039 0.4353;0.4471 0.8039 0.3922;0.5059 0.8 0.349; ↙
0.5608 0.7961 0.3059;0.6157 0.7882 0.2627;0.6706 0.7804 0.2235;0.7255 0.7686 0.1922; ↙
0.7725 0.7608 0.1647;0.8196 0.749 0.1529;0.8627 0.7412 0.1608;0.902 0.7333 0.1765; ↙
0.9412 0.7294 0.2118;0.9725 0.7294 0.2392;0.9961 0.7451 0.2353;0.9961 0.7647 0.2196; ↙
0.9961 0.7882 0.2039;0.9882 0.8118 0.1882;0.9804 0.8392 0.1765;0.9686 0.8627 0.1647; ↙
0.9608 0.8902 0.1529;0.9608 0.9137 0.1412;0.9647 0.9373 0.1255;0.9686 0.9608 0.1059; ↙
0.9765 0.9843 0.0824];
            app.UIFigure.Position = [100 100 656 504];
            app.UIFigure.Name = 'UI Figure';

            % Create TextArea
            app.TextArea = uitextarea(app.UIFigure);
            app.TextArea.Visible = 'off';
            app.TextArea.Position = [260 464 130 27];
            app.TextArea.Value = {'Choose filters for data'};

            % Create ClearTableButton
            app.ClearTableButton = uibutton(app.UIFigure, 'push');
            app.ClearTableButton.ButtonPushedFcn = createCallbackFcn(app, ↙
@ClearTableButtonPushed, true);
            app.ClearTableButton.Visible = 'off';
            app.ClearTableButton.Position = [507 62 100 22];
            app.ClearTableButton.Text = 'Clear Table';

            % Create Graph
            app.Graph = uiaxes(app.UIFigure);
            title(app.Graph, '')
            xlabel(app.Graph, 'Year')
            ylabel(app.Graph, 'Ratio of Deaths')
            app.Graph.PlotBoxAspectRatio = [1.74792243767313 1 1];
            app.Graph.NextPlot = 'replace';
            app.Graph.Visible = 'off';
            app.Graph.Position = [42 99 565 366];

            % Create PlotButton
            app.PlotButton = uibutton(app.UIFigure, 'push');
            app.PlotButton.ButtonPushedFcn = createCallbackFcn(app, ↙
@PlotButtonPushed, true);
            app.PlotButton.Visible = 'off';
            app.PlotButton.Position = [65 62 100 22];
            app.PlotButton.Text = 'Plot';

            % Create CitiesListBoxLabel
            app.CitiesListBoxLabel = uilabel(app.UIFigure);
            app.CitiesListBoxLabel.HorizontalAlignment = 'right';
            app.CitiesListBoxLabel.Visible = 'off';
            app.CitiesListBoxLabel.Position = [67 426 35 22];
            app.CitiesListBoxLabel.Text = 'Cities';

            % Create CitiesListBox
            app.CitiesListBox = uilistbox(app.UIFigure);
            app.CitiesListBox.Items = {'Boston, MA', 'Hartford, CT', 'New Haven, CT',↙
'Providence, RI', 'Albany, NY', 'Buffalo, NY', 'New York, NY', 'Syracuse, NY',↙
'Philadelphia, PA', 'Pittsburgh, PA', 'Trenton, NJ', 'Chicago, IL', 'Cleveland, OH',↙
'Columbia, OH', 'Detroit, MI', 'Indianapolis, IN', 'Des Moines, IA', 'Kansas City,↙
KS', 'Minneapolis, MN', 'Saint Louis, MO', 'Atlanta, GA', 'Baltimore, MD',↙
'Charlotte, NC', 'Richmond, VA', 'Miami, FL', 'Tampa, FL', 'Washington, DC',↙
'Birmingham, AL', 'Lexington, KY', 'Memphis, TN', 'Nashville, TN', 'Austin, TX',↙
'Dallas, TX', 'Houston, TX', 'Little Rock, AR', 'New Orleans, LA', 'Tulsa, OK',↙
'Albuquerque, NM', 'Colorado Springs, CO', 'Denver, CO', 'Las Vegas, NV', 'Phoenix,↙
AZ', 'Tucson, AZ', 'Salt Lake City, UT', 'Honolulu, HI', 'Los Angeles, CA', 'San↙
Diego, CA', 'San Francisco, CA', 'Portland, OR', 'Seattle, WA'};
            app.CitiesListBox.Multiselect = 'on';
            app.CitiesListBox.Visible = 'off';
            app.CitiesListBox.Position = [117 141 174 309];
            app.CitiesListBox.Value = {'Boston, MA'};

            % Create StartYearSpinnerLabel
            app.StartYearSpinnerLabel = uilabel(app.UIFigure);
            app.StartYearSpinnerLabel.HorizontalAlignment = 'right';
            app.StartYearSpinnerLabel.Visible = 'off';
            app.StartYearSpinnerLabel.Position = [346 392 59 22];
            app.StartYearSpinnerLabel.Text = 'Start Year';

            % Create StartYearSpinner
            app.StartYearSpinner = uispinner(app.UIFigure);
            app.StartYearSpinner.Limits = [1962 2015];
            app.StartYearSpinner.Editable = 'off';
            app.StartYearSpinner.Visible = 'off';
            app.StartYearSpinner.Position = [420 392 100 22];
            app.StartYearSpinner.Value = 1962;

            % Create EndYearSpinnerLabel
            app.EndYearSpinnerLabel = uilabel(app.UIFigure);
            app.EndYearSpinnerLabel.HorizontalAlignment = 'right';
            app.EndYearSpinnerLabel.Visible = 'off';
            app.EndYearSpinnerLabel.Position = [349 345 55 22];
            app.EndYearSpinnerLabel.Text = 'End Year';

            % Create EndYearSpinner
            app.EndYearSpinner = uispinner(app.UIFigure);
            app.EndYearSpinner.Limits = [1963 2016];
            app.EndYearSpinner.Editable = 'off';
            app.EndYearSpinner.Visible = 'off';
            app.EndYearSpinner.Position = [419 345 100 22];
            app.EndYearSpinner.Value = 1963;

            % Create CollectDataButton
            app.CollectDataButton = uibutton(app.UIFigure, 'push');
            app.CollectDataButton.ButtonPushedFcn = createCallbackFcn(app, ↙
@CollectDataButtonPushed, true);
            app.CollectDataButton.Position = [279 262 100 22];
            app.CollectDataButton.Text = 'Collect Data';

            % Create WelcomeText
            app.WelcomeText = uitextarea(app.UIFigure);
            app.WelcomeText.Position = [159 319 332 146];
            app.WelcomeText.Value = {'Welcome,'; ''; 'This app takes the ratio of↙
deaths by Pneumonia and/or influenza vs overall deaths of many major US cities. You ↙
can take this data and plot it against time, in any time range you may wish from 1962 ↙
to 2016.'; ''; 'In order to start, please press the button below for the data to be ↙
collected. An internet connection is required.'};

            % Create TimeRangeInfo
            app.TimeRangeInfo = uitextarea(app.UIFigure);
            app.TimeRangeInfo.Visible = 'off';
            app.TimeRangeInfo.Position = [362 243 213 60];
            app.TimeRangeInfo.Value = {'(Some cities do not have data before 1981, if ↙
they are selected, the graph will be empty in that time region for those cities).'};

            % Create Wait
            app.Wait = uitextarea(app.UIFigure);
            app.Wait.HorizontalAlignment = 'center';
            app.Wait.Visible = 'off';
            app.Wait.Position = [249 274 151 29];
            app.Wait.Value = {'Please Wait...'};

            % 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 = HW4_Code_exported

            % 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
```