LAMPIRAN

Pemrograman untuk data latih

```
%%% telur ayam kalkun
% nama folder
nama_folder = 'telur ayam kalkun';
nama_file = dir(fullfile(nama_folder,'*.jpg'));
%membaca jumlah file
jumlah_file = numel(nama_file);

%variabel ciri telur ayam kalkun dan targetnya
ciri_telurayamkalkun = zeros(jumlah_file,4);
target_telurayamkalkun = zeros(jumlah_file,1);

for k = 1:jumlah_file
    % Baca gambar
    Img = imread(fullfile(nama_folder,nama_file(k).name));

    % Konversi ke grayscale
    Img_gray = rgb2gray(Img);

    % Binarisasi dengan metode adaptive untuk hasil yang
lebih baik
    bw = imbinarize(Img_gray, 'adaptive', 'Sensitivity',
0.5);

    % Pastikan ada objek dalam gambar biner
    if sum(bw(:)) > 0
        % Ekstraksi properti region
        s = regionprops(bw, 'Area', 'MajorAxisLength',
'MinorAxisLength', 'Perimeter');

        % Pilih region terbesar (biasanya objek utama)
        if ~isempty(s)
            % Temukan indeks region terbesar
            [~, idx] = max([s.Area]);

            % Simpan properti region terbesar
            ciri_telurayamkalkun(k,1) =
s(idx).MajorAxisLength;
            ciri_telurayamkalkun(k,2) =
s(idx).MinorAxisLength;
            ciri_telurayamkalkun(k,3) = s(idx).Area;
            ciri_telurayamkalkun(k,4) = s(idx).Perimeter;
        else
            % Tangani kasus tidak ada region
            warning('Tidak ada region terdeteksi pada
gambar: %s', nama_file(k).name);
```

```
            ciri_telurayamkalkun(k,:) = NaN;
        end
    else
        % Tangani kasus gambar biner kosong
        warning('Gambar biner kosong: %s',
nama_file(k).name);
        ciri_telurayamkalkun(k,:) = NaN;
    end

    % Set target
    target_telurayamkalkun(k) = 1;
end

%%% telur ayam kampung
% nama folder
nama_folder = 'telur ayam kampung';
nama_file = dir(fullfile(nama_folder,'*.jpg'));
%membaca jumlah file
jumlah_file = numel(nama_file);

%variabel ciri Basmati dan targetnya
ciri_telurayamkampung = zeros(jumlah_file,4);
target_telurayamkampung = zeros(jumlah_file,1);

for k = 1:jumlah_file
    % Baca gambar
    Img = imread(fullfile(nama_folder,nama_file(k).name));

    % Konversi ke grayscale
    Img_gray = rgb2gray(Img);

    % Binarisasi dengan metode adaptive untuk hasil yang
lebih baik
    bw = imbinarize(Img_gray, 'adaptive', 'Sensitivity',
0.5);

    % Pastikan ada objek dalam gambar biner
    if sum(bw(:)) > 0
        % Ekstraksi properti region
        s = regionprops(bw, 'Area', 'MajorAxisLength',
'MinorAxisLength', 'Perimeter');

        % Pilih region terbesar (biasanya objek utama)
        if ~isempty(s)
            % Temukan indeks region terbesar
            [~, idx] = max([s.Area]);

            % Simpan properti region terbesar
```

```
            ciri_telurayamkampung(k,1) =
s(idx).MajorAxisLength;
            ciri_telurayamkampung(k,2) =
s(idx).MinorAxisLength;
            ciri_telurayamkampung(k,3) = s(idx).Area;
            ciri_telurayamkampung(k,4) = s(idx).Perimeter;
        else
            % Tangani kasus tidak ada region
            warning('Tidak ada region terdeteksi pada
gambar: %s', nama_file(k).name);
            ciri_telurayamkampung(k,:) = NaN;
        end
    else
        % Tangani kasus gambar biner kosong
        warning('Gambar biner kosong: %s',
nama_file(k).name);
        ciri_telurayamkampung(k,:) = NaN;
    end

    % Set target
    target_telurayamkampung(k) = 2;
end

%%% telur ayam petelur
% nama folder
nama_folder = 'telur ayam petelur';
nama_file = dir(fullfile(nama_folder,'*.jpg'));
%membaca jumlah file
jumlah_file = numel(nama_file);

%variabel ciri telur ayam petelur dan targetnya
ciri_telurayampetelur = zeros(jumlah_file,4);
target_telurayampetelur = zeros(jumlah_file,1);

 for k = 1:jumlah_file
    % Baca gambar
    Img = imread(fullfile(nama_folder,nama_file(k).name));

    % Konversi ke grayscale
    Img_gray = rgb2gray(Img);

    % Binarisasi dengan metode adaptive untuk hasil yang
lebih baik
    bw = imbinarize(Img_gray, 'adaptive', 'Sensitivity',
0.5);

    % Pastikan ada objek dalam gambar biner
    if sum(bw(:)) > 0
        % Ekstraksi properti region
```

```
        s = regionprops(bw, 'Area', 'MajorAxisLength',
'MinorAxisLength', 'Perimeter');

        % Pilih region terbesar (biasanya objek utama)
        if ~isempty(s)
            % Temukan indeks region terbesar
            [~, idx] = max([s.Area]);

            % Simpan properti region terbesar
            ciri_telurayampetelur(k,1) =
s(idx).MajorAxisLength;
            ciri_telurayampetelur(k,2) =
s(idx).MinorAxisLength;
            ciri_telurayampetelur(k,3) = s(idx).Area;
            ciri_telurayampetelur(k,4) = s(idx).Perimeter;
        else
            % Tangani kasus tidak ada region
            warning('Tidak ada region terdeteksi pada
gambar: %s', nama_file(k).name);
            ciri_telurayampetelur(k,:) = NaN;
        end
    else
        % Tangani kasus gambar biner kosong
        warning('Gambar biner kosong: %s',
nama_file(k).name);
        ciri_telurayampetelur(k,:) = NaN;
    end

    % Set target
    target_telurayampetelur(k) = 3;
end

 %menyusun variabel ciri_latih dan target_latih
 ciri_latih = [ciri_telurayamkalkun;
ciri_telurayamkampung; ciri_telurayampetelur];
 target_latih = [ones(size(ciri_telurayamkalkun,1), 1);
                 2*ones(size(ciri_telurayamkampung,1), 1);
                 3*ones(size(ciri_telurayampetelur,1), 1)];

  options = statset('UseParallel',true);
  t =
templateSVM('Standardize',true,'SaveSupportVectors',true);

  Mdl = fitcecoc(ciri_latih,target_latih); %Model SVM
  CVMdl = crossval(Mdl,'Options',options); %Model Conmat
SVM

  kelas_keluaran = predict(Mdl,ciri_latih);
  oofLabel = kfoldPredict(CVMdl,'Options',options);
```

```
  %Conmat
  ConfMat =
confusionchart(target_latih,oofLabel,'RowSummary','total-
normalized');
  ConfMat.InnerPosition = [0.10 0.12 0.85 0.85];
  ConfMat.RowSummary = 'row-normalized';
  ConfMat.ColumnSummary = 'column-normalized';

  L = size(Mdl.CodingMatrix,5); % Number of SVMs
  sv = cell(L,3); % Preallocate for support vector indices
  for j = 3:L
    SVM = Mdl.BinaryLearners{j};
    sv{j} = SVM.SupportVectors;
    sv{j} = sv{j}.*SVM.Sigma + SVM.Mu;
  end

  figure
  gscatter(ciri_latih(:,1),ciri_latih(:,2),target_latih);
  hold on
  markers = {'ko','ro','bo'}; % Should be of length L
  for j = 3:L
    svs = sv{j};
    plot(svs(:,1),svs(:,2),svs(:,3),markers{j},...
        'MarkerSize',10 + (j - 1)*3);
  end

  title('SVM')
  xlabel('metric')
  ylabel('eccentricity')
  %legend([target_latih,{'Support vectors - SVM 1',...
    %'Support vectors - SVM 2';'Support vectors - SVM
3'}],...
    %'Location';'Best')

legend('telurayamkalkun','telurayamkampung','telurayampete
lur')
  %xlabel('metric')
  %ylabel('eccentricity')
  %title('SVM')
  hold off

  jumlah_benar = 0;
  for k = 1:jumlah_file
      if isequal(kelas_keluaran,target_latih)
          jumlah_benar = jumlah_benar+1;
      end
  end
```

```
  jumlah_benar = sum(kelas_keluaran == target_latih);
akurasi_pelatihan = jumlah_benar / numel(target_latih) *
100;


  save Mdl Mdl
```

Pemrograman untuk data uji

```
%%% telur ayam kalkun
% nama folder
nama_folder = 'Uji Kalkun';
nama_file = dir(fullfile(nama_folder,'*.jpg'));
%membaca jumlah file
jumlah_file = numel(nama_file);

%variabel ciri telur ayam kampung dan targetnya
ciri_uji_kalkun = zeros(jumlah_file,4);
target_uji_kalkun = zeros(jumlah_file,1);

for k = 1:jumlah_file
    % Baca gambar
    Img = imread(fullfile(nama_folder,nama_file(k).name));

    % Konversi ke grayscale
    Img_gray = rgb2gray(Img);

    % Binarisasi dengan metode adaptive untuk hasil yang
lebih baik
    bw = imbinarize(Img_gray, 'adaptive', 'Sensitivity',
0.5);

    % Pastikan ada objek dalam gambar biner
    if sum(bw(:)) > 0
        % Ekstraksi properti region
        s = regionprops(bw, 'Area', 'MajorAxisLength',
'MinorAxisLength', 'Perimeter');

        % Pilih region terbesar (biasanya objek utama)
        if ~isempty(s)
            % Temukan indeks region terbesar
            [~, idx] = max([s.Area]);

            % Simpan properti region terbesar
            ciri_uji_kalkun(k,1) = s(idx).MajorAxisLength;
            ciri_uji_kalkun(k,2) = s(idx).MinorAxisLength;
            ciri_uji_kalkun(k,3) = s(idx).Area;
            ciri_uji_kalkun(k,4) = s(idx).Perimeter;
        else
            % Tangani kasus tidak ada region
```

```
            warning('Tidak ada region terdeteksi pada
gambar: %s', nama_file(k).name);
            ciri_uji_kalkun(k,:) = NaN;
        end
    else
        % Tangani kasus gambar biner kosong
        warning('Gambar biner kosong: %s',
nama_file(k).name);
        ciri_uji_kalkun(k,:) = NaN;
    end

    % Set target
    target_uji_kalkun(k) = 1;
end

%%% telur ayam kampung
% nama folder
nama_folder = 'Uji Kampung';
nama_file = dir(fullfile(nama_folder,'*.jpg'));
%membaca jumlah file
jumlah_file = numel(nama_file);

%variabel ciri telur ayam kampung dan targetnya
ciri_uji_kampung = zeros(jumlah_file,4);
target_uji_kampung = zeros(jumlah_file,1);

for k = 1:jumlah_file
    % Baca gambar
    Img = imread(fullfile(nama_folder,nama_file(k).name));

    % Konversi ke grayscale
    Img_gray = rgb2gray(Img);

    % Binarisasi dengan metode adaptive untuk hasil yang
lebih baik
    bw = imbinarize(Img_gray, 'adaptive', 'Sensitivity',
0.5);

    % Pastikan ada objek dalam gambar biner
    if sum(bw(:)) > 0
        % Ekstraksi properti region
        s = regionprops(bw, 'Area', 'MajorAxisLength',
'MinorAxisLength', 'Perimeter');

        % Pilih region terbesar (biasanya objek utama)
        if ~isempty(s)
            % Temukan indeks region terbesar
            [~, idx] = max([s.Area]);
```

```
            % Simpan properti region terbesar
            ciri_uji_kampung(k,1) =
s(idx).MajorAxisLength;
            ciri_uji_kampung(k,2) =
s(idx).MinorAxisLength;
            ciri_uji_kampung(k,3) = s(idx).Area;
            ciri_uji_kampung(k,4) = s(idx).Perimeter;
        else
            % Tangani kasus tidak ada region
            warning('Tidak ada region terdeteksi pada
gambar: %s', nama_file(k).name);
            ciri_uji_kampung(k,:) = NaN;
        end
    else
        % Tangani kasus gambar biner kosong
        warning('Gambar biner kosong: %s',
nama_file(k).name);
        ciri_uji_kampung(k,:) = NaN;
    end

    % Set target
    target_uji_kampung(k) = 2;
end



%%% telur ayam petelur
% nama folder
nama_folder = 'Uji Petelur';
nama_file = dir(fullfile(nama_folder,'*.jpg'));
%membaca jumlah file
jumlah_file = numel(nama_file);

%variabel ciri telur ayam petelur dan targetnya
ciri_uji_petelur = zeros(jumlah_file,4);
target_uji_petelur = zeros(jumlah_file,1);

for k = 1:jumlah_file
    % Baca gambar
    Img = imread(fullfile(nama_folder,nama_file(k).name));

    % Konversi ke grayscale
    Img_gray = rgb2gray(Img);

    % Binarisasi dengan metode adaptive untuk hasil yang
lebih baik
    bw = imbinarize(Img_gray, 'adaptive', 'Sensitivity',
0.5);
```

```
    % Pastikan ada objek dalam gambar biner
    if sum(bw(:)) > 0
        % Ekstraksi properti region
        s = regionprops(bw, 'Area', 'MajorAxisLength',
'MinorAxisLength', 'Perimeter');

        % Pilih region terbesar (biasanya objek utama)
        if ~isempty(s)
            % Temukan indeks region terbesar
            [~, idx] = max([s.Area]);

            % Simpan properti region terbesar
            ciri_uji_petelur(k,1) =
s(idx).MajorAxisLength;
            ciri_uji_petelur(k,2) =
s(idx).MinorAxisLength;
            ciri_uji_petelur(k,3) = s(idx).Area;
            ciri_uji_petelur(k,4) = s(idx).Perimeter;
        else
            % Tangani kasus tidak ada region
            warning('Tidak ada region terdeteksi pada
gambar: %s', nama_file(k).name);
            ciri_uji_petelur(k,:) = NaN;
        end
    else
        % Tangani kasus gambar biner kosong
        warning('Gambar biner kosong: %s',
nama_file(k).name);
        ciri_uji_petelur(k,:) = NaN;
    end

    % Set target
    target_uji_petelur(k) = 3;
end

%menyusun variabel ciri_latih dan target_latih
 ciri_uji = [ciri_uji_kalkun; ciri_uji_kampung;
ciri_uji_petelur];
 target_uji = [ones(size(target_uji_kalkun,1), 1);
               2*ones(size(target_uji_kampung,1), 1);
               3*ones(size(target_uji_petelur,1), 1)];
%memanggil model SVM
load Mdl

kelas_keluaran = predict(Mdl, ciri_uji);

 L = size(Mdl.CodingMatrix,5); % Number of SVMs
  sv = cell(L,3); % Preallocate for support vector indices
  for j = 3:L
    SVM = Mdl.BinaryLearners{j};
```

```
    sv{j} = SVM.SupportVectors;
    sv{j} = sv{j}.*SVM.Sigma + SVM.Mu;
  end

  figure
  gscatter(ciri_uji(:,1),ciri_uji(:,2),target_uji);
  hold on
  markers = {'ko','ro','bo'}; % Should be of length L
  for j = 3:L
    svs = sv{j};
    plot(svs(:,1),svs(:,2),svs(:,3),markers{j},...
        'MarkerSize',10 + (j - 1)*3);
  end

  title('SVM')
  xlabel('metric')
  ylabel('eccentricity')
  %legend([target_latih,{'Support vectors - SVM 1',...
    %'Support vectors - SVM 2';'Support vectors - SVM
3'}],...
    %'Location';'Best')

legend('telurayamkalkun','telurayamkampung','telurayampete
lur')
  %xlabel('metric')
  %ylabel('eccentricity')
  %title('SVM')
  hold off
```

Pemrograman untuk aplikasi AI_TELUR_AYAM

```
classdef AI_TELUR_AYAM < matlab.apps.AppBase


% Properties that correspond to app components
properties (Access = public)
UIFigure matlab.ui.Figure
UIAxes matlab.ui.control.UIAxes
UIAxes_2 matlab.ui.control.UIAxes
UIAxes_3 matlab.ui.control.UIAxes
SVM_TELUR_AYAMLabel matlab.ui.control.Label
AmbilGambarButton matlab.ui.control.Button
EditField matlab.ui.control.EditField
SegmentasiButton matlab.ui.control.Button
EkstrasiCiriButton matlab.ui.control.Button
KlasifikasiButton matlab.ui.control.Button
EditField_2 matlab.ui.control.EditField
ResetButton matlab.ui.control.Button
UITable matlab.ui.control.Table
```

```
end


properties (Access = public)
Property % Description
end


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


% Button pushed function: AmbilGambarButton
function AmbilGambarButtonPushed(app, event)
% Buka dialog pemilihan file
[nama_file, nama_folder] = uigetfile('*.jpg');
drawnow
figure(app.UIFigure)


if ~isequal(nama_file,0)
Img = imread(fullfile(nama_folder,nama_file));
imshow(Img,'Parent',app.UIAxes)
title(app.UIAxes,'RGB Image')
app.EditField.Value = nama_file;
app.Property.Img = Img;
app.Property.nama_file = nama_file;
else
return
end
end


% Button pushed function: SegmentasiButton
function SegmentasiButtonPushed(app, event)
Img = app.Property.Img;


% Konversi ke grayscale
Img_gray = rgb2gray(Img);
imshow(Img_gray,[],'Parent',app.UIAxes_2)
title(app.UIAxes_2,'Grayscale Image')
% Binarisasi dengan metode adaptive untuk hasil yang lebih
baik
bw = imbinarize(Img_gray, 'adaptive', 'Sensitivity', 0.5);
```

```
imshow(bw,[],'Parent',app.UIAxes_3)
title(app.UIAxes_3,'Biner Image')
app.Property.bw = bw;
end


% Button pushed function: EkstrasiCiriButton
function EkstrasiCiriButtonPushed(app, event)
bw = app.Property.bw;


if sum(bw(:)) > 0
% Ekstraksi properti region
s = regionprops(bw, 'Area', 'MajorAxisLength',
'MinorAxisLength', 'Perimeter');
% Pilih region terbesar (biasanya objek utama)
if ~isempty(s)
% Temukan indeks region terbesar
[~, idx] = max([s.Area]);
% Simpan properti region terbesar
ciri_uji(1,1) = s(idx).MajorAxisLength;
ciri_uji(1,2) = s(idx).MinorAxisLength;
ciri_uji(1,3) = s(idx).Area;
ciri_uji(1,4) = s(idx).Perimeter;




else
% Tangani kasus tidak ada region
warning('Tidak ada region terdeteksi pada gambar: %s',
nama_file(k).name);
ciri_uji_petelur(k,:) = NaN;
end
else
% Tangani kasus gambar biner kosong
warning('Gambar biner kosong: %s', nama_file(k).name);
ciri_uji_petelur(k,:) = NaN;
end
data_tabel = cell(2,2);
data_tabel{1,1} = 'Major Axis';
data_tabel{2,1} = 'Minor Axis';
data_tabel{3,1} = 'Area';
data_tabel{4,1} = 'Perimeter';
data_tabel{1,2} = num2str(s(idx).MajorAxisLength);
```

```
data_tabel{2,2} = num2str(s(idx).MinorAxisLength);
data_tabel{3,2} = num2str(s(idx).Area);
data_tabel{4,2} = num2str(s(idx).Perimeter);
app.UITable.Data = data_tabel;
app.Property.data_uji = ciri_uji;
end


% Button pushed function: KlasifikasiButton
function KlasifikasiButtonPushed(app, event)
ciri_uji = app.Property.data_uji;


%memanggil model SVM
load Mdl
kelas_keluaran = predict(Mdl, ciri_uji);
%kelas_keluaran1 = num2str(kelas_keluaran); % Converts
numeric to string
%kelas keluaran
switch kelas_keluaran
case 1
kelas_keluaran = 'Ayam Kalkun';
case 2
kelas_keluaran = 'Ayam Kampung';
case 3
kelas_keluaran = 'Ayam Petelur';
otherwise
kelas_keluaran = 'Tidak Terklasifikasi';
end
app.EditField_2.Value = kelas_keluaran;
app.Property.kelas_keluaran = kelas_keluaran;
app.Property.Mdl = Mdl;
end


% Button pushed function: ResetButton
function ResetButtonPushed(app, event)
app.EditField.Value = '';
app.EditField_2.Value = '';
cla(app.UIAxes,'reset')
app.UIAxes.XTick = [];
app.UIAxes.YTick = [];app.UIAxes_3
cla(app.UIAxes_2,'reset')
app.UIAxes_2.XTick = [];
app.UIAxes_2.YTick = [];
```

```
cla(app.UIAxes_3,'reset')
app.UIAxes_3.XTick = [];
app.UIAxes_3.YTick = [];
app.UITable.Data = [];
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 = [100 100 735 480];
app.UIFigure.Name = 'MATLAB App';


% Create UIAxes
app.UIAxes = uiaxes(app.UIFigure);
title(app.UIAxes, '')
xlabel(app.UIAxes, '')
ylabel(app.UIAxes, '')
app.UIAxes.PlotBoxAspectRatio = [1.04516129032258 1 1];
app.UIAxes.XTick = [];
app.UIAxes.YTick = [];
app.UIAxes.Position = [153 284 188 197];


% Create UIAxes_2
app.UIAxes_2 = uiaxes(app.UIFigure);
title(app.UIAxes_2, '')
xlabel(app.UIAxes_2, '')
ylabel(app.UIAxes_2, '')
app.UIAxes_2.PlotBoxAspectRatio = [1 1.05521472392638 1];
app.UIAxes_2.XTick = [];
app.UIAxes_2.YTick = [];
app.UIAxes_2.Position = [340 284 188 197];


% Create UIAxes_3
```

```
app.UIAxes_3 = uiaxes(app.UIFigure);
title(app.UIAxes_3, '')
xlabel(app.UIAxes_3, '')
ylabel(app.UIAxes_3, '')
app.UIAxes_3.PlotBoxAspectRatio = [1 1.05521472392638 1];
app.UIAxes_3.XTick = [];
app.UIAxes_3.YTick = [];
app.UIAxes_3.Position = [527 284 188 197];


% Create SVM_TELUR_AYAMLabel
app.SVM_TELUR_AYAMLabel = uilabel(app.UIFigure);
app.SVM_TELUR_AYAMLabel.FontName = 'Times New Roman';
app.SVM_TELUR_AYAMLabel.FontSize = 16;
app.SVM_TELUR_AYAMLabel.Position = [1 459 153 22];
app.SVM_TELUR_AYAMLabel.Text = 'SVM_TELUR_AYAM';


% Create AmbilGambarButton
app.AmbilGambarButton = uibutton(app.UIFigure, 'push');
app.AmbilGambarButton.ButtonPushedFcn =
createCallbackFcn(app, @AmbilGambarButtonPushed, true);
app.AmbilGambarButton.Position = [1 427 144 22];
app.AmbilGambarButton.Text = 'Ambil Gambar';


% Create EditField
app.EditField = uieditfield(app.UIFigure, 'text');
app.EditField.Position = [1 397 144 22];


% Create SegmentasiButton
app.SegmentasiButton = uibutton(app.UIFigure, 'push');
app.SegmentasiButton.ButtonPushedFcn = createCallbackFcn(app,
@SegmentasiButtonPushed, true);
app.SegmentasiButton.Position = [1 366 144 22];
app.SegmentasiButton.Text = 'Segmentasi';


% Create EkstrasiCiriButton
app.EkstrasiCiriButton = uibutton(app.UIFigure, 'push');
            app.EkstrasiCiriButton.ButtonPushedFcn =
createCallbackFcn(app, @EkstrasiCiriButtonPushed, true);
            app.EkstrasiCiriButton.Position = [1 336 144 22];
            app.EkstrasiCiriButton.Text = 'Ekstrasi Ciri';
```

```
            % Create KlasifikasiButton
            app.KlasifikasiButton = uibutton(app.UIFigure,
'push');
            app.KlasifikasiButton.ButtonPushedFcn =
createCallbackFcn(app, @KlasifikasiButtonPushed, true);
            app.KlasifikasiButton.Position = [1 306 144 22];
            app.KlasifikasiButton.Text = 'Klasifikasi';


            % Create EditField_2
            app.EditField_2 = uieditfield(app.UIFigure,
'text');
            app.EditField_2.Position = [1 276 144 22];


            % Create ResetButton
            app.ResetButton = uibutton(app.UIFigure, 'push');
            app.ResetButton.ButtonPushedFcn =
createCallbackFcn(app, @ResetButtonPushed, true);
            app.ResetButton.Position = [1 246 144 22];
            app.ResetButton.Text = 'Reset';


            % Create UITable
            app.UITable = uitable(app.UIFigure);
            app.UITable.ColumnName = {'Ciri'; 'Nilai'};
            app.UITable.RowName = {};
            app.UITable.Position = [173 100 302 185];


            % 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 = AI_TELUR_AYAMm
```

```
            % 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
```