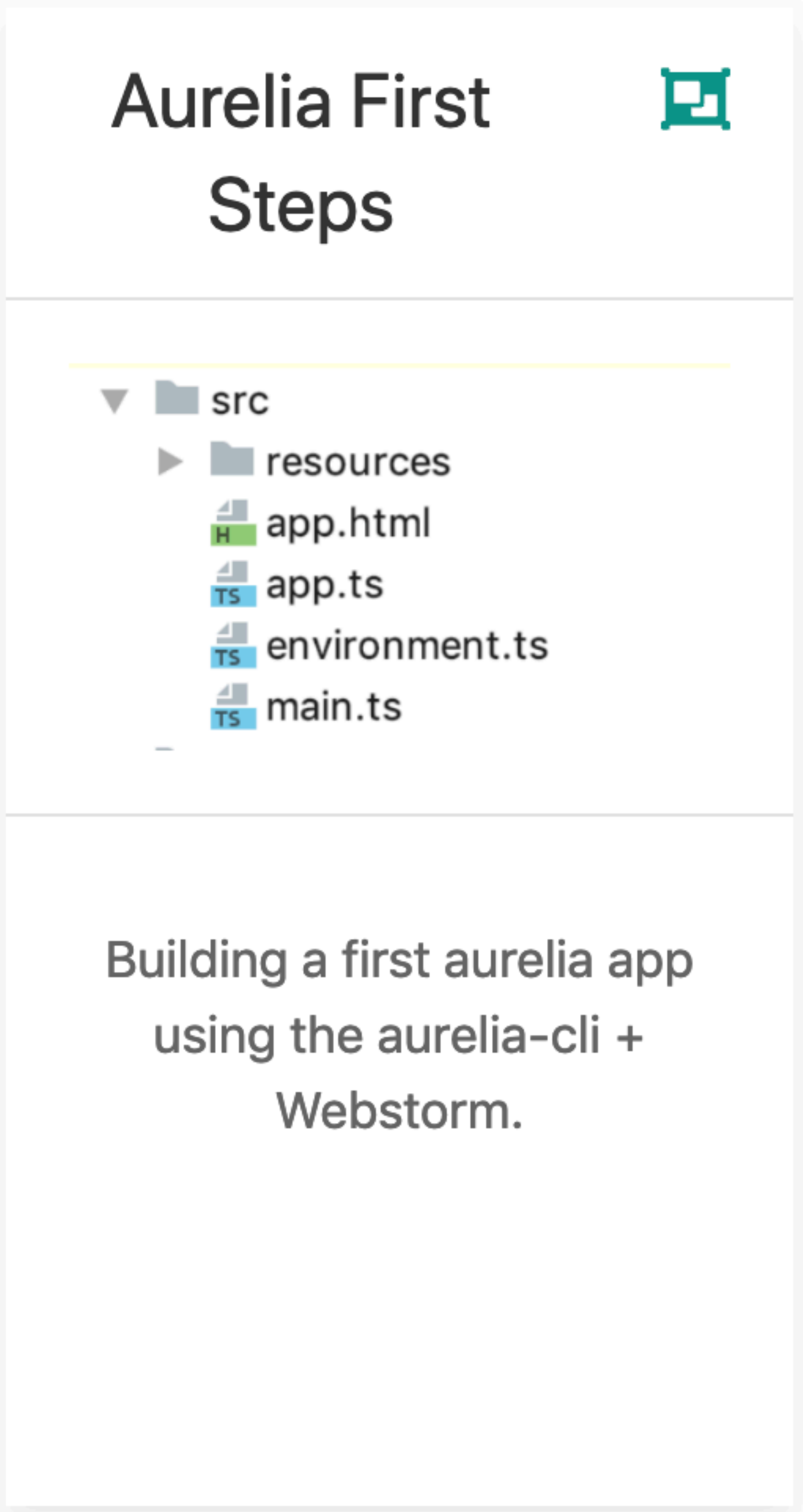
Aurelia First Steps

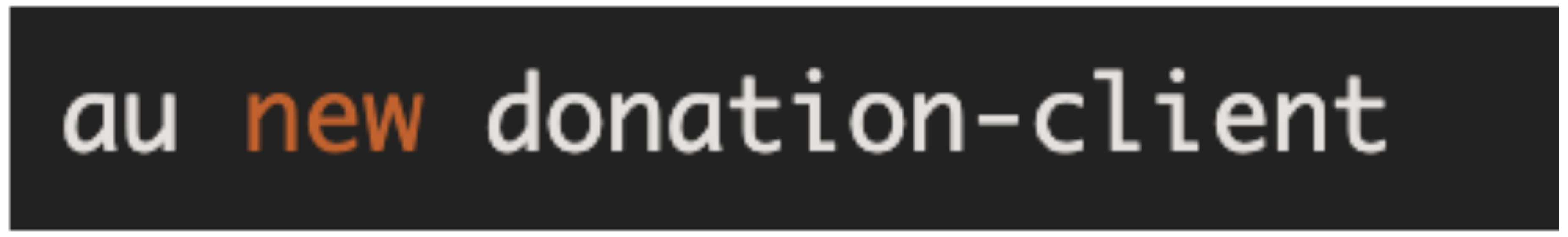

Aurelia First Steps

```
src
  resources
  app.html
  app.ts
  environment.ts
  main.ts
```

Building a first aurelia app using the aurelia-cli + Webstorm.

Creating A New Aurelia Project

- To create a project:



```
au new donation-client
```

- Select ‘TypeScript’ as the default language

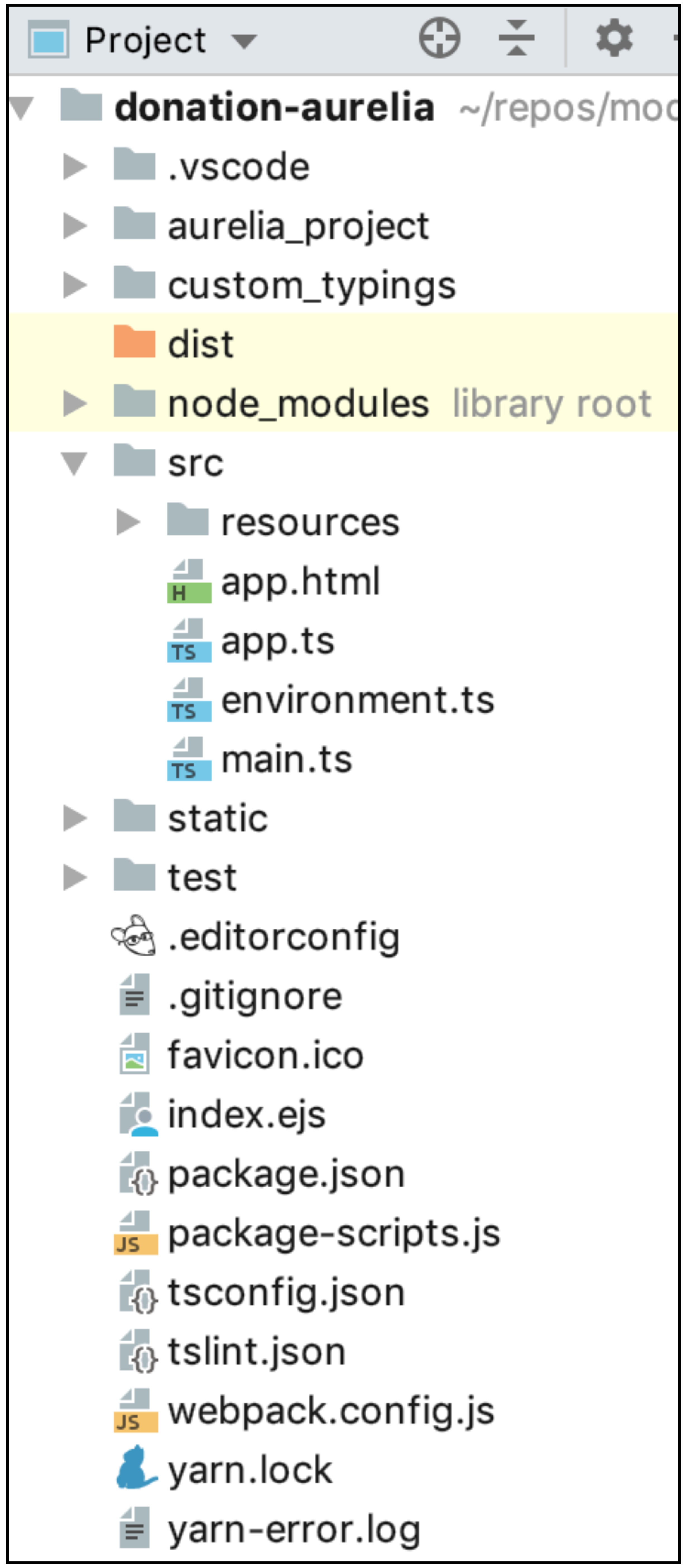
- Select ‘Yarn’ as the default package installer

- Open project in Webstorm

- Launch command

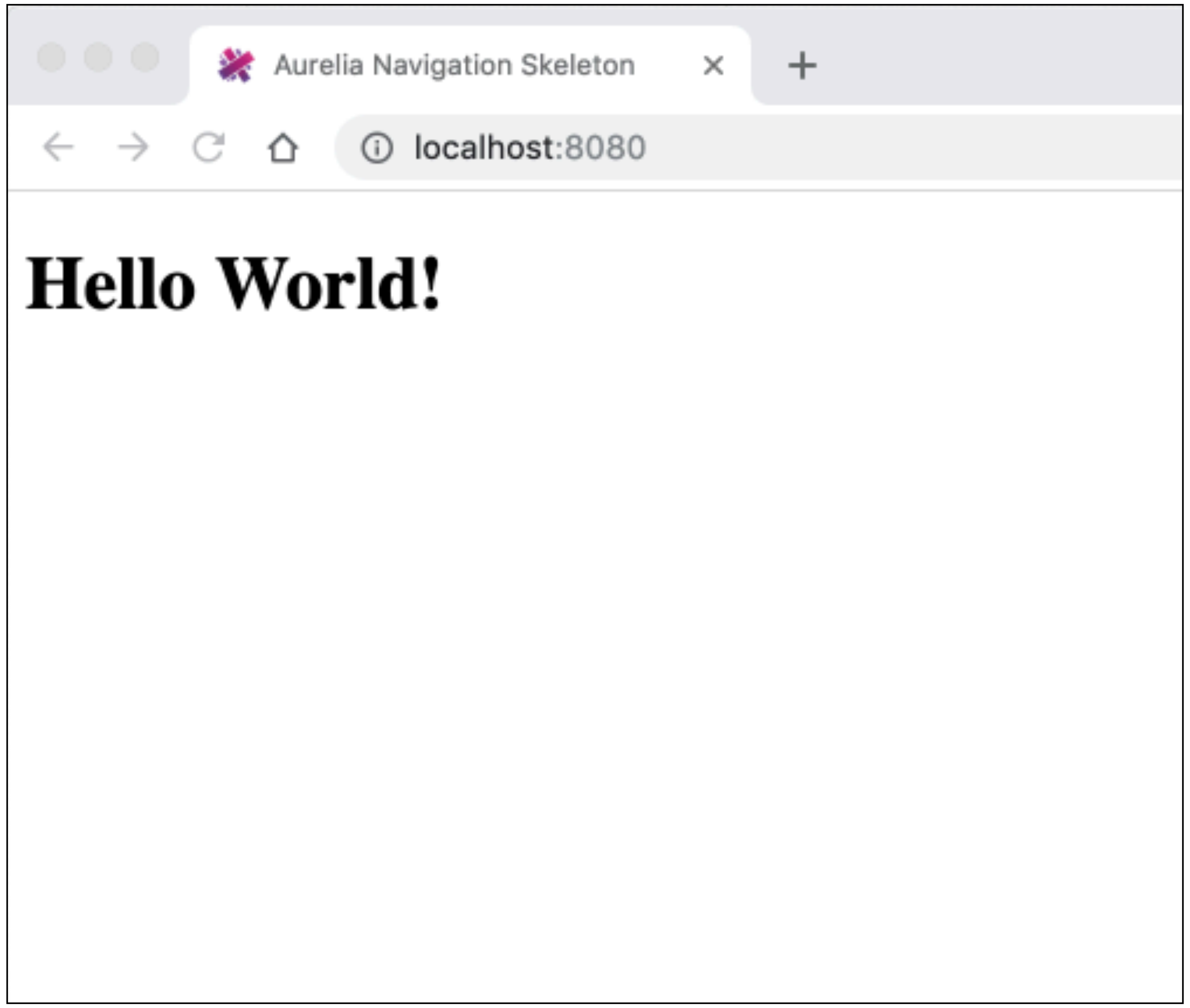
```
au run --watch
```

http://localhost:8080/



Project

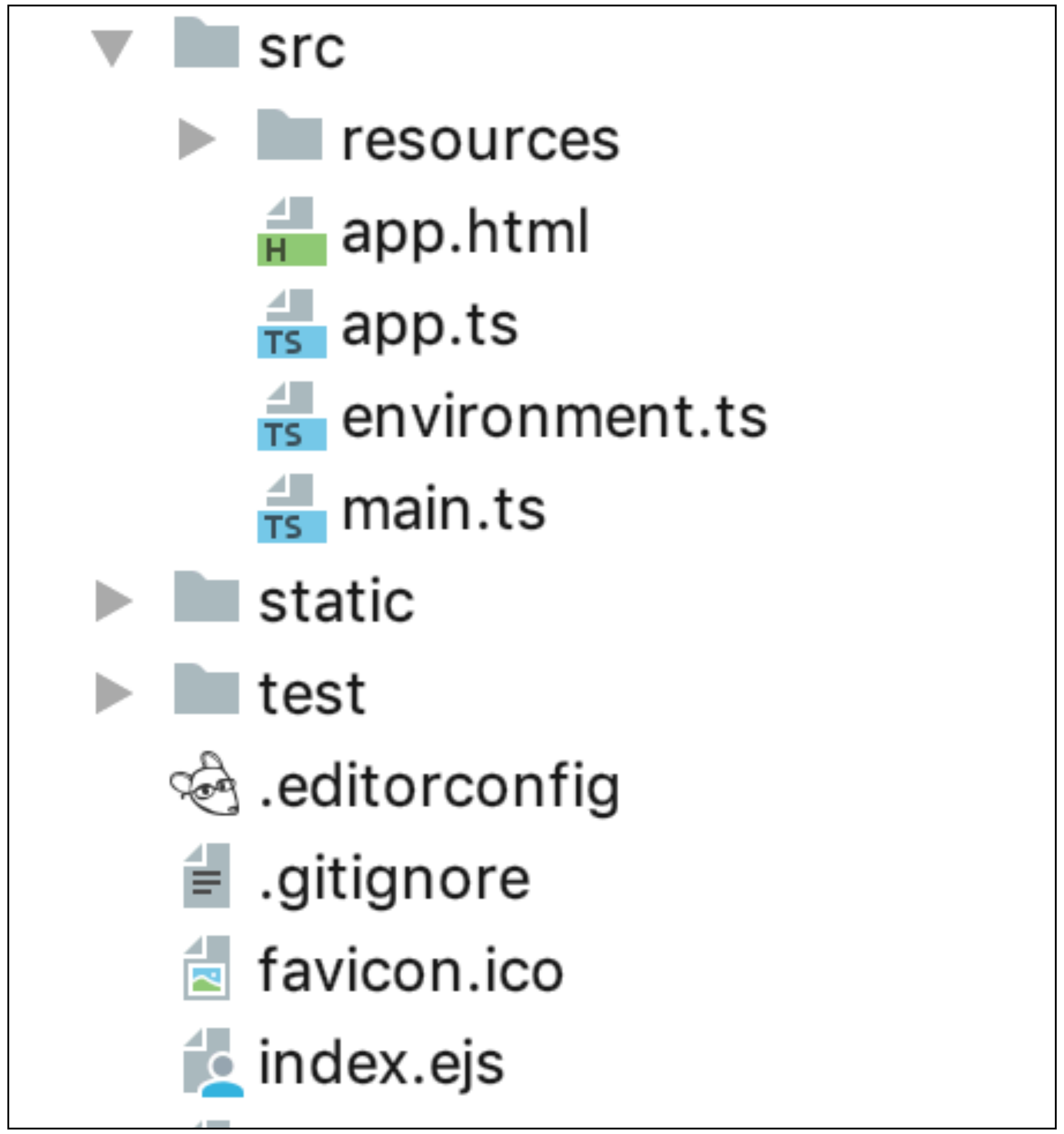
```
donation-aurelia ~/repos/moc
  .vscode
  aurelia_project
  custom_typings
  dist
  node_modules library root
  src
    resources
    app.html
    app.ts
    environment.ts
    main.ts
  static
  test
  .editorconfig
  .gitignore
  favicon.ico
  index.ejs
  package.json
  package-scripts.js
  tsconfig.json
  tslint.json
  webpack.config.js
  yarn.lock
  yarn-error.log
```



Aurelia Navigation Skeleton

localhost:8080

Hello World!

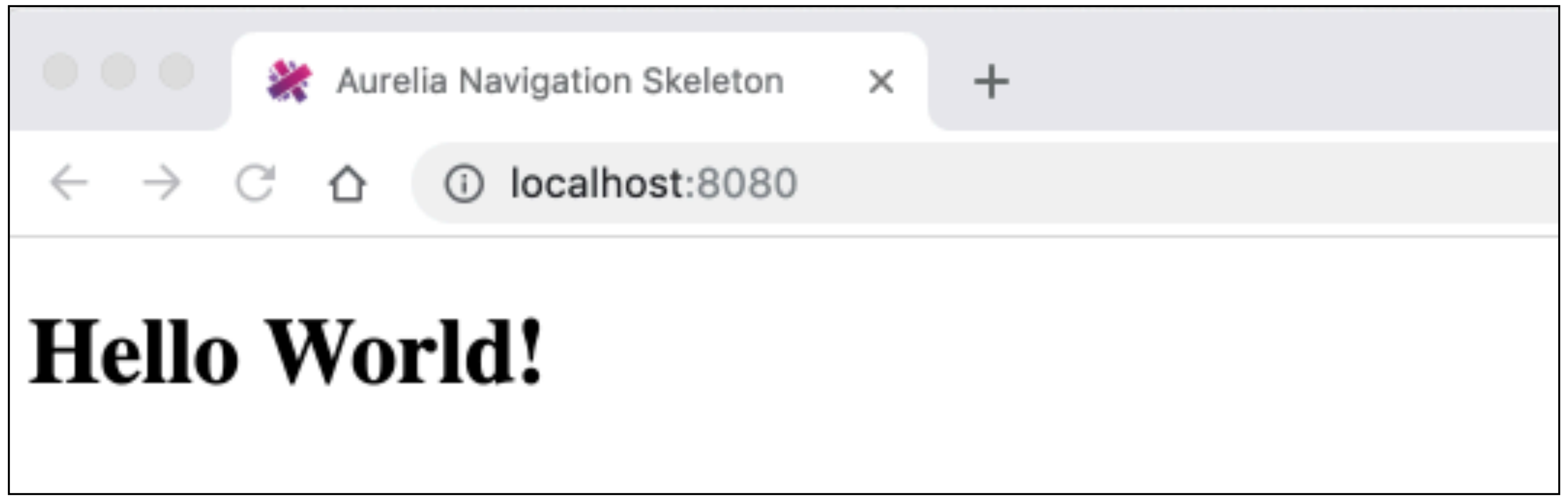

```
▼ src
    ▶ resources
      app.html
      app.ts
      environment.ts
      main.ts
  ▶ static
  ▶ test
    .editorconfig
    .gitignore
    favicon.ico
    index.ejs
```

```html
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title><%- htmlWebpackPlugin.options.metadata.title %></title>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <base href="<%- htmlWebpackPlugin.options.metadata.baseUrl %>">
  </head>
  <body aurelia-app="main">
  </body>
</html>
```

```html
<template>
  <h1>${message}</h1>
</template>
```

```typescript
export class App {
  message = 'Hello World!';
}
```

- Default generated application



Hello World!

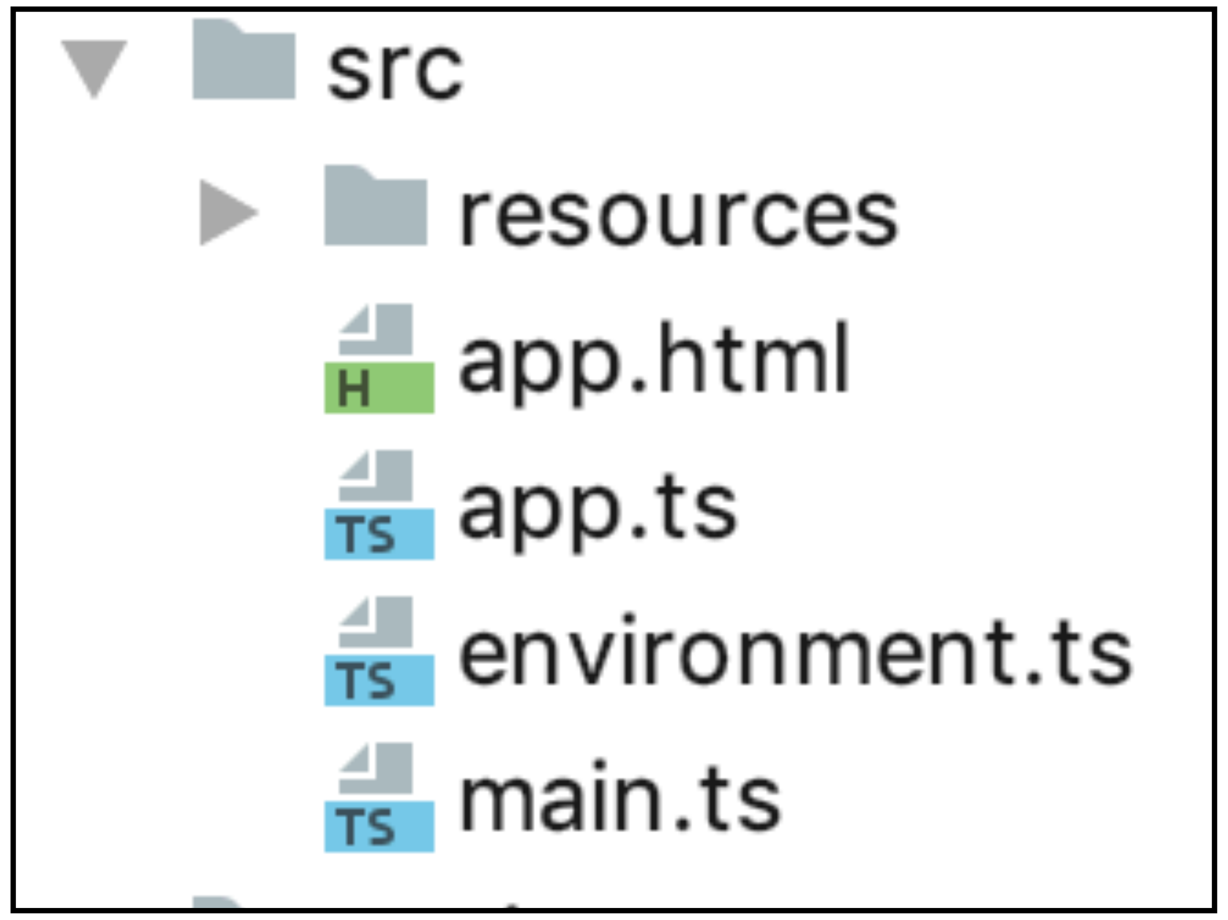
- Include Semantic UI Libraries for all components

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title><%- htmlWebpackPlugin.options.metadata.title %></title>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <base href="<%- htmlWebpackPlugin.options.metadata.baseUrl %>">
    <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.1/jquery.min.js"></script>
    <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/semantic-ui/2.4.1/semantic.min.js"></script>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/semantic-ui/2.4.1/semantic.min.css" type="text/css">
    <!-- imported CSS are concatenated and added automatically -->
  </head>
  <body aurelia-app="main">
  </body>
</html>
```

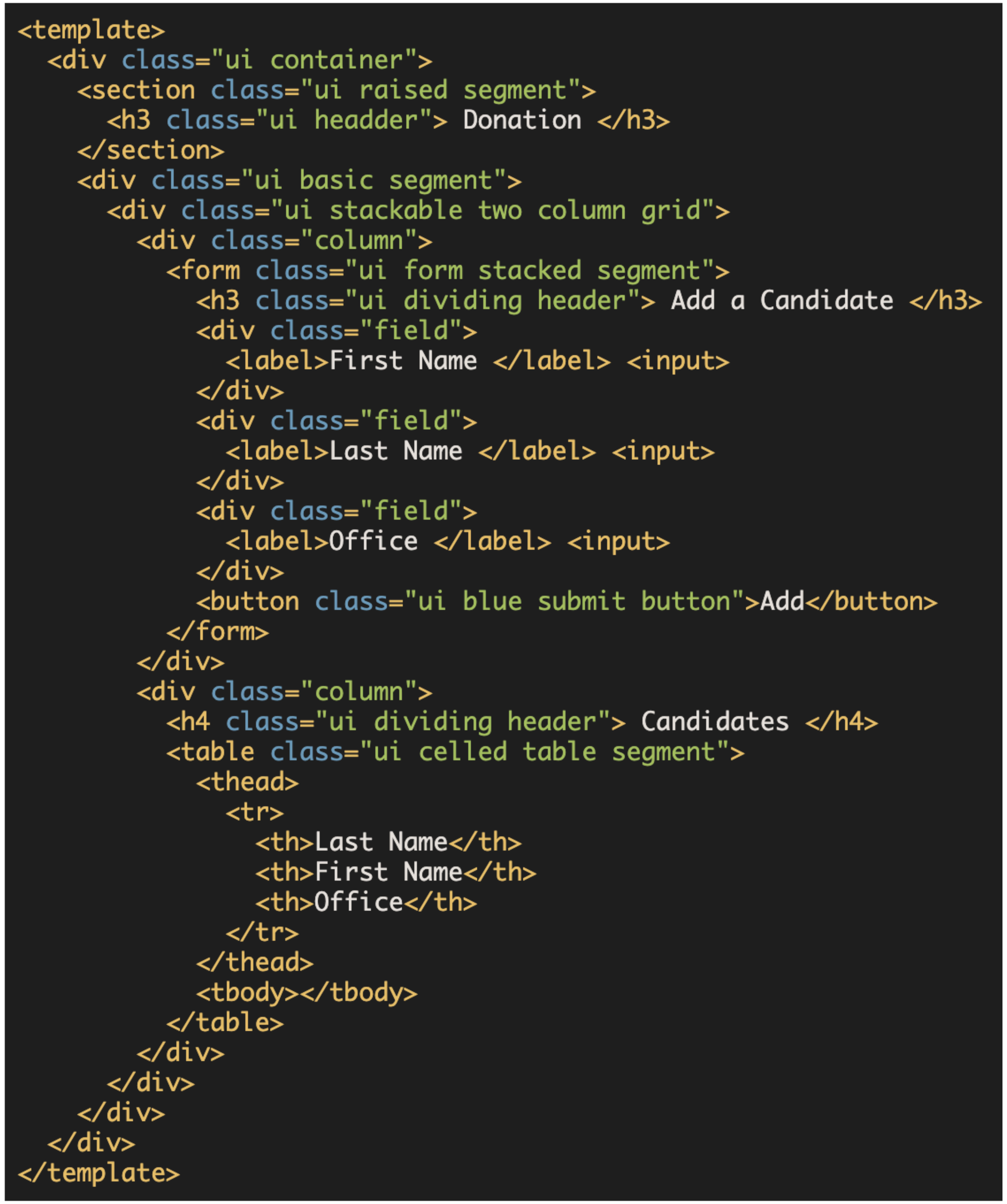
the View

app.html

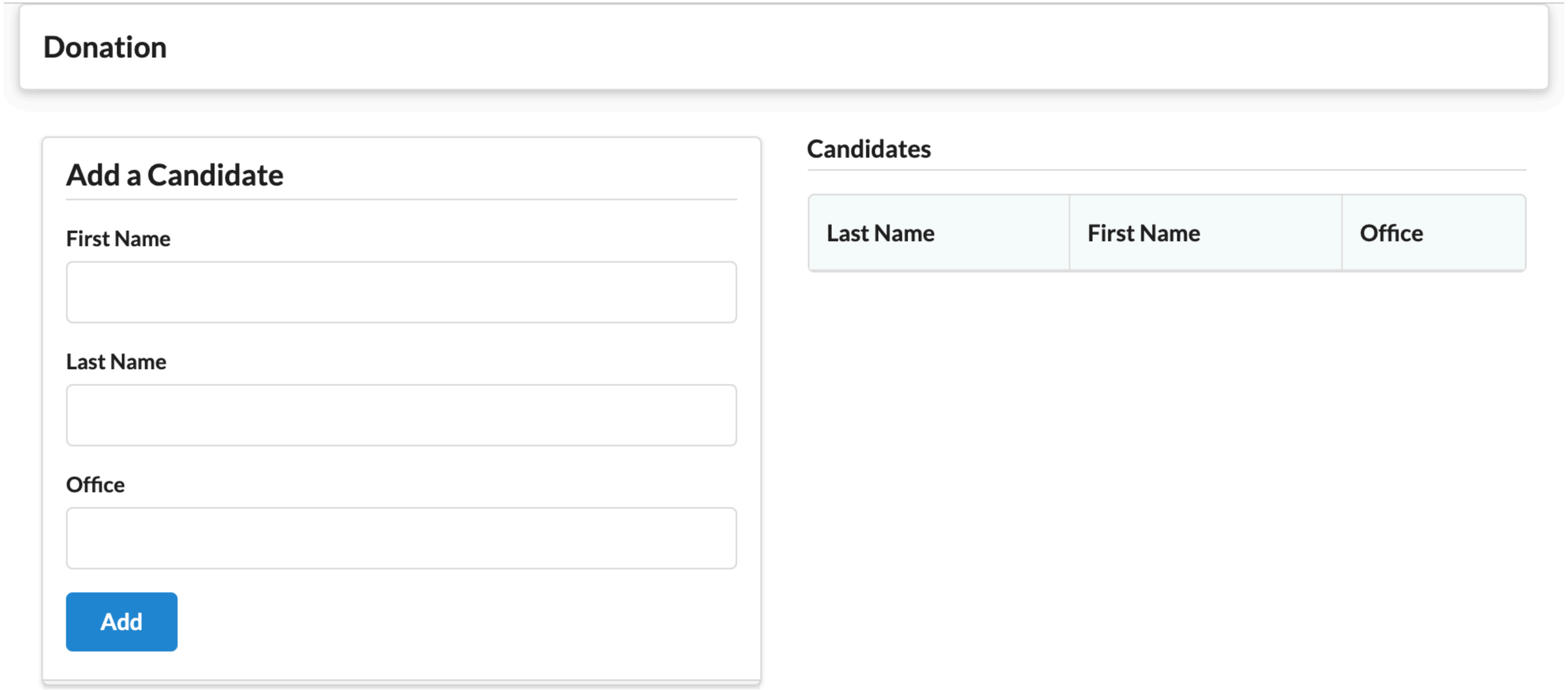
```
<template>
  <h1>${message}</h1>
</template>
```



```
src
  resources
  app.html
  app.ts
  environment.ts
  main.ts
```



```
<template>
  <div class="ui container">
    <section class="ui raised segment">
      <h3 class="ui headder"> Donation </h3>
    </section>
    <div class="ui basic segment">
      <div class="ui stackable two column grid">
        <div class="column">
          <form class="ui form stacked segment">
            <h3 class="ui dividing header"> Add a Candidate </h3>
            <div class="field">
              <label>First Name </label> <input>
            </div>
            <div class="field">
              <label>Last Name </label> <input>
            </div>
            <div class="field">
              <label>Office </label> <input>
            </div>
            <button class="ui blue submit button">Add</button>
          </form>
        </div>
        <div class="column">
          <h4 class="ui dividing header"> Candidates </h4>
          <table class="ui celled table segment">
            <thead>
              <tr>
                <th>Last Name</th>
                <th>First Name</th>
                <th>Office</th>
              </tr>
            </thead>
            <tbody></tbody>
          </table>
        </div>
      </div>
    </div>
  </div>
</template>
```



Donation

Add a Candidate

First Name

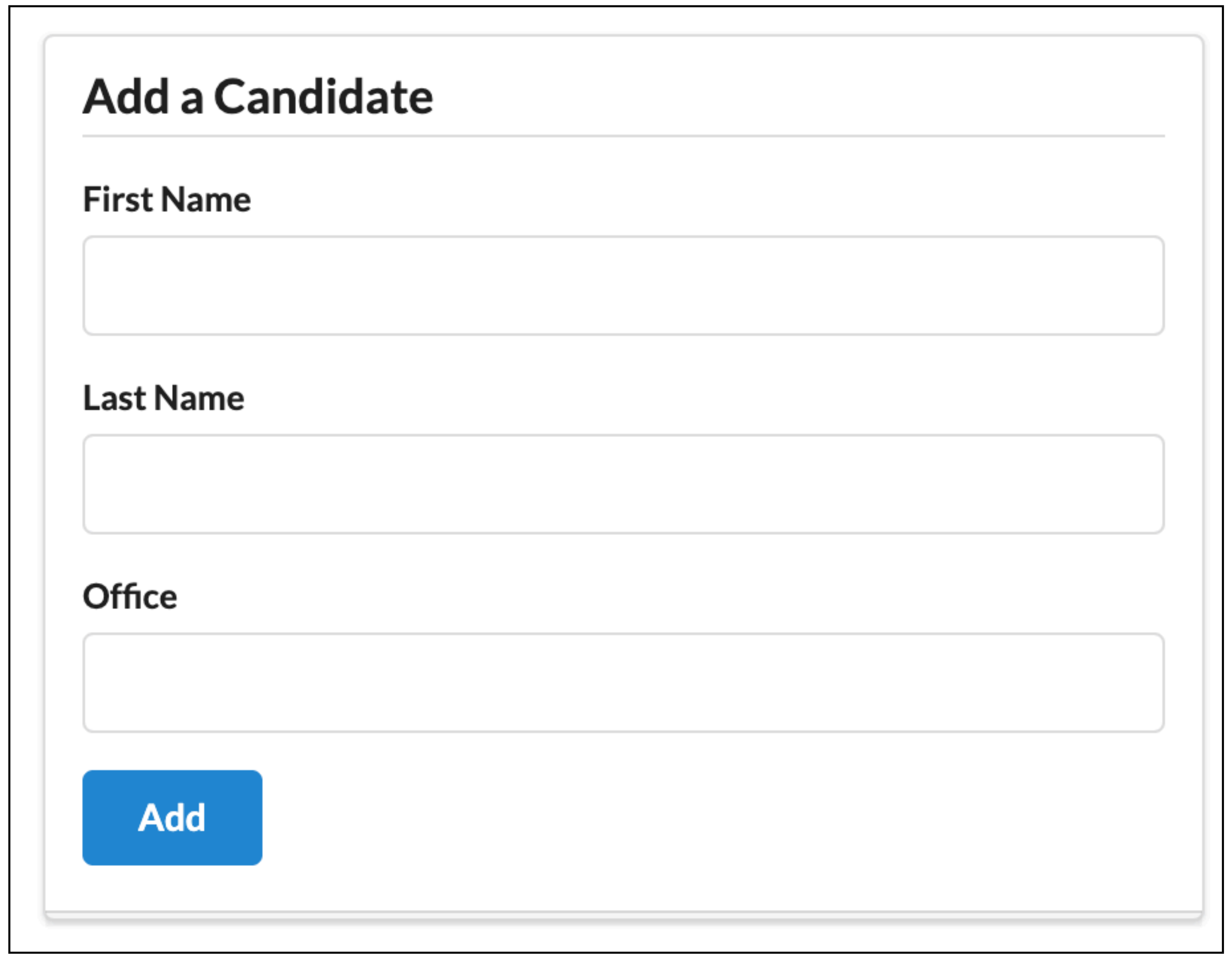
Last Name

Office

Candidates

Last Name	First Name	Office

the View

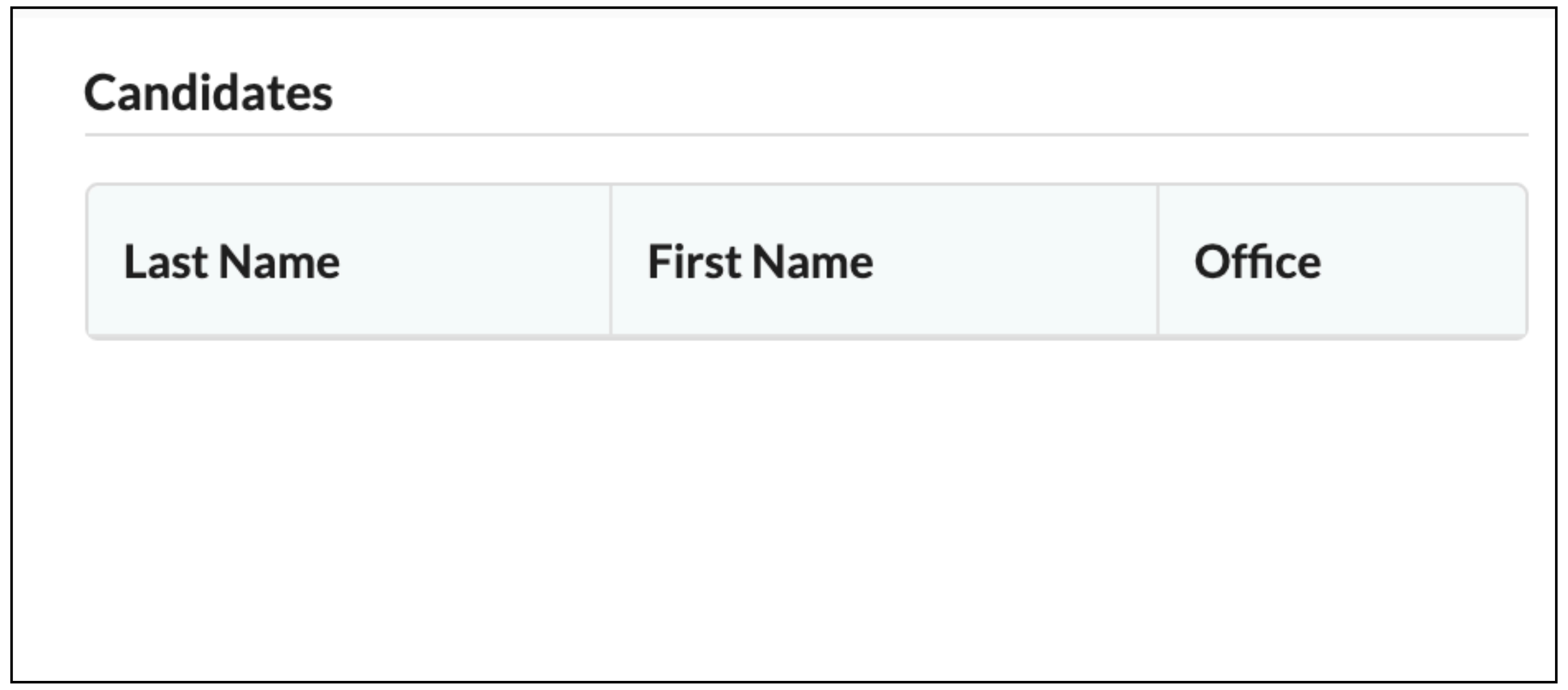


Add a Candidate

First Name

Last Name

Office



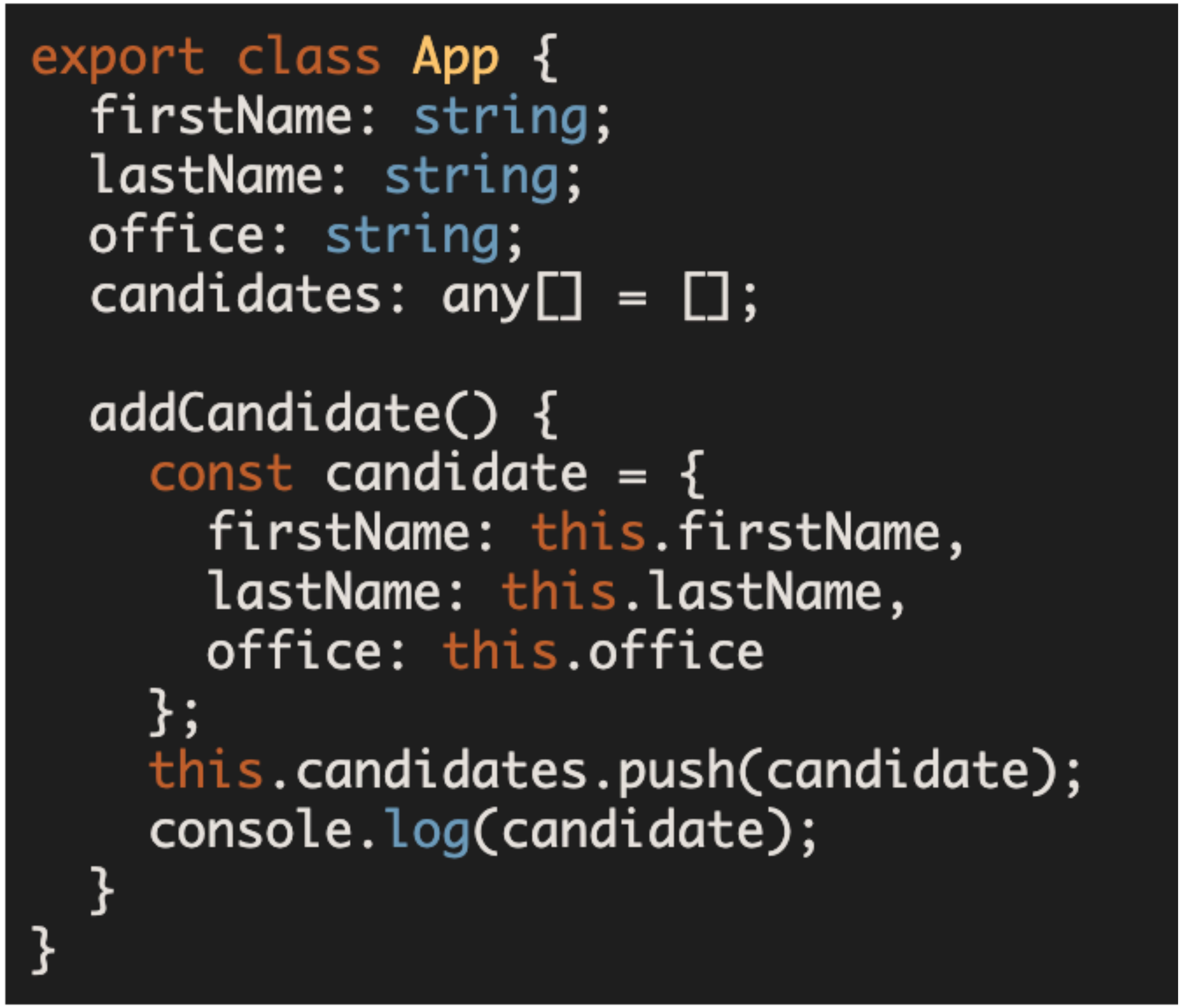
Candidates

Last Name	First Name	Office

src/app.htnl

```
<template>
  <div class="ui container">
    <section class="ui raised segment">
      <h3 class="ui headder"> Donation </h3>
    </section>
    <div class="ui basic segment">
      <div class="ui stackable two column grid">
        <div class="column">
          <form class="ui form stacked segment">
            <h3 class="ui dividing header"> Add a Candidate </h3>
            <div class="field">
              <label>First Name </label> <input>
            </div>
            <div class="field">
              <label>Last Name </label> <input>
            </div>
            <div class="field">
              <label>Office </label> <input>
            </div>
            <button class="ui blue submit button">Add</button>
          </form>
        </div>
        <div class="column">
          <h4 class="ui dividing header"> Candidates </h4>
          <table class="ui celled table segment">
            <thead>
              <tr>
                <th>Last Name</th>
                <th>First Name</th>
                <th>Office</th>
              </tr>
            </thead>
            <tbody></tbody>
          </table>
        </div>
      </div>
    </div>
  </div>
</template>
```

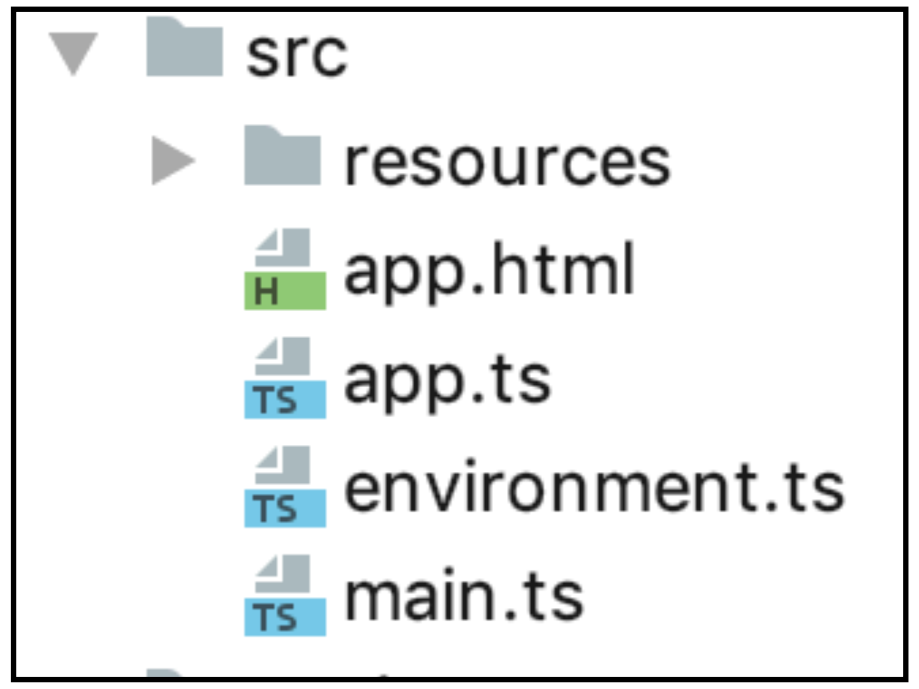
app.ts



```
export class App {
  firstName: string;
  lastName: string;
  office: string;
  candidates: any[] = [];

  addCandidate() {
    const candidate = {
      firstName: this.firstName,
      lastName: this.lastName,
      office: this.office
    };
    this.candidates.push(candidate);
    console.log(candidate);
  }
}
```

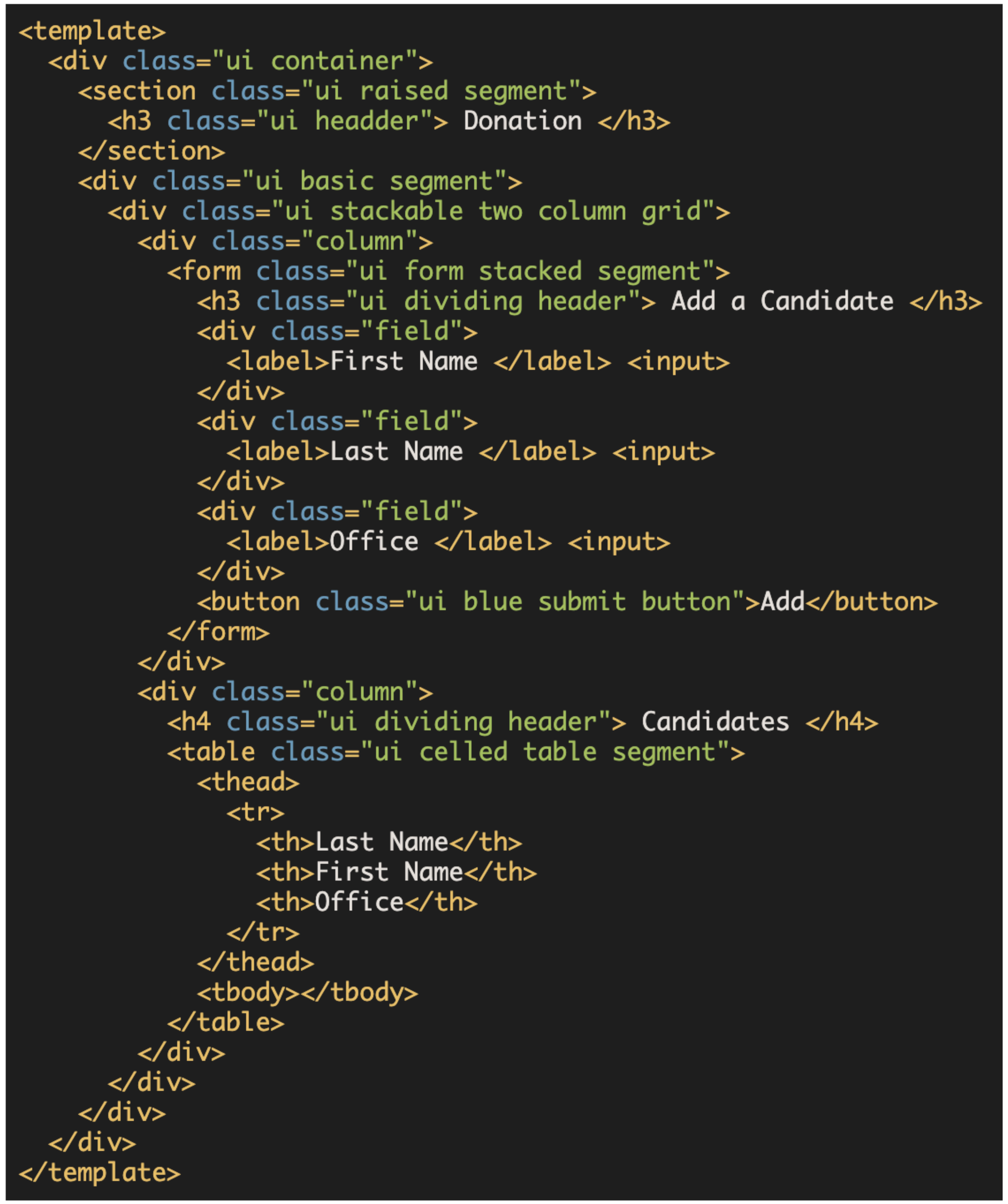
the ViewModel

- TypeScript Class
- encapsulates:
  - details for one candidate (firstName, lastName, office)
  - array of candidates
  - method to add candidate to candidate array



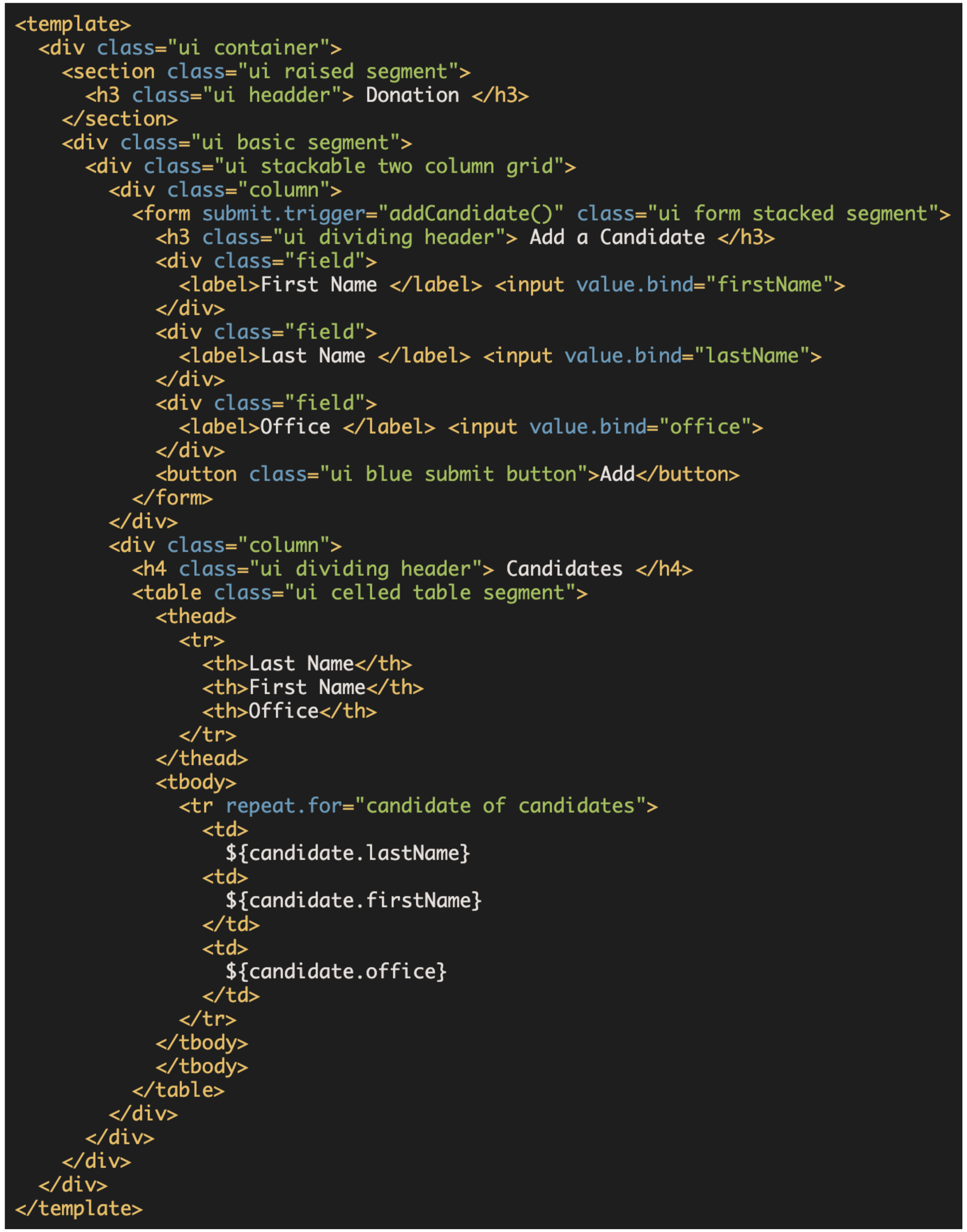
```
▼ src
  ▶ resources
    app.html
    app.ts
    environment.ts
    main.ts
```

Data Binding



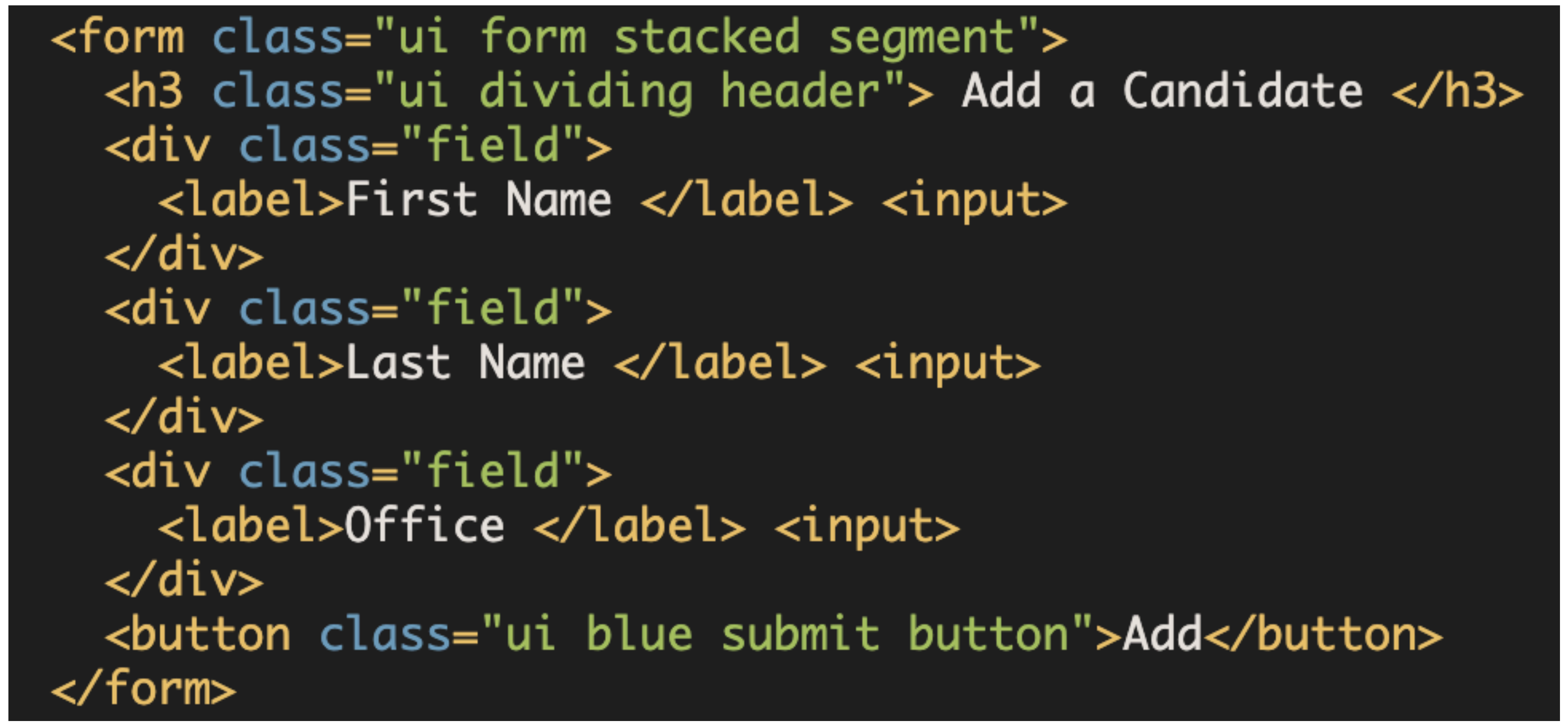
```html
<template>
  <div class="ui container">
    <section class="ui raised segment">
      <h3 class="ui headder"> Donation </h3>
    </section>
    <div class="ui basic segment">
      <div class="ui stackable two column grid">
        <div class="column">
          <form class="ui form stacked segment">
            <h3 class="ui dividing header"> Add a Candidate </h3>
            <div class="field">
              <label>First Name </label> <input>
            </div>
            <div class="field">
              <label>Last Name </label> <input>
            </div>
            <div class="field">
              <label>Office </label> <input>
            </div>
            <button class="ui blue submit button">Add</button>
          </form>
        </div>
        <div class="column">
          <h4 class="ui dividing header"> Candidates </h4>
          <table class="ui celled table segment">
            <thead>
              <tr>
                <th>Last Name</th>
                <th>First Name</th>
                <th>Office</th>
              </tr>
            </thead>
            <tbody></tbody>
          </table>
        </div>
      </div>
    </div>
  </div>
</template>
```

app.html

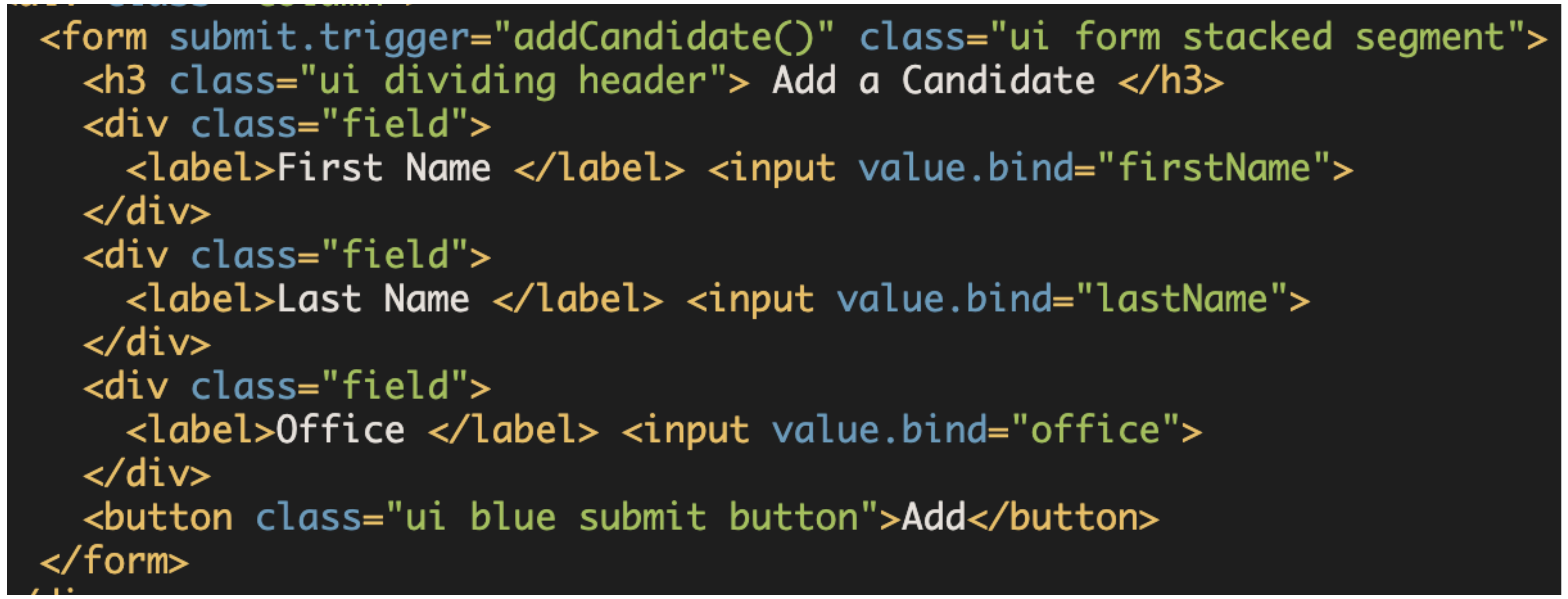


```html
<template>
  <div class="ui container">
    <section class="ui raised segment">
      <h3 class="ui headder"> Donation </h3>
    </section>
    <div class="ui basic segment">
      <div class="ui stackable two column grid">
        <div class="column">
          <form submit.trigger="addCandidate()" class="ui form stacked segment">
            <h3 class="ui dividing header"> Add a Candidate </h3>
            <div class="field">
              <label>First Name </label> <input value.bind="firstName">
            </div>
            <div class="field">
              <label>Last Name </label> <input value.bind="lastName">
            </div>
            <div class="field">
              <label>Office </label> <input value.bind="office">
            </div>
            <button class="ui blue submit button">Add</button>
          </form>
        </div>
        <div class="column">
          <h4 class="ui dividing header"> Candidates </h4>
          <table class="ui celled table segment">
            <thead>
              <tr>
                <th>Last Name</th>
                <th>First Name</th>
                <th>Office</th>
              </tr>
            </thead>
            <tbody>
              <tr repeat.for="candidate of candidates">
                <td>
                  ${candidate.lastName}
                <td>
                  ${candidate.firstName}
                </td>
                <td>
                  ${candidate.office}
                </td>
              </tr>
            </tbody>
            </tbody>
          </table>
        </div>
      </div>
    </div>
  </div>
</template>
```

submit.trigger & value.bind



```
<form class="ui form stacked segment">
  <h3 class="ui dividing header"> Add a Candidate </h3>
  <div class="field">
    <label>First Name </label> <input>
  </div>
  <div class="field">
    <label>Last Name </label> <input>
  </div>
  <div class="field">
    <label>Office </label> <input>
  </div>
  <button class="ui blue submit button">Add</button>
</form>
```

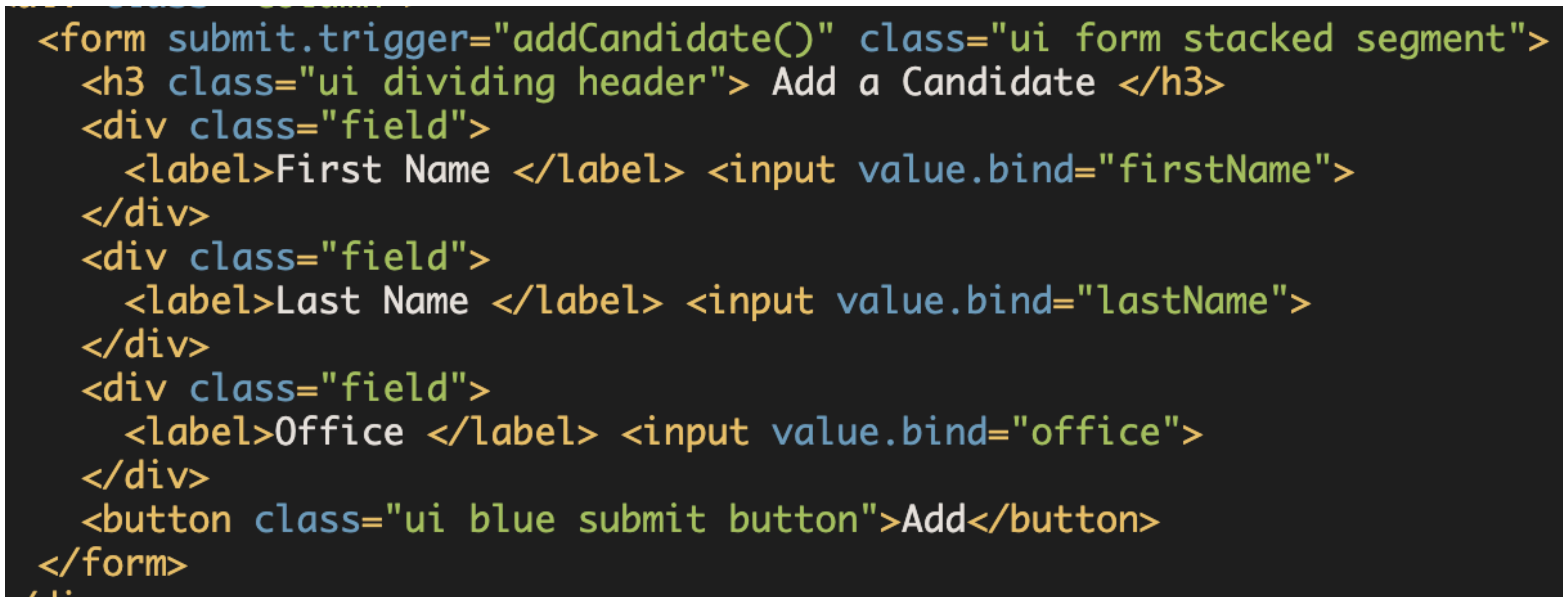


```
<form submit.trigger="addCandidate()" class="ui form stacked segment">
  <h3 class="ui dividing header"> Add a Candidate </h3>
  <div class="field">
    <label>First Name </label> <input value.bind="firstName">
  </div>
  <div class="field">
    <label>Last Name </label> <input value.bind="lastName">
  </div>
  <div class="field">
    <label>Office </label> <input value.bind="office">
  </div>
  <button class="ui blue submit button">Add</button>
</form>
```

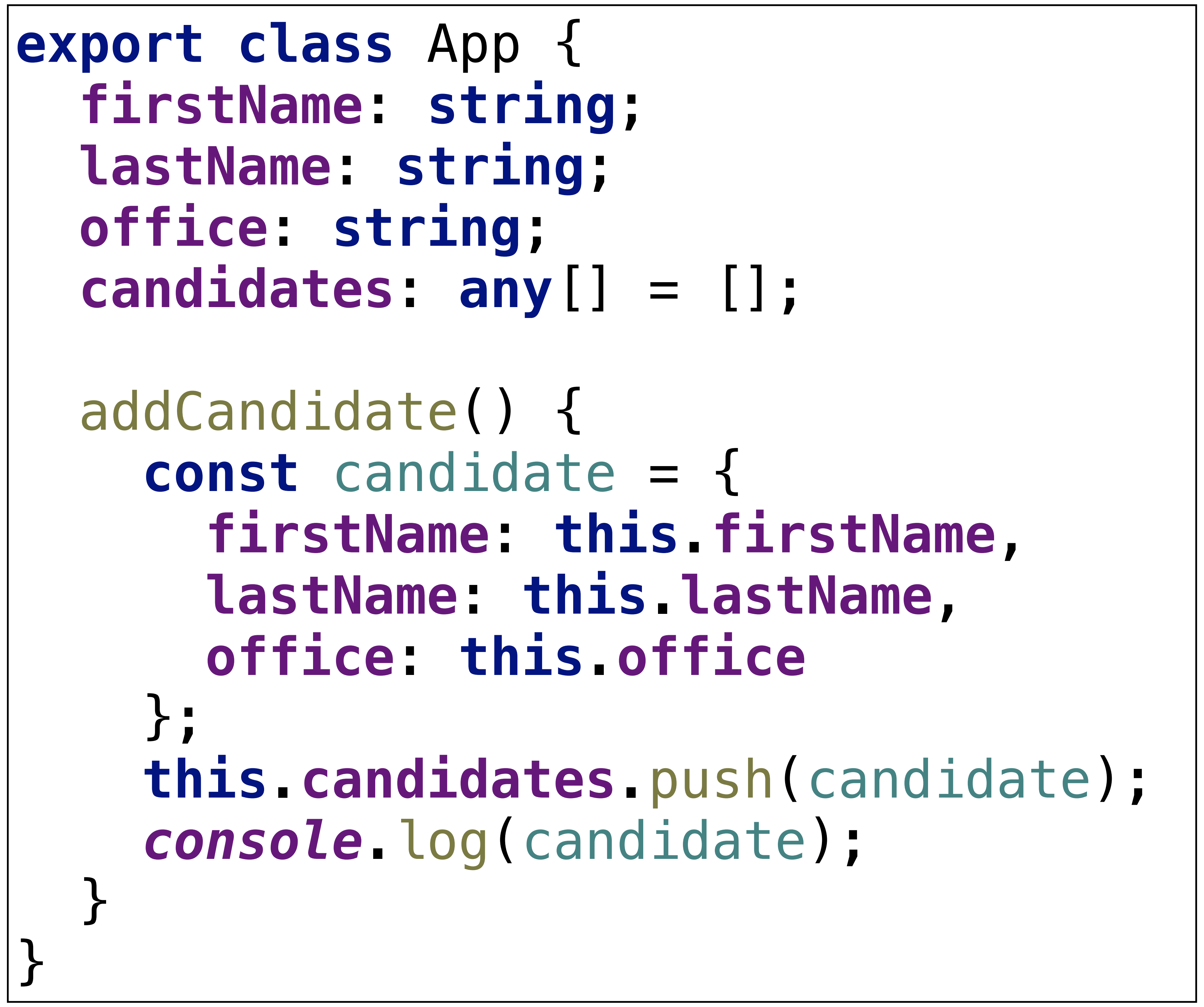
- submit.trigger => denote function to be called in ViewModel

- value.bind => denote field to be updated in ViewModel

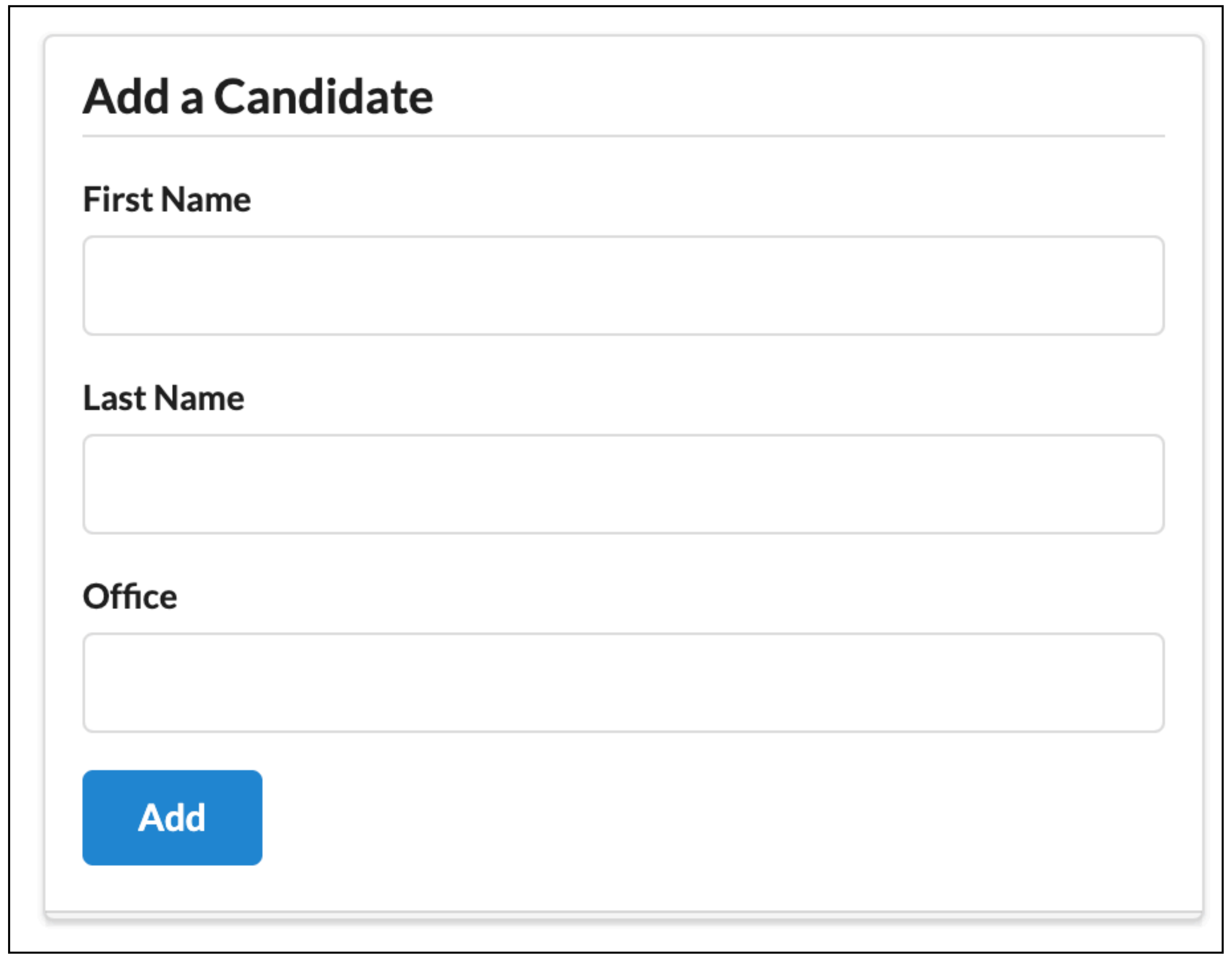
View / ViewModel Interaction



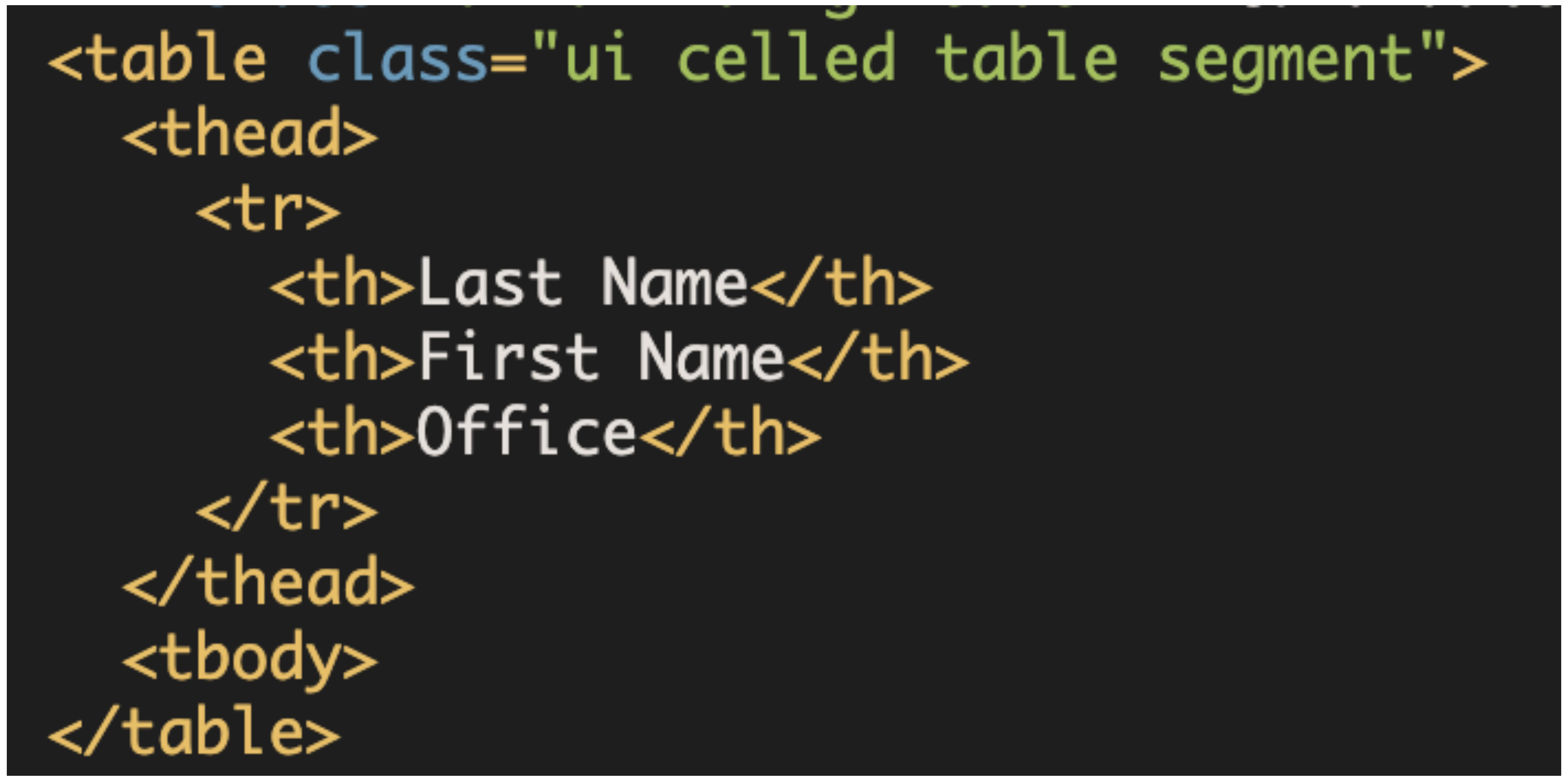
```html
<form submit.trigger="addCandidate()" class="ui form stacked segment">
  <h3 class="ui dividing header"> Add a Candidate </h3>
  <div class="field">
    <label>First Name </label> <input value.bind="firstName">
  </div>
  <div class="field">
    <label>Last Name </label> <input value.bind="lastName">
  </div>
  <div class="field">
    <label>Office </label> <input value.bind="office">
  </div>
  <button class="ui blue submit button">Add</button>
</form>
```



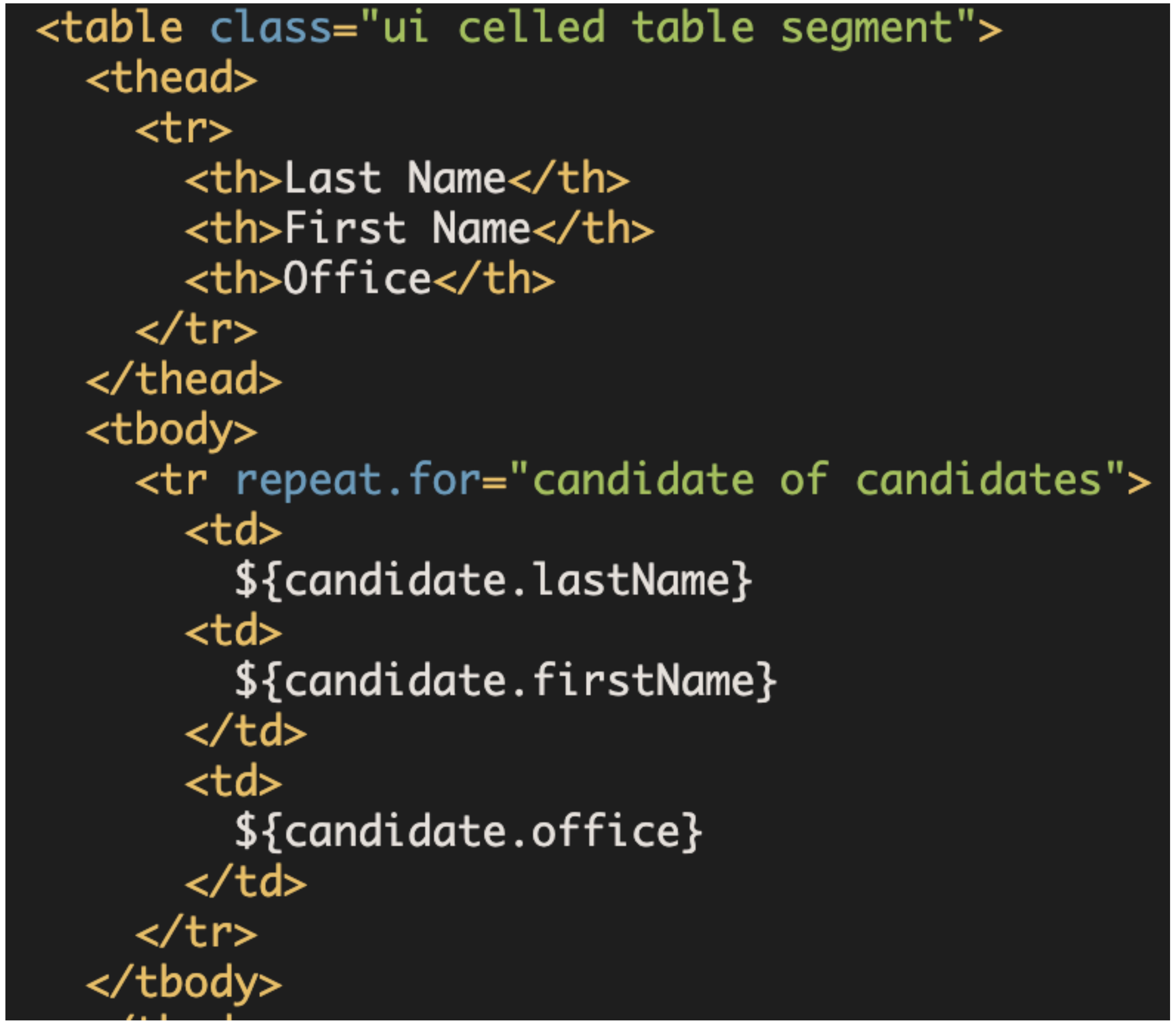
```typescript
export class App {
  firstName: string;
  lastName: string;
  office: string;
  candidates: any[] = [];

  addCandidate() {
    const candidate = {
      firstName: this.firstName,
      lastName: this.lastName,
      office: this.office
    };
    this.candidates.push(candidate);
    console.log(candidate);
  }
}
```



- Filling form + pressing ‘Add’ triggers addCandidate() method

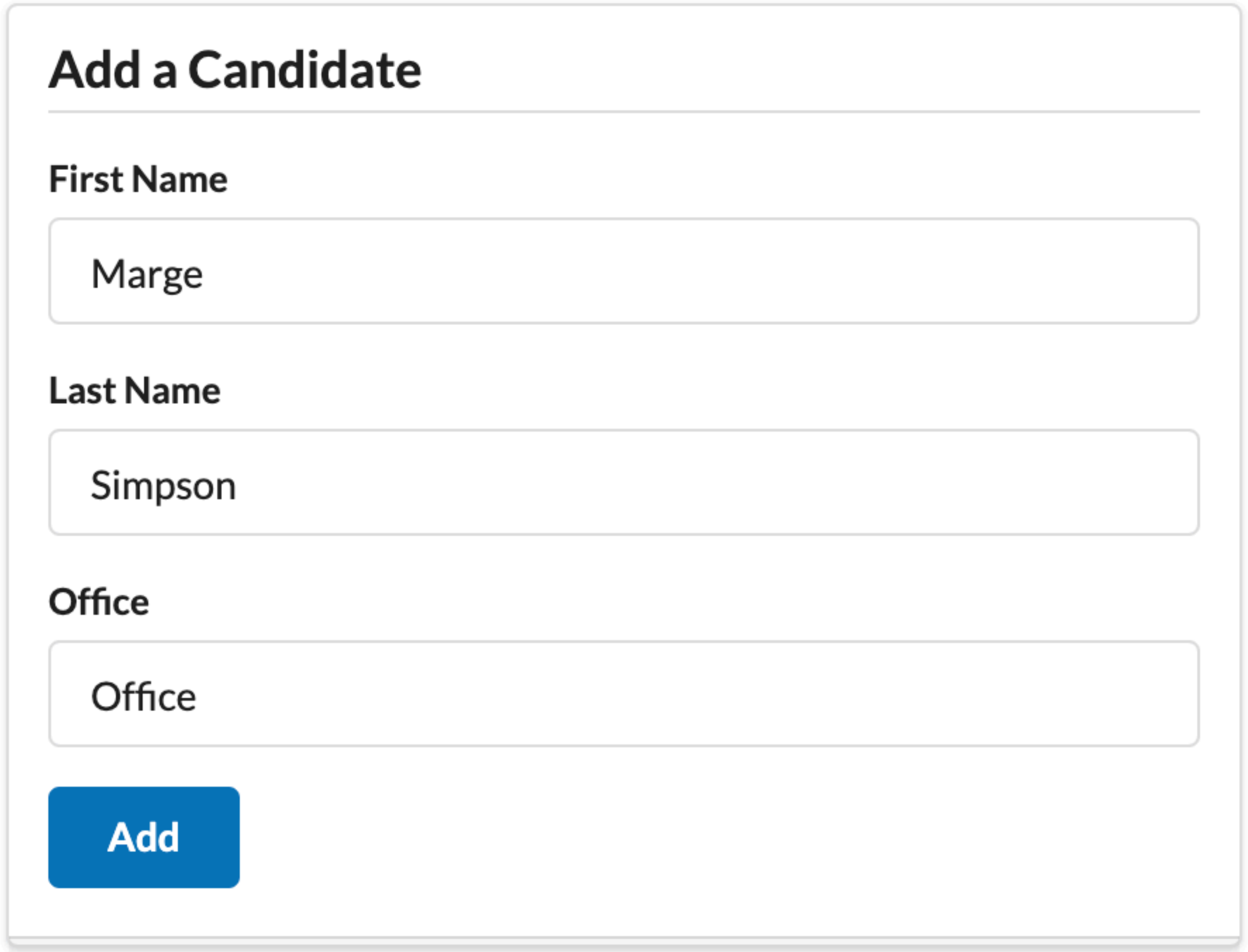


```
<table class="ui celled table segment">
  <thead>
    <tr>
      <th>Last Name</th>
      <th>First Name</th>
      <th>Office</th>
    </tr>
  </thead>
  <tbody>
</table>
```



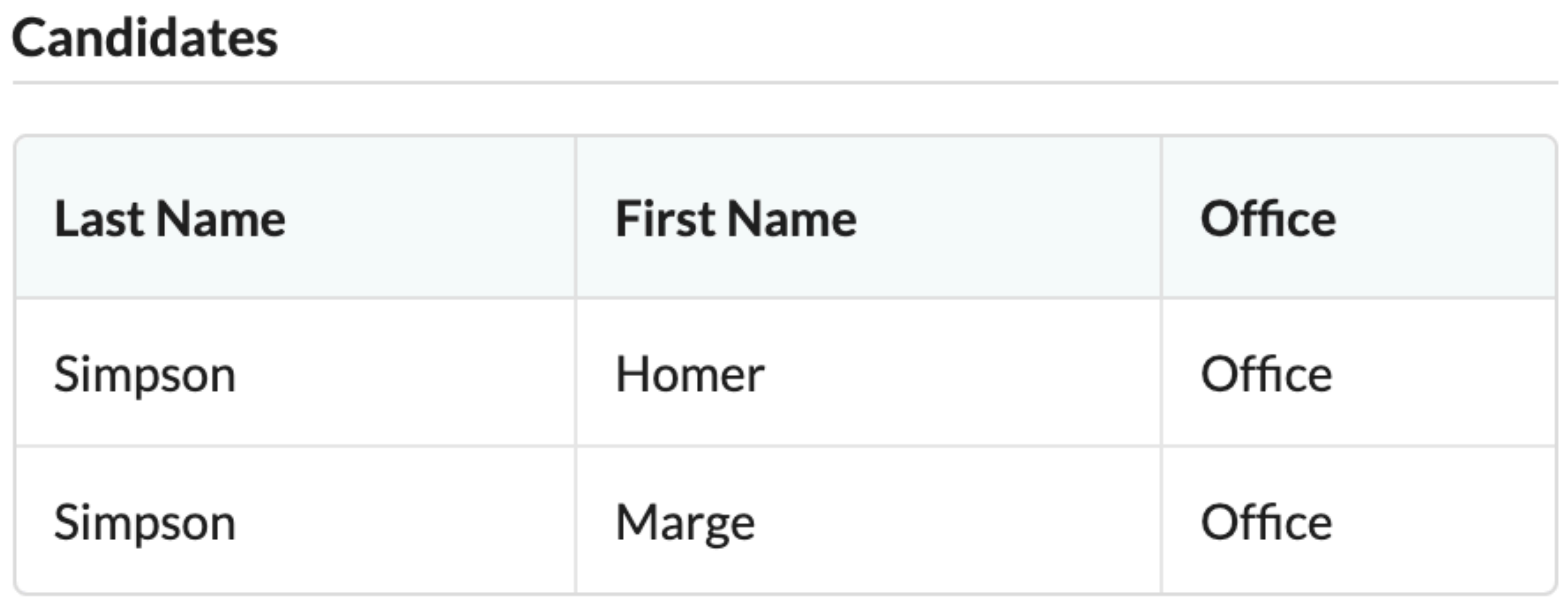
```
<table class="ui celled table segment">
  <thead>
    <tr>
      <th>Last Name</th>
      <th>First Name</th>
      <th>Office</th>
    </tr>
  </thead>
  <tbody>
    <tr repeat.for="candidate of candidates">
      <td>
        ${candidate.lastName}
      <td>
        ${candidate.firstName}
      </td>
      <td>
        ${candidate.office}
      </td>
    </tr>
  </tbody>
```

Data Binding



Add a Candidate

First Name	Marge
Last Name	Simpson
Office	Office

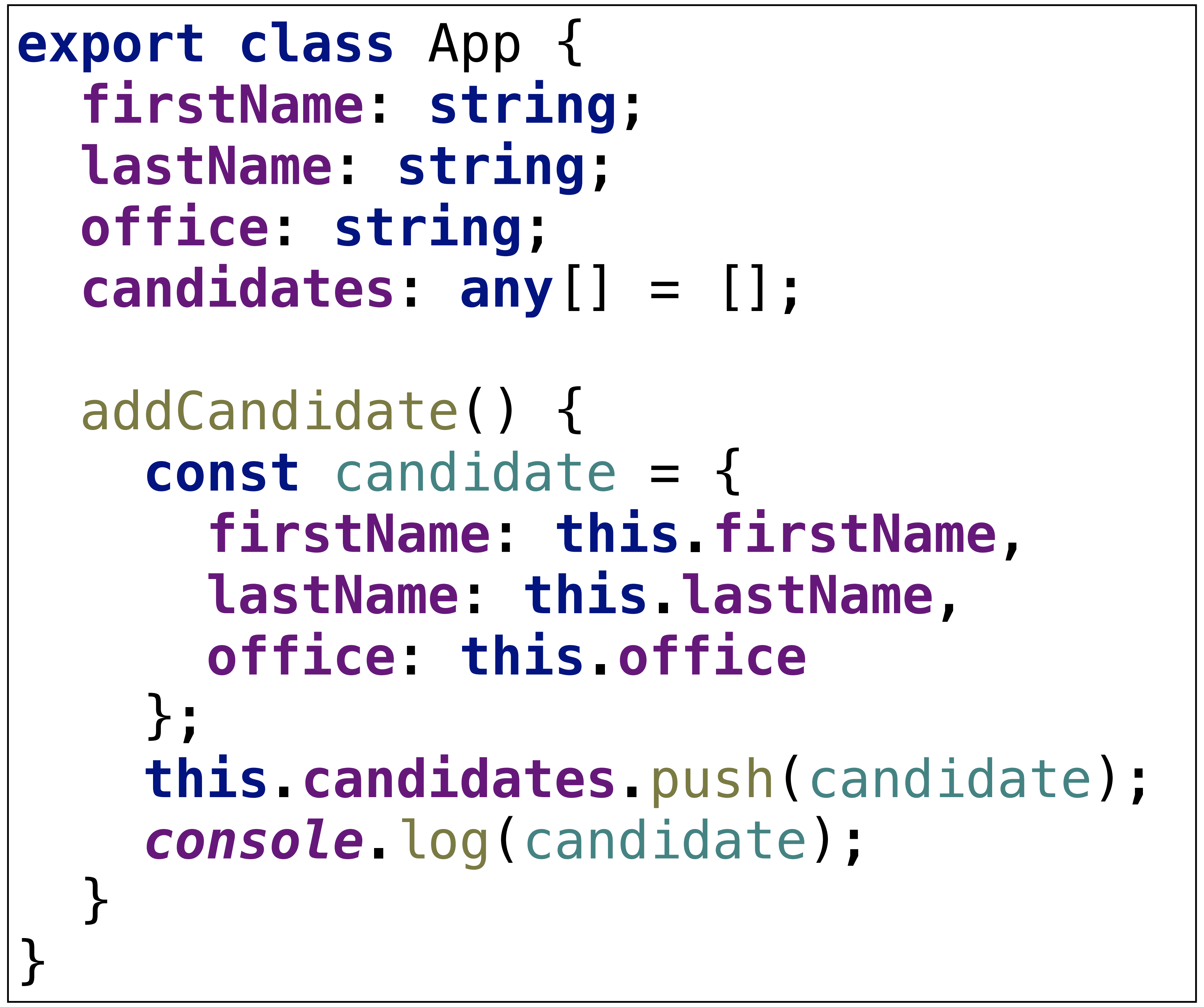


Candidates

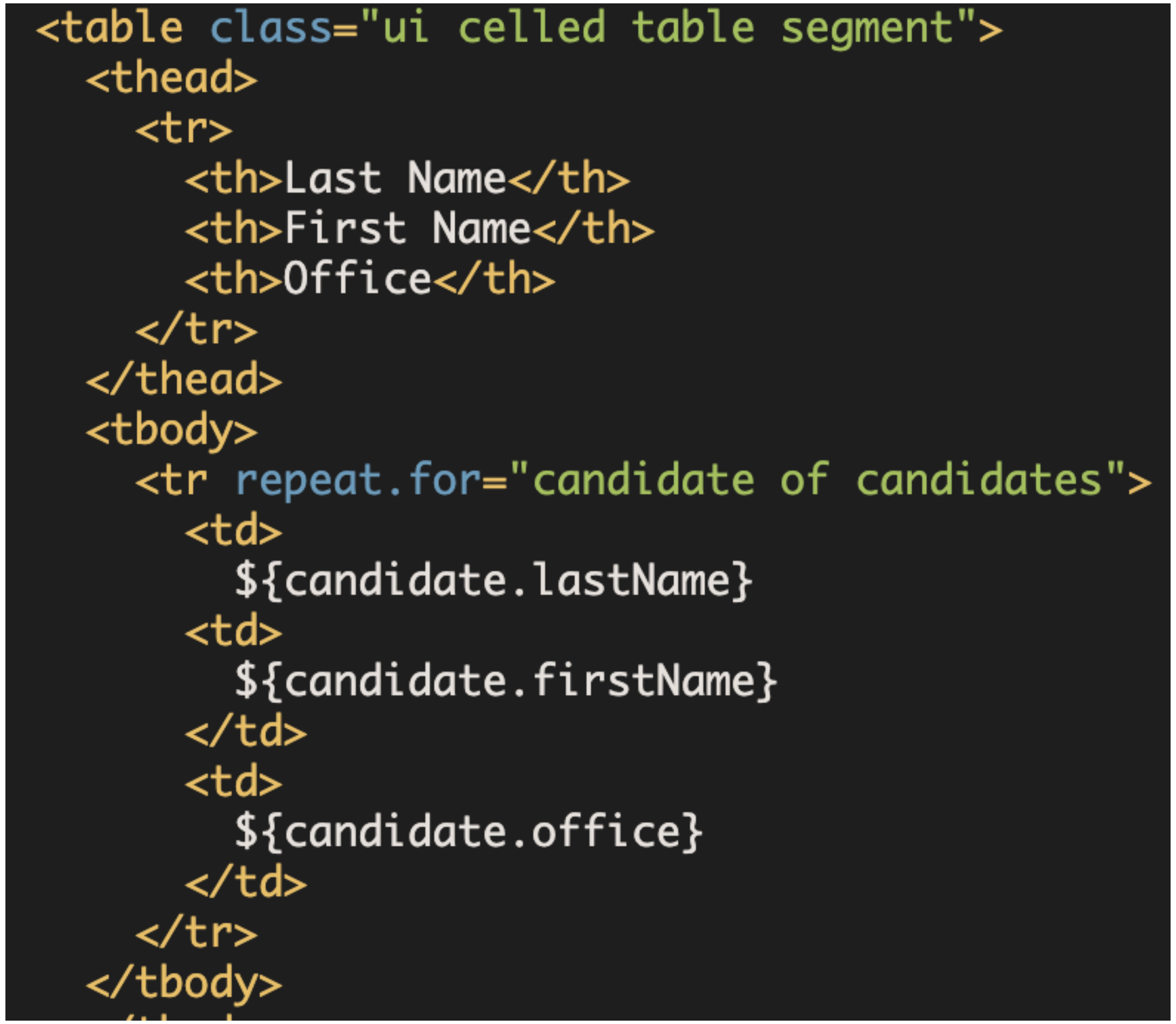
Last Name	First Name	Office
Simpson	Homer	Office
Simpson	Marge	Office

- Add candidates in form
- Table populated with new candidates

repeat.for



```
export class App {
  firstName: string;
  lastName: string;
  office: string;
  candidates: any[] = [];

  addCandidate() {
    const candidate = {
      firstName: this.firstName,
      lastName: this.lastName,
      office: this.office
    };
    this.candidates.push(candidate);
    console.log(candidate);
  }
}
```



```
<table class="ui celled table segment">
  <thead>
    <tr>
      <th>Last Name</th>
      <th>First Name</th>
      <th>Office</th>
    </tr>
  </thead>
  <tbody>
    <tr repeat.for="candidate of candidates">
      <td>
        ${candidate.lastName}
      <td>
        ${candidate.firstName}
      </td>
      <td>
        ${candidate.office}
      </td>
    </tr>
  </tbody>
```

- Iterate over candidates array
- Automatically triggered whenever a new element is added to the array