Angular Forms

Lose your personal data in a ‘classy’ way.

What is a form?

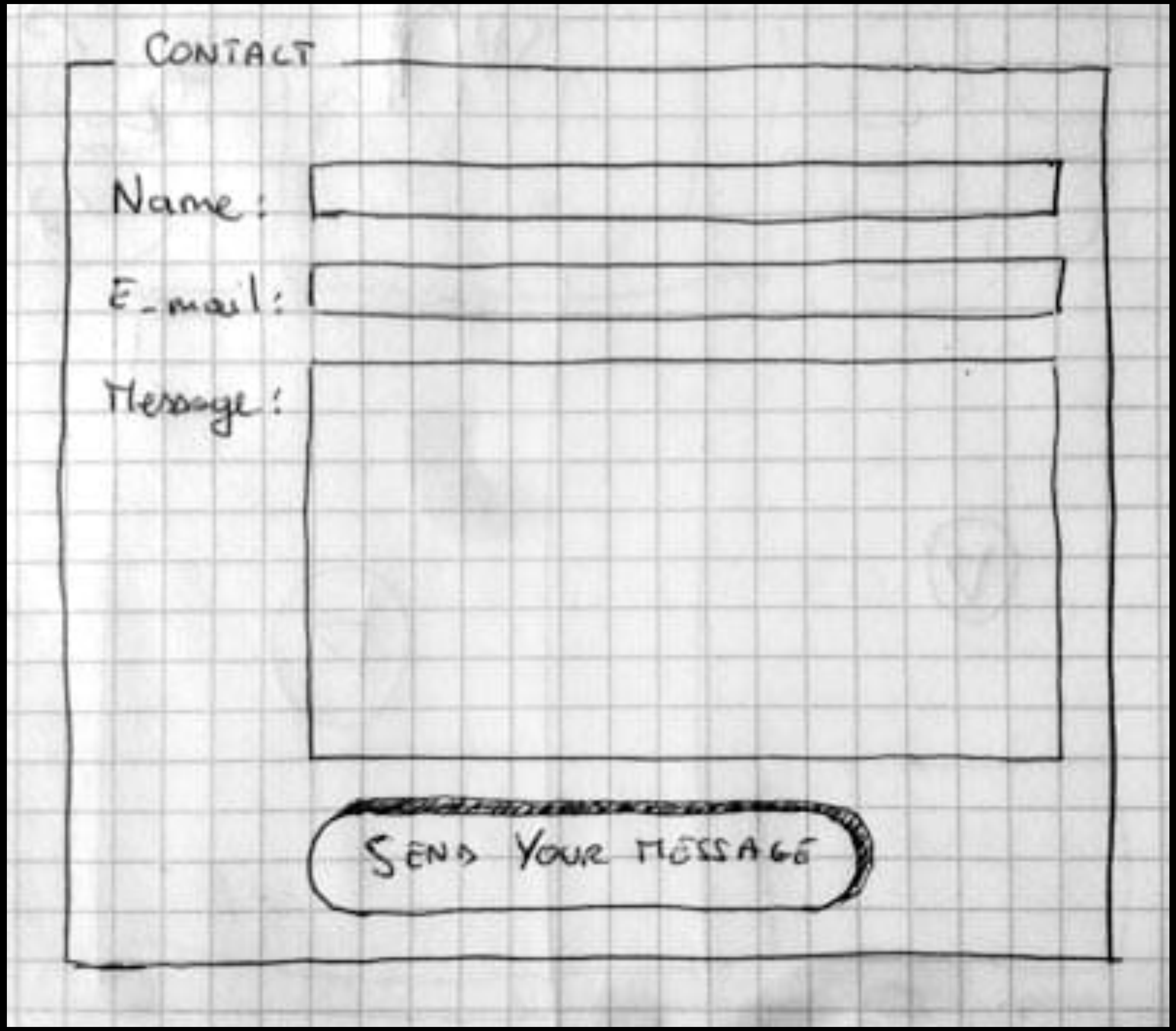
A HTML form is the standard way to gather user input in a website.

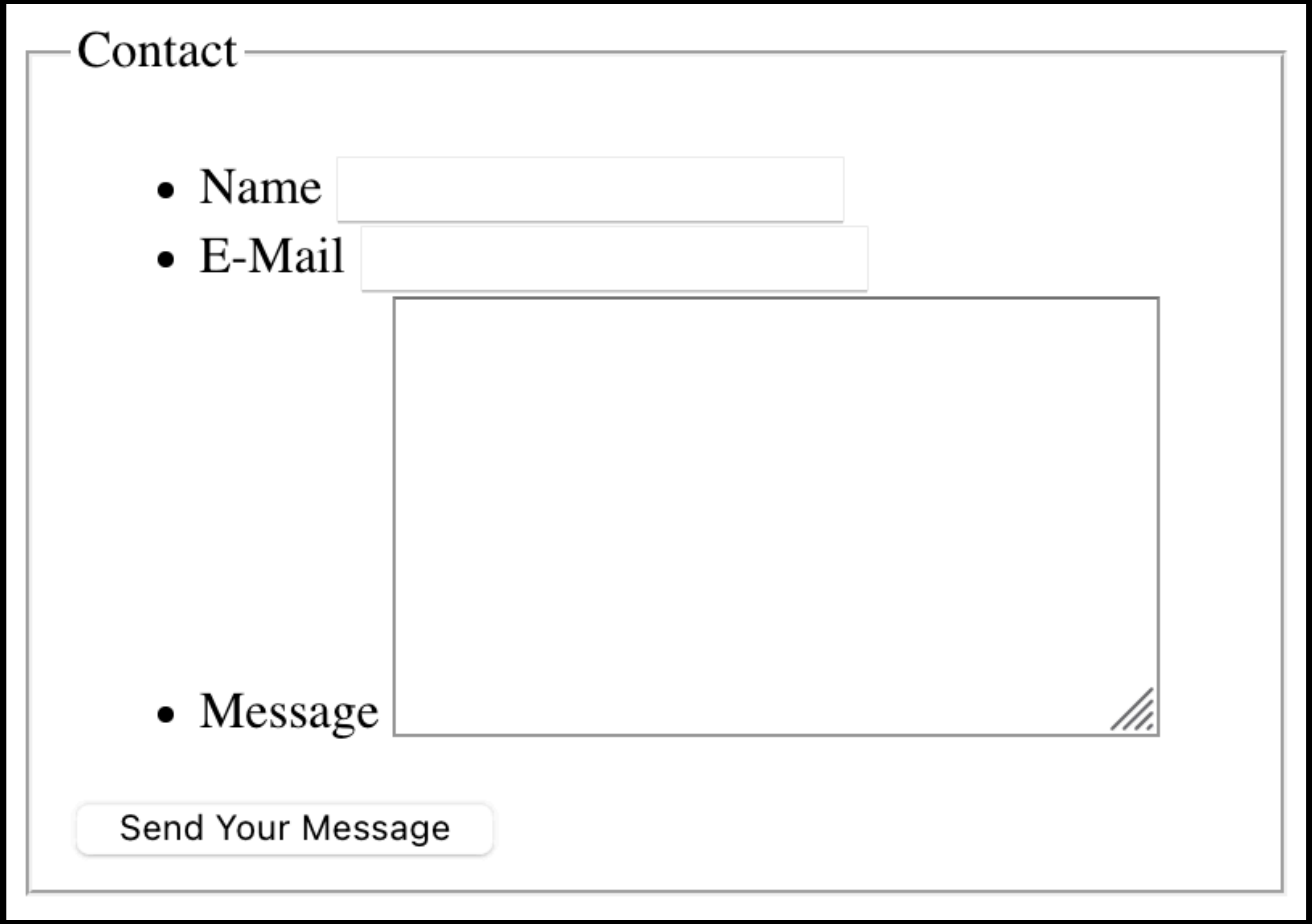
```
<form>
    <input>
    <!--< ... >-->
</form>
```

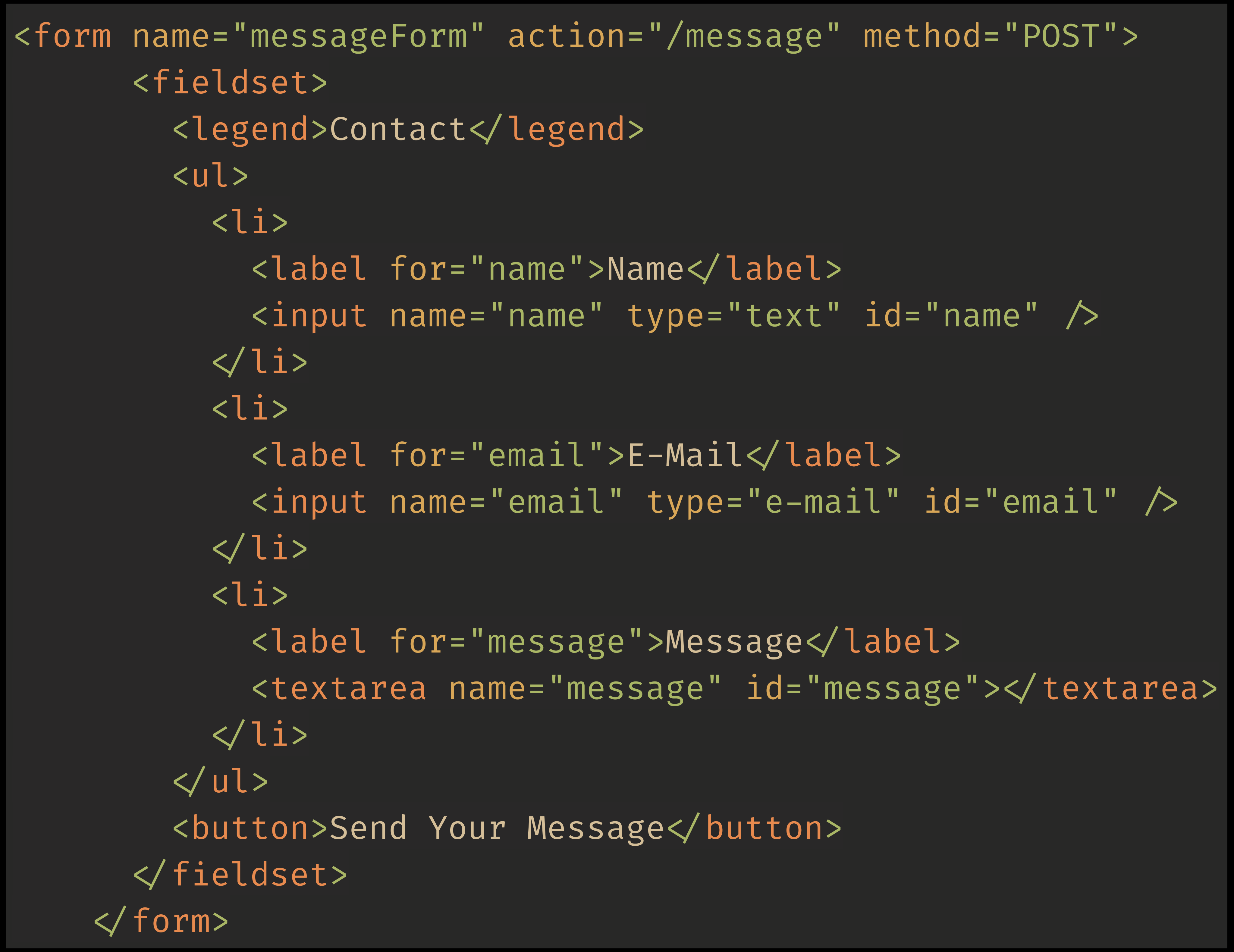
What are form controls?

- Form controls are the basic building elements of a form to facilitate user input of data.
- Lots of different native inputs allow browser-native configuration and validation of input values.

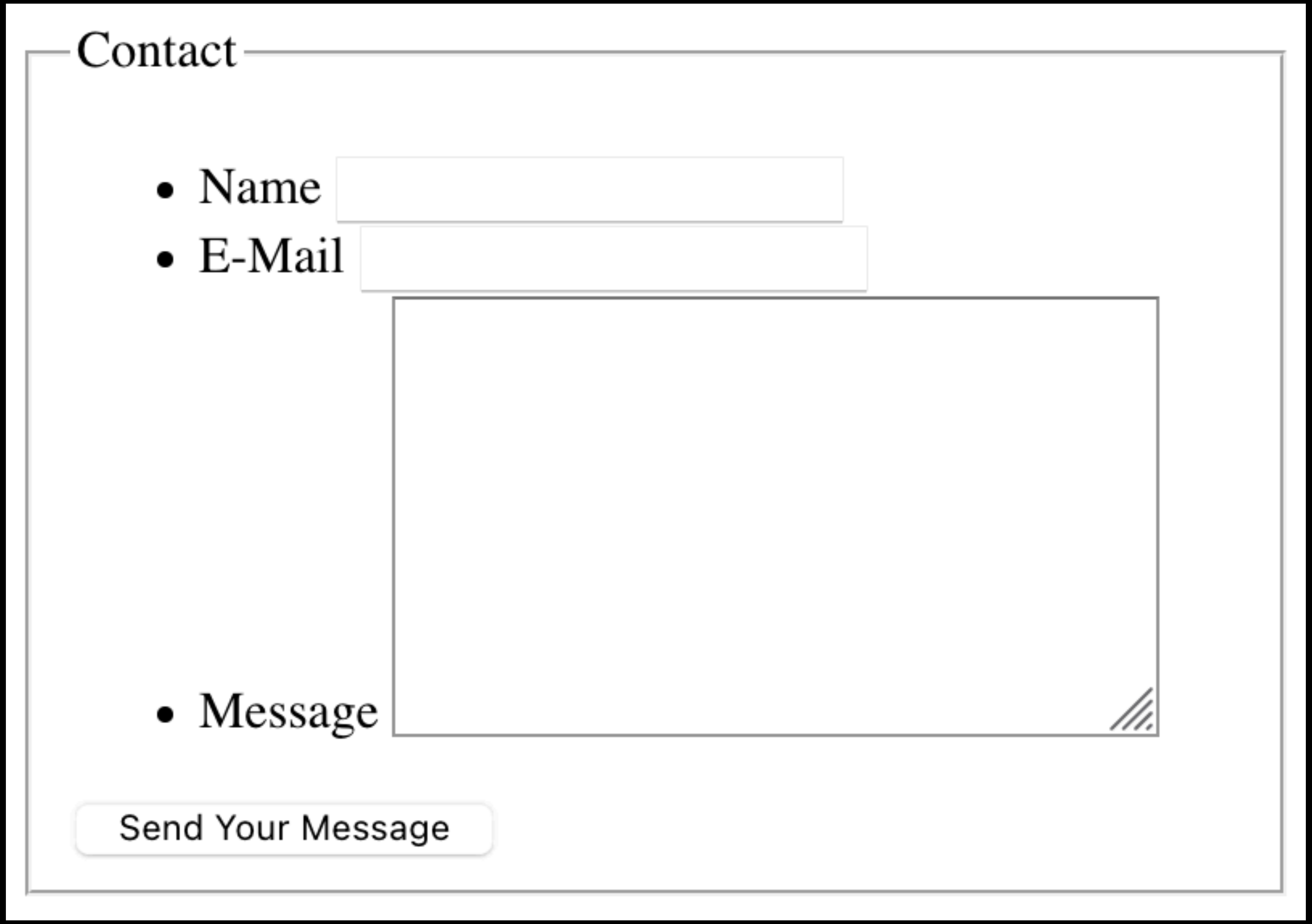
```html
<form>
  <ul>
   <li>
    <label for="text">Text</label>
    <input id="text" type="text" />
   </li>
   <li>
    <label for="number">Number</label>
    <input id="number" type="number" />
   </li>
   <li>
    <label for="date">Date</label>
    <input id="date" type="date" />
   </li>
   <li>
    <label for="password">Password</label>
    <input id="password" type="password" />
   </li>
   <li>
    <label for="color">Color</label>
    <input id="color" type="color" />
   </li>
   <li>
    <label for="range">Range</label>
    <input id="range" type="range" />
   </li>
  </ul>
 </form>
```


CONTACT
Name:
E-mail:
Message:
SEND YOUR MESSAGE


Contact
Name
E-Mail
Message
Send Your Message



```html
<form name="messageForm" action="/message" method="POST">
      <fieldset>
        <legend>Contact</legend>
        <ul>
          <li>
            <label for="name">Name</label>
            <input name="name" type="text" id="name" />
          </li>
          <li>
            <label for="email">E-Mail</label>
            <input name="email" type="e-mail" id="email" />
          </li>
          <li>
            <label for="message">Message</label>
            <textarea name="message" id="message"></textarea>
          </li>
        </ul>
        <button>Send Your Message</button>
      </fieldset>
    </form>
```



Is it really? <form>

https://menti.com - 7622 2587

Angular forms.

Template driven & Reactive.

- Template driven forms
  - Use two-way data-binding to map data changes between template and code. Validation uses native html validators.
  - Great for simple, quick forms.
- Reactive forms
  - Use an immutable data model for form data, which is independent of the template and unique every time.
  - Good for complex data-heavy forms with cross field or asynchronous validation. Possible to unit test.

Template driven forms

```
import { FormsModule } from '@angular/forms';
@NgModule({
  declarations: [
    AppComponent,
  ],
  imports: [FormsModule],
  providers: [],
  bootstrap: [AppComponent],
})
export class AppModule {}
```

```
<form #expenseForm (ngSubmit)="addExpense()">
  <div class="row">
    <div class="col">
      <input
        type="text"
        class="form-control"
        placeholder="Expense name"
        name="name"
        [(ngModel)]="model.name"
      />
    </div>
```

Template driven forms

- Provided by ‘FormsModule’.
- Exposes a form model in the template. Each control in the form corresponds to a property of the model.
- Allows binding of different form controls with the ‘ngForm’ directive.

```
@NgModule({
  declarations: [
    AppComponent,
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    HttpClientModule,
    FormsModule,
  ],
  providers: [],
  bootstrap: [AppComponent],
})
export class AppModule {}
```

```
export class Expense {
  constructor(
    public name: string,
    public date: string,
    public amount: string
  ) {}
}
```

Template Form task #1: Create Expense Form Component

1. Create a separate model file for an expense with the correct interface: expense name, expense date and expense amount.

2. Generate a component for the form.

3. Update all the form controls with the “name” attribute. The “name” should define the “Expense” parameter that is being updated.