# Checkout Form - Layout

# Overview of Angular Forms

- Can easily build forms in Angular
- Supports form data-binding, validation and processing
- Angular provides two types of forms
  - Reactive forms
  - Template-driven forms



© luv2code LLC

# Forms

- **Reactive forms**
  - Leverages programmatic API for form building
  - Scaleable solution that is designed for large, complex forms
  - Forms can be easily reused and tested

- **Template-driven forms**
  - Targeted for small, simple forms
  - Not a scalable solution for large, complex forms

**We will use Reactive forms**


© luv2code LLC

# Comparison

- For comparison of Reactive forms and Template-driven forms

**https://angular.io/guide/forms-overview**


© luv2code LLC

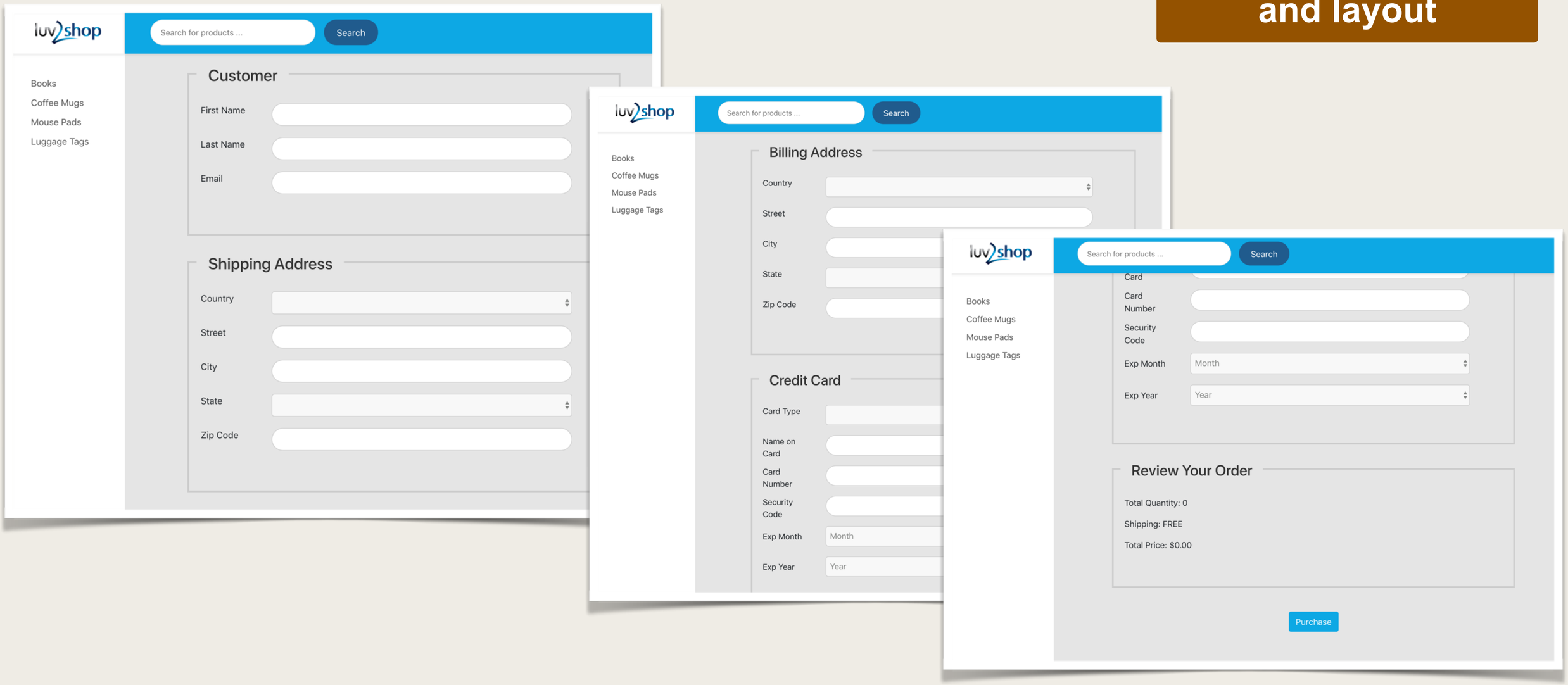
# Key Components

| Name | Description |
| --- | --- |
| **FormControl** | Individual control that tracks the value and validation status |
| **FormGroup** | A collection of controls. Can create nested groups. |
| ***others*** | *...* |


© luv2code LLC

# Our Checkout Form

**Focus on form construction and layout**






© luv2code LLC

# Development Process


1. Generate our checkout component
2. Add a new route for checkout component
3. Create a new checkout button and link to checkout component
4. Add support for reactive forms
5. Define form in component .ts file
6. Layout form controls in HTML template
7. Add event handler for form submission



© luv2code LLC

# Step 1: Generate our checkout component

```
> ng generate component components/checkout
```



© luv2code LLC

# Step 2: Add a new route for checkout component

File: app.module.ts

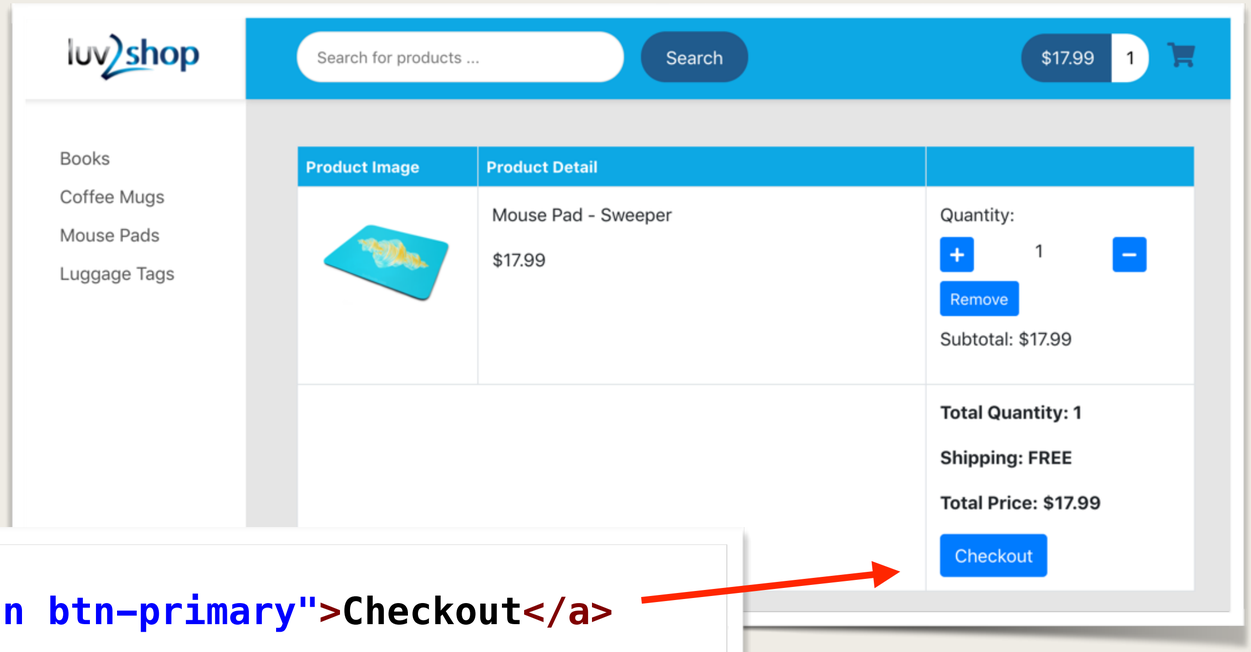
```
const routes: Routes = [
  {path: 'checkout', component: CheckoutComponent},
  ...

];
```



© luv2code LLC

# Step 3: Create new checkout button and link to checkout component




**File: cart-details.component.html**

```
...
<a routerLink='/checkout' class="btn btn-primary">Checkout</a>
...
```


© luv2code LLC

# Step 4: Add support for reactive forms

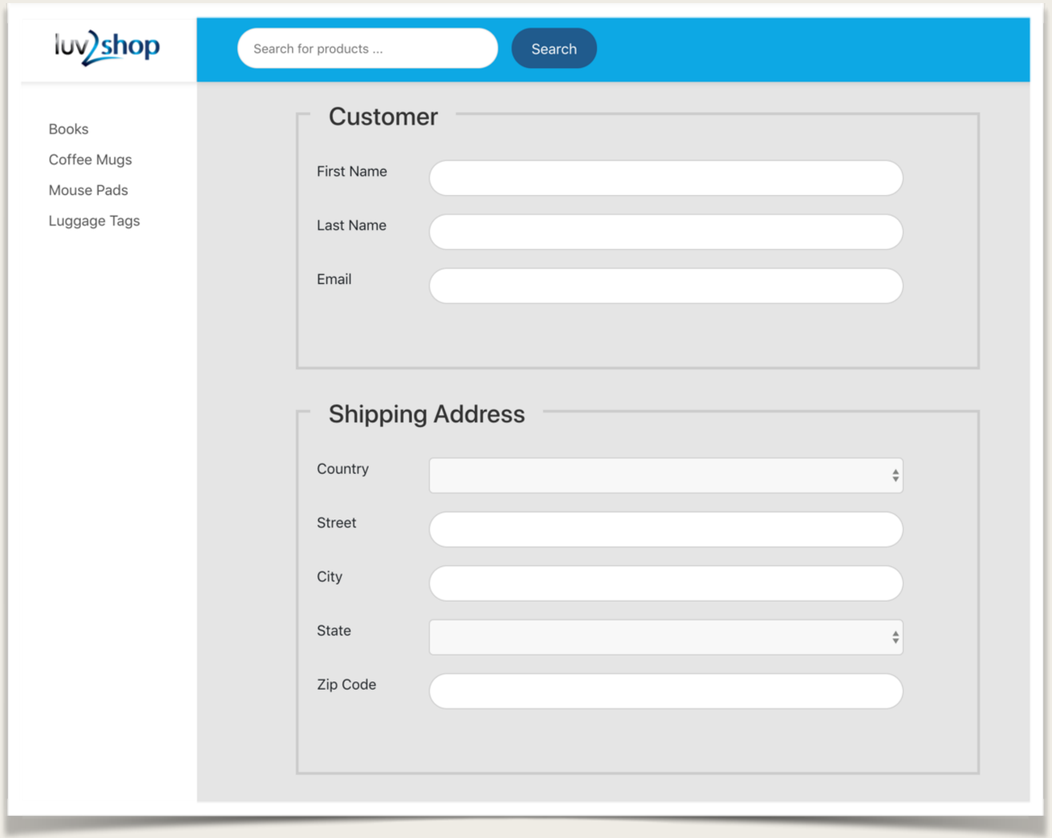
File: app.module.ts

```
...
  imports: [
    ...
    ReactiveFormsModule
  ],
...
```



© luv2code LLC

# Step 5: Define form in component .ts file





© luv2code LLC

# Step 5: Define form in component .ts file

File: checkout.component.ts

```
export class CheckoutComponent implements OnInit {

  checkoutFormGroup: FormGroup;

  constructor(private formBuilder: FormBuilder) { }

  ngOnInit(): void {

    this.checkoutFormGroup = this.formBuilder.group({
      customer: this.formBuilder.group({
        firstName: [''],
        lastName: [''],
        email: ['']
      })
    });
  }

  onSubmit() {
    console.log("Handling the submit button");
    console.log(this.checkoutFormGroup.get('customer').value);
  }
}
```

Inject the form builder

Declare our form group

Build the form

Method to call when submit button is clicked



© luv2code LLC

# Step 6: Layout form controls in HTML template

**Name of property from component class**

File: checkout.component.html

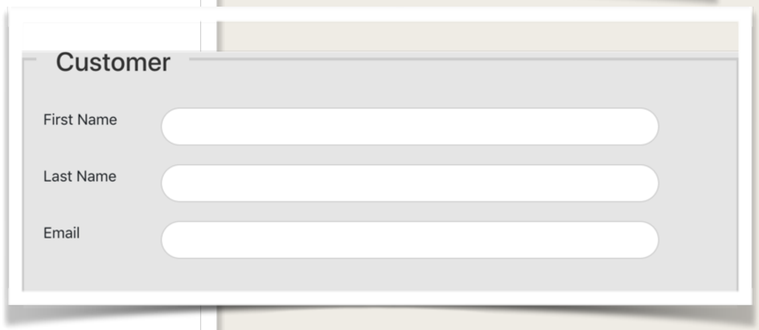
```
<form [formGroup]="checkoutFormGroup">

    <!-- customer form group -->
    <div formGroupName="customer" class="form-area">
        <h3>Customer</h3>

        <div class="row">
            <div class="col-md-2"> <label>First Name</label></div>
            <input formControlName="firstName" type="text">
        </div>
        ...
    </div>
...
</form>
```

```
export class CheckoutComponent implements OnInit {

 checkoutFormGroup: FormGroup;
```




**Repeat process for rows: Last Name, Email**


© luv2code LLC

# Step 7: Add event handler for form submission

File: checkout.component.html

```
<form [formGroup]="checkoutFormGroup" (ngSubmit)="onSubmit()">

    <!-- customer form group -->
    <div formGroupName="customer" class="form-area">
        <h3>Customer</h3>

        <div class="row">
            <div class="col-md-2"> <label>First Name</label></div>
            <input formControlName="firstName" type="text">
       </div>
    </div>

    <div class="text-center">
          <button type="submit" class="btn btn-info">Purchase</button>
    </div>

...
</form>
```

Method to call when submit button is clicked

```
onSubmit() {
  console.log("Handling the submit button");
  console.log(this.checkoutFormGroup.get('customer').value);
}
```

Read the form data

Our submit button



© luv2code LLC

# Step 7: Add event handler for form submission

```
onSubmit() {
  console.log("Handling the submit button");
  console.log(this.checkoutFormGroup.get('customer').value);
}
```

Read the form data

```
console.log("Email is " + this.checkoutFormGroup.get('customer').value.email);
```

Output

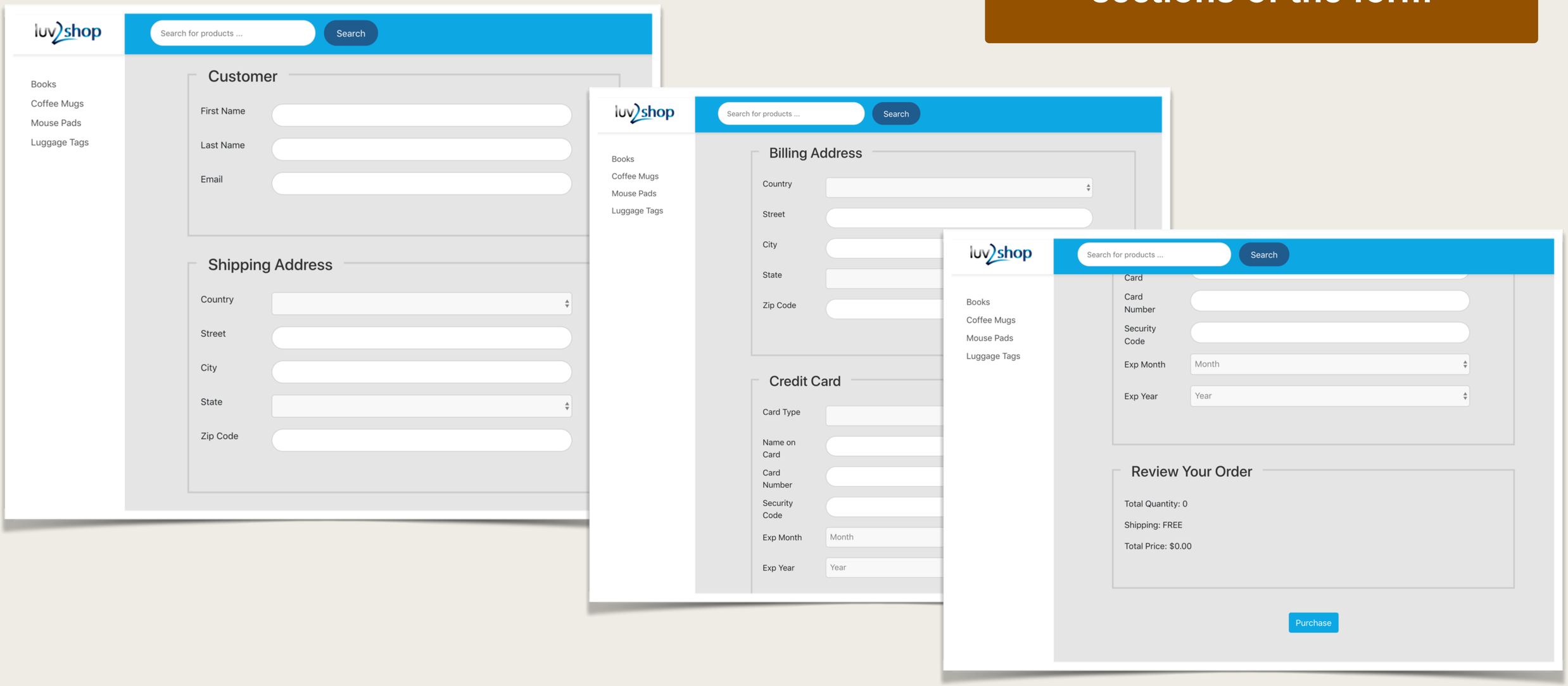
```
Handling the submit button
▼ {firstName: "John", lastName: "Doe", email: "john.doe@luv2code.com"}
    email: "john.doe@luv2code.com"
    firstName: "John"
    lastName: "Doe"
  ▶ __proto__: Object
```



© luv2code LLC

# Our Checkout Form

**Repeat the process for other sections of the form**






© luv2code LLC