Step 7: Add Stripe Elements to Checkout form

- We are going to **delete** some fields/code from our checkout form

- In previous videos, we created fields for
  - Credit card number, ccv, expiration month and years

- We are removing that code (gasp!)
  - I originally was going to use a different payment processor: **authorize.net**

Older ... traditional

- However, **Stripe** provides these form fields with **Stripe Elements**
  - Simplifies development and helps provide PCI compliance

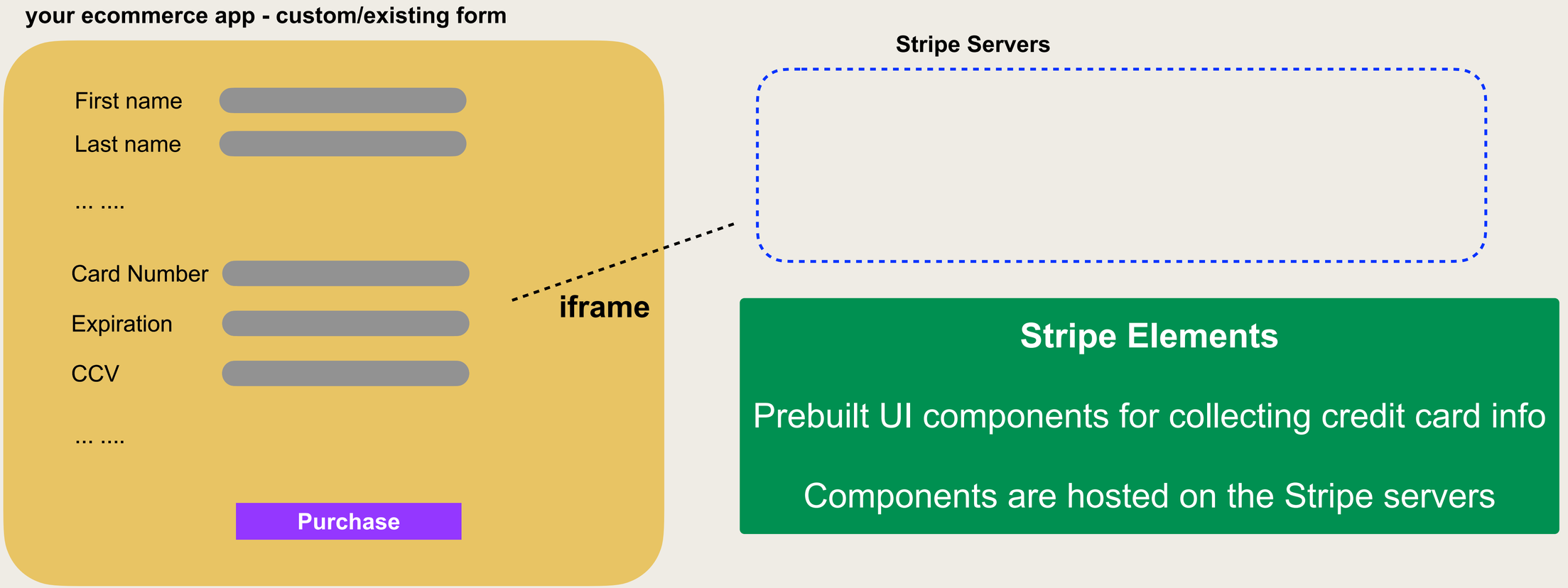
Cool ... modern

Stripe Integration: Custom Code Integration

- Stripe provides **Stripe Elements**
  - Prebuilt UI components for collecting credit card info
  - Components are hosted on the Stripe servers
  - Can easily integrate into your existing/custom checkout form
  - Provide seamless integration
  - Supports PCI ... the credit card data is never sent to your servers

Stripe Integration: Stripe Elements

Provides seamless integration

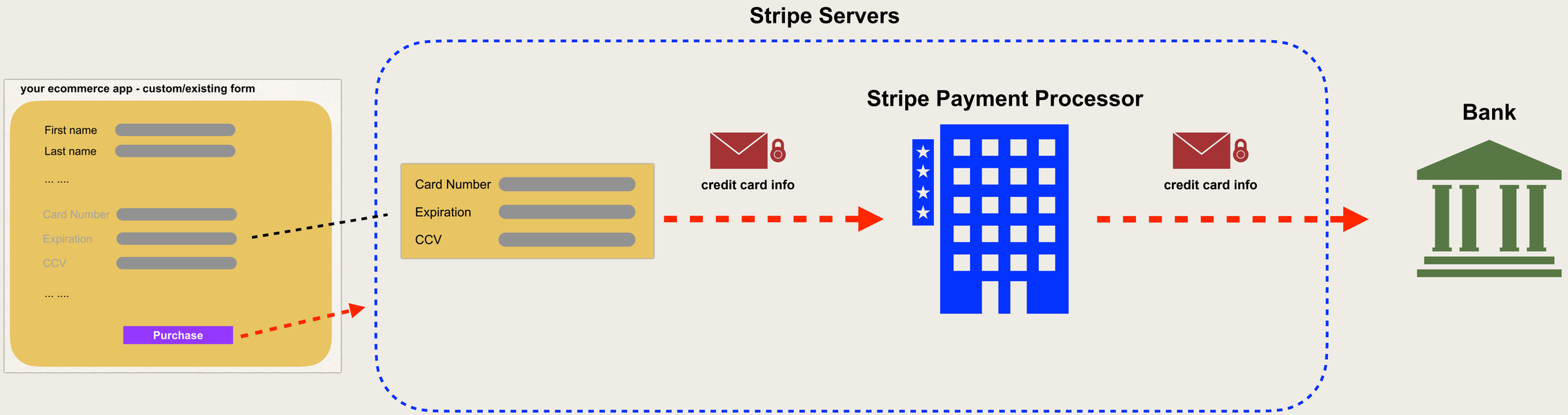


Stripe Elements

Prebuilt UI components for collecting credit card info

Components are hosted on the Stripe servers

Stripe Integration: Stripe Elements



Step 7: Add Stripe Elements to Checkout form

- **Delete** original form fields: card number, ccv, expiration month, year

checkout.component.html

```
...
                <!-- Credit Card -->
                <div formGroupName="creditCard" class="form-area">
                    <h3>Credit Card</h3>

                    <div class="row">
                        <div class="col-md-2"> <label>Card Number</label></div>
                        <div class="col-md-9">
                            <div class="input-space">
                                <input formControlName="cardNumber" type="text">

                            </div>
                        </div>
                    </div>
...
```

Step 7: Add Stripe Elements to Checkout form

- Add support for Stripe Elements

checkout.component.html

```
...
        <!-- Stripe Elements Credit Card Section -->
        <div formGroupName="creditCard" class="form-area">

            <h3>Credit or Debit Card</h3>

            <div id="card-element">
                <!-- a Stripe Element will be inserted here. -->
            </div>

            <!-- Used to display form errors -->
            <div id="card-errors" class="displayError.textContent!==': 'alert alert-danger mt-1'" ></div>

        </div>
...
```

We will configure Stripe to insert the Stripe Elements here

Covered in next couple of slides ...

Step 7: Add Stripe Elements to Checkout form

- Initialize Stripe API
- Create supporting fields

environment.ts

```
export const environment = {
  production: false,
  luv2shopApiUrl: "https://localhost:8443/api",
  stripePublishableKey: "pk_test_xxyyzz112233..."
};
```

checkout.component.ts

```
export class CheckoutComponent implements OnInit {

  // initialize Stripe API
  stripe = Stripe(environment.stripePublishableKey);

  paymentInfo: PaymentInfo = new PaymentInfo();
  cardElement: any;
  displayError: any =

  ...

}
```

Our custom DTO that we created earlier

Will hold a reference to Stripe Elements card component

Step 7: Add Stripe Elements to Checkout form

- Update ngOnInit() method to set up the Stripe payment form

checkout.component.ts

```
export class CheckoutComponent implements OnInit {

  ngOnInit(): void {

    // setup Stripe payment form
    this.setupStripePaymentForm();

    ...

  }

}
```

We'll create code for this method on next slide

Step 7: Add Stripe Elements to Checkout form

checkout.component.ts

```
setupStripePaymentForm() {

  // get a handle to stripe elements
  var elements = this.stripe.elements();

  // Create a card element ... and hide the zip-code field
  this.cardElement = elements.create('card', { hidePostalCode: true });

  // Add an instance of card UI component into the 'card-element' div
  this.cardElement.mount('#card-element');

  // Add event binding for the 'change' event on the card element
  this.cardElement.on('change', (event) => {

    // get a handle to card-errors element
    this.displayError = document.getElementById('card-errors');

    if (event.complete) {
      this.displayError.textContent = "";
    } else if (event.error) {
      // show validation error to customer
      this.displayError.textContent = event.error.message;
    }

  });

}
```

checkout.component.html

```
...
        <!-- Stripe Elements Credit Card Section -->
        <div formGroupName="creditCard" class="form-area">

            <h3>Credit or Debit Card</h3>

            <div id="card-element">
                <!-- a Stripe Element will be inserted here. -->
            </div>

            <!-- Used to display form errors -->
            <div id="card-errors" class="displayError.textContent!==''': 'alert

        </div>
...
```

Step 7: Add Stripe Elements to Checkout form

- In onSubmit() method ... compute the total amount
- Convert amount in dollars to amount in cents. Multiply by 100

checkout.component.ts

```
export class CheckoutComponent implements OnInit {

  onSubmit() {

    // compute payment info
    this.paymentInfo.amount = this.totalPrice * 100;
    this.paymentInfo.currency = "USD";

    ...
  }

}
```

Step 7: Add Stripe Elements to Checkout form

- Make REST API calls for payment processing
  - Create payment intent
  - Confirm card payment
  - Place order

```
export class CheckoutComponent implements OnInit {

  onSubmit() {
      ...
      this.checkoutService.createPaymentIntent(this.paymentInfo).subscribe(
        (paymentIntentResponse) => {
          this.stripe.confirmCardPayment(paymentIntentResponse.client_secret,
            {
              payment_method: {
                card: this.cardElement
              }
            }, { handleActions: false })
          .then(function(result) {
            if (result.error) {
              // inform the customer there was an error
              alert(`There was an error: ${result.error.message}`);
            } else {
              // call REST API via the CheckoutService
              this.checkoutService.placeOrder(purchase).subscribe({
                next: response => {
                  alert(`Your order has been received.\nOrder tracking number: ${response.orderTrackingNumber}`);

                  // reset cart
                  this.resetCart();
                },
                error: err => {
                  alert(`There was an error: ${err.message}`);
                }
              })
            }
          }.bind(this));
        }
      );
    ...
```

Create payment intent
Spring Boot REST API

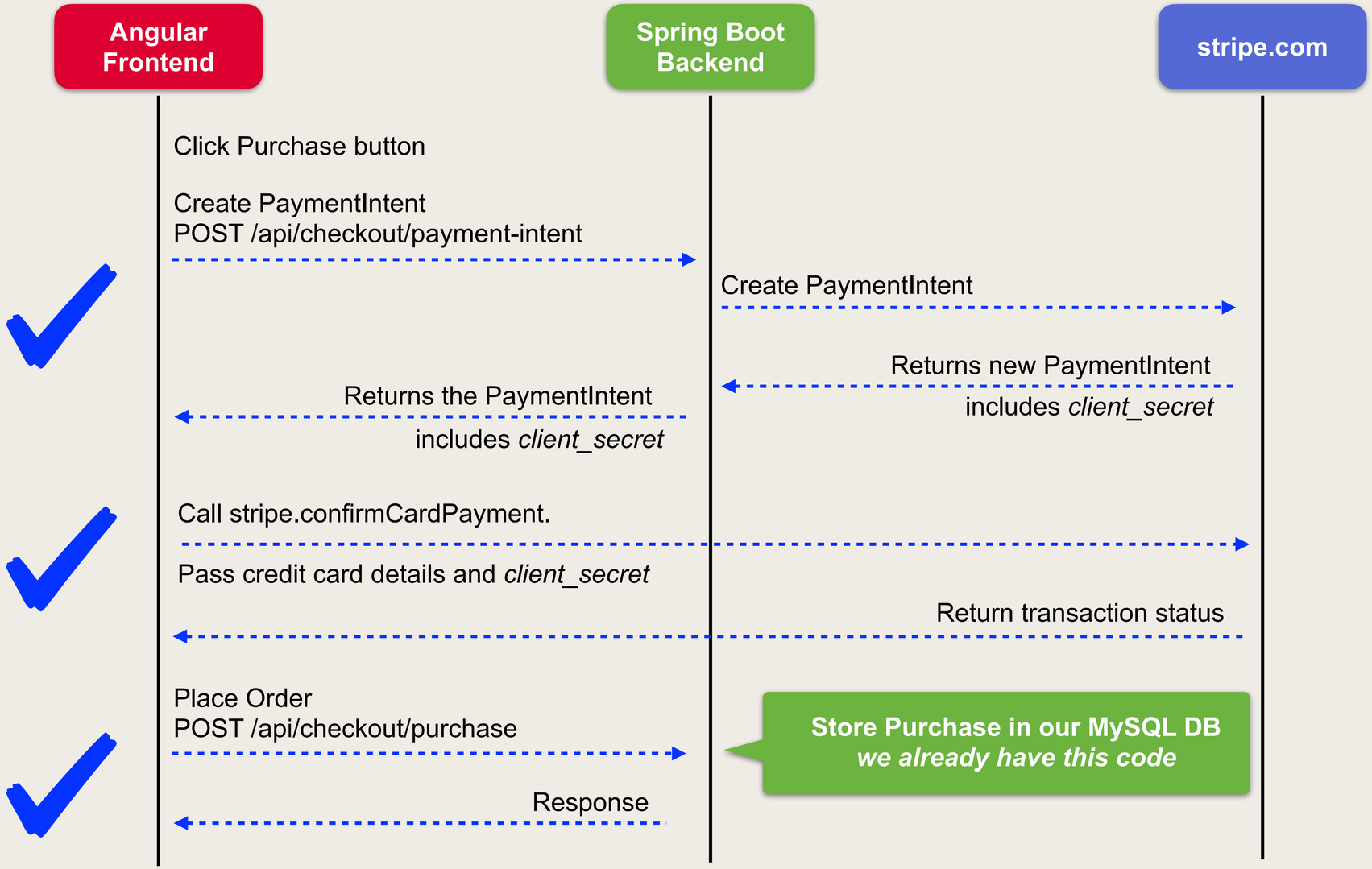
Confirm card payment
stripe.com

Send credit card data directly to stripe.com servers

Reference the Stripe Elements component: cardElement

Place order ... store in MySQL DB
Spring Boot REST API

Stripe Payment Processing Flow



Testing Credit Card Payments

- Stripe provides a **test** sandbox ... no charges are sent to banks
- Sample test card numbers are available for various scenarios
- Real credit card information can not be used in test mode
    - Your personal credit card information will not work in test mode

Testing Credit Card Payments

- Test using any of the following test card numbers
- Valid expiration date in the future
- CVC number can be any three digits

Number	CVC	Expiration Date	Result
4242 4242 4242 4242	Any 3 digits	Any future date	Success
4000 0000 0000 0002	Any 3 digits	Any future date	Card Declined
4000 0000 0000 9995	Any 3 digits	Any future date	Card Declined - Insufficient funds
4000 0000 0000 9979	Any 3 digits	Any future date	Card Declined - Stolen card
...	...	...	...

Testing Credit Card Payments

- Additional test card numbers and scenarios available

https://stripe.com/docs/testing