luv2code

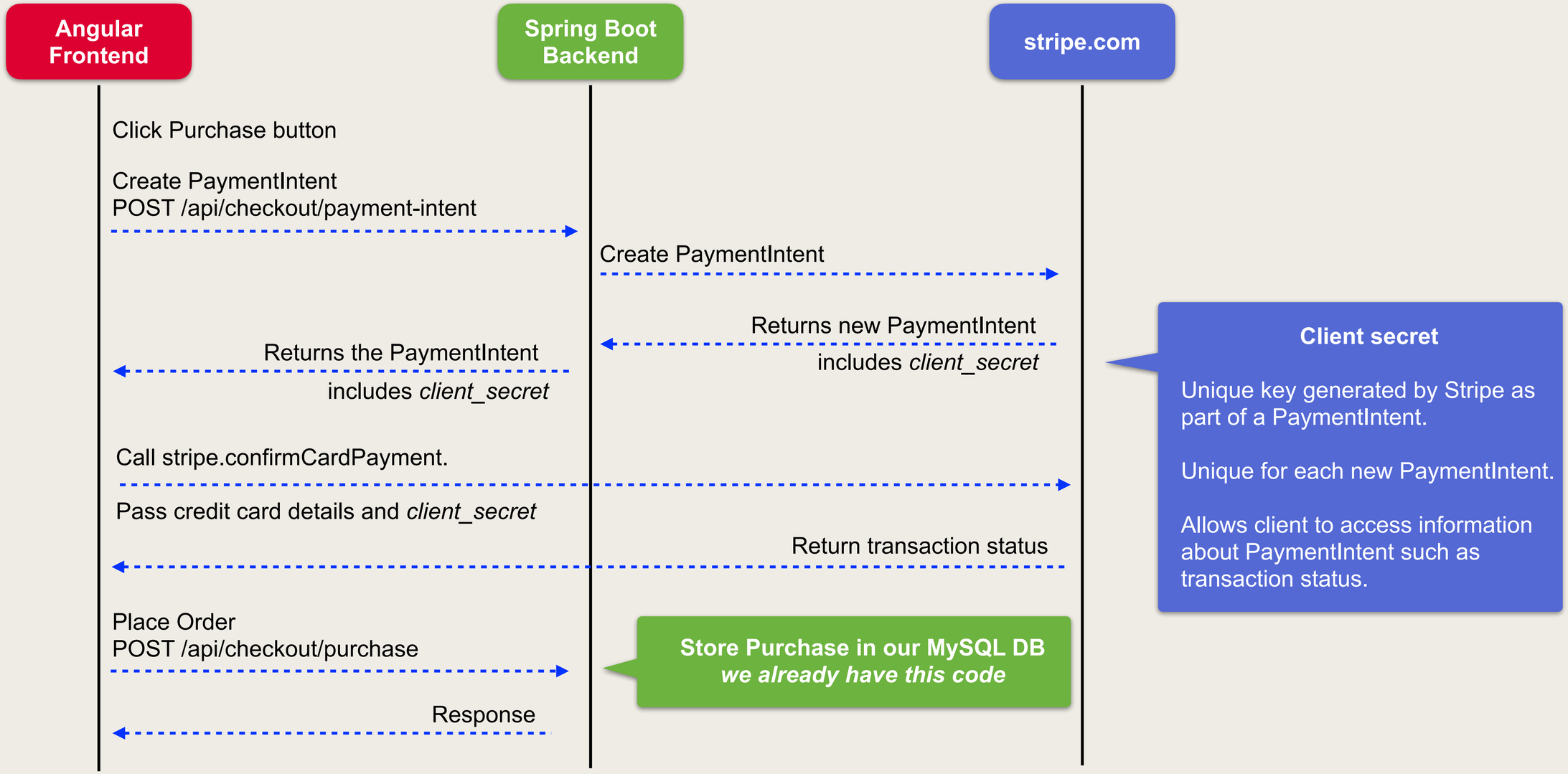
Payment Processing with Stripe

Development Process

Stripe PaymentIntent

- A **PaymentIntent** describes the components of a payment
  - Payment method type
  - Amount
  - Currency

Stripe Payment Processing Flow



Stripe Documentation

www.stripe.com/docs

Development Process Overview

- We will split the process up into backend and frontend

- Start on backend

Development Process - Backend

1. Create Stripe Developer Account
2. Add Stripe Maven Dependency
3. Configure Stripe API Key
4. Create custom PaymentInfo DTO
5. Update CheckoutService to create a PaymentIntent
6. Update CheckoutController to expose /payment-intent endpoint

Step 1: Create Stripe Developer Account

- Create a free account

- Visit: https://dashboard.stripe.com/register

Step 2: Add Stripe Maven Dependency

- Stripe provides a Java library for integration

- Maven dependency

```
<dependency>
  <groupId>com.stripe</groupId>
  <artifactId>stripe-java</artifactId>
  <version>${VERSION-NUMBER}</version>
</dependency>
```

Step 3: Configure Stripe API Key

- Your account has unique API keys
- API keys are available on Stripe Developer Dashboard
- Copy secret key value and place in application.properties

application.properties

```
stripe.key.secret=...
```

Step 4: Create custom PaymentInfo DTO

- We'll create a Data Transfer Object (DTO)
- Contains info regarding payment: amount and currency

PaymentInfo.java

```
package com.luv2code.ecommerce.dto;

import lombok.Data;

@Data
public class PaymentInfo {

    private int amount;
    private String currency;

}
```

Step 5: Update CheckoutService to create a PaymentIntent

CheckoutService.java

```
package com.luv2code.ecommerce.service;

import com.luv2code.ecommerce.dto.PaymentInfo;
import com.luv2code.ecommerce.dto.Purchase;
import com.luv2code.ecommerce.dto.PurchaseResponse;
import com.stripe.exception.StripeException;
import com.stripe.model.PaymentIntent;

public interface CheckoutService {

    PurchaseResponse placeOrder(Purchase purchase);

    PaymentIntent createPaymentIntent(PaymentInfo paymentInfo) throws StripeException;

}
```

From Stripe API

New method

Step 5: Update CheckoutService to create a PaymentIntent

CheckoutServiceImpl.java

```
@Service
public class CheckoutServiceImpl implements CheckoutService {

    private CustomerRepository customerRepository;

    public CheckoutServiceImpl(CustomerRepository customerRepository,
                               @Value("${stripe.key.secret}") String secretKey) {

        this.customerRepository = customerRepository;

        // initialize Stripe API with secret key
        Stripe.apiKey = secretKey;
    }
```

application.properties

```
stripe.key.secret=...
```

Inject secret key from properties file

Initialize Stripe API with secret key

Step 5: Update CheckoutService to create a PaymentIntent

CheckoutServiceImpl.java

```
@Service
public class CheckoutServiceImpl implements CheckoutService {

   ...

   @Override
   public PaymentIntent createPaymentIntent(PaymentInfo paymentInfo) throws StripeException {

       List<String> paymentMethodTypes = new ArrayList<>();
       paymentMethodTypes.add("card");

       Map<String, Object> params = new HashMap<>();
       params.put("amount", paymentInfo.getAmount());
       params.put("currency", paymentInfo.getCurrency());
       params.put("payment_method_types", paymentMethodTypes);

       return PaymentIntent.create(params);
   }

   ...
}
```

Our custom DTO

PaymentIntent is from Stripe API

Step 6: Update CheckoutController to expose /payment-intent endpoint

CheckoutController.java

```
@RestController
@RequestMapping("/api/checkout")
public class CheckoutController {

    private CheckoutService checkoutService;

    ...

    @PostMapping("/payment-intent")
    public ResponseEntity<String> createPaymentIntent(@RequestBody PaymentInfo paymentInfo) throws StripeException {

        PaymentIntent paymentIntent = checkoutService.createPaymentIntent(paymentInfo);
        String paymentStr = paymentIntent.toJson();

        return new ResponseEntity<>(paymentStr, HttpStatus.OK);
    }

}
```

Our custom DTO

PaymentIntent is from Stripe API