Payment System

We design a payment system. E-commerce & other services online have exploded in popularity across the world. What makes every transaction possible is a payment-system running behind the scenes. A reliable, scalable, and flexible payment system is essential.

What is a payment system? According to Wikipedia, “A payment system is any system used to settle financial transactions through the transfer of monetary value. This includes the institutions, instruments, people, rules, procedures, standards, and technologies that make its exchange possible”.

- A **merchant** is any individual or business that sells goods or services and accepts payments from customers using card payments, UPI, net banking, wallets, or other payment methods.
- A **payment system** processes online payments. It enables merchants to accept credit cards and other forms of electronic payments(Netbanking, UPI, Wallets, BNPL-Buy Now Pay Later, etc) securely over the Internet.
- A **payment service provider (PSP)** is a company that offers end-to-end payment processing services to merchants, including payment gateways, merchant accounts, and integration with multiple payment methods (cards, wallets, UPI, etc.) Popular payment service providers include PayPal, Stripe, Square etc.

Handles the entire payment flow (gateway + processing + settlement).

- A **payment gateway** is a technology that facilitates the transfer of payment information between a merchant's website/app and the payment processor. It is responsible for encrypting and securely transmitting transaction details.

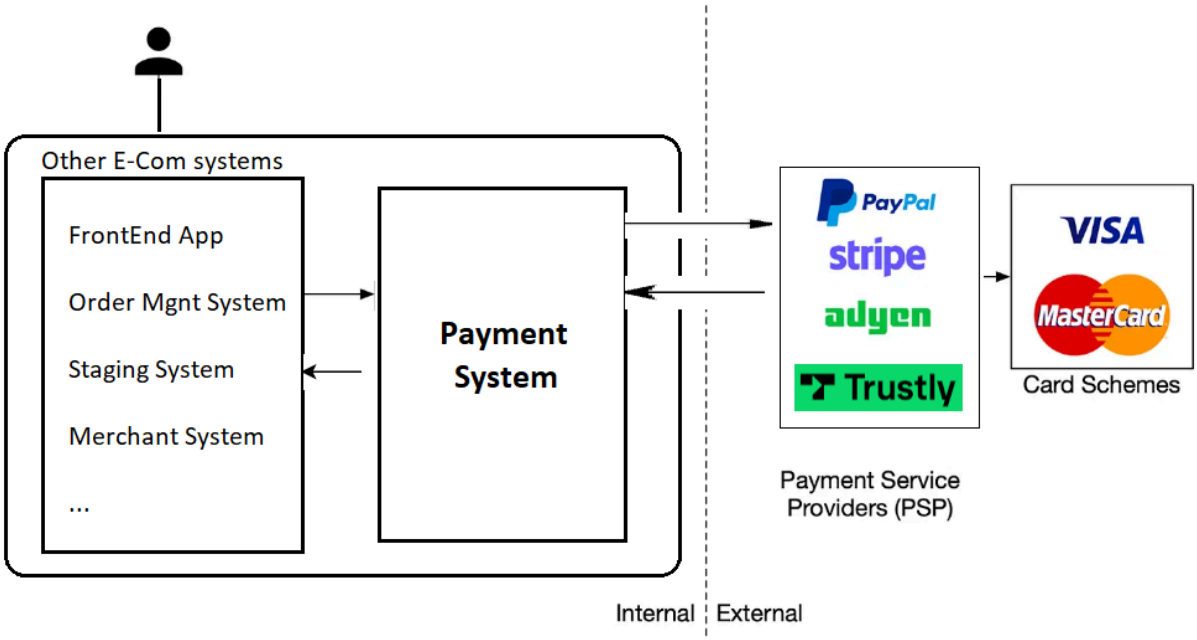
The payment system is easy to understand on the surface but is also intimidating for many developers to work on. A small slip could potentially cause significant revenue loss and destroy credibility among users.

Key Pointers:

- Assume you are building a payment backend for an e-commerce application like Amazon.com. When a customer places an order on Amazon.com, the payment system handles everything related to money movement.
- The payment system should support all of these options in real life. Like Cards or Alternate Payment Methods (APM) - (Netbanking, UPI, Wallets, BNPL-Buy Now Pay Later etc).
- PaymentMethod: Card, APM. When we directly capture & process card details on our website, then we will use paymentMethod: Card. Else we use APM (Including capture card indirectly at PSP)

- For Direct Card Processing, we generally tend to use third-party payment processors (Cardinity, Kalixa, etc). Due to extremely high security and compliance requirements, storing card numbers directly in our system needs Payment Card Industry Data Security Standard (PCI DSS).
- For Alternate Payment Methods (APM) processing, we integrate with the corresponding 3rd party provider. We need to understand end-to-end integration with each of the 3rd party. And build a secure, stable integration and enable the end customer to make payment via that 3rd party provider.
- Load/Scale of around 1 million transactions per day. 10-15 transactions per second.
- PaymentMethod: APM, Card
- Payment Service Provider (PSP): Stripe, Paypal, Trustly, Square etc
- PaymentType: SALE (One time payment), RECC (Subscriptions)
- Hosted Page - the Payment page provided by the PSP. Ecom application redirects the customer to the hosted page, where they complete their transactions. And would return back to the ecom website.
- Reliability and fault tolerance. Failed payments need to be carefully handled.

Pay-in / Deposit / Sale / Checkout flow (Payment System Overview)



Our Project Scope

PSP integration

We are integrating with Stripe PSP. Stripe is a leading payment service provider renowned for its seamless integration and robust infrastructure, offering businesses around the globe a versatile platform to accept online payments securely and efficiently. With its user-friendly interface and extensive range of features, Stripe empowers merchants of all sizes to effortlessly manage transactions, handle subscriptions, and expand their customer base, thereby revolutionizing the e-commerce landscape. Trusted by millions of businesses worldwide, Stripe continues to set the standard for payment processing solutions, driving innovation and simplifying the complexities of online commerce..

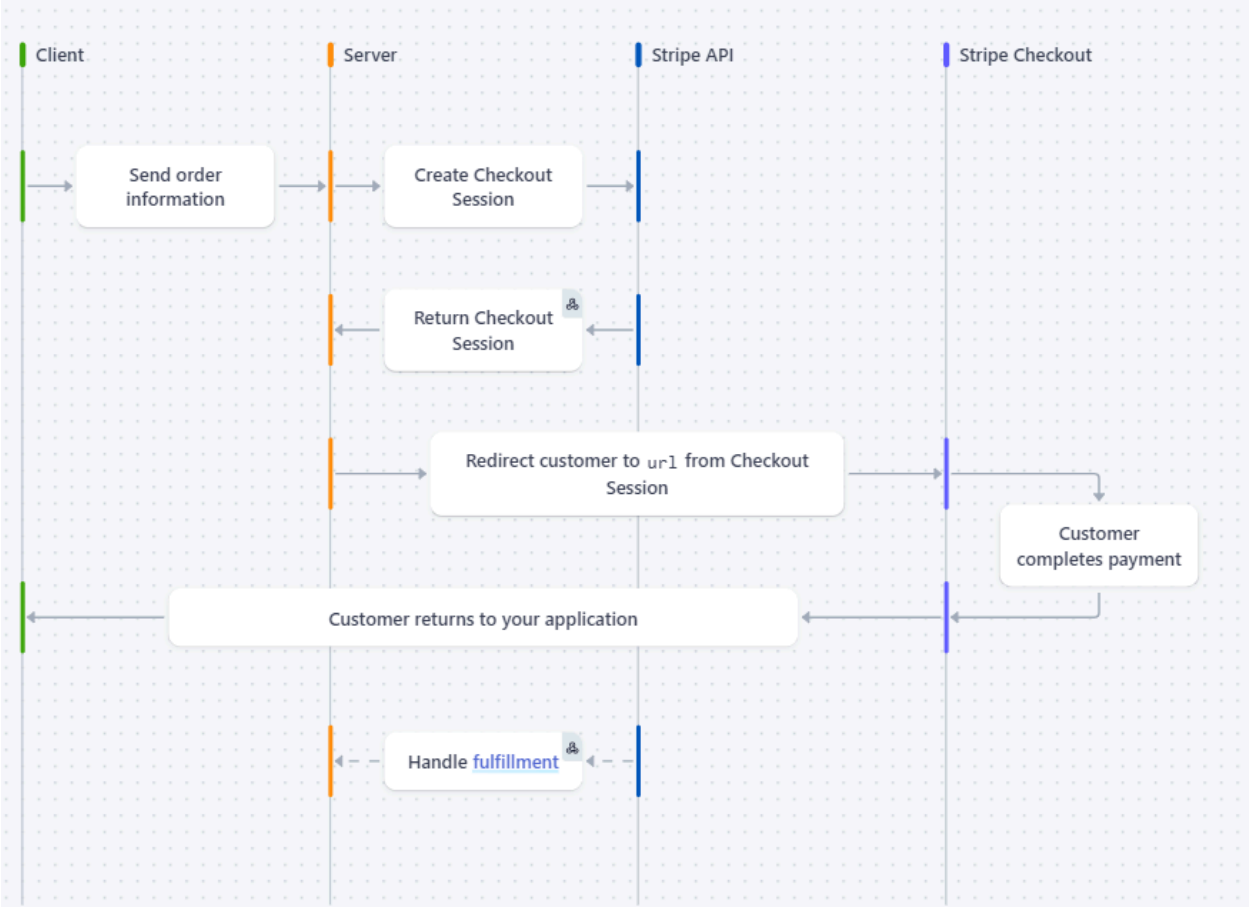
Stripe Payment Integration:

- https://docs.stripe.com/payments/checkout/how-checkout-works
- https://docs.stripe.com/api
- https://docs.stripe.com/api/checkout/sessions

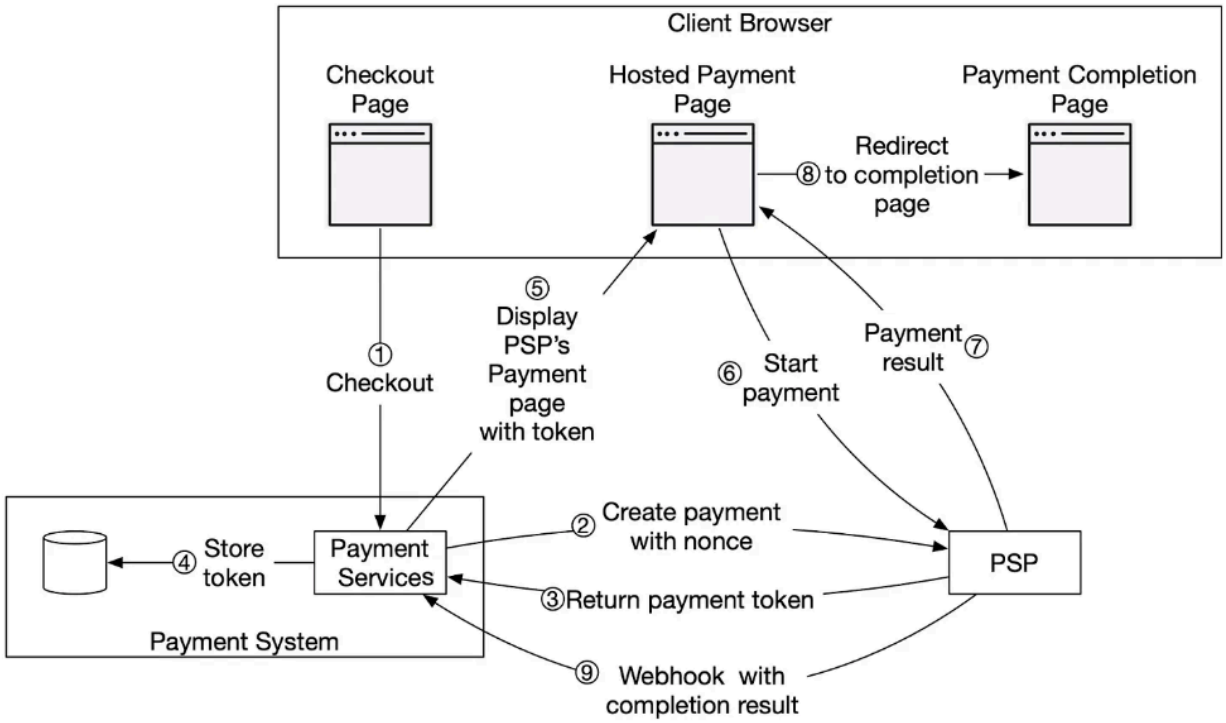
- https://docs.stripe.com/checkout/fulfillment
- https://docs.stripe.com/stripe-cli
- https://docs.stripe.com/testing

Stripe Checkout Flow:

1. When customers are ready to complete their purchase, your application creates a new Checkout Session.
2. The Checkout Session provides a URL that redirects customers to a Stripe-hosted payment page.
3. Customers enter their payment details on the payment page and complete the transaction.
4. After the transaction, a **webhook fulfills the order** using the **checkout.session.completed** API event.



End-to-end payment Flow



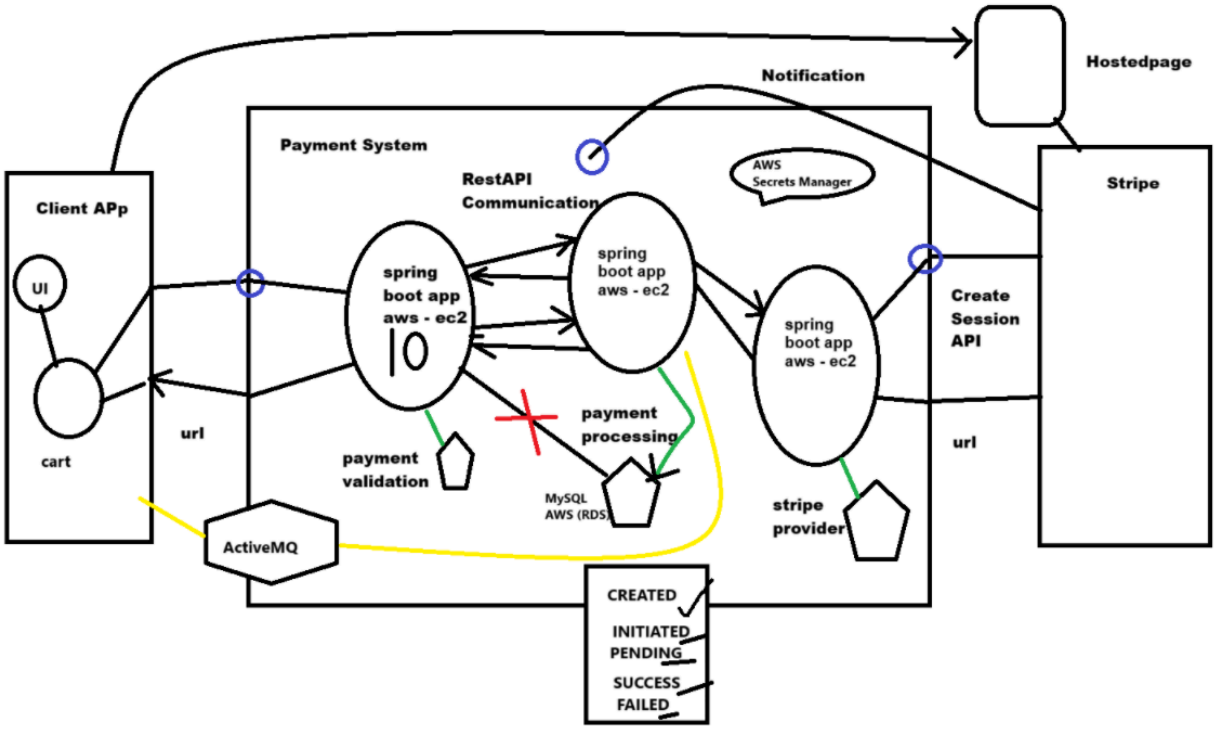
1. The user clicks the “checkout” button in the client browser. The client calls the payment service with the payment order information.
2. After receiving the payment order information, the payment service sends a payment registration request to the PSP. This registration

request contains payment information, such as the amount, currency, etc, and the redirect URL. Because a payment order should be registered only once, there is a unique field(id of payment order) to ensure the exactly-once registration.

3. The PSP returns a token back to the payment service. A token is an id on the PSP side that uniquely identifies the payment registration. We can examine the payment registration and the payment execution status later using this token.
4. The payment service stores the token in the database before calling the PSP-hosted payment page.
5. Once the token is persisted, the client displays a PSP-hosted payment page.
6. The user fills in the payment details on the PSP's web page, logs in to the bank, and authorizes the payment, by clicking the pay button. The PSP starts the payment processing.
7. The PSP returns the payment status.
8. The web page is now redirected to the redirect URL. The payment status that is received in step 7 is typically appended to the URL.
9. Asynchronously, the PSP calls the payment service with the payment status via a webhook. The webhook is an URL on the payment system side that was registered with the PSP during the initial setup with the PSP. When the payment system receives payment events through the webhook, it extracts the payment

status and updates the *payment_order_status* field in the Payment Order database table.

Architecture Diagram:



INFRA DIAGRAM