

Topic: Payment Methods: Stripe & Paypal

Speaker: Personal / Notebook: Health Management System Using Django



This Health System uses the payment methods, Stripe and Paypal.

To test it as a developer in paying via Stripe, use the test card details provided [here](#).

The screenshot shows the Stripe Developer Tools documentation page at <https://docs.stripe.com/testing>. The page has a sidebar with links like 'Get started', 'Payments', 'Finance automation', 'Platforms and marketplaces', 'Banking as a service', 'Developer tools' (which is currently selected), and 'Cards by brand'. The main content area is titled 'Cards by brand' and contains a table of test card details. A red box highlights the table, and another red box highlights the 'Card numbers' tab in the navigation bar.

BRAND	NUMBER	CVC	DATE
Visa	4242 4242 4242 4242	Any 3 digits	Any future date
Visa (debit)	4000 0566 5566 5556	Any 3 digits	Any future date
Mastercard	5555 5555 5555 4444	Any 3 digits	Any future date
Mastercard (2-series)	2223 0031 2200 3222	Any 3 digits	Any future date
Mastercard (debit)	5200 8282 8282 8210	Any 3 digits	Any future date
Mastercard (prepaid)	5105 1051 0510 5100	Any 3 digits	Any future date
American Express	3782 822463 10005	Any 4 digits	Any future date
American Express	3714 496353 98431	Any 4 digits	Any future date
Discover	6011 1111 1111 1117	Any 3 digits	Any future date
Discover	6011 0009 9013 9424	Any 3 digits	Any future date

FRONTEND FOR STRIPE PAYMENT - CHECKOUT.HTML:

```
<script src="https://www.paypal.com/sdk/js?client-id={{paypal_client_id}}&currency=USD"></script>

<script src="https://js.stripe.com/v3/"></script>

<script>
```

```

var stripe = Stripe("{{stripe_public_key}}");

var checkoutButton = document.getElementById("stripe-payment");

try {

  checkoutButton.addEventListener("click", function () {

    var email = "{{billing.patient.email}}";

    checkoutButton.innerHTML = "Processing <i class='fas fa-spinner fa-spin ms-2'></i>";

    fetch("/stripe_payment/{{billing.billing_id}}/", {
      method: "POST",
      body: JSON.stringify({ email: email }),
    })

    .then(function (response) {

      return response.json();
    })

    .then(function (session) {

      return stripe.redirectToCheckout({ sessionId: session.sessionId });
    })

    .then(function (result) {

      if (result.error) {

        alert(result.error.message);
      }
    })

    .catch(function (error) {

      console.log("Error: ", error);
    });
  });

} catch (error) {

  console.log(error);
}

</script>

```

For Paypal, use the Sandbox. If you are testing as the MERCHANT, use the business account. If you are testing as the buyer, use the PERSONAL ACCOUNT. See the documentation [here](#).

The screenshot shows the PayPal Developer Sandbox Accounts page. The left sidebar has a red box around the 'Accounts' link. The main content area has a red box around the 'Current' tab. The 'Overview' section contains the following text:

The PayPal sandbox supports these account types:

Account type	Represents
Personal	The customer in a transaction.
Business	The merchant in a transaction.

To test a typical PayPal transaction, you must use both types of accounts.

When you register as a PayPal developer on the [developer site](#), the PayPal sandbox creates these sandbox accounts.

- A business account and associated API test credentials. For example, `pp.merch01-facilitator@example.com`.
- A default personal account. For example, `pp.merch01-buyer@example.com`.

You can [create additional sandbox accounts](#) on the developer site or directly on the sandbox site, <https://www.sandbox.paypal.com>.

FRONTEND FOR PAYPAL PAYMENT - CHECKOUT.HTML:

```
<script src="https://www.paypal.com/sdk/js?client-id={{paypal_client_id}}&currency=USD"></script>

<script>

function initPayPalButton() {

    paypal

    .Buttons( {

        style: {

            shape: "rect",

            color: "gold",

            layout: "vertical",

            label: "paypal",

        },

        createOrder: function (data, actions) {

            return actions.order.create({

                purchase_units: [{ amount: { currency_code: "USD", value: "{{billing.total}}" } }],


            });

        },


        onApprove: function (data, actions) {

            return actions.order.capture().then(function (orderData) {

                // Full available details

                console.log("Capture result", orderData, JSON.stringify(orderData, null, 2));

                // Show a success message within this page, for example:

                const element = document.getElementById("paypal-button-container");

                element.innerHTML = "";


            });

        }

    })

    .render("#paypal-button-container");
}

initPayPalButton();
```

```
element.innerHTML = "<h5>Verifying payment...</h5>";

window.location.href = `/paypal_payment_verify/{${billing.billing_id}}/?transaction_id=${orderData.id}`;

}),

onError: function (err) {

//console.log(err);

console.error("PayPal Button Error:", err);

alert("An error occurred during the payment process. Please try again.");

}),

})

.render("#paypal-button-container");

}

initPayPalButton();

</script>
```

Copyright © Personal Digital Notebooks | By Rosilie | Date Printed: Aug. 9, 2025, 2:46 p.m.