💸 Payment Intents

Friction Log For One-time Payments

Sabina Robinov - 09/02/2020

Goal:
Successfully Integrate Stripe's PaymentIntents into a website and pass all test cases on Stripe.com, keeping a registry of all the successful payments.

User Story:
As an impatient developer for an eCommerce site that sells donuts, I want to be able to use best practices to quickly and efficiently spike, implement, test and monitor Stripe's PaymentsIntent API integration for one-time payments.

Assumptions:

- The user already has an existing eCommerce site, and needs to implement a new form, incl. backend logic, to integrate with Stripe Payment Intents.
- The focus of the friction report is on the Payment Intents integration.
- The user knows they want to use the PaymentsIntent API for the job, not covering the research they might have to do before deciding on their course of action.

Environment:

- Mac OS & ReactJS.
- Visual Studio Code.
- Google Chrome.

Summary

The overall experience was good, some areas were really delightful like using Stripe Elements and the testing scenarios, while other areas had some room for imporvement. The total time I spent on the exercise was around ~3-4 hours, as a solo developer with no prior experience using Stripe, and I think this already says a lot about the ease-of-use of the PaymentsIntent API. With that said, I was still able to identify a few pain points during the learning experience that solving them could better equip the impatient developer to quickly create high quality code and provide a better user experience.

The friction log is divided into 4 steps: Spike, Implementation, Testing and Monitoring.

Step 1: Spike (1hr effort)

Goal:
Estimate the effort required to complete my task by gathering the information I need to solve it quickly and efficiently, using best practices.

Actions:

1. Reviewing "Accept a payment" docs (link provided in prompt).
2. Reviewing the sequence diagram, the flow was clear but I was missing information on **why** the intent and confirmation had to be processed separately. It seems like the 2 steps can be combined to a single transaction.
3. Reviewing the rest of the page, it wasn't clear how the steps in the sequence diagram relate to the step-by-step instructions.
4. I started with step 1, installing and importing Stripe to my project with `'npm install --save stripe'`.
5. Reviewing step 2, it wasn't clear where I could find the API key. So I skipped this step to create a Stripe account where I saw that there are multiple types of API keys, and it wasn't clear which one I should use. This was the point where I decided to skip the docs altogether and started googling for other tutorials and guides - looking for an example for implementing PaymentIntents from scratch.
6. I searched for `'react stripe elements payment integration example'` and decided to go with the first video result by Leigh Halliday, which was also the shortest video on the list of results (27min, 13min on playback speed 2). In addition, it seemed like a good match because:
    a. The presenter used an official example from Stripe that's available on GitHub.
    b. The presenter used Visual Studio Code which is the editor I was familiar with.
    c. The video covers creating an integration from scratch, without prerequisites / assumptions on my level of knowledge.
    d. The presenter explained the motivation behind each step in the tutorial.

Step 2: Implementation (1hr effort)

Goal:
Follow through with the plan from the spike and prepare the app for testing.

Actions:

1. Based on the video, I created the integration using the form in the tutorial. I found it helpful that:
    a. The presenter showed the Stripe UI and which API key you should use.

    b. I didn't have to go through the API docs and had a ready-made example to work with.
    c. I was able to integrate in under 30min, without any prior knowledge of Stripe.

2. Both the docs and the videos I ran into were covering the "happy path", but in a real world scenario, I'd be looking for something more in-depth for creating a best-in-class experience. I was still missing information on general best practices, for example:
    a. **Security**: safely handling API keys.
    b. **UX**: preventing a user from accidentally submitting the same payment twice.
    c. **Error handling**: what happens when a card is rejected, the user has a typo, or if any of the steps fail.

3. **Side note:** later on, I ran into the The Payment Intents API docs – which were linked from the Stripe API Reference – but those were hard to find and not directly accessible from Accept a payment. The naming conventions were also confusing, it was hard to distinguish between the API reference, the API docs and the API guides.

Step 3: Testing (1 hr effort with redo)

Goal:
Make sure the integration passes all tests, and ready to be deployed to production.

Actions:

1. Going back to Accept a payment, I skipped to step 5 and found it very helpful that I didn't have to write any of the tests on my own! The experience was seamless and straightforward.
2. BUT, "Submitted the payment to Stripe" did not pass in any of the test suites.
3. I noticed the guidelines for how to resolve the issue and those were really helpful in figuring out what went wrong.
4. I realized that the reason the test failed is because the PaymentIntent and confirmCardPayment were not separated (although I could see the transaction completed successfully on the Stripe dashboard).

Step 3.1: Spike + Implementation + Testing Redo

1. At this stage, I went back to research mode to look for more examples on how to integrate. My preferred way of learning is through practical examples (vs. snippets of partial examples), so I went back to Google and looked for another path forward.
2. This time around, I found an official video from Stripe Developers and skimmed through it to see that the intent and the actual payment are implemented separately. Just like the other video, it used a React example and Visual Studio Code which reinforced my decision to follow this example. I wish this video was linked from the docs! Kudos to the

presenter. It also addressed questions I had that weren't covered in the docs and the other video I ran into, like:
    a. UX changes for Stripe Elements.
    b. How to handle API keys.
    c. Touching on best practices (avoiding double payments).
3. I implemented the PaymentIntent integration based on the new example.
4. I re-ran the tests and they all passed! 🎉
5. It would have been helpful to have test cases or recommendations that focus on the user's experience in addition to the provided test scenarios. Another test that Stripe could provide is making sure that the price is protected from being modified in the frontend.

Step 4: Monitoring (10min)

Goal:
Making sure there's a way to monitor and track transactions

Actions:
1. After the tests passed, I went back to Stripe's dashboard to verify that the payments actually went through with the right amount.