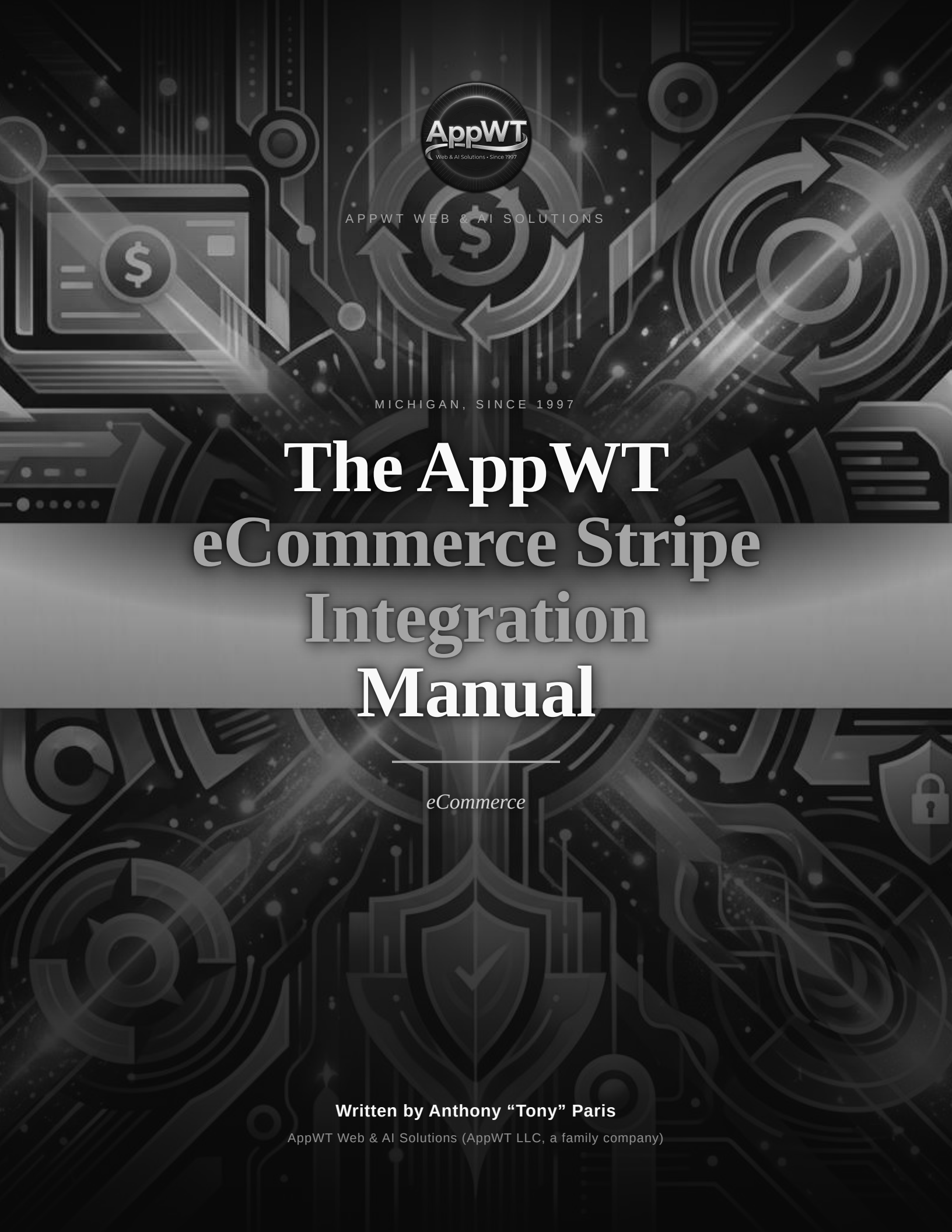

APPWT WEB & AI SOLUTIONS

MICHIGAN, SINCE 1997

The AppWT eCommerce Stripe Integration Manual

eCommerce

Written by Anthony “Tony” Paris

AppWT Web & AI Solutions (AppWT LLC, a family company)

The eCommerce Stripe Integration Manual

eCommerce

Written by Anthony “Tony” Paris

Published by AppWT Web & AI Solutions

AppWT LLC, a family company

The eCommerce Stripe Integration Manual

eCommerce

Written by Anthony “Tony” Paris

AppWT Web & AI Solutions (AppWT LLC, a family company)

Second Edition, August 2026

Published in Livonia, Michigan, United States of America.

A NOTE ON SPELLING

This manual is read in the United States, Canada, the United Kingdom and Australia. Where a word is spelled differently in different countries, both spellings appear the first time the word is used, like color/colour. After that, one spelling is used so the page stays easy to read.

HOW TO USE THE TECHNICAL NOTES

Smaller italic notes marked *For your tech person* hold the deeper detail: exact settings, file names and commands. You can skip every one of them and the manual still makes complete sense. Hand them to whoever looks after your website.

IMPORTANT

This manual describes security practices that AppWT uses and recommends. It is written as general guidance for business owners. It is not legal advice, and it is not a guarantee that any system cannot be attacked. Security reduces risk. No honest provider will promise to remove risk entirely.

Every incident described in this manual comes from work AppWT actually performed. Client identities are deliberately withheld, because a client's security incident is not ours to publish. Where the business involved is AppWT itself, we name ourselves.

FOREWORD

A Note From Tony

In 1997 we took card numbers in a form we wrote ourselves, mailed the order to the shop, and kept a paper log. It worked, and it was a bad idea. Almost thirty years later the wiring is far better, and the way a store loses money is far quieter.

Payments get sold to business owners as a switch. Install the plugin, paste two keys, take money. That is honestly true for a while. It stays true until the first quiet failure, and then it is not true at all.

Quiet is the word that matters. Nearly every payment problem I get called about makes no sound. Nothing turns red. No error page appears. The store keeps taking orders and looking healthy while one step in the middle has stopped running. By the time a human notices, the evidence is a week old and the customer has already written a review.

This book exists because that middle step has never been explained to the person who owns the store. Developers know it. Support agents know it. The owner gets a dashboard, a plugin settings screen, and no map. So the first chapter draws the map, and every chapter after it fixes one named place on it.

Here is what I promise. By the end you will be able to point at any payment problem and name the stage it lives at. You will have a checkout you have tested on real phones, a listener you trust to record sales, an order record that cannot be faked by a browser, and a month-end sheet with four numbers that must agree. Every one of those is something you can check yourself.

Here is what I will not promise. I cannot promise you more sales, because I have never seen your product or your prices. I am not your accountant and I am not your attorney. Tax registration, subscription cancellation law and card network rules all change by where you sell and what you sell. I tell you which questions to put to a qualified adviser, and I tell you to write the answer down with a date. I also will not call any platform or provider bad. You get facts and the judgment stays yours.

Work the book in order the first time, even if your store already takes money. Part One asks you to set up a sandbox and run test cards before you touch anything live. People skip that and then debug an authorization/authorisation problem on real buyers at nine on a Friday night. Do it the calm way instead. One change, one test, one dated line in a plain text file kept with the site.

This is a second edition, written for 2026. That matters more here than in most subjects. Some of the best known sample code online still teaches methods that have been retired, and it looks perfectly reasonable on the page. Where a practice is retired I say so plainly and say what replaced it. Where something is genuinely unsettled, and a few things are, I label it as emerging rather than telling you it is decided.

One last point, and it is the one I care about most. The payment account belongs in your business name, with you as the owner, whoever builds the integration. The keys are yours. The bank connection is yours. Every page here is written so you can do the work yourself, or hand a precise list to somebody else and still check the result.

Anthony “Tony” Paris

Founder, AppWT Web & AI Solutions · Livonia, Michigan

BEFORE YOU BEGIN

How to Use This Manual

This is a working manual for a store that already takes money, or is about to. Read it with your payment dashboard open on one screen and a notepad beside you. Almost every chapter ends with something you can set up, test or verify the same day.

The parts

Part One shows you how the money actually moves. Chapter 1 draws the full path from the card field to the line on your bank statement and names the nine stages. Chapter 2 names the records a 2026 integration uses and says which older ones are retired. Chapter 3 compares the four ways to take a card. Chapter 4 covers keys, sandboxes and a test run you can repeat before every release. Read these four in order even if your store is live.

Part Two is about buyers who wanted to pay and could not. These failures produce no error in your log and no email in your inbox, so they can run for weeks. Four chapters cover the security header that leaves a blank box where the card fields belong, the wallet buttons that appear for you and not for your customer, the decline codes and the authentication step, and the autofill and accessibility problems that stop a willing buyer before the card field.

Part Three turns a payment into an order. Chapter 9 builds a listener that verifies what it receives, answers fast, and refuses to do the same work twice. Chapter 10 sets the rule the rest of the book rests on, which is that your server decides a sale is real and the browser never does. Chapter 11 explains why your order report and your bank deposit will never show the same number, then gives you the month-end match.

Part Four is for recurring revenue. It covers prices you can raise without touching existing subscribers, usage metering the current way, failed renewals and the settings that recover them, and cancellation under the rules as they stand in 2026. If you sell one-time orders only, skip this part and come back the week somebody suggests a monthly plan.

Part Five is defense and rhythm. Chapter 16 covers card testing, which is fraud that arrives as a flood of tiny attempts on your public payment form. Chapter 17 covers disputes, the evidence that wins them, and the network monitoring programs. Chapter 18 gives you a quarterly review sheet and an honest account of what is still unsettled.

The boxes, and what each one means

> DO THIS TODAY
>
> A gold box is one action you can finish now. If you only ever act on the gold boxes, you will still be far safer than when you started.

FROM OUR FILES

A dark box is a real incident from AppWT work. Nothing in these boxes is invented. Where a detail was never verified, we say so rather than fill the gap.

FOR YOUR TECH PERSON

A smaller italic note like this one carries the professional detail: exact file names, headers, settings and commands. Skip every one and the manual still reads completely. Forward them to your web person.

The code blocks and the technical notes

There is very little code in this book, and none of it is written for one language. The blocks are short and plain, so a developer can read them in whatever they already use. Every line is explained in ordinary words underneath. The technical notes are aimed at whoever holds server access, and that may not be you. Skipping one never breaks a chapter. Hand it over instead, then check the result yourself using the test printed beside it.

What this book does not promise

No revenue figure is promised, because I have never seen your product or your traffic. This is not tax advice, legal advice or accounting advice, and the chapters that touch those subjects say so on the page and tell you to confirm your own position. This book does not cover paying out to third-party sellers, in-person card readers, or building your own card form, which is a trade with its own audits and costs. It also does not replace your provider's own documentation, which changes more often than any printed book can.

If you only have one afternoon

Do four things, in this order. Open your event delivery screen and look for failed deliveries in the last month. Confirm your store creates the order from a verified event and not from the thank-you page. Search your page source and your code repository for a secret key, and replace it today if you find one. Then buy something from your own store on a phone you do not normally test with, and refund it by hand. Those four checks find most of what is quietly broken.

Practice in a sandbox, never on a customer

Every procedure in this book can be rehearsed without a real card. A sandbox is a separate copy of your payment account with its own keys and its own data. Test cards let you force a decline, an authentication challenge and a dispute whenever you want to see one. Rehearse there until the steps are dull, then do the live version once and refund it. A probe that cannot fail proves nothing, so make each test actually fail first.

Contents

PART ONE

How the Money Actually Moves

A card payment and a store order are two records held in two systems. These four chapters draw the whole path from the card field to the bank statement, name the objects Stripe uses in 2026, pick a checkout surface on purpose, and set up keys, sandboxes and a repeatable test run before anything reaches a real buyer.

CHAPTER 1 PART ONE

The Payment Went Through and the Order Never Appeared

Chapter 1. Why a card payment and a store order are two different records, and the nine stages between them

A successful payment in Stripe and a missing order in your store are not a contradiction. They are two records held by two systems that only agree when something keeps them in step. This chapter draws the whole path a payment takes, names the nine stages, and shows you the five places it commonly breaks so you can find your own problem on the map.

It is a little after seven on a Tuesday. The owner of a small online store sits down with coffee and opens three windows. The first is the store's order list. The second is the Stripe dashboard. The third is the shop email inbox. On most mornings those three windows tell the same story. Orders came in overnight, payments came in beside them, and the day starts with packing. This morning they do not agree, and the disagreement is the kind that makes a person feel slightly ill.

The inbox holds a polite message from a customer. She paid last night. She has the receipt email open in front of her. She would like to know when her package ships. The Stripe dashboard agrees with her completely. There is the payment, marked Succeeded, timed just after nine the previous evening. The card brand is listed. The last four digits are listed. The money is real and it is sitting in the balance. The store's order list, though, shows nothing for that customer. No pending order. No paid order. No sign she was ever in the shop.

The first instinct is to refresh the page, then to search the order list by email, then to search by name, then to decide the store software is broken. None of that helps, because nothing is broken in the way the owner is imagining. The two screens are not two views of one thing. They are two separate records, written by two separate systems, at two separate moments. Most of the time they match. When they stop matching, the reason is almost always that a single step between them did not run.

Two Records, Two Systems

Here is the plain version. Stripe keeps a record of money. Your store keeps a record of business. Stripe's record says a card was charged and the funds were collected. Your store's record says a person bought two items, chose shipping, and is waiting for a box. Neither record can create the other on its own. Something has to carry the news from one system to the other, and that something is a message sent over the internet from Stripe to your server. If the message does not arrive, or arrives and is thrown away, Stripe stays right and your store stays empty. Both systems are working correctly. They are simply no longer talking.

Six words do most of the work in this book. It is worth spending sixty seconds on them now, because every later chapter assumes them.

- **Processor.** The company that moves a payment request between your store, the card networks and the banks.
- **Gateway.** The software layer that collects card details safely from your page and hands them to the processor.
- **PaymentIntent.** Stripe's record of one attempt to collect one payment, tracked from the first click to the final answer.
- **Client secret.** A short one-time string that lets a single browser session work on one PaymentIntent and nothing else.
- **Webhook.** A message Stripe posts to a web address you own whenever something happens to a payment.
- **Order record.** The row in your own database that says what was bought, by whom, and what stage it has reached.

Stripe is both the gateway and the processor in almost every small store setup, which is why the two words get blurred in ordinary conversation. That blurring is harmless. The dangerous blur is the one between the PaymentIntent and the order record. They are not the same object, they do not live in the same place, and they do not update each other automatically. They synchronize/synchronise only because a webhook fires and your server acts on it. Remove that one link and you get exactly the morning described above, with a happy customer, a real payment, and an order list that has never heard of her.

The Path One Payment Takes

Nine things happen between a buyer typing a card number and a deposit landing in your bank account. They always happen in the same order. Some take a fraction of a second. One of them takes days. Most owners have never seen the list written down, which is why a payment problem feels like fog rather than like a single broken step. Read the nine slowly. You are going to use them for the rest of the book as addresses, the way you would use street numbers.

1. **Card details are collected.** The buyer types a card number into a field on your checkout page. That field is usually not yours. It is a small frame served by Stripe and dropped into your page, so the raw digits never touch your server or your database.
2. **A payment method is created.** Stripe converts those digits into a PaymentMethod object and hands your page a reference to it. From this point on, your code passes a harmless identifier around instead of a card number.
3. **Authorization/authorisation is requested.** Stripe sends the request through the card network to the bank that issued the card. The bank checks the balance, its own fraud rules and sometimes the billing address, then answers yes or no. If the bank wants the buyer to prove who they are, it asks for that here.
4. **Funds are captured.** In a normal store setup, capture happens in the same moment as the approval. The money is now committed, the buyer's available balance drops, and the receipt goes out. This is the moment that puts Succeeded on the dashboard.
5. **An event is sent to your server.** Stripe writes a small message describing what just happened and posts it to a web address you control. This message is the webhook. It is the only routine way your store finds out that money arrived.
6. **The order record is written.** Your server reads the message, checks that it genuinely came from Stripe, and creates or updates the order in your own database. Until this step runs, your store does not believe a sale happened, no matter what the dashboard says.
7. **Goods are sent.** Someone picks and packs a box, or a download link goes out, or a booking is confirmed. This is fulfillment/fulfilment. Stripe knows nothing about it unless you choose to write it back.
8. **A payout is made.** Stripe gathers up settled payments, subtracts its fees and any refunds, and sends the remaining balance to your bank on whatever schedule your account uses.
9. **A line appears on your bank statement.** Days after the sale, one deposit shows up covering many orders at once. It will not match any single order total, and it is not supposed to.

FOR YOUR TECH PERSON

The client secret in stage one is designed to be visible in the browser. It grants power over exactly one PaymentIntent and expires with it. Your secret key is the opposite and must never appear in page source, in front-end JavaScript, or in a public code repository. Chapter 4 covers the difference and the procedure when a key leaks.

Notice how ownership changes along the way. Stages one through five happen mostly inside Stripe and inside the buyer's browser. You configure them, but you do not run them. Stages six and seven happen entirely inside your own systems, and Stripe has no visibility into either one. Stages eight and nine happen in banking, where neither you nor Stripe controls the timing. A great deal of confusion comes from asking one party for evidence about a stage they never saw. Stripe cannot tell you whether a box shipped. Your store cannot tell you why an issuing bank said no.

What Hides at Each Stage

STAGE	WHAT STRIPE CALLS IT	THE FAILURE THAT HIDES HERE
1. Card details collected	Payment Element, or a Checkout Session	The card field never renders and the buyer stares at a blank rectangle.
2. Payment method created	PaymentMethod	A wallet button is missing on phones, so the buyer gives up rather than type.
3. Bank asked for approval	PaymentIntent, status requires_action or processing	The bank declines, or the buyer abandons the extra security step half way.
4. Funds captured	Charge, PaymentIntent status succeeded	An approval is held but never captured, and it quietly expires days later.
5. Event sent to your server	Event, delivered to an endpoint	Delivery fails, or your server rejects the message because the signature check went wrong.
6. Order written	Nothing. This record is yours alone.	The order is created from the thank-you page instead of from the event, so a closed laptop erases it.
7. Goods sent	Nothing. Stripe never sees this.	A slow payment method is treated as paid, and goods ship before the money clears.
8. Payout made	Payout and BalanceTransaction	Fees and refunds are never booked, so the deposit looks wrong every single month.

STAGE	WHAT STRIPE CALLS IT	THE FAILURE THAT HIDES HERE
9. Bank statement line	Nothing. Your bank names this one.	The statement wording is unrecognizable to the buyer, who files a dispute instead of an email.

Nine stages, and in practice the path breaks at five of them far more than at the rest. That is not a guess about your store. It is the shape of the work. The five break points are where two different parties have to agree about something, and agreement is always the fragile part. The rest of this book is organized around those five, in the order a payment meets them.

1. **The buyer never gets a working payment form.** Stages one and two. The card fields do not appear, the wallet button is missing, or the form cannot be completed on a phone or with a keyboard. Nothing lands in your logs because nothing was ever attempted. Part Two covers this, in Chapters 5, 6 and 8.
2. **The bank says no, or the security step is never finished.** Stage three. A decline code arrives and nobody in the business knows whether to retry it, or a buyer closes the tab in the middle of an authentication challenge. Chapter 7 covers reading decline codes and handling the paused state.
3. **The event never lands, or lands and gets discarded.** Stage five. Your endpoint is unreachable, slow, or rejecting messages it should accept. This is the failure in the opening story, and it is the single most common cause of a payment with no order. Chapter 9 covers it.
4. **The order is written from the wrong signal, or never written at all.** Stages six and seven. The store trusts the browser instead of the server, or ships goods on a payment method that has not actually cleared. Chapter 10 covers the order states and which event to act on.
5. **The money in the bank does not match what the store thinks it sold.** Stages eight and nine. Fees, refunds, tax and payout timing all pull the deposit away from the order total. Chapter 11 covers the month-end match that ties the two back together.

FOR YOUR TECH PERSON

Stripe keeps a log of every event it generated and every attempt it made to deliver that event to you, including the response code your server returned each time. You can also resend a past event by hand. That log is the fastest way to test stage five without writing any code, and it turns most stage five mysteries into a two-minute answer.

FIND THE STAGE BEFORE YOU CHANGE ANYTHING

When a payment problem appears, resist the urge to edit code. Ask three questions in order. Did Stripe record a payment attempt at all? If no, the problem is at stage one, two or three. If yes, did the event reach your server and get a success response? If no, the problem is at stage five. If yes, does your own log show the order write running? If no, the problem is at stage six. Three questions place almost every payment problem at a single stage, and a problem with an address is a problem you can hand to a developer in one sentence.

THE TUESDAY MORNING, RESOLVED

The owner in the opening had moved her store to a new host over the weekend. Everything looked fine on Monday because Monday's orders came from a batch she had already packed. When she opened the payment in the Stripe dashboard and looked at its events, the picture was immediate. The event had been generated on time. Stripe had tried to deliver it, received a not-found response, waited, and tried again several times over the following hours. The web address for the endpoint had changed with the move, and nobody had updated it in the dashboard. She corrected the address, resent the stored events, and the missing orders appeared in the store within a minute. Then she did the part that mattered more. She set up an alert that emails her when webhook delivery starts failing, so the next time this happens she learns about it in minutes instead of learning about it from a customer. Total repair time was under an hour. Total time spent believing her store software was broken was two days.

FOR YOUR TECH PERSON

Stages six and seven exist only inside your own systems, so no amount of Stripe configuration can prove they ran. Your application log is the only evidence that an order write was attempted, and your shipping system is the only evidence that a box left the building. If those logs do not exist yet, create them before you need them.

That is the whole map. One payment, nine stages, five common break points, and a chapter waiting at each one. Before you read on, take five minutes and write your own current problem, or your most recent one, next to a stage number. If you cannot decide between two stages, use the three questions in the callout above and the answer will arrive quickly. Owners who can name the stage get help fast, because a developer can act on "events are not reaching our endpoint" and cannot act on "checkout is broken." The rest of this book assumes you are holding that map. Part One finishes the groundwork by naming the objects Stripe actually uses in 2026, choosing a checkout surface on purpose, and setting up keys and a test run so mistakes stay cheap.

WHAT TO REMEMBER

- A card payment and a store order are two separate records in two separate systems, and they agree only because a webhook keeps them in step.
- Every sale travels the same nine stages, from card field to bank statement line, and each stage has a distinct failure that hides inside it.
- Stages one through five belong mostly to Stripe and the browser, stages six and seven belong entirely to you, and stages eight and nine belong to banking.
- A payment marked Succeeded with no matching order is almost always a stage five problem, meaning the event did not reach your server or was rejected there.
- Three questions place almost any payment problem at a single stage: did Stripe record an attempt, did the event get a success response, and did your order write run.
- Name the stage before you change any code, because a problem with an address gets fixed in an hour and a vague one takes days.