

Stripe Checkout Integration

This document explains the Stripe checkout integration implemented on the VerifyLens landing page.

Overview

The landing page now supports Stripe Checkout for purchasing credit packages. When a user clicks “Purchase Package”, they are redirected to Stripe’s hosted checkout page where they can securely enter their payment information.

Architecture

1. Frontend Integration (`app/credits/page.tsx`)

- **Stripe.js:** Initialized with the publishable key from environment variables
- **Purchase Flow:**
 1. User clicks “Purchase Package” button
 2. Frontend calls `/api/checkout` API endpoint with package details
 3. API returns a Stripe Checkout Session URL
 4. User is redirected to Stripe’s hosted checkout page
- **Loading States:** Shows a loading spinner on the clicked button while processing
- **Cancel Handling:** Shows a toast notification if user cancels checkout

2. API Endpoint (`app/api/checkout/route.ts`)

- **Purpose:** Creates Stripe Checkout Sessions
- **Input Parameters:**
 - `packageName` : Name of the credit package (e.g., “Starter”, “Professional”)
 - `credits` : Number of credits in the package
 - `price` : Price in cents (e.g., 1000 = \$10.00)
 - `email` : Customer email (optional, can be collected by Stripe)
 - `customerId` : Customer ID from the main app database (optional)
 - `customerName` : Customer name (optional)
- **Metadata:** The checkout session includes metadata that the webhook needs:
 - `packageName`
 - `credits`
 - `price`
 - `customerId`
 - `customerEmail`
 - `customerName`
 - `source` : Set to “landing_page” to identify purchases from the landing page

3. Success Page (`app/credits/success/page.tsx`)

- Displayed after successful payment
- Shows confirmation message and session ID

- Provides links to:
- Main VerifyLens app (www.verifylens.com)
- Landing page home
- Contact support

4. Cancel Handling

- Users who cancel are redirected back to `/credits?canceled=true`
- A toast notification is shown informing them they can try again

Environment Variables

The following environment variables must be set in `.env` :

```
# Stripe Configuration
NEXT_PUBLIC_STRIPE_PUBLISHABLE_KEY="pk_test_..."
STRIPE_SECRET_KEY="sk_test_..."
STRIPE_WEBHOOK_SECRET="whsec_..."
WEBHOOK_ENDPOINT="https://www.verifylens.com/api/credits/webhook"
```

Webhook Integration

The main app (www.verifylens.com) handles the Stripe webhook at `/api/credits/webhook` . When a payment is successful:

1. Stripe sends a `checkout.session.completed` event to the webhook
2. The webhook extracts the metadata (customer info, credits, etc.)
3. The webhook adds the credits to the customer's account in the database
4. The customer can then use the credits in the main app

Testing

Test Cards

Stripe provides test cards for testing the integration:

- **Success:** 4242 4242 4242 4242
- **Decline:** 4000 0000 0000 0002
- **Requires Authentication:** 4000 0025 0000 3155

Use any future expiration date and any 3-digit CVC.

Testing Flow

1. Navigate to the credits page: `http://localhost:3000/credits`
2. Click "Purchase Package" on any package
3. Enter test card information on Stripe checkout
4. Complete the payment
5. Verify redirect to success page
6. Check that the webhook receives the event (on main app)

Security

- **PCI Compliance:** Stripe handles all payment card data, so the landing page never touches sensitive information
- **HTTPS:** Stripe requires HTTPS for production webhooks
- **Environment Variables:** Sensitive keys are stored in environment variables, never in code
- **Webhook Verification:** The webhook secret ensures that webhook events are authentic

Production Checklist

Before deploying to production:

1. ☐ Replace test API keys with live keys
2. ☐ Update `WEBHOOK_ENDPOINT` to production URL
3. ☐ Configure Stripe webhook endpoint in Stripe Dashboard
4. ☐ Test with real cards in test mode
5. ☐ Verify webhook is receiving events
6. ☐ Monitor Stripe logs for errors
7. ☐ Set up error alerting

Troubleshooting

Checkout Session Creation Fails

- Check that `STRIPE_SECRET_KEY` is set correctly
- Verify the API endpoint is accessible
- Check browser console for errors

Redirect Fails After Payment

- Verify `success_url` and `cancel_url` are correct
- Check that the domain is added to Stripe's allowed domains

Webhook Not Receiving Events

- Verify webhook URL is accessible from Stripe
- Check that `STRIPE_WEBHOOK_SECRET` matches Stripe Dashboard
- Check webhook logs in Stripe Dashboard

Future Enhancements

- Add customer portal for managing subscriptions
- Implement promo codes/coupons
- Add volume discounts for enterprise packages
- Support for ACH/bank transfers
- International payment methods