Stripe Connect Express: Embedded Onboarding Analysis

Question

Can Stripe Connect Express account onboarding be done entirely “inside the app” (embedded/API-driven)?

Answer: Partially Yes, with Important Limitations

TL;DR

Stripe Connect Express accounts **cannot** be fully onboarded with a completely custom, API-driven UI. However, Stripe provides **embedded onboarding** options that allow the onboarding process to happen inside your app using Stripe’s hosted components, significantly improving UX compared to full redirects.

Onboarding Options for Stripe Connect Express

1. Embedded Components (Recommended - Best UX)

Status: ✅ Available for Express accounts

Stripe provides **Connect Embedded Components** that allow you to embed the onboarding flow directly in your app without full-page redirects.

How It Works:

- Use the **Stripe Connect JS SDK** to embed onboarding UI components
- The components are hosted by Stripe but rendered inside your app
- Users never leave your application during onboarding
- Maintains your app’s look and feel around the embedded components

Implementation:

```
// 1. Create Account Session (backend)
const accountSession = await stripe.accountSessions.create({
  account: connectedAccountId,
  components: {
    account_onboarding: { enabled: true },
  },
});

// 2. Initialize Stripe Connect JS (frontend)
import { loadConnectAndInitialize } from '@stripe/connect-js';

const stripeConnectInstance = loadConnectAndInitialize({
  publishableKey: 'pk_...',
  fetchClientSecret: async () => accountSession.client_secret,
});

// 3. Render embedded component
<ConnectAccountOnboarding
  stripeConnectInstance={stripeConnectInstance}
  onExit={() => {
    // Handle completion
  }}
/>
```

Pros:

- ✅ Best user experience - no full-page redirects
- ✅ Users stay in your app throughout
- ✅ Stripe handles all compliance, KYC, and verification
- ✅ Responsive and mobile-friendly out of the box
- ✅ Automatic security and PCI compliance

Cons:

- ⚠️ Still uses Stripe-hosted UI (not fully custom)
- ⚠️ Limited styling customization (brand colors only)
- ⚠️ Requires Stripe Connect JS SDK

---

2. Account Links (Current Implementation)

Status: ✅ Currently used in this app

How It Works:

- Create an `AccountLink` via Stripe API
- Redirect user to Stripe-hosted onboarding page
- User completes onboarding on `connect.stripe.com`
- Redirected back to your app via `return_url`

Implementation (Current):

```
const accountLink = await stripe.accountLinks.create({
  account: stripeAccountId,
  refresh_url: `${appUrl}/dashboard/creator?onboarding=refresh`,
  return_url: `${appUrl}/dashboard/creator?onboarding=success`,
  type: 'account_onboarding',
});

// Redirect user
window.location.href = accountLink.url;
```

Pros:

- ✅ Simple to implement
- ✅ Fully compliant with Stripe requirements
- ✅ No custom code needed for complex forms

Cons:

- ❌ Full-page redirect away from your app
- ❌ Breaks user flow
- ❌ Less control over UX

---

3. Connect Embedded Components - Account Management

Status: ✅ Available (for post-onboarding management)

After onboarding, you can use embedded components for:
- Updating bank account information
- Viewing payout history
- Managing tax forms
- Updating business details

Implementation:

```
// Backend: Create account session for account management
const accountSession = await stripe.accountSessions.create({
  account: connectedAccountId,
  components: {
    account_management: { enabled: true },
    payouts: { enabled: true },
  },
});

// Frontend: Render account management component
<ConnectAccountManagement
  stripeConnectInstance={stripeConnectInstance}
/>
```

Fully Custom API-Driven Onboarding

Can You Build Your Own Custom Onboarding UI?

Answer: ❌ No, not for Express accounts

Stripe **does not** allow fully custom, API-driven onboarding for Express accounts for the following reasons:

Why Not?

1. **Compliance Requirements:**
   - Stripe must ensure KYC (Know Your Customer) compliance
   - Identity verification must be done through Stripe's verified processes
   - PCI DSS compliance for handling sensitive data

2. **Legal Requirements:**
   - Stripe is legally responsible for verifying account holders
   - Cannot delegate this responsibility to third parties
   - Must collect specific information in specific ways

3. **Security:**
   - Protects against fraud
   - Ensures data is collected securely
   - Prevents misuse of sensitive financial information

What About Standard and Custom Accounts?

- **Standard Accounts:** Users sign up directly with Stripe (not applicable here)
- **Custom Accounts:** ✅ You can build custom onboarding UI, but:
- Requires significant compliance overhead
- You're responsible for collecting and verifying all information
- More complex to implement and maintain
- Not recommended unless you need full control

Recommendation for This Platform

Best Approach: Embedded Components

For "Call a Star", we recommend using **Stripe Connect Embedded Components**:

Implementation Plan:

Phase 1: Account Creation (Immediate)
- Create Stripe Express account when user becomes a creator
- Store `stripeAccountId` in database
- Set account status to "pending onboarding"

Phase 2: Embedded Onboarding
- When creator clicks "Setup Payments", show embedded onboarding component
- Component loads inside your app (no redirect)
- User completes onboarding seamlessly
- Webhook `account.updated` notifies when complete

Phase 3: Account Management
- After onboarding, show embedded account management component
- Allow creators to update bank details, view payouts, etc.
- All within your app's UI

When to Create the Stripe Account?

Option A: Immediately (Recommended ✅)

Create account when user becomes a creator

```
// When user is promoted to CREATOR role
const account = await stripe.accounts.create({
  type: 'express',
  capabilities: {
    card_payments: { requested: true },
    transfers: { requested: true },
  },
  metadata: {
    creatorId: creator.id,
    userId: user.id,
  },
});

await prisma.creator.update({
  where: { id: creator.id },
  data: { stripeAccountId: account.id },
});
```

Pros:
- ✅ Account ID available immediately
- ✅ Can start tracking account status
- ✅ Simpler logic (one account = one creator)
- ✅ Creator dashboard can show "Complete Setup" immediately

Cons:
- ⚠️ Creates Stripe accounts for users who may not complete onboarding
- ⚠️ Slight increase in Stripe account count

Option B: After Profile Completion

Create account only after creator completes profile

Pros:
- ✅ Fewer inactive Stripe accounts
- ✅ Only create accounts for "serious" creators

Cons:
- ❌ More complex state management
- ❌ Need to handle "profile complete but no Stripe account" state
- ❌ Harder to track onboarding progress

Verdict: Option A (Immediate Creation) ✅

Create the Stripe account immediately when a user becomes a creator. This provides:
- Simpler implementation
- Better tracking
- Clearer user flow
- Easier debugging

Implementation Comparison

Current Implementation (Account Links)

```
User clicks "Setup Payments"
    ↓
Create AccountLink
    ↓
Redirect to stripe.com
    ↓
User completes form on Stripe
    ↓
Redirect back to app
    ↓
Check status via GET /api/stripe/connect-onboard
```

Recommended Implementation (Embedded Components)

```
User clicks "Setup Payments"
    ↓
Create Account Session
    ↓
Load Stripe Connect JS
    ↓
Render embedded component IN APP
    ↓
User completes form (never leaves app)
    ↓
onExit callback triggered
    ↓
Refresh account status
```

Migration Path

Step 1: Add Embedded Components SDK

```
npm install @stripe/connect-js
```

Step 2: Update Backend (Create Account Session)

```
// app/api/stripe/connect-onboard/route.ts
export async function POST(request: NextRequest) {
  // ... existing account creation logic ...

  // Instead of AccountLink, create AccountSession
  const accountSession = await stripe.accountSessions.create({
    account: accountId,
    components: {
      account_onboarding: { enabled: true },
    },
  });

  return NextResponse.json({
    clientSecret: accountSession.client_secret,
  });
}
```

Step 3: Update Frontend (Use Embedded Component)

```
// app/dashboard/creator/payment-setup/page.tsx
import { loadConnectAndInitialize } from '@stripe/connect-js';
import { ConnectAccountOnboarding } from '@stripe/react-connect-js';

export default function PaymentSetupPage() {
  const [clientSecret, setClientSecret] = useState('');

  // Fetch client secret
  useEffect(() => {
    fetch('/api/stripe/connect-onboard', { method: 'POST' })
      .then(res => res.json())
      .then(data => setClientSecret(data.clientSecret));
  }, []);

  // Initialize Stripe Connect
  const stripeConnectInstance = loadConnectAndInitialize({
    publishableKey: process.env.NEXT_PUBLIC_STRIPE_PUBLISHABLE_KEY,
    fetchClientSecret: async () => clientSecret,
  });

  return (
    <div className="payment-setup">
      <h1>Setup Your Payments</h1>
      <ConnectAccountOnboarding
        stripeConnectInstance={stripeConnectInstance}
        onExit={() => {
          // Refresh status
          window.location.reload();
        }}
      />
    </div>
  );
}
```

Conclusion

Final Recommendation

Use Stripe Connect Embedded Components for the best user experience:

1. ✅ **Create account immediately** when user becomes creator
2. ✅ **Use embedded onboarding** to keep users in your app
3. ✅ **Use embedded account management** for post-onboarding updates
4. ✅ **Leverage webhooks** (`account.updated`) for real-time status updates

This approach provides:
- **Best UX:** No full-page redirects
- **Compliance:** Stripe handles all KYC and verification
- **Security:** PCI compliant by default
- **Simplicity:** Less code to maintain than AccountLinks
- **Future-proof:** Aligns with Stripe's recommended practices

Resources

- Stripe Connect Embedded Components (https://stripe.com/docs/connect/get-started-connect-embedded-components)
- Account Onboarding Component (https://stripe.com/docs/connect/embedded-components/onboarding)
- Account Management Component (https://stripe.com/docs/connect/embedded-components/account-management)
- Account Sessions API (https://stripe.com/docs/api/account_sessions)

Last Updated: December 26, 2025
Platform: Call a Star
Stripe Account Type: Connect Express