Database Schema Fix - Migration Guide

Problem Identified

The `customers` table had an inappropriate unique constraint on the `name` field (company name), which was causing this error when creating API keys:

```
duplicate key value violates unique constraint "customers_name_key"
```

Why this is wrong:
- Multiple different companies can legitimately have similar names (e.g., “Test Company”, “ABC Corp”, etc.)
- The unique constraint should be on the `contact_email` field instead, to prevent duplicate accounts with the same email

Changes Made

1. Prisma Schema Changes (`prisma/schema.prisma`)

Customer Model:
- ✅ **Removed** `@unique` decorator from `name` field (line 15)
- ✅ **Added** `@unique` decorator to `contact_email` field (line 19)

CreditTransaction Model:
- ✅ **Fixed** `metadata` field type from `String?` to `Json?` (line 226)

2. Database Migration Created

A migration file has been created at:

```
prisma/migrations/20251102091938_fix_customer_unique_constraints/migration.sql
```

How to Apply the Migration

Option 1: Automatic Migration (Recommended for Vercel)

When you deploy to Vercel, the migration will be automatically applied as part of the build process if you have Prisma migrations set up correctly.

Make sure your `package.json` has the following script:

```
{
  "scripts": {
    "postinstall": "prisma generate",
    "vercel-build": "prisma migrate deploy && next build"
  }
}
```

Option 2: Manual Migration (For Local/Production Database)

If you need to apply the migration manually, you can either:

A. Using Prisma CLI:

```
cd /path/to/roblox-tool
npx prisma migrate deploy
```

B. Using Direct SQL (if Prisma CLI is not available):

```
-- Drop unique constraint on name field
ALTER TABLE "customers" DROP CONSTRAINT IF EXISTS "customers_name_key";

-- Add unique constraint on contact_email field
ALTER TABLE "customers" ADD CONSTRAINT "customers_contact_email_key" UNIQUE ("con-
tact_email");
```

Option 3: Reset Database (Development Only - CAUTION: Will Delete All Data)

⚠️ **WARNING: This will delete all existing data!**

If you're in development and want a clean slate:

```
cd /path/to/roblox-tool
npx prisma migrate reset
```

Testing the Fix

After applying the migration, you can test the API endpoint with the following scenarios:

Test 1: Create First Customer

```
curl -X POST https://your-app-url.vercel.app/api/v1/keys/create \
  -H "Content-Type: application/json" \
  -d '{
    "email": "company1@example.com",
    "companyName": "Test Company"
  }'
```

✅ **Expected**: Success - Creates customer with company name "Test Company"

Test 2: Create Second Customer with Same Company Name (Different Email)

```
curl -X POST https://your-app-url.vercel.app/api/v1/keys/create \
  -H "Content-Type: application/json" \
  -d '{
    "email": "company2@example.com",
    "companyName": "Test Company"
  }'
```

✅ **Expected**: Success - Creates another customer with same company name but different email

Test 3: Create Customer with Duplicate Email (Should Fail)

```
curl -X POST https://your-app-url.vercel.app/api/v1/keys/create \
  -H "Content-Type: application/json" \
  -d '{
    "email": "company1@example.com",
    "companyName": "Another Company"
  }'
```

❌ **Expected**: Error - Email already exists (because we now enforce unique emails)

Postman Testing

Use these request bodies in Postman:

Successful Test 1:

```
{
  "email": "lawfirm1@example.com",
  "companyName": "Smith & Associates"
}
```

Successful Test 2 (Same company name, different email):

```
{
  "email": "lawfirm2@example.com",
  "companyName": "Smith & Associates"
}
```

Successful Test 3 (Different company):

```
{
  "email": "lawfirm3@example.com",
  "companyName": "Johnson Legal Group"
}
```

Verification

After applying the migration, verify the constraints:

```
-- Check constraints on customers table
SELECT
    conname AS constraint_name,
    contype AS constraint_type,
    a.attname AS column_name
FROM pg_constraint AS c
JOIN pg_class AS t ON c.conrelid = t.oid
JOIN pg_attribute AS a ON a.attnum = ANY(c.conkey) AND a.attrelid = t.oid
WHERE t.relname = 'customers'
AND contype IN ('u', 'p')
ORDER BY constraint_name;
```

Expected Results:
- ✅ `customers_contact_email_key` - UNIQUE constraint on `contact_email`
- ✅ `customers_pkey` - PRIMARY KEY on `id`
- ✅ `customers_stripe_customer_id_key` - UNIQUE constraint on `stripe_customer_id`
- ❌ `customers_name_key` - Should NOT exist (removed)

Summary

Before:
- ❌ Company name was unique (incorrect)
- ❌ Email was not unique (incorrect)
- ❌ Multiple companies couldn't have similar names

After:
- ✅ Company name is NOT unique (correct)
- ✅ Email is unique (correct)
- ✅ Multiple companies can have similar names
- ✅ One account per email address

Next Steps

1. ✅ **Schema fixed and committed** to Git
2. ⏳ **Apply migration** to your Vercel database
3. ⏳ **Test the endpoint** with the examples above
4. ⏳ **Verify** that duplicate company names work correctly
5. ⏳ **Verify** that duplicate emails are properly rejected

Commit: `7f35218`
Branch: `main`
Status: ✅ Pushed to GitHub