Deployment Instructions - API Key Creation Fix

🎉 All Issues Have Been Fixed!

All the fixes have been committed and pushed to GitHub. Here's what you need to do next:

Quick Start (5 Minutes)

Step 1: Run the Database Migration

Choose one of these options:

Option A: Using Prisma (Recommended)

```
cd /home/ubuntu/github_repos/roblox-tool
export DATABASE_URL="postgresql://postgres:password@localhost:5432/roblox_verifier"
npx prisma migrate deploy
```

Option B: Using psql directly

```
psql -h localhost -U postgres -d roblox_verifier -f prisma/migrations/
20251103160532_fix_api_issues/migration.sql
```

Option C: Using pgAdmin or your preferred database client

Open the migration file and execute it:
- File: `prisma/migrations/20251103160532_fix_api_issues/migration.sql`

Step 2: Restart Your Application

```
# If running locally
cd /home/ubuntu/github_repos/roblox-tool
npm run dev

# If deployed on Vercel, the push will automatically trigger a redeploy
# Just wait for the deployment to complete
```

Step 3: Test the Fixed Endpoint

Use Postman or curl to test:

```
curl -X POST http://localhost:3000/api/v1/keys/create \
  -H "Content-Type: application/json" \
  -d '{
    "email": "newuser@example.com",
    "companyName": "New Test Company",
    "password": "SecurePassword123!"
  }'
```

Expected response:

```
{
  "success": true,
  "data": {
    "apiKey": "vrl_live_xxxxxxxxxxxxx",
    "customerId": 18,
    "companyName": "New Test Company",
    "email": "newuser@example.com",
    "keyId": 1,
    "scopes": ["verify:read", "credits:read", "credits:write", "usage:read"],
    "rateLimit": 1000
  },
  "message":
"Customer created successfully. Save your API key securely - it cannot be retrieved
later."
}
```

Step 4: Verify the Fixes

1. **Check contact_email is set:**
   `sql`
   `SELECT id, name, contact_email FROM customers WHERE name = 'New Test Company';`
   Should show the email address, not NULL

2. **Test user login:**
   - Go to your login page
   - Login with email: `newuser@example.com`
   - Password: `SecurePassword123!`
   - Should successfully authenticate

3. **Test API key works:**
   `bash`
   `curl -X GET http://localhost:3000/api/v1/search/user?username=test \`
   `-H "Authorization: Bearer vrl_live_xxxxxxxxxxxxx"`

What Was Fixed

1. ✅ VARCHAR(10) Constraint

- **Problem:** API key prefix needed 12 characters but database only allowed 10
- **Fix:** Migration changes `api_keys.key_prefix` from VARCHAR(10) to VARCHAR(20)

2. ✅ NULL Contact Email

- **Problem:** Contact email was always NULL when creating customers
- **Fix:** Updated `createCustomerWithAdmin()` to set contact_email field

3. ✅ Password Not Used

- **Problem:** API was ignoring the password from request body
- **Fix:** Updated endpoint to use provided password (with 8 char minimum validation)

4. ✅ Login Failures

- **Problem:** Users couldn't login after registration
- **Fix:** Now uses bcrypt hashed password from request, allowing successful login

Files Changed

```
✅ prisma/migrations/20251103160532_fix_api_issues/migration.sql (NEW)
✅ src/app/lib/db/index.ts (MODIFIED)
✅ src/app/api/v1/keys/create/route.ts (MODIFIED)
✅ FIX_SUMMARY.md (NEW - comprehensive testing guide)
```

Verification Checklist

- [ ] Database migration completed successfully
- [ ] Application restarted
- [ ] POST /api/v1/keys/create returns 200 (not 500)
- [ ] Response includes valid API key (vrl_live_...)
- [ ] customers.contact_email is populated (not NULL)
- [ ] User can login with provided password
- [ ] API key works for API requests

Troubleshooting

Issue: “Can’t reach database server”

```
# Check if PostgreSQL is running
sudo systemctl status postgresql

# Start if needed
sudo systemctl start postgresql
```

Issue: “Error: relation ‘api_keys’ does not exist”

Your database schema is not up to date. Run:

```
npx prisma migrate deploy
```

Issue: “Duplicate key value violates unique constraint”

Customer with this email already exists. Use a different email or delete the existing customer.

Issue: Still getting VARCHAR(10) error

The migration didn’t run. Manually run:

```
export DATABASE_URL="postgresql://postgres:password@localhost:5432/roblox_verifier"
npx prisma migrate deploy
```

For Production (Vercel)

1. Environment Variables

Make sure these are set in Vercel:
- `DATABASE_URL` - Your production PostgreSQL connection string

- `NEXTAUTH_URL` - Your production URL
- `NEXTAUTH_SECRET` - Your NextAuth secret

2. Deploy

The push to GitHub will automatically trigger a Vercel deployment.

3. Run Migration on Production Database

After deployment, run the migration on your production database:

```
# Using Vercel CLI
vercel env pull .env.production
export $(cat .env.production | xargs)
npx prisma migrate deploy
```

Or connect to your production database directly and run the migration SQL.

Testing in Production

After deploying to production:

```
curl -X POST https://your-domain.vercel.app/api/v1/keys/create \
  -H "Content-Type: application/json" \
  -d '{
    "email": "production-test@example.com",
    "companyName": "Production Test Company",
    "password": "SecurePassword123!"
  }'
```

Support

If you encounter any issues:

1. **Check application logs:**
   - Local: Terminal where `npm run dev` is running
   - Vercel: Dashboard → Your Project → Deployments → Latest → Logs

2. **Check database:**
   ```sql
   – Verify migration ran
   SELECT * FROM _prisma_migrations ORDER BY applied_at DESC LIMIT 5;

– Check key_prefix column size
SELECT character_maximum_length
FROM information_schema.columns
WHERE table_name = 'api_keys' AND column_name = 'key_prefix';
```

1. **Review detailed fix summary:**
   - See `FIX_SUMMARY.md` in the project root

Next Steps

1. ✅ Run the database migration (Step 1)
2. ✅ Restart your application (Step 2)
3. ✅ Test with Postman (Step 3)
4. ✅ Verify all fixes (Step 4)
5. 🎉 Start using the API!

Success Criteria

You'll know everything is working when:
- ✅ POST /api/v1/keys/create returns 200 (not 500)
- ✅ Response includes a valid API key
- ✅ Customer record has contact_email populated
- ✅ User can login with the password they provided
- ✅ API key works for making API requests

Need help?
- See `FIX_SUMMARY.md` for detailed testing instructions
- Check the commit message for technical details
- Review the migration SQL to understand database changes