My Study Tool - Testing and Deployment Documentation

Your Name

February 23, 2025

Contents

1 Introduction

This document provides a step-by-step guide for testing, debugging, and deploying the My Study Tool application. It covers local development setup, using Stripe webhooks with the Stripe CLI, and deploying to a production environment (e.g., Vercel).

2 Local Development Setup

2.1 Clone the Repository

Open your terminal and run:

```
git clone https://github.com/yourusername/my-study-tool.git
cd my-study-tool
```

2.2 Install Dependencies

Run the following command:

```
npm install
```

Alternatively, if you use Yarn:

```
yarn install
```

2.3 Configure Environment Variables

Create a file named `.env.local` in the root of your project and add the following content:

```
NEXT_PUBLIC_SUPABASE_URL=https://your-supabase-project-url.supabase.co
NEXT_PUBLIC_SUPABASE_ANON_KEY=your_supabase_anon_key
STRIPE_SECRET_KEY=your_stripe_secret_key
STRIPE_PRICE_ID=your_stripe_price_id
STRIPE_WEBHOOK_SECRET=your_stripe_webhook_secret
NEXT_PUBLIC_BASE_URL=http://localhost:3000
NEXT_PUBLIC_STRIPE_PUBLISHABLE_KEY=your_stripe_publishable_key
```

Adjust the values as needed.

2.4 Database Setup

Ensure your Supabase project contains the required tables:

- **Profiles:** Stores extra user details such as `stripe_customer_id` and `subscription_status`.
- **Habits:** Tracks user habits, completion status, and points.
- **Study Sessions:** Logs study sessions and earned points.

3 Local Testing

3.1 Running the Development Server

Start your Next.js app in development mode:

```
npm run dev
```

Then, visit http://localhost:3000 in your browser to test pages such as Home, Dashboard, and Subscription.

3.2 Testing Supabase Integration

- **User Data & Tables:** Use the Supabase dashboard to verify that your tables (`profiles`, `habits`, `study_sessions`) are correctly receiving data when you create or update records from your application.
- **Real-Time Updates:** If you use Supabase's real-time features, check that changes are immediately reflected in your application UI.

3.3 Testing Stripe Webhooks

3.3.1 Using the Stripe CLI

Install the Stripe CLI: Follow the installation guide available at https://stripe.com/docs/stripe-cli#install. For example, on macOS:

```
brew install stripe/stripe-cli/stripe
```

Login and Authenticate: Run:

```
stripe login
```

Forward Webhook Events: Run the following command to forward Stripe events to your local webhook endpoint:

```
stripe listen --forward-to http://localhost:3000/api/stripe-webhook
```

The CLI will output a webhook secret; verify that this secret matches your `STRIPE_WEBHOOK_SECRET` in your `.env.local`.

Trigger Test Events: To simulate a checkout session completion, run:

```
stripe trigger checkout.session.completed
```

Check your terminal and logs to ensure that the event is received and processed by your webhook endpoint (e.g., updating the Supabase `profiles` table).

4 Automated Testing

4.1 Unit and Integration Tests

Consider setting up tests using frameworks such as Jest and React Testing Library. Example commands:

```
npm run test
```

or

```
yarn test
```

4.2 Linting

Ensure your code adheres to style guidelines:

```
npm run lint
```

or

```
yarn lint
```

Note: You may need to add specific test configurations and scripts based on your chosen testing framework.

5 Deployment Guide

5.1 Pre-deployment Checklist

Before deploying your application, ensure:

- All environment variables are set correctly for production (use live API keys for Stripe and the production URL for Supabase).
- Your Supabase tables are correctly configured.
- The application has been thoroughly tested locally.
- You have set up your webhook endpoint in Stripe to point to your production URL (e.g., https://yourdomain.com/api/stripe-webhook).

5.2 Deploying with Vercel

5.2.1 Push Your Code to GitHub (or your Git provider)

Ensure your latest changes are committed and pushed.

5.2.2 Sign Up / Log In to Vercel

Go to https://vercel.com and link your GitHub repository.

5.2.3 Configure Environment Variables on Vercel

In your project dashboard on Vercel, navigate to **Settings > Environment Variables** and add:

```
NEXT_PUBLIC_SUPABASE_URL
NEXT_PUBLIC_SUPABASE_ANON_KEY
STRIPE_SECRET_KEY
STRIPE_PRICE_ID
STRIPE_WEBHOOK_SECRET
NEXT_PUBLIC_BASE_URL   % Set to your production URL
NEXT_PUBLIC_STRIPE_PUBLISHABLE_KEY
```

5.2.4 Deploy Your Application

Vercel automatically deploys when you push to the main branch. You can also trigger manual deployments from the Vercel dashboard.

5.2.5 Verify Deployment

- Visit your production URL.
- Test key features (e.g., creating checkout sessions, webhook events, and Supabase integration).

6 Troubleshooting and Next Steps

6.1 Logs and Monitoring

- Use Vercel's dashboard for real-time logs.
- Monitor the Stripe Dashboard and Supabase logs for any errors.

6.2 Local Debugging

- Continue using the Stripe CLI to simulate webhook events.
- Utilize browser developer tools and your IDE's debugging features for frontend issues.

6.3 Documentation Updates

- Update this document as you add features or change configurations.
- Maintain version control for this documentation alongside your code.

6.4 Continuous Integration

Consider setting up CI/CD (e.g., using GitHub Actions) to run tests automatically on each push.

7 Further References

For additional details, please refer to:

- Stripe Documentation
- Supabase Documentation
- Next.js Documentation
- Vercel Documentation