# SafePlay Automatic Deployment Guide

## Overview

This project now has automatic deployment configured using GitHub Actions and Vercel. This eliminates the need for manual deployment requests and enables rapid testing through production deployment.

## Current Setup

### Version 1.4.1 Features

- ✅ Complete subscription management system working
- ✅ Geoapify autocomplete with 4-5 clickable suggestions working
- ✅ Billing address functionality working
- ✅ Success message system working
- ✅ Authentication and UI systems working
- ✅ Payment integration with Stripe working
- ✅ Comprehensive documentation and API guides

### Automatic Deployment Workflow

The project uses GitHub Actions to automatically deploy:
- **Production deployments**: Triggered on pushes to `main` branch
- **Preview deployments**: Triggered on pull requests

## Required GitHub Secrets

The following secrets must be configured in your GitHub repository settings:

1. `VERCEL_TOKEN` - Your Vercel access token
2. `VERCEL_ORG_ID` - Your Vercel organization ID
3. `VERCEL_PROJECT_ID` - Your Vercel project ID

### How to Get These Values

1. **VERCEL_TOKEN**:
   - Go to Vercel Dashboard → Settings → Tokens
   - Create a new token with appropriate permissions

2. **VERCEL_ORG_ID** and **VERCEL_PROJECT_ID**:
   - Found in `.vercel/project.json` after running `vercel link`
   - Or check your existing Vercel project settings

## Deployment Methods

### Method 1: Automatic Deployment Script (Recommended)

Use the provided deployment script for version updates:

```
# Make script executable (first time only)
chmod +x scripts/deploy.sh

# Deploy new version
./scripts/deploy.sh 1.2.9
```

This script will:
- Update version in `package.json`
- Update version in `components/version-tracker.tsx`
- Create a commit with proper versioning message
- Push to trigger automatic deployment

### Method 2: Manual Git Push

For immediate deployment of current changes:

```
git add .
git commit -m "feat: your feature description"
git push origin main
```

This triggers the production deployment workflow automatically.

### Method 3: Manual Workflow Trigger

You can manually trigger deployments from GitHub:
1. Go to Actions tab in your GitHub repository
2. Select "Vercel Production Deployment" workflow
3. Click "Run workflow" button

## Workflow Files

### Production Deployment (`.github/workflows/deploy-production.yml`)

- Triggers on pushes to `main` branch
- Builds and deploys to Vercel production environment
- Uses production environment variables

### Preview Deployment (`.github/workflows/deploy-preview.yml`)

- Triggers on pull requests
- Creates preview deployments for testing
- Provides preview URLs in PR comments

## Monitoring Deployments

### GitHub Actions

- Monitor deployment progress in the "Actions" tab
- View detailed logs for each deployment step
- Check for any build or deployment errors

### Vercel Dashboard

- View deployment history and status

- Access production and preview URLs
- Monitor performance metrics

# Version Management

## Current Version Tracking

The current version is tracked in:
- `package.json` - Main version field
- `components/version-tracker.tsx` - Display version in UI

## Version Naming Convention

Use semantic versioning: `MAJOR.MINOR.PATCH`
- `MAJOR` : Breaking changes
- `MINOR` : New features (backward compatible)
- `PATCH` : Bug fixes

Examples:
- `1.2.8` - Current version with Geoapify fixes
- `1.2.9` - Next patch version
- `1.3.0` - Next minor version with new features

# Troubleshooting

## Common Issues

1. **Deployment Fails**
   - Check GitHub Actions logs for specific errors
   - Verify all required secrets are set correctly
   - Ensure Vercel project is properly linked

2. **Build Errors**
   - Check for TypeScript errors (currently disabled but may cause issues)
   - Verify all dependencies are properly installed
   - Check environment variables are set correctly

3. **Vercel Authentication Issues**
   - Regenerate `VERCEL_TOKEN` if expired
   - Verify `VERCEL_ORG_ID` and `VERCEL_PROJECT_ID` are correct

## Skip CI

Add `[skip ci]` to commit message to skip deployment:

```
git commit -m "docs: update README [skip ci]"
```

# Rollback Process

If a deployment needs to be rolled back:

1. **Quick Rollback**:
   - Go to Vercel dashboard

- Find previous successful deployment<br>
- Click “Promote to Production”

2. **Git Rollback**:

```
bash
   git revert <commit-hash>
   git push origin main
```

## Environment Variables

Production environment variables are managed in:
- Vercel Dashboard → Project Settings → Environment Variables
- Separate variables for Development, Preview, and Production

## Security Notes

- Never commit `.vercel/` directory to git
- Keep GitHub secrets secure and rotate tokens regularly
- Use environment-specific variables for sensitive data

## Support

For deployment issues:
1. Check GitHub Actions logs first
2. Verify Vercel dashboard for deployment status
3. Review this documentation for troubleshooting steps

---

**Last Updated**: January 13, 2025<br>
**Current Version**: 1.4.1<br>
**Deployment Status**: ✅ Automatic deployment active