

BCH Content Hub - GitHub Deployment Guide

Overview

This guide will help you deploy your BCH Content Hub to GitHub and GitHub Pages, connect it with your Supabase database, and set up a development workflow.

Prerequisites

- GitHub account
- Git installed on your computer
- Node.js installed (version 16 or higher)
- Your Supabase credentials
- YouTube Data API key

Step 1: Prepare Your Local Project

1.1 Download Your Project Files

First, ensure you have all the project files from your current deployment:

```
# Create a new directory for your project
mkdir bch-content-hub
cd bch-content-hub

# Initialize git repository
git init
```

1.2 Set Up Project Structure



Copy all your project files into this directory. Your structure should look like:

```
bch-content-hub/
├─ public/
├─ src/
├─ supabase/
├─ package.json
├─ vite.config.ts
├─ tailwind.config.js
├─ tsconfig.json
└─ README.md
```

Step 2: Create GitHub Repository

2.1 Create Repository on GitHub

1. Go to [GitHub.com](https://github.com)
2. Click the "+" icon → "New repository"
3. Repository name: `bch-content-hub`
4. Description: "Bitcoin Cash Content Curation Platform"
5. Set as Public (required for free GitHub Pages)

6.  Add a README file
7.  Add .gitignore (Node template)
8. Click "Create repository"

2.2 Connect Local Project to GitHub

```
# Add your GitHub repository as remote
git remote add origin https://github.com/YOUR_USERNAME/bch-content-
hub.git

# Create and switch to main branch
git branch -M main

# Add all files
git add .

# Make initial commit
git commit -m "Initial commit: BCH Content Hub platform"


# Push to GitHub
git push -u origin main
```

Step 3: Configure Environment Variables

3.1 Create Environment File

Create a `.env` file in your project root:

```
# .env
VITE_SUPABASE_URL=your_supabase_url_here
VITE_SUPABASE_ANON_KEY=your_supabase_anon_key_here
VITE_YOUTUBE_API_KEY=your_youtube_api_key_here
```

 **IMPORTANT:** Never commit `.env` file to GitHub!

3.2 Update .gitignore

Ensure your `.gitignore` includes:

```
# Environment variables
.env
.env.local
.env.production

# Dependencies
node_modules/

# Build output
dist/
build/

# IDE
.vscode/
.idea/

# OS
.DS_Store
Thumbs.db
```

3.3 Create Environment Template

Create `.env.example` for other developers:

```
# .env.example
VITE_SUPABASE_URL=your_supabase_url_here
VITE_SUPABASE_ANON_KEY=your_supabase_anon_key_here
VITE_YOUTUBE_API_KEY=your_youtube_api_key_here
```

Step 4: Set Up GitHub Pages Deployment

4.1 Install GitHub Pages Plugin

```
npm install --save-dev gh-pages
```

4.2 Update package.json

Add deployment scripts to your `package.json`:

```
{
  "scripts": {
    "dev": "vite",
    "build": "tsc && vite build",
    "preview": "vite preview",
    "predeploy": "npm run build",
    "deploy": "gh-pages -d dist"
  },
  "homepage": "https://YOUR_USERNAME.github.io/bch-content-hub"
}
```

4.3 Update Vite Configuration

Update `vite.config.ts` for GitHub Pages:

```
import { defineConfig } from 'vite'
import react from '@vitejs/plugin-react'

export default defineConfig({
  plugins: [react()],
  base: '/bch-content-hub/', // Your repository name
  build: {
    outDir: 'dist',
    assetsDir: 'assets'
  }
})
```

4.4 Deploy to GitHub Pages

```
# Build and deploy
npm run deploy
```

Your site will be available at: `https://YOUR_USERNAME.github.io/bch-content-hub`

Step 5: Configure GitHub Actions (Optional but Recommended)

Create `.github/workflows/deploy.yml` for automatic deployment:

```

name: Deploy to GitHub Pages

on:
  push:
    branches: [ main ]
  pull_request:
    branches: [ main ]

jobs:
  build-and-deploy:
    runs-on: ubuntu-latest

    steps:
      - name: Checkout
        uses: actions/checkout@v3

      - name: Setup Node.js
        uses: actions/setup-node@v3
        with:
          node-version: '18'
          cache: 'npm'

      - name: Install dependencies
        run: npm ci

      - name: Build
        run: npm run build
        env:
          VITE_SUPABASE_URL: ${ secrets.VITE_SUPABASE_URL }
          VITE_SUPABASE_ANON_KEY: $
            {{ secrets.VITE_SUPABASE_ANON_KEY }}
          VITE_YOUTUBE_API_KEY: ${ secrets.VITE_YOUTUBE_API_KEY }

      - name: Deploy to GitHub Pages
        uses: peaceiris/actions-gh-pages@v3

```

```
with:
  github_token: ${ secrets.GITHUB_TOKEN }
  publish_dir: ./dist
```

5.1 Add GitHub Secrets

1. Go to your repository on GitHub
2. Settings → Secrets and variables → Actions
3. Click "New repository secret"
4. Add these secrets:
 - `VITE_SUPABASE_URL` : Your Supabase URL
 - `VITE_SUPABASE_ANON_KEY` : Your Supabase anonymous key
 - `VITE_YOUTUBE_API_KEY` : Your YouTube API key

Step 6: Database Connection Verification

6.1 Test Supabase Connection

Create a test file to verify your database connection:


```
// src/utils/testConnection.ts
import { supabase } from '../supabase'

export async function testDatabaseConnection() {
  try {
    const { data, error } = await supabase
      .from('videos')
      .select('count')
      .limit(1)

    if (error) {
      console.error('Database connection failed:', error)
      return false
    }

    console.log('Database connection successful!')
    return true
  } catch (err) {
    console.error('Connection test failed:', err)
    return false
  }
}
```

6.2 Check Environment Variables

Add this to your main component for debugging:

```
// In your main App.tsx or similar
useEffect(() => {
  console.log('Supabase URL:',
import.meta.env.VITE_SUPABASE_URL?.substring(0, 20) + '...')
  console.log('API Key present:', !!
import.meta.env.VITE_YOUTUBE_API_KEY)
}, [])
```

Step 7: Troubleshooting Common Issues

7.1 GitHub Pages Not Loading

- Check that repository is public
- Verify GitHub Pages is enabled in repository settings
- Ensure `base` in `vite.config.ts` matches repository name
- Wait 5-10 minutes for deployment to propagate

7.2 Environment Variables Not Working

- Ensure variable names start with `VITE_`
- Check GitHub secrets are properly set
- Verify `.env` file is not committed to repository
- Clear browser cache and rebuild

7.3 Database Connection Issues

- Verify Supabase URL and keys are correct
- Check Supabase project is active
- Ensure database tables exist
- Test connection in Supabase dashboard

7.4 Build Failures

- Check all dependencies are installed
- Verify TypeScript types are correct
- Ensure all imports are properly resolved
- Check for console errors during build

Step 8: Quick Reference Commands

```
# Local development
npm run dev

# Build for production
npm run build

# Preview production build
npm run preview

# Deploy to GitHub Pages
npm run deploy

# Update from GitHub
git pull origin main

# Commit and push changes
git add .
git commit -m "Your commit message"
git push origin main
```

Next Steps

- Set up custom domain (optional)

- Configure SSL certificate
 - Set up monitoring and analytics
 - Consider migrating to Vercel or Netlify for better performance
-

Need Help?

- GitHub Pages Documentation: <https://pages.github.com/>
- Supabase Documentation: <https://supabase.com/docs>
- Vite Documentation: <https://vitejs.dev/guide/>