How to Push Your Code to GitHub

Your code is ready to be pushed to GitHub! Follow these instructions to authenticate and push.

📋 Prerequisites

- GitHub account
- Repository already exists: `https://github.com/Stefanbotes/Studiov2.git`
- Git configured locally (already done)

🔐 Authentication Options

GitHub requires authentication for pushing code. Choose one of these methods:

Option 1: Personal Access Token (Recommended)

This is the most common and secure method for HTTPS authentication.

Step 1: Create a Personal Access Token

1. Go to GitHub → **Settings** → **Developer settings**
   - Direct link: github.com/settings/tokens (https://github.com/settings/tokens)

2. Click **"Personal access tokens"** → **"Tokens (classic)"**

3. Click **"Generate new token"** → **"Generate new token (classic)"**

4. Configure your token:
   - **Note**: `Studio 2 Deployment Token`
   - **Expiration**: Choose duration (90 days recommended)
   - **Scopes**: Select these permissions:

     - ✅ `repo` (Full control of private repositories)
     - ✅ `workflow` (Update GitHub Action workflows)

5. Click **"Generate token"**

6. **IMPORTANT**: Copy the token immediately (you won't see it again!)
   - Format: `ghp_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx`
   - Save it securely (password manager recommended)

Step 2: Push with Personal Access Token

```
cd /home/ubuntu/studio_2_app/nextjs_space

# Push to GitHub using the token as password
git push -u origin master

# When prompted:
# Username: Stefanbotes
# Password: <paste your personal access token>
```

Note: The token is your password - not your actual GitHub password!

Option 2: GitHub CLI (Easiest)

If you have GitHub CLI installed:

```
# Install GitHub CLI (if not already)
curl -fsSL https://cli.github.com/packages/githubcli-archive-keyring.gpg | sudo dd of=/usr/share/keyrings/githubcli-archive-keyring.gpg
echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/githubcli-archive-keyring.gpg] https://cli.github.com/packages stable main" | sudo tee /etc/apt/sources.list.d/github-cli.list > /dev/null
sudo apt update
sudo apt install gh

# Authenticate
gh auth login
# Follow the prompts:
# - Choose: GitHub.com
# - Choose: HTTPS
# - Authenticate via web browser

# Push
cd /home/ubuntu/studio_2_app/nextjs_space
git push -u origin master
```

Option 3: SSH Key (Advanced)

If you prefer SSH authentication:

Step 1: Generate SSH Key

```
# Generate SSH key
ssh-keygen -t ed25519 -C "your_email@example.com"

# Press Enter to accept default location
# Enter passphrase (or press Enter for no passphrase)

# Copy the public key
cat ~/.ssh/id_ed25519.pub
```

Step 2: Add SSH Key to GitHub

1. Go to GitHub → **Settings** → **SSH and GPG keys**
   - Direct link: github.com/settings/keys (https://github.com/settings/keys)

2. Click **"New SSH key"**

3. Fill in:
   - **Title**: `Studio 2 Server`
   - **Key**: Paste the output from `cat ~/.ssh/id_ed25519.pub`

4. Click **"Add SSH key"**

Step 3: Update Remote and Push

```
cd /home/ubuntu/studio_2_app/nextjs_space

# Change remote to SSH
git remote set-url origin git@github.com:Stefanbotes/Studiov2.git

# Test connection
ssh -T git@github.com
# Should say: "Hi Stefanbotes! You've successfully authenticated..."

# Push
git push -u origin master
```

🚀 Push Your Code

Once you've chosen and set up authentication, run:

```
cd /home/ubuntu/studio_2_app/nextjs_space
git push -u origin master
```

Expected output:

```
Enumerating objects: X, done.
Counting objects: 100% (X/X), done.
Delta compression using up to Y threads
Compressing objects: 100% (X/X), done.
Writing objects: 100% (X/X), Y MiB | Z MiB/s, done.
Total X (delta Y), reused Z (delta W)
To https://github.com/Stefanbotes/Studiov2.git
   abc1234..def5678  master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

✅ Verify Your Push

1. **Visit your GitHub repository**
   - Go to: github.com/Stefanbotes/Studiov2 (https://github.com/Stefanbotes/Studiov2)

2. **Check for new files**
   - You should see:

     - ✅ `vercel.json`
     - ✅ `VERCEL_DEPLOYMENT_GUIDE.md`
     - ✅ `VERCEL_ENV_VARIABLES.md`
     - ✅ `TROUBLESHOOTING.md`
     - ✅ `.env.example`
     - ✅ Updated `next.config.js`
     - ✅ Updated `prisma/schema.prisma`
3. **Check commit message**
   - Should show: “Prepare for Vercel deployment: Configure for production”

🔄 After Pushing Successfully

Once your code is on GitHub, you can:

1. **Deploy to Vercel**
   - Follow the VERCEL_DEPLOYMENT_GUIDE.md (./VERCEL_DEPLOYMENT_GUIDE.md)
   - Import your GitHub repository to Vercel
   - Configure environment variables
   - Deploy!

2. **Future Updates**
   ```bash
   # Make changes to your code
   git add .
   git commit -m “Your update message”
   git push

# Vercel automatically deploys on push to master
```

🆘 Troubleshooting Push Issues

Issue: “Permission denied (publickey)”

Solution: Use Personal Access Token or add SSH key (see above)

Issue: “Repository not found”

Solution: Check repository URL and your GitHub permissions

```
# Verify remote URL
git remote -v

# Update if needed
git remote set-url origin https://github.com/Stefanbotes/Studiov2.git
```

Issue: “Failed to push some refs”

Solution: Pull latest changes first

```
git pull origin master --rebase
git push origin master
```

Issue: “Support for password authentication was removed”

Solution: GitHub no longer accepts passwords - use Personal Access Token or SSH

📝 Quick Reference

Check Current Status

```
git status
git log --oneline -5
```

View Remote Configuration

```
git remote -v
```

Force Push (⚠️ Use with caution)

```
git push -f origin master
# WARNING: This overwrites remote history
```

Create a New Branch (Recommended)

```
# Create and switch to new branch
git checkout -b vercel-deployment

# Push new branch
git push -u origin vercel-deployment
```

🎯 Next Steps

After successfully pushing to GitHub:

1. ✅ Verify files are on GitHub
2. ✅ Follow VERCEL_DEPLOYMENT_GUIDE.md (./VERCEL_DEPLOYMENT_GUIDE.md)
3. ✅ Import repository to Vercel
4. ✅ Configure environment variables
5. ✅ Deploy and test!

Need Help?

- GitHub Authentication Docs (https://docs.github.com/en/authentication)
- GitHub Personal Access Tokens (https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token)
- GitHub SSH Keys (https://docs.github.com/en/authentication/connecting-to-github-with-ssh)

Last Updated: October 2025