ArchitectPro - Vercel Deployment Guide

🚀 Complete Step-by-Step Deployment Instructions

Prerequisites

✅ GitHub/GitLab/Bitbucket account
✅ Vercel account (free tier works)
✅ PostgreSQL database (Vercel Postgres, Neon, Supabase, or AWS RDS)

Step 1: Prepare Your Project

1.1 Update Prisma Schema for Vercel

Edit `prisma/schema.prisma` and update the `generator` block:

```
generator client {
  provider = "prisma-client-js"
  binaryTargets = ["native", "rhel-openssl-3.0.x"]
}

datasource db {
  provider = "postgresql"
  url      = env("DATABASE_URL")
}
```

Important:
- Remove the custom `output` path if present
- Add `"rhel-openssl-3.0.x"` for Vercel's Node.js runtime
- Keep `"native"` for local development

1.2 Add `postinstall` Script

Edit `package.json` and add to scripts:

```
{
  "scripts": {
    "dev": "next dev",
    "build": "next build",
    "start": "next start",
    "lint": "next lint",
    "postinstall": "prisma generate"
  }
}
```

1.3 Create `.vercelignore` (Optional)

Create a `.vercelignore` file in the root:

```
.env
.env.local
node_modules
.next
.git
*.log
.DS_Store
```

1.4 Remove Development Database URL

Create `.env.example` without actual credentials:

```
DATABASE_URL="postgresql://user:password@host:5432/database"
```

DO NOT COMMIT `.env` with real credentials!

Step 2: Set Up PostgreSQL Database

Option A: Vercel Postgres (Recommended for Vercel)

1. Go to your Vercel dashboard
2. Click **Storage** → **Create Database**
3. Select **Postgres**
4. Choose a region (same as deployment for lower latency)
5. Click **Create**
6. Vercel will automatically set `DATABASE_URL` environment variable

Option B: Neon (Free Tier Available)

1. Sign up at neon.tech (https://neon.tech)
2. Create a new project
3. Copy the connection string:
   `postgresql://username:password@ep-xxx.region.aws.neon.tech/dbname?sslmode=require`
4. Save this for Step 4

Option C: Supabase (Free Tier Available)

1. Sign up at supabase.com (https://supabase.com)
2. Create a new project
3. Go to **Settings** → **Database**
4. Copy the **Connection String** (Session mode)
5. Save this for Step 4

Option D: Railway (Free Tier Available)

1. Sign up at railway.app (https://railway.app)
2. Create a new project → **Add PostgreSQL**
3. Copy the **DATABASE_URL** from the connection tab
4. Save this for Step 4

Step 3: Push Code to Git Repository

3.1 Initialize Git (if not already)

```
cd /home/ubuntu/architect_pro/nextjs_space
git init
```

3.2 Create `.gitignore`

```
# Dependencies
node_modules

# Next.js
.next
out

# Environment variables
.env
.env.local
.env.production.local
.env.development.local
.env.test.local

# Vercel
.vercel

# Logs
*.log
npm-debug.log*
yarn-debug.log*
yarn-error.log*

# OS
.DS_Store
Thumbs.db

# IDE
.idea
.vscode
*.swp
*.swo

# Prisma
node_modules/.prisma
```

3.3 Commit and Push

```
git add .
git commit -m "Initial commit - ArchitectPro floor plan designer"

# For GitHub
git remote add origin https://github.com/yourusername/architect-pro.git
git branch -M main
git push -u origin main

# For GitLab
git remote add origin https://gitlab.com/yourusername/architect-pro.git
git branch -M main
git push -u origin main
```

Step 4: Deploy to Vercel

4.1 Import Project

1. Go to vercel.com/new (https://vercel.com/new)
2. Click **Import Project**
3. Select your Git provider (GitHub/GitLab/Bitbucket)
4. Authorize Vercel to access your repository
5. Select the **architect-pro** repository

4.2 Configure Project

Framework Preset: Next.js (auto-detected)
Root Directory: `nextjs_space`
Build Command: `next build` (default)
Output Directory: `.next` (default)
Install Command: `yarn install` (default)

4.3 Add Environment Variables

Click **Environment Variables** and add:

Name	Value	Example
`DATABASE_URL`	Your PostgreSQL connection string	`postgresql://user:pass@host:5432/db`

Important Security Notes:
- ✅ Use `?sslmode=require` at the end of connection string for cloud databases
- ✅ Ensure your database allows connections from Vercel's IP ranges
- ✅ Use strong passwords

4.4 Deploy

Click **Deploy** and wait 2-5 minutes.

Vercel will:
1. Clone your repository

2. Install dependencies (`yarn install`)
3. Run `postinstall` (generates Prisma Client)
4. Build Next.js app (`next build`)
5. Deploy to edge network

Step 5: Initialize Database Schema

5.1 Run Database Migration

After first deployment, you need to create database tables.

Option A: Using Vercel CLI (Recommended)

Install Vercel CLI:

```
npm install -g vercel
```

Login and link project:

```
vercel login
vercel link
```

Run Prisma commands:

```
vercel env pull .env.local
npx prisma db push
```

Option B: Using Local Machine with Production DATABASE_URL

```
# Set production database URL temporarily
export DATABASE_URL="your_vercel_database_url"

# Push schema
npx prisma db push

# Seed database (optional)
npx prisma db seed
```

Option C: Using Prisma Data Platform

1. Go to cloud.prisma.io (https://cloud.prisma.io)
2. Import your project
3. Run migrations from web UI

5.2 Seed Database (Optional)

If you want to populate with initial templates:

```
# Using Vercel CLI
vercel env pull .env.local
npx tsx --require dotenv/config scripts/seed_new.ts
```

Step 6: Verify Deployment

6.1 Check Deployment Status

1. Go to Vercel dashboard
2. Click your project
3. Check **Deployments** tab
4. Click on the latest deployment
5. View **Build Logs** to ensure no errors

6.2 Test Your Application

1. Open the deployment URL (e.g., `architect-pro.vercel.app`)
2. Navigate to `/designer`
3. Try selecting different BHK types and property types
4. Verify floor plans load correctly
5. Test dimension adjustments

6.3 Check API Routes

```
# Test templates API
curl https://your-app.vercel.app/api/templates

# Test with filters
curl https://your-app.vercel.app/api/templates?bhkType=2BHK&propertyType=Apartment
```

Step 7: Configure Custom Domain (Optional)

7.1 Add Domain

1. Go to project settings
2. Click **Domains**
3. Click **Add**
4. Enter your domain (e.g., `architectpro.com`)

7.2 Configure DNS

Add the following DNS records at your domain registrar:

For Apex Domain (architectpro.com):

```
Type: A
Name: @
Value: 76.76.21.21
```

For WWW Subdomain (www.architectpro.com):

```
Type: CNAME
Name: www
Value: cname.vercel-dns.com
```

7.3 Wait for Propagation

DNS changes can take 24-48 hours to propagate globally.

Step 8: Continuous Deployment

Automatic Deployments

Vercel automatically deploys when you push to your repository:

```
# Make changes
git add .
git commit -m "Update floor plan features"
git push origin main

# Vercel automatically:
# 1. Detects push
# 2. Builds project
# 3. Deploys to production
```

Preview Deployments

Every pull request gets a unique preview URL:

```
# Create feature branch
git checkout -b feature/new-templates

# Make changes and push
git push origin feature/new-templates

# Create pull request on GitHub/GitLab
# Vercel creates preview deployment
```

🔧 Troubleshooting

Issue 1: Prisma Client Not Generated

Symptom: Build fails with “Cannot find module ‘@prisma/client’“

Solution:

```
// Ensure package.json has:
{
  "scripts": {
    "postinstall": "prisma generate"
  }
}
```

Issue 2: Database Connection Failed

Symptom: API routes return 500 errors

Solution:
1. Check `DATABASE_URL` in Vercel environment variables
2. Ensure `?sslmode=require` is in connection string
3. Verify database allows connections from Vercel IPs
4. Check database firewall settings

Issue 3: Build Timeout

Symptom: Build exceeds 15 minutes (free tier limit)

Solution:
1. Remove heavy dependencies
2. Optimize `node_modules`
3. Consider upgrading to Pro plan
4. Use `yarn install --frozen-lockfile`

Issue 4: API Routes Return 404

Symptom: `/api/templates` returns 404

Solution:
1. Ensure `nextjs_space` is set as root directory
2. Check file structure: `app/api/templates/route.ts`
3. Verify Next.js App Router is used (not Pages Router)

Issue 5: Environment Variables Not Working

Symptom: `process.env.DATABASE_URL` is undefined

Solution:
1. Re-add environment variables in Vercel dashboard
2. Trigger a new deployment (redeploy)
3. Ensure variables are set for **Production** environment

📊 Monitoring & Analytics

Vercel Analytics

1. Go to project settings
2. Enable **Analytics**
3. View:
   - Page views
   - Unique visitors
   - Core Web Vitals
   - API response times

Vercel Logs

1. Go to project
2. Click **Functions**

3. Select an API route
4. View real-time logs

Database Monitoring

Vercel Postgres:
- View metrics in Storage tab
- Connection count
- Query performance

Neon/Supabase:
- Use their built-in dashboards
- Monitor connection pooling
- Check slow queries

🔐 Security Best Practices

1. **Environment Variables**
   - Never commit `.env` to Git
   - Use different databases for dev/prod
   - Rotate database passwords regularly

2. **Database Security**
   - Enable SSL connections (`?sslmode=require`)
   - Use strong passwords (20+ characters)
   - Limit IP ranges if possible
   - Enable connection pooling

3. **API Security**
   - Add rate limiting (consider Vercel Pro features)
   - Validate all inputs
   - Implement CORS if needed
   - Use proper HTTP status codes

4. **Monitoring**
   - Set up error alerts
   - Monitor API usage
   - Track database performance
   - Review security logs

💰 Cost Considerations

Vercel Pricing

Free (Hobby):
- ✅ Unlimited deployments
- ✅ 100 GB bandwidth/month
- ✅ Serverless functions
- ❌ No commercial use
- ❌ Basic analytics

Pro ($20/month):
- ✅ Commercial use
- ✅ 1 TB bandwidth
- ✅ Advanced analytics
- ✅ Password protection
- ✅ Custom domains

Database Pricing

Vercel Postgres:
- $0.20/GB storage
- $0.40/1M queries

Neon (Free):
- 0.5 GB storage
- 10 GB data transfer

Supabase (Free):
- 500 MB database
- 5 GB bandwidth

🚀 Performance Optimization

1. Enable Edge Functions

Add to API routes:

```
export const runtime = 'edge';
```

2. Database Connection Pooling

For Prisma, use connection pooling:

```
datasource db {
  provider = "postgresql"
  url      = env("DATABASE_URL")
  directUrl = env("DIRECT_URL")
}
```

3. Image Optimization

Use Next.js Image component:

```
import Image from 'next/image';
```

4. Caching

Add caching headers:

```
export async function GET() {
  return NextResponse.json(data, {
    headers: {
      'Cache-Control': 'public, s-maxage=3600, stale-while-revalidate=7200'
    }
  });
}
```

📝 Deployment Checklist

Pre-Deployment

- [ ] Update Prisma schema `binaryTargets`
- [ ] Add `postinstall` script to `package.json`
- [ ] Create `.gitignore` file
- [ ] Remove sensitive data from code
- [ ] Test build locally (`npm run build`)
- [ ] Create `.env.example` file

Deployment

- [ ] Push code to Git repository
- [ ] Set up PostgreSQL database
- [ ] Import project to Vercel
- [ ] Configure root directory (`nextjs_space`)
- [ ] Add `DATABASE_URL` environment variable
- [ ] Deploy application

Post-Deployment

- [ ] Run database migration (`prisma db push`)
- [ ] Seed database (optional)
- [ ] Test all pages and API routes
- [ ] Verify floor plan rendering
- [ ] Check console for errors
- [ ] Set up custom domain (optional)
- [ ] Enable analytics
- [ ] Configure monitoring/alerts

🆘 Support & Resources

- **Vercel Documentation:** vercel.com/docs (https://vercel.com/docs)
- **Prisma Documentation:** prisma.io/docs (https://www.prisma.io/docs)
- **Next.js Documentation:** nextjs.org/docs (https://nextjs.org/docs)
- **Vercel Community:** github.com/vercel/vercel/discussions (https://github.com/vercel/vercel/discussions)

🎉 Congratulations!

Your ArchitectPro application is now live on Vercel! 🚀

Next Steps:

1. Share your deployment URL
2. Monitor performance and errors
3. Implement user feedback
4. Add new features and templates
5. Consider adding authentication for saved designs

Deployment URL: `https://your-project.vercel.app`