🚀 Complete Deployment Guide - Cosmic Outfits 3D Builder

> NOTE: This repository's documentation has been consolidated into `README.md`. Please consult the main `README.md` for the canonical, up-to-date guide. Original content follows for reference.

📋 Prerequisites

Before deploying, ensure you have:

- Node.js 18+ installed locally
- A Supabase account
- A Vercel account (recommended for deployment)
- Git repository set up

🗄️ Database Setup

1. Create Supabase Project

1. Go to supabase.com
2. Create a new project
3. Wait for the project to be ready
4. Note down your project URL and anon key

2. Run Database Scripts

Execute the following scripts in your Supabase SQL editor in order:

```sql -- 1. First run the complete database setup -- Copy and paste the content from scripts/00-complete-database-setup.sql

-- 2. Then run the profile table updates -- Copy and paste the content from scripts/05-update-profiles-table.sql

-- 3. Finally, clean up any duplicates -- Copy and paste the content from scripts/06-fix-duplicate-profiles.sql ```

### 3. Configure Authentication

In your Supabase dashboard:

1. Go to Authentication → Settings
2. Set Site URL to: `https://your-domain.com` (for production)
3. Add redirect URLs:
   - `https://your-domain.com/auth/callback`

## 🔧 Environment Configuration

### 1. Create Environment File
```

Create `.env.local` in your project root:

```env

# Supabase Configuration

NEXT_PUBLIC_SUPABASE_URL=your_supabase_project_url
NEXT_PUBLIC_SUPABASE_ANON_KEY=your_supabase_anon_key
SUPABASE_SERVICE_ROLE_KEY=your_supabase_service_role_key

# Database URLs (automatically provided by Supabase)

POSTGRES_URL=your_postgres_connection_string POSTGRES_PRISMA_URL=your_postgres_prisma_url
POSTGRES_URL_NON_POOLING=your_postgres_non_pooling_url POSTGRES_USER=your_postgres_user
POSTGRES_HOST=your_postgres_host POSTGRES_PASSWORD=your_postgres_password
POSTGRES_DATABASE=your_postgres_database ```

### 2. Get Your Supabase Keys

1. Go to your Supabase project dashboard
2. Navigate to Settings → API
3. Copy the Project URL and anon/public key
4. Copy the service_role key (keep this secret!)

## 🚀 Local Development

### 1. Install Dependencies

```bash npm install

# or

yarn install ```

### 2. Run Development Server

```bash npm run dev

# or

yarn dev ```

### 3. Build for Production

```bash npm run build

# or

yarn build ```

### 4. Test the Application

1. Visit `http://localhost:3000`
2. Test user registration and login
3. Verify profile creation and updates
4. Test image uploads to Supabase storage
5. Check all dashboard features
6. Verify 3D model loading
7. Test responsive design on mobile
8. Confirm email notifications work
9. Test outfit saving and loading

## 🌐 Production Deployment

### 1. Deploy to Vercel

**Option A: Deploy with Vercel CLI**

```bash

# Install Vercel CLI

npm i -g vercel

# Deploy

vercel

# Set environment variables

vercel env add NEXT_PUBLIC_SUPABASE_URL vercel env add NEXT_PUBLIC_SUPABASE_ANON_KEY

# ... add all other environment variables

```

Option B: Deploy via GitHub

1. Push code to GitHub repository
2. Connect repository to Vercel
3. Add environment variables in Vercel dashboard

4. Deploy automatically on push

2. Update Supabase Settings

1. Update Site URL to your production domain
2. Add production callback URLs
3. Update CORS settings if needed

3. Verify Production

1. Test user registration/login
2. Verify email confirmations work
3. Test profile updates
4. Check all features work correctly

🧪 Testing Checklist

Authentication Flow

- ☐ User can register with email/password
- ☐ Email confirmation works
- ☐ User can log in
- ☐ User can log out
- ☐ Protected routes work correctly

Profile Management

- ☐ Profile loads on login
- ☐ User can update name
- ☐ User can update address
- ☐ User can upload profile picture
- ☐ Changes persist after logout/login
- ☐ Error handling works properly

Dashboard Features

- ☐ Dashboard loads correctly
- ☐ All components render properly
- ☐ Real-time updates work
- ☐ Navigation works correctly

Image Uploads

- ☐ User can upload images to avatars bucket
- ☐ User can upload images to outfit-images bucket
- ☐ User can upload images to product-images bucket

3D Model Loading

- ☐ 3D models load correctly

- [ ] Real-time preview works

Shopping Cart and Checkout

- [ ] User can add items to shopping cart
- [ ] User can complete checkout flow

🔍 Troubleshooting

Common Issues

"Multiple rows returned" Error

- Run the duplicate cleanup script: `scripts/06-fix-duplicate-profiles.sql`
- Check for duplicate user profiles in the database

Authentication Not Working

- Verify environment variables are correct
- Check Supabase authentication settings
- Ensure callback URLs are properly configured

Profile Not Loading

- Check database permissions and RLS policies
- Verify the profiles table exists and has correct structure
- Check browser console for errors

Image Upload Issues

- Verify file size limits (2MB max)
- Check file type restrictions
- Ensure proper error handling

3D Model Issues

- Check model file paths
- Verify model loading scripts

Debug Steps

1. Check browser console for errors
2. Verify environment variables
3. Test database connection
4. Check Supabase logs
5. Verify RLS policies

📊 Performance Optimization

Database

- Indexes are created for frequently queried columns
- Connection pooling is configured
- RLS policies are optimized for performance

Frontend

- Components use React.memo where appropriate
- Images are optimized automatically
- Loading states prevent UI blocking

Image Optimization

- Use Next.js Image component for automatic optimization
- Configure Supabase storage for CDN delivery
- Implement lazy loading for 3D models

🔒 Security Considerations

Database Security

- Row Level Security (RLS) is enabled
- Users can only access their own data
- Service role key is kept secure

Frontend Security

- Input validation on all forms
- File upload restrictions
- XSS protection through React
- CSRF protection through Supabase
- HTTPS is enforced in production

📈 Monitoring and Analytics

Error Tracking

Consider adding error tracking services:

- Sentry for error monitoring
- Vercel Analytics for performance insights
- Supabase Dashboard for database monitoring

Performance Monitoring

- Monitor Core Web Vitals
- Track 3D model loading times
- Monitor database query performance

🔄 Updates and Maintenance

- Regular dependency updates
- Database maintenance and optimization
- Performance monitoring and improvements
- Security patches and updates
- Feature enhancements based on user feedback

🎯 Key Features Deployed

✅ **Complete User Authentication System** ✅ **Full Profile Management with Image Uploads** ✅ **3D Outfit Builder with Real-time Preview** ✅ **Responsive Dashboard with Analytics** ✅ **Shopping Cart and Checkout Flow** ✅ **Real-time Data Synchronization** ✅ **Mobile-Optimized Experience**

📞 Support

For deployment issues:

1. Check Vercel deployment logs
2. Monitor Supabase dashboard for errors
3. Review browser console for client-side issues
4. Verify all environment variables are set correctly

🔄 Updates and Maintenance

- Regular dependency updates
- Database maintenance and optimization
- Performance monitoring and improvements
- Security patches and updates
- Feature enhancements based on user feedback

Your Cosmic Outfits 3D Builder is now ready for production! 🚀