Staging Deployment Guide

1. Environment Setup

Vercel Project Setup

```
# Install Vercel CLI
npm i -g vercel

# Link project
vercel link

# Set environment variables
vercel env add DATABASE_URL
vercel env add NEXTAUTH_SECRET
vercel env add AWS_BUCKET_NAME
vercel env add AWS_ACCESS_KEY_ID
vercel env add AWS_SECRET_ACCESS_KEY
```

Required Environment Variables

```
# Database (Neon.tech staging)
DATABASE_URL="postgresql://username:password@staging-db.neon.tech/yalla_london_staging"

# Authentication
NEXTAUTH_SECRET="your-staging-nextauth-secret-32-chars-min"
NEXTAUTH_URL="https://yalla-london-staging.vercel.app"

# AWS S3 Staging Bucket
AWS_BUCKET_NAME="yalla-london-staging"
AWS_FOLDER_PREFIX="staging/"
AWS_ACCESS_KEY_ID="AKIA..."
AWS_SECRET_ACCESS_KEY="..."
AWS_REGION="us-east-1"

# Feature Flags
FEATURE_SEO=1
FEATURE_EMBEDS=1
FEATURE_MEDIA=1
FEATURE_HOMEPAGE_BUILDER=1

# AI Content Generation
ABACUSAI_API_KEY="your-abacus-ai-key"

# Analytics (Optional)
NEXT_PUBLIC_GOOGLE_ANALYTICS_ID="G-STAGING123"
```

2. Database Setup

Create Staging Database

1. Create Neon.tech staging project
2. Run migrations: `yarn prisma migrate deploy`

3. Generate client: `yarn prisma generate`
4. Seed data: `yarn tsx scripts/seed-staging.ts`

Backup Configuration

```
# Test backup creation
curl -X POST https://staging-url/api/database/backups \
  -H "Content-Type: application/json" \
  -d '{"backupName": "staging-test", "backupType": "manual"}'
```

3. S3 Staging Bucket Setup

IAM Policy (Least Privilege)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::yalla-london-staging/*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::yalla-london-staging"
    }
  ]
}
```

4. Deployment Commands

Deploy to Staging

```
# Deploy with environment
vercel --prod --env DATABASE_URL=@database_url_staging

# Run post-deployment seed
vercel env ls
yarn tsx scripts/seed-staging.ts
```

Validation Tests

```
# Health check
curl https://yalla-london-staging.vercel.app/api/health

# Feature validation
curl https://yalla-london-staging.vercel.app/api/social-embeds
curl https://yalla-london-staging.vercel.app/api/media
curl https://yalla-london-staging.vercel.app/api/homepage-blocks
```

5. Expected Staging URLs

- **Base:** https://yalla-london-staging.vercel.app
- **Admin:** https://yalla-london-staging.vercel.app/admin
- **API:** https://yalla-london-staging.vercel.app/api/*
- **Blog:** https://yalla-london-staging.vercel.app/blog
- **Arabic:** https://yalla-london-staging.vercel.app?lang=ar

6. Validation Checklist

Phase 3.2 - Social Embeds

- [ ] Admin can add Instagram/TikTok/YouTube/Facebook URLs
- [ ] Frontend shows CLS-safe embed cards
- [ ] Click loads official embed or modal
- [ ] GA4 events tracked (video_play, embed_open)
- [ ] VideoObject JSON-LD generated
- [ ] Usage analytics visible in admin

Phase 3.3 - Media Library

- [ ] File upload to S3 works
- [ ] WebP/AVIF variants generated
- [ ] Metadata editing (alt, title, tags)
- [ ] “Set as Hero” updates homepage
- [ ] Usage map shows asset locations
- [ ] Responsive srcset in HTML

Phase 3.4 - Homepage Builder

- [ ] Drag-drop block reordering
- [ ] Live preview works
- [ ] Publish/rollback functionality
- [ ] EN/AR language switching
- [ ] Lighthouse scores ≥ 0.9

Phase 3.5 - Database & Backups

- [ ] Manual backup creation
- [ ] Backup download from S3
- [ ] Database restore test
- [ ] Migration safety checks
- [ ] Daily backup scheduling

7. Phase 4 Readiness Verification

Content Automation Hooks

```
// Verify these APIs work in staging:
POST /api/social-embeds (programmatic embed creation)
POST /api/media/assign-automatically (AI image assignment)
POST /api/homepage-blocks/feed-content (automated content streams)
POST /api/content/schedule-batch (bulk content scheduling)
```

Schema Validation

```
-- Verify multi-tenant ready schema
SELECT column_name FROM information_schema.columns
WHERE table_name = 'BlogPost' AND column_name LIKE '%tenant%';

-- Verify automation queue support
SELECT table_name FROM information_schema.tables
WHERE table_name LIKE '%queue%';
```