Deployment Instructions

Quick Start

Admin Access

- **URL**: `/admin/login`
- **Email**: admin@cms.com
- **Password**: password123

Development Server

```Bash
php artisan serve
```

Then visit: http://localhost:8000

Production Deployment

1. Server Requirements

- PHP 8.1+
- Composer
- Web server (Apache/Nginx)
- Database (MySQL/PostgreSQL/SQLite)

2. Upload Files

Upload all project files to your web server's document root.

3. Environment Configuration

```Bash
cp .env.example .env
```

Edit `.env` file:

```Plain Text
```

```
APP_NAME="Your Site Name"
APP_ENV=production
APP_DEBUG=false
APP_URL=https://yourdomain.com

DB_CONNECTION=mysql
DB_HOST=localhost
DB_PORT=3306
DB_DATABASE=your_database
DB_USERNAME=your_username
DB_PASSWORD=your_password
```

4. Install Dependencies

Bash

```bash
composer install --optimize-autoloader --no-dev
```

5. Generate Application Key

Bash

```bash
php artisan key:generate
```

6. Database Setup

Bash

```bash
php artisan migrate
php artisan db:seed --class=AdminUserSeeder
php artisan db:seed --class=SampleDataSeeder
```

7. Storage Setup

Bash

```bash
php artisan storage:link
chmod -R 755 storage/
chmod -R 755 bootstrap/cache/
```

8. Optimize for Production

Bash

```
php artisan config:cache
php artisan route:cache
php artisan view:cache
```

9. Web Server Configuration

Apache (.htaccess)

Plain Text

```
<IfModule mod_rewrite.c>
    RewriteEngine On
    RewriteRule ^(.*)$ public/$1 [L]
</IfModule>
```

Nginx

Plain Text

```
server {
    listen 80;
    server_name yourdomain.com;
    root /path/to/your/project/public;

    add_header X-Frame-Options "SAMEORIGIN";
    add_header X-Content-Type-Options "nosniff";

    index index.php;

    charset utf-8;

    location / {
        try_files $uri $uri/ /index.php?$query_string;
    }

    location = /favicon.ico { access_log off; log_not_found off; }
    location = /robots.txt  { access_log off; log_not_found off; }

    error_page 404 /index.php;

    location ~ \.php$ {
        fastcgi_pass unix:/var/run/php/php8.1-fpm.sock;
```

```
        fastcgi_param SCRIPT_FILENAME $realpath_root$fastcgi_script_name;
        include fastcgi_params;
    }

    location ~ /\.(?!well-known).* {
        deny all;
    }
}
```

Database Migration from SQLite to MySQL

1. Export SQLite Data

Bash

```
sqlite3 database/database.sqlite .dump > backup.sql
```

2. Create MySQL Database

SQL

```
CREATE DATABASE your_database CHARACTER SET utf8mb4 COLLATE
utf8mb4_unicode_ci;
```

3. Update .env

Plain Text

```
DB_CONNECTION=mysql
DB_HOST=localhost
DB_DATABASE=your_database
DB_USERNAME=your_username
DB_PASSWORD=your_password
```

4. Run Migrations

Bash

```
php artisan migrate:fresh
php artisan db:seed --class=AdminUserSeeder
```

SSL Certificate Setup

Using Let's Encrypt (Certbot)

Bash

```bash
sudo apt install certbot python3-certbot-nginx
sudo certbot --nginx -d yourdomain.com
```

Update .env for HTTPS

Plain Text

```
APP_URL=https://yourdomain.com
```

Backup Strategy

Database Backup

Bash

```bash
# MySQL
mysqldump -u username -p database_name > backup.sql

# SQLite
cp database/database.sqlite backup/database_backup_$(date +%Y%m%d).sqlite
```

File Backup

Bash

```bash
tar -czf backup_$(date +%Y%m%d).tar.gz storage/app/public/
```

Monitoring and Maintenance

Log Files

- Laravel logs: `storage/logs/laravel.log`
- Web server logs: `/var/log/nginx/` or `/var/log/apache2/`

Regular Tasks

```bash
# Clear cache
php artisan cache:clear

# Clear logs (be careful!)
php artisan log:clear

# Update dependencies
composer update
```

Performance Monitoring

- Monitor disk space for uploads
- Check database size growth
- Monitor server resources

Security Checklist

- [ ] Set `APP_DEBUG=false` in production
- [ ] Use HTTPS with valid SSL certificate
- [ ] Set proper file permissions (755 for directories, 644 for files)
- [ ] Keep Laravel and dependencies updated
- [ ] Regular database backups
- [ ] Monitor access logs for suspicious activity
- [ ] Use strong passwords for admin accounts

Troubleshooting

Common Issues

1. **500 Internal Server Error**
   - Check Laravel logs
   - Verify file permissions
   - Ensure .env file exists

2. **Storage Link Issues**
3. **Database Connection Error**
    - Verify database credentials
    - Check database server status
    - Ensure database exists
4. **File Upload Issues**
    - Check PHP upload limits
    - Verify storage permissions
    - Ensure sufficient disk space

Debug Mode (Development Only)

```
Plain Text
```

```
APP_DEBUG=true
```

Support

For technical support or questions:

1. Check the documentation
2. Review Laravel logs
3. Contact the development team

Important: Always test deployments on a staging environment before deploying to production.