Webs Framework - cPanel/Namecheap Deployment Guide

Quick Start for cPanel Shared Hosting

Step 1: Upload Your Files

1. **Upload the framework file** (`webs.py`) to your domain's root directory (usually `public_html`)
2. **Create your application** (`app.py`):

```python
```

```python
#!/usr/bin/env python3
from webs import webs, Model, Field, render_template

# Create app instance
app = webs(__name__)

# Configure for production
app.config.update({
    'SECRET_KEY': 'change-this-to-a-random-secret-key',
    'DEBUG': False,  # Always False in production
    'DATABASE_URL': 'sqlite:///production.db'
})

# Example model
class User(Model):
    username = Field(str, nullable=False, unique=True)
    email = Field(str, nullable=False, unique=True)
    role = Field(str, default='user')

# Routes
@app.route('/')
def home(request):
    return '''
    <h1>Welcome to Webs Framework!</h1>
    <p>Your application is running successfully on cPanel.</p>
    <a href="/test">Test Page</a>
    '''

@app.route('/test')
def test(request):
    return {
        'message': 'API is working!',
        'framework': 'Webs Production Ready',
        'deployment': 'cPanel Success'
    }

# Freeze ORM for production
app.freeze_orm()

# This is required for Passenger
application = app.wsgi
```

```
if __name__ == '__main__':
    app.run()
```

Step 2: Create Required Files

The framework will auto-generate these files, but here they are manually:

`passenger_wsgi.py` (Required for cPanel):

```python
#!/usr/bin/env python3
import sys
import os

# Add your project directory to Python path
sys.path.insert(0, os.path.dirname(__file__))

# Import your application
from app import app

# Create WSGI application for Passenger
application = app.wsgi
```

`.htaccess` (For subdirectory deployment):

```apache
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule ^(.*)$ passenger_wsgi.py/$1 [QSA,L]
```

`requirements.txt` (Empty - zero dependencies!):

```
# Zero dependencies required!
# This framework runs on pure Python 3.7+
```

Step 3: Set Permissions

In cPanel File Manager, set permissions:

- `passenger_wsgi.py`: 755
- `app.py`: 644

- `webs.py`: 644
- `.htaccess`: 644

Step 4: Configure Python App in cPanel

1. Go to **"Python Apps"** or **"Setup Python App"** in cPanel
2. Click **"Create Application"**
3. Set:
   - **Python Version**: 3.7 or higher
   - **Application Root**: `/` (or your subdomain folder)
   - **Application URL**: Your domain or subdomain
   - **Application Startup File**: `passenger_wsgi.py`
   - **Application Entry Point**: `application`

Step 5: Test Your Application

Visit your domain - you should see "Welcome to Webs Framework!"

Advanced Production Setup

Database Configuration

For SQLite (recommended for shared hosting):

```python
app.config['DATABASE_URL'] = 'sqlite:///production.db'
```

The database file will be created automatically in your application directory.

Environment Variables

Create a `.env` file (not web-accessible):

```bash
SECRET_KEY=your-super-secret-key-here
DEBUG=False
DATABASE_URL=sqlite:///production.db
```

Then load in your app:

```python
import os
from webs import webs

app = webs(__name__)

# Load from environment or default
app.config.update({
    'SECRET_KEY': os.environ.get('SECRET_KEY', 'default-secret'),
    'DEBUG': os.environ.get('DEBUG', 'False').lower() == 'true',
    'DATABASE_URL': os.environ.get('DATABASE_URL', 'sqlite:///production.db')
})
```

Security Configuration

```python
from webs import SecurityHeaders

# Add security middleware
@app.after_request
def add_security_headers(request, response):
    return SecurityHeaders.add_security_headers(request, response)

# Enable CSRF protection for forms
from webs import csrf_protect

@app.route('/form', methods=['POST'])
@csrf_protect
def handle_form(request):
    return "Form processed securely!"
```

Session Management

For production, use database sessions:

```python
```

```python
from webs import DatabaseSessionStore

# Use database for sessions (more reliable than memory)
if not app.config.get('DEBUG'):
    session_store = DatabaseSessionStore(app._get_db_pool(app))
    app.use_session_store(session_store)
```

Complete Production Application Example

python

```python
#!/usr/bin/env python3
from webs import (
    webs, Model, Field, Response,
    login_required, roles_required, login_user, logout_user, current_user,
    SecurityHeaders, CORSMiddleware, csrf_protect
)
import os

# Create app
app = webs(__name__)

# Production configuration
app.config.update({
    'SECRET_KEY': os.environ.get('SECRET_KEY', 'change-this-in-production'),
    'DEBUG': False,
    'DATABASE_URL': 'sqlite:///production.db'
})

# Add monitoring
app.add_monitoring()

# Add CORS support
app.middleware.append(CORSMiddleware(
    allowed_origins=['https://yourdomain.com'],
    allow_credentials=True
))

# Security headers
@app.after_request
def add_security(request, response):
    return SecurityHeaders.add_security_headers(request, response)

# Models
class User(Model):
    username = Field(str, nullable=False, unique=True)
    email = Field(str, nullable=False, unique=True)
    password_hash = Field(str, nullable=False)
    role = Field(str, default='user')
    created_at = Field(str, default=lambda: str(time.time()))

class Post(Model):
    title = Field(str, nullable=False)
    content = Field(str)
```

```
    author_id = Field(int, nullable=False)
    created_at = Field(str, default=lambda: str(time.time()))

# Routes
@app.route('/')
def home(request):
    user = current_user(request)
    posts = Post.all(app)[:10]  # Latest 10 posts

    return f'''
    <!DOCTYPE html>
    <html>
    <head>
        <title>My Webs App</title>
        <link rel="stylesheet" href="/static/w
```