Accompany

INSY7315

MO4 & MO5:

Website Systems Manual

Group Members:

Abdul Baari Davids - ST10267411
Ethan Smyth - ST10255309
Ethan Donaldson - ST10318621
Jacques du Plessis - ST10329686
Joshua Wood (PM) - ST102961671

Table of Contents

1. System Overview

1.1 Purpose

The ZATech Public Website serves as the primary marketing and information portal for South Africa's largest tech community. It provides:

- **Community Information**: About ZATech, mission, values
- **Membership Details**: How to join, community guidelines
- **Sponsorship Information**: Sponsor tiers and benefits
- **Social Links**: Connect to Slack, social media platforms

1.2 Key Components

Component	Purpose	Technology
Frontend	User interface	React 19, Vite
Build System	Bundle and optimize	Vite build pipeline
CDN	Global content delivery	Cloudflare Pages
CI/CD	Automated deployments	GitHub Actions
Testing	Quality assurance	Vitest (unit), Playwright (E2E)
Security	CSP, HSTS headers	Browser security policies

1.3 Infrastructure

Hosting: Cloudflare Pages (Free Tier)

Performance Characteristics:

- **Global Latency**: 10-30ms (edge cache hit)
- **Bandwidth**: Unlimited
- **Requests**: Unlimited
- **SSL**: Automatic (Let's Encrypt)
- **Uptime SLA**: 100%

Build Specifications:

- **Build Time**: ~30-60 seconds
- **Bundle Size**: ~300-500 KB (gzipped)
- **Build Concurrency**: 1 concurrent build (Free tier)

2. Deployment & Hosting

2.1 Cloudflare Pages Setup

Initial Setup (One-Time)

1. **Create Cloudflare Account**:
   - Visit https://dash.cloudflare.com/sign-up
   - Verify email address
2. **Connect GitHub Repository**:
   - Navigate to **Pages** in Cloudflare Dashboard
   - Click **Create a project**
   - Select **Connect to Git**
   - Authorize Cloudflare to access GitHub
   - Select repository: `zatech-website`
3. **Configure Build Settings**:

Setting	Value
Project name	`zatech-website`
Production branch	`main`
Build command	`npm run build`
Build output directory	`dist`
Root directory	`/` (root)

4. **Environment Variables**:
   - None required for production build

5. **Deploy**:

    - Click **Save and Deploy**
    - First build will start automatically

Custom Domain Setup

1. **Add Domain to Cloudflare**:

    - Navigate to **Add site**
    - Enter domain: `zatech.co.za`
    - Select **Free plan**
    - Update nameservers at domain registrar

2. **Configure Pages Custom Domain**:

    - Pages → `zatech-website` → **Custom domains**
    - Click **Set up a custom domain**
    - Enter: `zatech.co.za`
    - DNS records created automatically

3. **SSL Configuration**:

    - SSL/TLS → **Overview**
    - Mode: **Full (strict)**
    - Universal SSL: Enabled (automatic)

2.2 DNS Configuration

DNS Records (managed by Cloudflare):

Type	Name	Target	Proxy	TTL
`CNAME`	`zatech.co.za`	`zatech-website.pages.dev`	✅ Proxied	Auto
`CNAME`	`www`	`zatech.co.za`	✅ Proxied	Auto

Verification:

```
# Check DNS propagation
dig zatech.co.za
dig www.zatech.co.za

# Expected: CNAME pointing to Cloudflare Pages
```

2.3 Deployment Workflow

Automatic Deployment (Production)

Trigger: Push to `main` branch

```
# 1. Make changes locally
git checkout main
git pull origin main

# 2. Make changes to code
# ... edit files ...

# 3. Test locally
npm run dev
npm run test:run
npm run test:e2e

# 4. Commit and push
git add .
git commit -m "Update hero section copy"
git push origin main

# 5. Cloudflare Pages automatically:
#    - Detects push to main
#    - Runs build: npm install && npm run build
#    - Deploys to production: zatech.co.za
#    - Purges CDN cache
```

Build Log Access:

- Cloudflare Dashboard → Pages → `zatech-website` → **Deployments**
- View build logs, build time, deployment status

Preview Deployments (Pull Requests)

Trigger: Open pull request

```
# 1. Create feature branch
git checkout -b feature/update-sponsorship

# 2. Make changes and commit
git add .
git commit -m "Add new sponsor tier"
git push origin feature/update-sponsorship

# 3. Create pull request on GitHub

# 4. Cloudflare Pages automatically:
#    - Builds preview deployment
#    - Posts preview URL in PR comments
#    - Example: https://abc123.zatech-website.pages.dev
```

Preview URLs:

- Unique URL per PR
- Auto-updates on new commits
- Deleted when PR is closed/merged

2.4 Manual Deployment (Wrangler CLI)

Setup

```
# Install Wrangler
npm install -g wrangler

# Login to Cloudflare
wrangler login
```

Deploy

```
# Build locally
npm run build

# Deploy to Cloudflare Pages
wrangler pages deploy dist --project-name=zatech-website
```

---

3. Configuration Management

3.1 Environment Variables

The website runs as a **static site** with no runtime environment variables. All configuration is baked into the build.

Build-Time Configuration (`vite.config.js`):

```
export default {
  base: '/',
  build: {
    outDir: 'dist',
    assetsDir: 'assets',
    minify: 'terser',
    sourcemap: false,
  },
  server: {
    port: 5173,
  },
}
```

3.2 Content Configuration

Site Metadata (`src/config/site.js`):

```
export const siteConfig = {
  name: 'ZATech',
  description: 'South Africa\'s Largest Tech Community',
  url: 'https://zatech.co.za',
  social: {
    slack: 'https://zatech.slack.com',
    twitter: 'https://twitter.com/zatechza',
    linkedin: 'https://linkedin.com/company/zatech',
  },
}
```

3.3 Security Configuration

Content Security Policy (`src/config/csp.js`):

```
export const cspConfig = {
  'default-src': ["'self'"],
  'script-src': ["'self'", "'unsafe-inline'", 'cdn.example.com'],
  'style-src': ["'self'", "'unsafe-inline'"],
  'img-src': ["'self'", 'data:', 'https:'],
  'font-src': ["'self'", 'data:'],
  'connect-src': ["'self'"],
  'frame-ancestors': ["'none'"],
}
```

Security Headers (Cloudflare Pages Settings):

Headers are configured via `_headers` file in `public/` directory:

```
/*
  X-Frame-Options: DENY
  X-Content-Type-Options: nosniff
  Referrer-Policy: strict-origin-when-cross-origin
  Permissions-Policy: geolocation=(), microphone=(), camera=()
  Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
```

3.4 Caching Configuration

Cache Rules (Cloudflare Page Rules):

Asset Type	Cache TTL	Browser Cache
HTML (`/`, `/*.html`)	1 hour	5 minutes
CSS/JS (`/assets/*`)	1 year	1 year
Images (`/images/*`)	1 month	1 month
Fonts	1 year	1 year

Cache-Control Headers (`_headers`):

```
# HTML - short cache
/*.html
  Cache-Control: public, max-age=300, s-maxage=3600

# Assets with hash - long cache
/assets/*
  Cache-Control: public, max-age=31536000, immutable

# Images
/images/*
  Cache-Control: public, max-age=2592000
```

---

4. Build & Release Process

4.1 Development Build

Local Development Server:

```
# Start dev server with hot reload
npm run dev

# Server runs at: http://localhost:5173
# Hot reload: Automatic on file changes
```

Access from Mobile Device:

```
# Start server on all interfaces
npm run dev -- --host 0.0.0.0

# Access from phone: http://<your-ip>:5173
# Find IP: ifconfig (macOS/Linux) or ipconfig (Windows)
```

4.2 Production Build

Build Command:

```
# Clean build
rm -rf dist node_modules
npm install
npm run build

# Output: dist/ directory
```

Build Artifacts:

```
dist/
├── index.html              # Main HTML
├── assets/
│   ├── index-a1b2c3.js    # JavaScript (with hash)
│   ├── index-d4e5f6.css   # Styles (with hash)
│   └── logo-g7h8i9.svg    # Images (with hash)
├── images/                 # Static images
└── _headers               # Cloudflare headers
```

Build Optimization:

- **Minification**: Terser for JS, cssnano for CSS
- **Tree Shaking**: Remove unused code
- **Code Splitting**: Separate vendor bundles
- **Asset Hashing**: Cache busting with content hashes

4.3 Quality Gates

Pre-Deployment Checks:

```
# 1. Linting
npm run lint
# Expected: 0 errors, 0 warnings

# 2. Unit tests
npm run test:run
# Expected: All tests pass

# 3. E2E tests
npm run test:e2e
# Expected: All journeys pass

# 4. Security audit
npm audit
# Expected: 0 vulnerabilities

# 5. Build verification
npm run build
# Expected: Clean build, no errors
```

4.4 Release Checklist

- ☐ All tests passing (unit + E2E)
- ☐ No linting errors
- ☐ No security vulnerabilities
- ☐ Updated changelog (if applicable)
- ☐ Reviewed PR (if applicable)
- ☐ Tested on Chrome, Firefox, Safari
- ☐ Tested on mobile (iOS + Android)
- ☐ Performance metrics acceptable (Lighthouse > 90)

---

5. Monitoring & Performance

5.1 Performance Metrics

Target Metrics (Lighthouse):

Metric	Target	Acceptable	Poor
Performance	> 95	> 90	< 90
Accessibility	> 95	> 90	< 90
Best Practices	100	> 95	< 95
SEO	100	> 95	< 95

Core Web Vitals:

Metric	Target	Acceptable
LCP (Largest Contentful Paint)	< 2.5s	< 4s
FID (First Input Delay)	< 100ms	< 300ms
CLS (Cumulative Layout Shift)	< 0.1	< 0.25

5.2 Monitoring Tools

Cloudflare Analytics (Built-In)

Access: Cloudflare Dashboard → Analytics

Metrics Available:

- **Requests**: Total requests, requests per second
- **Bandwidth**: Data transfer volume
- **Cache Hit Rate**: % of requests served from edge
- **Status Codes**: 200, 404, 500, etc.
- **Top Countries**: Geographic distribution
- **Top URLs**: Most requested pages

5.3 Uptime Monitoring

UptimeRobot (Free)

Setup:

1. Create account: https://uptimerobot.com
2. Add HTTP(S) monitor:
    - **URL**: `https://zatech.co.za`
    - **Interval**: 5 minutes
    - **Alert Contacts**: Email, SMS

Alert Thresholds:

- **Down**: Fails 2 consecutive checks (10 minutes)
- **Slow**: Response time > 3 seconds

6. Content Updates

6.1 Text Content Updates

Simple Text Changes

```
# 1. Clone repository (if not already cloned)
git clone https://github.com/zatech/zatech-website.git
cd zatech-website

# 2. Create feature branch
git checkout -b content/update-about-page

# 3. Edit content files
nano src/pages/About.jsx

# 4. Preview changes
npm run dev

# 5. Commit and push
git add .
git commit -m "Update About page mission statement"
git push origin content/update-about-page

# 6. Create pull request on GitHub
# 7. Review preview deployment
# 8. Merge to main → auto-deploys to production
```

Content Components

Hero Section: `src/components/ui/HeroSection.jsx` **About Page**: `src/pages/About.jsx` **Sponsorship Page**: `src/pages/Sponsorship.jsx`

6.2 Image Updates

Adding New Images

```
# 1. Add image to assets
cp new-sponsor-logo.png public/images/sponsors/

# 2. Optimize image (recommended)
# Use ImageOptim (Mac) or TinyPNG (web)

# 3. Reference in code
<img src="/images/sponsors/new-sponsor-logo.png" alt="Sponsor Name" />

# 4. Commit and deploy
git add public/images/sponsors/new-sponsor-logo.png
git commit -m "Add new sponsor logo"
git push origin main
```

Image Optimization Guidelines

Image Type	Format	Max Size	Dimensions
Hero images	WebP/JPEG	< 200 KB	1920x1080
Logos	SVG/PNG	< 50 KB	200x200
Icons	SVG	< 10 KB	32x32
Photos	WebP/JPEG	< 150 KB	1200x800

6.3 Navigation Updates

Edit Menu: `src/components/common/Navbar.jsx`

```
const menuItems = [
  { label: 'Home', path: '/' },
  { label: 'About', path: '/about' },
  { label: 'Sponsorship', path: '/sponsorship' },
  { label: 'Events', path: '/events' }, // New item
]
```

6.4 SEO Metadata Updates

Update Meta Tags: `index.html`

```
<head>
  <title>ZATech - South Africa's Largest Tech Community</title>
  <meta name="description" content="Join 18,000+ tech professionals in
South Africa's premier tech community">
  <meta property="og:title" content="ZATech Community">
  <meta property="og:description" content="South Africa's largest tech
community">
  <meta property="og:image"
content="https://zatech.co.za/images/og-image.png">
</head>
```

7. Security Operations

7.1 Security Headers

Configured Headers (`public/_headers`):

```
/*
  # HSTS - Force HTTPS
  Strict-Transport-Security: max-age=31536000; includeSubDomains; preload

  # Prevent clickjacking
  X-Frame-Options: DENY

  # Prevent MIME sniffing
  X-Content-Type-Options: nosniff

  # Referrer policy
  Referrer-Policy: strict-origin-when-cross-origin

  # Permissions policy
  Permissions-Policy: geolocation=(), microphone=(), camera=()

  # CSP - Content Security Policy
  Content-Security-Policy: default-src 'self'; script-src 'self'
'unsafe-inline'; style-src 'self' 'unsafe-inline'; img-src 'self' data:
https:
```

Verify Headers:

```
curl -I https://zatech.co.za | grep -i "security\|frame\|content"
```

7.2 Dependency Security

Automated Vulnerability Scanning

GitHub Dependabot (Enabled):

- Automatically scans for vulnerabilities
- Creates PRs for security updates
- Weekly security checks

Manual Audit:

```
# Check for vulnerabilities
npm audit

# Fix automatically (if possible)
npm audit fix

# Fix with breaking changes
npm audit fix --force
```

Security Update Workflow:

1. Dependabot creates PR with security fix
2. Review changes in PR
3. Check preview deployment
4. Run tests: `npm run test:run && npm run test:e2e`
5. Merge if tests pass

7.3 SSL/TLS Configuration

Cloudflare SSL Settings:

- **Mode**: Full (Strict) - End-to-end encryption
- **Minimum TLS Version**: TLS 1.2
- **Opportunistic Encryption**: Enabled
- **TLS 1.3**: Enabled
- **Automatic HTTPS Rewrites**: Enabled

Certificate Details:

- **Provider**: Let's Encrypt (via Cloudflare)
- **Type**: Universal SSL
- **Validity**: 90 days (auto-renewed)
- **Coverage**: `zatech.co.za`, `www.zatech.co.za`

7.4 DDoS Protection

Cloudflare DDoS Protection (Automatic):

- **Layer 3/4 Protection**: Network and transport layer attacks
- **Layer 7 Protection**: HTTP flood attacks
- **Rate Limiting**: Automatic throttling of suspicious traffic
- **Bot Management**: Block malicious bots (Free tier has basic protection)

Custom Rate Limiting (Optional - Paid feature):

Configure in Cloudflare Dashboard → Security → WAF → Rate limiting rules

7.5 Web Application Firewall (WAF)

Cloudflare WAF (Free tier includes basic rules):

Enabled Protections:

- ✅ OWASP Top 10 vulnerabilities
- ✅ SQL injection attempts
- ✅ XSS (Cross-Site Scripting) attacks
- ✅ Known malicious user agents

Managed Rules:

- Cloudflare Managed Ruleset: Enabled
- OWASP ModSecurity Core Rule Set: Enabled (partial on Free tier)

8. Testing Procedures

8.1 Unit Tests

Technology: Vitest

Run Tests:

```
# Run all unit tests
npm run test:run

# Run with coverage
npm run test:coverage

# Watch mode (during development)
npm run test
```

Test Coverage Targets:

- **Statements**: > 80%
- **Branches**: > 75%
- **Functions**: > 80%
- **Lines**: > 80%

Example Test (`src/components/HeroSection.test.jsx`):

```
import { render, screen } from '@testing-library/react'
import HeroSection from './HeroSection'

test('renders hero heading', () => {
  render(<HeroSection />)
  expect(screen.getByText(/ZATech/i)).toBeInTheDocument()
})
```

8.2 End-to-End Tests

Technology: Playwright (cross-browser)

Run E2E Tests:

```
# Install browsers (first time only)
npx playwright install

# Run E2E tests (headless)
npm run test:e2e

# Run with UI (debug mode)
npx playwright test --ui

# Run specific browser
npx playwright test --project=chromium
```

Test Coverage:

User Journey	Test File	Browsers
Homepage load	`tests/e2e/homepage.spec.js`	Chrome, Firefox, Safari
Navigation	`tests/e2e/navigation.spec.js`	Chrome, Firefox, Safari
Responsive design	`tests/e2e/responsive.spec.js`	Chrome Mobile, Safari Mobile
Forms	`tests/e2e/forms.spec.js`	Chrome, Firefox

Example E2E Test:

```
// tests/e2e/homepage.spec.js
import { test, expect } from '@playwright/test'

test('homepage loads and displays hero', async ({ page }) => {
  await page.goto('/')
  await expect(page.locator('h1')).toContainText('ZATech')
})
```

8.3 Manual Testing

Pre-Release Testing Checklist:

Desktop Browsers:

- ☐ Chrome (latest)
- ☐ Firefox (latest)
- ☐ Safari (latest) - macOS only
- ☐ Edge (latest)

Mobile Browsers:

- ☐ Chrome Mobile (Android)
- ☐ Safari Mobile (iOS)

Test Cases:

- ☐ Homepage loads correctly
- ☐ All navigation links work
- ☐ Images load properly
- ☐ Forms submit successfully
- ☐ Responsive design works (mobile, tablet, desktop)
- ☐ No console errors
- ☐ No CSP violations

8.4 Performance Testing

Lighthouse Audit:

WebPageTest (https://www.webpagetest.org):

1. Enter URL: `https://zatech.co.za`
2. Select location: Johannesburg, South Africa
3. Run test
4. Review:
    - **First Contentful Paint**: < 1.5s
    - **Largest Contentful Paint**: < 2.5s
    - **Total Blocking Time**: < 300ms