Quickcart – End-to-End Deployment SOP / SMP

Version: 1.0
Date: 04-Nov-2025
Owner: Noothan S
Systems: Azure App Service (Linux), GitHub Actions, Node.js, Vite/React, Prisma, Stripe

---

1) Purpose

This SOP documents the **complete** setup, deployment, and operations for the Quickcart application: **backend API (Node/Express + Prisma)** and **frontend SPA (Vite/React)**. It includes local/dev procedures, CI/CD, configuration, and a detailed problem log with root causes and fixes.

---

2) Canonical URLs & Identities (Final)

- **Backend (prod/dev same app):** `https://my-backend-app123.azurewebsites.net`
- **Frontend (prod/dev same app):** `https://my-frontend-app123.azurewebsites.net`
- **Azure Subscription / RG:** Azure subscription 1 / `RG-CTB45-AZ-400-2`
- **App Service Plans:**
- Backend: (noted in portal) Linux
- Frontend: `frontend-plan (F1: 1)` Linux

---

3) Secrets / Keys

Stored in the relevant places below. **Stripe key intentionally included** (per request).

- **Stripe (Publishable) – Frontend:**
`pk_test_51OVWkVHFRFCfVZp8dwEwzuDWjUpgnMadf3sHxxb8mf3xza bCF3m5KEDKLHbGOa fbOvQFi2NlokV0TB0wAQDaLDt500R0LHclLj`

- **Stripe (Secret)** – *not used by frontend; keep server-side only if ever introduced*.

- **Azure publish profiles** – stored as **GitHub Secrets**:
`AZUREAPPSERVICE_PUBLISHPROFILE_3EB3453C96564A268C3FAB990081303A` (backend)
`AZUREAPPSERVICE_PUBLISHPROFILE_02DD8D6C42AD4230A2FB3FE628E7ACA8` (frontend)

---

4) High-Level Architecture

- **Frontend:** Vite/React SPA hosted on **Azure App Service (Linux)** as static content, served by **pm2** with SPA fallback.
- **Backend:** Node.js/Express + Prisma hosted on **Azure App Service (Linux)**.
- **Browser → Frontend** (static) → calls **Backend API** over HTTPS.
- **Stripe** used by frontend via publishable key.

---

5) Repository Layout (root)

```
Noothan-CT-quick-cart-shop/
├─ .github/workflows/
│   ├─ main_my-backend-app123.yml
│   └─ main_my-frontend-app123.yml
├─ backend/
│   ├─ src/ ... (Express app)
│   ├─ prisma/
│   ├─ dist/ (built)
│   ├─ package.json, tsconfig.json, .env(.example)
│   └─ vercel.json (legacy, not used in Azure)
└─ frontend/
    ├─ src/ ... (React)
    ├─ public/
    │   └─ web.config (NOT used on Linux; we now generate
`staticwebapp.config.json` at build)
    ├─ dist/ (built)
    ├─ package.json, tsconfig.*.json, vite.config.ts
    └─ .env
```

---

6) Local Development

6.1 Backend (Local)

Prereqs: Node 20.x, npm, SQLite/Postgres (as per prisma schema), Prisma CLI.

```
cd backend
npm ci
npx prisma generate
npm run build   # tsc
```

```
node dist/index.js
# or add an npm script: "start": "node dist/index.js"
```

Health check / smoke test - Open `http://localhost:5000` (or the logged PORT). Should respond with **Hello World!** (initial route) or API health.

Common local issues & fixes - **EPERM rename (Windows) during** `prisma generate` **:** - Cause: Windows file locking of `query_engine-windows.dll.node` in `node_modules/.prisma/client`. - Fix:

```
# Close Node processes
taskkill /F /IM node.exe /T
# Clean and reinstall
rm -r node_modules  # PowerShell: Remove-Item -Recurse -Force node_modules
npm cache clean --force
npm install
npx prisma generate
npm run build
```

- **Prisma versions mismatch** (warning): align versions:

```
npm i -D prisma@latest
npm i @prisma/client@latest
npx prisma generate
```

6.2 Frontend (Local)

Prereqs: Node 20.x, npm

Config file: `frontend/src/config/index.ts`

```
export const config = {
  server_url: "https://my-backend-app123.azurewebsites.net",
  stripe_publishable_key:
"pk_test_51OVWkVHFRFCfVZp8dwEwzuDWjUpgnMadf3sHxxb8mf3xza bCF3m5KEDKLHbGOa
fbOvQFi2NlokV0TBOwAQDaLDt5OOR0LHclLj",
};
```

> **Note:** We fixed type errors by ensuring `stripe_publishable_key` exists on the exported `config` object used by `stripe-payment.tsx`.

Run locally:

```
cd frontend
npm ci
npm run dev  # Vite dev server
# Build and serve the production bundle locally
npm run build
npx serve -s dist
```

Smoke test: open `http://localhost:3000` → site loads; banner carousel works.

---

7) Backend – Azure Dev (App Service Linux)

7.1 App Service configuration

- **Runtime:** Node 20 LTS, Linux
- **PORT:** App Service provides `PORT` env var. The Express app must use `process.env.PORT || 8080`.
- **CORS:** In code, allow the **frontend URL**.
- **Final allowed origin:** `https://my-frontend-app123.azurewebsites.net`

Example CORS snippet (backend `src/index.ts`):

```
import cors from "cors";

app.use(
  cors({
    origin: [
      "https://my-frontend-app123.azurewebsites.net",
      "http://localhost:5173"  // local dev
    ],
    credentials: true,
    methods: ["GET","POST","PUT","PATCH","DELETE","OPTIONS"],
    allowedHeaders: ["Content-Type","Authorization"]
  })
);
```

7.2 GitHub Actions (backend)

`.github/workflows/main_my-backend-app123.yml`

```
name: Build and deploy Node.js backend to Azure Web App - my-backend-app123

on:
```

```
  push:
    branches: [ main ]
  workflow_dispatch:

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - uses: actions/setup-node@v3
        with:
          node-version: '20.x'
      - name: Install deps, generate Prisma, build
        working-directory: backend
        run: |
          npm ci
          npx prisma generate
          npm run build
      - name: Package app for deploy
        run: |
          mkdir pkg && cp -r backend/dist backend/package*.json backend/prisma
pkg/
          # (Optional) include node_modules if you prefer no post-deploy install
          # cp -r backend/node_modules pkg/
          cd pkg && zip -r ../backend.zip .
      - uses: actions/upload-artifact@v4
        with:
          name: backend-zip
          path: backend.zip

  deploy:
    runs-on: ubuntu-latest
    needs: build
    steps:
      - uses: actions/download-artifact@v4
        with:
          name: backend-zip
      - name: Deploy to Azure Web App
        uses: azure/webapps-deploy@v3
        with:
          app-name: my-backend-app123
          publish-profile: $
{{ secrets.AZUREAPPSERVICE_PUBLISHPROFILE_3EB3453C96564A268C3FAB990081303A }}
          package: backend.zip
```

✅ After fixing CORS and deploying, `https://my-backend-app123.azurewebsites.net/` returned **Hello World!** and API started serving.

---

8) Frontend – Azure Dev (App Service Linux)

8.1 Why the first deploy showed *"Your web app is running and waiting for your content"*

- The App Service default `hostingstart.html` was taking precedence because either the **dist** content was not deployed correctly or the platform didn't know to treat this as a SPA.
- On Linux, `web.config` **is ignored** (it's for IIS/Windows). We must use one of: 1) A **staticwebapp.config.json** with SPA fallback; and/or 2) A **startup command** to serve SPA via **pm2**.

8.2 Final working approach

- **Generate SPA routing config** during CI: `staticwebapp.config.json` inside `dist`:

```
{
  "navigationFallback": {
    "rewrite": "/index.html",
    "exclude": ["/assets/*", "favicon.ico"]
  }
}
```

- **Startup command (Portal → App Service → Configuration (preview) → Stack settings → Startup command):**

```
pm2 serve /home/site/wwwroot --no-daemon --spa
```

- Ensures SPA fallback and static hosting from `/home/site/wwwroot`.

8.3 GitHub Actions (frontend)

`.github/workflows/main_my-frontend-app123.yml`

```
name: Build and deploy Vite React app to Azure Web App - my-frontend-app123

on:
  push:
    branches: [ main ]
  workflow_dispatch:

jobs:
  build:
```

```
    runs-on: ubuntu-latest
    steps:
      - name: Checkout repository
        uses: actions/checkout@v4
      - name: Set up Node.js
        uses: actions/setup-node@v3
        with:
          node-version: '20.x'
      - name: Install dependencies and build
        working-directory: frontend
        run: |
          npm ci
          npm run build
      - name: Create routing config (Linux)
        working-directory: frontend/dist
        run: |
          echo '{
            "navigationFallback": {"rewrite": "/index.html", "exclude": ["/
assets/*", "favicon.ico"]}
          }' > staticwebapp.config.json
      - name: Zip build output
        run: |
          cd frontend/dist
          zip -r ../../frontend.zip .
      - name: Upload artifact
        uses: actions/upload-artifact@v4
        with:
          name: react-build
          path: frontend.zip

  deploy:
    runs-on: ubuntu-latest
    needs: build
    steps:
      - name: Download artifact
        uses: actions/download-artifact@v4
        with:
          name: react-build
      - name: Deploy to Azure Web App (Linux)
        uses: azure/webapps-deploy@v3
        with:
          app-name: 'my-frontend-app123'
          publish-profile: $
{{ secrets.AZUREAPPSERVICE_PUBLISHPROFILE_02DD8D6C42AD4230A2FB3FE628E7ACA8 }}
          package: frontend.zip
```

8.4 Validation / Smoke tests

- Visit `https://my-frontend-app123.azurewebsites.net/` → **Home page renders**.
- Browser **Network** tab loads bundle `index-*.js`, CSS, and images from `/assets/*`.
- API calls hit `https://my-backend-app123.azurewebsites.net` and succeed (CORS OK).

---

9) Problem Log (Issues, Root Cause, Fix)

9.1 Frontend showed *"Your web app is running and waiting for your content"*

- **Cause:** Deployed content didn't override the default placeholder or SPA routing not configured (Linux ignores `web.config`).
- **Fix:** Ensure `dist` contents deployed to `/home/site/wwwroot`, create `staticwebapp.config.json`, and add pm2 startup command.

9.2 503 Service Unavailable & App Service "Application Error" placeholders

- **Cause:** Broken/partial deploys while experimenting; the site served Azure placeholder pages.
- **Fix:** Redeploy clean artifacts; confirm via Kudu SSH that `/home/site/wwwroot` contains **index.html, assets, staticwebapp.config.json**; restart app.

9.3 Kudu SSH appeared "blank"

- **Cause:** WebSSH opens with a blank prompt until you run a command.
- **Fix:** Use the prompt or switch to **Bash** tab and run `ls`, `cd /home/site/wwwroot`, etc.

9.4 Prisma `EPERM: operation not permitted, rename ... query_engine-windows.dll.node.tmp...` (Windows local)

- **Cause:** Windows lock on prisma engine during generation.
- **Fix:** Kill Node, delete `node_modules`, clean cache, reinstall, `npx prisma generate`, then `npm run build`.

9.5 Prisma versions mismatch (warning)

- **Cause:** `prisma` and `@prisma/client` versions out of sync.
- **Fix:** `npm i -D prisma@latest && npm i @prisma/client@latest`.

9.6 Frontend build error: `Property 'stripe_publishable_key' does not exist on type { server_url: string; }`

- **Cause:** Type shape of `config` object didn't include `stripe_publishable_key`.
- **Fix:** Update `frontend/src/config/index.ts` to export both `server_url` and `stripe_publishable_key`.

9.7 CORS errors from browser

- **Cause:** Backend CORS didn't include the new frontend origin.
- **Fix:** Add `https://my-frontend-app123.azurewebsites.net` to `cors({ origin: [...] })`.

9.8 GitHub Actions error: `No package found with specified pattern: react-build`

- **Cause:** Artifact name / path mismatch between **upload** and **download** steps.
- **Fix:** Standardize on `name: react-build` and ensure `path` points to the zip. Also, deploy the **zip file** (not a folder) with `package: frontend.zip`.

9.9 `cp: cannot stat 'public/web.config': No such file or directory`

- **Cause:** We later removed reliance on `web.config` for Linux.
- **Fix:** Generate `staticwebapp.config.json` instead.

---

10) Operational Runbook

10.1 Verify service health

- **Frontend:** open `/` and check console network tabs.
- **Backend:** open `/` (Hello World) or `/health` if added.

10.2 Restart

- Portal → App Service → **Restart** (use for both apps).
- Kudu WebSSH is not used to restart; use portal or the SCM API if needed.

10.3 Inspect deployment content

- **Kudu → SSH** → `cd /home/site/wwwroot && ls -la`.
- Files expected (frontend): `index.html`, `assets/`, `staticwebapp.config.json`, `favicon.ico`.

10.4 Logs

- Portal → App Service → **Log stream**.
- For frontend (pm2), startup messages confirm serving `/home/site/wwwroot`.

---

11) Change Management (SMP)

- **Branching:** Work on feature branches → PR → **main**.
- **Triggers:** Both workflows trigger on `push` to `main` or manual dispatch.

- **Approvals:** Use PR reviewers as gatekeepers.
- **Rollback:** Re-run a previous GitHub Actions deployment using the earlier successful artifact (or redeploy a known-good commit).

---

12) Security / Compliance Notes

- Frontend holds only the **publishable** Stripe key (OK).
- If secret Stripe server keys are ever introduced, keep them in **App Service → Configuration** (or Key Vault) and **never** in the repo.
- Enable **HTTPS Only** on both apps (on by default).
- Consider adding **Application Insights** for backend.

---

13) Future Enhancements

- Add `/health` and `/ready` endpoints to backend for monitoring.
- Move artifacts to Azure Storage/static hosting or Azure Static Web Apps if you prefer a fully managed static SPA pipeline.
- Add load testing and autoscaling for backend.

---

14) Final Checklist

- [x] Backend reachable at `https://my-backend-app123.azurewebsites.net` (Hello World / API OK)
- [x] Frontend renders at `https://my-frontend-app123.azurewebsites.net`
- [x] CORS configured for frontend origin
- [x] CI/CD for both apps green
- [x] `staticwebapp.config.json` present in `/home/site/wwwroot`
- [x] Startup command configured: `pm2 serve /home/site/wwwroot --no-daemon --spa`
- [x] Stripe publishable key present in `frontend/src/config/index.ts`

Status: ✅Operational