Debug Wix Frontend and EC2 Backend Test Summary: Passed, Failed, and Next Steps

Status Report `Jun 19, 2025`

Here's a **clear summary** of all tests run — grouped by **passed**, **failed**, and related to **critical integrations between Wix Frontend and EC2 Backends (Registry + Backend nodes)**.

✅ Tests That Passed

✅ Test	Result	Notes
PM2 process boots for registry-backend	✔️ `online`	PM2 confirms the registry node is running without crashes.
`server.js` **listens on** `0.0.0.0:3000`	✔️	Allows external EC2 traffic to reach it — confirmed in code.
Health check endpoint defined (`/health` **)**	✔️	Present in server.js and correctly placed before middleware.
Federation sync job (`runFederationSync()` **) is called**	✔️	Automatically starts on boot.
Correct CORS settings for Wix (`https://www.ntari.org`)	✔️	Production-safe CORS with credentials enabled.
All core routes present	✔️	Including `/api/auth`, `/users`, `/products`, `/marketplace`, etc.
No syntax or duplicate declaration errors	✔️	`userRoutes.js` was successfully debugged and restructured.

❌ Tests That Failed

❌ Test	Result	Root Cause
`curl -i http://localhost:3000/health` **(from inside EC2)**	❌ Failed to connect	This *initially* failed due to broken server.js or port not binding — later fixed.
`curl -i http://18.234.49.114:3000/health` **(from backend EC2)**	❌ Still failing	Implies that even though PM2 is running, the port is still not exposed or reachable.

`lsof -iTCP:3000 -sTCP:LISTEN` **shows no output**	❌ Port not bound	Server is running, but not actively listening — possibly a logic or port conflict.
Federation test: `POST /federation/users`	❌ 502/connection refused	Either nginx is misconfigured or registry is not responding behind it.
Frontend ↔ Registry Sync (via curl from Wix or localhost)	❌ Fails to connect	Confirmed failure to connect to port 3000 — either due to EC2 firewall, missing `server.listen()`, or pm2 crashing before binding.

⚠️ Mixed or Incomplete Tests

🟡 Test	Status	Notes
MongoDB external connection	🟡 Confirmed running	But not all backend services confirmed as connected.
Socket.IO **bindings**	🟡 No crash, but no events tested	Function present, but socket endpoints haven't been verified yet.
Wix API automation & webhook integration	🟡 Unknown	Not tested yet — needs to be verified via Wix Dev Mode.

🔄 What Must Be Verified Now (To Reach Stable Integration)

🔧 Task	Command / Location
Confirm `server.listen(PORT, '0.0.0.0')` **works**	Try `curl -i http://localhost:3000/health` again inside registry node
Verify binding on port 3000	`sudo lsof -iTCP:3000 -sTCP:LISTEN -n -P`
Restart PM2 fresh and re-check	`pm2 delete all && pm2 start server.js --name registry-backend`
Open EC2 security group for port 3000 (ingress)	Confirm in AWS Console: port 3000 is open from 0.0.0.0/0
Test Federation again	`curl -X POST http://18.234.49.114:3000/federation/users` with API key

🌐 Wix ↔ Backend/Registry Communication Critical Paths

Component	Endpoint	Status
Wix Frontend ↔ Backend EC2	`/api/auth`, `/users`, `/products`	✔️ (assumed)
Wix Frontend ↔ Registry EC2	`/federation/users`, `/trends`, `/deposit`	❌ Currently failing
Backend ↔ Registry	`/federation/users`, `POST/GET`	❌ No external port exposure

🧩 Summary

> Fruitful app is **very close to stable federation launch**.

The **code structure is now correct**, federation sync logic is present, and core routes are back in place. What's failing now is **port binding + network visibility**.

✅ Next Steps for Full Pass

1. **Fix port binding issue** (`lsof -iTCP:3000`) → ensure server.listen is active.
2. **Double-check server crash logs** with `pm2 logs registry-backend`.
3. **Check AWS security group** for inbound TCP/3000.
4. **Temporarily disable nginx reverse proxy**, or check nginx config if it's intercepting and failing.
5. **Retry federation test** after server is verified listening.

✅ LIVE DEBUG CHECKLIST – REGISTRY NODE

🧪 **Live Debug Checklist** for getting the **Registry Node** back to a stable state and ensuring **Wix ↔ EC2 Federation Sync** works.

🔍 Step 1: SSH into Registry EC2

```
ssh ec2-user@ec2-18-234-49-114.compute-1.amazonaws.com
```

📦 Step 2: Check that the App is Running via PM2

```
pm2 list pm2 logs registry-backend --lines 50
```

✅ **Pass:** Status is `online`
❌ **Fail:** Status is `errored` → fix `server.js` or dependency issues

🔎 Step 3: Check Port Binding

```
sudo lsof -iTCP:3000 -sTCP:LISTEN -n -P
```

✅ **Pass:** Shows `node` process listening
❌ **Fail:** No output → PM2 didn't start properly or server not listening

---

🛠️ Step 4: Restart PM2 with Updated Environment

```
pm2 delete all pm2 start server.js --name registry-backend --update-env
```

Then **verify logs again:**

```
pm2 logs registry-backend --lines 50
```

🧪 Step 5: Test Server from Inside EC2

```
curl -i http://localhost:3000/health
```

✅ **Pass:** Returns `200 OK` with `OK`
❌ **Fail:** Logs will show why app isn't binding or routing

🌐 Step 6: Test Server from Backend Node

From **backend EC2**:

```
curl -i http://18.234.49.114:3000/health
```

✅ **Pass:** Registry node is publicly reachable
❌ **Fail:** Likely an AWS Security Group or NGINX proxy config issue

---

🔐 Step 7: Verify EC2 Inbound Rules

1. Go to AWS EC2 console.
2. Open **Security Group** for registry node.
3. Confirm **Inbound Rule** for:
   - **Type:** Custom TCP
   - **Port:** `3000`
   - **Source:** `0.0.0.0/0` (for testing only)

📡 Step 8: Federation Test from Backend Node

```
curl -X POST http://18.234.49.114:3000/federation/users \   -H "x-api-key: secret-key-1234" \   -H "Content-Type: application/json" \   -d '{"name":"Calvin","email":"test@agrinet.org","location":"KY","role":"farmer"}'
```

✅ **Pass:** Returns `User federated` or `already federated`
❌ **Fail:** Look for:

- CORS misconfig
- authMiddleware blocking
- crash in `/federation/users` route

🧰 Step 9: Fix and Validate server.js

Ensure this is in your `server.js`:

```
const PORT = process.env.PORT || 3000; server.listen(PORT, '0.0.0.0', () => {   console.log(`Registry Server running on port ${PORT}`); });
```

Don't use: `app.listen(...)` when `http.createServer(app)` is defined.

🧯 Step 10: If Still Broken, Run in Foreground

Use Node directly instead of PM2:

```
node server.js
```

This helps you **see the crash** in real-time for fast fixes.

📌 Optional: NGINX Conflict Check

```
sudo systemctl stop nginx curl -i http://localhost:3000/health
```

If this works now, your NGINX reverse proxy config is interfering.