Railway Reconfiguration Guide - FastAPI + React Architecture

Date: February 11, 2026
Current App: https://jcn-dashboard-production.up.railway.app/
GitHub Repo: https://github.com/alexbernal0/JCN-dashboard
Branch: master (now updated with FastAPI + React)

🎯 Goal

Reconfigure your existing Railway deployment to run the new **FastAPI + React** architecture, which requires **TWO separate services** instead of one.

📋 Prerequisites

- ✅ Code merged to master branch (DONE)
- ✅ Railway account access
- ✅ Existing Railway project: `jcn-dashboard-production`

🚀 Step-by-Step Instructions

Step 1: Access Railway Dashboard

1. **Open Railway Dashboard**
    - Go to: https://railway.app/dashboard
    - Log in with your account
2. **Find Your Project**

- Look for: `jcn-dashboard-production` (or similar name)
- Click on the project to open it

Screenshot Reference: You should see your current Streamlit deployment

---

Step 2: Understand Current Setup

Current Configuration (Old - Streamlit):

```
┌─────────────────────────────────┐
│   Single Service (Streamlit)    │
│   - Runs app.py                 │
│   - Port: $PORT                 │
└─────────────────────────────────┘
```

New Configuration (FastAPI + React):

```
┌──────────────────┐     ┌──────────────────┐
│  Backend Service │     │ Frontend Service │
│  (FastAPI)       │◄────│  (React/Vite)    │
│  Port: 8000      │     │  Port: 3000      │
│  /backend dir    │     │  /frontend dir   │
└──────────────────┘     └──────────────────┘
```

Step 3: Create Backend Service

3.1 Add New Service

1. **Click "New Service"** button (or "+ New" button)
   - Location: Top right or center of project dashboard
2. **Select "GitHub Repo"**
   - Choose: `alexbernal0/JCN-dashboard`
   - Branch: `master`

- Click “Add Service”

3.2 Configure Backend Service

1. **Name the Service**
   - Click on the service name (top left)
   - Rename to: `jcn-backend`
   - Press Enter to save
2. **Set Root Directory**
   - Click on **Settings** tab (or gear icon)
   - Scroll to **“Root Directory”** section
   - Enter: `/backend`
   - Click “Update” or it saves automatically
3. **Verify Build Settings**
   - Still in Settings tab
   - Look for **“Build”** section
   - Should show:
     - Builder: `NIXPACKS` (auto-detected)
     - Build Command: (leave empty, auto-detected)
     - Install Command: (leave empty, auto-detected)
4. **Set Start Command**
   - In Settings tab, find **“Deploy”** section
   - Set **Start Command** to:

     ```
     uvicorn app.main:app --host 0.0.0.0 --port $PORT
     ```

   - Click “Update”
5. **Set Environment Variables**

- Still in Settings tab
- Find **“Variables”** section
- Click **”+ New Variable”**
- Add these variables one by one:

| Variable Name | Value | Notes | |———–——|——-|——-| | `PORT` | `8000` | Backend port | | `DEBUG` | `false` | Production mode | | `MOTHERDUCK_TOKEN` | `[your-token]` | Optional - only if using real data |

To add each variable:

- Click “+ New Variable”
- Enter Variable Name
- Enter Value
- Click “Add”
- Repeat for each variable

1. **Deploy Backend**
   - Click **“Deploy”** button (top right)
   - Or wait for auto-deploy to trigger
   - Watch the **“Deployments”** tab for progress
2. **Get Backend URL**
   - Once deployed, go to **“Settings”** tab
   - Find **“Domains”** section
   - You’ll see a URL like: `https://jcn-backend-production-xxxx.up.railway.app`
   - **COPY THIS URL** - you’ll need it for frontend setup

Step 4: Create Frontend Service

4.1 Add Another Service

1. **Click “New Service”** button again

- In the same project

2. **Select "GitHub Repo"** again
   - Choose: `alexbernal0/JCN-dashboard`
   - Branch: `master`
   - Click "Add Service"

4.2 Configure Frontend Service

1. **Name the Service**
   - Click on the service name
   - Rename to: `jcn-frontend`
   - Press Enter to save
2. **Set Root Directory**
   - Click on **Settings** tab
   - Scroll to **"Root Directory"** section
   - Enter: `/frontend`
   - Click "Update"
3. **Set Build Command**
   - In Settings tab, find **"Build"** section
   - Set **Build Command** to:

     ```
     npm install && npm run build
     ```

   - Click "Update"
4. **Set Start Command**
   - In Settings tab, find **"Deploy"** section
   - Set **Start Command** to:

```
npx serve -s dist -l $PORT
```

- Click “Update”

5. **Set Environment Variables**
   - In Settings tab, find **“Variables”** section
   - Add these variables:

Variable Name	Value	Notes
`PORT`	`3000`	Frontend port
`VITE_API_URL`	`[backend-url-from-step-3.7]`	Backend URL you copied
`VITE_USE_MOCK`	`true`	Use mock data initially

Important: For `VITE_API_URL`, use the backend URL from Step 3.7

- Example: `https://jcn-backend-production-xxxx.up.railway.app`
- Do NOT include trailing slash

1. **Deploy Frontend**
   - Click **“Deploy”** button
   - Watch the **“Deployments”** tab for progress
2. **Get Frontend URL**
   - Once deployed, go to **“Settings”** tab
   - Find **“Domains”** section
   - You’ll see a URL like: `https://jcn-frontend-production-yyyy.up.railway.app`
   - **This is your new dashboard URL!**

Step 5: Update Backend CORS Settings

The backend needs to allow requests from the frontend domain.

1. **Go to Backend Service**

- Click on `jcn-backend` service

2. **Add Frontend URL to Environment Variables**
   - Go to **Settings** → **Variables**
   - Click **"+ New Variable"**
   - Add:
     - Variable Name: `FRONTEND_URL`
     - Value: `[your-frontend-url-from-step-4.7]`
   - Click "Add"
3. **Redeploy Backend** (if needed)
   - Backend should auto-redeploy with new variable
   - Or click "Deploy" button

Note: The backend code already includes `https://*.railway.app` in CORS origins, so this should work automatically.

Step 6: Handle Old Streamlit Service (Optional)

You now have 3 services in your project:

1. Old Streamlit service (original)
2. New Backend service (FastAPI)
3. New Frontend service (React)

Option A: Keep Old Service (Recommended initially)

- Leave it running as backup
- You can access it at the old URL
- Delete later once new app is confirmed working

Option B: Delete Old Service

1. Click on the old Streamlit service
2. Go to **Settings** tab

3. Scroll to bottom
4. Click **“Delete Service”**
5. Confirm deletion

Step 7: Test the New Deployment

7.1 Test Backend

1. **Open Backend URL**
   - Go to: `https://[your-backend-url]/health`
   - Should see: `{"status": "healthy"}`
2. **Test API Docs**
   - Go to: `https://[your-backend-url]/api/docs`
   - Should see Swagger UI with API documentation
3. **Test Mock Data Endpoint**
   - Go to: `https://[your-backend-url]/api/v1/mock/portfolios/`
   - Should see JSON with portfolio list

7.2 Test Frontend

1. **Open Frontend URL**
   - Go to: `https://[your-frontend-url]`
   - Should see the new React dashboard with:
     - Modern header
     - Sidebar navigation
     - Portfolio cards on home page
2. **Test Navigation**
   - Click “Persistent Value” portfolio card
   - Should see:

        - Portfolio performance chart
        - Sector allocation chart
        - Stock table
        - Metrics

3. **Test Stock Search**
    - Click “Stock Analysis” in sidebar
    - Enter a stock symbol (e.g., “AAPL”)
    - Click “Search”
    - Should see stock details and charts

Step 8: Update Custom Domain (If You Have One)

If you have a custom domain pointing to the old Streamlit app:

1. **Go to Frontend Service**
    - Click on `jcn-frontend` service
2. **Add Custom Domain**
    - Go to **Settings** → **Domains**
    - Click **”+ Custom Domain”**
    - Enter your domain (e.g., `dashboard.jcn.com`)
    - Railway will provide DNS instructions
3. **Update DNS Records**
    - Go to your domain registrar (GoDaddy, Namecheap, etc.)
    - Update CNAME record to point to Railway URL
    - Wait for DNS propagation (5-30 minutes)

🔍 Troubleshooting

Backend Issues

Problem: Backend deployment fails

Solutions:

1. Check **Deployments** tab for error logs
2. Verify Root Directory is set to `/backend`
3. Verify Start Command is correct
4. Check that `requirements.txt` exists in backend folder

Problem: Backend shows 500 errors

Solutions:

1. Check environment variables are set correctly
2. Look at logs in **Deployments** tab
3. Try setting `DEBUG=true` temporarily to see detailed errors
4. Check if Python dependencies installed correctly

Problem: CORS errors when frontend calls backend

Solutions:

1. Verify frontend URL is in backend CORS settings
2. Check that `VITE_API_URL` in frontend has correct backend URL
3. Make sure backend URL doesn't have trailing slash

Frontend Issues

Problem: Frontend deployment fails

Solutions:

1. Check **Deployments** tab for error logs
2. Verify Root Directory is set to `/frontend`
3. Verify Build Command and Start Command are correct
4. Check that `package.json` exists in frontend folder

Problem: Frontend shows blank page

Solutions:

1. Open browser console (F12) and check for errors
2. Verify `VITE_API_URL` is set correctly
3. Try setting `VITE_USE_MOCK=true` to use mock data
4. Check that frontend service is running (not crashed)

Problem: API calls fail from frontend

Solutions:

1. Check Network tab in browser DevTools
2. Verify backend URL is correct in `VITE_API_URL`
3. Test backend health endpoint directly
4. Check CORS configuration in backend

Problem: Charts not showing

Solutions:

1. Check browser console for errors
2. Verify mock data is being returned from backend
3. Try refreshing the page
4. Check that ECharts library loaded correctly

📊 Expected Results

Backend Service

URL: `https://jcn-backend-production-xxxx.up.railway.app`

Endpoints:

- `/health` → `{"status": "healthy"}`
- `/api/docs` → Swagger UI
- `/api/v1/mock/portfolios/` → Portfolio list
- `/api/v1/mock/portfolios/persistent_value` → Portfolio details
- `/api/v1/mock/stocks/AAPL` → Stock details

Frontend Service

URL: `https://jcn-frontend-production-yyyy.up.railway.app`

Features:

- ✅ Modern React UI
- ✅ Responsive design
- ✅ Interactive charts (ECharts)
- ✅ Sortable/filterable tables
- ✅ Stock search
- ✅ Portfolio details
- ✅ Fast page navigation

🎯 Success Checklist

Before considering deployment complete, verify:

- [ ] Backend service deployed successfully

- [ ] Backend health endpoint returns 200 OK
- [ ] Backend API docs accessible
- [ ] Frontend service deployed successfully
- [ ] Frontend loads without errors
- [ ] Can navigate to portfolio pages
- [ ] Charts display correctly
- [ ] Tables are interactive (sorting, filtering)
- [ ] Stock search works
- [ ] No CORS errors in browser console
- [ ] Mobile responsive (test on phone or resize browser)

📈 Performance Expectations

Load Times

- **Backend API**: < 500ms (mock data)
- **Frontend Initial Load**: 1-3 seconds
- **Page Navigation**: < 100ms (instant)
- **Chart Rendering**: < 500ms

Comparison to Old Streamlit App

- **90-95% faster** page loads
- **Instant navigation** (no page reloads)
- **Better mobile experience**
- **More interactive** charts and tables

🔄 Switching from Mock to Real Data

Once everything is working with mock data:

1. **Update Frontend Environment Variable**
    - Go to `jcn-frontend` service
    - Settings → Variables
    - Change `VITE_USE_MOCK` from `true` to `false`
    - Redeploy
2. **Add MotherDuck Token to Backend** (if not already added)
    - Go to `jcn-backend` service
    - Settings → Variables
    - Add `MOTHERDUCK_TOKEN` with your token
    - Redeploy
3. **Test Real Data**
    - Open frontend URL
    - Navigate to portfolio
    - Wait 2-5 seconds for real data to load
    - Verify data looks correct

💰 Railway Costs

Free Tier

- **Execution Time**: 500 hours/month
- **Services**: Unlimited
- **Cost**: $0

With 2 Services

- **Estimated Usage**: ~720 hours/month (if running $^{24}/_{7}$)
- **Recommendation**: Upgrade to Pro ($20/month)

Pro Tier

- **Execution Time**: Unlimited
- **Cost**: $20/month + usage
- **Includes**: All features, better performance

📞 Need Help?

If you encounter issues:

1. **Check Deployment Logs**
    - Go to service → Deployments tab
    - Click on latest deployment
    - Read error messages
2. **Check Browser Console**
    - Press F12 in browser
    - Look for errors in Console tab
    - Check Network tab for failed requests
3. **Test Endpoints Directly**
    - Use curl or browser to test backend URLs
    - Verify responses are correct
4. **Share Error Messages**
    - Copy exact error messages
    - Share deployment logs
    - Describe what you see vs. what you expect

🎉 Next Steps After Successful Deployment

1. **Test All Features**
    - Browse all portfolio pages
    - Test stock search
    - Try on mobile device
    - Check all charts and tables
2. **Update Bookmarks**
    - Update any bookmarks to new URL
    - Share new URL with team/clients
3. **Monitor Performance**
    - Check Railway dashboard for usage
    - Monitor for any errors in logs
    - Watch for slow endpoints
4. **Plan Enhancements**
    - Review UI/UX Quick Reference Guide
    - Request design changes
    - Add new features

📝 Summary

What You're Doing:

- Creating 2 new Railway services (backend + frontend)
- Configuring each with correct settings
- Testing the new architecture
- Optionally removing old Streamlit service

Time Required: 15-20 minutes

Difficulty: Medium (just follow steps carefully)

Result: Modern, fast, scalable dashboard

Ready to start? Follow the steps above and let me know if you need help at any point!

Last Updated: February 11, 2026