Vercel Deployment Fix - VITE_BACKEND_URL Issue

Problem

The frontend deployed on Vercel was trying to connect to `localhost:3001` instead of the production backend URL (`https://workigom.onrender.com`), even though the `VITE_BACKEND_URL` environment variable was set in Vercel's dashboard.

Root Cause

Vite environment variables must be available at **build time**, not runtime. The variable was set in Vercel but wasn't being properly injected during the build process.

Solution Applied

1. Created `vercel.json` Configuration

- Added explicit environment variable handling
- Configured rewrites for SPA routing
- Added security headers

2. Created `.env.production` File

- Set default production backend URL
- Provides fallback if Vercel env var fails

3. Updated `.gitignore`

- Ensured `.env.local` files are not committed
- Kept `.env.example` and `.env.production` in repo

4. Verified `lib/api.ts` Configuration

The existing API configuration already properly handles the environment variable:

```
const backendUrl = import.meta.env.VITE_BACKEND_URL || 'http://localhost:3001';
```

Files Modified/Created

1. `src-frontend/vercel.json` - New file
2. `src-frontend/.env.production` - New file
3. `src-frontend/.gitignore` - New file
4. `VERCEL_DEPLOYMENT_FIX.md` - This documentation

Next Steps for Vercel Deployment

Step 1: Push Changes to GitHub

```
git add src-frontend/vercel.json src-frontend/.env.production src-frontend/.gitignore
VERCEL_DEPLOYMENT_FIX.md
git commit -m "fix: Configure Vercel environment variables for production build"
git push origin master
```

Step 2: Verify Vercel Environment Variable

1. Go to your Vercel project dashboard
2. Navigate to **Settings** > **Environment Variables**
3. Ensure `VITE_BACKEND_URL` is set to `https://workigom.onrender.com`
4. Make sure it's enabled for **Production** environment
5. Variable should be set as **Plain Text** (not Secret)

Step 3: Trigger New Deployment

Option A - Automatic (Recommended):
- Vercel will auto-deploy when you push to master

Option B - Manual:
1. Go to Vercel dashboard
2. Click on **Deployments** tab
3. Click **...** menu on the latest deployment
4. Select **Redeploy**
5. ✅ Check **"Use existing Build Cache"** should be **UNCHECKED**

Step 4: Verify the Fix

After deployment completes:
1. Visit https://workigom.vercel.app
2. Open browser DevTools (F12)
3. Go to **Console** tab
4. Look for: `🔧 API Configuration:` log
5. Verify it shows:

`VITE_BACKEND_URL: "https://workigom.onrender.com"`
`finalApiUrl: "https://workigom.onrender.com/api"`

Step 5: Test API Calls

1. Try to view jobs list
2. Check Network tab in DevTools
3. API calls should go to: `https://workigom.onrender.com/api/jobs`
4. NOT to: `localhost:3001/api/jobs`

Troubleshooting

If Still Seeing localhost:3001

Problem: Old build is cached

Solution:
1. Go to Vercel project settings
2. **Settings** > **General**
3. Scroll to **Build & Development Settings**
4. Click **Edit** next to Output Directory
5. Change from `dist` to `dist-new` and save
6. Trigger new deployment
7. Change back to `dist`
8. This forces a complete rebuild

If API Calls Still Fail

Check Backend Status:

```
curl https://workigom.onrender.com/api/health
```

Should return:

```
{
  "success": true,
  "message": "API is running"
}
```

Check CORS Settings on backend:
- Ensure `https://workigom.vercel.app` is in allowed origins
- Check backend `.env` has: `CORS_ORIGIN=https://workigom.vercel.app`

Technical Details

Why This Works

1. **vercel.json**: Tells Vercel to explicitly inject env vars during build
2. **.env.production**: Provides fallback for production builds
3. **Vite**: Reads `VITE_*` variables at build time and replaces them in the bundle
4. **import.meta.env**: Vite's way of accessing env variables (not process.env)

Environment Variable Precedence

1. Vercel Dashboard Environment Variables (highest)
2. `.env.production` file
3. `.env` file
4. Hardcoded fallback in code (lowest)

Expected Outcome

✅ Frontend calls backend at `https://workigom.onrender.com/api/*`
✅ No more `ERR_CONNECTION_REFUSED` errors
✅ Jobs list loads successfully
✅ All API features work in production

Verification Complete

Once deployed, you should see in the browser console:
- ✅ No connection errors
- ✅ Correct backend URL in API config log
- ✅ Successful API responses (status 200)