Vercel Build Fix Documentation

Issue Summary

The Vercel deployment for QuantPay Stack was failing due to TypeScript compilation errors in the `qpc-v2-core` package. The build command being executed was:

```
cd ../../../qpc-v2-core && npm install && npm run build && cd - && npm run build
```

TypeScript Compilation Errors

The following TypeScript compilation errors were occurring:

1. **Cannot find name 'console'** - Missing DOM lib in tsconfig
2. **Cannot find name 'setTimeout'** - Missing DOM lib in tsconfig
3. **Missing type declarations for 'uuid' module** - Requires @types/uuid
4. **Missing type declarations for 'libsodium-wrappers' module** - Requires @types/libsodium-wrappers

Root Cause

The `tsconfig.json` in the `qpc-v2-core` package had an incomplete `lib` configuration:

```
"lib": ["ES2020"]
```

This configuration only included ES2020 features but excluded DOM APIs like `console`, `setTimeout`, `setInterval`, etc., which are commonly used in Node.js and browser environments.

Solution Applied

1. Fixed tsconfig.json

Updated the `lib` array in `qpc-v2-core/tsconfig.json` to include DOM support:

```
"lib": ["ES2020", "DOM"]
```

File Changed: `qpc-v2-core/tsconfig.json`

Change Details:
- **Before:** `"lib": ["ES2020"]`
- **After:** `"lib": ["ES2020", "DOM"]`

2. Type Declarations

The `package.json` already contained all necessary `@types` packages in `devDependencies`:
- ✅ `@types/uuid: ^9.0.7`

- ✅ `@types/libsodium-wrappers: ^0.7.14`
- ✅ `@types/node: ^20.9.0`

No additional packages needed to be installed.

Verification

The fix was verified locally by running:

```
cd qpc-v2-core
npm install
npm run build
```

Result: ✅ Build completed successfully with no TypeScript errors.

The compiled output was generated in the `dist/` directory as expected.

Commit Details

Commit Message:

```
fix: resolve TypeScript compilation errors in qpc-v2-core

- Add DOM library to tsconfig.json lib array to enable console, setTimeout, and other
DOM APIs
- This fixes TypeScript compilation errors preventing Vercel deployment
- All @types packages were already present in package.json
- Build verified locally with npm run build
```

Commit Hash: `8f0a58e`

Expected Outcome

With this fix applied and pushed to the main branch, the Vercel deployment should now:

1. ✅ Successfully compile the `qpc-v2-core` package
2. ✅ Recognize DOM APIs (console, setTimeout, etc.)
3. ✅ Find all required type declarations
4. ✅ Complete the full build pipeline without TypeScript errors

Next Steps

1. **Trigger a new Vercel deployment** by pushing to the main branch (already done)
2. **Monitor the Vercel build logs** to confirm the build succeeds
3. **Verify the deployed application** functions as expected

Technical Notes

Why Add DOM to TypeScript lib?

Even in Node.js environments, many APIs like `console`, `setTimeout`, and `setInterval` are defined in TypeScript's DOM library. Including DOM in the `lib` array:
- Provides type definitions for these global APIs
- Does not affect the runtime environment
- Is a common practice for Node.js TypeScript projects
- Ensures compatibility with both Node.js and browser environments

TypeScript Library Configuration

The `lib` option in `tsconfig.json` tells TypeScript which built-in type definitions to include:
- **ES2020**: ECMAScript 2020 features (Promise, async/await, etc.)
- **DOM**: Browser and global APIs (console, setTimeout, Document, etc.)

Troubleshooting

If the Vercel build still fails after this fix:

1. **Clear Vercel Cache:** Go to Vercel deployment settings and clear the build cache
2. **Check Node Version:** Ensure Vercel is using Node.js 18.0.0 or higher (as specified in package.json engines)
3. **Verify File Changes:** Confirm the tsconfig.json change is present in the deployed branch
4. **Check Build Command:** Verify the Vercel build command matches the expected command

Additional Fix: Moving @types to Production Dependencies

Issue Identified

After the initial tsconfig.json fix, the Vercel build was still failing with the following errors:

```
Could not find a declaration file for module 'uuid'
Could not find a declaration file for module 'libsodium-wrappers'
```

Root Cause: Vercel runs `npm install` in production mode (without devDependencies), which was indicated by only 39 packages being installed instead of the full 435 packages. The `@types` packages were in `devDependencies`, making them unavailable during the TypeScript compilation phase.

Local Build Success: The build worked locally because devDependencies are installed by default in local development environments.

Solution Applied

Moved TypeScript type declaration packages from `devDependencies` to `dependencies` in `qpc-v2-core/package.json`:

Packages Moved:
- `@types/uuid: ^9.0.7`
- `@types/libsodium-wrappers: ^0.7.14`
- `@types/node: ^20.9.0`

Kept in devDependencies:
- `typescript: ^5.2.2` (build tool)
- `@types/jest: ^29.5.8` (only needed for tests)
- Other development and testing tools

Why This Fix is Necessary

In Vercel's deployment environment:

1. **Production Mode Install:** Vercel runs `npm install --production` or sets `NODE_ENV=production`, which skips devDependencies
2. **Build Phase Requirements:** TypeScript compilation happens during the build phase and requires type declarations
3. **Type Packages Must Be Available:** For production builds, `@types` packages must be in `dependencies` to be available during compilation

This is a common pattern for monorepos and packages that need to be built in CI/CD environments.

Verification Steps

```
cd qpc-v2-core
npm install       # Updated package-lock.json
npm run build     # ✅ Build successful
```

Result: Build completed successfully with all type declarations available.

Commit Details

Commit Message:

```
fix: move @types packages to dependencies for Vercel build
```

Commit Hash: `51fc894`

Files Changed:
- `qpc-v2-core/package.json`
- `qpc-v2-core/package-lock.json`

Impact on Package Size

This change adds approximately 3 small type declaration packages to the production dependencies:
- Type packages are TypeScript definition files only (.d.ts)
- They have minimal impact on bundle size
- They are not included in the final JavaScript runtime

Best Practice Note

For packages that need to be compiled (especially in monorepos or CI/CD environments), it's a best practice to include `@types` packages in `dependencies` rather than `devDependencies` to ensure build compatibility across different environments.

Contact

For questions or issues related to this fix, please contact:
- **Author:** Franco Mengarelli
- **Email:** fmengarelli@gmail.com

Last Updated: November 4, 2025
Status: ✅ Both Fixes Applied and Pushed to Main Branch
Commits:
- Initial fix: `8f0a58e` (tsconfig.json)
- Additional fix: `51fc894` (@types packages)