Boeing Printer Parts Lookup - Complete Setup Guide

Overview

This app lets you search printers by Asset Number and instantly see:

- Printer location (floor/room)
- Model information
- Required parts with current stock levels from BarCloud

Prerequisites

- Node.js 16+ installed
- Your two Excel files ready:
 - (AssetGroup_AllAssets4dfed.xls) (48 active printers from XSM)
 - Export 1.xlsx (inventory from BarCloud)

Step-by-Step Setup

1. Create Project Structure

```
mkdir printer-parts-lookup

cd printer-parts-lookup

mkdir public database
```

2. Copy All Code Files

Copy these files I created above into your project:

Root directory:

- (package.json)
- (server.js)
- (discloud.config)
- (create with content below)

public/ folder:

- (index.html)
- (style.css)
- (app.js)

database/ folder:

- (schema.sql)
- (seed.js) (use the final version)
- (convert-excel.js)

3. Create .env File

Create (.env) in the root folder:

```
env

PORT=8080

SESSION_SECRET=change-this-to-random-string-in-production

NODE_ENV=development
```

4. Install Dependencies

bash

npm install

This installs:

- express (web server)
- express-session (login sessions)
- bcrypt (password hashing)
- better-sqlite3 (database)
- doteny (environment variables)
- cors & helmet (security)
- xlsx (Excel reading)

5. Prepare Your Data

bash

```
# Copy your Excel files to the database folder

cp /path/to/AssetGroup_AllAssets4dfed.xls database/

cp /path/to/"Export 1.xlsx" database/

# Convert Excel to JSON

node database/convert-excel.js
```

This creates three JSON files:

- (printers.json) (48 printers)
- (inventory.json) (42 unique SKUs)
- (model-parts.json) (mappings between models and parts)

6. Seed the Database

bash
node database/seed.js

This creates (database/printers.db) with all your data and a default user.

7. Test Locally

bash
npm start

Visit: http://localhost:8080

Default Login:

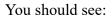
- Username: admin
- Password: (boeing2025)

▲ IMPORTANT: Change this password before deploying!

8. Test the Search

Try searching for an asset number from your list, for example:

- (A4064127)
- Any other asset number from your 48 printers



- Printer details (model, floor, room)
- List of required parts
- Current stock levels for each part

Deploying to Discloud

Prerequisites

- Discloud account with Platinum plan or higher (required for APIs)
- Your project working locally

Steps

1. Update .env for production:

```
PORT=8080
SESSION_SECRET=your-super-secret-random-string-here
NODE_ENV=production
```

2. Create .gitignore:

```
node_modules/
.env
database/*.xls
database/*.xlsx
database/*.json
npm-debug.log
```

3. Package your project:

bash			

```
# Make sure database/printers.db exists and has your data
# Remove Excel files (they're already converted)
rm database/*.xls database/*.xlsx
# Create a zip file
zip -r printer-parts-lookup.zip . -x "node_modules/*" ".git/*"
```

4. Upload to Discloud:

- Log into Discloud dashboard
- Create new application
- Upload (printer-parts-lookup.zip)
- Set environment variables:
 - (SESSION_SECRET): (your random secret)
 - (NODE ENV): (production)

5. Discloud will automatically:

- Run (npm install)
- Start your app with (node server.js)
- Make it available at your Discloud URL

File Structure Reference

```
printer-parts-lookup/
   — package.json
                           # Dependencies and scripts
    - server.js
                        # Express server
    - discloud.config
                            # Discloud deployment config
                       # Environment variables
    - .env
    - .gitignore
                        # Files to ignore in git
    - public/
                        # Frontend files
        - index.html
                          # Main HTML page
       - style.css
                         # Styling
                        # Frontend JavaScript
       – app.js
     database/
                         # Database files
        schema.sql
                          # Database structure
       - seed.js
                       # Data seeding script
```

```
--- convert-excel.js  # Excel to JSON converter
--- printers.db  # SQLite database (generated)
--- printers.json  # Converted printer data
--- inventory.json  # Converted inventory data
--- model-parts.json  # Model-part mappings
```

Common Commands

```
# Install dependencies
npm install

# Convert Excel files to JSON
node database/convert-excel.js

# Set up database
node database/seed.js

# Run locally (development)
npm start

# Run with auto-reload (requires nodemon)
npm run dev
```

Troubleshooting

"File not found" during conversion

- Make sure Excel files are in the database/ folder
- Check file names match exactly

"Login failed"

- Make sure you ran (node database/seed.js)
- Check browser console for errors
- Default credentials: admin / boeing2025

"Printer not found"

- Asset numbers are case-sensitive
- Make sure data was seeded correctly
- Check (database/printers.db) exists

Port already in use

- Change PORT in (.env) to a different number (e.g., 3000)
- Or stop the other process using port 8080

Security Notes

- 1. Change default password before deploying
- 2. **Set strong SESSION_SECRET** (use a password generator)
- 3. **Enable HTTPS** when deploying (Discloud handles this)
- 4. **Don't commit .env** to git
- 5. Don't commit Excel files to git

Next Steps (Future Enhancements)

1. BarCloud API Integration

- Auto-sync inventory without manual exports
- Real-time stock updates

2. Multi-user Support

- Add user registration
- Role-based permissions

3. Advanced Features

- QR code scanning
- Low stock alerts
- Order management
- Physical location tracking (with shelf codes)

4. Mobile App

• React Native version

• Offline mode with sync

Support

If you run into issues:

- 1. Check the console logs (npm start) output)
- 2. Check browser console (F12 \rightarrow Console tab)
- 3. Verify your Excel files have the correct structure
- 4. Make sure all files are copied correctly

You now have everything you need to run the app!

Start with step 1 and work through the setup. Once it's running locally and you've tested it, you can deploy to Discloud.