SunPrints website documentation

This document contains explanations of the SunPrints internal website for managing customer orders and stock purchases.

There are three sections:

* For developers
* For system administrators
* For users

# Developers

The source code is in <https://github.com/wdporter/sunprints.git>

The website stack is built on

[SQLite](https://www.sqlite.org/index.html) database

[Expressjs](https://expressjs.com/) web server (node)

[Datatables](https://datatables.net/) for frontend tables

[Vue 3](https://vuejs.org/) for some editing and displaying tasks

[GDCSS](https://gdcss.netlify.app/) semantic css framework

[Fontawesome](https://fontawesome.com/) icons

## Installing the development environment

The source code is in <https://github.com/wdporter/sunprints.git>

Just clone it and run

>npm install

Get a copy of the database file, sunprints.db and put it in the root folder. It’s too big to put in source control. With the database file, you will need to do an insert into the User table for yourself, more details below.

You also need nodemon in your path

>npm install -g nodemon

With this, you can hit F5 in VS Code and the web server will start up and you are debugging in VS Code. Point your browser to localhost:3000.

## Deployment

There is no build process. Deployment is a straight xcopy to the server (see system administration section).

## SQLite

The database engine is the free and open source SQLite. SQLite consists of a single database file and a library of APIs. To use the APIs with the node-based web server, we use [better sqlite 3](https://github.com/WiseLibs/better-sqlite3) npm package.

There are any number of tools for viewing and interacting with the database. During development we found [DB Browser for SQLite](https://sqlitebrowser.org/) helpful.

Here is a brief list of tables (some are listed more than once):

* Customer 1→\* Orders 1→\* Order Garment 1→\* StockOrderGarment \*←1 Garment
* SalesRep 1→\* Orders
* PrintDesign 1→\* ScreenPrintDesign \*←1 Screen
* EmbroideryDesign 1→\* UsbEmbroideryDesign \*←1 Usb
* TransferDesign 1→\* TransferNameTransferDesign \*←1 TransferName
* PrintDesign 1→\* OrderGarment
* EmbroideryDesign 1→\* OrderGarment
* TransferDesign 1→\* OrderGarment
* Screen 1→\* OrderGarment
* Usb 1→\* OrderGarment
* TransferName 1→\* OrderGarment
* Supplier 1→\* StockOrder 1→\* StockOrderGarment \*←1 Garment
* SalesTotal 1→\* Sales
* AuditLog 1→\* AuditLogEntry
* User

View the database schema in your browser tool to view columns, keys and so forth.

Each table has auditing columns: CreatedBy, CreatedDateTime, LastModifiedBy and LastModifiedDateTime

Every time a change is made to the database, these columns are filled in. On insert — Created and LastModified are set to the same. On update — LastModified is changed.

To manually generate a password hash to use with a manual insert into the users table, use this node command

>require('crypto').createHash('md5').update("MY\_PASSWORD").digest("hex")

## Expressjs

Expressjs is a node based web server framework.

The port number is 3000.

Start the web server on the production server by running

>npm run start

For local development, we can use the nodemon package, already installed in package.json. Nodemon reloads the web server each time it detects a change in file (that is, on saving a file). You can run it with:

>npx nodemon ./bin/www

but it’s better to set up a VS Code task for it.

We use a basic auth package to do user authentication. It’s not very secure but more secure solutions would have been overkill for a system of this size and user base. It’s only value is to restrict access to the “admin”pages, and also to provide a name for auditing columns CreatedBy/LastModifiedBy.

The user of expressjs is rudimentary — the “next” function is never used. Each router method does it’s work in the database and returns when all is complete. There are no asynchronous methods called.

## Datatables

The use of datatables library is reasonably complex. All tables use server side processing. Each route includes a “/dt” path for populating the table. Some of the features used in various places includes

* custom-rendered columns,
* loading extra information on select row,
* buttons add-on,
* fixed headers and
* and so forth.

The tables are really bad at small screen sizes. This will need fixing.

## Vue 3

Vue 3 is used simply with the options API and without components

## GDCSS

This is a semantic classless css framework. Some minor customisations are in theme.css.

## Font awesome

We’re using 4.7 because we couldn’t figure out how to use a later free version from a cdn.

# System Administrators

The website is installed to the share s:\website

The website is running under a node command prompt. To verify it is running, check task manager and see if there is a node process

The website is started with the simple command

>npm run start

There is a simple startup script to restart the site when the server is rebooted. “e:\sunprints\website\start sunprints web app.cmd”. This should run automatically. There is a shortcut to this file in “c:\programdata\microsoft\windows\start menu\programs\startup”

# Users

The website is available at <http://spdc01:3000>

If you need a new user account, ask an admin to create an account via the users page <http://spdc01:3000/users>

For instructions on how to install the web app to a local pc. This is intended for sales reps to take their laptops on the road. There is no provision for synchronising any changes they save.