FSVA Online Development Guide

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1 Introduction

FSVA (Food Security and Vulnerability Atlas) Online is a web application to present information about FSVA program that WFP has.

2 Structure

FSVA Online can be divided into three main parts:

1. Main Web Page
2. Custom Pages
3. Local Database

2.1 Main Web Page

The FSVA itself is a kind of SPA (Single Page Application). The front page consist of three main panes:

1. Menu Pane - *located on left side*
2. Narration Pane - *located on the center, retractable*
3. Content Pane - *located on right side*

2.2 Custom Pages

Custom pages contains pages that will be shown on Content Pane on the Main Web Page. User can click on the menu and Content Pane will load the assigned Custom Pages for that particular menu.

2.3 Local Database

FSVA Online uses file as database for portability. Most of the data that is required to be shown is not that big (less than 1MB). The database file (in csv and json) format are loaded to the browser and queried locally. Some functions like download cvs are also using the local database file loaded on server side to generate file that will be downloaded by user's browser.

3 Installation

3.1 Prerequisite

- Nodejs >= 4 (nodejs.org)
- Nginx for deployment (nginx.org)

Most of Javascript files on FSVA Online uses ES6 format. For browser execution the javascript files will be packed by webpack and transpiled on the fly with Babel JS back to ES5 format. For server side execution, io.js is used. Io.js is a javascript for server side, a fork of the original NodeJs but with more ES6 support.

For Operating System, the three prominent OS (Linux / Windows / Mac OS) will work for hosting FSVA Online app. But Linux would be easier to work with and FSVA Online has been tested under Linux only.

There are other external libraries used in FSVA Online for browser side and server side. Those will be described later.

3.2 Extract FSVA Online Source Code

```
unzip fsva-online.zip /path/to/fsva
```

3.3 Install Server-side dependencies

Server-side dependencies are managed by NPM. The NPM bootstrap file (`package.json`) is located under the root of FSVA Online directory. The shell needs to be in root folder of FSVA Online then call this command:

```
npm install
```

The dependencies that will be installed currently are:

```
  "dependencies": {
    "JSONPath": "^0.10.0",
    "body-parser": "^1.14.2",
    "ckeditor-releases": "^1.0.0",
    "compression": "^1.6.0",
    "datatables": "^1.10.9",
    "express": "^4.13.3",
    "fast-csv": "^1.0.0",
    "fs-promise": "^0.3.1",
    "highcharts-browserify": "^2.0.2",
    "jquery": "^2.2.0",
    "lodash": "^4.0.0",
    "moment": "^2.11.1",
    "normalize.css": "^3.0.3",
    "numeral": "^1.5.3",
    "purecss": "^0.6.0",
    "sprintf-js": "^1.0.3",
    "tiny-modal": "0.0.2",
    "vue": "^1.0.15"
  },
  "devDependencies": {
    "babel-core": "^6.1.21",
    "babel-loader": "^6.1.0",
    "babel-plugin-transform-runtime": "^6.1.18",
    "babel-preset-es2015": "^6.1.18",
    "babel-register": "^6.4.3",
    "babel-runtime": "^5.8.0",
    "bower": "^1.7.2",
    "css-loader": "^0.21.0",
    "file-loader": "^0.8.5",
    "napa": "^2.2.0",
    "node-sass": "^3.4.2",
    "sass-loader": "^3.1.2",
    "style-loader": "^0.13.0",
    "url-loader": "^0.5.7",
    "vue-hot-reload-api": "^1.2.0",
    "vue-html-loader": "^1.0.0",
    "vue-loader": "^7.2.0",
    "webpack": "^1.12.2",
    "webpack-dev-server": "^1.12.0"
  }
```

Then wait until NPM finished fetching remote dependencies. There will be several dependencies outside npm repository managed by https://github.com/shama/napa [napa], to install:

```
npm run napa
```

Then wait until Bower finished fetching remote dependencies.

3.4 Transpile ES6 and CssNext Files

FSVA Online client side source codes are located on `public/` directory. Most of the source codes are using newer format that's not yet supported on standard web browser. So It uses Webpack along with BabelJS and CssNext to combine newer format source codes (ES6 + CSS4) and produce older format (ES5 + CSS3) located on public/dist that would be supported on modern browser today.

To pack the source codes, go to the FSVA root directory and run this command:

```
webpack
```

3.5 Run the FSVA Online Server

To run the FSVA Online, run the server side Javascript app. From root directory, run:

```
node index.js
```

The port that will be used is based on `server/config.js` file

4 Deployment

4.1 About logging

The node process mentioned before will not log the request made to the server extensively. To log the external request, use nginx as the front server, then reverse proxy the request made to nginx to FSVA nodemon process. Nginx will have the log stored on its own.

4.2 Node Server App

Note that if the node server were run using ssh connection is using ssh, the server app will be terminated if the ssh session is closed. It's required for the deployment server to use daemon to run the server app. Several choices available (also depends on OS that is used).

4.3 Language Header

It's required for the node server app to receive *fsva_lang* header, to render the result according to the lang value. Use external webserver such as nginx to listen for several subdomains e.g. `en.fsva-online.com` and `id.fsva-online.com` then pass the language information to node application server.

4.4 PDF Files

FSVA Online has so many PDF attachment(s). As such these files are not included on the original source code. For each deployment, manually copy the files into `public/attachments` folder. Or do a vanilla copy of that folder from old deployment to new deployment.

4.5 Setup Automation

If those setup and deployment notes seem complicated (it actually is), head to `scripts/ansible` directory for automated deployment script. It's using ansible dialect.