Project Name: dbPatcher

Project Type: A Database patcher

Why: This was a temporary in-house project to help data authors in our department to overcome certain limitation existed in the data migration process. I was learning AngularJS at the time and wanted use it for real world application. Ask me in-person if more information if needed. (Sensitive information has been scrubbed from the version found here).

Server side: Spring Boot, Groovy

Client-side (Frameworks/Libraries)

- AngularJS
- Bootstrap
- Yeoman
- Grunt

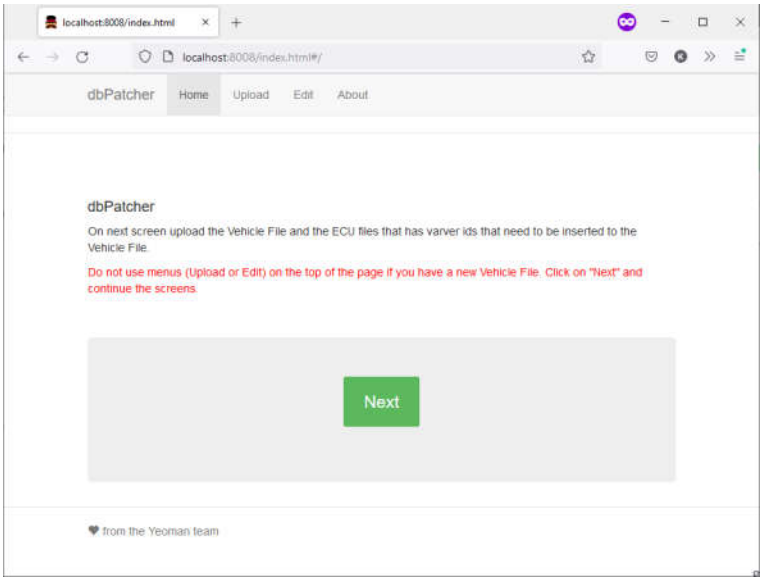
How to start the app

Start the application with this command "**java -jar dbPatcher-0.0.1.jar**". This is packaged as a Spring boot jar so it comes with everything needed to run a website.

Next type "http://localhost:8008/index.html" in the address bar of the browser.

Flow of Actions.

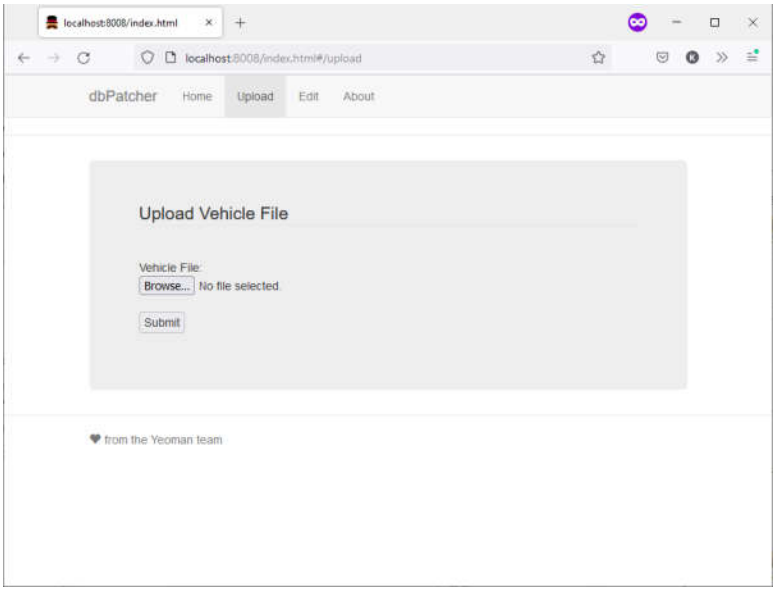
1. Screen 1: Click "Next".
2. Screen 2: Upload the "Vehicle File" on screen 2 (not needed in the demo, can click on edit).
3. Screen 3: Upload one or more ECU File(s) (not needed in the demo, can click on edit).
4. Screen 4: Patch the selected vehicle configuration with ECU information obtained in step 3.
5. Screen 5: Go to final (screen 5) by clicking on "Done" in screen 4 and download the updated "Vehicle File".



Screen 01

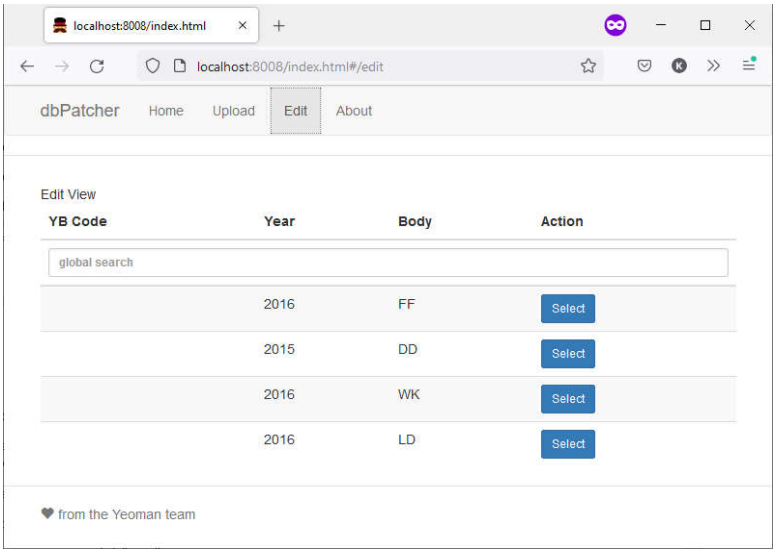
If "Next" is clicked, it will take to Database upload page. But the Vehicle and ECU file information in this demo are hardcoded. Therefore, uploading a file has no effect. But if anyone needs to test the functionality, one can upload any file and they will be uploaded to a folder named "upload-dir" - created at the same location where the .jar file is and to a “vehicles” folder and “ecus” folder respectively.

Therefore, clicking on "Edit" menu item will take the user to next screen.



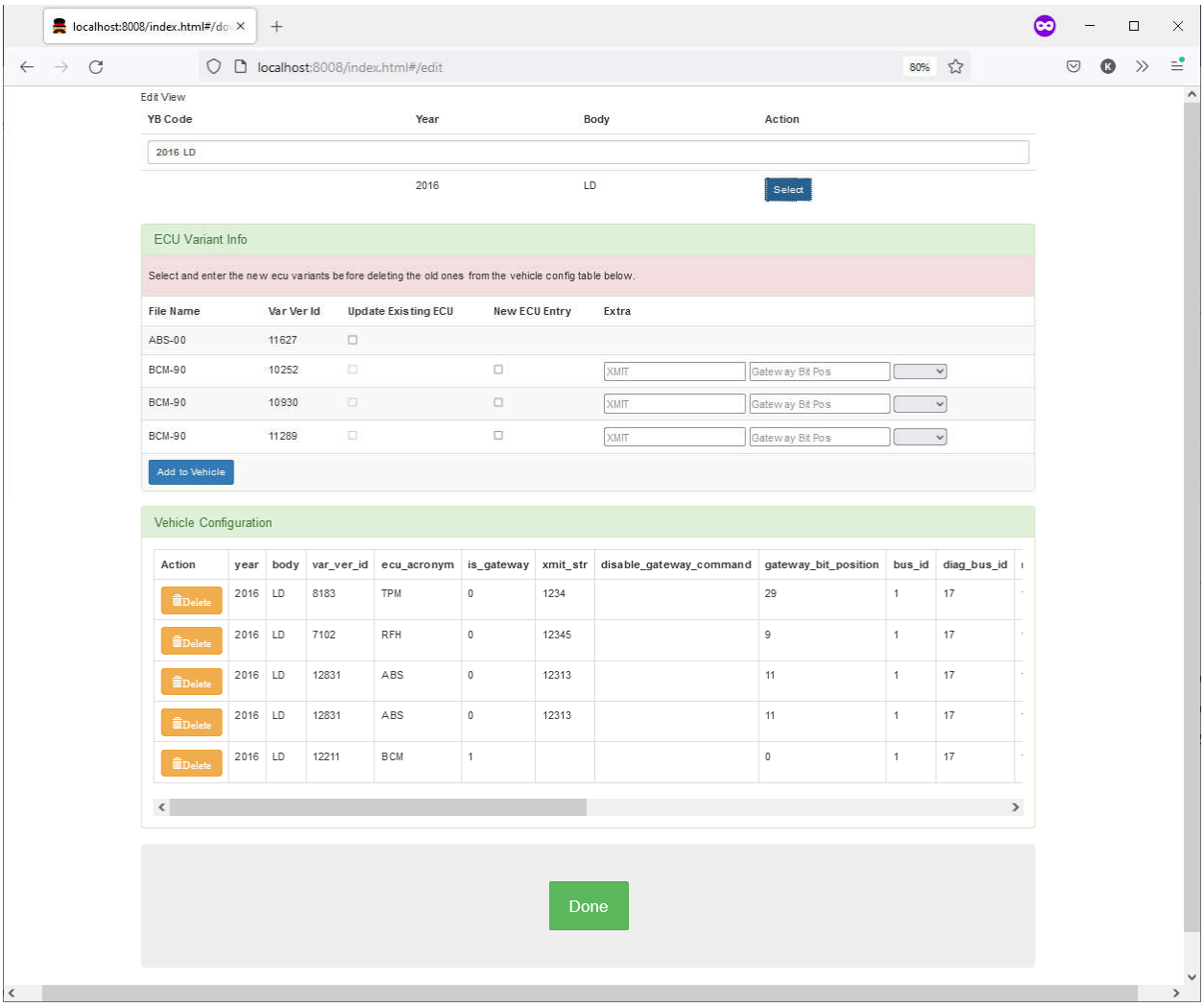
Screen 02

Entering a year or a body-code will filter the year-and model list appropriately.



Screen 03

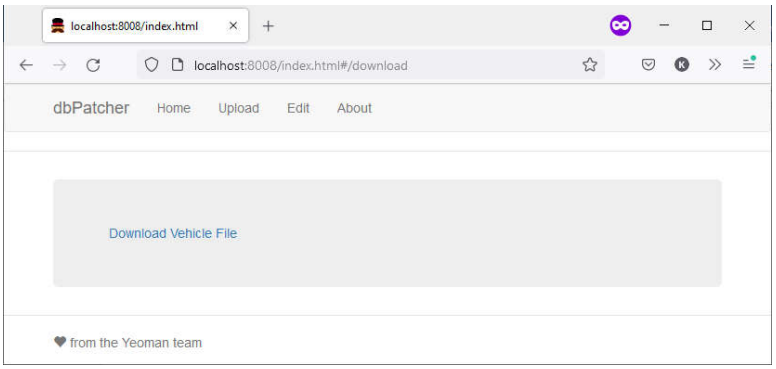
Click "Select" to go to next screen.



Screen 04

This is where the actual “magic” happens. The "ECU Variant Info" area shows what new ecu informatoion needs to be included in a vehicle configuration. They are taken from the ECU Files that were uploaded in the screen 3 in the real application (but that information is hardcoded in this demo). Depending whether the application has already found existing information for each ECU, data it will show a check box under "Update Existing ECU" or "New ECU Entry". "Extra" fields are optional. Once the necessory selections are made and clicked on "Add to Vehicle", the app will update the the Vehicle File and the new data will be populated under the "Vehicle Configuration" (but this demo doesn’t do that).

Once the user is done, clicking on “Done” will take the user to the next screen to download the patched Vehicle File. The real application generates a new Vehicle File based on the uploaded file and the modification, but the demo doesn’t create that patched file.



Screen 05

Clicking on the link in this demo, will generate a 500 error as there is no real “Vehicle File” to be downloaded.D: