BPMN-PYTHON-WEBAPP FIX GUIDE by Wojtek Lupin

How to get that software running

Linux/MacOS/WLS2

From he instruction provided by the developer, we can expect the software to be working only by providing one line of simple command:

```
./build-app.bash
```

During the building of the application, user can encounter multitude of errors. They have been sorted into two categories: the ones that can be experienced on the client side and the ones that could cause issues on the server side. If there are no errors in the console, the software should have been installed properly and can be executed:

```
./run-app.bash
```

Errors:

1. Client side

- While dealing with the client side of the projekt, user might experience an issue with node module called **"Node-gyp"**. Node-gyp is a cross-platform command-line tool written in Node.js for compiling native addon modules for Node.js. It my case it provided errors for missing **cflags** in node-gyp module, but the real problem was present in a different module.
- **Please ensure that you have 'node' installed on your machine to use commands listed below.**
- The real issue was caused by deprecated module Node-Sass. Node-sass is a library that provides binding for Node.js to LibSass, the C version of the popular stylesheet preprocessor, Sass. It allows you to natively compile **.scss** files to css at incredible speed and automatically via a connect middleware.
- To combat the issue the user will need to download the newest version of Node-Sass in the **client** directory of the project. First un-install the problem causing node nodule using:

  ```
  npm uninstall node-sass
  ```

- And then in can be installed again using command:

  ```
  npm install node-sass@latest
  ```

- If necessary, user can also install **node-gyp** using the same command, but changing the package name to **node-gyp**
- **If the reinstall does not fix the issue try using flag —global or -g for global installation of the node module.**

2. Server side

- Server side might be more tricky as it can execute properly, will return a warning… But when the user connects to the locally hosted app, it can return an error. Why is that? It's because one of server's modules asks for an incorrect version of **Flask** - micro web framework.
- **Werkzeug** (the module in question) is asking to be provided with the **newest** version of Flask. The user can check what version they are running by executing command:

```
flask –version
```

- Or by using:

```
pip freeze | grep flask
```

- If the version is not the newest one, please install the newest version.

```
pip3 install flask==[version]
```

- Afer installing the newest version of flask, try executing the run command again, if the errors continue to appear, please follow the next steps:
- Run application server and check for errors

```
Python server.py
```

- Go to server directory in the projekt and execute the following command to see if the Flask framework was assign to the variable correctly.

```
echo $FLASK_APP
```

- If the variable $FLASK_APP is empty or does not have the server filename inside of it, execute the following command:

```
export $FLASK_APP = server.py
```

- After those steps the server should be running

Issue of **Flask** and **Werkzeug** version conflict might be solved by modifying **required.txt** file so version designation for **Flask** is no longer provided. Without version designation pip will download most up to date versions of both libraries. This way server side should work after initial build without need of any user integration.