IGHS web

Getting Up and Running locally

1. `brew install node`
2. `npm install`
3. `[sudo] npm install -g LiveScript gulp`
4. `gulp` to prepare assets. You can leave the watch running by using `gulp dev`

5. Add the following to your `/etc/hosts`:

```
127.0.0.1       elegou.cn.tesco.com         # China
127.0.0.1       nakup.itesco.cz             # Czech
127.0.0.1       bevasarlas.tesco.hu         # Hungary
127.0.0.1       eshop.tesco.com.my          # Malaysia
127.0.0.1       ezakupy.tesco.pl            # Poland
127.0.0.1       potravinydomov.itesco.sk    # Slovakia
127.0.0.1       kapimda.kipa.com.tr         # Turkey
127.0.0.1       shoponline.tescolotus.com   # Thailand
127.0.0.1       uk.tesco.com                # UK
```

6. `npm start` will start the site *and* the mock API on ports 3000 and 3001 respectively. You can use `npm run start-test-api` if you want to just start the Mock API on its own (port 3001)

7. Visit one of the sites up above that were added to the `/etc/hosts`

IMPORTANT: Adding new PRODUCTION dependencies

We are committing *production* dependencies of the *main* app into the repository so that we can guarantee which exact versions of our dependencies are running in production. This does not include any dependencies of the test/mock API which is located in `specs/acceptance/api` or any dev-dependecies at all.

1. `npm install --save --ignore-scripts module-name`
2. remove `.gitignore` files in module: `find node_modules/module-name -name .gitignore -exec rm {} \;`
3. `npm shrinkwrap --production`
4. commit to git
5. `npm rebuild`
6. `git status` and add these files to `.gitignore`
7. commit to git and push

IMPORTANT: Adding new DEV dependencies

1. `npm install --save-dev module-name`
2. add `node_modules/module-name` to `.gitignore`

Provisioning production (Windows) servers

Application Request Routing 3.0 (with URL rewrite 2.0)

This may well be installed already. You can check because it adds an ARR icon to the features view of the root node in IIS Manager. Otherwise it can be installed using the Web Platform Installer. Open the icon mentioned above and choose proxy configuration from the right hand pane. There is a checkbox to enable reverse proxying that needs to be checked.

node.js

Install node. We need a special build that includes internationalization support from the ICU library. (See notes above). When you install node, check that npm is installed with it. Check that you can run both node and npm from a Windows command prompt (you may need to adjust the path settings).

iisnode

There is an x64 MSI link and install instructions here: https://github.com/tjanczuk/iisnode. Global dependencies The application requires the LiveScript compiler to run. Until the built artefacts are included in the deployment package, weâ€™ll also need gulp to build them.

```
npm install -g LiveScript
npm install -g gulp
```

Note: when using npm behind a corporate proxy do the following:

1. run Fiddler
2. tell npm to use Fiddler as it's proxy

```
npm config set proxy http://localhost:8888
npm config set https-proxy http://localhost:8888
```

Note: for url rewriting provider development info see http://www.iis.net/learn/extensions/url-rewrite-module/developing-a-custom-rewrite-provider-for-url-rewrite-module

Setup of production servers

Little web (mobile, node.js) and Big web (PC, ASP.NET MVC4) run side by side on ports 82 and 83. Requests are reverse proxied to them by a despatcher site running on port 80. This uses Application Request Routing to proxy requests to the correct site based on the rules listed below.

Move Big web site from port 80 to 83

This is a PowerShell command to add the Grocery site (Big web) with just the main domains modified to listen on port 83 instead of 80:

```
appcmd add site /name:"Grocery"  /physicalPath:$groceryPath /bindings:"http/*:83:Nakup.itesco.cz,https/*:443:Zabezpeceni.itesco.cz,htt
```

add dispatcher site with url rewriting rules

This site is empty apart from the web.config The URL rewriting rules in the web config, in order, are:

1. querystring ux=mobile to port 82
2. querystring ui=* to port 83
3. cookie with ux=mobile to port 82
4. cookie with ui=* to port 83
5. device family == mobile to port 82
6. default to port 83

You need to install the custom rewrite provider in the GAC so that the 5th rule above can run.

Note: when creating the 2 sites (ighs-web and dispatcher), create appropriate service account and set the relevant application pool to use these credentials

Deployment to production servers

On successful merge of pull request into master on github.

1. fetch master branch from github
2. install local dependencies:

   ```
   set NODE_ENV=development
   npm install
   ```

3. Build assets

   ```
   set NODE_ENV=production
   gulp
   ```

   The gulp build step does the following:

   1. Less -> css
   2. use browserify to package client side script
   3. use envify to suggest redundant code blocks for removal by uglify
   4. use uglify to minify JS
   5. insert content hashes into file names
   6. massage css to reflect new filenames in urls
   7. minify css

4. zip it all up
5. create MSI (Wix)
6. create manifests (partial application manifest) and reference the MSI
7. Pass to NUI, Nolio

Licenses

This software is owned by Tesco plc. It is proprietary, private and confidential and may not be not released under any public license.

It depends on open source modules with permissive licences. A list of these dependencies and their licences can be found here.

To update:

```
npm install -g npm-license
npm-license --include=all > ./licenses.txt
```

It is important that only modules with permissive licenses are used.

Important: Before including a dependency, please see check http://en.wikipedia.org/wiki/Comparison_of_free_software_licences and ensure that both columns on the right of the comparison table are green for the relevant license(s).

Development with Vagrant boxes

This project contains provisioning profiles for a quick launch of a Vagrant VM box. You can do all development within that box. This allows to quickly spawn new boxes, as well as ensures that everyone has the same environment.

Prerequisites

Following tools should be installed on your Mac:

- Virtualbox: https://www.virtualbox.org/wiki/Downloads
- Vagrant : http://www.vagrantup.com/downloads.html
- Ansible: http://docs.ansible.com/intro_installation.html#installation

If you encounter problems while installing fresh version of Ansible, check this helpful answer on SO: http://stackoverflow.com/a/22427311/668086

To spawn a new box

Run `vagrant up`. If this is your first time, this command will download an Ubuntu image, setup all the dependencies, install apt packages, run `npm install` for local and global packages, as well as `gulp` to build the project itself. It will also setup an Nginx server.

After that you can SSH to your box with `vagrant ssh`. Navigate to `/vagrant` folder where you'll find the project assets. This folder is a shared folder, which mirrors your local project folder - this way you can edit project locally, using any local software you like to write code and push it to GitHub.

All build related tasks should be performed on Vagrant box.

Modify your local /etc/hosts

On your Mac add following line to `/etc/hosts`:

```
192.168.50.60       elegou.cn.tesco.com
192.168.50.60       nakup.itesco.cz
192.168.50.60       bevasarlas.tesco.hu
192.168.50.60       eshop.tesco.com.my
192.168.50.60       ezakupy.tesco.pl
192.168.50.60       potravinydomov.itesco.sk
192.168.50.60       kapimda.kipa.com.tr
192.168.50.60       shoponline.tescolotus.com
```

Run the app

SSH to the Vagrant box, `cd /vagrant && sudo lsc index` After that try opening http://elegou.localhost:3000 with your local browser.

`gulp` command will be run during the provisioning, but you might want to run `gulp dev` watcher, and just leave it running in background - this will watch for style changing and recompile necessary files on the go.

Handy vagrant commands

- `vagrant halt` - shut down the VM
- `vagrant destroy` - destroy the VM with all the settings
- `vagrant up` - (re)build the VM from scratch, or just launch the provisioned box if it is ready
- `vagrant provision` - reprovision VM

Vagrant Troubleshooting

After destroying and rebuilding vagrant box, you will likely encounter SSH related problems when Ansible will try to provision the box. Resolve this by modifying `~/.ssh/known_hosts` file - simply remove the line starting with `192.168.50.60` address.

On `vagrant up` sometimes you might get following error:

```
Failed to mount folders in Linux guest. This is usually because
the "vboxsf" file system is not available. Please verify that
the guest additions are properly installed in the guest and
can work properly.
```

Here is a helpful answer to this issue on SO: http://stackoverflow.com/a/22723807/668086

If you get SSH connection timeout errors, try adding this to the `/Vagrant` file:

```
config.vm.provider "virtualbox" do |v|
  v.memory = 2048
  v.gui = true
end
```

This will enable GUI of the box, and you can check if the Ubuntu actually asking you for something during the boot. Related SO question:
http://stackoverflow.com/questions/22575261/vagrant-stuck-connection-timeout-retrying

Running the BDD tests locally

Short description how to run the automated tests locally

Prerequisites

PhantomJS,CucumberJS and Selenium-webdriver should be installed on the local environment (npm install does that)

Start the headless browser (phantom js)

- Run this command in the command line: `phantomjs --webdriver=4444 --ignore-ssl-errors=true`

Run the tests

- You should stay at the specs\acceptance folder
- If you want to start all the tests (it will make sense when all the tests are ready in the new world as well) run this: `cucumber-js -f pretty`
- If you want to start specific tests with a certain tag, run this: `cucumberjs -t @tag,@tag2,@etc -f pretty`

Tags for test runs:

- @full - for the complete test suite
- @signin - for the sign related in tests
- @signout - for the signout related tests
- @forgottenpassword - for the forgotten password related tests
- @plp - plp related tests
- @bookaslot - book a slot related tests
- @homepage - homepage related tests
- @footer - footer related tests
- @registration_1 - Registration phase 1 related tests
- @registration_2 - Registration phase 2 related tests
- @registration_3 - Registration phase 3 related tests

- @registration - every registration related tests
- @search - search related tests

Should you have any question, feel free to ask @ukpeti