Report

Live demo

http://testforcoolgames.co.nf/

Source code

https://github.com/vtsitlak/bestgames

Technologies used:

HTML5, Less, Bootstrap, JavaScript, AngularJS, jQuery, npm, bower

Installation:

npm install

gulp

Development:

All source files, JavaScript, AngularJS, images, HTML5 are on src folder. Public folder contains all the ready files for the project. Npm installs gulp and several gulp plugins that helping with the development. Bower installs libraries and packages for the project. When running gulp, several tasks are running and extracts the final project on Public folder.

Gulp tasks

First task, 'bower', installs all bower items from bower json file to bower_components folder.

Task 'fonts' move fonts to public folder.

'htaccess' moves .htaccess file to public folder

'html' minify and moves all html files to public folder

'json' task minify and moves json file to public folder

'scripts' concatenate, minify and moves JS files to public folder

'less' compiles to css, concatenate and minify less files

'css' task concatenate the previous less compiled file with the rest css files from several packages and libraries, minify them and moves the final result to public folder as main.min.css

'img' optimizes the image files and moves them to the public folder

'watch' task watch for changes on files and updates the files, is useful while in production, you can comment out this task so that gulp task stops run.

Bower components

jQuery: The library for jQuery

Angular: The library for AngularJS (1.5.8)

Bootstrap: The library for Bootstrap

Font-awesome: Font awesome package

Angular-Bootstrap: An Angular and bootstrap library

Angular-Animate: An angular package for animations

Angular-ui-router: An angular package to handle routes

Angular touch: An angular module to add directives for touch devices

Angular-metatags: Module for providing dynamic Meta Tags to Angular routes

angular-rating: a module to help you create a rating with some customizable options

angular-sanitize: AngularJS module for sanitizing HTML

JQuery components

Used the flagstrap plugin for creating Bootstrap 3 compatible country select boxes with flags. Handled with that the language selection on navbar.

Project structure

Use of separate folders for separate file types. There are folders for HTML, JavaScript, Images, fonts and Less and Data (json) files.

On HTML folder there is the index.html file and on ‘parts’ folder there are the templates for the rest of the pages and angular directives

On Less folder we use the main.less file to import all less files

On JS folder there are two subfolders, one for jQuery plugins and the other one for the AngularJS scripts. On Angular folder there is the app.js for the module and the routes, and there are separate folders for the controllers, services and directives.

Main projects functionalities

Design: Design is very similar with the original page. Is a fully responsive mobile first web site. The main changes are on navigation menu, where the ‘my game’, set language and ‘join’ pages are hidden on the expandable hamburger menu.

Carousel: I used the angular-bootstrap-ui carousel. The difference with the carousel on the original page is that I kept the left and right control, so that the user be able to navigate manually on the feature games. I also add the game description on the bottom of the carousel. I used the angular-sanitize to sanitize the HTML code from the game description tag on Jason file. I also set different colors to the three button, play, play later and favorite, to be easily for the user to recognize them fast. When the user clicks on favorite button then the heart icon becomes red, you can click again to become back white.

gameItem: For each game item we see the icon, the title and the star rating, using the angular-ratting directive. On some cases where we have broken links for the icon, we use the imageErrordirective to set a default icon.

Data loading: the data from json file are loading with the http.get command from the gamesService

Routing: Using the angular-ui-router and set the html5 mode to true to avoid the '#' symbol. Needs to manually edit the .htaccess file to set the index.html location, and also define the base href location on index.html file, depending on server structure.

Site optimization: All html, JavaScript, css, images and json files are minified and optimized using the gulp tasks.

SEO : Added description, author and keyword metatags. When we create dynamic a new state for the game links, we add the game dynamic to these pages the name, seoKeywords, description, categoryIds, tags and id of the game, as they are on json files. To add dynamic these metatags we use the Angular-metatags module, just after we create dynamic a new state, on stateService. We also add the same texts on hidden elements on Carousel and on gameItem.

Link to games: The games are loading inside the page, by creating dynamic a new state when the user clicks on play button, or on gameItem icon. Using the json data about the game we call the stateService and creating there the new state, if not already exist.

IMPORTANT! To prevent the No 'Access-Control-Allow-Origin' header error I used a CORS proxy, the game url link is 'http://cors.io/?' + game link . Because of the limitations of the CORS proxy, and also from the limitations on the free server that I have used for the demo, the game links often are not working, or working very slowly. Also I think that for security reasons from your server some links are not working, as I am getting a wrong key message sometimes. The dynamic creations of the states and the metadata are working without problems. You can comment out the addState function on mainContoller and uncomment the window.location.replace function to just go to the external link of the game