CS3715 - Group 6 – sc-6 - port 3336

GIT Repository: <https://github.com/CS3715Group6/steve-university-server>

Main page : [sc-6.cs.mun.ca](http://www.cs.mun.ca/~ajg468/EduApp_3_6/)

Our application includes seven main html pages:

* index.html (main university page)
* eastern.html
* northern.html
* southern.html
* directions.html
* student.html
* preCourses.html

Index for displaying the main university page, three campus pages describing each campus, student for displaying student registration data, previous courses for displaying courses no longer offered and a directions page to actively display a route from the users current location to the university.

Each of the seven pages is styled using two css files:

* University.css
* Campus.css

These css files provide a slightly different look to each of the pages but provide a similar overall theme. To improve readability each file includes media queries for a number of common display types including mobile phones (portrait and landscape), tablets and desktop.

Content for each page is rendered both statically and dynamically. Dynamic content is processed and then rendered using the following javascript files:

* addEvent.js
* campusDescription.js
* campusinit.js
* courseDisplay.js
* directions.js
* indexinit.js
* preCourseInit.js
* registration.js
* studentinit.js

In the original implementation, registration information was stored in the browser local storage using an array for accuracy and efficiency.

In the server phase of this project node js was used to implement a server to provide webpages to the user. The server was built using the four main components described in the nodebeginner textbook and listed below.

* index.js
* router.js
* server.js
* requestHander.js

To implement our project we have updated and modified each of the above components to load the html pages and associated files described above. The index file was updated to include filtering for various file access requests, which are routed to the correct handlers using an updated router file. Once the request is routed the request handlers provide file access using the node file system utilities to read the requested file (html, js, css). The response object has been included as a parameter in the request handler allowing the data to be read asynchronously from the node file system utilities and then written to the response object using the write method. During this phase, json files were migrated to node JS file system storage utilities.