**Special Intro**:

Js important stuff:

**which** var means the object we are talking about right now, for example the user clicks on add => student, which will be that value and all the functions act the way they have to, depends on our -this time- **which**.

Endpoints:

Endpoint: *localhost: 8000/index.php* => (localhost:8000)

|  |  |
| --- | --- |
| Description | Method + Page |
| Our routing main page and page Template. he gets user URI and determine which page to require, this page also includes some switch cases inside, depends on user session for example- what part to show of nav bar and if to show it at all. scripts and more...  Scripts are depending on page URI, you can find them at bottom of the page where is switch case. | ROUTING  Cases: [ '/login' , '/home' , '/']  This will route to /Viewers/login.php  --  Case: [ '/school']  This will route to /Viewers/school.php  --  Case: [ '/administration']  This will route to /Viewers/ administration.php  --  Everything else:  /Viewers/404.php |

Endpoint: localhost:8000/login.php

|  |  |
| --- | --- |
| Description | Method + Page |
| This page purpose is to log in to an existing admin user, it happens via some Js validations and then ajax call to check in Db.  --  First: check Email format with php filters, no touch in DB.  --  Second: check if there is a match in admins table (email , password) | 2 calls.  First:  POST  Controllers/loginControl.php  Triggered by function: checkEmailFormat();  who's called by try to log in function (user calls in via click on button or enter).  -----------------------  Second:  POST  Controllers/loginControl.php  Triggered by function: matchMarker();  previous ajax call it if email format response is positive. |

Endpoint: localhost:8000/school.php

Endpoint: localhost:8000/administration.php

--

Both working on same script file which called by index. Same functions again depend by **which** var.

--

In school page **which** can be "student" or "course"

In administration page **which** can be "administrator"

|  |  |
| --- | --- |
| Description | Method + Page |
| Those Pages are holding the index template article part, and some Db call requests happens in their relevant javascript to php ajax calls. Coming from js file, called mainPages.js  --  DB calls:  --  **1st**: function to trigger:  getAllAsked();  -gets objects to show in viewer left scroll  --  **2nd**: function to trigger:  cookCoursesIfNeededToChooseSave();  -get courses to select when creating new student  --  **3rd**: function to trigger:  checkUploadedImage();  -check potentially image to upload file type and size  --  **4th**: function to trigger:  checkCourseNameExistence ();  -check course name existence in Db, course name is unique.  --  **5th**: function to trigger:  checkEmailFormat ();  -check email in Db, email is unique. | Calls:  Main purpose is on red.  **1st**:  POST  Controllers/adderAndAll/' + which + 'sController.php  - gets **which** var onload of page, sends it and ask Db for all objects in the relevant table for showing them in our objects scroll glila.  for example: all students, for showing them to user at left of the main container in school page. Php brings back objects array along with **which** var for next function getSpecificObject(); onclick event of objects div.  ------------------------  **2nd**:  POST  Controllers/adderAndAll/coursesController.php  - gets **which** var from user clicking add button from our left side scroll bar, its detecting cases if it’s a student it goes Db and ask for all courses for option to select courses when creating new student, cook var texture for next function to be checkbox, holding course id in it.  Sends it to showAddCont(); function.  In other which cases, it's just send which var to showAddCont(); function.  -----------------------  **3rd**:  POST  Controllers/multipleController.php  -gets onchange of input file on adding / editing an object form and checking if it is a valid image and size for help us determine our validations for uploading an image.  -----------------------  **4th**:  POST  Controllers/multipleController.php  -gets onblur event input name of course value and check if course name is taken or not. Courses name's are unique.  -----------------------  **5th**:  POST  Controllers/multipleController.php  -gets onblur event input email of object value and check if email is already taken or not. emails are unique between students and administator. Fully unique. |

|  |  |
| --- | --- |
| Description | Method + Page |
| --  **6th**: function to trigger:  checkEmailFormat ();  -check phone in Db, phone number is unique.  --  **7th**: function to trigger:  XHRcall ();  - getting a from to add to Db(object for example admin) from SaveObject function(does all validations together with previous functions).  --  **8th**: function to trigger:  getSpecificObject ();  - gets from XHRcall of add or from ajax call of edit data an object id and which to collect details about him from Db  --  **9th**: function to trigger:  showData ();  - gets getSpecificObject function an object array [0] is object data and sons array(courses for students) and etc, and in cell [1] theres which var  --  **10th**: function to trigger:  cookCoursesIfNeededToChooseEdit();  - called by show object data container EDIT button.  Gets object and in student case ask DB to get all courses are able to choose as student and send it with object to our function showEditCont(); there it will run on foreach of selected course and say true to the right checkboxes. | Calls:  Main purpose is on red.  **6th**:  Same as 5th, with phones  --------------------  **7th**:  POST  Controllers/adderAndAll/' + which + 'sController.php  -gets formData of adding object from add form, triggered by SaveObject functions after all needed validations while **what** var is add.  -sends form to php and let php to add asked image file to folder, saving object data in Db and at end of response to give the right answer, if something happened then what happened and let user know it.  If everything went smooth. XHRcall trigger two functions: getSpecificObject() who gets an object with all needed data to show it to user and **which** var back from php. Second it triggers getAllAsked() function with which var to show all objects in glila.  -----------------------  **8th**:  POST  Controllers/GetSpecific/' + getObject.which + 'Controller.php  -gets **objectId** and **which** var and sends it to controller in php to gather data about object, collecting for students and courses their relations from deals table too.  Sending object got back from php to function showData();  -----------------------  **9th**:  Actually there's no request in 9th function, just to notice it determines with which var how to build the form to show our object, and it uses at bottom of function also object role and user role to check what he is allowed or not.  -----------------------  **10th**:  POST  Controllers/adderAndAll/coursesController.php  -call db to get all courses to check for our edit cont student. In other which cases, moving on without even calling post.. |

|  |  |
| --- | --- |
| Description | Method + Page |
| --  **11th**: function to trigger:  saveNUpdate();  -$.ajax call  - getting a from to update in Db(object for example admin) from SaveObject function(does all validations together with previous functions).  --  **12th**: function to trigger:  removeExistingImage ();  -$.ajax call  - quite before updating an object, checks if chose to remove existing img or not, in case he does, it goes to object db and change image value to NULL (default) and delete it locally.  Very short one.  --  **13th**: function to trigger:  checkIfFoundAdmin ();  -$.ajax call  - before transferring ownership checks if email to transfer is valid.  --  **14th**: function to trigger:  trasferOwnership ();  -$.ajax call  - after validation check, transfers ownership to selected admin via email. | Calls:  Main purpose is on red.  **11th**:  POST  Controllers/existEditor/" + which + "Controller.php  -gets formData of editing object from edit form, triggered by SaveObject functions after all needed validations while **what** var is edit.  -sends form to php and let php to delete old object image if needed and replace it. saving new object data in Db and at end of response to give the right answer, if something happened then what happened and let user know it.  If everything went smooth. will trigger two functions: getSpecificObject() who gets an object with all needed data to show it to user and **which** var back from php. Second it triggers getAllAsked() function with which var to show all objects in glila.  **13th**:  POST  Controllers/existEditor/administatorController.php  -gets onblur emailToCheck if exist but not owner, (check via id) and returns if exist or not.  **14th**:  POST  Controllers/existEditor/administatorController.php  -gets onclick event, takes email of existing admin who chosen to be our new owner and replace his role to owner, and previous owner to manage.  Once it succeed it logs out from former owner account |

Endpoint: localhost:8000/deals.php

|  |  |
| --- | --- |
| Description | Method + Page |
| This page purpose is to give Sales Administrators a page for themselves, where they can see all the courses-students relations and do their Own Statictics. Used dataTable Plugin.  --  First: get all deals and set them on table;  Function to trigger: getAllDeals();  --  Second: get all courses from courses table to check which courses have sales and which are not.  Function to trigger: getAllCourses(); | 2 calls.  First:  POST  Controllers/multipleController2.php  Triggered by function: getAllDeals ();  Called onload of page.  In Db it asks deals Table for all existing deals and gets all Students and courses Id's  It organize it together as an object in an array and send it back in the ajax response.  Second:  POST  Controllers/adderAndAll/coursesController.php  Triggered by function: getAllCourses();  Called when deals table finished to built,  Purpose to get all courses to send it to createGraph Function(), which compare each course and deal name, check how many times it appears and calculate it, display it on screen. |