Isaac Betancourt – Development Standard

Backend Code (Server)

Standard for code development.

The standard from which we are going to be governed to make more understandable and readable code for people who are in the development area will be the following:

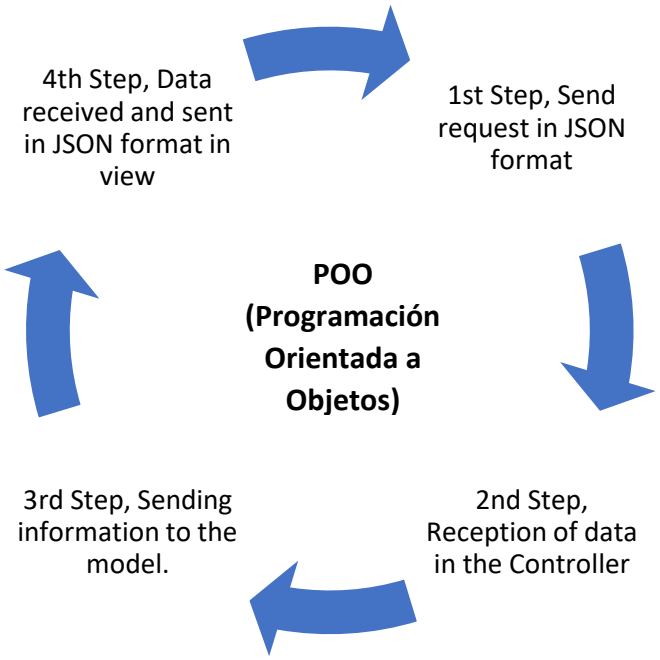
Names and Organization of Files Within the Module

- **Model** : Everything related to Database interaction.
- **Controller** : Everything related to the logic of the System.
- **Sight** : Everything related to sight.

Interaction sending process with the backend from the frontend

- Request captured with javascript with the corresponding validation of the form
- Information received by the controller, the entire necessary process is carried out.
- The information is returned in JSON format for the view

Scheme:



Development Nomenclatures

camelCase (camel script)

Spaces and indentations

- 2 lines of spaces must be applied between classes
- 1 line of space must be applied between functions
- Each start of the function must begin with a comment indicating that it performs the process.
- All comments, class names, functions and variables must be written in English.
- All function, class and variable names have a name related to the system logic
- All external files, called libraries, must be at the beginning of each file
- hierarchy indentations must be respected , starting from a single indentation .

Examples:

```
////////////////////////////////////////////////// ////////////////////
////////DEVELOPED BY: User Sample ////////////////////////////////
////////DESCRIPTION: Description//////////////
////////////////////////////////////////////////// ////////////////////

include ( " testFile.php " );


class classExample {
//Comment example
      private $ lc_connect ;
      private $ lc_dates ;
}


functions nameFunction ($Arg) {
      //Comment example
      $var = 'Example';
      $result = 'Text';
            If( $var == 'Example'){
                  $result = '1';
            }else {
                  $result = '2';
}
      return $result;
}

    functions nameFunction ($Arg) {
```

```
      //Comment example
      $var = 'Example';
      $result = 'Text';
            If( $var == 'Example'){
                  $result = '1';
            }else {
                  $result = '2';
}
      return $result;
}
```

HTML (FrontEnd)

CSS styles

- All CSS styles have to be called apart in the top section of the document, they should never be used in the HTML tag line.
- Once the CSS is structured, it must be minified for execution.

javascript

- javascript files (Libraries) must be called at the bottom of the document.
- Must work with ES6
- All events and data sending to the server must be made with AJAX or FETCH requests in JSON format. **Note:** it is a standard to send data with JSON, but if it is required to send it in another format it can be done.

Structure and Indentation :

- indentation space must be made for each main label block, followed by the children.
- No spaces allowed between tags.

Security in development

5 layers of security are implemented in the code structure:

1st layer : CSS Style; security layer that they carry out at a visual level from the client.

2nd layer : Javascript ; The corresponding validations are carried out from the frontend with javascript , data type validation and shipments.

3rd layer : Validation with the browser's native HTML tags

4th layer : Validation with the server; Once the data is received, validation is carried out, this includes the **CSRF token, captcha,** among other methods.

5th layer : Validation by the database server, which includes the type of incoming data and the execution limit.

Database

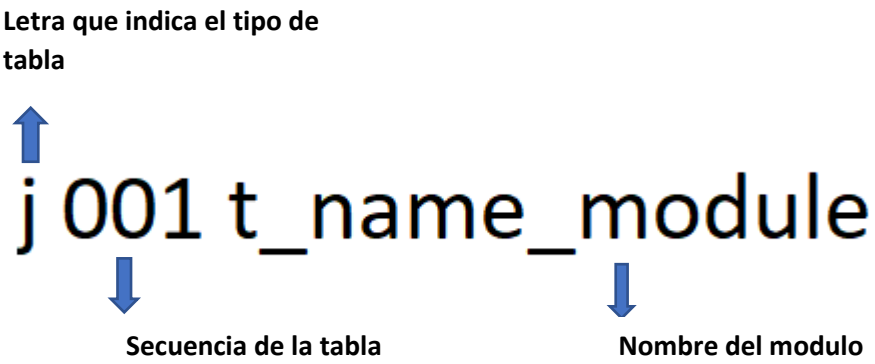
Database structures

The database name must be related to the project followed by the version.

Table structures

- **Base tables** : they are tables used as a catalog, they are tables with static data that can be called by several modules, (some to many)
- **Transactional tables** : they are tables used for the insertion of constant records, (many to one)
- **Intersection tables** : they are tables that are used to join two tables using a record, (many-to-many)

Example of nomenclatures:



i, e : base tables

x, j : transactional tables

c : intersection tables

Git (Version Control)

Name of the branches

When we work with git what we will do is create new branches derived from the version in which the correction will be made, with which the name that will be used for this new branch is **YEAR**

MONTHInitialsCommiter / TareaAsignada , for example 2017-05HO/FDU-1000 , that serves the following:

● Git indexes the branches by folders, so what we achieve with this is to organize the branches by folder and make git find them faster.

● Have better control of the tasks we will perform when correcting bugs.

● Recognize who uploaded such change and on what date the correction was uploaded to the branches.

Link the branch to the task in question. Commit messages The prefixes for the commits will be the following:

● [IMP] for improvements

● [FIX] for bug fixes

● [REF] for refactorings

● [ADD] to add new resources (files, folders, etc.)

● [REM] for resource removal

In the message, specify which part of the code is being impacted by the changes (module name , lib , transversal object , ...) and their description. Always include a meaningful message: it should be a long enough explanatory message including the name of the module that has been changed and the reason for these changes. Do not use words like " bugfix " or "improvements".

● Avoid commits that impact several modules simultaneously. Try to split into different commits where the impacted modules are different (It will be useful if we need to revert module changes separately).

Structure and work scheme

Operating systems:

- Windows 10
- Linux Ubuntu

Code editor:

- Visual Studio Code
- Sublime Text

Test and Run Navigator:

- Google Chrome

Software structure and equipment:

- Processor: Ryzen 7 Processor 4.20 GHz
- RAM: 80GB
- Monitor: Phatom 27”
- PS3 eye 1080 WEB camera
- Headset with microphone.
- Storage capacity : 4 TB.
- Internet speed : 80MB

Note:

It should be noted that this is a standard structure of personal development, this can change if you use technologies such as:

- Backend development framework such as LARAVEL, CODEIGNITER, DJANGO, among others.
- Frontend like VUE, JQUERY, ANGULAR.
- Interaction of the models with the ORMs of each Framework.