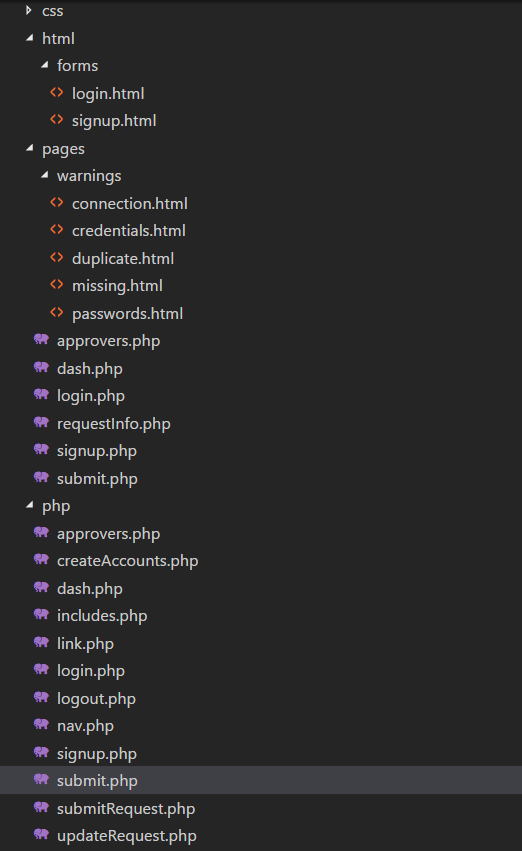
# **Refactoring Report for Group Eh?**

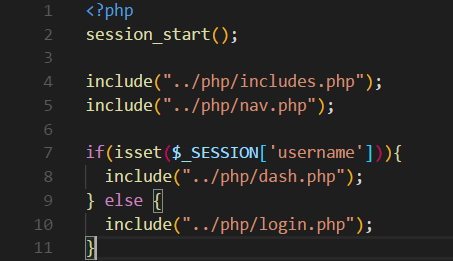
## **ENSE Master Race**

Ultimately, Group Eh?'s file structure was exceptionally well organized and the code was clean and smell-free. *Figure 1* below shows how well the file structure was organized, with all files being contained in folders and no floating files.

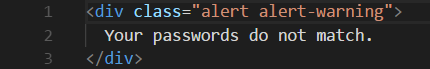




While it is hard to use object-oriented programming for web development, Group Eh?’s code follows a component-based architecture, building each page with multiple smaller components, similar to composition in OOP ( *Figure 2)*. Along with that, smaller sections such as error code messages were broken down into their own components and could be substituted anywhere in the code if needed (see *Figure 3*).





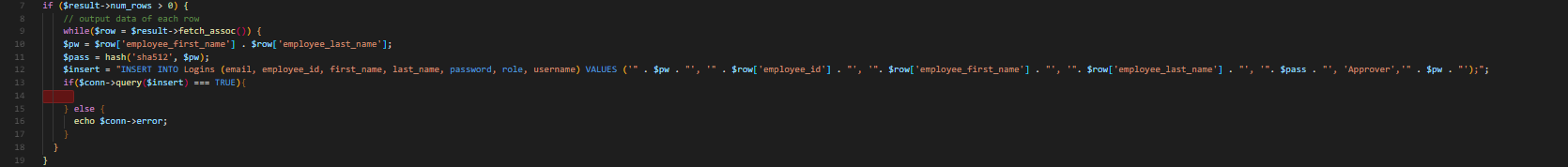




Another aspect regarding the overall architecture is that every component is completely decoupled from each other. Any information a page needs, it fetches. This isn’t to say this is clean code or dirty code, as there are both positives and negatives to this approach. It doesn’t scale amazingly well with more users making concurrent calls as the database then becomes a bottleneck, but for an app of this scale, it makes the code easy to understand.

For the actual code part, ever parameter is named reasonably and clearly. This makes it easy for new programmers to understand. Long methods and long parameter lists are also absent. For maintenance, none of the components is change prevented. Programmer doesn’t need to change stuff around to suit one new change.

In the end, there wasn’t any “dirty” code, and anything mentioned here is simply nitpicking, but it is still worth mentioning. The first is that in a handful of files, there were some excessively long lines of codes, generally SQL statements, that could have some line breaks to make it easier to read (see *Figure 4*).





There was also a single instance of repeated code that could’ve been pulled out (see *Figure 5*).





The last point here is extra nitpicky, but as is the nature of PHP, there were many occurrences where PHP code was nested within HTML or vice versa, and some blank lines are helpful to easily differentiate between the two languages (see *Figure 6* and *Figure 7*).









Ultimately, there are not really any refactors that need to be made.