Star Wars website – Data Entry Optimization

Felipe .M Iglesias



CONTENT

[EXECUTIVE SUMMARY 2](#_Toc187690744)

[SUMMARY OF FIXES 3](#_Toc187690745)

[IMPLEMENTATION 4](#_Toc187690746)

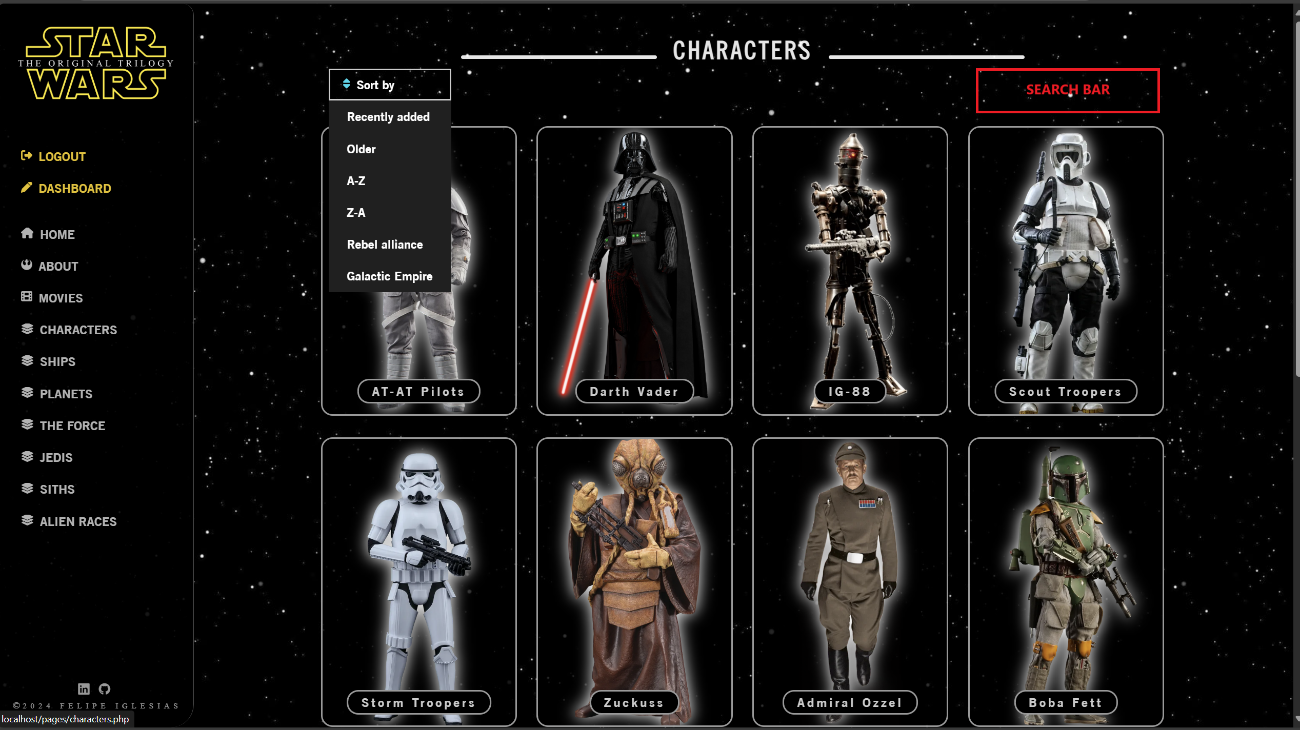
[DESCRIPTION OF CHANGES 6](#_Toc187690747)

# EXECUTIVE SUMMARY

In this document data entry for the Star Wars website project will be reviewed.

# SUMMARY OF FIXES

A way to optimize the data entry would be to implement a search bar in each data display section. Through this, the user will be able to quickly find any record by just typing any letter related with it. If he fails the search due to a typo or a non-existent data, the option to make another search from the same page he is now would be possible. Moreover, the typed word could be shown while the search is going on, in this way it would be easier for him to correct any typo. In case the search is not successful, a message with guidance about why it wasn’t successful would be displayed, helping the user to improve the search.



# IMPLEMENTATION

To implement the search bar, I added a form to each data display section. The form is built with just an input that is submitting text. I set the *name* of the input to **“keyword”**. The method used is **GET**, as the data entered is not personal shouldn’t be a problem that is being showed on the URL bar. I decided to use this method as well to avoid the inconvenience of resubmitting the form each time the user wants to go back during the navigation. I implemented the **GET memory** held in the variable **$keyword** as the *value* of the form. In this way the user can keep track of what he typed once the form was submitted. (It is working for each public and private sections in the same way).

A screenshot of a computer program

Description automatically generated

From here, the **GET memory** is sent to a file named **“charactersPublicSideSearchByName.php”** and held in the previous mentioned variable **$keyword**.

A screen shot of a computer

Description automatically generated

The function that is making the search possible. Is named **searchCharacterPublicSide($dbConnect, $keyword, $succesfullSearch)**. Besides the database connection and the $keyword variables, is using another variable named **$succesfullSearch** which is allowing to display the error message in case the search is not successful. It is set to false before the iteration through each row on the data base starts. Once started, it turns into true independently if the search was successful or not. At the end of the search, **if not** successful, a message will be displayed giving the user some guidance on how to improve the search.

The variable **$keyword** is surrounded by the **%** **wildcard**, in this way, will match any string that contains the keyword the user submitted anywhere within it (beginning or end). For example, if the user is looking for ***“wan”***it will match **“**obi-**wan** kenobi**”**.

A screenshot of a computer program

Description automatically generated

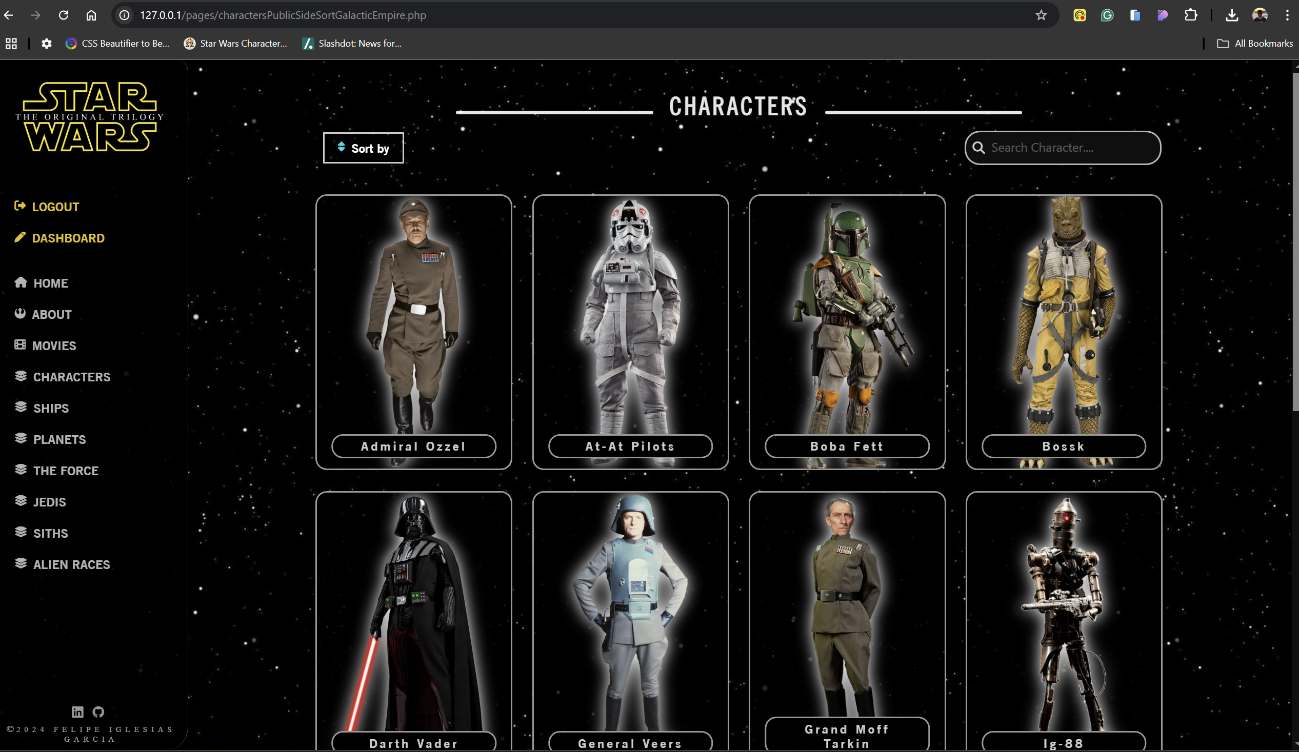
This function is called in **charactersPublicSideSearchByName.php** to show the results.

A screenshot of a computer program

Description automatically generated

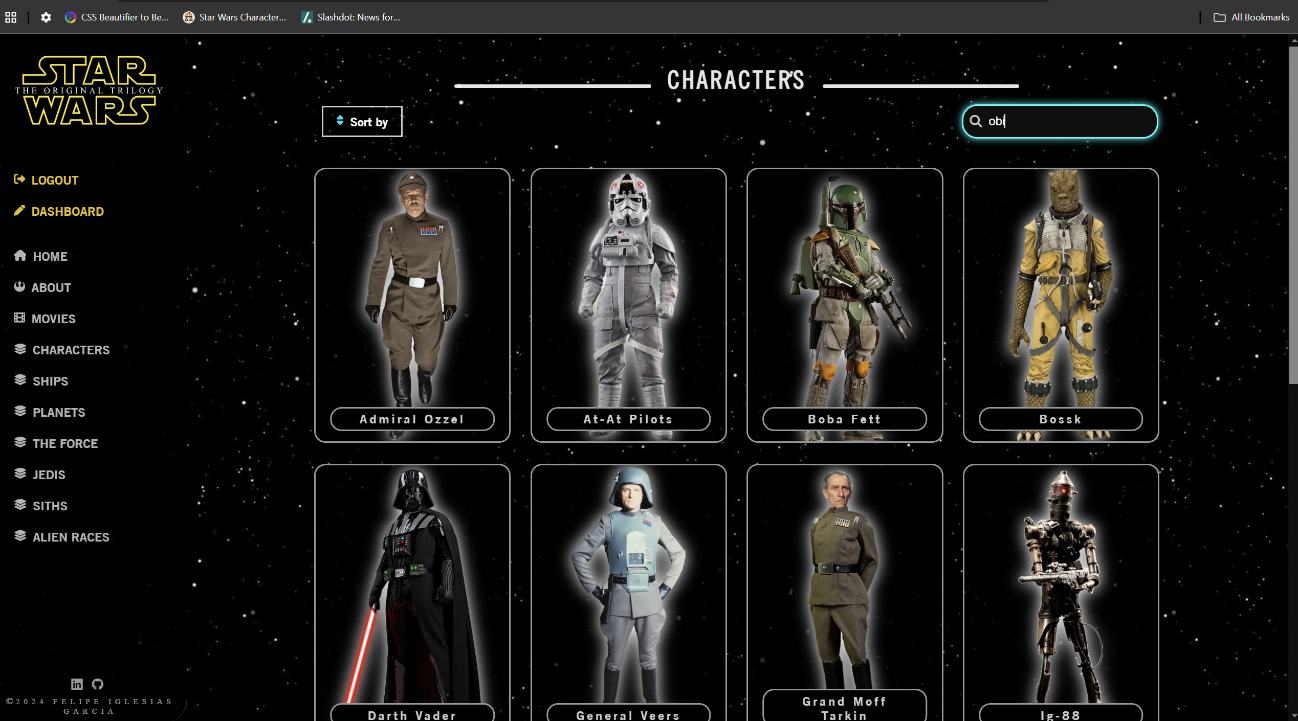
# DESCRIPTION OF CHANGES

Search bar implemented on the top right side of the navigation for a quick and intuitive location.



Functionality demonstration:

The user types “obi” on the search bar.



The results of the search are showed with a message that displays the keyword used on the search. In addition, on the search bar the user can see the last search as well.

A screenshot of a video game

Description automatically generated

Instead, if the user fails on the search due to a non-existent record or a typo, a message giving him some recommendations is displayed.

