Developer’s log:

**22/04/2024:**

Sections implemented and well commented:

Employee

Manager (Serves both MG1 and MG2)

Securities:

All the pages URL parameters are checked, none of them displays any PHP error, invalid URL parameter leads to 404 error or other trap states.

SQL queries does extra checks like, to prevent invalid access, details about these checks are well commented in the source code.

New additions:

System to support Leave Reason.

Current system:

Everyone is considered as an employee. Hence, HR, Mg1, Mg2 can login either as an employee or as per their respective ranks.

In employee section leave requests can be posted.

Mg1 views those Leave requests at first then gives his consent.

Mg2 views them after Mg1 reads his consent and approves or gives his consent.

We have used a system “self-redirect once” to clean up previous inputs of a form while going back to it. This system is used only with emp and leave rule input systems as there we have greater chance of entering same data twice and its very bad, but we don’t use it with leave request form as multiple same leave requests input won’t cause a problem, employee can delete unnecessary ones very easily.

Current issues and Future solutions:

1. Leave requests by mg1 will be seen by himself at first, we need to fix it so that they are only seen by mg2. Solution: We keep track of the original rank of an employee when he enters as an Employee with session variables. For mg1 we fill up mg1 consent automatically during his leave request application.
2. We still don’t know who will approve mg2 leave request.
3. Till now we haven’t implemented the concept of departments.
4. Mg2 should be able to see rank of the employee who makes the leave request.

**25/05/2024:**

**Here we describe some checks that must be made while receiving data from a form in PHP**

* Check if all the required inputs are given or not. Even if you add required to the HTML tags or do any kind of frontend checking, never ignore this checking as the HTML tags can be modified very easily before submitting the form.
* If the length is important, check it, but for most of the inputs length is unimportant, if user somehow enters a very large input, add constraints to the table your database to make sure that such inputs gets rejected and form submission fails. And don’t forget to let user know what is the maximum size allowed.
* There may be some other aspects of the inputs that you should check, don’t let them go unchecked.

*(in current project only emp input form has all these checks, due to shortage of time)*

**04/06/2024:**

**Auto refilling the form while asking user to input data again during an error**

If user inputs any invalid data and form submission fails user may receive one of the two types of error messages,

1. Error messages directly under the input fields of the form, mentioning what went wrong (for little issues).
2. A message in separate page describing what went wrong (for bigger issues).

How things were done in past?

**For type 1 error messages**, we used to redirect to the same form using **header(“location: …”)** function and trigger (display) required error messages by sending certain URL parameters. But the input fields in the new form (one with the error messages) would remain empty, and the previous inputs (both valid or invalid) will be lost.

**For type 2 error messages**, we used to call **die(“…”)** with the error message as its parameter, when that error occurred the program would stop and the error message would get displayed in a blank webpage. Only disadvantage of this approach is that, if this error message is displayed after a form submission, it will give form resubmission warning each time you reload or go back to this page containing error message.

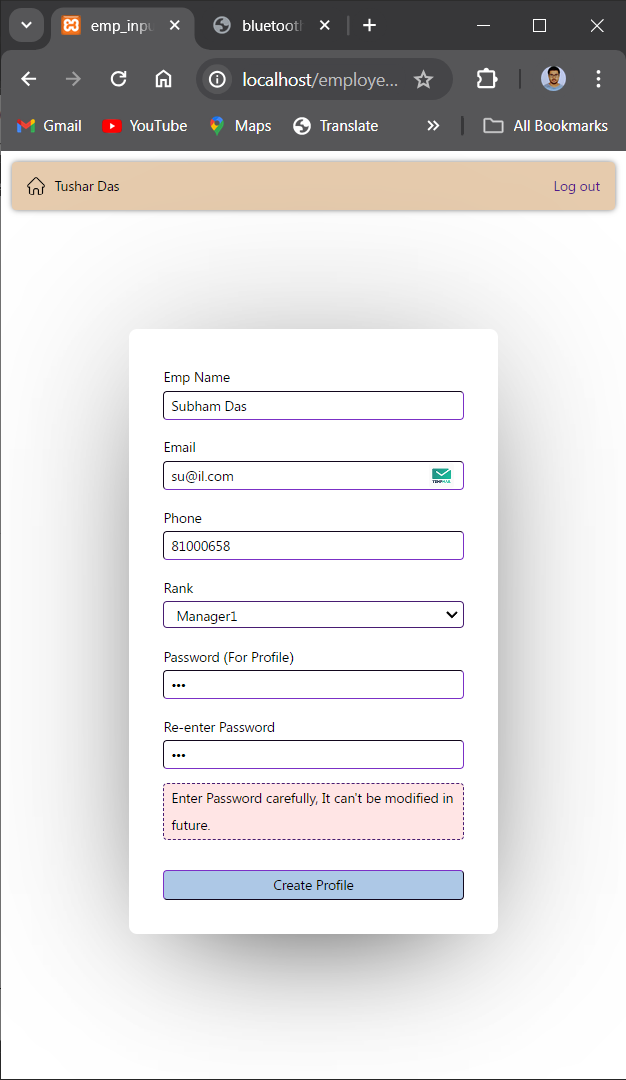
How things are done now?

We have added a **system to recall the past inputs in case of an error this helps both type 1 and 2 errors messages**. When the form is submitted, we keep backup of the inputs in as a session variable. Only if the inputs are processed successfully, we delete the back up. If some error is detected the back up remains untouched and is fed into the input fields, the first time the form is loaded after the error and is deleted afterwards.

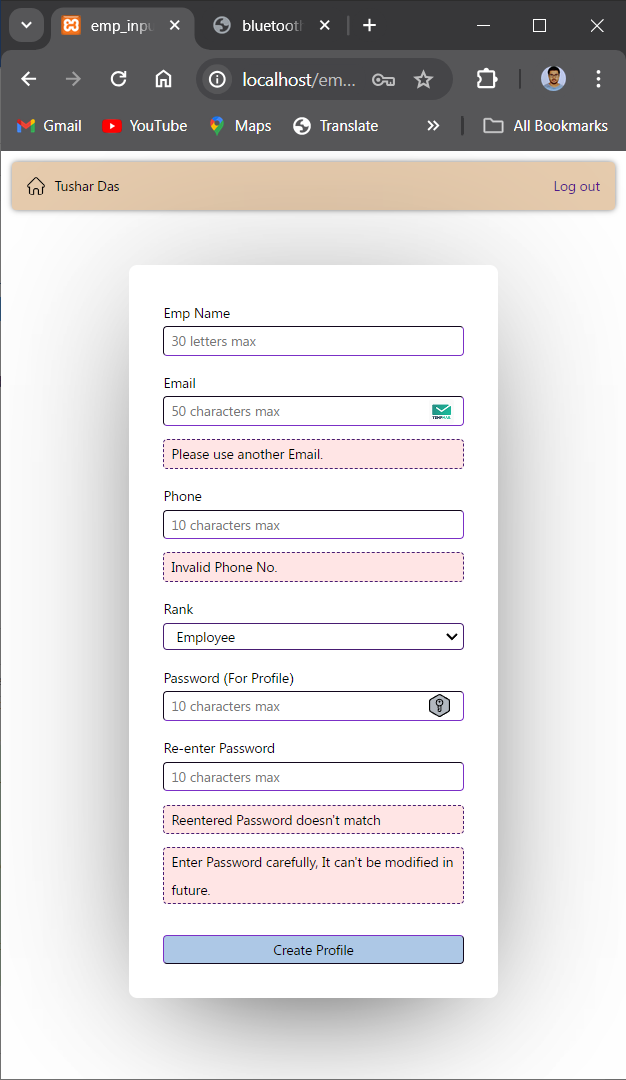
**Type 2 error messages are also updated**, instead of using die(“…”), we create a new webpage, that receives a message to display and a URL to attach to an OK button. To use it we define a function that receives the message and the URL and redirects to that webpage. After the reading the massage user can click the OK button to reach next web page according to the URL.

*(in current project only emp input form and a few others have the new feature, due to shortage of time)*

Form with invalid input:



After submission (old version) [doesn’t retain the inputs submitted]:



After submission (new version) [retains the inputs submitted]:

