This is a long term project. This is the first phase. If you can do a good job in this first phase I can give you repeated projects.

Please watch this video first :-

<https://drive.google.com/file/d/1qOMk-ERMU0ObXHD8WYizjV9HGq-3MSUX/view>

I want to create a common report output form which can be used by all reports to display the report output. All the php reports name will be stored in a table called Reports together with the description of the report and the location of the report in the web folder. This table will be manually updated by me directly in the phpadmin table. Here is the table format for reports :-

|  |  |
| --- | --- |
| **Report Table Columns** | **Remarks** |
| Report Title |  |
| Description |  |
| File name | Name of the php file name |
| Location | Web folder where the php file is physically stored |

When the user logs in, he will see all the report names listed for him first in the first page. It should be read from the reports table and displayed in the page. For example if there are 2 reports called customer sales and material listing in the reports table, then the following will be displayed :-

**List Of Reports**

Customer Sales **<Run>**

Material Listing **<Run>**

The logic of the report can be different from report to report but all reports will finally call the common output display form by passing the final report output table to show the results. For sample logic of such reports please check the Appendix A below.

When the user runs one of the these reports, the php file stored in the report location should be executed and a common report output form should be displayed so that all reports will show the results in an uniformed report display format with the same report layout functions.

This common form will be called by all reports by passing a report output table. The common form will accept the report output table as an input parameter and display the table contents in a grid as follows :-

**Common Report Output display modeless form**

{Display the report output table values in a grid}

For example this is a sample report output from customer sales report :-

|  |  |  |  |
| --- | --- | --- | --- |
| **Customer Code** | **Customer Name** | **Invoice Number** | **Amount** |
|  |  |  |  |
|  |  |  |  |

Another sample of report output for a different report called Inventory Listing :-

|  |  |  |
| --- | --- | --- |
| **Material Code** | **Material Name** | **Quantity** |
|  |  |  |
|  |  |  |

* All numeric columns must be totalled automatically.

**<Search Button>**

{When the search button is pressed the grid table values will be erased and empty columns will be available for the user to key in any search parameters in any columns and when the search button is pressed again the values matching the search parameters will be display in the grid.}

**Menu options on this common report output form :-**

**Main Menu item : Column Maintenance**

**Sub-item : New Custom Columns**

{Using this option the user can add custom columns into the report output grid}

**<Add Column> <Edit Column> <Delete Column>**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Column Name** | **Description** | **Type** | **Size** | **Decimal Size** | **Manual Editing?** | **Mass Update?** |
|  |  | Character/Combobox List/Numeric/Date/Computed Column |  |  | Yes/No | Yes/No |

\*\* Computed columns are always display only field.

\*\* Manual editing is to control whether the column can be edited manually on the report output itself.

\*\* Mass update is to control whether the column can be mass updated using the mass update function illustrated below.

**Maintain Combobox List Values**

{Using this option the user can maintain the list values for the combobox columns}

**<Add Value> <Edit Value> <Delete Value>**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Value Code** | **Value Description** |
|  |  |  |
|  |  |  |

**Maintain Computed Column Formula**

{Using this option the user can maintain the formula with mathematical operations for computed columns}

**<Add Formula> <Edit Formula> <Delete Formula>**

|  |  |
| --- | --- |
| **Column Name** | **Formula** |
|  | Columnname1  +  Columnname2) |
|  |  |

**Sub item : Mass Update Custom Column**

{Using this option the user can mass update the custom columns where the manual update is allowed. Even if the manual editing is set to be not allowed and mass update is set to be allowed the mass update should be allowed in this case. If the custom column is a combobox you need to display the combobox to select the values}

|  |  |
| --- | --- |
| **Custom Column Name** | **Custom Value** |
| <Select Columns> |  |
|  |  |

**<Mass Update>**

**Sub item : Auto-Update Custom Columns**

{Provide a form to setup derivation rules to auto-update custom column values based on some rules. All custom columns can be auto updated using this function regardless of whether the manual editing is allowed or not. If an existing column matches a value, then the source column value or a source constant value will be assigned to the target custom column.}

**<Add New Rule> <Save> <Cancel>**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Existing Column** | **Match** | **Match Value** | **Source Column** | **Source**  **Constant** | **Target Custom Column** | **Actions** |
| <Combobox> | =  <>  >  <  >=  <=  In | <Editbox> | <Combobox> | <Editbox> | <Combobox> | **<Edit Rule>**  **<Copy Rule>**  **<Delete Rule>**  **<Run Rule>** |
| <Combobox> | =  <>  >  <  >=  <=  In | <Editbox> | <Combobox> | <Editbox> | <Combobox> | **<Edit Rule>**  **<Copy Rule>**  **<Delete Rule>**  **<Run Rule>** |

**Sub item : Reorder Columns**

{Using this option the user can reorder the columns}

<Select column> <Move After/Move Before/Move To First/Move To Last> <Select Column>

**<Save>**

**Sub item : Sort Columns**

{Using this option the user can sort the columns}

<Listbox with all columns> <Add> <Listbox with added values>

<Remove>

<Add All>

<Remove All>

**<Save>**

**Sub item : Hide/Unhide Columns**

{Using this option the user can hide and unhide columns}

<Listbox with all columns> <Add> <Listbox with added values>

<Remove>

<Add All>

<Remove All>

**<Save>**

{Using this option, the user can subtotal numeric columns by other columns}

<Listbox with all columns> <Add> <Listbox with added values>

<Remove>

<Add All>

<Remove All>

**<Save>**

**Main Menu item : Layouts**

**Sub item : Save Current layout**

Provide a form to save the layout of the above grid :-

Layout Name : <Enter layout name>

Description : <>

**<Save Current Layout>**

**Sub Item : Layout Maintenance**

Provide a form to maintain the layouts :-

**<Choose Selected> <Edit Selected> <Set Default> <Delete Selected>**

|  |  |
| --- | --- |
| **Layout Name** | **Description** |
|  |  |
|  |  |

**Main Menu item : Access Rights**

Access rights can be given at the respective user id, reports and layouts levels :-

**<Select User Id> <Select Reports> <Select Layouts>**

|  |  |
| --- | --- |
| **Functions Access** | **Yes/No** |
| Custom Columns Maintenance |  |
| Mass Update Custom Columns |  |
| Auto-Update Custom Columns |  |

**<Save>**

Access rights to be given to the selected specific report :-

**<Select User Id>**

|  |  |
| --- | --- |
| **Reports Access** | **Yes/No** |
| Report 1 |  |
| Report 2 |  |
| Report 3 |  |

**<Save>**

Access rights to be given to the selected specific layout of the particular report :-

**<Select User Id> <Select Reports>**

|  |  |
| --- | --- |
| **Layouts Access** | **Yes/No** |
| Layout 1 |  |
| Layout 2 |  |
| Layout 3 |  |

**<Save>**

**Required Table Design**

|  |  |
| --- | --- |
| **Table Name : Reports\_Custom\_Columns** | |
| **Table to store the custom column details** | |
| **Field Name** | **Remarks** |
| Report\_Name | This report name is passed as an input parameters to the common form |
| Layout\_Name | This is the layout name given by the users. First use a unique random layout name to store the records and once the user saves the layout, replace this random name with the proper layout name decided by the user. |
| Custom\_Column |  |
| Description |  |
| Type |  |
| Size |  |
| Decimal Size |  |
| Formula |  |

A separate table needs to be created based on the input parameter report name so that the user entered values in the custom columns can be saved into this new table. The primary keys of the report name table should also be the primary keys for this new table as well.

For example assume that the following report name Customer Sales is passed to the common form with the random unique table name : msjuiia

(All reports will use only random unique table names instead of cursors)

|  |  |  |
| --- | --- | --- |
| **Table Name : msjuiia** | | |
| **Customer Id** | **Invoice Number** | **Invoice Amount** |
| Primary Key 1 | Primary Key 2 | Amount field |
|  |  |  |
|  |  |  |

And 3 custom columns have been added by the user as follows :-

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Column Name** | **Description** | **Type** | **Size** | **Decimal Size** | **Formula** |
| Invoice description | To store the description about the invoice | Character | Unlimited |  |  |
| Payment Date | To store the payment date | Date |  |  |  |
| Payment Amount | To store the payment amount | Numeric | 10 | 2 |  |

The following new table report name combined with a random unique layout name have to be created to store the values of the custom columns. The primary keys from the main report name table will also be used as the primary keys for this table. I have used a sample random unique layout name hsjwusk to demonstrate this :-

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Table Name : Customer\_Sales\_hsjwusk\_Custom\_Columns** | | | | | |
| **Customer Id**  **(Primary Key)** | **Invoice Number**  **(Primary Key)** | **Invoice Amount** | **Invoice Description**  **(Char)** | **Payment Date**  **(Date)** | **Payment Amount**  **(Numeric)** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Finally, if the user decides to save the layout, the given layout name will be used to replace the above random layout name in the table name. If the user decides not to save the layout, then the above table is discarded.

|  |  |
| --- | --- |
| **Table Name : Reports\_Custom\_Column\_Rules** | |
| **Table to store the custom column rules for auto updating** | |
| **Field Name** | **Remarks** |
| Report\_Name | This report name is passed as an input parameters to the common form |
| Layout\_Name | This is the layout name given by the users. First use a unique random layout name to store the records and once the user saves the layout, replace this random name with the proper layout name decided by the user. |
| Column\_Name |  |
| Match |  |
| Match\_value |  |
| Source\_Column |  |
| Source\_Constant |  |
| Custom\_Column |  |

|  |  |
| --- | --- |
| **Table Name : Reports\_Columns\_Properties** | |
| **Table to store the columns Position, Sort order and Visibility** | |
| **Field Name** | **Remarks** |
| Report\_Name | This report name is passed as an input parameters to the common form |
| Layout\_Name | This is the layout name given by the users. First use a unique random layout name to store the records and once the user saves the layout, replace this random name with the proper layout name decided by the user. |
| Column\_Name |  |
| Columns\_Position |  |
| Sort\_Order |  |
| Visibility |  |

|  |  |
| --- | --- |
| **Table Name : Reports\_Layouts** | |
| **Table to store the Layouts** | |
| **Field Name** | **Remarks** |
| Report\_Name | This report name is passed as an input parameters to the common form |
| Layout\_Name |  |
| Description |  |

**APPENDIX A**

**Sample PHP report for Customer Sales**

<?php

// configure mysql database connection to `program1`

$con = mysqli\_connect('localhost', 'root', '', 'program1');

// handle form submission

if(isset($\_POST['from']) && isset($\_POST['to'])) {

$from = intval($\_POST['from']);

$to = intval($\_POST['to']);

// validate $from and $to

if($from > $to)

echo "<p style='color: red'><b>[from]</b> must be less than or equal to <b>[to]</b></p>";

else {

// delete all records from `report\_output` (comment out if not needed)

$query = "DELETE FROM `report\_output` WHERE 1";

mysqli\_query($con, $query);

// === MAIN SOLUTION =======================================================================================

// fetch data from `customer\_sales` from $from and to $to

$query = "SELECT \* FROM `customer\_sales` WHERE `Customer Code` BETWEEN $from AND $to ORDER BY `Customer Code`";

$result = mysqli\_query($con, $query);

// insert new processed data to `report\_output`

while($row = mysqli\_fetch\_assoc($result)) {

// multiple amount field by 2

$row['Amount'] \*= 2;

$query = "INSERT INTO `report\_output` ";

$query .= "VALUES(";

$query .= $row['Customer Code'] . ", ";

$query .= "'" . mysqli\_real\_escape\_string($con, $row['Customer Name']) . "', ";

$query .= "'" . mysqli\_real\_escape\_string($con, $row['Invoice Number']) . "', ";

$query .= $row['Amount'];

$query .= ")";

mysqli\_query($con, $query);

}

// =========================================================================================================

}

}

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Program 1</title>

<style>

body {

font-family: monospace;

}

th, td, input, button {

padding: 5px;

}

input {

width: 60px;

}

nav a {

text-decoration: none;

}

nav a.active {

font-weight: bold;

}

</style>

</head>

<body>

<!-- Little Navigation -->

<nav>

<a href="index.php" class="active">First Program</a> |

<a href="../program2/index.php">Second Program</a>

</nav>

<hr/>

<h1>First Program</h1>

<!-- Display Input Table -->

<h2>Input Table</h2>

<table border="1" cellspacing="0">

<thead>

<tr>

<th>Customer Code</th>

<th>Customer Name</th>

<th>Invoice Number</th>

<th>Amount</th>

</tr>

</thead>

<tbody>

<?php

$query = "SELECT \* FROM `customer\_sales` ORDER BY `Customer Code`";

$result = mysqli\_query($con, $query);

while($row = mysqli\_fetch\_assoc($result)) {

echo "<tr>";

echo "<td>" . $row['Customer Code'] . "</td>";

echo "<td>" . $row['Customer Name'] . "</td>";

echo "<td>" . $row['Invoice Number'] . "</td>";

echo "<td>" . $row['Amount'] . "</td>";

echo "</tr>";

}

?>

</tbody>

</table>

<br/><br/>

<form method="POST" action="<?= $\_SERVER['PHP\_SELF'] ?>">

<b>Customer Code:</b>

from

<input type="number" name="from" required>

to

<input type="number" name="to" required>

<button type="submit">EXECUTE</button>

</form>

<br/><hr/>

</body>

</html>

**Sample PHP report for Material Listing**

<?php

// configure mysql database connection to `program2`

$con = mysqli\_connect('localhost', 'root', '', 'program2');

// handle form submission

if(isset($\_POST['from']) && isset($\_POST['to'])) {

$from = intval($\_POST['from']);

$to = intval($\_POST['to']);

// validate $from and $to

if($from > $to)

echo "<p style='color: red'><b>[from]</b> must be less than or equal to <b>[to]</b></p>";

else {

// delete all records from `report\_output` (comment out if not needed)

$query = "DELETE FROM `report\_output` WHERE 1";

mysqli\_query($con, $query);

// === MAIN SOLUTION =======================================================================================

// fetch data from `material\_listings` from $from and to $to

$query = "SELECT \* FROM `material\_listings` WHERE `Material Code` BETWEEN $from AND $to ORDER BY `Material Code`";

$result = mysqli\_query($con, $query);

// insert new processed data to `report\_output`

while($row = mysqli\_fetch\_assoc($result)) {

// multiple amount field by 2

$row['Quantity'] \*= 2;

$query = "INSERT INTO `report\_output` ";

$query .= "VALUES(";

$query .= $row['Material Code'] . ", ";

$query .= "'" . mysqli\_real\_escape\_string($con, $row['Material Name']) . "', ";

$query .= $row['Quantity'];

$query .= ")";

mysqli\_query($con, $query);

}

// =========================================================================================================

}

}

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Program 2</title>

<style>

body {

font-family: monospace;

}

th, td, input, button {

padding: 5px;

}

input {

width: 60px;

}

nav a {

text-decoration: none;

}

nav a.active {

font-weight: bold;

}

</style>

</head>

<body>

<!-- Little Navigation -->

<nav>

<a href="../program1/index.php">First Program</a> |

<a href="index.php" class="active">Second Program</a>

</nav>

<hr/>

<h1>Second Program</h1>

<!-- Display Input Table -->

<h2>Input Table</h2>

<table border="1" cellspacing="0">

<thead>

<tr>

<th>Material Code</th>

<th>Material Name</th>

<th>Quantity</th>

</tr>

</thead>

<tbody>

<?php

$query = "SELECT \* FROM `material\_listings` ORDER BY `Material Code`";

$result = mysqli\_query($con, $query);

while($row = mysqli\_fetch\_assoc($result)) {

echo "<tr>";

echo "<td>" . $row['Material Code'] . "</td>";

echo "<td>" . $row['Material Name'] . "</td>";

echo "<td>" . $row['Quantity'] . "</td>";

echo "</tr>";

}

?>

</tbody>

</table>

<br/><br/>

<form method="POST" action="<?= $\_SERVER['PHP\_SELF'] ?>">

<b>Material Code:</b>

from

<input type="number" name="from" required>

to

<input type="number" name="to" required>

<button type="submit">EXECUTE</button>

</form>

<br/><hr/>

</body>

</html>