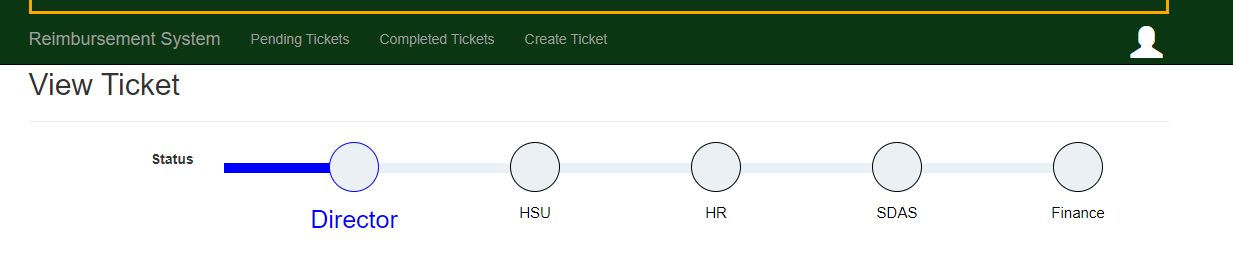
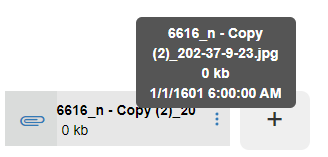
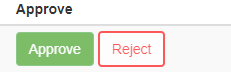
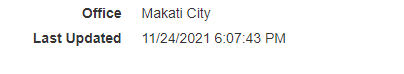
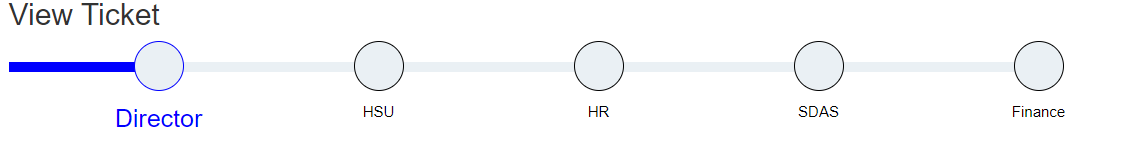
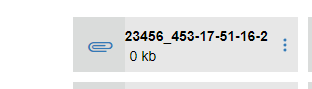
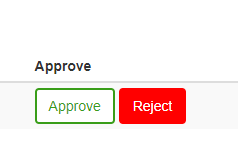
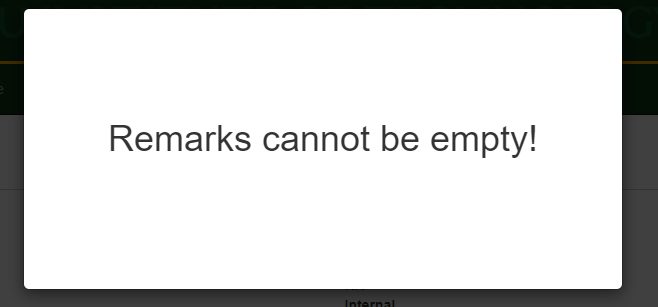
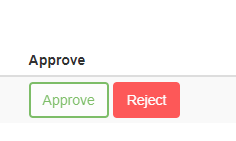
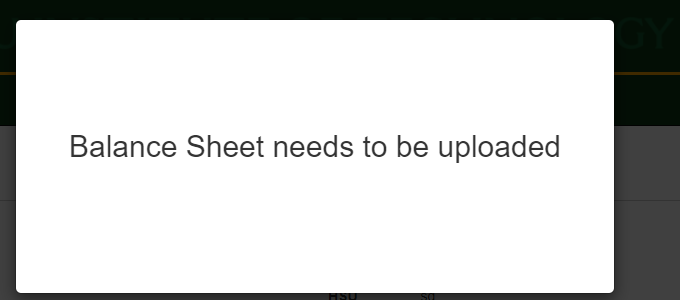
# Changes files

* Controllers/AdminController.cs
* Controllers/UserController.cs
* Models/Ticket.cs
* Views/Admin/CompletedTickets.cshtml
* Views/Admin/Home.cshtml
* Views/Admin/PendingTickets.cshtml
* Views/Admin/ViewCompletedTicket.cshtml
* Views/Admin/ViewTicket.cshtml
* Views/User/CreateTicket.cshtml
* Views/User/PendingTickets.cshtml
* Views/User/ViewCompletedTicket.cshtml
* Web.config

# Changed Functionalities for Users

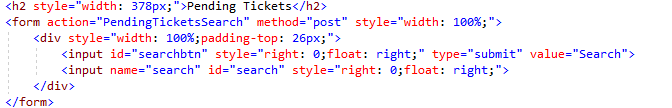
* Added Search Textbox with Search Button on PendingTickets and Completed Tickets Page
* Added Status Progress.
* Added Uploaded file design. And on clip button clip it will redirect to that uploaded file. And Plus Button. 
* Changed Approved and Rejected Icon In View Completed Ticket Page. 
* After Update any Ticket or after Create any Ticket there has auto update date.

# Changed Functionalities for Admin

* On admin home page there has shown two Graphs.
* On the Pending Tickets and Completed Tickets Page has the same Search Box with Search Button
* Has Last Updated Date on View Ticket Page. 
* Admin can see Progress of Ticket with approved or Rejected. 
* Has same uploaded file Design on View Ticket page. 
* Reject and Approved Button design Changed. 
* Added Remarks Empty Warning. 
* Added Save Confirmation. 
* For HR role Approve and Reject Button has disabled. 
* For HR role and HSU role there has file upload system. 
* For HR role if no file is uploaded then it will says Balance sheets need to be uploaded. 

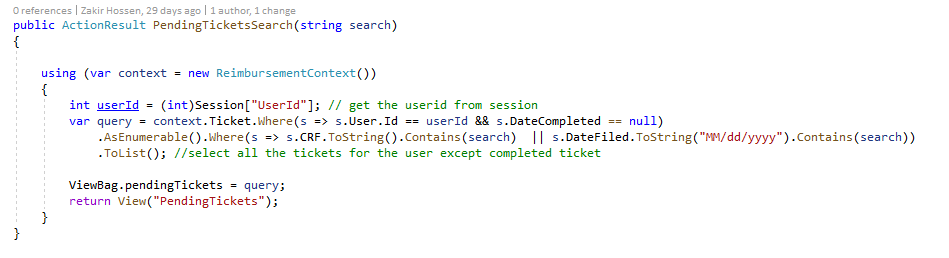
# Search Textbox with Search Button

For the search I have used this code in those pages.



Here I have used Form that is redirecting to PendingTickeSearch page as post.

And I have put these codes in the usercontroller.cs



When it goes to that url then this will work. It will first select the userid and then search. It is search where userid is same as current and is not completed. And gave the contains with text or date. All the functions same as this in controllers are:

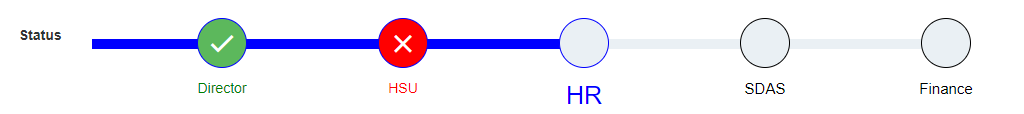
* public ActionResult CompletedTicketsSearch(string search) ---User
* public ActionResult PendingTicketsSearch(string search) ---User

All the cshtml :

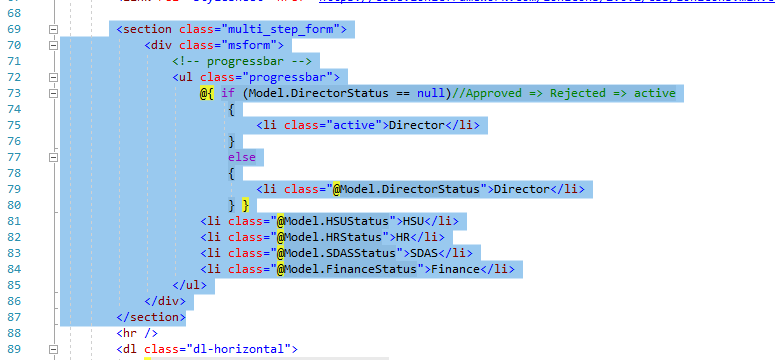
<form action="CompletedTicketsSearch" method="post" style="width: 100%;">

<form action="PendingTicketsSearch" method="post" style="width: 100%;">

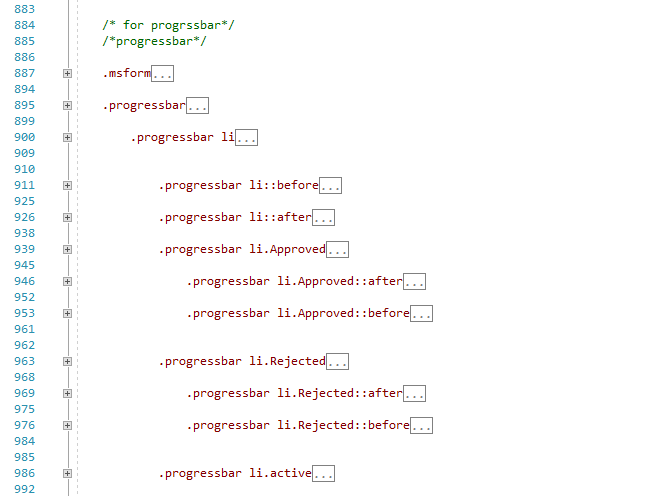
# Status Progress



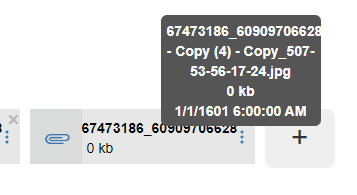
This is on all ViewTicket.cshtml page. For this need to use lots of css.



Here for the multi\_step\_form, msform, progressbar, active, Rejected, Approved there has css . It is getting the status from the database. For css down of the codes.



# Uploaded file design





Here it is using display:table; in the cshtml code.

@\*This is the main container\*@

<div style="display:table;">

<div style="width: 104%;margin-left: -20px;">@\*This is the second container\*@

@\*This is using External css\*@

<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Roboto:300i,400,400i,500,700,900">

<link rel="stylesheet" href="https://code.ionicframework.com/ionicons/2.0.1/css/ionicons.min.css">

@\*Here we are getting Medias from the selected Ticket using For Loop method.\*@

@for (int i = 0; i < Model.Medias.Count(); i++)//this is inside a media

{

<div class="image-container" style="margin: 5px; float: left; position: relative">

@\*These are for updating. Just for hiding information.\*@

@Html.HiddenFor(model => Model.Medias[i].Id, new { @class = "media-id" })

@Html.HiddenFor(model => Model.Medias[i].ImagePath)

@Html.HiddenFor(model => Model.Medias[i].TicketCRF)

<b class="images">

<div style="display: flex;">

@\*ClipPaper Code\*@

<a href="@Model.Medias[i].ImagePath" style="background: #dadcdb;display: -webkit-box;width: 52px;cursor: pointer;">

<div class="ion-paperclip" style="height: 30px;width: 30px;font-size: 32px;transform: rotate(90deg);margin: 25px -10px 0px 18px;"> </div>

</a>

<div style="display: flex;background: #e7e7e7;">

<div style="padding: 9px 0px 0px 0px; width: 152px;">

@\*Getting filename, size and uploaded date\*@

@{ string filename = ""; if (Path.GetFileName(Model.Medias[i].ImagePath).Length > 20) { filename = Path.GetFileName(Model.Medias[i].ImagePath).Substring(0, 20); } else { filename = Path.GetFileName(Model.Medias[i].ImagePath); };

string uploadDir = "Ticket\_Images"; //server folder name

string filepath = Path.Combine(Server.MapPath("~/" + uploadDir), Path.GetFileName(Model.Medias[i].ImagePath)); FileInfo fi = new FileInfo(filepath);

long fileSizeInBytes = 0;

try

{

fileSizeInBytes = (fi.Length / 1024);

}

catch (Exception e) { }; }

<k style="align-content: end;color: black;">@filename</k>

<br>

<k style="margin: -19px -2px 5px 5px;color: black;font-weight: normal;">@fileSizeInBytes kb</k>

</div>

@\*Here start for Popup\*@

<div style="padding: 20px 7px 13px 0px; width: 21px;">

<a onclick="myFunction(this.id)" id="@i" class="popup" style="cursor: pointer;">

<span class="popuptext" id="myPopup">

<span>@Path.GetFileName(Model.Medias[i].ImagePath) </span> @\*getting the filefullname\*@

<br>

<span> @fileSizeInBytes kb</span> @\*Getting the file Size\*@

<br />

<span> @fi.CreationTime</span> @\*Getting the Created File time\*@

</span>

<svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-three-dots-vertical" viewBox="0 0 16 16">

<path d="M9.5 13a1.5 1.5 0 1 1-3 0 1.5 1.5 0 0 1 3 0zm0-5a1.5 1.5 0 1 1-3 0 1.5 1.5 0 0 1 3 0zm0-5a1.5 1.5 0 1 1-3 0 1.5 1.5 0 0 1 3 0z"></path>

</svg>

</a>

</div>

@\*Here End for Popup\*@

</div>

</div>

</b>

</div>

} @\*Here loop again\*@

@{

if (role.Equals(Reimbursement\_Web\_System.Models.Role.HR) || role.Equals(Reimbursement\_Web\_System.Models.Role.HSU))

{

//Here for Plus Icon

<div id="beforetheadd" style="float: left;width: 70px;height: 59px;margin: 5px;">

<input type="button" value="+" style="width: 100%; height: 100%; display: block; border: none; border-radius: 9px; font-size: 30px; " onclick="uploadfile();" />

</div> }

}

<br />

<br />

</div>

</div>

<svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" fill="currentColor" class="bi bi-three-dots-vertical" viewBox="0 0 16 16">

<path d="M9.5 13a1.5 1.5 0 1 1-3 0 1.5 1.5 0 0 1 3 0zm0-5a1.5 1.5 0 1 1-3 0 1.5 1.5 0 0 1 3 0zm0-5a1.5 1.5 0 1 1-3 0 1.5 1.5 0 0 1 3 0z"></path>

</svg>

</a>

</div>

@\*Here End for Popup\*@

</div>

</div>

</b>

</div>

} @\*Here loop again\*@

@{

if (role.Equals(Reimbursement\_Web\_System.Models.Role.HR) || role.Equals(Reimbursement\_Web\_System.Models.Role.HSU))

{

//Here for Plus Icon

<div id="beforetheadd" style="float: left;width: 70px;height: 59px;margin: 5px;">

<input type="button" value="+" style="width: 100%; height: 100%; display: block; border: none; border-radius: 9px; font-size: 30px; " onclick="uploadfile();" />

</div> }

}

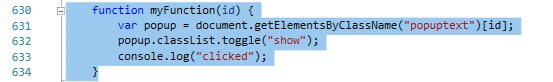
<br />

<br />

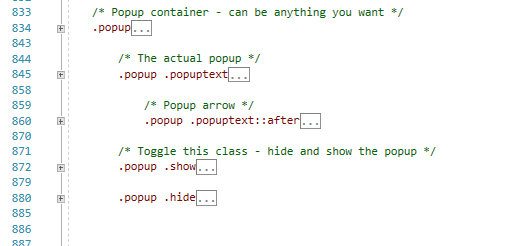
</div>

</div>

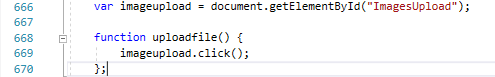
It also required Javascript in order to open the popup.



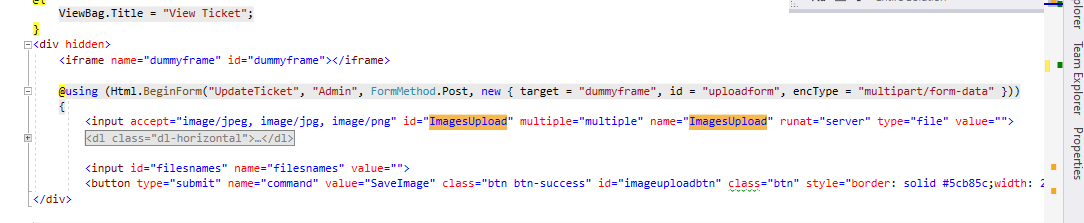
This are the css codes.



For the add button it is clicking uploadfile();



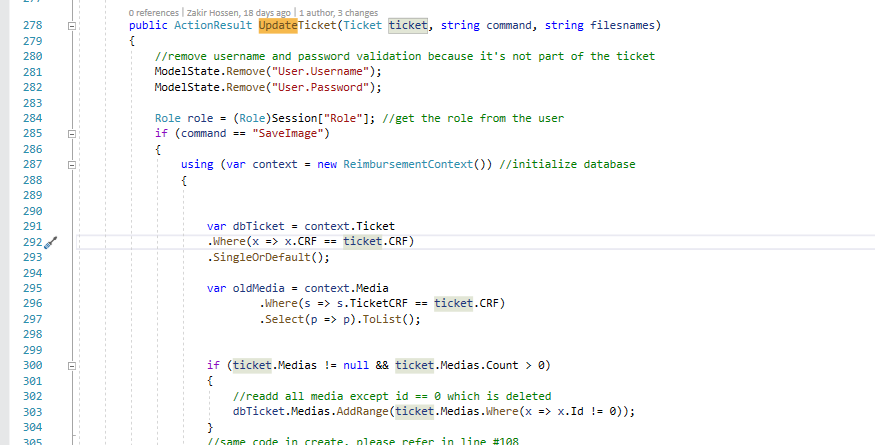
This is javascript code. And here ImagesUpload Is a choosefile. It is hidden.



Here I have used an iframe. Using inside it used Form. It goes to UpdateTicket. For users it goes to user controller.cshtml and for admin it goes to admin controller.cshtml . Here is the codes:

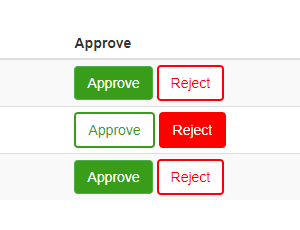
<button type="submit" name="command" value="SaveImage"

Here command is SaveImage. In the controller.

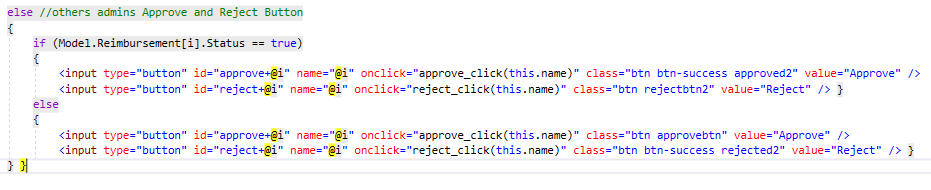


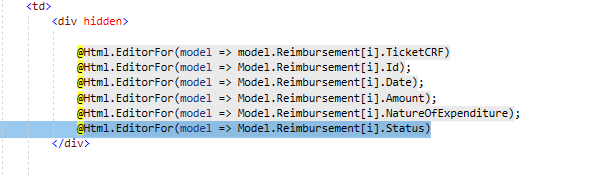
Checking the command if it is SaveImage or others. Then it store that image. And these are happening in that iframe.

# Reject and Approved Button design



These are added in the Reimbursement table. The codes are blew:



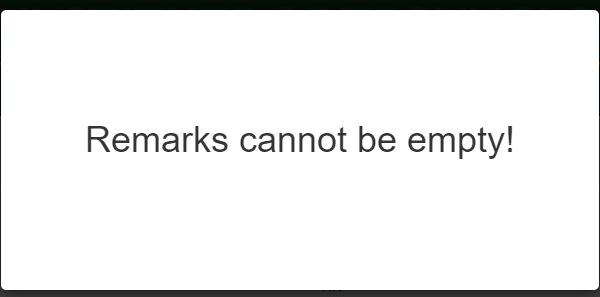
Here if that Reimbursement status is Approve then it will be the above one. And if not it will take the else part code. And here @i mean serial of Reimbursement. Here when clicking on any of those button then it is using the function with that button name which is basically id number. This is the main hidden checkbox. Using it because we will need it for saving/updating.

The jabascript codes are:



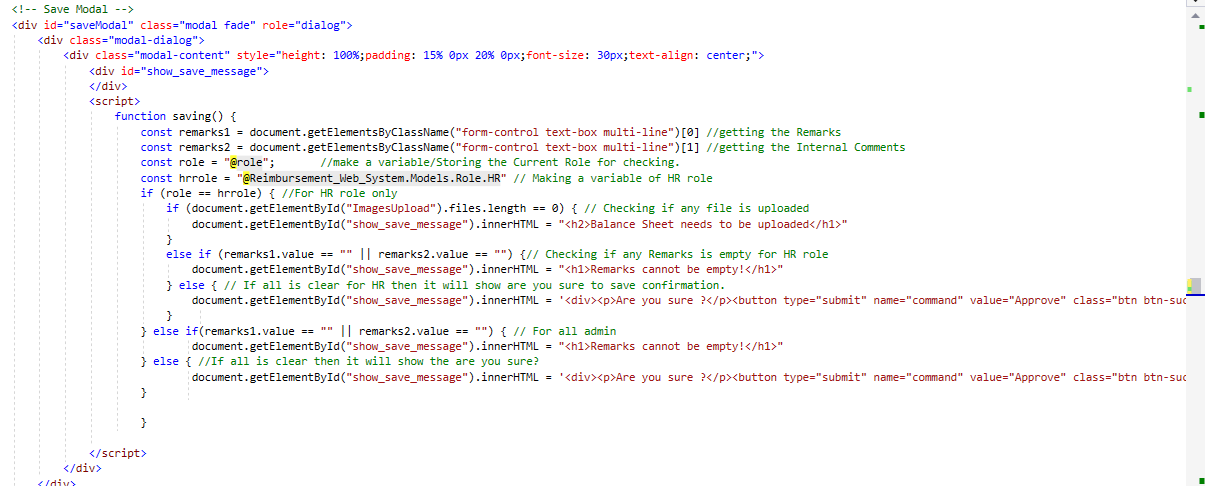
After click there here it is changing that status checkedbox with the id.

# Remarks Empty Warning

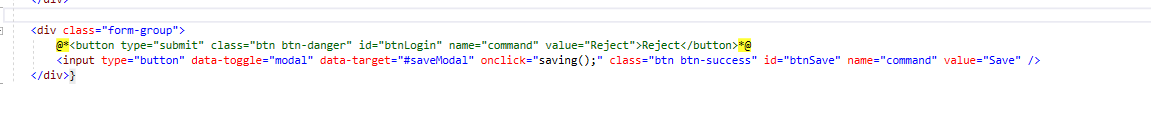


Here

If any of these are empty and then click on Save(Admin) then it will show this popup.



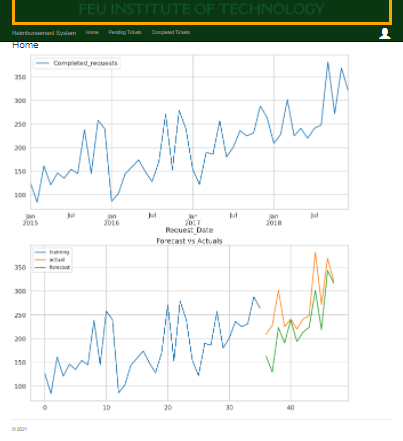
These are the codes for Save Model/Popup. And here is for the Save button code.



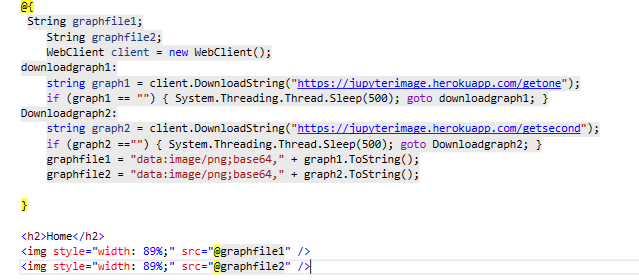
When all is clear then it shows.



# Admin Home Graph



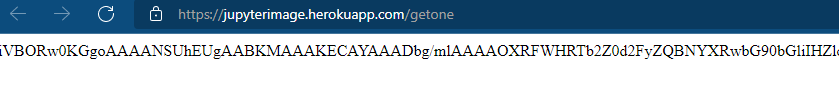
Here these two are Picture. But the picture is in Base64 decoded. And the code is :

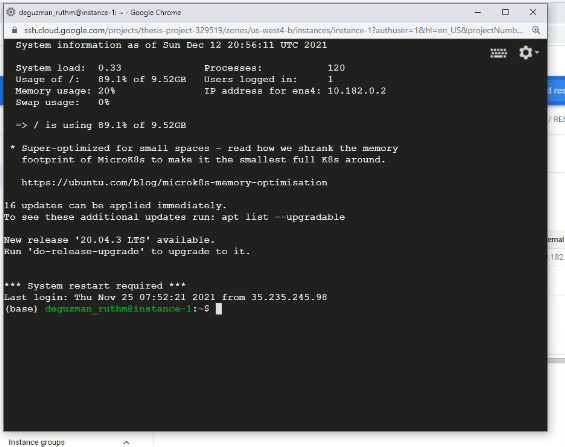


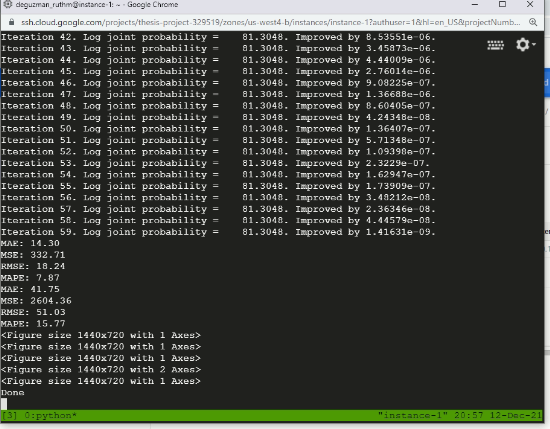
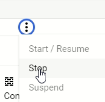
Just these. And the output code in browser is.



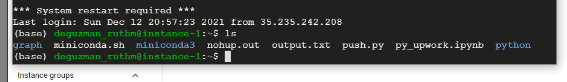
The method to get these are. It is getting the value for number one graph here. <https://jupyterimage.herokuapp.com/getone> From this link.



And this are coming from Google cloud. And in the code has nothing to update. For the updating Jupyter Python code. We need to login into google cloud. After logged in and gone to VM instances. There has an instance. Which is basically a virtual Ubuntu pc. To update the python first go click on SSH. and it will show a popup of Ubuntu Terminal. 

And if we write there tmux attacth and hit enter. Then it will show a process. It is basically running the Jupyter Python code . Even if we close the window it will keep working because fo tmux. And if you want to update the python code then you will need to stop it then click on 

After click it all will be stopped. And then you can continue with files setuping or restarting.



Here if we write ls. Then it will show all the files are here. If you want to update that ipynb file. Then delete that and then use the <https://Transfer.sh> to upload.

For the download you can also use the same.

After uploaded for the running the code.

Type “tmux attach” -----------This is must.

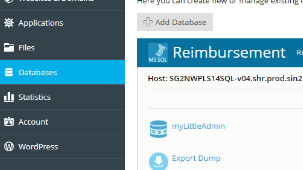
After that you can type “ipython py\_upwork.ipynb” . And then you can close that window.

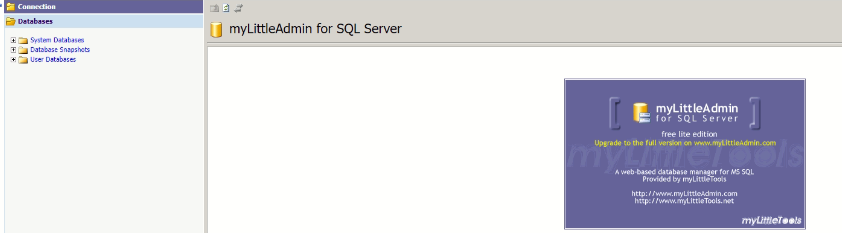
# For the uploading files on Plesk

After login into plesk. Then go to file manager. And then inside the httpdocs. Upload that solution files as zip. And then delete all the files except that zip file. And after deleted you can extract that zip inn that folder.

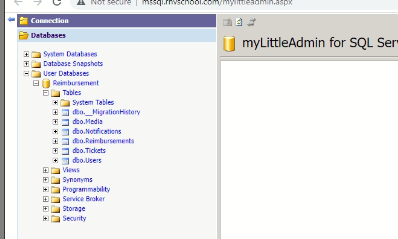
# For the Database changing

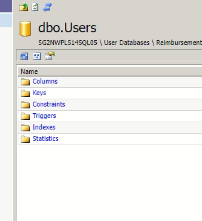
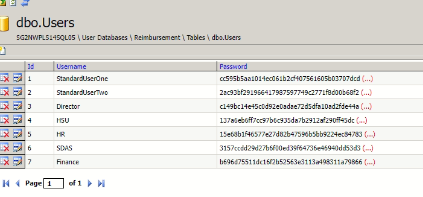
You can’t use your Local Database server anymore. But to create any table/any column or update name then follow here on Plesk:

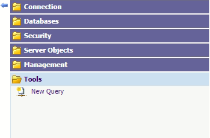


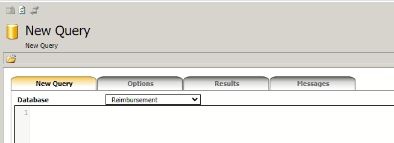
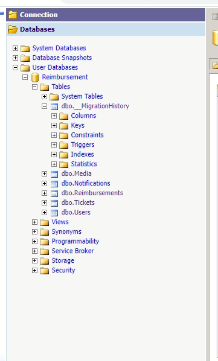
Here click on myLittleAdmin. And it will redirect to this. 

And here is the database:



To view any table data. Then click on that any table. And then here click on that Open Table button. Here you can see the table datas.

To add any column/add any new table or update anything from database. Then You will need to use Query 

After click on New Query. You will see this. Here you will have to use sql query. You can search on google. For example. [SQL - CREATE Table (tutorialspoint.com)](https://www.tutorialspoint.com/sql/sql-create-table.htm) you can go to this url. And copy the query from there for creating table. And then put in there with your information. And after execute you will have to refresh your browser page. And then you will see changes on here or where you wanted.