Dashboard.component.html

```
<div class="h-100" [@routerTransition]>
     <div class="row content-center h-20" *ngIf="detailsDashboard" >
              <div class="col-xl-12 pl-xl-12 pr-xl-12 col-lg-12 col-md-12 pl-md-12 pr-md-12 col-sm-12 h-45 " >
                <table class="tr">
                <tr class="tr">
          <td><a (click)="commonFunction.open(request);getRequest();" value="download" style="text-decoration: underline;">Request Status</a></td>
                    <td> Claim Status:<span class="blinking"><b>{{status}}</b></span></td>
                    <td><a (click)="commonFunction.open(content);getData();" value="download" style="text-decoration: underline;">Payment Due</a></td>
                </tr>
            </table>
    </div>
     </div>
     <br><br>
    <div class="row content-center h-80" *ngIf="detailsDashboard">
        <div class="col-xl-2 pl-xl-4 pr-xl-4 col-lg-2 col-md-3 pl-md-4 pr-md-4 col-sm-3 h-45 dash_shadow" >
            <app-stat class="h-100" [topBgClass]="'top_name '" [bgClass]="' name '" [bgCount]="'editn_count '" [icon]="'fa-tasks'" [count]="this.personal" [label]="'Personal Details'" [routerLink]="['/personal']" [routerLinkActive]="['router-link-active']" [queryParams]="{inprogress:0}" skipLocationChange></app-stat>
        </div>
        <div class="col-xl-2 pl-xl-4 pr-xl-4 col-lg-2 col-md-3 pl-md-4 pr-md-4 col-sm-3 h-45 dash_shadow">
    <app-stat class="h-100" [topBgClass]="'top_payment'" [bgClass]="'payment'" [bgCount]="'payment_count'" [icon]="'fa-shopping-cart'" [count]="this.payment" [label]="'Payment'" [routerLink]="['/payment']" [routerLinkActive]="['router-link-active']" [queryParams]="{inprogress:1}" skipLocationChange></app-stat>
        </div>
        <div class="col-xl-2 pl-xl-4 pr-xl-4 col-lg-2 col-md-3 pl-md-4 pr-md-4 col-sm-3 h-45 dash_shadow" >
```

```
 <app-stat class="h-100" [topBgClass]="'top_policy'" [bgClass]="'policy'" [bgCount]="'policy_count'" [icon]="'fa-support'" [count]="this.policy" [label]="'Policy Document'" [routerLink]="['/download']" [routerLinkActive]="['router-link-active']" [queryParams]="{inprogress:2}" skipLocationChange ></app-stat>
        </div>
 <div class="col-xl-2 pl-xl-4 pr-xl-4 col-lg-2 col-md-3 pl-md-4 pr-md-4 col-sm-3 h-45 dash_shadow" >
 <app-stat class="h-100" [topBgClass]="'top_personal_details '" [bgClass]="'personal_details '" [bgCount]="'personal_details_count '" [icon]="'fa-support'" [count]="this.req" [label]="'Request'" [routerLink]="['/request']" [routerLinkActive]="['router-link-active']" [queryParams]="{inprogress:2}" skipLocationChange ></app-stat>
        </div>
          <div class="col-xl-2 pl-xl-4 pr-xl-4 col-lg-2 col-md-3 pl-md-4 pr-md-4 col-sm-3 h-45 dash_shadow" >
           <app-stat class="h-100" [topBgClass]="'top_request'" [bgClass]="'request'" [bgCount]="'request_count'" [icon]="'fa-support'" [label]="'Edit Payment Frequency'" [routerLink]="['/editfreq']" [routerLinkActive]="['router-link-active']" [queryParams]="{inprogress:2}" skipLocationChange ></app-stat>
        </div>
  <div class="col-xl-2 pl-xl-4 pr-xl-4 col-lg-2 col-md-3 pl-md-4 pr-md-4 col-sm-3 h-45 dash_shadow" >
<app-stat class="h-100" [topBgClass]="'top_frequency'" [bgClass]="'frequency'" [bgCount]="'editf_count'" [icon]="'fa-support'" [label]="'Edit Name'" [routerLink]="['/editname']" [routerLinkActive]="['router-link-active']" [queryParams]="{inprogress:2}" skipLocationChange ></app-stat>
 </div>
    <div class="col-xl-2 pl-xl-4 pr-xl-4 col-lg-2 col-md-3 pl-md-4 pr-md-4 col-sm-3 h-45 dash_shadow" >
 <app-stat class="h-100" [topBgClass]="'top_cs '" [bgClass]="' cs '" [bgCount]="'cs_count '" [icon]="'fa-tasks'" [label]="'Claim Submitions'" [routerLink]="['/claimsubmition']" [routerLinkActive]="['router-link-active']" [queryParams]="{inprogress:0}" skipLocationChange></app-stat>
        </div>
            <div class="col-xl-2 pl-xl-4 pr-xl-4 col-lg-2 col-md-3 pl-md-4 pr-md-4 col-sm-3 h-45 dash_shadow" >
           <app-stat class="h-100" [topBgClass]="'top_rh '" [bgClass]="' rh '" [bgCount]="'rh_count '" [icon]="'fa-tasks'" [label]="'Request History'" [routerLink]="['/reqhist']" [routerLinkActive]="['router-link-active']" [queryParams]="{inprogress:0}" skipLocationChange></app-stat>
        </div>
```

```
      </div>
        <div class="row content-center h-20" *ngIf="nomineeDashboard" >
       <div class="col-xl-12 pl-xl-12 pr-xl-12 col-lg-12 col-md-12 pl-md-12 pr-md-12 col-sm-12 h-45 " >
              <table class="tr">
            <tr class="tr">
                <td>{{space}}</td>
                <td>{{space}}</td>
                <td> Claim Status:<span class="blinking"><b>{{status}}</b></span></td>
            </tr>
          </table>
    </div>
     </div>
      <div class="row content-center h-80" *ngIf="nomineeDashboard">
      <div class="col-xl-2 pl-xl-4 pr-xl-4 col-lg-2 col-md-3 pl-md-4 pr-md-4 col-sm-3 h-45 dash_shadow" >
 <app-stat class="h-100" [topBgClass]="'top_cs '" [bgClass]="' cs '" [bgCount]="'cs_count '" [icon]="'fa-tasks'"
[label]="'Claim Submitions'" [routerLink]="['/claimsubmition']" [routerLinkActive]="['router-link-active']"
[queryParams]="{inprogress:0}" skipLocationChange></app-stat>
     </div>

    <!--  #F3E2A9 -->
      </div>
        <div class="row content-center h-100" *ngIf="CustomerDashboard">
            <div class="col-xl-2 pl-xl-4 pr-xl-4 col-lg-2 col-md-3 pl-md-4 pr-md-4 col-sm-3 h-45 dash_shadow" >
          <app-stat class="h-100" [topBgClass]="'top_name '" [bgClass]="' name '" [bgCount]="'editn_count '"
[icon]="'fa-tasks'" [label]="'Customer Search'" [routerLink]="['/Customer']" [routerLinkActive]="['router-link-
active']" [queryParams]="{inprogress:0}" skipLocationChange></app-stat>
          </div>
       <div class="col-xl-2 pl-xl-4 pr-xl-4 col-lg-2 col-md-3 pl-md-4 pr-md-4 col-sm-3 h-45 dash_shadow">
          <app-stat class="h-100" [topBgClass]="'top_c'" [bgClass]="'payment'" [bgCount]="'payment_count'"
[icon]="'fa-shopping-cart'" [count]="this.claim" [label]="'Claims'" [routerLink]="['/summary']"
[routerLinkActive]="['router-link-active']" [queryParams]="{inprogress:1}" skipLocationChange></app-stat>
```

```
        </div>
    </div>
     <div class="row content-center h-100" *ngIf="ClaimDashboard">
        <div class="col-xl-2 pl-xl-4 pr-xl-4 col-lg-2 col-md-3 pl-md-4 pr-md-4 col-sm-3 h-45 dash_shadow" >
            <app-stat class="h-100" [topBgClass]="'top_cm'" [bgClass]="'policy'" [bgCount]="'policy_count'" [icon]="'fa-tasks'" [count]="this.claim" [label]="'Claim Management'" [routerLink]="['/summary']" [routerLinkActive]="['router-link-active']" [queryParams]="{inprogress:0}" skipLocationChange></app-stat>
        </div>
        <div class="col-xl-2 pl-xl-4 pr-xl-4 col-lg-2 col-md-3 pl-md-4 pr-md-4 col-sm-3 h-45 dash_shadow">
            <app-stat class="h-100" [topBgClass]="'top_claim'" [bgClass]="'claim '" [bgCount]="'claim_count '" [icon]="'fa-support'" [label]="'Claims'" [routerLink]="['/claim']" [routerLinkActive]="['router-link-active']" [queryParams]="{inprogress:2}" skipLocationChange ></app-stat>
        </div>
            <div class="col-xl-2 pl-xl-4 pr-xl-4 col-lg-2 col-md-3 pl-md-4 pr-md-4 col-sm-3 h-45 dash_shadow" >
            <app-stat class="h-100" [topBgClass]="'top_doc'" [bgClass]="' doc '" [bgCount]="'doc_count '" [icon]="'fa-tasks'" [count]="this.doc" [label]="'Document View'" [routerLink]="['/docview']" [routerLinkActive]="['router-link-active']" [queryParams]="{inprogress:0}" skipLocationChange></app-stat>
       </div>
  </div>
      <div class="row content-center h-100" *ngIf="RequestDashboard">
       <div class="col-xl-2 pl-xl-4 pr-xl-4 col-lg-2 col-md-3 pl-md-4 pr-md-4 col-sm-3 h-45 dash_shadow" >
            <app-stat class="h-100" [topBgClass]="'top_personal_details '" [bgClass]="'personal_details '" [bgCount]="'personal_details_count '" [icon]="'fa-support'" [count]="this.req" [label]="'Request'" [routerLink]="['/request']" [routerLinkActive]="['router-link-active']" [queryParams]="{inprogress:2}" skipLocationChange ></app-stat>
        </div>
        </div>
  <div class="row content-center h-100" *ngIf="customerService">
  <div class="row pt-3 h-80 overflow-y">
  <form class="max-height col-12 p-0" role="form" method="POST" [formGroup]="customerSearchForm"
   autocomplete="off" novalidate>
     <div class="row m-0">
       <div class="col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12">
      <mat-form-field>
       <input matInput placeholder="Name" formControlName="clientName" >
      </mat-form-field>
    </div>
   </div>
   <div class="row m-0">
    <div class="col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12">
      <mat-form-field>
       <input type="number" matInput placeholder="Policy No." formControlName="policyNo" >
      </mat-form-field>
    </div>
   </div>
   <div class="row m-0">
    <div class="col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12">
      <mat-form-field>
       <input type="number" matInput placeholder="NRIC" formControlName="clientNric" >
      </mat-form-field>
    </div>
   </div>
   <div class="row m-0 center-align text-right">
  <div class="col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12">
<button type="button" mat-stroked-button class="custom-btn btn-border"
(click)="customerSearch(contentNewCase)" [disabled]="!customerSearchForm.valid" >Search</button>
   <button type="reset" class="custom-btn submit-btn btn-border" mat-stroked-button style="margin-left:
15px;">Reset</button>
  </div>
  </div>
       </form>
   </div>
    </div>
   <div class="container-fluid h-50" *ngIf="viewTable">
<div class="row">
    <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6 d-flex align-items-center">
      <label class="tasklistLabel"></label>
    </div>
    <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6 text-right">
      </div>
</div>
<div class="row m-0 h-80">
  <div class="col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12 p-0 h-100">
    <div class="mat-elevation-z8 example-container h-100 table-striped">
      <table mat-table [dataSource]="dataSource" matSort>
        <!-- ID Column -->
      <ng-container matColumnDef="id">
          <th mat-header-cell *matHeaderCellDef mat-sort-header> No. </th>
          <td mat-cell *matCellDef="let row; let i = index;"> {{i+1}} </td>
        </ng-container>
<ng-container matColumnDef="creationDate">
          <th mat-header-cell *matHeaderCellDef mat-sort-header> Claim Date </th>
          <td mat-cell *matCellDef="let row"> {{row.claimDate}} </td>
        </ng-container>
        <!-- Progress Column -->
        <ng-container matColumnDef="appNo">
          <th mat-header-cell *matHeaderCellDef mat-sort-header> App No </th>
          <td mat-cell *matCellDef="let row"> {{row.appNo}} </td>
        </ng-container>
```

```
<!-- Name Column -->
<ng-container matColumnDef="agentCode">
  <th mat-header-cell *matHeaderCellDef mat-sort-header> Agent Code </th>
  <td mat-cell *matCellDef="let row"> {{row.agentCode}} </td>
</ng-container>
<!-- Color Column -->
<ng-container matColumnDef="custName">
  <th mat-header-cell *matHeaderCellDef mat-sort-header> Cust. Name </th>
  <td mat-cell *matCellDef="let row" > {{row.clientName}} </td>
</ng-container>
<ng-container matColumnDef="planName">
  <th mat-header-cell *matHeaderCellDef mat-sort-header> Plan Name </th>
  <td mat-cell *matCellDef="let row" > {{row.planName}} </td>
</ng-container>
<ng-container matColumnDef="premiumAmt">
  <th mat-header-cell *matHeaderCellDef mat-sort-header> Sum Assured($) </th>
  <td mat-cell *matCellDef="let row" > {{row.claimAmount}} </td>
</ng-container>
<ng-container matColumnDef="inprogress" *ngIf="isHideWIP">
  <th mat-header-cell *matHeaderCellDef mat-sort-header> Status </th>
  <td mat-cell *matCellDef="let row" > {{row.inprogress}} </td>
</ng-container>
<ng-container matColumnDef="datasource" >
  <th mat-header-cell *matHeaderCellDef mat-sort-header> Data Source </th>
  <td mat-cell *matCellDef="let row" > {{row.datasource}} </td>
</ng-container>
  <ng-container matColumnDef="submitDate" *ngIf="isHistory">
  <th mat-header-cell *matHeaderCellDef mat-sort-header> Submitted <br> Date & Time </th>
  <td mat-cell *matCellDef="let row" > {{row.submitDate}} </td>
</ng-container>
```

```
        <ng-container matColumnDef="edit" >
          <th mat-header-cell *matHeaderCellDef mat-sort-header>  </th> <!-- Edit|View -->
          <td mat-cell *matCellDef="let row" >
 <button  mat-button *ngIf="isEdit" (click)="editData(row)" class="edit-btn small-vertical-line-right"></button>
            <button  mat-button (click)="taskView(row)" class="view-btn"></button></td>
        </ng-container>
         <tr mat-header-row *matHeaderRowDef="displayedColumns; sticky: true"></tr>
        <tr mat-row *matRowDef="let row; columns: displayedColumns; let i = index; " >
        </tr>
      </table>
      <mat-paginator [pageSizeOptions]="[5, 10, 25, 100]" class="mat-paginator-sticky"></mat-paginator>
    </div>
  </div>
</div>
    </div>
</div>
<ng-template #contentNewCase  let-c="close" let-d="dismiss">
  <div class="modal-header">
      <h4 class="modal-title">Client Details</h4>
      <button type="button" id="closeBtn" class="close" aria-label="Close" (click)="[c('Cross click'),
claimData.reset(), customerSearchForm.reset()]">
          <span aria-hidden="true">&times;</span>
      </button>
  </div>
  <div class="modal-body">
    <form class="max-height col-12 p-0" role="form" [formGroup]="claimData" method="POST"
      autocomplete="off" novalidate>
<div class="row m-0">
        <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6">
          <mat-form-field>
            <input type="number" matInput placeholder="Policy No." formControlName="policyNo">
          </mat-form-field>
        </div>
        <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6">
          <mat-form-field>
            <input matInput placeholder="Plan Name" formControlName="planName">
          </mat-form-field>
        </div>
      </div>
<div class="row m-0">
        <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6">
          <mat-form-field>
            <input matInput placeholder="Client Name" formControlName="clientName">
          </mat-form-field>
        </div>

        <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6">
          <mat-form-field>
            <input matInput placeholder="Nominee Name" formControlName="nomineeName">
          </mat-form-field>
        </div>
      </div>
      <div class="row m-0">
        <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6">
          <mat-form-field>
            <input type="number" matInput placeholder="Nominee NRIC" formControlName="nomineeNric">
          </mat-form-field>
        </div>
        <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6">
          <mat-form-field>
```

```
            <input type="number" matInput placeholder="Claim Amount(in $)" formControlName="claimAmount"
>
          </mat-form-field>
        </div>
      </div>
      <div class="row m-0">
        <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6">
          <mat-form-field>
            <input matInput placeholder="Status" formControlName="status">
          </mat-form-field>
        </div>
        <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6">
          <mat-form-field class="example-full-width">
            <input matInput [matDatepicker]="picker" placeholder="Claim Date" name="dp"
formControlName="claimDate">
            <mat-datepicker-toggle matSuffix [for]="picker"></mat-datepicker-toggle>
            <mat-datepicker #picker disabled="false"></mat-datepicker>
          </mat-form-field>
        </div>
      </div>
      <div class="row m-0">
        <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6">
          <mat-form-field>
            <input  matInput placeholder="Nominee Bank Name" formControlName="nomineeBank" >
          </mat-form-field>
        </div>
        <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6">
          <mat-form-field>
    <input type="number" matInput placeholder="Account Number" formControlName="nomineeAccountNo">
          </mat-form-field>
    </div>
   </div>
   <div class="row m-0">
    <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6">
   <mat-form-field>
        <input matInput placeholder="Death Certificate" formControlName="remark">
      </mat-form-field>
    </div>
   </div>
  </form>
 </div>
</ng-template>
<ng-template #content let-c="close" let-d="dismiss" class="scale50"  style="position:relative;">
 <div class="modal-header border-0" id="mydiv" >
<div class="d-inline-flex pr-2  w-100 small-vertical-line-right" >
  <div class="row m-0 w-100  text-left">
            <div class="w-100 col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12 p-0 ">
             <h4>Payment Due</h4>
             <br>
             <div class="mat-elevation-z8 example-container  table-striped ">
               <div class="w-100 scroll">
              <table mat-table [dataSource]="data" id="mytable" matSort>
              <!-- ID Column -->
             <ng-container matColumnDef="id">
                <th mat-header-cell *matHeaderCellDef mat-sort-header>Due Date </th>
                <td mat-cell *matCellDef="let row;"> {{da}} </td>
               </ng-container>
                 <ng-container matColumnDef="NomineeDateOfBirth">
                <th mat-header-cell *matHeaderCellDef mat-sort-header>Application No</th>
                <td mat-cell *matCellDef="let row" > {{row.appNo}} </td>
```

```
                    </ng-container>
                     <ng-container matColumnDef="appNo">
                      <th mat-header-cell *matHeaderCellDef mat-sort-header> Premium Amount </th>
                      <td mat-cell *matCellDef="let row"> {{row.Premium}} </td>
                    </ng-container>
                    <ng-container matColumnDef="creationDate" >
                      <th mat-header-cell *matHeaderCellDef mat-sort-header>Action</th>
                      <td mat-cell *matCellDef="let row"> {{action}} </td>
                    </ng-container>
                    <tr mat-header-row *matHeaderRowDef="displayColumns; sticky: true"></tr>
                    <tr mat-row *matRowDef="let row; columns: displayColumns; let i = index; " >
                    </tr>
                   </table>
                   </div>
        <mat-paginator [pageSizeOptions]="[5, 10, 25, 100]" class="mat-paginator-sticky"></mat-paginator>
                  </div>
                 </div>
                </div>
              </div>
       <button type="button" class="close " aria-label="Close" (click)="d('Cross click')" >
        <span aria-hidden="true">&times;</span>
       </button>
  </div>
</ng-template>
<ng-template #request let-c="close" let-d="dismiss" class="scale50"  style="position:relative;">
  <div class="modal-header border-0" id="mydiv" >
     <div class="d-inline-flex pr-2  w-100 small-vertical-line-right" >
    <div class="row m-0 w-100  text-left">
                 <div class="w-100 col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12 p-0 ">
                  <h4>Request Status:</h4>
```

```
                    <br>
                    <div class="mat-elevation-z8 example-container  table-striped ">
                      <div class="w-100 scroll">
                     <table mat-table [dataSource]="dataStatus" id="mytable" matSort>
                      <!-- ID Column -->
                     <ng-container matColumnDef="id">
                        <th mat-header-cell *matHeaderCellDef mat-sort-header>No. </th>
                        <td mat-cell *matCellDef="let row; let i = index;"> {{i+1}} </td>
                      </ng-container>
                          <ng-container matColumnDef="NomineeDateOfBirth">
                        <th mat-header-cell *matHeaderCellDef mat-sort-header>Request Type</th>
                        <td mat-cell *matCellDef="let row" > {{row.requestType}} </td>
                      </ng-container>
                       <ng-container matColumnDef="appNo">
                        <th mat-header-cell *matHeaderCellDef mat-sort-header> Status </th>
                        <td mat-cell *matCellDef="let row;let c=index" > {{stat[c]}} </td>
                      </ng-container
                      <tr mat-header-row *matHeaderRowDef="dispColumns; sticky: true"></tr>
                      <tr mat-row *matRowDef="let row; columns: dispColumns; let i = index; " >
                      </tr>
                     </table>
                     </div>
             <mat-paginator [pageSizeOptions]="[5, 10, 25, 100]" class="mat-paginator-sticky"></mat-paginator>
                    </div>
                   </div>
                  </div>
    </div>
        <button type="button" class="close " aria-label="Close" (click)="d('Cross click')" >
          <span aria-hidden="true">&times;</span>
         </button>
```

```
  </div>
</ng-template>
```

Dashboard.component.scss

```
.margin-top-2{
margin-top: 2%;
}
.dash_shadow:after {
		content: "";
		display: block;
		width: 100%;
		height: 15%;
		background: linear-gradient(#e8e8e8,#fff);
		margin-top: .75rem;
}
.topnav-right {
  float: right;
  text-align:right !important;
}
.hiddenClass{
  display: none !important;
}
.newcase{
  background-color: #b12a31;
  color: #fff;
}
a{
  text-decoration:underline;
}
.newtask{
  background-color: #faeba5;
```

```
}
.example-container {
  display: flex;
  flex-direction: column;
}
.example-container > * {
  width: 100%;
}
.max-height{
    height: 100% !important;
}
.mat-form-field{
  width: 100%;
  font-size: .87rem;
}
.form-container{
		/*border-top: 2px solid rgba(182, 182, 182, 0.5);*/
		padding: 30px;
  /*margin-top: 30px;*/
  padding-bottom: 0px;
}
.mat-form-field-appearance-legacy  .mat-form-field-prefix  .mat-datepicker-toggle-default-icon,  .mat-form-field-
appearance-legacy .mat-form-field-suffix .mat-datepicker-toggle-default-icon{
			color: #b12a31;
}
.zero-padding{
  padding-right: 0px !important;
  padding-left: 0px !important;
}
.tr{
  border:0em;
  text-align:center;
}
.blinking{
    animation:blinkingText 2s infinite;
}
@keyframes blinkingText{
    0%{     color: #000;    }
    49%{    color: #000; }
    60%{    color: transparent; }
    99%{    color:transparent;  }
    100%{   color: #000;    }
}
.mat-form-field {
  font-size: 14px;
  width: 100%;
}
.example-container {
  // height: 400px;
  overflow-y: auto;
}
.mat-elevation-z8{
		box-shadow: none!important;
}
.scale50{
  height:50em !important;
}
button.custom-btn.task-btn{
  font-size: .9rem !important;
  line-height: 33px !important;
  border: none !important;
}
.wip{
  background-color: #ffc350;
}
.rc{
  background-color: #ff9250;
}
.newcase{
  background-color: #b12a31;
  color: #fff;
}
.newtask{
		background-color: #faeba5;
}
.uw{
background-color: #f07450;
}
.dm{
background-color: #ef5f50;
}
tr.mat-footer-row, tr.mat-row {
    height: 40px !important;
}
.mat-cell {
    font-size: 12px;
}
.filterClass{
		width: 50%;
}
```

```
.tasklistLabel{
		font-size: 1rem;
		// margin-left: 22px;
}
tr.mat-header-row{
  height: 2.5em !important;
}
tr.mat-row{
  height: 3.5em !important;
}
table{
  width:100% !important;
}
.content-center{
  display: flex;
  justify-content: center;
}
::selection {
    text-shadow: none;
    background: #FE2E64;
    color: #fff;
}
```

Dashboard.component.ts

```
export interface UserData {
  id: string;
  name: string;
  progress: string;
  color: string;
}
@Component({
    selector: 'app-dashboard',
    templateUrl: './dashboard.component.html',
    styleUrls: ['./dashboard.component.scss'],
    //encapsulation: ViewEncapsulation.None,
    animations: [routerTransition()]
})
export class DashboardComponent implements OnInit {
 configChange: EventEmitter<ImageViewerConfig> = new EventEmitter();
  subscription: Subscription;
displayColumns: string[] = ['id','NomineeDateOfBirth', 'appNo' ,'creationDate'];
dispColumns: string[] = ['id','NomineeDateOfBirth', 'appNo'];
  data: MatTableDataSource<UserData>;
  dataStatus: MatTableDataSource<UserData>;
  @ViewChild(MatPaginator) paginator: MatPaginator;
  @ViewChild(MatSort) sort: MatSort;
@Input() userId: string;
public action:string;
public users:any=[];
public details="0";
public personal="0";
public payment="0";
public policy=0;
public req="0";
public claim="0";
public space=" ";
public doc="0";
nomineeDashboard: boolean = false;
detailsDashboard:boolean=false;
CustomerDashboard:boolean=false;
ClaimDashboard:boolean=false;
```

```
RequestDashboard:boolean=false;
customerService: boolean =false;
viewTable:boolean=false;
public da;
public status;
customerSearchForm = new FormGroup({
   policyNo: new FormControl(''),
   clientNric: new FormControl(''),
   clientName: new FormControl(''),
  });
claimData = new FormGroup({
   policyNo: new FormControl(''),
   planName: new FormControl(''),
   nomineeName: new FormControl(''),
   nomineeNric: new FormControl(''),
   nomineeBank: new FormControl(''),
   nomineeAccountNo: new FormControl(''),
   appNo: new FormControl(''),
   clientName: new FormControl(''),
   claimAmount: new FormControl(''),
   status: new FormControl(''),
   claimDate: new FormControl(''),
   remark: new FormControl('',Validators.required)
  });
  closeResult: string;
  disp:Array<any>;
  searchResult: any[];
ReadonlyVAPID_PUBLIC_KEY="BAhICen0IkJ9xUpAxpRV6zQDhD3kNojC2KAA-PeRNq-
Vlfzi7sPh0hlk0idalzKmiKP-So4gMDCRKz7slMGNlns";
   public appNo:number;
```

```
    public alerts: Array<any> = [];
    public sliders: Array<any> = [];
constructor(private dataService:DataService,private personalService:PersonalService,private downloadService:DownloadService,private paymentService:PaymentService,private docviewService:DocviewService,private summaryService:SummaryService,private requestService:RequestService,public commonFunction:CommonFunction, private modalService: NgbModal) {
    }
    ngOnInit() {
    this.customerSearchForm.setValidators(this.atLeastOneValidator())
   var txt = JSON.parse(localStorage.getItem('usersinfo'));
this.appNo=txt[0]["appNo"];
this.userId=txt[0]["_id"];
    var teamId=(txt[0]["teamId"]);
 var body={userId:this.userId , teamId:teamId, inprogress:[0,1,2]};
if(teamId==11){
  this.customerService=true;
}
else if(teamId==13){
  this.commonDashboard=false;
  this.PruBsnDashboard=false;
  this.detailsDashboard=true;
  var bod={appNo:this.appNo}
     this.personalService.getClaimDetailsFromDB(bod).subscribe(res=>{
      var dataObj=JSON.stringify(res["data"]);
      var obj=JSON.parse(dataObj);
      var status=obj[0]['status'];
      if(status=='Claim initiated' || status=='Claim Accepted')
      {
        this.status="pending";
      }
```

```
else if(status=='Claim Approved')
{
this.status="Claim Approved";
}
else if(status=='Claim Rejected')
{
this.status="Claim Rejected";
}
else
{
this.status=" ";
}
console.log("Status------->"+status);
});
 var b={appNo:this.appNo,userId:this.userId , teamId:teamId, inprogress:[0,1,2]};
 this.personalService.getPersonalDetailsFromDB(b).subscribe(res=>{
   var dataObj=JSON.stringify(res['data']);
   var obj=JSON.parse(dataObj);
   this.personal=obj.length;
 });
  this.paymentService.getDetailsFromDB(b).subscribe(res=>{
   var dataObj=JSON.stringify(res['data']);
   var obj=JSON.parse(dataObj);
   this.payment=obj.length;
 });
  this.downloadService.getclaim(b).subscribe(res=>{
   var dataObj=JSON.stringify(res['data']);
   var obj=JSON.parse(dataObj);
   this.policy=res['data'].length;
   console.log(obj);
```

```
          console.log("length"+this.policy);
        });
        this.requestService.getDetailFromDB(b).subscribe(res=>{
          var dataObj=JSON.stringify(res['data']);
          var obj=JSON.parse(dataObj);
          this.req=obj.length;
          console.log("LEngth------>"+this.req);
        });
}
 else if(teamId==14){
  this.commonDashboard=false;
  this.PruBsnDashboard=false;
  this.CustomerDashboard=true;
  var b={appNo:this.appNo,userId:this.userId , teamId:teamId, inprogress:[0,1,2]};
        this.summaryService.getDetailFromDB(b).subscribe(res=>{
          var dataObj=JSON.stringify(res['data']);
          var obj=JSON.parse(dataObj);
          this.claim=obj.length;
        });
  console.log("customerDashboard---->"+this.CustomerDashboard);
}
 else if(teamId==15){
  this.commonDashboard=false;
  this.PruBsnDashboard=false;
  this.ClaimDashboard=true;
   var b={appNo:this.appNo,userId:this.userId , teamId:teamId, inprogress:[0,1,2]};
        this.summaryService.getDetailsFromDB(b).subscribe(res=>{
          var dataObj=JSON.stringify(res['data']);
          var obj=JSON.parse(dataObj);
          this.claim=obj.length;
```

```
       });
       var b={appNo:this.appNo,userId:this.userId , teamId:teamId, inprogress:[0,1,2]};
       this.docviewService.getImageDetailFromDB(b).subscribe(res=>{
         var dataObj=JSON.stringify(res['data']);
         var obj=JSON.parse(dataObj);
         this.doc=obj.length;
       });
}
 else if(teamId==16){
  this.commonDashboard=false;
  this.PruBsnDashboard=false;
  this.RequestDashboard=true;
  var b={appNo:this.appNo,userId:this.userId , teamId:teamId, inprogress:[0,1,2]};
   this.requestService. getDetailsFromDB(b).subscribe(res=>{
         var dataObj=JSON.stringify(res['data']);
         var obj=JSON.parse(dataObj);
         this.req=obj.length;
         console.log("LEngth------>"+this.req);
       });
}
 else if(teamId==17){
  this.commonDashboard=false;
  this.PruBsnDashboard=false;
  this.nomineeDashboard=true;
   var bod={appNo:this.appNo}
     this.personalService.getClaimDetailsFromDB(bod).subscribe(res=>{
      var dataObj=JSON.stringify(res["data"]);
      var obj=JSON.parse(dataObj);
      var status=obj[0]['status'];
      if(status=='Claim initiated' || status=='Claim Accepted')
      {
        this.status="pending";
      }
      else if(status=='Claim Approved')
      {
      this.status="Claim Approved";
      }
      else if(status=='Claim Rejected')
      {
      this.status="Claim Rejected";
      }
      else
      {
      this.status=" ";
      }
      console.log("Status------->"+status);
      });
}
  customerSearch(content)
{
this.claimData.reset();
  var customerSearchObj = this.customerSearchForm.getRawValue();
  console.log("policyNo:"+customerSearchObj.policyNo);
  var body={customerSearchForm:customerSearchObj};
    this.personalService.searchCustomerDetails(body).subscribe(res =>
    {
      if(res['data']=="" || res['data']==[] || res['data']=={}){
          this.commonFunction.openDialog(res['status'],  '');
          this.customerSearchForm.reset();
          this.claimData.reset();
    }else{
    console.log("data---->"+res['data']);
    var claimDataObj= JSON.stringify(res["data"])
    var obj=JSON.parse(claimDataObj);
  if(obj!="" && obj!=[] && obj!=null)
    {
        this.claimData.patchValue(obj[0]);
        this.claimData.disable();
        this.customerSearchForm.setValidators(this.atLeastOneValidator())
        this.customerSearchForm.reset();
        this.openNewCaseModle(content)
    }
}
 });
}
private atLeastOneValidator = () => {
  return (controlGroup) => {
      let controls = controlGroup.controls;
      if ( controls ) {
          let theOne = Object.keys(controls).find(key=> controls[key].value!=='');
          if ( !theOne ) {
              return {
             atLeastOneRequired : {
                   text : 'At least one should be selected'
                 }
              }
          }
      }
      return null;
  };
```

```
};
 modalReference = null;
   openNewCaseModle(content) {
   this.modalReference=this.modalService.open(content,
       { centered: true, backdrop: 'static', keyboard: false, backdropClass: 'dark-backdrop', windowClass: 'custom-
modal' })
        this.modalReference.result.then((result) => {
            this.closeResult = `Closed with: ${result}`;
        }, (reason) => {
            this.closeResult = `Dismissed ${this.getDismissReason(reason)}`;
        });
    }
private getDismissReason(reason: any): string {
    this.customerSearchForm.reset();
      if (reason === ModalDismissReasons.ESC) {
            return 'by pressing ESC';
        } else if (reason === ModalDismissReasons.BACKDROP_CLICK) {
            return 'by clicking on a backdrop';
        } else {
            return  `with: ${reason}`;
        }
    }
uses:Array<any>;
tasklist:Array<any>;
public stats=[];
public stat=[];
user:Array<any>;
dataSource: MatTableDataSource<UserData>;
    viewDetails(){
var financeFormObj = this.financeForm.getRawValue();
var body={financeForm:financeFormObj};
    this.personalService.financeReport(body).subscribe(res =>
   {
     this.viewTable=true;
      console.log("data---->"+res['data']);
      var claimDataObj= JSON.stringify(res["data"])
      var obj=JSON.parse(claimDataObj);
      if(obj!="" && obj!=[] && obj!=null)
      {
      }
 var dataObj= JSON.stringify(res['data'])
      console.log("data ojb -->"+dataObj);
      var obj=JSON.parse(dataObj);
      this.tasklist=obj;
      const users = Array.from(this.tasklist);
    this.dataSource = new MatTableDataSource(users);
    this.dataSource.paginator = this.paginator;
    this.dataSource.sort = this.sort;
   });
    }
  displayedColumns: string[] = ['id','creationDate', 'appNo', 'agentCode', 'custName', 'planName', 'premiumAmt',
'datasource','edit' ];
resetForm(){
  this.viewTable=false;
}
setFrequency(value){
console.log("frequencyValue--->"+value);
this.isMonthly=false;
this.isQuarterly=false;

  if(value=='Monthly'){
```

```
  this.isMonthly=true;
  }else if(value=='Quarterly'){
  this.isQuarterly=true;
  }
}

isMonthly:boolean=false;
isQuarterly:boolean=false;
/*dialogue gets opens or called*/
 private open(content) {
       this.modalService.open(content,
        { centered: true, backdrop: 'static', keyboard: false, backdropClass: 'dark-backdrop'})
          .result.then((result) => {
           width:"100%";
           //height:"100em";
           this.closeResult = `Closed with: ${result}`;
       }, (reason) => {
           this.closeResult = `Dismissed ${this.getDismissReason(reason)}`;
        });
        this.getData();
    }
  /*dialogue is closed or dismissed*/
getRequest()
{
   var bod={appNo:this.appNo}
     this.personalService.getRequestDetailsFromDB(bod).subscribe(res=>{
      var dataObj=JSON.stringify(res["data"]);
      var obj=JSON.parse(dataObj);
      for(var i=0;i<=obj.length-1;i++)
      {
```

```
      console.log(obj[i]['status']);
       this.stats[i]=obj[i]['status'];
       if(this.stats[i]=="Request intiated"||this.stats[i]=="Documents Uploaded")
       {
       this.stat[i]="pending";
       }
       else if(this.stats[i]=="Request Accepted")
       {
       this.stat[i]="Request Accepted";
       }
       else if(this.stats[i]=="Request Rejected")
       {
       this.stat[i]=="Request Rejected";
       }
       else if(this.stats[i]=="inprogress")
       {
       this.stat[i]=="Upload Documents";
       }
       console.log(this.stat[i]);
      }
      this.user=obj;
    console.log("users--------->"+this.user);
     const users = Array.from(this.user);
        console.log("users--------->"+users);
    this.dataStatus= new MatTableDataSource(users);
    this.dataStatus.paginator = this.paginator;
    this.dataStatus.sort = this.sort;
 });
}
getData()
```

```
{
  var bod={appNo:this.appNo}
    this.personalService.getPolicyDetailsFromDB(bod).subscribe(res=>{
     var dataObj=JSON.stringify(res["data"]);
     var obj=JSON.parse(dataObj);
     var date=obj[0]['DateCreated'];
     var full=date.split("-");
     var y=full[0];
     var m=full[1];
     var d=full[2];
     var year=Number(y);
     var month=Number(m);
     var day=Number(d);
     console.log("Year------->"+year);
     console.log("Month------->"+typeof(month));
     console.log("Date------->"+day);
     var freq=obj[0]['PaymentFrequency'];
      var dt = new Date(full);
     if(freq=="quarterly")
     {
        dt.setMonth( dt.getMonth() + 4 );
       this.da=dt.toISOString().split('T')[0];
        console.log("Date--->"+this.da);
     }
     else if(freq=="HalfYearly")
     {
      dt.setMonth( dt.getMonth() + 6 );
        this.da=dt.toISOString().split('T')[0];
        console.log("Date--->"+this.da);
     }
```

```
      else if(freq=="monthly")
      {
         dt.setMonth( dt.getMonth() + 1 );
         this.da=dt.toISOString().split('T')[0];
         console.log("Date--->"+this.da);
      }
      else if(freq=="yearly")
      {
         dt.setFullYear(dt.getFullYear() + 1);
         this.da=dt.toISOString().split('T')[0];
         console.log("Date--->"+this.da);
      }
      this.action="pending";
      this.uses=obj;
    console.log("users--------->"+this.uses);
     const users = Array.from(this.uses);
        console.log("users--------->"+users);
    this.data = new MatTableDataSource(users);
    this.data.paginator = this.paginator;
    this.data.sort = this.sort;
 });
}
}
```

Header.component.html

```
<nav class="navbar navbar-expand">
    <a class="navbar-brand font-color-red" [routerLink]="['/dashboard']" [routerLinkActive]="['router-link-
active']"
    skipLocationChange (click)="hideSidebar()" >
        <img src="../../../../assets/images/logo_white.png" width="29" />
    </a>
    <div class="collapse navbar-collapse">
        <ul class="navbar-nav ml-auto">
            <li class="nav-item" (click)="hideSidebar()" *ngIf="admin"><label
class="teamnameLabel">{{userName}} <b>{{teamName}} team</b></label></li>
            <li class="nav-item" (click)="hideSidebar()" *ngIf="user"><label
class="teamnameLabel">{{userName}} <b>{{teamName}}</b></label></li>
            <li class="nav-item dropdown" ngbDropdown placement="bottom-right">
                <a href="javascript:void(0)" class="nav-link" ngbDropdownToggle (click)="hideSidebar()">
                    <img src="../../../assets/images/notification.png" style="max-width: 45%;">
                    <b class="caret"></b><span class="sr-only">(current)</span>
                </a>
                <div class="dropdown-menu-right" ngbDropdownMenu>
                    <a class="dropdown-item"
                    [routerLink]="['/tasklist']" [routerLinkActive]="['router-link-active']"
[queryParams]="{inprogress:0}"
                    skipLocationChange >
                        {{ 'New Task' | translate }} <span class="badge badge-info">{{newTaskCount}}</span>
                    </a>
                    <a  class="dropdown-item"
                    [routerLink]="['/tasklist']" [routerLinkActive]="['router-link-active']"
[queryParams]="{inprogress:1}" skipLocationChange >
                        {{ 'Work In Progress' | translate }} <span class="badge badge-info"> {{workInProgress}}</span>
                    </a>
                     <a  class="dropdown-item"
                     [routerLink]="['/tasklist']" [routerLinkActive]="['router-link-active']"
[queryParams]="{inprogress:2}" skipLocationChange >
                        {{ 'Rescaned Cases' | translate }} <span class="badge badge-info"> {{rescanCase}}</span>
                    </a>
                    <a  class="dropdown-item"
```

```
                 [routerLink]="['/policy']"                                   [routerLinkActive]="['router-link-active']"
[queryParams]="{inprogress:3}" skipLocationChange *ngIf="pol">
                    {{ 'Policy Service' | translate }} <span class="badge badge-info"> {{policyService}}</span>
                </a>
                <a  class="dropdown-item"
                 [routerLink]="['/personal']"                                 [routerLinkActive]="['router-link-active']"
[queryParams]="{inprogress:3}" skipLocationChange *ngIf="per">
                    {{ 'Personal Details' | translate }} <span class="badge badge-info"> {{personalDetails}}</span>
                </a>
                <a  class="dropdown-item"
                 [routerLink]="['/Customer']"                                 [routerLinkActive]="['router-link-active']"
[queryParams]="{inprogress:3}" skipLocationChange *ngIf="cus">
                    {{    'Customer    Service'    |    translate    }}    <span    class="badge    badge-info">
{{CustomerService}}</span>
                </a>
                <!-- <a href="javascript:void(0)" class="dropdown-item">
                    {{ 'Mail' | translate }} <span class="badge badge-info"> 45</span>
                </a> -->
                <li class="dropdown-divider"></li>
                <a                  [routerLink]="['/tasklist']"                 [routerLinkActive]="['router-link-active']"
[queryParams]="{inprogress:'AllRecord'}"skipLocationChange    href="javascript:void(0)"    class="dropdown-
item">
                    {{ 'View All' | translate }}
                </a>
            </div>
        </li>
        <li class="nav-item dropdown" ngbDropdown placement="bottom-right">
            <a href="javascript:void(0)" class="nav-link" ngbDropdownToggle (click)="hideSidebar()">
                <!-- <i class="fa fa-language"></i> {{ 'Language' | translate }} <b class="caret"></b> -->
                <img src="../../../assets/images/language.png" style="max-width: 55%;">  <b class="caret"></b>
```

```
  </a>
  <div class="dropdown-menu-right" ngbDropdownMenu>
    <a class="dropdown-item" href="javascript:void(0)" (click)="changeLang('en')">
      {{ 'English' | translate }}
    </a>
    <a class="dropdown-item" href="javascript:void(0)" (click)="changeLang('mr')">
      {{ 'Marathi' | translate }}
    </a>
  </div>
</li>
<li class="nav-item dropdown" ngbDropdown placement="bottom-right">
  <a href="javascript:void(0)" class="nav-link" ngbDropdownToggle (click)="hideSidebar()">
    <!-- <i class="fa fa-user"></i> <b class="caret"></b> -->
    <img src="../../../assets/images/user.png" style="max-width: 50%;"> <b class="caret"></b>
  </a>
  <div class="dropdown-menu-right" ngbDropdownMenu>
    <a class="dropdown-item" href="javascript:void(0)">
      <i class="fa fa-fw fa-user"></i> {{ 'Profile' | translate }}
    </a>
    <a class="dropdown-item" href="javascript:void(0)">
      <i class="fa fa-fw fa-envelope"></i> {{ 'Inbox' | translate }}
    </a>
    <a class="dropdown-item" href="javascript:void(0)">
      <i class="fa fa-fw fa-gear"></i> {{ 'Settings' | translate }}
    </a>
    <a class="dropdown-item" [routerLink]="['/login']" (click)="onLoggedout()">
      <i class="fa fa-fw fa-power-off"></i> {{ 'Log Out' | translate }}
    </a>
  </div>
</li>
```

```
          <li class="nav-item">
   <button class="navbar-toggler displayBlock h-100" id="navbarBtn" type="button" (click)="toggleSidebar()">
                    <i class="fa fa-bars text-white" aria-hidden="true"></i>
                </button>
          </li>
        </ul>
    </div>
</nav>
<div class="mask" id="mask" (click)="toggleSidebar()"></div>
```

Header.component.scss

```
$topnav-background-color: #b12a31;
:host {
    .navbar {
        background-color: $topnav-background-color;
        padding: 0.025rem 1rem;
        font-size: .82rem;
        height: 100%;
        border-bottom: 2px solid #053c6e;
        .navbar-brand {
            color: #fff;
        }
        .nav-item > a {
            color: #fff;
            &:hover {
                color: #fff;
            }
        }
        .dropdown-menu{
            min-width: fit-content;
        }
```

```
  }
  ::selection {
  text-shadow: none;
  background: #FE2E64;
  color: #fff;
  }
  .messages {
    width: 300px;
    .media {
      border-bottom: 1px solid #ddd;
      padding: 5px 10px;
      &:last-child {
        border-bottom: none;
      }
    }
    .media-body {
      h5 {
        font-size: 13px;
        font-weight: 600;
      }
      .small {
        margin: 0;
      }
      .last {
        font-size: 12px;
        margin: 0;
      }
    }
  }
}
```

```
.font-color-red{
color:#b12a31 !important;
font-weight: 900;
}

.displayBlock{
    display: block!important;
}

.l-0px{
     left: 0px;
}
.l-235px{
    left: 235px;
}

.teamnameLabel{
    margin-bottom: 0;
    margin-top: 7px;
    color: #fff;
    margin-right: 10px;
}

.navbar-expand-lg .navbar-nav .dropdown-menu-right {
    right: 0;
    left: auto!important;
}

a.dropdown-item{
    padding: 0.25rem .85rem;
    color: #333;
    font-size: .85rem;
    &:hover {
        color: #b12a31;
    }
    &:active {
        background: transparent !important;
    }
}
```

Header.component.ts

```
@Component({
    selector: 'app-header',
    templateUrl: './header.component.html',
    styleUrls: ['./header.component.scss']
})
export class HeaderComponent implements OnInit {


public newTaskCount:string="0";
public workInProgress:string="0";
public rescanCase:string="0";
public policyService:string="0";
public personalDetails:String="0";
public CustomerService:String="0";
public pol:boolean=false;
public per:boolean=false;
public cus:boolean=false;
userName="Welcome";
teamName:string="";
userId: string;
teamId: string;
admin:boolean=true;
```

```
user:boolean=false;
    pushRightClass: string = 'push-left';

    constructor(private translate: TranslateService, public router: Router, private dataService:DataService) {

        this.translate.addLangs(['en', 'fr', 'ur', 'es', 'it', 'fa', 'de', 'zh-CHS','mr']);
        this.translate.setDefaultLang('en');
        const browserLang = this.translate.getBrowserLang();
        this.translate.use(browserLang.match(/en|fr|ur|es|it|fa|de|zh-CHS|mr/) ? browserLang : 'en');

        this.router.events.subscribe(val => {
        //this.toggleSidebar();


        //document.getElementById("navbarBtn").classList.add('displayNone');

            // if (
            //     val instanceof NavigationEnd &&
            //     window.innerWidth <= 1200 &&
            //     this.isToggled()
            // ) {
            //     this.toggleSidebar();
            // }
        });
    }
    ngOnInit() {
var txt = JSON.parse(localStorage.getItem('usersinfo'));
console.log("username---->"+txt[0]["username"]);

for (let i = 0; i < localStorage.length; i++){
  let key = localStorage.key(i);
  let value = localStorage.getItem(key);
  console.log(key, value);
```

```
}

this.userName=this.userName+" "+txt[0]["name"]+",";
this.userId=txt[0]["_id"];
 this.teamId=(txt[0]["teamId"]);
this.teamName=txt[1]["teamName"];
console.log("teamName--->"+this.teamName);
if(this.teamId=="11")
{
    this.pol=true;
}
else if((this.teamId=="13")||(this.teamId=="17"))
{
    this.per=true;
    this.user=true;
    this.admin=false;
}
else if(this.teamId=="14")
{
    this.cus=true;
}

this.dataService.setTaskCount$.subscribe((data) => {


                this.newTaskCount=data["pendingCounter"];
                this.workInProgress=data["inprogressCounter"];
                this.rescanCase=data["rescanCaseCounter"];
                this.policyService=data["policyServiceCounter"];
                this.personalDetails=data["detailsCounter"];
                this.CustomerService=data["CustomerServiceCounter"]
                console.log("responseObj---------->"+data["pendingCounter"]);
```

```
            }
        );
    }
    isToggled(): boolean {
        const dom: Element = document.querySelector('body');
        return dom.classList.contains(this.pushRightClass);
    }

    toggleSidebar() {
        const dom: any = document.querySelector('body');
        dom.classList.toggle(this.pushRightClass);


        if(dom.classList.contains(this.pushRightClass)){


var el = document.getElementById("sideBarIsActive");
console.log("el-------->"+el)
if(el!=null){
        if (el.classList.contains('r-0px')) {
            // document.getElementById("sideBarIsActive").classList.remove('l-0px');
            document.getElementById("sideBarIsActive").classList.remove('r-0px');


        }



// document.getElementById("sideBarIsActive").classList.add('l-235px');
document.getElementById("sideBarIsActive").classList.add('r-235px');
document.getElementById("mask").style.display = 'block';
// document.getElementsByTagName("app-sidebar")[0].setAttribute("style", "width:18.5% !important");
//  document.getElementsByTagName("section")[0].setAttribute("style", "width:81.5% !important");
}
```

```
}else{
        var el = document.getElementById("sideBarIsActive");
        if(el!=null){
        if (el.classList.contains("r-235px")) {
            document.getElementById("sideBarIsActive").classList.remove('r-235px');
            document.getElementById("mask").style.display = 'none';
        }


         document.getElementById("sideBarIsActive").classList.add('r-0px');
    //   document.getElementsByTagName("app-sidebar")[0].setAttribute("style", "width:0% !important");
    //    document.getElementsByTagName("section")[0].setAttribute("style", "width:100% !important");
       }
         }
    }

    rltAndLtr() {
        const dom: any = document.querySelector('body');
        dom.classList.toggle('rtl');
    }

    onLoggedout() {
        localStorage.removeItem('isLoggedin');
    }

    changeLang(language: string) {
        this.translate.use(language);
    }

    hideSidebar(){
        let el = document.getElementById("sideBarIsActive");
        if(el!=null){
            if (el.classList.contains("r-235px")){
                document.getElementById("sideBarIsActive").classList.remove('r-235px');
                document.getElementById("sideBarIsActive").classList.add('r-0px');
                document.getElementById("mask").style.display = 'none';
            }
        }
    }
}//class closing
```

Sidebar.component.html

```
<nav class="sidebar " id="sideBarIsActive" [ngClass]="{sidebarPushRight: isActive}">
    <div class="list-group">
        <a routerLink="/dashboard" [routerLinkActive]="['router-link-active']" class="list-group-item"
        (click)="toggleSidebar()">
            <i class="fa fa-fw fa-dashboard"></i>&nbsp;{{ 'Dashboard' | translate }}
        </a>
       <!--  <a [routerLink]="['/charts']" [routerLinkActive]="['router-link-active']" class="list-group-item">
            <i class="fa fa-fw fa-bar-chart-o"></i>&nbsp;{{ 'Charts' | translate }}
        </a> -->
        <a [routerLink]="['/tasklist']" [queryParams]="{inprogress:'AllRecord',callreason:'summary'}"
skipLocationChange
        *ngIf="isVisible" [routerLinkActive]="['router-link-active']" class="list-group-item"
(click)="toggleSidebar()">
            <i class="fa fa-fw fa-th-list"></i>&nbsp;{{ 'Summary List' | translate }}
        </a>

        <a [routerLink]="['/summary']" *ngIf="listViewForAllTeam" [routerLinkActive]="['router-link-active']"
        class="list-group-item" [queryParams]="{inprogress:'AllRecord',callreason:'summary'}"
skipLocationChange (click)="toggleSidebar()" >
            <i class="fa fa-fw fa-th-list"></i>&nbsp;{{ 'Summary List' | translate }}
        </a>
    <a [routerLink]="['/tasklist']" [routerLinkActive]="['router-link-active']"
    class="list-group-item" [queryParams]="{inprogress:'AllRecord',callreason:'history'}" skipLocationChange
(click)="toggleSidebar()" *ngIf="user">
      <i class="fa fa-fw fa-th-list"></i>&nbsp;{{ 'User History' | translate }}
    </a>
    <!-- <a [routerLink]="['/tasklist']" *ngIf="isAllTask" [routerLinkActive]="['router-link-active']" class="list-
group-item" [queryParams]="{inprogress:'AllRecord'}"skipLocationChange>

      <i class="fa fa-fw fa-bar-chart-o"></i>&nbsp;{{ 'Summary List' | translate }}

    </a> -->
    <a [routerLink]="['/policy']" [queryParams]="{inprogress:'AllRecord',callreason:'summary'}"
skipLocationChange
    *ngIf="isCustomer" [routerLinkActive]="['router-link-active']" class="list-group-item"
(click)="toggleSidebar()">
      <i class="fa fa-fw fa-bar-chart-o"></i>&nbsp;{{ 'Policy Servicing' | translate }}
    </a>

     <a [routerLink]="['/personal']" [queryParams]="{inprogress:'AllRecord',callreason:'summary'}"
skipLocationChange
    *ngIf="isDetails" [routerLinkActive]="['router-link-active']" class="list-group-item"
(click)="toggleSidebar()">
      <i class="fa fa-fw fa-user"></i>&nbsp;{{ 'Personal Details' | translate }}
    </a>
    <a [routerLink]="['/payment']" [queryParams]="{inprogress:'AllRecord',callreason:'summary'}"
skipLocationChange
    *ngIf="isPayment" [routerLinkActive]="['router-link-active']" class="list-group-item"
(click)="toggleSidebar()">
      <i class="fa fa-fw fa-download"></i>&nbsp;{{ 'Payment' | translate }}
    </a>
```

```
 <a [routerLink]="['/Customer']" [queryParams]="{inprogress:'AllRecord',callreason:'summary'}"
skipLocationChange
    *ngIf="isService" [routerLinkActive]="['router-link-active']" class="list-group-item"
(click)="toggleSidebar()">
      <i class="fa fa-fw fa-bar-chart-o"></i>&nbsp;{{ 'Customer Servicing' | translate }}
    </a>

 <a [routerLink]="['/download']" [queryParams]="{inprogress:'AllRecord',callreason:'summary'}"
skipLocationChange
    *ngIf="isDoc" [routerLinkActive]="['router-link-active']" class="list-group-item"
(click)="toggleSidebar()">
      <i class="fa fa-fw fa-file-o"></i>&nbsp;{{ 'Policy Documents' | translate }}
    </a>
  <a [routerLink]="['/claim']" [queryParams]="{inprogress:'AllRecord',callreason:'summary'}"
skipLocationChange
    *ngIf="isClaim" [routerLinkActive]="['router-link-active']" class="list-group-item"
(click)="toggleSidebar()">
      <i class="fa fa-fw fa-th-list"></i>&nbsp;{{ 'Claims' | translate }}
    </a>

    <a [routerLink]="['/request']" [queryParams]="{inprogress:'AllRecord',callreason:'summary'}"
skipLocationChange
    *ngIf="isRequest" [routerLinkActive]="['router-link-active']" class="list-group-item"
(click)="toggleSidebar()">
      <i class="fa fa-fw fa-th-list"></i>&nbsp;{{ 'Request' | translate }}
    </a>

    <a [routerLink]="['/editfreq']" [queryParams]="{inprogress:'AllRecord',callreason:'summary'}"
skipLocationChange
    *ngIf="isFrequency" [routerLinkActive]="['router-link-active']" class="list-group-item"
(click)="toggleSidebar()">
```

```
    <i class="fa fa-fw fa-money"></i>&nbsp;{{ 'Edit Payment Frequency' | translate }}
  </a>

  <a [routerLink]="['/editname']" [queryParams]="{inprogress:'AllRecord',callreason:'summary'}" skipLocationChange
  *ngIf="isName" [routerLinkActive]="['router-link-active']" class="list-group-item" (click)="toggleSidebar()">
    <i class="fa fa-fw fa-upload"></i>&nbsp;{{ 'Edit Name' | translate }}
  </a>
   <a [routerLink]="['/claimsubmition']" [queryParams]="{inprogress:'AllRecord',callreason:'summary'}" skipLocationChange
  *ngIf="isSubmit" [routerLinkActive]="['router-link-active']" class="list-group-item" (click)="toggleSidebar()">
    <i class="fa fa-fw fa-upload"></i>&nbsp;{{ 'Claim Submitions' | translate }}
  </a>
    <a [routerLink]="['/docview']" [queryParams]="{inprogress:'AllRecord',callreason:'summary'}" skipLocationChange
  *ngIf="isDo" [routerLinkActive]="['router-link-active']" class="list-group-item" (click)="toggleSidebar()">
    <i class="fa fa-fw fa-eye"></i>&nbsp;{{ 'Document View' | translate }}
  </a>

  <a [routerLink]="['/reqhist']" [queryParams]="{inprogress:'AllRecord',callreason:'summary'}" skipLocationChange
  *ngIf="isReq" [routerLinkActive]="['router-link-active']" class="list-group-item" (click)="toggleSidebar()">
    <i class="fa fa-fw fa-history"></i>&nbsp;{{ 'Request History' | translate }}
  </a>

   <div class="nested-menu">
    <a class="list-group-item" *ngIf="isAdmin" (click)="addExpandClass('pages')">
      <span><i class="fa fa-plus"></i>&nbsp; {{ 'Users' | translate }}</span>
```

```
        </a>
        <li class="nested" [class.expand]="showMenu === 'pages'">
            <ul class="submenu">
                <li>
                    <a [routerLink]="['/user']" [routerLinkActive]="['router-link-active']" class="list-group-item">
                    <i class="fa fa-fw fa-bar-chart-o"></i>&nbsp;{{ 'All User' | translate }}
                    </a>
                </li>
               <!--  <li>
                    <a [routerLink]="['/image']" [routerLinkActive]="['router-link-active']" class="list-group-item">
                    <i class="fa fa-fw fa-bar-chart-o"></i>&nbsp;{{ 'Image' | translate }}
                    </a>
                </li>
                <li>
                    <a    [routerLink]="['/chatapp']"    [routerLinkActive]="['router-link-active']"    class="list-group-
item">
                    <i class="fa fa-fw fa-bar-chart-o"></i>&nbsp;{{ 'Chat' | translate }}
                    </a>
                </li> -->
            </ul>
        </li>
    </div>
    <!-- <a [routerLink]="['/tables']" [routerLinkActive]="['router-link-active']" class="list-group-item">
        <i class="fa fa-fw fa-table"></i>&nbsp;{{ 'Tables' | translate }}
    </a> -->
    <!-- <a [routerLink]="['/forms']" [routerLinkActive]="['router-link-active']" class="list-group-item">
        <i class="fa fa-fw fa-edit"></i>&nbsp;{{ 'Forms' | translate }}
    </a> -->
    <!-- <a [routerLink]="['/bs-element']" [routerLinkActive]="['router-link-active']" class="list-group-item">
        <i class="fa fa-fw fa-desktop"></i>&nbsp;{{ 'Bootstrap Element' | translate }}
```

```
 </a> -->
<!--  <a [routerLink]="['/grid']" [routerLinkActive]="['router-link-active']" class="list-group-item">
    <i class="fa fa-fw fa-wrench"></i>&nbsp;{{ 'Bootstrap Grid' | translate }}
 </a> -->
 <!-- <a [routerLink]="['/components']" [routerLinkActive]="['router-link-active']" class="list-group-item">
    <i class="fa fa-th-list"></i>&nbsp;{{ 'Component' | translate }}
 </a> -->
 <!-- <div class="nested-menu">
    <a class="list-group-item" (click)="addExpandClass('pages')">
        <span><i class="fa fa-plus"></i>&nbsp; {{ 'Menu' | translate }}</span>
    </a>
    <li class="nested" [class.expand]="showMenu === 'pages'">
        <ul class="submenu">
            <li>
                <a href="javascript:void(0)"><span>{{ 'Submenu' | translate }}</span></a>
            </li>
            <li>
                <a href="javascript:void(0)"><span>{{ 'Submenu' | translate }}</span></a>
            </li>
            <li>
                <a href="javascript:void(0)"><span>{{ 'Submenu' | translate }}</span></a>
            </li>
        </ul>
    </li>
 </div> -->
 <!-- <a [routerLink]="['/blank-page']" [routerLinkActive]="['router-link-active']" class="list-group-item">
    <i class="fa fa-file-o"></i>&nbsp;{{ 'Blank Page' | translate }}
 </a> -->
<!--  <a class="list-group-item more-themes" href="http://www.strapui.com/" >
    {{ 'More Theme' | translate }}
    </a> -->
    <div class="header-fields">
      <a (click)="rltAndLtr()" class="list-group-item">
        <span><i class="fa fa-arrows-h"></i>&nbsp; RTL/LTR</span>
      </a>
      <div class="nested-menu">
        <a class="list-group-item" (click)="addExpandClass('languages')">
          <span><i class="fa fa-language"></i>&nbsp; {{ 'Language' | translate }} <b
class="caret"></b></span>
        </a>
        <li class="nested" [class.expand]="showMenu === 'languages'">
          <ul class="submenu">
            <li>
              <a href="javascript:void(0)" (click)="changeLang('en')">
                {{ 'English' | translate }}
              </a>
            </li>
           <!-- <li>
              <a href="javascript:void(0)" (click)="changeLang('fr')">
                {{ 'French' | translate }}
              </a>
            </li>
            <li>
              <a href="javascript:void(0)" (click)="changeLang('ur')">
                {{ 'Urdu' | translate }}
              </a>
            </li>
            <li>
              <a href="javascript:void(0)" (click)="changeLang('es')">
                {{ 'Spanish' | translate }}
                </a>
            </li>
            <li>
                <a href="javascript:void(0)" (click)="changeLang('it')">
                    {{ 'Italian' | translate }}
                </a>
            </li>
            <li>
                <a href="javascript:void(0)" (click)="changeLang('fa')">
                    {{ 'Farsi' | translate }}
                </a>
            </li>
            <li>
                <a href="javascript:void(0)" (click)="changeLang('de')">
                    {{ 'German' | translate }}
                </a>
            </li>-->
            <li>
                <a href="javascript:void(0)" (click)="changeLang('mr')">
                    {{ 'Marathi' | translate }}
                </a>
            </li>
        </ul>
    </li>
</div>
<div class="nested-menu">
    <a class="list-group-item" (click)="addExpandClass('profile')">
        <span><i class="fa fa-user"></i>&nbsp; {{userName}}</span>
    </a>
    <li class="nested" [class.expand]="showMenu === 'profile'">
```

```
                    <ul class="submenu">
                        <li>
                            <a href="javascript:void(0)">
                                <span><i class="fa fa-fw fa-user"></i> {{ 'Profile' | translate }}</span>
                            </a>
                        </li>
                        <li>
                            <a href="javascript:void(0)">
                                <span><i class="fa fa-fw fa-envelope"></i> {{ 'Inbox' | translate }}</span>
                            </a>
                        </li>
                        <li>
                            <a href="javascript:void(0)">
                                <span><i class="fa fa-fw fa-gear"></i> {{ 'Settings' | translate }}</span>
                            </a>
                        </li>
                        <li>
                            <a [routerLink]="['/login']" (click)="onLoggedout()">
                                <span><i class="fa fa-fw fa-power-off"></i> {{ 'Log Out' | translate }}</span>
                            </a>
                        </li>
                    </ul>
                </li>
            </div>
        </div>
    </div>
</nav>
```

Sidebar.component.scss

```
$topnav-background-color: #b12a31;
.sidebar {
```

```
border-radius: 0;

position: fixed;

z-index: 1000;

top: 6%;

// left: 235px;

// left: 0;

right: 0;

width: 255px;

margin-right: -255px;

border: none;

border-radius: 0;

overflow-y: auto;

background-color: $topnav-background-color;

bottom: 0;

overflow-x: hidden;

padding-bottom: 40px;

-webkit-transition: all 0.2s ease-in-out;

-moz-transition: all 0.2s ease-in-out;

-ms-transition: all 0.2s ease-in-out;

-o-transition: all 0.2s ease-in-out;

transition: all 0.2s ease-in-out;

// border-top: 1px solid rgba(255,255,255,0.3);

.list-group {

    a.list-group-item {

        background: $topnav-background-color;

        border: 1px solid #053c6e;

        border-radius: 0;

        color: #fff;

        font-size: .85rem;

        text-decoration: none;
```

```
            padding: 0.65rem 1.25rem;
            border-left: 0;
            border-right: 0;
            // border-bottom: 1px solid #053c6e;
            .fa {
                margin-right: 10px;
            }
        }
        // a:hover {
        //     background: darken($topnav-background-color, 5%);
        //     color: #fff;
        // }
        a.router-link-active {
            // background: darken($topnav-background-color, 5%);
            // color: #fff;
            font-weight: 800;
        }
        // .header-fields {
        //     padding-top: 10px;

        //     > .list-group-item:first-child {
        //         border-top: 1px solid rgba(255, 255, 255, 0.2);
        //     }
        // }
    }
   ::selection {
    text-shadow: none;
    background: #FE2E64;
    color: #fff;
}
```

```
.sidebar-dropdown {
    *:focus {
        border-radius: none;
        border: none;
    }
    .panel-title {
        font-size: 1rem;
        height: 50px;
        margin-bottom: 0;
        a {
            // color: #999;
            text-decoration: none;
            font-weight: 400;
            background: $topnav-background-color;
            span {
                position: relative;
                display: block;
                padding: 0.75rem 1.5rem;
                padding-top: 1rem;
            }
        }
        a:hover,
        a:focus {
            color: #fff;
            outline: none;
            outline-offset: -2px;
        }
    }
    .panel-title:hover {
        background: darken($topnav-background-color, 5%);
        }
        .panel-collapse {
            border-radius: 0;
            border: none;
            .panel-body {
                .list-group-item {
                    border-radius: 0;
                    background-color: $topnav-background-color;
                    border: 0 solid transparent;
                    a {
                        color: #999;
                    }
                    a:hover {
                        color: #fff;
                    }
                }
                .list-group-item:hover {
                    background: darken($topnav-background-color, 5%);
                }
            }
        }
    }
}
.nested-menu {
    .list-group-item {
        cursor: pointer;
    }
    .nested {
        list-style-type: none;
    }
    ul.submenu {
        display: none;
        height: 0;
    }
    & .expand {
        ul.submenu {
            display: block;
            list-style-type: none;
            height: auto;
            li {
                a {
                    color: #fff;
                    padding: 10px;
                    display: block;
                }
            }
        }
    }
}
// @media screen and (max-width: 992px) {
//     .sidebar {
//         top: 54px;
//         left: 0px;
//     }
// }
@media print {
    .sidebar {
        display: none !important;
    }
}
```

```
@media (min-width: 992px) {
    .header-fields {
        display: none;
    }
}

::-webkit-scrollbar {
    width: 8px;
}

::-webkit-scrollbar-track {
    -webkit-box-shadow: inset 0 0 0px rgba(255, 255, 255, 1);
    border-radius: 3px;
}

::-webkit-scrollbar-thumb {
    border-radius: 3px;
    -webkit-box-shadow: inset 0 0 3px rgba(255, 255, 255, 1);
}

.l-0px{
     left: 0px;
}
.l-235px{
    left: 235px;
}
.r-0px{
    right: 0px !important;
}
.r-235px{
    right: 255px !important;
}
```

Sidebar.component.ts

```
@Component({
    selector: 'app-sidebar',
    templateUrl: './sidebar.component.html',
    styleUrls: ['./sidebar.component.scss']
})
export class SidebarComponent {
    isService:boolean=false;
    isActive: boolean = false;
    isPayment: boolean = false;
    isDoc:boolean=false;
    isReq:boolean=false;
    isClaim:boolean=false;
    isName:boolean=false;
    isFrequency:boolean=false;
    isSubmit:boolean=false;
    showMenu: string = '';
    pushRightClass: string = 'push-left';
    isVisible:boolean = false;
    isAllTask:boolean = false;
    isAdmin:boolean=false;
    isDo:boolean=false;
    isCustomer:boolean=false;
    isDetails:boolean=false;
    isRequest:boolean=false;
    user:boolean=false;
    listViewForAllTeam:boolean=true;
    userName="";
    constructor(private translate: TranslateService, public router: Router) {
```

```
    this.translate.addLangs(['en', 'fr', 'ur', 'es', 'it', 'fa', 'de','mr']);

    this.translate.setDefaultLang('en');

    const browserLang = this.translate.getBrowserLang();

    this.translate.use(browserLang.match(/en|fr|ur|es|it|fa|de|mr/) ? browserLang : 'en');


    this.router.events.subscribe(val => {

    //this.toggleSidebar();




        // if (

        //     val instanceof NavigationEnd &&

        //     window.innerWidth <= 1200 &&

        //     this.isToggled()

        // ) {

        //     this.toggleSidebar();

        // }

    });

}


ngOnInit() {



var txt = JSON.parse(localStorage.getItem('usersinfo'));

var obj=(txt[0]["teamId"]);

this.userName=this.userName+" "+txt[0]["username"];



if(obj==6){

this.isVisible=true;

this.isAllTask=true;

this.listViewForAllTeam=false;

}else if(obj==1 || obj==2 || obj==5){

this.isAllTask=true;
this.isVisible=true;
this.listViewForAllTeam=false;
}else if(obj==8){
this.isAdmin=true;
this.isVisible=true;
this.listViewForAllTeam=false;
}else if(obj==11){
this.isCustomer=true;
this.isVisible=true;
this.listViewForAllTeam=false;
}
else if(obj==13){
this.isDetails=true;
this.isVisible=false;
this.listViewForAllTeam=false;
this.user=false;
this.isPayment=true;
this.isDoc=true;
this.user=false;
this.isRequest=true;
this.isFrequency=true;
this.isName=true;
this.isSubmit=true;
this.isReq=true;
}
else if(obj==14)
{
this.isService=true;
this.listViewForAllTeam=true;
}
```

```
else if(obj==15)

{

this.listViewForAllTeam=true;

this.isClaim=true;

this.isDo=true;

}

else if(obj==16)

{

this.listViewForAllTeam=false;

this.isRequest=true;

this.user=false;

}

else if(obj==17)

{

this.isSubmit=true;

this.listViewForAllTeam=false;

}

}


eventCalled() {

    this.isActive = !this.isActive;

}


addExpandClass(element: any) {

    if (element === this.showMenu) {

        this.showMenu = '0';

    } else {

        this.showMenu = element;

    }

}


isToggled(): boolean {
        const dom: Element = document.querySelector('body');
        return dom.classList.contains(this.pushRightClass);
    }

    toggleSidebar() {
        const dom: any = document.querySelector('body');
        dom.classList.toggle(this.pushRightClass);



        if(dom.classList.contains(this.pushRightClass)){
            document.getElementById("sideBarIsActive").classList.remove('r-0px')
            document.getElementById("sideBarIsActive").classList.add('r-235px');
            document.getElementById("mask").style.display = 'block';

// document.getElementsByTagName("app-sidebar")[0].setAttribute("style", "width:18.5% !important");
// document.getElementsByTagName("section")[0].setAttribute("style", "width:81.5% !important");

        }else{
        document.getElementById("sideBarIsActive").classList.remove('r-235px');
         document.getElementById("sideBarIsActive").classList.add('r-0px');
         document.getElementById("mask").style.display = 'none';
        //  document.getElementsByTagName("app-sidebar")[0].setAttribute("style", "width:0% !important");
        //  document.getElementsByTagName("section")[0].setAttribute("style", "width:100% !important");
         }

    }

    rltAndLtr() {
        const dom: any = document.querySelector('body');
        dom.classList.toggle('rtl');
    }

    changeLang(language: string) {
        this.translate.use(language);
    }

    onLoggedout() {
        localStorage.removeItem('isLoggedin');
    }
```

login.component.html

```
<div class="login-page h-100">
    <div class="container-fluid h-100">
        <div class="row m-0 h-100">
            <div class="col-12 col-sm-7 h-100 left_div">
                <div class="row justify-content-md-center">
                    <div class="col-12 pt-4">
<img width="110px" class="ml-3" src="login.png" />


                    </div>
                </div>
            </div>
            <div class="col-12 col-sm-5 h-100 right_div p-5">
                <div class="row m-0 welcomeDiv">
                    <div class="col-12 text-white">Welcome</div>
                    <div class="col-12 text-white pt-2">Login to access your account</div>
                </div>
                <form class="pl-3 pr-3 pt-5" role="form" (submit)="onLoggedin($event)" method="POST"
name="login">
                    <div class="form-content">
                        <div class="form-group">
                            <label class="col-12 text-white p-0 login-label">Username</label>
                            <input type="text" ng-model="name"
                                class="form-control input-lg"
                        placeholder="Username" id="username"
                        name="username" autocomplete="off">
                    </div>
                    <div class="form-group pt-4">
                      <label class="col-12 text-white p-0 login-label">Password</label>
                      <input type="password"
                        class="form-control input-lg"
                        placeholder="Password" id="password"
                        name="password">
                    </div>
                    <div class="row text-center pt-4">
                      <div class="col-sm-12">
                        <button type="submit" class="btn rounded-btn signInBtn">Sign In</button>
                       </div>
                    </div>
                  </div>
                </form>
              </div>
            </div>
          </div>
        </div>
```

login.component.scss

```
::selection {
  text-shadow: none;
  background: #FE2E64;
  color: #fff;
}
$topnav-background-color: #222;
:host {
  display: block;
```

```
}
.login-page {

  .col-lg-4 {
    padding: 0;
  }
  .left_div{
    background: url('logo.png');
    background-repeat: no-repeat;
    background-position-x: center;
    background-position-y: 80%;
    background-size: 73%;
  }
  .right_div{
    background: #E9202E;
  }
  .input-lg {
    height: 46px;
  }
  .form-control.input-lg{
    color: #333;
  }
  .form-control.input-lg:focus{
    box-shadow: none;
    border-color: transparent;
  }
  .form-group {
    padding: 0px 0;
    margin-bottom: 0px;
    input::-webkit-input-placeholder {
        color: #95989a;
    }
    input:-moz-placeholder {
        /* Firefox 18- */
        color: #95989a !important;
    }

    input::-moz-placeholder {
        /* Firefox 19+ */
        color: #95989a !important;
    }

    input:-ms-input-placeholder {
        color: #95989a !important;
    }
}
.form-content {
    padding: 20px 0;
}
.user-avatar {
    -webkit-border-radius: 50%;
    border-radius: 50%;
    border: 2px solid #fff;
}
.loginform{
    background-color: #fff;
}
.logoClass{
    margin-right: 15px;
}
.formcontainer{
        top: 35%;
        position: fixed;
    }
    .links{
        color: #000!important;
        font-size: 12px;
    }
    .logoH1{
        vertical-align: middle;
        display: inline;
    }
}

body{
    font-family: 'Open Sans', sans-serif;
}

.signInBtn{
    background: #12D4CD;
    color: #fff;
    width: 100%;
    height: 46px;
    font-weight: 600;
}

.welcomeDiv{
    padding-top: 25%;

    div:first-child{
        font-size: 2.5rem;
    }
    div:last-child{
        font-weight: 600;
    }
}

.login-label{
    font-size: .9rem;
    font-weight: 600;
}
```

login.component.ts

```
@Component({
    selector: 'app-login',
    templateUrl: './login.component.html',
    styleUrls: ['./login.component.scss'],
    animations: [routerTransition()],
    encapsulation: ViewEncapsulation.None,


})
export class LoginComponent implements OnInit {


  userId=0;







    user;
    searchResult: any[];
constructor(private Auth: DataService,
   public router: Router,
   private userIdle: UserIdleService, private modalService: NgbModal,
   private commonFunction: CommonFunction) { }
    ngOnInit() {
   //Start watching for user inactivity.
    this.userIdle.startWatching();

    // Start watching when user idle is starting.
    this.userIdle.onTimerStart().subscribe(count => console.log(count));

    // Start watch when time is up.
    this.userIdle.onTimeout().subscribe(() => {console.log('Time is up!')
    //localStorage.removeItem('isLoggedin');
    //this.router.navigate(['/login'])
    });
if (localStorage.getItem('isLoggedin')){
this.router.navigate(['/dashboard'])
}
    }
    stop() {
    this.userIdle.stopTimer();
  }
  stopWatching() {
    this.userIdle.stopWatching();
  }
  startWatching() {
    this.userIdle.startWatching();
  }
  restart() {
    this.userIdle.resetTimer();
  }
    onLoggedin(event) {
```

```
    event.preventDefault()
            console.log(event)
            const target = event.target
            const username = target.querySelector('#username').value
            const password = target.querySelector('#password').value
            this.Auth.getUserDetails(username, password).subscribe(data => {
    if(data.body['status']==300)
    {
    this.commonFunction.openDialog(data.body['data'],'');
    localStorage.removeItem('isLoggedin')
    }else{
            console.log(data,"data from server");
    var dataObj=data.body['data_user'];
    this.searchResult=data.body['data_user'];
    if(dataObj.length) {
    console.log("user name ---->"+this.searchResult);
    this.userId=dataObj[0]['_id'];
    this.user=dataObj[0]['name'];
    this.Auth.setLoggedIn(true);
    this.Auth.setUserDetails(data.body['data_user']);
      this.router.navigate(['/dashboard'])
    } else {
      localStorage.removeItem('isLoggedin')
      //window.alert("Wrong Credentials")
      this.commonFunction.openDialog('Wrong Credentials','');
    }
    }
  })
    }
closeResult: string;
```

```
   forgotPasswordPopup(content)
{
      this.modalService.open(content,
        { centered: true, backdrop: 'static', keyboard: false, backdropClass: 'dark-backdrop' })
        .result.then((result) => {
          this.closeResult = `Closed with: ${result}`;
          console.log("closed result"+this.closeResult);
      }, (reason) => {
          this.closeResult = `Dismissed ${this.getDismissReason(reason)}`;
      });
}
 private getDismissReason(reason: any): string
 {
      if (reason === ModalDismissReasons.ESC) {
      return 'by pressing ESC';
      } else if (reason === ModalDismissReasons.BACKDROP_CLICK) {
          return 'by clicking on a backdrop';
      } else {
          return  `with: ${reason}`;
      }
   }
   emailFormControl = new FormControl('', [
   Validators.required,
   Validators.email,
  ]);
  matcher = new MyErrorStateMatcher();
updatePassword(){
  var body={email:this.emailFormControl.value};
   this.Auth.updatePasswordInDB(body).subscribe(res => {
   console.log("emailFormControl---------->"+this.emailFormControl.value);
```

```
      this.commonFunction.openDialog(res['data'],'');
  });
}
}
export class MyErrorStateMatcher implements ErrorStateMatcher {
  isErrorState(control: FormControl | null, form: FormGroupDirective | NgForm | null): boolean {
    const isSubmitted = form && form.submitted;
    return !!(control && control.invalid && (control.dirty || control.touched || isSubmitted));
  }
}
```

claim.component.html

```
<div class="container-fluid h-100" ><!-- start of container -->
  <form class="max-height col-12 p-0" role="form" [formGroup]="Details1" (submit)='addDetails()'
method="POST" #myForm="ngForm" autocomplete="off" novalidate>
              <div class="row m-0 h-90"><!-- start of row -->
                            <div class="col-sm-2 col-md-2 col-lg-2 col-xl-2 col-2 col-lg-offset-2 center-block h-100 ">
      <br><br>
                                          <mat-form-field class="example-full-width">
            <input matInput [matDatepicker]="picker" placeholder="From Date" formControlName="fromdate"
name="fromdate" disabled readonly="readonly" style="width: 5em">
            <mat-datepicker-toggle matSuffix [for]="picker"></mat-datepicker-toggle>
            <mat-datepicker #picker disabled="false" touchUi></mat-datepicker>
          </mat-form-field>
          <br><br><br><br><br><br>
            <mat-form-field class="example-full-width">
            <input matInput [matDatepicker]="picker2" placeholder="To Date" formControlName="todate"
name="todate" disabled readonly="readonly" style="width: 5em">
```

```
            <mat-datepicker-toggle matSuffix [for]="picker2"></mat-datepicker-toggle>

            <mat-datepicker #picker2  touchUi color="secondary" disabled="false"></mat-datepicker>

          </mat-form-field>

          <br><br><br><br><br><br>

          <div class="input-group form-group input-group-sm mb-3 h-100">

             <div class="input-group-prepend h-100" >

<span                                   class="input-group-text"                                   id="inputGroup-sizing-sm"
[ngStyle]="{'background':'white','height':'3.5em','width':'4.5em'}">

               <mat-card [ngStyle]="{'height':'2.5em','width':'3em','padding':'0.5em'}">Type</mat-card>


              </span>

              <mat-form-field class="b">

              <div class="h-100">

             <mat-select  #matSelect  [panelClass]="a" (selectionChange)="display()"  formControlName="claim">

             <mat-option value="Accepted Claim">Accepted Claim</mat-option>

             <mat-option value="Rejected Claim">Rejected Claim</mat-option>


            </mat-select>

          </div>

            </mat-form-field>


             </div>

          </div>



                          </div>

      <div class="col-sm-1 col-md-1 col-lg-1 col-xl-1 col-1 col-lg-offset-1  h-100 ">

        <mat-divider [vertical]="true"></mat-divider>

      </div>

                          <div class="col-sm-9 col-md-9 col-lg-9 col-xl-9 col-9  h-100" *ngIf="A">

             <div class="mat-elevation-z8  h-100 table-striped ">

      <div id="contentToConvert"  class="scroll h-100">
```

```
 <h3>{{Title}}</h3>
<table mat-table [dataSource]="dataSource" class="example-container" matSort>

 <!-- ID Column -->

<ng-container matColumnDef="id" >
  <th mat-header-cell *matHeaderCellDef mat-sort-header> No. </th>
  <td mat-cell *matCellDef="let row; let i = index;"> {{i+1}} </td>
 </ng-container>



 <ng-container matColumnDef="AcceptedDate" >
  <th mat-header-cell *matHeaderCellDef mat-sort-header> Accepted Date </th>
  <td mat-cell *matCellDef="let row"> {{row.AcceptedDate}} </td>
 </ng-container>

 <!-- Progress Column -->
 <ng-container matColumnDef="appNo">
  <th mat-header-cell *matHeaderCellDef mat-sort-header> App No </th>
  <td mat-cell *matCellDef="let row"> {{row.appNo}} </td>
 </ng-container>

 <!-- Name Column -->
 <ng-container matColumnDef="PolicyNo">
  <th mat-header-cell *matHeaderCellDef mat-sort-header> Policy No </th>
  <td mat-cell *matCellDef="let row"> {{row.PolicyNo}} </td>
 </ng-container>

 <ng-container matColumnDef="name">
  <th mat-header-cell *matHeaderCellDef mat-sort-header> Person's Name</th>
  <td mat-cell *matCellDef="let row" > {{row.name}} </td>
 </ng-container>
```

```
        <!-- Color Column -->
        <ng-container matColumnDef="ApprovedAmt">
          <th mat-header-cell *matHeaderCellDef mat-sort-header> Approved Amt.</th>
          <td mat-cell *matCellDef="let row" > {{row.AcceptedSum}} </td>
        </ng-container>
        <tr mat-header-row *matHeaderRowDef="displayedColumns; sticky: true"></tr>
        <tr mat-row *matRowDef="let row; columns: displayedColumns; let i = index; " class="sc">
        </tr>
      </table>
      </div>
      <mat-paginator [pageSizeOptions]="[5, 10, 25, 100]" class="mat-paginator-sticky"></mat-paginator>
    </div>

                    </div>


             </div><!-- end of row -->
 </form>
<div class="row m-0 h-10" *ngIf="A" >
  <div class="col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12 p-0 h-100">

  </div>
  <button type="button" id="float" (click)="convertHTMLToPDf()" class="float-right" style="position: absolute;
right: 2em; bottom: 3em;"><i class="fa fa-download"></i></button>
</div>
</div><!-- end of container →
```

claim.component.css

```
mat-select{
            width: 100%;
}
.example-full-width {
    display: block;

    position: relative;

    flex: auto;

    min-width: 0;

    width: 12em !important;

}

.b{

    width: 8em;

}

.example-container {


  overflow: auto;

}


th.mat-cell{

  position: sticky;

}


div.mat-elevation-z8{

    box-shadow: none!important;

}

button{

  height: 6%

}

#float{

              float: right;

}

::selection {

    text-shadow: none;

    background: #FE2E64;

    color: #fff;

}
```

```
td.mat-cell:first-of-type, td.mat-footer-cell:first-of-type, th.mat-header-cell:first-of-type {
    padding-left: 112px;
}
.vl {
  border-left: 0.11em solid black;
  height: 100%;
}
mat-divider{
    height: 38em;
    width: 0.11em;
    background-color: lightslategrey;
}
```

claim.component.ts

```
export interface UserData {
  id: string;
  name: string;
  progress: string;
  color: string;
}

@Component({
  selector: 'app-claim',
  templateUrl: './claim.component.html',
  styleUrls: ['./claim.component.css']
})
export class ClaimComponent implements OnInit {
subscription: Subscription;
displayedColumns: string[] = ['id','AcceptedDate', 'appNo', 'PolicyNo', 'name',
'ApprovedAmt'/*,'NomineeDateOfBirth',
'NomineeGender','NomineeHouseNo','NomineeBuildingName','NomineeLocality', 'NomineeStreet',
'NomineeLandmark',                    'NomineeCity',                    'NomineeState',                    'NomineeCountry',
'NomineePinCode','NomineeMobileNo','NomineeEmail','NomineeBank',     'NomineeBranch',     'NomineeIFSC',
'NomineeAccountNo', 'NomineeAccountType'*/];
  dataSource: MatTableDataSource<UserData>;

  @ViewChild(MatPaginator) paginator: MatPaginator;
  @ViewChild(MatSort) sort: MatSort;
claim:string;
fromdate:string;
todate:string;
A:boolean=false;
userId: string;
teamId: string;
name: string;
appNo:string="";
public taskListStatusValue:number=0;
 public isProgressValue: number= 0
 public callReason:string;
public progressStatus: string="";
 tasklist:Array<any>;
  disp:Array<any>;
   roleId:string;
   count=0;
   public Title:string;
Details1 = new FormGroup({
    fromdate: new FormControl({value:''},Validators.required),
    todate: new FormControl({value:''},Validators.required),
    claim:new FormControl({value:''},Validators.required)
  });
  constructor(public router: Router, private route: ActivatedRoute,private fb: FormBuilder, public
commonFunction:CommonFunction, private dialog: MatDialog , private sharedDataService: DataService, private
```

```
http: HttpClient,@Optional() @Inject('config') public moduleConfig: ImageViewerConfig,private modalService:
NgbModal,private datePipe: DatePipe,public claimService:ClaimService) {
  this.subscription =claimService.subj$.subscribe(val=>{
    console.log(val);


     this.loadData(val);
     });
      this.route.queryParams.subscribe(params => {


this.isProgressValue=Number(params["inprogress"]);
this.callReason=params["callreason"];


         if(params["inprogress"]=="AllRecord"){
            this.progressStatus="AllRecord";
            this.ngOnInit();
         }else{
            this.progressStatus="";
            this.ngOnInit();
         }



        });
  }


  ngOnInit() {

    var txt = JSON.parse(localStorage.getItem('usersinfo'));
  this.Details1 = this.fb.group
({
    fromdate:[' ',Validators.required],
    todate:[' ',Validators.required],
    claim:[' ',Validators.required]
});
this.userId=txt[0]["_id"];
this.name=txt[0]["name"];
 this.teamId=(txt[0]["teamId"]);
 this.roleId=(txt[0]["roleId"]);
  var body={userId:this.userId, teamId:this.teamId,
inprogress:[this.isProgressValue],claim:this.claim,fromdate:this.fromdate,todate:this.todate};
  }
  /**display function is to display the table which holds the accepted or rejected claims**/
  display()
  {
    this.claim=this.Details1.get('claim').value;
  this.fromdate=this.Details1.get('fromdate').value;
  this.todate=this.Details1.get('todate').value;
  var body={userId:this.userId, teamId:this.teamId,
inprogress:[this.isProgressValue],claim:this.claim,fromdate:this.fromdate,todate:this.todate};
  this.loadData(body);
  }
  /**loadData() function is used to load data into the table**/
  loadData(body)
  {

   if(this.claim=="Accepted Claim")
   {
   this.Title="Accepted Claims";
   /**getDetailsFromDB connects the ts file to the service file which holds the function to retrieve the data from
the DB**/
     this.claimService.getDetailsFromDB(body)
        .subscribe(res => {
     this.A=true;
```

```
        var dataObj= JSON.stringify(res['data'])
      var obj=JSON.parse(dataObj);

      for (var sub in obj )
      {
       var objSub = obj[sub];
       var elementsArray = Object.keys(objSub);


        console.log("tasklist response---->"+obj[sub]['inprogress']);
        if(Number(this.teamId)==5 || Number(this.teamId)==4){

        }else{
        if(obj[sub]['inprogress']==1){
        obj[sub]['inprogress']="Work In Progress"
        }else if(obj[sub]['inprogress']==0){
        obj[sub]['inprogress']="New Task"
        }else if(obj[sub]['inprogress']==2){
        obj[sub]['inprogress']="Rescaned Cases"
        }
        }

      }

      this.tasklist=obj;
      const users = Array.from(this.tasklist);



    this.dataSource = new MatTableDataSource(users);
    this.dataSource.paginator = this.paginator;
    this.dataSource.sort = this.sort;
  });
```

```
   }
   else if(this.claim=="Rejected Claim")
   {
   this.Title="Rejected Claims";
   /**getDetailFromDB connects the ts file to the service file which holds the function to retrieve the data from
the DB**/
    this.claimService.getDetailFromDB(body)
        .subscribe(res => {
     this.A=true;
        var dataObj= JSON.stringify(res['data'])
      var obj=JSON.parse(dataObj);
      for (var sub in obj )
      {
       var objSub = obj[sub];
       var elementsArray = Object.keys(objSub);


        console.log("tasklist response---->"+obj[sub]['inprogress']);
        if(Number(this.teamId)==5 || Number(this.teamId)==4){

        }else{
        if(obj[sub]['inprogress']==1){
        obj[sub]['inprogress']="Work In Progress"
        }else if(obj[sub]['inprogress']==0){
        obj[sub]['inprogress']="New Task"
        }else if(obj[sub]['inprogress']==2){
        obj[sub]['inprogress']="Rescaned Cases"
        }
        }
      }
      this.tasklist=obj;
      const users = Array.from(this.tasklist);
   this.dataSource = new MatTableDataSource(users);
   this.dataSource.paginator = this.paginator;
   this.dataSource.sort = this.sort;
});
  }
 }
 /*download()
 {
 var body={}
 this.claimService.createpdfFromDB(body).subscribe(res => {
 console.log("File downloaded");
 });
 }*/
/* this function converts a part of the html to a pdf and downloads it to a particular directory*/
   convertHTMLToPDf()
 {
     this.count++;
     console.log("in function to convert html to pdf..");

    let htmlContent = document.getElementById('contentToConvert').innerHTML;

     var body={appNo:this.appNo,htmlContent:htmlContent,count:this.count,name:this.name};

      this.claimService.convertHtmlToPdf(body).subscribe(res => {
             var filename=res["data"];
             this.commonFunction.openDialog("File has been converted", "");


              console.log(filename);
              var bod={filename:filename,name:this.name};
              this.claimService.downloads(bod).subscribe(res =>
              {
                          Console
.log("response---->"+res);
                          this.count++;
                         var a="claimreport";
                         var c=this.count.toString();
                         var b=a.concat(c);
                          var filename=b.concat(".pdf");
                          saveAs(res,filename);

                         //this.spinner.hide();
               });
      },
      error=>
      {
        console.log(error);
      });
  }
}
```

claim.service.ts

```
@Injectable({
  providedIn: 'root'
})
export class ClaimService {
private sub = new Subject();
  subj$ = this.sub.asObservable();
result:any;
  constructor(private _http:Http,private http:HttpClient) { }
   getDetailsFromDB(inputValue)
   {

      return this.http.post("/api/getClaimDesFromDB", { /**calls the getClaimDesFromDB function in api.js**/
      inputValue
      })
    }
     getDetailFromDB(inputValue)
   {


      return this.http.post("/api/getClaimDetsFromDB", {  /**calls the getClaimDetsFromDB function in api.js**/
      inputValue
      })
    }
     convertHtmlToPdf(inputValue) /*calls the function in html_pdf_route.js */
   {
      console.log("in service function to generate pdf...");
      return this.http.post("/htmlPdf/createhtmlpdf", {
      inputValue
      })
    }
      downloads(inputValue)
   {
    return this.http.get('/api/downloadCFile/', {
        params:{filename:inputValue['filename'],name:inputValue['name']},responseType:"blob"
       //  search: // query string if have
       })
    }
}
```

claimsubmition.component.html

```
 <div class="container-fluid h-48 w-100">
 <div class="row m-0">
    <div class="col-4 small-font mt-2 p-0">
      <h3>Claim Submitions</h3>
      &nbsp;&nbsp;&nbsp;
    </div>
    <div class="col-4 p-0 large-font text-center">
    </div>
    <div class="col-4 p-0">
    </div>
  </div>
  <hr class="mb-0 border-top-1-5" />
<div class="container-fluid h-100">
              <form class="max-height col-12 p-0" role="form" [formGroup]="Details" (submit)='addDetails()'
method="POST" #myForm="ngForm" autocomplete="off" novalidate>
  <div class="row m-0">


              <div class="col-sm-4 col-md-4 col-lg-4 col-xl-4 col-4">
              </div>
               <div class="col-sm-4 col-md-4 col-lg-4 col-xl-4 col-4">
                            <!--  <div class="input-group form-group input-group-sm mb-3"> -->
             <!-- <div class="input-group-prepend" > -->
             <!--   <span class="input-group-text" id="inputGroup-sizing-sm" [ngStyle]="{'background':'white'}">
                            <label>Select a document</label>


              </span> -->
              <mat-form-field >
              <div>
             <mat-select #matSelect placeholder="Select a document" [panelClass]="a"
formControlName="claim">
              <mat-option value="AadharCard">Aadhar Card</mat-option>
              <mat-option value="VotersID">Voter's ID</mat-option>
                        <mat-option value="DrivingLicense">Driving License</mat-option>
                        <mat-option value="PanCard">Pan Card</mat-option>
```

```
                    <mat-option value="MarriageCertificate">Marriage Certificate</mat-option>
                    <mat-option value="DeathCertificate">Death Certificate</mat-option>
                    <mat-option value="MedicalReport">Medical Report</mat-option>
                    <mat-option value="CancelledCheque">Cancelled Cheque</mat-option>
                    <mat-option value="FilledClaimForm">Filled Claim Form</mat-option>
            </mat-select>
          </div>
           </mat-form-field>
            <!--  </div> -->
         <!--  </div> -->
              </div>
                       <div class="col-sm-4 col-md-4 col-lg-4 col-xl-4 col-4">
                       </div>
                      </div>
                        <div class="row m-0">
                              <div class="col-sm-4 col-md-4 col-lg-4 col-xl-4 col-4">
                       </div>
               <div class="col-sm-4 col-md-4 col-lg-4 col-xl-4 col-4">

                       <input #file id="input" type="file" placeholder="PDF src" name={{this.appNo}}
ng2FileSelect [uploader]="uploader" (change)="onSelectFile($event,file)" id="file" download/><label
for="file" *ngIf="label" >{{lab}}</label>
               </div>
               <div class="col-sm-4 col-md-4 col-lg-4 col-xl-4 col-4">
                       </div>
              </div>
                <div class="row m-0">
                              <div class="col-sm-4 col-md-4 col-lg-4 col-xl-4 col-4">
                       </div>
                              <div class="col-sm-4 col-md-4 col-lg-4 col-xl-4 col-4">
```

```
                                        <button type="button" mat-stroked-button class="custom-btn btn-border"
(click)="addDetails();uploader.uploadAll();load();" [disabled]="!Details.valid" *ngIf="btn2" style="margin-
left: 15px;">Save</button>
                    </div>
                              <div class="col-sm-4 col-md-4 col-lg-4 col-xl-4 col-4">
                    </div>


    </div>
          </form>
  </div>
</div>
<br><br>
<div class="container-fluid h-50 w-100"  *ngIf="table">
<div class="row m-0 h-100 w-100">
<div class="h-100 w-100 col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12 text-center">
<div class="row m-0 w-100 h-80">
 <div class="w-100 col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12 p-0 h-100">
<div class="mat-elevation-z8 example-container h-100 table-striped ">
<div class="w-100 scroll h-100">
<table mat-table [dataSource]="dataSource" id="mytable" matSort>
                                                    <!-- ID Column -->
<ng-container matColumnDef="id">
<th mat-header-cell *matHeaderCellDef mat-sort-header> No. </th>
<td mat-cell *matCellDef="let row; let i = index;"> {{i+1}} </td>
                                                    </ng-container>
 <ng-container matColumnDef="NomineeDateOfBirth">
<th mat-header-cell *matHeaderCellDef mat-sort-header>UserId</th>
 <td mat-cell *matCellDef="let row" > {{row.UserId}} </td>
 </ng-container>
<ng-container matColumnDef="appNo">
```

```
<th mat-header-cell *matHeaderCellDef mat-sort-header> App No </th>
 <td mat-cell *matCellDef="let row"> {{row.appNo}} </td>
 </ng-container>
<ng-container matColumnDef="creationDate" >
<th mat-header-cell *matHeaderCellDef mat-sort-header>ImageType</th>
<td mat-cell *matCellDef="let row"> {{row.imgType}} </td>
</ng-container>
<!-- Progress Column -->
 <ng-container matColumnDef="NomineeName">
<th mat-header-cell *matHeaderCellDef mat-sort-header> name</th>
  <td mat-cell *matCellDef="let row" > {{row.name}} </td>
 </ng-container>
   <!-- Name Column -->
  <ng-container matColumnDef="PolicyNo">
<th mat-header-cell *matHeaderCellDef mat-sort-header style="display: none;">filename</th>
 <td mat-cell *matCellDef="let row" style="display: none;"> {{row.imageName}} </td>
 </ng-container>
<!-- Color Column -->
  <ng-container matColumnDef="DateOfBirth" >
<th mat-header-cell *matHeaderCellDef mat-sort-header>Image View</th>
  <td mat-cell *matCellDef="let row" > <button type="button" class="btn" [ngClass]="{'active':
selectedItemSecondList == newImg, 'highlight':secondListItem == newImg}"
(click)="commonFunction.open(content);editData(row);" ><i class="fa fa-image"></i></button>
                                        </td>
                                            </ng-container>


 <tr mat-header-row *matHeaderRowDef="displayedColumns; sticky: true"></tr>
<tr mat-row *matRowDef="let row; columns: displayedColumns; let i = index; " > </tr>
   </table>
   </div>
```

```
<mat-paginator [pageSizeOptions]="[5,10,25,50,75,100]" class="mat-paginator-sticky"></mat-paginator>
                                                                </div>
                                                              </div>
                                                            </div>

                                                  </div>
                                                  </div>
                                      </div>
        <div class="container-fluid h-2 "  *ngIf="table" >
                                                  <div class="row m-0 " style="float: right;">
              <button type="button" mat-stroked-button class="custom-btn btn-border" [disabled]="disable"
(click)="onDownload()" style="margin-left: 15px;height: 2em"><i class="fa fa-download"></i> Claim
Form</button>
                                                  </div>
                                      </div>
<ng-template #content let-c="close" let-d="dismiss" class="scale50"  style="position:relative">
  <div class="modal-header border-0 " id="mydiv" >

    <div class="d-inline-flex pr-2 small-vertical-line-right" >
     <div >
         <button mat-button class="zoomIn-btn p-0" (click)="zoomIn();"></button>
          </div>
          <div class="d-inline-flex pr-2 small-vertical-line-right">
            <button mat-button class="zoomOut-btn p-0" (click)="zoomOut();"></button>
          </div>
        </div>
          <img src={{imagetoShow}} [ngStyle]="style" id="imgViewer" name="imgViewer"
        (dragstart)="onDragStart($event)" class="img-viewer-size" annotorious-annotate />


    <button type="button" class="close " aria-label="Close" (click)="d('Cross click')" >
      <span aria-hidden="true">&times;</span>
      </button>
  </div>
</ng-template>
```

claimsubmition.component.css

```
mat-select {
    width: fit-content !important;
    min-width: 100%;
   /*background-color: white;*/
}

::ng-deep .mat-form-field-infix {
    min-width: 10px !important;
    width: fit-content !important;
     /*background-color: white;*/
}

::ng-deep mat-select .mat-select-value {
    width: fit-content;
    min-width: 2ch;
    max-width: 25ch;
    /* background-color: white;*/
}
::ng-deep mat-select .mat-select-placeholder {
    width: fit-content;
    min-width: 21ch;
    /*background-color: white;*/
}
/* Style buttons */
.btn {
  background-color: DodgerBlue; /* Blue background */
  border: none; /* Remove borders */
  color: white; /* White text */
  padding: 12px 16px; /* Some padding */
  font-size: 16px; /* Set a font size */
  cursor: pointer; /* Mouse pointer on hover */
}

/* Darker background on mouse-over */
.btn:hover {
  background-color: RoyalBlue;
}

.scale50 {
    -webkit-transform: scale(3.0);
    -moz-transform: scale(3.0);
    transform: scale(3.0);
}

.modal-xl .modal-lg {
  max-width: 100%;
}

.modal-header{
  overflow-x: scroll !important;
  overflow-y: scroll !important;
}
input[type=file]{
    width:100px;
    color:transparent;
}
.scroll{
  overflow: auto;
}
```

```
.imgViewer{
  width: 100%;
}
div.mat-form-field-infix {
    display: block;
    position: relative;
    flex: auto;
    min-width: 0;
    width: 15em;
}
/*div.example-container {
  // height: 400px;
  overflow-y: auto;
}*/

/*button{
  height: 50%
}*/
tr{
  text-align: left;
}
th{
  height: 1em !important;
  }
 th.mat-header-cell {
    padding-left: 2em !important;
    text-align: left;
    height: 1em !important;
}
tr.mat-header-row{
  height: 2em !important;
}
td.mat-cell{
     padding-left: 2em !important;
  text-align: left;
}
div.mat-elevation-z8{
  box-shadow: none!important;
}
 /*th.mat-header-cell {
    padding-left: 7em !important;
}*/

::selection {
    text-shadow: none;
    background: #FE2E64;
    color: #fff;
}
```

claimsubmition.component.ts

```
export interface UserData {
  id: string;
  name: string;
  progress: string;
  color: string;
}

@Component({
  selector: 'app-claimsubmition',
  templateUrl: './claimsubmition.component.html',
  styleUrls: ['./claimsubmition.component.css']
})
export class ClaimsubmitionComponent implements OnInit {
```

```
closeResult: string;
@Input()
  config: ImageViewerConfig;


 @Output()
  configChange: EventEmitter<ImageViewerConfig> = new EventEmitter();
  subscription: Subscription;
displayedColumns: string[] = ['id','NomineeDateOfBirth','creationDate', 'appNo', 'PolicyNo', 'NomineeName', 'DateOfBirth', /* 'NomineeGender','NomineeHouseNo','NomineeBuildingName','NomineeLocality' ,'NomineeStreet', 'NomineeLandmark', 'NomineeCity', 'NomineeState', 'NomineeCountry', 'NomineePinCode','NomineeMobileNo','NomineeEmail','NomineeBank', 'NomineeBranch', 'NomineeIFSC', 'NomineeAccountNo', 'NomineeAccountType'*/];
  dataSource: MatTableDataSource<UserData>;
  @ViewChild(MatPaginator) paginator: MatPaginator;
  @ViewChild(MatSort) sort: MatSort;
  lab:string;
  //isAddFlag="add";
  isUpdateFlag=false;
  show=false;
  public loading = false;
  public select="Aadhar Card";
//public urlPDF:string="https://vadimdez.github.io/ng2-pdf-viewer/assets/pdf-test.pdf";
imgURL: any;
public items=[];
userId: string;
name:string;
teamId: string;
appNo:string="";
public taskListStatusValue:number=0;
 public isProgressValue: number= 0
 public callReason:string;
public progressStatus: string="";
table:boolean=false;
 tasklist:Array<any>;
  disp:Array<any>;
   roleId:string;
   agentCode:String;
   dateCreated:string;
    isprogress:string;
    Details = new FormGroup({
    claim:new FormControl({value:''},Validators.required)
  });
   btn1: boolean=false;
 label:boolean=true;
 btn2:boolean=true;
  btn3:boolean=false;
   private translateX = 0;
  private translateY = 0;
  rotation:number=0;
  private scale = 1;
  originalSize: boolean = true;
saveBtn:boolean=false;
exportBtn:boolean=true;
  imageChangedEvent: any = '';
  croppedImage: any = '';
  showCropper = false;
    fileList:Array<any>;
    isCommentUpdate=false;
  title: string;
  msg: string;
  reqType:string;
  imgType:string;
  exportedImg:boolean=false;
  Detailid:string="";
  isDisabled:boolean=true;
  public policyNo: string;
  disable:boolean=true;
  nomId=0;
  images = [

  ];

  imagesNew = [


  ];

  uploadedImages = [


  ];
  navbarOpen = false;

  toggleNavbar() {
    this.navbarOpen = !this.navbarOpen;
  }
  public filename:string;
  public personalDataid:string;
  constructor(public router: Router, private route: ActivatedRoute,private fb: FormBuilder, public
commonFunction:CommonFunction,private dialog: MatDialog , private sharedDataService: DataService,private
http: HttpClient,@Optional() @Inject('config') public moduleConfig: ImageViewerConfig,private modalService:
```

```
NgbModal,private  datePipe:  DatePipe,public  claimsubmitionService:ClaimsubmitionService,private  spinner:
NgxSpinnerService) {
  this.subscription =  claimsubmitionService.subj$.subscribe(val=>{
    console.log(val);

     //this.loadData(val);
     });
     this.route.queryParams.subscribe(params => {

this.isProgressValue=Number(params["inprogress"]);
this.callReason=params["callreason"];

         if(params["inprogress"]=="AllRecord"){
            this.progressStatus="AllRecord";
            this.ngOnInit();
         }else{
            this.progressStatus="";
            this.ngOnInit();
         }



        });
         }
  public uploader: FileUploader = new FileUploader({url:"/api/SubUpload", additionalParameter: {
      comments: 'sdfsfsdfsdfsdfsdf'
   },itemAlias: 'photo'



      });
  ngOnInit() {
  this.table=false;
```

```
        var txt = JSON.parse(localStorage.getItem('usersinfo'));
  this.Details = this.fb.group
({
    claim:[' ',Validators.required]
});
this.userId=txt[0]["_id"];
this.lab="no file Added";
this.label=true;
 this.teamId=(txt[0]["teamId"]);
 this.roleId=(txt[0]["roleId"]);
 this.name=(txt[0]["name"]);
 console.log("name->"+this.name);

  /*gets application number from the DB*/
      var body={"userId":this.userId};
 this.claimsubmitionService.getAppFromDB(body).subscribe(res =>{
    var dataObj=JSON.stringify(res["data"]);
    var obj=JSON.parse(dataObj);
    var len=(obj.length-1);
     console.log("len---->"+len);
    if(obj!="" && obj!=[] && obj!=null){
    console.log("object valu--->"+obj[0]["appNo"])
      this.appNo=obj[0]["appNo"];
      var                                   body1={userId:this.userId,                                   teamId:this.teamId,
inprogress:[this.isProgressValue],appNo:this.appNo,imgType:this.imgType,name:this.name};
        body1={userId:this.userId,                                                                       teamId:this.teamId,
inprogress:[0,1,2],appNo:this.appNo,imgType:this.imgType,name:this.name};
        this.loadData(body1);
      console.log("App No--->"+this.appNo);
  this.claimsubmitionService.getNomIdFromDB(body).subscribe(res =>{  /*gets Nominee Id from the DB*/
    var dataObj=JSON.stringify(res["data"])
    var obj=JSON.parse(dataObj);
    var len=(obj.length-1);
     console.log("len---->"+len);
    if(obj!="" && obj!=[] && obj!=null){
    console.log("object valu--->"+obj[0]["appNo"])
      this.nomId=obj[0]["_id"];
      console.log("Nomine ID--->"+this.nomId);
        var a={appNo:this.appNo,nomId:this.nomId};

    this.claimsubmitionService.getNewRequest(a).subscribe(res => { /*gets personalData Id from the DB*/
      var perDataObj= JSON.stringify(res["data"])
      console.log("perDataObj" +perDataObj);


      var obj=JSON.parse(perDataObj);


      if(obj!="" && obj!=[] && obj!=null)
      {
          this.personalDataid = obj[0]["_id"];
          console.log("this.personalDataid-------->"+this.personalDataid);
          console.log("object = "+JSON.stringify(obj[0]));

      }},error=>
    {


      console.log(error);
    });
    var body={appNo:this.appNo};


  this.claimsubmitionService.getFileListFromDirectory(body).subscribe(res =>   /*gets all the files names from
the Directory*/
```

```
 {
console.log(res);
 var dataObj= JSON.stringify(res)

   var obj=JSON.parse(dataObj);
console.log(obj["data"]);
  this.uploadedImages = obj["data"];

  console.log("value of uploadedImages"+this.uploadedImages);


 },error=>
   {

     console.log(error);
   });
   /*the below function gets all the files names from the Directory*/
   this.claimsubmitionService.getFileListFromDirectory(body).subscribe(res =>
 {
console.log(res);
 var dataObj= JSON.stringify(res)

   var obj=JSON.parse(dataObj);
console.log(obj["data"]);
  this.uploadedImages = obj["data"];

  this.imgURL=this.uploadedImages;
  console.log("value of uploadedImages"+this.uploadedImages);


   var dataObj= JSON.stringify(res)
     var obj=JSON.parse(dataObj);
```

```
      for (var sub in obj )
      {
        var objSub = obj[sub];
        var elementsArray = Object.keys(objSub);
        this.lab=obj[sub]['originalName'];
this.imgURL='data:image/png;base64,'+obj[sub]['b64'];

       this.btn1=true;
       this.label=true;

      }





  },error=>
    {

      console.log(error);
    });
/*the below function gets all the files names from the DB*/
  this.claimsubmitionService.getFileListFromDB(body).subscribe(res => {
      var dataObj= JSON.stringify(res["data"])
      console.log("fileName--->"+dataObj);

      var obj=JSON.parse(dataObj);
      if(obj!="" && obj!=[] && obj!=null){
      var subobj=obj["filename"];
      console.log("fileName--->"+subobj);
      if(subobj!=""){
```

```
     this.isUpdateFlag=true;

     }

     this.imagesNew=obj[0]["filename"];

     }


 },error=>

   {


     console.log(error);

   });


}},error=>

   {


     console.log(error);

   });

    }},error=>

   {


     console.log(error);

   });

   console.log("request type----->"+this.imgType);


   this.uploader.onBuildItemForm = (fileItem: any, form: any) => {

   form.append('appNo', this.appNo); //note comma separating key and value

   form.append('teamId', this.teamId);

   form.append('reqType',this.imgType);

   form.append('name',this.name);

   };

this.uploader.onAfterAddingFile = (file) => { file.withCredentials = false; };
```

```
      this.uploader.onCompleteItem = (item: any, response: any, status: any, headers: any) => {
      console.log(response);

      var dataObj=response;
      console.log(dataObj);
      //this.items.push(JSON.parse(response));
      //this.commonFunction.modalReference.close();
//this.commonFunction.openDialog("File uploaded successfully." , '');
     // alert('File uploaded successfully');
      this.claimsubmitionService.getFileListFromDirectory(body).subscribe(res =>
  {
 console.log(res);
  var dataObj= JSON.stringify(res)
   console.log(dataObj);
    var obj=JSON.parse(dataObj);
 console.log(obj["data"]);
   this.uploadedImages = obj["data"];

   console.log("value of uploadedImages"+this.uploadedImages);


  },error=>
    {


      console.log(error);
    });
  }


    console.log("hight of screen-->"+screen.height);


   this.lab="No file Added";
   this.btn1=false;

   this.label=true;


  }
/*this function is to send data from one component to another or to another part of the same component*/
  editData(selectedItem: any){
  //console.log("selectedItem"+selectedItem.appNo);
  console.log(selectedItem.name);
  this.claimsubmitionService.send(selectedItem);
  var
body={filename:selectedItem.imageName,appNo:selectedItem.appNo,reqType:selectedItem.imgType,name:sel
ectedItem.name};
  this.preview(body);
}

public style = { transform: '', msTransform: '', oTransform: '', webkitTransform: '' };
commentId:string="";
comment:string="";
selectedItem="";
selectedItemSecondList="";
highlightedItem="";
hovering: boolean = false;
secondListItem="";
imageName="";
imageNameListOld:any=[];
imageNameList:any=[];
imagetoShow:any=[];
oldImage:any=[];
imageIndexOne:number = 0;
imageIndexList:number = 0;
lastImgIndex:number=0;
```

```
isHide=true;

public imagePath;


  public message: string;

/*this function is to enable customers to look at a preview of the documents uploaded by them*/

  preview(body) {

  /*function to display an image*/

    this.claimsubmitionService.displayImage(body).subscribe(res =>

      {

this.table=true;

           var dataObj= JSON.stringify(res)

               var obj=JSON.parse(dataObj);

               console.log("response---->"+obj[0]['b64']);

               /*this.policyNo=this.Details.get('policyNo').value;

               var filename=this.policyNo.concat(".pdf");

               saveAs(res,filename);*/

               this.imagetoShow='data:image/png;base64,'+obj[0]['b64'];

               console.log("imagetoShow--------->"+this.imagetoShow);

               });

  }


/*this function is used to call or open a dialogue*/

 private open(content) {

       this.modalService.open(content,

        { centered: true, backdrop: 'static', keyboard: false, backdropClass: 'dark-backdrop'})

          .result.then((result) => {

           width:"100%";

           this.closeResult = `Closed with: ${result}`;

       }, (reason) => {

           this.closeResult = `Dismissed ${this.getDismissReason(reason)}`;

        });
    }
/*this function is used to close or dismiss a dialogue*/
    private getDismissReason(reason: any): string {
        if (reason === ModalDismissReasons.ESC) {
            return 'by pressing ESC';
        } else if (reason === ModalDismissReasons.BACKDROP_CLICK) {
            return 'by clicking on a backdrop';
        } else {
            return  `with: ${reason}`;
        }
    }
url:any = '';
/*this function is called when the user selects a new file and the filename is retrieved*/
    onSelectFile(event,file: HTMLInputElement) {
    if (event.target.files && event.target.files[0]) {
      var reader = new FileReader();
      this.lab=event.target.files[0].name;
      this.filename=event.target.files[0].name;
      console.log("filename-->"+this.filename);
      reader.readAsDataURL(event.target.files[0]); // read file as data url

      reader.onload = (event) => { // called once readAsDataURL is completed
        this.url = reader.result;
       this.btn1=true;

      }
    }
  }
    /*this function is used to update the style of a component*/
    private updateStyle() {
```

```
    this.style.transform    =    `translate(${this.translateX}px,    ${this.translateY}px)    rotate(${this.rotation}deg)
scale(${this.scale})`;
    this.style.msTransform = this.style.transform;
    this.style.webkitTransform = this.style.transform;
    this.style.oTransform = this.style.transform;
  }
/*this function is called to zoomin to an image*/
zoomIn(){

  var myImg = document.getElementById("imgViewer");
        var currWidth = myImg.clientWidth;
        var currHeight=myImg.clientHeight;
        if(currWidth == 2500 && currHeight) return false;
         else{
            myImg.style.width = (currWidth + 100) + "px";
             myImg.style.height = (currHeight + 100) + "px";
        }
        }



/*this function is called to zoom out of a picture*/
  zoom: number = 0.5;
  zoomOut()
  {
  var myImg = document.getElementById("imgViewer");
        var currWidth = myImg.clientWidth;
        var currHeight=myImg.clientHeight;
        if(currWidth == 100 && currHeight==100) return false;
     else{
            myImg.style.width = (currWidth - 100) + "px";
```

```
          myImg.style.height = (currHeight -100) + "px";
      }
  }
  /*this function is to add Details into the DB*/
  addDetails()
{
var input = document.getElementById( 'input' );
var DetailsObj = this.Details.getRawValue(); // returns only the value of fields which are enabled and skips the
value of disabled fields.
this.Details.addControl('_id', new FormControl(this.Detailid));
this.Details.patchValue({_id: this.Detailid});
this.imgType=this.Details.get('claim').value;
 console.log("imageTpe--->"+this.imgType);
var
body1={Details:DetailsObj,userId:this.userId,appNo:this.appNo,imgType:this.imgType,filename:this.filename,
name:this.name};
/*this function is used to add all the details into the db*/
    this.claimsubmitionService.addDetailInDB(body1).subscribe(res =>
   {

     this.commonFunction.openDialog(res['data'],  '');
      var bod={name:this.name};
    /*this function is used to get the application number from the db*/
      this.claimsubmitionService.getAppNo(bod).subscribe(res =>{
          var dataObj=JSON.stringify(res["data"])
          var obj=JSON.parse(dataObj);
          var application=obj[0]['appNo'];
          var len=obj.length-1;
          /*this function is used to get Policy number from the db*/
      var body={appNo:application,name:this.name};
    this.claimsubmitionService.getPolicy(body).subscribe(res =>{
          var dataObj=JSON.stringify(res["data"])
          var obj=JSON.parse(dataObj);
          var policy=obj[0]['PolicyNo'];
          var len=obj.length-1;
          for(var i=0;i<=len;i++)
          {
          var holder=obj[i]['name'];
              var sub=obj[i]['personaldetails'];
              for(var j=0;j<sub.length;j++)
              {
              var na=sub[j]['_id'];
              }
          }
          console.log("personalDataId---->"+this.personalDataid);
          var
a={teamId:this.teamId,userId:this.userId,appNo:this.appNo,policy:policy,personalid:na,nomId:this.nomId,perso
nalDataid:this.personalDataid};
/*this function is used to add a new request into the db*/
          this.claimsubmitionService.addNewRequest(a).subscribe(res =>
          {
              this.commonFunction.openDialog(res['data'],  '');

                  this.personalDataid = res['generatedId'];

                  console.log("presonal data id returned = "+this.personalDataid);
                  /*this function is used to create a new request*/
                   this.claimsubmitionService.requestCase(a).subscribe(res=>
                   {
                      this.commonFunction.openDialog(res['data'],  '');

                   });
             });
          });
          });
            this.load();
           this.table=true;

            this.Details.reset();
   });



}
/*this function is used to call the loadData() function*/
load()
{
      var body={userId:this.userId, teamId:this.teamId,
inprogress:[this.isProgressValue],appNo:this.appNo,imgType:this.imgType,name:this.name};
       body={userId:this.userId, teamId:this.teamId,
inprogress:[0,1,2],appNo:this.appNo,imgType:this.imgType,name:this.name};
       this.loadData(body);
}
/*this function is used to load all the datas into the table*/
loadData(body){
   this.claimsubmitionService.getImageDetailFromDB(body)/*this function is used to get data from the image
table*/
       .subscribe(res => {
       console.log("res['data'].length--->"+res['data'].length);



       console.log("res['data']--->"+res['data']);
       var dataObj= JSON.stringify(res['data'])
     var obj=JSON.parse(dataObj);

    /* for (var sub in obj )
```

```
{
 var objSub = obj[sub];
 var elementsArray = Object.keys(objSub);


  console.log("tasklist response---->"+obj[sub]['inprogress']);
  if(Number(this.teamId)==5 || Number(this.teamId)==4){

  }else{
  if(obj[sub]['inprogress']==1){
  obj[sub]['inprogress']="Work In Progress"
  }else if(obj[sub]['inprogress']==0){
  obj[sub]['inprogress']="New Task"
  }else if(obj[sub]['inprogress']==2){
  obj[sub]['inprogress']="Rescaned Cases"
  }
  }

}*/

this.tasklist=obj;


const users = Array.from(this.tasklist);
  console.log("data length----->"+users.length);
if(users.length==0)
{
this.table=false;
}
else{
this.table=true;
}
if(users.length>=8)
{
        this.disable=false;
      }
      else
      {
      this.disable=true;
      }
    this.dataSource = new MatTableDataSource(users);
    this.dataSource.paginator = this.paginator;
    this.dataSource.sort = this.sort;
 });

}
```

/*this function is used to enable the user to download the claim from once all the necessary documents are uploaded*/

```
onDownload(){
this.spinner.show();
var body={};
```

/*this function is used for calling the function the downloads the pdf*/

```
   this.claimsubmitionService.downloads(body).subscribe(res =>
   {
              console.log("response---->"+res);
              this.policyNo="claimform";
              var filename=this.policyNo.concat(".pdf");
              saveAs(res,filename);
              this.commonFunction.openDialog("The Claim form has been downloaded please fill it completely and upload the filled form in the same page",  '');




      this.spinner.hide(); // To hide the spinner
   });

}

}
```

claimsubmition.service.ts

```
@Injectable({
  providedIn: 'root'
})
export class ClaimsubmitionService {
               result:any;
  private sub = new Subject();
  subj$ = this.sub.asObservable();

  private subpopup = new Subject();
  subpopupj$ = this.subpopup.asObservable();
  constructor(private _http:Http,private http:HttpClient) { }
  /*this function is used to display an image when called*/
    getimage(body)
  {



      return this.http.post('/api/downloadPImage',body)/*this function is used to display an image when called*/
        .pipe(map((result: Response) => this.result = result));



   }
   /*this function is used to get new request int the db*/
    getNewRequest(inputValue)
   {
      console.log("in getPODetailsFromDB service function..."+JSON.stringify(inputValue));
      return this.http.post("/api/getNewRequest", {
      inputValue
      })
     }
      /*this function is used to add a new request int the db*/
       requestCase(body) {

    console.log("in submit case..."+JSON.stringify(body));
       return this.http.post('/api/requestCase',body)
        .pipe(map((result: Response) => this.result = result));
      }
        /*this function is used to add details into the db*/
addDetailInDB(inputValue)
   {
      console.log("cash value "+JSON.stringify(inputValue));
      return this.http.post("/api/addImageDetailsinDB", {
      inputValue
      })
     }

   /*this function is used to add a new request int the db*/
    addNewRequest(inputValue)
   {
      console.log("cash value "+JSON.stringify(inputValue));
      return this.http.post("/api/addNewRequest", {
      inputValue
      })
     }

   /*this function is used to submit a new case*/
```

```
 submitCase(body) {


   console.log("in submit case..."+JSON.stringify(body));
      return this.http.post('/api/submitCase1',body)
       .pipe(map((result: Response) => this.result = result));
     }
     getFileListFromDirectory(body)
 {
/* return this._http.get("/api/downloadfImage", {params: {
       appNo: appNo


     }
     })
     .pipe(map(result => this.result = result.json().data));
     */

     return this.http.post('/api/downloadfImage',body)
       .pipe(map((result: Response) => this.result = result));




  }

  /*this function is used to get appNo from the db*/
      getAppFromDB(inputValue)
  {


     return this.http.post("/api/getAppFromDB", {
     inputValue
     })
   }
 /*this function is used to get Nominee Id from the db*/
   getNomIdFromDB(inputValue)
 {

   return this.http.post("/api/getNomIdFromDB", {
   inputValue
   })
  }

 /*this function is used to get claim Detail from the db*/
       getClaimDetailFromDB(inputValue)
 {

   return this.http.post("/api/getClaimDetailFromDB", {
   inputValue
   })
  }

 /*this function is used to get image Details from the db*/
   getImageDetailFromDB(inputValue)
{
   return this.http.post("/api/getImageDetailsDB", {
   inputValue
   })
}

 /*this function is used to display Images*/
displayImage(inputValue)
 {

  /*return this.http.get('/api/displayImage/', {
         responseType:"blob"


        //  search: // query string if have
        })*/

        return this.http.post('/api/dispImage',inputValue)
         .pipe(map((result: Response) => this.result = result));




        /* return this.http.post('/api/downloadFile','')
         .pipe(map((result: Response) => this.result = result));
*/

    }

   /*this function downloads the file*/
     downloads(inputValue)
   {

    return this.http.get('/api/downloadClaimFile/', {
         responseType:"blob"


        //  search: // query string if have
        })




        /* return this.http.post('/api/downloadFile','')
         .pipe(map((result: Response) => this.result = result));
*/

    }
```

```
 /*this function is used to get policy number From the db*/
     getPolicy(inputValue)
{
    return this.http.post("/api/getPolicy", {
    inputValue
    })
}

 /*this function is used to get Application number  from the db*/
  getAppNo(inputValue)
 {

    return this.http.post("/api/getAppNo", {
    inputValue
    })
  }


send(value: string) {
  this.sub.next(value);
}


openScanningTeamPopup(value: string) {
  this.subpopup.next(value);
}


sendData(value: string) {
  this.sub.next(value);
}

getFileListFromDB(body) {
 /*return this._http.get("/api/filelist")
    .pipe(map(result => this.result = result.json().data));*/

     return this.http.post('/api/filelist',body)
      .pipe(map((result: Response) => this.result = result));

}

}
```

customer.component.html

```
<div class="container-fluid h-100 scroll" *ngIf="search">
        <div class="row m-0 h-20">
                <div class="col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12 text-center h-100" >
                        <form class="max-height col-12 p-0" role="form" [formGroup]="Details" (submit)='addDetails()' method="POST" #myForm="ngForm" autocomplete="off" novalidate (keydown.enter)="$event.preventDefault()">
                                <div class="input-append h-100">
                                <mat-form-field class="i">
         <input type="number" (keyup.enter)="get()" matInput placeholder="Enter Application number" formControlName="appNo" >
        </mat-form-field>
        <button type="button" (click)="get()" class="btn btn-info circle" data-toggle="button" [disabled]="!Details.valid"><i class="fa fa-search icon"></i></button>
                                </div><button type="button" class="btn btn-outline-info mr-2 float-right" (click)="form()" style="line-height: normal;margin-left: 15px;width: 4.5em;height: 2em;text-align: center;" *ngIf="btn1">Claim</button>



                        </form>
                </div>
```

```
              </div>
                          <div class="row m-0 h-40">
                                      <div class="col-sm-12 col-md-12 h-100 col-lg-12 col-xl-12 col-12 text-center">
                                                  <div *ngIf="div" class="h-100 div1">
                                                  <h3>Personal Details</h3>
     <table>
   <thead>
     <tr>
      <th *ngFor="let head of dbArray1">{{head}}</th>
     </tr>
   </thead>
   <tbody>
     <tr *ngFor="let item of items1[0]">
      <td >{{item._id}}</td>
      <td>{{item.appNo}}</td>
      <td>{{item.name}}</td>
      <td>{{item.DateOfBirth}}</td>
      <td>{{item.Gender}}</td>
      <!-- <td>{{item.Address}}</td> -->
      <td>{{item.creationDate}}</td>
      <td>{{item.MobileNo}}</td>
      <td>{{item.Email}}</td>
      <td>{{item.personType}}</td>
       <td>{{item.AddressId}}</td>
<!--      <td>{{item.status}}</td> -->
<!--      <td>{{item.MaritalStatus}}</td> -->
    <!--  <td>{{item.AccountNo}}</td>
      <td>{{item.Bank}}</td>
      <td>{{item.Branch}}</td>
      <td>{{item.AccountType}}</td>

      <td>{{item.IFSC}}</td>

       -->

    </tr>


  </tbody>

</table>

                                                        </div>

                                                </div>

                                        </div>

                                        <br><br><br>

                                        <div class="row m-0 h-40">

                                                <div class="h-100 col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12 text-center">

                                                        <div *ngIf="div1" class="h-100 div1">

                                                <h3>Address Details</h3>

     <table>

  <thead>

    <tr>

      <th *ngFor="let head of dbArray2 ">{{head}}</th>

    </tr>

  </thead>

  <tbody>

     <tr *ngFor="let item of items2[0]">

      <td>{{item._id}}</td>

      <!-- <td>{{item.PersonType}}</td>

      <td>{{item.personaldetails_id}}</td> -->

      <td>{{item.AddressLine1}}</td>

      <td>{{item.AddressLine2}}</td>

     <!--  <td>{{item.Locality}}</td>

      <td>{{item.street}}</td>
      <td>{{item.landmark}}</td> -->
      <td>{{item.city}}</td>
      <td>{{item.state}}</td>
      <td>{{item.country}}</td>
      <td>{{item.PinCode}}</td>
 <!--     <td>{{item.status}}</td> -->
  <!--    <td>{{item.creationDate}}</td> -->
    </tr>

  </tbody>
</table>
                                                  </div>
                                      </div>
                          </div>
  </div>
  <div class="container-fluid h-100 scroll" *ngIf="isclaim">
        <div class="row pt-3 h-80 overflow-y" >
    <form class="max-height col-12 p-0" role="form" [formGroup]="Details1" (submit)='addDetails()'
method="POST" #myForm="ngForm" autocomplete="off" novalidate>
      <!-- <div class="form-group"> -->

        <!-- <div class="example-container max-height"> -->
      <h3>ClAIM FORM</h3>
      <div class="row m-0">
        <div class="col-sm-3 col-md-3 col-lg-3 col--xl-3 col-3">
            <mat-form-field>
                <input type="text" matInput placeholder="Application No." formControlName="apNo" >
            </mat-form-field>
        </div>
        <div class="col-sm-3 col-md-3 col-lg-3 col--xl-3 col-3">
          <mat-form-field>
```

```
            <input type="text" matInput placeholder="Policy No." formControlName="policyNo" >
          </mat-form-field>
        </div>
        <div class="col-sm-3 col-md-3 col-lg-3 col--xl-3 col-3">
          <mat-form-field>
            <input type="text" matInput placeholder="Name(Life Assured's)" formControlName="name" >
          </mat-form-field>
        </div>
        <div class="col-sm-3 col-md-3 col-lg-3 col--xl-3 col-3">
          <mat-form-field class="example-full-width">
            <input matInput [matDatepicker]="picker2" placeholder="Life Assured Date Of Birth"
formControlName="dob" name="dob" disabled readonly="readonly">
            <mat-datepicker-toggle matSuffix [for]="picker2"></mat-datepicker-toggle>
            <mat-datepicker #picker2 disabled="false" touchUi></mat-datepicker>
          </mat-form-field>
        </div>
      </div>
      <hr>
      <h3>NOMINEE'S DETAILS</h3>
      <hr>
      <div class="row m-0">
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
              <mat-form-field>
            <input type="text" matInput placeholder="Nominee Name" formControlName="name1" >
          </mat-form-field>
        </div>



        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
          <mat-form-field class="example-full-width">
```

```
            <input matInput [matDatepicker]="picker" placeholder="Nominee's Date OF Birth"
formControlName="dob1" name="dp" disabled readonly="readonly" >
            <mat-datepicker-toggle matSuffix [for]="picker"></mat-datepicker-toggle>
            <mat-datepicker #picker disabled="false" touchUi></mat-datepicker>
          </mat-form-field>
             </div>
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">


      <label class="radio-btn-label mb-1 col-sm-6 p-0">Nominee's Gender</label>
        <mat-radio-group class ="col-sm-6" id="Gender1" name="Gender1" formControlName="Gender1"><br>
            <mat-radio-button value="Male" class="mr-1 custom-radio-btn">Male</mat-radio-button>
            <mat-radio-button value="Female" class="custom-radio-btn">Female</mat-radio-button>
            <mat-radio-button value="Transgender" class="custom-radio-btn">Transgender</mat-radio-button>
        </mat-radio-group>
        </div>
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
         <mat-form-field >
              <textarea matInput placeholder="Address line 1" formControlName="addressline1" ></textarea>
          </mat-form-field>
        </div>
      </div>


      <div class="row m-0">
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
             <mat-form-field >
            <textarea matInput placeholder="Address line 2" formControlName="addressline2" ></textarea>
          </mat-form-field>
        </div>
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
          <!-- <mat-form-field >
           <input type="text" matInput placeholder="Nominee's Locality" name="Locality1"
formControlName="Locality1">

          </mat-form-field> -->
           <mat-form-field >
             <input type="text" matInput placeholder="Nominee's city" formControlName="city1">
          </mat-form-field>
        </div>
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
         <!--  <mat-form-field >
             <input type="text" matInput placeholder="Nominee's street" formControlName="street1" >
          </mat-form-field> -->
          <mat-form-field >
           <input type="text" matInput  placeholder="Nominee's state" name="state1" formControlName="state1"
>

          </mat-form-field>
        </div>
          <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
             <!-- <mat-form-field >
           <textarea matInput placeholder="Nominee's landmark" name="landmark1"
formControlName="landmark1" ></textarea>

          </mat-form-field> -->
              <mat-form-field >
             <input type="text" matInput placeholder="Nominee's country" formControlName="country1" >
          </mat-form-field>
      </div>
      </div>


       <div class="row m-0">
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
         <mat-form-field >
           <input type="text" matInput placeholder="Nominee's PinCode" name="PinCode1"
formControlName="PinCode1" >


         </mat-form-field>
        </div>
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
              <mat-form-field >
            <input type="number" matInput placeholder="Nominee's Mobile No" formControlName="MobileNo1"
           pattern=".{10}">
         </mat-form-field>
        </div>
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
         <mat-form-field >
           <input type="text" matInput placeholder="Nominee's Email" name="Email1"
formControlName="Email1"  pattern="[a-z0-9._%+-]+@[a-z0-9.-]+\.[a-z]{2,}$">


         </mat-form-field>



        </div>
              <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">


       <mat-form-field >
           <input type="text" matInput placeholder="Nominee's Account No" name="AccNo"
formControlName="AccNo">
```

```
          </mat-form-field>


        </div>
    </div>

        <div class="row m-0">
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
          <mat-form-field >
            <input type="text" matInput  placeholder="IFSC Code" name="ifsc" formControlName="ifsc">


          </mat-form-field>
        </div>
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
             <mat-form-field >
            <input type="text" matInput placeholder="Nominee's Bank name" name="bank"
formControlName="bank">


          </mat-form-field>
        </div>
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
    <mat-form-field >
            <input type="text" matInput placeholder="Nominee's Branch" name="branch"
formControlName="branch">


          </mat-form-field>
        </div>

        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
          <mat-form-field >
            <input type="text" matInput placeholder="Nominee's Account Type" name="accType"
formControlName="accType">
```

```
    </mat-form-field>
   </div>
  </div>
  <div class="row m-0">
   <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
     <div class="input-group form-group input-group-sm mb-3">
      <div class="input-group-prepend" >
       <span class="input-group-text" id="inputGroup-sizing-sm"
[ngStyle]="{'background':'white',width:'8em',height:'4em'}">
        <mat-card [ngStyle]="{'height':'3em','padding-left':'0em','width':'7em'}">Type Of claim:</mat-card>

       </span>
       <mat-form-field class="a">
       <div>
      <mat-select #matSelect [panelClass]="a" formControlName="claim" placeholde="Type of claim"
id="claim">
      <mat-option value="DeathClaim">Death Claim</mat-option>
      <mat-option value="NonDeathClain">Non Death Claim</mat-option>

     </mat-select>
    </div>
     </mat-form-field>
      </div>
    </div>
   </div>
   <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">

   </div>
   <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
```

```
        </div>
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
          <button type="button" mat-stroked-button class="btn btn-outline-danger mr-2" (click)="addDetails();" [disabled]="!Details1.valid" *ngIf="btn3" style="margin-left: 15px;width: 0.5em;height: 2em;text-align: center;" [routerLink]="['/summary']">Save</button>

          <button type="button" mat-stroked-button class="submit-btn btn-border mr-2 red" (click)="addDetails1();" [disabled]="!Details1.valid" *ngIf="btn5" style="margin-left: 15px;width: 0.5em;height: 2em;text-align: center;"  [routerLink]="['/summary']">Submit</button>
          <button type="button" mat-stroked-button class="btn btn-outline-danger mr-2 " *ngIf="btn5" style="margin-left: 15px;width: 0.5em;height: 2em;text-align: center;" [routerLink]="['/summary']">Go Back</button><br><br>
          <button type="button" mat-stroked-button class="btn btn-outline-danger mr-2 " *ngIf="btn6" style="margin-left: 15px;width: 0.5em;height: 2em;text-align: center;" (click)="reset()" [routerLink]="['/Customer']">Go Back</button><br><br>
          <button type="button" mat-stroked-button class="custom-btn submit-btn red" (click)="addDetails();" [disabled]="!Details1.dirty" *ngIf="btn2" style="margin-left: 15px;width: 0.5em;height: 2em;text-align: center;"  [routerLink]="['/summary']">Update</button>
        <button type="button" mat-stroked-button class="btn btn-outline-danger mr-2 " *ngIf="btn4" style="margin-left: 15px;width: 0.5em;height: 2em;text-align: center;" [routerLink]="['/summary']">Go Back</button>
          <button type="button" mat-stroked-button class="btn btn-outline-danger mr-2 " (click)="commonFunction.open(content);" [disabled]="!Details1.valid" *ngIf="btn4" style="margin-left: 15px;width: 0.5em;height: 2em;text-align: center;" >Submit</button>
        </div>
    </div>
        <div class="row m-0">
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
        </div>
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
        </div>
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
<!--
    [disabled]="!Details.valid"  -->

        </div>
         <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">


        </div>
    </div>
      <div class="row m-0">
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
        </div>
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">


        </div>
        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">




        </div>

        <div class="col-sm-3 col-md-3 col-lg-3 col-xl-3 col-3">
        </div>
      </div>
   </form>
</div>
</div>
```

```
<!--                    d,appNo,name,DateOfBirth,Gender,Address,creationDate,MobileNo,Email,personType,Marital
Status,AccountNo,Bank,Branch,AccountType,IFSC,_id,PersonType,personaldetails_id,appNo,buildingname,Ho
useNo,Locality,street,landmark,city,state,country,PinCode,creationDate -->



<ng-template #content let-c="close" let-d="dismiss" class="scale50"  style="position:relative">
  <div class="modal-header border-0 " id="mydiv" >
    <div class="row m-0">


      <div class="col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12">
         <div class="d-inline-flex pr-2 small-horizontal-line-right" >
      <textarea placeholder="Enter Comments" required style="width: 100%;height: 9em"></textarea>
    </div>
      </div>

      <div class="col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12">
      </div>

    <br>
    <br>
      <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6">
            <div class="d-inline-flex pr-2 small-vertical-line-right">
             <button mat-button class="submit-btn btn-border mr-2 p-0 red" (click)="updateDetails()"
[routerLink]="['/claim']" style="width: 100%">Accept</button>
      </div>
        </div>
          <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6">
             <div class="d-inline-flex pr-2">
             <button type="button" mat-button class="submit-btn  btn-border mr-2 p-0 red" (click)="reject()"
[routerLink]="['/claim']" style="width: 100%">Reject</button>
           </div>
        </div>



    </div>
         <button type="button" class="close " aria-label="Close" (click)="d('Cross click')" >
        <span aria-hidden="true">&times;</span>
      </button>

  </div>
</ng-template>
```

customer.component.css

```
div{
		width: 100%;
}
.mat-form-field-infix {
		display: block;
    position: relative;
    flex: auto;
    min-width: 0;
		width:100%;
}
mat-form-field{
		width: 65%;
}
table{
		width: 100%;
		align-content: center;
}
/*td{
		padding: 1em;
		text-align:center;
```

```
}*/
tr{
     padding: 1em;
     overflow: auto;

    text-align: left;

 }
 td,th{
      padding-left: 2em !important;
 }
 /* Chrome, Safari, Edge, Opera */
input::-webkit-outer-spin-button,
input::-webkit-inner-spin-button {
  -webkit-appearance: none;
  margin: 0;
}

/* Firefox */
input[type=number] {
  -moz-appearance:textfield;
}
i{
    margin-left: 1px;
    font-size: 15px;
    height: 30px;
    vertical-align: middle;
}
.div1{
              overflow-x:scroll;
}
.scroll{
```

```
              overflow: auto;
}
.modal-content{
                  width: 1000em;
    height: 1100em;
}
h3{
              text-align:left;
}
.a {
    width: 43%;
}
button{
  height: 2em;
 text-align: center;
    line-height: 29px;
}
mat-select {
    width: fit-content !important;
    min-width: 50%;
   /*background-color: white;*/
}

::ng-deep .mat-form-field-infix {
    min-width: 10px !important;
    width: fit-content !important;
     /*background-color: white;*/
}

::ng-deep mat-select .mat-select-value {
    width: fit-content;
    min-width: 2ch;
    max-width: 25ch;
    /* background-color: white;*/
}
::ng-deep mat-select .mat-select-placeholder {
    width: fit-content;
    min-width: 10ch;
    /*background-color: white;*/
}
::selection {
    text-shadow: none;
    background: #FE2E64;
    color: #fff;
}
.red:hover {
  background-color: #DF2406;
}
.white:hover {
  background-color: #D8D8D8;
}
```

customer.component.ts

```
@Component({
  selector: 'app-customer',
  templateUrl: './customer.component.html',
  styleUrls: ['./customer.component.css']
})
export class CustomerComponent implements OnInit {
isAddFlag="add";
 closeResult: string;
 Detailid:string="";
div:boolean=false;
div1:boolean=false;
btn1:boolean=false;
btn2:boolean=true;
btn3:boolean=false;
btn4:boolean=false;
btn5:boolean=false;
btn6:boolean=true;
perId:string;
isclaim:boolean=false;
search:boolean=true;
public dbArray1=[];
public items1=[];
public dbArray2=[];
public items2=[];
public dbArray=[];
public items=[];
userId: string;
teamId: string;
appNo:string;
agentCode:string;
dateCreated:string;
isprogress:string;
name1:string;
name:string;
dob1:string;
Gender1:string;
Address1:string;
MobileNo1:number;
```

```
Email1:string;

HouseNo1:string;

buildingname1:string;

Locality1:string;

street1:string;

landmark1:string;

city1:string;

state1:string;

country1:string;

PinCode1:string;

AccNo:number;

bank:string;

ifsc:string;

branch:string;

accType:string;

dob:string;

apNo:string;

claimId:number;

policyNo:string;

claim:string;

disabledBtn:boolean=false;

isDelete:boolean=false;

public names=[];

Details = new FormGroup({

    appNo: new FormControl({value:''},Validators.required)

  });

  Details1 = new FormGroup({

    apNo:new FormControl({value:''},Validators.required),

    policyNo:new FormControl({value:''},Validators.required),

    name:new FormControl({value:''},Validators.required),
    dob:new FormControl({value:''},Validators.required),
    name1: new FormControl({value:''},Validators.required),
    dob1: new FormControl({value:''},Validators.required),
    Gender1: new FormControl({value:''}),
    addressline1: new FormControl({value:''},Validators.required),
    addressline2: new FormControl({value:''},Validators.required),
    city1: new FormControl({value:''},Validators.required),
    state1: new FormControl({value:''}, Validators.required),
    country1: new FormControl({value:''},Validators.required),
    PinCode1: new FormControl({value:''}, Validators.required),
    MobileNo1: new FormControl({value:''},Validators.required),
    Email1: new FormControl({value:''}, Validators.required),
    AccNo: new FormControl({value:''},Validators.required),
    bank: new FormControl({value:''}, Validators.required),
    ifsc: new FormControl({value:''},Validators.required),
    branch: new FormControl({value:''}, Validators.required),
    accType: new FormControl({value:''},Validators.required),
    claim: new FormControl({value:''},Validators.required)
  });
  constructor(public router: Router, private route: ActivatedRoute,private fb: FormBuilder, public
commonFunction:CommonFunction, private dialog: MatDialog , private sharedDataService: DataService, private
http: HttpClient,@Optional() @Inject('config') public moduleConfig: ImageViewerConfig,private modalService:
NgbModal,private datePipe: DatePipe,public customerService:CustomerService)
{   this.route.queryParams.subscribe(params => {
        this.appNo = params["appNo"];
        this.agentCode = params["agentCode"];
         this.dateCreated = params["dateCreated"];
         this.isprogress=params["isprogress"];


        console.log("appNo--->"+this.appNo+"----name---"+this.name);
        });

            }

  ngOnInit() {
  var txt = JSON.parse(localStorage.getItem('usersinfo'));


  this.userId=txt[0]["_id"];

  this.teamId=(txt[0]["teamId"]);


  this.Details = this.fb.group
({
    appNo:[' ',Validators.required]
});
this.Details1 = this.fb.group
({
    apNo:['',Validators.required],
    policyNo:['',Validators.required],
    name:['',Validators.required],
    dob:['',Validators.required],
    name1:['',Validators.required],
    dob1:['',Validators.required],
    Gender1:['',Validators.required],
    addressline1:['',Validators.required],
    addressline2:['',Validators.required],
    city1:['',Validators.required],
    state1:['',Validators.required],
    country1:['',Validators.required],
    PinCode1:['',Validators.required],
    MobileNo1: ['',Validators.required],
    Email1: ['',Validators.required],
    AccNo: ['',Validators.required],
    bank: ['',Validators.required],
    ifsc: ['',Validators.required],
    branch:['',Validators.required],
    accType:['',Validators.required],
    claim:['',Validators.required]
});

  this.isAddFlag="add";
  this.btn2=true;
  this.btn3=false;
  this.btn5=true;
  this.btn6=false;
  if(this.teamId=="15")
{
  this.btn4=true;
  this.btn2=false;
  this.btn3=false;
  this.btn5=false;
   this.div=false;
   this.btn5=false;
   this.btn6=false;
this.div1=false;
this.btn1=false;
this.isclaim=true;
this.search=false;
var body={appNo:this.appNo}

  /*gets customer details from db*/
this.customerService.getDetails1FromDB(body).subscribe(res => {
    var dataObj= JSON.stringify(res['data'])

    var obj=JSON.parse(dataObj);

          this.name=obj[0]['name'];

          this.claimId=obj[0]['_id'];

          console.log("claimId---->"+this.claimId);

         /* this.dob1=obj[0]['NomineeDateOfBirth'];

          this.Gender1=obj[0]['NomineeGender'];

          this.MobileNo1=obj[0]['NomineeMobileNo'];

          this.Email1=obj[0]['NomineeEmail'];*/

          var bod={claimId:this.claimId};


/*gets customer's address from the DB*/

         this.customerService.getAddress(bod).subscribe(res => {

          var data= JSON.stringify(res['data'])

    var ob=JSON.parse(data);

          this.HouseNo1=ob[0]['AddressLine1'];

          this.buildingname1=ob[0]['AddressLine2'];

         /* this.Locality1=obj[0]['NomineeLocality'];

          this.street1=ob[0]['NomineeStreet'];

          this.landmark1=ob[0]['NomineeLandmark'];*/

          this.city1=ob[0]['city'];

          this.state1=ob[0]['state'];

          this.country1=ob[0]['country'];

          this.PinCode1=ob[0]['PinCode'];

          this.Details1.patchValue({addressline2: this.buildingname1});

          this.Details1.patchValue({addressline1: this.HouseNo1});

          this.Details1.patchValue({city1:this.city1});



            this.Details1.patchValue({state1: this.state1});
         this.Details1.patchValue({country1:this.country1});

         this.Details1.patchValue({PinCode1:this.PinCode1});

        });

         var bod={claimId:this.claimId}

 /*gets Nominee Details from the DB*/

        this.customerService.getNominee(bod).subscribe(res => {

         var done= JSON.stringify(res['data'])

          var on=JSON.parse(done);

        this.name1=on[0]['name'];

        this.dob1=on[0]['DateOfBirth'];

         this.Gender1=on[0]['Gender'];

        this.MobileNo1=on[0]['MobileNo'];

        this.Email1=on[0]['Email'];

        console.log("MobileNo---->"+this.MobileNo1);

         this.Details1.patchValue({Gender1: this.Gender1});


         this.Details1.patchValue({MobileNo1:this.MobileNo1});

         this.Details1.patchValue({Email1:this.Email1});


         this.Details1.patchValue({name1: this.name1});

         this.Details1.patchValue({dob1:this.dob1});

        });

     /*gets Nominee's Bank Details from the DB*/

        this.customerService.getBank(bod).subscribe(res => {

         var d= JSON.stringify(res['data'])

          var o=JSON.parse(d);

        this.AccNo=o[0]['AccountNo'];

        this.bank=o[0]['Bank'];


        this.ifsc=o[0]['IFSC'];
```

```
            this.branch=o[0]['Branch'];

            this.accType=o[0]['AccountType'];

                    this.Details1.patchValue({AccNo:this.AccNo});

              this.Details1.patchValue({bank:  this.bank});

              this.Details1.patchValue({ifsc: this.ifsc});

              this.Details1.patchValue({branch:this.branch});

              this.Details1.patchValue({accType:this.accType});

            });

            this.dob=obj[0]['DateOfBirth'];

            this.apNo=obj[0]['appNo'];

            this.policyNo=obj[0]['PolicyNo'];

            this.claim=obj[0]['claimType'];

            console.log("name--->"+this.claim);

            console.log("claimId--->"+this.claimId);

                console.log("Bank----->"+this.bank);

            this.Details1.patchValue({apNo: this.apNo});

              this.Details1.patchValue({policyNo:this.policyNo});

              this.Details1.patchValue({name: this.name});

              this.Details1.patchValue({dob:this.dob});


              /*this.Details1.patchValue({Locality1: this.Locality1});

              this.Details1.patchValue({landmark1: this.landmark1});

              this.Details1.patchValue({street1:  this.street1});*/




              this.Details1.patchValue({claim:this.claim});

        });

}

else{

    if(!this.appNo)
    {
      this.isAddFlag
    }
    else if(this.appNo)
    {
    this.isAddFlag="edit";
    this.div=false;
this.div1=false;
this.btn1=false;
this.isclaim=true;
this.search=false;
var body={appNo:this.appNo}

  /*gets customer's details from the DB*/
this.customerService.getDetailsDB(body).subscribe(res => {



      var dataObj= JSON.stringify(res['data'])
      var obj=JSON.parse(dataObj);


            this.name=obj[0]['name'];


             this.claimId=obj[0]['_id'];


            var bod={claimId:this.claimId}

  /*gets customer's address from the DB*/
this.customerService.getAddress(bod).subscribe(res => {
               var data= JSON.stringify(res['data'])
      var ob=JSON.parse(data);
            this.HouseNo1=ob[0]['AddressLine1'];
```

```
          this.buildingname1=ob[0]['AddressLine2'];
         /* this.Locality1=obj[0]['NomineeLocality'];
          this.street1=ob[0]['NomineeStreet'];
          this.landmark1=ob[0]['NomineeLandmark'];*/
          this.city1=ob[0]['city'];
          this.state1=ob[0]['state'];
          this.country1=ob[0]['country'];
          this.PinCode1=ob[0]['PinCode'];
            this.Details1.patchValue({addressline2: this.buildingname1});
            this.Details1.patchValue({addressline1: this.HouseNo1});
             this.Details1.patchValue({city1:this.city1});

            this.Details1.patchValue({state1: this.state1});
            this.Details1.patchValue({country1:this.country1});
            this.Details1.patchValue({PinCode1:this.PinCode1});
          });
            var bod={claimId:this.claimId}

/*gets Nominee's details from the DB*/
          this.customerService.getNominee(bod).subscribe(res => {
                var dose= JSON.stringify(res['data'])
             var os=JSON.parse(dose);
          this.name1=os[0]['name'];
          this.dob1=os[0]['DateOfBirth'];
           this.Gender1=os[0]['Gender'];
          this.MobileNo1=os[0]['MobileNo'];
          this.Email1=os[0]['Email'];
            this.Details1.patchValue({Gender1: this.Gender1});
                   console.log("MobileNo---->"+this.MobileNo1);
            this.Details1.patchValue({MobileNo1:this.MobileNo1});
            this.Details1.patchValue({Email1:this.Email1});
```

```
              this.Details1.patchValue({name1: this.name1});
              this.Details1.patchValue({dob1:this.dob1});
            });

/*gets customer's Bank details from the DB*/
            this.customerService.getBank(bod).subscribe(res => {
           var d= JSON.stringify(res['data'])
               var o=JSON.parse(d);
            this.AccNo=o[0]['AccountNo'];
            this.bank=o[0]['Bank'];
                console.log("Bank----->"+this.bank);
            this.ifsc=o[0]['IFSC'];
            this.branch=o[0]['Branch'];
            this.accType=o[0]['AccountType'];
                          this.Details1.patchValue({AccNo:this.AccNo});
              this.Details1.patchValue({bank:  this.bank});
              this.Details1.patchValue({ifsc: this.ifsc});
              this.Details1.patchValue({branch:this.branch});
              this.Details1.patchValue({accType:this.accType});
            });
            this.dob=obj[0]['DateOfBirth'];
            this.apNo=obj[0]['appNo'];
            this.policyNo=obj[0]['PolicyNo'];
            this.claim=obj[0]['claimType'];
            console.log("name--->"+this.name);
            console.log("claimId--->"+this.claimId);
                console.log("Bank----->"+this.bank);
            this.Details1.patchValue({apNo: this.apNo});
              this.Details1.patchValue({policyNo:this.policyNo});
              this.Details1.patchValue({name: this.name});
```

```
             this.Details1.patchValue({dob:this.dob});

             /*this.Details1.patchValue({landmark1: this.landmark1});
             this.Details1.patchValue({Locality1: this.Locality1});
             this.Details1.patchValue({street1:  this.street1});*/


             this.Details1.patchValue({claim:this.claim});
        });

    }
    }
}


  /*gets customers details from db*/
get(){
    this.items1=[];
    this.items2=[];
     this.dbArray1=[];
    this.dbArray2=[];
             this.appNo=this.Details.get('appNo').value;
             if(this.appNo==' ')
             {
             this.div=false;
             this.div1=false;
             this.btn1=false;
             }
             else{
  var a=[];
             this.div=true;
             this.div1=true;
```

```
            this.btn1=true;
this.btn2=false;
this.btn5=false;
this.btn6=true;
this.btn3=true;
            console.log("appNo---->"+this.appNo);
             var body={appNo:this.appNo};
this.customerService.getDetailsFromDB(body).subscribe(res=>{
                            var dataObj=JSON.stringify(res["data"]);
                            var obj=JSON.parse(dataObj);
                            var sub=JSON.parse(dataObj);
                            var len=(obj.length-1);


    for (var sub1 in obj )
    {
     var objSub = obj[sub1];
     console.log("object value---->"+Object.keys(objSub))
    this.dbArray1 = Object.keys(objSub);

   this.items1=Array.of(obj);
    this.perId=objSub['AddressId'];
    console.log("personal Id---->"+this.perId);
        var bod={perId:this.perId}
 this.customerService.getAddDetailsFromDB(bod).subscribe(res=>{
    var dataObj=JSON.stringify(res["data"]);
    var obj=JSON.parse(dataObj);
    var sub=JSON.parse(dataObj);
    var len=(obj.length-1);
```

```
     for (var sub1 in obj )

     {

     a=this.items2;

      var objSub = obj[sub1];

      console.log("object value---->"+Object.keys(objSub));

     this.dbArray2 = Object.keys(objSub);

    this.items2=Array.of(obj);


      }

   if (this.dbArray2.length === 0) {

  this.div1=false;

  this.btn1=false;

  this.items2=this.items2.concat(a);

}



 console.log("dbArray1---->"+this.dbArray1);

 console.log("items1----->"+this.items1);

  console.log("dbArray2---->"+this.dbArray2);

 console.log("items2----->"+this.items2);

 this.dbArray=this.dbArray1.concat(this.dbArray2);

   console.log("dbArray---->"+this.dbArray);

 console.log("items----->"+this.items);

  })

      }

      if (this.dbArray1.length === 0) {

  this.div=false;

  this.div1=false;

  this.btn1=false;
```

```
}




  })
              }


}
reset()
{
 this.isclaim=false;
this.search=true;
this.Details.reset();
}

  /*hides the search bar and displays form*/
 form(){
this.div=false;
this.div1=false;
this.btn1=false;
this.isclaim=true;
this.search=false;
 }

  /*adds claim details into Database*/
 addDetails()
 {
 var DetailsObj = this.Details1.getRawValue(); // returns only the value of fields which are enabled and skips the
value of disabled fields.
this.Details1.addControl('_id', new FormControl(this.Detailid));
this.Details1.patchValue({_id: this.Detailid});
```

```
if(this.teamId=="15")
{
  this.isDelete=true;
  this.isAddFlag="edit";
}
var                                                         body={Details:DetailsObj,
userId:this.userId,appNo:this.appNo,isAddFlag:this.isAddFlag,isDelete:this.isDelete,claimId:this.claimId};

    this.customerService.addNomDetailInDB(body).subscribe(res =>
   {
this.Detailid = res['generatedId'];

this.disabledBtn=false;
     this.commonFunction.openDialog(res['data'],  '');

   });
 }

  /*update Claim details and submits the case*/
 addDetails1()
 {
 var DetailsObj = this.Details1.getRawValue(); // returns only the value of fields which are enabled and skips the
value of disabled fields.
this.Details1.addControl('_id', new FormControl(this.Detailid));
this.Details1.patchValue({_id: this.Detailid});
var                                                         body={Details:DetailsObj,
userId:this.userId,appNo:this.appNo,isAddFlag:this.isAddFlag,isDelete:this.isDelete,claimId:this.claimId};

    this.customerService.addNom1DetailInDB(body).subscribe(res =>
   {
this.Detailid = res['generatedId'];
this.disabledBtn=false;
     this.commonFunction.openDialog(res['data'],  '');

   });
 }

  /*Updates customer details into database and accepts the claim*/
 updateDetails()
 {
 var DetailsObj = this.Details1.getRawValue(); // returns only the value of fields which are enabled and skips the value of disabled fields.
this.Details1.addControl('_id', new FormControl(this.Detailid));
this.Details1.patchValue({_id: this.Detailid});
var body={Details:DetailsObj, userId:this.userId,appNo:this.appNo,isAddFlag:this.isAddFlag,isDelete:this.isDelete,claimId:this.claimId};
   this.customerService.getNomineeDetailsFromDB(body).subscribe(res=>{
      var dataObj=JSON.stringify(res["data"]);
      var obj=JSON.parse(dataObj);
      var sub=JSON.parse(dataObj);
      var len=(obj.length-1);
      for(var i=0;i<=len;i++)
      {
      var
body1={appNo:obj[i]['appNo'],name:obj[i]['name'],username:obj[i]['name'],password:obj[i]['name'],email:obj[i]['Email'],gender:obj[i]['Gender'],teamId:this.teamId,userId:this.userId};
      console.log("body1--->"+body1);
          this.customerService.updateUserInDB(body1).subscribe(res =>
             {
                 this.Detailid = res['generatedId'];

                 this.disabledBtn=false;
                      this.commonFunction.openDialog(res['data'],  '');
```

```
            });
      }
       this.customerService.updateDetailInDB(body).subscribe(res =>
         {
                this.Detailid = res['generatedId'];

                this.disabledBtn=false;
                    this.commonFunction.openDialog(res['data'],  '');

         });
        });

 }

  /*Updates customer details into database and accepts the claim*/
 reject()
 {
 var DetailsObj = this.Details1.getRawValue(); // returns only the value of fields which are enabled and skips the
value of disabled fields.
this.Details1.addControl('_id', new FormControl(this.Detailid));
this.Details1.patchValue({_id: this.Detailid});
var                                                                                    body={Details:DetailsObj,
userId:this.userId,appNo:this.appNo,isAddFlag:this.isAddFlag,isDelete:this.isDelete,claimId:this.claimId};
       this.customerService.reject(body).subscribe(res =>
         {
                this.Detailid = res['generatedId'];

                this.disabledBtn=false;
                    this.commonFunction.openDialog(res['data'],  '');

         });
 }

  /*the function is used to call or open dialogue*/
  private open(content) {
       this.modalService.open(content,
        { centered: true, backdrop: 'static', keyboard: false, backdropClass: 'dark-backdrop'})
          .result.then((result) => {
           width:"100%";
           this.closeResult = `Closed with: ${result}`;
       }, (reason) => {
           this.closeResult = `Dismissed ${this.getDismissReason(reason)}`;
        });
    }

  /*the function is called when the dialogue is to be closed/dismissed*/
    private getDismissReason(reason: any): string {
        if (reason === ModalDismissReasons.ESC) {
            return 'by pressing ESC';
        } else if (reason === ModalDismissReasons.BACKDROP_CLICK) {
            return 'by clicking on a backdrop';
        } else {
            return  `with: ${reason}`;
        }
    }
}
```

customer.service.ts

```
@Injectable({
  providedIn: 'root'
})
export class CustomerService {
```

```
constructor(private _http:Http,private http:HttpClient) { }

/*gets customer's details from the DB*/
 getDetailsFromDB(inputValue)
 {


    return this.http.post("/api/getPerDesFromDB", {
    inputValue
    })
  }

/*gets customer's address from the DB*/
  getAddDetailsFromDB(inputValue)
 {


    return this.http.post("/api/getAddrDesFromDB", {
    inputValue
    })
  }

/*gets nominee details from the DB*/
    addNomDetailInDB(inputValue)
 {
    console.log("cash value "+JSON.stringify(inputValue));
    return this.http.post("/api/addNomDetailinDB", {
    inputValue
    })
  }

/*update nominee details*/
   addNom1DetailInDB(inputValue)
 {
    console.log("cash value "+JSON.stringify(inputValue));

    return this.http.post("/api/addNom1DetailinDB", {

    inputValue

    })

  }

/*gets customer's details from the DB*/

     getDetails1FromDB(inputValue)

 {


    return this.http.post("/api/getSum1DesFromDB", {

    inputValue

    })

  }

/*gets customer's details from the DB*/

   getDetailsDB(inputValue)

 {


    return this.http.post("/api/getSum", {

    inputValue

    })

  }

/*gets customer's address from the DB*/

   getAddress(inputValue)

 {


    return this.http.post("/api/Address", {

    inputValue

    })

  }
```

```
/*gets Nominee Details from the DB*/
  getNominee(inputValue)
 {


    return this.http.post("/api/Nominee", {
    inputValue
    })
  }

/*gets Bank details from the DB*/
   getBank(inputValue)
 {


    return this.http.post("/api/Bank", {
    inputValue
    })
  }

/*gets nominee details from the DB*/
    getNomineeDetailsFromDB(inputValue)
 {


    return this.http.post("/api/getNomineeDetailsFromDB", {
    inputValue
    })
  }

/*accept claim*/
     updateDetailInDB(inputValue)
 {
      return this.http.post("/api/updateDetailinDB", {
      inputValue
      })
    }
   /*reject claim*/
       reject(inputValue)
   {

      return this.http.post("/api/reject", {
      inputValue
      })
    }

  /*creates new user*/
      updateUserInDB(inputValue)
   {
      return this.http.post("/api/updateUserinDB", {
      inputValue
      })
    }
}
```

docdisp.component.html

```
<div class="container-fluid max-height">
  <div class="max-height max-width fullscreen" [ngxToggleFullscreen]="commonFunction.fullscreen"
id="mainblock">
    <div class="row m-0 h-10">
      <div class="col-12 p-0">
        Application Number <b>{{appNo}}</b>
      </div>
      <div class="col-12 small-font mt-2 p-0">
      Date <b>{{dateCreated}}</b>
      &nbsp;&nbsp;&nbsp;
      Name <b>{{name}}</b>
      &nbsp;&nbsp;&nbsp;
      Application Type <b>Type1</b>
    </div>
  </div>
  <hr class="mb-0 border-top-1-5" />
    <div class="row m-0 height-80">
    <div class="col-xl-2 col-lg-3 col-md-3 col-sm-3 vertical-line p-0 overflow-y h-90">
      <ul class="list-group sortable img-view-list" >
        <li class="list-group-item pl-3"
        [ngClass]="{'active': selectedItem == item, 'highlight':highlightedItem == item}"
        (mouseover)="changeStyle($event, item)" (mouseleave)="changeStyle($event, item)"
        *ngFor="let item of disp let i=index" (click)="displayImg($event, item)"
        (drop)="onItemDrop($event, item)">
        <span><a>{{item}}</a></span>
<!--        <button mat-button class="image-delete-btn float-right"
                (click)="deleteImageFromDB($event, item, i)"></button> -->
        </li>
      </ul>
    </div>
    <div class="col-xl-10 col-lg-9 col-md-9 col-sm-9 p-2 h-100">
      <div class="row m-0 p-1 pb-2 icon-btn-border-bottom" >
        <!-- *ngIf="disp.length > 0" -->
        <div class="col-9 p-0 d-flex align-items-center">
          <div class="d-inline-flex pr-2 small-vertical-line-right">
            <button mat-button class="zoomIn-btn p-0" (click)="zoomIn()"></button>
          </div>
          <div class="d-inline-flex pl-2 pr-2 small-vertical-line-right">
            <button mat-button class="zoomOut-btn p-0" (click)="zoomOut(0)"></button>
          </div>
          <div class="d-inline-flex pl-2 pr-2 small-vertical-line-right">
            <button mat-button class="rotate-anti-clockwise-btn p-0" (click)="rotateClockwise(-90)"></button>
          </div>
          <div class="d-inline-flex pl-2 pr-2 small-vertical-line-right">
            <button mat-button class="rotate-clockwise-btn p-0" (click)="rotateClockwise(90)"></button>
          </div>
          <!-- <div class="d-inline-flex pl-2 pr-2 small-vertical-line-right">
            <img src="../../../assets/images/crop-enable.png" width="16" height="16" />
          </div> [sortablejs]="disp" [sortablejsOptions]="clone1Options"-->
          <div class="d-inline-flex pl-2">
            <button mat-button class="zoom-btn p-0" (click)="commonFunction.toggleFullscreen()"></button>
          </div>
        </div>
        <div class="col-3 pl-2 p-0 d-flex align-items-center justify-content-end">
          <div class="d-inline-flex pr-2 small-vertical-line-right">
            <button mat-button class="prevImage-btn p-0" (click)="prevImage('')" [disabled]="imageIndexList ==
      0"></button>
          </div>
          <div class="d-inline-flex pl-2">
            <button mat-button class="nextImage-btn p-0" (click)="nextImage('')"
            [disabled]="imageIndexList == lastImgIndex"></button>
          </div>
        </div>
      </div>
      <div class="img-container overflow-y h-85" id="draggable" name="draggable"
      (dragover)="onDragOver($event)">
          <img src={{imagetoShow[imageIndexOne]}} [ngStyle]="style" id="imgViewer" name="imgViewer"
          (dragstart)="onDragStart($event)" class="img-viewer-size" />
```

```
          <div></div>




        </div>

      <!-- Action Buttons -->

      </div>
    </div>
    <div class="container-fluid max-height" id="footer"  style="float: right;">
                              <button type="button" class="custom-btn btn-border mr-2 white" (click)="commonFunction.open(comment);" mat-stroked-button>Comment</button>&nbsp;&nbsp;&nbsp;
        <button  type="button" class="custom-btn btn-border mr-2 white"  [routerLink]="['/docview']" mat-stroked-button>Go Back</button>&nbsp;&nbsp;&nbsp;
        <button type="button" mat-stroked-button class="custom-btn-btn btn-border white" [disabled]="disable" style="margin-left: 15px;" (click)="commonFunction.open(need);">Need More Documents</button>&nbsp;&nbsp;&nbsp;
         <button type="button" mat-stroked-button class="custom-btn red btn-border submit-btn mr-2" (click)="convertHTMLToPDf()" [routerLink]="['/docview']">Accept</button>
</div>

<!-- <ng-template #content let-c="close" let-d="dismiss" class="scale50"  style="position:relative">
  <div class="modal-header border-0 " id="mydiv" >
    <div class="row m-0">

      <div class="col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12">
        <div class="d-inline-flex pr-2 small-horizontal-line-right" >
      <textarea placeholder="Enter Comments" required style="width: 100%;height: 9em"></textarea>
    </div>
```

```
      </div>

      <div class="col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12">
      </div>

    <br>
    <br>
      <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6">
            <div class="d-inline-flex pr-2 small-vertical-line-right">
             <button mat-button class="submit-btn btn-border mr-2 p-0" (click)="updateDetails()"
[routerLink]="['/summary']" style="width: 100%">Accept</button>
      </div>
        </div>
          <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6">
             <div class="d-inline-flex pr-2">
             <button type="button" mat-button class="submit-btn  btn-border mr-2 p-0" [routerLink]="['/summary']"
style="width: 100%">Reject</button>
            </div>
        </div>



    </div>
         <button type="button" class="close " aria-label="Close" (click)="d('Cross click')" >
        <span aria-hidden="true">&times;</span>
      </button>

  </div>
</ng-template>
 -->

<ng-template #comment let-c="close" let-d="dismiss" class="scale50"  style="position:relative">
  <div class="modal-header border-0 " id="mydiv" >
```

```
    <div class="row m-0">
    <form class="max-height col-12 p-0" role="form" [formGroup]="Detail" (submit)='addDetails()'
method="POST" autocomplete="off" novalidate>
   <!--   <div class="col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12"> -->
        <div class="txt d-inline-flex pr-2 small-horizontal-line-right" >

                                    <mat-form-field>
                                    <textarea matInput placeholder="Enter Comments" formControlName="com"
required style="width: 100%;height: 3em"></textarea>
              </mat-form-field>

    </div>
      <!-- </div>

      <div class="col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12">
      </div> -->

    <br>
    <br>
<!--      <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6"> -->
           <div class="d-inline-flex pr-2 small-vertical-line-left" id="foot">
            <button type="button" mat-stroked-button class="submit-btn btn-border mr-2 p-0 red"
(click)="addDetails()" style="float:right;width: 100%">submit</button>
      </div>
   <!--     </div> -->
        <!--  <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6">
            <div class="d-inline-flex pr-2">
            <button type="button" mat-button class="submit-btn  btn-border mr-2 p-0" [routerLink]="['/summary']"
style="width: 100%">Reject</button>
          </div>
       </div> -->
      </form>


    </div>
         <button type="button" class="close " aria-label="Close" (click)="d('Cross click')" >
        <span aria-hidden="true">&times;</span>
      </button>

  </div>
</ng-template>
<ng-template #need let-c="close" let-d="dismiss" class="scale50"  style="position:relative">
  <div class="modal-header border-0 " id="mydiv" >
    <div class="row h-100 m-0">
     <form class="max-height p-0" role="form" [formGroup]="Details" (submit)='addDetails()' method="POST"
#myForm="ngForm" autocomplete="off" novalidate>
  <!--     <div class="col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12"> -->
         <div class="txt d-inline-flex pr-2 small-horizontal-line-right" >


                            <mat-form-field>
                                      <textarea  matInput  placeholder="Enter  Comments"  formControlName="comments"
required ></textarea>
                                    </mat-form-field>


                             </div>
<!--       </div> -->

 <!--      <div class="col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12">
      </div> -->

    <br>
    <br>
<!--       <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6"> -->
             <div class="d-inline-flex pr-2 small-vertical-line-left" id="foot">
```

```
              <button type="button" mat-stroked-button class="submit-btn btn-border mr-2 p-0 red" (click)="addDetails()" [routerLink]="['/summary']" style="float:right; width: 100%">Submit</button>
      </div>
       <!--  </div> -->
   <!--        <div class="col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6">
              <div class="d-inline-flex pr-2">
              <button type="button" mat-button class="submit-btn  btn-border mr-2 p-0" [routerLink]="['/summary']" style="width: 100%">Reject</button>
            </div>
        </div> -->
        </form>


    </div>
         <button type="button" class="close " aria-label="Close" (click)="d('Cross click')" >
        <span aria-hidden="true">&times;</span>
      </button>

  </div>
</ng-template>
```

docdisp.component.css

```
.img-viewer-size{
  display: block;
  margin-left: auto;
  margin-right: auto;
  width: 50%;
}

li { background: green; }
li:nth-child(odd) { background: red; }
.img-container{
```

```
                height: 25em !important;
                overflow-x: auto;
                overflow-y: auto;
}
.red{
 height: 2em;
 text-align: center;
    line-height: 29px;
}

/* Darker background on mouse-over */
.red:hover {
  background-color: #DF2406;
}
.white:hover {
  background-color: #D8D8D8;
}
#footer {
    position: fixed;
    bottom: 2em;
    left: 45em;
    z-index: 100;
}
mat-form-field{
    width: 27em !important;
}
#foot{
                /*position: fixed;*/
    bottom: 15em;
    left: 25em;
    z-index: 100;
```

```
}
::selection{
    text-shadow: none;
    background: #FE2E64;
    color: #fff;
}
textarea{
width: 100% !important;
height: 8em !important;
}
value of disabled fields.
//this.Detail.addControl('_id', new FormControl(this.Detailid));
//this.Detail.patchValue({_id: this.Detailid});
console.log("COMMENT- Function->"+this.Detail.get('com').value);

var body={Detail:DetailsObj, userId:this.userId,appNo:this.appNo,name:this.name};

    this.docdispService.addComments(body).subscribe(res =>
   {
this.Detailid = res['generatedId'];

this.disabledBtn=false;
     this.commonFunction.openDialog(res['data'],  '');


   });

}
imagetoShow:any=[];
oldImage:any=[];
originalSize: boolean = true;
saveBtn:boolean=false;
```

```
exportBtn:boolean=true;

selectedItem="";

selectedItemSecondList="";

highlightedItem="";

hovering: boolean = false;

secondListItem="";

imageName="";

imageNameListOld:any=[];

imageNameList:any=[];

rotation:number=0;


number=0;


comment:string="";

  /*displays Image*/

  displayImg(event, newValue)

  {

  this.imagetoShow=[];

  this.imagetoShow=this.oldImage;

  console.log("left side------->"+this.imageIndexOne);

  this.lastImgIndex=this.disp.length-1;

  this.exportedImg=false;

  this.isDisabled=false;

  this.commonFunction.spinner.show();

  console.log("New value ----->disp img--->"+newValue);

  this.showCropper=false;

 this.isHide=true;

  this.croppedImage=[];

this.saveBtn=false;

this.exportBtn=true;
```

```
//this.imageNameList=[];
this.selectedItemSecondList="";
this.reset();
this.navbarOpen=false;
    var nm=newValue;
    this.imageName=newValue;
    var body={filename:newValue, appNo:this.appNo,name:this.name};
    if(newValue!=undefined){
   if(this.imageNameListOld.indexOf(newValue)==-1){


    console.log(this.disp.length);




  /*gets Image*/
    this.docdispService.getimage(body).subscribe(res=>
    {
      var dataObj= JSON.stringify(res)
      var obj=JSON.parse(dataObj);
      console.log("obje data--->"+obj.length);
      for (var sub in obj )
      {
        var objSub = obj[sub];
        var elementsArray = Object.keys(objSub);

       this.oldImage.push('data:image/png;base64,'+obj[sub]['b64']);
       this.imageNameListOld.push(obj[sub]['originalName']);
       this.imagetoShow=this.oldImage;
       this.isCommentUpdate=false;
       if(obj[sub]['comment']!="" && obj[sub]['comment']!=null){
       this.comment=obj[sub]['comment']
       this.isCommentUpdate=true;
```

```
          }
         }

         //this.createImageFromBlob(newObj[0]['b64'], newValue);

         this.selectedItem = newValue;

         this.imageIndexOne=this.imageNameListOld.indexOf(newValue);
         this.imageIndexList=this.disp.indexOf(newValue);
         this.getCommentFromDB('');
   this.commonFunction.spinner.hide();



       },error=>
       {

         console.log(error);
       });



     }else{



      this.selectedItem = newValue;
      this.imageIndexOne=this.imageNameListOld.indexOf(newValue);
      this.imageIndexList=this.disp.indexOf(newValue);
      this.getCommentFromDB('');
      this.commonFunction.spinner.hide();
     }
   }else{
     this.commonFunction.spinner.hide();
```

```
}

 }

 /*resets dimensions of image back to original*/
 private reset() {
   this.scale = 1;
   this.rotation = 0;
   this.translateX = 0;
   this.translateY = 0;
   this.updateStyle();
   var img = document.getElementById('imgViewer');
  console.log("isHide-------->"+img);
     if(img!=null){
     img.removeAttribute('style');
     }

     //this.brightnessRange=60;
 }
 navbarOpen = false;

 toggleNavbar() {
   this.navbarOpen = !this.navbarOpen;
 }
 commentId:string="";

 /*gets comment's from the DB*/
 getCommentFromDB(isProperty){
 this.commentId="";
 this.comment="";
 this.isCommentUpdate=false;
```

```
  var body={filename:this.imageName, appNo:this.appNo};
  this.docdispService.getComment(body).subscribe(res=>
    {
    var dataObj= JSON.stringify(res['data'])
      var obj=JSON.parse(dataObj);

if(obj!="" && obj.length!=0){
var draggable = document.getElementById('draggable');
  draggable.setAttribute('style', 'border-bottom: 1px solid rgba(182, 182, 182, 0.5)');

if(isProperty=="imgProperty"){

 if(obj[0].brightnessValue!="" && obj[0].brightnessValue!=null && obj[0].brightnessValue!=0){
  var img = document.getElementById('imgViewer');
  img.setAttribute('style', 'filter:brightness(' + obj[0].brightnessValue + '%)');
 }
 }
  this.commentId = obj[0]._id;
      this.comment=obj[0].comment;
       this.isCommentUpdate=true;

      }
    },error=>
    {

      console.log(error);
    });
  }
  private prevX: number;
  private prevY: number;

  /*update dimension of the image*/
```

```
    private updateStyle() {
    this.style.transform = `translate(${this.translateX}px, ${this.translateY}px) rotate(${this.rotation}deg)
scale(${this.scale})`;
    this.style.msTransform = this.style.transform;
    this.style.webkitTransform = this.style.transform;
    this.style.oTransform = this.style.transform;
  }

  /*enables dragging of image from one place to another*/
 onDragStart(evt) {
    evt.dataTransfer.setDragImage(evt.target.nextElementSibling, 0, 0);
    this.prevX = evt.clientX;
    this.prevY = evt.clientY;
  }

  /*changes the location of the image*/
  onDragOver(evt) {
    this.translateX += (evt.clientX - this.prevX);
    this.translateY += (evt.clientY - this.prevY);
    this.prevX = evt.clientX;
    this.prevY = evt.clientY;
    this.updateStyle();
  }

  /*enables zooming in of the image*/
zoomIn(){

  var myImg = document.getElementById("imgViewer");
        var currWidth = myImg.clientWidth;
        var currHeight=myImg.clientHeight;
        if(currWidth == 2500 && currHeight) return false;
         else{
            myImg.style.width = (currWidth + 100) + "px";
             myImg.style.height = (currHeight + 100) + "px";
        }
        }



  /*enables zooming out of the image*/
  zoom: number = 0.5;
  zoomOut(amount:number)
  {
  var myImg = document.getElementById("imgViewer");
        var currWidth = myImg.clientWidth;
        var currHeight=myImg.clientHeight;
        if(currWidth == 100 && currHeight==100) return false;
     else{
            myImg.style.width = (currWidth - 100) + "px";
             myImg.style.height = (currHeight -100) + "px";
        }
  }

  /*calls the next image*/
nextImage(event) {
  //console.log("imageIndexList---------->"+this.imageIndexList+"--------"+this.lastImgIndex);
    if (this.imageIndexOne < this.disp.length - 1) {
       this.imageIndexList=this.disp.indexOf(this.imageName);

      this.imageIndexOne=this.imageIndexOne+1;
      this.imageIndexList=this.imageIndexList+1;
```

```
   this.isHide=true;
this.croppedImage=[];
  this.showCropper=false;



     /* if(this.imageNameList.length>0){
       this.imageName=this.imageNameList[this.imageIndexOne];
       this.selectedItemSecondList=this.imageName;
       this.selectedItem="";
      }else{
      this.imageName=this.imageNameListOld[this.imageIndexOne];
      this.selectedItem=this.imageName;
      this.selectedItemSecondList="";
      }*/
       this.reset();
      if(this.exportedImg){
        //if (this.imageIndexOne < this.imagesNew.length - 1){
        this.lastImgIndex=this.imagesNew.length-1;
          this.displayImgSecondList('', this.imagesNew[this.imageIndexList]);
        //}
      }else{
      this.lastImgIndex=this.disp.length-1;
        this.displayImg('',this.disp[this.imageIndexList]);
      }




    }
  }
```

```
/*calls the previous image*/
prevImage(event) {


    this.isHide=true;
    this.croppedImage=[];
    this.showCropper=false;




    this.reset();
    this.imageIndexOne=this.imageIndexOne-1;
    if(this.exportedImg){
     this.imageIndexList=this.imagesNew.indexOf(this.imageName);

        this.imageIndexList=this.imageIndexList-1;
        this.lastImgIndex=this.imagesNew.length-1;
        this.displayImgSecondList('', this.imagesNew[this.imageIndexList]);

    }else{
     this.imageIndexList=this.disp.indexOf(this.imageName);

        this.imageIndexList=this.imageIndexList-1;

        this.lastImgIndex=this.disp.length-1;
        this.displayImg('',this.disp[this.imageIndexList]);
    }



}
```

```
  /*rotates image in clockwise direction*/
    rotateClockwise(amount: number){

   this.rotation =this.rotation+amount;
  this.updateStyle();

    }

  /*gets the brightness range of the image*/
brightnessRange:number=0;
 getfilters() {
    return this._sanitizer.bypassSecurityTrustStyle('brightness('+ this.brightnessRange +'%)')
}

  /*increases the brightness of the image*/
increaseBrightness(amount: number){
  this.brightnessRange=this.brightnessRange+10;

  var img = document.getElementById('imgViewer');


        img.setAttribute('style', 'filter:brightness(' + this.brightnessRange + '%)');
        /* -webkit-filter:brightness(' + this.brightnessRange + '); -moz-filter:brightness(' + this.brightnessRange +
')');*/


         //return this._sanitizer.bypassSecurityTrustStyle('brightness('+ this.brightnessRange +'%)')

}


  /*change the style of the image*/
  changeStyle($event, newValue)
  {
    this.highlightedItem = $event.type == 'mouseover' ? newValue : false;
  }


  /*change the style of the image in the secondlist*/
  changeStyleSecondList($event, newValue)
  {
    this.secondListItem = $event.type == 'mouseover' ? newValue : false;
  }

  /*performs a certain action when the object is dropped*/
  onItemDrop(event, newValue){
console.log("Item name--->"+newValue);
var imgCommentValue=$('.commentForImg').val();
if(imgCommentValue==""     ||     imgCommentValue==[]     ||     imgCommentValue==null     ||
imgCommentValue==undefined){
imgCommentValue="";
}

  /*save image details in db*/
var     body={filename:newValue,     rotation:this.rotation,     pathValue:'export',     appNo:this.appNo,
imgComment:imgCommentValue,     brightnessRange:this.brightnessRange,     commentId:this.commentId,
commentValue:this.comment};
this.docdispService.saveImageInDB(body).subscribe(res => {


   var dataObj=res;
      console.log("obje data--->"+dataObj['name']);
      this.saveImageSeqInDB("", "");
```

```
});



}

  /*save mage sequence in db*/
saveImageSeqInDB(fromAction, imgName){

console.log("fromAction----------"+fromAction);
var deleteStatus="";
if(fromAction=="delete"){
 deleteStatus="delete";
}
 var   body={filename:this.imagesNew,   appNo:this.appNo,   isUpdateFlag:this.isUpdateFlag,verifiedImages:[],
deleteStatus:deleteStatus, delImgName:imgName};
    this.docdispService.imgSequenceDB(body).subscribe(res => {


    if(fromAction=="export"){

    if(this.imagesNew.length<=this.applicationFormCount){
    this.addExpandClass('applicationForm');
    }else if(this.imagesNew.length<=this.siForm){
    this.addExpandClass('siForm');
    }else if(this.imagesNew.length<=this.uniqueID){
    this.addExpandClass('uniqueId');
    }else if(this.imagesNew.length<=this.drivingLicense){
    this.addExpandClass('dl');
    }else if(this.imagesNew.length<=this.passport){
    this.addExpandClass('passport');
    }else if(this.imagesNew.length<=this.otherID){
    this.addExpandClass('otherId');
    }
    this.imageIndexOne=this.imageIndexOne+1
    this.imageIndexList=this.imageIndexList+1;
    this.displayImg('', this.disp[(this.imageIndexList)]);





    //this.displayImgSecondList("", imgName);
    }else if(fromAction=="save"){
      this.displayImgSecondList("", imgName);
    }
this.isUpdateFlag=true;
//alert(res['data']);
if(fromAction=="export"){

this.commonFunction.openDialog("Image Export successfully",  '');


//setTimeout(this.dataVerification(imgName), 6000);
}else if(fromAction=="delete"){
this.commonFunction.openDialog("Delete Image successfully",  '');
}else{
this.commonFunction.openDialog(res['data'],  '');
}
});
}

dataVerification(imgName){
var body={filename:imgName, appNo:this.appNo, isUpdateFlag:this.isUpdateFlag,verifiedImages:[]};
```

```
this.docdispService.callAiVerificationApi(body).subscribe(res => {




});
}

  /*provides expansion facility*/
showMenu: string = '';
addExpandClass(element: any) {
        if (element === this.showMenu) {
            this.showMenu = '0';
        } else {
            this.showMenu = element;
        }
    }
    newImagetoShow:any=[];
  displayImgSecondList(event, newValue)
  {
  this.imagetoShow=[];
  this.imagetoShow=this.newImagetoShow;
  console.log("right side------->"+this.imageIndexOne);
  this.lastImgIndex=this.imagesNew.length-1;
  this.exportedImg=true;
  this.isDisabled=false;
this.commonFunction.spinner.show();
this.saveBtn=true;
this.exportBtn=false;
//this.imageNameListOld=[];
this.selectedItem="";
this.reset();

this.navbarOpen=false;

      var nm=newValue;

    this.imageName=newValue;

    var body={filename:newValue, appNo:this.appNo};

    //var body={filename:imageFormArray, appNo:this.appNo};


   if(this.imageNameList.indexOf(newValue)==-1){




    this.docdispService.getNewimage(body).subscribe(res=>

    {

      var dataObj= JSON.stringify(res)

      var obj=JSON.parse(dataObj);


      for (var sub in obj )

      {

        var objSub = obj[sub];

        var elementsArray = Object.keys(objSub);

       this.newImagetoShow.push('data:image/png;base64,'+obj[sub]['b64']);

       this.imageNameList.push(obj[sub]['originalName']);

       this.imagetoShow=this.newImagetoShow;


      }


      //this.createImageFromBlob(newObj[0]['b64'], newValue);

      this.selectedItemSecondList = newValue;

      this.imageIndexOne=this.imageNameList.indexOf(newValue);

      console.log("imageIndexOne---------->"+this.imageIndexOne);
      this.imageIndexList=this.imagesNew.indexOf(newValue);
       this.getCommentFromDB('imgProperty');
      this.commonFunction.spinner.hide();
    },error=>
    {

      console.log(error);
    });



  }else{

   this.selectedItemSecondList = newValue;
   this.imageIndexOne=this.imageNameList.indexOf(newValue);
    this.imageIndexList=this.imagesNew.indexOf(newValue);
   this.getCommentFromDB('imgProperty');
   this.commonFunction.spinner.hide();
  }





  }
count:number=0;
  /*opens up the dialog*/
   private open(content) {
       this.modalService.open(content,
        { centered: true, backdrop: 'static', keyboard: false, backdropClass: 'dark-backdrop'})
          .result.then((result) => {
           width:"100%";
           this.closeResult = `Closed with: ${result}`;
      }, (reason) => {
          this.closeResult = `Dismissed ${this.getDismissReason(reason)}`;
       });
   }

 /*closes the dialog*/
   private getDismissReason(reason: any): string {
       if (reason === ModalDismissReasons.ESC) {
           return 'by pressing ESC';
       } else if (reason === ModalDismissReasons.BACKDROP_CLICK) {
           return 'by clicking on a backdrop';
       } else {
           return  `with: ${reason}`;
       }
   }

 /*creates a pdf*/
convertHTMLToPDf()
 {


     console.log("in function to convert html to pdf..");
     const dialogRef = this.dialog.open(CustomDialogComponent, {
     data: {message: 'Do you really want to accept case?', btn1:'No', btn2:'Yes'}
   });
       dialogRef.afterClosed().subscribe(result => {
     if (result === 2) {
         var amount=40000;
     var total=36000;
     var bod={name:this.name};
     this.docdispService.getAppNo(bod).subscribe(res =>{
                                               var dataObj=JSON.stringify(res["data"])
```

```
                                        var obj=JSON.parse(dataObj);

                                        var application=obj[0]['appNo'];

                                        var len=obj.length-1;

    var body={appNo:application,amt:amount,total:total,name:this.name};

                                this.docdispService.getPolicy(body).subscribe(res =>{

                                                var dataObj=JSON.stringify(res["data"])

                                        var obj=JSON.parse(dataObj);

                                        var policy=obj[0]['PolicyNo'];

                                        var len=obj.length-1;

                                        for(var i=0;i<=len;i++)

                                        {

                                        var holder=obj[i]['name'];

            var sub=obj[i]['personaldetails'];

            for(var j=0;j<sub.length;j++)

            {

            var na=sub[j]['name'];

            }

                                        }

                                        var

body={appNo:application,amt:amount,total:total,name:this.name,policy:policy,holder:holder,na:na};

                                    this.docdispService.convertHtmlToPdf(body).subscribe(res => {

                                      var filename=res["data"];

           this.commonFunction.openDialog("File has been converted", "");


            console.log(filename);

            var bod={filename:filename,name:this.name,policy:policy,appNo:application};

            this.docdispService.downloads(bod).subscribe(res =>

            {

                    console.log("response---->"+res);

                    this.count++;
                    var a="claimdoc";

                    var c=this.count.toString();

                    var b=a.concat(c);

                     var filename=b.concat(".pdf");

                     saveAs(res,filename);


                    //this.spinner.hide();

            });

                         /*console.log("after service call...data "+res["data"]);


                           var fileNameList = JSON.stringify(res["data"]);


                           var fileNameObj = JSON.parse(fileNameList);

                      */



                           },

                           error=>

                           {

                             console.log(error);

                           });

                           });

                           });


var body={userId:this.userId,appNo:this.appNo,reqType:this.reqType,name:this.name};

  this.docdispService.updateAccept(body).subscribe(res =>

                          {

                         this.commonFunction.openDialog(res['data'],  '');

                                        });

                                        //this.router.navigate(['/dashboard'], navigationExtras)

                                       }

                                         /*else

                                         {
                                                this.router.navigate(['/reqdisp4'], navigationExtras)
                                                } */
                                               });


  }
}
```

docdisp.service.ts

```
import { Injectable } from '@angular/core';
import { Http, Headers, RequestOptions } from '@angular/http';
import {map} from 'rxjs/operators'
import { Observable} from 'rxjs/Observable';
import { HttpClient, HttpClientModule ,HttpHeaders,HttpRequest} from '@angular/common/http';
import { Response } from '@angular/http';
import { Subject } from 'rxjs/Subject';
@Injectable({
  providedIn: 'root'
})
export class DocdispService {
result:any;
  constructor(private _http:Http,private http:HttpClient) { }

  /*gets File list from directory*/
  getFileListFromDirectory(body)
  {
 /* return this._http.get("/api/getList", {params: {
        appNo: appNo


      }
      })
      .pipe(map(result => this.result = result.json().data));
```

```
    */

    return this.http.post('/api/getFileList',body)
      .pipe(map((result: Response) => this.result = result));



 }

/*gets Image from directory*/
getimage(body)
{



    return this.http.post('/api/getImage',body)
      .pipe(map((result: Response) => this.result = result));


 }

/*gets application number from the DB*/
 getAppNo(inputValue)
 {

    return this.http.post("/api/getAppNo", {
    inputValue
    })
  }

/*gets userId from the DB*/
     getUser(inputValue)
{
    return this.http.post("/api/getUser", {
```

```
    inputValue
    })
}

/*gets name from the DB*/
  getName(inputValue)
{
    return this.http.post("/api/getName", {
    inputValue
    })
}

/*gets policy number from the DB*/
  getPolicy(inputValue)
{
    return this.http.post("/api/getPolicy", {
    inputValue
    })
}

/*creates a pdf*/
     convertHtmlToPdf(inputValue)
 {
    console.log("in service function to generate pdf...");
    return this.http.post("/htmlPdf/createpdf", {
    inputValue
    })
  }

/*Approving the case*/
     updateAccept(inputValue)
 {
      return this.http.post("/api/updateAccept", {
      inputValue
      })
    }

  /*adding the comments into the db*/
    addComments(inputValue)
    {
    return this.http.post("/api/addComments",{
    inputValue
    })
    }
       downloads(inputValue)
   {

    return this.http.get('/api/downloadAFile/', {
        params:{filename:inputValue['filename'],name:inputValue['name'],policy:inputValue['policy'],appNo:input
Value['appNo']},responseType:"blob"

       //  search: // query string if have
       })



       /* return this.http.post('/api/downloadFile','')
        .pipe(map((result: Response) => this.result = result));
*/

    }


   getComment(body)
```

```
{



   return this.http.post('/api/getComment',body)
     .pipe(map((result: Response) => this.result = result));



 }


  getCommentedFileList(body)
{



   return this.http.post('/api/getCommentedFileList',body)
     .pipe(map((result: Response) => this.result = result));



 }

 getNewimage(body)
{
  /*return this.http.post('/api/downloadNewImage',body,{
      responseType : 'blob',
      headers:new HttpHeaders().append('Content-Type','application/json')
     });*/
       console.log(" in service function to get new image "+JSON.stringify(body));
      return this.http.post('/api/downloadNewImage',body)
     .pipe(map((result: Response) => this.result = result));
 }
```

```
  getNewPDFimage(body)
{
  /*return this.http.post('/api/downloadNewPDFImage',body,{
       responseType : 'blob',
       headers:new HttpHeaders().append('Content-Type','application/json')
      });*/
        console.log(" in service function to get new image "+JSON.stringify(body));
       return this.http.post('/api/downloadNewPDFImage',body)
      .pipe(map((result: Response) => this.result = result));
 }

 saveImageInDB(imagetoShow){

return this.http.post("/api/uploadImage", {
    imagetoShow
  })




 }


   imgSequenceDB(imagetoShow){

    return this.http.post("/api/imgSequenceDB", {
    imagetoShow
  })




 }
```

```
  serSaveCommentInDB(imagetoShow){

     return this.http.post("/api/saveCommentDB", {
     imagetoShow
   })
 }

 callAiVerificationApi(imagetoShow){

     return this.http.post("/api/callAiVerificationApi", {
     imagetoShow
   })
 }
}
```

docview.component.html

```
<div class="container-fluid h-100 w-100">
<div class="row m-0 h-100 w-100">
<div class="h-100 w-100 col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12 text-center">
<div *ngIf="div" class="div1 h-100 w-100">
  <div class="h-100 w-100 col-sm-6 col-md-6 col-lg-6 col-xl-6 col-6 text-right">
 <mat-form-field class="filterClass">
<input matInput (keyup)="applyFilter($event.target.value)" placeholder="Search">
</mat-form-field>
 </div>
</div>
<div class="row m-0 w-100 h-80 text-left">
 <div class="w-100 col-sm-12 col-md-12 col-lg-12 col-xl-12 col-12 p-0 h-100">
<div class="mat-elevation-z8 example-container h-100 table-striped ">
<div class="w-100 scroll h-100">
<table mat-table [dataSource]="dataSource" id="mytable" matSort>
                                            <!-- ID Column -->
```

```
    <ng-container matColumnDef="id">
   <th mat-header-cell *matHeaderCellDef mat-sort-header> No. </th>
 <td mat-cell *matCellDef="let row; let i = index;"> {{i+1}} </td>
       </ng-container>
          <ng-container matColumnDef="NomineeDateOfBirth">
<th mat-header-cell *matHeaderCellDef mat-sort-header>UserId</th>
  <td mat-cell *matCellDef="let row" > {{row.userId}} </td>
 </ng-container>
      <ng-container matColumnDef="appNo">
             <th mat-header-cell *matHeaderCellDef mat-sort-header> App No </th>
              <td mat-cell *matCellDef="let row"> {{row.appNo}} </td>
             </ng-container>
      <ng-container matColumnDef="creationDate" >
             <th mat-header-cell *matHeaderCellDef mat-sort-header>RequestType</th>
                <td mat-cell *matCellDef="let row"> {{row.requestType}} </td>
</ng-container>

                                               <!-- Progress Column -->
             <ng-container matColumnDef="NomineeName">
            <th mat-header-cell *matHeaderCellDef mat-sort-header> Name</th>
             <td mat-cell *matCellDef="let row" > {{row.name}} </td>
             </ng-container>
                                        <!-- Name Column -->
            <ng-container matColumnDef="PolicyNo">
            <th mat-header-cell *matHeaderCellDef mat-sort-header style="display: none;">filename</th>
             <td mat-cell *matCellDef="let row" style="display: none;"> {{row.imageName}} </td>
             </ng-container>
                                        <!-- Color Column -->
              <ng-container matColumnDef="DateOfBirth" >
            <th mat-header-cell *matHeaderCellDef mat-sort-header>Image View</th>
```

```
<td mat-cell *matCellDef="let row" >
                                <button  mat-button (click)="taskView(row)" class="view-btn"></button>
                                                        </td>
                                                                </ng-container>
                <tr mat-header-row *matHeaderRowDef="displayedColumns; sticky: true"></tr>
<tr mat-row *matRowDef="let row; columns: displayedColumns; let i = index; " >
                                                                </tr>
                                                              </table>
                                                              </div>
              <mat-paginator [pageSizeOptions]="[5, 10, 25, 100]" class="mat-paginator-sticky"></mat-paginator>
                                                            </div>
                                                          </div>
                                                        </div>

                                         </div>
                                         </div>
                           </div>
```

docview.component.css

```
.btn {
  background-color: DodgerBlue; /* Blue background */
  border: none; /* Remove borders */
  color: white; /* White text */
  padding: 12px 16px; /* Some padding */
  font-size: 1em; /* Set a font size */
  cursor: pointer; /* Mouse pointer on hover */
}

/* Darker background on mouse-over */
.btn:hover {
  background-color: RoyalBlue;
}
```

```
.scale50 {
    -webkit-transform: scale(3.0);
    -moz-transform: scale(3.0);
    transform: scale(3.0);
}

.modal-xl .modal-lg {
  max-width: 100%;
}

.modal-header{
  overflow-x: scroll !important;
  overflow-y: scroll !important;
}
input[type=file]{
    width:100px;
    color:transparent;
}
.scroll{
  overflow: auto;
}
.imgViewer{
  width: 100%;
}
div.mat-form-field-infix {
    display: block;
    position: relative;
    flex: auto;
    min-width: 0;
    width: 15em;
}
```

```
/*div.example-container {
  // height: 400px;
  overflow-y: auto;
}*/

::selection {
    text-shadow: none;
    background: #FE2E64;
    color: #fff;
}
/*.button{


}*/
tr{
  text-align: left;
}
 th.mat-header-cell {
    padding-left: 2em !important;
    text-align: left;
}
td.mat-cell{
     padding-left: 2em !important;
  text-align: left;
}
div.mat-elevation-z8{
  box-shadow: none!important;
}
 th.mat-header-cell {
    padding-left: 7em !important;
    text-align: left !important;
}
```

```
td.mat-cell{
  padding-left: 6em !important;
  text-align: left !important;
}
.red{
 height: 2em;
 text-align: center;
    line-height: 29px;
}

/* Darker background on mouse-over */
.red:hover {
  background-color: #DF2406;
}
/*h4{
  text-align: center;
}*/

.edit-btn{
  background: url('../../../assets/images/edit.png');
  min-width: 15px !important;
  height: 13px !important;
  border-right: 1.5px solid rgba(182, 182, 182, 0.5) !important;
}

.view-btn{
  background: url('../../../assets/images/view.png');
  min-width: 35px !important;
  height: 11px !important;
}

.edit-btn, .view-btn{
  background-repeat: no-repeat;
  background-position: center;
  background-size: contain;
}
```

claim.component.ts

```
export interface UserData {
  id: string;
  name: string;
  progress: string;
  color: string;
}

@Component({
  selector: 'app-docview',
  templateUrl: './docview.component.html',
  styleUrls: ['./docview.component.css']
})
export class DocviewComponent implements OnInit {
closeResult: string;
@Input()
  config: ImageViewerConfig;


 @Output()
  configChange: EventEmitter<ImageViewerConfig> = new EventEmitter();
  subscription: Subscription;
displayedColumns: string[] = ['id','NomineeDateOfBirth', 'creationDate', 'appNo', 'NomineeName', 'PolicyNo',
'DateOfBirth'/*,                                                    'NomineeDateOfBirth',
'NomineeGender','NomineeHouseNo','NomineeBuildingName','NomineeLocality'          ,'NomineeStreet',
'NomineeLandmark',            'NomineeCity',            'NomineeState',            'NomineeCountry',
'NomineePinCode','NomineeMobileNo','NomineeEmail','NomineeBank',    'NomineeBranch',    'NomineeIFSC',
'NomineeAccountNo', 'NomineeAccountType'*/];
  dataSource: MatTableDataSource<UserData>;
  @ViewChild(MatPaginator) paginator: MatPaginator;
  @ViewChild(MatSort) sort: MatSort;
  lab:string;
  //isAddFlag="add";
  isUpdateFlag=false;
  show=false;
  public loading = false;
//public urlPDF:string="https://vadimdez.github.io/ng2-pdf-viewer/assets/pdf-test.pdf";
imgURL: any;
public items=[];
public names=[];
public user=[];
userId: string;
teamId: string;
public n:string;
   btn1: boolean=false;
 label:boolean=true;
appNo:string="";
public taskListStatusValue:number=0;
 public isProgressValue: number= 0
 public callReason:string;
public progressStatus: string="";
table:boolean=false;
 tasklist:Array<any>;
  disp:Array<any>;
   roleId:string;
   agentCode:String;
   dateCreated:string;
    isprogress:string;
     private translateX = 0;
  private translateY = 0;
  rotation:number=0;
  private scale = 1;
  originalSize: boolean = true;
saveBtn:boolean=false;
exportBtn:boolean=true;
  imageChangedEvent: any = '';
  Details1 = new FormGroup({
    claim:new FormControl({value:''},Validators.required)
  });
  Details = new FormGroup({
    claim:new FormControl({value:''},Validators.required)
  });
  croppedImage: any = '';
  showCropper = false;
    fileList:Array<any>;
    isCommentUpdate=false;
  title: string;
  msg: string;
  reqType:string;
  imgType:string;
  exportedImg:boolean=false;
  Detailid:string="";
  isDisabled:boolean=true;
  public policyNo: string;
  disable:boolean=true;
  images = [
  ];

  imagesNew = [



  ];

  uploadedImages = [



  ];
  navbarOpen = false;

  toggleNavbar() {
    this.navbarOpen = !this.navbarOpen;
  }
  public filename:string;
  constructor(public router: Router, private route: ActivatedRoute,private fb: FormBuilder, public
commonFunction:CommonFunction, private dialog: MatDialog , private sharedDataService: DataService, private
http: HttpClient,@Optional() @Inject('config') public moduleConfig: ImageViewerConfig,private modalService:
NgbModal,private datePipe: DatePipe,public docviewService:DocviewService,private spinner:
NgxSpinnerService) { this.subscription = docviewService.subj$.subscribe(val=>{
    console.log(val);


     this.loadData(val);
     });
     this.route.queryParams.subscribe(params => {


this.isProgressValue=Number(params["inprogress"]);
this.callReason=params["callreason"];
```

```
         if(params["inprogress"]=="AllRecord"){

            this.progressStatus="AllRecord";

            this.ngOnInit();

         }else{

            this.progressStatus="";

            this.ngOnInit();

         }




        });}

   ngOnInit() {

     var txt = JSON.parse(localStorage.getItem('usersinfo'));


this.userId=txt[0]["_id"];


 this.teamId=(txt[0]["teamId"]);

 this.roleId=(txt[0]["roleId"]);

   this.Details1 = this.fb.group

({

     claim:[' ',Validators.required]

});

this.Details = this.fb.group

({

     claim:[' ',Validators.required]

});

  /*gets Application number from the DB*/

 var body={"userId":this.userId};

 this.docviewService.getAppFromDB(body).subscribe(res =>{

     var dataObj=JSON.stringify(res["data"])

     var obj=JSON.parse(dataObj);
```

```
    var len=(obj.length-1);
     console.log("len---->"+len);
    if(obj!="" && obj!=[] && obj!=null){
    console.log("object valu--->"+obj[0]["appNo"])
      this.appNo=obj[0]["appNo"];
      console.log("App No--->"+this.appNo);

      var body={appNo:this.appNo};

   var                                              body1={userId:this.userId,                                              teamId:this.teamId,
inprogress:[this.isProgressValue],appNo:this.appNo,imgType:this.imgType};
        body1={userId:this.userId,                                                                                          teamId:this.teamId,
inprogress:[0,1,2],appNo:this.appNo,imgType:this.imgType};
        this.loadData(body1);
        }},error=>
    {


      console.log(error);
    });
  }

  /*sends data from one component to another*/
 editData(selectedItem: any){
  //console.log("selectedItem"+selectedItem.appNo);
  console.log(selectedItem.name);
  this.docviewService.send(selectedItem);
  var
body={filename:selectedItem.imageName,appNo:selectedItem.appNo,reqType:selectedItem.imgType,name:sel
ectedItem.name};
  this.preview(body);
}
public style = { transform: '', msTransform: '', oTransform: '', webkitTransform: '' };
```

```
commentId:string="";
comment:string="";
selectedItem="";
selectedItemSecondList="";
highlightedItem="";
hovering: boolean = false;
secondListItem="";
imageName="";
imageNameListOld:any=[];
imageNameList:any=[];
imagetoShow:any=[];
oldImage:any=[];
imageIndexOne:number = 0;
  imageIndexList:number = 0;
  lastImgIndex:number=0;
isHide=true;
public imagePath;


  public message: string;

  /*displays a preview of the document that has been submitted*/
  preview(body) {
    this.docviewService.displayImage(body).subscribe(res =>
      {

          var dataObj= JSON.stringify(res)
              var obj=JSON.parse(dataObj);
              console.log("response---->"+obj[0]['b64']);

              this.imagetoShow='data:image/png;base64,'+obj[0]['b64'];
              console.log("imagetoShow--------->"+this.imagetoShow);
              });
  }


  /*opens a dialogue*/
 private open(content) {
       this.modalService.open(content,
        { centered: true, backdrop: 'static', keyboard: false, backdropClass: 'dark-backdrop'})
          .result.then((result) => {
           width:"100%";
           this.closeResult = `Closed with: ${result}`;
       }, (reason) => {
           this.closeResult = `Dismissed ${this.getDismissReason(reason)}`;
        });
    }

  /*closes/dismisses the dialogue*/
    private getDismissReason(reason: any): string {
        if (reason === ModalDismissReasons.ESC) {
            return 'by pressing ESC';
        } else if (reason === ModalDismissReasons.BACKDROP_CLICK) {
            return 'by clicking on a backdrop';
        } else {
            return  `with: ${reason}`;
        }
    }
url:any = '';

  /*gets activated when the customer changes or selects a file*/
    onSelectFile(event,file: HTMLInputElement) {
    if (event.target.files && event.target.files[0]) {
      var reader = new FileReader();
      this.lab=event.target.files[0].name;
      this.filename=event.target.files[0].name;

      console.log("filename-->"+this.filename);

      reader.readAsDataURL(event.target.files[0]); // read file as data url


      reader.onload = (event) => { // called once readAsDataURL is completed

        this.url = reader.result;

       this.btn1=true;


      }

    }

  }


  /*update the style of the image*/

    private updateStyle() {

    this.style.transform    =    `translate(${this.translateX}px,    ${this.translateY}px)    rotate(${this.rotation}deg)
scale(${this.scale})`;

    this.style.msTransform = this.style.transform;

    this.style.webkitTransform = this.style.transform;

    this.style.oTransform = this.style.transform;

  }


  /*allows the user to zoomin to the picture*/

zoomIn(){


  var myImg = document.getElementById("imgViewer");

        var currWidth = myImg.clientWidth;

        var currHeight=myImg.clientHeight;

        if(currWidth == 2500 && currHeight) return false;

         else{

            myImg.style.width = (currWidth + 100) + "px";

             myImg.style.height = (currHeight + 100) + "px";

        }
        }




  /*allows the user to aoom out of the picture*/

  zoom: number = 0.5;

  zoomOut()

  {

  var myImg = document.getElementById("imgViewer");

        var currWidth = myImg.clientWidth;

        var currHeight=myImg.clientHeight;

        if(currWidth == 100 && currHeight==100) return false;

     else{

            myImg.style.width = (currWidth - 100) + "px";

             myImg.style.height = (currHeight -100) + "px";

        }

  }

  /*loads data onto the table*/

  loadData(body){

   this.docviewService.getImageDetailFromDB(body)

        .subscribe(res => {

       /*    if(res['data'].length==0)

      {

      this.table=false;

      }

      else{

      this.table=true;

      }

       */

        console.log("res['data']--->"+res['data']);
```

```
  var dataObj= JSON.stringify(res['data'])
 var obj=JSON.parse(dataObj);

/* for (var sub in obj )
 {
  var objSub = obj[sub];
  var elementsArray = Object.keys(objSub);


  console.log("tasklist response---->"+obj[sub]['inprogress']);
  if(Number(this.teamId)==5 || Number(this.teamId)==4){

  }else{
  if(obj[sub]['inprogress']==1){
  obj[sub]['inprogress']="Work In Progress"
  }else if(obj[sub]['inprogress']==0){
  obj[sub]['inprogress']="New Task"
  }else if(obj[sub]['inprogress']==2){
  obj[sub]['inprogress']="Rescaned Cases"
  }
  }

 }*/

 this.tasklist=obj;
 console.log("data----->"+this.tasklist);
 const users = Array.from(this.tasklist);
 this.disable=false;
 /*if(users.length>=8)
 {
   this.disable=false;
 }
 else
```

```
      {
      this.disable=true;
      }*/
    this.dataSource = new MatTableDataSource(users);
    this.dataSource.paginator = this.paginator;
    this.dataSource.sort = this.sort;
 });

}

  /*gets customers name from the database*/
name()
{
    var x=0;
              var body={};
               this.docviewService.getUser(body).subscribe(res =>{
                            var dataObj=JSON.stringify(res["data"])
              var obj=JSON.parse(dataObj);
              var len=(obj.length-1);
                  for(var i=0;i<=len;i++)
                  {
                    this.user[i]=obj[i];
                    console.log("users-->"+this.user[i]);
                    var body1={user:this.user[i]};
          this.docviewService.getName(body1).subscribe(res =>{
                                          var data=JSON.stringify(res["data"])
                            var ob=JSON.parse(data);
                                  this.names[x]=ob[0]['name'];
                                  console.log("name-->"+this.names[x]);
                                  x++;
                  });
                  }
});
}


  /*enables the linking of two components that is allows to move from component 1 to component 2 on click of a
button in this case component docdisp is called*/
   taskView(selectedItem: any){
 var dataObj= JSON.stringify(selectedItem)


      var obj=JSON.parse(dataObj);




  var body={appNo:selectedItem.appNo, isprogress:1, teamId:this.teamId};
    this.docviewService.updateProgressStatus(body).subscribe(res =>
  {
   let navigationExtras: NavigationExtras = {
            queryParams: {
                "appNo": selectedItem.appNo,
                "userId": selectedItem.userId,
                "reqType":selectedItem.requestType,
                "name":selectedItem.name
            },
            skipLocationChange: true
        };
         console.log("selected name-->"+selectedItem.Name);

    var txt = JSON.parse(localStorage.getItem('usersinfo'));
    this.userId=txt[0]["_id"];
    var teamId=(txt[0]["teamId"]);
```

```
      this.router.navigate(['/docdisp'], navigationExtras)

  });
}
}
```

docview.service.ts

```
@Injectable({
  providedIn: 'root'
})
@Injectable({
  providedIn: 'root'
})
export class DocviewService {
   result:any;
  private sub = new Subject();
  subj$ = this.sub.asObservable();

  private subpopup = new Subject();
  subpopupj$ = this.subpopup.asObservable();
  constructor(private _http:Http,private http:HttpClient) { }


  /*gets application number from db*/
       getAppFromDB(inputValue)
   {


      return this.http.post("/api/getAppFromDB", {
      inputValue
      })
    }

  /*gets application number from db*/
        getAppNo(inputValue)
   {

      return this.http.post("/api/getAppNo", {
      inputValue
      })
    }

  /*gets claim Details from the DB*/
         getClaimDetailFromDB(inputValue)
   {

      return this.http.post("/api/getClaimDetailFromDB", {
      inputValue
      })
    }

  /*gets image details from the DB*/
      getImageDetailFromDB(inputValue)
  {
      return this.http.post("/api/getImageDetailFromDB", {
      inputValue
      })
  }

  /*gets userId from the DB*/
      getUser(inputValue)
  {
      return this.http.post("/api/getUser", {
```

```
    inputValue
    })
}

/*gets customer's name from the DB*/
  getName(inputValue)
{
    return this.http.post("/api/getName", {
    inputValue
    })
}

/*gets policy no from the DB*/
  getPolicy(inputValue)
{
    return this.http.post("/api/getPolicy", {
    inputValue
    })
}

/*displays the image wanted*/
displayImage(inputValue)
 {

  /*return this.http.get('/api/displayImage/', {
      responseType:"blob"


     //  search: // query string if have
     })*/

     return this.http.post('/api/dispImage',inputValue)
      .pipe(map((result: Response) => this.result = result));
```

```
        /* return this.http.post('/api/downloadFile','')
         .pipe(map((result: Response) => this.result = result));
*/

     }

   /*display image or downloads image*/
      downloads(inputValue)
    {

     return this.http.get('/api/downloadClaimFile/', {
         responseType:"blob"


        //  search: // query string if have
        })



        /* return this.http.post('/api/downloadFile','')
         .pipe(map((result: Response) => this.result = result));
*/

     }

   /*creates pdf*/
      convertHtmlToPdf(inputValue)
    {
       console.log("in service function to generate pdf...");
       return this.http.post("/htmlPdf/createpdf", {
       inputValue
       })
    }

  send(value: string) {
    this.sub.next(value);
  }

  openScanningTeamPopup(value: string) {
    this.subpopup.next(value);
  }

  sendData(value: string) {
    this.sub.next(value);
  }

updateProgressStatus(body)
  {

      return this.http.post('/api/updateProgressStatus',body)
        .pipe(map((result: Response) => this.result = result));

   }
}
```

4.3.3 Html_pdf.js

```
router.post('/createRequesthistorypdf', async (req,res) =>
{
 console.log("directory path "+req);
dirpath=path.join(__dirname,'../../uploads/pdf_output/request/'+req.body.inputValue['name']+
'/');
 console.log("directory path "+dirpath);
    function mkdirpath(dirPath)
    {
        console.log("in mkdir function dir path"+dirPath);
        if(!fs.existsSync(dirPath))
        {
           console.log("in if.. "+dirPath)
```

```
           try
           {
              console.log("in try.. "+dirPath)
              fs.mkdirSync(dirPath);
           }
           catch(e)
           {
              mkdirpath(path.dirname(dirPath));
              mkdirpath(dirPath);
              console.log("in catch.. "+dirPath)
           }
        }
     }
      mkdirpath(dirpath);
     (async function()
     {
        var imgArray = [];
        try
        {
           var fn="request";
           var filename=fn.concat(req.body.inputValue['count']);
           filename=filename.concat(".pdf");
           console.log(filename);
           const browser = await puppeteer.launch();
           const page = await browser.newPage();
await  page.setContent(`<html>`+req.body.inputValue['htmlContent']+`</html>`);
           await page.emulateMedia('screen');

filepath=path.join(__dirname,'../../uploads/pdf_output/request/'+req.body.inputValue['name'])
+ '/' + filename;
           console.log("file name with path "+filepath);
           await page.pdf({
              printBackground: true,
              path: filepath
           });
           console.log("done");
           await browser.close();
```

4.3.4 Freq.bat

```
@ECHO OFF
java -jar "C:\Users\Admin\Documents\work\editfreq.jar"
```

4.3.5 ConnectToDb.java

```
package trial;
```

```
import com.mongodb.client.FindIterable;
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;
import com.mongodb.client.model.Filters;
import com.mongodb.client.model.Updates;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import org.bson.Document;
import org.json.JSONException;
import org.json.JSONObject;
import com.mongodb.BasicDBObject;
import com.mongodb.MongoClient;
import com.mongodb.MongoCredential;
public class ConnecttoDB implements Runnable {
  public static void main( String args[] ) throws JSONException {    }
@Override
public void run() {

            Logger mongoLogger = Logger.getLogger( "org.mongodb.driver" );
            mongoLogger.setLevel(Level.SEVERE);
  MongoClient mongo = new MongoClient( "localhost" , 27017 );
  MongoCredential credential;
  credential = MongoCredential.createCredential(" ", "PolicyServicing",
    "".toCharArray());
  System.out.println("Connected to the database successfully");
  MongoDatabase database = mongo.getDatabase("PolicyServicing");
  MongoCollection<Document> collection = database.getCollection("product");
  System.out.println("Collection sampleCollection selected successfully");
  BasicDBObject andQuery = new BasicDBObject();
  List<BasicDBObject> obj = new ArrayList<BasicDBObject>();
  obj.add(new BasicDBObject("isUpdate", " "));
  obj.add(new BasicDBObject("Edited Value",new BasicDBObject("$ne"," ")));
  andQuery.put("$and", obj);
  Object b = new ArrayList();
  System.out.println(andQuery.toString());
  FindIterable<Document> iterDoc = collection.find(andQuery);
  int i = 0;
```

```
  Iterator it = iterDoc.iterator();
  while (it.hasNext()) {
    JSONObject x;
        try {
                x = new JSONObject(((Document) it.next()).toJson());
                 System.out.println("x-------->"+x.get("Term"));
            String Edit=x.getString("Edited Value");
            String pay=x.getString("PaymentFrequency");
            System.out.println(pay);
            System.out.println("Edited Value"+Edit);
            collection.updateOne(Filters.eq("isUpdate","    "),    Updates.set("PaymentFrequency",
Edit));
            collection.updateOne(Filters.eq("isUpdate"," "), Updates.set("Edited Value", pay));
            collection.updateOne(Filters.eq("isUpdate"," "), Updates.set("isUpdate", "Done"));
             System.out.println("Document updated successfully...");
            } catch (JSONException e) {
                        e.printStackTrace();
            }
   i++;

   }
   System.out.println("Length of doc---->"+i);
}
}
var  filePath=path.resolve(__dirname,'../../uploads/pdf_output/request/'+req.body.inputValue['
name'] + '/' + filename);
                 response.data=filename;
            res.json(response);

        } catch (e){
            console.log("error ..", e);
            res.send(e.message);
        }
      })();
```