Sprint-2 Plan

Employee Detail Management

1. Establishing Project Folder Architecture
2. Server App Initialization
3. **Sprint-2 Execution** [2(a), **2(b)**, 2(c), 2(d), 2(e), 2(f)]
4. Completing Sprint-1 Objectives

2(a) Employee Module - Analysis, Design & DB-Preparation

Data Design, UI Design, Sample Data, Database Preparation

2(b) Employee View

1. Controller Design
2. Coding-Server App
   - Version-1 (Client Pagination)
     - Entity Coding, DAO Coding, Controller Coding, Testing
   - Version-2 (Server Pagination)
     - Controller Modification, Testing
3. Coding-Client App
   - Version-1 (Client Pagination)
     - UI Preparation, Entity Coding, Service Coding, Controller Coding, Testing
   - Version-2 (Server Pagination)
     - Service Modification, Controller Modification, Testing

2(c) Employee Search
2(d) Employee Insert
2(e) Employee Update
2(f) Employee Delete

==========

2(b) Employee View

1. Controller Design
2. Coding Server-App
3. Coding Client-App **Version-1 (Client Pagination)**
   1. Client App Preparation for Sprint-2
   2. UI Preparation with Static Data & Testing
   3. Coding Entity Classes
   4. Data Loading with Dynamic Data & Testing
   5. Coding Service Classes for Dynamic Data & Testing (With & Without DI)
   6. Error Correction – Server App – Employee Data Service
   7. Data loading from Server & Testing, Promise-Pending/Resolve/Reject Status
   8. Client Pagination
   9. Client Search – (i) Normal (ii) Using Template (a) Direct Property (b) Processed
   10. **Formal Method Architecture**
   11. Code Explanation
   12. Code Modification

Version-2 (Server Pagination)

==========

Formal Method Architecture – View and Client Search

Discuss following

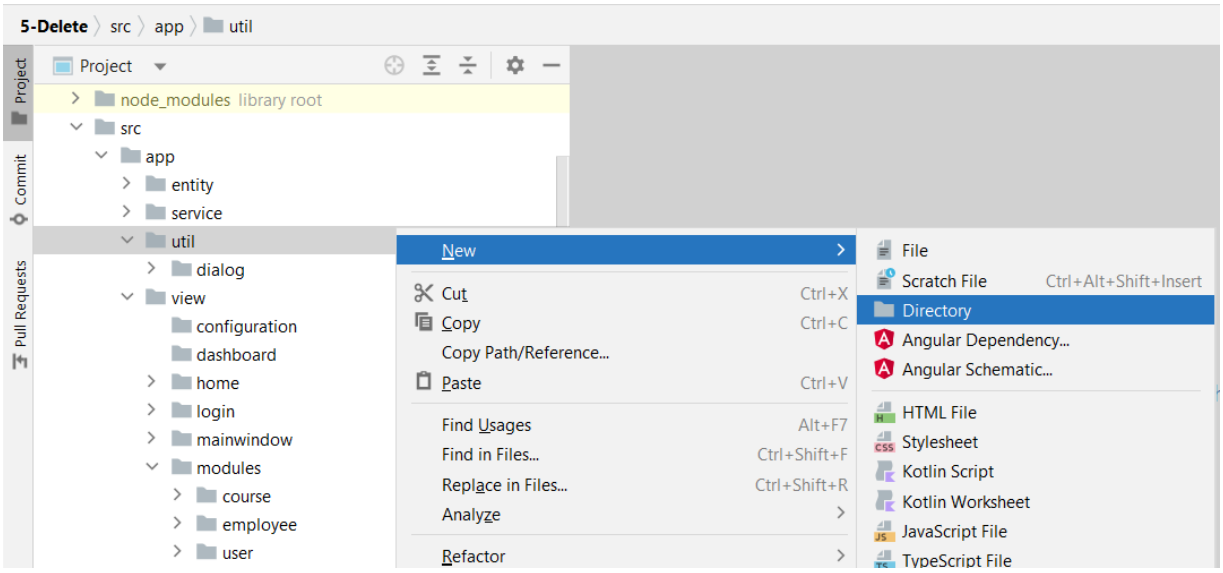
Routing Mechanism of Angular Framework
Life Cycle of Angular App
Life Cycle Hook
Compare “ngOnInit()” with “constructor()”

1. Separating Module Independent Service Codes to Supportive Utility Classes

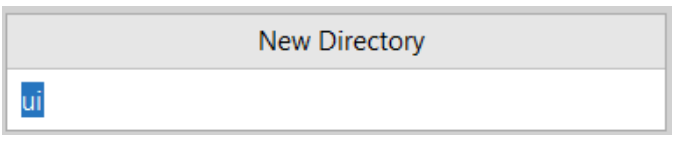
employee.component.ts

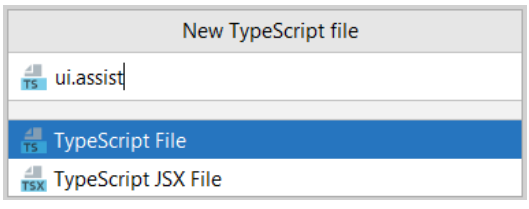
```
getProperty(element: {}, reference: string): string {

  if (!reference.endsWith("()")) {
    const value = reference.split( separator: '.').reduce((o : {} , a : string ) => {
      //@ts-ignore;
      return o[a];
    }, element) as string;
    return value;
  } else {
    reference = reference.substring(0, reference.indexOf('('));
    //@ts-ignore;
    return this[reference](element);
  }

}
```

(a) Create a Package to store UI related Tasks

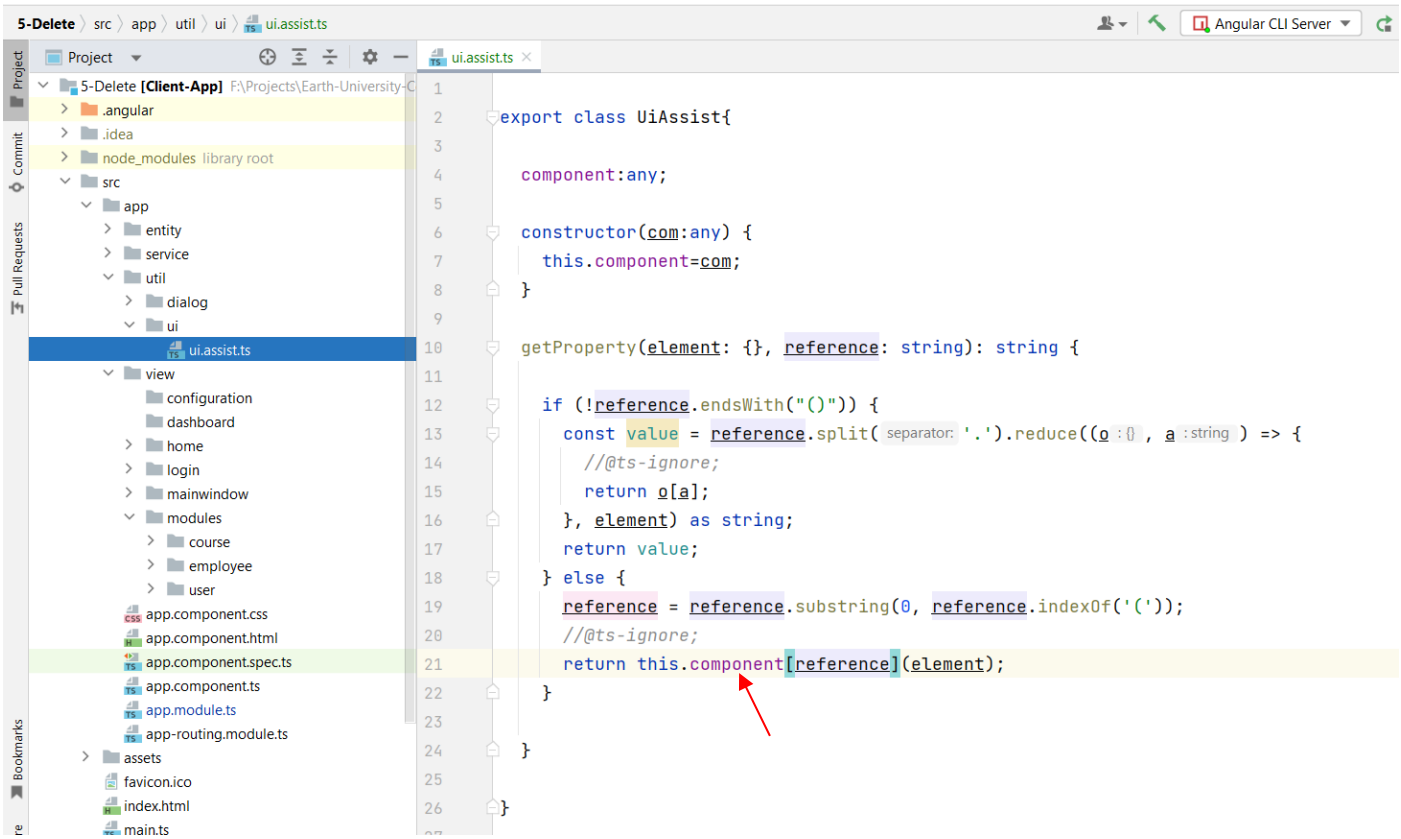


(b) Create a new Folder “ui” inside the “util”
(c) Create a Type Script File “ui.assist” inside the above folder





(d) Create a Class “UiAssist” in the above file as follows
Copy the “getProperty()” method into this class.



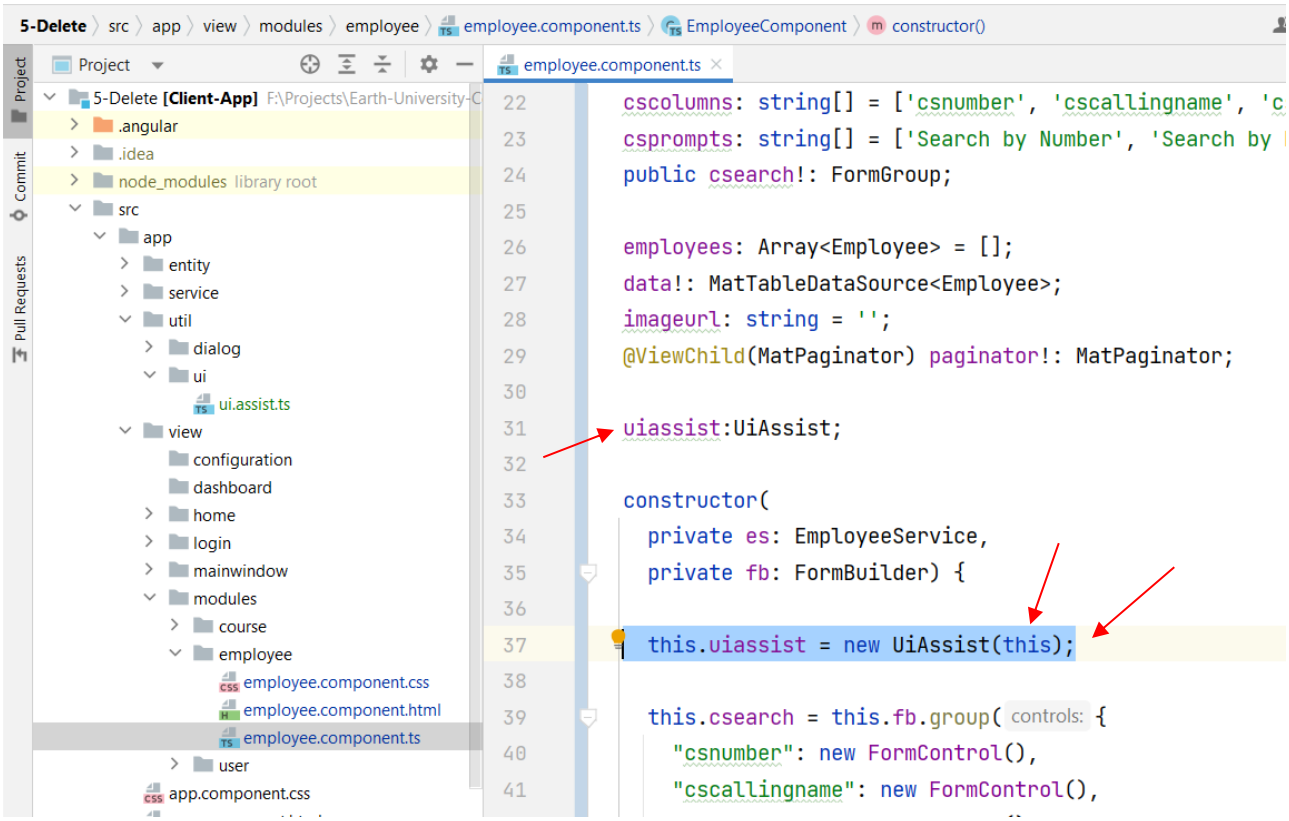

```
export class UiAssist{

  component:any;

  constructor(com:any) {
    this.component=com;
  }

  getProperty(element: {}, reference: string): string {
    if (!reference.endsWith("()")) {
      const value = reference.split('.').reduce((o, a) => {
        //@ts-ignore;
        return o[a];
      }, element) as string;
      return value;
    } else {
      reference = reference.substring(0, reference.indexOf('('));
      //@ts-ignore;
      return this.component[reference](element);
    }
  }

}
```

Discuss the design architecture with “UiAssist” and its content

Ui + Ui-Controller + Ui-Assistent

- Ui – UI Template Codes
- Ui-Controller – Component Specific Codes
- Ui-Assistant – Common UI Supportive Codes

(e) Changes in UI Controller (Every UI-Controller has a UI-Assistant for support within it)
UiAssistent instance must refer the relevant Component(UI-Controller)

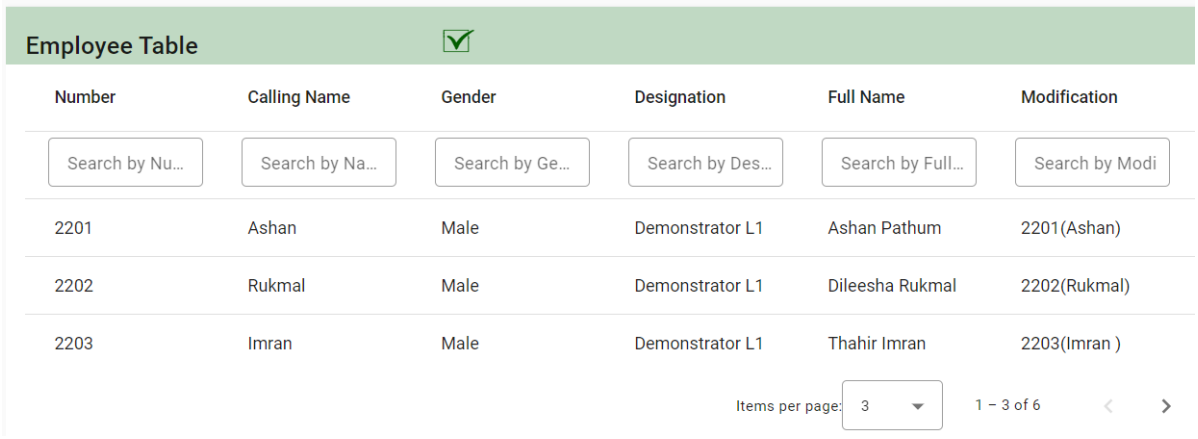


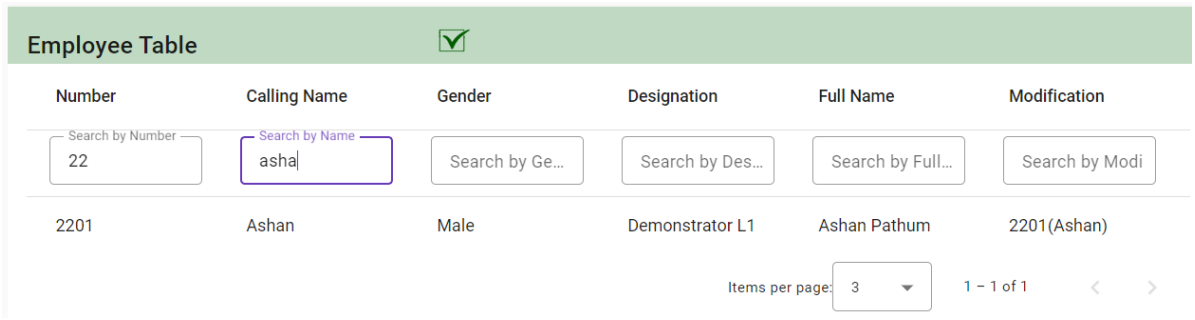
(f) Changes in UI

```
<mat-table [dataSource]="data">

  <ng-container *ngFor="let column of columns;let i = index;" [matColumnDef]="column">
    <mat-header-cell *matHeaderCellDef> {{headers[i]}} </mat-header-cell>
    <mat-cell *matCellDef="let element">{{ uiassist.getProperty(element,binders[i]) }} </mat-cell>
  </ng-container>

  <ng-container *ngFor="let cscolumn of cscolumns;let i = index;" [matColumnDef]="cscolumn">
    <mat-header-cell *matHeaderCellDef>
      <mat-form [formGroup]="csearch" class="csearch">
        <mat-form-field appearance="outline">
          <mat-label>{{csprompts[i]}}</mat-label>
          <input matInput [formControlName]="cscolumn"
                 (keyup)="filterTable()">
        </mat-form-field>
      </mat-form>
    </mat-header-cell>
  </ng-container>

  <mat-header-row  *matHeaderRowDef="columns"></mat-header-row>
  <mat-header-row *matHeaderRowDef="cscolumns"></mat-header-row>
  <mat-row *matRowDef="let row; columns: columns;"></mat-row>

</mat-table>
```

(g) Delete the “getProperty()” function from “EmployeeComponnet”

(h) Observe the Output





2. Formalize Function Architecture of the Ui-Controller making Code Reusable, Maintainable and Readable. Discuss the Structure, Functionality and Interaction of Code Blocks

(a) Ui-Template Codes

Form Section
Search Section

employee.component.html

```
<mat-grid-list [cols]="12" rowHeight="4rem">

 <mat-grid-tile [colspan]="4" [rowspan]="10">
   <mat-card>
     <mat-card-header>
       <mat-card-title>Employee Detail</mat-card-title>
     </mat-card-header>
     <mat-card-content> </mat-card-content>
   </mat-card>
 </mat-grid-tile>

 <mat-grid-tile [colspan]="8" [rowspan]="4">
   <mat-card>
     <mat-card-header>
       <mat-card-title>Employee Search</mat-card-title>
     </mat-card-header>
     <mat-card-content> </mat-card-content>
   </mat-card>
 </mat-grid-tile>
```

View Section

Table

Column Header Template,
Search Header Template,
Row Template

Pagination

```
<mat-grid-tile [colspan]="8" [rowspan]="6">
  <mat-card>
    <mat-card-header>
      <mat-card-title>Employee Table</mat-card-title>
      <div id="loading">
        <img  [src]="imageurl">
      </div>
    </mat-card-header>
```

```
    <mat-card-content>

      <mat-table [dataSource]="data">

        <ng-container *ngFor="let column of columns;let i = index;" [matColumnDef]="column">
          <mat-header-cell *matHeaderCellDef> {{headers[i]}} </mat-header-cell>
          <mat-cell *matCellDef="let element">{{ uiassist.getProperty(element,binders[i]) }} </mat-cell>
        </ng-container>

        <ng-container *ngFor="let cscolumn of cscolumns;let i = index;" [matColumnDef]="cscolumn">
          <mat-header-cell *matHeaderCellDef>
            <mat-form [formGroup]="csearch" class="csearch">
              <mat-form-field appearance="outline">
                <mat-label>{{csprompts[i]}}</mat-label>
                <input matInput [formControlName]="cscolumn"
                       (keyup)="filterTable()">
              </mat-form-field>
            </mat-form>
          </mat-header-cell>
        </ng-container>
```

```

        <mat-header-row  *matHeaderRowDef="columns"></mat-header-row>
        <mat-header-row *matHeaderRowDef="cscolumns"></mat-header-row>
        <mat-row *matRowDef="let row; columns: columns;"></mat-row>

      </mat-table>

      <mat-paginator [length]="20" [pageSize]="3" [pageSizeOptions]="[3,5,10,20]"></mat-paginator>

    </mat-card-content>

  </mat-card>
</mat-grid-tile>

</mat-grid-list>
```

(b) Ui-Controller Codes

Imports

employee.component.ts

```
import {Component, ViewChild} from '@angular/core';
import {Employee} from "../../../entity/employee";
import {EmployeeService} from "../../../service/employeeservice";
import {MatPaginator} from "@angular/material/paginator";
import {MatTableDataSource} from "@angular/material/table";
import {FormBuilder, FormControl, FormGroup} from "@angular/forms";
import {UiAssist} from "../../../util/ui/ui.assist";

```

Angular Declaration

```

@Component({
  selector: 'app-employee',
  templateUrl: './employee.component.html',
  styleUrls: ['./employee.component.css']
})
```

Data for Binding

```

export class EmployeeComponent {

  columns: string[] = ['number', 'callingname', 'gender', 'designation', 'fullname', 'modi'];
  headers: string[] = ['Number', 'Calling Name', 'Gender', 'Designation', 'Full Name', 'Modification'];
  binders: string[] = ['number', 'callingname', 'gender.name', 'designation.name', 'fullname', 'getModi()'];

  cscolumns: string[] = ['csnumber', 'cscallingname', 'csgender', 'csdesignation', 'csname', 'csmodi'];
  csprompts: string[] = ['Search by Number', 'Search by Name', 'Search by Gender',
                                          'Search by Designation', 'Search by Full Name', 'Search by Modi'];
  public csearch!: FormGroup;

  employees: Array<Employee> = [];
  data!: MatTableDataSource<Employee>;
  imageurl: string = '';
  @ViewChild(MatPaginator) paginator!: MatPaginator;

  uiassist:UiAssist;
```

Constructor – Injection and Creation of dependencies

```

  constructor(
    private es: EmployeeService,
    private fb: FormBuilder) {

    this.uiassist = new UiAssist(this);

    this.csearch = this.fb.group( controls: {
      "csnumber": new FormControl(),
      "cscallingname": new FormControl(),
      "csgender": new FormControl(),
      "csdesignation": new FormControl(),
      "csname": new FormControl(),
      "csmodi": new FormControl(),
    });

  }
```

Creating 3 UI-Sections (View/Search/Form) by Property Adjustments of

```
ngOnInit() {
  this.initialize();
}

initialize() {
  this.createView();
  this.createSearch();
  this.createForm();
}

createView() {
  this.imageurl = 'assets/pending.gif';
  this.loadTable();
}

createSearch(){ }

createForm(){ }
```

Data Lording for

Table – Main Entity

Form – Combos – Supportive Entities

```
loadForm(){ }


loadTable() {

  this.es.getAll() Promise<Employee[]>
    .then((emps: Employee[]) => {
      this.employees = emps;
      this.imageurl = 'assets/fullfilled.png';
    }) Promise<void>
    .catch((error) => {
      console.log(error);
      this.imageurl = 'assets/rejected.png';
    }) Promise<void>
    .finally( onfinally: () => {
      this.data = new MatTableDataSource(this.employees);
      this.data.paginator = this.paginator;
    });

}
```

Data Pipes for Custom Cell Value Factories for Table Columns

Filtering For Client-Search

```
92
93    getModi(element: Employee) {
94      return element.number + '(' + element.callingname + ')';
95    }
96
97
98    filterTable(): void {
99
100     const cserchdata = this.csearch.getRawValue();
101
102     this.data.filterPredicate = (employee: Employee, filter: string) => {
103       return (cserchdata.csnumber == null || employee.number.toLowerCase().includes(cserchdata.csnumber)) &&
104         (cserchdata.cscallingname == null || employee.callingname.toLowerCase().includes(cserchdata.cscallingname)) &&
105         (cserchdata.csgender == null || employee.gender.name.toLowerCase().includes(cserchdata.csgender)) &&
106         (cserchdata.csdesignation == null || employee.designation.name.toLowerCase().includes(cserchdata.csdesignation)) &&
107         (cserchdata.csname == null || employee.fullname.toLowerCase().includes(cserchdata.csname)) &&
108         (cserchdata.csmodi == null || this.getModi(employee).toLowerCase().includes(cserchdata.csmodi));
109     };
110
111     this.data.filter = 'xx';
112
113   }
114
115 }
```

What is your 3rd Sprint?
Write down Codes required for your Next Module in a Paper.