**ASE LAB-8 Source Code Report LABID:5\_2**

**Sandeep Reddy Salkuti(24)**

**Pranathi Nalluri(22)**

**www:** This code snippet below describes how to activate environment with some port number and creating http server and making it to listen on the port number. Some common error handling messages are displayed on failure of connection

#!/usr/bin/env node

/\*\*

 \* Module dependencies.

 \*/

var app = require('../app');

var debug = require('debug')('mean-angular6:server');

var http = require('http');

/\*\*

 \* Get port from environment and store in Express.

 \*/

var port = normalizePort(process.env.PORT || '3000');

app.set('port', port);

/\*\*

 \* Create HTTP server.

 \*/

var server = http.createServer(app);

/\*\*

 \* Listen on provided port, on all network interfaces.

 \*/

server.listen(port);

server.on('error', onError);

server.on('listening', onListening);

/\*\*

 \* Normalize a port into a number, string, or false.

 \*/

function normalizePort(val) {

  var port = parseInt(val, 10);

  if (isNaN(port)) {

    // named pipe

    return val;

  }

  if (port >= 0) {

    // port number

    return port;

  }

  return false;

}

/\*\*

 \* Event listener for HTTP server "error" event.

 \*/

function onError(error) {

  if (error.syscall !== 'listen') {

    throw error;

  }

  var bind = typeof port === 'string'

    ? 'Pipe ' + port

    : 'Port ' + port;

  // handle specific listen errors with friendly messages

  switch (error.code) {

    case 'EACCES':

      console.error(bind + ' requires elevated privileges');

      process.exit(1);

      break;

    case 'EADDRINUSE':

      console.error(bind + ' is already in use');

      process.exit(1);

      break;

    default:

      throw error;

  }

}

/\*\*

 \* Event listener for HTTP server "listening" event.

 \*/

function onListening() {

  var addr = server.address();

  var bind = typeof addr === 'string'

    ? 'pipe ' + addr

    : 'port ' + addr.port;

  debug('Listening on ' + bind);

}

**App.module.ts:** Below code explains importing all modules and components created and giving routing paths to the components created it no path is given default it routes t book

import { BrowserModule } from '@angular/platform-browser';

import { NgModule } from '@angular/core';

import { FormsModule, ReactiveFormsModule } from '@angular/forms';

import { HttpClientModule } from '@angular/common/http';

import { AppComponent } from './app.component';

import { RouterModule, Routes } from '@angular/router';

import { BookComponent } from './book/book.component';

import { BookDetailComponent } from './book-detail/book-detail.component';

import { BookCreateComponent } from './book-create/book-create.component';

import { BookEditComponent } from './book-edit/book-edit.component';

import { BrowserAnimationsModule } from '@angular/platform-browser/animations';

import {

  MatInputModule,

  MatPaginatorModule,

  MatProgressSpinnerModule,

  MatSortModule,

  MatTableModule,

  MatIconModule,

  MatButtonModule,

  MatCardModule,

  MatFormFieldModule } from '@angular/material';

const appRoutes: Routes = [

  {

    path: 'books',

    component: BookComponent,

    data: { title: 'Book List' }

  },

  {

    path: 'book-details/:id',

    component: BookDetailComponent,

    data: { title: 'Book Details' }

  },

  {

    path: 'book-create',

    component: BookCreateComponent,

    data: { title: 'Create Book' }

  },

  {

    path: 'book-edit/:id',

    component: BookEditComponent,

    data: { title: 'Edit Book' }

  },

  { path: '',

    redirectTo: '/books',

    pathMatch: 'full'

  }

];

@NgModule({

  declarations: [

    AppComponent,

    BookComponent,

    BookDetailComponent,

    BookCreateComponent,

    BookEditComponent

  ],

  imports: [

    RouterModule.forRoot(appRoutes),

    BrowserModule,

    FormsModule,

    ReactiveFormsModule,

    HttpClientModule,

    BrowserAnimationsModule,

    MatInputModule,

    MatTableModule,

    MatPaginatorModule,

    MatSortModule,

    MatProgressSpinnerModule,

    MatIconModule,

    MatButtonModule,

    MatCardModule,

    MatFormFieldModule

  ],

  providers: [],

  bootstrap: [AppComponent]

})

export class AppModule { }

**index.html:** The below code explains basic html code acts as first html page where route tag is defined.

<!doctype html>

<html lang="en">

<head>

<link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">

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

  <meta charset="utf-8">

  <title>MeanAngular6</title>

  <base href="/">

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

  <link rel="icon" type="image/x-icon" href="favicon.ico">

</head>

<body>

  <app-root></app-root>

</body></html>

**App.js:** The below code explains connection to mongodb database using string obtained from db with user and password in it and join all modules to main app.

var createError = require('http-errors');

var express = require('express');

var path = require('path');

var favicon = require('serve-favicon');

var logger = require('morgan');

var mongoose = require('mongoose');

mongoose.connect('mongodb+srv://sandeep:Sunny111@cluster0-lpgnn.mongodb.net/test?retryWrites=true&w=majority')

  .then(() => console.log('connection successful'))

  .catch((err) => console.error(err));

var apiRouter = require('./routes/book');

var app = express();

app.use(logger('dev'));

app.use(express.json());

app.use(express.urlencoded({extended: false}));

app.use(express.static(path.join(\_\_dirname, 'dist/mean-angular6')));

app.use('/books', express.static(path.join(\_\_dirname, 'dist/mean-angular6')));

app.use('/book-details/:id', express.static(path.join(\_\_dirname, 'dist/mean-angular6')));

app.use('/book-create', express.static(path.join(\_\_dirname, 'dist/mean-angular6')));

app.use('/book-edit/:id', express.static(path.join(\_\_dirname, 'dist/mean-angular6')));

app.use('/api', apiRouter);

// catch 404 and forward to error handler

app.use(function (req, res, next) {

  next(createError(404));

});

// error handler

app.use(function (err, req, res, next) {

  // set locals, only providing error in development

  res.locals.message = err.message;

  res.locals.error = req.app.get('env') === 'development' ? err : {};

  // render the error page

  res.status(err.status || 500);

  res.send(err.status);

});

module.exports = app;

**book.component.html:** The below code shows html view of how application will be after landing like with basic header isbn, author, title and cells with some data. Thereafter it routes to book detail showing all details.

<div class="button-row">

  <a mat-raised-button color="primary" [routerLink]="['/book-create']">

    <mat-icon>add</mat-icon>

  </a>

</div>

<div class="bg"></div>

<div class="example-container mat-elevation-z8">

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

    <!--- Note that these columns can be defined in any order.

          The actual rendered columns are set as a property on the row definition" -->

    <!-- Title Column -->

    <ng-container matColumnDef="isbn">

      <th mat-header-cell \*matHeaderCellDef> ISBN</th>

      <td mat-cell \*matCellDef="let element" class="isbn-col"> {{element.isbn}} </td>

    </ng-container>

    <!-- Title Column -->

    <ng-container matColumnDef="title">

      <th mat-header-cell \*matHeaderCellDef> Title</th>

      <td mat-cell \*matCellDef="let element"> {{element.title}} </td>

    </ng-container>

    <!-- Author Column -->

    <ng-container matColumnDef="author">

      <th mat-header-cell \*matHeaderCellDef> Author</th>

      <td mat-cell \*matCellDef="let element"> {{element.author}} </td>

    </ng-container>

    <tr mat-header-row \*matHeaderRowDef="displayedColumns"></tr>

    <tr mat-row \*matRowDef="let row; columns: displayedColumns;" [routerLink]="['/book-details/', row.\_id]"></tr>

  </table>

</div>

**Book.component.ts:** The below code imports services and book component implements OnInit function where on connection it gets book details.

import {Component, OnInit} from '@angular/core';

import {ApiService} from '../api.service';

import {DataSource} from '@angular/cdk/collections';

import {Observable} from 'rxjs';

@Component({

  selector: 'app-book',

  templateUrl: './book.component.html',

  styleUrls: ['./book.component.css']

})

export class BookComponent implements OnInit {

  books: any;

  displayedColumns = ['isbn', 'title', 'author'];

  dataSource = new BookDataSource(this.api);

  constructor(private api: ApiService) {

  }

  ngOnInit() {

    this.api.getBooks()

      .subscribe(res => {

        console.log(res);

        this.books = res;

      }, err => {

        console.log(err);

      });

  }

}

export class BookDataSource extends DataSource<any> {

  constructor(private api: ApiService) {

    super();

  }

  connect() {

    return this.api.getBooks();

  }

  disconnect() {

  }

}

**book-create.component.html:** Below code explains on clicking add button on front page it opens form with some fields like isbn, author, description etc. I user miss any field then save option id disabled and error validation message displays at that particular field.

<div class="bg"></div>

<div class="button-row">

  <a mat-raised-button color="primary" [routerLink]="['/books']">

    <mat-icon>list</mat-icon>

  </a>

</div>

<form [formGroup]="bookForm" (ngSubmit)="onFormSubmit(bookForm.value)">

  <mat-form-field class="example-full-width">

    <input matInput placeholder="ISBN" formControlName="isbn"

           [errorStateMatcher]="matcher">

    <mat-error>

      <span \*ngIf="!bookForm.get('isbn').valid && bookForm.get('isbn').touched">Please enter ISBN</span>

    </mat-error>

  </mat-form-field>

  <mat-form-field class="example-full-width">

    <input matInput placeholder="Title" formControlName="title"

           [errorStateMatcher]="matcher">

    <mat-error>

      <span \*ngIf="!bookForm.get('title').valid && bookForm.get('title').touched">Please enter Book Title</span>

    </mat-error>

  </mat-form-field>

  <mat-form-field class="example-full-width">

    <input matInput placeholder="Author" formControlName="author"

           [errorStateMatcher]="matcher">

    <mat-error>

      <span \*ngIf="!bookForm.get('author').valid && bookForm.get('author').touched">Please enter Book Author</span>

    </mat-error>

  </mat-form-field>

  <mat-form-field class="example-full-width">

    <textarea matInput placeholder="Description" formControlName="description"

              [errorStateMatcher]="matcher"></textarea>

    <mat-error>

      <span \*ngIf="!bookForm.get('description').valid && bookForm.get('description').touched">Please enter Book Description</span>

    </mat-error>

  </mat-form-field>

  <mat-form-field class="example-full-width">

    <input matInput placeholder="Publisher" formControlName="publisher"

           [errorStateMatcher]="matcher">

    <mat-error>

      <span \*ngIf="!bookForm.get('publisher').valid && bookForm.get('publisher').touched">Please enter Publisher</span>

    </mat-error>

  </mat-form-field>

  <mat-form-field class="example-full-width">

    <input matInput placeholder="Published Year" formControlName="published\_year"

           [errorStateMatcher]="matcher">

    <mat-error>

      <span \*ngIf="!bookForm.get('published\_year').valid && bookForm.get('published\_year').touched">Please enter Published Year</span>

    </mat-error>

  </mat-form-field>

  <div class="button-row">

    <button type="submit" [disabled]="!bookForm.valid" mat-raised-button color="primary">

      <mat-icon>save</mat-icon>

    </button>

  </div>

</form>

**book-create.component.ts:** It checks for all fields without null values and if there are any displays as required. After getting all details it posts data to book detail.

import {Component, OnInit} from '@angular/core';

import {Router} from '@angular/router';

import {ApiService} from '../api.service';

import {FormControl, FormGroupDirective, FormBuilder, FormGroup, NgForm, Validators} from '@angular/forms';

@Component({

  selector: 'app-book-create',

  templateUrl: './book-create.component.html',

  styleUrls: ['./book-create.component.css']

})

export class BookCreateComponent implements OnInit {

  bookForm: FormGroup;

  isbn: string = '';

  title: string = '';

  description: string = '';

  author: string = '';

  publisher: string = '';

  published\_year: string = '';

  constructor(private router: Router, private api: ApiService, private formBuilder: FormBuilder) {

  }

  ngOnInit() {

    this.bookForm = this.formBuilder.group({

      'isbn': [null, Validators.required],

      'title': [null, Validators.required],

      'description': [null, Validators.required],

      'author': [null, Validators.required],

      'publisher': [null, Validators.required],

      'published\_year': [null, Validators.required]

    });

  }

  onFormSubmit(form: NgForm) {

    this.api.postBook(form)

      .subscribe(res => {

        let id = res['\_id'];

        this.router.navigate(['/book-details', id]);

      }, (err) => {

        console.log(err);

      });

  }

}

**book-detail.component.html:** This page shows book detail entered by user. It displays with headers as title and description with some fields below them. At the end edit and Delete buttons are given for further operations.

<div class="button-row">

  <a mat-raised-button color="primary" [routerLink]="['/books']"><mat-icon>list</mat-icon></a>

</div>

<mat-card class="example-card">

  <mat-card-header>

    <mat-card-title><h2>{{book.title}}</h2></mat-card-title>

    <mat-card-subtitle>{{book.description}}</mat-card-subtitle>

  </mat-card-header>

  <mat-card-content>

    <dl>

      <dt>ISBN:</dt>

      <dd>{{book.isbn}}</dd>

      <dt>Author:</dt>

      <dd>{{book.author}}</dd>

      <dt>Publisher:</dt>

      <dd>{{book.publisher}}</dd>

      <dt>Publish Year:</dt>

      <dd>{{book.published\_year}}</dd>

      <dt>Update Date:</dt>

      <dd>{{book.updated\_date | date}}</dd>

    </dl>

  </mat-card-content>

  <mat-card-actions>

    <a mat-raised-button color="primary" [routerLink]="['/book-edit', book.\_id]"><mat-icon>edit</mat-icon></a>

    <a mat-raised-button color="warn" (click)="deleteBook(book.\_id)"><mat-icon>delete</mat-icon></a>

  </mat-card-actions>

</mat-card>

**book-detail.component.ts:** It gets all book details and post them on the page further there is option of deleting the records. At end it routed to front page book.

import {Component, OnInit} from '@angular/core';

import {ActivatedRoute, Router} from '@angular/router';

import {ApiService} from '../api.service';

@Component({

  selector: 'app-book-detail',

  templateUrl: './book-detail.component.html',

  styleUrls: ['./book-detail.component.css']

})

export class BookDetailComponent implements OnInit {

  book = {};

  constructor(private route: ActivatedRoute, private api: ApiService, private router: Router) {

  }

  ngOnInit() {

    this.getBookDetails(this.route.snapshot.params['id']);

  }

  getBookDetails(id) {

    this.api.getBook(id)

      .subscribe(data => {

        console.log(data);

        this.book = data;

      });

  }

  deleteBook(id) {

    this.api.deleteBook(id)

      .subscribe(res => {

          this.router.navigate(['/books']);

        }, (err) => {

          console.log(err);

        }

      );}}

**book-edit.component.html:** For editing the fields the routes back to create page and save button disabled all fields needed gets filled.

<div class="button-row">

  <a mat-raised-button color="primary" [routerLink]="['/books']">

    <mat-icon>list</mat-icon>

  </a>

</div>

<form [formGroup]="bookForm" (ngSubmit)="onFormSubmit(bookForm.value)">

  <mat-form-field class="example-full-width">

    <input matInput placeholder="ISBN" formControlName="isbn"

           [errorStateMatcher]="matcher">

    <mat-error>

      <span \*ngIf="!bookForm.get('isbn').valid && bookForm.get('isbn').touched">Please enter ISBN</span>

    </mat-error>

  </mat-form-field>

  <mat-form-field class="example-full-width">

    <input matInput placeholder="Title" formControlName="title"

           [errorStateMatcher]="matcher">

    <mat-error>

      <span \*ngIf="!bookForm.get('title').valid && bookForm.get('title').touched">Please enter Book Title</span>

    </mat-error>

  </mat-form-field>

  <mat-form-field class="example-full-width">

    <input matInput placeholder="Author" formControlName="author"

           [errorStateMatcher]="matcher">

    <mat-error>

      <span \*ngIf="!bookForm.get('author').valid && bookForm.get('author').touched">Please enter Book Author</span>

    </mat-error>

  </mat-form-field>

  <mat-form-field class="example-full-width">

    <textarea matInput placeholder="Description" formControlName="description"

              [errorStateMatcher]="matcher"></textarea>

    <mat-error>

      <span \*ngIf="!bookForm.get('description').valid && bookForm.get('description').touched">Please enter Book Description</span>

    </mat-error>

  </mat-form-field>

  <mat-form-field class="example-full-width">

    <input matInput placeholder="Publisher" formControlName="publisher"

           [errorStateMatcher]="matcher">

    <mat-error>

      <span \*ngIf="!bookForm.get('publisher').valid && bookForm.get('publisher').touched">Please enter Publisher</span>

    </mat-error>

  </mat-form-field>

  <mat-form-field class="example-full-width">

    <input matInput placeholder="Published Year" formControlName="published\_year"

           [errorStateMatcher]="matcher">

    <mat-error>

      <span \*ngIf="!bookForm.get('published\_year').valid && bookForm.get('published\_year').touched">Please enter Published Year</span>

    </mat-error>

  </mat-form-field>

  <div class="button-row">

    <button type="submit" [disabled]="!bookForm.valid" mat-raised-button color="primary">

      <mat-icon>save</mat-icon>

    </button>

  </div>

</form>

**book-edit.component.ts:** It gets all edited data from user on form submission and updates them in book view page which is front page

import {Component, OnInit} from '@angular/core';

import {Router, ActivatedRoute} from '@angular/router';

import {ApiService} from '../api.service';

import {FormControl, FormGroupDirective, FormBuilder, FormGroup, NgForm, Validators} from '@angular/forms';

@Component({

  selector: 'app-book-edit',

  templateUrl: './book-edit.component.html',

  styleUrls: ['./book-edit.component.css']

})

export class BookEditComponent implements OnInit {

  book = {};

  bookForm: FormGroup;

  isbn: string = '';

  title: string = '';

  description: string = '';

  author: string = '';

  publisher: string = '';

  published\_year: string = '';

  constructor(private router: Router, private route: ActivatedRoute, private api: ApiService, private formBuilder: FormBuilder) {

  }

  ngOnInit() {

    this.bookForm = this.formBuilder.group({

      'isbn': [null, Validators.required],

      'title': [null, Validators.required],

      'description': [null, Validators.required],

      'author': [null, Validators.required],

      'publisher': [null, Validators.required],

      'published\_year': [null, Validators.required]

    });

    this.getBook(this.route.snapshot.params['id']);

  }

  getBookDetails(id) {

    this.api.getBook(id)

      .subscribe(data => {

        console.log(data);

        this.book = data;

      });

  }

  onFormSubmit(form: NgForm) {

    let id = this.route.snapshot.params['id'];

    console.log(form)

    this.api.updateBook(id, form)

      .subscribe(res => {

        this.router.navigate(['/book-details', id]);

      }, (err) => {

        console.log(err);

      });

  }

  getBook(id) {

    this.api.getBook(id).subscribe(data => {

      id = data.\_id;

      this.bookForm.setValue({

        isbn: data.isbn,

        title: data.title,

        description: data.description,

        author: data.author,

        publisher: data.publisher,

        published\_year: data.published\_year

      });

    });

  }

}