

E-LEARNING PLATFORM

A Project Progress Report

Submitted in fulfillment for the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

Submitted by

T.Bala Sai Teja – (R170054)

Under the Esteem Guidance of

Ms C.Suneetha



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Rajiv Gandhi University of Knowledge and Technologies - R.K.Valley

Kadapa, Andhra Pradesh-516330

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES



RGUKT

(A.P.Government Act 18 of 2008)

RGUKT, RK VALLEY

Department of Computer Science and Engineering

CERTIFICATE FOR PROJECT COMPLETION

This is certify that the project entitled “**E-LEARNING PLATFORM**” submitted by **T Bala Sai Teja (R170041)** under our supervision for the degree **Bachelor of Technology in Computer Science and Engineering** during the academic year September 2022 - April 2023 at RGUKT, RK VALLEY. To the best of my knowledge, the results embodied in this dissertation work have not been submitted to any University or Institute for the award of any degree or diploma.

Project Internal Guide

Ms C.Suneetha

Assistant Professor,

RGUKT, RK Valley.

Head of the Department

Mr.N.Satyanandaram

HOD Of CSE,

RGUKT, RK Valley.



Rajiv Gandhi University of Knowledge Technologies
RK Valley, Kadapa (Dist), Andhra Pradesh, 516330

DECLARATION

We hereby declare that the report of the B.Tech Major Project Work entitled “**E-LEARNING PLATFORM**” which is being submitted to Rajiv Gandhi University of Knowledge Technologies, RK Valley, for fulfillment of the requirements for the award of Degree of Bachelor of Technology in Computer Science and Engineering, is a bonafide report of the work carried out by me. The material contained in this report has not been submitted to any university or institution for award of any degree.

T Bala Sai Teja – R170054

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts success. I am extremely grateful to our respected Director, Prof. K. SANDHYA RANI for fostering an excellent academic climate in our institution. I also express my sincere gratitude to our respected Head of the Department Mr. N. SATYANANDARAM for their encouragement, overall guidance in viewing this project a good asset and effort in bringing out this project. I would like to convey thanks to our guide at college **Ms. C. SUNEETHA** for their guidance, encouragement, co-operation and kindness during the entire duration of the course and academics. My sincere thanks to all the members who helped me directly and indirectly in the completion of project work. I express my profound gratitude to all our friends and family members for their encouragement.

INDEX

1. Abstract	6
2. Introduction	7
3. Technologies Used	7
4. Requirements Specification	10
5. Proposed Model and Flow of the project	11
6. Source code and outputs	13
7. Conclusion	29
8. Future Enhancement	29
9. References	29

ABSTRACT

Turito is an E-learning platform that provides online live learning sessions to young students preparing for various National competitive exams such as IIT JEE/ NEET etc. They also offer foundation courses for students from Grade 1 to Grade 12. Our vision is to make students across the globe directly interact with the teachers for guidance.

There are two different web projects like Student management system for the students to access learning platform and a Teacher management system for the teachers to access, analysis and maintain the students data. We are using the technologies like Scala, slack and Postgre SQL for the backend and for the database purpose, also we are using Angular, Bootstrap for the frontend of the project.

INTRODUCTION

Turito is a revolutionary E-learning platform that wants to bring a balance in the realm of Live Online Learning as today, only a few students or coaching centers have access to the best teachers. We at Turito want to ensure, all aspiring students across various countries get access to the best faculty at an affordable price. Now we are going to develop the website furtherly so that it would be more interactive for the students and teachers. Those features are bookmarking and rescheduling the classes, connecting with tutors, searching for a doubt, quiz zone, and showing the student's progress. Also a separate management portal for the teachers to access features like, 24*7 doubts clarification, video lectures, study materials, etc., and conducts mock tests and performs post-exam Analysis. After adding all these features, it moves to the testing phase. I mainly worked on the front-end part of the project.

OVERVIEW

We need to collect the data from the database. After collecting the data, we perform preprocessing using various technologies. We need to build APIs so that we can fetch the data from the database and display it in the front end by making API calls. Our team lead will assign some small tasks mostly based on Angular, we are going to explain the different technologies we have used in this project.

TECHNOLOGIES USED

- FRONTEND : Angular, Bootstrap, MathJax, SendBirdUIKit .
- BACKEND : Scala, Postgre SQL.
- ENVIRONMENT : Visual Studio Code.

Angular

Angular is a TypeScript-based, free and open-source web application framework led by the Angular Team at Google and by a community of individuals and corporations. Angular is a complete rewrite from the same team that built AngularJS. Angular is a Single Page

Application (SPA) Framework which is used for creating Fast Web Applications. It uses concepts of SPA in which UI is delivered in the beginning of application request and later only data is requested which makes SPA applications fast.

The architecture of an Angular application relies on certain fundamental concepts. The basic building blocks of the Angular framework are Angular components that are organized into NgModules. NgModules collect related code into functional sets; an Angular application is defined by a set of NgModules. An application always has at least a root module that enables bootstrapping, and typically has many more feature modules.

- Components define views, which are sets of screen elements that Angular can choose among and modify according to your program logic and data
- Components use services, which provide specific functionality not directly related to views. Service providers can be injected into components as dependencies, making your code modular, reusable, and efficient.

Modules, components and services are classes that use decorators. These decorators mark their type and provide metadata that tells Angular how to use them.

Bootstrap

Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains HTML, CSS and (optionally) JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components. The primary purpose of adding it to a web project is to apply Bootstrap's choices of color, size, font and layout to that project. As such, the primary factor is whether the developers in charge find those choices to their liking. Once added to a project, Bootstrap provides basic style definitions for all HTML elements. The result is a uniform appearance for prose, tables and form elements across web browsers. In addition, developers can take advantage of CSS classes defined in Bootstrap to further customize the appearance of their contents. For example, Bootstrap has provisioned for light- and dark-colored tables, page headings, more prominent pull quotes, and text with a highlight.

Angular Material

Angular Material is a User Interface (UI) component library that developers can use in their Angular projects to speed up the development of elegant and consistent user interfaces. Angular Material offers you reusable and beautiful UI components like Cards, Inputs, Data Tables, Datepickers, and much more.

Each component is ready to go with default styling that follows the Material Design Specification. Nonetheless, you can easily customize the look and feel of Angular Material components. The list of available Angular Material components continues to grow with each iteration of the library.

Sendbird UIKit

UIKit is a Sendbird Chat SDK add-on with user interfaces that enables easy and fast integration of standard chat features into new or existing client apps.

If you would like a sample app with embedded UI, see UIKit Quickstart for Android.

Access control list

Sendbird provides various access control options when using the Chat SDK. By default, the following attributes are turned on to avoid unexpected errors when creating sample apps and sending your first message:

- Allow retrieving user list
- Allow updating user metadata
- Allow creating open channels
- Allow creating group channels

MathJax

MathJax is a cross-browser JavaScript library that displays mathematical notation in web browsers, using MathML, LaTeX and ASCII MathML markup. MathJax is released as open-source software under the Apache License.

MathJax is downloaded as part of a web page, scans the page for mathematical markup, and typesets the mathematical information accordingly. Thus, MathJax requires no installation of software or extra fonts on the reader's system. This allows MathJax to run in any browser with JavaScript support, including mobile devices.^[16]

MathJax can display math by using a combination of HTML and CSS or by using the browser's native MathML support, when available. The exact method MathJax uses to typeset math is determined by the capabilities of the user's browser, fonts available on the user's system, and configuration settings. MathJax v2.0-beta introduced SVG rendering.

REQUIREMENT SPECIFICATION

Hardware Configuration:

Client side:

Ram	512 MB
Hard disk	10GB
Processor	1.0 GHz

Server side:

Ram	4GB
Hard disk	200GB
Processor	2.0GHz

Software Requirements:

Front end	Angular,Html,Css
Server side Language	Scala
Database Server	Postgre SQL
Web Browser	Firefox,Chrome or any compatible Browser
Operating System	Ubuntu,Windows or any Compatible Browser
Software	Turito

PROPOSED MODEL AND FLOW OF THE PROJECT

Proposed model

Our application will call APIs and fetch data from the backend. We will test API using Postman. If we want to get the user data, we need to use an API call with params [GET method]. If we want to post or update some data in the database, we need to send the body with an API call [POST, UPDATE method].

We write the API handlers in DBweaver using Scala programming language. In the initial loading of the page, an API call will be called using `getServerSideProps` which assures prerendering. After that for every API call, Axios is used. The fetched data will be formatted and will be displayed to the user in the front end. When the user hits an API call the server receives the API call and requests data in the elastic search database. The elastic search database will send the requested data to the server. The server will then receive the data and format it and send it to the client. In this chapter, we included the proposed model and what are the steps we have taken to complete this project.

Flow of the project

Collecting Data: We build APIs and give them to our clients. After that our clients will push the data into the collector service using APIs provided by us. That data includes everything from the clients. That is raw data. It will have a lot of redundant and unnecessary information. So, we need to remove those unwanted data and structure the data.

Refining the data: As we see in the first step, we will get data from the collector service which is raw data. So, in this step, we will refine the data to make it more structured. After the collector service completed collecting the data, it will then push the data to Kafka which will refine this data and make some useful aggregations. After refining the data, A consumer service will take that data and push it into Elastic Search DB which is a NoSQL database mostly used for data that need calculations, searching, and aggregations. After this, Data will be available to the server.

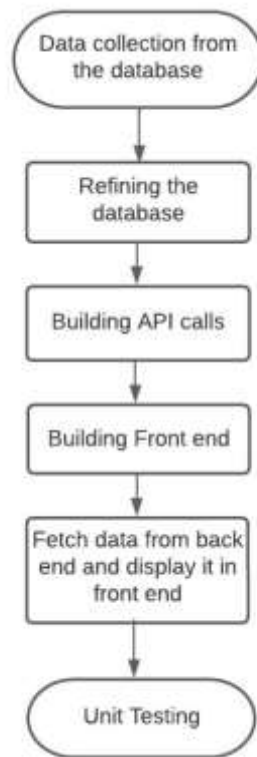


Fig:1 Flow of the project

Building API Calls: After completion of storing the data in the elastic db. We will establish a connection between the server and the database. The server will access the data from the database by using queries that are written in elastic Search db language. We build API calls so that front-end can access the data. When the front-end calls an API, the server will get the request then the server will process the request and the server will access the data from the elastic search database. And send a response to the front end with that data.

GET: It will retrieve data from the API.

POST: It sends new data into an API.

Building Frontend and Fetch Data: After building the API calls. The front end needs to be developed so that it can call APIs and show data to the user in a good manner. In Frontend we use Angular

Initially when the site is loading API calls will be done and data will be fetched and we will get that data and display that data. Whenever the client wants to see different data, they will click on the buttons which will call APIs and the corresponding data will be displayed. We didn't use any kind of libraries and packages for styling like Bootstrap and material UI. We just used plain CSS which is provided by the UX designers. We implemented the front end so that it exactly looks like as design given by the UX designers.

SOURCE CODE AND OUTPUTS

appcomponent.html

```
<!DOCTYPE html>
<html lang="en" *ngIf="loadingcnt == 0" style="background: #f7f9fc !important;">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/css/bootstrap.min.css" rel="stylesheet"
    integrity="sha384-
eOJMYsd53ii+sc0/bJGFsiCZc+5NDVN2yr8+0RDqr0Ql0h+rP48ckxlpbzkGwra6"
crossorigin="anonymous">
  <link href="https://maxcdn.bootstrapcdn.com/font-awesome/4.4.0/css/font-
awesome.min.css" rel="stylesheet">
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
  <link rel="stylesheet"
href="https://use.fontawesome.com/releases/v5.7.0/css/all.css"
    integrity="sha384-
1ZN37f5QgtY3VHgisS14W3ExzMWZxybE1SJSEsQp9S+oqd12jhcu+A56Ebc1zFSJ"
crossorigin="anonymous">
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/js/bootstrap.bundle.min.js"
    integrity="sha384-
JEW9xMcG8R+ph31jmWH6WWP0WintQrMb4s7Z0dauHnUtxwoG2vI5DkLtS3qm9Ekf"
    crossorigin="anonymous"></script>
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"><
/script>
```

```

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.15.1/css/all.min.css">

    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></scrip
t>

</head>

<body>
    <top-bar *ngIf="this.localStorageService.getItem('isLogin') == 'true'"></top-
bar>
    <!-- Notify the end user about current deployment -->
    <div class="d-flex">
        <div *ngIf="!isWatchcompActive">
            <sidebar *ngIf="this.localStorageService.getItem('isLogin') ==
'true'" #sidebar class="side_navbar"
                (click)="opensidebar()"></sidebar>
        </div>
        <div class="main_content" (click)="closeSidebar()">
            <!-- #mypagecontent -->
            <app-custom-notifier></app-custom-notifier>
            <router-outlet (activate)="changeOfRoutes()"
(click)="closeSidebar()"> </router-outlet>
        </div>
    </div>
    <!-- <div class="row">
        <footer class="col-12 bottom-footer" style="background-
color:lavender;"></footer>
    </div> -->
</body>

</html>
<div class="preload_spinner mt-4" *ngIf="loadingcnt > 0">
    <div></div>
    <div></div>
</div>

```

appcomponent.ts

```

import { SidebarComponent } from './sidebar/sidebar.component';
@Component({
    selector: 'app-root',
    templateUrl: './app.component.html',
    styleUrls: ['./app.component.css']

```

```

}))
export class AppComponent {
  title = 'Turito-Teacher-Management';
  loggedIn: boolean = false;
  loadingcnt: number = 0;
  public loader = {};
  private loggedInSubscription = {};
  public moduleLoader: number = 0;
  fromautologin: boolean = false;
  autologinurl: string = '';
  autologinurl_ar: string[] = [];
  @ViewChild('sidebar') sidebar !: ElementRef
  @ViewChild(SidebarComponent) private sidebarwidth!: SidebarComponent;
  @ViewChild('mypagecontent') mypagecontent !: ElementRef
  constructor(private router: Router, public dataManager: DataManager, public
userService: UserService,
    public localStorageService: LocalStorageService, private loaderService:
LoaderService,
    private activatedRoute: ActivatedRoute, private location: Location) {

    let userSession = JSON.parse(this.localStorageService.getItem("user-Session-
Details")) ? JSON.parse(localStorageService.getItem("user-Session-Details")) : {}

    //---- to get web view for mobile device, autologin functionality.
    //--- checking url includes autologin , if the flag is set routing to
autologin component with boxid and sessionid values

    let autologin_url = window.location.href
    if (autologin_url.includes('autologin')) {
      this.router.navigate([window.location.pathname]);
    }
    else if (userSession.sessionId == undefined) {
      this.dataManager.getTokenApi().subscribe(data => {
        if (data['status'] == 1) {
          AppConfig.TENANT_NAME = data['response'].tenantCode
          let info = {
            sessionId: data['response'].sessionId,
          }
          this.userService.setUserSessionDetails(info);
          this.dataManager.setSessionId(data['response'].sessionId);
          dataManager.sessionId = data['response'].sessionId
          this.loggedIn = this.localStorageService.getItem("isLogin") == "true" ?
true : false

```

```

        let isUserTeacher = this.localStorageService.getItem("isUserTeacher")
        == "true" ? true : false
        // if (this.loggedIn && isUserTeacher) {
        //   this.router.navigate(['/teacher/calender']);
        // }
        // else if (this.loggedIn) {
        //   this.router.navigate(['/courses']);
        // }
        if (!this.loggedIn) {
            this.router.navigate(['/login']);
        }
    }
    else {
        console.log("No Response")
    }

    });
}
this.loggedIn = this.localStorageService.getItem("isLogin") == "true" ? true
: false
let isUserTeacher = this.localStorageService.getItem("isUserTeacher") ==
"true" ? true : false
if (this.loggedIn && isUserTeacher) {
    this.loaderService.loaderSource.subscribe(d => {
        this.loader = d;
    })
    let totalurl = window.location.href
    router.events.subscribe((event) => {
        if (event instanceof RouteConfigLoadStart) {
            this.moduleLoader++;

        } else if (event instanceof RouteConfigLoadEnd) {

            this.moduleLoader--;

        }
    });
} else if (this.loggedIn) {
    this.loaderService.loaderSource.subscribe(d => {
        this.loader = d;
    })
    let totalurl = window.location.href
    router.events.subscribe((event) => {
        if (event instanceof RouteConfigLoadStart) {
            this.moduleLoader++;

```



```

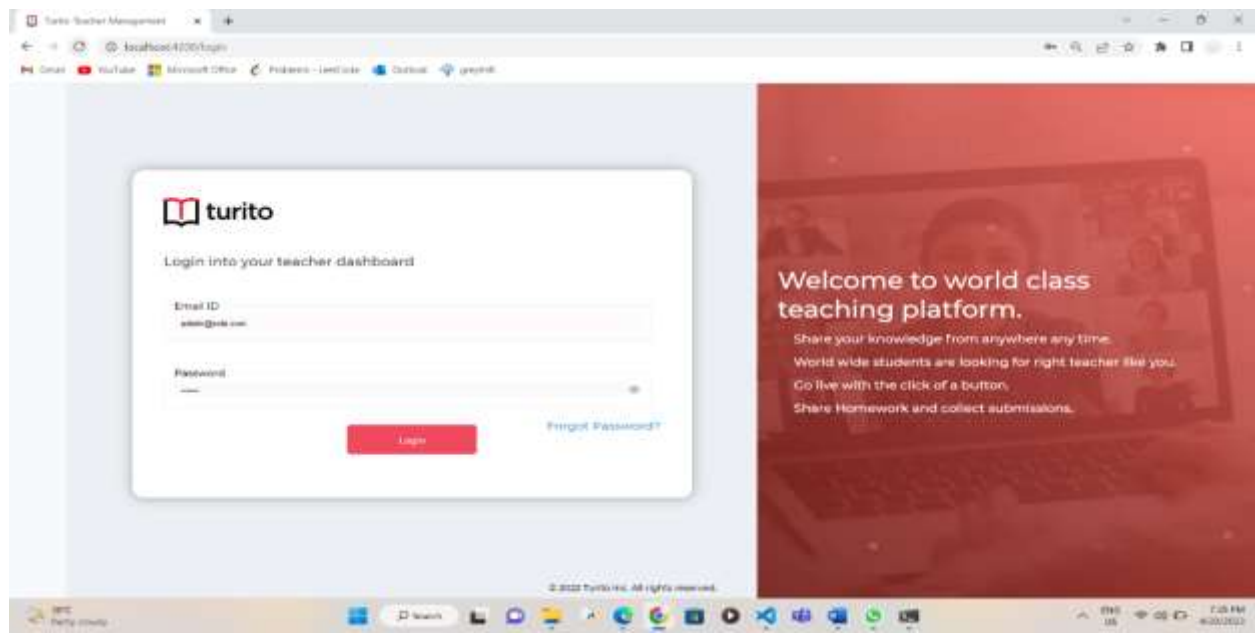
    } else if (event instanceof RouteConfigLoadEnd) {

        this.moduleLoader--;

    }
    });
}
else {
    if (!autologin_url.includes('autologin'))
        this.router.navigate(['/login']);
}
// this.mypagecontent.nativeElement.classList.add('main_content')
} //end of constructor
ngOnInit() {
    this.userService.onAppInIt();
    this.loaderService.loaderSource.subscribe(d => {
        this.loadingcnt++;
        this.loader = d;
        setTimeout(() => {
            this.loadingcnt--;
        }, 150);
    });
}
}

```

Output



Sidebar.html

```
<div class="side_navbar" [ngClass]="sidebarActivated ? 'sidebar-width' : ''
*ngIf="!roleService.isSchoolUser()">
  <ul class="side_menubar">
    <ng-container *ngFor="let menu of sidebarData; let i = index">
      <!-- single tab -->
      <li *ngIf="menu.type == 'single' && menu.access" [id]='menuItem' + i">
        <details [open]="menu.isOpen">
          <summary (click)="$event.preventDefault(); activateMenu(i)" class="d-
flex align-items-center"
            [routerLink]="menu.route" (click)="highlightMenu(i)">
            <div class="d-flex w-100 list_items_container">
              <span class="svg-container" [innerHTML]="menu.svg | safeResource"
                [routerLinkActive]='["active_tab"]"></span>
              <span class="d-flex" [routerLinkActive]='["active_tab"]">{{
menu.name }}
                </span>
              <span class="svg-container-arrow ml-auto"
[routerLinkActive]='["active_tab"]"></span>
            </div>
          </summary>
        </details>
      </li>
      <!-- multi tab -->
      <li #parent *ngIf="
        (menu.type == 'parent' || menu.type == 'external') && menu.access
        " [id]='menuItem' + i">
        <details [open]="menu.isOpen">
          <summary (click)="$event.preventDefault(); activateMenu(i)">
            <div class="d-flex w-100 list_items_container">
              <span class="svg-container" [innerHTML]="menu.svg | safeResource"
                [ngClass]="{ activTabColor: menu.isActive }"></span>
              <span [ngClass]="{ activTabColor: menu.isActive }">{{
                menu.name
              }}</span>
              <span class="svg-container-arrow ml-auto"
[innerHTML]="sidebarService.arrowIcon | safeResource"
                [ngClass]="{ activTabColor: menu.isActive }"></span>
            </div>
          </summary>
        </li>
        <ng-container *ngFor="let child of menu.children; let ci = index">
          <li [routerLink]="child.route" *ngIf="child.access">
```

```

        <a (click)="highlightMenu(i)" *ngIf="menu.type == 'parent'"
[routerLinkActive]="['active_tab']"
        [ngClass]="
            child?.backLinksActivated || backTrack.isActive
            ? 'active_tab'
            : 'non_active_tab'
        " [routerLinkActiveOptions]="{ exact: true }">
        {{ child.name }}
    </a>
    <!-- [ngClass]="{'active_tab': child?.backLinksActivated}" -->
    <!-- -->
    <a *ngIf="menu.type == 'external'" [href]="child.route"
target="_blank">{{ child.name }}</a>
    </li>
    <div *ngIf="child.backLinks">
        <a routerLink="{{ innerChild.route }}" *ngFor="let innerChild of
child.backLinks"
            [routerLinkActiveOptions]="{ exact: false }" routerLinkActive
#rla="routerLinkActive"
            style="display: none">
            {{ getValue(rla.isActive, ci, i) }}
        </a>
    </div>
    <a routerLink="{{ child.route }}" [routerLinkActiveOptions]="{
exact: true }" routerLinkActive
        #backTrack="routerLinkActive" style="display: none">
    </a>
    </ng-container>
</ol>
</details>
</li>
</ng-container>
</ul>
</div>

```

Sidebarcomponent.ts

```

import { SidebarService } from './sidebar.service';

@Component({
    selector: 'sidebar',
    templateUrl: './sidebar.component.html',
    styleUrls: ['./sidebar.component.css'],
    providers:[SidebarService]

```

```

}))
export class SidebarComponent implements OnInit {
  isUserTeacher: boolean = false;
  isUserCounsellor: boolean = false;
  pollApiId: any = {};
  sidebarActivated: boolean = false;

  Roles = Roles;
  Modules = Modules;
  Features = Features;

  sidebarData: any = []

  constructor(
    public localStorageService: LocalStorageService,
    public roleService: RoleService,
    private userService: UserService,
    private dataManager: DataManager,
    private router: Router,
    private location: Location,
    public sidebarService: SidebarService
    // @Inject(DOCUMENT) document: Document
  ) {
    //route instance
    this.sidebarData = this.sidebarService.sidebarData;
    this.router.events.subscribe((event: Event) => {
      if (event instanceof NavigationEnd) {
        // Show progress spinner or progress bar
        //reverting backLinks to false
        if(this.sidebarData[this.parentId]){
          this.sidebarData[this.parentId].isOpen = false
          this.sidebarData[this.parentId].isActive = false
        }
        if(this.sidebarData[this.parentId]?.children != undefined){
          this.sidebarData[this.parentId].children[this.childId].backLinksActivated = false
        }
        this.activateDefault();
        // console.log(this.sidebarData[this.parentId].children[this.childId])
      }
    })
    this.activateDefault();

    this.isUserTeacher =

```

```

        this.localStorageService.getItem('isUserTeacher') == 'true'
            ? true
            : false;
let userId = this.userService.getUserData()?.userId;
if (this.isUserTeacher) {
    this.teacherPollApis(userId);
    this.pollApiId = setInterval(() => {
        this.teacherPollApis(userId);
    }, 60000);
}
this.isUserCounsellor =
    this.localStorageService.getItem('isUserCounsellor') == 'true'
        ? true
        : false;
}

ngOnInit(): void {
}
ngAfterViewInit() {
    // this.checkActiveParent();
}

ngAfterContentChecked(){

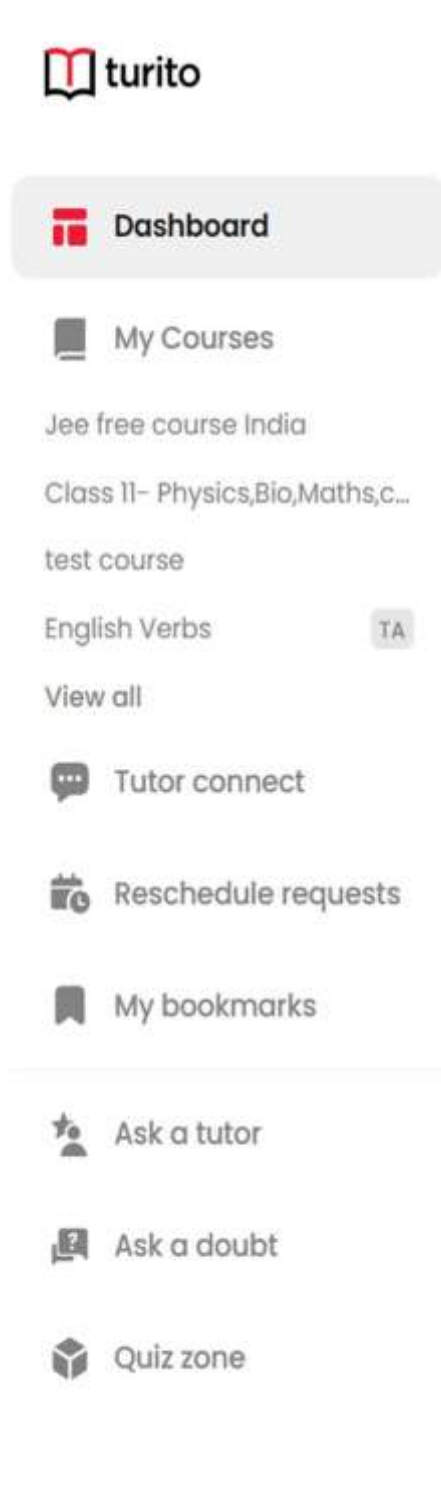
}

ngOnDestroy() {
    if (this.pollApiId) {
        clearInterval(this.pollApiId);
    }
}

teacherPollApis(userId: number) {
    let body: any = {};
    body.userId = userId;
    body.role = 'teacher';
    body.source = 'tms';
    body.deviceId = 5;
    let entity = 'service/api/education/v1/poll/user';
    this.dataManager.postApi(entity, body).subscribe((data) => {
        if (data['status'] == 1) {
        } else {
        }
    });
}

```

Side Bar



- **Dashboard:** In this, we will display greetings to the user and no. of classes or tests that are scheduled today. And it shows the weekly and monthly calendar so that we can see the classes that are scheduled for that week/month.
- **My Courses:** The courses that are enrolled by the user will be displayed in this section.
- **My Bookmarks:** If the user wants to revisit the study material or video lecture, he can bookmark that one. Whenever he wants to read/watch it, he can open these.
- **Reschedule requests:** If the student/ teacher wants to reschedule the class due to an unavoidable situation, they can raise a request. ☐
- **Ask a tutor:** By this, we can connect to a tutor if we had any doubts or want to interact with them a little. They can clarify them.
- **Ask a Doubt:** Here we can ask the doubt either by typing it or using an image.
- **Discover Courses:** It lets you go through all the courses provided by Turito.

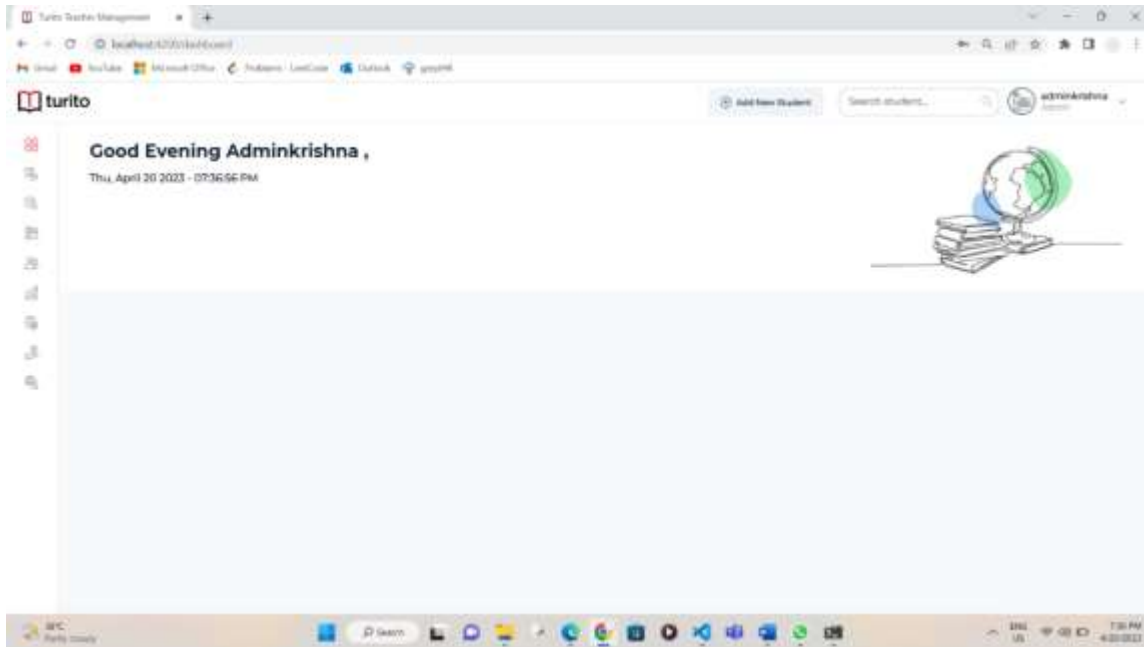
Dashboard.html

```
<div class="content-area" style="cursor: default; background-color: #fff;">
  <div class="container-fluid">
    <div class="Main_box">
      <div class="main_txt">
        <h4>{{greet}} {{userName}} , </h4>
        <div class="time_box">
          <clock style=" color: #06152b;
                    font-size: 18px;
                    font-family: 'Montserrat-semibold'; font-weight: 600;
"></clock>
          <span>
            <dashboard-quote *ngIf="widgetDisplay.quotes"></dashboard-quote>

          </span>
        </div>
      </div>
      <div>
        
        </div>
        <dashboard-broadcast-message
*ngIf="widgetDisplay.broadcastMessages"></dashboard-broadcast-message>
      </div>
    </div>
  </div>
</div>
```

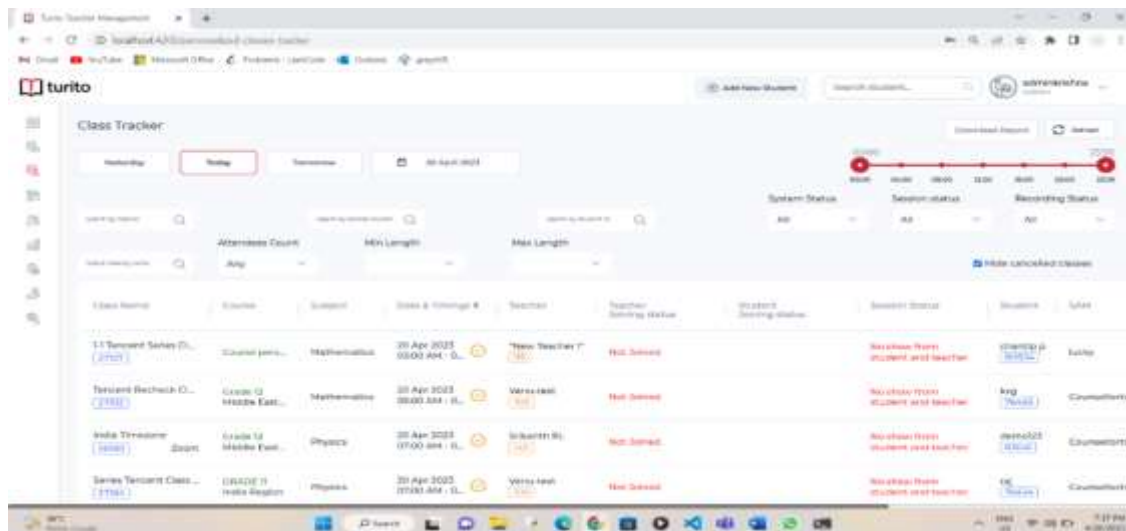
Dashboard

In the dashboard, I need to update the greeting as well as the background image based on timing. For this `todaydate()` function is used. If the user was already signed in, we need to show the SEO of the dashboard as 'username's dashboard | Turito' based on the login information of the user. Otherwise, we just need to show the 'Dashboard' only.



Displaying classes information

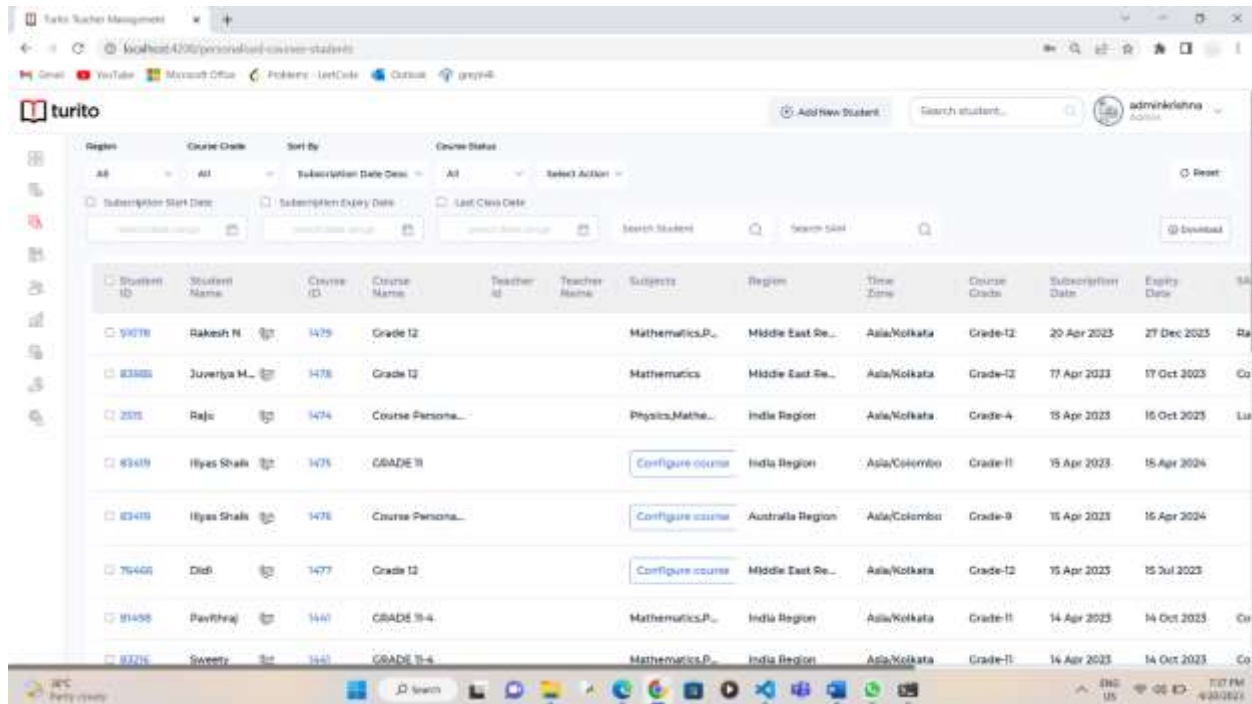
We need to show the classes that are scheduled by the user in the dashboard based on the date as shown in the below figure. By default, it will show today's class information. For this, we need to make an API call to get class information from the backend. It will return some details like the teacher's name, timings of the class, subject and chapter names, and whether the class has been started or not based on class timings. It will show various buttons like Replay if the class was already completed, join, etc.



Classes that are scheduled

Redirecting to course page

In my courses page, a list of courses that are enrolled by the user will be displayed as shown in the below figure. Every course has a unique id. After the user clicks on a particular course, we need to redirect the user to that course page based on the id. To do this, we need to make an API call by passing the id as a parameter.



The screenshot shows the 'Turito Teacher Management' interface. At the top, there's a navigation bar with the Turito logo and a user profile 'admin@turito'. Below this is a filter section with tabs for 'Region', 'Course Grade', 'Sort By', and 'Course Status'. The 'Region' tab is active, showing a list of courses. The table has columns for Student ID, Student Name, Course ID, Course Name, Teacher ID, Teacher Name, Subjects, Region, Time Zone, Course Grade, Subscription Date, Expiry Date, and an action column. The data rows show various courses with details like 'Rakesh N.', 'Juveriya M.', 'Raju', 'Ilyas Shaik', 'Dilip', 'Payalraj', and 'Sweety'.

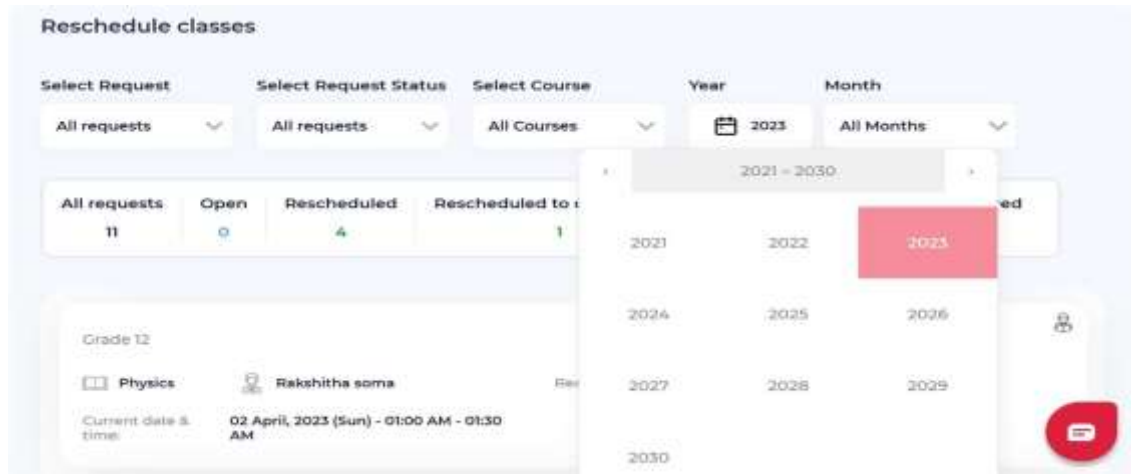
Student ID	Student Name	Course ID	Course Name	Teacher ID	Teacher Name	Subjects	Region	Time Zone	Course Grade	Subscription Date	Expiry Date	Action
9078	Rakesh N.	1479	Grade 12			Mathematics, P...	Middle East Re...	Asia/Kolkata	Grade-12	20 Apr 2023	27 Dec 2023	Ra
8388	Juveriya M.	1478	Grade 12			Mathematics	Middle East Re...	Asia/Kolkata	Grade-12	17 Apr 2023	17 Oct 2023	Co
255	Raju	1474	Course Persona...			Physics, Mathe...	India Region	Asia/Kolkata	Grade-4	18 Apr 2023	18 Oct 2023	Lu
8348	Ilyas Shaik	1475	GRADE 11			Configure course	India Region	Asia/Colombo	Grade-11	18 Apr 2023	18 Apr 2024	
8348	Ilyas Shaik	1476	Course Persona...			Configure course	Australia Region	Asia/Colombo	Grade-9	18 Apr 2023	18 Apr 2024	
7640	Dilip	1477	Grade 12			Configure course	Middle East Re...	Asia/Kolkata	Grade-12	18 Apr 2023	18 Jul 2023	
81458	Payalraj	1440	GRADE 11-4			Mathematics, P...	India Region	Asia/Kolkata	Grade-11	14 Apr 2023	14 Oct 2023	Co
8326	Sweety	1440	GRADE 11-4			Mathematics, P...	India Region	Asia/Kolkata	Grade-11	14 Apr 2023	14 Oct 2023	Co

List of courses

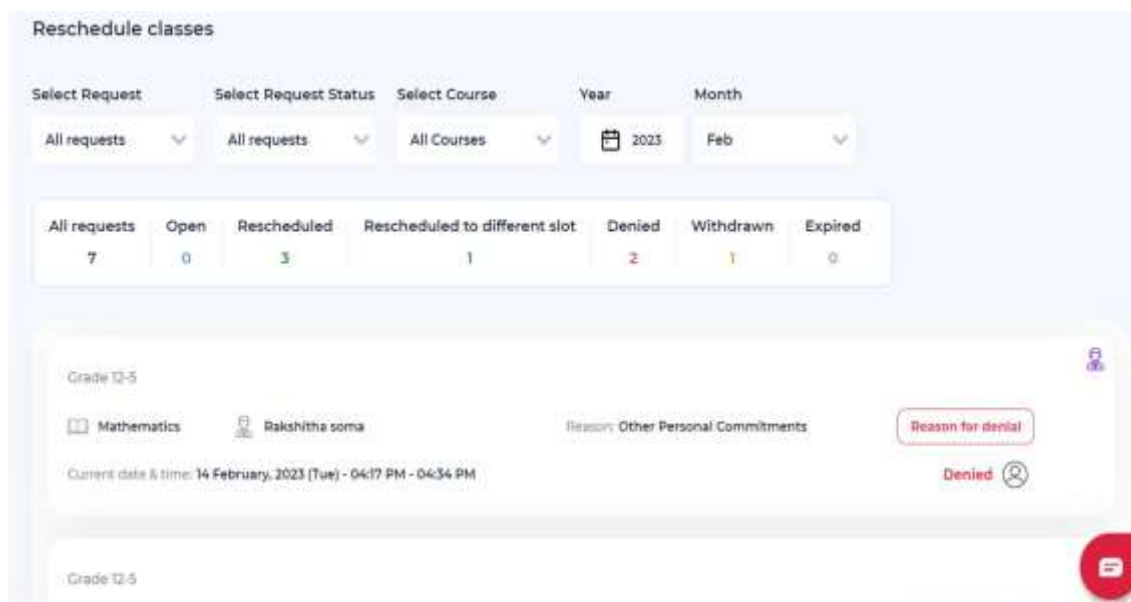
Rescheduling classes

Students or Teachers reschedule some classes due to some unavoidable situation as shown in the below figure. There are some filters like select request, select request status, select course, select year, and select month. Based on the selected values of the filter, we need to make an API call and display the classes that are rescheduled. If we don't select any values, it will show the classes that are rescheduled in that particular year as shown in Figures.

I worked on two filters that are select year and select month. For the Select Year Filter, I used a react calendar component to select the year from the calendar. The default value for this filter is the current year. And for the Select Month filter, the default value is All Months. I used a react dropdown component to show all the months.



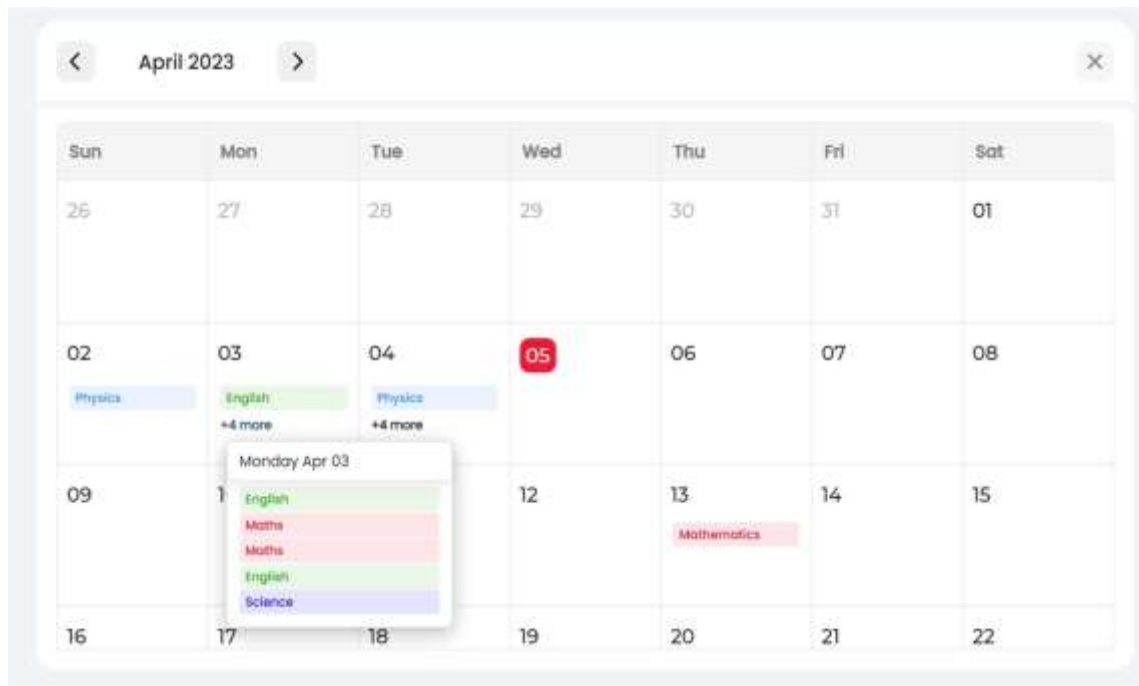
Selecting the year



Classes that are rescheduled in a particular month and year

Displaying Classes in a monthly calendar

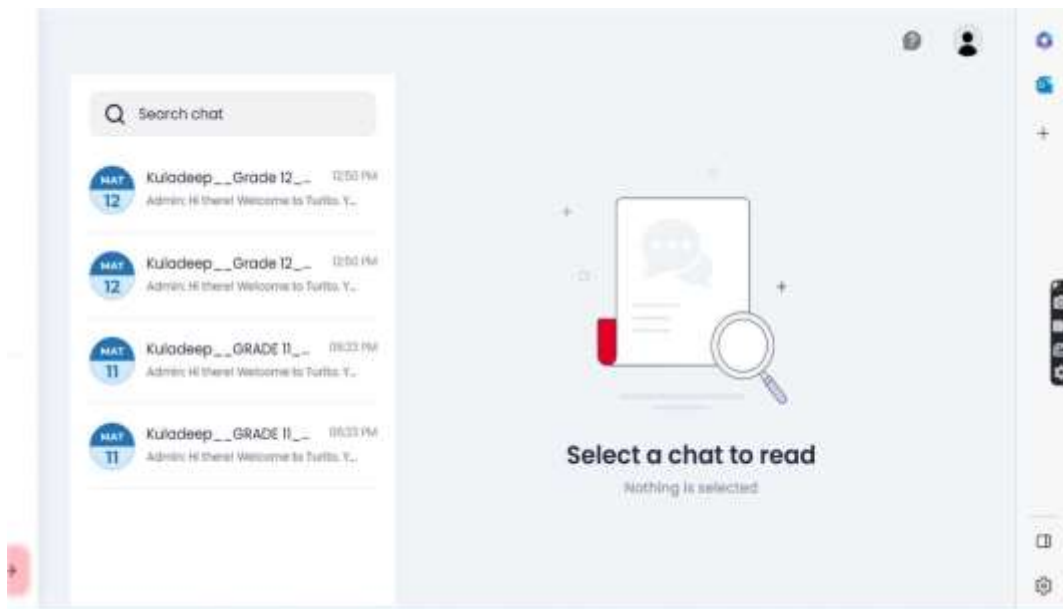
Based on courses that are enrolled by students, we need to show the classes in a monthly calendar as shown in the figure below. I need to add different colors and background colors to different subjects. To implement this, we need to use react-big-calendar component that provides a lot of features to customize the calendar however we want.



Classes info in monthly calendar

Displaying chats

A student can connect with teachers to clarify their doubts. At first, it will load information of only 10 chats. If we scroll the scroll bar to the end, it will load information about the next 10 chats, and it repeats as shown in the below figure. To implement this, we need to use the infinite scroll component. Params of API will be updated based on the scrolling. But the current user doesn't have that many chats so we are unable to see the implementation right now. And in the search bar, we can search for students/ teachers.



Chat information

CONCLUSION

Turito as a online education platform to be providing the best user interface for the students to learn and also make the better teacher experience for educating. Implemented the front end as per the design given by the UX designers.

FUTURE WORKS

In Turito, we want to add some more features like Ask a tutor that is a chat screen for the clarification of doubts, a new Enrollment center module for parents and student so that it would be easy for the students to enroll and also to develop the teachers management portal to more advance level, like by making class tracker on hourly basis to arrange the classes for students without any conflicts, a simple chat UI for the tutors to connect with students so that the communication between the student and teacher will be in sync.

REFERENCES

1. **<https://material.angular.io/>**
2. **<https://angular.io/docs>**
3. **<https://ng-bootstrap.github.io/#/home>**