Online Lunch Registration

CENTER FOR DEVELOPMENT OF ADVANCED COMPUTING

PG-DAC

PRN	Name
200250120015	Bhadri Uma Satya Lakshmi Annapurna
200250120018	Zaki Nisar Ahmed Bhatkar
200250120040	Kanike Raghava
200250120063	Pilligundla Keerthan
200250120065	Ponna Hemantha Veera Sanjay

About Project:

1. The main objective of the project is to convert the traditional lunch registration of **CDAC KP** to online.
2. There are three modules in this project.
    a. Admin
    b. Staff
    c. Student
3. Admin has privileges to
    a. Add Students and Staff members to database from UI.
        i. Members can be added individually.
        ii. Members can be added as batch (Uploaded as a **CSV file**).
    b. Reset student/staff password in case they forget there login credentials.
    c. Download students list who have registered for lunch as a file for a particular day.
    d. Delete student/staff details individually.
        i. Students can be deleted as a batch.
        ii. Delete the registered students list.
    e. View students list and payment list.
4. Staff has privileges to
    a. Update his/her profile and password.
    b. View the students list who have registered for lunch and he can approve for lunch.
    c. Can issue tea/coffee based on students coupons count.
5. Student has privileges to
    a. Update his/her profile and password.
    b. Buy coupons for lunch and tea.
    c. Register for lunch himself and his friends.

Student Lunch Registration Scenario:

1. Presentation Layer:
    a. Student can register for lunch by entering his/her PRN and can also add up to 5 valid PRNs by clicking on add button.
    b. PRN number is validated on pointer out event.
    c. Until Student enter the valid PRN, Registration button won't be enabled.
    d. Lunch Registration is accepted only before 10.A.M. and after 4.P.M.
    e. Entered student PRNs are added to list and sent to service layer via http method post along with JWT token in header.
    f. Response from service layer
        i. If student doesn't have enough coupons to register will get response from service layer as no enough coupons.
        ii. If student is already registered for that day will get response as student has already registered, if not he will be registered for lunch.

2. Service and DAO Layer:
    a. Via http post method, accepting list of students in body and JWT in header.
    b. Student is authenticated via JWT, if JWT is invalid it sets HTTP response as firebase exception with status 502.
    c. If JWT is valid then it checks timestamp and give response appropriately.
    d. If time is valid, it calls lunch registration method, here it gets student PRN by decrypting JWT token from firebase.
    e. Now getting details of that Particular PRN for checking coupons count from StudentList available in firebase.
    f. If student have enough lunch registration coupons then it checks the PRN in lunchRegisteredList collection, if PRN is present then it is added to registered student list or else added to unregistered list.
    g. Now unregistered list is iterated and add PRNs to lunchRegisteredList collection appropriately.
    h. Now coupons are debited from studentList collection for that particular PRN.
    i. Now the acknowledgement will be sent to student via UI.

3. Buying Coupons: (React Native Application)
    a. Student selects the coupons he/she wishes to buy (Tea / Lunch) and clicks on the pay button.
    b. On clicking the pay button , the call is made to the NodeJs server which creates a razorpay order Id and returns it as json data.
    c. The FrontEnd uses that orderId in razorpay options and opens the razorpay window.
    d. The student enters the required details for payment in the razorpay window and clicks on pay now button.
    e. The order if it's a success , executes the success handler which makes the call to the NodeJs Server to check if the order is valid and capture the payment.

f. If the order is a failure , the user is redirected to the razorpay window to retry the payment.
g. On successful payment , depending on the purchase of the student (Lunch/Tea), the specified 30 coupons are added to the coupons wallet of the student.

Scenarios where we got stuck:

1. In Angular, for check all scenario in delete registration list got stuck for changing checked state to unchecked state, So that taken two events one for select all and other for individual checkbox and added isSelected property to every checkbox value.
    a. In individual checkbox event, checking whether the check box is selected or not based on isSelected property, if isSelected is set to true then that value will be pushed in to checked list array.
    b. In Select All checkbox event, Iterating the loop and assigning the value returned by the above event to isSelected and again pushing values to checked list array.
    c. Then sending these list of values to service layer using append functionality in httpparams.

2. In Service Layer,
    a. The initial requirement for us is to generate a collection every day for lunchRegisteredList.
    b. So to achieve these requirement we have searched for good practises, Where we found about Scheduler.
    c. So we used @Scheduler annotation with corn expression which was successfully meeting our requirement.
    d. After deploying the spring boot application in Heroku, the required collection is not generated because if no request hit the server for 30mins the Heroku server goes into sleep state to save resources.
    e. So it was unable to trigger the scheduler at the given time. To tackle this scenario we came with an alternative solution.
    f. When student hits the server for lunch registration after 4pm,we check the collection exists or not, If exists we add the student to that collection list else we create a collection and add the student.

3. React Native Application
    a. Ejecting the React Native from Managed Workflow to Bare WorkFlow and making it work under react native CLI.
    b. We were able to solve the above problem by studying the various errors during compilation through forums like stackoverflow and tutorials on YouTube.
    c. Also we faced issue in integrating Razorpay in bare Workflow , but the documentation of razor pay was to the point which helped us to resolve the issue.
    d. Also we faced issue in integrating Razorpay in bare Workflow , but the documentation of razorpay was to the point which helped us to resolve the issue.

Learnings during the project

1. In Angular, learnt how to send data using httpparams, httpHeader in different kinds of http methods.
2. Learned and implemented template driven, reactive forms, angular guards, angular routing.
3. Learnt how to lazy load modules and how to perform custom validations using reactive forms.
4. Email authentication using firebase Auth in angular.
5. How to reduce code using packages ngx-pagination, sweetalert2, file-saver available in node package manager for pagination, alerts and to download file.
6. Learnt different api's and intergrated them in our application like Swagger, Jackson, Apache poi , Opencsv, for logger log4j, firebase, etc.
7. Various technologies like firebase and its database called firestore, also handling custom authentication in firebase.
8. Good Code Practices such as Do Not Repeat Yourself , abstraction in order to simply the code , following camelCase naming convention throughout the project etc.