**Group – 2**

**Common Faculty – Student Interaction Portal**

**Sub Group – 1 : Courses Part.**

1. **Annanya Pratap Singh** (170101008)
2. Arpit Gupta (170101012)
3. Chandra Prakash Meena (160101023)(BACK Logger was never present)
4. Deepak Gami (170101020)
5. **Date: 2nd April, 2019**

* Planning of our subgroup to do Courses done.
* Installation of android v3.3.2 done by all the members.
* Starting to get familiar with android and firebase.

1. **Date: 5th April, 2019**

* Classes for Courses, events, course material and discussion forums made.
* First activity for addition of course made in which professor will first enter the course ID and then we will check in firebase, if it already exists then prompt a message or else take him to a new form

1. **Date: 6th April, 2019**

* Next activity where professor can fill the details of the course I.e. the title, syllabus, marks distribution and description. All these details are added to the database under the courseID section.

1. **Date: 7th April, 2019**

* New activity for adding course material added.
* Menu was added to the top bar of activity and then two buttons of add event and add material were added to it. An alert box will pop up on clicking add material.
* In the alert box we can add the title of class and then select file from a new intent which displays all our files present in the phone. Then we can add the file in storage section of firebase by clicking on add class button.

**Sub Group – 2 : Time table, To-Do-List.**

1. **Aman** **Mishra** (170101005)
2. **Rythum Singla** (170101058)
3. **Vineet Malik** (170101078)
4. Mantri Sandeep Naidu (160101041) (never present)
5. **Date: 2nd April, 2019**

* Planning of project was done.
* All the members installed Android Studio v3.3.2 .
* Daily Timetable of Students and Profs and Exam Timetable module assigned to our group.
* Starting to get familiar with Android Studio and Firebase.

1. **Date: 5th April, 2019**

* Database entries were designed.
* To have a better understanding of firebase, we decided to create a to-do list, an additional feature of our software, before creating the timetable in one go.

1. **Date: 6th April, 2019**

* We decided to create separate database for showing timetable. It includes date as a node, 13 time slots as its children. All the classes being held during that slot on that day shall be entered as value separated by comma. For showing today’s timetable to a person, he will be searched in all the list of enrolled students in all the courses of every slot of the day. This will take O(n) time, where n is the number of users of the app. Hence, it is feasible and practical.

1. **Date: 8th April, 2019**

* There was a brief meeting of our sub-group, in which we decided the format of to-do. we decided to create to-do list of one user, for now.
* **Aman Mishra** created architecture(design) of the new class; created mechanism to write and read data from firebase data.
* **Rythum Singla** created the mechanism to write data in database and basic bug fixing.
* **Vineet Malik** created the feature to delete the task from database and also created an offline to-do-list, so that the user doesn’t have to depend on the internet to see his ToDos.

**Sub Group – 3 : Login Signup , Profile Page etc.**

1. **Parvindar Singh** (170101044)
2. Sri Ram Reddy (170101016)
3. Rohith Ravi Teja (170101017)
4. Rakesh Reddy (170101071)
5. **Date: 2st April ,2019**

* Fixed the problems occuring gradle sync in android studio.
* Members were registered on firebase.
* Created “users” database in firebase database.
* **Parvindar Singh** – Discussed the basic structure of courses , discussion forums, users with other subteam ,Made the the basic layout of Login page and Sign up page and functions. And uploaded on Github repo so that others can clone it and start working.
* **All four team members** Discussed How how the users will be stored in the databasedecided the flow, how the user will login and regsiter and will go to home page or signup page.
* All memebers decided to see some tutorials of using android studio and firebase.

1. **Date: 6th April, 2019**

* **Sri Ram Reddy** modified the UI of Login Page to make it look more Interesting.
* **Rohit Ravi Teja** modified the Layout of Signup page to Make the lyout fit in every screen size and looks more interesting.
* **Rakesh Reddy** tested the login and signup functions.
* **Parvindar Singh – I**mplemented login via outlook , user will not have to enter their information we will get it directly from outlook, initially some error was coming in authentication but later it got resolved. Whenever the user click on **login via outlook button** if there info is not present in the database then they get registered, if there info is already present in database then they get logined .Also also decided to do manual login and signup, via storing user information in firebase database. Created a UserInfo Class which Contains Public Static variables to store logined user information so that it can be accessed from any other class. Created Profile page for user filling it with basic user information.

1. **Date: 8th April, 2019**

* Login via outlook successfully working.
* **Parvindar Singh –** Made a profile page of user with Showing their basic info of the user from there outlook acount and profile picture.Having difficulty in decoding profile pic from outlook. So added sample profile pic for the time.
* **Rakesh Reddy** Added a list of enrolled courses of the user in Profile page , if we click on the course then we go to that course page and modified the profile layout and Modified Home Page a little .
* **Sri Ram Reddy and Rohit Ravi Teja** Modified the Home page to create the function of logout button.

**Sub Group – 4: Courses Feedback**

1. **Devaishi Tiwari** (170101021)
2. Aayush Patni (170101001)
3. Nayanjyoti Deury (170101042)
4. Rashi Singh (170101052)
5. **Date: 2nd April, 2019**

* Planning of project was done.
* Feedback was assigned to our group.
* All the members installed android studio.

1. **Date: 6th April, 2019**

* Database entries were designed.
* We decided to create separate database for feedback questions. It includes course-wise questions and their options and students that have filled it.
* This will ensure that a student is able to fill the form only once.

1. **Date: 9th April, 2019**

* There was a brief meeting of our sub-group, in which we decided the format of teachers’ feedback page.
* We made the teacher’s feedback page where teacher can dynamically choose the questions for each of its course.
* We have planned to finish the teachers and students feedback page till next Tuesday.

1. **Date: 15th April, 2019**

**Rashi and Devaishi**

* Populated the existing database and modified the structure a little.
* Made the students’ feedback form and worked on dynamically retrieving data from database
* Made some modification to teacher's feedback page and worked on dynamically adding nested data into the database.

**Nayanjyoti and Aayush**

* Planned the homepage for teachers' feedback and learned to insert bar graphs and pie charts.

**TOTAL CONTRIBUTION**

After milestone 2, due to the size of features, we decided to shuffle our subgroups. We worked individually to avoid conflicts. Contributions of Individual members are :-

**Parvindar Singh (170101044)**

1. Created Login, Signup, Login/Signup **via** **Outlook**, Profile page.. Described above before Milestone 2.
2. I helped **Rohit** and **Sriram** in designing the Login Screen by setting its constraints and color of layout.
3. Created an **in-app Messaging** Feature, any person who is registered on this app can send messages to anyone who is registered on this app.  
    Created a message list activity which regularly updates the incoming and outgoing messages.( It updates whenever there is a change in the messages part of the user in the database, so it is **real time**.)

Created a custom list adaptor which shows a specific layout in the message list of the user ( that layout contains Subject, Sender, Receiver, Time, body(a little part of the body if the body is long.), and a message direction (it is sent or received)). The layout will contain different information for different messages and inserted in list view.

Created a message box activity, on clicking on any of the messages in the message list, the user will be redirected to the message box and all information of the message will be shown there( similar to email). From there, the receiver can delete, reply or forward that message to another person.

The unread messages in the message list are highlighted, when the user reads the message then that message in message list gets unhighlighted.

Implement toast notification whenever a user receives a new message.

Implemented push **notification** with **Ananya** for a newly received message.

1. Created an **in-app Polling** Feature.

Each course has its own Polls section. Any user (Student enrolled or Professor) of that course can Start a Poll, and anyone can choose their options. One user can choose only one option.

Polls layout was made by making **subclass options inside Polls class** and custom list adaptor. then inserted the Options layout into list view by that custom list adaptor.

Creating a poll will be flexible as a user can insert any number of options as he/she wants (still not more than 50 options.)

The database of Polls Stores the username of the user who started the poll. And it stores username of users who have polled for any option with options they have chosen, So, a user can also change his/her choice and select a new option.

Poll for any question **can be closed** by the creator of Poll or Professor.

Implemented **swipe to refresh** feature in polls layout so that votes refreshes every time user swipes down. Votes also **refresh** whenever a user selects an option. ( there was an issue in updating the list view of options layout in **real time**).

1. Also created the **auto-login** feature. If the user has logged in, then he will only be logged out when he manually does log out by selecting log out option in the nav drawer.

I am storing the username of logged in user in **shared preferences** inside the app (also storing a boolean if the user is logged in).

1. Also created a **remember username** feature. Whenever a user logout, next time he/she opens the app, the username of previously logged in user is already entered in the username field of the login screen. This feature is also implemented using shared preferences.
2. Helped **Aman** in writing code for enrolling in a new course, showing course list.
3. **I and Rashi** merged the Feedback part ( took around 5hrs. ) into the main branch initially. later by pulling the latest update from the main branch, some modified code got erased and later I, Arpit and Ananya merged it properly in 2hrs.
4. **I and Arpit** converted some **Activities to Fragments.**
5. Was involved with **Arpit and Ananya** in **Merging** different modules into the main branch.
6. Was Involved with **Arpit and Ananya** in **Fixing Bugs**. We fixed bugs of our codes and other modules’ codes. Ananya has described major bug fixes below.
7. Made the basic structure of the **about page** with **Rohith and** **Sriram**.
8. Was involved in testing and debugging the app on last day with other team members.

**Arpit Gupta (170101012)**

1. Created the course add part. Where the prof can enter courseID, if it exists a new ID is demanded if not then prof is redirected to course add content part. This part was done by me and Ananya.
2. Made the add course add content part. Here the prof can enter the course name, description, syllabus, marks distribution and enrollment key. This much part was done along with Ananya.
3. In the same page addition of time slots, midsem and endsem dialog parts.
4. Made the course main page for prof. Here the professor can see all the details he entered in previous page while registering the course. Professor also has the option of editing the description, syllabus, marks distribution and key by simply long pressing on the text boxes after which he can edit the feild and click on update button.
5. Made the add material section of this page. Here the prof can enter the title of class and also upload the file. At the time of uploading a notification is also shown. The added material is shown in list below in the page.
6. Made the add event section of this page. Here the prof can add the title, description, date, time and select the type of event. All these events and materials are also visible to the students who are enrolled to this course. **The alert dialog box codes used here were later on used in other parts of the project.**
7. Used context menu to allow professor to delete events, materials and course projects by simply long pressing the section you want to delete.
8. Wrote the code to allow the user to download the file from materials and project section by simply pressing the item in the list.
9. Made the basic layout of this (course main page professor) page. The same layout was also used in student information page.
10. Made the student information page which quite similar to the course main of professor.
11. Made the basic layout of app which was late on taken up by Deepak Gami and furnished in detail by him.
12. **Merging of various modules was done by me, Annanya and Parvinder.**
13. **Bug fixing was also done by the three of me, Annanya and Parvinder.**
14. Testing and debugging of the app along with all others who were present on last day.

**Annanya Pratap Singh Chauhan (170101008)**

1. Created the add course part with Arpit. This page allows the user to enter a Course ID . The Course ID is checked in the database. If there is a conflict than we don’t allow the Prof to add a course.
2. Created the basic add course content page with Arpit. This allows the prof to add the details about his or her course.
3. Made Add Project part of Course main page prof. This allows the Professor to add Project to his or her course. He can give a submission deadline. The file is uploaded to the database and the file can be downloaded by any student enrolled in the database. Added Progress bar to the upload progress for the files for both add material and add project. Made the basic layout for showing the course project part. Added the dialog box to add a project
4. Made the whole Discussion forum. The discussion forum works on the basis of Thread Replies format. Created the Threads page. Any student or professor part of the course can Create a thread. The professor is given power to either delete a thread or close a thread. If a thread is closed than you can no longer reply to a thread. I used context menu to show delete thread and close thread options. **This code was later borrowed in different part of the project.**
5. Made the Replies Page of the Discussion Forum. You can click on any thread that you want to reply to. This directs you to the reply page. ANY user can post a reply to thread if it is not closed. The replies are update real-time.
6. Merged the To-do page. This is displayed on the main page. Any user can create a to-do list. This is saved in the database. A user can delete a TODO if he wants. He can also update a to-do. The delete and update options are given using a context menu.
7. Added Notifications throughout the project. Added progress bar in uploading a file.
8. Added notifications which are triggered whenever a user receives a message.
9. Was involved with **Parvinder, Arpit in fixing the bugs** throughout the projects.
10. Was involved in **merging the project with Arpit and Parvinder**.
11. Was involved in testing.

**BUGS FIXED (Project Level) AND MERGING (done by Annanya, Arpit and Parvinder)**

1. App kept on crashing after the code was converted to fragments. This was happening whenever we were updating any listview. It was due to the fragment lifecycle. Whenever a fragment is destroyed it takes some time to return a view or activity. The problem was that whenever we updated value for a list view the data change was triggered immediately and the view was also destroyed. But the data change function required some view which was null at that time. This was leading to null pointer exception. We rectified this error by checking for null value of getView or getActivity before loading a list view on the page.(time taken: 3hrs).
2. There was some error in the create course part as we were adding the course id whenever a prof entered a course id to add course page. But the problem was that once the course id is added without its course content, it is displayed in all courses list part of the project. If any user now clicks on such a course from this list, the app crashes. (2 hrs)
3. There is a problem with using context menu on different fragments of the same activity. One of the context menu will stop working out of no where. So to sort this error you have to account user visibilty also. So we needed to use getUserVisibiltyHint() in one of the fragment’s context menu so that we can use context menu for multiple fragments inside an activity.(1 hr)
4. Notification was not showing properly in Android 8 and above. This was due to the fact that we need notification channel in Android 8 and above. So we modified the code so that it can run on android 8 and above. (1.5 hr).
5. We were not able to download files in every device that we had. It was happening in phones with API level less than 27. This was rectified by using descendantFocusability="blocksDescendants"(1 hr).
6. **Merging of the project**. We merged all the modules .We resolved all the connectivity issues in the app.

**Aman Mishra (170101005)**

1. Created code and layout for timetable feature. It takes date from user in date-time picker.
2. Created code and layout for Search in all the courses. It gives output in list.
3. Created code and layout for course description page which is accessed after clicking on search results.
4. Created code for enrolling a student in course.
5. Created code and layout for events fragment in homepage, code for dialog box of brief description of event.
6. Created code and layout of delete course feature.

**Deepak Gami (170101020)**

1. Design basic flow of the app which defines what activity or fragment will open one after another. This was the basic structure which then eventually filled up with proper components developed by other team members.
2. Create the homepage of app, the first thing which will be shown after login (student or Prof.)
3. Added redirect to Outlook feature in the app. Using this user can directly redirected to their outlook app if installed already or else a dialogue box will pop up showing “app is not installed”.
4. Wrote XML code for Navigation bar and for different other layouts and customized buttons.
5. Design layouts and set constraints of most of the activities which make them user friendly. Now app could load properly on any screen resolution.
6. Different XML files like style, color and drawable were updated.
7. With Arpit, merge all the layout files developed/edited by me in the main project.
8. Debugging and testing of final App with other team members.

**Rythum Singla(170101058)**

1. “**Todo list**” module (code and layout). It adds your todo item and can also edit any previous todo or delete it. Each user has its own todo list.
2. Merged the todo list with Annanya in the module. Some changes were made as my module used recycler view and other used list view.
3. Made the layout for “**Feedback form**” , basic templates how it would look on both professor side and student side.
4. Created the database structure for feedback form with Rashi and implemented count for each option created by professor and feedback given by student.
5. Implemented the feature that no student will be able to fill the feedback form twice.
6. Done the **testing and debugging** for the feedback form
7. Also done testing for the whole app along with other team members.
8. Me and Rashi merged the Feedback form with the other part of the app.

**Shri Raam(170101016)**

1. Modified the layout of **Login** and **Signup** page with Rohith. To make it look nice
2. Modified home page to create the function of **logout button**
3. Helped Parvinder in creating **in**-**app polling** with Rohith (excluding swipe to refresh option). A professor can create a poll in any course, and students of that course can vote on any one option in that poll. This poll will show the total no of voters, votes and percentage of votes for each option. A professor can also delete a poll if he wants to (this makes the poll to not accept any more votes)
4. Created the **About page** with Rohith. Here we can see a very brief description of the app. And also the total team members with photos, names and e-mail ids.

**Devaishi Tiwari(170101021)**

1. Worked on finalizing the structures and features of the feedback module along with planning the nodes to be added in the database.
2. Worked with Rashi on “Create Feedback Form” and “View Feedback Form” features of the faculty’s feedback form till Milestone2.
3. Did extensive layout and execution testing of the various prototypes of the software along with many other group members.
4. Completed the student as well as faculty documentation for the whole software in a structured way along with the images of running software

**Rakesh (170101071)**

1. Course homepage and additional Layout,
2. For course homepage layout appropriate tools and usage of tools according to time
3. In course addition button to add materials and events so that prof can add materials link to download
4. FAQ : come up with idea of putting buttons in faq which make it interactive and select which questions must be the part of faq. Depending on logged in user, different sets of questions will show up for students and professors.
5. UML Diagrams:

Created UML for project .There are basically 4 diagram about project .

1.class diagram

2.user case diagram

3.state diagram

4.sequence diagram

Each diagram explain well about project. By diagram I have shown activity flow of app from login page to homepage , feedback , FAQ, course ,timetable ,events page.

Class diagram shows what objects are avaiable to user

1. Technical Doc:

Contain details view of backend of this app .at tools and flow of functions are well explained based on condition user do and all activities are well explained . Database sructure usage for each activity is explained well.

**Rohith (170101017)**

1. Modified the layout of **Login** and **Signup** page with Shri Raam. To make it look nice
2. Modified home page to create the function of **logout button**
3. Helped Parvinder in creating **in**-**app polling** with shri raam(excluding swipe to refresh option ) . A professor can create a poll in any course, and students of that course can vote on any one option in that poll. This poll will show the total no of voters, votes and percentage of votes for each option. A professor can also delete a poll if he wants to (this makes the poll to not accept any more votes)
4. Created the **About page** with Shri Raam . Here we can see a very brief description of the app. And also the total team members with photos, names and e-mail ids.

**Nayanjyoti (170101042)**

1. Made technical doc.
2. Helped in the initial stage of feedback module.

**Vineet Malik (170101078)**

1. "Exam Timetable" module (code and layout). It shows the exams of the user for the courses he is enrolled to, in the sorted order of the date of the exam. The user has options to select the type of the exams (Mid Semesters, End Semesters), for which he wants to see the timetable.
2. Exam part of “Create Course” Page. Every course needs to have exams (Mid Semesters, End Semesters). While creating a course there are dialog boxes in which the user can fill about the details of the exams.
3. "Offline To-Do-List" module, shown during milestone one. Later on, the idea was shifted to online To-Do-List created by Rythum and Annanya.
4. Basic "Course Feedback" module, presented during milestone two. It showed Questions present in the database which appeared one after the other using the Prev and Next buttons. This version was not deployed**.**
5. Final cleaning of the database and populated the final database with valid data.
6. "App Logo" and testing.

**Rashi Singh (170101042) and Rythum (170101058)**

**Feedback Module.**

Module Description

Feedback module aims at creating a uniform yet personalized feedback for each course to

frequently assess the course and smoothen the student – teacher interaction.

Activities in a nutshell

1. Main feedback page for professors. This contains three options:

A. Create feedback form –

➢ This consists of customizable model questions, which are displayed

dynamically. The professor can not only choose questions, but also the options

eliminating the obsolete ones.

➢ The professor can also click on add button to type in the personalized questions

and options as many times as he wants.

B. View feedback questions -

➢ This shows all the questions added to the form. It also contains a button to

redirect the app to the add question page.

➢ It also contains a “float" option to finalize the form.

\* We are planning on adding delete and edit functionalities for feedback form at

this stage

C. View feedback status – This shows the stats of the choices filled by the students in

the form of pie chart.

2. Main features for students -

A. Fill Feedback Form – This form is only for the students, where they fill their choices

and submit, redirecting them to the View feedback status form.

Contributions and logs -

• 5-6 April (1 st milestone) -

 Decided the modules in the app and the database to be used.

 team division.

• 7 April - Decided the features to be implemented into the feedback module

 Decided the layout of the firebase database, to implement separate yet

personalized feedback forms for different courses and users.

Started learning more about android studio and json database format.

• 8-9 April (1 st milestone) -

 Implemented the first form (create feedback form), using dynamically

allocated checkboxes and personalized questions.

If a user decides to add a greater number of model questions, it can be

 implemented without even writing a single extra line of code.

This further included the feature of being able to select “optional” options for

any feedback question.

• 10-11 April –

Learned more about:

− recycler form

− list views

− reading and writing from database

Implemented add question form, using which the teacher can add his own

questions into his feedback form

• 15-16 April (2 nd milestone) -

 Implemented the View Question form, which made use of simple technique of

reading from the database and viewing it in the recycler view.

• 17 April 2018 -

 Implemented the Student Fill Feedback Form, which included the following

problems to be solved -

− Show the variable number of options in the fixed number of textboxes

− Show the selected option on clicking prev/next buttons, for the user to

review his options.

− Fixed the null-pointer exceptions and out of index exceptions for prev

and next button of the form

• 18 April 2018 –

Implemented some other bug fixes, including nested activity instances and

exception handling for Student fill Feedback form including corner cases for

submission of feedback form and multiple submit button clicks to improve stack

performance.

Implemented the update of respective feedback stats on choosing an option in

Student Fill Feedback form

* 19-20 April 2018 -

 Implemented the pie chart for the View feedback stat using PhilJack Github

 package for Stat Charts for android studio.

Made the UI for the View Feed Stat form.

 Fixed the bugs for next/prev buttons for the View feedback stat form,

including out of index null exceptions and multiple button clicks to improve

stack performance.

Started merging with the main Student – Professor interaction Portal github

app. Fixed the merge errors and missing code snippets from gradle files and

some other forms form android studio.

 Implemented conditions for “floating” the feedback form and not letting the

student fill a form twice.

 Remaining debugging and testing was done along with the rest of the team.