



**The Faculty of Information and Communication Technology
Mahidol University**

ITCS424 Wireless and Mobile Computing

Presented by

Saranphon Phaitton 6088114

Supakarn Wongnil 6088137

Komolmarch Treechaiworapong 6088213

Section 1

Presented to

Assoc. Prof. Dr. Vasaka Visoottiviseth

This document is submitted in partial fulfillment of
the requirements of the Wireless and Mobile Computing Semester 2, 2019-20

The google drive link :

<https://drive.google.com/file/d/1MaFRD4OwLkdG6fpuCM6wJvGj6yLEEVvg/view?usp=sharing>

Github link : <https://github.com/maijsp/penquiz>

Selected Topic : Learning material to help users learn the computer network class, especially TCP/IP and OSI models, ARQs, Routing protocols, TCP congestion control, including quiz, and solutions.

PENQUIZ - Application Overview

The application will basically give the users the best practice for computer network content within the area specified above. Firstly, the user needs to create his/her profile in order to keep the record and the best score. Then, the user will choose a quiz which they want to do. After they choose, the application will show the content list in various topics of computer network content. To make the user access quiz quicker, the application has a search feature to suit this. Another feature is that users can keep tracking his/her best score for a quiz, the application will allow users to track the mood when they finish the quiz. The application supports both English and Thai.

The following are application features included in the PENQUIZ mobile application

1. Quiz Time

- The application will provide the computer network content. The level of the quiz would be simple and give basic knowledge to the users. The quiz will have one screen per question.

2. Score calculation with reviews

- The score of the quiz will be calculated at the end of the quiz.
The application will show a quiz score as percentage in the dialog box and display the number of correct/wrong answers.

3. Searching Quiz

- The application has the search box for the users to search the typical name of the quiz. If users search TCP model, the quiz about TCP model will show in the application. This will allow users to do the quiz they want.

4. Create your profile

- The application can create the profile and allow a user to change language in the application.

5. History Page

- The application will keep tracking the best score of each quiz. This will allow the users to challenge themselves.

6. Contact us

- The application allows a user to contact or give feedback to the developer team of this application. It also displays the location service of our headquarters on a map.

Wireframe

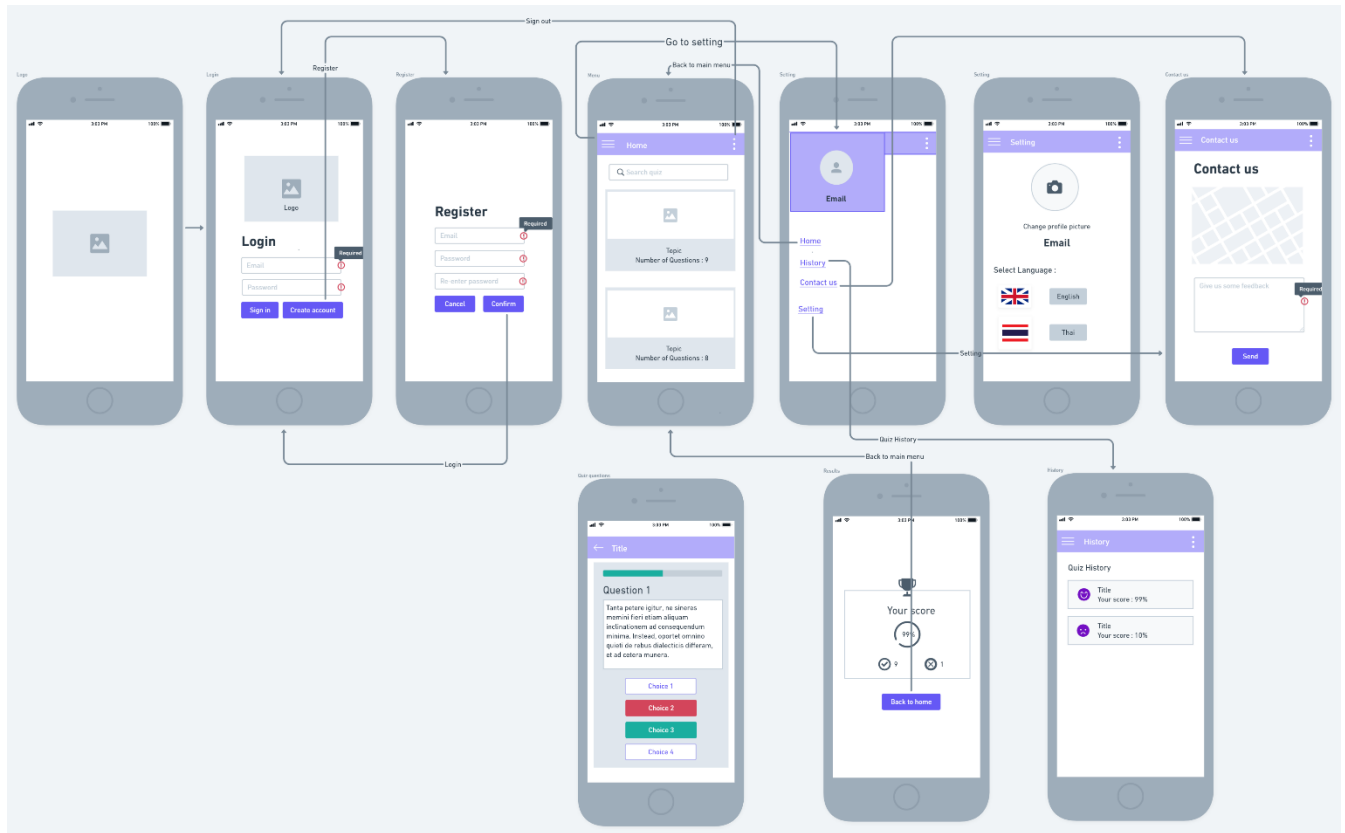


Figure 1. Wireframe of Penquiz Application

PENQUIZ - Checkpoint Overview

In the previous checkpoints, we have created activities including a login system and register system, home page, contact us page, setting profile page on the PENQUIZ application and keeping data on the firebase. In this fifth checkpoint, we have completed activities consisting of implementing a home page, implementing contact us page, implementing a setting page, implementing quiz question page, history page, and implementing setting profile page. Additionally, the PENQUIZ application includes all required features stated in project description, and also adds extra features. The following show features we have completed.

- **required features in the application**
 - an application support both Thai and English language
 - an application provides user authentication and registration
 - store information on Firebase database
- **extra features in the application**
 - linking to camera : to set user's profile
 - linking to network : to send and receive data from/to the firebase
 - Google Map : to display location of the application developer's company
 - Dialog : to display dialog message when users require/complete action
 - Recycleview or cardview : to display quiz topic list to users from API
 - Other advanced Android features
 - Animation when viewholder is created – HomeFragment
 - Use Circular and Normal Progress Bar by Github (<https://github.com/akexorcist/Android-RoundCornerProgressBar>)

The objective of the project

This project aims to encourage users to learn the material from the computer network class by doing an online quiz on the mobile application. The users must create an account for the first login before using an application so the users can access an application to do online quizzes. Once users login to the system users can select any topics that display on the Home page. All quiz questions will be pulled from the database, firebase, and display on each page while users are doing a quiz. Moreover, users can see the final score when they finish doing a quiz. After users complete the quiz, they can give any feedback or comment about the system to the administrator on the Contact Us page such as provide more questions of each topic.

The application designs

- **Register System**

- Before users can access an PENQUIZ application to do a quiz, users are required to register but if users have an account already, he can directly access an application via login system. In the registration system, we decided to connect an application to a local database, Firebase, to keep information of registered users, questions because it provides us with a lot of functionality and an easy to manage database. We create an interface of this action which consists of three main elements including username, password, and confirmed password block. Moreover, it provides two buttons including the cancel and submit button. On this interface, users must fill these three blocks after users complete filling information, users must click submit button otherwise clicking cancel button. When a submit button is clicked, all filled information will be sent to Firebase. Once users have registered, they can skip the login step.

Emergency calls only 100% 6:03 PM

Register

test@gmail.com

.....

.....

CANCEL CONFIRM

Emergency calls only 100% 5:29 PM

Register

Email

Password

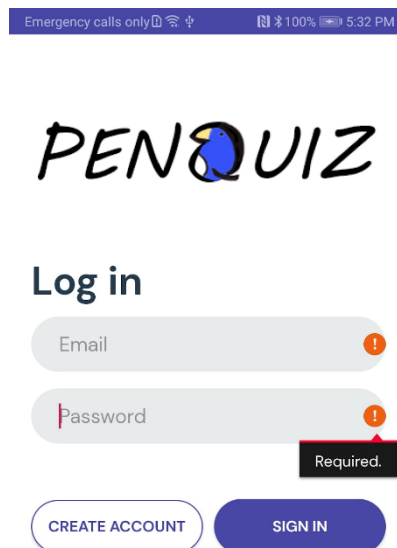
Re-enter password

Required.

CANCEL CONFIRM

- Log in System

- Since users register, the firebase has already kept the information of the user ,including email and password. Therefore, when users open the application, they can type email and password that they did register before. If users type the username and password wrong, users can not sign in. In contrast, if user type username and password correctly. Next, users can click the sign in button to go to do the quiz. In addition, the interface of login system, it will show that user need to fill the username and password for using the quiz service.



Emergency calls only 100% 5:32 PM

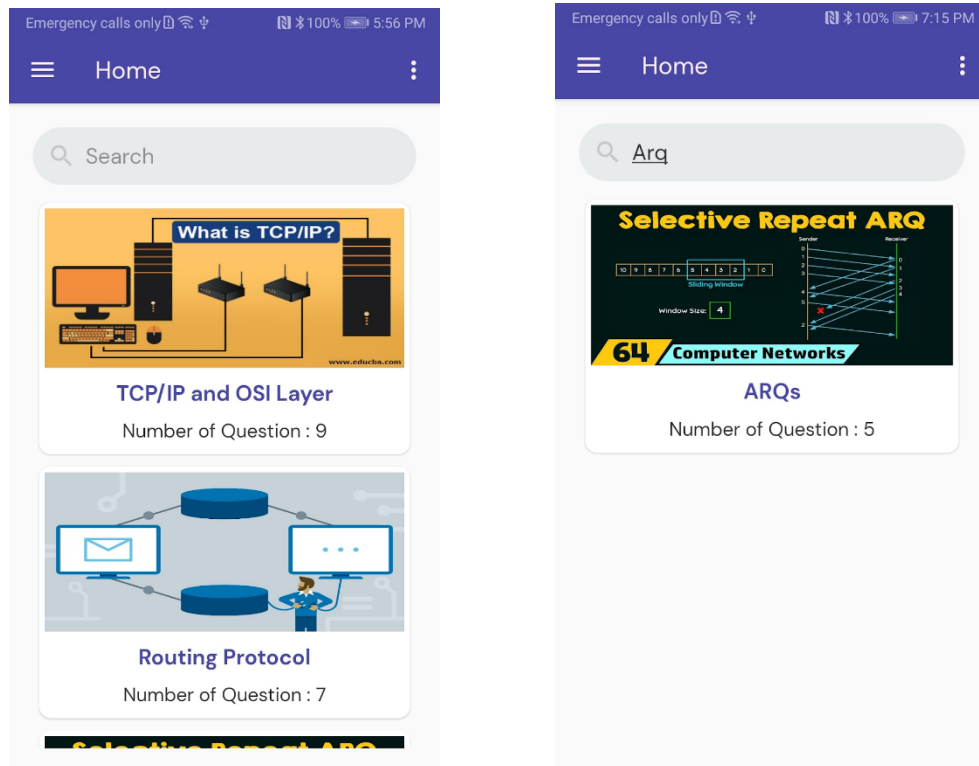
PENQUIZ

Log in

Required.

CREATE ACCOUNT SIGN IN

- Home Page
 - show the quiz objects for users to choose the quiz they want to do. Moreover, users can search the quiz that they require to do



1. This fragment contains a [RecyclerView](#) with [Quizes](#) object retrieved from database by

```
quizesRef=FirebaseDatabase.getInstance().getReference().child("Quizes")
```

Table 1.

2. We `fetchData()` by using [FirebaseDatabase.getInstance\(\)](#) to retrieve data snapshot by snapshot
3. We use [onCallBack](#) from interface we created to bind data to [dataLi](#)

4. Therefore, we use [QuizAdapter](#) to bind the [Quizes](#) in to view object.

```
fun bind(quiz: Quizes) {
    itemView.apply {
        container.animation = AnimationUtils.loadAnimation(context,
R.anim.fade_scale_animation)
        quizimage.animation = AnimationUtils.loadAnimation(context,
R.anim.fade_transition_animation)
        var image:ImageView = findViewById(R.id.quizimage)
        Picasso.get().load(quiz.imageUrl).into(image)
        quiz_name.text = quiz.title
        quiz_detail.text = "Number of Question : ${quiz.num.toString()}"
    }
}
```

Table 2. This code snippet show how we bind Quizes object to view

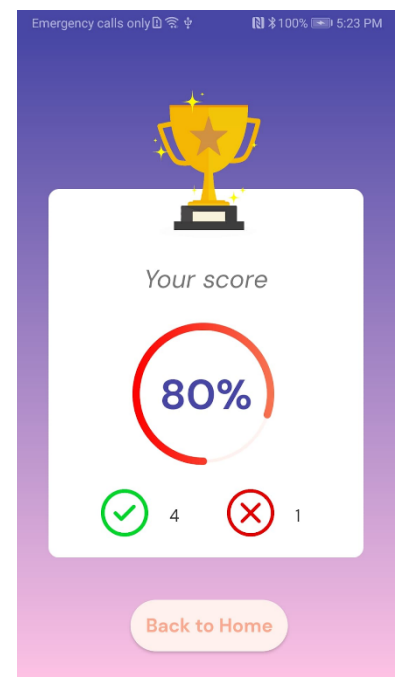
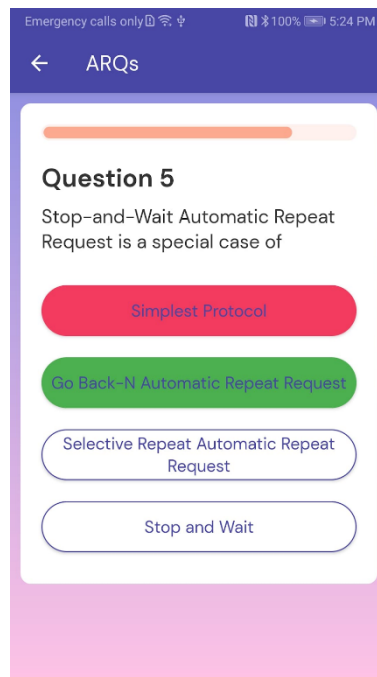
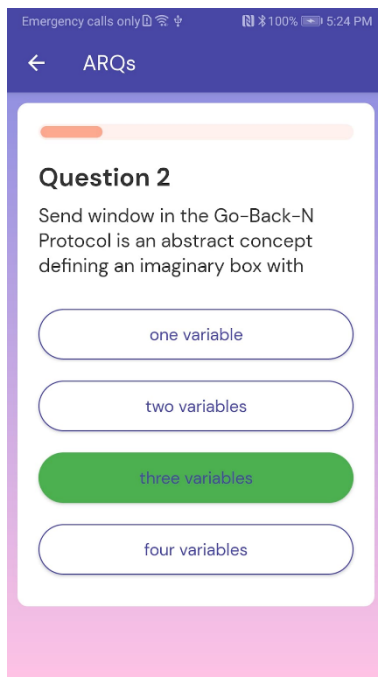
5. In addition, we also perform search filter to filter the result corresponding to the input from users by implementing [Filterable](#) interface

```
override fun getFilter(): Filter {
    return object : Filter() {
        override fun performFiltering(constraint: CharSequence?): FilterResults {
            val key: String = constraint.toString()
            if(key.isEmpty()) {
                myDatasetFilter = dataset
            }
            else {
                val listFiltered: ArrayList<Quizes> = ArrayList()
                for (row:Quizes in dataset) {
                    if(row.title!!.toLowerCase().contains(key.toLowerCase())) {
                        listFiltered.add(row)
                    }
                }
                myDatasetFilter = listFiltered
            }
            val filterRes: FilterResults = Filter.FilterResults()
            filterRes.values = myDatasetFilter
            return filterRes
        }
    }
}
```

Table 3. Filtering results

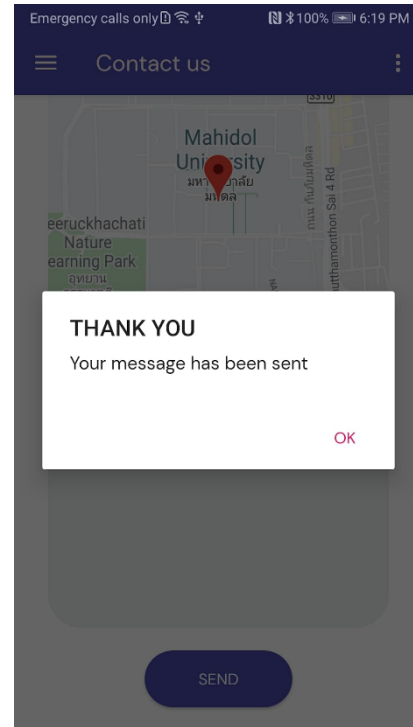
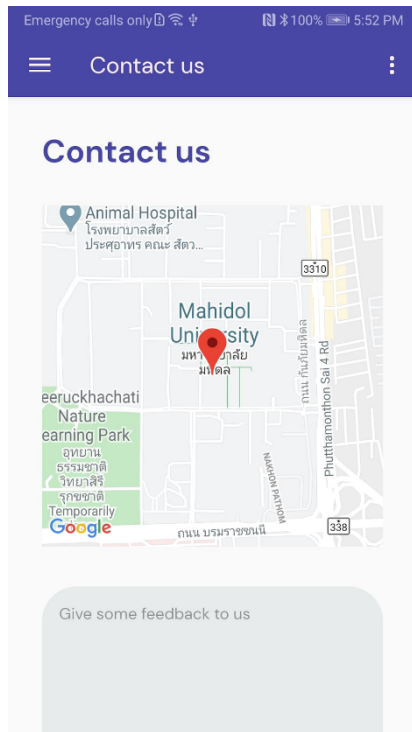
- **Quiz Questions Page**

- show questions according to users' selected topic in a home page. Each topic provides a different number of questions. Each question page will display question text and question number at the top page as well as display 4 options with the description in square block. Moreover, users can see their progress from the above progression bar. When users select an option, it will navigate users to the next question page, and also display the green block if the answer is correct. Otherwise, it will display the red block and show the correct answer. After users finish all questions, the quiz score will be calculated as percentage and displayed on the next page. Also, the number of correct/wrong answers will be shown.



- **Contact us Page**

- show the application developer's company and allow users to give feedback to them.



- **Google Map API**

Contact us page will mainly show the location of the application developer's company, Mahidol university, by using Google Map API. In order to create a Google Map, we need [google_map_key](#) given by Google.

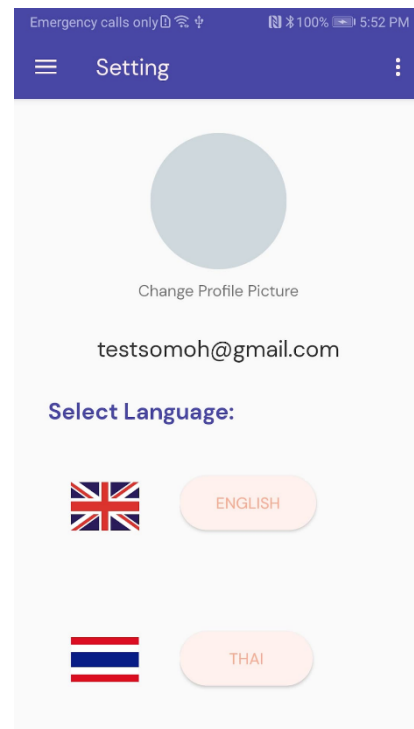
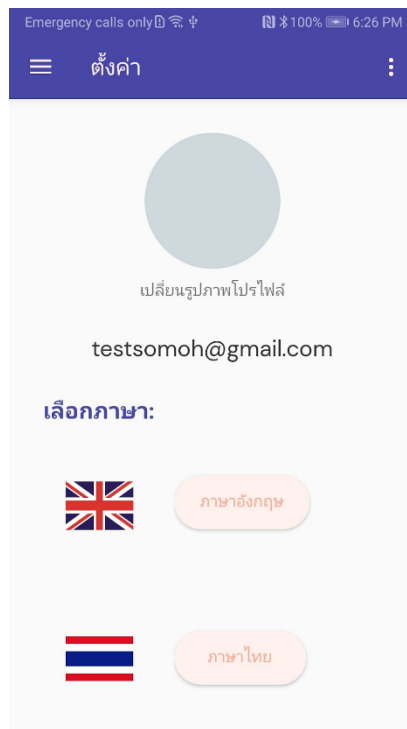
- **Giving feedback**

Users are able to give feedback to application developer's by typing in the text box below the Google Map and clicking the submit button. After the submit button is clicked, the alert dialog box will be shown to display a thank you message. However, if the users click the send button without input any message, the message block will alert an error to notify users to input some message.

- **Setting Profile Page**

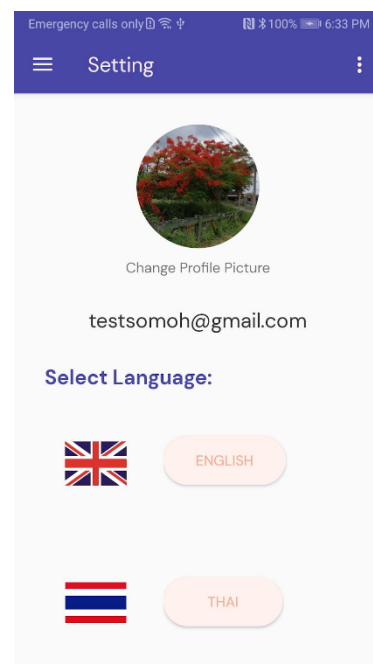
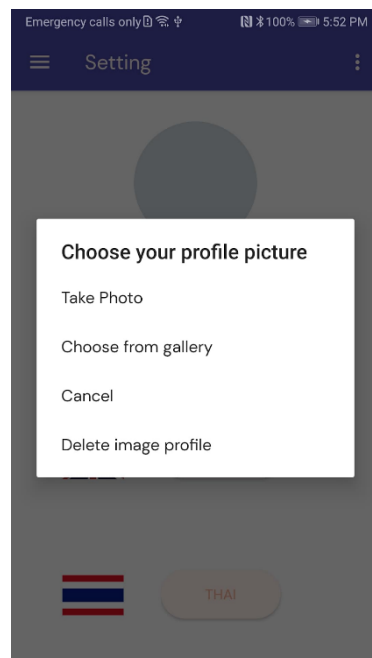
- This page shows the profile of the user. Users can set the language and picture of the user's profile.
- **Language**

The application provides multiple languages which are English and Thai. The user can select the language by pressing the button as the user requires. After the user has finished the setting, every page will be changed based on language selected and saved automatically.



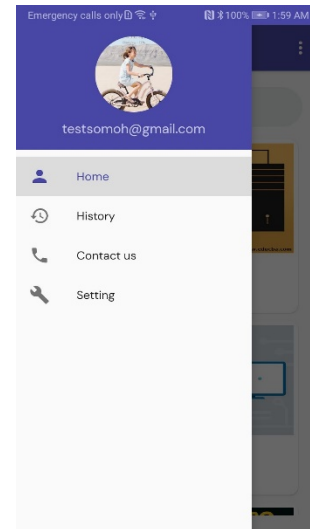
- **Profile Image**

- Users can set their image profile on the settings page. User clicks on the grey circle image, it will ask the user for selecting a photo from the gallery or taking a photo and the user can cancel if they don't want to do this process and it will save it automatically. Moreover, users can delete the image that they uploaded. Users' images will be stored on the Firebase so every time users access the application. Their image's profile will be returned from firebase automatically.



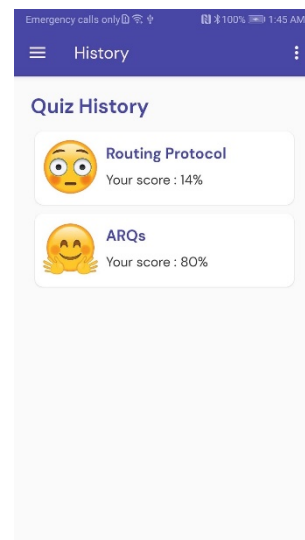
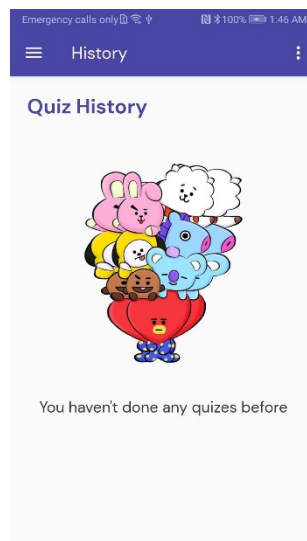
- **Navigation Bar**

- a left bar that links to appropriate sections/pages in an application helps users in traversing the online document. Users can go to each menu including home page, setting page, and contact us page. Moreover, users can see their profile image and email on a navigation header.



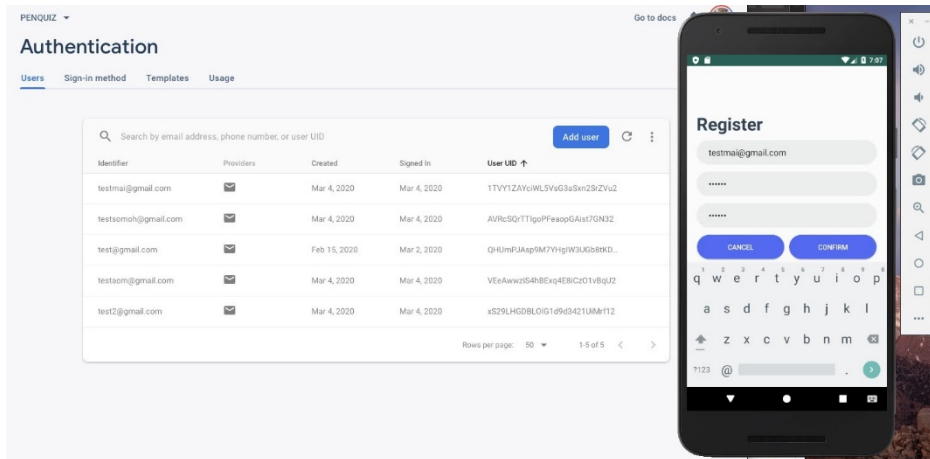
- **History page**

- this page will show the quiz score report of users when users complete doing a quiz. If users haven't done any quiz yet, the history page will not show anything. However, if users have done the quiz, it will display the latest quiz score of each topic. Moreover, It has emoji to show the face based on your score of the quiz.



Database design

- Users' Authentication
 - When users register an account, they have to fill their email and password for authentication and their information will be kept on the Firebase database.



User	
PK	<u>userid</u>
	username
	password
	signin_date
	create_date

- Feedback from user
 - When users write some feedback on the mobile application and click submit button, all feedback will be transmitted and stored in the firebase database. On the firebase, there is a root called Messages that will save the real time feedback from each user. Every time users send feedback, each feedback will be pushed into a sender ID branch and feedback will be saved according to their unique key (Figure 1). The unique key of each user is generated once users register for the first time. For example, this unique key, 1TVY1ZAYciWL5VsG3aSxn2SrZVu2, is generated for the first user.

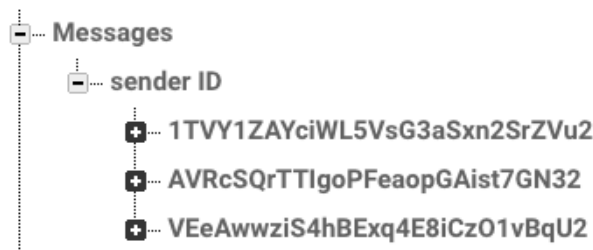


Figure: 1 display path storing the real time feedback according to unique key of each user

- To push the real time feedback object in firebase database, the feedback object will be saved to the specific database reference by write operation like,

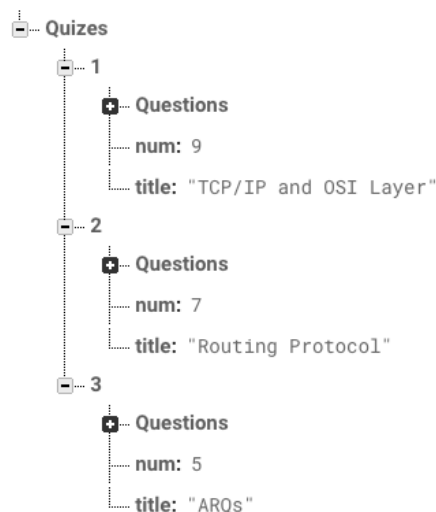
```
myRef.child("Messages").child("senderID").child(senderID.toString()).push().setValue(message)
```

Table 4.

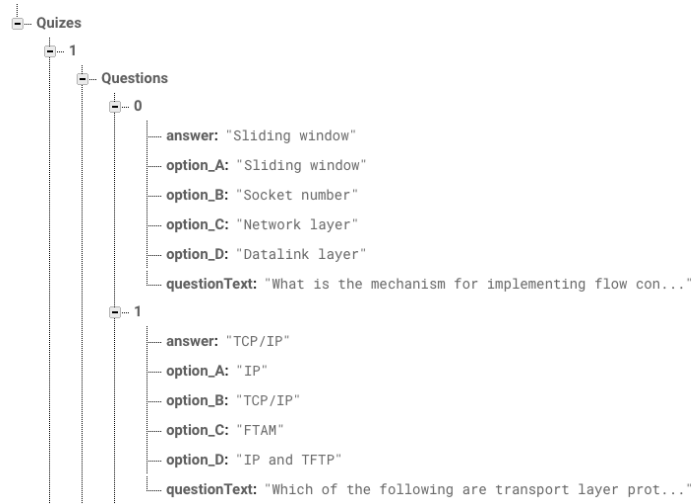
- When the feedback object is saved in the database, the object will be mapped according to specific path location in a nested fashion.



- Quiz questions
 - our group stores questions about the computer network area including TCP/IP and OSI models, Routing protocols, and ARQs on the firebase. On the firebase, it will have another root to retrieve quiz questions called Quizzes. This root contains three subbranches to represent the identification(ID) of each quiz topic. For example, the TCP/IP and OSI model topic has ID 1, routing protocols topic has ID 2, and ARQs topic has ID 3. With this database design, it helps us to retrieve the questions of each topic or information of each topic easier when we want to read content from the firebase and display it on the application. Moreover, each subbranch consists of three child nodes which are the important information of each topic including Questions, total number of questions, and question topic.



- In the Questions child, it contains the identification number of each question ranging from 0 to the last number of questions. Each identification number consists of several descendants of child nodes including question text, choice from a to d, and correct answer to that question. Therefore, we can retrieve any information from the firebase by following the path that we specify on the firebase.



• Users' record

- our group will store users' information on the firebase database based on user id. The root called Users will include user id and each user id contains two child nodes including history path and image path.



- The history child node will contain information about quiz score and quiz title that users have completed. Therefore, when users complete doing a quiz, their latest score will be stored in this path.

- Also, their image profile(URI) will be stored in **the image child node** of Users root. There are two types of image files that are stored in the firebase. First, if a user takes a picture for setting the profile, it will bring the image path of the taking image to upload in the firebase storage and generate URI for later use. URI will be stored in the firebase database too. Then the system will bring the URI from the firebase database to set in the profile. Second, if the user selects the image from his/her gallery, the image that the user picks up, it will store in the firebase storage and bring the URL like taking an image from the camera. To retrieve URI of an image from a firebase database, we use Picasso to set the user's profile image.
- To update images of users to firebase storage, we create firebase storage reference to generate URI of image and set URI value based on user id to firebase database at specific path.

```
var uid= FirebaseAuth.getInstance().currentUser?.uid
var reference: StorageReference =
FirebaseStorage.getInstance().getReference().child("profileImage");//.child(uid+
".jpg");
var uploadTask = reference.putFile(uri!!)
uploadTask.addOnSuccessListener{
    reference.downloadUrl.addOnSuccessListener {
        mDatabase.child(uid!!).child("image").setValue(it.toString())
        Toast.makeText(activity!!.getApplicationContext(),"success!!"+it.toString(),To
ast.LENGTH_SHORT).show()
        Log.d("DIRECTLINK",it.toString())
        //Picasso.get().load(it).into(imageProfile)
    }
}
```

- To retrieve a user's image profile, we get the URL from a specific path in the firebase database.

```
val image = snapshot.child("image").getValue(String::class.java)
Picasso.get()
    .load(image)
    .into(imageProfile)
```

The system architecture

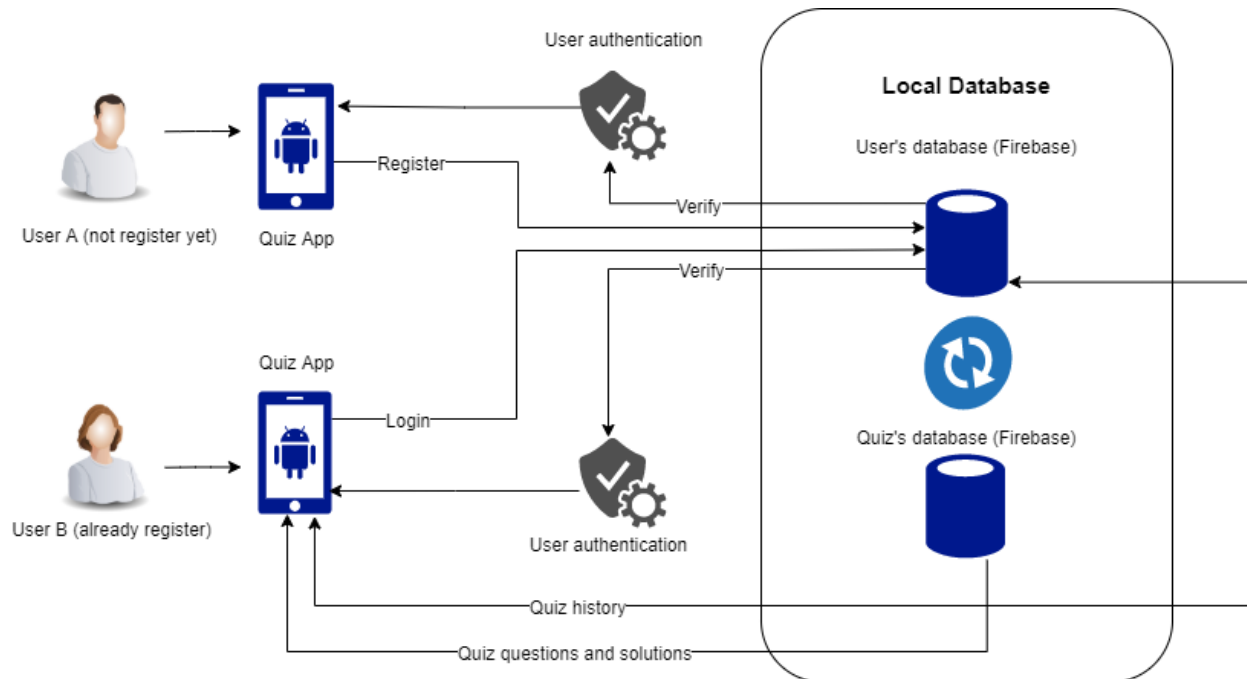


Figure 2. The system architecture

From the picture(Figure2), it illustrates the system of quiz service. It has two cases of users which are user A and user B.

1. User A (who does not register yet)

- Firstly, users must register before using an application, the information of the user required are username and password. Therefore, the system will create the user's database on firebase and bring that information to keep in the database. Next, the system will tell the users to do the users authentication and then the users can log in now. After that, users can access the quiz service. When user do the quiz, system will create the quiz's database on firebase for keeping.

2. User B(who already do register)

- Users can access the quiz without doing the register. We already create the username and password (username : testsomoh@gmail.com , password: 123456) that keep in the user's database on firebase. Therefore, users can do the quiz. And when users do the quiz, the system will keep it on the database and send it to the quiz app.

3. Profile Image's Database

- If users take an image from camera, it will store image's path in the firebase storage to generate the URL and store in firebase database.
- If users select an image from user's gallery, it will store image's path in the firebase storage to generate the URL and store in firebase database.
- If users delete an image on a profile page, it will delete the URL's image that is stored in the firebase database and delete that image on the profile page automatically

4. Local Database

- The database of this application consists of two mains database which will interact with the users

4.1 Quiz's Database

- In this database, it keeps several quizzes that users require to do. One quiz consists of QuestionID,questionText, answer,option 1,2,3 and 4. When a user does the quiz, the system will show quiz questions and options returned from firebase. Also, the wrong and true choice will be displayed. After that, when the user finishes doing the quiz, it will show the total score that user got from that topic.

4.2 User's Database

- After the user has done registration, the system will keep information about the user. There are UserID, username, email, password, signing date, login date. Moreover, it also keeps a profile image that the user sets in the setting page.

4.2.1 Comment's Database

- The message given by users will be forwarded to the firebase database in real time when users type something and click the send button. Each comment will be stored separately based on users' id.

4.2.2.1 History's Database

- When users have done the quiz, the latest quiz score will be forward and stored in the firebase database. Also, this database will return the latest quiz score of each topic to display in an application. firebase. Also, the wrong and true choice will be displayed. After that, when the user finishes doing the quiz, it will show the total score that user got from that topic.

Steps of using PENQUIZ application

The screenshot shows the 'Register' screen of the PENQUIZ application. At the top, there is a status bar with 'Emergency calls only', signal strength, 100% battery, and the time 5:29 PM. Below the status bar, the word 'Register' is displayed in a large, bold, black font. There are three input fields: 'Email', 'Password', and 'Re-enter password'. Each field has a red exclamation mark icon to its right, indicating a required field. A black tooltip with the text 'Required.' is visible next to the 'Re-enter password' field. At the bottom, there are two buttons: 'CANCEL' (white with a blue border) and 'CONFIRM' (solid blue).

1. Register

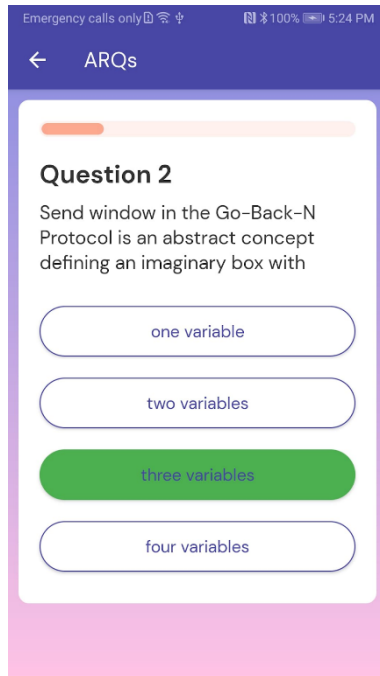
The screenshot shows the 'Log in' screen of the PENQUIZ application. At the top, there is a status bar with 'Emergency calls only', signal strength, 100% battery, and the time 5:32 PM. Below the status bar, the word 'PENQUIZ' is displayed in a large, stylized, black font. There are two input fields: 'Email' and 'Password'. Each field has a red exclamation mark icon to its right, indicating a required field. A black tooltip with the text 'Required.' is visible next to the 'Password' field. At the bottom, there are two buttons: 'CREATE ACCOUNT' (white with a blue border) and 'SIGN IN' (solid blue).

2. Login

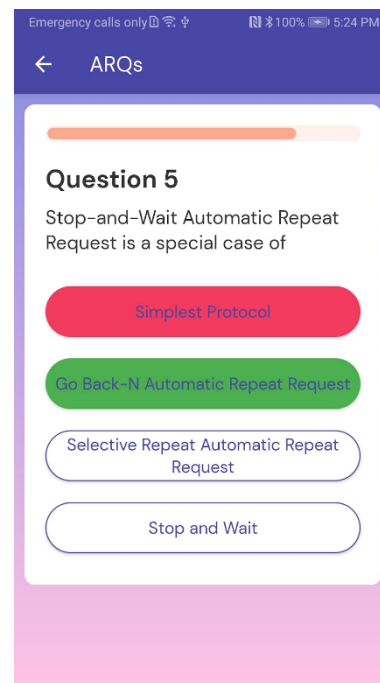
The screenshot shows the 'Home' screen of the PENQUIZ application. At the top, there is a status bar with 'Emergency calls only', signal strength, 100% battery, and the time 5:56 PM. Below the status bar, there is a navigation bar with a hamburger menu icon, the word 'Home', and a three-dot menu icon. Below the navigation bar, there is a search bar with a magnifying glass icon and the word 'Search'. Below the search bar, there are two quiz cards. The first card is titled 'What is TCP/IP?' and 'TCP/IP and OSI Layer' with 'Number of Question : 9'. The second card is titled 'Routing Protocol' with 'Number of Question : 7'. At the bottom, there is a partially visible card titled 'Selective Repeat ARQ'.

3. Search and Select Quiz

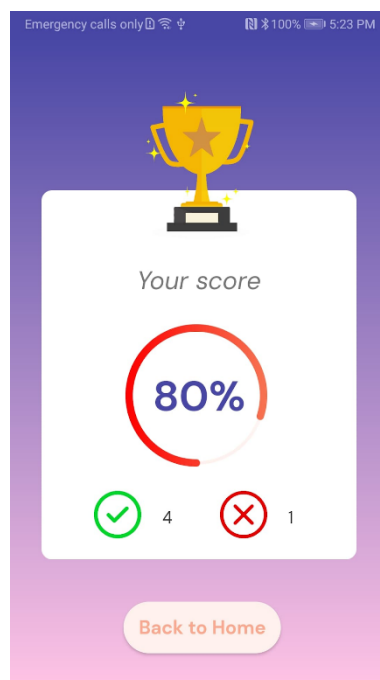
4. Do the quiz



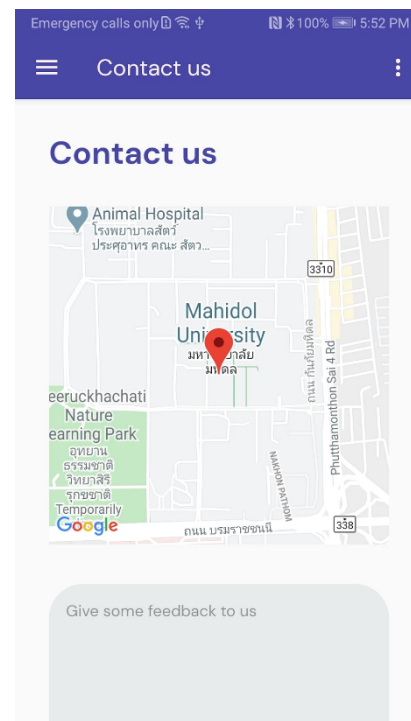
4.1 If users select wrong choice.



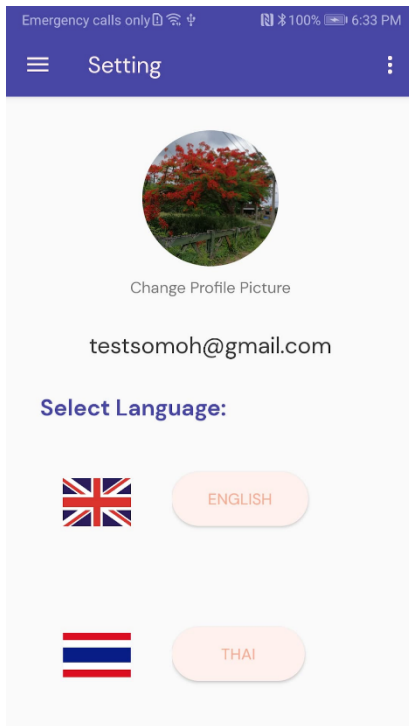
4.2 If users select wrong choice.



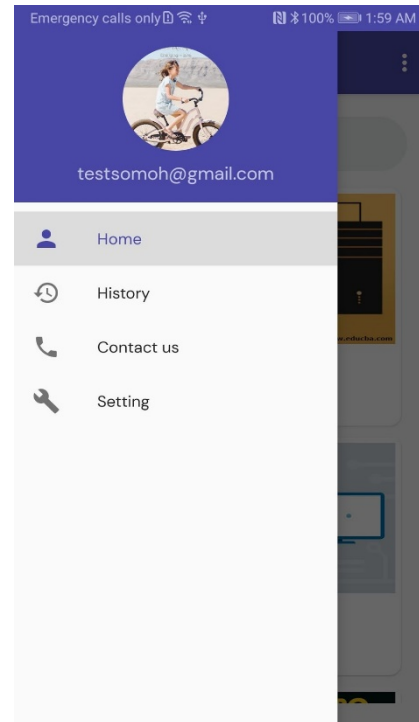
5. Score that user receive (Result)



6. Contact us

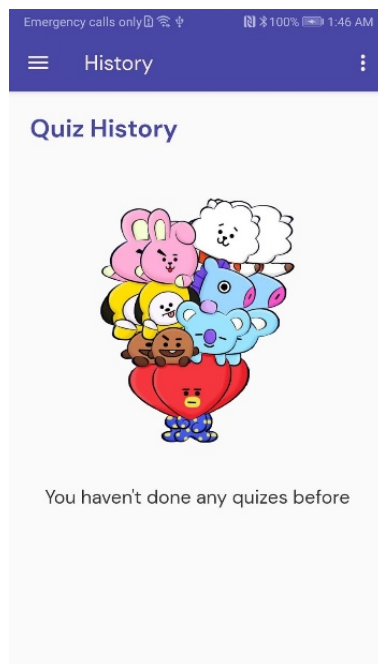


5. Score that user receive (Result)

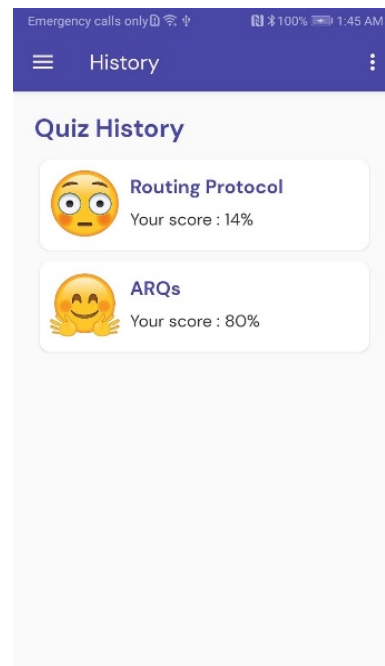


6. Navigation bar

7. Quiz History



7.1 Don't have quiz before



7.2 Quiz thar user used to do