TUNKU ABDUL RAHMAN UNIVERSITY COLLEGE

BEYOND EDUCATION

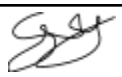
FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY

Year 2

AAC3353 Mobile Application Development

2020/2021

Programme: DST2 **Tutorial Group:** G2

Student Names	Student ID	Signature	Marks(%)
Eric Ng Boon Lee	1905539		
Eow Song Kai	1902858		
Wong Sai Seng	1903211		
Wong Ken Yee	1905593		
Teh Jin Yang	1902956		
	Total:		

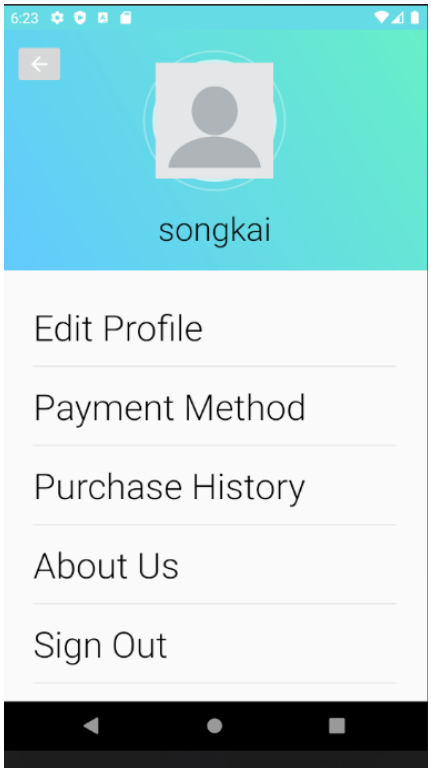
Date of Submission: 16/09/2020

Date Received by Tutor: _____

Comments:

Wong Sai Seng

Profile module

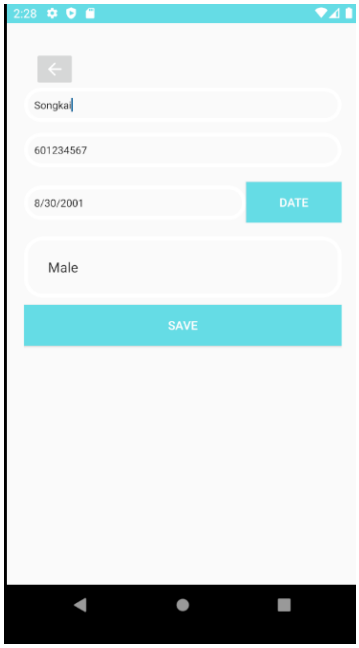



```
public void goToProfileDetail(View arg0) {
    Intent it1 = new Intent(getApplicationContext(), editProfileView.class);
    startActivity(it1);
}

public void goToPurchaseHistory(View arg0) {
    Intent it2 = new Intent(getApplicationContext(), purchaseHistoryView.class);
    startActivity(it2);
}

public void goToAboutUs(View arg0) {
    Intent it3 = new Intent(getApplicationContext(), aboutUsView.class);
    startActivity(it3);
}

public void paymentmethod(View view)
{
    Intent payment = new Intent( packageContext: this, PaymentMethod.class);
    startActivity(payment);
}

public void logOut(View arg0) {

    SharedPreferences preferences = getSharedPreferences( name: "checkbox", MODE_PRIVATE);
    SharedPreferences.Editor editor = preferences.edit();
    editor.putString( s: "stayLogged", s1: "false");
    editor.apply();

    fBaseAuth.signOut();
    Intent intent = new Intent(getApplicationContext(), MainActivity.class);
    startActivity(intent);
    finish();
```

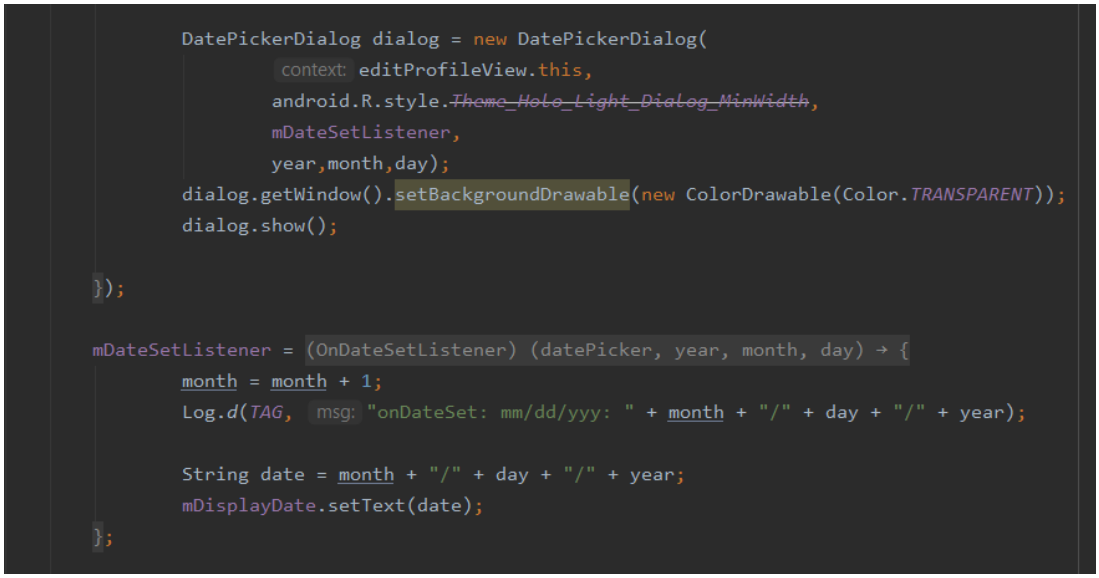
```
private void userInfoDisplay() {
    DatabaseReference UsersRef = FirebaseDatabase.getInstance().getReference().child("Users");

    UsersRef.child(currentID).addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
            if(dataSnapshot.exists())
            {

                userName = dataSnapshot.child("username").getValue(String.class);
                Log.d( tag: "hello", userName);
                nameText.setText(userName);
            }
        }

        @Override
        public void onCancelled(@NonNull DatabaseError databaseError) {

        }
    });

}
```

So for the profile page module, most of all the linear layout for edit profile, payment method and so on are using the onclick function to access through those modules. Besides that, the customer name below the picture I got uses the data management to store at firebase. If the customer wants to change the name then they can change it at the edit profile module and after then the name in the profile module will be changed also.

Edit Profile module



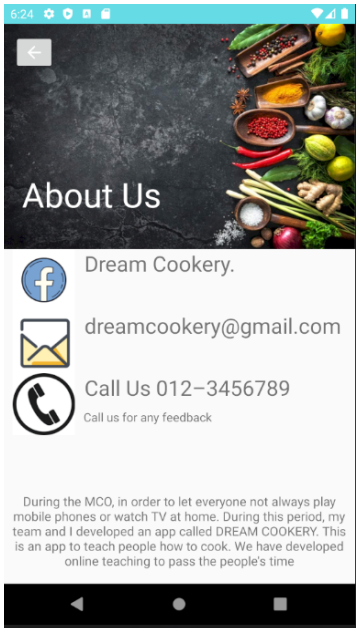

```
    Save.setOnClickListener(new View.OnClickListener(){
        @Override
        public void onClick(View v){
            eName = editName.getText().toString();
            ePhone = Integer.parseInt(editPhone.getText().toString());
            eBirthday = mDisplayDate.getText().toString();
            eGender = spin.getSelectedItem().toString();
            SharedPreferences prefs = PreferenceManager.getDefaultSharedPreferences( context: editProfileView.this);
            SharedPreferences.Editor editor = prefs.edit();
            editor.putString( s: "eName",eName);
            editor.putInt( s: "ePhone",ePhone);
            editor.putString( s: "eBirthday",eBirthday);
            editor.putString( s: "eGender",eGender);
            editor.apply();


            //firebase functions
            FirebaseDatabase fDatabase = FirebaseDatabase.getInstance();
            DatabaseReference fReference;
            fReference = fDatabase.getReference( path: "Users").child(currentID);
            fReference.child("username").setValue(eName);
            fReference.child("phoneNumber").setValue(ePhone);
            fReference.child("gender").setValue(eGender);
            fReference.child("birthday").setValue(eBirthday);

        }
    });
}
```



```
        DatePickerDialog dialog = new DatePickerDialog(
                context: editProfileView.this,
                android.R.style.Theme_Holo_Light_Dialog_MinWidth,
                mDateSetListener,
                year,month,day);
        dialog.getWindow().setBackgroundDrawable(new ColorDrawable(Color.TRANSPARENT));
        dialog.show();

});

mDateSetListener = (OnDateSetListener) (datePicker, year, month, day) -> {
        month = month + 1;
        Log.d(TAG,  msg: "onDateSet: mm/dd/yyy: " + month + "/" + day + "/" + year);

        String date = month + "/" + day + "/" + year;
        mDisplayDate.setText(date);
};
```

So, the data that the key in by the users will store into the firebase and I also use the shared preferences to make sure the data that the user key in didn't get lost. Above is the code that i write to store the user details. Besides that, I also use the spinner to let the users choose their gender. After that, date picker also consists of this application. Users can use the date picker to display their birthday in this application.

About Us module

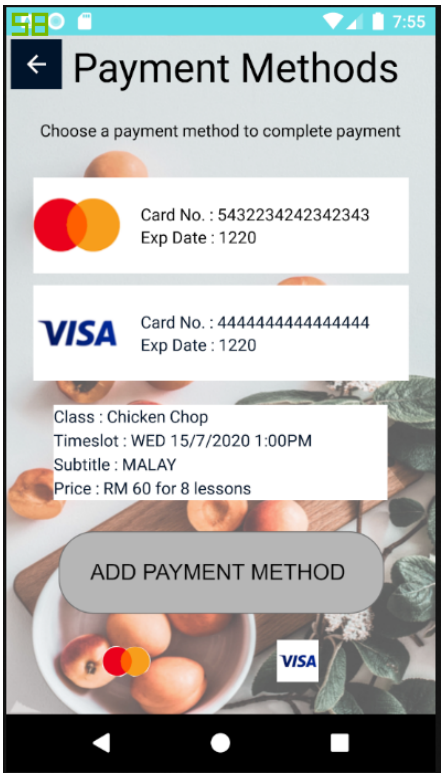



```
ImageView img = (ImageView) findViewById(R.id.facebook);
img.setOnClickListener((v) → {
        Intent intent = new Intent();
        intent.setAction(Intent.ACTION_VIEW);
        intent.addCategory(Intent.CATEGORY_BROWSABLE);
        intent.setData(Uri.parse("https://www.facebook.com/"));
        startActivity(intent);
});
```

Besides that, for each of the icons that I put inside this application, the user can click on it to go to the facebook, email or contact page to give us feedback or some comment. So, above is the code that I do in java that can let users click on it. After that, below the about us page got some description for our application and why we will do this application public.

Wong Ken Yee

Payment Method Module



The function of Payment method is to let users choose their payment method which is using a credit card to make payment for their purchases. The function could save the users' credit card information in the Firebase database and display the users credit card information on this screen such as credit card number and expiry date. The information about the purchase details such as class name, timeslot, subtitle and price will be displayed on this screen as well. Following is the screenshot of the code of the function. DataSnapShot is used to retrieve the credit card information and class purchase order information from the Firebase real time database and then using setText to display them on the specific location. SetVisibility is used to make adjustments on the visibility of the credit card or purchase order information. The information will be invisible if there is no credit card or purchase order store in the Firebase real time database.

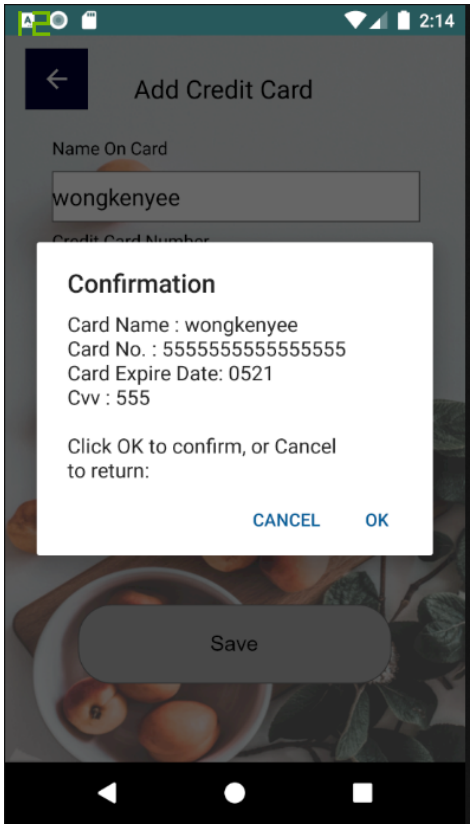
```
protected void onStart() {
    super.onStart();
    databaseReference.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
            total = dataSnapshot.child("total").getValue(int.class);
            total++;
            String mcNum = dataSnapshot.child("CC_Info").child("mastercard").child("ccNum").getValue(String.class);
            String mcDate = dataSnapshot.child("CC_Info").child("mastercard").child("ExpDate").getValue(String.class);
            String vcNum = dataSnapshot.child("CC_Info").child("visa").child("ccNum").getValue(String.class);
            String vcDate = dataSnapshot.child("CC_Info").child("visa").child("ExpDate").getValue(String.class);
            cClass = dataSnapshot.child("cart").child("classname").getValue(String.class);
            cSub = dataSnapshot.child("cart").child("subtitle").getValue(String.class);
            cTime = dataSnapshot.child("cart").child("timeslot").getValue(String.class);
            cPri = dataSnapshot.child("cart").child("price").getValue(String.class);
            if(cClass!=null)
            {
                list3.setVisibility(View.VISIBLE);
                cClassName.setText("Class : " + cClass);
                cSubtitle.setText("Subtitle : "+ cSub);
                cPrice.setText("Price : " + cPri);
                cTimeSlot.setText("Timeslot : "+ cTime);
            }
            else
                list3.setVisibility(View.GONE);

            if(mcNum != null )
            {
                list1.setVisibility(View.VISIBLE);
                mccNum.setText("Card No. : " + mcNum);
                mccDate.setText("Exp Date : "+ mcDate);
            }
            else
                list1.setVisibility(View.GONE);

            if(vcNum != null )
            {
                list2.setVisibility(View.VISIBLE);
                vccNum.setText("Card No. : " + vcNum);
                vccDate.setText("Exp Date : "+ vcDate);
            }
            else
                list2.setVisibility(View.GONE);
```

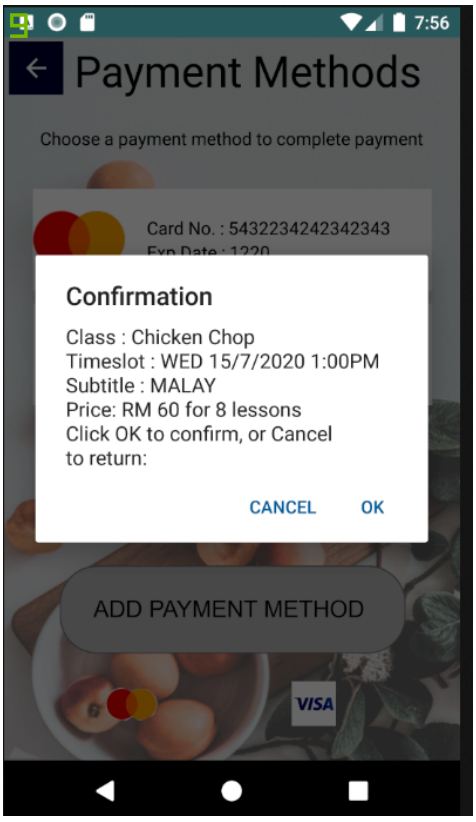
Add Credit Card Details

On this screen, there are few boxes to let users input their credit card information. There will be a hint on the box to let the user know what type information needed to input. After users input their information and press the save button, the system will validate the information. If wrong type of information had been key in, the function would toast message out.

Following is the code for the information validation. System will toast messages for the conditions such as no information input and the length of information is incorrect.

```
if (ccName == "EMPTY")
    Toast.makeText(AddVisa.this, "Please fill up your credit card name.", Toast.LENGTH_SHORT).show();
else if (ccNum.isEmpty() || cardnum.length() != 16 )
    Toast.makeText(AddVisa.this, "Please input the correct credit card number.", Toast.LENGTH_SHORT).show();
else if (ccExp.isEmpty() || ccExp.length() != 4 )
    Toast.makeText(AddVisa.this, "Please input the correct credit card expire date.", Toast.LENGTH_SHORT).show();
else if (ccCvv.isEmpty() || ccCvv.length() != 3 )
    Toast.makeText(AddVisa.this, "Please input the correct credit card cvv.", Toast.LENGTH_SHORT).show();
```



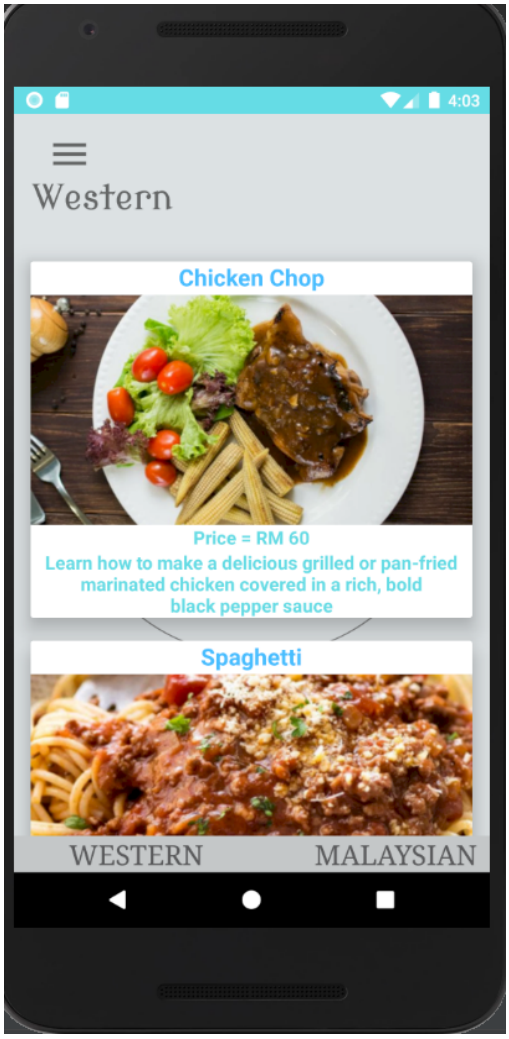


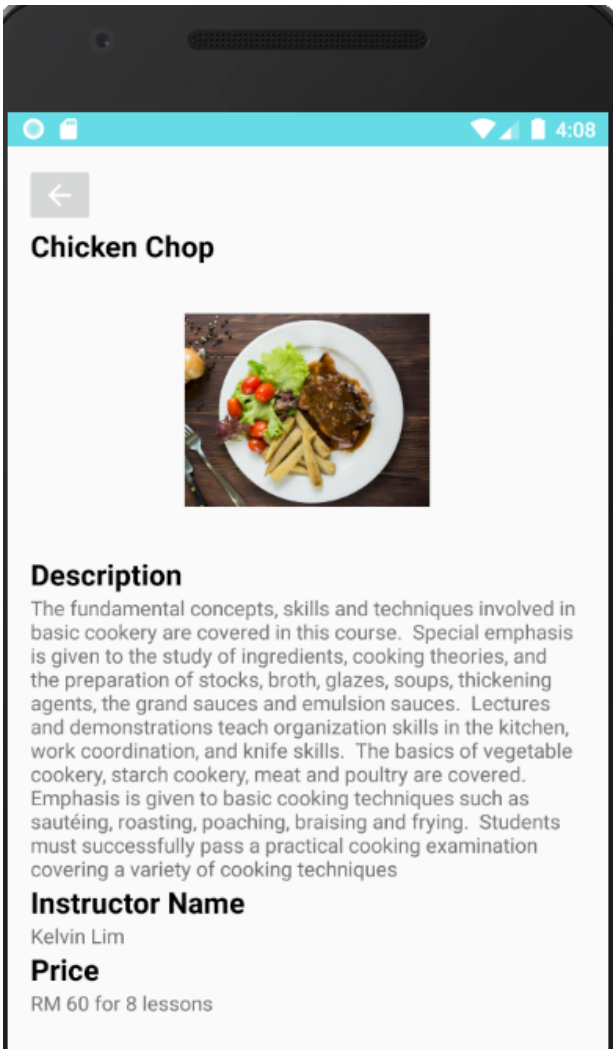
After the information has been validated, the system will prompt out the confirmation message to the user. Users could double check the information before confirming the process. The system will store the credit card information in the Firebase realtime database after clicking the ok button. This same method is implemented on the payment screen. Users will be prompted for confirmation messages before making their payment and store the purchase information in the purchase history folder in the Firebase. Following is the code for the above function.

```
        final AlertDialog.Builder myAlertBuilder = new AlertDialog.Builder(AddVisa.this);
        myAlertBuilder.setTitle("Confirmation");
        myAlertBuilder.setMessage("Card Name : " + ccName + "\n" +
                "Card No. : " + ccNum + "\n" +
                "Card Expire Date: " + ccExp + "\n" +
                "Cvv : " + ccCvv + "\n\n" +
                "Click OK to confirm, or Cancel to return:");
        myAlertBuilder.setPositiveButton("OK", (dialog, which) -> {
                        addCreditCard (ccName, ccNum, ccExp, ccCvv);
                    Intent successAddedCard = new Intent(getApplicationContext(), SuccessAddedCard.class);
                    startActivity(successAddedCard);
                });
        myAlertBuilder.setNegativeButton("Cancel", new DialogInterface.OnClickListener()
        {public void onClick(DialogInterface dialog, int which) { }
                });
        myAlertBuilder.show();
    }
}

private void addCreditCard ( final String ccName, final String ccNum, final String ccExp, final String ccCvv) {
    final FirebaseAuth fBaseAuth = FirebaseAuth.getInstance();
    String userID = fBaseAuth.getCurrentUser().getUid();
    char first = ccNum.charAt(0);
    final FirebaseDatabase database = FirebaseDatabase.getInstance();
    DatabaseReference myRef = database.getReference("Users").child(userID).child("CC_Info");
    if (first == '2' || first == '5') {
        myRef.child("mastercard").child("ccname").setValue(ccName);
        myRef.child("mastercard").child("ccNum").setValue(ccNum);
        myRef.child("mastercard").child("ExpDate").setValue(ccExp);
        myRef.child("mastercard").child("cvv").setValue(ccCvv);
    } else {
        myRef.child("visa").child("ccname").setValue(ccName);
        myRef.child("visa").child("ccNum").setValue(ccNum);
        myRef.child("visa").child("ExpDate").setValue(ccExp);
        myRef.child("visa").child("cvv").setValue(ccCvv);
    }
```

Teh Jin Yang

Class Info Overview module



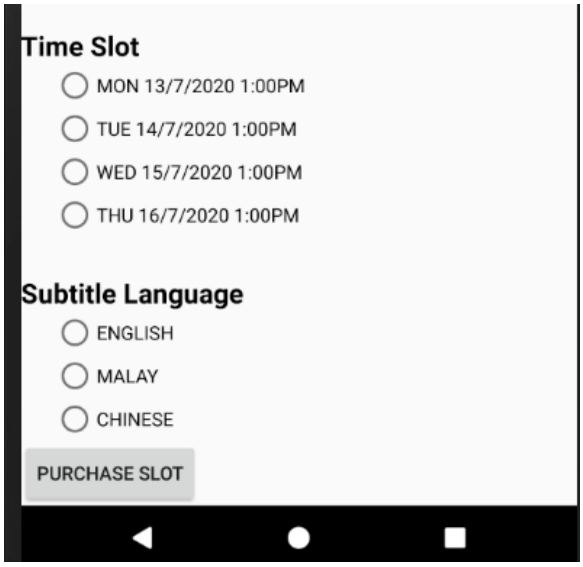



```
classID = getIntent().getStringExtra("cID");
category = getIntent().getStringExtra("Category");
```

This code is to get the class ID and the category of the class in the database. This “cID” and “Category” is the data passed by the Eric module(ClassViewFragment) For example, when the user click on the chicken chop, the cID will be the western1 which represent the chicken chop class and the category will be the western.

```
    getProductDetails(classID);
}


private void getProductDetails(String classID)
{
    DatabaseReference classRef = FirebaseDatabase.getInstance().getReference("Classes").child(category);

    classRef.child(classID).addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
            if(dataSnapshot.exists())
            {
                Classes classes = dataSnapshot.getValue(Classes.class);
                name.setText(classes.getcName());
                description.setText(classes.getcInfoDescription());
                Picasso.get().load(classes.getcImage()).into(image);
                insName.setText(classes.getcInsName());
                price.setText("RM " + classes.getcPrice() + " for 8 lessons");
            }
        }

        @Override
        public void onCancelled(@NonNull DatabaseError databaseError) {

        }
    });
}
```

After getting the “cID” and “category”, I declare a classRef to link to the database. DataSnapShot is used to retrieve the class data information. Then, it will set the value to the layout by the variable that already findviewbyID. So, in the figure, the Name, Description, Image,Instructor Name and Price are get from the firebase and set to the layout. If the datasnapshot does not exist, it will be cancelled.




```
public void onPurchaseSlot(View view){
    FirebaseAuth fBaseAuth = FirebaseAuth.getInstance();
    String userID = fBaseAuth.getCurrentUser().getUid();
    FirebaseDatabase firebaseDatabase = FirebaseDatabase.getInstance();
    DatabaseReference getCart = firebaseDatabase.getReference("Users").child(userID);
    String sub,time;
    Intent intent=new Intent(ClassInfoOverview.this, PaymentMethod.class);
    String pri=price.getText().toString();
    String t1=timeslot1.getText().toString();
    String t2=timeslot2.getText().toString();
    String t3=timeslot3.getText().toString();
    String t4=timeslot4.getText().toString();
    String s1=english.getText().toString();
    String s2=malay.getText().toString();
    String s3=chinese.getText().toString();
    String classNm=name.getText().toString();
    intent.putExtra("price",pri);
    intent.putExtra("className",classNm);
```

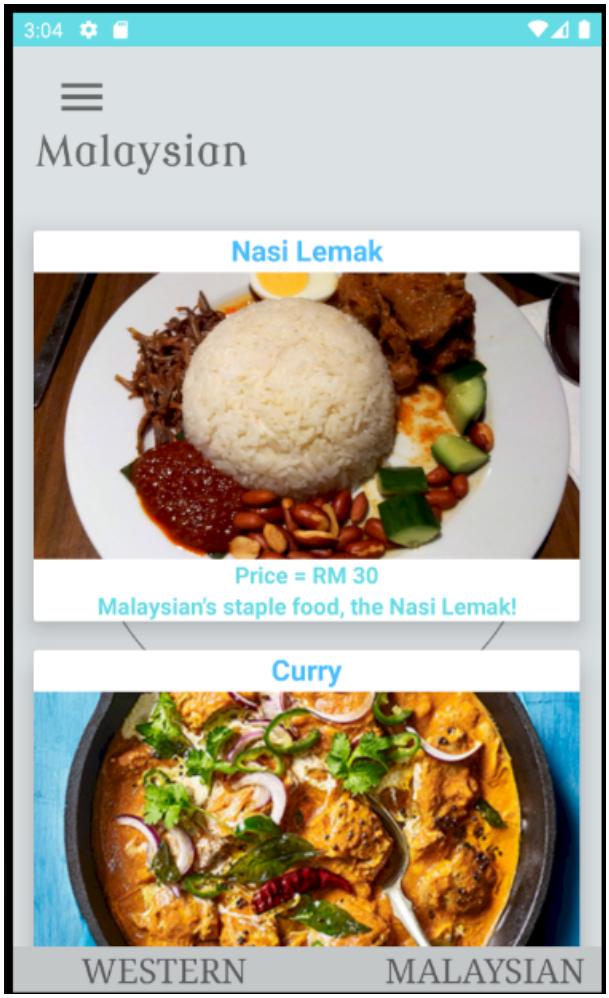
For the timeslot and the language, customers need to select the timeslot and subtitle language they want and click the purchase slot button.To get the data the customer selected I have declared some variables to get the data by using getText().toSting(). On the above figure, after I get the data from the price and name. I will send the data to the PaymentMethod class which is my teammate, Ken module by using the intent.putExtra() for him to do the payment details.

```
if(timeslot1.isChecked()) {
    intent.putExtra("timeSlot", t1);
    getCart.child("cart").child("timeslot").setValue(t1);
}
else if(timeslot2.isChecked()) {
    intent.putExtra("timeSlot", t2);
    getCart.child("cart").child("timeslot").setValue(t2);
}
else if(timeslot3.isChecked()) {
    intent.putExtra("timeSlot", t3);
    getCart.child("cart").child("timeslot").setValue(t3);
}
else {
    intent.putExtra("timeSlot", t4);
    getCart.child("cart").child("timeslot").setValue(t4);
}

if(english.isChecked()){
    intent.putExtra("subtitle",s1);
    getCart.child("cart").child("subtitle").setValue(s1);
}
else if(malay.isChecked()){
    intent.putExtra("subtitle",s2);
    getCart.child("cart").child("subtitle").setValue(s2);
}
else{
    intent.putExtra("subtitle",s3);
    getCart.child("cart").child("subtitle").setValue(s3);
}

getCart.child("cart").child("classname").setValue(classNm);
getCart.child("cart").child("price").setValue(pri);
```

For the radio button of time slot and subtitle language, I use the isChecked() function to check whether the customer selects which time slot and which subtitle language. Then, same with the above explanation, I pass the data by using intent.putExtra() for the payment details. Besides, I will also insert the data into the firebase(realtime database) of the “cart” which is going to be the data for purchase history.

Eric Ng Boon Lee

Main Class View



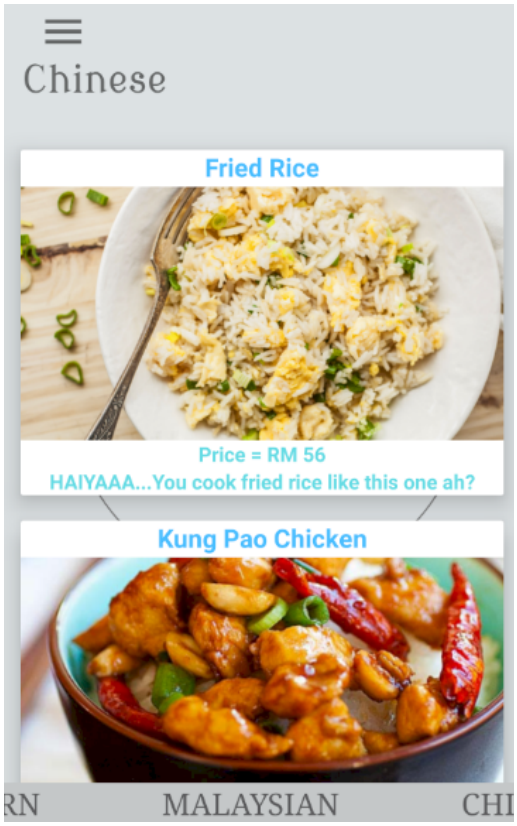
```
<FrameLayout
    android:id="@+id/flFragment"
    android:name="com.example.dream_cookery.MenuFragment"
    android:layout_width="0dp"
    android:layout_height="0dp"
    app:layout_constraintBottom_toTopOf="@+id/horizontalScrollView"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/menuIcon">

</FrameLayout>
```

The main class view layout holds a frame layout which handles the display of different fragments.

```
public void openWesternClassViewFragment() {

    ClassViewFragment newFragment = new ClassViewFragment();
    fragmentTransaction = fragmentManager.beginTransaction();
    fragmentTransaction.replace(R.id.flFragment, newFragment).addToBackStack(null);
    fragmentTransaction.commit();

    bundle = new Bundle();
    bundle.putString("mainText", "Western");

    newFragment.setArguments(bundle);
    isClassViewFragmentDisplayed = true;
}

public void openMalaysianClassViewFragment()
{
    ClassViewFragment newFragment = new ClassViewFragment();
    fragmentTransaction = fragmentManager.beginTransaction();
    fragmentTransaction.replace(R.id.flFragment, newFragment).addToBackStack(null);

    fragmentTransaction.commit();

    bundle = new Bundle();
    bundle.putString("mainText", "Malaysian");

    newFragment.setArguments(bundle);
    isClassViewFragmentDisplayed = true;
}
```

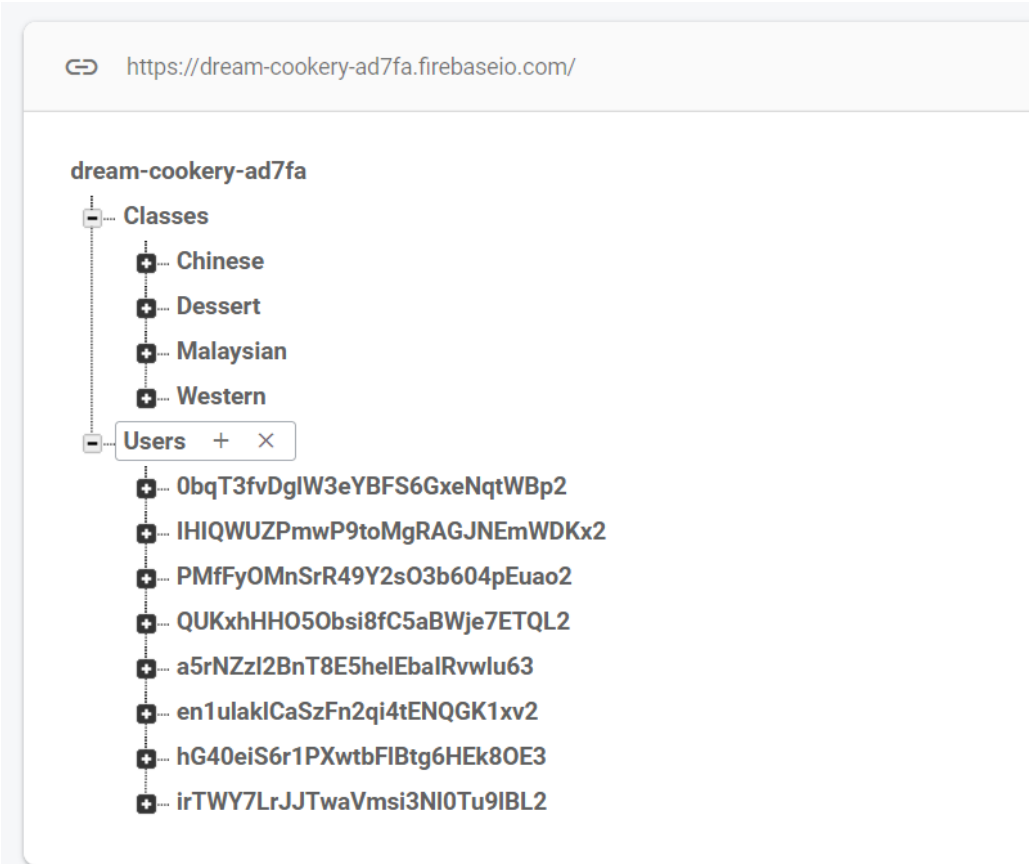
The way the different categories of classes is displayed is through replacing the fragment that is hosted in the frame layout. A bundle is created and passed into the Class View Fragment to change the main text to the category names, and to also change the contents of the recycler view.

Class View Fragment



```java
@Override
public void onStart() {
    super.onStart();

    if(getArguments().getString( key: "mainText")== "Western")
    {
        mainTextString = getArguments().getString( key: "mainText");
        ClassesRef = FirebaseDatabase.getInstance().getReference( path: "Classes").child(mainTextString);
        Log.d( tag: "Hello",  msg: "Western");
        mainText.setText(mainTextString);
    }
    else if(getArguments().get("mainText") == "Malaysian")
    {
        mainTextString = getArguments().getString( key: "mainText");
        ClassesRef = FirebaseDatabase.getInstance().getReference( path: "Classes").child(mainTextString);
        Log.d( tag: "Hello",  msg: "Malaysian");
        mainText.setText(mainTextString);
    }
    else if(getArguments().get("mainText") == "Chinese")
    {
        mainTextString = getArguments().getString( key: "mainText");
        ClassesRef = FirebaseDatabase.getInstance().getReference( path: "Classes").child(mainTextString);
        Log.d( tag: "Hello",  msg: "Chinese");
        mainText.setText(mainTextString);
    }
    else if(getArguments().get("mainText") == "Dessert")
    {
        mainTextString = getArguments().getString( key: "mainText");
        ClassesRef = FirebaseDatabase.getInstance().getReference( path: "Classes").child(mainTextString);
        Log.d( tag: "Hello",  msg: "Dessert");
        mainText.setText(mainTextString);
```

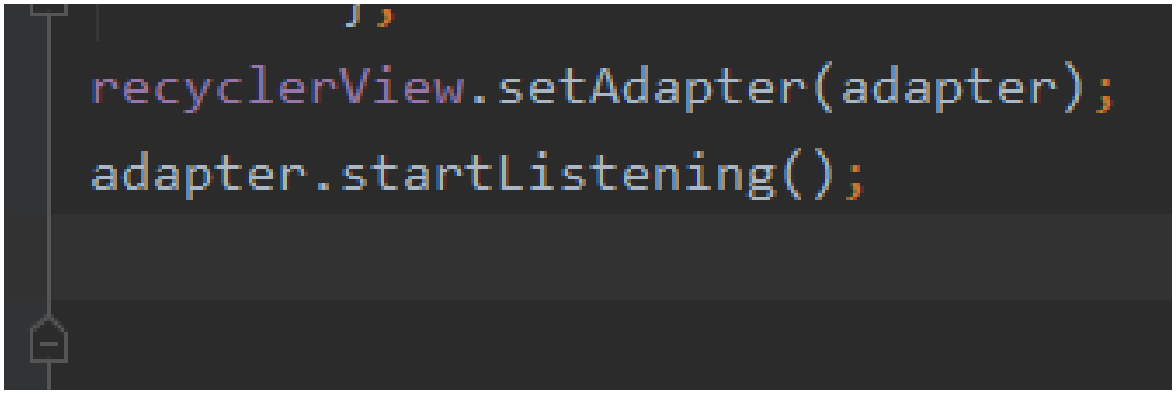
During the onStart lifecycle of the fragment, the fragment will first get the bundle string that is passed from Main Class View. An if statement decides the category of the cooking classes. Example, if it is Chinese, then the heading will change to Chinese and the database instance will look into the tree where the child is also “Chinese”.



The classes are displayed by retrieving the child from the “Classes” branch.

```
FirebaseRecyclerOptions<Classes> options =
        new FirebaseRecyclerOptions.Builder<Classes>()
                .setQuery(ClassesRef, Classes.class)
                .build();

FirebaseRecyclerAdapter<Classes, ClassViewHolder> adapter =
        new FirebaseRecyclerAdapter<~>(options) {
            @Override
            protected void onBindViewHolder(@NonNull ClassViewHolder classViewHolder, int i, @NonNull final Classes classes)
            {
                classViewHolder.txtClassName.setText(classes.getcName());
                classViewHolder.txtClassDescription.setText(classes.getcDescription());
                classViewHolder.txtClassPrice.setText("Price = RM " + classes.getcPrice() );
                Picasso.get().load(classes.getcImage()).into(classViewHolder.imageView);

                classViewHolder.itemView.setOnClickListener((view) -> {
                        Intent intent = new Intent(getActivity().getApplicationContext(), ClassInfoOverview.class);
                        intent.putExtra( name: "cID", classes.getcID());
                        intent.putExtra( name: "Category", mainTextString);
                        startActivity(intent);
                    });
            }

            @NonNull
            @Override
            public ClassViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
                View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.classes_layout, parent,  attachToRoot: false);
                ClassViewHolder holder = new ClassViewHolder(view);
                return holder;
            }
```

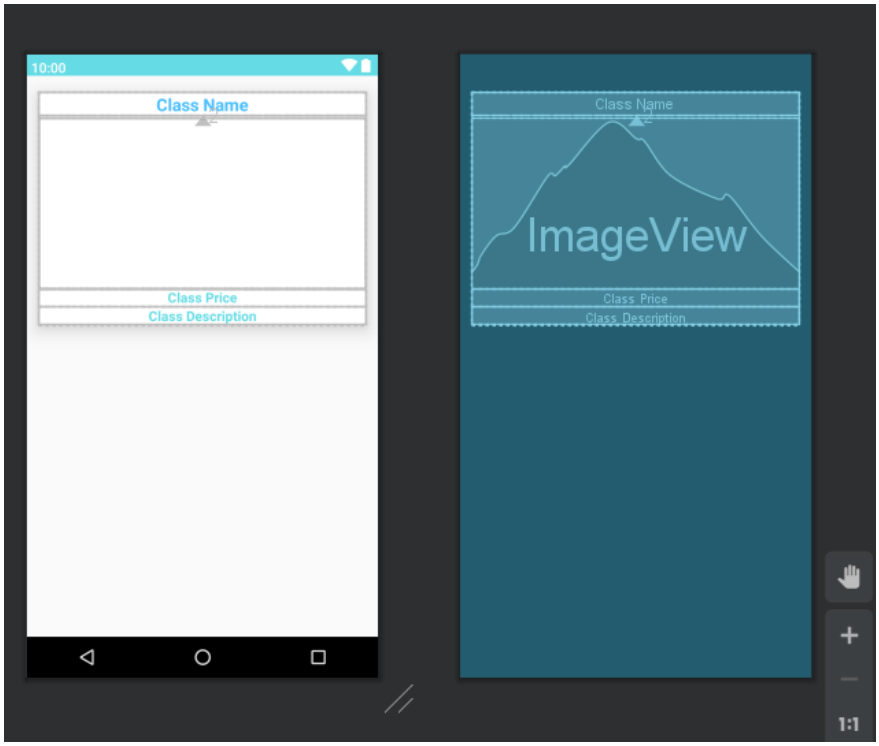
After the database instance is decided, a FirebaseRecylerOptions is created to create queries. Then the FirebaseRecylerAdapter takes the Models, and the View Holder to help create the Recyclerview. Later an intent is sent to the Class Info Overview to set the details of the cooking class that is clicked.



```
recyclerView.setAdapter(adapter);
adapter.startListening();
```

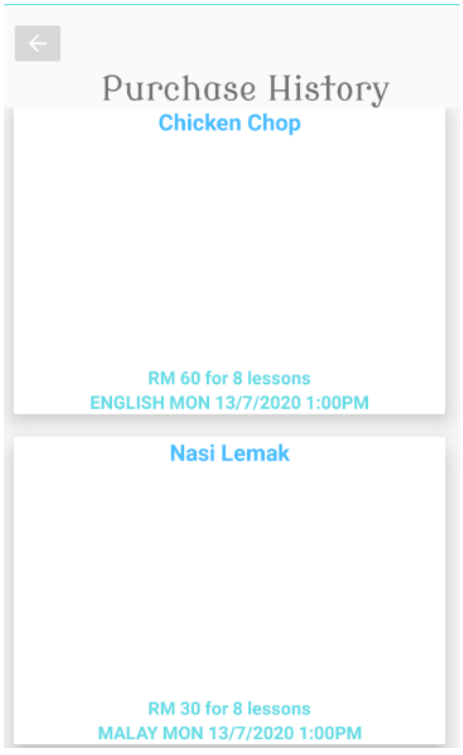
After that, the recycler view adapter is set, and the classes are displayed.

```
public class ClassViewHolder extends RecyclerView.ViewHolder implements View.OnClickListener
{
    public TextView txtClassName, txtClassDescription, txtClassPrice;
    public ImageView imageView;
    public ItemClickListener listener;

    public ClassViewHolder(@NonNull View itemView) {
        super(itemView);

        imageView = itemView.findViewById(R.id.class_image);
        txtClassName = (TextView) itemView.findViewById(R.id.class_name);
        txtClassDescription = (TextView) itemView.findViewById(R.id.class_description);
        txtClassPrice = (TextView) itemView.findViewById(R.id.class_price);
    }

    public void setItemClickListener(ItemClickListener listener) { this.listener = listener; }

    @Override
    public void onClick(View view) {
        listener.onClick(view, getAdapterPosition(), isLongClick: false);

    }
}
```



View Holder layout.

Purchase History/ Class Schedule



The Purchase History is very similar to the Class View Fragment which uses the Firebase Recycler Adapter. Due to time constraints, the View Holder for the Purchase History uses the same one from the Class View Fragment.

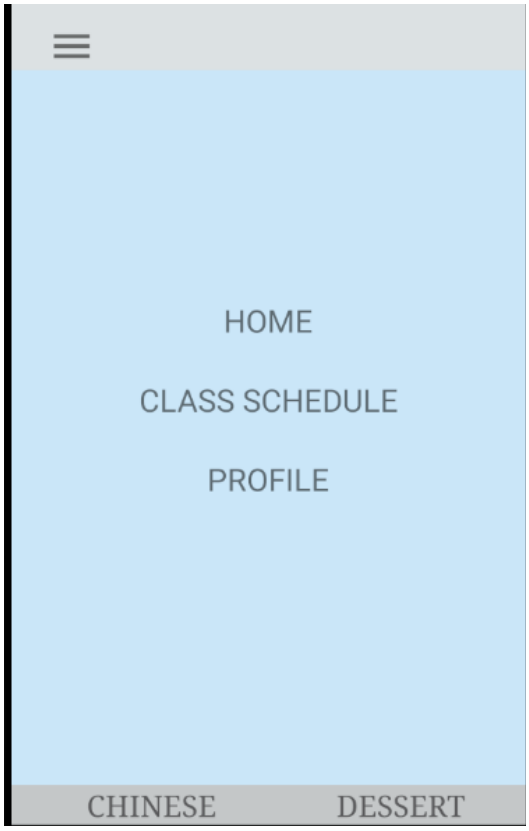
```
verride
otected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.purchase_history);

  ImageButton backPress = findViewById(R.id.backPurchaseHistory);
  backPress.setOnClickListener((view) → {
        Intent intent = new Intent( packageContext: purchaseHistoryView.this, profilePageView.class);
        startActivity(intent);
  });


  recyclerView = findViewById(R.id.PurchaseHistory);
  recyclerView.setHasFixedSize(true);
  layoutManager = new LinearLayoutManager( context: this);
  recyclerView.setLayoutManager(layoutManager);

  firebaseAuth = FirebaseAuth.getInstance();
  currentID = firebaseAuth.getCurrentUser().getUid();

  HistoryRef = FirebaseDatabase.getInstance().getReference( path: "Users").child(currentID).child("Purchase");
```

```
otected void onStart() {
  super.onStart();
  FirebaseRecyclerOptions<History> options =
        new FirebaseRecyclerOptions.Builder<History>()
              .setQuery(HistoryRef, History.class)
              .build();
  Log.d( tag: "hello",  msg: "no");
  FirebaseRecyclerAdapter<History, ClassViewHolder> adapter =
        new FirebaseRecyclerAdapter<~>(options) {
            @Override
            protected void onBindViewHolder(@NonNull final ClassViewHolder classViewHolder, int i, @NonNull final History history)
            {
                classViewHolder.txtClassName.setText(history.getClassName());
                classViewHolder.txtClassDescription.setText(history.getSubtitle() + " " + history.getTimeSlot());
                classViewHolder.txtClassPrice.setText(history.getPrice());

            }

            @NonNull
            @Override
            public ClassViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
                View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.classes_layout, parent,  attachToRoot: false);
                ClassViewHolder holder = new ClassViewHolder(view);
                return holder;
            }
        };
  recyclerView.setAdapter(adapter);
  adapter.startListening();
```

Like the Class View Fragment, the onStart lifecycle will display the Recycler View will be displayed by first setting the queries through FirebaseRecyclerOptions using the History Models to retrieve the data from the users. The Purchase History is stored in the Users branch and it is UserID specific which means that different users have different purchase history.


Users
0bqT3fvDglW3eYBFS6GxeNqtWBp2
CC_Info
Purchase
P1
ClassName: "Chicken Chop"
Price: "RM 60 for 8 lessons"
Subtitle: "ENGLISH"
TimeSlot: "MON 13/7/2020 1:00PM"
P3
P4
P5
birthday: "9/14/2020"
cart
email: "dame@gmail.com"
gender: "Male"
phoneNumber: 2013212314
total: 5
userID: "0bqT3fvDglW3eYBFS6GxeNqtWBp2"
username: "damenanoyo"
lHIQWUZPmwP9toMgRAGJNEmWDKx2
PMfFyOMnSrR49Y2sO3b604pEuao2

Menu Fragment



Like the Class View Fragment, the Menu Fragment uses the same Frame Layout in the Main Class View Activity.

```java
public void openMenuFragment()
{
    MenuFragment menuFragment = new MenuFragment();
    FragmentManager fragmentManager = getSupportFragmentManager();
    FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();

    fragmentTransaction.add(R.id.flFragment, menuFragment).addToBackStack(null).commit();
    isMenuFragmentDisplayed = true;
}

public void closeMenuFragment()
{
    FragmentManager fragmentManager = getSupportFragmentManager();
    MenuFragment menuFragment = (MenuFragment) fragmentManager.findFragmentById(R.id.flFragment);
    if(menuFragment != null)
    {
        FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();
        fragmentTransaction.remove(menuFragment).commit();
    }
    isMenuFragmentDisplayed = false;

}

public void onSavedInstanceState(Bundle savedInstanceState)
{
    savedInstanceState.putBoolean(STATE_FRAGMENT, isMenuFragmentDisplayed);
    super.onSaveInstanceState(savedInstanceState);
}
```

```java
    menuIcon.setOnClickListener((view) -> {
            if(!isMenuFragmentDisplayed) {
                openMenuFragment();

            }

            else
                closeMenuFragment();
    });

    if(savedInstanceState != null)
    {
        isMenuFragmentDisplayed = savedInstanceState.getBoolean(STATE_FRAGMENT);
    }

}
```

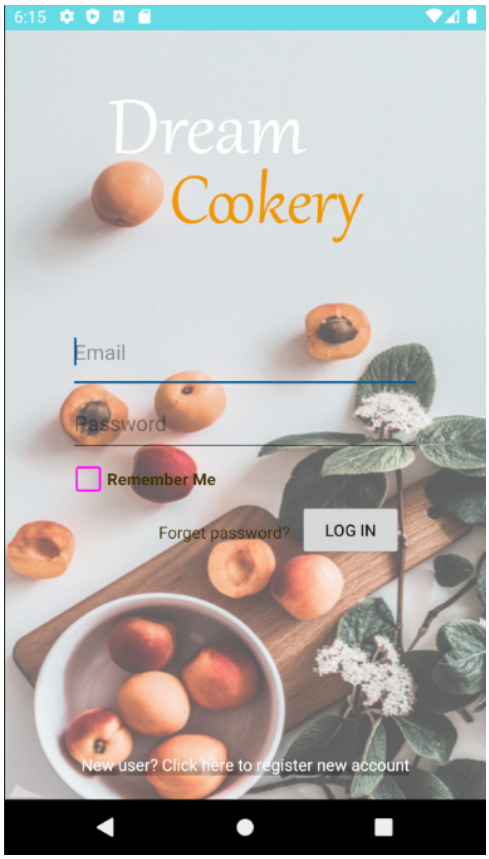
The Main Class View has 2 functions which handles the closing and opening of the Menu Fragment. The logic is displayed above. To prevent overlapping and multiple Menu Fragments, booleans are used to control the display of Fragments. There is also savedInstanceState to help the device to remember the state of the Fragment in different lifecycles.

```java
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
    final View rootView = inflater.inflate(R.layout.fragment_menu, container,  attachToRoot: false);

    classSchedule = rootView.findViewById(R.id.class_schedule);
    classSchedule.setOnClickListener((view) -> {
            Intent intent = new Intent(getActivity(), purchaseHistoryView.class);
            startActivity(intent);
    });

    homeText = rootView.findViewById(R.id.home);
    homeText.setOnClickListener((view) -> { goToHome(); });

    profileText = rootView.findViewById(R.id.profile);
    profileText.setOnClickListener((view) -> { goToProfile(); });


    return rootView;
}
```

The Menu Fragment has a total of 3 clickable text which brings the users to different activities, which is Home, Profile, and Class Schedule.

Eow Song Kai

Login



```
    StayLoggedIn();
    SetFAuthStateListener();

    loginButton.setOnClickListener((view) -> { LoginUser(); });

    registerLink.setOnClickListener((view) -> { openRegisterActivity(); });
    //remember me
    stayLogged.setOnCheckedChangeListener((compoundButton, b) -> {
            if(compoundButton.isChecked())
            {
                SharedPreferences preferences = getSharedPreferences( name: "checkbox", MODE_PRIVATE);
                SharedPreferences.Editor editor = preferences.edit();
                editor.putString( s: "stayLogged", s1: "true");
                editor.apply();
            }
            else if (!compoundButton.isChecked())
            {
                SharedPreferences preferences = getSharedPreferences( name: "checkbox", MODE_PRIVATE);
                SharedPreferences.Editor editor = preferences.edit();
                editor.putString( s: "stayLogged", s1: "false");
                editor.apply();
            }
    });

}

private void StayLoggedIn()
{
    // stay logged in
    SharedPreferences preferences = getSharedPreferences( name: "checkbox", MODE_PRIVATE);
    String checkbox = preferences.getString( s: "stayLogged",  s1: "");
    if (checkbox.equals("false"))
    {
        // Logout
        FirebaseAuth.getInstance().signOut();
        // Set stay logged checkbox
        SharedPreferences.Editor editor = preferences.edit();
        editor.putString( s: "stayLogged", s1: "true");
        editor.apply();
    }
```

There is a ‘remember me’ checkbox, if the user wants to stay logged in, shared preferences will store the user choice, when the next time the user comes back to the application, the application will stay the user login.

```
private void LoginUser()
{
    String email = inputEmail.getText().toString();
    String password = inputPassword.getText().toString();

    if(TextUtils.isEmpty(email))
    {
        Toast.makeText( context: this,  text: "Please write your email...", Toast.LENGTH_SHORT).show();
    }
    else if(TextUtils.isEmpty(password))
    {
        Toast.makeText( context: this,  text: "Please write your password...", Toast.LENGTH_SHORT).show();
    }
    else if (password.length() < 6)
    {
        Toast.makeText( context: this,  text: "Password has to be 6 characters or longer .", Toast.LENGTH_SHORT).show();
    }
    else
    {
        loadingBar.setTitle("Login Account");
        loadingBar.setMessage("Please wait, while we are checking the credentials.");
        loadingBar.setCanceledOnTouchOutside(false);
        loadingBar.show();

        AllowAccessAccount(email, password);
    }
```

```
private void AllowAccessAccount(final String email, final String password) {
    fBaseAuth.signInWithEmailAndPassword(email, password).addOnCompleteListener( activity: MainActivity.this, (task) -> {
            if(task.isSuccessful())
            {
                loadingBar.dismiss();
                openMainClassView();

            }
            else
            {
                Toast.makeText( context: MainActivity.this,  text: "Login Failed!!! ", Toast.LENGTH_SHORT).show();
                loadingBar.dismiss();
            }
    });
```

If the user enters the valid email and password it will bring the user to the main class view.

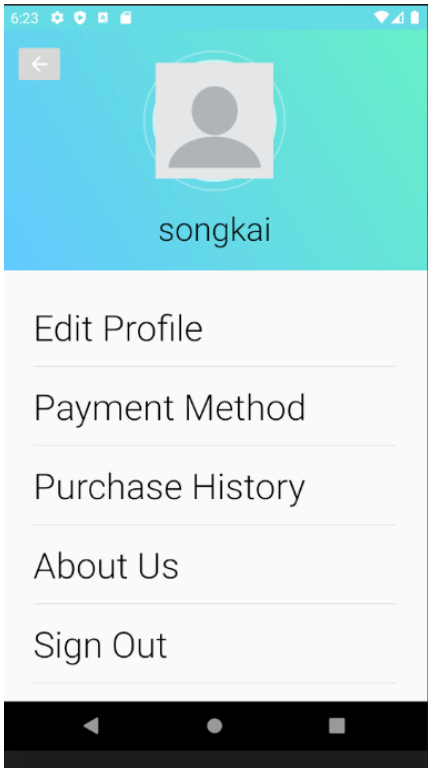
Register


6:17
Dream
Cookery
Email
New Username
Phone Number
New Password
CREATE ACCOUNT

```java
private void ValidateCredentials(final String email, final String username, final String phoneNum, final String password)
{
    final FirebaseAuth fBaseAuth = FirebaseAuth.getInstance();
    fBaseAuth.createUserWithEmailAndPassword(email, password).addOnCompleteListener( activity: RegisterActivity.this, (task) -> {
            if (task.isSuccessful())
            {
                Toast.makeText( context: RegisterActivity.this,  text: "Congratulations, your account has been created.", Toast.LENGTH_SHORT).show();
                loadingBar.dismiss();
                openMainActivity();
                FirebaseUser fUser = fBaseAuth.getCurrentUser();
                fUser.sendEmailVerification().addOnSuccessListener((OnSuccessListener) (aVoid) -> {
                        Toast.makeText( context: RegisterActivity.this,  text: "Verification Email has been sent.", Toast.LENGTH_SHORT).show();
                }).addOnFailureListener((e) -> {
                        Toast.makeText( context: RegisterActivity.this,  text: "Verification Email failed to be sent.", Toast.LENGTH_SHORT).show();
                });
                String userID;
                int total = 0;
                userID = fBaseAuth.getCurrentUser().getUid();
                FirebaseDatabase fDatabase = FirebaseDatabase.getInstance();
                DatabaseReference fReference;
                fReference = fDatabase.getReference( path: "Users").child(userID);
                fReference.child("userID").setValue(userID);
                fReference.child("username").setValue(username);
                fReference.child("email").setValue(email);
                fReference.child("phoneNumber").setValue(phoneNum);
                fReference.child("total").setValue(total);

                //shared preferences
                SharedPreferences preferences = getSharedPreferences( name: "checkbox", MODE_PRIVATE);
                SharedPreferences.Editor editor = preferences.edit();
                editor.putString( s: "stayLogged", s1: "false");
                editor.apply();
                finish();
            }
            else
            {
                loadingBar.dismiss();
                Toast.makeText( context: RegisterActivity.this,  text: "Error: Please try again...", Toast.LENGTH_SHORT).show();
            }
```

The firebase database authentication will store the user password, and the realtime database will store user email, phone number, username and etc that are entered by the user. If the task is successful the user will be brought to the login page, if the task is unsuccessful the user would be prompted to register again.

Sign Out




```
public void logOut(View arg0) {

    SharedPreferences preferences = getSharedPreferences( name: "checkbox", MODE_PRIVATE);
    SharedPreferences.Editor editor = preferences.edit();
    editor.putString( s: "stayLogged", s1: "false");
    editor.apply();

    fBaseAuth.signOut();
    Intent intent = new Intent(getApplicationContext(), MainActivity.class);
    startActivity(intent);
    finish();

}
```

When the user selects sign out, the ‘stayLogged’ will be false and go to the login page.