Assignment 4 – Implementation

Change Log:

a. Architecture: We intended to use the Client-Server architecture in the app but wasn’t able to incorporate it into the demo.
b. Version Control: Instead of using the SRCT GitLab as our version control, we decided to use GitHub.
c. Initially we did not plan to use any database to submit orders. This was changed as the project matured. We used the Firebase database to link to the various images that how the items on our menu as well providing a description for the items. We submitted new order requests to the firebase database.

Technology Used:

a. The technology that was used in the implementation of the Cake Ordering App was mainly Android Studio which uses the Java language. Using Android Studio limits the devices that can use the mobile app to only Android or Android emulating devices. We also incorporated the AgendaCalendarView using the Java language into our mobile app to create a calendar that the users can see when choosing a date for their cake orders. Along with that, we implemented the OpenGL ES api that was used to create the 3D modeling function that lets the users virtually build their own cake before they consult with their intended bakery along with Blender that let us export models used for the cakes. Some patterns that were used were the Facade pattern which pushed all the heavy lifting in the backend, only showing the user the front end and keeping them ignorant of all the computations happening behind the scenes. Another pattern being implemented is the Model-View-Controller pattern which is mostly being used in the 3D model.

   Android uses several different support libraries that define how the android OS reacts to the application such as what happens an activity is created, paused, resumed, or binded. Buttons, layout, and text are organized in xml layout that receive input from various other xml files. Items in the xml files that have an id defined for them can have an associated action with them their associated java class implementing the onClick method and associating a listener for that object.

   This project also used databases to grab remote information from Firebase by using the Firebase API. SQLLite was used to generate the database for order requests.

Descriptions of the Files:

In Android Studio, there are xml files that specify the layout of their correspond activity classes which are only java files. They usually have similar names to show which xml layout goes with which class. All files that aren't specified are files that are mainly code that has been reused.

a. CakeModelingActivity.java
    i. This file is the main activity class for the 3D modeling function. This class sets up the 3D modeling view and allows the users to customize what they want on their cake visually.
    ii. Developed by: Victoria

b. Activity_CakeModeling.xml
    i. This file sets up the layout for the CakeModelingActivity.java class. This makes sure that everything looks nice to the user when they are using it.
    ii. Developed by: Victoria

c. Torus.java
    i. This file turns the cake1.obj object into text that can be drawn to the screen when the CakeModelingActivity.java class calls for it to.
    ii. Developed by: Victoria

d. FormActivity.java
    i. This file implements the form used to order a cake instead of using the cake modeling function. It is mainly to help the cake ordering process go faster for users who already know what they want.
    ii. Developed by: Victoria

e. Activity_Form.xml
    i. This file constructs the layout for the FormActivity.java file and makes sure it looks appealing to the user.
    ii. Developed by: Victoria

f. ViewCalendar.java
    i. This is the main class for the calendar. This class sets up the calendar and displays a bakeries schedule
    ii. Developed by: Kamil

g. Activity_View_Calendar.xml
    i. This xml file sets up the initial layout of the calendar. This includes the colors for both the header and text of the calendar so it looks nice and pretty.

    ii. Developed by: Kamil

h. DrawableCalendarEvent.java
    i. This file deals with creating events that are to be displayed on the calendar. This class contains the information that is stored in the event layout.
    ii. Developed by: Kamil

i. View_Agenda_Drawable_Event.xml
    i. This file deals with creating the layout for created events. It either creates an event and displays it or displays a message stating no event
    ii. Developed by: Kamil

j. DrawableEventRenderer.java
    i. This file was originally intended to enhance the layout of created events. It had other fields present that we didn’t have the time to quite implement correctly. This includes an image of the product and a description
    ii. Developed by: Kamil

k. SelectBakeryActivity.java
    i. This is a simple class that allows customers to select which bakery they would like to place an order from.
    ii. Developed by Kevin

l. Cart.java
    i. This is the Model for the Cart it imports the cart_layout and a recycler view card window for each order object in the cart. Once the customer selects to submit order button a request database object is submitted to the Firebase database which would be the bakery’s database to keep track of orders.
    ii. Developed by Kevin

m. CartViewHolder.java
    i. This class gathers data about each individual cart item and computes the total for each order object in the cart.
    ii. Developed by Kevin

n. Category.java
    i. This class defines the Categories of items for the ConfectionList.
    ii. Developed by Kevin

o. Confection.java
   i. This class the attributes that are available for each confection item such as Description, Image, MenuId, Name, Price.
   ii. Developed by Kevin

p. ConfectionList.java
   i. This class pulls in the recycler view of confections from the content_home.xml that describe each confection. Every Image can be clicked on to go to the confection activity.
   ii. Developed by Kevin

q. ConfectionDetail.java
   i. This displays details about the individual confection item as well as its image on top of the screen. It provides a button to select up to 20 items of that particular confection. The Cart button adds the item and quantity to the order as an individual order item.

r. PastryOrder.java
   i. This class pulls in a recycler view from content_home.xml for each of the categories of non-cake items. A sample image is displayed for each category which can be clicked to enter the activity of the particular confection activity.

   ii. The customer can also click return to this page to view the contents of their cart and finalize their order.
   iii. Developed by Kevin

s. Order.java
   i. This class holds the ordering information for each individual order instance which was placed in the cart. The values of the fields ProductId, ProductName, Quantity, Price, will later be transferred to the request database which is then sent to Firebase.
   ii. Developed by Kevin
t. Request.java
   i. This class Holds the final data that will be placed into the BakeryOrder.db database object to submit to Firebase.
   ii. Developed by Kevin

u. MainActivity.java

i. This class is an activity that generates the home page for the App and displays a nice Cake Icon and a button to enter the rest of the App.
ii. Developed by Kevin and Victoria. 50%/ 50%.

v. EmailInfo.java
   i. This class pulls the recipient, subject, and body from activity_email_info, opens up the gmail app and adds them to the correct boxes in the email.
   ii. Developed by Chris using
w. First.xml,second.xml,third.xml
   i. Three xml files used for implementing the previousOrders slide

x. All other xml Files are basically just the layouts for their corresponding java classes There are other general xml files such as colors.xml and styles.xml which define the attributes for the layout.

Software Reuse:

a. Calendar
   i. AgendaCalendarView(https://github.com/Tibolte/AgendaCalendarView)
      1. This is where the main functionality of calendar came from. It came with a sample of the calendar in action that was a great reference on how the final product would be. It saved a lot of work of having to create our own calendar.
   ii. Kotlin-AgendaCalendarView(https://github.com/ognev-zair/Kotlin-AgendaCalendarView);
      1. This calendar is based on the above calendar but programmed in another language, Kotlin. The design of this calendar was much more appealing than the above calendar so design changes were made to the above calendar based on this calendar.
   iii. Caldroid Calendar(https://github.com/roomorama/Caldroid); Colors and other resources
      1. This calendar was a simple calendar layout which displayed booked dates. This calendar was unattractive but contained a wide variety of colors that were used in the final product. It saved time from having to test our own colors.
   iv. https://www.c-sharpcorner.com/blogs/send-to-email-app-in-android-application-using-android-studio
      1. This was used as a backbone for the
b. Tutorial Videos:

i. The following tutorials were used as code reuse sources which were very beneficial for our project. Much of the code reused was modified to make many great features for our project.

Android Studio Tutorial For Beginners - 1 Edureka:
https://www.youtube.com/watch?v=ZLNO2c7nqjw

Android Studio Tutorial For Beginners - 2 Edureka:
https://www.youtube.com/watch?v=D-iqMlLOrec

Android using Firebase Tutorial 2:
https://www.youtube.com/watch?v=dJm7LACOn80

Android using Firebase Tutorial 3:
https://www.youtube.com/watch?v=k1RUOexThGs

Android Studio Tutorial - Order Foods Part 4:
https://www.youtube.com/watch?v=T19qTLVDFV0&t=70s

Android Studio Tutorial - Order Foods Part 5:
https://www.youtube.com/watch?v=nlQTN7vkc0c