Module Code: CSCU9YH

Module Name: Mobile Application Development

Assignment Report

Student Number: 3058058

Table of contents

Structure

The main structure of the application for the Diary basically has three pages that the client will proceed using fragments. The initial page has date selection where the client can choose the date to which the client wants to insert their diary entry. After Selecting and confirming their chosen date, the client will be taken to the second page (fragment) where the selected date is displayed as well as a place to add the text for their diary entry. The client will have to option to clear the current text or add the entry into the database where the entry and the date will be stored. The client will be taken to the third page (fragment) where, after clicking the refresh button, can see all the entries that has been stored. The client will have the option to filter an entry by the date and this will make the delete filtered entry button active, so the client may delete the current filtered entry. There is also toolbar for each fragment, to allow the client to move in between the fragments.

System overview

Page 1 (Date Selection)

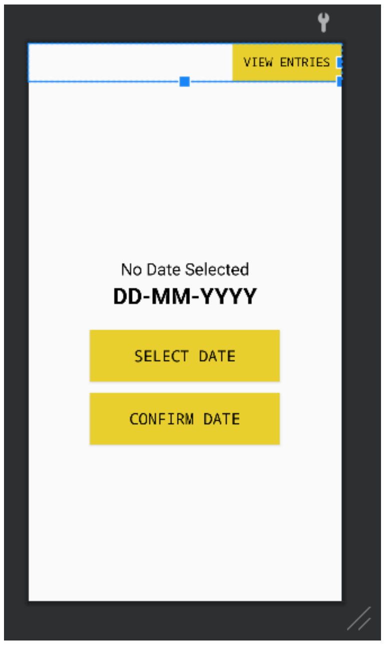
In the first page the client must select a date in order to proceed to the diary entry. The page has a few buttons, these include the "Confirm Date", "Select Date" and "View Entries" buttons. When the client clicks the "Select Date" button then the DatePickerDialog is shown alongside a calendar so that the client can choose a date. Once the date is chosen the "Confirm Date" button will be active and clickable allowing the client to proceed to the second page. There is also a "View Entries" button at the top right to allow the client to immediately proceed to the third page where all the diary entries are shown.

Page 2 (Diary entry)

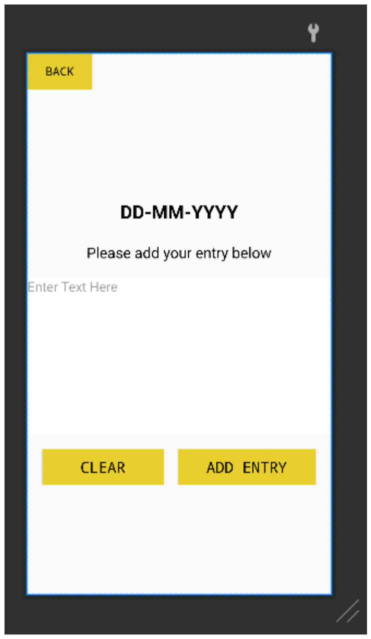
In the second page the selected date is shown, and the client can input text for their diary entry in the text field provided. There are three buttons here, the "Add entry", "Clear" and "Back" buttons. The "Add Entry" button will store the input text from the client into the database when clicked. The "Clear" button will allow the client to quickly clear/remove all the text quickly from the text field so that the client can easily redo their diary entry. The "Back" button is used to go back to the first page where the date selection is. Also, once the "Add Entry" button is clicked the client will be taken to the third page.

Page 3 (Diary entry logs)

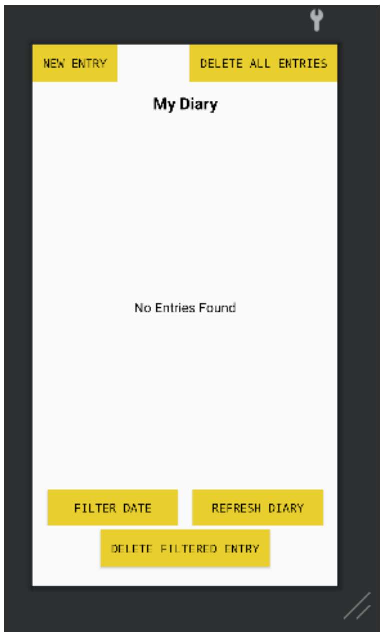
In the third page, there will be several buttons displayed. These buttons are the “New Entry”, “Delete All Entries”, “Refresh Diary”, “Filter Date” and “Delete Filtered Entry”. When the client clicks on the “Refresh Diary” the new diary entries made by the client will be shown along side the old ones. If the client is looking for a specific entry on a specific date, then they can use the “Filter Date” button which filter the entry of the specified date. If “Filter Date” is used, then the “Delete Filtered Entry” will be active and clickable to allow the client to delete the filtered entry. There is also the “Delete All Entries” button which will allow the client to delete all the diary entries in the database/storage. Finally, there is the “New Entry” button which will allow the client to go back to the first page to make a fresh new entry.



Page-1



Page-2



Page-3

Design of Structure

Key features

Date Picker

The Date Picker is in the first page of the application where the client can choose the date for their diary entry. Once the client has chosen their date, can then proceed to confirm the date with the ok button that is already implemented in the DatePickerDialog.

Add Entry

In the second page the client can input text for their diary entry for the selected date from the first page. The text field in the middle of the page is where the client can enter their text and once, they click the “Add Entry” button the data is then saved in the database.

View Entries

In the third page, the client can view all the diary entries (after clicking the “Refresh Diary” button) that they have made and see the dates of those entries. The diary entries will be shown at center of the page. However, if there are no diary entries in the database then a text saying “no entries found” will be shown in the middle of the page.

Delete Diary Entries

In the third page, at the top right corner there is a “Delete All Entries” button. When clicked this button will delete all the Diary entries ever made and stored by the client. If the client clicks the “Refresh Diary” button after that then a text saying “no entries found” will be shown in the middle of the page.

Filter Date

If the client uses the “Filter Date” button, then they can use the filter to find a diary entry made on a specific date as the client has the option to choose the date.

Delete Filtered Entry

For deleting the filtered entries, the client can click on the “Delete Filtered Entry” button at the bottom center of the page which will allow the client to delete the filtered entry. However, the “Filter Date” button has to be used to find the entry else the “Delete Filtered Entry” button will remain unclickable and unusable.

Database

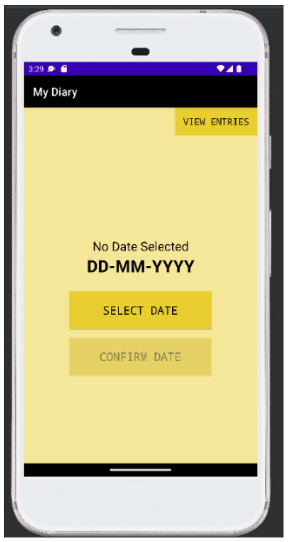
For the database, a RoomDataBase is used to store and get the diary entries from. The use of this allows the functionality to create and store the diary entry with the dates of each entry, the deleting of all the entries or specific ones in the database as well as checking if a date entry in the database exists and is all done in DairyDataQuery

UI Design

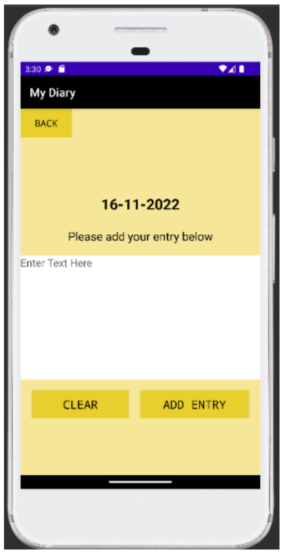
It is important that the experience of the client when using the application is good, this can be done by having a good, attractive, functional and simple UI design for the application. The simplicity of the application will allow the client to easily navigate the application with minimal difficulties. Therefore, the simple UI only has buttons which the client may need or may find useful.

Also, having the proper colors will make the application more attractive without sacrificing functionality or design, the simple colors used will allow the client to easily identify the different components of the application.

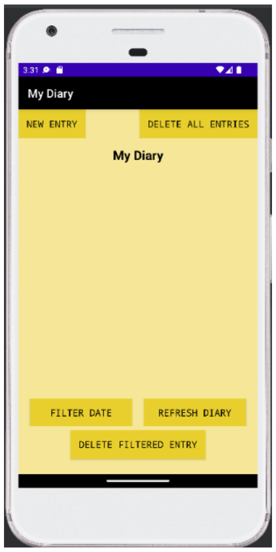
Most of the major elements of the application is around the middle of the screen so that the client will have an easier time understanding the application and can easily navigate through the application. Additional, navigation buttons have been placed at the top corners of the application, while the diary entry, the date selection and the diary text input is placed around the center, whereas additional functions are placed at the bottom center and/or corners.



Page-1



Page-2



Page-3

UI Design

Reflection

The main issue with the application would be the layout of the application as it seems a bit congested in some parts of the application. Another issue would be the color scheme as it also seems a bit murky which may make it difficult for individuals with eye sight issues. Also, the some of the buttons in different pages are right at the corners of the application as well as a few buttons are also not the appropriate size which in turn will sacrifice the overall design of the application. This could be fixed afterwards either with changing of text or font size or just change the overall size of the button.

Code

XML

1. activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#DDDDCB"
        android:orientation="vertical">

        <!--Toolbar at top for fragments-->
        <androidx.appcompat.widget.Toolbar
            android:id="@+id/toolsBar"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:background="?attr/colorPrimary"
            android:elevation="6dp"
            app:layout_constraintBottom_toTopOf="@+id/tab_layout"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent" />

        <!--View pager Used to display the background for every fragment-->
        <androidx.viewpager2.widget.ViewPager2
            android:id="@+id/pagerView"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_weight="1"
            android:background="#F6E798"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintStart_toStartOf="@+id/tab_layout"
            app:layout_constraintTop_toBottomOf="@+id/tab_layout"  />

    </LinearLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

2. diarydate.xml

```
3. <?xml version="1.0" encoding="utf-8"?>
   <LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
```

```xml
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    android:orientation="vertical"
    tools:context=".DiaryDatePage1">


    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="right"
        android:orientation="horizontal">

        <!--View Entries button goes to DiaryLogPage3-->
        <Button
            android:id="@+id/viewEntryButton"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:background="#E8CF2D"
            android:fontFamily="monospace"
            android:padding="14dp"
            android:text="View Entries"
            android:textColor="@color/black"
            android:textSize="16sp" />
    </LinearLayout>


    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center"
        android:orientation="vertical">



        <!--Status of date selection shown here-->
        <TextView
            android:id="@+id/returnDate"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:paddingTop="25dp"
            android:text="No Date Selected"
            android:textAlignment="center"
            android:textColor="@color/black"
            android:textSize="22sp"  />

        <!--Selected date will be displayed in the text view below-->
        <TextView
            android:id="@+id/showDate"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:paddingBottom="25dp"
            android:text="DD-MM-YYYY"
            android:textAlignment="center"
            android:textColor="@color/black"
```

```
                android:textSize="30sp"
                android:textStyle="bold"/>


            <!--Date selection button used to select specific date-->
            <Button
                android:id="@+id/pickDateButton"
                android:layout_width="250dp"
                android:layout_height="wrap_content"
                android:layout_marginBottom="15dp"
                android:background="#E8CF2D"
                android:fontFamily="monospace"
                android:padding="20dp"
                android:text="Select date"
                android:textColor="@color/black"
                android:textSize="20dp" />
            <!--Confirmation button here is used to confirm and store the
    date in the DiaryDateStore-->
            <Button
                android:id="@+id/confirmDate"
                android:layout_width="250dp"
                android:layout_height="wrap_content"
                android:background="#E8CF2D"
                android:fontFamily="monospace"
                android:padding="20dp"
                android:text="Confirm Date"
                android:textColor="@color/black"
                android:textSize="20dp" />
        </LinearLayout>

    </LinearLayout>
```

3. diaryentry.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".DiaryEntryPage2">


    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <!--Back button is used to go to diarydate-->
        <Button
            android:id="@+id/backButton"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:background="#E8CF2D"
            android:fontFamily="monospace"
            android:padding="14dp"
            android:text="Back"
            android:textColor="@color/black"
            android:textSize="16sp" />
    </LinearLayout>


    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center"
        android:orientation="vertical">

        <!--Stored date from diarydate shown here-->
        <TextView
            android:id="@+id/dateText"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:paddingBottom="25dp"
            android:text="DD-MM-YYYY"
            android:textAlignment="center"
            android:textColor="@color/black"
            android:textSize="25sp"
            android:textStyle="bold" />

        <!--Text to tell User where to enter text for Diary entry-->
        <TextView
            android:id="@+id/textLabel"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:paddingBottom="20dp"
            android:text="Please add your entry below"
            android:textAlignment="center"
            android:textColor="@color/black"
            android:textSize="20sp" />

        <!--Text to enter Diary entry with autocorrect and capitalization of
sentences. -->

        <EditText
            android:id="@+id/entryDetail"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:background="@color/white"
            android:gravity="start|top"
            android:hint="Enter Text Here"

android:inputType="textMultiLine|textCapSentences|textAutoCorrect"
            android:lines="10"
            android:textColor="@color/black" />

        <TextView
            android:id="@+id/entryText"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="" />
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal">

            <!--Clear text box button-->
            <Button
                android:id="@+id/clearEntryButton"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:background="#E8CF2D"
                android:layout_marginLeft="20dp"
                android:layout_marginRight="10dp"
                android:fontFamily="monospace"
                android:text="Clear"
                android:textColor="@color/black"
                android:textSize="20dp" />

            <!--The add entry button stores the user inputted text into the
diaryDataStore-->
            <Button
                android:id="@+id/addEntryButton"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:layout_marginLeft="10dp"
                android:layout_marginRight="20dp"
                android:background="#E8CF2D"
                android:fontFamily="monospace"
                android:text="Add Entry"
                android:textColor="@color/black"
                android:textSize="20dp" />
        </LinearLayout>

    </LinearLayout>

</LinearLayout>
```

4. diaryentrylog.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    android:orientation="vertical"
    tools:context=".DiaryLogPage3">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">
        <RelativeLayout
            android:id="@+id/relativeLayout1"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_weight="1">

            <!--New Entry Button is used to go to diarydate page-->
            <Button
                android:id="@+id/newEntryButton"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_alignParentStart="true"
                android:background="#E8CF2D"
                android:fontFamily="monospace"
                android:padding="14dp"
                android:text="New Entry"
                android:textColor="@color/black"
                android:textSize="16dp" />

            <!--Delete all entries button used to delete all stored entries--
>

            <Button
                android:id="@+id/deleteAllEntry"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_alignParentEnd="true"
                android:background="#E8CF2D"
                android:fontFamily="monospace"
                android:padding="14dp"
                android:text="Delete All Entries"
                android:textColor="@color/black"
                android:textSize="16dp" />

        </RelativeLayout>

        <!--TextView is used to display the heading here-->
        <TextView
            android:id="@+id/myDiaryText"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:padding="14dp"
            android:text="My Diary"
            android:textStyle="bold"
            android:textAlignment="center"
            android:textColor="@color/black"
            android:textSize="22sp" />
    </LinearLayout>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="bottom"
        android:layout_weight="1"
        android:gravity="center">

        <!--TextView used to show all entries-->
        <TextView
```

```
            android:id="@+id/entryLogText"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:gravity="left"
            android:padding="20dp"
            android:scrollbars="vertical"
            android:text="No Entries Found"
            android:textColor="@color/black"
            android:textSize="18dp" />
    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <!-- Filter Date Button used to find entry of specified date from
DiaryDataStore-->
        <Button
            android:id="@+id/dateFilterButton"
            android:layout_width="70dp"
            android:layout_marginLeft="20dp"
            android:layout_marginRight="10dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:background="#E8CF2D"
            android:fontFamily="monospace"
            android:text="Filter date"
            android:textColor="@color/black"
            android:textSize="16dp" />

        <!--Refresh Diary Button used to show the new stored entries-->
        <Button
            android:id="@+id/refreshButton"
            android:layout_width="70dp"
            android:layout_height="wrap_content"
            android:layout_marginLeft="10dp"
            android:layout_marginRight="20dp"
            android:layout_weight="1"
            android:background="#E8CF2D"
            android:fontFamily="monospace"
            android:text="Refresh Diary"
            android:textColor="@color/black"
            android:textSize="16dp" />
    </LinearLayout>



    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:padding="6dp"
        android:layout_gravity="bottom"
        android:layout_weight="0">
        <!--Delete Filtered Entry is used to delete the entry that was 
specifically filtered-->
        <Button
            android:id="@+id/deleteDateButton"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
            android:background="#E8CF2D"
            android:fontFamily="monospace"
            android:padding="14dp"
            android:text="Delete Filtered Entry"
            android:textColor="@color/black"
            android:layout_marginBottom="20dp"
            android:textSize="16dp" />
    </LinearLayout>

</LinearLayout>
```

Kotlin

1.MainActivity.kt

```
package MAD.Assignment

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.Toast
import androidx.appcompat.widget.Toolbar
import androidx.viewpager2.widget.ViewPager2

class MainActivity : AppCompatActivity(), DairyParaPass{
    private lateinit var viewPager : ViewPager2
    //ViewPager here is used for setting view

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        //Toolbar is placed at top of page for the fragments
        val toolbar = findViewById<Toolbar>(R.id.toolsBar)
        setSupportActionBar(toolbar)

        viewPager = findViewById<ViewPager2>(R.id.pagerView)

        viewPager.isUserInputEnabled = false
        val adapter = DiaryAdapter(this,3)
        viewPager.adapter = adapter
        //Wipe the database on start up
        DiaryDatabase.getInstance(this@MainActivity).noteDao().nukeTable()
          }

    /**
     * This function is used to change the view between each fragment
     * @param page - The page number that has to be switched also
     */
    override fun onDataPass(page: Int) {
        viewPager.currentItem = page
    }

    /**
     * The function here is for adding a new diary entry to the database
     * @param date - The date selected to add entry
     * @param entry - The diary entry body is added here
     */
    override fun newEntry(date: String, entry: String) {
        val newN = DiaryDataStore(date, entry)

        //Here it is checked if a entry for date selected exists
        if(!containsPrimaryKey(date)){
            //If entry doesn't exist, create new entry and add it to the
database
            //shows toast message also

DiaryDatabase.getInstance(this@MainActivity).noteDao().insert(newN)
            Toast.makeText(this@MainActivity, "New entry for $date added,
please refresh entry list", Toast.LENGTH_LONG).show()
        }
        else {

            //If entry for this date already exists
            // Also add to existing entry
            val noteFound =
DiaryDatabase.getInstance(this@MainActivity).noteDao().loadAllByIds(date)
            //Remove the date from the not found.
            // Else entry toString would contains 2 same dates
            val notesString =
noteFound.toString().drop(date.length+3).replace("[", "").replace("]",
"").replace(",","").trim()
            val updatedNote = ("$notesString. $entry")

DiaryDatabase.getInstance(this@MainActivity).noteDao().updateNote(date,
updatedNote)
            Toast.makeText(this@MainActivity, "DiaryDataStore for $date
already exists, adding to your entry ...", Toast.LENGTH_LONG).show()

        }
    }

    /**
     * This function is used to delete all entries in database
     */
    override fun onDeletion()
    {
        DiaryDatabase.getInstance(this@MainActivity).noteDao().nukeTable()
    }

    /**
     * This function is used to get all entries in the database
     * @return List<DiaryDataStore> - The list of diary entries stored
     */
    override fun getEntries(): List<DiaryDataStore> {
        return
DiaryDatabase.getInstance(this@MainActivity).noteDao().getAll()
    }

    override fun deleteEntry(date_id: String) {

DiaryDatabase.getInstance(this@MainActivity).noteDao().deleteNote(date_id)
    }


    /**
     * This function is used to delete an entry by selected ID
     */
    override fun loadAllByIds(date: String): List<DiaryDataStore> {
        return
DiaryDatabase.getInstance(this@MainActivity).noteDao().loadAllByIds(date)
    }

    /**
     * This function is usd to check if an entry for selected date already
exists in database
     */
    override fun containsPrimaryKey(date_id:String): Boolean {
        return
DiaryDatabase.getInstance(this@MainActivity).noteDao().containsPrimaryKey(dat
e_id)
    }



}
```

2. DiaryDatePage1.kt

```
package MAD.Assignment

import android.app.DatePickerDialog
import android.content.Context
import android.os.Bundle
import androidx.fragment.app.Fragment
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.DatePicker
import androidx.lifecycle.ViewModelProvider
import MAD.Assignment.databinding.DiarydateBinding
import java.util.*


class DiaryDatePage1 : Fragment(R.layout.diarydate),
DatePickerDialog.OnDateSetListener{

    //Fragments are bound here to gain access to elements
    private var _binding : DiarydateBinding? = null
    private val binding get() = _binding!!

    //Variables for the Date
    var day = 0
    var month = 0
    var year = 0

    //Values saved for when date is needed to be displayed
    var savedDay = 0
    var savedMonth = 0
    var savedYear = 0

    //Date is inserted here
    lateinit var model: SharedViewModel
    //data is passed from here
    lateinit var dataPasser: DairyParaPass

    override fun onCreateView(inflater: LayoutInflater, container:
ViewGroup?, savedInstanceState: Bundle?): View {
        _binding = DiarydateBinding.inflate(inflater, container, false)
        return binding.root
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)
        model =
ViewModelProvider(requireActivity())[SharedViewModel::class.java]

        //Disabled confirm button until date is selected
        binding.confirmDate.alpha = .4f
        binding.confirmDate.isClickable = false

        //Provides a calender to allow client to select date
        binding.pickDateButton.setOnClickListener {
            getDateCalendar()
            DatePickerDialog(requireContext(), this, year, month, day).show()
            //below method is to confirm the date is picked
            pickDate()
        }
        //View entry button goes to diary entry log page
        binding.viewEntryButton.setOnClickListener {
            passData(2)
        }
    }

    /**
     *The function here is used for the date of the calender
     */
    private fun getDateCalendar(){
        val cal = Calendar.getInstance()
        day = cal.get(Calendar.DAY_OF_MONTH)
        month = cal.get(Calendar.MONTH)
        year = cal.get(Calendar.YEAR)
    }

    /**
     *The function here is used to confirm the date
     */
    private fun pickDate(){
            binding.confirmDate.setOnClickListener {
                //Date is saved in ViewModel and the displayed on diaryentry
page
                model.saveDate(savedDay, savedMonth, savedYear)
                binding.confirmDate.alpha = .4f
                binding.confirmDate.isClickable = false

                //Reset the date to nothing
                savedDay=0
                savedMonth=0
                savedYear=0

                //goes back to default values and goes to diary entry page
                binding.showDate.text="DD-MM-YY"
                binding.returnDate.text="No Date Selected"
                passData(1)
            }
    }

    /**
     * This function is used to destroy the binding when activity ends
     */
    override fun onDestroy() {
        super.onDestroy()
        _binding = null
    }

    /**
     * the function here is used to attach datapasser to context
     * for DairyParaPass
     */
    override fun onAttach(context: Context) {
        super.onAttach(context)
        dataPasser = context as DairyParaPass
    }

    /**
     * Data is passed to datapasser
     */
    private fun passData(data: Int){
        dataPasser.onDataPass(data)
    }

    /**
     * This function will run once the date is selected
     *
     * @param datePick - The datepicker
     * @param year - the year selected
     * @param month - the month selected
     * @param day - the day selected
     */
    override fun onDateSet(datePick: DatePicker?, year: Int, month: Int, day:
Int) {
        //the selected dates are saved here
        savedDay = day
        savedMonth = month+1
        savedYear = year
        //Disables confirm button
        binding.confirmDate.alpha = 1f;
        binding.confirmDate.isClickable = true;

        //Shows the selected date
        binding.returnDate.text="Date Selected"
        binding.showDate.text = "$savedDay-$savedMonth-$savedYear"
    }
}
```

3. DiaryEntryPage2.kt

```
package MAD.Assignment

import android.content.Context
import android.os.Bundle
import androidx.fragment.app.Fragment
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.Toast
import androidx.lifecycle.Observer
import androidx.lifecycle.ViewModelProvider
import MAD.Assignment.databinding.DiaryentryBinding


class DiaryEntryPage2 : Fragment() {

    //Variables for binding, viewModel and dataPasser here
    private var _binding : DiaryentryBinding? = null
    private val binding get() = _binding!!
    lateinit var model: SharedViewModel
    lateinit var dataPasser: DairyParaPass

    override fun onCreateView(inflater: LayoutInflater, container:
ViewGroup?, savedInstanceState: Bundle?): View {
        _binding = DiaryentryBinding.inflate(inflater, container, false)
        return binding.root
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)
        model =
ViewModelProvider(requireActivity())[SharedViewModel::class.java]

        model.date.observe(viewLifecycleOwner, Observer {
            //Shows the observed date
            binding.dateText.text= it.toString()
        })
        //Add Entry button actions here
        binding.addEntryButton.setOnClickListener {

            //This checks if there is text in the entryDetail
            if (binding.entryDetail.text.isEmpty()) {
                //If there is no input
                //Toast message if the entry is empty
                Toast.makeText(activity, "Your Diary Entry cannot be empty!",
Toast.LENGTH_LONG).show()

            }else{
                //If there is an input
                //takes date and diary entry to newNote class
                newEntry(binding.dateText.text.toString(),
binding.entryDetail.text.toString())
                binding.entryDetail.setText("")
                //Change screen to diaryentrylog
                passData(2)
            }
        }
        binding.backButton.setOnClickListener {
            //Deletes the text input by client and changes view to diarydate
            binding.entryDetail.setText("")
            passData(0)
        }
        binding.clearEntryButton.setOnClickListener {
            //Clear button used to delete input text and show toast message
            binding.entryDetail.setText("")
            Toast.makeText(activity, "Diary Entry Cleared",
Toast.LENGTH_LONG).show()
        }
    }

    /**
     * This function is used to destroy the binding when activity ends
     */
    override fun onDestroy() {
        super.onDestroy()
        _binding = null
    }

    /**
     * the function here is used to attach datapasser to context
     * for DairyParaPass
     */
    override fun onAttach(context: Context) {
        super.onAttach(context)
        dataPasser = context as DairyParaPass
    }

    /**
     * Data is passed to datapasser
     */
    private fun passData(data: Int){
        dataPasser.onDataPass(data)
    }

    /**
     * This function adds diary entry into the database
     * @param date - selected date
     * @param entry - entered DiaryDataStore
     */
    private fun newEntry(date:String, entry:String){
        dataPasser.newEntry(date, entry)
    }

}
```

4. DiaryLogPage3.kt

```
package MAD.Assignment

import MAD.Assignment.databinding.DiaryentrylogBinding
import android.app.DatePickerDialog
import android.content.Context
import android.os.Bundle
import androidx.fragment.app.Fragment
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.text.method.ScrollingMovementMethod
import android.widget.DatePicker
import android.widget.Toast
import androidx.lifecycle.ViewModelProvider
import java.util.*


class DiaryLogPage3 : Fragment(), DatePickerDialog.OnDateSetListener {

    private var _binding : DiaryentrylogBinding? = null
    private val binding get() = _binding!!
    lateinit var model: SharedViewModel
    lateinit var dataPasser: DairyParaPass

    //date variables
    var day = 0
    var month = 0
    var year = 0

    //values for date saved here
    var savedDay = 0
    var savedMonth = 1
    var savedYear = 0

    override fun onCreateView(inflater: LayoutInflater, container:
ViewGroup?, savedInstanceState: Bundle?): View {
        _binding = DiaryentrylogBinding.inflate(inflater, container, false)
        return binding.root
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)
        model =
ViewModelProvider(requireActivity())[SharedViewModel::class.java]

        //Scroll wheel option if there are several diary entries
        binding.entryLogText.movementMethod = ScrollingMovementMethod()
        val notes = getEntries()
        //Below brackets and colons are removed when printing diary entry
        val notesString = notes.toString().replace("[", "").replace("]",
"").replace(",","").trim()
        binding.entryLogText.text=notesString
        //Refresh button actions below
        binding.refreshButton.setOnClickListener {
            //deleteDateButton is active when diary entry is filtered
            binding.deleteDateButton.alpha = .4f
            //Client may refresh when on filter, so deleteDateButton is set
to not clickable if this happens
            binding.deleteDateButton.isClickable = false
            //Toast message here for client
            Toast.makeText(activity, "Refreshing Entries...",
Toast.LENGTH_LONG).show()
            //Here the entries are changed to a printable form
            val notes = getEntries()
            var notesString = notes.toString()
            if (notesString == "[]"){
                //If no entries are found
                binding.entryLogText.text="No Entries Found"
            }else{
                notesString = notesString.replace("[", "").replace("]",
"").replace(",","").trim()
                binding.entryLogText.text=notesString
            }
        }
        //Clear diary entry button below
        binding.deleteAllEntry.setOnClickListener {
            binding.deleteDateButton.alpha = .4f
            binding.deleteDateButton.isClickable = false
            // Method below delete all diary entries in the database
            clearNotes()
            //Shows message to tell client
            binding.entryLogText.text="All Entries Deleted"
            //toast message to tell client
            Toast.makeText(activity, "All entries have been deleted",
Toast.LENGTH_SHORT).show()
        }
        binding.dateFilterButton.setOnClickListener {
            getDateCalendar()
            DatePickerDialog(requireContext(), this, year, month, day).show()
        }
        //New Diary Entry button
        binding.newEntryButton.setOnClickListener {
            binding.deleteDateButton.alpha = .4f
            binding.deleteDateButton.isClickable = false
            //goes back to diarydate page
            passData(0)
        }
    }

    /**
     *The function here is used for the date of the calender
     */
    private fun getDateCalendar(){
        val cal = Calendar.getInstance()
        day = cal.get(Calendar.DAY_OF_MONTH)
        month = cal.get(Calendar.MONTH)
        year = cal.get(Calendar.YEAR)
    }

    /**
     * This function is used to destroy the binding when activity ends
     */
    override fun onDestroy() {
        super.onDestroy()
        _binding = null
    }

    /**
     * the function here is used to attach datapasser to context
     * for DairyParaPass
     */
    override fun onAttach(context: Context) {
        super.onAttach(context)
        dataPasser = context as DairyParaPass
    }

    /**
     * The function here is to delete date and diary entry
     */
     private fun deleteEntry(date_id: String) {
        dataPasser.deleteEntry(date_id)
    }

    /**
     * Data is passed to datapasser
     */
    private fun passData(data: Int){
        dataPasser.onDataPass(data)
    }

    /**
     * Function to get all the notes currently stored in the database
     * This function is used to get all the entries in the database
     * @return List<DiaryDataStore> - This is the list of all the diary
entries from the database
     */
    private fun getEntries():List<DiaryDataStore>{
        return dataPasser.getEntries()
    }

    private fun containsPrimaryKey(date: String): Boolean {
        return dataPasser.containsPrimaryKey(date)
    }

    private fun loadAllByIds(date: String): List<DiaryDataStore> {
        return dataPasser.loadAllByIds(date)
    }

    /**
     * The function here is to delete all entries in database
     */
    private fun clearNotes(){
        dataPasser.onDeletion()
    }

    /**
     * Function that runs when the date is selected
     *
     * @param datePick - The datepicker
     * @param year - the year selected
     * @param month - the month selected
     * @param day - the day selected
     */
    override fun onDateSet(datePick: DatePicker?, year: Int, month: Int, day:
Int) {
        savedDay = day
        savedMonth = month+1
        savedYear = year
        displayFilters()
    }

    /**
     * the function below will only display the diary entry specified by
selected date
     */
    private fun displayFilters(){
        var dateFilter = "$savedDay-$savedMonth-$savedYear"

        //Checks if there are any entries for the specified date
        if (containsPrimaryKey(dateFilter)){
            //If DiaryDataStore found
            //Toast message telling client that entry has been found
            Toast.makeText(activity, "Diary Entry for $dateFilter have been
found", Toast.LENGTH_SHORT).show()
            //Gets entries for this date
            val notesDate = loadAllByIds(dateFilter)
            val notesString = notesDate.toString().replace("[",
"").replace("]", "").replace(",","").trim()
            //Display the entries
            binding.entryLogText.text=notesString
            binding.deleteDateButton.alpha = 1f
            //delete button active here
            binding.deleteDateButton.isClickable = true
            binding.deleteDateButton.setOnClickListener {
                //Disable delete button so the client cant edit
                binding.deleteDateButton.alpha = .4f
                binding.deleteDateButton.isClickable = false
                deleteEntry(dateFilter)
                binding.entryLogText.text="No entries for $dateFilter have
been found"
                Toast.makeText(activity, "Entries for $dateFilter have been
deleted", Toast.LENGTH_SHORT).show()

            }

        }else if(!containsPrimaryKey(dateFilter)) {
            // Error message when no entry is found
            binding.entryLogText.text="No Entry Found"
            Toast.makeText(activity, "No entries have been found for
$dateFilter", Toast.LENGTH_SHORT).show()
        }
    }
}
```

5.DiaryDataStore.kt

```
package MAD.Assignment

import androidx.room.ColumnInfo
import androidx.room.Entity
import androidx.room.PrimaryKey

/**
 * the data class is used to store the date and the diary entry
 * @param date - The date is stored here
 * @param noteBody - The diary entry is stored here
 */
@Entity
data class DiaryDataStore(
    @PrimaryKey val date: String,
    @ColumnInfo(name = "note_body") val noteBody: String?
)

/**
 * The toString method below is used to print the diary entry in
diaryentrylog
 * @return - prints the string of for the diary entry
 */
{
    override fun toString(): String = "$date - $noteBody\n\n"

}
```

6.DiaryDataQuery.kt

```
package MAD.Assignment

import androidx.room.*

/**
 * Here are database interactions needed for functioning
 */
@Dao
interface DiaryDataQuery{

    //Query to get list of all diary entries
    @Query("SELECT * FROM DiaryDataStore")
    fun getAll(): List<DiaryDataStore>

    //For inserting a new diaryDataStore
    @Insert
    fun insert(diaryDataStore:DiaryDataStore)
    //Getting entry from a selected date for filtered function
    @Query("SELECT * FROM DiaryDataStore WHERE date IN (:date_id)")
    fun loadAllByIds(date_id: String): List<DiaryDataStore>

    //Delete filtered entry from the database
    @Query("DELETE FROM DiaryDataStore WHERE date = :date_id")
    fun deleteNote(date_id: String);

    //The query used here is to find diary entry of specified date
    @Query("SELECT count(*)!=0 FROM DiaryDataStore WHERE date IN (:date_id)")
    fun containsPrimaryKey(date_id: String): Boolean

    @Query("UPDATE DiaryDataStore SET note_Body=:newNote WHERE date=:dateID")
    fun updateNote(dateID: String, newNote: String)

    //Delete all diary entries stored in database
    @Query("DELETE FROM DiaryDataStore")
    fun nukeTable()
}
```

7.DiaryDatabase.kt

```
package MAD.Assignment

import android.content.Context
import androidx.room.Database
import androidx.room.Room
import androidx.room.RoomDatabase


/**
 * The database here is used to store the diary entries by the client
 */
@Database(entities = [DiaryDataStore::class], version = 1)
abstract class DiaryDatabase : RoomDatabase() {
    abstract fun noteDao(): DiaryDataQuery //Database Access Object for
database interactions
    companion object{
        var INSTANCE: DiaryDatabase?=null
        fun getInstance(context: Context):DiaryDatabase
        {
            if(INSTANCE==null){ //If an instance of this class doesn't
already exist
                INSTANCE = Room.databaseBuilder(
                    context.applicationContext,
                    DiaryDatabase::class.java, "note_database"   //Name of
database
                ).allowMainThreadQueries().build()
            }
            return INSTANCE!!   //Return this instance
        }
    }
}
```

8.DiaryAdapter.kt

```
package MAD.Assignment

import androidx.fragment.app.Fragment
import androidx.fragment.app.FragmentActivity
import androidx.viewpager2.adapter.FragmentStateAdapter
class DiaryAdapter(fa: FragmentActivity, private val mNumOfTabs: Int) :
    FragmentStateAdapter(fa) {
    /**
     * Used to get the number of pages
     */
    override fun getItemCount(): Int {
        return mNumOfTabs
    }
    /**
     * Assigning position to fragments
     */
    override fun createFragment(position: Int): Fragment {
        return when (position) {
            0 -> DiaryDatePage1()
            1 -> DiaryEntryPage2()
            2 -> DiaryLogPage3()
            else -> DiaryDatePage1()
        }
    }
}
```

9.DiaryParaPass.kt

```
package MAD.Assignment

/**
 * Here is where the data is getting passed using the functions below
 */
interface DairyParaPass {
        fun onDataPass(page: Int)

        fun newEntry(date: String, entry: String)

        fun onDeletion()

        fun getEntries():List<DiaryDataStore>

        fun deleteEntry(date_id:String)

        fun containsPrimaryKey(date_id:String):Boolean

        fun loadAllByIds(date:String): List<DiaryDataStore>

}
```

10.fragmentViewModel.kt

```
package MAD.Assignment

import androidx.lifecycle.LiveData
import androidx.lifecycle.ViewModel
import androidx.lifecycle.MutableLiveData

/**
 * ViewModel class used for storing and observing the date from page 1
fragment to page 2 using LiveData
 */
class SharedViewModel: ViewModel() {

    private var _date = MutableLiveData("")
    val date: LiveData<String> = _date

    /**
     * This function is used for saving the data in a suitable format to the
liveData variable
     * @param newDay - Day being added here
     * @param newMonth - Month being added here
     * @param newYear - Year being added here
     */
    fun saveDate(newDay: Int, newMonth:Int, newYear:Int){
        _date.value = ("$newDay-$newMonth-$newYear")
    }
}
```