Studying Buddy
Project Proposal

Version 1.0

Prepared by Andrew Le, Andrew Noel, Nathan Raras, Ayoub Ibrahim, Salma Rashed, Salma Elkadi

California State University San Marcos

May 11th, 2023

Reasons We Chose This Project

The main goal of our application is to drift from the concept of studying without a proper plan by implementing multiple studying techniques and methods to allow a more optimal learning environment for students reviewing all types of course material. In addition to creating an efficient studying environment, developing a studying application can be beneficial for the following reasons:

Accessibility: A studying app can be easily accessed from anywhere, any time, and on any device, making it a convenient tool for students who are always on the go or prefer to study in different environments other than at home.

Personalized Learning: Students can customize their learning experience by selecting specific studying methods and techniques that are the most optimal for their desired subject to ensure efficiency and productivity. With a more personalized learning environment, students are able to avoid burnout while focusing on areas they feel the need to improve on and spend less time on topics they are already comfortable with.

Interactive Learning: This application can offer a variety of interactive learning tools such as quizzes, flashcards, and tests in addition to selecting from a pool of studying methods. With an abundance of available tools to utilize for studying, it will push more engagement from the user ultimately becoming more effective than studying from a textbook or powerpoint slides.

Tracking Progress: In order to keep students motivated, the application can track students' progress, identify their strengths and weaknesses with areas they are confident and not confident in, and provide personal feedback to help improve their learning outcomes.

Time Management: To refrain from overly studying and burnout within students, the application is able to help students manage their time studying while following the studying techniques and methods. In addition, the application encourages the user to take scheduled study breaks in between their study session to avoid burnout and help in retaining the information the student had reviewed. The application is also time sensitive for when students have a limited time to study or review in which they can still obtain the knowledge regardless of the time they have.

Potential Kotlin concepts

The first Kotlin concept that would need to be implemented is proper user interface design. This is a critical feature for the application as it is directly related to the success of the application in respect to user usage. It should be designed to be easily navigable, intuitive, and visually appealing. This can be achieved by implementing proper button displays and keeping in mind intuitive placement. Another addition that would help assist with this is implementation of list and scroll views. This would allow for better movement and flow for each of the user-interactable screens.

Finally, another important Kotlin feature that is integral to the services of Studying Buddy is the usage of notifications. The application should be able to notify the user whenever there is an instance within the study methodology that requires the user's attention. Some examples would be when the student has reached a break period or when the student's break period is about to end. These notifications should not just be system notifications, but they should have sound cues and visual alarms as well. This will ensure that the student can study effectively given the parameterized constraints that were set or in place. Furthermore, the system notifications should also account for the psychology of students as there should be warnings for state transitions and an option to create a "pleasant" transition or a more abrupt one for students that require different study environments.

Potential Features

Study Technique Selection: A main feature of the app would be to allow users to select a studying technique based on their preference and the subject they are studying for. You can provide a list of examples of studying techniques: SQ3R, Cornell Method, Feynman Technique, etc.

Customizable Study Techniques*:* Allowing users to customize their own study technique. Users could add or remove specific steps, change the time allotted for each step, and save their customized techniques for future use.

Progress Tracking: The app can track the user's progress, including the amount of time spent on each study session, the number of completed sessions, and the percentage of the chapter/subject covered. This will help users stay motivated and see the progress they're making over time.

Login Credentials: The app will launch and ask the user to login using their email and password. If the user puts in their email and password and hits the button labeled “create account” the app will then search throught the firebase authentication database and check to see if there is already an account with that email. If there is an email already in the database, it will tell the user that there was an authentication error. If there is no email in the database that the user wants to use, it will saed the email and username into the database. All emails used must be in correct format meaning it must be in the syntax of an email (ending int @gmail.com, @yahoo.com, etc.). WHen the user inputs their email and password and clicks “login” the app will check with the database to make sure the credentials match on both sides. If they do not match or there is no entry in the database for that email, it will tell the user “autheticaion error”. If the credentials match in the database, it will take the user to the home page. If the user wishes not to create an account, they can hit the guest button and this will take them straight to the main page without checking the authentication database.

App Description:

Study Buddy - the ultimate studying app designed to help students achieve their full potential! With a variety of studying techniques and methods to choose from, Study Buddy is the perfect tool to create an efficient studying environment. The app can be easily accessed from anywhere, at any time, and on any device, making it a convenient tool for students who are always on the go or prefer to study in different environments.

Personalized learning is also a key feature of Study Buddy, allowing students to customize their learning experience by selecting specific studying methods and techniques that are most optimal for their desired subject. With an abundance of interactive learning tools such as quizzes, flashcards, and tests, students can engage with the material and track their progress.

Time management is also made simple with Study Buddy. The app helps students manage their time studying while following the studying techniques and methods. The app also encourages the user to take scheduled study breaks in between their study session to avoid burnout and help in retaining the information the student had reviewed.

Using Kotlin, we have implemented proper user interface design to create an intuitive and visually appealing app. Our usage of notifications is another integral feature, ensuring that the student can study effectively given the parameterized constraints that were set or in place. This feature provides warnings for state transitions or a more abrupt one for students that require different study environments.

Limitations:

Many restrictions were found as the StudyBuddy 2.0 app was being developed. The user experience and overall functionality of the app may be impacted by these restrictions. The main restrictions encountered are as follows:

Lack of Error Handling:

Comprehensive error handling is not present in the current implementation. The program displays a generic Toast warning whenever an incorrect technique is chosen or necessary data is insufficient. The user's capacity to comprehend and resolve particular problems is so constrained. Future enhancements should concentrate on introducing more robust error handling, offering thorough error messages, and gently managing unforeseen events.

Limited Study Techniques:

The program currently only provides SQ3R, Pomodoro, and Feynman as its predefined study methods. Although these strategies are useful, not all users' preferences or study styles may be accommodated by all. Future upgrades might offer more study approaches or give users the option to make their own unique study procedures in order to increase user pleasure and customisation.

Lack of User Feedback:

The current implementation does not provide users with clear feedback when they choose a technique or begin a study session To enhance the user experience and deliver greater feedback on the performance of the software, visual indicators of progress updates could be incorporated. Users would be better able to grasp the results of their activities and would have a more enjoyable learning experience if such feedback methods were included.

UI Design Considerations:

Potential user interface design restrictions are not addressed by the code. Future improvements should take into account things like accessibility, device responsiveness, and uniform styling throughout the app. A user-friendly and aesthetically pleasing UI will improve the StudyBuddy 2.0 app's overall usability and happiness.

Lack of Background Timer Functionality:

The absence of a background timer feature in the StudyBuddy 2.0 software is one of its drawbacks. Currently, the timer will pause and not keep running in the background if the user closes the app or opens another one while it is in use. Due to the possibility of being unable to multitask or use other applications while keeping track of their study time, this restriction may negatively affect the user's ability to learn. If background timer capability is added, the timer will continue to run even when the app is not active. Users can utilize other apps or complete other chores on their device in this manner without interfering with their learning session.

Addressing these limitations would significantly improve the StudyBuddy 2.0 app, providing a more robust and user-friendly study tool for users.

Contributions:

Contributions	**Memebers**
API Implementations	Ayoub Ibrahim, Nathan Raras, Andrew Le, Andrew Noel, Salma Elkadi
UI Design	Salma Rashed
Writing Report	Salma Rashed, Salma Elkadi, Andrew Noel

Conclusion:

In conclusion, our studying application aims to revolutionize the traditional approach to studying by providing a comprehensive and personalized learning environment for students. By incorporating multiple studying techniques and methods, the app ensures that students can optimize their learning experience and achieve better outcomes across various course materials.

One of the key advantages of our app is its accessibility. Students can easily access the application from any location, at any time, and on any device, allowing for flexibility and convenience in their study routines. This feature is particularly beneficial for students who are constantly on the go or prefer studying in different environments.

Personalized learning is another important aspect of our application. Students have the freedom to choose specific studying methods and techniques that align with their preferred subjects, enabling them to study efficiently and effectively. This customization not only helps avoid burnout but also allows students to focus on areas where they need improvement while spending less time on topics they are already comfortable with.

Our application promotes interactive learning through various tools such as quizzes, flashcards, and tests. By offering a range of interactive features, we aim to enhance student engagement and make the studying process more enjoyable and effective than traditional methods like textbooks or slide presentations.

Furthermore, our application incorporates progress tracking, which plays a vital role in keeping students motivated. By monitoring their progress, identifying strengths and weaknesses, and providing personalized feedback, students can continuously improve their learning outcomes and stay motivated throughout their study journey.

Time management is another significant aspect addressed by our application. It helps students manage their study time effectively by following suggested studying techniques and methods.

The app also encourages students to take scheduled study breaks, preventing burnout and aiding information retention. Additionally, the application is designed to accommodate students with limited study time, ensuring that they can still acquire knowledge efficiently within their available timeframe.

To conclude, our studying application offers a comprehensive solution to enhance the learning experience for students. By combining accessibility, personalized learning, interactive tools, progress tracking, and effective time management, we aim to provide a platform that empowers students to study smarter, achieve better results, and foster a positive attitude towards learning.

Code:

GitHub repo: https://github.com/andrewnoel99/StudyBuddy.git

MainActivity.kt

```
package com.example.studybuddy20

import android.content.Intent
import android.media.MediaPlayer
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.os.CountDownTimer
import android.view.Menu
import android.view.MenuItem
import android.view.View
import android.widget.Button
import android.widget.TextView
import com.example.studybuddy20.databinding.ActivityMainBinding
import com.google.android.material.bottomnavigation.BottomNavigationView

class MainActivity : AppCompatActivity(), View.OnClickListener {

   private lateinit var studyMethods: Button
   private lateinit var timerButton: Button
   private lateinit var mediaPlayer: MediaPlayer


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

       studyMethods = findViewById(R.id.StudyMethodsBtn)
       timerButton = findViewById(R.id.btnTimer)

       studyMethods.setOnClickListener(this)
       timerButton.setOnClickListener(this)

       val bottomNavigation: BottomNavigationView =
findViewById(R.id.bottomNav)
       bottomNavigation.setOnItemSelectedListener { item ->
           onNavigationItemSelected(item)
       }
   }

   override fun onClick(view: View?) {
       when (view?.id) {
           R.id.btnTimer ->{
               val intent = Intent(this, timerTaskActivity::class.java)
               startActivity(intent)
           }
           R.id.StudyMethodsBtn ->{
               val intent = Intent(this, StudyActivity::class.java)
               startActivity(intent)
           }
       }
   }

   private fun onNavigationItemSelected(item: MenuItem): Boolean{
       when (item.itemId){
           R.id.home ->{
               val intent = Intent(this, MainActivity::class.java)
               startActivity(intent)
           }
           R.id.music ->{
               val intent = Intent(this, MusicPlayer::class.java)
               startActivity(intent)
           }
           R.id.logout ->{
               if(::mediaPlayer.isInitialized) {
                   mediaPlayer.stop()
                   mediaPlayer.release()

               }
               val intent = Intent(this, LoginActivity2::class.java)
               startActivity(intent)
               finish()
           }
       }
       return false;
   }

}
```

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:id="@+id/scroll"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".MainActivity"
   android:background="@color/beige">


   <ImageView
       android:id="@+id/logoImage"
```

```
        android:layout_width="259dp"
        android:layout_height="242dp"
        android:src="@drawable/kindpng_793803"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/WelcomeText"
        app:layout_constraintVertical_bias="0.248" />

    <ImageView
        android:id="@+id/grad_hat"
        android:layout_width="58dp"
        android:layout_height="72dp"
        android:src="@drawable/kindpng_490791"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.065"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.077" />

    <TextView
        android:id="@+id/WelcomeText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="Welcome to Study Buddy"
        android:textColor="@color/black"
        android:textSize="30sp"
        android:textStyle="italic"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.134" />

    <Button
        android:id="@+id/btnTimer"
        android:layout_width="170dp"
        android:layout_height="51dp"
        android:text="Timer"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:textColor="@android:color/white"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.933"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/logoImage"
        app:layout_constraintVertical_bias="0.381" />
    <Button
        android:id="@+id/StudyMethodsBtn"
        android:layout_width="185dp"
        android:layout_height="51dp"
        android:text="Study Methods"
        android:textAppearance="?android:attr/textAppearanceLarge"

        android:textColor="@android:color/white"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.101"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/logoImage"
        app:layout_constraintVertical_bias="0.381" />

    <com.google.android.material.bottomnavigation.BottomNavigationView
        android:id="@+id/bottomNav"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="63dp"
        android:layout_marginEnd="10dp"
        android:layout_marginBottom="1dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/btnTimer"
        app:menu="@menu/bottom_nav"/>

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

LoginActivity2.kt

```
package com.example.studybuddy20
import android.content.ContentValues.TAG
import android.content.Intent
import android.os.Bundle
import android.util.Log
import android.widget.Button
import android.widget.EditText
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import com.google.firebase.auth.FirebaseAuth

class LoginActivity2 : AppCompatActivity() {

    private lateinit var mAuth: FirebaseAuth

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_login2)

        mAuth = FirebaseAuth.getInstance()

        val emailEditText = findViewById<EditText>(R.id.userEmailET)
        val passwordEditText = findViewById<EditText>(R.id.userPasswordET)
        val loginButton = findViewById<Button>(R.id.login_button)
        val createAccountButton =
findViewById<Button>(R.id.create_account_button)
        val guest_button = findViewById<Button>(R.id.guestButton)

        loginButton.setOnClickListener {
            val email = emailEditText.text.toString()
            val password = passwordEditText.text.toString()

            mAuth.signInWithEmailAndPassword(email, password)
                .addOnCompleteListener(this) { task ->
                    if (task.isSuccessful) {
                        val intent = Intent(this, MainActivity::class.java)
                        startActivity(intent)
                        finish()
                    } else {
                        Log.d(TAG, "signInWithEmail:failure", task.exception)
                        Toast.makeText(
                            this,
                            "Authentication failed.",
                            Toast.LENGTH_SHORT
                        ).show()
                    }
                }
        }

        createAccountButton.setOnClickListener {

            val email = emailEditText.text.toString()
            val password = passwordEditText.text.toString()

            mAuth.createUserWithEmailAndPassword(email, password)
                .addOnCompleteListener(this) { task ->
                    if (task.isSuccessful) {
                        Toast.makeText(
                            this,
                            "Account created.",
                            Toast.LENGTH_SHORT
                        ).show()
                        val intent = Intent(this, MainActivity::class.java)
                        startActivity(intent)

                        finish()
                    } else {
                        Toast.makeText(
                            this,
                            "Account creation failed.",
                            Toast.LENGTH_SHORT
                        ).show()
                    }
                }
        }

        //when guest button is clicked, go to main activity
        guest_button.setOnClickListener {
            val intent = Intent(this, MainActivity::class.java)
            startActivity(intent)
        }
    }
}
```

Activity_login2.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:id="@+id/LoginActivity2"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:background="@color/beige"
   tools:context=".LoginActivity2">

   <Button
       android:id="@+id/create_account_button"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:layout_marginTop="44dp"
       android:text="Create Account"
       app:layout_constraintEnd_toEndOf="parent"
       app:layout_constraintHorizontal_bias="0.497"
       app:layout_constraintStart_toStartOf="parent"
       app:layout_constraintTop_toBottomOf="@+id/login_button" />

   <EditText
       android:id="@+id/userEmailET"
       android:layout_width="295dp"
       android:layout_height="56dp"
       android:layout_marginTop="120dp"
       android:ems="10"
       android:hint="Email"
       android:inputType="textEmailAddress"
```

```
        android:padding="16dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/userPasswordET"
        android:layout_width="295dp"
        android:layout_height="59dp"
        android:layout_marginTop="40dp"
        android:ems="10"
        android:hint="Password"
        android:inputType="textPassword"
        android:labelFor="@+id/userPasswordET"
        android:padding="16dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.497"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/userEmailET" />

    <Button
        android:id="@+id/login_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="68dp"
        android:text="Login"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.501"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/userPasswordET" />

    <Button
        android:id="@+id/guestButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="46dp"
        android:text="Guest"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/create_account_button" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="36dp"
        android:layout_marginBottom="36dp"
        android:text="Study Buddy"
        android:textColor="#000000"
        android:textSize="35sp"
        android:textStyle="bold|italic"
        app:layout_constraintBottom_toTopOf="@+id/userEmailET"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.497"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

TimerActivity.kt

```
package com.example.studybuddy20

import android.content.Intent
import android.graphics.Color
import android.os.Bundle
import android.os.CountDownTimer
import android.util.Log
import android.view.MenuItem
import android.view.View
import android.widget.Button
import android.widget.ImageButton
import android.widget.ProgressBar
import android.widget.TextView
import androidx.appcompat.app.AppCompatActivity
import com.google.android.material.bottomnavigation.BottomNavigationView

class TimerActivity : AppCompatActivity() {

    private lateinit var techniqueTitle: String
    private lateinit var stepDurations: List<Pair<String, Long>>
    private lateinit var currentStep: String
    private var currentStepIndex: Int = 0
    private var timeRemaining: Long = 0
    private lateinit var timer: CountDownTimer
    private lateinit var pause: Button
    private lateinit var play: Button


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



        // Retrieve the technique title and step durations from the intent
        val intent = intent
        techniqueTitle = intent.getStringExtra("techniqueTitle")!!
        stepDurations = intent.getSerializableExtra("steps") as
List<Pair<String, Long>>
```

```
        // Set the current step to the first step
        currentStepIndex = 0
        currentStep = stepDurations[currentStepIndex].first

        // Start the timer with the duration of the first step
        timeRemaining = stepDurations[currentStepIndex].second
        startTimer()

        // Set the technique title and current step on the screen
        findViewById<TextView>(R.id.current_step_text_view).text =
techniqueTitle
        findViewById<TextView>(R.id.text_current_step).text = currentStep



        findViewById<Button>(R.id.pause_button).setOnClickListener { view ->
            onPauseButtonClick(view)
        }

        findViewById<Button>(R.id.play_button).setOnClickListener { view ->
            onPlayButton(view)
        }


        findViewById<Button>(R.id.play_button).setOnClickListener { view ->
            onPlayButton(view)
            findViewById<Button>(R.id.reset_button).isEnabled = true
        }


        val bottomNavigation: BottomNavigationView =
findViewById(R.id.bottomNav)
       bottomNavigation.setOnItemSelectedListener { item ->
            onNavigationItemSelected(item)
        }

    }

    private fun startTimer() {
        timer = object : CountDownTimer(timeRemaining * 1000, 1000) {
            override fun onFinish() {
                // If this is not the last step, move to the next step and
start the timer again
                if (currentStepIndex < stepDurations.size - 1) {
                    currentStepIndex++
```

```kotlin
                    currentStep = stepDurations[currentStepIndex].first
                    timeRemaining = stepDurations[currentStepIndex].second
                    startTimer()

                    // Update the screen with the new step
                    findViewById<TextView>(R.id.text_current_step).text =
currentStep
                } else {
                    // If this is the last step, finish the activity
                    finish()
                }
            }

            override fun onTick(millisUntilFinished: Long) {
                // Update the time remaining on the screen
                timeRemaining = millisUntilFinished / 1000
                val minutes = timeRemaining / 60
                val seconds = timeRemaining % 60
                findViewById<TextView>(R.id.timer_text_view).text =
                    String.format("%02d:%02d", minutes, seconds)


            }
        }.start()
    }

    override fun onDestroy() {
        super.onDestroy()
        timer.cancel()
    }

    private fun onPauseButtonClick(view: View) {
        timer.cancel()
        findViewById<Button>(R.id.play_button).isEnabled = true
        findViewById<Button>(R.id.pause_button).isEnabled = false
    }

    private fun onPlayButton(view: View) {
        startTimer()
        findViewById<Button>(R.id.play_button).isEnabled = false
        findViewById<Button>(R.id.pause_button).isEnabled = true
    }
    fun onResetButton(view: View) {
        // Cancel the current timer
        timer.cancel()

        // Start a new timer with the duration of the current step
        timeRemaining = stepDurations[currentStepIndex].second
        startTimer()
        // Update the screen with the new time remaining
        val minutes = timeRemaining / 60
        val seconds = timeRemaining % 60
        findViewById<TextView>(R.id.timer_text_view).text =
String.format("%02d:%02d", minutes, seconds)


    }

    private fun onNavigationItemSelected(item: MenuItem): Boolean{
        when (item.itemId){
            R.id.home ->{
                val intent = Intent(this, MainActivity::class.java)
                startActivity(intent)
            }
            R.id.music ->{
                val intent = Intent(this, MusicPlayer::class.java)
                startActivity(intent)
            }
            R.id.logout ->{
                val intent = Intent(this, LoginActivity2::class.java)
                startActivity(intent)
            }
        }
        return false;
    }

}
```

Activity_timer.xml

```
package com.example.studybuddy20

import android.content.Intent
import android.graphics.Color
import android.os.Bundle
import android.os.CountDownTimer
import android.util.Log
import android.view.MenuItem
import android.view.View
import android.widget.Button
import android.widget.ImageButton
import android.widget.ProgressBar
import android.widget.TextView
import androidx.appcompat.app.AppCompatActivity
import com.google.android.material.bottomnavigation.BottomNavigationView

class TimerActivity : AppCompatActivity() {

    private lateinit var techniqueTitle: String
```

```
   private lateinit var stepDurations: List<Pair<String, Long>>
   private lateinit var currentStep: String
   private var currentStepIndex: Int = 0
   private var timeRemaining: Long = 0
   private lateinit var timer: CountDownTimer
   private lateinit var pause: Button
   private lateinit var play: Button


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



       // Retrieve the technique title and step durations from the intent
       val intent = intent
       techniqueTitle = intent.getStringExtra("techniqueTitle")!!
       stepDurations = intent.getSerializableExtra("steps") as
List<Pair<String, Long>>

       // Set the current step to the first step
       currentStepIndex = 0
       currentStep = stepDurations[currentStepIndex].first

       // Start the timer with the duration of the first step
       timeRemaining = stepDurations[currentStepIndex].second
       startTimer()

       // Set the technique title and current step on the screen
       findViewById<TextView>(R.id.current_step_text_view).text =
techniqueTitle
       findViewById<TextView>(R.id.text_current_step).text = currentStep




       findViewById<Button>(R.id.pause_button).setOnClickListener { view ->
           onPauseButtonClick(view)
       }

       findViewById<Button>(R.id.play_button).setOnClickListener { view ->
           onPlayButton(view)
       }



       findViewById<Button>(R.id.play_button).setOnClickListener { view ->
```

```
            onPlayButton(view)
            findViewById<Button>(R.id.reset_button).isEnabled = true
        }


        val bottomNavigation: BottomNavigationView =
findViewById(R.id.bottomNav)
        bottomNavigation.setOnItemSelectedListener { item ->
            onNavigationItemSelected(item)
        }

    }

    private fun startTimer() {
        timer = object : CountDownTimer(timeRemaining * 1000, 1000) {
            override fun onFinish() {
                // If this is not the last step, move to the next step and
start the timer again
                if (currentStepIndex < stepDurations.size - 1) {
                    currentStepIndex++
                    currentStep = stepDurations[currentStepIndex].first
                    timeRemaining = stepDurations[currentStepIndex].second
                    startTimer()

                    // Update the screen with the new step
                    findViewById<TextView>(R.id.text_current_step).text =
currentStep
                } else {
                    // If this is the last step, finish the activity
                    finish()
                }
            }

            override fun onTick(millisUntilFinished: Long) {
                // Update the time remaining on the screen
                timeRemaining = millisUntilFinished / 1000
                val minutes = timeRemaining / 60
                val seconds = timeRemaining % 60
                findViewById<TextView>(R.id.timer_text_view).text =
                    String.format("%02d:%02d", minutes, seconds)


            }
        }.start()
    }

    override fun onDestroy() {
        super.onDestroy()
        timer.cancel()
    }

    private fun onPauseButtonClick(view: View) {
        timer.cancel()
        findViewById<Button>(R.id.play_button).isEnabled = true
        findViewById<Button>(R.id.pause_button).isEnabled = false
    }

    private fun onPlayButton(view: View) {
        startTimer()
        findViewById<Button>(R.id.play_button).isEnabled = false
        findViewById<Button>(R.id.pause_button).isEnabled = true
    }
    fun onResetButton(view: View) {
        // Cancel the current timer
        timer.cancel()

        // Start a new timer with the duration of the current step
        timeRemaining = stepDurations[currentStepIndex].second
        startTimer()

        // Update the screen with the new time remaining
        val minutes = timeRemaining / 60
        val seconds = timeRemaining % 60
        findViewById<TextView>(R.id.timer_text_view).text =
String.format("%02d:%02d", minutes, seconds)


    }

    private fun onNavigationItemSelected(item: MenuItem): Boolean{
        when (item.itemId){
            R.id.home ->{
                val intent = Intent(this, MainActivity::class.java)
                startActivity(intent)
            }
            R.id.music ->{
                val intent = Intent(this, MusicPlayer::class.java)
                startActivity(intent)
            }
            R.id.logout ->{
                val intent = Intent(this, LoginActivity2::class.java)
                startActivity(intent)
            }
        }
        return false;
    }

}
```

StudyActivity.kt

```
package com.example.studybuddy20

import android.content.Intent
import android.graphics.Color
import android.media.MediaPlayer
import android.os.Bundle
import android.util.Log
import android.view.MenuItem
import android.widget.Button
import android.widget.TextView
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import androidx.recyclerview.widget.LinearLayoutManager
import androidx.recyclerview.widget.RecyclerView
import com.google.android.material.bottomnavigation.BottomNavigationView
import java.io.Serializable

class StudyActivity : AppCompatActivity() {

    private lateinit var startButton: Button
    private lateinit var recyclerView: RecyclerView
    private lateinit var selectedMethodTextView: TextView
    private lateinit var mediaPlayer : MediaPlayer



    private val techniques = listOf(
        Technique(
            "SQ3R", listOf(
                "Survey : skimming the chapter and taking notes",
                "Break",
                "Question : formulate questions around the chapter's content",
                "Break",
                "Read: read full chapter and look for answers to the questions
you made",
                "Break",
                "Recite: summarize what you just read, recall and identify
major points",
                "Break",
                "Review: review material, quiz yourself"
            )
        ),
        Technique(
            "Pomodoro", listOf(
                "Work: focus on a task",
                "Break",
                "Work: focus on a task",
                "Break",
                "Work: focus on a task",
                "Break",
                "Work: focus on a task",
                "Break"
            )
        ),
        Technique(
            "Feynman", listOf(
                "Study: focus on a topic",
                "Break",
                "Explain: explain what you just studied, as if you were
 teaching it to someone",
                "Break",
                "Identify gaps: identify areas you struggled with and revisit
 them",
                "Break",
                "Review: review material, quiz yourself"
            )
        )
    )

    val myTechniques = mapOf(
        "SQ3R" to listOf(
            "Survey \n\n Skimming the chapter and taking notes" to 1,
            "Break" to 1,
            "Question \n\n Formulate questions around the chapter's content"
 to 1,
            "Break" to 1,
            "Read \n\n Read full chapter and look for answers to the questions
 you made" to 1,
            "Break" to 1,
            "Recite \n\n Summarize what you just read, recall and identify
 major points" to 1,
            "Break" to 1,
            "Review\n\n Review material, quiz yourself" to 2
        ),
        "Pomodoro" to listOf(
            "Work \n\n Focus on a task" to 25,
            "Break" to 5,
            "Work \n" +
                    "\n" +
                    " Focus on a task" to 25,
            "Break" to 5,
            "Work \n" +
                    "\n" +
                    " Focus on a task" to 25,
            "Break" to 5,
            "Work \n" +
                    "\n" +
```

```kotlin
                   " Focus on a task" to 25,
           "Break" to 5
       ),
       "Feynman" to listOf(
           "Learn" to 30,
           "Break" to 5,
           "Explain" to 30,
           "Break" to 5,
           "Review" to 30
       )
   )



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

       startButton = findViewById(R.id.button_start_studying)
       startButton.setBackgroundColor(Color.parseColor("#356859"))

       // Find the RecyclerView and set its layout manager
       recyclerView = findViewById(R.id.recycler_view)
       recyclerView.layoutManager = LinearLayoutManager(this)

       // Initialize the selected method TextView
       selectedMethodTextView = findViewById(R.id.text_view)

       // Set up the adapter with the list of techniques and a click listener
       val adapter = TechniqueAdapter(techniques) { technique ->
           selectedMethodTextView.text = "${technique.title} Method"
           startButton.isEnabled = true
           startButton.setOnClickListener {
               // Check if the selected technique is in the predefined
techniques list
               if (myTechniques.containsKey(technique.title)) {
                   val steps = myTechniques.getValue(technique.title)
                   val stepDurations = steps.map { it.first to
it.second.times(60).toLong() }

                   val intent = Intent(this, TimerActivity::class.java).apply
{
                       putExtra("techniqueTitle", technique.title)

                       val stepsBundle = Bundle().apply {
                           putSerializable("steps", ArrayList(stepDurations))
                       }
                       putExtras(stepsBundle)
                   }
                   startActivity(intent)



               } else {
                   Toast.makeText(
                       this,
                       "Please select a valid technique. Selected technique:
${technique.title}, Available techniques: ${myTechniques.keys}",
                       Toast.LENGTH_SHORT
                   ).show()
               }
           }
       }

       recyclerView.adapter = adapter
       // Find the start button and disable it by default
       startButton = findViewById(R.id.button_start_studying)
       startButton.isEnabled = false
       selectedMethodTextView.text = "Select a study method"

       val bottomNavigation: BottomNavigationView =
findViewById(R.id.bottomNav)
       bottomNavigation.setOnItemSelectedListener { item ->
           onNavigationItemSelected(item)
       }
   }

   private fun onNavigationItemSelected(item: MenuItem): Boolean{
       when (item.itemId){
           R.id.home ->{
               val intent = Intent(this, MainActivity::class.java)
               startActivity(intent)
           }
           R.id.music ->{
               val intent = Intent(this, MusicPlayer::class.java)
               startActivity(intent)
           }
           R.id.logout ->{
               if(::mediaPlayer.isInitialized) {
                   mediaPlayer.stop()
                   mediaPlayer.release()

               }
               val intent = Intent(this, LoginActivity2::class.java)
               startActivity(intent)
               finish()
           }
```

```
        }
        return false;
    }

}
```

Activity_study.xml

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

   <TextView
       android:id="@+id/text_view"
       android:layout_width="match_parent"
       android:layout_height="wrap_content"
       android:gravity="center_horizontal"
       android:text="Select a study method"
       android:textSize="24sp"
       android:textColor="@android:color/black"
       android:textStyle="bold"/>

   <Button
       android:id="@+id/button_start_studying"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:layout_gravity="center_horizontal"
       android:enabled="false"
       android:text="Start Studying" />

   <androidx.recyclerview.widget.RecyclerView
       android:id="@+id/recycler_view"
       android:layout_width="match_parent"
       android:layout_height="match_parent" />


</LinearLayout>
```

TechniqueAdaptor.kt

```
package com.example.studybuddy20

import android.graphics.Color
import android.util.Log
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.TextView
import androidx.core.content.ContextCompat
import androidx.recyclerview.widget.RecyclerView

class TechniqueAdapter(private val techniques: List<Technique>, private val
onItemClick: (Technique) -> Unit) :
    RecyclerView.Adapter<TechniqueAdapter.TechniqueViewHolder>() {

    private var selectedPosition = RecyclerView.NO_POSITION
    inner class TechniqueViewHolder(itemView: View) :
RecyclerView.ViewHolder(itemView) {
        val techniqueTitle: TextView =
itemView.findViewById(R.id.technique_name)
        val techniqueSteps: TextView =
itemView.findViewById(R.id.technique_steps)
    }

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int):
TechniqueViewHolder {
        val view =
LayoutInflater.from(parent.context).inflate(R.layout.item_techniques, parent,
false)
        return TechniqueViewHolder(view)
    }

    override fun onBindViewHolder(holder: TechniqueViewHolder, position: Int)
{
        val technique = techniques[position]
        holder.techniqueTitle.text = technique.title
        holder.techniqueSteps.text = technique.steps.joinToString(separator =
"\n")

        if (selectedPosition == position) {

holder.itemView.setBackgroundColor(ContextCompat.getColor(holder.itemView.con
text, R.color.transparent_grey))
        } else {
            holder.itemView.setBackgroundColor(Color.TRANSPARENT)
        }

        holder.itemView.setOnClickListener {
            Log.d("TechniqueAdapter", "Selected technique:
${technique.title}")
            onItemClick(technique)

            val previousSelectedPosition = selectedPosition
            selectedPosition = holder.adapterPosition

            if (previousSelectedPosition != RecyclerView.NO_POSITION) {
                notifyItemChanged(previousSelectedPosition)
            }
```

```
                notifyItemChanged(selectedPosition)
            }
        }



    override fun getItemCount(): Int {
        return techniques.size
    }
}
```

activity_account.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:id="@+id/LoginActivity2"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/beige"
    tools:context=".LoginActivity2">

    <Button
        android:id="@+id/create_account_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="44dp"
        android:text="Create Account"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.497"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/login_button" />

    <EditText
        android:id="@+id/userEmailET"
        android:layout_width="295dp"
        android:layout_height="56dp"
        android:layout_marginTop="120dp"
        android:ems="10"
        android:hint="Email"
        android:inputType="textEmailAddress"
        android:padding="16dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/userPasswordET"
        android:layout_width="295dp"
        android:layout_height="59dp"
        android:layout_marginTop="40dp"
        android:ems="10"
        android:hint="Password"
        android:inputType="textPassword"
        android:labelFor="@+id/userPasswordET"
        android:padding="16dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.497"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/userEmailET" />

    <Button
        android:id="@+id/login_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="68dp"
        android:text="Login"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.501"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/userPasswordET" />

    <Button
        android:id="@+id/guestButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="46dp"
        android:text="Guest"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/create_account_button" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="36dp"
        android:layout_marginBottom="36dp"
        android:text="Study Buddy"
        android:textColor="#000000"
        android:textSize="35sp"
        android:textStyle="bold|italic"
        app:layout_constraintBottom_toTopOf="@+id/userEmailET"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.497"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
```

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

Activity_studyconstraint.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"
    android:orientation="vertical"
    android:background="@color/light_green">

    <TextView
        android:id="@+id/text_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center_horizontal"
        android:text="Select a study method"
        android:textColor="@android:color/black"
        android:textSize="24sp"
        android:textStyle="bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/button_start_studying"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_marginStart="120dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="120dp"
        android:enabled="false"
        android:text="Start Studying"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/text_view" />

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recycler_view"
        android:layout_width="406dp"
        android:layout_height="611dp"
        android:layout_marginStart="2dp"
        android:layout_marginTop="84dp"
        android:layout_marginEnd="3dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/text_view" />

    <com.google.android.material.bottomnavigation.BottomNavigationView
        android:id="@+id/bottomNav"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="-55dp"
        android:layout_marginBottom="1dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/recycler_view"
        app:menu="@menu/bottom_nav" />


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

timerTaskActivity.kt

```
package com.example.studybuddy20

import android.content.Intent
import android.media.MediaPlayer
import android.os.Bundle
import android.os.CountDownTimer
import android.view.MenuItem
import android.view.View
import android.widget.*
import androidx.appcompat.app.AppCompatActivity
import com.example.studybuddy20.databinding.ActivityMainBinding
import com.google.android.material.bottomnavigation.BottomNavigationView


class timerTaskActivity : AppCompatActivity() {

    fun createArray(): Array<String> {
        return Array(61) { i -> String.format("%02d", i) }
    }

    private val options = createArray()

    private var selectedItemSec: String = ""
    private var selectedItemMin: String = ""

    private var countdownLengthMin: Long = 0
    private var countdownLengthSec: Long = 0
    private var countdownLength: Long = 0
    private var countdown: CountDownTimer? = null
    private var isPaused = false
    private var remianingTime = 0L

    private lateinit var mediaPlayer : MediaPlayer

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

        val adapter = ArrayAdapter<String>(this,
android.R.layout.simple_spinner_dropdown_item, options)
        findViewById<Spinner>(R.id.setTimerMin).adapter = adapter
        findViewById<Spinner>(R.id.setTimerSec).adapter = adapter

        findViewById<Spinner>(R.id.setTimerMin).setOnItemSelectedListener(
            object : AdapterView.OnItemSelectedListener {
                override fun onItemSelected(
                    parent: AdapterView<*>,
                    view: View?,
                    position: Int,
                    id: Long
                ) {
                    selectedItemMin = ""
                    selectedItemMin =
parent.getItemAtPosition(position).toString()
                    val selectedItemMinLong = selectedItemMin.toLong()

                    countdownLengthMin = selectedItemMinLong * 60 * 1000
                    findViewById<TextView>(R.id.txtMinTV).text =
selectedItemMin
                }
                override fun onNothingSelected(parent: AdapterView<*>?) {
                    // Do nothing
                }
            })

        findViewById<Spinner>(R.id.setTimerSec).setOnItemSelectedListener(
            object : AdapterView.OnItemSelectedListener {
                override fun onItemSelected(
                    parent: AdapterView<*>,
                    view: View?,
                    position: Int,
                    id: Long
                ) {
                    selectedItemSec = ""
                    selectedItemSec =
parent.getItemAtPosition(position).toString()
                    val selectedItemSecLong = selectedItemSec.toLong()
```

```
                   countdownLengthSec = selectedItemSecLong * 1000
                   findViewById<TextView>(R.id.txtSecTV).text =
selectedItemSec
               }

               override fun onNothingSelected(parent: AdapterView<*>?) {
                   // Do nothing
               }
           })


       findViewById<Button>(R.id.btnStart).setOnClickListener {
           startCountdown()
       }

       findViewById<Button>(R.id.btnPause).setOnClickListener {
           isPaused = true
           countdown?.cancel()
           findViewById<Button>(R.id.btnStart).isEnabled = true
       }

       findViewById<Button>(R.id.btnRepeat).setOnClickListener {
           countdown?.cancel()
           findViewById<Button>(R.id.btnStart).isEnabled = true
           countdownLength = countdownLengthMin + countdownLengthSec
           remianingTime = countdownLength
           findViewById<TextView>(R.id.txtMinTV).text = selectedItemMin
           findViewById<TextView>(R.id.txtSecTV).text = selectedItemSec
       }

       val bottomNavigation: BottomNavigationView =
findViewById(R.id.bottomNav)
       bottomNavigation.setOnItemSelectedListener { item ->
           onNavigationItemSelected(item)
       }
   }

   private fun startCountdown() {
       findViewById<Button>(R.id.btnStart).isEnabled = false
       countdown?.cancel()
       countdownLength = countdownLengthMin + countdownLengthSec

       val TimeRemaining = if (isPaused) remianingTime else countdownLength

       countdown = object : CountDownTimer(TimeRemaining, 1000) {
           override fun onTick(millisUntilFinished: Long) {
               remianingTime = millisUntilFinished
               isPaused = false
```

```kotlin
                val minutes = millisUntilFinished / 1000 / 60
                val seconds = (millisUntilFinished / 1000) % 60

                val formattedTimeMin = String.format("%02d", minutes)
                val formattedTimeSec = String.format("%02d", seconds)
                findViewById<TextView>(R.id.txtMinTV).text = formattedTimeMin
                findViewById<TextView>(R.id.txtSecTV).text = formattedTimeSec
            }

            override fun onFinish() {
                findViewById<TextView>(R.id.txtMinTV).text = "00"
                findViewById<TextView>(R.id.txtSecTV).text = "00"
                findViewById<Button>(R.id.btnStart).isEnabled = true
            }
        }.start()
    }

    fun onClick(view: View?) {
        when (view?.id) {
            R.id.home ->{
                val intent = Intent(this, MainActivity::class.java)
                startActivity(intent)
            }
        }
    }

    private fun onNavigationItemSelected(item: MenuItem): Boolean{
        when (item.itemId){
            R.id.home ->{
                val intent = Intent(this, MainActivity::class.java)
                startActivity(intent)
            }
            R.id.music ->{
                val intent = Intent(this, MusicPlayer::class.java)
                startActivity(intent)
            }
            R.id.logout ->{
                if(::mediaPlayer.isInitialized) {
                    mediaPlayer.stop()
                    mediaPlayer.release()

                }
                val intent = Intent(this, LoginActivity2::class.java)
                startActivity(intent)
                finish()
            }
        }
        return false;
    }
```

```
}
```

Technique.kt

```kotlin
package com.example.studybuddy20

import android.os.Parcel
import android.os.Parcelable

data class Technique(val title: String, val steps: List<String>) : Parcelable
{
   constructor(parcel: Parcel) : this(
       parcel.readString()!!,
       parcel.createStringArrayList()!!
   )

   override fun writeToParcel(parcel: Parcel, flags: Int) {
       parcel.writeString(title)
       parcel.writeStringList(steps)
   }

   override fun describeContents(): Int {
       return 0
   }

   companion object CREATOR : Parcelable.Creator<Technique> {
       override fun createFromParcel(parcel: Parcel): Technique {
           return Technique(parcel)
       }

       override fun newArray(size: Int): Array<Technique?> {
           return arrayOfNulls(size)
       }
   }
}
```

Item_techniques.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<!-- item_technique.xml -->
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:orientation="vertical"
   android:padding="16dp">

   <TextView
       android:id="@+id/technique_name"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:textAppearance="?android:textAppearanceMedium"
       android:textColor="@android:color/black"
```

```
        android:textStyle="bold" />

    <TextView
        android:id="@+id/technique_steps"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:textAppearanceSmall"
        android:textColor="@android:color/black" />

</LinearLayout>
```

MusicPLayer.kt

```
package com.example.studybuddy20

import android.annotation.SuppressLint
import android.content.Intent
import android.media.MediaPlayer
import android.os.Bundle
import android.view.MenuItem
import android.view.View
import android.widget.AdapterView
import android.widget.ArrayAdapter
import android.widget.Spinner
import androidx.appcompat.app.AppCompatActivity
import com.google.android.material.bottomnavigation.BottomNavigationView

class MusicPlayer : AppCompatActivity() {
   private lateinit var mediaPlayer: MediaPlayer
   private val mp3Files = arrayOf(R.raw.lofi, R.raw.jazz)
   private var selectedFile: Int? = null
   private var playbackPos = 0
   private var isPaused = false

   override fun onCreate(savedInstanceState: Bundle?) {
       super.onCreate(savedInstanceState)
       setContentView(R.layout.music_player)
       val mp3Spinner: Spinner = findViewById(R.id.mp3_spinner)

       ArrayAdapter.createFromResource(
           this,
           R.array.mp3_files,
           android.R.layout.simple_spinner_item
       ).also { adapter ->

adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item
)
           mp3Spinner.adapter = adapter
       }
```

```
        mp3Spinner.onItemSelectedListener = object :
AdapterView.OnItemSelectedListener {
            @SuppressLint("DiscouragedApi")
            override fun onItemSelected(p0: AdapterView<*>?, p1: View?,
position: Int, p3: Long) {
                selectedFile =
resources.getIdentifier(mp3Files[position].toString(), "raw", packageName)

            }

            override fun onNothingSelected(p0: AdapterView<*>?) {
                //does nothing
            }

        }

        val bottomNavigation: BottomNavigationView =
findViewById(R.id.bottomNav)
        bottomNavigation.setOnItemSelectedListener { item ->
            onNavigationItemSelected(item)
        }
    }


    fun play(view: View){
        if(!::mediaPlayer.isInitialized){
            mediaPlayer = MediaPlayer.create(this, selectedFile!!)
            mediaPlayer.start()
        }
        else {
            try {
                mediaPlayer.seekTo(playbackPos)
                mediaPlayer.start()
            } catch (e: IllegalStateException) {

                mediaPlayer = MediaPlayer.create(this, selectedFile!!)
                mediaPlayer.start()
            }
        }


    }
    fun pause(view: View){
        if (::mediaPlayer.isInitialized && mediaPlayer.isPlaying){
            mediaPlayer.pause()
            playbackPos = mediaPlayer.currentPosition
            isPaused = true
        }
    }
    fun stop(view: View){
        if (::mediaPlayer.isInitialized){
            mediaPlayer.stop()
            mediaPlayer.release()
            playbackPos = 0
            isPaused = false
        }
        else {
            //do nothing
        }
    }



    private fun onNavigationItemSelected(item: MenuItem): Boolean{
        when (item.itemId){
            R.id.home ->{
                val intent = Intent(this, MainActivity::class.java)
                startActivity(intent)
            }
            R.id.music -> {
                return true
            }
            R.id.logout ->{
                if(::mediaPlayer.isInitialized) {
                    mediaPlayer.stop()
                    mediaPlayer.release()
                    playbackPos = 0
                    isPaused = false
                }
                val intent = Intent(this, LoginActivity2::class.java)
                startActivity(intent)
                finish()
            }
        }
        return false
    }
}
```

Music_player.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">


    <Spinner
```

```
        android:id="@+id/mp3_spinner"
        android:layout_width="233dp"
        android:layout_height="51dp"
        android:layout_margin="16dp"
        android:prompt="@string/select_mp3_prompt"
        app:layout_constraintBottom_toBottomOf="@+id/bottomNav"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.502"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.259"
        tools:ignore="MissingConstraints" />

    <Button
        android:id="@+id/playMusicBtn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="play"
        android:text="Play"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.413" />

    <Button
        android:id="@+id/pauseMusicBtn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="pause"
        android:text="Pause"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.499" />

    <Button
        android:id="@+id/stopMusicBtn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="stop"
        android:text="Stop"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.501"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.59" />
```

```
   <com.google.android.material.bottomnavigation.BottomNavigationView
       android:id="@+id/bottomNav"
       android:layout_width="0dp"
       android:layout_height="wrap_content"
       android:layout_marginTop="95dp"
       android:layout_marginEnd="10dp"
       android:layout_marginBottom="1dp"
       app:layout_constraintBottom_toBottomOf="parent"
       app:layout_constraintEnd_toEndOf="parent"
       app:layout_constraintStart_toStartOf="parent"
       app:menu="@menu/bottom_nav" />


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

Activity_timer_task.xml

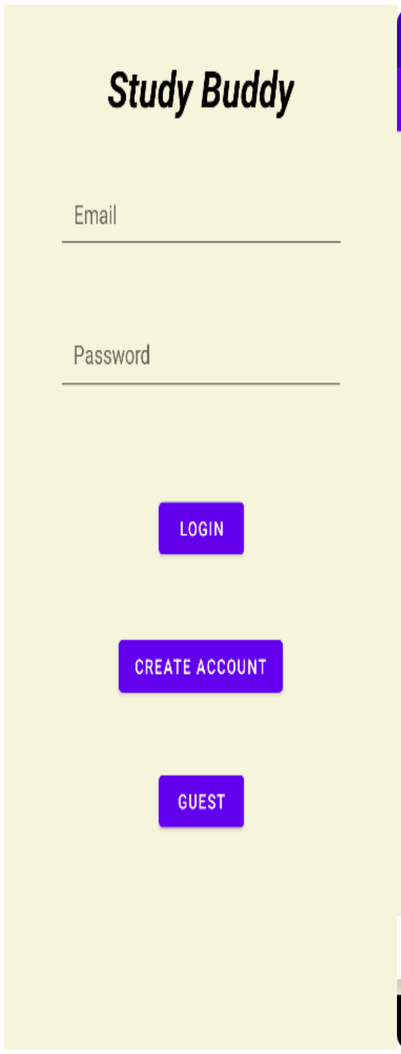
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".timerTaskActivity"
   android:background="@drawable/img_2">

   <TextView
       android:id="@+id/txtTimerTV"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:layout_marginTop="60dp"
       android:gravity="center"
       android:text=":"
       android:textAlignment="center"
       android:textAppearance="?android:attr/textAppearanceLarge"
       android:textSize="40sp"
       app:layout_constraintEnd_toEndOf="parent"
       app:layout_constraintStart_toStartOf="parent"
       app:layout_constraintTop_toTopOf="parent" />

   <Spinner
       android:id="@+id/setTimerSec"
       android:layout_width="wrap_content"
       android:layout_height="wrap_content"
       android:layout_marginStart="31dp"
       android:layout_marginTop="196dp"
       android:layout_marginEnd="130dp"
       android:minWidth="10dp"
```

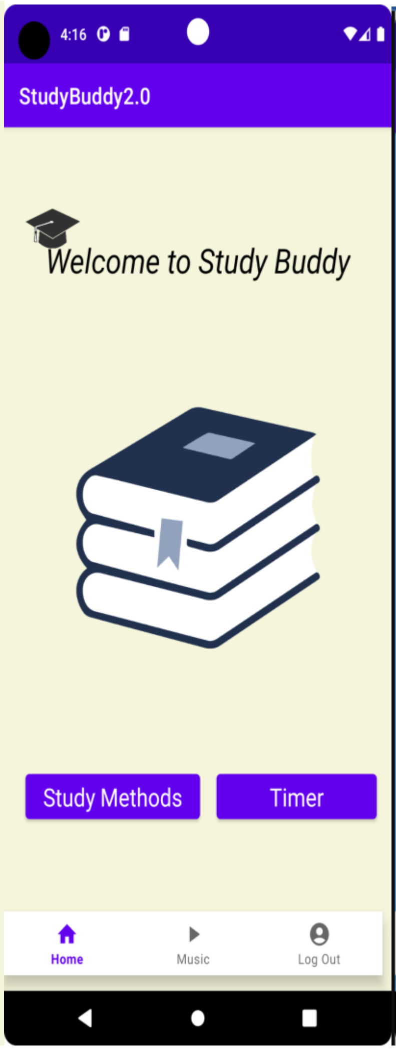
```
        android:minHeight="35dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toEndOf="@+id/txtSec"
        app:layout_constraintTop_toTopOf="parent"
        tools:ignore="SpeakableTextPresentCheck,TouchTargetSizeCheck" />

    <TextView
        android:id="@+id/txtMin"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="120dp"
        android:layout_marginTop="53dp"
        android:layout_marginEnd="10dp"
        android:text="Minutes:"
        app:layout_constraintEnd_toStartOf="@+id/setTimerSec"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/txtTimerTV" />

    <TextView
        android:id="@+id/txtTTF"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="280dp"
        android:textAlignment="center"
        android:text="Time To Focus!"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:textColor="@color/black"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Spinner
        android:id="@+id/setTimerMin"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="35dp"
        android:layout_marginTop="160dp"
        android:layout_marginEnd="130dp"
        android:layout_marginBottom="1dp"
        android:minWidth="10dp"
        android:minHeight="35dp"
        app:layout_constraintBottom_toTopOf="@+id/setTimerSec"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toEndOf="@+id/txtMin"
        app:layout_constraintTop_toTopOf="parent"
        tools:ignore="SpeakableTextPresentCheck,TouchTargetSizeCheck" />

    <Button
```

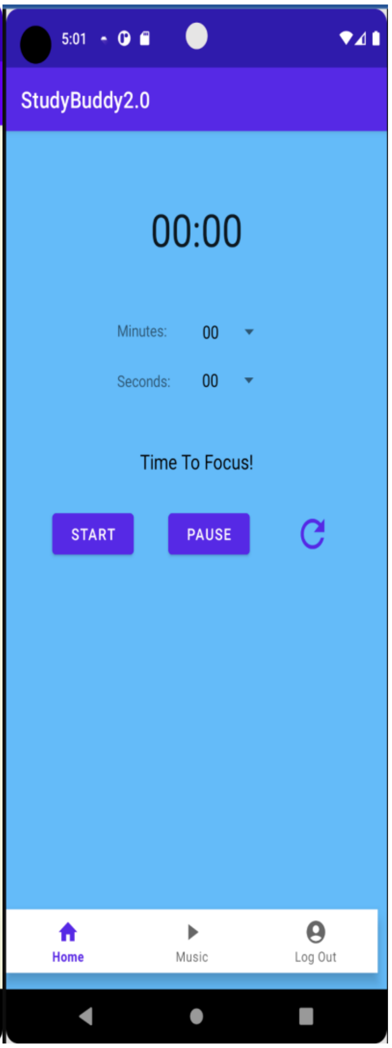
```xml
        android:id="@+id/btnStart"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="50dp"
        android:layout_marginTop="332dp"
        android:text="Start"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/txtSec"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="120dp"
        android:layout_marginTop="25dp"
        android:layout_marginEnd="13dp"
        android:text="Seconds:"
        app:layout_constraintEnd_toStartOf="@+id/setTimerMin"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/txtMin" />

    <Button
        android:id="@+id/btnRepeat"
        android:layout_width="61dp"
        android:layout_height="64dp"
        android:layout_marginTop="324dp"
        android:layout_marginEnd="50dp"
        android:background="@drawable/restart_foreground"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        tools:ignore="SpeakableTextPresentCheck" />

    <TextView
        android:id="@+id/txtMinTV"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="60dp"
        android:text="00"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:textSize="40sp"
        android:gravity="left"
        app:layout_constraintEnd_toStartOf="@+id/txtTimerTV"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/txtSecTV"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="60dp"
        android:text="00"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:textSize="40sp"
        android:gravity="right"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toEndOf="@+id/txtTimerTV"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/btnPause"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="332dp"
        android:text="Pause"
        app:layout_constraintEnd_toStartOf="@+id/btnRepeat"
        app:layout_constraintStart_toEndOf="@+id/btnStart"
        app:layout_constraintTop_toTopOf="parent" />

    <com.google.android.material.bottomnavigation.BottomNavigationView
        android:id="@+id/bottomNav"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="286dp"
        android:layout_marginEnd="10dp"
        android:layout_marginBottom="1dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/btnRepeat"
        app:menu="@menu/bottom_nav"/>

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

Output screenshot for the app:

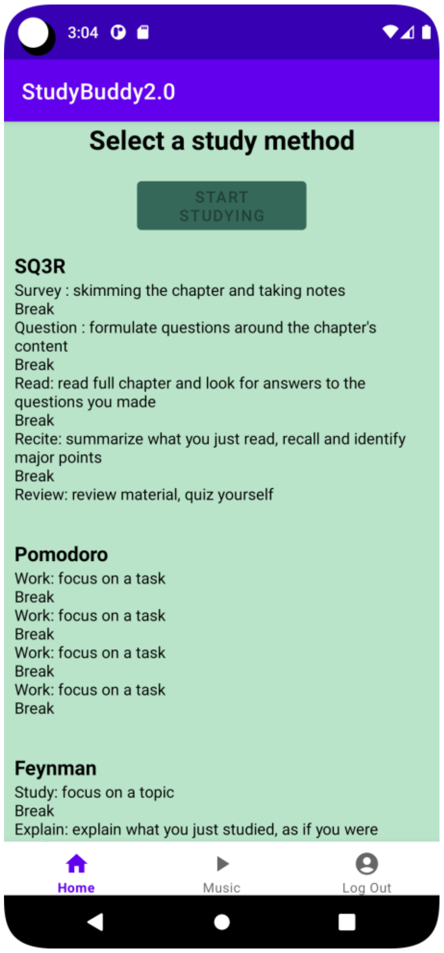
Front Page: Second Page Timer Page .







Study Methods Page:



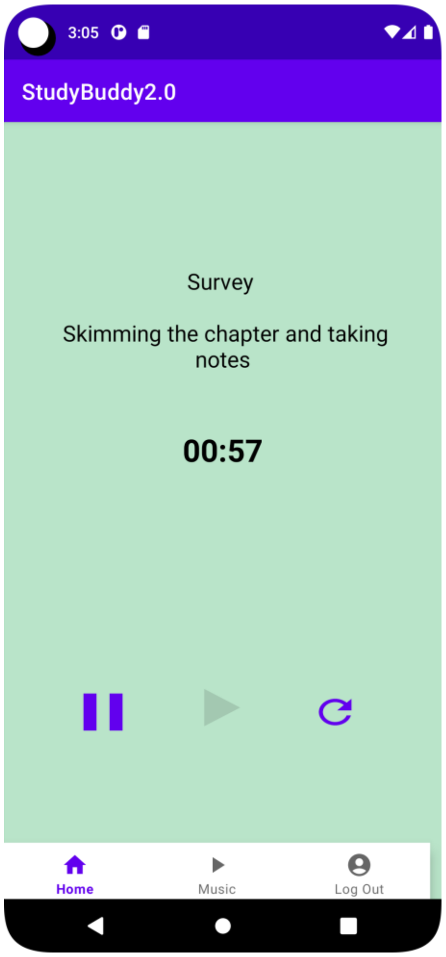
StudyBuddy2.0

Select a study method

SQ3R
Survey : skimming the chapter and taking notes
Break
Question : formulate questions around the chapter's content
Break
Read: read full chapter and look for answers to the questions you made
Break
Recite: summarize what you just read, recall and identify major points
Break
Review: review material, quiz yourself

Pomodoro
Work: focus on a task
Break
Work: focus on a task
Break
Work: focus on a task
Break
Work: focus on a task
Break

Feynman
Study: focus on a topic
Break
Explain: explain what you just studied, as if you were



StudyBuddy2.0

Survey

Skimming the chapter and taking notes

00:57