BigMathDocumentation

Sharang Deshpande

Abstract

This document comprises the details of an application titled BigMath to implement better standards to develop an individual's teaching and learning skills. The statement includes a literature review containing 7 case studies on mobile applications that involve any user with Mathematics based puzzles. From the review, it can be concluded that most applications in this area fail to provide the user with a good gameplay experience due to missing in-demand features like a good GUI and various puzzles. The better-performing applications fail to create a ‘learning’ experience and thus fall out somehow. The application BigMath would be developed to provide the target audience, i.e. end-users like teachers and students, with a fun, modern and interactive way to learn mathematics. The methodology of the document states how implementation of 7-8 exciting features between 2 user classes can practically help achieve the application’s objectives. The defined data flow and the decision tree models show how certain parts of the application will function. The coding section presents the logic behind how the core parts of the application are programmed. It goes over concepts like GestureListeners and Timers. The UI/UX section contains diagrams modelling the application's user interface. The testing section comprises various tests carried out on the application to determine performance.

Introduction

We aim to develop an interactive application titled 'BigMath,' which includes a series of puzzle modules based on a difficulty chosen by the end user. A multiplayer option to connect between devices and play the puzzles will be incorporated for competitive play. Multiple features like region-based ranking, level-based achievements, profile creation and events can be incorporated to make the application more stirring.

The business objective of the project BigMath is to provide the audience with a fun and interactive way to learn mathematics. The application targets parents and teachers who want their children with the goal of improving mathematical skills while making it more accessible and enjoyable for them.

The application should have many daily active users, indicating that children are engaged and using the application regularly. It should receive positive feedback, indicating that the games are engaging and effective in teaching mathematical concepts. In the future, the application may generate revenue through subscriptions or advertising. It should see high social sharing levels directly affecting reputation and popularity.

BigMath aims to provide an engaging learning experience for youngsters to help them build arithmetic abilities and confidence. The vision is to increase productivity by deploying interactive game-based learning approaches that intrigue and motivate them. The application aims to deliver a practical learning experience covering fundamental arithmetic ideas and themes. It seeks to instil a love of mathematics in youngsters and build a growth mentality by challenging them to think creatively and solve problems in novel ways.

The scope of BigMath has been divided into 2 parts.

The acceptance criterion establishes what needs to be achieved in the current scope.

The Project exclusions contain the part(s) that cannot or need not be achieved in the current project states.

Acceptance Criterion:

The game should be age-friendly.

The code should be secure and well-formatted, preferably designed object-oriented.

The game can comprise a multiplayer function in later versions.

The game should be equipped with enough accessibility features.

The application should have embedded facilities to support users’ issues.

Literature Review

Introduction

This is a literature review done on Math Learning apps and games. Below are seven applications developed to run on the android operating system.

The literature review has been written on the key points listed below:

- Size of the application on the system.
- CPU Usage of the application after every cycle.
- Appearance and colours of the GUI.
- Notable features in the application.
- Notable drawbacks in the application.

To keep the environment consistent, the applications were tested on one single platform with the following specifications:

- Model: Redmi Note 9 Pro Max
- Android version: 12 SKQ1.211019.001
- RAM: 6 GB
- CPU: Octa-core Max 2.32GHz

Case 1: Math Games: Math for Kids

Developer:

RV AppStudios

Required Minimum Android version:

4.4

Application Size:

32-70 MB

Average CPU usage:

128 MB per game

Appearance:

The theme is light, and the colours are bold.

The application has implemented multiple bright colours to create the app attractive for kids.

Observed Features:

Multiple difficulty levels depending on the ‘Grade’ selected.

More than 20 different activities for almost every mathematical operation.

Covers the syllabus till 7th grade.

Involves topics like fractions, decimals and square roots.

A challenging gameplay experience.

After analysing the tasks the user completes, a report card is provided.

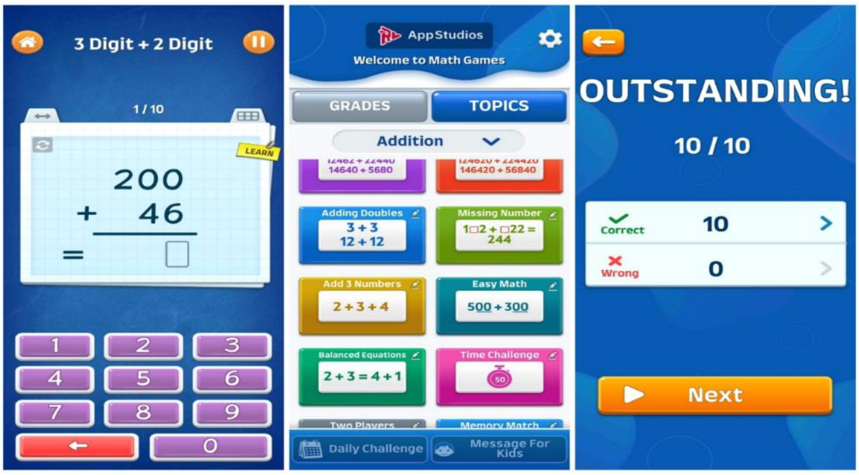
The game performs offline.

Observed Drawbacks:

No multiplayer feature.

Repetitive questions.

Screenshots:



Case 2: Math Kids: Math Games for Kids

Developer:

RV AppStudios

Required Minimum Android version:

4.4

Application Size:

25-60 MB

Average CPU usage:

250 MB per game

Appearance:

The app is very colourful and cartoony.

A set theme with animated cartoons intrigues the user.

Observed Features:

Provides hints if questions go wrong frequently.

Examples and images are based on real life.

‘Counting and compare’ operations have been incorporated.

It contains features that help a parent monitor their child’s progress.

The game performs offline.

Observed Drawbacks:

It can only be played at a toddler’s level.

No convenient way of checking the logged progress.

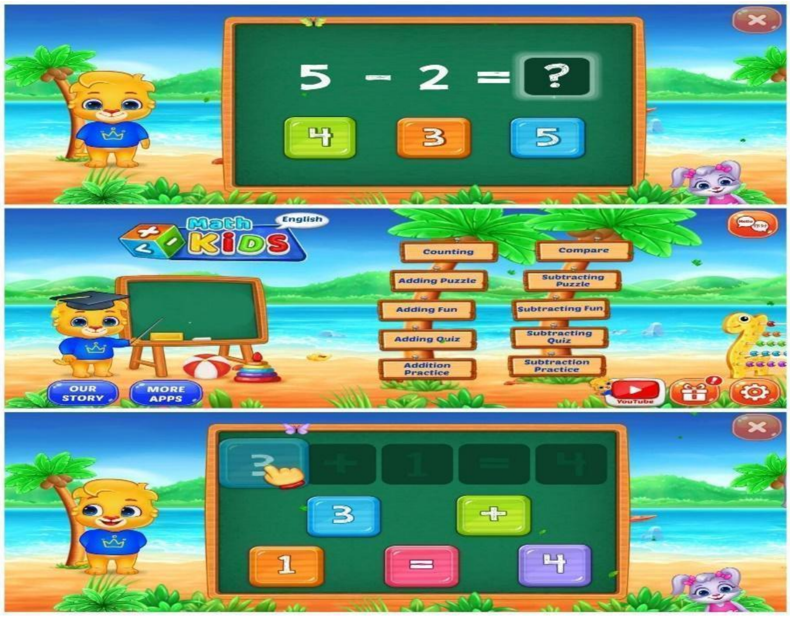
No multiplayer.

The cartoons lag a lot.

CPU-heavy gameplays.

Repetitive questions.

Screenshots:



Case 3: Math Games, Learn Add Multiply

Developer:

GunjanApps Studios

Required Minimum Android version:

4.4

Application Size:

20-40 MB

Average CPU usage:

117 MB per game

Appearance:

Light theme with primary colours. No shadings or animations have been incorporated into the app.

Observed Features:

Multiplayer mode with a split screen.

Extra puzzles like sudoku.

The game is in the form of worksheets.

Additional functions like squares and exponents for exercises.

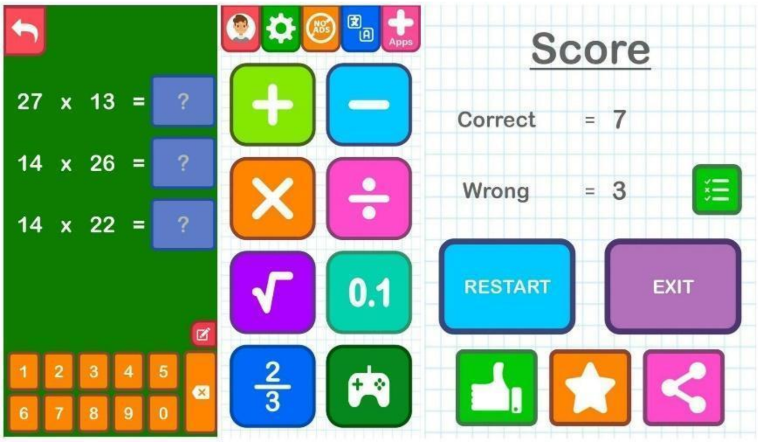
Multiplayer allows users to compete globally in the ‘World Challenge’.

Observed Drawbacks:

Heavy lag on the system.

Advertisements that cause many interruptions mid-gameplay.

Screenshots



27 x 13 = ?

14 x 26 = ?

14 x 22 = ?

Score

Correct = 7

Wrong = 3

Case 4: Math Games

Developer:

Godline Studios

Required Minimum Android version:

4.4

Application Size:

20-40 MB

Average CPU usage:

117 MB per game

Appearance:

Light theme, primarily light blue and white. No shadings or animations have been incorporated into the app.

Observed Features:

Multiplayer mode and a global ranking system.

Performance-based achievements that use ‘coins’ as a unit.

The user can use these coins to unlock other modes in the game.

The game implements a model based on worksheets with multiple additional features like the famous 2048 puzzle.

Multiplayer allows users to compete globally in the ‘World Challenge’.

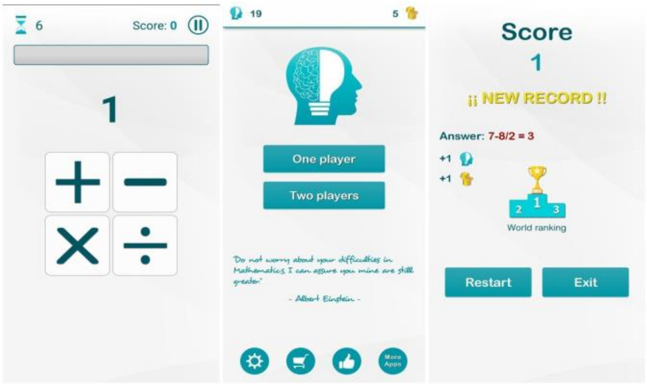
Observed Drawbacks:

These are fundamental problems.

Not a learning-oriented app but a stress buster.

Not an attractive GUI.

Screenshots:



"Do not worry about your difficulties in Mathematics, I can assure you mine are still greater"

- Albert Einstein -

Answer: 7-8/2 = 3

Case 5: Math Games: to Learn Math

Developer:

Jigar Education Hub

Required Minimum Android version:

4.4

Application Size:

10-30 MB

Average CPU usage:

220 MB per game

Appearance:

It only uses two colours, dark blue and white.

Buttons are not appropriately aligned.

Observed Features:

A region-based ranking system that displays the top 10 players in every difficulty option.

A practice mode before attempting the puzzles.

Split screen interface for multiplayer.

Observed Drawbacks:

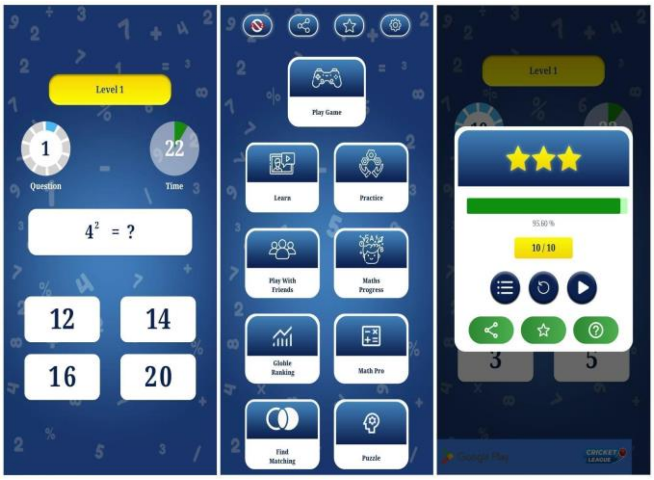
These are fundamental problems.

Not a learning-oriented app.

Not an attractive GUI.

Many bugs, especially with multiplayer.

Screenshots:



Case 6: Mathematics

Developer:

NixGame

Required Minimum Android version:

4.4

Application Size:

10-20 MB

Average CPU usage:

160 MB per game

Appearance:

Light mode with primary colours.

Observed Features:

It has a unique feature called speed calculation to train mental arithmetic.

It has a manual feature to help with the application features.

A very well-designed evaluation and progress report system.

A simple and unique multiplication table feature.

Observed Drawbacks:

Not a learning-oriented app.

No progress reports or well-displayed results.

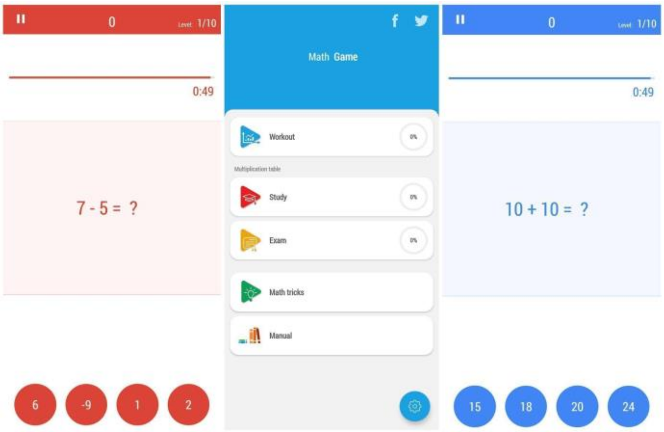
Not an attractive GUI.

The GUI is stressful on the eyes.

No multiplayer.

Not enough content.

Screenshots:



Case 7: Math Games - Brain Training

Developer:

Pavel Olegovich

Required Minimum Android version:

4.4

Application Size:

20-40 MB

Average CPU usage:

160 MB per game

Appearance:

The UI is only Blue and white.

Observed Features:

The app benefits adults who like to solve intriguing puzzles and problems.

The game covers almost the entire range of arithmetic exercises.

The app comprises 20+ exciting math mind games like:

Math riddles, Logic math games, Multiplication games, and other math logic puzzles.

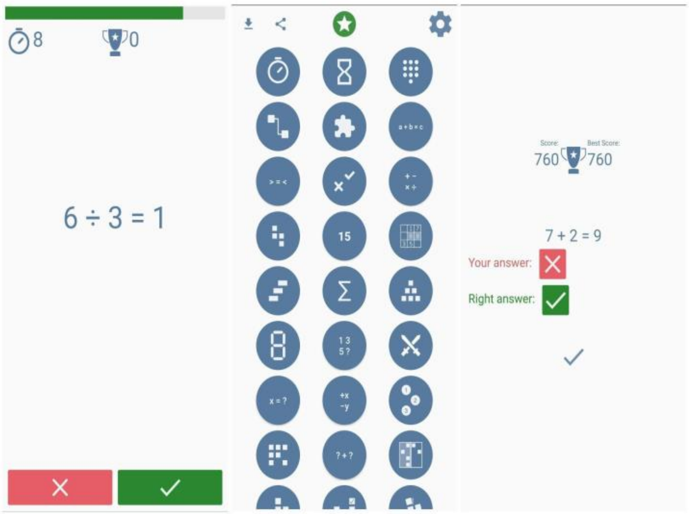
Observed Drawbacks:

No guidance for a poorly designed GUI.

This makes the application very restricting.

It is also not kid-friendly.

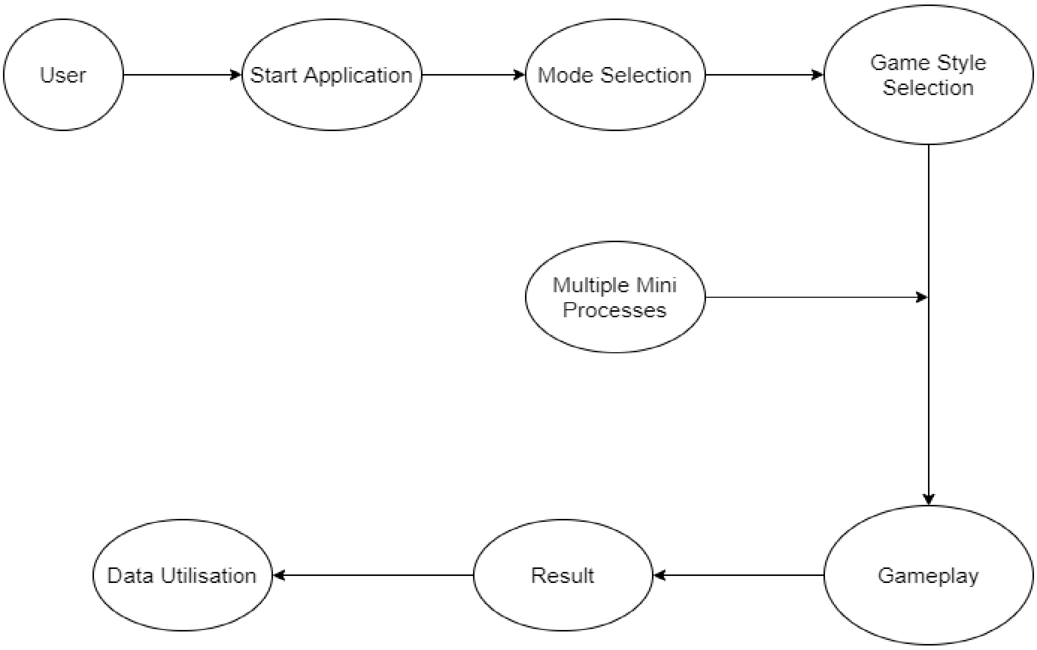
Screenshots:



Methodology

Product Perspective:

The product perspective diagram below shows a simple flow of the production processes that involve the user selecting the type of game-based learning experience they desire. The chart expands further in the document based on the mode the end user picks.



User classes and characteristics:

Admin-Class: Admin-class will include configuring and developing the application. It will consist of access to adding, updating, and deleting the functionalities involved in the system. The class also provides access to user records. The admin class is mainly comprised of Project Managers and Leads.

User-Class: User-class will include the end users who will utilise the developed application for their benefit. The user class is divided into two main categories:

Primary User class

The primary user class comprises the students or learners directly under the application's influence.

Secondary User class

The secondary class users involve entities like teachers, parents and guardians who use the application for the primary class.

Operating Environment:

The application can be run in an android-based environment with API level 26 or above.

Design and Implementation Constraints:

A flexible design with the most features implemented statically should be followed.

Assumptions and Dependencies

The following assumptions and dependencies can be noted for the application:

- The user is well-versed in English.
- The system has a stable internet connection when accessing updates and multiplayer.
- The user is of an appropriate age to access the application.
- The hardware involved meets the application requirements.
- The end user is not wholly new to the use of the application.
- The OS is Android 7 or higher.
- The device running the application has at least 4GB ram.

System Features:

Theme customisations

The user can set the application appearance to their preference from an available set of options.

Basic Math marathons:

The most straightforward set of mini-games based on mathematical operations like addition and subtraction.

Sudden Death Mode:

The mini-game set is based on a category of mathematical operations that terminate when the user enters an incorrect input.

Rapid fire mode

The set of mini-games is to be finished in a minimal amount of time and only terminate or display results at the end of the session.

Local Multiplayer / Wi-fi based Multiplayer

Helpful mathematics-related content

Helpful third-party content which the user can refer to.

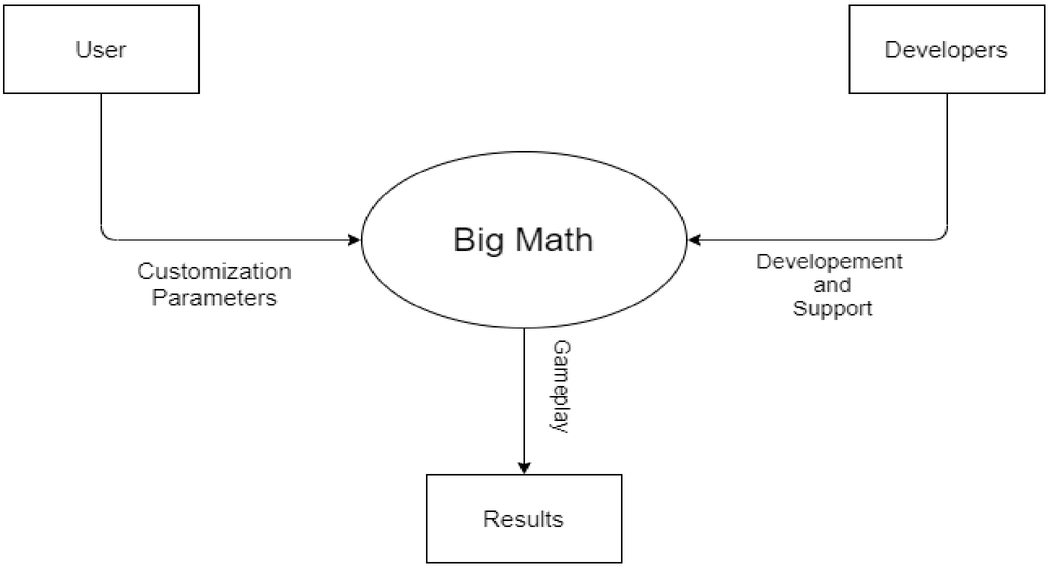
Timely reports

The user is provided with a timely analysis of their progress within the app.

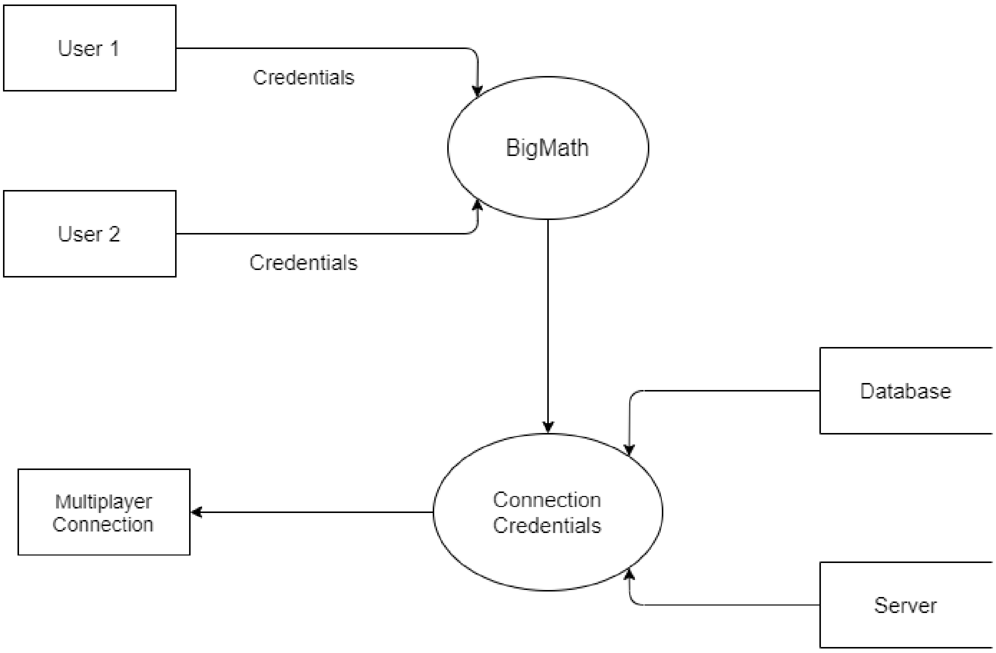
Analysis Models:

Data flow models-

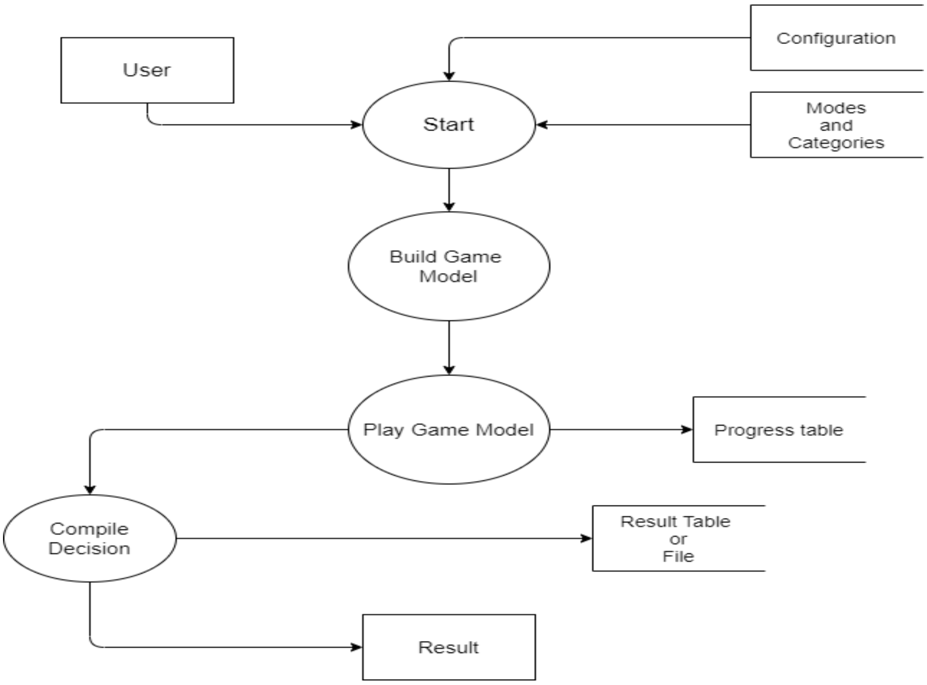
BigMath Zero Level:



Multiplayer Zero Level:

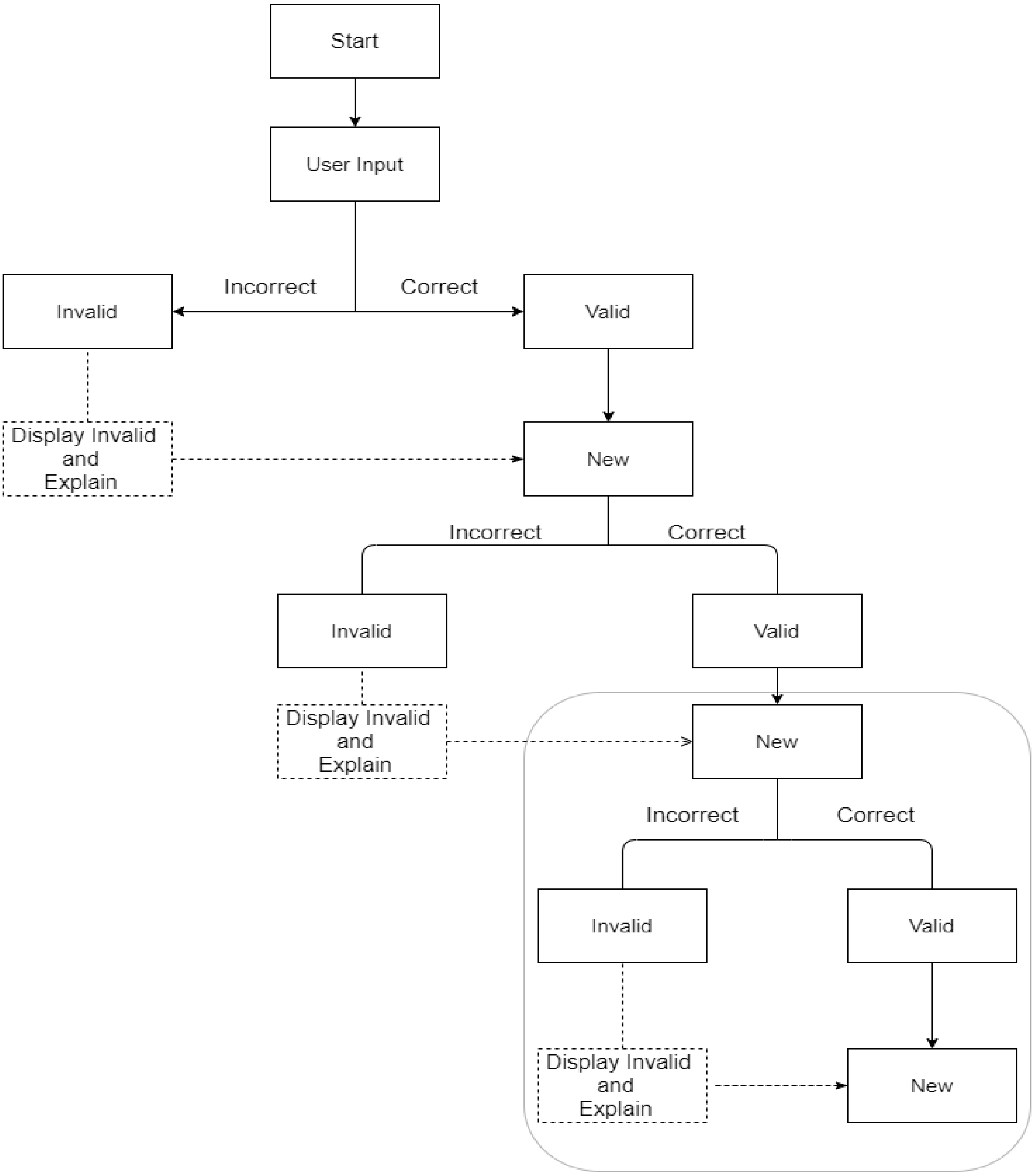


BigMath One Level:

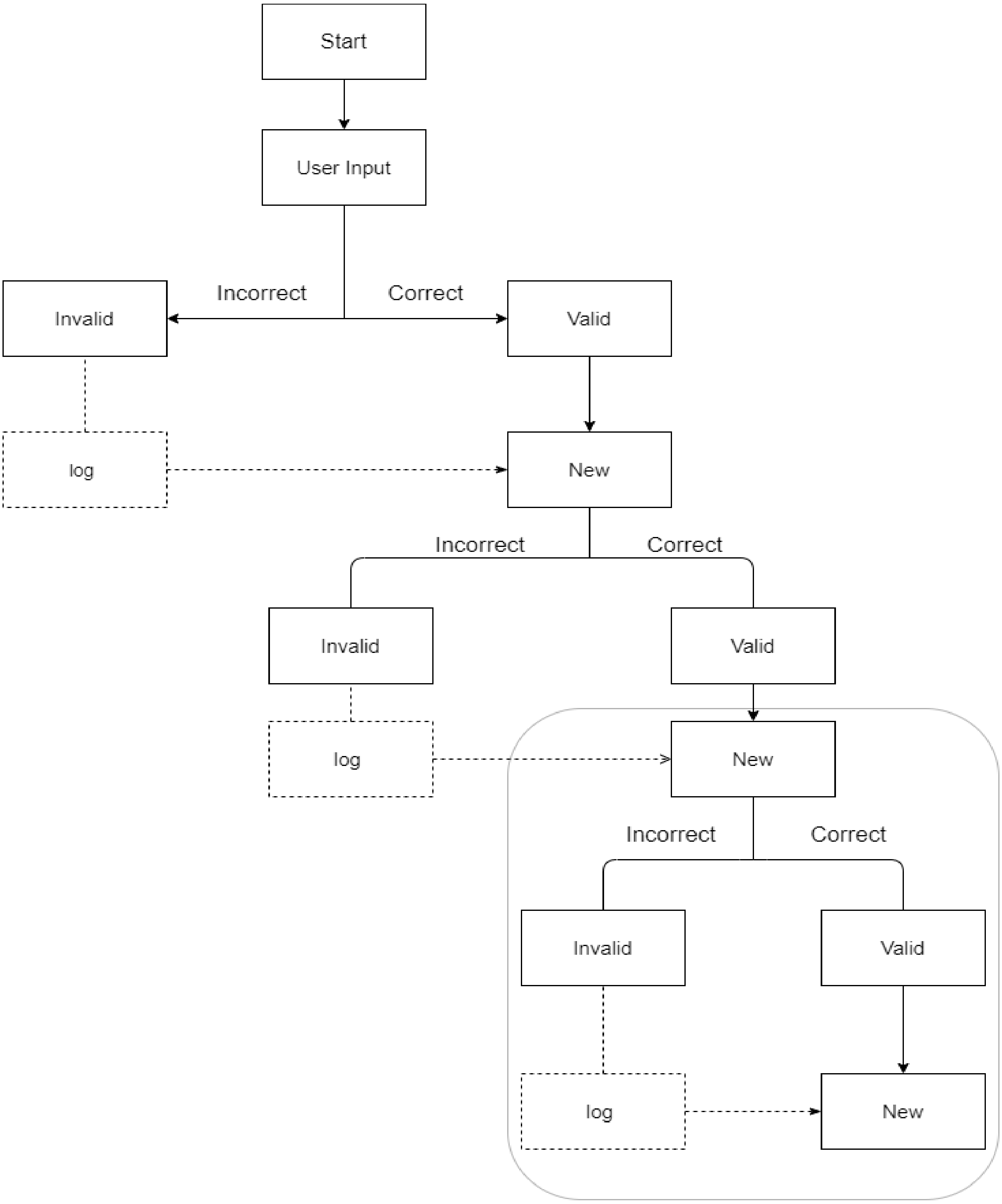


Decision trees

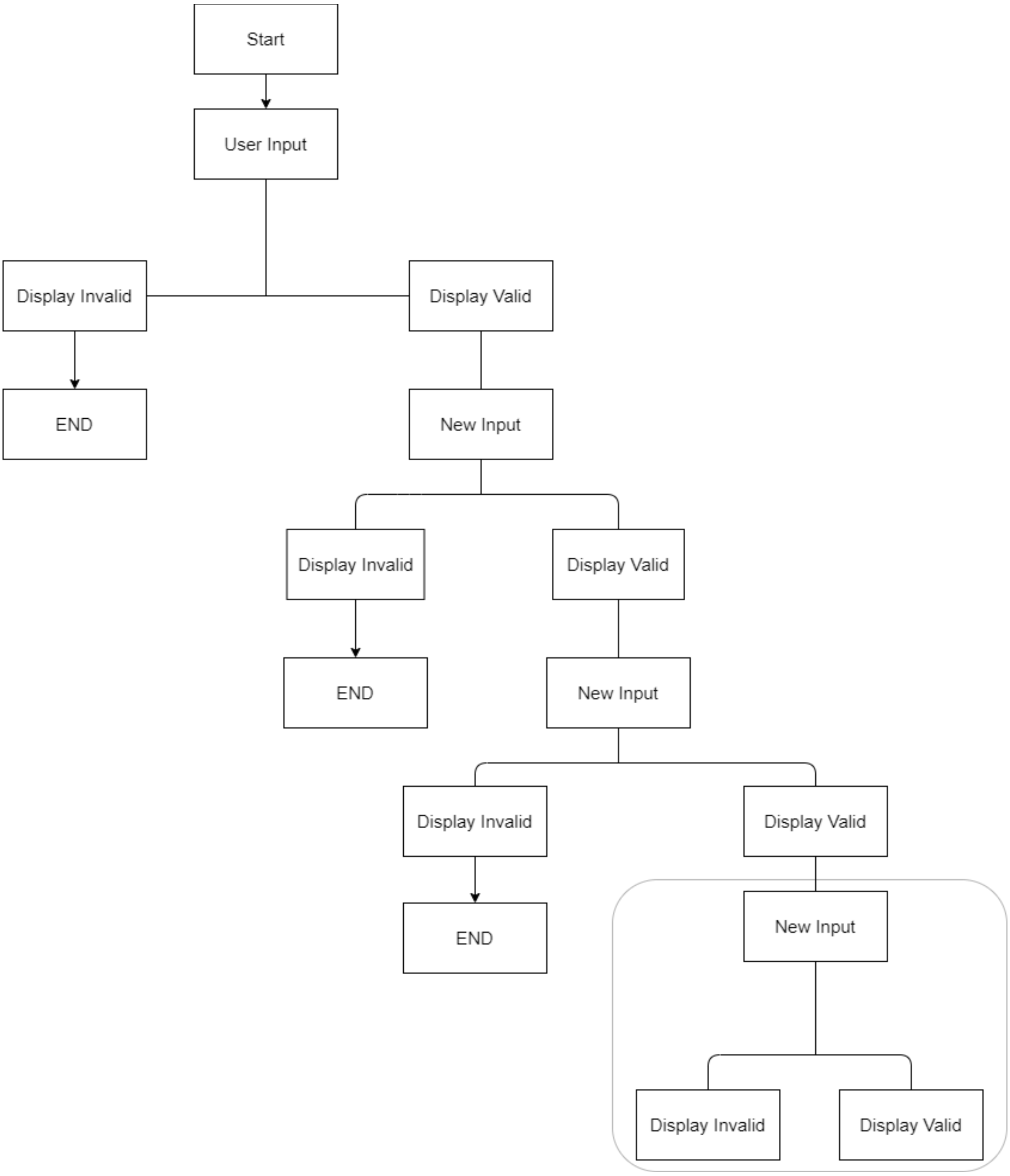
Base Features:



Rapid Fire Feature:

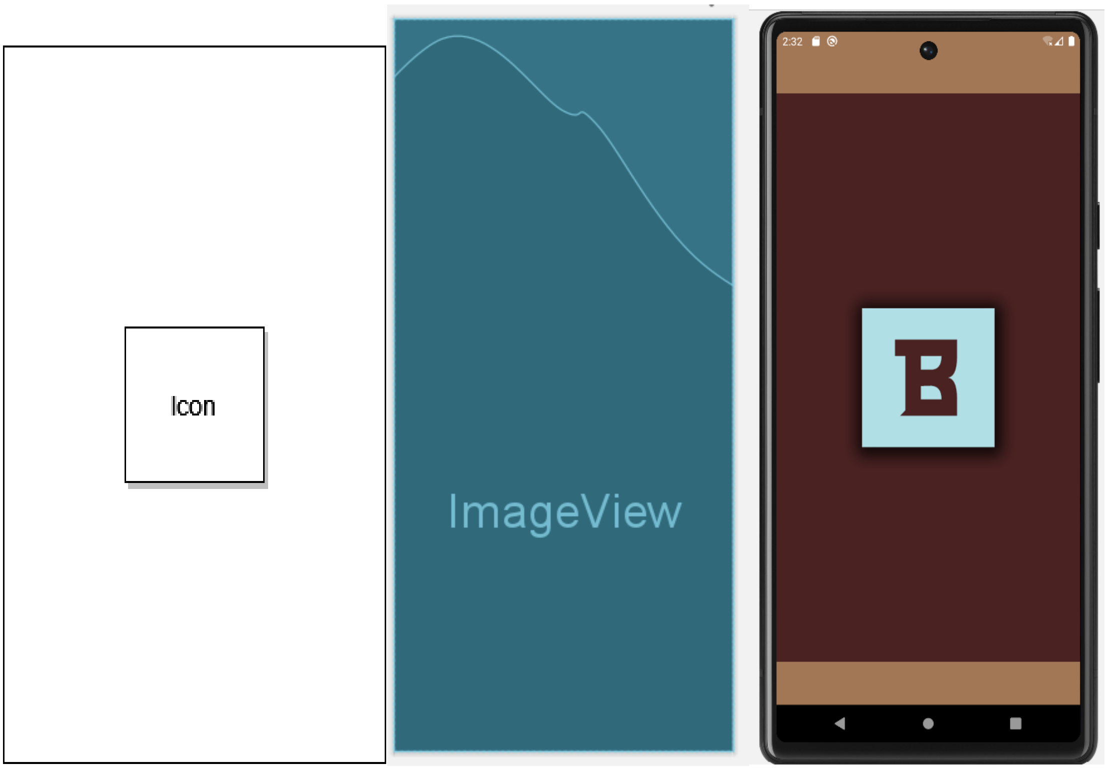


Sudden Death Feature:

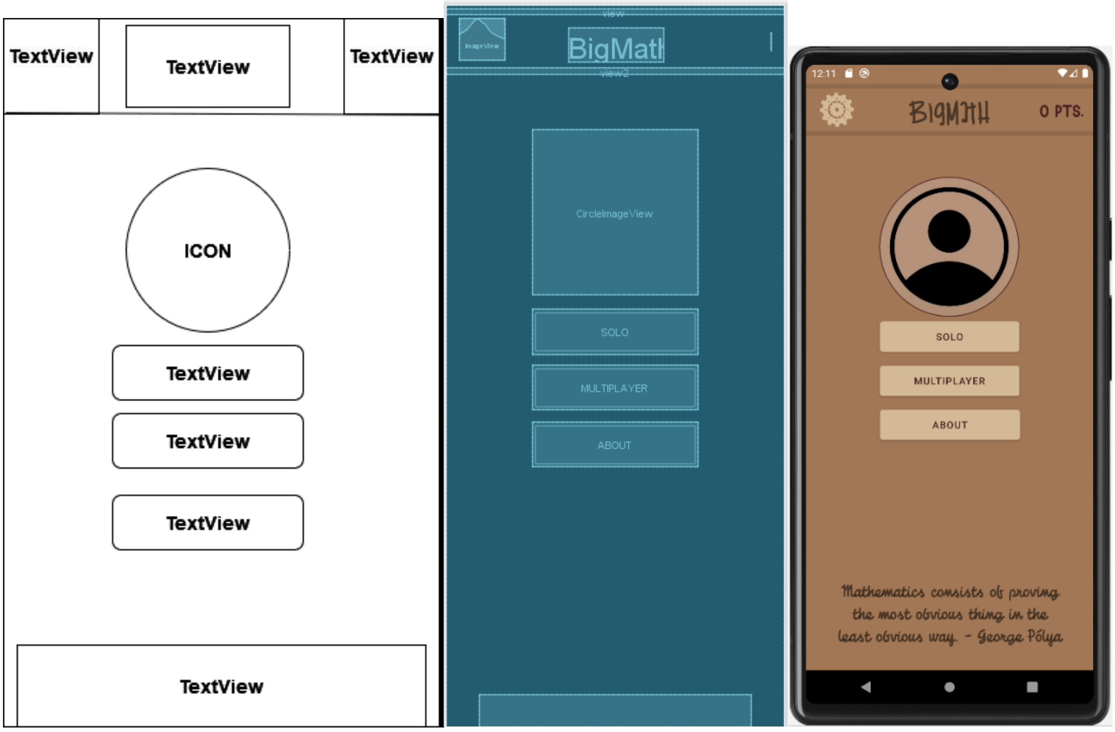


UI/UX

Loading Screen

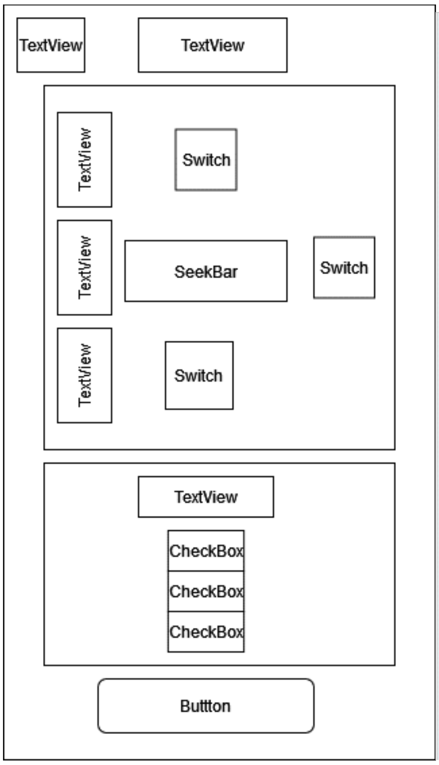


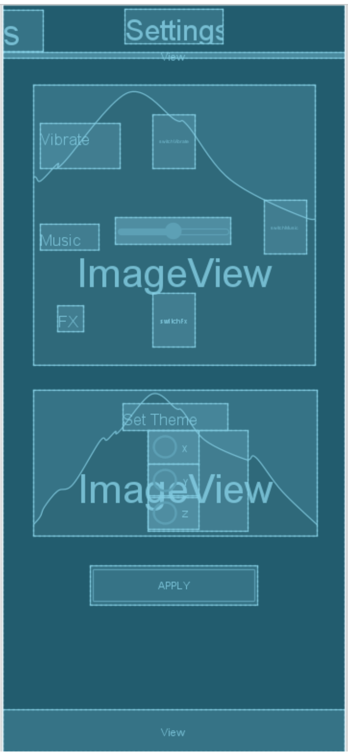
Home

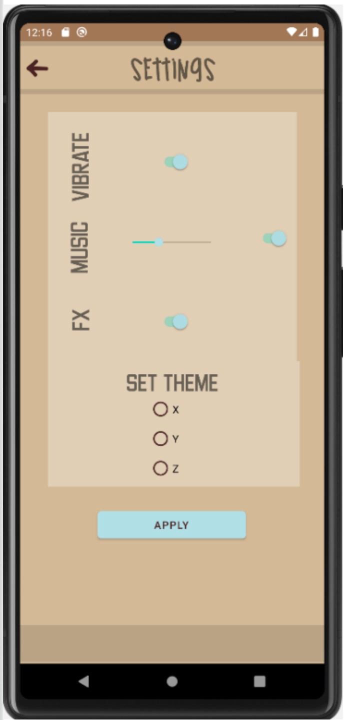


Mathematics consists of proving the most obvious thing in the least obvious way. - George Pólya

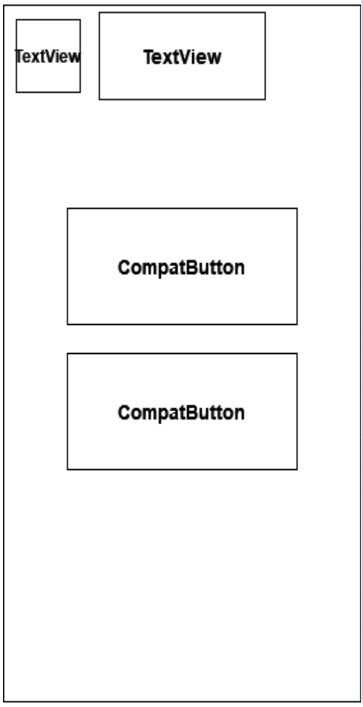
Settings

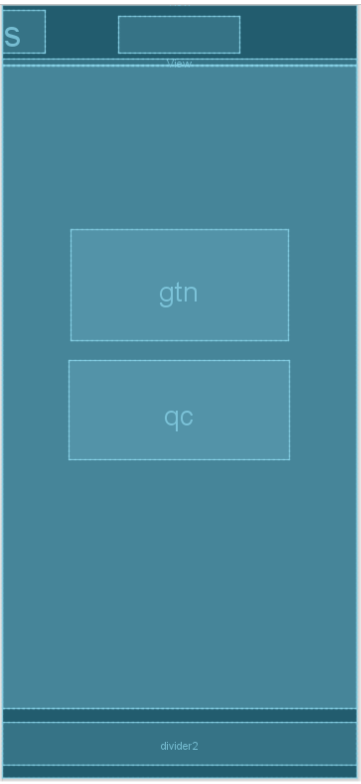


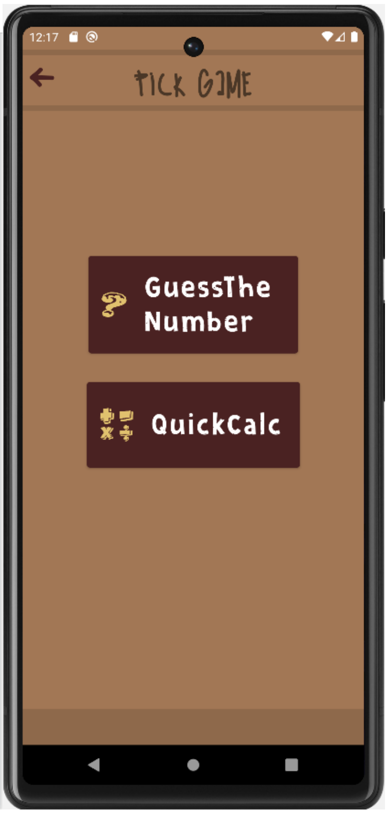




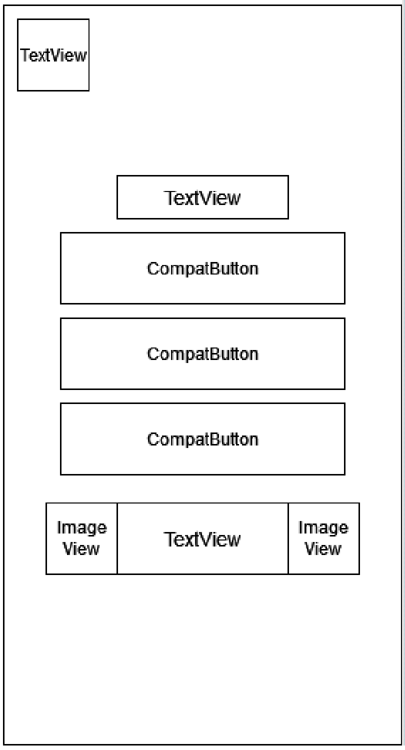
Choose Game

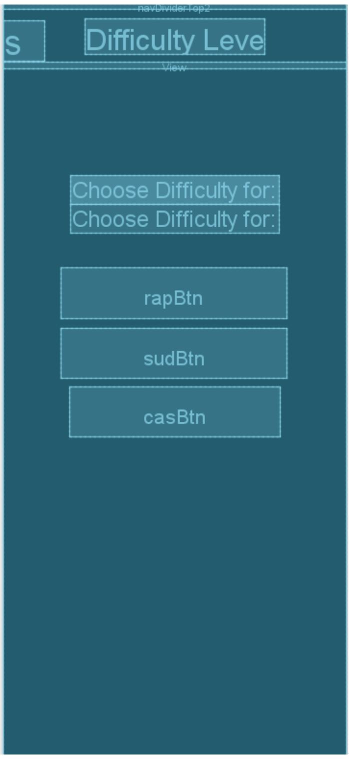


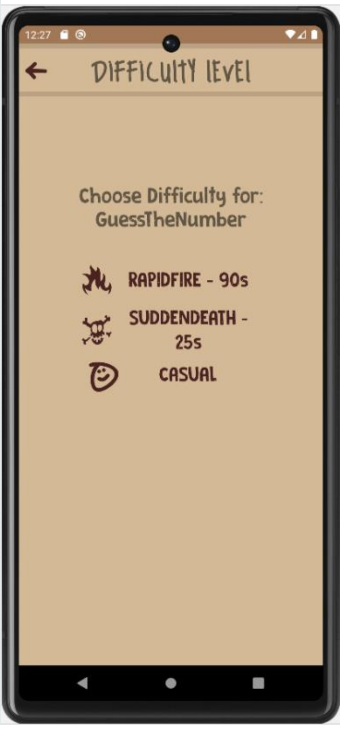




Speed Screen

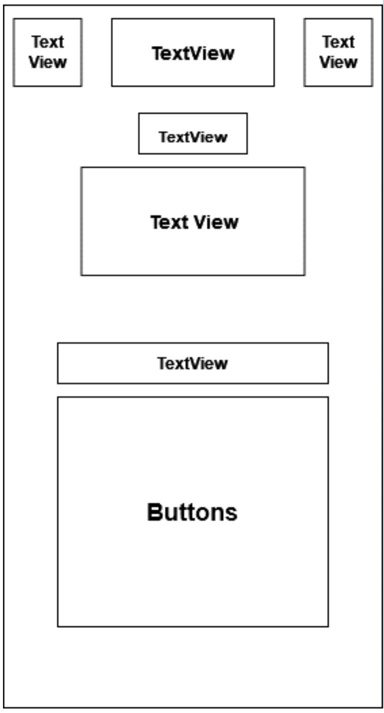


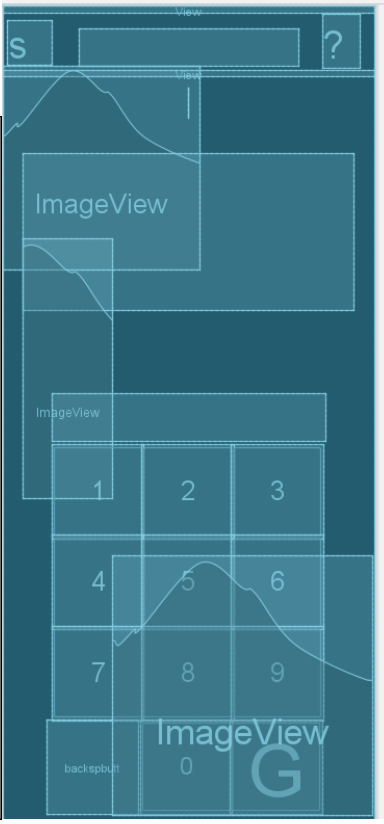


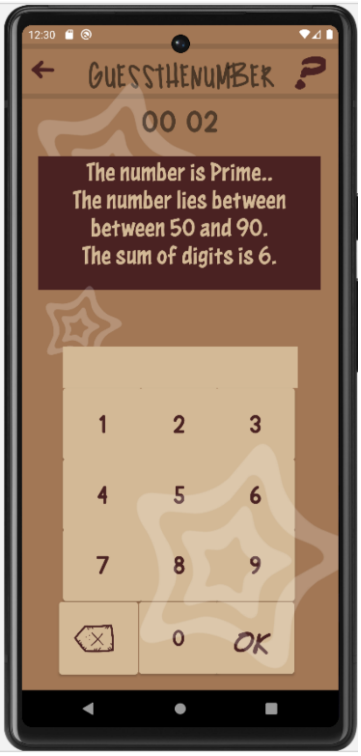


Choose Difficulty for:
GuessTheNumber

GuessTheNumber

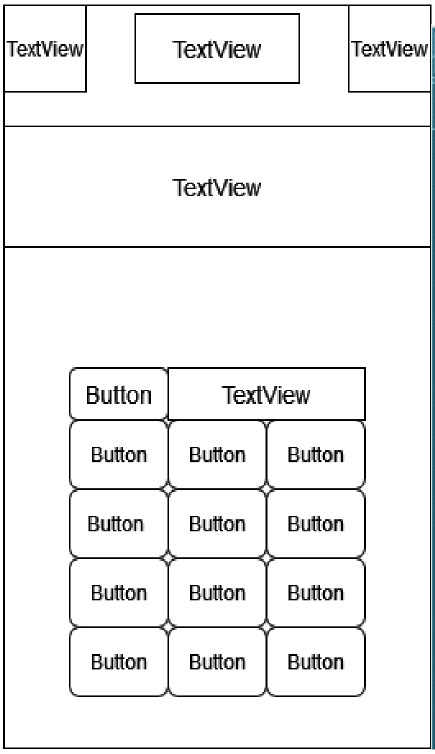


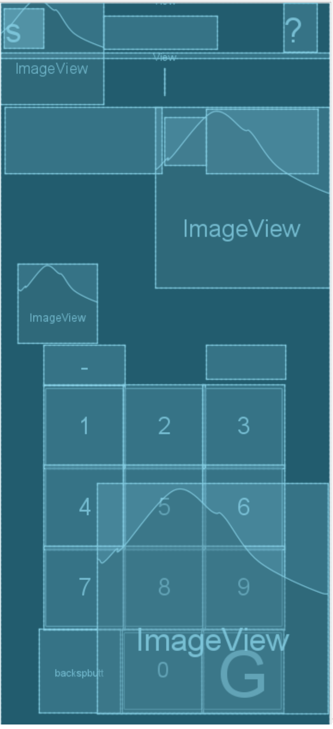


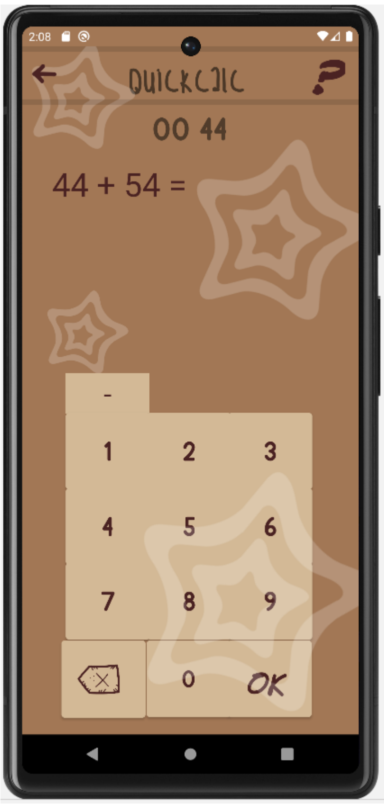


The number is Prime..
The number lies between
between 50 and 90.
The sum of digits is 6.

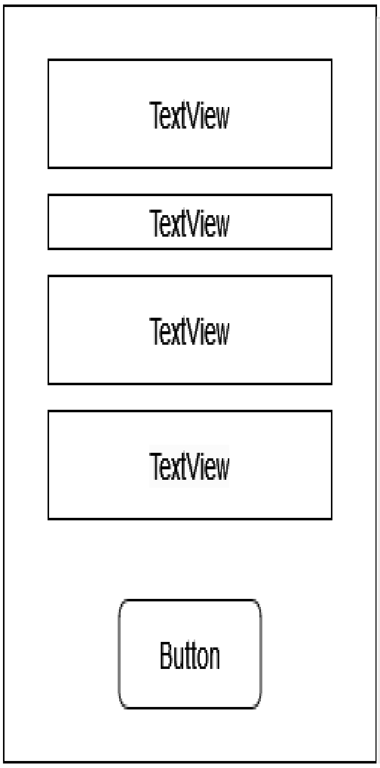
QuickCalc

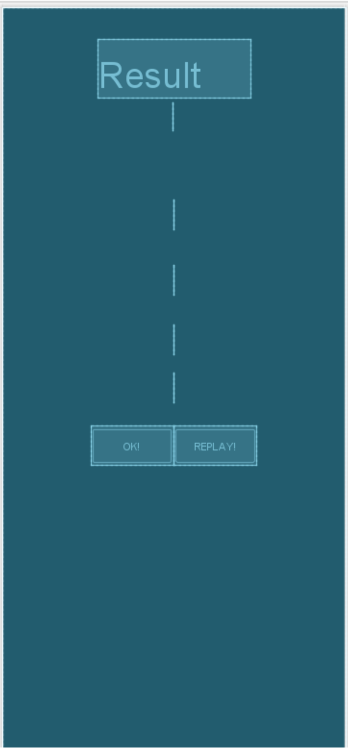


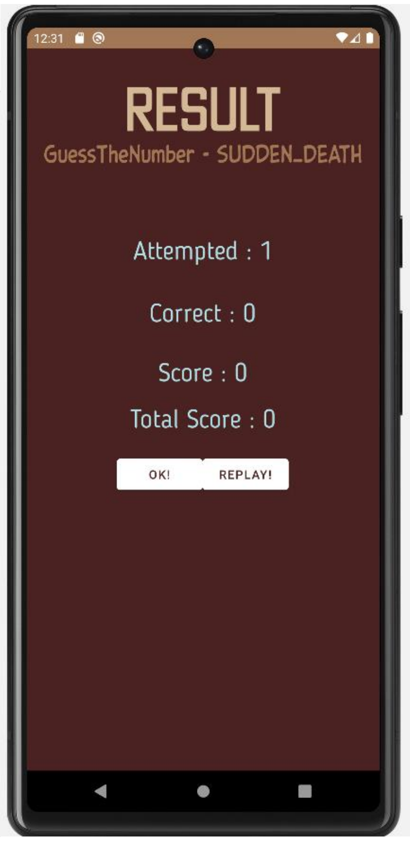




Result Activity







Attempted : 1

Correct : 0

Score : 0

Total Score : 0

Coding
AndroidManifest.xml

```
    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/Theme.BigMath"
        tools:targetApi="31">
        <activity
            android:name=".AboutPage"
            android:exported="false" />
        <activity
            android:name=".ResultActivity"
            android:exported="false" />
        <activity
            android:name=".QuickCalc"
            android:exported="false" />
        <activity
            android:name=".GuessTheNumber"
            android:exported="false" />
        <activity
            android:name=".SpeedScreen"
            android:exported="false" />
        <activity
            android:name=".Settings"
            android:exported="false" />
        <activity
            android:name=".SinglChooseGame"
            android:exported="false" />
        <activity
            android:name=".Home"
            android:exported="false" />
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
```

Directory structure and files in BigMath\app\src\main

ExampleInstrumentedTest.java
AndroidManifest.xml

ic_launcher-playstore.png

AboutPage.java
BigMathGestureListener.java
BigMathTimer.java
BigMathTouchEvents.java
DbaseImplementation.java
GuessTheNumber.java
Home.java
MainActivity.java
QuickCalc.java
ResultActivity.java
Settings.java
SinglChooseGame.java
SpeedScreen.java
BigMathDao.java
BigMathDatabase.java
BigMathScoresDB.java

background2.png
backspace.png
calc_ic.png
cas_d.png
coin.png
gear_d3b996.png
gtn_ic.png
profile_icon1.png
rap_n.png
start_a.png
sud_y_incaps.png
white.png
white_bg_star.png
white_horizontal.png

astounder_squared_lcbb.otf
blogger_sans.otf
cc_samaritan_tall_lower_reg.otf
giant_sized_spectacular_script_bb.otf
headlines_deluxe.otf
madsense.otf
mouth_breather_bb_bold.ttf
pw_new_arrows.ttf
surprze_birthday.ttf
zombie_starfish_eroded_regular.otf

activity_about_page.xml
activity_guess_the_number.xml
activity_home.xml
activity_main.xml
activity_quick_calc.xml
activity_result.xml
activity_settings.xml
activity_singl_choose_game.xml
activity_speed_screen.xml
help_dig.xml
quit_dlg.xml

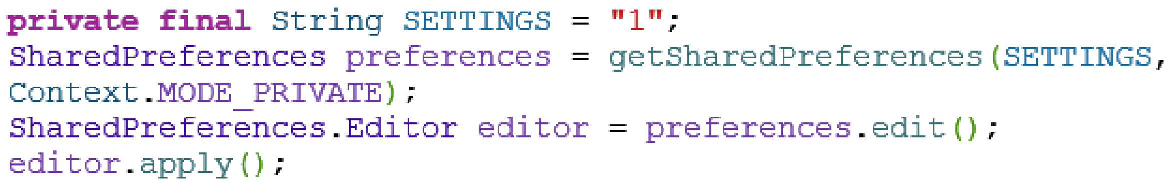
ic_launcher.xml
ic_launcher.png
ic_launcher_background.png
ic_launcher_foreground.png
ic_launcher.png
ic_launcher_background.png
ic_launcher_foreground.png
ic_launcher.png
ic_launcher_background.png
ic_launcher_foreground.png
ic_launcher.png
ic_launcher_background.png
ic_launcher_foreground.png
ic_launcher.png
ic_launcher_background.png
ic_launcher_foreground.png

iono_break.ogg
iono_end.ogg
iono_home.ogg
iono_main.ogg
kalimba_c.ogg
kalimba_d.ogg
kalimba_e.ogg
perc.ogg
perc_snap.ogg
sword.ogg

colors.xml
strings.xml
themes.xml
themes.xml

backup_rules.xml
data_extraction_rules.xml
ExampleUnitTest.java

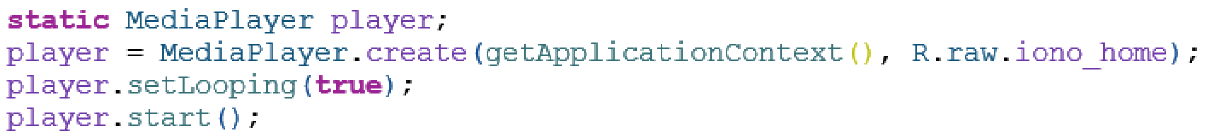
Use of shared preferences



```
private final String SETTINGS = "1";
SharedPreferences preferences = getSharedPreferences(SETTINGS,
Context.MODE_PRIVATE);
SharedPreferences.Editor editor = preferences.edit();
editor.apply();
```

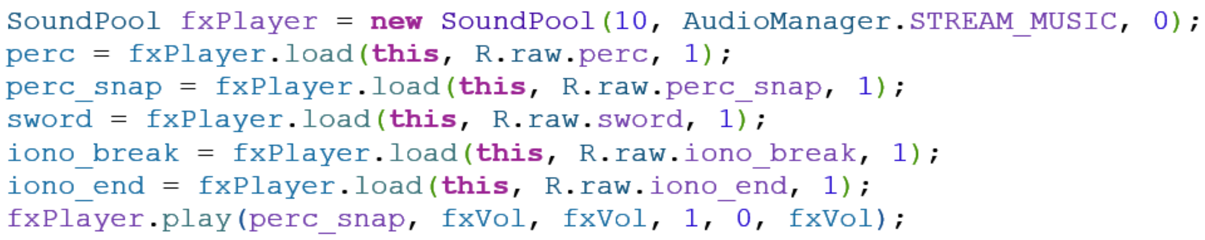
Music player

MediaPlayer.class implementation



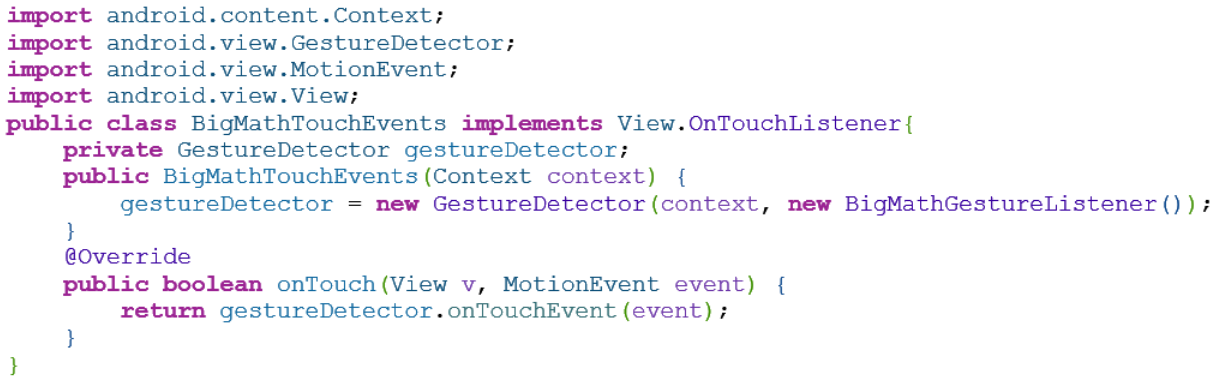
```
static MediaPlayer player;
player = MediaPlayer.create(getApplicationContext(), R.raw.iono_home);
player.setLooping(true);
player.start();
```

SoundPool.class implementation

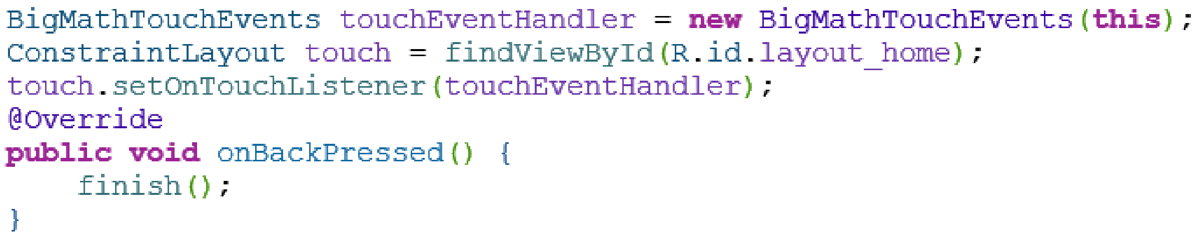


```
SoundPool fxPlayer = new SoundPool(10, AudioManager.STREAM_MUSIC, 0);
perc = fxPlayer.load(this, R.raw.perc, 1);
perc_snap = fxPlayer.load(this, R.raw.perc_snap, 1);
sword = fxPlayer.load(this, R.raw.sword, 1);
iono_break = fxPlayer.load(this, R.raw.iono_break, 1);
iono_end = fxPlayer.load(this, R.raw.iono_end, 1);
fxPlayer.play(perc_snap, fxVol, fxVol, 1, 0, fxVol);
```

Touchgestures listener



```
import android.content.Context;
import android.view.GestureDetector;
import android.view.MotionEvent;
import android.view.View;
public class BigMathTouchEvents implements View.OnTouchListener{
    private GestureDetector gestureDetector;
    public BigMathTouchEvents(Context context) {
        gestureDetector = new GestureDetector(context, new BigMathGestureListener());
    }
    @Override
    public boolean onTouch(View v, MotionEvent event) {
        return gestureDetector.onTouchEvent(event);
    }
}
```



```
BigMathTouchEvents touchEventHandler = new BigMathTouchEvents(this);
ConstraintLayout touch = findViewById(R.id.layout_home);
touch.setOnTouchListener(touchEventHandler);
@Override
public void onBackPressed() {
     finish();
}
```

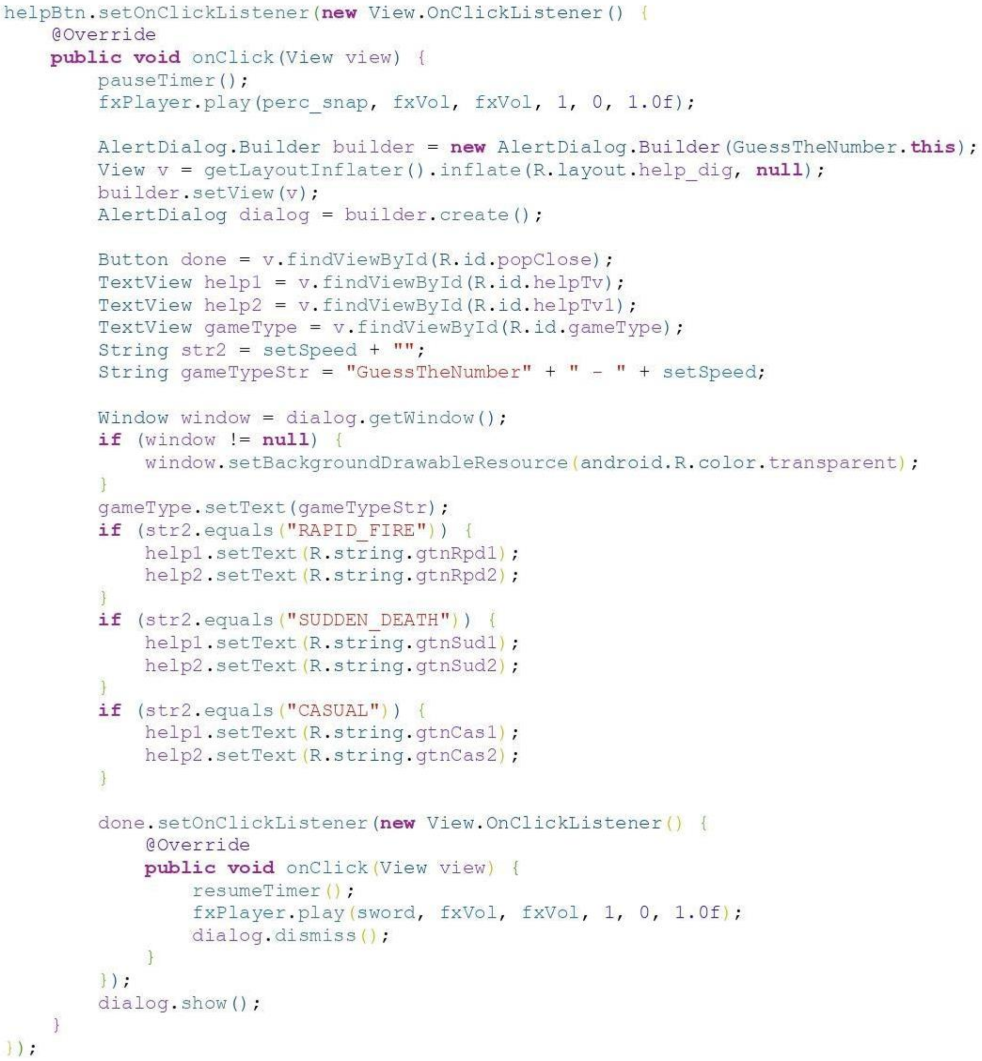
Setting Math Quotes

```
private void setQuote() {
     // selecting a random quote from strings resources
     int[] quoteData = {R.string.quote1, R.string.quote2, R.string.quote3,
R.string.quote4, R.string.quote5,
                R.string.quote6, R.string.quote7, R.string.quote8,
R.string.quote9, R.string.quote10};
     int randomIndex = new Random().nextInt(10);
     String quote = getString(quoteData[randomIndex]);
     quoteView.setText(quote);
}
```

Update Timer logic

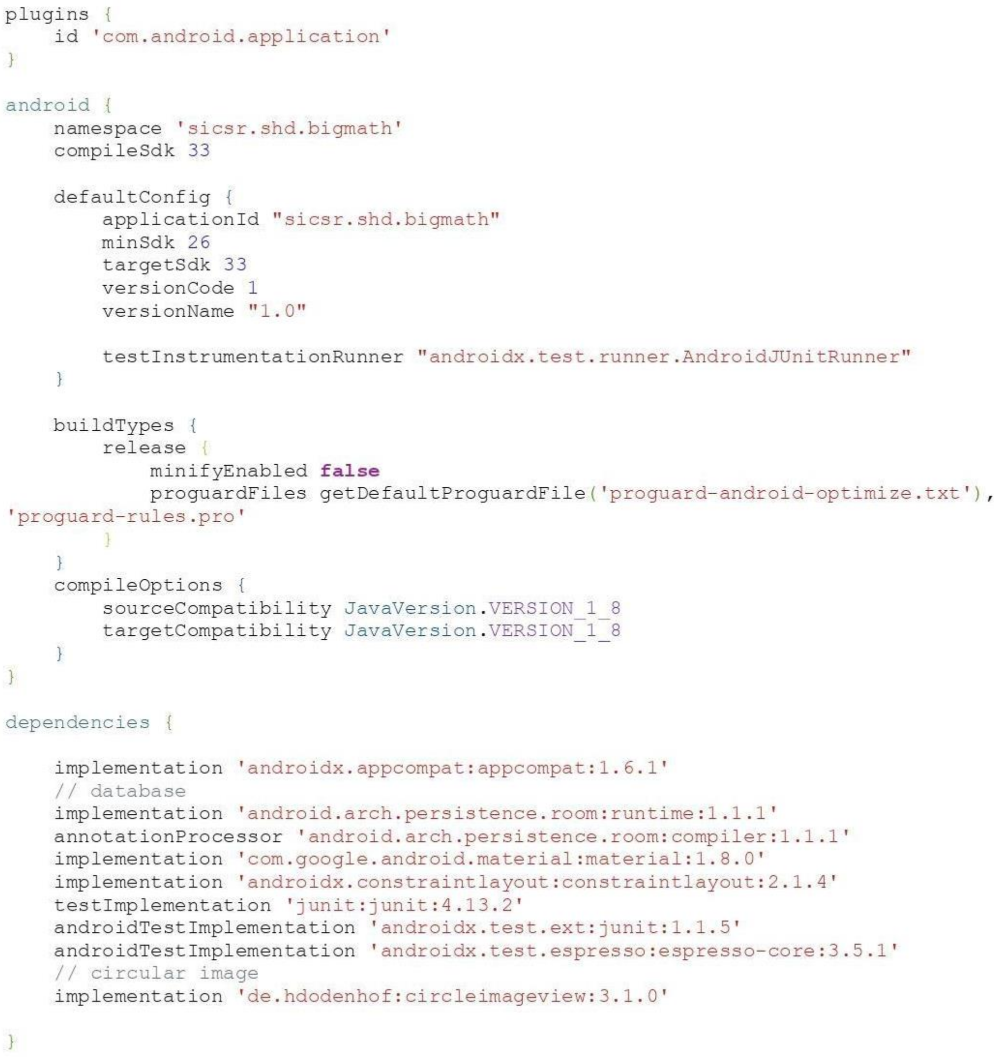
```
private void updateTimer() {
    int minutes = (int) (timeLeftInMillis / 1000) / 60;
    int seconds = (int) (timeLeftInMillis / 1000) % 60;

    String timeLeftFormatted = String.format(Locale.getDefault(), "%02d %02d", minutes,
seconds);
    timerTv.setText(timeLeftFormatted);
}
```

Help dialogue



```
helpBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        pauseTimer();
        fxPlayer.play(perc_snap, fxVol, fxVol, 1, 0, 1.0f);

        AlertDialog.Builder builder = new AlertDialog.Builder(GuessTheNumber.this);
        View v = getLayoutInflater().inflate(R.layout.help_dig, null);
        builder.setView(v);
        AlertDialog dialog = builder.create();

        Button done = v.findViewById(R.id.popClose);
        TextView help1 = v.findViewById(R.id.helpTv);
        TextView help2 = v.findViewById(R.id.helpTv1);
        TextView gameType = v.findViewById(R.id.gameType);
        String str2 = setSpeed + "";
        String gameTypeStr = "GuessTheNumber" + " - " + setSpeed;

        Window window = dialog.getWindow();
        if (window != null) {
            window.setBackgroundDrawableResource(android.R.color.transparent);
        }
        gameType.setText(gameTypeStr);
        if (str2.equals("RAPID_FIRE")) {
            help1.setText(R.string.gtnRpd1);
            help2.setText(R.string.gtnRpd2);
        }
        if (str2.equals("SUDDEN_DEATH")) {
            help1.setText(R.string.gtnSud1);
            help2.setText(R.string.gtnSud2);
        }
        if (str2.equals("CASUAL")) {
            help1.setText(R.string.gtnCas1);
            help2.setText(R.string.gtnCas2);
        }

        done.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                resumeTimer();
                fxPlayer.play(sword, fxVol, fxVol, 1, 0, 1.0f);
                dialog.dismiss();
            }
        });
        dialog.show();
    }
});
```

build.gradle app



```
plugins {
    id 'com.android.application'
}

android {
    namespace 'sicsr.shd.bigmath'
    compileSdk 33

    defaultConfig {
        applicationId "sicsr.shd.bigmath"
        minSdk 26
        targetSdk 33
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'),
'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
}

dependencies {

    implementation 'androidx.appcompat:appcompat:1.6.1'
    // database
    implementation 'android.arch.persistence.room:runtime:1.1.1'
    annotationProcessor 'android.arch.persistence.room:compiler:1.1.1'
    implementation 'com.google.android.material:material:1.8.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.4'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.5'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'
    // circular image
    implementation 'de.hdodenhof:circleimageview:3.1.0'

}
```

An essential entry in gradle.properties

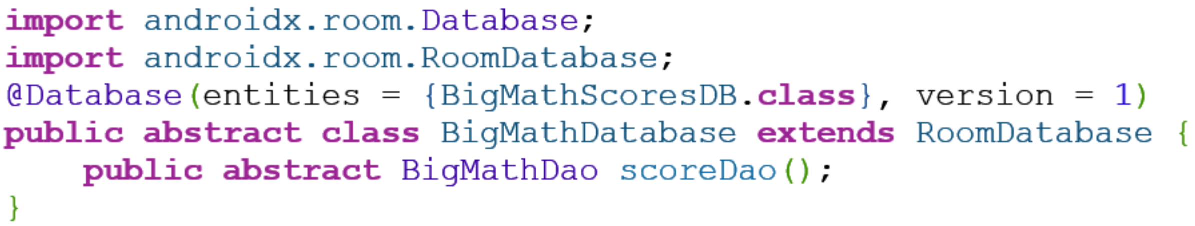
```
android.enableJetifier=true
```

Timer Logic

```java
import android.os.CountDownTimer;
public class BigMathTimer {
    private CountDownTimer countDownTimer;
    private long timeRemaining;
    public BigMathTimer(long timeInMillis) {
        this.timeRemaining = timeInMillis;
        this.countDownTimer = new CountDownTimer(timeInMillis, 1000) {
            @Override
            public void onTick(long millisUntilFinished) {
                timeRemaining = millisUntilFinished;
            }

            @Override
            public void onFinish() {
                timeRemaining = 0;
            }
        };
    }
    public void start() {
        countDownTimer.start();
    }
    public void pause() {
        countDownTimer.cancel();
    }
    public void resume() {
        countDownTimer = new CountDownTimer(timeRemaining, 1000) {
            @Override
            public void onTick(long millisUntilFinished) {
                timeRemaining = millisUntilFinished;
            }

            @Override
            public void onFinish() {
                timeRemaining = 0;
            }
        };
        countDownTimer.start();
    }
    public long getTimeRemaining() {
        return timeRemaining;
    }
    public void setTimeRemaining(long timeRemaining) {
        this.timeRemaining = timeRemaining;
    }
}
```

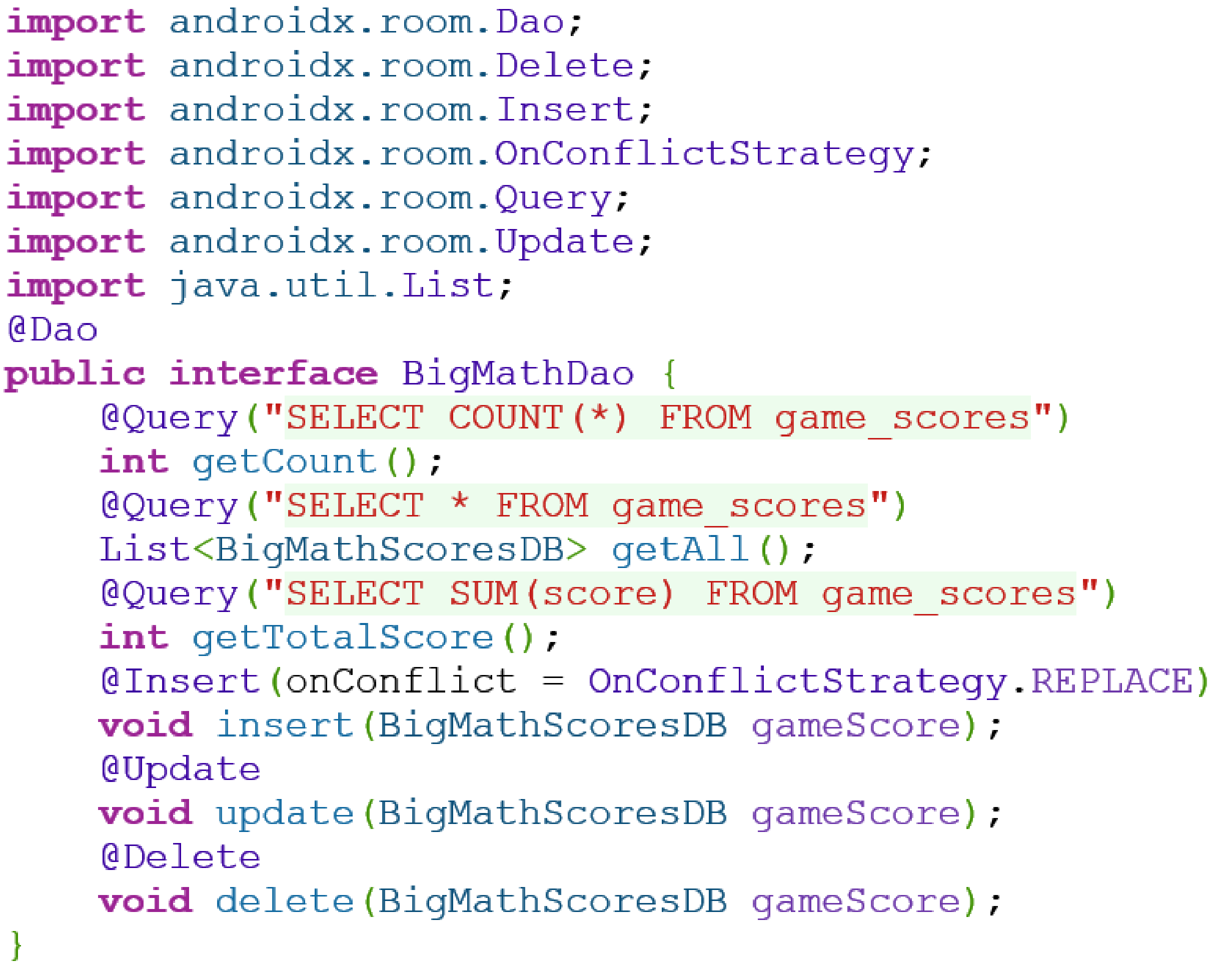
Update timer logic

Database Model



```
import androidx.room.Database;
import androidx.room.RoomDatabase;
@Database(entities = {BigMathScoresDB.class}, version = 1)
public abstract class BigMathDatabase extends RoomDatabase {
    public abstract BigMathDao scoreDao();
}
```

Data Access Object



```
import androidx.room.Dao;
import androidx.room.Delete;
import androidx.room.Insert;
import androidx.room.OnConflictStrategy;
import androidx.room.Query;
import androidx.room.Update;
import java.util.List;
@Dao
public interface BigMathDao {
    @Query("SELECT COUNT(*) FROM game_scores")
    int getCount();
    @Query("SELECT * FROM game_scores")
    List<BigMathScoresDB> getAll();
    @Query("SELECT SUM(score) FROM game_scores")
    int getTotalScore();
    @Insert(onConflict = OnConflictStrategy.REPLACE)
    void insert(BigMathScoresDB gameScore);
    @Update
    void update(BigMathScoresDB gameScore);
    @Delete
    void delete(BigMathScoresDB gameScore);
}
```

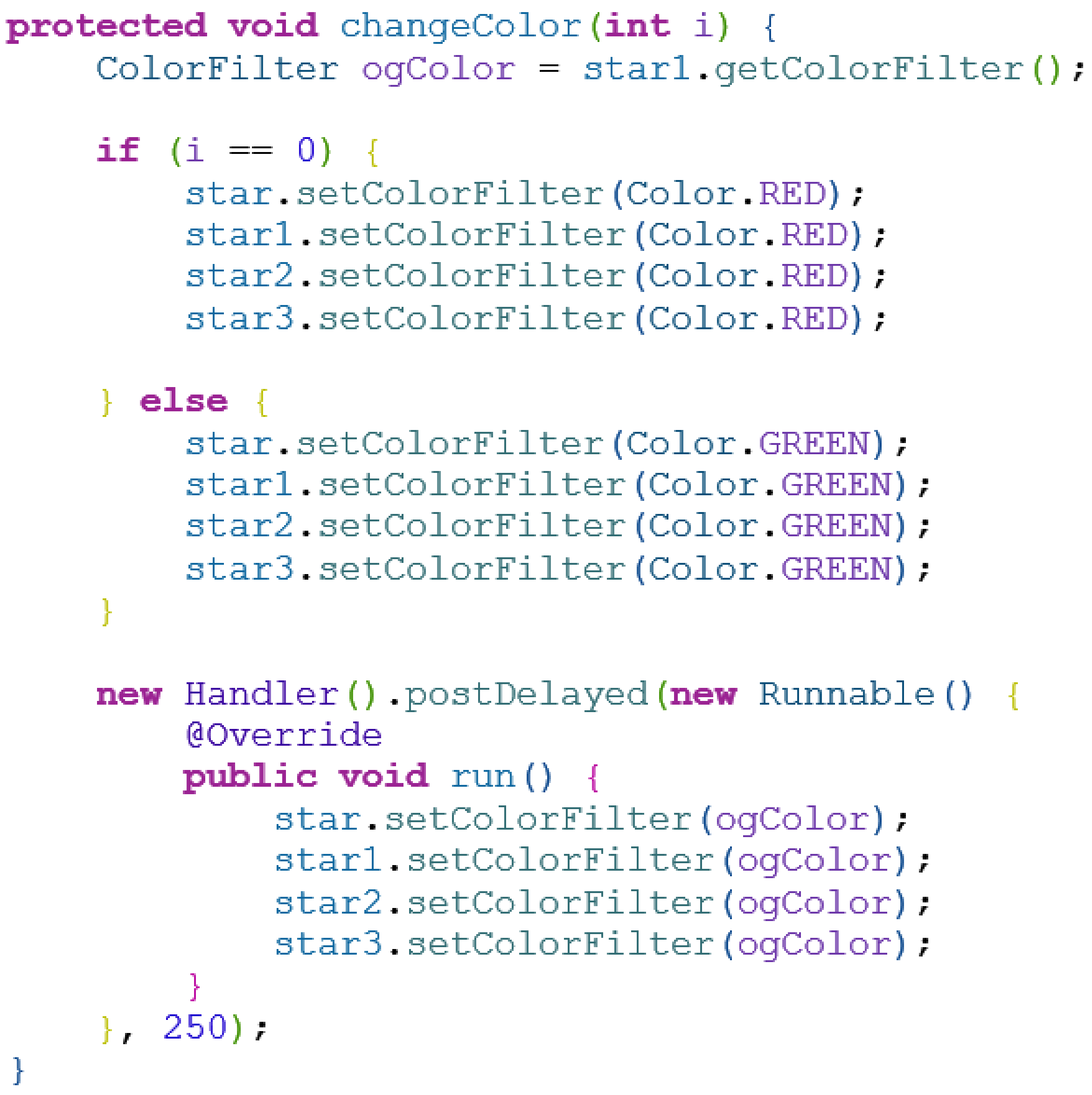
Database

```
import androidx.room.ColumnInfo;
import androidx.room.Entity;
import androidx.room.PrimaryKey;
@Entity(tableName = "game_scores")
public class BigMathScoresDB {
    @PrimaryKey(autoGenerate = true)
    public int id;
    @ColumnInfo(name = "game")
    public String gameName;
    @ColumnInfo(name = "score")
    public int earnedScore;
    @ColumnInfo(name = "attempted")
    public int questions;
    public BigMathScoresDB(String gameName, int earnedScore, int questions) {
        this.gameName = gameName;
        this.earnedScore = earnedScore;
        this.questions = questions;
    }
    public void setId(int id) {
        this.id = id;
    }
}
```

Carrying out operations

```
new Thread(new Runnable() {
    @Override
    public void run() {
        BigMathDatabase dBase = Room.databaseBuilder(getApplicationContext(),
BigMathDatabase.class, "game_scores").build();
        BigMathScoresDB obj = new BigMathScoresDB("GuessTheNumber", score, quesNo);
        dBase.scoreDao().insert(obj);
        totScore = dBase.scoreDao().getTotalScore();
    }
});
```

Valid-Invalid logic



```
protected void changeColor(int i) {
    ColorFilter ogColor = star1.getColorFilter();

    if (i == 0) {
        star.setColorFilter(Color.RED);
        star1.setColorFilter(Color.RED);
        star2.setColorFilter(Color.RED);
        star3.setColorFilter(Color.RED);

    } else {
        star.setColorFilter(Color.GREEN);
        star1.setColorFilter(Color.GREEN);
        star2.setColorFilter(Color.GREEN);
        star3.setColorFilter(Color.GREEN);
    }

    new Handler().postDelayed(new Runnable() {
        @Override
        public void run() {
            star.setColorFilter(ogColor);
            star1.setColorFilter(ogColor);
            star2.setColorFilter(ogColor);
            star3.setColorFilter(ogColor);
        }
    }, 250);
}
```

Testing

Initial Project Testing

SL No.	Test Case description	Expected Result	Actual Result	Status	Fixes(if any)
1.1	Application Build	No run-time errors or dependency failures	Achieved	Pass	
1.2	A single instance of Background music generation	The application should play music tracks	Achieved	Pass	
1.3	Fx generation	Use of SoundPool to play sounds quickly	Achieved	Pass	
1.4	Initialize Vibrator	Get user permission to allow the use of the Vibrate service	Achieved	Pass	
1.5	Save settings locally	Use of shared preferences to save constantly needed data locally	Achieved	Pass	

Advanced Project Testing

SL No.	Test Case description	Expected Result	Actual Result	Status	Fixes(if any)
2.1	Application Build	No run-time errors or dependency failures	Achieved	Pass	
2.2	No overlap of background music	The MediaPlayer should stop when switching certain activities	The media player only controls and transfers when going ahead, and not parsing backwards.	Failed	Switch from MediaPlayer to MusicService thread.
2.3	Run GTN cycle successfully	The GuessTheNumber game should run properly	Achieved	Pass	
2.4	Flexible UI	The UI should look identical on all	The UI seems different on	Failed	

		devices, irrespective of screen size.	specific screen sizes		
2.5	Result Activity	Display result activity through shared preferences, intent parameters and database	Achieved	Pass	

Intensive features

SL No.	Test Case description	Expected Result	Actual Result	Status	Fixes(if any)
3.1	Touch Gesture initialisation	Implement back swipe feature	Achieved	Pass	
3.2	SQLite database; (RoomDatabase)	The scores should be saved and totalled correctly.	Valid data model	Pass	
3.3	Application Build	No Runtime errors while gradle build	Runtime errors due to duplicate classes were found but fixed.	Pass	Required jetifier

Further scope for development

Mathematics and Android Application development are vast subjects. Gameplay experiences can be implemented in various ways concerning the scope of a project. Every application follows a chain of multiple and consistent releases with new updates. The number of these releases varies on the scope of the project. Project BigMath fulfilled 75% of its initial scope in release 1.0. However, the build-up and implementation of the multiplayer model and heavy focus on customisation options still needed to be achieved. These crucial points can be worked on in future releases. The multiplayer model can be built based on local and network session modes. The theme and music customisations can be offered on preset modes or made with a model allowing the user complete access to the service.

It is certain that allowing the user only 2 modes does not meet the standards and specifications of a solid mini-games application. It is crucial to work on more algorithms and game logic that can be combined with the GUI to total new game modes. These can include minor modes like questions on square roots to GUI-heavy features like 2048 and Sudoku.

An additional and vital feature includes the cloud application and setting up a service to synchronise the user’s scores to a real-time database. This service helps users easily migrate from one device to other. The data from these synced scores can also be utilised to further strengthen the multiplayer model and create a regional or worldwide raking system. Overall, with further development, the BigMath project could stand up to its name and be a feature-heavy application by meeting its potential scope.

References

Literature Review Case Study Sources

Case 1: Math Games: Math for Kids	https://bit.ly/3JMq3sb
Case 2: Math Kids: Math Games for Kids	https://bit.ly/2NMm3sp
Case 3: Math Games, Learn Add Multiply	https://bit.ly/3XaH6Hi
Case 4: Math Games	https://bit.ly/3DNzQdF
Case 5: Math Games: to Learn Math	https://bit.ly/2Mfixru
Case 6: Mathematics	https://bit.ly/3XifdgD
Case 7: Math Games - Brain Training	https://bit.ly/2pIW7Tr