Application “Easy Learn”

Development Summary Report

For Lecture Practical Course on Mobile Application Development with Android (Lab)

Developers

Mengru.Ji 21737249 mengru.ji@stud.uni-goettingen.de

Junqi.Sun 21737212 junqi.sun@stud.uni-goettingen.de

1 Introduction

1.1 Motivation

The reason that why we come up with this app idea is because of the lack of passion to learn German. So the purpose of developing this application is to make German learning software more suitable for our own needs. For example, we want to use this application to repeatedly practice the German grammar exercises or vocabulary exercises that we have organized by ourselves. As long as the exercises are stored in the file in the prescribed format, this application can get the exercises organized in the file.

2 Development Process

2.1 Function Specification

The app consists of multiple activities. All the questions, answers and corresponding data will be stored in a SQLite database. The user's account information will be stored in the Room Database. Achievements and Bonus will be unlocked after you reach the goals.

2.2 Main Classes of Application

- Signup Activity
- Login Activity
- Main Activity(AlarmReceiver)
- Daily Attendance Activity(SpecialCalendar, SquareGridView, SquareRelativeLayout, CheckInAdapter)
- Mission Activity(BackDialog,TimerDialog)
- Mission Result Activity(MyProgressBar)
- Wrong Mode Activity
- Competition Start Activity
- Competition Activity
- Question Book Activity(TabPagerAdapter, PagerFragment)
- Timer Activity
- Achievement Activity
- DatabaseCreator(User, UserDao)
- MySQLite(Questions)

2.3 Plan and Division

Mengru.Ji	**Junqi.Sun**
Daily Attendance Activity	Login Activity
Mission Activity	Signup Activity
Mission Result Activity	Main Activity
Wrong Mode Activity	Competition Start Activity
Question Book Activity	Competition Activity
Timer Activity	Achievement Activity
MySQLite	DatabaseCreator

2.4 Relationship between activities

After running the application, the first thing that runs is the login activity. If the user already has an account, you can log in directly, and the app will jump directly to the Main Activity. Otherwise, the app will jump to the Signup Activity. After successful registration, user can log in to the account and jump to the Main Activity.

Click the button on the main activity interface, the application will jump to the corresponding activities(Check in Activity, Mission Activity, Competition Start Activity, Question Book Activity, Timer Activity, Achievement Activity).

After completing an answer task (in the Mission Activity), the application will jump to the Mission Result Activity. On this interface, user can click the corresponding button to enter the Wrong Mode Activity interface to check the wrong questions and all questions.

After reading the competition rules(in the Competition Start Activity), the user will enter the Competition Activity. Every time the game ends, the user can choose to quit the game(back to Main Activity), try again(Start Competition Activity again), or check the German questions(will jump to Wrong Mode Activity) that appear in this game.

Click the logout button on the action bar of the main activity, it will jump to the login activity, and the user can re-select the account to login.

2.5 Function Introduction of Activities

2.5.1 Local Room Database

```
@Database(entities = {User.class}, version = 6,exportSchema = false)

public abstract class DatabaseCreator extends RoomDatabase {
    private static DatabaseCreator instance;

    public static synchronized DatabaseCreator getInstance(Context context) {
        if (instance == null) {
            instance = create(context);
        }
        return instance;
    }

    public static DatabaseCreator create(Context context){
        return Room.databaseBuilder(context,DatabaseCreator.class, name: "users")
                .fallbackToDestructiveMigration().allowMainThreadQueries().build();
    }

    public abstract UsersDao usersDao();
    //public abstract QuestionListDao questionListDao();
    //public  abstract QuestionOptionDao questionOptionDao();


}
```

User objects will be stored in the local room database. To use this database for storing object information, an interface which declared all method that will be used during the editing of your object need to be created.

```java
@Dao
public interface UsersDao {

    /**
     * insert new user into table
     * @param user
     */
    @Insert(onConflict = OnConflictStrategy.REPLACE)
    void insertUsers(User user);

    /**
     * update a set of user
     * @param users
     */
    @Update
    void updateUsers(User... users);

    /**
     * update user
     * @param user
     */
    @Update
    void updateUser(User user);

    /**
     * delete a set of users
     * @param users
     */
    @Delete
    void deleteUsers(User...users);

    /**
     * get all users
     * @return
     */
    @Query("select * from users")
    List<User> loadAllUsers();
```

```java
    /**
     * get user object by username
     * @param username
     * @return
     */
    @Query("select * from users where username = :username")
    User findUserWithName(String username);
```

2.5.2 MySQLite

All Objects that related to Questions are stored in SQLite Database.

```java
//for filepath
db.execSQL("create table " + TABLE_PATH + "(_id integer primary key autoincrement,filepath text)");

//for questions
db.execSQL("create table "+TABLE_QUESTION
        + "(_id integer primary key autoincrement," +
        "description text," +
        "optionA text,optionB text, optionC text, optionD text, " +
        "answer integer, " +
        "type integer," +
        "iscollect integer)");

//for wrong questions
db.execSQL("create table "+TABLE_WRONG
        + "(_id integer primary key autoincrement," +
        "description text," +
        "optionA text,optionB text, optionC text, optionD text, " +
        "answer integer, " +
        "type integer," +
        "iscollect integer," +
        "wrongTime integer)");

//for favorites
db.execSQL("create table "+TABLE_COLLECT
        + "(_id integer primary key autoincrement," +
        "description text," +
        "optionA text,optionB text, optionC text, optionD text, " +
        "answer integer, " +
        "type integer," +
        "iscollect integer)");
```

```
public MySQLite(Context context, String db_name){
    super(context, db_name, factory: null, version: 5);
    db = getWritableDatabase();
}
```

Each user will have a database named after their user name. Each database contains four tables, which store the file name, all question objects, wrong questions, and questions that users add to favorites.

Use ContentValue to insert data into table(should check if already inserted or not). Use Cursor to traverse the entire table and get all questions in this table. Use Assetmanager and BufferedReader to read Questions which organized in the File.

```
AssetManager assetManager = context.getApplicationContext().getAssets();
try {
    InputStream inputStream = assetManager.open( fileName: "questionList1");
```

```
while ((line1 = buffReader.readLine()) != null && (line2 = buffReader.readLine()) != null
        && (line3 = buffReader.readLine()) != null) {
```

2.5.3 Login Activity

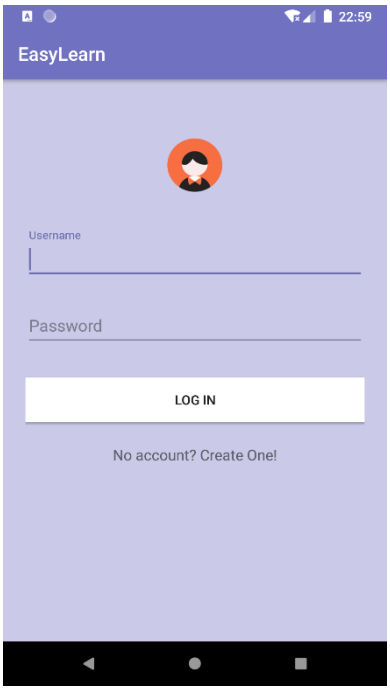


Figure 1

If you already have an account, you can choose to log in directly. After clicking the login button, the application will search in the database based on the user name you entered. If the user name is not registered, an error message will appear, otherwise the application will compare the found password with the password you submitted. If correct, the login is successful, otherwise an error message will appear.

Without an account, you should click the gray font below “No Account? Create One!” and jump to the Signup activity to register.

```
<!--  Username Label -->
<com.google.android.material.textfield.TextInputLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:layout_marginBottom="8dp">
    <EditText
        android:id="@+id/et_login_username"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="textPersonName"
        android:hint="Username" />
</com.google.android.material.textfield.TextInputLayout>
```

In Login and Signup Activity, input will be obtained via Edit Text in Text Input Layout.

2.5.4 Signup Activity

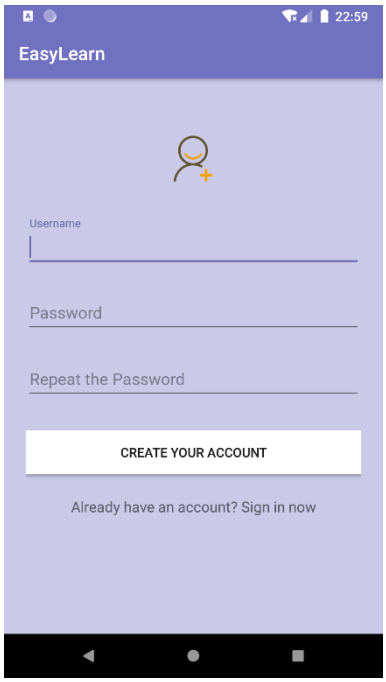


Figure II

Please complete the required fields and then click the register button. If the user name you fill in is not registered, a new data will be created in the corresponding table in the database, otherwise an error message will appear.

2.5.5 Main Activity

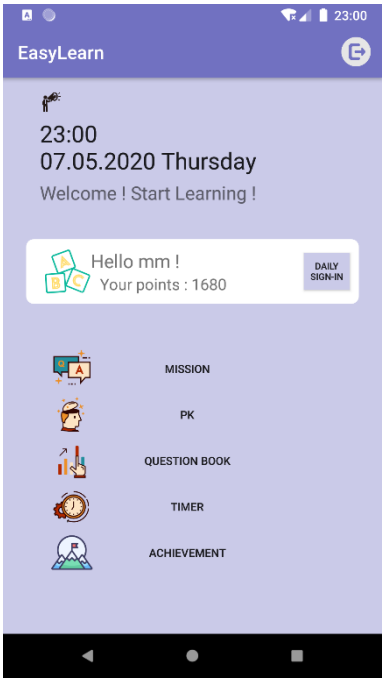


Figure III

```
private void setAlarm() {

    NotificationManager notificationManager = (NotificationManager) this.getSystemService(Context.NOTIFICATION_SERVICE);


    Intent intent = new Intent( packageContext: this, AlarmReceiver.class);
    Bundle bundle = new Bundle();
    bundle.putString("username",loginUser.getUserName());
    bundle.putString("psw",loginUser.getUserPw());
    bundle.putString("date",loginUser.getLastDate());
    bundle.putInt("checkin",loginUser.getCheckInDays());
    bundle.putInt("points",loginUser.getPoints());
    intent.putExtras(bundle);

    intent.setAction("NotificationAlarm");
    PendingIntent pendingIntent = PendingIntent.getBroadcast( context: this, requestCode: 101,intent,PendingIntent.FLAG_UPDATE_CURRENT);
    AlarmManager alarm = (AlarmManager)getSystemService(ALARM_SERVICE);
    //alarm.set(AlarmManager.RTC_WAKEUP,System.currentTimeMillis()+2*1000,pendingIntent);

    Calendar calendar = Calendar.getInstance();
    if(calendar.get(Calendar.HOUR_OF_DAY) > 20){
        calendar.add(Calendar.DAY_OF_MONTH, i1: 1);
    }else if(calendar.get(Calendar.HOUR_OF_DAY) == 20 && calendar.get(Calendar.MINUTE) > 0){
        calendar.add(Calendar.DAY_OF_MONTH, i1: 1);
    }

    calendar.set(Calendar.HOUR_OF_DAY,20);
    calendar.set(Calendar.MINUTE,0);
    calendar.set(Calendar.SECOND,0);
    calendar.set(Calendar.MILLISECOND,0);

    alarm.setRepeating(AlarmManager.RTC_WAKEUP,calendar.getTimeInMillis(), intervalMillis: 24*60*60*1000,pendingIntent);
    ring.setBackgroundResource(R.drawable.icon_notify_black);
    Toast.makeText( context: this, text: "Alarm is set",Toast.LENGTH_SHORT).show();

}
```

On the Main Activity interface, the time will be displayed in real time, and whether you have set a reminder(Black villain with a horn). You can also check your points. Jump to the different activities can be achieved by clicking on corresponding buttons.

2.5.6 Daily Attendance Activity

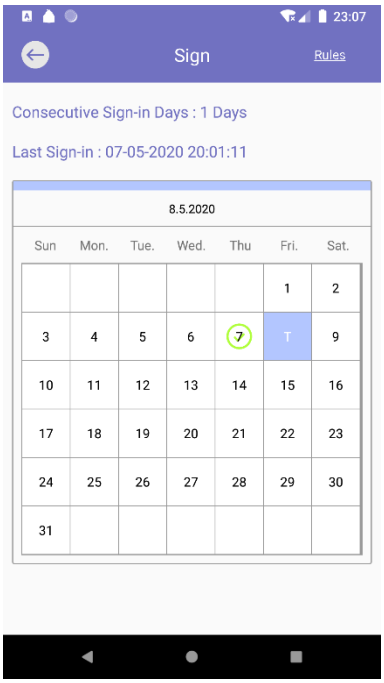


Figure V

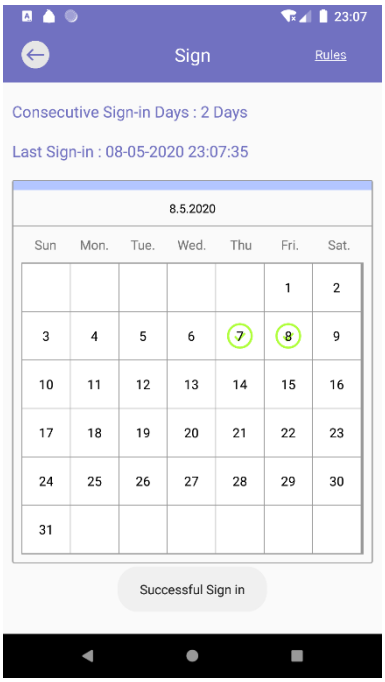


Figure IV

In this activity, you can view the current number of consecutive daily attendance and your last attendance time. In the calendar below, if you have not clocked today, today's date will be marked with a T. Dates that have been clocked will be marked with a green check mark. This calendar will only mark the date of continuous daily attendance. If you interrupt that, the number of days will be cleared. At the beginning of each month, the number of days will also be cleared. The number of consecutive daily attendance days for each user will be recorded in the database.

To determine whether the check-in status is continuous, user’s last check-in date should be obtained and compared with the date of today and yesterday by calling the method “checkAllotSigIn(lastCheckDate)”. If this method returns 1, that means user has already checked-in today. If returns 0, that means user is now in continuous check-in state, but have not checked-in today yet. If returns 2, that means the number of check-in days is 0(last check-in date is before the day before yesterday) and have not checked-in today.

```
try {
    lastCheckDate = sdf.parse(user.getLastDate());
    checkResult = checkAllotSigIn(lastCheckDate);
} catch (Exception e) {
    e.printStackTrace();
}
```

```
//update consecutive check in days
if(checkResult == 2){
    checkInDays = 0;
}

//Zero at the beginning of the month
if(mDay == 1){
    if(checkResult == 1){
        checkInDays = 1;
    }else{
        checkInDays = 0;
    }
}
```

The value of “checkInDays”(the number of user’s continuous check-in days) will be update according to the result of method “checkAllotSigIn(lastCheckDate)”.

Class Special Calendar, Square Relative Layout(extends Relative Layout), Square Grid View(extends Grid View) and Check In Adapter(extends Base Adapter) are also implemented to calculate and draw the calendar of the current month.

```
<com.example.finalproject.View.SquareGridView
    android:id="@+id/registration_calendar_gv"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginBottom="10dp"
    android:layout_marginLeft="10dp"
    android:layout_marginRight="10dp"
    android:background="#999999"
    android:horizontalSpacing="1dp"
    android:numColumns="7"
    android:padding="1dp"
    android:scrollbars="none"
    android:verticalSpacing="1dp" />
```

```
<com.example.finalproject.View.SquareRelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="#fff">

    <TextView
        android:id="@+id/day"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="1"
        android:gravity="center"
        android:textSize="14sp"
        android:layout_margin="10dp"
        android:textColor="@color/colorBlack"
        android:layout_centerInParent="true"
        />
</com.example.finalproject.View.SquareRelativeLayout>
```

Set up listener for each small grid on the calendar. First, whether the clicked grid represents today will be checked. If it is today, it will be checked whether the user has already checked in today. After this, the information corresponding to the interface and the user will be updated, and a corresponding message will be toasted.

If the above two conditions are not met, a corresponding message will also be toasted. That means, when the grid that representing other date is clicked, corresponding date will be toasted.

```
@Override
public void onItemClick(AdapterView<?> adapterView, View view, int i, long l) {

    String today = mYear + "-" + mMonth + "-" + l;

    if (l!=0){
        if (l == mDay){
            Log.i( tag: "TestDate", msg: "checkResult"+checkResult);
            if(checkResult != 1) {
                checkInDays++;
                user.setCheckInDays(checkInDays);
                Date date = new Date();
                SimpleDateFormat sdf = new SimpleDateFormat( pattern: "dd-MM-yyyy HH:mm:ss");
                String time = sdf.format(date);
                user.setLastDate(time);
                try {
                    checkResult = checkAllotSigIn(date);
                } catch (Exception e) {
                    e.printStackTrace();
                }

                TextView today_tv = view.findViewById(R.id.day);
                today_tv.setBackgroundResource(R.drawable.icon_ok);
                today_tv.setTextColor(Color.BLACK);
                today_tv.setText(l + "");

                tv_checkInDays.setText(checkInDays + " Days");
                tv_lastDate.setText(time);
                view.setBackgroundColor(Color.parseColor( colorString: "#ffffff"));
                Toast.makeText(view.getContext(), text: "Successful Sign in", Toast.LENGTH_SHORT).show();
                //checkResult = 1;
            }else{
                Toast.makeText(view.getContext(), text: "Already signed-in today",Toast.LENGTH_SHORT).show();
            }
        }else{
            Toast.makeText(view.getContext(), text: "You choose : "+today,Toast.LENGTH_SHORT).show();

        }
    }
```

2.5.7 Mission Activity

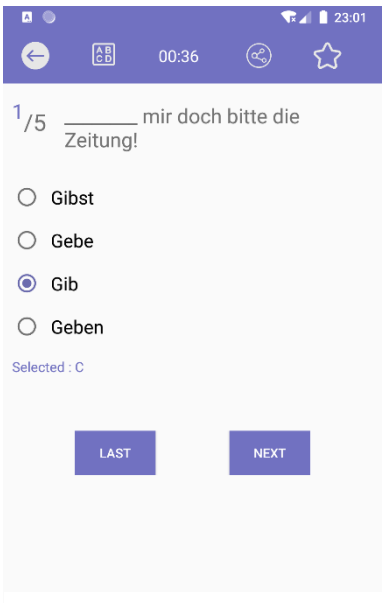


Figure VIII

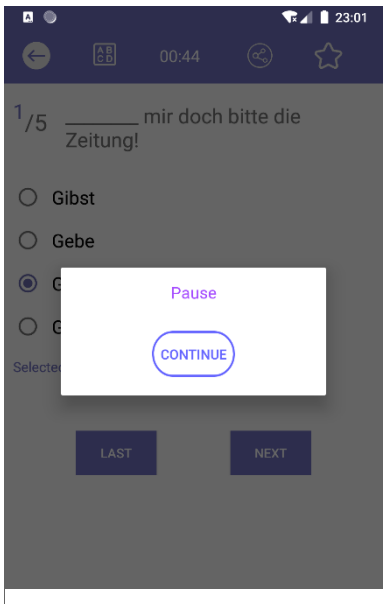


Figure VII

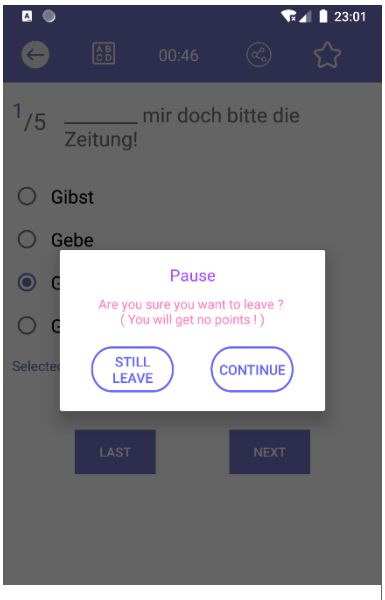


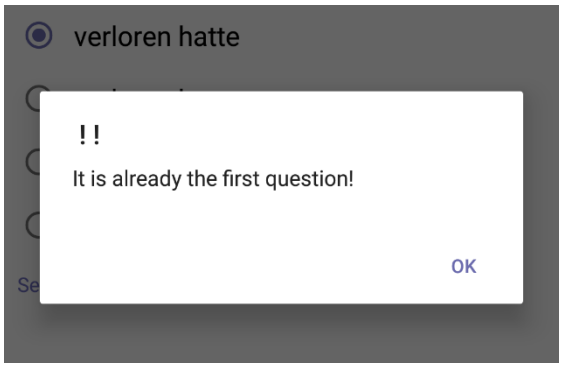
Figure VI

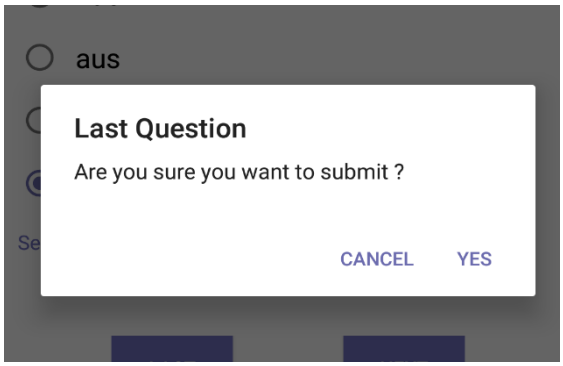
Each time the Mission Activity is run, five questions will be randomly selected from the database.

```
final List<Questions> list = mySQLite.getQuestion();

int[] randomIndex = randomSet( n: 5, list.size());

randomList = new ArrayList<>();
for (int i = 0; i < 5; i++) {
    randomList.add(list.get(randomIndex[i]));
}
```

The options are composed of four radio buttons in the radio group, and the selected options will be displayed on the interface. Click the previous or next button to jump to the previous or next question(Figure VIII). The position of the current question list will be marked by a global variable named "current".

If it is already the first question or the last question, a prompt dialog box will appear.





Five buttons implemented by Text View are placed on the Tool Bar. From left to right: back button (return to the main activity), answer card button (unimplemented), stopwatch (real-time display of time spent answering questions), and share button (to share the current question in text form) and the favorite button (add or delete the current question from the favorites).

Clicking the stopwatch and the back button will display the corresponding prompt dialog box(Figure VII and Figure VI), which are implemented by Class “TimerDialog” and Class “BackDialog”, and the stopwatch will also pause the timing.

2.5.8 Mission Result Activity

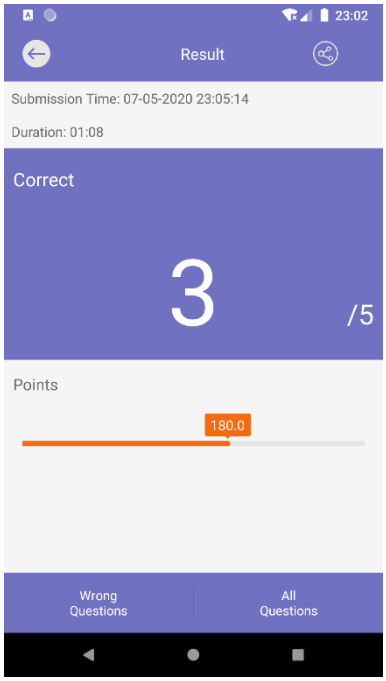
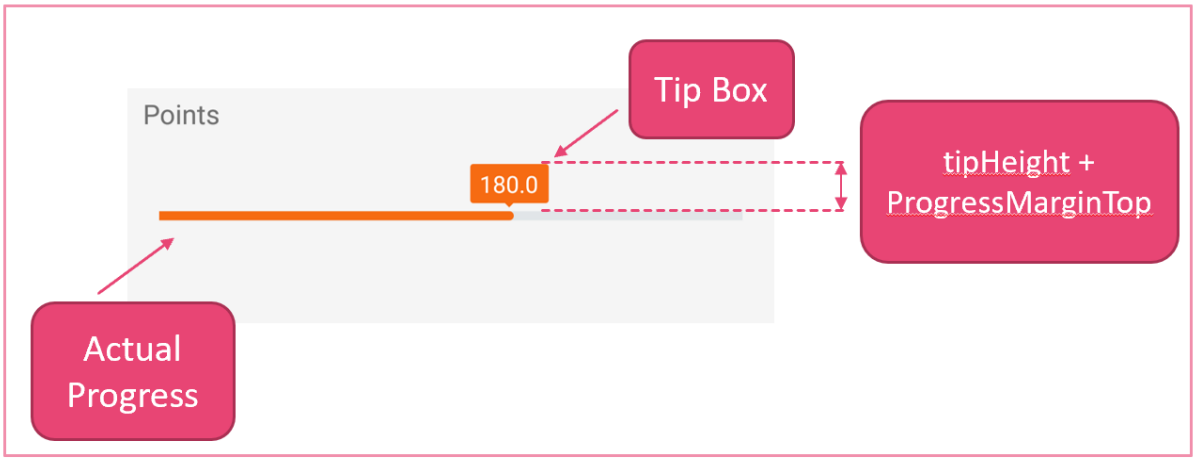


Figure IX

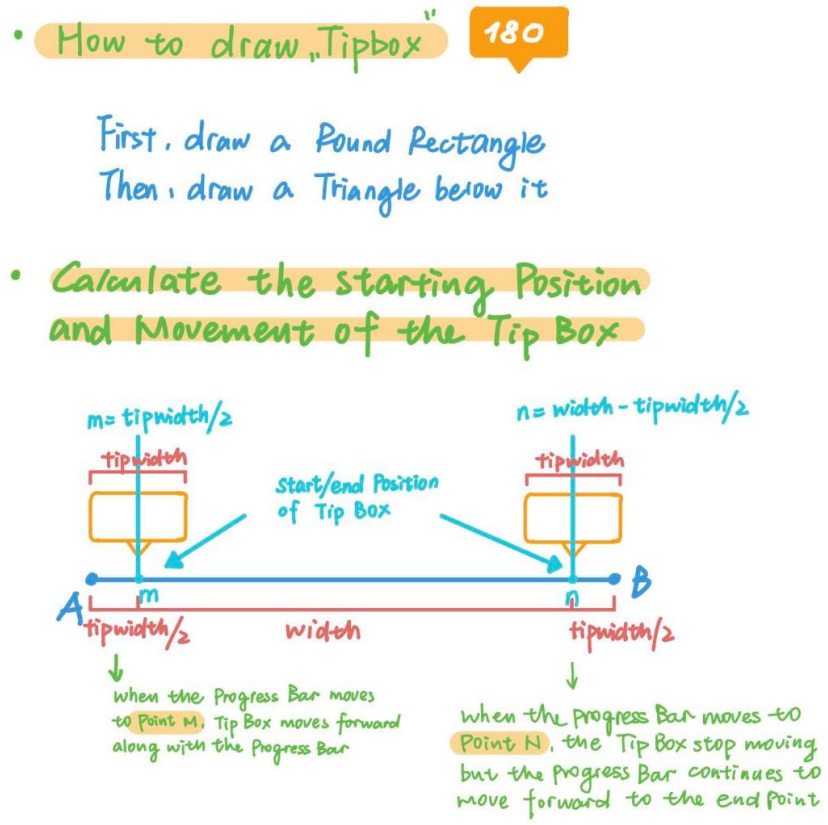
After the quiz is successfully submitted, the app will automatically jump to the Mission Result Activity.

In the current interface, the duration and submission time will be displayed. The correct number of questions will also be displayed in the middle container. Below that, there will be an animated progress bar(implemented by Class MyProgressBar) to show the score of this quiz (300 is the full score, and the score is calculated by multiplying the correct rate).



Because we customize the view, all the elements we see are drawn by the method onDraw(). First draw the bottom progress bar (the background color), and then draw the actual Progress.

```
@Override
protected void onDraw(Canvas canvas) {
    super.onDraw(canvas);

    //draw Background
    canvas.drawLine(getPaddingLeft(),
            startY: tipHeight + progressMarginTop,
            getWidth(),
            stopY: tipHeight + progressMarginTop,
            bgPaint);

    //draw current Progress
    canvas.drawLine(getPaddingLeft(),
            startY: tipHeight + progressMarginTop,
            currentProgress,
            stopY: tipHeight + progressMarginTop,
            progressPaint);

    drawTipView(canvas);
    drawText(canvas, textString);

}
```



Use ValueAnimation to create Animation for our Progress Bar.

```
progressAnimator.addUpdateListener(new ValueAnimator.AnimatorUpdateListener() {
    @Override
    public void onAnimationUpdate(ValueAnimator valueAnimator) {

        float value = (float) valueAnimator.getAnimatedValue();
        //Only show Integer
        textString = formatNum(format2Int(value));
        //Convert the current percentage progress to the proportion corresponding to
        currentProgress = value * mWidth / 300;
        //
        if (progressListener != null) {
            progressListener.currentProgressListener(value);
        }
        // Move the percentage tip box, only when the current progress reaches the
        // stop moving when the progress box moves to the far right, but the progres
        // moveDis is the distance the tip box moves
        if (currentProgress >= (tipWidth / 2) &&
                currentProgress <= (mWidth - tipWidth / 2)) {
            moveDis = currentProgress - tipWidth / 2;
        }
        invalidate();
        setCurrentProgress(value);
    }
});
progressAnimator.start();
```

Based on this score, the user information stored in the database will be updated. At the bottom of the interface are two buttons implemented with Text View. Clicking on these two parts will jump to Wrong Mode Activity to view the wrong questions and all the questions and answers of this quiz. A boolean value will be transferred to the next activity to mark whether the user wants to view wrong questions or all questions. If there are no wrong questions, after clicking “Wrong Questions”, a dialog box(Alert Dialog) will prompt that all the answers are correct. Click the share button in the toolbar to share the test results in text form, and click the back arrow to return to the main activity.

2.5.9 Wrong Mode Activity

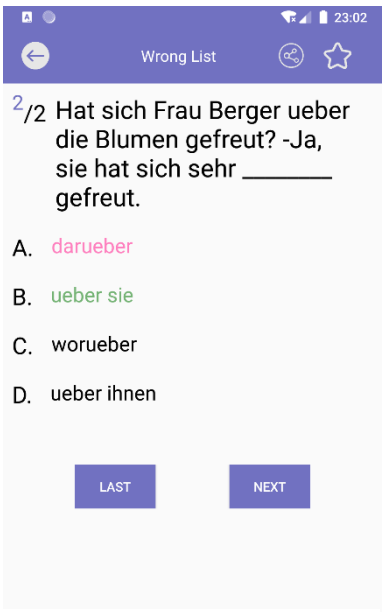


Figure X

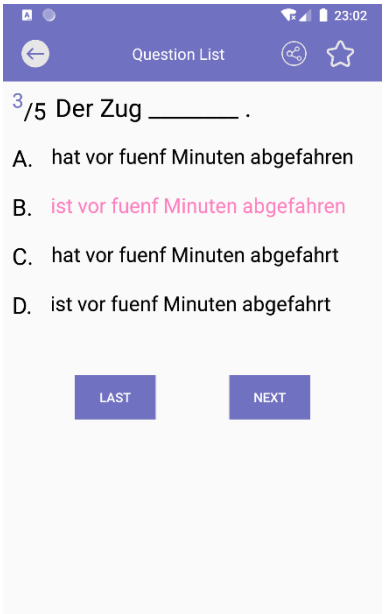


Figure XI

According to the obtained boolean value, it should be judged whether the wrong question layout or all the exercise layout should be displayed. In the wrong question mode(Figure XI), the correct option will be marked in pink and the option selected by the user will be marked in green. In all problem modes(Figure X), only the correct option is marked in pink. In this activity, the current problem can also be shared as a text or bookmarked in the user's favorites. The wrong question will also be automatically added to the user's wrong question book.

2.5.10 Competition Start Activity

View the rules of the game, click on the picture to start the game(Jump to Competition Activity).

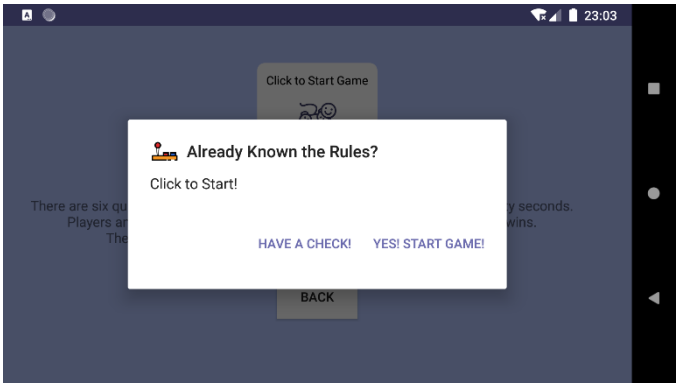


Figure XIII

2.5.11 Competition Activity

Figure XIV

In this activity, the interface will be divided into three containers. The left and right sides are the answer areas for red and blue. There are four option buttons placed in the container. The middle part is divided into upper and lower sub-parts, the upper part is used to display the current score and the countdown of each round, and the lower part is used to display the details of the current question.

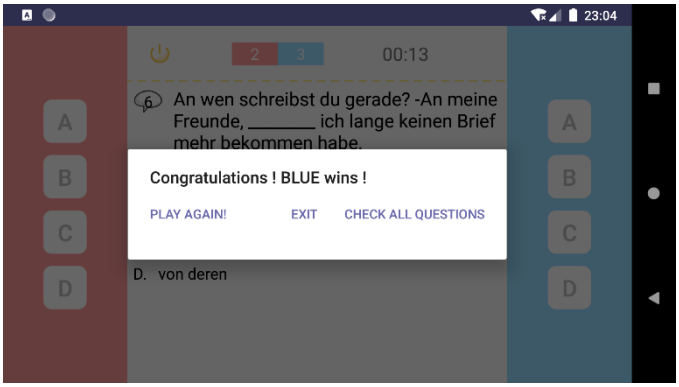


Figure XV

Each game six questions will be randomly selected from the database, and the red and blue sides will take turns to answer the questions. Each time a participant answers a question, the application will determine whether the question has been answered correctly and update the scores of both parties in real time. When all questions have been answered, the score will be used to determine which side wins(Prompt via AlertDialog).

After the game is over, the user can choose to view the questions answered (jump to Wrong Mode Activity), play again (wait for a second and re-run), or exit (return to the main activity).

2.5.12 Question Book Activity

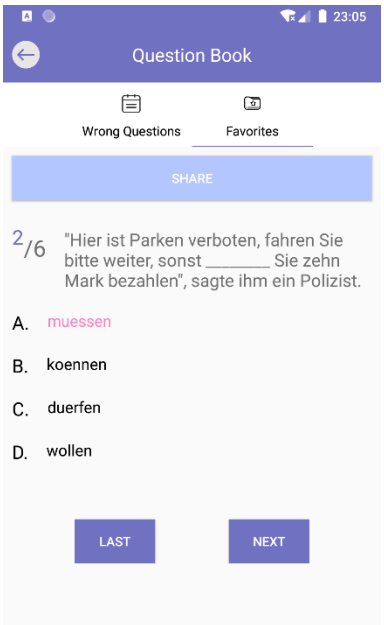


Figure XVI

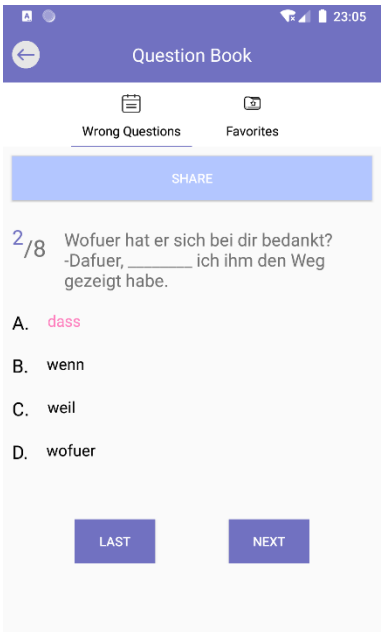


Figure XVII

In this activity, through the combination of TabLayout, TabItem, ViewPager and Fragment, the sliding switch between the wrong book and the favorites was implemented. Other functions are similar with Wrong Mode Activity.

TabPagerAdapter

```
public TabPagerAdapter(FragmentManager fm, Context context, MySQLite mySQLite) {
    super(fm);
    this.context=context;
    this.mySQLite = mySQLite;
}

@Override
public Fragment getItem(int position) {
    List<Questions> wrongList = mySQLite.getWrongQuestions();
    List<Questions> collectList = mySQLite.getCollectQuestions();
    return PagerFragment.newInstance( page: position+1,wrongList,collectList);
}

@Override
public int getCount() { return PAGE_COUNT; }
```

Set up Adapter and View Pager

```
tabLayout = findViewById(R.id.statistics_tabLayout);
viewPager = findViewById(R.id.statistics_view_pager);
tabAdapter = new TabPagerAdapter(getSupportFragmentManager(), context: this,mySQLite);
viewPager.setAdapter(tabAdapter);
tabLayout.setupWithViewPager(viewPager);
```

Get Instance of Fragment

```
public static PagerFragment newInstance(int page, List<Questions> wrongList,List<Questions> collectList){
    Bundle bundle=new Bundle();
    bundle.putInt(ARG_PAGE,page);
    bundle.putSerializable("wrongList",(Serializable)wrongList);
    bundle.putSerializable("collectList",(Serializable)collectList);
    PagerFragment pageFragment=new PagerFragment();
    pageFragment.setArguments(bundle);
    return pageFragment;
}
```

Set Tab View(Default is the first Tab)

```
for (int i = 0; i < 2; i++) {

    TabLayout.Tab tab = tabLayout.getTabAt(i);
    //Customized View
    tab.setCustomView(tabAdapter.getCustomView(i));

    //
    if (i == 0) {
        (tab.getCustomView().findViewById(R.id.tab_iv)).setSelected(true);
        (tab.getCustomView().findViewById(R.id.tab_tv)).setSelected(true);
    }
}
```

Add On Tab Selected Listener

```
tabLayout.addOnTabSelectedListener(new TabLayout.OnTabSelectedListener() {
    @Override
    public void onTabSelected(TabLayout.Tab tab) {
        (tab.getCustomView().findViewById(R.id.tab_iv)).setSelected(true);
        (tab.getCustomView().findViewById(R.id.tab_tv)).setSelected(true);
        viewPager.setCurrentItem(tab.getPosition());
    }
    @Override
    public void onTabUnselected(TabLayout.Tab tab) {
        (tab.getCustomView().findViewById(R.id.tab_iv)).setSelected(false);
        (tab.getCustomView().findViewById(R.id.tab_tv)).setSelected(false);
    }
    @Override
    public void onTabReselected(TabLayout.Tab tab) {

    }
});
```

2.5.13 Timer Activity

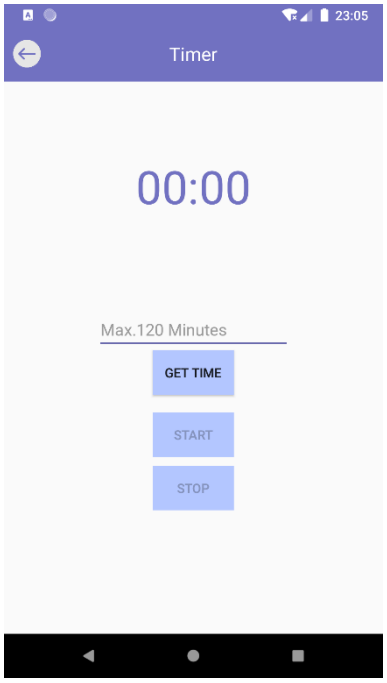


Figure XIX

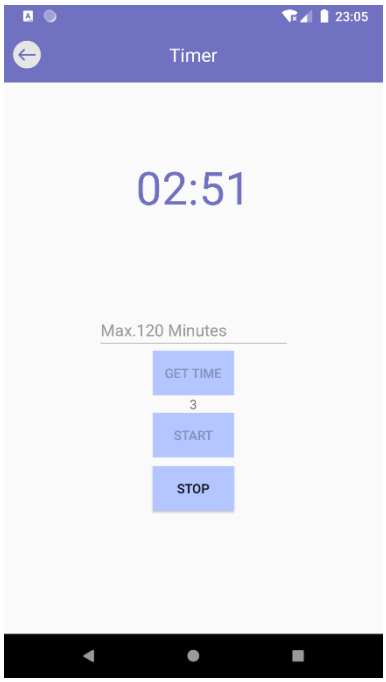


Figure XVIII

This Activity uses a countdown component to implement a simple Pomodoro.

The user can enter the number of minutes in EditText (more than 120 or less than or equal to 0, an error message "Invalid Input" will appear), click the "Get Time" button to get the time, and the "Start" button to start the countdown. The "Stop" button is only available after the countdown starts. Click the button to obtain the time without typing any number, then an error message will appear.

2.5.14 Achievement Activity

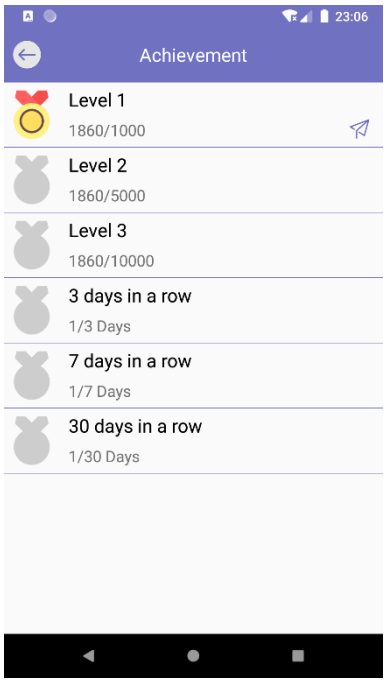


Figure XX

After meeting certain conditions, achievements will be unlocked. Achievements can be shared in text form. More achievements and bonus have not been added.

3 Summary

One of the biggest problems in developing this application is that before the project started, there was only a general idea of the function of the entire app in mind, without systematic planning. This caused the entire process to be very messy, and wasted a large portion of time at the beginning of the project to rewrite.

The second question is about notification. In fact, so far, this problem has not been solved. In the process of debugging on our own device, due to the API version of the device (API 23), channel ID (minSDK 26) is not available. If you do not add a Channel for notification, the notification will not be triggered.

The last problem is about our original motivation to develop this application. We want to increase our interest in learning German by implementing some special achievements. However, due to the demand for art designing, we were unable to complete this part for the time being, so in this project, only a German learning app that could meet our own needs was first developed.

The gain in this project is to try many components that have not been used in the exercise sheet, such as TabLayout, ViewPager and so on. We also try to customize UI components, use codes to draw graphics, calendars, etc.

We also want to try more components and classes in the future, such as maps and some animations.