INTI International College Penang School of Engineering and Technology 3+0 Bachelor of Science (Hons) in Computer Science, in collaboration with Coventry University, UK 3+0 Bachelor of Science (Hons) in Computing, in collaboration with Coventry University, UK

Coursework cover sheet

Section A - To be completed by the student

Full Name: Alwyn Chua Zheng Feng, Nicholas Kong Whey Kit, Lim Ting Feng	
CU Student ID Number: 13459835, 13459961, 13459950	
Semester: Aug 2022	
Lecturer: Mdm. Koo Lee Chun	
Module Code and Title: A202SGI – Android Development Skills	
Assignment No. / Title: Coursework	% of Module Mark 100%
Hand out date: 14 Sept 2022(CT1/CT2) 15 Sept 2022 (CS)	Due date: Final Project - 21 November 2022 VIVA : 23 Nov and onwards Note: Each group to submit a project title/proposal for approval by 25 September 2022
Penalties: No late work will be accepted. If you are unable to submit coursework on time due to extenuating circumstances you may be eligible for an extension. Please consult the lecturer.	

Declaration: I/we the undersigned confirm that I/we have read and agree to abide by the University regulations on plagiarism and cheating and Faculty coursework policies and procedures. I/we confirm that this piece of work is my/our own. I/we consent to appropriate storage of our work for plagiarism checking.

chua

Signature(s): --------------------------------------

kong

Signature(s): --------------------------------------

Lim

Signature(s): --------------------------------------

Section B - To be completed by the module leader

Intended learning outcomes assessed by this work:

1. Demonstrate familiarity with the Java Programming language and the Android Studio IDE.
2. Design applications suitable for Android devices.
3. Use the Android software development kit and emulator to develop applications for the Android platform.
4. Make use of the main modes of interaction available on a smartphone platform.

Marking scheme	Max	Mark
1. App design	20	
2. Interface	20	
3. Database	20	
4. Coding	20	
5. Report Writing	20	
Total	100	

Lecturer's Feedback

Internal Moderator's Feedback

Github Link: https://github.com/AlwynChua/A202SGI-Group-Project.git

Youtube URL:
https://www.youtube.com/watch?v=oi4D93X1DvE

Table of Contents

1.0 Introduction

LBM is a library management system which allows library employees to manage library books through a smartphone.

For our database, we have chosen to use SQLite. SQLite is a software library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. You can link it statically or dynamically as per your requirement with your application and accesses its storage files directly, unlike other databases, SQLite is not a standalone process.

The purpose of LBM is to have a better management of a library system and also to reduce manual work for managing information regarding users and books.

With our system, library employees can use their smartphones to keep track of the current status of books in the library, manage users' borrowings and add, update and delete books with their IDs to the current book list.

2.0 Application Output and Code

2.1 Login Screen

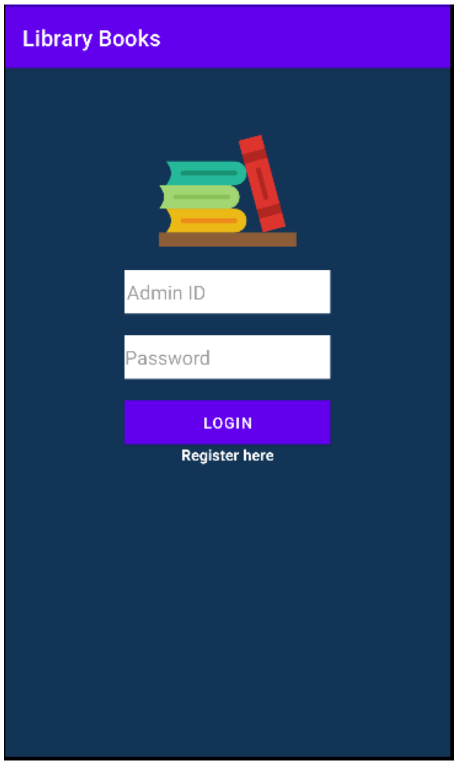
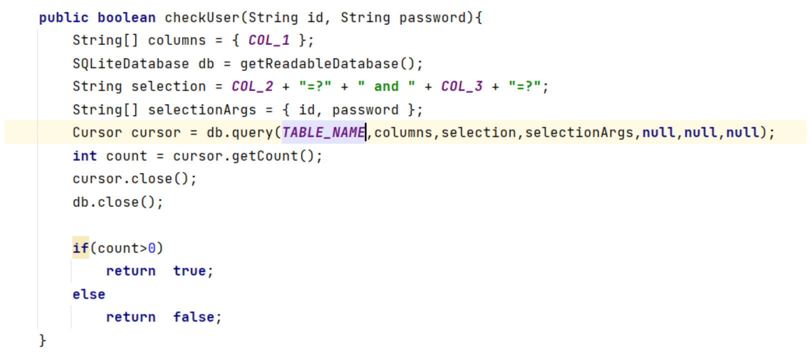


Figure 2.1.1 Login Activity

When the application is run, the first thing users will see is a login screen, there are the editTexts where users can input admin id and password and click on the Login button, if text boxes are empty or the system is not able to find existing IDs and password, the login will fail. Another button is the register here button which will intent to another activity called RegisterActivity.java.



```java
public boolean checkUser(String id, String password){
    String[] columns = { COL_1 };
    SQLiteDatabase db = getReadableDatabase();
    String selection = COL_2 + "=?" + " and " + COL_3 + "=?";
    String[] selectionArgs = { id, password };
    Cursor cursor = db.query(TABLE_NAME,columns,selection,selectionArgs,null,null,null);
    int count = cursor.getCount();
    cursor.close();
    db.close();

    if(count>0)
        return  true;
    else
        return  false;
}
```

Figure 2.1.2 LoginDatabase checkUser method

2.2 Registration Screen

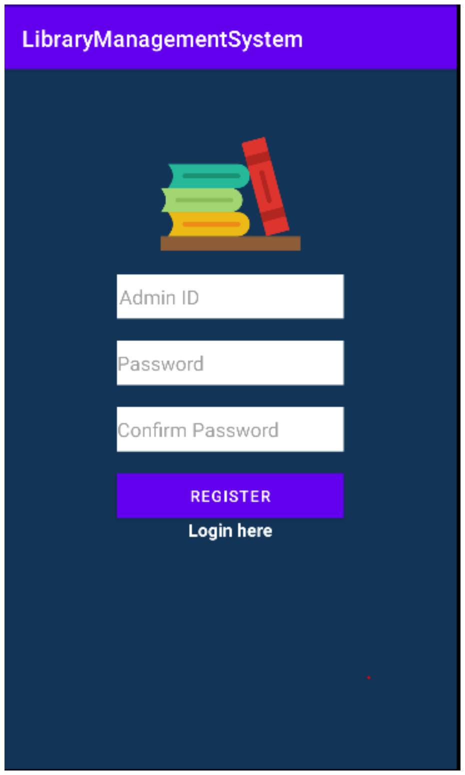


Figure 2.2.1 Registration Activity

In the registration screen, much like login screen, it has almost the same layout, only different inputs. The textbox will have to be entered by users like admin id, password, and confirm password, the fields cannot be empty, and the passwords have to match or it will not proceed when the Register button is clicked on. If no error occurs, it will insert the inputs given into the SQLite database, where back in the Login Screen, users can then use those same inputs to login.

```
public long addUser(String id, String password){
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues contentValues = new ContentValues();
    contentValues.put("studentid",id);
    contentValues.put("password",password);
    long res = db.insert("registeruser",null,contentValues);
    db.close();
    return  res;
}
```

Figure 2.2.2 LoginDatabase addUser method

2.3 Menu

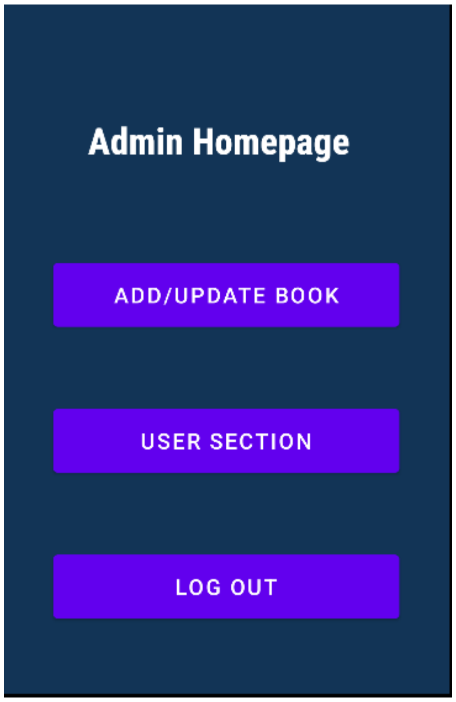
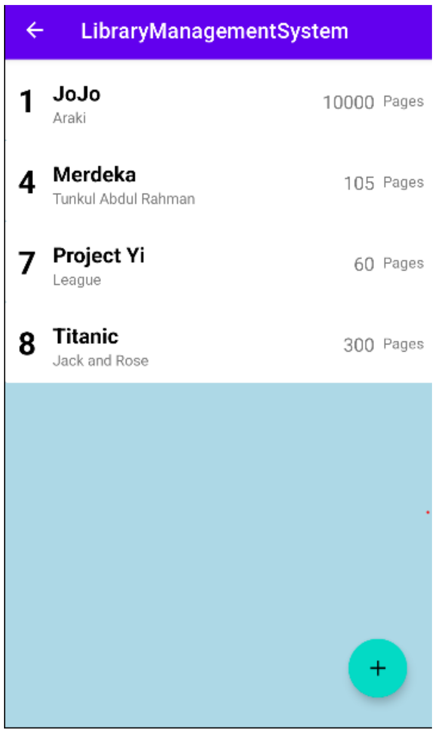


Figure 2.3.1 Menu Activity

After successfully logging in, the users can then choose from the 3 buttons. The add/update book button if clicked, will intent over to ViewBooks activity. The User Section button will intent over to ViewBorrowedBooks intent. The log out button will log out and intent the user back to login screen.

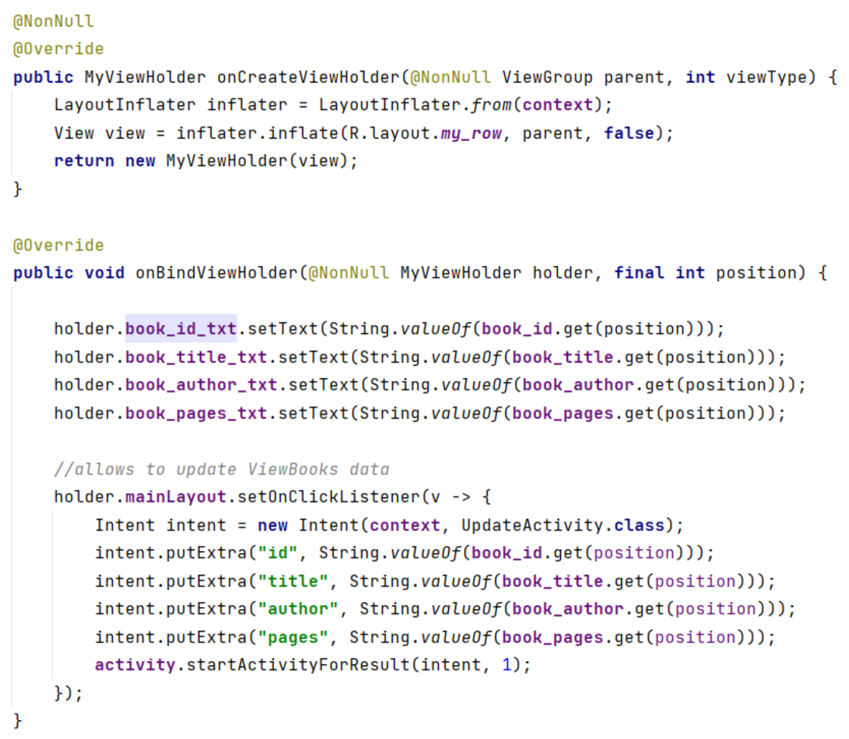
2.4 ViewBooks Activity


LibraryManagementSystem

1	JoJo Araki	10000	Pages
4	Merdeka Tunkul Abdul Rahman	105	Pages
7	Project Yi League	60	Pages
8	Titanic Jack and Rose	300	Pages

Figure 2.4.1 ViewBooks activity

The ViewBooks activity will display books information in a recyclerView. It is related to another activity called CustomAdapter, this is an adapter class that set which has to be displayed to the user in recyclerView. It is like the main responsible class to bind the views and display it. ViewHolder is another type of a helper class that helps us to draw the UI for individual items that we want to draw on the screen. LayoutManager in recyclerView helps us to figure out how we need to display the items on the screen. It can be linearly or in a grid. RecyclerView provides by default a few implementations of layoutManager out of the box.



```java
@NonNull
@Override
public MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
    LayoutInflater inflater = LayoutInflater.from(context);
    View view = inflater.inflate(R.layout.my_row, parent, false);
    return new MyViewHolder(view);
}

@Override
public void onBindViewHolder(@NonNull MyViewHolder holder, final int position) {

    holder.book_id_txt.setText(String.valueOf(book_id.get(position)));
    holder.book_title_txt.setText(String.valueOf(book_title.get(position)));
    holder.book_author_txt.setText(String.valueOf(book_author.get(position)));
    holder.book_pages_txt.setText(String.valueOf(book_pages.get(position)));

    //allows to update ViewBooks data
    holder.mainLayout.setOnClickListener(v -> {
        Intent intent = new Intent(context, UpdateActivity.class);
        intent.putExtra("id", String.valueOf(book_id.get(position)));
        intent.putExtra("title", String.valueOf(book_title.get(position)));
        intent.putExtra("author", String.valueOf(book_author.get(position)));
        intent.putExtra("pages", String.valueOf(book_pages.get(position)));
        activity.startActivityForResult(intent, 1);
    });
}
```

Figure 2.4.2 viewHolder

2.4.3 AddActivity activity

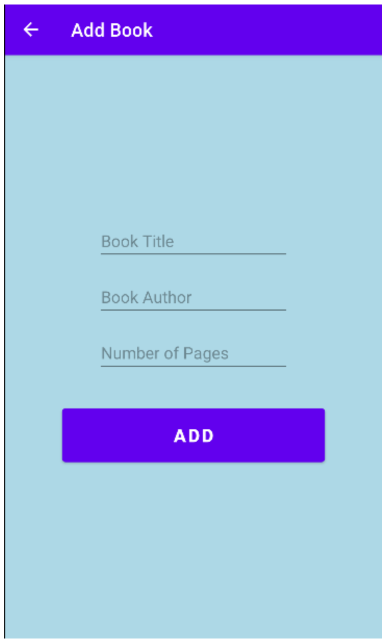
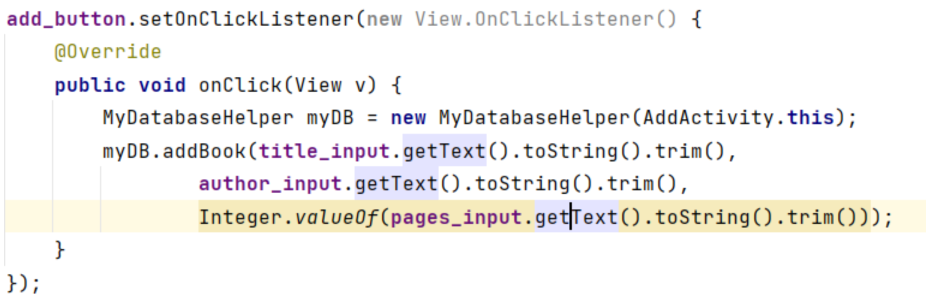
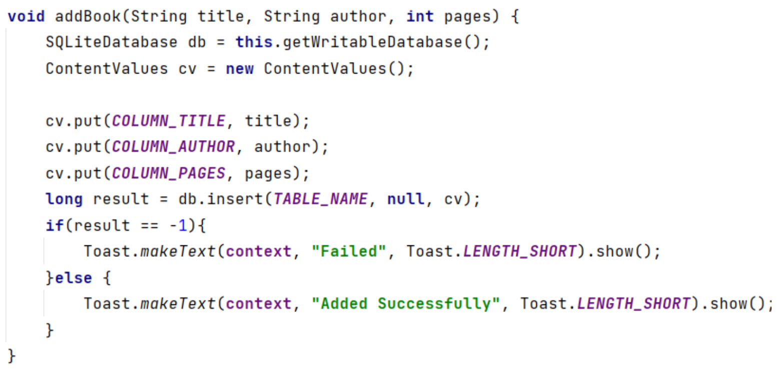


Figure 2.4.4 AddActivity

After clicking on the icon on the bottom right of ViewBooks, it will intent to an activity called AddActivity, here the library staff can add new books by typing in the inputs and clicking on the Add button will insert the info on the 3 textboxes into the SQLite database table. It will then read all the data in the SQLite table then store it in the array for the recyclerView to display.



```java
add_button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        MyDatabaseHelper myDB = new MyDatabaseHelper(AddActivity.this);
        myDB.addBook(title_input.getText().toString().trim(),
                author_input.getText().toString().trim(),
                Integer.valueOf(pages_input.getText().toString().trim()));
    }
});
```



```java
void addBook(String title, String author, int pages) {
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues cv = new ContentValues();

    cv.put(COLUMN_TITLE, title);
    cv.put(COLUMN_AUTHOR, author);
    cv.put(COLUMN_PAGES, pages);
    long result = db.insert(TABLE_NAME, null, cv);
    if(result == -1){
        Toast.makeText(context, "Failed", Toast.LENGTH_SHORT).show();
    }else {
        Toast.makeText(context, "Added Successfully", Toast.LENGTH_SHORT).show();
    }
}
```

2.4.5 Update/Delete Activity

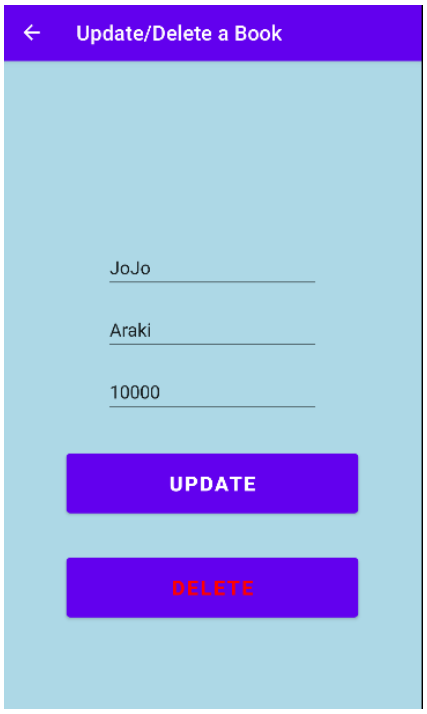


Figure 2.4.6

By clicking on one of the rows in ViewBooks, the user will go go into UpdateActivity, here, the textboxes will be data intent from the previous page, filling in the data. The user can then choose to either update or delete. For update users can just change the data on the editTexts,, then click update, this will then call the updateData method and updating the table. Clicking on Delete button will call the deleteOneRow method and delete the row from the table.

```java
void updateData(String row_id, String title, String author, String pages){
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues cv = new ContentValues();
    cv.put(COLUMN_TITLE, title);
    cv.put(COLUMN_AUTHOR, author);
    cv.put(COLUMN_PAGES, pages);

    long result = db.update(TABLE_NAME, cv, "_id=?", new String[]{row_id});
    if(result == -1){
        Toast.makeText(context, "Failed to Update", Toast.LENGTH_SHORT).show();
    }else{
        Toast.makeText(context, "Successfully Updated", Toast.LENGTH_SHORT).show();
    }
}

void deleteOneRow(String row_id ) {
    SQLiteDatabase db = this.getWritableDatabase();
    long result = db.delete(TABLE_NAME, "_id=?", new String[]{row_id});
    if(result == -1){
        Toast.makeText(context, "Failed to Delete", Toast.LENGTH_SHORT).show();
    }else{
        Toast.makeText(context, "Successfully Deleted", Toast.LENGTH_SHORT).show();
    }
}
```

2.5 ViewBorrowedBooks

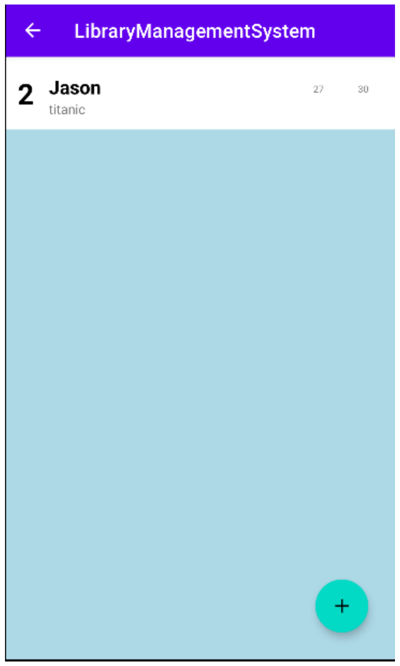
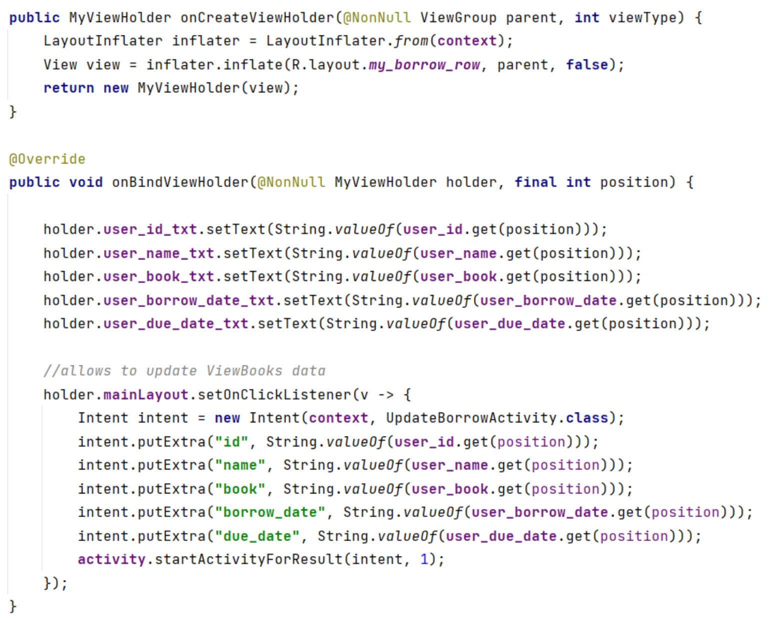


Figure 2.5.1 ViewBorrowedBooks

This activity is basically the same ViewBooks, only difference is it displays lists of people that have borrowed books from the library, with name, book titles, borrow date and return date. And as the same as ViewBooks, the bottom right button leads to AddBorrowedBooks.



```java
public MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
    LayoutInflater inflater = LayoutInflater.from(context);
    View view = inflater.inflate(R.layout.my_borrow_row, parent, false);
    return new MyViewHolder(view);
}

@Override
public void onBindViewHolder(@NonNull MyViewHolder holder, final int position) {

    holder.user_id_txt.setText(String.valueOf(user_id.get(position)));
    holder.user_name_txt.setText(String.valueOf(user_name.get(position)));
    holder.user_book_txt.setText(String.valueOf(user_book.get(position)));
    holder.user_borrow_date_txt.setText(String.valueOf(user_borrow_date.get(position)));
    holder.user_due_date_txt.setText(String.valueOf(user_due_date.get(position)));

    //allows to update ViewBooks data
    holder.mainLayout.setOnClickListener(v -> {
        Intent intent = new Intent(context, UpdateBorrowActivity.class);
        intent.putExtra("id", String.valueOf(user_id.get(position)));
        intent.putExtra("name", String.valueOf(user_name.get(position)));
        intent.putExtra("book", String.valueOf(user_book.get(position)));
        intent.putExtra("borrow_date", String.valueOf(user_borrow_date.get(position)));
        intent.putExtra("due_date", String.valueOf(user_due_date.get(position)));
        activity.startActivityForResult(intent, 1);
    });
}
```

2.6 AddBorrowedBooks

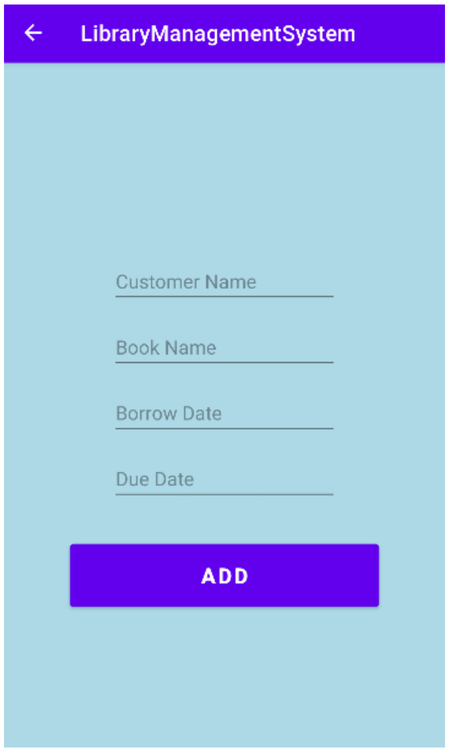
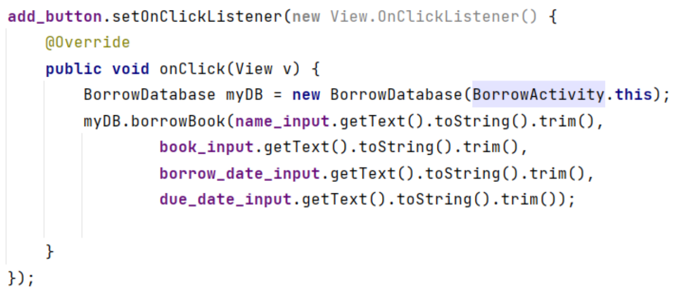
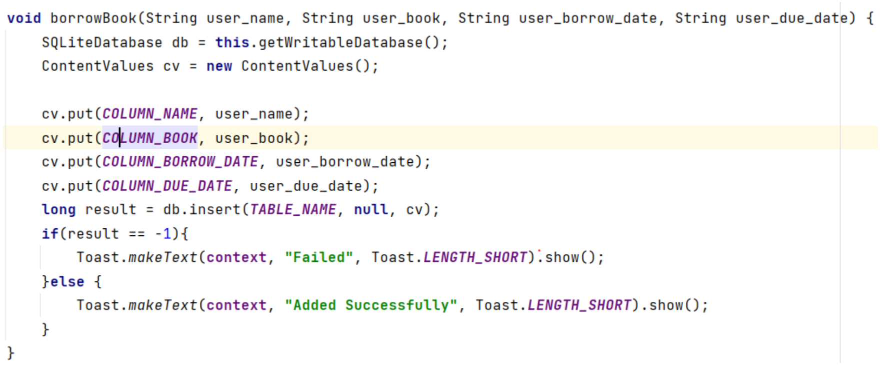


Figure 2.6.1 AddBorrowedBooks

Same structures as AddBooks activity, contains editTexts, and an Add button, users will need to input information and click Add button to insert data into SQLite database table.

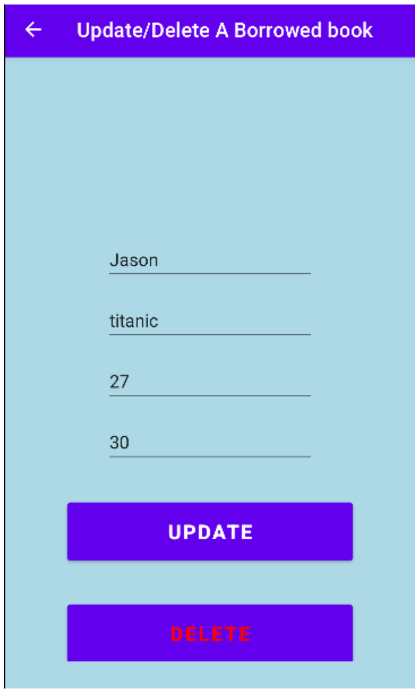


```
add_button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        BorrowDatabase myDB = new BorrowDatabase(BorrowActivity.this);
        myDB.borrowBook(name_input.getText().toString().trim(),
                book_input.getText().toString().trim(),
                borrow_date_input.getText().toString().trim(),
                due_date_input.getText().toString().trim());

    }
});
```



```
void borrowBook(String user_name, String user_book, String user_borrow_date, String user_due_date) {
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues cv = new ContentValues();

    cv.put(COLUMN_NAME, user_name);
    cv.put(COLUMN_BOOK, user_book);
    cv.put(COLUMN_BORROW_DATE, user_borrow_date);
    cv.put(COLUMN_DUE_DATE, user_due_date);
    long result = db.insert(TABLE_NAME, null, cv);
    if(result == -1){
        Toast.makeText(context, "Failed", Toast.LENGTH_SHORT).show();
    }else {
        Toast.makeText(context, "Added Successfully", Toast.LENGTH_SHORT).show();
    }
}
```

2.7 UpdateBorrowActivity



Jason

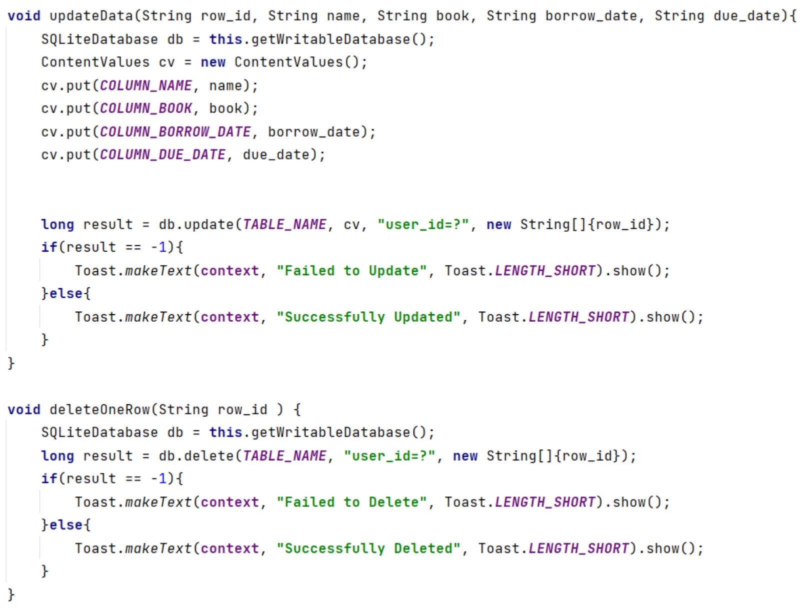
titanic

27

30

Figure 2.7.1 UpdateBorrowActivity

Same like UpdateActivity, only different variables and different database and tables. Users can click on the row they want to update, they will intent to this screen, where they change the data in the editTexts, and click Update button to update the database table. Users can also click on Delete button and delete the row from the database table.



```
void updateData(String row_id, String name, String book, String borrow_date, String due_date){
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues cv = new ContentValues();
    cv.put(COLUMN_NAME, name);
    cv.put(COLUMN_BOOK, book);
    cv.put(COLUMN_BORROW_DATE, borrow_date);
    cv.put(COLUMN_DUE_DATE, due_date);


    long result = db.update(TABLE_NAME, cv, "user_id=?", new String[]{row_id});
    if(result == -1){
        Toast.makeText(context, "Failed to Update", Toast.LENGTH_SHORT).show();
    }else{
        Toast.makeText(context, "Successfully Updated", Toast.LENGTH_SHORT).show();
    }
}

void deleteOneRow(String row_id ) {
    SQLiteDatabase db = this.getWritableDatabase();
    long result = db.delete(TABLE_NAME, "user_id=?", new String[]{row_id});
    if(result == -1){
        Toast.makeText(context, "Failed to Delete", Toast.LENGTH_SHORT).show();
    }else{
        Toast.makeText(context, "Successfully Deleted", Toast.LENGTH_SHORT).show();
    }
}
```

3.0 Weakness and Issues

When we making this project, we encounter the database problem. At the start we were planning to use the firebase because it is a cloud database and is harder to complement to the project so we choose SQLite database instead.

The project does not have a user side, making the user need to borrow books at the physical library instead of online. The app has no secured login making anyone can use the app without any authentication since the user can just register and login on site.

4.0 Reflection

Nicholas Kong

First of all, due to a huge load of other assignments, we did not really manage to make full use of our time in this project. We didn't had time to implement the new skills we have learnt in class such as fragment, which has its own lifecycle which can handle its own layout and input events.

However, this project made me recalled some of the skills that I have learnt before, such as intent, designing the xml file, recyclerView and so on. I was able to implement those skills on this project.

Finally, I have discovered how to manage my time more effectively so that everything will not be put on last minute in the future. As a team member of the group, I think we did everything in order as we were constantly communicating and giving feedback. We also had video calls to share our progress and problems faced.

Lim Ting Feng

I had learnt a lot how to use the android studio to create an application. When we encounter problem in the project, my team member and I discussed together and tried to solved the problem. I think the biggest challenge that we facing is we need to self-learn and researching online some knowledge from the Internet in a short amount of time. I think that working together as a team is important since helping each other will make the outcome more productive.

Alwyn Chua

First of all, I want to thank both of my group members for being cooperative throughout this project, they are responsive and active when doing this project and they will reply to me when in need of assistance.

With that out of the way, the project itself is not too difficult to the point that we cannot finish it, it's the fact that there other projects in other module pulling me down in terms of effectiveness, I had to prioritize the ones that have closer due date, and because of that this project is the least effort put into it. For our initial plan, we were going to try and implement Firebase, a cloud database, but that would take a lot of time to do and we still have other 3 projects that require coding. That's why we decided to go with SQLite instead, but it isn't very effective in terms of functionality, it still get the job done, but it really cant be compared to apps in google play.

I did learned a lot of things in class, as I would like to thank Ms Koo for being our lecturer for this module. I get another chance to learn android studio, new things and some old things too, like recyclerview and adapter, which I didn't really get how it works, back in Diploma. But what I learned most of all is to manage my time properly.

5.0 Conclusion

Through this project, we have a better understanding on Android Studio. Some of us has learned Android Studio before, but this project allowed us to take it to another level. Even though the result of our project is mediocre, we worked together and pulled it off and made a functional android project. We would have calls to discuss what part would work, and how to solve issues. After this project, we grow in knowledge, and also mentality.

6.0 References

1. Anupam Chugh, 2022, Android SQLite Database Example Tutorial
https://www.digitalocean.com/community/tutorials/android-sqlite-database-example-tutorial

2. Himanshu Singh, 20th May 2020, How does RecyclerView work internally?

https://blog.mindorks.com/how-does-recyclerview-work-internally