Nama : Rahmad Nurhuda

NIM : 14116116

TUGAS PRAKTIKUM #5

Pengembangan Aplikasi Mobile

1. **Activity_main.xml**

```xml
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="16dp"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/instructions"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/instructions"
        android:textAppearance="@style/TextAppearance.AppCompat.Title"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"/>

    <EditText
        android:id="@+id/bookInput"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="8dp"
        android:hint="@string/input_hint"
        android:inputType="text"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/instructions"/>

    <Button
        android:id="@+id/searchButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="8dp"
```

```xml
<Button
    android:id="@+id/searchButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:onClick="searchBooks"
    android:text="@string/button_text"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/bookInput"/>

<TextView
    android:id="@+id/titleText"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="16dp"
    android:textAppearance="@style/TextAppearance.AppCompat.Headline"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/searchButton"/>

<TextView
    android:id="@+id/authorText"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:textAppearance="@style/TextAppearance.AppCompat.Headline"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/titleText"/>

android.support.constraint.ConstraintLayout>
```

2. MainActivity.java

```java
public class MainActivity extends AppCompatActivity
        implements LoaderManager.LoaderCallbacks<String> {

    private EditText mBookInput;
    private TextView mTitleText;
    private TextView mAuthorText;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        mBookInput = findViewById(R.id.bookInput);
        mTitleText = findViewById(R.id.titleText);
        mAuthorText = findViewById(R.id.authorText);

        if (getSupportLoaderManager().getLoader(0) != null) {
            getSupportLoaderManager().initLoader(0, null, this);
        }
    }

    /**
     * onClick handler for the "Search Books" button.
     *
     * @param view The view (Button) that was clicked.
     */
    public void searchBooks(View view) {
        // Get the search string from the input field.
        String queryString = mBookInput.getText().toString();

        // Hide the keyboard when the button is pushed.
        InputMethodManager inputManager = (InputMethodManager)
                getSystemService(Context.INPUT_METHOD_SERVICE);
        if (inputManager != null) {
```

```java
                InputMethodManager.HIDE_NOT_ALWAYS);
    }

    // Check the status of the network connection.
    ConnectivityManager connMgr = (ConnectivityManager)
            getSystemService(Context.CONNECTIVITY_SERVICE);
    NetworkInfo networkInfo = null;
    if (connMgr != null) {
        networkInfo = connMgr.getActiveNetworkInfo();
    }

    // If the network is available, connected, and the search field
    // is not empty, start a BookLoader AsyncTask.
    if (networkInfo != null && networkInfo.isConnected()
            && queryString.length() != 0) {

        Bundle queryBundle = new Bundle();
        queryBundle.putString("queryString", queryString);
        getSupportLoaderManager().restartLoader(0, queryBundle, this);

        mAuthorText.setText("");
        mTitleText.setText(R.string.loading);
    }
    // Otherwise update the TextView to tell the user there is no
    // connection, or no search term.
    else {
        if (queryString.length() == 0) {
            mAuthorText.setText("");
            mTitleText.setText(R.string.no_search_term);
        } else {
            mAuthorText.setText("");
            mTitleText.setText(R.string.no_network);
        }
    }
}
```

```java
@NonNull
@Override
public Loader<String> onCreateLoader(int id, @Nullable Bundle args) {
    String queryString = "";

    if (args != null) {
        queryString = args.getString("queryString");
    }

    return new BookLoader(this, queryString);
}

@Override
public void onLoadFinished(@NonNull Loader<String> loader, String data) {
    try {
        // Convert the response into a JSON object.
        JSONObject jsonObject = new JSONObject(data);
        // Get the JSONArray of book items.
        JSONArray itemsArray = jsonObject.getJSONArray("items");

        // Initialize iterator and results fields.
        int i = 0;
        String title = null;
        String authors = null;

        // Look for results in the items array, exiting when both the
        // title and author are found or when all items have been checked.
        while (i < itemsArray.length() &&
                (authors == null && title == null)) {
            // Get the current item information.
            JSONObject book = itemsArray.getJSONObject(i);
            JSONObject volumeInfo = book.getJSONObject("volumeInfo");
```

```java
                    authors = volumeInfo.getString("authors");
                } catch (JSONException e) {
                    e.printStackTrace();
                }

                // Move to the next item.
                i++;
            }

            // If both are found, display the result.
            if (title != null && authors != null) {
                mTitleText.setText(title);
                mAuthorText.setText(authors);
                //mBookInput.setText("");
            } else {
                // If none are found, update the UI to show failed results.
                mTitleText.setText(R.string.no_results);
                mAuthorText.setText("");
            }

        } catch (Exception e) {
            // If onPostExecute does not receive a proper JSON string,
            // update the UI to show failed results.
            mTitleText.setText(R.string.no_results);
            mAuthorText.setText("");
            e.printStackTrace();
        }

    }

    @Override
    public void onLoaderReset(@NonNull Loader<String> loader) {
        // Do nothing.  Required by interface.
    }
```

3. NetworkUtils.java

```java
public class NetworkUtils {

    private static final String LOG_TAG = NetworkUtils.class.getSimpleName();

    // Constants for the various components of the Books API request.
    //
    // Base endpoint URL for the Books API.
    private static final String BOOK_BASE_URL =
            "https://www.googleapis.com/books/v1/volumes?";
    // Parameter for the search string.
    private static final String QUERY_PARAM = "q";
    // Parameter that limits search results.
    private static final String MAX_RESULTS = "maxResults";
    // Parameter to filter by print type.
    private static final String PRINT_TYPE = "printType";


    /**
     * Static method to make the actual query to the Books API.
     *
     * @param queryString the query string.
     * @return the JSON response string from the query.
     */
    static String getBookInfo(String queryString) {

        // Set up variables for the try block that need to be closed in the
        // finally block.
        HttpURLConnection urlConnection = null;
        BufferedReader reader = null;
        String bookJSONString = null;

        try {
```

```java
Uri builtURI = Uri.parse(BOOK_BASE_URL).buildUpon()
        .appendQueryParameter(QUERY_PARAM, queryString)
        .appendQueryParameter(MAX_RESULTS, "10")
        .appendQueryParameter(PRINT_TYPE, "books")
        .build();

// Convert the URI to a URL.
URL requestURL = new URL(builtURI.toString());

// Open the network connection.
urlConnection = (HttpURLConnection) requestURL.openConnection();
urlConnection.setRequestMethod("GET");
urlConnection.connect();

// Get the InputStream.
InputStream inputStream = urlConnection.getInputStream();

// Create a buffered reader from that input stream.
reader = new BufferedReader(new InputStreamReader(inputStream));

// Use a StringBuilder to hold the incoming response.
StringBuilder builder = new StringBuilder();

String line;
while ((line = reader.readLine()) != null) {
    // Add the current line to the string.
    builder.append(line);

    // Since this is JSON, adding a newline isn't necessary (it won't
    // affect parsing) but it does make debugging a *lot* easier
    // if you print out the completed buffer for debugging.
    builder.append("\n");
}

if (builder.length() == 0) {
```

```
            return null;
        }

        bookJSONString = builder.toString();

    } catch (IOException e) {
        e.printStackTrace();
    } finally {
        // Close the connection and the buffered reader.
        if (urlConnection != null) {
            urlConnection.disconnect();
        }
        if (reader != null) {
            try {
                reader.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

    // Write the final JSON response to the log
    Log.d(LOG_TAG, bookJSONString);

    return bookJSONString;
```

4. BookLoader.java

```java
public class BookLoader extends AsyncTaskLoader<String> {

    private String mQueryString;

    BookLoader(Context context, String queryString) {
        super(context);
        mQueryString = queryString;
    }

    @Override
    protected void onStartLoading() {
        super.onStartLoading();

        forceLoad();
    }

    @Nullable
    @Override
    public String loadInBackground() { return NetworkUtils.getBookInfo(mQueryString); }
}
```
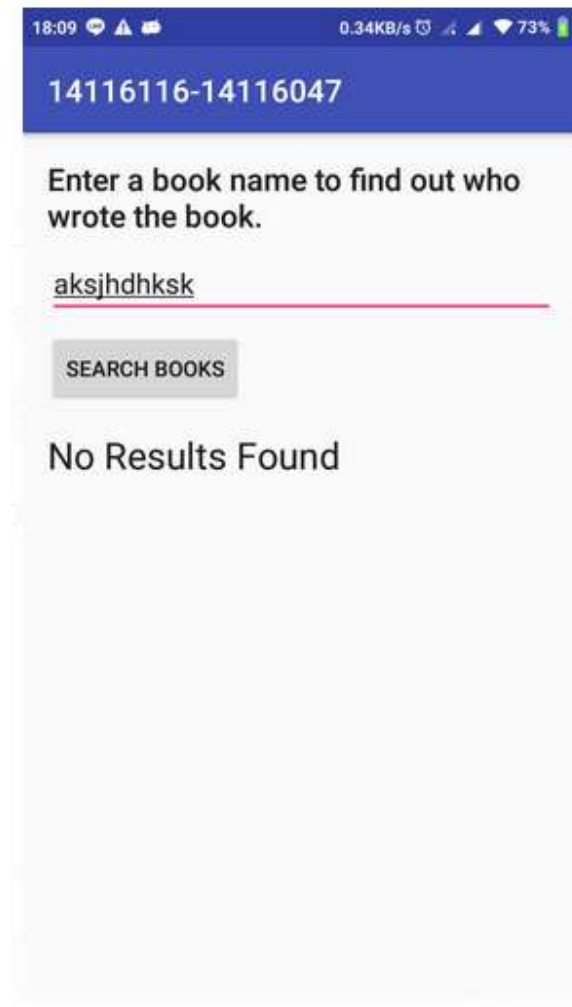
Screenshoot Hasil

14116116-14116047

Enter a book name to find out who wrote the book.

huda

SEARCH BOOKS

Lembaga Keuangan Islam

["Nurul Huda","Mohamad Heykal"]

14116116-14116047

Enter a book name to find out who wrote the book.

dimas

SEARCH BOOKS

Jalan yang Dituju Dimas

["Fajar A_a"]

14116116-14116047

Enter a book name to find out who wrote the book.

Book Title

SEARCH BOOKS