Assignment # 02
SQLiteDatabase – GroceryApp
Due: Monday, August 14, 2023, 11:59 PM
Maximum Marks: 10

In this assignment, you have to integrate an SQLite database into your app. You can customize your application to how you see fit. The only stipulation is that you must use an SQLite database to store your information. This is an individual assignment.

You must read this document carefully and follow all the instructions.

Scenario:
You currently work for the mobile app development company **AppForAll** and you have been tasked by your manager to create a custom app that will be used by groceries to keep track of their items. Incorporate Database, Tables and CRUD operations using SQLiteOpenHelper class. This document will serve as a requirements document for the necessary functionality for this app.

Functional Requirements:
The following functional requirements must be implemented to complete the assignment. Marking sheet is provided at the end of this document.

Before you begin:
- Make yourself familiarize with the hands-on examples done in the lab during class time.
- Make sure to fulfill all programming and assignment standards. The Standards Summary is available at eConestoga.
- See HRPrima App hands-on program in details.

Task 1: Create Assignment 2 Project
- Create an empty views activity type Android project in Android Studio and name it: XXGrocery (where XX is your initials).

Task 2: Create database and tables
- You will create an SQLite database 'XXGrocery.db' (where XX is your initials).
- Create the below tables in this database:
  - **User** (username Text ,emailId Text, password Text)
  - **Stock** (itemCode Integer, itemName Text, qtyStock Integer, price Float, taxable Boolean). The itemCode field will be primary key auto generated.
  - **Sales** (orderNumber Integer, itemCode Integer, customerName Text, customerEmail Text, qtySold Integer, and dateOfSales Date). The orderNumber field will be primary key auto generated.
  - **Purchase** (invoiceNumber Integer, itemCode Integer, qtyPurchased Integer, and dateOfPurchase Date). The invoiceNumber field will be primary key auto generated.

  Note: **very important**: Please note that you can purchase or sell an item which has an entry in the stock.

Task 3: Create Login Screen
- Login screen with sign up option.
- New user signup details should be saved in the User table.
- A test user should be registered with the username as 'test', email id as 'test@gmail.com' and password as 'test'.
- Credentials should be validated for login from the User table.
- Upon successful login, the user should be navigated to the home page.
- Display an alert in a Dialog if the credentials entered are invalid.
- Use shared preferences for maintaining session.

Task 4: Create Home Page

- The username should be displayed at the top of the page as 'Welcome' xx where xx is the username of the user who is currently logged in
- Design a navigation drawer menu with the following options:
  - Add Stock
  - Sales
  - Purchase
  - Search Stock
  - List Stock
  - Log Out

- Each Menu option will load the corresponding screen as explained in the next tasks. **These screens must be implemented using fragments.**

Task 5: Add Stock

- Upon selecting the 'Add Stock' option from the menu, should load a screen for adding an item to the Grocery stock.
- This screen will have the following input fields and two buttons:
  - Item Name – EditText
  - Qty In Stock - EditText
  - Price – EditText
  - Taxable, NonTaxable – Two radio buttons
- A button titled 'Cancel'. Clicking this button should navigate back to the home page.
- A button titled 'Save'. Clicking this button should save the item details in the table 'Stock' after validation.
- Alerts should be displayed if the validation has been failed.
- Alert should be displayed to indicate that an item has been successfully added to the stock.

Task 6: Sales

- Upon selecting the 'Sales' option from the menu, should load a screen for selling an item to a customer
- Please note that you can sell an item only if it has an entry in the stock.
- This screen will have the following input fields and two buttons:
  - Item Code – editTextNumber
  - Customer Name – editText
  - Customer Email – editTextEmailAddress
  - Qty Sold – editTextNumber
  - Date of Sales – Date Picker
- Date of Sales should not be a future date.
- A button titled 'Cancel'. Clicking this button should navigate back to the home page.
- A button titled 'Submit'. Clicking this button should do the following:
  - The item code that you enter should be available in the stock. Display suitable alert if the item does not exist in the stock.
  - Item cannot be sold if the available quantity in the stock is 0 or the quantity sold is greater than the available quantity in the stock. Display suitable alert if there is not enough quantity in the stock.
  - Alerts should be displayed if the validation has been failed.
  - Save the sales details in the table 'Sales', if the validation has been successful.
  - Alert should be displayed to indicate that an item has been successfully sold.
  - The 'Stock' table should be updated to change the qtyStock of the item according to the qty sold. (qtyStock = qtyStock – qtySold)

Task 7: Purchase

- Upon selecting the 'Purchase Item' option from the menu, should load a screen for purchasing an item to the stock.
- Please note that you can purchase an item only if it has an entry in the stock.
- This screen will have the following input fields and two buttons:
  - Item Code – editTextNumber

- Qty Purchased – editTextNumber
- Date of Purchase – Date Picker

- Date of Purchase should not be a future date.
- A button titled 'Cancel'. Clicking this button should navigate back to the home page.
- A button titled 'Submit'. Clicking this button should do the following:
  - Alerts should be displayed if the validation has been failed.
  - Alerts should be displayed if the item does not exist in the stock.
  - Save the purchase details in the table 'Purchase' if the validation has been successful.
  - Alert should be displayed to indicate that an item has been successfully purchased.
  - The 'Stock' table should be updated to change the qtyStock of the item according to the qty purchased. (qtyStock = qtyStock + qtyPurchased)

Task 8: Search Item

- Upon selecting the 'Search Item' option from the menu, should load a screen for searching an item in the stock.
- This screen will have the following input field and two buttons:
  - Item Code– editTextNumber
  - Cancel – Button
  - Search – Button
- Clicking the 'Cancel' button should navigate back to the home page.
- Clicking the 'Search' button should do the following:
  - Search for the item id entered in the 'Stock' table, after validation.
  - If the item is found, display its details in a dialog including item name, qty in stock, and price.
  - If the item does not exist in the stock, display an alert message 'Item Not Found'.

Task 9: List Items

- Upon selecting the 'List Items' option from the menu, should load a screen for listing all items in the stock.
- This screen will have the following components:
  - RecyclerView
  - Cancel – Button
- List the details of all items in the stock in a RecyclerView as a scrollable view, when the page is loaded. The details should include item code, item name, qtyStock, and price.
- Clicking the 'Cancel' button should navigate back to the home page.

Task 10: Log Out

- Upon selecting the 'Log Out' option from the menu, user should be logged out from the application.
- User should be navigated back to 'Login' Page.
- Session should be cleared. (Hint: Clear shared preference)

Task 11: Implement the best practices and validations.

Task 12: Material Design Guideline Principles should be followed to design a good-looking UI.

Task 13: Implement Object Oriented Programming principles with at least one Java class in which encapsulation is applied. You are free to choose a Java class that is relevant in the application.

What to Submit:

- The entire application package as a zipped file.
- A **separate Word document** consisting of screen shots of the running application and the commented Java code. Commented code should be pasted as text. **Submission will not be considered for evaluation if commented code is pasted as an image and a grade of 0 (zero) will be awarded for the assignment. Submission without the Word document or zipped Word document will be awarded a grade of 0 (zero). The Word document should not be included in the zip of the application package.**

Marking Sheet
Total Marks: 10

Description	Marks Allocated	Marks Achieved
Database and tables created according to spec	1	
Login Page: • Page is designed as specified • User registration is implemented properly, and the record is saved to User table • User navigated to the home page if the login credentials are valid • Username is maintained in session • Validations are properly implemented	1	
Home Page: • Page is designed as specified • Username is displayed at the top of the page • Navigation drawer menu is designed as specified • Each menu option navigates to the related screen	1	
Add Stock Page: • Page is designed as specified using fragment • 'Save' button works as specified • 'Cancel' button works as specified • Validations are properly implemented	1	
Sales Page: • Page is designed as specified using fragment • 'Submit' button works as specified • 'Cancel' button works as specified • Validations are properly implemented	1	
Purchase Page: • Page is designed as specified using fragment • 'Submit' button works as specified • 'Cancel' button works as specified • Validations are properly implemented	1	
Search Item Page: • Page is designed as specified using fragment	1	

• 'Search' button works as specified • 'Cancel' button works as specified • Validations are properly implemented		
List Items Page: • Page is designed as specified using fragment • List of items are loaded into a RecyclerView as a scrollable view, when the page is loaded • 'Cancel' button works as specified	1	
Log Out: • Successfully logged out from the application and navigates back to login page • Session variables are cleared	1	
Implementation of OOP principles with at least one Java class having encapsulation applied	1	
Deduction		
Non submission of unzipped Word document file with all relevant screenshots and commented Java code pasted as text	100%	
ViewBinding not implemented	50%	
SQLiteOpenHelper class is not used for implementing CRUD operations	50%	
Fragments are not used for designing various screens that are loaded when the menu options are selected	75%	
Runtime errors	10% x --------	
Hardcoded text found in the UI	3%	
Hardcoded color found in the UI	3%	
Validations not implemented	1% x --------	
Material design guidelines not followed in the UI design	4%	
Assignment Standard (proper project name, submission docs name etc)	2% x --------	
Programming Standards	1% x --------	
Late Submission (softcopy)	-------- days	
Bugs (including requirements mentioned in this specification)	3-10% based on severity	
Total		

Late Penalty (Softcopy submission)

Days Late	Penalty %
1	5

2	10
3	20
4	40
5	80
6	100

Please note: How "Days Late" is calculated: Your assignment is due on Sunday 11:59pm. You are considered 1 day late if you submit on anytime on Monday (until 11:59PM). If you submit anytime on Tuesday (until 11:59PM). you are considered 2 days late.