**NOTE MAKING APP**

**A PROJECT REPORT**

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

***Under the Guidance of***

**Mrs. mEGHALI DAS**

***Project Carried Out At***

****

**Ardent Computech Pvt Ltd (An ISO 9001:2015 Certified)**

**SDF module-132, Sector - 3, Salt Lake City, Kolkata - 700 064**

***Submitted By***

**tanuj ghosh**

**CHANDAN SHAW**

**animesh kumar**

**PRASHONTYA kar**

**DIWASH kapil chettri**

**UMESH YADAV**

****

***(Note:* *All entries of the proforma of approval should be filled up with appropriate and complete information. Incomplete proforma of approval in any respect will be summarily rejected.)***

1. Title of the Project:  **NOTE MAKING APP**

2. Project Members: 1. CHANDAN SHAW 2. tanuj ghosh 3. animesh kumar

4. DIWASH kapil chettri 5. PRASHONTYA kar 6. umesh yadav

3. Name and Address of the Guide: MRS. MEGHALI DAS

Subject Matter Expert & Assistant Technical Head (.NET Domain) Ardent Computech Pvt Ltd (An ISO 9001:2015 Certified) Module 132, SDF Building, Sector 5, Salt Lake, Kolkata – 64

Ph.D**\***  M.Tech**\*** BE\***/**B.Tech.\*MCA**\*** M.Sc.**\***

Y

Y

Y

4. Educational Qualification of

the Guide:

5. Working / Training experience of the Guide: **2 Years**

6. Project Version Control History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Primary Authors** | **Description of Version** | **Date Completed** |
| Final | Umesh Yadav, CHANDAN SHAW,PRASHONTYA kar,DIWASH kapil chettri,tanuj ghosh,animesh kumar | Project Report | 10th July, 2018 |

1.

2.

3.

4.

5.

6.

Signatures of Team Members Signature of Approval

Date: Date:

**For Office Use Only**

**Mrs. mEGHALI DAS**

Project Proposal Evaluator

**Approved**

**Not Approved**

**Project Responsibility Form**

**note making app**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | GROUP NO. | SL.NO. | NAME OF MEMBER | RESPONSIBILITY | | 1 | 1 | tanuj ghosh ,chandan shaw,animesh kumar | Project Leader & PPT Coding & Designing | | | 2 | umesh yadav,diwash kapil chettri,prashontya kar | documentation | |   Each group member must participate in project development and developing the ideas for the required elements. Individual group members will be responsible for completing tasks which help to finalize the project and the performance. All group members must be assigned a task. | |  | | --- | |  | |

Date:

Name of the Students

1. tanuj ghosh
2. CHANDAN SHAW
3. animesh kumar
4. DIWASH kapil chettri
5. Umesh Yadav
6. prashontya kar

Signatures of the students

a.

b.

c.

d.

e.

f.

**DECLARATION**

We hereby declare that the project work being presented in the project proposal “NOTE MAKING APP**”** in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING** at **ARDENT COMPUTECH PVT. LTD, SALTLAKE, KOLKATA, WEST BENGAL,** is an authentic work carried out under the guidance of **MRS. MEGHALI DAS**. The matter embodied in this project work has not been submitted elsewhere for the award of any degree of our knowledge and belief.

Date:

Name of the Students

1. tanuj ghosh
2. CHANDAN SHAW
3. animesh kumar
4. DIWASH kapil chettri
5. Umesh Yadav
6. prashontya KAR

Signature of the students

a.

b.

c.

d.

e.

f.

****

**Ardent Computech Pvt Ltd (An ISO 9001:2008 Certified)**

**CF-137, Sector - 1, Salt Lake City, Kolkata - 700 064**

**CERTIFICATE**

This is to certify that this proposal of minor project entitled **“NOTE MAKING APP”** is a record of bona fide work, carried out by **1. UMESH YADAV 2. CHANDAN SHAW 3. PRASHONTYA kar 4. DIWASH kapil chettri 5. tanuj ghosh 6. animesh kumar** under my guidance at **Ardent Computech Pvt Ltd**.In my opinion, the report in its present form is in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING** and as per

regulations of the **Ardent*®.*** To the best of my knowledge, the results embodied in this report, are original in nature and worthy of incorporation in the present version of the report.

**Guide / Supervisor**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Mr. Arnab Chakraborthy**

Subject Matter Expert & Assistant Technical Head (.NET Domain)

Ardent Computech Pvt Ltd (An ISO 9001:2015 Certified)

Module 132, SDF Building, Sector 5, Salt Lake, Kolkata - 64

**ACKNOWLEDGEMENT**

Success of any project depends largely on the encouragement and guidelines of many others. We take this sincere opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project work.

Our heartfelt thanks to Ms. G Bhavya H.O.D. of NIT ANDHRA PRADESH, for providing us the opportunity to develop the project at Ardent Computech Pvt.Ltd.

We would like to show our greatest appreciation to Mr. Arnab Chakraborthy, Project Manager at Ardent, Kolkata. We always feel motivated and encouraged every time by his valuable advice and constant inspiration; without his encouragement and guidance this project would not have materialized.

We wish to express our deep sense of gratitude to our internal guide, Mr V Kishor Babu, Assistant Professor NIT ANDHRA PRADESH for her able guidance and useful suggestions, which helped us in completing the project work in time.

Words are inadequate in offering our thanks to the other trainees, project assistants and other members at Ardent Computech Pvt. Ltd. for their encouragement and cooperation in carrying out this project work. The guidance and support received from all the members and who are contributing to this project, was vital for the success of this project.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.NO.** | **Name of the Topic** | **Page No.** |
| **1.** | **Introduction** | **12** |
| 1.1 | Objective | 13 |
| 1.2 | Scope | 14 |
|  |  |  |
| **2.** | **System Analysis** | **15-24** |
| 2.1 | Identification of Need | 16 |
| 2.2 | Feasibility Study | 18 |
| 2.3 | Work flow | 19 |
| 2.4 | Functional Requirements | 21 |
| 2.5 | Non-Functional Requirements | 22 |
| 2.6 | Software Requirements | 24 |
|  |  |  |
| **3.** | **System Design** | **25-43** |
| 3.1 | Data Flow Diagram (DFD) | 26 |
| 3.2 | Entity-Relationship Diagram | 33 |
| 3.3 | Use-case Diagram | 34 |
| 3.4 | Modularization Details | 35 |
| 3.5 | User Interface Design | 38-43 |
|  |  |  |
| **4.** | **CODING** | 44 |
| **5.** | **Conclusion** | **62** |
|  |  |  |
| **6.** | **Future Scope & Further Enhancements** | **64** |
|  |  |  |
| **7.** | **Bibliography/References** | **65** |

**1.INTRODUCTION**

**What is note making APP?**

Taking notes has helped humans remember and convey information and ideas for millennia. Naturally, the process of note-taking has evolved considerably since the first jottings, and it’s now firmly in the domain of the digital.

Usually, when someone writes notes, they’re attempting to capture information for future use. For instance, if you’re in a meeting or seminar, it’s crucial to take down notes so you can remember important facts and data for later usage.

Viewed by some as a form of self-discipline, taking notes has a variety of benefits. For instance, it can help you improve your attention span, retention, organizational skills and learning ability. But most importantly, note-taking indirectly helps to relieve stress. There’s nothing worse than being unprepared for a business meeting because you’re missing some crucial pieces of information.

**OBJECTIVE**

The primary objective of the Note making app program is two-fold. The first benefit is to make entries such as notes, comments, etc. about the program currently running. The second objective is to receive the tabled information from a graphing tool and save it in text form. This then can be opened in a more advanced spreadsheet or graphing application in order to do mode detailed analysis not available in the Graphing Tool.

**SCOPE**

The project is all about providing users the opportunity to take notes of various subject for easy to understand and to grave the knowledge easily .Suppose a Student is not able to buy a book of required subject ,in this situation student can access our note making app simply by downloading our app and they can search their required subject and can read .even though if a student want to make a notes for future reference then he /she can make notes simply by downloading app and by starting the app making process. The project also gives each user the chance to go through highlights, statistics, images, videos of the book so that they can go through them and understand the subject more deeply and can easily make notes.

System Analysis

**IDENTIFICATION OF NEED**

System analysis is a process of gathering and interpreting facts, diagnosing problems and the information to recommend improvements on the system. It is a problem solving activity that requires intensive communication between the system users and system developers. System analysis or study is an important phase of any system development process. The system is studies to the minutest detail and analyzed. The system analyst plays the role of the interrogator and dwells deep into the working of the present system. The System is viewed as a whole and the input to the system are identified. The outputs from the organization are traced to the various processes. System analysis is concerned with becoming aware of the problem, identifying the relevant and Decisional variables, analysis and synthesizing the various factors and determining an optimal or at least a satisfactory solution or program of action.

A detailed study of the process must be made by various techniques like interviews, questionnaires etc. The data collected by these sources must be 9scrutinized to arrive to a conclusion. The conclusion is an understanding of how the system functions. This system is called the existing system. Now the existing system is subjected to close study and problem area are identified. The designer now function as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by the user .The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user is satisfied with proposal.

**FEASIBILITY STUDY**

Feasibility study is made to see if the project on completion will serve the purpose the organization for the amount of work.

Effort and the time that spend on it. Feasibility study lets the developer foresee the future of the project and the usefulness. A feasibility study of a system proposal is according to its workability, which is the impact on the organization, ability to meet their user needs and effective use of resources. Thus when a new application is proposed it normally goes through a feasibility study before it is approved for development.

The document provide the feasibility of the project that is being designed and lists various area that were considered very carefully during the feasibility study of this project such as Technical, Economic and operational feasibilities.

**WORK FLOW**

This Document plays a vital role in the development life cycle (SDLC) as it describes the complete requirement of the system. It is meant for use by the developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process.

WATER FALL MODEL was being chosen because all requirements were known beforehand and the objective of our software development is the computerization/automation of an already existing manual working system.

**Communicated Requirements**

**Requirements Specification**

**Design Specification**

**Executable Software Modules**

**Integrated Software Product**

**Delivered Software Product**

**Changed Requirements**

**Requirements Engineering**

**Design**

**Programming**

**Integration**

**Delivery**

**Maintenance**

**Product**

**Product**

**Input**

**Output**

**Process**

**Fig: Water Fall Model**

The developer is responsible for:

* Developing the system, which meets the SRS and solving all the requirements of the system?
* Demonstrating the system and installing the system at client's location after the acceptance testing is successful.
* Submitting the required user manual describing the system interfaces to work on it and also the documents of the system.
* Conducting any user training that might be needed for using the system.
* Maintaining the system for a period of one year after installation.

**FUNCTIONAL REQUIREMENTS**

**Modules:**

The modules will be used in this app are as follows:

* **Home:** This page contains an overview of highlights from other pages and information about the NOTE MAKING APP.
* **Draw:** This page explains to the users the format of notes to be made.
* **Gallery:** In this module, there are three pages: **IMAGES, VIDEOS, CLASSICS** where user can view and download photos, VIDEO SLIDES & MAJOR NOTES from

all time greats respectively.

**NON-FUNCTIONAL REQUIREMENTS**

* **Usability Requirement**: The app shall allow the users to access from any os device mobiles, no special training is required. The app user friendly and the system is written in simple English.
* **Availability Requirement**: The system is available 100% for the user and is used by 24 hours a day and 365 days a year. The system shall be operational 24 hours a day and 7 days a week.
* **Accuracy**: The system should accurately provide real time information taking into consideration various issues. The system shall provide 100% access reliability.
* **Performance Requirement**: The information is refreshed at regular intervals depending upon whether some updates have occurred or not. The system shall respond the member in less than 2 seconds.
* **Security Requirement**: System will use a secured database and the system will have different users and each user has different types of constraints. Only admins have the rights to update database information of other users.
* **Reliability Requirement**: The system has to be 100% reliable due to the importance of data and the damages that can be caused by incorrect data. The system will run 7 days a week and 24 hours a day.

**SOFTWARE REQUIREMENTS to build and run this app**

**SOFTWARE REQUIREMENTS**

* WINDOWS OS (XP/2000/200 Server/2003 Server/Vista or7)
* Internet Information Server 8.0 (IIS)
* SQL Server Express Edition
* ANDROID STUDIO VERSION 3.0
* emulator
* android sdk

System design

**DATA FLOW DIAGRAM**

|  |
| --- |
| A Data Flow Diagram (DFD) is a diagram that describes the flow of data and the processes that change data throughout a system. A structured analysis and design tool that can be used for flowcharting in place of or in association with information. Oriented and process oriented system flowcharts. When analysts prepare the Data Flow Diagram, they specify the user needs at a level of detail that virtually determines the information flow into and out of the system and the required data resources. This network is constructed by using a set of symbols that do not imply physical implementations. The Data Flow Diagram reviews the current physical system, prepares input and output specification, specifies the implementation plan etc. |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Four basic symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes.   |  | | --- | | **DATA FLOW DIAGRAM SYMBOLS**    Source or Destination of Data    Data Flow    Process    Storage | |  |  | | **Steps to Construct Data Flow Diagram**  Four Steps are generally used to construct a DFD.   * Process should be named and referred for easy reference. Each name should be representative of the reference. * The destination of flow is from top to bottom and from left to right. * When a process is distributed into lower level details they are numbered. * The names of data stores, sources and destinations are written in capital letters.   Rules for constructing a Data Flow Diagram-   * Arrows should not cross each other. * Squares, Circles, Files must bear a name. * Decomposed data flow squares and circles can have same names. * Draw all data flow around the outside of the diagram.   **DATA FLOW DIAGRAM**  **Level 0**  C:\Users\UMESH YADAV\Desktop\nm00.png  **Level 1**  C:\Users\UMESH YADAV\Desktop\nm01.png  **Level 2**  **C:\Users\UMESH YADAV\Desktop\02-level-one-dfd.png** |  | | |  |  | |  |  | |  |  | |  |  | |  |  | |  |
|  |  |
| **USE CASE DIAGRAM**  A **use case diagram** at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different [use cases](http://en.wikipedia.org/wiki/Use_Case) in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well.  So only static behavior is not sufficient to model a system rather dynamic behavior is more important than static behavior. In UML there are five diagrams available to model dynamic nature and use case diagram is one of them. Now as we have to discuss that the use case diagram is dynamic in nature there should be some internal or external factors for making the interaction.  These internal and external agents are known as actors. So use case diagrams are consists of actors, use cases and their relationships. The diagram is used to model the system/subsystem of an application. A single use case diagram captures a particular functionality of a system. So to model the entire system numbers of use case diagrams are used. The purpose of use case diagram is to capture the dynamic aspect of a system. But this definition is too generic to describe the purpose. Because other four diagrams (activity, sequence, collaboration and State chart) are also having the same purpose. So we will look into some specific purpose which will distinguish it from other four diagrams.  Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. So when a system is analyzed to gather its functionalities use cases are prepared and actors are identified.  Now when the initial task is complete use case diagrams are modelled to present the outside view. So in brief, the purposes of use case diagrams can be as follows:   * Used to gather requirements of a system. * Used to get an outside view of a system. * Identify external and internal factors influencing the system. * Show the interacting among the requirements are actors.  **How to draw Use Case Diagram?** Use case diagrams are considered for high level requirement analysis of a system. So when the requirements of a system are analyzed the functionalities are captured in use cases. So we can say that uses cases are nothing but the system functionalities written in an organized manner. Now the second things which are relevant to the use cases are the actors. Actors can be defined as something that interacts with the system.  The actors can be human user, some internal applications or may be some external applications. So in a brief when we are planning to draw a use case diagram we should have the following items identified.   * Functionalities to be represented as an use case * Actors * Relationships among the use cases and actors.   Use case diagrams are drawn to capture the functional requirements of a system. So after identifying the above items we have to follow the following guidelines to draw an efficient use case diagram.   * The name of a use case is very important. So the name should be chosen in such a way so that it can identify the functionalities performed. * Give a suitable name for actors. * Show relationships and dependencies clearly in the diagram. * Do not try to include all types of relationships. Because the main purpose of the diagram is to identify requirements. * Use note whenever required to clarify some important points. |  |
|  |  |
|  |  |
|  |  |
| **MODULARIZATION DETAILS**  As Modularization has gained increasing focus from companies outside its traditional industries of aircraft and automotive, more and more companies turn to it as strategy and product development tool. I intend to explain the importance aspects of modularization and how it should be initiated within a company. After determining the theoretical steps of modularization success described in literature, I intend to conduct a multiple case study of companies who have implemented modularization in order to find how real world modularization was initiated and used to improve the company’s competitiveness. By combining theory and practical approach to modularization I will derive at convergence and divergence between theoretical implementation to modularization and real world implementation to modularization. This gives a valuable input for both implantations in companies as well as new aspects to be further. |  |

**DATA INTEGRITY AND CONSTRAINTS**

Data integrity is normally enforced in a [database system](http://en.wikipedia.org/wiki/Database_system) by a series of [integrity constraints](http://en.wikipedia.org/wiki/Integrity_constraints) or rules. Three types of integrity constraints are an inherent part of the relational data model: entity integrity, referential integrity and domain integrity:

* [*Entity integrity*](http://en.wikipedia.org/wiki/Entity_integrity) concerns the concept of a [primary key](http://en.wikipedia.org/wiki/Primary_key). Entity integrity is an integrity rule which states that every table must have a primary key and that the column or columns chosen to be the primary key should be unique and not null.
* Concerns the concept of a [foreign key](http://en.wikipedia.org/wiki/Foreign_key). The referential integrity rule states that any foreign-key value can only be in one of two states. The usual state of affairs is that the foreign-key value refers to a primary key value of some table in the database. Occasionally, and this will depend on the rules of the data owner, a foreign-key value can be [null](http://en.wikipedia.org/wiki/Null_(SQL)). In this case we are explicitly saying that either there is no relationship between the objects represented in the database or that this relationship is unknown.
* *Domain integrity* specifies that all columns in a relational database must be declared upon a defined domain. The primary unit of data in the relational data model is the data item. Such data items are said to be non-decomposable or atomic. A domain is a set of values of the same type.

**USER INTERFACE DESIGN**

**User interface design (UID)** or **user interface engineering** is the [design](http://en.wikipedia.org/wiki/Design) of [user interfaces](http://en.wikipedia.org/wiki/User_interface) for [machines](http://en.wikipedia.org/wiki/Machine) and [software](http://en.wikipedia.org/wiki/Software), such as computers, [home appliances](http://en.wikipedia.org/wiki/Home_appliance), [mobile devices](http://en.wikipedia.org/wiki/Mobile_device), and other [electronic devices](http://en.wikipedia.org/wiki/Electronics), with the focus on maximizing the [user experience](http://en.wikipedia.org/wiki/User_experience). The goal of user interface design is to make the user's interaction as simple and efficient as possible, in terms of accomplishing user goals ([user-centered design](http://en.wikipedia.org/wiki/User-centered_design)).

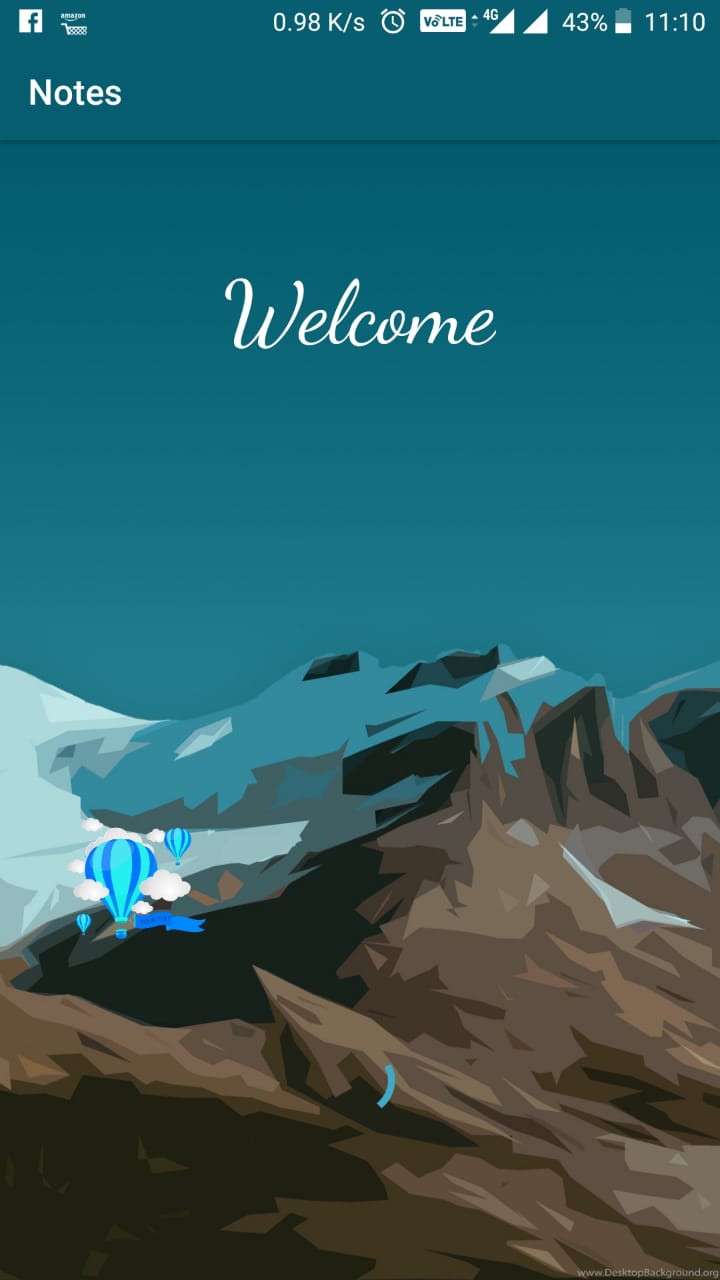
Good user interface design facilitates finishing the task at hand without drawing unnecessary attention to it. [Graphic design](http://en.wikipedia.org/wiki/Graphic_design) and typography are utilized to support its [usability](http://en.wikipedia.org/wiki/Usability), influencing how the user performs certain interactions and improving the aesthetic appeal of the design; design aesthetics may enhance or detract from the ability of users to use the functions of the interface. The design process must balance technical functionality and visual elements (e.g., [mental model](http://en.wikipedia.org/wiki/Mental_model)) to create a system that is not only operational but also usable and adaptable to changing user needs.

Interface design is involved in a wide range of projects from computer systems, to cars, to commercial planes; all of these projects involve much of the same basic human interactions yet also require some unique skills and knowledge. As a result, designers tend to specialize in certain types of projects and have skills centered on their expertise, whether that be [software design](http://en.wikipedia.org/wiki/Software_design), user research, [web design](http://en.wikipedia.org/wiki/Web_design), or [industrial design](http://en.wikipedia.org/wiki/Industrial_design).

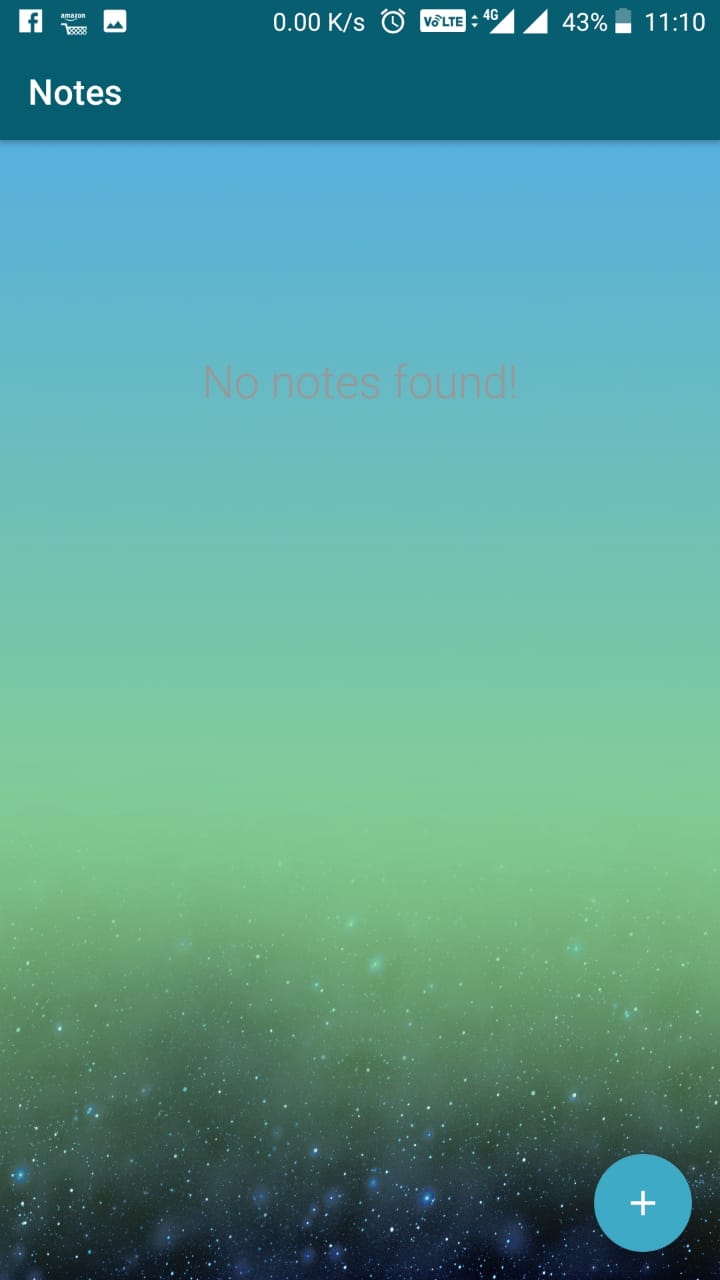
**SNAPSHOTS**

**Mobile view**

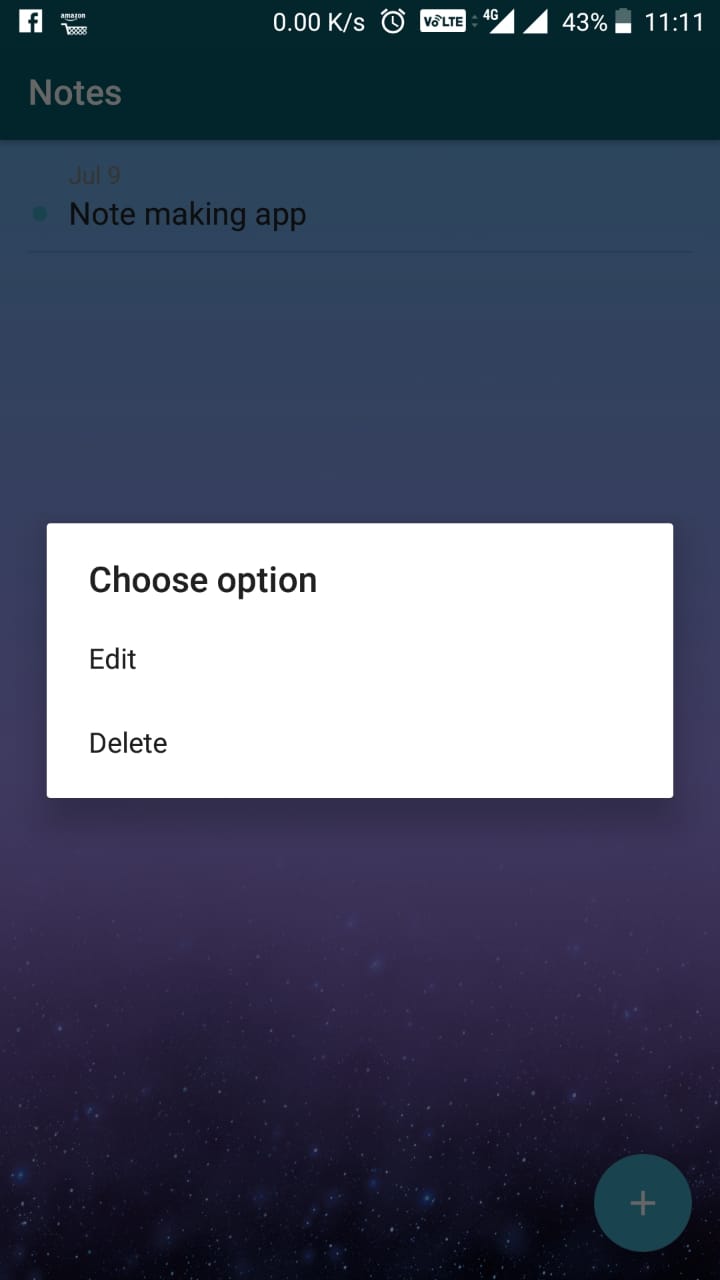
**welcome screen**

****

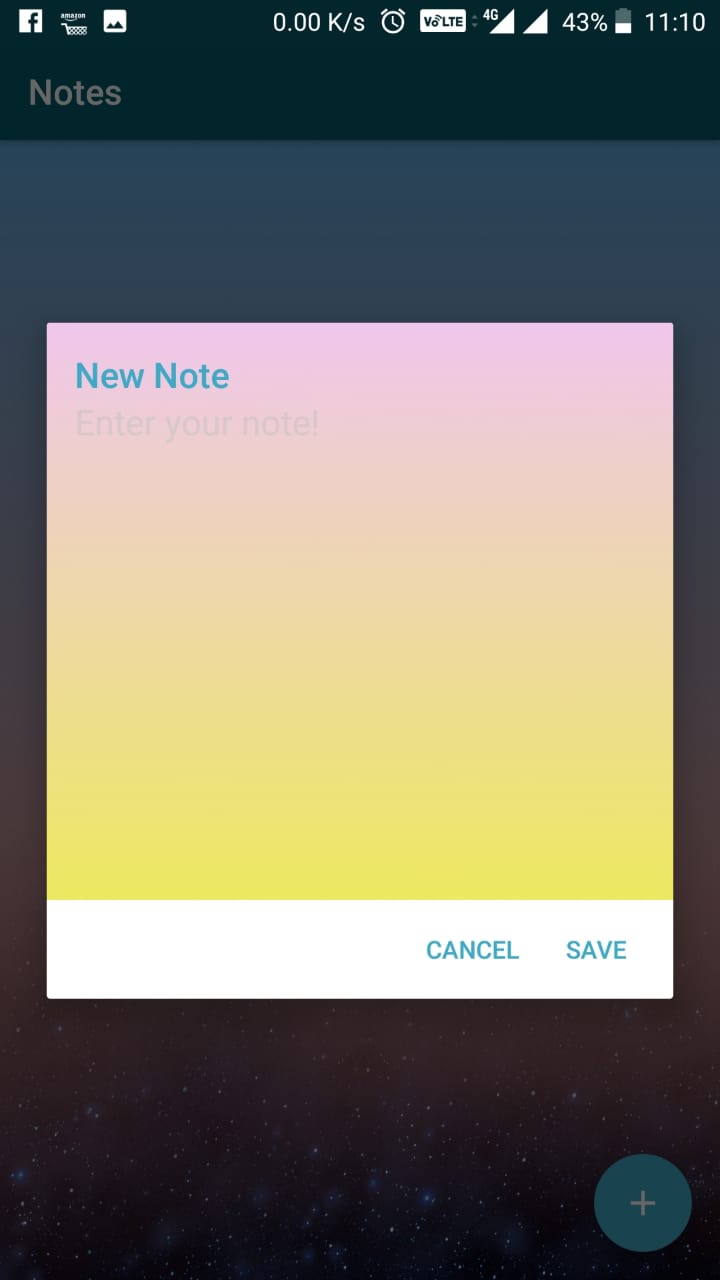
**next layout**

****

**this page enables user to edit or delete notes**

****

**edit layout**

****

**note created**

****

**CODING**

**activity home xml file**

*<?***xml version="1.0" encoding="utf-8"***?>*<**RelativeLayout 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:paddingBottom="16dp"  
 android:paddingTop="16dp"  
 android:paddingLeft="16dp"  
 android:paddingRight="16dp"  
 android:background="@drawable/welcomee"  
 tools:context=".view.HomeActivity"**>  
  
 <**TextView  
 android:id="@+id/textView"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentTop="true"  
 android:layout\_centerHorizontal="true"  
 android:layout\_marginTop="48dp"  
 android:fontFamily="cursive"  
 android:padding="5dp"  
 android:text="Welcome"  
 android:textColor="#ffffff"  
 android:textSize="50sp"** />  
  
 <**ProgressBar  
 android:id="@+id/progressBar"  
 style="?android:attr/progressBarStyle"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentBottom="true"  
 android:layout\_centerHorizontal="true"  
 android:layout\_marginBottom="74dp"** />  
  
 <**ImageView  
 android:id="@+id/imageView"  
 android:layout\_width="96dp"  
 android:layout\_height="181dp"  
 android:layout\_above="@+id/progressBar"  
 android:layout\_toStartOf="@+id/textView"  
 app:srcCompat="@drawable/cloudr"** />

**activity main xml**

*<?***xml version="1.0" encoding="utf-8"***?>*<**android.support.design.widget.CoordinatorLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:id="@+id/coordinator\_layout"  
 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:background="@drawable/my\_bg\_anim"  
 tools:context="info.androidhive.sqlite.view.MainActivity"**>  
  
 <**android.support.design.widget.AppBarLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:theme="@style/AppTheme.AppBarOverlay"**>  
  
 <**android.support.v7.widget.Toolbar  
 android:id="@+id/toolbar"  
 android:layout\_width="match\_parent"  
 android:layout\_height="?attr/actionBarSize"  
 android:background="?attr/colorPrimary"  
 app:popupTheme="@style/AppTheme.PopupOverlay"** />  
  
 </**android.support.design.widget.AppBarLayout**>  
  
 <**include layout="@layout/content\_main"** />  
  
 <**android.support.design.widget.FloatingActionButton  
 android:id="@+id/fab"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="bottom|end"  
 android:layout\_margin="@dimen/fab\_margin"  
 app:backgroundTint="@color/colorAccent"  
 app:srcCompat="@drawable/ic\_add\_white\_24dp"** />  
  
 <**ImageView  
 android:id="@+id/imageView5"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 app:srcCompat="@drawable/star1r"** />  
  
</**android.support.design.widget.CoordinatorLayout**>

**content main xml**

*<?***xml version="1.0" encoding="utf-8"***?>*<**RelativeLayout 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"  
 app:layout\_behavior="@string/appbar\_scrolling\_view\_behavior"  
 tools:context="info.androidhive.sqlite.view.MainActivity"  
 tools:showIn="@layout/activity\_main"**>  
  
 <**android.support.v7.widget.RecyclerView  
 android:id="@+id/recycler\_view"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"** />  
  
 <**TextView  
 android:id="@+id/empty\_notes\_view"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_centerHorizontal="true"  
 android:layout\_marginTop="@dimen/margin\_top\_no\_notes"  
 android:fontFamily="sans-serif-light"  
 android:text="@string/msg\_no\_notes"  
 android:textColor="@color/msg\_no\_notes"  
 android:textSize="@dimen/msg\_no\_notes"** />  
  
</**RelativeLayout**>

**note dialog xml**

*<?***xml version="1.0" encoding="utf-8"***?>*<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:id="@+id/ba"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="@drawable/fourth\_layer"  
 android:orientation="vertical"  
 android:paddingLeft="@dimen/activity\_margin"  
 android:paddingRight="@dimen/activity\_margin"  
 android:paddingTop="@dimen/activity\_margin"**>  
  
 <**TextView  
 android:id="@+id/dialog\_title"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true"  
 android:layout\_alignParentTop="true"  
 android:fontFamily="sans-serif-medium"  
 android:lineSpacingExtra="8sp"  
 android:text="@string/lbl\_new\_note\_title"  
 android:textColor="@color/colorAccent"  
 android:textSize="@dimen/lbl\_new\_note\_title"  
 android:textStyle="normal"** />  
  
 <**EditText  
 android:id="@+id/note"  
 android:layout\_width="match\_parent"  
 android:layout\_height="287dp"  
 android:layout\_below="@+id/dialog\_title"  
 android:layout\_centerHorizontal="true"  
 android:background="@android:color/transparent"  
 android:gravity="top"  
 android:hint="@string/hint\_enter\_note"  
 android:inputType="textCapSentences|textMultiLine"  
 android:lines="4"  
 android:textColorHint="@color/hint\_enter\_note"  
 android:textSize="@dimen/input\_new\_note"** />  
  
</**RelativeLayout**>

**note list xml**

*<?***xml version="1.0" encoding="utf-8"***?>*<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:clickable="true"  
 android:foreground="?attr/selectableItemBackground"  
 android:paddingBottom="@dimen/dimen\_10"  
 android:paddingLeft="@dimen/activity\_margin"  
 android:paddingRight="@dimen/activity\_margin"  
 android:paddingTop="@dimen/dimen\_10"**>  
  
 <**TextView  
 android:id="@+id/dot"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="@dimen/dot\_height"  
 android:layout\_marginRight="@dimen/dot\_margin\_right"  
 android:layout\_marginTop="@dimen/dimen\_10"  
 android:includeFontPadding="false"  
 android:textColor="@color/colorAccent"  
 android:lineSpacingExtra="0dp"  
 android:textSize="@dimen/dot\_text\_size"** />  
  
 <**TextView  
 android:id="@+id/timestamp"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_toRightOf="@id/dot"  
 android:textColor="@color/timestamp"  
 android:textSize="@dimen/timestamp"** />  
  
 <**TextView  
 android:id="@+id/note"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_below="@id/timestamp"  
 android:layout\_toRightOf="@id/dot"  
 android:textColor="@color/note\_list\_text"  
 android:textSize="@dimen/note\_list\_text"** />  
  
</**RelativeLayout**>

**java files**

**1.model**

**note.java**

**package** info.androidhive.sqlite.database.model;  
  
*/\*\*  
 \* Created by ravi on 20/02/18.  
 \*/***public class** Note {  
 **public static final** String ***TABLE\_NAME*** = **"notes"**;  
  
 **public static final** String ***COLUMN\_ID*** = **"id"**;  
 **public static final** String ***COLUMN\_NOTE*** = **"note"**;  
 **public static final** String ***COLUMN\_TIMESTAMP*** = **"timestamp"**;  
  
 **private int id**;  
 **private** String **note**;  
 **private** String **timestamp**;  
  
  
 *// Create table SQL query* **public static final** String ***CREATE\_TABLE*** =  
 **"CREATE TABLE "** + ***TABLE\_NAME*** + **"("** + ***COLUMN\_ID*** + **" INTEGER PRIMARY KEY AUTOINCREMENT,"** + ***COLUMN\_NOTE*** + **" TEXT,"** + ***COLUMN\_TIMESTAMP*** + **" DATETIME DEFAULT CURRENT\_TIMESTAMP"** + **")"**;  
  
 **public** Note() {  
 }  
  
 **public** Note(**int** id, String note, String timestamp) {  
 **this**.**id** = id;  
 **this**.**note** = note;  
 **this**.**timestamp** = timestamp;  
 }  
  
 **public int** getId() {  
 **return id**;  
 }  
  
 **public** String getNote() {  
 **return note**;  
 }  
  
 **public void** setNote(String note) {  
 **this**.**note** = note;  
 }  
  
 **public** String getTimestamp() {  
 **return timestamp**;  
 }  
  
 **public void** setId(**int** id) {  
 **this**.**id** = id;  
 }  
  
 **public void** setTimestamp(String timestamp) {  
 **this**.**timestamp** = timestamp;  
 }  
}

**database helper.java**

**package** info.androidhive.sqlite.database;  
  
**import** android.content.ContentValues;  
**import** android.content.Context;  
**import** android.database.Cursor;  
**import** android.database.sqlite.SQLiteDatabase;  
**import** android.database.sqlite.SQLiteOpenHelper;  
  
**import** java.util.ArrayList;  
**import** java.util.List;  
  
**import** info.androidhive.sqlite.database.model.Note;  
  
*/\*\*  
 \* Created by ravi on 15/03/18.  
 \*/***public class** DatabaseHelper **extends** SQLiteOpenHelper {  
  
 *// Database Version* **private static final int *DATABASE\_VERSION*** = 1;  
  
 *// Database Name* **private static final** String ***DATABASE\_NAME*** = **"notes\_db"**;  
  
  
 **public** DatabaseHelper(Context context) {  
 **super**(context, ***DATABASE\_NAME***, **null**, ***DATABASE\_VERSION***);  
 }  
  
 *// Creating Tables* @Override  
 **public void** onCreate(SQLiteDatabase db) {  
  
 *// create notes table* db.execSQL(Note.***CREATE\_TABLE***);  
 }  
  
 *// Upgrading database* @Override  
 **public void** onUpgrade(SQLiteDatabase db, **int** oldVersion, **int** newVersion) {  
 *// Drop older table if existed* db.execSQL(**"DROP TABLE IF EXISTS "** + Note.***TABLE\_NAME***);  
  
 *// Create tables again* onCreate(db);  
 }  
  
 **public long** insertNote(String note) {  
 *// get writable database as we want to write data* SQLiteDatabase db = **this**.getWritableDatabase();  
  
 ContentValues values = **new** ContentValues();  
 *// `id` and `timestamp` will be inserted automatically.  
 // no need to add them* values.put(Note.***COLUMN\_NOTE***, note);  
  
 *// insert row* **long** id = db.insert(Note.***TABLE\_NAME***, **null**, values);  
  
 *// close db connection* db.close();  
  
 *// return newly inserted row id* **return** id;  
 }  
  
 **public** Note getNote(**long** id) {  
 *// get readable database as we are not inserting anything* SQLiteDatabase db = **this**.getReadableDatabase();  
  
 Cursor cursor = db.query(Note.***TABLE\_NAME***,  
 **new** String[]{Note.***COLUMN\_ID***, Note.***COLUMN\_NOTE***, Note.***COLUMN\_TIMESTAMP***},  
 Note.***COLUMN\_ID*** + **"=?"**,  
 **new** String[]{String.*valueOf*(id)}, **null**, **null**, **null**, **null**);  
  
 **if** (cursor != **null**)  
 cursor.moveToFirst();  
  
 *// prepare note object* Note note = **new** Note(  
 cursor.getInt(cursor.getColumnIndex(Note.***COLUMN\_ID***)),  
 cursor.getString(cursor.getColumnIndex(Note.***COLUMN\_NOTE***)),  
 cursor.getString(cursor.getColumnIndex(Note.***COLUMN\_TIMESTAMP***)));  
  
 *// close the db connection* cursor.close();  
  
 **return** note;  
 }  
  
 **public** List<Note> getAllNotes() {  
 List<Note> notes = **new** ArrayList<>();  
  
 *// Select All Query* String selectQuery = **"SELECT \* FROM "** + Note.***TABLE\_NAME*** + **" ORDER BY "** +  
 Note.***COLUMN\_TIMESTAMP*** + **" DESC"**;  
  
 SQLiteDatabase db = **this**.getWritableDatabase();  
 Cursor cursor = db.rawQuery(selectQuery, **null**);  
  
 *// looping through all rows and adding to list* **if** (cursor.moveToFirst()) {  
 **do** {  
 Note note = **new** Note();  
 note.setId(cursor.getInt(cursor.getColumnIndex(Note.***COLUMN\_ID***)));  
 note.setNote(cursor.getString(cursor.getColumnIndex(Note.***COLUMN\_NOTE***)));  
 note.setTimestamp(cursor.getString(cursor.getColumnIndex(Note.***COLUMN\_TIMESTAMP***)));  
  
 notes.add(note);  
 } **while** (cursor.moveToNext());  
 }  
  
 *// close db connection* db.close();  
  
 *// return notes list* **return** notes;  
 }  
  
 **public int** getNotesCount() {  
 String countQuery = **"SELECT \* FROM "** + Note.***TABLE\_NAME***;  
 SQLiteDatabase db = **this**.getReadableDatabase();  
 Cursor cursor = db.rawQuery(countQuery, **null**);  
  
 **int** count = cursor.getCount();  
 cursor.close();  
  
  
 *// return count* **return** count;  
 }  
  
 **public int** updateNote(Note note) {  
 SQLiteDatabase db = **this**.getWritableDatabase();  
  
 ContentValues values = **new** ContentValues();  
 values.put(Note.***COLUMN\_NOTE***, note.getNote());  
  
 *// updating row* **return** db.update(Note.***TABLE\_NAME***, values, Note.***COLUMN\_ID*** + **" = ?"**,  
 **new** String[]{String.*valueOf*(note.getId())});  
 }  
  
 **public void** deleteNote(Note note) {  
 SQLiteDatabase db = **this**.getWritableDatabase();  
 db.delete(Note.***TABLE\_NAME***, Note.***COLUMN\_ID*** + **" = ?"**,  
 **new** String[]{String.*valueOf*(note.getId())});  
 db.close();  
 }  
}

**utils**

**divideritems decoration.java**

**package** info.androidhive.sqlite.utils;  
  
*/\*\*  
 \* Created by ravi on 18/01/18.  
 \*/***import** android.content.Context;  
**import** android.content.res.Resources;  
**import** android.content.res.TypedArray;  
**import** android.graphics.Canvas;  
**import** android.graphics.Rect;  
**import** android.graphics.drawable.Drawable;  
**import** android.support.v7.widget.LinearLayoutManager;  
**import** android.support.v7.widget.RecyclerView;  
**import** android.util.TypedValue;  
**import** android.view.View;  
  
  
**public class** MyDividerItemDecoration **extends** RecyclerView.ItemDecoration {  
  
 **private static final int**[] ***ATTRS*** = **new int**[]{  
 android.R.attr.***listDivider*** };  
  
 **public static final int *HORIZONTAL\_LIST*** = LinearLayoutManager.HORIZONTAL;  
 **public static final int *VERTICAL\_LIST*** = LinearLayoutManager.VERTICAL;  
  
 **private** Drawable **mDivider**;  
 **private int mOrientation**;  
 **private** Context **context**;  
 **private int margin**;  
  
 **public** MyDividerItemDecoration(Context context, **int** orientation, **int** margin) {  
 **this**.**context** = context;  
 **this**.**margin** = margin;  
 **final** TypedArray a = context.obtainStyledAttributes(***ATTRS***);  
 **mDivider** = a.getDrawable(0);  
 a.recycle();  
 setOrientation(orientation);  
 }  
  
 **public void** setOrientation(**int** orientation) {  
 **if** (orientation != ***HORIZONTAL\_LIST*** && orientation != ***VERTICAL\_LIST***) {  
 **throw new** IllegalArgumentException(**"invalid orientation"**);  
 }  
 **mOrientation** = orientation;  
 }  
  
 @Override  
 **public void** onDrawOver(Canvas c, RecyclerView parent, RecyclerView.State state) {  
 **if** (**mOrientation** == ***VERTICAL\_LIST***) {  
 drawVertical(c, parent);  
 } **else** {  
 drawHorizontal(c, parent);  
 }  
 }  
  
 **public void** drawVertical(Canvas c, RecyclerView parent) {  
 **final int** left = parent.getPaddingLeft();  
 **final int** right = parent.getWidth() - parent.getPaddingRight();  
  
 **final int** childCount = parent.getChildCount();  
 **for** (**int** i = 0; i < childCount; i++) {  
 **final** View child = parent.getChildAt(i);  
 **final** RecyclerView.LayoutParams params = (RecyclerView.LayoutParams) child  
 .getLayoutParams();  
 **final int** top = child.getBottom() + params.bottomMargin;  
 **final int** bottom = top + **mDivider**.getIntrinsicHeight();  
 **mDivider**.setBounds(left + dpToPx(**margin**), top, right - dpToPx(**margin**), bottom);  
 **mDivider**.draw(c);  
 }  
 }  
  
 **public void** drawHorizontal(Canvas c, RecyclerView parent) {  
 **final int** top = parent.getPaddingTop();  
 **final int** bottom = parent.getHeight() - parent.getPaddingBottom();  
  
 **final int** childCount = parent.getChildCount();  
 **for** (**int** i = 0; i < childCount; i++) {  
 **final** View child = parent.getChildAt(i);  
 **final** RecyclerView.LayoutParams params = (RecyclerView.LayoutParams) child  
 .getLayoutParams();  
 **final int** left = child.getRight() + params.rightMargin;  
 **final int** right = left + **mDivider**.getIntrinsicHeight();  
 **mDivider**.setBounds(left, top + dpToPx(**margin**), right, bottom - dpToPx(**margin**));  
 **mDivider**.draw(c);  
 }  
 }  
  
 @Override  
 **public void** getItemOffsets(Rect outRect, View view, RecyclerView parent, RecyclerView.State state) {  
 **if** (**mOrientation** == ***VERTICAL\_LIST***) {  
 outRect.set(0, 0, 0, **mDivider**.getIntrinsicHeight());  
 } **else** {  
 outRect.set(0, 0, **mDivider**.getIntrinsicWidth(), 0);  
 }  
 }  
  
 **private int** dpToPx(**int** dp) {  
 Resources r = **context**.getResources();  
 **return** Math.*round*(TypedValue.*applyDimension*(TypedValue.***COMPLEX\_UNIT\_DIP***, dp, r.getDisplayMetrics()));  
 }  
}

**recycler touch listener.java**

**package** info.androidhive.sqlite.utils;  
  
**import** android.content.Context;  
**import** android.support.v7.widget.RecyclerView;  
**import** android.view.GestureDetector;  
**import** android.view.MotionEvent;  
**import** android.view.View;  
  
*/\*\*  
 \* Created by ravi on 21/02/18.  
 \*/***public class** RecyclerTouchListener **implements** RecyclerView.OnItemTouchListener {  
  
 **private** ClickListener **clicklistener**;  
 **private** GestureDetector **gestureDetector**;  
  
 **public** RecyclerTouchListener(Context context, **final** RecyclerView recycleView, **final** ClickListener clicklistener) {  
  
 **this**.**clicklistener** = clicklistener;  
 **gestureDetector** = **new** GestureDetector(context, **new** GestureDetector.SimpleOnGestureListener() {  
 @Override  
 **public boolean** onSingleTapUp(MotionEvent e) {  
 **return true**;  
 }  
  
 @Override  
 **public void** onLongPress(MotionEvent e) {  
 View child = recycleView.findChildViewUnder(e.getX(), e.getY());  
 **if** (child != **null** && clicklistener != **null**) {  
 clicklistener.onLongClick(child, recycleView.getChildAdapterPosition(child));  
 }  
 }  
 });  
 }  
  
 @Override  
 **public boolean** onInterceptTouchEvent(RecyclerView rv, MotionEvent e) {  
 View child = rv.findChildViewUnder(e.getX(), e.getY());  
 **if** (child != **null** && **clicklistener** != **null** && **gestureDetector**.onTouchEvent(e)) {  
 **clicklistener**.onClick(child, rv.getChildAdapterPosition(child));  
 }  
  
 **return false**;  
 }  
  
 @Override  
 **public void** onTouchEvent(RecyclerView rv, MotionEvent e) {  
  
 }  
  
 @Override  
 **public void** onRequestDisallowInterceptTouchEvent(**boolean** disallowIntercept) {  
  
 }  
  
 **public interface** ClickListener {  
 **void** onClick(View view, **int** position);  
  
 **void** onLongClick(View view, **int** position);  
 }  
}

**views**

**home activity.java**

**package** info.androidhive.sqlite.view;  
  
**import** android.content.Intent;  
**import** android.os.Handler;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.widget.EditText;  
**import** android.widget.ImageView;  
**import** android.widget.TextView;  
  
**import** info.androidhive.sqlite.R;  
  
**public class** HomeActivity **extends** AppCompatActivity {  
 **private static int** *SPLASH\_TIME*=3500;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_home);  
 TextView textView=findViewById(R.id.textView);  
 ImageView imageView=findViewById(R.id.imageView);  
 textView.setTranslationX(-1000f);  
 textView.animate().translationXBy(1000f).setDuration(1000);  
 imageView.animate().translationXBy(1000f).setDuration(20000);  
  
  
 **new** Handler().postDelayed(**new** Runnable(){  
 @Override  
 **public void** run(){  
 Intent homeIntent=**new** Intent(HomeActivity.**this**,MainActivity.**class**);  
 startActivity(homeIntent);  
 finish();  
 }  
 },*SPLASH\_TIME*);  
 }  
}

**main activity.java**

**package** info.androidhive.sqlite.view;  
  
**import** android.content.DialogInterface;  
**import** android.graphics.drawable.AnimationDrawable;  
**import** android.os.Bundle;  
**import** android.support.design.widget.CoordinatorLayout;  
**import** android.support.design.widget.FloatingActionButton;  
**import** android.support.v7.app.AlertDialog;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.support.v7.widget.DefaultItemAnimator;  
**import** android.support.v7.widget.LinearLayoutManager;  
**import** android.support.v7.widget.RecyclerView;  
**import** android.support.v7.widget.Toolbar;  
**import** android.text.TextUtils;  
**import** android.view.LayoutInflater;  
**import** android.view.Menu;  
**import** android.view.MenuInflater;  
**import** android.view.View;  
**import** android.widget.EditText;  
**import** android.widget.ImageView;  
**import** android.widget.RelativeLayout;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**import** java.util.ArrayList;  
**import** java.util.List;  
  
**import** info.androidhive.sqlite.R;  
**import** info.androidhive.sqlite.database.DatabaseHelper;  
**import** info.androidhive.sqlite.database.model.Note;  
**import** info.androidhive.sqlite.utils.MyDividerItemDecoration;  
**import** info.androidhive.sqlite.utils.RecyclerTouchListener;  
*//android:icon="@mipmap/ic\_launcher"  
//android:roundIcon="@mipmap/ic\_launcher\_round"***public class** MainActivity **extends** AppCompatActivity {  
  
 **private static final int *time***=2000;*//Press again to exit* **private long mBackPressed**;  
 **private** NotesAdapter **mAdapter**;  
 **private** List<Note> **notesList** = **new** ArrayList<>();  
 **private** CoordinatorLayout **coordinatorLayout**;  
 **private** RecyclerView **recyclerView**;  
 **private** TextView **noNotesView**;  
  
 **private** DatabaseHelper **db**;  
  
 RelativeLayout **ba**;  
 AnimationDrawable **animationDrawable**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_main);  
 ImageView imageView5=findViewById(R.id.imageView5);  
 imageView5.setTranslationY(530f);  
 *//ImageView imageView6=findViewById(R.id.imageView6);  
 //imageView6.setTranslationY(-530f);  
  
  
  
  
  
 //noNotesView.setTextColor(Color.RED);* Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);  
 setSupportActionBar(toolbar);  
*//insta colour* **coordinatorLayout** = findViewById(R.id.coordinator\_layout);  
  
 **animationDrawable**=(AnimationDrawable)**coordinatorLayout**.getBackground();  
 **animationDrawable**.setEnterFadeDuration(4500);  
 **animationDrawable**.setExitFadeDuration(4500);  
 **animationDrawable**.start();  
  
  
 **recyclerView** = findViewById(R.id.recycler\_view);  
 **noNotesView** = findViewById(R.id.empty\_notes\_view);  
  
 **db** = **new** DatabaseHelper(**this**);  
  
 **notesList**.addAll(**db**.getAllNotes());  
  
 FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab);  
 fab.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 showNoteDialog(**false**, **null**, -1);  
 }  
 });  
  
 **mAdapter** = **new** NotesAdapter(**this**, **notesList**);  
 RecyclerView.LayoutManager mLayoutManager = **new** LinearLayoutManager(getApplicationContext());  
 **recyclerView**.setLayoutManager(mLayoutManager);  
 **recyclerView**.setItemAnimator(**new** DefaultItemAnimator());  
 **recyclerView**.addItemDecoration(**new** MyDividerItemDecoration(**this**, LinearLayoutManager.VERTICAL, 16));  
 **recyclerView**.setAdapter(**mAdapter**);  
  
 toggleEmptyNotes();  
  
 */\*\*  
 \* On long press on RecyclerView item, open alert dialog  
 \* with options to choose  
 \* Edit and Delete  
 \* \*/* **recyclerView**.addOnItemTouchListener(**new** RecyclerTouchListener(**this**,  
 **recyclerView**, **new** RecyclerTouchListener.ClickListener() {  
 @Override  
 **public void** onClick(View view, **final int** position) {  
 showActionDialog1(position);  
 }  
  
 @Override  
 **public void** onLongClick(View view, **int** position) {  
 showActionsDialog(position);  
 }  
 }));  
 }  
  
  
 @Override  
 **public void** onBackPressed() {  
 **if**(**mBackPressed**+***time***>System.*currentTimeMillis*()){  
 **super**.onBackPressed();  
 System.*exit*(0);  
 }  
 **else**{  
 Toast.*makeText*(**this**, **"Tap again to exit"**, Toast.***LENGTH\_SHORT***).show();  
 **mBackPressed**=System.*currentTimeMillis*();  
 }  
 }  
  
 */\* @Override  
 public boolean onCreateOptionsMenu(Menu menu) {  
 MenuInflater menuInflater=getMenuInflater();  
 menuInflater.inflate(R.menu.menu\_main,menu);  
 return super.onCreateOptionsMenu(menu);  
 }\*/  
  
 /\*\*  
 \* Inserting new note in db  
 \* and refreshing the list  
 \*/* **private void** createNote(String note) {  
 *// inserting note in db and getting  
 // newly inserted note id* **long** id = **db**.insertNote(note);  
  
 *// get the newly inserted note from db* Note n = **db**.getNote(id);  
  
 **if** (n != **null**) {  
 *// adding new note to array list at 0 position* **notesList**.add(0, n);  
  
 *// refreshing the list* **mAdapter**.notifyDataSetChanged();  
  
 toggleEmptyNotes();  
 }  
 }  
  
 */\*\*  
 \* Updating note in db and updating  
 \* item in the list by its position  
 \*/* **private void** updateNote(String note, **int** position) {  
 Note n = **notesList**.get(position);  
 *// updating note text* n.setNote(note);  
  
 *// updating note in db* **db**.updateNote(n);  
  
 *// refreshing the list* **notesList**.set(position, n);  
 **mAdapter**.notifyItemChanged(position);  
  
 toggleEmptyNotes();  
 }  
  
 */\*\*  
 \* Deleting note from SQLite and removing the  
 \* item from the list by its position  
 \*/* **private void** deleteNote(**int** position) {  
 *// deleting the note from db* **db**.deleteNote(**notesList**.get(position));  
  
 *// removing the note from the list* **notesList**.remove(position);  
 **mAdapter**.notifyItemRemoved(position);  
  
 toggleEmptyNotes();  
 }  
  
 */\*\*  
 \* Opens dialog with Edit - Delete options  
 \* Edit - 0  
 \* Delete - 0  
 \*/  
  
 //for options on long click* **private void** showActionsDialog(**final int** position) {  
 CharSequence colors[] = **new** CharSequence[]{**"Edit"**, **"Delete"**};  
  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(**this**);  
 builder.setTitle(**"Choose option"**);  
 builder.setItems(colors, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 **if** (which == 0) {  
 showNoteDialog(**true**, **notesList**.get(position), position);  
 } **else** {  
 deleteNote(position);  
 }  
 }  
 });  
 builder.show();  
 }  
 *//for directly editing on click* **private void** showActionDialog1(**final int** position){  
 showNoteDialog(**true**,**notesList**.get(position),position);  
 }  
  
  
 */\*\*  
 \* Shows alert dialog with EditText options to enter / edit  
 \* a note.  
 \* when shouldUpdate=true, it automatically displays old note and changes the  
 \* button text to UPDATE  
 \*/* **private void** showNoteDialog(**final boolean** shouldUpdate, **final** Note note, **final int** position) {  
 LayoutInflater layoutInflaterAndroid = LayoutInflater.*from*(getApplicationContext());  
 View view = layoutInflaterAndroid.inflate(R.layout.note\_dialog, **null**);  
  
 AlertDialog.Builder alertDialogBuilderUserInput = **new** AlertDialog.Builder(MainActivity.**this**);  
 alertDialogBuilderUserInput.setView(view);  
  
 **final** EditText inputNote = view.findViewById(R.id.note);  
 TextView dialogTitle = view.findViewById(R.id.dialog\_title);  
 dialogTitle.setText(!shouldUpdate ? getString(R.string.lbl\_new\_note\_title) : getString(R.string.lbl\_edit\_note\_title));  
  
 **if** (shouldUpdate && note != **null**) {  
 inputNote.setText(note.getNote());  
 }  
 alertDialogBuilderUserInput  
 .setCancelable(**false**)  
 .setPositiveButton(shouldUpdate ? **"update"** : **"save"**, **new** DialogInterface.OnClickListener() {  
 **public void** onClick(DialogInterface dialogBox, **int** id) {  
  
 }  
 })  
 .setNegativeButton(**"cancel"**,  
 **new** DialogInterface.OnClickListener() {  
 **public void** onClick(DialogInterface dialogBox, **int** id) {  
 dialogBox.cancel();  
 }  
 });  
  
 **final** AlertDialog alertDialog = alertDialogBuilderUserInput.create();  
 alertDialog.show();  
  
 alertDialog.getButton(AlertDialog.***BUTTON\_POSITIVE***).setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 *// Show toast message when no text is entered* **if** (TextUtils.*isEmpty*(inputNote.getText().toString())) {  
 Toast.*makeText*(MainActivity.**this**, **"Enter note!"**, Toast.***LENGTH\_SHORT***).show();  
 **return**;  
 } **else** {  
 alertDialog.dismiss();  
 }  
  
 *// check if user updating note* **if** (shouldUpdate && note != **null**) {  
 *// update note by it's id* updateNote(inputNote.getText().toString(), position);  
 } **else** {  
 *// create new note* createNote(inputNote.getText().toString());  
 }  
 }  
 });  
 }  
  
 */\*\*  
 \* Toggling list and empty notes view  
 \*/* **private void** toggleEmptyNotes() {  
 *// you can check notesList.size() > 0* **if** (**db**.getNotesCount() > 0) {  
 **noNotesView**.setVisibility(View.***GONE***);  
 } **else** {  
 **noNotesView**.setVisibility(View.***VISIBLE***);  
 }  
 }  
}

**notes adapter.java**

**package** info.androidhive.sqlite.view;  
  
  
**import** android.content.Context;  
**import** android.graphics.drawable.AnimationDrawable;  
**import** android.support.v7.widget.RecyclerView;  
**import** android.text.Html;  
**import** android.util.Log;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.RelativeLayout;  
**import** android.widget.TextView;  
  
**import** java.text.ParseException;  
**import** java.text.SimpleDateFormat;  
**import** java.util.Date;  
**import** java.util.List;  
  
**import** info.androidhive.sqlite.R;  
**import** info.androidhive.sqlite.database.model.Note;  
  
**public class** NotesAdapter **extends** RecyclerView.Adapter<NotesAdapter.MyViewHolder> {  
  
 **private** Context **context**;  
 **private** List<Note> **notesList**;  
  
  
  
  
 **public class** MyViewHolder **extends** RecyclerView.ViewHolder {  
 **public** TextView **note**;  
 **public** TextView **dot**;  
 **public** TextView **timestamp**;  
  
 **public** MyViewHolder(View view) {  
 **super**(view);  
 **note** = view.findViewById(R.id.note);  
 **dot** = view.findViewById(R.id.dot);  
 **timestamp** = view.findViewById(R.id.timestamp);  
 }  
 }  
  
  
 **public** NotesAdapter(Context context, List<Note> notesList) {  
 **this**.**context** = context;  
 **this**.**notesList** = notesList;  
 }  
  
 @Override  
 **public** MyViewHolder onCreateViewHolder(ViewGroup parent, **int** viewType) {  
 View itemView = LayoutInflater.*from*(parent.getContext())  
 .inflate(R.layout.note\_list\_row, parent, **false**);  
  
 **return new** MyViewHolder(itemView);  
 }  
  
 @Override  
 **public void** onBindViewHolder(MyViewHolder holder, **int** position) {  
 Note note = **notesList**.get(position);  
  
 holder.**note**.setText(note.getNote());  
  
 *// Displaying dot from HTML character code* holder.**dot**.setText(Html.*fromHtml*(**"&#8226;"**));  
  
 *// Formatting and displaying timestamp* holder.**timestamp**.setText(formatDate(note.getTimestamp()));  
 }  
  
 @Override  
 **public int** getItemCount() {  
 **return notesList**.size();  
 }  
  
 */\*\*  
 \* Formatting timestamp to `MMM d` format  
 \* Input: 2018-02-21 00:15:42  
 \* Output: Feb 21  
 \*/* **private** String formatDate(String dateStr) {  
 **try** {  
 SimpleDateFormat fmt = **new** SimpleDateFormat(**"yyyy-MM-dd HH:mm:ss"**);  
 Date date = fmt.parse(dateStr);  
 SimpleDateFormat fmtOut = **new** SimpleDateFormat(**"MMM d"**);  
 **return** fmtOut.format(date);  
 } **catch** (ParseException e) {  
  
 }  
  
 **return ""**;  
 }  
}

**CONCLUSION**

This project has been appreciated by all the users in the organization. It is easy to use, since it uses the GUI provided in the user dialog. User friendly screens are provided.The usage of app increases the efficiency, decreases the effort. It has been thoroughly tested and implemented.

The project **“note making app”** is the ideal place for every users but this app is mainly focusing to help students to make digital notes, note making is one of the most important elements in preparing for an exam. Our time is limited, so reducing mass amounts of information and summarising key points helps us to better focus our time and be much more efficient. Each student can develop [note taking strategies](https://www.examtime.com/blog/4-note-taking-strategies/) that best fits their learning style. using digital notes also provides sharing option so that other can get benefit from it.with further enhancements speech to text conversion can also help user to use “note making app” more quick and no more illegible handwriting.time saved with increased efficiency and less keypad typing and workflow visibility.

**FUTURE SCOPE AND FURTHER ENHANCEMENTS**

In future, we would like to keep working on this project and make new additions to provide users with more advanced features and more detailed information. We have set our sights on the following additions in future-

### Rich input methods：Keyboard, handwriting, or drawing. Just pick one method that best fit the information you want to take note

### Various attachment formats：Photo, Audio, Video, Text File, Time Stamp, Stamp, and Shape formats are supported. You can take a note with rich content type

### Notebook management functions：Lock / unlock notebook, reorder pages, and edit notebook cover. Also, you can select different template of notebook to fulfill different purpose of notes

### Sync and share：you can sync your notebooks via cloud storage to access it anytime, on any devices, and you can also share your notebook with your family or friends by mail, Facebook, IM…etc

### Enjoy your life and record everything by using Note making app

**REFERENCES**

1. <https://www.w3schools.com>
2. <https://www.slideshare.com>
3. <https://www.scribd.com>
4. <https://www.tutorialspoint.com>
5. https://www.youtube.com

THANK YOU