**VIVEKANAND EDUCATION SOCIETY'S POLYTECHNIC,**

**SINDHI SOCIETY, CHEMBUR, MUMBAI - 400 071**



**PROJECT REPORT ON**

**MINI KINDLE**

**SUBMITTED BY**

ANUJ DUBEY [65]

ATHARVA KENY [66]

MUNTZAR SAYYED [67]

SAGAR SONI [68]

**GUIDED BY**

MRS MEENA TALELE

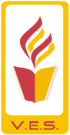
NBA ACCREDITED FROM 2017 TO 2018

DEPARTMENT OF COMPUTER TECHNOLOGY

2018-2019

**VIVEKANAND EDUCATION SOCIETY'S POLYTECHNIC,**

**SINDHI SOCIETY, CHEMBUR, MUMBAI - 400 071**



**CERTIFICATE OF APPROVAL OF PROJECT**

This is to certify that,

Mr./Ms.- Anoj Dubey, Atharva Keny, Muntzar Sayyed, Sagar Soni.

Roll No. – 65, 66.67 & 68of VI Semester Diploma in Computer Technology have completed the term satisfactorily in Industrial Project Development for academic year 2018-2019 as prescribed in the MSBTE curriculum.

Place: Chembur, Mumbai. Enrollment No:1700040231

1700040232

1700040233

1700040234

Date: Exam No:

Principal IC. Head of Department

Project Guide Examiner

**ACKNOWLEGMENT**

We would like to thank all those people who helped us in successful completion of the project “MiniKindle (Online Library)”

We would like to thank **Mr. Vikrant Joshi**, Principal of V.E.S Polytechnic for his immense support and motivation.

We would like to thank **Mr. Avinash Dangwani**, IC. Head of the Department, Computer Technology of V.E.S Polytechnic, for his guidance. We are wholeheartedly thankful to him for giving us his valuable time & attention and for providing us a systematic way for completing our Project in time.

We would like to thank **Mrs. Meena Talele,** our project guide for his guidance. We have learn so many things from him and he motivated us and strengthened our confidence in doing the thesis. We express our deepest gratitude for his valuable suggestions and constant motivation that greatly helped the project work to successfully complete. Throughout the project work, his useful suggestions, constant encouragement has given us a right direction and shape to our learning.

We would also thank all the faculty members who have been a constant source and encouragement during the entire course of our study in this college.

**TABLE OF CONTENT**

|  |  |  |
| --- | --- | --- |
| **Sr No.** | **Topic** | **Page No.** |
| **1** | **INTRODUCTION** | **01** |
| 1.1 | Overview of the project | 02 |
| 1.2 | Existing System | 02 |
| 1.5 | Advantages of Proposed System | 03 |
| **2** | **SYSTEM DEVELOPMENT** | **04** |
| 2.1 | System Specification | 05 |
| 2.2 | Technology Used | 06 |
| **3** | **FEATURES** | **9** |
| **4** | **CLASS DIAGRAM** | **11** |
| **5** | **ACTIVITY DIAGRAM** | **14** |
| **6** | **FLOWCHART** | **17** |
| **7** | **ALGORITHM** | **19** |
| **8** | **GANTT CHART** | **21** |
| **9** | **SNAPSHOTS** | **28** |
| **10** | **CONCLUSION** | **56** |
| **11** | **FUTURE SCOPE** | **58** |
| **12** | **REFERENCES** | **60** |
| **13** | **TESTING** | **62** |
| 13.1 | Testing Details | 64 |
| 13.2 | Test Cases | 67 |
| **14** | **BUG REPORT** | **68** |

CHAPTER 1

**INTRODUCTION**

**Chapter 1**

**INTRODUCTION**

* 1. **OVERVIEW OF THE PROJECT**

Our Project uses Android Development to store, organize and display books segregated and categorized according to their genres and themes. It assigns profiles to users so they can have individual recommendations of books and series according to their previous interest. It remembers the chapters the user has read so they can continue to read from the exact point they stopped reading (Bookmarks).

**1.2 EXISTING SYSTEM**

E-bookstore mainly in the book content providers in the role of the consumer's vision. Compared to the traditional paper books, its advantage is low cost, massive storage, convenient search positioning, personalized service, low carbon environmental protection and a variety of services. In general, operators in addition to the need to provide a large amount of reading data, to provide a perfect terminal reading application is also very important.

**1.3 LIMITATIONS OF EXISTING SYSTEM**

* Lack of security of data.
* More man power.
* Time consuming.
* Difficult to manage the financial details.
  1. **PROPOSED SYSTEM**

Our system will give the ease to read the books as per the user’s need. Updates will be sent to all the members via mobile messages and emails. Updating of the server with new books will be sent to the user. Basically, the application provides different magazines, novels, comics, manga etc.

* 1. **ADVANTAGES OF PROPOSED SYSTEM**

The system is very simple in design and to implement. The system requires very low system resources and the system will work in almost all configurations. It has got following features:

• Security of data.

• Ensure data accuracy’s.

• Proper control of the higher authority.

• Minimize manual data entry.

• Minimum time needed for the various processing.

• Greater efficiency.

• Better service.

• User friendliness and interactive

• Minimum time required

• Bookmark for the last page you read.

• Recommendation.

.

CHAPTER 2

**SYSTEM DEVELOPMENT**

**Chapter 2**

**SYSTEM DEVELOPMENT**

**2.1 SYSTEM SPECIFICATION**

2.1.1 Hardware Requirements

(Minimum of the hardware required)

* Processor: Min Intel Core i3.
* RAM Capacity: Min 2 GB
* Hard Disk Capacity: Min 10 GB.

2.1.2 Software Requirements

(Any higher version will do)

*  Operating System: Android 4.4.0 KitKat
*  Coding Language: Java, XML

**2.2 TECHNOLOGY USED**

**2.2.1 Java**

What is Java?

Java is a general-purpose [computer-programming language](https://en.m.wikipedia.org/wiki/Programming_language) that is [concurrent](https://en.m.wikipedia.org/wiki/Concurrent_computing), [class-based](https://en.m.wikipedia.org/wiki/Class-based_programming), [object-oriented](https://en.m.wikipedia.org/wiki/Object-oriented_programming), and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "[write once, run anywhere](https://en.m.wikipedia.org/wiki/Write_once,_run_anywhere)" (WORA), meaning that [compiled](https://en.m.wikipedia.org/wiki/Compiler) Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to [bytecode](https://en.m.wikipedia.org/wiki/Java_bytecode) that can run on any [Java virtual machine](https://en.m.wikipedia.org/wiki/Java_virtual_machine) (JVM) regardless of [computer architecture](https://en.m.wikipedia.org/wiki/Computer_architecture). As of 2016, Java is one of the most [popular programming languages in use](https://en.m.wikipedia.org/wiki/Measuring_programming_language_popularity), particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by [James Gosling](https://en.m.wikipedia.org/wiki/James_Gosling) at [Sun Microsystems](https://en.m.wikipedia.org/wiki/Sun_Microsystems) (which has since been [acquired by Oracle Corporation](https://en.m.wikipedia.org/wiki/Sun_acquisition_by_Oracle)) and released in 1995 as a core component of Sun Microsystem’s [Java platform](https://en.m.wikipedia.org/wiki/Java_(software_platform)). The language derives much of its original features from [Small Talk](https://en.m.wikipedia.org/wiki/SmallTalk), with a [syntax](https://en.m.wikipedia.org/wiki/Syntax_(programming_languages)) similar to [C](https://en.m.wikipedia.org/wiki/C_(programming_language)) and [C++](https://en.m.wikipedia.org/wiki/C%2B%2B), but it has fewer [low-level](https://en.m.wikipedia.org/wiki/Low-level_programming_language) facilities than either of them

**2.2.2 Android Studio**

What is Android Studio?

Android Studio is the official [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) for [Google](https://en.wikipedia.org/wiki/Google)'s [Android](https://en.wikipedia.org/wiki/Android_(operating_system)) [operating system](https://en.wikipedia.org/wiki/Operating_system), built on [JetBrains](https://en.wikipedia.org/wiki/JetBrains)' [IntelliJ IDEA](https://en.wikipedia.org/wiki/IntelliJ_IDEA) software and designed specifically for development. It is available for download on [Windows](https://en.wikipedia.org/wiki/Windows), [macOS](https://en.wikipedia.org/wiki/MacOS) and [Linux](https://en.wikipedia.org/wiki/Linux) based operating systems. It is a replacement for the [Eclipse Android Development Tools](https://en.wikipedia.org/wiki/Eclipse_(software)#Android_Development_Tools) (ADT) as the primary IDE for native Android application development. Android Studio was announced on May 16, 2013 at the [Google I/O](https://en.wikipedia.org/wiki/Google_I/O) conference. It was in early access preview stage starting from version 0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014. The first stable build was released in December 2014, starting from version 1.0. The current stable version is 3.3, which was released in January 2019.

**2.2.3 XML**

What IS XML?

Extensible Markup Language (XML) is a [markup language](https://en.wikipedia.org/wiki/Markup_language) that defines a set of rules for encoding [documents](https://en.wikipedia.org/wiki/Electronic_document) in a [format](https://en.wikipedia.org/wiki/File_format) that is both [human-readable](https://en.wikipedia.org/wiki/Human-readable_medium) and [machine-readable](https://en.wikipedia.org/wiki/Machine-readable_data). The [W3C](https://en.wikipedia.org/wiki/World_Wide_Web_Consortium)'s XML 1.0 Specification and several other related specifications all of them free [open standards](https://en.wikipedia.org/wiki/Open_standard)—define XML.The design goals of XML emphasize simplicity, generality, and usability across the Internet. It is a textual data format with strong support via [Unicode](https://en.wikipedia.org/wiki/Unicode) for different [human languages](https://en.wikipedia.org/wiki/Language). Although the design of XML focuses on documents, the language is widely used for the representation of arbitrary [data structures](https://en.wikipedia.org/wiki/Data_structure) such as those used in [web services](https://en.wikipedia.org/wiki/Web_service).

**2.2.4 FIREBASE**

What IS Firebase?

Firebase provides a real time database and backend as a service. The service provides application developers an API that allows application data to be synchronized across clients and stored on Firebase's cloud. The company provides client libraries that enable integration with [Android](https://en.wikipedia.org/wiki/Android_(operating_system)), [iOS](https://en.wikipedia.org/wiki/IOS), [JavaScript](https://en.wikipedia.org/wiki/JavaScript), [Java](https://en.wikipedia.org/wiki/Java_(programming_language)), [Objective-C](https://en.wikipedia.org/wiki/Objective-C), [Swift](https://en.wikipedia.org/wiki/Swift_(programming_language)) and [Node.js](https://en.wikipedia.org/wiki/Node.js) applications. The database is also accessible through a REST API and bindings for several [JavaScript frameworks](https://en.wikipedia.org/wiki/JavaScript_frameworks) such as [AngularJS](https://en.wikipedia.org/wiki/AngularJS), [React](https://en.wikipedia.org/wiki/React_(JavaScript_library)), [Ember.js](https://en.wikipedia.org/wiki/Ember.js) and [Backbone.js](https://en.wikipedia.org/wiki/Backbone.js) The REST API uses the [Server-Sent Events](https://en.wikipedia.org/wiki/Server-sent_events) protocol, which is an API for creating HTTP connections for receiving push notifications from a server. Developers using the real time database can secure their data by using the company's server-side-enforced security rules. Cloud Firestore which is Firebase's next generation of the Real time Database was released for beta use

CHAPTER 3

**FEATURES**

**Chapter 3**

**Features**

In today’s world, it is very important to be technically advanced in all fields so by using this software , we can have an organized system to provide data and security .This model will help the user to read the various types of books..

1. Security of data.
2. Ensure data accuracy’s.
3. Proper control of the higher authority.
4. Minimize manual data entry.
5. Minimum time needed for the various processing.
6. Greater efficiency.
7. Better service.
8. User friendliness and interactive.
9. Minimum time required.
10. Cost Efficient.

CHAPTER 4

**CLASS DIAGRAM**

**Chapter 4**

**CLASS DIAGRAM**

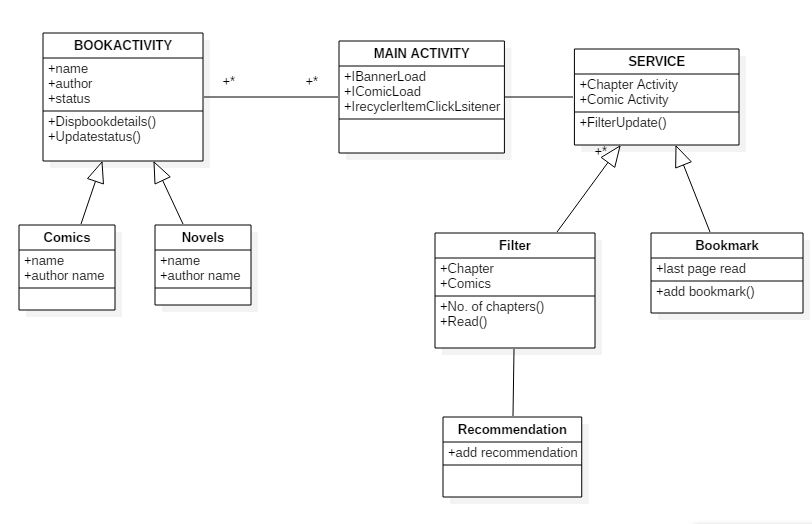


Fig 4.1 Use Case Diagram

CHAPTER 6

**FLOWCHART**

**Chapter 6**

**FLOWCHART**



Fig 6.1 Flowchart for User’s Activity

CHAPTER 7

**ALGORITHM**

**Chapter 7**

**ALGORITHM**

1. Start
2. Open the application.
3. Select any of the book you want to read.
4. Search the Name of the Book.
5. Select the chapters from the book.
6. Filter the Book.
7. Add the bookmark on the last page you read, to continue from the from the last page you read.
8. Add your Feedback on the feedback panel.
9. Stop.

CHAPTER 8

**GANTT CHART**

**Chapter 8**

**GANTT CHART**

* A **Gantt Chart** is a type of bar chart that illustrates a project schedule.
* Gantt charts illustrate our start and finish dates of the terminal elements and summary elements of a project.
* It is one of the most popular and useful ways of showing activities (tasks or events) displayed against time.

**GANTT CHART-1**

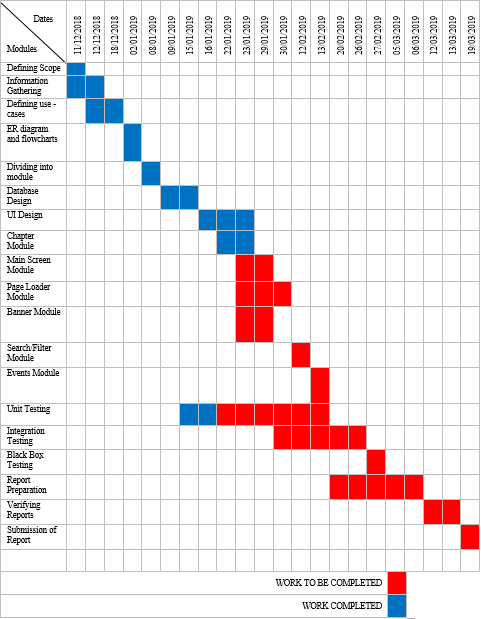


Fig 8.1 – GANTT CHART 1

**GANTT CHART-2**

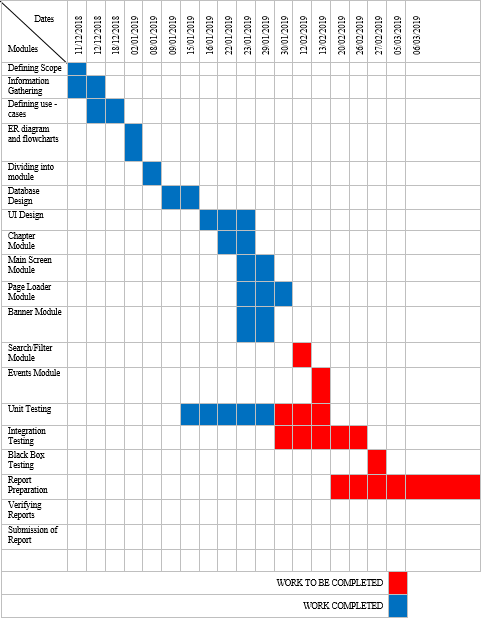
****

Fig 8.2– GANTT CHART 2

**GANTT CHART-3**

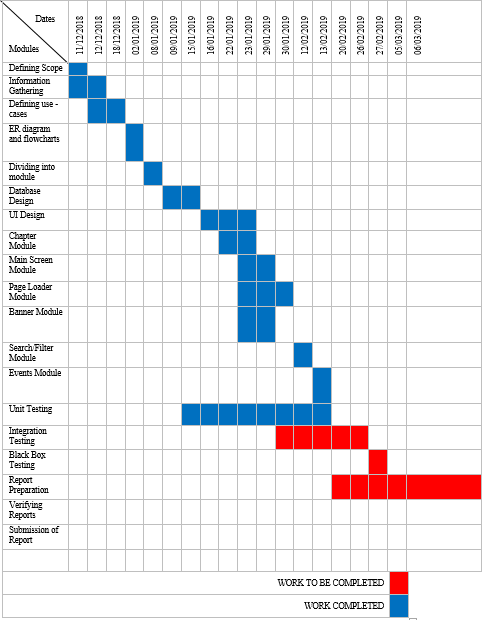
****

Fig 8.3 – GANTT CHART 3

**GANTT CHART-4**

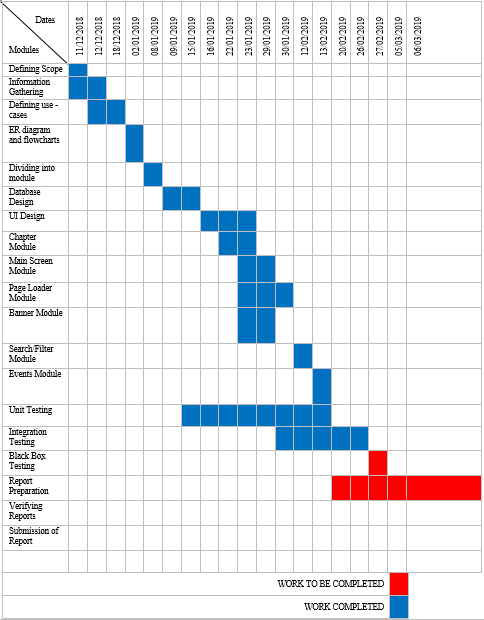
****

Fig 8.4 – GANTT CHART 4

**GANTT CHART-5**

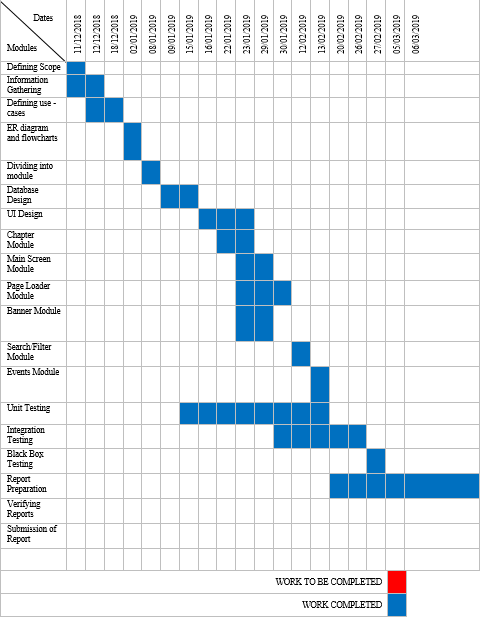


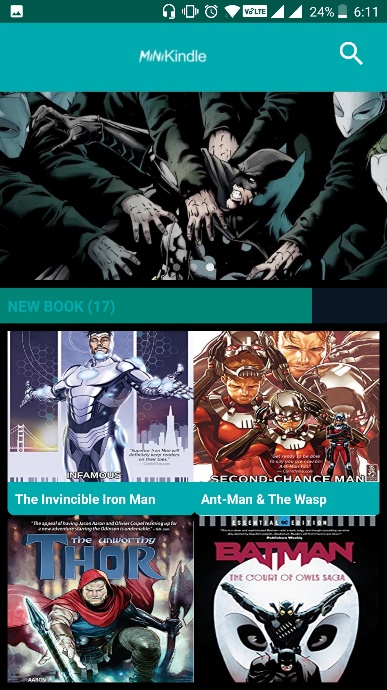
FIG 8.5 – GANTT CHART 5

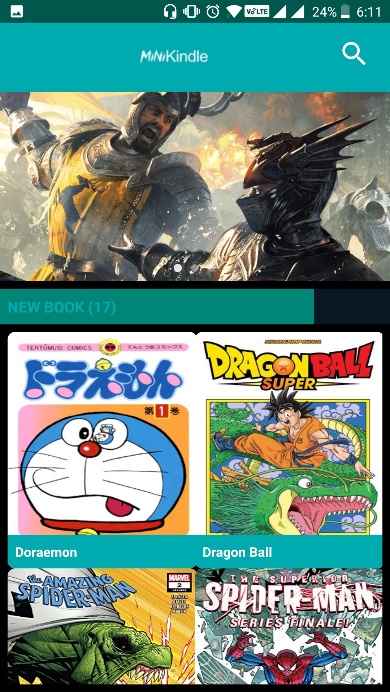
CHAPTER 9

**SNAPSHOTS**

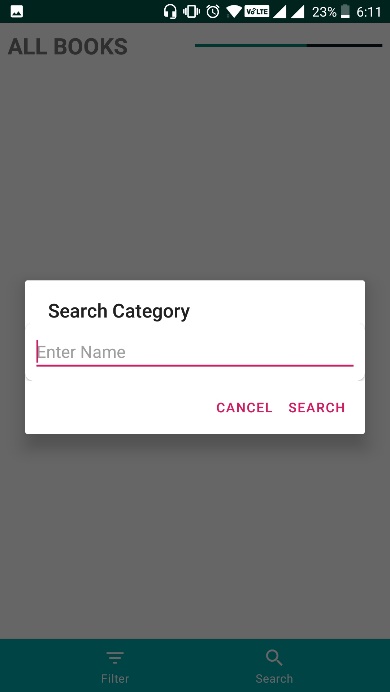
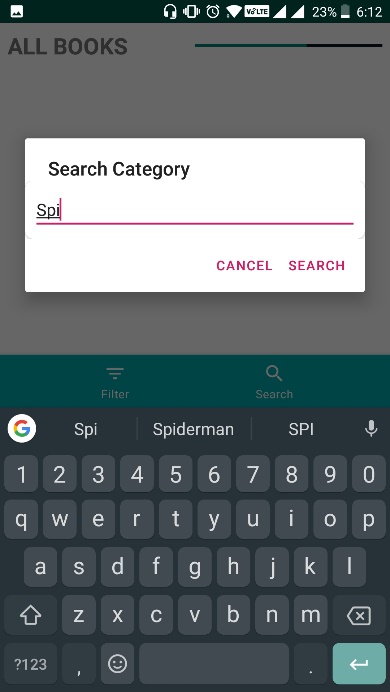
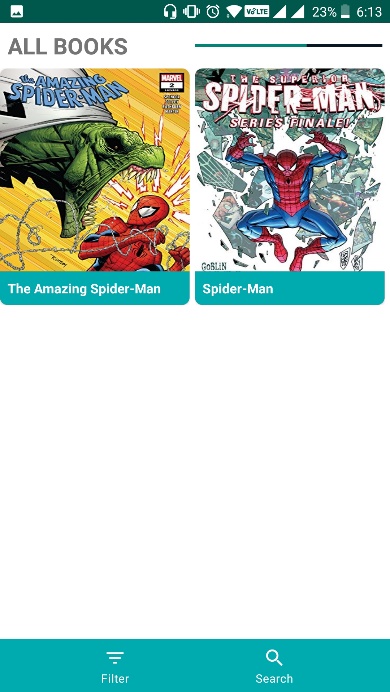
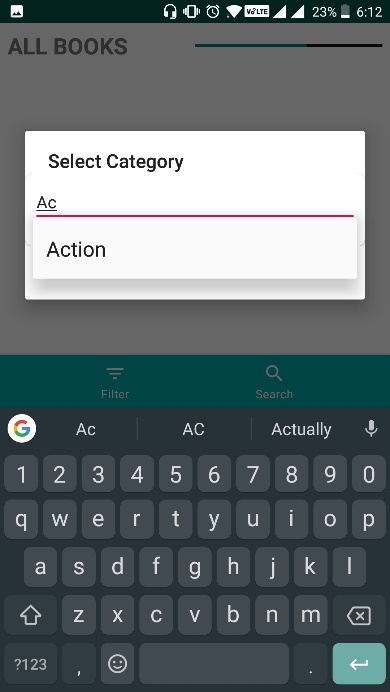
**CHAPTER 9**

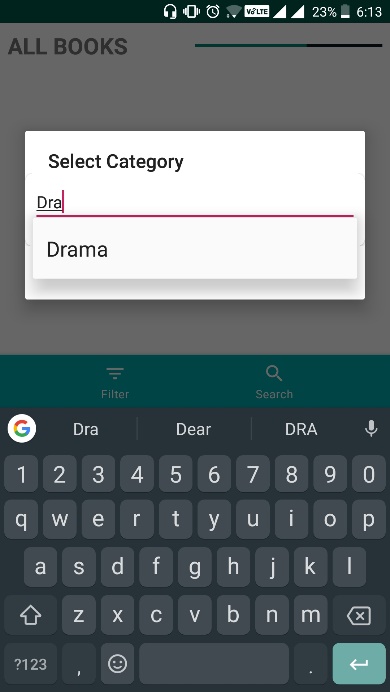
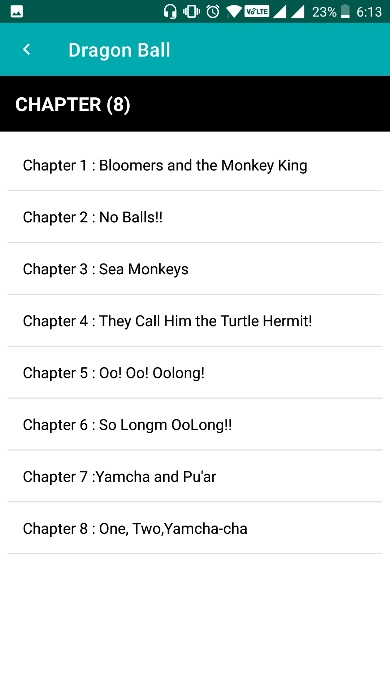
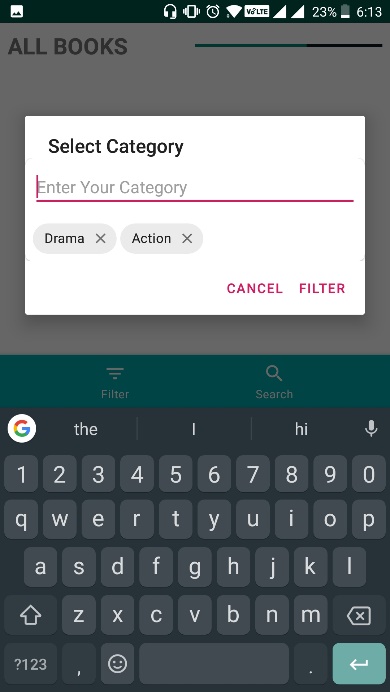
**SNAPSHOTS**

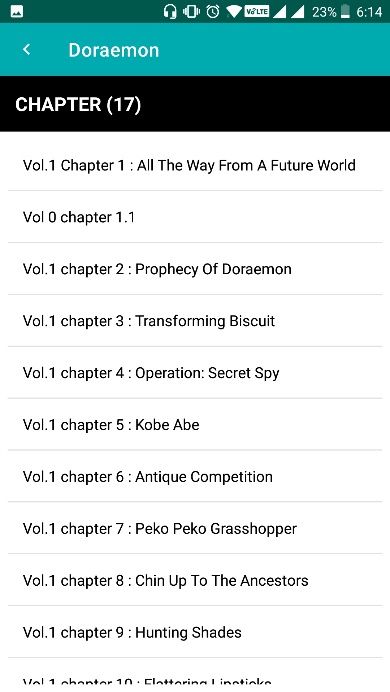












CHAPTER 10

**SOURCE CODE**

package com.example.minikindle;

import android.content.Intent; import android.os.Bundle;

import android.support.annotation.NonNull;

import android.support.v4.widget.SwipeRefreshLayout; import android.support.v7.app.AppCompatActivity; import android.support.v7.widget.GridLayoutManager; import android.support.v7.widget.RecyclerView; import android.view.View;

import android.widget.ImageView; import android.widget.TextView; import android.widget.Toast;

import com.example.minikindle.Adapter.MyComicAdapter; import com.example.minikindle.Adapter.MySliderAdapter; import com.example.minikindle.Common.Common;

import com.example.minikindle.Interface.IBannerLoadDone; import com.example.minikindle.Interface.IComicLoadDone; import com.example.minikindle.Model.Comic;

import com.example.minikindle.Service.PicassoLoadingService; import com.google.firebase.database.DataSnapshot;

import com.google.firebase.database.DatabaseError; import com.google.firebase.database.DatabaseReference; import com.google.firebase.database.FirebaseDatabase; import com.google.firebase.database.ValueEventListener;

import java.util.ArrayList; import java.util.List;

import dmax.dialog.SpotsDialog; import ss.com.bannerslider.Slider;

public class MainActivity extends AppCompatActivity implements IBannerLoadDone, IComicLoadDone {

Slider slider;

SwipeRefreshLayout swipeRefreshLayout; RecyclerView recycler\_comic;

TextView txt\_comic; ImageView btn\_filter\_search;

//Database

DatabaseReference banners,comics;

//Listener

IBannerLoadDone bannerListener; IComicLoadDone comicListener;

android.app.AlertDialog alertDialog; @Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState); setContentView(R.layout.activity\_main);

//init Database

banners = FirebaseDatabase.getInstance(). getReference("Banners");

comics = FirebaseDatabase.getInstance().getReference ("Comic");

//init Listener bannerListener = this; comicListener = this;

btn\_filter\_search = (ImageView)findViewById(R.id. btn\_show\_filter\_search);

btn\_filter\_search.setOnClickListener(new View.

OnClickListener(){

@Override

public void onClick(View v){

startActivity(new Intent(MainActivity.this, FilterSearchActivity.class));

}

});

slider = (Slider) findViewById(R.id.slider); Slider.init(new PicassoLoadingService());

swipeRefreshLayout = (SwipeRefreshLayout) findViewById(R.id.swipe\_to\_refresh);

swipeRefreshLayout.setColorSchemeResources(R.color. colorPrimary,

R.color.colorPrimaryDark);

swipeRefreshLayout.setOnRefreshListener(new SwipeRefreshLayout.OnRefreshListener() {

@Override

public void onRefresh() { loadBanner();

loadComic();

}

});

swipeRefreshLayout.post(new Runnable() { @Override

public void run() { loadBanner(); loadComic();

}

});

recycler\_comic = (RecyclerView)findViewById(R.id. recycler\_comic);

recycler\_comic.setHasFixedSize(true); recycler\_comic.setLayoutManager(new

GridLayoutManager(this,2));

txt\_comic = (TextView)findViewById(R.id.txt\_comic);

}

private void loadComic() {

//Show Dialog

alertDialog = new SpotsDialog.Builder().setContext(

this)

.setCancelable(false)

.setMessage("Please Wait...")

.build();

if(!swipeRefreshLayout.isRefreshing()) alertDialog.show();

comics.addListenerForSingleValueEvent(new ValueEventListener() {

List<Comic> comic\_load = new ArrayList<>(); @Override

public void onDataChange(@NonNull DataSnapshot dataSnapshot) {

getChildren()){ Comic.class);

for(DataSnapshot comicSnapShot:dataSnapshot.

Comic comic = comicSnapShot.getValue( comic\_load.add(comic);

}

comicListener.onComicLoadDoneListener(

comic\_load);

}

@Override

public void onCancelled(@NonNull DatabaseError databaseError) {

Toast.makeText(MainActivity.this, ""+ databaseError.getMessage(), Toast.LENGTH\_SHORT).show();

}

});

}

private void loadBanner() { banners.addListenerForSingleValueEvent(new

ValueEventListener() {

@Override

public void onDataChange(@NonNull DataSnapshot dataSnapshot) {

.getChildren())

String.class);

bannerList);

}

List<String> bannerList = new ArrayList<>(); for(DataSnapshot bannerSnapShot:dataSnapshot

{

String image = bannerSnapShot.getValue( bannerList.add(image);

}

//Call Listener bannerListener.onBannerLoadDoneListener(

@Override

public void onCancelled(@NonNull DatabaseError databaseError) {

Toast.makeText(MainActivity.this, ""+ databaseError.getMessage(), Toast.LENGTH\_SHORT).show();

}

});

}

@Override

public void onBannerLoadDoneListener(List<String> banners) {

slider.setAdapter(new MySliderAdapter(banners));

}

@Override

public void onComicLoadDoneListener(List<Comic> comicList) {

Common.comicList = comicList;

recycler\_comic.setAdapter(new MyComicAdapter( getBaseContext(),comicList));

txt\_comic.setText(new StringBuilder("NEW BOOK (")

.append(comicList.size())

.append(")"));

if(!swipeRefreshLayout.isRefreshing()) alertDialog.dismiss();

}

}

package com.example.minikindle;

import android.content.Intent; import android.os.Handler;

import android.support.v7.app.AppCompatActivity; import android.os.Bundle;

public class SplashScreen extends AppCompatActivity { @Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

new Handler().postDelayed(new Runnable() { @Override

public void run() {

startActivity(new Intent(SplashScreen.this, MainActivity.class));

}

},3000);

}

}

package com.example.minikindle;

import android.os.Build;

import android.support.annotation.RequiresApi; import android.support.v7.app.AppCompatActivity; import android.os.Bundle;

import android.support.v7.widget.DividerItemDecoration; import android.support.v7.widget.LinearLayoutManager; import android.support.v7.widget.RecyclerView;

import android.view.View; import android.widget.TextView;

import android.support.v7.widget.Toolbar;

import com.example.minikindle.Adapter.MyChapterAdapter; import com.example.minikindle.Common.Common;

import com.example.minikindle.Model.Comic;

public class ChaptersActivity extends AppCompatActivity { RecyclerView recycler\_chapter;

TextView txt\_chapter\_name;

LinearLayoutManager layoutManager;

@RequiresApi(api = Build.VERSION\_CODES.LOLLIPOP) @Override

protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.activity\_chapters);

//View

txt\_chapter\_name = (TextView)findViewById(R.id. txt\_chapter\_name);

recycler\_chapter = (RecyclerView)findViewById(R.id. recycler\_chapter);

recycler\_chapter.setHasFixedSize(true); layoutManager = new LinearLayoutManager(this); recycler\_chapter.setLayoutManager(layoutManager); recycler\_chapter.addItemDecoration(new

DividerItemDecoration(this,layoutManager.getOrientation()));

Toolbar toolbar = (Toolbar)findViewById(R.id.toolbar)

;

toolbar.setTitle(Common.comicSelected.Name);

//SetIcon toolbar.setNavigationIcon(R.drawable.

ic\_chevron\_left\_black\_24dp); toolbar.setNavigationOnClickListener(new View.

OnClickListener() {

@Override

public void onClick(View v) { finish();

}

});

fetchChapter(Common.comicSelected);

}

private void fetchChapter(Comic comicSelected) { Common.chapterList = comicSelected.Chapters; recycler\_chapter.setAdapter(new MyChapterAdapter(

this,comicSelected.Chapters));

txt\_chapter\_name.setText(new StringBuilder("CHAPTER (").append(comicSelected.Chapters.size()).append(")"));

}

}

package com.example.minikindle;

import android.support.v4.view.ViewPager; import android.support.v7.app.AppCompatActivity; import android.os.Bundle;

import android.view.View; import android.widget.TextView; import android.widget.Toast;

import com.example.minikindle.Adapter.MyViewPagerAdapter; import com.example.minikindle.Common.Common;

import com.example.minikindle.Model.Chapter; import com.wajahatkarim3.easyflipviewpager. BookFlipPageTransformer;

public class ViewComicActivity extends AppCompatActivity { ViewPager viewPager;

TextView txt\_chapter\_name;

View back,next;

@Override

protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.activity\_view\_comic);

viewPager = (ViewPager)findViewById(R.id.view\_pager)

;

txt\_chapter\_name = (TextView)findViewById(R.id.

txt\_chapter\_name);

back = findViewById(R.id.chapter\_back); next = findViewById(R.id.chapter\_next);

back.setOnClickListener(new View.OnClickListener() { @Override

public void onClick(View view) { if(Common.chapterIndex == 0)

Toast.makeText(ViewComicActivity.this, " You Are Reading First Chapter", Toast.LENGTH\_SHORT).show();

else

{

.chapterIndex));

}

Common.chapterIndex--; fetchLinks(Common.chapterList.get(Common

}

});

next.setOnClickListener(new View.OnClickListener() { @Override

public void onClick(View view) { if(Common.chapterIndex == Common.chapterList.

size()-1)

Toast.makeText(ViewComicActivity.this, "

You Are Reading Last Chapter", Toast.LENGTH\_SHORT).show(); else

{

.chapterIndex));

}

Common.chapterIndex++; fetchLinks(Common.chapterList.get(Common

}

});

fetchLinks(Common.chapterSelected);

}

private void fetchLinks(Chapter chapter) { if(chapter.Links != null)

{

if(chapter.Links.size() > 0)

{

MyViewPagerAdapter adapter = new MyViewPagerAdapter(getBaseContext(),chapter.Links);

viewPager.setAdapter(adapter);

txt\_chapter\_name.setText(Common.formatString (Common.chapterSelected.Name));

//Animation BookFlipPageTransformer

bookFlipPageTransformer = new BookFlipPageTransformer(); bookFlipPageTransformer.

setScaleAmountPercent(10f);

viewPager.setPageTransformer(true, bookFlipPageTransformer);

}

else{

Toast.makeText(this, "No Image Here", Toast.

LENGTH\_SHORT).show();

}

}

else{

Toast.makeText(this, "This Chapter Is Translating...", Toast.LENGTH\_SHORT).show();

package com.example.minikindle;

import android.content.DialogInterface; import android.support.annotation.NonNull; import android.support.design.chip.Chip; import android.support.design.chip.ChipGroup;

import android.support.design.widget.BottomNavigationView; import android.support.v7.app.AlertDialog;

import android.support.v7.app.AppCompatActivity; import android.os.Bundle;

import android.support.v7.widget.GridLayoutManager; import android.support.v7.widget.RecyclerView; import android.view.LayoutInflater;

import android.view.MenuItem; import android.view.View;

import android.widget.AdapterView; import android.widget.ArrayAdapter;

import android.widget.AutoCompleteTextView; import android.widget.EditText;

import android.widget.TextView; import android.widget.Toast;

import com.example.minikindle.Adapter.MyComicAdapter; import com.example.minikindle.Common.Common;

import com.example.minikindle.Model.Comic;

import java.util.ArrayList; import java.util.Collection; import java.util.Collections; import java.util.List;

public class FilterSearchActivity extends AppCompatActivity

{

BottomNavigationView bottomNavigationView; RecyclerView recycler\_filter\_search;

@Override

protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.activity\_filter\_search);

recycler\_filter\_search = (RecyclerView) findViewById (R.id.recycler\_filter\_search);

recycler\_filter\_search.setHasFixedSize(true); recycler\_filter\_search.setLayoutManager(new

GridLayoutManager(this, 2));

bottomNavigationView = (BottomNavigationView) findViewById(R.id.bottom\_nav);

bottomNavigationView.inflateMenu(R.menu.main\_menu); bottomNavigationView.

setOnNavigationItemReselectedListener(new BottomNavigationView.OnNavigationItemReselectedListener() {

@Override

public void onNavigationItemReselected(@NonNull MenuItem menuItem) {

switch (menuItem.getItemId()) { case R.id.action\_filter:

showFiltersDialog(); break;

case R.id.action\_search: showSearchDialog(); break;

default:

break;

}

}

});

}

private void showSearchDialog() {

final AlertDialog.Builder alertDialog = new AlertDialog.Builder(FilterSearchActivity.this);

alertDialog.setTitle("Search Category");

final LayoutInflater inflater = this. getLayoutInflater();

final View search\_layout = inflater.inflate(R.layout

.dialog\_search, null);

final EditText edt\_search = (EditText) search\_layout

.findViewById(R.id.edt\_search);

alertDialog.setView(search\_layout); alertDialog.setNegativeButton("CANCEL", new

DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterfac

dialogInterface, int which) {

dialogInterface.dismiss();

}

});

alertDialog.setPositiveButton("SEARCH",new DialogInterface.OnClickListener(){

@Override

public void onClick(DialogInterface dialogInterface,int i){

fetchSearchComic(edt\_search.getText(

).toString());

}

});

alertDialog.show();

}

private void fetchSearchComic(String query){ List<Comic> comic\_search = new ArrayList<>(); for(Comic comic:Common.comicList)

{

if(comic.Name.contains(query)) comic\_search.add(comic);

}

if(comic\_search.size() > 0) recycler\_filter\_search.setAdapter(new

MyComicAdapter(getBaseContext(),comic\_search)); else

Toast.makeText(this, "No result", Toast.

LENGTH\_SHORT).show();

}

private void showFiltersDialog() {

final AlertDialog.Builder alertDialog = new AlertDialog.Builder(FilterSearchActivity.this);

alertDialog.setTitle("Select Category");

final LayoutInflater inflater = this. getLayoutInflater();

final View filter\_layout = inflater.inflate(R.layout

.dialog\_options, null);

final AutoCompleteTextView txt\_category = ( AutoCompleteTextView) filter\_layout.findViewById(R.id. txt\_category);

final ChipGroup chipGroup = (ChipGroup)

filter\_layout.findViewById(R.id.chipGroup);

//Create Autocomplete

ArrayAdapter<String> adapter = new ArrayAdapter<>( this, android.R.layout.select\_dialog\_item, Common.categories

);

txt\_category.setAdapter(adapter); txt\_category.setOnItemClickListener(new AdapterView.

OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

//Clear txt\_category.setText("");

//Create tags

Chip chip = (Chip)inflater.inflate(R

.layout.chip\_item,null,false);

chip.setText(((TextView)view).

getText());

chip.setOnCloseIconClickListener(new

View.OnClickListener() {

@Override

public void onClick(View view) { chipGroup.removeView(view);

}

});

chipGroup.addView(chip);

}

});

alertDialog.setView(filter\_layout); alertDialog.setNegativeButton("CANCEL", new

DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialogInterface, int which) {

dialogInterface.dismiss();

}

});

alertDialog.setPositiveButton("FILTER", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface

dialog, int which) {

ArrayList<>(); StringBuilder(""); getChildCount();j++)

getChildAt(j); toString());

are sorted A->Z

",");

List<String> filter\_key = new StringBuilder filter\_query = new for(int j=0;j<chipGroup.

{

Chip chip = (Chip)chipGroup. filter\_key.add(chip.getText().

}

//Because in our Database Chapters

//So we need sort our filter\_key Collections.sort(filter\_key);

//Convert list to string for(String key:filter\_key)

{

filter\_query.append(key).append(

}

length()-1);

//Remove Last "," filter\_query.setLength(filter\_query.

toString());

}

});

//Filter by this query fetchFilterCategory(filter\_query.

alertDialog.show();

}

private void fetchFilterCategory(String query) { List<Comic> comic\_filtered = new ArrayList<>(); for(Comic comic:Common.comicList){

if(comic.Category !=null) if(comic.Category.contains(query)) comic\_filtered.add(comic);

}

if(comic\_filtered.size() > 0) recycler\_filter\_search.setAdapter(new

MyComicAdapter(getBaseContext(),comic\_filtered)); else

Toast.makeText(this, "No result", Toast.

LENGTH\_SHORT).show();

}

}

CHAPTER 11

**CONCLUSION**

**CHAPTER 11**

**CONCLUSION**

Mini Kindle is a simple application to let the users read the magazines,books or comics whenever they want or depends what they want.

The filter search is available with the specified requirements in the provided system.

This decreases almost all the disadvantages of searching a book for longer time.

This system helps the user to provide suggestions and read all the required book available at the recent versions .

Provides security to the user’s information.

Mini Kindle lets the user to interact with an easy UI, letting them interact with the application easily.

CHAPTER 12

**FUTURE SCOPE**

**CHAPTER 12**

**FUTURE SCOPE**

Future Scope means the forms our project can take in the future. It’s the prediction of the successes of our project.

**The Future Scope of our project is as follow:**

* User will be able to provide a bookmark in the future updates of the project.
* User will be able to save the books offline.
* Extending this application by providing Authorization service.
* Sending user the updates via Email or a text message.
* Making the application compatible on various platforms i.e. iOS, windows,etc.

CHAPTER 13

**BIBLIOGRAPHY**

**CHAPTER 13**

**BIBLIOGRAPHY**

1. **www.google.com**
2. **www.wikipedia.org**
3. **www.tutorialpoints.com**
4. **www.w3schools.com**
5. **www.w3schools.in**
6. [www.youtube.com](https://www.youtube.com)
7. [www.youtube.com](https://www.youtube.com/watch?v=fgYsc9sft-U)

h)  **http://www.pcspl.in/**

CHAPTER 14

**TESTING**

**CHAPTER 14**

**TESTING**

**13.1 Testing Details**

Testing is a verification process for quality assessment and improvement. Testing is basically done to find errors, faults in the system. The basic goal of software development process is to produce the software that has very few or no errors. In an effort to detect errors soon after they are introduced each phase ends with verification activity such as reviews. However most of these verification activities in the early phase of the software development are based on human evaluation and cannot detect all the errors. Testing plays an important role in quality assurance for the software. It is a dynamic method for the verification and validation, where the system to be tested is executed and the behavior of the system is observed.

**13.1.1 Black box testing**

Black-box testing is a method of[software testing](http://en.wikipedia.org/wiki/Software_testing) that examines the functionality of an application (e.g. what the software does) without peering into its internal structures or workings (see  [white-box testin](http://en.wikipedia.org/wiki/White-box_testing)g). This method of test can be applied to virtually every level of software testing:  [unit, integration system](http://en.wikipedia.org/wiki/Unit_test) and  [acceptance.](http://en.wikipedia.org/wiki/Acceptance_test) It typically comprises most if not all higher level testing, but can also dominate  [unit testing](http://en.wikipedia.org/wiki/Unit_testing) as well.

**13.1.2 White box testing**

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing software  that tests internal structures or workings of an application, as opposed to its functionality (i.e. black-box testing). In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs.

**13.1.3 Unit testing**

Unit testing, also known as component testing, refers to tests that verify the functionality of a specific section of code, usually at the function level. In an object-oriented environment, this is usually at the class level, and the minimal unit tests include the constructors and destructors.

These types of tests are usually written by developers as they work on code (white-box style), to ensure that the specific function is working as expected. One function might have multiple tests, to catch corner cases or other branches in the code. Unit testing alone cannot verify the functionality of a piece of software, but rather is used to ensure that the building blocks of the software work independently from each other.

Unit testing is a software development process that involves synchronized application of a broad spectrum of defect prevention and detection strategies in order to reduce software development risks, time, and costs. It is performed by the software developer or engineer during the construction phase of the software development life cycle. Rather than replace traditional QA focuses, it augments it. Unit testing aims to eliminate construction errors before code is promoted to QA; this strategy is intended to increase the quality of the resulting software as well as the efficiency of the overall development and QA process.

Depending on the organization's expectations for software development, unit testing might include static code analysis, data flow analysis, metrics analysis, peer code reviews, code coverage analysis and other software verification practices.

**13.1.4 Integration Testing**

Integration testing is any type of software testing that seeks to verify the interfaces between components against a software design. Software components may be integrated in an iterative way or all together ("big bang"). Normally the former is considered a better practice since it allows interface issues to be located more quickly and fixed.

Integration testing works to expose defects in the interfaces and interaction between integrated components (modules).

**13.1.5 Component interface testing**

The practice of component interface testing can be used to check the handling of data passed between various units, or subsystem components, beyond full integration testing between those units. The data being passed can be considered as "message packets" and the range or data types can be checked, for data generated from one unit, and tested for validity before being passed into another unit. One option for interface testing is to keep a separate log file of data items being passed, often with a time stamp logged to allow analysis of thousands of cases of data passed between units for days or weeks. Tests can include checking the handling of some extreme data values while other interface variables are passed as normal values. Unusual data values in an interface can help explain unexpected performance in the next unit. Component interface testing is a variation of black-box testing, with the focus on the data values beyond just the related actions of a subsystem component.

**13.1.6 System testing**

System testing, or end-to-end testing, tests a completely integrated system to verify that it meets its requirements.For example, a system test might involve testing a logon interface, then creating and editing an entry, plus sending or printing results, followed by summary processing or deletion (or archiving) of entries, then logoff.

In addition, the software testing should ensure that the program, as well as working as expected, does not also destroy or partially corrupt its operating environment or cause other processes within that environment to become inoperative (this includes not corrupting shared memory, not consuming or locking up excessive resources and leaving any parallel processes unharmed by its presence).

**13.1.7 Alpha Testing**

Alpha testing is a type of acceptance testing; performed to identify all possible issues/bugs before releasing the product to everyday users or public.  The focus of this testing is to simulate real users by using blackbox and whitebox techniques.

The aim is to carry out the tasks that a typical user might perform. Alpha testing is carried out in a lab environment and usually the testers are internal employees of the organization. To put it as simple as possible, this kind of testing is called alpha only because it is done early on, near the end of the development of the software, and before beta testing.

**13.1.8 Beta Testing**

Beta Testing of a product is performed by "real users" of the software application in a "real environment" and can be considered as a form of external user acceptance testing.

 Beta version of the software is released to a limited number of end-users of the product to obtain feedback on the product quality. Beta testing reduces product failure risks and provides increased quality of the product through customer validation.

It is the final test before shipping a product to the customers. Direct feedback from customers is a major advantage of Beta Testing. This testing helps to tests the product in real time environment.



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

CHAPTER 15

**BUG REPORT**

**CHAPTER 15**

**BUG REPORT**

**GROUP. BUG:** - #0001

**SOFTWARE: MiniKindle (Online Library App)**

**RELEASE VERSION:** 1.0

**TESTER: Sagar Soni** **DATE:** 25-02-2019

**ASSIGNED TO**: Atharva Keny **PRIORITY:** 1

**TITLE:** App does not refresh after swipe up.

**DESCRIPTION:**

When the user swipes up to refresh the home screen to load new books into recommendations, it continues to load and then eventually crash.

**RESOLUTION:** FIXED

**DATE RESOLVED:** 29-02-2019 **RESOLVED BY: Atharva Keny**

**VERSION:** 1.0

**RETESTED BY: Atharva Keny VERSION TESTED:**1.0

**RETEST COMMENT:** THE BUG WAS FIXED PROPERLY AND THERE ARE NO MORE ISSUES OF VALIDATION.

**SIGNATURES**

**ORIGINATOR: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ TESTER: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**PROGRAMMER: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ PROJECT MANAGER: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**