CHAPTER 1

INTRODUCTION

1.1 Overview

This project is aimed at upgrading the current system of SRM University library pertaining to renewal of library books. Android is the widely used operating system in present smartphones and thus developing application based on this platform will make it available to majority of students. The project implements a simple idea of book renewal process by migrating from the current system to a more flexible, easy to use platform. The application consists of the android as front end and oracle database that acts as the backend which processes queries. The major task carried out is of scanning barcodes of books and student id. This data is enough to retrieve the list of books issued by the student as well as carry out the task of renewing them. Students first need to register to the application. During registration, the authenticity of a student is checked from SRM University student portal. If authentication is successful then the student data is entered into the database. Students can now login and access the required information from any android mobile device. Once students login into the application they are directly directed to the list of books issued by them and the renewal date for each book is also displayed. Students can now select a book and renew it, if it is not past its renewal date. In case of a book being overdue, student will have to take it to the library, pay the incurred fine and then renew the book. The same process needs to be carried out while returning any book.

Another advantage of the android application is the use of GCM for generating user notifications that will notify the students about the renew dates of the book. A scheduler is responsible for running a script daily to carry out the task. Also once a student logins in the application, his registration number and password is stored in shared preferences so that repetitive logins are not required every time the application is closed. Consequently, only those users that are currently logged in will get cloud notifications. Other features include the splash screen that displays SRM University logo while carrying out the task of checking data stored in shared preferences, in background and directing users to corresponding activity.

This application is first of its kind being developed that addresses the current situation in our college and can be easily upgraded in future to include more services. Detailed, best practices implementation methodology and processes ensure that the application adheres to correct standards, offer optimum functionality and is bug free. The application can be customized to offer similar services in other colleges as well which use identical method in their college libraries, and at the same time ensuring smooth transition from current system to the android platform.

1.2 Problem Statement

The current system used in the college library lacks mobility. Students are required to visit the library to renew a book. This project is aimed at upgrading this system so as to make it flexible and provide it the necessary mobility that it lacks.

The present system is an application developed in visual basic that acts as a frontend for the library database. It is handled by the library administrators. Book renewal and issue is done using this application and hence each time a student needs to renew a book, he needs to visit the library. Another application developed in visual basic running on different system enables students to search and check availability of different books. This too mandates a student's actual physical presence in the library in order to interact with the application. This project addresses the first problem of book renewal, while the second problem of checking availability of books is not addressed but considered in future improvements that may be added to the android application.

Another issue that needs to be addressed is the availability of a static IP address for the system on which the server is made to run. At present the VB application interacts with local database using a server that runs on local network. Internet connection is available but provides a dynamic IP address which cannot be used by our android application.

1.3 Objective

This project is an implementation of the idea that students should be able to carry out tasks related to library like book renewal or search at any time or from any location using the latest available technology with ease and convenience. Android is an operating system based on the Linux kernel, and designed primarily for touchscreen mobile devices such as smartphones and tablet computers. Initially developed by Android, Inc., which Google backed financially and later bought , Android was unveiled along with the founding of the Open Handset Alliance, a consortium of hardware, software, and telecommunication companies devoted to advancing open standards for mobile devices. Android's source code is released by Google which allows everyone to freely modify and distribute Android under various names except by reusing the original trademark; however, device manufacturers and wireless carriers have licensed the trademark from Google. Most Android devices ship with a combination of open source and proprietary software. Android is popular with technology companies which require a ready-made, low-cost and customizable operating system for high-tech devices. This project aims to achieve the following objectives:

- To develop an application based on android that will enable library administrators to work dynamically as well as bring convenience to students.

- To achieve essential mobility that is lacking in the task of book renewal on account of current system.

- To enable the college to adopt a cloud based platform by implementing Google Cloud Messenger in the application for the purpose of generating notifications.

- To make it easier for the college to introduce other similar improvements in future by providing the necessary groundwork achieved by developing an application based on android platform.

1.4 Organization of Report

1.4.1 Introduction

The basic description of the project along with the problem encountered in existing system has been explained. Discussion about the solution of the problem has also been done.

1.4.2 Literature Survey

It is basically reading different papers and analysing the problems encountered in the existing system.

1.4.3 System design

The basic introduction about the design of whole project has been addressed. The complete architecture of the proposed system is flowchart and Use case Diagram.

1.4.4 Module Description

Description of all the three modules implemented in the project. Flowchart, Work Flow diagrams for each module has been provided.

1.4.5 System Implementation

Overall description of the platform used to make project. Simulating parameters used in the project Sample Coding of the project. Screenshot of the project activities have been provided.

CHAPTER 2

LITERATURE SURVEY

2.1 Introduction

This chapter deals with the various aspects of the present system that has been employed by our college library in order to handle tasks relating to management of the books like issue and renewal, searching, checking availability of books etc. It also addresses the issue of shortcomings of this system and the advantages of implementing a new system based on android platform.

The current system is library management system based in VB platform and hence the proposed system is more inclined towards its implementation aspect rather than being research based. As this is first of its kind implementation of solving the problem by upgrading the current system, very few research papers were identified that might aid in achieving the solution. Information was gathered by visiting the college library and knowing how the current system has been implemented.

Majority of the features that were added to the project were searched and tested using online resources. Origin and advantages of such features has been clearly discussed in this chapter. The research papers that were studied were related to the advantages of using JSON for data exchange and using it for implementing REST architecture. This aspect will be addressed more clearly in this chapter's summary.

The next topic deals with the existing system and its features, followed by its shortcomings and how the proposed system addresses those issues.

2.2 Existing System

The present system installed in our college library is VB based application that manages tasks like book issue and renewal. It also performs additional functions like searching for books by name, author name etc. as well as checking their availability. It is connected to local oracle 9i database that stores and retrieves all the information. As the entire system is local, no web server has been used and there is not transfer of data via internet.

A scanner is connected to the system on which the library application has been installed. It is operated by library administrators. The entire procedure of issuing and renewing books is carried out by them. In order to issue or renew a book, student needs to provide the book as well as his student ID card. The administrator scans the barcode on the ID and the book. The application then fetches relevant data using the book number and ID number as data in queries. The application performs three major functions i.e. book issue, renewal and return.

The VB application comprises the frontend of the management system. The oracle database handles all queries and stores data. It has many different tables that store specific kind of data. Some of the tables have been listed as follows:

- ISSUE: This table stores data pertaining to the books issued by students. Entries are made for each book issued by each student. Every book is identified by a unique number. Copies of same book have different id numbers as well.

- LIBMAIN: This table retains information of each book like title, author name, ISBN number, number of pages etc. Any data on a book can be obtained from this table.

- SCARD: This table stores data of students that have registered to the library. It contains the student name, registration number, departments among other details

- RENEWAL: It contains list of books that are renewed by students.

Data is exchanged between the VB application and database locally. Our project aims at using the same database for storing and retrieving information, by using WAMP web server. To reduce complexity, data from android devices is sent to the webserver in the form of JSON. The webserver handles all the queries that are used for interacting with oracle database.

2.3 Issues in Existing System

This section explains the drawbacks of existing system and also explains the implementation of JSON for data exchange between the android devices and severs.

One of the major drawbacks of current system employed by college library is the inconvenience caused to students. In order to renew a book it is mandatory for students to visit the library. As a result day scholars need to carry books, which are often heavy throughout the entire duration of college. Another disadvantage is the lack of timely remainders. Students tend to forget the renewal dates and end up paying a considerable amount as fine. Library administrators also face considerable amount of work load while renewing books issued by students.

Another important issue is the type of mode to be implemented for data transfer between the application and server. The paper ‘Using JSON for Data Exchanging in Web Service Applications’ by Dunlu PENG, Lidong CAO, Wenjie XU discusses the advantages of using JSON for exchange of web data and deals with how it is implemented. It addresses the topic of performance comparison between JSON and XML, as preferred mode of data exchange.

The paper ‘On Using JSON-LD to Create Evolvable RESTful Services’ by Markus Lanthaler and Christian Gütl, discusses the implementation of JSON for building REST services. It was referred for the purpose of understanding how JSON and REST services work together effectively.

2.4 Summary of Literature Survey

This chapter gave an overview of the existing system and discussed its features and implementation details. It also addressed the drawbacks in the existing system and their disadvantages to students and library administrators. The papers referred for improving the project, were also discussed.

It is necessary to carry an actual survey for such kind of projects that involve development of real time applications. The very first task performed was to get an idea about how the current system employed in the library works and for this purpose, the library was visited. Library administrators explained the details and working of the system, the technology used. They also provided information about the database involved. The next task was comprised of comparative study of any similar applications. The different available applications available in android were studied, for understanding their functioning, design, user convenience offered by them and so on. Finally list of requirements was generated that penned down the database and hardware requirements.

CHAPTER 3

SYSTEM DESIGN

3.1 Introduction

System Design helps in better understanding and visualization of the project by providing structures like work flow diagram, flow chart etc. It explains how various modules of the project interact with each other and the functions performed by them as individual units and as a single system. The quality of a project can be easily determined by examining the system designs. It decides the methodology that needs to be followed while handling the system. System design should be such that the occurrences of errors are minimum and the flow of data through different modules, accurate.

The system flow chart deals with how various components interact with each other and what data processing occurs. The flowchart for frontend explains in detail, functioning of the application on android devices. It deals with how data in processed in the activities and in what order.

The flowchart for backend helps to visualize the functioning of the webserver and oracle database. It explains the process where server gets requests from the application and interacts with server database to retrieve required information. This data is encoded in JSON form and returned to the application.

The work flow diagram explains the interaction between all the frontend and backend modules and the flow of control throughout the entire system. It also explains the how the system interacts with GCM. It helps in better understanding of the system as a whole as well as how various modules gain control when specific events are triggered.

3.2 System Architecture

3.2.1 Flow chart

3.2.1.1 Frontend

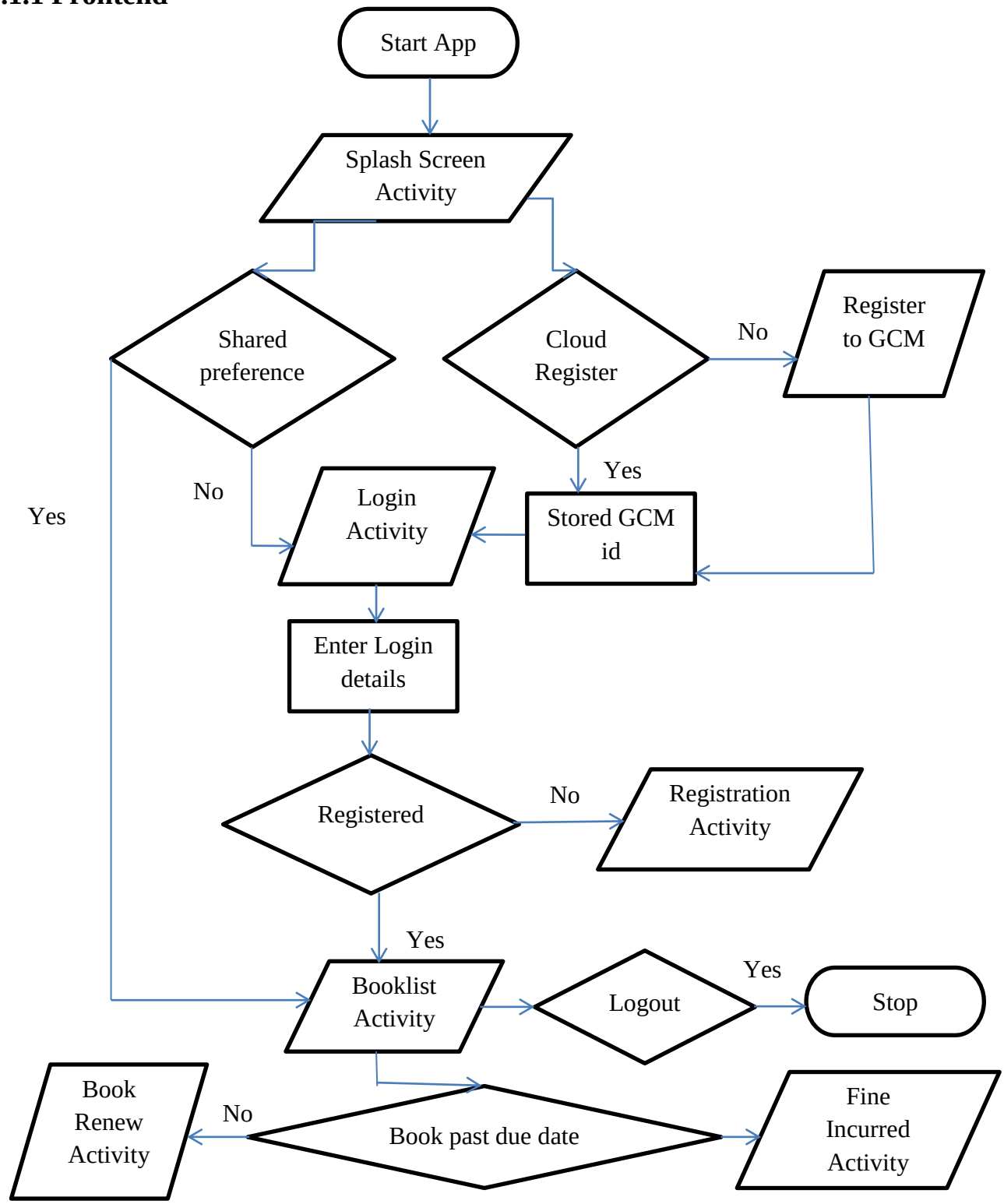

Fig 3.1 System Frontend Flowchart

3.2.1.2 Backend

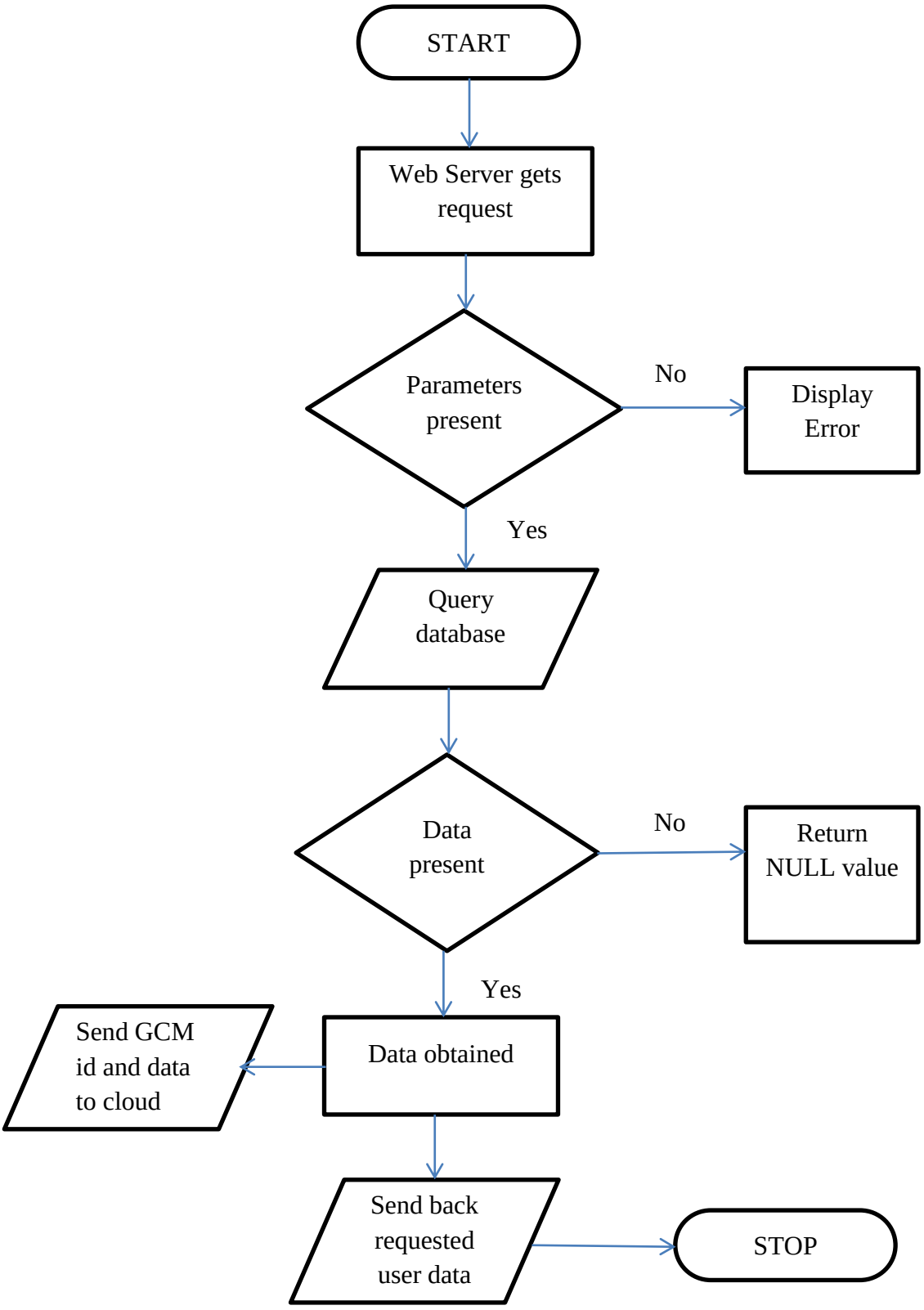

Fig 3.2 System Backend Flowchart

3.2.2 Work flow Diagram

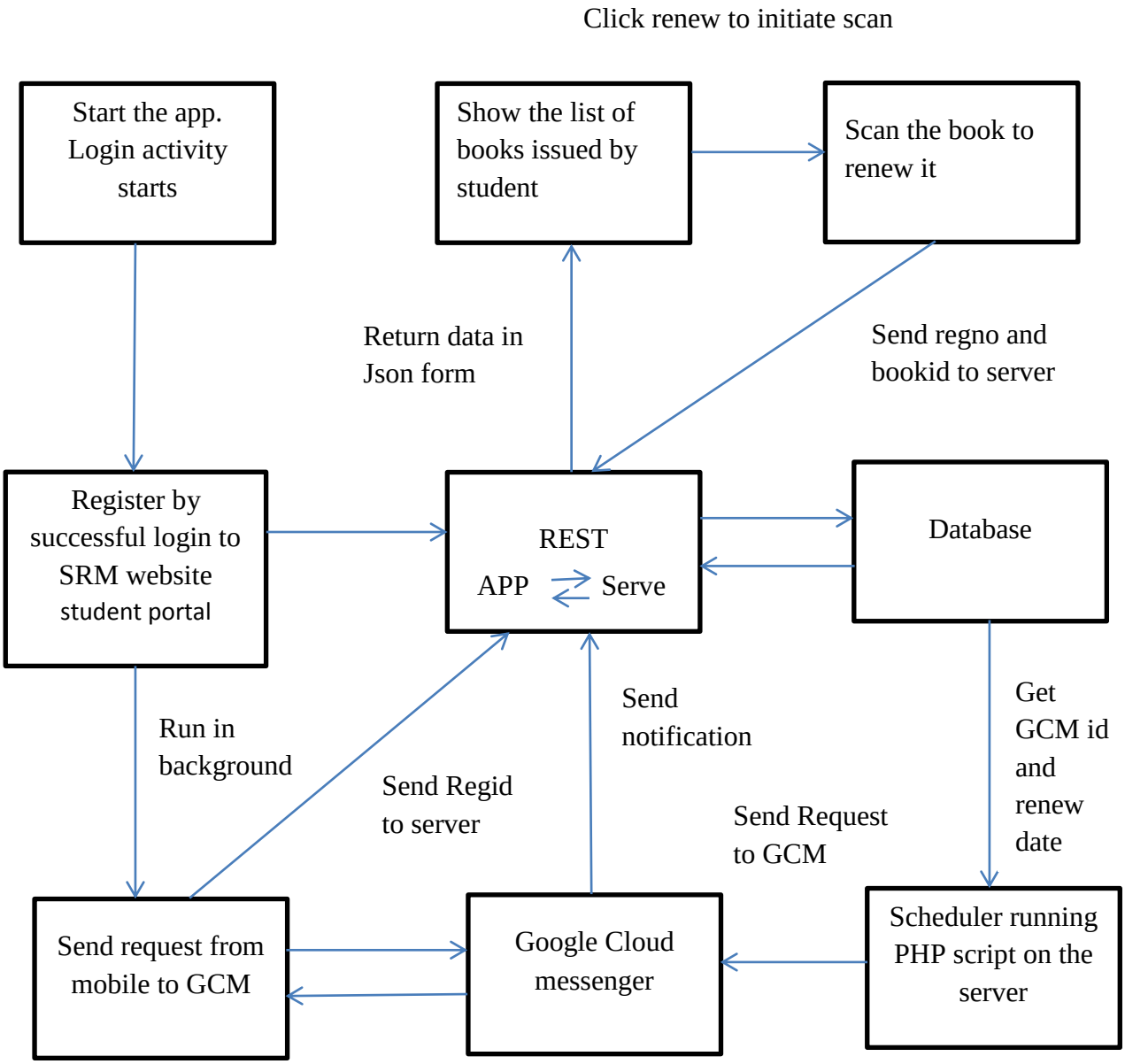

Fig 3.3 System Work Flow Diagram

3.2.3 Use Case Diagram

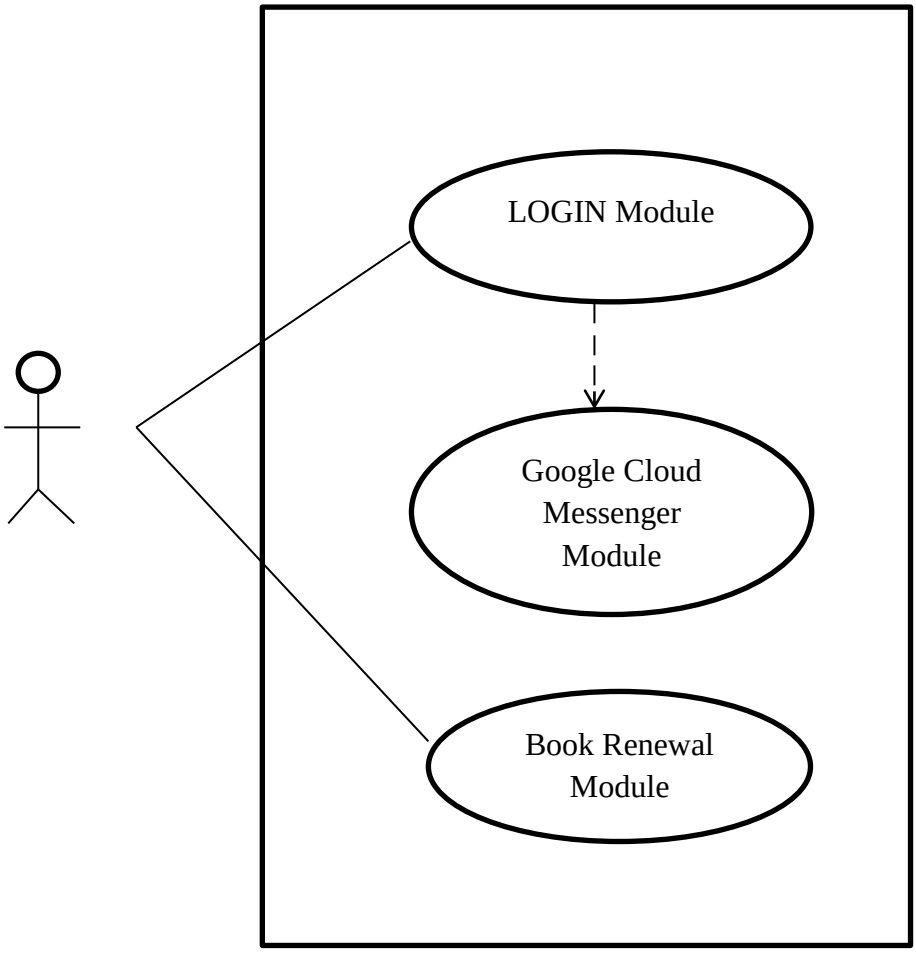

Fig 3.4 System Use Case Diagram

3.2.4 Description

This project is based on android platform. It is an app that enables students to renew their library book using any smartphone that runs the android OS. The three basic modules that are at its core are Login and Registration module, Book renewal module and GCM module. Each of them is explained in detail in the following chapter.

The first activity to start is the ‘Splash Screen’ that displays the SRM logo and simultaneously checks for login details in shared preferences. It also checks whether the device has registered to GCM and if not, the registration process occurs. If the login details are present then the user is directed to the ‘Book list’ activity, else the ‘Login and Registration’ activity starts. Users can now register themselves and login into the application. During registration the credentials for SRM student portal is sent to the server, where a PHP script checks their validity using CURL library. If successful, the registration process occurs and user data is stored in library server. Users can also scan their ID cards to register or login.

Android is developed in private by Google until the latest changes and updates are ready to be released, at which point the source code is made available publicly. This source code will only run without modification on select devices, usually the Nexus series of devices. The source code is, in turn, adapted by OEMs to run on their hardware. Android's source code does not contain the often proprietary device drivers that are needed for certain hardware components. The green Android logo was designed for Google in 2007 by graphic designer Irina Blok. The design team was tasked with a project to create a universally identifiable icon with the specific inclusion of a robot in the final design. After numerous design developments based on science-fiction and space movies, the team eventually sought inspiration from the human symbol on restroom doors and modified the figure into a robot shape. As Android is open-sourced, it was agreed that the logo should be likewise, and since its launch the green logo has been reinterpreted into countless variations on the original design.

Once login is successful, users get a list of books issued by them. The application implements REST architecture i.e. it sends GET or POST request to library web server. PHP script running on the server polls the database and retrieves list of books for the registration ID sent by the server. That data is encoded in JSON format by the script and sent back to the application. The app uses a list view adapter to display the book names along with their renewal dates. On selecting a book it is user is directed to ‘Book Renewal’ or ‘Fine incurred’ activity based on whether it is past it due date or not. Notifications pertaining to renewal dates are sent from server by using GCM registration IDs of registered devices.

3.3 System Requirements

As the project is a real time application, the requirements are based on the current system used by college library

3.3.1Hardware Requirements:

System	:	Intel core i3, Smartphone
Monitor	:	15 VGA colour
Mouse	:	Logitech
RAM	:	512 Mb (Minimum)
Memory	:	64 GB (Minimum)

3.3.2 Software Requirements:

Operating System	:	Windows XP, Android (min 2.3.3, max 4.4)
Database	:	Oracle 9i, Oracle Instant Client (10gR2)
Server	:	Apache (WAMP)
Script	:	PHP
IDE	:	Eclipse, Android SDK

GCM (Google Cloud messenger) will be used for generating the notifications.

Zxing open source library for scanner activity

Server will require a static IP address.

3.4 Summary

This chapter dealt with the flow chart and work flow of the project, inspecting and discussing its various modules and how they interact with each other. The flow of control between various modules was studied as well.

The project requirements were also stated within this chapter. Android can be subdivided into four main layers: the kernel, libraries, applications framework, and applications. As previously mentioned the kernel is Linux. The libraries that come with Android provide much of the graphics, data storage, and media capabilities. Embedded within the libraries layer is the Android runtime which contains the Dalvik virtual machine, which powers the applications. The applications framework is the API that all applications will use to access the lowest level of the architecture. The various hardware and software requirements were stated. Other requirements like external libraries and type of internet connectivity were also given.

CHAPTER 4

MODULE DESCRIPTION

4.1 Introduction

The android application project consists of three major modules which form its core.

The login and registration module is the first to activate, every time the application starts. It provides users with functions like login, registration, login and registration by scanning id. The data after registration is sent to library server, so that users can login from any device that has the application installed. The second module performs the function of displaying the list of books issued by users from SRM college library. It also provides the function of renewing those issued books or informs the user that their book has incurred a fine if it is past due date. The GCM module interacts with the free cloud service provided by Google to generate and deliver notifications to registered android devices. It first registers the device with GCM service and the GCM ID is sent to server. The server uses it to send data in the form of notifications to the android device.

These modules interact with each other to offer desired services to users. Each of them is explained in detail with corresponding flowcharts and work flow diagrams during the course of this chapter.

4.2 List of Modules

Module 1: Login and Registration
Module 2: Book renewal
Module 3: GCM module

4.3 Login and Registration Module

4.3.1 Flow chart

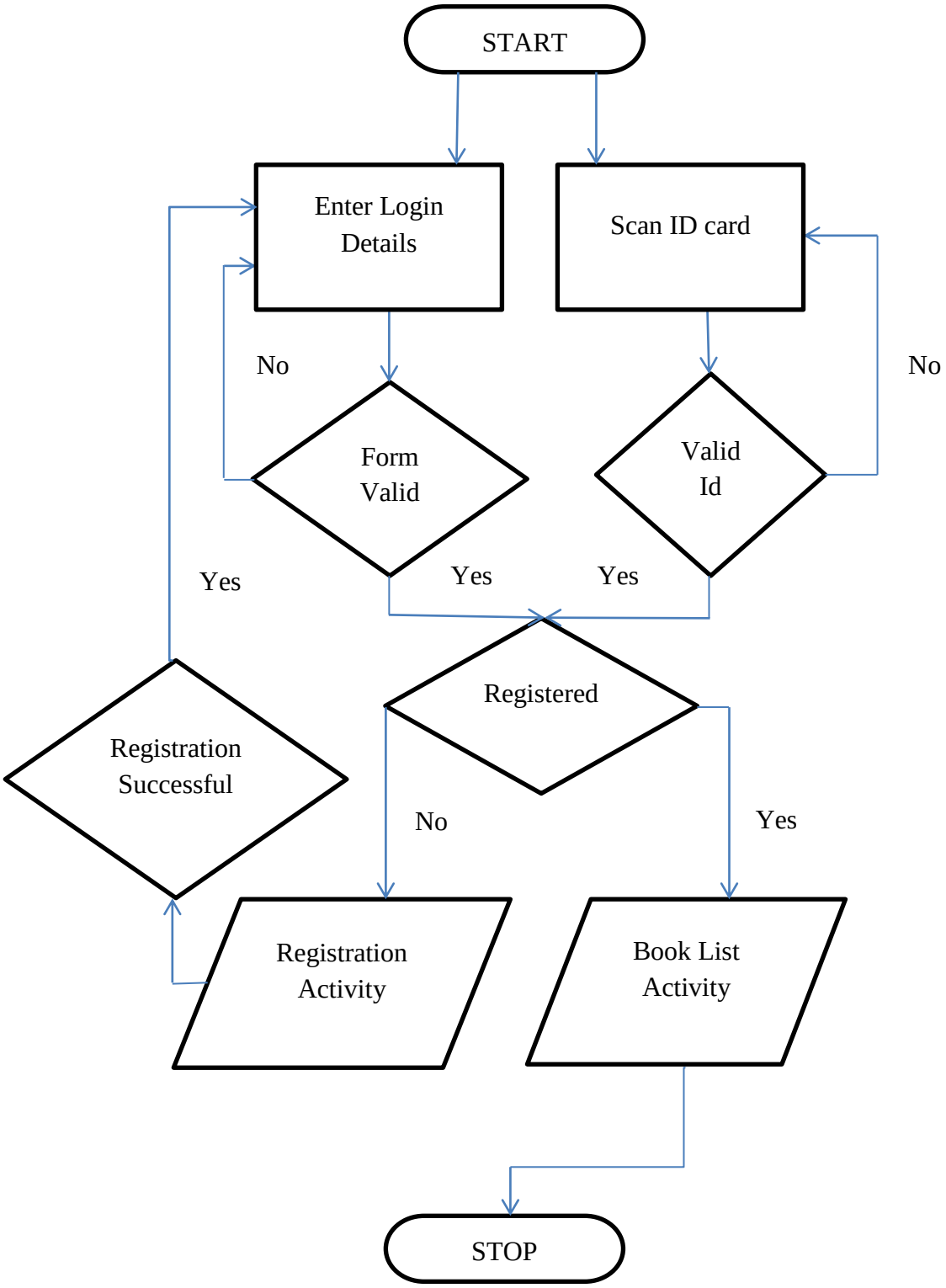

Fig 4.1 Login Module Flowchart

4.3.2 Work flow Diagram

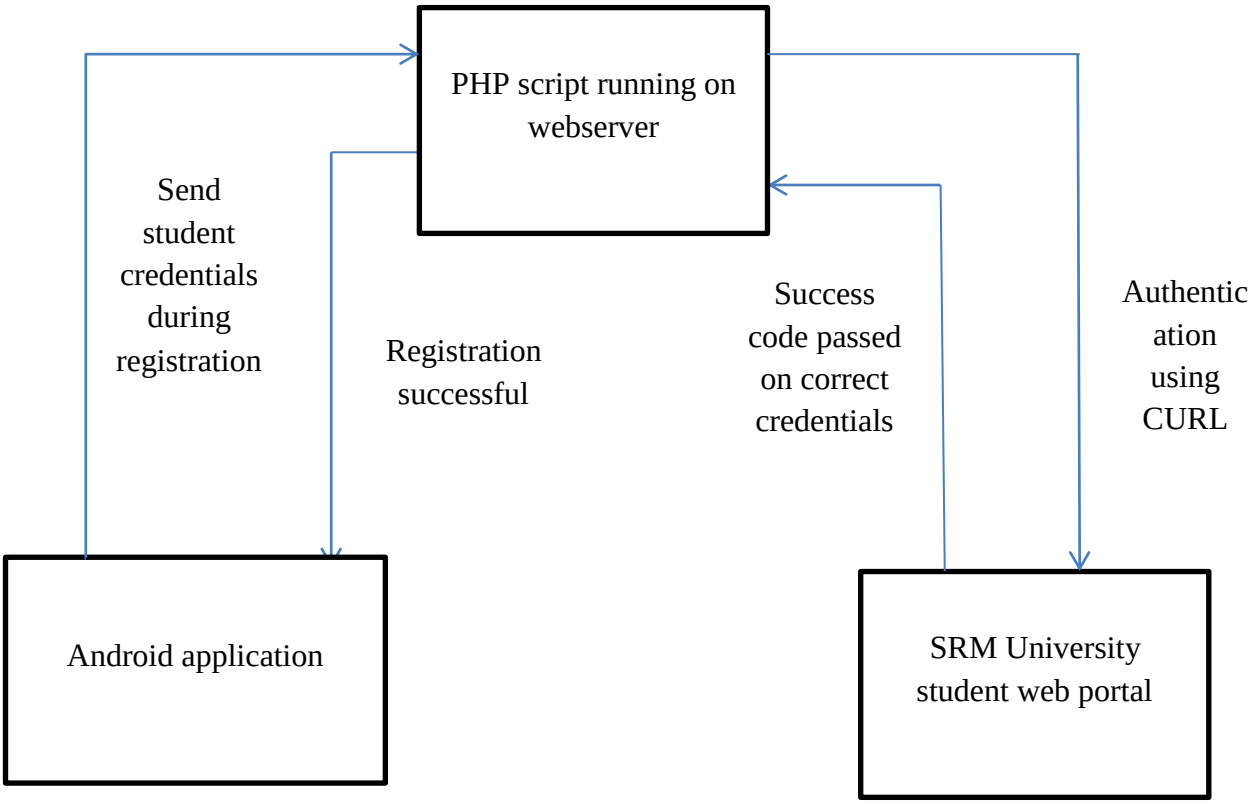

Fig 4.2 Login Module Work flow Diagram

4.3.3 Description

The Login and Registration module enables students to create a unique account on the application, which makes it possible to access their data from any android device on which the application has been installed. Notifications are sent to the device where the student is currently logged in. There are two ways of logging or registration and a user may choose any method based on convenience. On successful registration, students can login using their own password.

As the application starts, it first checks whether any registration number and password has been stored in shared preferences. Shared preferences are useful in storing login data so that once logged in, a user need not login in again if the application is closed. If the user wishes to

end his session, then they can do so by opting to logout, in which case no data will be stored in shared preferences. If relevant data is found in shared preferences, then the user is redirected directly to the booklist activity. If no data is found in shared preference, it indicates that the user has logged out and hence no session has been maintained. In such condition, users will be directed to the login activity. The booklist activity can now be reached by entering proper credentials, or by scanning valid student ID.

If the user is using the application first time, then they need to register themselves first. During registration a student needs to enter their SRM University student portal credentials, email, mobile number and a password. Registration will be successful if the user is an actual student of SRM University and the registration form validation is successful. Any blank fields will prompt error messages. If user chooses to scan their ID for logging in, then they will be directed to the registration activity when a valid ID is scanned for the first time. Android applications run in a sandbox, an isolated area of the system that does not have access to the rest of the system's resources, unless access permissions are explicitly granted by the user when the application is installed. Before installing an application, Play Store displays all required permissions: a game may need to enable vibration or save data to an SD card, for example, but should not need to read SMS messages or access the phonebook. After reviewing these permissions, the user can choose to accept or refuse them, installing the application only if they accept. The sandboxing and permissions system lessens the impact of vulnerabilities and bugs in applications, but developer confusion and limited documentation has resulted in applications routinely requesting unnecessary permissions, reducing its effectiveness.[136] Google has now pushed an update to Android Verify Apps feature, which will now run in background to detect malicious processes and crack them down.

All the registration details of the student are sent to library server, so that a user is not restricted to as single android device. They can use any device as long as it has the application installed and the login details are accurate.

4.4 Book Renewal Module

4.4.1 Flow chart

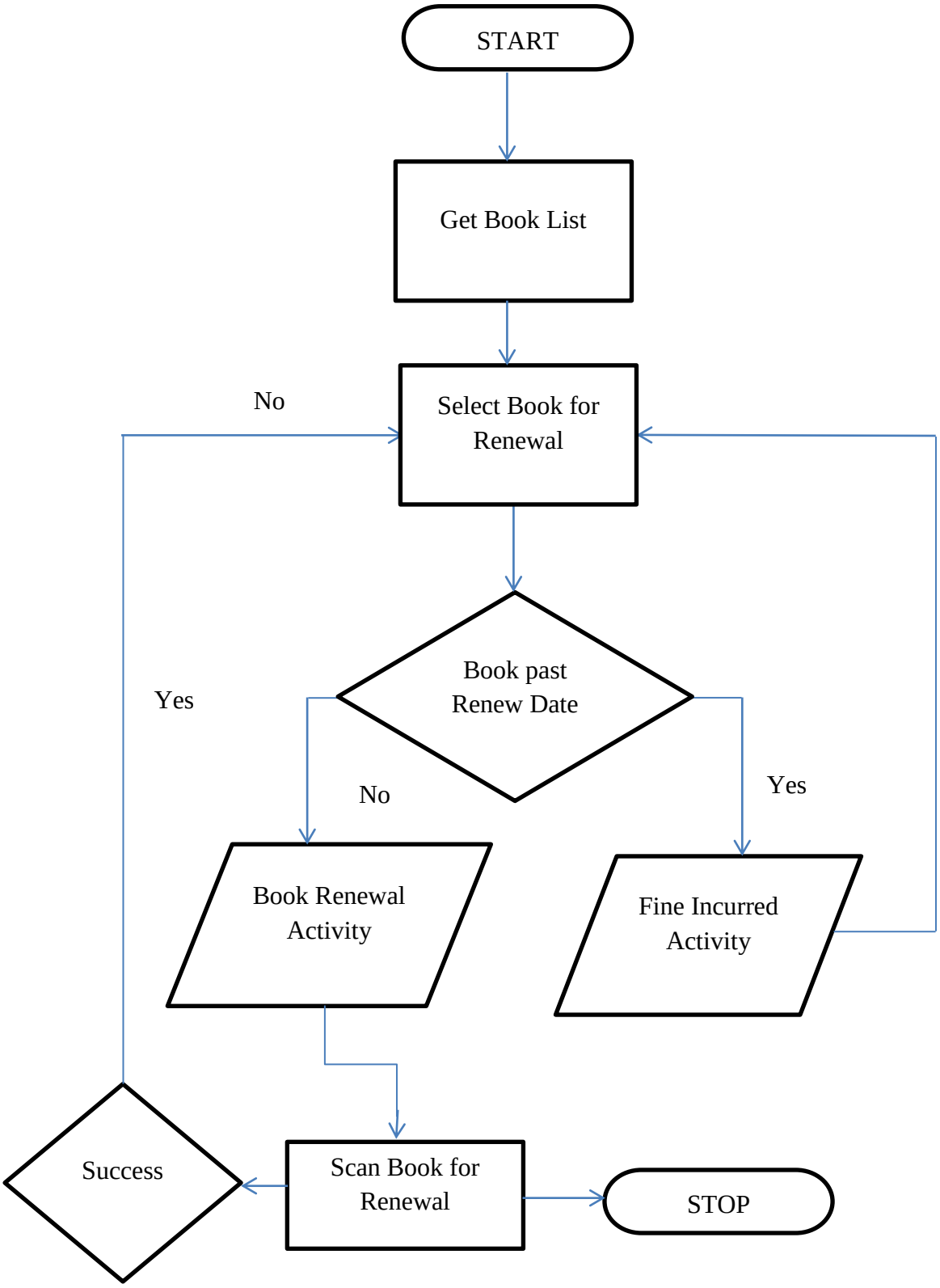

Fig 4.3 Book Renewal Module Flowchart

4.4.2 Work Flow Diagram

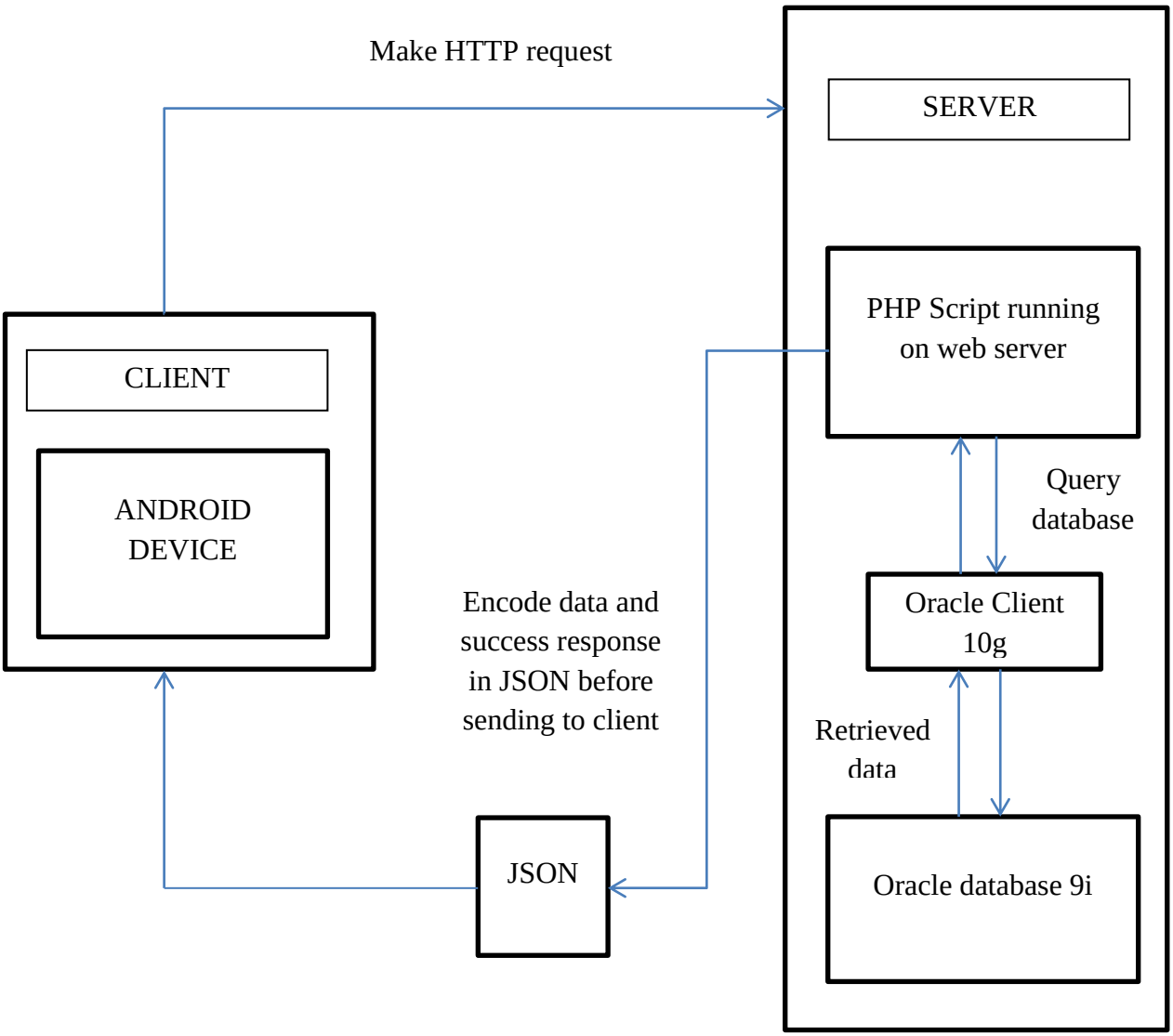

Fig 4.4 Book Renewal Module Work flow diagram

4.4.3 Description

This module performs the task of displaying the list of books issued by the user and allows them to renew those books. This module starts directly if login details are stored in shared preferences; else users need to login first.

The booklist is displayed using listview adapter. This adapter arranges the data in a scrollable list format. When this activity starts, a HTTP GET request is sent to the library server with the book ID and student registration number as parameters. GET is a read-only operation. It can be repeated without affecting the state of the resource (idem-potent) POST is a read-write operation and may change the state of the resource and provoke side effects on the server. This data is encoded in JSON format before being sent. When the web server receives this request, the PHP script running on it selects a mode for fetching book list from the database. This data is then encoded in JSON and returned to the application. Each book name and its corresponding renewal date are displayed for every list component. The list is clickable i.e. on clicking a list component, a new activity starts. A logout button is also provided which the users may use to logout of the application.

If the book is not past its due date then on select it a new activity starts. This activity displays the basic information of the book and provides a button for scanning book ID. On clicking the button the scanner starts and now the corresponding book can be scanned and renewed. On scanning the ID of the correct book, its renewal date is updated in database. All of Android's software is written in Java, which is interpreted by the Dalvik virtual machine. Even the most core features such as the phone and the contacts application reside in this layer. This layer contains software written by the Android team as well as any third-party software that is installed on the device. An effect of allowing third-party developers access to this layer is that the user interface can be overhauled comparatively easily. Third party applications can handle any event that the Android team's application could see. This means that so long as there is a replacement application for the application, anyone could potentially write their own. Given this model we might expect that, as Android becomes more robust, the user will be able to specify what applications should handle which events.

Contrary, if the book is past its due date, then the user is directed to the activity that tells user to renew or return the book to library after paying the amount that has been incurred as fine.

4.5 GCM Module

4.5.1 Flow chart

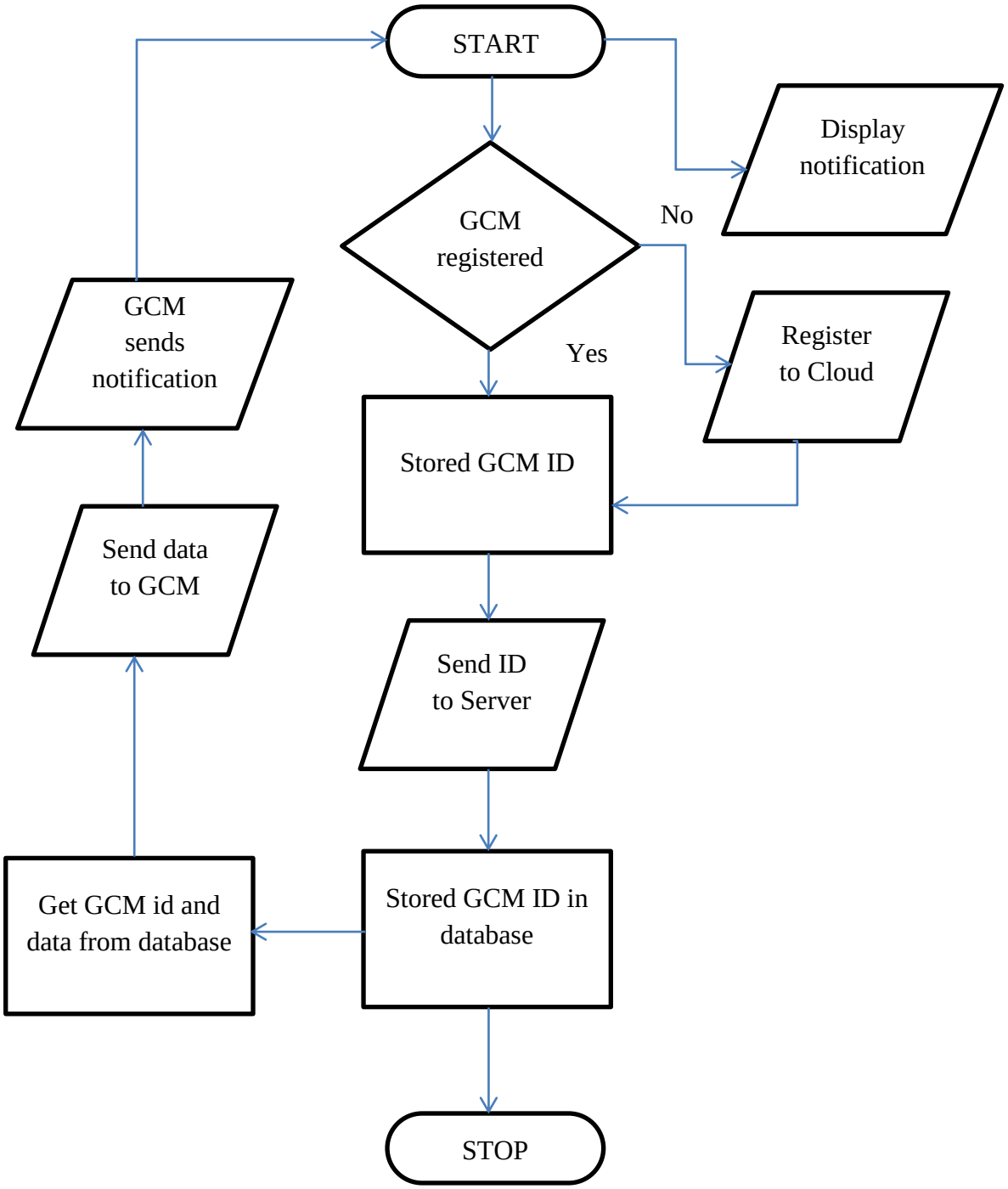

Fig 4.5 GCM Module Flowchart

4.5.2 Work flow Diagram

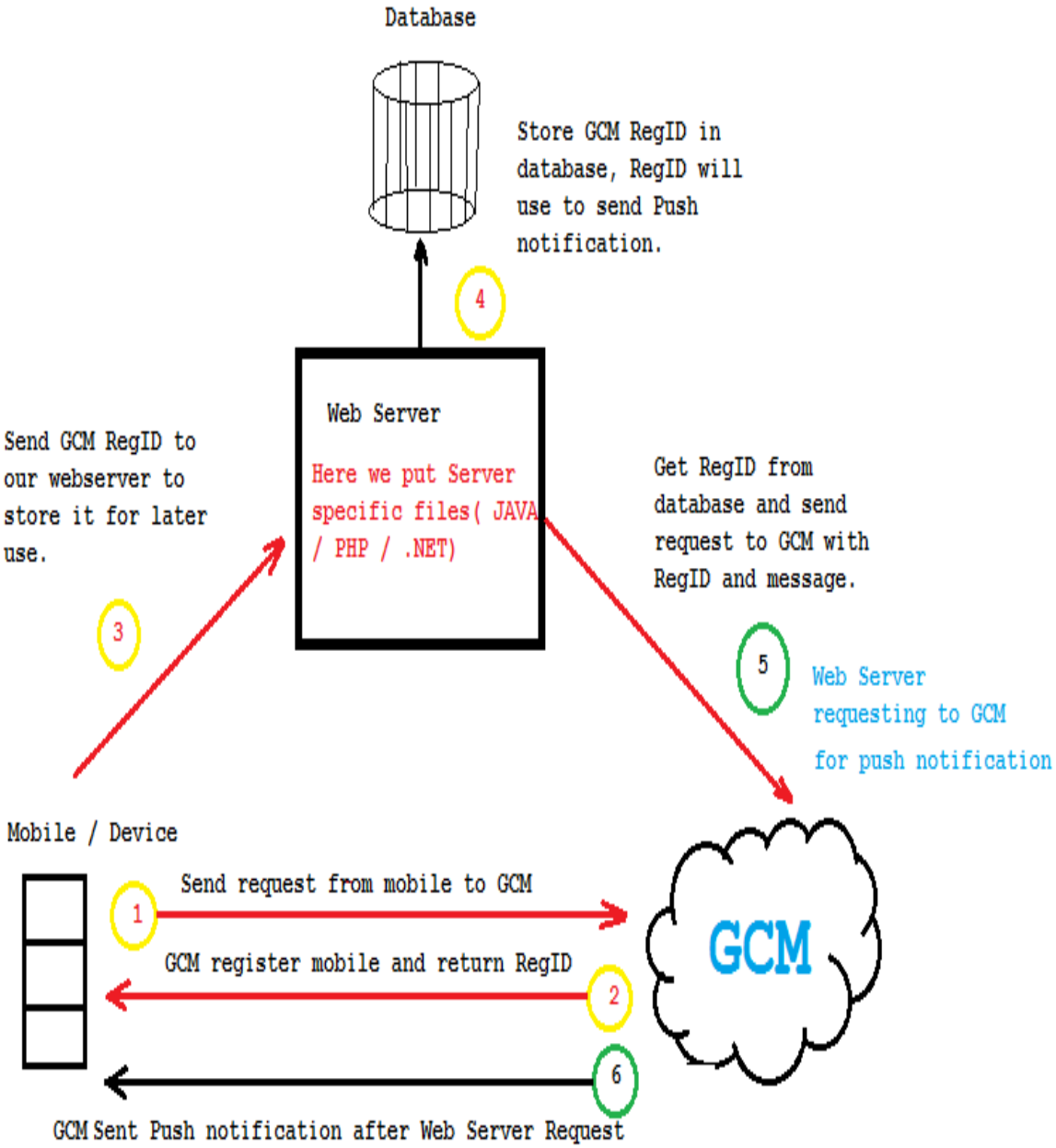

Fig 4.6 GCM Module Work flow Diagram

4.5.3 Description

The Google Cloud Messenger module interacts with the free cloud service provided by Google for generating and delivering notifications to registered android devices. The library server sends data to Google cloud by using the registration id of the devices.

As the android application starts in the device, it checks whether that device has been registered to cloud or not. If it is not registered, then it sends a request to the cloud. The Google cloud uses the sender ID to identify the project. Sender ID is obtained by creating a project in Google's API console website. Enabling the Google Cloud Messenger API for that project will allow our android application to receive notifications using that sender ID. Once the device is registered, the registration ID is sent to library server.

On the library server, a scheduler is used to run the PHP script that sends reminders to users. This script runs every 24 hours and fetches book names and their renewal dates which are due in two days' time. It then fetches the student registration ID which have issued those books and their corresponding GCM registration ID. Using those IDs, it sends the retrieved data to the cloud.

The Google cloud on receiving data form the library server generates notifications for that data and delivers it to the android devices with the corresponding GCM registration IDs. Once the notifications reach the intended device, a broadcast receiver is triggered which handles the notification and displays its content to the user. Using this way, users can get reminders about which books are to be renewed within the next couple of days.

4.6 Summary

The modules of android application were discussed in this chapter. Each module was thoroughly scrutinized and flowcharts as well as work flow diagrams were provided for the same. This was followed by in depth description of each module which discussed how data and control flow within each module had been. How these modules comprise of interactions between frontend, backend and the cloud was also explained in detail in the description.

CHAPTER 5

SYSTEM IMPLEMENTATION

5.1 Introduction

The system implementation deals with two issues namely android platform overview, and details of project implementation. It explains the nuances of android platform and also gives details about project implementation by providing sample codes.

The platform overview explains the features of android operating system. It deals with its importance, areas where it has been implemented its advantages. Android is the most widely used operating system in smartphones and hence it was chosen as the development platform. It is open source and offers numerous libraries that can be used for developing any kind of applications. Developed applications are available for download at the Google play store. Our project is free and hence users only need to allow certain permissions before downloading the application.

Project implementation details covers simulation parameters which have been used for test purposes, and the screen shots of each activity. These screen shots depict features of the application that have been discussed in the preceding chapters so far.

The following section gives details about android platform, and the section after that explains the implementation details by providing simulation parameters and activity screen shots.

5.2 Overview of Android Platform

Android is an operating system based on the Linux kernel, and designed primarily for touchscreen mobile devices such as smartphones and tablet computers. Initially developed by Android, Inc., which Google backed financially and later bought in 2005, Android was unveiled in 2007 along with the founding of the Open Handset Alliance—a consortium of hardware, software, and telecommunication companies devoted to advancing open standards for mobile devices. The first publicly available smartphone running Android, the HTC Dream, was released on October 22, 2008.

The user interface of Android is based on direct manipulation, using touch inputs that loosely correspond to real-world actions, like swiping, tapping, pinching, and reverse pinching to manipulate on-screen objects. Internal hardware—such as accelerometers, gyroscopes, and proximity sensors—is used by some applications to respond to additional user actions, for example adjusting the screen from portrait to landscape depending on how the device is oriented. Android allows users to customize their home screens with shortcuts to applications and widgets, which allow users to display live content, such as emails and weather information, directly on the home screen. Applications can further send notifications to the user to inform them of relevant information, such as new emails and text message

Android's source code is released by Google under the Apache License 2.0, which allows everyone to freely modify and distribute Android under various names except by reusing the "Android" trademark; however, device manufacturers and wireless carriers have licensed the trademark from Google. Most Android devices ship with a combination of open source and proprietary software. Android is popular with technology companies which require a ready-made, low-cost and customizable operating system for high-tech devices. Despite being primarily designed for phones and tablets, it also has been used in televisions, games consoles, digital cameras, and other electronics.

5.2.1 Interface

Android's user interface is based on direct manipulation, using touch inputs that loosely correspond to real-world actions, like swiping, tapping, pinching, and reverse pinching to manipulate on-screen objects. The response to user input is designed to be immediate and provides a fluid touch interface, often using the vibration capabilities of the device to provide haptic feedback to the user. Internal hardware such as accelerometers, gyroscopes and proximity sensors are used by some applications to respond to additional user actions, for example adjusting the screen from portrait to landscape depending on how the device is oriented, or allowing the user to steer a vehicle in a racing game by rotating the device, simulating control of a steering wheel.

Android devices boot to the home screen, the primary navigation and information point on the device, which is similar to the desktop found on PCs. Android home screens are typically made up of app icons and widgets; app icons launch the associated app, whereas widgets display live, auto-updating content such as the weather forecast, the user's email inbox, or a news ticker directly on the home screen. A home screen may be made up of several pages that the user can swipe back and forth between, though Android's home screen interface is heavily customisable, allowing the user to adjust the look and feel of the device to their tastes. Third-party apps available on Google Play and other app stores can extensively re-theme the home screen, and even mimic the look of other operating systems, such as Windows Phone. Most manufacturers, and some wireless carriers, customise the look and feel of their Android devices to differentiate themselves from their competitors.

5.2.2 Applications

Android has a growing selection of third party applications, which can be acquired by users either through an app store such as Google Play or the Amazon App store, or by downloading and installing the application's APK file from a third-party site. Google Play Store allows users to browse, download and update applications published by Google and third-party developers, and the Play Store client application is pre-installed on devices that comply with Google's compatibility requirements and license the Google Mobile Services software. The client application filters the list of available applications down to those compatible with the user's device, and developers may restrict their applications to particular carriers or countries for

business reasons. Purchases of unwanted applications can be refunded within 15 minutes of the time of download, and some carriers offer direct carrier billing for Google Play application purchases, where the cost of the application is added to the user's monthly bill.

Applications ("apps"), that extend the functionality of devices, are developed primarily in the Java programming language using the Android software development kit (SDK). The SDK includes a comprehensive set of development tools, including a debugger, software libraries, a handset emulator based on QEMU, documentation, sample code, and tutorials. The officially supported integrated development environment (IDE) is Eclipse using the Android Development Tools (ADT) plugin. Other development tools are available, including a Native Development Kit for applications or extensions in C or C++, Google App Inventor, a visual environment for novice programmers, and various cross platform mobile web applications frameworks.

5.2.3 Memory management

Since Android devices are usually battery-powered, Android is designed to manage memory (RAM) to keep power consumption at a minimum, in contrast to desktop operating systems which generally assume they are connected to unlimited mains electricity. When an Android app is no longer in use, the system will automatically suspend it in memory – while the app is still technically "open", suspended apps consume no resources (for example, battery power or processing power) and sit idly in the background until needed again. This has the dual benefit of increasing the general responsiveness of Android devices, since applications do not need to be closed and reopened from scratch each time, and also ensuring that background applications do not consume power needlessly.

Android manages the apps stored in memory automatically: when memory is low, the system will begin killing apps and processes that have been inactive for a while, in reverse order since they were last used. This process is designed to be invisible to the user, such that users do not need to manage memory or the killing of apps themselves. However, confusion over Android memory management has resulted in third-party task killers becoming popular on Google Play store; these third-party task killers are generally regarded as doing more harm than good.

5.3 Implementation Details

5.3.1 Simulation Parameters

The simulation parameters which were used for testing the project's functionality are as follows based on the activities and modules they were used in.

5.3.1.1 Login parameters

- Registration number : 1031020018, 1031020022
- Password : ppol, jotish
- Scanned ID : 8901425011676

5.3.1.2 Registration parameters

- Registration number : 1031020018, 1031020022
- Password : SRM University student portal password
- Email ID : x@y.com
- Phone Number : 7866543289

5.3.1.3 Backend database parameters

- Books: 101 - database book, 102 – coding book, 103 - AI book, 104 – book4, 105- data structure book, 8902519001993 - Book by Ninad

- GCM id - APA91bEwCwc2ZNFEzpaQtVZ7Z8z9PLNGdwVb0yJc6RG3VMbxWBY_QM55sjDFyiA_In7RgsTiPgs9SFurcw8J6DPQMonD7g_ZFC_stgW2wldOCykZtZTPJz29czcY8Nsm8JhD4VhnIo8jTu0PU7rz1jjcZrYZNdfTw

5.3.2 Sample coding

5.3.2.1 Login Activity:

```
package com.example.libraryisbn;

import java.io.IOException;

import java.io.InputStream;

import java.io.UnsupportedEncodingException;

import java.util.ArrayList;

import java.util.List;

import org.apache.http.client.utils.URLEncodedUtils;

import org.apache.http.impl.client.DefaultHttpClient;

import org.apache.http.message.BasicNameValuePair;

import org.json.JSONException;

import org.json.JSONObject;

import android.app.Activity;

import android.content.Context;

public class AppLoginOrRegister extends Activity {

        private EditText regnum;

        private EditText appPass;

        private Button scan;

        private static final String STATUS = "status";

        private static final String MSG = "msg";

        JSONParser jsonParser = new JSONParser();
```

```
public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.curl_login);
         TextView Register=(TextView)findViewById(R.id.textView2);
        TextView Login=(TextView)findViewById(R.id.textView1);
        regnum=(EditText)findViewById(R.id.editText1);
        appPass=(EditText)findViewById(R.id.editText2);
        scan = (Button) findViewById(R.id.button1);
        session = new SessionManagement(getApplicationContext());
        pLoad = (RelativeLayout) findViewById(R.id.LoadingProgress);
        mainDisp = (RelativeLayout) findViewById(R.id.MainScreen)
        pLoad.setVisibility(View.GONE);
         registerViews();
        status1=0;
        //Register a new user.
 Register.setOnClickListener(new OnClickListener() {
        public void onClick(View v) {
        Intent i = new Intent(AppLoginOrRegister.this, CurlLogin.class);
        startActivity(i);          }});
 //Login for existing user using id and password
Login.setOnClickListener(new OnClickListener() {
        public void onClick(View v) {
```

```
        if(checkValidation()){
        new ChkStuLogin().execute();}
        else{
Toast.makeText(AppLoginOrRegister.this,"Fill all Fields",Toast.LENGTH_SHORT).show();
}}});
//Login or register by scanning id
  scan.setOnClickListener(new OnClickListener(){
@Override
public void onClick(View arg0) {
    Intent intent = new Intent("com.google.zxing.client.android.SCAN");
 intent.putExtra("com.google.zxing.client.android.SCAN.SCAN_MODE",
"QR_CODE_MODE");
startActivityForResult(intent, 0);}});}
        public void onActivityResult(int requestCode, int resultCode, Intent intent) {
            if (requestCode == 0) {
                if (resultCode == RESULT_OK) {
                    contents = intent.getStringExtra("SCAN_RESULT");
                    new ChkStuLoginScan().execute();
                }
            else if (resultCode == RESULT_CANCELED) {
Toast.makeText(AppLoginOrRegister.this, "Scan it again Please!!!",
Toast.LENGTH_SHORT).show();            }}}
//Check student Id and password from server
class ChkStuLogin extends AsyncTask<String, String, String> {
```

```
        protected void onPreExecute() {
        super.onPreExecute();
Toast.makeText(AppLoginOrRegister.this,"Logging in...",Toast.LENGTH_SHORT).show();
        mainDisp.setVisibility(View.GONE);
        pLoad.setVisibility(View.VISIBLE); }
protected String doInBackground(String... args) {
        String mode_chkstulogin = "chkStudentLogin";
        String regno = regnum.getText().toString();
        String pass = appPass.getText().toString();
        // Building Parameters
        List<NameValuePair> paramsLogin= new ArrayList<NameValuePair>();
        paramsLogin.add(new BasicNameValuePair("mode", mode_chkstulogin));
        paramsLogin.add(new BasicNameValuePair("regid", regno));
        paramsLogin.add(new BasicNameValuePair("pwd", pass ));
// getting JSON string from URL
JSONObject jsonLogin = jsonParser.makeHttpRequest(ConnectionDetector.URL, "GET",
paramsLogin);
try {
status1 = jsonLogin.getInt(STATUS);
msg = jsonLogin.getString(MSG);
Log.d("Status1" , ">" + status1 );
} catch (JSONException e) {
e.printStackTrace();}
```

```
if(status1==1){
 // this code will send registration id of a device to our own server.
String urlgcm = ConnectionDetector.URL;
String mode_gcmid = "GCM_ID";
List<NameValuePair> paramsGCM = new ArrayList<NameValuePair>();
paramsGCM.add(new BasicNameValuePair("gcmid", StartSplashScreen.GCMregid));
paramsGCM.add(new BasicNameValuePair("regid", regno));
                try{
        DefaultHttpClient httpClient = new DefaultHttpClient();
        String paramString = URLEncodedUtils.format(paramsGCM, "utf-8");
        urlgcm += "?" + paramString;
        HttpGet httpGet = new HttpGet(urlgcm);
         HttpResponse httpResponse = httpClient.execute(httpGet);
        HttpEntity httpEntity = httpResponse.getEntity();
        is = httpEntity.getContent();              }
        catch (UnsupportedEncodingException e) {
                        e.printStackTrace();
                } catch (ClientProtocolException e) {
                   e.printStackTrace();
                } catch (IOException e) {
                   e.printStackTrace();} }
                      return null;}
protected void onPostExecute(String file_url) {
```

```
Toast.makeText(AppLoginOrRegister.this, msg, Toast.LENGTH_SHORT).show();
if(status1==1){
Regno = regnum.getText().toString();
session.createLoginSession(Regno);
Intent getBooks= new Intent(getApplicationContext(), Load_Renew_Books.class);
getBooks.putExtra("yourid",Regno);
startActivity(getBooks);
finish();}
else{
pLoad.setVisibility(View.GONE);
mainDisp.setVisibility(View.VISIBLE); } }}
class ChkStuLoginScan extends AsyncTask<String, String, String> {
protected void onPreExecute() {
super.onPreExecute();
Toast.makeText(AppLoginOrRegister.this,"Checking...", Toast.LENGTH_SHORT).show();
mainDisp.setVisibility(View.GONE);
pLoad.setVisibility(View.VISIBLE); }

protected String doInBackground(String... args) {
        String mode_chkstulogin = "chkStudentLoginScan";
        String regno = contents;
        // Building Parameters
        List<NameValuePair> paramsLoginScan= new ArrayList<NameValuePair>();
```

```
        paramsLoginScan.add(new BasicNameValuePair("mode", mode_chkstulogin));
        paramsLoginScan.add(new BasicNameValuePair("regid", regno));
        // getting JSON string from URL
JSONObject jsonLogin = jsonParser.makeHttpRequest(ConnectionDetector.URL, "GET",
paramsLoginScan);
try {
status1 = jsonLogin.getInt(STATUS);
msg = jsonLogin.getString(MSG);
} catch (JSONException e) {
e.printStackTrace();}
return null;}
 protected void onPostExecute(String file_url) {
          Toast.makeText(AppLoginOrRegister.this, msg, Toast.LENGTH_SHORT).show();
        pLoad.setVisibility(View.GONE);
 if(status1==1){
        session.createLoginSession(contents);
        Intent getBooks= new Intent(getApplicationContext(), Load_Renew_Books.class);
        getBooks.putExtra("yourid",contents);
        startActivity(getBooks);
finish();} else{
        Intent scanreg= new Intent(getApplicationContext(), ScanRegister.class);
        scanreg.putExtra("yourid",contents);
        startActivity(scanreg);
```

```
            finish();            }         } }
private void registerViews() {
        // TextWatcher would let us check validation error on the fly
            //Check registration number
    regnum.addTextChangedListener(new TextWatcher() {
             public void afterTextChanged(Editable s) {
            Validation.isRegistrationNumber(regnum, true);    }
            public void beforeTextChanged(CharSequence s, int start, int count, int after){}
            public void onTextChanged(CharSequence s, int start, int before, int count){}  });}
    private boolean checkValidation() {
            boolean ret = true;
            if (!Validation.isRegistrationNumber(regnum,true) &&
            !Validation.hasText(appPass)) ret = false;
            return ret;    }}
```

5.3.2.2 Login 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="match_parent"
    android:background="@color/background_color"
    android:orientation="vertical" >

<RelativeLayout
    android:id="@+id/MainScreen"
    android:layout_width="match_parent"
```

```
android:layout_height="match_parent"
android:background="@color/background_color"
android:orientation="vertical" >

<Button
   android:id="@+id/button1"
   style="@style/ButtonText"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_below="@+id/textView3"
   android:layout_centerHorizontal="true"
   android:background="@drawable/purple_btn"
   android:text="@string/SCANID" />

<EditText
   android:id="@+id/editText2"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_below="@+id/editText1"
   android:layout_centerHorizontal="true"
   android:layout_marginTop="31dp"
   android:ems="10"
   android:hint="@string/pass"
   android:inputType="textPassword" />

<EditText
   android:id="@+id/editText1"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_alignLeft="@+id/editText2"
   android:layout_alignParentTop="true"
   android:layout_alignRight="@+id/editText2"
   android:layout_marginTop="98dp"
   android:ems="10"
```

```
        android:hint="@string/Regno"
        android:inputType="number" />

    <TextView
        android:id="@+id/textView1"
        style="@style/ButtonText3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/editText2"
        android:layout_marginLeft="34dp"
        android:layout_marginTop="50dp"
        android:clickable="true"
        android:text="@string/Login"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:textColor="@color/text_color" />

</RelativeLayout>
<RelativeLayout
    android:id="@+id/LoadingProgress"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/background_color"
    android:orientation="vertical" >

    <ProgressBar
        android:id="@+id/load"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:indeterminateDrawable="@drawable/loader" />
    </RelativeLayout>
</RelativeLayout>
```

5.3.2.3 Backend PHP script

```
<?php
require 'config.php';
$result=array('status'=> 0,'msg'=>'Error in executing query!');

if(!isset($_REQUEST['mode']))
$mode='FAILURE';
else
{
 $mode=$_REQUEST['mode'];
switch ($mode)
                {
        case 'GET_RENEW_BOOKS':
                {
                  if(!isset($_REQUEST['regid']))
                 break;
                  else
                  {
                $regid=$_REQUEST['regid'];
        $q="SELECT ID, BOOKNAME, TIMESTAMP, RENEWDATE, AUTHOR FROM
renewal where regno = '".$regid."'";
                $stid = executeQuery($q);
                $msg = array();
                $i=0;
                while($row=fetchArray($stid))
                {$msg[$i++]=$row;}
                $result=array('status'=> 1,'msg'=>$msg);
                freeDB($stid);
                }}
                        break;}
echo json_encode($result);

?>
```

5.3.3 Screen Shots with

5.3.3.1 Splash Screen

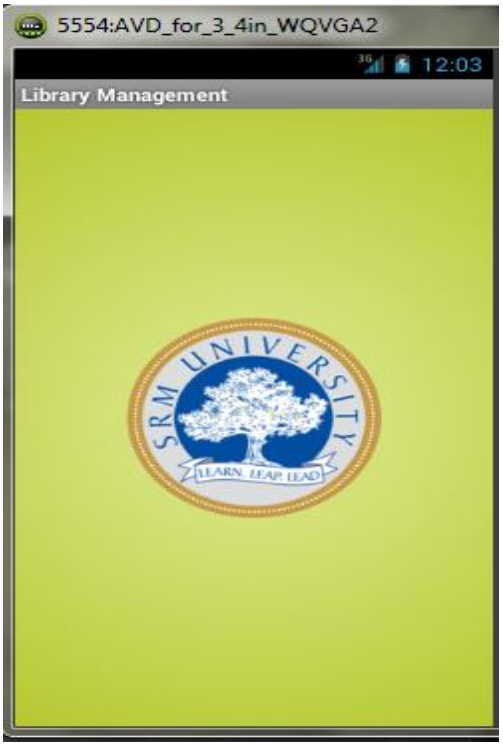

Fig 5.1 Splash Screen

5.3.3.2 Login Activity

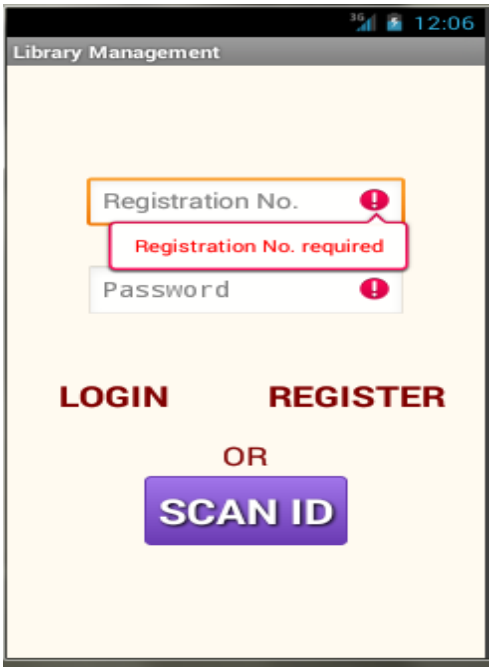

Fig 5.2 Login activity

5.3.3.3 Registration Activity

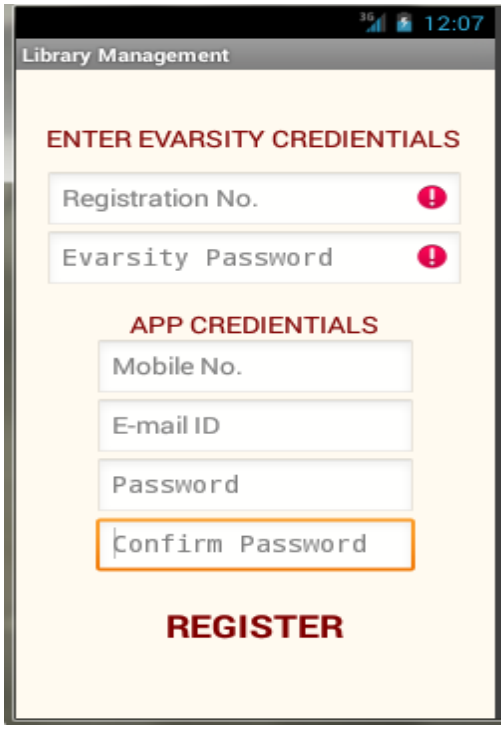

Fig 5.3 Registration activity

5.3.3.4 Registration using scanner

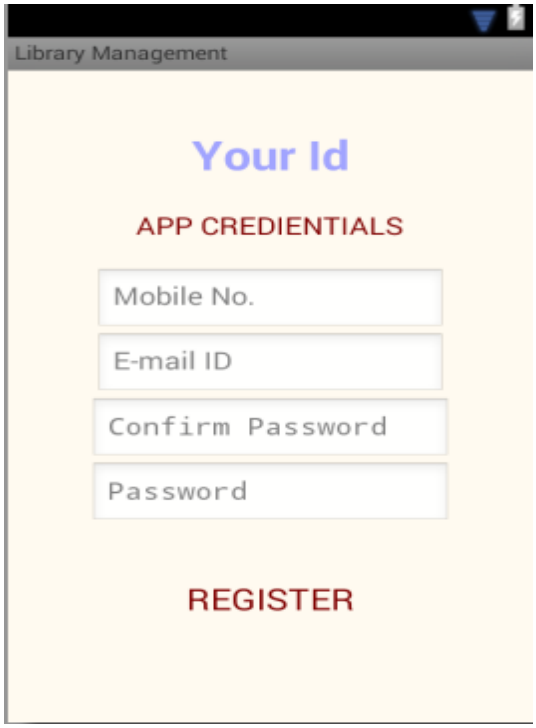

Fig 5.4 Scan register

5.3.3.5 Loading screen

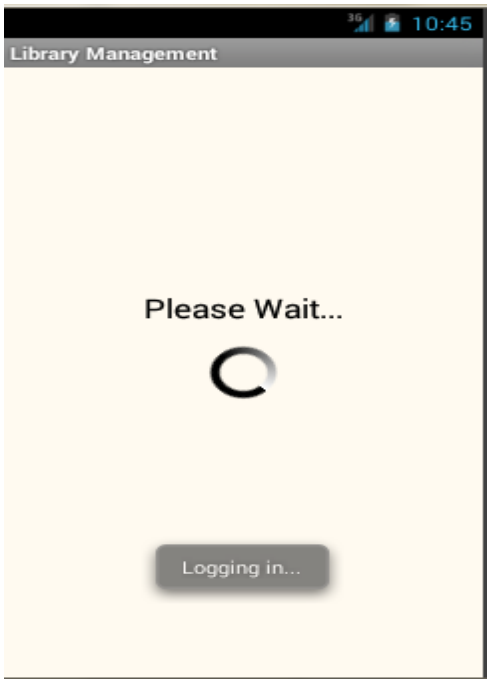

Fig 5.5 Loading screen

5.3.3.6 Booklist Activity

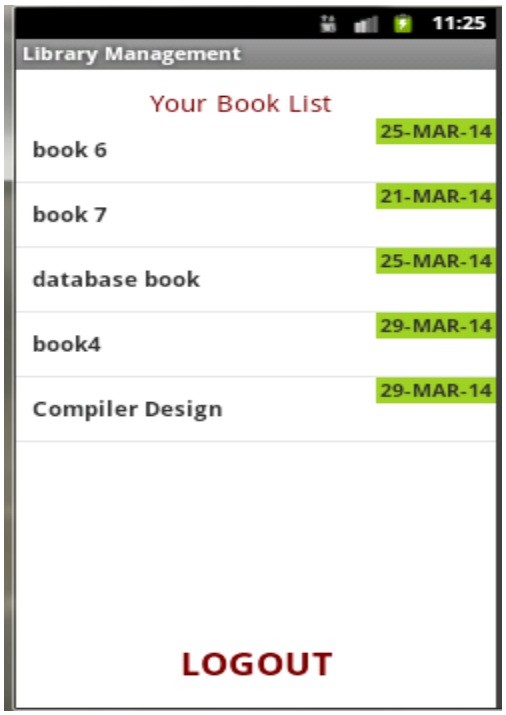

Fig 5.6 Booklist activity

5.3.3.7 Fine Incurred Activity

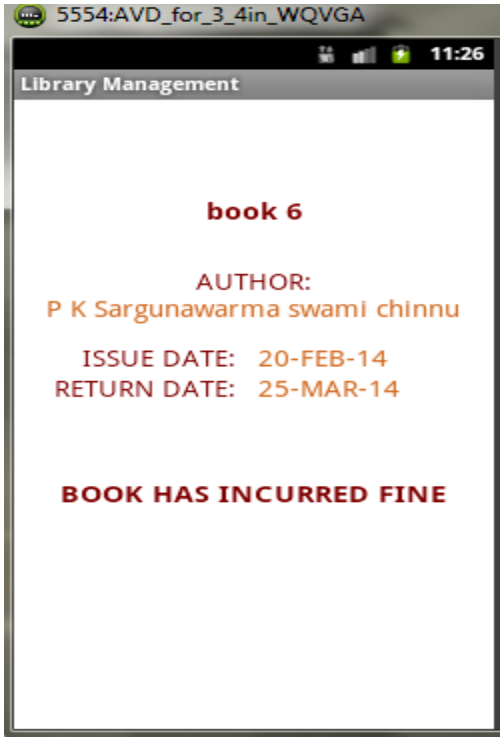

Fig 5.7 Fine incurred activity

5.3.3.8 Book Renewal Activity

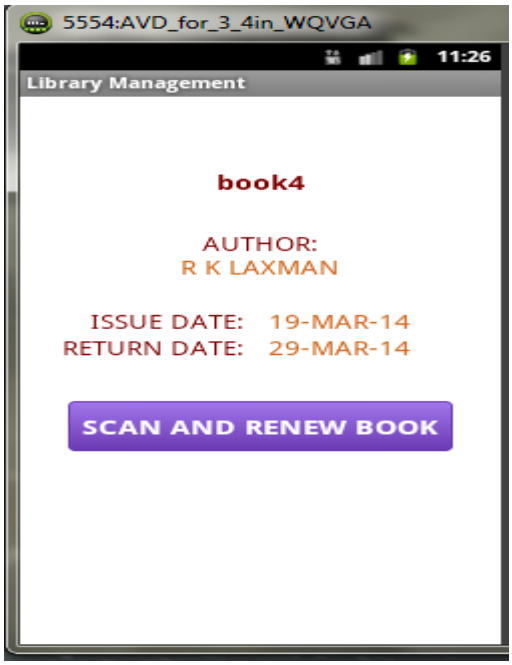

Fig 5.8 Book renewal

5.3.3.9 Receiving Notifications

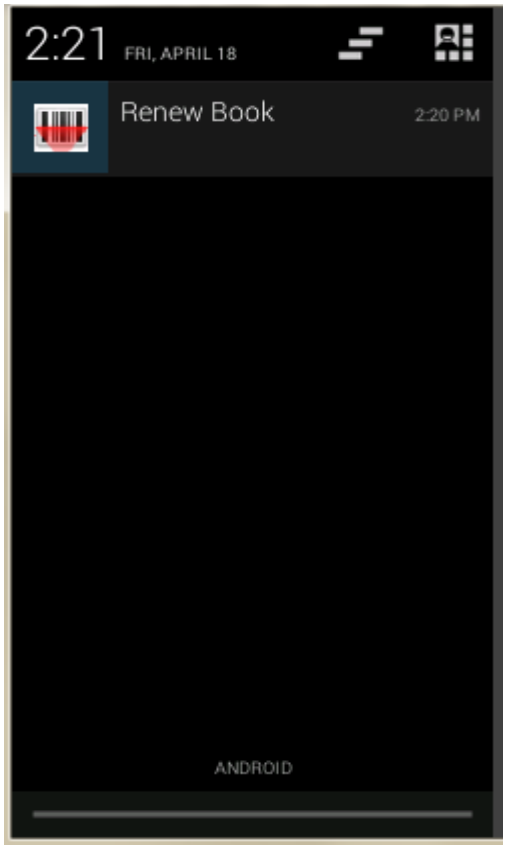

Fig 5.9 Notifications

5.4 Summary

This chapter gave a brief introduction about android platform and its features. It also successfully gave an idea about the working of the application by use of sample codes for activities, xml layout and backend PHP script. The screen shots were instrumental in providing a visual format of the application.

CHAPTER 6

CONCLUSION AND FUTURE WORK

Thus, the documentation provides a clear and complete idea about how the android application project was completed. SRM University students can conveniently renew their books without much hassle. The library administrators will also be benefited, as long queues of students wanting to renew their books, will become scarce. The application will provide advantageous to college as it will encourage similar kind of initiatives in near future.

This developed app can be upgraded further by introducing more features. Facility can be added for searching available books. This can be done by adding a search button that starts a new activity. The activity has a textbox for taking the book name, author name or book code as input. Books could also be scanned in this activity. On searching for the book, the related information can be displayed. Another feature that can be integrated is that students can access other user's profile to view the list of books issued by them. Students can recommend good books to other users. A student can 'follow' another students profile to get information about the books they issue. A book recommender can be added as well. It will recommend the list for books for each subject based the year in which the student is currently in. All in all, this is much scope for upgrading the current developed application.

REFERENCES

[1] http://developer.android.com/index.html

[2] http://www.androidhive.info

[3] http://en.wikipedia.org/wiki/Android_(operating_system)

[4] http://www.php.net/manual/en/book.curl.php

[5] http://curl.haxx.se/docs/manpage.html

[6] http://en.wikipedia.org/wiki/CURL

[7] http://www.oracle.com/technetwork/indexes/downloads/index.html#database

[8] http://www.php.net/manual/en/curl.examples-basic.php

[9] http://www.lornajane.net/posts/2011/posting-json-data-with-php-curl

[10] http://srmapi-hmm.rhcloud.com/magic.php

[11] http://techlovejump.in/2013/11/android-push-notification-using-google-cloud-messaging-gcm-php-google-play-service-library/

[12] http://www.vogella.com/tutorials/AndroidJSON/article.html

[13] https://code.google.com/p/zxing/downloads/list

[14] http://tausiq.wordpress.com/2013/01/19/android-input-field-validation/

[15] http://stackoverflow.com/

[16] http://www.wampserver.com/en/

[17] Using JSON for Data Exchanging in Web Service Applications Dunlu PENG, Lidong CAO, Wenjie XU, School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China

[18] On Using JSON to Create Evolvable RESTful Services ,Markus Lanthaler ,Institute for Information Systems and Computer Media Graz University of Technology, Graz, Austria, Christian Gütl, School of Information Systems, Curtin University of Technology .Perth, Australia