**DESIGN AND IMPLEMENTATION OF A MOBILE-BASED PERSONAL DIGITAL ASSISTANT FOR STUDENTS**

**BY**

**OGBUNUDE**,CHIBUEZE CHARLES

(13CG015945)

**A PROJECT SUBMITED TO THE DEPARTMENT OF COMPUTER AND INFORMATION SCIENCES, COLLEGE OF SCIENCE AND TECHNOLOGY, COVENANT UNIVERSITY OTA, OGUN STATE.**

**IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE BACHELOR OF SCIENCE (HONOURS) DEGREE IN COMPUTER SCIENCE**

**AUGUST, 2020**

**CERTIFICATION**

This is to certify that the project work titled **“DESIGN AND IMPLEMENTATION OF A MOBILE-BASED PERSONAL DIGITAL ASSISTANT FOR STUDENTS”** is a bona fide work carried out by **Ogbunude Chibueze Charles** (13CG015945) and was supervised by me and submitted to the Department of Computer and Information Sciences, College of Science and Technology, Covenant University, Ota.

1. Name: Dr. Oluranti Jonathan

(**Supervisor)**

Signature ­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date ­­­­­­­­­­­­­­­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. Name: Prof. Ambrose A. Azeta

(**Head of Department)**

Signature \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date ­­­­­­­­­­­­­­­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**DEDICATION**

I dedicate this project to God who has and is my strength, who enables me to deliver more that I am capable of and conquer what is too strong for me.

**ACKNOWLEDGEMENTS**

My sincere appreciation goes to God, for His strength and ever present help. For His sovereign and loving influence in situations that needed influence. He equipped me with the necessary skills and blessed the works of my hands.

My parents, Mr and Mrs. Ogbunude, for your full support and advice. Their prayers and loving words were not in vain. They motivated me to go the extra mile, to burn the extra candle and to sleep an hour less. Their impact in my life makes me speechless.

My supervisor, Dr. Oluranti Jonathan, an extension of my parents on campus. God uses people to influence my life and you are one of those people. When I was going off track you put me back on track, you gave me the support only the best supervisors give.

My colleagues, Ruky, Maureen, and Tega. As your faces are different, so also you are talented and influence in various ways. By our cooperation and support of each other, we developed a synergy, which resulted our greater productivity.

My friends, who had direct influence on this project, Uchenna, for your technical and emotional support, one who has a bizarre skill. Timothy, for your kindness and support, you are the man with the resources, from you I learnt resilience and gentleness.

**TABLE OF CONTENTS**

**Title** **Page**

|  |  |
| --- | --- |
| Certification | ii |
| Dedication | iii |
| Acknowledgement | iv |
| List of Tables | viii |
| List of Figures | ix |
| Abstract | x |

**CHAPTER ONE: INTRODUCTION**

|  |  |  |
| --- | --- | --- |
| 1.1 | Background Information | 1 |
| 1.2 | Statement Of the Problem | 2 |
| 1.3 | Aim and Objectives Of The Study | 2 |
| 1.4 | Research Methodology | 2 |
| 1.5 | Significance Of The Study | 3 |
| 1.6 | Limitations Of the Project | 3 |
| 1.7 | Project Outline | 4 |

**CHAPTER TWO: LITERATURE REVIEW**

|  |  |  |
| --- | --- | --- |
| 2.1 | Introduction | 5 |
| 2.2 | History of Personal Digital Assistants | 5 |
| 2.3 | Personal Digital Assistants Today | 5 |
| 2.4 | Review of Existing Systems | 8 |
| 2.4.1 My Effectiveness | | 8 |
| 2.4.2 Accomplish | | 9 |
| 2.4.3 My study Life | | 11 |
| 2.4.4 Roubit | | 12 |
| 2.5 | Features of Student Personal Digital Assistant | 13 |
| 2.5.1 Android Creative Vision | | 14 |
| 2.5.2 Ease of Use | | 14 |
| 2.5.3 Notification | | 14 |
| 2.5.4 Planning and Scheduling | | 14 |
| 2.5.5 Information | | 15 |
| 2.6 | Issues with Student Personal Digital Assistants | 15 |
| 2.6.1 Lack of Features | | 15 |
| 2.6.2 Privacy and Security | | 15 |
| 2.6.3 Learning Curve | | 15 |
| 2.6.4 Internet Access | | 16 |
| 2.7 | Terms Associated with Student PDAs | 16 |

**CHAPTER THREE: SYSTEM ANALYSIS AND DESIGN**

|  |  |  |  |
| --- | --- | --- | --- |
| 3.1 | Introduction | | 17 |
| 3.1.1 Requirement Analysis | 17 |
| 3.1.1.1 Functional Requirement | 17 |
| 3.1.1.2 Non-Functional Requirement | 18 |
| 3.2 | | System Architecture | 19 |
| 3.3 | | System Design | 20 |
| 3.3.1 Physical Design | | | 20 |
| 3.3.1.1 Input Design | | | 20 |
| 3.3.1.2 Output Design | | | 20 |
| 3.3.2 Logical Design | | | 20 |
| 3.3.2.1 Class Diagram | | | 21 |
| 3.3.2.2 Use Case Diagram | | | 22 |
| 3.3.2.3 Activity Diagram | | | 23 |
| 3.3.2.4 Sequence Diagram | | | 24 |
| 3.3.3 Conceptual Design | | | 28 |
| 3.3.3.1 Description of Tables | | | 29 |

**CHAPTER FOUR: SYSTEM IMPLEMENTATION**

|  |  |  |
| --- | --- | --- |
| 4.1 | Introduction | 34 |
| 4.2 | System Requirements | 34 |
| 4.3 | The Implementation Tools Used | 35 |
| 4.4 | Software Development Methodology | 35 |
| 4.5 | The Program Modules and Interfaces | 36 |
| 4.5.1 The Dashboard | | 37 |
| 4.5.2 The Navigation Drawer | | 38 |
| 4.5.3 Lecture Module | | 39 |
| 4.5.4 To-Do List Module | | 42 |
| 4.5.5 Assignment Module | | 46 |

**CHAPTER FIVE: SUMMARY, RECOMMENDATIONS AND CONCLUSION**

|  |  |  |
| --- | --- | --- |
| 5.1 | Summary | 50 |
| 5.2 | Recommendations | 50 |
| 5.3 | Conclusion | 51 |
| **REFERENCES** | | 52 |
| **APPENDIX** | | 53 |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **Table** | **Page** | |
| Table 3.1 | Use case narrative for the user | 23 |
| Table 3.2 | Task\_List table | 29 |
| Table 3.3 | Lecture table | 30 |
| Table 3.4 | Assignment table | 31 |
| Table 3.5 | Project table | 32 |
| Table 3.6 | Contact table | 33 |
| Table 4.1 | The Software Requirements | 34 |
| Table 4.2 | The Hardware Deployment Requirements | 34 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Figure** | **Page** | |
| Figure 2.1 | Image displaying Assistant mobile application | 6 |
| Figure 2.2 | Image displaying My Study Life in operation | 7 |
| Figure 2.3 | Image displaying My Effectiveness in operation | 8 |
| Figure 2.4 | Image displaying Accomplish in operation | 10 |
| Figure 2.5 | Image displaying My Study Life in operation | 11 |
| Figure 2.6 | Image displaying Roubit in operation | 13 |
| Figure 3.1 | Class diagram representing a mobile-based student PDA | 21 |
| Figure 3.2 | Use-case diagram for mobile-based student PDA | 22 |
| Figure 3.3 | Activity diagram for user saving a task | 24 |
| Figure 3.4 | Activity diagram for user deleting a task | 25 |
| Figure 3.5 | Activity diagram for adding a project | 26 |
| Figure 3.6 | Sequence diagram for user saving a task | 27 |
| Figure 3.7 | Sequence diagram for user deleting a task | 28 |
| Figure 4.1 | The agile development process | 36 |
| Figure 4.2 | The dashboard | 37 |
| Figure 4.3 | The navigation drawer | 38 |
| Figure 4.4 | Lectures view | 40 |
| Figure 4.5 | Add lecture activity | 41 |
| Figure 4.6 | Task view | 43 |
| Figure 4.7 | Notification sample | 44 |
| Figure 4.8 | Add task activity | 45 |
| Figure 4.9 | Assignment view | 46 |
| Figure 4.10 | Add assignment activity | 47 |
| Figure 4.11 | Project view activity | 48 |
| Figure 4.12 | Add project activity | 49 |

**ABSTRACT**

A few centuries ago, one might not have needed to have a To-Do list or a daily plan. Today a lot of things are fighting for our time. There is more to do today that was not possible a hundred years ago, but a day is still 24 hours, so there is a very high probability that because we have so much to do in little time we end up choosing the activities which do not make the best use of our time. Few decades ago, organizers started to become popular, this was vital and resulted in what we have today. In the 21st century, information is moving at the speed of light and students need more than a pen and paper to organize their day. In today’s world, where a phone is a must have and where the power in a mobile device is astonishing, there is no better way to reach students with a solution than through the mobile platform.

The aim of this project is to develop a personal digital assistant for students. The system will be mobile-based, specifically on the Android platform. The system adopts Google’s material design guidelines for the design of its user interface, Android Java was the main programming language for the system, SQLite was used for the database and the development environment used was Android studio.

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

**CHAPTER ONE**

**INTRODUCTION**

* + 1. **BACKGROUND INFORMATION**

In today’s busy world, everyone needs a personal assistant. The truth is that even though everyone needs a personal assistant not everyone can get one. A human who is a personal assistant is prone to make mistakes, get tired, forget important events and make decisions based on emotions. To prevent that mess, we use technology, this has resulted in what we know today as personal digital assistants. Students are the focus this time because these days they are tasked with more things than past generations were ever tasked with as students. Students have to keep track of assignments, deadlines, projects, personal pursuits and all these on the long run this can wear out students.

Decades ago, personal digital assistants were not available; unlike what is available today; all things had to be put down on paper. People still have diaries and books where they did their major planning, but with the advent of mobile computing, applications and artificial intelligence, those methods are being replaced.

As for students having a personal digital assistant will enable them have time for things which are more important, rather than spend their time, energy and resources on thing that can be handled effectively and even better by an application. With the growth in sales of mobile devices, there is a great opportunity to help students through the mobile platform.

According to Google, over 50 billion apps have been downloaded of Google play, that figure alone shows that a lot of people do download and use applications. The implication is that a lot of people can be influenced through mobile applications. Another staggering statement that Google made was that there are over 1-billon active Android users and 1.5 million devices are activated daily. It is obvious that if a digital personal assistant for students is to be deployed on the mobile platform a good amount of students will be able to download and benefit from it.

There are personal digital assistants already existing on Google play store, however the nature of the student personal digital assistant is uniquely geared towards students as the target users and will provide features unique to the user. The traditional method of using pen and paper to document tasks is still very good, but that cannot keep up with our world where information is traveling at the speed of light and things change at the speed of thought.

**1.2 STATEMENT OF THE PROBLEM**

The daily life of student is tasking, a lot of things to thinks about and remember. It is no stranger to hear some students actually have high blood pressure, at such young ages. Students who do not properly plan their day get involved in unproductive activities. It will be better if the pressure can be reduced by helping the students take their mind off things that are not critical, these things can be handled by digital assistants giving the student more time to relax and engage in more productive thinking instead of worrying and forgetting.

* + 1. **AIM AND OBJECTIVES OF STUDY**

The aim of this project is to design and implement a student personal digital assistant.

The objectives of this project include:

1. To carry out a study of existing student personal digital assistant.
2. Design and model a student personal digital assistant.
3. Implement the student personal digital assistant.
   1. **RESEARCH METHODOLOGY**

Research methodology goes further to explain the best method intended to achieve the stated objectives.

**Objective 1: Carry out study of existing student personal digital assistant**

In order to carry out a research on existing systems, a thorough study of such applications on Google play will be carried out, what they did, how they did it, their flaws and their merits.

**Objective 2: Design and model a student personal digital assistant**

To design and model the system, UML was used for the system modelling and design. Diagrams such as sequence, activity and use case was used to model the system.

**Objective 3: Implement the student personal digital assistant**

This will be achieved by the implementation of an Android application; the user interface design was guided by Google’s creative vision design principles. The system is intended to run on a device running an Android operating system. This mean it will not run on Apple devices or Windows phone. The development tool used was Android studio, which is the most popular integrated development environment for developing android applications, the former tool used was Eclipse but Android studio is better and more supported. The main programming language used was Java, XML (extensible markup language) was used for the design while SQLite was used for the database, SQLite is portable and fast.

* 1. **SIGNIFICANCE OF THE STUDY**

The importance of this project cannot be over-emphasized; little drops of water will fill a bucket over time so also doing little menial tasks repetitively can wear out a student. It is not that these tasks are irrelevant but these can be handled by a personal digital assistant. A student using a personal digital assistant is generally more productive than a student who is not, this implies that every single student should have a personal digital assistant.

* 1. **LIMITATION OF THE STUDY**
* At the first release, application may not have all the features desired by students.
* Due to time constraint, the first release may not have all intended functionalities, though future applications will have.
* The application can only work on devices running on an Android operating system, which means for students who only have Apple devices or windows phone may not enjoy the features of this application.
* This personal digital assistant is only for students and not designed for lecturers or people working in companies.
  1. **PROJECT OUTLINE**

Chapter One of the project contains the introduction, the statement of the problem, the aim and objectives of the study, the significance of the study, the methodology used and the structure of the study.

Chapter Two is a summary of literature and consists of researches, journals, articles attempts and projects related to student personal digital assistant.

Chapter Three presents the student personal digital assistant and its system design, which includes the physical and logic design of the physical and logical design; it presents the system architecture and conceptual design.

Chapter Four contains the system implementation, the tools used, the development methodology, program modules, interfaces and system development process.

Chapter Five contains the general overview of the project, findings from the existing works in relation to the new system. It also includes the result and significance of the work and finally suggestions to the research community or system users.

**CHAPTER TWO**

**LITERATURE REVIEW**

**2.1 INTRODUCTION**

An assistant is simply one who assists. To assist means to give support or help, to make it easier for someone to do something or for something to happen. For something to be personal, it means it belongs or relates to a particular person. In the modern day, one cannot always get a personal assistant. Students today have more things to learn and deal with than any past generation. Students are bombarded daily with distractions and alternatives, without proper planning and adequate help a student can stray away. Students need not bother about some tasks, which can be handled by a digital assistant. Remembering when to submit assignments or keeping track of a project can be done better with the help of a digital assistant. Those tasks may sound simple but they clog the mind and inhibit productive thinking. If a digital assistant can make a student more productive, the question of whether it should be used should not even be asked because the answer obvious.

**2.2 HISORY OF PERSONAL DIGITAL ASSISTANTS**

Before the term personal digital assistant came to be, there were personal planners, which were in paper format already existing. As people were more engaged and nations grew, things needed to be organised. Personal Digital Assistants (PDA) gradually became popular, but were not as powerful as what we have today. Actually, the term PDA referred to a device, which was sold to assist people and help them organize and plan their schedule. Today PDA’s are not just devices but can also be software, which run on different devices. Today the PDA’s are intelligent and can make use of the available hardware power in today’s computer (McCarthy, 2013).

**2.3 PERSONAL DIGITAL ASSISTANTS TODAY**

Today personal digital assistants go from the simple to the complex where artificial intelligence is involved. We all use personal assistant everywhere without knowing, and soon it will become a part of life such that everybody will have one in his or her home. Today, each new Microsoft Windows system comes with a popular system known as Cortana. Cortana is an intelligent personal assistant that uses voice commands to execute tasks on Windows operating system. There are also the less intelligent versions, which can be on a smart phone as an application. Assistant, which is also an intelligent personal assistant, is a mobile application unlike Cortana, which has a desktop version. It does tasks like telling you the nearest location to get a drink or buy some clothes (Assistant Android Application, 2017).



Figure 2.1 Image displaying ASSITANT in operation

(Source: techcrunch.com)

Above, Figure 2.1 is showing Assistant in operation, it is quite intelligent ad doing well as an assistant. It is so good that it gives information faster that a human personal digital assistant will give. There are other personal assistant such as *My Study Life*, which do not make use of artificial intelligence. Not every PDA needs artificial intelligence; in fact, some PDAs are better off without the complexities of implementing artificial intelligence.

The aim of all these is not just to make life more fun, but to assist in daily tasks giving us more time to pay attention to things that cannot be delegated to a computer. The importance of this for students cannot be overemphasized. There are hardworking students who have their days planned, weeks planned and even months planned but they use the old-fashioned ways of using books and documenting that is less productive. No doubt, using a pen and paper is good, but software goes the extra mile, there are somethings that just cannot be accomplished with paper.

A paper cannot alert you when to execute a task neither can you change your mind and cancel a task to be done without messing up the paper. In addition, papers and books wear out or get lost a lot more than digital devices.

Figure 2.2 shows the interface of My Study Life, which is quite different from the other application shown in Figure 2.1 though they all can be called digital assistants.

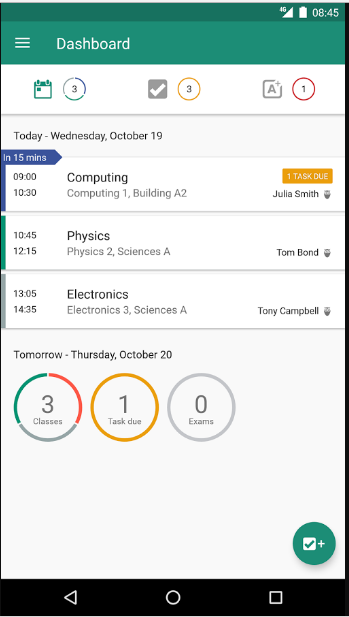


Figure 2.2 Image displaying My Study Life in operation

(Source: play.google.com)

**2.4 REVIEW OF EXISTING SYSTEMS**

There are existing personal digital assistants available on Google’s play store. Google’s play store is an online application store where developers can upload their applications and user can download any application of their choice. Some are free while others involve some kind of payment before or after download.

**2.4.1 My Effectiveness**

On Google’s play store this is one of the best personal digital assistants, the developer have carefully designed the application with excellent features. In a review, students were asked to choose between five alternatives, it turned out that if they choose My Effectiveness as one of the best applications. Figure 2.3 is displaying the user interface of My Effectiveness.

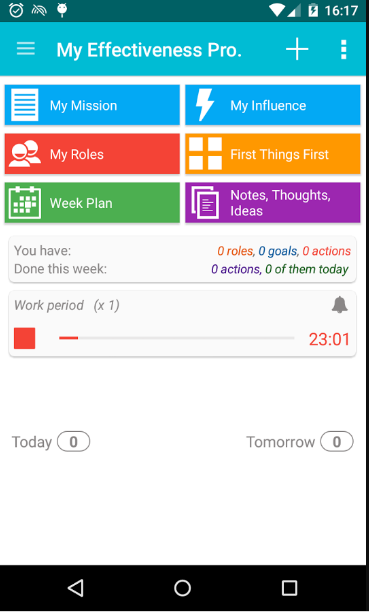


Figure 2.3 Image displaying My Effectiveness in operation

(Source: play.google.com)

Some features of the application they say help increase productivity are:

* You can define your principles which is your mission
* Document your influences and concerns
* Set your priorities, which is helpful when there is a choice between two alternatives
* Put down your life roles
* Identify and put down your goals
* Create a comprehensive Action plan
* You can put the tasks into Projects/Actions/Checklists
* You can use unique Weekly Plan view.

**2.4.2 Accomplish**

Accomplish is another great application on Google’s play store. You can easily recognize it with its authentic reddish logo. It helps you graphically plan your time, which is a lot easier and actually, more fun that using a pen and paper. It uses a simple and conventional to-do list. Figure 2.4 displays accomplish in operation.

Accomplish is made up of two parts:

1. The Accomplish To-do list

This is that part of the application where you save all your tasks you are to do. The list is placed in a list that has features of a navigation panel. The list slides out from the right side of the application.

1. The Accomplish Day-view

This is the wider view of all tasks in a calendar like form. Editing tasks here is actually easier than it is in most calendars. Seeing tasks in calendar view enables the user of the application see everything in a wider view, which is better for those who want to plan ahead but want to see their appointments to avoid any clahes.

Accomplish’s features:

* The day view spoken about synchronizes with Google calendar
* The user interface is smooth and fast with gesture based controls
* The colourful nature of the tasks and themes are beautiful
* It has powerful custom popup reminders
* It has a clean modern design

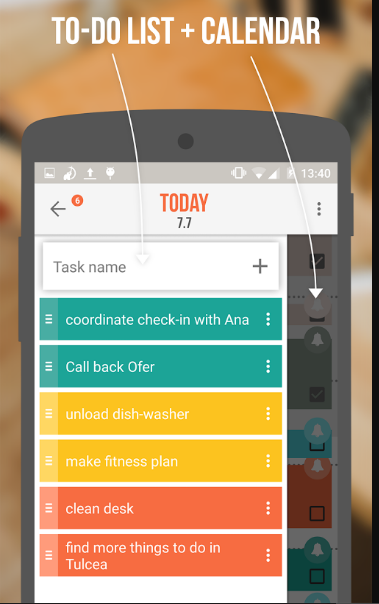


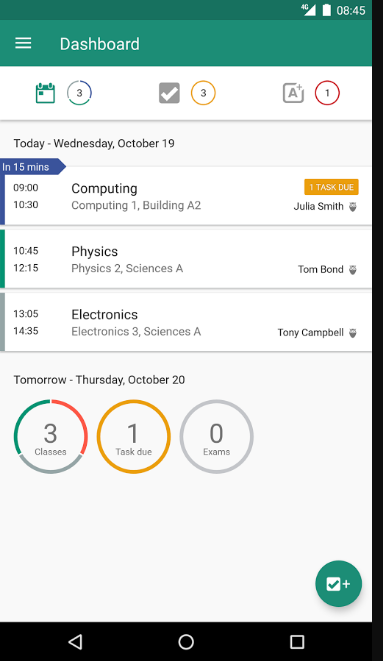
Figure 2.4 Image displaying Accomplish in operation

(Source: play.google.com)

**2.4.3 MSL**

MSL or My study Life is another successful digital assistant play store with success in the likes of My Effectiveness. My study life has 1,000,000 to 5,000,000 downloads on play store with over 23,000 5-star ratings.

One great feature about MSL application is the face that it was designed for students, teachers and lecturers. It is to make ones study life easier to manage. It has functionalities to store assignments and exams, which use the cloud to make it available on any device. It even has features to show home works due for submission, classes that conflict with exams and even that ability to add a revision task for a specific exam. Figure 2.5 shows My Study Life in operation.



**Figure 2.5 Image displaying My Study Life in operation**

(Source: play.google.com)

Some features of My Study Life include:

**Track** – Have real time information about your homework, assignment and your revision.

**Store your exams** – Instead of writing down exam times and pasting on the wardrobe, one can keep all exams alongside classes and get notified if there are clashes.

**Manage classes** – Manage classes, view timetables and revisions.

**Get notified** – Get reminded of due assignments, clashing classes and much more, this one feature that the traditional method of putting information on paper cannot accomplish.  
**Share time tables**- My Study Life actually allows teachers to create their timetables and if they wish, share it with students in the same school.

**2.4.4 Roubit**

Like the others, Roubit is another good digital assistant, though not with as many features as the others offer. It is an application that specializes in routine work that someone does every day. In a review Roubit was also one of the most loved application when compare with alternatives it has 100,000 to 500,000 downloads on play store and 922 5-star ratings.

Roubit has a simple but well-designed interface. Roubit has an authentic blue colour, which is the default theme of the application. It has days of the week from Monday to Sunday and you can add tasks on those days using the floating button. The floating button is the button at the right bottom of the screen with a plus sign on it. It is also equipped with reminders so that you get notified when the time for a task has reached. The application makes use of material design animation such as circular reveal animation and custom check boxes.

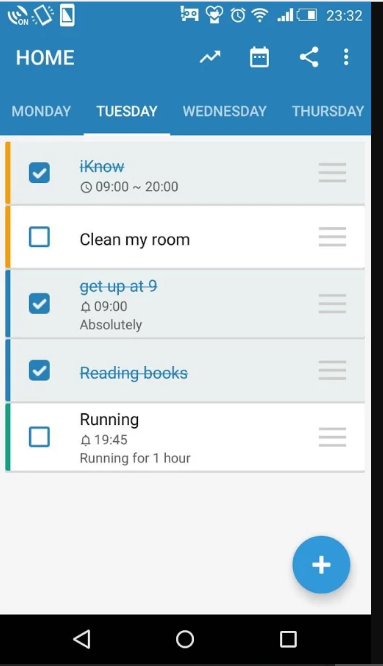


Figure 2.6 Image displaying Roubit in operation

(Source: play.google.com)

**2.5 FEATURES OF A STUDENT PERSONAL DIGITAL ASSISTANT**

There are some common features of mobile-based student personal digital assistants, these features should be present in all others, and they are discussed below. These features may not be available in every single student PDA but they are feature of a modern standard PDA for students.

**2.5.1 Android Creative Vision**

The Android creative vision and design principles were created to keep user’s best interest in mind. Most modern applications are designed using android creative vision. Google even says they will soon only allow applications that use material design to feature on Play Store. Apps that work in expected ways are instantly familiar to Android users, gain their trust, and ensure they engage with the app’s consent, functionality, and features.

**2.5.2 Ease of Use**

The users of these applications are students so therefore, the user interface should not be complex and as much as possible the learning curve should not be steep. Addition of task and deletion should not take time or involve a long process and the app should contain short cuts of operations to enable students easily accomplish what they want with a few clicks.

**2.5.3 Notification**

With the old method of using papers, there was no notification, but with digital power, there is a lot that can be done, one of which is notification. A notification can be a simple pop up and beep that notifies the user of when a task is due, this helps the user remember because he is prone to even forgetting what he has planned. A notification runs on a service that is available even when the student is not using the application.

**2.5.4 Planning and Scheduling**

With digital assistants, students can make plans and not just for the day but for future days, they can look at their schedule for the day and make changes where necessary. With the old method of documenting on paper, planning was possible but not as convenient as planning with a digital assistant because you can edit with a digital assistant and delete peradventure the user’s schedule changes. When using digital assistants you can see a whole month at a glance an see what and when you have appointments. Some PDAs even have the functionality of informing you when you have clashing tasks, which is priceless information.

**2.5.5 Information**

If a student was to plan his year with the traditional method of writing it down, he may not be able to give account how many goals he has per month, how many are due, how many he has accomplished. He may have little information about all this, but why go through the stress when a digital assistant can give real time information about all things that pertains the students life, his assignments, projects and to-do list. A PDA can give the student a summarized view of all he has documented down so that he can use that information to plan further.

**2.6 ISSUES WITH STUDENT PERSONAL DIGITAL ASSISTANTS**

As with most technologies there are some shortcomings, student personal digital assistants are no exception.

**2.6.1 Lack of Features**

A common reason for low star rating of applications on play store is lack of features, the issue is that an application may not be able to have all the feature desired by a particular student. A good application should be one that each student feels the application was specially designed for him or her. Some students want all in one package, while some other students love simplicity; they want the application to include just what the need and not every possible functionality. This does not make it easy for developers because they have to include the right amount to functionality to satisfy the majority of users.

**2.6.2 Privacy and Security**

Personal information is stored such as personal tasks or goals. Unauthorized persons could have access to those personal information and cause havoc. Personal things should remain personal and when designing applications that keep personal information, they should be designed to be reliable and dependable.

**2.6.3 Learning Curve**

The learning curve of some applications are actually steep though they may be great applications, and example is My Effectiveness. Learning curve indicates how hard it is to learn a software or in other words how fast a user can understand and get used to a software. A steep learning curve is one that is not so easy to understand and can get complex quite quickly. My Effectiveness is a great application, one of the great names but it is complex, it has everything and some students may not like the complexity, but there are simple apps such as Roubit or the most successful of the listed, which is My Study Life, has a very simple self-explanatory interface.

**2.6.4 Internet Access**

There are parts of the world where internet access is not readily available, and there are student digital assistants that work with the internet. A lot of the great PDAs synchronize and this involves the use of internet connection. Google advices developers not to develop applications that are data intensive or need constant connection. This is so that users in developing countries can use all the functionalities of these applications; updates should be less frequent also.

**2.7 TERMS ASSOCIATED WITH STUDENT PDAs**

Below are some terms used and their definitions.

1. **Task**

This is usually as assigned piece of work often to be finished within a certain time.

1. **Schedule**

This is a plan of things that will be done and the times when they will be done.

1. **Notification**

Something that gives official information to someone, in this case the application notifies the use using an android background service. A background service runs even if the application is not running.

1. **Synchronize**

When an application synchronizes with the other devices of the student it means that if the other devices have the application, whatever the student does in one device reflects in his or her other devices.

1. **Dashboard**

This is like the home page of the application, normally this is where the student is given the summarized information of all the tasks and schedules put down.

**CHAPTER THREE**

**SYSTEM ANALYSIS AND DESIGN**

**3.1 INTRODUCTION**

The analysis phase is where the question of who will use the system is answered. In addition, the question of where the system will be used, when it will be used and what the system is actually to do. First, the analysis strategyis developed, and this is to guide the team’s effort. This strategy involves studying the already existing system and its issues and from the findings of the research build a new and improved system. (Post, 2015).

After that, the next step is the requirement gathering. These is where the requirements of the system are gotten. A lot of projects fail because of wrong requirements gathering, so great attention is to be given to this phase.

The proposed system is to be used by students in the university or secondary school. The system is to act as an assistant that helps organise the student’s daily activities.

**3.1.1 Requirement Analysis**

Software system requirements are classified into functional and non-functional requirements; these will be discussed with respect to the mobile-based personal assistant system (Sommerville, 2011).

**3.1.1.1 Functional Requirements**

In requirement gathering, functional requirements tell whoever is reading it what the system should do, the behaviour of the system in relation to the system’s functionality.

Imprecision in the requirements specification has caused many software engineering problems and for mobile applications, this is very important because to the user the interface is the product. Below are the requirements for the mobile-based personal assistant.

1. The user shall be able to add a new task for the day
2. The user shall be able to edit and delete added tasks
3. The user shall be able to view the list of all saved task
4. The system shall be able to notify the user of a task using notification manager.
5. An ID shall uniquely identify each notification in the system.
6. The user shall be able to view the classes he has in a day
7. The user shall be able to add new classes to any day he wishes.
8. The user shall be able to edit and delete his classes.
9. The system shall show the user pending tasks for a day
10. The system shall allow the user view the number of classes he has per day.
11. The user shall be able to add, edit and delete new assignments.
12. The user shall be able to add, edit and delete new projects.
13. The system shall inform the user of pending assignments and projects.

**3.1.1.2 Non-functional requirement**

Requirements that are not directly concerned with the specific services derived by the system to the users are referred to as non-functional requirements. Non-functional requirements, such as security, availability or performance usually specify or constrain characteristic of the system as a whole.

Non-functional requirements may come from required characteristics of the software (product requirements), the organisation developing the software (organizational requirements) or from external sources.

Some of the product requirements include:

1. Usability requirement: The system being a mobile-based system should have a simple and clear user interface. The use of colours and text should enable users perform tasks easily. The use of icons should also explain the text actions button perform.
2. Performance requirement: The system should make use of the android resources well and work smoothly, the system should not crash, and the use of memory and services should be in such a way such that it does not affect the performance of other applications of the system as a whole.
3. Reliability requirement: In the case of a failure the user data should persist, the application should have little or no crashes.
4. Security requirement: The application should be secure from malicious attacks and user sensitive information should be kept confidential and remain personal.

**3.2 SYSTEM ARCHITECTURE**

The application is made up of independent modules, which are integrated to make up the whole application as a whole. There is a task module which handles the to-do list and works with android notification manager to alert the user when it is time for a particular task.

Then there is the assignment module, which handles the assignments of the student, this module does not work with notification manager. The project module helps the student keep record of due projects and their progress. As time goes by the student can update the project and increase the project’s progress percentage level.

The contact module is there to save the contacts of the student so that students can have the contacts of their fellow students for easy access to their information.

The entire feature-set of the Android OS is available to you through APIs written in the Java language. These APIs form the building blocks you need to create Android apps by simplifying the reuse of core, modular system components and services, which include the following:

* A rich View System you can use to build an app’s UI, including lists, grids, text boxes, buttons, and even an embeddable web browser
* A Resource Manager is available, which provides access to non-code resources such as localized strings, graphics, and layout files
* A Notification Manager, the notification manager is key when you want to use a service to display custom alerts in the status bar
* An Activity Manager that manages the lifecycle of apps and also provides a common navigation back stack
* A Content Providers that enable apps to access data from other apps.

**3.3 SYSTEM DESIGN**

The system is designed based on the requirements documented. It is in three specifications, which are physical design, the logical design and conceptual design.

**3.3.1 Physical Design**

In system design, physical design is one of the specifications; it can be broken down into input design and output design.

**3.3.1.1 Input Design**

Input mechanisms facilitate the entry of data into the computer system, whether highly structured data, such as order information such as items, tasks or unstructured information such as comments. Input design means designing the screens used to enter the information, as well as any forms on which users write or type information.

The goal of input design for the mobile-based personal assistant is to capture accurate information for the system simply and easily. The input design will make use of textboxes, spinners, checkboxes and similar widgets to get input from the user.

All data entered into the mobile-based personal assistant must be validated in order to ensure accuracy. Ideally, to prevent invalid information from entering the mobile system, the system should not accept data that fail any important validation check (Vetter & Kumar, 2011).

**3.3.1.2 Output Design**

In system design, the goal of the output mechanism is to display information to users so that they can accurately understand at first glance. The fundamental principles for output design reflect how the outputs are used and ways to make it simpler for users to understand them. In android development and apt user interface and a simple display of information cannot be over emphasized.

**3.3.2 Logical Design**

Logical design describe processes, and this is without suggesting how they are conducted. It involves defining business entities and relationships. Physical details are defined during the design phase when these logical models are refined into physicalmodels. All these provide information needed to build the system.

**3.3.2.1 Class Diagram**

Class Diagram provides an overview of the target system by describing the objects and classes inside the system and the relationships between them. Bellow displays the various classes of the system, from the task list also called to-do list to the Lectures which is also called classes, to the project, contacts and assignment (Sparx, 2011). Figure 3.1 shows the class diagram of the student PDA.

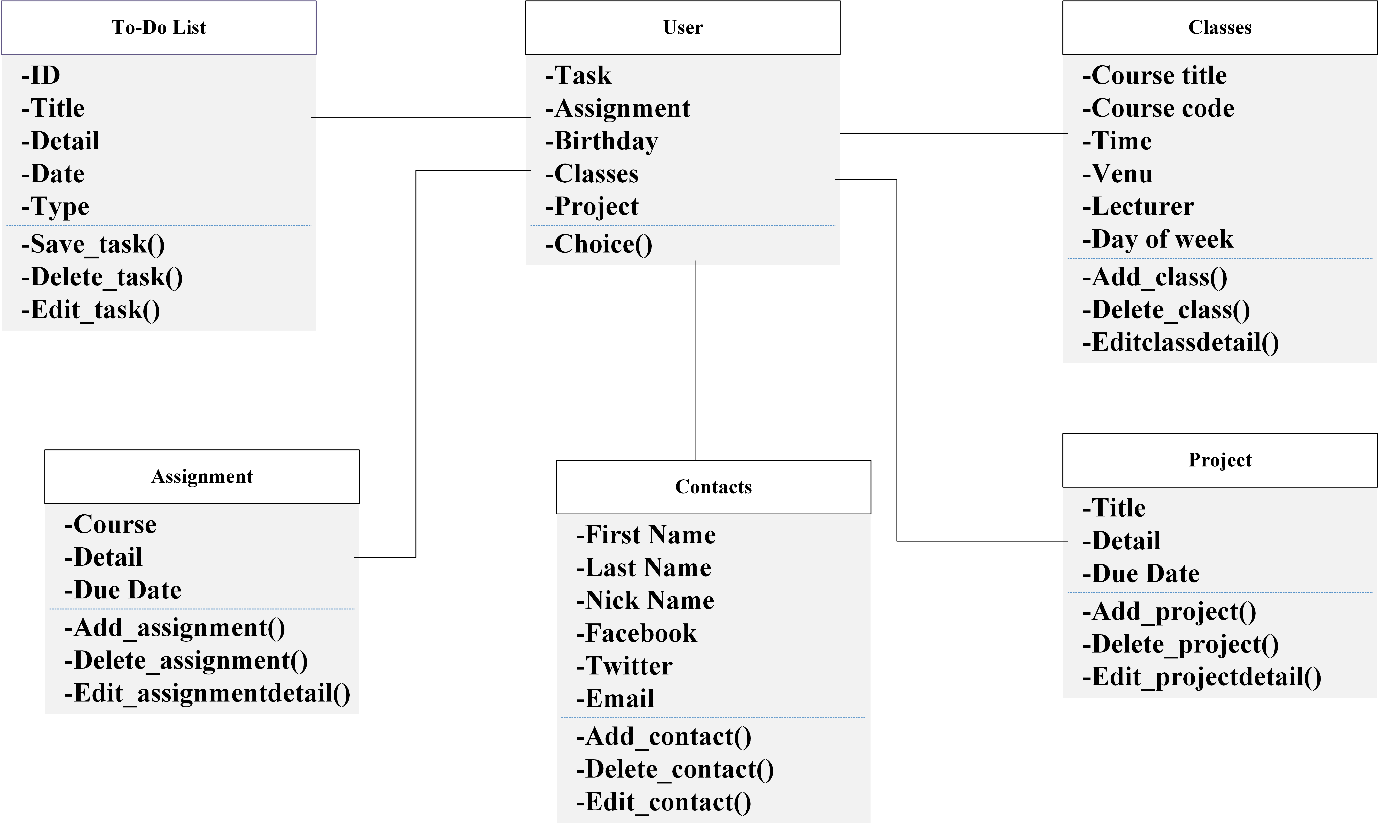


Figure 3.1: Class diagram representing a mobile-based student PDA

**3.3.2.2 Use Case Diagram**

A usecasedescribes how users interacts with the system to perform some activity, such as adding a task, creating an alarm, or even searching for information. The use cases are used to identify and to communicate the requirements for the system to the programmers who must write the system (Blaha & Rumbaugh, 2005). Figure 3.2 shows the use case diagram of the student PDA.

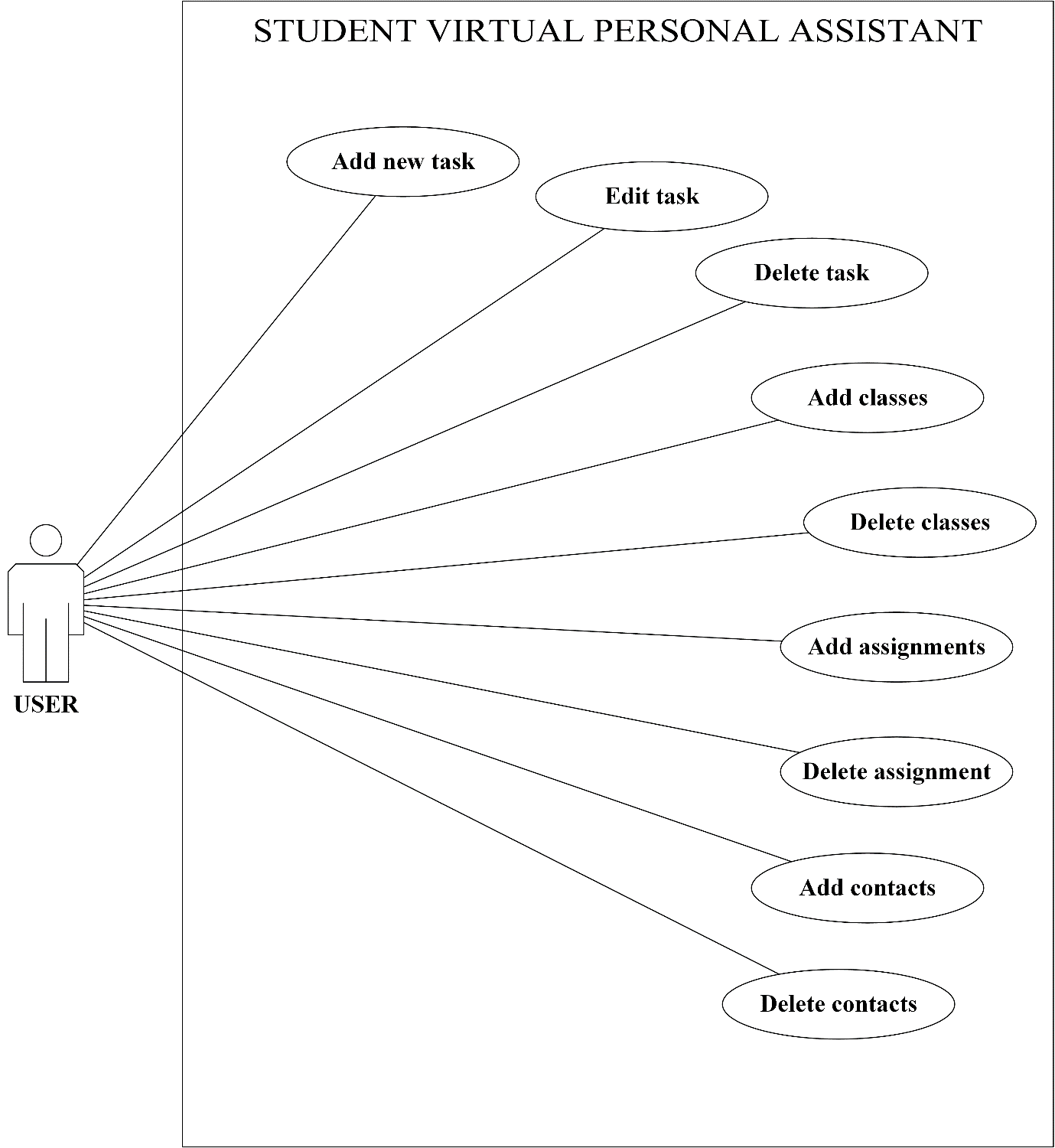


Figure 3.2: Use-case diagram for mobile-based student PDA

Table 3.1 Use case narrative for user saving new task

|  |  |
| --- | --- |
| Use Case 1 | User |
| Goal in content | User should be able to add new task |
| Priority | High |
| Preconditions | User must have the task title, task detail and time, user cannot set reminder in past date. |
| Post-condition (success end) | Save was successful |
| Post-condition (failure end) | Save was unsuccessful |
| Actor | User |
| Trigger | A user request to add a new task |
| Description (event flow) | |  |  | | --- | --- | | Actor action | System response | | 1. Select add floating button | 1. Open new activity to display form to add new task | | 1. Fill form and save | 1. Display outcome of save either a success or failed | |

**3.3.2.3 Activity Diagram**

Activity diagrams are graphical representations of workflows; they are representations of stepwise activities and actions, which include support for choice, iteration and concurrency.

Activity diagrams help to document the program flow; it shows how activities progress from one to another.

The used shape types:

* rounded rectangles represent actions;
* diamonds represent decisions;
* a black circle represents the start (initial state) of the workflow;
* An encircled black circle represents the end (final state).

Figures 3.3, 3.7 and 3.8 shows the activity diagram for a user saving a task, user deleting a task and user adding a project respectively.

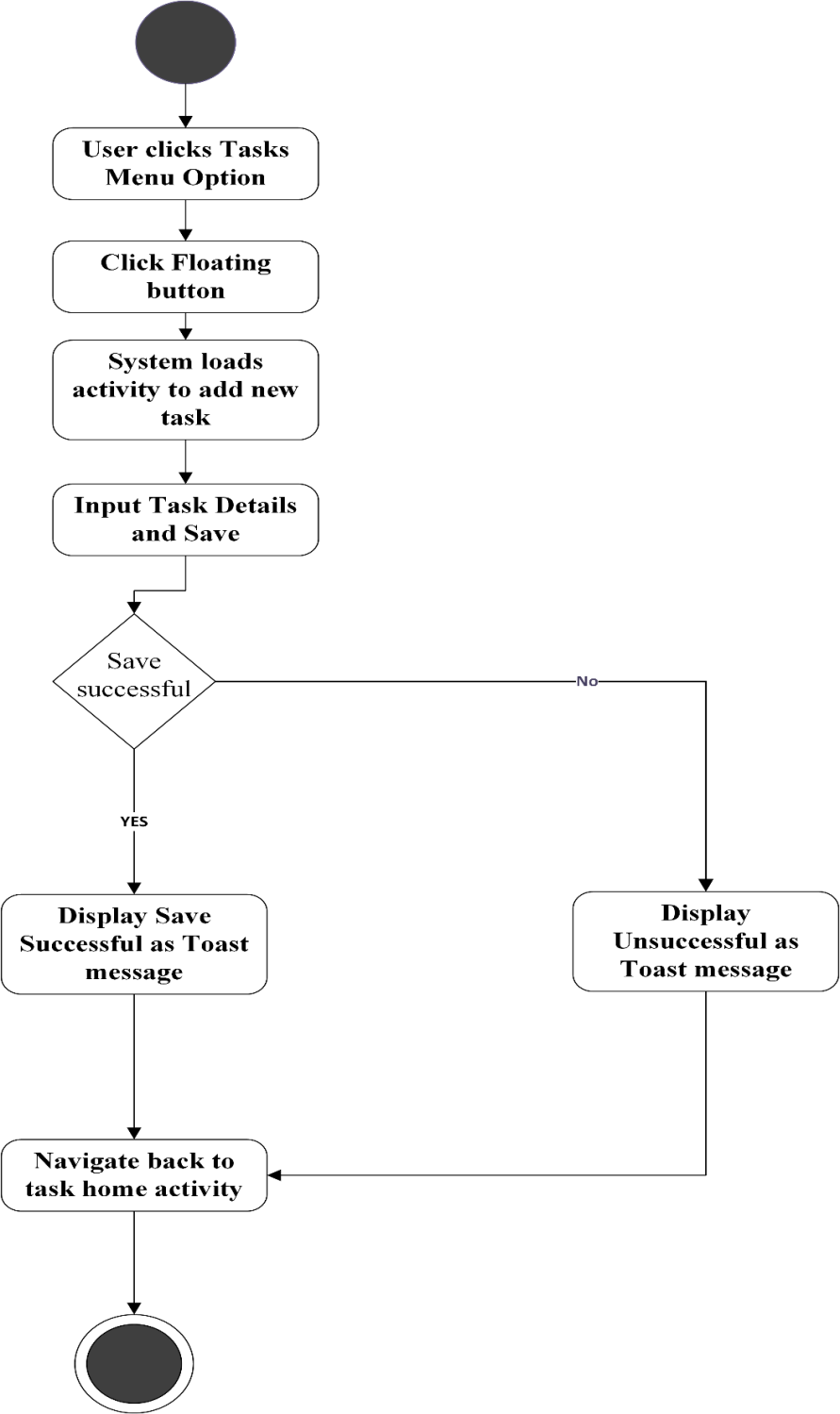


Figure 3.3: Activity diagram for user saving a task

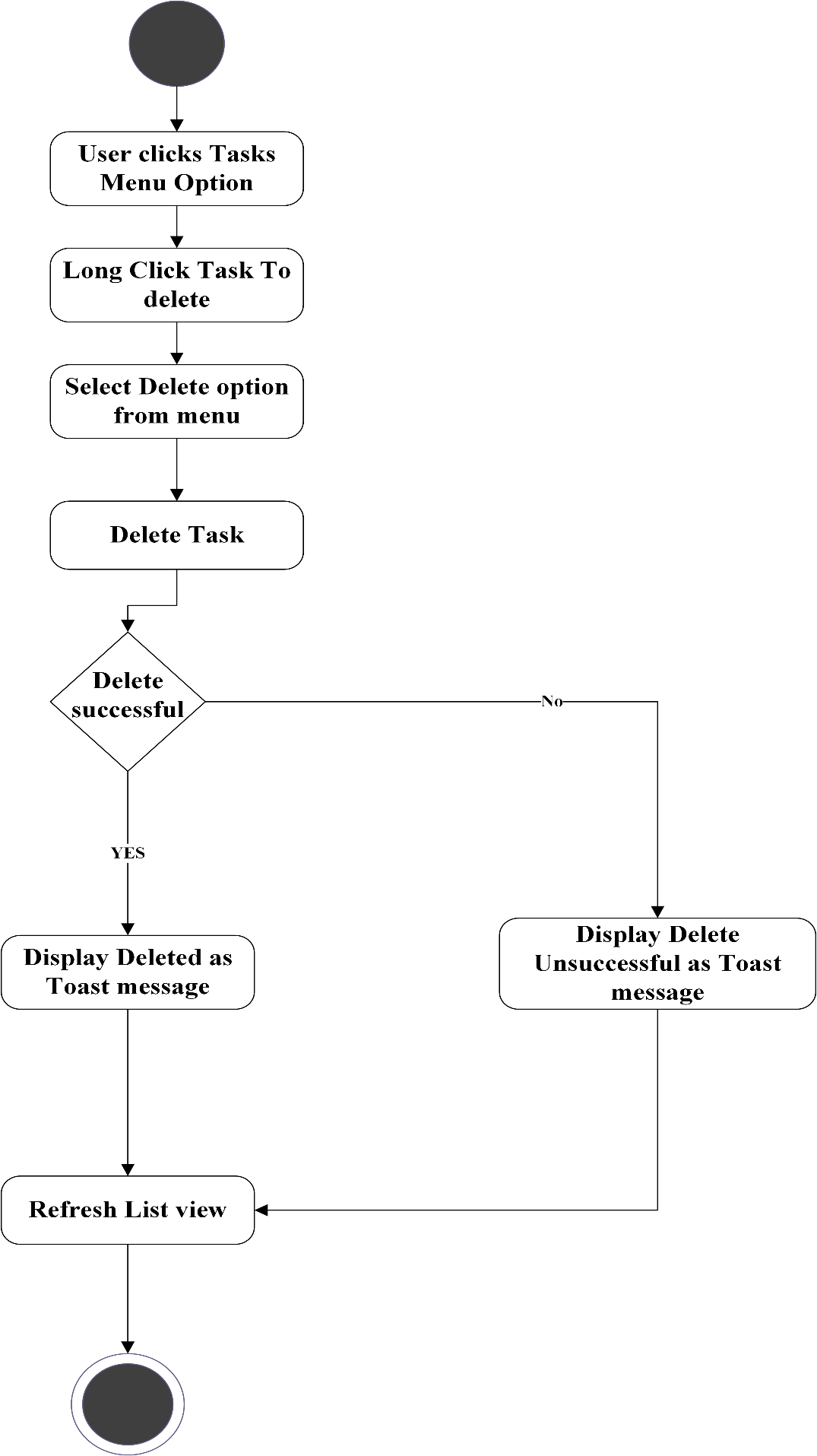


Figure 3.4: Activity diagram for user deleting a task

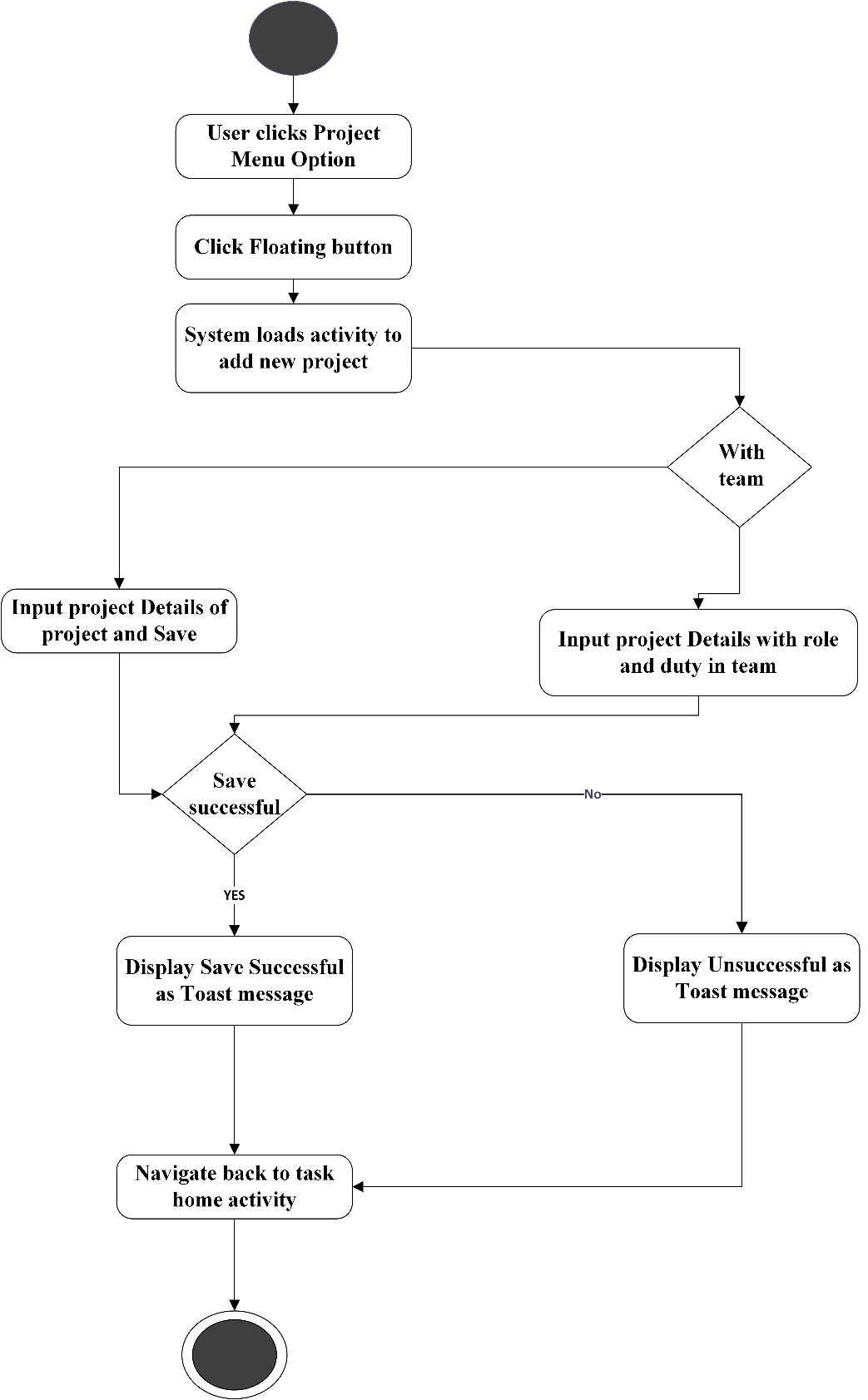
****

Figure 3.5: Activity diagram for user adding a project

**3.3.2.4 Sequence Diagram**

The sequence diagram is used primarily to show the interactions between objects in the sequential order that those interactions occur.

A Sequence diagram is used for different things but one of its primary uses is in the transition from requirements expressed as use cases to the next and more formal level of refinement. Use cases are often refined into one or more sequence diagrams (Bell, 2004). Figures 3.6 and 3.7 shows the sequence diagram for a user saving a task and deleting a task respectively.

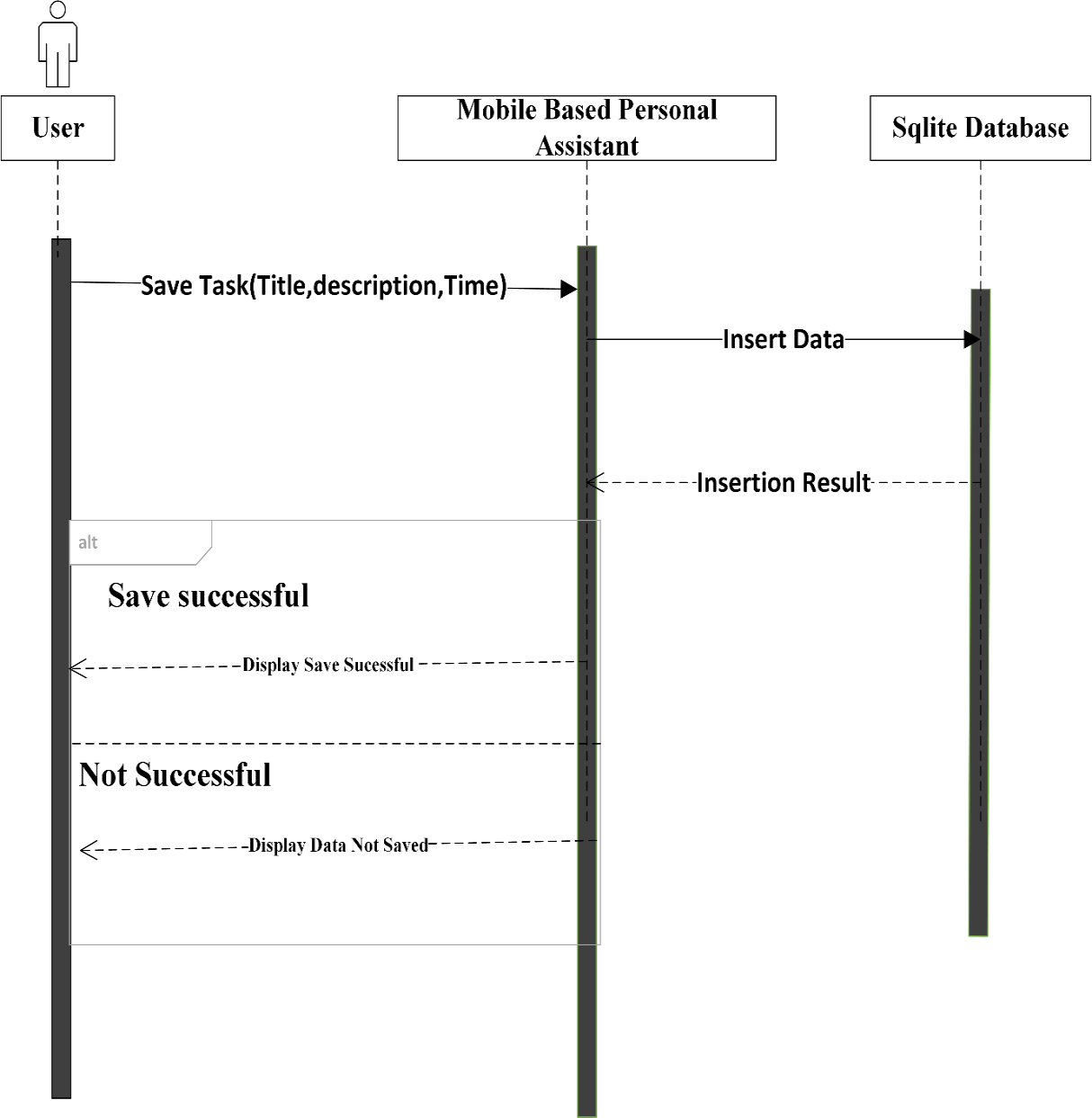


Figure 3.6: Sequence diagram for user saving a task

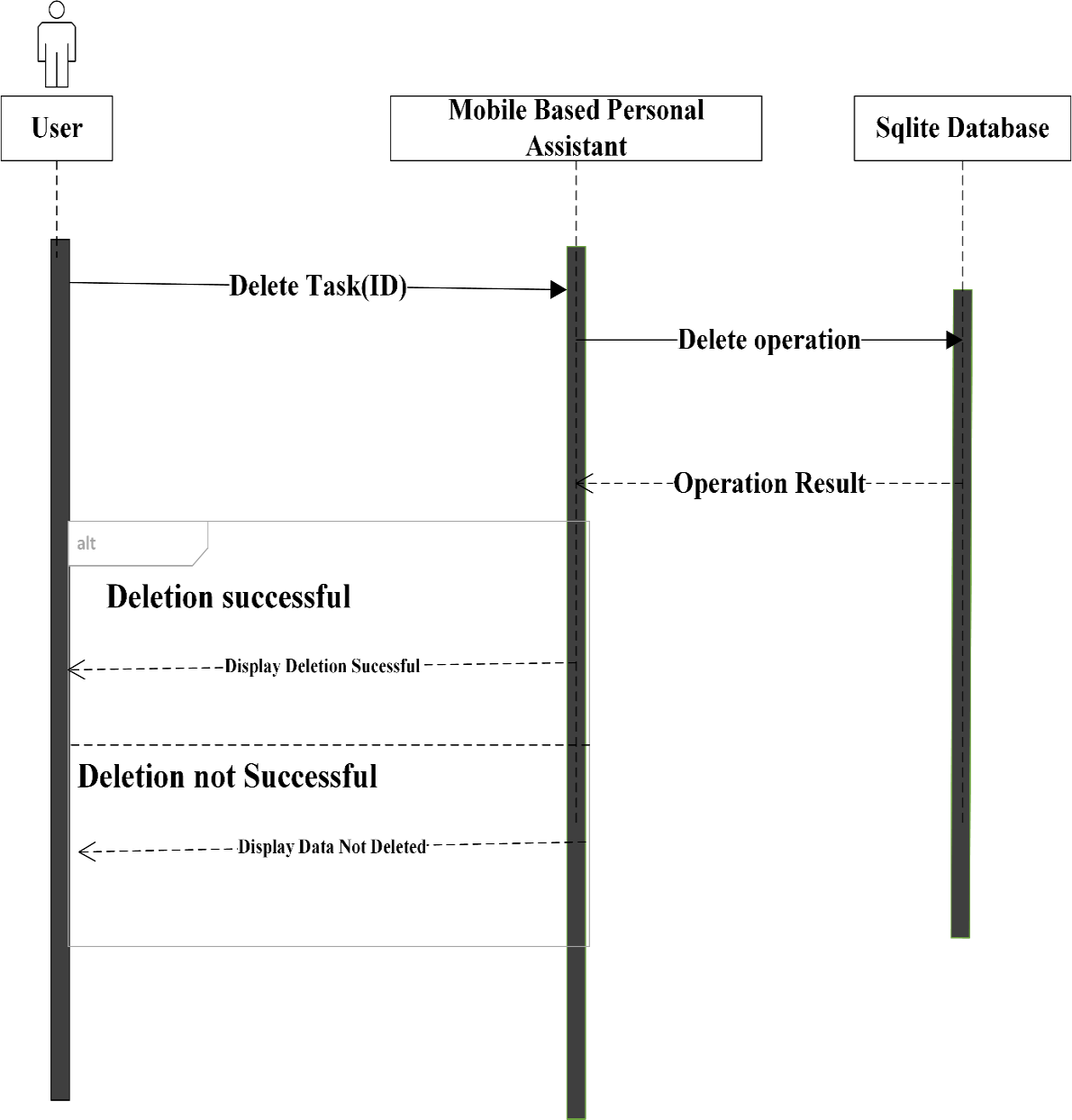


Figure 3.7: Sequence diagram for user deleting a task

**3.3.3 Conceptual Design**

The next step after all the requirements for the system have been collected and analysed is to take is to create a conceptual schema for the database, using a high-level conceptual data model. This is called the conceptual design.

The android application uses SQLite as its database, SQLite is portable and easy to use for mobile applications, it is best for situations where there is no client server design and the database is to be packaged with the application itself.

SQLite is an in-process library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine (sqlite.org, 2017).

**3.3.3.1 Description of Tables**

The student digital personal assistant database design consists of some tables that are described in the sections bellow, the name of the main SQLite database of the application is lemutask.db.

1. TASK\_LIST TABLE

The task list table is the table that stores the tasks that the user wants to accomplish, the time and the details.

Table 3.2 Task\_list table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Description** | **Data** **Type** | **Null** | **Key** |
| \_id | Unique task identifier | Integer | No | PRIMARY |
| Title | Title of task | Text | Yes |  |
| Description | Detail of task | Text | Yes |  |
| Stime | Start time of task | Text | Yes |  |
| Isdone | If task has been done | Integer | Yes |  |
| Etime | End time of task | Text | Yes |  |
| IsTimed | Check to see if task is timed | Integer | Yes |  |
| Day | Date of task | Integer | Yes |  |
| Month | Month of task | Integer | Yes |  |
| Year | Year of task | Integer | Yes |  |

1. LECTURE TABLE

This table stores the lectures of the student, the days they occur, the venue, the lecturer teaching the course and the time of the class.

Table 3.3 Lecture table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Description** | **Data** **Type** | **Null** | **Key** |
| \_id | Unique class identifier | Integer | No | PRIMARY |
| Title | Title of course | Text | Yes |  |
| Code | Code of Course | Text | Yes |  |
| Venue | Venue of lecture | Text | Yes |  |
| Lecturers | Lecturer taking course | Text | Yes |  |
| SrtClassTime | Start time of class | Text | Yes |  |
| StpClassTime | End time of class | Text | Yes |  |
| OriginalTime | Raw time of class(unedited) | Integer | Yes |  |
| Day\_C | Day of the week class is holding | Text | Yes |  |
| Credit | Credit unit of course | Integer | Yes |  |

1. ASSIGNMENT TABLE

The assignment table is the table that stores the pending assignments that the user has the time and the details.

Table 3.4 Assignment table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Description** | **Data** **Type** | **Null** | **Key** |
| \_id | Unique assignment identifier | Integer | No | PRIMARY |
| Title | Title of assignment | Text | Yes |  |
| Description | Detail of what to do | Text | Yes |  |
| Due time | Due time of assignment | Text | Yes |  |
| Isdone | If assignment has been done | Integer | Yes |  |
| Day | Due day of assignment | Integer | Yes |  |
| Month | Due Month of assignment | Integer | Yes |  |
| Year | Due Year of assignment | Integer | Yes |  |

1. PROJECT TABLE

The project table is the table that stores the pending projects that the user has the time and the details.

Table 3.5 Project table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Description** | **Data** **Type** | **Null** | **Key** |
| \_id | Unique project identifier | Integer | No | PRIMARY |
| Title | Title of project | Text | Yes |  |
| Description | Detail of project | Text | Yes |  |
| Role | Role in project, solo or team | Text | Yes |  |
| Isdone | If project has been done | Integer | Yes |  |
| Percent finished | How much percentage of work has been done | Integer | Yes |  |
| Day | Due Day of project | Integer | Yes |  |
| Month | Due Month of project | Integer | Yes |  |
| Year | Due Year of project | Integer | Yes |  |

1. CONTACT TABLE

The project table is the table that stores all the important contact the student needs to save, especially in the university where student meet themselves and exchange contacts.

Table 3.6 Contact table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **Description** | **Data** **Type** | **Null** | **Key** |
| \_id | Unique contact identifier | Integer | No | PRIMARY |
| First Name | First name of contact | Text | Yes |  |
| Last Name | Last name of contact | Text | Yes |  |
| Nick Name | Nick name of contact | Text | Yes |  |
| Description | Detail who contact is | Text | Yes |  |
| Phone number | Phone number of contact | Text | Yes |  |
| Facebook | Facebook contact | Text | Yes |  |
| Twitter | Twitter contact of person | Text | Yes |  |
| Instagram | Instagram contact of person | Text | Yes |  |
| Snapchat | Snapchat contact of person | Text | Yes |  |
| Blog | Blog of person if he/she has | Text | Yes |  |
| Address | Address of person | Text | Yes |  |
| Birthday | Birthday of contact | Text | Yes |  |

**CHAPTER FOUR**

**SYSTEM IMPLEMENTATION**

**4.1 INTRODUCTION**

In this chapter, the implementation of the student personal digital assistant is discussed. System implementation is the practice of using all the requirements and system design to construct a new system.

The student personal digital assistant has been designed with material design guidelines which one of its sayings is “having a simple interface is important”. According to Google, elegant, usable, highly rated apps also have simple user interfaces.

**4.2 SYSTEM REQUIREMENTS**

The system requirements are what is required to run the system, the minimum software and hardware that must be available in the system.

Table 4.1: The Software Requirements

|  |  |
| --- | --- |
| **Requirement** | **Software** |
| Operating system | Android OS |
| Minimum SDK version | 17 |
| Database | SQLite |
| Development tool | Android Studio |
| Programming language | Java, XML |

Table 4.2: The Hardware Deployment Requirements

|  |
| --- |
| **Minimum Requirement** |
| Screen size not more than 7 inches |
| 1GB RAM |
| 5 GB hard drive |

**4.3 THE IMPLEMENTATION TOOLS USED**

Android studio is the tool used for the development and deployment of the application. Android java and XML (extensible markup language) being the main programming languages used. Android java, which is object oriented, is for the functionalities and logic while XML was used for the design and presentation of the application.

SQLite is a portable version of the popular SQL database, unlike the server-client nature of the typical SQL database SQLite is used when a local, portable, lightweight database is needed. Therefore, it should be no surprise that a lot of android application use SQLite.

Android studio is free and very powerful, with its own device emulator, which is fast enough for development, debugging and testing purposes.

**4.4 SOFTWARE DEVELOPMENT METHODOLOGY**

A methodology is a formalized approach to implementing the SDLC (software development lifecycle). Many organizations have their own methodologies that have been refined over the years, for this project, the agile development methodology will be used. Agile development methodology is a group of programming-centric methodologies that focus on streaming the SDLC. Much of the modelling and documentation overhead is eliminated (Rapid application development, 2017).

The aim of the agile software development process is to produce useful software quickly. Figure 4.1 shows the agile development process.

Agile principles were first described by Beck et al. and have evolved ever since (Sommerville, 2011). These principles can be expressed in a series of sayings such as:

1. Satisfy the customer by delivering a working software
2. Even if change is introduced late in development, it should be embraced
3. Delivering functioning software not just incrementally but also frequently
4. Encourage customers and analysts to daily work together
5. Individuals who are trust motivated to get the job done
6. Face-to-face conversation is promoted
7. Getting software to work is the main focus
8. Continuous, regular, and sustainable development is encouraged
9. Adopt agility with attention to mindful design
10. Self-organizing teams are supported
11. Rapid feedback
12. Quality is encouraged
13. Review and adjust behaviour occasionally, and
14. Adopt simplicity

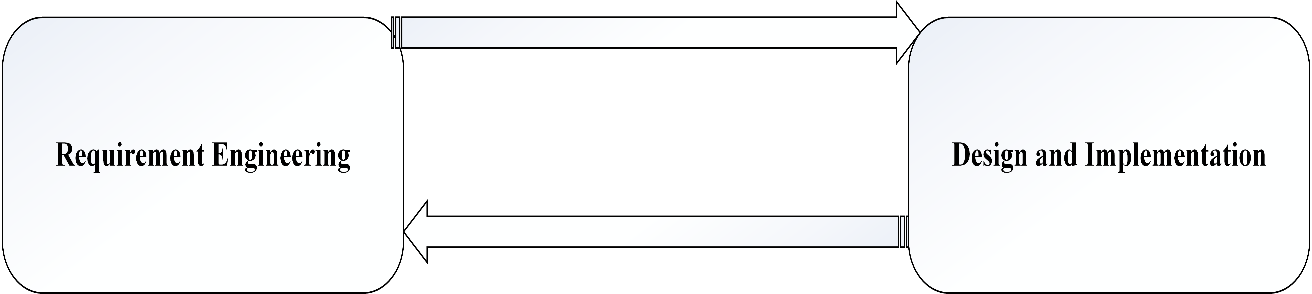
****

Figure 4.1: The agile development process

**4.5 THE PROGRAM MODULES AND INTERFACES**

This section describes the various modules of the application:

The application is made of different module which are independent of each other, the dashboard gets information from all these modules and displays it to the user in a summarized version for quick viewing.

* Student ‘s Lectures
* Student’s to-do list
* Student’s Assignment
* Student’s Project
* The Dashboard

**4.5.1 The Dashboard**

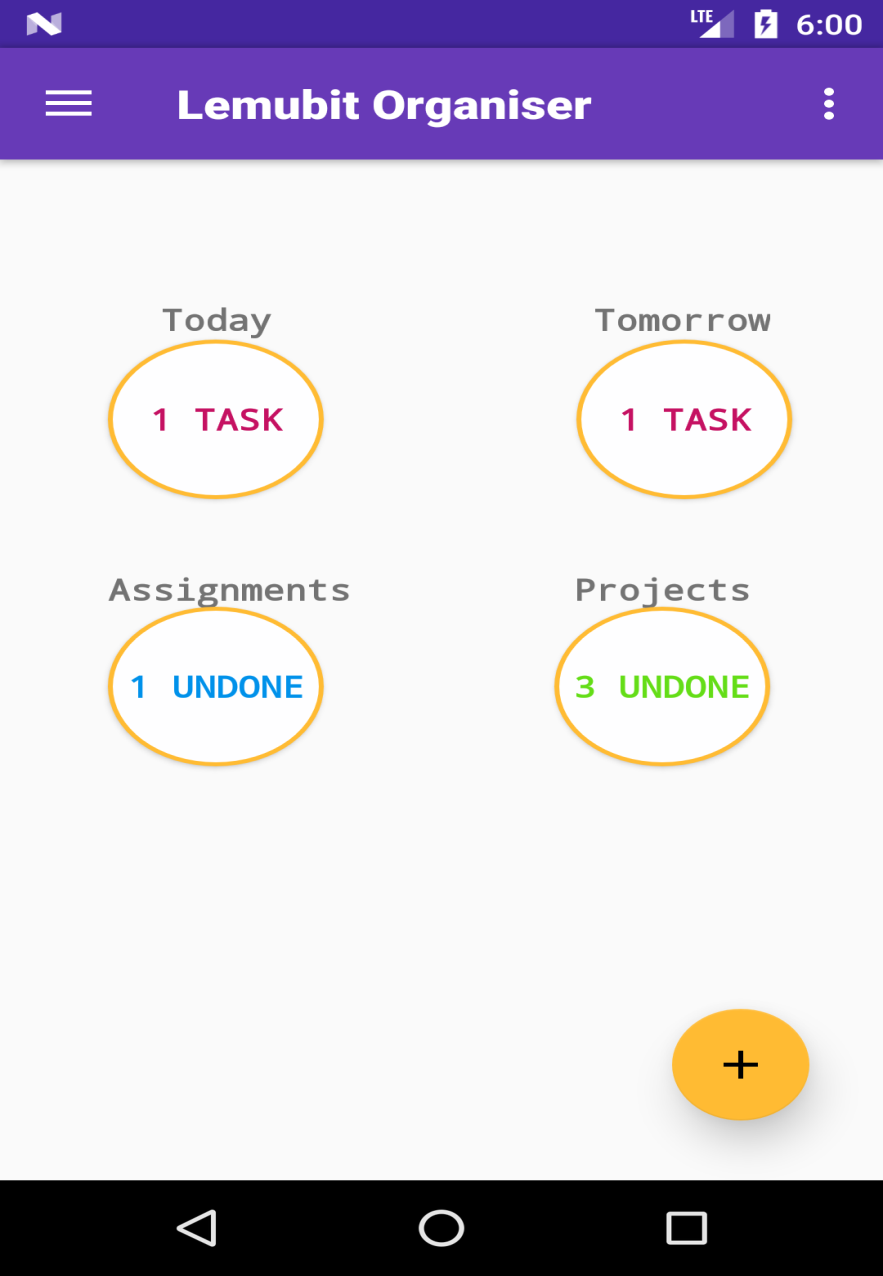
****

Figure 4.2: The Dashboard

The diagram in Figure 4.2 is the dashboard, which is the first activity the user sees after launching the application. It displays the status of things concerning the user, like due tasks, due assignments, and projects. On the bottom-right is a floating button, which is a shortcut to add task, assignment or project without using the navigation drawer. The dashboard is what sort of links all the modules together and gives the student a summarized information of pending tasks due assignments and similar information.

**4.5.2 The Navigation Drawer**

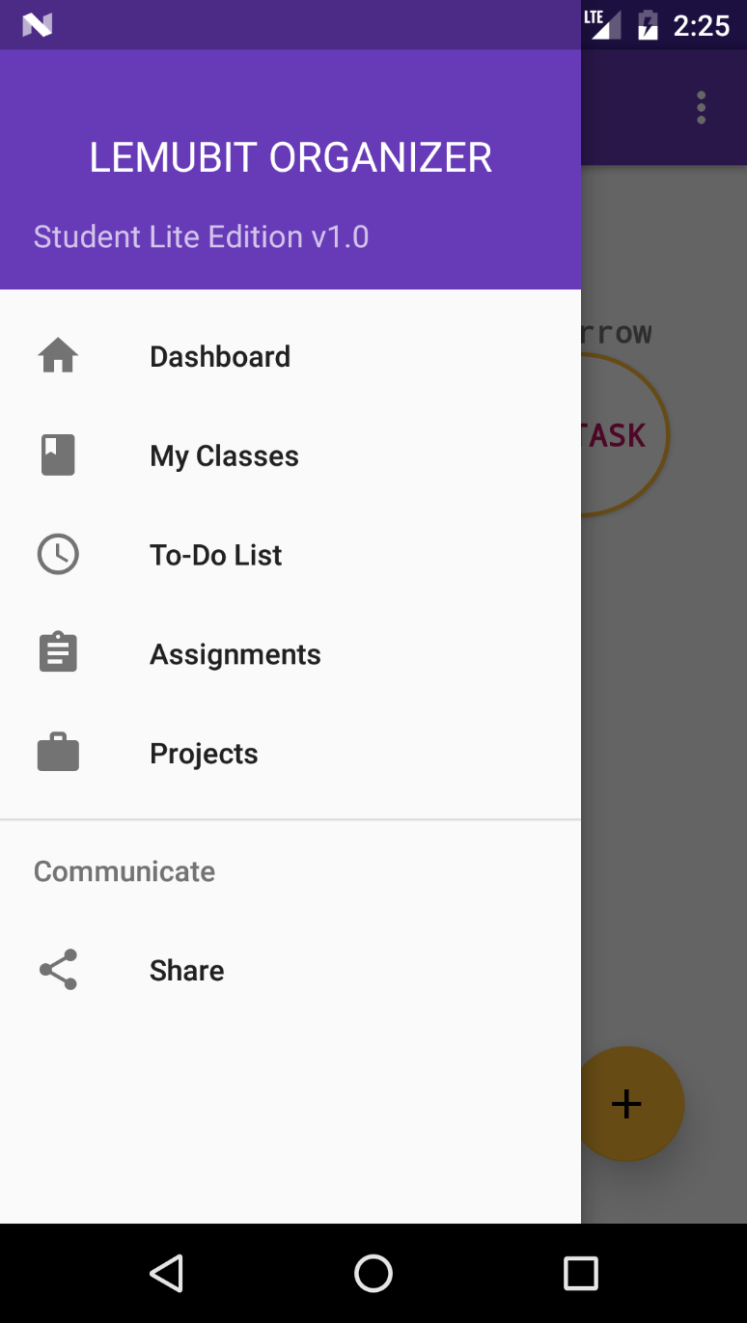
****

Figure 4.3: The Navigation Drawer

Figure 4.3 which is the navigation drawer is the part of the user interface that displays available modules or functions of the application, the user just has to swipe right on the screen to open the navigation drawer and swipe left to close it, this enables and organised navigation with delightful motion.

**4.5.3 Lecture Module**

This part of the application is to help the student keep record of his classes and know when and where each is taking place.

* Lectures view

This are tabbed fragments that arrange the classes offered by the student in days of the week in and organised form.

The classes are also arranged in such a way that classes that start earlier are at the top so that the students can view the classes without looking for which class if first and which is next. Each tabbed fragment is a class on its own; there is a tab for Monday, Tuesday Wednesday, Thursday, Friday, and Saturday.

The dashboard is made a parent of the activity so that the student can navigate back to the dashboard with the arrow seen at the top left corner of the application. The default colour of the activity is blue, which is a nice colour combination with the orange colour of the floating button at the bottom left of the activity. In order to add a new class the student clicks the floating button and the add lecture activity pops up so that the student can insert the necessary details into the application. The lectures are displayed in a list view using just the course code and the time of the lecture.

The lectures are displayed in such a way that it is arranged from the first class to the last class for the day. This is known using the time of the class, this means that students can just input any class at any time and the system will display it in the list view in an arranged format. Figure 4.4 shows the activity that displays all lectures.

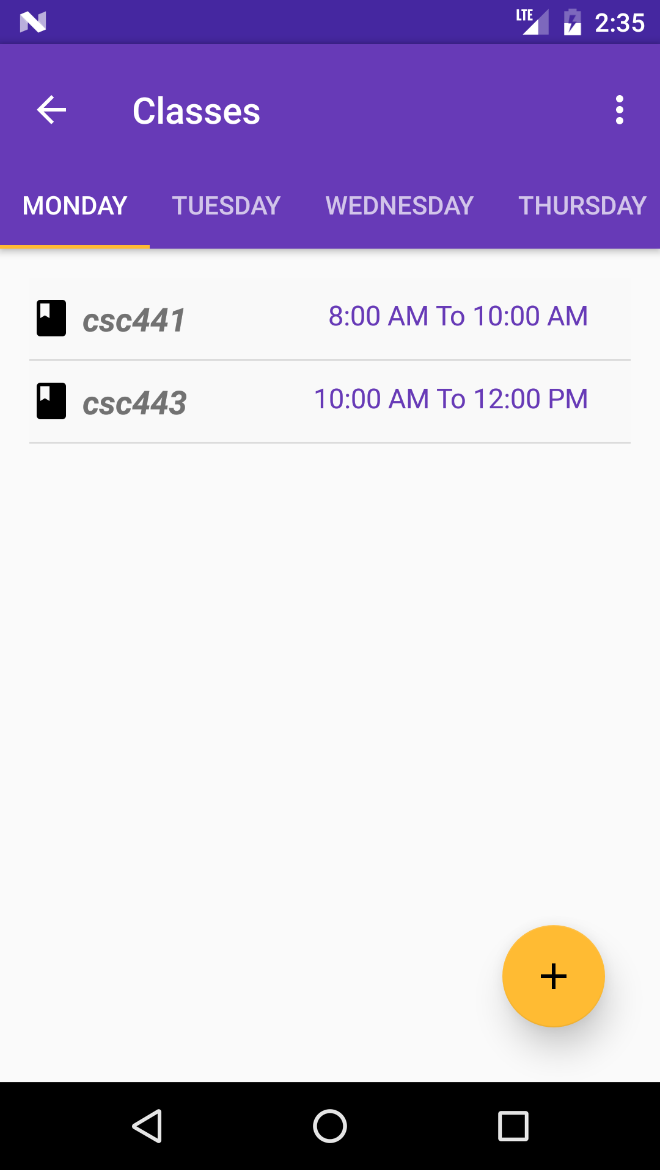


Figure 4.4: Lectures view

* Add Lecture

This activity enables the user to add courses he/she is offering, after that the system then saves this into the SQLite database and notifies the student if the save was successful. The save and cancel button are at the top right corner of the application for a better user interface look and feel. The cancel button is placed there in case the user decides to no longer save a lecture. In the activity there is the title of the course, this is the title or name of the course and not the course code, for example, this could be *introduction to computer programming*. The text box also have floating labels which is a feature of material design, the colour is the accent colour of the application, another feature of the application where you can find that colour is the floating button.

The next it the course credit unit which can be referred as to the weight of the course, or the unit of the course. The course code, for example *CSC111*, the venue of the class, this is where the lecture is holding, next is the lecturer of lecturers that are taking that course. Then the time and day of the week that it occurs is chosen. Figure 4.5 displays the add lecture activity.

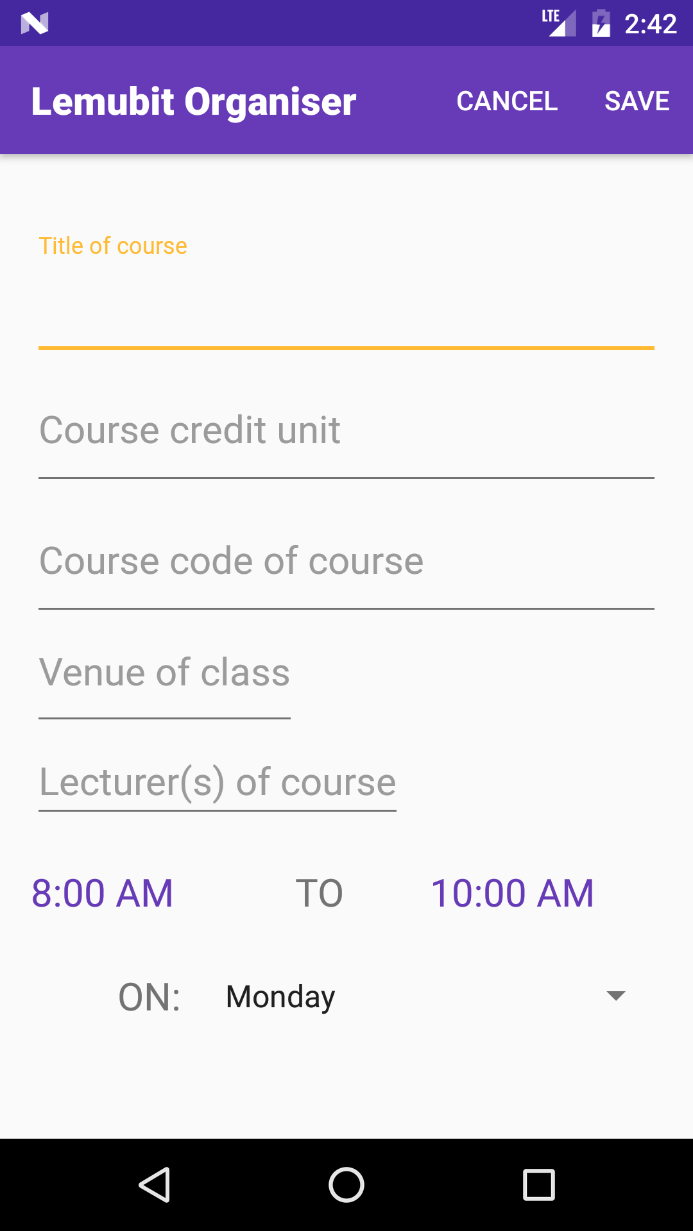


Figure 4.5 Add lecture activity

**4.5.4 To-Do List Module**

This is where the student inputs their task for the day or future days. The student can go back to check saved tasks, the system notifies the student through a notification service when time is due for a particular task.

* Task View

This is the activity that displays all the tasks which have been put down by the user, it is into two tabs showing two categories of tasks which are tasks with specific time duration and tasks which have only date. Each tab is a class on its own; this task view has the dashboard class as its parent so that that user can navigate back to the dashboard using the arrow at the top left corner of the activity.

The activity uses a list view to display the tasks that are due, the list has a vertical scroll view so that even if all the tasks cannot be displayed on the screen, the user can scroll down to see all the tasks. Beside the task is a check box at the left corner, which the user can tick after the task is done. At the bottom right hand of the activity is the floating button, which is coloured orange which is the accent of the application. On the list view, the title of the task the time of the task and the due date of the task is displayed.

If the user long clicks the list view item a popup menu displays showing a delete option, this is available so that if the user wants to delete a task they just click a particular task on the list and delete it using its identity number in the SQLite database.

When saving if the user decides not to add a time for a task it saves in the without set hourtab. This is good for tasks that have no specific timing. For example, you may want to repair your shoe, but no specific time, those kind of limits should fall into that category. Figure 4.6 shows how all the tasks are displayed.

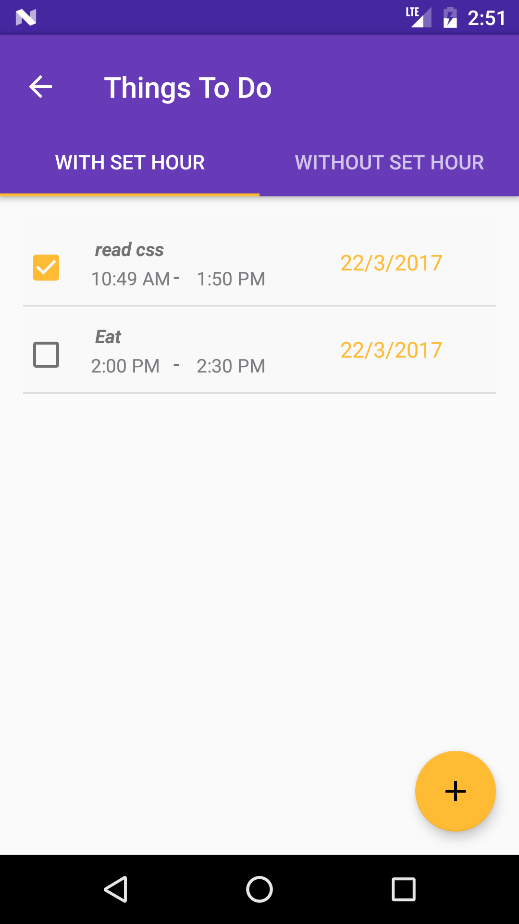


Figure 4.6: Task View

* Notification

The notifications used are handled by a background service, which still runs even after the application has been closed. Figure 4.7 shows a sample floating notification with a white background for a task on a phone running android nougat.

A service in android runs even when the application is no longer running. This is useful because the point of using a notification is to remind the user when the user is no longer using the application.

Notifications have priority and for this application, the priority was set to *high* so that the system makes sure that the user actually sees the notification. On the notification, the logo of the application is displayed alongside the name of the application. The title of the notification is not gotten from the database but is included with the service at the creation. A bundle with the intent will be sent to the ring service, which had the title and detail of the task.

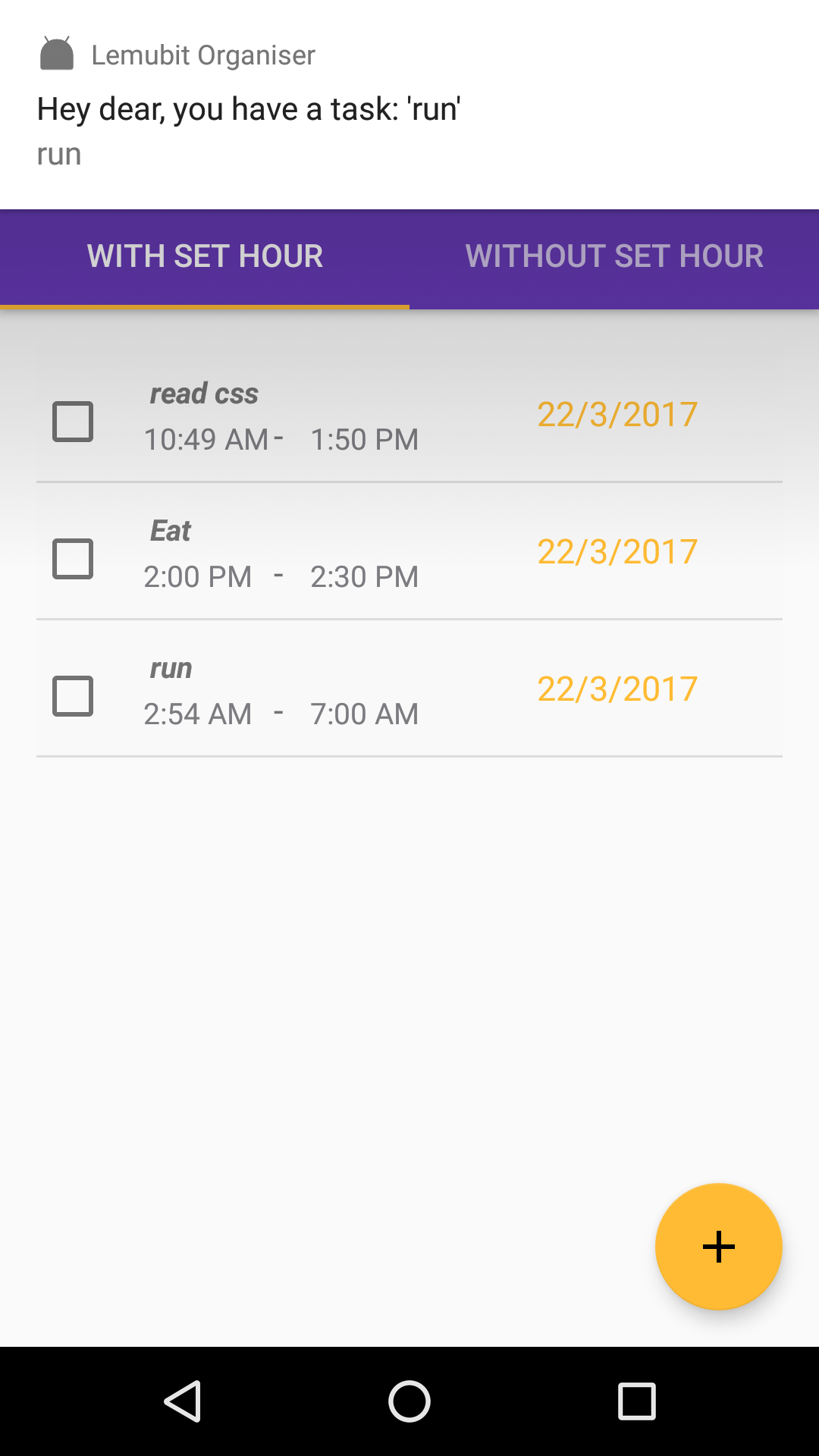


Figure 4.7 Notification Sample

* Add Task

For the user to have a task it has to be save, in order to do that the add task activity is used to save new tasks. At the top right hand corner of the activity there is the cancel and save button. The first text box is the action; the action is the title of the action needed to be done, while the next text box gives the user opportunity to write in detail what the task involves. Bellow that is a check box, which allows the user to choose whether the task has a set time or not. This will later reflect in the task view as whether it is with or without set hour.

Figure 4.8 shows the add task activity used to add tasks for the students.

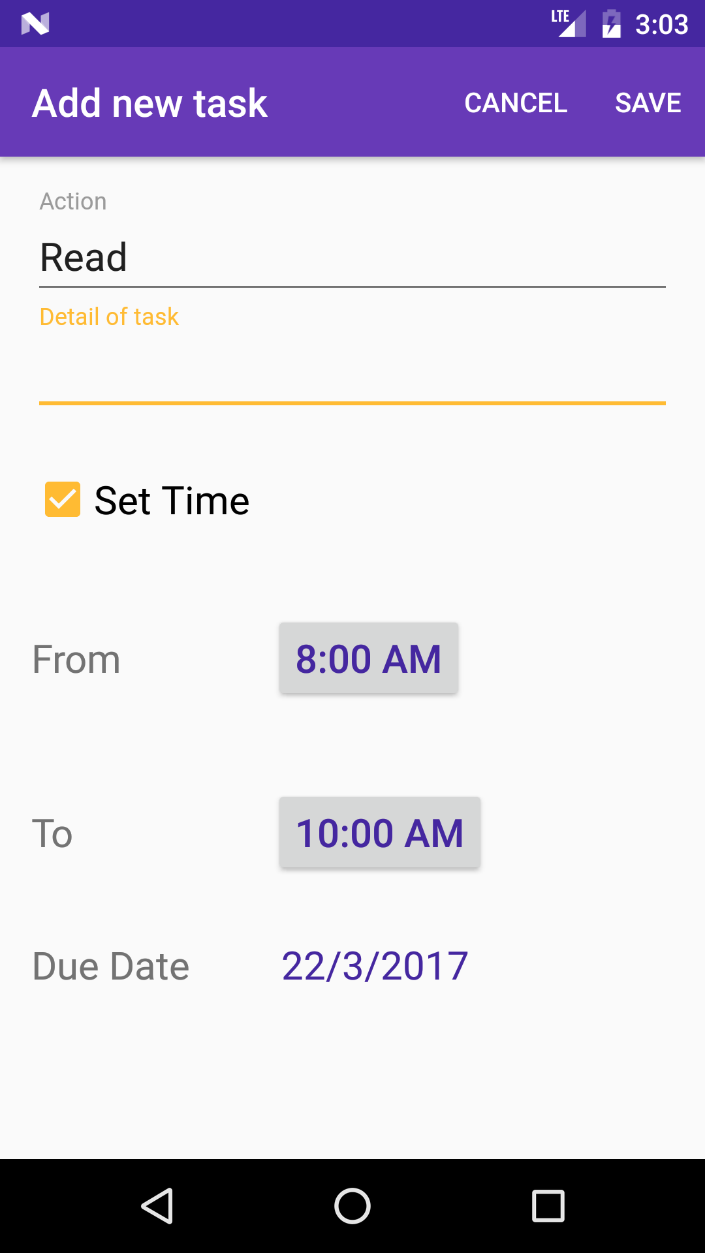


Figure 4.8 Add Task Activity

**4.5.5 Assignment Module**

If the students have pending assignments or fresh ones just given, then this is where to document them in order to help student give account of what is pending and what has been done.

* Assignment view

This is the list view where all assignments are shown for the student to view which is pending and which has been done. Figure 4.9 shows a sample assignment view showing pending assignments with a check box which allows the student to check the assignment if it has been done.

The dashboard is a parent of the assignment activity so that the user can navigate back to the dashboard using the arrow at the top left corner of the application.

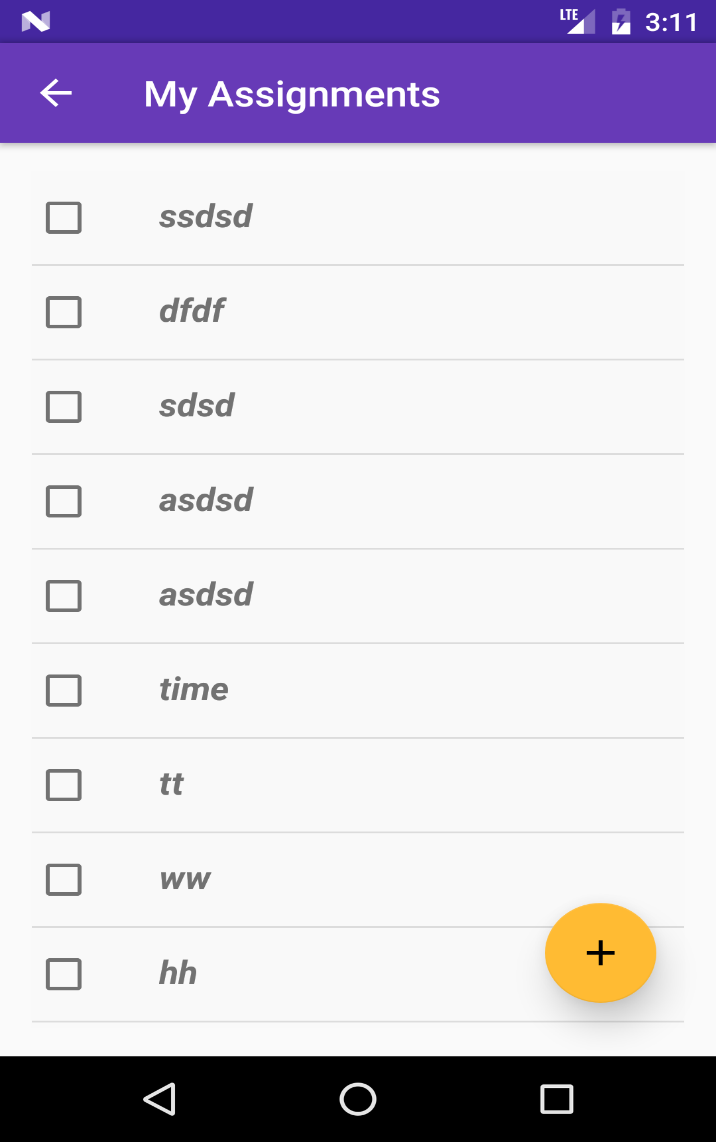


Figure 4.9: Assignment View

* Add assignment activity

As with other modules and activities there has to be a way to input the values into the database and this is done through the add activity, for assignment to add a new assignment the student should click the orange coloured floating button to open the add assignment activity, Figure 4.10 shows a sample add assignment activity.

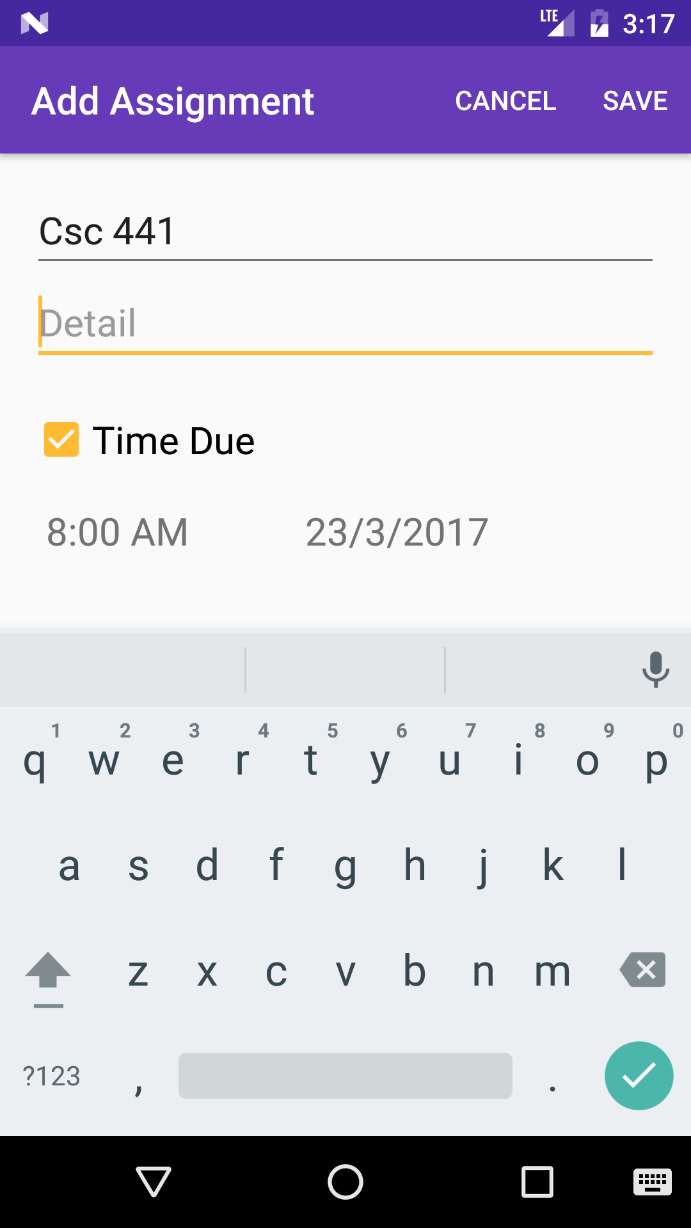


Figure 4.10 Add assignment activity

**4.5.6 Project module**

The project module is used to store projects that the student is working on, this is better that documenting it on paper, if the student is part of a team there is also a functionality of letting the user include his role in that team.

* Project View

This is where the due projects are displayed in list view, displaying the due project displaying the project title and the percentage complete. The percentage complete show how complete the project is this helps the student keep track of the progress in the project. Figure 4.11 shows the activity that views all projects.

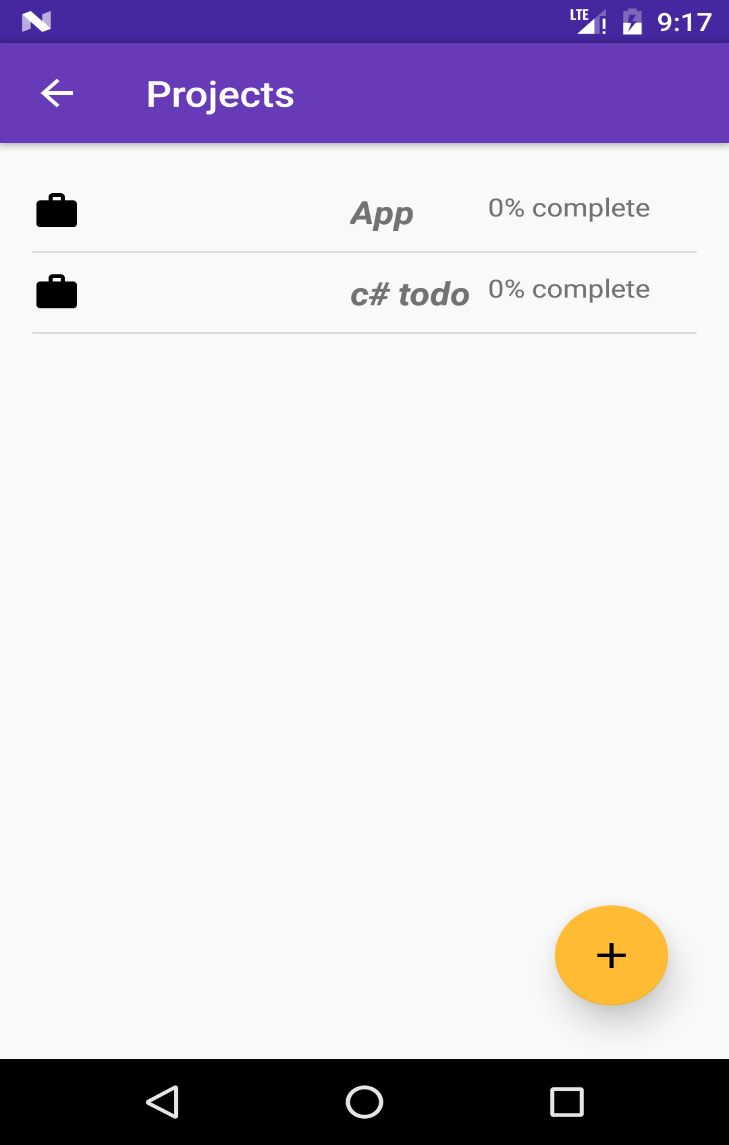


Figure 4.11 Project view activity

* Add Project view

The add project view is where new projects are added by the user. The information saved includes, the title of the project, the detail of the project, which describes what it involves, its requirements and terms. The date below is the due date of the project. Next is a check box, which can be clicked if the project is being done with a team. If the project is being done with a team the student can tick the check box and include the role played in the team and the task assigned to that role. Figure 4.12 shows the activity that adds a new project.

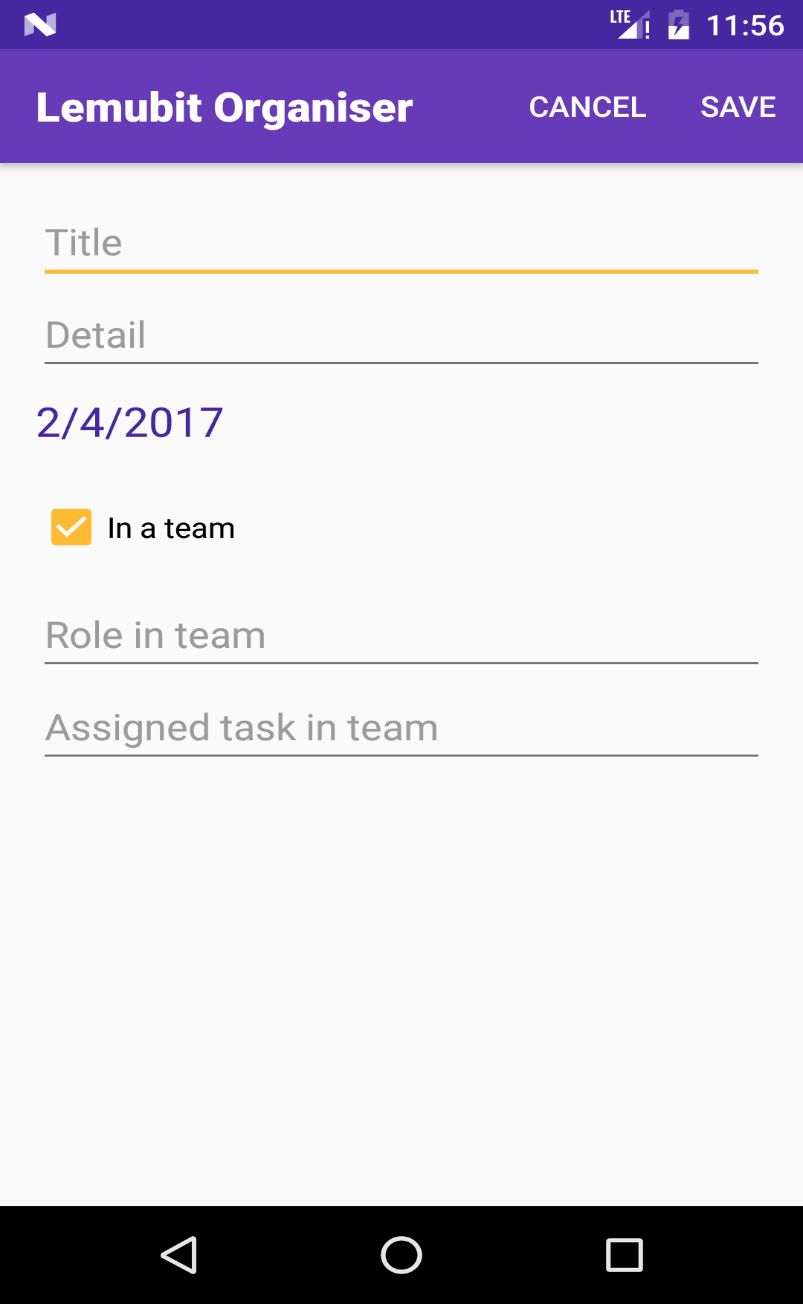


Figure 4.12 Add project activity

**CHAPTER FIVE**

**CONCLUSION AND RECOMMENDATIONS**

**5.1 SUMMARY**

The student personal digital assistant is an android application, which comprises of different major modules. It was designed with a user-oriented approach; the user interface is kept simple using Google’s material design.

The application is designed to run on a minimum API level of 17, which is also known as jellybean. The integrated development environment used to develop is android studio. Android java, XML (extensible markup language) are the programming languages used. SQLite was used to host the database.

The major modules of the project include the student’s classes, student to-do list, assignments, projects and contacts. The system being a mobile application is portable and will have significant in the student’s life.

The system makes use of android notification to notify the user of a task, which is due. This is obviously better than the method of writing tasks down on paper because a paper cannot notify a student.

The project module has an interesting feature, which allows the student to save how much of the project he has completed.

**5.2 RECOMMENDATIONS**

In order for the system to be used effectively, students need to be adequately educated about the system and how to use it. Students should also be encouraged to use the application because a user cannot know how good a product is neither can he benefit from it if he has not used it. For best experience, students should use android marshmallow for the performance and beautiful aesthetics.

The to-do module should be used every day to organise tasks the assignment module should be used to save assignments so that the student can keep note of due assignments and the ones done.

**5.3 CONCLUSION**

The student mobile personal assistant will aid the students in their day-to-day activities and planning. Students who were asked to review the applications gave a positive review; this is good because it signifies that students are satisfied with the functionalities and user interface. The material design guideline was used to give a nice look and feel to the application.

More and even better improvements will be made to the application’s future releases to include or remove features desired by students. Any student using such an application will be more productive than his or her peers, will have less things to think about and will be more organised.

**REFERENCES**

Accomplish (2017). Retrieved from http://play.google.com

Assistant Android Application (2017). Retrieved from <http://www.techcrunch.com>

Blaha, M., & Rumbaugh, J. (2005). Object-oriented modeling and design with UML2 (2nd ed.). Upper Saddle River, NJ: Pearson Education.

Donald, B. (2004). The sequence diagram. Retrieved from

http://www.ibm.com/developerworks/rational/library/3101.html

Dennis, Wixom, and Roth (2012), *System Analysis and Design*. Fifth edn. p. 13

Ian Sommerville (2011), *Software Engineering*. Ninth edn. p. 82

McCarthy, M. (2013). *The Daily American Planner*. Retrieved from

<https://www.bostonglobe.com/ideas/2013/06/01/the-daily-planner-american-history/WncDRG5hq7B9m0w3cE5jkM/story.html>

My Effectiveness (2017). Retrieved from <http://play.google.com>

My Study Life (2017). Retrieved from http://play.google.com

Post, A. (2015). *System Analysis and Design*. Lecture, University of Kentucky.

Rapid Application Development (2017). Retrieved from <http://w3computing.com>

Roubit (2017). Retrieved from http://play.google.com

SQLite Database (2017). Retrieved from <https://www.sqlite.org/about.html>

Sparx (2011). *UML Structural Models.* Retrieved from Sparx systems:

<http://www.sparxsystems.com/enterprise_architecture_user_guide/9.3/standard_uml_models/structuraldiagrams.html>

The Android Software Stack (2017). Retrieved from <https://developer.google.com>

Vetter, D. J., & Kumar, P.D. (2011). *Overview of System Analysis & Design.* Retrieved

from <http://download.nos.org/cca/ccal.pdf>

What is Systems Architecture? Retrieved from

<http://www.lix.polythechnique.fr/~golden/systems_architecture.html>

**APPENDIX**

Code of Wednesday fragment in lecture module:

package com.example.charl.lemutask;

import android.content.Context;

import android.database.Cursor;

import android.os.Bundle;

import android.support.annotation.Nullable;

import android.support.v4.app.Fragment;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.View;

import android.view.ViewGroup;

import android.widget.ListView;

/\*\*

\* Created by Charles on 19/02/2017.

\*/

public class class\_wednesday extends Fragment {

View view;

sql\_helper\_lemutask sqlht;

public ListView lvItems;

@Nullable

@Override

public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {

view = inflater.inflate(R.layout.classes\_wednesday\_fragment, container, false);

sqlht=new sql\_helper\_lemutask(getActivity());

lvItems = (ListView) view.findViewById(R.id.wednesdaylist);

return view;

}

@Override

public void onResume() {

super.onResume();

// Query for items from the database and get a cursor back

Cursor todoCursor = sqlht.getwednesdayclass();

if(todoCursor.getCount()==0)

{

Log.i("Message","No data");

return;

}

else{

//Log.i("Message","There is data "+ todoCursor.getColumnCount()+todoCursor.getColumnName(0)+todoCursor.getColumnIndexOrThrow("\_id"));

// Setup cursor adapter using cursor from last step

Context c=view.getContext();

classes\_adapter todoAdapter = new classes\_adapter(c, todoCursor,true);

lvItems.setAdapter(todoAdapter);

}

}

}