READ: Moodle Reading Assistant

L. P. D. S. Chandraweera
Department of Computer Science and Engineering, University of Moratuwa
Katubedda, Sri Lanka
sugeesh.13@cse.mrt.ac.lk

***Abstract*—Moodle is the world's most popular learning platform designed to connect educators, administrators and learners. Moodle is widely used in universities and institutes. Most university modules have corresponding recommended books. Chapters of those books are used in each lecture. Students will be able to understand the lectures clearly if they read the related course material before the lecture, but unfortunately most of the time it doesn't happen. "READ" is a reading assistant for Moodle. "READ" application will help students to familiarize with the related reading material before the lecture by issuing weekly reading material reminders for each course they are following.**

Keywords— User Interface (UI); Graphical User Interface (GUI)

I. INTRODUCTION

Moodle is the world most popular learning platform designed to connect educators, administrators and learners. Currently Moodle is used by 79 million users worldwide [1]. It's an open source project and one of the most successful open source projects ever. The Moodle platform is coordinated by "Moodle HQ" and it's very accurate. This is provided freely as Open Source software, under the GNU General Public License. Moodle has one of the largest open source communities in the world. Moodle supports multiple languages and currently there are more than 120 languages supported by Moodle [1]. Moodle provides developer docs for developers who work with Moodle code, plugins and themes. Moodle community encourage for Moodle plugin development all over the world [2].

Moodle is widely used in universities and institutes. Most university modules have corresponding recommended books. Chapters of that book are used in each lecture. Students will be able to understand the lectures clearly if they read the related course material before the lecture, but unfortunately most of the time it does not happen. "READ" is a reading assistant for Moodle which will help students to familiar with the related reading material before the lecture. "READ" mobile application will remind students about their reading material for each course in every week. This process will be practical because usually students use their mobile devices every day. Mobile app will notify students, what they have to read before the lecture. Students will also be able to check what they have missed and what they have read, easily.

The paper is structured as follows. Section 2 discusses the literature review of this mobile application. Section 3 indicates the system models and section 3 is about system implementation procedures.

II. LITERATURE REVIEW

Moodle is the most popular learning management platform that is used in the world. It is PHP based open source project and had large community. Developers can use Moodle docs to learn about the source code of the project and also they can mainly develop plugins in several categories. They can contribute to the Moodle mobile application as well [3]. "READ" plugin should be able to add new two resource types to the Moodle as well as it should be able to connect with mobile application.

There is a Moodle activity plugin named "Book" which maintained by Petr Skoda [4]. This book module makes easy to create multi-page resources with a book-like format. This module can be used to build complete book-like websites inside the Moodle course. But it doesn't allow adding existing book as Book resource and also it doesn't support add chapters to each week timeline. READ Moodle plugin will be able to add each chapter of the recommended book to the course time line in a particular week.

Moodle mobile official application provides some good features to the users but it will not be able to work with "READ" Moodle plugin [5]. "READ" Mobile application will be a separate mobile application. This Mobile application provides users the ability to know about the reading material that related to each course. It will notify students two days before the lecture about the reading material they have to read. By comparing the READ application with the Moodle mobile official application, READ application is a simple application developed only to remind and store the reading materials of the course. Official application is much complex than READ application and it stores and works with many features than READ application. "READ" mobile application could be developed as a plugin for Moodle Mobile Application, but because of high complexity of getting data and notify things will be much complex.

There are many effective reminding apps in the mobile market, but in all those apps, reminders have to be added manually by the user. "READ" mobile application will add the reading materials automatically by sync with Moodle.

III. System Models

A. System Requirement

Functional requirements of the system mainly include requirements related to the mobile application and Moodle plugin. Moodle plugin requirements include adding, editing, deleting "Recommend Book" for each course and "Recommend Chapter" for each course week. Mobile application requirements include notification generation, mark finished chapters, check missed chapters and sync with the Moodle server. The non-functional requirements of the plugin are Moodle backup system should work with the plugin, uninstalling plugin.

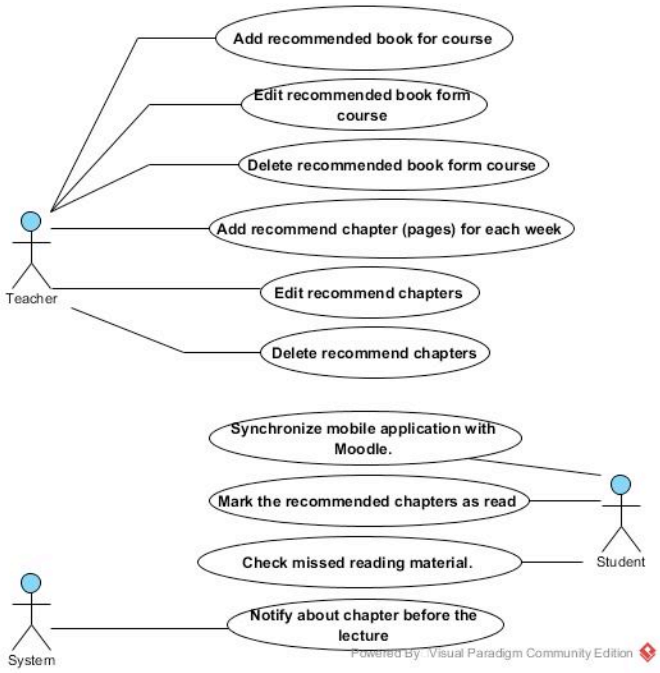

Fig. 1. Use case diagram

As in Figure 1, teachers will able to add/ edit / delete "Recommend Book" for each course. Also Teachers will be able to add "Recommend chapter" to each week of the course. Teacher should first add the "Recommend Book" to the course and then he will be able to add "Recommend Chapters" to the course of that book. Teachers can add multiple Recommend Books for the course and add different chapters of different books for each week. Student should be able to sync the mobile application with the Moodle server by providing Moodle server URL, username and password. Students should be able to mark the recommended chapters as finished. Also students should be able to check the missed reading material. Mobile application will automatically notify about recommended chapters two days before the lecture.

B. System Design

1) Mobile Application Architecture

Because of mobile application use Ionic Framework, the system has multi-layered architecture where the system is divided into three layers namely the physical layer, the logical layer and the data access layer [6]. As in Figure 2, the data access layer communicates with the SQLite database. Logical layer will communicate between Data access layer and Physical layer. Logical Layer has the READ application navigation system and logics (notifications generation methods). Data access layer uses Cordova SQLite plugin [7].

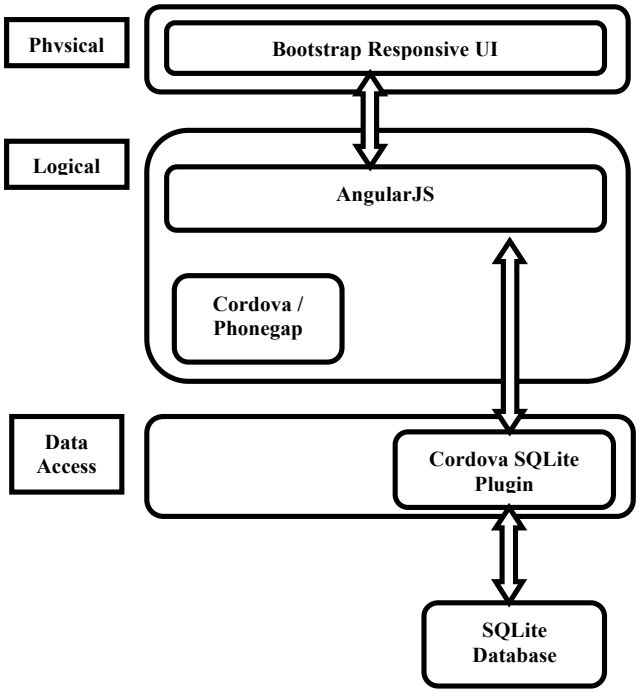

Fig. 2. Mobile application architecture

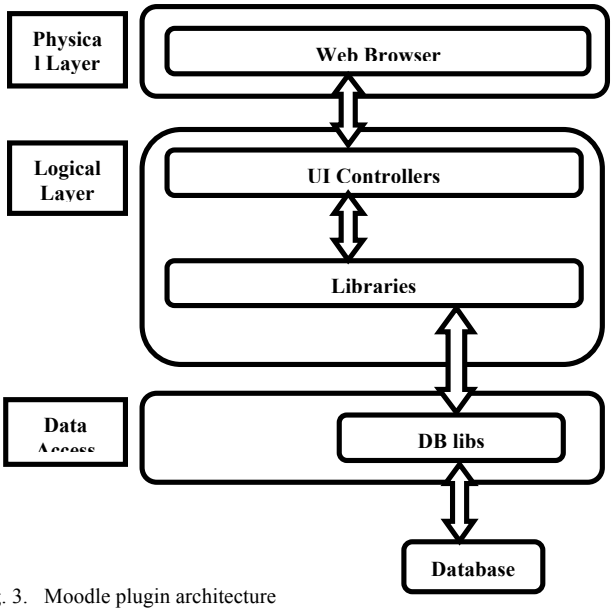

Fig. 3. Moodle plugin architecture

2) Moodle Plugin Architecture

As in Figure 3, Moodle uses backend database connected with Moodle Library. Plugin designing is done in Logical Layer. Data access layer predefined and GUI components predefined. In Plugin developers able to use those predefined methods in Data Access layer and GUI components.

Plugin contains "db/install.xml" file which contains the tables that should be newly added to the database. Also it has "db/access.php" file which contains plugin accessibility for each user types. It contains "mod_form.php" file which displays the GUI part for add/edit for resource module.

C. Database Design

1) Mobile Database

Moodle mobile SQLite database contain mainly three tables. As Figure 4, it will store Courses, Recommend Books and Recommend Chapters. When the database is synchronized with server databases, the needed data will retrieve from Course table in Moodle server. Also the data of Recommended Book and Recommended Chapter tables in the server will return to mobile application with the relevant section number. This section number will be used to identify the week of the course and reading material that should be added.

2) Moodle Plugin

Figure 5 shows the ER diagram for updated Moodle server database. Whole Moodle database contain more than 250 tables. This shows the tables used by the READ plugin and how they are connected. Some tables have more than 20 columns. So in this diagram only the most essential attributes are shown. Recommend books and Recommend chapter tables are newly added to the database. When synchronizing with the mobile application, other tables are used. The enrolled courses for the user is stored in enroll table. When installing the Plugin it will add two rows to Module table. When adding a module to the course it will add another row to "Course Module". Course_Section id will add to Recommend Book and Recommend Chapter.

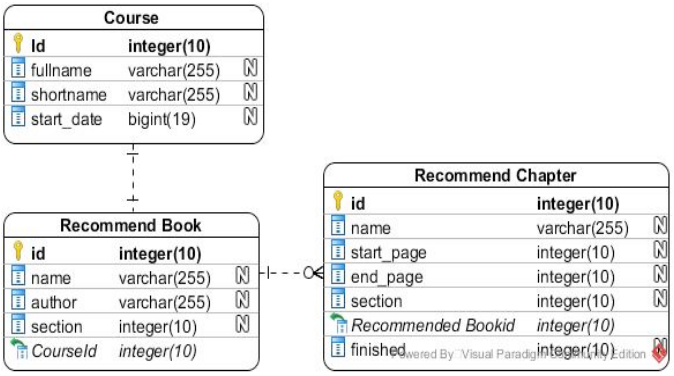

Fig. 4. ER diagram for Mobile application

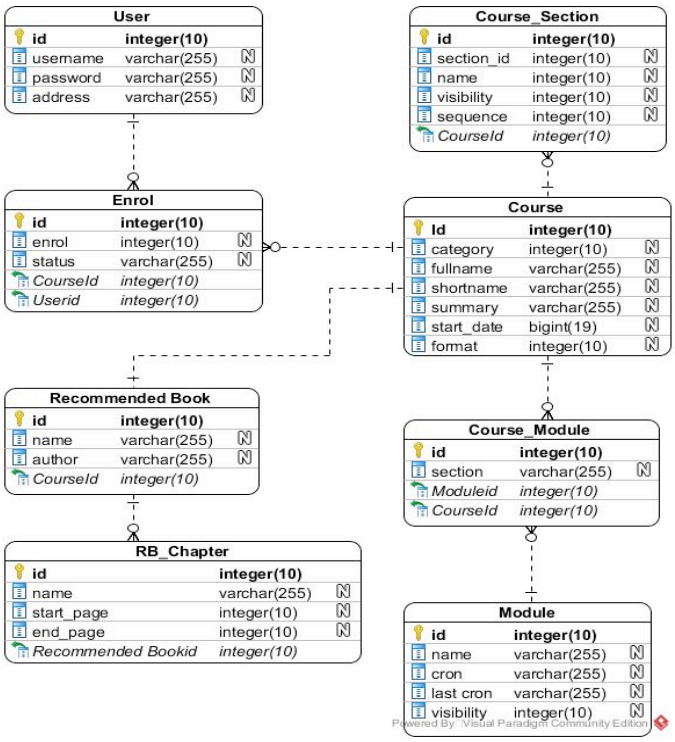

Fig. 5. ER diagram for Moodle plugin

IV. System Implementation

A. Implementation Procedure

1) Mobile Application

Ionic mobile app framework is used to develop the mobile application. Ionic offers a library of mobile-optimized HTML, CSS and JS components and tools for building highly interactive native and progressive web apps [8]. Atom IDE was used for developing the application. "Ionic Creator" was used for developing ionic mobile application basic views. It is a very easy tool to develop ionic application GUI. "Ionic Lab" used for check the preview of the application in basic developing steps. "cordova/phonegap sqlite storage" plugin is used to store SQLite database with the application. "cordova Local-notification " plugin is used for generating local notifications in the mobile application. After adding these two plugins to the mobile application, "Ionic Lab" was unable to give a preview as it is. So actual android device was used for previewing the application afterward.

2) Moodle Plugin

Moodle source codes use PHP language. PHPStorm is used for building the plugin because it provides many additional features when coding in PHP language. XMLDB editor is used for creating the table for the database (The XMLDB editor is a tool for making the .xml files) [9]. Official activity plugin documentation of Moodle used to identify the basic necessary files for activity plugin and identify the coding that need in each file. [10]

```
allCourses = get courses from server for user;
For each course from allCourses
        If it is new course add the course to
        local database;
        Get all books for course id;
        Insert new books into local database;
        Get all chapters for course id;
        Insert new chapter into local database;
        Check unnecessary books and chapters in
        local database and delete them;
Check unnecessary courses in local database and
delete them;

Delete all existing notifications;
notFinishedChapters = get All Not Finished
                      chapters;

for each chapter from notReadChapters
        alarmdate = set date according to the
                    section;
        reduce 2 days from date.
```

Fig. 6. Sync Algorithm Pseudocode

Syncing with the Server is the most complex algorithm in the program. First, for the corresponding user, app must get all the details of courses user have been enrolled. Then for each course, obtaining all books and checking whether that course exist in the database is done. If it is new course, then it is inserted it to the local database. Then for each course all new books and chapters for that course are obtained and are inserted to the local database. Delete the unnecessary books, chapters stored in the local database. Delete the unnecessary courses with their relevant details from the local database. Then delete all existing local notifications. Then get all not read chapter details form the database. Then for each chapter get Unix epoch time and dates according to section value (section value contains which week the resource is added). Reduce two days from data (Notification must be raised two days prior to the lecture). Convert epoch date to JavaScript date and modify its time to 8.00AM. Then set local notification settings and add the notification.

B. *Main Interfaces*

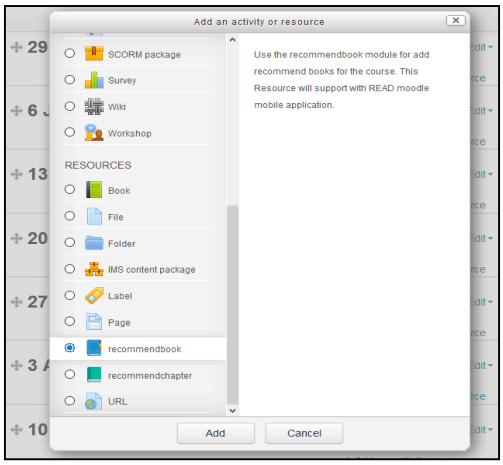

Fig. 7. Add Recommend Book/Chapter Interface

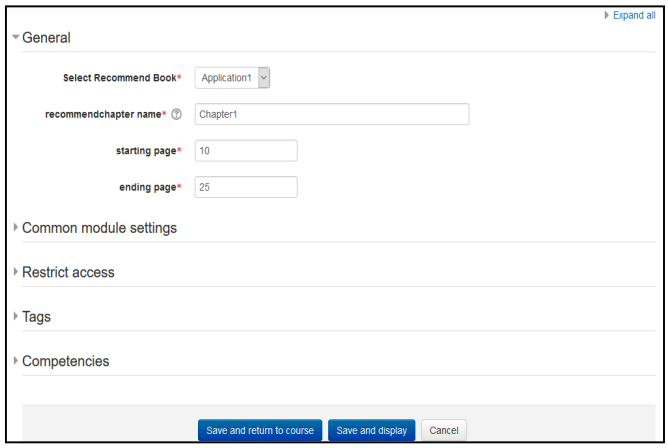

Fig. 8. Add Recommend Chapter Form Interface

Fig. 9. Mobile application main interface

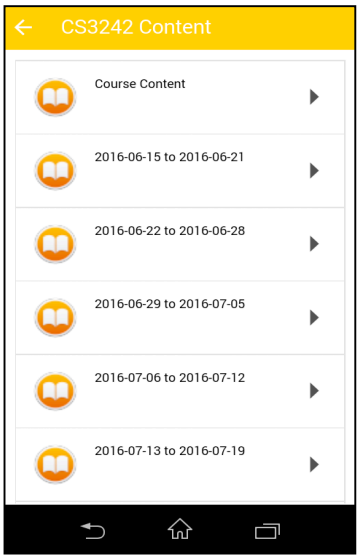

Fig. 10. Course content Interface

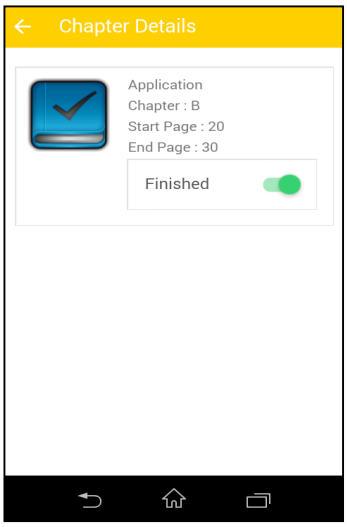

Fig. 11. Chapter detail interface

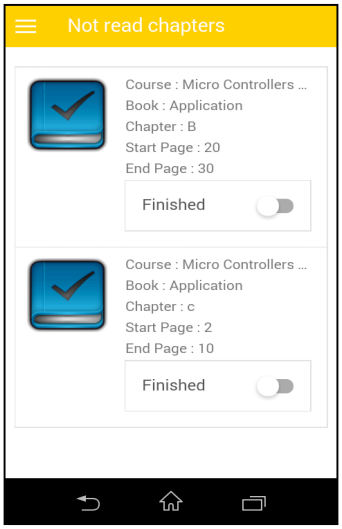

Fig. 12. Not finished chapters Interface

Figure 7 shows add recommend book/chapter resources in the standard resource adding interface of the course. Adding recommend book is a very easy procedure and only has to enter the recommend book name and author name to enter recommend book. Figure 8 shows the interface of the recommend chapter adding. When adding recommend chapter, user has to select a book added for the particular course. User can select the page range for the chapter and save it. Figure 9 shows the main interface of mobile application. All the courses student had registered, will be displayed in this main interface. To get the courses followed by the user to the main screen, first user has to sync with the Moodle server. Figure 10 shows the main interface of mobile applications. All week for the course will be displayed on this interface and user has to select one of those time period from it. Figure 11 shows chapter details interface. This interface will help user to check whether the particular chapter has been read or not. Users can mark the finished toggled button after reading the recommend chapter. Figure 12 shows not read chapter details interface. This interface will help user to identify what they have missed to read.

V. SYSTEM TESTING AND ANALYSIS

The testing of the application is carried out under several categories. They are Data and Database Integrity Testing, Function Testing, User Interface Testing and Configuration Testing. These categories would comprise both manual and automated testing appropriately.

PHPUnit is used to write Unit Test cases for dataset access methods of Moodle plugin. This verifies that data can be added to the database tables and verifies the data in data tables can be modified and deleted. Functional testing for the mobile application is done by karma, jasmine and angular-mocks. The main functions will be checked through functional testing (Mark Chapter as Read, Get Missed Chapters). Interface tests are done for the Moodle plugin generated views navigations and mobile application views navigations and functional requirements.

Configuration testing is done by installing the Moodle plugin on different versions of Moodle Servers and check those all versions install the plugin and work correctly and check all those versions connect with the mobile application and work correctly.

TABLE 1.
TEST RESULT, CONFIGURATIONS TESTING FOR MOODLE VERSION

Moodle Server Version	**Supported / Not Supported**
Versions Lower than Moodle 2.9	Not Supported
Moodle 2.9	Not Supported
Moodle 3.0	Supported
Versions Higher than Moodle 3.0	Supported

TABLE 2.
TEST RESULT, CONFIGURATIONS TESTING FOR ANDROID VERSION

Android version	**Supported/ Not Supported**
Versions lower than 4.0 Ice cream Sandwich	Not Supported
Version 4.0 Ice cream Sandwich	Not Supported
Version 4. 1 Jelly Bean	Supported
Versions higher than 4. 1 Jelly Bean	Supported

VI. CONCLUSION AND FUTURE WORK

"READ" Plugin can improve in several aspects. Currently, Recommend Book plugin only stores book name and its author. This can be improved by providing ability to add a soft-copy of the book to the Moodle. The mobile application shows only the recommended book name and chapter details. It can be improved by facilitating the students to read the chapter (from the soft-copy added) through the mobile application. Also Mobile application can be developed to add questions for the particular reading material. Then students will be able to provide answers for those questions and it will create a platform to discuss about the reading material.

REFERENCES

[1] "About Moodle - MoodleDocs", *Docs.moodle.org*, 2016. [Online]. Available: https://docs.moodle.org/31/en/About_Moodle. [Accessed: 18- Jun- 2016].

[2] "Plugin contribution - MoodleDocs", *Docs.moodle.org*, 2016. [Online]. Available: https://docs.moodle.org/dev/Plugin_contribution. [Accessed: 15- Oct- 2016].

[3] "MoodleDocs", *Docs.moodle.org*, 2016. [Online]. Available: https://docs.moodle.org/dev/Main_Page. [Accessed: 18- Jun- 2016].

[4] "Moodle plugins directory: Book", *Moodle.org*, 2016. [Online]. Available: https://moodle.org/plugins/mod_book. [Accessed: 15- Jun - 2016].

[5] "Moodle Mobile", *Download.moodle.org*, 2016. [Online]. Available: https://download.moodle.org/mobile/. [Accessed: 15- Oct- 2016].

[6] "Build Amazing Native Apps and Progressive Web Apps with Ionic Framework and Angular", *Ionicframework.com*, 2016. [Online]. Available: http://ionicframework.com/. [Accessed: 15- Oct- 2016].

[7] "litehelpers/Cordova-sqlite-storage", *GitHub*, 2016. [Online]. Available: https://github.com/litehelpers/Cordova-sqlite-storage. [Accessed: 18- Jun- 2016].

[8] "Ionic Documentation Overview - Ionic Framework", *Ionicframework.com*, 2016. [Online]. Available: http://ionicframework.com/docs/overview/. [Accessed: 15- Oct- 2016].

[9] "XMLDB editor - MoodleDocs", *Docs.moodle.org*, 2016. [Online]. Available: https://docs.moodle.org/dev/XMLDB_editor. [Accessed: 18- Jun- 2016].

[10] "Activity modules - MoodleDocs", *Docs.moodle.org*, 2016. [Online]. Available: https://docs.moodle.org/dev/Activity_modules#access.php. [Accessed: 18- Jun- 2016].