Mobile Integration for Sakai LMS and class schedule platform at an higher education institution

Pedro Filipe Oliveira
Research Centre in Digitalization and Intelligent Robotics (CeDRI)
Laboratório Associado para a Sustentabilidade e Tecnologia em Regiões de Montanha (SusTEC),
Instituto Politécnico de Bragança, Campus de Santa Apolónia, 5300-253 Bragança, Portugal
Email: poliveira@ipb.pt

Paulo Matos
Research Centre in Digitalization and Intelligent Robotics (CeDRI)
Laboratório Associado para a Sustentabilidade e Tecnologia em Regiões de Montanha (SusTEC),
Instituto Politécnico de Bragança, Campus de Santa Apolónia, 5300-253 Bragança, Portugal
Email: pmatos@ipb.pt

2023 International Conference on Engineering and Emerging Technologies (ICEET) | 979-8-3503-1692-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICEET60227.2023.10525775

***Abstract*—In the age of mobile technology, the demand for seamless access to educational resources has become essential. This work presents a project the integration of a Learning Management System (LMS) and a class schedule platform of a higher education institution, developing for that a mobile application aimed at integrating with the LMS used at the institution. The application seeks to enhance the learning experience by providing students, instructors, and administrators with anytime, anywhere access to the LMS core functionalities. The proposed mobile application will serve as a comprehensive extension to the LMS, extending its capabilities to mobile devices such as smartphones and tablets. The development process was done with continuous feedback from educators and students, with different testing and quality assurance measures implemented to ensure the security, reliability, and performance.**

***Index Terms*—mobile-application, learning-analytics, sakai, lms**

I. Introduction

Currently, the Leaning Management System (LMS) and summaries institutional platforms, of a higher education institution, are optimized for use through a computer or browser on the smartphone. The LMS platform is relevant for structuring teaching content and providing information between teachers and students and has the potential to provide personal organization tools for students [8] [1].

Since smartphones are devices with greater portability and daily presence in the personal lives of teachers and students, and there is a lack of adaptation of the platform for the optimized use of these devices, this work proposes an architecture to support a mobile application to combine these existing platforms [2]. Thus, the objective of this project is to develop a mobile application and all the needed back-end and web-services aimed at the institution's students, which allows access to the information and tools of the platforms, keeping a copy of this information offline for when internet access is not possible [3].

Leveraging the ubiquity and convenience of mobile devices, students will be empowered to engage in learning activities beyond the traditional classrooms. As principal features we identify the following: seamless access, users will be able to log in securely to the LMS from their mobile devices, accessing their course materials, assignments, discussion boards, and grades effortlessly; course notifications, push notifications will alert students to new announcements, upcoming assignments, and other important events reducing the risk of missed deadlines; content consumption, students will have the ability to access diverse learning materials, including documents, multimedia files, and interactive content, optimizing their learning experience on mobile devices and progress tracking, the mobile application will provide students with a comprehensive overview of their academic progress, including grades, attendance records, and completion status for assignments, empowering them to stay organized and motivated.

In this sense, it is intended to add academic management instruments, unite both platforms in a single interface and develop tools to enable the creation of a customized and individual schedules and calendar for the student, as well reminders with programmable alert notifications [4] [5].

II. Materials and methods

The institution's students and users of the referred platforms, note that the hustle and bustle of student life implies an increased effort to carry out all the tasks proposed by the teachers of each subject. It is frequent that some information is lost when it is only verbalized in the classroom context, implying that there is a need to clarify doubts with teachers about matters related to test/work deadlines and even schedules.

The LMS platform and summaries are optimized for desktop use [6] [7]. If you need to access both via a smartphone, in addition to having to use a browser, it will be essential to have an Internet connection. In the formulation and planning of this work, a list of problematic points was identified for

which we intend to present a proposed solution when using a smartphone:

- student cannot use the platforms as a tool for managing academic life, being subjugated only to the content shared by the teachers or external tools use;
- at the LMS platform, accessing the summary platform involves a series of steps, if this is not the method of accessing the summary, it implies, in the browser, opening a new tab to directly access the refereed platform;
- difficulty in consulting the school schedule. Each schedule, even if personal, contains excessive information that is not of primary relevance, such as the display of shifts that do not concern the student/user, as well time overlaps, permanently exposed, even if resolved;
- the LMS platform only formalizes assessment moments (tests) that are carried out/submitted on it. If there are moments of face-to-face evaluation, these are made known by the teacher to the students during the class and/or via e-mail or in the notices at the LMS platform;
- moments of continuous assessment, namely delivery of practical work, forms or others, are only part of a schedule agreed between professor and student, not appearing in the subject information on the platform.

In figure 1 we can see the graphical representation of the proposed architecture.

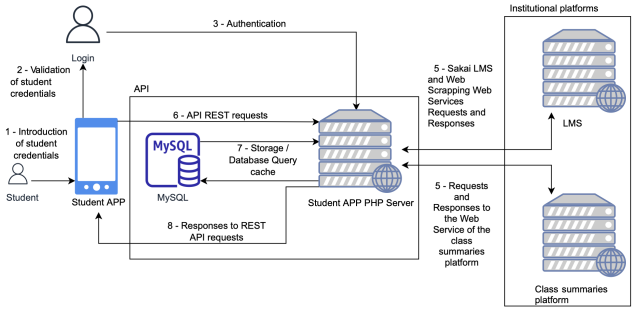


Fig. 1: System Architecture.

Following we detail this architecture functioning:

- student will enter their credentials from the institution's platforms in the application to perform authentication;
- credentials will be validated directly on the institution's server through *Sakai* web-services for data related to the LMS and the summary platform's web-services for accessing school timetable data;
- after a successful verification of credentials, the student will be authenticated;
- a *sessionId* for the mobile application will be returned. With this, the necessary authorization is obtained to make requests to the *Sakai* web-services available through our server;
- after, the mobile application will make *REST* requests to the implemented server, exchanging information with the database;
- the server will be responsible for storing and querying our *MySQL* database, which will contain the data acquired by requests to the *Sakai* web-services and *web scrapping* performed on the LMS platform, as well the summary platform web-services.;
- our server will send responses to *REST* requests to the mobile application.

III. Results

The following subsections detail all the developed work, and the mobile application functionalities.

A. Login

On this mobile application screen, the login form is displayed, in which the student is asked to enter their access credentials to the institution's platforms. A *REST* request is made to the "*index*" service of the developed *API* and if it is successfully authenticated, a request is made to the "*createSubjects*" and "*getInscription*" services, which return the subjects to which the student is enrolled.

In this login form, characters are checked in the "*Mechanographic Number*" field so that only the insertion of mechanographic numbers is possible (eg: a11111). This allows for greater security against threats to the system. After authentication, the entered data is saved in *AsyncStorage*, so that if the user closes the application and logs back in, authentication will be performed automatically. If the session is terminated, the data will be removed from *AsyncStorage* (Figure 2).

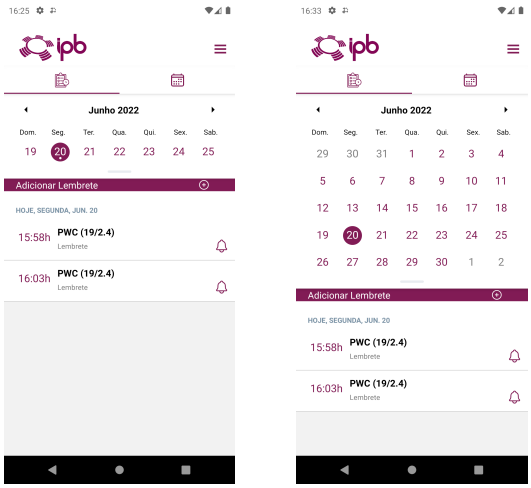
Fig. 2: Mobile Application - Login Page.

B. Schedule

After authentication, the first screen displayed is the *Schedule*. On this screen, a calendar and list of reminders near to the current date are displayed using the *React Native Calendars* component as a basis.

It is possible to create new reminders and consult the already created as well delete them if desired. To get the reminders, our *API* service "*getReminder*" is used. When creating new reminders, a *Modal* view is displayed with the form, which was developed using the *Formik* component that makes it possible to manage the filling in of fields, making it impossible

for the user to enter any data that is not allowed in the database, increasing the system security (Figure 3).



(a) Schedule page. (b) Expanded schedule.

Fig. 3: Schedule page and Expanded schedule.

C. Timetable

The second screen of the top tab *Timetable* shows the student's school timetable, which was developed based on *React Native Week View*. This presents in class blocks the data already formatted after being received through the request to the developed API service *fetchTimetable*.

Also exists the possibility of editing the color of each class block, as well its removal (changes are saved in the database through the *BlockClass*" services).

The functionality of customizing the timetable was also added, allowing the user to change the number of days displayed in the timetable (default is one day) as well the daily time interval. In this functionality there is also the option to load a new version of the timetable, this option makes a request to the *getTime* service.

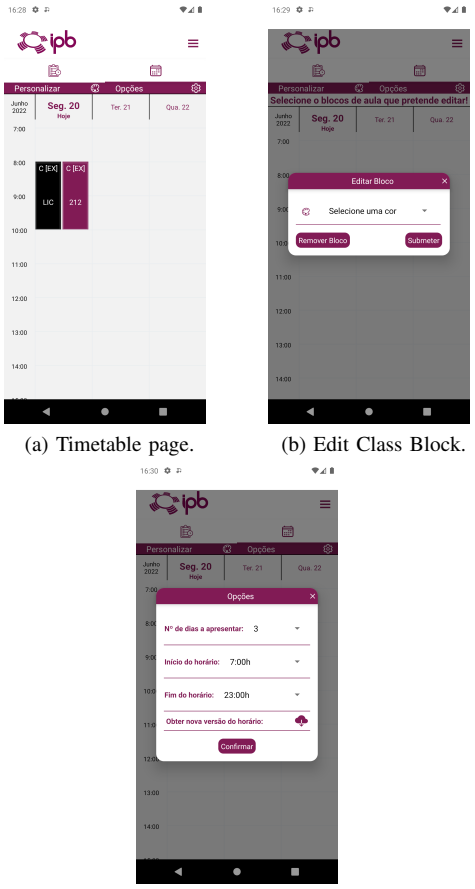
The new version of the timetable is not uploaded automatically, but by student choice. The reason for this is to allow students to keep their edited version of the timetable and only update to the latest one issued on the summary platform if they wish. It is also possible to consult the details of each class block presented (Figure 4).

D. Subjects

In the mobile application *Drawer* menu, the list of subjects to which the student is enrolled in the current academic year is displayed. Selecting the subject opens a new navigation with five new screens in the form of a *Top Tab* (Notices, Tests, Activities, Resources, Reminders) (Figure 5).

E. Announcements

By accessing the *Announcements* screen, the *getAnnouncement* service is called, which returns the announcements of the chosen subject that already exist in the database.



(a) Timetable page. (b) Edit Class Block.

(c) Timetable Options.

Fig. 4: Schedule Page, Edit Class Block and Schedule Options.

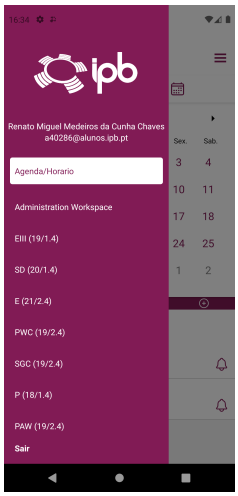
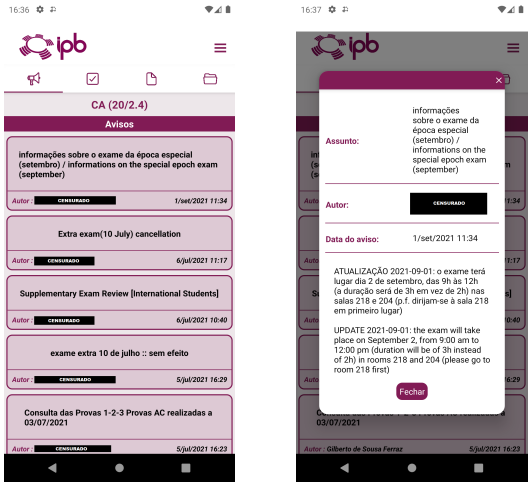


Fig. 5: Mobile Application - Navigation Drawer.

If there are still no announcements for that subject in the database or if they already exist, but the student performs an update on the screen, the *fetchAnnouncement* service is called, which performs the *Web Scrapping* and subsequently populates the announcements table in the database. Given that

the announcements belong to a subject and not a student, any update to the table is reflected in all students who share the subject. After obtaining the announcements, they are displayed in a *flatlist*.

Web Scrapping is not carried out whenever the *Announcements* screen is opened for reasons of application performance and fluidity. If *Web Scrapping* were performed every time the user entered the "Announcements" screen, the loading of information would be unnecessarily slower. Each *flatlist* item is clickable and allows access to the details of each announcement (Figure 6).



(a) Announcements Page. (b) Announcements Details.

Fig. 6: Mobile Application - Announcements and Announcements Details Page.

F. Online Tests

By accessing the "*Tests*" screen, the *getTests* web-service is called, which returns the tests of the chosen subject that already exist in the database.

If there are still no tests for that subject in the database or if they already exist, but the student updates the screen, the *createTests* web-service is called, which performs the *Web Scrapping* and subsequently populates the tests table in the database.

After obtaining the tests, they are presented in a *flatlist*. At the same way, *Web Scrapping* is not performed whenever the *Tests* screen is opened for reasons of application performance and fluidity. If *Web Scrapping* were performed every time the user entered the *Tests* screen, the loading of information would be unnecessarily slower (Figure 7).

G. Activities

By accessing the *Activities* screen, the *getActivities* web-service is called, which returns the activities of the chosen subject that already exist in the database.

If there are still no activities for that subject in the database or if they already exist, but the student performs an update on the screen, the *createActivities* web-service is called, which performs the *Web Scrapping* and subsequently populates the activities table in the database. Since activities belong to a subject and not a student, any update to the table is reflected in all students who share the subject.

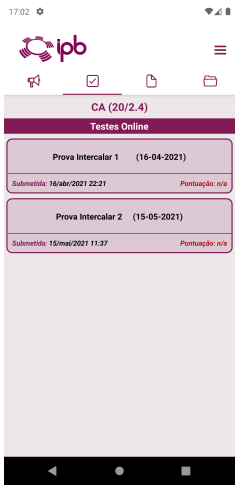


Fig. 7: Mobile Application - Tests Page.

After obtaining the activities, they are presented in a *flatlist*. *Web Scrapping* is not carried out whenever the *Activities* screen is opened for reasons of application performance and fluidity (Figure 8).

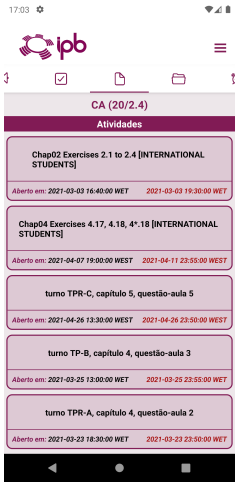


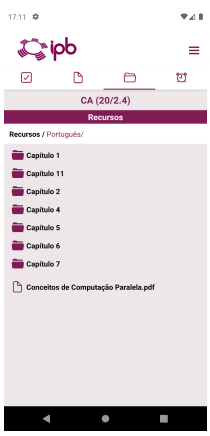
Fig. 8: Mobile Application - Activities Page.

H. Resources

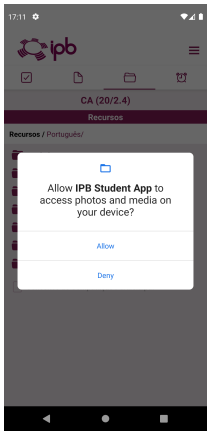
By accessing the *Resources* screen, the *getResources* web-service is called, which returns a *JSON* with the resources of the chosen subject that already exist on the server. With this *JSON* we developed an algorithm for conditional rendering of components consisting of several arrays, which are responsible for saving the current and previous path depending on the navigation made by the existing folders.

The algorithm distinguishes files and folders according to their extension, allowing the transfer to the mobile device in

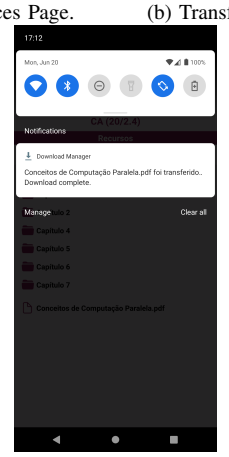
the case of a file, in order to simulate a file management system. To transfer the files, we use the *React Native Fetch Blob* component and ask the student for write permissions on the smartphone's storage (Figure 9).



(a) Resources Page.



(b) Transfer Permission.



(c) Transfer Notification.

Fig. 9: Mobile Application - Resources Page, Download Permission and Download Notification.

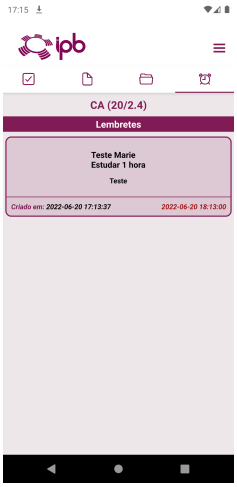
I. Reminders

Opening the *Reminders* screen, the *getReminders* web-service is called, which returns the reminders created by the student for the specified subject and presents them in a *flatlist*. It is possible to open the reminder in order to consult its complete content and also delete the reminder if desired.

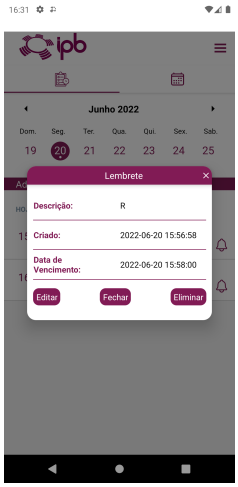
When creating or editing reminders, the user will have the option of creating an alert for the reminder. At this option, student will be able to choose its repetition and anticipation. Notifications are displayed whether the user has the application open, minimized or even with the application closed. Also a button is shown that allows the notification termination (Figure 10).

IV. Conclusions

With this work, we develop an *API*, which underwent a thorough verification of the available *SAKAI* web-services,



(a) Discipline Reminders Page.



(b) Reminder Details.

Fig. 10: Mobile Application - Reminders Page and Create or Edit Reminder without Alert.

from which we prepared a list of web-services and their respective responses. We verified the lack of some essential services to achieve the proposed objectives, and as such, we made a comparison between two possible solutions.

One solution would be to create the *SAKAI* services we needed and the other would be *Web Scrapping*. After analyzed the two solutions, we decided that the best way would be to carry out *Web Scrapping* on the institution LMS platform.

After finishing the *API* design and mobile application, we were able to develop an *API* capable of obtaining and processing the information present on the institution online platforms and make it available through *REST* services.

This *API* enabled the development of a fast, intuitive and extremely useful mobile application for a student at the institution. This application has features such as a personal schedule, school timetable with the possibility of editing each class block individually, reminders with programmable alerts as well access to specific information for each subject area provided by the institution services.

Although we have developed an extremely useful and ready-to-use mobile application, we assume that there is always something to improve or something new to develop.

In future work, we would propose the development of missing *SAKAI* services to obtain more complete data and improve the *API* performance.

The mobile application can also be used offline when the institution platforms were not available or when access to the internet was not possible, as well having an automatic data synchronization system so that it was only necessary to consult them directly in the database, and the student was notified of the latest updates in their different subjects.

Acknowledgment

The authors are grateful to the Foundation for Science and Technology (FCT, Portugal) for financial support

through national funds FCT/MCTES (PIDDAC) to CeDRI (UIDB/05757/2020 and UIDP/05757/2020) and SusTEC (LA/P/0007/2021).

REFERENCES

[1] F. Huertas and A. Navarro, "Web services availability in e-learning platforms," in *2011 7th International Conference on Next Generation Web Services Practices*. IEEE, 2011, pp. 170–175.

[2] N. N. M. Kasim and F. Khalid, "Choosing the right learning management system (lms) for the higher education institution context: A systematic review." *International Journal of Emerging Technologies in Learning*, vol. 11, no. 6, 2016.

[3] F. Ozdamli and N. Cavus, "Basic elements and characteristics of mobile learning," *Procedia-Social and Behavioral Sciences*, vol. 28, pp. 937–942, 2011.

[4] T. Majchrzak and T.-M. Grønli, "Comprehensive analysis of innovative cross-platform app development frameworks," 2017.

[5] C. A. Talib, H. Aliyu, A. Malik, A. Maimun, K. H. Siang, I. Novopashenny, and M. Ali, "Sakai: A mobile learning platform." *International Journal of Interactive Mobile Technologies*, vol. 13, no. 11, 2019.

[6] G. Ssekakubo, H. Suleman, and G. Marsden, "Designing mobile lms interfaces: learners' expectations and experiences," *Interactive Technology and Smart Education*, vol. 10, no. 2, pp. 147–167, 2013.

[7] M. A. Forment, M. J. C. Guerrero, and J. P. Poch, "Towards mobile learning applications integration with learning management systems," in *Virtual Learning Environments: Concepts, Methodologies, Tools and Applications*. IGI Global, 2012, pp. 726–738.

[8] M. O'Brien, D. Slattery, and P. Hyland, "Sakai device and platform usage: A four-year campus-wide analysis," in *ECEL 2018 17th European Conference on e-Learning*. Academic Conferences and publishing limited, 2018, p. 438.

[9] H. Wan, Q. Yu, J. Ding, and K. Liu, "Students' behavior analysis under the sakai lms," in *2017 IEEE 6th International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*. IEEE, 2017, pp. 250–255.

[10] A. A. Piña, "An overview of learning management systems," *Virtual Learning Environments: Concepts, methodologies, tools and applications*, pp. 33–51, 2012.

[11] M. J. Casany, M. Alier, and E. Mayol, "Application of mobile and software as a service (saas) in learning processes," in *Proceedings of the First International Conference on Technological Ecosystem for Enhancing Multiculturality*, 2013, pp. 207–208.

[12] M. R. Elabnody, "A survey of top 10 open source learning management systems," *International Journal of Scientific & Technology Research*, vol. 4, no. 8, pp. 7–11, 2015.