A Mobile App Design Model with Offline Support for the Botswana Open University Student Management System

F. Kaniwa and M. Phuthego
Botswana Open University
fkaniwa@gmail.com

Abstract

Botswana is a developing country in Southern Africa with approximately 64% of the population in urban areas. The other remaining population suffers poor services in remote regions such as unstable power supply, internet connectivity and other services. Botswana Open University (BOU) is an Open and Distance Learning (ODL) institution which aims at improving access to education especially to marginalised regions. As an effort to mitigate such challenges, we propose a Mobile App Model with Offline Support. The design model will enable development of a mobile app with offline support to be used in the marginalised regions of Botswana for the BOU Student Management System.

1. Introduction

Botswana is a developing country with approximately 64% of the population in urban areas (Botswana, 2011). The other remaining population (36%) suffers poor services in remote regions such as power cuts, internet connectivity and other services. As a developing country, the country faces challenges of low adult literacy, low infrastructure development in rural areas and the sparse populations across the country brings a challenge of accessibility to Information and Communication Technologies (ICT) (Technology, 2007). Botswana Open University (BOU) is an Open and Distance Learning (ODL) public institution in Botswana which aims at improving access to education especially to marginalised regions. Smart phones particularly lower-cost android phones with reasonable specifications have become ubiquitous because of their affordability. Generally internet connectivity in developing countries is still not as smooth as in the developed world. Fig 1 illustrate the world internet usage over the years (Sanou, 2017). Africa remains the highest growth rate on Mobile internet usage on the global level. Hence there is an increasing need for app designers to take advantage of that and to design apps that give all customers almost an equal responsive experience with their app despite the region. Therefore the core requirement for every app is to enable users to have a smooth and responsive user experience even in areas where internet connectivity is a challenge.

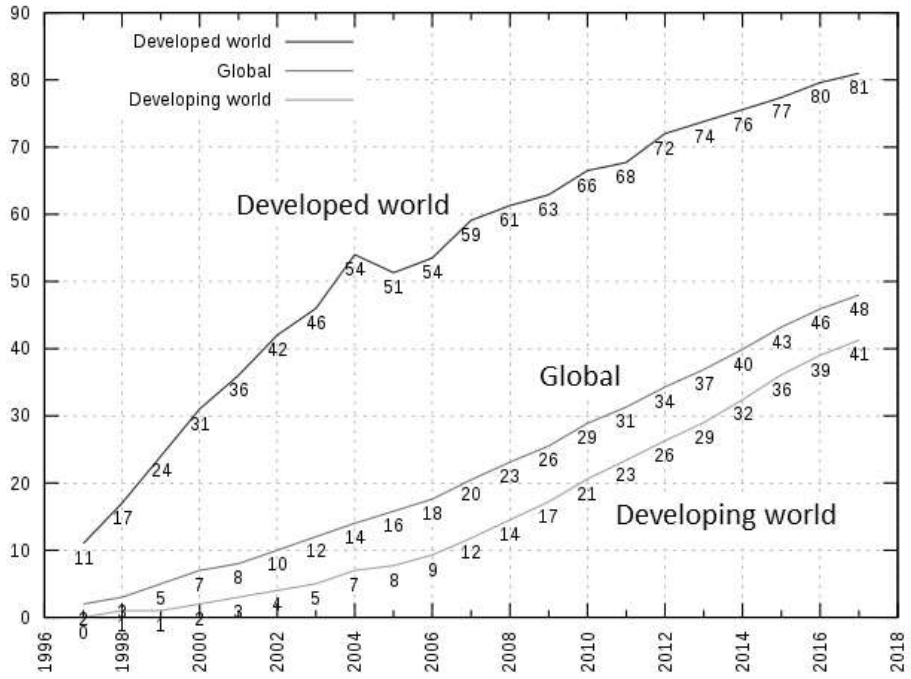

Figure 1 Internet User per 100 inhabitants

Consequently mobile app development is on the rise as well as the demand for quality and responsive user experience. Responsive app experience has increasingly become important in areas with internet connection challenges. As an effort to mitigate such challenges, we propose the Mobile App Design Model with Offline Support.

BOU has an online student management system called Integrated Tertiary System (ITS). The system is currently used for student applications, registration and recording of assessment marks. Learners use ITS for application, registration and to check their results after been published. In 2009, Mascom Wireless (Botswana Mobile Network Provider) entered into a Private Public Partnership (PPP) with the then Ministry of Communications Science and Technology, to roll out communication services to rural communities, under the Rural Telecommunications Development Programme, which resulted in Kitsong Centres. Kitsong is a Setswana brandname for these centres meaning "knowledge centres". Kitsong centres are centres around the country used for internet access. This was part of government initiative to bring internet connectivity to remote areas.

The proposed mobile learning app design model is meant to increase access to this student management system ,(ITS) and make information available on mobile devices in marginalised regions with internet connectivity challenges. BOU uses Moodle as a Learning Management System (LMS) and currently has a mobile app with offline support however ITS does not have any mobile app or offline support. In this paper we propose a design model for the mobile app with offline support. The proposed design is intended to be used as a guide to build an 'offline ready' mobile app.

2. Background

There are basically two ways of synchronising data, which are, automatic and manual. Automatic synchronisation involves synchronisation of a process that generates offline data when internet connection is available. Manual synchronisation involves generating data offline and manually triggering the app or use availed option to synchronise the data. Synchronisation process can be timed to run every certain number of minutes to save data. Unified Modelling Language (UML) is a standard language used for providing a way to visualise a system and uses mostly graphical notations (Booch, 2005). UML diagrams are used to show behaviour and structure of the proposed software system. Activity diagram is one such diagram under UML diagrams used to represent a control flow from one operation to another (Mall, 2018). An activity diagram is used to model the dynamic aspects of the system. Swimlane activity diagram is a variant of the standard activity diagram with more information about which role is performing which activity. An example of a swimlane activity diagram is shown in Fig 2.

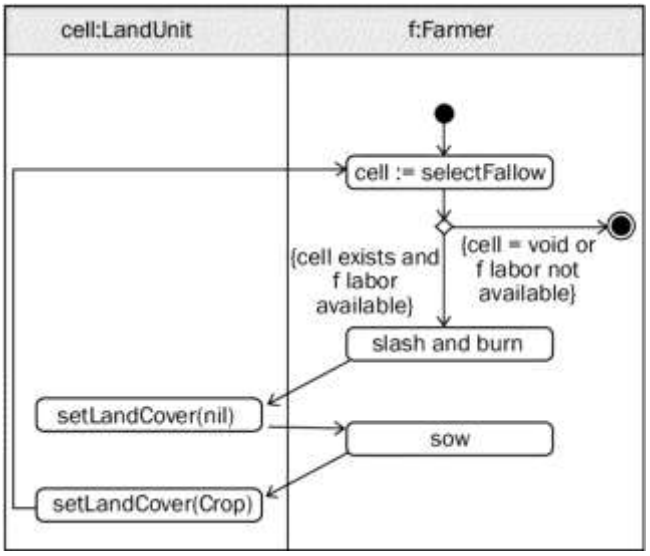

Figure 2 Swimlane activity diagram

A software architecture is an abstraction, metaphor or blueprint of a software system (Bass, Clements, & Kazman, 2003; Perry & Wolf, 1992). It shows how the system will be built. The major advantage of a software architecture is that of providing the ability to verify the proposed software to meet its specifications. We use the same instrument for our proposed design model. There are various architectural approaches used, which includes blackboard, data-centric, broker, client-server, service-oriented, publish-subscribe, monolithic application based, plug-ins, multi-tenancy architecture and so on (Sharma, Kumar, & Agarwal, 2015). All these

approaches are subject to attributes such as complexity, reliability, scalability, functionality, efficiency, maintainability and usability. Other common architectures are the Model-View-Controller (MVC) and Model-View-Presenter (MVP). MVC is used mostly for desktop applications due to its advantages of de-coupling components and high cohesion leading to easy code re-use and parallel development (Reenskaug & Coplien, 2009). The MVC breaks an application into Model (data), View (interface to view and modify model data) and Controller (operations that can be performed on the model data). MVC is mostly used for web applications and mobile apps (Qureshi & Sabir, 2014). The controller in MVC architecture has more control on the user interface than in MVP, the presenter in MVP is passive and only presents the information through the user interface.

User interface design is one important aspect included in the design model. Interface design has increasingly become pivotal in an app's usability and functionality. There are various techniques used in user interface design. One such approach is called wireframing (Cooper, Reimann, Cronin, & Noessel, 2014). A wireframe design is two-dimensional illustration of a system's interface focusing on space allocation, content, available functionalities and expected behaviours of the system. The advantages of wireframing can be summarised in Fig 3 (Faranello, 2012). Interface design principles are a set of guidelines used when designing usable interfaces. They are abstractions intended to orient interface designers to think about different aspects of the design (Sharp, 2015). These include; Visibility, Feedback, Constraints, Consistency and Affordance (Sharp, 2015).

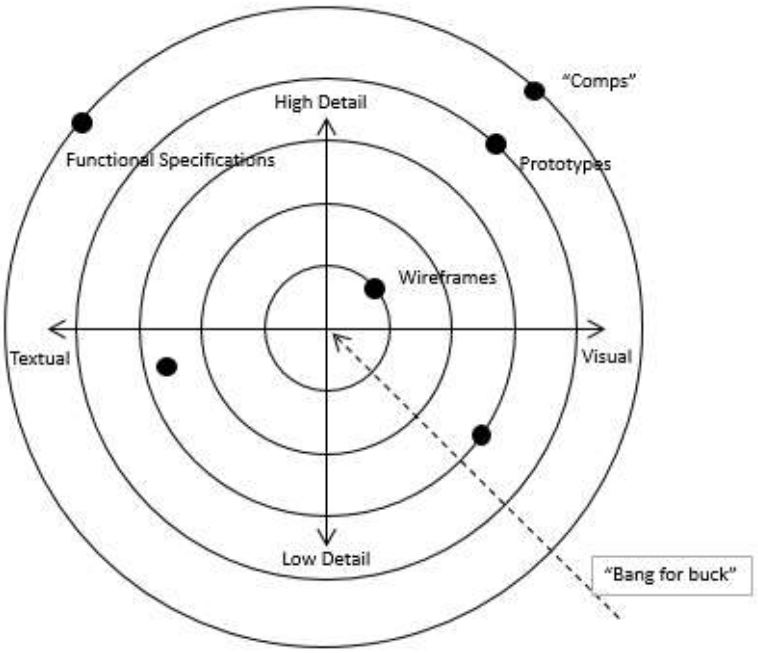

Figure 3 Wireframe advantages

3. Related work

In this section we provide brief related literature to our study. There are increasingly a number of apps being designed with offline support to improve responsive user experience. Moodle is a free and open-source learning management system accessible through a web application and a mobile app. The Mobile App version 3.1.3 (Moodle, 2016) introduced an offline feature for its mobile app. The feature allows users to work without internet connection. Students can download courseware and other resources for later viewing. Synchronisation happens when the user reconnects to the internet. Google offers a free e-mail service called Gmail which has an offline feature that allows one to access Gmail when the user is offline (Dias, 2009). This functionality is also available on Google Chrome browser or its HTML powered Gmail mobile app. It allows users to view their emails where there is no internet connectivity. For instance, a user is able to catch up with all the emails on a plane where there is no internet connectivity. The feature further allows users to reply to emails offline and the emails will be automatically sent when internet becomes available. The offline feature works as shown in Fig 4. Google has rolled out the offline feature to most of its apps including Google Maps. Google Maps app allows the user to download the entire map of the area they want to visit and make it available for navigation in offline mode where there is no internet connection. Google apps further allows the user to determine where the map file can be saved, that is, internal storage or on external memory such as SD Card.

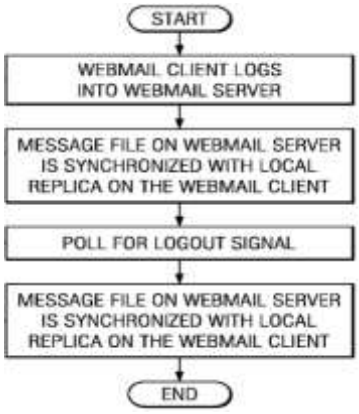

Figure 4 Gmail off-line feature

Trello is an app tool for managing projects tasks and personal tasks. It has offline support which makes use of the client-server architecture with a thin web layer that downloads the main app (Trello).

4. Design Model

In this section we outline our design model using a UML diagram called Swimlane Activity Diagram. We make use of the MVP architectural pattern for the software architecture of the proposed mobile app. We further show our wireframe design model for the user interface. The learner will be able to apply, register and view results when internet connection is not available. Therefore when they are offline they are able to use the app. The offline app will synchronize the data with a central repository when the internet connection becomes available.

ITS is a Web based student management information system used by BOU with no support for mobile and offline access to the best of our knowledge. We outline the design models to design a mobile app with offline support for the BOU ITS system. Modelling techniques used help the developer to follow a suitable architecture according to their application specifications. The proposed app should use offline-first approach to ensure a responsive user experience. The approach works by checking the local storage to start up the app before checking for internet connectivity which is likely to cause a delay (or 'freezing') if internet is not available. For instance, the login page is first fetched from the local storage before checking for internet availability. The app will use automatic synchronisation to sync data with the server. The local disk is more reliable than the network and the ability to cache and reuse previously fetched resources is a critical aspect of optimizing for performance.

A. Model View Presenter Architecture

The MVP design pattern is proposed for the mobile app design because on the envisioned app the presenter directly interact with the view and therefore easy to supply information for a particular model contrary to the MVC architecture where the model is linked directly to the view. Furthermore, the proposed mobile app will not have any business logic in the view as in MVC design pattern. The MVP architecture guarantees an app which is easier to maintain and adapt to new requirements. The app will work with local storage and then sync globally. The Model will be persistent, if there is new data from server, the persistent model is updated and Presenter will always check with the Model first before updating the View. The proposed MVP for the mobile app is shown in Fig 5.

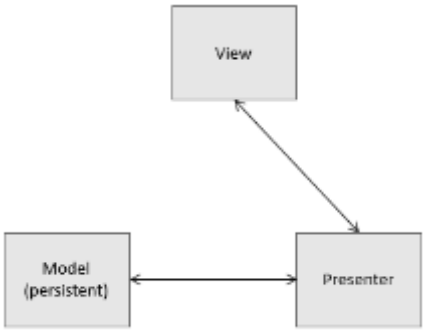

Figure 5 Model-View-Presenter

B. Swimlane Activity Diagram

This UML diagram shows the process flow between the client side (app side) and the server-side (ITS server) where the data resides of the proposed app. The swimlane feature of the activity diagram is used to show an integrated process map therefore providing clarity of the process map on roles versus processes making it easy to implement. The proposed swimlane activity diagrams are shown in Figures 6, 7 and 8. Client represents the app side (mobile phone) and the server represents the ITS server side.

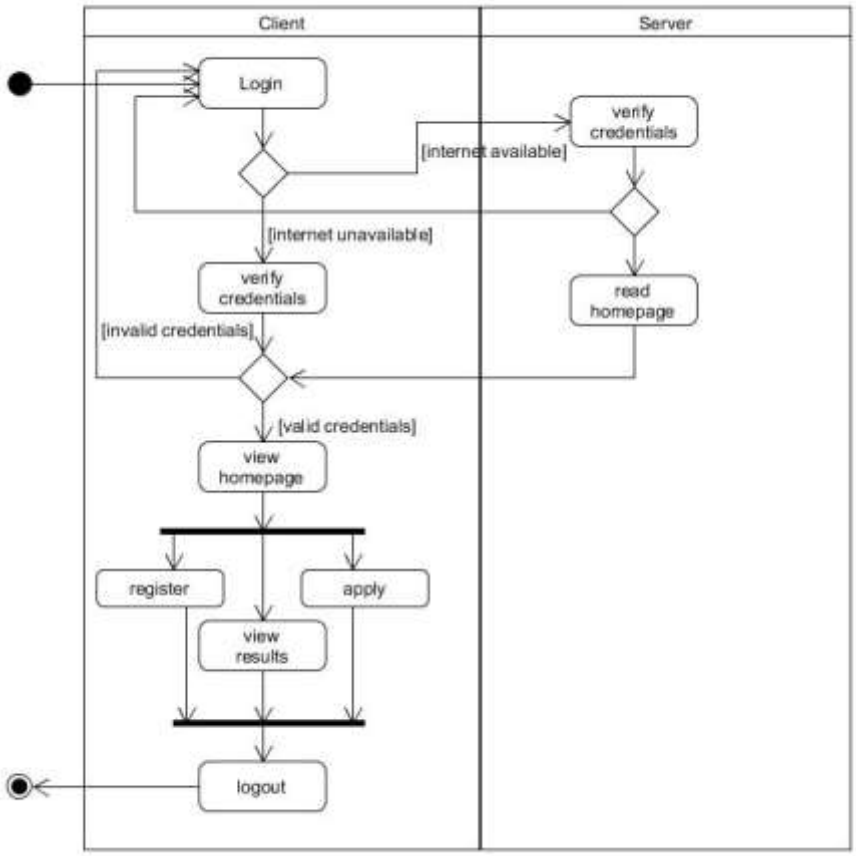

Figure 6 Login Swimlane Activity Diagram

Fig 6 shows a swimlane activity diagram for the login process. The processes can either occur on the clientside (app side) or the server side (ITS server) as shown by the two swimlanes. The login can occur when internet connectivity is either unavailable or available. The internet connection is checked before determining if the connection is local or remote. In case of internet connectivity being unavailable, the local cached copy is retrieved. If internet connectivity is avaible during the login process then the authentication is done with the server and the most recent homepage is fetched from the ITS server.

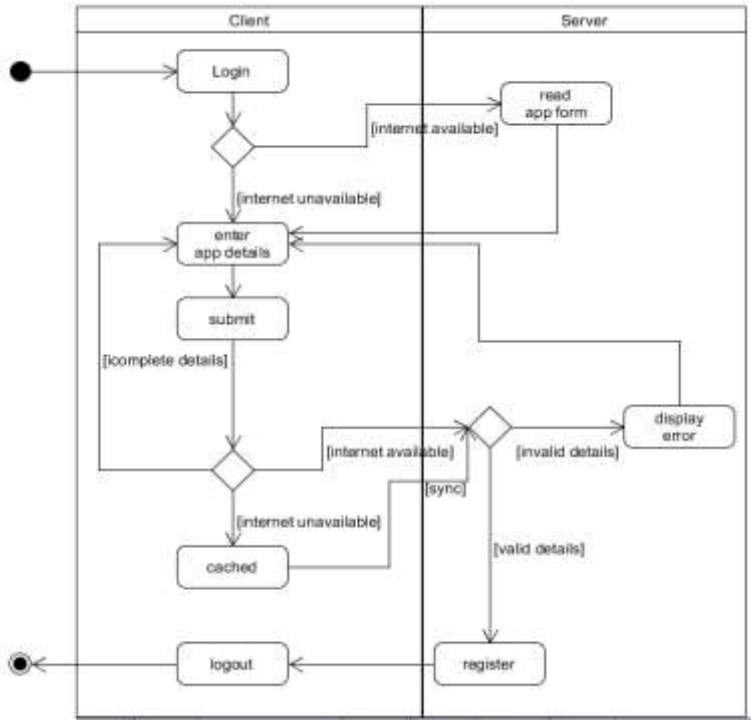

Figure 7 Application Swimlane Activity Diagram

Fig 7 shows the swimlane activity diagram for the application process. The login occurs as described in Fig 6. The application form is fetched from the local cache or server depending on the internet availability. After the application details are entered, the details are checked for validity and completeness. If the submitted information is not complete, for instance, payment details, then the student is alerted to re-submit and these operations are carried out on the mobile device. In case the internet is unavailable then the details are cached and will be synchronised when internet becomes available and then registration will be completed.

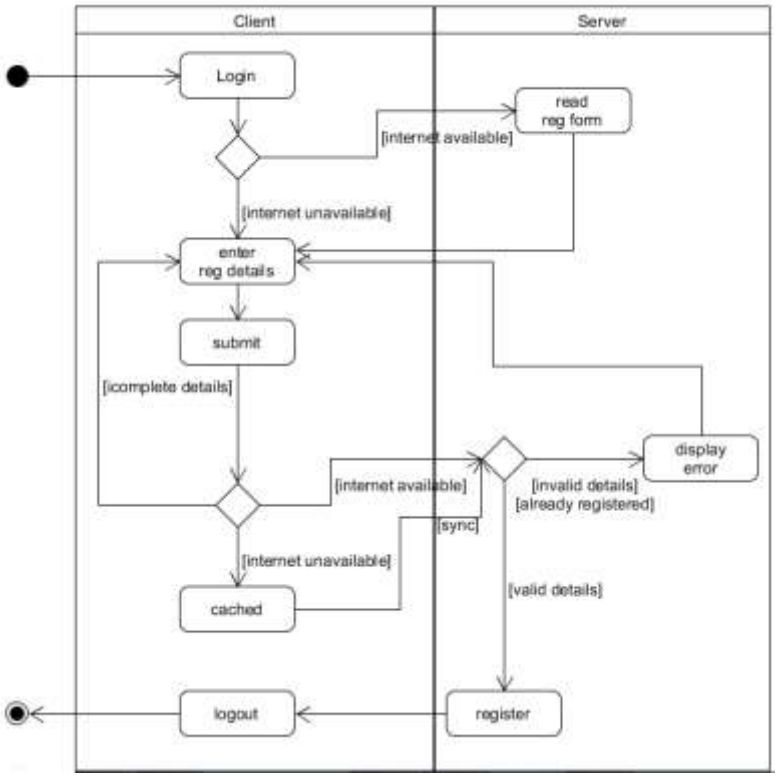

Figure 8 Registration Swimlane Activity Diagram

Fig 8 shows the swimlane activity diagram for the student registration process. The process takes a similar route as the one for application process (Fig 7), except the part the system checks if the user is already registered and generates an error with a redirection for the student to re-enter the correct details on the registration form.

C. Wireframe

Wireframing is the most common and effective way of designing user interfaces. A wireframe provides a blue print of the expected user interface. User interfaces have become increasingly important especially in mobile apps hence the relevance to show its design (Faranello, 2012). The app will use a Graphical User Interface (GUI) interface type which is easy to interact with. The design of our interfaces is based on the design principles by (Sharp, 2015). The target audience for this app is mostly adults hence our design will have that specific focus.

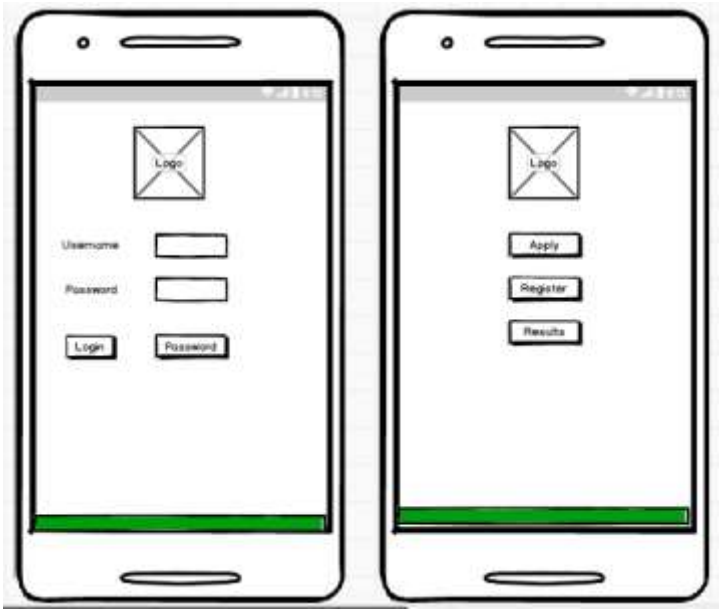

Figure 9 Login and Homepage Wireframes

Fig 9 shows screen designs for the login screen and the start page (home page) respectively. The user can log in or recover their password through the "password" button. The bottom green bar shows the status of the internet connectivity. Green indicates that internet is available, red indicates that internet is not available and amber colour indicates that there is no internet however there is data ready (cached) to be synced when the connection becomes available. Once the data is synced, the color coded horizontal bar (color bar) becomes green.

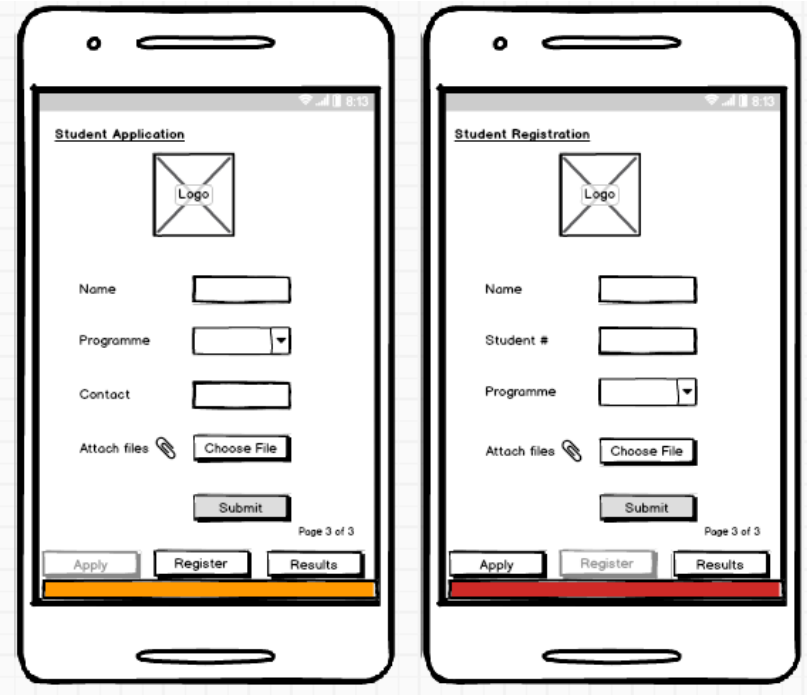

Figure 10 Student Application and Registration Wireframes

Fig 10 shows wireframes of student application and registration design screens respectively. The design layouts are similar therefore consistent and shows standard menu at the bottom of the screen for easy access to other screens.

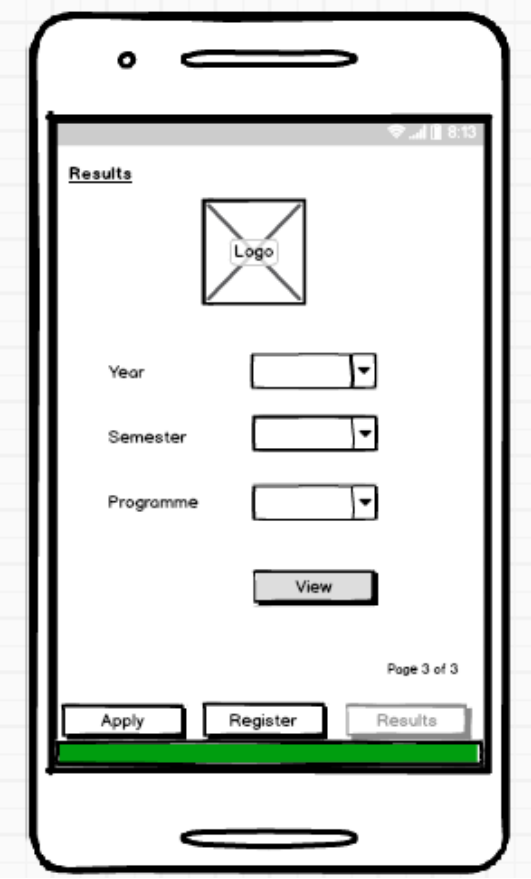

Figure 11 View Results Wireframe

Fig 11 shows the design screen used to view the results. Students are able to specify their details such as year semester and so on in-order to view their published results.

D. Design Principles

The following section shows how the interface designs catered for the design principles specified by (Sharp, 2015).

i) Visibility

The principle emphasises on making the functions of the system visible so that users are more likely to know what to do next such as controls in car like indicators, horn, steering wheel and so on (Norman, 1983). The functions of the app designs are clearly visible on the homepage and every other page (at the bottom screen).

ii) Feedback

This involves sending back the information to the action performed and what has been achieved. Feedback can be in the form of audio, verbal, visual or combination of these. The principle is demonstrated by the navigation from one screen to another, for instance after clicking the 'apply' menu option, the application form shows up. The color coded horizontal bar on every screen also show the presence or absence of internet connectivity. Red bar indicates no internet connectivity, amber bar indicates cached and ready to be synced and green represents that internet connection is available. Furthermore, page numbering on each screen is shown to show progress of student application or registration process as well as feedback of current page.

iii) Constraints

This involves restrictions to functions that can be performed at a given moment, for instance, deactivating some features of the screen which do not apply. Bottom menus are deactivated showing functions which do not apply and thereby reducing user errors. Provision of dropdown boxes is made to constraint users of valid available options and also reducing possibilities of user errors.

iv) Consistency

This refers to a design with similar operations and similar elements. In our wireframe designs, a default font for all interfaces should be specified. Interface designs are consistent on all screens as well as their operations to make users easy to remember. Interface layouts are consistent, logo placement, colors, color bar, and so on.

v) Affordance

This refers to intuitive design, or feature of an object that allows people to know how to use it (it gives a clue). For instance a mouse button invites a push. Our design has standard graphical elements such as buttons which invites pushing therefore making it obvious. Deactivation of menu not applicable at current screens suggests functions that are unavailable.

Trade-offs can always be done when applying the design principles that is, prioritising one over another. For instance, when a designer tries to constrain an interface, the less visible the interface will be. The ideology behind the proposed design is simplicity to make the app easy to use since the target audience is for the adults.

5. Conclusion

The paper proposed a design model to be used as a guide for the development of the Botswana Open University mobile app for its learners to access its Student management system. The design model caters for the offline support of the envisioned app given the fact that most of its learners reside in areas where internet connectivity is unreliable. Model-View-Presenter design pattern, Swimlane activity diagram and wireframing were used for software architecture, system design and interface design respectively. This design model can also be used as guide to any other app development wishing to have offline support. Future work includes the next phase which will be the development of the mobile app and its success will be tested and evaluated.

References

Bass, L., Clements, P., & Kazman, R. (2003). *Software architecture in practice*: Addison-Wesley Professional.

Booch, G. (2005). *The unified modeling language user guide*: Pearson Education India.

Botswana, S. (2011). Population and Housing Census, 2011. *Gaborone: Statistics Botswana*.

Cooper, A., Reimann, R., Cronin, D., & Noessel, C. (2014). *About face: the essentials of interaction design*: John Wiley & Sons.

Dias, E. W. B. (2009). Method of accessing web e-mail off-line: Google Patents.

Faranello, S. (2012). *Balsamiq wireframes quickstart guide*: Packt Publishing Ltd.

Mall, R. (2018). *Fundamentals of software engineering*: PHI Learning Pvt. Ltd.

Moodle. (2016). Online learning that you can access offline? Welcome to Moodle Mobile 3.1.3. Retrieved 03/06/2019, 2019, from https://moodle.com/news/online-learning-can-access-offline-welcome-moodle-mobile-3-1-3/?highlight=offline

Norman, D. A. (1983). Some observations on mental models. Mental Models. *Lawrence Erlbaum:*, 99 citation_lastpage= 129.

Perry, D. E., & Wolf, A. L. (1992). Foundations for the study of software architecture. *ACM SIGSOFT Software engineering notes, 17*(4), 40-52.

Qureshi, M., & Sabir, F. (2014). A comparison of model view controller and model view presenter. *arXiv preprint arXiv:1408.5786*.

Reenskaug, T., & Coplien, J. O. (2009). The DCI architecture: A new vision of object-oriented programming. *artima developer*.

Sanou, B. (2017). ICT facts and figures 2017. *International telecommunications union. Geneva*.

Sharma, A., Kumar, M., & Agarwal, S. (2015). A complete survey on software architectural styles and patterns. *Procedia Computer Science, 70*, 16-28.

Sharp, H. R., Yvonne & Preece, Jenny. (2015). *Interaction design* (Fourth ed.): John Wiley & Sons.

Technology, M. o. C. S. a. (2007). *Maitlamo National ICT Policy*. Gaborone, Botswana.

Trello. What is Trello? , 2019, from https://trello.com/about