**MAKERERE UNIVERSITY**

**COLLEGE OF HEALTH SCIENCES/COMPUTING**

**SCHOOL OF PUBLIC HEALTH/SCHOOL OF INFORMATION TECHNOLOGY**

**Program: Master of Health Informatics**

**Course: Advanced programming**

**Course code: MHI 8101**

|  |  |
| --- | --- |
| **Name** | **Registration number** |
| **Mpiima Jamiru** | **2016/HD07/2067U** |
| **Atukunda Patience** | **2016/HD07/2054U** |
| **Irene Wanyana** | **2016/HD07/2079U** |

**Assignment Submitted to: Professor Engineer Bainomugusisha**

**Date: 14th September 2017**

**Mama care**

**Introduction**

This is a hybrid mobile platform consisting of a mobile app that enables health workers monitor the health of expectant mothers and as well as provide access to relevant maternal and child health information plus medical expertise, all at the touch of a button. The solution helps improve primary health care service delivery, strengthen accountability, and generate real-time data for improved monitoring and evaluation. It has a login function for health workers so that no unauthorized access if information. The app also enables the health worker register expectant mothers, search for a mother and update the health card of the mother. The app also has a health worker community where health workers can discuss hard to manage issues instantly. Our target beneficiaries are healthcare professionals and expectant mothers and other women in the reproductive age (15-45 years).

**What are the goals of Mamacare?**

* MamaCare is a mobile health tool that enables health workers do three basic things:
* Monitor the health of mothers and that of their children from pregnancy, through child birth.
* Provide access to timely, relevant and personalized maternal and child health information.
* Provide access to medical expertise in real-time

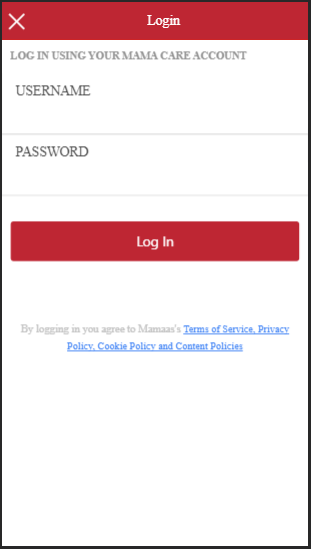
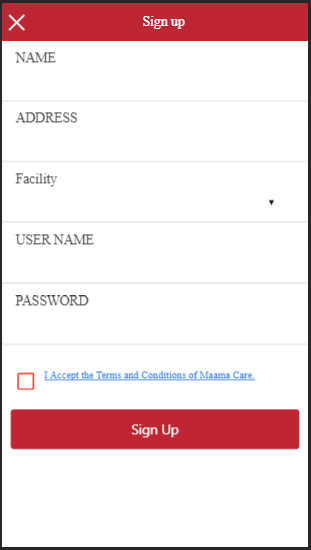


**Key program components**

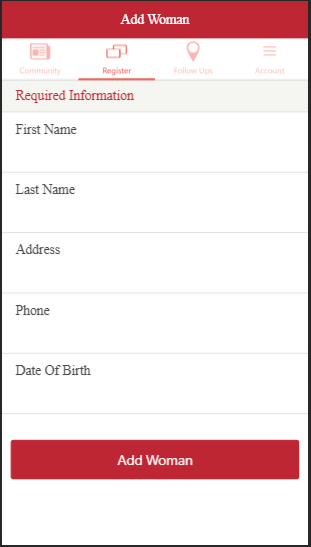
The mobile platform offers the following services: Immunization Tracking, Growth Monitoring, Oral Rehydration Therapy, Breast Feeding, Female Education, Family Planning, Mother's Community. Other features of the application include: SMS Alerts, Calendar synchronization and alerts, Automatic Vaccine Tracker, Medical Suggestions and Solutions, and a Chat feature to access the "Mothers' Community."

**The app has the following components.**

**Log-in function**: This enables the health workers to sign up and create accounts. The health worker enters their bio data into the app.

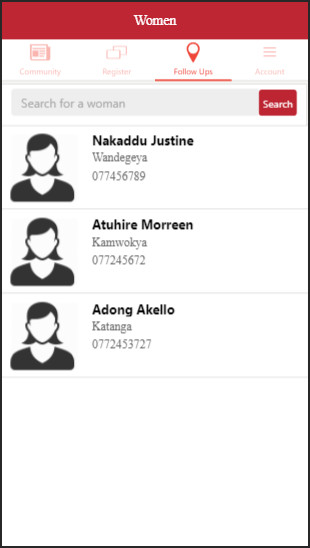
 

**Bio Data of the mother:** Every mother is unique, and as such the app allows health workers to register mothers, their bio data.

****

**Health workers' Community:** Join a number of mothers in your community, sharing awesome experience on child nurturing & its difficulties, and gain great nurturing insights with ever ready doctors on standby to tend to complex & unanswered questions.

**Follow up and close:** This allows the health worker to search for the mother, add information like the weight, updated immunization, date of birth and also some information on the child.



**Architectural pattern of Mamacare**

Hybrid development is primarily built using HTML5 and JavaScript, that is then wrapped inside a thin native container that provides access to native platform features.

For the most part, hybrid apps provide the best of both worlds. Existing web developers that have become gurus at optimizing JavaScript, pushing CSS to create beautiful layouts, and writing compliant HTML code that works on any platform can now create sophisticated mobile applications that don’t sacrifice the cool native capabilities. Hybrid apps run on most platforms like windows, android and IOS which allows for a bigger usability and usage.

**Advantages of hybrid apps**

* Limited access to native device capabilities
* Excellent portability across platforms
* Easy to deploy like native applications
* Greatest potential for code re-use
* Decreased ownership cost
* Faster time to market

**Description of Mama care design architecture**

Mamacare application followed the MVC architectural pattern. Model View Controller or MVC as it is popularly called, is a software design pattern for developing web applications. A Model View Controller pattern is made up of the following three parts:

Model - The lowest level of the pattern which is responsible for maintaining data.

View - This is responsible for displaying all or a portion of the data to the user.

Controller - Software Code that controls the interactions between the Model and View.

MVC is popular as it isolates the application logic from the user interface layer and supports separation of concerns. Here the Controller receives all requests for the application and then works with the Model to prepare any data needed by the View. The View then uses the data prepared by the Controller to generate a final presentable response. The MVC abstraction can be graphically represented as follows.

View

Controller

Model

**The model**

This is responsible for managing the data of mama mobile application. It responds to the request made by a healthy worker from the view and it also responds to instructions he commands from the controller to update itself. The model consists of a database where all the data about the mothers is stored

**The view**

This is a presentation of data in a particular format as required, it is triggered by a controller's decision to present the data collected from the mothers. View enables the health worker to login and add the information of each mother in Mamacare database. View consists of user interface, login screens and data entry screens.

**The controller**

The controller is responsible for responding to user input and performs interactions on the data model objects. It’s a logic that manipulates the database whenever a user gives a command. For example if the healthy worker wants to update Sheila’s information and clicks on update, the controller responds by bring up Sheila’s information so that he can update it. The controller receives the input; then validates the it and performs the business operation that modifies the state of the data model.

**Reasons for choosing The MVC Architecture**

1. Divide and conquer: Three components are independently designed.

2. Increase cohesion: Components have stronger layer cohesion than if the view and controller were together in a single UI layer.

3. Reduce coupling: Minimal communication channels among the three components.

6. Increase reuse: The view and controller normally make extensive use of reusable components for various kinds of UI controls.

7. Design for flexibility: It is usually quite easy to change the UI by changing the view, the controller, or both.

10. Design for testability: Can test application separately from the UI.

**Implementation and Maintenance of the app**

Health providers interact with the OpenSRP application using mobile handsets while providing health services to their clients. All data entry is done using the platform, which can run on any Android device. The platform integrates a client registry for enumeration and continuity of care; birth and death registration; electronic forms with embedded logic and decision-support, including checklists and search.

The team plans to add work-planning, scheduling and service reminder tools; multimedia and interactive voice response (IVR) content for client counseling and reminders; and automated reporting into sub-district and national reporting systems.

**App testing:** the app will be tested before it is deployed to make all the functionalities suite the requirements.

**Updates of the app:** The mobile application will evolve over time. If properly managed, updates can demonstrate to the users that the mobile app is regularly maintained and that the developer is invested heavily into improving the mobile app. The team plans to keep improving the app.