Implement Bootstrap as a foundation for Reference Application UI

Google Summer of Code - 2019

Project Documentation

A. K. Ayeshmantha Perera
akayeshmantha@gmail.com

Aug-12-2019

Implement Bootstrap as a foundation for Reference Application UI

Introduction

Primary Mentor:
Stephen Senkomago Musoke.

Backup Mentors:
Juliet Wamalwa.

Main Objective.
Migrate Reference Application to BootStrap 4.0.

Main Influence for the Project

What is OpenMRS & Where is it?

OpenMRS is a software platform and a reference application which enables the design of a customized medical records system with no programming knowledge (although medical and systems analysis knowledge is required). It is a common platform upon which medical informatics efforts in developing countries can be built. The system is based on a conceptual database structure which is not dependent on the actual types of medical information required to be collected or on particular data collection forms and so can be customized for different uses.

OpenMRS is now in use around the world (see the OpenMRS Atlas), including South Africa, Kenya, Rwanda, Lesotho, Zimbabwe, Mozambique, Uganda, Tanzania, Haiti, India, China, United States, Pakistan, the Philippines, and many other places. This work is supported in part by many organizations including international and government aid groups, NGO's, as well as for-profit and nonprofit corporations.

As mentioned above openMRS is very popular among the African countries.And it's a very well known fact that mobile devices are very popular among these countries.Without having mobile responsive UI it's very hard for users who are using mobile devices to use the application.This is the main influence for the project to be take part on the GSoC'19.

Sub Sections Covered By The GSoC Project

1. Login Page Section.
2. Home Page Section.
3. Patient Dashboard.
4. Clinician Facing Dashboards.
5. Appointment Scheduling Section.
6. Capture Vitals Section.
7. Registration Section.
8. Data Management Section.
9. Configure MetaData Section.
    - Encounters
    - Forms
    - Locations
    - Metadata Mappings
    - Patients
    - Providers
    - Visits
10. Manage Modules Section.
11. Mange Global Properties Section.
12. Manage Accounts Section.
13. Chart Search Section.

1. Integrating BootStrap 4 to the application

During the time of starting the project Bootstrap release 4.0 was stable enough to make it use for the application and it contains all the new functionality as a framework.Integrating bootstrap 4 is possible in few ways

- Having CDN integrated.
- Having minified css/js files locally.
- Having as an npm package.
- Nuget package for c# mvc api projects.

Reference application is modularize application where end users can install modules as per the requirement.With the template these pages are using there is only one place that needs bootstrap integration,since this template will be covering all the pages of the application(There are a few exceptions).

The common page is standardEmrPage.gsp which can be found in the AppUI module.As the JS/CSS best practices integrated the cdn files to this page.

As mentioned above there are some exceptional cases to handle.

- Login Page :- Login page is similar to other pages handled in the same way.

- OWA based modules :- Not covered from the project scope.

Login Page

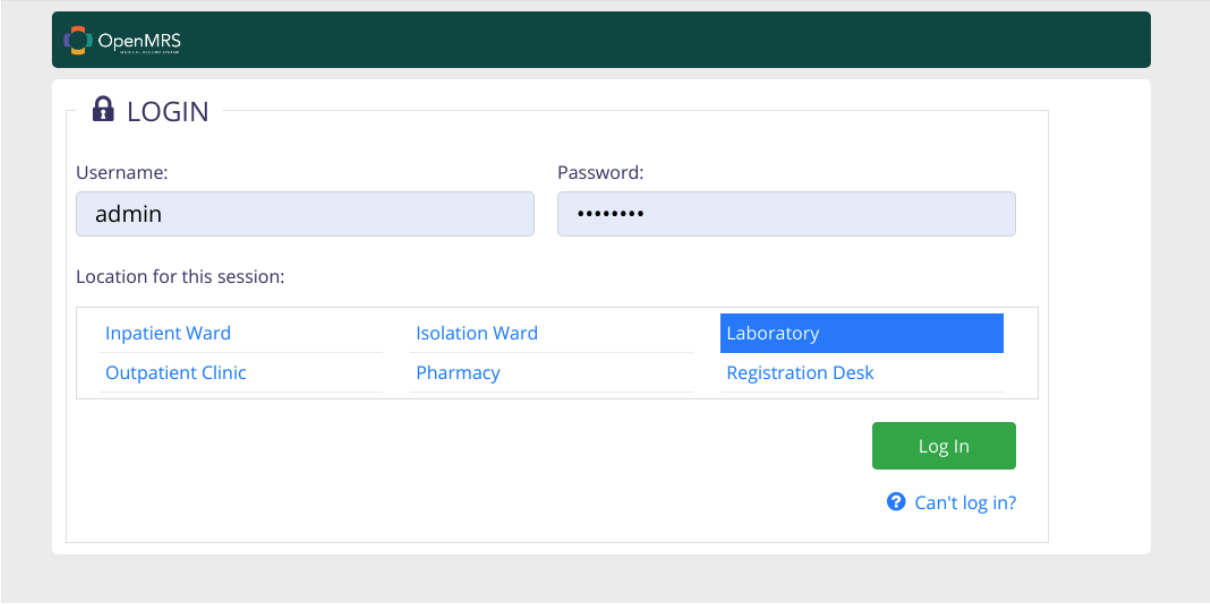
1. Adding bootstrap 4 reference css and js files.
2. Adding meta tag for viewport to support mobile responsiveness.

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

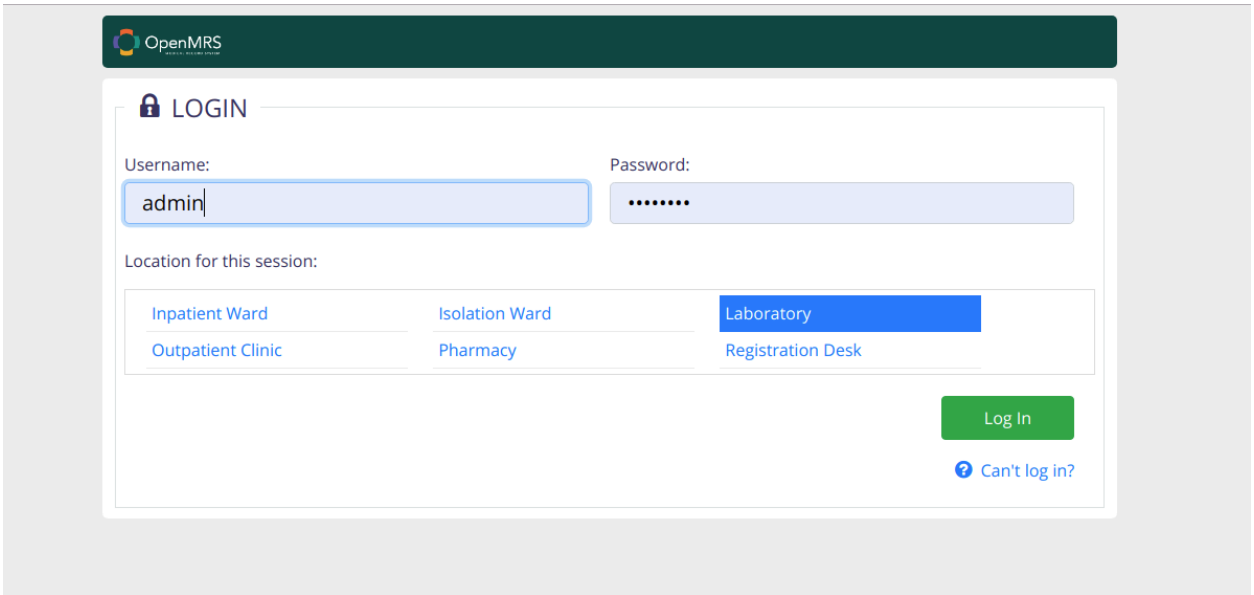
3. Removing login.css reference from the page.

With the file it conflicts <legend> tag features which are already provided by bootstrap.

With the file is referred to how the broken page look.



Without the file how it looks like



As it shows the legend fully covers the page with the bootstrap supported tags.

4. The responsive grid**.**

To maintain the responsiveness of the page there was a need to add the below code segment.

```
107
108  <div id="content" class="container-fluid">
109      <div class= "row">
110          <div class="col-12 col-sm-12 col-md-12 col-lg-12">…
112          </div>
113      </div>
114      <div class= "row">
115          <div class="col-12 col-sm-12 col-md-12 col-lg-12">…
123          </div>
124      </div>
125
126      <div class= "row">
127          <div class="col-12 col-sm-12 col-md-12 col-lg-12">…
186          </div>
187      </div>
188      <div class= "row">
189          <div class="col-12 col-sm-12 col-md-12 col-lg-12">…
201          </div>
202      </div>
203
```

With the reference of another developer, I added a screenshot with the line numbers

These are the predefined width for the tags by bootstrap.

The Bootstrap 4 grid system has five classes:

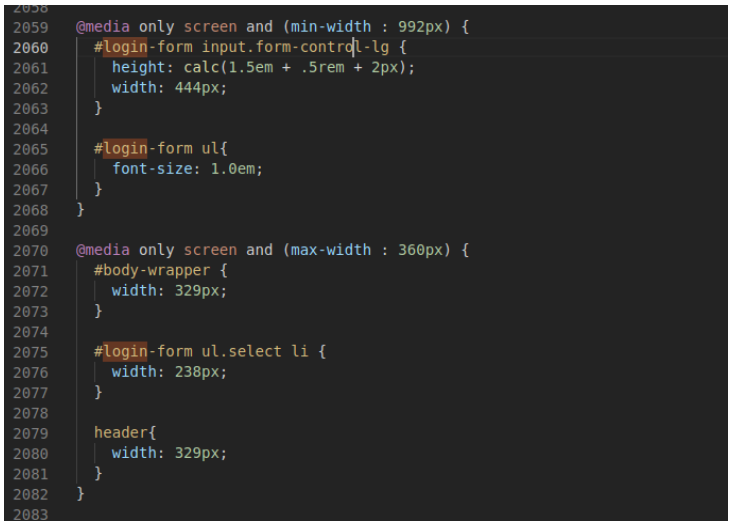
- `.col-` (extra small devices - screen width less than 576px)
- `.col-sm-` (small devices - screen width equal to or greater than 576px)
- `.col-md-` (medium devices - screen width equal to or greater than 768px)
- `.col-lg-` (large devices - screen width equal to or greater than 992px)
- `.col-xl-` (xlarge devices - screen width equal to or greater than 1200px)

5. Form control with responsiveness.

```
<input class="form-control form-control-sm form-control-lg
form-control-md" id="password" type="password" name="password"
placeholder="${ui.message("reference
application.login.password.placeholder") }"/>
```

This is added to have the responsiveness of form controls as suggested by bootstrap.

But in the medium size devices (medium devices - screen width equal to or greater than 768px) seems to be the input field sizes were not enough for a user to interact with.Inorder to fix this **had to add media query for medium size devices.And it was a similar situation for old devices like iphone 5s and samsung galaxy S5.**

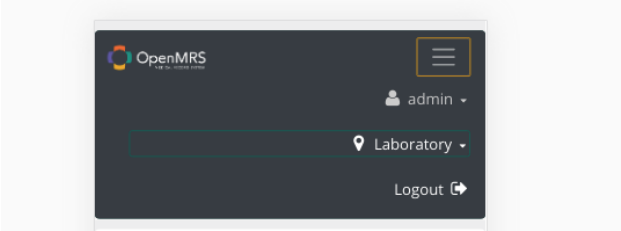


```
@media only screen and (min-width : 992px) {
  #login-form input.form-control-lg {
    height: calc(1.5em + .5rem + 2px);
    width: 444px;
  }

  #login-form ul{
    font-size: 1.0em;
  }
}

@media only screen and (max-width : 360px) {
  #body-wrapper {
    width: 329px;
  }

  #login-form ul.select li {
    width: 238px;
  }

  header{
    width: 329px;
  }
}
```

Header Component & Home Page Section

Adding new navbar with collapse on mobile views.

As the current reference application is not responsive to mobiles after having the support to bootstrap it was a must to have a navigation bar with collapse support.



Identifying the mobile interfaces in order to disable the hover events in header dropdowns

This feature was also a must in order to align with new responsiveness.

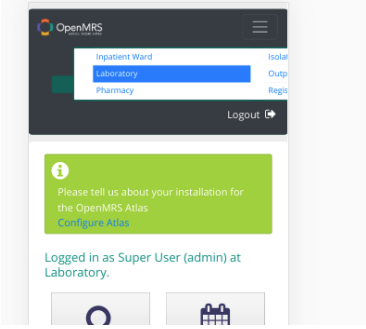
```
var event = ('ontouchstart' in window) ? 'click' : 'mouseenter
mouseleave';
```

With the event changing according to the device changed the jquery function to change and work accordingly.

```
        jq('.identifier').on(event,function(){
                jq('.appui-toggle').toggle();
                // jq(this).toggleClass("appui-toggle");
                jq('.appui-icon-caret-down').toggle();
        });
```

The location selector fix for mobile screens.

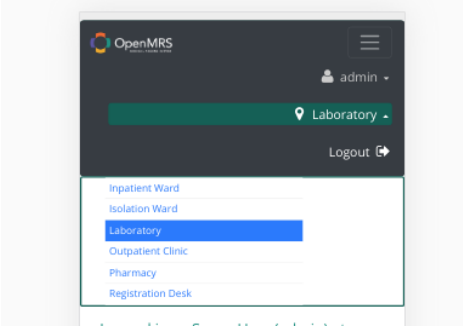
With the new ui framework the location selector became unresponsiveness because of the fixed width and top values already set.The work around was to add a media query which unsets the unwanted values for the mobile phones.



The media query for the fix.

```
@media only screen and (max-width : 576px) {
   header #session-location {
       display: flex;
       top:unset !important;
       width:unset !important;
       left:unset !important;
   }
 }
```

After the fix how it looks alike.



Home Page Grid.

To maintain the Home Page grid responsiveness like in the Login section used the same grid system given by bootstrap.

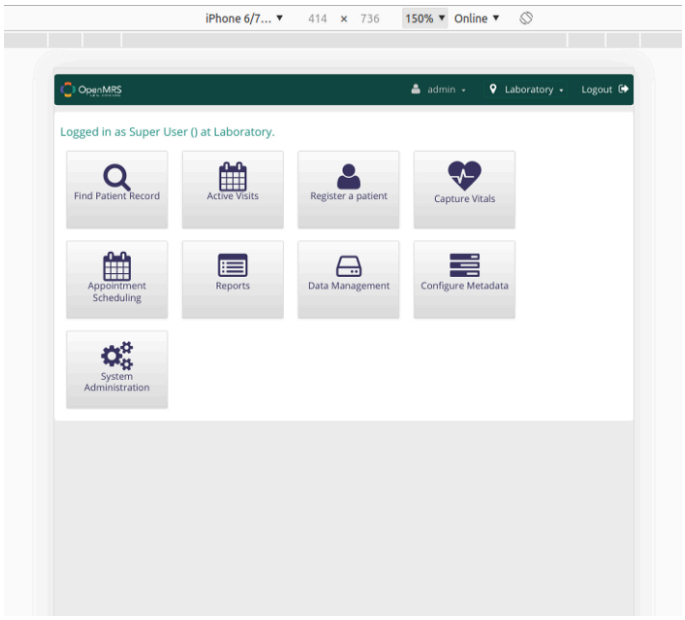
Apart from that used the below classes with already existing classes to maintain the sizes of the buttons of the grids.

```
   <div  class="col-12 col-sm-12 col-md-12 col-lg-12 homeList">
            <% extensions.each { ext -> %>
                <a id="${ htmlSafeId(ext) }" href="/${ contextPath
}/${                  ext.url }" class="btn btn-default btn-lg button
app big align-self-center" type="button">
                </a>
            <% } %>
   </div>
```

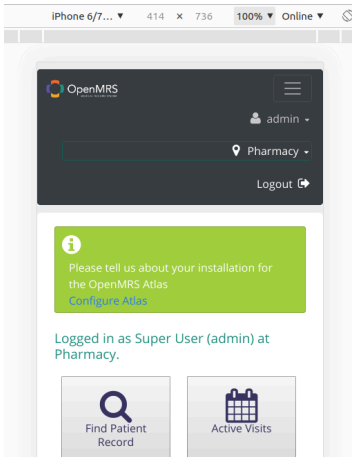
And also to maintain the responsiveness of the page in mobile devices had to introduced few media queries.

```
@media only screen and (min-width : 320px) and (max-width : 365px)  {
 .homeList a {
   width: 117px !important;
 }
}
@media only screen and (max-width : 410px) and (min-width : 375px)  {
 .homeList a {
   width: 140px !important;
 }
}
@media only screen and (max-width : 414px) and (min-width : 411px)  {
 .homeList a {
   width: 158px !important;
 }
}
@media only screen and (min-width : 768px) and (max-width : 991px)  {
 .homeList a {
   width: 156px !important;
 }
}
```

Before the migration view on a mobile phone.



After the migration view on the mobile phone.



Patient Dashboard Section

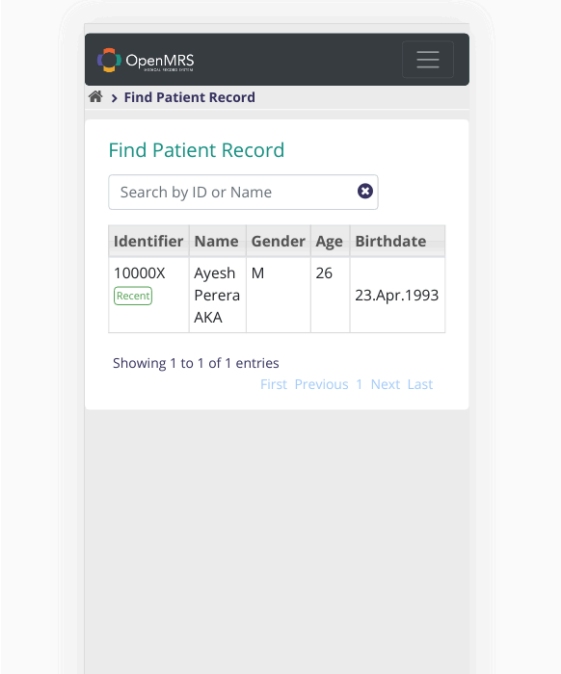
Patient Search Page.

It was a bit tricky when working with this groovy page.It was challenging to keep all the functionalities and the same look and feel of the page.It was a bit tricky since having responsive tables which was now supported by bootstrap newest release only.And since lack of documentation on this and have multiple tickets opened for bootstrap itself was a little bit misleading but finally I was able to find the correct way and after the fix the page was looking good with mobile responsiveness.

Had to introduce the below code segment which makes the tables responsive.

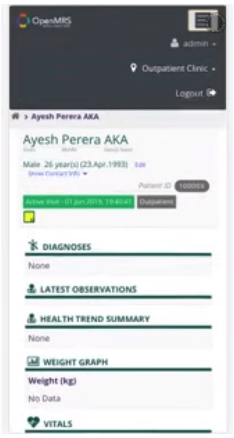
```
<table class="table table-sm table-responsive-sm table-responsive-md
table-responsive-lg table-responsive-xl">
</table>
```

After migration how page looks alike.



Patient Page.

With the usage of bootstrap grid system it was easy to migrate this page in to bootstrap 4.Nothing much added on the page it self.After migration how it looked alike is like below.



Patient Dashboard Page.

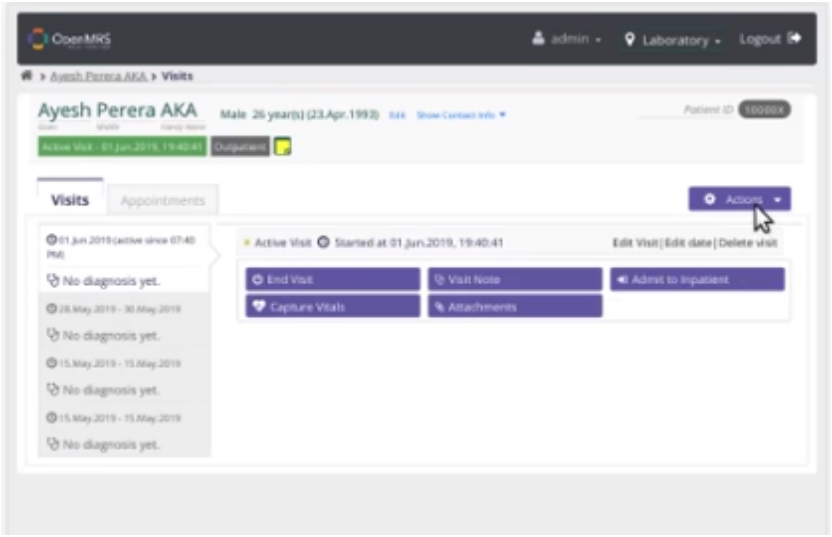
Patient Dashboard had lots of changes when migrating to bootstrap 4.

Migrating the action cog and the drop down

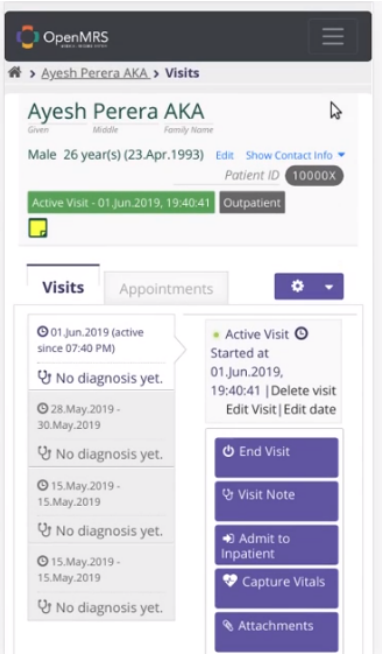
Migrating the action cog and making it responsive with the screen size was little bit tricky.The approach taken was making the text disappear on the cog button on smaller screens.To achieve the target used Bootstrap 4 native tags **Display property.**It's possible with the below code segment to easily get this working with bootstrap 4.

```
<span class="d-none d-sm-none d-md-inline d-lg-inline"> ${
ui.message("coreapps.actions") } </span>
```

On Larger Screens

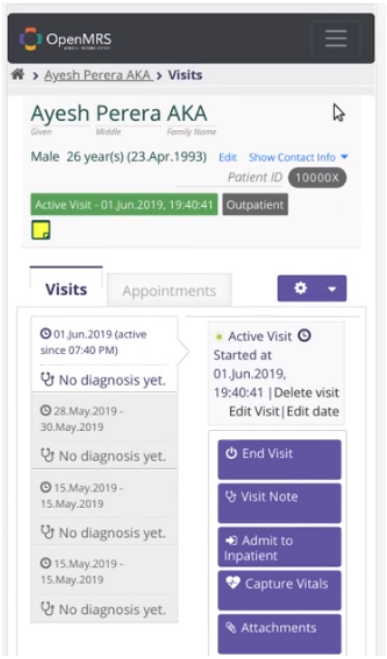


On Smaller Screens



Migrating the visits section

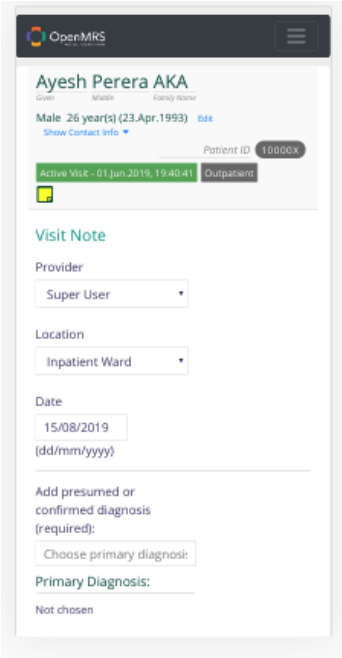
Migrating the visits details is another major challenge.Had to introduce few media queries in order to make it work on mobile devices as well.



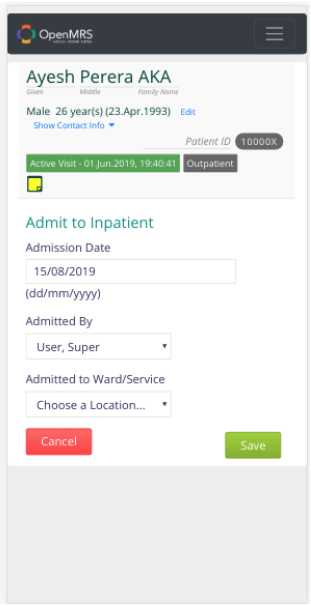
Visit Note Page, Mark Patient Dead & Admit to impatiens Page.

Both Pages are in the same template only the labels are changing according to the page.Introduced bootstrap forms elements to the pages which are making the forms responsive.

Visit Note Page

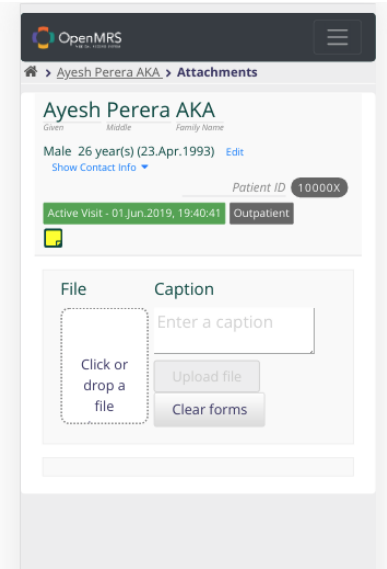


Admit to impatiens Page



Attachments Page.

In there have to align the components which were there to upload image with the width of the page.And also to make it responsive.



Merge Visits Page.

For the merge visits page introduced the responsive tables with the below code segment.Since it's only the element which has to be migrated and the header component and the user details component is already migrated in the common template page.

```
<table class="table table-sm table-responsive-sm table-responsive-md
table-responsive-lg table-responsive-xl" id="active-visits" width="100%"
border="1" cellspacing="0" cellpadding="2">
```

Appointment Scheduling Section.

Home Page.

To maintain the Home Page grid responsiveness like in the ref app home page section used the same grid system given by bootstrap.

Apart from that used the below classes with already existing classes to maintain the sizes of the buttons of the grids.

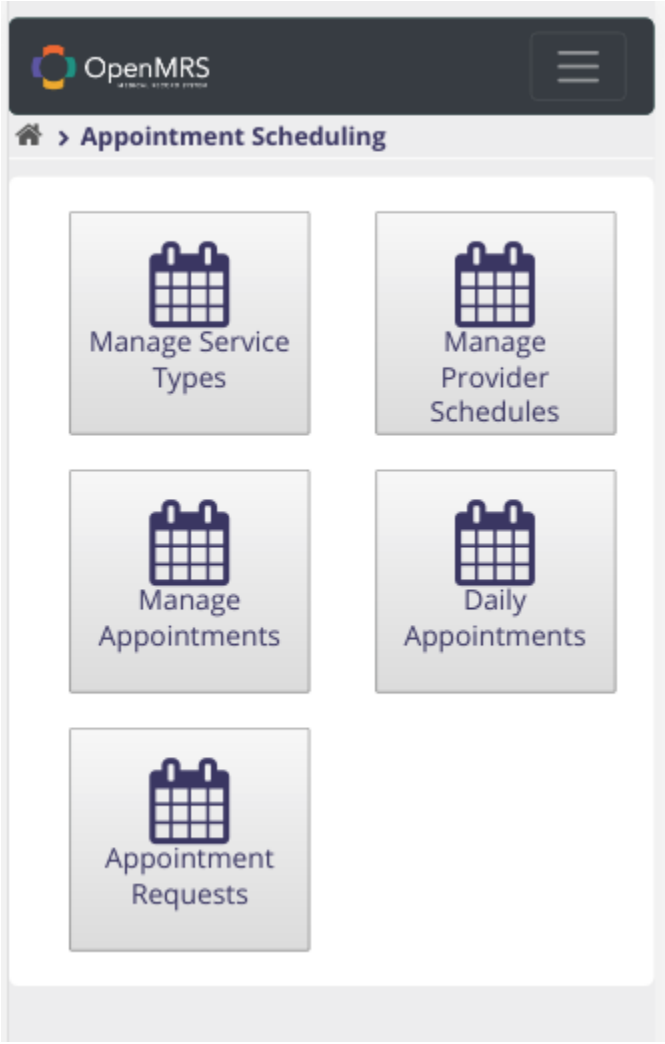
```
   <div  class="col-12 col-sm-12 col-md-12 col-lg-12 homeList">
           <% extensions.each { ext -> %>
               <a id="${ htmlSafeId(ext) }" href="/${ contextPath
}/${                  ext.url }" class="btn btn-default btn-lg button
app big align-self-center" type="button">
               </a>
           <% } %>
   </div>
```

And also to maintain the responsiveness of the page in mobile devices had to introduced few media queries.

```
@media only screen and (min-width : 320px) and (max-width : 365px)  {
 .homeList a {
   width: 117px !important;
 }
}
@media only screen and (max-width : 410px) and (min-width : 375px)  {
 .homeList a {
   width: 140px !important;
 }
}
@media only screen and (max-width : 414px) and (min-width : 411px)  {
 .homeList a {
   width: 158px !important;
 }
}
@media only screen and (min-width : 768px) and (max-width : 991px)  {
 .homeList a {
   width: 156px !important;
```

```
  }
}
```

After Migration How Page Looks Alike

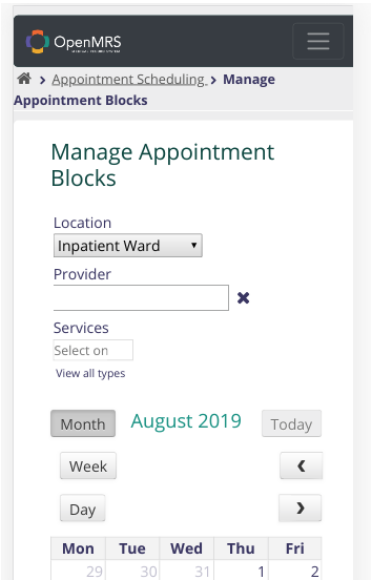


Manage Service Types & Add Service Type Pages.

Other than introducing the bootstrap grid to the page had to make sure the table is responsive as well.As like in other tables responsive table classes were introduced to the page.And adding a new service page is a form component where had to integrate bootstrap form elements.

Manage Service Types Page.

In the managed service type page there are few changes to do.Making the time filters responsive and pagination responsive are the main tasks.With the bootstrap grid system it was achievable.



Manage Appointments Page, Appointment Requests & Daily Appointments Page.

It’s the same as the patient's search page by introducing the responsive table was done.

Capture Vitals Section.

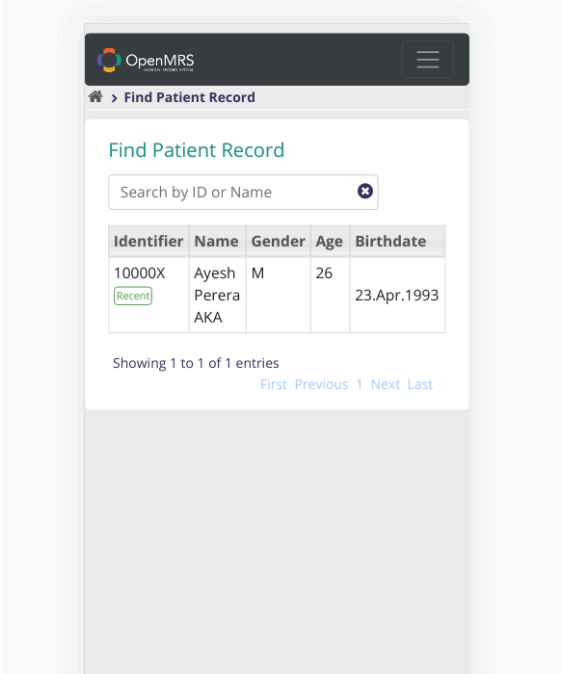
Patient Search Page.

It was the same page as the patient search page.It was a bit tricky when working with this groovy page.It was challenging to keep all the functionalities and the same look and feel of the page.It was a bit tricky since having responsive tables which was now supported by bootstrap newest release only.And since lack of documentation on this and have multiple tickets opened for bootstrap itself was a little bit misleading but finally I was able to find the correct way and after the fix the page was looking good with mobile responsiveness.

Had to introduce the below code segment which makes the tables responsive.

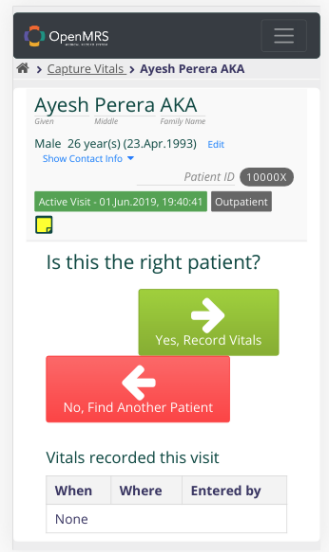
```
<table class="table table-sm table-responsive-sm table-responsive-md
table-responsive-lg table-responsive-xl">
</table>
```

After migration how page looks alike.



Vitals Section Patient Page.

In this page the main action item was to make the table which shows vital responsive as well as make the two arrows(going back and proceed) responsive.With the table responsive tags made the changes and after the change the page is like below.

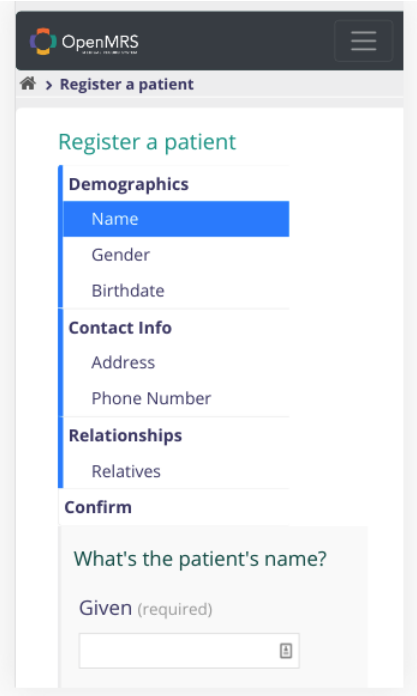


Vitals Capture Form Page.

There was nothing much to do in this section since already the forms are made to be responsive in previous sections.

Register Patient Section.

This section is to register a new patient to the openMRS application.It has a step by step guide when proceeding forward.As like in other form sections this forms input elements been migrated to automatically because of the form section changes previously.



Data Management Section.

Data Management Page.

To maintain the Data Management Page grid responsiveness like in the Login section used the same grid system given by bootstrap.

Apart from that used the below classes with already existing classes to maintain the sizes of the buttons of the grids.

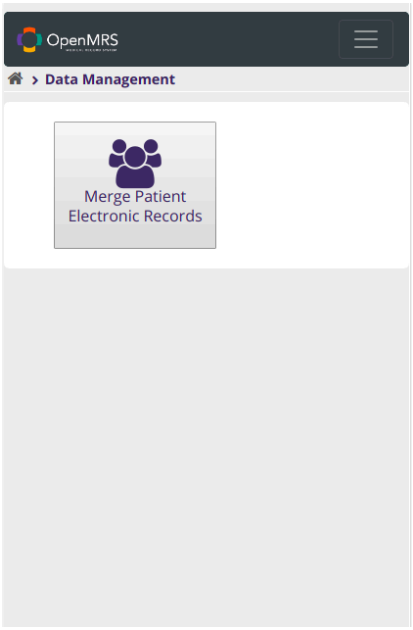
```
<div  class="col-12 col-sm-12 col-md-12 col-lg-12 homeList">
```

```
            <% extensions.each { ext -> %>
                <a id="${ htmlSafeId(ext) }" href="/${ contextPath
}/${                    ext.url }" class="btn btn-default btn-lg button
app big align-self-center" type="button">
                </a>
            <% } %>
    </div>
```

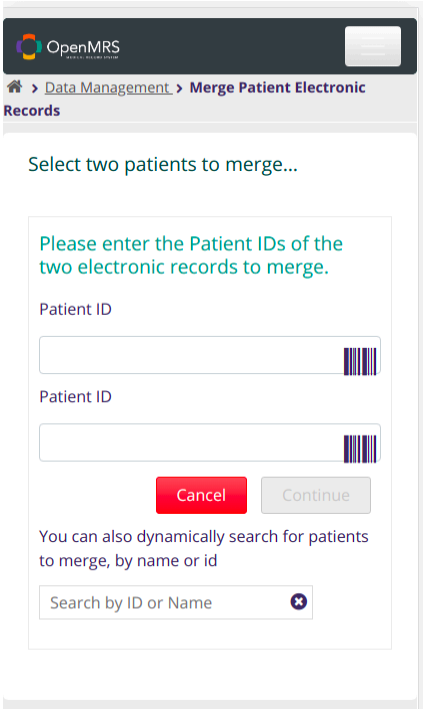
And also to maintain the responsiveness of the page in mobile devices had to introduced few media queries.

```
@media only screen and (min-width : 320px) and (max-width : 365px)  {
 .homeList a {
   width: 117px !important;
 }
}
@media only screen and (max-width : 410px) and (min-width : 375px)  {
 .homeList a {
   width: 140px !important;
 }
}
@media only screen and (max-width : 414px) and (min-width : 411px)  {
 .homeList a {
   width: 158px !important;
 }
}
@media only screen and (min-width : 768px) and (max-width : 991px)  {
 .homeList a {
   width: 156px !important;
 }
}
```

After migration how page looks like.



Merge Patients Page



This page contains a form which takes two patients id to do the merging.Introduced the form elements to make the page responsive.

Configure MetaData Section.

Main page contains below sections.

- Encounters

A patient visits a health center or hospital. For each electronic form completed for a patient, a new encounter is created. Each will have a unique encounter_id and encounter_type. Forms could be completed by different departments (ie. drug pickup, visit with an HIV clinician, Diabetes visit, food package received), and will have an associated encounter_type (ie. ART Drug Regimen Pickup, Adult intake, food assistance, lab test, etc). Each encounter has an encounter type, date/time, location and provider.

- Forms

OpenMRS provides **Form**, **Field** and **FormField** objects through the API as well as schema creation in the webapp. FormEntry Module, HTML Form Entry Module and XForms Module all utilize the built-in Form infrastructure. OpenMRS also provides API calls to store serializable resources or metadata on any given Form instance.

- Locations

OpenMRS has the privilege based access control implementations. Like that one, we implemented a Location based Access control system. It will manage the access to the services based on the locations. Some implementations want to register the users and patients (the people) in certain selected locations. Then access them based on the location that someone has logged in. That way, if someone is logged in a certain location, they should see only those encounters, observations, and patients registered in that location.

- Metadata Mappings

The goal of Metadata Mapping is to solve many of the problems around metadata management by providing an easy & explicit way for metadata within a system (encounter types, location, etc.) to be mapped to external vocabularies.

- Patients

A patient identifier is any unique number that can identify a patient. Examples are a Medical Record Number, a National ID, a Social Security Number, a driver's license number, etc. A patient can have any number of identifiers. The Patient Identifier Type table defines what type of identifiers are collected in your system.

A patient can only have multiple identifiers of each type defined in your system. E.g. a patient could have 5 identifiers of type of "Medical Record Number" because they were seen at 5 different hospitals that collected 5 different types of ids.

The patient search screen searches across all identifiers that are still active for a patient.

New identifier types are generally created if they have different characteristics. E.g. one identifier can be only a string of numbers, another is a number with a hyphen plus a check digit, etc.

- Providers

An OpenMRS Service Provider is an organization that has the commitment and capacity to perform software development, advising, training, and/or implementation work using and promoting products in the OpenMRS ecosystem.

- Visits

Visit in OpenMRS represents exactly what it sounds like: a time period when a patient is actively interacting with the healthcare system, typically at a location. The metadata differentiating different types of visits is a Visit Type. Visit Types are displayed in the user interface, and can be searched against.

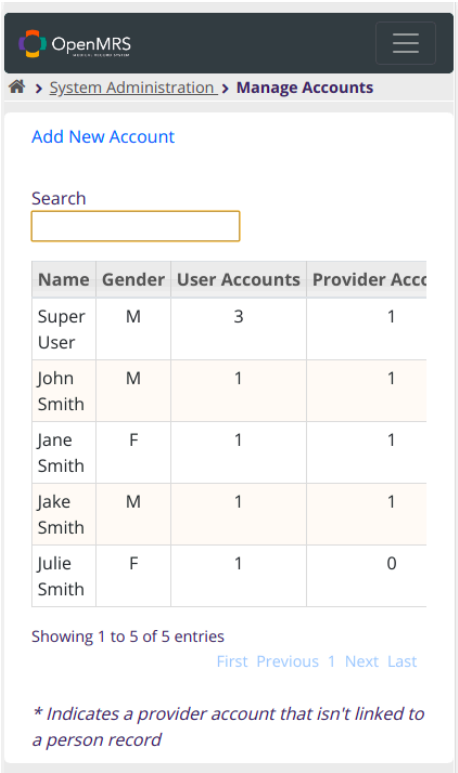
A visit contains encounters, which store more granular data about treatments or services.

At the Amani Clinic, a patient might typically check-in at registration, be seen by a doctor, and receives medication dispensed in the pharmacy. This would be recorded as one visit of type of Outpatient, and contain three encounters (Registration, Consultation, and Dispensing).

All the above sections contains same template of pages.Which has table to display the contents.And a fom which can add new items to the sections.

Manage Accounts Section.

As most of the other pages this page also contains a search bar and a table.After integrating with bootstrap tables it looks like below.



Next Step.

Next step will be testing the features in the testing server.