CliniConnect

Group 1

Benjamin Chrysler (ID:08)

Shweta Parihar (ID:42)

Luke, McDuff(ID:27)

Sri Patluri (ID:44)

Project Goal and Objectives (revised)

Overall goal: This application is to improve contact between patient (in particular, low income demographic) and there clinic which they attend.

Objectives:

We are planning to create a mobile application that will do the following:

- Patient can fill the form prior to clinic appointment, which in turn reduce the patient and doctor waiting time and will also improve the wait time in busy inner city clinics.
- Patient can fill the exit satisfaction form as per his comfort.
- Patient will be able to get his lab order for bloodwork etc.… electronically, which can be scanned.
- Patient will get reminder notification of his appointment.
- Patient who are required to monitor their blood pressure and blood sugar daily or weekly can keep track of their readings using this application.
- A map that will show from whatever location opened, the directions to the clinic, this would benefits patients who may require rides from friends or family members unfamiliar with the location of the clinic.
- Administrator will be able to administer the application using admin page.

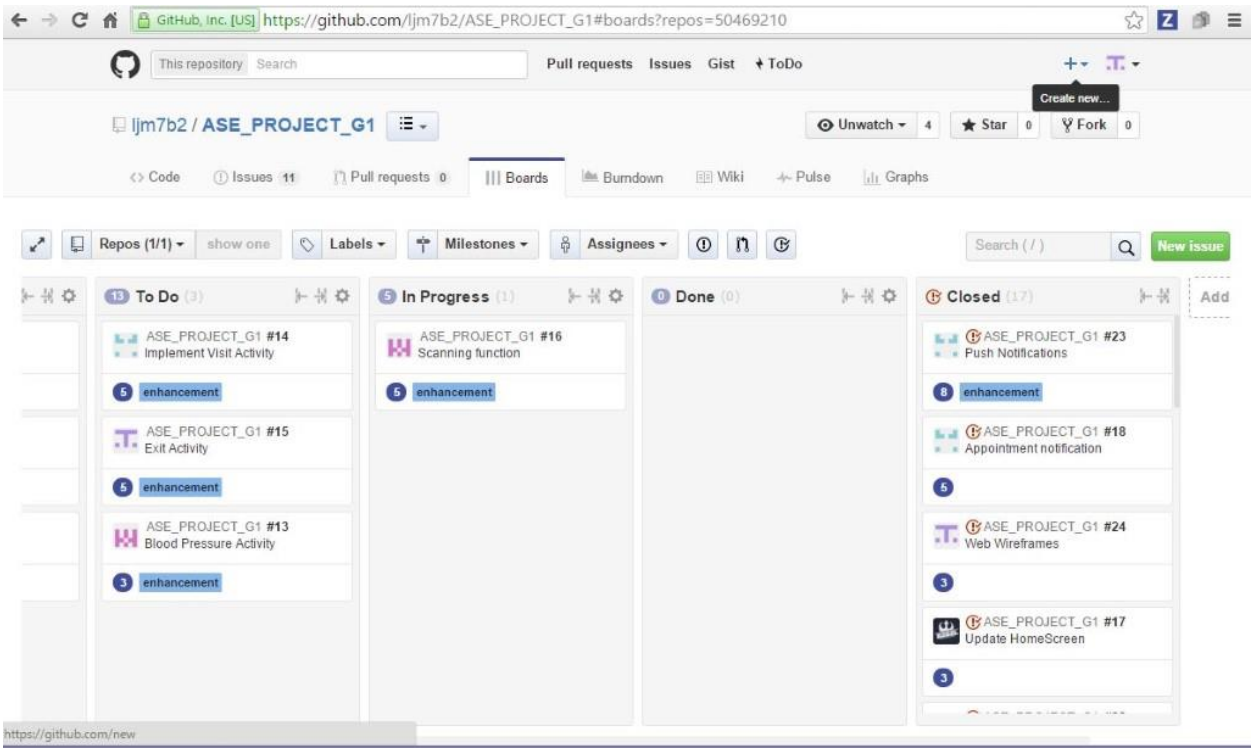
Features:

- New patient can be registered in the registration page of the web application.
- Already registered patient can login to the application using the login page.
- Prior clinic visit form page with the exact replica of the actual physical form.
- Exit satisfaction form page.
- Bloodwork details page with the facility to be scanned by pathology labs.
- Reminder icon on the status bar for the upcoming appointment.
- Blood pressure page for updating and keeping track of daily or weekly blood pressure readings.
- Blood sugar page for updating and keeping track for blood sugar readings.
- A map that will show from whatever location opened, the directions to the clinic, this would benefits patients who may require rides from friends or family members unfamiliar with the location of the clinic.
- Administrator can add new patient to the system, send blood work notification to the patient.

Significance: While some patient-to-clinic apps exist, they are typically patient portals which can contain an overwhelmingly large amount of data and features. By streamlining a few key important services we can help increase patient health.

Project Plan

Zenhub screenshot



Schedule for the four different increments.

Stories (Issues): The third increment contains a few additional improvements to our user android application administrator side of our service. We have incorporated an ionic framework to our administration side, making it more streamlined than before. Also, forms were added to have the Users fill out information about their visit and reason for visiting. The User Application now includes a blood log utility that allows users to input new blood logs that are stored on the server and can be retrieved on the administration side as well. Additionally, the information about the clinic is posted from the administration page to the database and is retrieved on the user app from the database.

Service Design: The service design at this stage, includes an update to the ionic framework for the admin site. Admin will be able to update information about the clinic on the web site. Also, Users and Admin can now send and receive information in forms about the User's reason for visiting. The User is now able to update their blood logs regularly, which the administrators can check on.

Service Implementation: The service implementation of the administration side has been updated to the ionic framework and is now more user friendly. Also, forms have been added to the administration side about the patient's visit. On the user application side, the blood log service has been implemented, which has also been updated on the admin site.

Project Timelines, Members, Task Responsibility:

The work division has been made in Zenhub please check our board for more detail.

https://github.com/ljm7b2/ASE_PROJECT_G1/milestones#boards?repos=50469210

User Stories: When the User visits the Clinic, either the user can fill out a reason for visit form on the app. Or, for initial visits the admin site can fill out the form. When the User takes blood pressure or blood sugar, the user can input the information into the blood log portion of the user app. When the Admin wants to Navigate to any other page the User can use the navigation bar to redirect to another page.

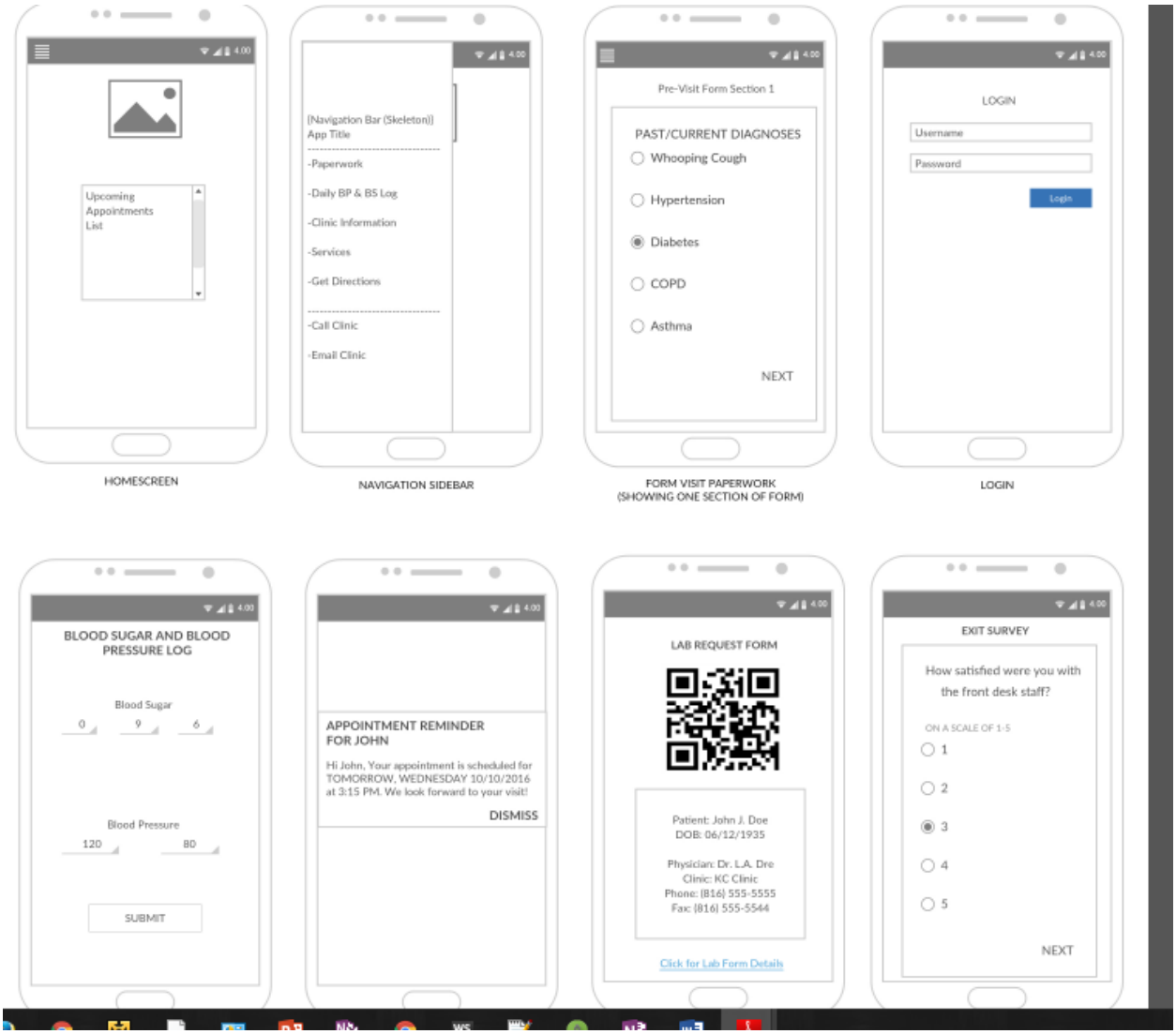
Use Case

Service description: CliniConnect is an service that helps patients handle their paperwork in an organized and timely fashion. The application is intended for those who either do not have the time to come early to an appointment to fill out paperwork, have a tendency to miss/forget appointments and even those who need more constant interaction with their medical clinic. The service includes an administration portal that serves as to connect the administration to the user. This service allows administrators to keep their patients properly up to date.

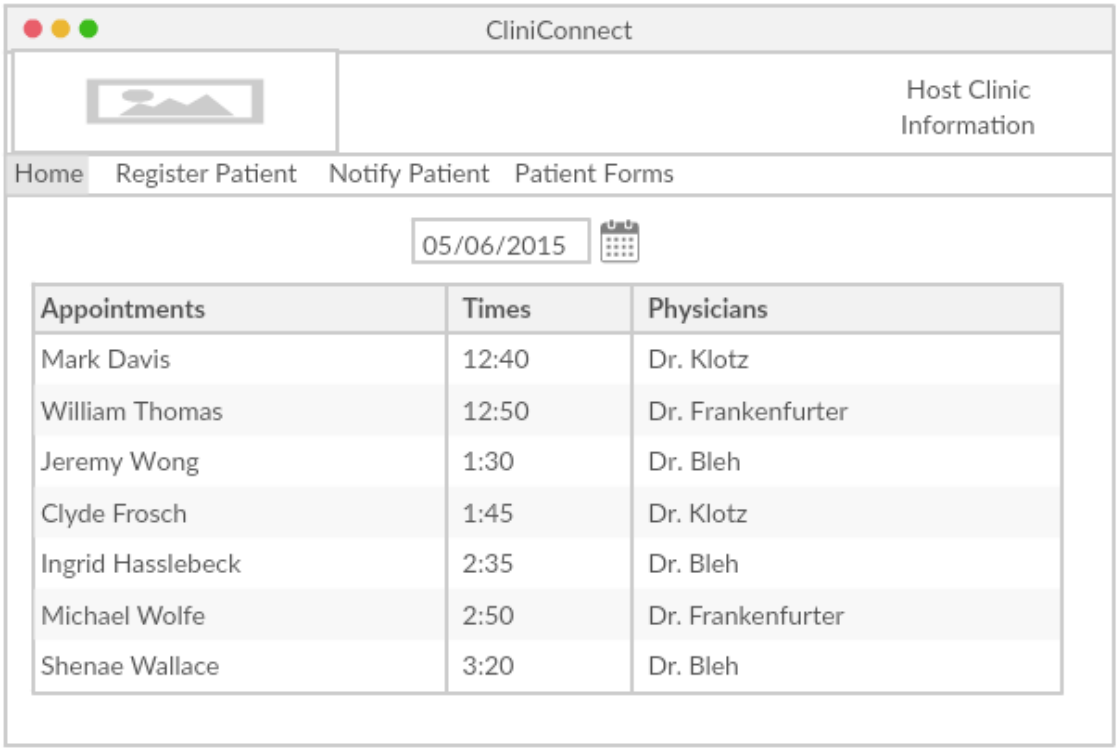
Second Increment Report

In this third increment of "**CliniConnect"** we have implemented versions that follow the overall structure and flow of the application and administration wireframes and UML diagrams. In this phase we have updated the implementation of the administration portal. Included in this increment, is the ability to fill out forms related to reasons for visiting. Also, we have implemented the blood log option in our user app. The blood log also comes with some basic text feedback based on the results of the blood.

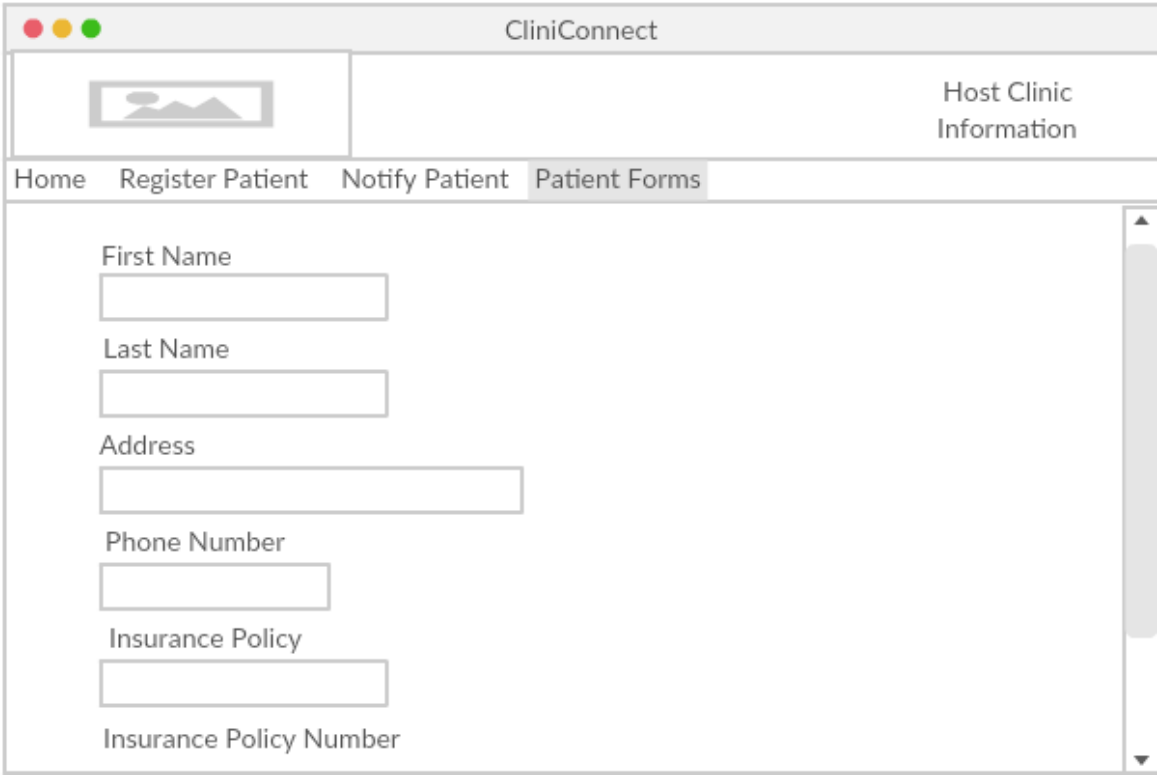
Detail Design: Wireframes

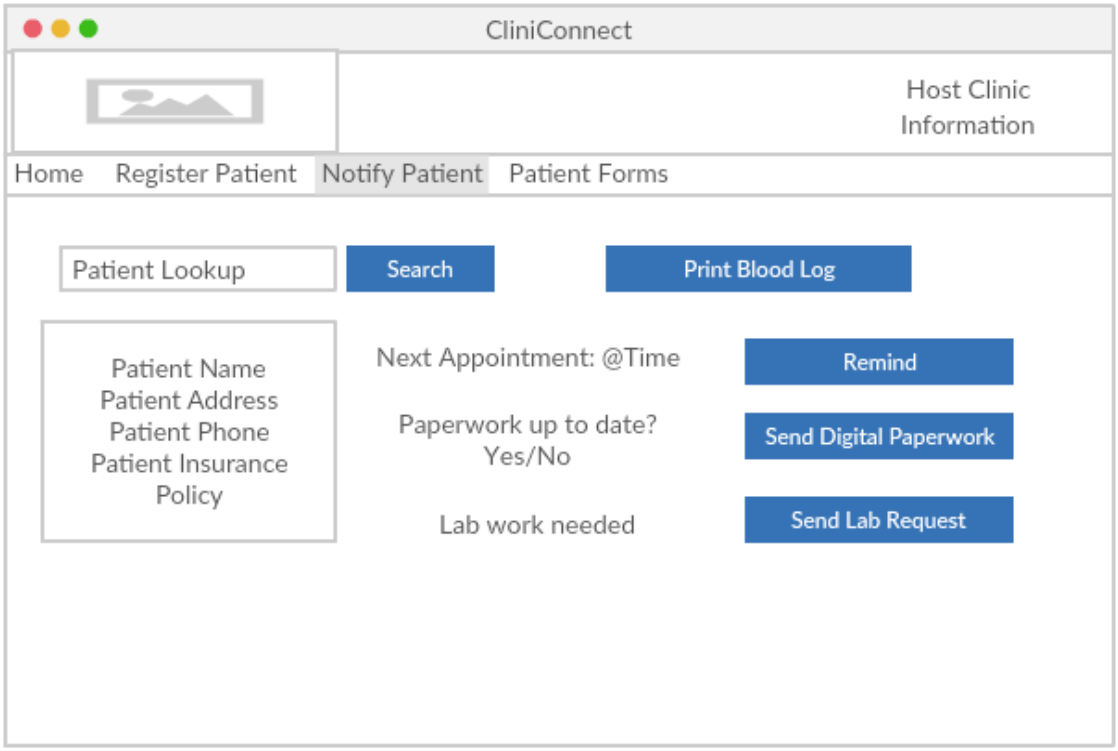


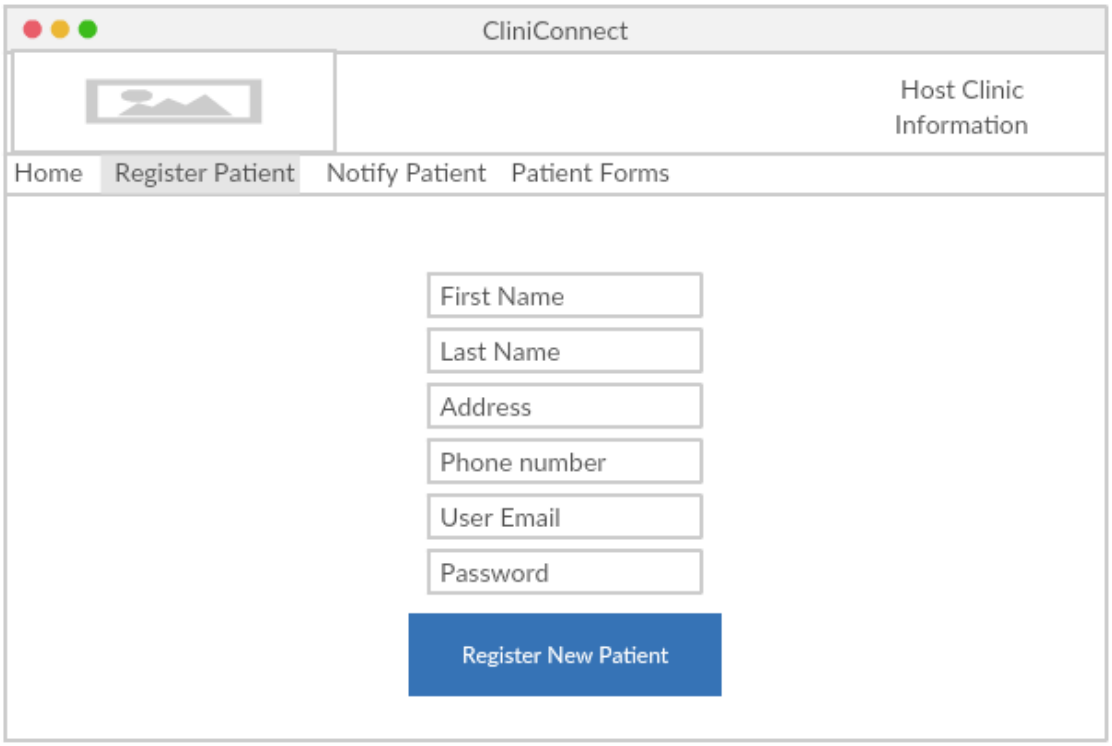
Web Wireframes:



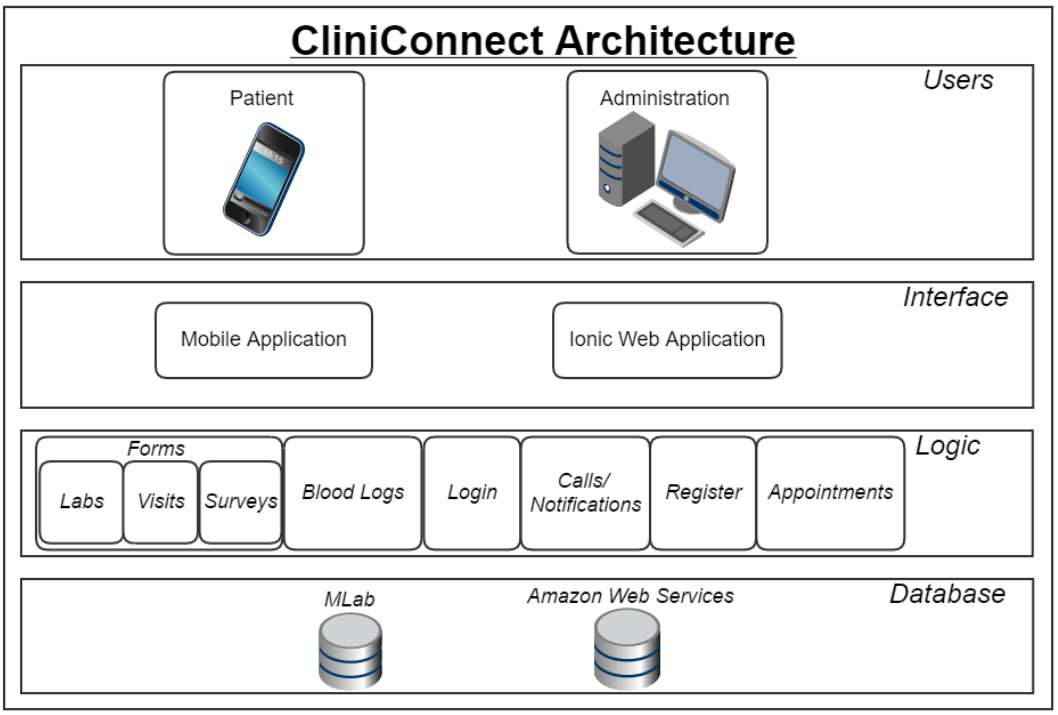
Appointments	Times	Physicians
Mark Davis	12:40	Dr. Klotz
William Thomas	12:50	Dr. Frankenfurter
Jeremy Wong	1:30	Dr. Bleh
Clyde Frosch	1:45	Dr. Klotz
Ingrid Hasslebeck	2:35	Dr. Bleh
Michael Wolfe	2:50	Dr. Frankenfurter
Shenae Wallace	3:20	Dr. Bleh



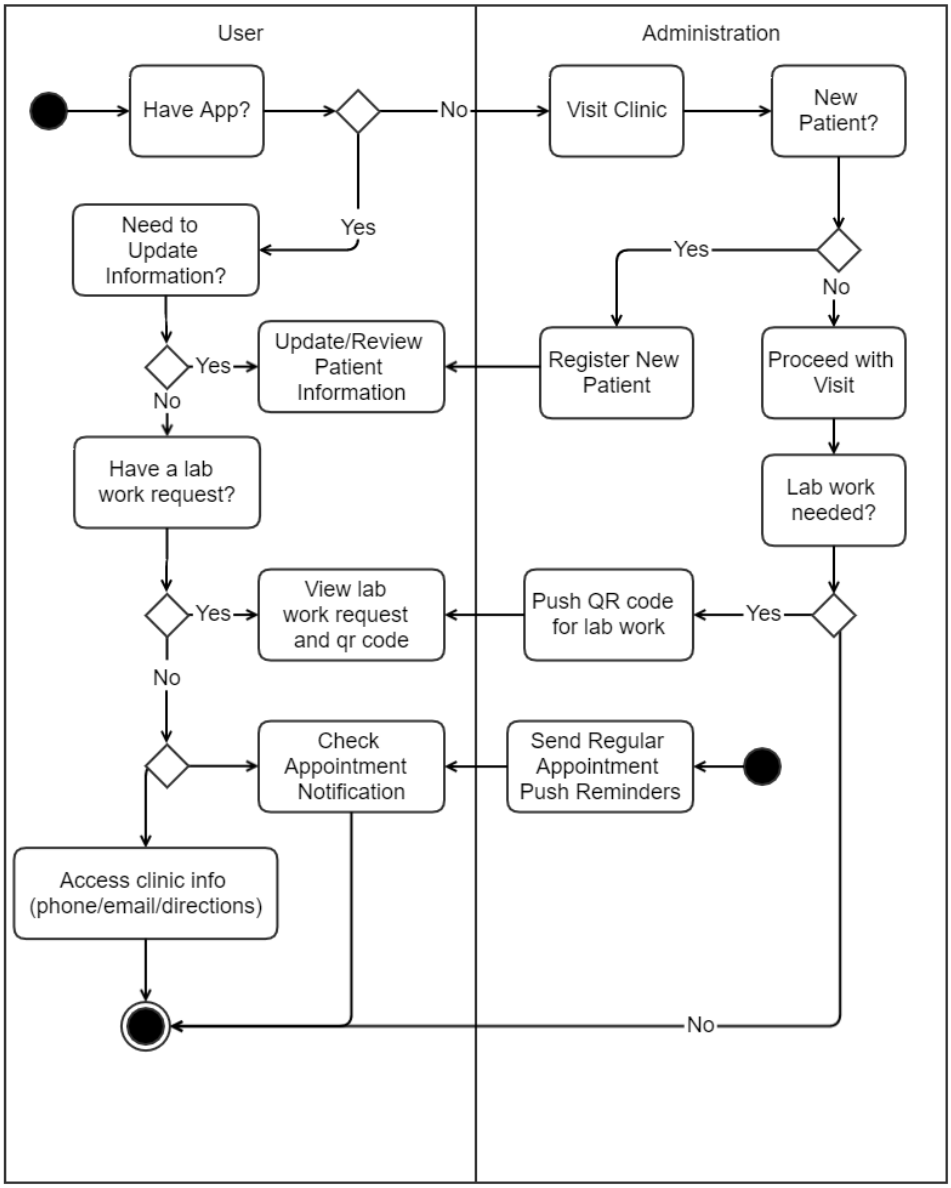




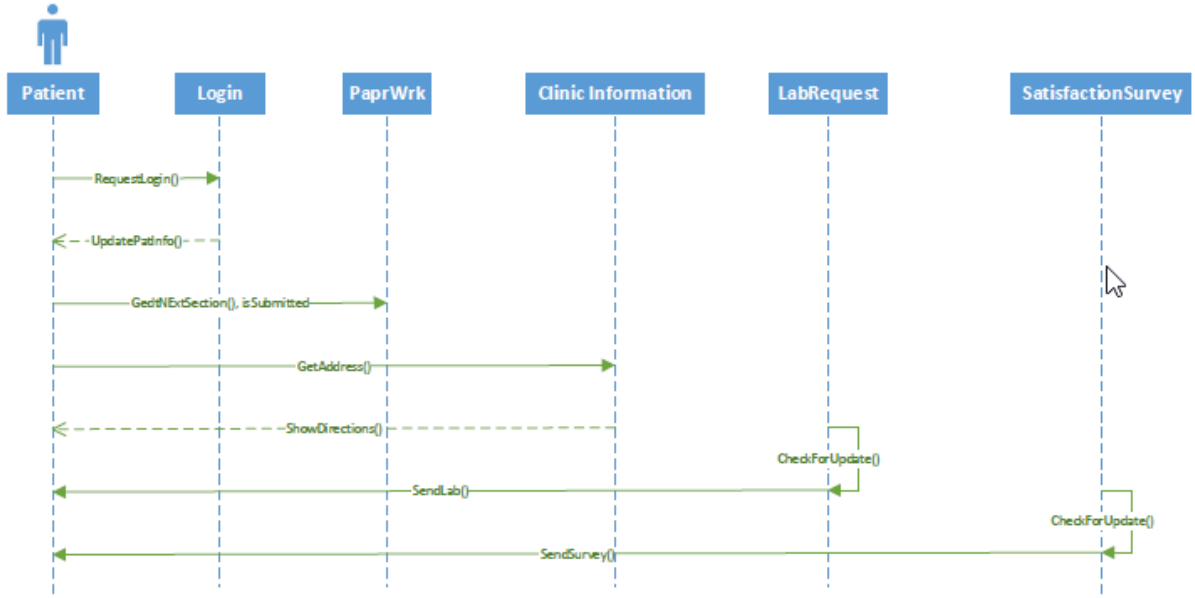
Architecture Diagram:



Activity Diagram:



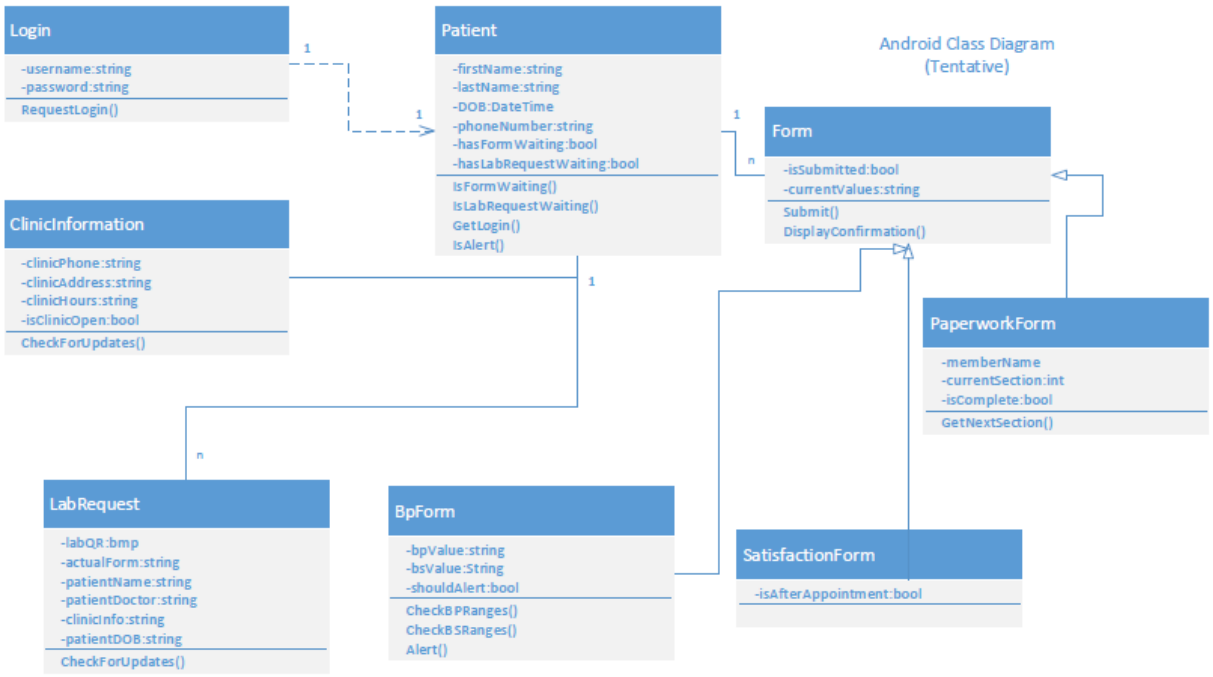
Sequence diagram:



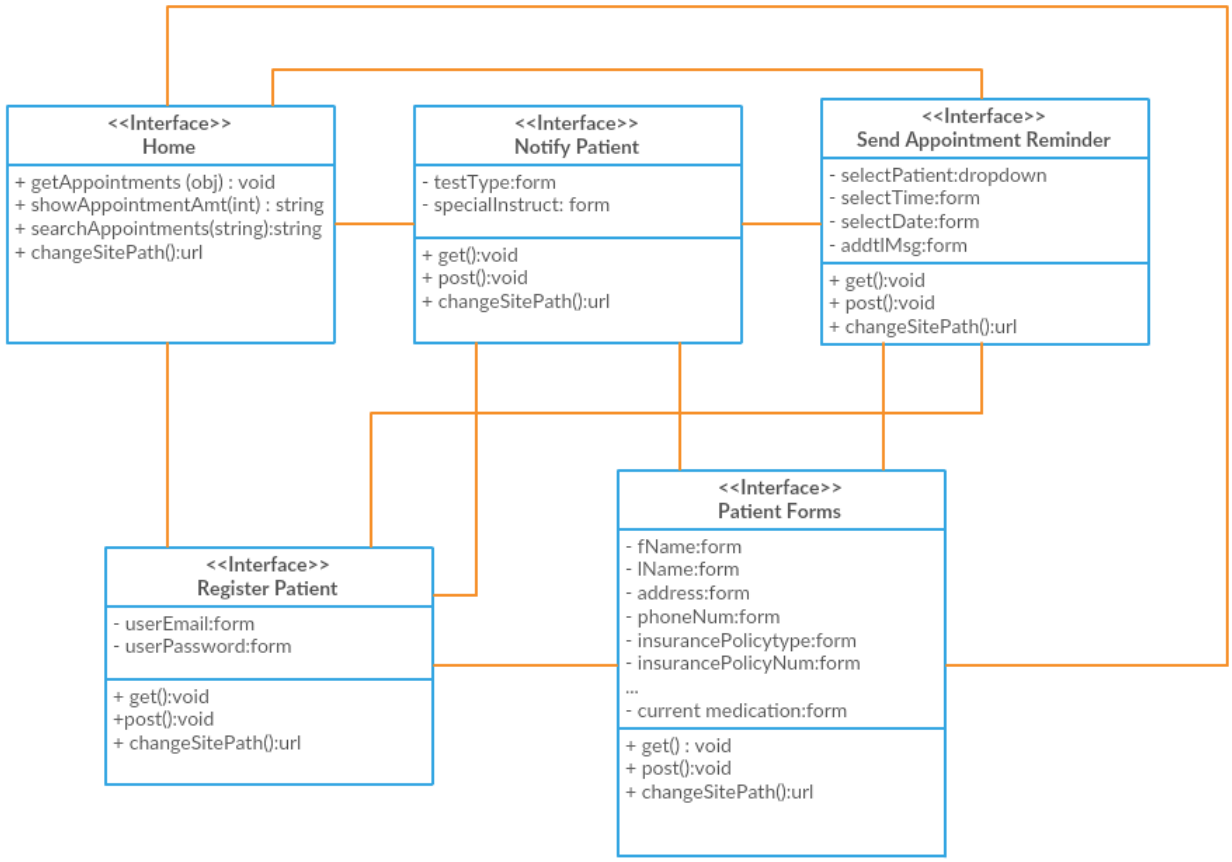
ANDROID SEQUENCE DIAGRAM (TENTATIVE)

Class diagram

Android


Android Class Diagram
(Tentative)
Login
-username:string
-password:string
RequestLogin()
Patient
-firstName:string
-lastName:string
-DOB:DateTime
-phoneNumber:string
-hasFormWaiting:bool
-hasLabRequestWaiting:bool
IsFormWaiting()
IsLabRequestWaiting()
GetLogin()
IsAlert()
Form
-isSubmitted:bool
-currentValues:string
Submit()
DisplayConfirmation()
ClinicInformation
-clinicPhone:string
-clinicAddress:string
-clinicHours:string
-isClinicOpen:bool
CheckForUpdates()
PaperworkForm
-memberName
-currentSection:int
-isComplete:bool
GetNextSection()
LabRequest
-labQR:bmp
-actualForm:string
-patientName:string
-patientDoctor:string
-clinicInfo:string
-patientDOB:string
CheckForUpdates()
BpForm
-bpValue:string
-bsValue:String
-shouldAlert:bool
CheckBPRanges()
CheckBSRanges()
Alert()
SatisfactionForm
-isAfterAppointment:bool
1
n

Web



Used existing Services/API

APIs:

- **Google Maps:** Used to get directions from the user's current location to the clinic in Google Maps.
- **ZXing (Zebra Crossing):** Used to convert lab request information into QRcode.

Widgets:

- **Phone Widget**: When selected, the user can call the clinic using existing phone services on the phone like Google Hangouts Dialer or the Phone Dialer.
- **Email Widget**: When Selected, the user can email the clinic using existing email applications on the phone like E-mail, GM
- **ImageView Widget:** Used to display the QR code after it had been processed by ZXing.
- **TextView Widget:** Used to display information about the information embedded in the QR code.

Services:

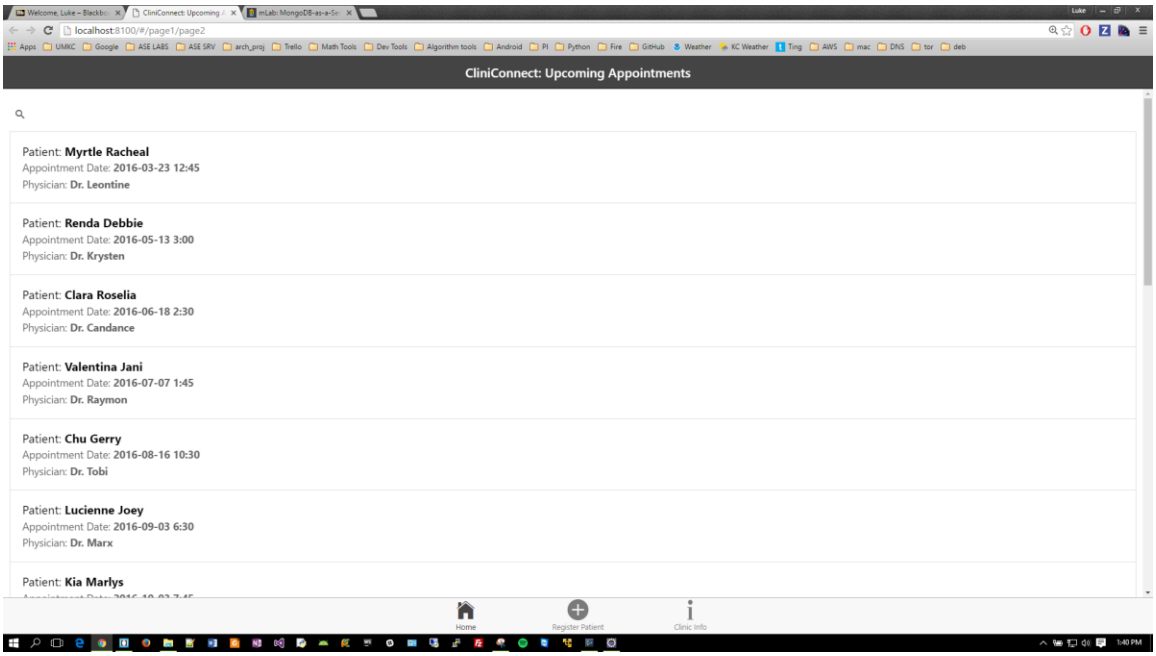
- **Amazon Web Services – EC2:** Used to host admin website.

Implementation and Deployment:

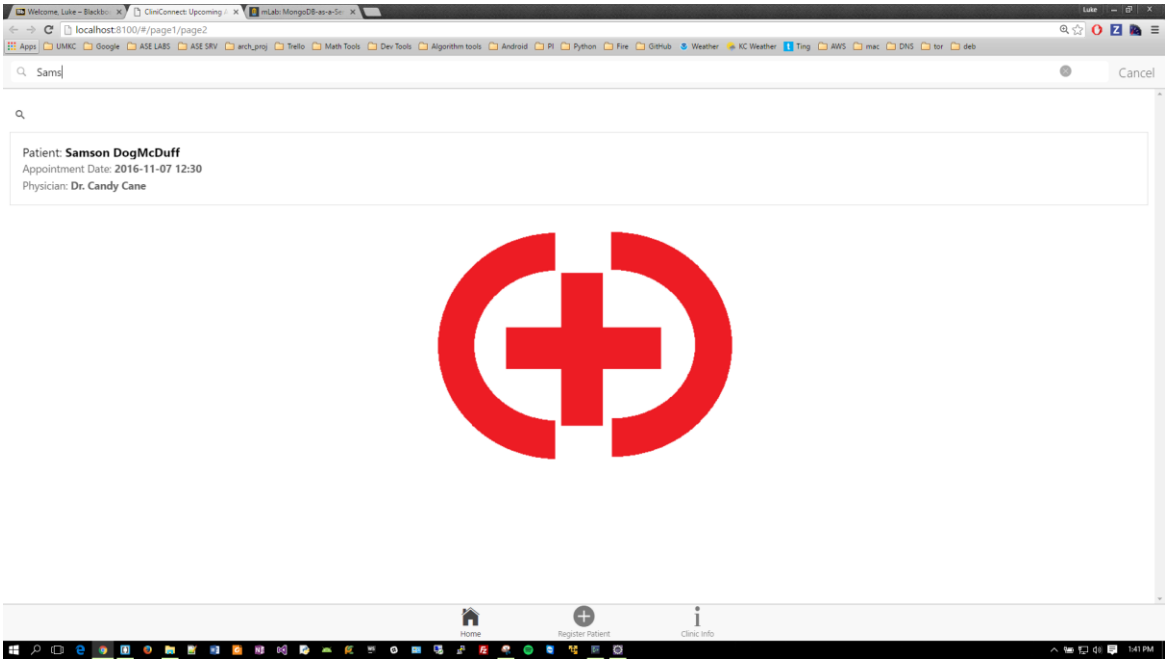
We have implemented an ionic framework for the admin web site, the blood log activity for user and admin sides, and the visit forms for admin and users. The admin website has been implemented with Apache http server on Amazon Web Services. A link to the admin site page. Below are the screenshots.

Web Page Ionic Conversion:

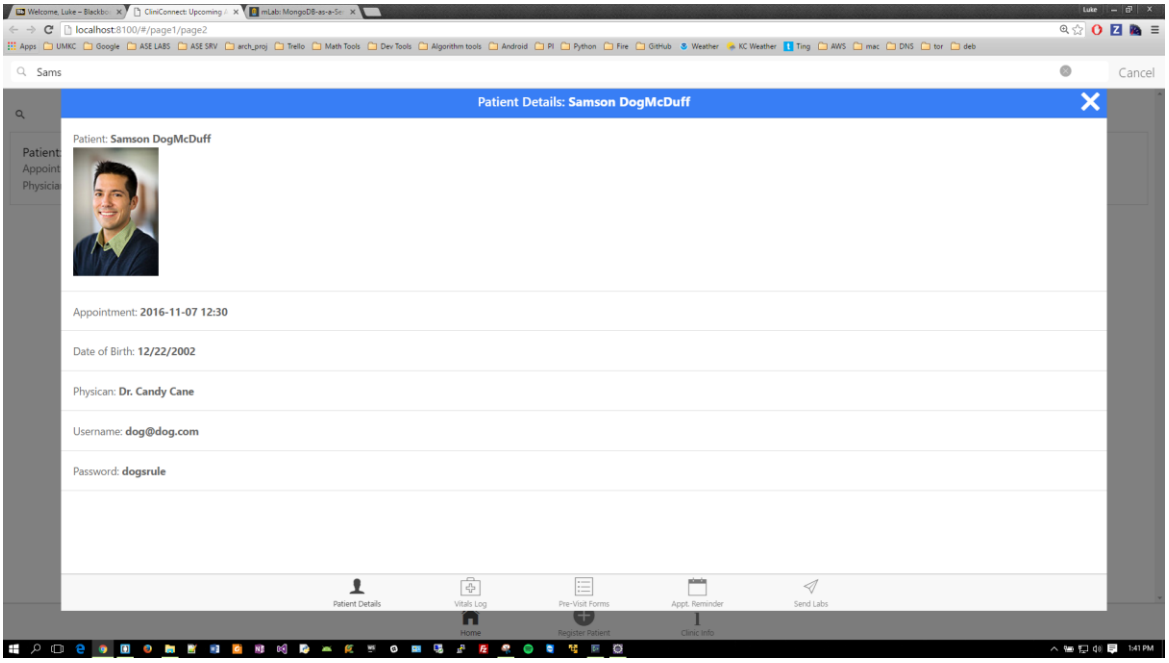
Appointment List:



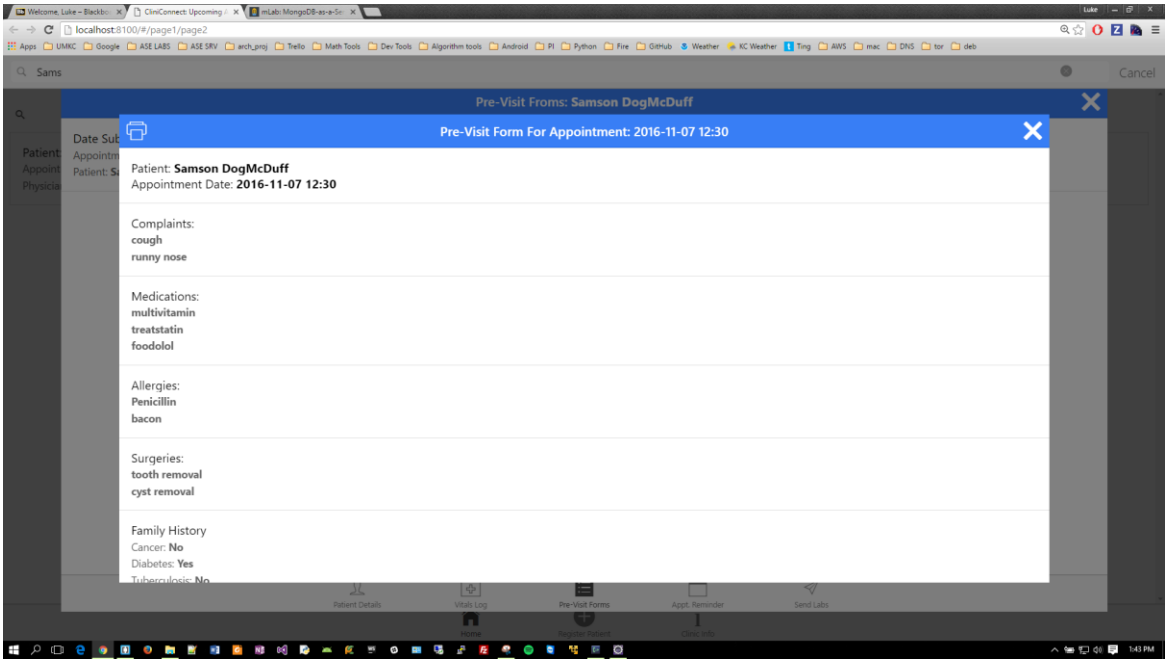
Home search page:



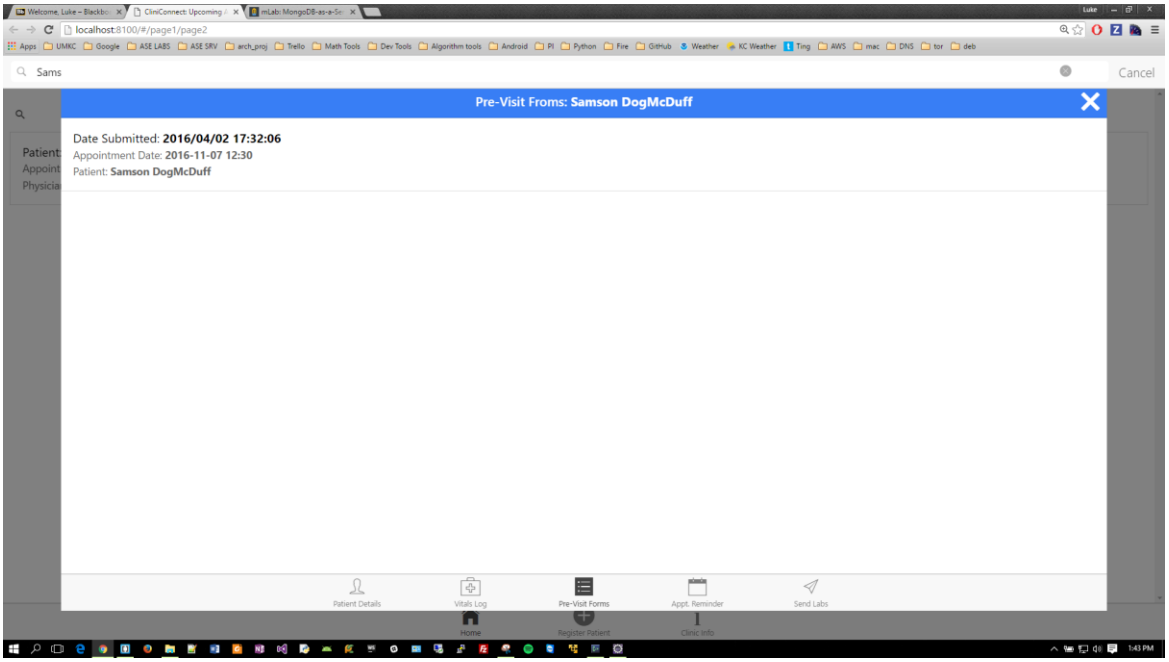
Patient Details:



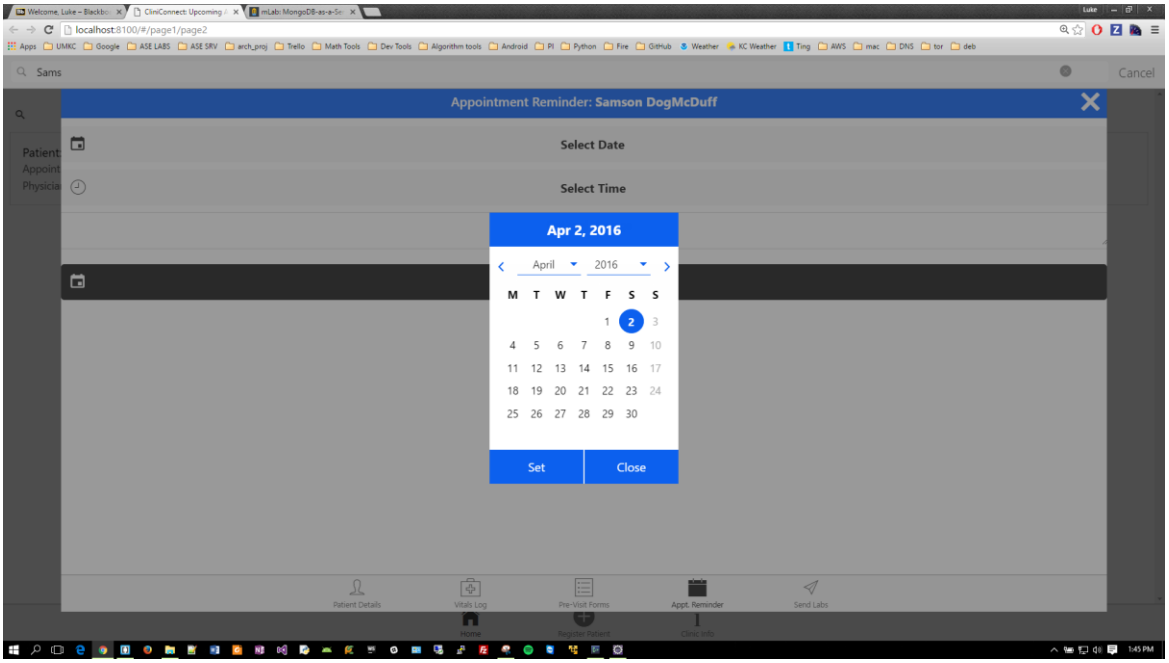
Patient visit form details:



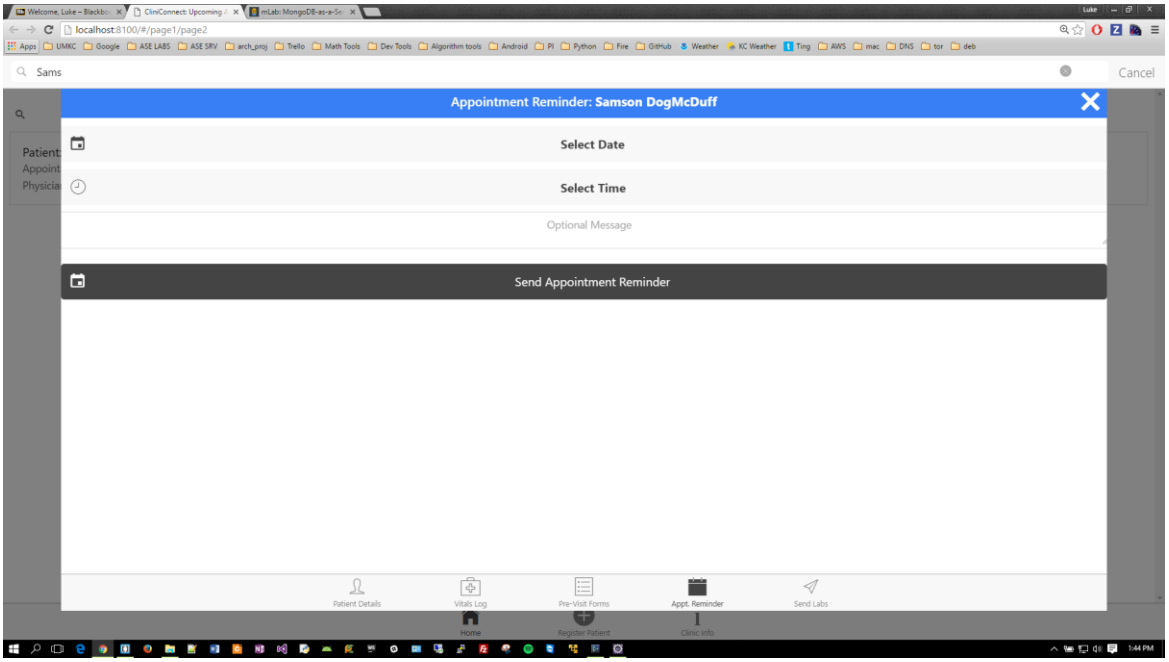
Patient visit form Log:



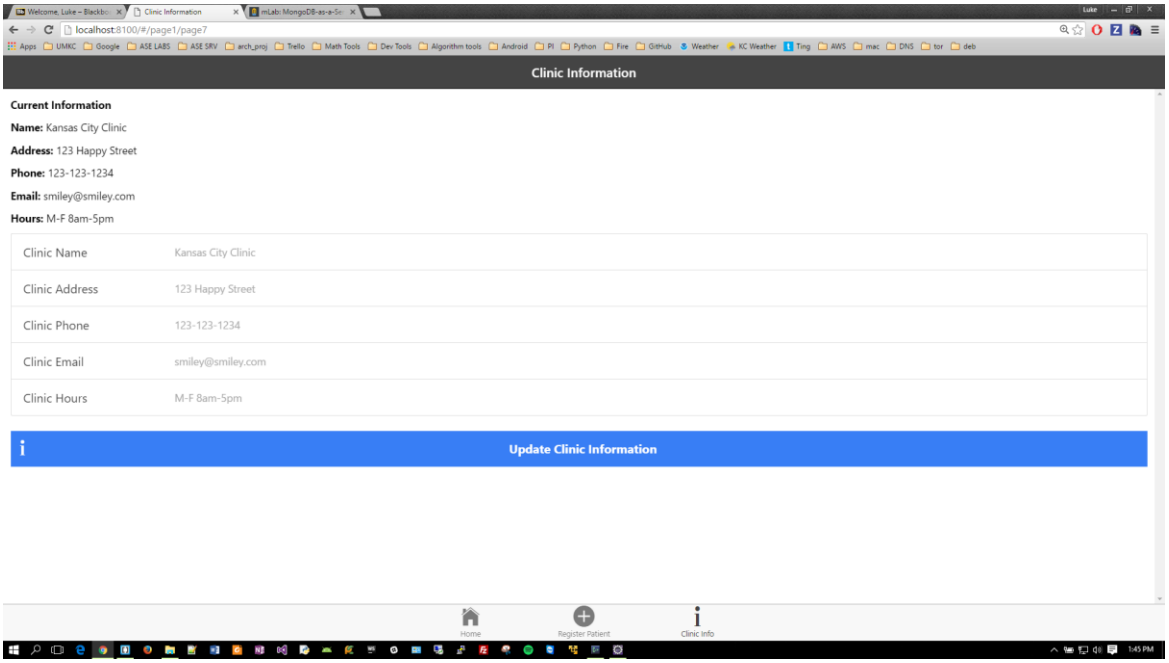
Reminder Date Picker:



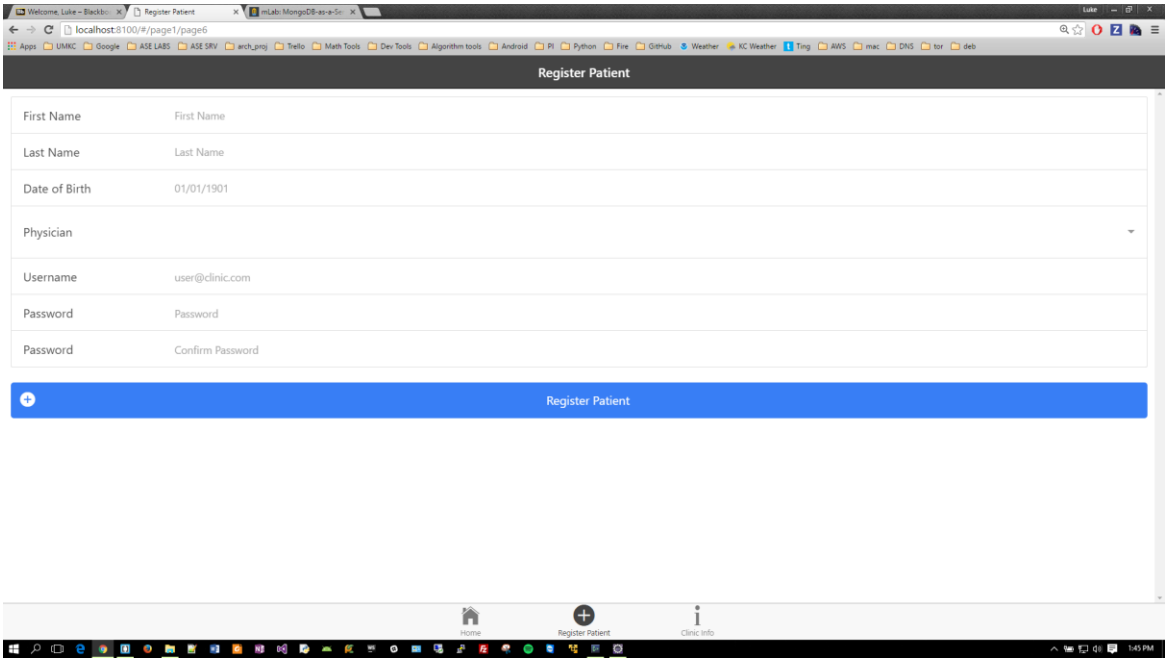
Appointment Reminder:



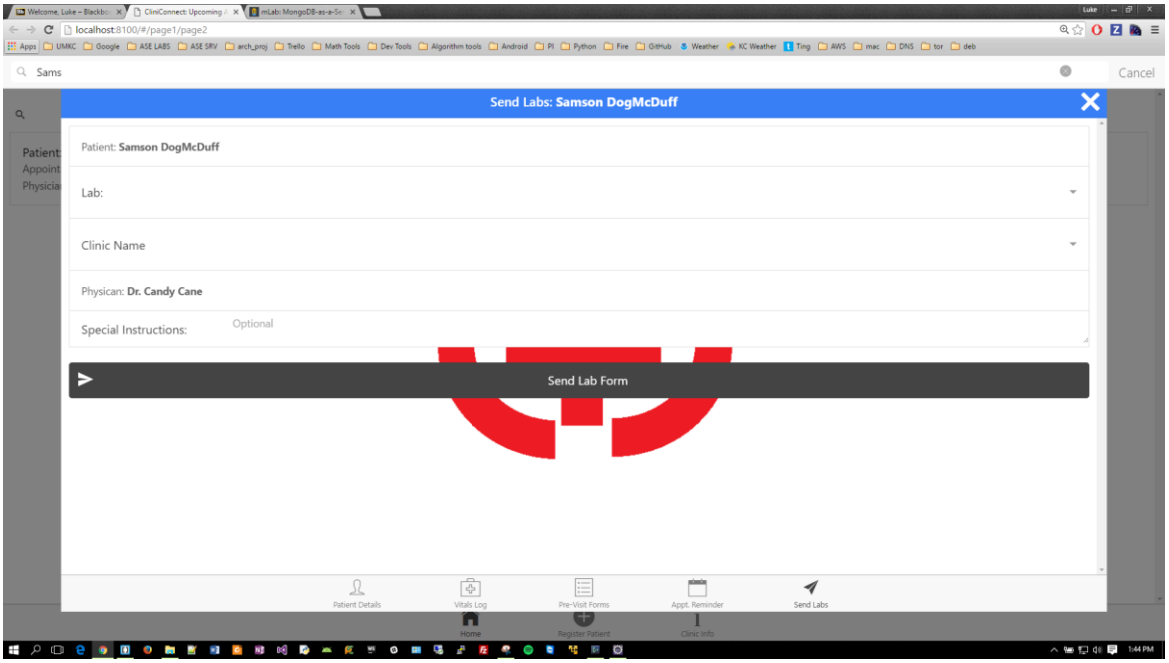
Admin Clinic Information (editable):



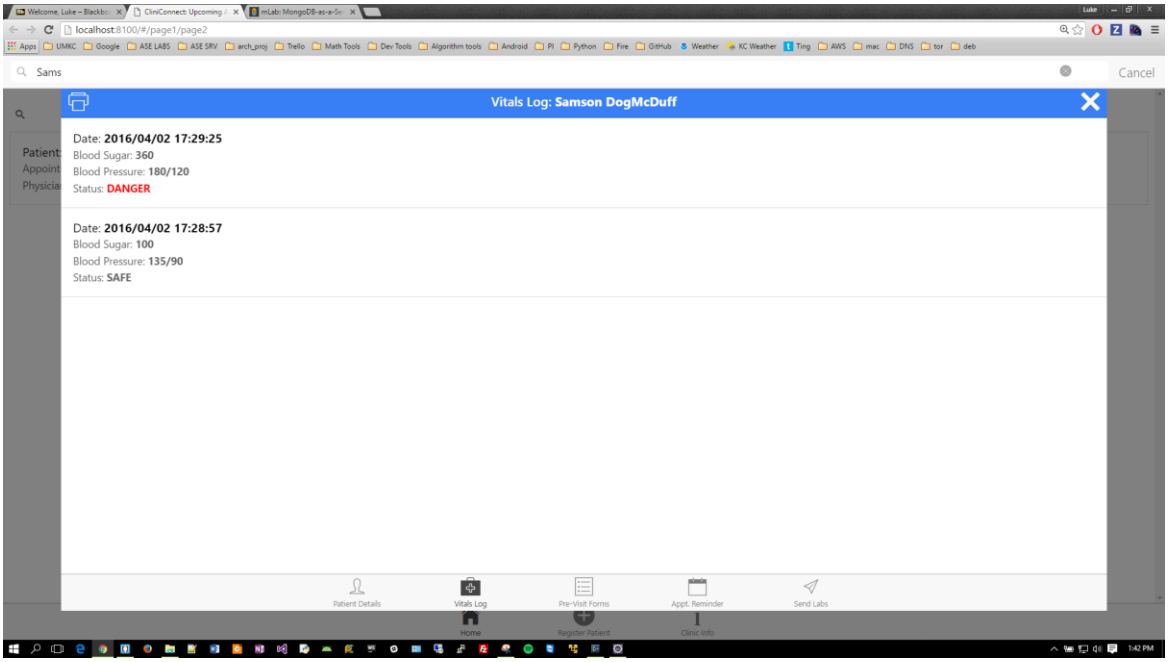
Patient Registration:



Lab Sending:

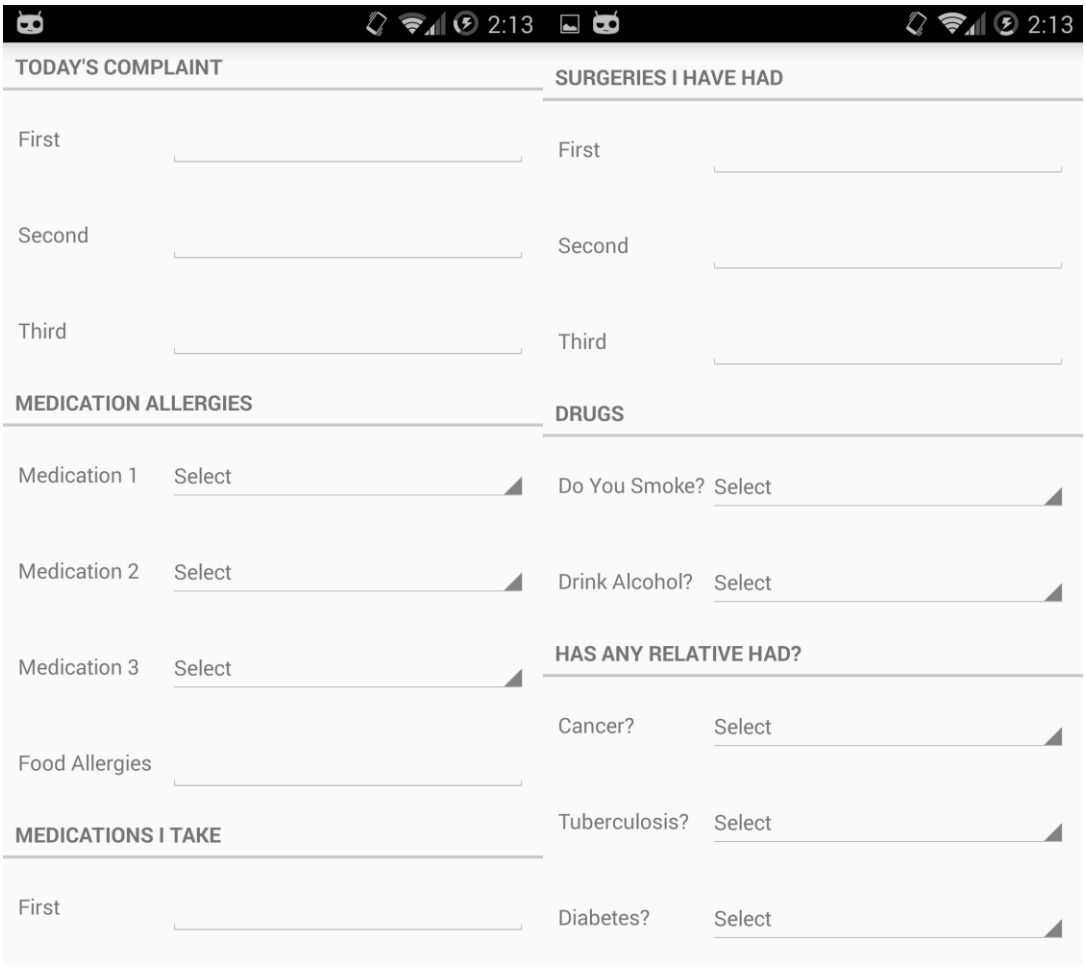


Blood log/Vitals log:

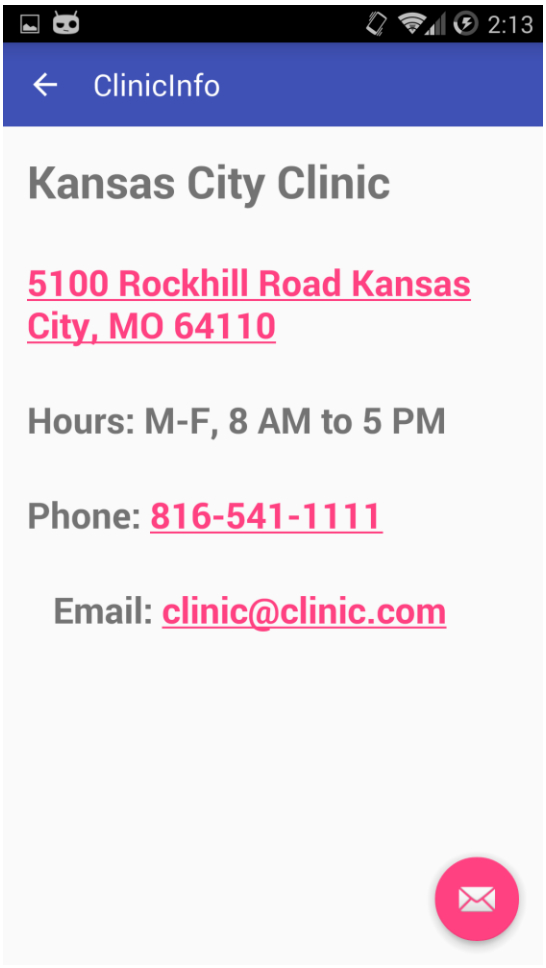


Patient App:

Visit forms pages:



Updated Info page:



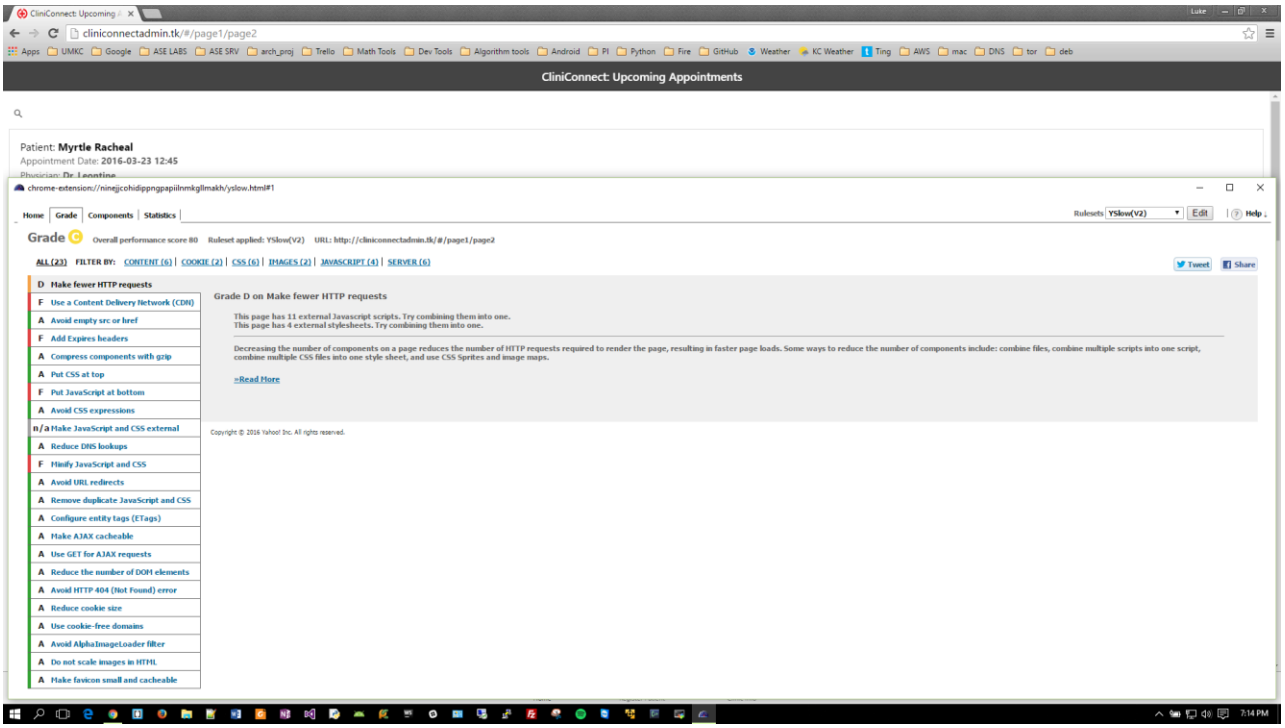
Unit Test Cases:

JUnits:

```java
import static org.junit.Assert.assertEquals;

import org.junit.Test;

public class LevelMonitorTest {

	@Test
	public void isBloodPressureSafeTrueTest() {
		String BP = "120/80";
		LevelMonitor lm = new LevelMonitor();
		assertEquals(true, lm.isBloodPressureSafe(BP));
	}


	@Test
	public void isBloodPressureSafeFalseHighTest() {
		String BP = "145/95";
		LevelMonitor lm = new LevelMonitor();
		assertEquals(false, lm.isBloodPressureSafe(BP));
	}

	@Test
	public void isBloodPressureSafeFalseLowTest() {
		String BP = "85/55";
		LevelMonitor lm = new LevelMonitor();
		assertEquals(false, lm.isBloodPressureSafe(BP));
	}

	@Test
	public void isBloddSugarSafeTrueTest(){
		String BS = "150";
		LevelMonitor lm = new LevelMonitor();
		assertEquals(true, lm.isBloodSugarSafe(BS));
	}

	@Test
	public void isBloddSugarSafeTrueTest(){
		String BS = "150";
		LevelMonitor lm = new LevelMonitor();
		assertEquals(true, lm.isBloodSugarSafe(BS));
	}

	@Test
	public void isBloddSugarSafeFalseHighTest(){
		String BS = "301";
		LevelMonitor lm = new LevelMonitor();
		assertEquals(false, lm.isBloodSugarSafe(BS));
	}

	@Test
	public void isBloddSugarSafeFalseLowTest(){
		String BS = "69";
		LevelMonitor lm = new LevelMonitor();
		assertEquals(false, lm.isBloodSugarSafe(BS));
	}
}
```

```
import static org.junit.Assert.assertEquals;
import org.json.*;
import org.junit.Test;

public class GetClinicInfoTest {

	@Test
	public void buildClinicInfoTest() throws JSONException {
		GetClinicInfo gci = new GetClinicInfo();
		JSONObject info = gci.buildClinicInfo("loc1", "umkc", "555-5555", "a@a.com",
"7-8");

		assertEquals(info.get("clinicName"), "loc1");
		assertEquals(info.get("clinicAddress"), "umkc");
		assertEquals(info.get("clinicPhone"), "555-5555");
		assertEquals(info.get("clinicEmail"), "a@a.com");
		assertEquals(info.get("clinicHours"), "7-8");
	}

}
```

Yslow:



Implementation status report

Work completed:

The work completed in this increment was, Ionic conversion for administration, visit forms(both user and admin sides), blood log(both user and admin sides), architecture and activity diagrams, and web hosting were completed.

Responsibility (Task, Person)

In this increment, work was divided into tasks to use time efficiently. The task of the blood log went to Shweta, implementing the blood log for both user and admin. Sri handled the forms for both user and admin. Ben, designed the architecture and activity diagrams and wrote the report. Luke implemented the ionic framework conversion and set up the web hosting, as well as performing various tests.

Work completed and work to be completed:

Everything has been updated in zenhub please visit our zenhub link below.

https://github.com/ljm7b2/ASE_PROJECT_G1#boards?repos=50469210

Bibliography

- Wireframes design http://creately.com/
- Testing Information:
    - http://developer.android.com/tools/testing-support-library/index.html
    - http://yslow.org
- Android Studio http://developer.android.com/sdk/index.html
- Google Maps API https://developers.google.com/maps/

Github: https://github.com/ljm7b2/ASE_PROJECT_G1