Team Members

[Venkatesh Bonageri](https://uncc.instructure.com/groups/17907/users/53827)

Sai Nishanth Dilly

Vikas Deshpande

***Abstract***

Health care is one of the very important industries in today’s world. The advancements happening in the world of technology have affected health care industries too and a lot of interesting events are happening in this field. SmartCare is an android application implemented using node.js, Research Stack, Android features and the web api’s are hosted on amazon aws server. This application sends notifications to users reminding/letting them know about a question or a survey which has been posted to the study group they are part of. Admin creates Study Coordinators and Users where in Study coordinators create users and studies.

***Introduction***

This application depends on the server which we have hosted on amazon aws and the android application. We used research stack for creating survey purposes, and all the api’s are written either in node.js or php.

Features Description:

***Admin can***

* Can create new user/patient
* Can create new study co-Ordinator
* Can view the list of existing users
* Can view the list of existing study co-Ordinator

***Study Co-Ordinator can***

* Can create new user/patient
* Can view the list of existing users
* Can create a new study
* Can view the list of existing studies
* Can and the users to the created studies
* Can design the study by adding the questions of both informative types and multiple choice questions
* Created questions can be scheduled to be sent
* Can create surveys in the studies with 3 types of questions open text, linkern type and multiple choice questions
* Can view the responses of the questions provided by users.
* Can view the responses of the surveys of each user
* Can print the responses in the printable format.

***Client Side Features***

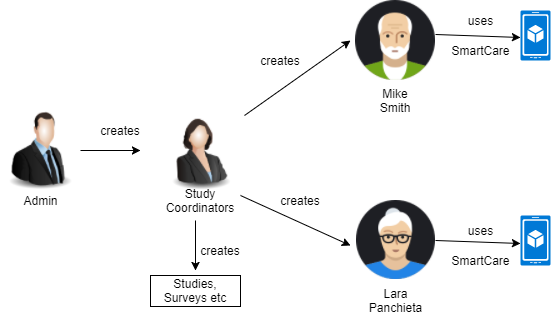
* Will receive the notification once a new question is added to the study
* Can opt-in or opt-out from receiving the notifications
* Will list the list of studies that has been enrolled
* Will show the list of questions in each study
* Can submit the responses for each mcq questions
* Can take the survey and submit the responses

**Features listed above are all the tasks which are completed.**

Tasks not completed:

* Sending notifications on creation of a survey
* Opening the study directly on clicking of the notification
* Improvisation on UI

***Design Architecture***



**Database Schema**

The tables created are,

**Admin table to store admin username and password.**

CREATE TABLE `admin` ( `username` varchar(50) NOT NULL, `password` varchar(50) NOT NULL ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

**Gcmids table is to store device ids of android phones and the email id of person using it.**

CREATE TABLE `gcmids` ( `devid` varchar(500) NOT NULL, `uemail` varchar(50) NOT NULL ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

**Questions table is to store question id and question text.**

CREATE TABLE `questions` ( `sid` int(11) NOT NULL, `qid` int(11) NOT NULL, `question` varchar(1000) DEFAULT NULL ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

**QuestionsOptions is to store the options given for a question, option id and option name**

CREATE TABLE `questionsOptions` ( `qid` int(11) NOT NULL, `optionid` int(11) NOT NULL, `optionname` varchar(50) DEFAULT NULL ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

**QuestionsResponse is to store question id, timeid, optionid, email.**

CREATE TABLE `questionsResponse` ( `qid` int(11) NOT NULL, `timeId` int(11) NOT NULL, `uemail` varchar(50) NOT NULL, `optionid` int(11) DEFAULT NULL ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

**QuestionsTime table stores qid,time in hours, time in minutes, isscheduled**

CREATE TABLE `questionsTime` ( `qid` int(11) NOT NULL, `qtimehh` int(11) NOT NULL, `qtimemm` int(11) NOT NULL, `isscheduled` int(11) NOT NULL DEFAULT '0' ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

QuestionsTimePosted stores question id, timeid, and date time.  
CREATE TABLE `questionsTimesPosted` ( `qid` int(11) DEFAULT NULL, `timeId` int(11) NOT NULL, `dtime` datetime DEFAULT CURRENT\_TIMESTAMP ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

QuestionsUsers stores qid, uemail.  
CREATE TABLE `questionsUsers` ( `qid` int(11) NOT NULL, `uemail` varchar(50) NOT NULL ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

Scheduler is to schedule push notifications which consists of id, hh and mm

CREATE TABLE `scheduler` ( `id` int(11) NOT NULL, `hh` int(11) DEFAULT NULL, `mm` int(11) DEFAULT NULL ) ENGINE=InnoDB DEFAULT CHARSET=utf8;

Studies table consists of sid, sname, sdescription

CREATE TABLE `Studies` ( `sid` int(11) NOT NULL, `sname` varchar(255) NOT NULL, `sdescription` varchar(255) DEFAULT NULL ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

StudiesUsers table consists of sid, uemail  
CREATE TABLE `StudiesUsers` ( `sid` int(11) NOT NULL, `uemail` varchar(50) NOT NULL ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

StudyCoordinator consists of his email, first name, lastname, password  
CREATE TABLE `StudyCoordinator` ( `semail` varchar(50) NOT NULL, `fname` varchar(50) DEFAULT NULL, `lname` varchar(50) DEFAULT NULL, `pwd` varchar(100) DEFAULT NULL ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

Survey table consists of surveyid, sid, surveyname,dtime

CREATE TABLE `Survey` ( `surveyid` int(11) NOT NULL, `sid` int(11) NOT NULL, `surveyname` varchar(50) NOT NULL, `dtime` datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

Survey questions table consists of sid, surveyed, qid, questionName, questionType.  
CREATE TABLE `surveyquestions` ( `sid` int(11) NOT NULL, `surveyid` int(11) NOT NULL, `qid` int(11) NOT NULL, `questionname` varchar(1000) DEFAULT NULL, `questiontype` varchar(100) DEFAULT NULL ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `surveyquestionsOptions` ( `qid` int(11) NOT NULL, `surveyid` int(11) NOT NULL, `optionid` int(11) NOT NULL, `optionname` varchar(50) DEFAULT NULL ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `surveyquestionsResponse` ( `qid` int(11) NOT NULL, `surveyid` int(11) NOT NULL, `responseText` varchar(100) DEFAULT NULL, `uemail` varchar(50) NOT NULL, `optionid` int(11) DEFAULT NULL ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `USERS` ( `uemail` varchar(50) NOT NULL, `fname` varchar(50) DEFAULT NULL, `lname` varchar(50) DEFAULT NULL, `pwd` varchar(100) DEFAULT NULL, `age` int(10) DEFAULT NULL ) ENGINE=InnoDB DEFAULT CHARSET=latin1;

***API Descriptions:***

All the api’s listed below are written in node.js.

1. /addidforgcm - adds gcm ids to devices. This api is important in receiving push notifications.
2. /deleteidforgcm – deletes the gcm ids if the user chooses not to receive notifications.
3. /sendnotification - this api sends notifications to all the users selected by the admin. We have used google’s cloud messaging service.
4. /getQuestions - This api is written in php, since to get questions we had to make inner joins of multiple tables, node.js code was getting messy and we were in callback hell. We used promises library in node.js and tried but were partially successful. So, this is written in php.
5. /submitresponse – This api is to submit responses from the user to server.
6. /getuserprofile – This api is to display users profile in android app.
7. /displayQuestionsToAdminPromise – Admin has an ability to check messages/questions sent to the user. We have used promise library to do the same.
8. /resetscheduler – This api is to schedule questions to the user
9. /viewresponses – Api to view user responses
10. /schedulemessages – Api to schedule messages to be sent to user
11. Createinformationquestion – Creates the information message in the study
12. Createmcqquestion – Creates the mcq question in the study
13. getUsers – lists the users in the study
14. createuser – creates a patient account
15. userlogin – for the validation of user login from the app
16. getSurveys- lists the surveys for the study
17. getEachSurveyUsers – lists the users for the survey selected
18. getEachSurveyResult – gets the response submitted for the survey by the user
19. addSurveyQuestions – adds the survey questions to the database for the selected study
20. getQuestionsForResponse -lists the questions that require users response
21. getQuestionResponseForAll – gets the response for the selected question for all users
22. login – login for the study coordinator
23. getUsersStdyFirstTime – lists all users to add into the study
24. getUsersForStudy – lists the current users in the study
25. createstudyusers – adds the user to the study
26. getStudies – lists the existing studies
27. createstudy – creates the study
28. getQuestionsForMobile - gets the questions for the current study
29. submitresponse – submits the response for each question
30. getuserprofile – gets the user profile
31. addSurveyAnswers – submits the survey responses for the user
32. getSurveyQuestions – gets the survey questions for the selected survey

***Tasks undertaken and completed by different team members***

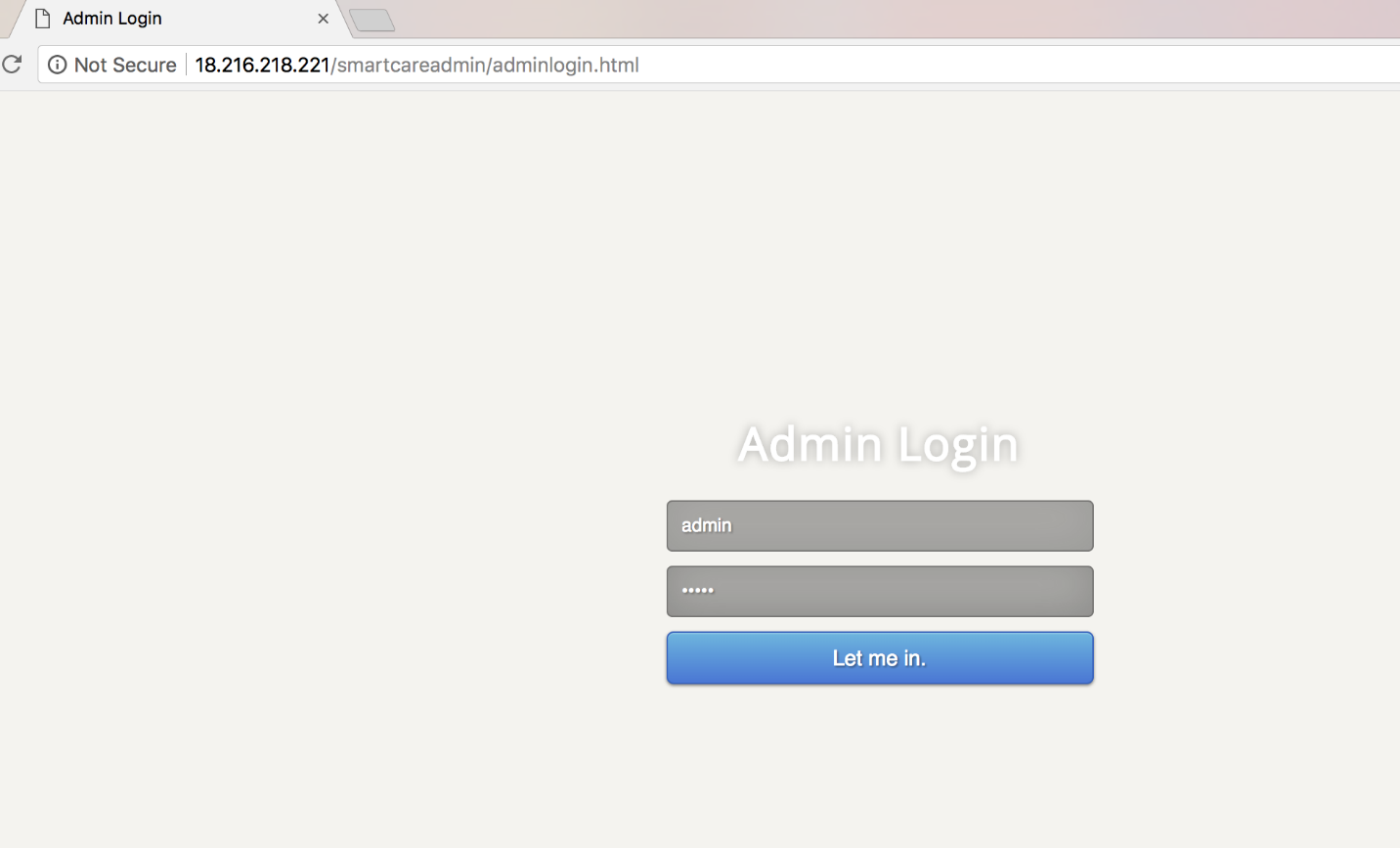
To make sure everyone in the team works on all areas we divided the work in such a way that, all of us worked on building api’s in node,js, front end in angular.js and android app development.

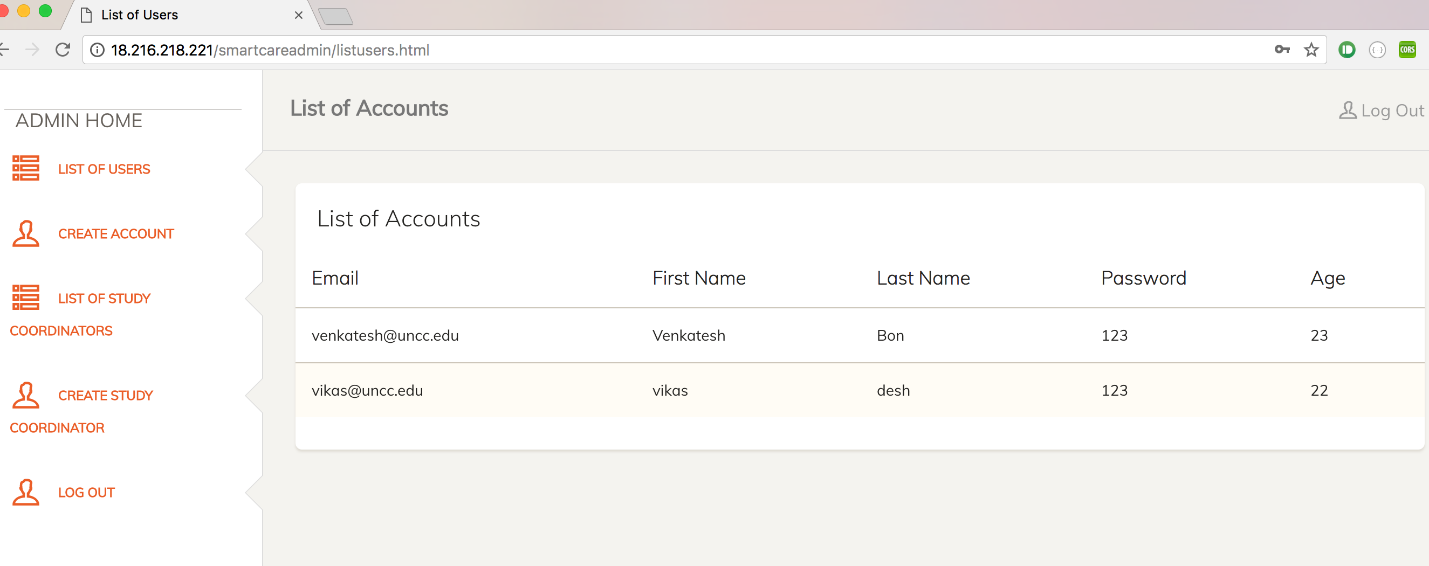
|  |  |
| --- | --- |
| Name | Tasks Completed |
| Sai Nishanth Dilly | Node api’s for adding questions, admin and study coordinator login, adding users, scheduling web portal front end for some parts using angular.js, database schema design, modularizing code into proper structure and files, android , schedulemessages node api, createstudy, Createinformationquestion, resetscheduler getEachSurveyResult |
| Venkatesh Bonageri | Node api’s for creating surveys, getting studies list, submitting survey responses adding gcm id’s, handling push notifications, database schema design, front end angular.js for some files, navigation bar and hamburger icon., getEachSurveyUsers node api’s, addSurveyQuestions , addidforgcm,getQuestionResponseForAll, Createmcqquestion |
| Vikas Deshpande | Node api’s for creating getting questions of a particular survey, adding users to particular survey, hosting in aws, database schema, learning research stack and implementation, android ui, angular.js some web portal files development, displaying questions to admin, getQuestionsForMobile, viewresponses, getStudies |

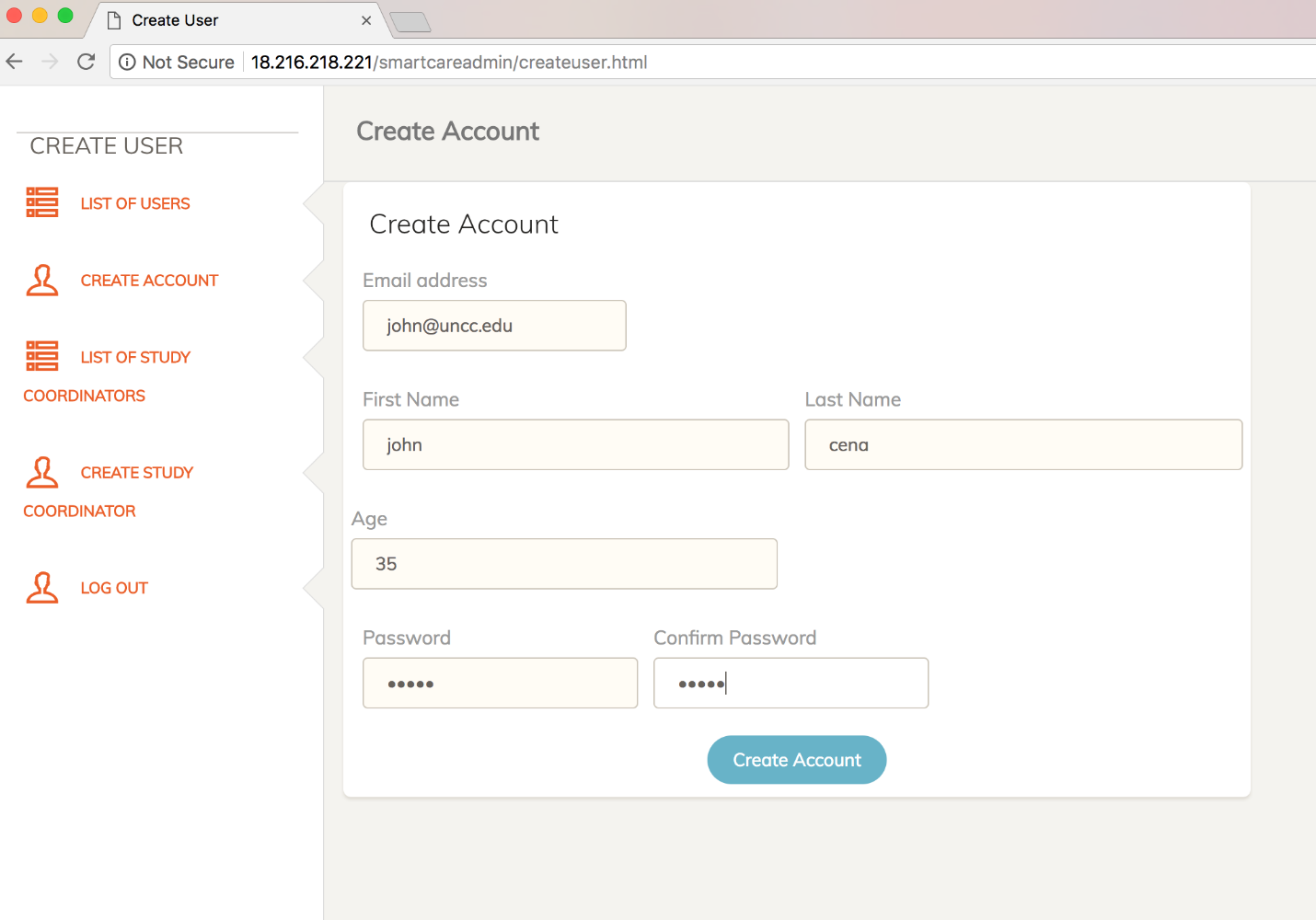
ScreenShots:

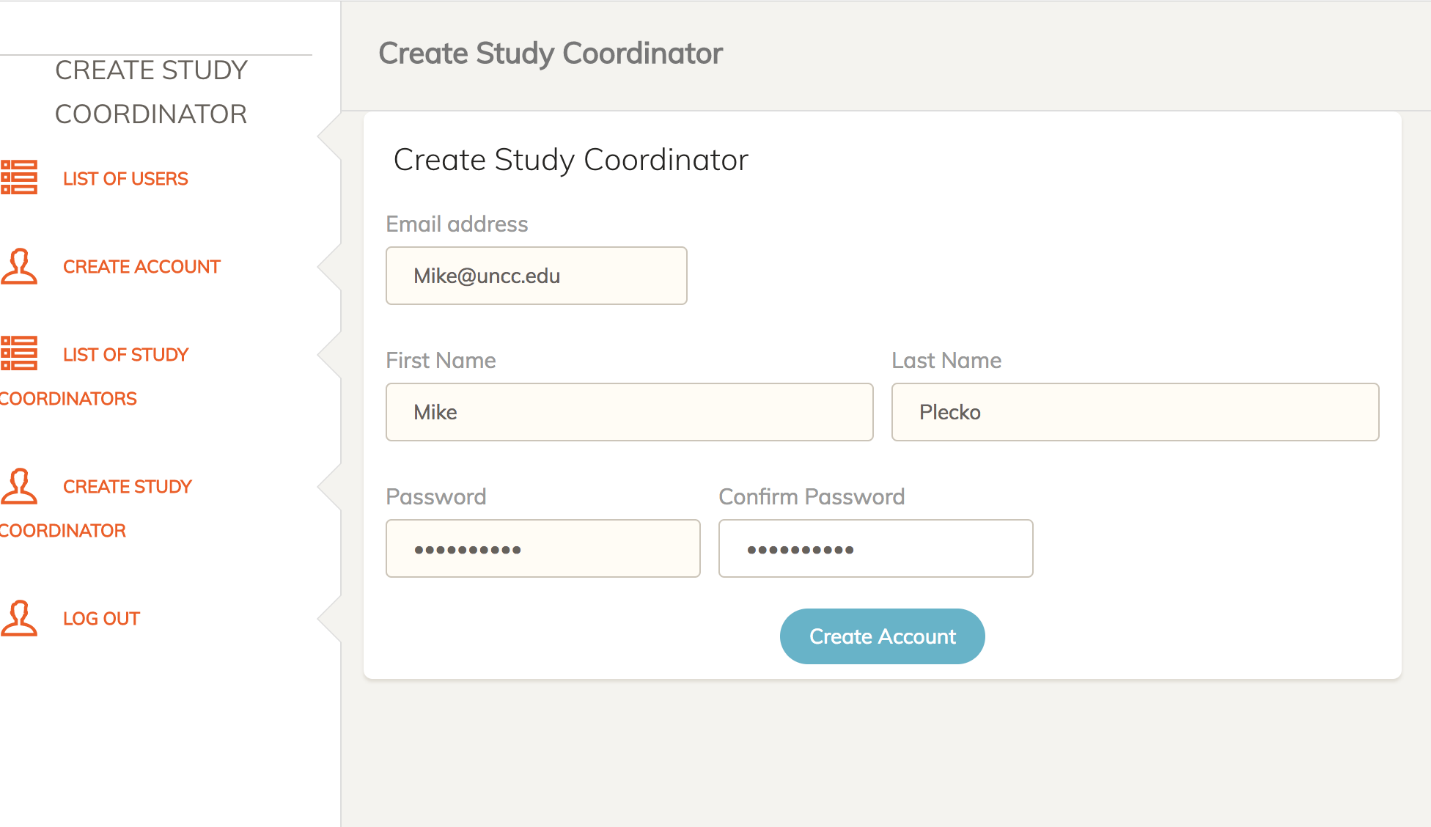
Web Portal:

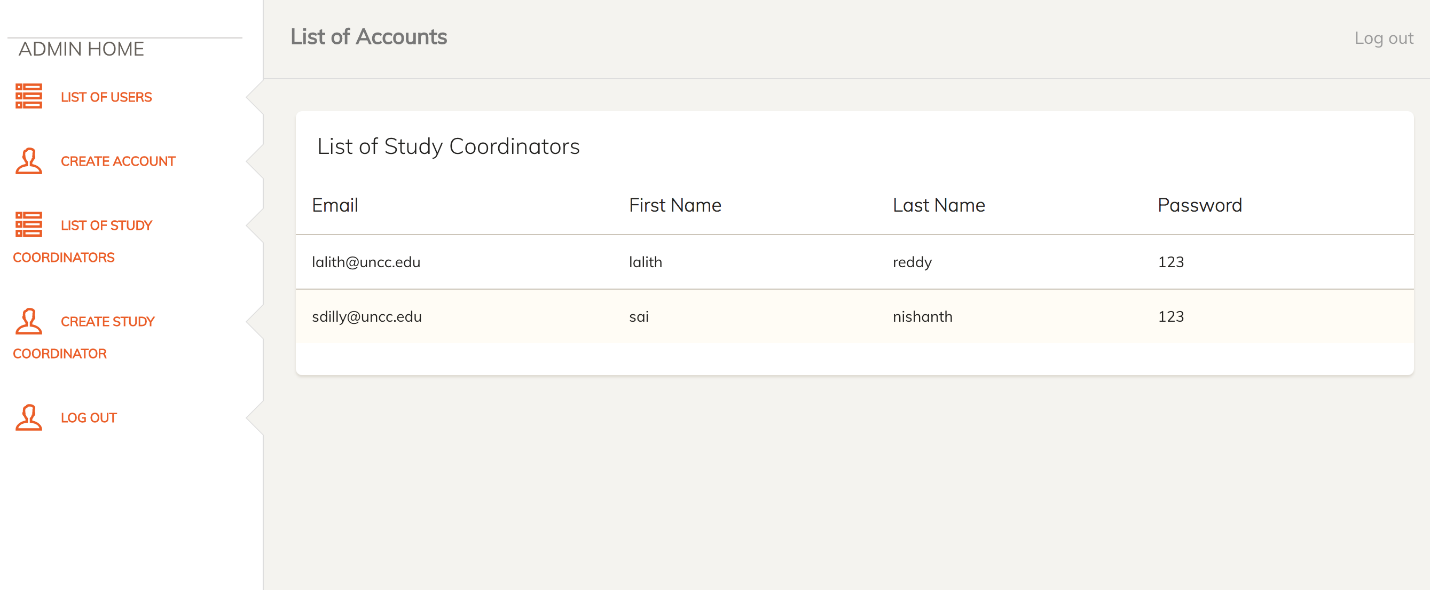
Admin Login Page.











For study coordinators:

