SRH Hochschule Heidelberg

Advanced Computer Science

HOPEFitness

Supervisor: Mr. Christoph Hahn

Submitted By: Nagashree Tadapatri
11007940

Date: 02-08-2016

Table of Contents

INTRODUCTION

The HOPEFitness app fulfils the objectives of Advanced Computer Science module by having employed a robust software development practice which makes it scalable and deployable on multiple environments; such as a mobile app or a web app.

Since its inception in the Software Application Development module, the prototype of HOPEFitness has undergone multiple changes, and the end product can accommodate a range of users, poses scope to work on multiple devices and is extremely user-friendly.

The vitalities of Agile software development practice has been explored in the Advanced Computer Science module. The planning and estimation that was done during the Software development practice has been realized in this module. While Agile development practices were employed during the planning phase, a simple Waterfall model was employed during development of the application to favour a single developer environment.

OBJECTIVE

The objective of HOPEFitness app is to build a platform that brings together fitness enthusiasts, fitness trainers and nutritionists. This app helps fitness enthusiasts find the best trainers and nutritionist that fit their goals and budgets, and helps trainers and nutritionists find new customers and keep track of their records with great ease.

This mobile app is built using Ionic framework and AngularJS. The data store employed for this app is Firebase. The app also integrates to Google APIs to fetch user's current location.

The app has a simple user interface with precise information, mostly easy to comprehend icons. All views are maintained on the same level, so as to enable quick navigation from any view to another.

The mobile app is tested OK on Android platform using Ionic lab, Genymotion emulator and an android device.

FEATURES

Following are the features of HOPEFitness app.

<u>For Fitness Enthusiasts</u>

- Provides a platform for fitness enthusiasts to find trainer and nutritionists based on their demands, such as location preferences, specialization of the trainer, etc.
- Allows the fitness enthusiast to subscribe to trainers and make appointment with nutritionists.
- Allows exchange of messages with trainers and nutritionists.
- Provides 'Edit my profile' features to update profile.
- Provides register, login and logout features.

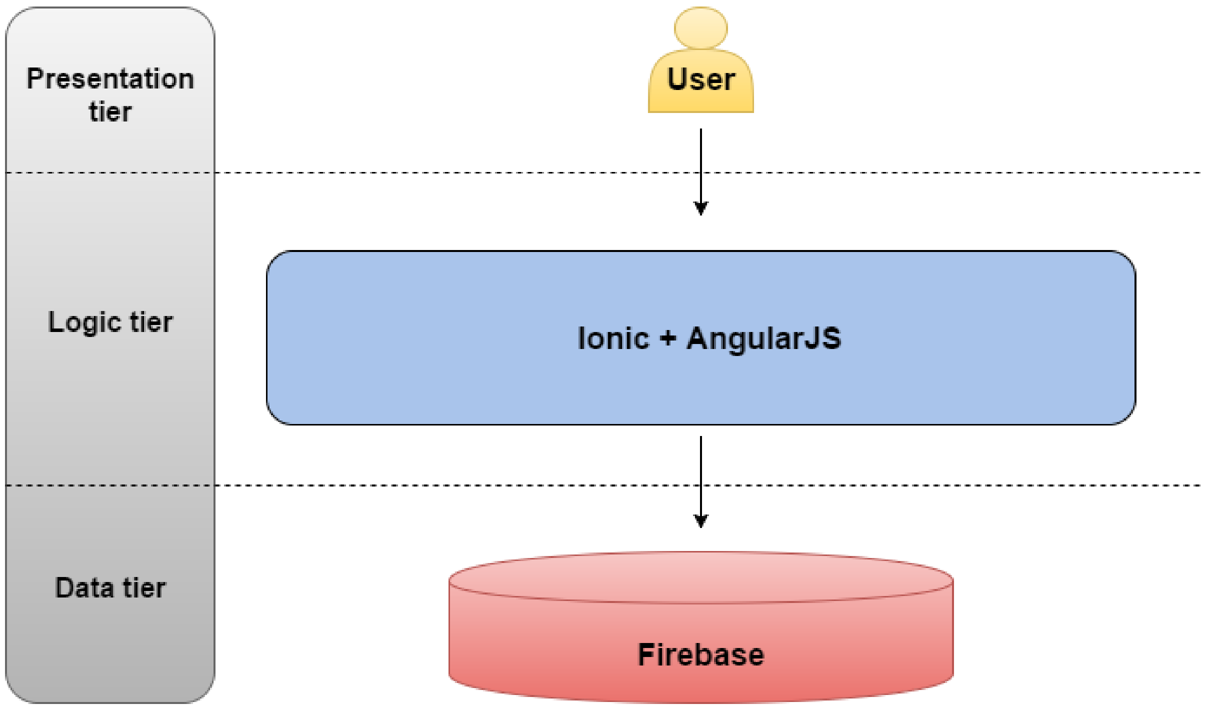
For Fitness Trainers

- Allows a trainer to view Subscriptions from his users.
- Allows a trainer to post advertisements which will be made available to the prospective users.
- Allows a trainer to exchange messages with users.
- Allows basic functionalities like register, login, update profile and logout.

For Nutritionists

- Allows a nutritionist to view Appointments requested by patients.
- Allows a nutritionist to post advertisements which will be made available to the prospective patients.
- Allows a nutritionist to exchange messages with patients.
- Allows basic functionalities like register, login, update profile and logout.

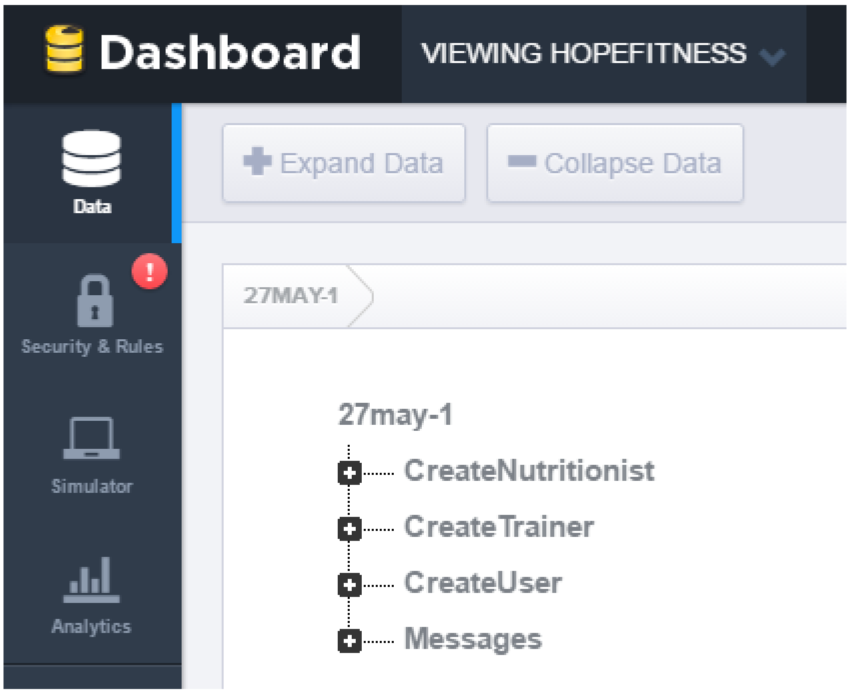
SYSTEM ARCHITECTURE



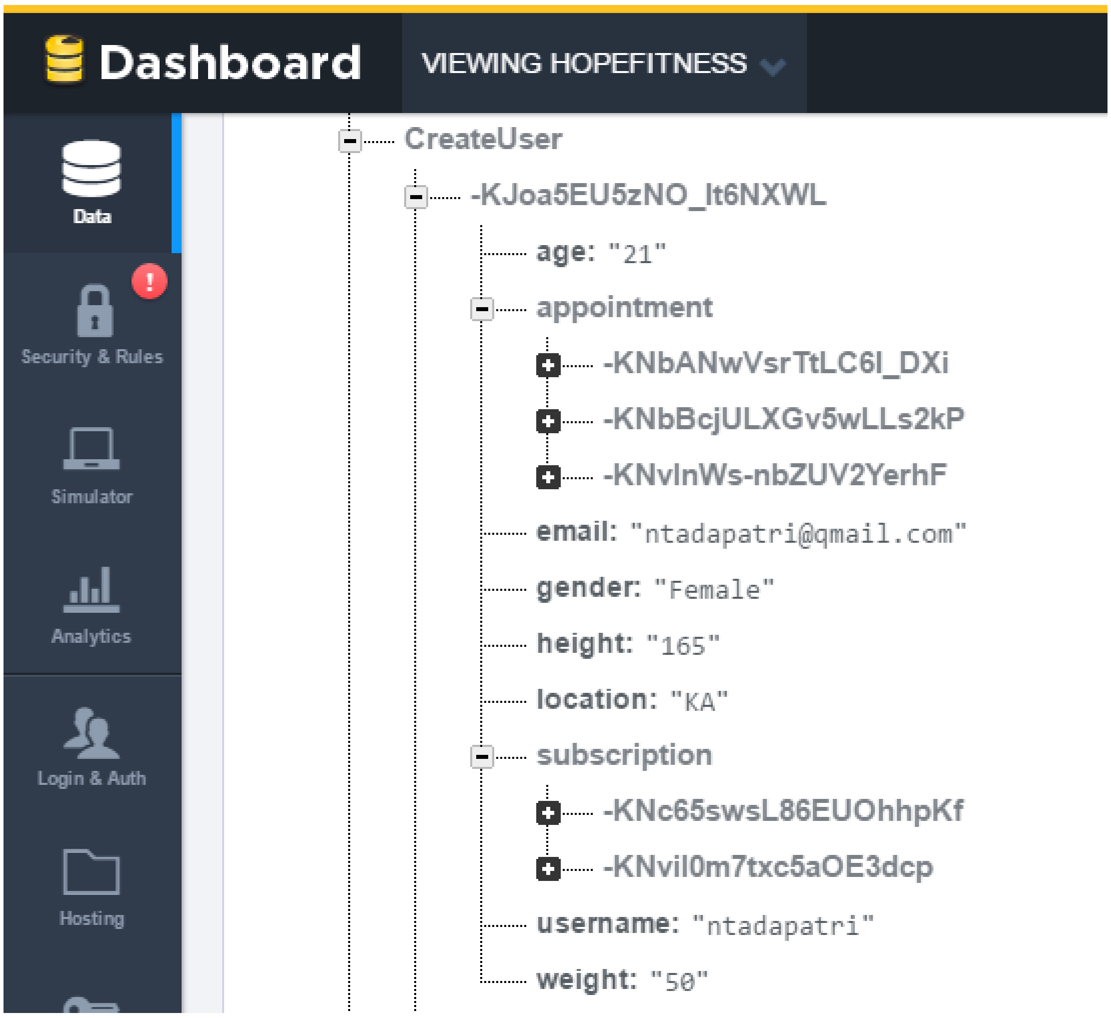
DATA MODEL

The data store used for this application is Firebase.

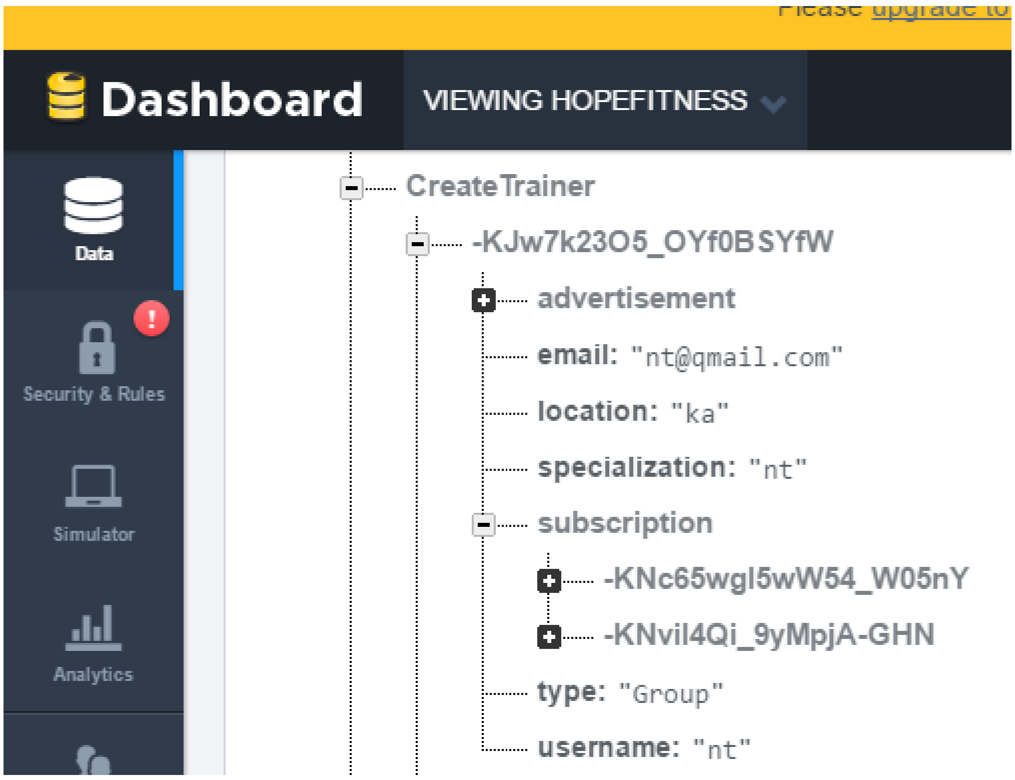
Following screen captures show the data model at different levels of data structuring.



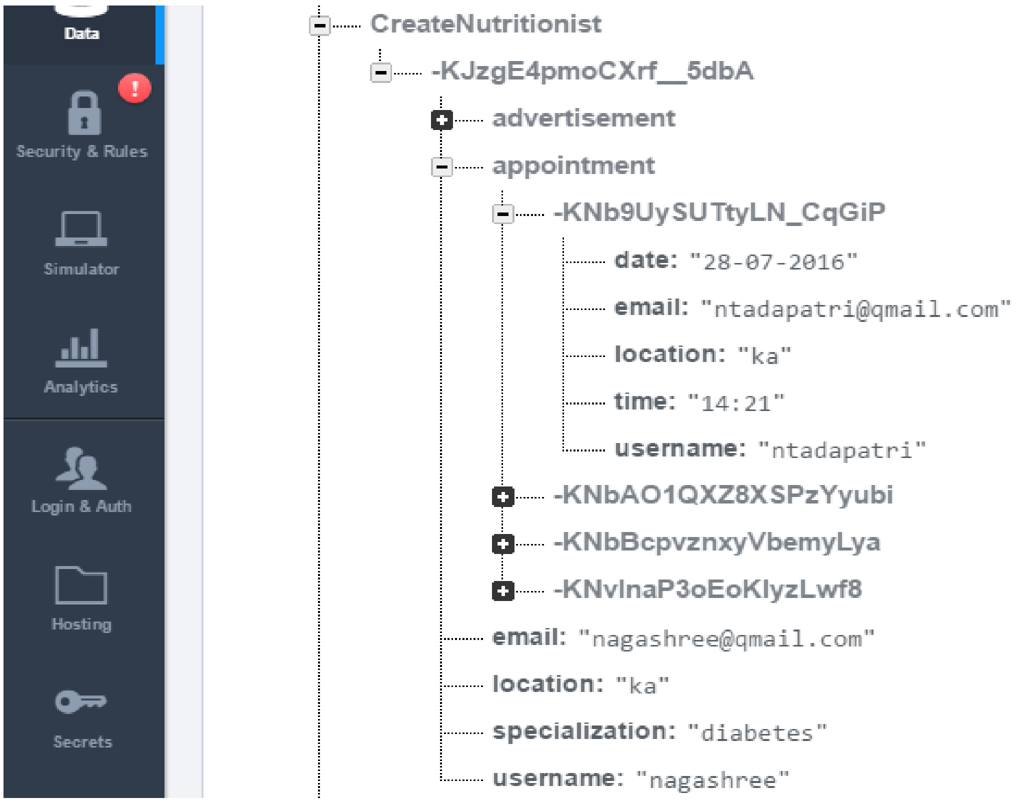
```
27may-1
├── CreateNutritionist
├── CreateTrainer
├── CreateUser
└── Messages
```



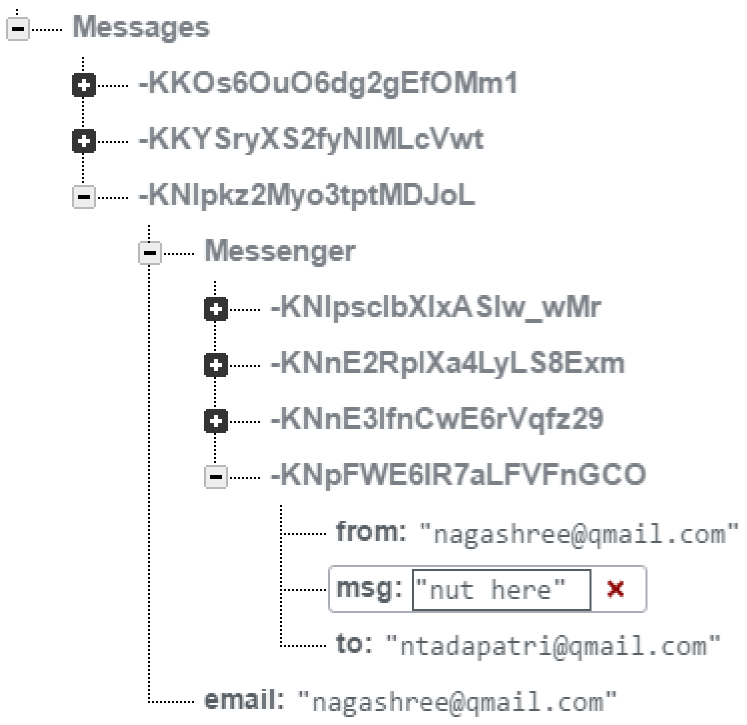
```
CreateUser
└── -KJoa5EU5zNO_It6NXWL
    ├── age: "21"
    ├── appointment
    │   ├── -KNbANwVsrTtLC6I_DXi
    │   ├── -KNbBcjULXGv5wLLs2kP
    │   └── -KNvInWs-nbZUV2YerhF
    ├── email: "ntadapatri@gmail.com"
    ├── gender: "Female"
    ├── height: "165"
    ├── location: "KA"
    ├── subscription
    │   ├── -KNc65swsL86EUOhhpKf
    │   └── -KNvil0m7txc5aOE3dcp
    ├── username: "ntadapatri"
    └── weight: "50"
```



```
CreateTrainer
  -KJw7k23O5_OYf0BSYfW
    advertisement
    email: "nt@qmail.com"
    location: "ka"
    specialization: "nt"
    subscription
      -KNc65wgl5wW54_W05nY
      -KNvil4Qi_9yMpjA-GHN
    type: "Group"
    username: "nt"
```



```
CreateNutritionist
  -KJzgE4pmoCXrf__5dbA
    advertisement
    appointment
      -KNb9UySUTtyLN_CqGiP
        date: "28-07-2016"
        email: "ntadapatri@qmail.com"
        location: "ka"
        time: "14:21"
        username: "ntadapatri"
      -KNbAO1QXZ8XSPzYyubi
      -KNbBcpvznxyVbemyLya
      -KNvlnaP3oEoKlyzLwf8
    email: "nagashree@qmail.com"
    location: "ka"
    specialization: "diabetes"
    username: "nagashree"
```

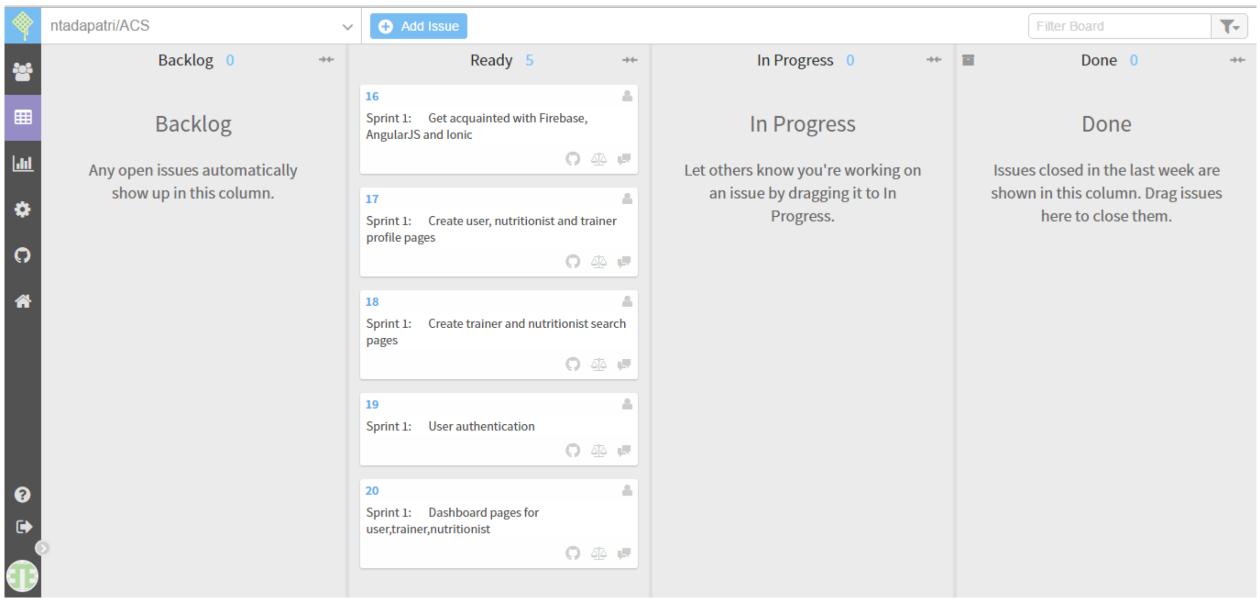


```
Messages
    -KKOs6OuO6dg2gEfOMm1
    -KKYSryXS2fyNIMLcVwt
    -KNlpkz2Myo3tptMDJoL
        Messenger
            -KNlpsclbXlxASlw_wMr
            -KNnE2RplXa4LyLS8Exm
            -KNnE3lfnCwE6rVqfz29
            -KNpFWE6lR7aLFVFnGCO
                from: "nagashree@qmail.com"
                msg: "nut here"
                to: "ntadapatri@qmail.com"
        email: "nagashree@qmail.com"
```

SOFTWARE DEVELOPMENT PLANNING

The development of HOPEFitness has been done in 5 Sprints, each Sprint spanning 1 week. Each Sprint began with a Sprint plan. At the end of the Sprint, a Sprint review was formulated followed by Sprint plan for the following Sprint.

Sprint 1 Plan


ntadapatri/ACS
Add Issue
Filter Board

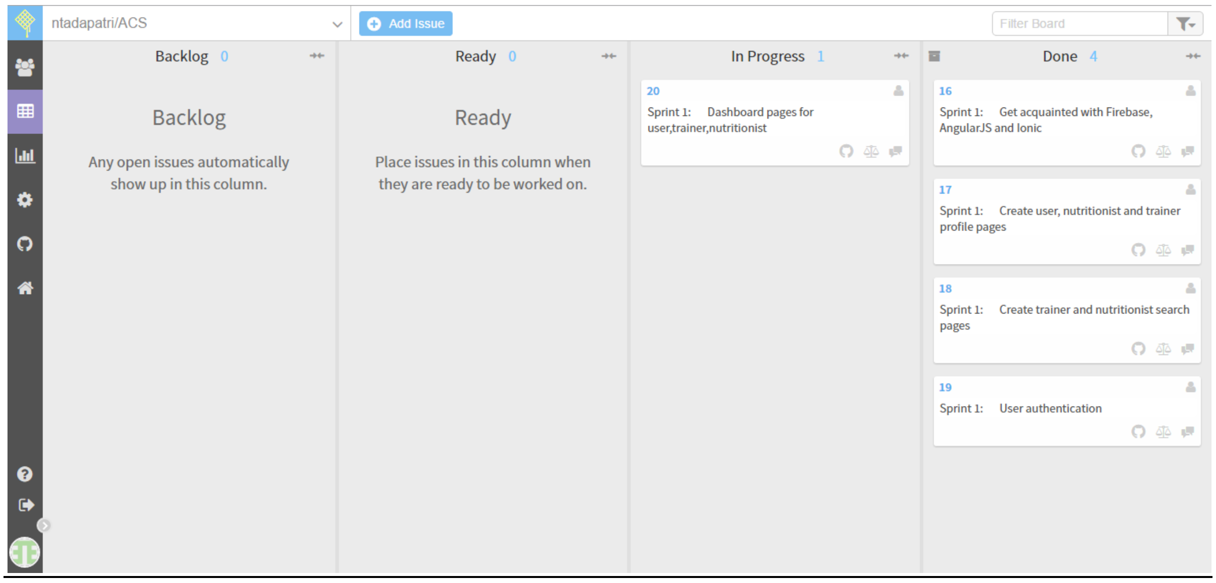
Backlog 0
Backlog
Any open issues automatically show up in this column.

Ready 5
16 Sprint 1: Get acquainted with Firebase, AngularJS and Ionic
17 Sprint 1: Create user, nutritionist and trainer profile pages
18 Sprint 1: Create trainer and nutritionist search pages
19 Sprint 1: User authentication
20 Sprint 1: Dashboard pages for user,trainer,nutritionist

In Progress 0
In Progress
Let others know you're working on an issue by dragging it to In Progress.

Done 0
Done
Issues closed in the last week are shown in this column. Drag issues here to close them.

Sprint 1 Review


ntadapatri/ACS
Add Issue
Filter Board

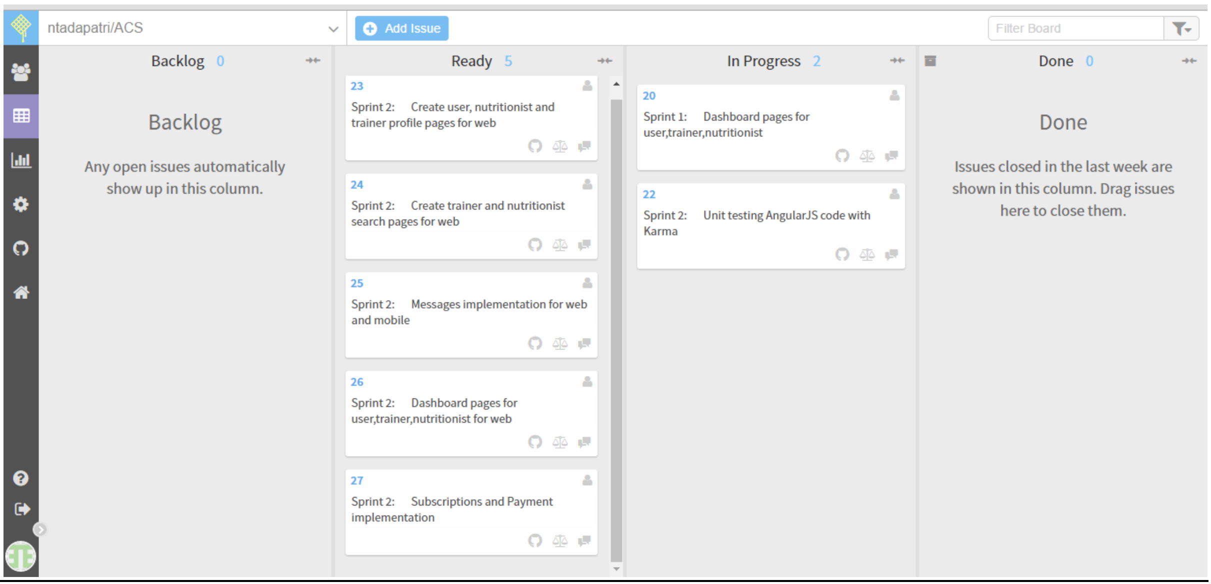
Backlog 0
Backlog
Any open issues automatically show up in this column.

Ready 0
Ready
Place issues in this column when they are ready to be worked on.

In Progress 1
20 Sprint 1: Dashboard pages for user,trainer,nutritionist

Done 4
16 Sprint 1: Get acquainted with Firebase, AngularJS and Ionic
17 Sprint 1: Create user, nutritionist and trainer profile pages
18 Sprint 1: Create trainer and nutritionist search pages
19 Sprint 1: User authentication

Sprint 2 Plan


ntadapatri/ACS

Add Issue

Filter Board

Backlog 0

Backlog

Any open issues automatically show up in this column.

Ready 5

23 Sprint 2: Create user, nutritionist and trainer profile pages for web

24 Sprint 2: Create trainer and nutritionist search pages for web

25 Sprint 2: Messages implementation for web and mobile

26 Sprint 2: Dashboard pages for user,trainer,nutritionist for web

27 Sprint 2: Subscriptions and Payment implementation

In Progress 2

20 Sprint 1: Dashboard pages for user,trainer,nutritionist

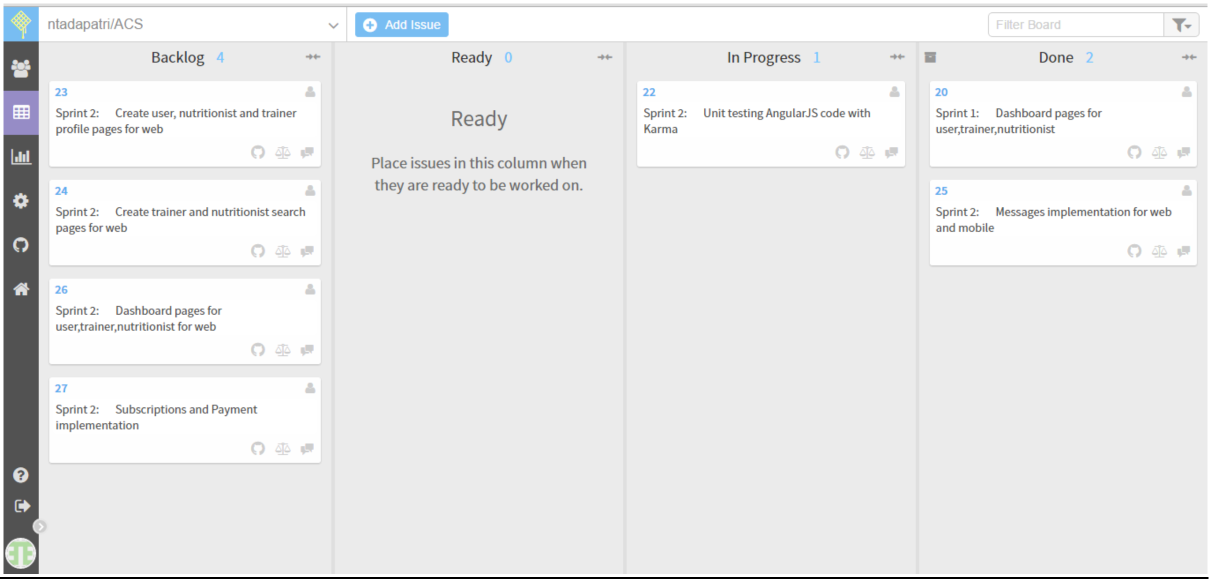
22 Sprint 2: Unit testing AngularJS code with Karma

Done 0

Done

Issues closed in the last week are shown in this column. Drag issues here to close them.

Sprint 2 Review


ntadapatri/ACS

Add Issue

Filter Board

Backlog 4

23 Sprint 2: Create user, nutritionist and trainer profile pages for web

24 Sprint 2: Create trainer and nutritionist search pages for web

26 Sprint 2: Dashboard pages for user,trainer,nutritionist for web

27 Sprint 2: Subscriptions and Payment implementation

Ready 0

Ready

Place issues in this column when they are ready to be worked on.

In Progress 1

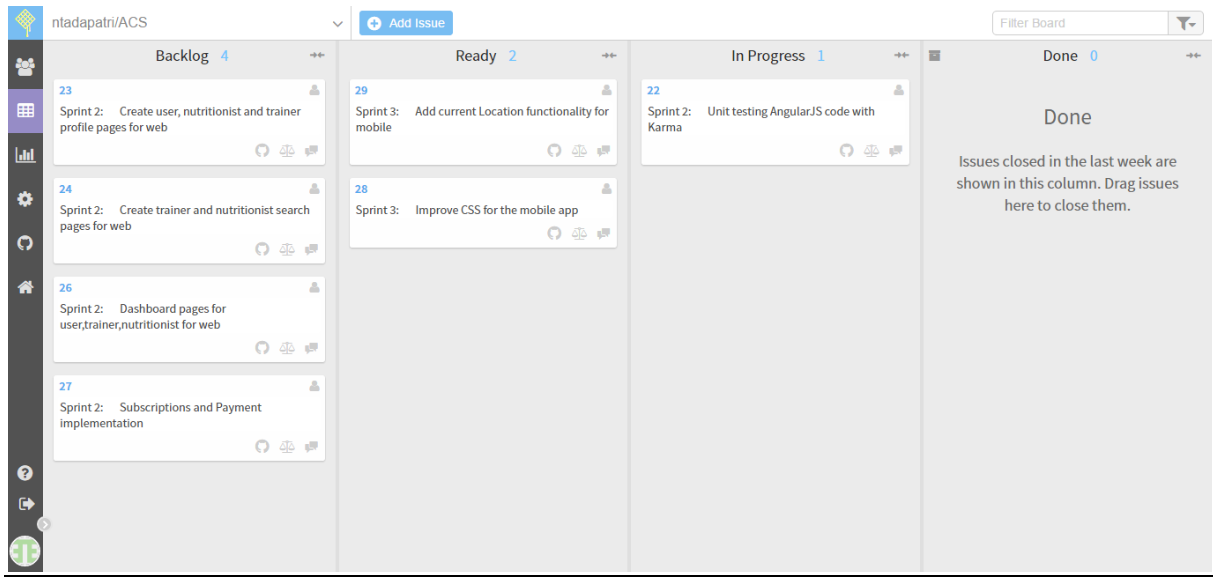
22 Sprint 2: Unit testing AngularJS code with Karma

Done 2

20 Sprint 1: Dashboard pages for user,trainer,nutritionist

25 Sprint 2: Messages implementation for web and mobile

Sprint 3 Plan


ntadapatri/ACS

Add Issue

Filter Board

Backlog 4

23 Sprint 2: Create user, nutritionist and trainer profile pages for web

24 Sprint 2: Create trainer and nutritionist search pages for web

26 Sprint 2: Dashboard pages for user,trainer,nutritionist for web

27 Sprint 2: Subscriptions and Payment implementation

Ready 2

29 Sprint 3: Add current Location functionality for mobile

28 Sprint 3: Improve CSS for the mobile app

In Progress 1

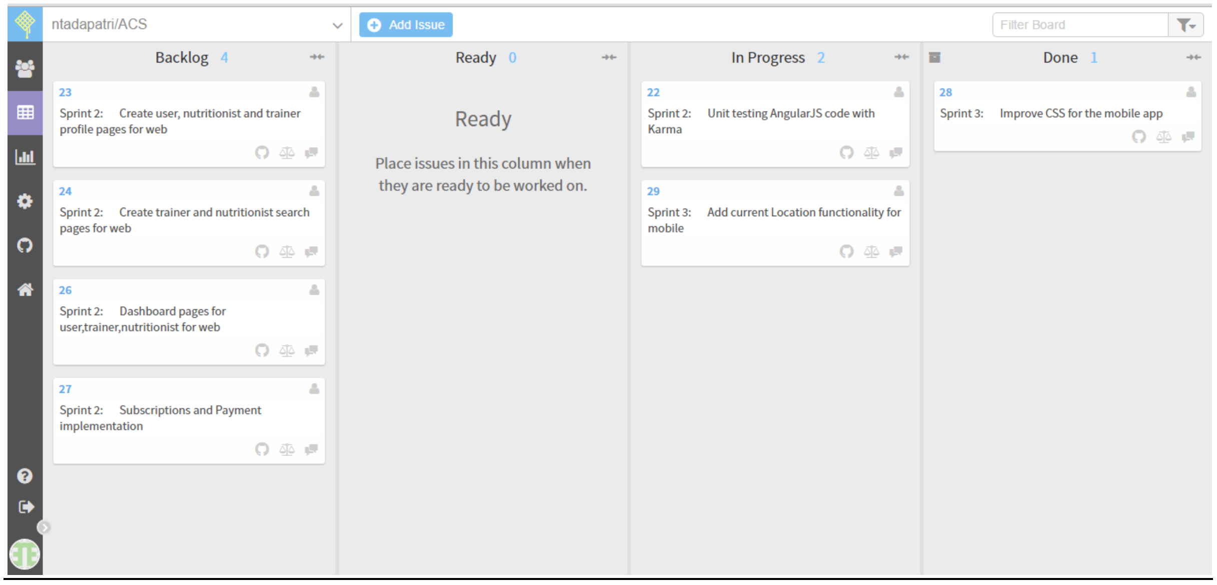
22 Sprint 2: Unit testing AngularJS code with Karma

Done 0

Done

Issues closed in the last week are shown in this column. Drag issues here to close them.

Sprint 3 Review


ntadapatri/ACS

Add Issue

Filter Board

Backlog 4

23 Sprint 2: Create user, nutritionist and trainer profile pages for web

24 Sprint 2: Create trainer and nutritionist search pages for web

26 Sprint 2: Dashboard pages for user,trainer,nutritionist for web

27 Sprint 2: Subscriptions and Payment implementation

Ready 0

Ready

Place issues in this column when they are ready to be worked on.

In Progress 2

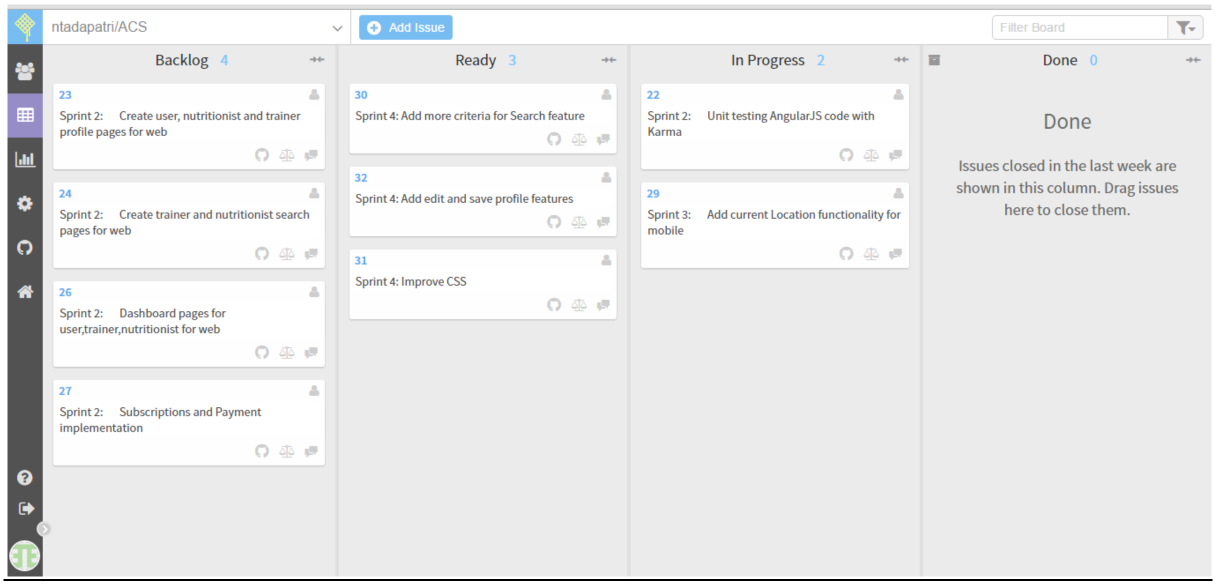
22 Sprint 2: Unit testing AngularJS code with Karma

29 Sprint 3: Add current Location functionality for mobile

Done 1

28 Sprint 3: Improve CSS for the mobile app

Sprint 4 Plan



ntadapatri/ACS

Add Issue

Filter Board

Backlog 4

23
Sprint 2: Create user, nutritionist and trainer profile pages for web

24
Sprint 2: Create trainer and nutritionist search pages for web

26
Sprint 2: Dashboard pages for user,trainer,nutritionist for web

27
Sprint 2: Subscriptions and Payment implementation

Ready 3

30
Sprint 4: Add more criteria for Search feature

32
Sprint 4: Add edit and save profile features

31
Sprint 4: Improve CSS

In Progress 2

22
Sprint 2: Unit testing AngularJS code with Karma

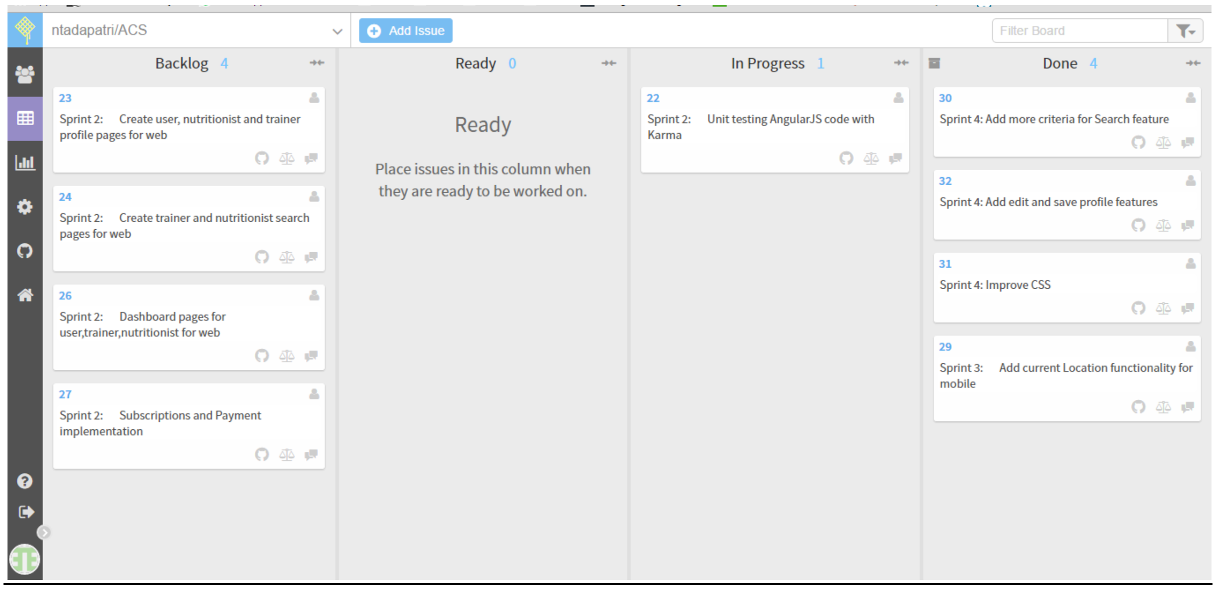
29
Sprint 3: Add current Location functionality for mobile

Done 0

Done

Issues closed in the last week are shown in this column. Drag issues here to close them.

Sprint 4 Review



ntadapatri/ACS

Add Issue

Filter Board

Backlog 4

23
Sprint 2: Create user, nutritionist and trainer profile pages for web

24
Sprint 2: Create trainer and nutritionist search pages for web

26
Sprint 2: Dashboard pages for user,trainer,nutritionist for web

27
Sprint 2: Subscriptions and Payment implementation

Ready 0

Ready

Place issues in this column when they are ready to be worked on.

In Progress 1

22
Sprint 2: Unit testing AngularJS code with Karma

Done 4

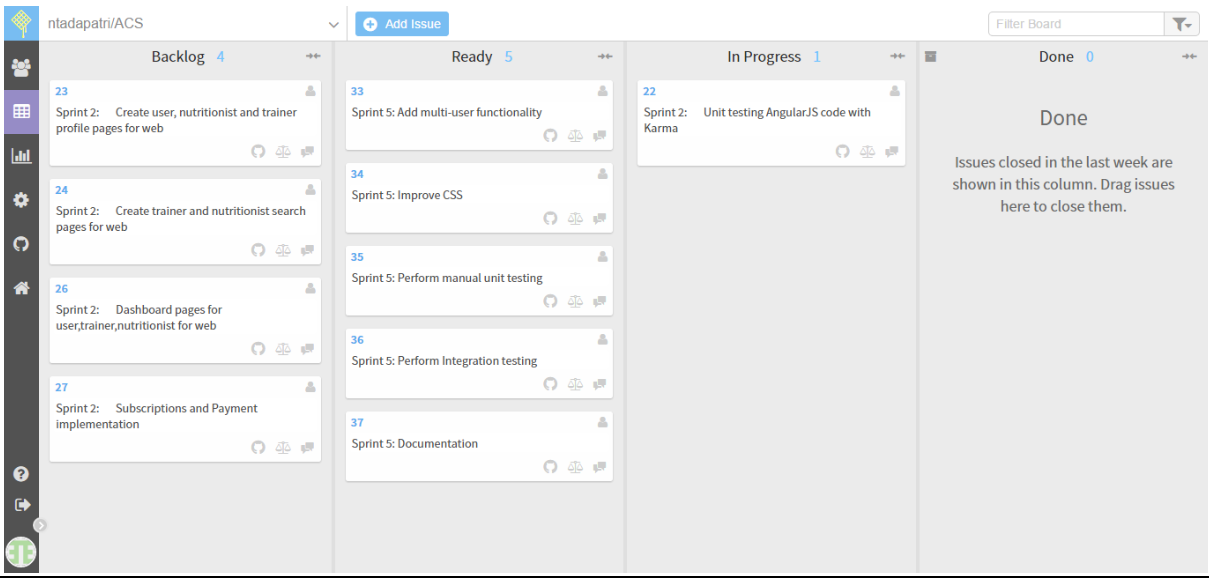
30
Sprint 4: Add more criteria for Search feature

32
Sprint 4: Add edit and save profile features

31
Sprint 4: Improve CSS

29
Sprint 3: Add current Location functionality for mobile

Sprint 5 Plan


ntadapatri/ACS

Add Issue

Filter Board

Backlog 4

23 Sprint 2: Create user, nutritionist and trainer profile pages for web

24 Sprint 2: Create trainer and nutritionist search pages for web

26 Sprint 2: Dashboard pages for user,trainer,nutritionist for web

27 Sprint 2: Subscriptions and Payment implementation

Ready 5

33 Sprint 5: Add multi-user functionality

34 Sprint 5: Improve CSS

35 Sprint 5: Perform manual unit testing

36 Sprint 5: Perform Integration testing

37 Sprint 5: Documentation

In Progress 1

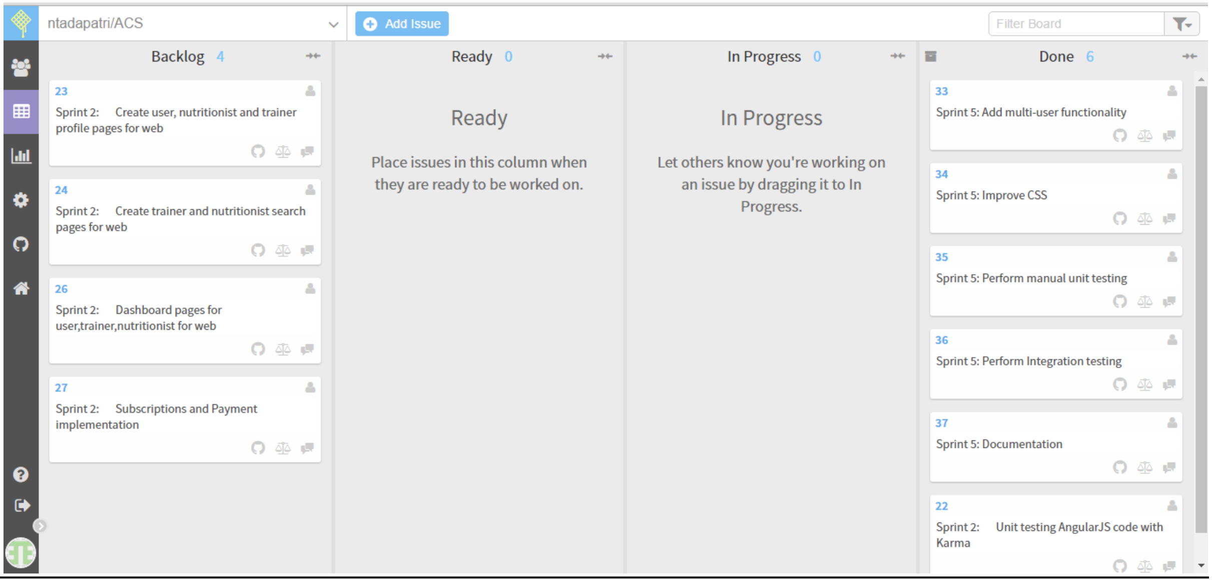
22 Sprint 2: Unit testing AngularJS code with Karma

Done 0

Done

Issues closed in the last week are shown in this column. Drag issues here to close them.

Sprint 5 Review


ntadapatri/ACS

Add Issue

Filter Board

Backlog 4

23 Sprint 2: Create user, nutritionist and trainer profile pages for web

24 Sprint 2: Create trainer and nutritionist search pages for web

26 Sprint 2: Dashboard pages for user,trainer,nutritionist for web

27 Sprint 2: Subscriptions and Payment implementation

Ready 0

Ready

Place issues in this column when they are ready to be worked on.

In Progress 0

In Progress

Let others know you're working on an issue by dragging it to In Progress.

Done 6

33 Sprint 5: Add multi-user functionality

34 Sprint 5: Improve CSS

35 Sprint 5: Perform manual unit testing

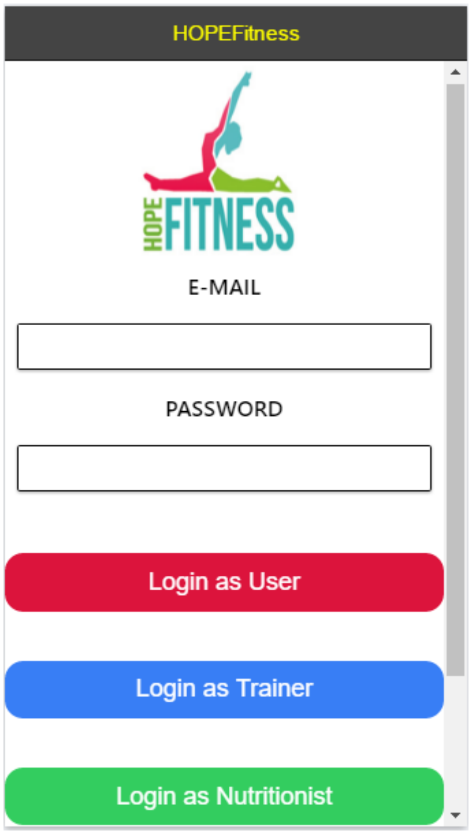
36 Sprint 5: Perform Integration testing

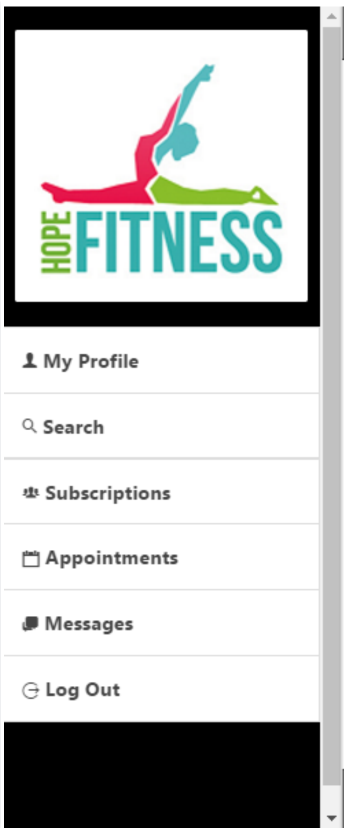
37 Sprint 5: Documentation

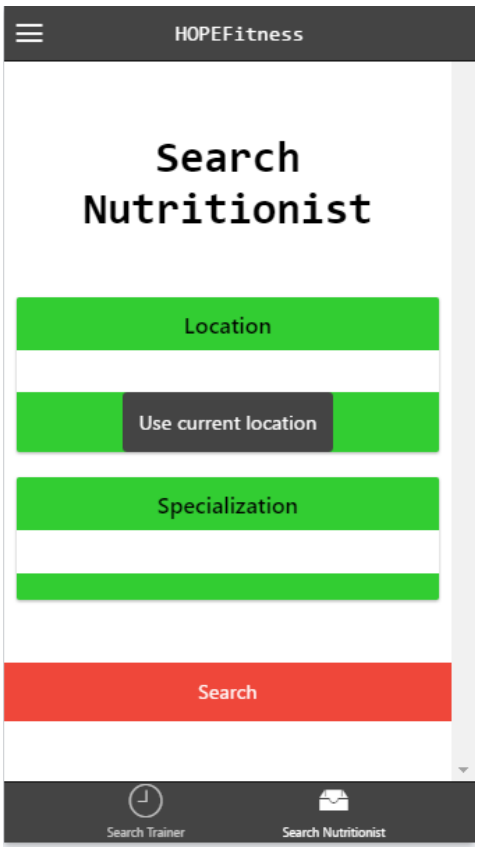
22 Sprint 2: Unit testing AngularJS code with Karma

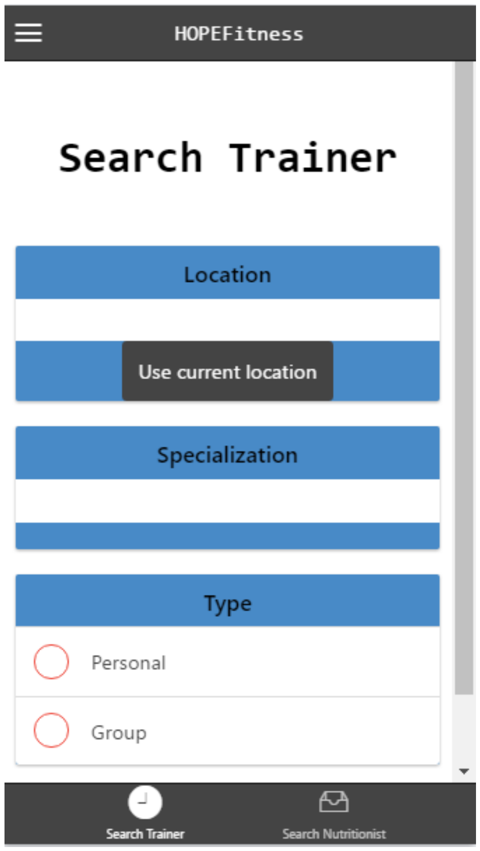
IMPLEMENTATION

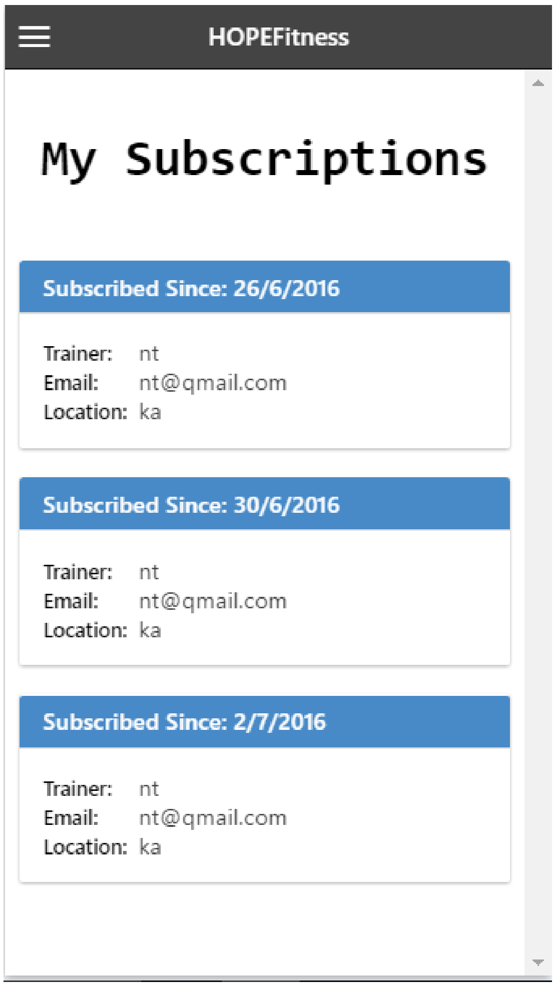
The following are some screen captures of the HOPEFitness application.










HOPEFitness

My Subscriptions

Subscribed Since: 26/6/2016

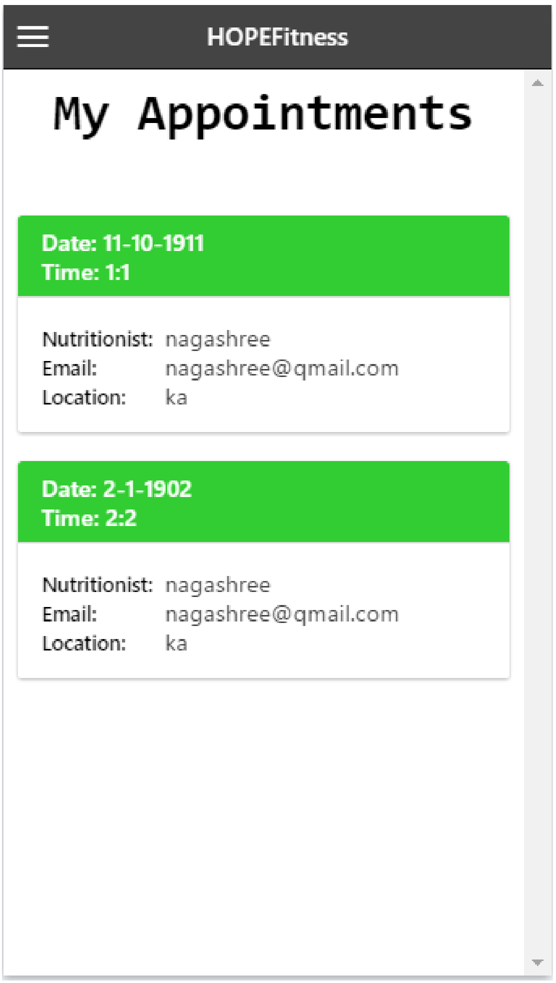
Trainer: nt
Email: nt@qmail.com
Location: ka

Subscribed Since: 30/6/2016

Trainer: nt
Email: nt@qmail.com
Location: ka

Subscribed Since: 2/7/2016

Trainer: nt
Email: nt@qmail.com
Location: ka


HOPEFitness

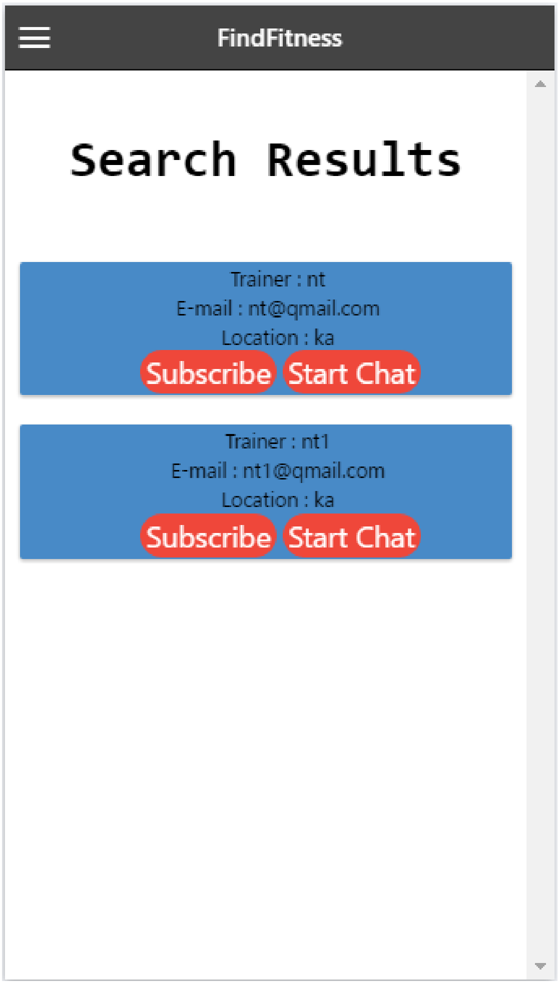
My Appointments

Date: 11-10-1911
Time: 1:1

Nutritionist: nagashree
Email: nagashree@qmail.com
Location: ka

Date: 2-1-1902
Time: 2:2

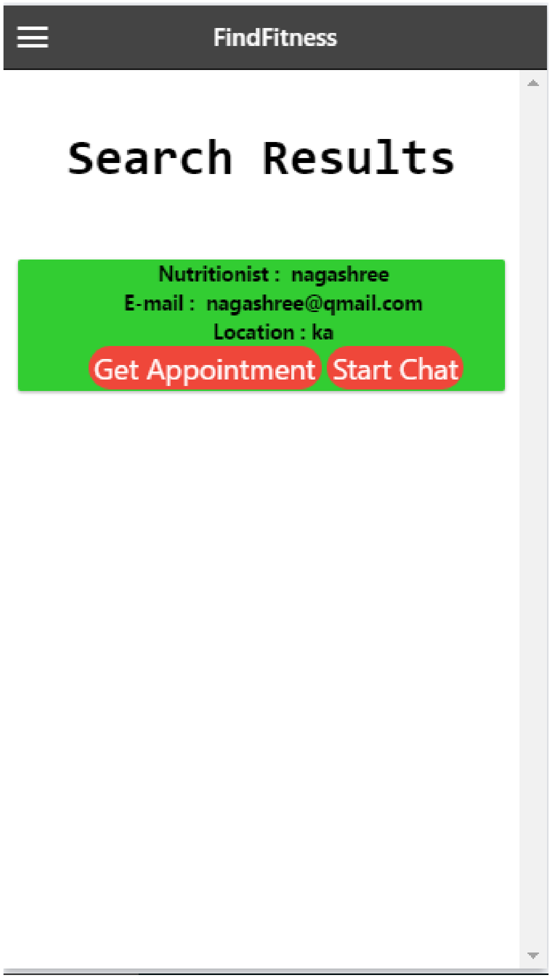
Nutritionist: nagashree
Email: nagashree@qmail.com
Location: ka


FindFitness

Search Results

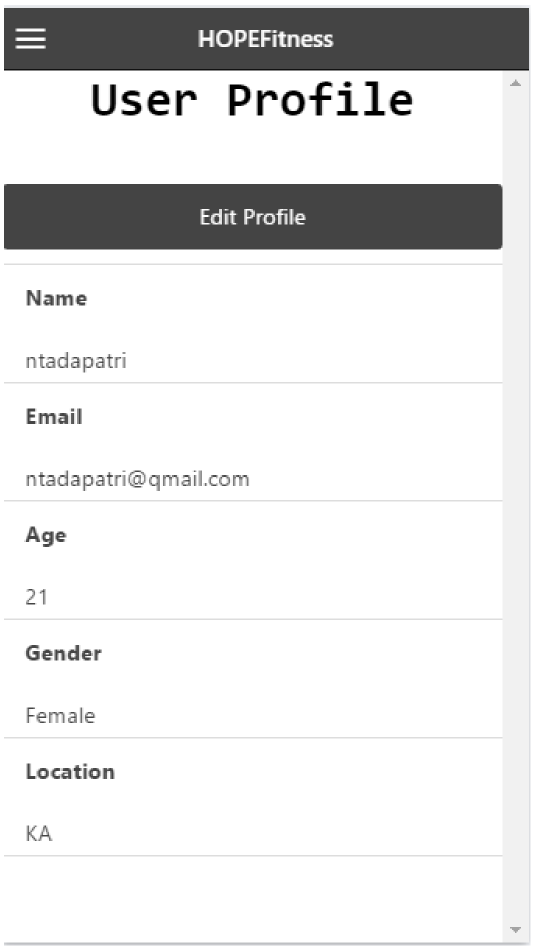
Trainer : nt
E-mail : nt@qmail.com
Location : ka

Trainer : nt1
E-mail : nt1@qmail.com
Location : ka


FindFitness

Search Results

Nutritionist : nagashree
E-mail : nagashree@qmail.com
Location : ka


HOPEFitness

User Profile

Edit Profile

Name

ntadapatri

Email

ntadapatri@qmail.com

Age

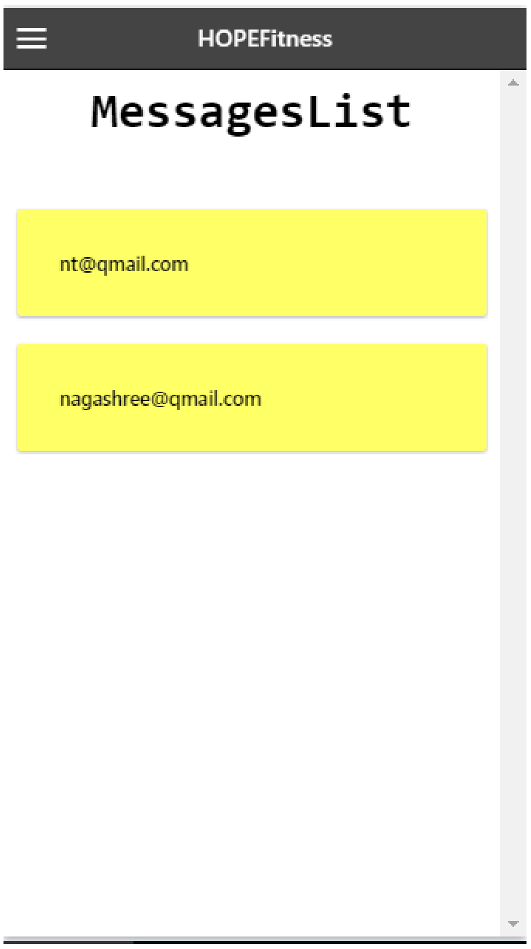
21

Gender

Female

Location

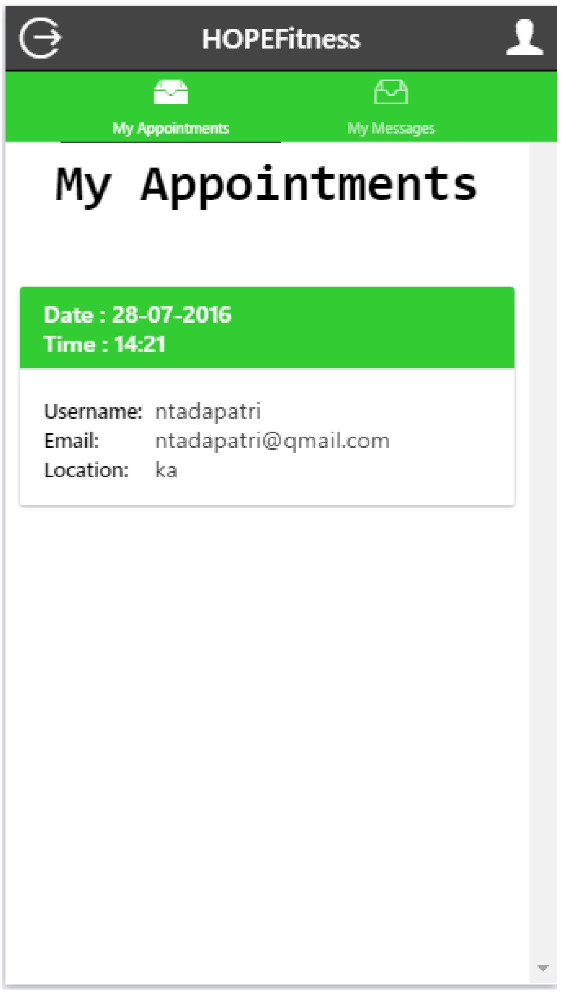
KA


HOPEFitness

MessagesList

nt@qmail.com

nagashree@qmail.com

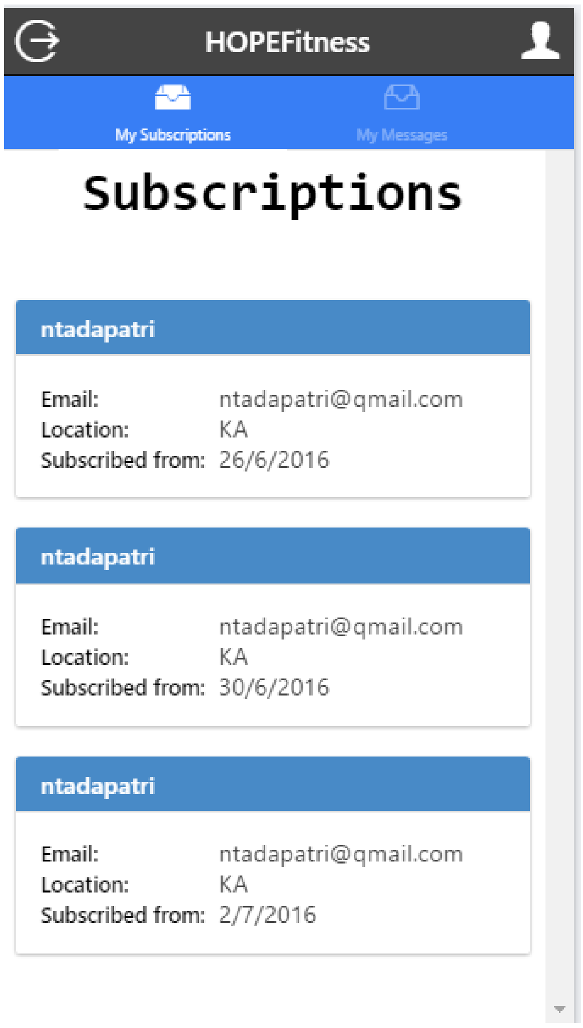

HOPEFitness

My Appointments | My Messages

My Appointments

Date : 28-07-2016
Time : 14:21

Username: ntadapatri
Email: ntadapatri@qmail.com
Location: ka


HOPEFitness

My Subscriptions | My Messages

Subscriptions

ntadapatri

Email: ntadapatri@qmail.com
Location: KA
Subscribed from: 26/6/2016

ntadapatri

Email: ntadapatri@qmail.com
Location: KA
Subscribed from: 30/6/2016

ntadapatri

Email: ntadapatri@qmail.com
Location: KA
Subscribed from: 2/7/2016

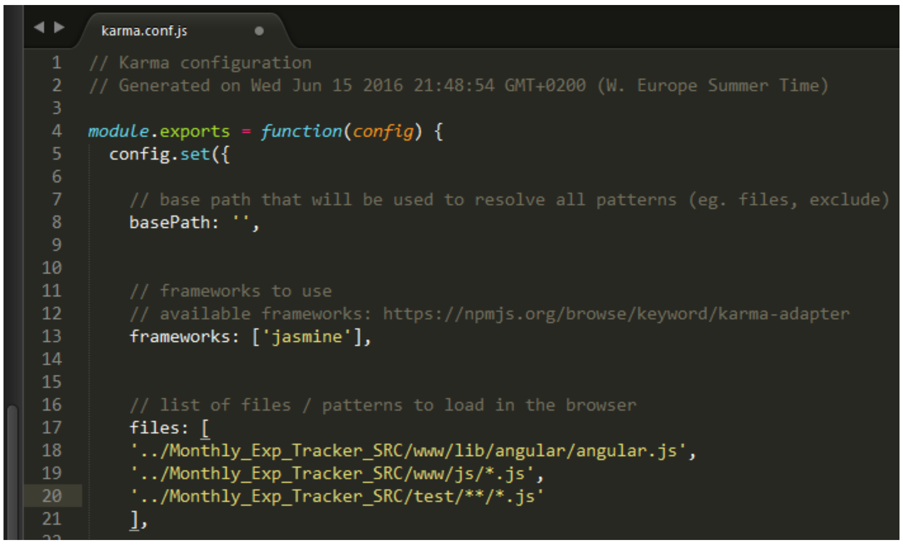
TESTING

UNIT TESTING

Unit Tests are fast and reliable because it involves testing small pieces of code without external dependencies.

Unit testing for HOPEFitness has been done at two stages.

1. Unit testing with Karma and Jasmine framework
   Unit testing with Karma and Jasmine is limited to merely testing the existence of the controllers in the code in this project. Although Karma offers a wide range of possibilities for unit testing the code, due to the limitations posed by the usage of Firebase in this application, Karma has been used only up to this extent.



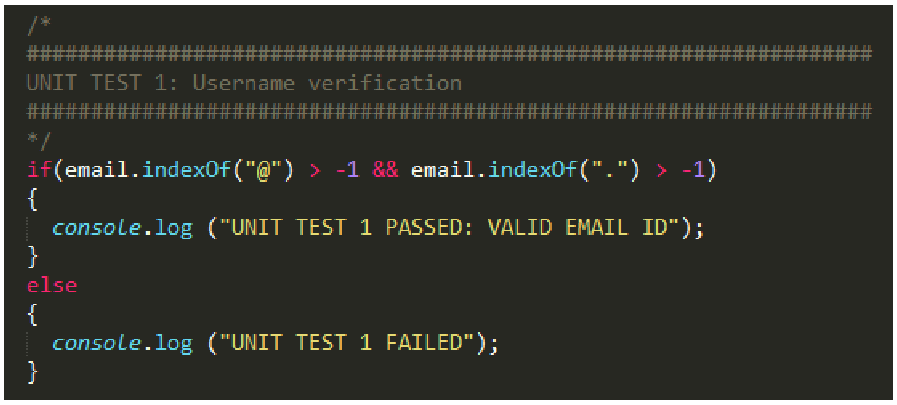
karma.conf.js

```
// Karma configuration
// Generated on Wed Jun 15 2016 21:48:54 GMT+0200 (W. Europe Summer Time)

module.exports = function(config) {
  config.set({

    // base path that will be used to resolve all patterns (eg. files, exclude)
    basePath: '',


    // frameworks to use
    // available frameworks: https://npmjs.org/browse/keyword/karma-adapter
    frameworks: ['jasmine'],


    // list of files / patterns to load in the browser
    files: [
    '../Monthly_Exp_Tracker_SRC/www/lib/angular/angular.js',
    '../Monthly_Exp_Tracker_SRC/www/js/*.js',
    '../Monthly_Exp_Tracker_SRC/test/**/*.js'
    ],
```

   A snapshot of Karma configuration file.

2. Manual unit testing
   The following cases have been tested by manually writing test code.

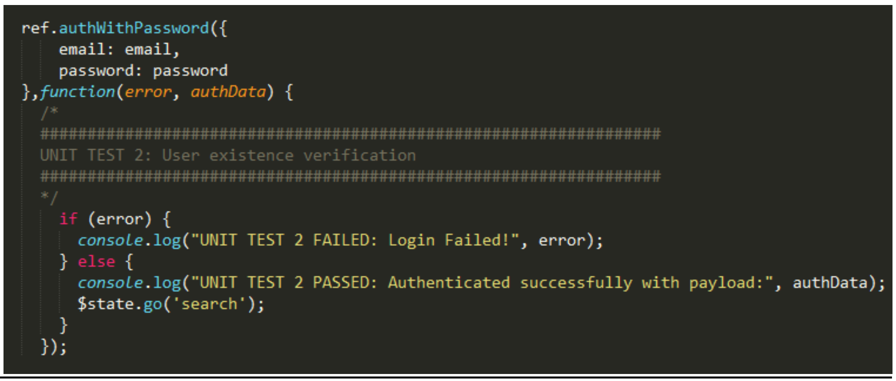
   UNIT TEST 1: Username verification
   The username during login is verified to check whether or not it contains a valid email ID structure.



```
/*
##################################################################
UNIT TEST 1: Username verification
##################################################################
*/
if(email.indexOf("@") > -1 && email.indexOf(".") > -1)
{
  console.log ("UNIT TEST 1 PASSED: VALID EMAIL ID");
}
else
{
  console.log ("UNIT TEST 1 FAILED");
}
```

   UNIT TEST 2: User existence verification

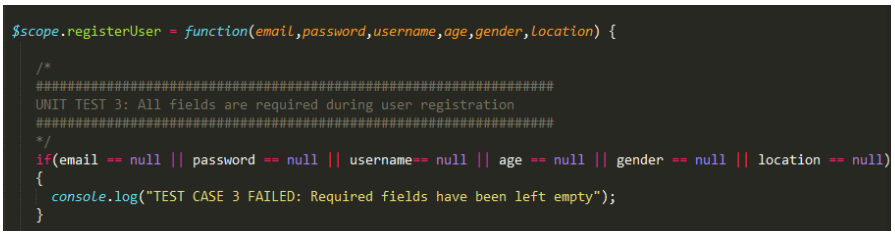
A simple verification to test if user is registered in Firebase



```
ref.authWithPassword({
    email: email,
    password: password
},function(error, authData) {
  /*
  ####################################################################
  UNIT TEST 2: User existence verification
  ####################################################################
  */
    if (error) {
      console.log("UNIT TEST 2 FAILED: Login Failed!", error);
    } else {
      console.log("UNIT TEST 2 PASSED: Authenticated successfully with payload:", authData);
      $state.go('search');
    }
  });
```

UNIT TEST 3: All fields are required during user registration

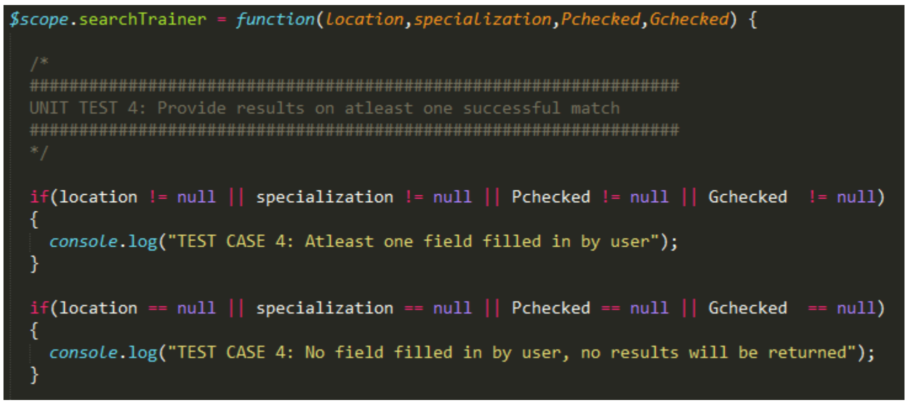
Test case to verify if all fields are filled in the registration form are filled in by the user.



```
$scope.registerUser = function(email,password,username,age,gender,location) {

  /*
  ####################################################################
  UNIT TEST 3: All fields are required during user registration
  ####################################################################
  */
  if(email == null || password == null || username== null || age == null || gender == null || location == null)
  {
    console.log("TEST CASE 3 FAILED: Required fields have been left empty");
  }
```

UNIT TEST 4: Provide results on at least one successful match

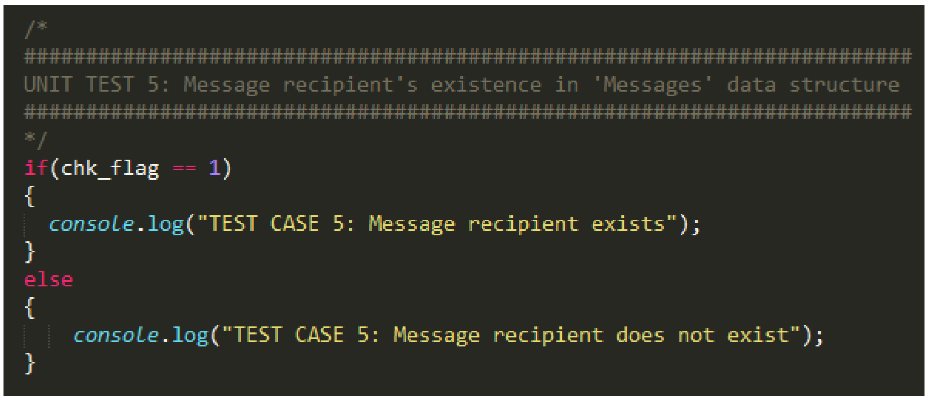
Test case to verify if the user has filled at least one field in the search page. Results are provided even if there exists one field that has an exact match to the data in database.



```
$scope.searchTrainer = function(location,specialization,Pchecked,Gchecked) {

  /*
  ####################################################################
  UNIT TEST 4: Provide results on atleast one successful match
  ####################################################################
  */

  if(location != null || specialization != null || Pchecked != null || Gchecked  != null)
  {
    console.log("TEST CASE 4: Atleast one field filled in by user");
  }

  if(location == null || specialization == null || Pchecked == null || Gchecked  == null)
  {
    console.log("TEST CASE 4: No field filled in by user, no results will be returned");
  }
```

UNIT TEST 5: Message recipient's existence in 'Messages' data structure

Test case to verify if the message recipient exists in the data store. If the recipient doesn’t exists then he is added to the data store.

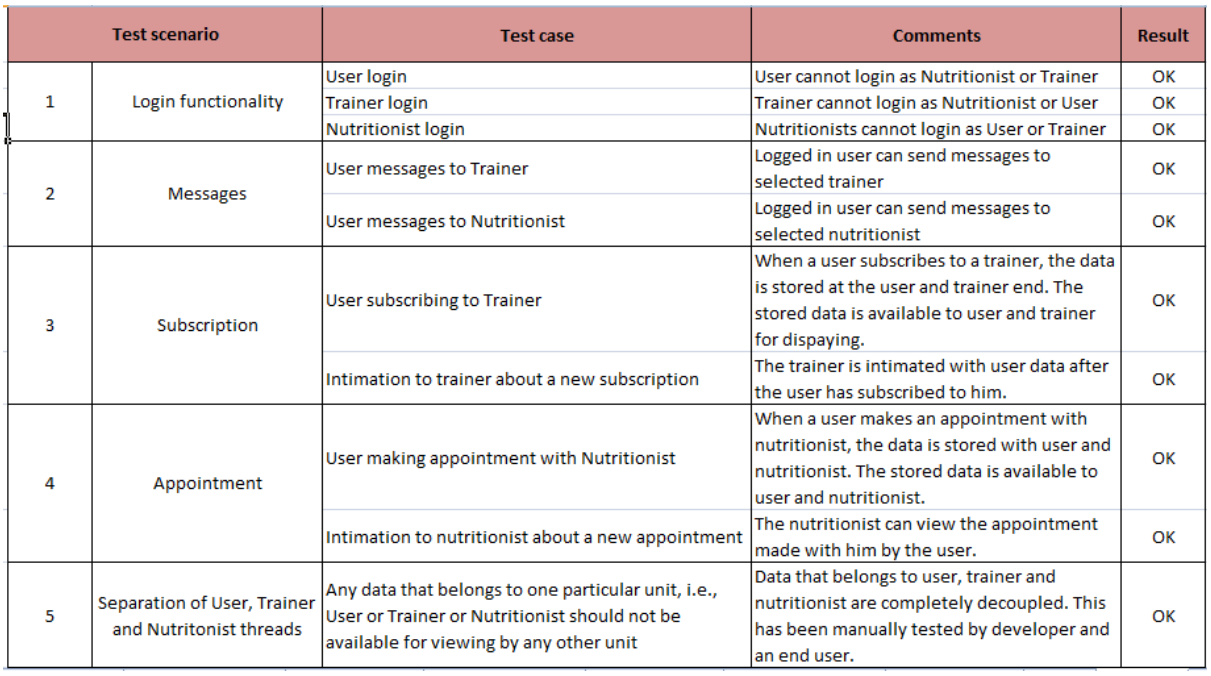


```
/*
#######################################################################
UNIT TEST 5: Message recipient's existence in 'Messages' data structure
#######################################################################
*/
if(chk_flag == 1)
{
  console.log("TEST CASE 5: Message recipient exists");
}
else
{
    console.log("TEST CASE 5: Message recipient does not exist");
}
```

INTEGRATION TESTING

The basic units of the app are Users, Trainers and Nutritionists. After unit tests have been written for the three basic units, manual integration tests have been performed to verify that all these units behave as expected when they are coupled together.

The test cases of manual integration testing and its results are shown below in a brief manner.



Test scenario		Test case	Comments	Result
1	Login functionality	User login	User cannot login as Nutritionist or Trainer	OK
		Trainer login	Trainer cannot login as Nutritionist or User	OK
		Nutritionist login	Nutritionists cannot login as User or Trainer	OK
2	Messages	User messages to Trainer	Logged in user can send messages to selected trainer	OK
		User messages to Nutritionist	Logged in user can send messages to selected nutritionist	OK
3	Subscription	User subscribing to Trainer	When a user subscribes to a trainer, the data is stored at the user and trainer end. The stored data is available to user and trainer for dispaying.	OK
		Intimation to trainer about a new subscription	The trainer is intimated with user data after the user has subscribed to him.	OK
4	Appointment	User making appointment with Nutritionist	When a user makes an appointment with nutritionist, the data is stored with user and nutritionist. The stored data is available to user and nutritionist.	OK
		Intimation to nutritionist about a new appointment	The nutritionist can view the appointment made with him by the user.	OK
5	Separation of User, Trainer and Nutritonist threads	Any data that belongs to one particular unit, i.e., User or Trainer or Nutritionist should not be available for viewing by any other unit	Data that belongs to user, trainer and nutritionist are completely decoupled. This has been manually tested by developer and an end user.	OK

PERFORMANCE TUNING

The performance tuning measures used in the app are listed below.

➔ Usage of Angular template cache to ensure all views are already cached at app startup. gulp-angular-templatecache plugin is used for this purpose.

➔ Enabling Ionic's Native Scrolling instead of using the default Javascript scrolling. It can be enabled per view by placing attribute overflow-scroll="true" on an ion-content tag. Or we can enable it for all views with $ionicConfigProvider.scrolling.jsScrolling(false);

➔ Adding Crosswalk plugin to obtain improvement on CSS animations. Crosswalk provides a bundled Chrome Webview with the app, resulting in better performance and predictability.

➔ Usage of “collection-repeat” instead of “ng-repeat” to iterate over items stored in arrays or data store.

➔ In case of animation usage, explicitly enabling animation serves as a performance improvement. As another option, Native transitions plugin was used which did not result in successful optimization of performance.

➔ Usage of Infinite-scroll and Pull-to-refresh options

FUTURE SCOPE

The following features that was part of the Software Development plan have not been implemented in the submitted version of HOPEFitness. Hence, the features listed below fall into the future scope of this project.

- Implementation of Payment methods for users to transfer money to their subscribed trainers and nutritionists. Also, offering multiple payment options.
- Post-Ad feature for trainers and nutritionists, and present to users those advertisements that are filtered based on users’ criteria.
- Much better CSS!
- Implementation of Splash screen during first time load of the mobile app.
- Perform End-to-end testing of the application.
- Build the web application for HOPEFitness.

RESOURCES

List of Plug-ins, external JavaScript functions and external AngularJS Directives used

Plugins:
Cordova-plugin-geolocation
Com.telerik.plugins.nativepagetransitions
Cordova.plugin.local-notification

JavaScript functions:
Locator-tpl.js
Angular-chart.min.js
Angularfire.min.js
Chart.min.js
Firebase.js
Ng-cordova.min.js

AngularJS Directives:
Locator.min.js

Emulator:
Genymotion

REFERENCES

- ✓ Github.com
- ✓ Waffle.io
- ✓ Stackoverflow.com
- ✓ ionicframework.com
- ✓ angularjs.org
- ✓ w3schools.com for HTML5
- ✓ Firebaseio.com
- ✓ diylogodesigns.com for Logo