``

ITHub Chat Application

Final Year Project

Aaron Kane
BSC in Computing

Declaration Cover Sheet for Project Submission

SECTION 1 *Student to complete*

Name: **Aaron Kane**
Student ID: **X12554547**
Supervisor: **Frances Sheridan**

SECTION 2 Confirmation of Authorship

The acceptance of your work is subject to your signature on the following declaration:

I confirm that I have read the College statement on plagiarism (summarised overleaf and printed in full in the Student Handbook) and that the work I have submitted for assessment is entirely my own work.

Signature: Aaron Kane Date: 5th May 2016

NB. If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the College's Disciplinary

Committee. Should the Committee be satisfied that plagiarism has occurred this is likely to lead to your failing the module and possibly to your being suspended or expelled from college.

Complete the sections above and attach it to the front of one of the copies of your assignment,

What constitutes plagiarism or cheating?

The following is extracted from the college's formal statement on plagiarism as quoted in the Student Handbooks. References to "assignments" should be taken to include any piece of work submitted for assessment.

Paraphrasing refers to taking the ideas, words or work of another, putting it into your own words and crediting the source. This is acceptable academic practice provided you ensure that credit is given to the author. Plagiarism refers to copying the ideas and work of another and misrepresenting it as your own. This is completely unacceptable and is prohibited in all academic institutions. It is a serious offence and may result in a fail grade and/or disciplinary action. All sources that you use in your writing must be acknowledged and included in the reference or bibliography section. If a particular piece of writing proves difficult to paraphrase, or you want to include it in its original form, it must be enclosed in quotation marks
and credit given to the author.

When referring to the work of another author within the text of your project you must give the author's surname and the date the work was published. Full details for each source must then be given in the bibliography at the end of the project

Penalties for Plagiarism

If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the college's Disciplinary

Committee. Where the Disciplinary Committee makes a finding that there has been plagiarism, the Disciplinary Committee may recommend

- that a student's marks shall be reduced
- that the student be deemed not to have passed the assignment
- that other forms of assessment undertaken in that academic year by the same student be declared void
- that other examinations sat by the same student at the same sitting be declared void

Further penalties are also possible including

- suspending a student college for a specified time,
- expelling a student from college,
- prohibiting a student from sitting any examination or assessment.,
- the imposition of a fine and
- the requirement that a student to attend additional or other lectures or courses or undertake additional academic work.

Contents

b. **Requirement Specification**
c. **Monthly Reports**

d.

1. Executive Summary

The objective of this project is to provide an real-time solution for IT personalities to get together and discuss programming in real-time. Individuals will have the ability to log into an application that will host multiple chat rooms at once, each with their own individual topic such as Java, JavaScript, SQL etc. Each message sent by an individual is logged in a collection so that they can be retraced by an administrator if needed. The ITHub application will be available on a website and will be available to use on Android mobile devices which means individuals can access the app while "on-the-go".

2. Introduction

2.1 Background

As an IT student studying computing, I have always had an interest in trying to learn and research into new technologies. I have always had an avid interest in programming as a whole however, programming would not be one of my strong points. There are many different programming languages out there and it is very hard to memorise the different characteristics of them all. Each language has their own problems, issues and constraints that comes along with them which need to be discussed. Research plays a big part in these problems however there isn't an application on hand where you could ask somebody straight away.

The idea behind the application ITHub was that the application could be a place where IT enthusiasts could get together to discuss issues and have their problems resolved in real-time. At the moment the main source for issues to be researched into is Stack Overflow. However if a question is asked on these types of sites you could be waiting up to a week or two before it gets answered. The question might not even get answered at all! ITHub would allow the user to be logged into a real-time chat room and have their issue answered efficiently and effectively based on the room they are in.

2.2 Aims

As stated in the previous section, the aim of ITHub was to be a hybrid application which at first is a web-based application that is also available on an android device. The website of the application allows the user to login using their own details or Facebook details. Each account will be added to a database so that multiple accounts cannot be created at one time. Once logged into the website the user will be put into a general chatroom which will allow them to talk with other users. The sidebar of the site will allow users to switch instantly between the four rooms and allow them to talk in these rooms instantly. All messages of each room will be monitored and be logged in a collection incase these need to be accessed at any time by an administrator.

The second part of the application as a whole is that it is converted into an Android mobile application so that users can access the rooms “on-the-go”. The reason behind this is because most users do not want to log onto their machine if they can reach the application quickly on their phone. Android is also becoming the most popular mobile Operating System ahead of iPhone. I also feel more comfortable developing for Android.

2.3 Technologies

As the application is going to be a hybrid application, the application will be developed using the Meteor framework. Behind the framework will be HTML5, CSS3 and JavaScript. In order to hold the collection of users and the collection of messages, a MongoDB database will be used. In order to access the database, Meteor comes with an add-on called minimongo which is needed to pull data from the database.

To convert the application to an Android mobile application, there are two options. One is using Meteor to convert the application using their own framework. The other is using the Cordova framework from Phonegap which takes the code produced and wraps it in an Android webview.

2.4 Structure of the Document

The report is structured in a way that can be easily readable and understandable. The reader will first get an overview of what the project is about and how the project will be used.

The second section shows the reader the background of the application i.e what was the idea behind the application, the aims of the project i.e what I intend to create and also what are the technologies that will be used in order to create the project.

The next section of the report details the functional requirements of the application which details what they are, what activates these requirements, if there are any issues that will arise and what are risks that could happen. These requirements will go into detail about data requirements, availability requirements, usability requirements

The next section describes the system design and architecture. This will describe in detail the implementation of the system, how the system was tested and how the results turned out and also will describe in detail the user interface design of the application.

The following section of the report contains any conclusions after the development of the project and if any changes could be made during the development.

The fifth section of the report will look at any further developments of the application and how the application can be improved based on user interaction.

The sixth section will outline the bibliography of the application and what resources were used in the development and completion of ITHub.

The seventh and final section of the report will show the appendix which will include the orginal project proposal, requirements specification and the monthly reflective journals outlining how the development of the application has changed overtime.

3. System

3.1 Requirements

3.1.1 Functional Requirements

3.1.1.1 Requirement One

Chat management

3.1.1.2 Description and Priority

This is the main functionality of the application where the collection of messages will be stored and maintained within the MongoDB collection. As more and more individuals chat within the rooms provided the collection will grow. Messages within the selected rooms can be viewed however they can not be manipulated.

This has been done in order to stop collections of users and messages being altered and deleted.

3.1.1.3 Requirement Activation

The chat is activated once the user has logged into the application using the login provided. Once logged in and a room has been selected from the sidebar, the user can then add messages to the selected room. These messages are stored in a MongoDB collection and each message is timestamped, dated and is given a unique ID if it needs to be retrieved.

3.1.1.4 Technical Issues

In order to access this chat function, the user must have either a computer with an internet enabled web browser or an Android device with internet enabled.

3.1.1.5 Risks

In order for this chat function to run successfully, the collection of messages must be available for administrators to access. As MongoDB is included within the Meteor framework there is a slim chance that the database will go down however there is always a small possibility that it will go down. If the database does decide to malfunction, users will be unable to chat or log into the application.

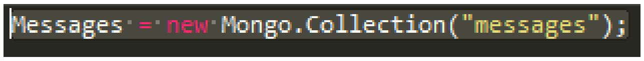
To reduce this risk, Meteor provides security around MongoDB in order to stop injections from individuals.

3.1.1.6 Dependencies with other requirements

For users to interact with the chat messages, they will need to login to the application. In order for the application to function at it's full potential, chat messages are essential.

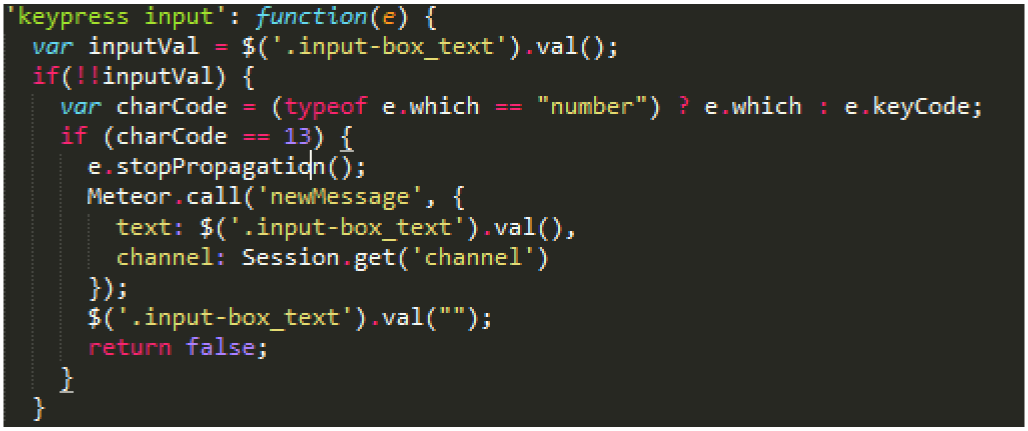
3.1.1.7 Code segment

The code segment details how the messages are first stored in a MongoDB collection and are then accessed by the main template. A mongo collection has no limit to the amount of storage it can have. The messages are stored using the simple following code:



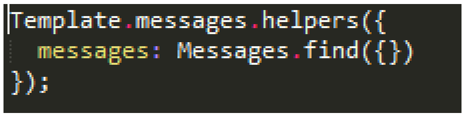
```
Messages = new Mongo.Collection("messages");
```

New messages are constantly being added to the collection and in order to do this, they must be "subscribed" to both the client and server side. These subscribed messages are then relayed back to the user. These messages are validated compared to an alpha-numeric alphabet to make sure there is no malicious code being added into the room:



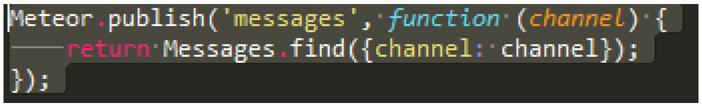
```
'keypress input': function(e) {
  var inputVal = $('.input-box_text').val();
  if(!!inputVal) {
    var charCode = (typeof e.which == "number") ? e.which : e.keyCode;
    if (charCode == 13) {
      e.stopPropagation();
      Meteor.call('newMessage', {
        text: $('.input-box_text').val(),
        channel: Session.get('channel')
      });
      $('.input-box_text').val("");
      return false;
    }
  }
```

Any messages that are okay are then found by the message “helper” which in turn helps return the messages to the user:



```
Template.messages.helpers({
  messages: Messages.find({})
});
```

Messages are found however they need to be published back towards the user in order for them to be seen. To combat this Meteor allows messages to be published instantly. Messages then become available to the user and can be read easily based on the channel that was selected by the user:



```
Meteor.publish('messages', function (channel) {
    return Messages.find({channel: channel});
});
```

3.1.2.1 Requirement Two

Templates and Routing of Templates

3.1.2.2 Description and Priority

The ITHub application will function on a series of templates which will react and switch in real-time based on the selection of the user. The switching of these templates happens by using a technology known as Iron Router. Moving between templates is easy using HTML5 however these templates are not reactive and would not provide the functionality needed for the chat to function.

Iron Router allows the templates to be reactive as well as switch instantly between the templates meaning the page does not need to refresh.

3.1.2.3 Requirement Activation

This requirement is activated when the user is logged in and selects a template to switch to. The main templates of the application are the header, footer and room selection templates. Once the user has selected their room to switch to, the app will switch instantly and there should be no refreshing or slow response from the page.

3.1.2.4 Technical Issues

For the templates to run efficiently, the user must be using a internet enabled web browser which supports HTML5. Most browsers at the moment support HTML5. If the user is accessing the application from their Android mobile device, the device must be able to support HTML5 also and needs to be internet enabled also.

3.1.2.5 Risks

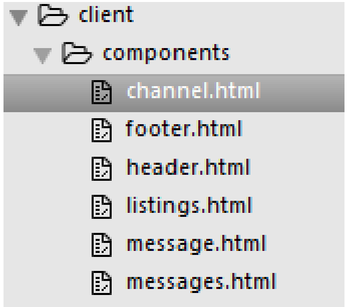
The templates will require a connection to the internet in order to switch between the channels of the application. If there is no connection, the user will be unable to access any part of the application i.e messages, login, chat rooms

3.1.2.6 Dependencies with other requirements

For any users who are sending messages and switching between the rooms they would also need to be connected to the internet. Templates will not function or switch without internet connection.

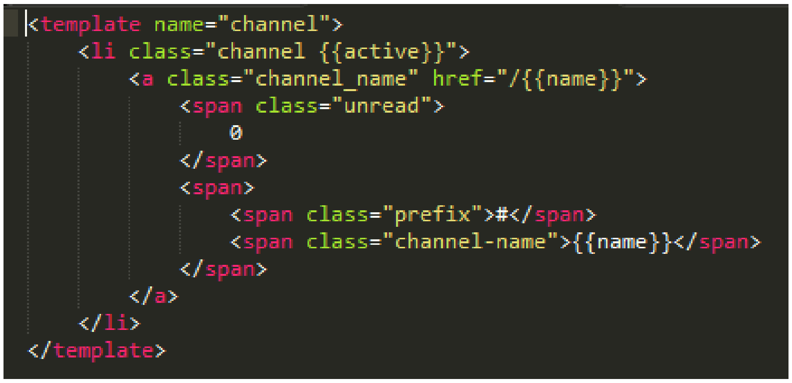
3.1.2.7 Code segment

Templates play a major part in the development of the application. These are key components and without them there would be no framework for the application to function. All of the html templates were stored in a components folder on the client side of the application:



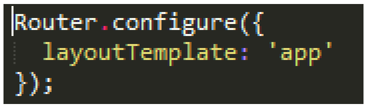
```
client
  components
    channel.html
    footer.html
    header.html
    listings.html
    message.html
    messages.html
```

Templates are structured differently compared to a normal HTML files as templates have their own <head> and <body> tag built into them which provides ease of access. As you can see above the channel file is selected and is structured below as follows:



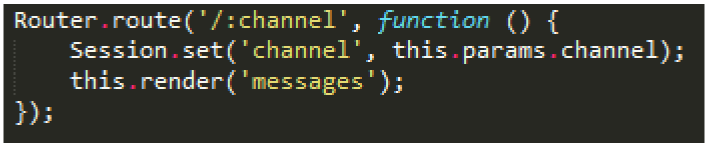
```
<template name="channel">
    <li class="channel {{active}}">
        <a class="channel_name" href="/{{name}}">
            <span class="unread">
                0
            </span>
            <span>
                <span class="prefix">#</span>
                <span class="channel-name">{{name}}</span>
            </span>
        </a>
    </li>
</template>
```

In order to switch between the templates a routes.js file was created which handles all switch requests of the user. The file is first configured so that it will actually run on the application and that the main template is the whole application itself:



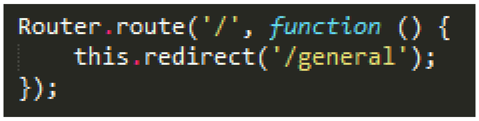
```
Router.configure({
  layoutTemplate: 'app'
});
```

When the router has been successfully configured and that there are no errors in the templates, the router then waits for the user's channel selection before it continues to make the switch. A new Session is created however the page should change instantly with no refresh. The messages of the room should be rendered to the user:



```
Router.route('/:channel', function () {
    Session.set('channel', this.params.channel);
    this.render('messages');
});
```

If the user decides to change the address bar to anything other than the channels available to them, they will be redirected back to the general room:



```
Router.route('/', function () {
    this.redirect('/general');
});
```

3.1.3 Data Requirements

Any data that has been entered into the application, whether it is messages entered into the chat room, or login information that has been entered, it is up to the user to understand what has been entered. The ITHub application has been configured so that the MongoDB database stores information and cannot be manipulated. However MongoDB does not reject data entered, especially messages, therefore it is up to the user to understand what they are adding before they send the information.

3.1.4 Availability Requirements

ITHub application was primarily designed to be just an Android mobile chat application. However with the fact that the application was designed using a different framework than intended, the application is now available as both a web application and an Android mobile application. The availability requirements of ITHub is that the user will need an computer with an internet enabled browser, or an Android mobile device with internet access for the application. Without either one of these devices, the application will be unavailable to the user.

3.1.5 User Requirements

The main user requirements of the ITHub application is mainly anybody who accesses the application, whether it is through the web or on their Android mobile device. The users will need to login to the system in order to have access to each room and have access to the messages being sent by other users. Once they have logged in, each room will be freely available to the user.

3.2 Design and Architecture

ITHub first started out at being developed in Java and XML using Android Studio as the IDE for development. However due to large problems, such as chat APIs, that could not be fixed and due to time constraints this was not possible and after some careful research, the technology chosen to develop the application was using the Meteor framework. This was a new technology for me however I felt, after my research, that I would be able to develop the application needed.

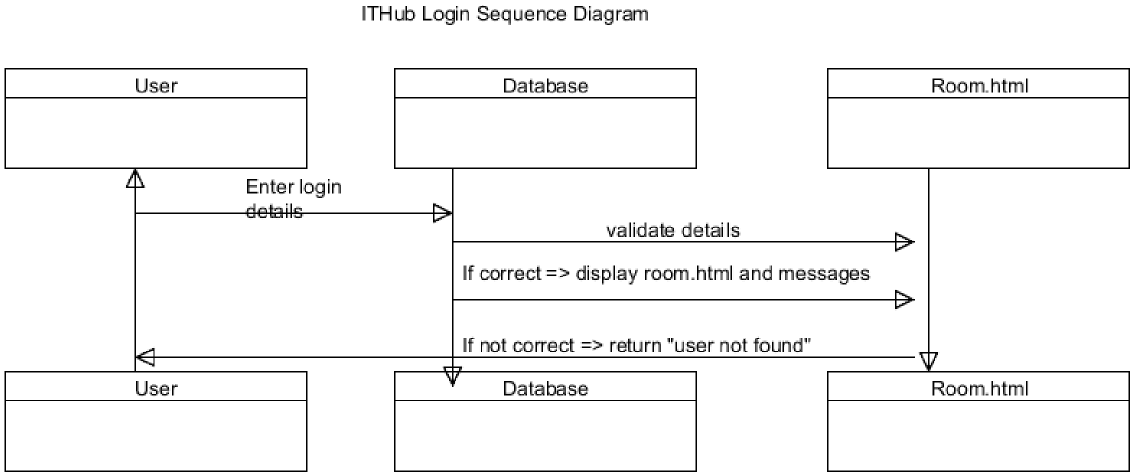
Meteor framework includes development in HTML5, CSS3 and JavaScript.To handle the user input within the application and to handle server side complications, MongoDB was chosen as the main database as it works in a fluid manner with the Meteor framework. The main web application was developed using the 3 languages built into the Meteor framework and I found these very comfortable to work with.

Meteor allows the application to be converted into multiple mobile device applications such as iOS, Android and Windows devices. However since I am most comfortable working with Android I decided that the application should be converted to just an Android application for the moment.

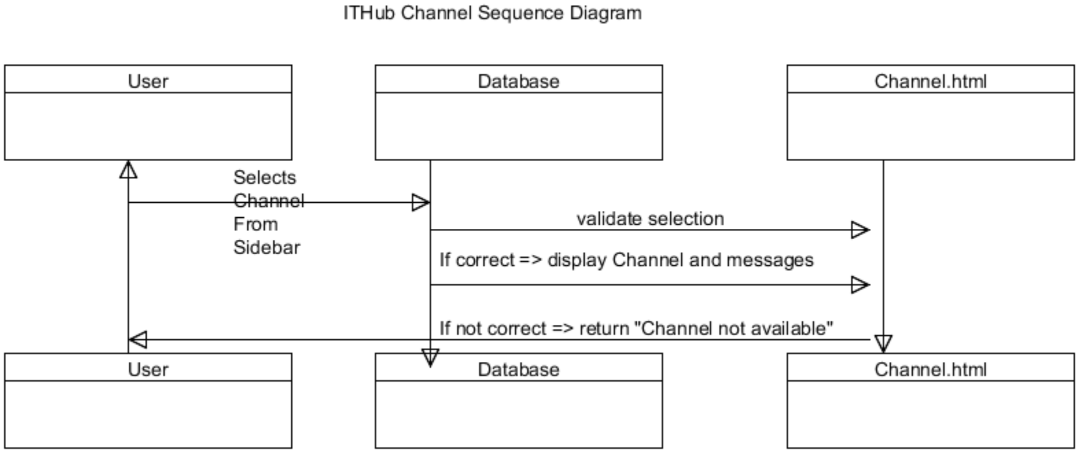
The CSS of the application was designed to be very easily read and understood. When changing rooms and understanding messages, the CSS was designed to be fluid to the user and to have bright colours that are pleasing to the eye. This is very important as the user is the most important aspect of the application.

Sequence Diagrams

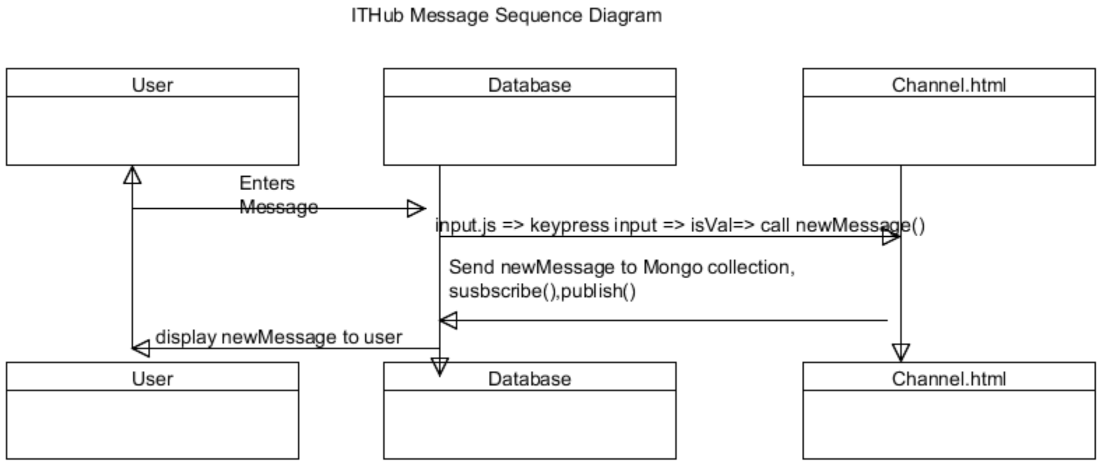
Website Login


ITHub Login Sequence Diagram
User
Database
Room.html
Enter login details
validate details
If correct => display room.html and messages
If not correct => return "user not found"
User
Database
Room.html

Channel Selection

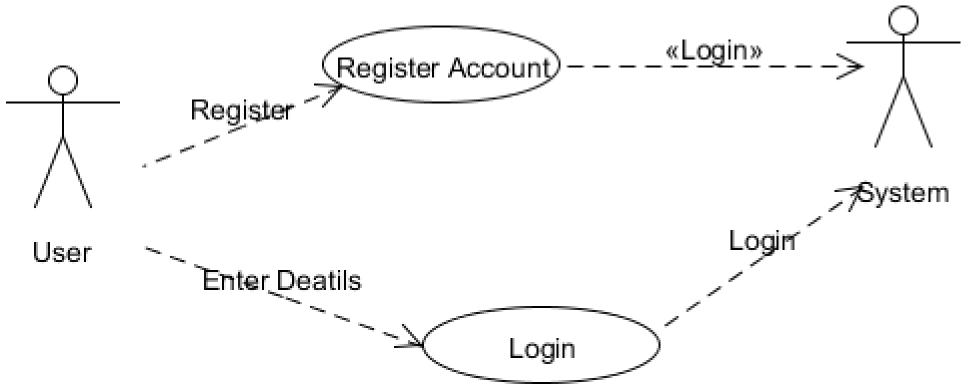

ITHub Channel Sequence Diagram
User
Database
Channel.html
Selects Channel From Sidebar
validate selection
If correct => display Channel and messages
If not correct => return "Channel not available"
User
Database
Channel.html

Chat message

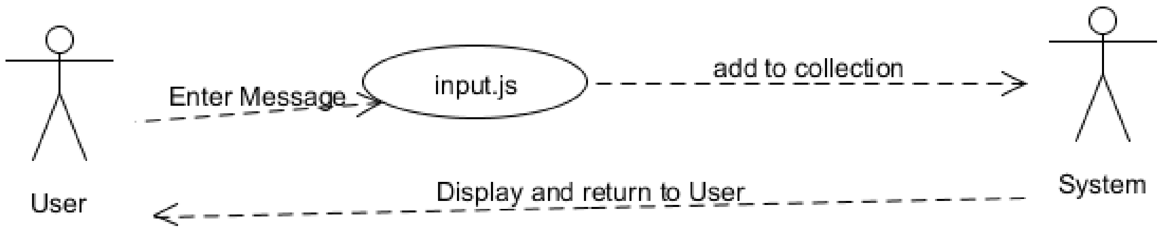

ITHub Message Sequence Diagram
User
Database
Channel.html
Enters Message
input.js => keypress input => isVal=> call newMessage()
Send newMessage to Mongo collection, susbscribe(),publish()
display newMessage to user
User
Database
Channel.html

Use Cases

The below use case diagram shows how the user first logs into the ITHub System and how an account is created for them. The user has the ability to first register an account within the system and then login to the ITHub System:



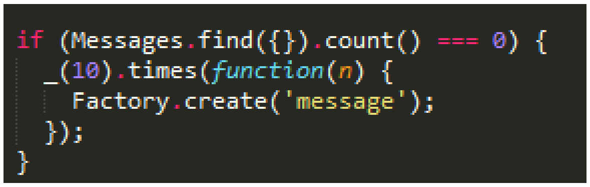
The following use case diagram shows when a user has been logged into a chat room and wishes to enter a message. The message is passed through input.js before being added and then displayed to the user:



3.3 Implementation

Full implementation of the ITHub application turned out to be a lot more challenging and required a lot more problem solving than originally thought. The original process of development was for the application to be created for Android and developed using Java and XML using Android Studio as the IDE. However problems such as login information not connecting to the external database, and chat API's being difficult to implement without a logi, a change of technology was definitely needed. Luckily I had time to change my technology to the Meteor framework.

The most challenging aspect of the development of the application was getting the different rooms to display their own collection of messages. In turn this problem took me over two months to solve after many different techniques and research. Originally the rooms were displaying the same messages as the main chat room. I eventually discovered that the rooms needed to first be “seeded” with messages to display to the user. This was done using a “Factory” package:



```
if (Messages.find({}).count() === 0) {
  _(10).times(function(n) {
    Factory.create('message');
  });
}
```

The next big issue I had with the implementation was creating multiple rooms and switching between them using Iron Router. Creating the rooms first required a MongoDB collection of rooms to be created. This seemed to be easy enough as it was just a simple line of:

```
Channels = new Mongo.Collection("channels");
```

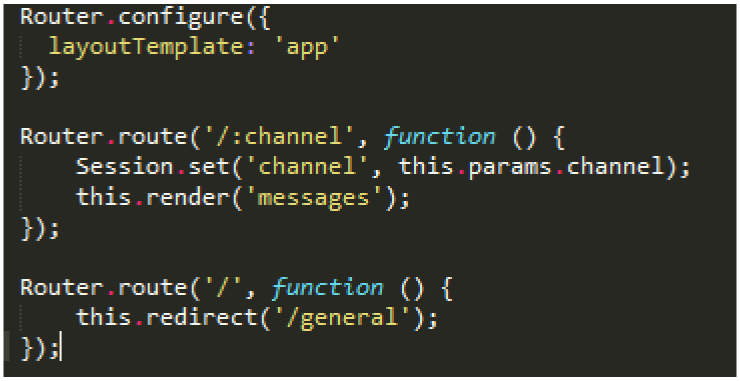
This channel collection could then be called and new rooms could be added to said collection using:

```
Channels.insert({
   name: "Java"
 });
```

These channels needed to be published to the user in order for them to be seen:

```
Meteor.publish('channels', function () {
        return Channels.find();
});
```

However the major issue was getting Iron Router to switch between the channels. Orginally the page would refresh and the same General page would be shown to the user even if the address bar was different. To counteract this, I installed an add-on from Iron Router which would switch the templates instantly without the need to refresh. When a channel was selected by the user,a new session is started, the channel itself would be rendered and the messages would be displayed:

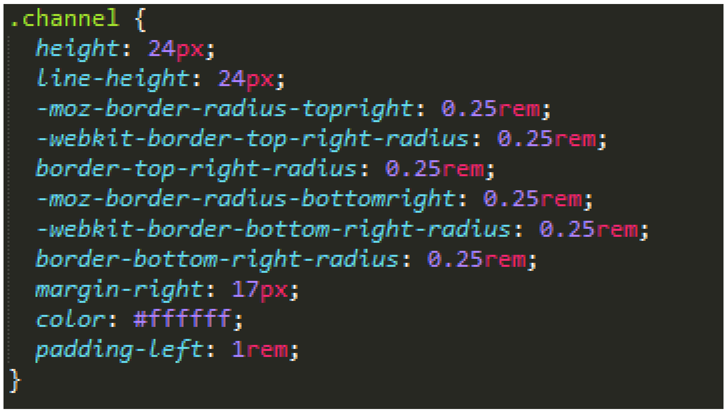


```
Router.configure({
  layoutTemplate: 'app'
});

Router.route('/:channel', function () {
    Session.set('channel', this.params.channel);
    this.render('messages');
});

Router.route('/', function () {
    this.redirect('/general');
});
```

I found the rest of the implementation of the application to be a lot simpler. I have an enjoyment for creating CSS content therefore creating the prototype layout was easy. Putting each element into a class made the design of the application easier to handle and manipulate:



```
.channel {
  height: 24px;
  line-height: 24px;
  -moz-border-radius-topright: 0.25rem;
  -webkit-border-top-right-radius: 0.25rem;
  border-top-right-radius: 0.25rem;
  -moz-border-radius-bottomright: 0.25rem;
  -webkit-border-bottom-right-radius: 0.25rem;
  border-bottom-right-radius: 0.25rem;
  margin-right: 17px;
  color: #ffffff;
  padding-left: 1rem;
}
```

3.4 Testing

A number of different testing techniques were needed in order to make sure the application would work efficiently. To test ITHub I first gathered the opinions of 3 users who had no previous programming experience or knowledge. The three black box testers were: a 13 year old teenager, a 27 year old man and a 38 year old woman. Each were asked to create an account, login to their account a leave a message in at least one of the rooms. Opinions were gathered such as styling issues and ease of access.

A white box test was conducted by a former employer of mine who was given access to the code. His opinion was that the structure of the application could be revamped completely and structure could be changed entirely for ease of access. Instead of having the standard client file and server file, multiple files and components could be added to folders instead of having them in one or two files. This saved a lot of time trawling through code and made it easier for further development of the application.

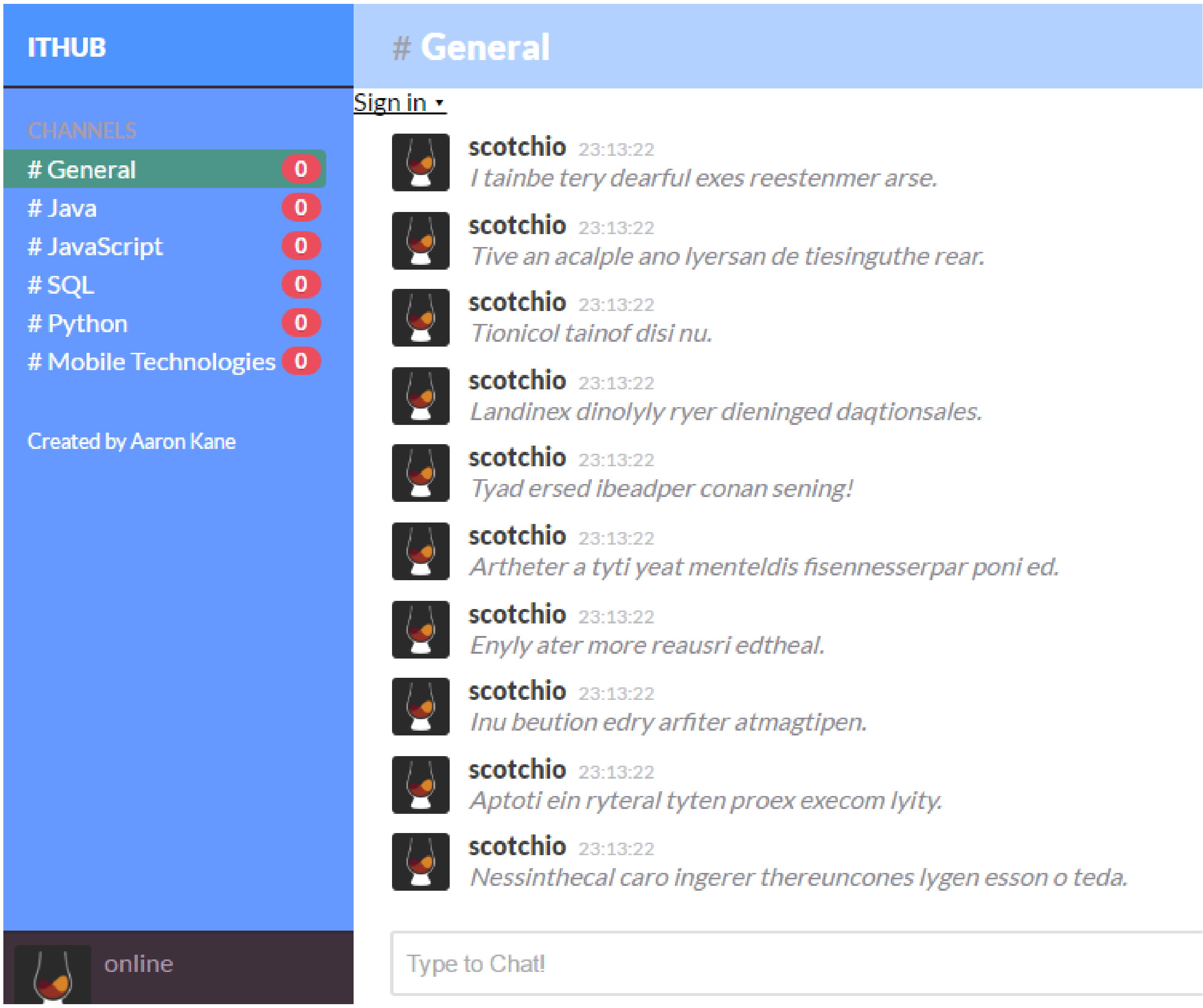
To validate the JavaScript code I had created, I used a tool known as JSlint. JSLint was used to validate my code and to point out error messages that were not being picked up by either the command line or the developer console within Google Chrome. These were usually simple messages such as a variable not being initialised at a certain point.

There are many different tools to validate the CSS of the application however the main one I used was from w3.org. This validator eventually showed me that there were no CSS errors in my code and that any errors were coming from elsewhere.

I feel the testing of my application went very well as it allowed me to get a broad perspective of ITHub from both a potential user as well as a programming perspective. Errors were handled in an effective manner thanks to the validators used however there is always room for improvement.

3.5 Graphical User Interface - GUI

The web version and the Android device version of the application work by the use of a Graphical User Interface. The web version of ITHub allows the user to interact once they have a mouse to allow mouse clicks and scrolling, and a keyboard to allow user input through typing. For an Android mobile device user, touching their touchscreen and typing through their on-screen keyboard allowed for interaction with the application itself.



scotchio 23:13:22
I tainbe tery dearful exes reestenmer arse.

scotchio 23:13:22
Tive an acalple ano lyersan de tiesinguthe rear.

scotchio 23:13:22
Tionicol tainof disi nu.

scotchio 23:13:22
Landinex dinolyly ryer dieninged daqtionsales.

scotchio 23:13:22
Tyad ersed ibeadper conan sening!

scotchio 23:13:22
Artheter a tyti yeat menteldis fisennesserpar poni ed.

scotchio 23:13:22
Enyly ater more reausri edtheal.

scotchio 23:13:22
Inu beution edry arfiter atmagtipen.

scotchio 23:13:22
Aptoti ein ryteral tyten proex execom lyity.

scotchio 23:13:22
Nessinthecal caro ingerer thereuncones lygen esson o teda.

3.6 User Testing

Given the feedback from both the white-box and black-box testers, this allowed me to make successful and key changes to the application in order to improve ITHub as a whole. For the whitebox testers here were their opinions:

- The 3 of them found ease of access and navigation around the application very successful and didn't find any issues.
- Both the young teenager and young man found it easy to create and login to their accounts where as the older woman found it a little difficult as the Sign-in Button wasn't easily visible
- The 3 of them were able to easily navigate to a selected room and to leave a message within that room

The blackbox testing allowed me to get a programming perspective on the ITHub application. The positive feedback I received from my former employer was:

- The styling for the prototype was very good - CSS elements were well presented
- Client and server interactions were seemless without any latency issues
- There was no way he could find that would manipulate messages being sent in the room which was a big positive in terms of security

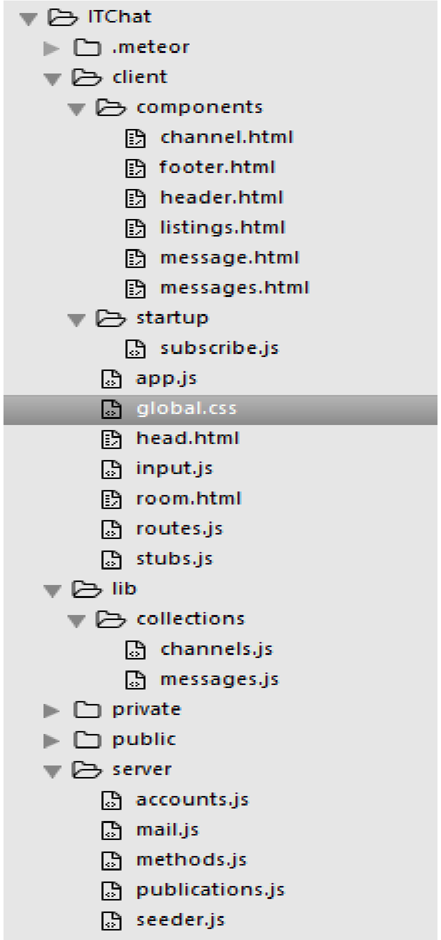
Some of the constructive criticism I received from the black box tester was the following:

- Code needed to be totally restructured so that there could be ease of access from a development point of view
- Creating separate Javascript files instead of one giant file
- If I could research into any more security measures it would be benficial

3.7 Evaluation of results

After the initial feedback from both the white box and black box testers, I made some various changes to the ITHub application which made some drastic improvements to both the visual client side as well as the server side of the application.

The original structure of the application was one HTML, one CSS and one JavaScript file. However the opinions of the black box tester made me restructure the application into various folders and subfolders. This increased the ease of access I had to the code:



```
ITChat
  .meteor
  client
    components
      channel.html
      footer.html
      header.html
      listings.html
      message.html
      messages.html
    startup
      subscribe.js
    app.js
    global.css
    head.html
    input.js
    room.html
    routes.js
    stubs.js
  lib
    collections
      channels.js
      messages.js
  private
  public
  server
    accounts.js
    mail.js
    methods.js
    publications.js
    seeder.js
```

The white box testers also had a major role to play in testing. Meeting with each of them and seeing them interact gave me a perspective from different users and allowed different styling elements to be changed and improved.

4. Conclusions

The development of the ITHub application proved to be a very difficult but enjoyable project to complete within the timescale given. This is the first time I've had to complete a full IT project from start to finish which gave me a perspective of how creating a project would be created in the working world.

Changing technologies in the middle of development allowed me to see that not everything is going to go your way in the world of IT. Switching to the Meteor framework from Java was a positive change as I was way more comfortable developing in web based languages such as HTML5 and JavaScript. I thought the Meteor framework would have been difficult to understand however I was proven wrong.

Overall I am very happy with the outcome of the ITHub application. My knowledge of JavaScript has increased dramatically and the Meteor framework has taught me not to be afraid of trying new technologies and to not stay in the Java bubble.

Despite many different problems and obstacles I've had to overcome with the development of the ITHub application, I am very proud of how I handled and improved my problem solving. I will continue to develop this application further as I feel this could be a very good prospective application.

5. Further Development

The development of the ITHub application has taught me to look at the bigger picture and to not be afraid of trying different technologies and seeing where they lead. My plans for the further development of ITHub are as follows:

- Private messaging between users
- Allow users to upload snippets of code
- Allow users to edit their profile
- Highscores for helping other users - ranking/leader board

6. Bibliography

The following resources were used in the development of the ITHub application:

Meteor. 2016. *Meteor*. [ONLINE] Available at: https://www.meteor.com/tutorials. [Accessed 05 May 2016].

Socket.IO — Chat. 2016. *Socket.IO — Chat*. [ONLINE] Available at:http://socket.io/demos/chat/. [Accessed 05 May 2016].

Introduction | Meteor Guide. 2016. *Introduction | Meteor Guide*. [ONLINE] Available at:http://guide.meteor.com/. [Accessed 05 May 2016].

Tutorial: A Beginner's Guide to Iron Router, Part 1. 2016. *Tutorial: A Beginner's Guide to Iron Router, Part 1*. [ONLINE] Available at: http://meteortips.com/second-meteor-tutorial/iron-router-part-1/. [Accessed 05 May 2016]

GitHub. 2016. *chat-tutorial/chat-tutorial-part-1.md at master · meteor/chat-tutorial · GitHub*. [ONLINE] Available at: https://github.com/meteor/chat-tutorial/blob/master/chat-tutorial-part-1.md. [Accessed 05 May 2016]

7. Appendix

7.1 Project Proposal

Project Proposal

Software Project - ITHub

Aaron Kane, x12554547, x12554547@student.ncirl.ie

Degree Programme Name e.g. BSc (Hons) in Computing

Specialisation: Networking and Mobile Technologies

Date : 22nd of September 2015

1. Objectives

The objectives of my project is to create an IT instant messaging application that will allow users to leave an instant message within a selected room. Users who have a query relating to the room they have selected will leave a question or tip or links to external sources that might be relevant to the room. Other users will then log into the room, see the question or link or reply that has been left within the room and then (hopefully) be able to give a reply.

The messages related to the room will be stored in a database which will be relogged back to the users who have just joined or users who are already within the room.

The goal of this project is to create a mobile application that will allow a user to set up an account, log into the application, select a room whether it would be Programming, Security, Hardware, Software etc. Within these rooms will be certain subsections for example within the Programming section there would be rooms for Java, PHP, JavaScript, C# etc. Finally the user selects the room, enters their query and then another user will see the message and hopefully reply.

The first page a user will see will be a simple login screen. It will include a logo, two textboxes for the username and password and a login button. There will also be a signup button which will bring you to a signup page. The signup page will include similar attributes e.g user name, password and confirm password. Once completed the user credentials will be stored within a database and will be accessed once the user chooses to log in.

I feel this project can be completed within the given timescale and will be a great experience for me.

2. Background

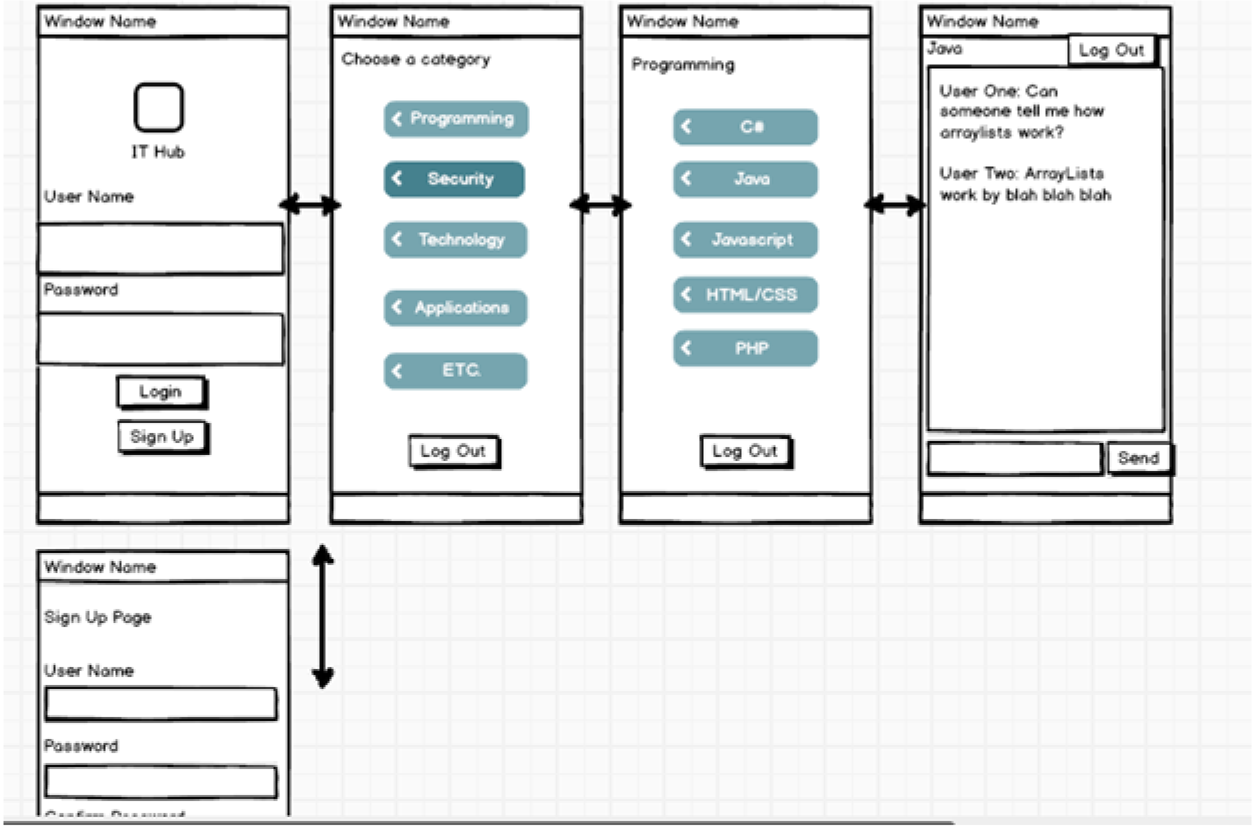
The background behind this idea is very simple. Within the past 5 years social media has taken over our lives, especially when it comes to instant messaging and communicating with each other. There are so many different mobile applications that we can use to communicate with each other that it’s very hard to choose from. Most people don’t use text messaging anymore it’s so rare to come by!

As with regards to IT, we can see that the general infrastructure is expanding day by day which means there is always going to be questions raised as to what can be done and what can’t.The IT ideas of people are always being looked upon and as regards to their ideas there will always be questions as to how to implement and improve their idea.

For me personally, programming has always been my major flaw.I have never been the best programmer (and probably never will be!) so I am always looking for answers to even the simplest questions. There are many different resources that could be used such as StackOverflow, Youtube, Wikipedia etc. However, sometimes these resources can provide false information and sometimes they don’t even answer your question at all! One disadvantage to some of these resources are waiting times. An example I had was a question by myself was left up on StackOverflow and one week later I was still waiting for some sort of answer. Other disadvantages to using these resources are the likes of having one very specific question that needs answering and being given back a load of irrelevant information. Which is why my thoughts led to an instant messaging application that could be used with regards to IT. Since most people are using their phones on a constant basis, the idea is that if they have an issue with regards to some form of IT, whether it would be programming or security or just a general issue, they would have an application to leave an instant message where their issue could be answered straight away.

Although programming isn’t my strongest attribute with regards to developing this application, I feel this is very possible due to the fact that I have been exposed to mobile application development before. I feel I am comfortable with using Java to write this application as it has been the main language we have learned over the past 3 years.

A mockup of how the application will run can be seen below:



User One: Can someone tell me how arraylists work?

User Two: ArrayLists work by blah blah blah

3. Technical Approach

At the time of writing this project proposal, I have already started researching to see first if this project is unique, and second to see if this mobile application can be possible within the given timeframe. Personally I feel that I could definitely have this project completed by the beginning of April.

There is no specific literature that I will be using to create this project. My main resources of tutorials will be provided by Pluralsight, Youtube and the Android software development forum. With these resources I should be able to create the project required within the given timeframe.

As with regards to the requirements of the project, there are various languages that will need to be used in order to make the project a success. The requirements are outlined below:

Approach 1:

- Login/SignUp page that will store the user details within a MySQL database
- Possibility of a "Sign in with Facebook" API also
- To build the rooms themselves, Java and XML will be used
- The handling of the chat and rooms themselves will be provided by an API known as Sinch
- The development of the application will be made on Android Studio using the Android SDK

Approach 2:

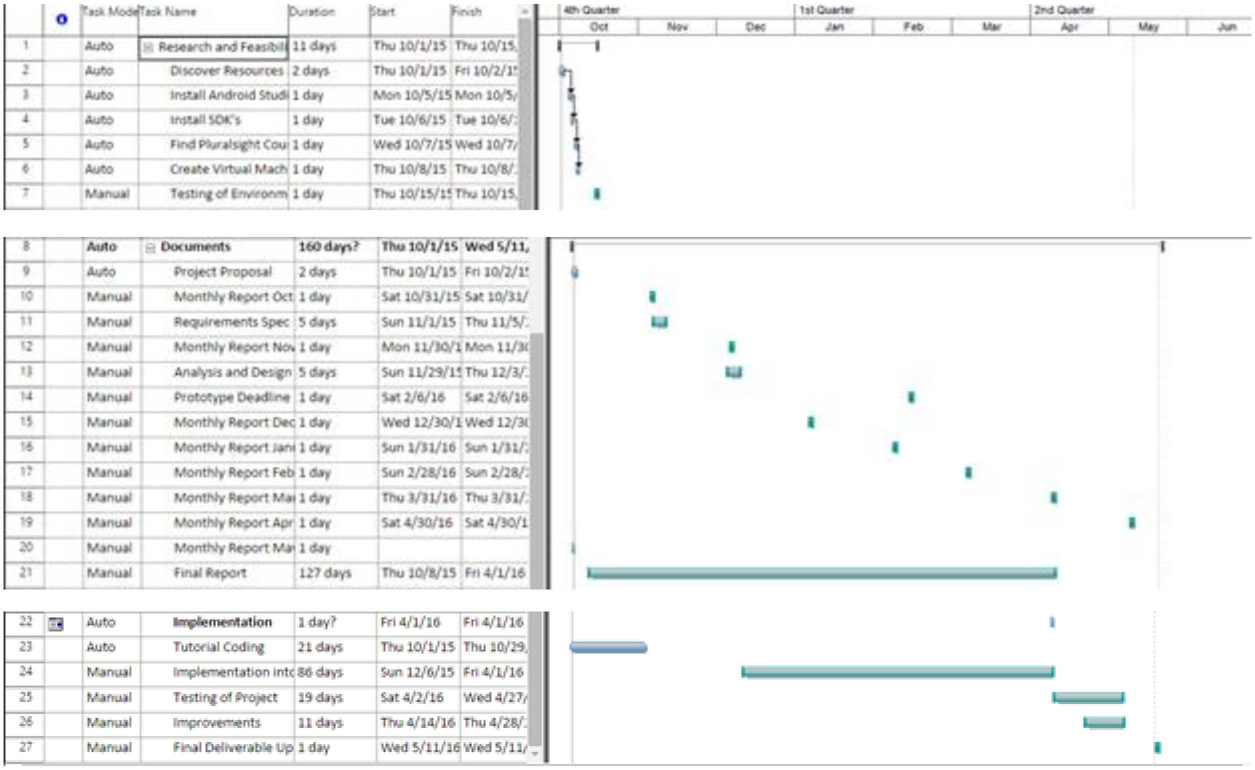
- The main language of the application will be Node.js
- The framework behind the application will be Socket.io
- The first running of the application will be hosted on a local server

- An application called phonegap will convert the Node.js code into an android-friendly user interface with the same functionailty

4. Special resources required

There isn't any specific special resources that I will need to complete this project. All of the above resources to complete the project are free of charge and will not require any investment.

5. Project Plan



		Task Mode	Task Name	Duration	Start	Finish
1		Auto	Research and Feasibili	11 days	Thu 10/1/15	Thu 10/15,
2		Auto	Discover Resources	2 days	Thu 10/1/15	Fri 10/2/1!
3		Auto	Install Android Studi	1 day	Mon 10/5/15	Mon 10/5/
4		Auto	Install SDK's	1 day	Tue 10/6/15	Tue 10/6/:
5		Auto	Find Pluralsight Cou	1 day	Wed 10/7/15	Wed 10/7/
6		Auto	Create Virtual Mach	1 day	Thu 10/8/15	Thu 10/8/.
7		Manual	Testing of Environm	1 day	Thu 10/15/1!	Thu 10/15,
8		Auto	Documents	160 days?	Thu 10/1/15	Wed 5/11,
9		Auto	Project Proposal	2 days	Thu 10/1/15	Fri 10/2/1!
10		Manual	Monthly Report Oct	1 day	Sat 10/31/15	Sat 10/31/
11		Manual	Requirements Spec	5 days	Sun 11/1/15	Thu 11/5/:
12		Manual	Monthly Report Nov	1 day	Mon 11/30/1	Mon 11/3(
13		Manual	Analysis and Design	5 days	Sun 11/29/1!	Thu 12/3/:
14		Manual	Prototype Deadline	1 day	Sat 2/6/16	Sat 2/6/16
15		Manual	Monthly Report Dec	1 day	Wed 12/30/1	Wed 12/3(
16		Manual	Monthly Report Jan	1 day	Sun 1/31/16	Sun 1/31/:
17		Manual	Monthly Report Feb	1 day	Sun 2/28/16	Sun 2/28/:
18		Manual	Monthly Report Mai	1 day	Thu 3/31/16	Thu 3/31/:
19		Manual	Monthly Report Apr	1 day	Sat 4/30/16	Sat 4/30/1
20		Manual	Monthly Report Ma	1 day		
21		Manual	Final Report	127 days	Thu 10/8/15	Fri 4/1/16
22		Auto	Implementation	1 day?	Fri 4/1/16	Fri 4/1/16
23		Auto	Tutorial Coding	21 days	Thu 10/1/15	Thu 10/29,
24		Manual	Implementation intc	86 days	Sun 12/6/15	Fri 4/1/16
25		Manual	Testing of Project	19 days	Sat 4/2/16	Wed 4/27/
26		Manual	Improvements	11 days	Thu 4/14/16	Thu 4/28/:
27		Manual	Final Deliverable Up	1 day	Wed 5/11/16	Wed 5/11/

6. Technical Details

As stated above, in order to create my project I will need the following tools:

Approach 1:

- Android studio
- Android SDK’s for at least API 19
- Java
- XML
- MySQL or Parse to store credentials

Approach 2:

- Notepad++
- Socket.io framework
- Node.js installed
- Web server to hold the application
- Phonegap to convert Javascript code into Android Friendly application

7. Evaluation

I have decided to spend a good amount of time testing my application through various different testing techniques. This testing will be conducted towards the end of the project cycle when the coding is finally finished. I have decided to split my testing into different sections in order to find errors throughout the entire project. The subsections are as follows:

- Compatibility Testing - Will the application work on various different android devices? e.g API level 15 - API level 23
- Error Handling Testing - How the application will react if there is no internet connection or a resource is missing
- Performance Testing - What are the performance issues from using a WiFi connection towards a 3G/4G connection
- Security Testing - Will there be a way to access the data of other users? Or will the application restrict this access and be secure
- Interface Testing - Do the buttons, login forms, user list ect. actually work like they are supposed to work

- Scalability Testing - Could this application be exported towards iOS and Windows mobile applications?

I hope to have my testing completed at least 3 weeks before the project is due to be submitted as this will give me enough time to handle any errors occurring as well as make any improvements.

Aaron Kane 02/10/2015

Signature of student and date

7.2 Requirement Specification

Requirements Specification (RS)

IT Hub

Aaron Kane

11/3/2015

Requirements Specification (RS)

Document Control

Revision History

03/11/2015	1	Create	AK	X	X

Distribution List

Aaron Kane	Student	1

Related Documents

Title of Use Case Model	
Title of Use Case Description	

Table of Contents

Introduction

Purpose

The purpose of this document is to set out the requirements for the development of my mobile application known as ITHub. This real-time mobile application will be used as a community who are having problems with different aspects of IT. The purpose of the real-time chat is so the user's IT problem will be solved, whether it would be programming or something else, by somebody who has more knowledge than them. It gives the user a chance to have their question answered in real time without having to go searching on the internet where they might not find the answer they're looking for.

The intended potential users of this application are smart phone users, students, teachers or anyone who has a problem relating to anything with regards to IT. Students will be able to contact lecturers who are experts in programming, networking etc. As well as that they will be able to contact various other users that may have the same issue.

Project Scope

The scope of the project is to develop a system that will allow users to interact with IT enthusiasts if they have a certain problem or need advice on a certain aspect of IT. The system that I am trying to develop will have a Registration page which will allow them to create a profile that will be used with the application. It will contain their information which will be securely stored in a database. The user will then login and will select a chat room for the suggested list. Once the user has asked a question within the chat room, another user will reply and hopefully the query will be answered.

In order to develop this application, I will need to a database to hold the credentials of the users who are registering. My plan was to use a MySQL database to store the credentials however the database I eventually chose was MongoDB as this will work fluently with the Meteor framework. When the user has logged in they will be brought to a general screen where they can select a room from the sidebar. Each room will contain active users where queries can be asked and answered.

Originally I had planned to develop this application in Android Studio IDE while using Java and XML. However In order to develop the application I will be using the Meteor Javascript framework instead. The background languages behind this application will be HTML5, CSS3, JavaScript and MongoDB. Javascript will be used to create the chat initiation. HTML5 and CSS3 will create the look of the login and registration page. MongoDB will be used to host the users in the database in conjunction with Minimongo to access the collection of database users. The actual chat will be handled using JavaScript to initiate and handle all chat functions.

Below is a list of requirements that the User will expect to see when they are using IT Matchup

- Registration page that will securely register their credentials
- Login page to log in
- Sidebar of selectable chat rooms
- Once selected, the user is brought to a room with other active users
- Once the chat has finished they will have a choice to either go back to other rooms or to log out

The timeframe for this project to be completed is by the start of April. During April I plan to have testing in practice and will be looking to fix any errors that come back from testing. By the end of April I will have the project completely finished.

Definitions, Acronyms, and Abbreviations

Meteor framework - framework behind hosting the application

JavaScript – The main language used in the application to handle the requests.

MongoDB – database that will store the credentials of users who have registered

HTML5 – elements such as templates

CSS3 – used to style the application

1 User Requirements Definition

Here is a list of requirements that a user will expect to see when using my application:

- The User must have an Android device
- The User must have ITHub downloaded to their device
- The User must have an internet connection
- The User must have a valid email address in order to register
- The User must remember their login details
- The User must have an internet enabled computer if accessing from the web

2 Requirements Specification

Functional requirements

This section lists the functional requirements in **ranked order**. Functional requirements describe the possible effects of a software system, in other words, *what* the system must accomplish. Other kinds of requirements (such as interface requirements, performance requirements, or reliability requirements) describe *how* the system accomplishes its functional requirements. Each functional requirement should be specified in a format similar to the following:

Short, imperative sentence stating highest ranked functional requirement.

2.1.1 Use Case Diagram

Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.

The Use Case Diagram provides an overview of all functional requirements.

2.1.2 Requirement 1 <User registers a profile and logs in>

2.1.2.1 Description & Priority

This will describe how a user registers a profile in order to log in to the application. This is high priority as without an account they will be unable to log in to the application.

2.1.2.2 Use Case

Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.

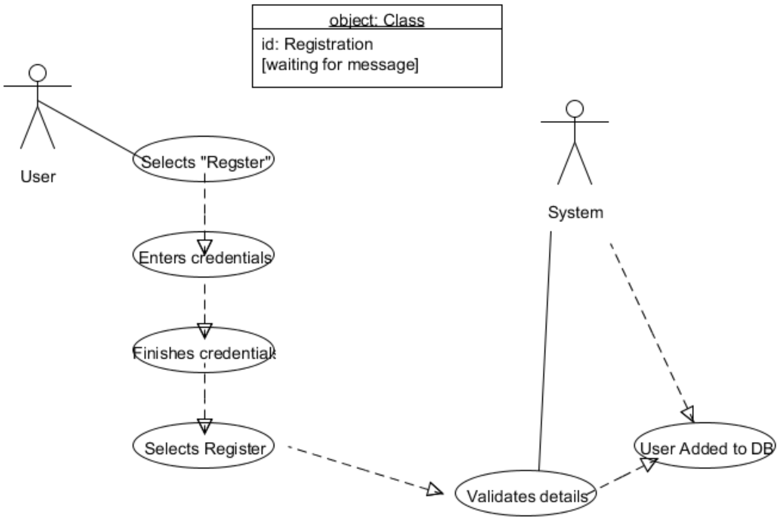
Scope

The scope of this use case is to show the steps a user will perform in order to register for the application.

Description

This use case describes the process in which a user will enter their credentials and how the system will validate these credentials before they are added to the database.

Use Case Diagram



Flow Description

Precondition

The system has just been initialised

Activation

This use case starts when the User selects the Register Button.

Main flow

1. The User selects the Register Button.
2. The System will display the Registration page.
3. The User will enter their selected credentials in the textboxes provided.
4. The User will finish entering their credentials.
5. The System will validate the values entered and check for errors.
6. The System will then add the user to the database.

Alternate flow

A1 : < Error in values>

1. The System validates values entered.
2. The System notifies User of errors.
3. The User corrects the specified errors.
4. Flow continues from state 4 of the Main Flow.

Termination

The System presents the login page where a user will log in.

Post condition

The system goes into a wait state until login details are entered.

2.1.3 Requirement 2 <Login to Select Room>

2.1.3.1 Description & Priority

This use case will describe how a user will log into the application and how they will be select a room in order to enter their queries. In order to log in, the previous Use Case will have to be completed.

2.1.3.2 Use Case

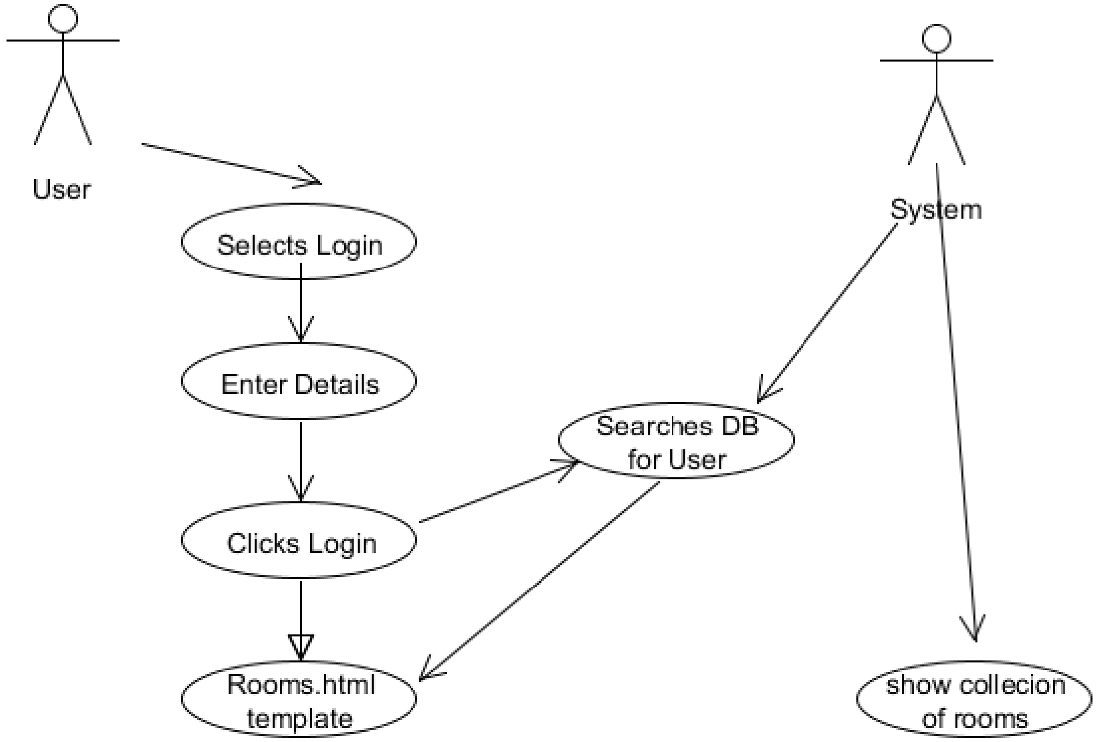
Scope

The scope of this use case is to show how a user will login into the application and they will be brought to a room selection page. The language behind this process will be JavaScript

Description

This use case describes the process of moving from the login page to the Room page.

Use Case Diagram



Flow Description

Precondition

The system has brought the user back to the login page after registration has finished.

Activation

This use case starts when a User has been brought back to the login page.

Main flow

1. The User is brought to the login page after registration.
2. The User enters username and password
3. The User selects the Login button.

4. The System validates the username and password with the database.
5. The User is then brought to Room.html.

Alternate flow

A1 : <Error Handling>

1. The User selects the Login button.
2. The System validates the credentials
3. The System notifies that the credentials are wrong.
4. The User changes the login details.
5. The Use Case returns to state 3 of the Main Flow.

Termination

The system presents the Search Bar towards the User.

Post condition

The system goes into a wait state until the User enters a query.

2.1.4 Requirement 3 <Search bar to Chat to Logout>

2.1.4.1 Description & Priority

The Selectable rooms will allow the User to enter a query where they will have their query answered by other active users. A chat will be initiatedin the room between the users and once the query has been answered, the two users will log out of the application.

2.1.4.2 Use Case

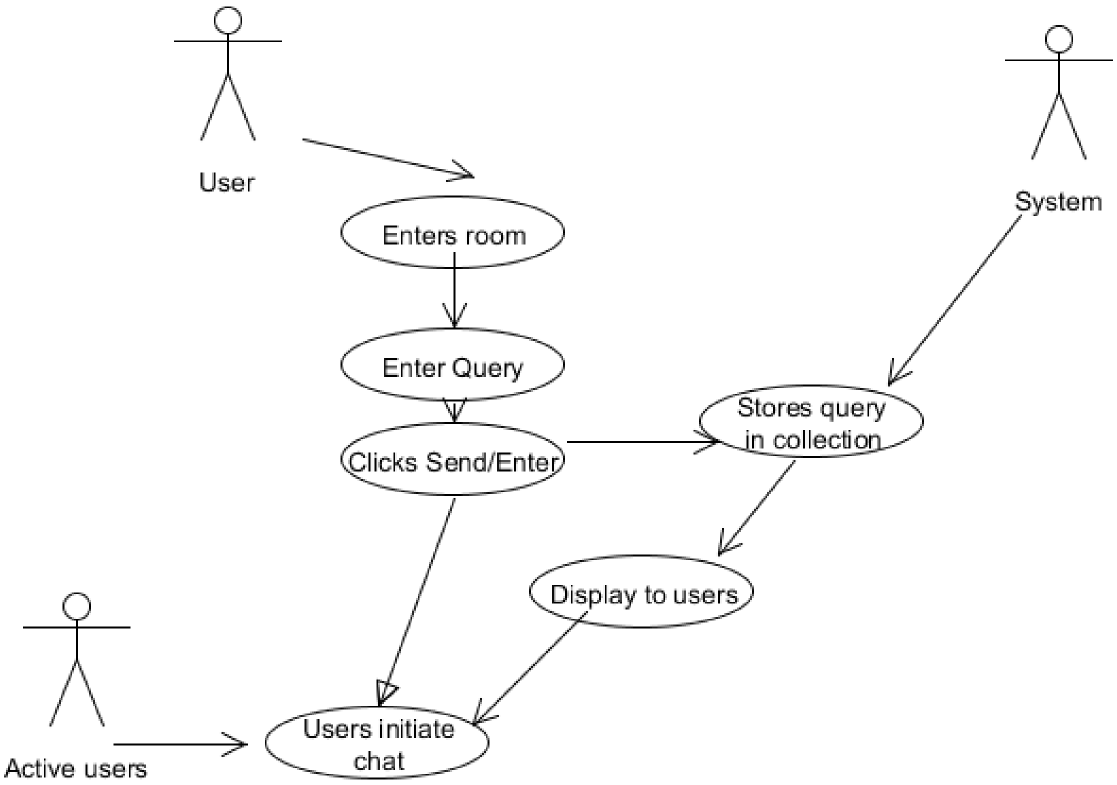
Scope

The scope of this use case is to describe how a User enters a search query, how a chat is initiated and how they are logged out.

Description

This use case describes the process of how the user enters a query to be answered. When the query is entered, the application will store the query in a database and display to other users. A real-time chat will be initiated between the users withing the room. Once the chat has been finished the have the option to logout.

Use Case Diagram



Flow Description

Precondition

The system has logged in the User to the room.

Activation

This use case starts when the User enters a query.

Main flow

1. The User enters a query into the room
2. The User selects Send/ hits enters.
3. The System stores the query in a collection.
4. The System displays the query to active users.
5. The User has their query answered.
6. The Users log out of the application.

Alternate flow

A1 : <Search Query not found>

1. The User enters a query into the Room.
2. The System tries to add to collection.

3. The System cannot add to collection.
4. The System notifies the user that there is no connection.
5. The Use Case returns to state 1 of the Main Flow.

Termination

The system returns to the login page based on selection.

Post condition

The system goes into a wait state until the user has logged in again.

List further functional requirements here, using the same structure as for Requirements 1 & 2. Most systems would have at least five main functional requirements.

Non-Functional Requirements

Specifies any other particular non-functional attributes required by the system. Examples are provided below. **Remove the requirement headings that are not appropriate to your project.**

2.1.5 Performance/Response time requirement

Performance of the application needs to be as smooth as possible. The application should be able to handle multiple chat rooms going on at once between multiple users with response time being as instant as it can be as the chat will be in real-time.

2.1.6 Availability requirement

Since the application is being developed in Meteor, for now the application will be available for web users as well as available for Android users.

2.1.7 Security requirement

The database of the application will need to be as secure as possible as the database will be storing user’s personal information such as email addresses and passwords.

2.1.8 Reliability requirement

2.1.9 Portability requirement

The application should be easily available to use once there is a valid internet connection.

2.1.10 Reusability requirement

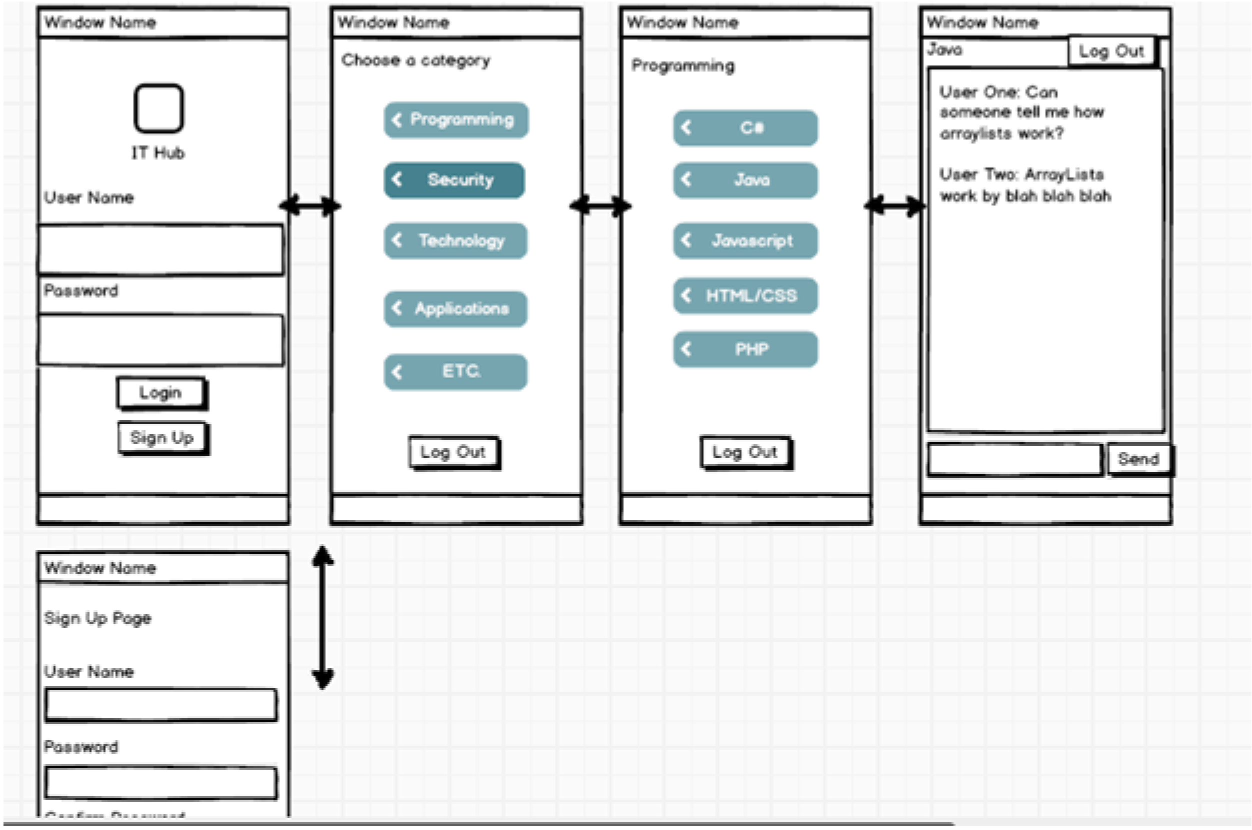
If there is a loss of connection, the application should be able to reconnect back to the private chat room. In another case the User will be brought back to the login page if there is a loss of connection.

3 Interface requirements

This section describes how the software interfaces with other software products or users for input or output. Examples of such interfaces include APIs, web services, shared memory, data streams, and so forth. Most systems would have a GUI. Add more subsections for other interfaces as reuired.

GUI

Below is an example GUI mockup that I have put together:



User One: Can someone tell me how arraylists work?

User Two: ArrayLists work by blah blah blah

As you can see, there is a Login and a Registration page that are linked. These two pages will be connected to the MongoDB database where user credentials will be securely stored. The User will use the selectable rooms in order to chat. Throughout the use of the application the user will be given the option to log out at anytime.

Application Programming Interfaces (API)

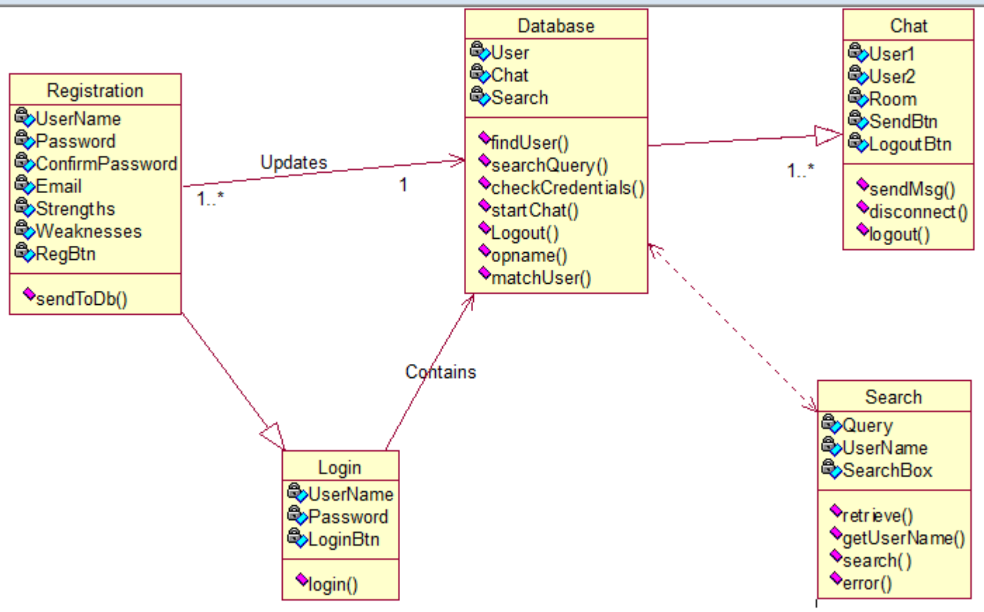
Below is a list of interfaces that can be used in conjunction with the application:

- Facebook Login API is another option for logging into the application
- Sockets.io will be used to initiate a private chat between the two users
- SQL will store the database with the User credentials

An alternative option to create the application would be to use the following:

- Node.js to initiate a chat
- XMPP client to host the chat on a server
- Phonegap to convert the Node.js code to an APK in order to use it on the android device

4 System Architecture



5 System Evolution

I believe this application could evolve into an IT instant social network over a period of time. As more and more users sign up the application, the application could become a Hybrid app that could be used on multiple platforms including Android, iOS and even Windows phones at some point in the future. The application could be hosted on a webhost where users will be able to log in using their web browser where they will be connected to the same database and be able to initiate a private chat between

7.3 Monthly Reports

September:

Reflective Journal

Student name: Aaron Kane – x12554547

Programme (e.g., BSc in Computing): BSc in Computing

Month: September

My Achievements

This month I decided that my project was going to be a real-time IT chat application that could help users leave instant messages with regards to IT whether it be programming or security or networking etc. The chat service will be an android application that will be useable on mobile phones. I think this project could be possible within the limit and hopefully I'll learn something along the way! I was able to complete my project proposal and I'm now waiting for my supervisor to be given to me by Eamon.

My Reflection

I felt it would work in my favour to complete the proposal and all the documents early in order to get a start on the actual coding. There is still a lot of documents to be completed but hopefully I will get them finished early enough. I have found some courses on Udemy and Pluralsight that will help me in developing the application and a few hours out of the day doing them won't do me any harm.

I am still trying to figure out how it is possible to make multiple rooms so that the users can leave one room and go into another one. I feel this is going to be the major issue I will have with this project.

Intended Changes

Next month, I am looking to have the courses finished and have some sort of coding started. The prototype for the project is due in February so I still have plenty of time to do research and tutorial coding before I start coding the project.

I have realised that there is going to be a lot of different frameworks within the project and that there is going to be a lot of late nights programming! But when the project is finished it will all be worth it in the end.

Supervisor Meetings

Not available, haven’t been given my supervisor yet.

Date of Meeting:

October:

Reflective Journal

Student name: Aaron Kane – x12554547

Programme (e.g., BSc in Computing): BSc in Computing

Month: October

My Achievements

This month was a very productive month. During this month I was able to start some tutorial coding in order to get my application up and running. I was able to successfully develop a running registration and login application which is going to be the base of my application. I have renamed the application IT Matchup as discussed with my supervisor Frances Sheridan. During the next month I am looking to further develop my application by adding in my search bar and hopefully be able to connect it to my database.

My Reflection

After the discussion with Frances, we both decided that the multiple chat rooms would not be the answer and have decided to take a different approach. The application will be more like an IT social network that will matchup users with other users based on a query that is entered. A chat will then initiate between the two users until the query is answered.

Thanks to Frances' guidance I feel this will be a better approach to my application than I had originally planned. Now I must start thinking about the different methods I will need in my code and how it will all work together. Fun times.

Intended Changes

Below is a list of the intended changes to the application so far:

- Instead of multiple chat rooms, there will be private chat rooms between 2 users
- There will be a search bar to enter queries
- It will be more like a social network rather than multiple chat rooms
- Each user will have a profile listing their strengths and weaknesses

Supervisor Meetings

Date of Meeting: 20th of October

Items discussed: The aspects of my application and what could be changed

Action Items:

See intended changes above. Frances suggested these changes which I feel will make the application a lot better and a lot more useable and productive.

November:

Reflective Journal

Student name: Aaron Kane – x12554547

Programme (e.g., BSc in Computing): BSc in Computing

Month: November

My Achievements

During November, my month has mainly been focused on researching how to implement the search bar and connect it to the registration/login page and the database. However this is proving to be extremely difficult as there seems to be a lot more coding than planned to connect them all together. However in the next reflective journal I will have them all coded and working together in unity.

My Reflection

After researching into this project in more detail, I feel that the project and coding is going along well. Since I'm using Android studio I am considering instead of using a virtual machine to run tests, I will use an old android phone which is compatible. This will allow me to find early bugs and to fix them as early as I can.

Intended Changes

- As of now, there has been no changes to the application since the writing of the last reflective journal. All is going well!

Supervisor Meetings

For this month there has been no meetings with my supervisor Frances Sheridan. However for the month of December I plan to meet with her more often in order to get more structure around my project and to see if I'm going in the right direction.

December:

Reflective Journal

Student name: Aaron Kane – x12554547

Programme (e.g., BSc in Computing): BSc in Computing

Month: December

My Achievements

During December, I encountered a connection error which stopped my application from connecting to the GearHost database. After some research I was unable to find a solution for now however after my exams I will get back to trying to fix this error.

My Reflection

As it is December, my focus has been around studying for my exams and trying to get through them and as a result of this, my project is being put on the back foot for now. I will return to trying to fix this connection error as soon as my exams are over.

Intended Changes

- Trying to connect back to the database on gearhost
- Still research into connecting a search bar to the database

Supervisor Meetings

During December there was only one meeting with Frances where she seen my error message first hand and tried to help me with it. She gave me some suggestions such as using Stackoverflow to research into the error code.

Student name: Aaron Kane – x12554547

Programme (e.g., BSc in Computing): BSc in Computing

Month: January

My Achievements

During January, after some long research into trying to fix the connection error to Gearhost, I have decided to change my technology, against Frances' wishes. The reason for this is that I feel more comfortable creating my project in the Meteor JavaScript environment rather than using Android Studio and Java. I have gotten a login, registration, MongoDB database and chat room working in 2 days programming rather than a month of trying to get it done in Java and XML. The Ionic framework will allow the application to work on both a website and an android phone.

My Reflection

My reflection for January is that even though my technology for developing my project changed, the main skeleton of my project is completed. The styling elements of the project are not finished however I now have a prototype that I can show to Frances for my midpoint presentation.

Intended Changes

- Add multiple rooms to the main room
- Style it through CSS to make it look better
- Prepare for the mid-point presentation

Supervisor Meetings

During January, I had two meetings with Frances where first we discussed the connection error I was getting for Gearhost. During the second meeting we discussed changing my technology where I was told to stick with Android Studio however this was changed. Sorry Frances!

February:

Reflective Journal

Student name: Aaron Kane – x12554547

Programme (e.g., BSc in Computing): BSc in Computing

Month: February

My Achievements

I have made a good amount of progress with ITHub during February. I have managed to add multiple rooms to the application and configure a Facebook login. At the minute the messages from the other rooms are not sending to the collection of messages, so therefore they are going into the cache. This is an error in my client side code which I will need to look over.

My Reflection

During February, I presented my prototype in my Mid-point presentation. Frances was pleased with how my prototype turned out even though I changed my technology for development. I received a respectable mark of 65% out of 100 so I'll take that!

Intended Changes

- Try get messages to send to multiple rooms instead of going to the cache
- Change some styling elements
- Take Frances' suggestion of a Highscore system and to implement a way for user's to post code to the room

Supervisor Meetings

Frances organised a meeting where she broke down my marks for the mid-point presentation. My presentation went well however it was my documentation that lost me marks however she explained to me that I will be able to change my documentation which is good.

March: