

Software Design Specification Document (CS360)

Stutor



Group Number 8

Hassaan Hassan

Muhammad Zain Qasmi

Abreeza Saleem

Nawal Khurram

Ali Ahsan

Course: Software Engineering CS360

Instructor: Suleman Shahid

University: Lahore University of Management
Sciences (LUMS)

Version: 1.0

Date: (02/04/2017)

Number of hours spent on this document: 50

Contents

Change Log	3
Project Scope	3
Change log	3
Introduction	4
Document Purpose	4
Product Scope	4
Intended Audience and Document Overview	4
Definitions, Acronyms and Abbreviations	5
References and Acknowledgments	6
Overall Description	6
System overview	6
System constraints	7
Architectural strategies	8
System Architecture	8
System Architecture	8
Subsystem Architecture	10
Database Model	11
Database scheme and detailed description	11
Database	13
External Interface Requirements	14
User Interfaces	14
Hardware Interfaces	14
User Interface Design	15
Description of the user interface	15
Information architecture	16
Screens	17
User interface design rules	24
Other Non-functional Requirements	25
Performance Requirements	25
Safety and Security Requirements	26
Software Quality Attributes	26

1 Change Log

1.1 Project Scope

Our project scope is same as that proposed in the SRS document. There have been no major changes or deviations from the initial objectives.

1.2 Change log

1. Algorithm: We have modified the algorithm to be used for searching to find matches for students/tutors. We have introduced new parameters such as use of ratings and made the search more intelligent.
2. Logging in: The Users can also sign up with their Facebook or Google Plus accounts.
3. MongoDB: We have decided to use MongoDB as our database instead of MySQL.
4. We have decided to use Captcha on the login screen.

2 Introduction

2.1 Document Purpose

The purpose of this document is to make the reader familiar with the STUTOR. It will discuss how STUTOR works, and what are the requirements needed to create it. It will list what measures will be taken for STUTOR to have a good performance and the assumptions under which it is to work. Any security issues and how they'll be addressed, will also be discussed. Thus, this document will serve as a guide for the development of STUTOR.

2.2 Product Scope

The aim is to make students and tutors accessible for each other. We intend to create such a platform that will let students find tutors in their nearby area through the use of geo-location (and vice versa) which will save both parties the hectic routine of travelling long. This product will also enforce validation of the authenticity of both parties by the use of a two way rating system which will establish the integrity of our product attracting more users. Students, who wish to become tutors, usually go through the hassle of registering themselves at academies and the difficulties associated with it. Now, they only have to sign up and they're good to go.

Working with this product will be simple and convenient:

1. Sign up for the appropriate profile (student or tutor)
2. Students can find tutors in their area
3. A wide-range of courses will be available
4. Students can choose tutors based on their ratings
5. Instant messaging will be available for both parties
6. Email based account verification.

2.3 Intended Audience and Document Overview

This document is intended to be read by our clients who would be students wanting to give tuition or take it. It is also to be read by our instructor, Suleman Shahid, and the teaching assistants.

Section 3 provides an overall view of how the system will work. It starts off by explaining how the client can use the website and also how the website will work. Then it lists the constraints under which the system will operate. This section concludes by listing the components our system will use. This includes the database to be used, how functionalities (like getting user location) will be implemented, which languages will be used by the developers and where the system will be deployed.

Section 4 explains how the back-end will work. This is mainly to be referenced by the developers. A component diagram is given that'll clearly show the relation between the components and an

activity diagram is also given that show how each component works. Any component that needs further explanation is further described in 4.2. To aid the developer, class diagrams and sequence diagrams are provided. Then, this section provides a detailed view of the database, what each table is made of and how these tables interact with each other. Lastly, the user interface of the system is explained.

Section 5 discusses front end in detail. It provides the tools that'll be used to develop the front end and also gives the screens that'll be used.

Lastly, this document states the performance requirements.

2.4 Definitions, Acronyms and Abbreviations

ADA	American Disabilities Act
Admin	A user who governs/moderates the website
API	A set of routines, protocols and tools for building software applications
Beta Testing	A software testing in which a sampling of the intended audience tries the product out
Buffer Overflow	Anomaly where a program, while writing data to a buffer, overruns the buffer's boundary and overwrites adjacent memory locations
Drop-Down Menu	A graphical control element, similar to a list-box that allows the user to choose one value from a list
DDOS	Distributed Denial of Service Attack.
Geographic Range	The user will input a radius and this will have the effect of pairing him with a student/teacher within a specified radius. If no match is found the user can increase the search area by increasing the radius
GUI	Graphical User Interface

Instant Messaging	In-app web based chatting service
Platform	Software(s) used to host our application
Proximity	Distance from the user's physical location
Quotation	The rate at which the tutor will charge the student
SQL Injection	Code injection technique, used to attack data driven applications in which in which SQL statements are inserted into an entry field for execution
Unit testing	Automated Test for small chunks of code
Usability Testing	Sitting down with people who haven't used the app before, giving them tasks and watching them see where they struggle in order to improve the easy use of the system

2.5 References and Acknowledgments

<http://www.dawn.com/news/686619> [1]

<http://www.techrepublic.com/blog/web-designer/creating-an-ada-compliant-website/> [2]

<https://www.draw.io/> [3]

<https://uxplanet.org/golden-rules-of-user-interface-design-19282aeb06b>[4]

<https://faculty.washington.edu/jtenenbg/courses/360/f04/sessions/schneidermanGoldenRules.html> [5]

<https://developers.google.com/maps/documentation/geolocation/intro> [6]

<https://proto.io/> [7]

3 Overall Description

3.1 System overview

The user will connect with our server using a browser via HTTPS connection. The system involves user belonging to two different categories : Students and Tutors. They will either register an account on Stutor or sign in using an email address, Facebook or Google + account. A signed in student will find nearby teachers using the Google Geolocation API for the course of their interest.

Relevant details and rating of tutors will be provided. Student can add filters such as demanded fees up to Rs 4000 per hour to customize their search. They can then message the tutor and the rest is up to them to set up teaching.

For teachers similarly, students interested in the teacher's course will be displayed. The follow up is the same as the students.

The summarized form is presented below:

Students:

- 1) The application automatically detects the location of the student
- 2) The student is asked in what grade he is currently enrolled and the subjects he is seeking tuitions for.
- 3) The student is then matched up with a tutor based on his location and rating.
- 4) After every two weeks the student is asked to 'rate' his teacher, so we can update our records.

Teachers:

- 1) The application automatically detects the location of the teacher
- 2) The student is asked what subjects he is interested in teaching
- 3) If a student then searches for a teacher and the teacher offers the same subject he is interested in, and lives in close proximity then both parties can connect with each other via a messaging application and decide if it is a suitable match.

The platforms and software to be used are discussed below :

- AngularJS 1.0 would provide the abstraction on the front end and create html pages which is basically what the user would see on the browser.
- MongoDB 3.4 will store the data collected from users i.e students and teachers. The rationale behind this is discussed in section 4.4.2.
- The application will be developed with JavaScript on NodeJS and will be connected with MongoDB.

3.2 System constraints

- As a web-based application, internet is required to access our webpage and so cannot be used offline. Error messages will be displayed whenever there are connection problems.
- Browsers have to support javascript to view our webpages properly.
- Currently our GUI is only in English.
- The users have to login with their credentials to use the functionality of Stutor.
- Locations where geolocation doesn't work (i.e due to bad weather) will not give accurate results. Moreover if someone is using a proxy they would not get the intended results.
- Few old browsers do not support HTML5 fully specifically the canvas tag, so we would advise users to use Chrome 4 or above, Internet Explorer 9 or above, Firefox 2 or above and Safari 3.1 or above.
- We are constrained by the limited budget so we might not be able to buy additional space on heroku once the limit for free usage ends.

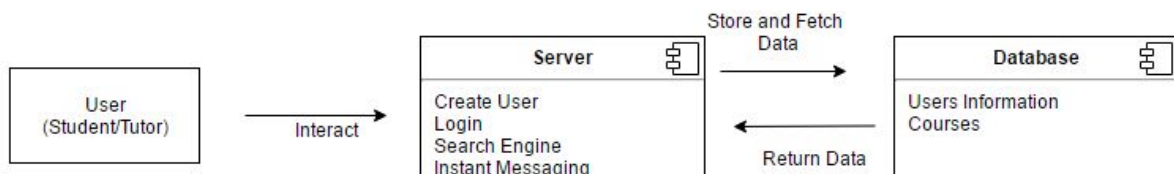
3.3 Architectural strategies

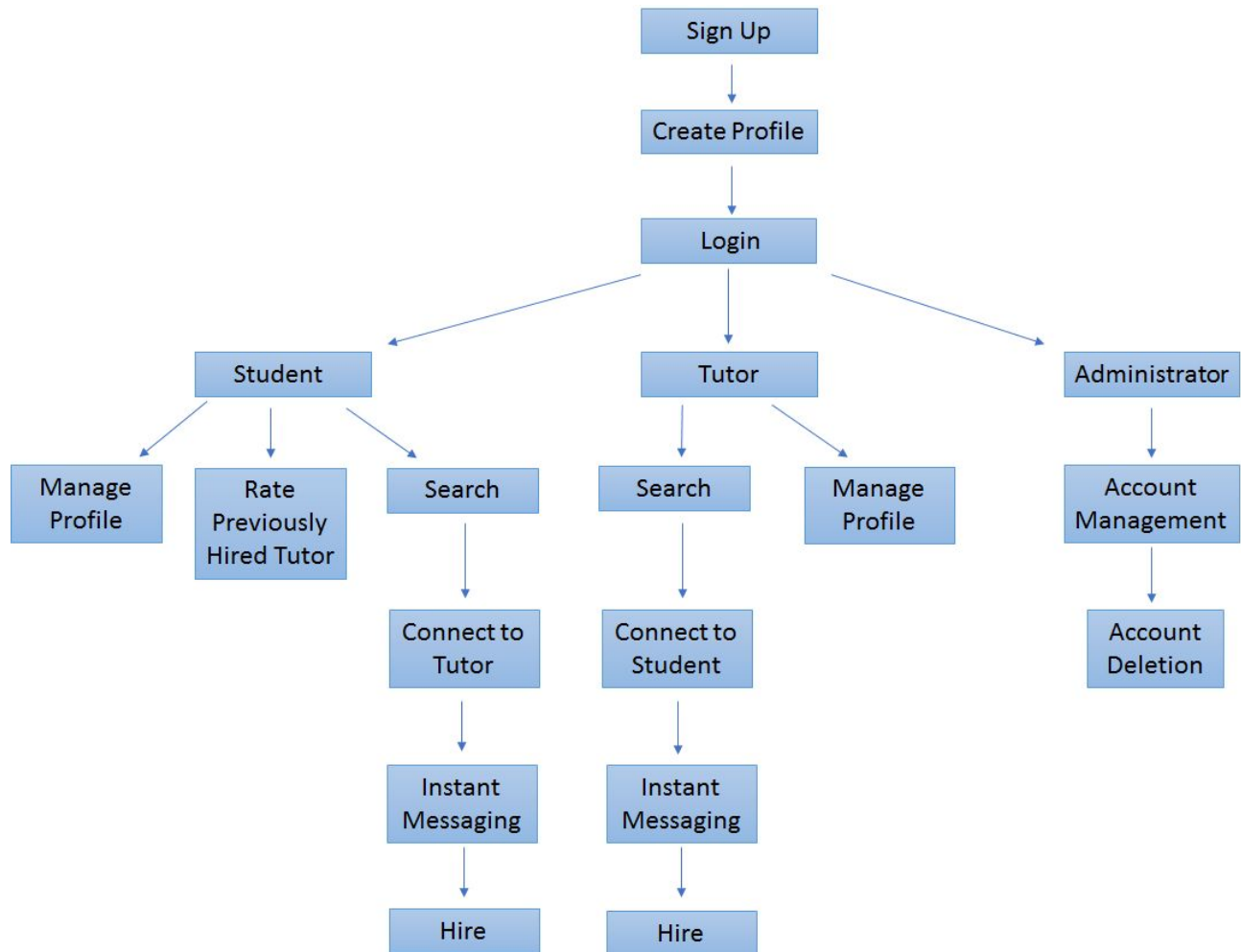
The architecture has been inspired by the following strategies:

- The system will be running a client-server connection. This is a widely used architecture and the developing team is comfortable in its deployment.
- We will be using “The Google Maps Geolocation API” in the frontend to show the map and indicate any interested tutors or student nearby in a given radius.
- The data from user accounts, courses and sessions will be stored on the MongoDB database. The storage and retrieval of data will be easier than doing these on the file system. Also MongoDB is favored over MySQL as its immune to SQL injection attacks.
- Communication with the clients will be done through HTTPS to provide an encrypted channel hence providing valuable security to the data.
- Node.js 7 will be used to set up the server. The developers are experienced in backend development in Node.js.
- The application will be deployed on heroku. We will be pushing our commits to our heroku git repository. Heroku is extremely easy to start and provides simple scaling abstractions.
- AngularJS as our front-end JavaScript framework will allow us to manage dynamic views - a feature that HTML inherently lacks.
- The 3-thir Yeoman workflow will be used to first scaffold out the skeleton of our application, then using either Gulp or Grunt build system we can quickly preview and test our project while we will be using bower package manager to download and manage scripts.

4 System Architecture

4.1 System Architecture



**Server:**

All the components are linked to this. We will be use NodeJS for building our application. The server is used to retrieve and update data in the database. The user interaction with the website sends requests to the server which handles them. It responds by sending the appropriate data back. It also handles the search queries.

Database:

This will store all the student and tutor records. It will also be used to retrieve data based on the search queries. MongoDB will be used for building the database.

Search Engine:

This will be built using the search algorithm we have developed. It will be used for finding matches for students and tutors based on the various parameters and will output the result. This is the main functionality of our project.

Instant Messaging:

This facilities communication between the student and tutor. It is not linked to the server, the server will handle all the messages alone and display them to the users. It is a subcomponent of the

server.

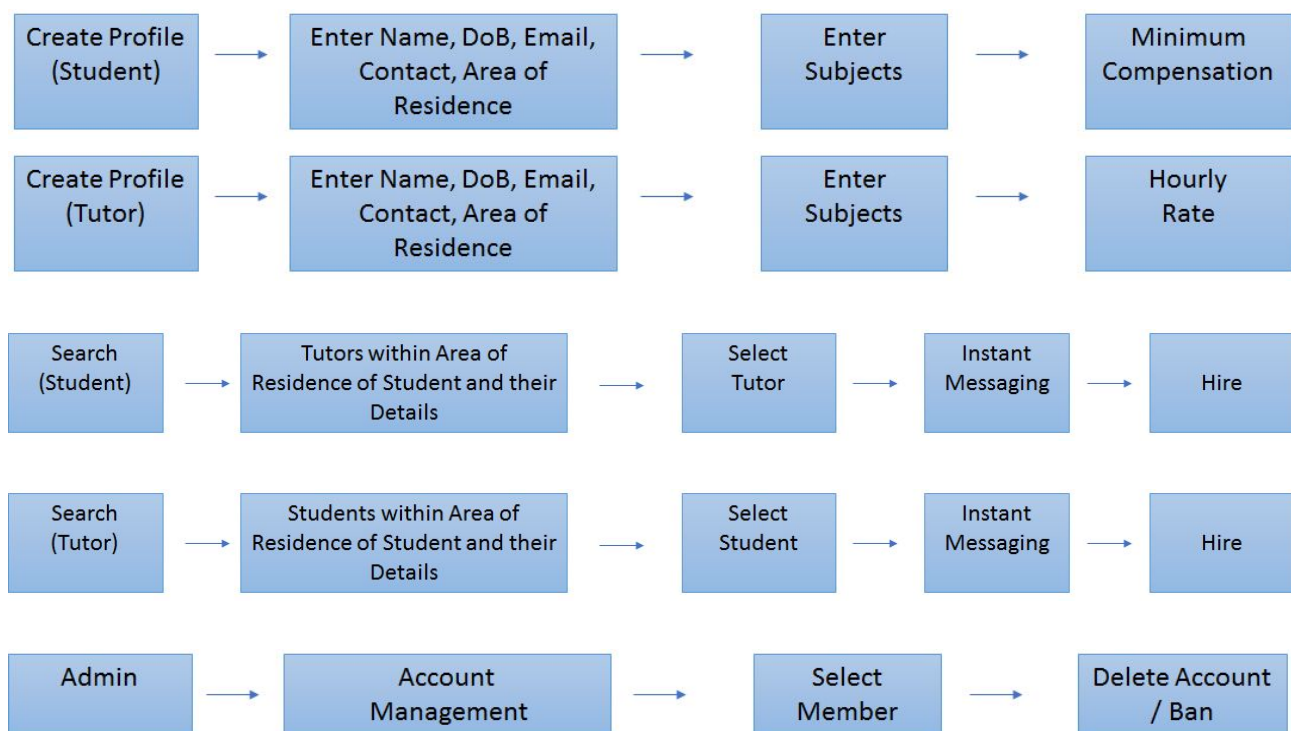
User Interface:

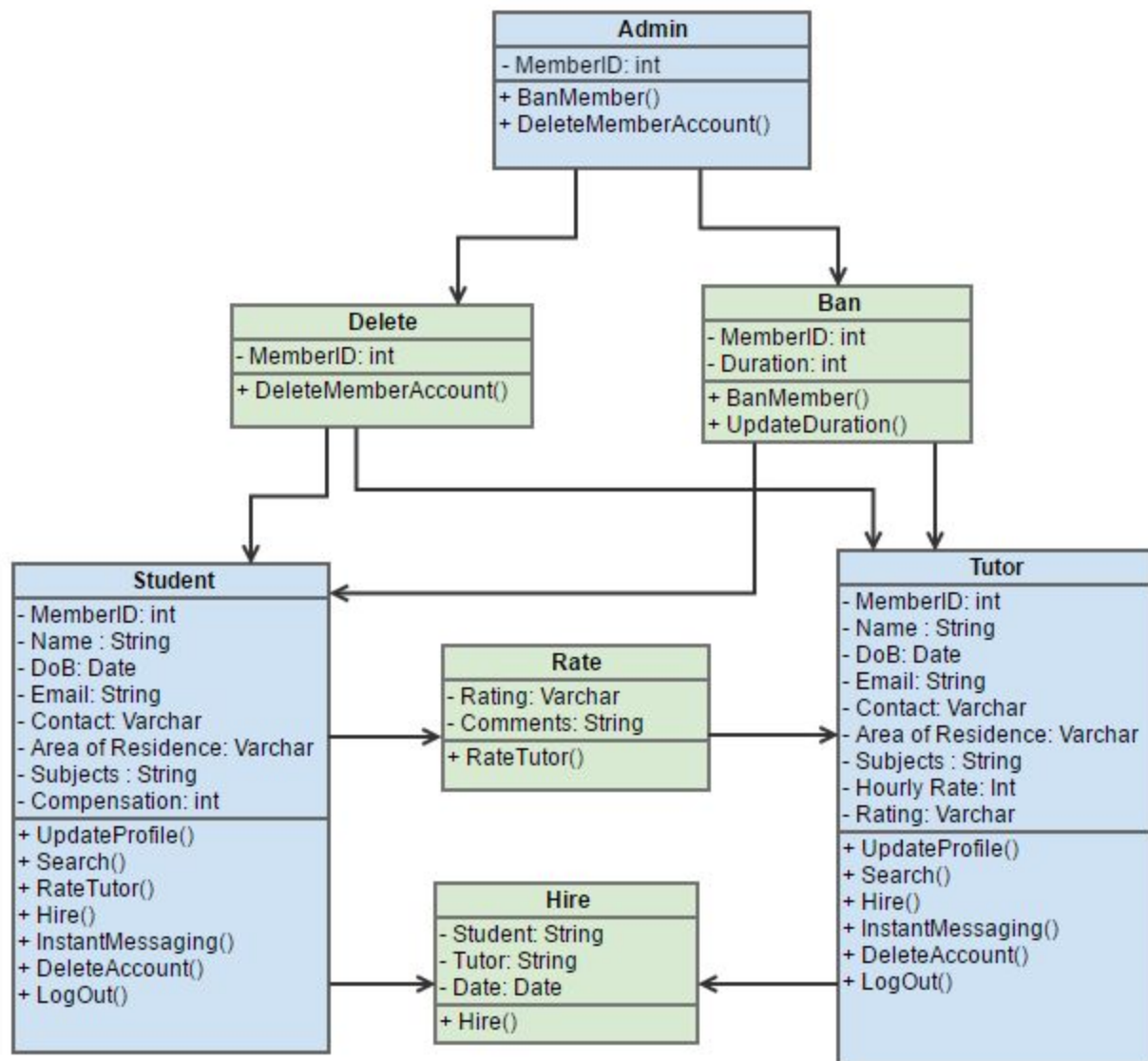
The UI provides interaction between the user and the server. It will be built using HTML and CSS.

Login and Sign Up:

We will design our own sign up system as well as use Facebook's and Google Plus' login APIs to implement this. This will be used to begin access to the application. It is a subcomponent of the server.

4.2 Subsystem Architecture





4.3 Database Model

4.3.1 Database scheme and detailed description

**Member**

Field Name	Type	Length	Constraints
MemberID	String	8	Unique and Not Null
Name	String	40	Unique
Address	String	60	Not Null
Phone Number	String	12	Not Null
Email Address	String	30	Has to be valid
DOB	Date	8	Year has to be greater than 1960
Username	String	10	Unique
Password	String	50	Has to be at least 32 characters

Teachers

Field Name	Type	Length	Constraints
Courses	Array	No limit	At least one element
Salary	Integer	5	Not Null
Ranking	Integer	1	Less than 11 and greater than 0

Students

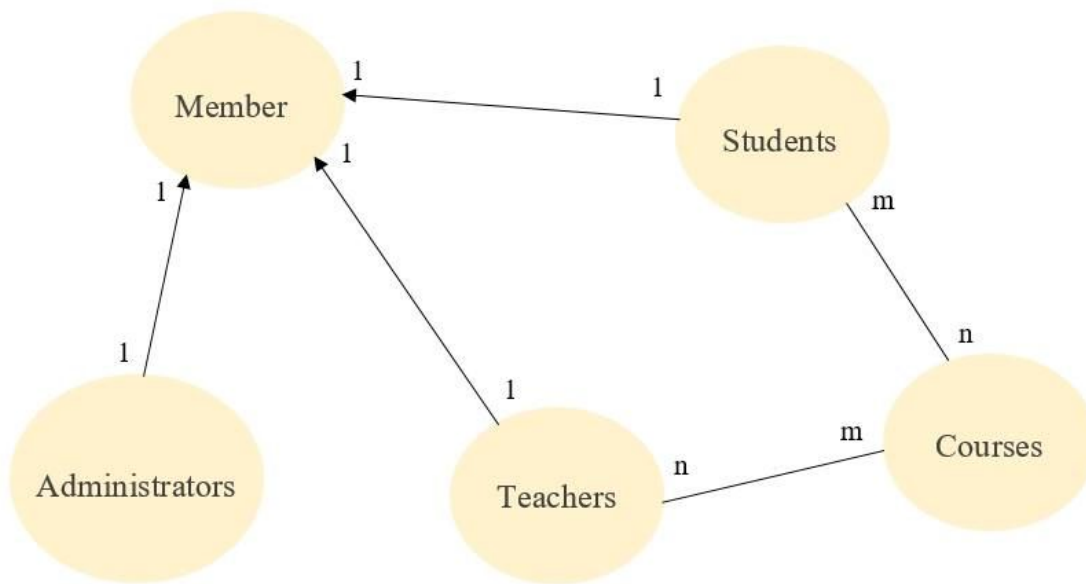
Field Name	Type	Length	Constraints
preferredPay	Integer	5	Greater than 0

Administrators

Field Name	Type	Length	Constraints
Job description	String	120	Not Null
isAdmin	Boolean	1	Not Null

Courses

Field Name	Type	Length	Constraints
CourseID	String	8	Unique
Course Name	String	40	Unique
TaughtBy	Array	No Limit	At least one element
TakenBy	Array	No Limit	



The tables Administrators, Teachers and Students are inherited from the table Members which contain the basic information about a user.

Relations:

- **Member – Administrators:** Each administrator's basic information will be given in the Members table. Because the information will be unique, there will be only one tuple corresponding to an administrator in the Members table. Hence, a one-to-one relation. The same logic can be applied to the relationship between **Member – Students** and **Member – Teachers** tables.
- **Students – Courses:** One student can enroll in multiple courses and one course can have multiple students enrolled. Hence, a many - many relation.
- **Teachers – Courses:** One teachers can teach multiple courses and one course can be taught by various teachers. Hence, a many-many relation.

4.3.2 Database

We have selected MongoDB 3.4 as the database for this project.

Since our data is location based, with MongoDB's built in spacial functions, finding relevant data from specific locations will be fast and accurate. Unlike SQL databases, MongoDB will enable us to build applications faster, handle highly diverse data types, and manage applications more efficiently at scale.

MongoDB's flexible data model also means that the database schema can evolve with our requirement since there is no rigid schema. It also allows flexibility in adding extra field to tables if needed.

4.4 External Interface Requirements

4.4.1 User Interfaces

The app will use a basic GUI with an elegant and simple to use design. The default theme used will be plain and not too colourful or bright. The page will have drop-down menus and will not redirect to a new page for each request. Adhering to the ADA standards [2], the app will not contain repeatedly flashing images. The page will support the option to increase the font size of the content for people with visual disabilities. Users will be requested not to upload offensive profile pictures.

4.4.2 Hardware Interfaces

A user will need a computer and an internet connection to access this web application. Other supported devices will include mobile phones and tablets. The application would be run on a web browser. The website will be hosted on an external server, where the database will also be stored.

5 User Interface Design

5.1 Description of the user interface

Front end tools and technologies:

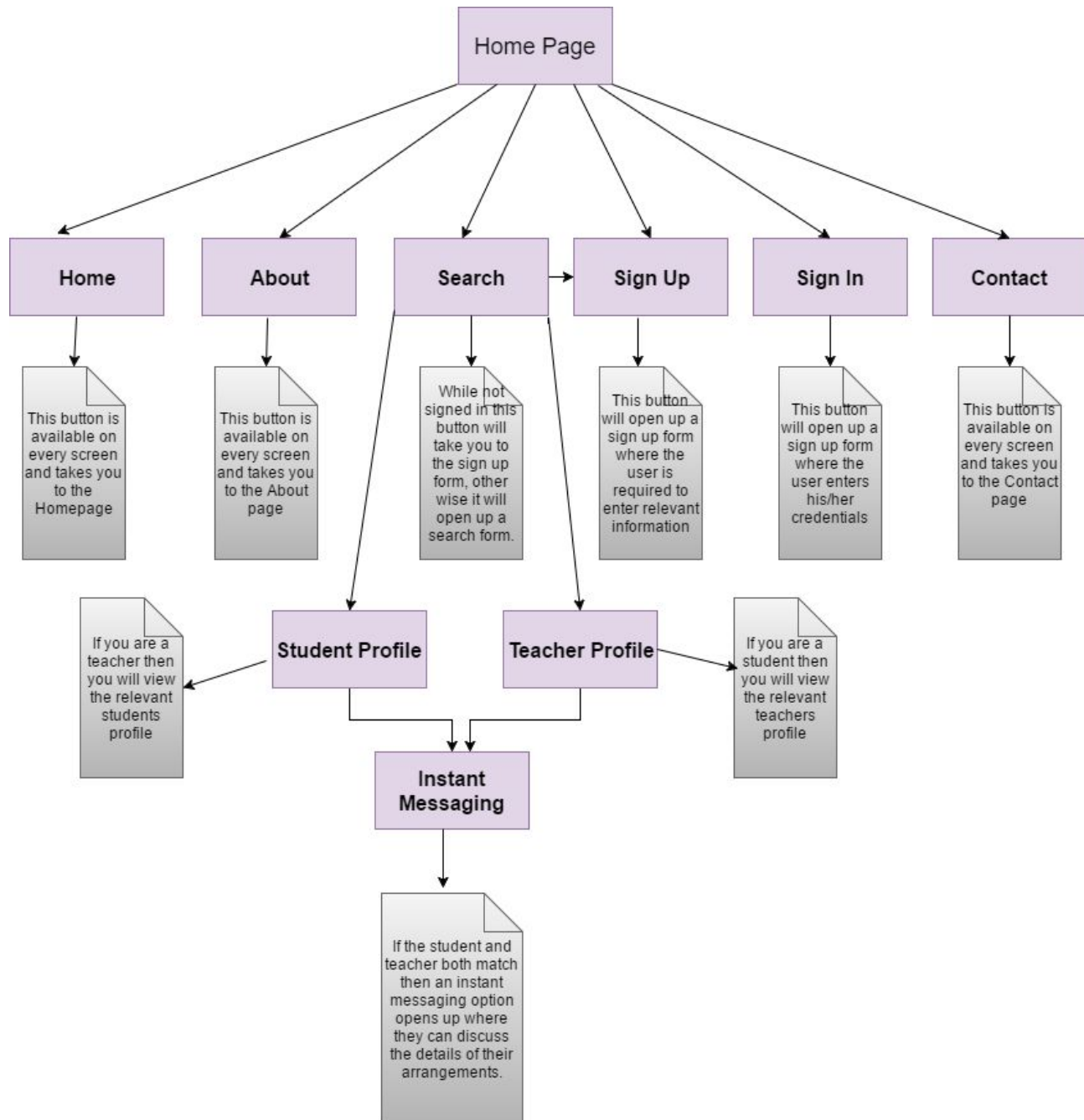
- **Text Editor: Sublime Text.** This will be our text editor of choice simply because it is featureful, stable and customisable.
- **Javascript Framework: AngularJS and Bootstrap.** Since HTML was not designed to manage dynamic views, AngularJS will help to fill in this glaring gap besides being flexible and easy to use while Bootstrap has a lot of relevant tools for creating web applications.
- **Generator: Yeoman.** This allows you to scaffold out a boilerplate for front-end development and quickly setup our project's skeleton with sensible defaults and best practices.
- **Preprocessor: Typescript.** It allows you to continue writing in the dialect of JavaScript while embracing the latest additions of ECMAScript 6.

Each menu will consist of various GUI components, such as buttons, labels, text fields, and list objects. These components will be arranged in such a way that the user will be able to quickly grasp the purpose of each menu and perform whatever task it is designed for - efficiently. A detailed description of these menus and their interactions with each other is described in section 5.2.

First time users will be greeted with the homepage where they have the option to "Sign Up" or "Login". Users who sign up will first have to verify their email address before they can login.

While returning users will have access to most of the features of our platform depending on the type of their user account. They can fill their profile, instant message other users, post new tuition or apply for a course and finally search for specific tutors and students using filters.

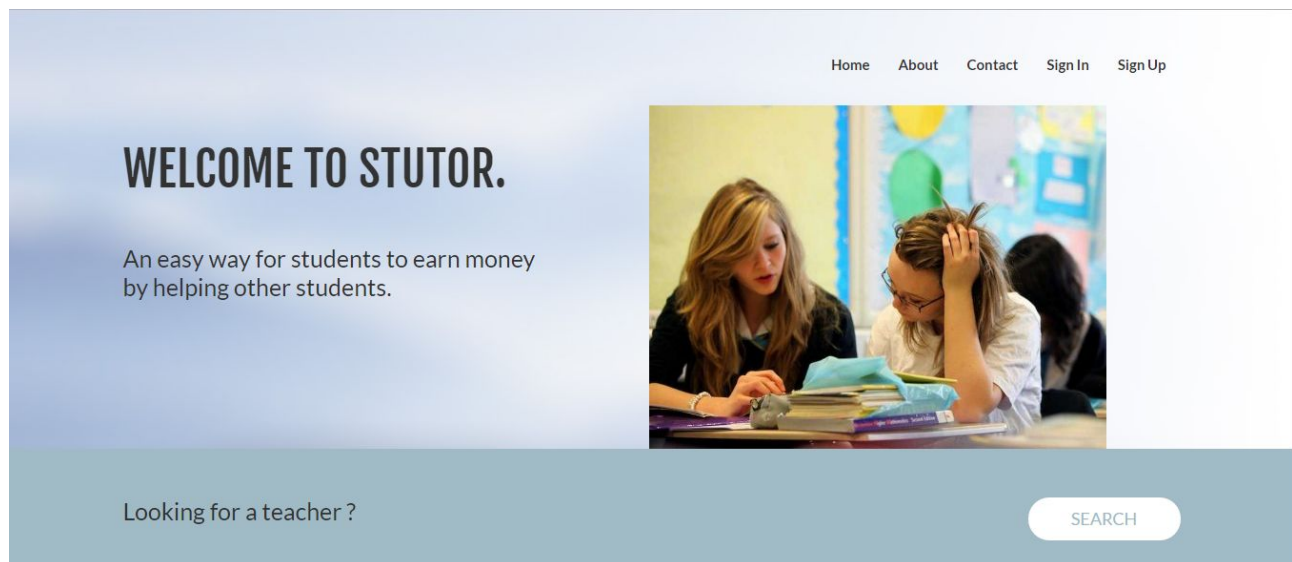
5.2 Information architecture



5.3 Screens

This section provides the graphic user interface for Stutor. Students and Tutors will be able to login the system via Home Page or separate Login and Signup pages from any computer connected to the internet. The following user interfaces will reflect the screens that will be seen when using Stutor's web portal:

5.3.1 Home



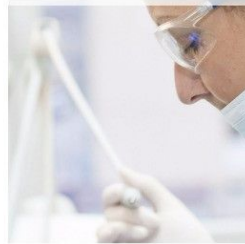
Purpose: This is the home page. It is the first page that the user will see.

User Interface and Navigation: The panel on the top right corner has five options. Clicking 'Home' from any page will bring the user to this page. 'About' will take the user to a page which gives detailed information about how to use our website and an overview about our team. Clicking 'Contact' will display our contact information. 'Sign In' will take the user to a login form. Similarly, 'Sign Up' will take the user to a signup form. If the user is not signed in and clicks 'Search' on the bottom left corner, he will be redirected to the signup form. If the user is signed in and clicks on the 'Search' button then he will be taken to the search form.

5.3.2 About

How it works?

Tutor helps to connect students who are willing earn some extra money by teaching other students. All the money you earn goes in your pocket! And the best part is you don't have to pay us a dime!



Step 1: Sign Up

Sign up. Either as a student or a teacher.



Step 2: Search

You will be matched to a student/teacher whose location and subject match with yours.



Step 3: Chat

A chat box will be opened up, and you can decide amongst yourselves whether you are suitable for each other. Simple!

OUR TEAM



Syed Hassaan Hasan
CEO



Zain Qasmi
CTO



Ali Ahsan
Chairman

Purpose: This page gives information about how to use our website and an overview about our team.

User Interface and Navigation: The top right panel is available on this page as well and will redirect the user to whichever option he/she clicks on.

5.3.3 Contact



Contact details.

LUMS, Opposite
Sector U, DHA,,
Lahore 54792
Phone:042 35608000



Working hours

Sun. 13:00 - 18:00
Mon. 09:00 - 20:00
Tues. 09:00 - 20:00
Wed. 09:00 - 20:00



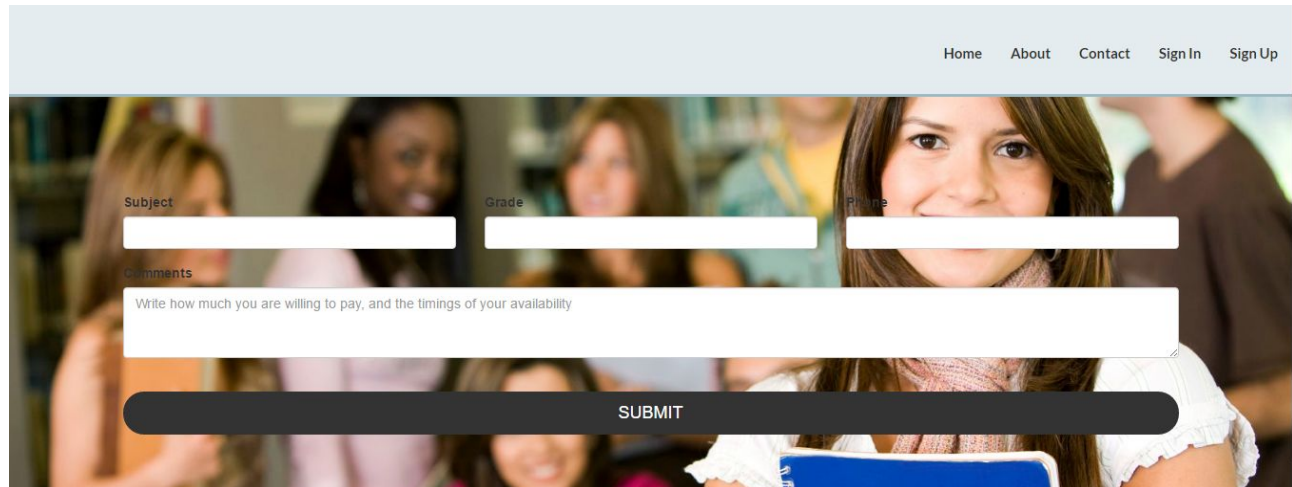
Home About Contact Sign In Sign Up



Purpose: This page displays our contact information and the location of our office via Google Maps.

User Interface and Navigation: The top right panel is available on this page as well and will redirect the user to whichever option he/she clicks on.

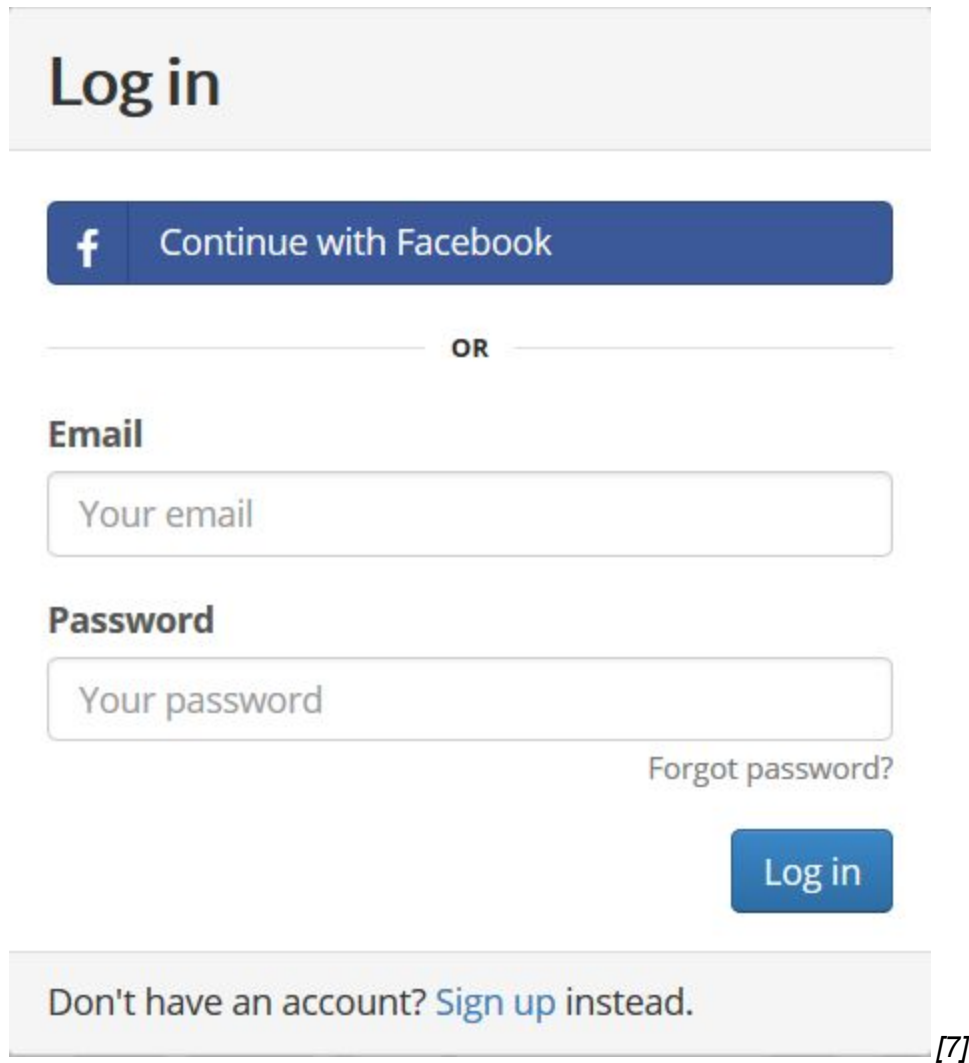
5.3.4 Search

The image shows a web form for searching students and teachers. The form is overlaid on a background image of a diverse group of students. At the top right of the form area, there is a navigation bar with links: Home, About, Contact, Sign In, and Sign Up. The form itself has four input fields: 'Subject', 'Grade', 'Phone', and 'Comments'. The 'Comments' field has a placeholder text: 'Write how much you are willing to pay, and the timings of your availability'. Below the input fields is a large, dark 'SUBMIT' button.

Purpose: This page is used to collect data on students and teachers, so we can match them using our algorithm.

User Interface and Navigation: The user is asked to enter what subject he is interested in teaching/learning, the grade he is currently in or interested in teaching and his/her phone number. The comments section encourages the user to give a brief overview of what are the suitable timings for the user and expected salary (The comments section is not mandatory).

5.3.5 Sign In



The image shows a user login form. At the top, there is a light gray header with the text "Log in". Below this is a blue button with a white Facebook 'f' icon and the text "Continue with Facebook". A horizontal line with the word "OR" in the center separates this from the email/password section. The "Email" section has a label "Email" and a text input field with the placeholder "Your email". Below that is the "Password" section with a label "Password" and a text input field with the placeholder "Your password". To the right of the password field is a link "Forgot password?". Below the password field is a blue "Log in" button. At the bottom, a light gray footer contains the text "Don't have an account? [Sign up](#) instead." followed by a reference marker "[7]" on the right.

Log in

f Continue with Facebook

OR

Email

Your email

Password

Your password

[Forgot password?](#)

Log in


Don't have an account? [Sign up](#) instead. [7]

Purpose: Allowing user of any type to access their account.

User Interface and Navigation: The user login will be the same for all types of users. Access control function will determine the level of access based on the user type.

5.3.6 Sign Up

Sign up

 Continue with Facebook

We don't post anything


OR

Name

Email

Password

☐ I'm not a robot


reCAPTCHA
[Privacy](#) - [Terms](#)

By clicking "Sign Up", you agree to our [terms of service](#).

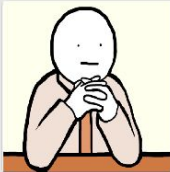
Sign up

Have an account? [Login here](#) instead.

Purpose: Allowing new users to create a profile to become a part of Stutor.

User Interface and Navigation: Sign up will only ask for Name, Email, and Password. And after successful email verification, users can opt for Student or Tutor type and update their profile information such as Birthday, Courses etc.

5.3.7 Student Profile



Ali Ahsan
mzainqasmi@gmail.com

- Inbox
- Quotes
- Tuitions Posted
- Courses**

ACCOUNT

- Settings
- Logout

Courses Beta

View the courses you are enrolled in.


A Levels Maths

By: Pakalu Papito

☆☆☆☆ 0 (no ratings) 1 student enrolled

Covers all the syllabus and curriculum content of GCSE A Levels Mathematics(4024)

5.3.8 Tutor Profile



Muhammad Zain Qasmi
zainqasmi@gmail.com

- Inbox
- Quotes
- Tuitions Posted**
- Courses

ACCOUNT

- Settings
- Logout

Tuitions Posted

[+ Post a Tuition](#)

Pending Staff Approval
Your tuition is pending approval from our staff and will be visible to tutors once it's approved
1 second ago

Job description
asdfk aaalu ka paratha

Subjects: Computer Studies Applied Information and Communication Technology	Grades: Nursery KG	Locality: DHA, Lahore
--	---------------------------------	---------------------------------

Purpose: Allowing users to see the personal information associated with a specific user. Users will be able to decide how much of their biodata could be accessed publicly.


User Interface and Navigation: Student and Tutor profiles will be very similar except there will be higher emphasis on Tutor's rating and qualification while students will be known by

their credit rating based on their payment history.

5.3.9 Rating and Reviews

REVIEWS


★★★★★ 5.0 out of 5 (based on 6 ratings)



5 out of 5

★★★★★

By: Agent Prime




5 out of 5

★★★★★

By: Agent 47

Had a very good experience with you sir. Definitely Recommend.




5 out of 5

★★★★★

By: Agent 48

Spent a very good time with you Sir. God bless you.




5 out of 5

★★★★★

By: Agent 49

Your teaching style is really very interesting Sir.




5 out of 5

★★★★★

By: Agent 50

It was the first time I realized that there's nothing more interesting than computer science. Thanks a lot for you precious time Sir.



5 out of 5

★★★★★


By: Agent 51

You're a legend sir. You don't waste any second of our time. It was a very good experience. Take a lot of care

Leave a review

Rate your experience

☆☆☆☆☆

 Add a title to your review

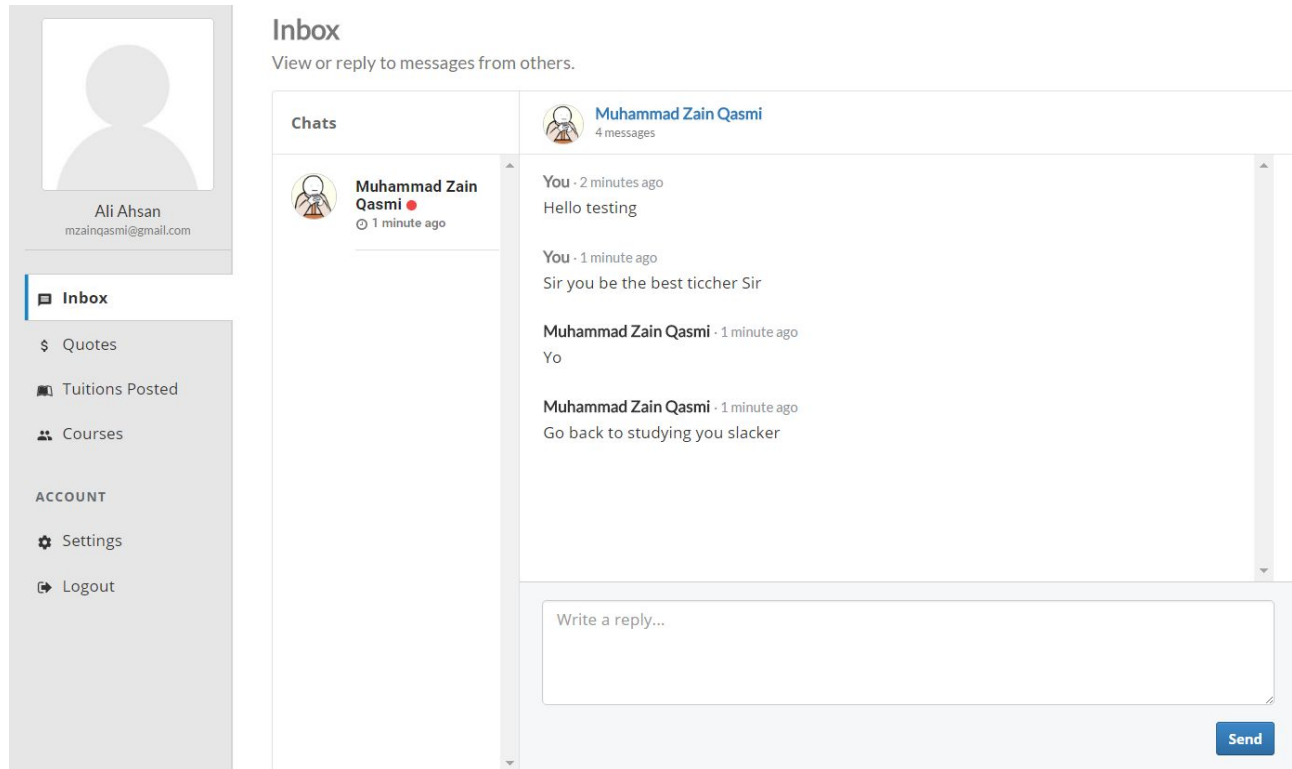
Write a review...

Purpose: To enable critical evaluation of a user.

User Interface and Navigation: Students will be able to rate and comment on their tutors. This will be one of the most critical components as ratings and reviews would enforce quality control in our application since all Tutors will be driven by acquiring a higher rating

so they can market themselves at a higher wage.

5.3.10 Instant Messaging



Purpose: Allowing users to instantly contact each other.

User Interface and Navigation: After a search query is resolved, any student or teacher interested in each other can immediately instant message each other from their profile so they can discuss and negotiate the details of their arrangements.

5.4 User interface design rules

The interface design rules are inspired from Shneiderman's "Eight Golden Rules of Interface Design", [4] [5] they are described below, as well as how they link to our website:

1) Strive for consistency

Consistent sequences of actions should be required in similar situations. Users should not have to wonder whether different words, situations, or actions mean the same thing. For example a certain style of button on our website always does the same thing.

2) Offer informative feedback.

For every operator action, there should be some system feedback. The system should always keep users informed about what is going on. For example when a user searches for a student/teacher, during the time span the algorithm is running in the back end there

should be a progress bar as shown below:



3) Aesthetic and Minimalist design

Keep the interface simple and uncluttered. Remove unnecessary elements and content that does not support the core functionality of our platform.

4) Help and documentation

Even though it is recommended that the interface is simple and intuitive, it is necessary to provide help and documentation. In our website we created an 'About' page that lists the steps you need to search for a student/teacher.

5) Simple error handling

Design the system such that the system detects the error and offers a solution. For example while the user is submitting his phone number on the 'Search' page and by mistake he inputs a letter in place of a digit, in that case there will be an error message displayed which says 'Only digits allowed'.

6 Other Non-functional Requirements

6.1 Performance Requirements

1. **The manual search feature will return the results in ~3 seconds** : Our system involves users searching for other users (tutors / students) and they would like to see the results immediately.
2. **The webpage will be loaded in 2 seconds or less** : Users hate to wait for a page to load so we will optimize the data send(i.e not send too many images at once) so the page loads up in a matter of seconds
3. **Won't contain annoying ads/ popups** : Our users won't go through the annoying experience of opening popups with the cursor clicks. We will provide the best ad experience without causing any trouble to the users
4. **Administrators assistance will be provided** : The contact page will aid users in reporting any hoax for example a student can report a fake tutor or vice versa.

5. **Our system will stay online at all times/ won't crash under any circumstances :** Overwhelming the server with the requests is very common which leads to unavailability of the website, this also happens in DDos Attacks (will be discussed in 4.2). So firewalls will be in place to filter non-critical protocols and block spamming IP addresses.
6. **The website would be optimized for mobiles :** The trend of mobile phone usage to access the web pages has increased tremendously in the past few years. The users often report problems of being forced to scroll from side-to-side, this leaves a bad impression which may force users to leave the website forever. So we'll optimizing for mobile phone usage too.

6.2 Safety and Security Requirements

1. We'll keep few checks for password validation such as minimum of 8 characters, usage of special characters, adding capital letters and few more. Passwords would be hashed by SHA-3 algorithm which currently is computationally infeasible (impossible to brute force) to break. The hashes would be stored at the server side and validation would be done every time user logs in.
2. Multiple accounts won't be allowed to set up against a single email account. Checks would be placed in code to prevent such redundancy.
3. The database is vulnerable to different form of attacks such as DDos and brute forcing all possible combinations, we will incorporate APIs that provide abstraction to secure the system against such attacks. Also we'll be using MongoDB instead of SQL which prevents SQL injection attacks.
4. Privacy of one would be respected. The contact information won't be displayed publicly until intended by the user to do so. Its up to the user to decide whether to share his/her personal contact number with the other person.
5. Manually, users will be able to report to us about malicious activity committed by another person for example a fake user may pretend to be an instructor, if it's rightly pointed the accused user will be removed from the system. Our system will support customer complaint service where the users can also report against any mishap.
6. We will incorporate Captcha to distinguish human from machine input to prevent spamming.

6.3 Software Quality Attributes

6.3.1 Availability

Stutor would remain online at all times. We'll be buying an online domain and hosting our data on external servers. If need be, more servers would be brought in the fold so our product will be available to users 24/7.

6.3.2 Correctness

We aim to provide accurate information to both the parties : students and teachers. The mechanism deployed will update databases as soon as a change is committed and the result will be immediately shown on the web pages. Rest assured the users will be getting the updated and correct information.

6.3.3 Usability

The goal is to provide simplicity and ease of access for users. We will achieve this by allowing the user to navigate and get his job done with minimum number of clicks, keystrokes, and page loads. The web pages' usage is described in section 3.3.1. They would contain the necessary buttons and images with minimal text, contrasting use of colors and consistent overall design so as to provide an easy-to-use experience for users.

6.3.4 Testability

Our product will go through rigorous testing before it goes live. This involves but is not limited to : unit testing, usability testing and automated UI testing. The beta version would be released before the real version so many people can test our product and point out any loopholes in the system.

6.3.5 Portability

We bring portability by optimizing our web product on mobile phones. We'll use tools such as "Google Mobile Optimizer" to reconfigure our webpages for different mobile phones/ tablets. This will surely bring a pleasing experience for the mobile phone customers.

Appendix A - Group Log

Our team worked together for around 50 hours inclusive of group meetings.

We met Humdah several times to discuss our progress.

Appendix B – Contribution Statement

<i>Name</i>	<i>Contributions in this phase</i>	<i>Approx. Number of hours</i>	<i>Remarks</i>
Abreeza Saleem	Worked on Section 4.4	10	
Ali Ahsan	Worked on Section 3	10	
Zain Qasmi	Worked on Section 5	10	
Nawal Khurram	Worked on Section 4.1,4.2,4.5	10	
Hassaan Hassan	Worked on Section 5	10	