



INSTITUTE OF ENGINEERING & MANAGEMENT

Department of Computer Science & Engineering

Design Lab

Code: CS891

System Requirements Specification For tutorsandstudents.in

Version 1.0

Prepared by Team Cerebro6.0

10/03/2016

Revision History

Name	Date	Reason For Changes	Version
Team Cerebro6.0	10/03/2016	Changes based on review	0.9

Validation

Approver Name	Title	Signature	Date

Contents

1.	Product Description.....	1
1.1	Purpose.....	1
1.2	Scope.....	1
1.3	Stakeholders and Users	1
1.4	Assumptions.....	2
1.5	Constraints.....	2
2.	Functional Requirements.....	3
2.1	Registration	3
2.2	Manage Profile	6
2.3	Manage Tutors	7
2.4	Negotiate.....	8
2.5	Search Tutors	9
2.6	Rate and review Tutors	12
2.7	Manage student records.....	13
3.	Interface Requirements.....	15
3.1	User Interfaces	15
4.	Use Case Model.....	22
4.1	Use Case Diagram.....	22
4.2	Use Case Description	22
5.	Glossary.....	33
6.	References.....	35

1. Product Description

1.1 Purpose

This document describes the requirements and specifications for a web based portal, which aims to provide an online meeting place for private tutors, students and offline agencies which try to connect tutors to students. Students, especially those studying in primary/secondary school find the process of looking for private tutors quite cumbersome. The situation is no different for students looking for tutors in non-academic subjects like arts, drawing, music, sports, dance, languages etc. The relatively lesser known private tutors often fail to get enough students due to their lack of visibility to students. In most cases, students hear of tutors by word of mouth. Also agencies which cater to the problem of connecting tutors and students' offline frequently complain about not getting their due commissions despite referring the students to the tutors, due to both party bypassing the agent and negotiating terms mutually. This portal aims to provide an online meeting place in order to simplify the aforesaid problems.

1.2 Scope

The web portal will have three classes of users – Tutors, Students and Agencies. Independent tutors will be able to build an online profile, filling in necessary details which they would normally use to advertise themselves offline. The System will allow them a fair chance of being visible based only on their filled in information and not any advertising skills.

Students can search for tutors, filling in all the necessary criteria they desire. The System will provide them an optimally ranked set of search results, which they would find very difficult to find offline, purely by word of mouth. They can also read reviews and ratings of various tutors, and check whether these tutors are verified by external agents.

Agents which connect tutors and students offline can use this system to ease their management process, and get larger visibility due to the system being online. Also the System allows them to control the negotiation between any tutor and student that they connect, so that they can have their commission from the tutor.

The System will provide all these functionalities to the users free of cost, with no registration, usage or any other hidden costs involved (Agents charging commission from tutors is beyond the scope of this system).

1.3 Stakeholders and Users

The following Stakeholders have been identified:

- 1) Private Tutors teaching any discipline/ subject (Tentative User)
- 2) Students looking for private tutors (Tentative User)
- 3) Agencies which connect tutors and students offline (Tentative Users)
- 4) Team Cerebro6.0 (Developers and Owners)

1.4 Assumptions

1. Tutors, Students and Agents registered in the portal are genuine in nature.
2. The agent is able to keep a tab on the negotiation process till final assignment of tutor to student, without divulging contacts of either party to the other.
3. The various parameters used while searching are sufficient to find optimal matches as actually required by the student.
4. The student should be honest about his rating and views about his/her private tutor.
5. The achievements of students being entered by the independent tutor/ agent are authentic in nature. Since the achievements can be of varied nature so it is extremely difficult to authenticate the same. So the System assumes that agents/independent tutors do not add incorrect achievements to their profiles.

1.5 Constraints

1. A student's search for tutors in a particular subject is constrained to the availability of registered tutors of that subject in the portal.

2. Functional Requirements

2.1 Registration

Description: Actors can register themselves with the system. This registration provides the actor with their own user account, by virtue of which they can navigate/use the different functionalities of the system. The Registration use case can be divided into three sub categories: Agent, Independent Tutor, and Student Registration.

2.1.1. The registration choice page should have three sections – Independent Tutor, Student, and Agent. Each section describes how the respective actor is benefitted by the System, and a “Register” Button. The button takes the actor to the registration page which has the registration form. Fields of registration will depend on the type of registration option selected

2.1.2. The registration form options will be as follows:

2.1.2.1 For Independent Tutors Section form fields will be

2.1.2.1.1 Name: This field will be a simple text box for name.

2.1.2.1.2 DOB: This field will be a date selection widget for selection of date of birth. System must ensure that the date of birth cannot be after the current System date.

2.1.2.1.3 Gender: This field will be set of radio option buttons with the options – Male and Female.

2.1.2.1.4 Email Address: This field will be a simple text box for email. System must ensure that the email entered is of valid format. Also Email is a unique field in the database, so one email can have only one account.

2.1.2.1.5 Contact Number: This field will be a simple text box for Contact Number. System must ensure that only valid Number formats (10-digit without special characters) is entered. This is not a mandatory field.

2.1.2.1.6 Physical address: This field will be a simple text area for address. This is not a mandatory field.

2.1.2.1.7 Discipline: Discipline which the independent tutor teaches. This is to be implemented as a multi-select drop down menu specifying the following options - Academics/Technical or vocational / Art/ Singing/ Instrument/ Dance/ Sports/ Others

2.1.2.1.8 DisciplineSub-category: The user can select the discipline sub categories. For each selection made in 2.1.2.1.7, there will be a drop down menu. The System will dynamically load the list of options depending on the selection made in 2.1.2.1.7. Selection will be disabled if user selected ‘Others’ in 2.1.2.1.7

2.1.2.1.9 Experience: The user mentions years of experience. This is to be implemented as a numeric spinner starting from 0 to 40, with an interval of 5. Default value will be 0 – signifying No experience.

2.1.2.1.10 Remuneration: The user can select the range of remuneration expected. This is to be implemented as two numeric spinners, one for start range – 2.1.2.1.10.A and other for end range - 2.1.2.1.10.B. The spinner values are to start from 0 to 10000, with an interval of 100. The System must ensure that the value selected at 2.1.2.1.10.A is always smaller than the one in 2.1.2.1.10.B.

2.1.2.1.11 Day Slots: The user can select preferred day slots. This is to be implemented as a radio button set with two options – ‘Any’, and ‘Specific Days’. In case ‘Specific Days’ is selected the system will display a multi-select list-box, with the 7 days of the week as options. System shall ensure that if user selects all days in list box, the entry made is ‘Any’ and not ‘Specific Days’ with 7 selected days.

2.1.2.1.12 Time Slots: The user can enter preferred time slots. This is to be implemented as a set of radio buttons with two options – ‘Any’, and ‘Specific Time’. In case ‘Specific Time’ is selected, System will display a spinner – 2.1.2.1.12.A to specify the number of time slots to be entered. The spinner will have values starting from 1 and end at 5, with an interval of 1. System will then generate twice the number of spinners, as mentioned in 2.1.2.1.12.A, to specify time ranges. We define these spinners as belonging to two sets – 2.1.2.1.12.Set1 for start time ranges, and 2.1.2.1.12.Set2 for end time ranges. All spinners in the above mentioned sets will have values from 00:00 to 23:30, with an interval of 00:30. The System will ensure that for each pair of ranges, the time in 2.1.2.1.12.Set1 is less than the corresponding 2.1.2.1.12.Set2 time.

2.1.2.1.13 Location: The user enters desired location preference. This is to be implemented as a radio button set with two options – ‘My Location’, and ‘Any Location’. The System will also provide a text box to mention specific area (System assumes that the city is Kolkata)

2.1.2.1.14 Batch Strength: The user specifies what batch strength they teach. This is to be implemented as a multi-select dropdown menu with the following options – ‘Single’, ‘Up to 5’, ‘Up to 10’, ‘More than 10’.

2.1.2.1.15 Description: This is a simple text area to enter any description about self.

2.1.2.1.16 Profile Picture Upload: This allows user to upload a profile picture. To be implemented using a file upload plug-in. This is not mandatory.

2.1.2.2 For Students Section form fields will be

2.1.2.2.1 Name: This field will be a simple text box for name.

2.1.2.2.2 Email Address: This field will be a simple text box for email. System must ensure that the email entered is of valid format. Also Email is a unique field in the database, so one email can have only one account.

2.1.2.2.3 Contact Number: This field will be a simple text box for Contact Number. System must ensure that only valid Number formats (10-digit without special characters) is entered. This is not a mandatory field.

2.1.2.2.4 Physical address: This field will be a simple text area for address. This is not a mandatory field.

2.1.2.2.5 Profile Picture Upload: This allows user to upload a profile picture. To be implemented using a file upload plug-in. This is not mandatory.

2.1.2.3 For Agents Section form fields will be

2.1.2.3.1 Name of Agency: This field will be a simple text box for name.

2.1.2.3.2 Name of Contact Person: This field will be a simple text box for name.

2.1.2.3.3 Email Address: This field will be a simple text box for email. System must ensure that the email entered is of valid format. Also Email is a unique field in the database, so one email can have only one account.

2.1.2.3.4 Contact Number: This field will be a simple text box for Contact Number. System must ensure that only valid Number formats (10-digit without special characters) is entered.

- 2.1.2.3.5 Physical address: This field will be a simple text area for address. This is not a mandatory field.
- 2.1.2.3.6 Profile Picture Upload: This allows user to upload a profile picture. To be implemented using a file upload plug-in. This is not mandatory.
- 2.1.3 Each form page will have a “Submit” Button. This button will submit all the data to System server. Server will validate all the data as mentioned above, and return a positive or negative acknowledgement.
- 2.1.4 If acknowledgement is positive, the server starts a new session and actor is redirected to the dashboard page.
- 2.1.5 If acknowledgement is negative, the actor is redirected to the landing page with an error message.

2.2 Manage Profile

Description: An actor can login to his profile from the landing page. An actor makes changes or adds new information to his/her profile. The respective user can update his/her existing profile. An actor can also view any other profile on the system.

- 2.2.1 Authentication for log-in
 - Input: email, password, “Log-in” option.
 - Output: dashboard page opened on success, landing page with error message on failure.
 - Processing: System matches entered email and password with database records. If there is a successful match, a new session is started and the actor is redirected to their dashboard. If no match is found, error message is shown, and actor is redirected to the landing page.
- 2.2.2 Update existing profile
 - 2.2.2.1 The dashboard menu will have an “Update Profile” option
 - Input: “Update Profile” option.
 - Output: Page with fields to update. The fields will be the same as used during registration (mentioned in 2.1.2), except Name and Email.
 - 2.2.2.2 Fill and save changes
 - Input: Filled up changes, “Submit” option.
 - Output: Updated profile.
 - Processing: System updates database with new values. On successfully updating database on all changes, system redirects actor to dashboard. If

unsuccessful, error message is shown, and actor is redirected to update profile page.

2.2.3 View a profile

Input: profile link.

Output: desired public profile page.

Processing: system searches database records for profile name which is clicked. If a match is found, system redirects actor to the public profile. If no match is found, suitable error message is shown.

2.3 Manage Tutors

Description: Agents register tutors with the System. This creates a separate type of tutor profile – one managed by the agent, with an agent authenticated tag accompanying the profile.

2.3.1 Registration – The agent should be able to register tutors. This kind of registration will create a new profile for such a tutor, which will have an “Agent Verified” label. This account can be managed only by the Agent.

2.3.1.1 The agent’s dashboard menu should have a “Register Tutor” Button. This button should open up a page showing a registration form similar to the one for independent tutor registration in 2.1.2.1.

2.3.1.2 The agent can fill up all the fields. Name of tutor, Contact Number, email address, physical address, are not mandatory fields.

2.3.1.3 There should be a “Submit” Button. This button will submit all the data to System server. Server will validate all the data as mentioned above, and return a positive or negative acknowledgement.

2.3.1.4 If acknowledgement is positive, the system should display that the account was successfully created and redirect the agent to the dashboard.

2.3.1.5 If acknowledgement is negative; the agent is redirected to the dashboard with an error message.

2.3.2 Managing Tutors profile – The agent should be able to manage all their registered tutors’ profiles.

2.3.2.1 The agent’s dashboard menu should have a “List my Tutors” button. This button should redirect agent to a page containing a list of all their registered tutors.

2.3.2.2 Each tutor listing should have “Remove” Button, and an “Edit Profile” Button.

2.3.2.2.1 The remove button should remove the tutor from the agent's list of registered tutors.

2.3.2.2.2 The edit profile button should take the agent to a page similar to the edit profile page in 2.2, where the agent can edit all the fields entered during registration. There should also be a "Save Changes" button to save all changes to database and redirect agent to dashboard.

2.4 Negotiate

Description: The student can request to negotiate the parameters of tuition, online, either directly with an independent private tutor, or with any agent registered tutor, whom the student has found from search results. In case the tutor in consideration is registered under an agent, the negotiations are made through the agent, with the agent being the single point of contact.

2.4.1 The public profile page of all tutors should have a "Negotiate" button. If the tutor is an independent private tutor, negotiation will happen with the independent tutor. If the tutor profile is registered by an agent, then the negotiation will happen with the agent.

2.4.2 Whenever a student clicks on the negotiate button, System should start a new negotiation thread between the controller of that profile (agent/independent tutor) and the student.

2.4.3 The negotiation option should redirect student to a message page (refer to User Interfaces section). There should be a pane showing all open negotiation. The student can select any one to view the exchanged messages.

2.4.4 The page will show a list of negotiable parameters. These parameters are editable for the independent tutor / agent, but not for the student. The values being set will be derived from the tutor's profile by default. The independent tutor / agent can change the parameters based on requests made by the student

2.4.5 There should be message box, where messages can be entered by the students to request change of terms.

2.4.6 There should be a "Close Deal" button for each thread. Clicking of this button by one party should prompt the other user with two new buttons – Acknowledge and Reject. In this state, negotiable parameters are un-editable on either side. The other party may choose to either acknowledge in which case System will mark the negotiation successful, and register the Student in the list of

registered students for the tutor. In case Reject is selected, both actors are brought back to the negotiation stage.

2.4.7 There should be a “Cancel Negotiation” button. Clicking this button should close the thread and negotiation should be considered unsuccessful.

2.4.8 The dashboard of all actors should have a See Negotiations option which links to the page mentioned above.

2.5 Search Tutors

Description: System shall allow students to search for tutors by providing search criteria based on various parameters. System then searches its database to find tutors who best match the given criteria, and tries to return results in an optimally ranked manner.

2.5.1 System shall provide a form to the user to enter search criteria. The criteria are as follows

2.5.1.1 Discipline: The user can select the discipline for which tutors are required. This is to be implemented as a drop down menu specifying the following options - Academics/Technical or vocational / Art/ Singing/ Instrument/ Dance/ Sports/ Others.
In case User selects ‘Others’, the system shall provide a prompt to specify what they want.

2.5.1.2 Discipline Sub-category: The user can select the discipline sub category for which tutors are required. This is to be implemented as a drop down menu. The System will dynamically load the list of options depending on the selection made in 25.1.1. Selection will be disabled if user selected ‘Others’ in 25.1.1.

2.5.1.3 Remuneration: The user can select the range of remuneration to be considered. This is to be implemented as two numeric spinners, one for start range – 25.1.3.A and other for end range - 25.1.3.B. The spinner values are to start from 0 to 10000, with an interval of 100. The System must ensure that the value selected at 25.1.3.A is always smaller than the one in 25.1.3.B.

2.5.1.4 Day Slot Preference: The user can select preferred day slots. This is to be implemented as a radio button set with two options – ‘Any’, and ‘Specific Days’. In case ‘Specific Days’ is selected System will display a multi-select list-box, with the 7 days of the week as options. System shall ensure that if user selects all days in list box, the search criteria sent to search engine is ‘Any’ and not ‘Specific Days’ with 7 selected days.

2.5.1.5 Time Slot Preference: The user can enter a preferred time slot. This is to be implemented as a set of radio buttons with two options – ‘Any’, and ‘Specific Time’. In case ‘Specific Time’ is selected, System will display a spinner – 25.1.5.A to specify the number of time slots to be entered. The spinner will have values starting from 1 and end at 5, with an interval of 1. System will then generate twice the number of spinners, as mentioned in 25.1.5.A, to specify time ranges. We define these spinners as belonging to two sets – 25.1.5.Set1 for start time ranges, and 25.1.5.Set2 for end time ranges. All spinners in the above mentioned sets will have values from 00:00 to 23:30, with an interval of 00:30. The System will ensure that for each pair of ranges, the time in 25.1.5.SetA is less than the corresponding 25.1.5.SetB time.

2.5.1.6 Location Preference: The user enters desired location preference. This is to be implemented as a radio button set with two options – ‘Home’, and ‘Any Location’. The System will also provide a text box to mention specific area (System assumes that the city is Kolkata).

2.5.1.7 Experience Preference: The user mentions experience required. This is to be implemented as a numeric spinner starting from 0 to 40, with an interval of 5. Default value will be 0 – signifying No experience.

2.5.1.8 Age Preference: The user selects a range of age required. To be implemented as a dropdown menu specifying the following age ranges – ‘18-25’, ‘25-30’, ‘30 - 40’, ‘40-50’, ‘Above 50’.

2.5.1.9 Gender Preference: The user specifies gender preference. This is to be implemented as a set of radio buttons with three options – ‘Male’, ‘Female’, ‘Any’.

2.5.1.10 Batch Preference: The user specifies what batch strength they are comfortable with. This is to be implemented as a dropdown menu with the following options – ‘Single’, ‘Up to 5’, ‘Up to 10’, ‘Any’.

2.5.1.11 External agent verification: The user specifies whether they need the tutor to be verified by external agents. This is to be implemented as set of radio buttons with two options – ‘Mandatory’, and ‘Not Mandatory’.

2.5.2 The System will generate a graphical box with all the filled value fields listed in the above mentioned order. The user can slide the fields across one another to decide an order of priority. After finalizing the priority the hit a ‘Finalize Priority’ Button to set priority.

2.5.3 The System shall provide a 'Find' Button to initiate the search process. Hitting the button should send the form data and priority to the search engine end point, with instruction to initiate a search process.

2.5.4 The System shall provide a search engine which return results meeting the following criteria.

2.5.4.1 The search algorithm will match all or most of the specified criteria, with priority assigned according to the list mentioned in 25.2.

2.5.4.2 Based on the final priority list, the search engine will provide higher ranks to tutors which match more number of criteria, the score weights being skewed according to the priority list (higher priority has higher score multiplier).

2.5.4.3 The algorithm will also assign higher score to tutors with higher certified rating values, number of positive certified reviews, and more number of student achievements.

2.5.4.4 The scores calculated from 2.5.4.2 and 2.5.4.3 is to be summed/aggregated to generate one final score for each profile.

2.5.4.5 Final results will be ranked by score, with higher scoring tutor profiles appearing higher in the list.

2.5.5 The System shall return results to the user in a well formatted results page, as a list of tutors in preordered rank, as determined by algorithm in 25.4. The page will have results, with each result having a 'Visit Profile' Button to allow users to visit the profile of the concerned tutor. There will also be a 'Contact' button which links to the Negotiation system. If the Tutor result is that of one registered by an Agent, the button will open negotiations with the Agent. Otherwise negotiation will be initiated with the concerned independent tutor. Refer to UC.7/27 for details on Negotiation functionality.

2.5.6 The result page shall have a 'Refresh Search' Button which will redirect the user to the search criteria form page with a new form, and the user can start the search process afresh.

2.6 Rate and review Tutors

Description: The system will allow students to rate a tutor, on a scale of maximum 5 stars, based on the tutor's performance and the student's level of satisfaction. The rating system is proposed as follows:

- i) 1 star- Very Poor
- ii) 2 stars- Poor
- iii) 3 stars- Satisfactory
- iv) 4 stars- Good
- v) 5 stars- Very Good

The student may also write a detailed review of the tutor if he/she wishes to.

2.6.1 The flow of events of the Rating Section will be as follows:

2.6.1.1 Student selects a Tutor to rate.

2.6.1.2 Student places a rating value and submits.

2.6.1.3 System checks whether the student is a registered current/ ex-student of the concerned tutor.

2.6.1.4 If the student is a registered current/ ex-student, the rating is added to the verified rating records of the tutor. Otherwise it is added to the unverified ratings records.

2.6.1.5 System acknowledges the rating and shows this rating in all subsequent calls for the tutor's profile.

2.6.2 The flow of events of the Review Section will be as follows:

2.6.2.1 Student selects a Tutor to review.

2.6.2.2 Student writes a review.

2.6.2.3 Student selects a 'Positive' or 'Negative' indicator.

2.6.2.4 Student hits 'Submit'.

2.6.2.5 System checks whether the student is a registered current/ ex-student of the concerned tutor.

2.6.2.6 If the student is a registered current/ ex-student, the review is marked and added to the verified review records of the tutor. Otherwise it is added to the unverified review records.

2.6.2.7 System acknowledges the review and shows this review in all subsequent calls for the tutor's profile.

2.6.3 The System will provide a scope for verification of the student as under:

2.6.3.1 The System will have a record of all students that a particular tutor has marked as a student.

2.6.3.2 If a particular student is either a current or ex-student of the concerned tutor and if the student rates/reviews that tutor, a "Verified Student" badge will be shown beside the name of that student in the rating/review section.

2.6.3.3 On the other hand, if a student rates/reviews a tutor to who is not recorded as a student of the concerned tutor, the "Verified Student" badge will not be added.

2.6.3.4 This Verified vs. Unverified ratings and reviews will affect the search ranking of the tutor.

2.6.4 The underlying assumptions are:

2.6.4.1 A student should be willing to help other students looking for tuitions in the same field by providing a rating and a review of the tutor from whom he/she has taken tuitions.

2.6.4.2 The student should be honest about his rating and views about his/her private tutor.

2.7 Manage student records

Description: System will allow agents/independent tutors to add the student achievements to the respective tutor's profile. Any form of achievement/record will be updated in the tutor's profile for determining his/her rank. This use case will play a significant role for the search results obtained for any tutor.

2.7.1 The independent tutor dashboard menu will have a "Manage Student Records" option. The Agents can access this option for their registered tutors by going to a specific registered tutor's update profile section as mentioned in 2.3.2.

2.7.1.1 The System should provide a list of Students registered by successful negotiations made by the subsystem 2.4

2.7.1.2 Each listing should come with a remove button. This button should remove the student from the list of registered students for that tutor.

2.7.2 The independent tutor dashboard menu will have an “Achievements” option. The Agents can access this option for their registered tutors by going to a specific registered tutor’s update profile section as mentioned in 2.3.2.

2.7.2.1 The System displays a list of previously added achievements.

2.7.2.2 There should be an “Add Achievement” button. This button should provide the actor with a text area to enter the description of achievement.

2.7.2.3 Along with the above mentioned text area there should also be Button “Add” which will add the achievement description to the list of achievements.

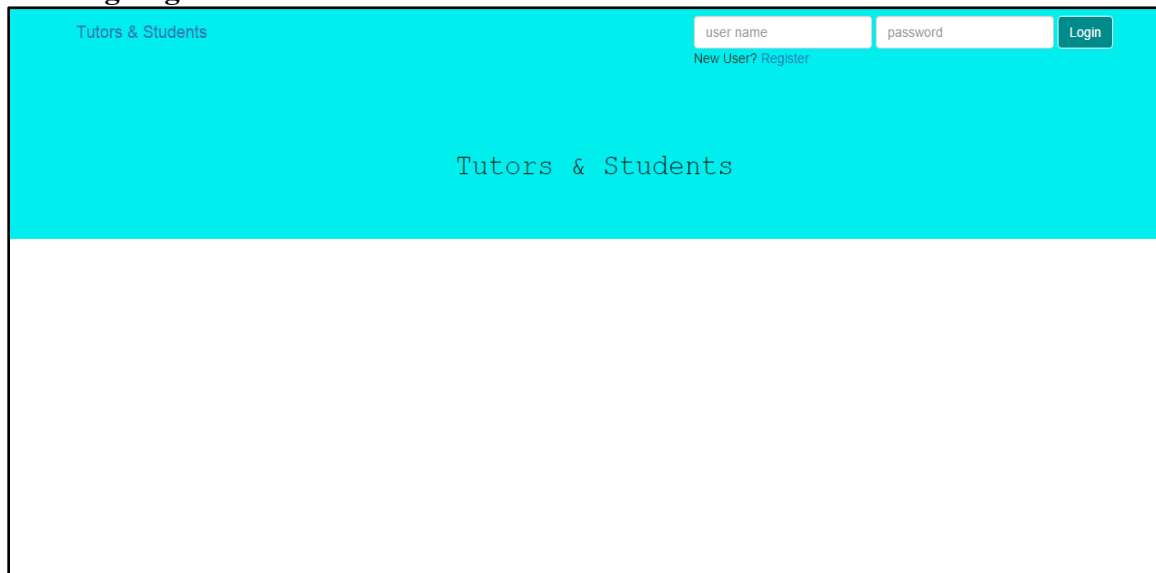
2.7.3 The public profiles of all tutors will have an “Achievements” tab, where the viewer can view all the achievements uploaded by the independent tutor/agent.

3. *Interface Requirements*

3.1 User Interfaces

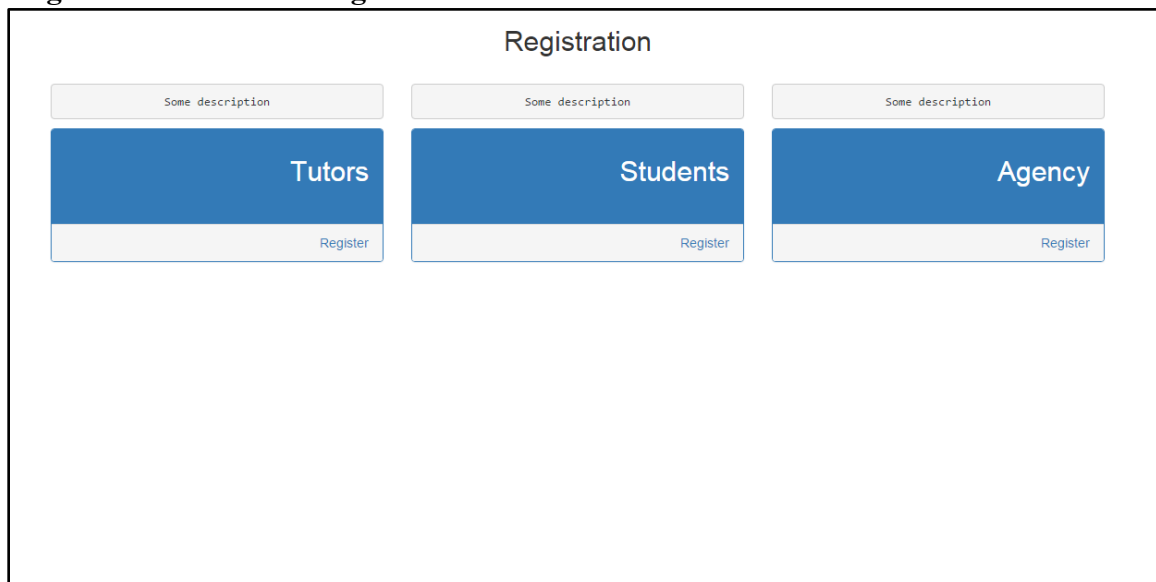
The various proofs of concept screenshots are shown below:

Landing Page



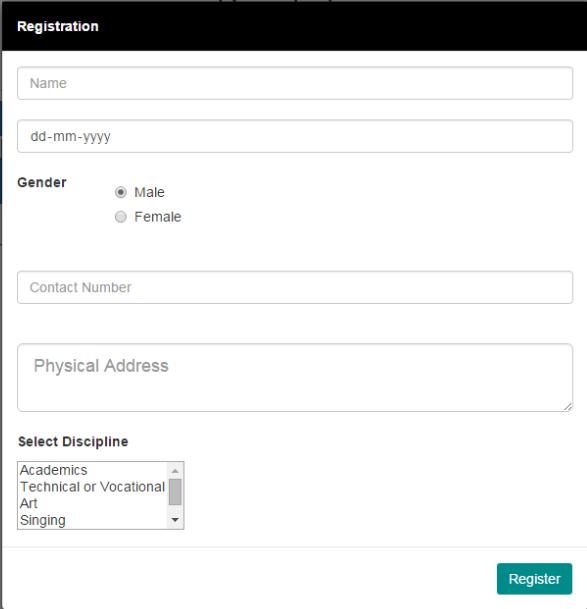
The screenshot shows a landing page with a bright cyan header. In the top left corner of the header is the text "Tutors & Students". In the top right corner, there is a login section with two input fields labeled "user name" and "password", followed by a dark blue "Login" button. Below the input fields is a link that says "New User? Register". In the center of the header, the text "Tutors & Students" is displayed in a monospaced font. The main body of the page is white and currently empty.

Registration-Selection Page



The screenshot shows a registration selection page with a white background. At the top center, the word "Registration" is displayed. Below it, there are three identical registration cards arranged horizontally. Each card has a light gray header with the text "Some description". The main body of each card is dark blue with white text. The first card is labeled "Tutors", the second "Students", and the third "Agency". At the bottom of each card, there is a light gray footer with the word "Register" in blue text.

Registration Form

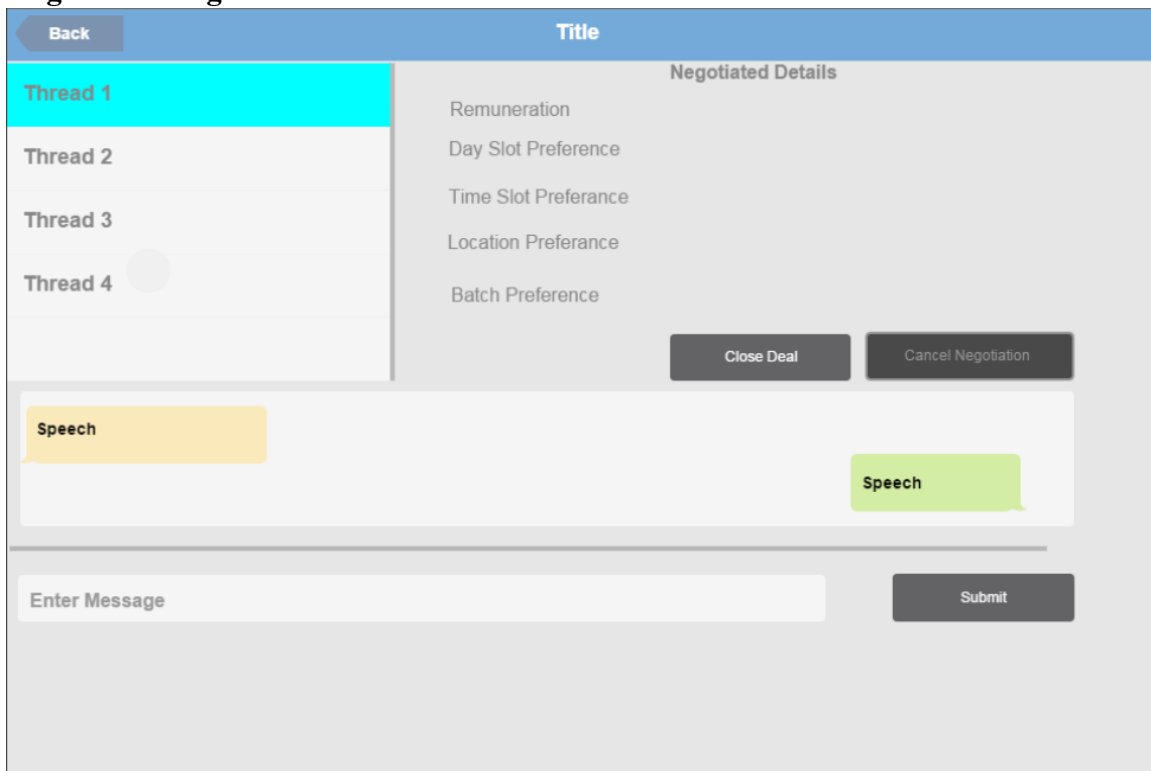


The registration form is a modal window with a black header labeled "Registration". It contains the following fields and options:

- Name:** A text input field.
- Date:** A text input field with the placeholder "dd-mm-yyyy".
- Gender:** Two radio button options: "Male" (selected) and "Female".
- Contact Number:** A text input field.
- Physical Address:** A text input field.
- Select Discipline:** A dropdown menu with the following options: "Academics", "Technical or Vocational", "Art", and "Singing".
- Register:** A green button at the bottom right of the modal.

The background shows a blurred interface with a "Register" button and the word "Agency".

Negotiation Page: Before Deal Closure



The negotiation page has a blue header with a "Back" button and a "Title" field. The main content area is divided into two sections:

- Thread List:** A list of four threads, with "Thread 1" highlighted in red. Each thread has a circular progress indicator next to it.
- Negotiated Details:** A list of details for the selected thread, including "Remuneration", "Day Slot Preference", "Time Slot Preference", "Location Preference", and "Batch Preference".

At the bottom of the "Negotiated Details" section are two buttons: "Close Deal" and "Cancel Negotiation".

Below the thread list is a "Speech" section with a yellow speech bubble on the left and a green speech bubble on the right, both labeled "Speech".

At the bottom of the page is a text input field labeled "Enter Message" and a "Submit" button.

Negotiation page: After Deal Closure Proposed

Back

Title

Thread 1

Thread 2

Thread 3

Thread 4

Negotiated Details

Remuneration

Day Slot Preference

Time Slot Preference

Location Preference

Batch Preference

Deal Closure Proposed.

Acknowledge Deal

Reject Deal

Speech

Speech

Enter Message

Submit

Student Dashboard

Back

Student Dashboard

Profile Name

Update Profile

Negotiations

Search Tutor

Name

Physical Address


Contact Number

Email

Independent Tutor Dashboard

[Back](#)

Tutor Dashboard



Profile Name

Update Profile

Negotiations

Manage Student Records


Achievements

Name		
DOB		Gender
Contact Number		
Email		
Disciplines		
Physical Address		
Experience		
Remuneration		
Day Slots		
Batch Strength		

Agent Dashboard

[Back](#)

Agency Dashboard



Profile Name

Update Profile

Negotiations

Register Tutor


List My Tutors

Name of the Agency	
Name of Contact Person	
Contact Number	
Email	
Physical Address	

Public Profile of Student

[Back](#)

Student Public Profile



Profile Name

Name	
Physical Address	
Contact Number	
Email	

Public Profile of Tutor

[Back](#)

Tutor Public Profile


Profile Name

Negotiate

Review

★ ★ ★ ★ ★
Rate

Basic Information


Achievements

Ratings and Reviews


Name		
DOB		Gender
Contact Number		
Email		
Disciplines		
Physical Address		
Experience		
Remuneration		
Day Slots		
Batch Strength		

Search Tutor-Results


Search Tutor



Tutor Name

 Agent Verified

View Profile




Tutor Name

View Profile

Refresh Search Criteria


List of Tutors registered under an Agent



Tutor Name

Remove

Update Profile




Tutor Name

Remove

Update Profile


List of Registered Students

List of registered students



Student Name

Remove

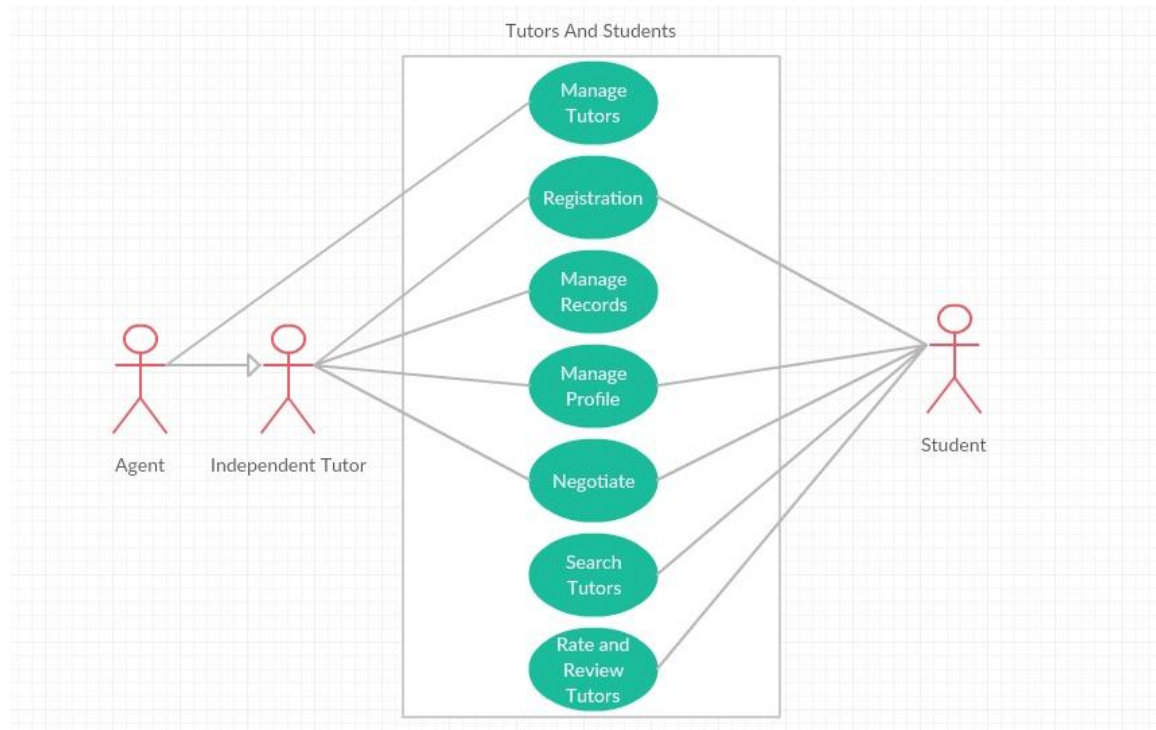


Student Name

Remove

4. Use Case Model

4.1 Use Case Diagram



4.2 Use Case Description

Use Case ID:	4.2.1		
Use Case Name:	Registration		
Created By:	BhaskarGhoshDastidar	Last Updated By:	BhaskarGhoshDastidar
Date Created:	27-02-2016	Date Last Updated:	03-03-2016

Actor:	Agent, Independent Tutor, Student
Description:	Actors can register themselves with the system. This registration provides the actor with their own user account, by virtue of which they can navigate/use the different functionalities of the system. The Registration use case can be divided into three sub categories: Agent, Independent Tutor, and Student Registration.
Preconditions:	<ol style="list-style-type: none">1. The actor is on the landing page of the portal, either deliberately (by visiting the portals URL) or due to some illegal access issue.2. Actor clicks on Register button.
Postconditions:	The actor is registered to the system and is redirected to the dashboard and a new session starts.
Priority:	High.

Frequency of Use:	High.
Flow of Events:	<ol style="list-style-type: none"> 1. The System redirects actor to the registration choice page, where the actor has the option to select one of three types of registration – Agent, Independent Tutor, or Student. Actor selects any one. 2. The System presents the actor with a form, which the actor fills up. The fields required for the three categories are – <ol style="list-style-type: none"> a) Agent – <ol style="list-style-type: none"> i) Name of Agency ii) Name of contact person. iii) Email address iv) Contact Number v) Physical address vi) Profile Picture upload. All fields except v) and vi) are mandatory. b) Independent Tutor – <ol style="list-style-type: none"> i) Name of Independent Tutor ii) Date of birth iii) Gender iv) Email address v) Contact Number vi) Physical Address vii) Discipline(s) Taught viii) Sub Category of discipline(s) taught ix) Number of years of experience. x) Expected Remuneration. xi) Day – Time slots xii) Location – Student’s location / Own Location. xiii) Batch strength. xiv) Self – Description (About me) xv) Profile picture upload. All fields except v), vi) and xv) are mandatory. c) Student – <ol style="list-style-type: none"> i) Name of Student. ii) Email address iii) Contact Number iv) Physical address. v) Profile picture upload. All fields except iii), iv) and v) are mandatory 3. Actor clicks the submit button. 4. System validates all data. 5. System acknowledges that registration has been made, and notifies the actor. 6. System then redirects actor to their dashboard page.
Alternative Flows:	<p>Normal flow up to step 4, following which –</p> <ol style="list-style-type: none"> 1. System detects some error in data or some server error leads to registration failure. 2. System notifies actor that registration was unsuccessful, and redirects the actor to the registration form page.
Exceptions:	

Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	4.2.2		
Use Case Name:	Manage Profile		
Created By:	Eric Joel Roy	Last Updated By:	Eric Joel Roy
Date Created:	27-02-2016	Date Last Updated:	02-03-2016

Actor:	Student, IndependentTutor or Agency		
Description:	An actor can login to his profile from the welcome page. An actor can also make changes or add new information to their profile. An actor can also view any other profile on the system.		
Pre-conditions:	<ol style="list-style-type: none"> 1. For logging in, the actor visits the landing page of the portal. 2. For updating own profile, or viewing profile of some other user, actor is already logged in. 3. For viewing profile, either a student discovers a profile from the search results, or any actor can click on the list all X option in their dashboard – In either case user is presented with a results page showing a list of all necessary profiles, each with a view profile button. 		
Post-conditions:	<ol style="list-style-type: none"> 1. The username and password entered for logging in should match the credentials in the database for a profile. 2. The changes made by the student/ independenttutor/agent are confirmed to be saved to his/her profile in the database by the respective user and the changed profile is shown from further logins. 3. Actor views the public profile of another actor. 		
Priority:	High		
Frequency of Use:	High.		
Flow of Events:	<p>Authentication:</p> <ol style="list-style-type: none"> 1. System displays a form to enter email and password. 2. Actor enters credentials and hits “Login” button. 3. System validates the credentials against information in its database. 4. System authenticates the actor and starts a new session. 5. Actor’s login is acknowledged and actor is redirected to dashboard page. <p>Update profile:</p> <ol style="list-style-type: none"> 1. Actor clicks the Update Profile button on the dashboard. 2. System redirects actor to a page having all their information. The form interface is similar to the one described in Use case 1.2 3. Actor changes any field(s) they want. 4. Actor hits the Save changes button. 5. System validates all changes. 		

	6. System acknowledges that changes were made. 7. Actor is redirected to the dashboard. View a profile: 1. Actor clicks on the View Profile 2. Actor is redirected to the public profile page of the concerned other actor.
Alternative Flows:	Login: 1. System finds that the combination for email, password is not a valid combination 2. System sends a message to the actor informing that the provided combination is invalid. 3. Actor acknowledges that invalid data was sent. 4. System redirects actor to landing page. Update: 1. System finds that one of the changes made by the actor is not allowed. 2. System saves other valid changes and informs actor about invalid changes. 3. Actor acknowledges that invalid data was sent 4. System redirects actor to the dashboard.
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	4.2.3		
Use Case Name:	Manage Tutors		
Created By:	Sayan Bose	Last Updated By:	Abhirup Mukherjee
Date Created:	03-03-2016	Date Last Updated:	04-03-2016

Actor:	Agent
Description:	Agents register tutors with the System. This creates a separate type of tutor profile – one managed by the agent, with an agent authenticated tag accompanying the profile.
Preconditions:	Agent has a registered account. Agent is logged in to the System.
Postconditions:	Agent completes various management activities regarding the agent-registered tutor profiles.
Priority:	High
Frequency of Use:	It will depend on the agent's wish, how many tutors he wants to register with the system.
Flow of Events:	Registration: 1. Agent clicks the register tutor option available in the agent dashboard. 2. System redirects agent to a form similar to the one for tutor registration in 1.2

	<ol style="list-style-type: none"> 3. Actor clicks the submit button. 4. System validates all data. 5. System acknowledges that registration has been made, and notifies the agent. 6. System then redirects agent to their dashboard page. <p>Management: The agent can keep updating the tutors' information.</p> <ol style="list-style-type: none"> 1. Agents clicks the list my tutors button. 2. System displays a list of all tutors registered by the agent. 3. Agent selects the update button corresponding to one of the results. 4. The System redirects Agent to a form similar to updating form mentioned in 2.2 5. Agent changes any field(s) they want. 6. Agent hits the Save changes button. 7. System validates all changes. 8. System acknowledges that changes were made. 9. Agent is redirected to the dashboard.
Alternative Flows:	<p>Registration:</p> <p>Normal flow up to step 4, following which –</p> <ol style="list-style-type: none"> 1. System detects some error in data or some server error leads to registration failure. 2. System notifies agent that registration was unsuccessful, and redirects the actor to the registration form page. <p>Management: Updating</p> <ol style="list-style-type: none"> 1. System finds that one of the changes made by the actor is not allowed. 2. System saves other valid changes and informs actor about invalid changes. 3. Actor acknowledges that invalid data was sent. 4. System redirects agent to the dashboard.
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	4.2.4		
Use Case Name:	Negotiate		
Created By:	Devanjan Banerjee	Last Updated By:	Abhirup Mukherjee
Date Created:	28-02-2016	Date Last Updated:	04-03-2016

Actor:	Student, IndependentTutor or Agency
Description:	The student can request for a face to face meeting, or negotiate terms of tutoring, online, either directly with an independentprivate tutor, or with any agent registered tutor, whom the student has found from search results. In case the tutor in consideration is

	registered under an agent, the negotiations are made through the agent, with the agent being the single point of contact.
Pre-conditions:	<ol style="list-style-type: none"> 1. The Actor is logged in. 2. For students - The “Negotiate” option is clicked. 3. For independenttutors/agents – The negotiation request is added to the ‘Open Negotiations’ list.
Post-conditions:	<ol style="list-style-type: none"> 1. The negotiation between student and independenttutor/ agent ends as successful. System marks the thread as a successful negotiation and adds the student to the concerned tutor’s registered students list. 2. The negotiation between student and independenttutor/ agent ends unsuccessful. System marks the thread as unsuccessful negotiation and the process stops.
Priority:	Medium
Frequency of Use:	Medium.
Flow of Events:	<p>Student hits the “Negotiate” button, on the profile page of a tutor, discovered from search results. There are two cases -</p> <p>A) The profile is of an independent private tutor. B) The profile is that of some tutor registered by an agency.</p> <p>Case A)</p> <ol style="list-style-type: none"> 1) The negotiation page shows the parameters as mentioned on the tutor’s profile. A message box is also shown for communicating. 2) Student sends the queries through messages in the message box. 3) IndependentTutor opens his/her message box at their convenience and answers the queries through messages. If they wish to modify the negotiation parameters, they can do the same. Parameters cannot be modified by the students. 4) When one of the parties is satisfied they click the “Close Deal” Button. 5) The other party can then either acknowledge or reject. 6) System marks the thread as a successful negotiation if acknowledged and adds the student in the registered students list of the independenttutor. 7) System returns to negotiations page if rejected. <p>Case B)</p> <p>Process mentioned above is the same, except that negotiations and parameter changing is done by the agent, as the tutor is not an actor in this case.</p>
Alternative Flows:	<p>Student follows normal flow till the end</p> <ol style="list-style-type: none"> a. Student is dissatisfied with the terms discussed during negotiation. b. Student hits “Renegotiate” button.

	c. System Marks the thread as an unsuccessful negotiation d. Student restarts the process from normal flow.
Exceptions:	
Includes:	
Special Requirements:	1. Development of a special messaging sub-system for the negotiation process to take place in a proper manner. 2. Development of a negotiation tracking sub-system for final confirmation of negotiation details and recording the same
Assumptions:	The agent is able to keep a tab on the negotiation process till final assignment of tutor to student, without divulging contacts of either party to the other.

Use Case ID:	4.2.5		
Use Case Name:	Search Tutors		
Created By:	Abhirup Mukherjee	Last Updated By:	Abhirup Mukherjee
Date Created:	27-02-2016	Date Last Updated:	03-03-2016

Actor	Student
Description:	Student searches for tutors by providing search criteria based on various parameters. The System then searches its database to find tutors who best match the given criteria, and tries to return results in an optimally ranked manner. The criteria to be considered and the rough definition of optimality are provided hereafter.
Criteria:	The search criteria to be specified are: <ol style="list-style-type: none"> 1. Discipline (Academics/Technical or vocational / Art/ Singing/ Instrument/ Dance/ Sports/ Others - specify). - Mandatory 2. Discipline Sub Category (Subject/Specific vocation / Type of art/ Type of singing [Indian - Classical, semi classical modern, Western, other], which instrument/ which type of dance/ which sport, etc.) - Mandatory 3. Remuneration (specify range) – Any by default 4. Day slot preferences (Daily, specific days) – Any by default 5. Time slot preferences (specify range) – Any by default 6. Location preference (At own location or at Tutors location, specify location) – Any by default 7. Experience preference (specify range/ qualifications) – Any by default 8. Age preference (specify range) – Any by default 9. Gender preference (select gender) – Any by default 10. Batch preference (Individual tutoring or in batch. If batch specify size of batch) – Any by default 11. Agent Authenticated preference (Mandatory/Not Mandatory) – Not Mandatory by default.
Optimality:	The optimality of search results refers to the following features: <ol style="list-style-type: none"> 1. The search algorithm will match all or most of the specified criteria, with higher priority assigned to filled value fields as compared to defaults.

	<ol style="list-style-type: none"> The search criteria form will also provide a graphical ordering box to arrange the filled value fields in a decreasing order of priority while searching. Default ordering will be as listed in the Criteria section. Based on the final priority list, determined from 1 and 2, the search engine will provide higher ranks to tutors which match more number of criteria, the score weights being skewed according to the priority list (higher priority has higher score multiplier) . The algorithm will also assign higher score to tutors with higher certified rating values, and more number of positive certified reviews. The scores calculated from 3 and 4 are to be summed/ aggregated to generate one final score for all profiles. Final results will be ranked by score, with higher scoring tutor profiles appearing higher in the list.
Preconditions:	<ol style="list-style-type: none"> Student is already logged in. Student has clicked on the “Find Tutor” Menu Button, and reached the screen for entering the various criteria.
Postconditions:	<ol style="list-style-type: none"> Student follows the link to Tutors’ profiles as provided on the results page or Student is dissatisfied with the results and hits the “Refresh search criteria” button and is redirected to the search criteria page to start afresh.
Priority:	High.
Frequency of Use:	High.
Flow of Events:	<ol style="list-style-type: none"> System displays the form for entering the search criteria. Student fills up the form according to needs. Student hits the “Find” button. System applies search algorithm as discussed previously and returns a ranked result set. Student browses through the results. Student likes particular results and follows the link to those tutors’ profiles.
Alternative Flows:	<p>Student follows normal flow till step 5 in Standard flow and then</p> <ol style="list-style-type: none"> Does not like any of the results. Hits “Refresh Search Criteria” button. System redirects to search criteria page. Student restarts the process from Standard flow step 1.
Exceptions:	<p>Search algorithm is unable to find any close matches either due to</p> <ol style="list-style-type: none"> Insufficient/ Ambiguously specified criteria or, Lack of proper tutors meeting specified criteria. <p>In such cases the System shows the nearest available result and in case of no matches, returns “No results Found” and a link to “Refresh Search Criteria”.</p>
Special Requirements:	<ol style="list-style-type: none"> Development of an optimal match-making algorithm to match tutors to search criteria and other factors from the student’s profile information.
Assumptions:	The various parameters used while searching are sufficient to find optimal matches as actually required by the student.

Notes and Issues:	
-------------------	--

Use Case ID:	4.2.6		
Use Case Name:	Rate and review Tutor		
Created By:	KaustavGhosh	Last Updated By:	KaustavGhosh
Date Created:	25-02-2016	Date Last Updated:	02-03-2016

Actor:	Student
Description:	<p>A student rates a tutor, on a scale of maximum 5 stars, based on the tutor's performance and the student's level of satisfaction. The rating system is proposed as follows:</p> <ul style="list-style-type: none"> i) 1 star- Very Poor ii) 2 stars- Poor iii) 3 stars- Satisfactory iv) 4 stars- Good v) 5 stars- Very Good <p>The student may also write a detailed review of the tutor if he/she wishes to. The System will have a record of all students that a particular tutor has marked as a student. If a particular student is either a current or ex-student of the concerned tutor and if the student rates/reviews that tutor, a "Verified Student" badge will be shown beside the name of that student in the rating/review section. On the other hand, if a student rates/reviews a tutor to who is not recorded as a student of the concerned tutor, the "Verified Student" badge will not be added. This Verified vs. Unverified ratings and reviews will affect the search ranking of the tutor.</p>
Preconditions:	Student is already logged in.
Postconditions:	The rating / review is added to the tutor's profile, and considered during subsequent search result rankings.
Priority:	Medium
Frequency of Use:	It will depend on the student's wish, i.e. whether he/she wants to rate/review a tutor or not.
Flow of Events:	<p>Rating:</p> <ol style="list-style-type: none"> 1. Student selects a Tutor to rate. 2. Student places a rating value and submits. 3. System checks whether the student is a registered current/ ex-student of the concerned tutor. 4. If the student is a registered current/ ex-student, the rating is added to the verified rating records of the tutor. Otherwise it is added to the unverified ratings records. 5. System acknowledges the rating and shows this rating in all subsequent calls for the tutor's profile. <p>Review:</p> <ol style="list-style-type: none"> 1. Student selects a Tutor to review. 2. Student writes the review. 3. Student selects a 'Positive' or 'Negative' indicator. 4. Student hits 'Submit'. 5. System checks whether the student is a registered current/

	<p>ex-student of the concerned tutor.</p> <p>6. If the student is a registered current/ ex-student, the review is marked and added to the verified review records of the tutor. Otherwise it is added to the unverified review records.</p> <p>7. System acknowledges the review and shows this review in all subsequent calls for the tutor's profile.</p>
Alternative Flows:	
Exceptions:	
Includes:	
Special Requirements:	<ol style="list-style-type: none"> 1. A rating section on the profile page of all tutors 2. A review section on the profile page of all tutors
Assumptions:	<ol style="list-style-type: none"> 1. A student should be willing to help other students looking for tuitions in the same field by providing a rating and a review of the tutor from whom he/she has taken tuitions. 2. The student should be honest about his rating and views about his/her private tutor.
Notes and Issues:	

Use Case ID:	4.2.7		
Use Case Name:	Manage Student Records		
Created By:	Devanjan Banerjee	Last Updated By:	Devanjan Banerjee
Date Created:	04-03-2016	Date Last Updated:	04-03-2016

Actor:	Agent, Independent Tutor.
Description:	The independenttutor updates his profile with the achievements of his students, or manages records. The agent visits the profile of his registered tutors and updates it with the student records / achievements.
Preconditions:	<ol style="list-style-type: none"> 1. Actor is already logged in. A) If the actor is an independent tutor he/she clicks the "Update Student Records/Achievements" button. B) If the actor is an agent he/she visits the corresponding tutor's profile and then clicks the "Update Student Records/Achievements" button.
Postconditions:	<ol style="list-style-type: none"> 1. Agent/IndependentTutor changes the respective fields for student records/achievements. 2. The changes are recorded and saved in the database.
Priority:	High, since it determines the ranking of the tutors.
Frequency of Use:	Depends on student achievements / records.
Flow of Events:	<p>Student records –</p> <ol style="list-style-type: none"> 1. The Actor selects the registered students tab with a list of all registered students. 2. Each result has a "Remove" button attached to it. 3. Clicking the remove button will remove the student

	<p>from the registered students list of the tutor.</p> <p>Student Achievements –</p> <ol style="list-style-type: none"> 1. Actor selects the achievements tab. 2. A list of previously added achievements is shown. 3. Actor selects the “Add Achievement” button. 4. A text area pops up, where the actor can enter the description of achievement, and submits. 5. The System adds the achievement to the list of achievements.
Alternative Flows:	
Exceptions:	
Special Requirements:	
Assumptions:	The achievements being entered by the independent tutor/ agent are authentic in nature. Since the achievements can be of varied nature so it is extremely difficult to authenticate the same. So the System assumes that agents/ tutors do not add incorrect achievements to their profiles.
Notes:	

5. *Glossary*

System: Refers to the Web portal provided as a service, and all activities that it encompasses.

Actor: Refers to one of the three classes of users of the System – Independent Tutor, Student, or Agent.

Independent Tutor: Private tutors who create and manage their own profiles i.e. they are not registered under any agent. They are actors.

Agent-Registered Tutor: These tutors are the ones whose profile is created and managed by some agent. They are not actors by default.

Tutor: Refers to both – Independent Tutors and Agent-Registered Tutors.

Student: Learner who require private tuitions in any field. Someone else (say parents) may also be managing their accounts, but that is beyond the lookout of the System. They are actors.

Agent: A representative/businessman who acts on behalf of private tutors in exchange for a commission. The tutors they register are the ‘Agent-Registered Tutors’. Agents are actors.

Dashboard: This is the default home page where an actor is redirected to when they log in. All profile management and other navigation activities can be done from this page.

Dashboard-Menu: This is the menu with various buttons which allow the actors to navigate throughout the System. It is an integral part of the Dashboard.

Register: An actor records or enters their name and other requisite details, and forms a valid account with the System. An Agent may also register private tutors, but such accounts can be operated only by the respective agents.

Log-in: An actor enters their e-mail and password to authenticate them to the system. This allows the actor to use the System functionalities depending on the type of actor they are.

Manage: Is the activity that refers to the general management of – Profile, Records, or Agent-Registered Tutors (specifically in case of actor being an Agent).

Records: A compilation of the list of registered students or achievements (of students) of a particular Tutor present. Records are maintained by either an Independent Tutor or by an Agent on behalf of Agent-Registered Tutors.

Profile: A summary or collection of information about an actor. This is viewed by other actors to know more about the actor.

Negotiate: Students and Independent Tutors/ Students and Agents discuss the terms of an arrangement/ confer with another in order to come to terms or reach an agreement.

Search: A Student tries to discover a Tutor in the field of his choice through the portal's search subsystem.

Rate: A Student assigns a rank or rating to a private Tutor based on the tutor's performance and the student's level of satisfaction.

Review: A Student appraises critically a private Tutor based on the tutor's performance and the student's level of satisfaction.

6. ***References***

Provide a list of all documents and other sources of information referenced in the SRS and utilized in developing the SRS. Include for each the document number, title, date and author.

Document No.	Document Title	Date	Author