**Teach.me**

*Real-time Video Tutoring Service for Math*

by Jack Li, Tommy Liu, Yuta Kato

*design document*

**brief description:**
We aim to create a dynamic, integrated online platform and community for real-time tutoring of math. Teach.me will allow users to ask for help on mathematical concepts that they would like clarified and receive immediate tutoring through live video interaction. Users can act as both students and tutors. As students, users can submit requests for tutoring help on a specific math topic. Other users acting as tutors can then elect to tutor a student on the topic for which they need help. A tutoring session will be a live video conference between a student and tutor on the Teach.me website. Users as students can then upvote or downvote their tutor based on their effectiveness.

**key goals and purpose of the app:**
Math tutoring is currently conducted almost completely in person and doesn’t take advantage of the emerging technologies of the internet. Online tutoring services do exist, but they are mostly scheduling tools that adhere to an old paradigm. There is no complete package solution, and everything is slow. We hope to bring learning into a new age. No more waits, no more nasty scheduling.

Our goal is to facilitate the connections between tutor and student by incorporating the entire tutor-searching process within our application. By hosting our own hassle-free video/audio system for users and instantly matching tutors with students, Teach.me will introduce a better, easier approach to tutoring.

**motivation for development:**
Current tutoring involves several scattered steps, such as finding a good tutor and planning out where/when to meet. **However,** Teach.me integrates the entire process into one easy application.

Scheduling may take days, if not weeks, between emails or other forms of communication and actually meeting may involve transportation for a face-to-face session or the installation of various software for an online session. **However,** we instantly match up students with willing tutors with video/audio feeds without the installation of any additional software.
concepts:

- **Users** - can act as both students and tutors. They can make Help Requests or start tutoring sessions.
- **Administrators** - the moderators of the website. They are able to ban questions and/or users as needed.
- **Tutoring Session** - a video session between two users initiated by one and accepted by the other.
- **Help Request** - a post the user makes public to other Teach.me users to indicate a request for help concerning a specific math topic.
- **Ratings** - accumulated upvotes or downvotes given to a user acting as a tutor by a user acting as a student.
- **Session Invitation** - An invitation from another user to join a Tutoring Session as a student.

feature descriptions:

- **Ask for help** - Users can request help concerning a specific subject.
- **Give help** - Users can initiate a video session with another user in order to give them tutoring.
- **File transfer** - Users will be able to share files and documents through our online service during a tutoring session.
- **Screen share** - our video feed will also allow users to share their screens so demonstrations are easier to show
- **Give and receive ratings** - Users can rate other users based on their performance as a tutor. Note that users can only receive ratings from users that they have tutored in a Tutoring Session.
context diagram:

![Context Diagram](image)

data model:

![Data Model](image)

wireframe diagram:

![Wireframe Diagram](image)
Note: the “User Home”, “Ask For Help”, and “User Bio” pages are accessible from all authenticated pages.

wireframe mockups:
security:
Potential security concerns and planned solutions:

- CSRF:
  - We will use Rails’ built-in CSRF protection.
- SQL Injection
We will use ActiveRecord's built-in sanitization.

- Packet Sniffing
  - We will send data over the HTTPS protocol.

- Theft of Password Data
  - We will use Rails' built-in has_secure_password function, which will salt and hash passwords before storing them.

- Cross-Site Scripting
  - We will escape user input before displaying it in the browser.

Higher level concerns:

- **private sessions**: we want our tutoring sessions to be one on one without any unexpected visitors intruding onto a video stream. Each room will be associated with a list of authorized users, and the app check each attempted visitor to the room against this list.

- **file transfer security**: we want the shared files to be accessible only to those in the session and not to just any user online. WebRTC takes care of secure media streaming and communication.

- **remote access of camera**: we will not allow remote access of the camera without any confirmation from the user to avoid violation of privacy. We take care of this issue by having the app ask for permission from the user to access their camera.

**design challenges:**

- **scheduling**: we want to manage the entire process of asking for and receiving tutoring through one web application. This endeavor requires us to think about scheduling of tutoring sessions. We know however, how difficult it is to implement effective scheduling in an app, so we decided to ease the scheduling process by having users start (offer) tutoring and accept (receive) tutoring on a real-time basis. If a user offers to tutor another person, they simply click on a button. If the tutee is available online to receive tutoring at that moment, they may accept or decline. Otherwise, a video connection is not made and the Help Request remains open.

- **tutors vs students**: we have to decide whether to differentiate between students and tutors. Will each user be both a student and a tutor or should we have them specify which type of account they are registering for in the beginning?
  - We have decided not to distinguish between the two potential classes of users. Instead, each user has the ability to ask for help or to help others. This allows for a more convenient and open platform.

- **user moderation**: we have to decide whether to have moderators, and how those moderators permissions are distributed. If we don’t have moderators, our system would likely have to encourage self-management through some sort of rating system.

- **rating system**: We considered two possible rating systems to use for tutees to rate the users who tutored them.
  - **up and down votes**
Advantages: This system is easy to understand and use - this would encourage users to actually use the rating system. Many sites like reddit.com and youtube use voting.

Drawbacks: The system does not allow a spectrum of rating, which would possibly be more informative than a binary rating system.

- **5 point scale**
  - Advantages: The ability to rate on a spectrum gives users more information about how a rater felt about their performance. The system is also widely used, such as with Yelp.
  - Drawbacks: The system requires the rater to take more time and thought/energy to use. This form of rating would likely be used less often than a voting system for the same user base.

Decision: We decided to use up and down voting because we want to encourage users to provide some kind of rating for their experiences.

- **cap on help requests**: To avoid cluttering and redundancy of questions asked, we will set a cap on help requests. Users may try to make too many Help Requests for themselves at any time. We can limit the amount of Help Requests users have open to 10.

**Teamwork**

Stakeholders
- Developers - Jack, Tommy, Yuta
- Mentors - Carolyn Zhang, Leonid Grinberg
- Consumers - Users: students and tutors who would like to give and receive help in math.

Resources
- **WebRTC** - “WebRTC offers web application developers the ability to write rich, realtime multimedia applications (think video chat) on the web, without requiring plugins, downloads or installs” - Webrtc.org
- **Heroku** - Online server to host our website
- **Computation** - 3 developers
- **Cost** - No cost requirement. We are using open source or free access software.
- **Time** - We have 5 weeks constrained by other classes and activities.

**Tasks**

<table>
<thead>
<tr>
<th>Section</th>
<th>Feature</th>
<th>Deadline</th>
<th>Developer</th>
</tr>
</thead>
<tbody>
<tr>
<td>MVP</td>
<td>Authentication</td>
<td>11/15/2013</td>
<td>Yuta</td>
</tr>
<tr>
<td>MVP</td>
<td>Video/audio</td>
<td>11/15/2013</td>
<td>Tommy</td>
</tr>
<tr>
<td>MVP</td>
<td>Help Requests</td>
<td>11/15/2013</td>
<td>Jack</td>
</tr>
<tr>
<td>Deliverable Type</td>
<td>Description</td>
<td>Date</td>
<td>Contributors</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------------------------</td>
<td>------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>MVP</td>
<td>User Homepage</td>
<td>11/15/2013</td>
<td>Yuta</td>
</tr>
<tr>
<td>MVP</td>
<td>Initial Landing Page</td>
<td>11/15/2013</td>
<td>Jack</td>
</tr>
<tr>
<td>Additional Deliverable</td>
<td>MVP Demo</td>
<td>11/18/13</td>
<td>Jack, Tommy, Yuta</td>
</tr>
<tr>
<td>Additional Deliverable</td>
<td>Design Critique</td>
<td>11/20/13</td>
<td>Jack, Tommy, Yuta</td>
</tr>
<tr>
<td>Additional Deliverable</td>
<td>Revised Design Document</td>
<td>11/24/2013</td>
<td>Jack, Tommy, Yuta</td>
</tr>
<tr>
<td>Additional Deliverable</td>
<td>Code Critique</td>
<td>12/2/13</td>
<td>Jack, Tommy, Yuta</td>
</tr>
<tr>
<td>Final Product</td>
<td>Administrator Accounts</td>
<td>12/2/13</td>
<td>Yuta</td>
</tr>
<tr>
<td>Final Product</td>
<td>Ratings</td>
<td>12/5/13</td>
<td>Tommy</td>
</tr>
<tr>
<td>Additional Deliverable</td>
<td>Final Demo</td>
<td>12/9/13</td>
<td>Jack, Tommy, Yuta</td>
</tr>
</tbody>
</table>

**Teamwork Risks and Management**
Covered in our team contract

**MVP**
Our MVP will consist of only the video/audio stream of our communications package as well as a basic matching of tutor and student in the context of mathematics with no rating system.