# Product Requirements for Peer Tutoring Mobile App

BB8

*Expected readership: Client Ray Liu, ECE651 TA and classmates, Professor Adel Fahmy*
*Version 1.0*

## Introduction

Peer-to-peer tutoring has been shown to be one of the most effective ways for students to learn, but currently, there aren't many services that help facilitate this. This peer tutoring mobile android app aims at Duke community and surrounding Durham community and helps match peer tutors with peer tutees at ease.

When first using, one can log in using the Duke NetID or other social media account or create a new account. Then the software will collect the user's primary information, including identity, interested fields and available time. After setting up, one can post a listing with a subject/time/location of a tutoring session or set up the filter to browse other tutors' offerings from the nearest ones. This app also has a payment system for the tutor to price their help.

The tutee can directly contact the interested tutor via a private chat system. The tutor and tutee will discuss and decide the final time and location to meet. When tutee quit the chat window, a window should pump up asking whether they have an agreement or not to keep track of history, as well as to offer information to the feedback system. Tutors and tutees can leave ratings and evaluations to each other. The tutor and tutee time will also be recorded and shown to others for reference.

The app is developed in Java language and released on the Google Play app store. The user data is stored and arranged on a Linux server system, so the app needs to communicate with the server to fetch data to the user.

## Glossary

| Index | Term | Explanation |
|---|---|---|
| 1 | Virtual Machine | An emulation of a computer system, which based on computer architectures and provides functionality of a physical computer |
| 2 | Server | A computer program or a device that provides functionality for other programs or devices |
| 3 | Linux | A free and open-source software operating system built around the Linux kernel |
| 4 | Unix | A multitasking, multiuser computer operating system that derives from the original AT&T Unix |
| 5 | Java | A general-purpose computer-programming language that is concurrent, class-based and object-oriented |
| 6 | Android | A mobile operating system developed by Google |
| 7 | Duke NetID | A serial of numbers that represent the identification of students of faculty at Duke University |
| 8 | SSL | Secure Socket Layer, which is a cryptographic protocol |

| | | |
|---|---|---|
| | | that provides communication security over a computer network |
| 9 | hash | The process that maps data of arbitrary size to data of fixed size |
| 10 | Salt | Random data added to a one-way function that hashes data |
| 11 | UI | User Interface, is the space where interactions between humans and machines occur |

## Functional Requirements

1. Registration:
    a. Users should be asked to register or login using other popular Apps when they enter the app.
    b. If they are students from Duke University, they should be able to login using their NetID. Besides, users outside of Duke should also be able to login using their Facebook or WhatsApp.
    c. If users do not want to login with other apps, they should also be able to register using email or phone number.
2. Info collection:
    a. at the first time a user logs in, he should be asked to complete a profile form. In the form, basic information including Nickname, age, job or which year in college will be collected, as well as what you can tutor and the time and location for the tutoring.
    ~~b. Additionally, tutor should be able to set a price for a tutoring session.~~
    b. The tutor could delete the session he or she has posted. The tutee who applied for this session and  the tutee who has made an appointment on this session will receive a message about this deletion.
3. Interface:
    a. an interface where user should be able to search by key word of intended subject to learn, ideal time and location.
    b. Results are shown after he click "search".

~~4. Result filter:~~
    ~~a. users should have the right to check results based on different priorities. For example, if user chooses "location", the rank of results should be based on how far the location of tutoring is from the location the user filled in when searching.~~
    ~~b. Filter options should include: location, tutor ranking, tutor experience and etc.~~

~~5.~~4. Tutor info check:
    a. when results of a search are shown, user should be able to click on the link to check for detail information.

Formatted

~~6. Private chat:~~

~~a. if user is satisfied with one tutoring session, he should be able to start a private chat with the tutor.~~

~~b. Every time user exits the chat session, a window should pump up on the user side to ask whether they've decided to start tutor or not, in order to trace the tutoring history of a tutor.~~

~~c. Once user clicked YES, feedback to the tutor will be available, as well as that the payment system is ready.~~

~~7. Payment system:~~

~~a. if an agreement has been reached between tutor and tutee, tutee should be able to pay through the app to the tutor. It's required to be safe and secure.~~

~~8.~~5. Feedback system: feedback should include various factors, for example, punctuality.

~~9. Interact with other system:~~

~~a. Login with other apps.~~

~~b. Link to library reservation in private chat session.~~

~~c. Export schedule to calendar.~~

7. Message system:

a. if an application is accepted by the tutor, the tutee will receive a notification message

b. if a tutee applies for a session, the tutor will receive a notification message

c. if a session is deleted, all the applicants will receive a notification message

8. Profile

a. User can see his profile in this page

b. User can also edit his profile

9. Confirmation

a. After tutor gets many applications for one session, he can accept one applicant. Then all other applications will be denied.

**Formatted:** Indent: First line: 3 ch

**Formatted:** Normal, No bullets or numbering

**Formatted:** Font: (Asian) SimSun, (Asian) Chinese (PRC)

## Non-functional Requirements

### Performance (Speed, Size)

NF 1 The server side should allow for at least 10000 logged in users.

NF 2 The server side should be able to process 10000 searches per second (throughput).

NF 3 The latency for the user to get search results should be no more than one second under perfect network conditions.

### Security

NF 4 Data should be transmitted via SSL between the mobile app and the server side.

NF 5 (optional) database should be encrypted.

NF 6 Passwords should be stored as salted hashes.

NF 7 The server side should validate any data received clients.

NF 8 The server side should not use weak passwords for administration features.

### Ease of Use

NF 9 The app should use UI components such as icons, tooltips, etc., to help user understand the functionalities.

NF 10 The app should contain a user manual or a link to online help.

### Reliability

NF 11 The server side should have at least 99% availability.

### Portability, Platforms supported, etc

NF 12 The app is free to be downloaded on Google Play Store.

NF 13 The app operates on Android platform and is compatible with most of the Android devices on the market.

NF 14 The server-side software is written in Java and runs on Linux.