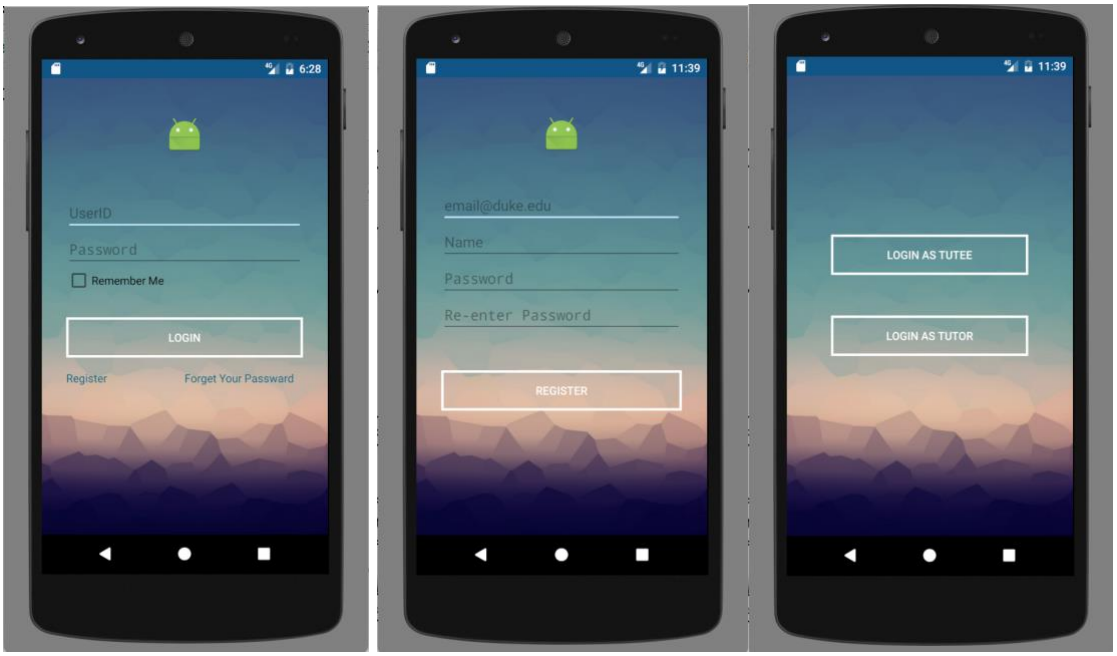
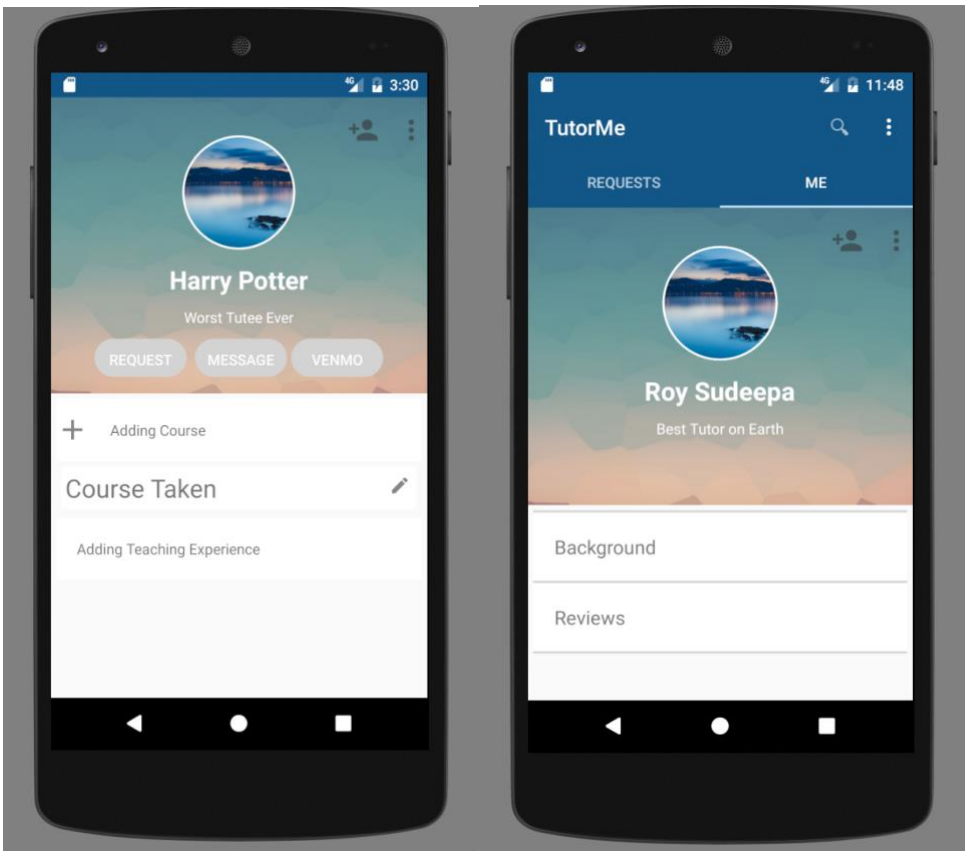
TutorMe Android App

1. Description

Our application is an android APP that provides a platform on which students can find tutors. There are a lot of tutor applications in the market, but none of them utilizes the course code system of Duke. Our APP, however, accurately guides Duke students to find the course they feel difficult with and accurately find the tutor who expertise in that course. We provide the scheduling system to help student make easy appointments. Each appointment specifies course code, time, place, payment and etc..We also provide a rating system that allows students to comment on the session taught by tutor.

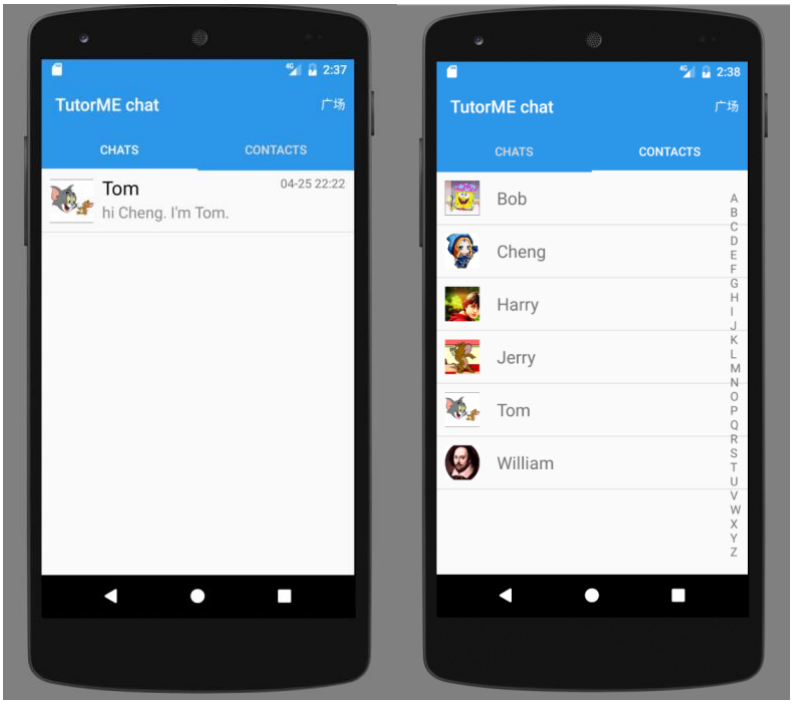
When the app is installed, user will be asked to login or sign up as a tutee. Email verification for signing up is required. After the registration, the user may apply for a tutor identity, but further verification is required. User applied to be tutor will have both the tutee and tutor identity in this app. They can switch to tutor or tutee mode at any time through the Choose Identity page below on the right. Every time a user login or the first time a new user register, he/she will be guided to the Choose Identity page first and they must choose an identity to login with.





The page on the left is a tutee personal page. User who logged in as tutee will be guided to this page. He/she can find a course, request a lesson, message a friend or venmo a tutor in this page. He/she will also see his “Course Taken” history in this page. To request a lesson, the tutee will first be asked for a course code. Then a list of tutors will show up according to the selection of course code. The tutee may select a preferred tutor and send a request to make an appointment.

The page on the right is a tutor personal page. User who logged in as tutor will be guided to this page. He/she can see the requests from student and his/her own reviews.

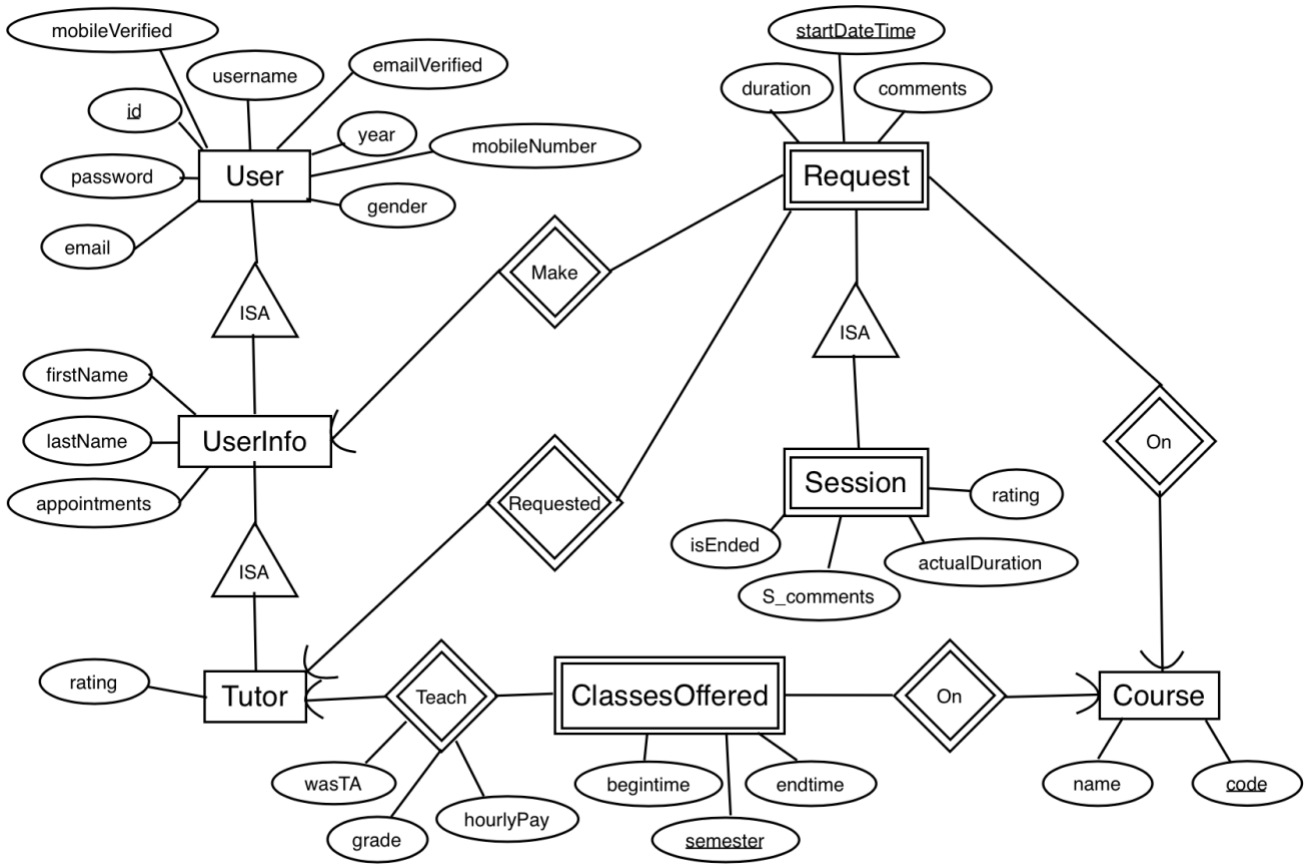


When the request is accepted by the tutor, the schedule will be validated in the session page. When the session is finished, attributes like “payment” and “actual duration” will be added to the session and a rating system will be available for the user.

Techniques:

The app is connected to an online database service called “LeanCloud”. It is an Non-Sql database with every tuple stored as an AVObject, each of which has a unique id. The online database uses pointers to tuples and pointers to relations (i.e. references) to connect different classes. The app also has a local SQLite database to store user information locally in order to save the server communications.

2. Final E/R diagram



We changed the E/R diagram quite a lot. Here are the changes:

a. In the previous two milestones, we separated the two groups of people -- Tutor and Tutee -- into two entities, but in the final version, we combine these two into a UserInfo entity and make Tutor an ISA under UserInfo. This is because in our final version of the app, everyone signs up as a tutee at first, and then one can choose whether to become a tutor. Also, this makes it easier to be both a tutor and a tutee for different classes.

b. We didn't think about the real situation about the app, so this part is completely new. When users sign up and login, we need to store their login information in a separate entity to protect everyone's privacy. Therefore, we made UserInfo an ISA of User, which contains sensitive information including emails, mobile phone numbers, usernames and passwords. We can then prevent all users from accessing this information.

c. Before, we had a tutor directly connected with a course. However, a tutor can't tutor a course multiple times in different semesters without deleting the old history. After adding the ClassesOffered entity, all history can be preserved, and we can store more info about the tutor's TA status, his/her grade on this particular class, and even the payroll info.

d. There used to be a Schedule entity containing both the requests and the tutoring history. This time, we made the Session entity (successful reservations) an ISA of Request, as they have the same tutorID, tuteeID, course code, and start time. When a tutor accepts a

request, we add a tuple into the Session entity. In this way, all request histories are still shown and we can clearly know which requests are actually accepted.

3. Assumptions

As demonstrated in the ER diagram, the data we have mainly decomposes into elements of user and relationship of tutor-tutee pair. The assumption we are making about the data is limited. We do not even assume people have to enroll in a class to be tutee for that class. In practice, everyone first registers as a tutee in the system (stored in user_info relation). When a user tries to log in as a tutor, he or she will be required to complete a form to register the tutor identity. The registration will include questions like what classes have been taken.

4. Database Table Schemas

```
Table User{
        id INTEGER NOT NULL PRIMARY KEY
        username VARCHAR(20) NOT NULL UNIQUE
        password VARCHAR(20) NOT NULL
        year VARCHAR(10)
        gender VARCHAR(10)
        email VARCHAR(100) NOT NULL UNIQUE
        emailVerified BOOLEAN
        mobileNumber VARCHAR(100) NOT NULL UNIQUE
        mobileVerified BOOLEAN
}

Table UserInfo{
        id INTEGER NOT NULL REFERENCE User.id
        year VARCHAR(10)
        gender VARCHAR(10)
        firstName VARCHAR(20) NOT NULL
        lastName VARCHAR(20) NOT NULL
        appointments INTEGER
        isTutor BOOLEAN NOT NULL
        rating FLOAT
}

Table Course{
        code VARCHAR(20) NOT NULL PRIMARY KEY
        name VARCHAR(100) NOT NULL
}
```

```
Table ClassesOffered{
        id INTEGER NOT NULL REFERENCE User.id
        code VARCHAR(20) NOT NULL REFERENCE Course.code
        wasTA BOOLEAN
        grade FLOAT
        hourlyPay FLOAT
        semester VARCHAR(20) NOT NULL
        begintime VARCHAR(10)
        endtime VARCHAR(10)
}

Table Request{
        tuteeid INTEGER NOT NULL REFERENCE User.id
        tutorid INTEGER NOT NULL REFERENCE User.id
        cid VARCHAR(20) NOT NULL REFERENCE Course.code
        startDateTime VARCHAR(20)
        duration INTEGER
        comments VARCHAR(1000)
}

Table Session{
        tuteeid INTEGER NOT NULL REFERENCE User.id
        tutorid INTEGER NOT NULL REFERENCE User.id
        cid VARCHAR(20) NOT NULL REFERENCE Course.code
        startDateTime VARCHAR(20) REFERENCE Request.startDateTime
        actualDuration INTEGER
        rating FLOAT
        isEnded BOOLEAN
        S_comments VARCHAR(1000)
}
```