For Google Login to Work

You need to have the key_file.txt I shared on slack, and put it in the root SlugChat directory, next to manage.py. Python will read that file and insert it into our home/templates/home/index.html template so we can communicate with google login using our private key.

You also need to use localhost:8000/, 127.0.0.1:8000 won't work because of some Google rule.

How to access our database

The database is stored in our home/ directory, so if you want to interact with it, you need to import the models:

```
from home.models import User
from home.models import Roster
from home.models import Course
```

How to get the current user from the database

We keep track of whether a user is logged in based on the 'email_address' session variable. If that variable is set, then the user is logged in (and has a corresponding User entry in the database).

```
if 'email_address' not in request.session:
    return HttpResponse('You need to log in to do that')
else:
    # email_address is the email of the current user
    email_address = request.session['email_address']
    # Get the user with this email address from the db
    user = User.objects.get(email=email_address)
```

How to get the courses a user is enrolled in

The courses a student is enrolled in is accessed via our Roster table. The roster table relates a User to a Course, so we can access all the classes a User is taking by accessing the corresponding Roster with our User.

```
email_address = request.session['email_address']
user = User.objects.get(email=email_address)
roster = user.roster_set.all().all()
for course in roster:
    print(course.courseID)
```

How to restrict a page for {Professors, TAs, Students}

We can check if there is a User in our database that has the status we are looking for.

```
email_address = request.session['email_address']
if not User.objects.filter(email=email_address,status="PR").exists():
    return HttpResponse('Only professors can see this page.')
```

How to tell if a user is logged in:

Pass the request object from your view function to the logged_in function. It returns true or false based on whether they are logged in. The logged_in function is a simple function I wrote to check if the email_address variable is in the request.session dictionary. It is located in slugchat/functions.py.

```
from slugchat.functions import logged_in
…
def myview(request):
    if not logged_in(request):
        return HttpResponseRedirect('/')
```

How our database is structured:

User model

1. firstName
2. lastName
3. School
4. studentID
5. email (This is the unique identification we use for every user.)
6. status ("TA", "PR", or "ST")
7. profile_pic (This row is set based on your google profile picture. You can also change it in /profile/buildprofile/?update=true)
8. completeProfile (This flag tells us when a user has completed their profile)

User.get_status(self):
This function makes getting the status choice more readable. You can call it on a User object and it will return either Teaching Assistant, Student, or Professor.

Course model

1. professor (A foreign key to a User)
2. ta (Another foreign key to a User)
3. school
4. title

Roster Model

1. studentID (A foreign key to a User)
2. courseID (A foreign key to a Course)

How database insertions happen right now

Brand new user:

1. They log in from the home page.
2. The javascript in /home/template/home/index.html sends a post request to tokensignin, which contains all of the information we have from their google account. This includes first name, last name, email address, and google profile picture.
3. A new User table is created and populated with all the information Google has given us. Their completeProfile row is still set to false, because they haven't yet completed their profile.
4. The user is redirected to profile/buildprofile. Here they can edit every field in their User database entry except their email address and completeProfile. Once this form has been successfully submitted, the completeProfile row is set to true and users will no longer be redirected to /buildprofile.
5. User now has a complete profile.

Old user:

1. After completing their profile, a user can go back and edit their profile from the /profile/ page. The link takes the user to /profile/buildprofile/?update=true. It is this session variable that tells /profile/ that we want to update the table.