BASS & V

TA & Co.

Fall 2017

Overview:

A web application for professors to manage their TAs. It will allow to assign assignments and track the hours worked for each assignment. We got this idea as from Ani as she is a TA and she thought that the way that the Professors and the TA's communicated was inefficient as the emails kept getting lost in her inbox. Also, having a specific application for specific task is convenient and organized. Our application is innovative because there doesn't exist an application for streamlined communication between professors and TAs that comes along with personalized displays of work assignments, charts, and updates.

Team Members:

Ani Gevorgyan
Salem Couja
Vincent Tran
Shivangi Singh
Brendon Delgado

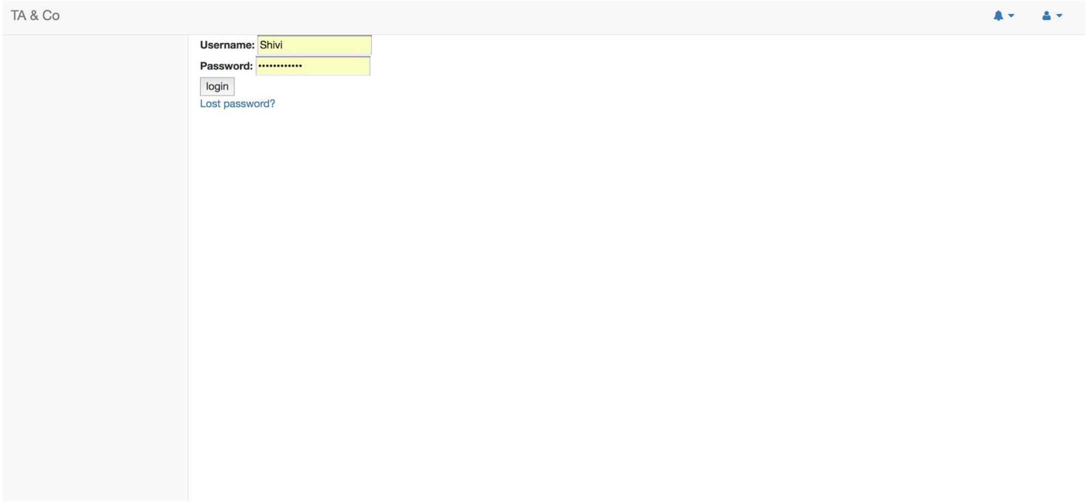
Github Repository:

https://github.com/Vinecnt/BASS-V

User Interface:

Login page

Allows the user to log in with the username and password



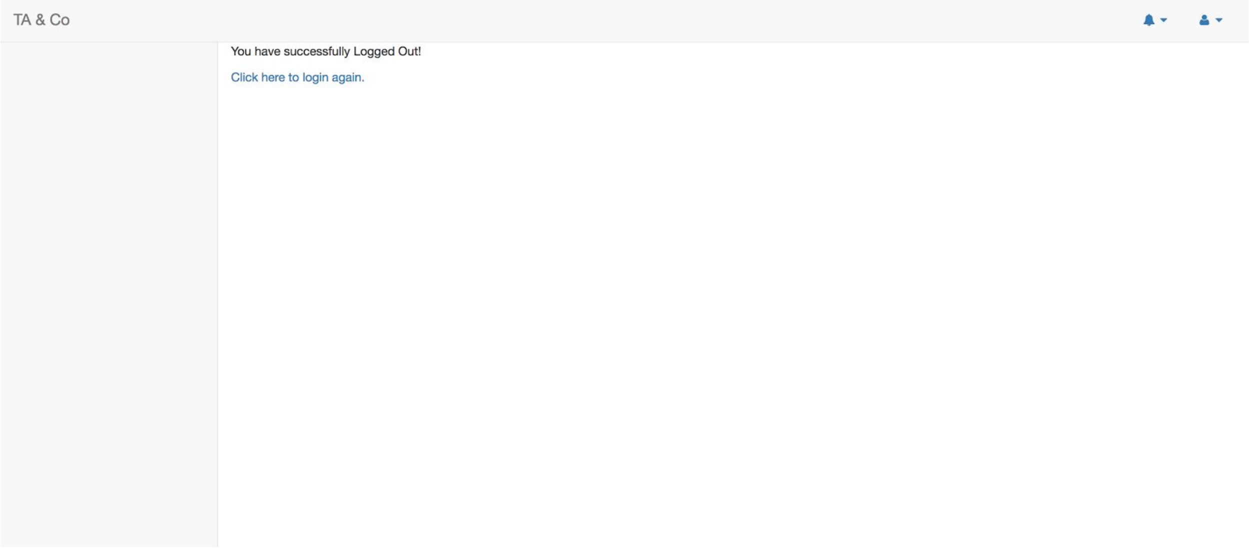
TA & Co

Username: Shivi

Password: ••••••••••••

Logout page

After the user is logged out of the system, they will see this page

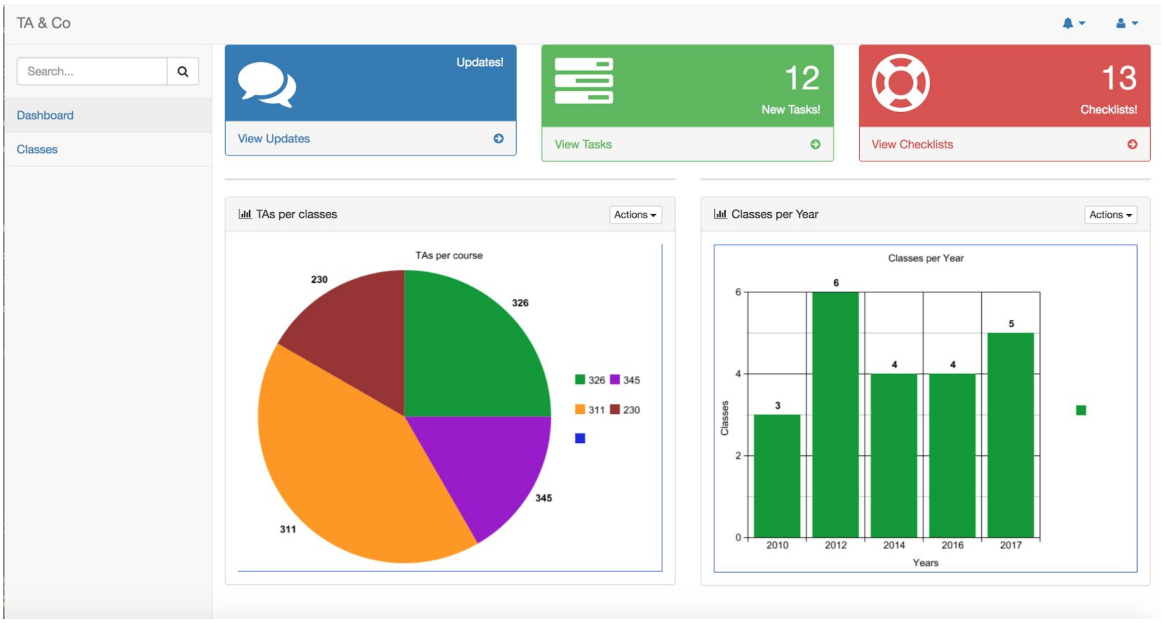


You have successfully Logged Out!

Click here to login again.

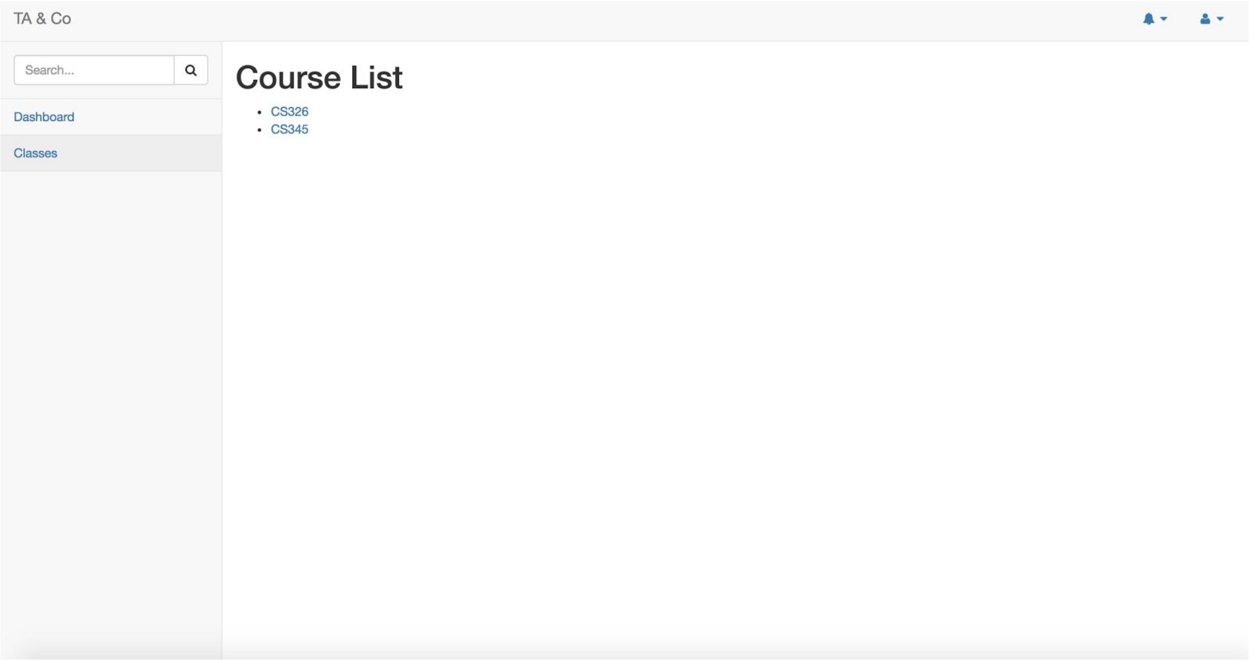
Dashboard

Provides a short information via charts specific to the user, has the options to view the updates, tasks and checklists(which is part of our future implementation)



Class list

Shows the list of the classes user is part of



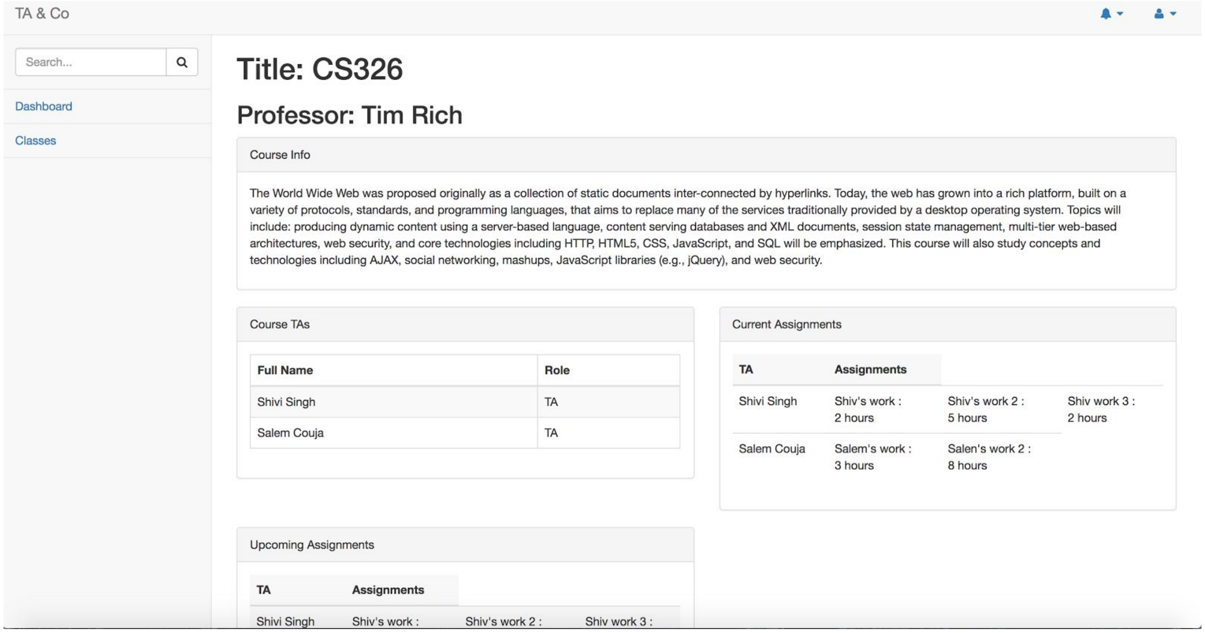
Course List

- CS326
- CS345

Course Detail-TA view

Provides the information about the course that the TA is part of:

Course name, the name of the professor who teaches the class, all the TAs from that class, current assignments for all the TAs in that class and the upcoming assignments. At the bottom of the page there is a place for the updates(didn’t fit in the screenshot)



Title: CS326

Professor: Tim Rich

Course Info

The World Wide Web was proposed originally as a collection of static documents inter-connected by hyperlinks. Today, the web has grown into a rich platform, built on a variety of protocols, standards, and programming languages, that aims to replace many of the services traditionally provided by a desktop operating system. Topics will include: producing dynamic content using a server-based language, content serving databases and XML documents, session state management, multi-tier web-based architectures, web security, and core technologies including HTTP, HTML5, CSS, JavaScript, and SQL will be emphasized. This course will also study concepts and technologies including AJAX, social networking, mashups, JavaScript libraries (e.g., jQuery), and web security.

Course TAs

Full Name	Role
Shivi Singh	TA
Salem Couja	TA

Current Assignments

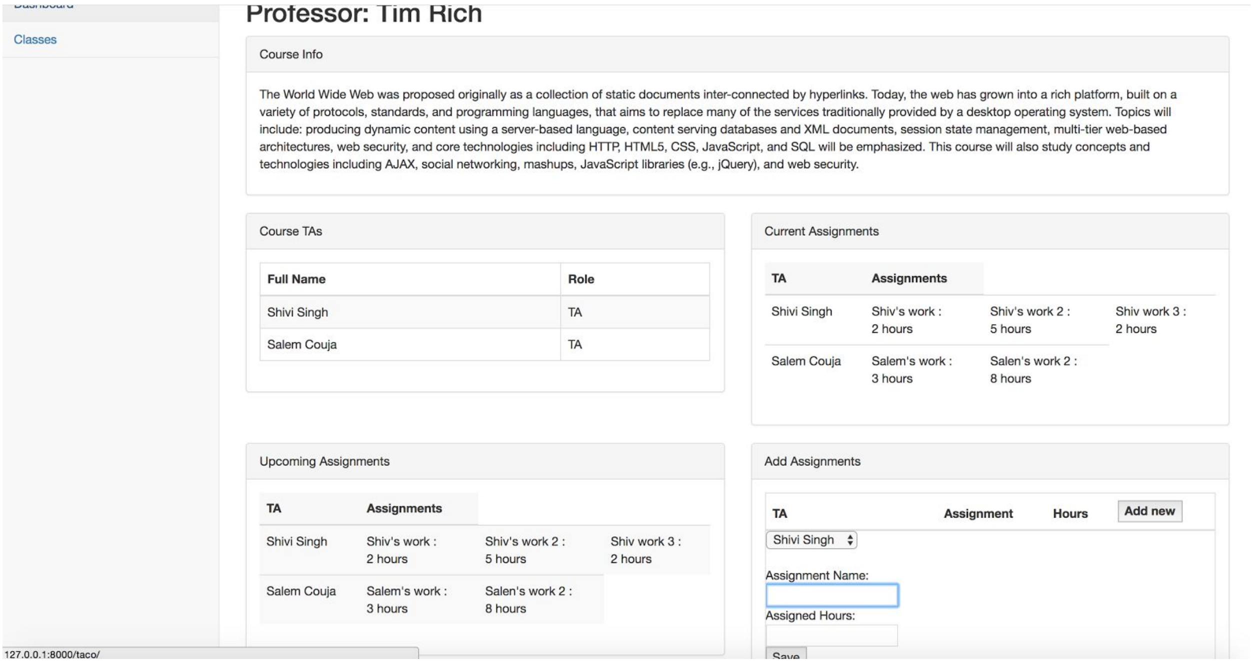
TA	Assignments		
Shivi Singh	Shiv's work : 2 hours	Shiv's work 2 : 5 hours	Shiv work 3 : 2 hours
Salem Couja	Salem's work : 3 hours	Salen's work 2 : 8 hours	

Upcoming Assignments

TA	Assignments		
Shivi Singh	Shiv's work :	Shiv's work 2 :	Shiv work 3 :

Course Detail-Professor’s view

This view is similar to the TAs course detail view, the only difference is that the professor is going to be able to add new assignments for the TAs and assign hours for those assignments



Dashboard

Classes

Professor: Tim Rich

Course Info

The World Wide Web was proposed originally as a collection of static documents inter-connected by hyperlinks. Today, the web has grown into a rich platform, built on a variety of protocols, standards, and programming languages, that aims to replace many of the services traditionally provided by a desktop operating system. Topics will include: producing dynamic content using a server-based language, content serving databases and XML documents, session state management, multi-tier web-based architectures, web security, and core technologies including HTTP, HTML5, CSS, JavaScript, and SQL will be emphasized. This course will also study concepts and technologies including AJAX, social networking, mashups, JavaScript libraries (e.g., jQuery), and web security.

Course TAs

Full Name	Role
Shivi Singh	TA
Salem Couja	TA

Current Assignments

TA	Assignments		
Shivi Singh	Shiv's work : 2 hours	Shiv's work 2 : 5 hours	Shiv work 3 : 2 hours
Salem Couja	Salem's work : 3 hours	Salen's work 2 : 8 hours	

Upcoming Assignments

TA	Assignments		
Shivi Singh	Shiv's work : 2 hours	Shiv's work 2 : 5 hours	Shiv work 3 : 2 hours
Salem Couja	Salem's work : 3 hours	Salen's work 2 : 8 hours	

Add Assignments

TA	Assignment	Hours	Add new

Shivi Singh

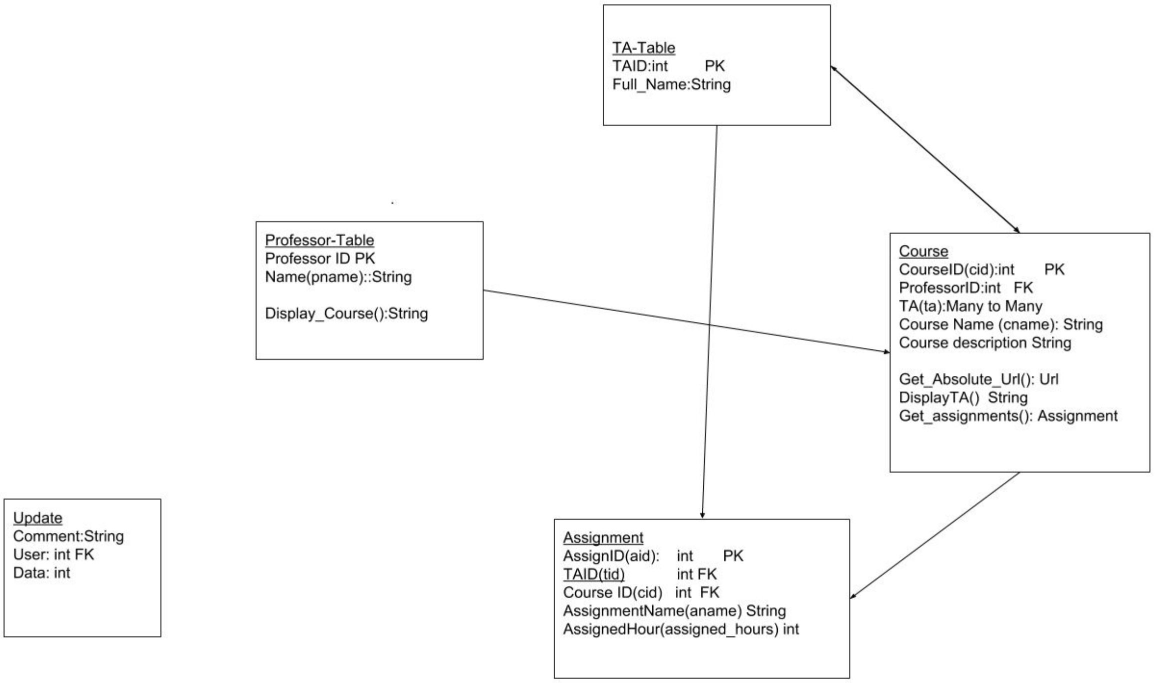
Assignment Name:

Assigned Hours:

Save

127.0.0.1:8000/taco/

Data Model



The professor table houses the professor information with a unique primary key and an accompanying name attribute. The professor table has a one to many relationship with courses since a professor can be in multiple courses. There is a function that displays the courses in admins

The Ta table has a primary for the teaching assistant and an name attribute. It has a many to many relationship with course since multiple ta's can be in multiple courses and multiple courses can have multiple ta's.

The course table has a course id for primary key, a professor ID as a foreign key as stated before, and a many to many relationship with TA as stated before. There are also name and course attributes as strings. There is a function to get the reverse url for the course detail page which is called to redirect to the course page after saving new assignments. There is a function to display the ta's associated with the course and a function that filters all assignment object that have the same course id.

The assignment table contains a primary key for Assignment and a foreign from Ta's and the course table. There is also two other attributes for the assignment name and required hours for the assignment.

The Update table is used for updates and names a foreign from user, a date field, and a comment field of type string.

URL Routes/Mappings:

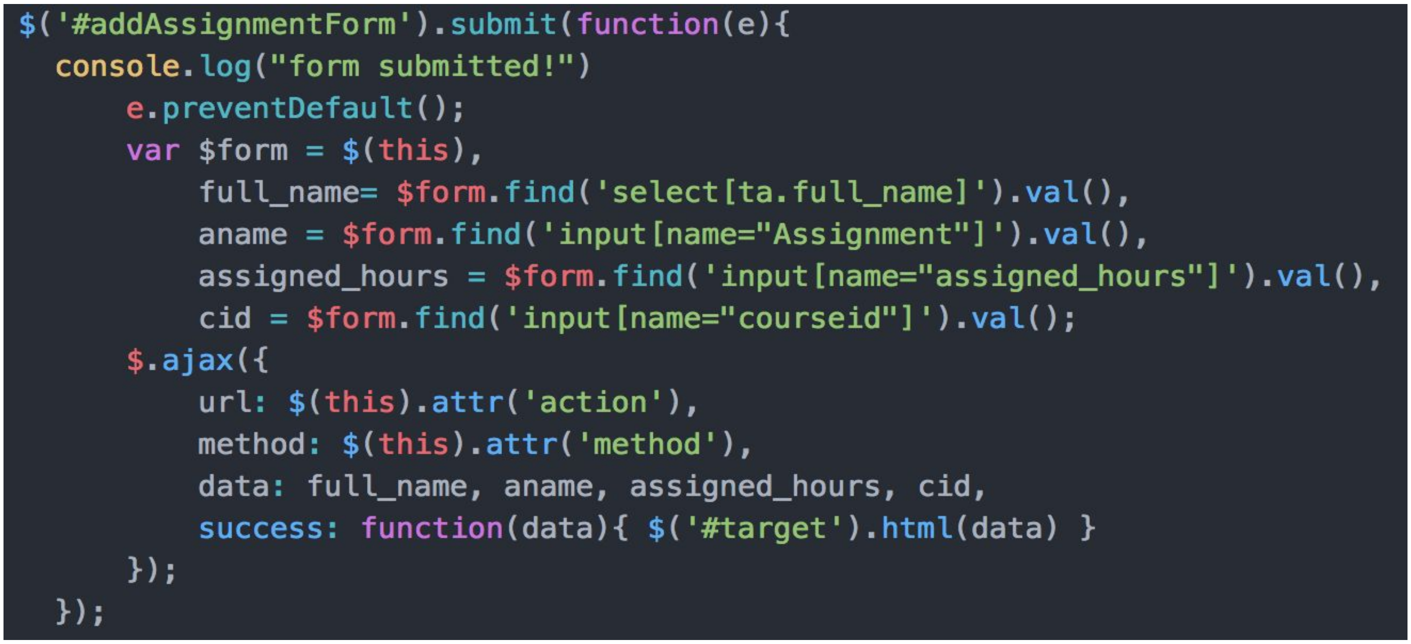
- **url(r'^$', views.index, name='index')** this URL is mapped to the home page, i.e. the Dashboard with the charts.
- **url(r'^courses/$', views.CourseListView.as_view(), name='courses') -** This URL is used to display the list of courses in the database.
- **url(r'^course/(?P<pk>[A-Za-z0-9-]+)$', views.CourseDetailView.as_view(), name='course-detail')-** This Url is mapped to give a detailed view of each class. Like the professor who is teaching the course, the course description, TA's for the course, Current Assignments and Upcoming Assignments, .
- **url(r'^update/$', views.UpdateListView.as_view(), name='update')-** This URL is used to display the latest updates that the professor creates to notify all the TA's in his/her class. The Updates are listed in the class overview page
- **url(r'^course/add/$', views.addAssignment, name='addAssignment')-** this URL is used to display the form for professors to add assignments for their TA's in a certain class

Authentication/ Authorization:

Only logged in users can see the Dashboard, Courses. The users are defined under two sets Teaching Assistants and Professors. The Professors can add Assignments, and when he/she does it is dynamically added to the upcoming Assignments for TA's. Only professors have the ability to add assignments. (We were trying to make the TA's able to edit the number of hours worked, but it was getting too messed up. Given the time constraints we decided to drop it.)

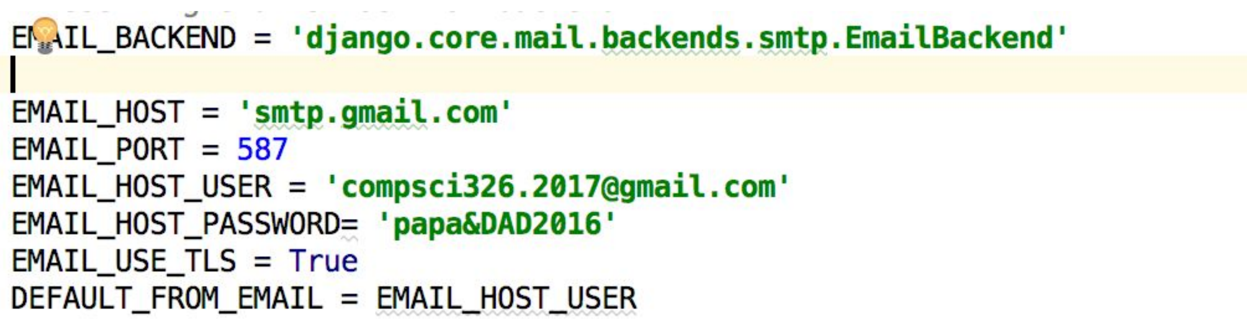
Team Choice:

For our team choice we used AJAX, jQuery and JSON to create our dynamically updating form. We had to modify our views, forms, and course detail pages to properly handle the AJAX requests and make sure form data saved to the database properly for each specific TA and course.



```
$('#addAssignmentForm').submit(function(e){
  console.log("form submitted!")
      e.preventDefault();
      var $form = $(this),
          full_name= $form.find('select[ta.full_name]').val(),
          aname = $form.find('input[name="Assignment"]').val(),
          assigned_hours = $form.find('input[name="assigned_hours"]').val(),
          cid = $form.find('input[name="courseid"]').val();
      $.ajax({
          url: $(this).attr('action'),
          method: $(this).attr('method'),
          data: full_name, aname, assigned_hours, cid,
          success: function(data){ $('#target').html(data) }
      });
  });
```

We also set the password reset link. It works, on a reset request you get an email from **compsci326.2017@gmail.com.** We didn't have to create any model for this, just added a couple of lines of code in system.py to get it functioning.



```
EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'

EMAIL_HOST = 'smtp.gmail.com'
EMAIL_PORT = 587
EMAIL_HOST_USER = 'compsci326.2017@gmail.com'
EMAIL_HOST_PASSWORD= 'papa&DAD2016'
EMAIL_USE_TLS = True
DEFAULT_FROM_EMAIL = EMAIL_HOST_USER
```

Conclusion:

Firstly, our team lacked communication and it was really hard to find times outside class that worked for everyone. This made the project challenging at first, but by the end of the semester we picked up pace and made an effort to notify all the members of the project's progress. Personally, we learnt a lot about working in teams with people who are really different from you. Which was one of the main things we learnt in this class. Also, we had to change our models various times as in the beginning we were not aware of how complicated things can get when we are trying to access data that is in another model. If you think about in SQL querying a tuple in another table given its Foreign Key was hard, but we got a way around it. If we were to start from scratch again we would have definitely designed our models differently. Setting up the forms was a hassle. Instead of making a basic form where a user clicks a button and is redirected to a page with a form to fill out, we decided to create a dynamic form. We used ajax, jQuery and json to accomplish this in a way that would allow a professor to add new assignments for specific TA's without the browser having to constantly refresh for each new assignment.

Part 4: Single Slide

Link to the Slide:
https://docs.google.com/presentation/d/1sogTrsNa74Yytib7qSEkVP0n7HpST5ssQrjF0z1E0C4/edit?usp=sharing