Teamworx Product and User Documentation
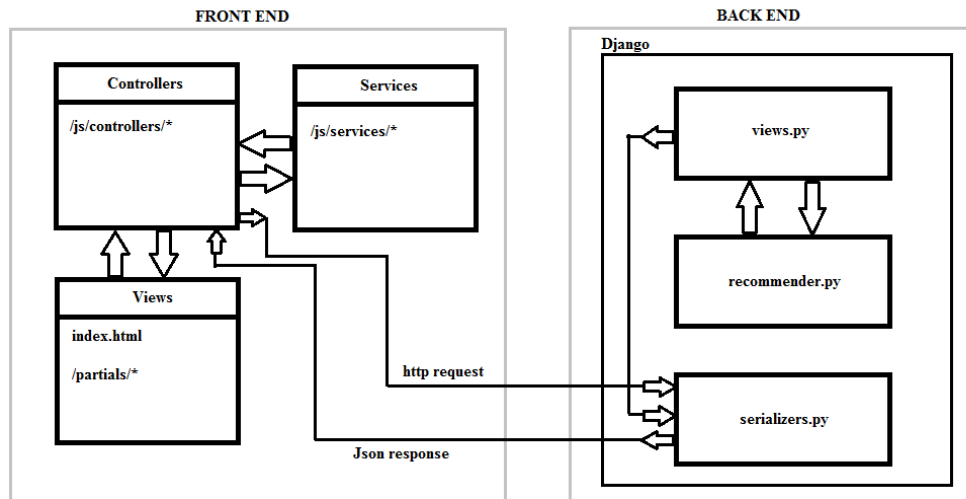
Jerry Yee
Thang Nguyen
John Murphy

Architecture

Teamworx is a standard client-server architecture. The front-end and back-end are in separate repositories. A diagram is shown below:
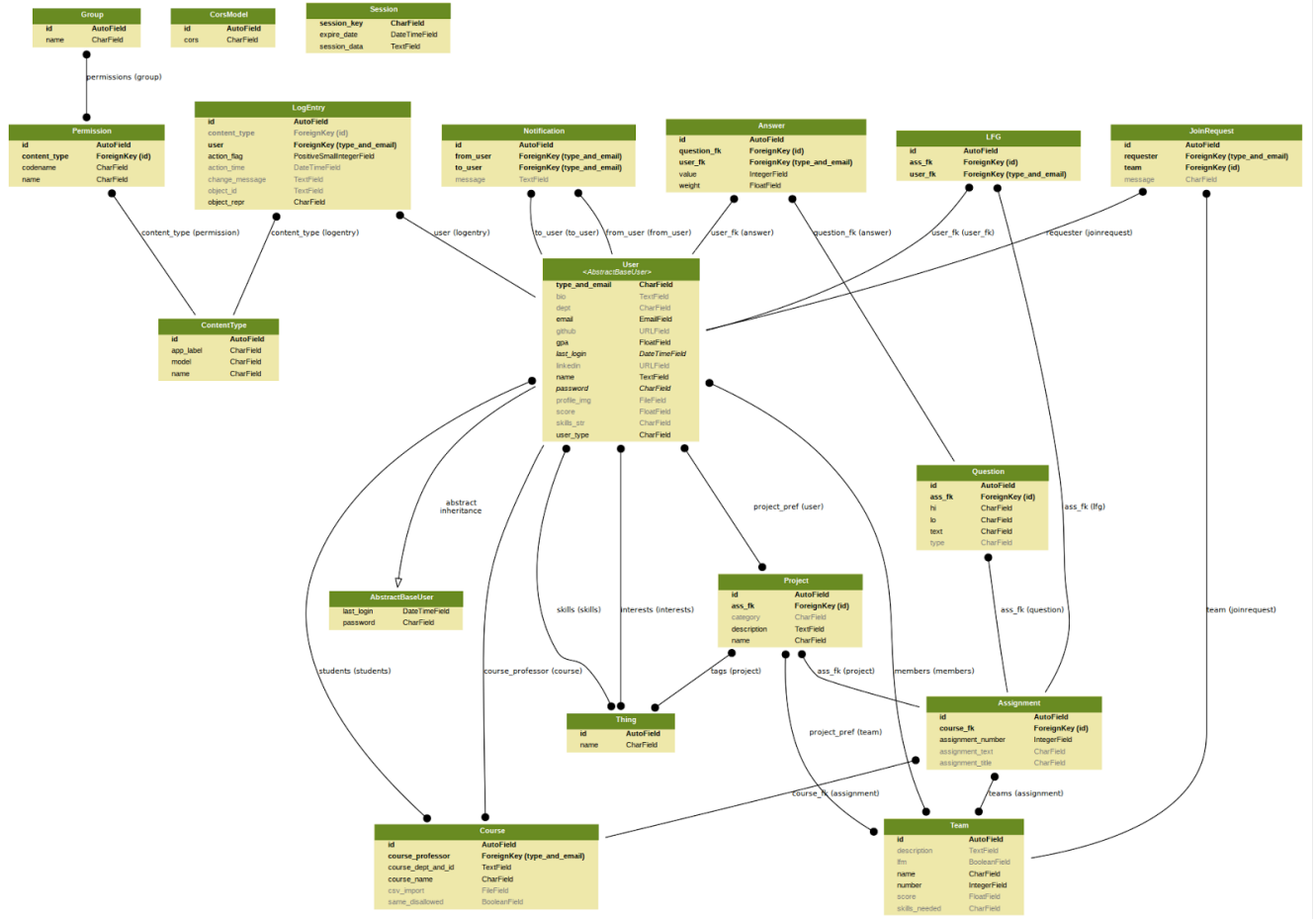

Detailed Design

All of the client-side HTML pages/templates are located under the partials directory. The teamworx WebApp directory in the teamfinder directory contains the settings.py and urls.py Python files. The settings.py file contains important settings for the Django sever such as what modules to load, whether debug mode is active, where static files are located, how to connect to the database, etc. The urls.py file contains the logic for the more basic URL patterns such as login page, register page, course page, team page, etc.

The bulk of the code is located under the teamwork-REST directory. The static files are located here under static folder. The important files are views.py, urls.py, and models.py. The models.py file defines all the database schema objects and also set up the admin CRUD views for them. The urls.py file has all the URLs that users could POST/GET/PUT/DELETE to. The views.py file has all of the methods that urls.py file connects to. The majority of the logic for the system is located in this file. Each of the methods here that correspond to GET/POST/PUT/DELETE requests will take in data representing this request and return data in the form of a JSON response.

Data Storage

Group
id	AutoField
name	CharField

CorsModel
id	AutoField
cors	CharField

Session
session_key	CharField
expire_date	DateTimeField
session_data	TextField

LogEntry
id	AutoField
content_type	ForeignKey (id)
user	ForeignKey (type_and_email)
action_flag	PositiveSmallIntegerField
action_time	DateTimeField
change_message	TextField
object_id	TextField
object_repr	CharField

Permission
id	AutoField
content_type	ForeignKey (id)
codename	CharField
name	CharField

Notification
id	AutoField
from_user	ForeignKey (type_and_email)
to_user	ForeignKey (type_and_email)
message	TextField

Answer
id	AutoField
question_fk	ForeignKey (id)
user_fk	ForeignKey (type_and_email)
value	IntegerField
weight	FloatField

LFG
id	AutoField
ass_fk	ForeignKey (id)
user_fk	ForeignKey (type_and_email)

JoinRequest
id	AutoField
requester	ForeignKey (type_and_email)
team	ForeignKey (id)
message	CharField

ContentType
id	AutoField
app_label	CharField
model	CharField
name	CharField

User
<AbstractBaseUser>
type_and_email	CharField
bio	TextField
dept	CharField
email	EmailField
github	URLField
gpa	FloatField
last_login	DateTimeField
linkedin	URLField
name	TextField
password	CharField
profile_img	FileField
score	FloatField
skills_str	CharField
user_type	CharField

Question
id	AutoField
ass_fk	ForeignKey (id)
hi	CharField
lo	CharField
text	CharField
type	CharField

AbstractBaseUser
last_login	DateTimeField
password	CharField

Project
id	AutoField
ass_fk	ForeignKey (id)
category	CharField
description	TextField
name	CharField

Assignment
id	AutoField
course_fk	ForeignKey (id)
assignment_number	IntegerField
assignment_text	CharField
assignment_title	CharField

Thing
id	AutoField
name	CharField

Course
id	AutoField
course_professor	ForeignKey (type_and_email)
course_dept_and_id	TextField
course_name	CharField
csv_import	FileField
same_disallowed	BooleanField

Team
id	AutoField
description	TextField
ffm	BooleanField
name	CharField
number	IntegerField
score	FloatField
skills_needed	CharField

Relationships:
- permissions (group)
- content_type (permission)
- content_type (logentry)
- user (logentry)
- to_user (to_user)
- from_user (from_user)
- user_fk (answer)
- question_fk (answer)
- user_fk (user_fk)
- requester (joinrequest)
- abstract inheritance
- project_pref (user)
- ass_fk (lfg)
- students (students)
- course_professor (course)
- skills (skills)
- interests (interests)
- tags (project)
- ass_fk (project)
- members (members)
- ass_fk (question)
- project_pref (team)
- course_fk (assignment)
- teams (assignment)
- team (joinrequest)

User Interface

Our site consists mainly of three pages, in the form Login --> Portal --> Main, with the user being able to access the pages in that order at first, and then switch between them. There is also the option for Registration from the Login page, which will then take the user back to the Login page.

Login

The Login page allows users to enter a predefined email and password, and then select whether they are logging in as a student or an instructor. There is also the option to register, for new users, in which the aforementioned information can be filled out.

Portal

The Portal page, otherwise known as the profile page, contains information that is specific to the user. In the case of an instructor profile, the user has the ability to edit their name, contact information, and biography in the Profile tab. In the Courses tab, users can access any courses that they have created in the past, delete courses by dragging them over the trashcan icon, or create new courses by filling out the course department, number, and name. In the case of a student profile, the user is additionally able to edit their skills in the Profile tab, as well as their Linkedin and Github links. In the Courses tab, student users can access any courses that they have been added to by their instructors.
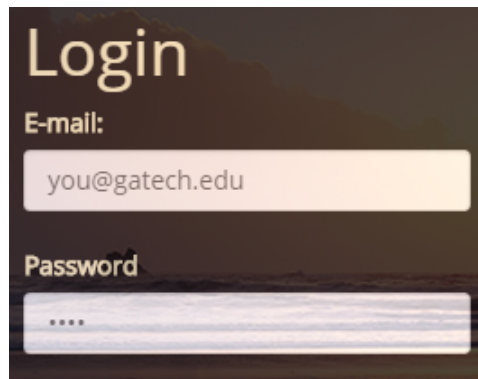
Main

The Main page is accessed by clicking on a course link in the Courses tab of the Portal page. The Main page contains an area for a student roster for the class, as well as an area for the teams associated with a particular assignment. Clicking on a student link will open that student's profile, and clicking on a team link will open that team's profile. Additionally, depending on whether the user is an instructor or a student, they can either assign questions for each user or answer them, as well as create new teams and assignments.

User Documentation
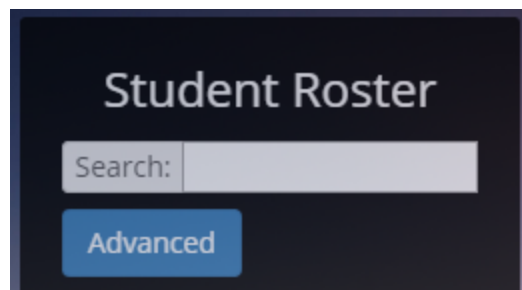
Instructor POV

From an instructor point of view, the user should begin at the login page and enter their information in order to access the site, or register to create their login information.


After entering their information and selecting "Instructor," the user is taken to their profile page, where they can create and edit their personal information, as well as create and access their course list. After selecting a course, the user will be taken to the main page, where they first must upload a student roster in the form of a CSV file. This can be done in the Student Roster section of the page.

Once the roster is uploaded, instructors have the ability to create teams manually and then individually add members by dragging, or randomly assign all students to randomly created teams. This can all be done in the Team section. Additionally, instructors can access student and team profiles simply by clicking on them. Clicking the "All Teams" tab will restore the page to its initial location. Instructors can also create new assignments in the navigation bar at the top, as well as new questions for students to answer in order to better find team members.
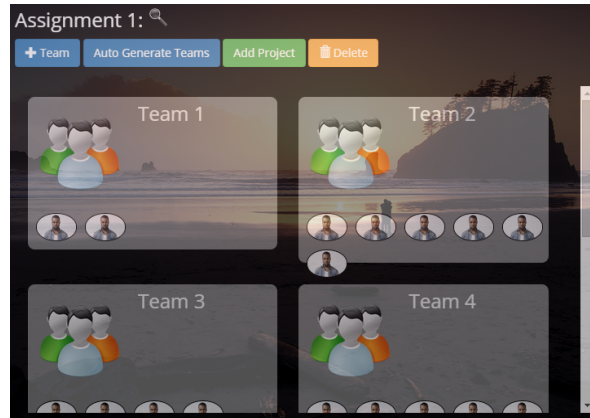
Student POV

From the student point of view, the login and registration functionality is the same as that of the instructor, except for of course choosing the "Student" option when logging in. The student has a few extra editable spaces on the Profile page, mainly their skills, Linkedin link, and Github link. These can be left blank, but it is easier for students to find compatible team members if their profile is complete.
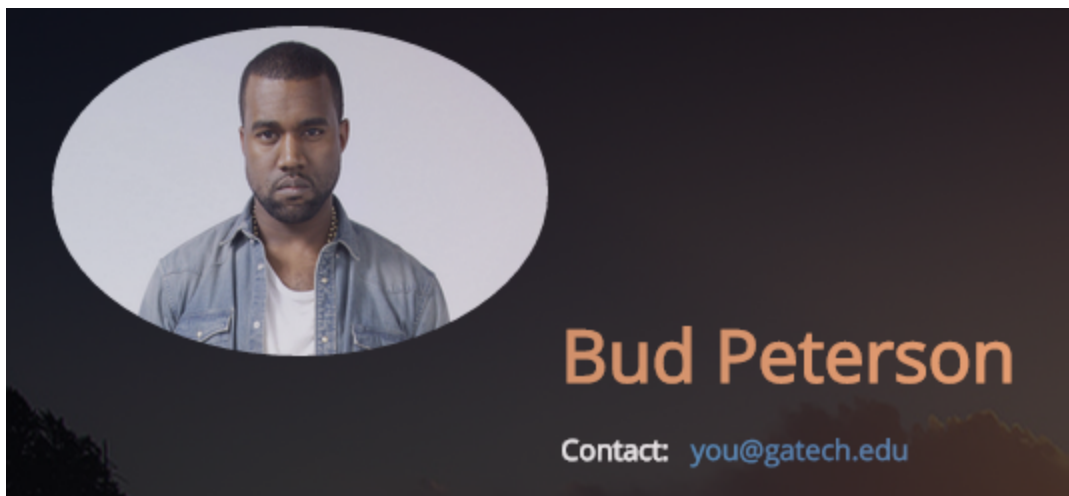
In the Courses tab of the Profile page, students can access any courses that they have been added to by their instructors. Students can access a course's main page by clicking on the course title. On the main page, students can create a team for a particular assignment in the Team section, as long as they are not already joined to a team by their instructor. Students can then access their team's profile under the My Team tab. Additionally, students will be able to view any other students who are on their team, and owners of the team can edit the team information, including the particular skills that their team is looking for in new members. Additional student profiles can be viewed from the Student Roster section, the roster can be filtered using the Search and "Advanced" functions.

For those students who do not wish to create their own team, there is the option to join another. This can be done first by answering the instructor's questions in the Course tab of the navigation bar. Once this survey is submitted, students can navigate to the Team Recommendation page under the Find Match tab, and choose from a list of already created teams that fit their skillset. Once they choose a team, they can simply click the "Send Join Request" button, and a request will be sent to the owner of that team. The owner can then view the request in the Pending Requests tab of their team page, and can choose to either accept or deny the student. The student can then find out if they were accepted or denied by checking their own Notification tab, located under the Course tab in the navigation bar.