User Manual

Student Performance Tracker

A course visualization, organization, and assessment system

Spring 2020

studentperformancetracker.xyz

Table of Contents

Section 1: Introduction

Purpose

The Student Performance Tracker is a pedagogical tool designed to help professors and students, centered around a curriculum which is designed like a directed, acyclical graph of relevant course topics, giving students the ability to see the ‘map’ of course topics throughout the curriculum and allowing them to progress through their course with more holistic knowledge concerning the class.

This program has been configured to present information to students and to handle information from professors pertaining to a course whose topics can easily be represented as a directed acyclical graph. Additionally, it has many modular characteristics, allowing different professors with differently-structured courses to customize the interface through which students interact with the course, and the mechanics put in place to administer a student’s grade, including an array of assignments, weights for certain topics, and in-site quizzes.

This manual is designed as a tool for users of this program, pointing out and explaining the means by which a student can participate in a course, a professor can regulate and design a course, and an administrator can maintain this software for themselves.

Section 2: The Professor Guide

Professor- Django Admin

The APIs for this web app are not completely covered by the front end. As such, there are some features and data that must be interacted with via the Django Admin. In order to get access to the Django Admin, you must first create a super user.

Open the folder containing this application's code. Then, navigate to the folder named 'src'. Then, open a terminal from that folder. Run the command '*docker-compose up*'. Leave this terminal running.

Then, open another terminal window. Run the command '*docker exec -it backend bash*'. This will change your terminal view, but you should still be able to run commands. Run the command '*python manage.py createsuperuser*'. This will begin setting up a super user; please follow the instructions in the terminal. Once finished, close both terminal windows.

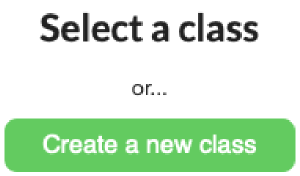
Once you complete this step navigate to {domain}*/admin* (e.g. https://studentperformancetracker.xyz/admin), and login with the credentials you just created. In development mode, the url will look like: localhost:8000/admin or spt-acas.com:8000/admin. From here, you can access every model in the application, and create, read, update, and delete everything as needed.

Note: Be very careful when adding and deleting data, as some of the interaction on the sites depends on the assumption that relationships between certain instances of certain models exist. For instance, a Student to Topic relationship also requires a Student to Course relationship so that the grade can cascade upwards.

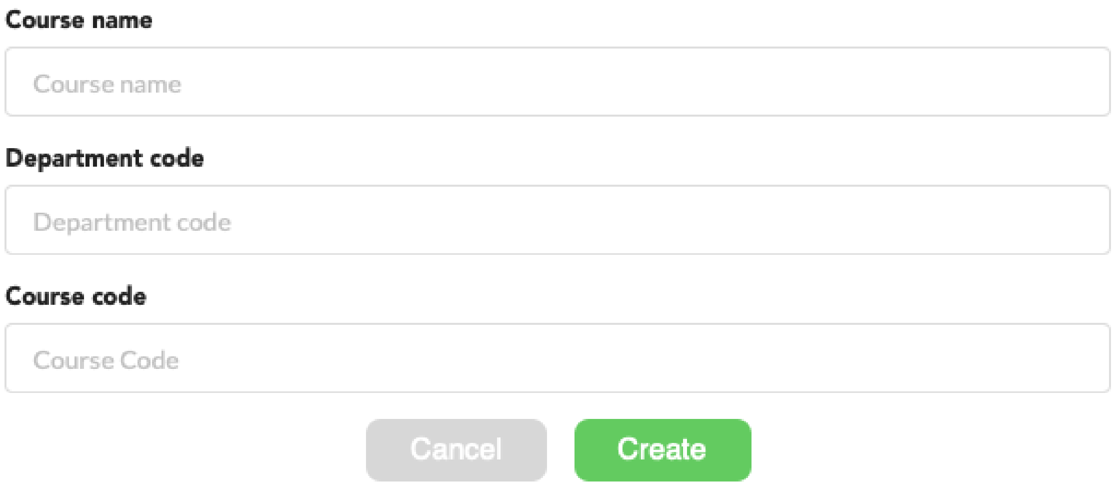
To create a professor account, navigate to the desired account and set is_professor to true. You may set a superuser account to be a professor so long as the email matches an email that the professor is able to sign into google with. Optionally, you may create an account from the frontend and then set that account to be a professor. Next, navigate to the Course in the django admin page and associate the professor account with the course.

Professor- Creating a Class

When signed in as a Professor, you will be greeted by a prompt to either select a preexisting class or to create a new class.



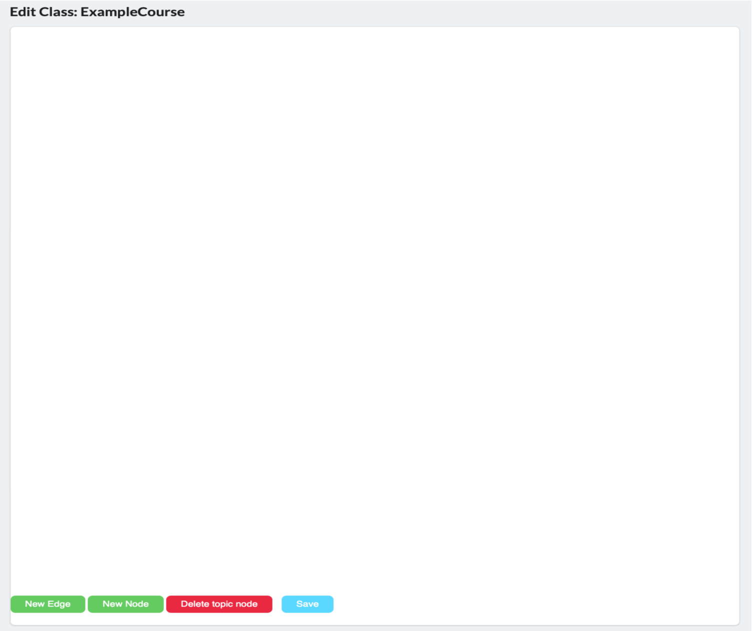
When you select “Create a new class,” you will be prompted to enter information about that class.



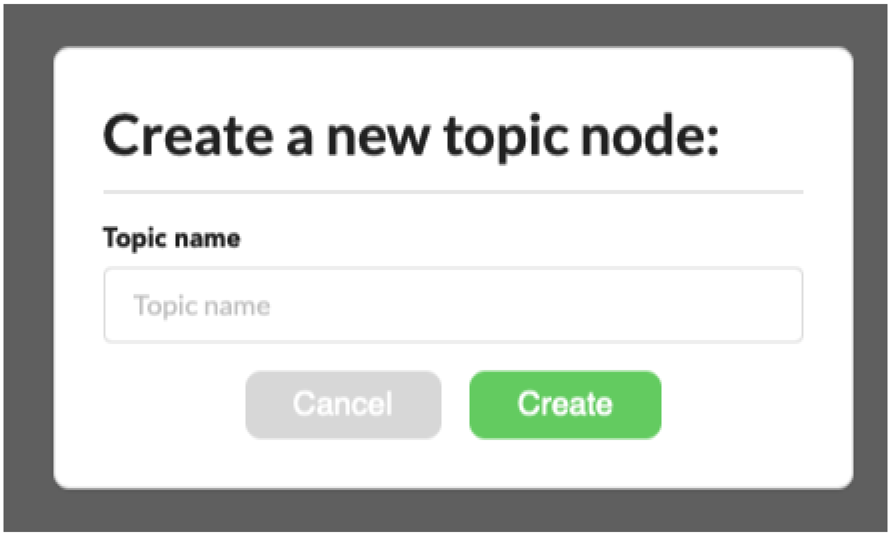
Pressing create will create the class.

Professor- Creating and Editing the Class Graph

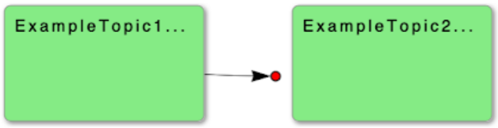
Immediately after creating a new class, or by pressing the “Edit graph” button on a class’s main page you will be presented with the class graph.

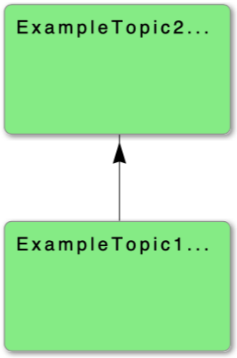


You can create a new topic by pressing the “New Node” button. The following prompt will appear.



After creating the desired topic(s), you can press the “New Edge” button to go into edge creation mode. Press and hold one topic and then drag to another topic to connect them, with the first topic being a prerequisite for the latter topic.

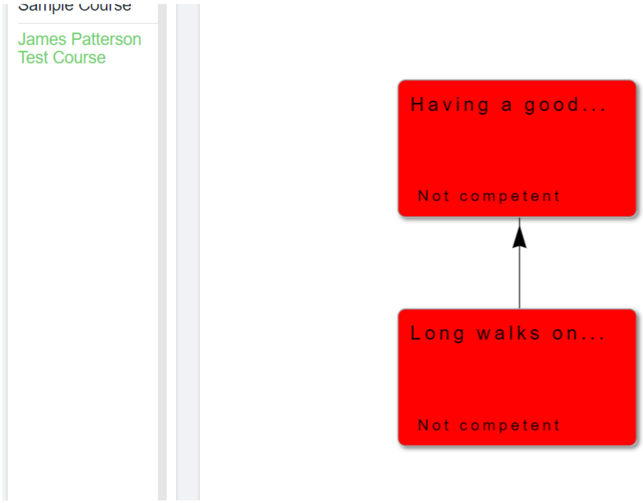




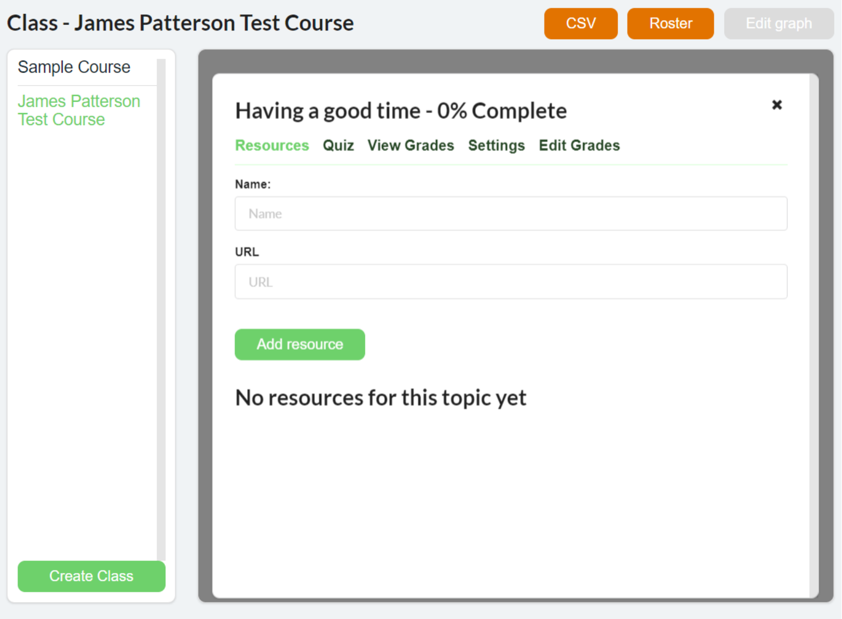
Pressing the "Delete topic node" puts you into node deletion mode. Clicking the desired topic and then confirming will delete a node and all edges attached to it.
To leave the Edit Class view, press Save.

Professor- Creating Resources

Adding resources to topics within a course can add a convenient means by which students can learn more about the topics in which they're interested, or the concepts with which they're struggling. To add a resource to a topic, navigate to a course for which you are a professor.

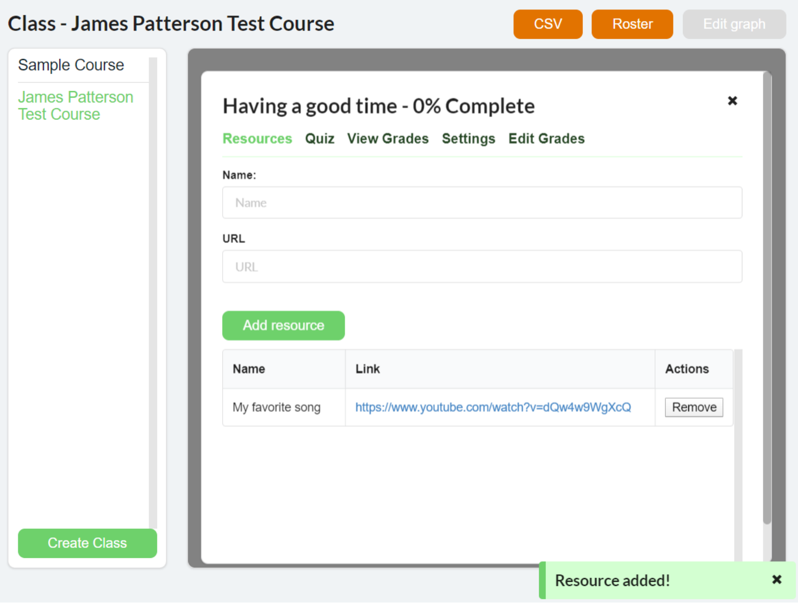


Once on this page, select the topic for which a new resource is required. On the menu which appears, select the “Resources” tab.



No resources for this topic yet

To add a resource, simply give it a name, and provide the URL by which that resource may be accessed. Then, click the “Add resource” button to make it available to your students.



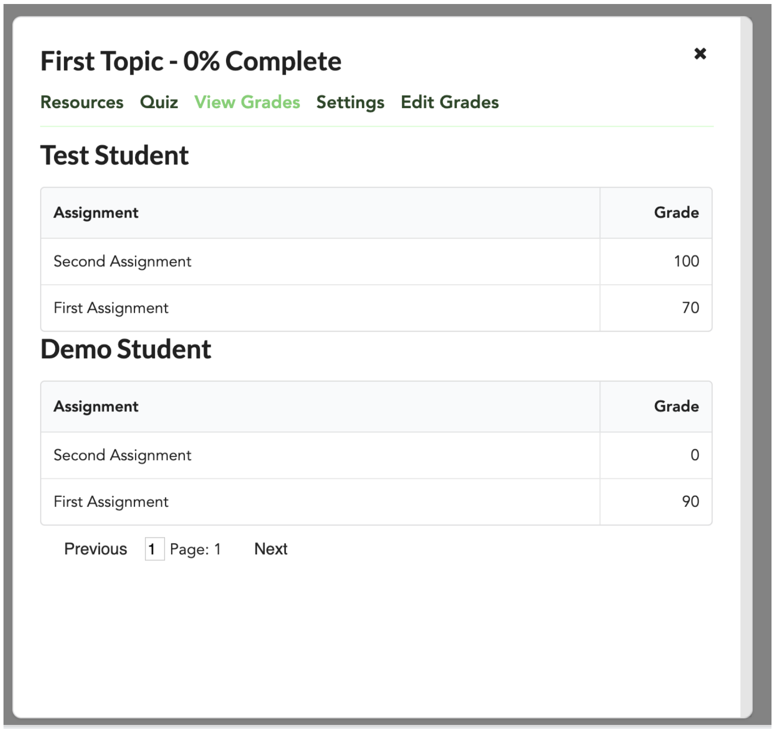
Name	Link	Actions
My favorite song	https://www.youtube.com/watch?v=dQw4w9WgXcQ	Remove

Resource added!

When a resource has been successfully added, you should be able to see it listed on the ‘Resources’ page, and may remove it by pressing the ‘Remove’ button.

Professor- Viewing Grades

In the same menu, select the “View Grades” tab to view a list of grades for this topic. Grades are organized by student, and then by assignment. If there are multiple assignments for this topic, a separate grade will be shown for each.



First Topic - 0% Complete

Test Student

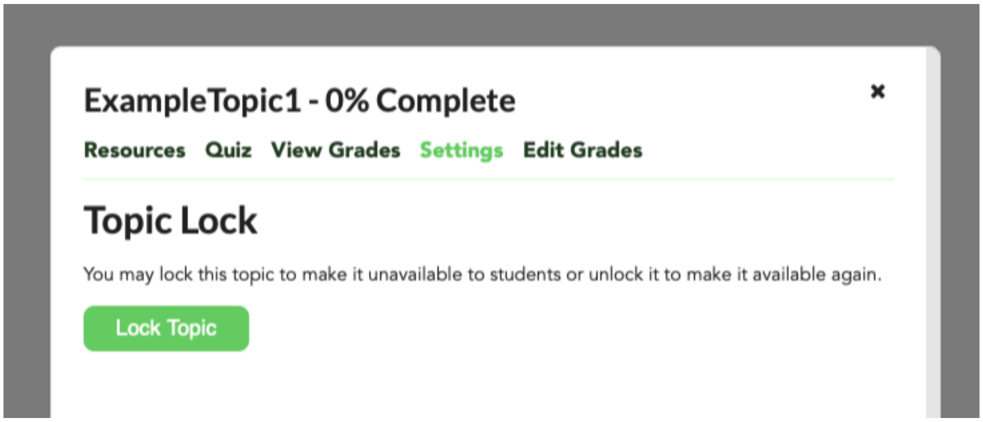
Assignment	Grade
Second Assignment	100
First Assignment	70

Demo Student

Assignment	Grade
Second Assignment	0
First Assignment	90

Professor- Topic Locks

Locking a topic prevents students from seeing a topic until it has been unlocked for the class. To lock a topic, click on the topic from the class’s main page and navigate to the “Settings” tab.

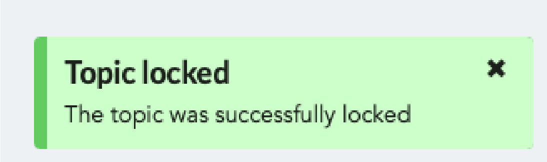


ExampleTopic1 - 0% Complete

Topic Lock

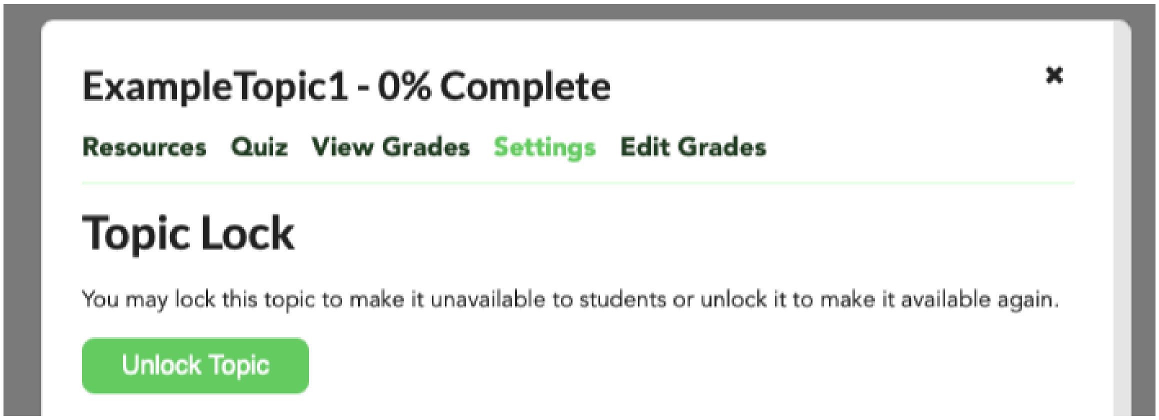
You may lock this topic to make it unavailable to students or unlock it to make it available again.

Pressing “Lock Topic” will lock the topic, and you will receive a confirmation notification. The button will change to an Unlock button, which you can use to unlock the topic.



Topic locked

The topic was successfully locked

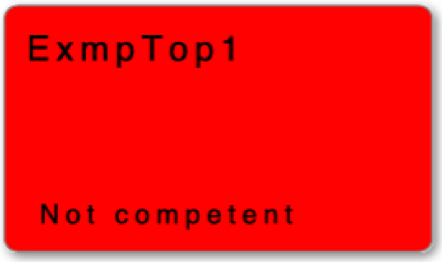


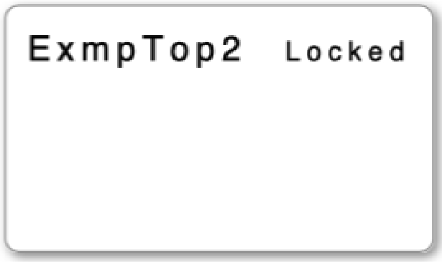
ExampleTopic1 - 0% Complete

Topic Lock

You may lock this topic to make it unavailable to students or unlock it to make it available again.

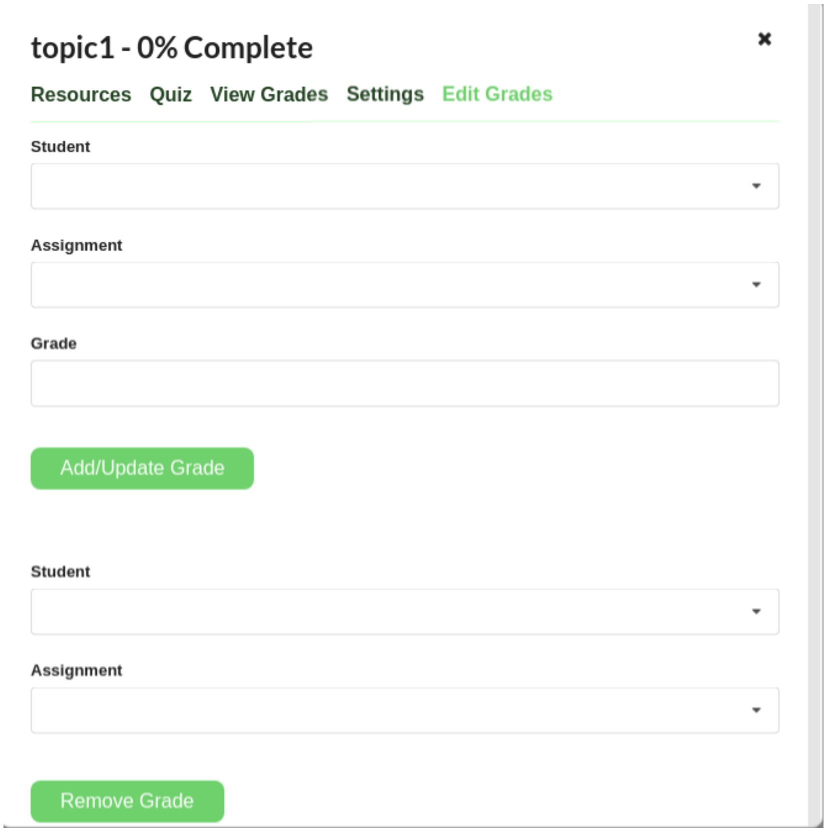
An unlocked topic’s appearance will change based on the competency of the user (usually red for a professor or TA as they will have no grade). A locked topic will appear white for a professor or TA and will not appear for a student.



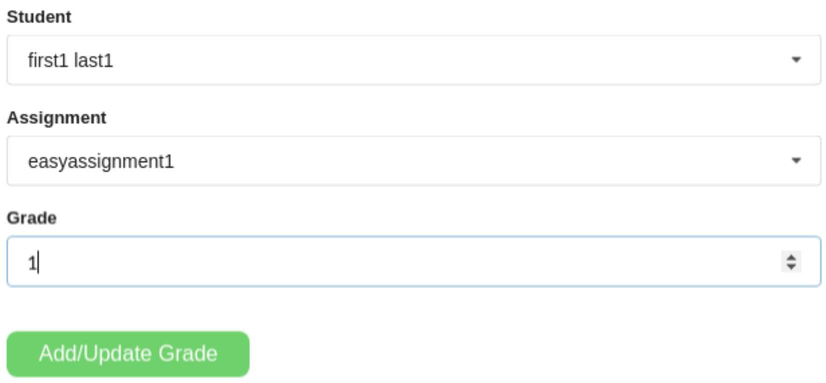


Professor- Modifying Grades

With this system, grades are based on the assignments within a topic. As a professor, there is the option to manually add, update, or remove a student's grade for a specific assignment. To do this, select the topic with the assignment in question, identical to how you would view the grades for that topic. In the menu that pops up, select "Edit Grades." The window should look like this:



To add or update a grade, you first need a student and an assignment for which to edit the grade for. Select the "Student" menu and begin typing the name of the student, you should see a drop down menu appear, from there select the student you wish to change the grade for. Next, do the same thing but for the assignment in question under the "Assignment" menu. With both fields set, you can type in the desired numerical grade for the student and that assignment. Finally, click the "Add/Update Grade" button.


Student

first1 last1

Assignment

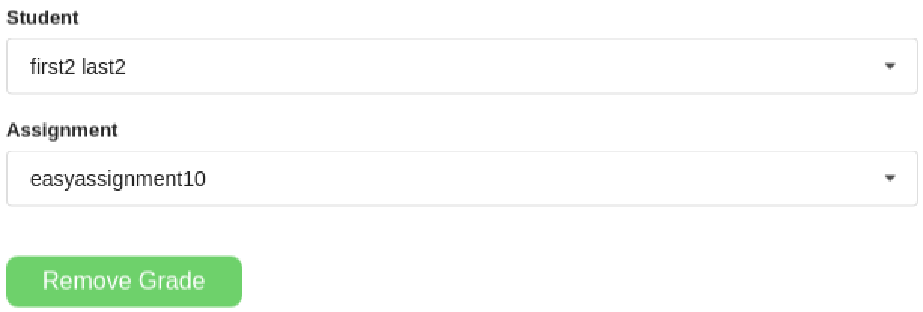
easyassignment1

Grade

1

When successful, you should see a green popup in the bottom right corner of the site, informing you of the success. When you then view the grades for that student, you should see the updated grade information.

To delete a grade, the process is similar. The process requires a student and an assignment for which to remove a grade. Just like above, begin typing the name of the student and the assignment in the respective bottom two fields under the “Add/Update Grade” button. Then click “Remove Grade.”


Student

first2 last2

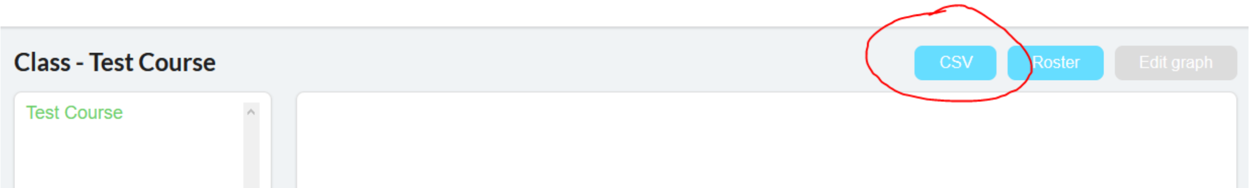
Assignment

easyassignment10

A green notification will appear in the bottom right of the site, just like with adding/updating a grade, informing you of the successful removal. When you check the student’s grades, you should see that there is no grade listed for the assignment, since the grade was deleted. The other students should still have their grades for that assignment.

Professor- CSV Uploads

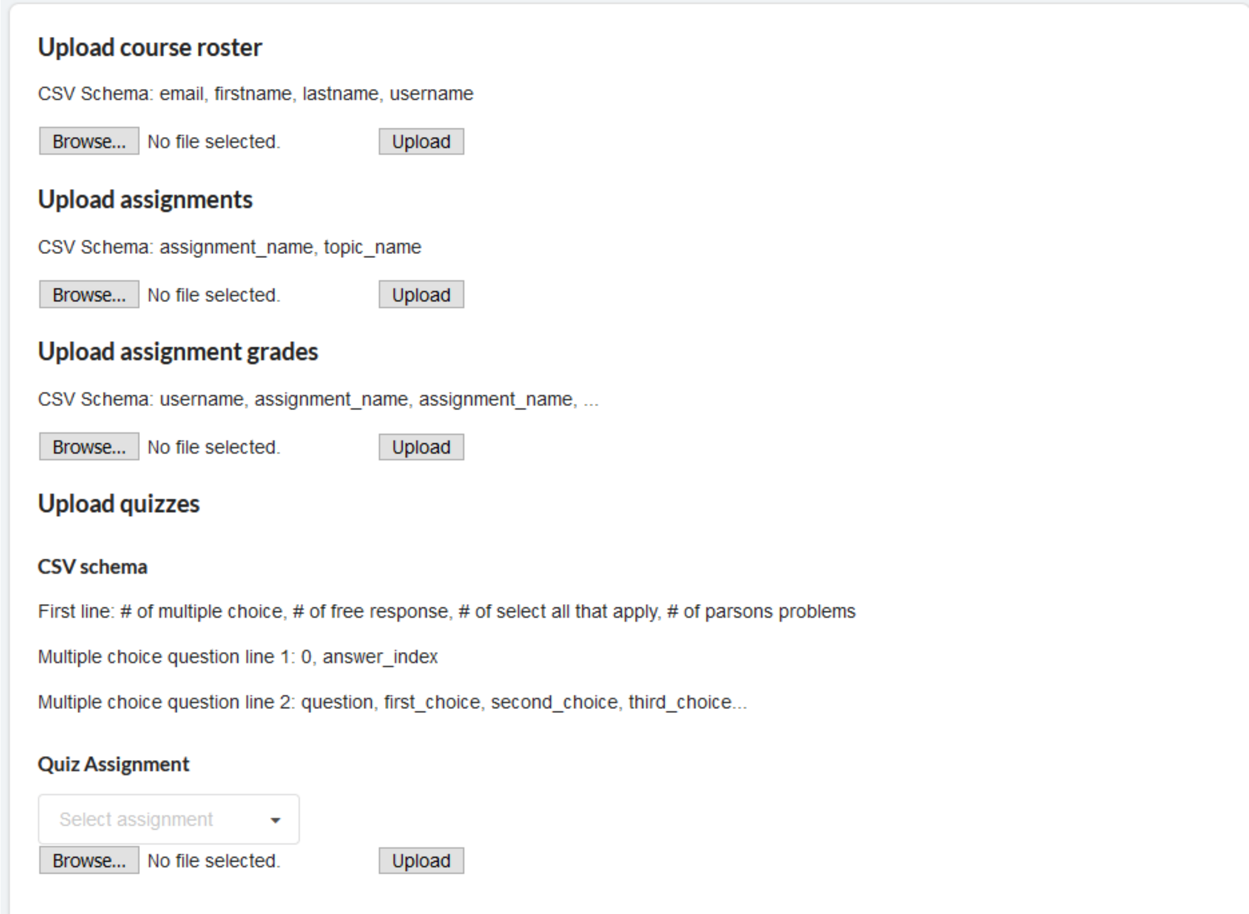
As a professor, you can manage your class roster, assignments, quizzes, and grades through uploading CSV files. To get to the CSV upload page, simply click on the class you want to edit on the home page, and look for the CSV button in the upper right hand corner.



Class - Test Course

Test Course

The CSV upload page will look like this.



Upload course roster

CSV Schema: email, firstname, lastname, username

Browse... No file selected. Upload

Upload assignments

CSV Schema: assignment_name, topic_name

Browse... No file selected. Upload

Upload assignment grades

CSV Schema: username, assignment_name, assignment_name, ...

Browse... No file selected. Upload

Upload quizzes

CSV schema

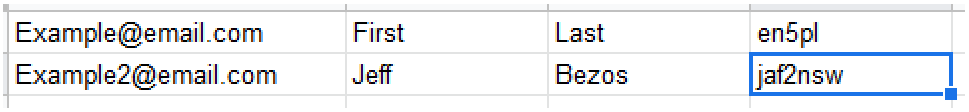
First line: # of multiple choice, # of free response, # of select all that apply, # of parsons problems

Multiple choice question line 1: 0, answer_index

Multiple choice question line 2: question, first_choice, second_choice, third_choice...

Quiz Assignment

--Course Roster Uploads--
CSV Schema: Email, first name, last name, username



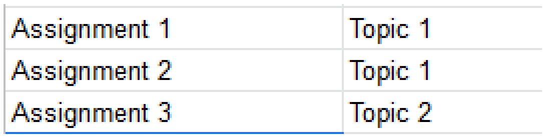
Example@email.com	First	Last	en5pl
Example2@email.com	Jeff	Bezos	jaf2nsw

To upload a course roster to your class, simply select your CSV file of students, and click ‘Upload’. Upon successful upload, you will receive a green notification in the bottom right corner of your screen. You will be notified of any students that failed to be added with red notifications.

--IMPORTANT--

Uploading the CSV will REPLACE the current roster. Differences between the previous roster and this CSV will be handled accordingly. New students will be added, old students will be dropped.

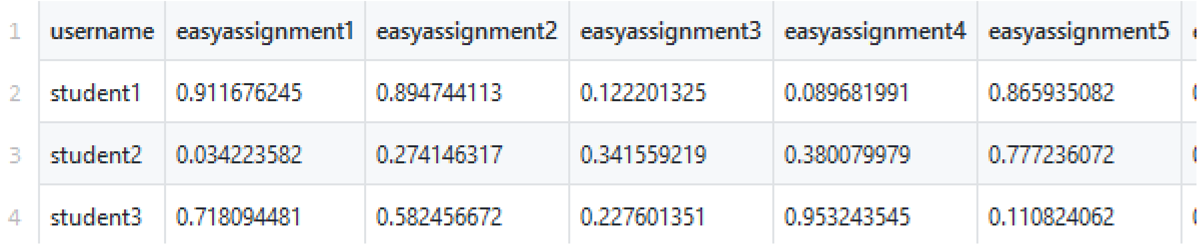
--Assignment Uploads--
CSV Schema: Assignment name, topic name



Assignment 1	Topic 1
Assignment 2	Topic 1
Assignment 3	Topic 2

This CSV upload feature allows you to quickly upload assignments for your class and attaches them to the specified topic. If the topic doesn't exist, you will receive a red notification error for that specific assignment, otherwise it will be a green success message.

--Assignment Grades Upload--
CSV Schema: Header row: anything, assignment name, assignment name, …
Subsequent rows: username, grade, grade, ...



username	easyassignment1	easyassignment2	easyassignment3	easyassignment4	easyassignment5
student1	0.911676245	0.894744113	0.122201325	0.089681991	0.865935082
student2	0.034223582	0.274146317	0.341559219	0.380079979	0.777236072
student3	0.718094481	0.582456672	0.227601351	0.953243545	0.110824062

This CSV upload allows you to quickly give students grades for previously created assignments. If the student or the assignment doesn't exist within the system, a red error notification for that grade will alert you, otherwise it will be a green success message.

-- Quiz Upload--
CSV Schema: First line: # of multiple choice, # of free response, # of select all that apply, # of parsons problems

Subsequent lines are grouped into 2.
First line: Type of question, index of answer(s)
Second line: Question, choice1, choice2, …

Parson's problems format:

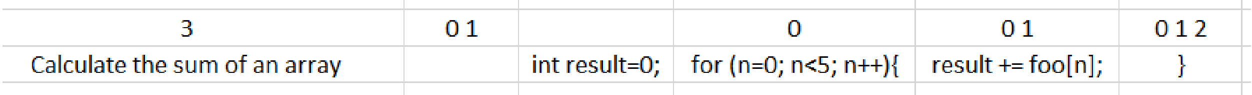
First line: type of question, fixed lines, dependencies 1, dependencies 2, ... , dependencies n
Second line: question,blank, line 1, line 2, …, line n

- Fixed lines represent the lines of the answer that should be in a fixed position. These values are space separated
- List of dependencies is a list of the lines that NEED to go before the current line in order for the answer to be correct. These values are space separated.
- The “blank” should be left empty. We did this so that the line of code lines up with the dependencies if you are using a spreadsheet application to edit the csv such as excel.

Example:

```
1,0,1,1
0,1
The first index of an array in C++ is:,-1,0,1,2
2,0,2,3
Select all valid array declarations,int foo[5],bool[4] foo,int foo [1] = {16},int array[]
3,0 1,,0,0 1,0 1 2
Calculate the sum of an array,,int result=0;,for (n=0; n<5; n++){,result += foo[n];,}
```

Example of parsons in excel:



3	0 1		0	0 1	0 1 2
Calculate the sum of an array		int result=0;	for (n=0; n<5; n++){	result += foo[n];	}

Question types are as follows:
0: multiple choice
1: free response
2: select all that apply
3: parson’s problems

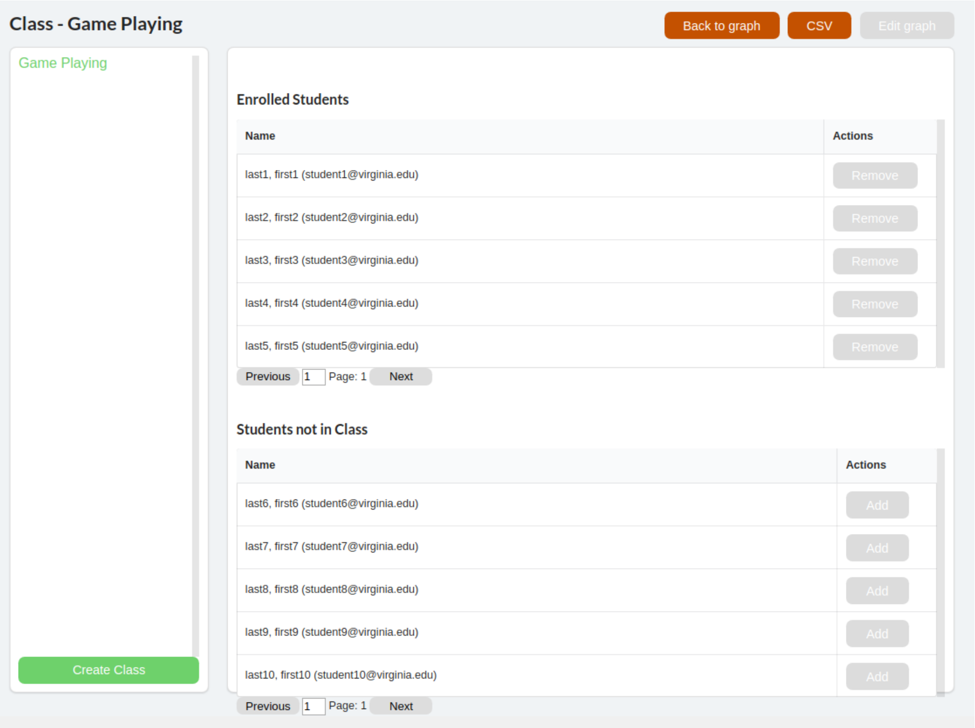
This CSV upload feature allows you to upload a set of quiz questions to the selected assignment category. Upon successful creation, a green notification will return, otherwise a red notification will return.

--IMPORTANT--

This feature will replace the set of quiz questions currently attached to the assignment, make sure you upload all questions that are intended to be part of the question pool.

Professor- Course Roster

In addition to bulk uploads by CSV, there is also the option to manually add students to a class who are already present in the system. Next to the “CSV” and “Edit Graph” tabs, there is a “Roster” tab. Navigating to this page will display two tables, one called “Enrolled Students” and the other called “Students not in Class.”



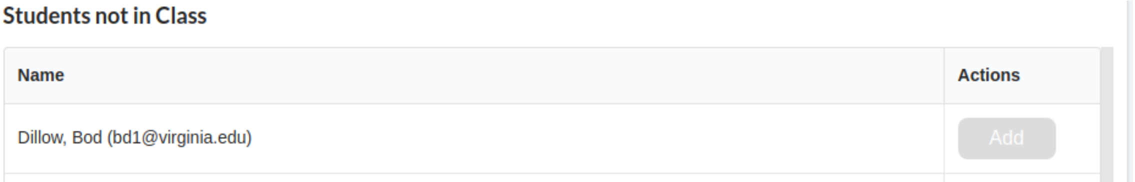
Enrolled Students

Name	Actions
last1, first1 (student1@virginia.edu)	Remove
last2, first2 (student2@virginia.edu)	Remove
last3, first3 (student3@virginia.edu)	Remove
last4, first4 (student4@virginia.edu)	Remove
last5, first5 (student5@virginia.edu)	Remove

Students not in Class

Name	Actions
last6, first6 (student6@virginia.edu)	Add
last7, first7 (student7@virginia.edu)	Add
last8, first8 (student8@virginia.edu)	Add
last9, first9 (student9@virginia.edu)	Add
last10, first10 (student10@virginia.edu)	Add

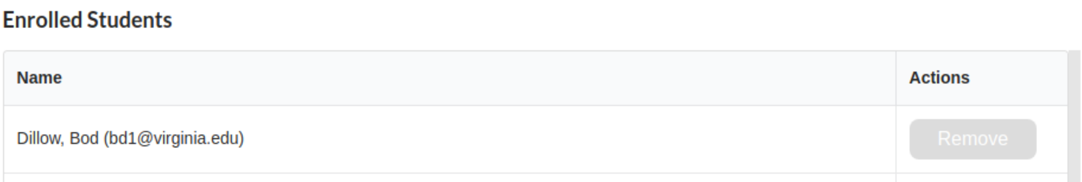
These tables, as their names suggest, contain all the students in the system; the first table contains the students who are currently enrolled in the class. The second table contains all students in the system who are currently not enrolled in the class. To add a student to the class, simply find their name in the “Students not in Class” table and click on the “Add” button next to their name.



Students not in Class

Name	Actions
Dillow, Bod (bd1@virginia.edu)	Add

This will add the student to the class, and you should see the student disappear from the “Students not in Class” table and appear in the “Enrolled students” table. To remove a student, the process is identical. In the “Enrolled Students” table, find the student you wish to remove and click “Remove.”



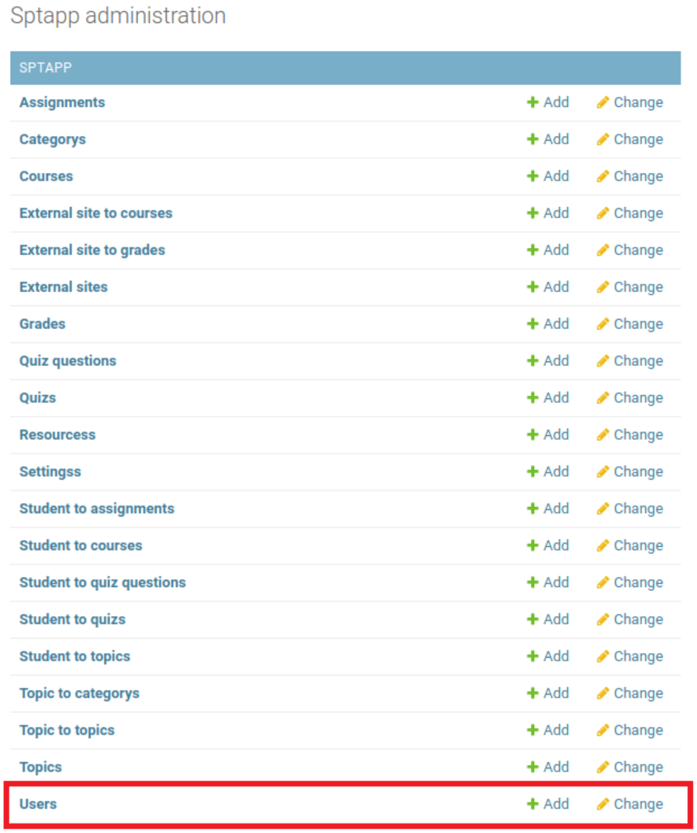
Enrolled Students

Name	Actions
Dillow, Bod (bd1@virginia.edu)	Remove

The student will then be moved from the "Enrolled Students" table to the "Students not in Class" table and the student will no longer be in the class.

Keep in mind that these tables will display up to 25 students on a single page. If there are more students, you may need to scroll through the table to see all the students or request the next page by using the "Previous" and "Next" buttons at the bottom of each table. If the page is empty, you will see "No students to show" instead of a table.

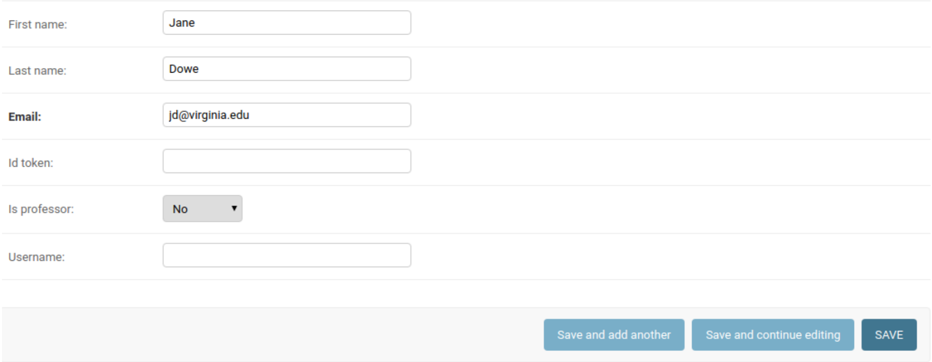
For students not yet in the system, you must either include them as part of the CSV upload, or alternatively, use the Django Admin to add them to the system on an individual basis. To do so, navigate to the admin page as detailed above and find the User tab and click "Add." (see next page for an example image)



Sptapp administration

SPTAPP		
Assignments	+ Add	Change
Categorys	+ Add	Change
Courses	+ Add	Change
External site to courses	+ Add	Change
External site to grades	+ Add	Change
External sites	+ Add	Change
Grades	+ Add	Change
Quiz questions	+ Add	Change
Quizs	+ Add	Change
Resourcess	+ Add	Change
Settingss	+ Add	Change
Student to assignments	+ Add	Change
Student to courses	+ Add	Change
Student to quiz questions	+ Add	Change
Student to quizs	+ Add	Change
Student to topics	+ Add	Change
Topic to categorys	+ Add	Change
Topic to topics	+ Add	Change
Topics	+ Add	Change
Users	+ Add	Change

This will bring you to the add user interface with many options. Most of these can safely be ignored, but all bolded fields must be filled. The important fields are: "First Name," "Last Name," and "Email," and set "Is Professor" to No. The "Password" field is obsolete but something is still required to be in it, so this can be filled with anything. When finished, press "Save."



First name:	Jane
Last name:	Dowe
Email:	jd@virginia.edu
Id token:	
Is professor:	No
Username:	

Once this is done, navigate back to the course roster page and add the student to the class, following the steps above (you may need to refresh the page for the student to show).

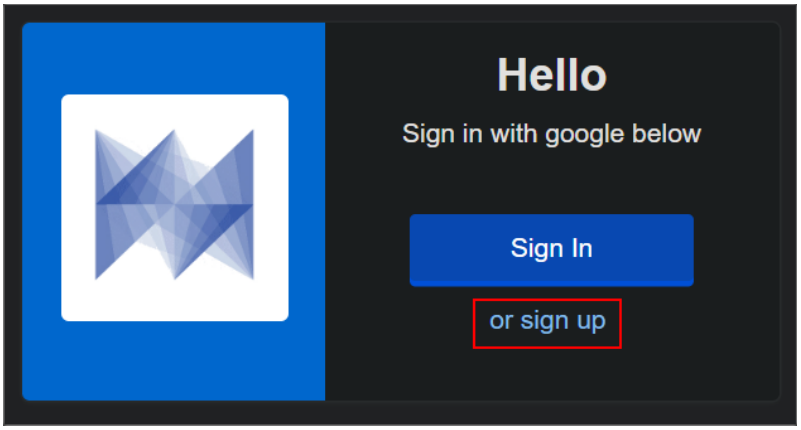
Section 3: The Student Guide

Student- Creating an Account

Note: this step is not necessary if the professor has already created an account for you

Go to {domain}/welcome (e.g. https://studentperformancetracker.xyz/welcome)

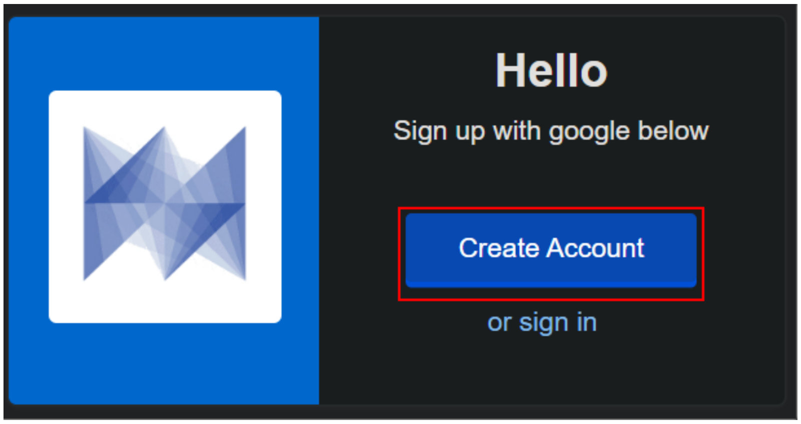
Click on "Sign Up"



Hello

Sign in with google below

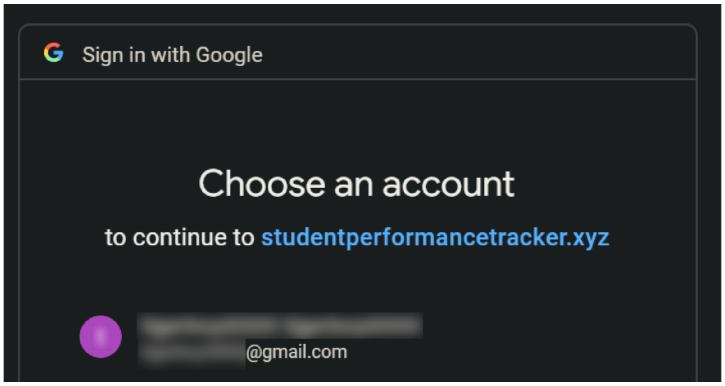
Click on "Create Account"



Hello

Sign up with google below

Choose a Google account to sign in with



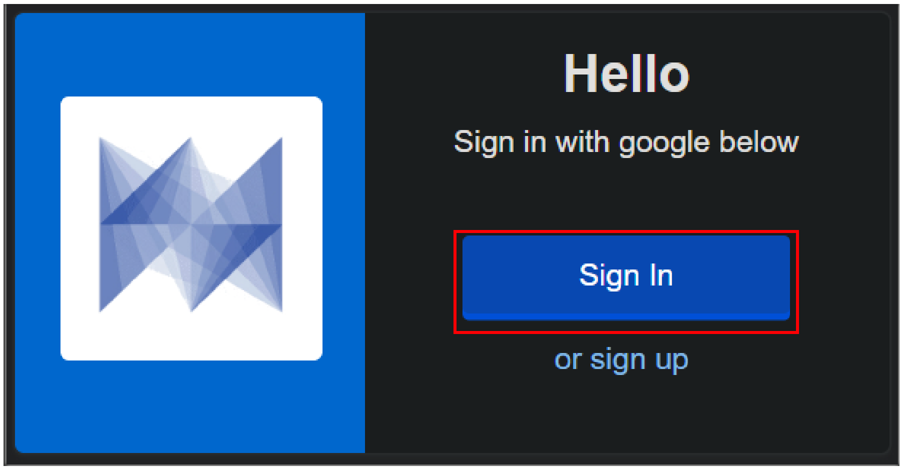
Choose an account

to continue to studentperformancetracker.xyz

[redacted]@gmail.com

Student- Signing In

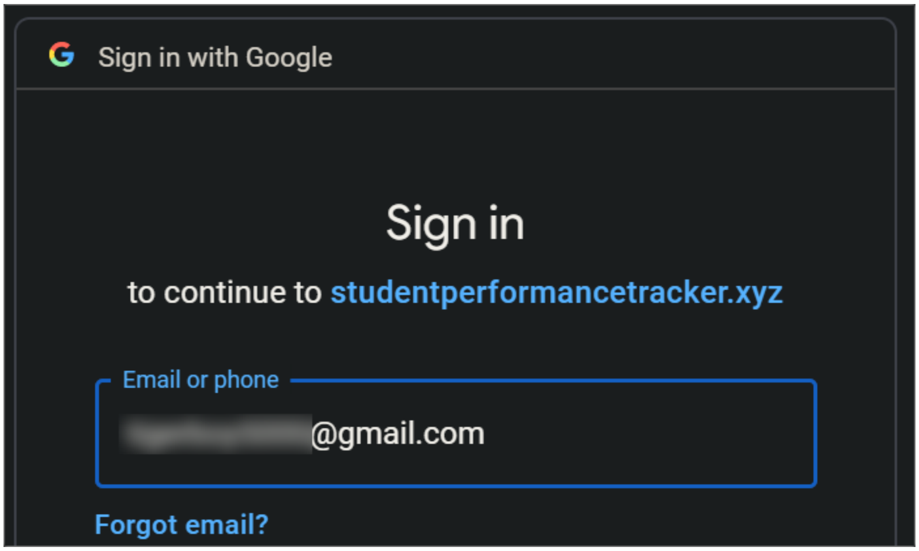
Go to {domain}/welcome (e.g. https://studentperformancetracker.xyz/welcome)
Click on “Sign In”



Hello

Sign in with google below

Sign in with the Google account that was used to create an account



Sign in with Google

Sign in

to continue to studentperformancetracker.xyz

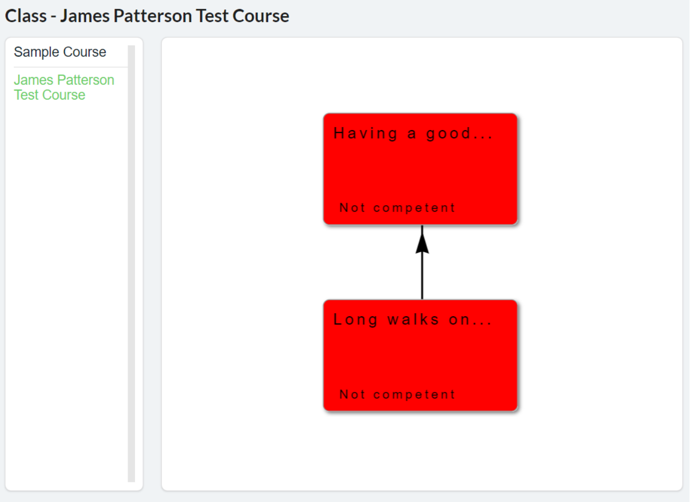
Email or phone

[redacted]@gmail.com

Forgot email?

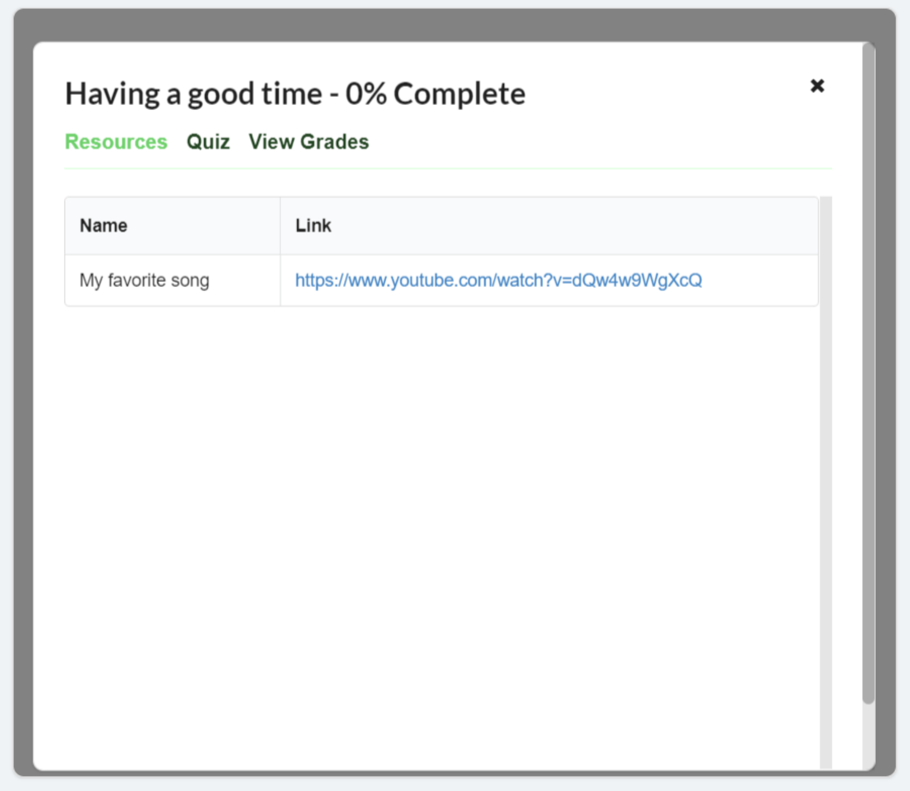
Student- Viewing Resources

When inspecting a topic for a class into which you're enrolled, you can find extra tools outlined by the professor on the 'Resources' tab. Things like these should be available to assist with assignments, or could be media designed to help you gain expertise with the topic. To view these resources, begin in a class in which you're enrolled.



Class - James Patterson Test Course

Then, select a topic in whose resources you're interested. A menu should appear; select the 'Resources' tab to see the list of provided resources.



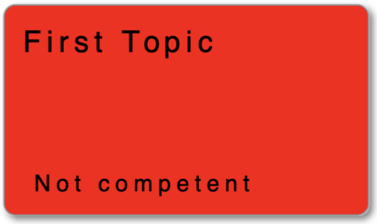
Having a good time - 0% Complete

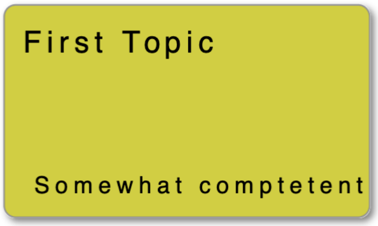
Name	Link
My favorite song	https://www.youtube.com/watch?v=dQw4w9WgXcQ

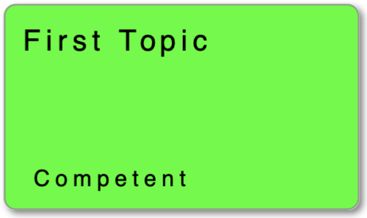
A list of resources provided by the professor should appear on this tab. Use these to help gain expertise on this topic!

Student- Viewing Grades

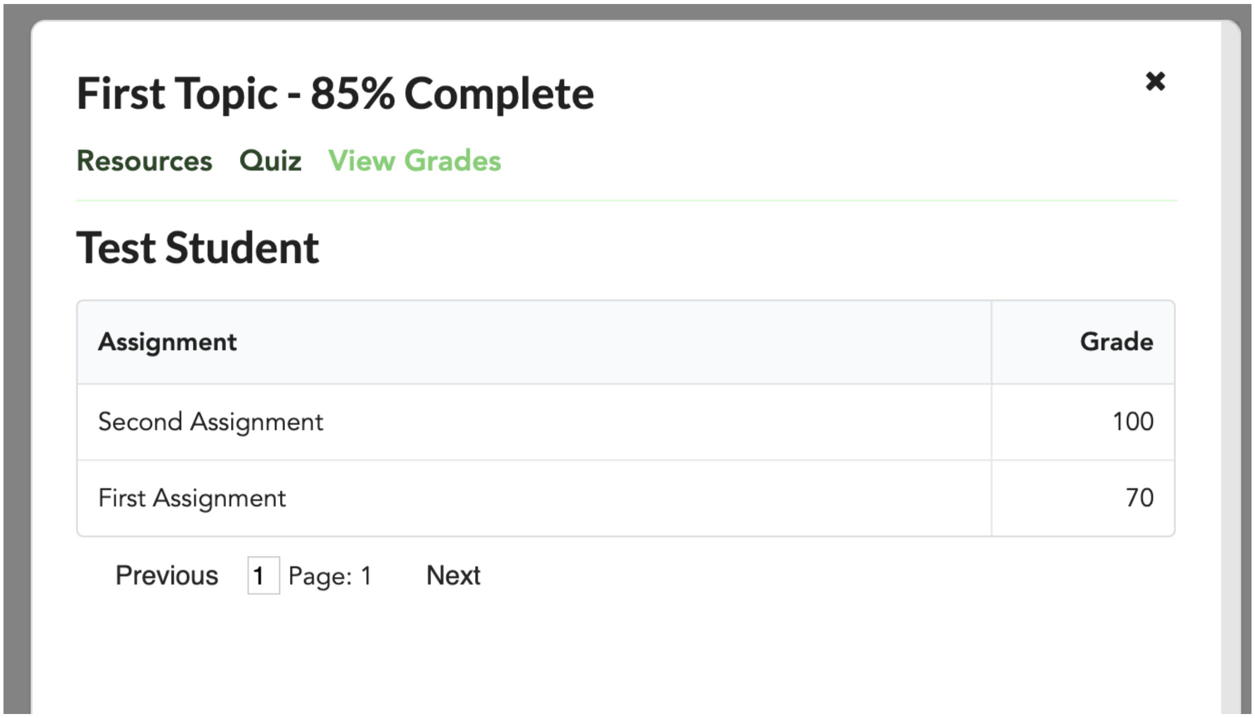
In the graph view, each topic is colored based on your current standing. Red indicates “Not Competent”, orange “Somewhat Competent, and green “Competent”. This standing is also written on each topic node.







For a more detailed breakdown of your grades, you can select a topic, bringing up the same menu as the “Viewing Resources” section. Select the “View Grades” tab, and you will see a list of your grades on each assignment for that topic.



First Topic - 85% Complete

Resources Quiz View Grades

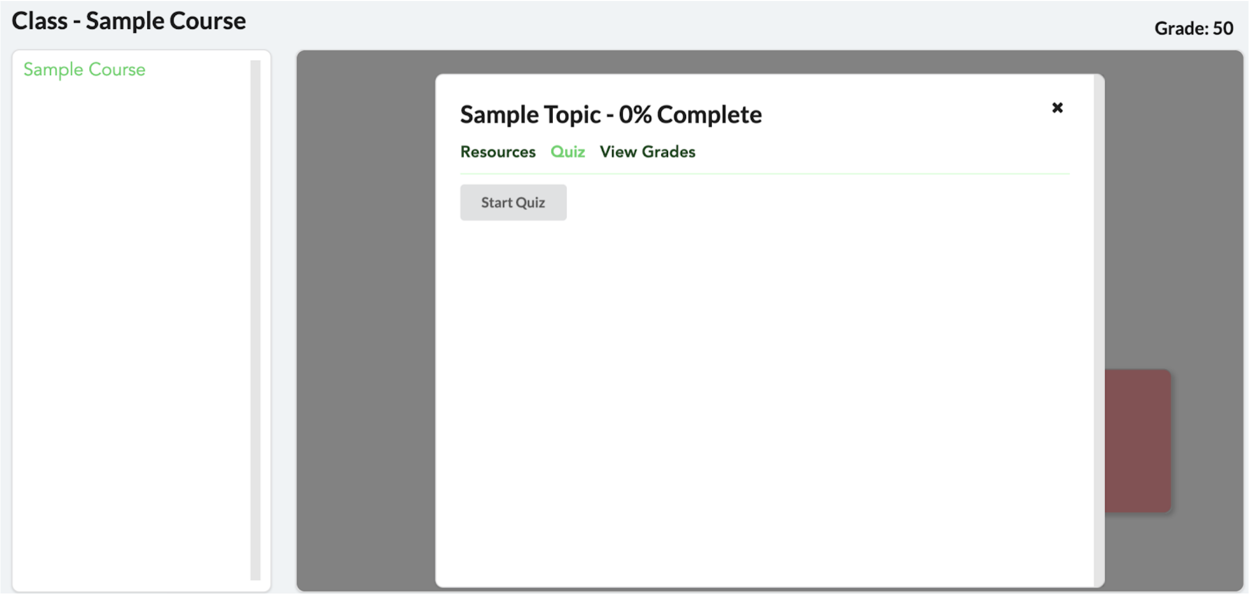
Test Student

Assignment	Grade
Second Assignment	100
First Assignment	70

Previous 1 Page: 1 Next

Student- Taking Quizzes

Navigate to the Course, and specific Topic you want to take a quiz for.



Press *Start Quiz*

Choose your answer and press Submit when you are ready.

Your grade is calculated dynamically by a specialized algorithm.

Continue until the quiz ends, or press *Exit Quiz* when you reach a satisfactory grade. Your grade is saved each time you answer a question!

Section 4: The Developer Guide

System Architecture

The system is split into two distinct parts: the frontend and the backend. The frontend uses Vue and Vuex, and the backend uses Django, the Django REST Framework, and PostgreSQL

Frontend

Vue is used to display and interact with the data on the backend. Vue typically makes an API call to the backend, loads the JSON into memory in javascript, and displays the information on the webpage. Vue also makes API requests to the backend that modify data in cases such as editing the class graph or taking a quiz. Vuex is used to provide authentication (login) information, user settings, and toast functionality (user feedback) to all Vue pages.

Backend

Django defines the API url patterns with urls.py, and the code that responds to those patterns are defined in api_views.py. Django interacts with PostgreSQL through a database abstraction layer, which means we use python code instead of SQL to interact with the database.

Authentication

All endpoints are secured with GoogleOAuth as defined by auth.py. HTTP authorization bearer tokens are authenticated with google and then stored in the database. When the id_token is verified, the user associated with that token becomes associated with the api request. Next, the permissions defined for a particular api_view are checked. These permissions are defined in permissions.py. If both authentication and permission checks pass, the api_view is executed. On the frontend, the id_token is stored in vuex and is sent with each api request.

Production Deployment Design

In production we use the docker-compose.prod.yml file instead of docker-compose.yml which is intended for development. This production file defines five containers and two volumes. The containers include the Django backend API, the database application PostgreSQL, the web frontend, the web server Nginx, and the certificate manager Certbot. One volume stores the database files and the other stores the frontend static files for the application and the django admin page. This production setup uses nginx to serve the entire application. Nginx routes backend requests to django and it serves the frontend directly as static html, css, and javascript files. The nginx configuration is defined in src/config/nginx/conf.d/spt.conf. The frontend container starts a webpack build to generate the static files, places the files into the static volume, and then exits. Certbot is mostly idle and checks every 12 hours to see if the certificates need to be renewed, and renews them if necessary.

There are several differences in the development and production setup that we were not able to separate through environment variables alone. As a result, we developed the apply_deployment_vars.sh bash script to easily apply the necessary changes to the files. This script is intended to be run once before starting docker-compose.prod.yml. Instructions for this script are included in the installation instructions. The init-letsencrypt.sh script collects certificates so that nginx can serve the application through HTTPS. Instructions for this script are also included in the installation instructions.

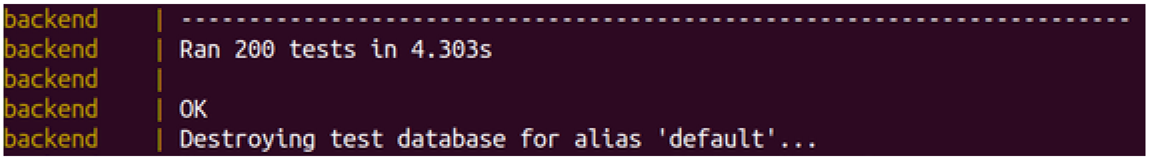
Developer- Running Tests

There is a simple shell script provided with the code that easily allows one to run the testing suite. First, navigate to the product's /src directory and then run one of the following commands:

$ sudo ./test-backend.sh

$ sudo ./test-frontend.sh

This will automatically start up the docker containers for the database and backend/frontend server, then run the corresponding tests.

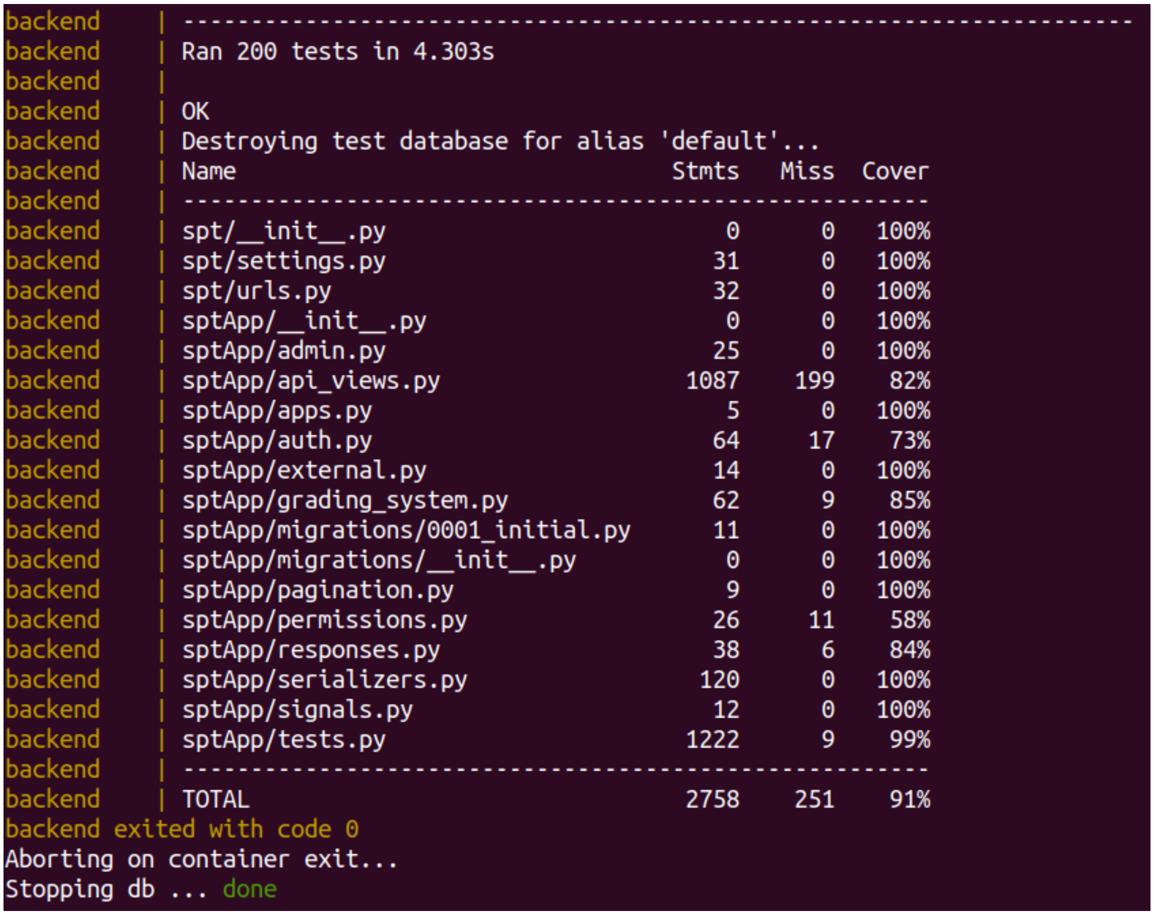


```
backend     | ----------------------------------------------------------------------
backend     | Ran 200 tests in 4.303s
backend     |
backend     | OK
backend     | Destroying test database for alias 'default'...
```

Should a test fail, the failing tests will be displayed with the reason why they failed.

Developer- Generating Code Coverage Reports

In addition to running the tests, the same shell script will generate a code coverage report following the completion of the tests. The coverage reports looks as follows:



```
backend      | ----------------------------------------------------------------------
backend      | Ran 200 tests in 4.303s
backend      |
backend      | OK
backend      | Destroying test database for alias 'default'...
backend      | Name                                    Stmts   Miss  Cover
backend      | ---------------------------------------------------------------
backend      | spt/__init__.py                             0      0   100%
backend      | spt/settings.py                            31      0   100%
backend      | spt/urls.py                                32      0   100%
backend      | sptApp/__init__.py                          0      0   100%
backend      | sptApp/admin.py                            25      0   100%
backend      | sptApp/api_views.py                      1087    199    82%
backend      | sptApp/apps.py                              5      0   100%
backend      | sptApp/auth.py                             64     17    73%
backend      | sptApp/external.py                         14      0   100%
backend      | sptApp/grading_system.py                   62      9    85%
backend      | sptApp/migrations/0001_initial.py          11      0   100%
backend      | sptApp/migrations/__init__.py               0      0   100%
backend      | sptApp/pagination.py                        9      0   100%
backend      | sptApp/permissions.py                      26     11    58%
backend      | sptApp/responses.py                        38      6    84%
backend      | sptApp/serializers.py                     120      0   100%
backend      | sptApp/signals.py                          12      0   100%
backend      | sptApp/tests.py                          1222      9    99%
backend      | ---------------------------------------------------------------
backend      | TOTAL                                    2758    251    91%
backend exited with code 0
Aborting on container exit...
Stopping db ... done
```

This information includes the number of tests ran and the files covered by the tests as well as the percentage of the lines in those files that were covered by the tests. Finally, the total code coverage is displayed at the bottom - in this case, it is 91%. If any of the tests fail, this will also notify you of those failing tests. Once the tests are complete and the table is shown, the shell script will automatically close the docker containers and exit normally.

To generate the web browser html report, execute the script htmlcov.sh. Then open backend/htmlcov/index.html with a web browser.