

Iowa State University Electronic Program of Study

Final Report

Team 4

Tina Prouty (Adviser) Maruf Ahamed (Academic Advisor)

Elisabeth Bair (Meeting Facilitator), Andrew Goluch (Frontend Lead),
Lucas Bell-Steckel (Backend Lead), William Peng (Full-Stack
Coordinator), Cavin Leeds (Test Engineer), and Thomas Hotard
(Report Manager)

sddec21-04@iastate.edu

<https://sddec21-04.sd.ece.iastate.edu/>

Revised: December 8th, 2021

1 Overview

1.1 ACKNOWLEDGMENT

Special thanks to Patrick Determan, a colleague of Tina Prouty, who contributed valuable insight to the design and functionality of this project.

1.2 PROBLEM AND PROJECT STATEMENT

Every student who attends Iowa State has certain coursework requirements they need to fulfill before they graduate. Meeting all requirements can be confusing and overwhelming, especially to freshman students who have a lot to learn and little experience with the courses available. Because of this, every student in the Electrical and Computer Engineering Department (including Software and Cyber Security Engineering majors) needs to create a Program of Study (POS) in their Introduction to Engineering course. These programs are checked by hand, taking enormous amounts of time and leaving room for mistakes to be made.

The solution to this problem is an Electronic Program of Study. This program will be electronic and create a simple and effective tool for the creation of a student's POS. This tool will include features like prerequisite checking, graduation requirement assurance, different course requirements based on student catalog, and more. All of these features will aid advisors and students to maximize their time and provide powerful knowledge to ensure success in the selection of classes.

By the end of this project, there will be a web application with the capability to store, update and validate the Programs of Study for students in Electrical, Computer, Software, and Cyber Security Engineering. Students will be able to create this POS, and advisors will be able to view students' POS, update available courses, and approve exceptions to general requirements. This program will be accessible via a created account and will require minimal software updates going forward, making this a tool that can be used well into the future.

1.3 INTENDED USERS AND USES

In this project, there are two primary users: students and advisors. These users have different needs and uses for this program.

First, students need a way to create a POS that can be a living document with immediate feedback about accuracy and potential errors. Most students make some changes when they select their first year at Iowa State, so they need to be able to access and edit their POS for at least four years after creation. Students also need to have access to requirements and a list of courses that satisfy the

requirements of their catalog. They also need to receive advisor feedback and identify errors and have some idea how to fix them.

Second and finally, advisors need to be able to make updates to degree requirements, access student POS's, and give feedback to students. They also need to have access to all student programs. This will allow for time to be saved when grading the POS assignment for the Intro to Engineering course.

1.4 FUNCTIONAL REQUIREMENTS

- Support the creation of programs of study for Electrical, Computer, Software, and Cyber Security Engineering majors
- Require students to accurately complete a program of study with the goal of graduation
- Provide warnings for missing classes
- Enable students to edit their program of study
- Allows advisors to add new classes to the program
- Classes are able to be labeled as required, based on a given catalog year
- Individual courses are able to have prerequisites that must be met within the POS
- Advisors must be able to approve exceptions to general requirements
- Advisors are able to give feedback to a student's POS
- Application can be used by future students and is maintainable

There are some functionalities not included in these requirements because they are not necessary to a successful end product.

1.5 NON-FUNCTIONAL REQUIREMENTS

- Website should load within three seconds
- Website able to be maintained and continually improved
- Login information securely stored and transmitted, unauthenticated user cannot view or save changes
- Checking the schedule takes less than two seconds

1.6 STANDARDS

The relevant standards to this project are P23026 - Systems and Software Engineering -- Engineering and Management of Websites for Systems, Software, and Services Information and IEEE/ISO/IEC 23026-2015 - ISO/IEC/IEEE International Standard - Systems and software engineering - Engineering and management of websites for systems, software, and services information. Both of these standards inform the development and design of informational websites and give goals on locating relevant and timely information, applying information

security management, facilitating ease of use, and providing for consistent and efficient development and maintenance practices. In practice, these standards have been applied to our project by informing how we planned to test the application and created the non-functional requirements.

1.7 ENGINEERING CONSTRAINTS AND REQUIREMENTS

The only requirements our group will need are access to our git repository, working code editors, database software, and access to any subscriptions we may need to edit or test code. The school will need to provide a server to support the application in the long term, though not for most of the development process.

2 Design

2.1 PREVIOUS WORK AND LITERATURE

There are no similar products that exist, though the advisors of the EE/SE/CprE/CyE departments have implemented certain procedures for the creation of a program of study for their students. Some of the previous methods that advisors have had their students use to create their program of study (POS) include filling out an Excel spreadsheet and filling out a pdf form. Both of these previous methods come with serious drawbacks however, being clunky and tedious to fill in as there is no convenient method to check for prerequisites or corequisites. They do have the benefit of being commonly used formats and the software to edit them are readily available.

2.2 DESIGN EVOLUTION FROM SE 491

For the backend, we used Django for the entirety of the backend implementation instead of flask or MySQL libraries. The reason for the switch was due to Django handling certain aspects of the application such as Users, authentication, authorization, and support for Google integration.

Due to the university not having a database for all the available classes, the only way we can get the classes is from the ISU catalog webpage. And because the format for prerequisites is not consistent in the catalog page, we changed the prerequisite checking for each class to checking what else is needed for completing the degree.

Because we could not get the server working by ETG in time, there was not enough time to have ETG set up and enable Okta for our project. So, we decided to use the Django authentication system to set up accounts for the project.

We decided to move away from adding flowchart arrows into the view. This is because of challenges with the arrows needing to be dynamic, as well as the issues we faced with the prerequisites in the webscraper.

In addition, the largest evolution from 491's design was the implementation of the schedule validator/check schedule feature. Because of complicated boolean logic and inconsistent pre-requisite standards in the course information, it was not possible to implement prerequisite checking at this time. However, because the goal of the project was to save advisors time in checking the schedules, functionality was still added to check the core courses of each degree program. This feature and functionality change was approved by our client.

2.3 SECURITY CONCERNS AND COUNTERMEASURES

Server: Currently, the server is not set up to handle https requests. This is due to lack of time for requesting a SSL certificate through ISU's IT Security Department. A simple fix for this is simply to acquire a SSL certificate and configure Apache and the server for https requests.

Okta: The application is not set up for 2FA with Okta. Anyone on the network can register an account and use the service. To counter this the application would need to be altered so as to use the Django module for Google/Okta integration.

3 Implementation

In general terms, we designed a web application to house the Program of Study. This application is supported by a database developed by the backend team. This database stores all the courses, degrees, users, their schedules and account privileges. The website, developed by the frontend team, is designed to create an easily interactable PoS which can be applied to the students 4 year plan. The product itself is a web application written in JavaScript using JQuery and Django. There are three account types, one for students, one for advisors/faculty, and one for admins. The web app stores some information from users, like their current 4 year plan as well as additional plans they chose to save. The app also allows both students and advisors to check their schedules against the core courses in a given degree program.

The Django framework takes care of a lot of the back end implementation and lets the back end be worked on as if it is at a high level. This included the log in, authentication, database, and site navigation components.

The database has 4 models, Student, Advisor, Class, and Degree. The Student and Advisor model has a one-to-one field connected to the Django.auth.user model. The Class model stores all the courses in the ISU catalog. The Degree model stores the required class for CYB E, S E, EE, and CPR E. Our database can be populated with the thousands of Iowa State courses using a python web scraper.

The front end consists of a student view and account which can be utilized to create and save schedule plans. The advisor account can view the students schedule and type comments that the student can view.

The courses themselves are HTML div objects with onclick functions to display all the relevant course information to students. They also use OnDrag event listeners in conjunction with the semester boxes so that the user can drag and drop courses into the semester of their choice.

Checking schedules/schedule validation was implemented using the list of courses in a student's schedule checked against the courses in a given degree. The degree courses consist of a list of core courses and a list of courses that students are able to choose from. After checking the student's schedule against those lists, any courses that are missing are displayed. Courses that are not core courses are also displayed so that an advisor can see if other requirements (like general education courses) are met.

For a more detailed implementation guide, see the operational manual in Appendix I.

4 Testing

All testing was abandoned for the project due to delays in the project leading to a late implementation of a working prototype. The original plan was to implement both unit and interface testing through Selenium. Acceptance testing would have been through the students taking the S E or CPR E 166 classes as an option for their final project. A major contributor to the delays was the importing of the data. Having been given no form of database for the classes, we resorted to using a python web scraper. After some difficulties figuring out all the edge cases of the formatting ISU's azcourses' website, a persistent challenge was interpreting the prerequisites. Courses could have just class names and punctuation or they could use english sentences or they could have a mix. With the severe inconsistencies, prerequisite parsing and interpretation was significantly more complex and challenging than initially believed.

5 Appendices

APPENDIX I- OPERATIONAL MANUAL

Server:

The server is being hosted by ETG. Jacob Grundmeier was our main point of contact with ETG for the final server. A bash script to pull from Gitlab named epos is located at /usr/local/bin.

Accessing the server:

SSH INFO:

Hostname: epos.ece.iastate.edu

User: vm-user

Password: EdcfokZWxwNM

To run:

Without Pulling from the Git repository:

Nothing. As long as the Apache service is running the web app will be running on the server.

Pulling from the Git repository:

Login into the server, and run sudo epos. Follow the given instructions.

Website use:

To utilize the website as a student user the student would navigate to epos.ece.iastate.edu and either log in or create an account.

IOWA STATE UNIVERSITY

Login

Username

Password

Login

Don't have an account? [Sign Up](#)

(Login Page) epos.ece.iastate.edu/

If the student or advisor needs to create an account they click on the Signup link. To login the user just enters username and password.



Register Account

Username:

Password:

Password confirmation:

Submit

Already have an account? [Login](#)

(registration page) epos.iastate.edu/registerPage/

When the user is registered they are directed to the login screen, where they can login and be directed to the homepage. If a new advisor account is created move to the advisor section.

Students

Once the students login they are taken to the student homepage where they see a variety of options. The first thing students should do is click on the Edit profile link to add profile information. Once the student clicks save text will pop up saying “Saved Successfully” and the user can click the home link.



User Profile

First Name:

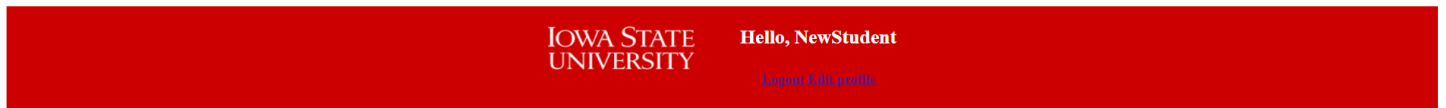
Last Name:

Email:

Major:

Save

(User Profile) epos.ece.iastate.edu/userProfile



Program of Study



Submit Add Semester Remove Semester



Class Description

Search Class

Department Name

Course Number

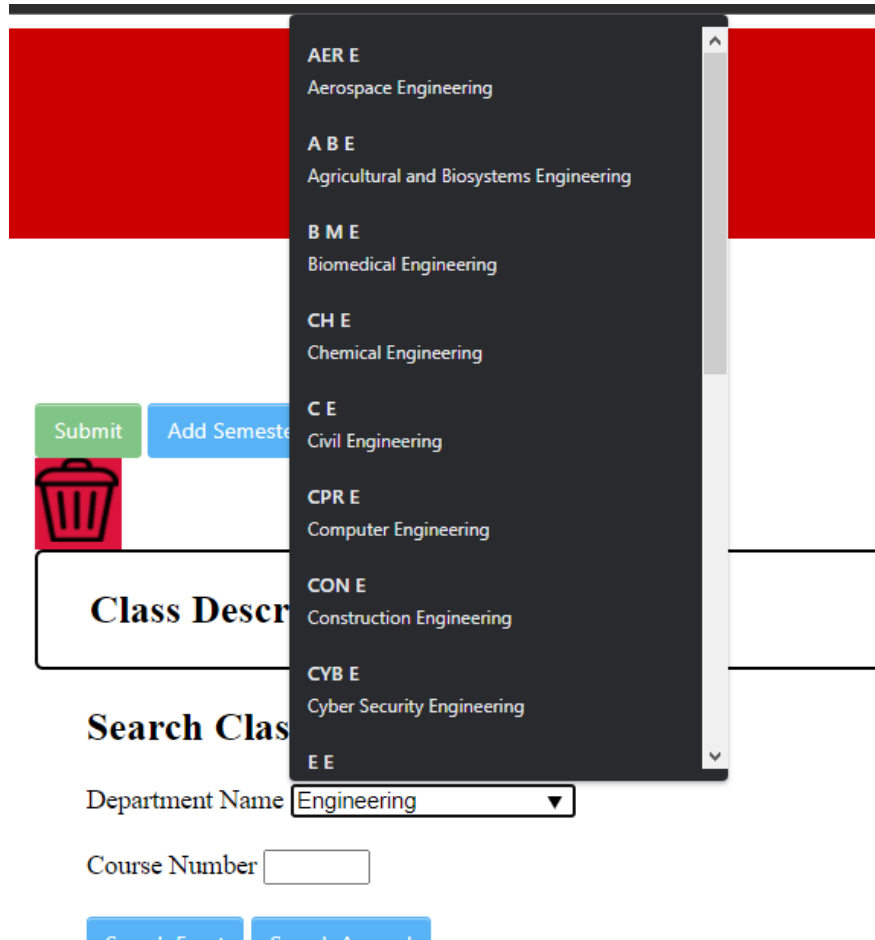
Search Exact Search Around

Check Schedule Add Core Courses

Advisor Comments: No Comments

(Student Home Page) epos.ece.iastate.edu/studentPage

Now the student can begin adding courses to the schedule, if they would like to search for a course they can use the search class section and enter in the Department name and course number. The student can also search around a course and the output will be the three courses before and after by course number if applicable.



Searching for a course the department autofills

Search Class

Department Name

Course Number

Search Exact

Search Around

Check Schedule

Add Core Courses

SE 101	SE 166	SE 185	SE 309
SE 317	SE 319	SE 329	

Searching around the course SE 309

Search Class

Department Name

Course Number

Search Exact

Search Around

Check Schedule

Add Core Courses

Advisor Comments: No Comments

SE 309

Searching exactly for SE 309

The student can then search for the exact course to add it to their flowchart which will add a draggable course box they can place into a semester. any class box can be click dragged into a semester. for more information on any course the user can click on the course and information will appear in the course description box.If the user wants to delete a course they can drag the course over the red trashcan and the course will be removed from the screen. The student also has full control over the number of semesters with the default semester always available for transfer credit. The student can click on the Add semester and Remove semester buttons to add add and remove respectively. At any point the student can click submit to save their progress.

Program of Study

The screenshot shows the 'Program of Study' interface. On the left, a grey bar represents a semester, with a blue box labeled 'S E 101' being dragged into it. A mouse cursor is visible over the blue box. To the right, there is a 'Class Description' section with a search form. The search form includes fields for 'Department Name' (containing 'S E') and 'Course Number'. Below the search form are buttons for 'Search Exact' and 'Search Around'. At the bottom right, there is a 'Check Schedule' button and an 'Add Core Courses' button. Below these buttons, a small blue box labeled 'S E 101' is shown, along with the text 'Advisor Comments: No Comments'.

Dragging a course into a semester

The screenshot shows the 'Program of Study' interface after a course has been added to a semester. The grey bar on the left now contains a blue box labeled 'S E 101'. To the right, the 'Class Description' section is expanded, showing the following details:

Class Description

S E 101: Software Engineering Orientation

Credits: R

Prereqs: db.Class.None, db.Class.None, db.Class.None, db.Class.None, db.Class.None, db.Class.None

Description: Introduction to the procedures, policies, and resources of Iowa State University and the Software Engineering Program

clicking on a course to get relevant details

Students can also check their program of study against the core curriculum for Software Engineering, Computer Engineering, Electrical Engineering, and Cyber Engineering, by selecting a degree and clicking the check schedule button. Then a list of all missing courses will be generated in red based on the courses in the current four year plan. If a student has courses that don't count for the core degree requirement they will appear in the Other courses section.

SE

Check Schedule

Add Core Courses

Advisor Comments: No Comments

Missing core courses: MATH 165,COM S 227,ENGL 150,MATH 166,S E 101,S E 166,LIB 160,PHYS 221,S E 185,CPR E 281,S E 319,ENGL 250,MATH 267,COM S 228,SP CM 212,COM S 363,COM S 311,S E 309,S E 317,S E 339,S E 491,S E 492,STAT 330,S E 421,CHEM 167, CHEM 177,COM S 327, CPR E 288,COM S 352, CPR E 308,COM S 230, CPR E 310,ENGL 314, ENGL 309,COM S 321, CPR E 381,ECON 101,

Other courses:

Result of check schedule

CYB E

Check Schedule

Add Core Courses

Advisor Comments: Need to add Chem req

Missing core courses: CHEM 167,CPR E 185,CPR E 166,ENGR 101,CPR E 294,CYB E 231,CYB E 230,CYB E 234,CPR E 381,CPR E 310,COM S 309,CYB E 331,CPR E 491,CPR E 492,CPR E 494

Other courses: S E 101,S E 166,S E 185,S E 319,COM S 230,S E 309,S E 339,COM S 363,S E 317,COM S 321,SP CM 212,S E 491,S E 421,S E 492

Example of someone with an SE degree checking against the CYB E degree

To save the student time they can generate all the needed courses for a specific degree by selecting the degree in the same way to the step above and clicking add Core courses. The display will show courses in blue which are all required and then courses in color pairing where there are multiple options for a degree requirement. Lastly the student can view any feedback written by an advisor in blue text on the page.

SE					
Check Schedule	Add Core Courses				
MATH 165	COM S 227	ENGL 150	MATH 166	SE 101	SE 166
LIB 160	PHYS 221	SE 185	CPR E 281	SE 319	ENGL 250
MATH 267	COM S 228	SP CM 212	COM S 363	COM S 311	SE 309
SE 317	SE 339	SE 491	SE 492	STAT 330	SE 421
CHEM 167	CHEM 177	COM S 327	CPR E 288	COM S 352	CPR E 308
COM S 230	CPR E 310	ENGL 314	ENGL 309	COM S 321	CPR E 381
ECON 101	ECON 102	IE 305	MATH 207	MATH 304	MATH 314
MATH 317	MATH 265	Advisor Comments: No Comments			

Output of Add core courses schedules

Advisor Comments: Missing Chemistry course

Advisor comments

IOWA STATE UNIVERSITY Hello, NewStudent
Logout | My Profile

Program of Study

Transfer Credits	MATH 165	COM S 227	ENGL 150	PHYS 221	ECON 102					
Semester 1	MATH 166	S E 101	S E 185	ENGL 250	CHEM 167	ECON 101				
Semester 2	S E 166	LIB 160	CPR E 281	COM S 228	MATH 207					
Semester 3	S E 319	CPR E 288	MATH 267							
Semester 4	S E 309	S E 317	CPR E 310							
Semester 5	COM S 363	COM S 311	S E 339	CPR E 308	ENGL 314	COM S 321				
Semester 6	SP CM 212	S E 401	STAT 330	S E 421						
Semester 7	S E 492									

SubmitAdd SemesterRemove Semester

Class Description

Search Class

Department Name

Course Number

Search Class Search Around

Check ScheduleAdd Core Courses

Advisor Comments: No Comments

Example of SE schedule.

Advisor

The three steps below will take an already created account and transform it into an Advisor account. The Advisors will create their accounts and login on the pages as students.

1. login to admin page at epos.ece.iastate.edu/admin/ (to create an admin account, run `python manage.py createsuperuser` then `python manage.py makemigrations` and `python manage.py migrate`)

Django administration

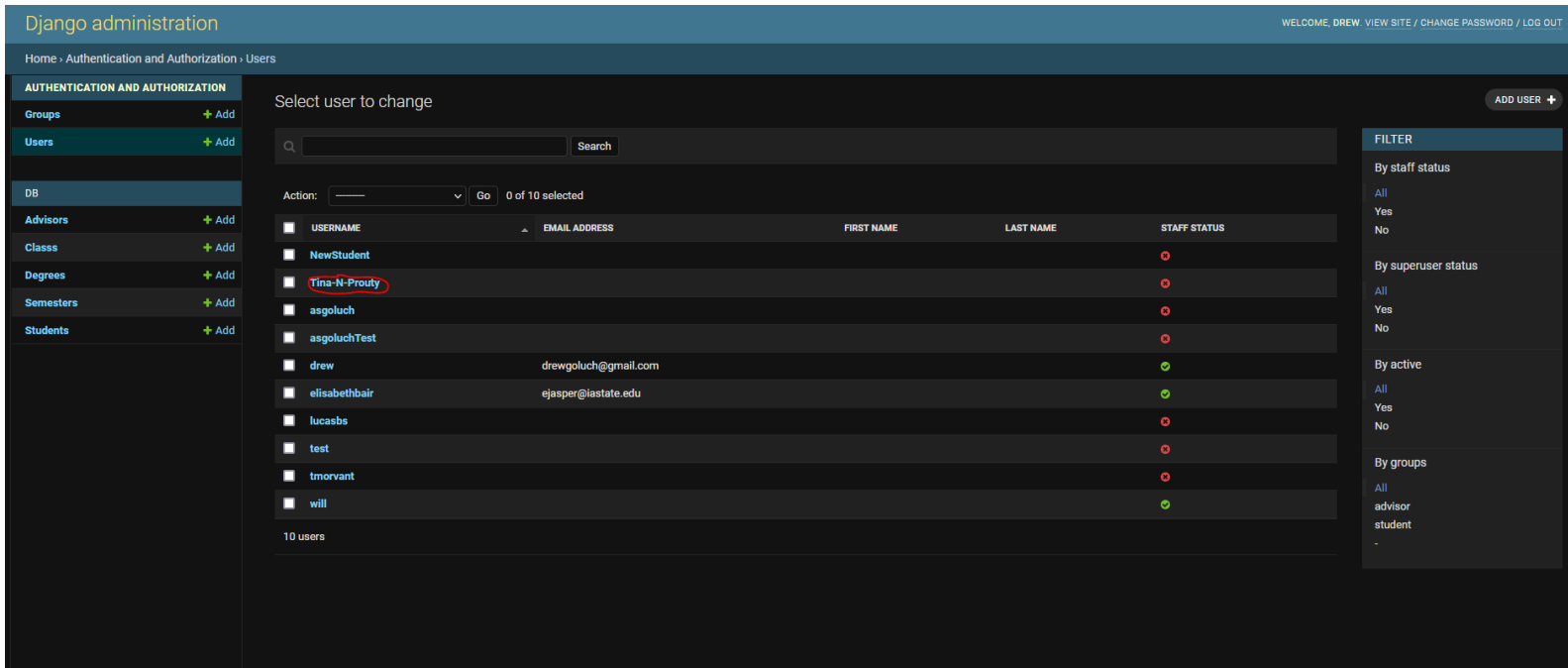
Username:

Password:

Log in

(admin login) epos.ece.iastate.edu/admin/

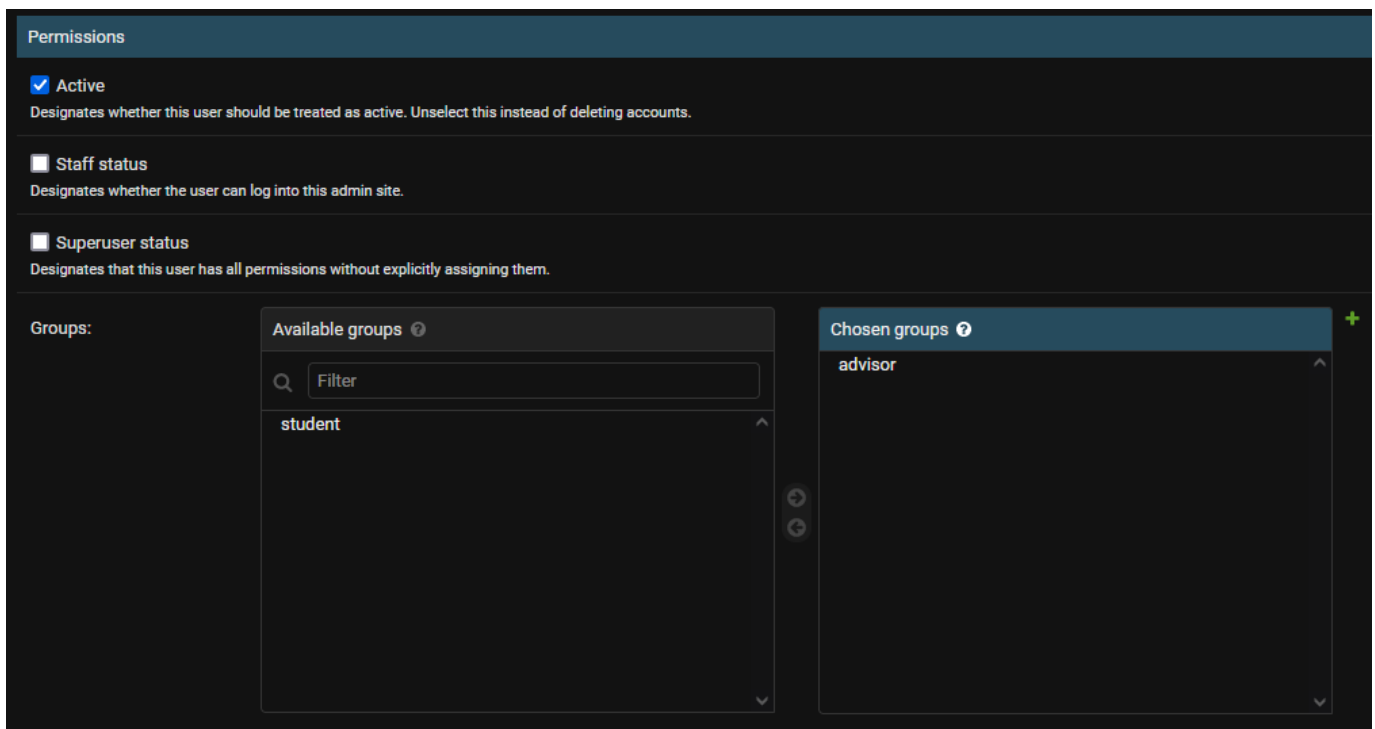
- In this page admin users can modify privileges of accounts to give certain accounts either advisor or admin privileges. To add an advisor click on “users” and then the account you wish to make an advisor



The screenshot shows the Django administration interface for the 'Users' section. The page title is 'Django administration' and the user is logged in as 'DREW'. The breadcrumb trail is 'Home > Authentication and Authorization > Users'. The left sidebar shows the 'AUTHENTICATION AND AUTHORIZATION' section with 'Users' selected. The main content area is titled 'Select user to change' and contains a search bar, an 'Action:' dropdown, and a table of users. The user 'Tina-N-Prouty' is highlighted with a red circle. The table has columns for USERNAME, EMAIL ADDRESS, FIRST NAME, LAST NAME, and STAFF STATUS. The right sidebar contains a 'FILTER' section with options for staff status, superuser status, and active status.

USERNAME	EMAIL ADDRESS	FIRST NAME	LAST NAME	STAFF STATUS
NewStudent				
Tina-N-Prouty				
asgoluch				
asgoluchTest				
drew	drewgoluch@gmail.com			
elisabethbair	ejasper@iastate.edu			
lucasbs				
test				
tmorvant				
will				

- Then under permissions the user needs to be added to the advisor group and removed from the student group, this is done by clicking on the name of the group and then the right or left arrow



The screenshot shows the Django administration interface for the 'Permissions' section. The page title is 'Permissions'. The 'Active' checkbox is checked. The 'Staff status' and 'Superuser status' checkboxes are unchecked. The 'Groups' section shows a list of available groups and a list of chosen groups. The 'Available groups' list contains 'student'. The 'Chosen groups' list contains 'advisor'. The page is dark-themed.

Permissions

- Active**
Designates whether this user should be treated as active. Unselect this instead of deleting accounts.
- Staff status**
Designates whether the user can log into this admin site.
- Superuser status**
Designates that this user has all permissions without explicitly assigning them.

Groups:

Available groups

- student

Chosen groups

- advisor

Once an advisor is created they can login to go to the advisor home page. In this page advisors can search for a student by full name and see their schedule. They can write any comments into the comments box and then clicking submit to share the comments with the student. Similar to students the advisor can check their schedule against the degree plans for Software Engineering, Computer Engineering, Electrical Engineering, and Cyber Engineering. by selecting a degree and clicking the check schedule button.

Advisor home page

Transfer	Semester 1	Semester 2	Semester 3	Semester 4	Semester 5	Semester 6	Semester 7
MATH 165	MATH 166	S E 166	S E 319	S E 309	COM S 363	SP CM 212	S E 492
COM S 227	S E 101	LIB 160	CPR E 288	S E 317	COM S 311	S E 491	
ENGL 150	S E 185	CPR E 281	MATH 267	CPR E 310	S E 339	STAT 330	
PHYS 221	ENGL 250	COM S 228			CPR E 308	S E 421	
ECON 102	CHEM 167	MATH 207			ENGL 314		
	ECON 101				COM S 321		

Advisor searching for student “New Student” viewing the schedule and adding comments

S E

Check Schedule

Missing core courses: CHEM 167, CHEM 177 Other courses:

Transfer	Semester 1	Semester 2	Semester 3	Semester 4	Semester 5	Semester 6	Semester 7
MATH 165	MATH 166	S E 166	S E 319	S E 309	COM S 363	SP CM 212	S E 492
COM S 227	S E 101	LIB 160	CPR E 288	S E 317	COM S 311	S E 491	
ENGL 150	S E 185	CPR E 281	MATH 267	CPR E 310	S E 339	STAT 330	
PHYS 221	ENGL 250	COM S 228			CPR E 308	S E 421	
ECON 102	ECON 101	MATH 207			ENGL 314		
					COM S 321		

Checking schedule with a missing requirement