Date: Nov 21, 2025
To: Elizabeth Bennett
University Registrar

Joining WebSOC and WebReg

Proposal for integration of WebSOC and WebReg to improve class registration process

Brandon Han, Shreya Pujari, Yaseen Patel, Zongze Li, Ava Kopecky

Table of Contents

Executive Summary:

What is this proposal for?

In their current state, WebSOC and WebReg are systems that do not adequately meet the needs of UCI students. An observation of student sentiment and other schools reveals that in their current state, WebSOC and WebReg needlessly complicate the registration process. This has even led students to create their own alternative tools to register for classes to make the process more bearable, like AntAlmanac. Equivalent systems in other schools like UC Riverside have features to streamline registration that UCI lacks, like an integrated schedule planner that allows for easy enrollment once registration windows open.

Aside from causing student frustration, this friction also strains the limits of the system. WebReg is limited in how many students it can handle simultaneously, which means that students taking longer to register holds back other students too. This restricts registration windows for students, as they have to be small to ensure the system isn't overburdened by too many trying to access it simultaneously. Fixing this problem would make it less likely that students run into issues with WebReg being overwhelmed, which would also make it possible for more students to have earlier registration windows.

Our proposed solution

We propose an integration of these two systems that would allow students to directly plan their courses in advance and rapidly register for those courses once their registration window opens. This solution improves the student experience for class registration by implementing features from alternative systems that students prefer while also serving to remove much of the friction that currently exists when going between WebSOC and WebReg to register for classes, allowing for faster and more efficient registration to lessen the burden placed on the system.

Budgeting and Timeline

This solution has a projected budget of around $1,000,000 dollars, and is to be implemented over the course of UCI's summer break (Mid June to Late September) so as to not overlap with UCI's school year. Given how vital registration is for students to be able to enroll in courses and graduate, it is imperative that this process be improved on for the well being of all students.

Systems in Question:

WebSOC:

UCI's online database that stores data about current, past, and future UCI classes to aid students in planning their schedules and signing up for classes. Students can find information about classes by selecting options from various filters, which will then cause the site to return a list of classes for the specified quarter. The search returns information such as course names, course codes, and available sections to aid students in registering for classes.

After searching WebSOC with their desired filters, students get a large table of results. Course codes are how the system identifies specific courses and their sections. For each course they want to take in the results, students analyze the available sections' time blocks and by hand save the course codes for the ones they selected.

WebReg:

The registration system used by students to sign up for their classes. Students are provided a registration window, during which they can log in to manage their current schedule. Courses and their sections are specified through course codes, and students can sign up for, drop, or modify the selected courses. Students can view the classes they have signed up for by looking at a text based study list output. Students can also view the classes they are currently waitlisted for along with their position on the waitlist.

In the current system, students must find their desired courses for their quarter outside of WebReg and copy the saved course codes back into the system through external tools.

Current Problems

Student Frustration with Enrollment

Because WebSOC is a separate application to WebReg, extra work is required to take information from WebSOC about classes and use it for registration on WebReg. This is compounded by the fact that WebSOC doesn't provide tools for saving or even copying course codes, forcing the use of manual copying or external tools if students want to use WebSOC to find their courses.

This leads to two major issues:

- Manual copying is error prone.
- Registering for classes is an involved and time consuming process.

Error Prone Input from WebSOC

The course codes are only found as plaintext that students must copy themselves, forcing them to manually save specific numbers in a list of results over the course of multiple searches. Manual effort like this introduces many possible points of error. A student might highlight the wrong row of the table, and copy paste a code for a different section or course entirely. Or they might fail to highlight the entire code and only copy a part of what they need to save. These flaws in WebSOC have already led UCI students to turn to external tools to make the registration process easier for themselves, as obvious in the student complaint in Figure 1.

AntAlmanac has many features that WebSOC doesn't have for the convenience of students, like being able to save searched classes and providing one-click copying of course codes. This reduces the number of mistakes students make, as each saved course can have its respective code copied with much less risk of error.

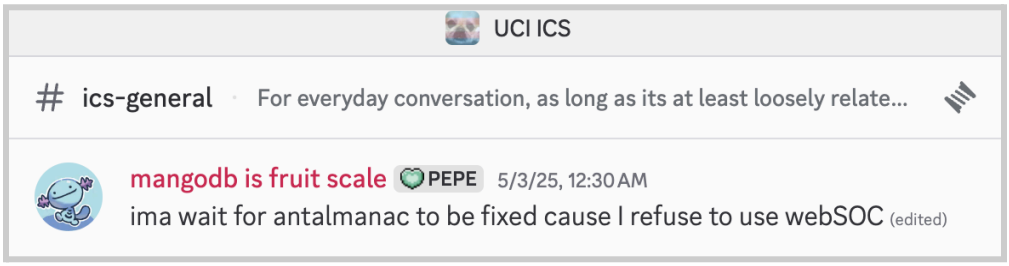

Figure 1. *An example of a student expressing frustration with the current state of WebSOC, choosing to use other tools like AntAlamanc instead (SoftLocked 2025)*

System Congestion for Students and Staff

The additional steps required to go from WebSOC to registration on WebReg creates unnecessary friction. A student who has to find classes again during their registration window must make multiple searches on a different site, copy the course code, and then come back to WebReg. Even students who plan ahead and save their courses in advance must go back and forth between their saved plan and the WebReg site to register, meaning that there is still time being wasted.

This wasted time accumulates over the thousands of students at UCI, which puts additional strain on the system, as students spending more time registering means more of them must access the system concurrently. This also causes registration windows to be smaller, since they must be planned around avoiding having too many students access WebReg at the same time. During peak Webreg usage times such as school restriction lifts or open enrollment, many students struggle with even accessing the site. Oftentimes it takes multiple attempts to log into the website, as WebReg quickly reaches its maximum user limit, leaving students frustrated and worried if they will be able to enroll in their needed classes.

If students spent less time registering, more could access the system in the same amount of time. This makes registration for students a worse experience, as more of them have to deal with late registration windows that make it harder for them to get the courses they need.

This additional strain also causes more pressure on registrar staff. WebReg has to be kept up as often as possible with little downtime in order to accommodate every student's registration windows.

Other universities, even ones within the University of California system, have better integration between their versions of WebSOC and WebReg to alleviate these issues. For example, as seen in Figure 2, UC Riverside's schedule of classes and registration are accessible in the same location, and they also provide a class planner for registered students to create term plans. These plans can be loaded directly when registration opens, vastly shortening the time required for students to register for their classes.

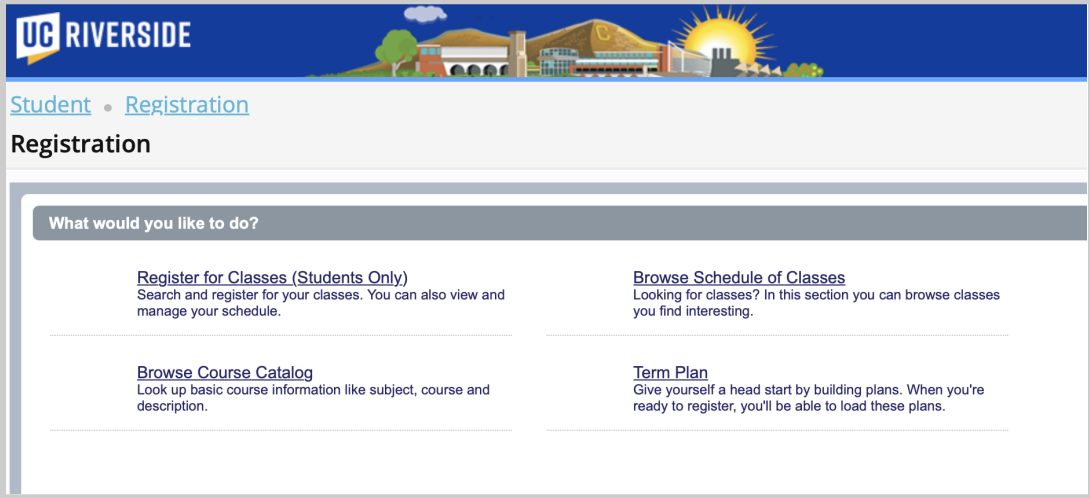

Figure 2. *UCR's Registration Page. Note the Term Plan Feature. (University of California Riverside)*

Our Proposal:
Integrating WebSOC and WebReg into a Single Workflow

Right now, students must switch between WebSOC and WebReg to find/register for classes, manually copy course codes, and keep track of conflicts on their own. Our solution creates one integrated workspace where students can:

- Search for classes in WebSOC
- Log into WebReg directly from WebSOC
- Click [Add to Schedule] under any course in WebSOC to build a tentative schedule
- Use a [Show Schedule] button in the WebReg side panel to see all tentatively added classes in either a calendar view or a list view
- Save draft schedules and, when their window opens, register everything with one click

The following steps describe exactly how this integrated system will work from a student's point of view and what we need to build at each step.

Step 1: Add a "Log in to WebReg" Panel Inside WebSOC

What the student sees:

1. The student goes to WebSOC as usual and selects filters (term, department, GE, etc.).
2. A new [Log in to WebReg] button appears in the top menu bar and remains visible on both the search and results pages.
3. When the student clicks this button, a WebReg panel opens on the right side of the screen (or as a fixed sidebar), without leaving WebSOC.
4. The panel prompts the student to sign in with their UCInetID and authenticate using DuoMobile, using the existing SSO system.

System changes required:

- Add a persistent [Log into WebReg] button to WebSOC's header.
- Embed WebReg inside WebSOC using a secure, authenticated panel.
- Ensure that, once logged in, the student stays authenticated as they navigate between WebSOC pages.

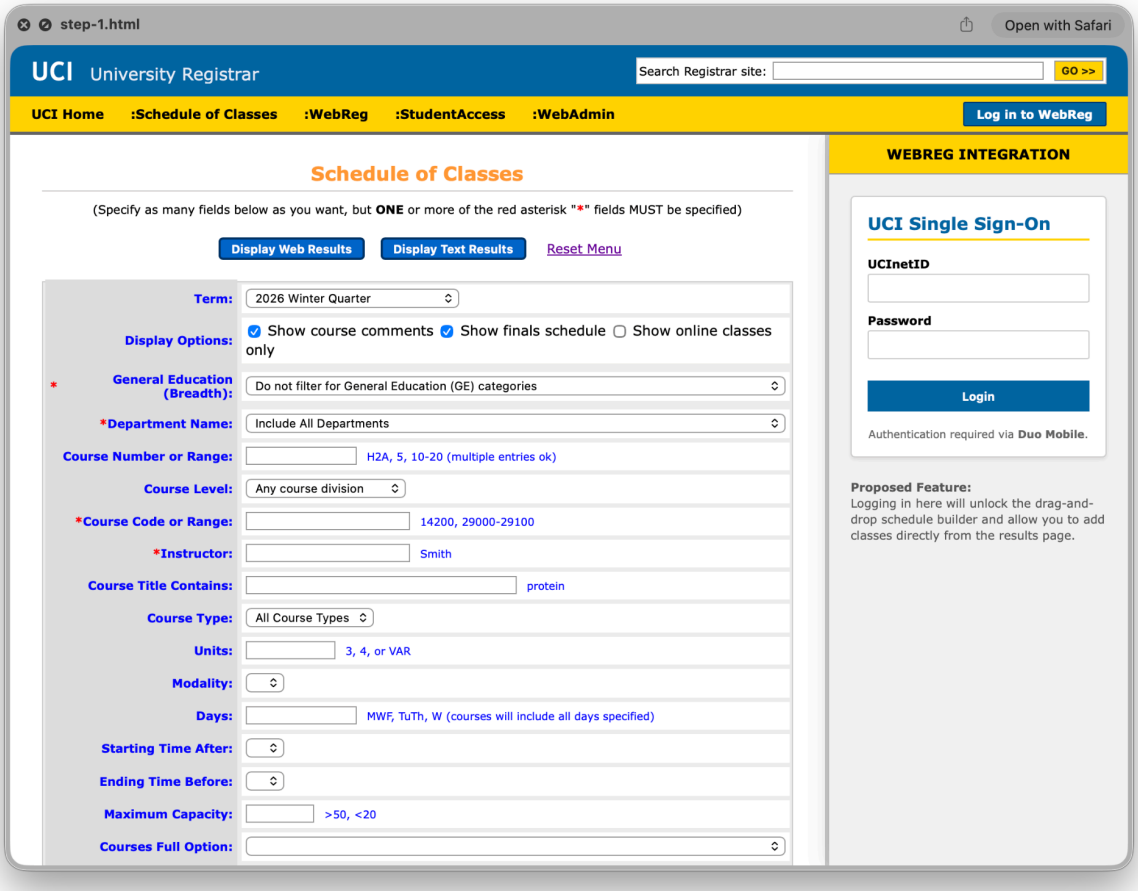

Figure 3. *Step 1 – A diagram of the WebSOC results page with a new [Log in to WebReg] button in the top navigation and a blank sidebar area on the right labeled "WebReg panel."*

Step 2: Add an [Add to Schedule] Button Under Each Course in WebSOC

What the student sees:

1. After running a search, the student sees the usual list of classes for the quarter.
2. Each course section (with course code, title, instructor, day, time, location, etc.) is displayed as a clean block or card.
3. Under the course information, there is a clear [Add to Schedule] button.
4. When the student clicks [Add to Schedule]:
    a. A short conformation appears (for example, a checkmark and "Added to

schedule”).

b. The button changes state to something like [Added] or [Remove from Schedule] so students can see what they’ve already selected.

System changes required:

- Keep the existing WebSOC search results but reorganize each course section so there is space for an [Add to Schedule] button.
- Implement a simple action that, when clicked, sends the selected course;s information (course code, meeting time, etc.) to the student’s tentative schedule in WebReg.
- Optionally, change the button state to show that the course has already been added, and allow toggling it off from WebSOC.

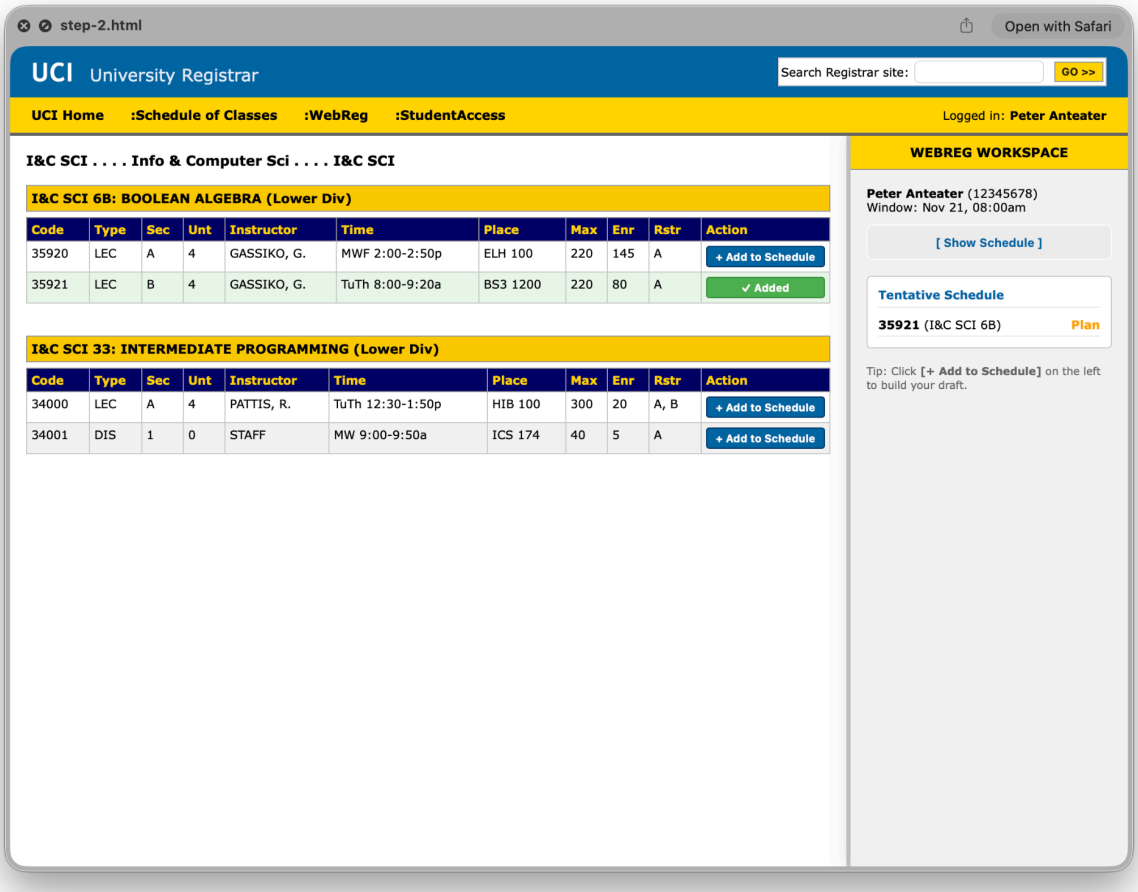

Figure 4. *Step 2 – A results page showing each course section with its usual information and a clear “Add to Schedule” button underneath the details.*

Step 3: Add a [Show Schedule] View (Calendar or List) Inside the WebReg Panel

What the student sees:

1. Once logged in, the WebReg panel shows a [Show Schedule] button.
2. When the student clicks [Show schedule], the panel displays their tentative schedule, made up of all the classes they clicked [Add to Schedule] on in WebSOC.
3. Students can choose between two display options:
    a. Calendar view: a weekly grid (e.g. Monday-Friday, 8am-10pm) with classes shown as blocks.
    b. List view: a simple list of all selected classes with course code, title, days, time, and location.
4. The student's currently enrolled classes are also visible as colored blocks, labeled with course code and location.
5. Below the schedule is a small legend explaining colors:
    a. Green = Enrolled
    b. Yellow = Planned (not yet registered)
    c. Red = Conflicting or unavailable sections

System changes required:

- A [Show Schedule] button inside the WebReg panel that opens the schedule view.
- Two view modes for the same underlying data:
    - A weekly calendar layout.
    - A compact list layout.
- A color coding of schedule entries that provide visual differentiation between:
    - Classes already enrolled via WebReg
    - Classes tentatively added via [Add to Schedule] in WebSOC.

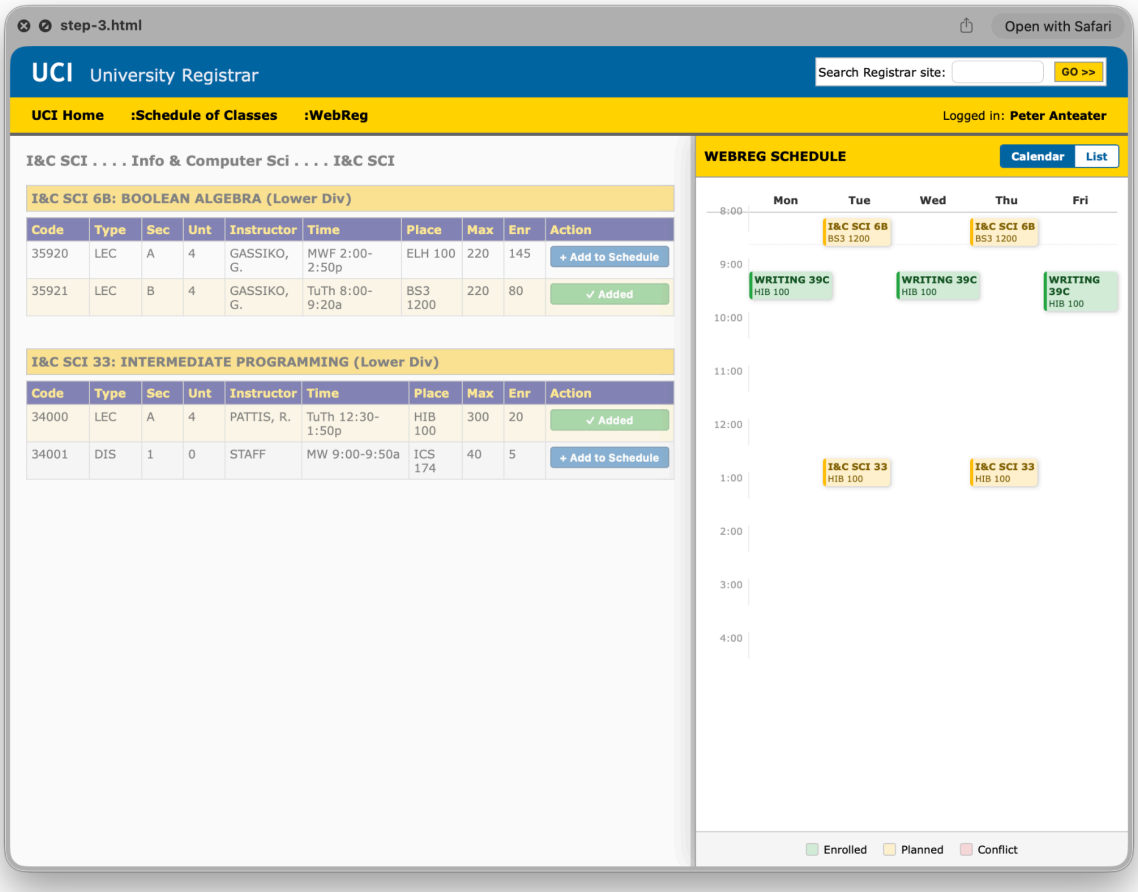

Figure 5. *Step 3 – A right-hand WebReg panel with a [Show Schedule] button at the top and a toggle for calendar / list view above the schedule display.*

Step 4: Let Students Manage Their Tentative Schedule Inside WebReg

What the student sees:

1. When the student opens [Show Schedule], they see all classes they've added via [Add to Schedule] in WebSOC.
2. Under each class in the schedule view (both calendar and list formats), there is a [Remove from Schedule] button.
3. When the student clicks [Remove from Schedule]:
    a. That class disappears from the schedule view.
    b. Optionally, the corresponding course in WebSOC updates so its button returns to [Add to Schedule] instead of [Added].
4. If a course conflicts in time with another selected or enrolled course, the

schedule view clearly flags it (Example: red border or icon and short text like [chosen class] “conflicts with I&C Sci 33”)

System changes required:

- A “tentative schedule” data structure that stores the list of courses the student has added.
- A [Remove from Schedule] button next to each class in the schedule view that updates this list.
- Logic that synchronizes state to ensure WebSOC and WebReg show consistent information about what is in the schedule.

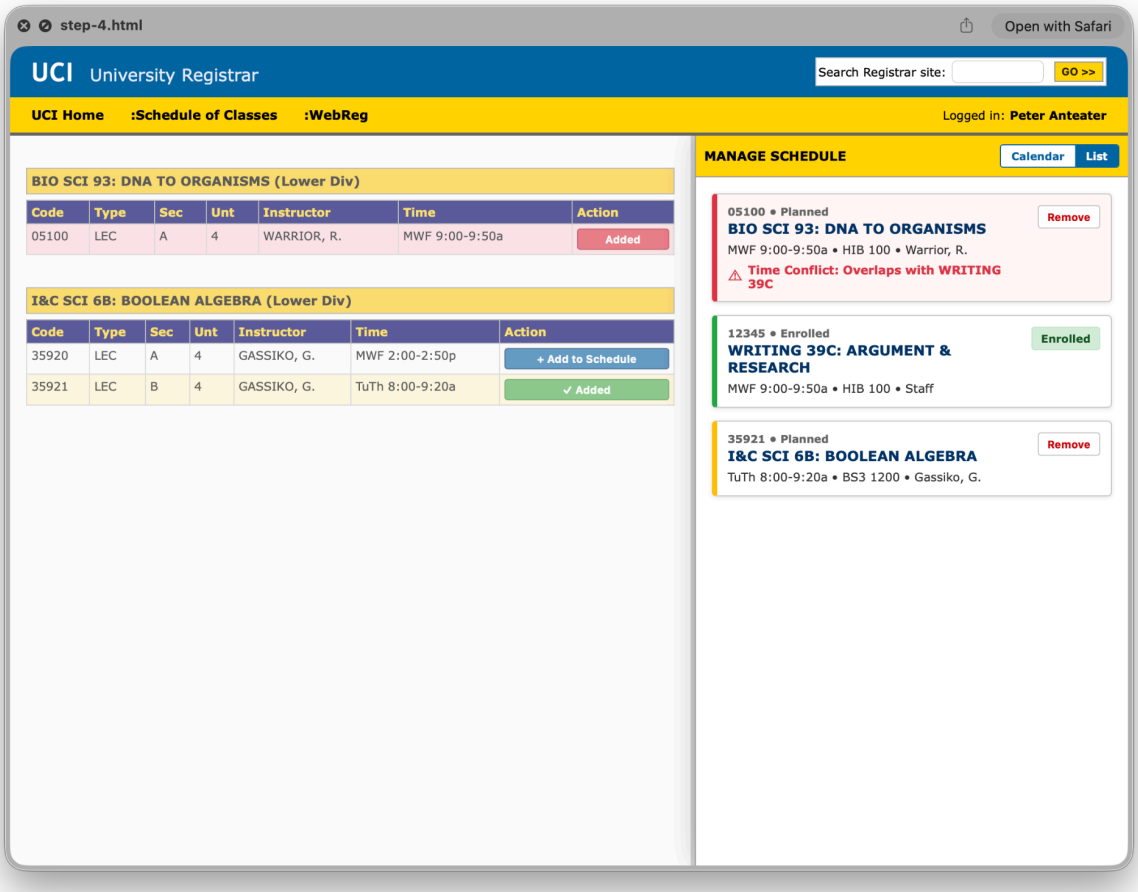

Figure 6. *Step 4 – A schedule view showing several classes, each with a small [Remove from Schedule] button under the details and a warning icon next to any conflicting class.*

Step 5: Allow Students to Save, Load, and Compare Draft Schedules

What the student sees:

1. As they plan, by clicking [Add to Schedule] in WebSOC and managing classes in the [Show Schedule] view, students can click [Save Draft Schedule] in the WebReg panel.
2. The system prompts them to name the schedule (e.g., “Winter 2026 Plan A”).
3. Later, they can click [Load Draft Schedule] to restore a previously saved plan to the calendar.

System changes required:

- A [Save Draft Schedule] function that stores:
    - The list of planned course sections.
    - Their statuses (open / waitlist / full when last checked).
    - The layout for the current quarter.
- A [Load Draft Schedule] function that restores these planned blocks onto the calendar.

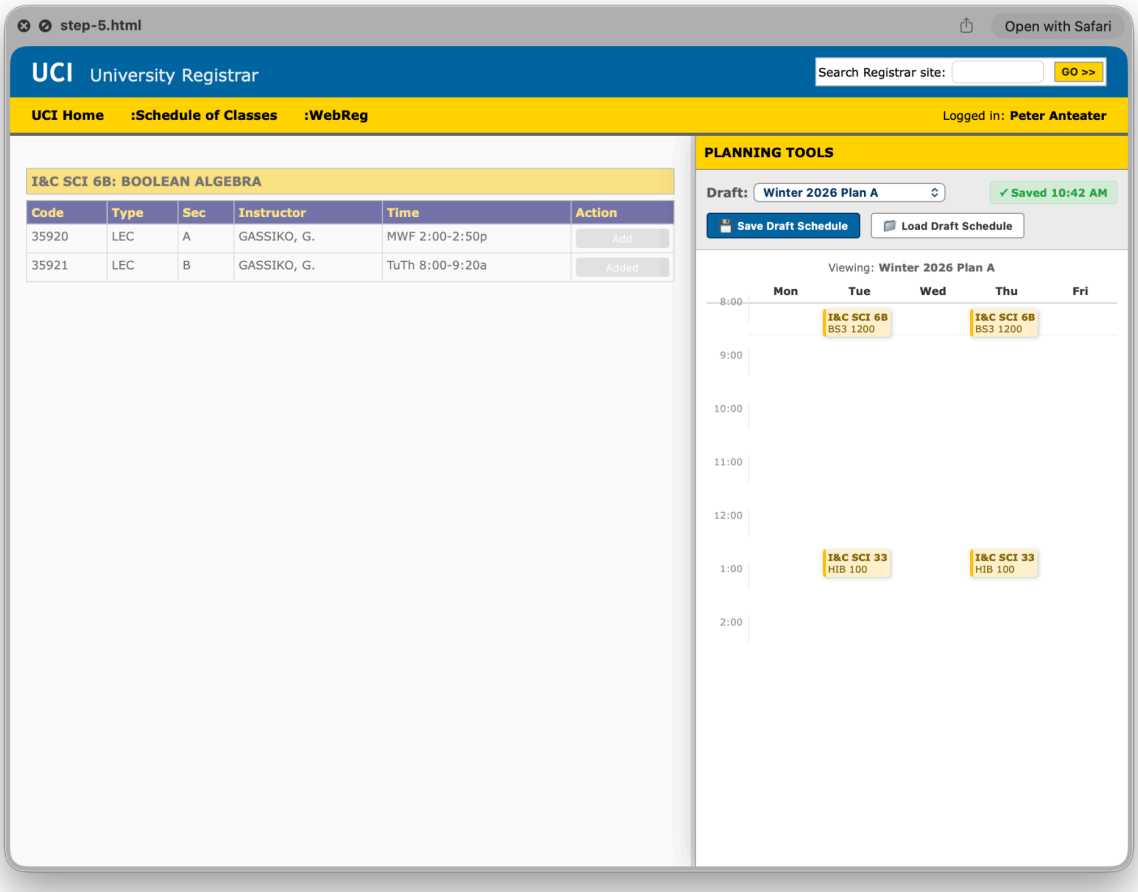

Figure 7. *Step 5 – A calendar view with a small panel showing saved schedule names and buttons labeled [Save Draft Schedule] and [Load Draft Schedule].*

Step 6: Enable One-Click [Register Schedule] When the Window Opens

What the student sees:

1. Before the student's registration window opens, the [Register Schedule] button is visible but is disabled and labeled with the date/time of their window.
2. Once their registration window opens, the button turns active and clearly indicates that they can now register their planned schedule.
3. When the student clicks [Register Schedule]:
    a. The system attempts to register each planned course currently in the student's tentative schedule (i.e., everything they added using [Add to Schedule]).

b. A progress bar or list shows each course as “Registering...”, “Enrolled”, “Waitlisted”, or “Failed.”

System changes required:

- Reuse the existing WebReg registration logic but provide a batch submission that:
  - Processes each course sequentially to avoid overloading the system.
  - Handles waitlist rules and prerequisites just as standard WebReg does.
- A progress interface that clearly lists each attempted section and its results in real time.

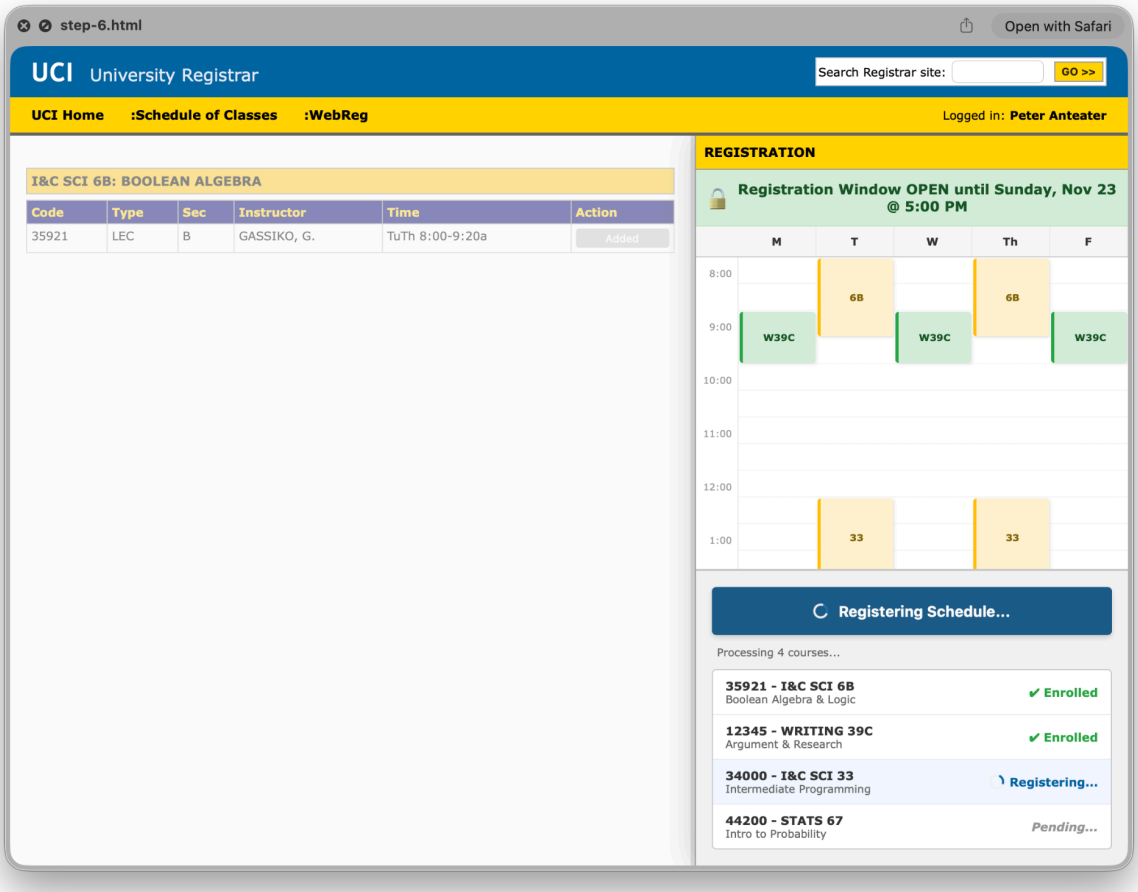

Figure 8. *Step 6 – A calendar with a visible [Register Schedule] button below it, plus a status list showing each course with a checkmark for success or an icon for errors.*

Step 7: Provide Clear Feedback, Next Steps, and Export Options

What the student sees:

1. After the registration attempt finishes, the student sees a summary panel with three columns:
    a. Courses successfully enrolled
    b. Courses waitlisted
    c. Courses that could not be added, with the reason (e.g., full, time conflict, prerequisite not met)
2. Each problem entry includes a short suggestion, such as:
    a. “Try a different discussion section.”
    b. “Contact your academic counselor about prerequisites.”
3. The student can then:
    a. Export their final schedule as a PDF, CSV, or calendar file (.ics).
    b. Print the schedule for in-person advising.
    c. Return to WebSOC results to adjust any problem courses.

System changes required:

1. A clear “Registration Results” screen embedded in the WebReg panel that:
    a. Lists all courses attempted and their final status.
    b. Provides user-friendly explanations and next steps.
2. Export tools that produce:
    a. A PDF with a clean weekly calendar.
    b. A CSV file listing course codes, titles, and meeting times.
    c. A calendar file that can be imported into common calendar apps.

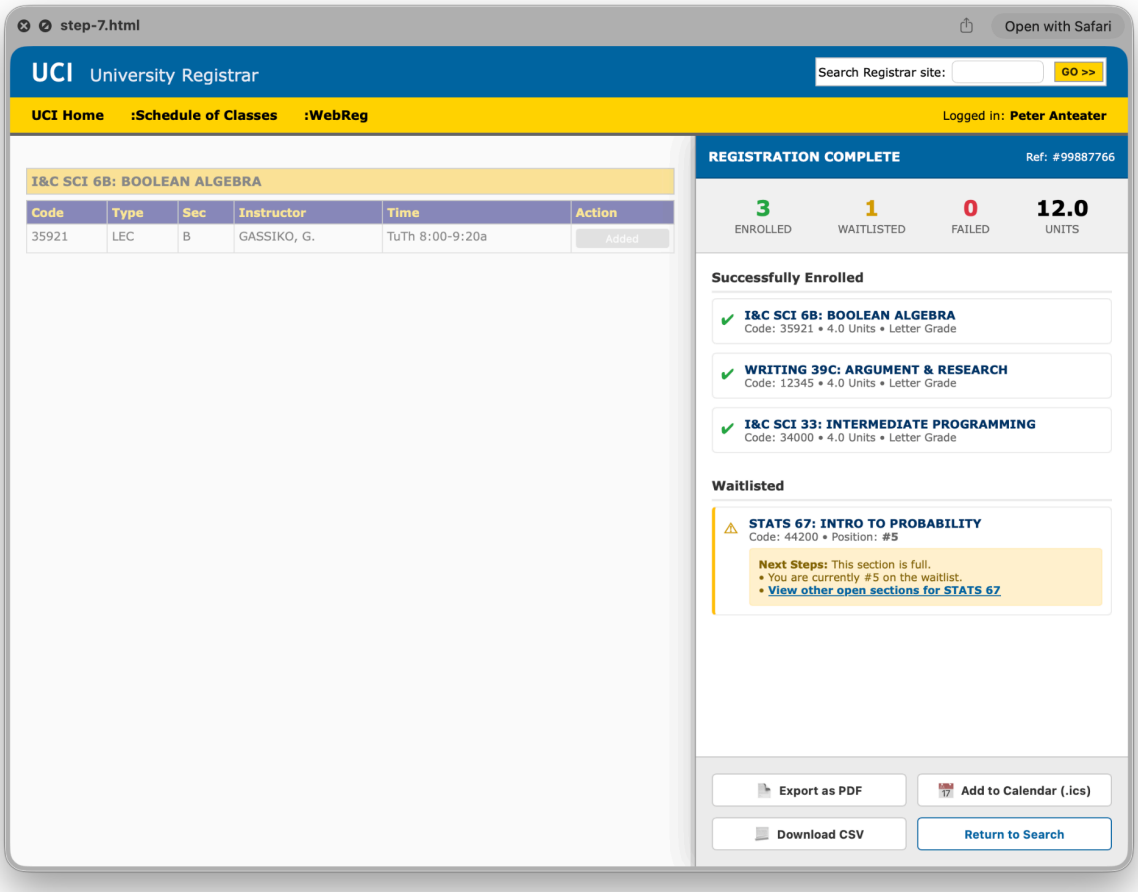

Figure 9. *Step 7 – A panel showing three lists (Enrolled, Waitlisted, Failed) plus buttons labeled [Export as PDF], [Download CSV], and [Add to Calendar (.ics)].*

Step 8: Maintain Accessibility, Mobile Usability, and Fallback Paths

What the student sees:

1. Students using screen readers or keyboard navigation can still:
    a. Use the “Add to Plan” button on each course.
    b. Navigate the calendar via keyboard controls (e.g., arrow keys to move between time slots, Enter to add/remove).
2. On small screens (phones and tablets), the system automatically switches to a stacked layout:
    a. WebSOC search results appear above the WebReg calendar, with a toggle to switch between them.
3. If there is any problem with the embedded panel (e.g., older browser), a clear

fallback link lets students open WebReg in a separate tab, so they are never blocked from registering.

System changes required:

- WCAG-compliant focus order, ARIA labels, screen reader accommodations, and keyboard shortcuts for all new controls.
- A responsive layout that reorganizes WebSOC + WebReg into a mobile-friendly vertical flow.
- Fallback behavior that preserves existing WebReg functionality if the embedded view fails.

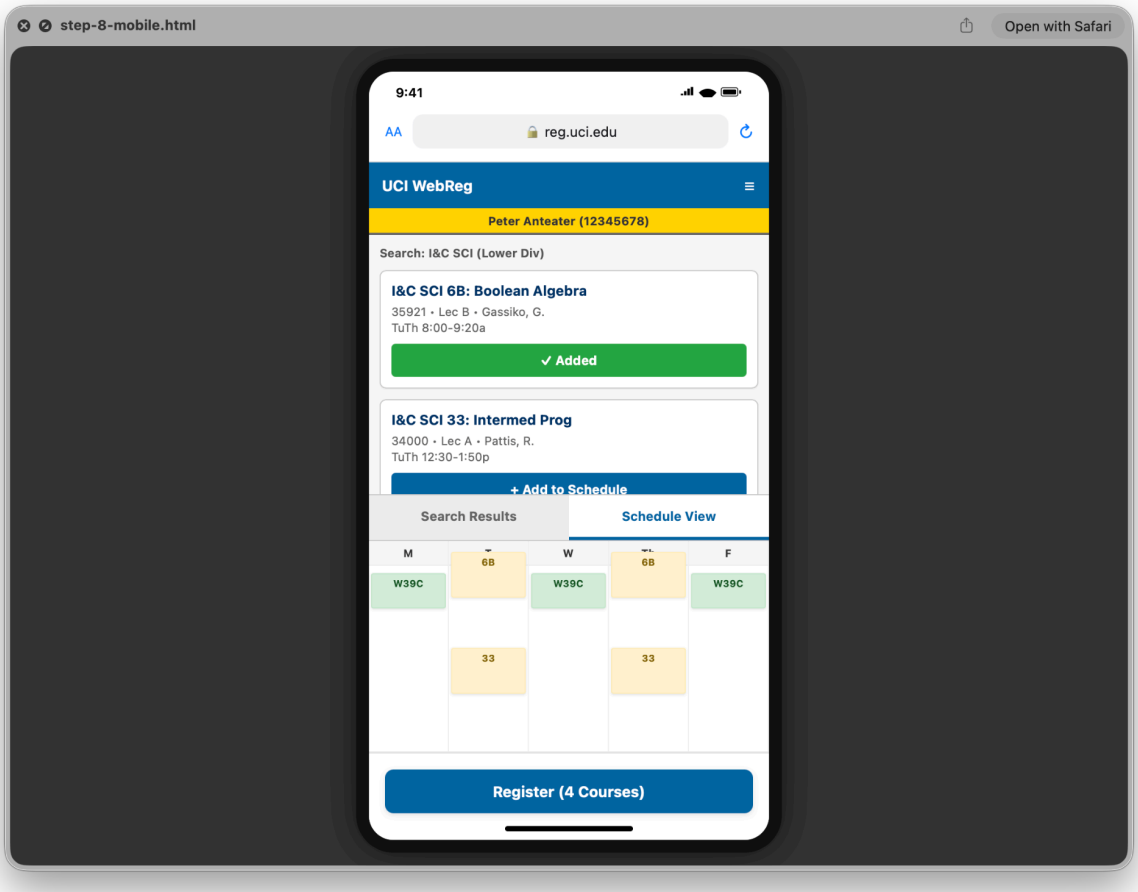

Figure 10a. *Step 8 – A diagram showing a mobile view with WebSOC above a simplified calendar.*

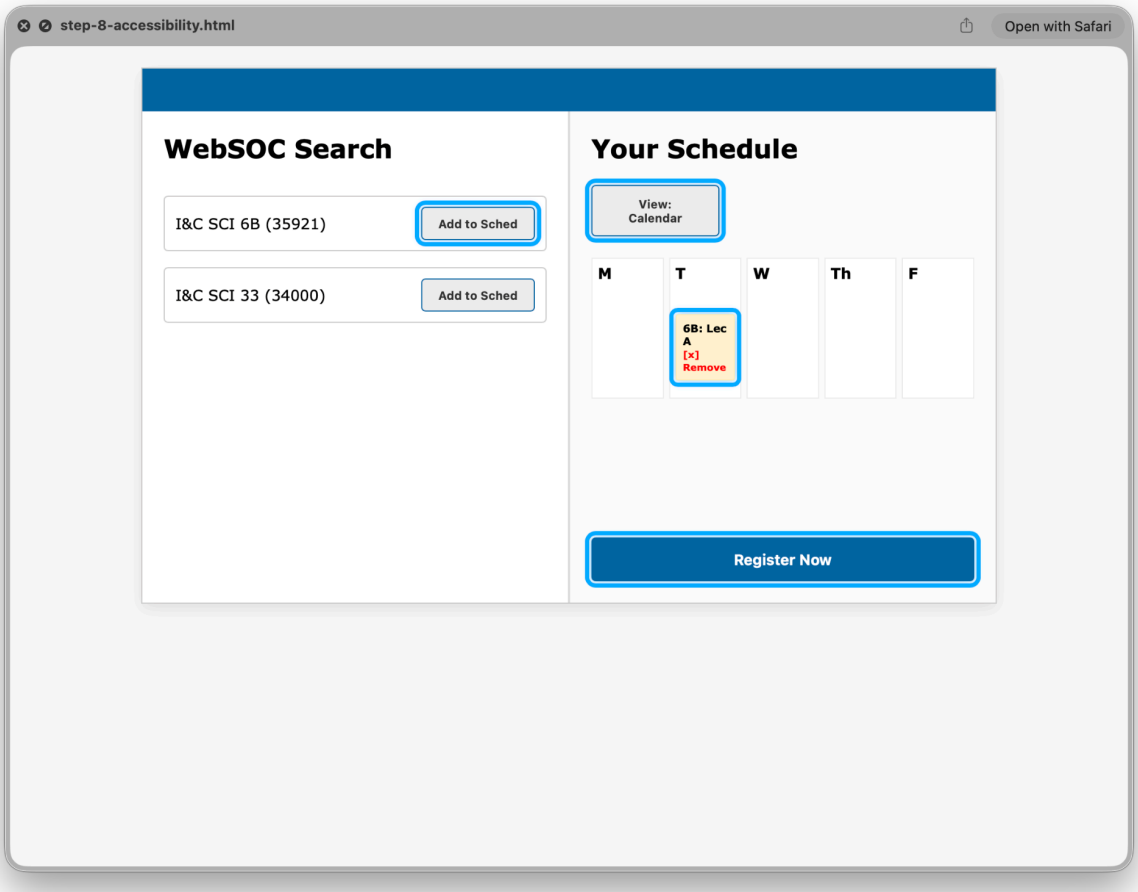

Figure 10b. *Step 8 – A diagram showing the keyboard focus path on a schedule and calendar cell.*

Logistics

Budget

Designing, developing, testing, and deploying an integrated WebSOC–WebReg system requires a cross-functional team and additional server capacity during peak registration periods. The following budget outlines estimated salary costs for hiring the necessary personnel for the duration of the project, as well as infrastructure and operational expenses. All salaries are based on publicly available Glassdoor and salary-aggregation data.

The project requires a small engineering and design team, along with QA and project oversight. Salary estimates are based on publicly available Glassdoor averages for comparable UC system technical roles.

Role	Headcount	Approx. Monthly Salary
Software Engineer	12	$15,000
UI/UX Designer	2	$10,000
QA Engineer	3	$9,000
Security Engineer	1	$16,000
DevOps Engineer	1	$14,000
Project Manager	1	$13,000
Product Manager	1	$14,000

In total, this project will take place over 4 months according to the timeline outlined below.

Cost Calculation (3 months):

Role	Headcount	Monthly Salary	Months (multiplier)	Total Cost
Software Engineers	12	$15,000	4	$720,000
UI/UX Designer	2	$10,000	4	$80,000
QA Engineer	3	$9,000	4	$108,000
Security Engineer	1	$16,000	4	$64,000
DevOps Engineer	1	$14,000	4	$56,000
Project Manager	1	$13,000	4	$52,000
Product Manager	1	$14,000	4	$56,000
Total Personnel Cost				**$1,136,000**

Timeline

We will be using the Waterfall development model to help stay on track with our current timeline and ensure all components are completed and successful before moving onto the next component.

The Waterfall development model is a linear approach to project management, ensuring that each task or goal is completed before moving on to the next. It is mainly used for large and intricate projects relating to software development. The Waterfall development model is more structured than other development models, which makes estimating total budget and time easier. This model allows us to define milestones to ensure that we are pacing this solution correctly. The linear approach of the Waterfall model matches perfectly with our solution's flow, as once we complete a step, we do not need to revise or change it later. This is due to the extensive planning documentation we have created beforehand, acting as our guide. Instead of revisiting previous steps, we spend the time to make sure they are right before moving on, that way we can use details from these steps in later stages. By implementing the Waterfall development model, we save time, money, and resources by eliminating unexpected errors and ambiguous goals by clearly defining milestones and ensuring each phase is perfected before moving on to the next.

Enacting the integration between the WebSOC and WebReg should take

approximately 16 weeks with the following timeline:

- Week 1-2: Gather requirements, define new features, and select API that would work best
- Week 3: Prototype UI of what new changes would look like
- Week 4: Implement embedded login pathway
- Week 5-8: Implement features as outlined in steps above
- Week 9-10: QA Testing
- Week 11-12: Small group testing with actual students (honors/sport students who normally apply earlier)
- Week 12-16: Further debugging given feedback, ideally a full release to the public before fall quarter

Given the importance of both WebSOC and WebReg by themselves, the recommended date to implement these changes would be between June and September. During this time, less students would be impacted from any downtime. UCI hosts SPOP during this period, with 3-day enrollment periods happening each week for roughly 4 hours. Because of this, we recommend that the website should only go down for full maintenance during non-peak hours 7pm-7am during SPOP sessions.

UCI's Summer Session students would not be impacted by this downtime as a different site is used.

Quality Assurance Testing

QA Testing will test the following:

1. Correct functionality: Users have the ability to add/drop/waitlist classes listed on the WebSOC website. Users can complete tasks using both course code implementation and drag and drop method.
2. Integrity: Student records are properly updated when changes to classes are made.
3. Usability: WebSOC is easily navigable and errors are easy to understand. WebSOC complies with accessibility standards outlined in WCAG and is screen reader compliant.
4. Security: Only UCI students have access to add/drop/waitlist classes. Students can only make changes to their own classes and not to others.
5. Performance: Integration can handle load during peak registration (around 10,000 students on at once), number of students enrolled/spot on waitlist for

each course is updated in real time and is accurate. Pages load quickly and requests are completed promptly.

For QA testing, we plan to focus on ensuring that SSO and login information is secure and accurate, that classes are correctly changed based on selection, and that this website is capable of running during high volume times. The security will be tested by creating fake student accounts and attempting to login to WebSOC, ensuring compatibility with DUO Mobile for two-step authentication, and testing that logging in with the fake information bars access from the site. To test the integrity of the website, we will add/drop/waitlist classes using the course code method and drag and drop method and ensure that those changes are properly documented in that account's student record. For performance, we can simulate high traffic to the site and certify that the site is able to be accessed by at least 10,000 students at once to simulate course restriction lift days.

For small group testing, we plan to release and test these changes with a small group of real UCI students, focusing our testing on UI and usability. The data gathered here will allow us to make any changes necessary and reintroduce WebSOC to these students until we get a 90% approval rating/90% of students in this group are able to navigate the website with ease. After reaching this goal, we will roll out our product incrementally starting with honors/sports/students who get priority registration, then release it to the entire student population. Any bugs or errors found after this stage will be reported to the project team and scheduled maintenance will be completed, with care that this maintenance does not occur during high volume times (major restriction lift days, registration weeks for next quarter's classes, etc.).

Conclusion

Given the current inefficiency in WebSOC and WebReg compatibility along with examples of other UC schools' more user friendly scheduling software, UCI's scheduling system is due for reform. With these changes students can plan for classes at ease without having to resort to third party software or wasting time. While overhauling a UCI's scheduling system will be a lengthy process, following our projected schedule it can be completed and tested before fall quarter, allowing for a natural transition into the next school year and disrupting fewer students than other timelines would. As registration is a vital process in allowing students to sign up for the courses needed for their academic success, it is imperative that UCI invest in this integration now. Implementing these changes will create a unified, efficient, and accessible system that better serves both students and staff.

Works Cited

APA Citation Format

Softlocked. (2025, May 3). *ima wait for antalmanac to be fixed cause I refuse to use webSOC* [Message in online forum channel #ics-general in UCI ICS server]. Discord. https://discord.com/channels/532433255371243531/590407931518910474/1368127570033508352

University of California Riverside. (n.d.). *Registration*. Student Registration. https://registrationssb.ucr.edu/StudentRegistrationSsb/ssb/registration

(WAI), W. W. A. I. (2025, October 20). *Web Content Accessibility Guidelines*. Web Accessibility Initiative (WAI). https://www.w3.org/WAI/standards-guidelines/wcag/

Salary.com. (2024). *California Technical Salary Benchmarks* (Software Engineer, UX Designer, QA Engineer, Security Engineer, DevOps Engineer, Product Manager). Salary.com. https://www.salary.com/research/salary

University of California. (2024). *UC Information Technology Job Postings* (Salary ranges for Software Engineering and Security roles). UC Careers Portal. https://jobs.universityofcalifornia.edu/