



TigerSnatch: Product Evaluation

Table of Contents

[Testing](#)

[User Evaluation](#)

[Expert Evaluation](#)

[Appendix: Beta Test User Tasks](#)

Testing

As of May 6, 2021, there are no known bugs in any part of the TigerSnatch system. All features, both user-facing and admin-facing, behave as intended (there are certain out-of-our-control fragilities discussed later). As a team, we carefully designed the site's frontend and backend infrastructures to be robust against typical and atypical usage. TigerSnatch also provides an Admin panel that allows site administrators to protect the site against malicious users and safely modify/reset sections of the database in the extreme case of corruption. In other words, TigerSnatch has built-in safeguards to protect site integrity, as well as tools to facilitate automatic and manual recovery. Below is a (non-exhaustive) list of important robustness features:

- All backend methods that write to the database contain extensive validity checks that protect against intentional malicious activity by users (e.g. modifying delivered JS or HTML code) or against unnoticed bugs in any TigerSnatch code.
- All exceptions and errors (both Python and HTTP) are caught and generate a styled error page. The error along with detailed information (e.g. specific endpoint, time, category, type) are written to system logs in case we - or future maintainers - need to fix or review an issue. We have designed TigerSnatch such that it is highly resistant (i.e. all exceptions and errors are caught and logged) to issues caused by users or the code itself.
- We have taken steps to prevent users - in a friendly way - from spamming frontend buttons and switches that result in writes to the database. For instance, we have implemented useful modals and confirmation dialogues whenever a user is about to perform an action causing changes that persist - either temporarily or permanently - across all TigerSnatch systems. Additionally, we have implemented UI-disabling functionality such as partial screen blurs, wait cursors, and button and switch locks whenever a user performs an action that may take a bit of time to process, e.g. clicking on a course search result that causes a MobileApp query or finding Trades. In this way,



we protect the TigerSnatch backend from an excessive amount of database writes by the same user.

- TigerSnatch is also robust against injection attacks in the form of HTML scripts, SQL commands (we use MongoDB, so a SQL injection attack is impossible), long queries, or queries with non-accepted characters. We have backend protections against HTML script injection, and both backend and frontend guards against long or abnormal string queries (e.g. during course search, or email address change).
- The system is robust against frequent course page visits. As discussed in Design Problem 1 in the Programmer's Guide, TigerSnatch performs a MobileApp query if a user visits a course page and it has been more than 5 minutes since any user has visited that course page, since it's expensive to update the course's data every single page visit. After the MobileApp query, new course data are inserted if a change is detected compared to the current entry in the database, but during this time, both the site and the course-specific page remain fully responsive to other users' interactions. In between 5 minute intervals, course data are retrieved instantly from our database and do not cause MobileApp queries.
- TigerSnatch provides solutions to administrators in the case of database corruption at the section level, course level, Subscriptions level, Trades level, or even the entire database level. In particular, the web-facing Admin panel allows administrators to safely reset or update specific sections of the database. For example, TigerSnatch offers a single-button solution to updating the entire system to the latest course term, decreasing the chance that a future site maintainer unintentionally corrupts the database (e.g. forgetting to clear users' Subscriptions or current enrollments) during this necessary site-wide change.
- In general, the Admin panel increases robustness in that it gives administrators a way to manage the system without giving them direct access to our source code and database and without requiring any potentially-dangerous shell usage, minimizing the risk of severe system corruption or failure.

TigerSnatch is fragile, however, in ways that are not within our control. Below, we discuss several of such fragilities as well as the steps we have taken to mitigate them.

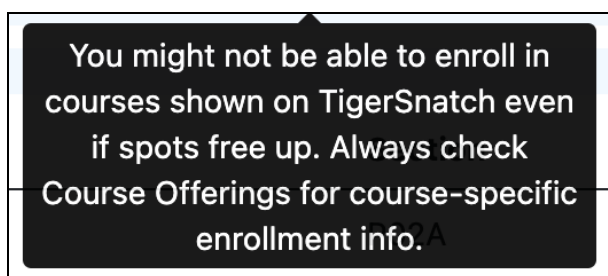
- TigerSnatch does not have access to reserved seating data from PeopleSoft/TigerSnatch

TigerSnatch is unable to directly check for grade-specific reserved spots (e.g. only 30 spots for juniors, etc.). For instance, if COS126 L01 reaches capacity for only junior reserved seats (i.e. it still has slots open for other class years), Subscriptions would still not be enabled for COS126 L01 because from the enrollment numbers alone, COS126 L01 has not reached capacity. Although these reservation data are available via the Course Offerings website, they are

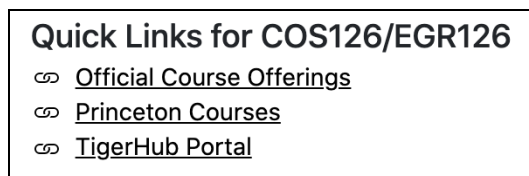


provided in inconsistent formats that could cause issues with web scrapers. As a result, it would be extremely challenging to separate Subscription eligibility by class year. If we were able to get grade-specific reserved spots in a standard data format, we would likely ask users to self-declare their class year so that, going back to the COS126 L01 example, only juniors could Subscribe to COS126 L01 notifications.

We attempt to mitigate this issue by placing disclaimers/warnings throughout the site, such as on the Landing, Dashboard, and Tutorial pages, and trust that users will not Subscribe to a section they are not eligible to enroll in (students tend to take course enrollment seriously). These disclaimers contain content similar to this:



In addition, we provide the link to each course's official Course Offerings page in Quick Links so that users can more easily check the Registrar's for enrollment requirements.



- TigerSnatch does not know whether a course has selective enrollment criteria (i.e. students cannot simply click "Enroll" in TigerHub)

TigerSnatch is unable to directly check whether a course admits students by application/interview only, or whether a course has restrictions on which students can enroll (e.g. COS333 requires that students be a COS major). These data are available via the Course Offerings website, but again they are provided in inconsistent formats that would make web scraping difficult and error prone. As a result, users can Subscribe to any full section in any course in TigerSnatch, even if they are not officially eligible to enroll in that course if a spot opens up. If we were able to get these data in a consistent data format, we would likely simply not display and/or remove courses in which enrollment is selective or by application only.

We attempt to mitigate this issue in the same way as in the previous disclosed fragility (disclaimers/warnings in multiple places on the site and Quick Links to Course Offerings).



- TigerSnatch does not have access to non-anonymized enrollment data (e.g. course rosters) from PeopleSoft/TigerHub

TigerSnatch is unable to verify that users are indeed enrolled in the sections in which they self-declare enrollment. This affects only the Trades feature. As a result, users can be dishonest about their current enrollment and lie to other users by contacting them about potential Trades, when in reality no such Trade exists.

We attempt to mitigate this issue by providing a pre-filled email that must be sent from the user's *own* email address. This creates accountability for a user wanting to Trade sections with another, since the user on the receiving side knows who the initiating user is. We believe that users are less likely to lie about their current enrollment status if they must click the send button upon an email and their own name is attached to the email message.

Additionally, as a last resort, we have implemented a blacklist feature that allows admins to ban a specific user from accessing any CAS-protected page in the TigerSnatch system if, for example, that user has abused the Trades system. Users may be unbanned too.

- TigerSnatch does not have a way to verify whether a user has in fact clicked "Send" on the email generated by initiating a Trade

The moment an email is generated through the Trades feature (i.e. when the user clicks the "Email" button for a match in the Trades pop-up), TigerSnatch adds an entry in both users' Activity pages. For the person on the receiving end of the Trade, the entry is:

May 03, 2021 @ 2:30 PM ET → sheh contacted you about swapping into your section COS126 P01B

Unfortunately, we have no way to check whether the user who initiated the Trade has really sent the email generated by TigerSnatch. As a result, we are unable to implement functionality that adds the Activity entries only if the email actually was sent. Thus, we reasonably assume that if a user goes through the multiple steps required to generate the email, they have the intention to send it.

We have taken steps to mitigate this issue mainly through the use of (very) clear warnings and a confirmation dialogue explaining what will happen if the user generates an email:



Potential Trades

⚠ Clicking 'Email' will add an entry for you & the match in Activity.

NetID	Current Section	Contact
sheh	P02A	✉ Email

tigersnatch.herokuapp.com says

Are you sure you want to email this user? They will be notified on their Activity page if you confirm!

[Cancel](#)[OK](#)

If a user still goes through with generating the email, we assume that they will click "Send" on the Gmail message and the generated Activity log entries will be accurate.

- TigerSnatch relies on OIT's MobileApp API for all course-related data

Because TigerSnatch relies on MobileApp to not only provide data, but also transmit it in a specific format, any structural changes to the API would cause fatal errors in the TigerSnatch system (e.g. course data could not be updated, the site would be unable to update to the latest term, and open slots could not be determined). Additionally, if TigerSnatch's subscription to the API were to be revoked by OIT, almost all core functionality would stop working.

It is very difficult to mitigate this fragility using warnings; changes to MobileApp would require us to modify the actual backend code to adjust to the changes. It is worth mentioning that we have already met with OIT - per their request - to discuss the needs of TigerSnatch and its reliance on MobileApp. Hopefully, OIT would give us (or whoever maintains TigerSnatch in the future) ample notice of any API changes so that TigerSnatch would not temporarily go down, especially during the critical period of course enrollment and add/drop periods.

In testing our product, we practiced both internal and external testing. For internal testing, we apply the following major techniques (a non-exhaustive list):



- All error-prone (i.e. may result in an auto-generated or a custom Exception) code is wrapped in try-except blocks, and all thrown Exceptions are caught as early as possible but are always eventually reconciled as to prevent fatal crashes.
- All methods, especially those that perform write operations on the database, contain thorough parameter validation if there is any potential for corruption. For example, when the system adds a user to a Subscription list for a section, it verifies that the passed-in classID is valid, that the user has an existing profile, that the user has not surpassed their Subscription limit, that the user is not already Subscribed to the section, that the section is indeed currently full, and several other checks.
- Each time a Database object is created and a MongoDB connection is established, the code performs a basic database integrity check to make sure that all collections - and only those collections - exist.
- Each time documents are inserted into the database, such as during the runtime of the algorithm that updates TigerSnatch to the current course term, the code performs invariant checks to ensure that inserted data are in the correct format.
- We utilized unit testing in all core Python code files to ensure proper functionality with general cases and edge cases.

As developers, we also practiced white box external testing, mainly via statement testing and indirect path testing, where we followed logical paths through our app from a typical user's perspective. Throughout the testing process, we worked hard to expose vulnerabilities by intentionally "breaking the system". For instance, we performed boundary testing - what happens if a user has 0 Subscriptions? What happens if there are 0 matches in the Trades system? What happens if the user enters an empty string as their search query? - and extended stress testing - What happens if the user copies and pastes a long string into a text field? What happens if the user tries to spam our system by turning on many switches? What happens if the user repeatedly turns on and off a switch? What happens if the user repeatedly clicks on the same course page entry in the search results? These questions of boundary conditions or high-stress inputs were constantly on our minds as we developed our application and motivated us to implement safeguards to protect against special corner cases and excessive stressors on our system. See the robustness bullet points at the beginning of this section for more information on the protections we built.

We also practiced field external testing following the alpha and beta stages of our application. To test our core features, we gave potential users (some Princeton students) a list of tasks to complete, but we also gave users free reign to explore the app and try out any features (even ones we hadn't given in the tasks), which they sometimes did in unexpected ways. In fact, in our beta field testing session, one of our users discovered a bug after entering the query "Computer Science" in the search form when we asked him to search for "COS" courses. Some courses do have "Computer Science" in their title, but they weren't showing up in the search results because we were incorrectly stripping whitespaces while processing search queries in the backend. We had previously never noticed this issue, since we were biased by our development process and



were habituated to always searching for courses by the department code, “COS”. Although it was somewhat embarrassing that we did not catch this significant bug, we were grateful to our field users for their experimentation with our app and definitely see the value in conducting real-world field evaluations of our product.

User Evaluation

Following the beta stage, we surveyed three users who have never used our system before. Overall, our users found our main features understandable and intuitive to use, and were impressed with our simple and informative UI design.

They also expressed interest in using our product after this semester, during future course enrollment and add/drop periods. Here, we break down the users’ positive comments to our features:

1. **Tutorial:** When they first logged in, all three users read through the Tutorial and were satisfied with the depth (e.g. detailed annotations on each page layout) and breadth (e.g. giving an overview of our main features) of the Tutorial. It seemed that the Tutorial provided sufficient information to get our users quickly started on TigerSnatch. Thanks to our informative tutorial, users were able to take initiative to perform course searches and engage in Subscriptions & Trades, even before we had instructed them to.
2. **Course search:** We notice that users were, in general, drawn to the search bar at first, but did take time to explore tooltips and status indicators if they didn’t understand a feature on the Dashboard and Course pages. The search feature was very intuitive for users – they had no problems at all finding a course, unless their query was not specific enough.
3. **Subscriptions:** Users figured out that if they “want to get into” a section, they need to be Subscribed to that section. They knew how to Subscribe with no problem and recognized that all Subscriptions were viewable from the dashboard. Users clearly understood that email was the method of contact if slots open up. When they received the email, they found the email’s content intuitive, understood that they should unsubscribe on TigerSnatch if they got the slot or no longer wish to get notifications, and knew to unsubscribe by toggling the switches off on the Dashboard. Furthermore, when they Subscribed/unsubscribed from a section, they noticed and responded positively to the pop-ups that indicated their actions’ success.
4. **Trades:** Users knew the purpose of Trades and showed interest in using it to switch sections in the future. They knew how to save their current section before finding matches, and how to contact their matches when they find one. Users also seemed impressed that the Trades email was auto-generated in their inbox.

Alongside overall positive feedback, our users also gave some some concrete suggestions to how we can improve user experience, all of which we implemented after the meeting:



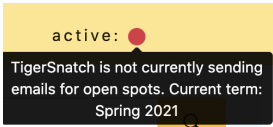

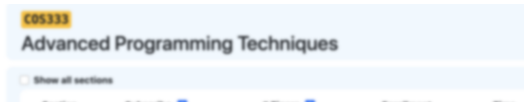
1. Sort course results by their level. For example, COS 126, COS 226, COS 217 should precede other COS courses when users search "COS." We added sorting to the search results after the meeting.
2. Clarify the purpose of the Subscriptions column on the Activity page. We added this explanation after the meeting.
3. Underline direct course links on the Dashboard to make it clear that they're links. We implemented this change after the meeting.
4. Clarify that the act of clicking the email button in Trades adds a message both users' Activity logs. After the meeting, we removed the tooltip with the disclaimer, and placed the disclaimer directly as text on the pop-up.
5. Some wording modifications – change wording of Trades Tracker tooltip to use "current enrollment," and change wording of status tooltip to use the phrase "sending emails." We implemented this after the meeting.
6. Add hyperlinks to course pages in Trades Tracker. We implemented this after the meeting.

We list the tasks used to prompt our users during our testing session in the Appendix.

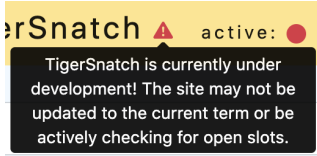
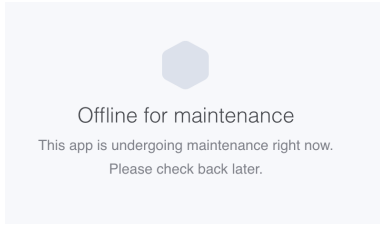
Expert Evaluation

Heuristic 1: Visibility of system status

The application has various conspicuous yet non-distracting widgets and mechanisms to keep users informed of the current system status, system changes, and processes happening in the backend, including...

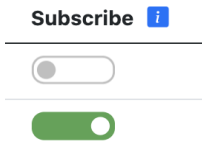
Description	Example
A color-coded status indicator to inform users whether TigerSnatch is actively sending emails for open slots.	
Descriptive pop-ups that inform users when their actions, such as Subscribing to/unsubscribing from a section, saving a currently enrolled section for Trades, or blacklisting a user on the Admin page, are successfully completed.	
The blurring of the page and the loading icon on the user's cursor while their request to load a course page is being processed. It may take a few seconds to pull the latest updates for a course page, so these visual mechanisms were put in	



<p>place to let users know that their request is being processed and that the site has not stopped working.</p> <p>For the same reason, the user's cursor and Subscription switches are briefly disabled while a user's request to Subscribe to or unsubscribe from a section is being processed.</p>	
<p>A conspicuous warning to let users know that TigerSnatch is under development, so app data may not be up to date.</p>	
<p>A maintenance message to prevent users from accessing the app when it is undergoing large-scale changes, such as when the admins are updating courses for the new term.</p>	

Heuristic 2: Match between system and the real world

The application uses terminology and feature names that cater to any user, regardless of a user's tech-savviness. For example, in our backend methods and database, we refer to Subscribing and unsubscribing as adding and removing a user from "unordered waitlists". On the front-end, since most users are familiar with the concept of subscriptions, in all our guides, tooltips, and labels, we tell users to "Subscribe" and "unsubscribe" from sections without using the term "unordered waitlists". In the backend and database, we record user's trades and Subscription notifications in "logs"; on the frontend "Activity" page, we substitute the word "logs" with "messages", since users may be unfamiliar with the concept of system logs. Also, since the application is tailored to Princeton students, we strayed from using technical labels like "user" and consistently refer to our users as "Tigers".


Technical term for developers	Familiar term for users	Example
"Add or remove from unordered waitlist"	"Subscribe or unsubscribe from a section"	

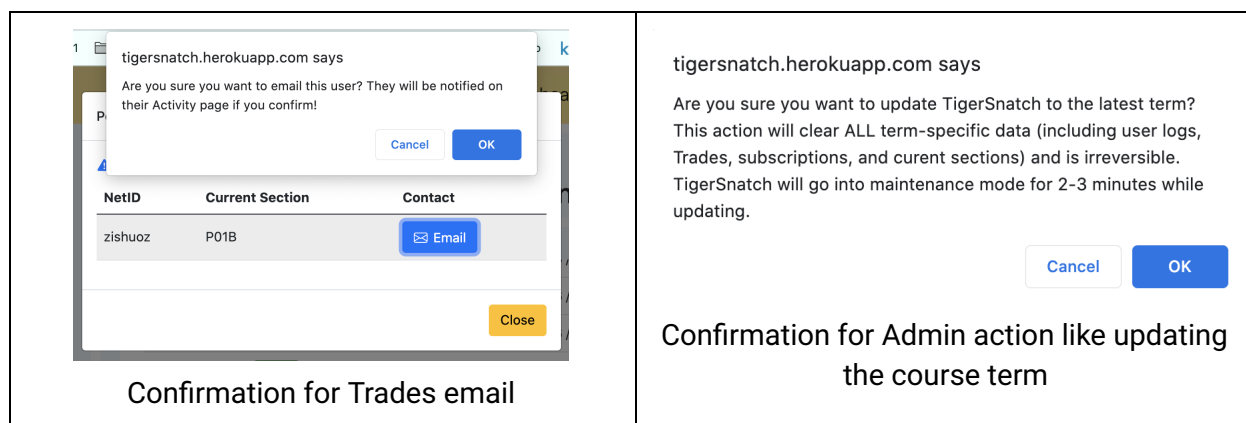


"System log"	"Activity message"	<div><div>Trades</div><div>Once you've initiated an email to Trade sections with another Tiger, a message will appear here!</div></div>
"App user"	"Tiger"	<div># Tigers i</div> <div>0</div> <div>1</div>

Heuristic 3: User control and freedom

The application allows users to safely exit from erroring states and adds extra layers of confirmation to allow users to easily undo and prevent accidental/unwanted actions. When a user performs an error-ful action, such as navigating to an endpoint that does not exist, the app presents the user with an error page, including a "Go to Dashboard" button that allows the user to quickly and safely return to the app's normal state. We intentionally used switches to make Subscribing and unsubscribing intuitive and fast; however, while clicking around, users may accidentally toggle a switch and initiate an un-subscription from a section. To prevent users from unsubscribing from their desired sections without noticing, the app displays a pop-up that allows the user to confirm or cancel their un-subscription -- to prevent users from performing a possibly mistaken action. Likewise, when users click "Email" upon a Trade match, the app displays an alert, informing the user that this action will cause a message to be displayed on the user and the match's Activity page. This gives the user a chance to reconsider and prevent a potentially undesirable and permanent action. It is important to note that many admin features require extra confirmation via alerts before being executed to encourage admins to evaluate the consequences of their potentially irreversible actions that will likely cause widespread system changes.

<div><div>Oops! Something went wrong.</div><div></div><div>Go to Dashboard →</div></div> <div>Safely return to Dashboard from error page</div>	<div><div>Confirm Unsubscribe</div><div>Are you sure you want to unsubscribe yourself from this section?</div><div>Cancel Confirm</div></div> <div>Confirmation for unsubscribing</div>
---	---



Heuristic 4: Consistency and standards

Throughout the app, we established consistency in both our terminology and our visual design. For example, we always refer to our main features as Subscriptions and Trades in our labels, guides, and tooltips. To avoid confusion, we are always careful to distinguish between a “course” and a “section”. In terms of design conventions, hyperlinks are always underlined so users know to click them; pop-ups always appear on the top-right corner to confirm that the user’s actions were successfully (green) or not successfully (failed) processed; on both the Dashboard and Course pages, Subscribing and unsubscribing always occur via switches; tooltips are typically blue info icons so users know to hover and read the text for help; the Dashboard, Course, and Admin pages were designed with the same layout, with search on the left side, a header, main table, and two useful widgets on the right side; when a feature does not contain any user data yet (e.g. empty search results, Dashboard Subscriptions table, Trades tracker panel), our app always displays placeholder text that tells the user how to populate this area. Overall, we have attempted to establish standards for wording and visual layout in order to enhance the user experience.

Heuristic 5: Error prevention

The app provides various safeguards to prevent errors, such as invalid inputs or unintentional actions, from occurring in the first place. In both the course search and contact preferences form, the app performs front-end validation to disallow users from submitting queries of the wrong format (e.g. no @ in the email address, HTML tags and other weird characters in the course search query) or of extremely long length to the back-end. The app also prevents strange characters or long queries if users attempt to enter their course search query via the URL. While a request is being processed in the backend, the app makes sure to disable features that may interfere with the request from successfully finishing and thus causing erroneous behavior. As previously mentioned, while a new course page is loading, the user’s cursor is disabled and the page is blurred to stop the user from continuing to click on features on the page. Likewise, when a switch is toggled for Subscriptions or a button on the Admin panel is clicked, all switches or the current button are disabled, respectively, to prevent erroneous behavior if the user or admin



spams mouse clicks or keystrokes as the process is loading. As described in Heuristic 3, the app includes pop-ups as confirmation of user actions that may have widespread changes (particularly on the Admin page), irreversible effects (like with clicking Email on a Trade), or may be undesirable (such as unnoticeably unsubscribing from a section).

Search by course department, number, or title! Please only enter alphanumeric, spaces, quotes, ;, ?, % characters. 100 characters.

Note: Please match the requested format.

Form validation for course search

TigerSnatch will notify you at this email when a slot frees up! It is recommended to use your @princeton.edu address to prevent TigerSnatch emails from being marked as spam.

Contact Preferences

email@gmail.com Change

Please match the requested format. Change

Form validation for contact preferences

Welcome, Admin

Admin functions are listed below.

Email notifications turned off! Reloading in a few seconds...

Description	Inputs	Action
Update to Latest Term (current: 1214)	-	Update
Toggle Email Notifications	-	Turn Off
Clear All User Logs	-	Clear
Clear All Trades	-	Clear

Admin buttons (under Action) are disabled while request to turn off email notifications is being processed

Heuristic 6: Recognition rather than recall

The app's interface presents up-to-date, personalized user data and logs to minimize each user's memory load. On the Dashboard, the app displays both the user's currently subscribed sections and their currently enrolled sections for Trades, so they do not have to remember this essential information. The contact preferences panel also shows the user's email address, so the user won't forget which email address they are expecting to receive notifications on. The Activity page logs recent actions associated with the user's account, including initiated Trades and email notifications about spots, so users don't need to dig through their emails to find previous Trades or Subscriptions notifications. We put both the Trades and Subscriptions feature on each course page (instead of having a separate Trades page), so that users do not have to memorize their Subscribed sections for a course before engaging in a Trade. Although



the Tutorial provides a detailed introduction of our features to first-time users, if users forget how a particular feature works, we have placed hoverable tooltips in various locations on the Dashboard, Course page, and search/email forms to succinctly remind users how to engage with each feature. We have also made the Tutorial page accessible via the nav bar so users can re-read it if needed. For convenience, we implemented our search feature such that the list of search results persists on the left panel and the app highlights in yellow the currently selected course. This way, users do not have to memorize their most recent course search query and can easily switch between similar course options.

Course	Section	Subscribe	Time	Days
COS126/EGR126	B03B	<input checked="" type="checkbox"/>	12:30 PM - 01:20 PM	T Th
COS126/EGR126	P01B	<input checked="" type="checkbox"/>	09:00 AM - 09:50 AM	F
COS226	P02	<input checked="" type="checkbox"/>	04:30 PM - 05:50 PM	Th

Contact Preferences
 sheh@princeton.edu Change

Trades Tracker
→ You seek to trade out of COS126 P02A
→ You seek to trade out of COS226 P10

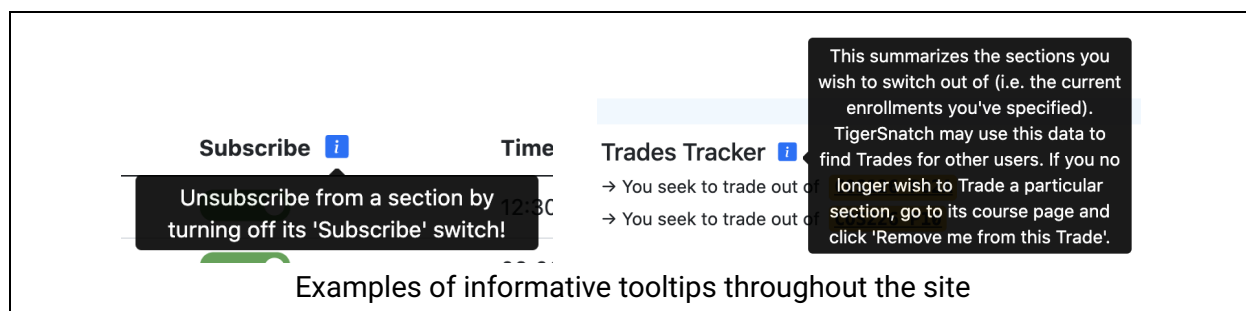
Current Subscriptions, enrolled sections for Trades, and email for notifications conveniently displayed on Dashboard

Subscriptions	Trades
May 03, 2021 @ 2:33 PM ET → 3 spots available in ORF309 P01	May 03, 2021 @ 2:30 PM ET → You contacted zishuoz to swap into COS126 P01B
May 03, 2021 @ 2:32 PM ET → 3 spots available in ORF309 P01	May 03, 2021 @ 2:30 PM ET → You contacted ntyp to swap into COS126 P01B

Recent user-related actions stored in Activity

<input type="text" value="COS"/>	COS126/EGR126														
Note: TigerSnatch subscriptions are separate from Registrar waitlists.															
EAS332/GSS429/COM352 Cosmopolitan Her: Writing in Late Capitalism	Computer Science:														
COS126/EGR126 Computer Science: An Interdisciplinary Approach	<input type="checkbox"/> Show all sections														
COS217 Introduction to Programming Systems	<table><tr><th>Section</th><th>Subscribe </th></tr><tr><td>B03A</td><td><input type="checkbox"/></td></tr><tr><td>B03B</td><td><input checked="" type="checkbox"/></td></tr><tr><td>B03C</td><td><input type="checkbox"/></td></tr><tr><td>P01B</td><td><input checked="" type="checkbox"/></td></tr><tr><td>P02A</td><td><input type="checkbox"/></td></tr><tr><td>P02C</td><td><input type="checkbox"/></td></tr></table>	Section	Subscribe	B03A	<input type="checkbox"/>	B03B	<input checked="" type="checkbox"/>	B03C	<input type="checkbox"/>	P01B	<input checked="" type="checkbox"/>	P02A	<input type="checkbox"/>	P02C	<input type="checkbox"/>
Section	Subscribe														
B03A	<input type="checkbox"/>														
B03B	<input checked="" type="checkbox"/>														
B03C	<input type="checkbox"/>														
P01B	<input checked="" type="checkbox"/>														
P02A	<input type="checkbox"/>														
P02C	<input type="checkbox"/>														
COS226 Algorithms and Data Structures															

Search results from most recent query persistently displayed



Heuristic 7: Flexibility and efficiency of use

Although our application is simple to navigate for both first-time and frequent TigerSnatch users, there are a few ways we have catered separately to novice and experienced users. For instance, to acclimate first-time users with TigerSnatch features and terminology, the Tutorial is automatically shown after first-time users login. This is not the case after the first login, but experienced users still have easy access to the Tutorial page via the nav bar. Moreover, the app includes non-intrusive tooltips scattered to briefly remind more experienced users about certain features. Another minor feature that can vastly improve efficiency for expert users are hyperlinks to specific course pages on our Dashboard. In particular, if an expert user is subscribed to many sections of different courses or is engaging in trades for various courses, the course pages for these Subscriptions and Trades are hyperlinked in their Subscriptions table and Trades Tracker, so that frequent users do not have to waste time searching for a course in the form each time they want to return to these courses. Moreover, in mobile view, we have implemented ease-of-use links that allow frequent users to auto-scroll down to their main Dashboard or course page ("Skip to Dashboard/Course"), or auto-scroll back up to the search form ("search"). Additionally, clicking/tapping on a direct course link from the Dashboard will automatically cause a scroll down to the course page's main content. Finally, by making our application CAS-authenticated, users who frequently visit the app do not have to login every time since CAS saves a user's login.

One quick note: although dynamic loading of the search results, as the user is typing their search query, would surely improve the efficiency of use for all users, we made the intentional design decision to not implement this, primarily to reduce the load on our backend search process. This may cause some confusion among expert users of other Princeton course-related sites, such as Princeton Courses and Recal, which do have a dynamic search, but we make the reasonable assumption that users will understand that they must explicitly submit a query like that on the official Course Offerings page.



Course	Section	
COS126/EGR126	B03B	Trades Tracker ⓘ → You seek to trade out of COS126 P02A → You seek to trade out of COS226 P07
COS126/EGR126	P01B	

Hyperlinks to quickly access course pages from the Dashboard

Note: TigerSnatch subscriptions are separate from Registrar waitlists.

Search for courses above!

[Skip to Dashboard/Course](#) ↓

Welcome, sheh ↑ [search](#)
Check out your Snatches below ⓘ

Ease-of-use auto-scrolling when in mobile view

Heuristic 8: Aesthetic and minimalist design

The app interface optimizes the space to visibly present only relevant information and less so additional details and instructions. However, we have still made these details easily accessible since the user may reference them sometimes. For example, users may occasionally forget how certain features work - the app has tooltips dispersed throughout to concisely explain the features. The tooltips are typically represented by a minimal blue (i) icon and simply require a mouseover to view the text. This way, the tooltip's text does not typically interfere with the rest of the page's more relevant content. Likewise, we factored out a detailed app walkthrough into a separate Tutorial page in the nav bar, so that this content-heavy guide, which users may occasionally need, is still accessible but does not compete with the frequently-viewed content on the Dashboard and Course pages.

On our course pages, we simply display a header with the course department, number, and title, along with the main section table with section name, enrollment, time, and day. We do not display extra course details like professor name, course description, distribution area, etc., in order to de-clutter the page and focus the user's attention on TigerSnatch's main features: Subscription and Trades. As an alternative, we created a simple Quick Links panel on each course page that links to a course's official Course Offerings page, a course's Princeton Courses page, and the user's TigerHub portal, so that users can easily access additional course details from our site. While some users may prefer to see all course information directly on TigerSnatch, we would like to emphasize that showing extra course details is not the purpose of TigerSnatch, since most users will visit the site already knowing which courses they are interested in. To further minimize irrelevant content on course pages, by default, the app only displays a table of *full* sections for a course -- students use TigerSnatch to quickly Subscribe to



full sections so listing all sections, available or not, would make it harder for users to find their desired full section. If users want to view all sections, they can simply tick the “Show all sections” checkbox above the sections table.

Throughout the site, to help guide the user’s eye to the most relevant information, we used a simplistic, non-flashy color scheme and font, and made the most important features or words more visible via highlights (e.g yellow highlight in course header), unique colors (e.g. subscribed switch is green, the only green on the page), or large sizing (e.g. making the subscriptions table biggest on the Dashboard).

Quick Links for COS126/EGR126

[Official Course Offerings](#)

[Princeton Courses](#)

[TigerHub Portal](#)

Quick Links on course pages

☒ Show all sections

Section	Subscribe	# Tigers	Enrollment	Time	Days
L01		0	222 / 250	11:00 AM - 12:20 PM	T Th
P01		0	26 / 27	03:00 PM - 04:20 PM	Th
P02	<input checked="" type="checkbox"/>	1	29 / 27	04:30 PM - 05:50 PM	Th
P03		0	25 / 27	11:00 AM - 12:20 PM	F
P04	<input type="checkbox"/>	0	27 / 27	11:00 AM - 12:20 PM	F
P05		0	23 / 27	01:30 PM - 02:50 PM	F

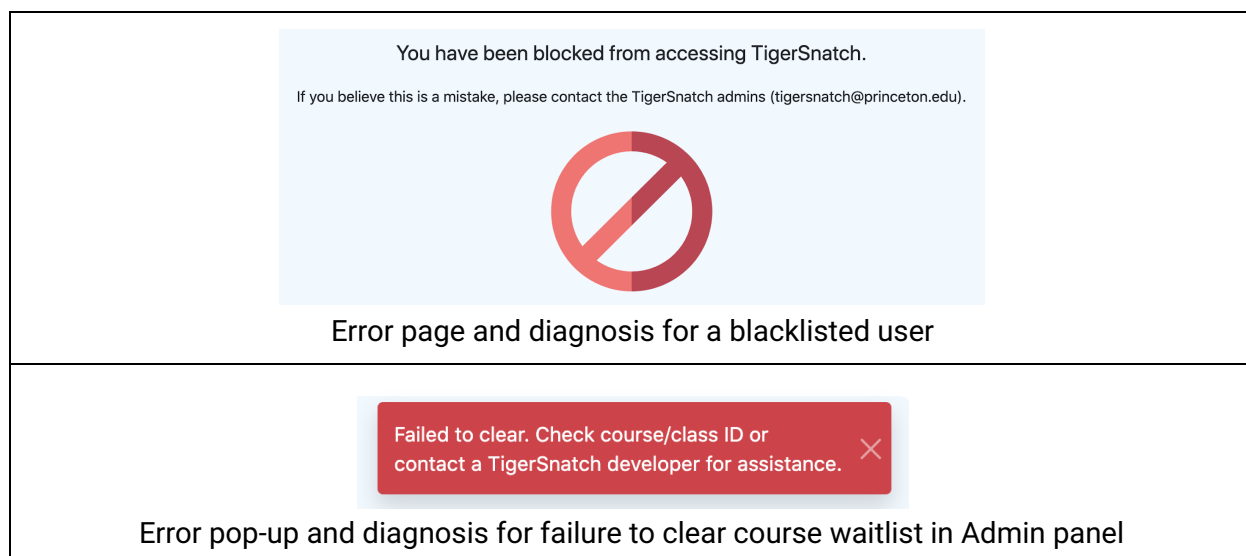
"Show all sections" checkbox on course pages (unchecked by default)

Heuristic 9: Help users recognize, diagnose, and recover from errors

In the unlikely event that a user ends up producing an error, the app guides the user in resolving the situation quickly. As previously discussed, when a user attempts to enter an invalid search query with forbidden characters or submit an invalid email, an error tooltip is displayed and asks the user to “Please match the requested format”. To diagnose why their query is erroneous, the user just needs to hover over the text field to read a tooltip about allowable characters. If the user attempts to access a non-existent endpoint or maliciously affects the app in an unauthorized way, they will be directed to a basic error page, saying “Something went wrong”. The simplest way to resolve this error stage is to click the “Go to Dashboard” button provided on the page. When a user has been blacklisted by an Admin (e.g. for malicious behavior), an error page is displayed for this user when they attempt to login and makes clear that to resolve this issue, they must reach out to the TigerSnatch admins. In the Admin page specifically, if an Admin enters an invalid input (e.g. a non-existent netid or course id) for an admin action, a pop-up is displayed notifying the Admin user that the action has failed and suggesting that to



diagnose the failure, the Admin should double-check that their input is valid before contacting the developers for assistance.



Heuristic 10: Help and documentation

Help and documentation for TigerSnatch is extremely easy to locate. Users always have access to a walkthrough of our site via the Tutorial page, which can be found in the nav bar. The Tutorial gives a general overview of all our features and includes helpful diagrams that annotate key components of our pages. As we have already discussed, the app includes many short and accessible tooltips (represented by the classic blue info icon) to guide users on how to use various features, while they are navigating the app.

Cognitive Walkthrough: Subscribing to a Section

Will the correct action be sufficiently evident to the user?

- How and where to Subscribe should be sufficiently obvious to the user. In the Tutorial shown to first-time users, we guide users to toggle switches to Subscribe to course sections, and include a screenshot of these switches. For first-time users, the Subscriptions table on the Dashboard will be empty - however, the table includes a placeholder message that instructs users to “Search to get started”. This instruction directs users to the search bar, so they can begin to search for courses and Subscribe to specific sections.

Will the user notice that the correct action is available?

- When the user navigates to a specific course page (e.g. COS126), the user should notice the Subscriptions switches that we mentioned in the Tutorial. The Subscription switches should be fairly obvious since they are displayed under the corresponding “Subscribe” column in the main course table and are visually distinct from the rest of the



texts/buttons on the page. We include an information tooltip next to the “Subscribe” column header to again instruct users to Subscribe to a section by “turning on its switch”.

Will the user associate and interpret the response from the action correctly?

- When the user toggles the switch on, there are multiple indicators that assure users they have properly Subscribed to a section. First, in the Subscribed state, the color of this switch becomes green to mark an active Subscription. Moreover, in the top-right corner, the app displays a green pop-up notifying the user that they have “Successfully Subscribed”. Finally, the user can check that their Subscription was indeed saved by returning to the Dashboard, and verifying that their Subscribed section is now displayed in their personalized Subscriptions table.

Appendix: Beta Test User Tasks

Task 1: Login to TigerSnatch. Read through the Tutorial and let us know if you have any questions about the tutorial. Get to the Dashboard.

Task 2: Explore the Dashboard page and tell me things you notice.

Task 3: Look for COS courses and go to COS126 page. Explore the COS126 page and tell us what you see.

Task 4: Let’s say you want to get into COS126 P01B. What should you do? [PAUSE] Search for any other course (e.g. your favorite) and subscribe to a couple more. Head back to the Dashboard and tell me if you see anything different.

- *In case a user can’t find another course with full sections, recommend ECO100 and ORF309.*

Task 5: Ask the user “how will you know when a slot opens up if you use TigerSnatch?”

Task 6: (on Dashboard) Actually, you no longer want to get into ____ (something that is not COS126 P01B). What should you do? [PAUSE] Do you notice any changes on the Dashboard?

- *User should unsubscribe. User should see that entry is removed from Dashboard.*

Task 7: Head back to the COS126 page. Let’s say you’re already enrolled in COS126 precept P03E. You want to get into P01B (without going through the hassle of waiting for someone to drop), so you want to find someone in P01B to switch with you. What should you do?

- *If user is getting lost, prompt them to look at the Trades panel at the bottom.*
- *User should choose P03E as the current section from dropdown and save it. User should click Find Trades.*



Task 8: You found a couple people who could switch precepts with you! How would you get in touch with one of them?

- *User should click Email upon one of the Trades.*

Task 9: Head back to the Dashboard. What sections have you specified current enrollment in? Head to the Activity page and tell me what you see.

- *User should point to Trades Tracker on Dashboard. User should see a new log from sending an email.*

Task 10: [manually send an "open spots" email to the user] This is the email you'll get if a spot opens up in one of your Subscribed sections. What should you do?

Dear _____,

Your subscribed section P01B in COS126: Computer Science: An Interdisciplinary Approach has one or more spots open!

Head over to TigerHub to Snatch your spot!

You'll continue to receive notifications for this section every 2 minutes if spots are still available. To unsubscribe from notifications for this section, please visit TigerSnatch.

Best,
TigerSnatch Team <3

Task 11: Continue exploring the page, check out some features that we might not have talked about. Do you have any questions?