– Independent Work Report Spring 2019 –

TigerNest: Visitor Flow

Michelle Yuen
Adviser: Dr. Robert Dondero

Abstract

This paper details the process behind, implementation of, and evaluation of TigerNest, a web application that generalizes the overnight event host-visitor matching process for the Princeton community. Specifically, this paper focuses on the off-campus visitor user experience and workflow. Another independent work project done by my peer Niranjan Shankar works in conjunction with this one to build out the remaining users' experiences, and may be mentioned for the sake of clarity in this paper. TigerNest is the first web application to provide a central site for event organizers, Princeton hosts, and off-campus visitors to plan and attend a successful event. In order to ensure continued use and user satisfaction, this paper also discusses the requirements gathering after interviews with several Princeton student-run organizations that host overnight events. Furthermore, this project also sought after requirements from the Undergraduate Student Government TigerApps committee to work towards becoming a sponsored TigerApp. Lastly, this project details a new approach to the host-matching process with the goal of lessening the load on event organizers and giving visitors more control in their choice of rooms.

1. Introduction

1.1. What is the Host-Visitor Matching Process?

Princeton University offers a wide variety of extracurricular clubs, and several of these clubs organize overnight events that involve inviting students from other universities to bring diverse perspectives to the Princeton event. In order to cut down on lodging and transportation costs for these student visitors, the Princeton club members organizing the event (event organizers) will ask other Princeton students to serve as "hosts" by providing their dorms for the off-campus visitors to sleep in. This is process is repeated for every overnight event hosted on campus.

However, this process can become very tedious for the event organizers, who must collect all their visitors' information, attract students to volunteer their rooms as hosts for the event, and then find a suitable arrangement of visitors to hosts so that all visitors and hosts are comfortable and have a place to stay. Furthermore, these events are often large, one organization we interviewed estimates an attendance of approximately 150 visitors twice a year. Another downside to this current workflow is that Princeton students looking to host do not have one centralized hub to sign up their rooms. Often, event organizers will "blast" email listservs trying to reach students, and these emails may get lost in student inboxes. For visitors, their event organizers will collect only their preference as to whether or not they are uncomfortable sleeping in a room with others of the opposite gender, but beyond this visitors do not have any other say in the process, such as the number of guests also sleeping in that room.

1.2. The Goal of TigerNest

Thus, this project seeks to address inefficiencies in the current way student organizations "match" off-campus visitors to Princeton hosts. By taking a new approach and consolidating this process into one web application available to the Princeton community, all events can be found in one place, making it easier for interested hosts to sign up. Additionally, event organizers will have much less to worry about with their event since this web application will handle the host-visitor matching process for them.

By letting visitors see available room types and sign up for a specific room type, this project approaches the host-visitor matching problem in a new way. Also, since no generalized web application for host-visitor matching for overnight events currently exists, this paper will also detail the requirements gathering process after speaking to 6 student organizations that host overnight events. Another long-term goal of this project was to be adopted as an Undergraduate Student Government

(USG) TigerApp, since the website is designed to serve the Princeton community. By enumerating requirements ahead of implementation, TigerNest is also the first project to be built with the explicit intention of becoming a TigerApp, as other TigerApps were built for other purposes before later becoming adopted.

It is important to note that while TigerNest is one website, there are three types of "users" that it targets: event organizers, hosts, and visitors. This project owns the entire visitor "flow" from end-to-end, and also full-stack (frontend and backend). Other user flows were built out in another project produced by Niranjan Shankar, and combined to create one hub for all three users. For the sake of clarity in this paper, the other user flows may be discussed.

2. Related Work

2.1. Working with Focus on the Visitor Experience

As mentioned above, though TigerNest has three types of users: event organizer, host, and visitor, this paper specifically focus on the implementation of the visitor's experience. However, due to the nature of the problem we are trying to solve, there must exist some interaction between users in terms of data. Because of this, all three users must share and interact with the same database as shown in Figure 1, which will not go into further detail for the time being.

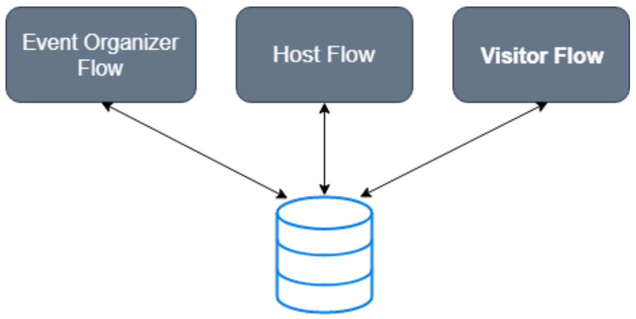


Figure 1: A simplified diagram of separate user flows but one centralized database.

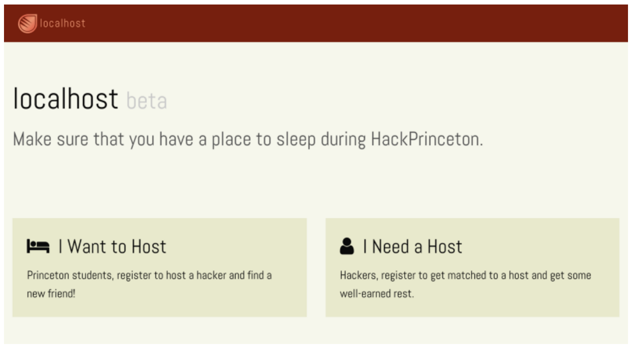
This paper will focus on the bolded flow, the visitor, and this paper will also explain in more detail as to the technologies behind the visitor flow and database. Overall, this is just to clarify the separation of work for this web application, and where the main source of connection lies.

2.2. Other Host-Visitor Matching Applications

In terms of work already done in this area of matching hosts to visitors, hackathons are often the catalyst for development in this field. However, the applications that will be discussed are all use-specific, as in they were created by hackathon organizers for the specific purpose of aiding host-visitor matching for that specific hackathon. There is currently no documented web application for general use host-visitor matching which is a key difference between TigerNest and the following web applications.

A widely attended hackathon on the East Coast, HackMIT, must perform matchings for approximately 350 hackers. To do so, the organizers send out a Google Form to both interested visitors and hosts and record their information. Then, to perform the matching, they represent their visitors and hosts as nodes in a bipartite graph, and use a max-flow algorithm (where the "flow" is defined by the number of guests) to complete the matchings [3]. Similar to this project, the HackMIT automated matching process strives to replace manual matching that used to be performed when there were fewer event attendees. Furthermore, HackMIT's work demonstrates a working model for handling large overnight events. However, the organizers must put in a significant amount of work collecting information from two different Google forms, and then processing information from over 600 people to output matchings. Additionally, it is worth noting that HackMIT does give visitors preferences regarding same-gender, non-smoking, and even pet-friendly rooms, but does not cover how their algorithm accounts for this in matching.

The most comprehensive host-matching service available for Princeton is a web application called "localhost" and is specific to the HackPrinceton event. Similar to HackMIT, localhost uses a max-flow algorithm for automated matching. Additionally, because the application is specifically for a Princeton event, localhost also utilizes Central Authentication Service for student hosts to login, and then a custom login for visitors, with a simple UI for the two users as seen in the appendix in Figure 2 [5].



Make sure that you have a place to sleep during HackPrinceton.

Princeton students, register to host a hacker and find a new friend!

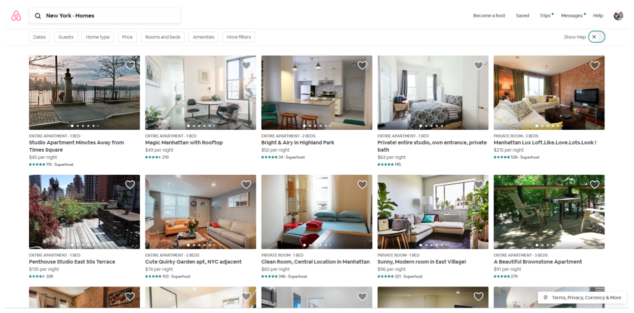
Hackers, register to get matched to a host and get some well-earned rest.

Figure 2: A screenshot of the one of the first iterations of HackPrinceton's localhost.

This layout will serve as inspiration for TigerNest's homepage, except TigerNest will have one more option for event organizers. Overall, both HackPrinceton and HackMIT use automated matching algorithms, but express that there is some downside. The HackPrinceton organizers in the article and also in an in-person interview discussed challenges with "weighting" gender preferences of visitors and hosts. For example, a HackPrinceton organizer said that while it would be simple to assign visitor with strict preferences to rooms first, this inadvertently "punishes" visitors who are more flexible. Additionally, we note that both applications give very little authority to the hosts and visitors beyond their initially listed preferences. Most importantly, both applications are event-specific, which means that other Princeton overnight event organizers are performing manual matchings. Furthermore, since this project is focused on the Princeton community, we also reached out to the HackPrinceton organizers for input and feedback, which will be discussed in later sections. We were also able to obtain HackPrinceton matching data from several previously hosted events, which was used in testing.

2.3. Giving the Visitor the Power

As we saw in other host-matching applications, the event organizer performs all the matching after taking in some preferences from the hosts and visitors. This led to a problem with "weighting" hosts and visitors when automated matching was performed in order to not punish hosts and visitors for having or not having certain preferences. This paper looks at a new approach to the matching process that will alleviate a significant amount of work for the event organizer by having visitors pick their matchings instead. This is similar to the Airbnb model of displaying rooms when searching in a specific area. These rooms can vary in terms of size and amenities as seen in 3, but the user is left with the autonomy of choice.



ENTIRE APARTMENT · 1 BED
Studio Apartment Minutes Away from Times Square
$45 per night
★★★★★ 115 · Superhost

ENTIRE APARTMENT · 1 BED
Magic Manhattan with Rooftop
$49 per night
★★★★★ 210

ENTIRE APARTMENT · 2 BEDS
Bright & Airy in Highland Park
$50 per night
★★★★★ 34 · Superhost

ENTIRE APARTMENT · 1 BED
Private! entire studio, own entrance, private bath
$63 per night
★★★★★ 195

PRIVATE ROOM · 3 BEDS
Manhattan Lux Loft.Like.Love.Lots.Look !
$215 per night
★★★★★ 526 · Superhost

ENTIRE APARTMENT · 1 BED
Penthouse Studio East 50s Terrace
$135 per night
★★★★★ 309

ENTIRE APARTMENT · 2 BEDS
Cute Quirky Garden apt, NYC adjacent
$74 per night
★★★★★ 103 · Superhost

PRIVATE ROOM · 1 BED
Clean Room, Central Location in Manhattan
$60 per night
★★★★★ 346 · Superhost

PRIVATE ROOM · 1 BED
Sunny, Modern room in East Village!
$96 per night
★★★★★ 321 · Superhost

ENTIRE APARTMENT · 3 BEDS
A Beautiful Brownstone Apartment
$91 per night
★★★★★ 279

Figure 3: A screenshot of the search results section of Airbnb, showing a card for each room with a summary of important details.

However, dormitory rooms are much less varied than Airbnb homes, so TigerNest uses a more simplified version of these cards for visitors. This way, a visitor can sign up for a specific room type as defined by the host gender, and the number of other visitors are in the room. As mentioned above, we were able to obtain data on previous HackPrinceton events, and noticed that many rooms when reduced to gender and number of visitors, were very similar. For example, hosts most frequently requested only one visitor, and an overwhelming majority of the rooms had a single spot open for visitors. In the design of the visitor experience, I wanted to reduce clutter on the screen and consolidated room "cards" if they were identical in attributes. This gave a more clean look to the search results page which will be shown in later sections.

3. Requirements Gathering

3.1. Clubs That Host Overnight Events

In order to gather data for requirements, we reached out to numerous student groups on campus that host overnight events, and were able to garner interest from 6 student groups: Princeton Debate

Panel, Princeton Model UN, TigerLaunch, Club Swim, Envision, and HackPrinceton. Of the groups we met with, HackPrinceton had the most experience with the host-matching process, having created their own application which is mentioned in the Related Work section above. These 6 groups agreed to meet with us to discuss their needs, requirements, and use cases/scenarios they had experienced before that were crucial to handle [6].

3.2. Initial Meetings

In initial meetings we discussed certain data requirements. For example, student group leaders often communicated with hosts and visitors via email, but would collect cellphone numbers in case either party needed to be contacted immediately. Almost all groups we spoke with also said they used Google Forms to collect this kind of information. Another requirement was a way for visitors and hosts to indicate whether or not they were comfortable sleeping in the same room as someone of the opposite gender, and cited this as the largest source of challenge when it came to matching.

We also asked groups how they were currently handling host-visitor matching, and all with the exception of HackPrinceton stated that they perform all the matching by hand, which can become tedious with larger events. More specifically, Envision stated that one of their organizers had once attempted to write a program to perform the matchings, but abandoned the idea after he was unable to get it to work. When speaking to HackPrinceton and the creators of localhost, we also discovered that their automated matching approach was not perfect, and they were still working on ways to have a more fair pairing process.

Additionally, we covered certain use cases and scenarios that users would encounter and also important scenarios a successful application must be able to handle. Because information about the host should be kept on a need-to-know basis, groups recommended giving visitors accounts so visitors would only see information about the host they were staying with. HackPrinceton has written their own login system for visitors, noting that not all universities use GSuite, and a Google Login would not be able to encompass all possible users. Another scenario the student groups were concerned about involved ensuring visitors with stricter preferences did not get left out or "punished" during matching. Lastly, the Envision club also stated that it was common for hosts and visitors to drop out of the event as the event date neared, and would need the host and visitor lists to be dynamic if people were to drop out.

With those requirements and user scenarios in mind, we drafted up paper and pencil drawn "mocks" of what the website would look like and how it would interact with the user. For my project, this involved drawing out the visitor experience from start to finish.

3.3. Follow-up Meetings

After drawing the mocks of the planned visitor experience, we met up with the student groups again to iterate on the design. In a meeting with HackPrinceton, I was able to identify some places in the design where the functionality was unclear to a user unfamiliar to the site, and made those changes. For example, we settled on calling what visitors sign up for as "room types" and not specific rooms. At this meeting I was also able to obtain HackPrinceton's pairing data from past events which provided a significant amount of insight into the kinds of rooms and visitors the site would have to handle. Furthermore, HackPrinceton is the largest student-run overnight event on campus, and this dataset enumerated the scalability requirement of the project, since HackPrinceton sees approximately 150 visitors twice a year.

We also showed these mocks to our advisor, Dr. Dondero, and made other clarifying changes to the mocks. Overall, these meetings served the purpose of identifying future users of TigerNest, and what they needed from the site. This works towards the goal of achieving widespread use by the Princeton community.

3.4. Meeting with TigerApps

Another goal of this project was to be adopted as a USG TigerApp, which is maintained by USG over time. TigerApps are essentially sponsored by the university and as a result will garner significant publicity and use by Princeton students. We met with the current head of the TigerApps committee, Reilly Bova, to pitch TigerNest and understand what would be required to one day become a TigerApp. He stated that TigerNest would fit into the standards of a TigerApp, which must serve the Princeton community in some way. He also listed some technical requirements for the application, which were to use React [8] and also deploy to Amazon Web Services.

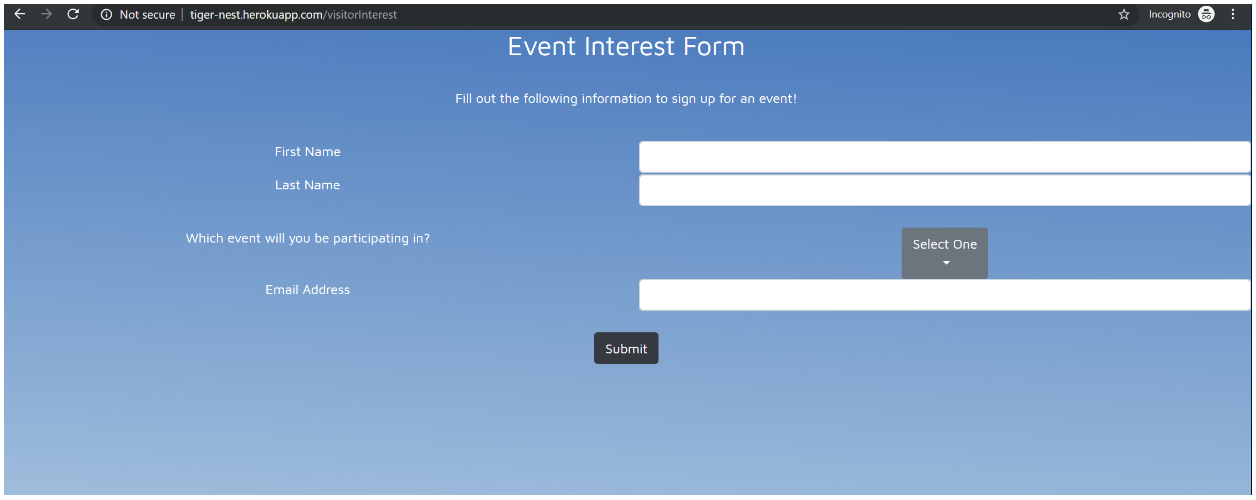
In follow-up meetings with Reilly Bova we also discussed our intended technologies for the site in addition to using React on the frontend. Once we got approval that the TigerApps committee would maintain a site that used our intended technologies (PostgreSQL and Flask), we could begin coding.

4. What It Does

TigerNest is a web application that can be accessed at `http://tiger-nest.herokuapp.com/`, I will walk through some sample visitor scenarios in the following sections.

4.1. Signing Up for an Event

Start by navigating to `http://tiger-nest.herokuapp.com/visitorInterest`. Then, enter your information and select an event to sign up for. Remember the email submitted here as it will serve a purpose later. No events will appear in the drop-down if the event organizer has not added any. To see how to add events, please refer to Niranjan Shankar's paper.

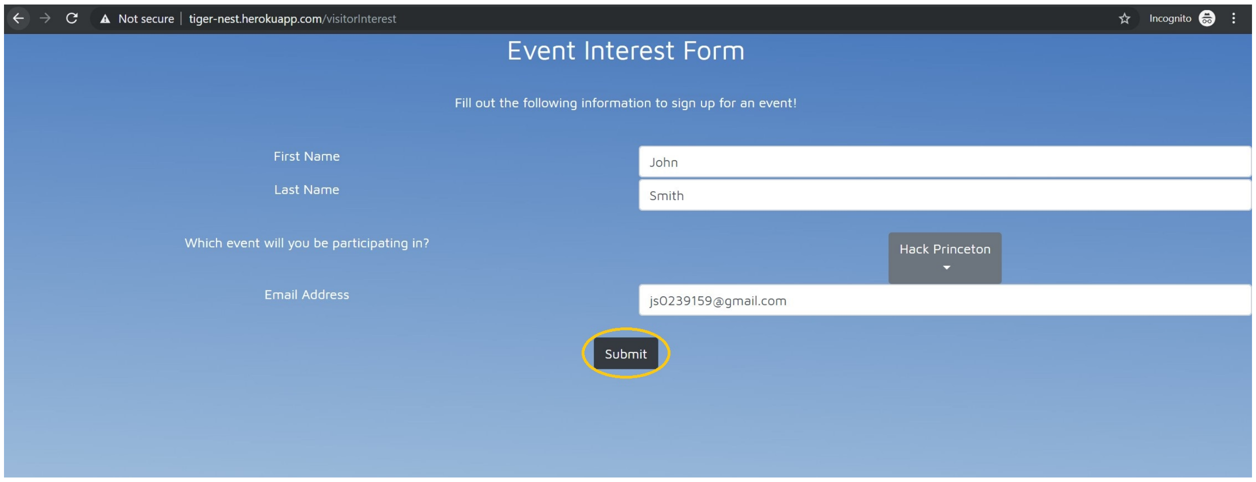

Not secure | tiger-nest.herokuapp.com/visitorInterest

Event Interest Form

Fill out the following information to sign up for an event!

First Name	
Last Name	
Which event will you be participating in?	Select One
Email Address	

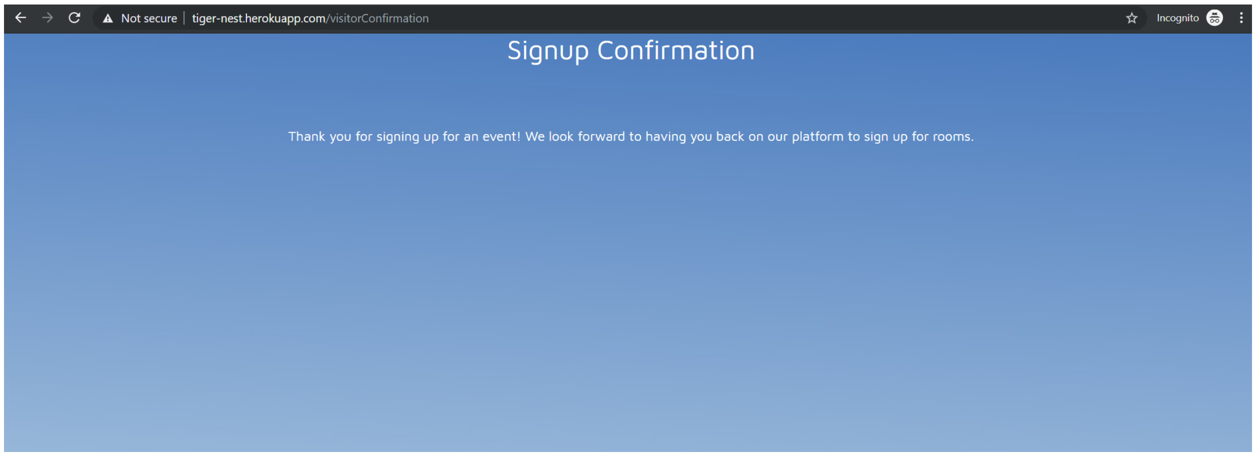
Once you have entered your information, press the "Submit" button to finalize your sign up.


Not secure | tiger-nest.herokuapp.com/visitorInterest

Event Interest Form

Fill out the following information to sign up for an event!

First Name	John
Last Name	Smith
Which event will you be participating in?	Hack Princeton
Email Address	js0239159@gmail.com


Not secure | tiger-nest.herokuapp.com/visitorConfirmation

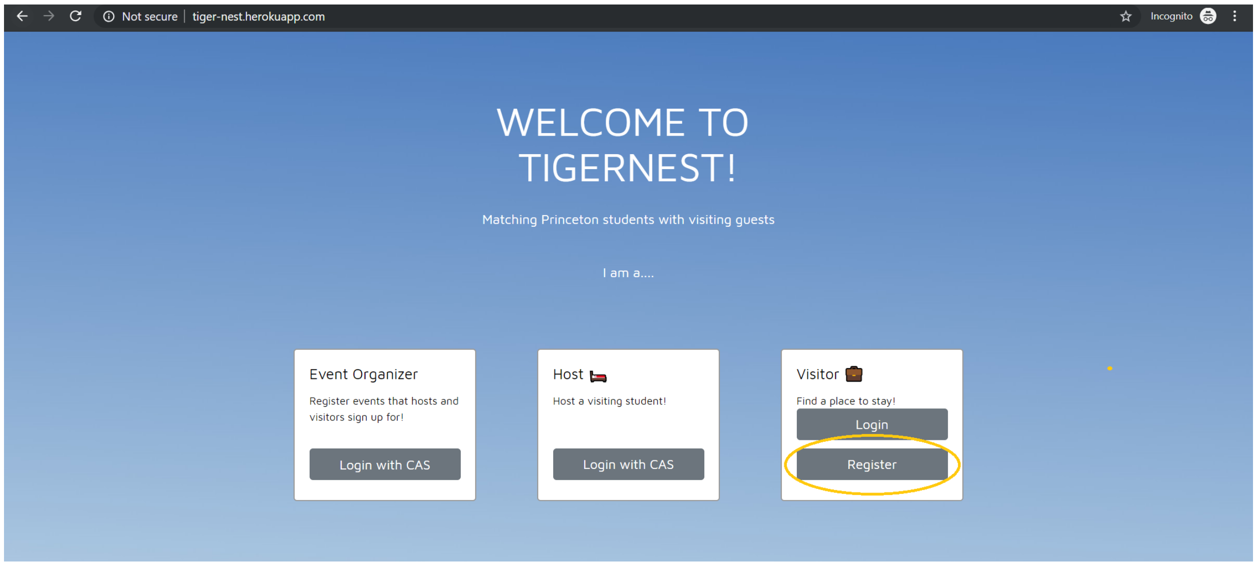
Signup Confirmation

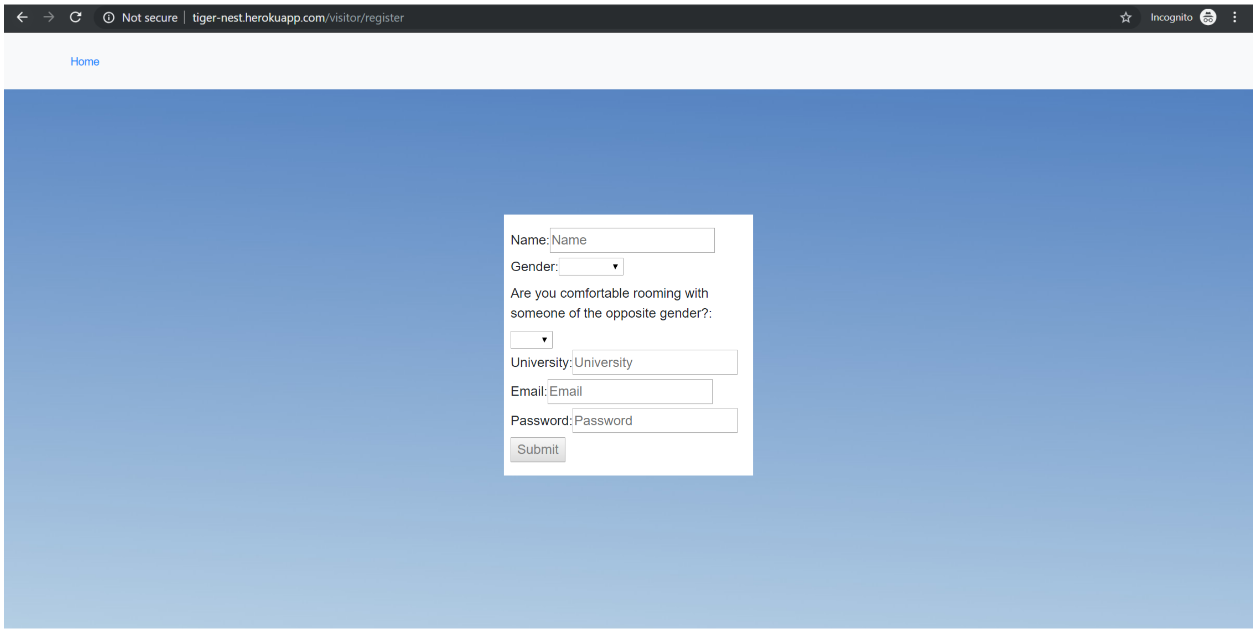
Thank you for signing up for an event! We look forward to having you back on our platform to sign up for rooms.

Now that you have signed up for an event, you can proceed to the main site.

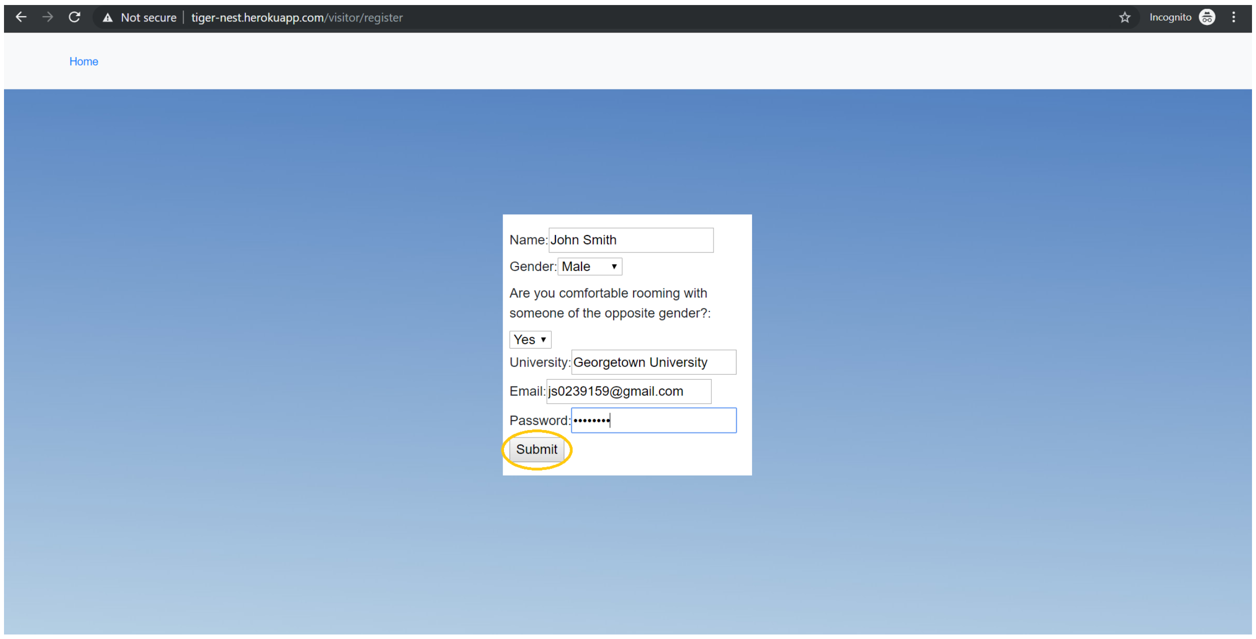
4.2. Signing up for a Room

Start by navigating to `http://tiger-nest.herokuapp.com/`. This is where all user flows can be accessed. For more information about the event organizer and host flows, refer to Niranjan Shankar's paper. To create a new user account, click on the "Register" button in the visitor panel.



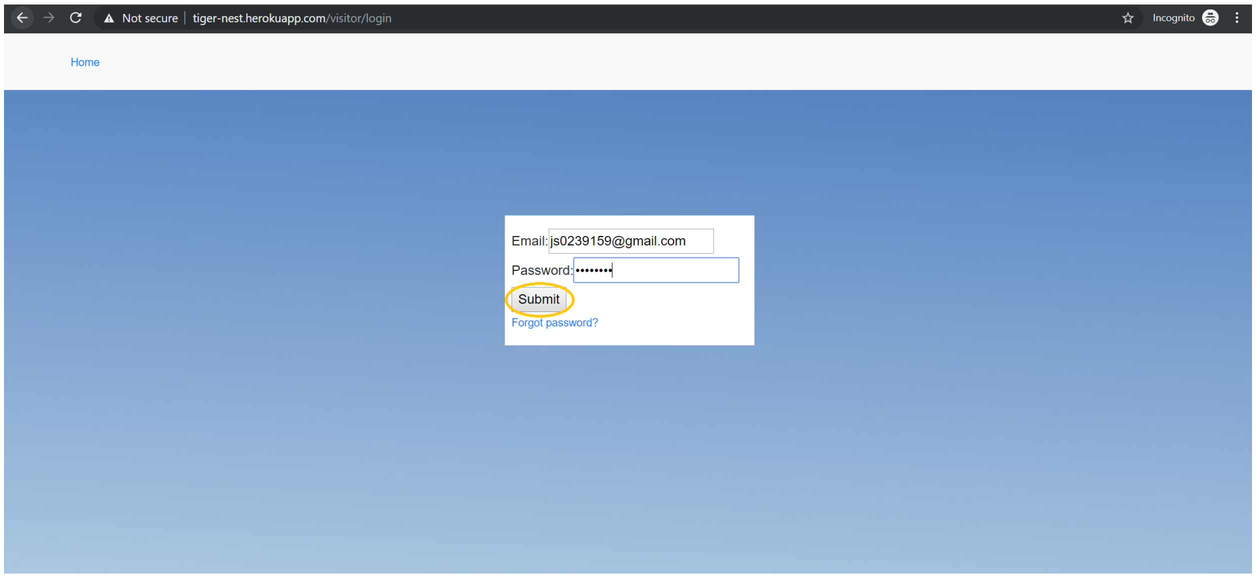

Not secure | tiger-nest.herokuapp.com/visitor/register
Incognito
Home
Name: Name
Gender:
Are you comfortable rooming with someone of the opposite gender?:
University: University
Email: Email
Password: Password
Submit

Then, enter your information and desired password in the fields. For email, make sure to use the same email you used to sign up for the event. Then, press "Submit".



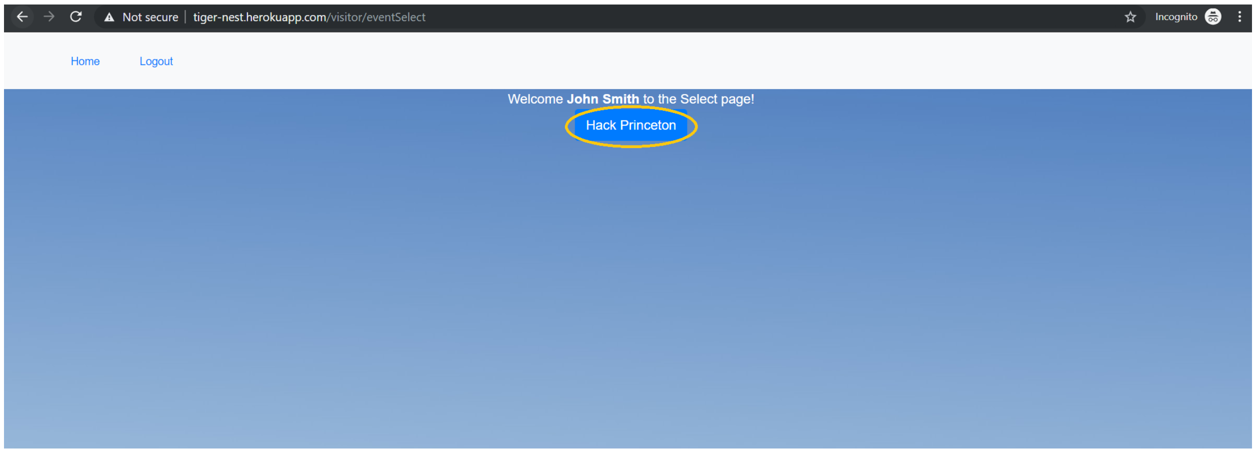
Name: John Smith
Gender: Male
Are you comfortable rooming with someone of the opposite gender?: Yes
University: Georgetown University
Email: js0239159@gmail.com
Password: ••••••••

If your registration is successful, you will be shown a login screen. Type in the email and password you just registered for the site and press "Submit".



Email: js0239159@gmail.com
Password: ••••••••

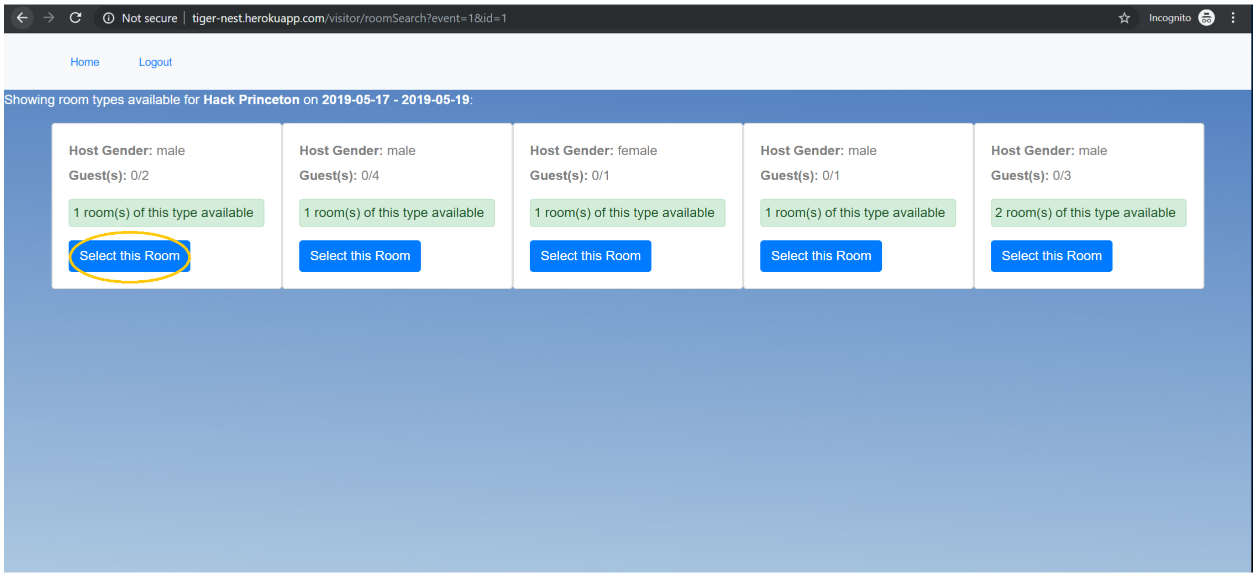
You will be shown a screen with all the events you have signed up for. If you have not yet signed up for a room for the event, the event button will appear in blue. Click on it to see what rooms are available for that event.


Not secure | tiger-nest.herokuapp.com/visitor/eventSelect

Home Logout

Welcome **John Smith** to the Select page!

You will be shown a a set of rooms that are available for that event given your gender and preferences set upon registration. Click on "Select this Room" to reserve it. If no hosts have signed up to host for the event, no rooms will appear. To see how to sign up as a host for an event, please refer to Niranjan Shankar's paper.

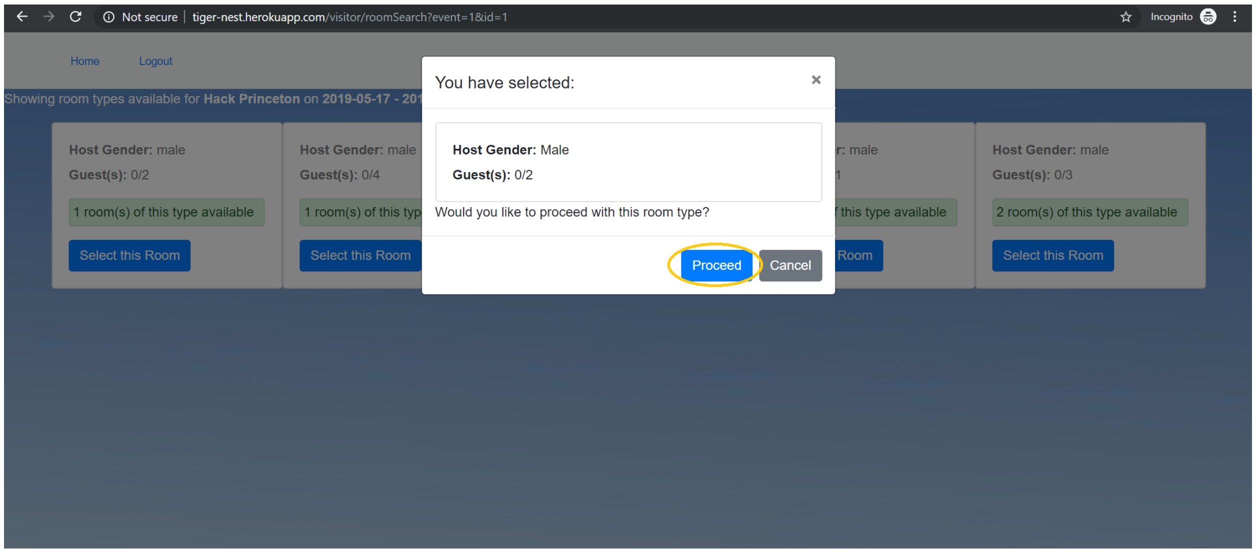

Not secure | tiger-nest.herokuapp.com/visitor/roomSearch?event=1&id=1

Home Logout

Showing room types available for **Hack Princeton** on **2019-05-17 - 2019-05-19**:

Host Gender: male	**Host Gender:** male	**Host Gender:** female	**Host Gender:** male	**Host Gender:** male
Guest(s): 0/2	**Guest(s):** 0/4	**Guest(s):** 0/1	**Guest(s):** 0/1	**Guest(s):** 0/3
1 room(s) of this type available	1 room(s) of this type available	1 room(s) of this type available	1 room(s) of this type available	2 room(s) of this type available
Select this Room	Select this Room	Select this Room	Select this Room	Select this Room

A modal will appear asking you to confirm your selection, you can select "Cancel" which will close the modal and go back to viewing all the rooms, or you can click "Proceed" to lock in your room type.


Not secure | tiger-nest.herokuapp.com/visitor/roomSearch?event=1&id=1

Home Logout

Showing room types available for **Hack Princeton** on **2019-05-17 - 201**

You have selected:

Host Gender: Male
Guest(s): 0/2

Would you like to proceed with this room type?

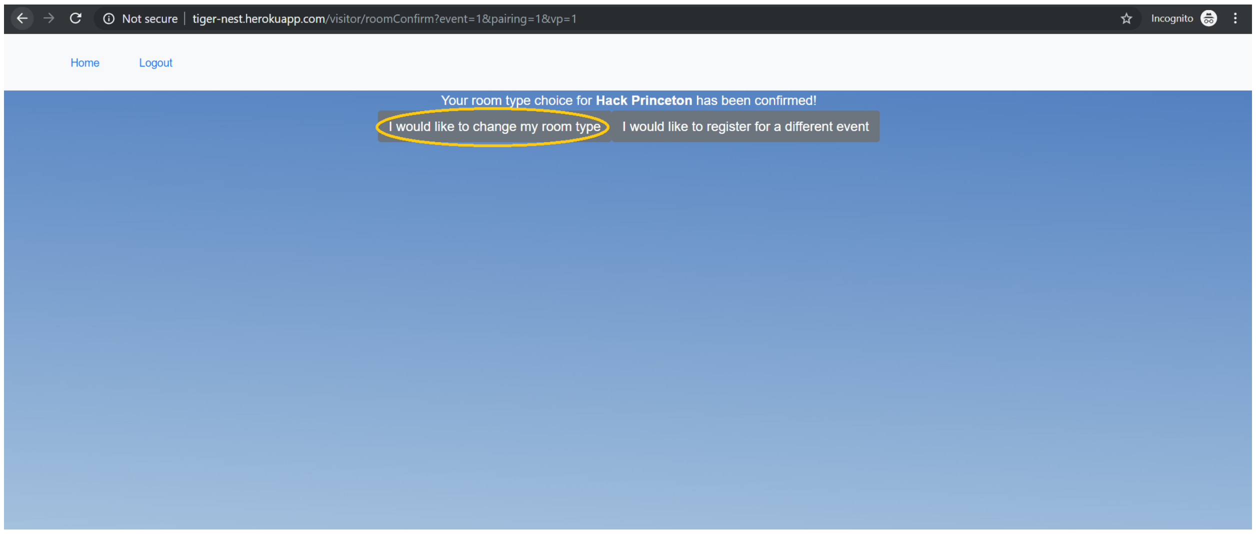
Proceed Cancel

Host Gender: male
Guest(s): 0/2
1 room(s) of this type available
Select this Room

Host Gender: male
Guest(s): 0/4
1 room(s) of this typ

Host Gender: male
Guest(s): 0/3
2 room(s) of this type available
Select this Room

Upon picking a room type, you will be shown to the confirmation screen. From here, you can edit your recent room selection by clicking "I would like to change my room type".

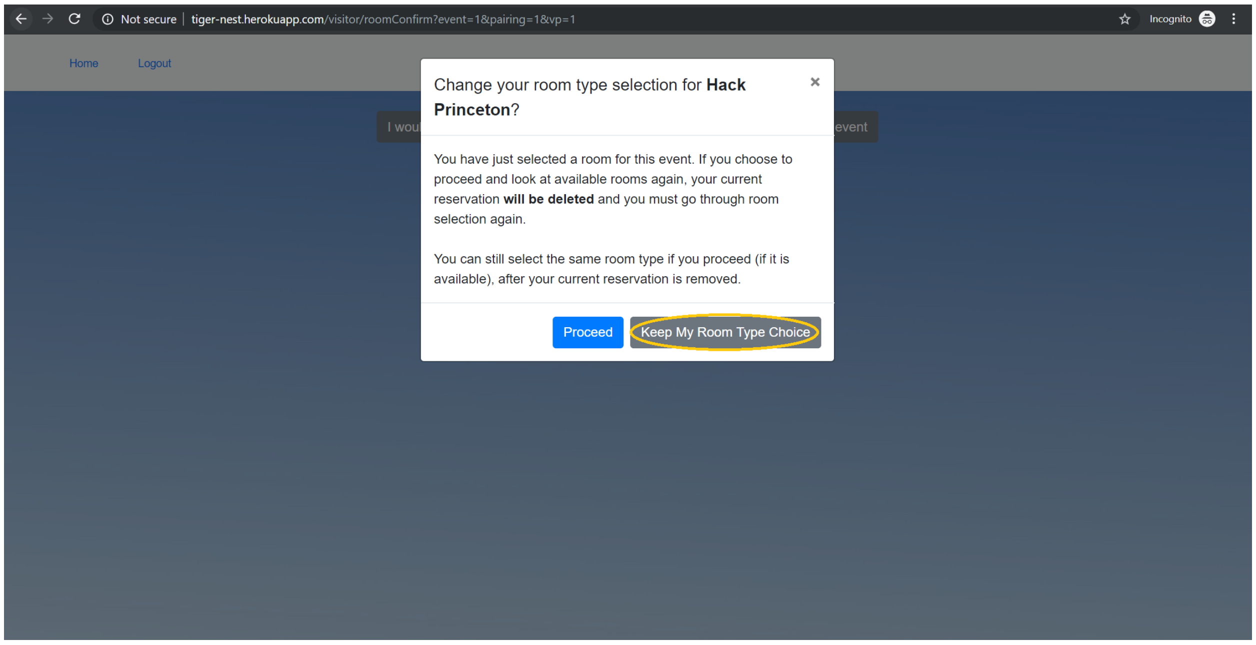

Not secure | tiger-nest.herokuapp.com/visitor/roomConfirm?event=1&pairing=1&vp=1

Home Logout

Your room type choice for **Hack Princeton** has been confirmed!

I would like to change my room type I would like to register for a different event

This will pop a modal asking you to confirm this decision. If you click "Proceed" you will be brought back to the screen showing available rooms, "Cancel" will close the modal and show the confirmation screen again.

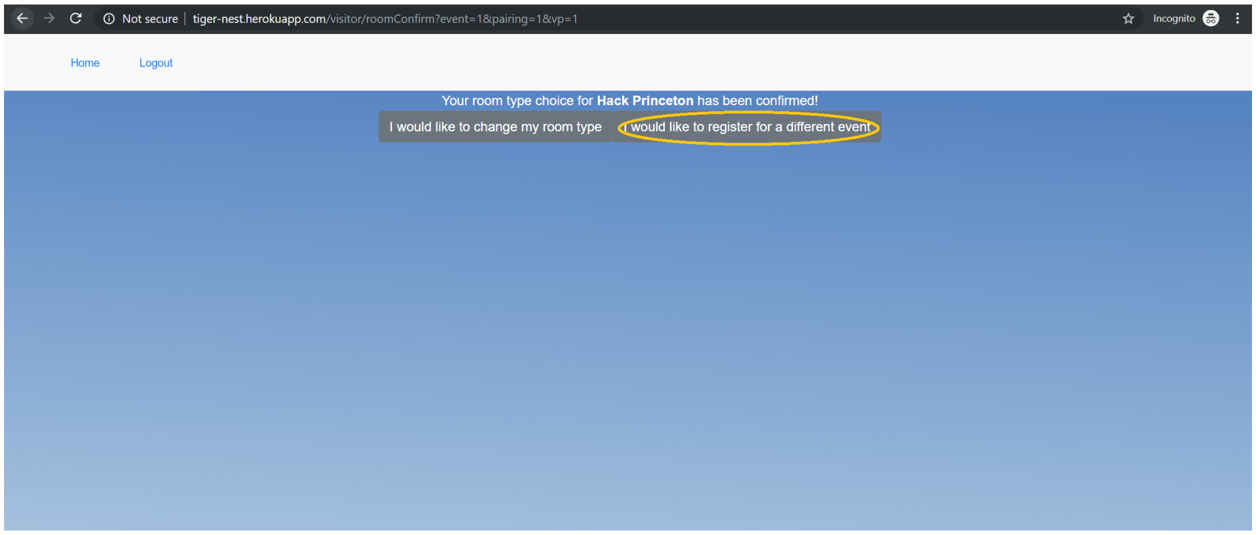

Not secure | tiger-nest.herokuapp.com/visitor/roomConfirm?event=1&pairing=1&vp=1

Change your room type selection for **Hack Princeton**?

You have just selected a room for this event. If you choose to proceed and look at available rooms again, your current reservation **will be deleted** and you must go through room selection again.

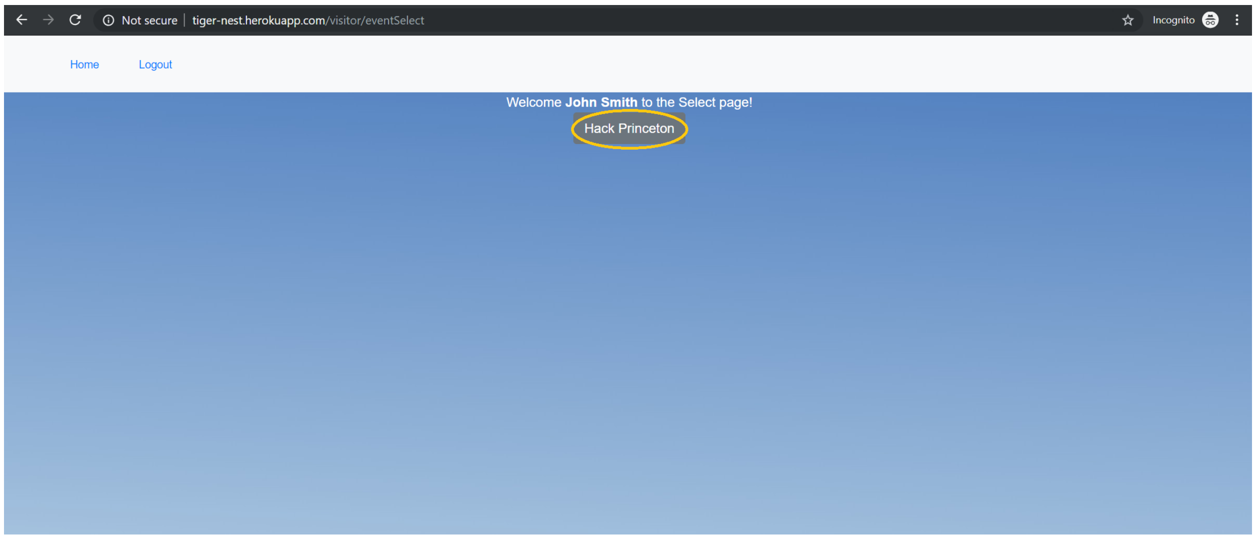
You can still select the same room type if you proceed (if it is available), after your current reservation is removed.

Then, click on "I would like to register for a different event" to be brought back to the event selection screen. Note that the button that was once blue is now grey to indicate you have already signed up for a room for that event.

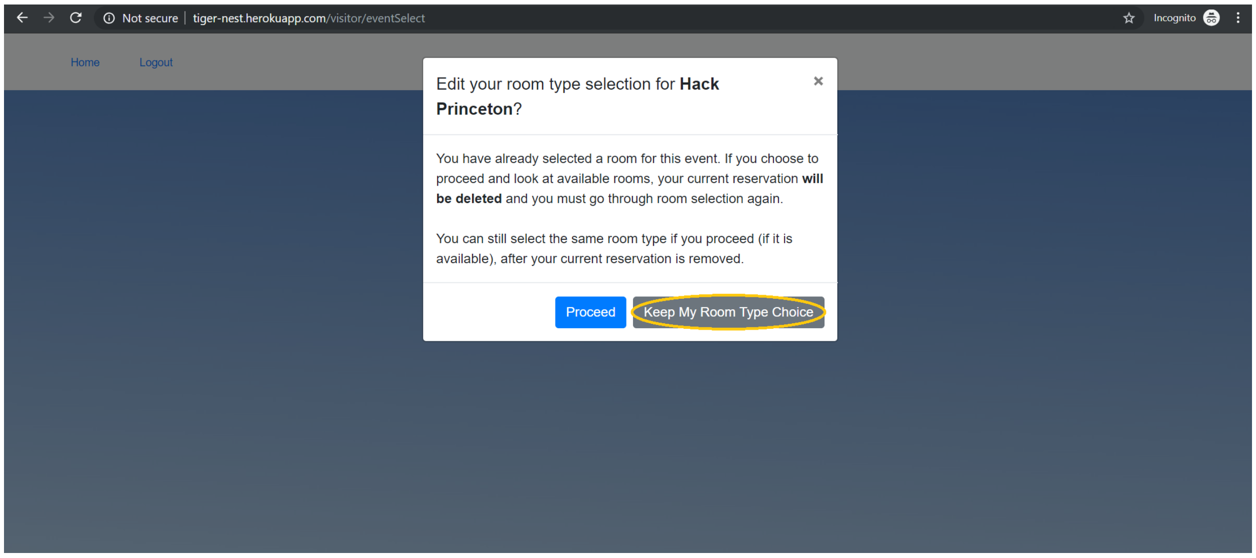

Not secure | tiger-nest.herokuapp.com/visitor/roomConfirm?event=1&pairing=1&vp=1

Your room type choice for **Hack Princeton** has been confirmed!

Home
Logout
I would like to change my room type
I would like to register for a different event


Not secure | tiger-nest.herokuapp.com/visitor/eventSelect
Incognito
Home
Logout
Welcome **John Smith** to the Select page!
Hack Princeton

Clicking on this button will open a modal asking you to confirm your desire to edit your room selection. Similar to the modal shown after room confirmation, clicking "Proceed" will show you available rooms, and clicking "Keep My Room Type Choice" will close the modal and return you to the event selection screen.

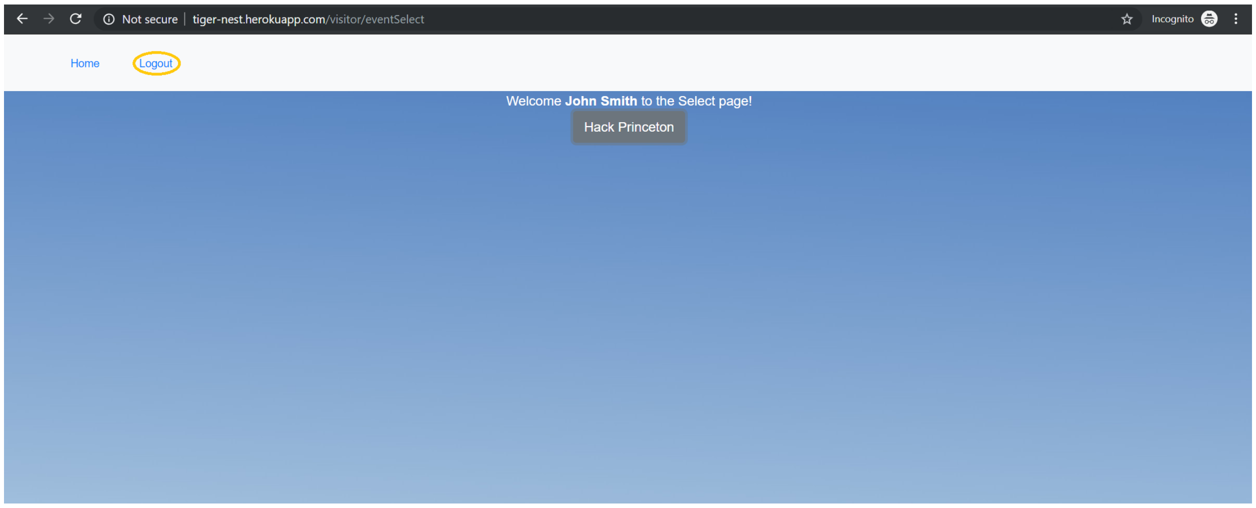


Edit your room type selection for **Hack Princeton**?

You have already selected a room for this event. If you choose to proceed and look at available rooms, your current reservation **will be deleted** and you must go through room selection again.

You can still select the same room type if you proceed (if it is available), after your current reservation is removed.

Then, to logout, simply click the "Logout" button in the navigation bar, which is available on any screen after login.

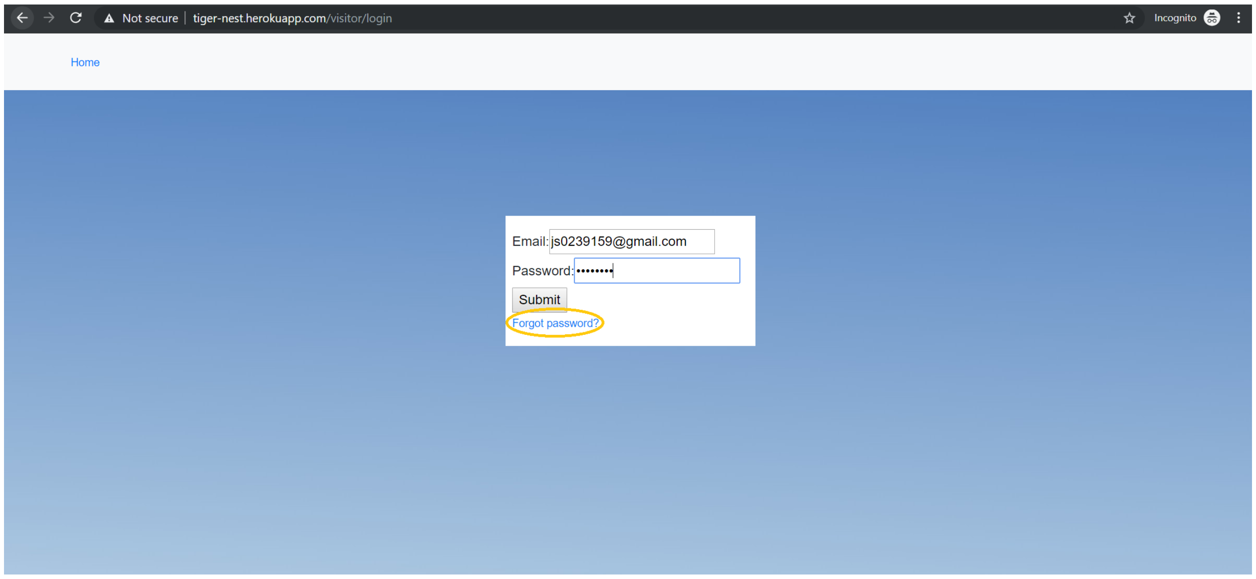


Welcome **John Smith** to the Select page!

This will bring you back to the homepage.

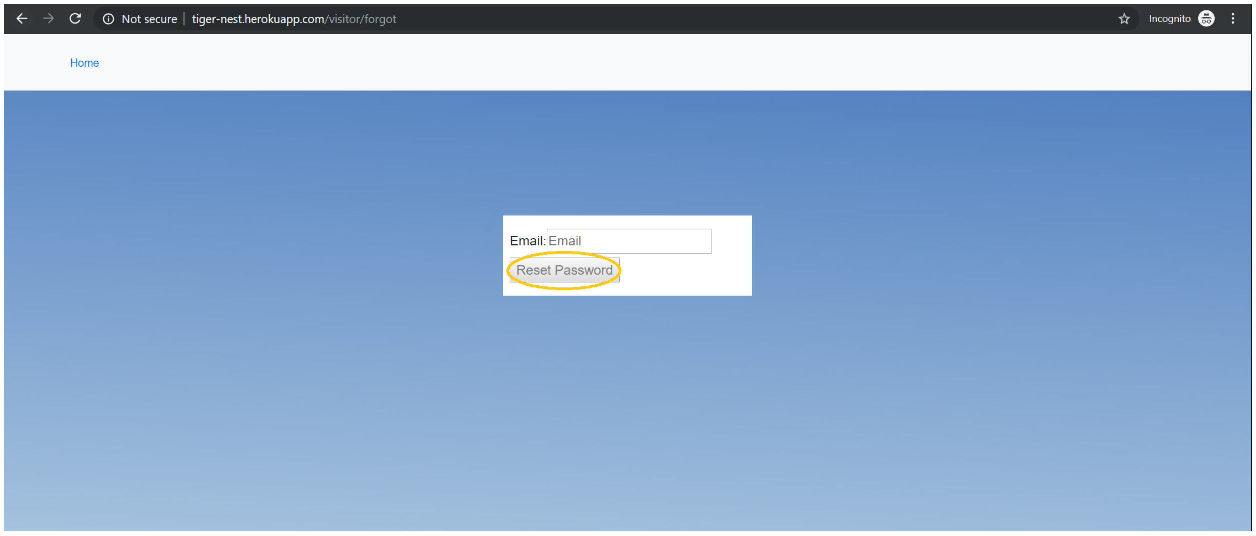
4.3. Forgetting Your Password

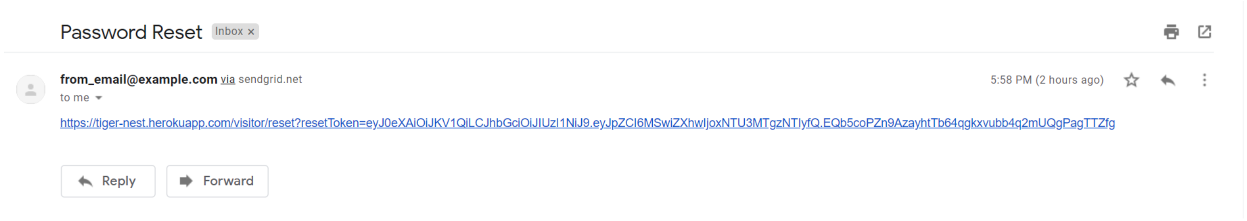
If a user forgets their password when trying to login, they can click on the "Forgot password?" link on the login page.



Email: js0239159@gmail.com
Password: ••••••••

After entering your email, click "Submit" and then check your email for a password reset link.



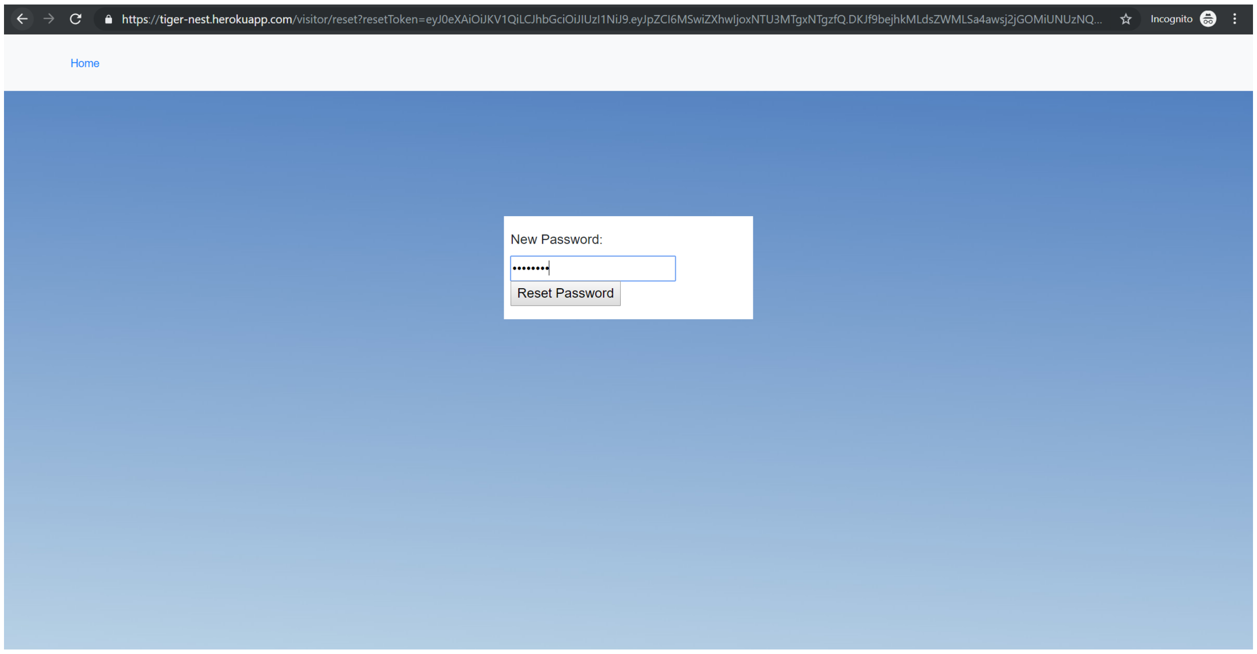

Password Reset Inbox ×

from_email@example.com via sendgrid.net
to me

5:58 PM (2 hours ago)

https://tiger-nest.herokuapp.com/visitor/reset?resetToken=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpZCI6MSwiZXhwIjoxNTU3MTgzNTIyfQ.EQb5coPZn9AzayhtTb64qgkxvubb4q2mUQgPagTTZfg

Following that link will bring you to a page where you can type in your new desired password. Pressing "Reset Password will bring you back to the login screen.


https://tiger-nest.herokuapp.com/visitor/reset?resetToken=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpZCI6MSwiZXhwIjoxNTU3MTgxNTgzfQ.DKJf9bejhkMLdsZWMLSa4awsj2jGOMiUNUzNQ...
Incognito
Home
New Password:
••••••••
Reset Password

5. How It Works

5.1. Technologies Used

Since a requirement gathered from TigerApps was to use React, which is a frontend library, we knew the frontend would also need to be supported by Express, which would run the server-side of the frontend [1]. This is because React is most compatible with Node.js, and Express is a Node.js web application framework. Then, for styling assistance, we used Bootstrap through the Reactstrap package.

Then, on the backend, we used Flask [7] to power the backend API, which uses SQLAlchemy [4] to query a PostgreSQL database [2]. Overall, with the other two user flows in mind, the interaction between these layers can be visualized in Figure 4. The frontend uses the Axios Node package to send requests to the REST API backend.

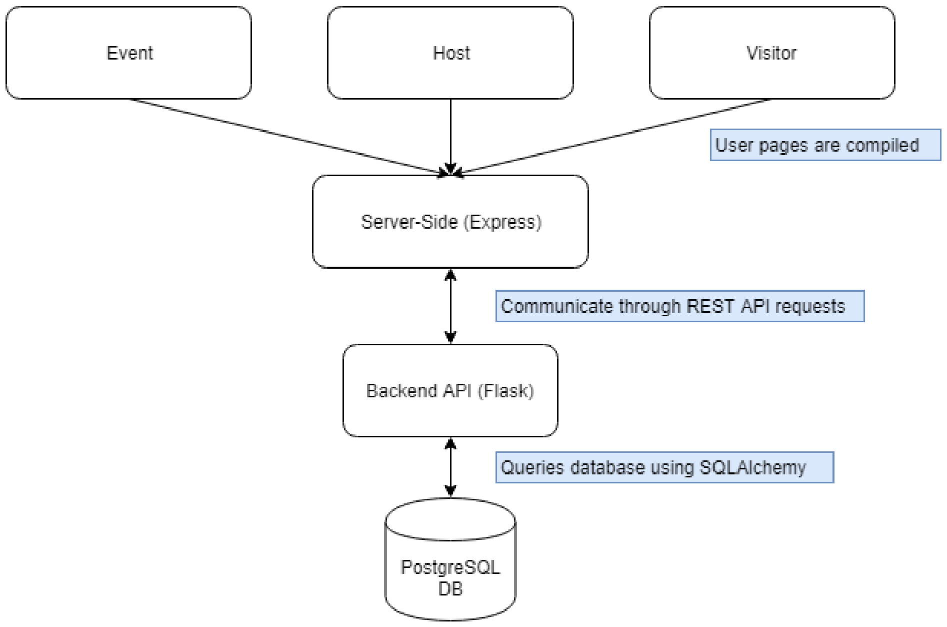


Figure 4: An overview of how all ends of the website interacted.

Another technology choice in this project was to use the NextJS framework for React [10]. Next makes React applications "production-ready" and also has more selective rendering of pages, which is ideal for this project since there are many pages to the website that do not need to be bundled all at once.

For deployment, we used Heroku to separately deploy the frontend and backend portions. As mentioned above, the two communicated using REST requests. Though this was not a requirement gathered from TigerApps, there was some difficulty meeting their AWS deployment requirement, which will be discussed below.

Since this paper focuses on the visitor flow, certain technologies utilized only in other user flows (ex: CAS Login) will not be covered in the following sections.

5.2. Navigating Pages

Because NextJS framework organizes the codebase into "pages," I will show how the visitor navigates from one page to another. Each page has its own file, with the exception of the navigation bar which is a component and rendered on every page post login.

5.3. Initial Interest Pages

In order to ensure the visitors signing up for rooms were verified by the event organizer as event attendees, we created a separate link for visitors to sign up for the event. The flow of pages is as shown in Figure 5.

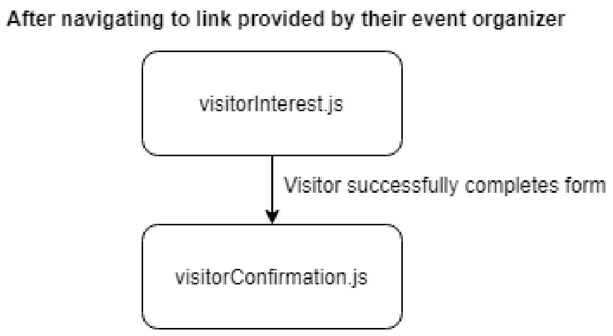
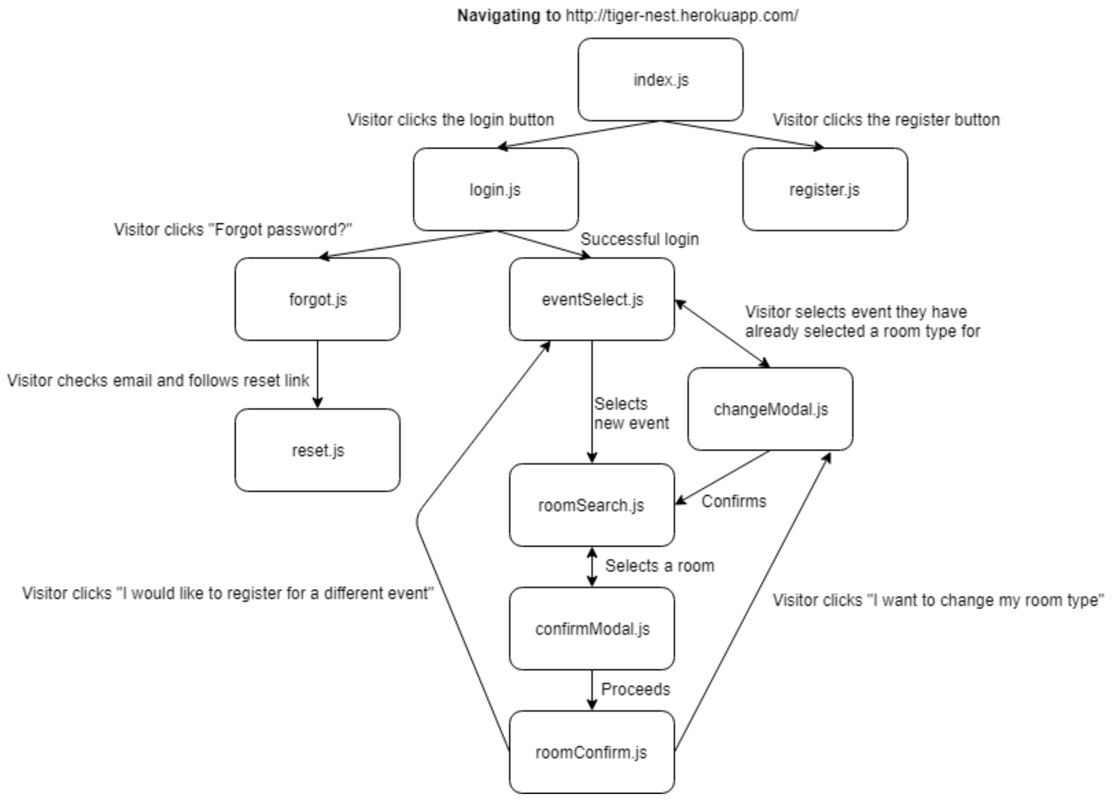

After navigating to link provided by their event organizer
visitorInterest.js
Visitor successfully completes form
visitorConfirmation.js

Figure 5: Diagram of how the visitor will navigate between the interest pages.

If the visitor does not fill out this form, no events will appear after they login. Visitors will then at a later time navigate to TigerNest to select their room.

5.4. Navigation from the Homepage

This section illustrates the main flow of the visitor as they navigate the site. The diagram in Figure 6 shows how the visitor might get from one page to another and how to do so.



Navigating to http://tiger-nest.herokuapp.com/
index.js
Visitor clicks the login button
Visitor clicks the register button
login.js
register.js
Visitor clicks "Forgot password?"
Successful login
forgot.js
eventSelect.js
Visitor selects event they have already selected a room type for
Visitor checks email and follows reset link
changeModal.js
Selects new event
reset.js
roomSearch.js
Confirms
Selects a room
Visitor clicks "I would like to register for a different event"
Visitor clicks "I want to change my room type"
confirmModal.js
Proceeds
roomConfirm.js

Figure 6: Diagram of how the visitor will navigate between the TigerNest pages to select a room.

Because of the nature of the NextJS framework, each page can be thought of as its own React application that is selectively rendered when it is needed. For this reason, there is little to no nesting of components that needs to be discussed.

5.5. Custom Login

Because visitors are not Princeton students, they cannot login with CAS. Furthermore, after requirements gathering, I learned that not all universities use GSuite (which is what Princeton currently uses), and so using the Google Login API would not be able to cover all potential visitors. Thus, I needed to create a custom login for TigerNest.

To do so, I would have visitors register for the site and save a hashed version of their password to the database. Their emails would serve as their usernames. Then, after the visitor successfully logged in (correctly entering their username and password), a JSON Web Token would be generated and stored in their browser. Then, when the visitor wanted to access certain pages with user-specific information, their browser would send that token as a form of authentication. Visitors with invalid tokens would not be able to retrieve information from the database. Then, when the user wants to logout, the token is deleted from their browser.

Another feature to login is allowing the user to reset their password if they forget it. This uses SendGrid API [9] to send the user an email with a password reset link that they can then follow to reset their password.

5.6. Challenges Faced

One of the largest overall challenges of this project was the use of new technologies. For me personally, I had never used Flask or SQLAlchemy. Additionally, though the NextJS framework made it much easier to build out the website, there was a significant learning curve that came with using it. For example, I previously used Create-React-App, and adding Reactstrap to that type of project is a quick install. However, on NextJS, because of the way bundling works, I had to configure and install a certain loader just to load the Reactstrap components. This was new to me and took some time to figure out how NextJS worked. However, overall I am happy with the choice to use NextJS since it sped up rendering and simplified the codebase.

In terms of learning new backend technologies with Flask and SQLAlchemy, I found that working in conjunction with Niranjan helped me learn more quickly and effectively. One difficulty we encountered was that since we were sharing a database and backend API, there were some errors that arose from miscommunication and misalignment in the files we were working on. We used Github for version control for this project, but in the future it may be better to push our changes more often and also communicate offline more frequently to avoid this problem.

Additionally, this was my first time building a custom login from scratch. In the past I had only used Google's Login API which is much simpler to install and add. I had to learn about password hashing and also password reset methods which were previously all covered by a third party.

Lastly, one of the challenges we were unable to conquer was deployment using Amazon Web Services. This was a requirement given by TigerApps but after a significant amount of fiddling from both me and Niranjan, we were still unsuccessful. However, after speaking to other TigerApp owners, I learned that the AWS requirement is new and I was unable to get assistance from them as they were also deployed on Heroku. Due to the time crunch we encountered during the semester, we still had to ensure that the website was deployed and opted to use Heroku.

6. Evaluation

In order to test the usability of the application, I asked users to perform a series of tasks and observed their behavior. I followed a standard method of evaluation where I gave little to no information to the users outside of the tasks they were given, and took notes on their actions [6]. I also asked users to speak aloud what their thought process was and took notes on that as well.

I was able to get three users to test the application, two being students of other universities (not Princeton) who had experience attending other overnight events at other schools, and one Princeton student who is a member of HackPrinceton.

6.1. User Evaluations

I asked users to perform two tasks and after each task I asked them for comments and what they found most difficult or confusing when performing the task. The task list is shown in the appendix in Figure 7. First, I noticed no difference in the speed of use between the Princeton and non-Princeton student users. All three users found the homepage clear to navigate and had no trouble finding which section of the site was for them. The Princeton user expressed some concern in Task 1 over possible misuse of the form for users to sign up for the wrong event mistakenly or maliciously. They suggested a possible remedy to this issue by generating a unique link for each event, which would prevent a visitor from signing up for any events that were not their own. Overall, for Task 1, users only expressed some aesthetic critiques, which will be worked on in future iterations.

For Task 2, two users commented afterwards that the registration form should have a drop-down list of universities instead of being a free-form string. I also asked all users afterwards about their understanding of the "Are you comfortable rooming with someone of the opposite gender?" question to ensure clarity. All three expressed the same understanding that I had which was confirmation that users would get the preferences they desired.

One user also expressed a desire to see the genders of the other guests in the room, which would add even more clarity to the room search page of the site. Additionally, all users were able to deduce that they could change their room type choice from the event selection screen by reselecting the same event. However, one user commented that the modal that appears is a bit wordy and that it may deter some people from reading it fully and understanding what they are doing.

Lastly, for Task 3, when I met with the Princeton user, they at first clicked the "Register" button to try to change their password, instead of "Login." Since neither is currently very explanatory, I am still working on possible wording to make this more clear to users. Additionally, after resetting their password, the user stayed on the Password Reset page, and commented that it would be smarter to push the user to the login screen instead. I fixed this before showing the other two non-Princeton students and both were able to easily login again after being rerouted to the login page after changing their passwords.

I also asked users to attempt to perform Task 2 without performing Task 1, which would not allow users to progress past step 3 of Task 2. This is due to no events showing for the users, but because I write a message when no events are available to the users, all three users then asked me for a sign-up page, which was intentional. Overall, users commented that the overall website aesthetic could be improved, and that room cards could be more informative by showing the genders of guests that have already signed up, which has been noted for future work. All three users commented that the site was simple-to-use and very fast, and one said they would attend an event that had this form of sign up for visitors.

7. Conclusion and Future Work

Overall, I learned a significant amount about web application development and new technologies like Flask and NextJS through this project. Additionally, I gained a better understanding of interaction design practices in terms of requirements gathering and evaluation methods. My previous experience with ground-up web development was limited to COS333 before, and I now feel as though I have a better grasp of the field and even improved my preexisting knowledge of React, which I had used before. Something I regret during this project was not starting technical development earlier and being unaware of the time crunch that comes with a semester-long project.

Going forward, I believe the visitor experience could be improved in terms of showing more information about room types as suggested by a user during testing. We will also meet with the head of TigerApps, and other student organizations to gather feedback and work on future iterations. Another change that could be made to the site is to deploy to AWS instead of Heroku as Reilly Bova requested.

8. Acknowledgements

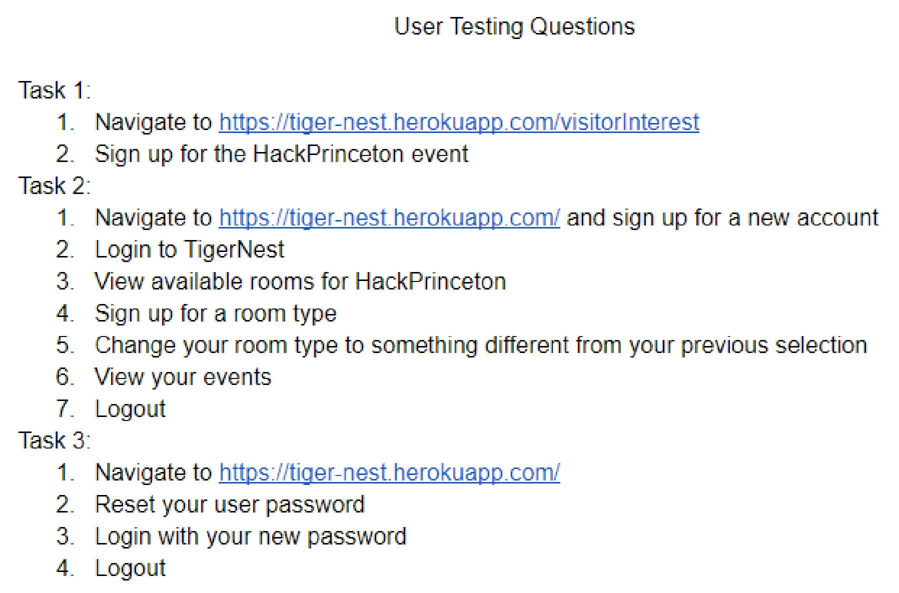
I would like thank my advisor, Dr. Dondero, for his guidance and feedback this semester and also as a lecturer in my COS333 class in the fall semester where I worked on ground-up web development for the first time. I would also like to thank my peer Niranjan Shankar, who worked on the event organizer and host flows of the application, for all this hard work, dedication, and help. This project would not exist without the student organizations that it targets, so I would also like to thank the student leaders of Princeton Debate Panel, Princeton Model UN, TigerLaunch, Club Swim, Envision, and HackPrinceton, as well as head of TigerApps, Reilly Bova, for their time. I would also like to thank Alex Yue for his time spent answering my questions about NextJS, React, and writing a custom login.

9. Appendix

9.1. Code

Due to the lengthiness of the code, please navigate to `https://github.com/mjyuen/TigerNest-Frontend` to view the frontend code behind the visitor flow. Additionally, to view the backend code shared by this project and Niranjan Shankar's project, please navigate to `https://github.com/mjyuen/TigerNest-Backend`.

9.2. Tasks for Usability Testing



User Testing Questions

Task 1:
1. Navigate to https://tiger-nest.herokuapp.com/visitorInterest
2. Sign up for the HackPrinceton event

Task 2:
1. Navigate to https://tiger-nest.herokuapp.com/ and sign up for a new account
2. Login to TigerNest
3. View available rooms for HackPrinceton
4. Sign up for a room type
5. Change your room type to something different from your previous selection
6. View your events
7. Logout

Task 3:
1. Navigate to https://tiger-nest.herokuapp.com/
2. Reset your user password
3. Login with your new password
4. Logout

Figure 7: List of tasks users were asked to perform for usability testing.

References

[1] "Node.js web application framework." [Online]. Available: https://expressjs.com/

[2] "The world's most advanced open source relational database." [Online]. Available: https://www.postgresql.org/

[3] A. Athalye, "Algorithms in the real world: Host matching," Sep 2015. [Online]. Available: https://medium.com/hackmit-stories/algorithms-in-the-real-world-host-matching-62baf4d7c165

[4] M. Bayer, "The python sql toolkit and object relational mapper," 2006. [Online]. Available: https://www.sqlalchemy.org/

[5] C. Chow, "Lessons learned from hackprinceton's host matching (part 1)," Dec 2017. [Online]. Available: https://medium.com/hackprinceton/lessons-learned-from-hackprincetons-host-matching-part-1-ad3cf1ef7e24

[6] Y. Rogers, J. Preece, and H. Sharp, *Interaction Design: beyond human-computer interaction*, 3rd ed. Wiley, 2011.

[7] A. Ronacher, "Flask," 2010. [Online]. Available: http://flask.pocoo.org/

[8] F. O. Source, "React – a javascript library for building user interfaces," 2019. [Online]. Available: https://reactjs.org/

[9] Twilio, "Email delivery service." [Online]. Available: https://sendgrid.com/

[10] Zeit, "Next.js." [Online]. Available: https://nextjs.org/