Rooted

Sprint 1 Demo (Backup)

October 11, 2019

Team 2

Julien San Diego, asandieg@purdue.edu
Cyrus San Santiago, santiac@purdue.edu
Ken Sodetz, sodetz@purdue.edu
Ben Malone, malone74@purdue.edu
Abigehl Slattery, aslatter@purdue.edu

Notes:

- Github repo
- Sprint 1 Slide deck (for Demo)
- On the commits in the repo, it only shows 4 contributors instead of 5. Julien’s git settings were not updated. Thus it shows “Julien San Diego authored and Julien San Diego committed” on the commits page.

Member Accomplishments

Abigehl

During Sprint 1, I focused on the front end and integration of the front end to the backend. I worked with both Ben and Cyrus on this and my main contribution was the profile page and the editing of the profile page. I also changed some of the CSS across pages. Aside from the frontend and frontend integration, I spent a lot of time looking at the backend and the interactions between the two. I created the Schema for users in the database as well as redefined the account object within the framework. In the end a large portion of the time I spent on this sprint I spent learning. While I have done web development many times before, the general types of web development I did were without frameworks, so learning angular has been like relearning web development. While I have come to understand a number of things about Angular, I still have quite a distance to go.

Focusing on the Profile page and what I accomplished there, I created a basic static Profile page that had hardcoded values. I used the Design Document as a reference and created a very similar look. From there I studied the structure and communication between the two layers. This made it possible to obtain a list of the groups a user is a part of. Realizing I had left out the hidden attribute of the entities, I added in check boxes and checked relevant documentation on how to retrieve and store values in forms using radio buttons. Then I realised that the account model did not contain all of the necessary components, so I updated that file to contain all of the attributes a user has. After a while of debugging I realized that the database schema (specifically the columns in the user table) had never been updated after it had been defined for login. So I went and redefined that portion of the schema. The code I had created for this page still was not pulling the correct data, so I spent a number of hours debugging. This consisted of emptying and repopulating the database, printing out the communications between the two and checking all of my code. After hours of debugging, Julien stepped in and used postman to determine that the page was not receiving the correct message even though it was able to print out the proper response. This was unable to be fixed until this week. This profile page also served as a blank template to display other user's data. While this page was unable to display the attributes a user had, it was still able to serve as an edit page and when users update the information in the fields, it will update the values in the database.

Some of the other contributions I made are briefly mentioned in the previous paragraph but for clarity I will mention them here. I updated the model for accounts and created a model containing information to be updated in a user's account. I created all the underlying methods for editing a user's account. I created the form on the component portion of the html page for editing account.

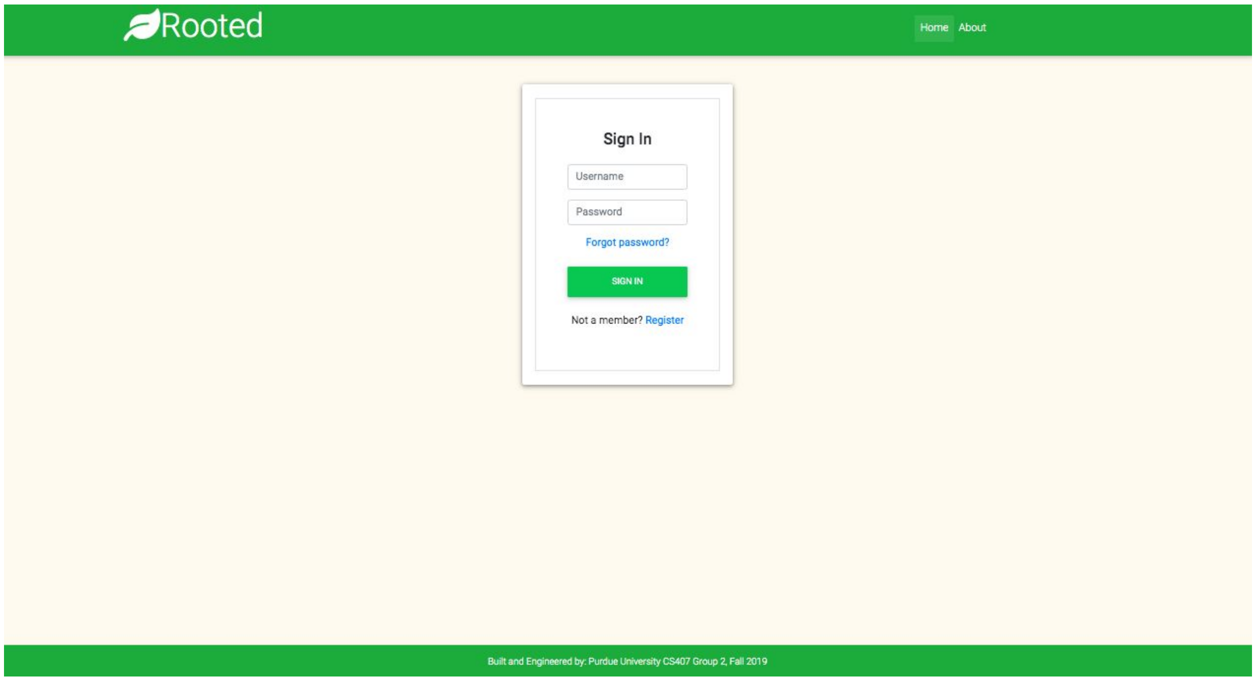
The only other contribution I made that has not been connected to underlying user stories yet is some functionality that I ported from a previous project. It's a simple drag and drop function for uploading files from users' computers. This will be attached once all underlying work for uploading files from computers is implemented.

Ben
For Sprint 1, most of my work was focused around frontend and integrating the pages I had created with the work others had done for the backend. I spent a majority of my time working on, alongside Cyrus, the frontend for displaying a Rooted tree which is the main page a user sees when they click one of Trees they are apart of. My main goal with this feature was to provide a temporary display for all the users in a Tree as well as provide a UI to both report this user and visit their profile all from the group page. My next main focuses were the reporting functions for our site which included both reporting users and reporting groups. Ken completed the backend for both of these tasks and my job was to create the frontend and link my pages to Ken's backend routes through a service. I was able to successfully implement the frontend for both reporting groups and users, however I was only able to successfully link reporting groups. In order to implement reporting a group, I passed the tree name variable from the tree's home page to the report group page and sent a request from there. The requests for both reporting users and groups were quite similar, a group required the name of the tree/group being reported, the reasoning for the report, and the reporter's username. For a user report, it required the user to report and the reason for the report.

The biggest challenge I encountered with this feature was determining how to navigate the user to the profile page of the user whose name they clicked on/reporting the proper user whose name was clicked on. The issue here was that despite displaying all the names of the users properly, I was unable to determine which name had been clicked on when it came to display the modal with the "View Their Profile" and "Report This User" buttons. Despite clicking the proper user's username, the modal was unaware of the name clicked and so was unable to pass this information to the next page. This is an issue because this prevented me from redirecting the user to the proper profile page (The frontend for viewing a user's public profile was never finished, so even if I did link to the proper user's profile, nothing would display). It also prevented me from properly reporting users to the group admins because my request lacked a username and thus failed each time. Without the proper username, the request would fail each time since username was a required header. If I were to hardcode the username into the request it worked properly, so the issue I need to solve moving forward is how to maintain the username between the modal and the report user page.

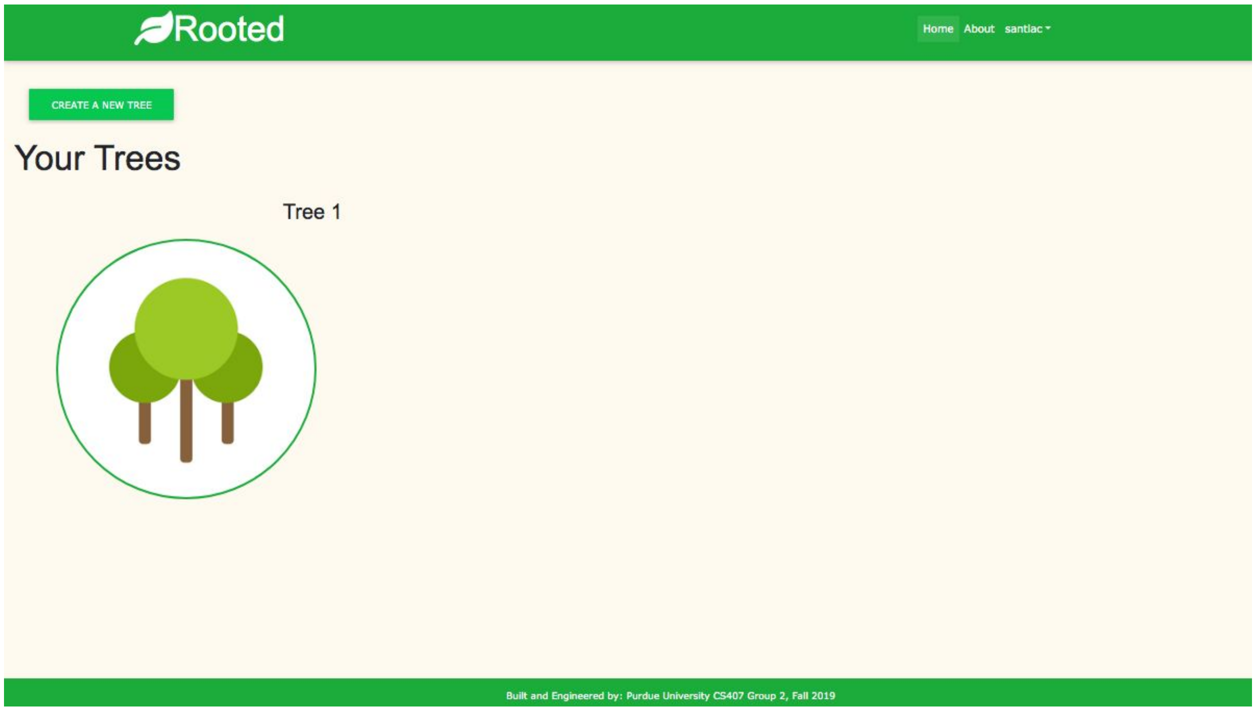
Cyrus

For Sprint 1, I mostly worked on both the frontend and backend for stories, as well as the integration of those components. I was mainly responsible for implementation of the boilerplate code that got the app running with barebones components. Such components include the navbar, corner logo with navigation to the main page, navigation to the about page, navigation to our github repo, and CSS to style the site with a cohesive theme.

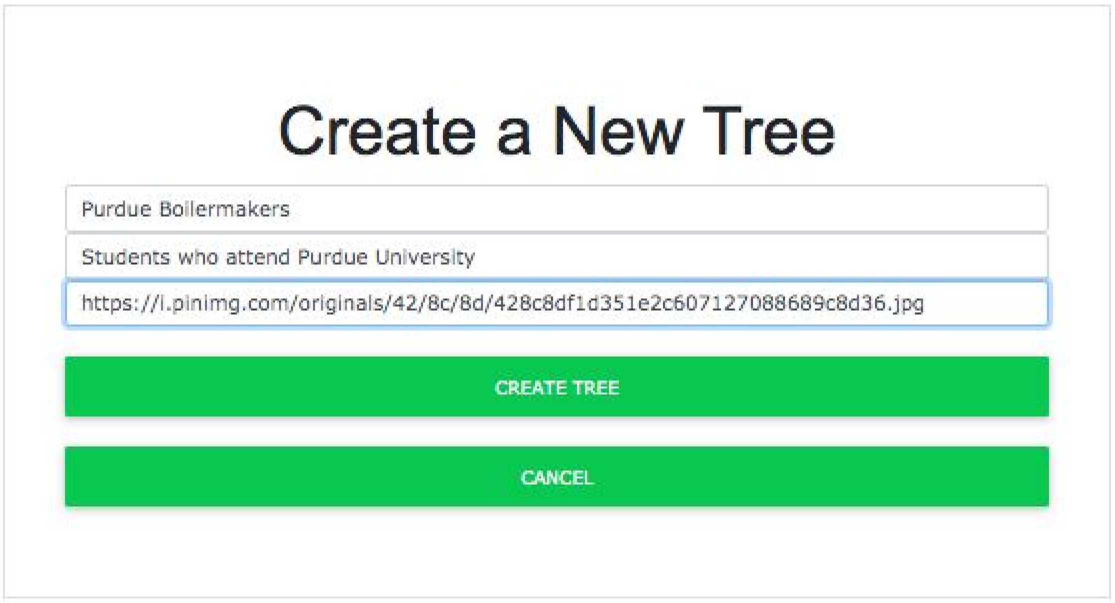


This is what kicked off development. From there my fellow group members were able to run the app on their computers and slowly construct more and more features piece by piece.

At this stage, I took on development for the page that displays all of the trees a user belongs to, functionality to create new trees, and routes that link to those specific trees' pages. I created the schema for trees alongside backend implementation to store, retrieve, and update data related to trees. Using this I created a frontend display to cleanly populate the page with large clickable cards that contain images of the tree's group photo, and the name of the tree.


Rooted
Home
About
santiac
CREATE A NEW TREE
Your Trees
Tree 1
Built and Engineered by: Purdue University CS407 Group 2, Fall 2019

Clicking the "create a new tree" button initiates a pop up modal form, where users can input info for new trees.

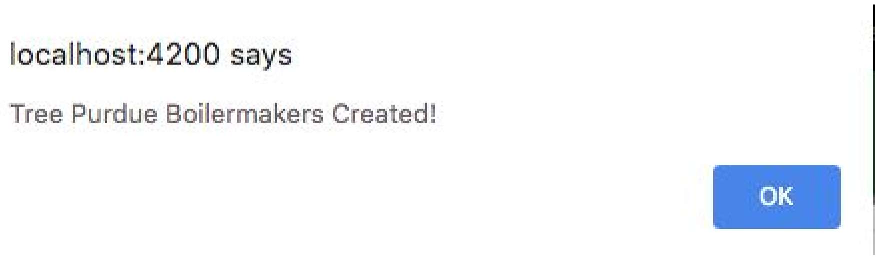

Create a New Tree

Purdue Boilermakers

Students who attend Purdue University

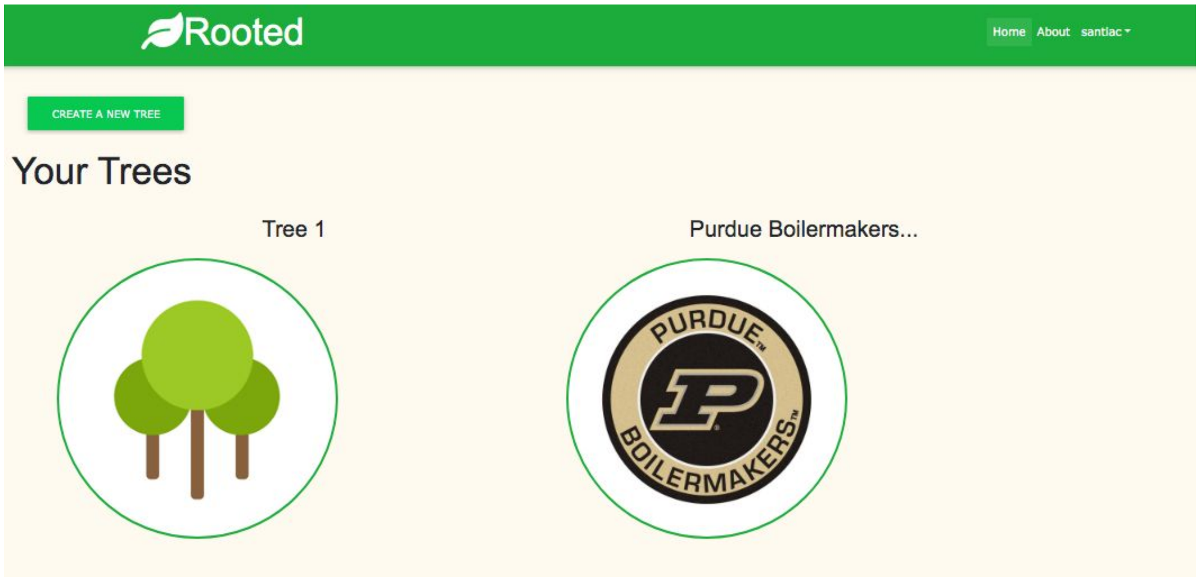
https://i.pinimg.com/originals/42/8c/8d/428c8df1d351e2c607127088689c8d36.jpg

Successful submission will refresh the page and the newly created tree will be present, with the user designated as the founder.

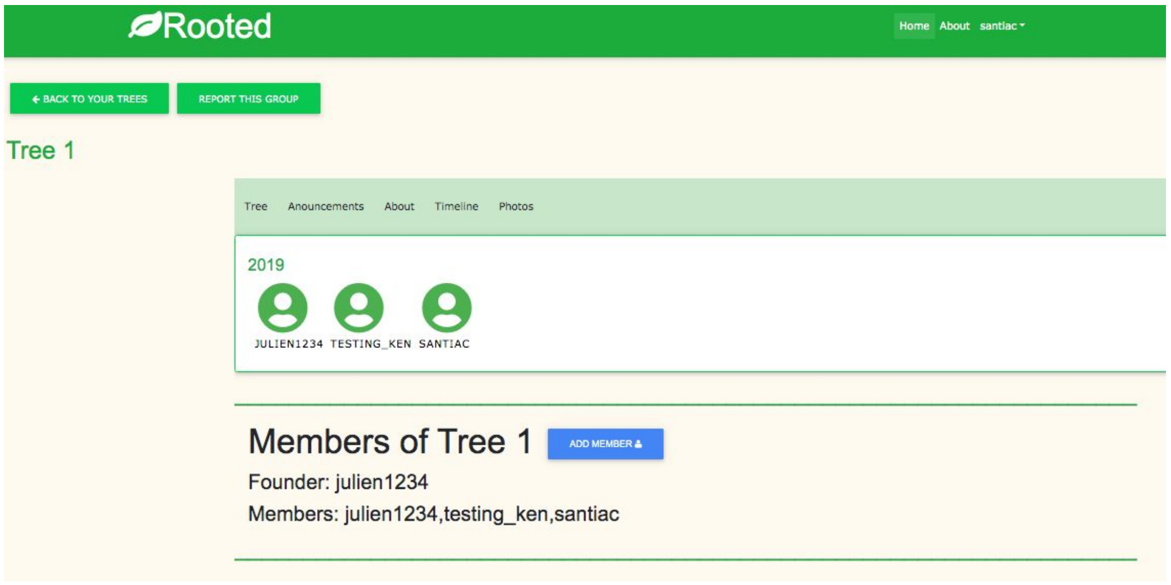


localhost:4200 says

Tree Purdue Boilermakers Created!



Clicking on a tree's card will route to a unique page dedicated to that tree and its members. The implementation and design of this specific tree page was handled by Ben, who I partnered with during integration.

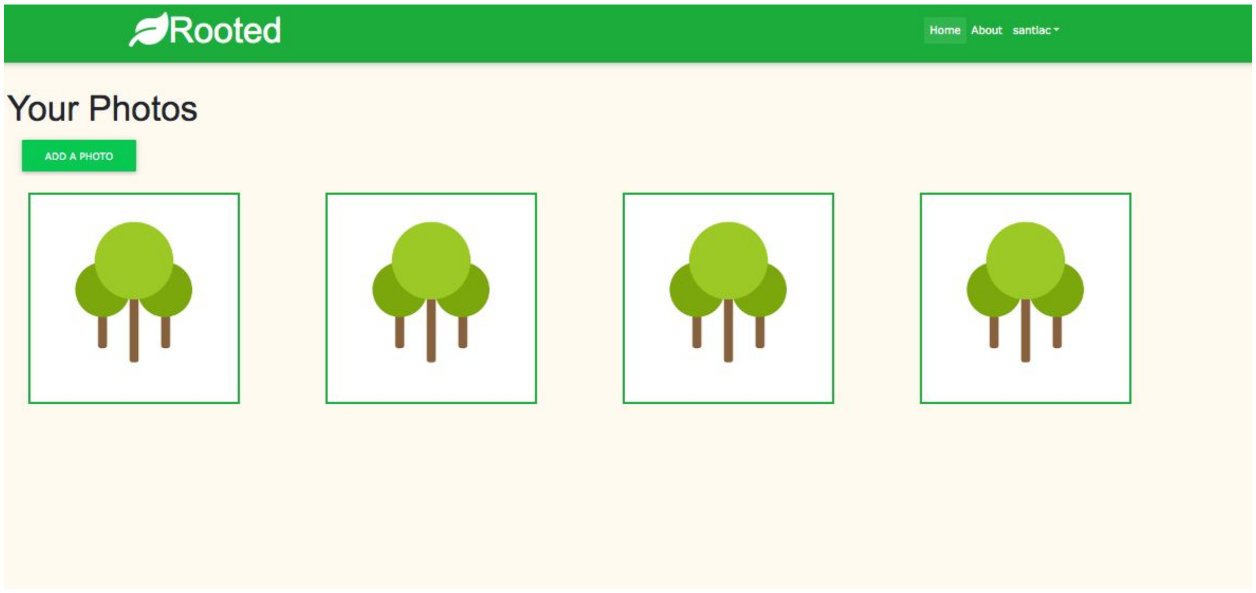


Members of Tree 1

Founder: julien1234

Members: julien1234,testing_ken,santiac

After this I took on development for the functionality to upload profile pictures as well as photos to a user's photo library. I was successful in calling the backend to retrieve all of the pictures in a user's photo library, and can even properly display those photos on the photo library page.



However, I really struggled and faced many challenges with the ability to upload the photos themselves. The image depicted above uses hardcoded images that are manually entered into my photo array in the database. I initially attempted to create a button that upon being clicked would initiate a prompt for a user to upload a file. This prompt successfully runs but submission of a file fails to be stored anywhere. After wrestling and wrestling with this, I turned to an alternative solution of storing the photos not as image files, but as URLs to images in the form of strings. I then created a modal form for users to input a string, that I would simply add as an entry to their array of photo URLs. For reasons unknown, this also failed.

Julien

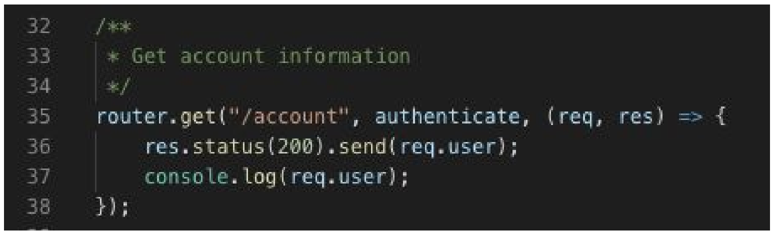
For Sprint 1, I mostly worked on the backend with Ken. Ken, however, was more technically skilled in the backend while I had little experience. Thus, he worked more on the backend user stories, especially with setting up the project. I mainly worked on the routes for uploading images , retrieving a specific tree information, and finding a user based on his or her username. I faced many challenges throughout this sprint. One of the challenges was backend. While I had done some backend coding, I have always struggled starting a project. However, once I had gotten the hang of it, I was able to better understand and remember how to build route, use Postman, and test if my routes work.

Another challenge I encountered was a CORS issue. I had never experienced it before and didn't quite understand how to fix it. Although I searched up some ways, I wanted to be careful to ensure that any fix won't cause any other issues. But, even when I tried some solutions I found online, I still kept getting the problem.

A challenge I encountered that took up so much time was building the route for retrieving a specific tree information. The route I built took in the parameter **treeid** as a header. When I tested it on postman and made the request, the request sends back the correct information for the specific tree I chose. I had tested it for multiple **treeids**. However, when I integrated it with the frontend, the route was not being called. I spent roughly a day trying to figure out why, and had even talked to someone on the web dev team to ask about the problem. Turns out, under Access-Control-Allow-Headers, **treeid** was not registered, and therefore, when I make a call on the frontend, **treeid** was not being accepted as a header parameter.

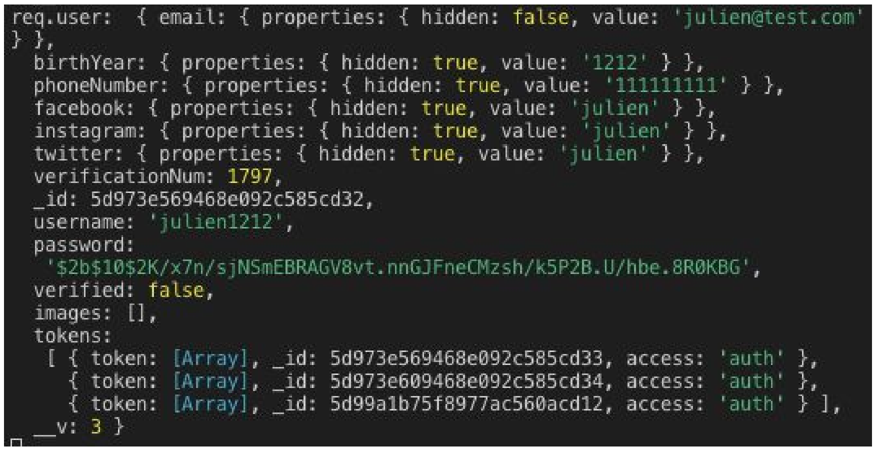
Another challenge was trying to send the correct response back in the /account route. This route is supposed to send back all information about the user. Interestingly enough, the output for console.log() is not the same as the response that is sent back.

Backend route for /account



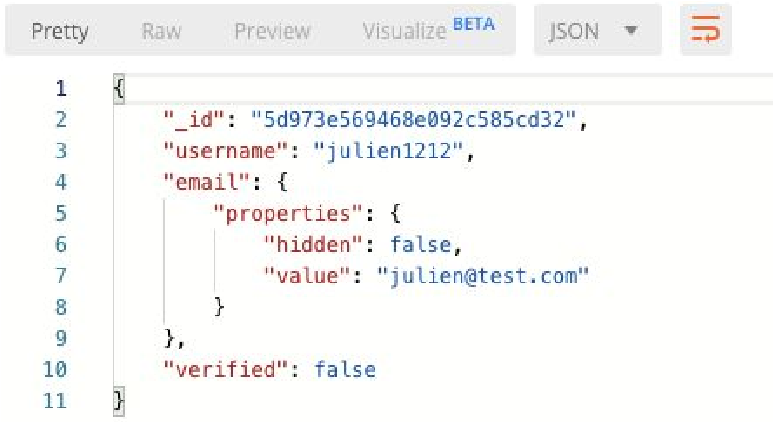
```
32  /**
33   * Get account information
34   */
35  router.get("/account", authenticate, (req, res) => {
36      res.status(200).send(req.user);
37      console.log(req.user);
38  });
```

Console.log(req.user) - line 37



```
req.user:  { email: { properties: { hidden: false, value: 'julien@test.com'
} },
  birthYear: { properties: { hidden: true, value: '1212' } },
  phoneNumber: { properties: { hidden: true, value: '111111111' } },
  facebook: { properties: { hidden: true, value: 'julien' } },
  instagram: { properties: { hidden: true, value: 'julien' } },
  twitter: { properties: { hidden: true, value: 'julien' } },
  verificationNum: 1797,
  _id: 5d973e569468e092c585cd32,
  username: 'julien1212',
  password:
   '$2b$10$2K/x7n/sjNSmEBRAGV8vt.nnGJFneCMzsh/k5P2B.U/hbe.8R0KBG',
  verified: false,
  images: [],
  tokens:
   [ { token: [Array], _id: 5d973e569468e092c585cd33, access: 'auth' },
     { token: [Array], _id: 5d973e609468e092c585cd34, access: 'auth' },
     { token: [Array], _id: 5d99a1b75f8977ac560acd12, access: 'auth' } ],
  __v: 3 }
```

res.status(200).send(req.user);



```
{
    "_id": "5d973e569468e092c585cd32",
    "username": "julien1212",
    "email": {
        "properties": {
            "hidden": false,
            "value": "julien@test.com"
        }
    },
    "verified": false
}
```

As shown above, the response received and the output on the console are different. This issue prevented Abigehl from completing the task on viewing another user's profile as the information is needed is only being passed partially, such that the response she gets is not complete, preventing her from mapping the rest of the information needed.

Ken

For sprint 1, I was tasked with setting up the backend and the database, as well as contributing to and testing the backend.

Setting up the database was more challenging than the last time I had done it, mainly due to the fact that MLab was purchased my Mongo, and I was now forced to use MongoAtlas, which in my opinion, is a mess of a process to go through in comparison to MLab. While technically more full featured, I definitely thought that MLab was much less painful to set up. Luckily, once everything is configured on mongo, connecting it to the app was the same as what I had been used to.

The backend routes I wrote were login, register, logout, and I contributed to nearly every other backend in one way or another. The big challenge was creating the JWT token for knowing if a user was logged in and building a custom function to salt eh passwords that are passed and stored in the database (as we don't want any plaintext passwords floating around). Using the jsonwebtoken module and building a middleware function to authenticate a user, protect routes that are only for logged in users, building these protections into the user's model function, I was able to secure all the routes by adding the token to the user's schema whenever they log in and checking it every time that a user attempted to access a protected route. Additionally, I stored this token in local storage when a user logged in as running a backend call every time to get the token would be incredibly expensive. For password salting, I used bycrypt and built a middleware function to salt the passwords. Logging out was a simple matter of purging the user's tokens from the database.

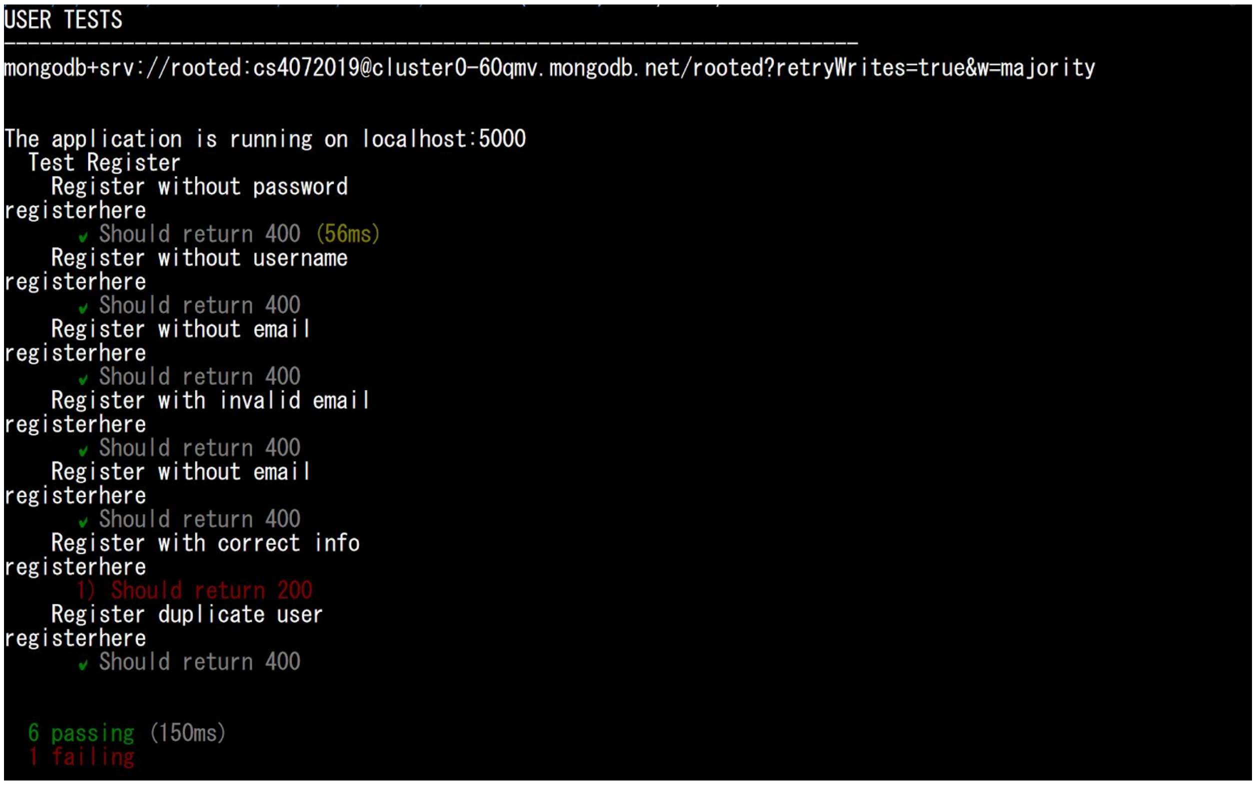
When changing a user's password, the same salting method was used as login and register. I assisted in implementing that functionality to the reset-password and change-password routes.

I was also responsible for the mailer function. This was a simple middleware implementation that could be reused whenever an email needs to be sent to the user. This was accomplished using nodemailer.

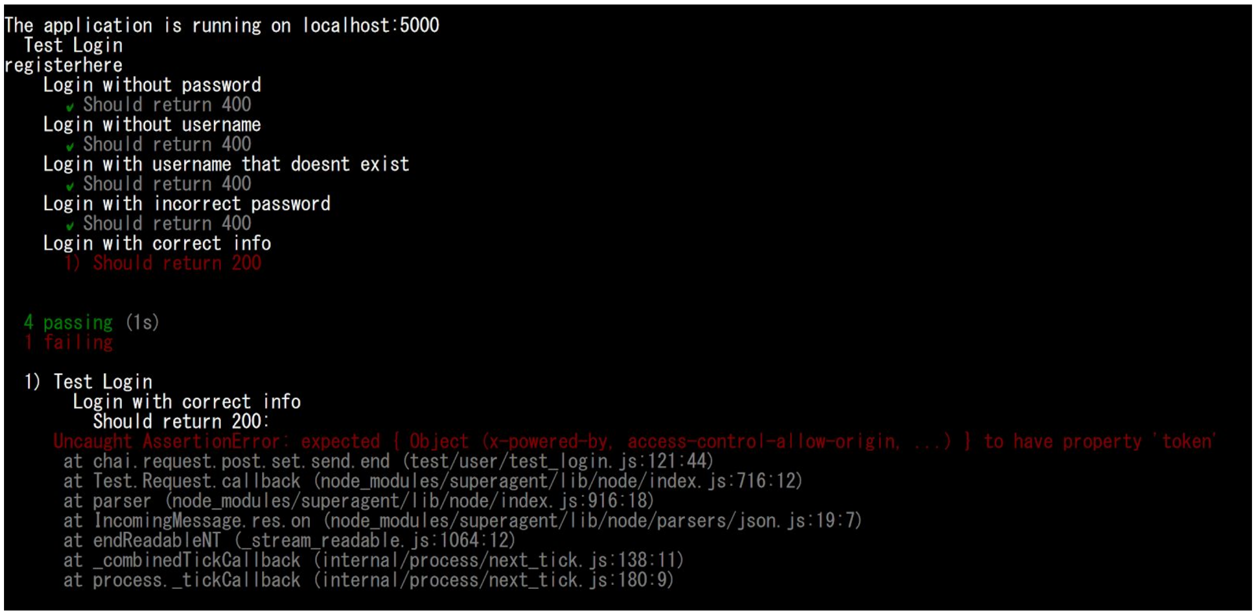
In order to protect any sensitive information from being accessed on our GitHub page (hardcoding is always a bad idea for this stuff), we ensured that the JWT secret, mailer password, and mogo keys were all stored in env files, with were assessed using process.env.

One big flaw I noticed is with reset-password (forgot). The backend currently just picks a random 6 digit integer and makes that the password, and sends that password to the user. A much better implementation would be to make a one-time reset password link, but we were unable to implement that in time, so definitely a needed improvement for the next sprint.

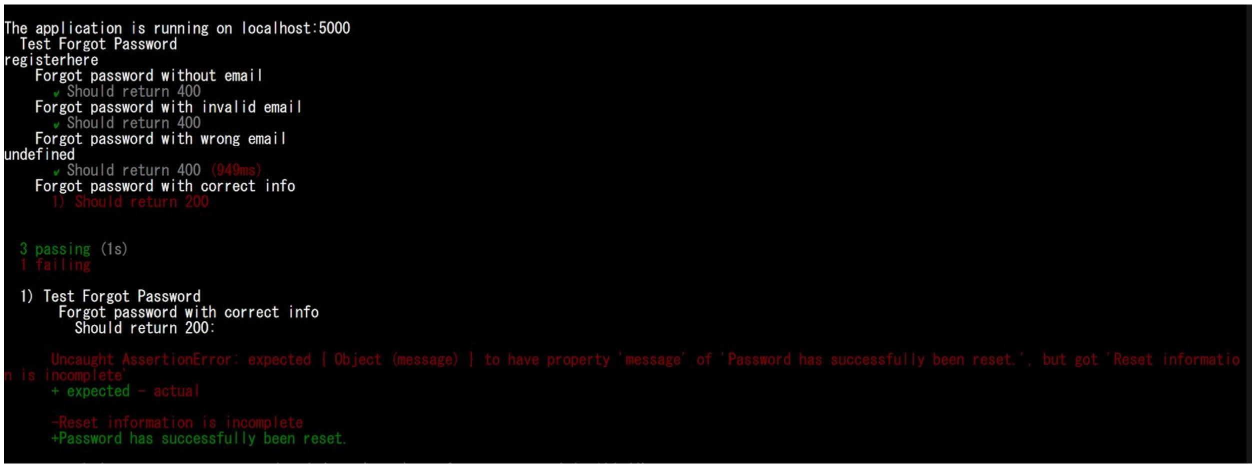
For backend testing, we used mocha and chai. I unfortunately am not super familiar with this testing framework and wasn't able to get every test passing, since I am having some issues with headers, but most of the user routes are fully implemented and can be run by `./tests/runtests.sh`.



```
USER TESTS
--------------------------------------------------------------------------
mongodb+srv://rooted:cs4072019@cluster0-60qmv.mongodb.net/rooted?retryWrites=true&w=majority


The application is running on localhost:5000
  Test Register
    Register without password
registerhere
      ✔ Should return 400 (56ms)
    Register without username
registerhere
      ✔ Should return 400
    Register without email
registerhere
      ✔ Should return 400
    Register with invalid email
registerhere
      ✔ Should return 400
    Register without email
registerhere
      ✔ Should return 400
    Register with correct info
registerhere
      1) Should return 200
    Register duplicate user
registerhere
      ✔ Should return 400


  6 passing (150ms)
  1 failing
```



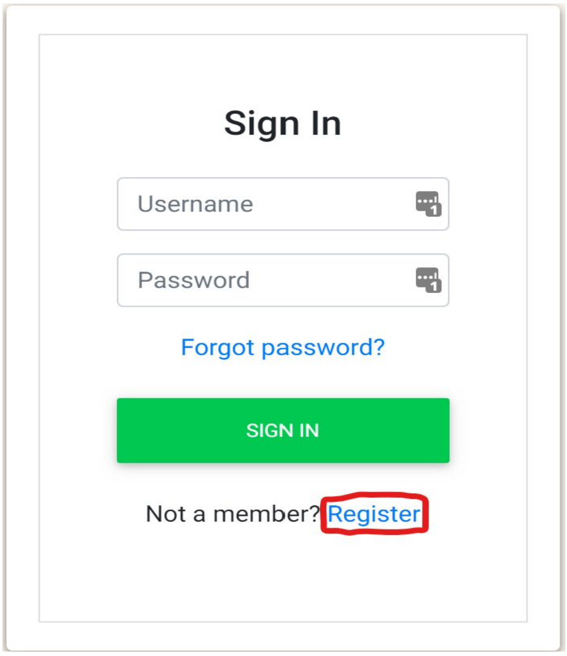
```
The application is running on localhost:5000
  Test Login
registerhere
    Login without password
      ✔ Should return 400
    Login without username
      ✔ Should return 400
    Login with username that doesnt exist
      ✔ Should return 400
    Login with incorrect password
      ✔ Should return 400
    Login with correct info
      1) Should return 200


  4 passing (1s)
  1 failing

  1) Test Login
       Login with correct info
         Should return 200:
     Uncaught AssertionError: expected { Object (x-powered-by, access-control-allow-origin, ...) } to have property 'token'
      at chai.request.post.set.send.end (test/user/test_login.js:121:44)
      at Test.Request.callback (node_modules/superagent/lib/node/index.js:716:12)
      at parser (node_modules/superagent/lib/node/index.js:916:18)
      at IncomingMessage.res.on (node_modules/superagent/lib/node/parsers/json.js:19:7)
      at endReadableNT (_stream_readable.js:1064:12)
      at _combinedTickCallback (internal/process/next_tick.js:138:11)
      at process._tickCallback (internal/process/next_tick.js:180:9)
```



```
The application is running on localhost:5000
  Test Forgot Password
registerhere
    Forgot password without email
      ✔ Should return 400
    Forgot password with invalid email
      ✔ Should return 400
    Forgot password with wrong email
undefined
      ✔ Should return 400 (949ms)
    Forgot password with correct info
      1) Should return 200


  3 passing (1s)
  1 failing

  1) Test Forgot Password
       Forgot password with correct info
         Should return 200:

      Uncaught AssertionError: expected { Object (message) } to have property 'message' of 'Password has successfully been reset.', but got 'Reset informatio
n is incomplete'
      + expected - actual

      -Reset information is incomplete
      +Password has successfully been reset.
```

User Stories Execution

As a user, I would like to be able to register and verify a Rooted account.



Sign In

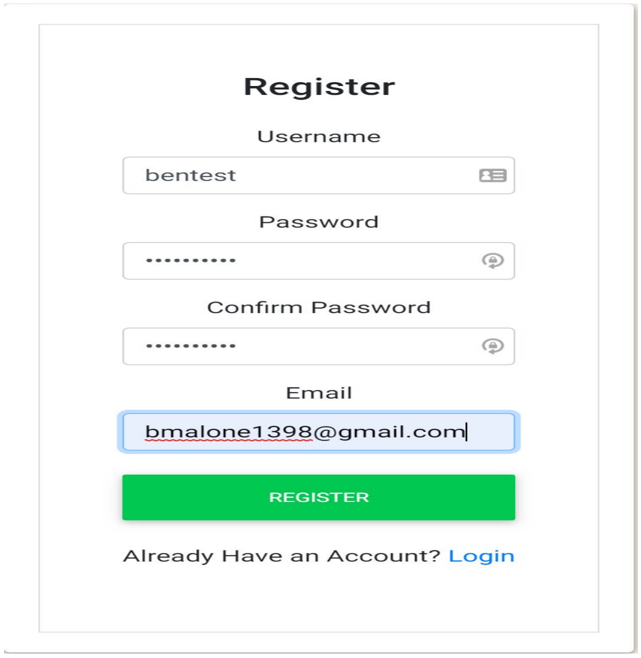
Username

Password

Forgot password?

SIGN IN

Not a member? Register



Register

Username

bentest

Password

•••••••••••

Confirm Password

•••••••••••

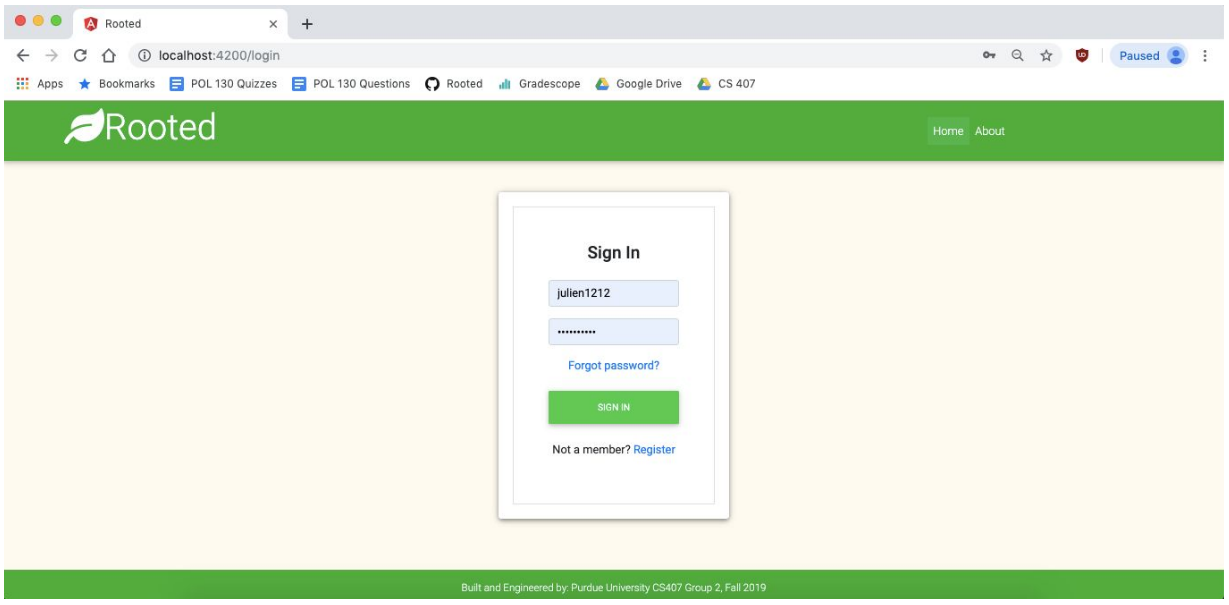
Email

bmalone1398@gmail.com

REGISTER

Already Have an Account? Login

As an existing user, I would like to be able to login



Sign In

julien1212

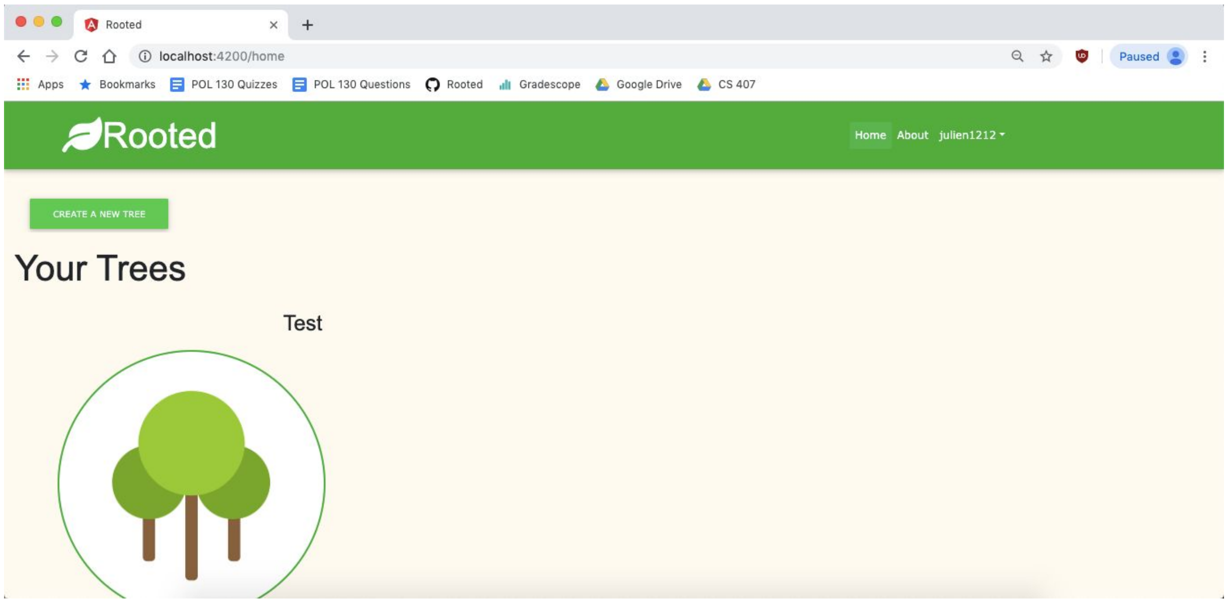
••••••••••

Forgot password?

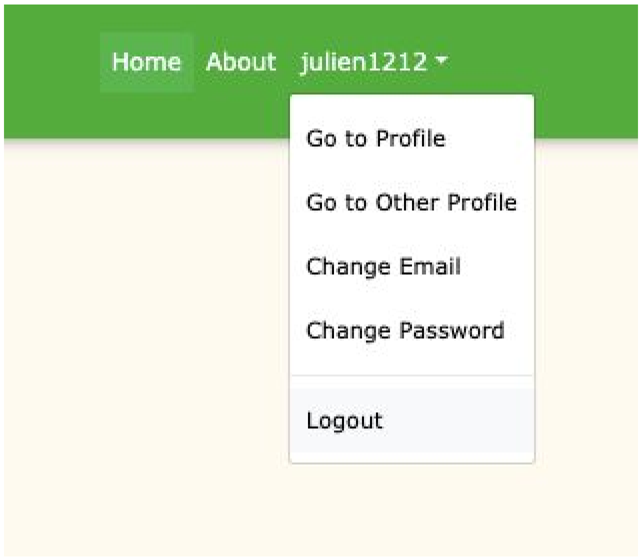
SIGN IN

Not a member? Register

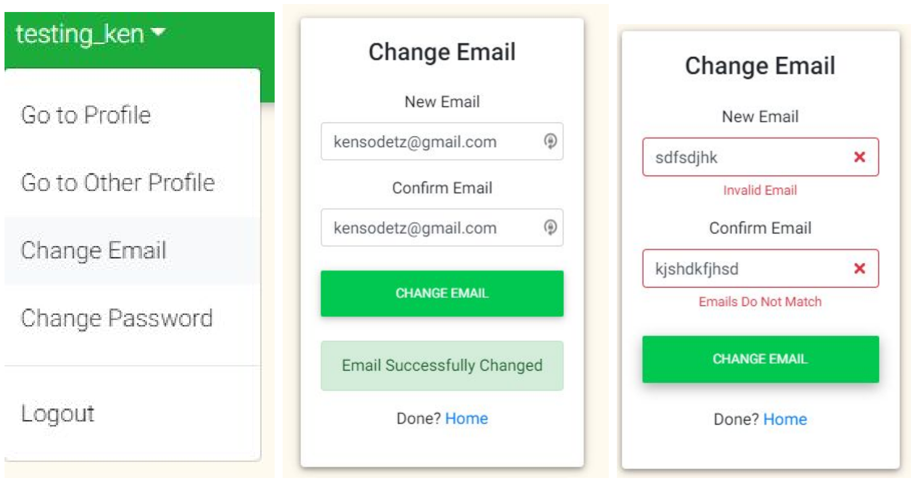
Built and Engineered by: Purdue University CS407 Group 2, Fall 2019



As an existing user, I would like to be able to logout.



As a user, I would like to be able to edit my email address.



Change Email

New Email

kensodetz@gmail.com

Confirm Email

kensodetz@gmail.com

Email Successfully Changed

Change Email

New Email

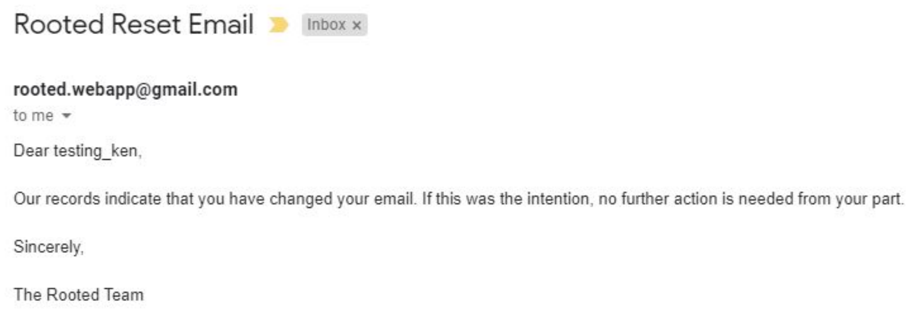
sdfsdjhk

Invalid Email

Confirm Email

kjshdkfjhsd

Emails Do Not Match



Rooted Reset Email Inbox ×

rooted.webapp@gmail.com
to me

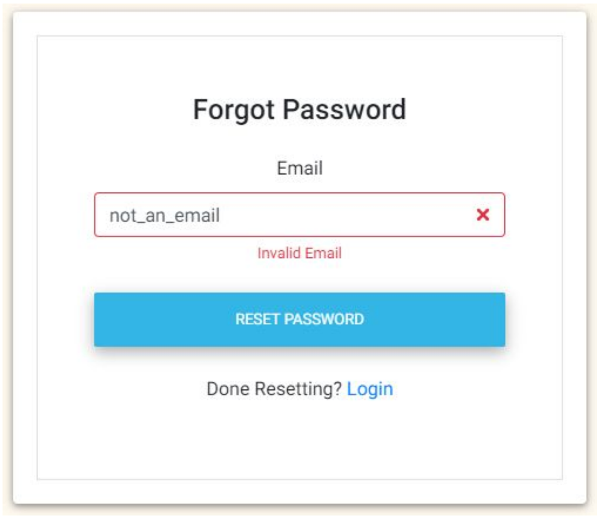
Dear testing_ken,

Our records indicate that you have changed your email. If this was the intention, no further action is needed from your part.

Sincerely,

The Rooted Team

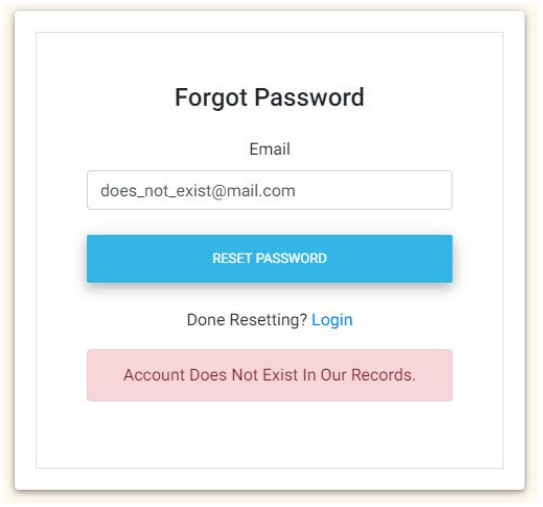
As a user, I would like my password to be reset if I forget it.


Forgot Password

Email

not_an_email

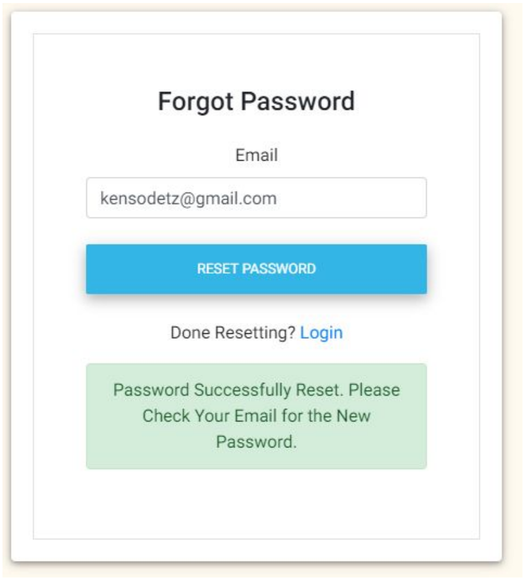
Invalid Email


Forgot Password

Email

does_not_exist@mail.com

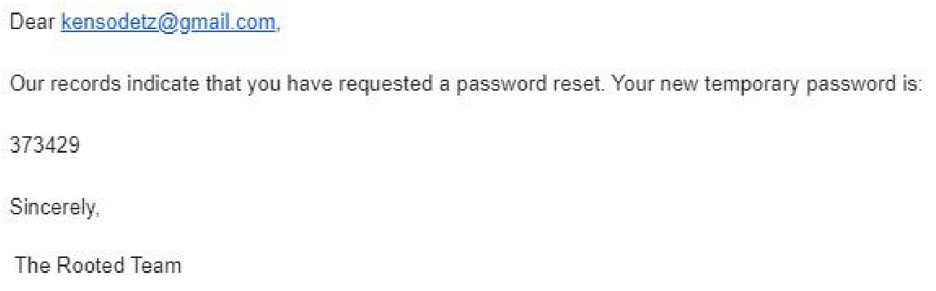
Account Does Not Exist In Our Records.


Forgot Password

Email

kensodetz@gmail.com

Password Successfully Reset. Please Check Your Email for the New Password.


Dear kensodetz@gmail.com,

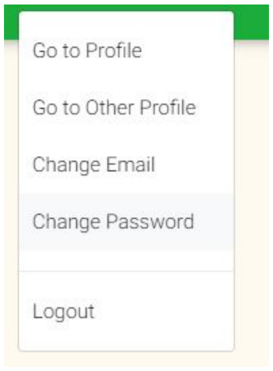
Our records indicate that you have requested a password reset. Your new temporary password is:

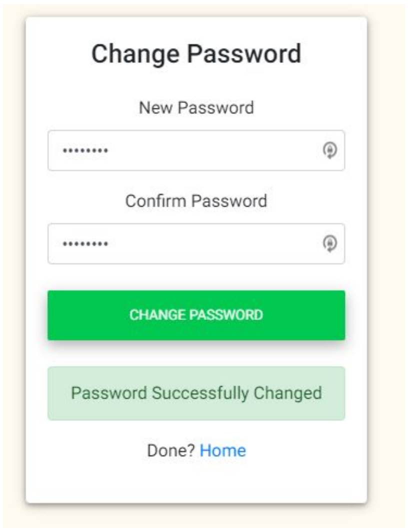
373429

Sincerely,

The Rooted Team

As a user, I would like to be able to change my password.



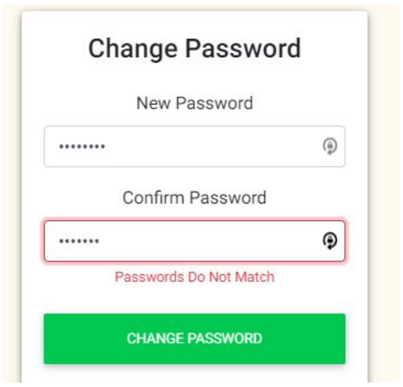

Change Password

New Password

Confirm Password

Password Successfully Changed

Done? Home

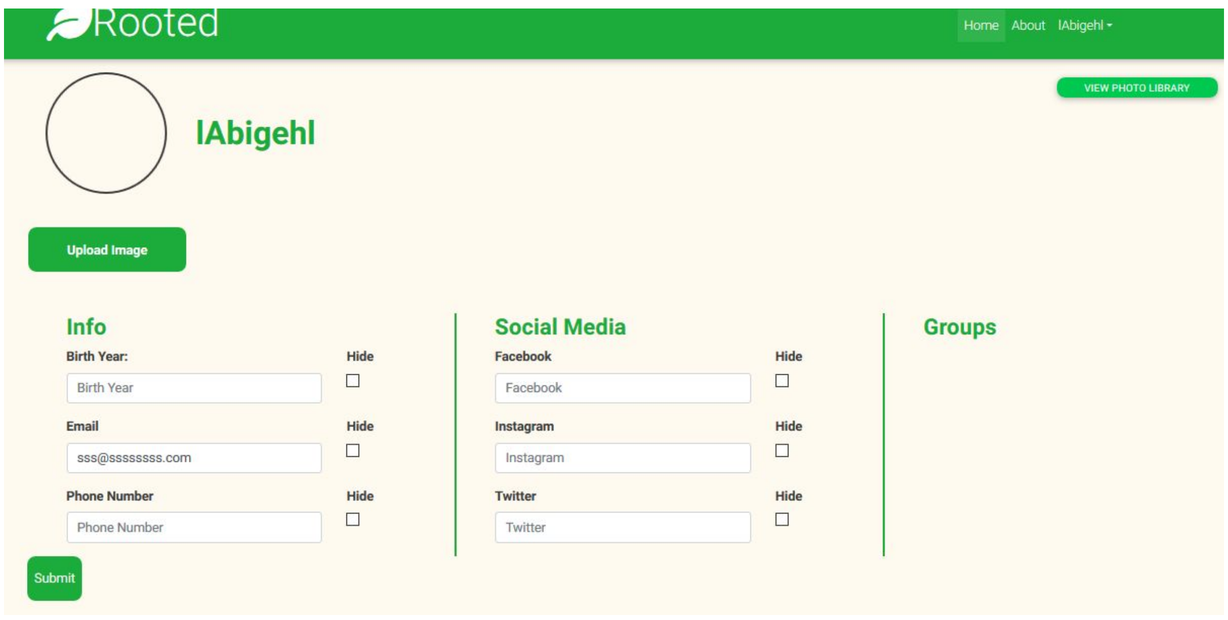

Change Password

New Password

Confirm Password

Passwords Do Not Match

As a user, I would like to be able to create and edit a general profile page, with info such as my name, birth year, bio, links to social media, etc.



IAbigehl

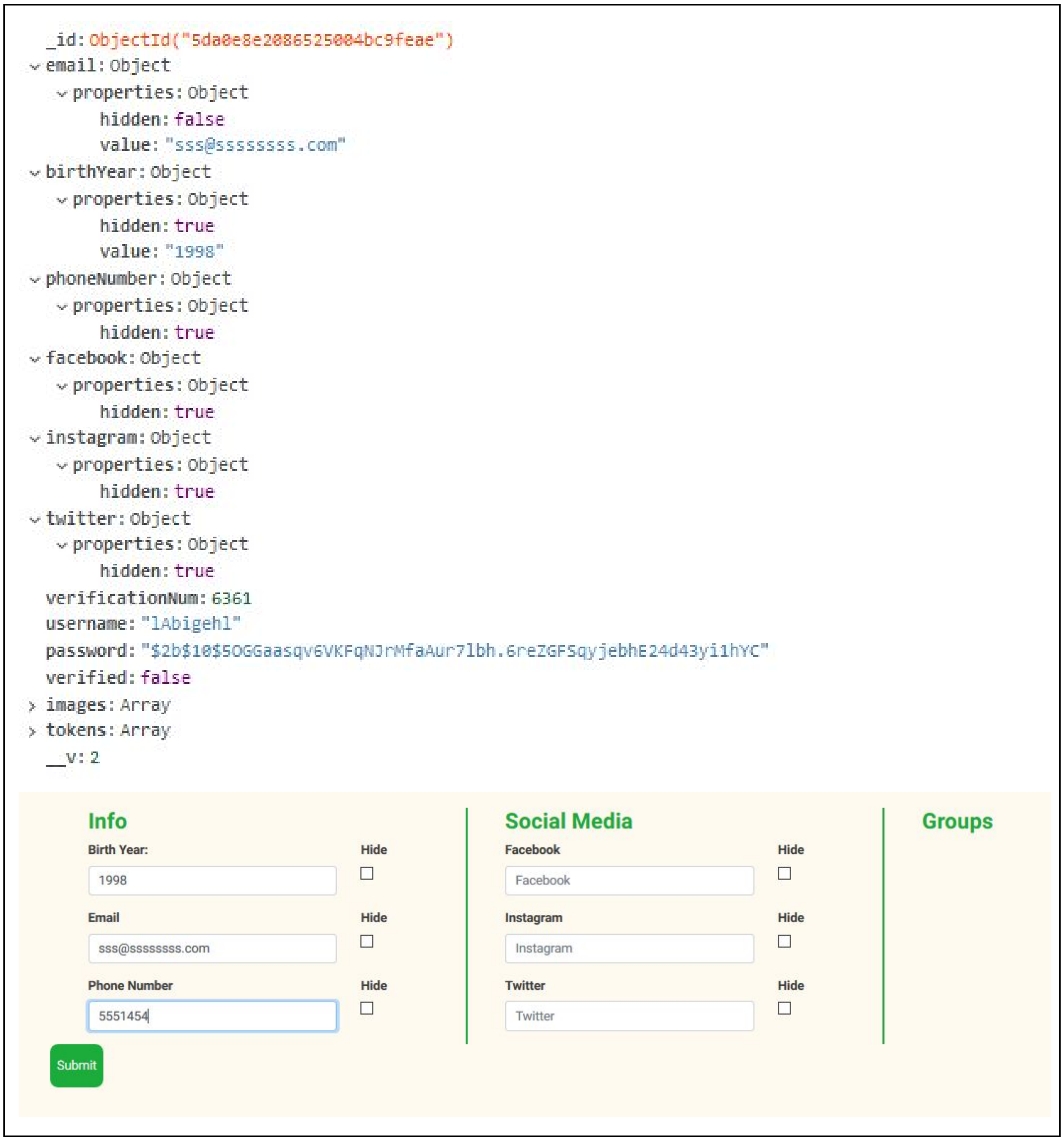
Info

Birth Year: Birth Year	Hide ☐
Email: sss@ssssssss.com	Hide ☐
Phone Number: Phone Number	Hide ☐

Social Media

Facebook: Facebook	Hide ☐
Instagram: Instagram	Hide ☐
Twitter: Twitter	Hide ☐

Groups



```
  _id: ObjectId("5da0e8e2086525004bc9feae")
v email: Object
    v properties: Object
          hidden: false
          value: "sss@sssssss.com"
v birthYear: Object
    v properties: Object
          hidden: true
          value: "1998"
v phoneNumber: Object
    v properties: Object
          hidden: true
v facebook: Object
    v properties: Object
          hidden: true
v instagram: Object
    v properties: Object
          hidden: true
v twitter: Object
    v properties: Object
          hidden: true
  verificationNum: 6361
  username: "lAbigehl"
  password: "$2b$10$5OGGaasqv6VKFqNJrMfaAur7lbh.6reZGFSqyjebhE24d43yi1hYC"
  verified: false
> images: Array
> tokens: Array
  __v: 2
```

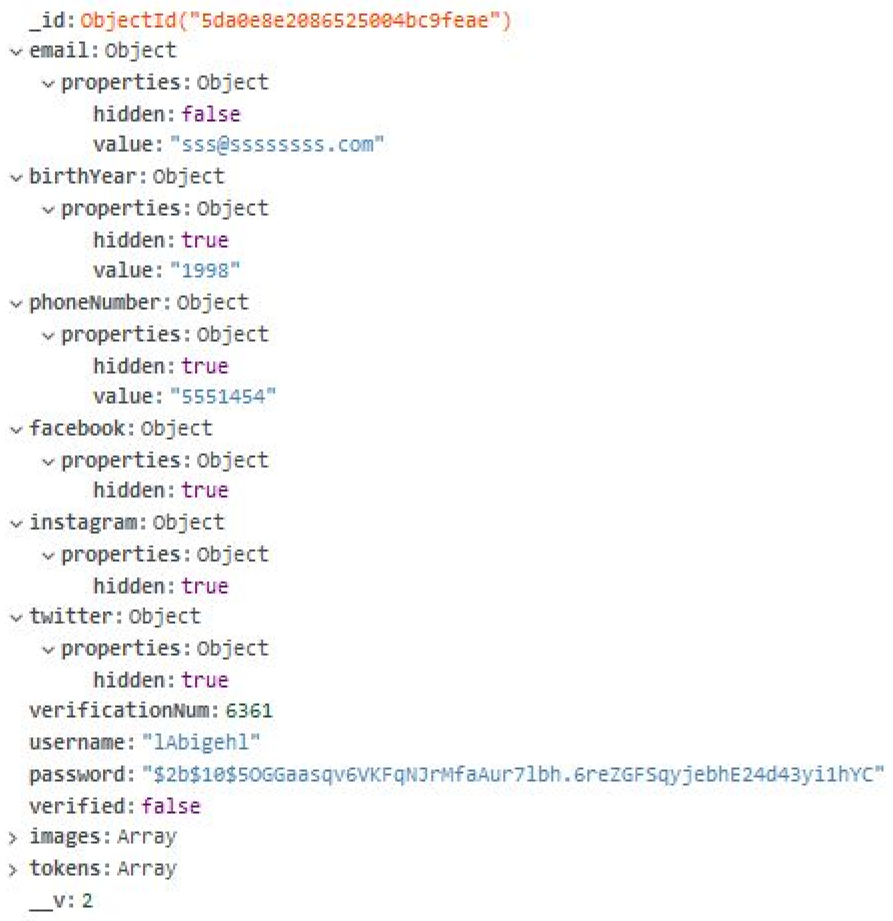
Info

Birth Year:	Hide
1998	☐
Email	Hide
sss@sssssss.com	☐
Phone Number	Hide
5551454	☐

Social Media

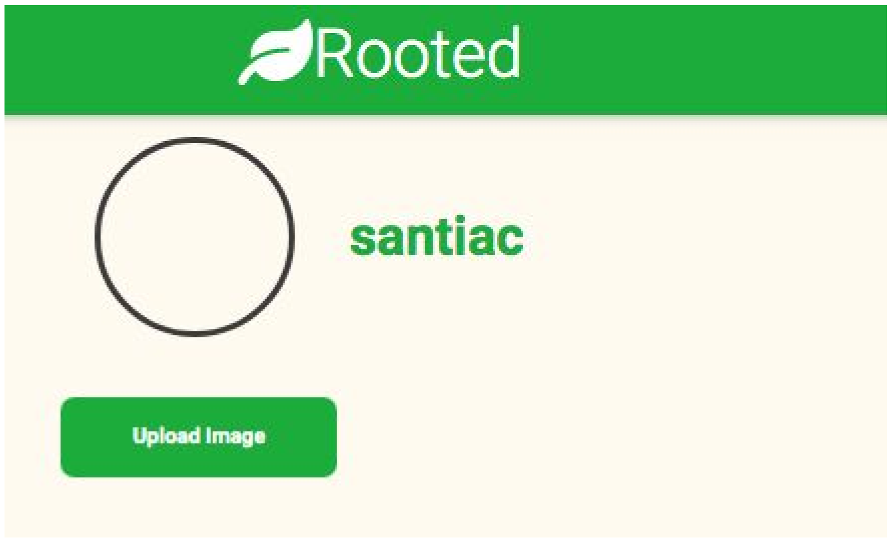
Facebook	Hide
Facebook	☐
Instagram	Hide
Instagram	☐
Twitter	Hide
Twitter	☐

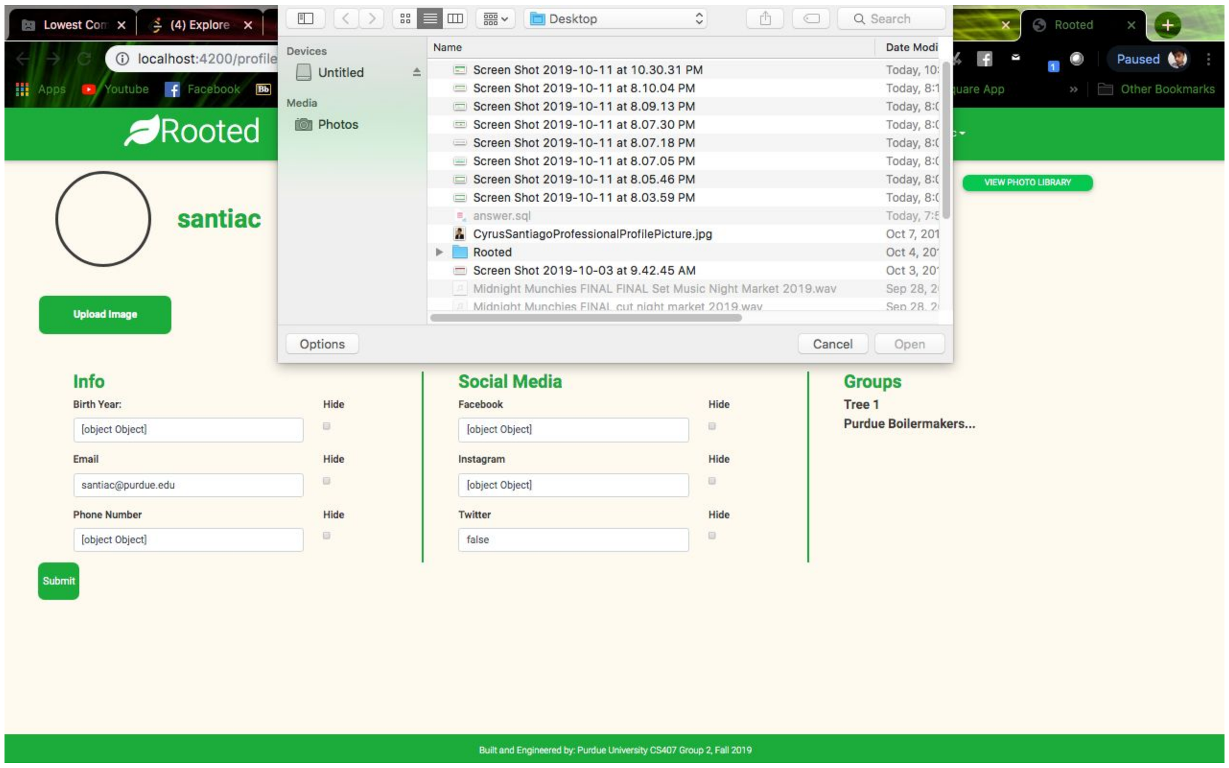
Groups



```
  _id: ObjectId("5da0e8e2086525004bc9feae")
v email: Object
  v properties: Object
      hidden: false
      value: "sss@sssssss.com"
v birthYear: Object
  v properties: Object
      hidden: true
      value: "1998"
v phoneNumber: Object
  v properties: Object
      hidden: true
      value: "5551454"
v facebook: Object
  v properties: Object
      hidden: true
v instagram: Object
  v properties: Object
      hidden: true
v twitter: Object
  v properties: Object
      hidden: true
  verificationNum: 6361
  username: "lAbigehl"
  password: "$2b$10$5OGGaasqv6VKFqNJrMfaAur7lbh.6reZGFSqyjebhE24d43yi1hYC"
  verified: false
> images: Array
> tokens: Array
  __v: 2
```

As a user, I would like to be able to upload a profile image





Name	Date Modi
Screen Shot 2019-10-11 at 10.30.31 PM	Today, 10:
Screen Shot 2019-10-11 at 8.10.04 PM	Today, 8:1
Screen Shot 2019-10-11 at 8.09.13 PM	Today, 8:(
Screen Shot 2019-10-11 at 8.07.30 PM	Today, 8:(
Screen Shot 2019-10-11 at 8.07.18 PM	Today, 8:(
Screen Shot 2019-10-11 at 8.07.05 PM	Today, 8:(
Screen Shot 2019-10-11 at 8.05.46 PM	Today, 8:(
Screen Shot 2019-10-11 at 8.03.59 PM	Today, 8:(
answer.sql	Today, 7:5
CyrusSantiagoProfessionalProfilePicture.jpg	Oct 7, 201
Rooted	Oct 4, 20
Screen Shot 2019-10-03 at 9.42.45 AM	Oct 3, 20
Midnight Munchies FINAL FINAL Set Music Night Market 2019.wav	Sep 28, 2
Midnight Munchies FINAL cut night market 2019.wav	Sep 28, 2

Info

Birth Year: [object Object] Hide

Email: santiac@purdue.edu Hide

Phone Number: [object Object] Hide

Social Media

Facebook: [object Object] Hide

Instagram: [object Object] Hide

Twitter: false Hide

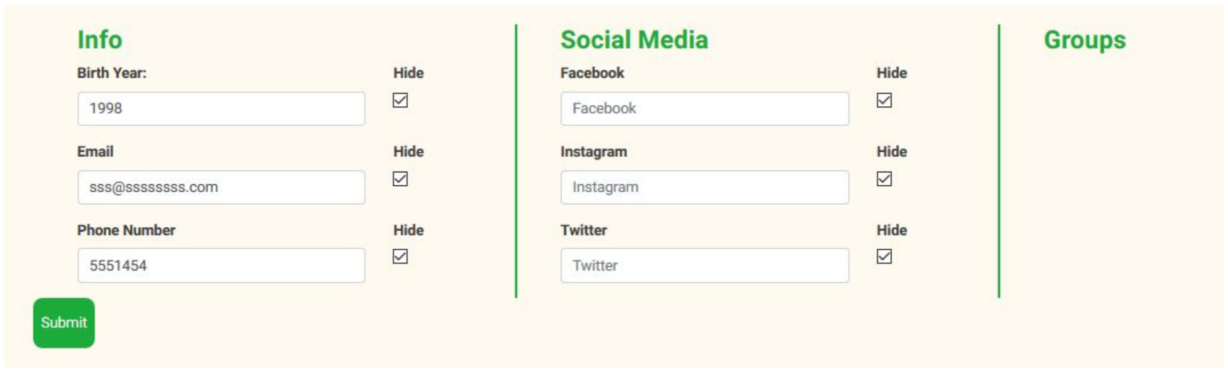
Groups

Tree 1

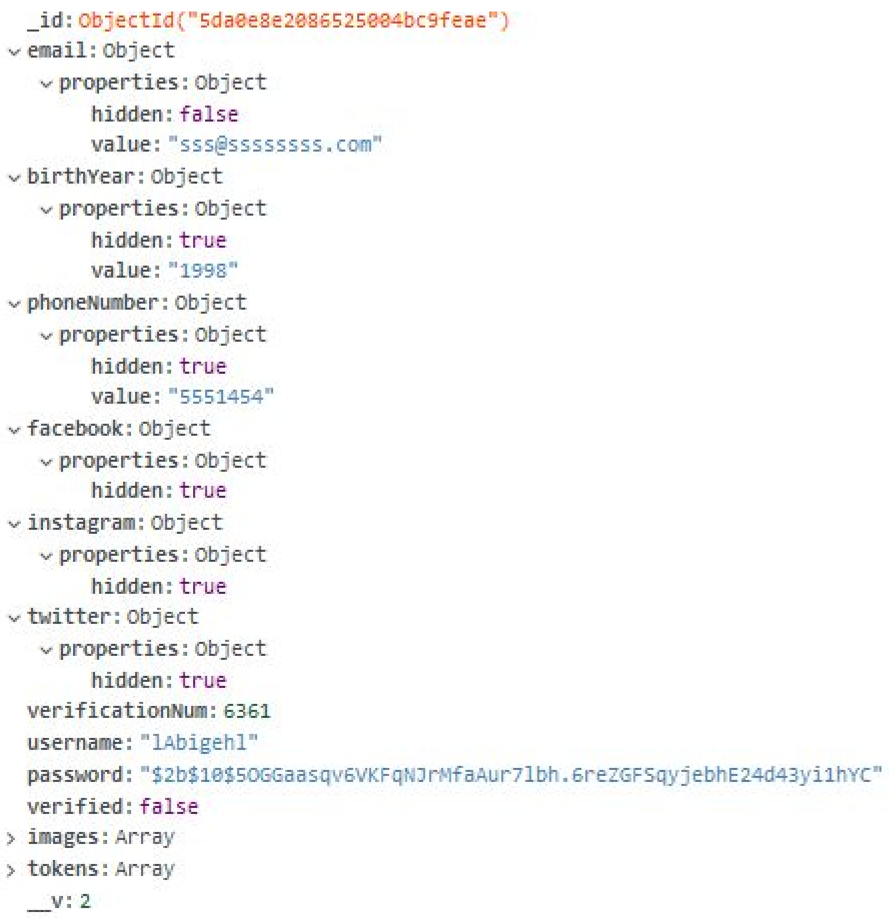
Purdue Boilermakers...

Built and Engineered by: Purdue University CS407 Group 2, Fall 2019

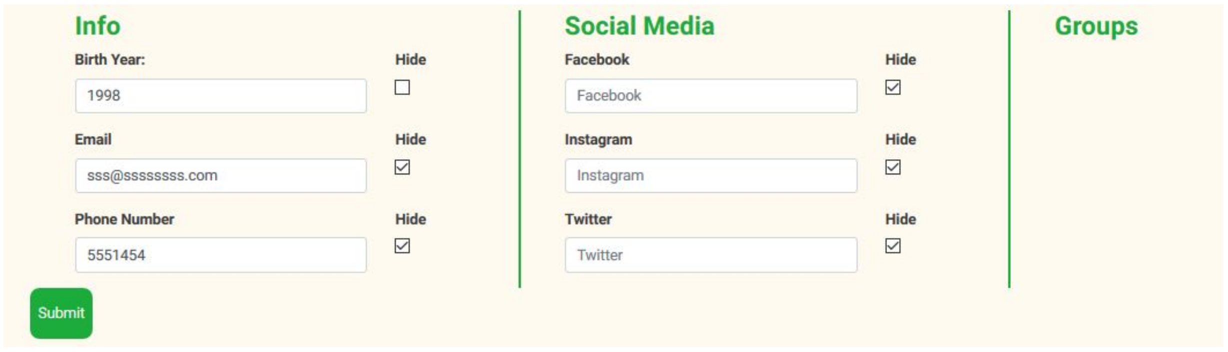
As a user, I would like to be able to selectively hide or display specific info on my profile page.



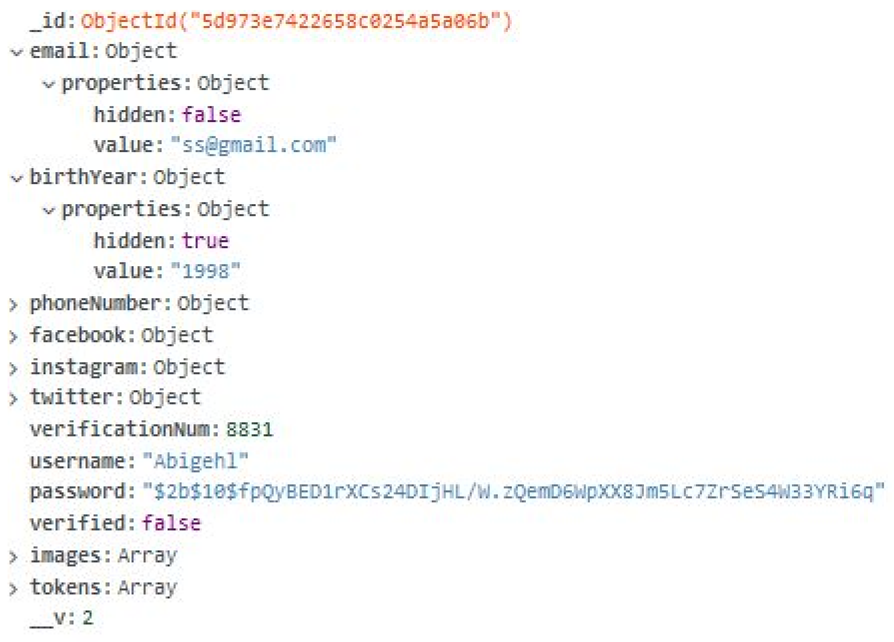
Info		Social Media		Groups
Birth Year: 1998	Hide ☑	Facebook: Facebook	Hide ☑	
Email: sss@sssssss.com	Hide ☑	Instagram: Instagram	Hide ☑	
Phone Number: 5551454	Hide ☑	Twitter: Twitter	Hide ☑	



```
  _id: ObjectId("5da0e8e2086525004bc9feae")
v email: Object
  v properties: Object
      hidden: false
      value: "sss@sssssss.com"
v birthYear: Object
  v properties: Object
      hidden: true
      value: "1998"
v phoneNumber: Object
  v properties: Object
      hidden: true
      value: "5551454"
v facebook: Object
  v properties: Object
      hidden: true
v instagram: Object
  v properties: Object
      hidden: true
v twitter: Object
  v properties: Object
      hidden: true
  verificationNum: 6361
  username: "1Abigeh1"
  password: "$2b$10$5OGGaasqv6VKFqNJrMfaAur71bh.6reZGFSqyjebhE24d43yi1hYC"
  verified: false
> images: Array
> tokens: Array
  __v: 2
```



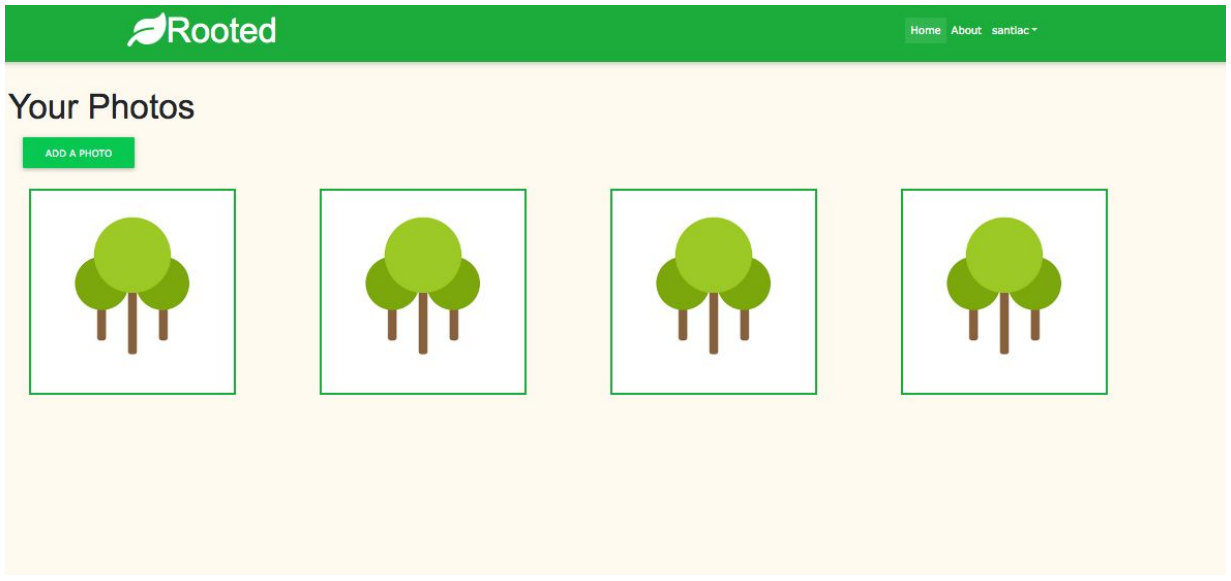
Info		Social Media		Groups
Birth Year: 1998	Hide ☐	Facebook: Facebook	Hide ☑	
Email: sss@sssssss.com	Hide ☑	Instagram: Instagram	Hide ☑	
Phone Number: 5551454	Hide ☑	Twitter: Twitter	Hide ☑	

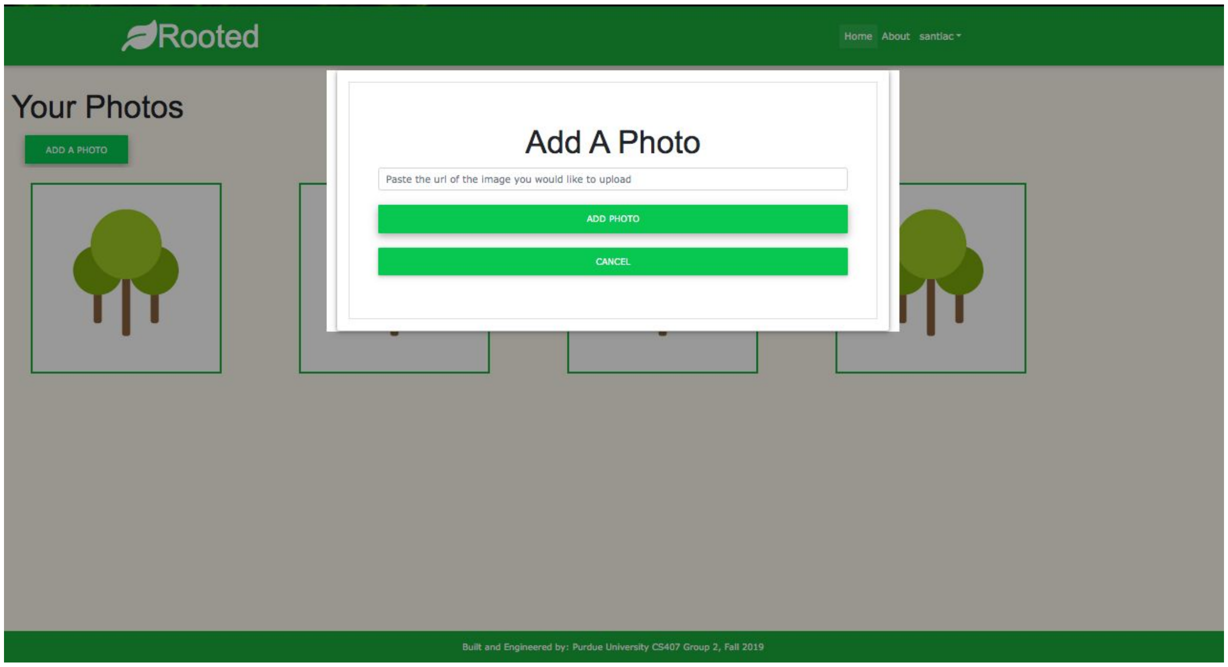


```
  _id: ObjectId("5d973e7422658c0254a5a06b")
⌄ email: Object
   ⌄ properties: Object
       hidden: false
       value: "ss@gmail.com"
⌄ birthYear: Object
   ⌄ properties: Object
       hidden: true
       value: "1998"
› phoneNumber: Object
› facebook: Object
› instagram: Object
› twitter: Object
  verificationNum: 8831
  username: "Abigehl"
  password: "$2b$10$fpQyBED1rXCs24DIjHL/W.zQemD6WpXX8Jm5Lc7ZrSeS4W33YRi6q"
  verified: false
› images: Array
› tokens: Array
  __v: 2
```

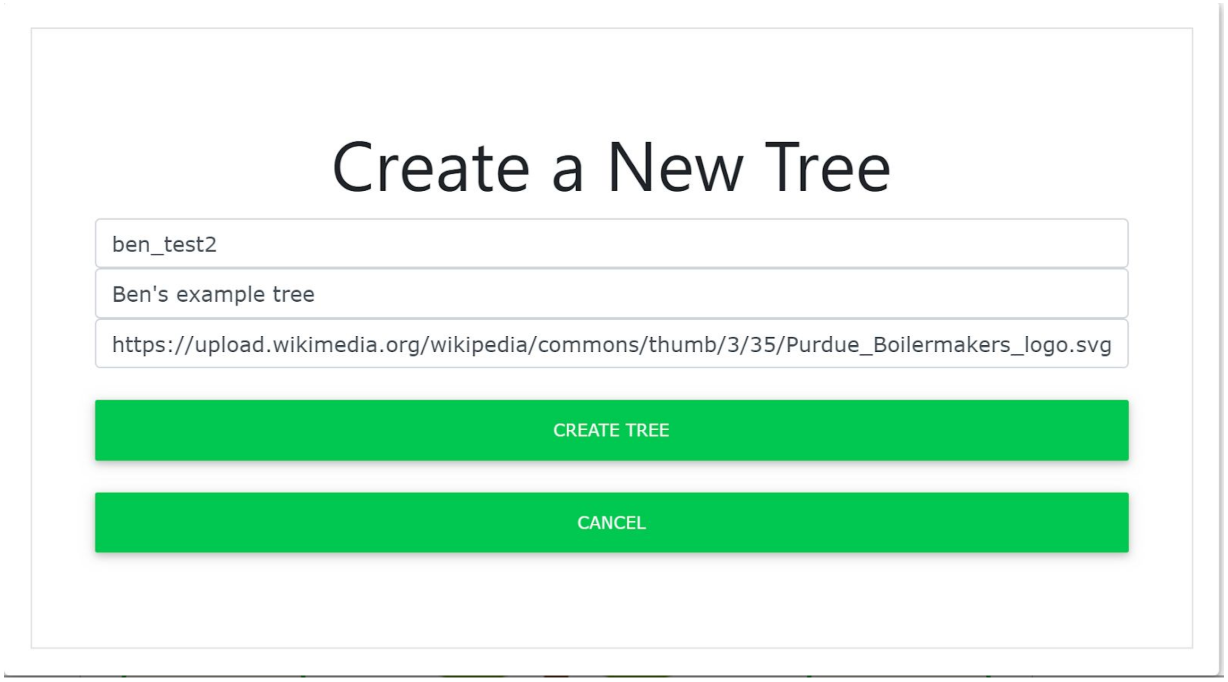
Does not yet work due to issues in communication between profile page and backend.

As a user, I would like to be able to upload photos to my profile page's photo library.

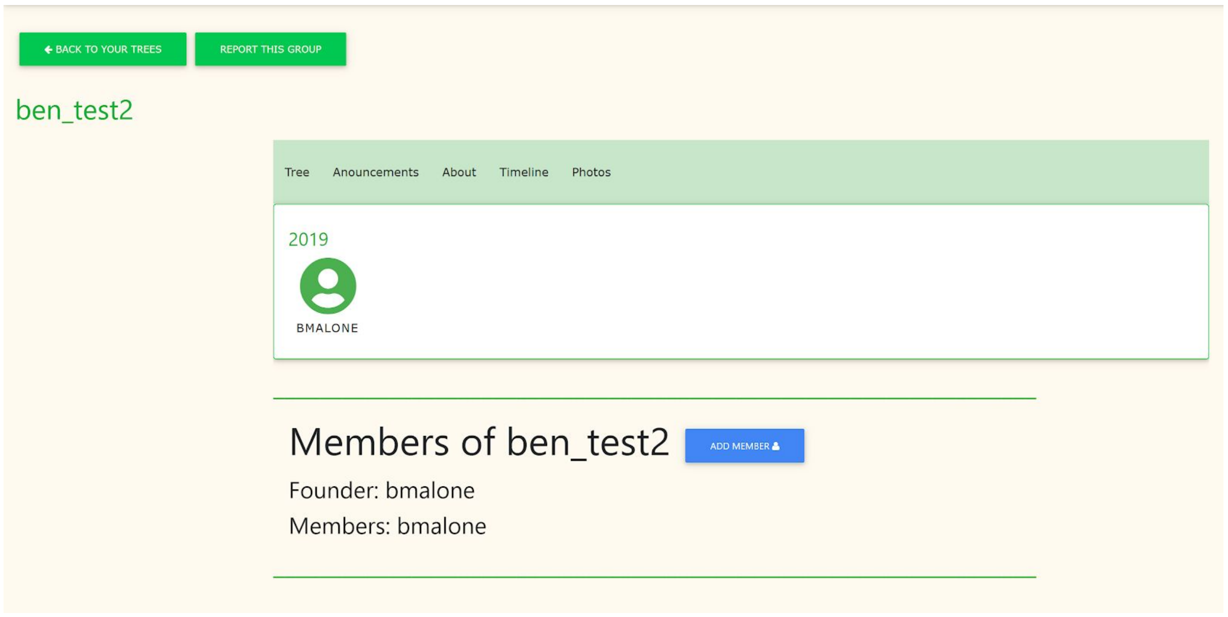



Rooted
Home About santiac
Your Photos
ADD A PHOTO
Add A Photo
Paste the url of the image you would like to upload
ADD PHOTO
CANCEL
Built and Engineered by: Purdue University CS407 Group 2, Fall 2019

As a user, I would like to be able to create a group and be granted group creator privileges



ben_test2
Ben's example tree
https://upload.wikimedia.org/wikipedia/commons/thumb/3/35/Purdue_Boilermakers_logo.svg

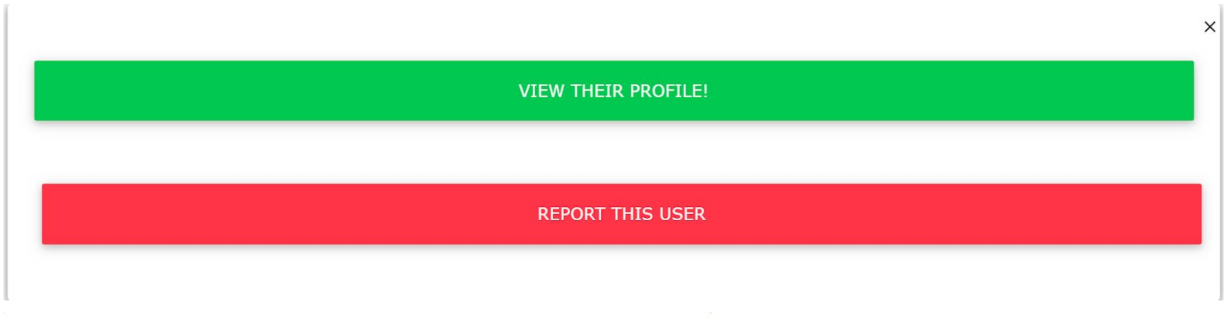


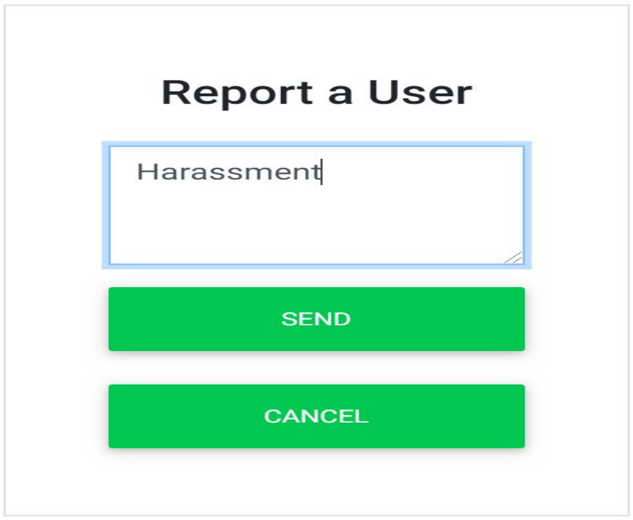
Members of ben_test2

Founder: bmalone

Members: bmalone

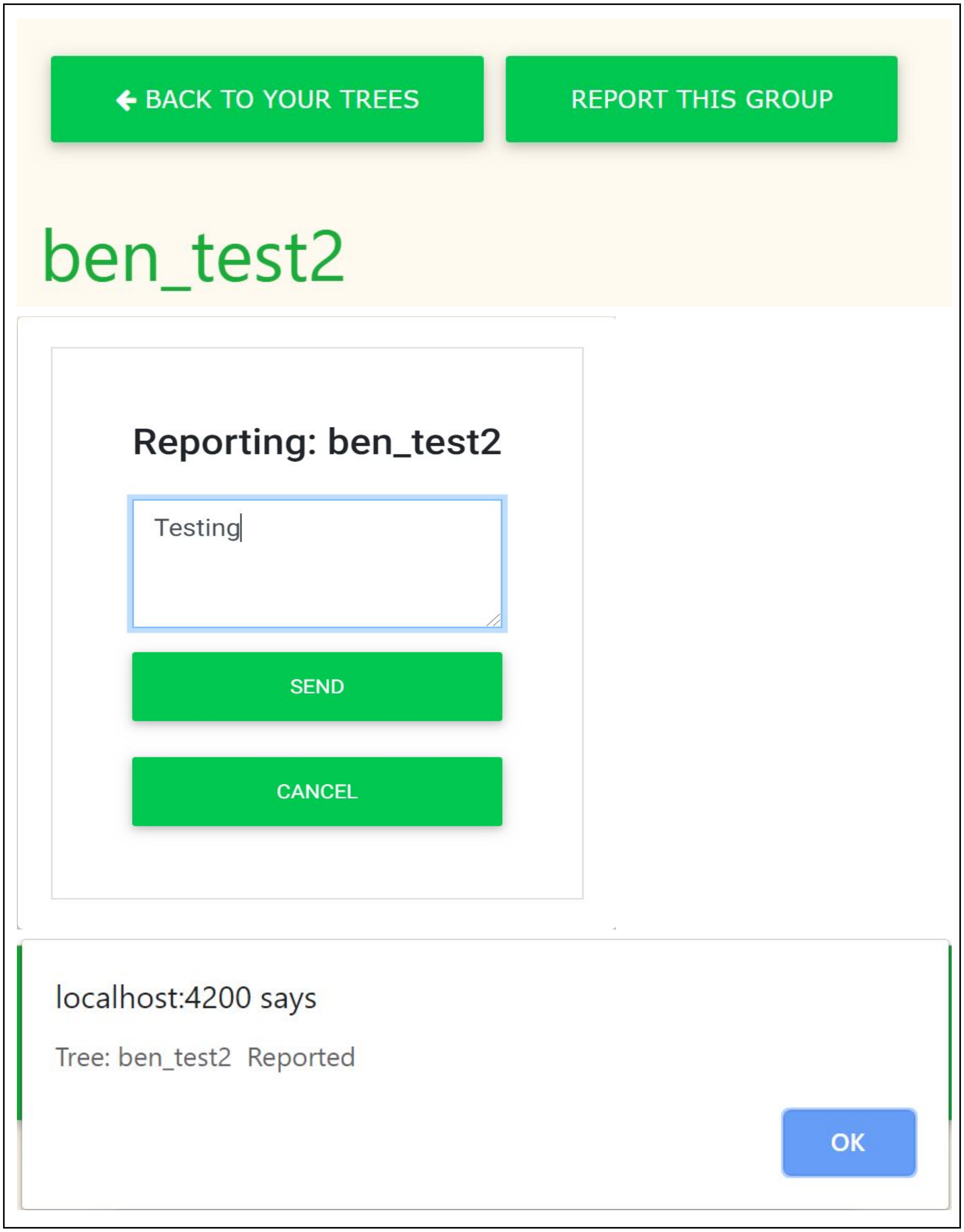
As a general user, I would like to be able to report users of a group to the group's moderation team





Harassment

As a general user, I would like to be able to report a group to the sitewide admins



ben_test2

Reporting: ben_test2

Testing

localhost:4200 says

Tree: ben_test2 Reported