

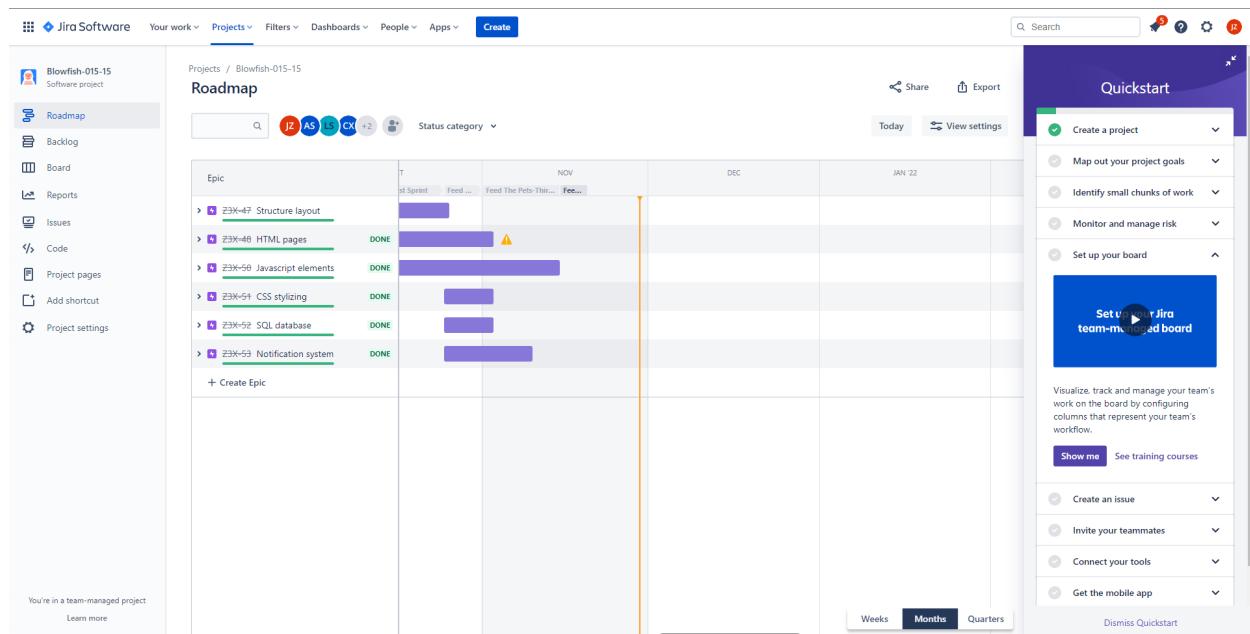
Title: Feed the pets/My Fitness Pet

Who: Catherine, Leo, Josh, Adam, Ayden

Project Description: Our project, Feed the Pets, or, My Fitness Pet (we didn't really decide on a name), is a web application designed to make it simple and easy to keep track of your pet's health. When you get to our website you are prompted to create an account, this account will hold information about what pets you have. These accounts can be shared so that you can keep track with your family to know who was the last person to feed the pets. On top of that each pet will have their own profile which will keep track of how much they need to be fed and when, it will also hold other important information about them. On our home page you will find intuitive navigation to each of the pages such as the calendar. We used Google calendar API to create a calendar that you can use to keep track of all your pets feeding times and important dates like vet appointments. We also have a notification system in place so that if you are busy you will still be reminded when it's time to feed the pets.

Project Tracker:

<https://csci-3308-fall21-015-15.atlassian.net/jira/software/projects/Z3X/boards/1/roadmap>

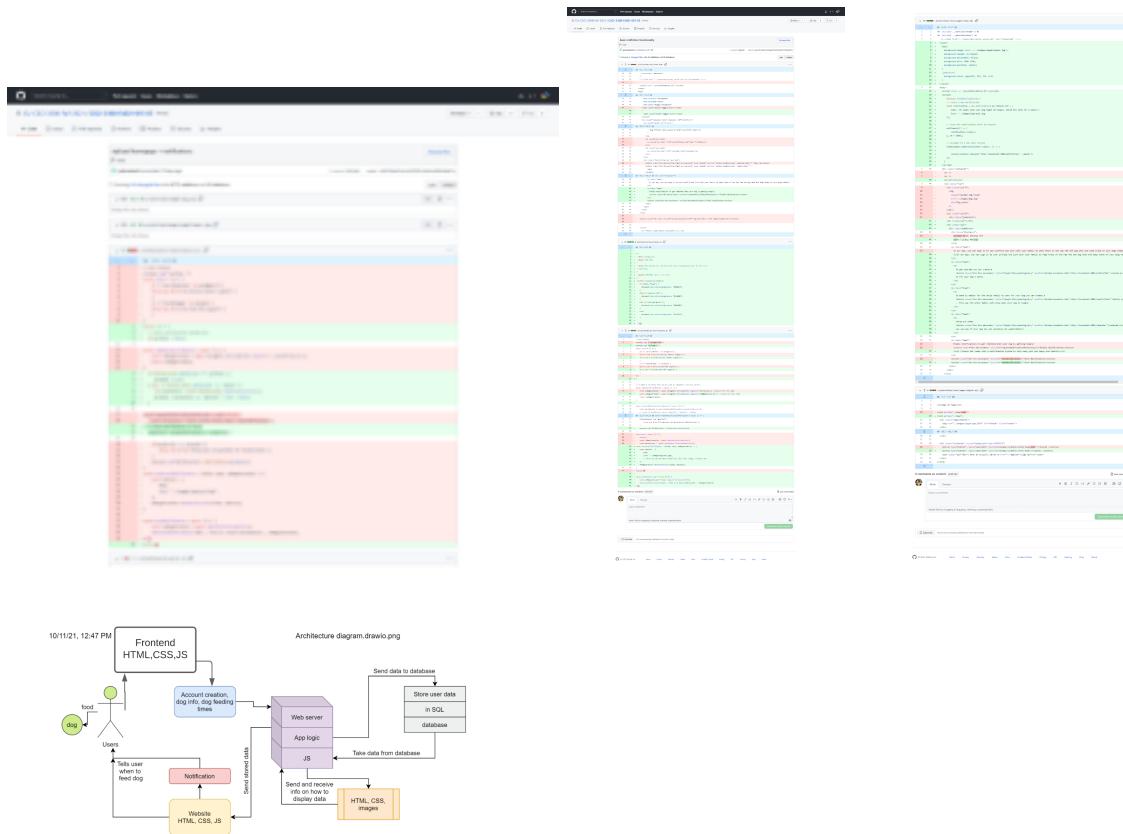


Video: <https://vimeo.com/647818382/5410b04385>

VCS: <https://github.com/CU-CSCI-3308-Fall-2021/CSCI-3308-Fall21-015-05>

Contributions:

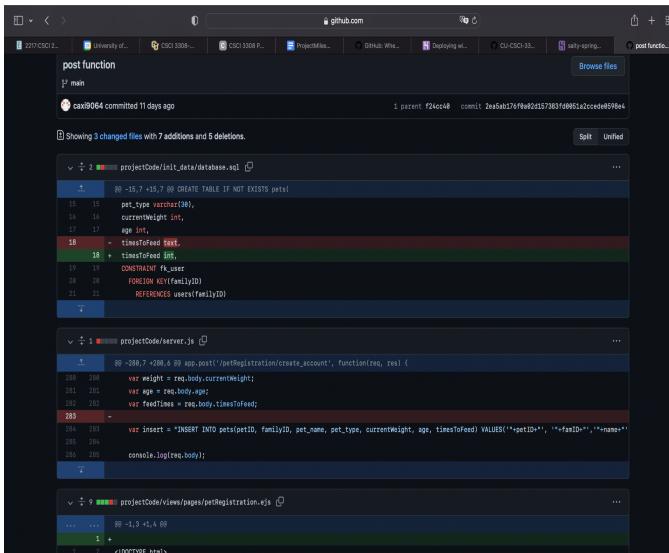
Josh: I created the architecture diagram in order to give some structure to our web app. I used an open source diagram creation tool to make the diagram. I spent a lot of time trying to get push notifications to work with our website but ended up having to settle with local notifications because of some issues I was having with the server. I just used the built in notifications that come with nodeJS. I stylized the frontpage to make it look more professional using bootstrap and CSS.



Catherine: Created database in the sql file and designed it using the db diagram app to help store the user info, pet info, and registration info. We did not need to use all the data tables displayed in the diagram and for the most part the calendar tables were not needed.

Collaborated with Ayden and Leo about the backend. I added a html pet profile page to display

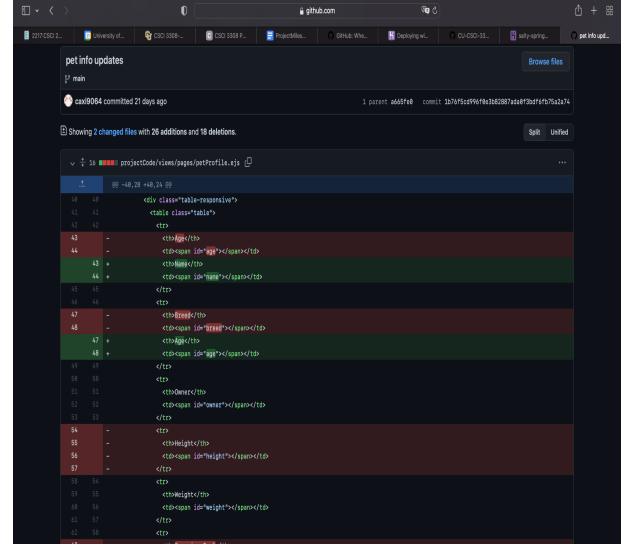
the pets info. I worked on getting the pet registration to connect with the database and display some information initially. There were difficulties initially with getting the proper tabs to match but Ayden was able to help with that.



```

post function
└── main
    caxi064 committed 11 days ago
    1 parent f24cc48 commit 2ea5ab37d9e082d1573837d60851a2ced8598e4
    Showing 3 changed files with 7 additions and 5 deletions.
    Split Unified
    projectCode/init_data/database.sql
    ...
    15 15 pet_type varchar(30),
    16 16 currentWeight int,
    17 17 age int,
    18 - timeToFeed tinyint,
    18 + timeToFeed int,
    19 19 CONSTRAINT fk_user
    20 20 FOREIGN KEY(familyID)
    21 21 REFERENCES users(familyID)
    ...
    projectCode/server.js
    ...
    238 238 var weight = req.body.currentWeight;
    239 239 var age = req.body.age;
    240 240 var feedTimes = req.body.timesToFeed;
    241 241
    242 242 var insert = `INSERT INTO pet(petID, familyID, pet_name, pet_type, currentWeight, age, timesToFeed) VALUES('${petID}', '${familyID}', '${name}', '${type}', ${currentWeight}, ${age}, ${timesToFeed})`;
    243 243
    244 244 console.log(req.body);
    ...
    projectCode/view/pages/petRegistration.ejs
    ...
    283 283 <td>${pet.pet_name}</td>
    284 284 <td>${pet.pet_type}</td>
    285 285 <td>${pet.currentWeight}</td>
    286 286 <td>${pet.age}</td>
    287 287 <td>${pet.timeToFeed}</td>
    ...

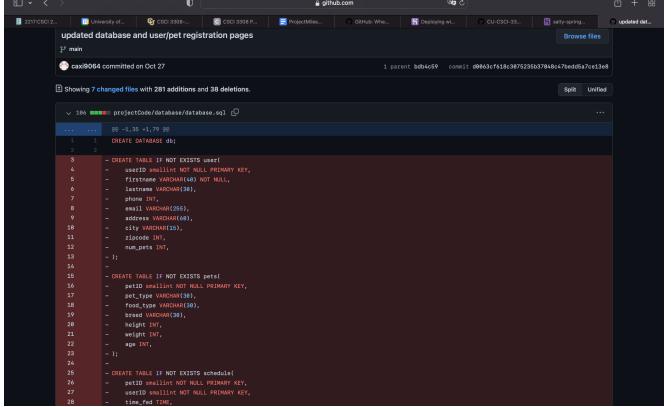
```



```

pet info updates
└── main
    caxi064 committed 21 days ago
    1 parent add65f48 commit 3a7f5fc4949a3d2077aef32cf5fb752a74
    Showing 2 changed files with 26 additions and 18 deletions.
    Split Unified
    projectCode/views/pages/petProfile.ejs
    ...
    40 40 <div class="table-responsive">
    41 41   <table class="table">
    42 42     <tr>
    43 43       <td>${pet.pet_name}</td>
    44 44       <td>${pet.pet_type}</td>
    45 45       <td>${pet.currentWeight}</td>
    46 46       <td>${pet.age}</td>
    47 47       <td>${pet.timeToFeed}</td>
    48 48       <td>${pet.timeToFeed}</td>
    49 49       <td>${pet.timeToFeed}</td>
    50 50     </tr>
    51 51   </table>
    52 52 </div>
    53 53 <div class="row">
    54 54   <div class="col-sm-6">
    55 55     <div class="form-group">
    56 56       <label>Weight</label>
    57 57       <input type="text" id="weight" value="100" class="form-control" placeholder="Weight" style="width: 100%; height: 35px;" />
    58 58     </div>
    59 59   </div>
    60 60   <div class="col-sm-6">
    61 61     <div class="form-group">
    62 62       <label>Age</label>
    63 63       <input type="text" id="age" value="10" class="form-control" placeholder="Age" style="width: 100%; height: 35px;" />
    64 64     </div>
    ...

```



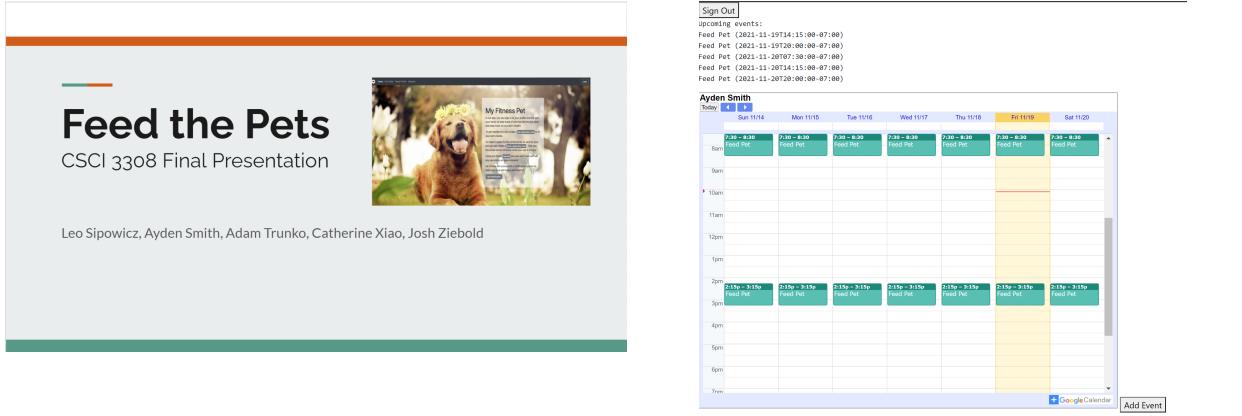
```

updated database and user/pet/registration pages
└── main
    caxi064 committed on Oct 27
    1 parent bd8c59 commit d863c7a58c9e73295b3794b42ed2d7ce1d8
    Showing 7 changed files with 281 additions and 38 deletions.
    Split Unified
    projectCode/database.sql
    ...
    1 1 CREATE DATABASE db;
    2 2
    3 3 -- CREATE TABLE IF NOT EXISTS user(
    4 4   --   user_id serial NOT NULL PRIMARY KEY,
    5 5   --   first_name VARCHAR(50),
    6 6   --   last_name VARCHAR(50),
    7 7   --   phone INT,
    8 8   --   email VARCHAR(50),
    9 9   --   address VARCHAR(50),
    10 10  --   city VARCHAR(50),
    11 11  --   zip_code INT,
    12 12  --   pet_id INT,
    13 13  -- );
    14 14
    15 15 -- CREATE TABLE IF NOT EXISTS pet(
    16 16   --   pet_id serial NOT NULL PRIMARY KEY,
    17 17   --   pet_type VARCHAR(50),
    18 18   --   name VARCHAR(50),
    19 19   --   breed VARCHAR(50),
    20 20   --   height INT,
    21 21   --   weight INT,
    22 22   --   age INT,
    23 23   -- );
    24 24
    25 25 -- CREATE TABLE IF NOT EXISTS schedule(
    26 26   --   pet_id serial NOT NULL PRIMARY KEY,
    27 27   --   user_id serial NOT NULL PRIMARY KEY,
    28 28   --   time_and_time
    29 29 );
    ...

```

Adam: Worked on front end database work and specifically the website design. This includes ensuring the success of the login page with the required login credentials, as well as cleaning up the home page which were both HTML and then converted. Spent a lot of time trying to successfully create/implement the Google Calendar API but had some difficulties,

Ayden was able to complete it and fully implement it into our application. Created and completed the Project Presentation and helped with assigning the slides to the team members.



The image shows a presentation slide titled "Feed the Pets" for "CSCI 3308 Final Presentation". The slide features a photo of a golden retriever and a "My Fitness Pet" mobile application interface. Below the slide is a list of names: Leo Sipowicz, Ayden Smith, Adam Trunko, Catherine Xiao, and Josh Ziebold. To the right is a screenshot of a Google Calendar for "Ayden Smith" showing events from November 14 to November 20. The calendar includes a repeating event for "Feed Pet" at various times each day. A "Sign Out" button is visible at the top of the calendar. At the bottom right of the calendar is a "Google Calendar" link and an "Add Event" button.



The image shows a GitHub commit history for a repository. It lists three commits:

- Commits on Nov 15, 2021:
 - meeting notes by ATrunks committed 14 days ago
- Commits on Nov 12, 2021:
 - updating google calendar API by ATrunks committed 17 days ago
- Commits on Nov 8, 2021:
 - Adding milestone5 by ATrunks committed 21 days ago

Ayden: I Worked on the Front End, Back end and tied the two together. I first incorporated some of the html we used in our labs in our projects and modified them accordingly. These pages included the login, registration, and profile pages, I then set up the google API and embedded calendar on the calendar page. Next I set up NodeJS and the docker yml file so we could run our project locally. I then worked with Leo and Cat on the postgreSQL database and collaborated to figure out what information we needed to include. I also got the user profile page tied to the database with NodeJS. After that I worked with Leo on user authentication and creation at the same time working with Cat on displaying pet info on the pet profile page.

Finally I set up and locally ran our demo.

The image shows two GitHub pull request screenshots side-by-side. Both pull requests are from the user 'aydenmyl' and have been committed 13 days ago.

- Pull Request 1:** A diff showing 3 additions and 2 deletions. The file is 'projectdev/reviews/googlecalendar.ejs'. The code includes client ID and key configuration for Google Calendar.
- Pull Request 2:** A diff showing 19 additions and 4 deletions. The file is 'projectcode/server.js'. It includes logic to handle Google Calendar API requests and render responses.

The image shows two GitHub pull request screenshots side-by-side. Both pull requests are from the user 'aydenmyl' and have been committed 14 days ago.

- Pull Request 1:** A diff showing 86 additions and 78 deletions. The file is 'projectCode/init_data/database.sql'. It contains SQL code to create a 'users' table and insert a single row for a user named 'Ayden'.
- Pull Request 2:** A diff showing 520,018 additions and 439 deletions. The file is 'projectcode/docker-compose.yml'. It defines a Docker compose file for running Node.js and MySQL services.

Leo: I worked on lots of different components but mostly focused on the backend and user account creation and authentication. I created the initial file structure and html layout for our app and then began work on server.js file and signin.ejs and register.ejs files. First I added functionality for users data to be stored in the database if all fields were allowed and the password had required criteria. Then I created a function in server.js to lookup the password associated with the imputed username and to check if it matched the imputed password. I also redesigned the jira to make it easier to read and understand for our whole group and made style and html changes to the apps front end.

```

app.post('/register/create_account', function(req, res) {
    //Params taken from form
    var FirstName = req.body.firstName;
    var Email = req.body.Email;
    var familyID = req.body.familyID;
    var password = req.body.password;

    //PostgreSQL insert with params from form
    var insert = "INSERT INTO users(familyID, familyName, email, pass, numPets) VALUES (" + familyID + ", 'JohnsFamily', '" + Email + "', '" + password + "', 3);"

    //On successful insert take to homepage and store email cred
    db.any(insert)
        .then(function (rows) {
            login = EnteredEmail
            res.render('pages/home', {
                my_title: "Pet Feeder",
            })
        })
        .catch(function (err) {
            console.log('error', err);
            res.render('pages/Home', {
                my_title: 'Pet Feeder',
            })
        })
    });
}

app.post('/signup/login/account', function(req, res) {
    var EnteredEmail = req.body.Email;
    var EnteredPassword = req.body.psw;

    // Query to find the correct password for entered email
    var query = "SELECT pass FROM users WHERE email = '" + EnteredEmail + "'";

    // Executes query
    db.any(query)
        .then(function(rows) {
            //Store correct password for entered email in "TruePass"
            var TruePass = rows[0].pass
            //If truepass == entered pass store the user and go to home page
            if (EnteredPassword == TruePass) {
                login = EnteredEmail
                res.render('pages/Home', {
                    my_title: 'Pet Feeder',
                })
            }
            //Else render the signin page so user can try again
            else {
                res.render('pages/signin', {
                    my_title: 'Pet Feeder',
                })
            }
        })
        .catch(function(err) {
            console.log('error', err);
            res.render('pages/signin', {
                my_title: 'Pet Feeder',
            })
        })
    });
}

```

November 2021

2020



Created 22 commits in 4 repositories



2019

[CU-CSCI-3308-Fall-2021/CSCI-3308-Fall21-015-05](#) 16 commits



[LeoSipowicz/BoostrTool](#) 2 commits



[CU-CSCI-3308-Fall-2021/lab10-heroku-setup-LeoSipowicz](#) 2 commits



[CU-CSCI-3308-Fall-2021/lab9-testing-LeoSipowicz](#) 2 commits



Deployment:

Our project is a local web application so we don't have any external access to our app. To access our app you must first pull from our github repo,

<https://github.com/CU-CSCI-3308-Fall-2021/CSCI-3308-Fall21-015-05>

The screenshot shows a GitHub repository page for 'CSCI-3308-Fall21-015-05'. The 'Code' tab is selected, displaying the contents of the 'projectCode' directory. The directory structure includes 'css', 'images', 'init_data', 'js', 'node_modules', 'views', '.DS_Store', 'docker-compose.yml', 'package-lock.json', 'package.json', and 'server.js'. Each file has a detailed commit history with author, message, and timestamp.

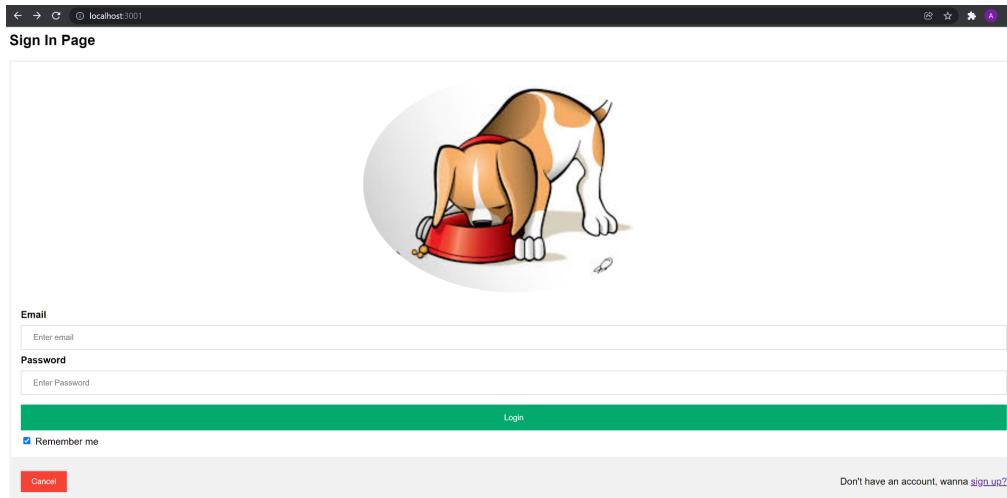
Then you must navigate to the projectCode directory.

After doing so you must run the command docker-compose up in your terminal and the init process will start and the app will be spun up

```
ayden@DESKTOP-UV2LCRF MINGW64 ~/OneDrive/Documents/CSCI 1300/HMW/CSCI3308/CSCI-3308-Fall21-015-05/projectCode (main)
$ pwd
/c/Users/ayden/OneDrive/Documents/CSCI 1300/HMW/CSCI3308/CSCI-3308-Fall21-015-05/projectCode

ayden@DESKTOP-UV2LCRF MINGW64 ~/OneDrive/Documents/CSCI 1300/HMW/CSCI3308/CSCI-3308-Fall21-015-05/projectCode (main)
$ docker-compose up
```

To access the working application you must finally navigate to the url <http://localhost:3001/> You should encounter a screen like this:



To log into the page you may either create an account with the hyperlink in the bottom right corner of the page or you may use one of the pre-existing accounts already loaded into the database.

Preloaded creds:

Email: `1234@colorado.edu`

Password: `Password1`