Honeycomb

System Design Document

EXECUTIVE SPONSOR : THOMAS MUSCARELLO

PROJECT TEAM : CHANCE ROSE, MATT WILSON, JIMMY CHEN, COREY HARDESTY, UNNATI THAKKAR, SAIF CHATHA

Original Plan Date: OCTOBER TWENTY-FIFTH

Revision Date: 10/18/2021

Revision: 1

Table of Contents

Audience

The design documentation is in general for anyone who wants to understand the system architecture and design of HoneyComb. The following groups are in particular the intended audience of the document.

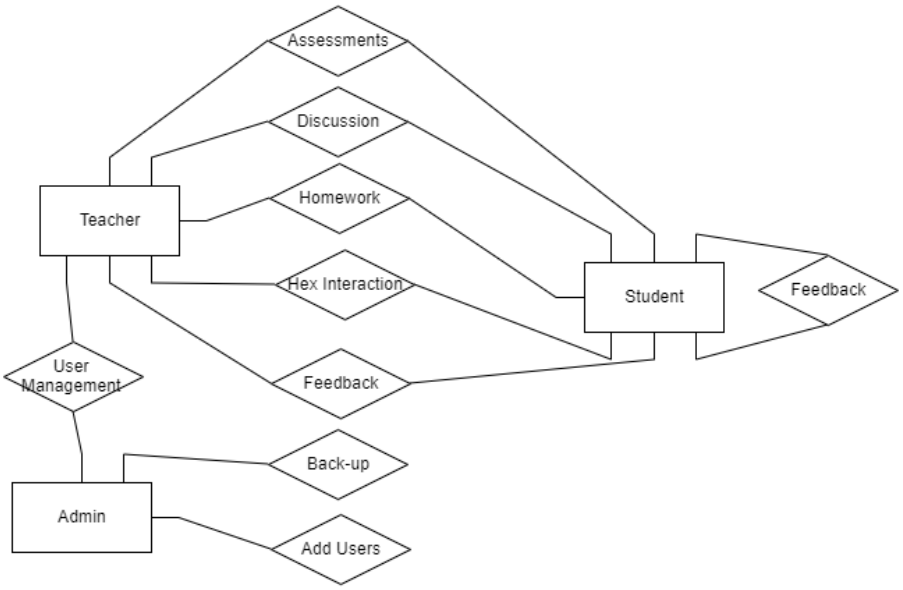
• HoneyComb executive sponsor – to evaluate that the system architecture and design support the requirements , to understand the architecture and follow the design to build the system, and to understand the internal workings of the system in order to administer the system effectively

Related Documentation

Testing and Assurance

Use Cases

Document Conventions



System Overview

Description

As stated in the requirements document, Honeycomb will enable students to join different hexes and then listen and interact with other students within that hex. Within that interaction they will be able to work on shared documents, voice chat will be enabled, and they will be able to provide feedback to one another after the interaction. The system will house all of this data in a separate database that will be connected to the web-application. Javascript will be used heavily within this project for all of the interactions.

System Architecture

HoneyComb is a web-application, consisting of primarily COTS products and customizations that are needed to meet the requirements.

Software Architecture

Honeycomb has the hexagon aspect which allows students and staff to connect, similar to a real life environment. Each hexagon represents a person whether it's a student or a staff member. If you click on one hexagon and another hexagon, you can pair them, so if it's two students, they can call and work together or be partners for a certain group activity. If you click on a hexagon which is a staff member and another which is a student, both can have a 1 on 1 video call.

Hardware Architectures

A web server running Apache to host the web application; storage for various application configuration files (e.g., CSS, images, etc.); and a pyrebase database server to host the 1-Click database.

Hardware Design

Hardware Components

Computer Systems

N/A

Peripherals

Monitor, Speaker, Webcam,Mouse, Keyboard, Microphone

Networks

HoneyComb would just require the internet to open a webpage.

Project Specific Hardware Items

N/A

Hardware Integration

Logical Design

The Model components represent application data and data access layers, the View components render data for the user, and the Controller components handle user inputs and manipulate both views and models.

Physical Design

The data is stored in firebase using python's pyrebase and flask.

Recovery Design

Pyrebase allows retrieval of the data from the database at any given time and point.

Software Design

Software Integrations include Zoom Api, react.js, node,js and hellenic react hex grid.

Software Packages

List and define all packages / modules. "Module" is synonymous with programs (libraries, executables, scripts, etc.)

- JavaScript React Library
- Hellenic react hex grid
- Mui material icons
- Node.js
- zoom api
- FireBase: DB
- PyreBase
- Python Flask Framework

Software Module

React.js, Node.js and Mui material icons are used for the creation of web pages.

Zoom api is used for the video calling aspect of the website.

Hellenic react hex grid is used for visuals of hex, which is an important aspect of our website and it's theme.

Software Integration

Unfortunately this whole project utilizes COTS for most if not all of its functionality and U.I. From a High-level view, users interact with webpage elements that are created and displayed using the React Javascript library. All user requests will be listened to, and interpreted by the Python Flask framework extension, including navigation. User authentication requests, file uploads/ downloads, and user account updates will all be handled through Pyrebase(FireBase Python extension), which is compatible with Flask and will relay the queries to FireBase, our database. FireBase will contain all data for User accounts, their authentication, user files, etc..

Once the request has been acknowledged/ fulfilled, the data will then be sent back to the Flask interface by use of PyreBase and will be used to populate data fields for display or relaying files back to the client. All site navigation will be controlled by Python Flask.

Data / Database / Files

Pyrebase is used as a database server, which features Firebase Real Time Database, Firebase Authentication and Firebase Storage.

Data Flow Diagrams

Provide different levels of DFDs: summary of top-level, system level (between system(s)/user/ device), for each major software module, and one-layer inside the software module.

Database Design

Pyrebase API does it all for us; it doesn't require running queries or having tables. The API takes care of authentication of the users as well the file storage and at some point we can also use it to show pull-up data for leaderboards etc.

Files

Data is all stored on the cloud on the firebase console and we can access it when needed.

Requirements Analysis Document

Purpose

Honeycomb is an interactive online environment used for remote learning and teaching. The main focus of Honeycomb is to emulate a real classroom online. Students can view assignments, submit, and interact with classmates. Also, students will be able to have access to voice and video calls functionalities with the professor.

Audience

Honeycomb's intended audience is for anyone that attends school (elementary, middle school, highschool, college, universities). That includes Faculty, staff and students. The honeycomb UI itself is user-friendly which makes it usable for all ages.

Template

Section/Topic	***Description***
1. Introduction 1.1 Purpose of the system 1.2 Scope of the system 1.3 Objectives and success criteria of the project 1.4 Definitions, acronyms, and abbreviations 1.5 References 1.6 Overview	*Honeycomb is an online interactive environment used for academic learning. It has features such as being able to view, submit, interact, and make video calls with other users of the platform. Students can hop into chat rooms with each other or with the professor. The professor has the ability to teach as a real life classroom (all at once) or even make groups room for students for certain activities. The objective of honeycomb is to bring online learning and teaching as close to real life learning in the real world.*
2. Current system	*Honeycomb will revolutionize online learning and teaching because it does what no other platform is able to do today, which is to imitate a real life classroom. Zoom is convenient for voice and video but not for organizing homeworks, projects or even being able to view or submit assignments or projects. D2L gives the ability to view,*

submit assignments and projects but lacks the communication side. Honeycomb brings it all into one platform making it easy and effective to be able to view, submit documents along with communication features. Also, Honeycomb has the friendliest UI for all age groups, including students attending elementary school.

3. Proposed system

Honeycomb is an interactive online learning environment similar to the d2l and blackboard where students can view assignments, submit, and interact with classmates. It will also have some basic video calling functionalities like zoom. What makes honeycomb unique will be the focus on one on one interaction with your peers and teachers using the special Honeycomb UI making it like an online classroom. Honeycomb would be a great way to host class online during things like the pandemic. It will ease the life of teachers, students as well as parents. Students will be able to give quick feedback to each of your peers using a scale and hop into multiple chat rooms seamlessly. Students can compete daily either by being the best at certain games, quizzes, or getting the highest homework score will be shown on a leaderboard and awarded honey points which will give real rewards.

3.1 Overview

Honeycomb is an interactive online learning environment similar to the d2l and google classroom where students can view assignments, submit, and interact with classmates.

3.2 Functional requirements ("shall lists")

3.2.1
3.2.2
3.2.3
3.2.4

Honeycomb's admin can add and remove users as and when needed, assign teachers to their classrooms, and students to their classes as well as link parent accounts to their child's account.
It allows teachers to post assignments, update grades, manage video calls via zoom, send reminders/messages to students/parents.The teacher also has the ability to click on any group of students to read their discussion history, talk to them individually, shuffle them around, and/or award points.The teacher also can give their power to a student or someone else to be their teaching assistant which will grant them same access.
It allows students to join video calls, interact with their peers, play games, view grades and submit assignments.
It allows parents to submit requested documents, view their child's progress, and receive reminders sent by teachers as well as they will be able to contact teachers.
All users can update info and adjust settings.
Authentication of all users when they try to log into the website.
Chat history and filters for teachers to view and set.

3.3 Nonfunctional requirements

3.3.1 Usability
3.3.2 Reliability
3.3.3 Performance
3.3.4 Supportability
3.3.5 Implementation
3.3.6 Interface
3.3.7 Packaging
3.3.8 Legal

The website is going to be mainly designed for k-6 graders so the UI is kept simple and easy to follow. The website will be capable enough to handle thousands of users without affecting its performance. The website will work across all browsers.

3.4 System models

3.4.1 Scenarios

3.4.2 Use case model

3.4.3 Analysis object model

3.4.4 Dynamic model

3.4.5 User interface navigational paths and screen mock-ups

Students would be able to interact, join calls and breakout rooms, view/ submit assignments, view grades and update account info. Parents would be able to view student grades (preferential feature) , update account info, receive notifications and contact teachers.
Teacher/professor can edit accounts, edit student accounts, upload and create assignments (projects), add grades, manage all types of calls (breakout rooms), message class and/or parents, etc. Admins can create classrooms, assign teachers/professors to classrooms, assign students and link parents accounts.

4. Glossary

A glossary of important terms, to ensure consistency in the specification and to ensure that we use the client's terms. A precurser to the ***Data Dictionary***

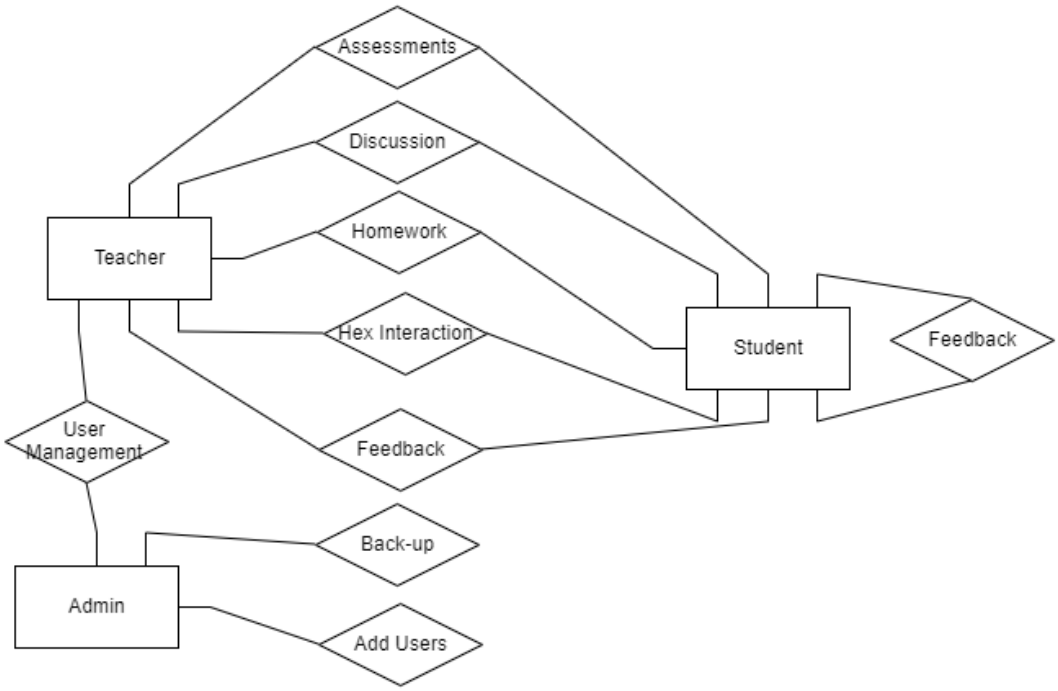
Honeycomb Use Cases and UML Diagram

1. **Title:** Teacher Uploading
2. **Goal:** Issue feedback, tests, homework
3. **Primary Actor:** Teacher
4. **Trigger:** Assignments need to be updated
5. **Scope:** Main interface, main database, teacher, and students
6. Throughout the school year the teacher will need to upload feedback, homework, tests, and assessments to Honeycomb. The teacher will access the application with credentials managed by the Administrator within the school. Then the teacher will select the proper folders to upload feedback, test and homework documents, and will also be able to submit grades within Honeycomb as well.

1. **Title:** Students Feedback
2. **Goal:** Provide feedback to teachers and other students
3. **Primary Actor:** Students
4. **Trigger:** students have finished working with each other
5. **Scope:** Discussion board, main hex
6. Students will be expected to provide feedback amongst themselves and for the teacher as well throughout the school year. Students will log on to the application using their credentials each day. Each day they will work within the main hex of honeycomb and utilize each other as resources. They will also work with each other on certain projects or homework assignments. Once any of these activities is completed, students will be given an (mandatory) opportunity to provide feedback regarding the other students they worked with. Every student will understand that everyone is able to provide feedback.

1. **Title:** School Administrator's Maintenance
2. **Goal:** Update class lists, add students and teacher assignments
3. **Primary Actor:** School Administrator
4. **Trigger:** New school year beginning or sudden changes
5. **Scope:** Classlists, internal database
6. Before every school year the classlists need to be updated and the teachers need to be assigned their classes. Since this is an elementary implementation, there will just be one set of students per teacher. The systems admin will login using their credentials and either add or take away students depending on the current situation. The systems administrator will also work with the teacher in assisting any user management issues. The systems admin is also responsible for maintenance and back-ups of the entire Honeycomb system.

1. **Title:** Everyday Class
2. **Goal:** Teacher and students interact within the hex and other forums
3. **Primary Actor:** Teacher
4. **Trigger:** The teacher initiates class at 8am.
5. **Scope:** Teacher, students, document usage, voice chat usage
6. Each day within the school year the teacher will logon using their credentials and ‘initiate’ the class at 8am, much like a normal elementary school beginning in the morning. Students will have until 8:15 to join the class, and will choose a hex when they sign in. Throughout the day the students will pair off in separate hexes in groups which enables them to work together. The entire hexboard, or Honeycomb, is modeled after a real life classroom, with the effort to make the learning process intuitive for children.



Gamification

Students can compete daily either by being the best at bee-bingo, wasp quiz, daily quest or getting the highest homework score which will then be reflected on a leaderboard and awarded honey points which will give real rewards.Also, the person(workerbee) to be on top of leaderboard will have a most valuable bee(MVB) badge on their profile which will last upto certain days. For daily quests, we will add a quest such as talk to 3 new people today = 5 points etc. Wasp quiz is basically like a pop quiz where teachers can add questions related to the topic they are teaching and quiz students. Bee-bingo is a regular bingo with a bee theme.

Notice: Nothing here is final

Honeycomb

Interactive Learning Environment for Children

Designed by Jimmy Chen

Group Members: Jimmy Chen, Chance Rose, Corey Hardesty, Unnati Thakkar, Matt Wilson, Saif Chatha

What is Honeycomb?

- Honeycomb is an interactive online learning environment similar to the d2l and blackboard where students can view assignments, submit, and interact with classmates. It will also have some basic video calling functionalities like zoom.

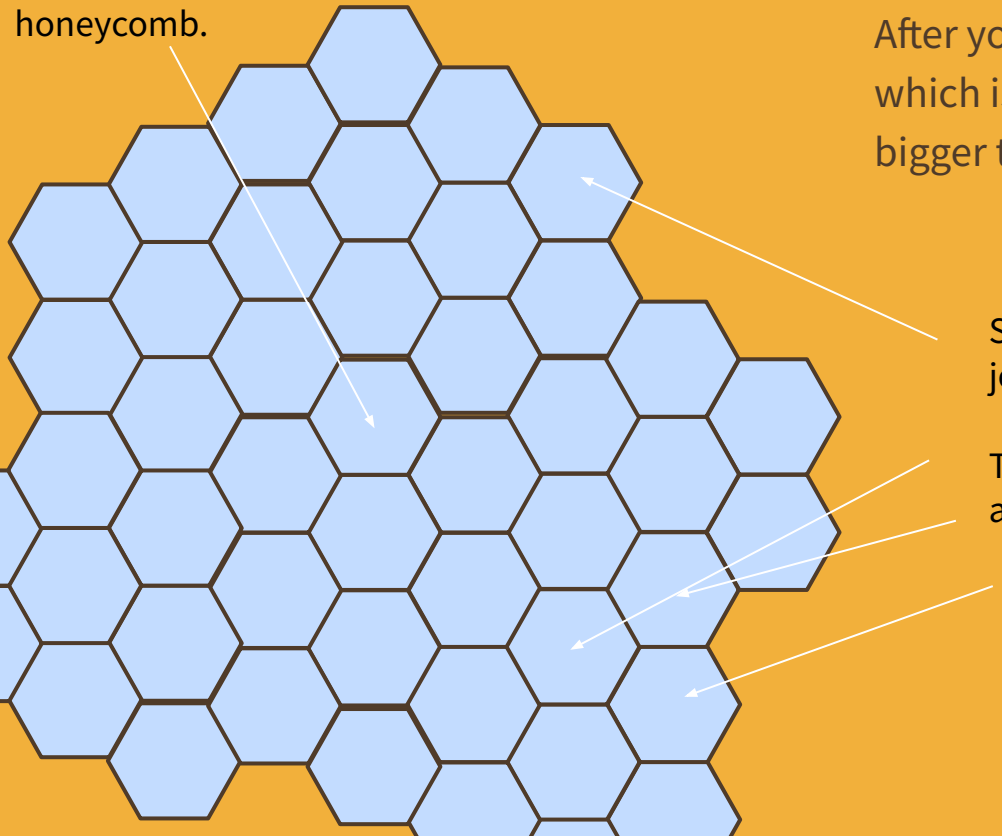
- What makes honeycomb unique will be the focus on one on one interaction with your peers and teachers using the special Honeycomb UI making it like an online classroom. Honeycomb would be a great way to host class online during things like the pandemic.

- You will be able to give quick feedback to each of your peers using a scale and hop into multiple chat rooms seamlessly.

- Students can compete daily either by being the best at certain games, quizzes, or getting the highest homework score will be shown on a leaderboard and awarded honey points which will give real rewards.

The Honeycomb - Student view

Middle square is reserved for the teacher and blows up their video on the top of the honeycomb.



After you login, you will be sent to the “Honeycomb” which is the central hub of this service. It’ll be a lot bigger than what is pictured here.

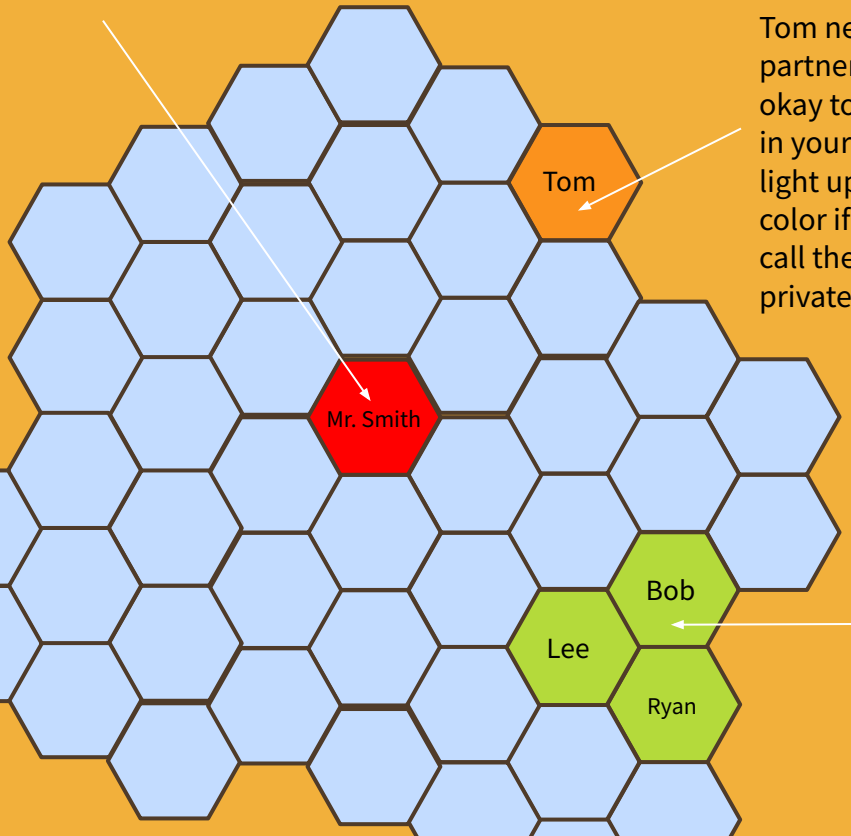
Students will click on a unoccupied hex to join that room.

Teachers can also drag a group of students to a respective hex and split off groups easily.

The bees don’t move and have the student’s name on it. They are in some UI hub on the right

The Honeycomb - Occupied space

The entire comb can only hear the teacher's voice and video



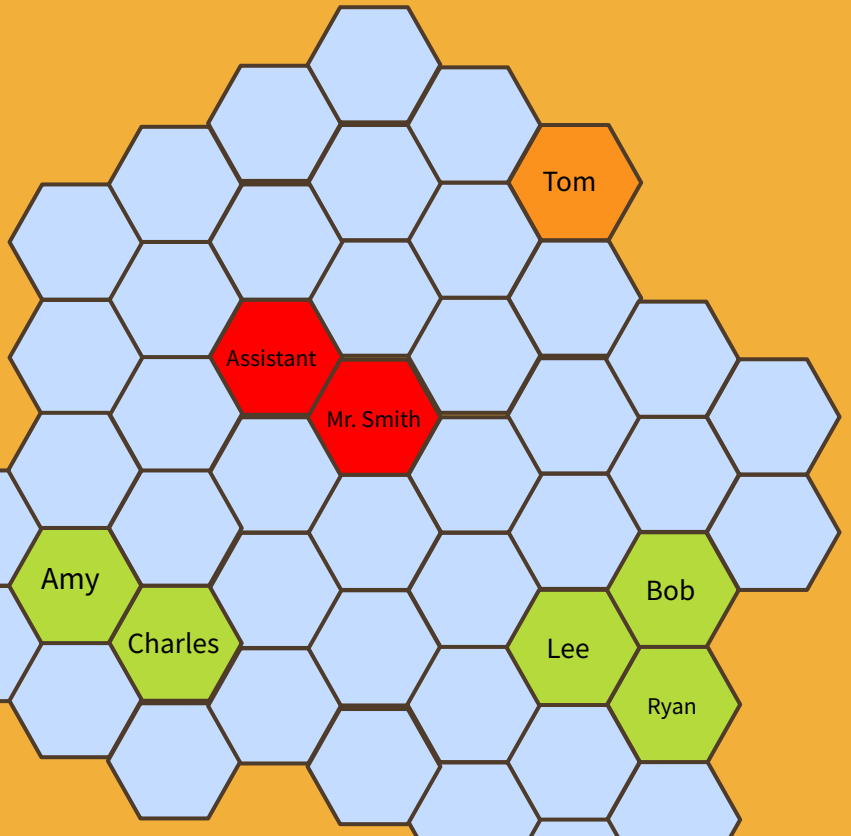
Tom needs a partner but it is okay to work alone in your hex. It will light up a different color if you need to call the teacher privately.

Once clicked, the comb will light up signal that it is occupied. Once two or more students occupy a comb next to each other, their individual chat rooms will pop up with the option for video and voice.

These students are engaged in a chatroom and may work together to answer questions the teacher proposes and on quizzes. They will all be awarded with full points and then give feedback to each other on who helped the most or knew the most. Enough good feedbacks will give u the title of MVB (Most valuable bee) for the day and give extra points.

The Honeycomb - Teacher Powers

Video of the teacher goes here



Universal chat box here

The teacher also has the ability to click on any group of students to read their discussion history, talk to them individually, shuffle them around, or award points.

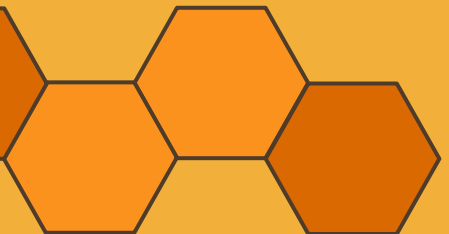
The teacher also can give a student or someone else power to be their teaching assistant which will grant them the same power.

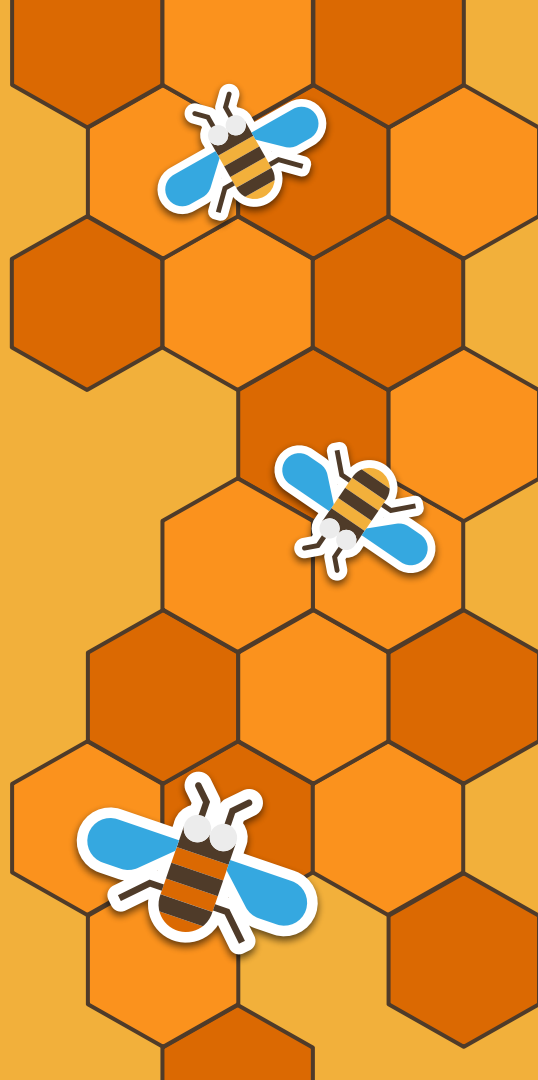
Teachers can also assign a group of students the ability to speak to the entire comb directly for presentation purposes.

Teachers can unlock certain games for each chat room during free time. These games award points.

Basic Web Layout Brainstorm

Classlist	Assignments	Honeycomb Central	Leaderboards	Honey Exchange
Everybody in your class. Can also be a honeycomb	Where you turn in your homework/ Calendar Best grade gets honey points.	The honeycomb	Daily, weekly, monthly, and yearly leaderboards for grades, games, and more	Exchange honeypoints for real merchandise





Extra features and naming convention

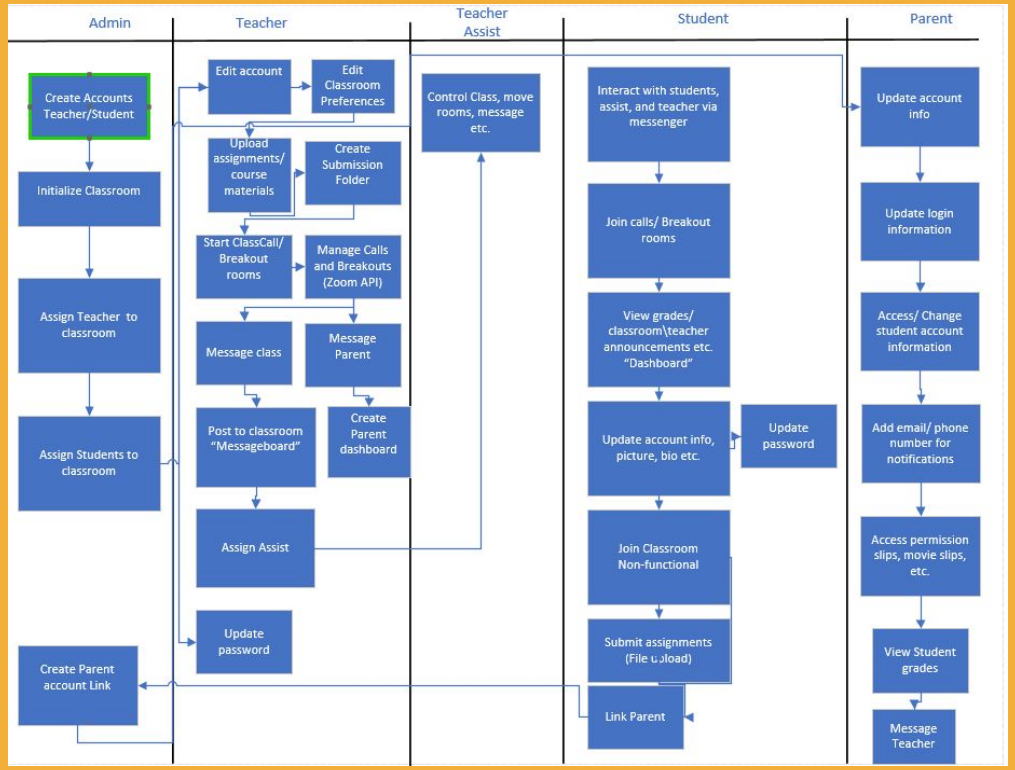
- Queen Bee: Teacher; male or female
- Worker bee: Students
- Wasp Quiz: Pop quiz?
- Spellingbee: Of course
- MVB (Most valuable bee)
- Beevatar
- Timer feature in each group of hexes to get to know people
- Chat filters
- Mini games related to bees/honey
- Daily quest? Talk to 3 new people today = +points
- Students can go on the honeycomb whenever they want (outside of class time) and click a button to notify the teacher or another student for 1 on 1 discussion or group work.
- There could be a password on the hexes
- The bee theme is intended for children, the honeycomb can have more serious uses. It can be used for a college lecture or your job. It could even be implemented at coffee shops, libraries, or even used for dating.

Team Members and Roles

Chance Rose - Documentation and Project Manager
Matt Wilson - Front-End/JS
Jimmy Chen - Front-End/Design
Corey Hardesty - Back-End/Database
Unnati Thakkar - Back-End/Database
Saif Chatha - Front-End/Back-End

Use Case Diagram

A mock up of potential WorkFlow



Logistics

For the next couple weeks

Front End

Most likely using Angular.js, React.js, or Veu.js

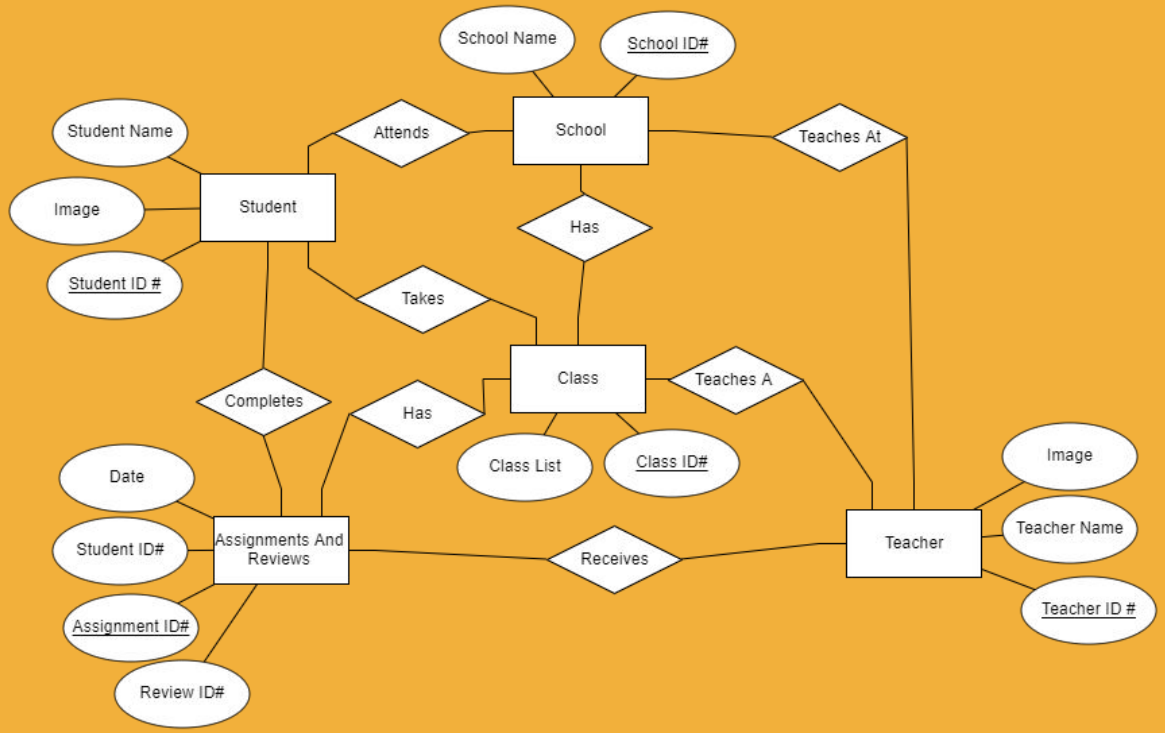
- Create “website” itself.
- Create classroom environment (GUI)

Backend

Most likely using Python Flask for JS

- Create backbone of classroom
- Create User accounts- with proper permissions

ER Diagram - Early stages



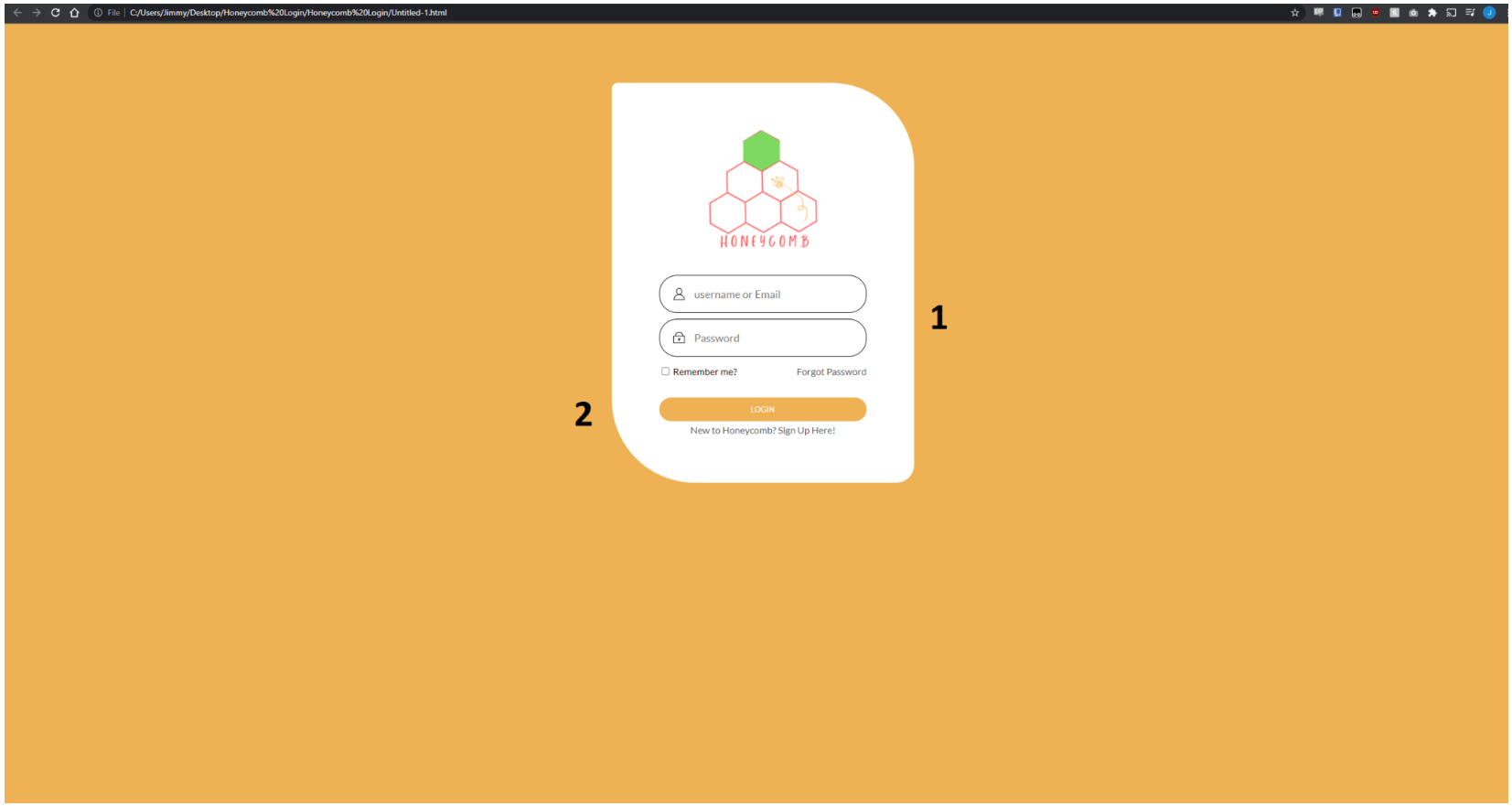
Data Dictionary for Database

Data Dictionary

Data Dictionary outlining a Database for HoneyComb

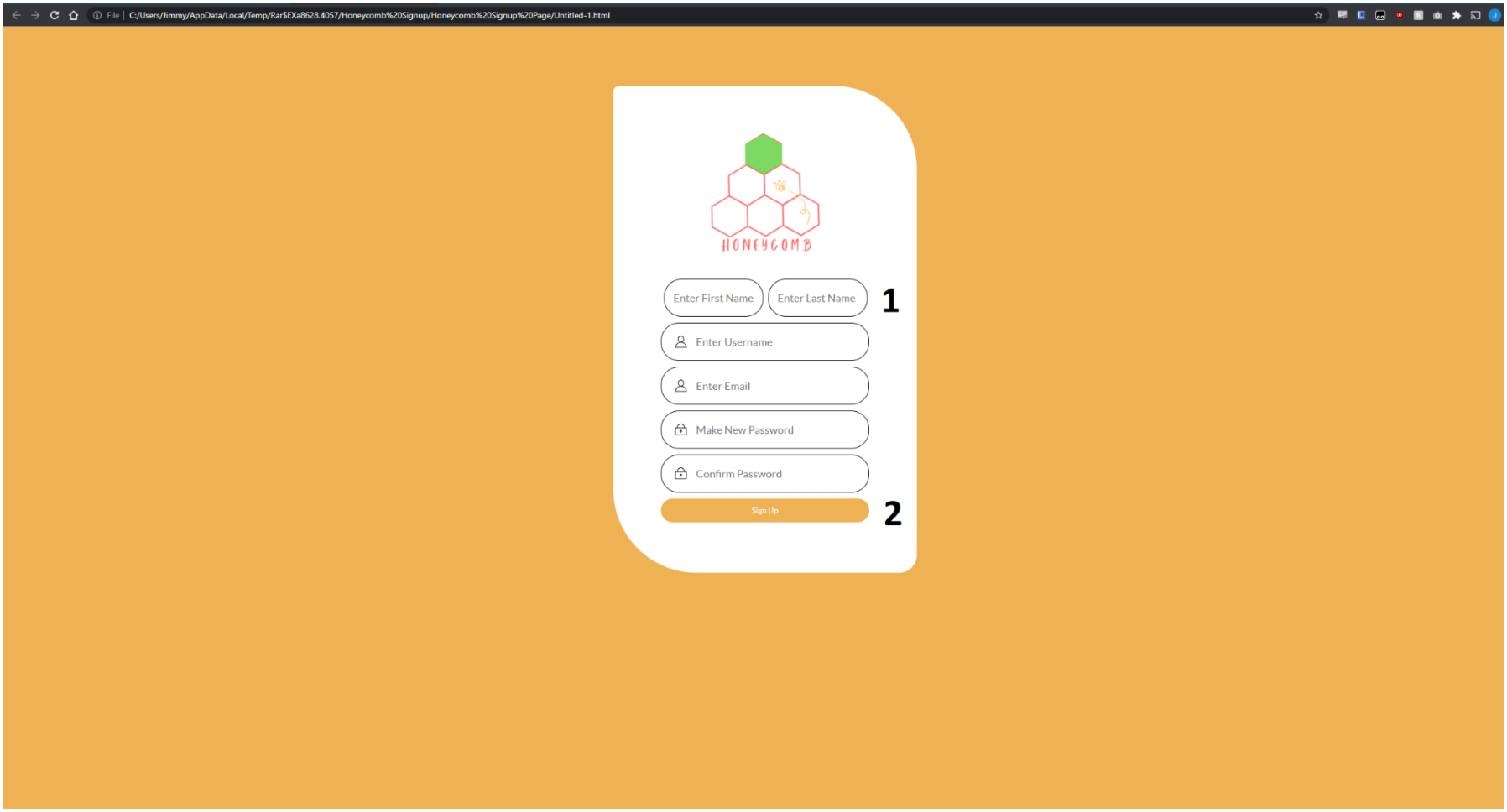
Field Name	Data Type	Data Format	Field Size	Description	Ex.
Student					
Student Name	Text		20	Studen'ts Name	Jimmy Chen
Student ID#	Integer	XNNNNNNNN	8	Unique identifier for each student	#12345678
Image	Image	.jpg	---	Image for students	---
School					
School Name	Text		20	Name of School	Lily Lake Elementary
School ID#	Integer	XNNNNNNNN		Unique identifier for school	
Teacher					
Teacher Name	Text		20	Name of teacher	Thomas Muscarello
Teacher ID#	Integer	XNNNNNNNN	8	Unique identifier for teacher	#87654321
Image	Image	.jpg	---	Image for teacher	---
Class					
Class List	List		15	List of students in each class	
Class ID#	Integer	XNNNNNNNN	8	Unique identifier for each class	#29387420
Assignments and Reviews					
Date	Date/Time	DD/MM/YYYY	10	Date for each assignment/review	09/27/2021
Student ID#	Integer	XNNNNNNNN	8	ID of student associated with assignment	#12345678
Assignment ID#	Integer	XNNNNNNNN	8	Unique identifier for each assignment	#00000001
Review ID#	Integer	XNNNNNNNN	8	Unique identifier for each review	#12000000

Login Page:



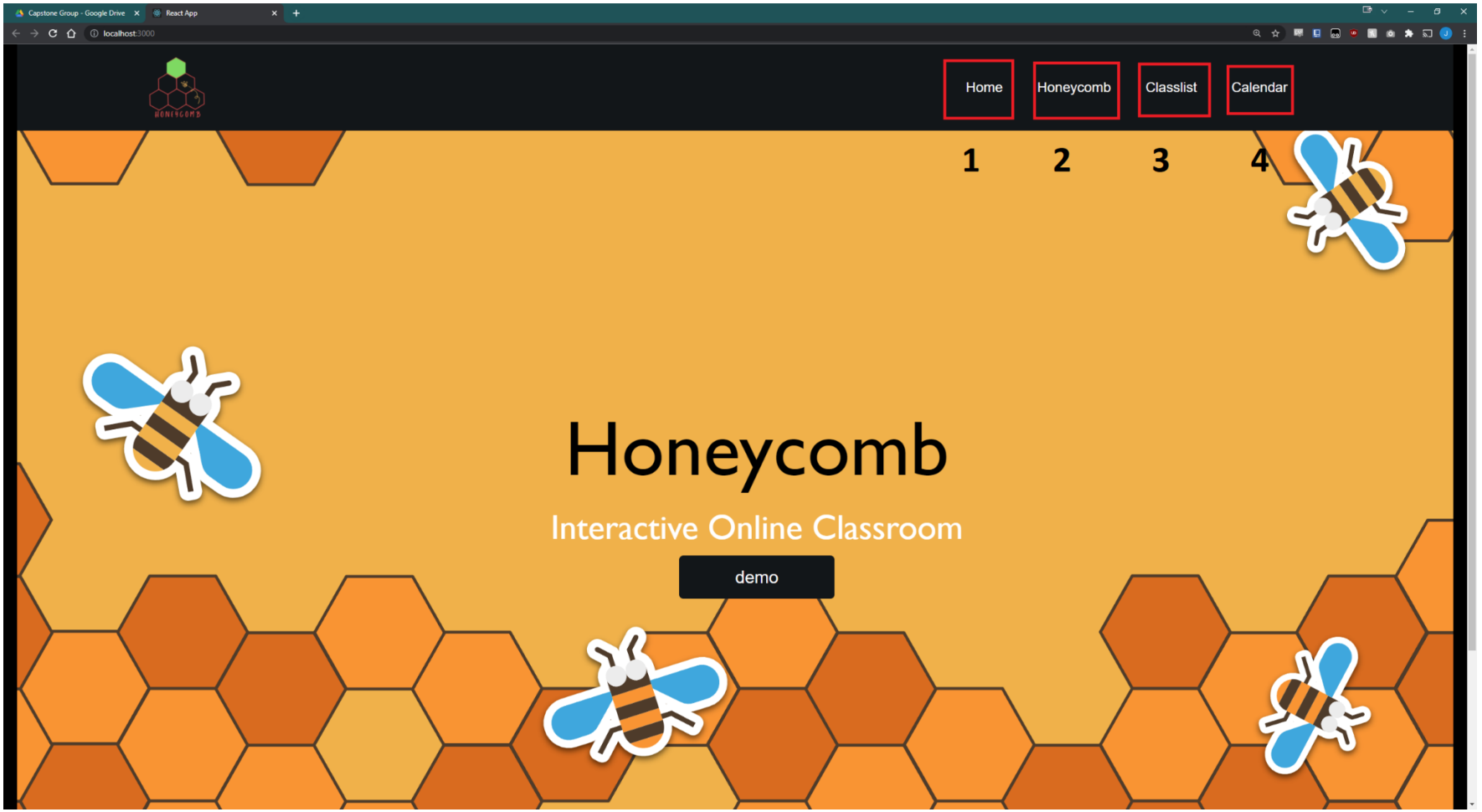
1. The first two fields are for username/email and password.
2. The button below the fields is to submit the fields and for the user to login.

Sign-Up Page:



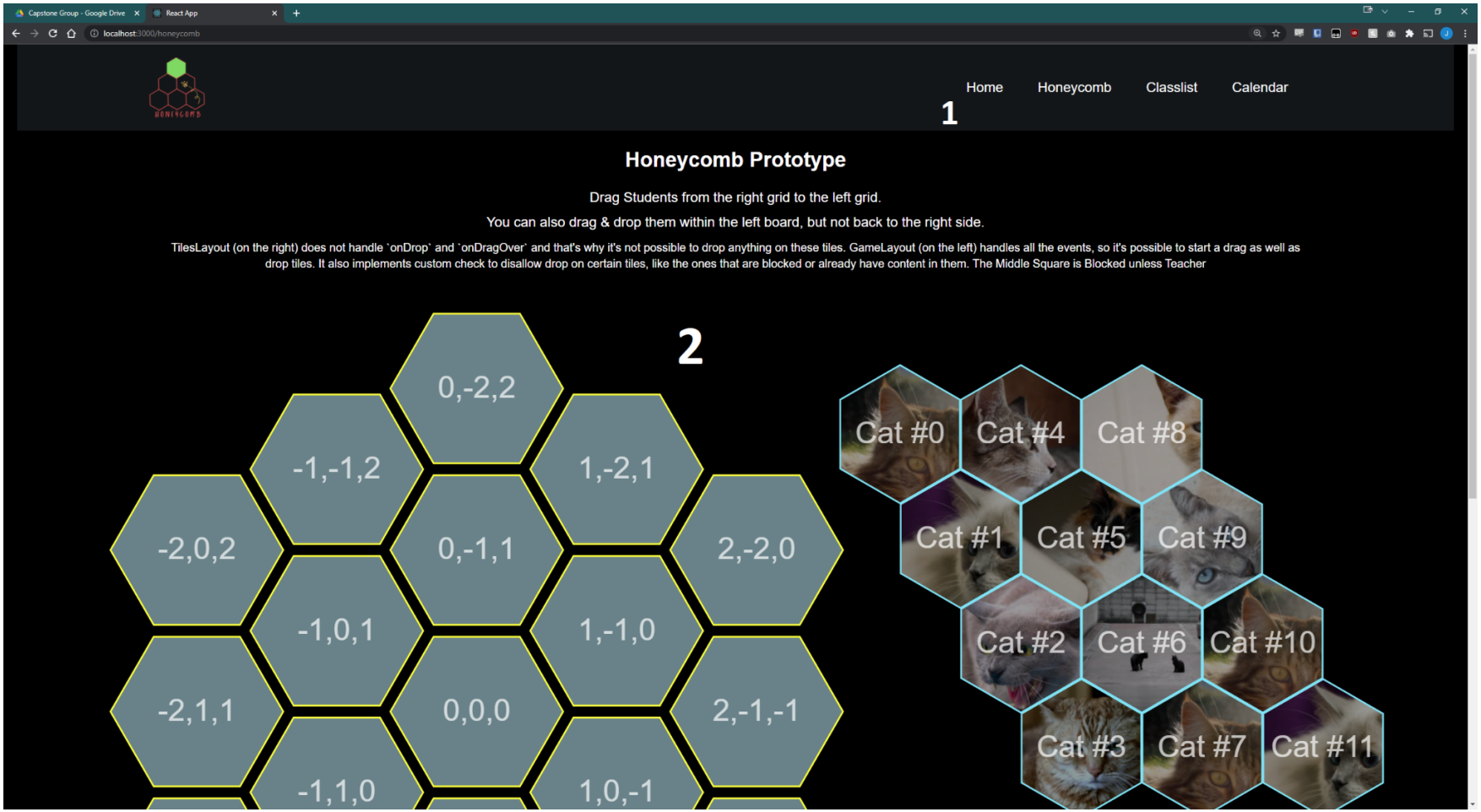
1. First Name, Last Name, Username, Email Address, Password, and Password Confirmation Fields
2. Sign Up button to submit the form

Main Page:



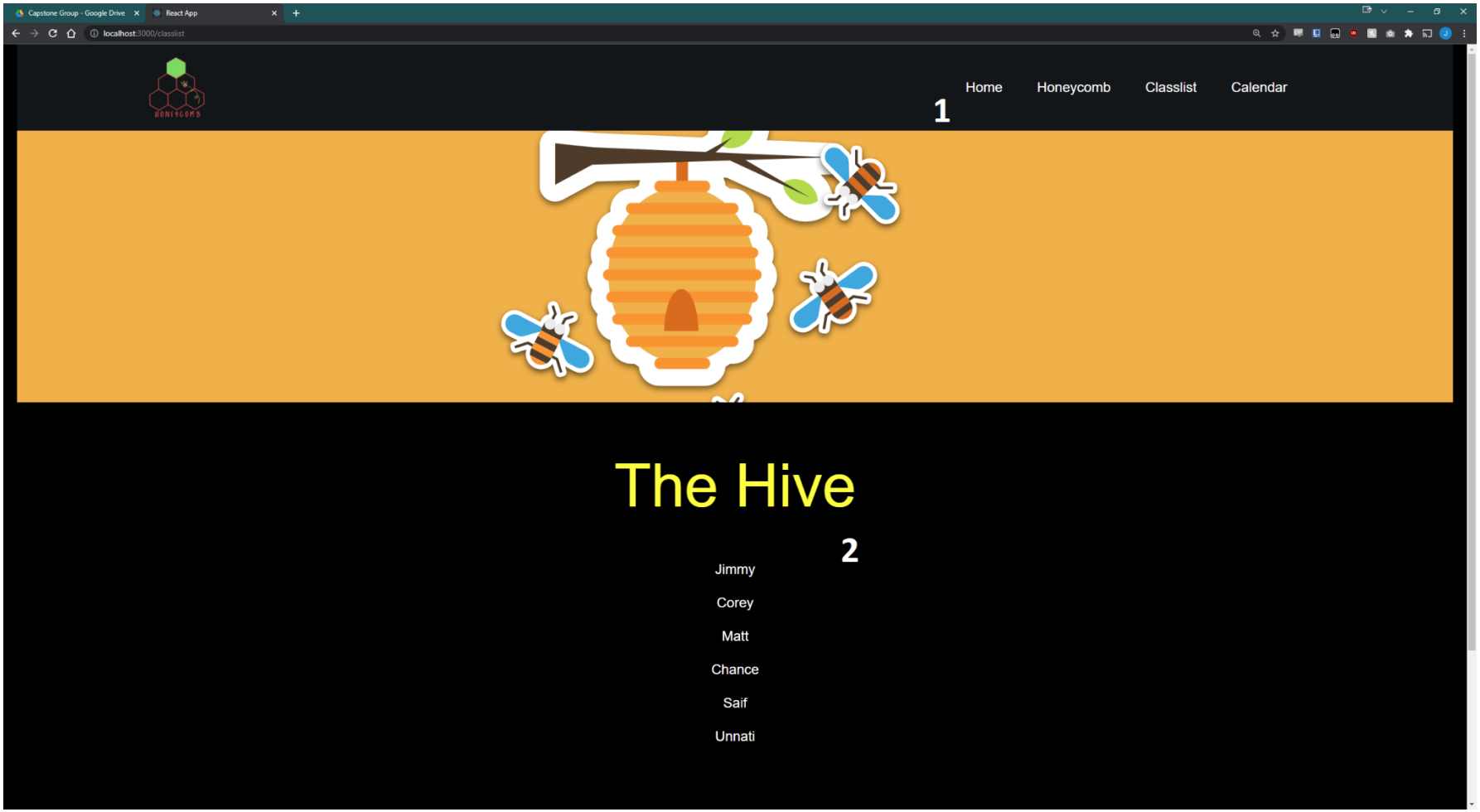
1. This top banner is on every page once they are logged in. The first selection takes the user to this landing page.
2. The second selection takes the user to the Honeycomb.
3. The third selection takes the user to the Classlist.
4. The fourth selection takes the user to the Calendar page.

The Hex:



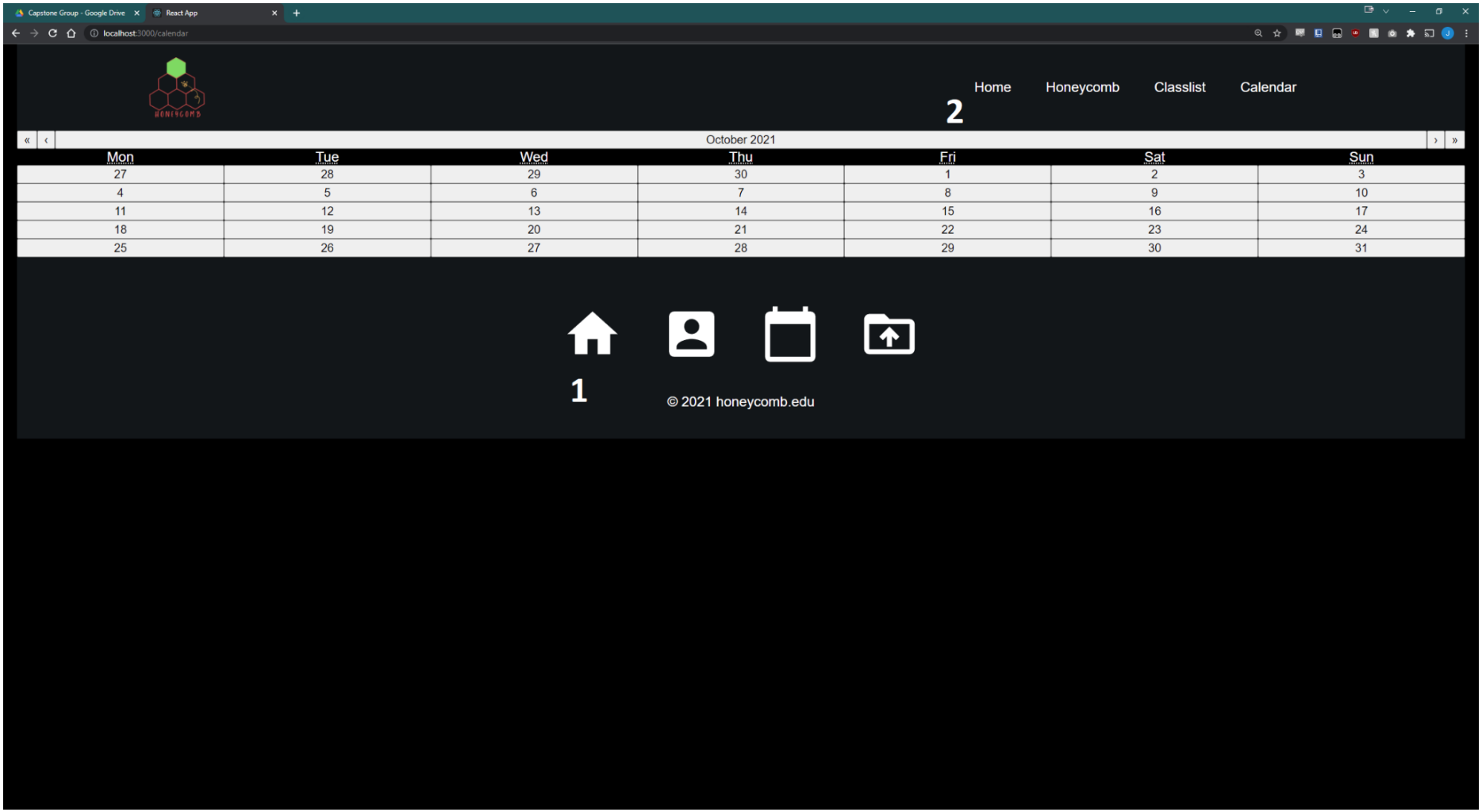
1. The top navigation banner, which appears in every page once signed in.
2. This is a prototype version of the main Honeycomb and its functionality.

The Classlist ‘The Hive’:



1. The top navigation banner, which appears in every page once signed in.
2. This is ‘The Hive’ which is the classlist for the current user’s Honeycomb.

The Calendar:



1. These bottom buttons also function as navigation below the calendar.
2. The top navigation banner, which appears in every page once signed in.

Repositories New
Find a repository...
depaulcp / csc347-20221-chenj181
ChenJimmu / honeycomb-main

Recent activity

When you take actions across GitHub, we'll provide links to that activity here.

All activity

unnati1028 pushed to **ChenJimmu/honeycomb-main** 2 days ago

1 commit to `master`

`c56fe1b` added game

fattymurkmcree pushed to **ChenJimmu/honeycomb-main** 9 days ago

2 commits to `master`

`906be38` Merge branch 'master' of https://github.com/ChenJimmu/honeycomb-main

`ef58ba0` added user class(may need changes)

mattwilson73 pushed to **ChenJimmu/honeycomb-main** 12 days ago

1 commit to `master`

`96d0521` Create newpage.js

mattwilson73 pushed to **ChenJimmu/honeycomb-main** 13 days ago

2 commits to `master`

`6278e2f` Merge branch 'master' of https://github.com/ChenJimmu/honeycomb-main

`e8c6c06` added quiz game

fattymurkmcree pushed to **ChenJimmu/honeycomb-main** 13 days ago

2 commits to `master`

`bcea5f6` change

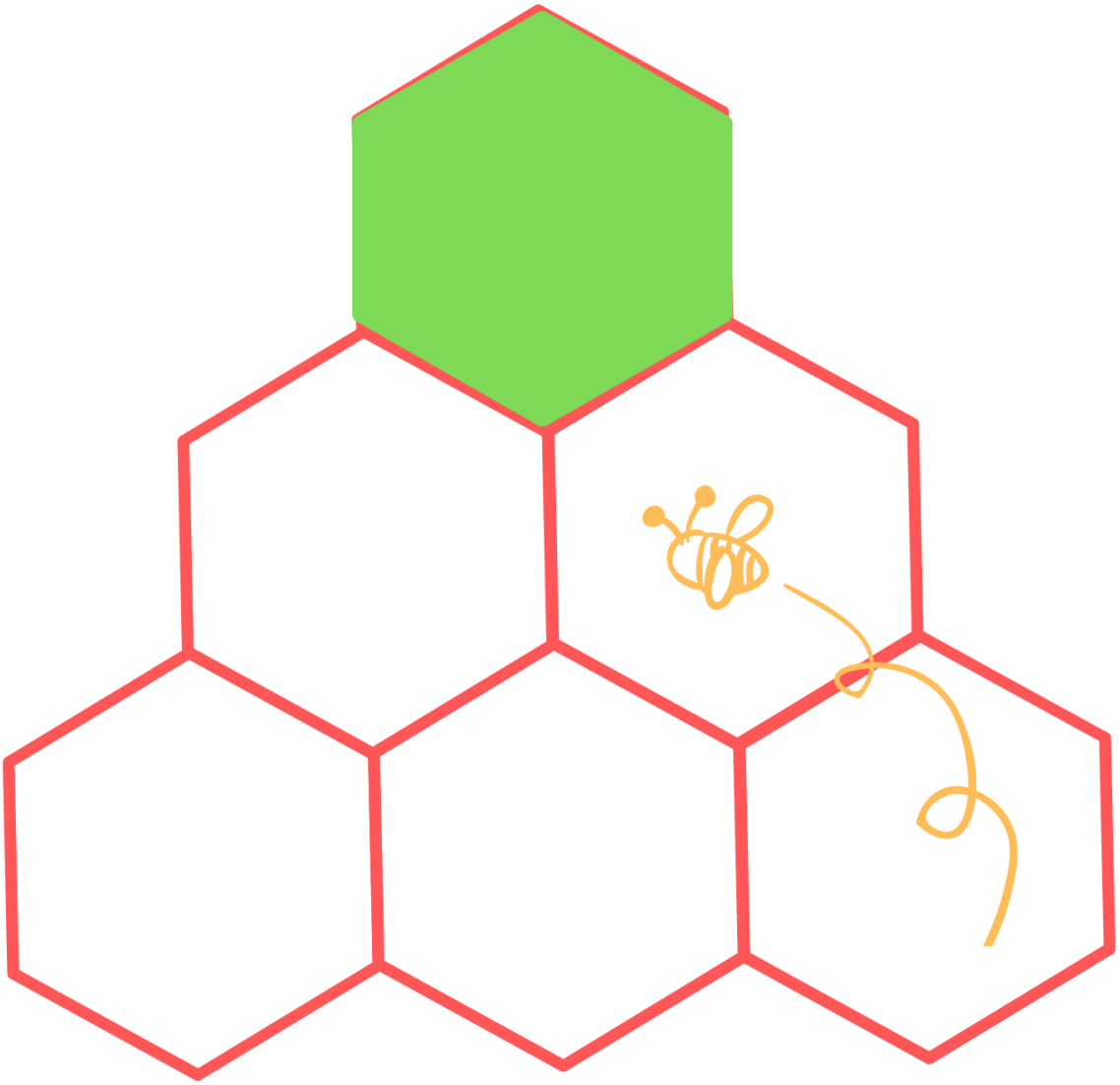
`bbc76d5` please push

fattymurkmcree forked **fattymurkmcree/honeycomb-main** from **ChenJimmu/honeycomb-main** 13 days ago

ChenJimmu/honeycomb-main

Python Updated Nov 20 ☆ Star

Systems Administration Manual



HONEYCOMB

Glossary

Bee - Astudent user account

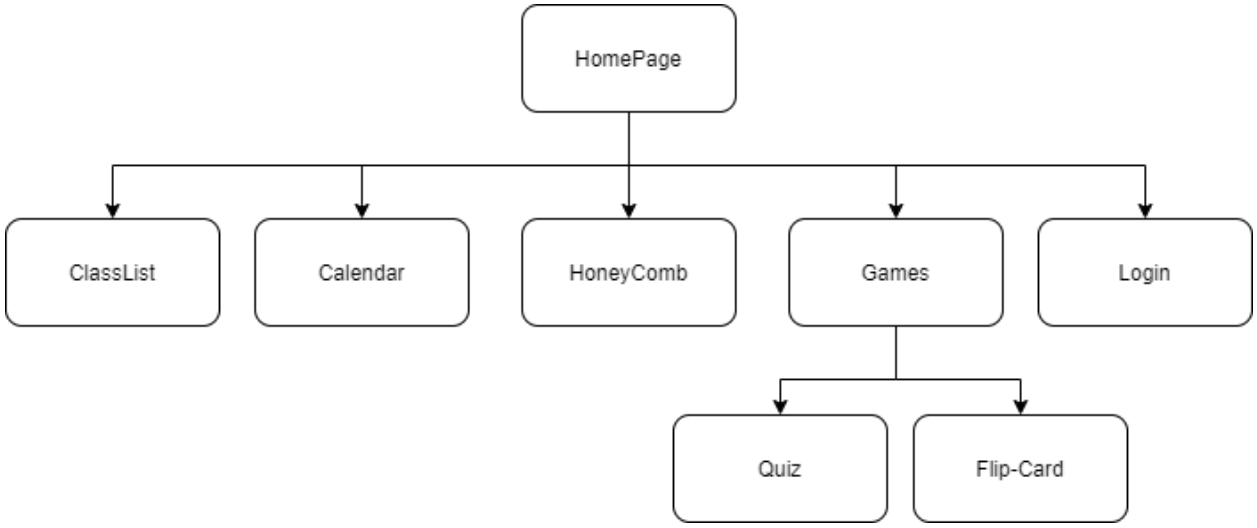
Queen - A teacher student account

SYSTEM OVERVIEW

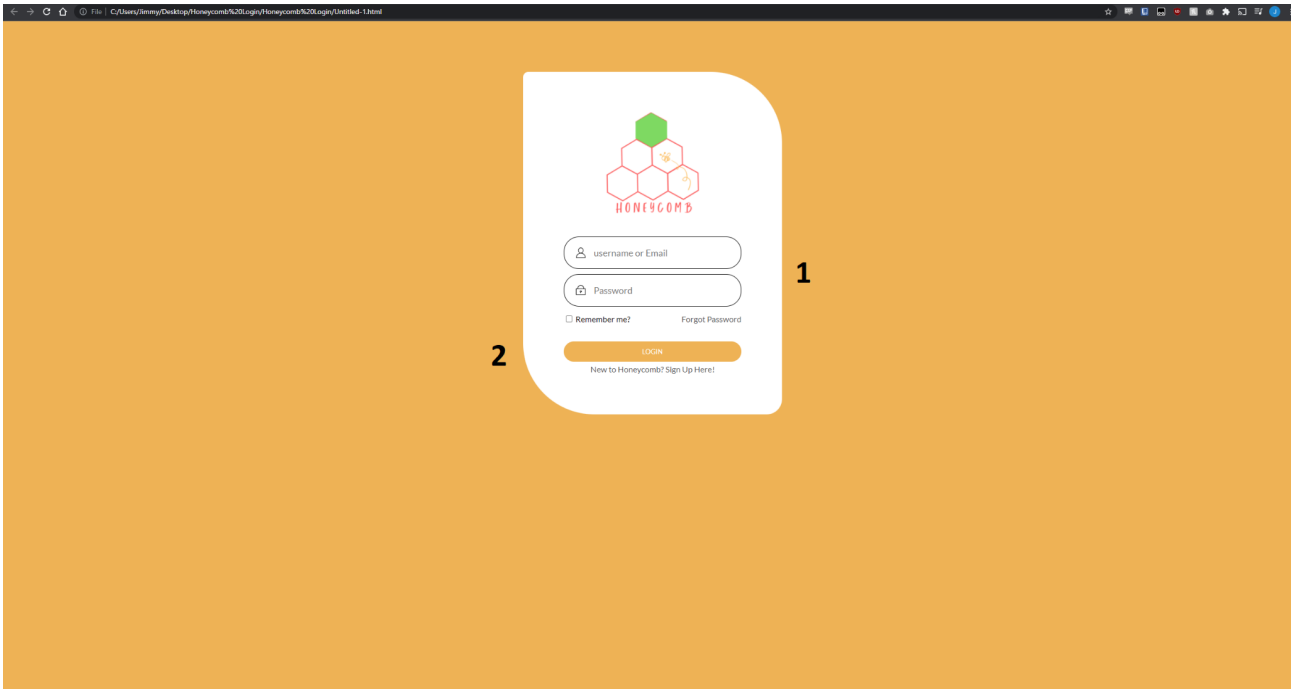
System Application

This section provides a brief description of the system, including its purpose and uses.

System Organization

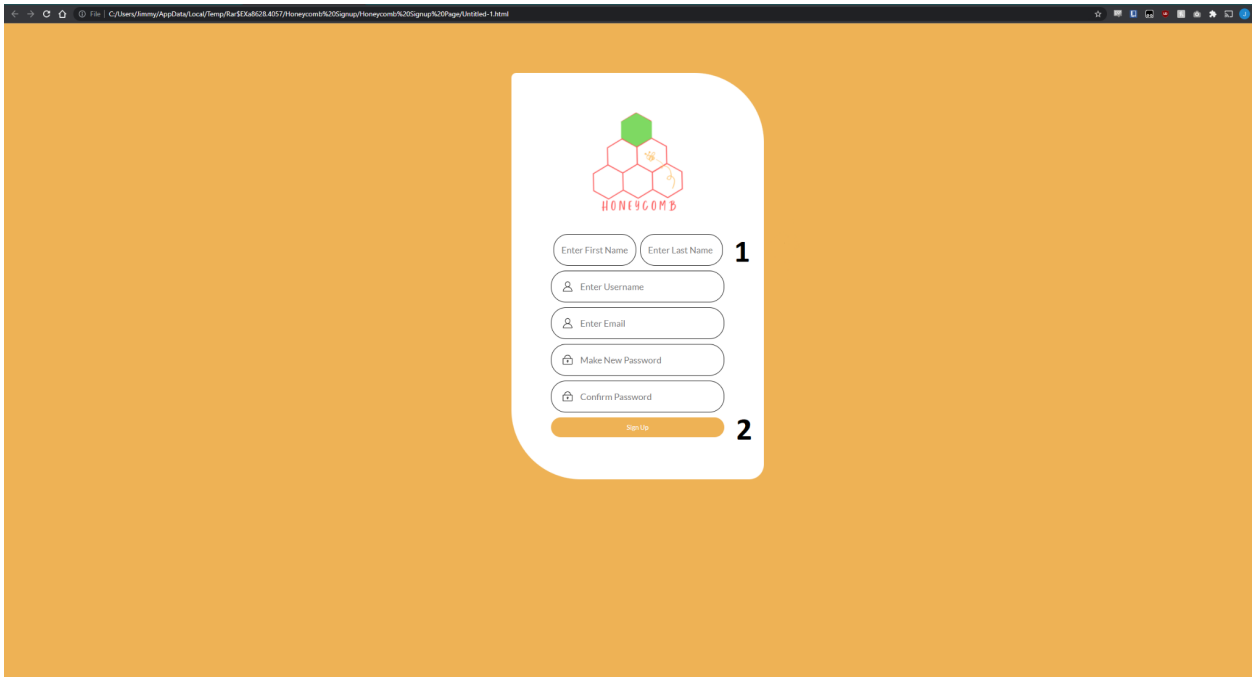


Login Page:



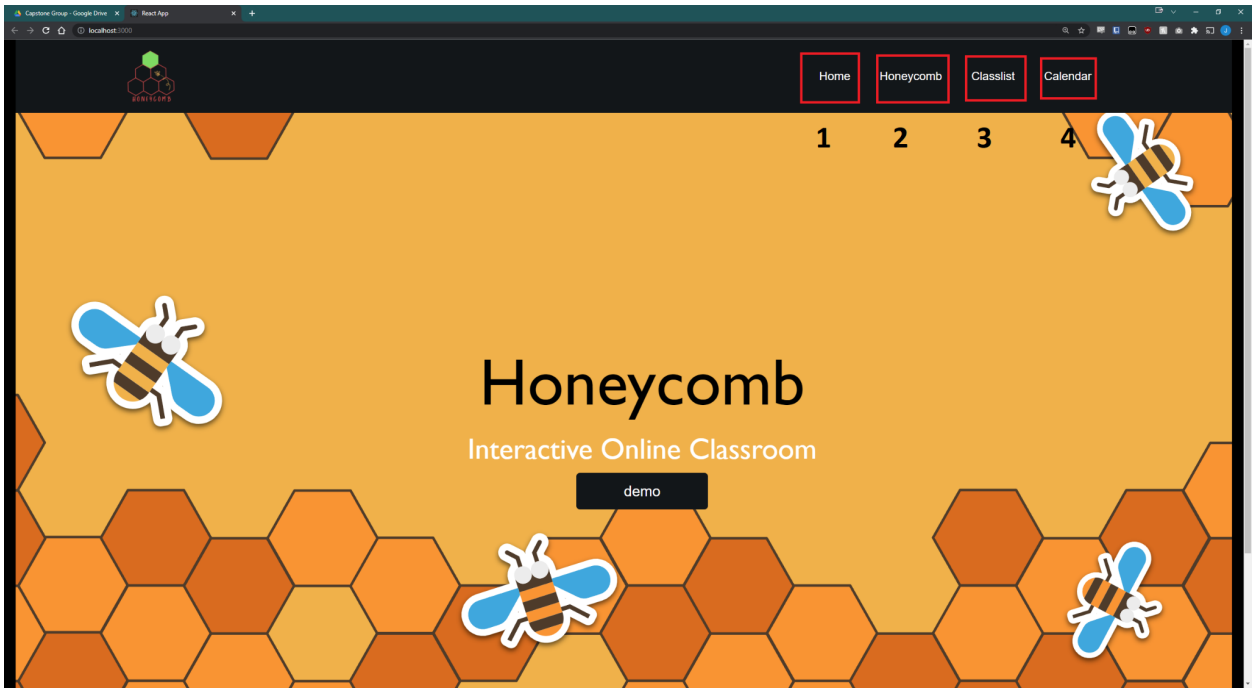
1. The first two fields are for username/email and password.
2. The button below the fields is to submit the fields and for the user to login.

Sign-Up Page:



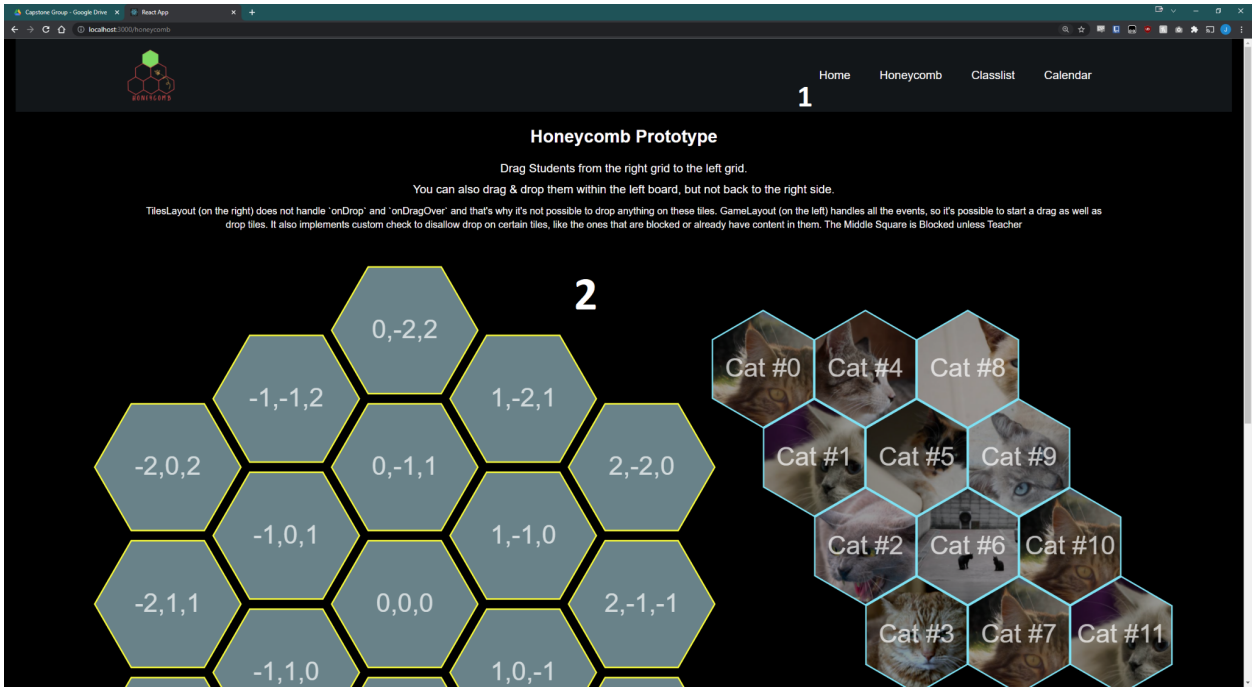
1. First Name, Last Name, Username, Email Address, Password, and Password Confirmation Fields
2. Sign Up button to submit the form

Main Page:



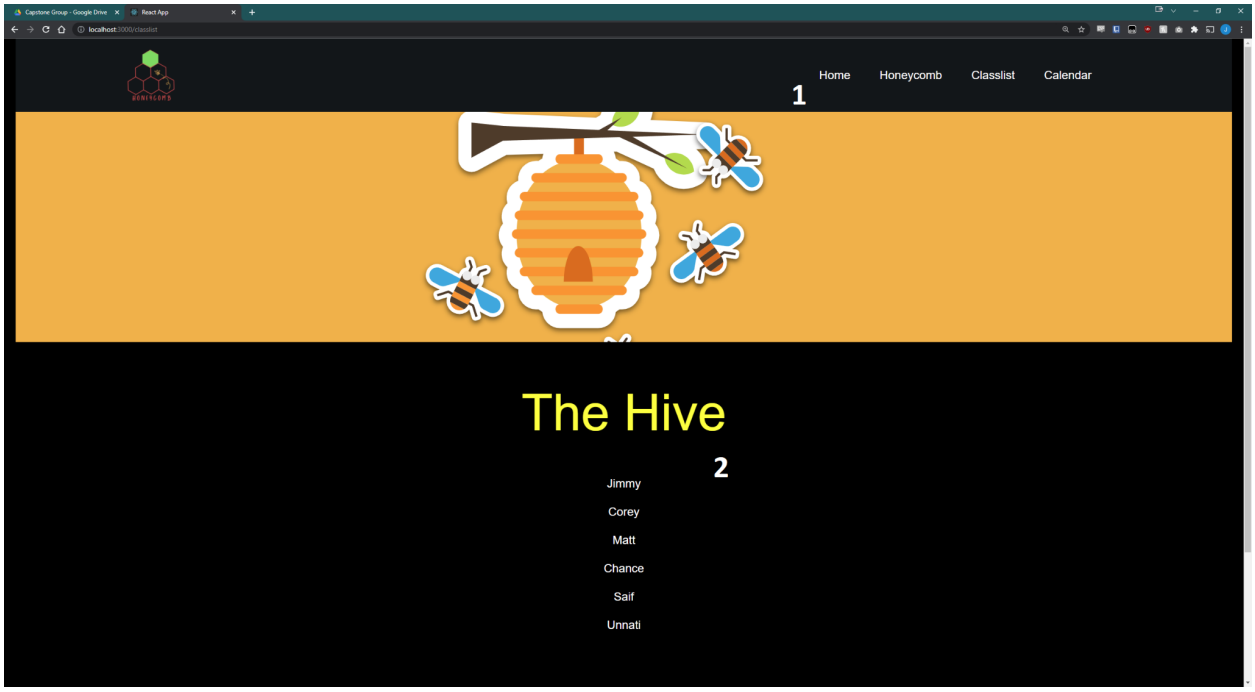
1. This top banner is on every page once they are logged in. The first selection takes the user to this landing page.
2. The second selection takes the user to the Honeycomb.
3. The third selection takes the user to the Classlist.
4. The fourth selection takes the user to the Calendar page.

The Hex:



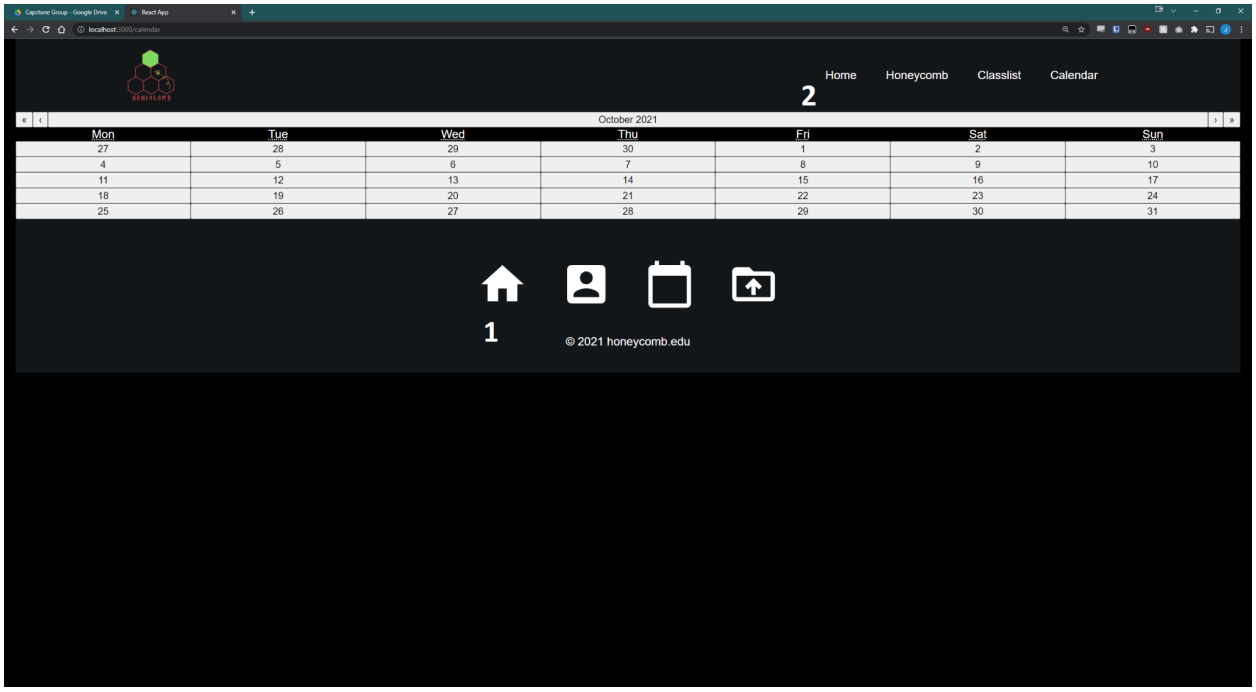
1. The top navigation banner, which appears in every page once signed in.
2. This is a prototype version of the main Honeycomb and its functionality.

The Classlist ‘The Hive’:



1. The top navigation banner, which appears in every page once signed in.
2. This is ‘The Hive’ which is the classlist for the current user’s Honeycomb.

The Calendar:



1. These bottom buttons also function as navigation below the calendar.
2. The top navigation banner, which appears in every page once signed in.

Resource Inventory

The administrator will need a working honeycomb API key and configuration keys.

Once the API key is acquired the administrator will need to set up certain objects that correspond to functional components of the database.

To create a firebase object use the command.

“firebase= pyrebase.initialize_app(firebaseConfig)”

**where firebaseConfig is the configuration key

To create an authentication object for user logins.

“Auth = firebase.auth()”

To create a storage accessing object.

“storage= firebase.storage()”

To create a database accessing object.

“db = firebase.database()”

SYSTEMS ADMINISTRATION

HoneyComb's system administrators will need to keep user authentication up to date along with the user data associated with the specific client. User data will be used to populate the client’s U.I when accessing the webapp. Administrator’s also have the task of structuring the database with specific courses and their corresponding sections, in order for client’s to receive the appropriate data.

User and Group Accounts

End user’s will consist of Parent’s, Students, and Teachers.

End User data will be stored under and with the following.

Teachers will be associated with the courses they teach, as well as the classlist of students attached with the teachers specific section.

Student’s will be associated with the courses they are in, as well as a potential “parent” account, that will give a parent access to their student’s data.

Parent’s will be associated with their student’s user data. Only being able to access what the student is allowed and also, will have access to contact their student’s teachers.

Adding/Deleting Users

To add a user the Administrator will need to access Firebase from a browser.

Once on the FireBase console, go to the “Authentication” tab.

Hit “Add User” and enter the email and password.

After the user authentication data is entered the administrator can either enter the user data through the command line or through the FireBase database GUI.

If the command line option is chosen, the administrator will create a data object of the form “data = {‘Email’: , “Name”: , “Courses”: []}”

Followed by the command “db.child(x).child(username).set(data)”

**where x is the desired user type(Teacher, Student, Parent)

If the FireBase web page GUI option is chosen follow the interface and create the directory as the example shows.

To Delete a User the Administrator must remove the desired account from the authentication list. Furthermore, if the user data is to be deleted the administrator must manually delete the user’s data from the database.

Setting User Permissions

User permissions pertaining to navigable options and pages will be handled by the user data object itself.

Controlling the viewing of individual grades, assignment submissions, etc. is controlled by the userdata object populating the paths at runtime for the specific user.

Database Maintenance

Database User/Group Access

The Administrator will have access to the full paths of the database. As of right now, the two “sub” databases that must be maintained are the user data database which holds all of the user data and the Courses database that will contain data pertaining to the given course.

The administrator will control the addition/deletion of user accounts and also the addition/deletion of Courses and their corresponding course section.

Setting User Permissions for Database

The permissions will be controlled by the backend through populating the database query with paths generated by a successful login. If a successful login did not occur, queries cannot be sent.

Adding/Deleting Groups for Database

Through the command line the administrator may enter the command

“ db.child(x).push(data)”

Where x is the name of the new group to be added and the “child” method acts as a subdirectory route.

Where data is the json object to be pushed to the db

Re-indexing Database

Indexing of the database is done by using the before mentioned “.child” method of the database object.

Updates to paths and data members can be done by using the ".update" method of the database object. Update can be used to create new paths, edit paths, or to update individual data entirely.

Application Maintenance

Application User/Group Access

FireBase authentication will handle user login. From there the User object will be returned. Within the user object will be the specific paths of courses, classlists, etc. that will be displayed dynamically through the applications front end using Javascript React.

Setting User Application Permissions

Permissions pertaining to file uploads and downloads can be set and accessed by the administrator , however, Teachers, for example, will have the ability to add permissions to their submission/ content folders through the front-end of the app. These options will trigger FireBase storage authentication read/write privileges for each specific folder and file.

Procedures to Start and Stop the Application

For Administrators:

To start the application, clone the Github repository and run "npm install".

Open one terminal, change directory into the "api" folder and run the command

" py -m app" which will launch the flask server. After that run the command,

"Npm start" which will launch the development server. Note that this must be done in Chrome or else the hex-grid functionality will be non-existent.

For Users:

Login by entering username and password.

Enjoy!

Description of Major Program or Sub-program Modules and Application Flow-Chart

Logging in will trigger a query to the database. This query will contain the “username” stripped from the email and is the key to the user data in the database. Once the User-data has been pulled the clients Flask will communicate with the React front-end, relaying course information(honeycomb to show) and any other notifications for the user.

The user data object will contain the user's name, and most importantly a link to the courses the student is a part of in order to properly display their pages as well as their spot in the navigation.

In the future, we plan that the database will be able to hold quizzes, which would allow a teacher to both A. make a quiz on honeycomb, and B. limit the amount of space used by a specific course in the Database.