Rently

Team 01

Chase Alexander - Team Lead, full-stack

Benjamin McCullough - Github Master, frontend

Lauren Barer - Scrum master, database, cloud, security

Yu Hang Lee - Frontend lead

Chu Cheng Situ - Backend lead

What Is Rently?

A Rental Experience to Suit A Variety of Needs

Our App and Our Customer Base

- What is Rently?
  - Rently is an online rental marketplace
  - Users can rent items
  - Users can list their own items to rent
  - Terms of the rental set by lister
- Who will use Rently?
  - Rently can be used by a variety of customers:
    - Anyone interested in a new hobby
    - Anyone starting a project/repair

Benefits & Pain Points

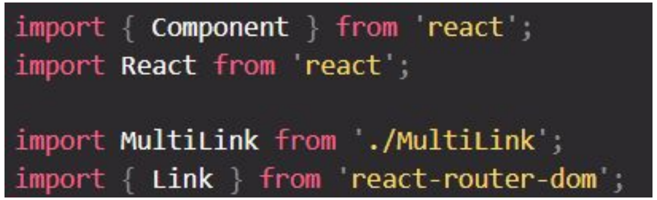
- Benefits to Rently
  - No Commitment-great for projects!
  - Community Based
  - List and make money off unused items
  - No expensive upfront costs
- Pain Points
  - Buying an item only to use it once
  - Overly expensive items

Standing Out from our Competitors

- Ebay and Craigslist
  - While both are user oriented, these websites do not offer rentals
- Amazon
  - While Amazon offers rentals, they are limited to books and digital media
- Rent-A-Center
  - Rent-A-Center does not offer delivery
  - Charges premiums and takes credit info for service

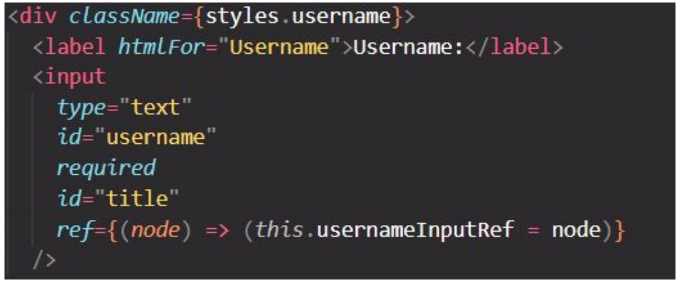
Front End

- ❖ Use React.JS framework
  - ➢ Use Component,React from React library
  - ➢ Use Link, useHistory and withRouter from React-Dom Library

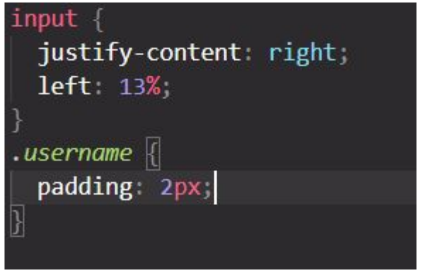


```
import { Component } from 'react';
import React from 'react';

import MultiLink from './MultiLink';
import { Link } from 'react-router-dom';
```

- ❖ For Design the Web Page Application
  - ➢ Custom CSS Styling by setting classname = {styles.name}, and create a CSS file and use the classname to design the web page.



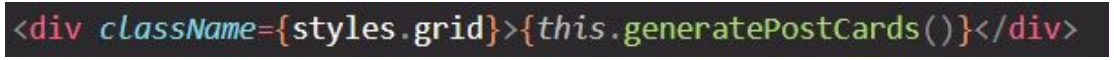
```
<div className={styles.username}>
  <label htmlFor="Username">Username:</label>
  <input
    type="text"
    id="username"
    required
    id="title"
    ref={(node) => (this.usernameInputRef = node)}
  />
```



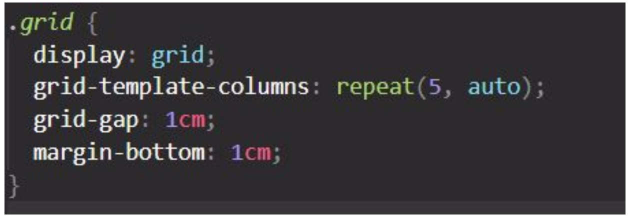
```
input {
  justify-content: right;
  left: 13%;
}
.username {
  padding: 2px;
}
```

Front End Challenges

- CSS Styling, I make the Card component to store the information and while trying to spacing it out with another Card. I am unable to do that by changing the styles. My solution for this is create a div function outside the Card component and in the CSS file, display as grid and grid-gap to make the spacing between the Cards

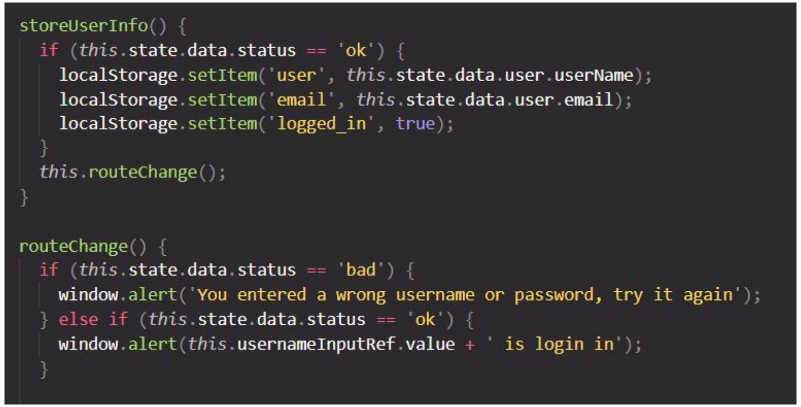


```
<div className={styles.grid}>{this.generatePostCards()}</div>
```



```
.grid {
  display: grid;
  grid-template-columns: repeat(5, auto);
  grid-gap: 1cm;
  margin-bottom: 1cm;
}
```

- Creating a window alert when User login / enter the wrong credentials. I can’t just put the function into the button, I need to put it where the function is checking the credentials.



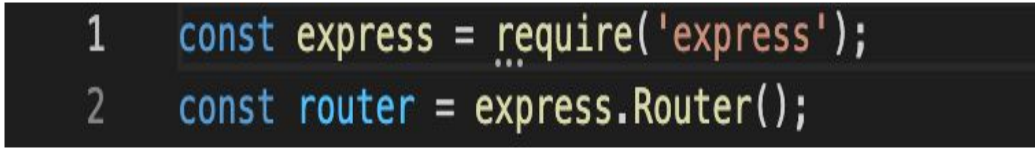
```
storeUserInfo() {
  if (this.state.data.status == 'ok') {
    localStorage.setItem('user', this.state.data.user.userName);
    localStorage.setItem('email', this.state.data.user.email);
    localStorage.setItem('logged_in', true);
  }
  this.routeChange();
}

routeChange() {
  if (this.state.data.status == 'bad') {
    window.alert('You entered a wrong username or password, try it again');
  } else if (this.state.data.status == 'ok') {
    window.alert(this.usernameInputRef.value + ' is login in');
  }
```

Back End

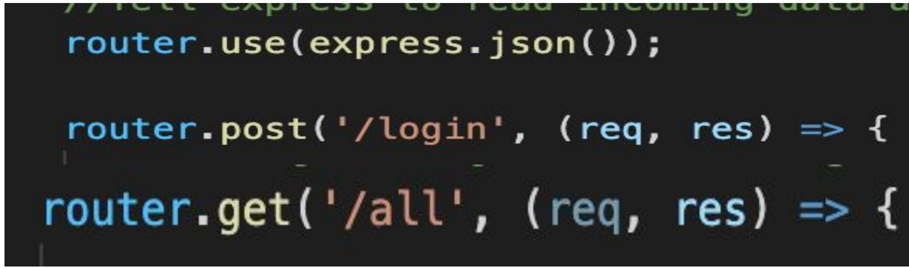
- Use Express.Js
  - Connect to the express router
  - Connect to the express server

- The Router files
  - The router files uses router.get or router.post to handle the request

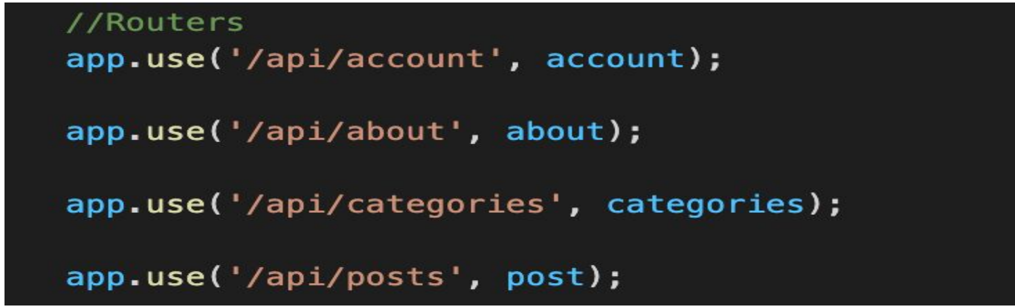
- Server.Js
  - Keep track of the router files by using app.use



```
const express = require('express');
const router = express.Router();
```

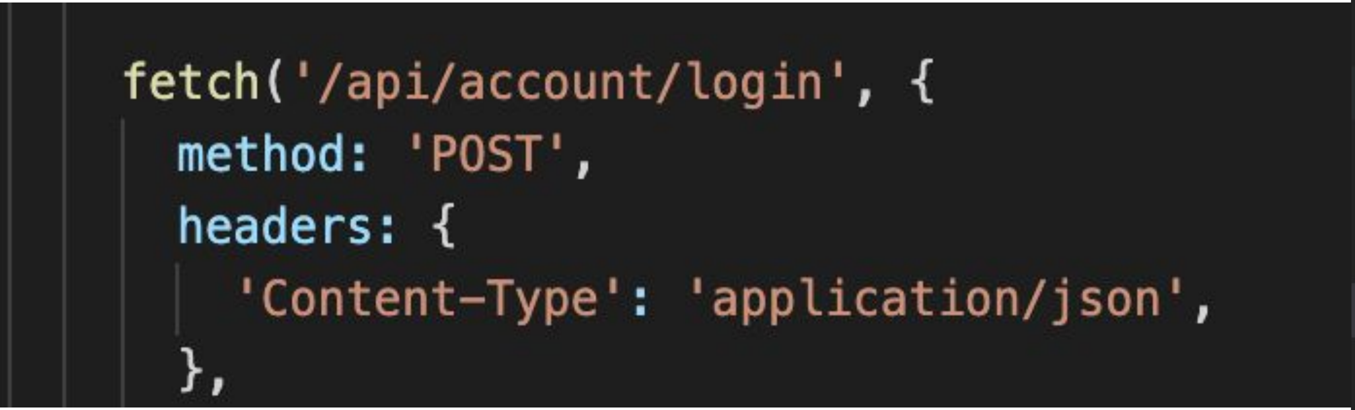


```
router.use(express.json());

router.post('/login', (req, res) => {

router.get('/all', (req, res) => {
```

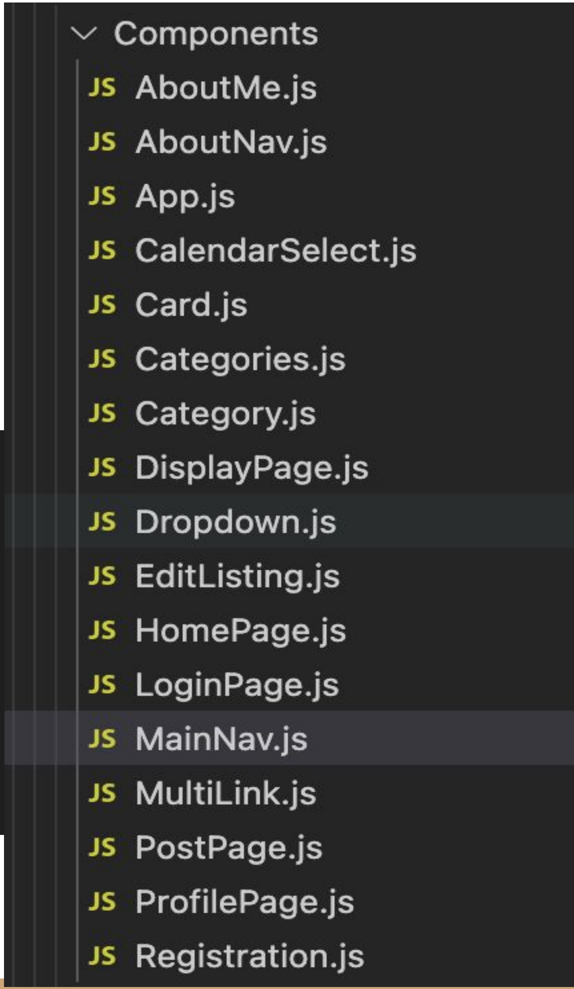


```
//Routers
app.use('/api/account', account);

app.use('/api/about', about);

app.use('/api/categories', categories);

app.use('/api/posts', post);
```

- Components Folder
  - The files in the folder to connect with front-end and back-end.
  - We are using fetch statement to keep track the data from back-end to front-end



```
fetch('/api/account/login', {
  method: 'POST',
  headers: {
    'Content-Type': 'application/json',
  },
```

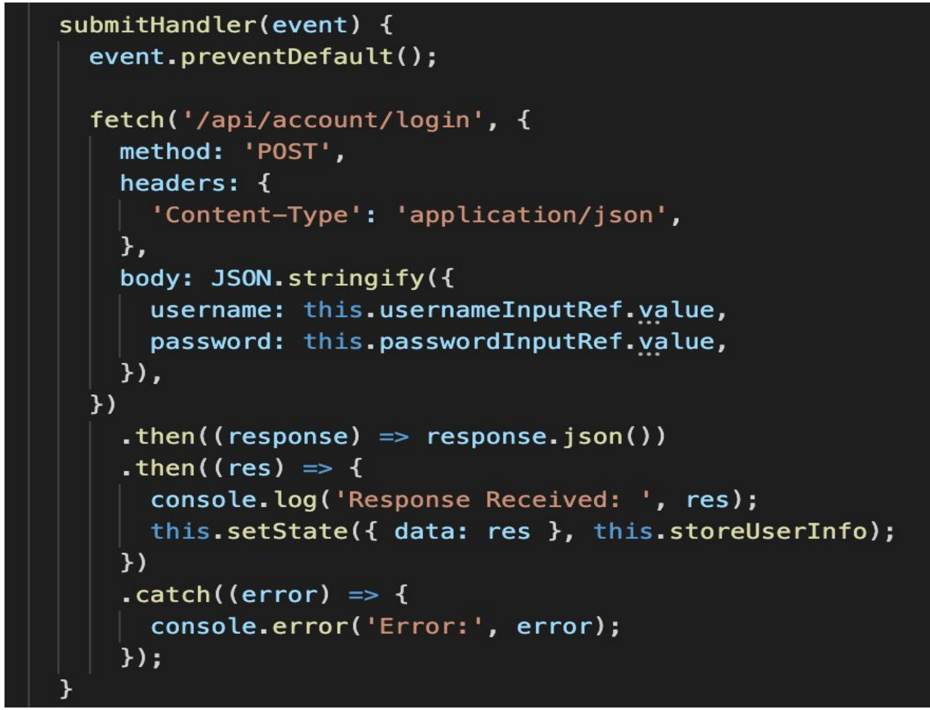


```
∨ Components
  JS AboutMe.js
  JS AboutNav.js
  JS App.js
  JS CalendarSelect.js
  JS Card.js
  JS Categories.js
  JS Category.js
  JS DisplayPage.js
  JS Dropdown.js
  JS EditListing.js
  JS HomePage.js
  JS LoginPage.js
  JS MainNav.js
  JS MultiLink.js
  JS PostPage.js
  JS ProfilePage.js
  JS Registration.js
```

Back End Challenges

- Changing Formatting and method

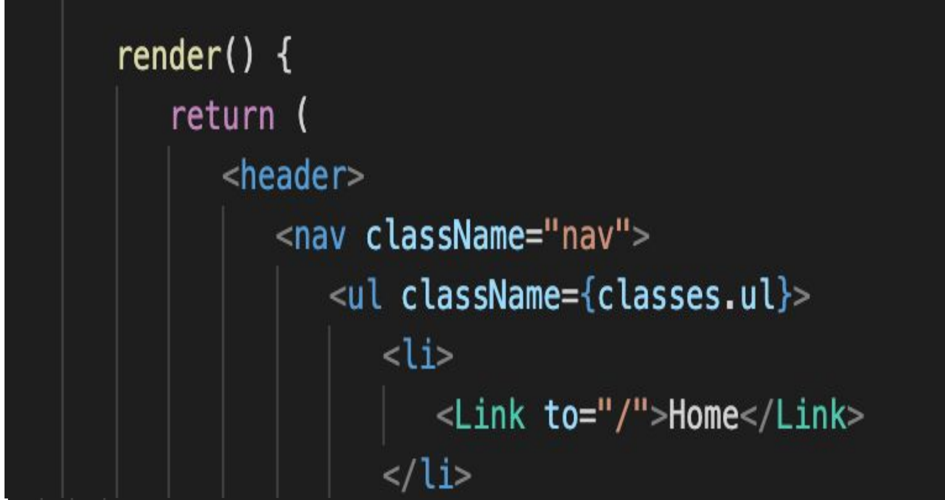
At first, we are planning to put everything in the server.js file. Then, after communication, we change our format. It separate with components folder, route folder and server.js file to be more organized. At that time, I saw it is pretty similar what I do in CSC 317. It messed me up that I using the method of what I do in CSC 317 to do in this project. Afterward, I figured out that we are using the fetch method to do this project and I am not familiar using the fetch method. In order to solve this problem, I am doing a lot of research and asking teammates to help me out with the fetch method.



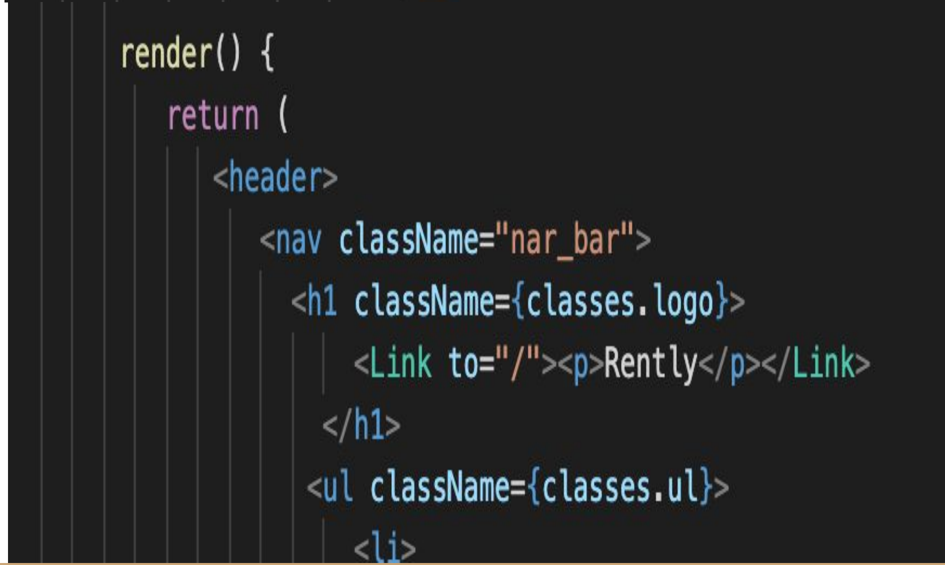
```
submitHandler(event) {
  event.preventDefault();

  fetch('/api/account/login', {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json',
    },
    body: JSON.stringify({
      username: this.usernameInputRef.value,
      password: this.passwordInputRef.value,
    }),
  })
    .then((response) => response.json())
    .then((res) => {
      console.log('Response Received: ', res);
      this.setState({ data: res }, this.storeUserInfo);
    })
    .catch((error) => {
      console.error('Error:', error);
    });
}
```

- Redirect the pages & Logout button

The page redirect and the logout button is working well but after we added some CSS and new stuff, the page redirect and logout button is not working. These two issues is hard to find out for me that I have no idea what file to check on at the beginning. Solving these two problems, I compare every files from previous version of the code to the recent files in Components. Finally, I found out that some files are overwritten. Then, I combined the previous version of the code and new version of the code together. Now, it is working as before.



```
render() {
  return (
    <header>
      <nav className="nav">
        <ul className={classes.ul}>
          <li>
            <Link to="/">Home</Link>
          </li>
```

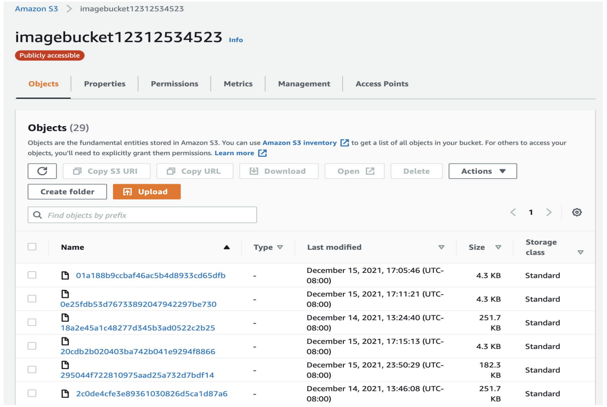


```
render() {
  return (
    <header>
      <nav className="nar_bar">
        <h1 className={classes.logo}>
          <Link to="/"><p>Rently</p></Link>
        </h1>
        <ul className={classes.ul}>
          <li>
```

AWS (Cloud) - RDS

Relational Database Service

- We used RDS in the AWS Cloud to hold our database and be able to run SQL through.
- In our case our cloud was used for data recovery, and a way to store everything.
- We also use AWS for s3 buckets to store images.


Amazon S3 > imagebucket12312534523

imagebucket12312534523 Info

Publicly accessible

Objects | Properties | Permissions | Metrics | Management | Access Points

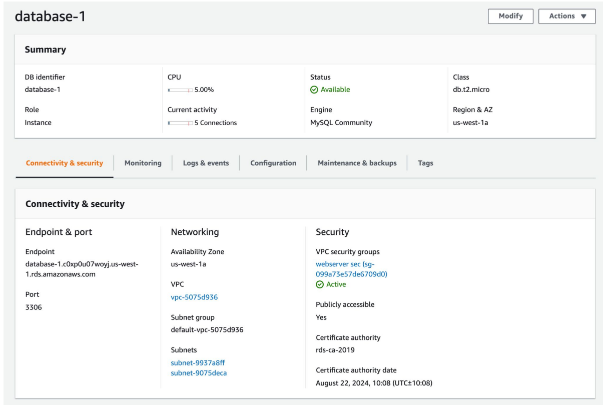
Objects (29)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Copy S3 URI | Copy URL | Download | Open | Delete | Actions | Create folder | Upload

Find objects by prefix

Name	Type	Last modified	Size	Storage class
01a188b9ccbaf46ac5b4d8933cd65dfb	-	December 15, 2021, 17:05:46 (UTC-08:00)	4.3 KB	Standard
0e25fdb53d76733892047942297be730	-	December 15, 2021, 17:11:21 (UTC-08:00)	4.3 KB	Standard
18a2e45a1c48277d345b3ad0522c2b25	-	December 14, 2021, 13:24:40 (UTC-08:00)	251.7 KB	Standard
20cdb2b020403ba742b041e9294f8866	-	December 15, 2021, 17:15:13 (UTC-08:00)	4.3 KB	Standard
295044f722810975aad25a732d7bdf14	-	December 15, 2021, 23:50:29 (UTC-08:00)	182.3 KB	Standard
2c0de4cfe3e89361030826d5ca1d87a6	-	December 14, 2021, 13:46:08 (UTC-08:00)	251.7 KB	Standard


database-1

Modify | Actions

Summary

DB identifier	CPU	Status	Class
database-1	5.00%	Available	db.t2.micro
Role	Current activity	Engine	Region & AZ
Instance	5 Connections	MySQL Community	us-west-1a

Connectivity & security | Monitoring | Logs & events | Configuration | Maintenance & backups | Tags

Connectivity & security

Endpoint & port
Endpoint
database-1.c0xp0u07woyj.us-west-1.rds.amazonaws.com
Port
3306

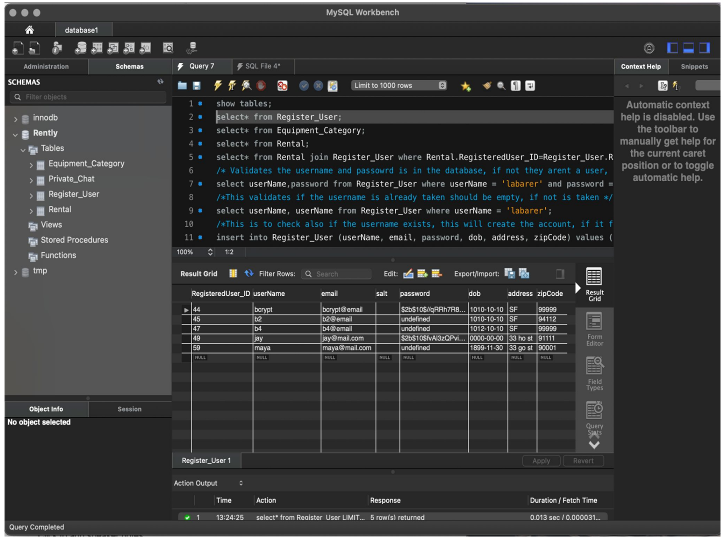
Networking
Availability Zone
us-west-1a
VPC
vpc-5075d936
Subnet group
default-vpc-5075d936
Subnets
subnet-9937a8ff
subnet-9075deca

Security
VPC security groups
webserver sec (sg-099a73e57de6709d0)
Active
Publicly accessible
Yes
Certificate authority
rds-ca-2019
Certificate authority date
August 22, 2024, 10:08 (UTC±10:08)

Database - SQL Workbench

Database - How its set up

- Our database is held in a schema which is called Rently.
- Rently has four tables, Equipment Category, Private Chat, Registered User, and Rental.
- This is where a lot of our information is stored, including account information, passwords (which are encrypted), and the rental post itself.
- The columns hold all of the subcategories of the table.
- When you create an account it saves it in the database.
- Similarly if you create a post it also saves in the database.

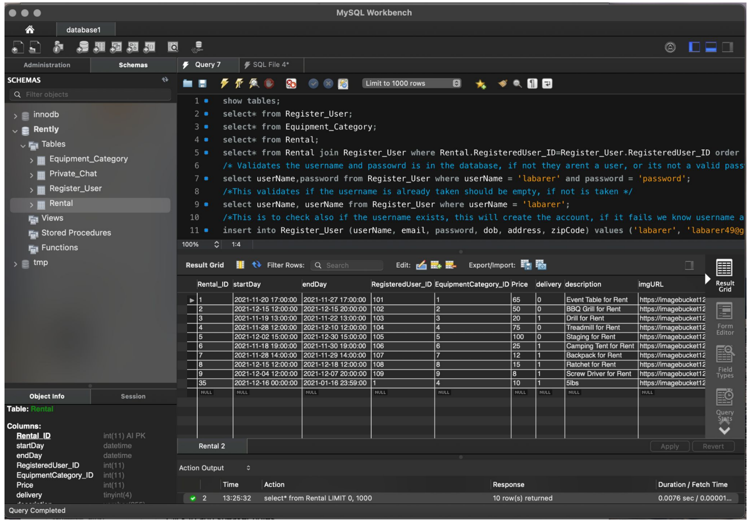


```
show tables;
select* from Register_User;
select* from Equipment_Category;
select* from Rental;
select* from Rental join Register_User where Rental.RegisteredUser_ID=Register_User.R
/* Validates the username and passowrd is in the database, if not they arent a user,
select userName,password from Register_User where userName = 'labarer' and password =
/*This validates if the username is already taken should be empty, if not is taken */
select userName, userName from Register_User where userName = 'labarer';
/*This is to check also if the username exists, this will create the account, if it f
insert into Register_User (userName, email, password, dob, address, zipCode) values (
```

RegisteredUser_ID	userName	email	salt	password	dob	address	zipCode
44	bcrypt	bcrypt@email		$2b$10$//qRRh7R8...	1010-10-10	SF	99999
45	b2	b2@email		undefined	1010-10-10	SF	94112
47	b4	b4@email		undefined	1012-10-10	SF	99999
49	jay	jay@mail.com		$2b$10$fvAl3zQPvi...	0000-00-00	33 ho st	91111
59	maya	maya@mail.com		undefined	1899-11-30	33 go st	90001
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Object Info: No object selected

```
Action Output
    Time      Action                          Response           Duration / Fetch Time
 1  13:24:25  select* from Register_User LIMIT...  5 row(s) returned  0.013 sec / 0.000031...
Query Completed
```



```
show tables;
select* from Register_User;
select* from Equipment_Category;
select* from Rental;
select* from Rental join Register_User where Rental.RegisteredUser_ID=Register_User.RegisteredUser_ID order
/* Validates the username and passowrd is in the database, if not they arent a user, or its not a valid pass
select userName,password from Register_User where userName = 'labarer' and password = 'password';
/*This validates if the username is already taken should be empty, if not is taken */
select userName, userName from Register_User where userName = 'labarer';
/*This is to check also if the username exists, this will create the account, if it fails we know username a
insert into Register_User (userName, email, password, dob, address, zipCode) values ('labarer', 'labarer49@g
```

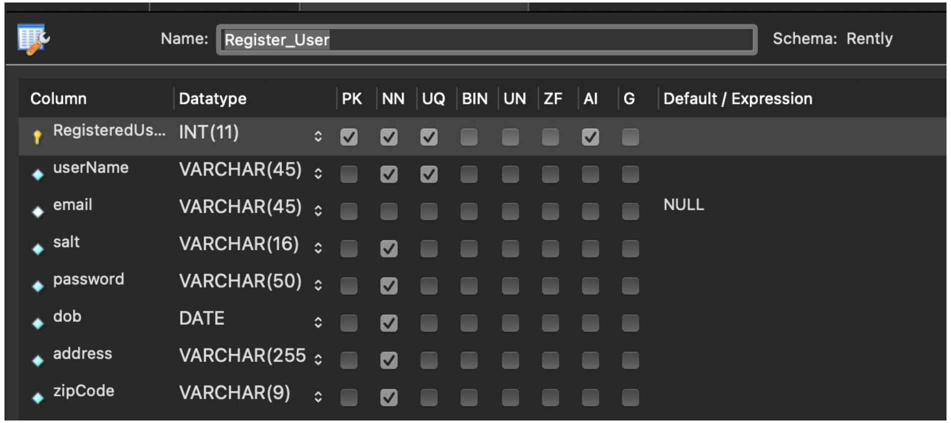
Rental_ID	startDay	endDay	RegisteredUser_ID	EquipmentCategory_ID	Price	delivery	description	imgURL
1	2021-11-20 17:00:00	2021-11-27 17:00:00	101	1	65	0	Event Table for Rent	https://imagebucket12
2	2021-12-15 12:00:00	2021-12-15 20:00:00	102	2	50	0	BBQ Grill for Rent	https://imagebucket12
3	2021-11-19 13:00:00	2021-11-22 13:00:00	103	3	20	1	Drill for Rent	https://imagebucket12
4	2021-11-28 12:00:00	2021-12-10 12:00:00	104	4	75	0	Treadmill for Rent	https://imagebucket12
5	2021-12-02 15:00:00	2021-12-30 15:00:00	105	5	100	0	Staging for Rent	https://imagebucket12
6	2021-11-18 19:00:00	2021-11-30 19:00:00	106	6	25	1	Camping Tent for Rent	https://imagebucket12
7	2021-11-28 14:00:00	2021-11-29 14:00:00	107	7	12	1	Backpack for Rent	https://imagebucket12
8	2021-12-15 12:00:00	2021-12-18 12:00:00	108	8	15	1	Ratchet for Rent	https://imagebucket12
9	2021-12-04 12:00:00	2021-12-07 20:00:00	109	9	8	1	Screw Driver for Rent	https://imagebucket12
35	2021-12-16 00:00:00	2021-01-16 23:59:00	1	4	10	1	5lbs	https://imagebucket12
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Table: Rental

Columns:

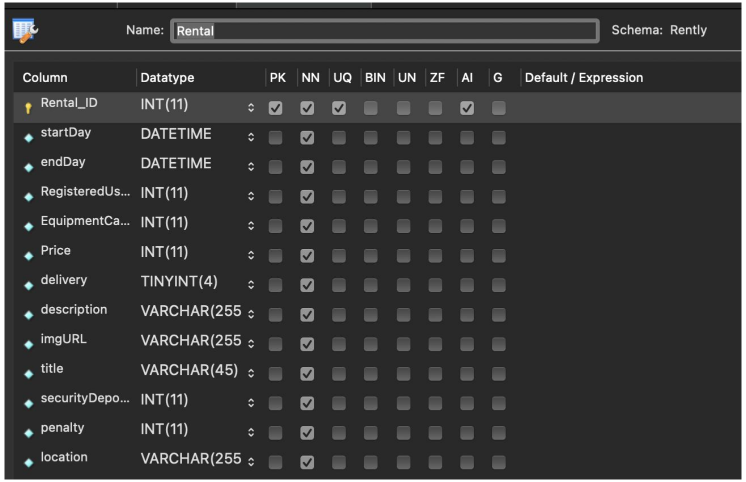
Rental_ID	int(11) AI PK
startDay	datetime
endDay	datetime
RegisteredUser_ID	int(11)
EquipmentCategory_ID	int(11)
Price	int(11)
delivery	tinyint(4)

```
Action Output
    Time      Action                           Response            Duration / Fetch Time
 2  13:25:32  select* from Rental LIMIT 0, 1000  10 row(s) returned  0.0076 sec / 0.00001...
Query Completed
```



Name: Register_User Schema: Rently

Column	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	G	Default / Expression
RegisteredUs...	INT(11)	✓	✓	✓				✓		
userName	VARCHAR(45)		✓	✓						
email	VARCHAR(45)									NULL
salt	VARCHAR(16)		✓							
password	VARCHAR(50)		✓							
dob	DATE		✓							
address	VARCHAR(255		✓							
zipCode	VARCHAR(9)		✓							



Name: Rental Schema: Rently

Column	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	G	Default / Expression
Rental_ID	INT(11)	✓	✓	✓				✓		
startDay	DATETIME		✓							
endDay	DATETIME		✓							
RegisteredUs...	INT(11)		✓							
EquipmentCa...	INT(11)		✓							
Price	INT(11)		✓							
delivery	TINYINT(4)		✓							
description	VARCHAR(255		✓							
imgURL	VARCHAR(255		✓							
title	VARCHAR(45)		✓							
securityDepo...	INT(11)		✓							
penalty	INT(11)		✓							
location	VARCHAR(255		✓							

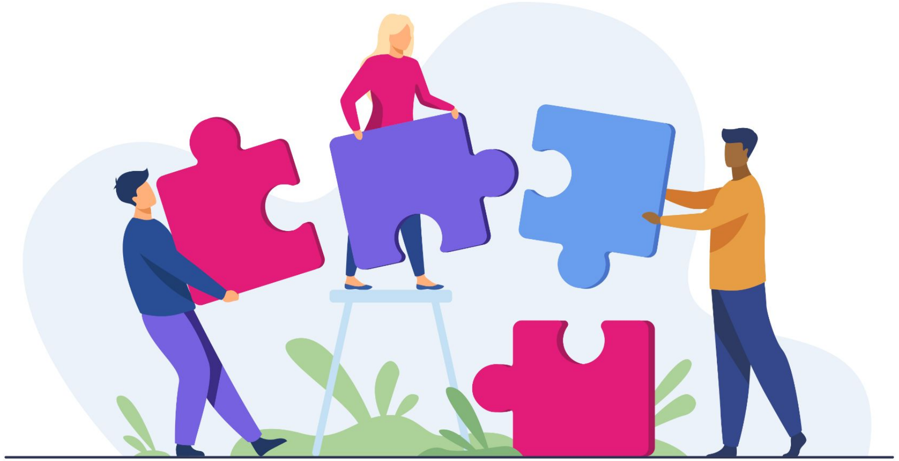
Team Management & Collaboration

Challenges, Solutions, and Successes

Team Collaboration - Challenges

- Time Management & Scheduling
  - Had to change our meeting day to adapt to new schedules
  - Difficult to find times to work together:
    - Work
    - Extracurriculars
    - Family emergencies

- Communication
  - Rough at the start
  - Improved as we progressed
  - Team and became more committ
  - Discord proved to be very useful



Team Collaboration - Successes

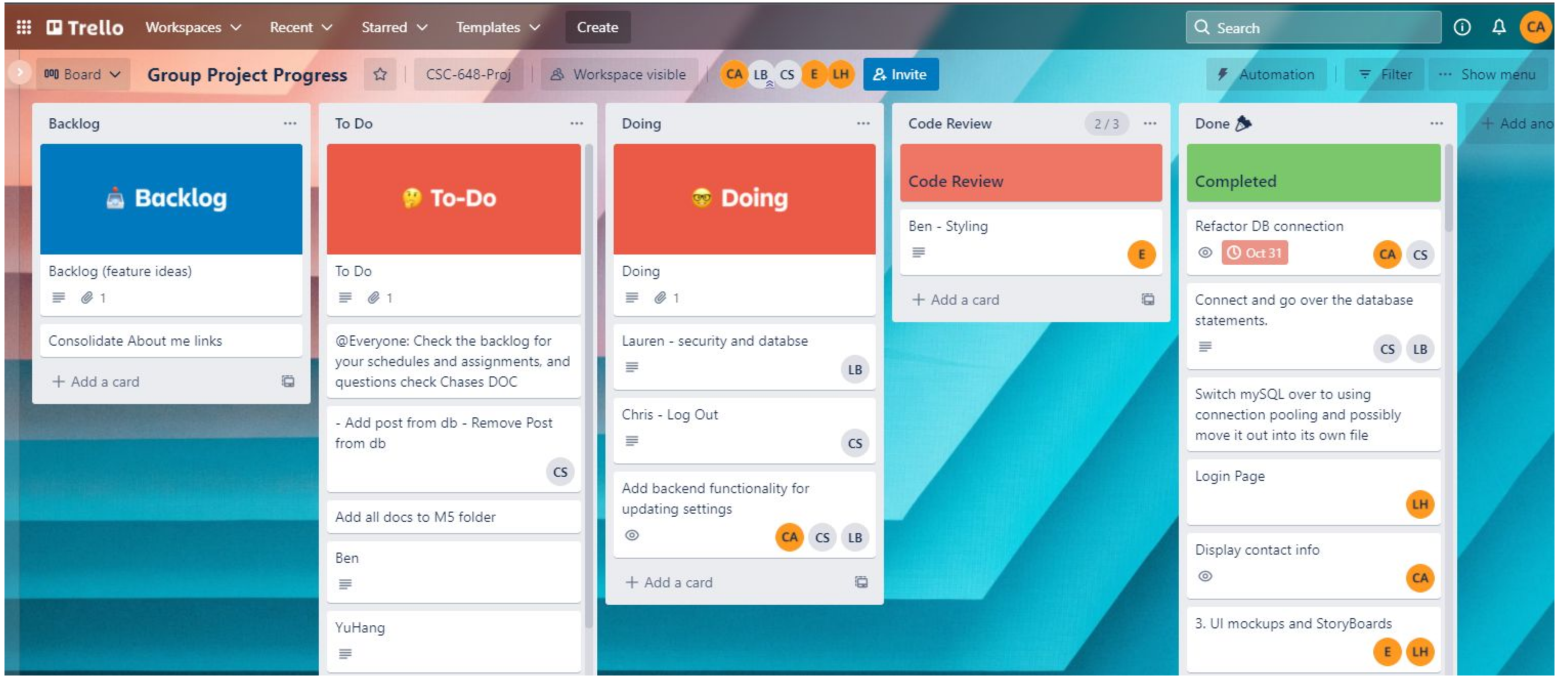
- Teamwork
  - All members did their fair share
  - Worked well together (minimal issues)
- Willingness to help and be helped
  - At some point every team member:
    - Reached out for help in our help channel
    - Provided help to another member in the help channel
    - Met outside of meetings to resolve issues
  - Used pair programming to work through challenges
- Constant team effort towards learning
  - No one had much web development experience
  - We picked it up quickly and learned continuously

Team Management - Challenges

- Load balancing
  - Week to week some areas need more work than others
  - Keeping it fair
  - If some members had a lighter load:
    - Helped out other “teams” (backend/frontend)
    - Worked ahead to get a jump on future work
- Keeping the project on schedule
  - Did this well but it was a challenge
  - Small teams make it more and less difficult
    - Need to put in more work as a team lead pulling your weight + planning
    - Less people makes it easier to manage the team as a whole
  - Helpful practices:
    - Breaking up all milestone work at the start
    - Setting deadlines for portions of work

Team Management - Successes

- Meeting Deadlines
    - Always had our milestones met on time even if it meant staying up all night to get it done
    - Breaking up the work at the start of milestones really helped
- Our team management tools proved very useful
    - Discord
    - Trello
        - Great for checking up on progress without needing to contact members everyday
    - Weekly meeting schedules
        - Kept track of things to discuss from previous meeting
        - Helped us to remember things that came up throughout working days
- Staying Flexible
    - Meeting at various hours with multiple different members
    - Helping out wherever it was needed



Backlog

Backlog

Backlog (feature ideas)

1

Consolidate About me links

+ Add a card

To Do

To-Do

To Do

1

@Everyone: Check the backlog for your schedules and assignments, and questions check Chases DOC

- Add post from db - Remove Post from db

CS

Add all docs to M5 folder

Ben

YuHang

Doing

Doing

Doing

1

Lauren - security and databse

LB

Chris - Log Out

CS

Add backend functionality for updating settings

CA CS LB

+ Add a card

Code Review 2 / 3

Code Review

Ben - Styling

E

+ Add a card

Done

Completed

Refactor DB connection

Oct 31

CA CS

Connect and go over the database statements.

CS LB

Switch mySQL over to using connection pooling and possibly move it out into its own file

Login Page

LH

Display contact info

CA

3. UI mockups and StoryBoards

E LH