

Smart Party Financial

Project Design

CS 157A by Team 31

Christian Castro

Cuong “Calvin” Nguyen

Pranika Bedi

Professor Mike Wu

December 11, 2019

Project Overview

Problem Statement

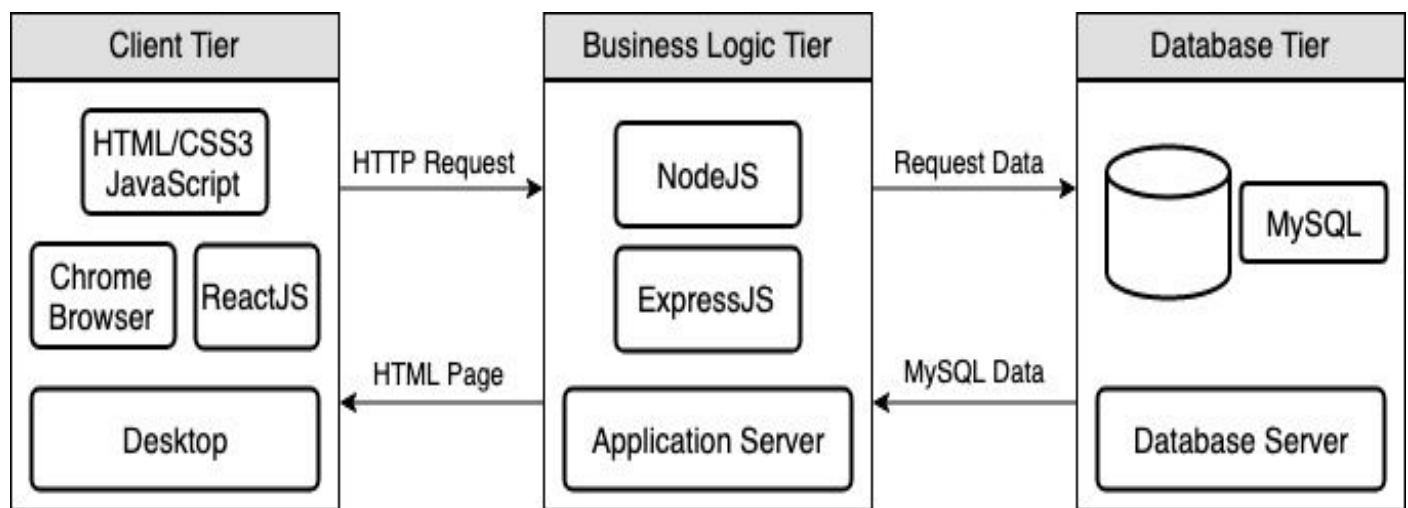
When you're on a trip with a group of friends, Person A spends \$95 on food, Person B spends about \$50 on gas, and Person C spends \$100 on amenities. In this given situation, it will be significantly time-consuming to figure out how much one should pay back to another. Some groups use Excel, but they have to create their own formula, which might result in mistakes and take even more time to fix and re-calculate again. Additionally, there is a possibility that they may request the wrong amount of money. This can lead to problems with trust issues, misunderstandings, and uneven money distribution. As a result, we want to create an application that will not only solve these issues but also provide everything needed to manage a trip on one platform.

Stakeholders: This app focuses on any travel group

Project Description:

We want to develop a database application that will serve as a platform for various groups to manage their events, event members, and finances involved with the event. The app will allow users to create different parties/trips/planners, invite other users of the app to be apart of the event, and enter the amount of money that each person has contributed to the event. Finally, the app will assist in calculating the total amount spent on the event, and then generate the amount of money that one person in the group may owe another. This will in effect create a way to fairly distribute the expenses used for the event. This application will also have a user-friendly interface that will allow users to easily take advantage of these properties.

System Environment



HW/SW: Desktop Web Browser, Mac, Windows, Linux

Overview:

The architecture of our application is based on a typical MVC model. Our Client tier (View) will be written in Javascript, HTML, and CSS, using ReactJS as the framework. This level of the architecture is what the user will interact with to access the features of our application. The Business Logic Tier (Controller) will be written using NodeJs and ExpressJS, and this tier represents the Application Server that will act as a bridge of communication for the Client Tier and Database Tier. This tier will serve HTML pages to the user's device and accept HTTP requests from the user and follow with the appropriate response. Our Database Tier (Model) will be hosting MySQL as our application's Relational Database Management System. This is where we will store all of the crucial data our application needs to function.

Functional Requirements

- **Any user is able to login/register/logout an account**

- Goal:
 - A new user can register an account with our app or log in with their email.
- Functional Processes:
 - Input: Email/Password
 - Output: Redirect to the user dashboard/homepage
- Prerequisite:
 - User should be able to make a new account with an active email.
 - User should be able to log in with correct email and password combination
- Post-Condition:
 - The user will be redirected to the dashboard/homepage where they will be able to see options to view their profile, groups, and events.
- Exceptions:
 - If the user inputs the wrong email/password combination, an error message will be displayed.

- **User is able to create new events**

- Goal:
 - Registered user clicks a “Create Event” button to create a new event/trip
- Functional Processes:
 - Input: User will log in to the web app, and click on the “Create Event” button. Fill in the form with the event: name, location, start date, and end date.
 - Output: The page will be redirected to the new event page where they can start adding users and money they spend on the trip
 - `Const express = require('express');`
- Prerequisite:
 - User is logged into their account
 - User completely fills out the form for event information
- Post-Condition:

- Event will be created and added to the database
 - User will be redirected to result page for new event
- Exceptions:
 - If the user does not completely fill out Event information form an error message will be displayed
- **User is able to create new parties**
 - Goal:
 - User can click on “Create Party” and create new parties.
 - Functional Processes:
 - Input: User logs into the application, clicks on the button to create a party, and inputs the name of the party.
 - Output: User will be redirected to a page where they can start inviting members to the party.
 - Prerequisites:
 - User is logged into their account
 - User completely fills out the party form: name.
 - Post-Condition:
 - New party is added to the database.
 - Exceptions:
 - If the user does not completely fill out Party information form an error will be displayed.
 - If the user is not logged in to their account they will not have access to the “Create Party” button
- **User is able to invite members to a party**
 - Goal:
 - User can click on the “+” to add an existing user to a party
 - Functional Processes:
 - Input: User clicks the “+” button and inputs the email of the user to be invited.
 - Output: The user that was invited will now be seen as a member within the party.

- Prerequisites:
 - User is logged into their account
 - User inputs an email belonging to an existing user
- Post-Condition:
 - New member is added to the database.
- Exceptions:
 - If the user tries to invite a member that does not exist an error message will be thrown
- **User is able to join parties and events they know the names of**
 - Goal:
 - User will be able to join parties and events by clicking “Join Party” or “Join Event” buttons.
 - Functional Processes:
 - Input: User is logged into the application, and clicks “Join Party” or “Join Event” and inputs the exact name of the party or event they wish to join.
 - Output: User will be redirected to see they have been added to an event or party.
 - Prerequisites:
 - User is logged into their account.
 - User inputs the correct name of event and party.
 - Post-condition:
 - User is added to an event or party and is able to view it on the home page.
 - Exceptions:
 - If the user inputs an incorrect event or party name (spelling, case sensitive, etc), they will not be able to join the party or event.
- **User is able to add and edit categories of what they paid for.**
 - Goal:
 - User can add and edit as many categories of what they paid for in an event.
 - Functional Processes:

- Input: User clicks on the “+” button on the event page. User will be prompted to complete a form, involving category name and amount of money. User can also click on “edit” to edit the information from the same form.
 - Output: User will see an updated table to various categories, distinguished by name of user, category name, and amount of money.
- Prerequisite:
 - User is logged into their account.
 - User completely fills out the form.
- Post-condition:
 - User views updated category table.
- Exceptions:
 - If the user does not fill out the form or inputs values of incorrect types, an error message will be thrown.
- **User is able to see a final calculation that is fairly distributed evenly among the members.**
 - Goal:
 - User enters the category of what they pay for, then the app will calculate and display the right amount for each person in the events.
 - Functional Processes:
 - Input: User clicks on “Calculate Final Price” button after categories have been added to the table.
 - Output: User will be redirected to the results page where the payee email, recipient email, and amount of money the payee owes the recipient will be displayed and updated.
 - Prerequisite:
 - User is already a member of an event.
 - Post-Condition:
 - New payment information will be applied to the event and database will be updated.
 - Exceptions:

- Price will be calculated if there are 2+ members within the event.
- **User is able to remove other members in the party**
 - Goal:
 - An existing user will be removed from a specific party.
 - Functional Processes:
 - Input: User will be able to see a list of users in a party when they are viewing a party. User can click on the “-” button next to the email of the user they wish to remove.
 - Output: The user will see that that the email of the deleted member no longer exists within the party.
 - Prerequisite:
 - User is logged into their account.
 - User is viewing a party they created.
 - Post-condition:
 - Party member changes will be applied, and the database will be updated.
- **User is able to see a list of events and groups they’re involved in.**
 - Goal:
 - User will be able to see all parties and events they are members of on their home page.
 - Functional Processes:
 - Input: User clicks on “Welcome user” button or “Smart Party Financial” to be redirected to the homepage where they can see “My Events” and “My Parties”.
 - Output: A list of party and event cards will be presented to the user
 - Prerequisite:
 - There are a list of events displaying on the UI
 - Post-Condition:
 - The list will be sorted based on order of creation.
 - Exceptions:
 - If the user has no events or parties, there will be nothing to display under “My Events” and “My Parties”.

Non-functional Requirements

Implementation

- Develop a responsive web application in HTML/CSS/JavaScript, and ReactJS library for real-time update
- Host and language the database/application server on Google Cloud Platform

Usability

- The application shall be displayed in English.

Reliability

- Support on major internet browsers such as Google Chrome.

Performance

- Respond to the user's action and query data within 5 seconds.

Security

- Use JWT to authenticate and check for the user's identity to protect users' privacy

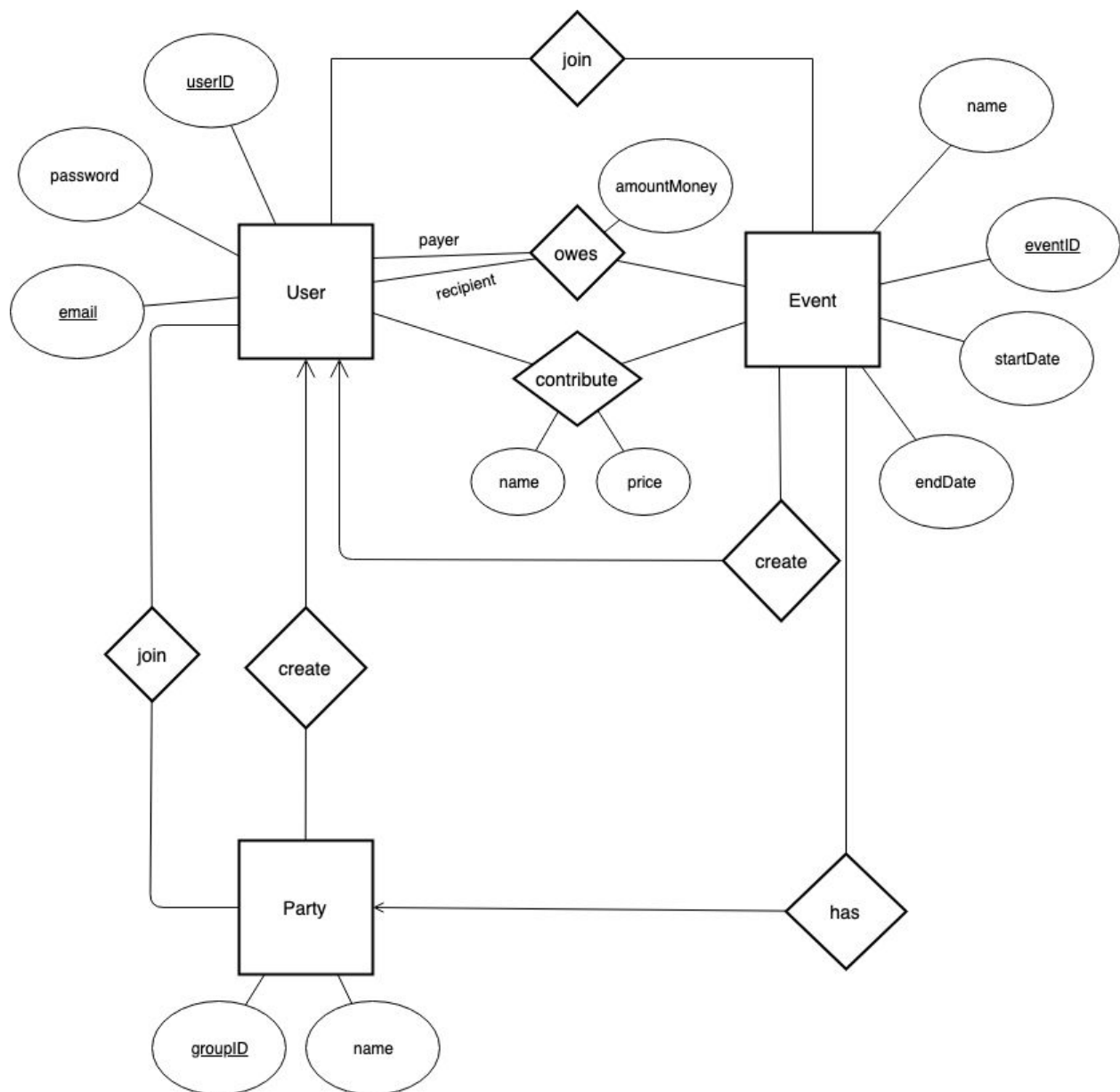
Scalability

- Multiple users will be able to access the application's database without any latency issues
- The application will be able to serve multiple users at once

Packaging

- The application will be available to use on these platforms without any additional setup:
 - Windows and Mac

Project Database Model



Entity Sets

1. User

The user has attributes password and email. This stands as the user's login information for the application.

2. Party

Groups is a weak entity set which means primary key must exist in order for it to be distinguished from another group. Each group has a unique name.

3. Events

Events is also a weak entity set and has a primary key, name. It also has other attributes, location, email, and endDate.

Relations between Entity Sets

1. User_Owes_User

The user has the option to request/charge for a certain amount of money. The application will save this request in the person's profile. However, the application does not support transactions.

2. User_Form_Party

The user can form a party with a name given by the user. The users in the party can add other existing members, as well as add events that only party members can view.

3. User_Create_Event

The user has the ability to create events that pertain to various locations, date range, name, ID, and email.

4. User_Join_Event

The user can join different events. This occurs when the owner of an event creates an event and invites other users to join the event. The user has the option to accept or deny this invitation.

5. User_Join_Party

A user can be a member of multiple party. The user can view their party in their dashboard.

Parties can be distinguished from one another by a unique ID and name.

6. Party_Has_Event

After a party is formed, the users have the option to create an event within the party. This event is accessed by all members of the party and can proceed to creating categories within the event.

7. User_Contribute_Event

User will record their financial contributions to an event in order for our application to determine how much money the user owes or is owed by other users.

Database Tables

Entity: User

	userID	email	password
►	1	calvin	\$2b\$10\$5hqEeX0xhOvtXrdbt7l41etd2Om8CTU3...
	2	pranika	\$2b\$10\$3jast2Qw8KYxnfaevtkzIuFp1nNTL6M1....
	3	christian	\$2b\$10\$Bc7MuH.2B085wBJrUk0Xze50Jy5xEvI...
	4	chris2	\$2b\$10\$6RtKKd58KYP2UKC3ucwfE.SYxBE0YiTo...
	5	mikeisawesome	\$2b\$10\$NepwGAHTc7HjPIIjg26sGe1khMrzUJNG...
	6	pranika1	\$2b\$10\$ze8fJVf44VENB2534A30uuHNzW5/okz...
	7	mike	\$2b\$10\$rXNwY.c8NB7dZDXCf1kB6eUGLde5SrTj...
	8	user@email.com	\$2b\$10\$ZEvl7IaZUxxQH/5Nhw7nZOO2O94QR....
	9	Joe	\$2b\$10\$tcz4d7HIuJfWqLBzKLMKuHYPfqKgBrrQ...
	10	Jon	\$2b\$10\$GRV0r77X6xGORuAmo9nUHOjFLENIhXi...
	11	Steve Jobs	\$2b\$10\$Ifgc3rz7M.RLVc6y6EmnIuvGKwdEX9BY...
	12	elon@email.com	\$2b\$10\$vmQ8Ew391qjPiyZn1JPMIe3JRXUTe82...
	13	bob@email.com	\$2b\$10\$BbUrmStNDcPfkDJZ/3gvfuc/LMgZ.KBJ7...
	14	pranika@email....	\$2b\$10\$K2TO54pfMig1cZIsBbEA9.TdqTQ.Mi59c...
	15	calvin@email.com	\$2b\$10\$In0uT8d95o3PFYWA/8HYHOGm1U795s...

Entity: Party

	partyID	name	totalMember
►	1	cs157a	1
	2	Party1	1
	3	Holiday Party	1
	4	Study Party	1
	5	LA	1
	6	Test Party	1
	7	LA	1
	8	Calvin Party	1
	9	Mike Party	1
	10	My Party	1
	11	SJSU Party	1
	12	December P...	1
	13	Graduation ...	1
	14	Tesla Party	1
	15	Engineering ...	1

Entity: Event

	eventID	name	startDate	endDate	location
▶	1	Winter 2019	12/17/2019	01/17/2020	Vietnam
	2	Summer 2020	Feb 4, 2019	09/12/2019	Vietnam
	3	Disneyland	12/10/19	12/25/19	Los Angeles
	4	Holiday Party	12/24/2019	12/24/2019	Mike's House
	5	Party	12/24/2019	12/24/2019	Los Angeles
	6	Library Study Party	12/10/19	12/17/19	SJSU Library
	7	Calvin's Event	12/17/2019	01/17/2020	Vietnam
	8	Vegas	12/15/19	12/19/19	Vegas
	9	Pranika Event	12/15/19	12/19/19	School
	10	Pranika Event	12/15/19	12/19/19	Vegas
	11	Christians Party	12/10/19	12/16/19	Christians H...
	12	Library Event 1	12/17/2019	09/12/2019	Vietnam
	13	Christian Party	12/17/2019	01/17/2020	Vegas
	14	Mike Event	Feb 4, 2019	09/12/2019	Mike House
	15	Mike Event Party	12/17/2019	01/17/2020	Vegas
	16	Tesla Release Party	1/1/2020	1/2/2020	Tesla in Fre...

Relation: User_Form_Party

	userID	partyID
▶	1	1
	4	2
	3	3
	3	4
	1	5
	3	6
	6	7
	1	8
	7	9
	8	10
	15	11
	15	12
	15	13
	12	14
	12	15

Relation:User_Join_Party

	userID	partyID
▶	3	1
	4	2
	1	1
	3	3
	5	3
	3	4
	1	4
	2	4
	5	4
	1	5
	3	6
	6	7
	1	8
	2	8
	3	8
	5	8
	7	9

Relation: User_Create_Event

	userID	eventID
►	1	1
	3	4
	1	7
	6	8
	2	9
	3	11
	7	14
	12	18
	3	19
	3	20
	3	21
	2	22
	2	23
	2	24
	2	25

Relation: User_Join_Event

	userID	eventID
►	1	1
	2	1
	3	1
	3	4
	1	7
	6	8
	2	9
	3	9
	3	11
	7	14
	7	1
	8	15
	12	18
	3	19
	3	20
	3	21
	~	~

Relation: Party_Has_Event

	eventID	partyID
▶	2	1
	3	1
	5	3
	6	4
	10	8
	12	4
	13	8
	15	9
	16	14
	17	14
	26	4
	27	4
	28	4
	29	4
	30	4

	eventID	partyID
▶	2	1
	3	1
	5	3
	6	4
	10	8
	12	4
	13	8
	15	9
	16	14
	17	14
	26	4
	27	4
	28	4
	29	4
	30	4

Relation: User_Contribute_Event

	eventID	userID	category	amount
►	1	1	food	55
	1	2	rent car gain	120
	1	3	gas	200
	9	2	Food	20
	9	3	Beer	50
	10	2	food	15
	10	3	Cups	10
	10	5	beer	5
	12	2	Food	10
	12	1	gas	80
	15	7	food	50
	15	3	gas	20
	15	8	Beer	100
	29	2	Drinks	10
	1	2	Games	60
	23	2	Shoe Money	20000
	23	2	Chairs	500

Relation: User_Owes_User

Screenshots - Demo of the app

Landing Page

<> Smart Group Financial

Split the bill easier for everyone

Easy to manage all your partys' trip financial

REGISTER

LOGIN

Register Page

<> Smart Group Financial

Register

Email

user@email.com

Password

.....

[Already has an account? Login](#)

REGISTER

Login Page

<> Smart Group Financial

Login

Email

user@email.com

Password

.....

Don't have an account? [Register](#)

LOGIN

Home Page (Redirected from Register/Login)

<> Smart Party Financial

Welcome user@email.com

[Join Event](#)

[Join Party](#)

[Create Party](#)

[Create Event](#)

[Logout](#)

My Events

My Parties

Join Existing Event (Click 'Join Event' button)

<> Smart Party Financial

Welcome user@email.com Join Event Join Party Create Party Create Event Logout

Mike Event Party

JOIN

<> Smart Party Financial

Welcome user@email.com Join Event Join Party Create Party Create Event Logout

My Events

Mike Event Party

Start Date: 12/17/2019
End Date: 01/17/2020
Location: Vegas

VIEW

My Parties

Name: My Party

Event Page for the event user joined (Click on 'View' on Event card)

Event ID: 15

Category Table				
<input type="checkbox"/>	ID	Name	Category	Amount Money
<input type="checkbox"/>	1	mike	food	50
<input type="checkbox"/>	2	christian	gas	20
Rows per page: 10 1-2 of 2 < >				



EDIT

CALCULATE FINAL PRICE

User Contributes to Event (Click '+' button)

Event ID: 15

Category Table				
<input type="checkbox"/>	ID	Name	Category	Amount Money
<input type="checkbox"/>	1	mike	food	50
<input type="checkbox"/>	2	christian	gas	20
Rows per page: 10 1-2 of 2 < >				



EDIT

CALCULATE FINAL PRICE

Add New Category

Category

Beer

Amount of Money

35

ADD NEW CATEGORY

<> Smart Party Financial

Event ID: 15

Welcome user@email.com Join Event Join Party Create Party Create Event Logout

Category Table				
<input type="checkbox"/>	ID	Name	Category	Amount Money
<input type="checkbox"/>	1	mike	food	50
<input type="checkbox"/>	2	christian	gas	20
<input type="checkbox"/>	3	user@email.com	Beer	35

Rows per page: 10 1-3 of 3 < >



EDIT

CALCULATE FINAL PRICE

Edit contribution (Click 'Edit' button)

<> Smart Party Financial

Event ID: 15

Welcome user@email.com Join Event Join Party Create Party Create Event Logout

Category Table				
<input type="checkbox"/>	ID	Name	Category	Amount Money
<input type="checkbox"/>	1	mike	food	50
<input type="checkbox"/>	2	christian	gas	20
<input type="checkbox"/>	3	user@email.com	Beer	35

Rows per page: 10 1-3 of 3 < >

+

EDIT

CALCULATE FINAL PRICE

Edit Category

Category ID






Beer

Amount of Money

100

EDIT CATEGORY

Event ID: 15






Category Table					    				
<input type="checkbox"/>	ID	Name	Category	Amount Money					
<input type="checkbox"/>	1	mike	food	60					
<input type="checkbox"/>	2	christian	gas	20					
<input type="checkbox"/>	3	user@email.com	Beer	100					
					Rows per page: 10 ▾ 1-3 of 3 < >				



EDIT

CALCULATE FINAL PRICE

Calculate distribution of money (Click 'Calculate Final Price' button)

Results					    				
<input type="checkbox"/>	Number	Payee Email	Recipient Email	Amount Money					
<input type="checkbox"/>	1	christian	user@email.com	40					
					Rows per page: 10 ▾ 1-1 of 1 < >				

BACK

Create Party Page (Click on 'Create Party' button on the home page)

<> Smart Party Financial

Welcome user@email.comJoin EventJoin PartyCreate PartyCreate EventLogout

My Party

CREATE

<> Smart Party Financial

Welcome user@email.comJoin EventJoin PartyCreate PartyCreate EventLogout

My Events

My Parties

Name: My Party

VIEW

Party Page (Click on 'View' on party card)

Events



Party Members



user@email.com



Add Event for Party (Click '+' button on Event header)

New Event

Name of Event

Spring 2020

Start Date

12/17/2019

End Date

11/20/2020

Location

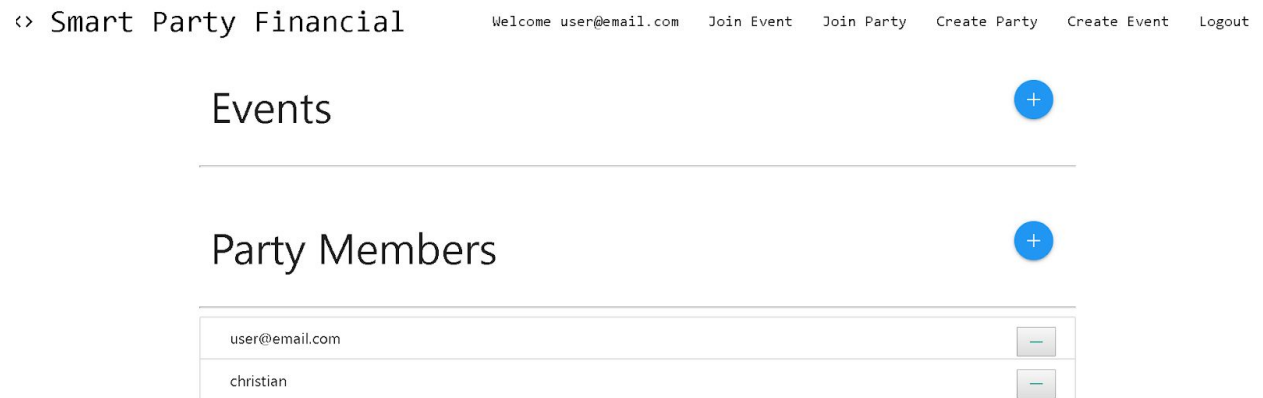
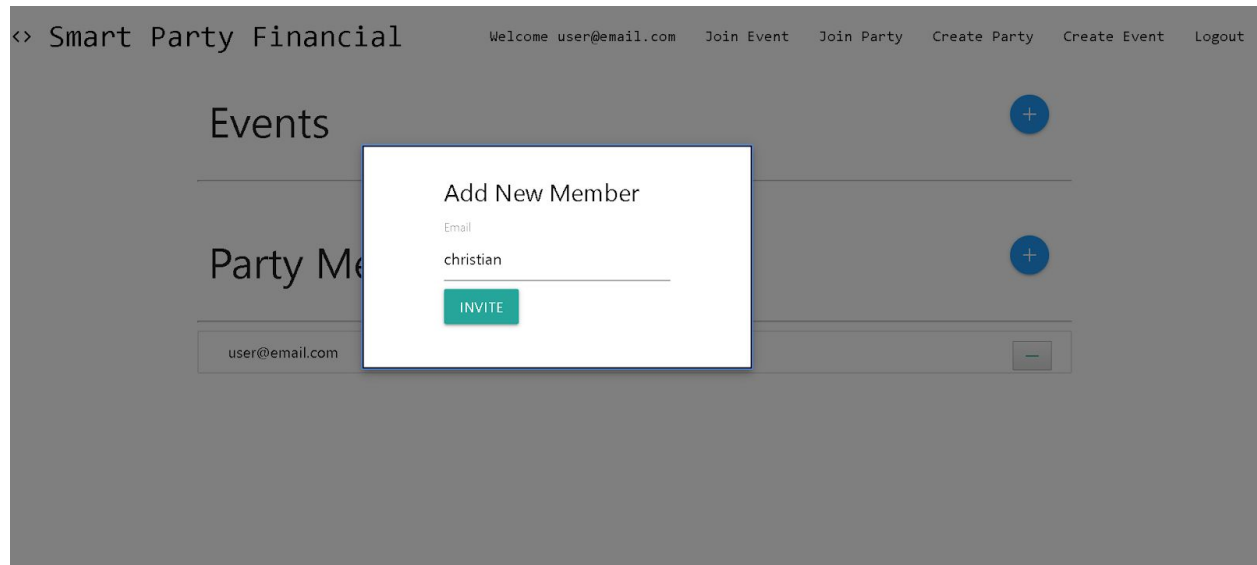
SJSU

CREATE

<div>Library Study Party</div> <div>Start Date: 12/10/19 End Date: 12/17/19</div> <div>VIEW</div>	<div>Library Event 1</div> <div>Start Date: 12/17/2019 End Date: 09/12/2019</div> <div>VIEW</div>	<div>Study Event</div> <div>Start Date: 12/8/19 End Date: 12/8/19</div> <div>VIEW</div>	<div>End of Semester</div> <div>Start Date: 12/16/19 End Date: 12/16/19</div> <div>VIEW</div>
<div>Study Event 2</div> <div>Start Date: 1/25/20 End Date: 1/25/20</div> <div>VIEW</div>	<div>Beginning of Winter Semester</div> <div>Start Date: 1/4/20 End Date: 1/4/20</div> <div>VIEW</div>	<div>End of Winter Semester</div> <div>Start Date: 1/21/20 End Date: 1/21/20</div> <div>VIEW</div>	<div>Spring 2020</div> <div>Start Date: 12/17/2019 End Date: 11/20/2020</div> <div>VIEW</div>

Result after adding Spring 2020 into event list of the party

Add Existing User to Party (Click '+' button on Party Member Header)



Implementation

The project is splitted into 2 folders: Client and frontend. After you clone the project, you need to follow the instruction in README on Github.

First you need to run ``npm run i`` to install everything.

Then, you will run ``npm run server`` in 1 terminal, and then run ``npm run client`` separately in another terminal.

Lesson Learned

Cuong (Calvin) Nguyen

I learned to integrate the project from the bakend and the frontend. Especially, I learned to work with MySQL, and React.js. At the same time, leading the team requires both knowledge of everything and time management. This project has taught me valuable lessons in working with connect Node.js with MySQL

Pranika

This project was a learning experience in many aspects. This project was my first big team project, and it taught me a lot about time management and ensuring I made time for weekly meetings. Additionally, I was exposed to ReactJS for the first time, allowing me to deepen my skills in CSS, HTML, and JavaScript. It was also very interesting to see how this three-tier application tied with the material we were learning in lecture, and it allowed me to understand the concepts more clearly.

Christian Castro

This was one of the first projects where I was heavily involved in front end development. I learned a lot of React, and I can say that I am now comfortable enough with the framework that I am looking forward to using it in my future projects. I had previous experience working with MySQL, but this project and CS 157A in general, taught me how to utilize query statements in a more efficient manner. I also had first hand experience working with ExpressJS and Axios for the first time and I have a better understanding of how to integrate the Frontend and Backend of a working application.

Future Improvement

There were many features we discussed implementing into our project at the beginning of the semester, but due to time constraints we decided to focus on the minimal marketable features. To

improve further on the project some functional features we would add are: the ability to search for a party/event, the ability to delete an event, allow the user to leave an event, and create a function to track the activity history of a user and a party. We would also like to create a restriction on modifying the party because as our application is now, any user can add event/members and remove members from a party. Ideally, we would only enable the user that created the party to perform these tasks. In addition, we would have also liked to spend more time polishing our UI. At the moment it is very minimalistic, but given more time we would have made it more appealing for users to look at.