GoOut!

GoOut With Us

Jeffrey Hernandez
Sina Karachiani
Steven Xu
Jessica Zhu

Alternative Approaches to Assembly Administration: The Science of Cartography Redefined as Community Integration via Collaborative Creation in the Ongoing and Evolving Paradigm of the Post-Network Media Environment— What Does It Mean To You? (featuring GoOut)

App Features

MAP

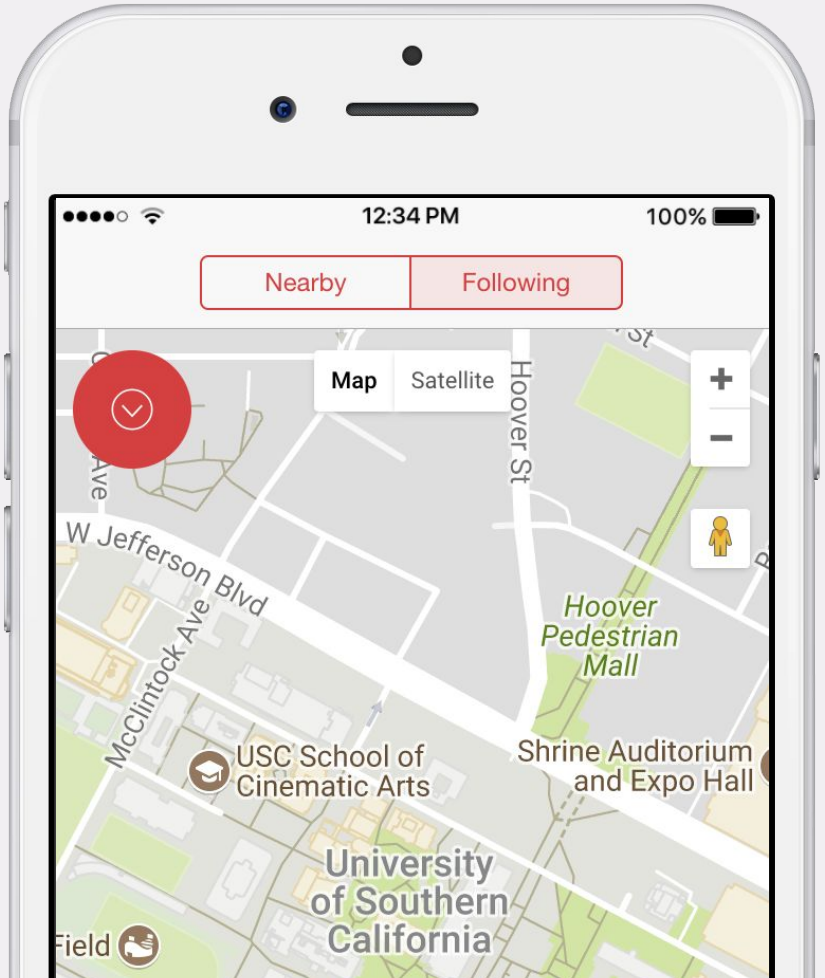
Find ongoing or upcoming events using Google Maps integration

Event Planning

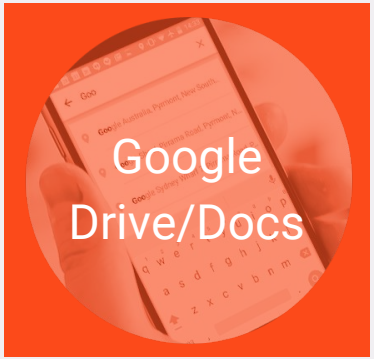
Create and follow events so you can always have a GoOut time

HYBRID

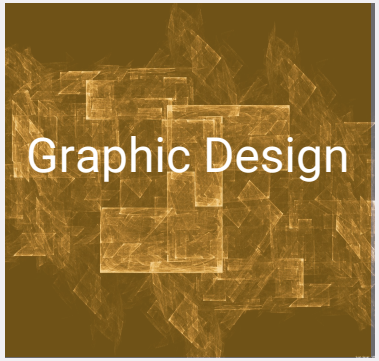
The same app runs on both Apple and Android!

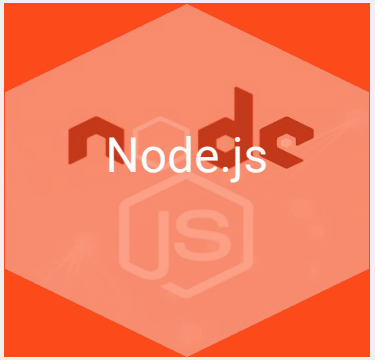


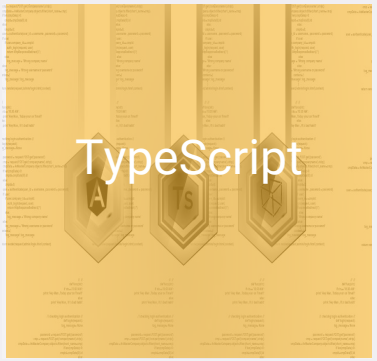
Outside Tools



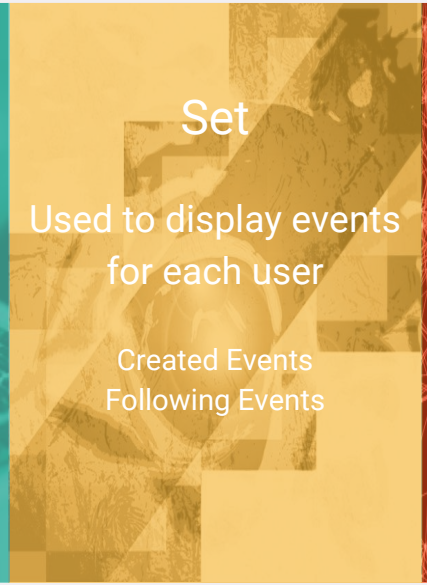
Outside Topics

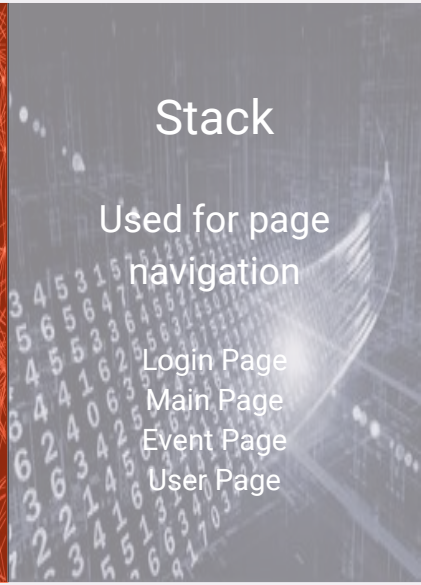






Data Structures





Classes

CSCI 103

Basic programming knowledge

Knowledge of primitive types

CSCI 104

Data structures
Fundamental Git practices

ITP 104

Internet Frameworks and Languages

Basic Web Publishing
Basic HTML
Basic CSS

Coursera, Udacity, Codecademy

Advanced web classes

JS, AngularJS
HTML, CSS

Online Tutorials and Official Documentation

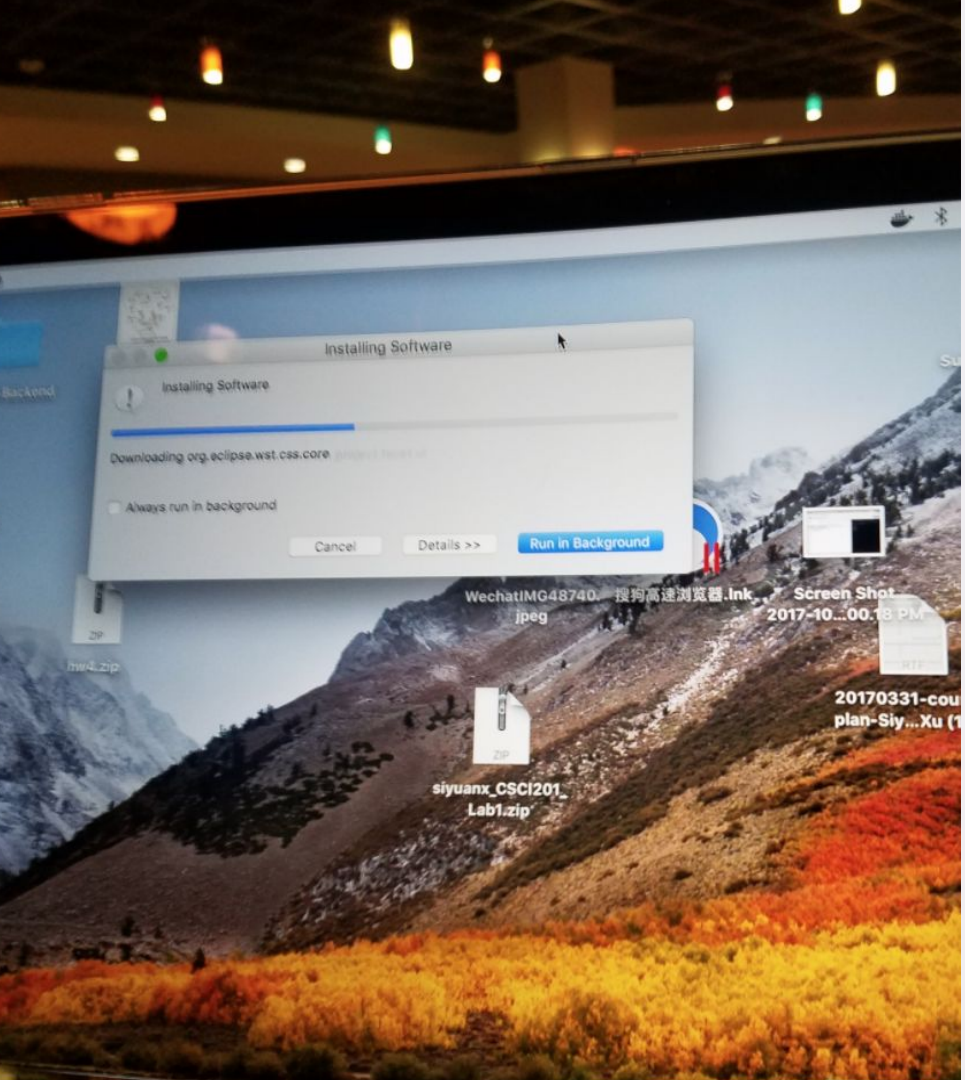
Troubleshooting, learning

Ionic Cordova
Node.js
TypeScript
AngularJS

And piano classes to calm ourselves when we cry

What Worked

- JDBC Servlets and Database
- Professional app design-- better than what we had initially designed
- More advanced functions were implemented with great effort
    - Notifications
- Meeting up at least once a week for more than two hours-- Especially for the past few weeks
- Collaborative effort with minimal arguments



What Didn't Work

- Eclipse deleting and reinstalling itself and Tomcat spontaneously
  - Forced retirement of private events/users
- Ionic Cordova: Steep learning curve
  - Format did not allow for searching for events in your immediate area
- Google Places API
- USC Events API
- Git conflicts caused code to break, be unable to be shared, and overall slowed down our progress exponentially

```java
public class Notification extends Thread{
    private GoOutServer gos;
    private Session s;
    private Calendar currentTime;
    private Calendar setTime;
    private int minuteDif;
    private String eventName;
    // passes in the notification time
    public Notification(Session s, GoOutServer gos, Calendar setTime, int minuteDif, String eventName){
        this.s = s;
        this.gos = gos;
        this.setTime = setTime;
        this.minuteDif = minuteDif;
        this.eventName = eventName;
        this.start();
    }
    public void run(){
        // when there is a certain minuntes left
        while(true){
            this.currentTime = Calendar.getInstance();
            System.out.println("Set Time 1: "+setTime.get(Calendar.MINUTE)+" currentTime "+currentTime.get(Calendar.MINUTE));
            if(differBy(setTime, currentTime, minuteDif)){
                System.out.println(Integer.toString(minuteDif)+" minutes before event starts!!!");
                gos.sendMessage(eventName+" is happening in "+Integer.toString(minuteDif)+" minutes.", s);
                break;
            }
        }
        // when it is the exact time
        while(true){
            this.currentTime = Calendar.getInstance();
            //System.out.println("Set Time 2: "+setTime.get(Calendar.MINUTE)+" currentTime "+currentTime.get(Calendar.MINUTE));
            if(sameTime(setTime, currentTime)){
                System.out.println("Event has now started!!!");
                gos.sendMessage(eventName+" is happening right now.", s);
                break;
            }
        }
    }
```

Multi-Threading

Notification

- Checks whether a new user has logged in and notifies the user of a new friend
- Regularly checks the time of an event and notifies the user when it is about to start or is going on

Networking

Front end: Phone app client

Backend: AWS hosted. Client sends information to the backend and vice versa

Purpose: To send and receive data about users, events, and notifications

Done through Javascript, AJAX, servlets, and MySQL

```
private static volatile Vector<Session> sessionVector = new Vector<Session>();
private static volatile ArrayList<Notification> notificationList = new ArrayList<Notification>();
@OnOpen
public void open(Session session) {
    System.out.println("Connection made!");
    sessionVector.add(session);
    broadcast("Another Trojan has joined GoOut. Make some friends!", session);
}
```

```
export class WebSocket2 {

  constructor() {
  }

  public static socketClient = null;

  public static connectToServer(toastCtrl: ToastController) {
    var SERVER_URL = 'ws://localhost:8080/GoOut-Backend/GoOutServer';
    this.socketClient = new WebSocket(SERVER_URL);
    console.log("initialized socket client");
    console.log(this.socketClient);

    this.socketClient.onopen = function(event) {
      console.log("Connected!");
      console.log("conected to "+SERVER_URL);
    };

    this.socketClient.onmessage = function(event) {
      console.log("Date received: "+event.data);
      let toast = toastCtrl.create({
        message: event.data,
        duration: 6000,
        position: 'top'
      });
      toast.onDidDismiss(() => {
        console.log('Dismissed toast');
      });

      toast.present();

    };
```

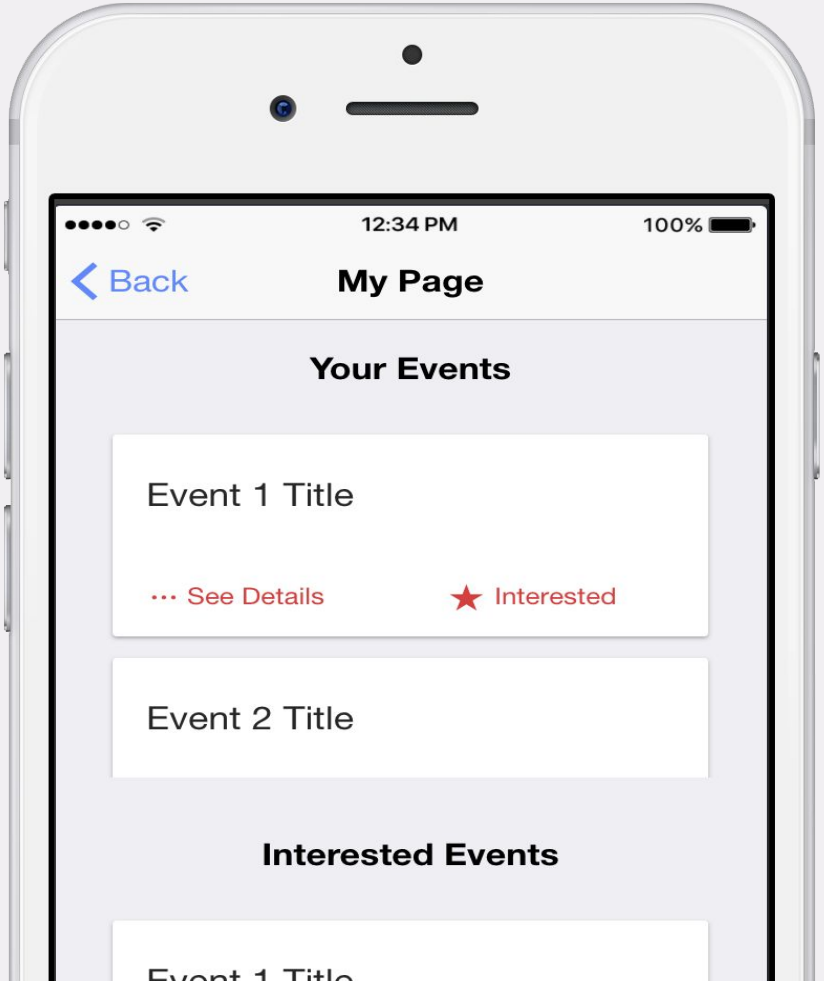
```
@OnMessage
// the message will be in format dd/MM/yyyy/hh/mm/ss/minuteDif/eventName
public void onMessage(String message, Session session) {
    int ARRAY_SIZE = 6;
    System.out.println("message: "+message);
    String[] parts = message.split("/");
    int day, month, year, hour, minute, minuteDif;
    String eventName = "";
    day = Integer.parseInt(parts[0]);
    month = Integer.parseInt(parts[1]);
    year = Integer.parseInt(parts[2]);
    hour = Integer.parseInt(parts[3]);
    minute = Integer.parseInt(parts[4]);
    minuteDif = 1;
    eventName = parts[7];
    Calendar tempCalendar = new GregorianCalendar(year, month-1, day, hour, minute, 00);
    //public Notification(Session s, GoOutServer gos, Calendar setTime, int minuteDif, String eventName){

    Notification n = new Notification(session, this, tempCalendar, minuteDif, eventName);
    System.out.println("Notification Added!!!");
    notificationList.add(n);
```

User Functionality

SIGNED IN USERS

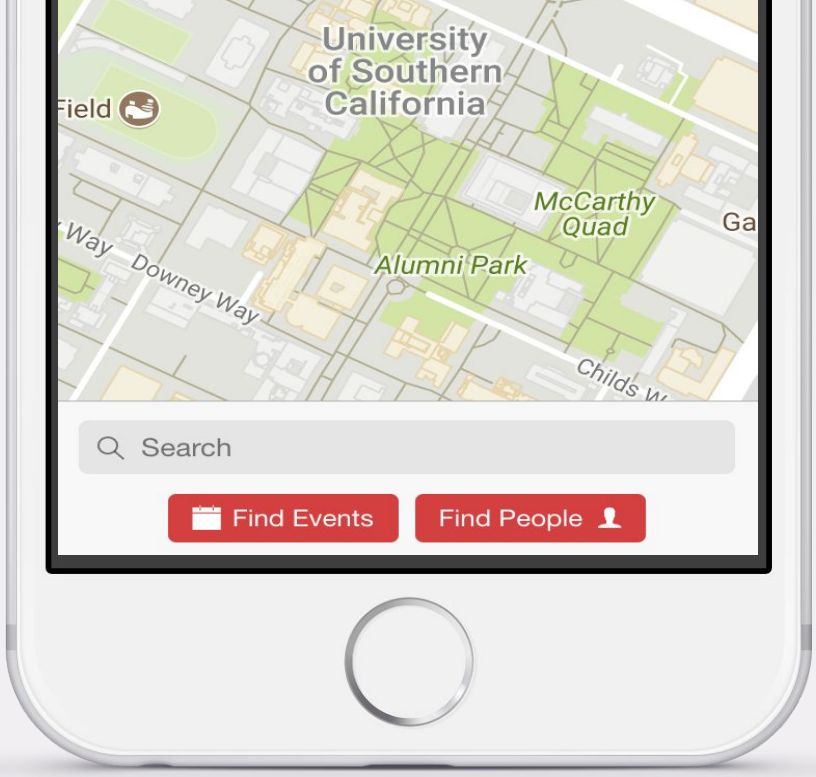
- Search for Events
- Search for Users
- See the Events on the Map
- Follow Users
- Create Events
- Receiving Notification



User Functionality

GUESTS

- Search for Events
- Search for Users
- See the Events on the Map




GoOut