Campus Feed

Team 2

Anubhaw Arya, Matt Gotteiner, Daniel Green, Mayank Jethva,

Sean McCullough, Pranav Menon, Mengyao Wang

Problem Statement

A prominent problem on campus is the fragmentation of event advertising. Instead of having to check Purdue Convocations, Facebook, Boilerlink, and fliers to find out what's happening on campus, students should have access to a one stop shop where everything they need to know is easily accessible.

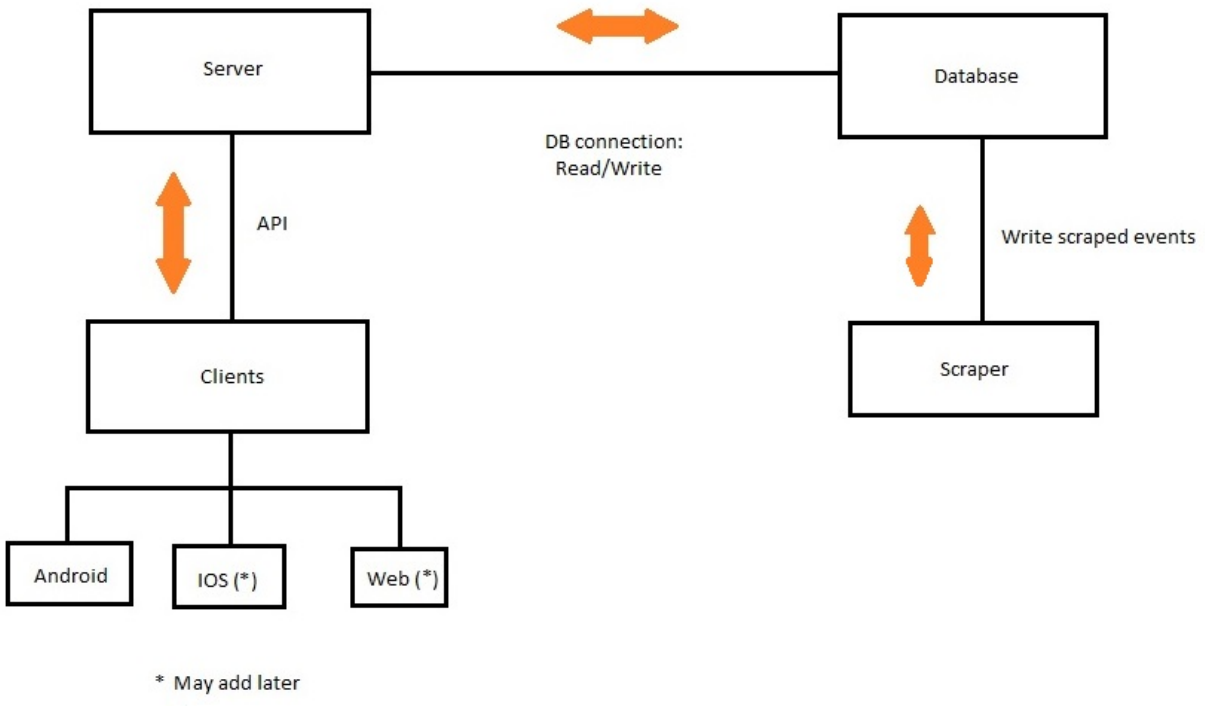
Background

There is no reliable service that provides up to date information about all the events happening on campus. The two primary methods of event advertisement are Fliers and Facebook events. Fliers are unreliable because they depend heavily on where and when you view them. For example, if there are interesting fliers posted in Beering, but I have never have time to stop and look at them when I'm in Beering, they don't do me any good. They also require a certain amount of work to design, print, and post and then become out of date, delivering valuable information too late. Facebook events seem to be the go to for event promoters, but their reach is limited by the event organizers' friends lists.

Our primary target is college students since, as college students, this is a population we're more familiar with and whose expectations we can more easily anticipate, but we're going to design our solution in such a way that it could be expanded to include more diverse populations.

There is a possible competitor in our domain called "Rumblii". The main limitation of Rumblii's solution is that event creators have to post to Rumblii for their users to see it. Having to post to multiple websites is a problem, and we address this by having a web scraper that will get event information from multiple services such as Boilerlink, Purdue Convocations, Facebook etc, so that event creators won't have to post their event to multiple websites. Because of this problem, Rumblii has trouble getting users, and is not widely used. Our event scraper will expand our user base since we will be able to show many events, not just the ones posted by users to our application. With our application, we will be able to centralize all the exciting events going on at Purdue, and provide a nice, and easy way view them.

System Model



- Server
  - Will be used to coordinate calls between clients and the database
  - Server to Database
    - Gets information that is needed for any given process from the database
      Examples:
      - Retrieving events
      - Retrieving admins of an event
    - Pushes information to the database
      Examples:
      - Creating a new event
      - Incrementing a view count
  - Server to Clients
    - Will accept requests from the clients and send back the information the client wants in the desired format
      Examples:
      - Accept search request and send back results after running requires methods
      - Logging in and authenticating a user

- Database
    - Will be used to store information
    - Send back information based on queries
    - Database to Server
        - → Will send back information based upon queries sent by server
          Examples:
            - Search for all events at Purdue and send it back to the server
    - Database to Scraper
        - → Will send back whether or not an event has been scraped or updated
          Examples:
            - Tell scraper to skip an event, as it has already been scraped or created
- Clients
    - The user interface will operate on three main platforms
        - → Android
        - → iOS
        - → Web
    - Client to Server
        - → Will send requests to the server based upon user's interaction
          Examples:
            - Request events near the user
            - Request account information
- Scraper
    - Will scrape event details from Purdue Convocations, Facebook, and Boilerlink
        - → Details include:
            - Time
            - Date
            - Location
            - Organizer
    - Scraper to Database
        - → Run queries on the database to update information
          Examples:
            - Check events online with events already scraped into the database
            - Update events in the database as they are modified on the internet

Requirements

Essential:

- Functional:
    - As a general user, I would like to:
        - → Log into the app using a social network
        - → Search for events using a filter

        - Receive notifications about events I have subscribed to
        - Create an event
        - RSVP to an event
    - As the creator of an event, I would like to:
        - Invite other users to event
        - Edit events created by me
        - Set my events public or private in terms of viewable scope
- Non-functional:
    - Data
        - Event
            - Time
            - Date
            - Location
            - Administrators
            - Attendees
            - Status
        - User
            - First name
            - Last name
            - Email
            - Facebook OAuth keys
    - Architecture
        - Database
            - MySQL
            - Language
                - SQL
        - Server
            - Amazon Linux AMI on AWS
            - Apache Tomcat
            - Language
                - Java
        - Scraper
            - Language
                - Java
        - Platform
            - Android
                - Language
                    - Java
                    - XML
            - Communicate with server via JSON

Nonessential

- Functional:

    - As a general user, I would like to:
        - View directions to an event
        - Invite users to go to event
        - Add event to watch list
    - As the creator of an event, I would like to:
        - See if other events conflict with my event
        - View event analytics
    - As a group member, I would like to:
        - Leave a group
        - Join a group
        - View group members
        - View group information, such as description
    - As a group leader, I would like to:
        - Set group visibility to public or private
        - Create a group
        - Add/remove group members
        - Make group specific events
- Non-functional:
    - Data
        - Group
            - Members
            - Admins
        - User
            - OAuth Keys
                - CampusFeed
    - Architecture
        - Web
            - Language
                - HTML
                - CSS
                - JavaScript
        - iOS
            - Language
                - Objective-C