EventHub

Erik Chou, Kun Wang, Yukio Maeda

Overall Product Description:

EventHub is an event management tool that allows organizers to post events and, vice-versa, allows residents to filter and choose which event they want to check out. By signing up, organizers are able to post details about their event which will be used to advertise when users are looking for things to do. Residents then use the website to filter (based on their interests) and select which of these events they want to check out.

Problem Background & Vision:

Generally, when people want to have a good time or do something exciting, they go out to Seattle. Despite this, people don't seem to know what's happening around their block much less their district or around the city. Unfortunately, this leads to lots of frustration or boredom as they end up doing the same old things.

Local newspapers and websites such as The Stranger or Seattle Weekly somewhat rectify this problem by recommending events to people. However, because of the generality of their audiences, many people may be uninterested in those events. Also, many events are excluded due to the differences in advertising procedures as well as the difficulty of competing against more profitable or favored events. Tools such as Twitter and Facebook also try to solve this problem by allowing people to look at an event calendar to see what is happening. However, these tools only recommend events that your friends or followees know about.

With EventHub, all these problems are solved as the website will allow residents to view and choose from all events that seem interesting to them. Additionally, EventHub will allow an easy posting procedure and unbiased event selection, making it easier for organizers to better advertise their events. Despite the fact that the scope of this project is restrained to the city of Seattle, our vision includes extending this feature to other cities. Our vision also includes full-social network integration as well as personal event recommendations based on a person's event attendance.

Existing Solutions

Seattle Weekly (http://www.seattleweekly.com/calendar)
+ Rating system with Facebook/Twitter integration (likes, tweets, etc.)
+ Specific info given
- Doesn't display many daily events (mainly weekly or monthly)
- Heavily advertises for certain events (unfair advertisement??)

The Stranger Suggests (http://www.thestranger.com/seattle/Suggests)
+ Interesting and appealing design (not completely blocky look = nice change)
+ Rating system with Facebook/Twitter integration (likes, tweets, etc.) and comments
- Only constrained to limited timeframe (can't plan ahead 2+ weeks)
- Very limited event choice (few in comparison to SCVB)
- No search (and default website's search option is also horrible)

Projected Architecture:

1. Front End
-Able to update instantaneously depending on filter options and event statuses
-Utilizes templates for better display
-Able to display varied number of events depending on time and website registration

2. Back End:
-Searches database for particular events based on filters
-Retrieves and organizes information from back-end before giving to front-end
-Weights events depending on user preference
-Filters out bad requests & checks for “bad” data
-Performs periodic maintenance of database (additions, deletions, etc.)

3. Database Layer:
-Stores event data and user info

Technology Stack

1. Front End:
-HTML/CSS/Javascript
-Possible use of Twitter Bootstrap

2. Back End:
-Java Language
-Spring Framework

3. Data Storage:
-MySQL or MongoDB

Minimum Viable Product (MVP)

Our minimum viable product should be able to perform four particular actions:
-View: Displays events on the website
-Search: Selects specific events based on event type and data
-Post: Event posting with specific information and description
-Register: People can sign-up and edit/monitor what they posted

Future Additions

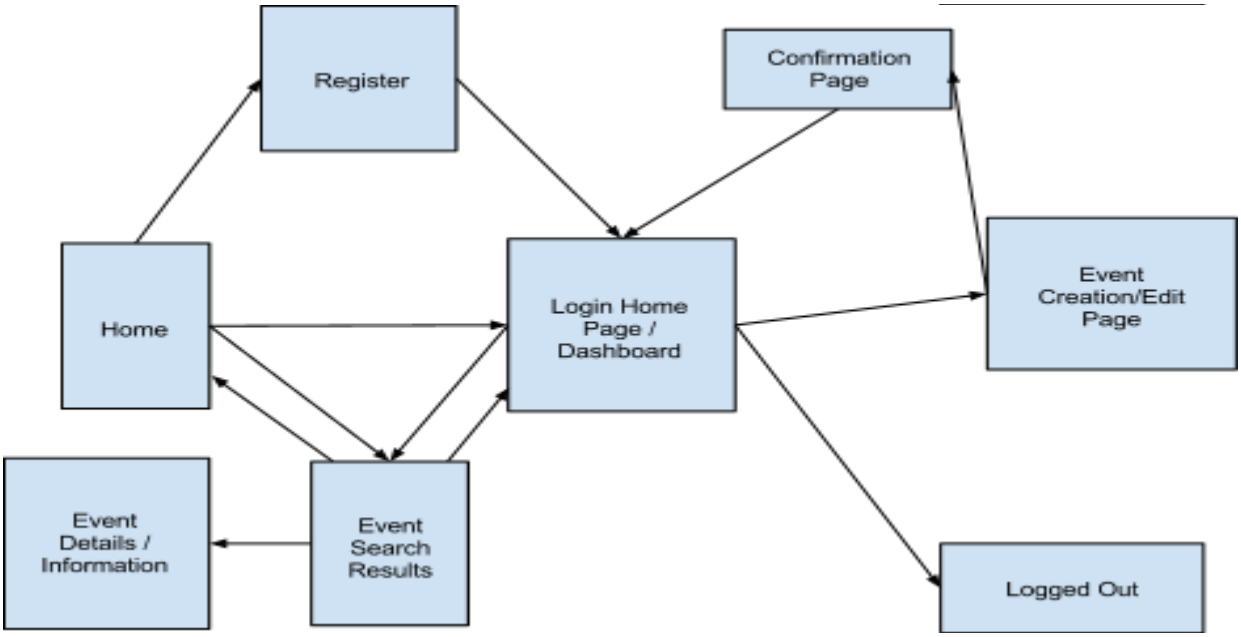
Here are a few additional features that can be implemented after completing the MVP:

-Commenting / “Hype” system (additional user feedback)
-Facebook/Twitter integration
-Recommendation of events based on previous selections
-Ticketing functionality
-Mobile app functionality

Risks

-Not much familiarity of Spring framework
-Inability to resolve information overload and/or detect information overlap (too much information to store or duplication of events = overload of databases)
-Basic event extraction/searching may require way too much overhead/work

Basic Application Flow



(Note: some events have been simplified or temporarily removed; doesn't reflect exact flow)