Free Food Engine Project Backlog

1/31/14

Kishen Sivalingam
Qing Wei
Yinchen Yang
Sagar S Pujara

Problem Statement:

In an University environment often times there are many free food available for students to enjoy and student would like to get information on how to get this information. The problem is there no any easy way to get this information around Purdue campus. Our solution involves creating a web application that solves this problem for everyone. This application stores information about free food that is available around Purdue campus in one easy to navigate web application.

Background Information:

Independence and financial foresight are qualities many students adapt in college. Expensive tuition fee and high cost of living forced students to learn to save money. Since, often on campuses that are many opportunity for students to get free food, students can save money. Unfortunately, there is no any easy way for student to find this free food. Our web application Free Food Engine sets out to make saving money a little easier by helping students find free food on campus and thus getting more involved in campus activities. Also giving the opportunities to the student organizations and local businesses a chance to post their free food information and at the same time promote their event to students.

Environment and System Models:

Free Food engine web application will be accessible through any web browser on any (modern) web-enabled device. The application will intelligently scale to fit the device's screen. The full system will be basically composed of 3 layers. And our job is to develop the first two layers (Server and Client) and the third layer is social services provided by social networking websites so that user could share or login using their social networking accounts. The first layer is the web server and database layer. The backend should be constructed mainly service oriented by using PHP and the job of the server is to handle the clients' requests and query the correct data from database and send them back to the frontend. The second layer is the client, in this case the web interface. The client should handle the user inputs and form the correct requests to the server and response the data back to users.

Non-Functional Requirements:

- Code Maintainability
    - The code should be easy to read and understand.
- Easy to navigate user environment
    - The web interface should be easy to use and user-friendly.

- Robustness
    - Minimize page crashes and server should not fail
- Scalability
    - Include more users and requests made to server
- Fast and efficient performance
    - < 2 second response for an action
- Security
    - Only save encrypted user credential information in database.
    - Users are notified before we retrieving data from facebook account.

Functional Requirements

- Add Free Food event's around Purdue Campus
    - Login
        - As an event organizer, I would like to login into the web application using facebook.
    - Logout
        - As an event organizer, I would like to logout the web application.
    - Admin privilege
        - As an event organizer, I would like to add event's to my account
    - Add event to map
        - As an event organizer, I would like my event to show up in a map with a customized marker.
    - Modify an event
        - As an event organizer, I would like to modify an event by changing its name, location, content and time.
- Navigation
    - Data Retrieve
        - As a user, I would like to see a list of free food information including where and when based on a daily basis.
        - As a user, I would like to view the free food information on the map widget based on selection.
- Social Media Integration
    - Facebook integration
        - As a user I would like to create a Free Food engine account through linking with my facebook account
        - As a user, I would like to log into my Free Food Engine account using my Facebook account.
        - As a user, I would like to share free food events to my friends on Facebook user's timeline, regardless of whether or not they use Free Food engine
        - As a user, I would like to suggest Free Food Engine to my friends.

        - As a user, I would like to like a post of Free Food Engine
- Platform support
    - Web interface that adapts to any device
        - As a user, I would like to view my saved items from any web-enabled device
- Suggestion
    - Based on what you and your friend like
        - As a user, I would like to suggest content to a specific friend Facebook friend
- Commentary Feature
    - Comment box for a specific events
        - As a user, I would like to comment on a specific event

Use Cases

Case 1: Event organizer wants to login	**System Response**
1) User goes to the login page	2) no response
3) User click the login with facebook	4) Facebook authentication box shows up
5) User enters facebook credential and clicks login	6) i) If authentication is successful, user will be directed to user's home page ii) If authentication fails, user will redirected to login page where user will be notified they authentication failed

Case 2 : Event organizer wants to log out	**System response**
1) User select logout from drop down menu in the navigation bar	2) User will be redirected to homepage not logged in

Case 3: Event organizer wants to add events to account	**System Response**
1)User clicks on add event button	2)A pop-up frame appears with a event form for event details

3)User fills out the form for event details	4)no response
5) User submit the event form	6 i) if data in the form are correctly formatted, a new event will be created ii) if data is invalid, the border of that field should turned red

Case: 4 Event organizer wants to show the event on map	**System Response**
1) User successfully submitted the event form	2) A specialize marker is put on the map, marking the location of the event
3) User change the location of the event manually, by drag and drop the marker to a specific point on the map	4) The system will automatically update the location of the event in the database.
5) User click's on the marker	6) A pop-up frame appears with information regarding the event

Case: 5 Event organizer wants to modify an event	**System Response**
1) User click modify an event in the list of events associated with the account	2) no response
3) User change the event information and submit the form	4) The system will automatically update the information by updating the database

Case 6 : User wants to see a list of events sorted by time	**System Response**
1) User click on the sort button	2) List of event is sorted in according time

Case 7 : User wants to see nearby events on map(Request login)	**System Response**
1) User click login button to go to login page	2) If authentication is successful , the user should be redirected to the homepage logged in.
	3) User will be prompt to allow website to get user's current location.
4) User's accept the request	5) System shows all event that are nearby the user.

Case 8 : User wants to update the status of an on-going events	**System Response**
1) User click an events to view or edit the detail of the event	2) User will be asked to allow website to get user's current location
3) User allows the website.	4) Website determines if the user is close enough to the event location i) If yes, user will be allowed to post updates about the event (g. run out of food, what kind of food is served, welcome to whom, writing down comment) ii) If no, user will not be allowed to update the event. User can only view the event

Case 9 : User wants to view the website on different devices	**System Response**
1) User uses a mobile browser to view website	2) System automatically adjusts website to fit into mobile browsers

Case 10 : Case: User wants to share the event on facebook timeline	**System Response**
1) User clicks the share button below each event	2) System communicates with facebook API and a pop up appears
3) User clicks share	4) System communicates with facebook API again and post the event user's facebook

	timeline.

Case 11 : User wants to like an event	**System Response**
1) User clicks like on an event	2) The click in recorded in the database and the counter is updated.

Case 12 : User wants to share content to a specific friend	**System Response**
User click on the share button	2) System communicates with facebook API and a pop up appears where user can choose to share the content to friends timeline
User chooses to share content on friends timeline.	4) System again communicates with facebook API and posts the content to user's friends timeline

Case 13 : User wants to comment on an event	**System Response**
1) User click the comment hyperlink below on every event	2) A text area appears for user to comment
3) User comments and press enter	4) Comment will be post in the comment area