**It’s Happening**

**Software Design**

**CSCI-P465/565 (Software Engineering I)**

**Project Team 6**

**Arpit Bhatnagar**

**Roja Raman**

**Jing Huang**

**Arpit Bansal**

**1. Introduction**

This section introduces the design approach to the software system.

**1.1 System Description**

“It’s Happening” is a Bloomington based online event management and ticketing website which allows users and organizers to browse, book, host and promote events. Through us, organizers can easily promote their events and get statistical data. Users can browse events by category and location on map and use promo codes along facility of pickup and drop with Uber. We also provide support through chat feature to help our users.

**1.2 Design Evolution**

This section is intended to document the rationale behind the selected design solution.

**1.2.1 Design Issues**

The application only requires access to a web browser on any PC and internet connection to access our system.

**1.2.2 Candidate Design Solutions**

We will be building majority of the system components using PHP for the backend and HTML, CSS, JQuery and JavaScript for the front end. The database being used will be Amazon RDS MySQL and deployed the application on the Amazon Web Server Ubuntu EC2 instance. We are using a third party client called Live Chat for implementing chat feature and Google Maps API , Google Adsense to implement maps and advertisement features, Event page , Organizer page, Admin page and Payment, Ticketing and Promotion coupon features . Uber API for uber ride estimate and ride booking feature.

**1.2.3 Design Solution Rationale**

PHP based web applications can be developed with any IDE on any platform with any localhost installed. It does not oblige any strict or specified framework or environment and since it is widely used, has a lot of documentation and examples available over the internet which would be helpful in case we run into any issues. Using JavaScript and CSS for the front end (along with PHP) provides us a great recipe for building highly dynamic and interactive websites.

AWS hosting is an easy to use and reliable medium to host php scripts and MySQL instance and database, helping us ensure high availability of the application we are interested in creating. We created an AWS Ubuntu EC2 instance and Amazon relational database service to deploy our php web application.

**1.3 Design Approach**

**1.3.1 Methods**

We exercise the Model-View-Controller design architecture in our System. That is, storing all data inside the Amazon RDS - MySQL database (Model), representing all user interactions on web pages (View), and manipulate data and update views through controller.

We have designed the Login and User registration features which allows a user to sign up with us and/or login to our website to gain seamless access to the varied options we will be providing them. Along with this, they also have the option of resetting passwords through our ‘forgot password’ feature (this will consist of resetting passwords either using a temporary password which we will be mailing them or by answering a security question on our site). We have also implemented OAuth using Google’s API. In addition to these, to provide a level of security, we are encrypting their passwords before storing them in the database.

The home page which contains a finite horizontal scrolling banner on the top for showcasing the featured events (which will be available as an optional feature to our organizers for a flat fee). The user can search for various events here as well by typing in the required keywords in the search box provided filtered on basis of date or category of event. We have a section for our users to check out upcoming popular events, which they can look up and register for easily. We also have a categories section wherein the user can explore events based on categories like Music, Arts, Sports, Business, etc. It navigates them to a browse page which provides a brief description of the event(s) along with the location displayed on google maps. The integration of google maps also allows a user to find events happening around him/her by picking their current location. We implemented a chat feature where the user can chat with the website administrator, which will assist them in resolving any kind of technical issues faster and provide a friendly and interactive user experience. Google ads will be displayed in the browse page. This used Google adsense technology and hence as per Google’s prerequisite rule, the ads will be displayed according to the traffic to the website. Organizer can create event for a nominal fee, view events pending for approval and upcoming events. Also, organizer can view the statistics of the events created by him/her, upcoming event and the revenue generated by the events. Admin page will show all the organizers registered to the website. Admin can then approve or disapprove an organizer. statistics of Number of users registered, organizers, revenue generated and number of events hosted on the website on the dashboard.Payment and Ticketing system is implemented for user to buy tickets through Paypal and enter promo codes for offers. User can book Uber rides through the website. User can view their tickets in the profile page, Number of tickets bought and event information. User can edit their profile information from profile screen. Also on event page, user can view their ticket information, scan a generated QR Code to get the ticket information in their phones.

We would be performing a unit, system integration and regression testing on our application. Unit tests are for stability and code predictability, which regression testing is for evaluating how users interact with the code.

**1.3.2 Standards**

Passwords are being encrypted using the MD5 algorithm which is a widely used hash function which produces a 128-bit hash value. This provides us basic immunity against man-in-the-middle attacks. After hashing, the hashed password is stored in the database to verify during the next login and authenticate the user. This is a standard security protocol regarding passwords.

We have made sure to use modular programming techniques to help make our code scalable and east to test and are adhering to the standards and best practices mentioned in the official documentation when using PHP and MySQL. Our features are implemented as separate functions\methods in the PHP code and form the core components of our offering. We are using HTML5 to design our pages and CSS3 to provide a better overall look and feel of the GUI.

**1.3.3 Tools**

· IDEs and Editors – Netbeans, Sublime, Notepad++

· Web Server - Xampp (Apache)

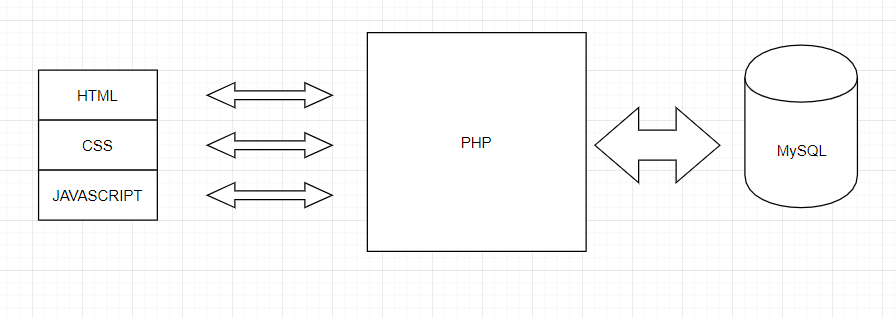
· Database - Amazon RDS - MySQL

· Hosting- AWS

**2. System Architecture**

**2.1 System Design**

At a high-level, our system will have classes and functions for each component, implementing a specific feature that we are planning to offer. We will ensure to keep the front and back end separate and such is the nature of the languages we have chosen to use as well. Many of the interfaces and features will be implemented using pre-existing libraries and use of open APIs. At a basic level, each functions has its own Logic, URL Routing and unit tests. We have also used a basic template for our HTML pages and the formatting will be inherited for all others which follow.



**2.2 External Interfaces**

Our system interacts with any desktop browser that runs on html, js, css and php. Each browser hosts an individual event. We use AWS to host web pages and Amazon RDS to store all data. Real time data (Live Chat) will be hosted through a 3rd party tool called “Live Chat” to facilitate interaction between user and admin. We also implemented “Google Ads” which will be displayed according to the traffic of the website.

For payment and ticketing, Paypal API is integrated as a secure payment gateway.

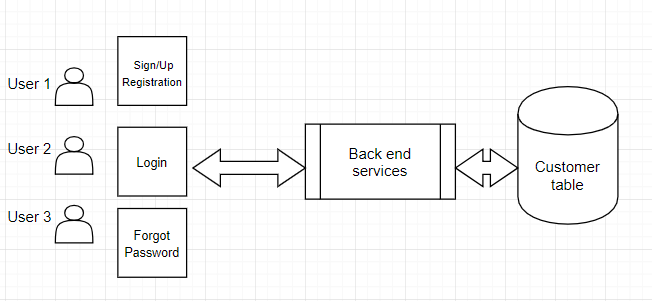
Uber API is used to help users book ride and find out the estimated ride cost.

**3. Component Design**  
The Component Design section details the proposed design of each system component.  A system component is a functional partition of the system.  Components may be organized as you see fit - a component may be a collection of objects, or a single object.  However, a system must be composed of multiple components (that is, a system cannot be one component).  The layout of this section is at your discretion, but please include the following (at a minimum) information for each component:

* **Component Name**  
  Login and Registration
* **Component Description**

The login form allows users to login with their credentials which are stored in the MySQL instance hosted on the Amazon RDS. When the correct credentials are provided, the user is redirected to the home page for our website wherein he can explore the available events (home page is still under construction). The login request is encrypted and so is the password stored on the database (using MD5). We also have a sign up page wherein a new user can provide few required details and sign up with us. We have also implemented OAuth using a third party API (Google) to enable users to directly register with their Google credentials.

* **Responsible Development Team Member**  
  All the team members divided tasks from this components into subtasks and worked together on the integration.
* **Component Diagram**



* **Component User Interface**  
  The user interface includes two text entries for username and password, login button, “remember me” box, “sign up” link, “forgot password?” link, and google login button. By clicking on “login”, registered user logs in with valid credentials. By clicking on “Sign up” user receives a pop up with sign up option. By clicking on “forgot password?” user resets password using valid credentials. By clicking on “Google login”, user receives a gateway to google accounts.
* **Component Objects**

*User Component*: A user who will visit our page using the website’s URL and then will either sign up if he is a new user or login using the credentials he created previously (existing user)

*Login Component*: The different objects under this are:

Login: Authenticate User based on the provided credentials

Registration: Form to register new user, contains required fields to collect user details and email

Forgot Password link: This will either redirect user to another page wherein he can answer the security questions and then create a new password for himself or will send a password reset link to his email address with a new default password which we will provide.

*Database Component*:

- Customer table: Stores all customer related data: username, password, email address, user type, name, etc. and is used to authenticate users based on the database calls made on the server.

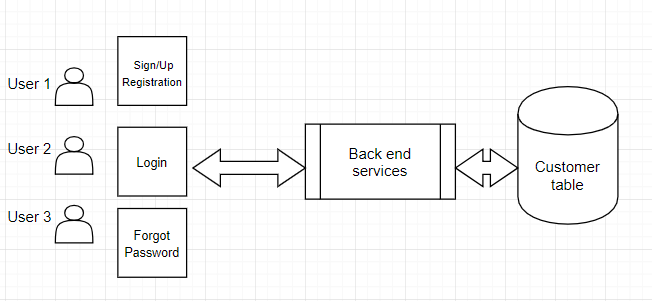
- Event table: Stores all event related data: organizer, location, event type, Tickets etc.

- Event Registration table: Stores relationships between events and event attendees.

* **Component Interfaces (internal and external)**  
  Component interacts with user through a desktop browser, and retrieves data from MySQL database through PHP.
* **Component Error Handling**  
  If a user tries to login with invalid credentials (username or password), they will he handled in the component.
* **Component Name**  
  Dashboard and Search
* **Component Description**

The Dashboard page allows logged in users to explore the different events which our site has to offer. The scrollable banner highlights the ads we will be featuring for our organizers at a fee and user will be able to navigate to these directly. This will also be used for any announcements, promotions and messages we will have for our users. The user can search for events of interest using the search box provided which queries the ‘events’ table in the database using the keywords just punched in. They are redirected to the browse page to display the results fetched from the database in JSON format. This page also allows for user’s location to be picked up on google maps and then displays upcoming events which are happening nearby. We also have created categories for popular types which are available on the homepage for the user to explore.

* **Responsible Development Team Member**  
  All the team members divided tasks from this components into subtasks and worked together on the integration.
* **Component Diagram**



* **Component User Interface**

*Homepage UI*:

- Banner: Automated image display.

- Search: Search for any event using search bar and filter by category or date.

- Upcoming Events: Horizontal scroll by use on viewing upcoming events.

- Browse by Category: Display cards of categories of events user wants to browse.

- Dashboard: Links user can click on to enter another page. (login, profile, browse, etc.)

- Live Chat: A panel for live chat throughout the website.

*Browse page UI*:

- Event Cards: Vertically display events.

- Map: Locates user’s location and events searched.

- Dashboard: Links user can click on to enter another page. (login, profile, browse, etc.)

- Live Chat: A panel for live chat throughout the website.

- Ads - This block will display ads and redirects to the 3rd party site when .

*Event page UI:*

Displays event details and provides payment options. The components are:

*-* Google Maps: It will display all the events nearby basis on the event category

and will also display images on the map related to the genre of the event.

- Facebook and twitter links for events - It will redirect the user to the event page on facebook and twitter through which a user can get additional details about the event.

- Payment - A guest or a registered customer can buy tickets on the basis of coupons or without coupons through paypal. A customer can pay his ticket amount either by logging in with his paypal account or as a guest. A customer can put his valid cc/dc details and can proceed with the payment. After successful payment a valid message will be prompted on the screen.

- Promo codes- A user based on his privileges will be allowed to use specific coupon codes and will get discount on his total amount during purchase of the ticket.

- View Ticket - Displays a QR Code, upon scanning which user can get ticket information on their phones.

- Uber ride booking- User can book rides to the event venue from our website by clicking on the button provided on view ticket popup.

*-* Similar events: Displays other events available on the site based on the genre of the event being looked at

*Admin dashboard:*

* Event list: Presents a list of newly registered organizers. Admin can approve and disapprove an organizer profile. On approval, organizer will be able to create events and host them on our website.
* Admin can view the statistics of Number of users registered, organizers, revenue generated and number of events hosted on the website on the dashboard.

*Organizer dashboard:*

* Overall charts: Displays statistics of the events hosted by the organizer
* Lists: Provides a list of Organizer’s events, events pending approval, and upcoming events
* Create events option: Provides a form to the organizers with fields requires to create an event which will be sent for admin review.
* **Component Objects**

*Homepage Component*:

- **Upcoming Events**: Displays events by upcoming dates.

- **Browse by Category**: Once user click on a specific category, calls database to search by category, and redirect to browse page.

- **Search**: Search for any event using search bar and filter by category or date. Once user click on search, data is fetched from the database accordingly and populated in browse page.

- **Live Chat**: A panel for live chat throughout the website. User can start a chat by entering name and email address in the chat panel and ask for any sort of query. It will show if the user is available or not, is typing and option to close the chat.

- **Admin dashboard:** This dashboard receives all the newly registered organizers and User, Events and Revenue statistics.

- **Organizer dashboard:** Displays upcoming events, pending events and events created with statistics related to it.

*Browse page Component*:

- **Event Cards**: Pulls data from json and display details on screen. Once user clicks on a specific event, browser redirects to that event’s homepage.

- **Map**: Calculutes user’s geolocation based on IP address. Mark user’s location on map and all events’ on this browsing event.

-**Live Chat**: A panel for live chat throughout the website. User can start a chat by entering name and email address in the chat panel and ask for any sort of query. It will show if the user is available or not, is typing and option to close the chat.

- **Ads** - This block will display ads from google.

- **Uber Integration -** User can book rides and find out estimated ride cost to event.

*Database Component*:

- Event table: Stores all event related data: organizer, location, event type, ticketing etc.

* **Component Interfaces (internal and external)**  
  Component interacts with user through a desktop browser, and retrieves data from MySQL database through PHP.
* **Component Error Handling**  
  If a user tries to login with invalid credentials (username or password) and perform incorrect search, this will be handled in the component.

**Revision History**

|  |  |  |
| --- | --- | --- |
| **Revision** | **Date** | **Change Description** |
| v.1.0.0 | 09/30/2018 | Sprint 1 Initial Version |
| v.2.0.0 | 10/14/2018 | Sprint 2 feature additions |
| v.3.0.0 | 10/28/2018 | Sprint 3 feature additions |
| v.4.0.0 | 11/10/2018 | Sprint 4 feature additions |
| v.5.0.0 | 12/02/2018 | Sprint 5 feature additions |