PLAN & PLAY

CS 428 Spring 2016

Divya Balakrishna (dbalakr2)
Felicia Chandra (fchandr2)
Gul Mariam (mariam2)
Hao Zhang (hzhang49)
Hathfi Hakim (hbhakim2)
Hong Jae Jeon (hjeon17)
Chris Foti (foti2)
Paarth Joshi (psjoshi2)

Table of Contents

1. Project Description 3
2. Process 3
    2.1. Iterative Development 3
    2.2. Collaborative Development 3
    2.3. Testing 4
    2.4. Refactoring 4
3. Requirements & Specifications 4
    3.1. Technical Specifications 4
    3.2. User Stories 5
        3.2.1. Sign Up, Login, Logout 5
        3.2.2. Activate Account 6
        3.2.3. Manage User Profile 6
        3.2.4. Reset Password 6
        3.2.5. Manage Event 7
        3.2.6. View Event 7
        3.2.7. Join & Leave Event 8
        3.2.8. Home Page 8
        3.2.9. Search by Location 8
        3.2.10. Event Comments 9
        3.2.11. Event Media 9
        3.2.12. Recreate Past Event 10
        3.2.13. View Participant Profile 10
        3.2.14. Notifications 11
4. Architecture & Design 11
5. Reflections 14
    5.1. Divya Balakrishna 15
    5.2. Felicia Chandra 15
    5.3. Gul Mariam 16
    5.4. Hao Zhang 16
    5.5. Hathfi Hakim 16

5.6. Hong Jae Jeon 16
5.7. Nicholas (Chris) Foti 16
5.8. Paarth Joshi 16

1. Project Description

Getting the capacity to rapidly organize and plan a sport activity with friends is difficult because of busy college schedules. Today, students use social media to ask their friends to join an activity, but this is not efficient especially if they need a big group where multiple layer messages are necessary to find and form a group to play with.

PLAN & PLAN is a web application which addresses the problem by enabling its users to organize sports-related events without the burden of searching for others to participate. Users can search for activities in their area which they are interested in, join the events, and discover new friends with similar interests.

2. Process

2.1. Iterative Development

We mainly followed Scrum for the iterative development methodology. At the beginning of the project, we came up with all features and user stories we would like to implement, including the estimated units to complete the user stories. Afterward, we worked in a two-week iteration to develop the application, with a total of 6 iterations throughout the whole semester. At the beginning of each iteration, we selected the user stories we would like to work on, usually around 3 to 5 user stories. At the end of each iteration, we recorded the actual units and re-estimated future user stories.

The main issue we ran into was that we sometimes estimated the difficulty of a task incorrectly. There were some user stories where we overestimated the units, while there were some others which took longer than expected, thus we had to continue working on the tasks in the next iteration.

2.2. Collaborative Development

For each user story, we usually assigned two people to work together. Each member was free to choose which user story they wanted to work on, and the two people who selected the same user story were then paired together. This way the pair assignment was different in each assignment, and therefore every member had a chance to work with various people in the team. At the completion of each user story, the pair would review each other's code to ensure that the coding standards were maintained and faults were found early.

The whole team would meet at least once a week to provide status updates and discuss any issues we encountered. We would help each other resolve the issues or re-adjust the tasks if necessary. In addition, we also used Facebook chat to facilitate discussion outside of the meeting times. Here, we could ask for opinion from team members or shared important knowledge we learnt during the development.

While developing collaboratively, we often encountered an issue while merging code. Sometimes there would be a conflict and it would break each other's code. To prevent this issue, we tried to pull from the repository frequently and push the changes regularly. When we

encountered a conflict, we would notify everyone in the Facebook group so the affected authors could help resolve the problem.

2.3. Testing

To prevent regression issues, we developed automated testing for the functions which interacted with the database. We created a separate testing database, in which the content would be refreshed with pre-determined testing data before running each test.

We also integrated Selenium to test the user interface. We particularly checked for the existence or non-existence of buttons and error messages depending on the scenarios we were testing for.

At the beginning of the project, we had difficulties with connecting PHPUnit to the database since no one in the team had any experience with the testing framework. We were able to make it work after multiple attempts of trial and error. We also had to downgrade the PHPUnit version in order for the Selenium testing to run.

2.4. Refactoring

At the beginning of every new iteration, we went through the code from the previous iteration and refactored the code wherever necessary. We specifically looked for repetition of code or other code smell that would cause problems in the future. We also wanted to ensure that the code format was consistent throughout the entire code set.

When refactoring the code, we had to be very careful and checked all parts which would be impacted by the changes. Sometimes the refactor caused some unit tests to fail, so we had to reflect the changes in the tests as well. Additionally, we had to ensure that the affected features were still working properly on the web application.

3. Requirements & Specifications

3.1. Technical Specifications

Platform: Web
Front-end: HTML, CSS, Javascript
Back-end: PHP
Database: MySQL
Testing: PHPUnit, Selenium
Framework: TINY, Bootstrap
API: Google Map
Web Server: Apache
Production Hosting: cPanel (http://plannplay.web.engr.illinois.edu)
Version Control: GitHub (https://github.com/divyabalakrishna/CS428PNP)

3.2. User Stories

3.2.1. Sign Up, Login, Logout

Users can create an account, login, and logout from the application.

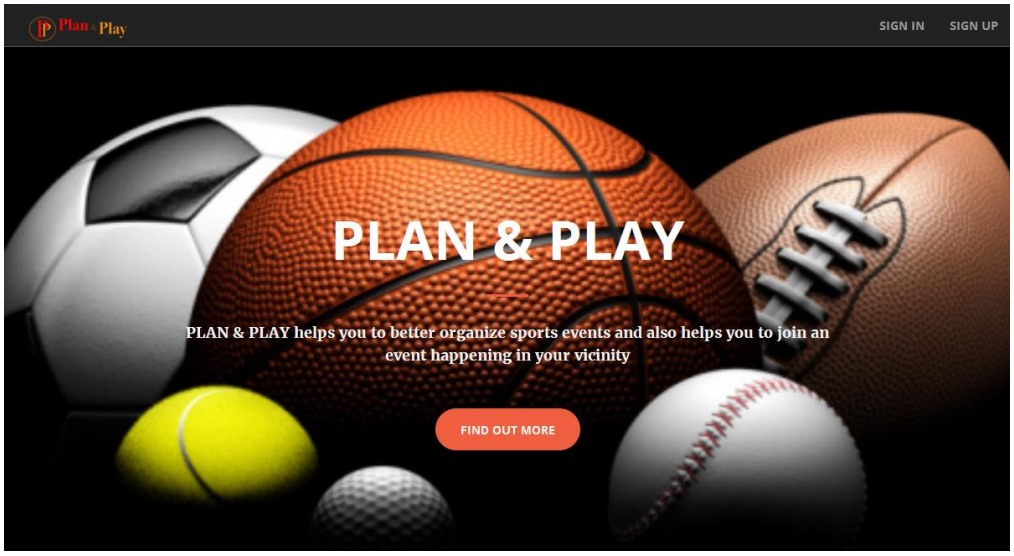

Figure 1: PLAN & PLAY Main Website

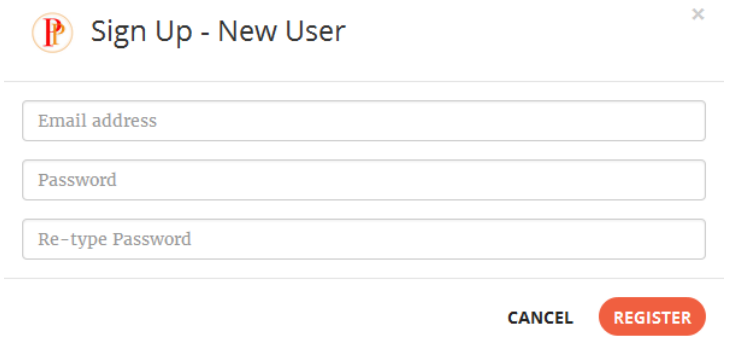

Figure 2: Sign Up

Figure 3: Sign In

3.2.2. Activate Account

To verify the email address, users will receive an activation code by email after creating an account. Users are required to activate the account to use the full features of the site.

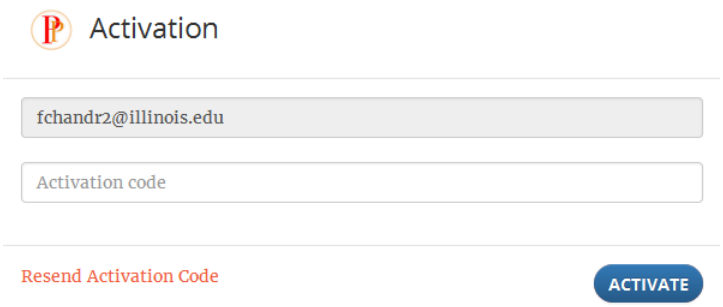

Figure 4: Activate Account

3.2.3. Manage User Profile

Users can edit their profile, which includes personal information, profile picture, and interests.

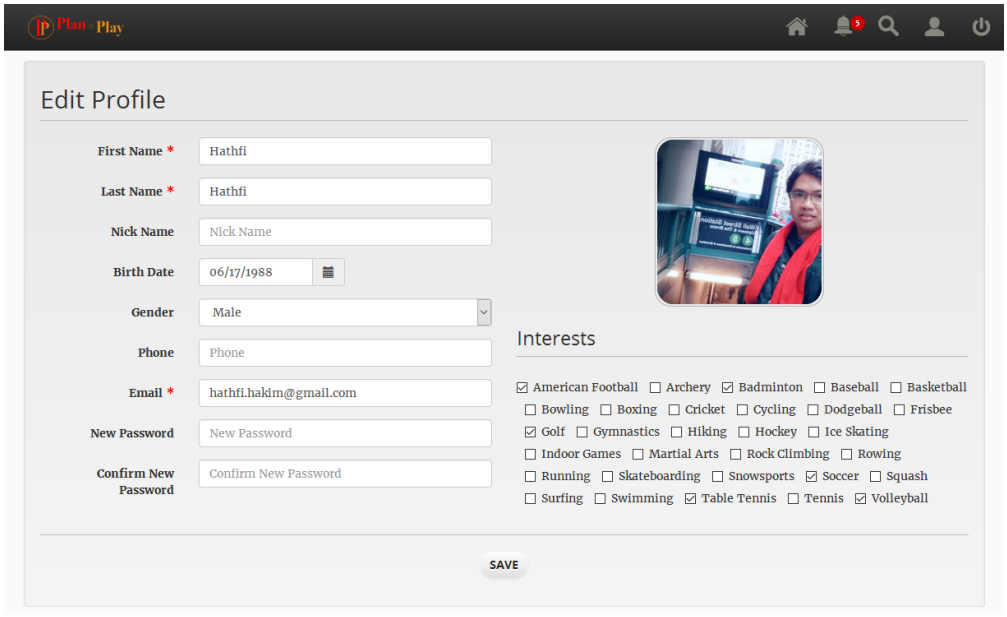

Figure 5: Edit Profile

3.2.4. Reset Password

Users can change their password by entering a new password on the edit profile page. Users can also use the Forgot Password link on the login window to receive an email with the instructions on how to reset their password.

3.2.5. Manage Event

Users can create an event. They can also edit and delete a future event they previously created.

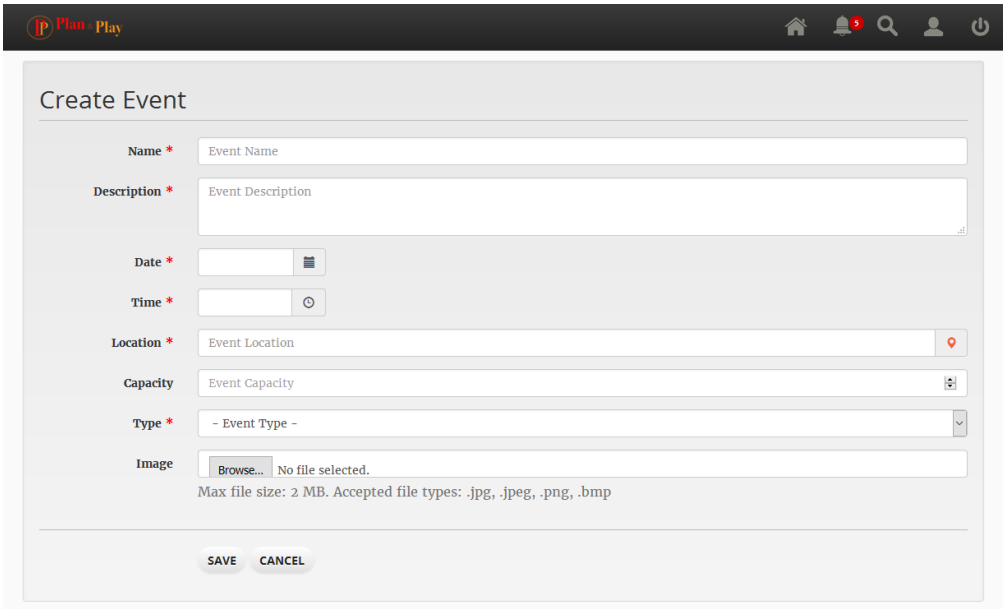

Figure 6: Create Event

3.2.6. View Event

Users can view the details of an event, including a list of participants who have joined the event.

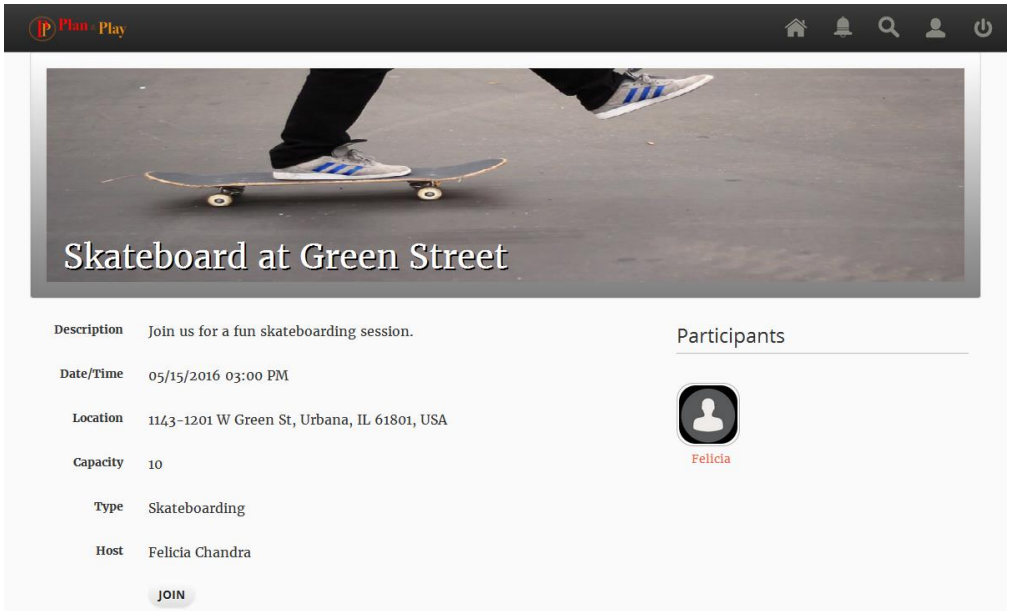

Figure 7: View Event

3.2.7. Join & Leave Event

Users can join a future event if the maximum capacity has not been reached. They can also leave a future event they previously joined.

3.2.8. Home Page

Users can view a list of events on the home page after they have logged in. There are four categories: hosted events, past events, joined events, and event feed. By default, the home page will show event feed, which is a list of future events created by other users and the user has not yet joined. Each event block displays a brief information, including title, description, venue, date and time.

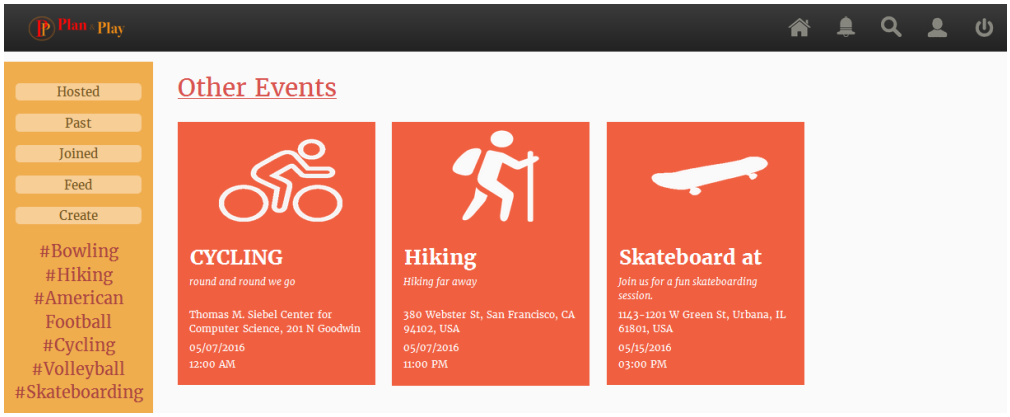

Figure 8: Home Page

3.2.9. Search by Location

Users can search for events which are going on around a particular location. By default, the system will use the user's current location.

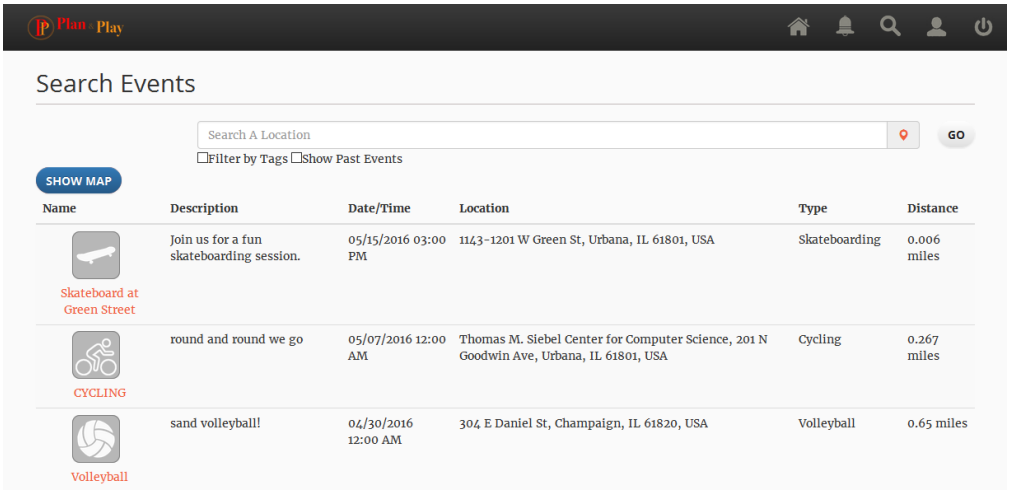

Figure 9: Search by Location

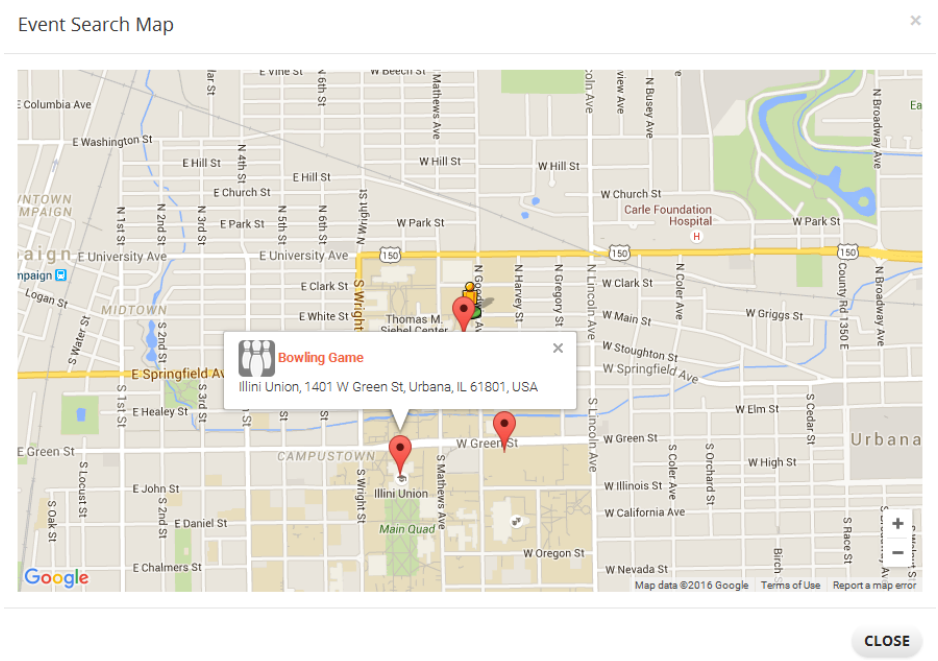

Figure 10: Event Search Map

3.2.10. Event Comments

Users can leave a comment for an event. Users can also reply to a comment.

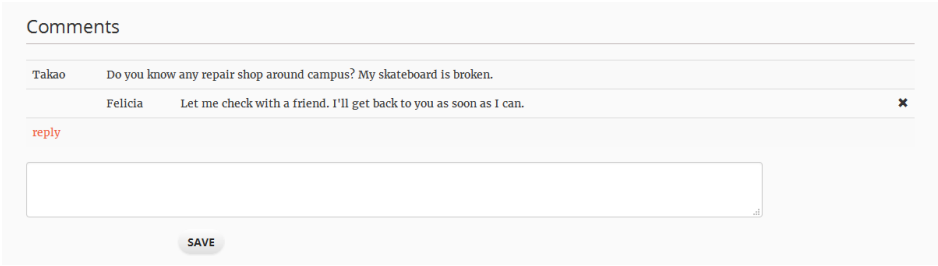

Figure 11: Event Comments

3.2.11. Event Media

Users can upload photos or videos to an event they participate in, and delete a media they previously uploaded. Users can also view a media gallery for an event, click on a specific photo to enlarge it, or click on a specific video to play it.

Figure 12: Event Media Gallery

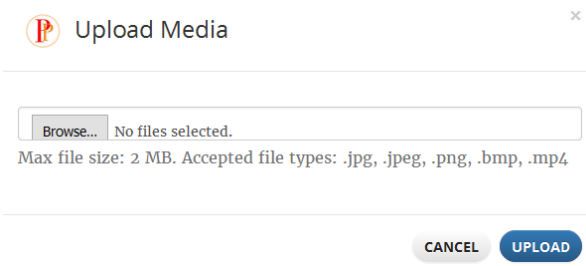

Figure 13: Upload Media

3.2.12. Recreate Past Event

Users can re-create a past event they previously created by entering a new date and time. Except for date and time, a new event will be created using the same details and participants.

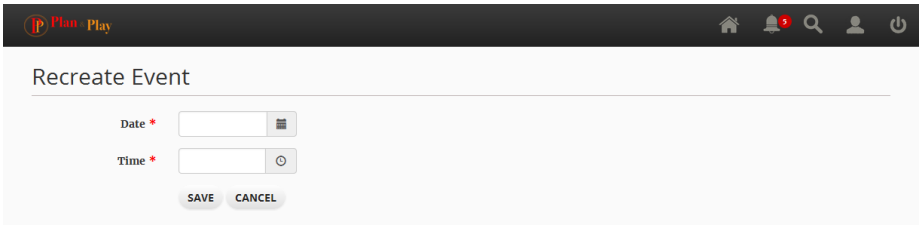

Figure 14: Recreate Event

3.2.13. View Participant Profile

Users can view the profile of a participant in an event. The page will show the participant's personal information, interests, and statistics.

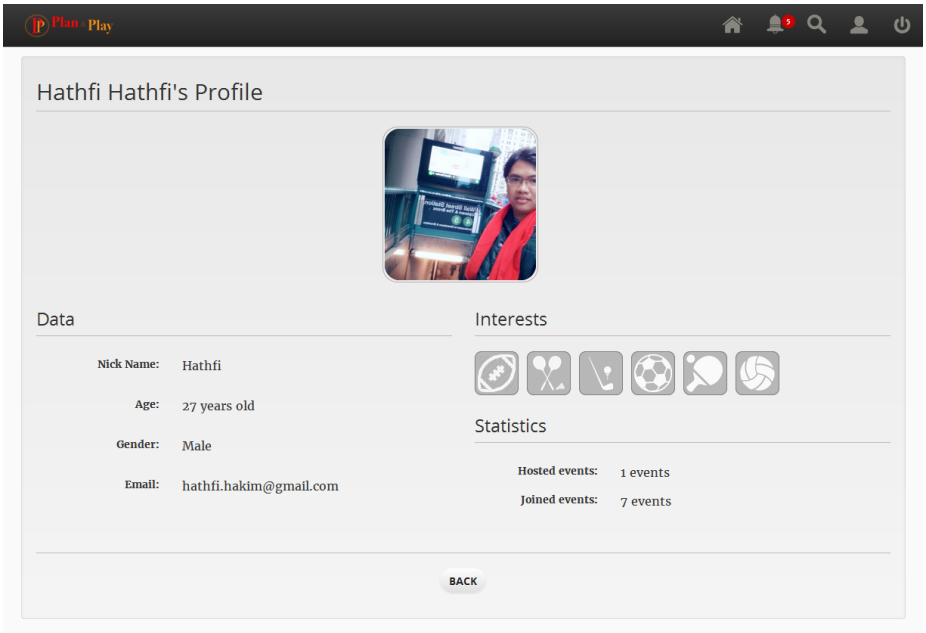

Figure 15: View Participant Profile

3.2.14. Notifications

Users will receive a notification when an event they participate in is about to begin. Users will also get a notification when other participants upload a new media to an event they previously joined.

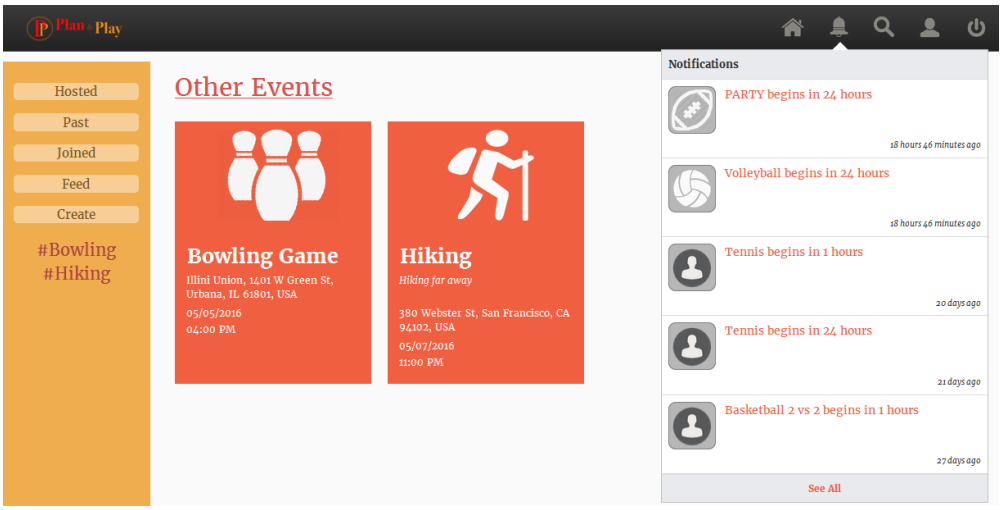

Figure 16: Notifications

4. Architecture & Design

4.1. Framework

We adopted the MVC framework to enable the separation of presentation code, application logic, and database queries. We chose the TINY framework as it provided a lightweight MVC framework with the additional benefit of pretty URLs. We also used PHPUnit to facilitate both back-end and front-end testing. For the latter case, PHPUnit provided a Selenium integration to perform automated browser testing.

Following TINY and PHPUnit folder structure, the source code is located in the `src` folder, while the test code is located in the `tests` folder. Inside the `src` folder, `application` contains PHP, HTML, and embedded Javascript code, while `public` folder contains other resources such as CSS, Javascript, font, and image files. The class names were also chosen based on TINY naming convention.

4.2. Major Components

4.2.1. Models

Model classes handle interaction with the database server. They are located in `src/application/models` and mainly contain SQL queries.

4.2.2. Views

View files take care of the data presentation. Header and footer files can be found in the `_templates` directory, while the other contents are grouped according to their modules. Views are located in `src/application/views`.

4.2.3. Controllers

Controller classes contain the application logic and handle interaction between views and models. They are located in `src/application/controllers`.

4.2.4. Helpers

Helper classes provide common utility functions. They can be found in located in `src/application/helpers`.

4.2.5. DBTestCase

As stated in section 2.3, we created a pre-determined data set for testing purposes. This data set is configured through the `getDataSet` function in `DBTestCase`.

4.3. UML Diagrams

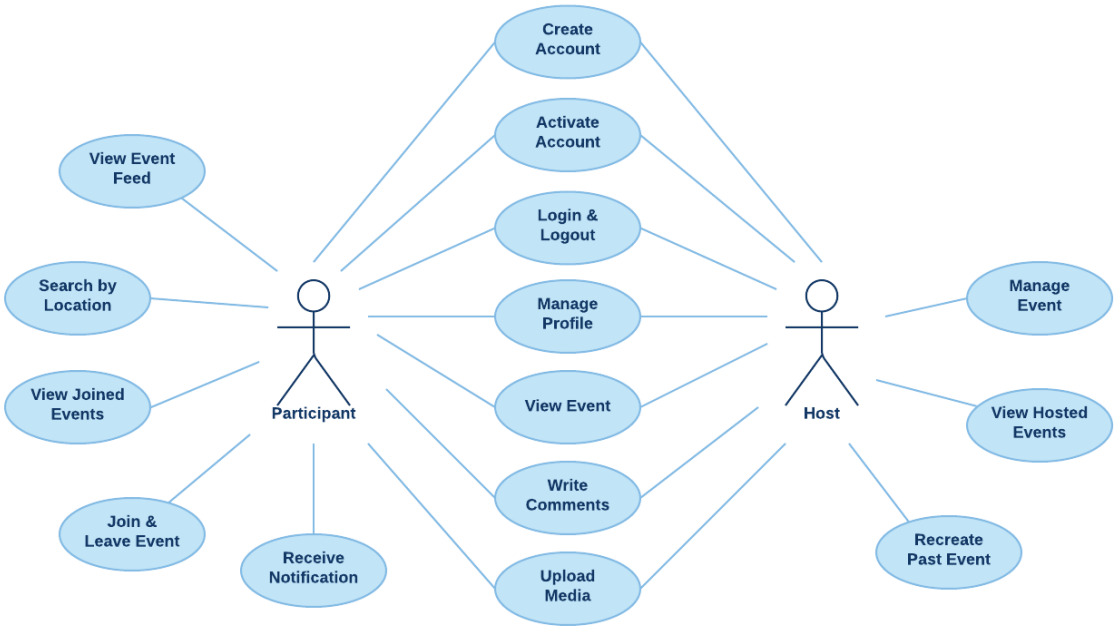

Figure 17: Use Case Diagram

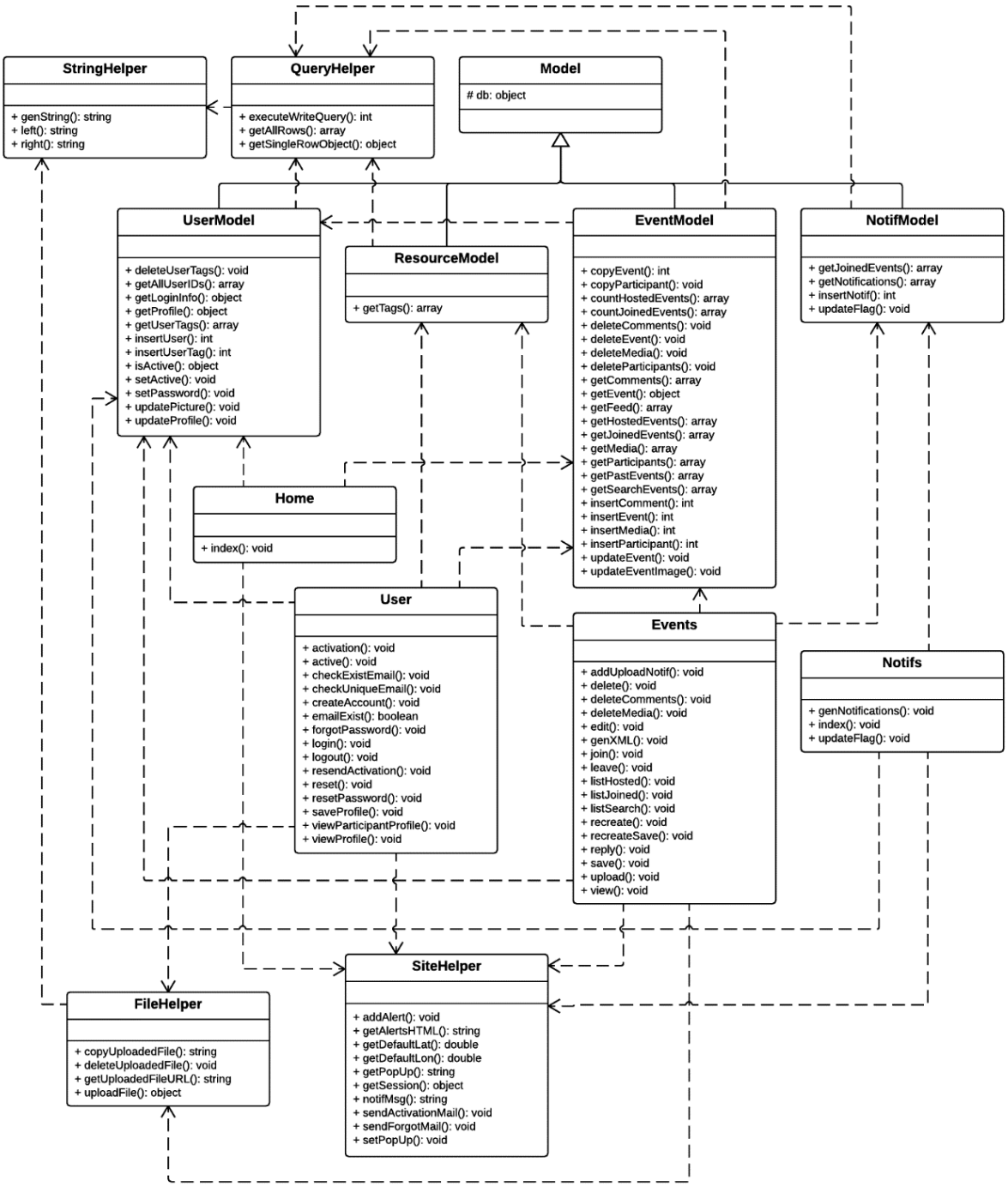

Figure 18: Class Diagram

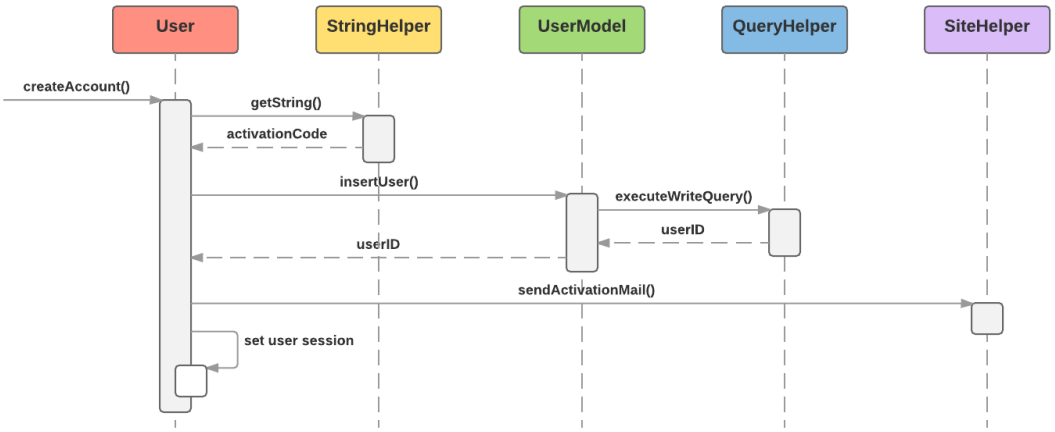

Figure 19: Create Account Sequence Diagram

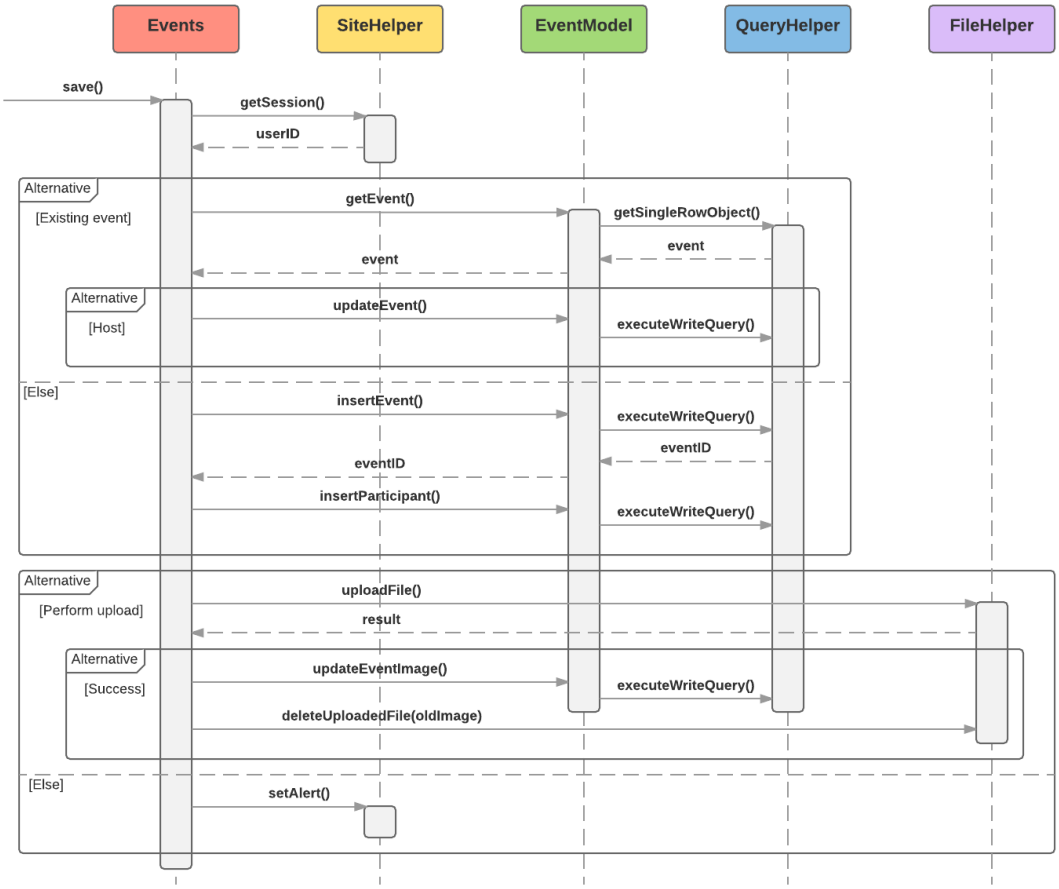

Figure 20: Manage Event Sequence Diagram

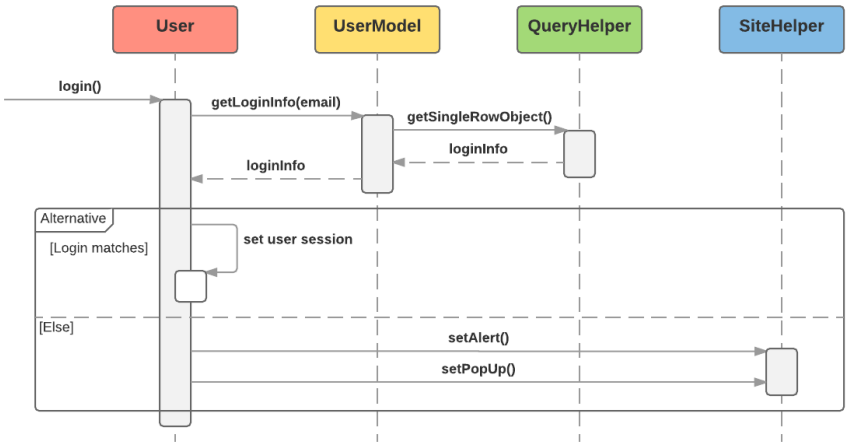

Figure 21: Login Sequence Diagram

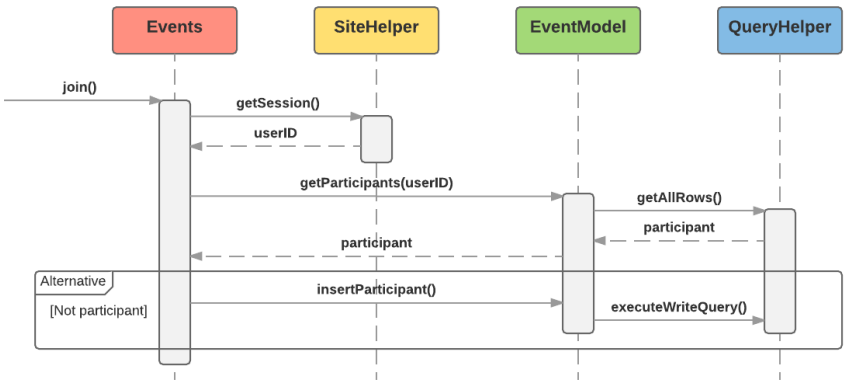

Figure 22: Join Event Sequence Diagram

5. Reflections

5.1. Divya Balakrishna

Working on this project was fun. All the team members were very helpful and encouraging. I got a chance to work with platforms I was not familiar with, which turned out to be a very good learning experience. The iteration meetings made sure the progress of the project was good. Working with a big team gave me a chance to learn to collaborate with a large team.

5.2. Felicia Chandra

I am glad I got the opportunity to learn the entire software development process, starting from requirements, design, development, testing, and documentation. The iterative process ensured we made a steady progress throughout the entire semester. Additionally, I believe we had a well-rounded team. We had members who preferred front-end, back-end, design, and even documentation. Everyone brought various skills to the table and we never had a problem finding someone to work on a particular task. Apart from gaining the experience of working in a large

team, I also learnt a great deal about unit testing, in particular GUI testing with Selenium. This project gave me a chance to experiment with things I had never done before.

5.3. Gul Mariam

I had a great time working on this project. It was a very good learning experience. I came across platforms that I had never used before and got a chance to explore. Moreover, the most useful experience was working with the team. My team was very enthusiastic and helpful. We all learnt a lot from each other and also learnt how to work well and coordinate in a big team. I am sure this experience is going to be very helpful working in the real world.

5.4. Hao Zhang

It is a great experience to work with this team. The teammates were great; they were very helpful and responsible. It was also my first time to do a project with web programming. I learnt a lot through the developing process. I am sure this experience will be useful for me in my future work.

5.5. Hathfi Hakim

This project was really interesting. Working with 7 other group members with different characters was really a challenge. Even to find a perfect time where all members could meet was not an easy task. But overall, I think we could manage all of those things. So everyone did a good job.

5.6. Hong Jae Jeon

Working on PLAN & PLAY was interesting, because I was able to learn new platform, framework, and language. All team members were helpful and responsible to finish their requirements. We all gained valuable experience from working in a large team. This project will help me achieve my professional goal.

5.7. Nicholas (Chris) Foti

I learnt a lot through working with this team. It was a great experience that taught me teamwork skills and helped me learn technical skills such as using Javascript and Google Map's API. Everyone was enthusiastic and helpful both with the project and in class.

5.8. Paarth Joshi