ZheNar
Project Report

Team Member:

Hu Hexiang

Ye Lyuyu

Cao Shan

Liu Dongyuan

3110102717 Hu Hexiang 3110000197 Ye Lyuyu
3110100638 Cao Shan 3110000420 Liu Dongyuan

Revision History

Revision Number	Date	Brief summary of changes
1.0.0	2013/06/25	First draft including Chapter Introduction/Overall Description/Specific Requirements/Maintenance
1.0.1	2013/6/26	Modify some little details about user stories.
1.0.2	2013/6/26	Add the future recommendations for the chapter 2 and modify some content of the module in chapter 3
1.0.3	2013/6/26	Rearrange the user story implementation. Add commits graph of Github repository of this project
1.0.4	2013/6/27	Add work distribution specification

Index

Chapter 1 Introduction

1.1 Purpose

This document is aimed to provide the ZheNar(Zhejiang University Navigation App for Route-finding) project with a concise definition and interface of the software, so that the developer as well as the stuff of maintenance will feel easy to understand the project. Besides there may be some other content like the team organization to illustrate how our team is working.

1.2 Software Description

The website as well as the mobile app is actually a Navigation System for the Zhejiang University. In this system, the users will be provided with the information of both places and recent events in a map, so that they will feel easy to get to the spots they want to go. Users can access these kinds of information without login.

However, if the user wants to create their places and events for the publics, they should register an account and log in with it. Then he will have the permission. What's more, after login, user can also check the top 10 events and top 10 places by clicking the "hot" button in the navigation bar. And user can also follow the event they like to make it be the hot event. The project may be a new method for School Organizations to advertise their activities, and also a new method for people to access information.

1.3 Environment

Web Branch

Item	Content
Operating System	Windows/Mac OS X
language	Python, HTML, CSS, JavaScript, MySQL
Server Software	Apache 2.2
Web Framework	Django 1.5.1

Mobile App Branch

Item	Content
Operating System	iOS
language	Objective-C,XML

Chapter 2 Team Organization

Generalization

Our team consists of four members, and in this project, **we are divided into two parts to develop the web branch and mobile branch of the application**. We organized our team as an agile team using the software process model of Extreme programming. And along the way of our software processing, we followed the principles of the XP model.

Work Distribution Specification

Web App Part:

胡鹤翔（Hu Hexiang），叶绿宇（Ye Lyuyu）

Mobile App Part:

曹珊（Cao Shan），柳东原（Liu Dongyuan）

As mentioned before, we are an agile team. So there is only a work distribution generally, there is no work distribution in details. The project will be separated into many components and issues, and each member of the team will assign their own task by themselves voluntarily.

Pair programming

As it is mentioned before, our group is formed by four members, and we are divided into two separate braches of the project. So we make our decision to try the XP process model with pair programming to ensure quality code.

Simple Design

As our user story is reasonably small, we tried to keep the design as simple as possible for the moment and don't add features that are not needed for current functionality (Only give those necessary functions with an appropriate interface).

Small releases

As we apply the simple design for the application, we have done lots of small releases for the project. According to the Git-hub statistics, we have made around over 300 commit for the project (maybe some are not meaningful and some are reverts), which proved that we have follow this principle of the XP process model

Refactoring

In the project, we first build the prototype of the application and then refactor the basic application with improvements. These make our software better along the process.

Standards of coding

We make some rules of the coding style and the system API for the web branch as well as the mobile branch, so as to make all the members easy to work.(You can find the API in '/static/api.html')

Project framework

Our project is developed based on the user stories and their corresponding implementation module, you can find the detailed information in the Chapter 3.

Future recommendations

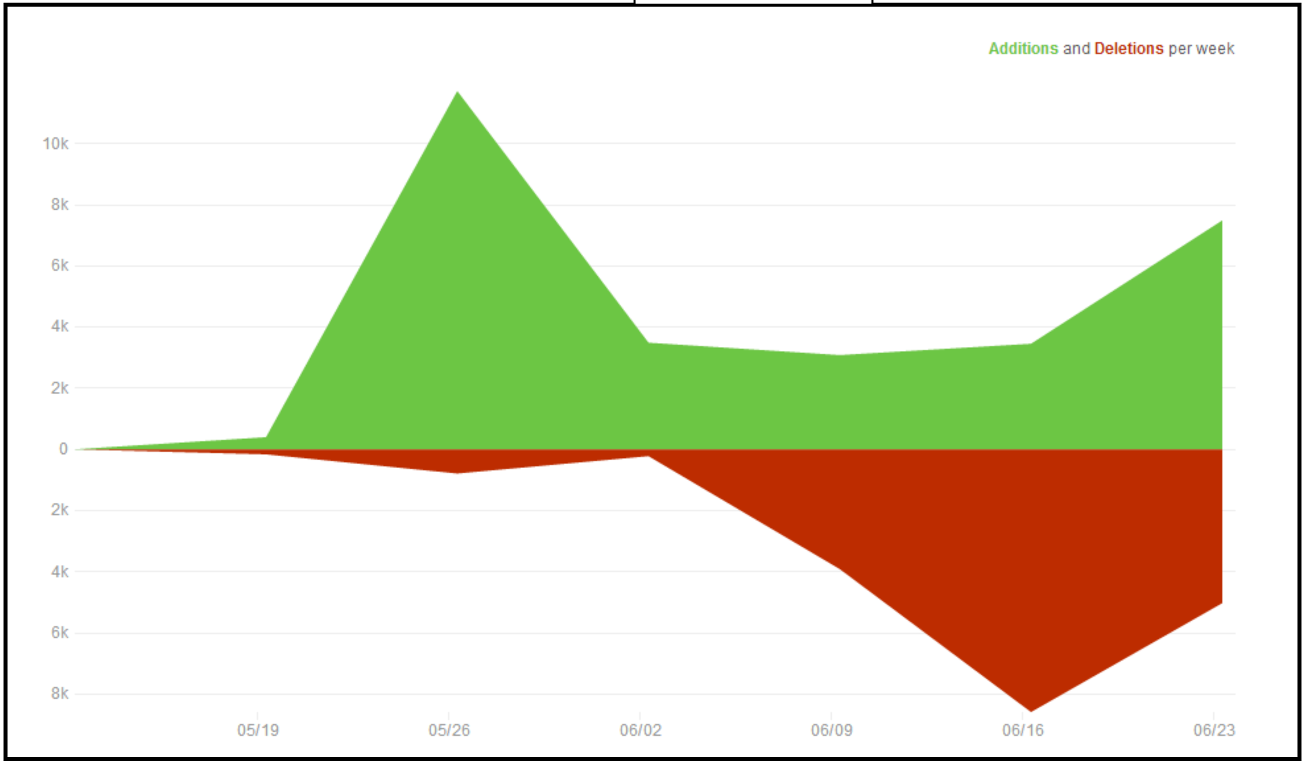
In the future, if the project is tackled again(It's of high probability), the following features will be added:

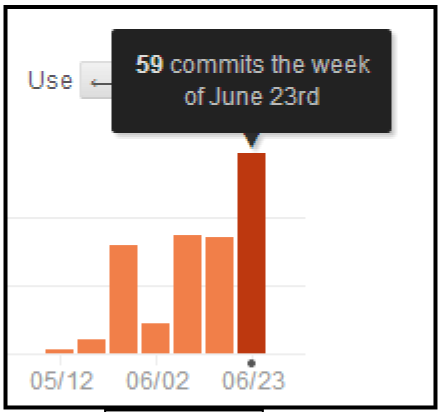
1. **A message board on the event's detail page.** This page is aim to make the user to make their comment for the events, so that the events information will be more detailed, with the events followers' communication
2. **A list of the user's followed events**. So that the user may check what events they have already followed.
3. **Notification Center.** User may be reminded before the start time of the event, so that the user will not face the problem that missing some events because they forgot it.
4. **An in-site Search for the events and places.** Make user convenient to access their interested information
5. **Ticket distribution system.** There are lots of organizations delivering their tickets for some events in the real world, which really mess up the campus sometime.(Because the waiting queue is so long and jams the street) We want to make an online ticket system with our events system, so that the user can apply for tickets online, and participate the events with their electronic tickets.
6. **Event rating system.** This feature is along with the No.5, for those who have participated in the events, they can make rate the events, so that the events will get a mark. (Unfortunately, We haven't come out with how to use these rate right now)

Chapter 3 Graph from Github

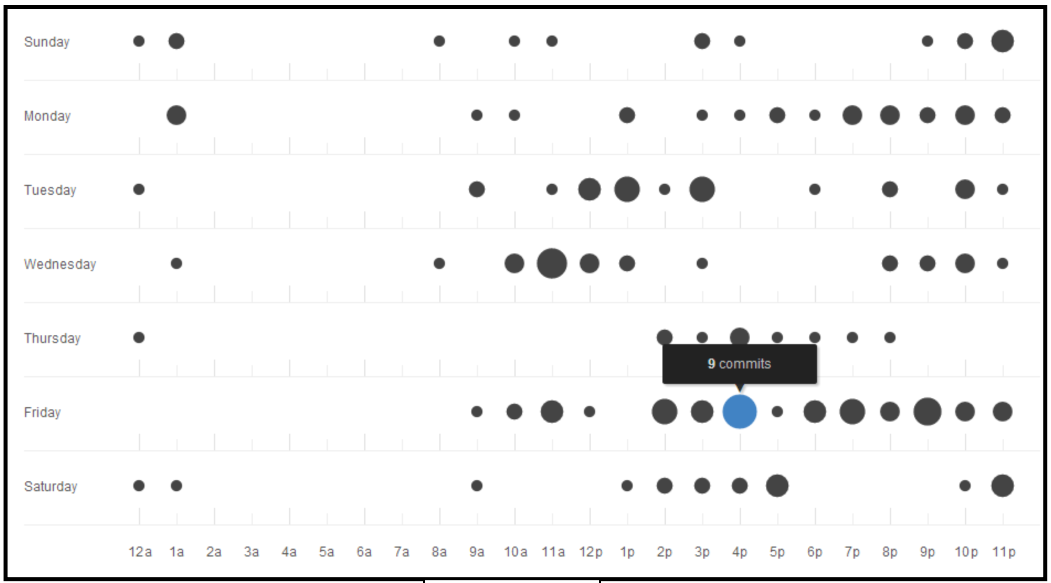
Web App

Code Frequency





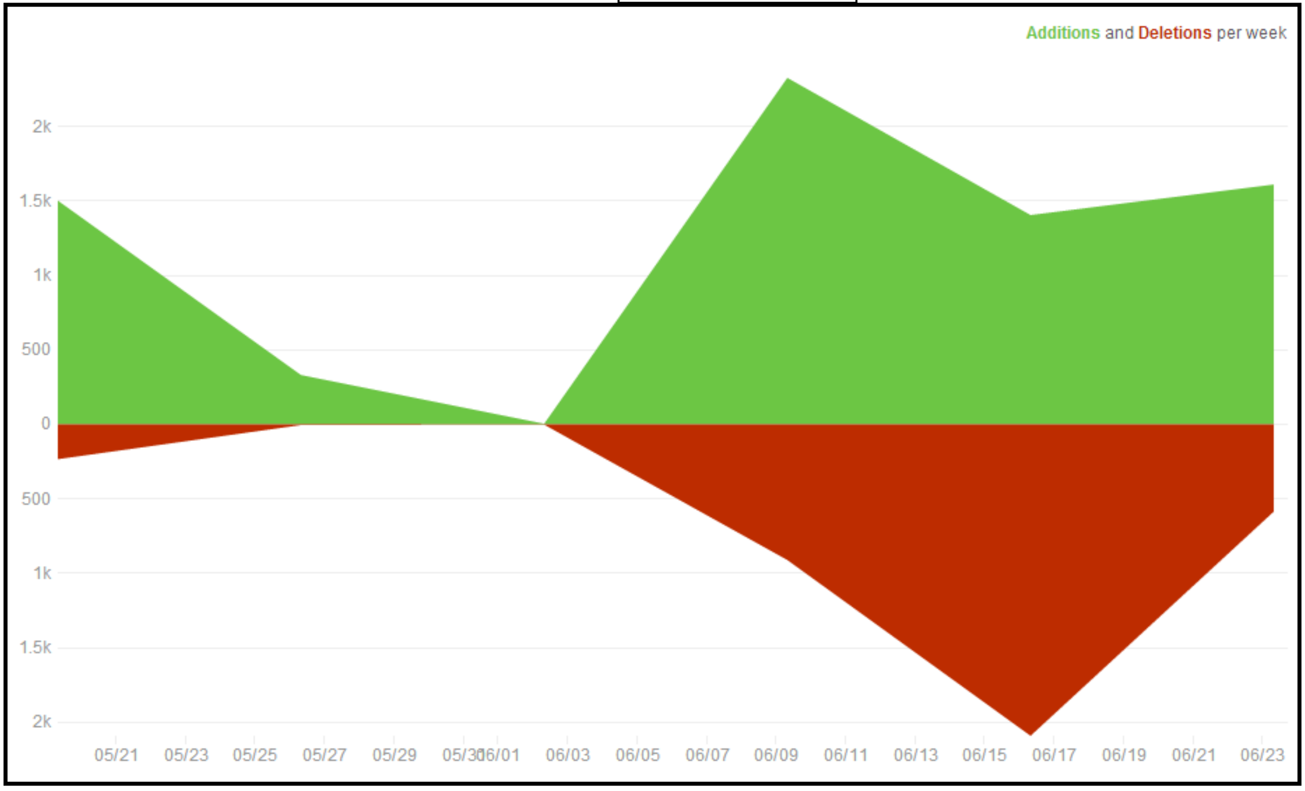
Commits

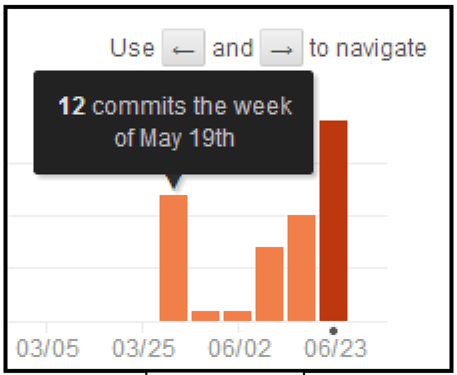


Punchcard

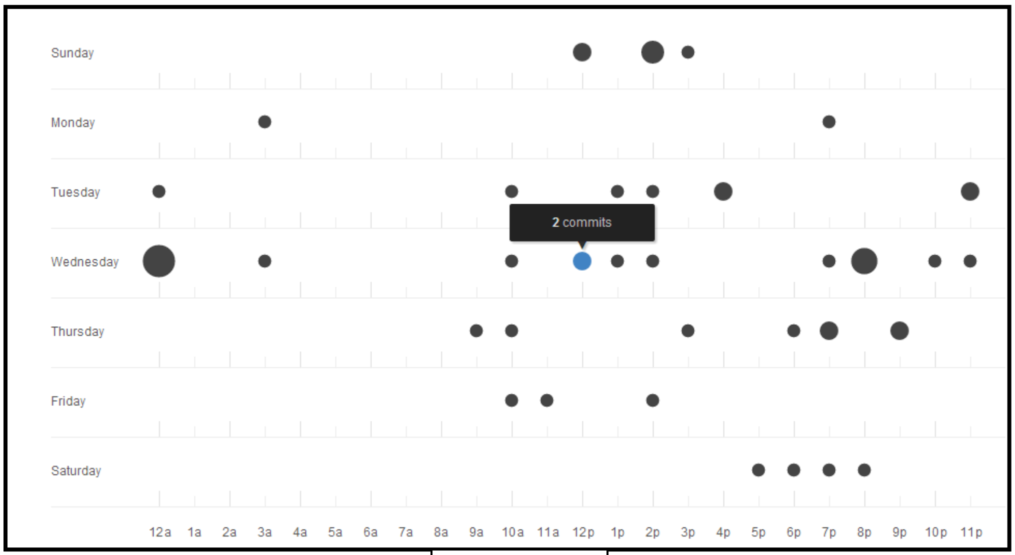
Mobile App

Code Frequency





Commits



Punchcard

Chapter 4 User Story and Implementation

Story Name	Description	Input	Output	Process

1. Register	Register the user for the application	Email Password User name	Return whether user is registered successfully. Return wrong information if the username or email has already been used.	Check if the email and user name is valid Check if the email and user name is occupied Save the user data into the database
2. Login	Login using the account	Email or Username Password	Return the session that mark the login status	Check if the email or username is correct with the password
3. Reset Password	When user forget their password, they can reset their password with this method	User's Registered Email	A email to the selected mail address	Send the user an email with Click the URL in email and it will show a interface where user can reset the password Save the password into database。
4. User manage activity	Make user able to check the events and places he created	None	The user created events list and places list	Check out the data about the user created events and places from the databases
5. Check the places and events with map	Check the places and events information in the map	None	The campus map provided by Google The marker created by the web admin and user The event information created by the web admin and user	Check out data from the database Make use of the Google Map Api to implement the map of Zijingang Campus and places marker
6. Places list	List the places which are accpeted	None	List the places information	Check out the place data from the database(filter them by their status)
7. Events list	List the events which are accepted and is not expired	None	List the events information	Check out the events data from the database(filter them by their status and expired data)

8. Check place detail	Check the detailed information of a place	Place ID(which is recorded by the UI, and can be accessed by click the button)	Detailed place information	Check out the corresponding data about the place from the database
9. Check event detail	Check the detailed information of a event	Event ID(which is recorded by the UI, and can be accessed by click the button)	Detailed event information	Check out the corresponding data about the evnet from the database
10. Create event	Create an event for the user	name places(which is created by user of web admin) detailed address description time event type	None	Check the validation of the data Add the data into the database Make the web admin to accept or reject the event request If accept, make it available to the public in the event list
11. Create event type	Create an event type for the user	name icon	None	1. Check the validation of the data 2. Add the data into the database 3. Make the web admin to accept or reject the event request 4. If accept, make it available to the public in the event type option list
12. Create place type	Create an place type for the user	name icon	None	1. Check the validation of the data 2. Add the data into the database 3. Make the web admin to accept or reject the event request 4. If accept, make it available to the public in the place type option list
13. Create place	Create a place for the user	name location(which is created using the Google map Api) description place type	None	1. Check the validation of the data 2. Add the data into the database 3. Make the web admin to accept or reject the request 4. If accept, make it available to the public in the place list
14. Edit Place	Allow user to edit the place he created	Place ID(which is recorded by the UI, and can be accessed by click the button)	The original place information will be displayed in the form	Check out the data about the user created place from the database Send it to the page Get the data user POST Modify the place information in the

		Place name Location(which is created using the Google map Api) Description Place type		database and save it
15. Edit Event	Allow user to edit the event he created	Event ID(which is recorded by the UI, and can be accessed by click the button) Event name Places(which is created by user of web admin) Detailed address Description Time Event type	The original event information will be displayed in the form	Check out the data about the user created place from the database Send it to the page Get the data user POST Modify the place information in the database and save it
16. Follow	Follow an event, and can make the event to be the hot event in the hot event list	Event ID(which is recorded by the UI, and can be accessed by click the button)	None	Add the follower information to the database Increase the number of followers
17. Unfollow	Unfollow an event, and make the number of followers to decreases	Event ID(which is recorded by the UI, and can be accessed by click the button)	None	Remove the follower information from the database decrease the number of followers
18. Check Hot List	Check the hottest places and events in the campus	None	The Top 10 events information The Top 10 places information	Check out the required data from the database Show the data to the user with the link to the detailed information page

Chapter 5 Database Design

Database Schema design

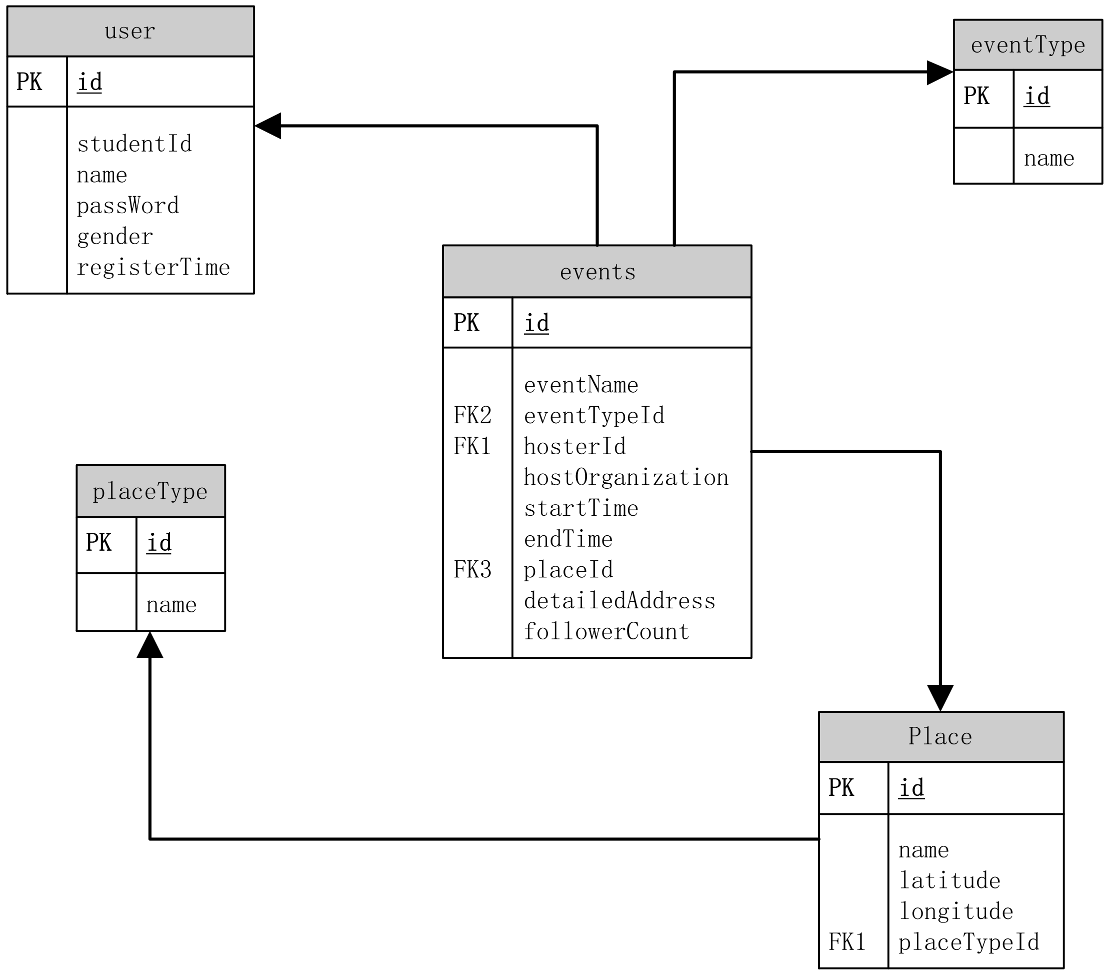


Figure 1 Database Schema

Database relation description

User Table

Name	Type	Attribute	Reference
id	INT(10)	PRIMARY KEY	
studentId	CHAR(10)	UNIQUE	
name	VARCHAR(255)	NOT NULL	
passWord	VARBINARY(255)	NOT NULL	
gender	ENUM	“F”, “M”	
registerTime	DATETIME		

Events Table

Name	Type	Attribute	Reference
id	INT(20)	PRIMARY KEY	
eventName	CHAR(30)	NOT NULL	
eventTypeId	INT(30)	FOREIGN KEY	eventType(id)
eventDescription	TEXT		
hosterId	INT(10)	FOREIGN KEY	user(id)
hostOrganization	VARCHAR(255)	NOT NULL	
startTime	DATETIME	NOT NULL	
endTime	DATETIME	NOT NULL	
placeId	INT(10)	FOREIGN KEY	place(id)
detailedAddress	TEXT	NOT NULL	
followerCount	INT(10)		

EventType Table

Name	Type	Attribute	Reference
id	INT(10)	PRIMARY KEY	
name	VARCHAR(255)	NOT NULL	

PlaceTable

Name	Type	Attribute	Reference
id	INT(10)	PRIMARY KEY	
name	VARCHAR(255)	NOT NULL	
latitude	DOUBLE	NOT NULL	
longitude	DOUBLE	NOT NULL	
placeTypeId	INT(10)	FOREIGN KEY	placeType

PlaceType Table

Name	Type	Attribute	Reference
id	INT(10)	PRIMARY KEY	
name	VARCHAR(255)	NOT NULL	