EVENT IT!

- AN EVENT REPOSITORY

OOAD PROJECT

SUBMITTED BY TEAM:

Group9.getEvent( )

Date: 04/18/2016

Team Members:

Namrata Singh

Naveenraj Palanisamy

Navya Vandanapu

Niveditha Varadha Chandrasekaran

Shravya Kuncha

INDEX

1. INTRODUCTION

1.1 Vision

To create a web application – EventIt!, a one stop event repository that helps people to know about the various events happening in and around their neighborhood. It also serves as a platform for the event hosts to promote their ticket free events.

1.2 Overview

The following functionalities are included in this system.

- Provides a platform to create events and promote them on the website
- Helps people to get to know the various events taking place
- Allows users to browse the events based on different categories
- Provides login to users to host or to register for events
- Allows users to reserve tickets for events based on the availability
- Allows the event hosts to update or delete events
- Allows users who have reserved tickets to rate the events once the event has happened
- Ensures that notifications are sent for every update
- Eliminates any inappropriate content present

2. PROJECT SCOPE

The following functionalities were addressed in each of the iteration.

Iteration 1:

- Creating the events and submitting it for approval.
- Management of worklist by the admin to approve or reject the event publish request.
- Browsing the events that are approved by the admin and published on the web site.

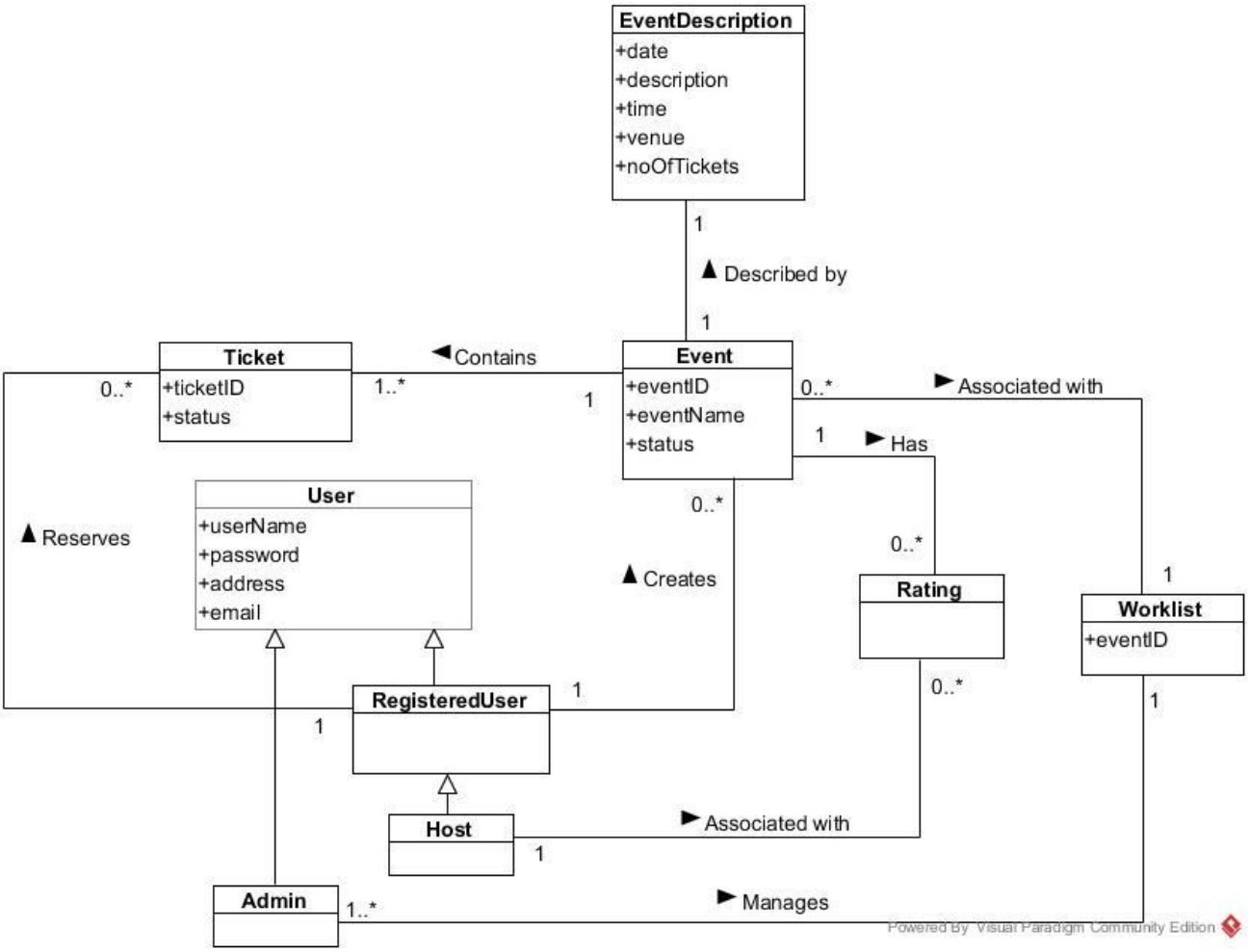
Iteration 2:

- Browsing the events based on category and date.
- Reserving tickets for events by the registered users.
- Cancelling the reserved tickets by the registered users.
- Backlogs from Iteration 1.

Iteration 3:

- Rating the past registered events and associating the rating to the host.
- Updating the event details by the host.
- Deleting the event by the host.
- Register to the application.
- Login into the application.
- Email Notifications to the user.
- Backlogs from Iteration 2.

3. DOMAIN MODEL



4. DESIGN SPECIFICATION

4.1 Functional requirements:

- Admin should be able to approve or reject event publish requests.
- Registered users should be able to create the events and submit them to the admin for approval.
- Registered users should be able to browse events, reserve tickets if needed and rate the events.
- General users should be able to browse events and sign up for the website if needed.
- Registered user should be able to rate the past event which the user has attended.

4.2 Non Functional requirements:

- *Usability:* This application provides usability by having a user friendly user experience. As the application is hosted in the web, anyone with Internet can access the website.

- *Security:* The application is password restricted for registered users and admin, so that general users and other anonymous users will not be able to see the controls that are available for the admin and registered users.

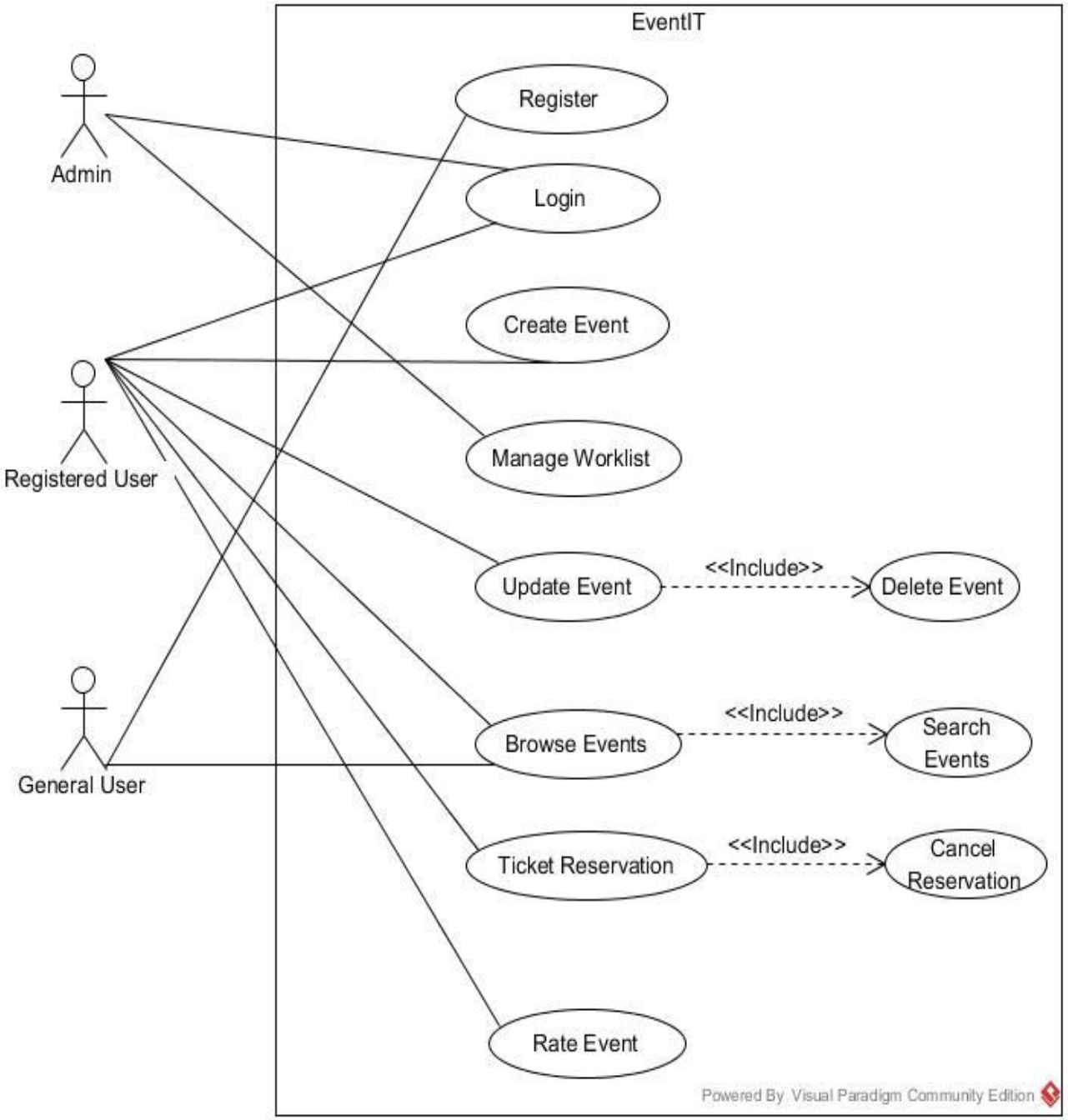
- *Deployment:* The application is deployed in web to get worldwide access.

- *Changeability:* The application changes its layout based on the screen size.

4.3 Constraints and Limitations

- The application will not let any user to create and promote their events until they sign up for the website.
- The application will not let any registered users to modify or delete the events which are not created by them.

5. USE CASE DIAGRAM



6. USE CASE TEXT

ACTORS: General User, Registered User, Admin
SYSTEM: The EventIt System

6.1 USE CASE UC1: CREATE EVENT

Scope: EventIt application
Level: User goals
Primary Actor: Registered user
Stakeholders and Interests:

Registered user: Can create, promote their events and browse and reserve tickets for published events

General user: Can browse published events

Pre-conditions: The user should be logged into the application
Post-conditions: The event created is sent for approval.
Main Success Scenario (Basic Flow):

0. The system displays the Home Page of EventIt web application
1. The user selects Create event option.
2. The system displays a form to input details like Event name, Event Location, EventDate, etc.
3. The user fills the form and submits it for the approval
4. The system sends an email notification to the admin for event approval request
5. The user receives a message that the event request has been sent to the admin

Extensions (or Alternative Flows):

3a. The user enters an event name which is already present

1. The system prompts a message about the duplicate event name

1a. The user ignores the message and submits the form with same event name

1b. The user changes the event name and submits the form

6.2 USE CASE UC2: MANAGE WORKLIST

Scope: EventIt application
Level: User goals **Primary**
Actor: Admin **Stakeholders**
and Interests:

Registered user: Can create, promote their events and browse and reserve tickets for published events

General user: Can browse published events

Pre-conditions: The admin should be logged into the application
Post-conditions: The worklist is updated and the approved events are published on the website
Main Success Scenario (Basic Flow):

0. The System should display the home page of the admin
1. The admin clicks on the worklist button
2. The system displays the event approval requests
3. The admin clicks on the event for its details
4. The system displays the details from the create event form submitted by the user
5. The admin approves or rejects the event
6. The system sends a notification to user regarding the event approval
7. The user receives a notification regarding the event approval

Extensions (or Alternative Flows):
5a. The admin approves the event
1. The system publishes the event on the website

6.3 USE CASE UC3: BROWSE EVENTS
Scope: EventIt application
Level: User goals
Primary Actors: General user, Registered user
Stakeholders and Interests:
Registered user: Can create, promote their events and browse and reserve tickets for published events
General user: Can browse published events
Pre-conditions: User must visit the web application
Post-conditions: List of all the events is displayed
Main Success Scenario (Basic Flow):
0. The system displays the Home page of the EventIt web application
1. The user clicks on the browse events tab
2. The system displays the list of all the upcoming events
3. The user clicks on the event of his/her choice
4. The system displays the event page with the event details
5. The user receives the information about the selected event
Extensions (or Alternative Flows):
2a. The system displays the message "No upcoming events"
2b. Includes UC6 Search events

6.4 USE CASE UC4: TICKET RESERVATION
Scope: EventIt application
Level: User goals
Primary Actor: Registered User
Stakeholders and Interests:
Registered user: Can create, promote their events and browse and reserve tickets for published events
General user: Can browse published events
Pre-conditions: The user should be logged into the application and select an event
Post-conditions: The event ticket is reserved for the user
Main Success Scenario (Basic Flow):
0. The system should display the particular event to which the user wants to reserve a ticket
1. The user clicks on the reserve ticket button on the event page
2. The system displays a confirmation form for ticket reservation
3. The user confirms by filling his details in the form and submits it
4. The system sends an email to the user regarding the confirmation of ticket reservation
5. The user receives a message "Ticket is reserved"
Extensions (or Alternative Flows):
*a. At any time system fails,
To support recovery, ensure the system's state is not changed and all the events are recovered from any step of the scenario.
2a. The system displays a message "Tickets are sold out"

6.5 USE CASE UC5: CANCEL TICKET RESERVATION

Scope: EventIt application
Level: User goals
Primary Actor: Registered User
Stakeholders and Interests:

Registered user: Can create, promote their events and browse and reserve tickets for published events

General user: Can browse published events

Pre-conditions: The user should have reserved a ticket for an event
Post-conditions: The reserved ticket is cancelled for the user
Main Success Scenario (Basic Flow):
1. The user selects the Ticket reservation history button
2. The system displays two tabs – Upcoming Events and Past Events on the reservation history page
3. The user selects the Upcoming events tab
4. The system displays all the events to which he has reserved a ticket
5. The user selects the particular ticket to cancel
6. The user clicks on the cancel ticket reservation button
7. The system sends an email notification to the user regarding the ticket cancellation
8. The user receives the message "Reserved ticket is cancelled"

Extensions (or Alternative Flows):
*a. At any time system fails,
To support recovery, ensure the system's state is not changed and all the events are recovered from any step of the scenario.
4a. The system shows a message "Ticket registration finalized. Cannot cancel the ticket"

6.6 USE CASE UC6: SEARCH EVENTS

Scope: EventIt application
Level: User goals
Primary Actor: General User, Registered User
Stakeholders and Interests:

Registered user: Can create, promote their events and browse and reserve tickets for published events

General user: Can browse published events

Pre-conditions: The user should select browse events tab
Post-conditions: The events are sorted and displayed based on the selected filter
Main Success Scenario (Basic Flow):
1. The user selects the filter by which events should be sorted and displayed
2. The system displays the list of events based on the selected filter
3. The user browses the events based on the search results

Extensions (or Alternative Flows):
*a. At any time system fails,
To support recovery, ensure the system's state is not changed and all the events are recovered from any step of the scenario.

6.7 USE CASE UC7: RATE EVENT

Scope: EventIt application
Level: User goals
Primary Actor: Registered User
Stakeholders and Interests:

Registered user: Can create, promote their events and browse and reserve tickets for published events

General user: Can browse published events

Pre-conditions: The user should be logged in and must have attended one event
Post-conditions: The rating for that particular event will be updated
Main Success Scenario (Basic Flow):

0. The system displays the Home page of EventIt application
1. The user selects Reservation History tab
2. The system displays user's Reservation history page
3. The user selects the Past Events tab in his Reservation history page
4. The system displays all the events which the user has attended
5. The user selects the particular event he wants to rate
6. The system displays the rate event page asking the user to rate it
7. The user rates the event and submits it

Extensions (or Alternative Flows):

*a. At any time system fails,
To support recovery, ensure the system's state is not changed and all the events are recovered from any step of the scenario

4a. The system prompts the user that he has rated this event once previously

6.8 USE CASE UC8: REGISTER

Scope: EventIt application
Level: User goals
Primary Actor: General User
Stakeholders and Interests:

Registered user: Can create, promote their events and browse and reserve tickets for published events

General user: Can browse published events

Pre-conditions: The user should be on the home page of Event It application
Post-conditions: The user will be registered into the application
Main Success Scenario (Basic Flow):

1. The user selects the Sign Up button
2. The system displays a form to input required information such as Username, Password etc.
3. The user fills in the form and submits it to become a registered user

Extensions (or Alternative Flows):

*a. At any time system fails,
To support recovery, ensure the system's state is not changed and all the events are recovered from any step of the scenario

6.9 USE CASE UC9: LOGIN

Scope: EventIt application
Level: User goals
Primary Actor: Registered User
Stakeholders and Interests:
Registered user: Can create, promote their events and browse and reserve tickets for published events
General user: Can browse published events
Pre-conditions: The user should be on the home page of Event It application
Post-conditions: The user will be logged into the application
Main Success Scenario (Basic Flow):
1. The user fills his username and password and clicks on Login button
2. The system displays the home page specific to the user
3. The user can browse on the website and reserve or cancel tickets for events

Extensions (or Alternative Flows):
*a. At any time system fails,
To support recovery, ensure the system's state is not changed and all the events are recovered from any step of the scenario

6.10 USE CASE UC10: UPDATE EVENT

Scope: EventIt application
Level: User goals
Primary Actor: Registered User
Stakeholders and Interests:
Registered user: Can create, promote their events and browse and reserve tickets for published events
General user: Can browse published events
Pre-conditions: The user should be logged into the Event It application
Post-conditions: The event will be edited based on the changes provided by the user
Main Success Scenario (Basic Flow):
0. The system displays the Home page of EventIt application
1. The user clicks on My Profile tab
2. The system displays user's Profile page
3. The user selects the particular event he wants to update in Hosted Events
4. The system displays the event form for the modifications to be done
5. The user edits the information content he wishes to change and clicks save
6. The system updates the Event information based on the changes made by the user
7. The user receives a message " Event information updated"

Extensions (or Alternative Flows):
*a. At any time system fails,
To support recovery, ensure the system's state is not changed and all the events are recovered from any step of the scenario

6.11 USE CASE UC11: DELETE EVENT

Scope: EventIt application

Level: User goals

Primary Actor: Registered User

Stakeholders and Interests:

Registered user: Can create, promote their events and browse and reserve tickets for published events

General user: Can browse published events

Pre-conditions: The user should be logged into the Event It application

Post-conditions: The event will be deleted from the Events list

Main Success Scenario (Basic Flow):

0. The system displays the home page of EventIt web application
1. The user selects My Profile tab
2. The system displays user's Profile page
3. The user selects the particular event he wants to delete in Hosted Events
4. The system displays the event form
5. The user clicks delete button
6. The system prompts the user to confirm the deletion
7. The user confirms the deletion by clicking Yes button
8. The system updates the Events page by deleting this particular event
9. The user receives a message " Event deleted from the Events List"

Extensions (or Alternative Flows):

*a. At any time system fails,

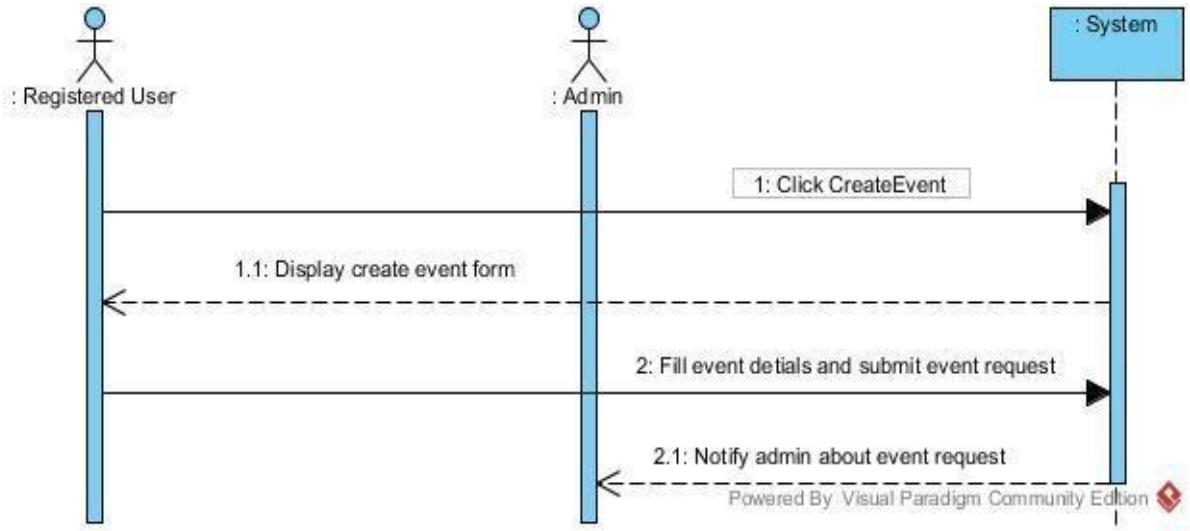
To support recovery, ensure the system's state is not changed and all the events are recovered from any step of the scenario

4a. The user clicks No button,

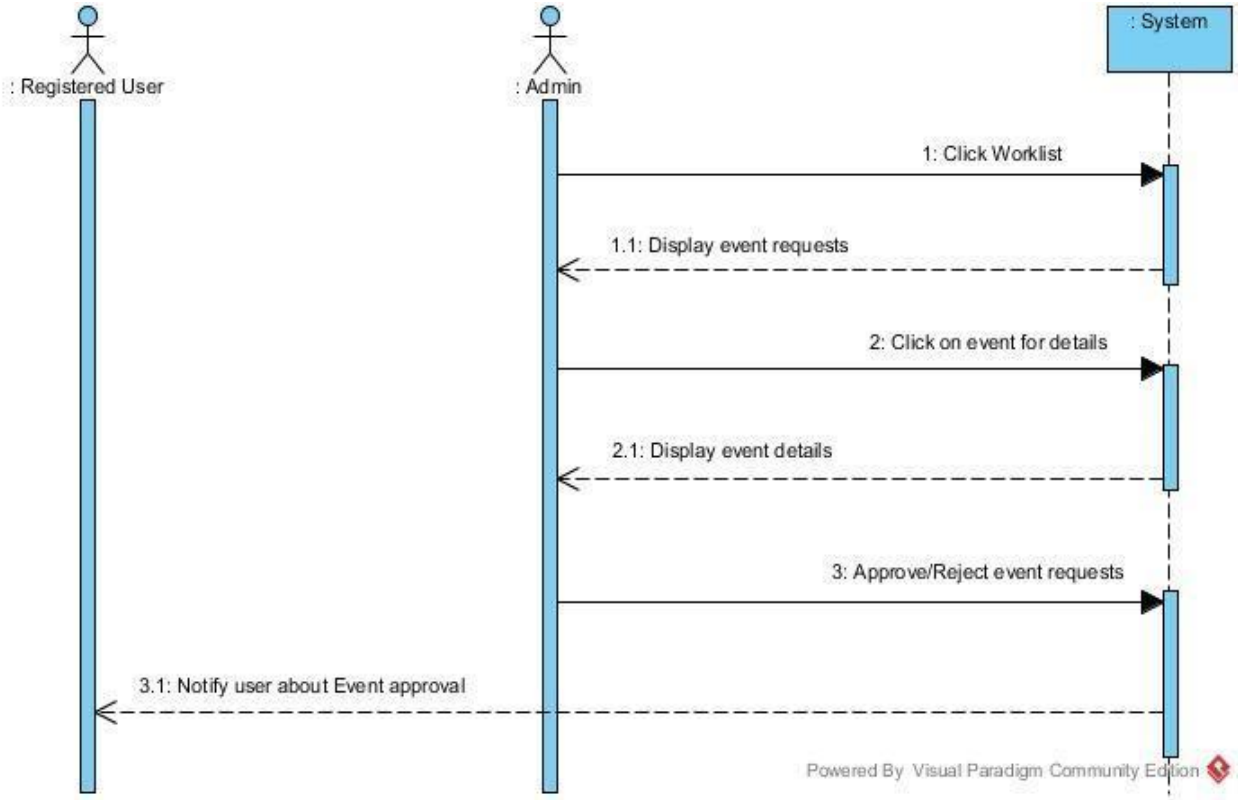
The system displays a message "The Event is not deleted"

7. SYSTEM SEQUENCE DIAGRAM

7.1 CREATE EVENT

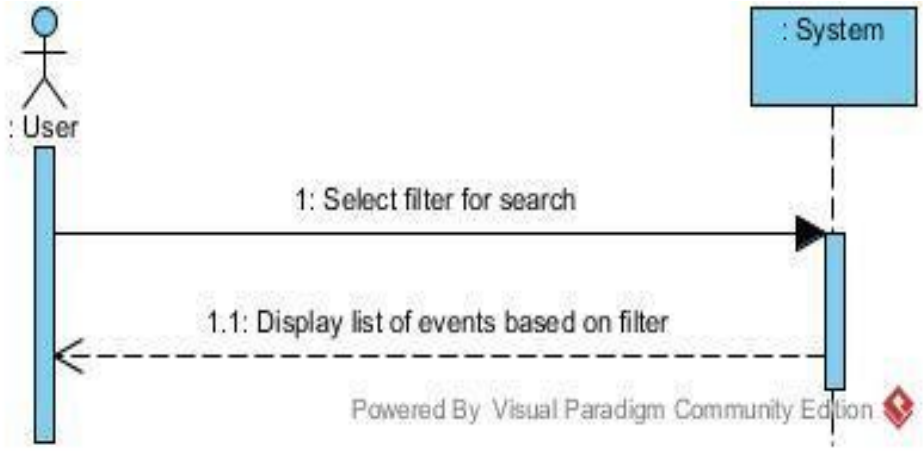


7.2 MANAGE WORKLIST

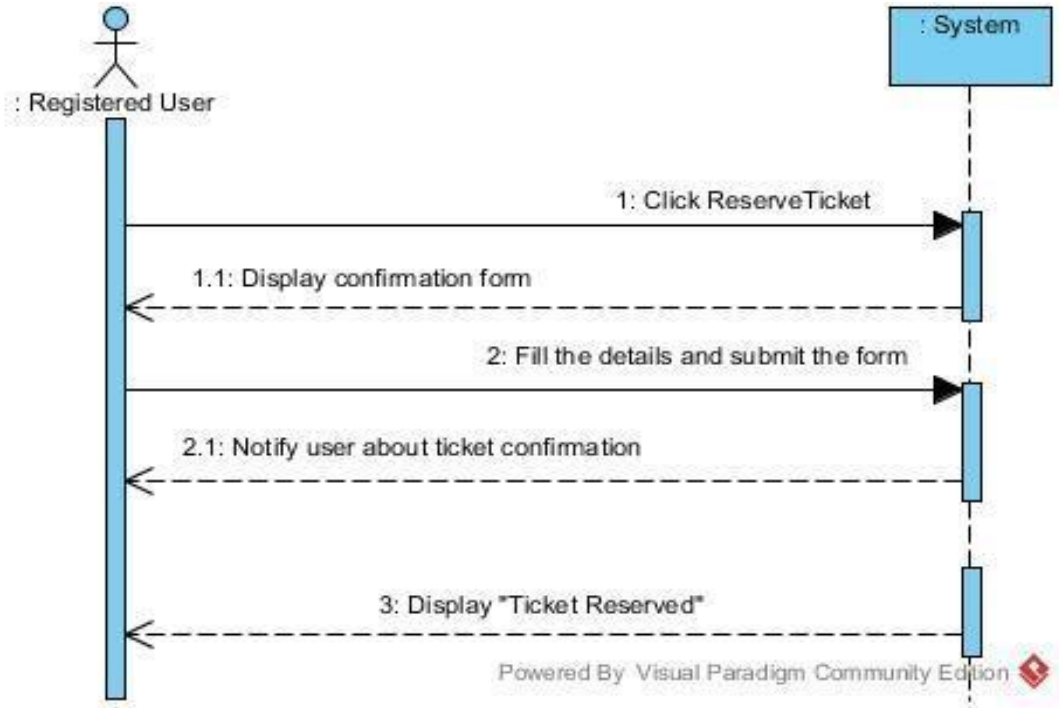


7.3 BROWSE EVENTS

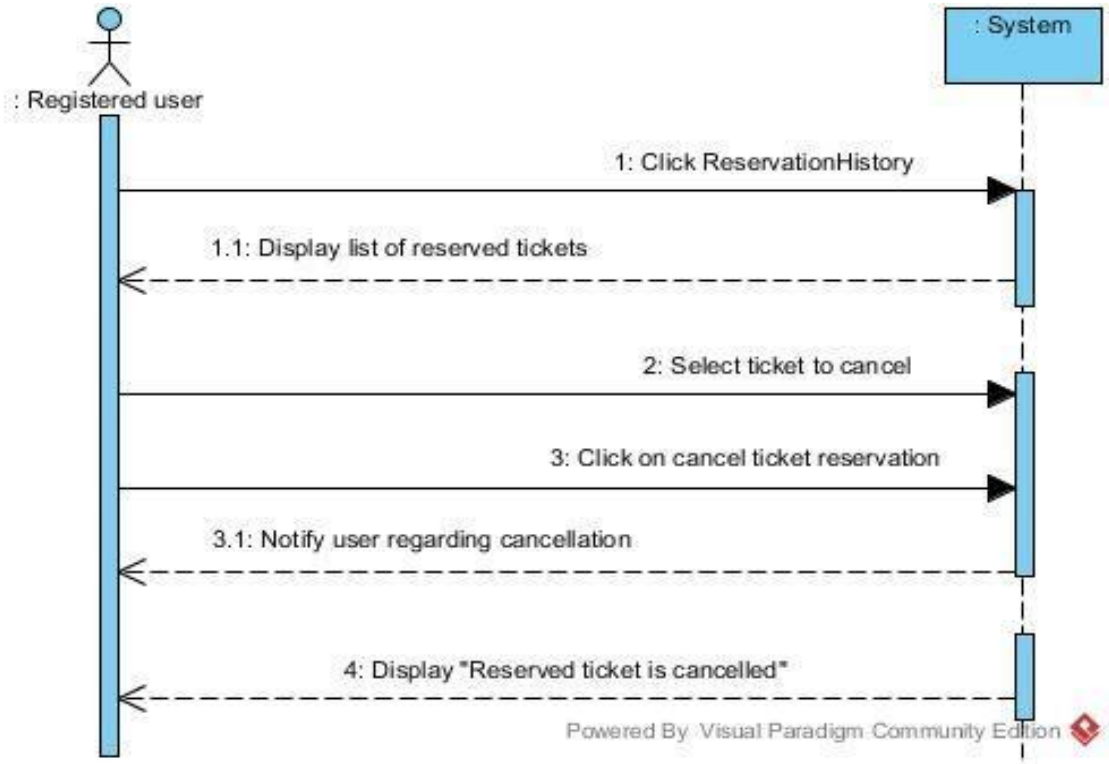
7.4 SEARCH EVENTS



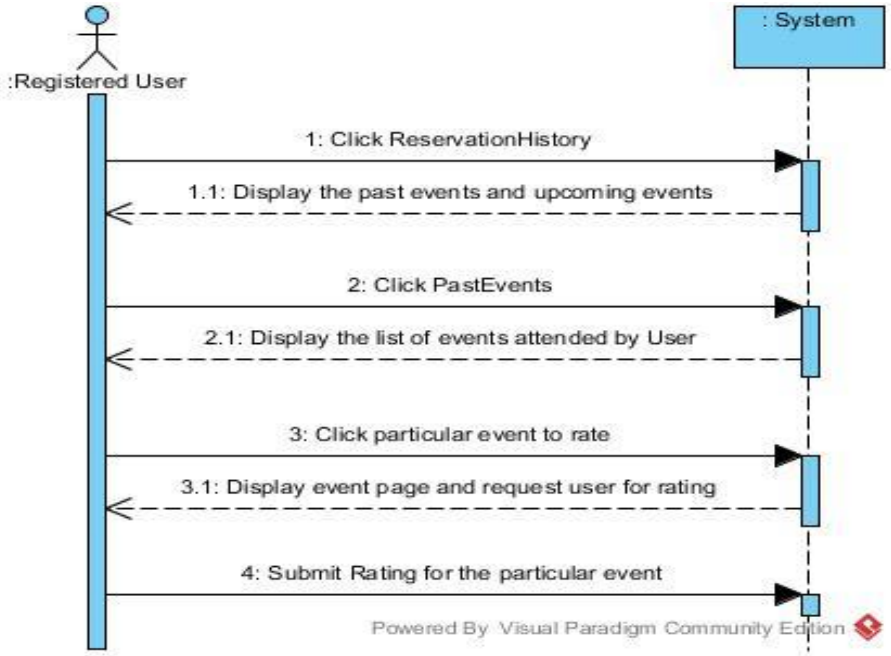
7.5 TICKET RESERVATION



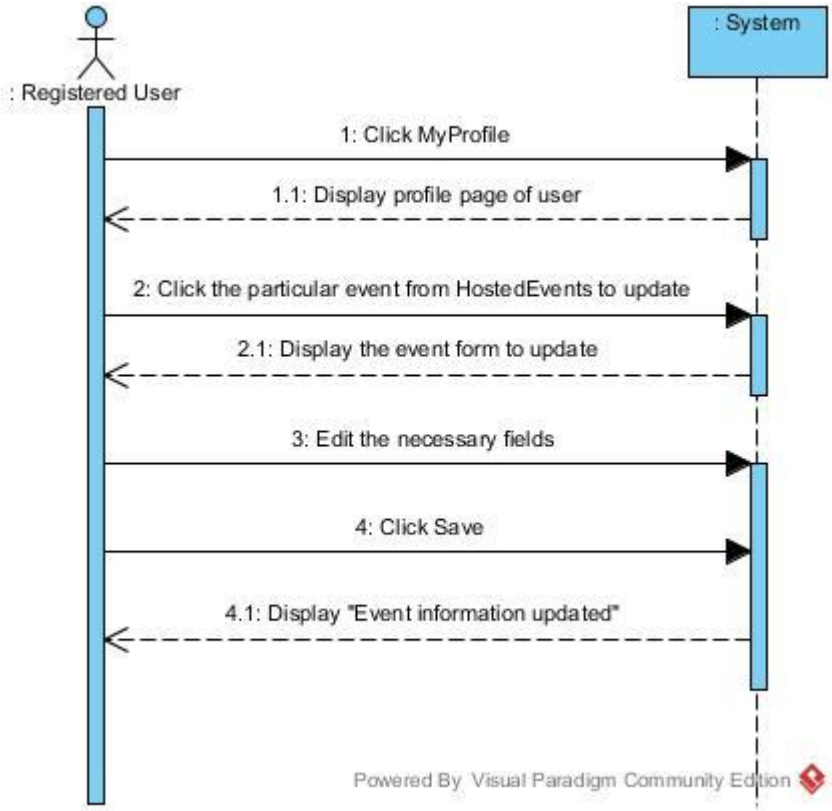
7.6 CANCEL TICKET RESERVATION



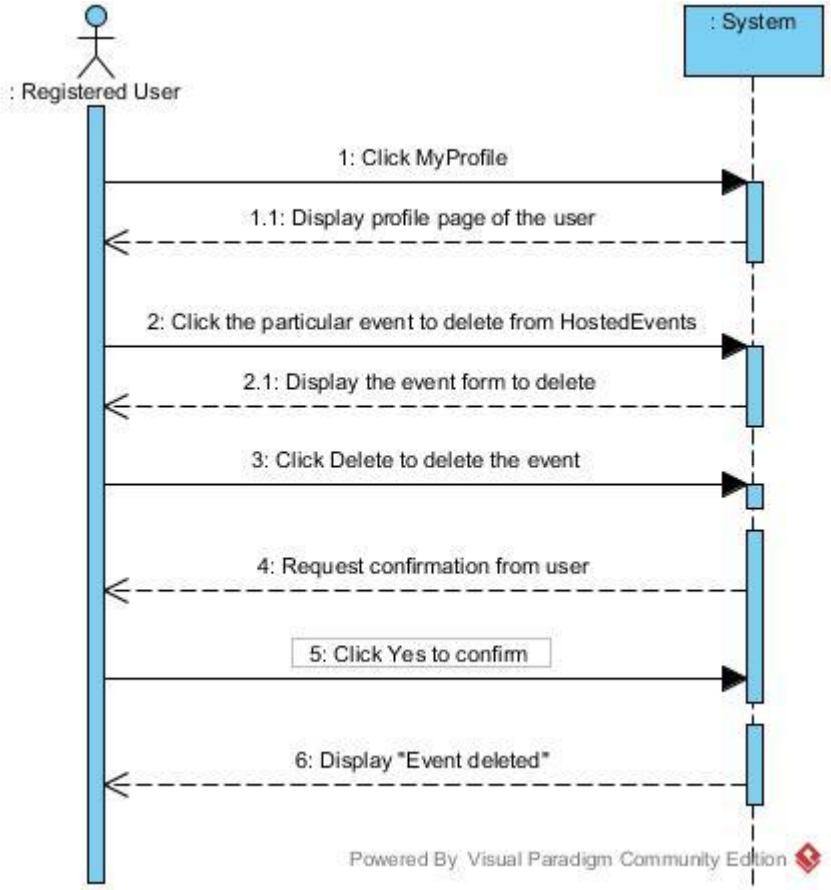
7.7 RATE EVENT



7.8 UPDATE EVENT



7.9 DELETE EVENT



8. OPERATION CONTRACT

Contract 1: Create Event
Operation: createEvent (EventName: varchar, Decr: varchar, Venue: varchar, Date: date)
Cross Reference: Use case: Create Event
Preconditions: none
Post conditions: Event instance E was created
Attributes of E were initialized
Work List instance W was created
E and W were associated

Contract 2: Manage Work List
Operation: updateWorkList(Event: varchar)
Cross Reference: Use case: Create Event
Preconditions: There is a Create event underway
Post conditions: Attributes of W were modified
Work instance W was updated

Contract 3: BrowseEvents
Operation: getallevents()
Cross Reference: Use case: Browse Events
Preconditions: none
Post conditions: E.display() became true

Contract 4: SearchEvents
Operation: getallevents()
Cross Reference: Use case: Search Events
Preconditions: There is a Browse event underway
Post conditions: Check instance C was created
Events E were associated with C
E.display() became true

Contract 5: ReserveTicket
Operation: Ticket()
Cross Reference: Use case: Ticket Reservation
Preconditions: There is at least one Ticket with T.status as false
Post conditions: Ticket instance T was created
T was associated with E
T.ticketId became Ticket ID
T.status became true

Contract 6: Cancel Ticket
Operation: getticket()
Cross Reference: Use case: Ticket Reservation
Preconditions: There should be a Ticket T for an event with T.status as true
Post conditions: T.status became false

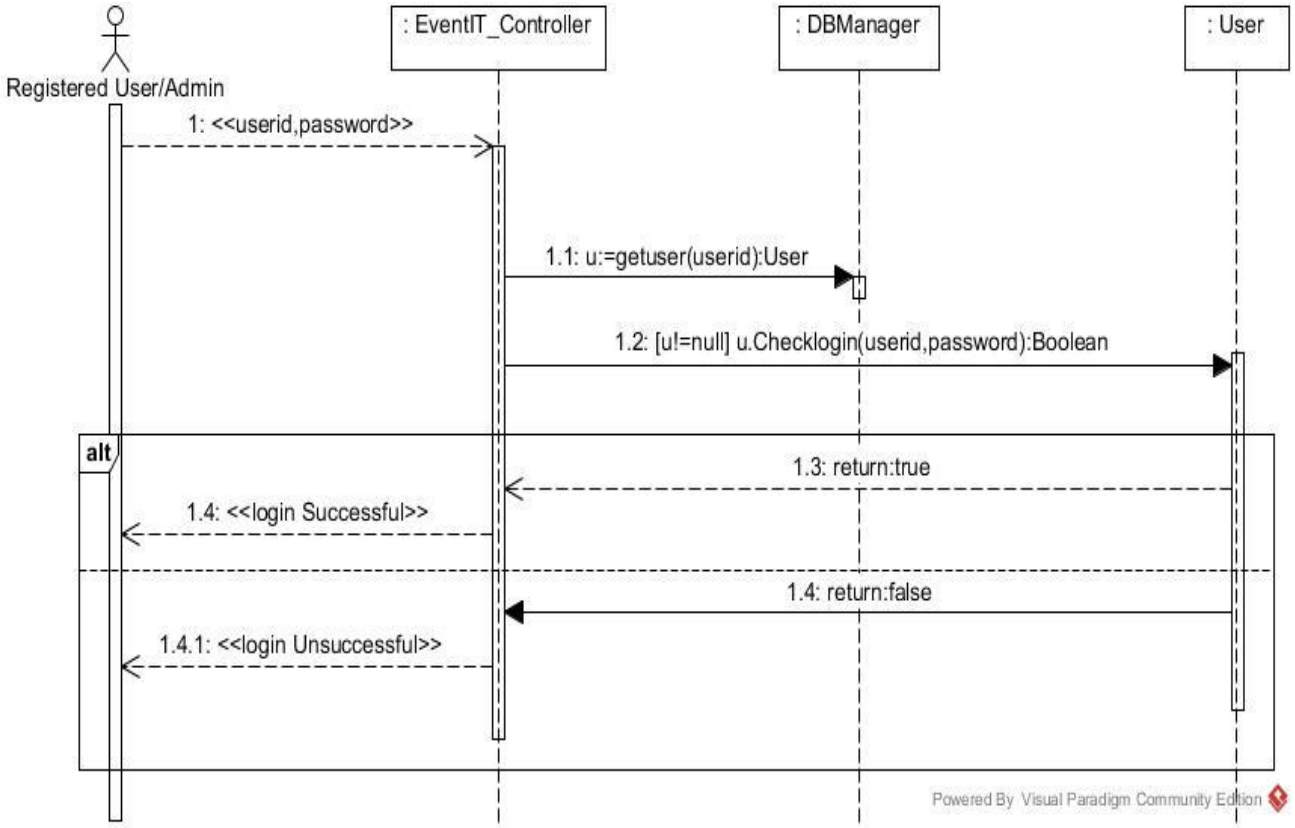
Contract 7: RateEvent
Operation: updateRating(rate: int)
Cross Reference: Use case: Rate Event
Preconditions: There is a Ticket T associated with Event E and an Event with E.rating
Post conditions: E.rating is updated
E.hostRating is updated

Contract 8: UpdateEvent
Operation: changeDetails(desc: varchar, date: Date , time: Time, venue: varchar)
Cross Reference: Use case: Update Event
Preconditions: There is an Event E with at least one attributes
Post conditions: Attributes of E were updated

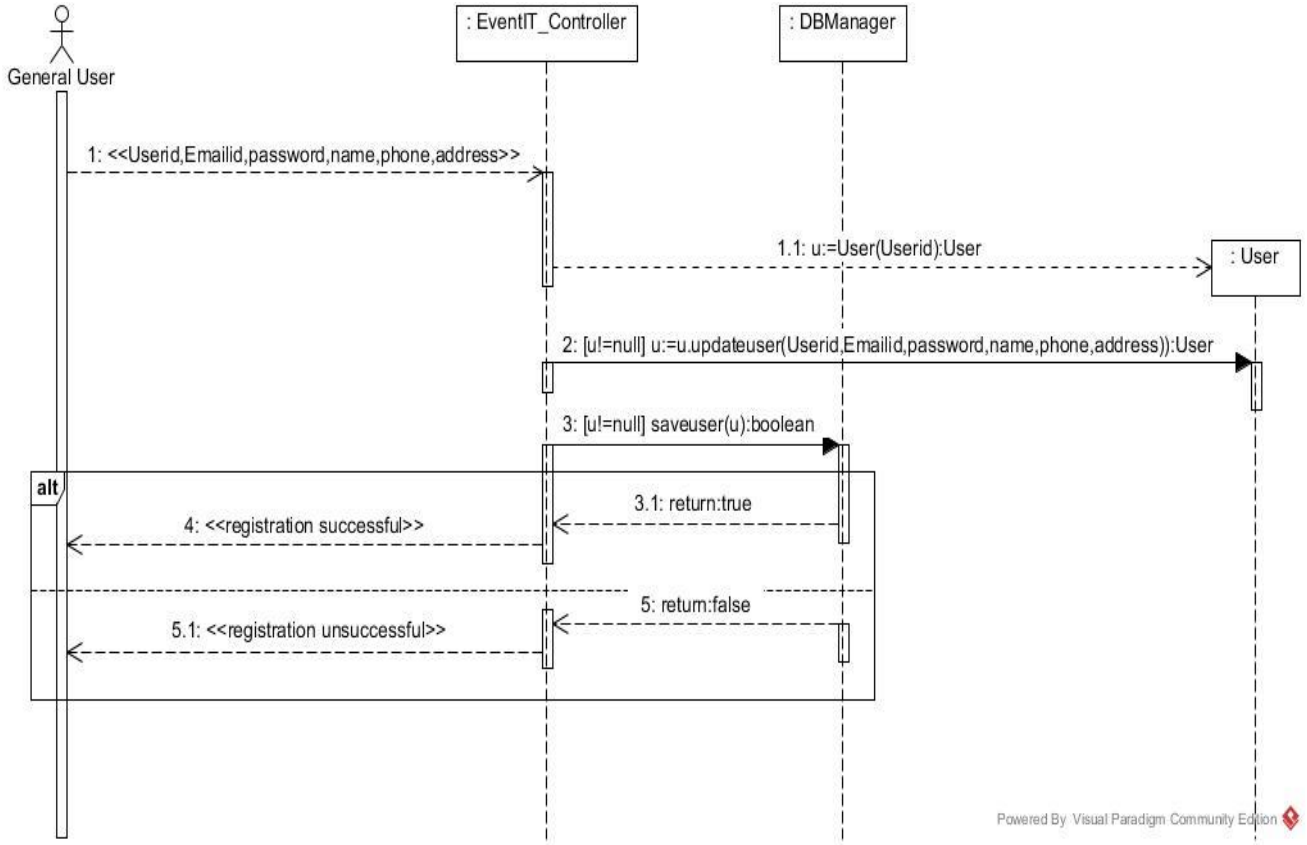
Contract 9: DeleteEvent
Operation: changeStatus(eventId: varchar)
Cross Reference: Use case: Update Event
Preconditions: There is an Event E with E.deleteStatus as False
Post conditions: E.deleteStatus became True

9. INTERACTION DIAGRAMS

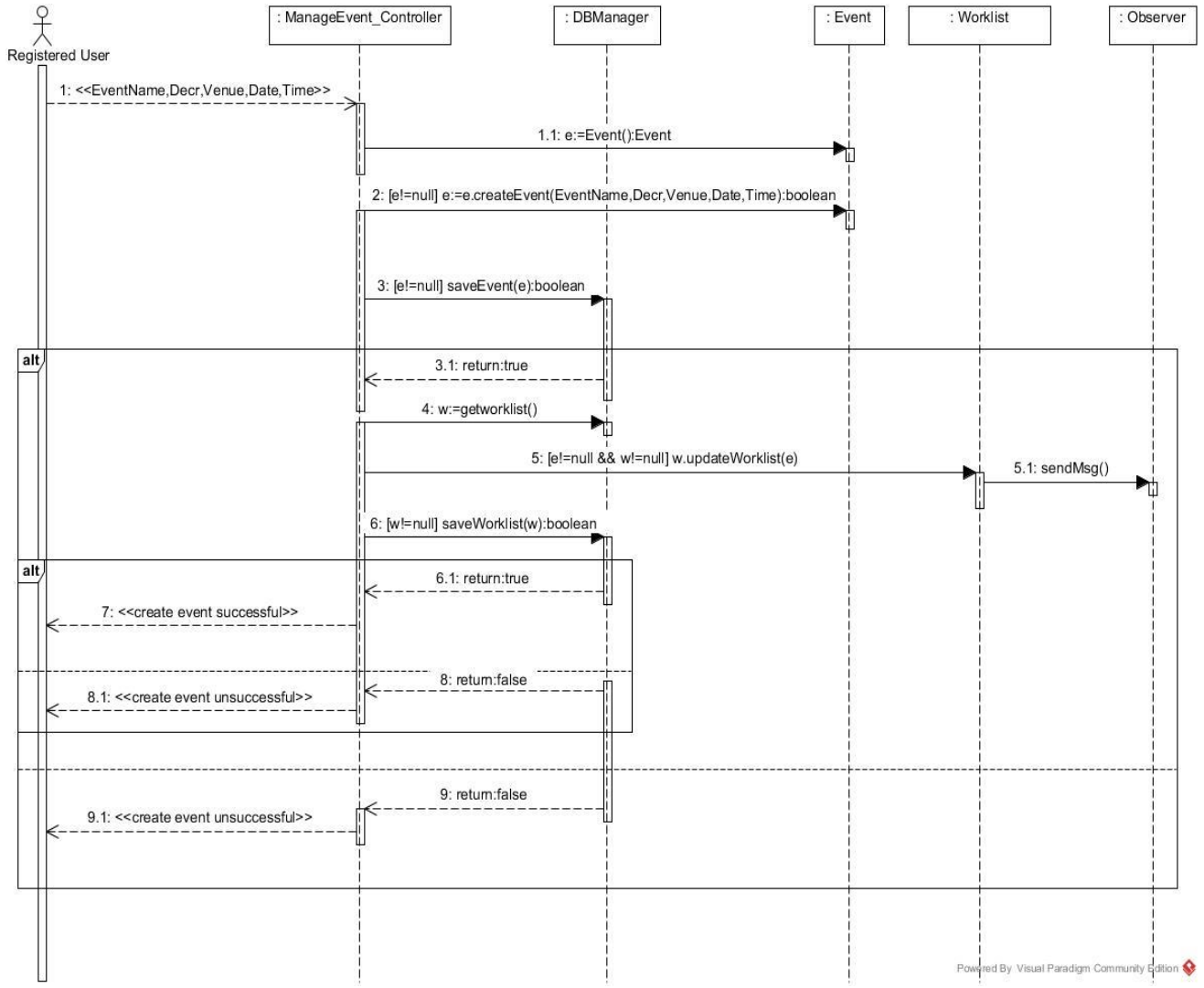
9.1 LOGIN



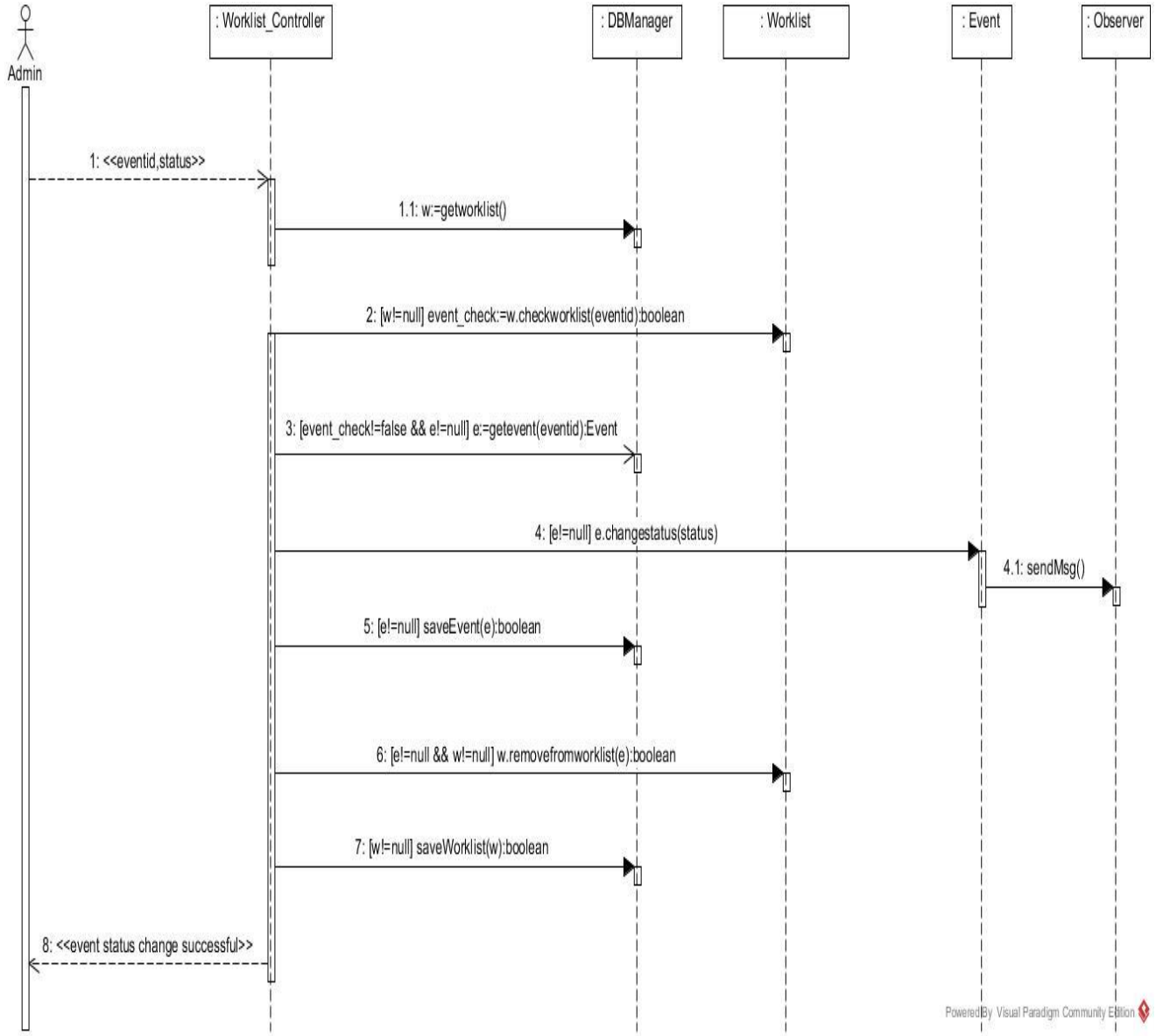
9.2 REGISTER


General User
: EventIT_Controller
: DBManager
1: <<Userid,Emailid,password,name,phone,address>>
1.1: u:=User(Userid):User
: User
2: [u!=null] u:=u.updateuser(Userid,Emailid,password,name,phone,address)):User
3: [u!=null] saveuser(u):boolean
alt
3.1: return:true
4: <<registration successful>>
5: return:false
5.1: <<registration unsuccessful>>
Powered By Visual Paradigm Community Edition

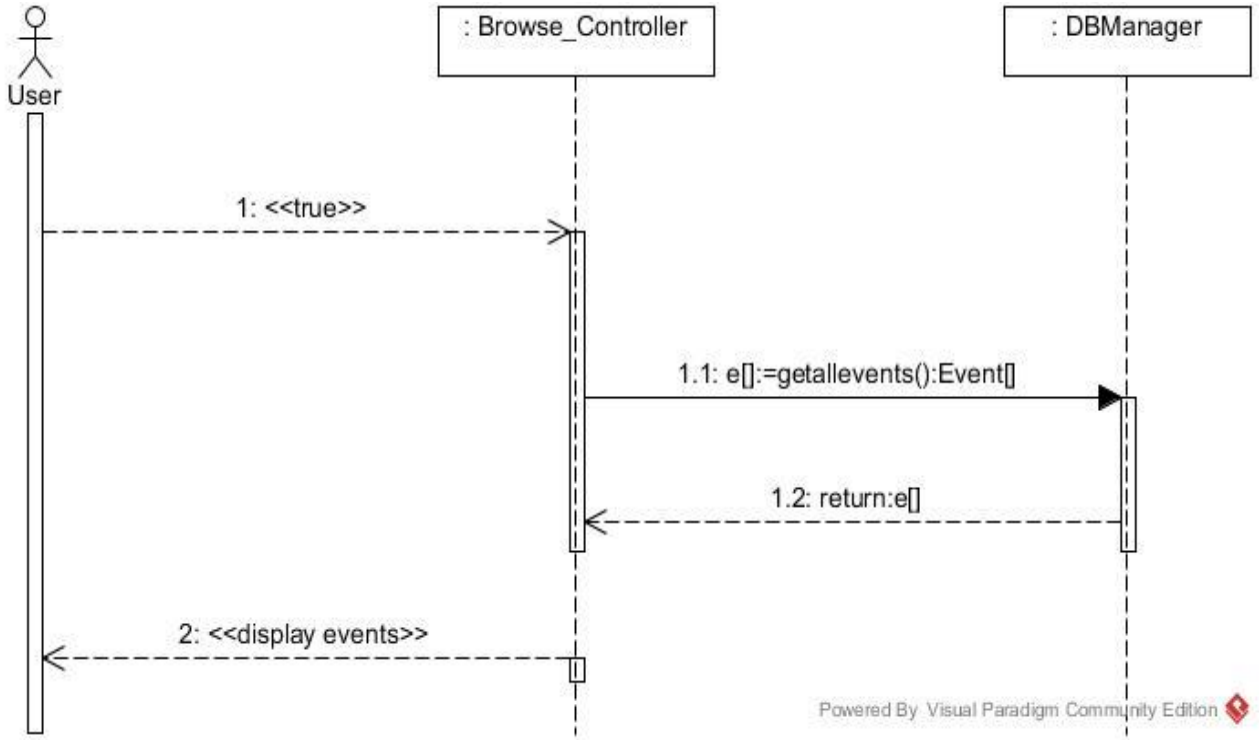
9.3 CREATE EVENT



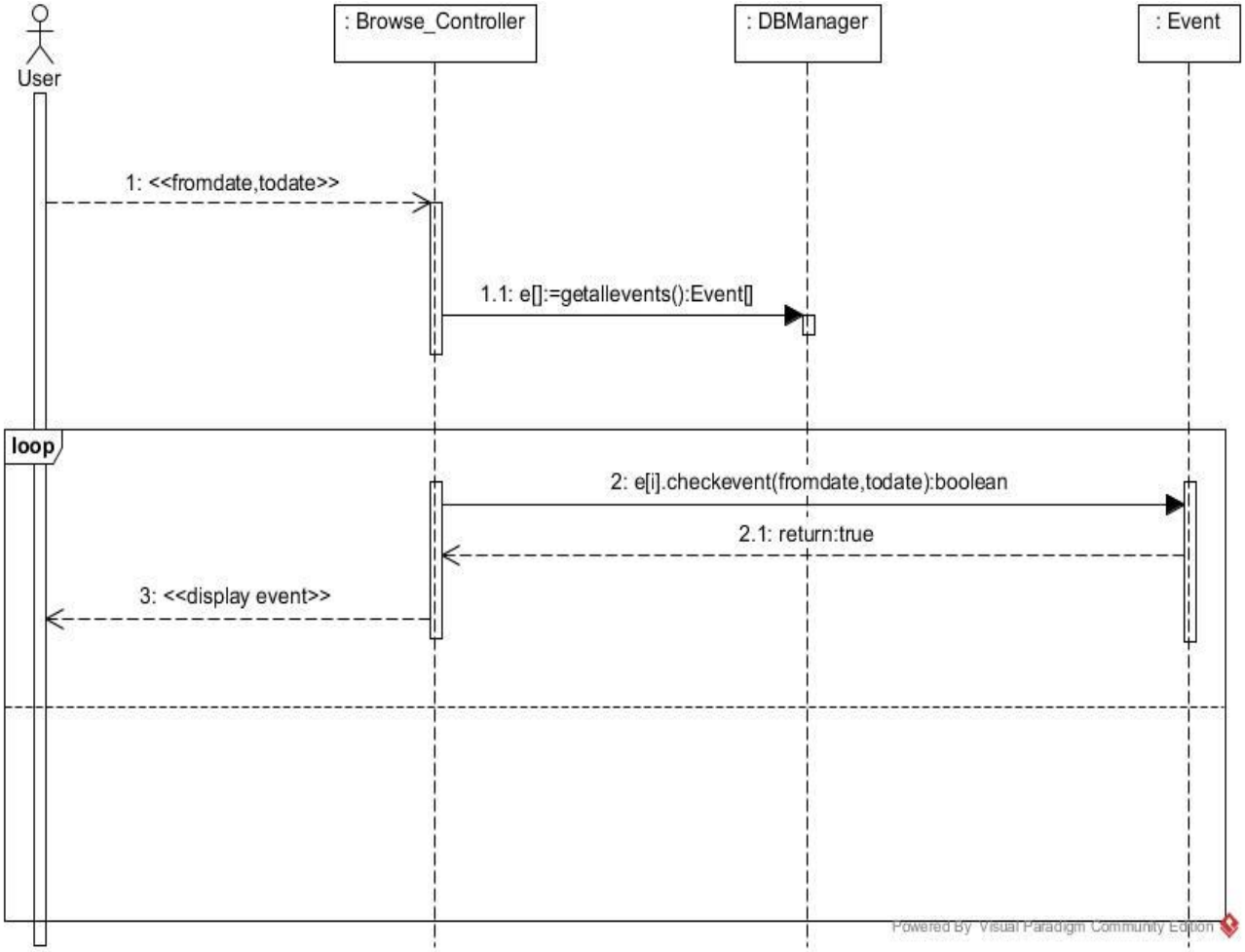
9.4 MANAGE WORKLIST



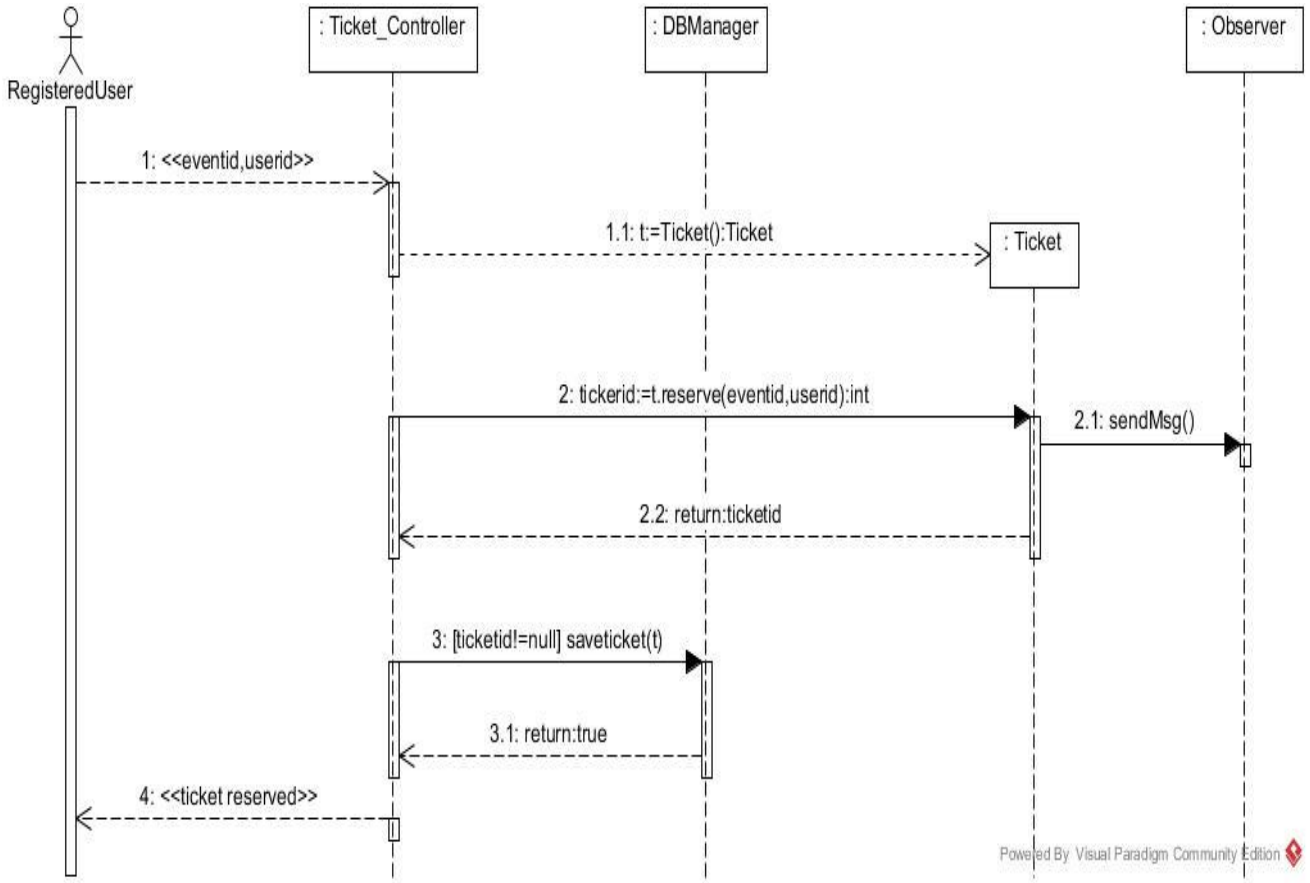
9.5 BROWSE EVENTS



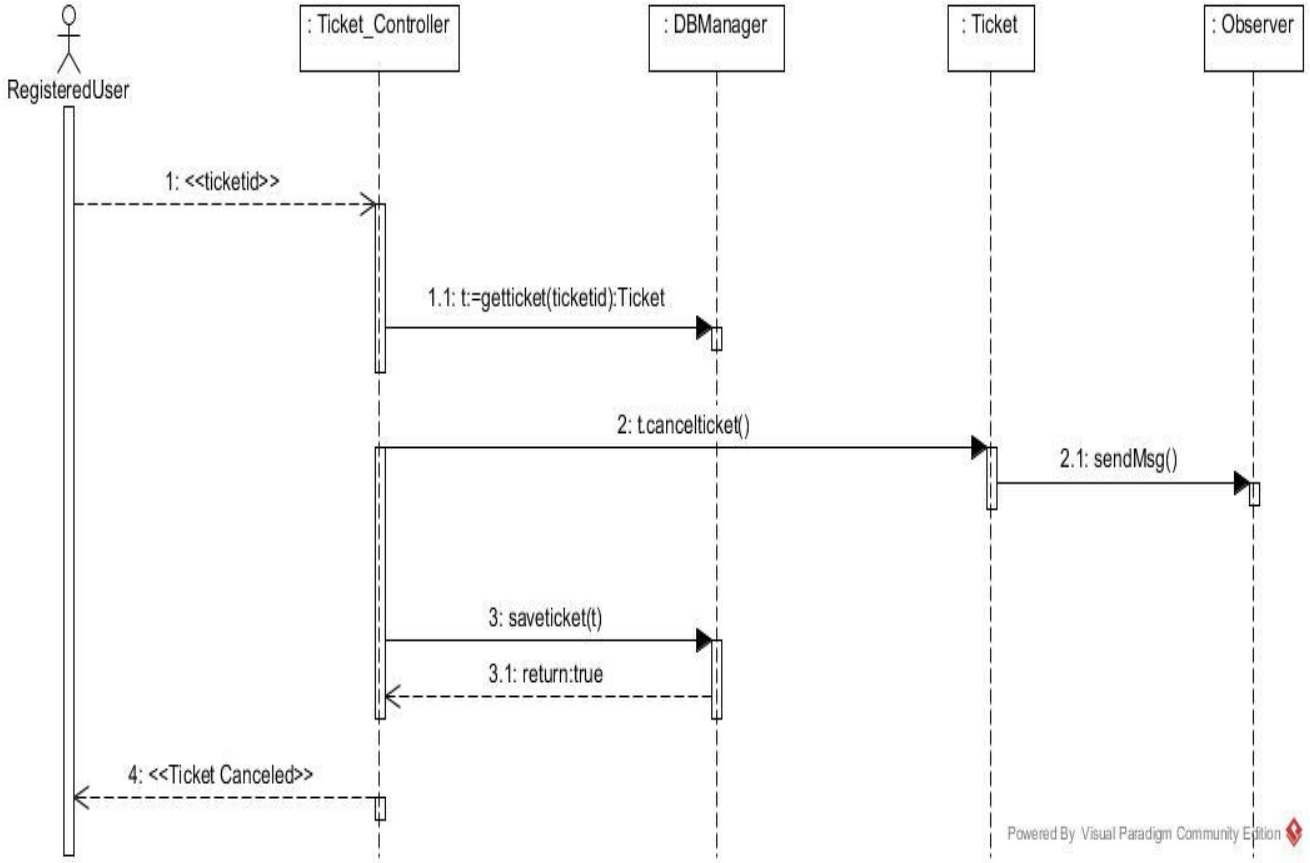
9.6 SEARCH EVENTS



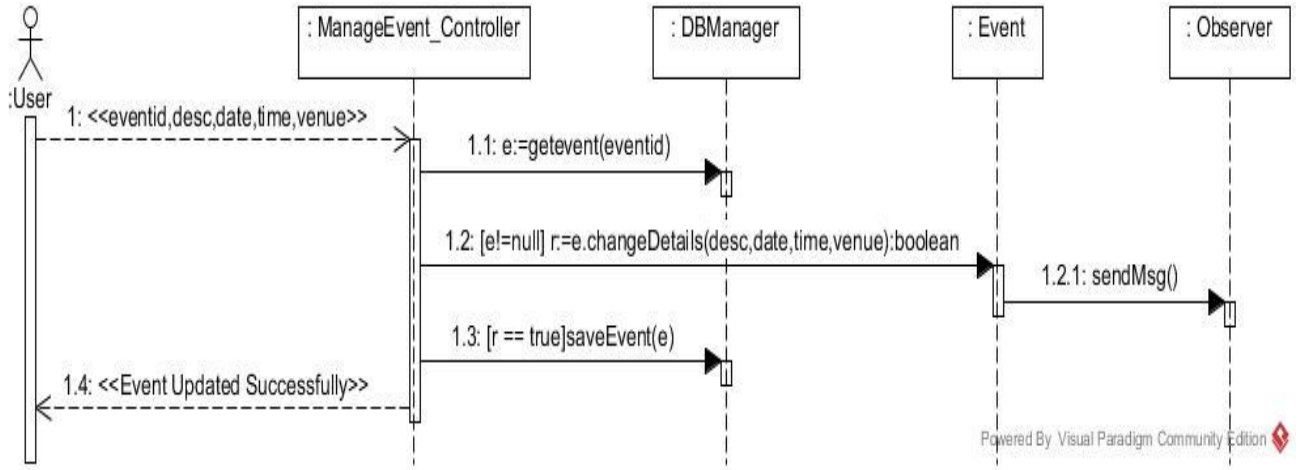
9.7 TICKET RESERVATION



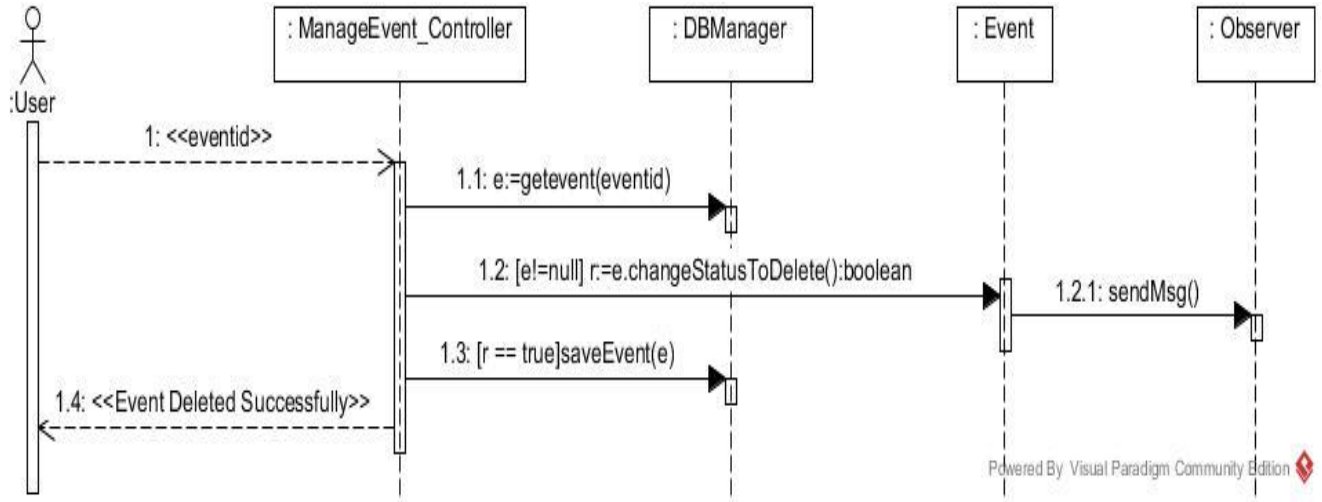
9.8 CANCEL RESERVATION



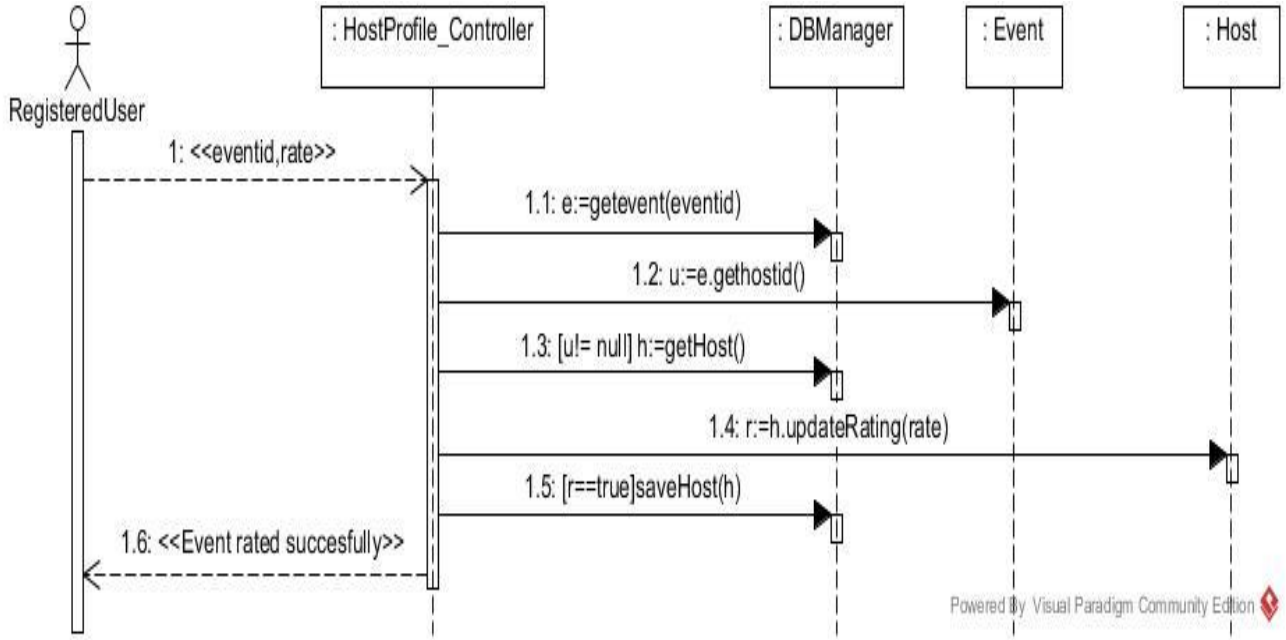
9.9 UPDATE EVENT



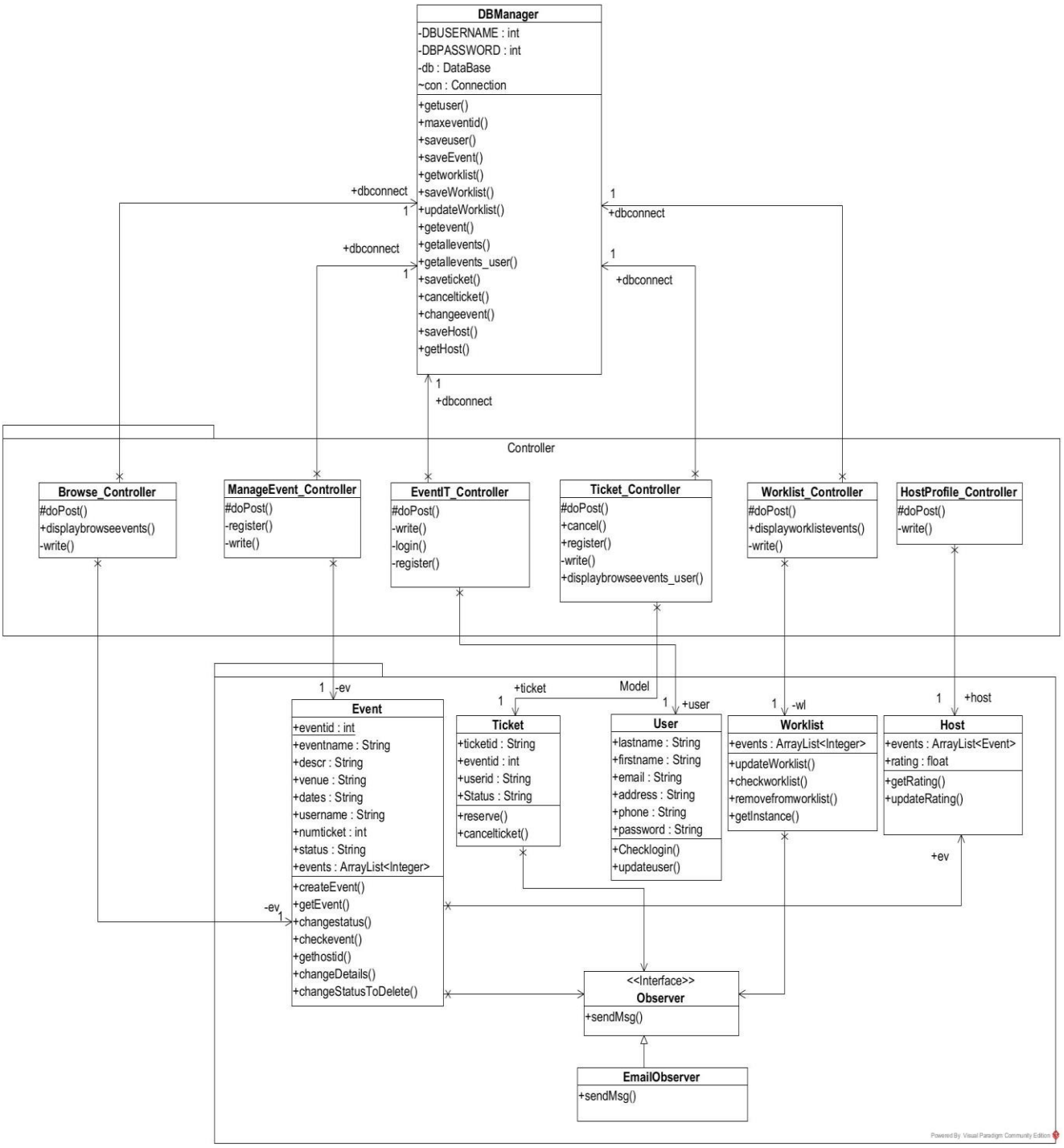
9.10 DELETE EVENT



9.11 RATE EVENT



10.CLASS DIAGRAM



11.SUPPLEMENTARY SPECIFICATION

- This program must run on all the platforms.
- This program must run on Internet Explorer, Google Chrome, Safari and Firefox.
- Developer will give a manual that explains how to use this program.

12.TESTING

Testing is done on the software to check if the quality is maintained and implementation is according to requirements stated earlier.

- Meets the requirements which guided design and development
- Works for correct inputs and handles incorrect inputs
- Get system response within time frame.
- Achieves the Stakeholder goal.

11.1 Test Plans

The following activities and test cases are used to check the quality of code and to map the implementation with the requirements of the system.

11.2 Unit Testing

- Unit testing is done for each use case.
- Testing is done by giving basic inputs.

S.No	Test Case	Test Case Description	Expected Output	Observed Output
1	Login by using email id/user id	1.The user enters URL and clicks enter. 2.User provides login details and clicks login	The user is able to login in	The user is able to login successfully and home page is displayed

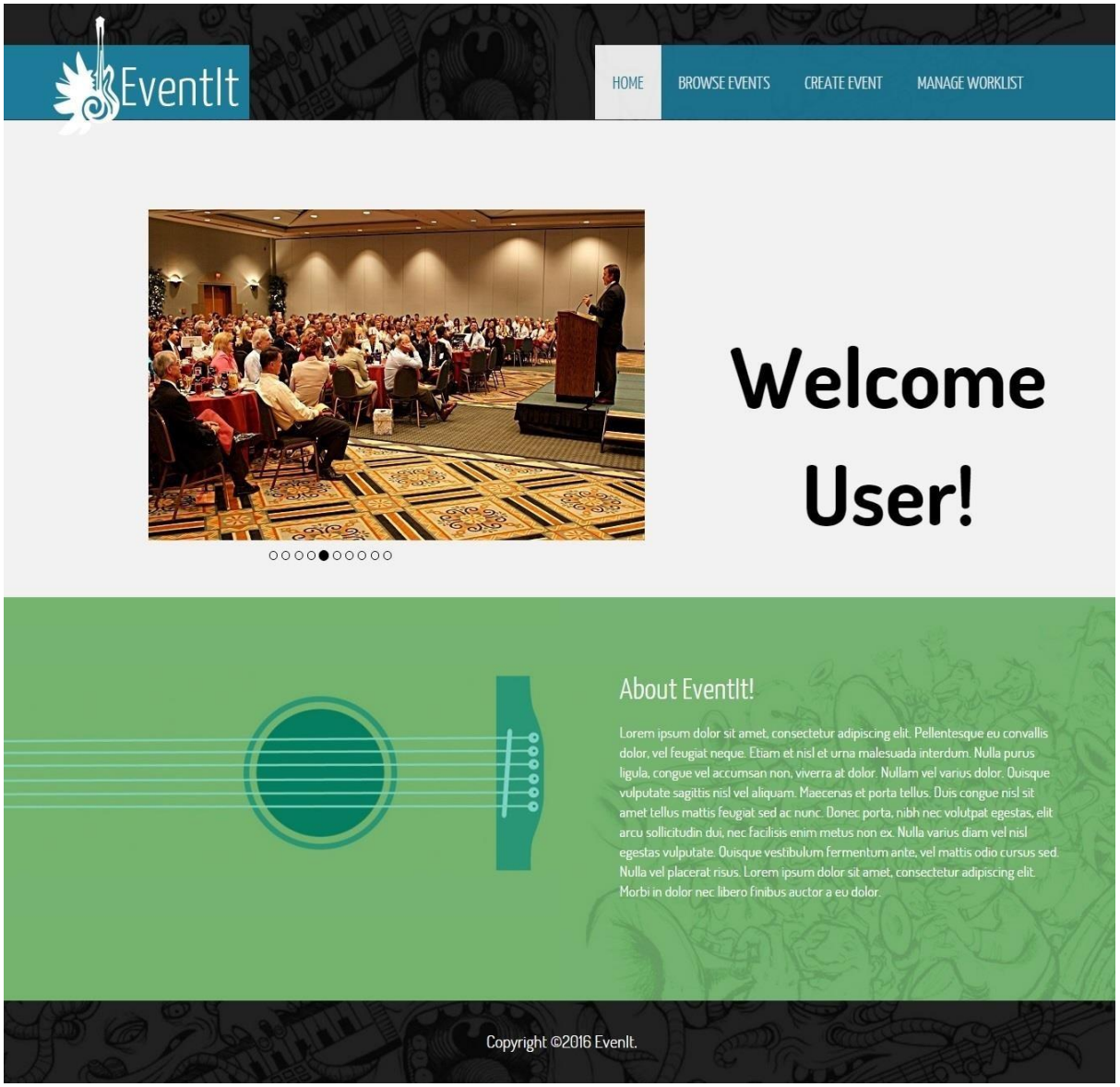
2	Sign up to the system	1.The user enters URL and clicks enter 2.User clicks sign up 3.Open Model page 4.User provides sign up details and clicks submit	The user is able to sign up	The user is able to sign up successfully and login in page is displayed
3	Create Event	1.The user enters URL and clicks enter 2.Open create event page 3.Provide event details 4.Clicks submit	The user is able to create event	The user is able to create event and event is sent for approval
4	Manage Worklist	1.The admin enters URL and clicks enter 2.Open worklist page 3.Clicks event to approval and clicks approve or reject	The admin is able to manage worklist	The user is able to approve event successfully and event is published.
5	Browse Events	1.The user enters URL and clicks enter 2.Click Browse events 3.Displays published events	The user is able to view events	The user is able to view event successfully
6	Ticket Reservation	1.The user enters URL and clicks enter 2.Clicks Ticket reservation 3.Displays reservation form	The user is able to reserve tickets	The user is able to reserve tickets successfully

7	Cancel Reservation	1.The user enters URL and clicks enter 2.Clicks reservation history 3.Clicks on ticket 4.Clicks cancel reservation	The user is able to cancel reserved tickets	The user is able to cancel reserved tickets successfully
8	Update event	1.The user enters URL and clicks enter 2.Goes to my profile and clicks on an event 3.edits the details and clicks on Update	The user is able to update the event details	The user is able to update the event details successfully
9	Delete Event	1.The user enters URL and clicks enter 2. Goes to my profile and clicks on an event 3.Clicks on delete event	The user is able to delete an event	The user is able to delete an event successfully
10	Rate Event	1.The user enters URL and clicks enter 2. Goes to Past tab of the reservation history page. 3. Rates an event.	The event is rated and the rating is associated with the host	The event is rated and the rating is associated with the host successfully.

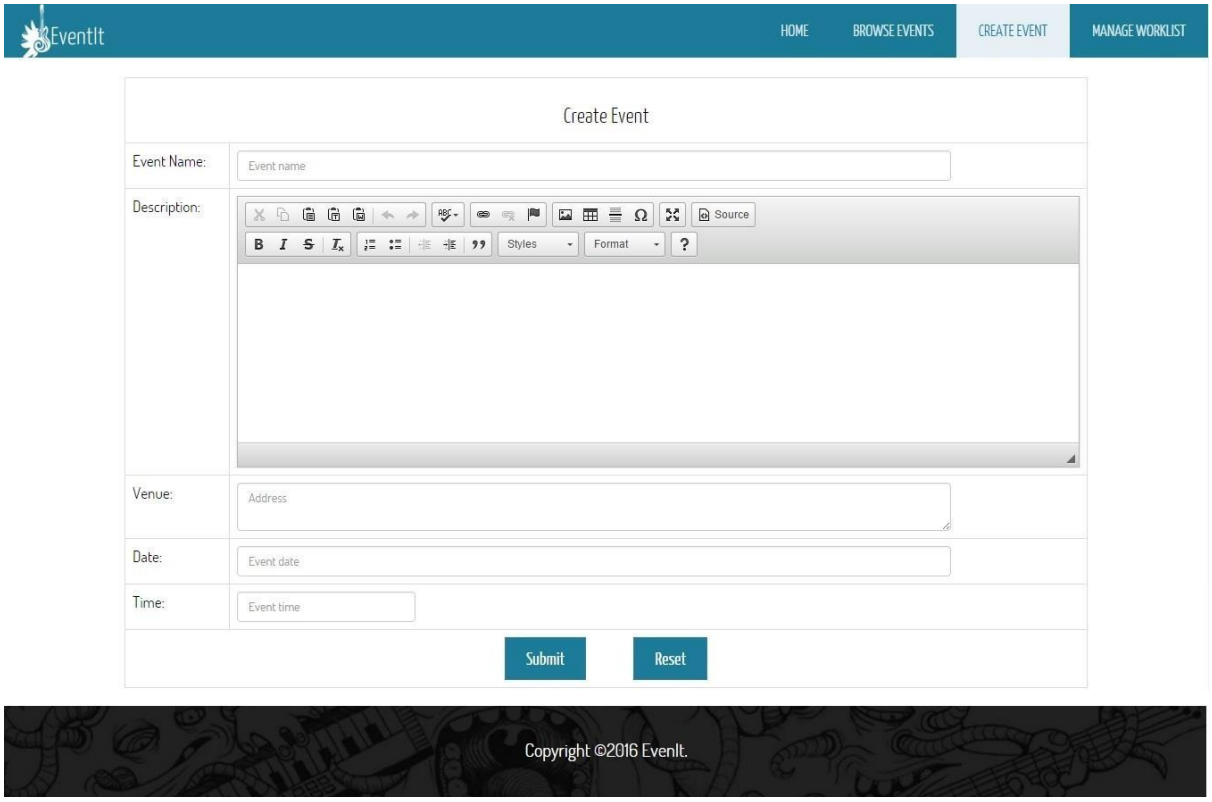
13.UI SCREENSHOTS

Login Page

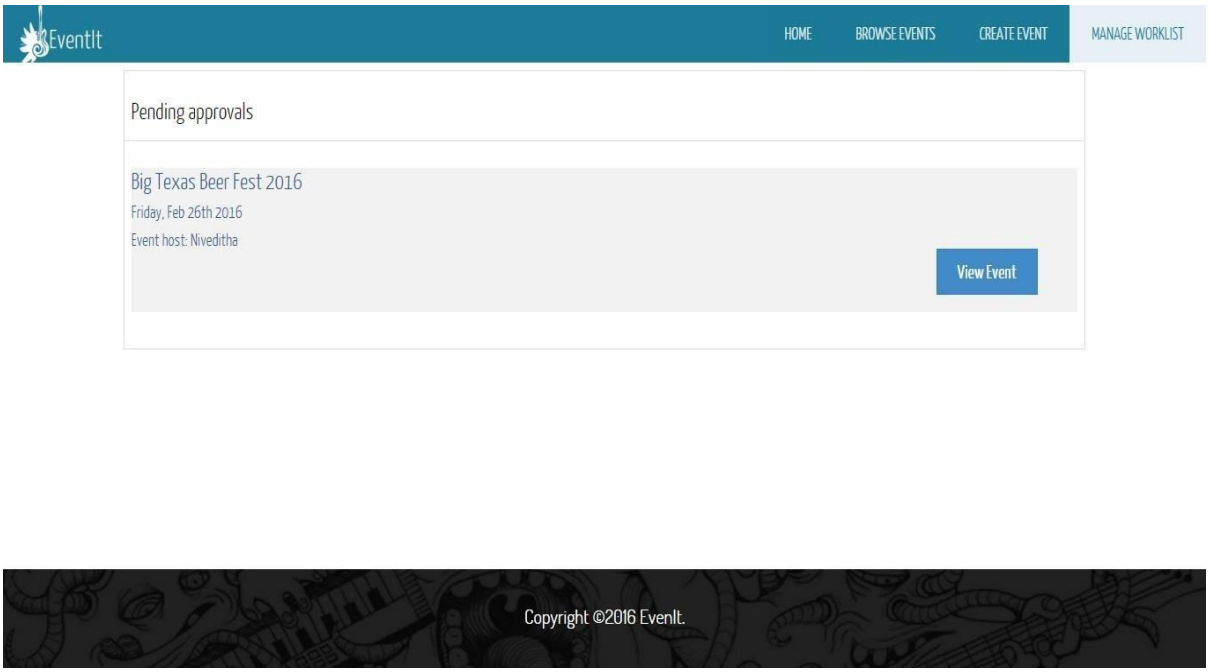
Home Page


EventIt
HOME
BROWSE EVENTS
CREATE EVENT
MANAGE WORKLIST
Welcome User!
About EventIt!
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque eu convallis dolor, vel feugiat neque. Etiam et nisl et urna malesuada interdum. Nulla purus ligula, congue vel accumsan non, viverra at dolor. Nullam vel varius dolor. Quisque vulputate sagittis nisl vel aliquam. Maecenas et porta tellus. Duis congue nisl sit amet tellus mattis feugiat sed ac nunc. Donec porta, nibh nec volutpat egestas, elit arcu sollicitudin dui, nec facilisis enim metus non ex. Nulla varius diam vel nisl egestas vulputate. Quisque vestibulum fermentum ante, vel mattis odio cursus sed. Nulla vel placerat risus. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi in dolor nec libero finibus auctor a eu dolor.
Copyright ©2016 EventIt.

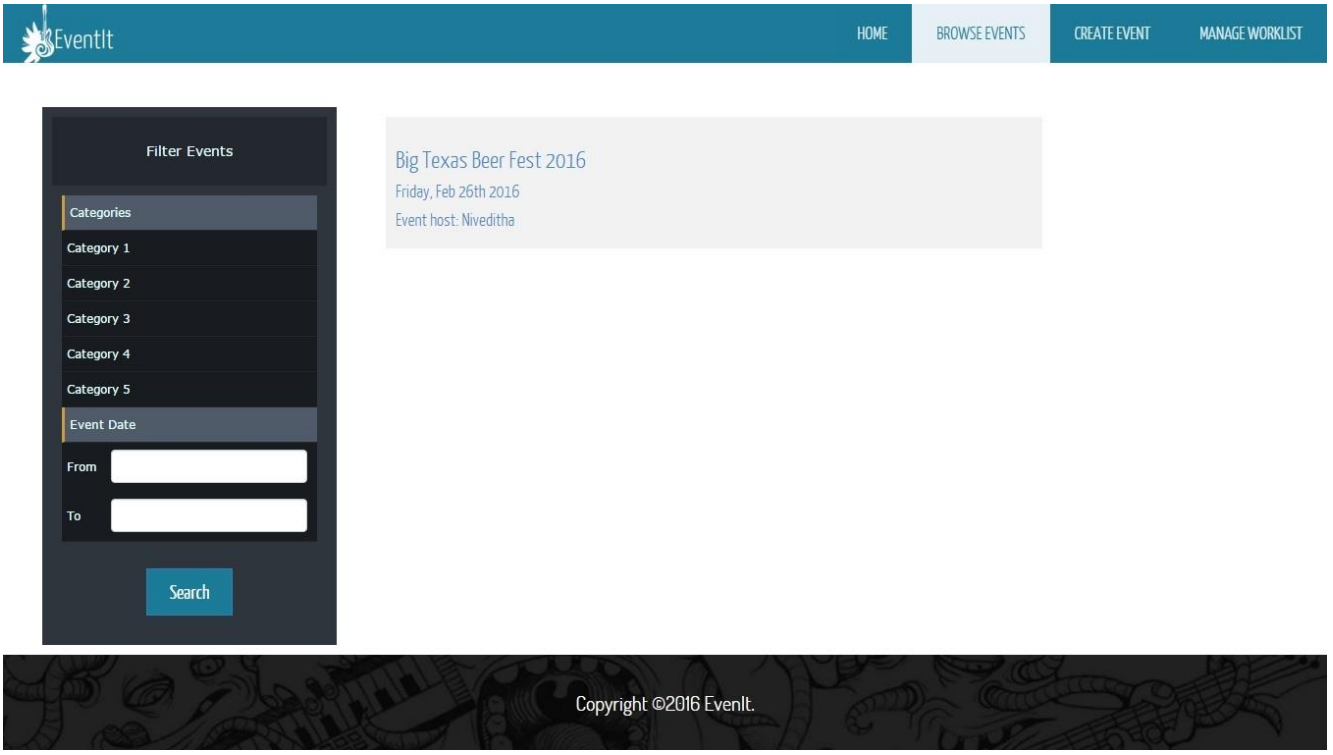
Create Event



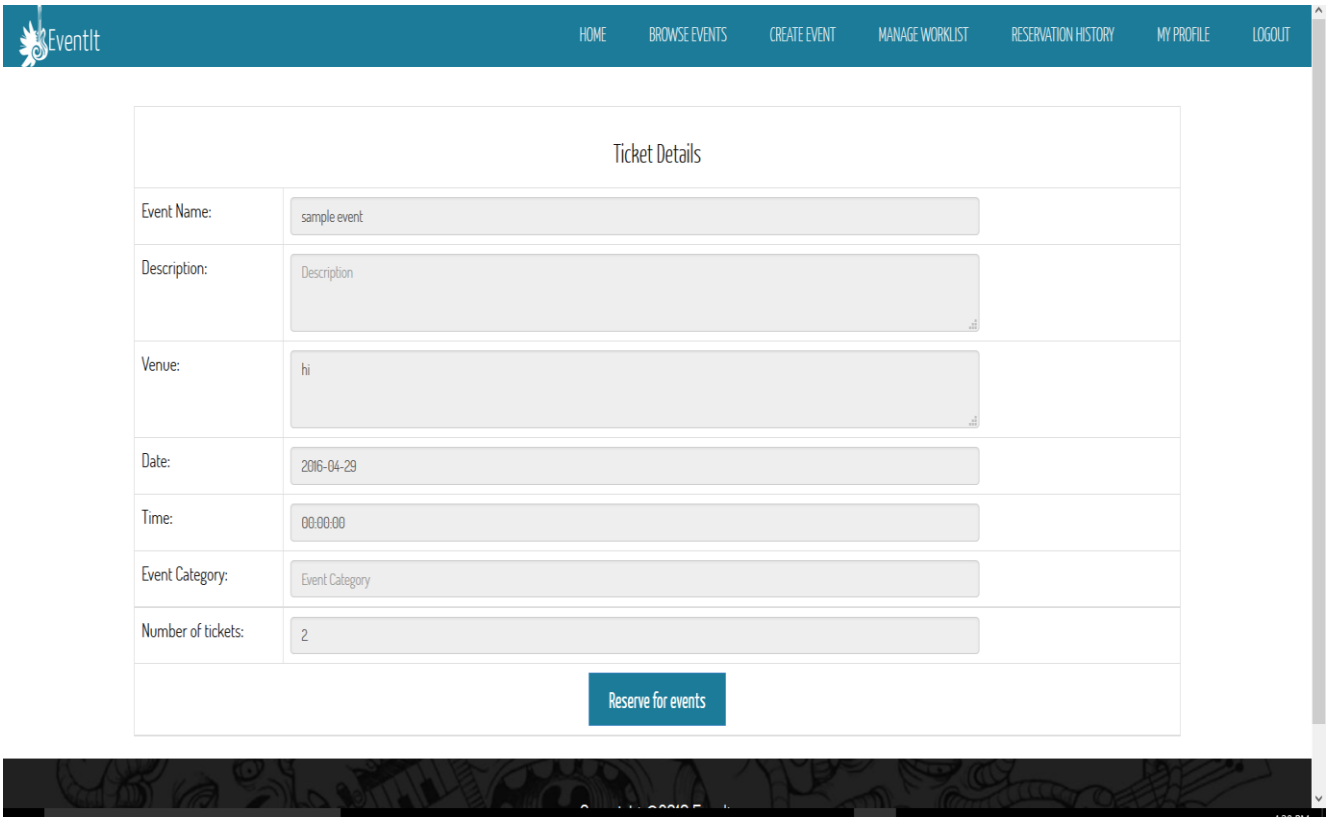
Manage Worklist



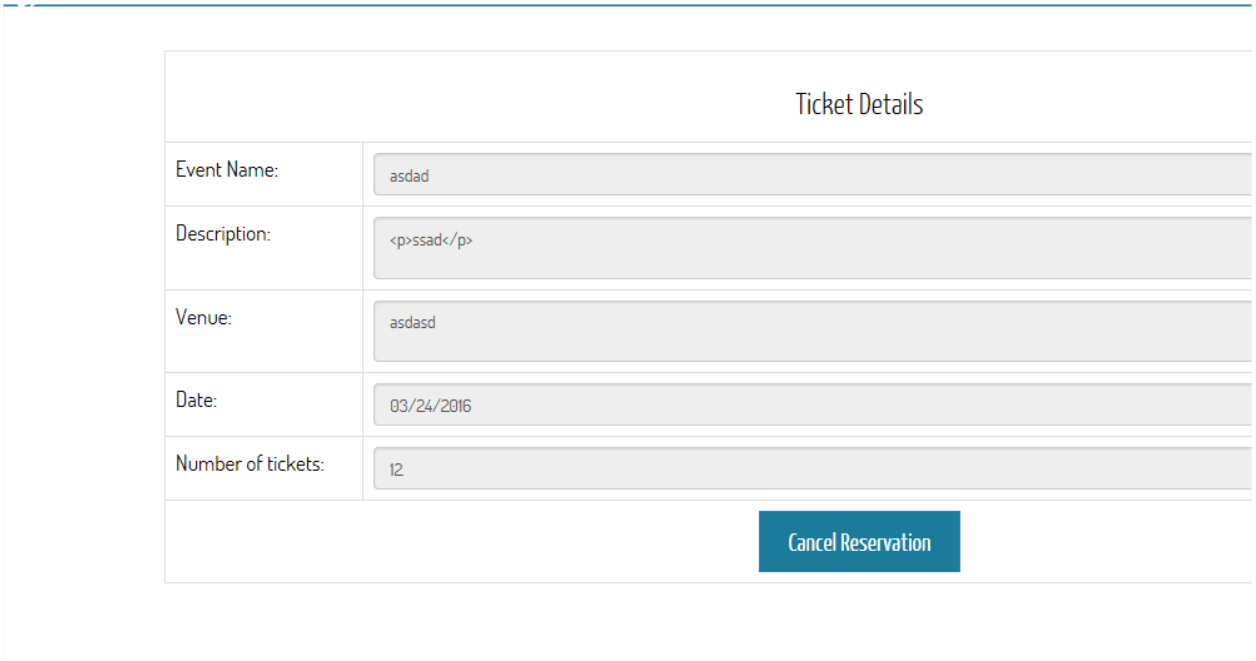
Browse Events



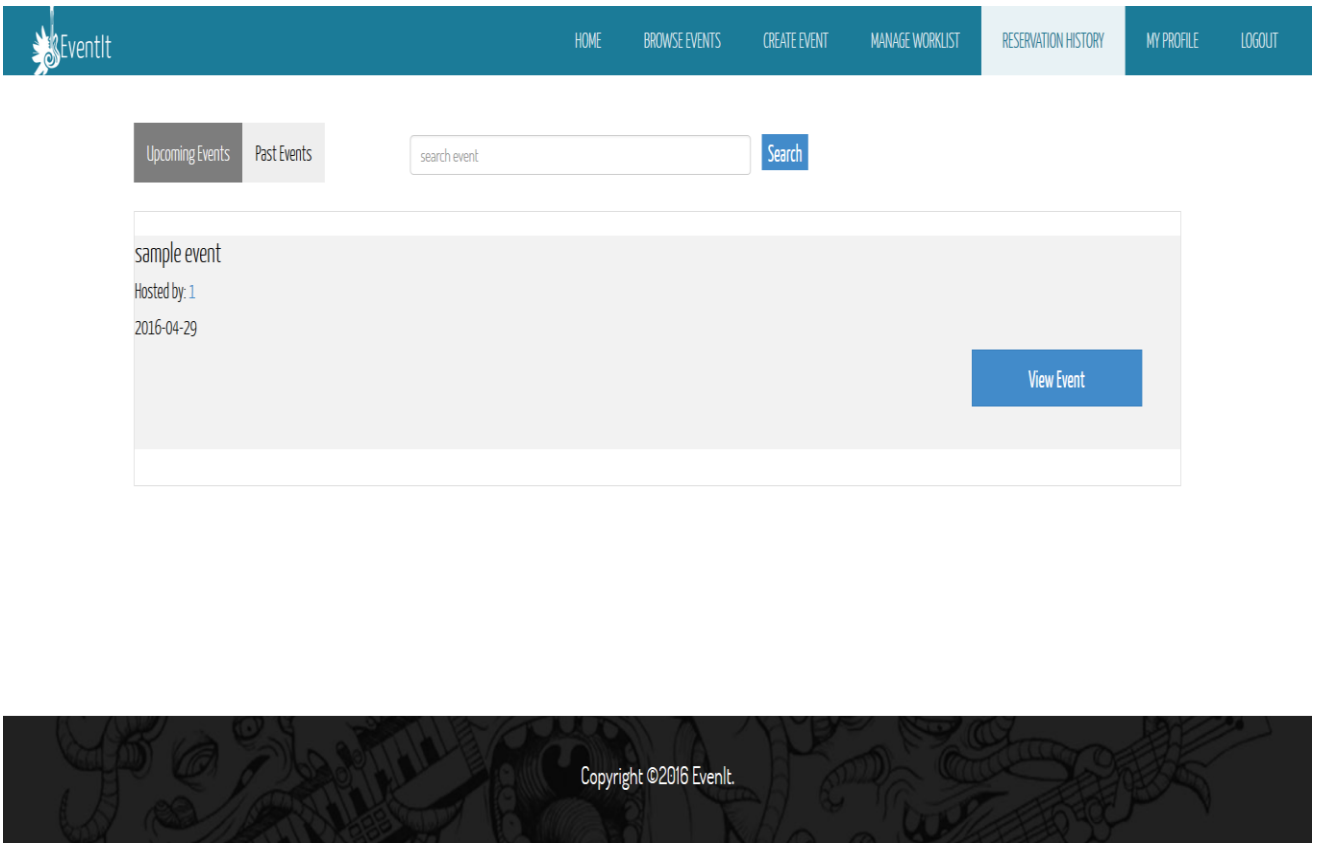
Ticket Reservation



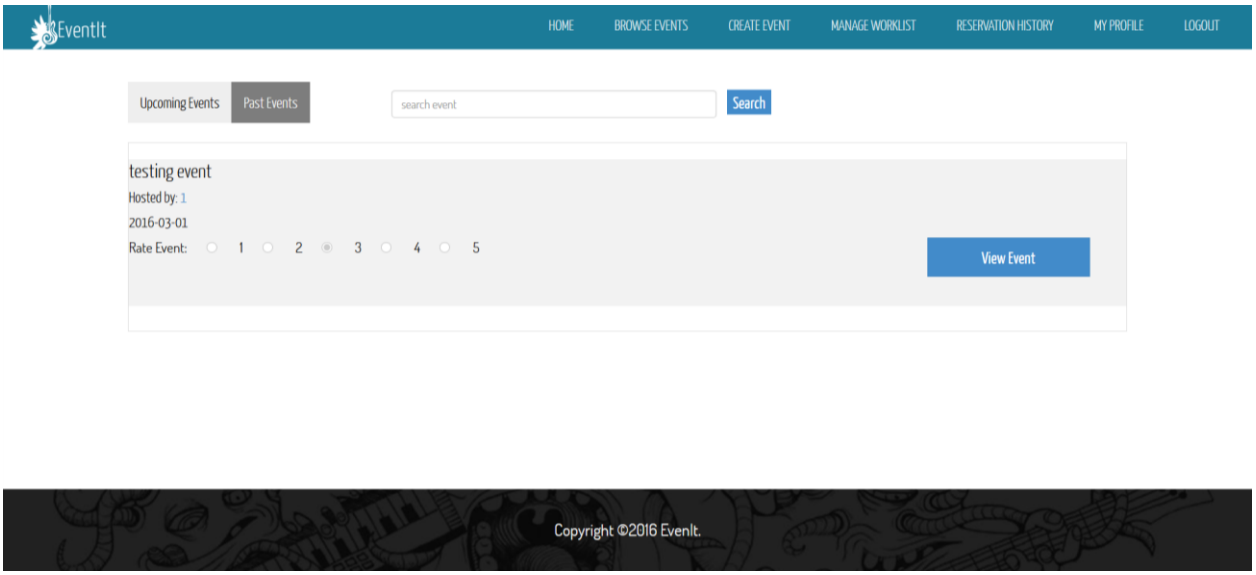
Cancel Ticket Reservation



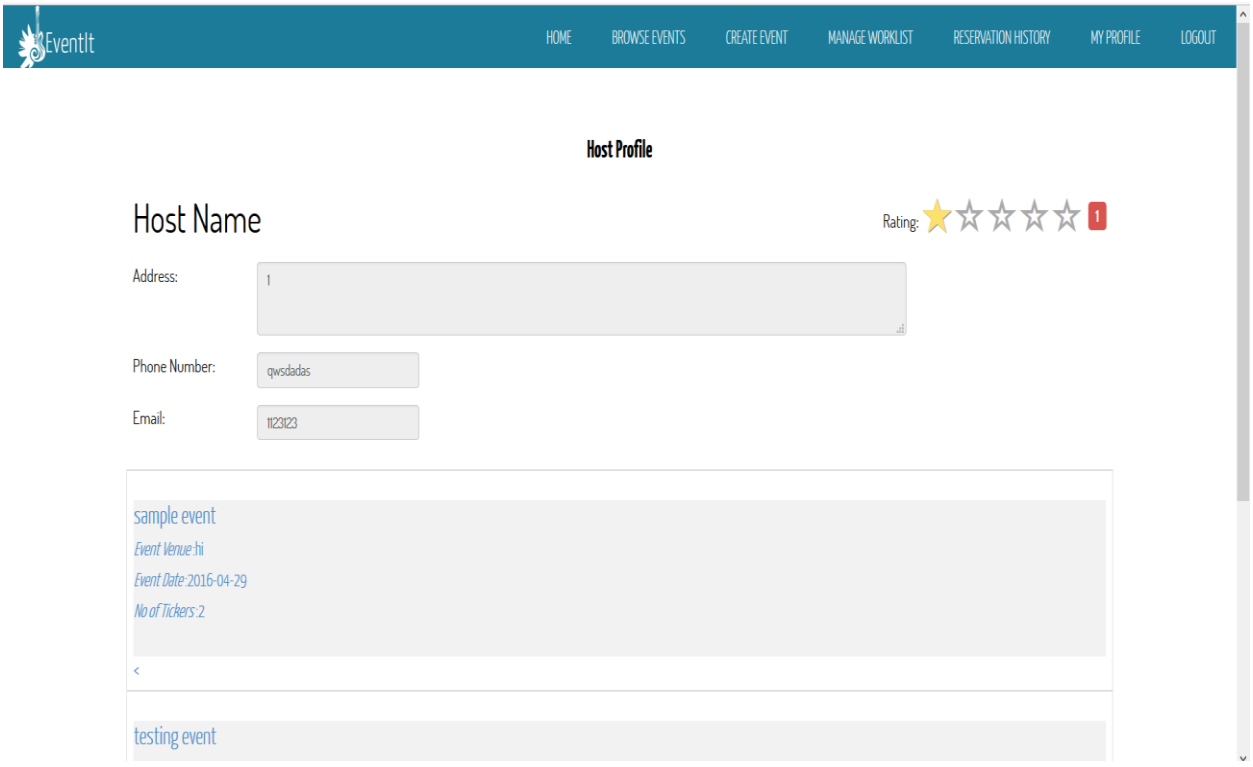
Reservation History



Rate Event



Host Profile



14. GLOSSARY

Glossary			
Revision History			
Version	**Date**	**Description**	**Author**
Iteration 1 document	02/22/2016	First draft to be refined in later iterations	Group 9
Iteration 2 document	03/20/2016	First draft to be refined in later iterations	Group 9
Iteration 3 document	04/18/2016	Final draft of the project document	Group 9

Definitions	
Terms	**Definition and Information**
Admin	Admin is responsible for the credibility of the events being published on the Website. Whenever a registered user creates an event, an event request is sent to the admin for approval, to check the authenticity of the event and its host, before being published on the site. This prevents the creation of fraudulent events on the site.
Registered User	Registered user is the user who has signed up for the website. These users have the privilege to create events and promote them after the event is approved by the admin. They can also edit the event details or delete the event. Not only that, but they can also reserve tickets for the events they wish to attend, which are hosted by other users. They will also have permission to rate the events they have attended.
General User	General users are the other users who visit the website. They can browse events based on different event types and look into the details but they will not be able to reserve tickets unless they sign up
Worklist	The list of events to be approved by the admin to be published on the web site.

15.DEVELOPMENT CASE

Discipline	Artifact	Iteration 1	Iteration 2	Iteration 3
Business Modeling Requirements	Domain Model Use- Case Model	s s	r r	 r
	Vision Supplementary Specification	s s	r r	
Design	Glossary Design Model	s	r s	r r
Implementation	N/A	s	r	r
Testing	N/A	s	r	r
Deployment	N/A		s	r