CS-251: Software Engineering (I)

Course Project: Event Organizer

DR. Mohamed El-Ramly

T.A. Sara Tarek

Requirements Document for Event Organizer

Team Leader: Mostafa Aly El-Sayad

mostafa.elsayad@hotmail.com

1/11/2014

Contents

CS251: Phase 1 – Event Organizer
Software Requirements Specifications

Document Purpose and Audience:

This document describes the software requirements for the event organizer application. This made specially to help the designer, developer, maintainer, and tester of the event organizer.

Introduction:

Software Requirements Specifications

> ## **Software Purpose:**

This software will be made to ease event organizing process for all people all over the world. Giving an opportunity for users to create and manage their special events with very simple steps.

> ## **Software Scope:**

This software focuses on letting people create, manage, and join different types of events. And also doesn't concern about the physical organization process of the event. Simply the software doesn't guarantee that all the created events will exist in the real world this is the responsibility of the event owner.

> ## **Definitions, acronyms, and abbreviations:**

Software Requirements Specifications

Term	Definition
Account:	An account represents the physical user which let him deal directly with the software. User can have only one account.
Event:	It is a social gathering situation that people meet together for a specific purpose.
Event Creator:	A person who creates an event
Event Attendee:	A person who attends an event created by an event creator
Ticket:	A permission that allows a user to attend paid events
Preferences:	A set of options that enables users to manage and edit their accounts
Notifications:	A trigger that tells software users about the latest updates about different events.
Profile:	contains the personal information about the software user

CS251: Phase 1 – Event Organizer
Software Requirements Specifications

Requirements:

> ## **Functional Requirements:**

Functional Requirement #1:

Description: Sign up: The user is requested to enter his first name, last name, username , password, email, and country
Input: User's details sent to the database system
Processing: Generating a unique ID for this user and storing his/her information.
Output: Add user to the system or reject.

Functional Requirement #2:

Description: Login: The user is requested to enter his username and password.
Input: User's details sent to the database system
Processing: Verify user's input.
Output: Login successful or login failure.

Functional Requirement #3:

Description: Create event: the user is requested to enter all the details of the event.
Input: Name, place, category, time, description, type of event (public or private) of the event.
Processing: Send the event details to the database system – checking on these details.
Output: Event created or rejected

Functional Requirement #4:

Description: Join event:the user selects an event from the recommended events or select an event from event searcher.
Input: Event name and user ID
Processing: Let user attend the event of applicable
Output: User joined the event or couldn't join

Functional Requirement #5:

Description: Event Searcher: the user is promoted to enter an event name
Input: Event name
Processing: Database searches for the requested event
Output: Event information or event couldn't be found

Functional Requirement #6:

Description:

Delete Event: the user is allowed to remove his/her own events from the system

Input:
Event name and user ID

Processing:
Database deletes the event if applicable

Output:
Event deleted • Notify joined users Or ,Event could not be deleted

Functional Requirement #7:

Description:
Buy ticket :a joined users for an event have to buy a ticket to attend this event

Input:
Event name and user ID

Processing:
Database checks on joining condition –process to payment method.

Output:
An event ticket has been issued or couldn't buy a ticket

Functional Requirement #8:

Description:
Log out: the user is promoted to close his/her account

Processing:
Close user account

Output:
Log out successful

Functional Requirement #9:

Software Requirements Specifications

Description:
View profile: the user can view his/her information
Input:
User ID
Processing:
Database looks up for the user's information
Output:
Display user's information

Functional Requirement #10:

Description:
Follow user: the user selects another registered user to follow him/her
Input:
User ID and the desired user's name
Processing:
Database looks for the user , link both of users accounts together if applicable
Output:
User is followed or couldn't be followed

Functional Requirement #11:

Description:
Discover events by category: the user is promoted to view all public events in a specific category
Input:
Event Category
Processing:
Database looks for the requested category and retrieves all events related to this category
Output:
Events of this category

Functional Requirement #12:

Software Requirements Specifications

Description:
View recommended events: provides a list of recommended events by the system based on the user preferences.
Input:
User ID
Processing:
Database will produce a list of events based on user's preferences
Output:
List of recommended events

Functional Requirement #13:

Description:
View recommended events: provides a list of recommended events by the system based on the user preferences.
Input:
User ID
Processing:
Database will produce a list of events based on user's preferences
Output:
List of recommended events

Functional Requirement #14:

Description:
View recently added events: provides a list of recently created events by the user's friends
Input:
User ID
Processing:
The database system will produce a list of recently created events by the user's friends
Output:
List of recently added event or empty list

> ## Non-Functional Requirements:

Software Requirements Specifications

❖ <u>Usability:</u>

This software should be easy to use and doesn't require any technical/programming skills.

❖ <u>Reliability:</u>

This application should be reliable in its implementation and doesn't contain any errors.

❖ <u>Security:</u>

This software should be highly secured to protect the users' data and personal information.

❖ <u>Maintainability:</u>

This software should have regular maintenance

❖ <u>Legal:</u>

This application doesn't break any rules or laws and usage of it doesn't acquire any types of payments.

❖ <u>Interface:</u>

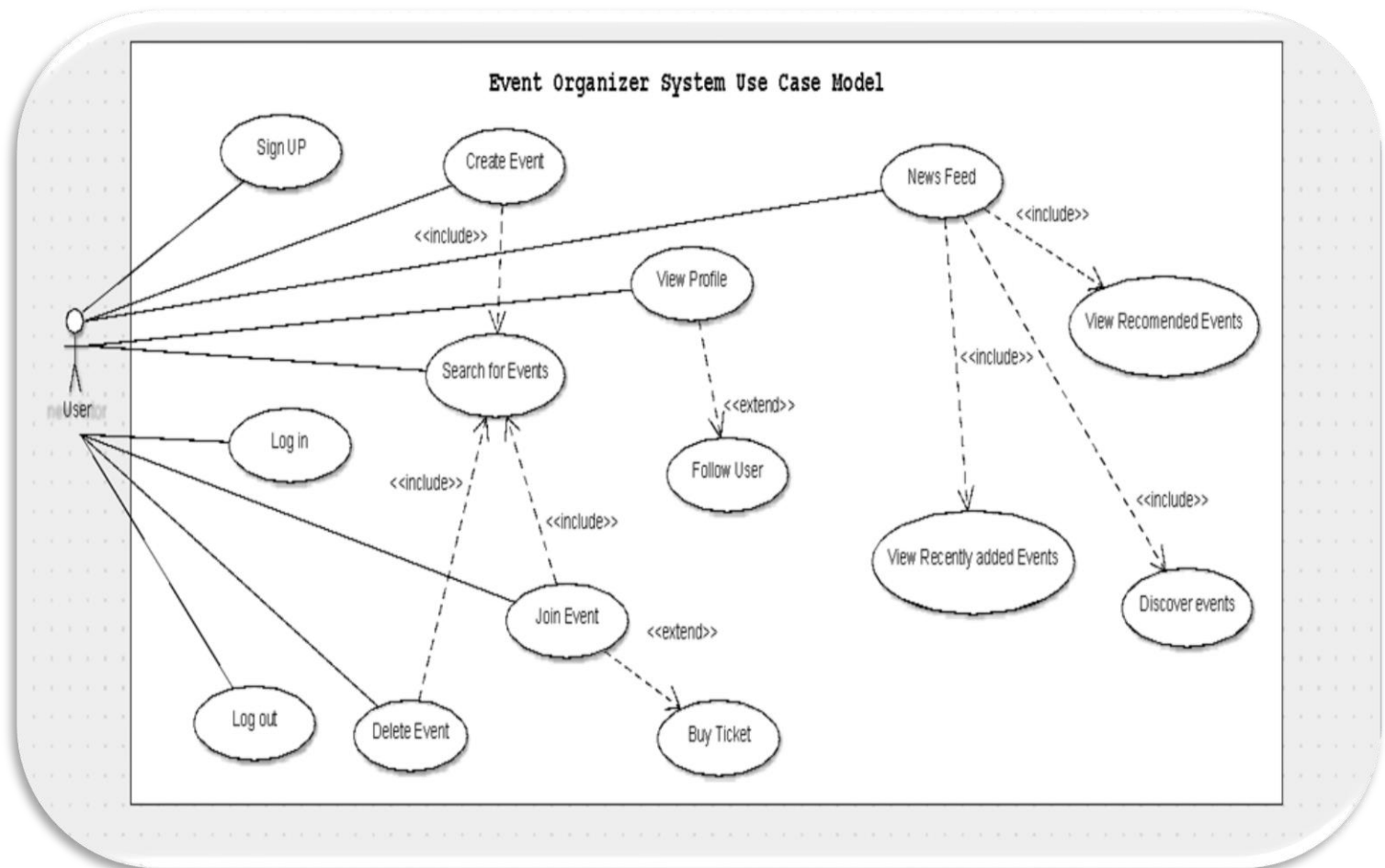This software provides a friendly user interface that allows him to interact easily with the software features.

System Models:

➢ Use Case Model:

Event Organizer System Use Case Model

➢ Use Case Tables:

Software Requirements Specifications

Use Case Table #1:

Use Case ID:	Functional_requirement_1	
Use Case Name:	User Signing up a new account	
Actors:	System user	
Pre-conditions:	Load Event Organizer application	
Post-conditions:	User gains a new account	
Flow of events:	**User Action**	**System Action**
	1- User Enters an e-mail and password	
		2- System validate the entered e-mail
	3- User promoted to enter his/her profile information	
		4- System creates a new account for the user
Exceptions:	Invalid e-mail or already existing e-mail	
Includes:	N/A	
Notes and Issues:	N/A	

Use Case Table #2:

Software Requirements Specifications

Use Case ID:	Functional_requirement_2	
Use Case Name:	Log in	
Actors:	System user	
Pre-conditions:	Load Event Organizer application	
Post-conditions:	User gains access to his/her account	
Flow of events:	**User Action**	**System Action**
	1- User Enters an e-mail and password	
		2- System authenticates user's input
	3- User will be directed to his/her profile stream	
Exceptions:	Incorrect e-mail or password	
Includes:	N/A	
Notes and Issues:	N/A	

Use Case Table #3:

Software Requirements Specifications

Use Case ID:	Functional_requirement_3	
Use Case Name:	User creates a new event	
Actors:	System user	
Pre-conditions:	Load Event Organizer application – Access your account	
Post-conditions:	New event is created for the user	
Flow of events:	**User Action**	**System Action**
	1- User Enters event's details (name , place , time , category , type)	
		2- System validate the entered e-mail
	3- User will be requested to confirm the event creation process	
		4- System creates a new event and associate the event to its respective user
Exceptions:	Already Existing Event	
Includes:	Searching for the existence of the entered event on the system	
Notes and Issues:	N/A	

Use Case Table #4:

Software Requirements Specifications

Use Case ID:	Functional_requirement_4	
Use Case Name:	User joins an event	
Actors:	System user	
Pre-conditions:	Load Event Organizer application – Access your account – Select an event	
Post-conditions:	User will be in the attendants list of the event	
Flow of events:	**User Action**	**System Action**
	1- User selects an event by entering its name	
		2- System validate the entered event
		3- System retrieves event's information
	4- Users preview the event information and confirming the process	
		5- System links user to the selected event
Exceptions:	Invalid Event or an expired event	
Includes:	Searching for the existence of the entered event on the system	
Notes and Issues:	Buying tickets for paid events	

Use Case Table #5:

Software Requirements Specifications

Use Case ID:	Functional_requirement_5	
Use Case Name:	User searches for an event	
Actors:	System user	
Pre-conditions:	Load Event Organizer application – Access your account	
Post-conditions:	N/A	
Flow of events:	**User Action**	**System Action**
	1- User Enters event's name	
		2- System searches for the event
		3- System retrieves the event information
Exceptions:	Not found event	
Includes:	N/A	
Notes and Issues:	N/A	

Use Case Table #6:

Software Requirements Specifications

Use Case ID:	Functional_requirement_6	
Use Case Name:	User deletes his/her own events	
Actors:	System user	
Pre-conditions:	Load Event Organizer application – Access your account – the possession of the selected event	
Post-conditions:	Event will be deleted	
Flow of events:	**User Action**	**System Action**
	1- User Enters event's name	
		2- System searches for the event
	3- User waits for the search result	
		4- System retrieves the event information
	5- User confirms the deletion process	
		6-System deletes the event
Exceptions:	Not found event – the user is not the owner of the event	
Includes:	Searching for the existence of the entered event on the system	
Notes and Issues:	N/A	

Use Case Table #7:

Use Case ID:	Functional_requirement_7	
Use Case Name:	Buy ticket	
Actors:	System user	
Pre-conditions:	Load Event Organizer application – Access your account – joined in a paid event	
Post-conditions:	User will possess a ticket for the event	
Flow of events:	**User Action**	**System Action**
	1- User Enters event's name	
		2- System searches for the event
	3- User waits for the search result	
		4- System retrieves the event information
	5- User confirms the purchase process	
		6-System issues a ticket for the user.
Exceptions:	Not found event – no sufficient balance	
Includes:	Searching for the existence of the entered event on the system	
Notes and Issues:	N/A	

Use Case Table #8:

Software Requirements Specifications

Use Case ID:	Functional_requirement_10	
Use Case Name:	Following a user	
Actors:	System user	
Pre-conditions:	Load Event Organizer application – Access your account – search for a user	
Post-conditions:	User will follow his/her friend	
Flow of events:	**User Action**	**System Action**
	1- User Enters another user's name	
		2- System searches for user
	3- User waits for the search result	
		4- System retrieves the user's information
	5- User confirms the adding process	
		6-System issues a ticket for the user.
Exceptions:	Not found user – maximum number of friends is reached	
Includes:	Searching for the existence of the entered user on the system	
Notes and Issues:	N/A	

Use Case Table #9:

Software Requirements Specifications

Use Case ID:	Functional_requirement_12	
Use Case Name:	Viewing recommended events	
Actors:	System user	
Pre-conditions:	Load Event Organizer application – Access your account – user must have set his/her preferences	
Post-conditions:	N/A	
Flow of events:	**User Action**	**System Action**
		2- System searches for events based on user's preferences
	3- User waits for the search result	
		4- System produce a list of these events
Exceptions:	Empty list if there is no user preferences	
Includes:	N/A	
Notes and Issues:	N/A	

Ownership Report:

Software Requirements Specifications

Items Table:

Item	Owner
Document purpose and audience	All team members
Introduction	Mostafa El-Sayad& Eslam Ibrahim
Functional Requirements (1-7)	Mostafa El-Sayad& Eslam Ibrahim & Ibrahim Abd El-Rheem
Functional Requirements (8-14)	Ahmed Saber & Ibrahim Abd El-Rheem
Non-Functional Requirements	Mostafa El-Sayad
System Models	Eslam Ibrahim

Jobs Table:

Job	Team Member
Brainstorming	All team members
Typing SRS Document	Eslam Ibrahim
Reviewing SRS Document	Mostafa El-Sayad
Creating GitHub Repository	Eslam Ibrahim
Committing in GitHub Repository	All team members