CS-251: Software Engineering (I)

Course Project: Event Organizer

DR. Mohamed El-Ramly

T.A. Sara Tarek

Software Design Specifications Document
For Event Organizer

Team Leader: Mostafa Aly El-Sayad

mostafa.elsayad@hotmail.com

7/12/2014

CS251: Phase 2
Project: Event Organizer
Software Design Specification

Contents

Software Design Specification

Team Members:

ID	Name	Email	Signature	Mobile
20120382	Mostafa Aly El-Sayad	mostafa.elsayad@hotmail.com	Mostafa El-Sayad	01009252995
20110017	Eslam Osama Mohamed Ibrahim	eom_realmadrid@live.com	Eslam Ibrahim	01092741971
20110058	Ahmed Saber	ahmed.saber99985@yahoo.com	Ahmed Saber	01110126543
20110006	Ibrahim Abd El-Reheem	ebrahimfci@yahoo.com	Ibrahim Abd El-Reheem	████████

CS251: Phase 2
Project: Event Organizer
Software Design Specification

Document Purpose and Audience:

This document provides all system models needed for the implementation process of the event organizer application. This made especially to help developer, maintainer, and tester of the event organizer and will not be available for users.
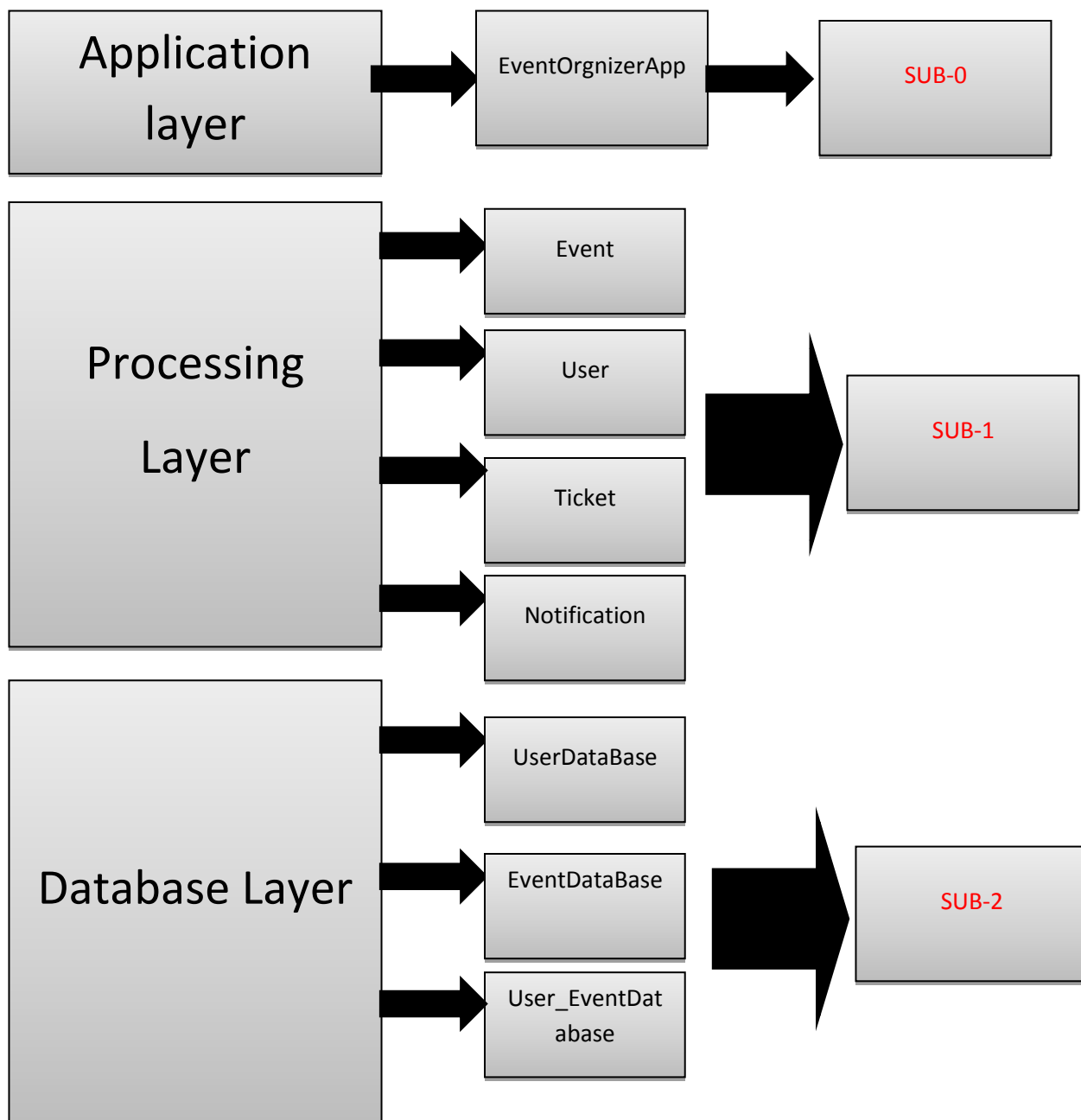
Software Design Specification

System Models:

➢ **System Decomposition:**

| Application layer | → | EventOrgnizerApp | → | SUB-0 |

		Event		
Processing Layer		User	→	SUB-1
		Ticket		
		Notification		

		UserDataBase		
Database Layer		EventDataBase	→	SUB-2
		User_EventDatabase		

Software Design Specification

> ➢ **Class Diagram:**

EventOrganizerApp
frame1 : javaFrameOne
main()

User
userID : int
userName : String
password : String
Email : String
categories : String*
signUP(String,String) : User
logIN(String,String) : User
viewProfile() : User
followUser(userID : int) : void
searchUser(UserName : String) : int
EditProfile(String,String,String,String) : void
EditPreferences(String*) : void

interact   M   1

Access   1

Create Event   1   M

Join Event   1   M

Delete Event   1   1   M

Event
eventName : String
EventID : int
Place : String
Category : String
Description : String
IsFree : boolean
time : String
addEvent(String,String,String,String,boolean,String) : void
joinEvent(String) : void
deleteEvent(String) : void
eventSearch(eventName : String) : Event
discoverEvents(eventCategory : String) : Event*

User_EventDatabase
recentlyAddedEvents(thisUser : User) : Notification
getRecommendedEvent(ThisUser) : Notification
AddJoinRecord(thisUser : User,EventName : String) : Notification

buy   Manages   1   1

1   1   1

M

Ticket
TicketID : int
userName : String
Place : String
setticketID(ID : int) : void
getticketID() : int
setUserName(Name : String) : void
getUserName() : String
buyTicket(thisUser : User) : Notification

EventDataBase
AddEvent(newEvenet : Event) : String
retrieveEvent(EventName : String) : Event
DeleteEventRecord(EventName : String) : String
GetdiscoverEvents(eventCategory : String) : Event*

Has

view   1

Notification
Text : String
ID : int
setText(Text : String) : void
getTexr() : String
setID(ID : int)
getID() : int

M

UserDataBase
AddUser(newUser : User) : String
retrieveUser(UserName : User) : User
checkPassword(Password,UserName) : Boolean
linkUsers(thisUser : User,userID : int) : Boolean
getRecentlyFollowers(thisUser : user) : Notification

Software Design Specification

Class ID	Class Name	Subsystem ID	Description
1	User	Sub-1	Representing the user entity in the system
2	Event	Sub-1	Representing the event entity in the system
3	EventOrganizer	Sub-0	Is the boundary class which will let the system user interact with all features in the application
4	User_EventDatabase	Sub-2	This class will directly contact with the system's database to do all operations related to user and event interaction
5	Ticket	Sub-1	This class will represent the ticket entity in the system
6	Notification	Sub-1	This class will represent the notification entity in the system
7	UserDataBase	Sub-2	This class will communicate directly with the user's table in the database to manage all operations that could be done on the user's Data
8	EventDatabase	Sub-2	This Class will be linked with the Event's table in the database to perform all operations on Event Entiy

Software Design Specification

➢ **Sequence Diagrams:**

        Sequence Diagram #1: "use case table #1"
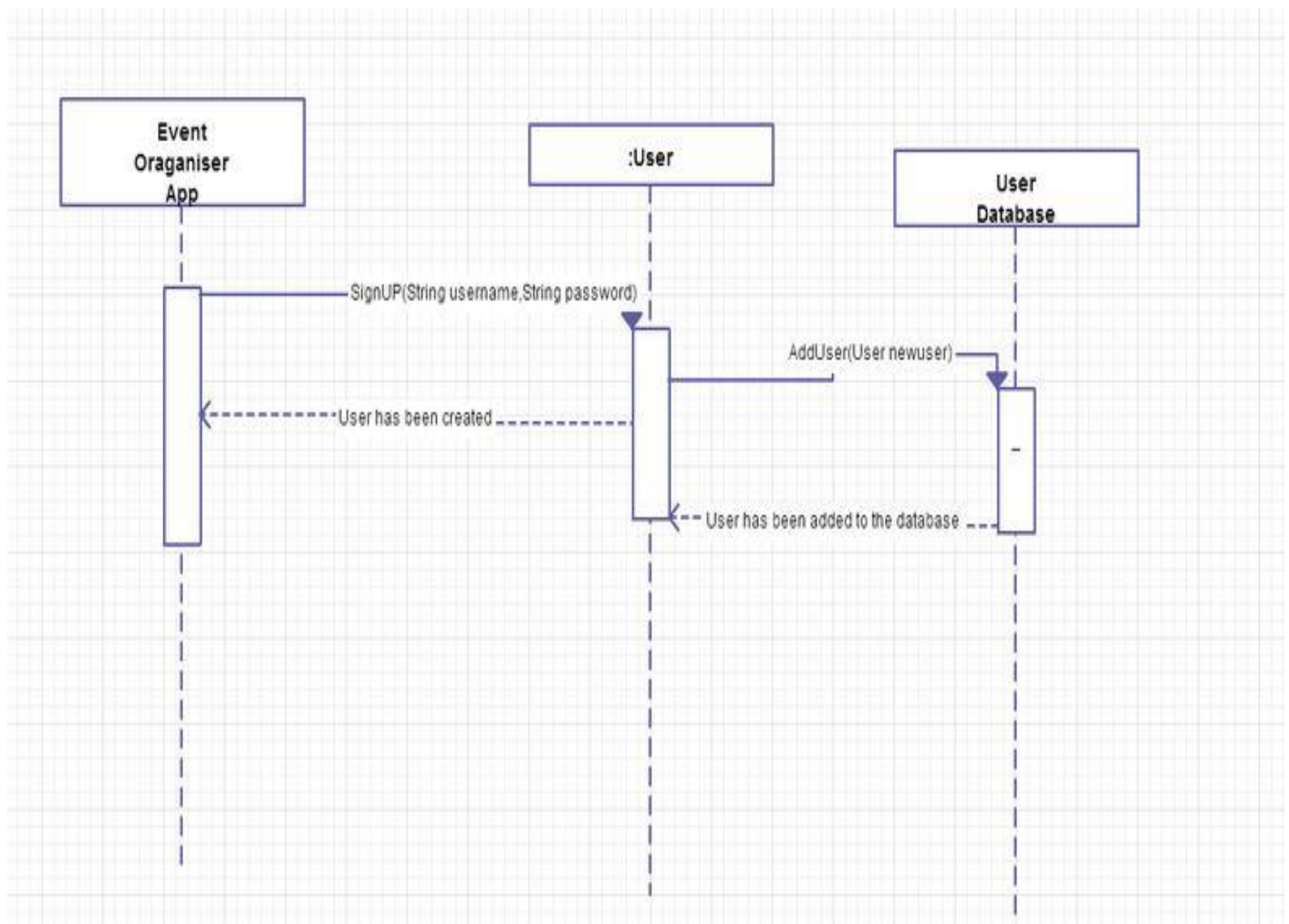
Software Design Specification

Sequence Diagram #2: "use case table #3"
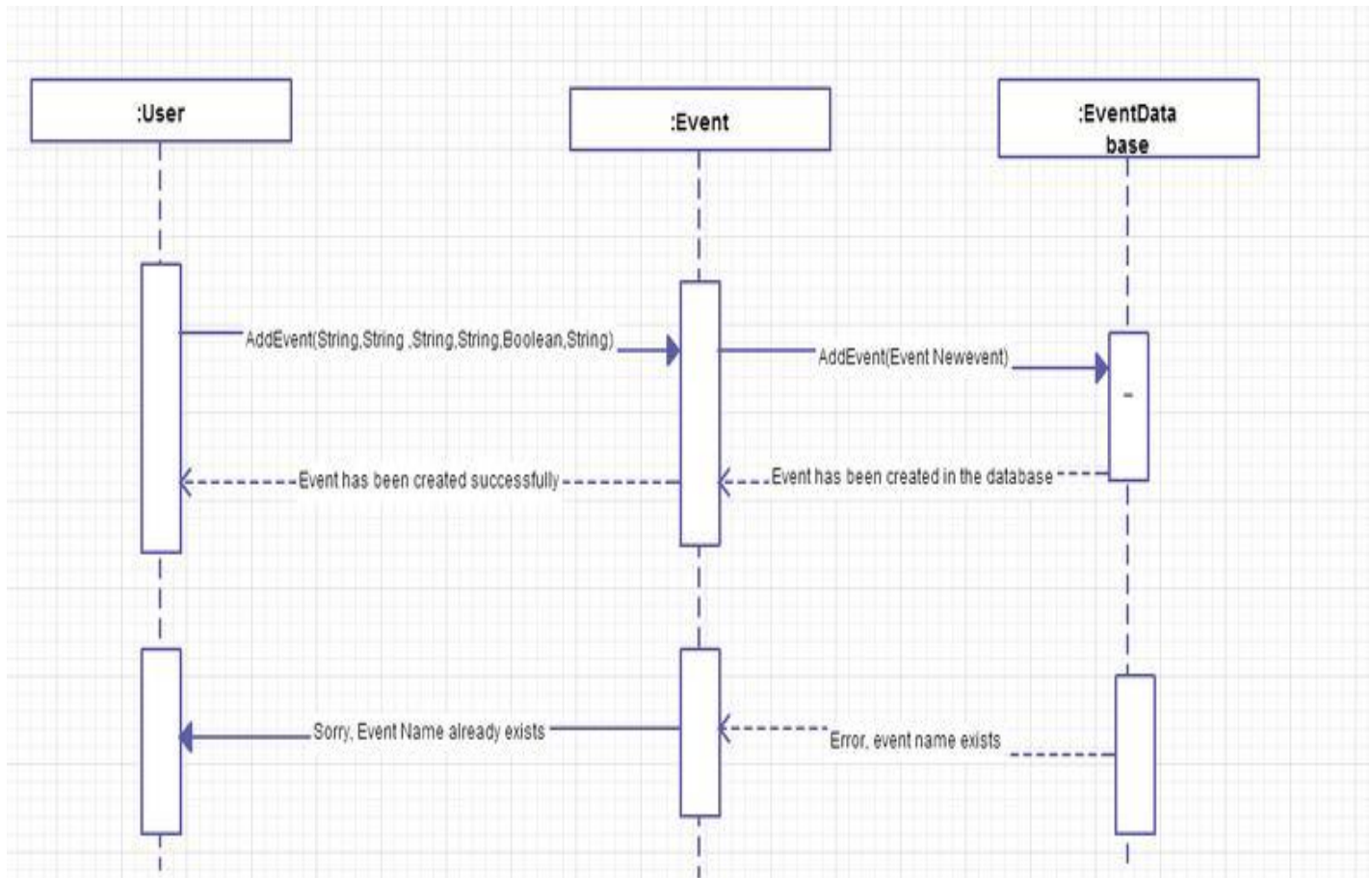
Software Design Specification

Sequence Diagram #3: "use case table #4"
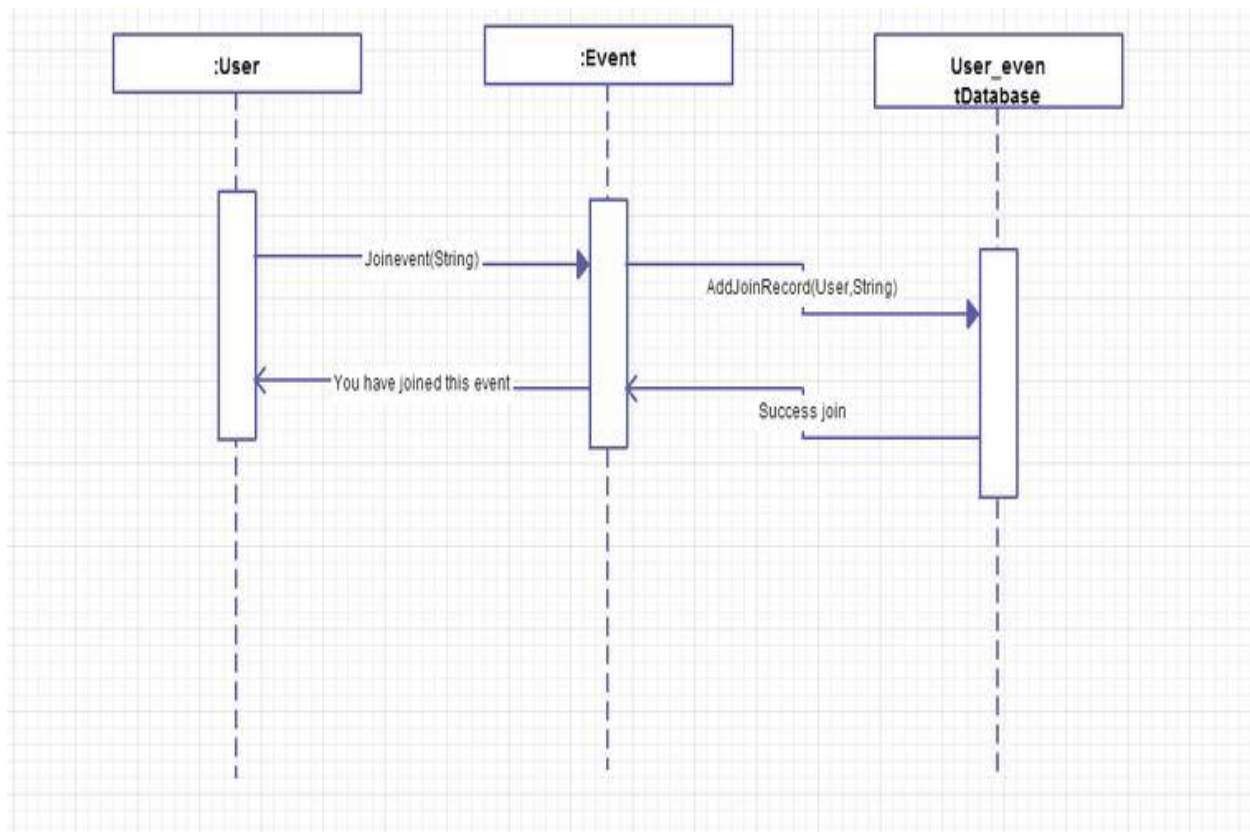
Software Design Specification

Sequence Diagram #4: "use case table #6"
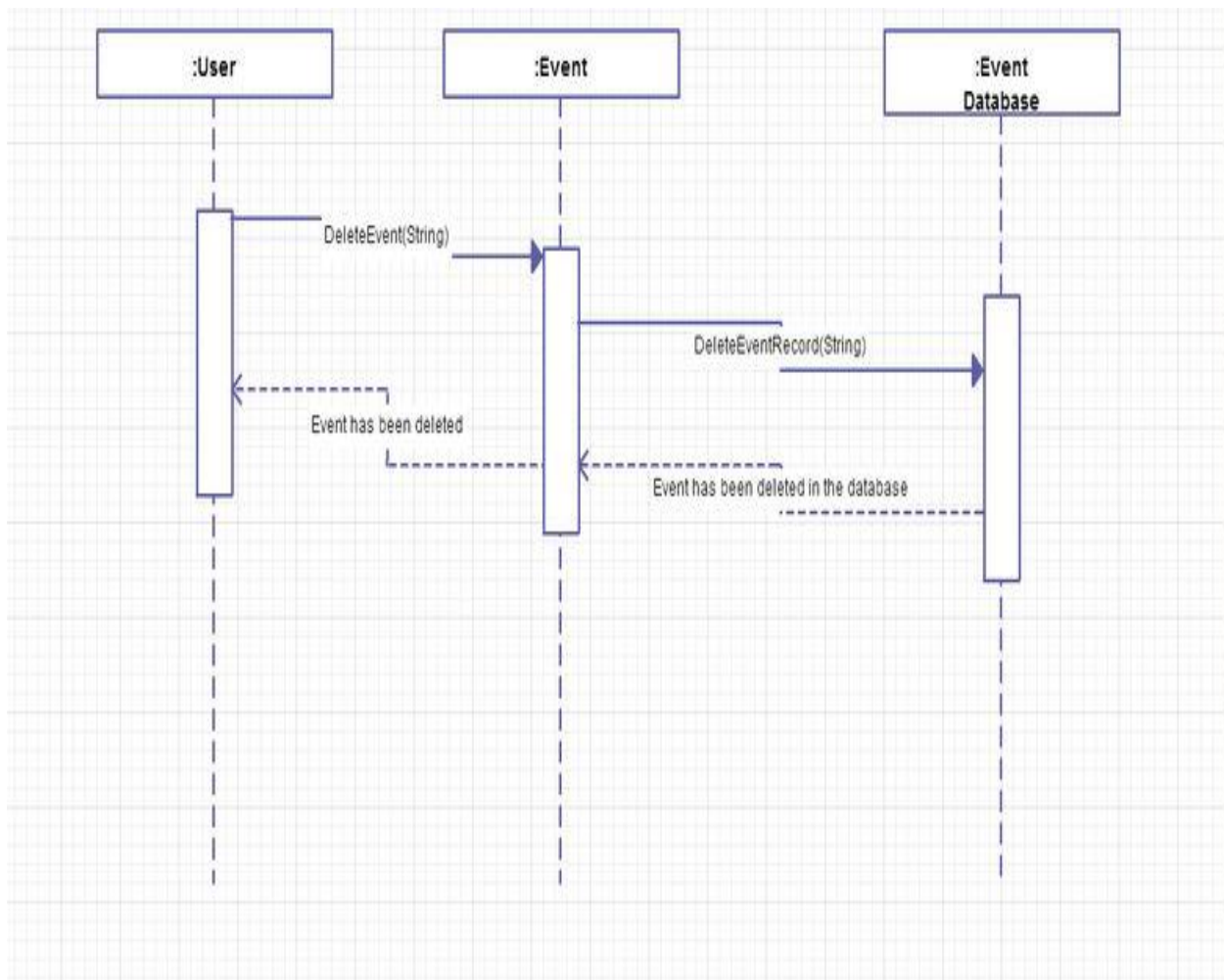
Software Design Specification

Sequence Diagram #5: "use case table #8"
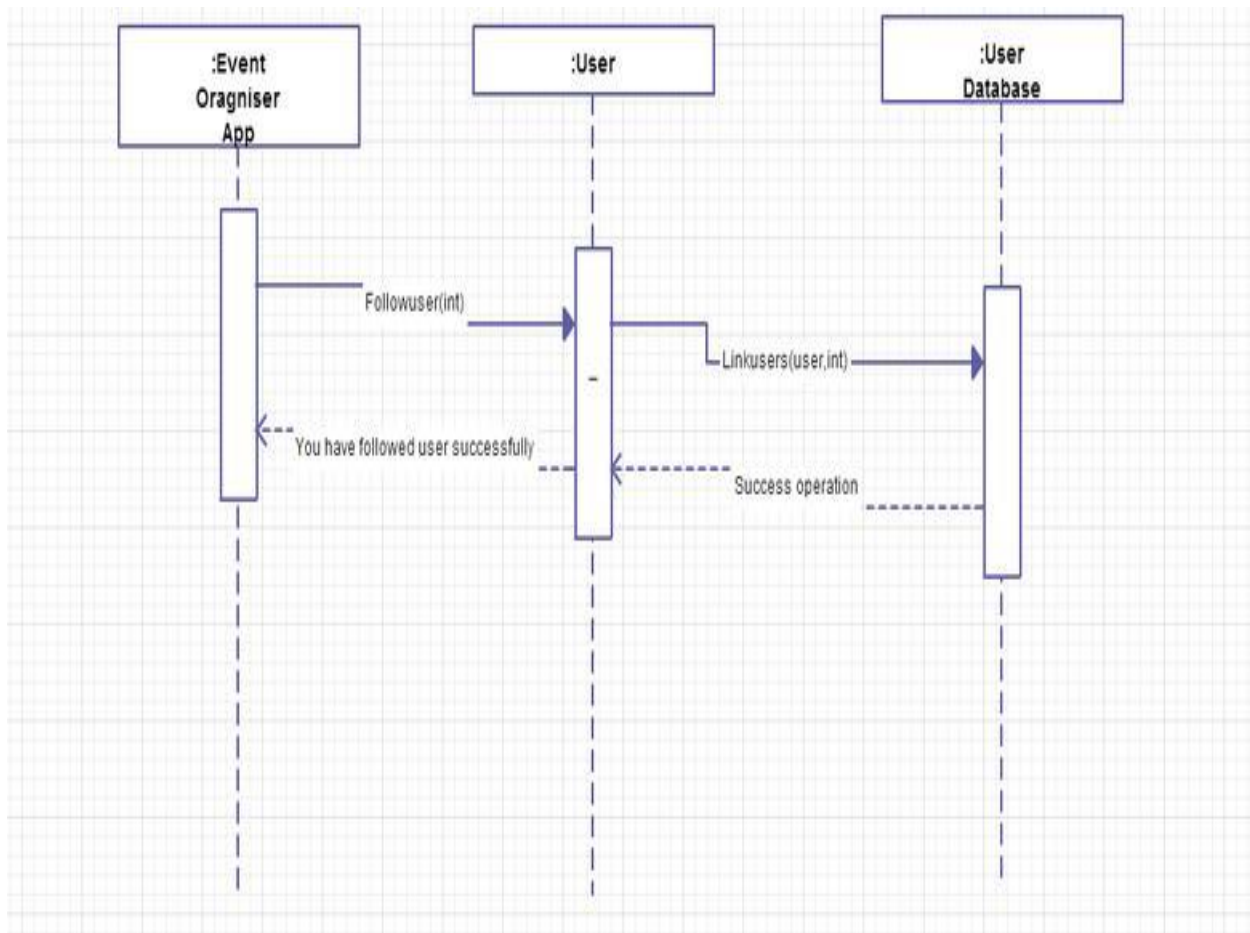
CS251: Phase 2
Project: Event Organizer
Software Design Specification

Sequence Diagram #6: "use case table #9"

Diagram content:
- :User
- :user_eventDatabase
- getRecommendedEvents(user)
- List of recommended events
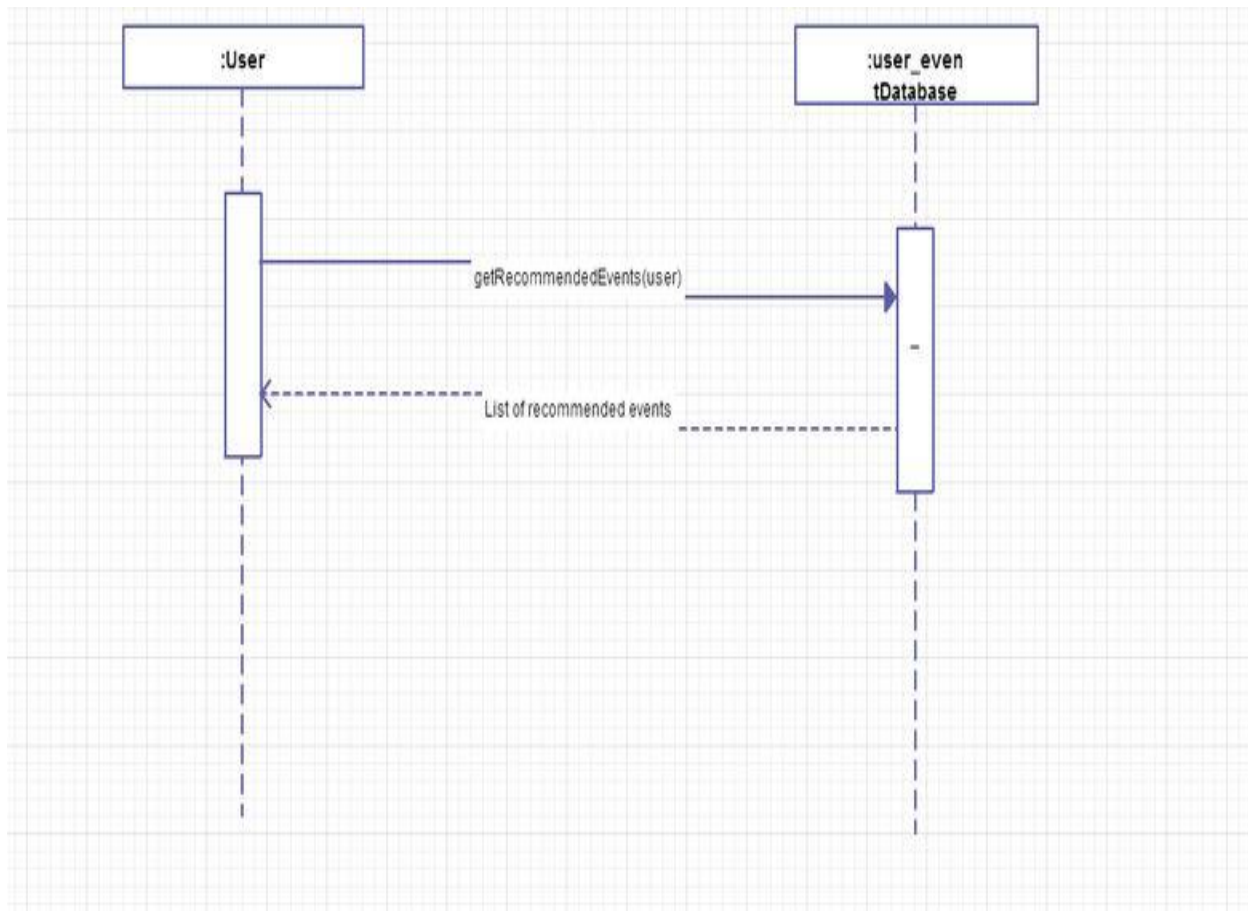
➢ **Class – Sequence Usage:**

Class Name	Sequence Diagrams	Overall used methods
User	All of sequence diagrams	signUP(), followUser()
Event	2,3,4	addEvent(),joinEvent(),deleteEvent()
EventOrganizer (as a boundary class)	1,5	Main()
UserDataBase	1,5	AddUser(),LinkUser()
EventDataBase	2,4	AddEvent(),deleteEventRecord()
User_EventDataBase	3,6	addJoinRecord(),getRecommendedEvents()

Software Design Specification

➢ **Physical Entity Relationship Diagram:**

Software Design Specification

➢ **User Interface Design:**

Screen One: "Home Screen"

Screen Two: "user's main screen"
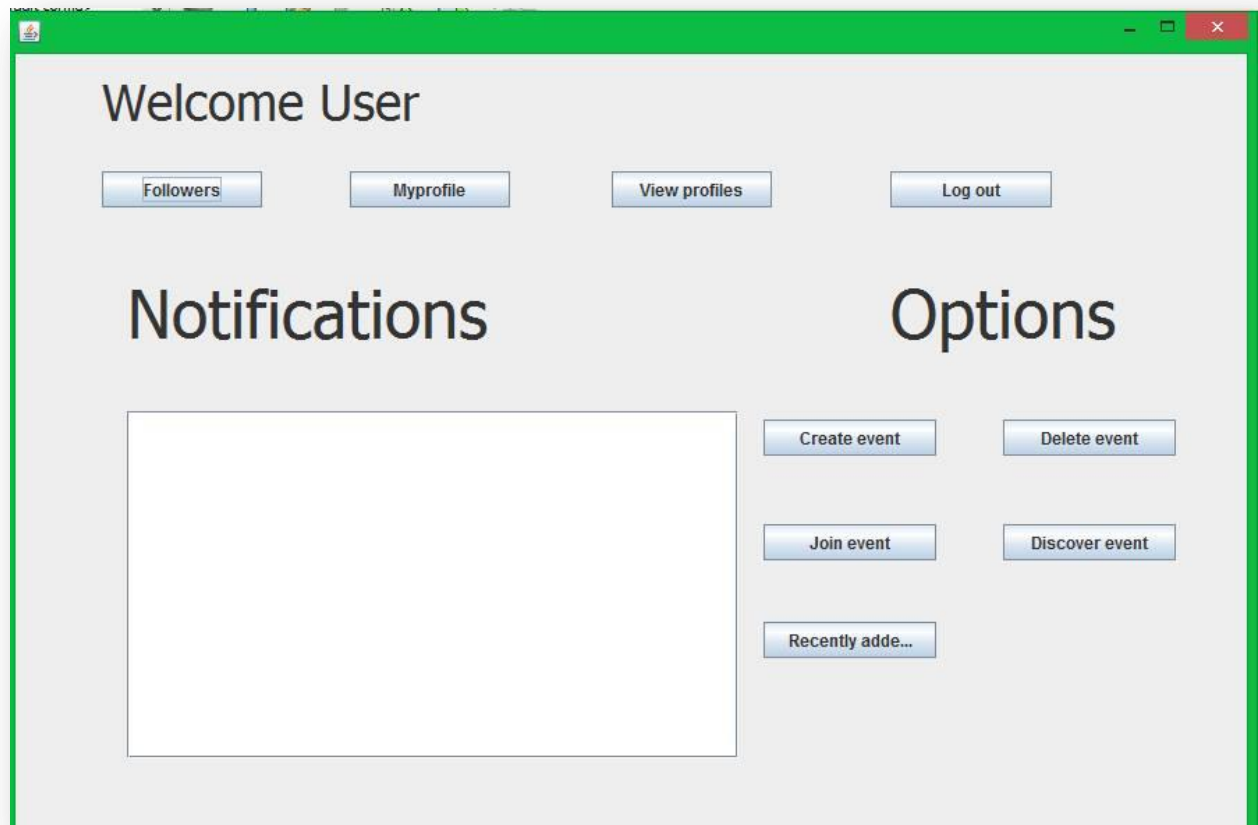
Software Design Specification

Ownership Report:

Items Table:

Item	Owner
Document purpose and audience	All team members
System Decomposition	Ahmed Saber & Ibrahim Abd El-Rheem
Class Diagram	Eslam Ibrahim & Mostafa El-Sayad
Sequence Diagrams	Mostafa El-Sayad
Physical Entity Relationship Diagram	Eslam Ibrahim
User Interface Design	Mostafa El-Sayad

Jobs Table:

Job	Team Member
Brainstorming	All team members
Typing SDS Document	Eslam Ibrahim&Mostafa El-Sayad
Reviewing SDS Document	Mostafa El-Sayad