

**DESIGN AND IMPLEMENTATION OF MULTIMEDIA EVENT  
MANAGEMENT SYSTEM (EVENTBOOK) WITH AMHARIC LANGUAGE**

**A project submitted to the school of Graduate Studies of  
Addis Ababa University**

In partial fulfillment of the requirements for Degree of Masters of  
Computer Science

By

Gizaw Tulu Bedana

\* \* \* \* \*

June 2008

Addis Ababa University  
School of Graduate Studies

DESIGN AND IMPLEMENTATION OF MULTIMEDIA EVENT MANAGEMENT SYSTEM  
(EVENTBOOK) WITH AMHARIC LANGUAGE

By

Gizaw Tulu Bedana  
Computer Science Department

Approved by:

1. Dr. Dida Midekso /Advisor/ \_\_\_\_\_
2. \_\_\_\_\_
3. \_\_\_\_\_
4. \_\_\_\_\_
5. \_\_\_\_\_

# **Acknowledgment**

I would like to forward my heartfelt thanks to my project advisor Dr. Dida Midekso for his relentless support and inspirational advises that I received throughout my stay in the University.

I also like to thank my beloved parents for their prayers and special wishes for successful completion of my MSc study.

Last but not least, I would like to forward my appreciation to my friends Tamirat Tesfaye and Fekadu Lemma for proof reading of my project.

# Table of contents

Abstract.....	1
Chapter 1 Introduction .....	2
1.1 General Background .....	2
1.2. Statements of problem .....	2
1.3 Objectives .....	3
1.4. Document Organization .....	3
Chapter 2 Literature Review .....	5
2.1. Personal Diary .....	5
2.2. Event Recording.....	6
2.3. Phone Book.....	6
2.4. Personal Website.....	6
Chapter 3: Requirements and Analysis .....	8
3.1 Functional Requirements .....	8
3.2 Non Functional Requirement .....	8
3.3 System Model.....	10
3.3.1 UML Use Case Diagram.....	10
3.3.2 Use Case Description .....	12
3.3.3 UML Class Diagram .....	26
3.3.4 UML Activity Diagram .....	28
3.3.5 UML Sequence Diagram .....	30
Chapter 4 Design: .....	31
4.1. Design Goal .....	31
4.2. Architecture .....	32
4.3. Tools .....	35
4.3.1. Web Embedding Font Tools (WEFT).....	35
4.3.2.Active Server Pages (ASP.NET 2.0).....	36
4.3.3.Internet Information Service (IIS).....	36
4.3.4. SQL Server 2005 Management Studio Express Edition DBMS .....	36
4.4. Interface Design and Usability .....	37

Chapter 5: Prototype.....	39
5.1. Web Tier – Overview.....	39
5.1.1 Configuration File (AmharicTyping.JS).....	40
5.2. Web-tier – Controller Implementation.....	40
5.2.1 Initializing the application.....	41
5.2.2. User authentication and authorization.....	41
5.2.3. Screen Flow.....	41
5.3. Web Tier – Model Implementation.....	42
5.3.1. Data Access Layer – DataHandler Classes.....	42
5.3.2. Connection state.....	42
5.4. Web Tier – View prototype.....	44
5.4.1. User –Post.....	44
5.4.2. User – View Records.....	46
5.4.3. View – System Administrator.....	47
5.5. Data Tier.....	48
Chapter 6: Testing.....	49
6.1. Application Functionality Testing.....	50
6.2 Performance Testing.....	52
6.3. Results and discussions.....	52
Chapter 7: Conclusion and Future Works.....	54
7.1 Conclusion.....	54
7.2 Future Works.....	56
Reference.....	57
Glossary.....	60
Annex A: AmharicTyping.JS.....	i
Annex B: Sequence Diagram.....	xii
Annex C: Amharic Keyboard Layout.....	xxxvi

# List of Tables

Table 6.1 Testing the login page.....	51
---------------------------------------	----

# List of Figures

Figure 3.1 use case Diagram of EventBook.....	11
Figure 3.2 Class Diagram of EventBook .....	27
Fig: 3.3 Activity Diagram of EventBook.....	29
Figure 4.1 Three Tier Architecture of EventBook .....	33
Figure 5.1: MVC design pattern within a three tier architecture of EventBook.....	42
Figure 5.2: Establishing a connection with the database. ....	43
Fig 5.3 Connection to the database .....	44
Figure 5.4: Screen shot of index.aspx page .....	45
Figure 5.5: Screen shot of the SearchEvent.aspx.....	47
Figure 5.6: Screen shot of DeleteAccount.aspx .....	48
End Class.....	51
Figure 6.1: Code listing of LoggingIn.aspx.....	51
Figure 6.2: LoggingIn.frm of EventBook.....	52
Table 6.1 testing the login page.....	53
FIG 4.5: Sequence Diagram for Create Account Use Case .....	xii
FIG 4.8: Sequence Diagram for Login Use Case .....	xiii
FIG 4.10: Sequence Diagram for Post Diary Use Case .....	xiv
FIG 4.12: Sequence Diagram for Post Events Use Case .....	xv
FIG 4.14 Sequence Diagram for Read Diary Use Case .....	xvi
FIG 4.16: Sequence Diagram for Read Event Use Case.....	xvii
FIG 4.17: Sequence Diagram for Read Events Alternate Course A .....	xviii
FIG 4.18: Sequence Diagram for Post Address Book Use Case.....	xix
FIG 4.19: Sequence Diagram for Post Address Book Alternate Course A.....	xx
FIG 4.20: Sequence Diagram for Create Personal Website Use Case.....	xxi

FIG4.21: Sequence Diagram for Create Personal Webpage Alternate Course A.....	xxii
FIG 4.22: Sequence Diagram for Create Personal Website Course B .....	xxiii
FIG 4.23: Sequence Diagram for Read website Use Case .....	xxiv
FIG4.24: Sequence Diagram for Read Personal Webpage Alternate Course A .....	xxv
FIG 4.25: Sequence Diagram for Delete Account Use Case .....	xxvi
FIG 4.26: Sequence Diagram for Edit Address Book Use Case.....	xxvii
FIG 4.27: Sequence Diagram for Delete Account Alternate Course A.....	xxviii
FIG 4.28: Sequence Diagram for Edit Address Book Use Case.....	xxix
Fig 4.29 Keyboard layout of EventBook System .....	xxx

## Abstract

This report describes the design and implementation of Multimedia Event Management system (EventBook) with Amharic Language over the web. EventBook allows both local and international users to post and read their personal diary, record special events, manage address book and simple personal website. The system also allows the users to use Amharic language as a means for reading and posting.

A number of architectures and design models were examined for this purpose but EventBook design pattern within three tier architecture was found to be the most suitable approach. Although a number of technologies such as Microsoft visual studio 2005 and Microsoft SQL server 2005 management studio express edition have been used for implementing the designed architecture, ASP 2.0 was deemed to be the best approach for web.

Finally, System prototype is implemented and tested to check whether the specified user's requirement is fulfilled or not.

# Chapter 1

## Introduction

### 1.1. General Background

Web-based diaries, event recording (blogs), address books and personal websites (hereafter referred as EventBook) are becoming increasingly common in both commercial and educational institutions for a number of reasons. Commercially, this kind of system is used in large institutions where it serves as effective means of organizing schedules. Scientifically, they can be used to carry out studies, researches or projects. Almost all Event recording sites are in English; few in French, and other languages also, but none of them are in Amharic.

### 1.2. Statement of the problem

EventBook are very helpful in many situations where people are interested to write their diaries, and visit others public events, to have their personal sites and to access their friends address online. Most of the services of the web are available in foreign language like English. However, using native language is of paramount importance in order to facilitate communication

Besides, it is natural for an individual to describe his/her idea better using his/her mother language than otherwise. Unfortunately, when we consider the local participants the inability of many Ethiopians to describe their ideas in English hinders to exploit their ideas. This poses a huge problem and requires considering Amharic multimedia Event management System on the web.

### 1.3. Objective

The main objective of this project is to develop a system that allows any individual to create his/her own Amharic EventBook in an efficient and simple manner with minimal effort.

Furthermore, a secondary aim of this project is to extend this facility to Internet users without the need to load any Amharic software on user's machine. Therefore, the system will also:

- Facilitate and improve communication between Amharic Language users.
- Make communication easy to use and adopt.
- Curtail the time taken by users to get information wherever they are, since, the system use Internet.

### 1.4. Document Organization

This report has seven chapters including this chapter. Chapter two 'Literature Review' gives a quick glimpse of what EventBook system looks like and the components of the system by comparing with similar systems.

Chapter three 'Requirements and Analysis' illustrate the functional and nonfunctional requirements of EventBook. It also shows the system models including use case models, object models, dynamic models and use case descriptions.

Chapter four 'Design' surveys the three-tier architecture, tools and techniques; and show how this architecture will be used for EventBook design.

Chapter five 'Prototype' shows how EventBook will be implemented; this chapter also show the sample codes and user interface.

Chapter six 'Testing' shows the performance and functionality test of EventBook after implementation.

Chapter seven 'Conclusion and Future work' describes the lesson I learned throughout this project and my future work for EventBook.

In the dynamic world of the Internet, sometimes it isn't enough to just have important web sites that have different features; we also need to incorporate different language. So far in this report, you'll see how to embed Amharic font with WebPages.

## Chapter 2

### Literature Review

There are different diaries, blogs, and phonebook systems that are available over the web, of which most of them use English language as a medium of communication.

This section gives a brief background to existing software applications that provide similar functionality with EventBook.

#### Personal Diary

Most of the personal diary sites provide a mechanism to keep/store information about a memoir, autobiography or biography, it is generally not written with the intention of being published as it stands. In recent years, however, there is internal evidence in some diaries that they are written with eventual publication in mind, with the intention of self-vindication, or simply for profit [14].

Many electronic diary systems have templates for daily, weekly, monthly or random entries. These sites have been designed to allow journals and diary writers to capture their thoughts as well as images, links to other sites.

A web based diary has been made available for all the web users, it allows computerized diary. Users can post, read and view others public diaries.

Furthermore, Internet users do not have access to this site's architecture and therefore, it was not possible to analyze the design or features implemented by this application in detail without actually testing it.

## **Event recording**

Many blogs provide commentary or news on a particular subject; others function as more personal online diaries. A typical blog combines text, images, and links to other blogs, web pages, and other media related to its topic. The ability for readers to leave comments in an interactive format is an important part of many blogs. Most blogs are primarily textual, although some focus on art (artlog), photographs (photoblog), sketchblog, videos (vlog), music (MP3 blog), audio (podcasting) are part of a wider network of social media. Micro-blogging is another type of blogging which consists of blogs with very short posts [15].

Most of the events are posted by different bloggers. This will help to share the information with others.

## **Phone book**

Most of the sites that provide a means to store the details in alphabetical order of people's names, although in paper-based address books entries can easily end up out of order as the owner inserts details of more individuals or as people move. Many address books use small ring binders that allow adding, removing and shuffling of pages to make room. But the automated one can simply sort by the name or with other parameters [16].

## **Personal website**

Personal web pages may be as simple as a single page or may be as elaborate as an online database with gigabytes of data. Many Internet service providers offer a few megabytes of space for customers to host their own personal web pages.

The content of personal web pages varies and can, depending on the hosting server, contain anything that any other websites do. However, typical personal web pages contain images, text and a collection of hyperlinks. Many can contain biographical information, résumés, and blogs. Many personal pages will include information about the author's hobbies and pastimes, and information of interest to friends and family of the author.

As we can see in this review, different recording systems are available on the internet; most of them use English language. In the architectural design, there is an event recording system designed by English language in combination with Indian language. This system helps to switch from one language to the other. [20]

As far as language concern, there is no difference in the environment of recording, and architecture, rather the main issue is to assist the user to use local language.

The most challenging issue is to find the internal architecture of system, contacting the web masters of a page through email is one option (address taken from the footer of a page) but most of them is not in the position to mail back; the other is not willing to provide the information. The other option is to download the website and to see the source code written in HTML, most of them use client side code of XML and XHTML, and HTML language in combination.

The most challenging in the design of event book system is:

- To find the information about the internal architecture of a other similar systems
- Finding the appropriate terms for Amharic language.
- Incorporating two or more language in a single data entry text boxes.

## Chapter 3

# Requirements and Analysis

This chapter outlines the functional and non functional requirements, use case and dynamic modeling diagrams.

### 3.1 Functional Requirement

This project's primary concern is to develop a system that allows Amharic Language users to post, read events like personal diary, event record, and provide user friendly simple personal sites using Amharic Language in an efficient and simple manner with minimal effort.

The system also be designed in such a way that the administrator(s) will be able to delete user account, and modify or delete events.

### 3.2 Non Functional Requirements

The non-functional requirement concerned with the system that must be carried out in addition to functional requirements stated. These requirements include security, Performance Characteristics, maintainability, portability, Error Handling and Extreme Condition, and efficiency.

## **Security**

Some of the security issues that are looked into the system include the following:

- EventBook shall grant access to authenticated users only
- EventBook shall have password encryption mechanism so as to make the system more secured
- EventBook shall grant access to users based on their responsibilities (users and administrators have different responsibilities in the system)

## **Performance Characteristics**

The system runs on an internet environment. All users on different computers can access the system at the same time without any performance problem.

## **Maintainability**

The system should be designed in such a way that it is easy to be modified.

## **Portability**

The system should be easy loaded to different machines.

## **Error Handling and Extreme Condition**

EventBook shall be an integrated system to trap errors. It should be able to respond descriptive error messages. Appropriate error message will be displayed when an unexpected input such as incorrect user name and/or password are encountered. Example: when the expected input is string the user, will not be able to provide numerical values.

## **Efficiency**

The system should not take too much time to log a user in. Each page should not take too much time to be loaded.

### **3.3 System Models**

A system model is an abstraction describing a problem domain and/or a solution to a problem domain (Ambler 2004).

#### **3.3.1 UML Use Case Diagrams**

UML use cases provide textual descriptions of the functionality of the system from the user's point of view. Figure 3.1 depicts use case diagram for the EventBook system.

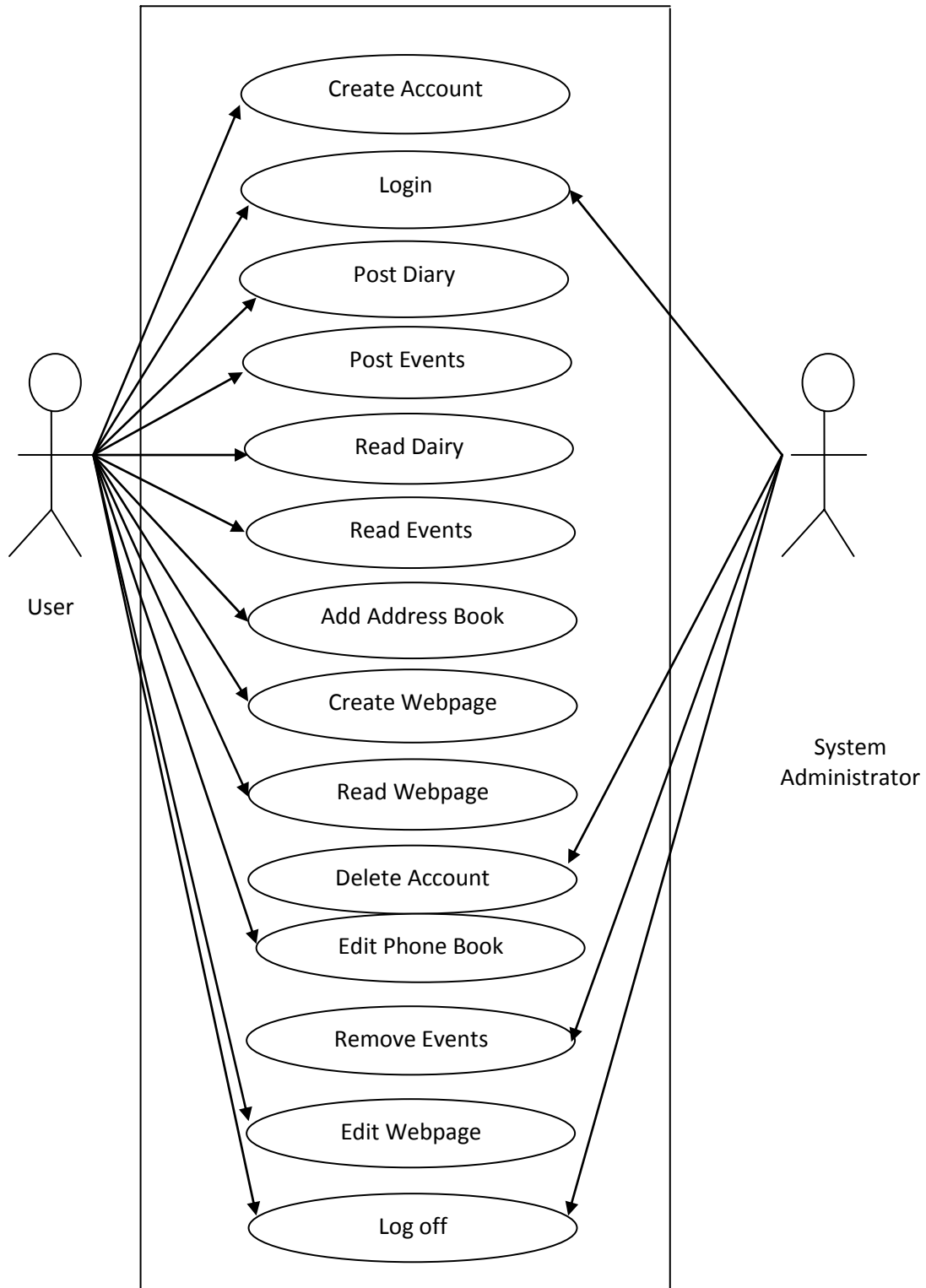


Figure 3.1 use case Diagram of EventBook

### 3.3.2 Use Case Description

**Name:** Create Account

**Identifier:** UC 01

**Description:** Allows user to create an account in EventBook System

**Precondition:** The user should activate EventBook System

**Post Condition:** Account will be created

**Basic Course of Action:**

1. The user activates “Create Account” option
2. The user completes and submits the Create Account form providing personal information (Alternative Course A: *Blank Field or Incorrect Information*)
3. The system registers the user (Alternative Course B: *User ID already exists*)
4. The system displays acknowledgment message
5. The use case ends

**Alternate Course A:** *Blank Field or Incorrect Information*

- A.2 The system displays error message
- A.3 The use case resumes at step 2

**Alternate Course B:** *User ID already exists*

- B.3 The system displays “User name already exists” error message
- B.4 The use case resumes at step 2

**Name:** Log In

**Identifier:** UC 02

**Description:** Allows the user to log into its account

**Precondition:** The user should have account at EventBook System

**Post Condition:** The user will access his/her homepage

**Basic Course of Action:**

1. The user activates EventBook Home page
2. The user completes and submits user ID and password (Alternative Course A: *User forgot its password*)
3. The system displays main Menu page (Alternative Course B: *Incorrect User ID or password*)
4. The use case ends

**Alternate Course A:** *User forgot its password*

- A.2 The user activates “Forgot password?” option
- A.3 The user completes and submits the security question and the answer
- A.4 The system displays “Change Password” form
- A.5 The user completes and submits the form
- A.3 The use case resumes at step 3

**Alternate Course B:** *Incorrect User ID or password*

- B.3 The system displays “Incorrect user ID or password” error message
- B.4 The use case resumes at step 2

**Name:** Post Diary

**Identifier:** UC 03

**Description:** Allows the user to Post his/her Diary

**Precondition:** The user should Log into EventBook System

**Post Condition:** The user will submit his diary to the system

**Basic Course of Action:**

1. The user activates “የግል ግጽ” option
2. The user completes and submits the subject, and the diary Alternative Course A: *Blank field*)
3. The system displays confirmation message
4. The use case ends

**Alternate Course A:** *Blank field*

- A.2 The system displays error message
- A.3 The use case resumes at step 2

**Name:** Post Events

**Identifier:** UC 04

**Description:** Allows the user to Post public events (Blog)

**Precondition:** The user should Log into EventBook System

**Post Condition:** The user will submit events (Blog) to the system

**Basic Course of Action:**

1. The user activates “*σγυ&C*” option
2. The user completes and submits the subject, and the Message, Pictures (if any)( Alternative Course A: *Blank field*)
3. The system displays confirmation message
4. The use case ends

**Alternate Course A:** *Blank field*

- A.2 The system displays error message
- A.3 The use case resumes at step 2

**Name:** Read Diary

**Identifier:** UC 05

**Description:** allows the user to read his/her own Diary

**Precondition:** The user should log into EventBook System

**Post Condition:** The user reads his diary

**Basic Course of Action:**

1. The user activates “የግል ማህደር ለማንበብ” option
2. The system displays list of diaries by date (Alternative Course A: *No diary to read*)
3. The user selects the diary to read
4. The system displays the diary to the user
5. The use case ends

**Alternate Course A:** *No diary message to read*

- A.2 The system displays “No diary to read” message
- A.3 The use case resumes at step 5

**Name:** Read Events

**Identifier:** UC 06

**Description:** allows the user to read events post by EventBook Users

**Precondition:** The user should log into EventBook System

**Post Condition:** The user reads Events

**Basic Course of Action:**

1. The user activates “**ማህደር ለማንበብ**” option
2. The system displays Events chronologically (Alternative Course A: *No Event to read*)
3. The user selects the Events to read
4. The system displays the Events to the user
5. The use case ends

**Alternate Course A:** *No Event to read*

- A.2 The system displays “No event to read” message
- A.3 The use case resumes at step 5

**Name:** Add Address Book

**Identifier:** UC 7

**Description:** Allows the user to keep contact addresses list

**Precondition:** The user should Log into EventBook System

**Post Condition:** Contact address will be added to address book

**Basic Course of Action:**

1. The user activates “የአድራሻ ደብተር” option
2. The user completes and submits the name, father’s name, grand father’s name, nick name email and contact address (Alternative Course A: *Blank field or incorrect information*)
3. The system adds the contact address
4. The system displays confirmation message
5. The use case ends

**Alternate Course A:** *Blank field or incorrect information*

- A.2 The system displays “Required fields cannot be blank” error message
- A.3 The use case resumes at step 2

**Name:** Create Webpage

**Identifier:** UC 08

**Description:** Allows user to create simple personal website

**Precondition:** The user should activate EventBook System

**Post Condition:** Personal Website will be created

**Basic Course of Action:**

6. The user activates “*ጽሁፊ ገፅ*” option
7. The user completes and submits the Create Personal website form providing personal information (Alternative Course A: *Blank Field or Incorrect Information*)
8. The system create a personal Site(Alternative Course B: *Website already exists*)
9. The system displays acknowledgment message
10. The use case ends

**Alternate Course A:** *Blank Field or Incorrect Information*

- A.2 The system displays error message
- A.3 The use case resumes at step 2

**Alternate Course B:** *website already exists*

- B.3 The system displays “Website already exists” error message
- B.4 The use case resumes at step 2

**Name:** Read Webpage

**Identifier:** UC 09

**Description:** allows the user to read personal website by EventBook Users

**Precondition:** The user should type the full address of site name in the web browser

**Post Condition:** The user reads Personal website

**Basic Course of Action:**

1. The user type the full address of site name in the web browser
2. The system displays Website (*Alternative Course A: Unknown website name*)
3. The system displays the website to the user
4. The use case ends

**Alternate Course A:** *Unknown website name*

A.2 The system displays "Page cannot be displayed" message

A.3 The use case resumes at step 4

**Name:** Delete Account

**Identifier:** UC 10

**Description:** Allows the administrator to delete accounts

**Precondition:** The administrator should Log into EventBook System

**Post Condition:** Selected account will be deleted

**Basic Course of Action:**

1. The administrator activates "Delete Account" option
2. The administrator completes and submits the user ID to be deleted
3. The system deletes the account
4. The system displays confirmation message
5. The use case ends

**Name:** Edit Phone Book

**Identifier:** UC 11

**Description:** Allows the user to edit its Address book

**Precondition:** The user should Log into EventBook System

**Post Condition:** The Address will be edited successfully

**Basic Course of Action:**

1. The user activates “የአድራሻ ለውጥ” option
2. The user edit and submits its Address Book
3. The system modifies the Address book
4. The system displays confirmation message
5. The use case ends

**Name:** Remove Events

**Identifier:** UC 12

**Description:** Allows the administrator to delete events

**Precondition:** The administrator should Log into EventBook System

**Post Condition:** Selected event to be deleted

**Basic Course of Action:**

1. The administrator activates "Delete Event" option
2. The administrator select the event to be deleted
3. The system deletes the event
4. The system displays confirmation message
5. The use case ends

**Name:** Edit webpage

**Identifier:** UC 13

**Description:** Allows the user to edit its own Personal Website

**Precondition:** The user should Log into EventBook System

**Post Condition:** The personal website will be edited successfully

**Basic Course of Action:**

1. The user activates “*ⓧ⓪Ⓛ 1⓪*” option
2. The user edit and submits its personal website
3. The system modifies the personal website
4. The system displays confirmation message
5. The use case ends

**Name:** Log off

**Identifier:** UC 14

**Description:** The user will be log out off the system

**Precondition:** The user should log into EventBook System

**Post Condition:** user will be redirected to EventBook Home Page

**Basic Course of Action:**

1. The user activates “logout” option
2. The system redirects to EventBook Home Page
3. The use case ends

### 3.3.3 UML Class Diagram

In the Unified Modeling Language (UML), a class diagram is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, and the relationships between the classes. The class diagram of EventBook is depicted below:

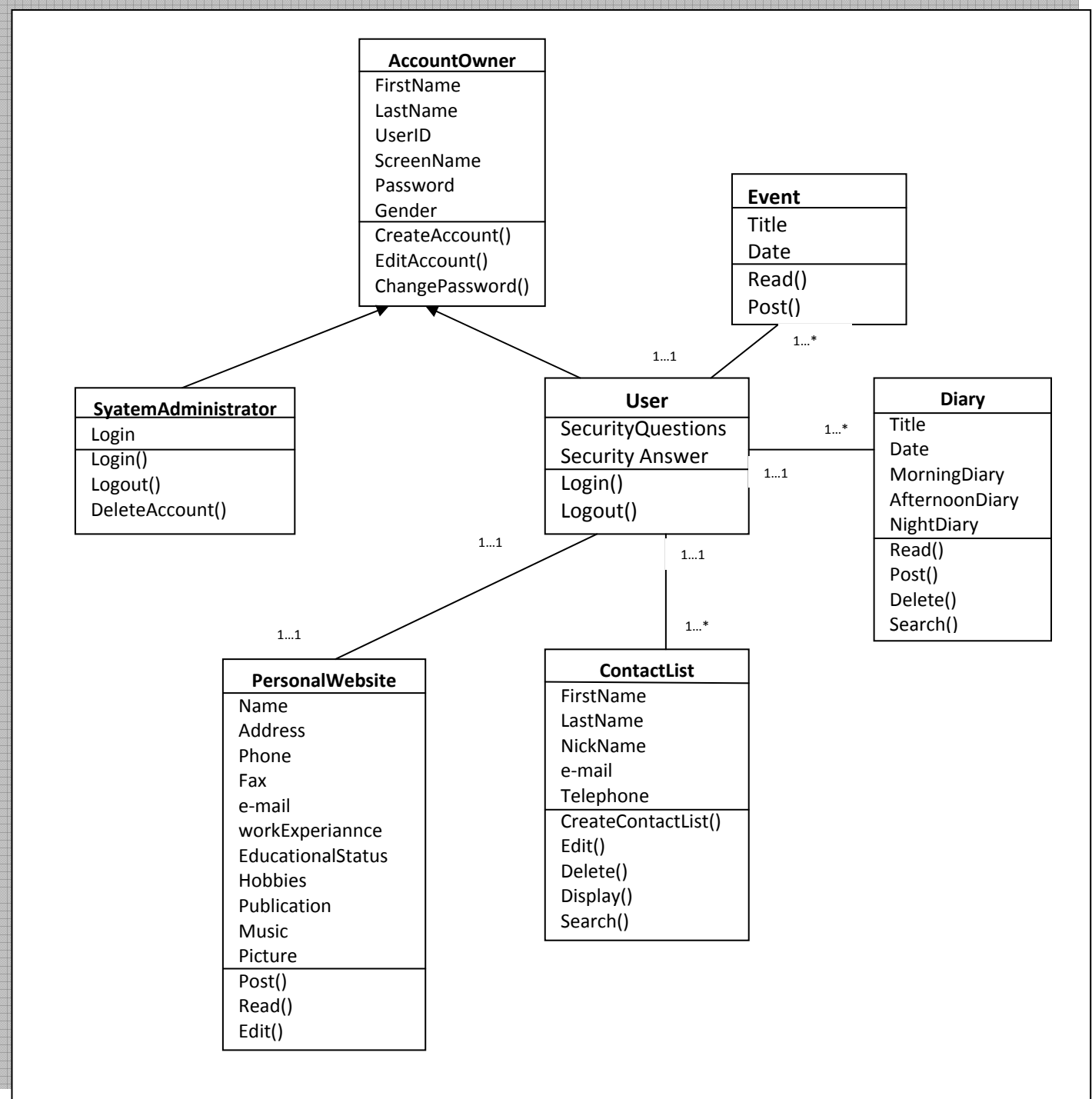


Figure 3.2 Class Diagram of EventBook

### 3.3.4 UML Activity Diagrams

An activity diagram is a variation of a state chart that focuses on a sequential flow of activity within an object. In an activity diagram, states are activities, each of which represents the execution of an operation. The transition out of a state is usually triggered by the completion of the respective operation **[13]**. The UML use case diagrams were used to create the UML activity diagrams for each of the functional units of EventBook. The Activity Diagram of EventBook is depicted below.

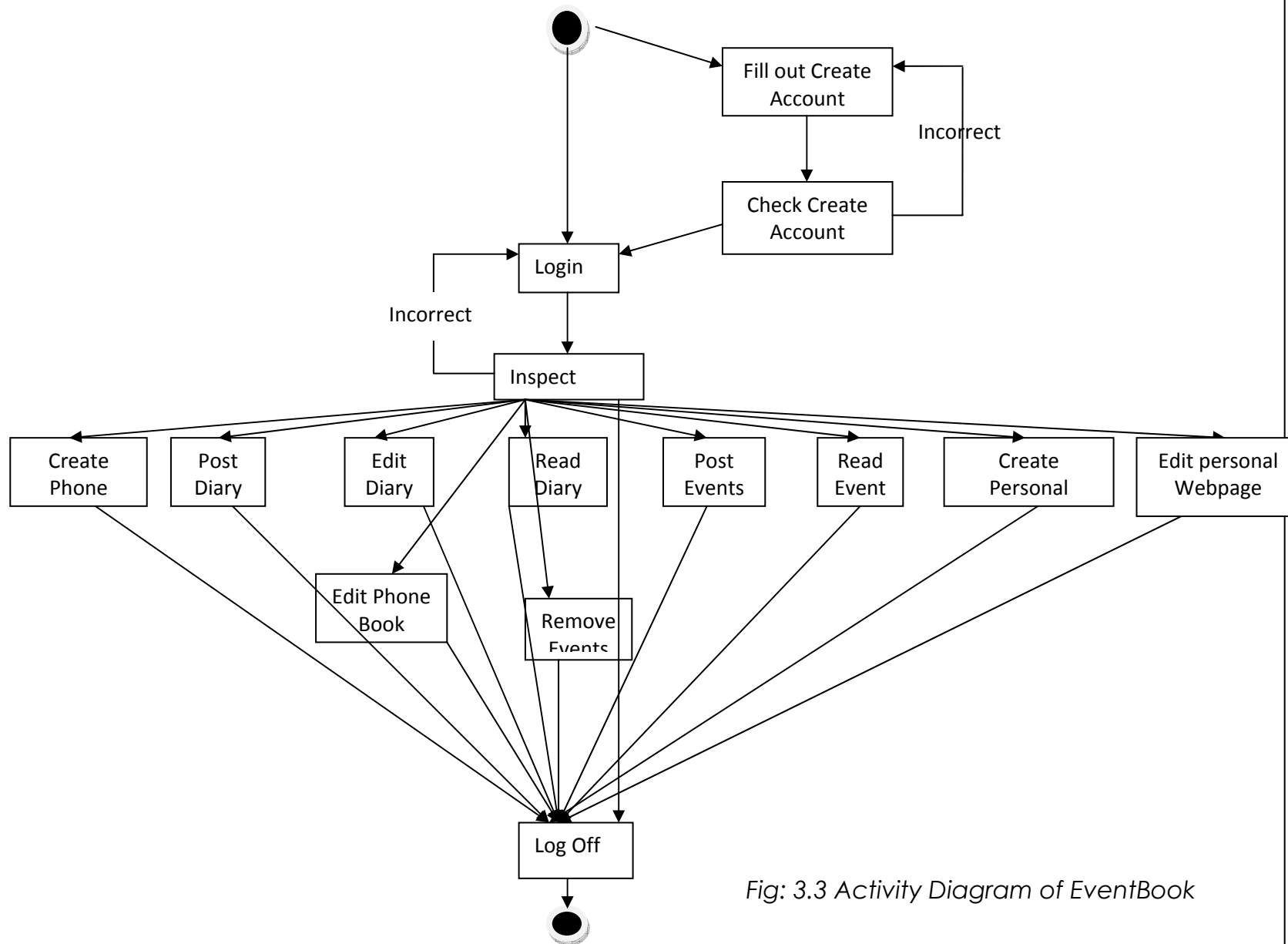


Fig: 3.3 Activity Diagram of EventBook

### 3.3.5 UML Sequence Diagrams:

A sequence diagram shows an interaction between objects arranged in a time sequence. A sequence diagram aims to represent the detailed object interaction that occurs for a single use case; therefore these diagrams can be seen as an expansion of use cases to the lowest possible level of detail [13]. Sequence diagrams drawn for EventBook can be seen in annex B.

## Chapter 4

### Design

This chapter presents the design goal, architecture, tools, and techniques used to design EventBook

#### 4.1 Design Goal

Web based EventBook serves as an ideal solution to the above concerns and like in any web based applications; EventBook also must satisfy the following:

- **Security:** Security is of paramount importance in any web application, both from the point of view of the owner and the users of the application. Therefore, EventBook should only be accessible to authorized users and furthermore all data stored in the database must be secure.
- **Efficiency:** Users are becoming increasingly unwilling to wait for pages to load; therefore the speed at which the site operates is vital for its success. In order to minimize the time it takes to generate (i.e. download time) and dispatch a new page for EventBook, interface design will not include any large graphic files and middle-tier processing code will be made as efficient as possible.
- **Robustness:** A web application should be rigorous enough to handle errors or exceptions. Even though sometimes errors in such applications are inevitable because of the factors that are out of the developer's hand, they should be dealt correctly, either by trying to solve the problem or displaying a helpful error message.

- **Usability:** Usability is an important consideration for web applications. Therefore, the EventBook should be easily navigable. Also, the interface of EventBook will be designed while keeping in mind Nielsen's Usability Heuristics [5].

## 4.2 Architecture

Architecture is one of the first considerations while developing any web based application. At a high level, the architecture of an application defines how different parts of the system are organized and logically separated yet ensuring that they work together. Furthermore, architectures can be split into a tiers and it is the concept of tiers that provides a convenient way to group different classes of architectures [19].

Designing the architecture in tiers is useful because it supports layering, promotes scalability and allows easy maintenance of the application. At present, three-tier architectures are most commonly employed in web based applications.

The EventBook architecture consists of the following tiers; client tier, web tier and the data tier as illustrated in the Figure 4.1.

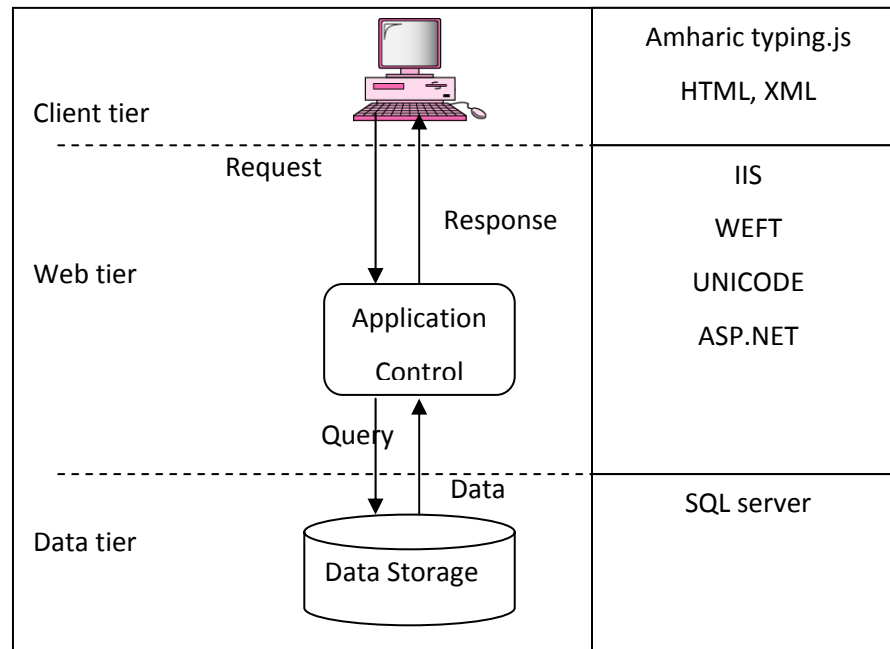


Figure 4.1 Three Tier Architecture of EventBook

**Client tier:** is the topmost layer. Its primary concern is to display the results of any user requests. This layer, in most cases will consist of a web browser running on the user's machine.

EventBook in this tier map the existing keyboard layout into UNICODE Amharic fonts.

**Web tier (Business logic layer):** This layer is primarily concerned with business logic (or tasks). Business logic includes [20] [19]:

- Validation of input
- Performing calculations (or tasks)
- Keeping track of sessions
- Access and retrieval of any data required for the presentation layer

EventBook this tier uses Web Embedding Font Tools (WEFT) to temporarily load Amharic fonts to the website. When the website goes off the font will be discarded.

**Data tier:** This layer is concerned with data storage and persistence issues and can be implemented by using a database or some sort of file storage mechanism. The data can either be stored on the web server itself or on a different machine; however it needs to be easily accessible by the web server.

EventBook in this tier use *SQL\_Latin1\_General\_CP850\_BIN* collation to store Amharic fonts directly in to the database

An alternative approach to three-tier architecture is the page-centric approach (two-tier architecture). The applications built using this approach normally consist of one or more programs running on the client's machine communicating over the network with a server based application. Such applications can be built by using a collection of ASPs that has direct access to database for servicing a client's request.

During the implementation of a project, a prototype was built using the "page centric with ASP.NET 2.0". However, on completion of the prototype, it was realized that

- The application's code is a mixture of presentation, business and data access logic.
- The application's business and presentation logic code is not reusable.
- The prototype application did not support scalability.
- It will be difficult to enhance or change a part of the system's functionality, if required to do so at a later stage.

In addition to the above factors, the main disadvantages of using this approach are as follows [21] [22]:

- Changing business logic layer involves recompiling and redeploying the client layer.
- In efficient use of network services as database drivers have to be installed and configured on the client tiers.
- Database connection costs are high because each client will have to establish a database connection.

Thus, by taking these factors into account, “three-tier architecture” is selected for EventBook as compared to “page centric approach”. Also, another key reason for choosing a three-tier architecture is that it splits the application into manageable components by allowing these components to exist as separate entities at each level and therefore each component can be developed in parallel to each other. Further benefits of using a three-tier architecture include:

- The resource allocation is more flexible because the middle tier can be changed dynamically according to the needs.
- The middle tier can also be used by other applications.
- Change in one tier does not necessarily have an effect on the other tier (e.g. data structure in the database can be changed without affecting the client tier) [23].

## 4.3 Tools

This section examines the technologies that can be used in the implementation of a web application that is structured within three tier architecture.

### **Web Embedding Font Tools (WEFT)**

**Web Embedding Fonts Tool**, or **WEFT**, is Microsoft's utility for generating embeddable web fonts. WEFT is used by webmasters to create 'font objects' that are linked to their web pages so that users using Microsoft's Internet Explorer web browser will see the

pages displayed in the font style contained within the font object. WEFT scans the HTML document file(s), the TrueType font file(s), and some additional parameters. It adjusts the HTML files and creates "Embedded OpenType" files for inclusion on the web site. These files usually use the extension ".eot". WEFT can embed most fonts, but it will not embed fonts that have been designated as 'no embedding' fonts by their designers. WEFT may reject other fonts because problems have been identified. Embedded fonts are widely used to generate non-English language websites [28].

Since EventBook uses Amharic Language, any user need not install the Amharic (Ethiopic) font on his/her machine in order to write, read or access the site. Therefore, the developer has to map each UNICODE value of the Amharic font to the keyboard layout.

### **Active Server Page (ASP.NET 2.0)**

The web tier of the application will consist of a collection of ASP 2.0, Active x controller and Visual Basic.NET 2005, which need to run on a web server. The basic task of the web server is to listen for HTTP (where HTTP is a protocol used to transfer information on the World Wide Web) requests on a network, receive HTTP requests by the user agents, serve the request, and return the HTTP responses that contain the required resources [1].

### **Internet Information Service (IIS)**

IIS which is part of the Windows Operating System, is a popular web server in windows application,

IIS is software services that support Web site creation, configuration, and management along with other Internet functions. Internet Information Services include Network News Transfer Protocol (NNTP), File Transfer Protocol (FTP), and Simple Mail Transfer Protocol (SMTP) [3].

## SQL Server 2005 management Studio Express Edition DBMS

Microsoft SQL Server is a relational database management system (RDBMS) produced by Microsoft. Its primary query language is Transact-SQL, an implementation of the ANSI/ISO standard Structured Query Language (SQL) used by both Microsoft and Sybase.

SQL Server supports different data types, including primary types such as Integer, Float, Decimal, Char (including character strings), Varchar (variable length character strings), binary (for unstructured blobs of data), Text (for textual data) and Ntext (UNICODE) among others. It also allows user-defined composite types (UDTs) to be defined and used. SQL Server also makes server statistics available as virtual tables and views (called Dynamic Management Views or DMVs). A database can also contain other objects including views, stored procedures, indexes and constraints, in addition to tables, along with a transaction log. SQL Server is one of the very popular Microsoft products

A key reason for deciding to use SQL Server is the author's familiarity with SQL from previous experience. And also it is compatible with Microsoft product (i.e. Microsoft visual Studio 2005) [9].

### 4.4 Interface Design and Usability

A good interface design determines the popularity of the system and therefore becomes a competitive advantage in the field that it is designed for. For EventBook, one of the key requirements is to build a system that is efficient and is easy to use; therefore the interface design will serve as a vital ingredient in the success of this project.

Design rules in interface design are in the form of standards and guidelines that provide direction for design, regarding issues of presentation, behavior and interaction for interface elements and controls [26]. Although the use of standards and guidelines supports decisions in design choices but these need to be adjusted according to the

needs of the interface. Thus, all design choices, while designing the interface for EventBook will be made according to the standard and the task for which the system will be made for.

Further to the interface design, it is also essential to consider the usability of the system, which is considered a key factor in the design of any software application. According to ISO 9241, usability is defined as “the effectiveness, efficiency and satisfaction with which specified users achieve specified goals in particular environment” [31]. In simple words, it denotes how well the users can make use of the functionalities of the system. One of the most accepted measures of such usability comes in the form of Nielsen’s Usability Heuristics, which are given below:

- Simple and natural dialogue
- Speak the user’s language (Amharic Language)
- Minimize the user’s memory load
- Consistency
- Feedback
- Help and documentation
- Clearly marked exits
- Prevent errors
- Good error messages [7]

## 4.5 Others

In addition to the above listed tools, it also uses Adobe Photoshop for graphic background and button design, Macromedia Flash for animation, and XML.

## Chapter 5

### Prototype

The chapter begins by giving a brief overview of the web tier by illustrating the file structure and how it supports scalability and maintainability. It then moves on to provide a detailed explanation of the principles and techniques employed in the implementation of each component of MVC design pattern. Finally, database implementation along with statistics concerning the delivered system is outlined.

#### 5.1 Web Tier – Overview

Establishing an effective and maintainable file structure for EventBook was an activity that was finalized over several weeks of implementation. All files are structured in terms of the role they play in the MVC design pattern. Moreover, for each component of MVC the files are further split according to the type of user. This approach supports:

- **Scalability** as it allows both the view and model to be extended for a particular user without having an effect on any of these components for the other users. For instance, in future if there is a need to include further functionality for the user, then the relevant ASP(s) and related dot net class(es) will be added in the users folder of the ASP and classes directory.
- **Maintainability** as files for both model and view are separated for each type of user demonstrating the ease with which the system can be maintained.

##### 5.1.1 Configuration File (amharictyping.JS)

Before work could begin on the actual functionality, the first step was to build a Ge'ez font mapping to the existing QWERTY keyboard layout. File is created using Jscript "amharictyping.js" that is located in the EventBook folder of the application directory.

This file identifies and establishes the values for the global UNICODE that are to be used within the EventBook. The global parameters defined in this file are:

- `getCharacterCursorPosition()` : Specify the position of a cursor in a certain textbox of the page
- `isRootLetter()` : most of Ge'ez alphabets are a combinations of two or more keys
- `withinLastArray()` : specify the range of Unicode value assign to Ge'ez fonts
- `convertEnglishConsonantToAmharic()` : used to convert the English keyboard layout to equivalent Ge'ez fonts

The complete source code of AmharicTyping.js is attached herewith in Annex A

## 5.2 Web Tier – Controller Implementation

The controller was implemented by using the ASP technology. Its responsibilities include:

- Initializing the application.
- Creating session variable (Like user first name and Father Name) to identify the user in each page he/she navigates
- User authentication and authorization.
- Screen flow i.e. deciding which view to display next.

### 5.2.1 Initializing the application

On initialization, the method `get_username()` and `get_password()` specified in the login page is invoked from the database. The controller reads the values of the session parameters from the database and stores them in the session variable, so that this information could be made readily available to the entire application until the user logoff.

### 5.2.2 User authentication and authorization

On receiving a user name and password submitted by the user, the controller instantiates a `Login` that extends the `LoginDataHandler()`. The controller then invokes the method `AuthenticateUser()` of the `Login`, which in turn calls the `AuthenticateUser()` method of the super class i.e. `LoginDataHandler()`. If the user is valid, then this method returns a main menu object containing the user details such as full name, navigating button etc otherwise it returns an Invalid login message.

### 5.2.3 Screen flow

Controller defines the behavior of the application by deciding which `aspx` page to be display next. The process involves obtaining the value of the action parameter from the Http GET or POST request and retrieving the user type stored in the session object of the user. As such, the URLs used to move between different pages in EventBook are of the form:

<http://localhost/EventBook/login.aspx>

The values obtained are passed as parameters for instantiating the `EventHandler` class and then invoking the method `getEvent()`. This method returns a URL by calling the relevant method depending on the type of the user.

For instance, if the user type is Administrator, then the Administrator method will be invoked. The method Administrator will return the URL of the relevant page depending on the value of the action variable.

The implementation of the controller illustrates the ease with which it allows interface pages to be added or removed. Any addition of a new interface page simply involves the addition of a single else if statement in the relevant method depending on the type of the user.

### 5.3 Web Tier – Model Implementation

For EventBook, the model component is responsible for interacting with the data tier and supplying the business logic to both the view and the controller, whilst hiding these details from them. Thus, the approach used in implementation resulted in splitting of model classes into two layers as depicted on figure 5.1

- **Data Access layer** – abstract data handler classes that interact with the data tier
- **Business layer** – ASPX extend the abstract data handler classes and act as an intermediate between the other components of MVC and abstract data access classes.

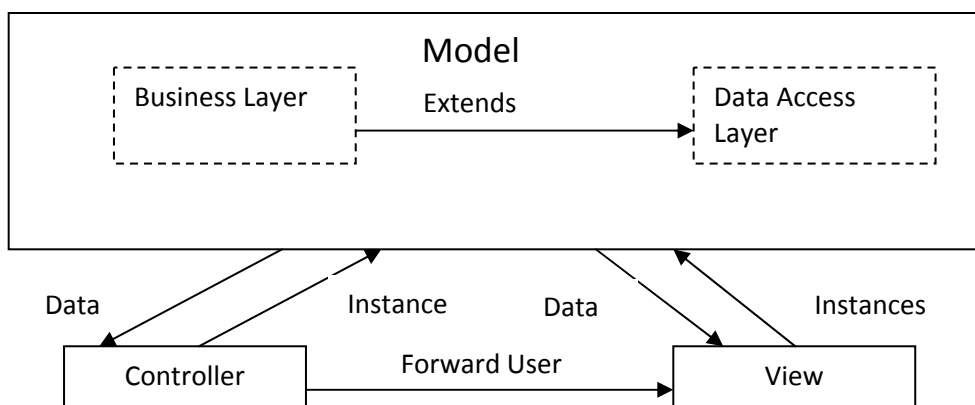


Figure 5.1: MVC design pattern within a three tier architecture of EventBook

### 5.3.1 Data Access Layer – DataHandler Classes

These are abstract Visual Studio.NET classes, which contain processes to govern access and modification of the backend data source. As such, all these classes extend the `DataAccessParameters` class that contains variables specifying the location, user name and password of the data source. This information is read by the controller on initialization from the database file and then stored in the session variable object.

Since, all data handler classes extend the `DataAccessParameters` class; a connection with the database is established as shown in Figure 5.1.

```
.CommandType = CommandType.StoredProcedure  
.CommandText = "dbo.pro_Chk_Account"  
.Parameters.Add("@UserID", SqlDbType.NVarChar, 50).Value = txtUserID.Text  
.Parameters.Add("@Password", SqlDbType.VarChar, 50).Value = txtPassword.Text  
rd = .ExecuteReader(CommandBehavior.CloseConnection)
```

*Figure 5.2: Establishing a connection with the database.*

The all methods are inherited from the `pro_Chk_Account` Stored procedure and these methods return the values for the location of the database, user name and password respectively as specified in the database file.

### 5.3.2 Connection State

To implement connection state, the initial approach involved creating a partial class called `Inherits System.Web.UI.Page`. This class maintained inherited object that uses `Connection` objects as keys and a `Boolean` object as stored values. This `Boolean` value indicates whether the connection is free or not. A program calls the `cnn.State = ConnectionState.Closed` method of this class to be assigned a connection. Once the connection is used, `ConnectionString ()` method is called that returns the connection back to the state as is illustrated in the Figure 5.3.

```
Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load
    With cnn
        If .State = ConnectionState.Closed Then
            .ConnectionString = "Data source =ISOFT\SQLEXPRESS; initial
catalog=EventBook; Integrated security =True"
            .Open()
        Else
            .Close()
            .ConnectionString = "Data source =ISOFT\SQLEXPRESS; initial
catalog=EventBook; Integrated security =True"
            .Open()
        End If
    End With
    txtUserID.Focus()
End Sub
```

*Fig 5.3 Connection to the database*

## 5.4 Web Tier – View Prototype

In EventBook, the Data Grid component is responsible for accessing data from the database and then indicating how the data should be presented. This component was mainly implemented by ASP technology to a small extent for changing the presentation ASP and input validation.

The following section gives a brief overview of the important aspects of the Grid view for User and system administrator.

### 5.4.1 User -Post

#### Login

As soon as the user types the appropriate EventBook URL, the official login page will be displayed. If the user have an already exist account, it can type his user name and password, if not, he/she can sign up to create a new account.

## Create account

If the user select the sign in control, the system display the create account forms, the user can now type the important components so as the system create account for the user and redirect to login page.

## Index page

Once a user logs in he/she is directed to main menu page (index.aspx), which displays four main major menus (Diary, Events, Phone Book and create website). Furthermore, the user can view the Amharic font keyboard layout located in the right hand corner.

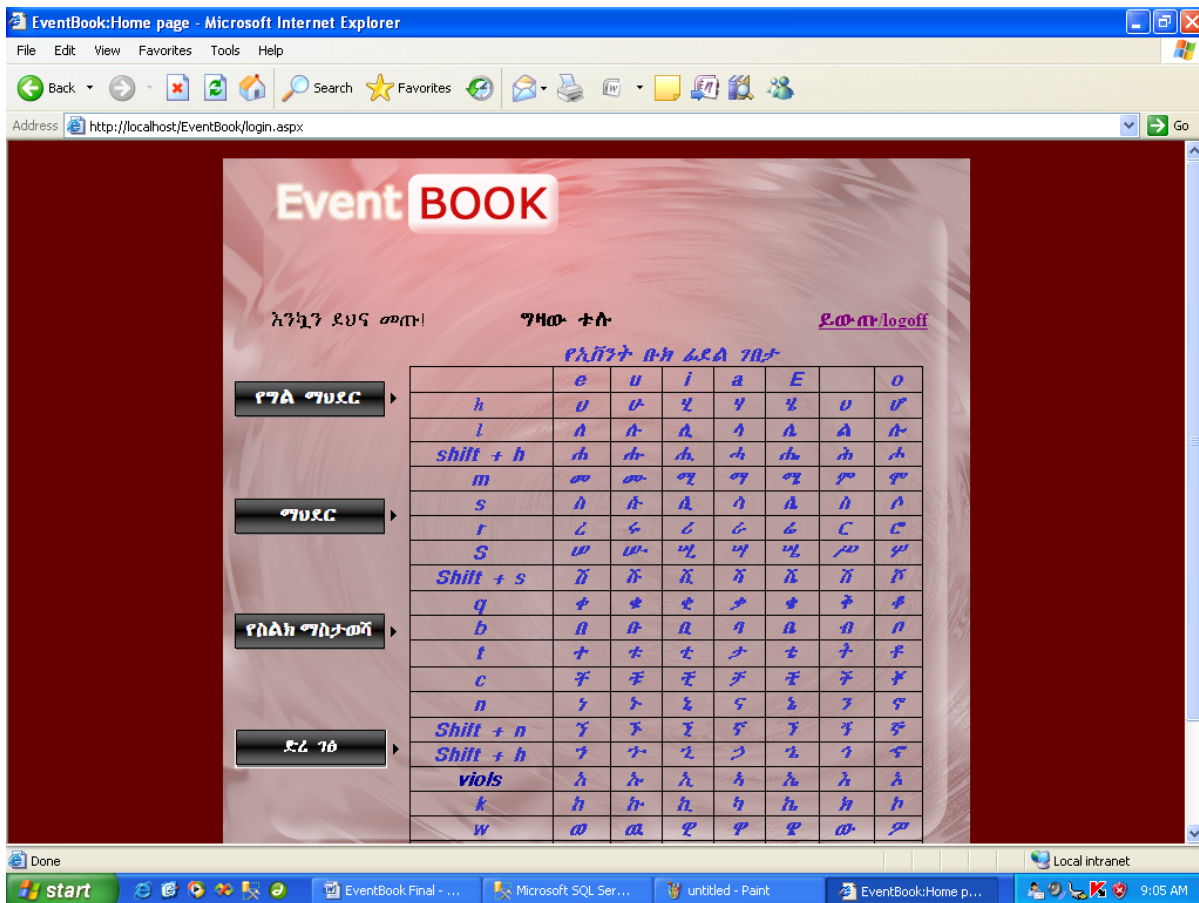


Figure 5.4: Screen shot of index.aspx page

## **Post Diary**

Out of the available menu the user can select the Diary menu, and can post, read, search, edit, and Delete his/her own Diary only.

## **Post Events**

Events are the same as Diary but can be visible to any user, a certain user can post, read, search, and edit

## **Manage Address book**

Ones the user can log into the system, he/she can manages the phone book, some of the functionality include, register a new address book, read, search, and edit.

## **Creating Personal Website**

The template used to produce the personal website is created, so that the user can generate the page by filling the form. One of the parameter of the template is to identify the URL. The URL finally used to access the website

### **5.4.2 User - View Records**

#### **Read Records**

On selection of the Read Diary, Events, and Phone book field from the menu, the user is directed to the "ReadDiary.aspx, ReadEvent.aspx, and ReadPhoneBook.aspx" respectively that displays all the posted information by user. The displayed Information can again be changed by simply clicking on selection button of the desired Diary or phonebook. Furthermore, if the user wishes to remove any Diary,

he/she can do so by checking the relevant Select options and then clicking on “Delete” button.

## Search Records

The search option in EventBook allows member of User to search the Event of a single or multiple user by selecting the concerned user(s).

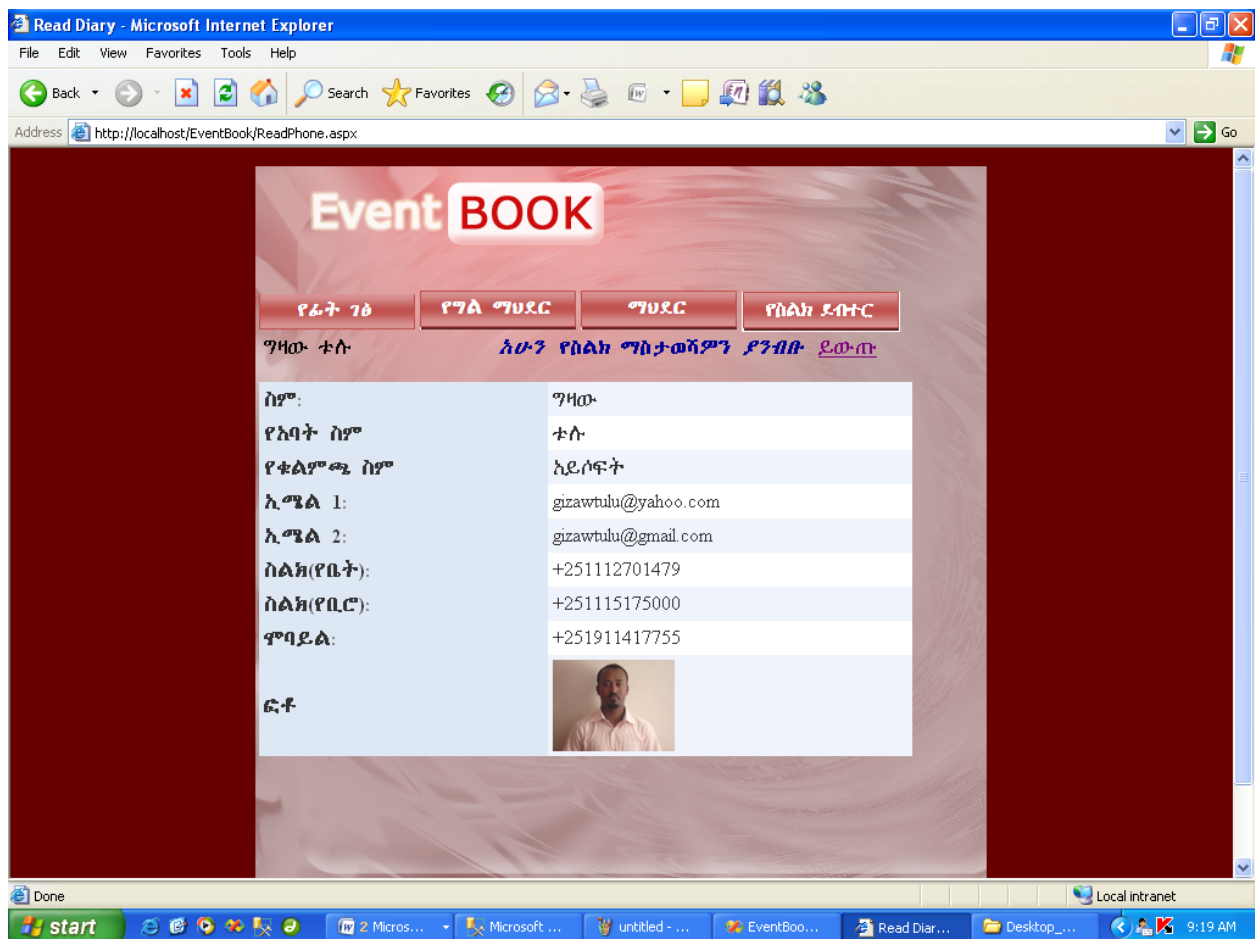


Figure 5.5: Screen shot of the SearchEvent.aspx

The search result returns posted date time, event title and event at which the entire selected users are available.

### 5.4.3 View – System Administrator

#### Manage User

The administrator can view all the users by selecting the “Delete Account or Delete Event” link from the menu. Users are displayed chronologically.

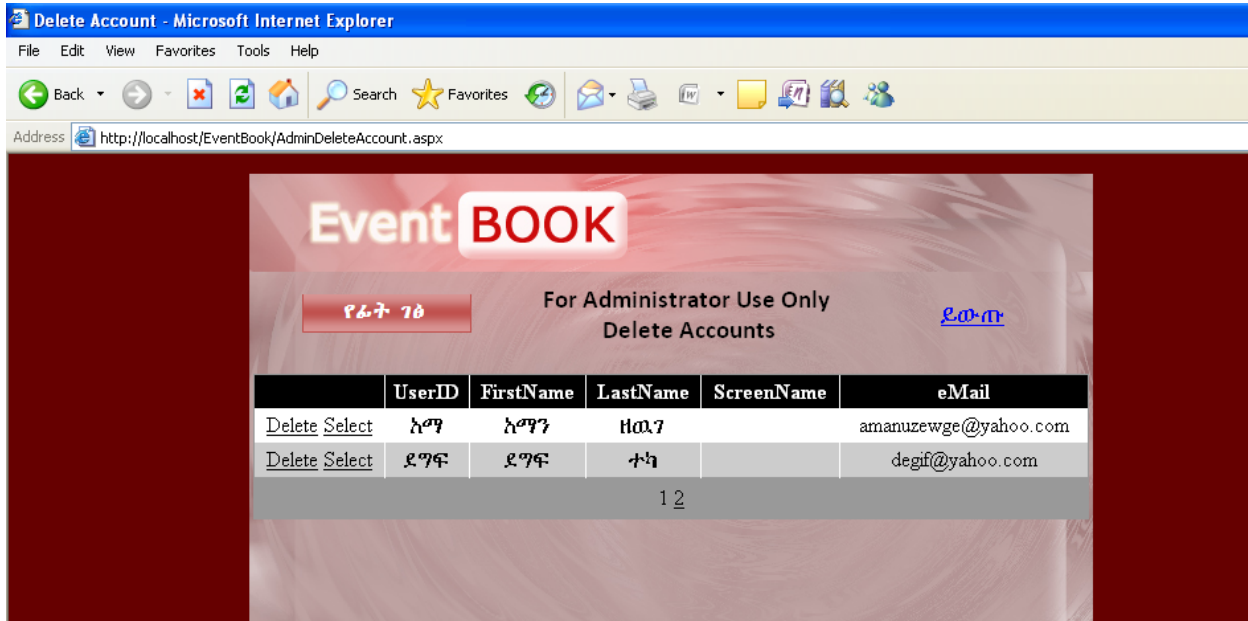


Figure 5.6: Screen shot of DeleteAccount.aspx

Administrator can delete an individual user by simply clicking on relevant link.

#### Delete Events

Like, the user account deletion, the system administrator can delete events posted by the user.

### 5.5 Data Tier – Prototype

The first task of implementation was to create the database tables. Initially, the data model developed during the design stage was implemented in Microsoft SQL server

2000 for the purpose of prototyping. However, this database was later replaced by a Microsoft SQL server 2005 management studio express edition. Since then, no design or implementation changes were made to the data tier, during the course of implementation.

## Chapter 6

# Testing

The testing process of EventBook was split into two major portions where:

- The first part of this process involves testing the compliance of the application against the functional requirements.
- The second part of the testing process is concerned with aspects such as performance and efficiency.

### 6.1 Application Functionality Testing

A major portion of the application functionality was tested whilst the system was being built. This involved the use of structural or white box testing on completion of each functional unit, by using the underlying knowledge of the code. The aim of this approach was to test boundary and decision conditions within the code. Furthermore, no formal documentation was produced for this part of the testing process.

On completion of EventBook, the black box testing method was applied to develop a set of test cases. Usually, for specification based testing methods some sort of formal specification is written. However, for EventBook no formal specification was provided because writing a formal specification would have meant spending less time on building the system.

For each functional unit categories, partitions and constraints were identified and then written in a test specification language. For instance, code for a user logging in can be seen in Figure 6.1

```

Imports System.Data
Imports System.Data.SqlClient
Partial Class login
    Inherits System.Web.UI.Page
    Dim cnn As New SqlConnection
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Me.Load
        With cnn
            If .State = ConnectionState.Closed Then
                .ConnectionString = "Data source =ISOFT\SQLEXPRESS; initial
catalog=EventBook; Integrated security =True"
                .Open()
            Else
                .Close()
                .ConnectionString = "Data source =ISOFT\SQLEXPRESS; initial
catalog=EventBook; Integrated security =True"
                .Open()
            End If
        End With
        txtUserID.Focus()
    End Sub
    Protected Sub btnLogIn_Click(ByVal sender As Object, ByVal e As
System.EventArgs) Handles btnLogIn.Click
        Dim sqlCmd As New SqlCommand
        Dim rd As SqlDataReader
        If txtUserID.Text <> "" And txtPassword.Text <> "" Then
            With sqlCmd
                .Connection = cnn
                If .Connection.State = ConnectionState.Closed Then
                    .Connection.Open()
                Else
                    .Connection.Close()
                    .Connection.Open()
                End If
                .CommandType = CommandType.StoredProcedure
                .CommandText = "dbo.pro_Chk_Account"
                .Parameters.Add("@UserID", SqlDbType.NVarChar, 50).Value =
txtUserID.Text
                .Parameters.Add("@Password", SqlDbType.VarChar, 50).Value =
txtPassword.Text
                rd = .ExecuteReader(CommandBehavior.CloseConnection)
            End With
            Do While rd.Read
                Session("UserName") = rd("FirstName") & " " & rd("LastName")
                Session("UserID") = rd("UserID")
                Session("FirstName") = rd("FirstName")
            Loop
            If Session("UserName") <> "" Then
                Server.Transfer("index.aspx")
            End If
            Else
                Session("UserName") = ""
                Response.Write("<font color='yellow'>" & Session("UserName") & "
Invalid Login!" & "</font>")
            End If
        End Sub
    End Class

```

Figure 6.1: Code listing of LoggingIn.aspx



Figure 6.2: LoggingIn.frm of EventBook

Each test frame contains set of values that need to appear in test cases. However, the generated values are not in a format that matches the system’s requirements and therefore need to be reformatted to produce actual sets of input values that are to be included in the test case. Examples of test cases produced from the Log in.frm are given in Table 6.1.

No	Inputs	Expected Output	Actual Output
1	username = null password = null	Error dialog displayed.	Error dialog displayed.
2	username = (invalid)	Error dialog displayed.	Error dialog displayed.

	password = null		
3	username = <b>᠙H᠘</b> (invalid) password = <b>᠙H᠘</b> (invalid)	Error dialog displayed.	Error dialog displayed.
4	username = <b>᠑H᠐</b> (valid) password = 1234 (invalid)	Error dialog displayed.	Error dialog displayed.
5	username = <b>᠙H᠘</b> (invalid) password = 1234 (valid)	Error dialog displayed.	Error dialog displayed.
6	username = <b>᠑H᠐</b> (valid) password = 123456 (valid)	User forwarded to index.aspx page.	User forwarded to index.aspx page.

*Table 6.1 testing the login page*

## 6.2 Performance Testing

This part of the testing process involved the use of a load testing application to simulate server usage at different loads. The aim was to ensure that EventBook will have a good response time during usage. Here, “response time” means “the time it takes between initial request and complete download of response i.e. rendering of an entire web page”.

It is possible to test the performance of EventBook using testing software like OpenSTA. Because of the license requirement to download from the internet, the developer of EventBook use a simple way of testing on different machine

All tests were carried out by using the following specification machine, with the EventBook website and load testing tool present on the same machine:

- **Processor:** 2 GHz
- **Memory:** 512 MB
- **Operating System:** Windows XP

## 6.3 Results and Discussions

Overall, all the testing conducted to test the functionality of the application led to a single conclusion that EventBook is compliant with the functional requirements detailed in the Requirements and Analysis stage. It was also felt that EventBook would perform substantially better as compared to its performance during the testing process mainly because of the following reason:

- The machine on which EventBook was hosted during testing cannot achieve the same performance level as a high-end web server because the server would have a faster processor and a lot more memory. Furthermore, during performance testing the load testing application was also being run on the same machine as EventBook, which means an increased burden on the machine resources as compared to normal circumstances.

## Chapter 7

# Conclusions and Future Works

### 7.1 Conclusions

Many valuable lessons were learnt during the course of this project on both technical and human level. Perhaps the most important of them being related to the extent of planning required for an implementation based project. The initial technology survey along with the requirements analysis provided valuable background knowledge for the implementation stage. Still setting up a working web application architecture where all the technologies interact with each other is a sizeable task in itself that was seriously miscalculated during the course of this project. Moreover, it was felt that if the initial plan had provided a deeper analysis of the steps involved in each of the tasks, it would have allowed better appreciation of the time involved in each stage of the build.

The evaluation of the model two architecture leads to the conclusion that it is very well suited for this purpose as it allows separation of logics between the different components and supports scalability. However at the same time the coupling between the model and the data tier was also highlighted. Any change in the data tier dramatically affects the model component. Therefore, in order to minimize this dependency in EventBook the traditional MVC design pattern was slightly modified as to split the model component in two layers namely the business and data access layers. In the modified version, the sole responsibility of the data access layer is to interact with the data tier, whereas the business layer not only acts as an intermediate between the data access layer and other components of MVC design pattern but it also provides the business logic required by the view and the controller.

Success can be measured on many different levels but for an implementation based project the primary measure of success can be deemed as the system's compliance with functionality required by the client. In terms of functionality it is clear that EventBook satisfies both its primary and secondary aim by providing the facility to Amharic Language users of the Internet.

## 7.2 Future Works

The future work of EventBook is to include other Amharic language applications that integrated with EventBook like

- Semantic searching of events and diary
- Amharic discussion forum
- The template for personal webpage will be designed according to user's preference and categories.
- Creating a big portal system

## Reference

1. Glenn Johnson and Tony Northrup, *Microsoft .NET FRAMEWORK 2.0 Web-Based Client Development*, Microsoft Press, 2007
2. Bakharia, A., *IIS & easy web development*, Prima Tech, 2004
3. Microsoft, *Windows XP help and support*, Microsoft Express, 2001
4. Date, C. J., *An Introduction to Database Systems*, Seventh Edition, Addison Wesley, 2000
5. Elmasri & Navathe, *Fundamentals of Database Systems*, Third Edition, Addison Wesley, 2004
6. Connoly, T., Begg, C., Strachan, A., *Database Systems: A Practical Approach to Design, Implementation and Management*, Second Addition, Addison Wesley, 2002
7. Baer, A., Saroiu, S., Koutsky, L.A., *Obtaining Sensitive Data Through the Web: An Example of Design and Methods*, 2002
8. Kline , K. E., *SQL in a nutshell: A Desktop Quick Reference*, O'Reilly, 2001
9. Kaurniawan, B., *Visual Studio 2005 for the Web with ASP*, Microsoft Express, 2007
10. Alexander, C., Ishikawa, S., Silverstein, M., *A Pattern Language: Towns, Buildings and Construction*, Oxford University Press, 1977
11. Lane, D., Williams, H.E., *Web Database Applications with ASP and SQL*, O'Reily, 2002
12. Eaglestone, B., Ridley, M., *Web Database Systems*, McGraw-Hill, 2001
13. Simon, B., McRobb, S., Farmer, R., *Object Oriented Systems Analysis and Design using UML*, McGraw-Hill Web References, 1999
14. <http://www.opendiary.com> last accessed on March 26, 2008

15. <http://www.blog.com> last accessed on March 26, 2008
16. <http://www.timeanddate.com> last accessed on March 26, 2008
17. <http://www.javaworld.com/javaworld/jw-01-2000/jw-01-ssj-tiers.html> last accessed on April 15, 2008
18. <http://www.oed.com>, last accessed on April 15, 2008
19. <http://chainlink.unifiedconsulting.com/cycle.php>, last accessed on April 15, 2008
20. <http://www.woodger.ca/archweb.htm>, last accessed on April 15, 2008
21. <http://www.csc.vill.edu/~rbalasub/netclass/disarti1.doc>, last accessed on April 15, 2008
22. <http://www.inf.ethz.ch/personal/iks/Other/PDDBS/pdf/2Vers3Tiers.pdf> ,last accessed on April 15, 2008
23. [http://www.cs.ucf.edu/~fhamza/papers/VirtualUniversity\\_Hamza-Lup.pdf](http://www.cs.ucf.edu/~fhamza/papers/VirtualUniversity_Hamza-Lup.pdf) , last accessed on April 15, 2008
24. <http://www.midrangeserver.com/mpo/mpo120502-story04.html> ,last accessed on April 15, 2008
25. [http://java.sun.com/blueprints/guidelines/designing\\_enterprise\\_applications\\_](http://java.sun.com/blueprints/guidelines/designing_enterprise_applications_) last accessed on April 15, 2008
26. <http://www.dcs.shef.ac.uk/~martin/teaching/ecommerce>, last accessed on April 15, 2008
27. [www.mgtaylor.com/mgtaylor/jotm/winter97/patintro.htm](http://www.mgtaylor.com/mgtaylor/jotm/winter97/patintro.htm), last accessed on April 15, 2008
28. <http://en.wikipedia.com/WEFT> last accessed on March 26, 2008

29. <http://www.computeruser.com/resources/dictionary/index.html> last accessed on March 26, 2008
30. <http://www.SQLSERVER.Microsoft.com/> last accessed on March 26, 2008
31. <http://www.nviron.co.uk/solutions/nable/technical/qualities.htm> last accessed on March 26, 2008

## Glossary

### **ASP**

Active Server Pages (ASP) is a Windows based server-side scripting language that combines static HTML with dynamic content.

### **Black Box Testing**

Testing a system without knowing its internal workings

### **EventBook**

EventBook is a web based system that store users diary, phone book, events... and retrieve accordingly.

### **HTTP:**

HTTP is a protocol used to transfer information on the World Wide Web.

### **Model-View-Controller (MVC)**

MVC is a design pattern used to separate the different parts of an application into the three components called Model, View and Controller.

### **SQL Server**

A Relational Database Management System (RDMS) provided by Microsoft

### **SQL (Structured Query Language)**

A unified language for defining, querying, modifying and controlling the data in a relational database

### **Three-tier Architecture**

Architecture design pattern that is used for web applications – the three tiers are the Client, Web and the Data tier.

### **RDMS (Relational Database Management System)**

A form of managing a database that involves storing data in the form of related tables

### **Microsoft Visual Studio**

Microsoft Visual Studio is Microsoft's flagship software development product for computer programmers

## Annex A: AmharicTyping.JS

```
// Copyright: (c) 2008 by EventBook.com
// Product title: Amharic Phonetic Typing Javascript
// Product URL: http://www.EventBok.com/js/
// Contact info: gizawtulu@yahoo.co.com

function getCharacterCursorPosition(){
    var last;
    last= document.selection.createLast();
    last.moveStart("Textedit", -1);
    return last.text.length;
};

function convertEnglishConsonantTounicodeFont(lastChar, KeyboardValue,
shiftKey)
    {
    var UnicodeKeyboardValue;
    var lastcharConsonant = lastChar;

    if( KeyboardValue == "q" )
    {
        UnicodeKeyboardValue = 4677;
    }else if ( KeyboardValue == "h" && shiftKey == true)
    {
        UnicodeKeyboardValue = 4629;

    } else if ( KeyboardValue == "k" && shiftKey == true)
    {
        UnicodeKeyboardValue = 4797;
    }else if ( KeyboardValue == "w" && shiftKey == false)
    {
        UnicodeKeyboardValue = 4813;
    } else if ( KeyboardValue == "r")
    {
        UnicodeKeyboardValue = 4653;
    } else if ( KeyboardValue == "y")
    {
        UnicodeKeyboardValue = 4845;
    }else if ( KeyboardValue == "t" && shiftKey == true)
    {
        UnicodeKeyboardValue = 4901;
    }else if ( KeyboardValue == "t" && shiftKey == false)
    {
        UnicodeKeyboardValue = 4725;
    }else if ( KeyboardValue == "p" && shiftKey == false)
    {
        UnicodeKeyboardValue = 4949;
    } else if ( KeyboardValue == "s" && shiftKey == false)
```

```
{
    UnicodeKeyboardValue = 4661;
} else if ( KeyboardValue == "d")
{
    UnicodeKeyboardValue = 4853;
} else if ( KeyboardValue == "f")
{
    UnicodeKeyboardValue = 4941;
} else if ( KeyboardValue == "g")
{
    UnicodeKeyboardValue = 4877;
} else if ( KeyboardValue == "h" && shiftKey == false)
{
    UnicodeKeyboardValue = 4613;
} else if ( KeyboardValue == "j")
{
    UnicodeKeyboardValue = 4869;
} else if ( KeyboardValue == "k" && shiftKey == false)
{
    UnicodeKeyboardValue = 4781;
} else if ( KeyboardValue == "l")
{
    UnicodeKeyboardValue = 4621;
} else if ( KeyboardValue == "z" && shiftKey == false)
{
    UnicodeKeyboardValue = 4829;
} else if ( KeyboardValue == "z" && shiftKey == true)
{
    UnicodeKeyboardValue = 4837;
} else if ( KeyboardValue == "x" && shiftKey == false)
{
    UnicodeKeyboardValue = 4933;
} else if ( KeyboardValue == "x" && shiftKey == true)
{
    UnicodeKeyboardValue = 4925;
} else if ( KeyboardValue == "c" && shiftKey == false)
{
    UnicodeKeyboardValue = 4733;
} else if ( KeyboardValue == "c" && shiftKey == true)
{
    UnicodeKeyboardValue = 4909;
} else if ( KeyboardValue == "v")
{
    UnicodeKeyboardValue = 4717;
} else if ( KeyboardValue == "b")
{
    UnicodeKeyboardValue = 4709;
} else if ( KeyboardValue == "n" && shiftKey == true)
{
    UnicodeKeyboardValue = 4765;
} else if ( KeyboardValue == "n" && shiftKey == false)
```

```

    {
        UnicodeKeyboardValue = 4757;
    }else if ( KeyboardValue == "m")
    {
        //unicodeFont.value = unicodeFont.value
+ "?" ;
        UnicodeKeyboardValue = 4637;
    }else if (KeyboardValue == "p" && shiftKey == true) //[
    {
        UnicodeKeyboardValue = 4917;
    } else if (KeyboardValue == "s" && shiftKey == true)
    {
        UnicodeKeyboardValue = 4669;

    } else if( KeyboardValue == ",")
    {
        UnicodeKeyboardValue = 4963;
    }else if( KeyboardValue == "-")
    {
        UnicodeKeyboardValue = 4959;
    }

    if ( UnicodeKeyboardValue )
    {
        return String.fromCharCode(UnicodeKeyboardValue);
    } else
    {
        return "";
    }
};

```

```

function isRootLetter(KeyboardValue)
{
    var isRoot = false;

    switch (KeyboardValue)
    {
        case 4613:
        case 4621:
        case 4629:
        case 4637:
        case 4637:
        case 4645:
        case 4653:
        case 4661:
        case 4669:
        case 4677:
        case 4709:
        case 4717:
        case 4725:
        case 4733:

```

```
    case 4741:
    case 4757:
    case 4765:
    case 4781:
    case 4797:
    case 4813:
    case 4821:
    case 4829:
    case 4837:
    case 4845:
    case 4853:
    case 4869:
    case 4877:
    case 4901:
    case 4909:
    case 4917:
    case 4925:
    case 4933:
    case 4941:
    case 4949:
    case 4959:

        {
            isRoot = true;
            break;
        }
    }
    return isRoot;
};

function withinLastArray(UniValue)
{
    if ( UniValue >= 4608 && UniValue <= 4988)
    {
        return true;
    } else{
        return false;
    }
};

function handleFontMapping(KeyEvent, unicodeFont){
    var keyCode = 0
    var isNetscape = false;
    var range;
    var newPos;
    var startPos = 0;
    var endPos = 0;

    if(window.event) {
        // for IE, e.keyCode or window.event.keyCode can be used
        keyCode = KeyEvent.keyCode;
```

```

        range = document.selection.createRange();
        newPos = getCursorPosition();
        startPos = getCursorPosition();
        endPos = getCursorPosition();
    }
    else if(KeyEvent.which)
    {
        // netscape
        keyCode = KeyEvent.which;
        isNetscape = true;
    }
    var shiftKey = false;
    var controlKey = false;
    if (window.event)
    { shiftKey = window.event.shiftKey; controlKey=window.event.ctrlKey;}
    else if (KeyEvent.which)
    { shiftKey = KeyEvent.shiftKey; controlKey=KeyEvent.ctrlKey;}
    if( controlKey) return true;

    var keyString = String.fromCharCode(keyCode).toLowerCase();
    var rawString = String.fromCharCode(keyCode);

    // let unicodeFont entry pass.
    if ( keyCode > 4000 || (keyCode >= 33 && keyCode <= 57 && keyCode
!= 44) || (keyCode >= 91 && keyCode <= 93) || (keyCode >= 60 &&
keyCode <= 64) || (keyCode >= 123 && keyCode <= 125))
    {
        return true;
    }

    if (unicodeFont.selectionStart || unicodeFont.selectionStart ==
'0')
    {
        startPos = unicodeFont.selectionStart;
        endPos = unicodeFont.selectionEnd;
        newPos = startPos;
    }

    var lastCharPlain = unicodeFont.value.charCodeAt(startPos-1);

    if (keyCode != 16 && keyCode != 8 && keyCode != 13 && keyCode !=
0 && keyCode != 32)//&& keyCode != 116) //shift)
    {
        var UnicodeKeyboardValue = "";
        var KeyboardValue = keyString;
        var tempExtraLetter =
unicodeFont.value.substring(0,unicodeFont.value.length - 1);

```

```

        var consonantRetVal =
convertEnglishConsonantTounicodeFont(lastCharPlain, KeyboardValue,
shiftKey);

        if (consonantRetVal != null && consonantRetVal.length > 0)
        {
            var tempStartPos = startPos;
            if( KeyboardValue == "h" && shiftKey == false)
            {
                var hconsValue = 0;
                if ( lastCharPlain == 4629)
                {hconsValue = 4741;}
                else if (lastCharPlain == 4613)
                {hconsValue = 4629;} else
                {hconsValue = 4613; newPos = newPos + 1;
tempStartPos = startPos + 1;}

                unicodeFont.value =
unicodeFont.value.substring(0, tempStartPos-1)
                    + String.fromCharCode(hconsValue )
                    + unicodeFont.value.substring(startPos,
unicodeFont.value.length);
            }else if ( KeyboardValue == "s" && shiftKey == false)
            {
                var tempStartPos = startPos;
                var hconsValue = 0;
                if ( lastCharPlain == 4661){
                    hconsValue = 4645;
                } else {hconsValue = 4661; tempStartPos =
startPos + 1; newPos = newPos + 1}
                unicodeFont.value =
unicodeFont.value.substring(0, tempStartPos-1)
                    + String.fromCharCode(hconsValue )
                    + unicodeFont.value.substring(startPos,
unicodeFont.value.length);
            } else if (KeyboardValue == ",")
            {
                var tempStartPos = startPos;
                var hconsValue = 0;
                if ( lastCharPlain == 4963){
                    hconsValue = 44;
                } else {hconsValue = 4963; tempStartPos =
startPos + 1; newPos = newPos + 1}
                unicodeFont.value =
unicodeFont.value.substring(0, tempStartPos-1)
                    + String.fromCharCode(hconsValue )
                    + unicodeFont.value.substring(startPos,
unicodeFont.value.length);

            }else
            {

```

```

        unicodeFont.value =
unicodeFont.value.substring(0, startPos)
                        + consonantRetVal
                        + unicodeFont.value.substring(endPos,
unicodeFont.value.length);

        newPos = newPos + 1;
    }
}

var aOffset = -2;
var eOffset = -5;
var iOffset = -3;
var uOffset = -4;
var oOffset = 1;
var eeOffset = -1;
var offset = 0;
var vowelCharacter ;
var otherVowelCharacter = 0;
if ( (KeyboardValue == "a" && shiftKey == false) ||
KeyboardValue == "4")
{
    offset = aOffset;
    vowelCharacter = 4768;
    if ( lastCharPlain == 4768) { otherVowelCharacter =
4816;}
} else if ( (KeyboardValue == "a" && shiftKey == true) )
{
    offset = aOffset;
    vowelCharacter = 4771;
    if ( lastCharPlain == 4771 ) { otherVowelCharacter =
4819;}
}
else if ( (KeyboardValue == "e" && shiftKey == false) ||
KeyboardValue == "1")
{
    offset = eOffset;
    vowelCharacter = 4773;
    if ( lastCharPlain == 4773 ) { otherVowelCharacter =
4821;}
}
else if ( (KeyboardValue == "i" ) || KeyboardValue == "3")
{
    offset = iOffset;
    vowelCharacter = 4770;
    if ( lastCharPlain == 4770 ) { otherVowelCharacter =
4818;}
}
else if (( KeyboardValue == "o" ) || KeyboardValue == "7")
{
    offset = oOffset;
    vowelCharacter = 4774;

```

```

        if ( lastCharPlain == 4774 ) { otherVowelCharacter =
4822;}
    }else if ( (KeyboardValue == "u") || KeyboardValue == "2")
    {
        offset = uOffset;
        vowelCharacter = 4769;
        if ( lastCharPlain == 4769 ) { otherVowelCharacter =
4817;}
    }else if ( (KeyboardValue == "e" && shiftKey ==true) ||
KeyboardValue == "5")
    {
        offset = eeOffset;
        vowelCharacter = 4772;
        if ( lastCharPlain == 4772 ) { otherVowelCharacter =
4820;}
    } else if( KeyboardValue == ";" )
    {
        unicodeFont.value = unicodeFont.value +
String.fromCharCode(4964);
        newPos = newPos + 1;

    } else if( KeyboardValue == ":" )
    {
        unicodeFont.value = unicodeFont.value +
String.fromCharCode(4961);
        newPos = newPos + 1;

    }

    var vowelCharacterChar =
String.fromCharCode(vowelCharacter);
    //var cursorIndex = unicodeFont.value.indexOf("|");
    //cursor.text = "" + cursorIndex;

    var lastCharEtymology =
unicodeFont.value.charCodeAt(startPos-1);

    if ( offset != 0 )
    { //?

        if ( unicodeFont.value.length < 1 )
        {
            unicodeFont.value = unicodeFont.value +
vowelCharacterChar ;
                newPos = newPos + 1;

            //KeyEvent.keyCode = 4773;
        }else if(unicodeFont.value.charCodeAt(startPos-1) ==
32 )
        {

```

```

        unicodeFont.value = unicodeFont.value +
vowelCharacterChar ;
                newPos = newPos + 1;
    }
    else if (lastCharEtymology == 4883 ||
lastCharEtymology == 4683 || lastCharEtymology == 4803 ||
lastCharEtymology == 4787 || lastCharEtymology == 4747)
    {
        if ( KeyboardValue == "e" && shiftKey == false)
lastCharEtymology = lastCharEtymology - 3;
        if ( KeyboardValue == "i") lastCharEtymology =
lastCharEtymology - 1;
        //if ( KeyboardValue == "a") lastCharEtymology =
lastCharEtymology ;
        if ( KeyboardValue == "e" && shiftKey == true)
lastCharEtymology = lastCharEtymology + 1;
        if ( KeyboardValue == "u") lastCharEtymology =
lastCharEtymology + 2;

                unicodeFont.value =
unicodeFont.value.substring(0, startPos-1)
                    +
String.fromCharCode(lastCharEtymology )
                    +
unicodeFont.value.substring(startPos, unicodeFont.value.length);

    } else
    {
        var lastchar =
unicodeFont.value.charCodeAt(startPos-1) + offset;

        if ( otherVowelCharacter > 0)
        {
            unicodeFont.value =
unicodeFont.value.substring(0, startPos-1)
                +
String.fromCharCode(otherVowelCharacter)
                +
unicodeFont.value.substring(startPos, unicodeFont.value.length);

        }
        else if ( isRootLetter(lastCharPlain) )
        {
            unicodeFont.value =
unicodeFont.value.substring(0, startPos-1)
                + String.fromCharCode(lastchar)
                +
unicodeFont.value.substring(startPos, unicodeFont.value.length);
        } else

```

```

        {
            unicodeFont.value =
unicodeFont.value.substring(0, startPos)
                + vowelCharacterChar
                +
unicodeFont.value.substring(startPos, unicodeFont.value.length);

                newPos = newPos + 1;

        }
    }

    //Output.value = unicodeFont.value.charCodeAt(startPos-1);
    if ( KeyboardValue == "w" && shiftKey == true
) //KeyboardValue == "/" ) // keycode == 191
    { //4775
        var lastchar2 = unicodeFont.value.charCodeAt(startPos-
1);

        if (unicodeFont.value.length < 1)
        {
            lastCharModified = 4775;
            unicodeFont.value = String.fromCharCode(4775)
            newPos = newPos + 1;
        }

        if( ! (lastchar2 == 4845 || lastchar2 == 4813 ||
lastchar2 == 4933 || lastchar2 == 4768 || lastchar2 == 4821) )
        {

            var lastCharModified;
            if (lastchar2 == 4677 || lastchar2 == 4877 ||
lastchar2 == 4781 )
            {
                lastCharModified = lastchar2 + 6;
            } else if (lastchar2 == 4613)
            {
                lastCharModified = 4747;
            } else if (lastchar2 == 4797)
            {
                lastCharModified = 4803;
            }
            else if ( lastchar2 == 4741)
            {
                lastCharModified = 4747;
            } else
            {
                lastCharModified = lastchar2 +2;
            }
        }
    }

```

```
        if (isRootLetter(lastchar2))
        {
            //Output.value = lastchar2;
            unicodeFont.value =
unicodeFont.value.substring(0, startPos-1)
                +
String.fromCharCode(lastCharModified)
                +
unicodeFont.value.substring(startPos, unicodeFont.value.length);

                //unicodeFont.value =
unicodeFont.value.substring(0, unicodeFont.value.charCodeAt(startPos-
1)) + String.fromCharCode(lastchar)
        }
    }

    if ( isNetscape )
    {
        unicodeFont.setSelectionRange(newPos, newPos);
    } else
    {
        range.collapse();
        range.moveStart("Character", newPos);
        range.select();
    }

    return false;
}

return true;
};

function withinRange(UniValue)
{
    if ( UniValue >= 4608 && UniValue <= 4951)
    {
        return true;
    } else{
        return false;
    }
};
```

## Annex B: Sequence Diagram

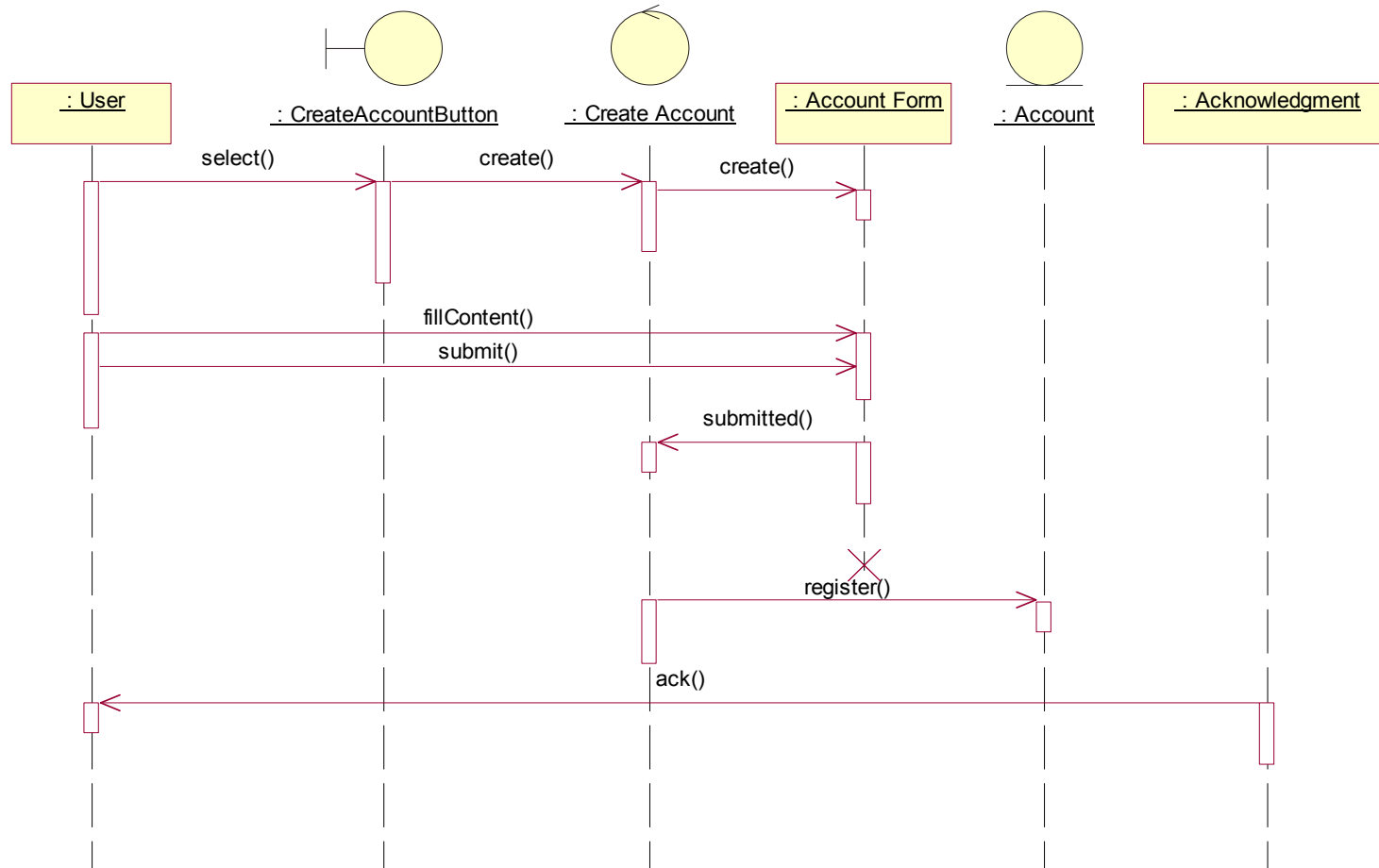


FIG 4.5: Sequence Diagram for Create Account Use Case

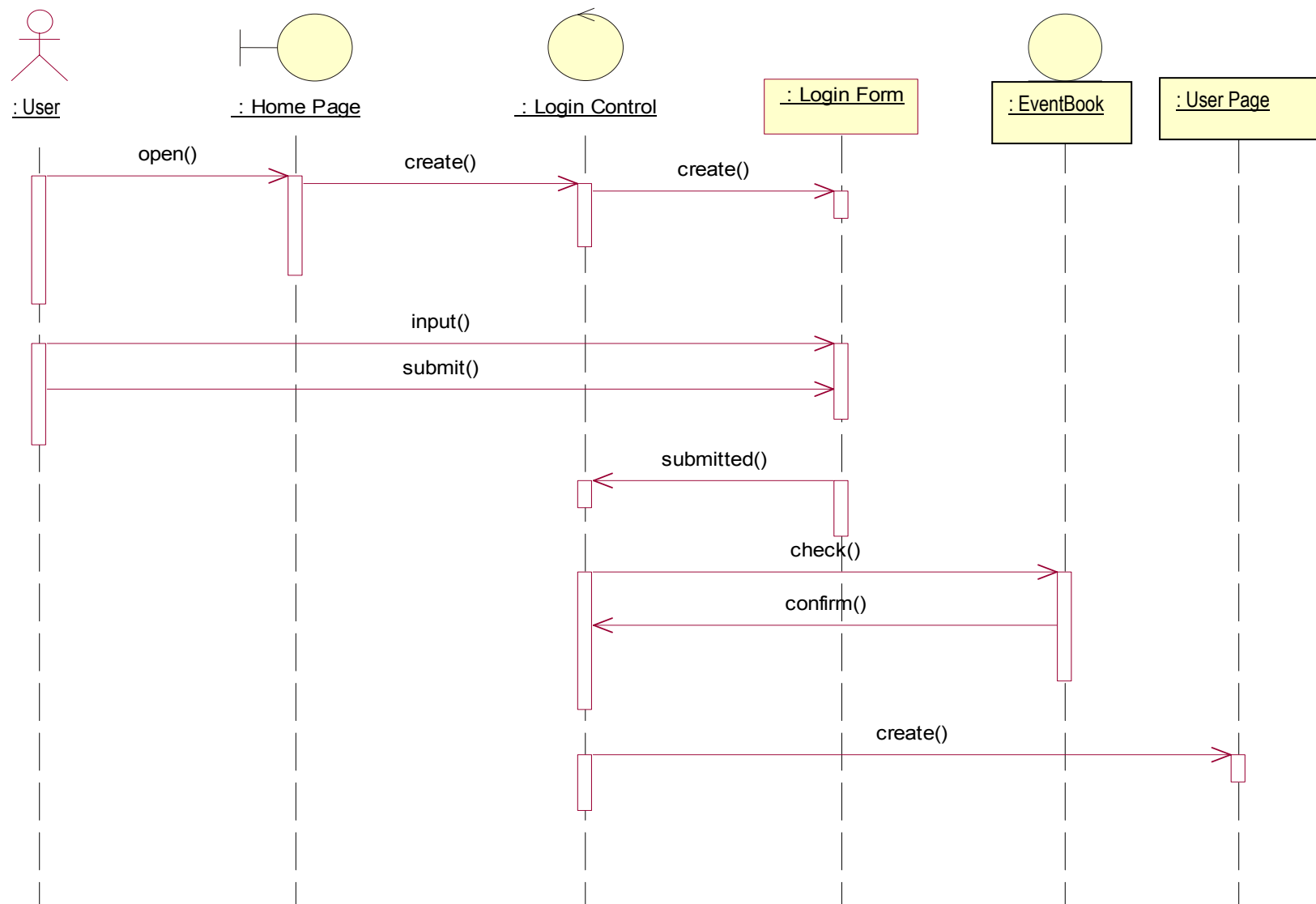


FIG 4.8: Sequence Diagram for Login Use Case

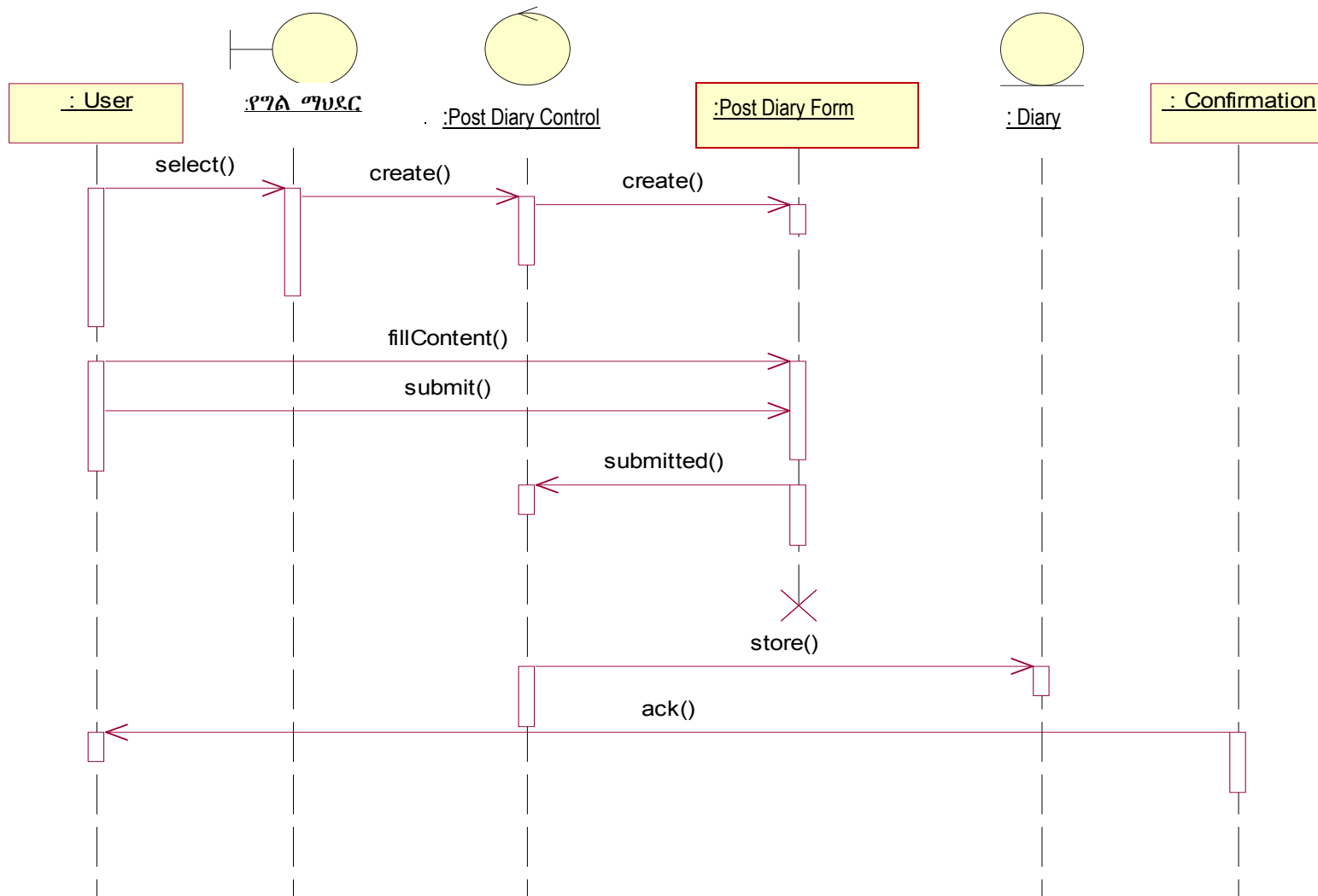


FIG 4.10: Sequence Diagram for Post Diary Use Case

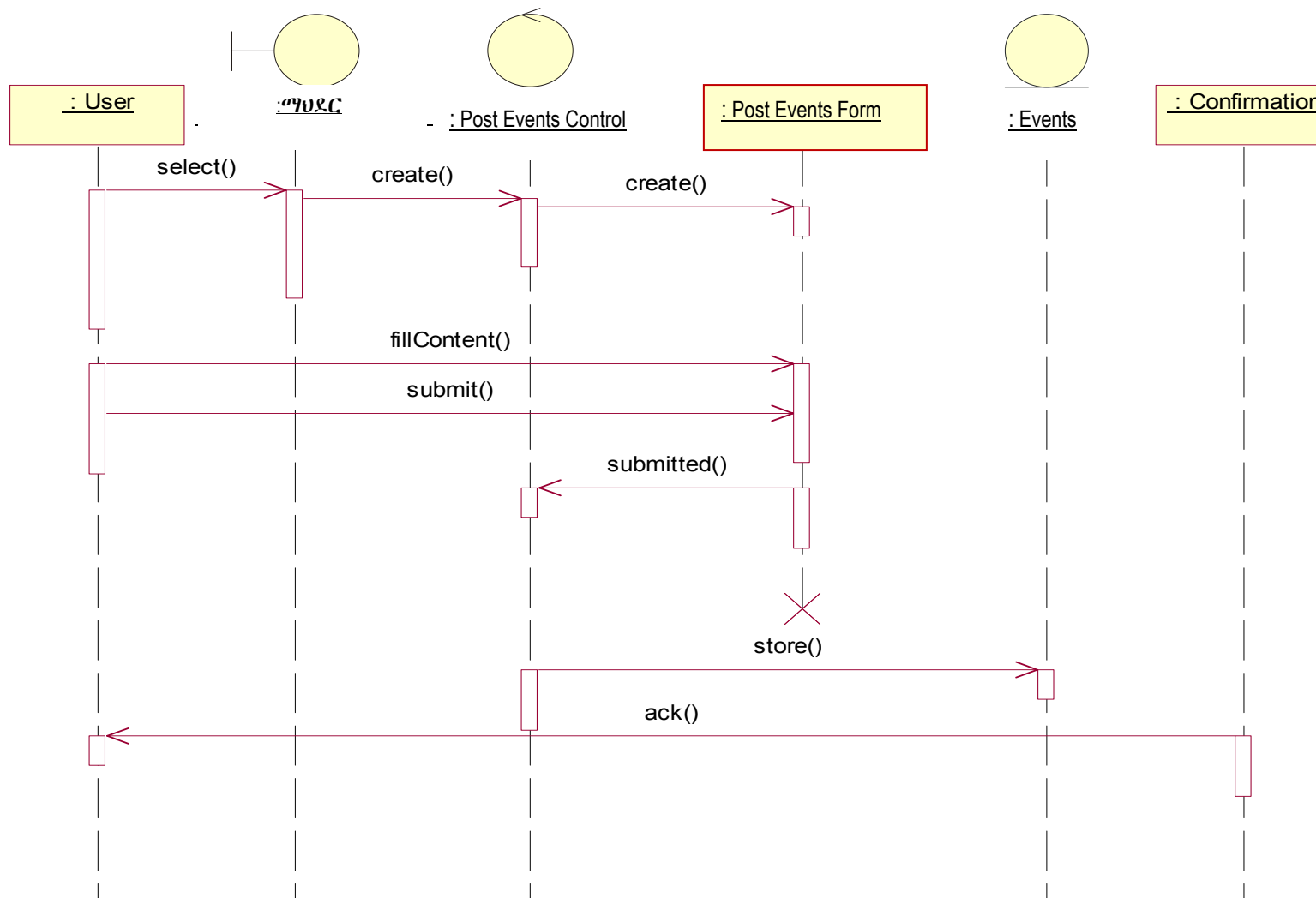


FIG 4.12: Sequence Diagram for Post Events Use Case

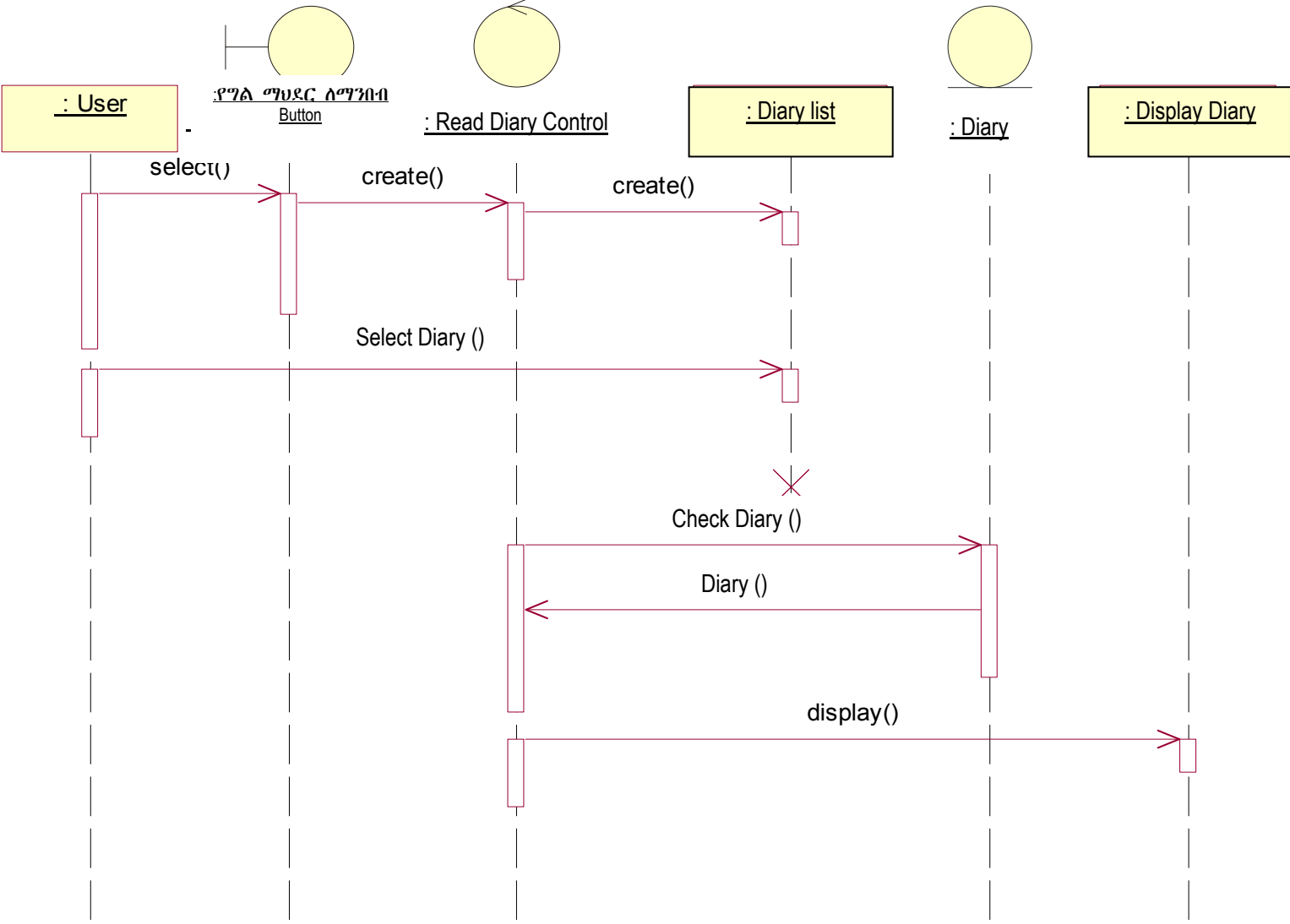


FIG 4.14 Sequence Diagram for Read Diary Use Case

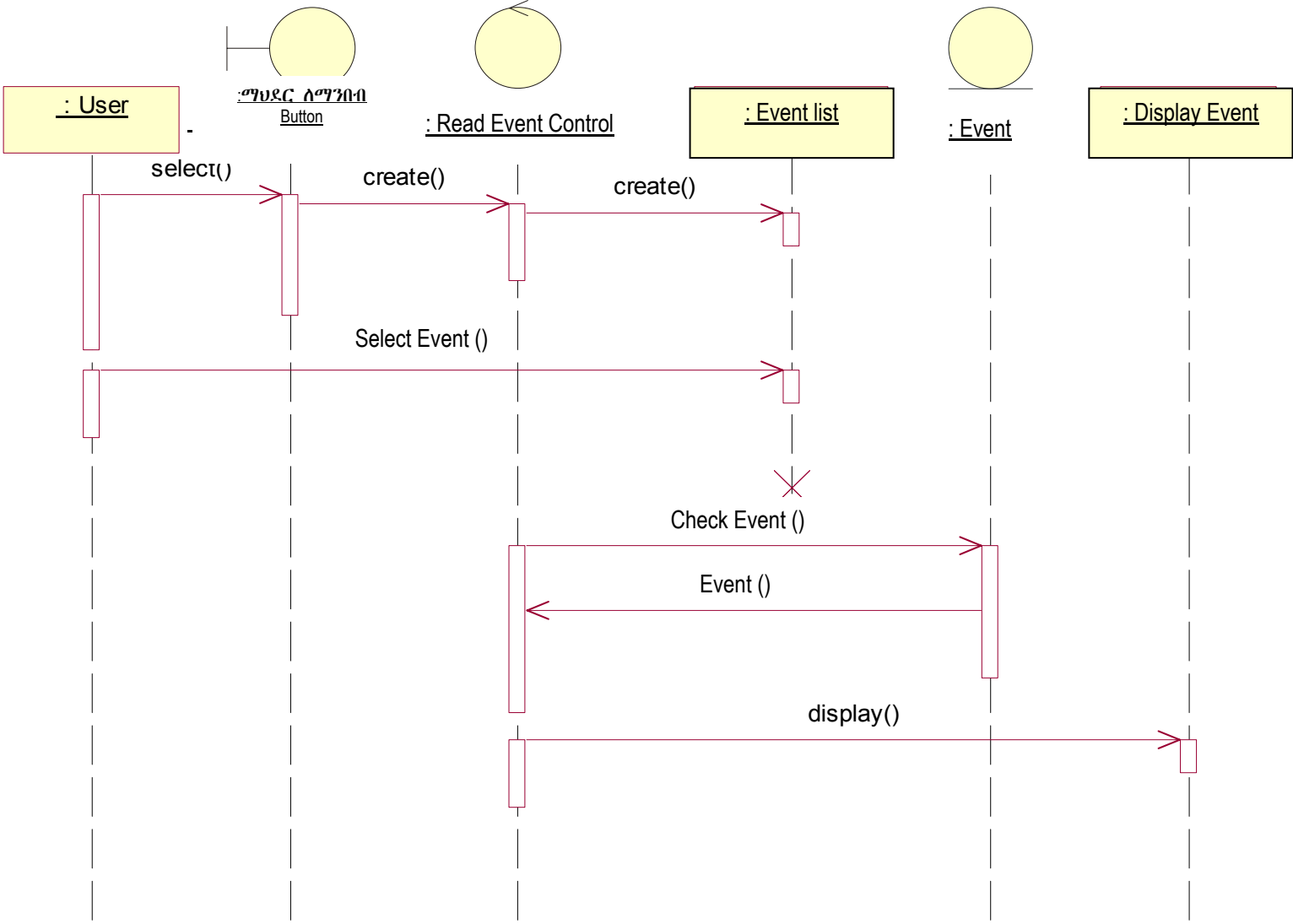


FIG 4.16: Sequence Diagram for Read Event Use Case

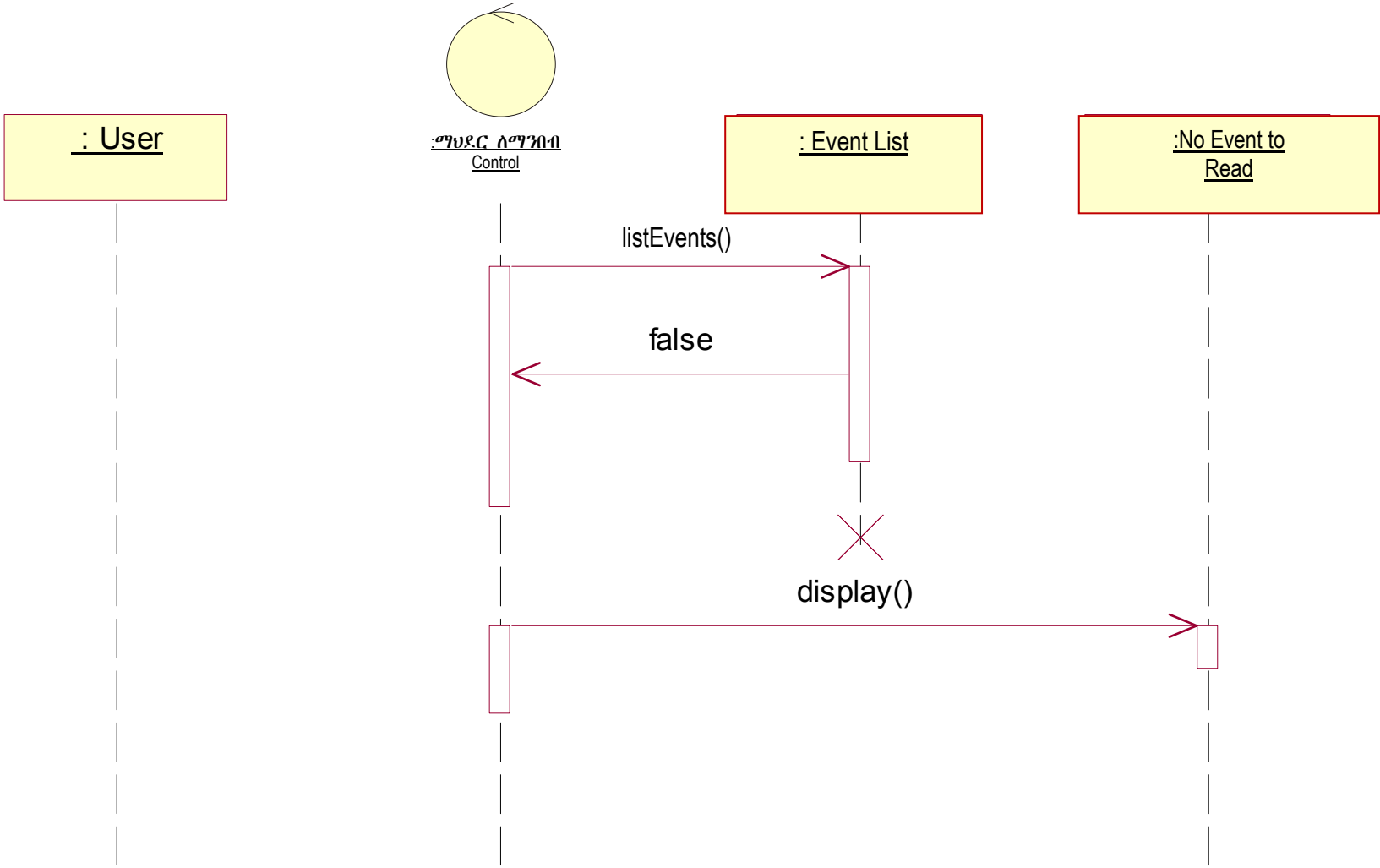


FIG 4.17: Sequence Diagram for Read Events Alternate Course A

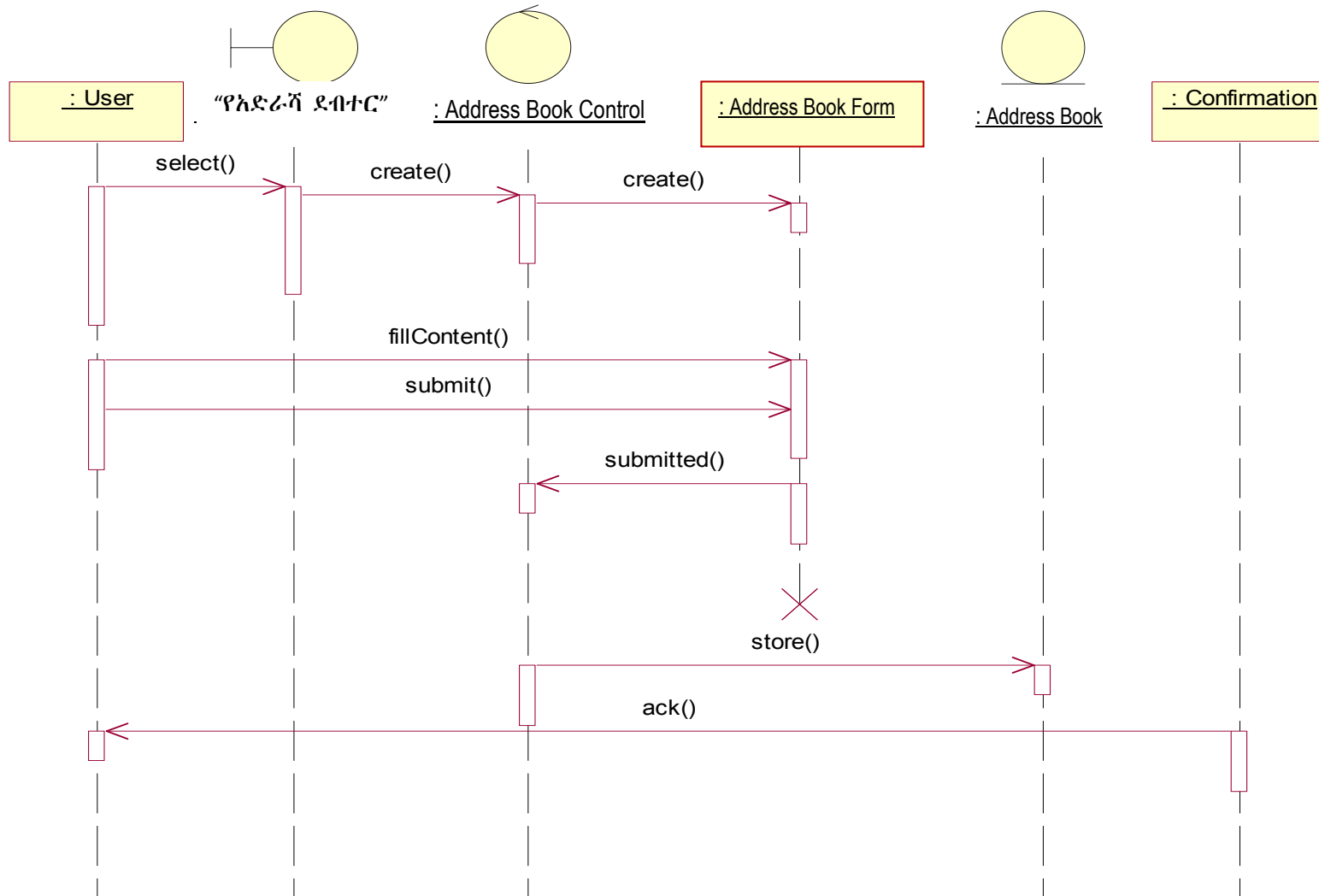


FIG 4.18: Sequence Diagram for Post Address Book Use Case

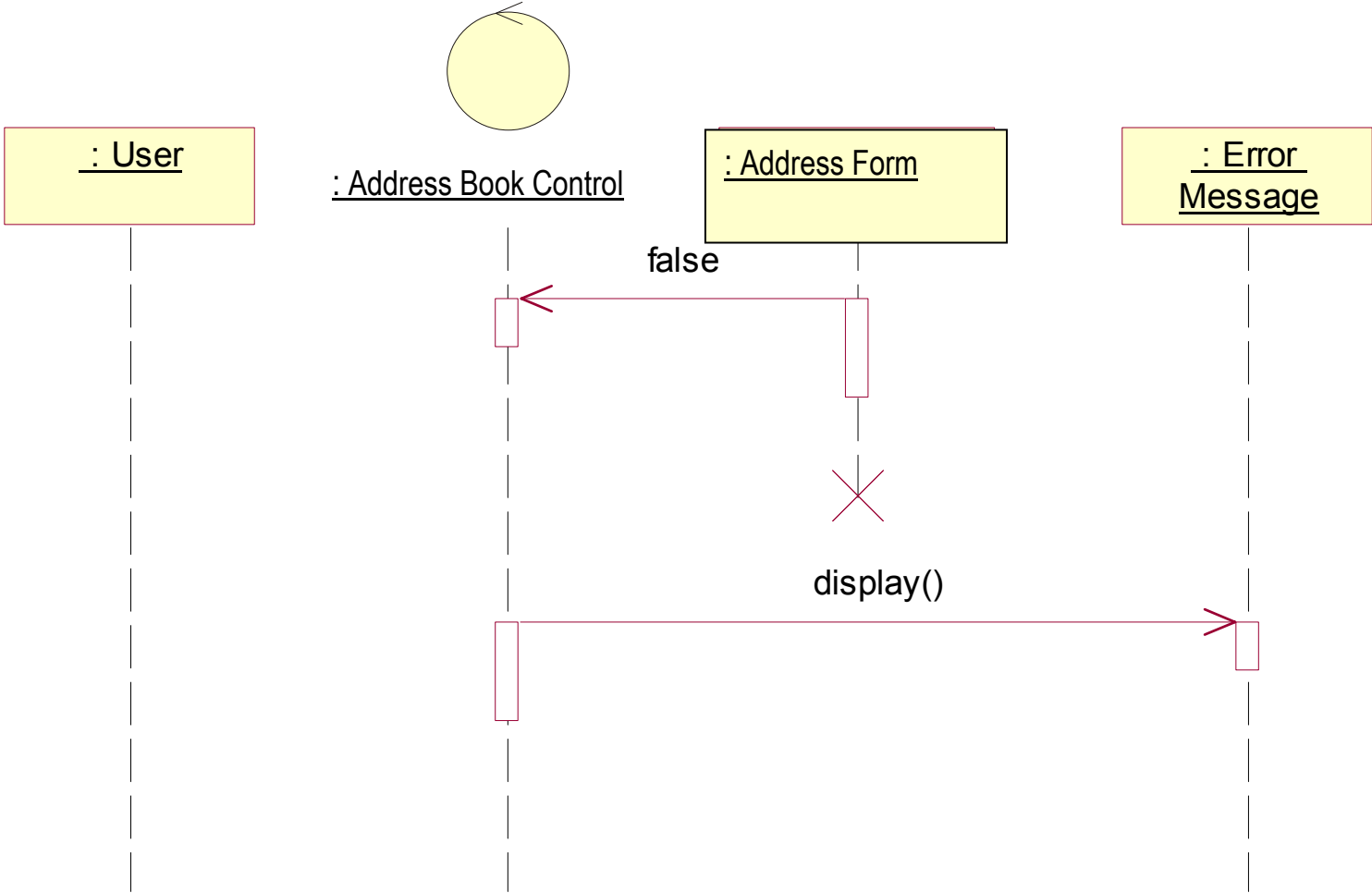


FIG 4.19: Sequence Diagram for Post Address Book Alternate Course A

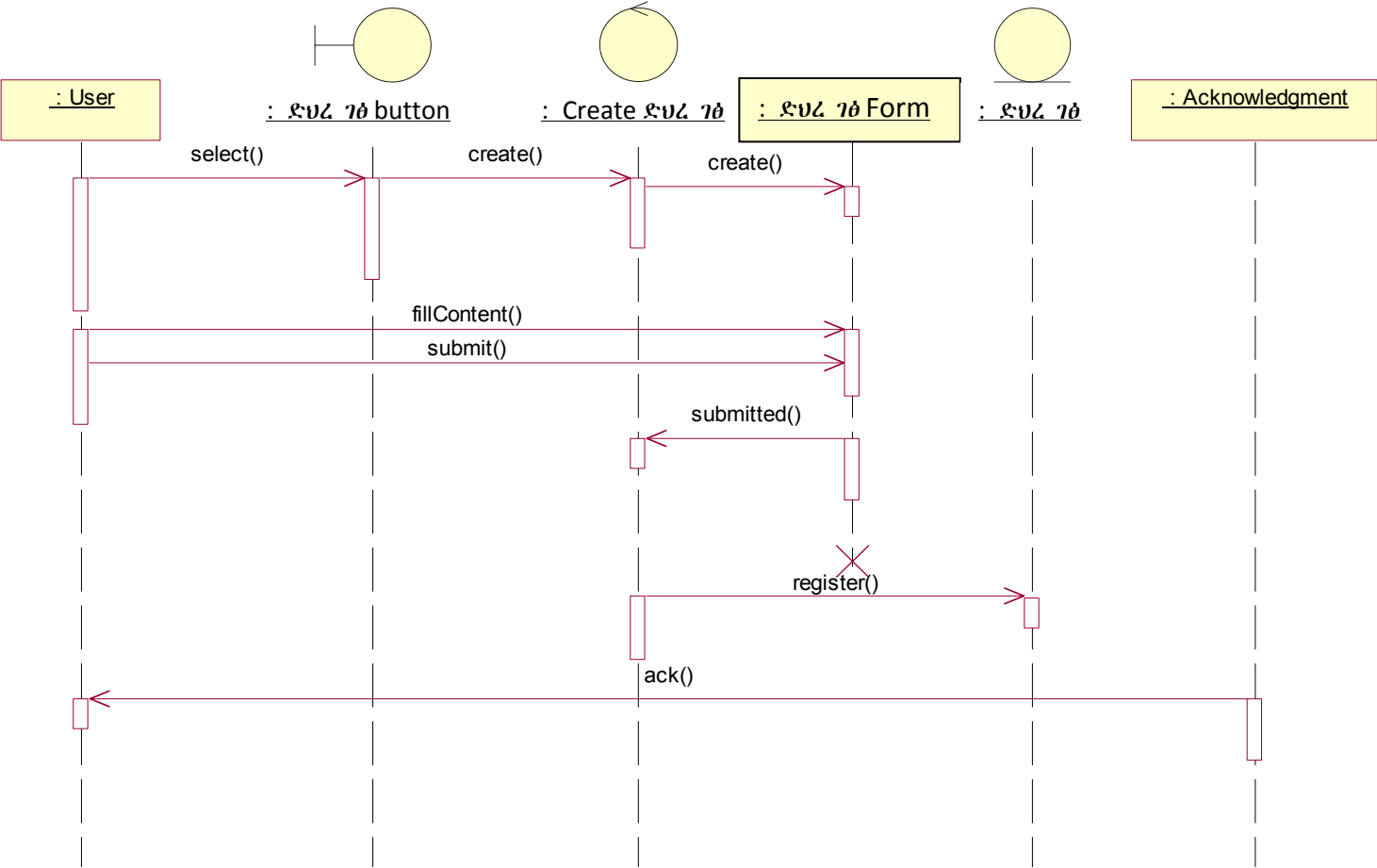


FIG 4.20: Sequence Diagram for Create Personal Website Use Case

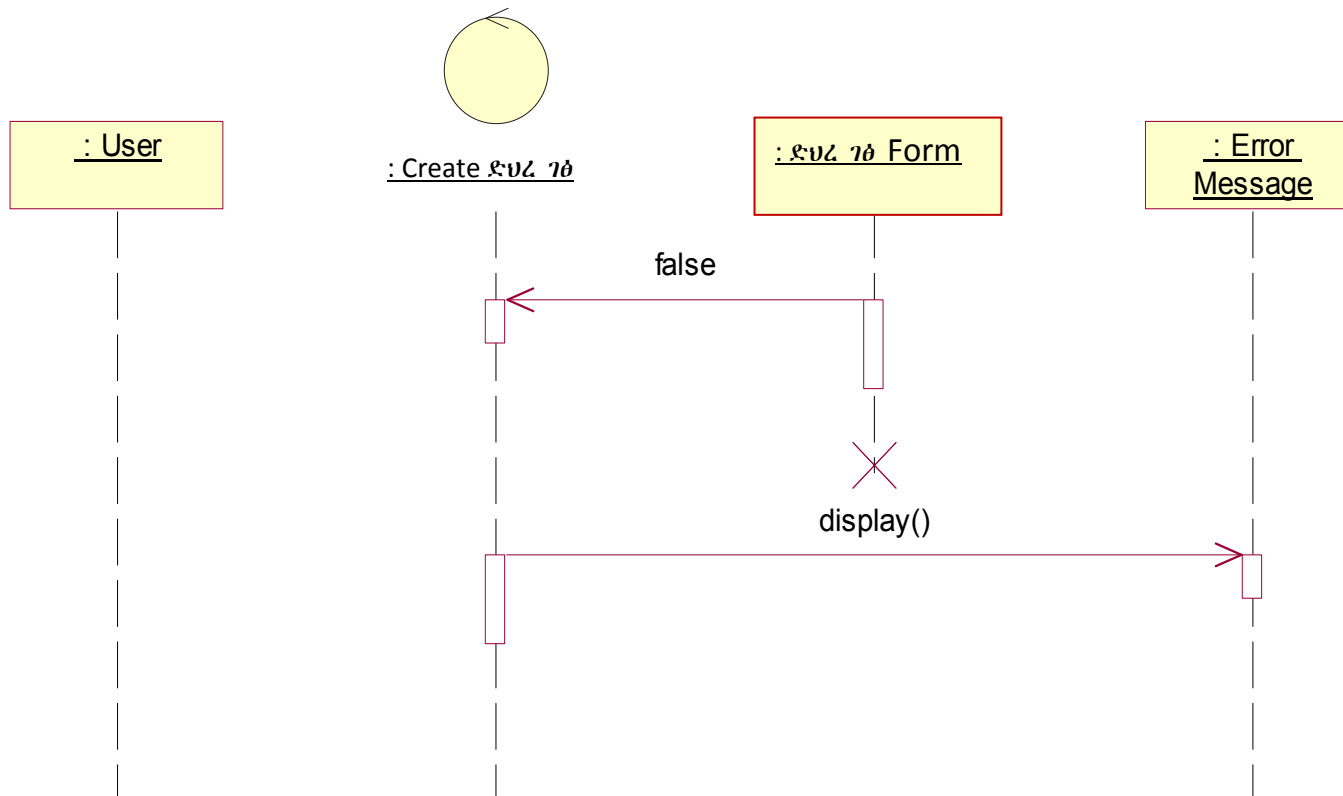


FIG4.21: Sequence Diagram for Create Personal Webpage Alternate Course A

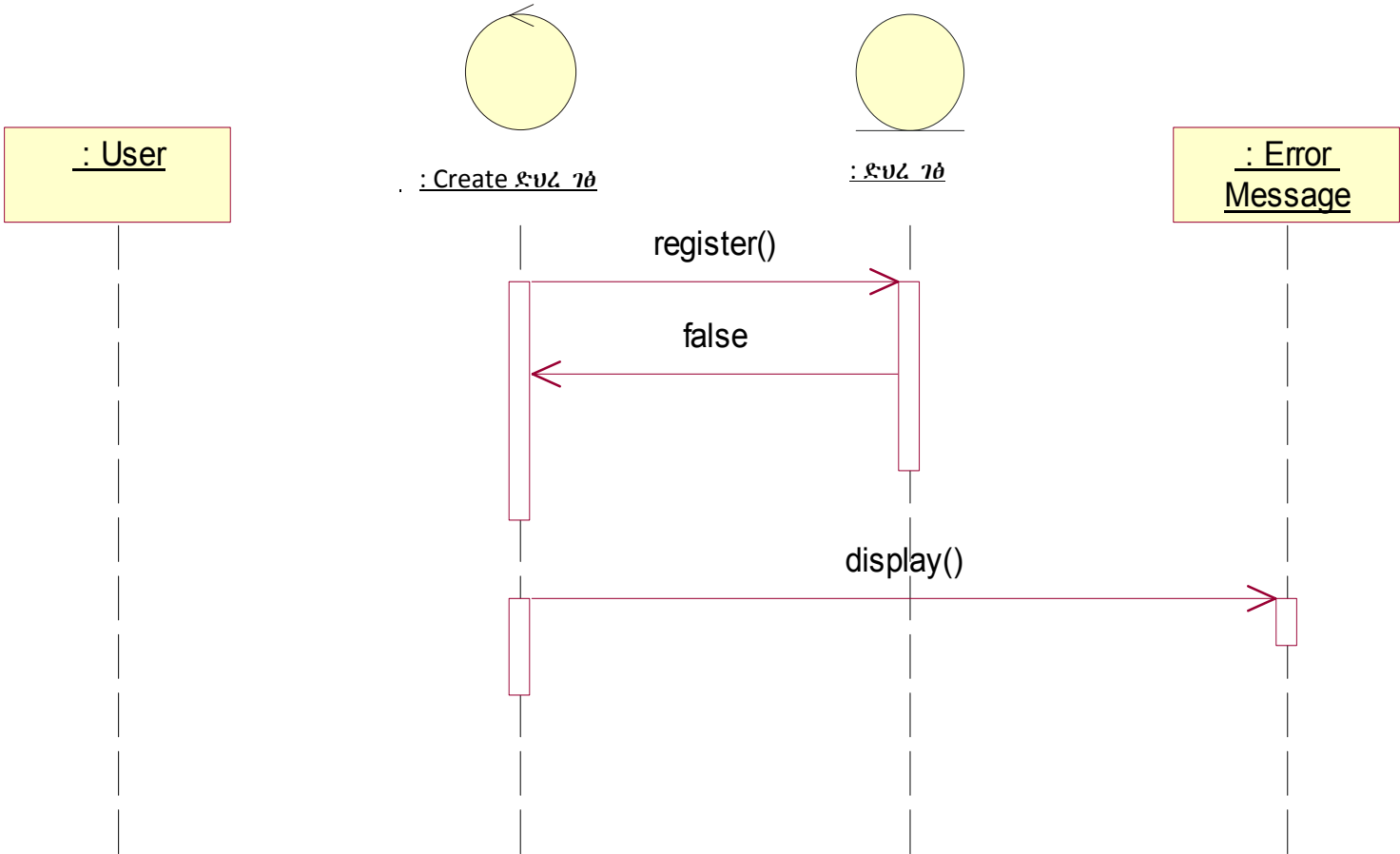


FIG 4.22: Sequence Diagram for Create Personal Website Course B

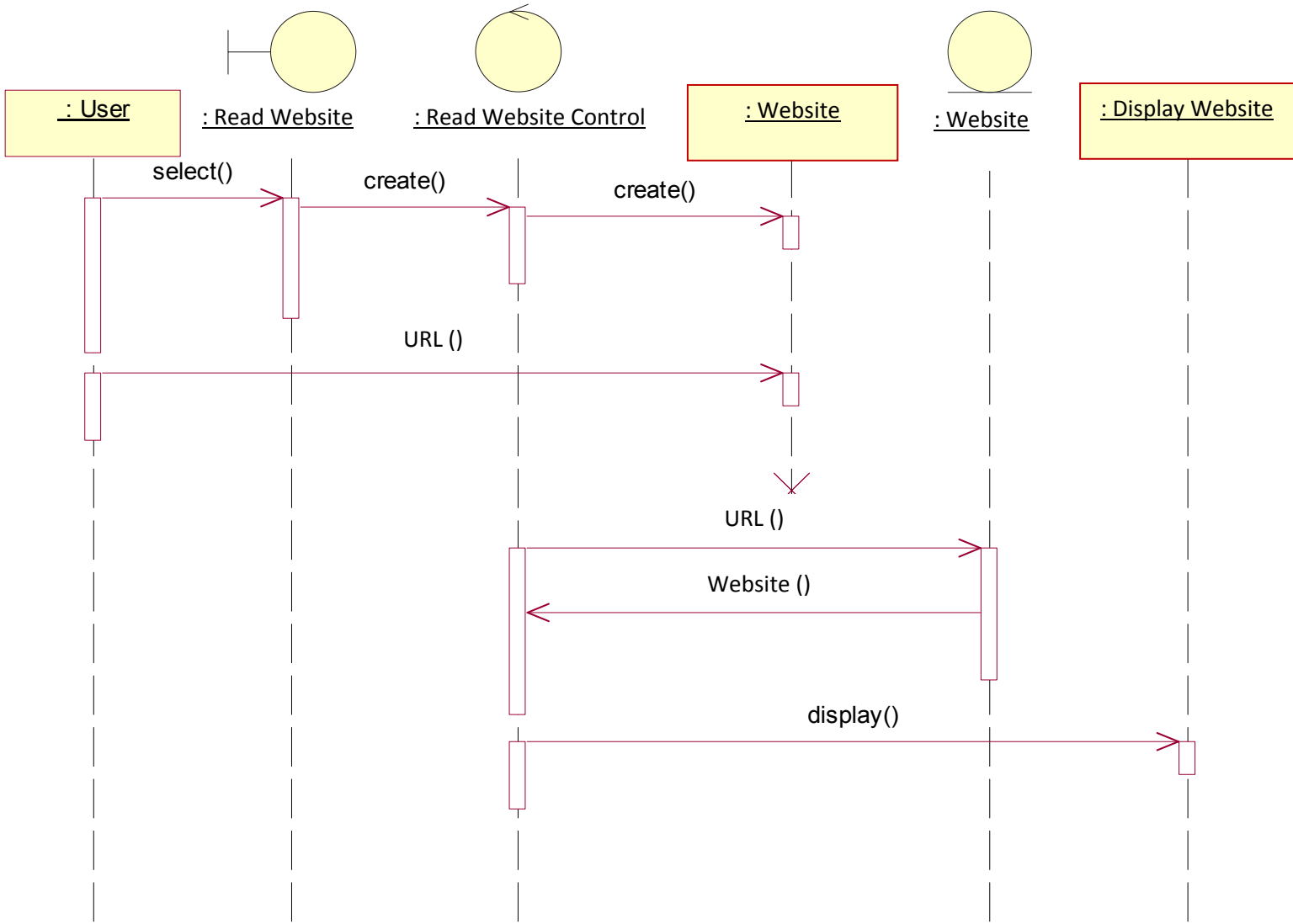


FIG 4.23: Sequence Diagram for Read website Use Case

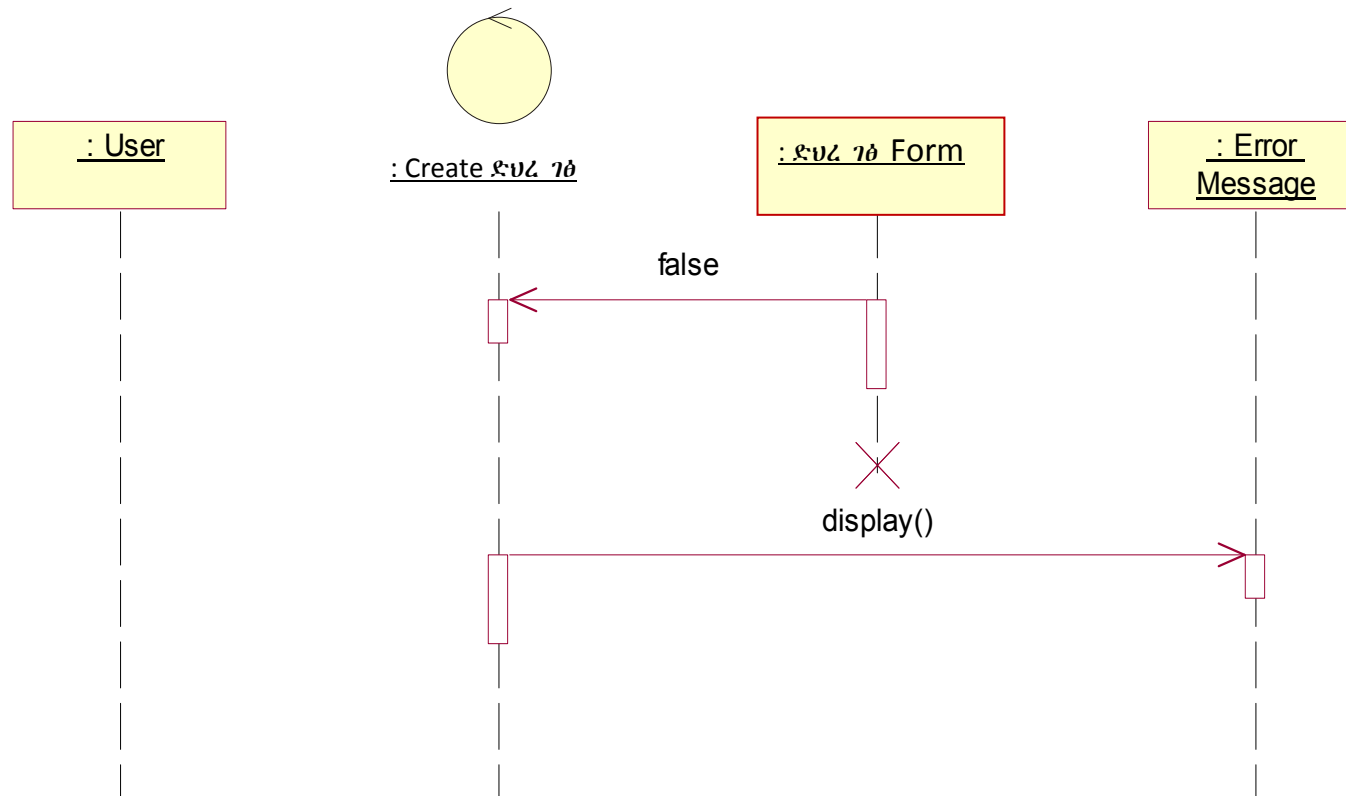


FIG4.24: Sequence Diagram for Read Personal Webpage Alternate Course A

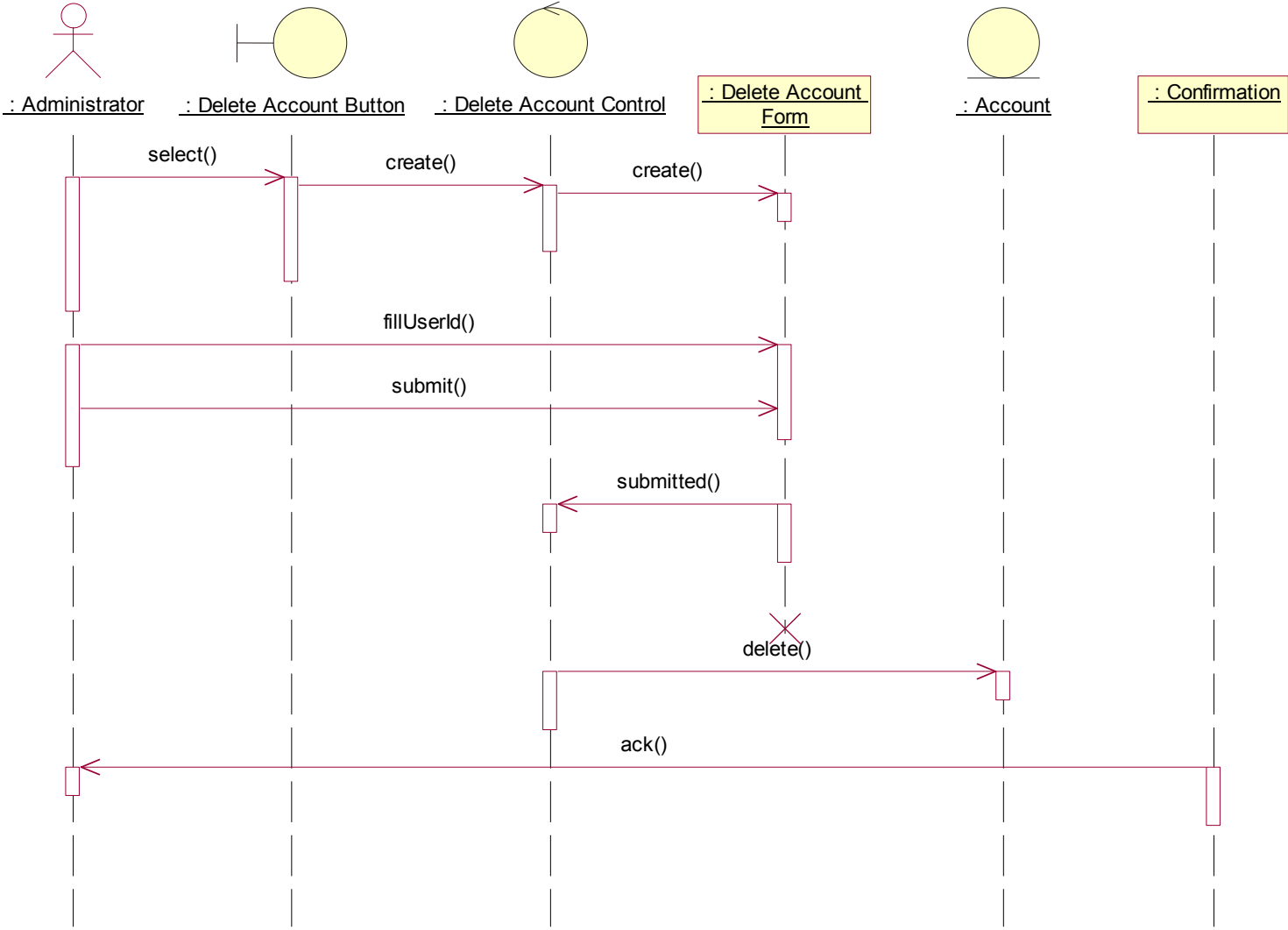


FIG 4.25: Sequence Diagram for Delete Account Use Case

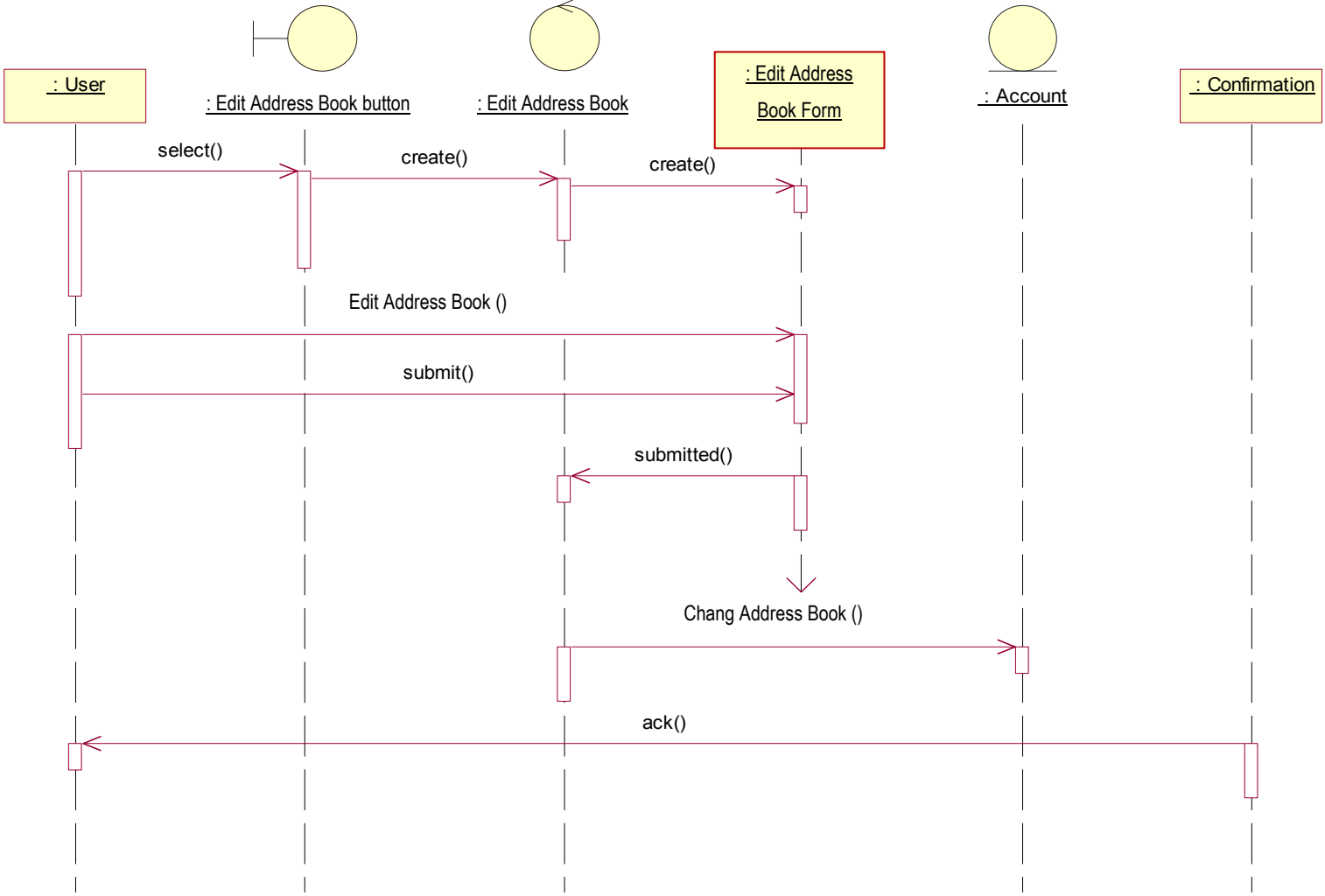


FIG 4.26: Sequence Diagram for Edit Address Book Use Case

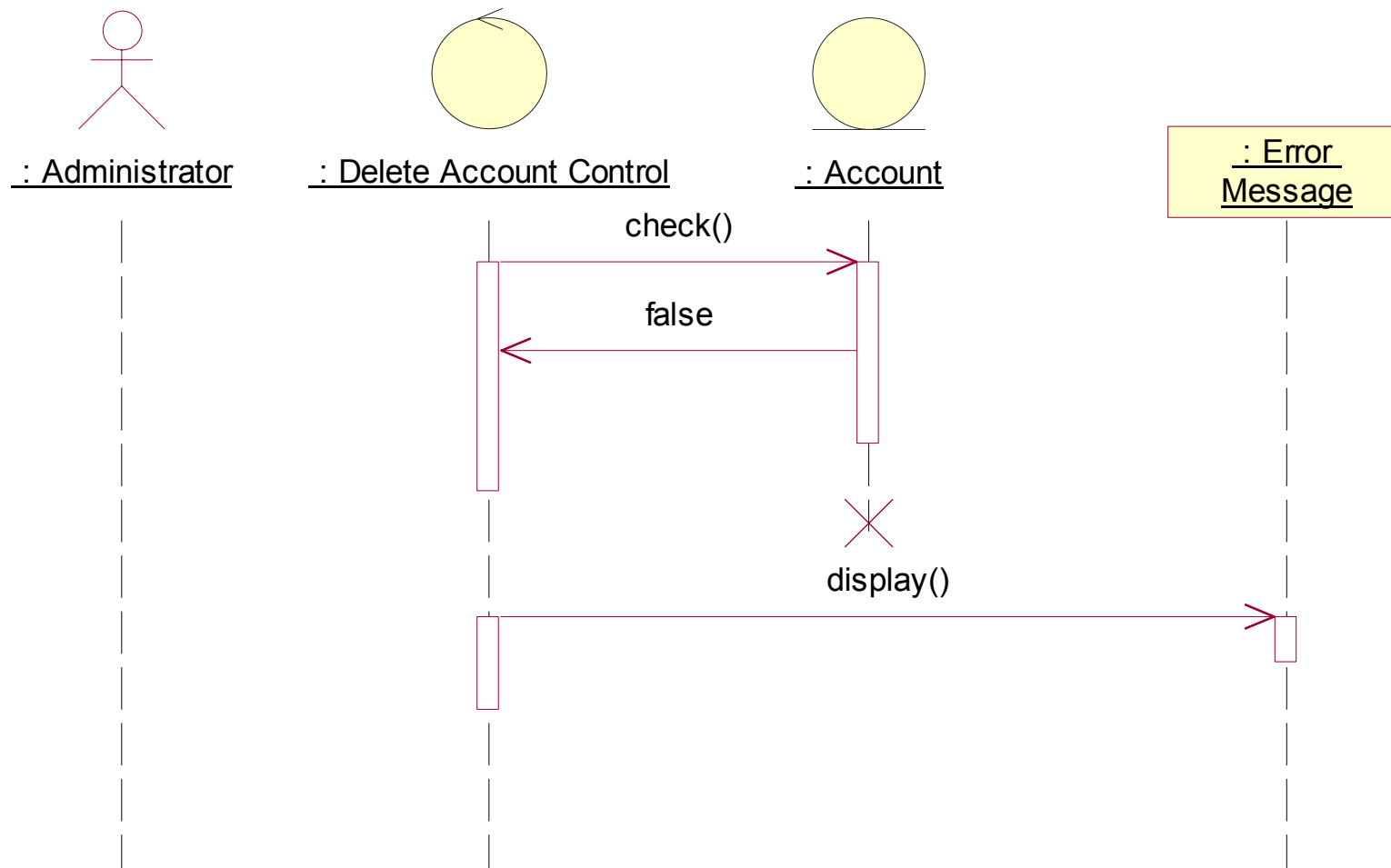


FIG 4.27: Sequence Diagram for Delete Account Alternate Course A

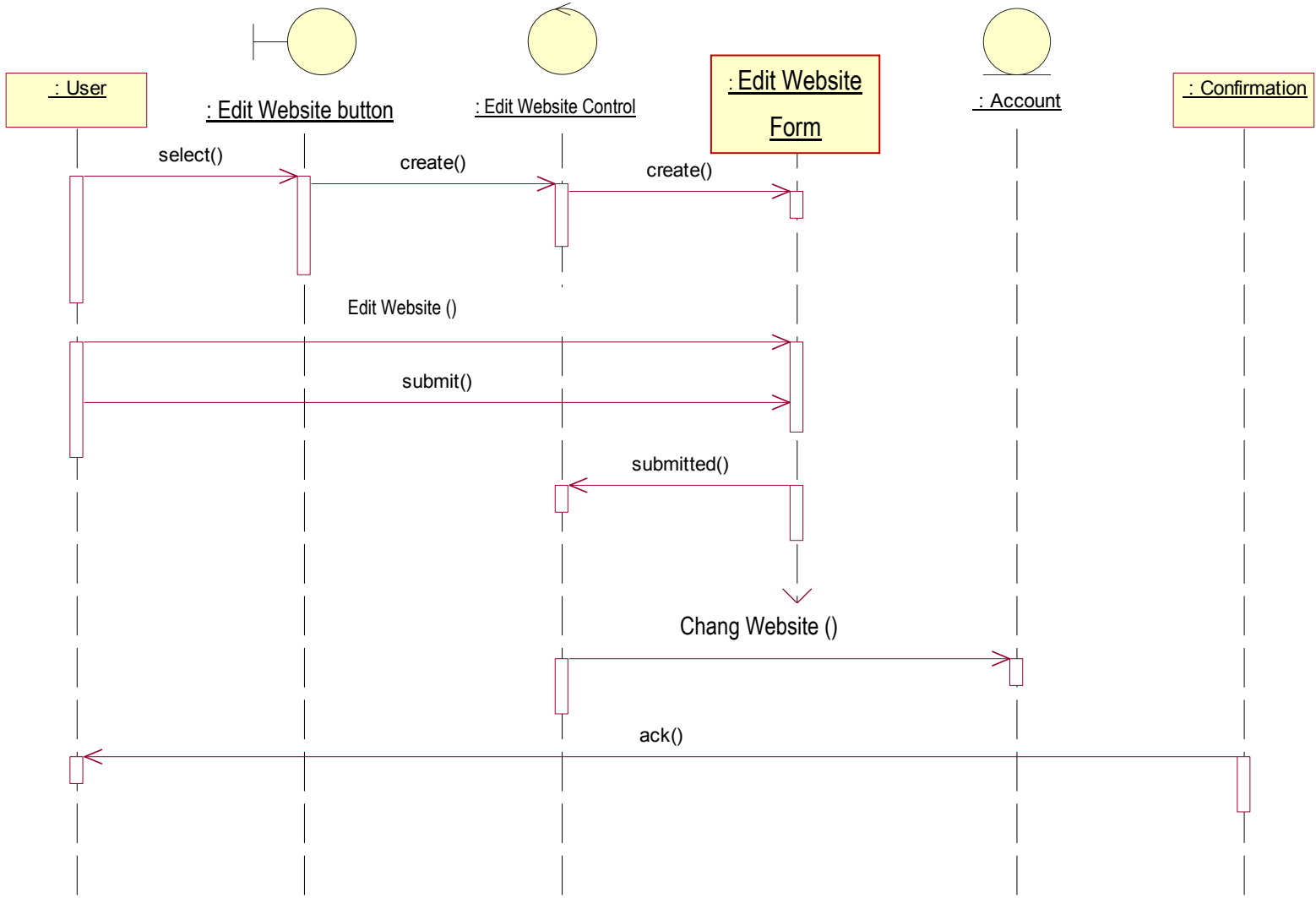


FIG 4.28: Sequence Diagram for Edit Address Book Use Case

## Annex C

### Amharic Keyboard Layout

	e	u	i	a	E		o
h	ሀ	ሁ	ሂ	ሃ	ሄ	ህ	ሆ
l	ለ	ሉ	ሊ	ላ	ሌ	ል	ሎ
H	ሐ	ሑ	ሒ	ሓ	ሔ	ሕ	ሖ
m	መ	ሙ	ሚ	ማ	ሜ	ም	ሞ
s	ሰ	ሱ	ሲ	ሳ	ሴ	ሶ	ሷ
r	ረ	ሩ	ሪ	ራ	ሪ	ር	ሮ
S	ሠ	ሡ	ሢ	ሣ	ሤ	ሥ	ሦ
Shift + s	ሸ	ሹ	ሺ	ሻ	ሼ	ሽ	ሾ
q	ቀ	ቁ	ቂ	ቃ	ቄ	ቅ	ቆ
b	በ	ቡ	ቢ	ባ	ቤ	ብ	ቦ
t	ተ	ቱ	ቲ	ታ	ቲ	ት	ቶ
c	ቸ	ቹ	ቺ	ቻ	ቼ	ች	ቾ
n	ነ	ኑ	ኒ	ና	ኔ	ን	ኖ
Shift + n	ኸ	ኹ	ኺ	ኻ	ኼ	ኽ	ኾ
Shift + h	ኀ	ኁ	ኂ	ኃ	ኄ	ኅ	ኆ
Viols	አ	ኦ	ኧ	ከ	ኩ	ኰ	኱
k	ከ	ኩ	ኪ	ካ	ኬ	ክ	ኸ
Caps lock + h	ኸ	ኹ	ኺ	ኻ	ኼ	ኽ	ኾ
w	ወ	ዉ	ደ	ደ	ደ	ወ	ወ
y	የ	ዩ	ዩ	ያ	ዩ	ይ	ዮ
Shift + a	ዐ	ዑ	ዒ	ዓ	ዔ	ዖ	ዘ
d	ደ	ደ	ደ	ደ	ደ	ደ	ደ
j	ጆ	ጇ	ገ	ገ	ገ	ገ	ገ
g	ገ	ገ	ገ	ገ	ገ	ገ	ገ
Shift + t	ጠ	ጡ	ጢ	ጣ	ጤ	ጥ	ጦ
Shift + c	ጨ	ጨ	ጨ	ጨ	ጨ	ጨ	ጨ
z	ዘ	ዘ	ዘ	ዘ	ዘ	ዘ	ዘ
Shift + z	ዘ	ዘ	ዘ	ዘ	ዘ	ዘ	ዘ
Caps Lock + t	ጸ	ጸ	ጸ	ጸ	ጸ	ጸ	ጸ
Caps Lock + Shift + t	ፀ	ፀ	ፀ	ፀ	ፀ	ፀ	ፀ
f	ፈ	ፈ	ፈ	ፈ	ፈ	ፈ	ፈ
p	ፐ	ፐ	ፐ	ፐ	ፐ	ፐ	ፐ
v	ቨ	ቨ	ቨ	ቨ	ቨ	ቨ	ቨ

*Fig 4.29 Keyboard layout of EventBook System*

---

I, the undersigned, declared that this project is an original work, and has not been presented for a degree in any other university. All the source of the materials used in this project is acknowledged.

Full Name

Gizaw Tulu Bedana

Signature

---

Adviser confirmation

**Full Name**

Dr. Dida Midekso

Department of Computer Science

Addis Ababa University

Addis Ababa

Ethiopia

**Signature**

---