KIKAKUYA Software Requirements Analysis and Design

COMP 3059 – CAPSTONE PROJECT I

Aline Neves Alencar – 101036808
Kie Ogiya – 100984638
Maria Alyssa Villacete – 100923181
Princess Ilasin - 100879176

Table of Contents

1. INtroduction 1
   1.1 Project Purpose 1
   1.2 Project Scope 1
      1.2.1 In Scope 1
      1.2.2 Out of Scope 2
2. System Overview 3
   2.1 Project Perspective 3
   2.2 System Context 3
   2.3 General Constraints 3
   2.4 Assumptions and Dependencies 3
      2.4.1 External dependencies 4
      2.4.2 Internal dependencies 4
3. Functional requirements 5
   3.1 Functional Requirement 5
   3.2 Use Cases 6
   3.3 Data Modelling and Analysis 7
      3.3.1 Normalized Data Model Diagram 7
      3.3.2 Activity Diagrams 8
      3.3.3 Sequence Diagrams 22
      3.3.4 UML Class Diagrams 30
   3.4 Process Modelling 31
      3.4.1 Data Flow Diagram 31
4. Non-functional requirements 32
5. logical database requirements 32
6. Approval 33

Table of Figures

Figure 1: Use Case Diagram 6
Figure 2: Normalized Data Model Diagram 7
Figure 3: Activity Diagram - User Log In / Sign Up 8
Figure 4: Activity Diagram - View Calendar 9
Figure 5: Activity Diagram - Manage Calendar 10
Figure 6: Activity Diagram - Create To-do List 11
Figure 7: Activity Diagram - Manage To-do List 12
Figure 8: Activity Diagram - Check To-do List 13
Figure 9: Activity Diagram - Search Vendors 14
Figure 10: Activity Diagram - Send Announcement 15
Figure 11: Activity Diagram - Send RSVP 16
Figure 12: Activity Diagram - Receive Announcement / RSVP 17
Figure 13: Activity Diagram - Response RSVP 18
Figure 14: Activity Diagram - Manage Budget 19
Figure 15: Activity Diagram - Create Guestlist 20
Figure 16: Activity Diagram - Manage Guestlist 21
Figure 17: Sequence Diagram - Create Profile 22
Figure 18: Sequence Diagram - Login User 22
Figure 19: Sequence Diagram - View Calendar 23
Figure 20: Sequence Diagram - Manage Calendar 23
Figure 21: Sequence Diagram - Create To-do List 24
Figure 22: Sequence Diagram - Manage To-do LIst 24
Figure 23: Sequence Diagram - Check To-do List 25
Figure 24:Sequence Diagram - Search Vendors 25
Figure 25: Sequence Diagram - Send Announcement 26
Figure 26: Sequence Diagram - Send RSVP 26
Figure 27: Sequence Diagram - Receive Announcements / RSVP 27
Figure 28: Sequence Diagram – Response RSVP 27
Figure 29: Sequence Diagram - Manage Budget 28
Figure 30: Sequence Diagram - Create Guest List 28
Figure 31: Sequence Diagram - Manage Guest List 29
Figure 32: UML Class Diagram 30
Figure 33: Data Flow Diagram 31

1. INTRODUCTION

1.1 Project Purpose

The purpose of the project is to build an online responsive web application that aims to assist its user in the planning of any kind of event. The application seeks to provide the user with an uncomplicated and centralized tool capable of assisting them in all aspects of the event planning process.

This document describes the high level software requirements for the system. It describes the system overview as well what it is expected to accomplish for the stakeholders, the system engineers and the project consultant.

The scope of the project was expanded regarding the profiles feature. From this point on, every user will have the ability to create multiple events in their account.

1.2 Project Scope

KIKAKUYA will be developed as Online Responsive Web Application. Following are the features that need to be accomplished to deliver a service.

1.2.1 In Scope

System of the application

1. **Type of the system**: The system developed will be an Online Responsive Web Application.
2. **Yelp**: The Yelp Fusion API (see glossary) will provide all the data regarding the vendors. Since it is an API that this application will rely on for the implementation of a few requirements, it is considered a risk to the success of the project.

Feature of the application

1. **Profile:** Users register their profiles, with details about the event they wish to plan.
2. **Dashboard:** The home screen of the application will provide easy access to the features of the application.
3. **Calendar:** It allows the user to overview their schedule and manage their appointments.
4. **Lists:** A user can have multiple lists with an option to tick off tasks that were completed or to update their status (started, completed).

5. **Budget Management:** Allows the user to manage their budgets by offering an overview of how much is allocated to each vendor, as well as a price comparison between different services/vendors.
6. **Search for Vendors:** Users can search for vendors in their areas according to their category.
7. **Guest List:** All guests invited can be listed in the guest list and will receive announcements such as invites and RSVPs (see glossary) by email. Their response to the RSVP will be done through a link in the email.

1.2.2 Out of Scope

System of the application

1. **Development:** Languages and frameworks that will be used to develop the application.
2. **Database:** The database management system where the data will be stored.
3. **Hosting Architecture Design:** The service where the application will be hosted.

Feature of the application

1. **Bookmark:** User's ability to bookmark their favorite vendors for future access.
2. **Multiple profiles:** The user will be allowed to create more than one profile in case of multiple events that are being simultaneously organized.
3. **Interface customization:** The customization of the colors in the user interface, with the purpose of making the dashboard more personal to the user.
4. **Countdown:** A visible countdown to the event's day in the user's dashboard.
5. **Price:** The price of goods and services offered by the vendors found through the application will not be automatically generated. The price will be inputted by the user in the budget control feature according to their own research.
6. **Collaborators:** Another type of user, with limited access to the event planner's account.

2. SYSTEM OVERVIEW

The KIKAKUYA is an online responsive web application that aims to assist its user in the planning of any kind of event. The application seeks to provide the event planner with an uncomplicated and centralized tool capable of assisting them in all aspects of the event planning process.

2.1 Project Perspective

The event planning management app, KIKAKUYA, is a new self-contained system. The web application will not rely on other existing event planning apps. All parts of the system will be developed in-house and within the team.

2.2 System Context

The KIKAKUYA expands from the initial concept of an event planning web app and takes into consideration the people who want to host and organize a big event but do not want to incur the extra cost of hiring a professional consultant. It is an all-in- one centralized application that would contain different tools and features which will help users plan an event easier and more effectively.

2.3 General Constraints

The following are the technical constraints that could impact the application:

1. **Yelp API**: Since Yelp is an external company, not related to this project's team, the reliable and constant functioning of the API is essential, and its failure might impose a limitation in the functionality of the application.
2. **Server Hosting**: Any internal errors coming from the server provider will directly impact the application.
3. **Responsiveness of other features**: While the application is responsive, the budget control and calendar occupy a bigger screen space, and their functionality on smaller screens might be limited.
4. **Browser Compatibility**: Since the application is web-based, the user will need a browser to access it. Additionally, the browser they use must be compatible with the technologies used in the application.

2.4 Assumptions and Dependencies

1. The Yelp Fusion API will work as intended, providing the best suggested vendors (and their details) in each area and category.

2. The devices used by the users will have all software and hardware required for the running of the application.
3. The users of the application will have a connection to the internet to access and interact with the application.
4. The users will have an email account to create their account for the application.
5. The guests will have an email account so they can receive announcements and respond to the invitation.

The following are internal and external dependencies that may impact the project's desired result:

2.4.1 External dependencies

1. The implementation of the vendors search feature depends on the reliable functioning of the Yelp Fusion API.
2. The web application is dependent on the well-functioning of the server where it will be hosted.
3. The efficiency and certainty in the retrieval of information saved by the user in the application depends on a consistent and well-structured database.

2.4.2 Internal dependencies

1. The analysis and gathering of software requirements is dependent on the delimitation of the project's scope.
2. The development of the application is dependent of the completion of the design and analysis of the system.
3. The deployment of the application is dependent of the success status in the testing phase.

3. FUNCTIONAL REQUIREMENTS

3.1 Functional Requirement

FR1 The system shall store a user's profile after a user fills out a form with details about an event, thus signing up for the application.

FR2 The system shall delete a user's profile from the database if they choose to delete their account.

FR3 The system shall allow access to the event details after the user is authenticated through a log in.

FR4 The system shall store the name and email of all guests.

FR5 The system shall send emails to the guests when the user chooses to send an announcement.

FR6 The system shall send an email to guests with a link for them to respond 'yes' or 'no' regarding their presence at the event, and store the response.

FR7 The system shall display the business' name, rating and contact information in the vendor search results.

FR8 The system shall send and in-app notification to the user when a reminder from the calendar approaches.

FR9 The system shall store the name, day, time and notes about an appointment when the user creates a new entry in the calendar, as well as update it and delete it when the user edits said entry.

FR10 The system shall store the title and items that belong to a list when the user creates one.

FR11 The system shall store the title and status of an item and the list it belongs to when the user adds an item in a list.

FR12 The system shall store the vendor name, price and notes about a service entered by the user in the different vendor categories in the budget management.

FR13 The system shall calculate the best combination of services, according to their prices, with the goal of displaying to the user the lowest cost possible.

FR14 The interface of the system shall be responsive.

3.2 Use Cases

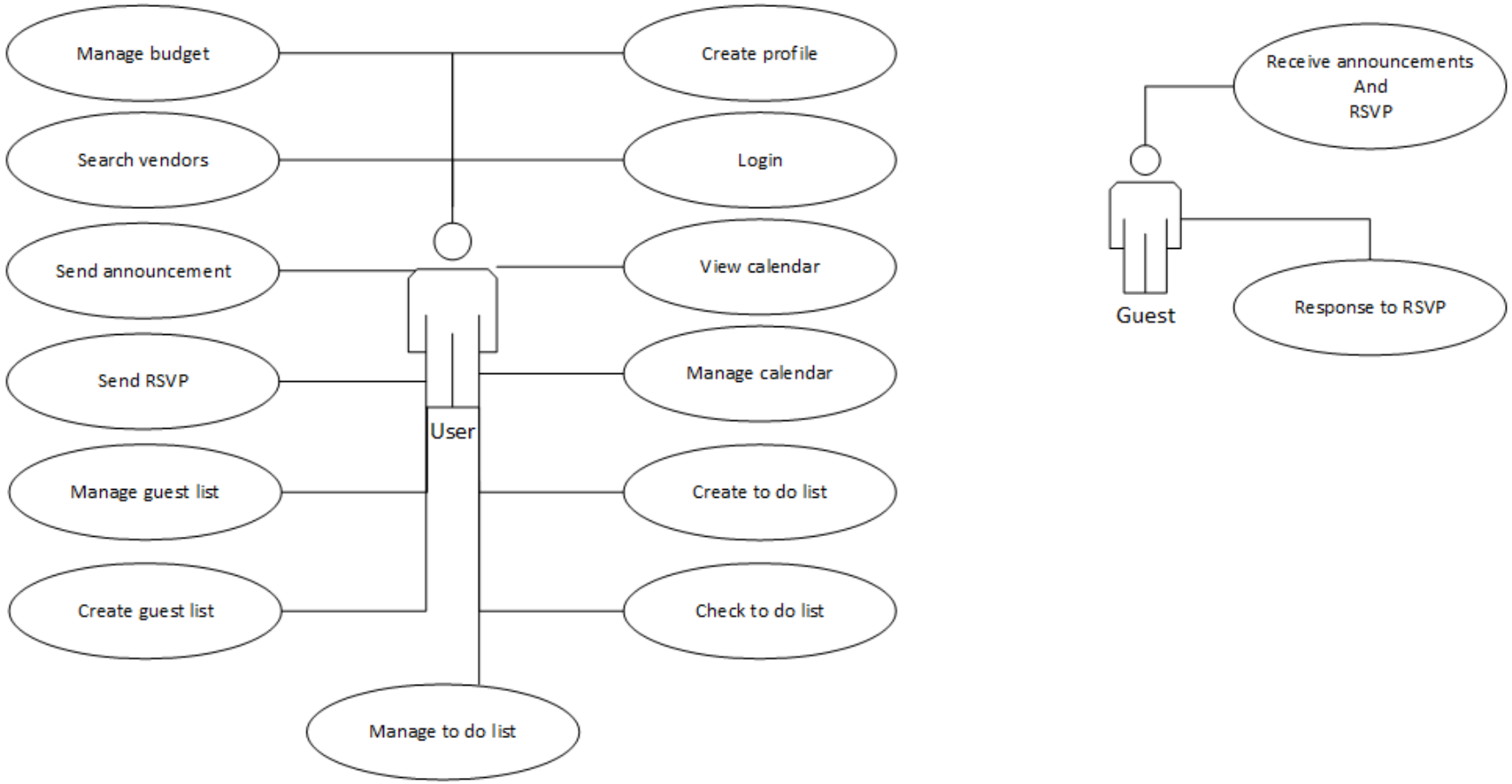

Figure 1: Use Case Diagram

3.3 Data Modelling and Analysis

3.3.1 Normalized Data Model Diagram

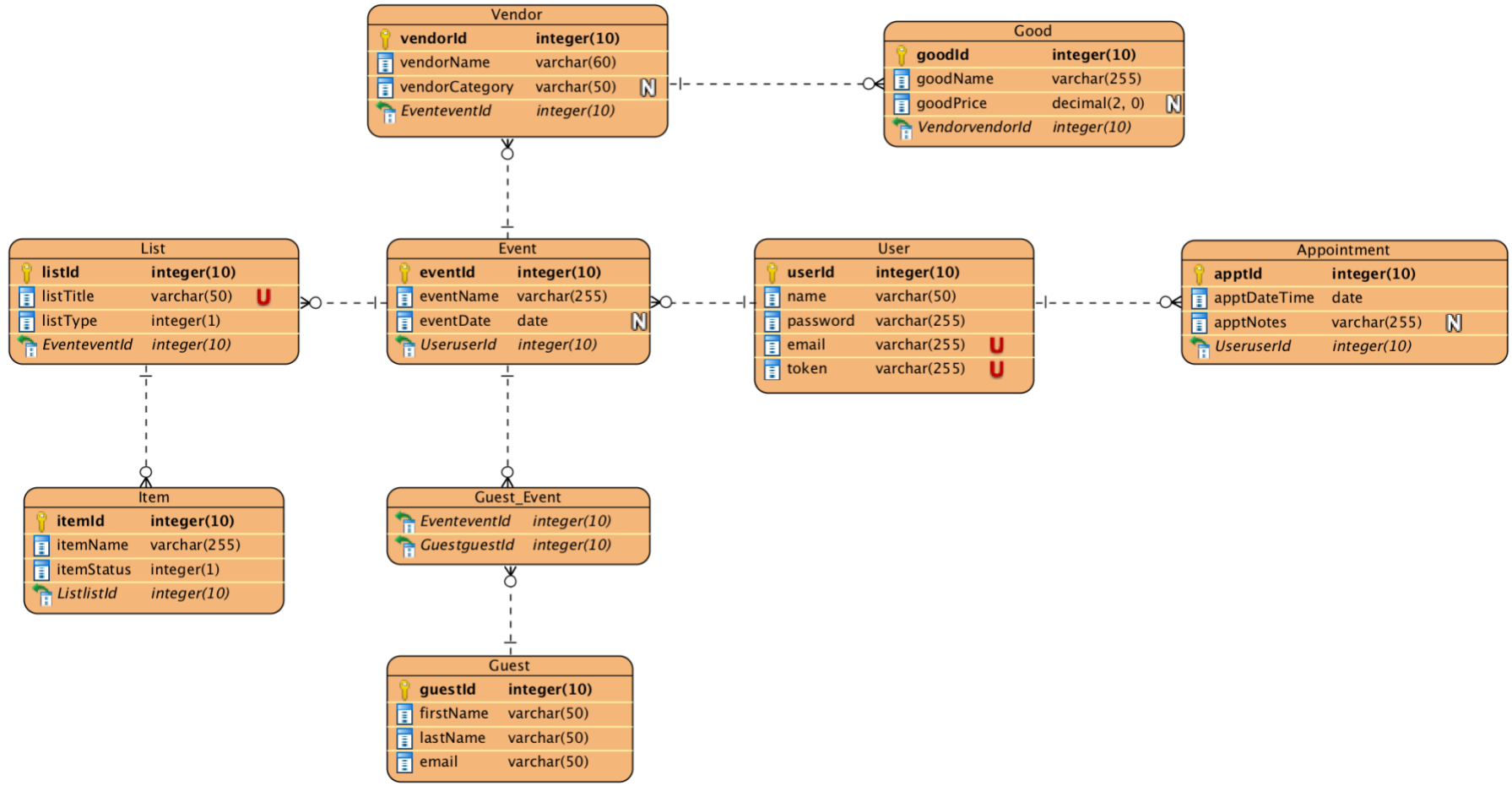

Figure 2: Normalized Data Model Diagram

3.3.2 Activity Diagrams

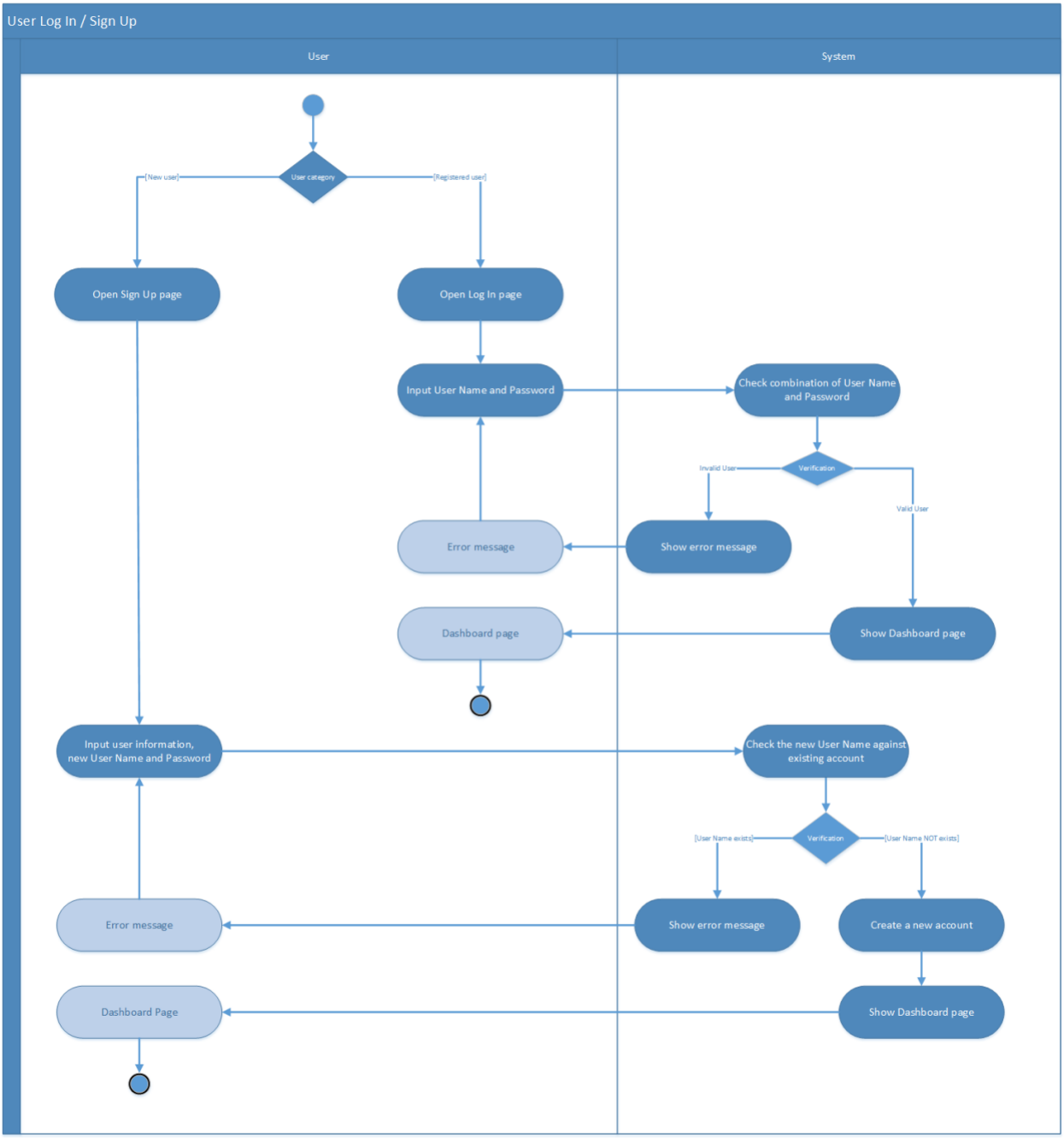

Figure 3: Activity Diagram - User Log In / Sign Up

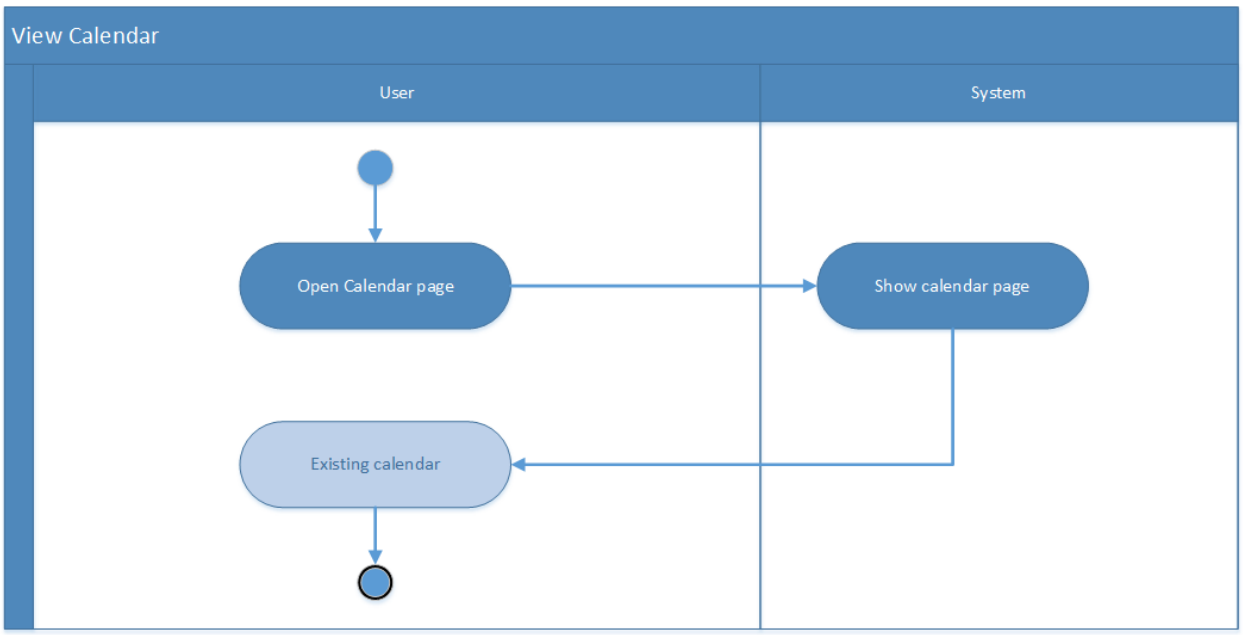

Figure 4: Activity Diagram - View Calendar

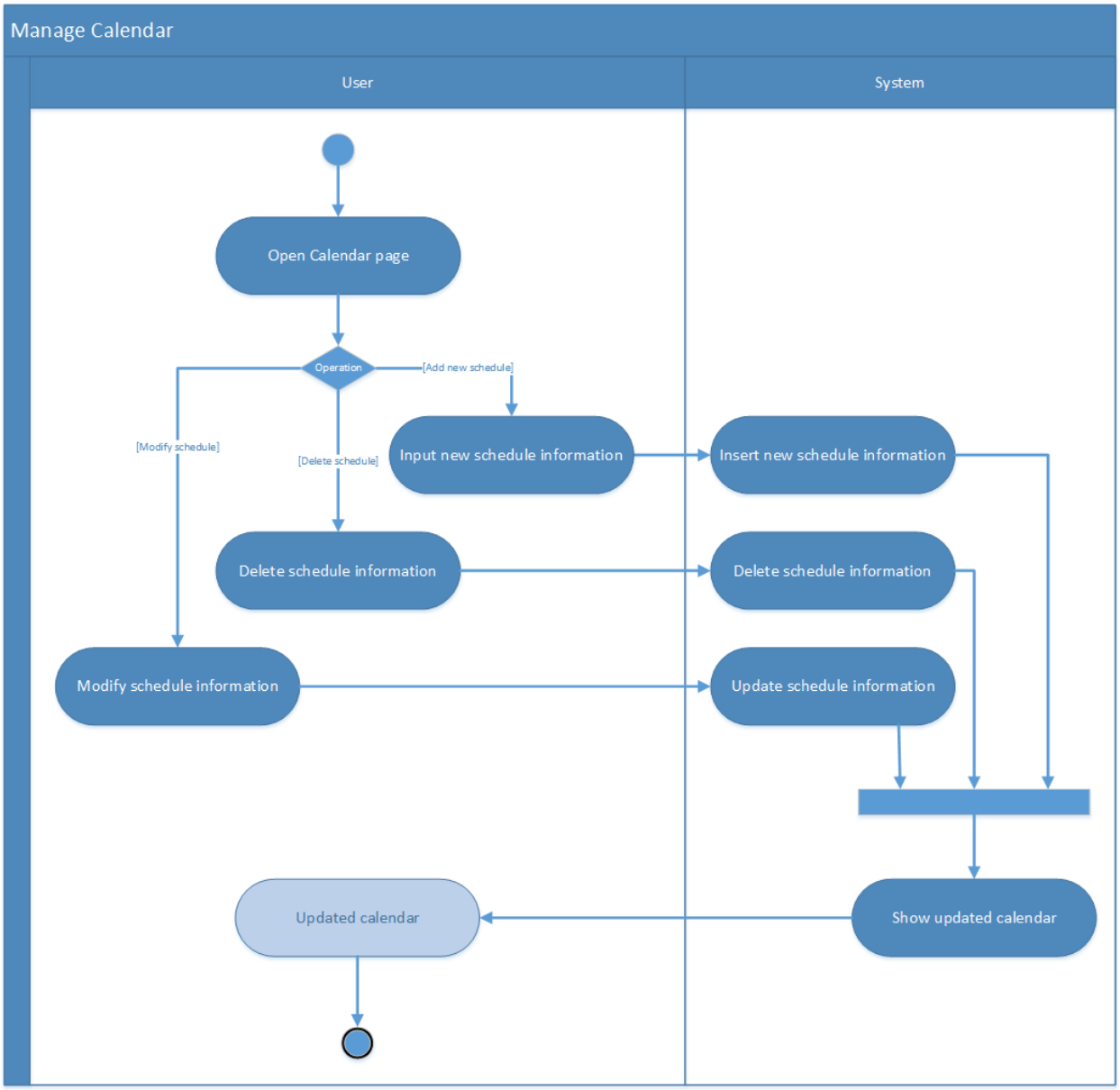

Figure 5: Activity Diagram - Manage Calendar

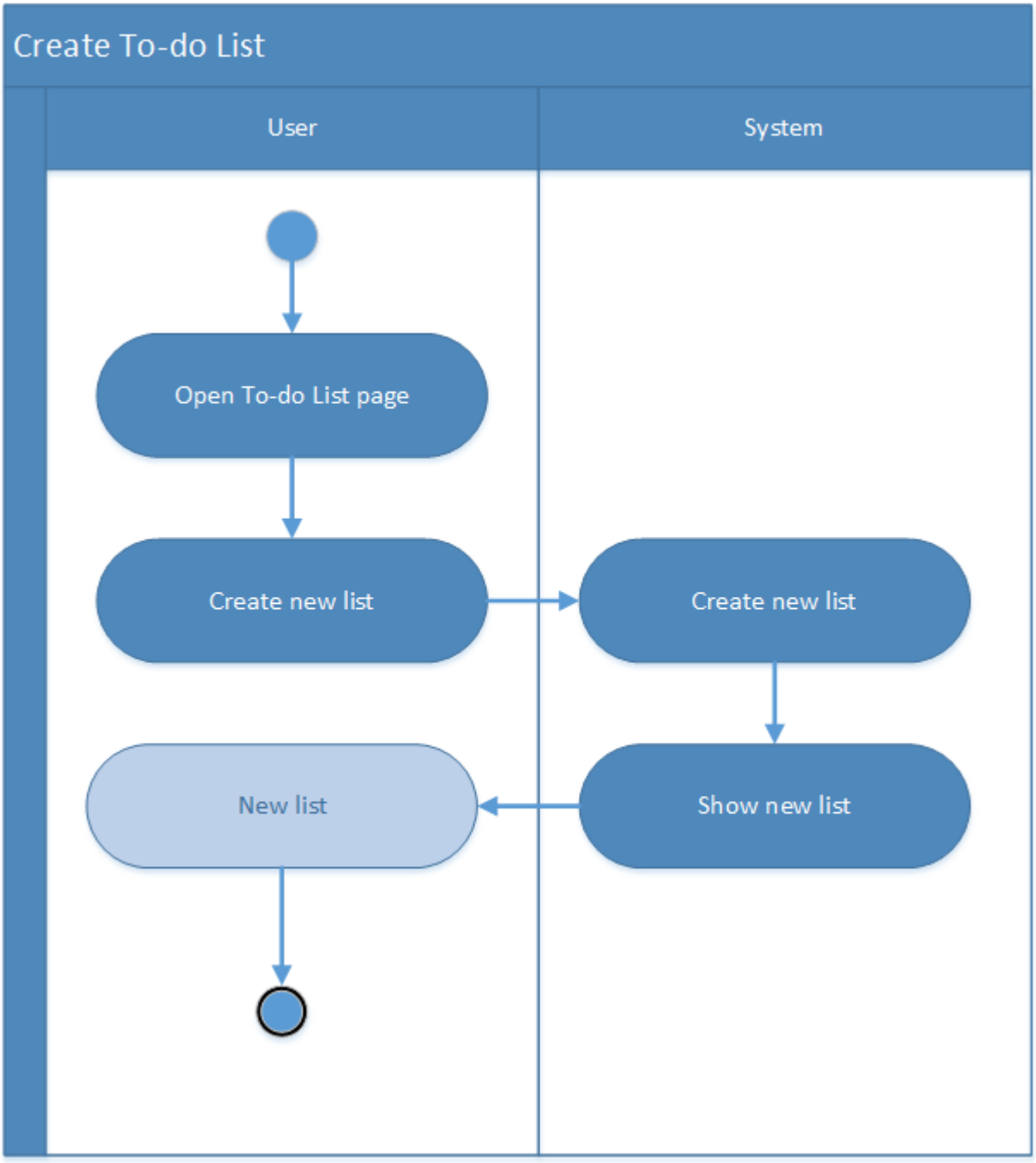

Figure 6: Activity Diagram - Create To-do List

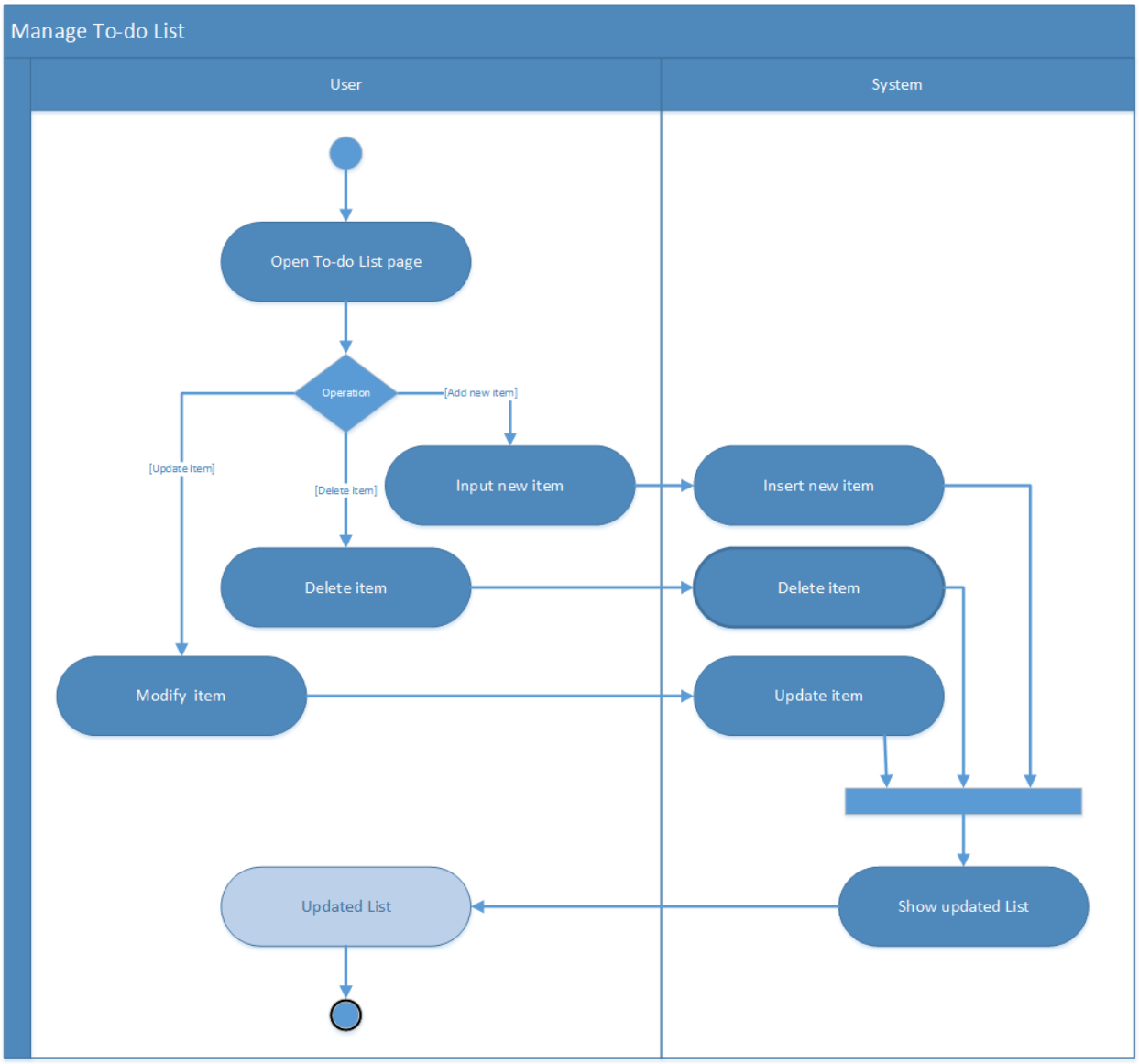

Figure 7: Activity Diagram - Manage To-do List

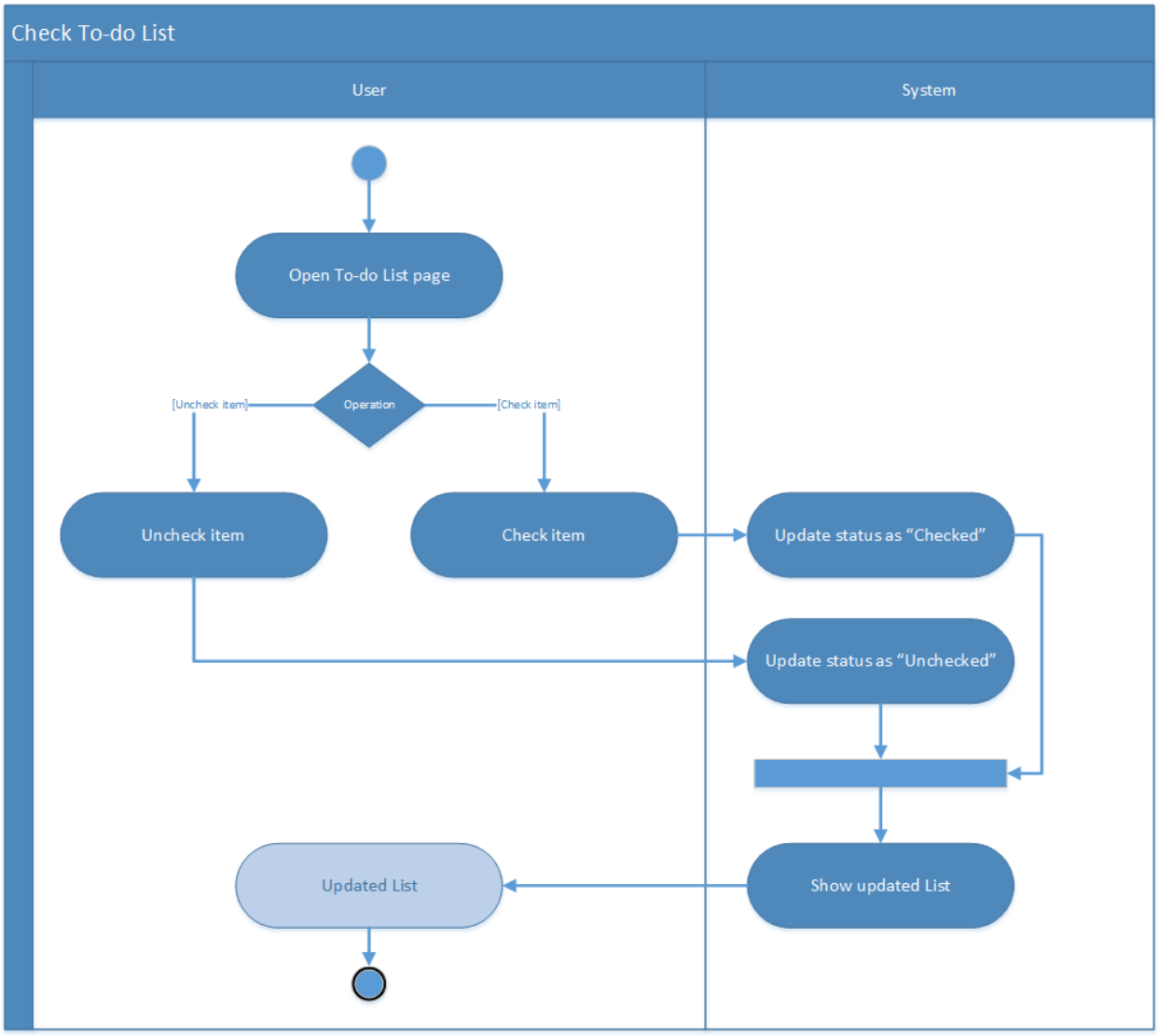

Figure 8: Activity Diagram - Check To-do List

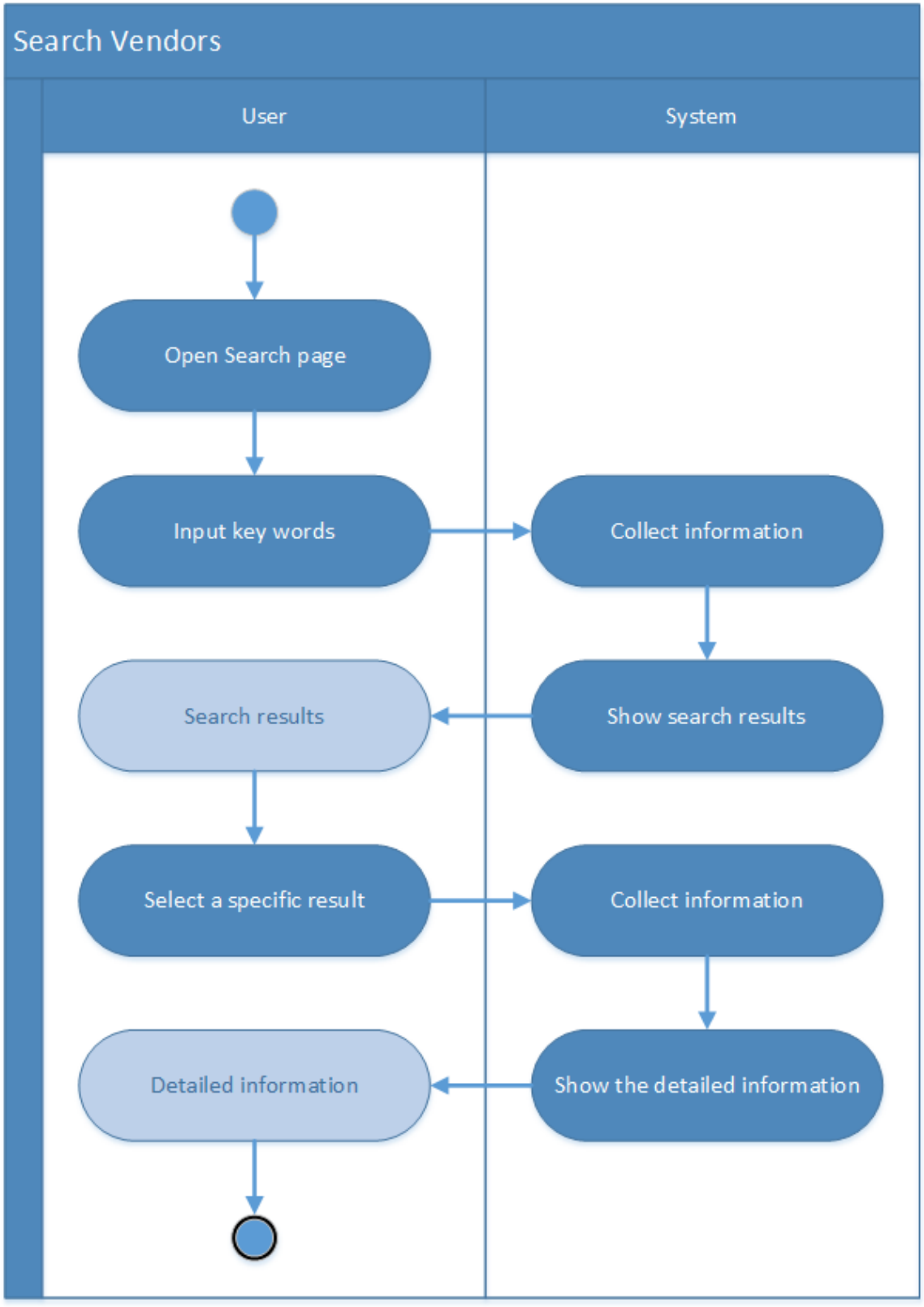

Figure 9: Activity Diagram - Search Vendors

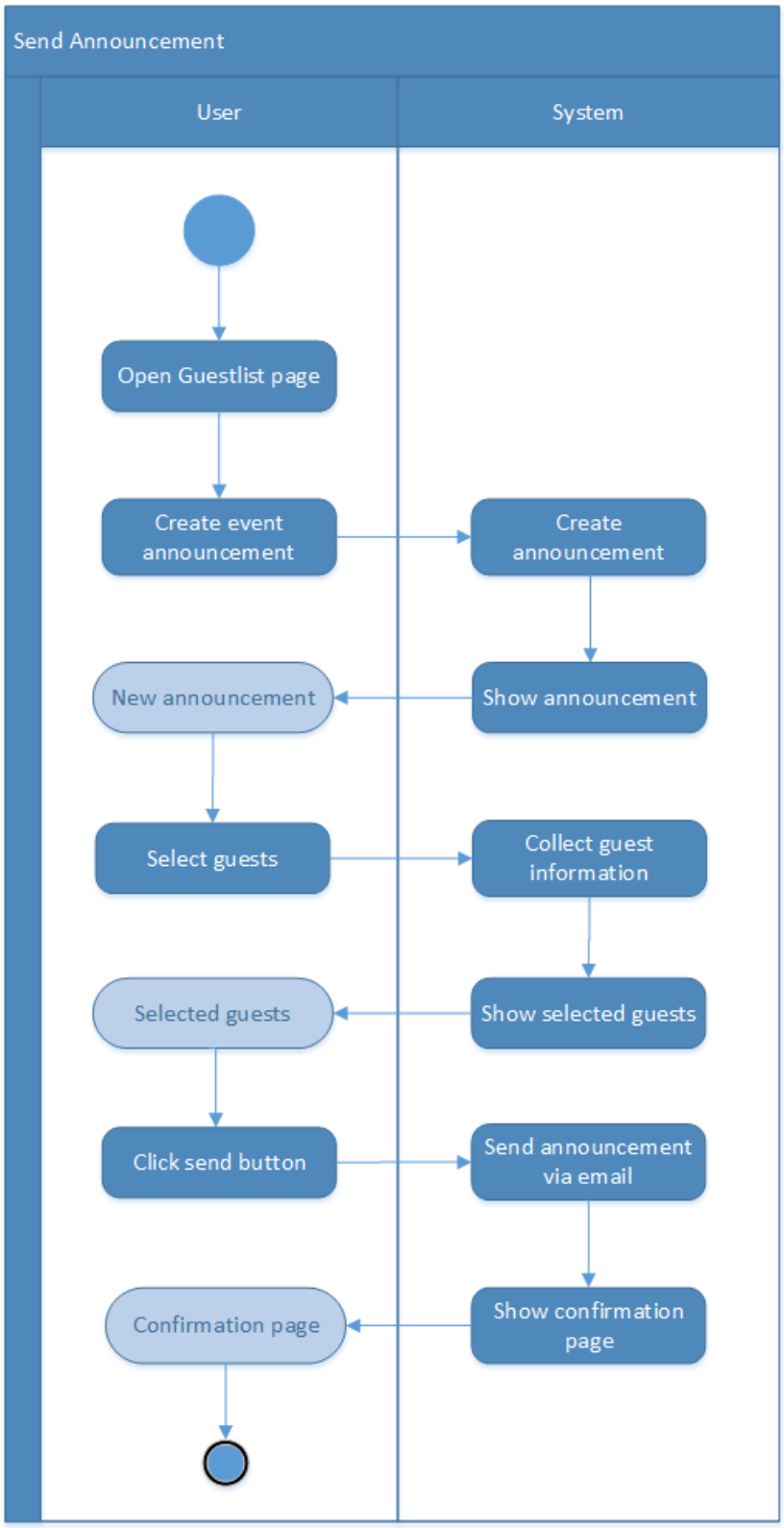

Figure 10: Activity Diagram - Send Announcement

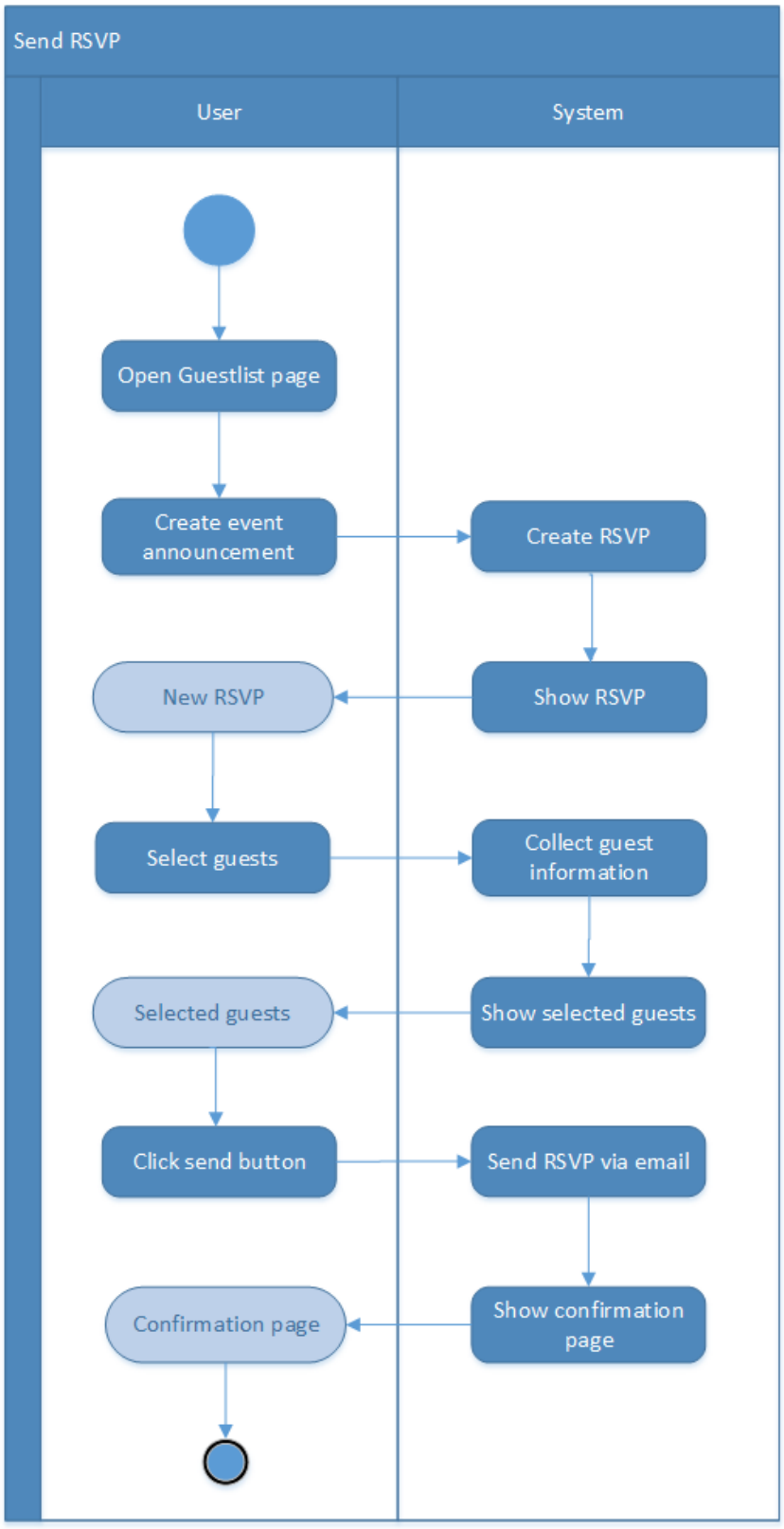

Figure 11: Activity Diagram - Send RSVP

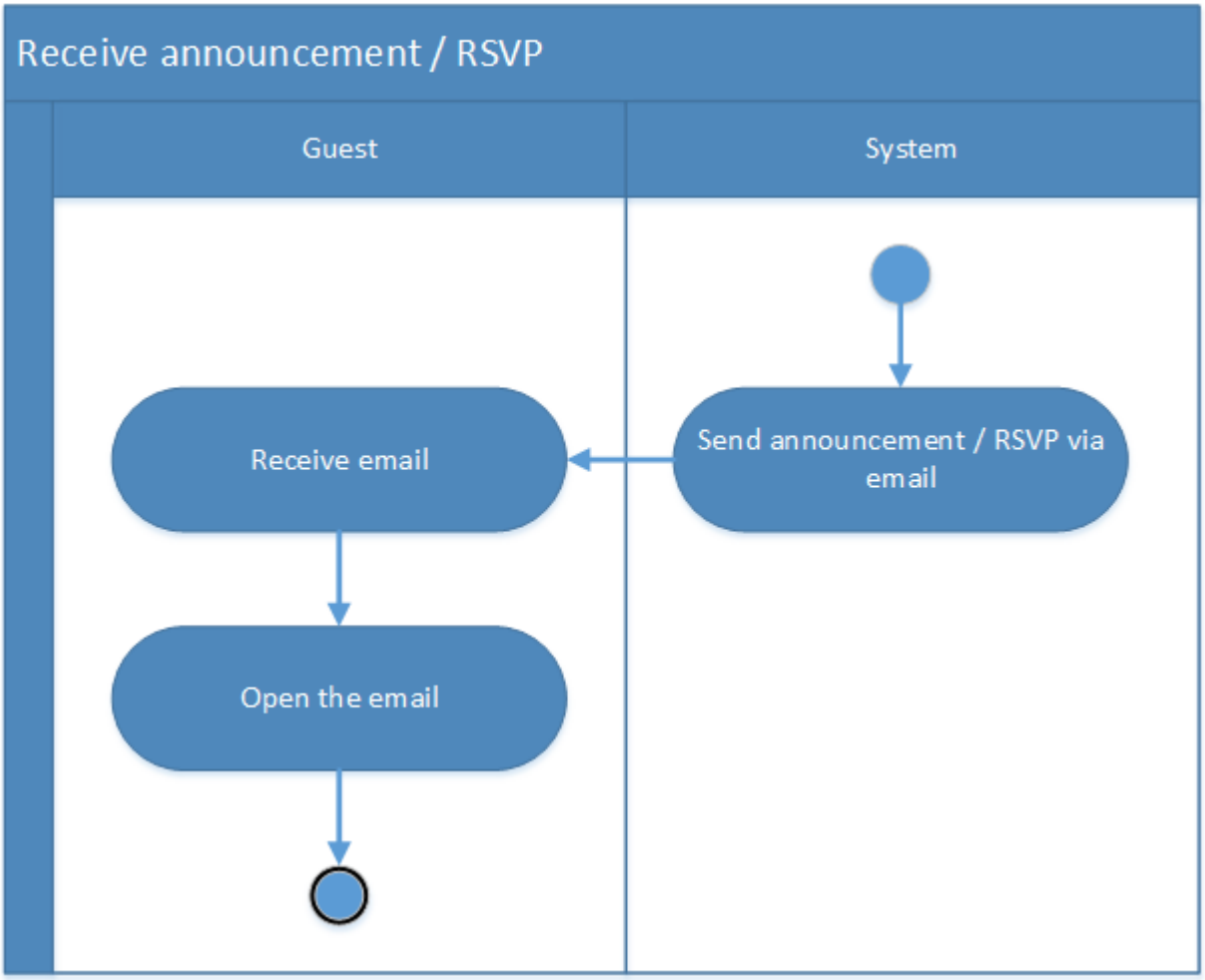

Figure 12: Activity Diagram - Receive Announcement / RSVP

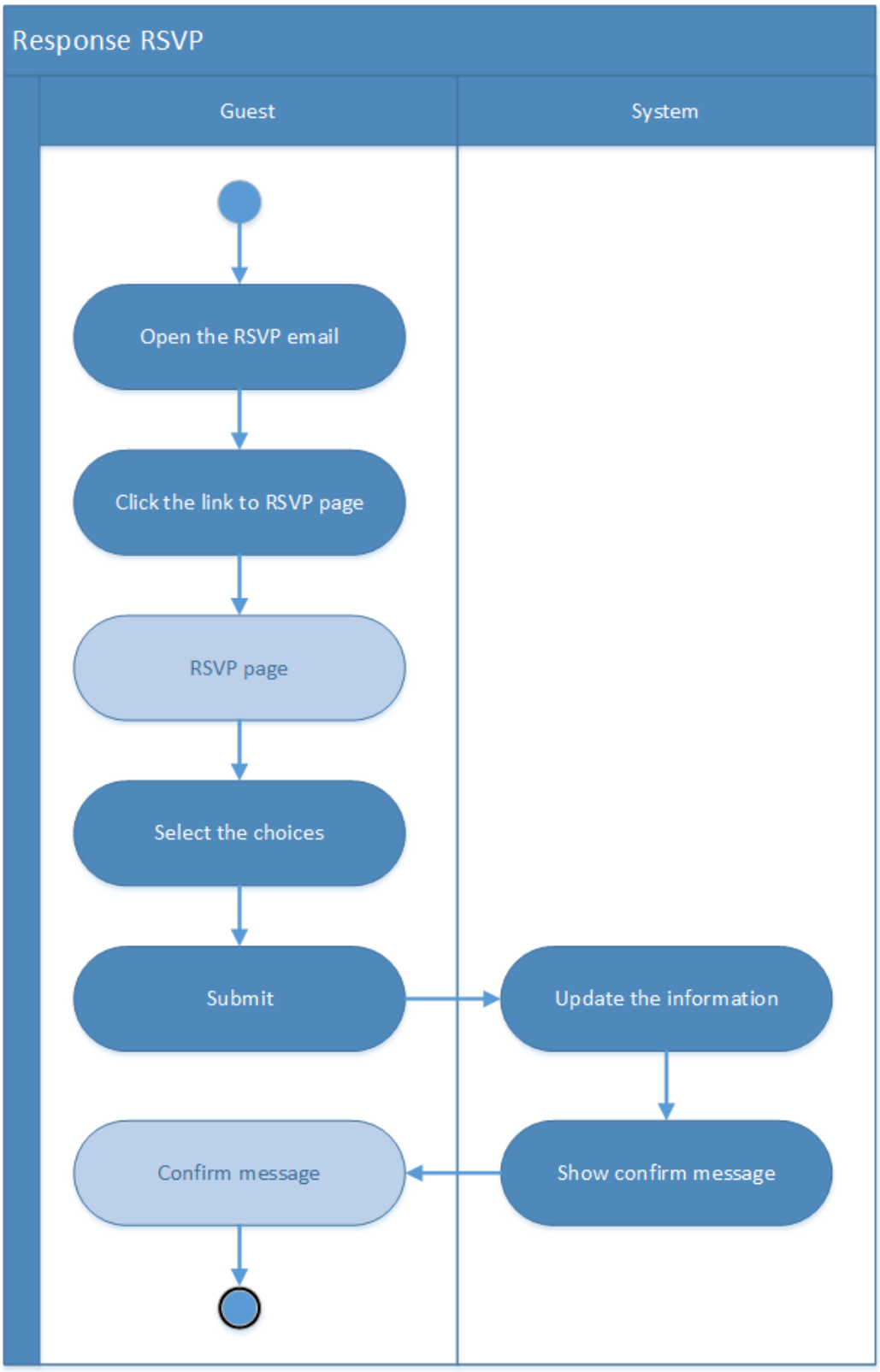

Figure 13: Activity Diagram - Response RSVP

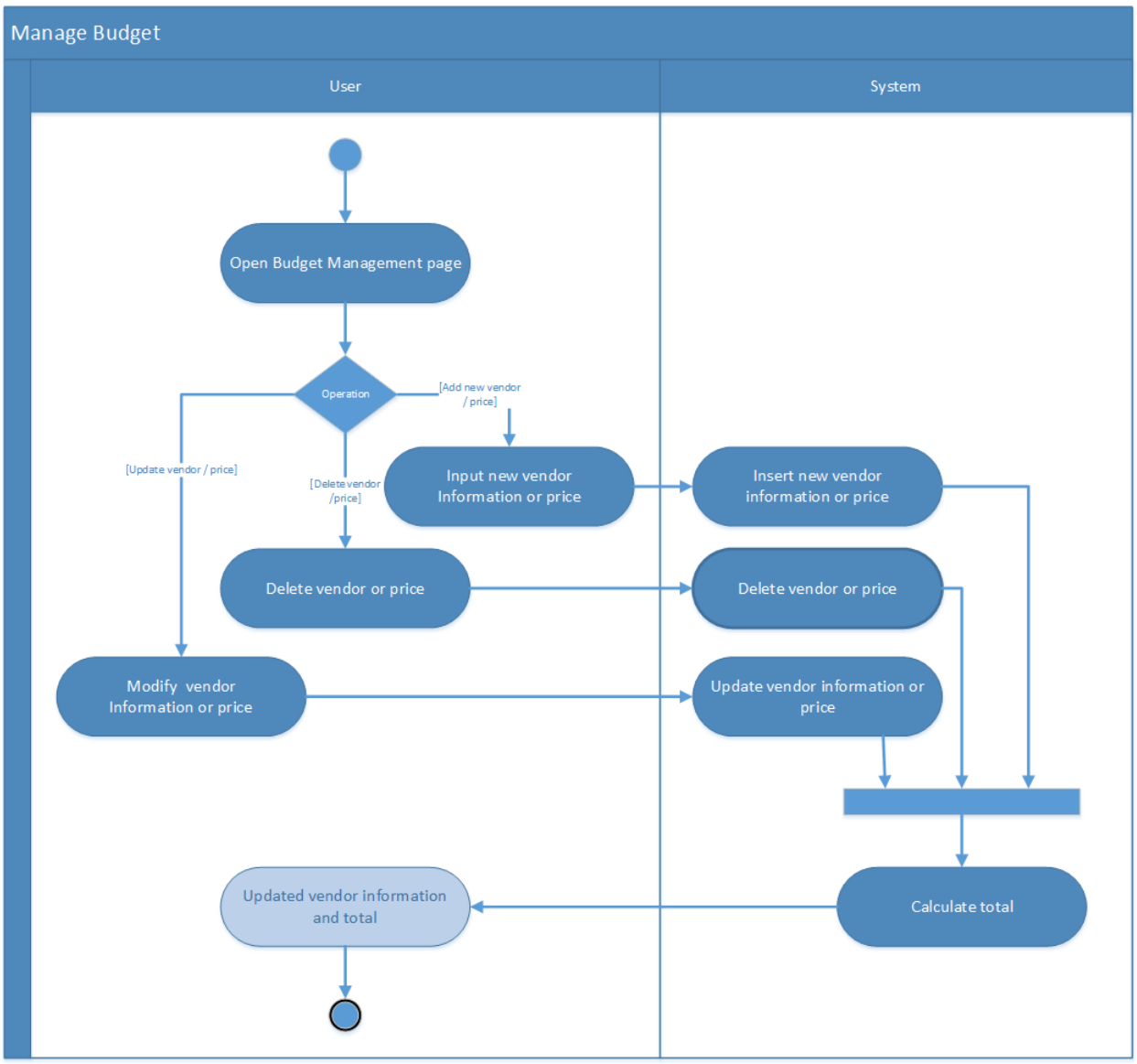

Figure 14: Activity Diagram - Manage Budget

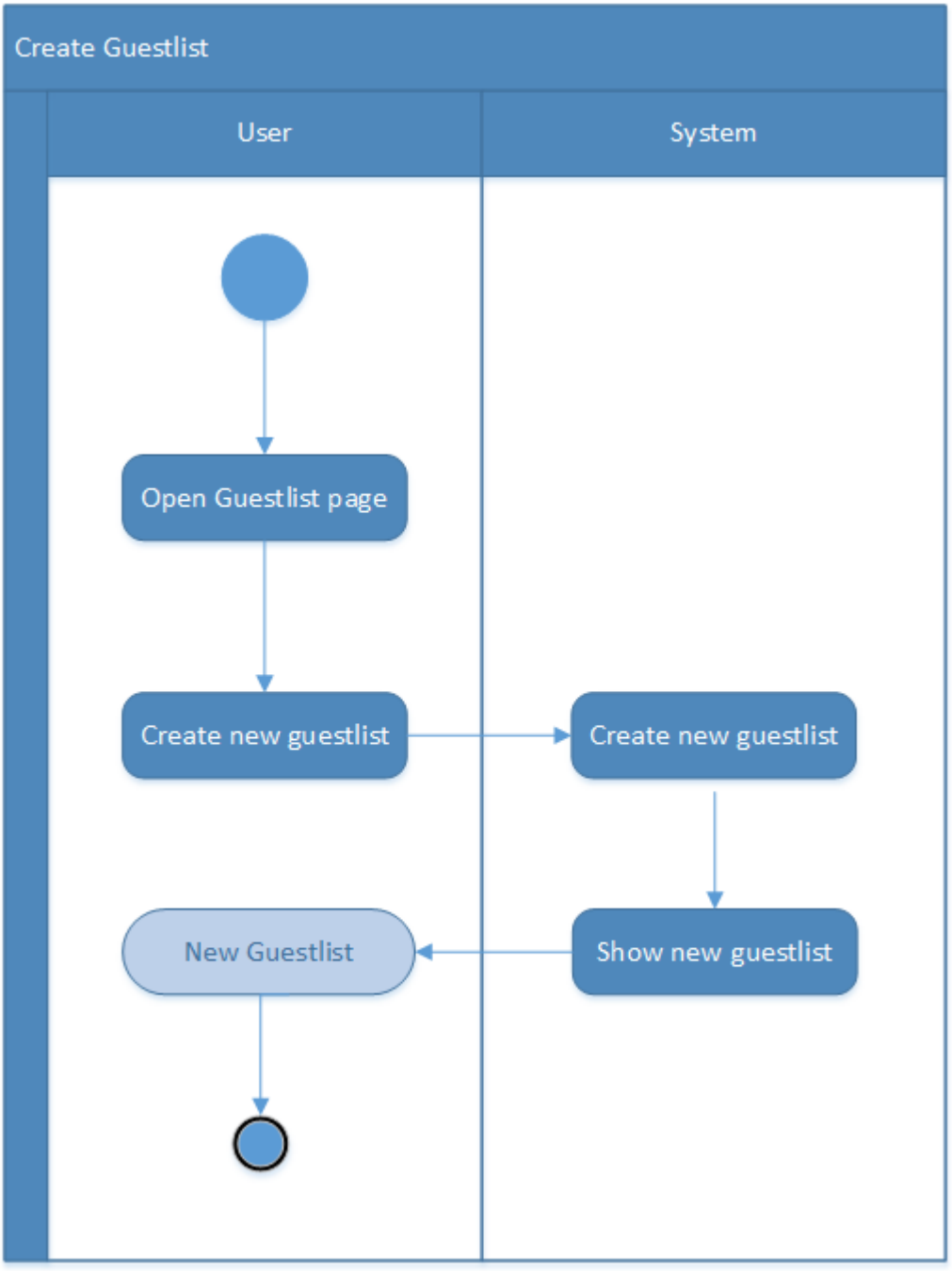

Figure 15: Activity Diagram - Create Guestlist

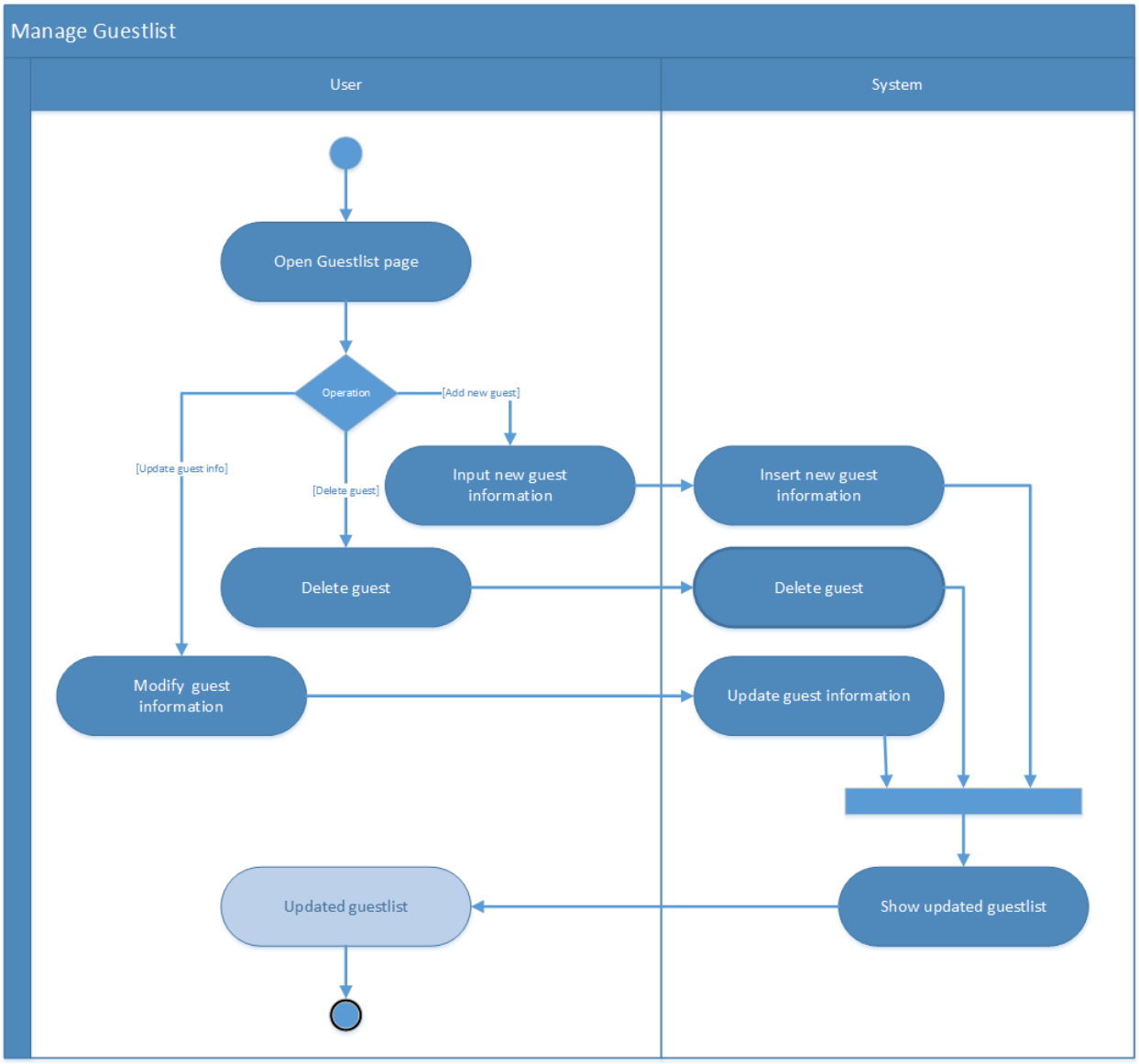

Figure 16: Activity Diagram - Manage Guestlist

3.3.3 Sequence Diagrams

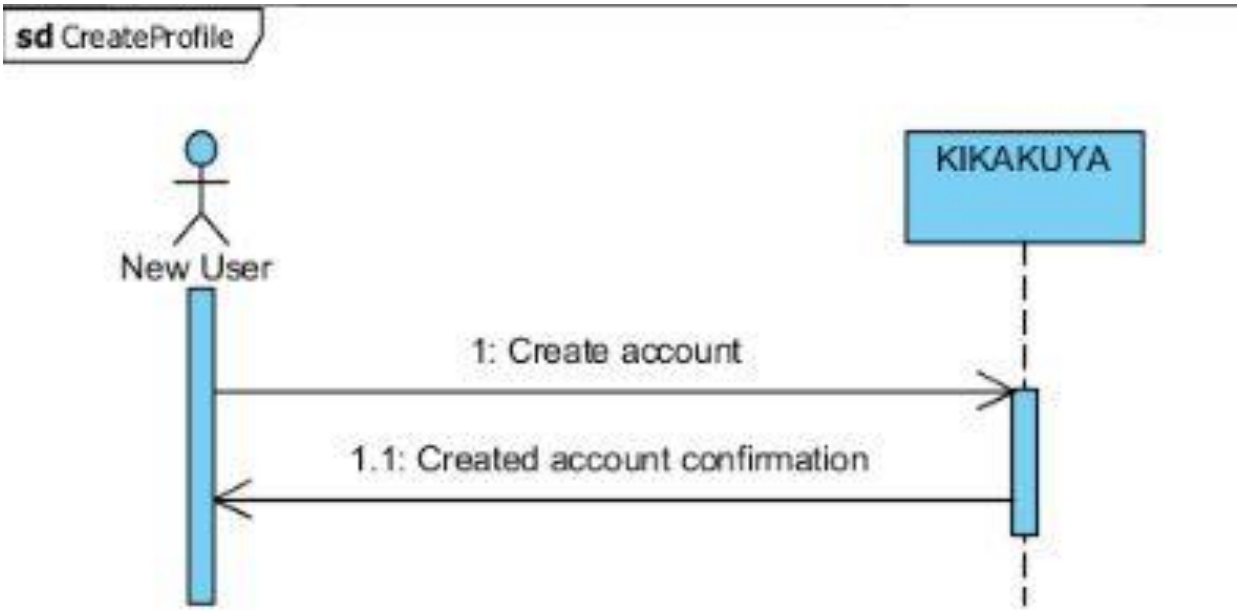

Figure 17: Sequence Diagram - Create Profile

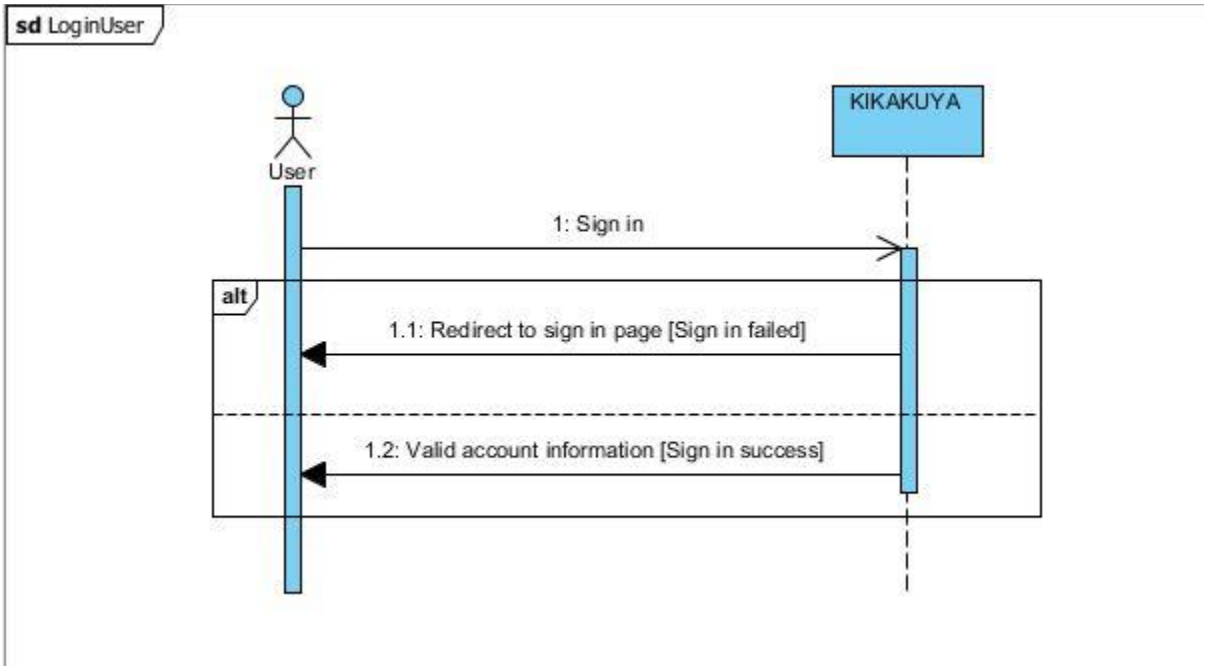

Figure 18: Sequence Diagram - Login User

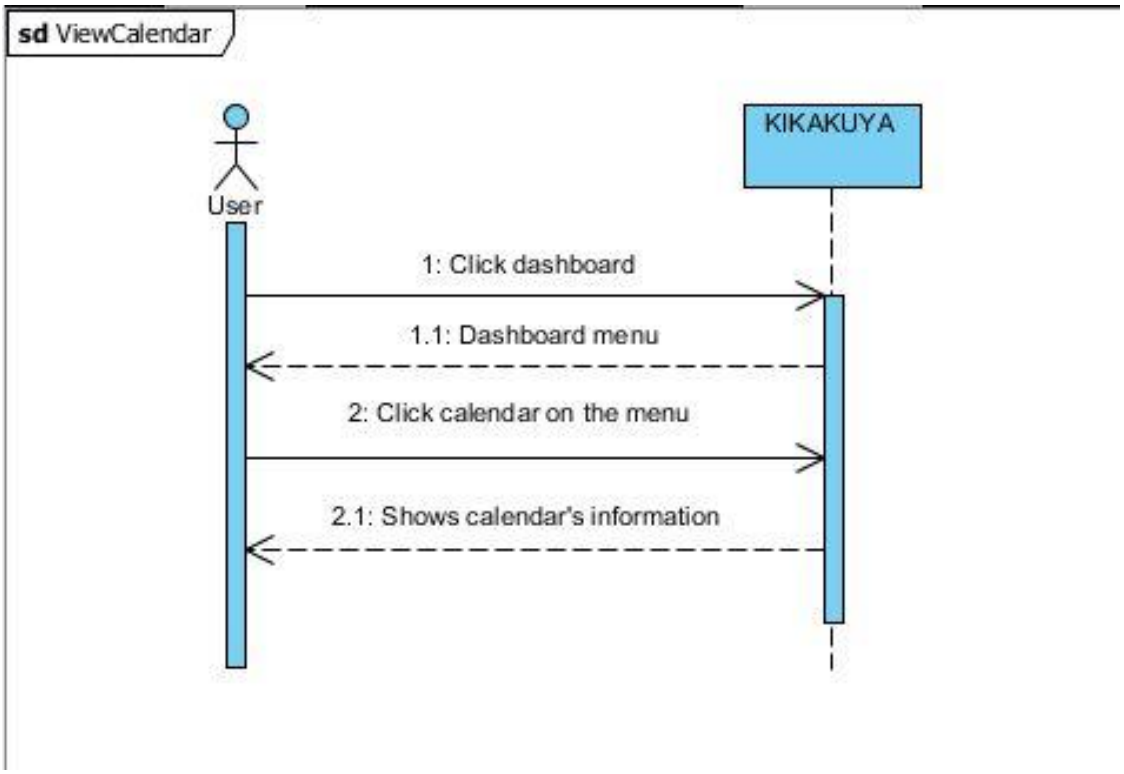

Figure 19: Sequence Diagram - View Calendar

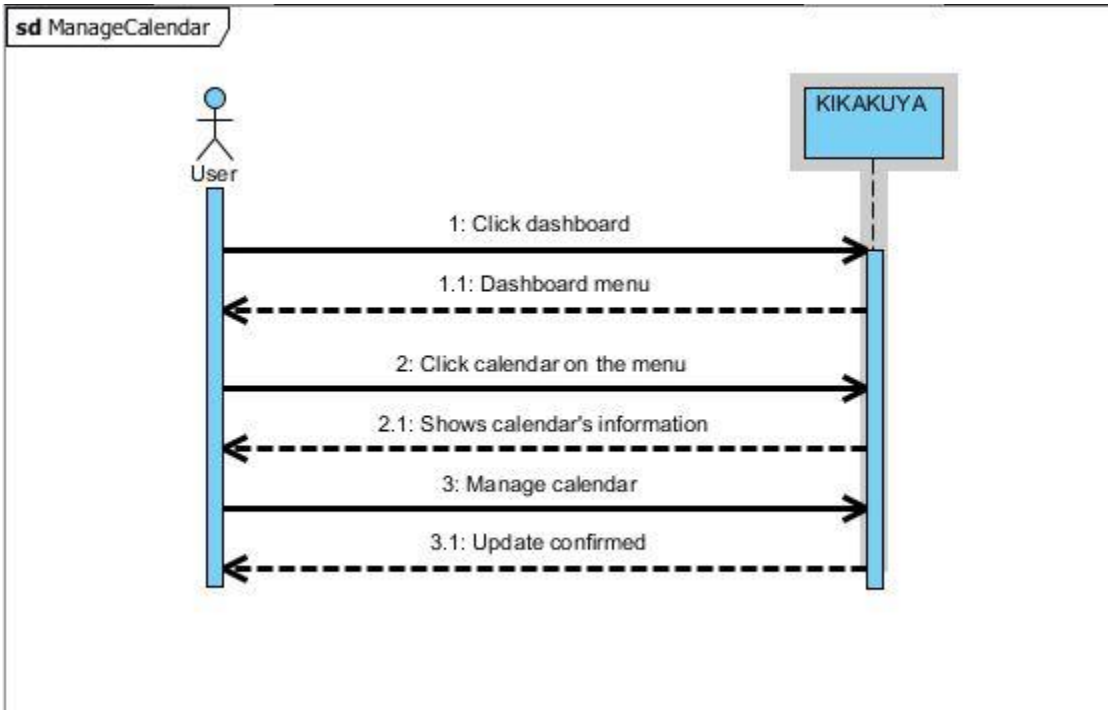

Figure 20: Sequence Diagram - Manage Calendar

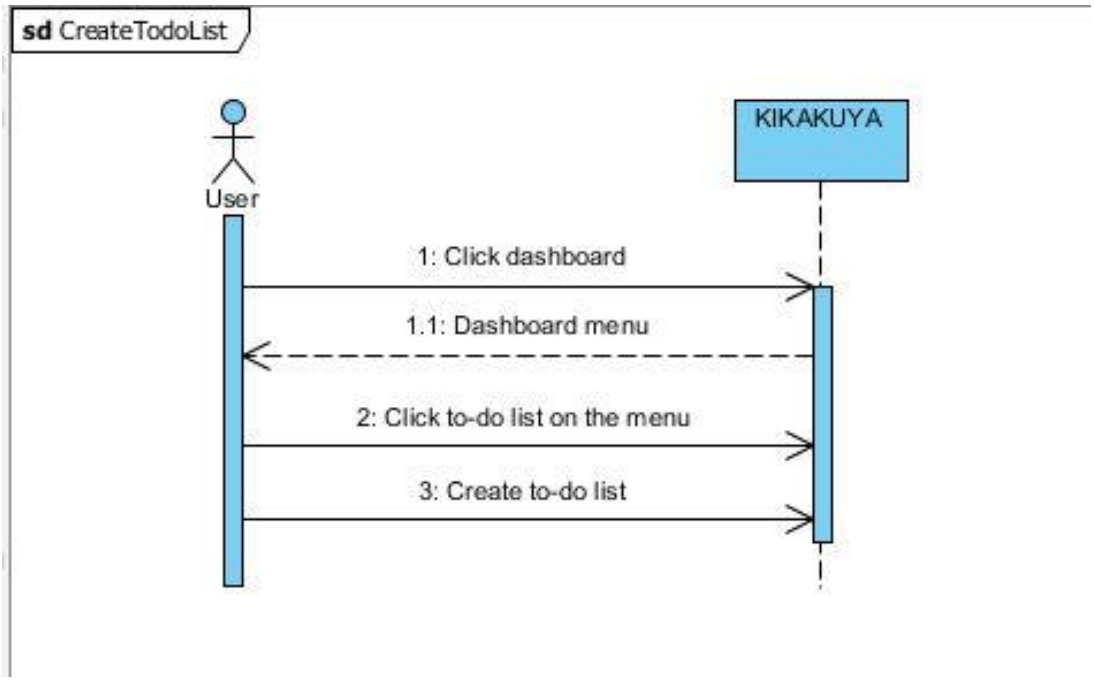

Figure 21: Sequence Diagram - Create To-do List

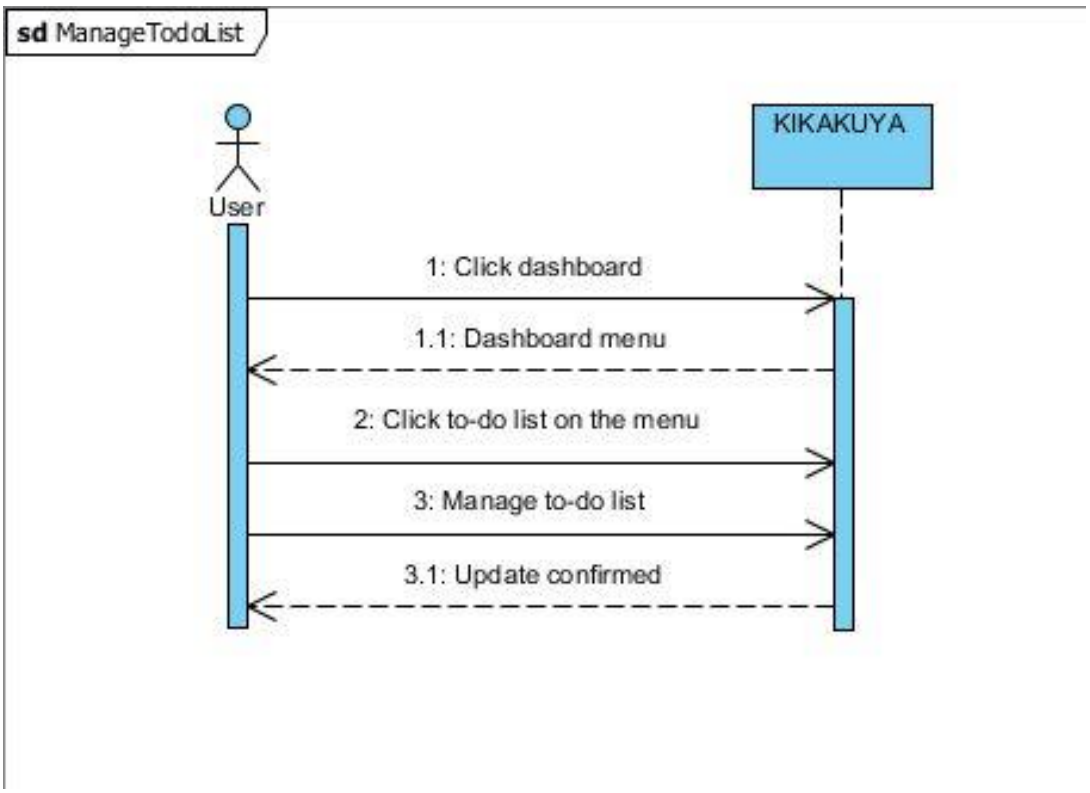

Figure 22: Sequence Diagram - Manage To-do LIst

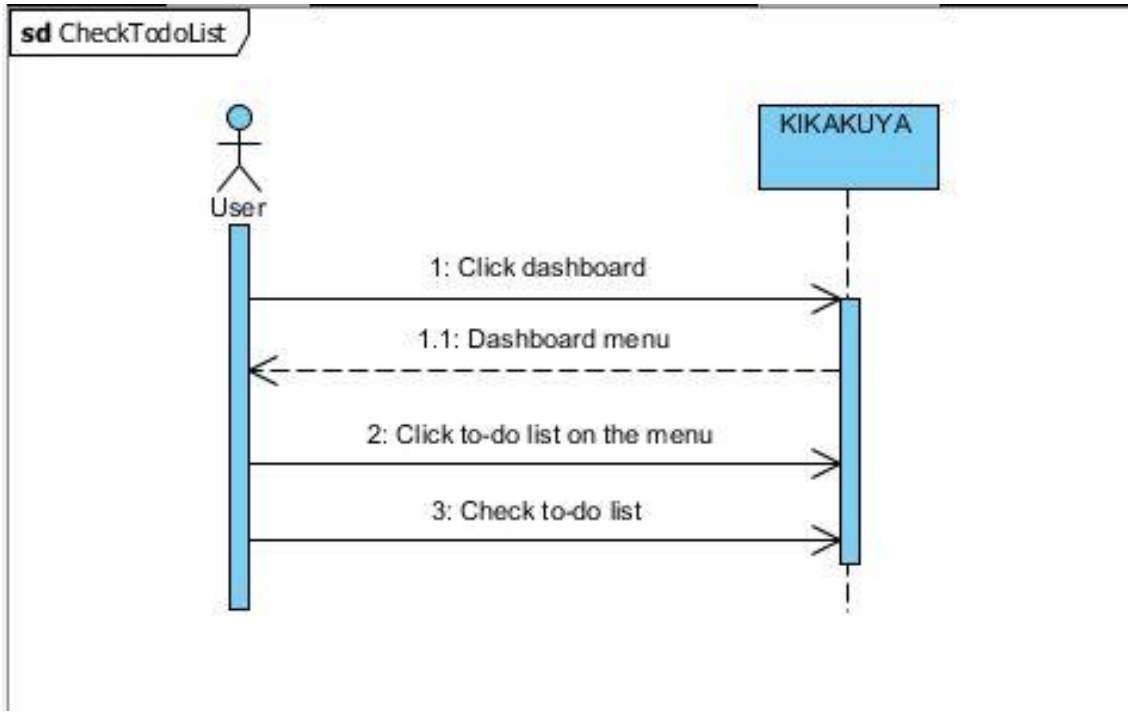

Figure 23: Sequence Diagram - Check To-do List

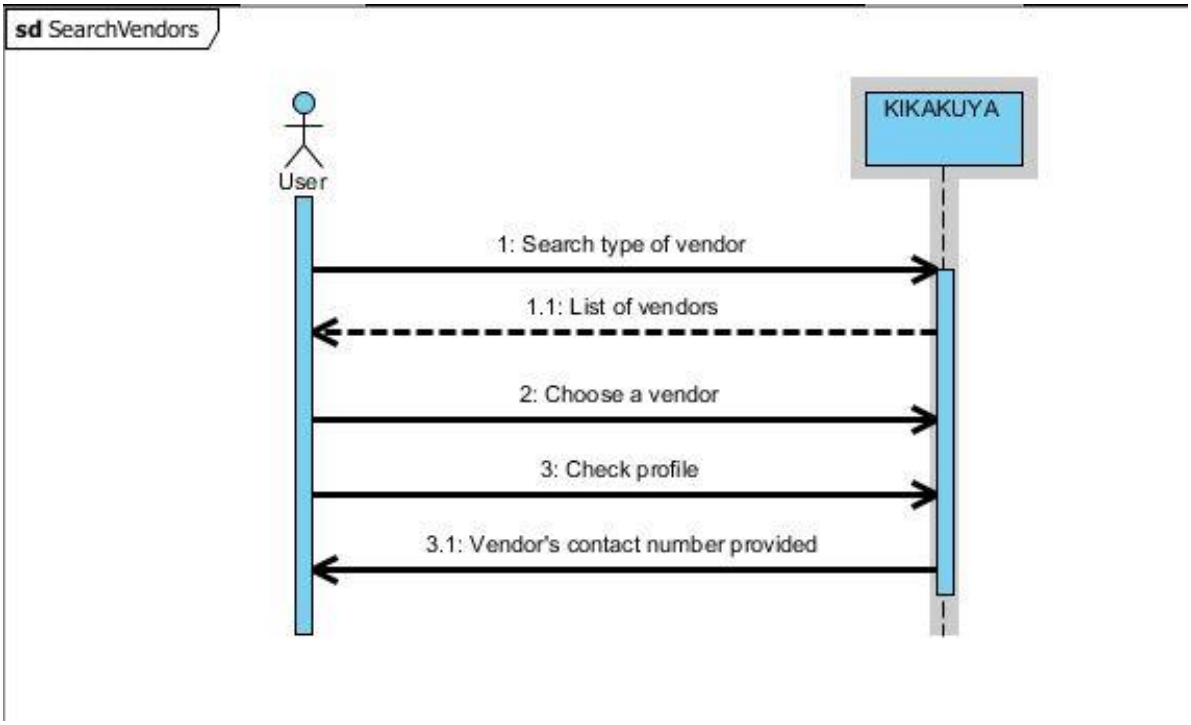

Figure 24:Sequence Diagram - Search Vendors

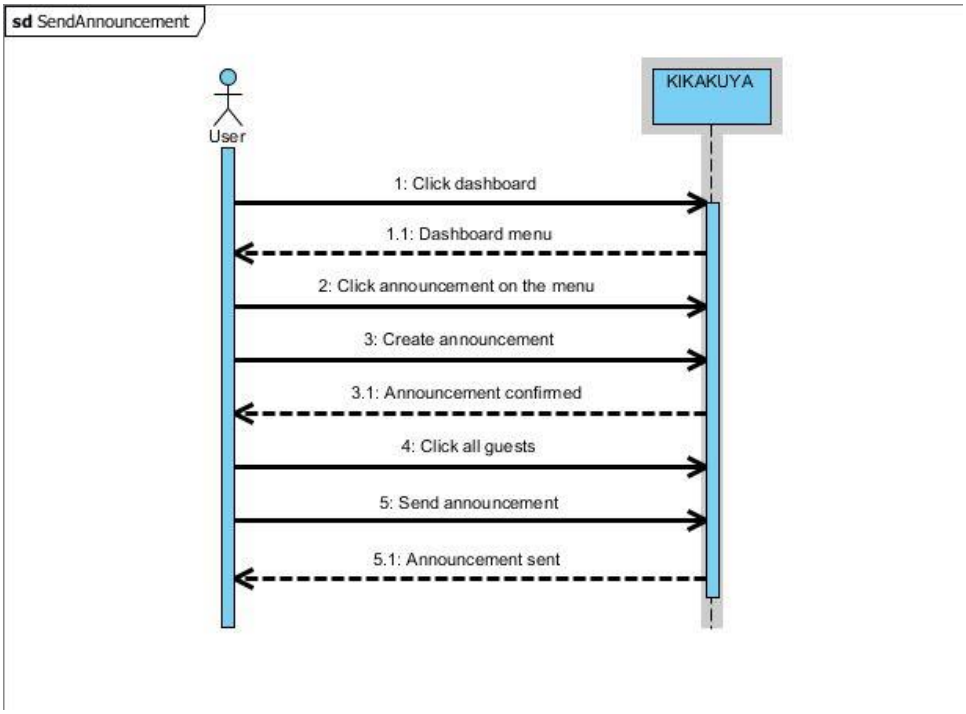

Figure 25: Sequence Diagram - Send Announcement

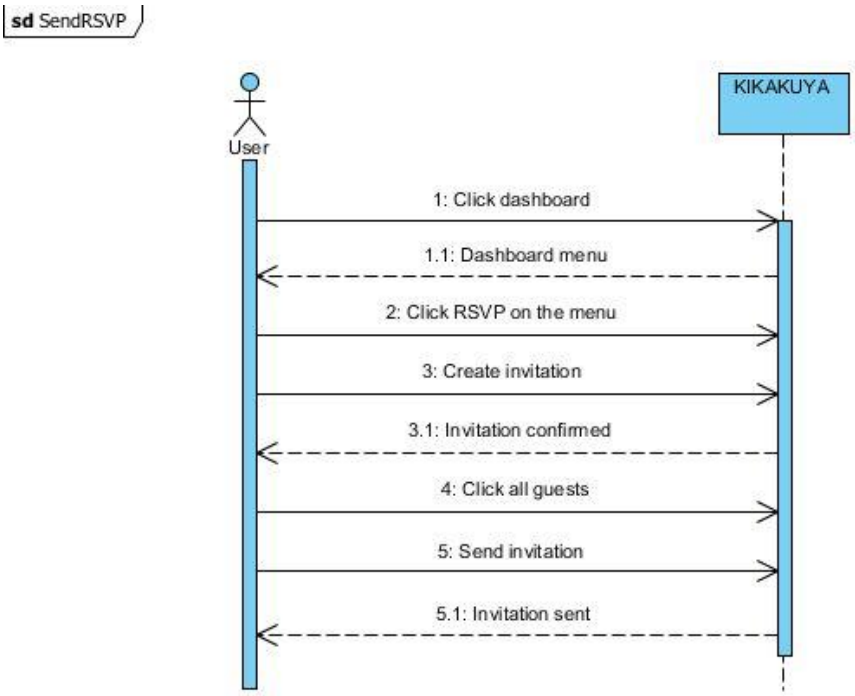

Figure 26: Sequence Diagram - Send RSVP

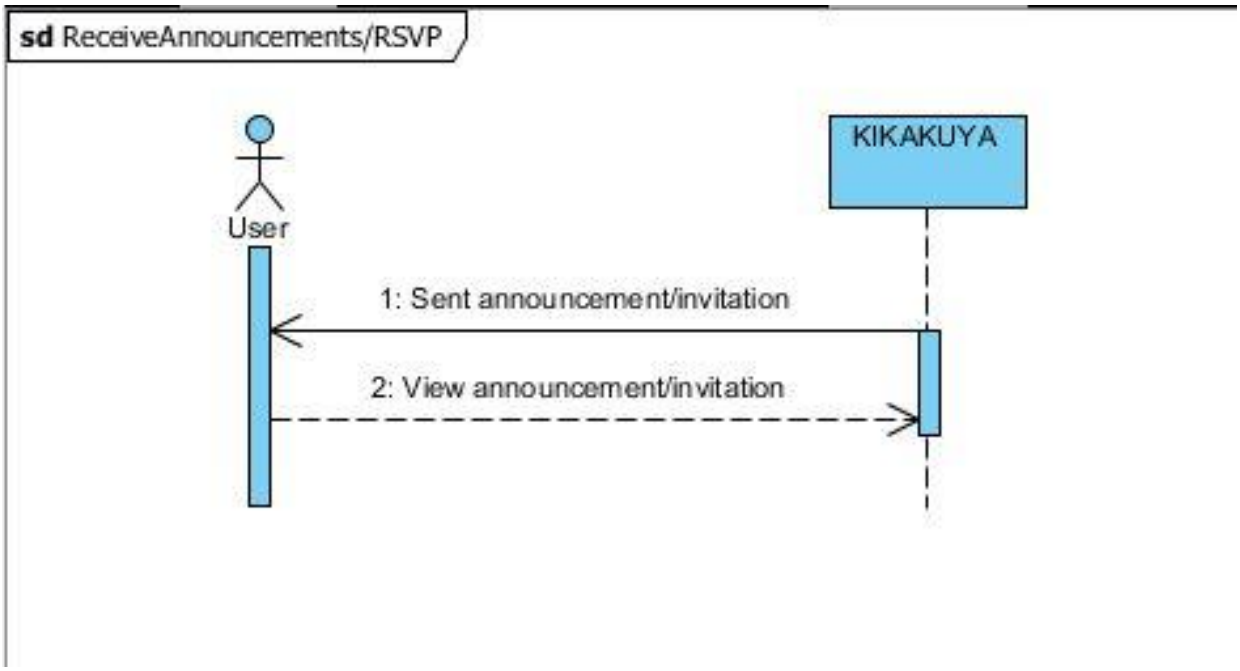

Figure 27: Sequence Diagram - Receive Announcements / RSVP

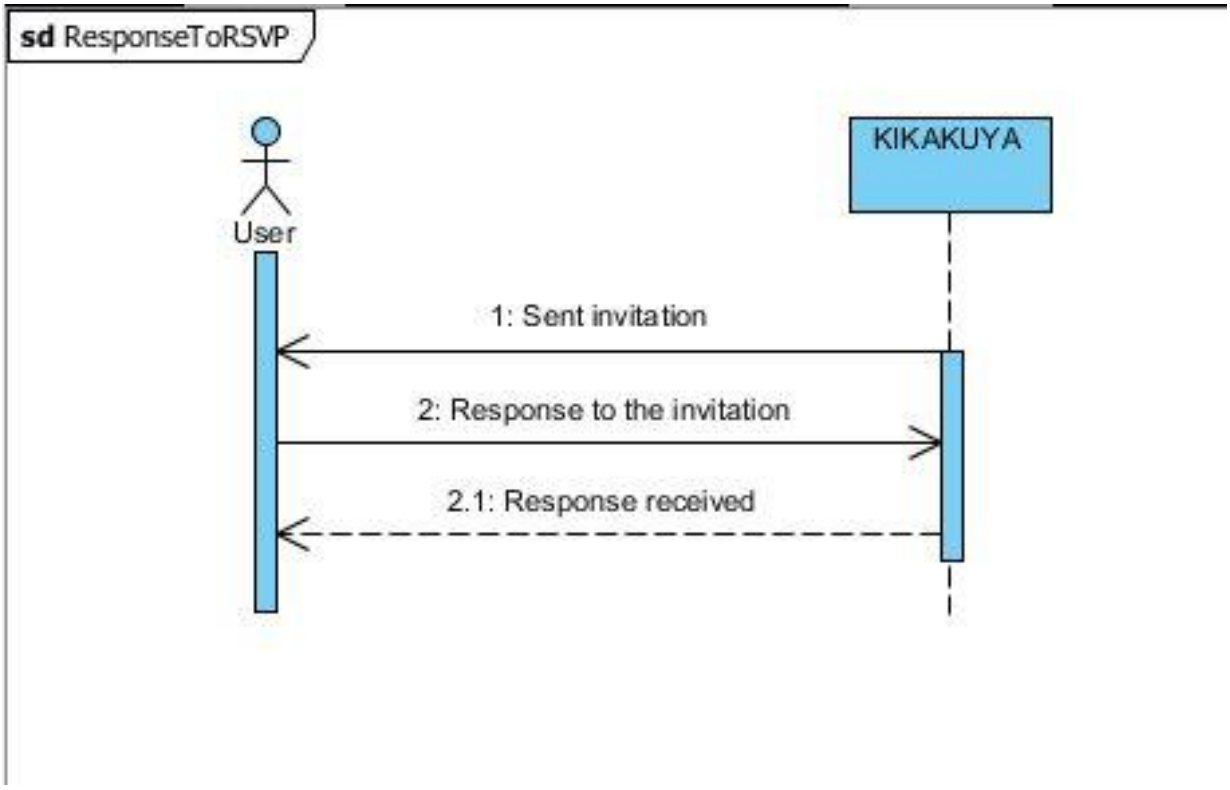

Figure 28: Sequence Diagram – Response RSVP

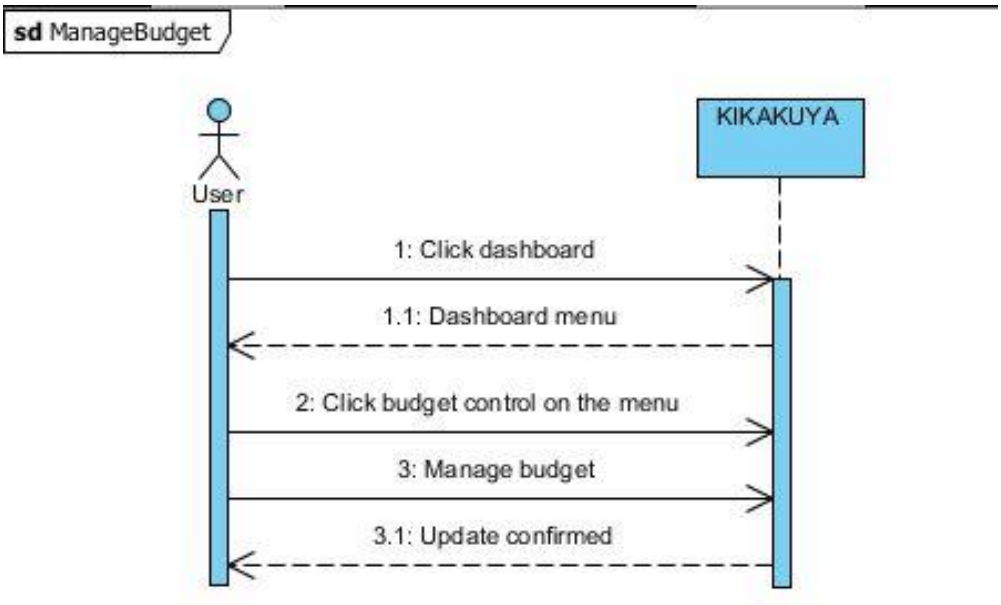

Figure 29: Sequence Diagram - Manage Budget

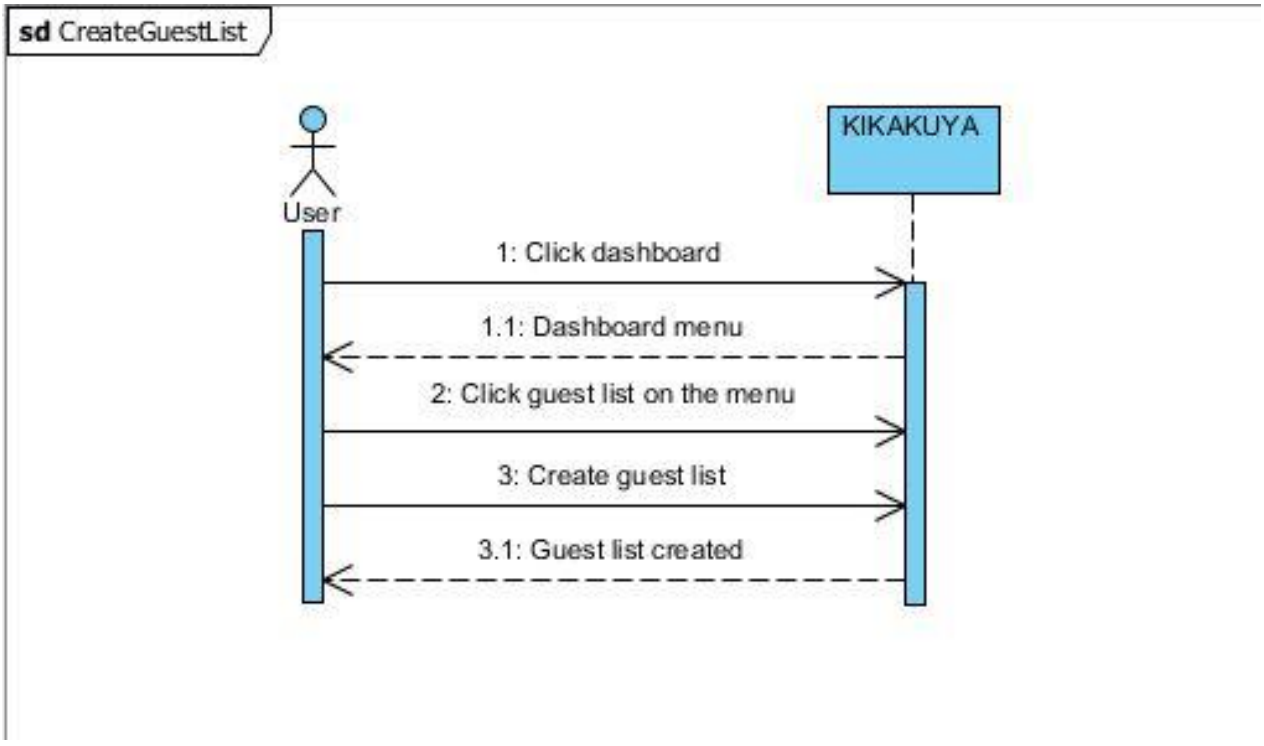

Figure 30: Sequence Diagram - Create Guest List

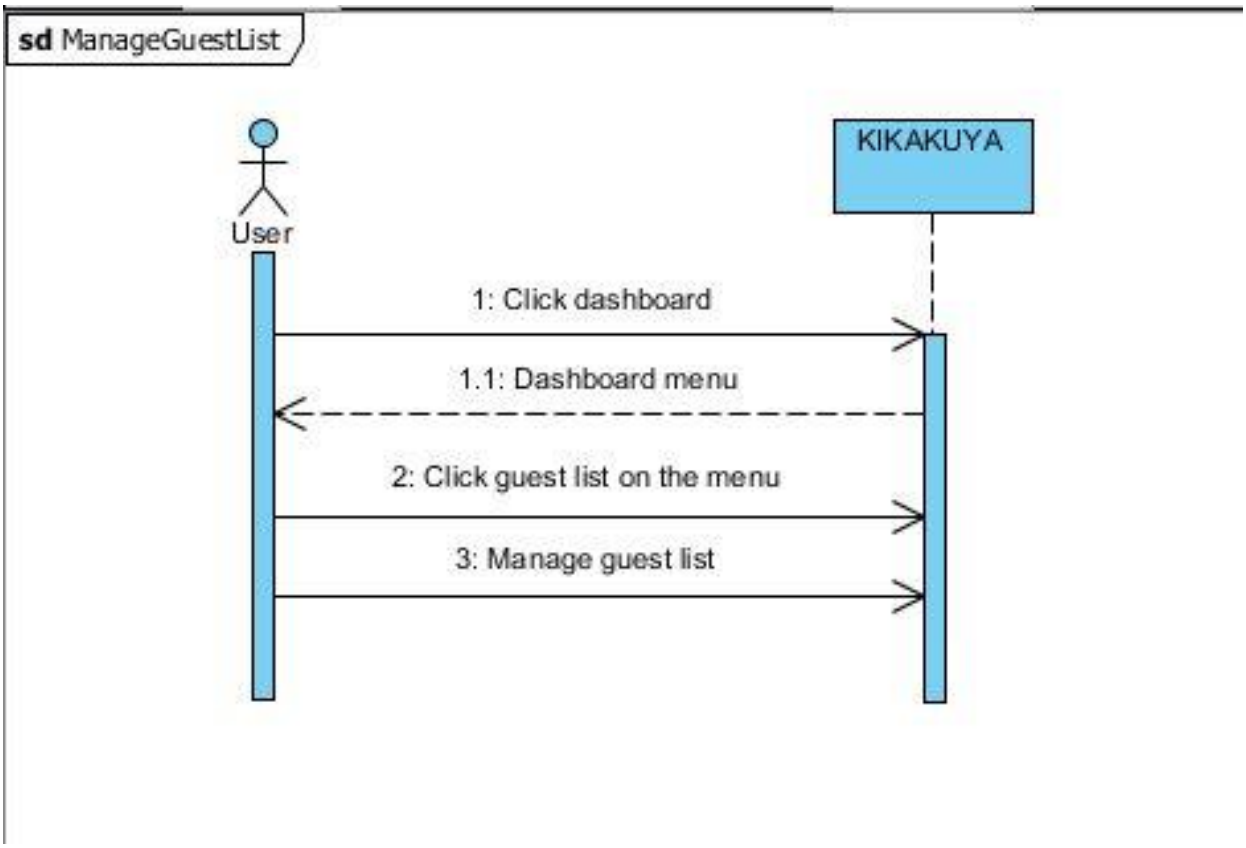

Figure 31: Sequence Diagram - Manage Guest List

3.3.4 UML Class Diagrams

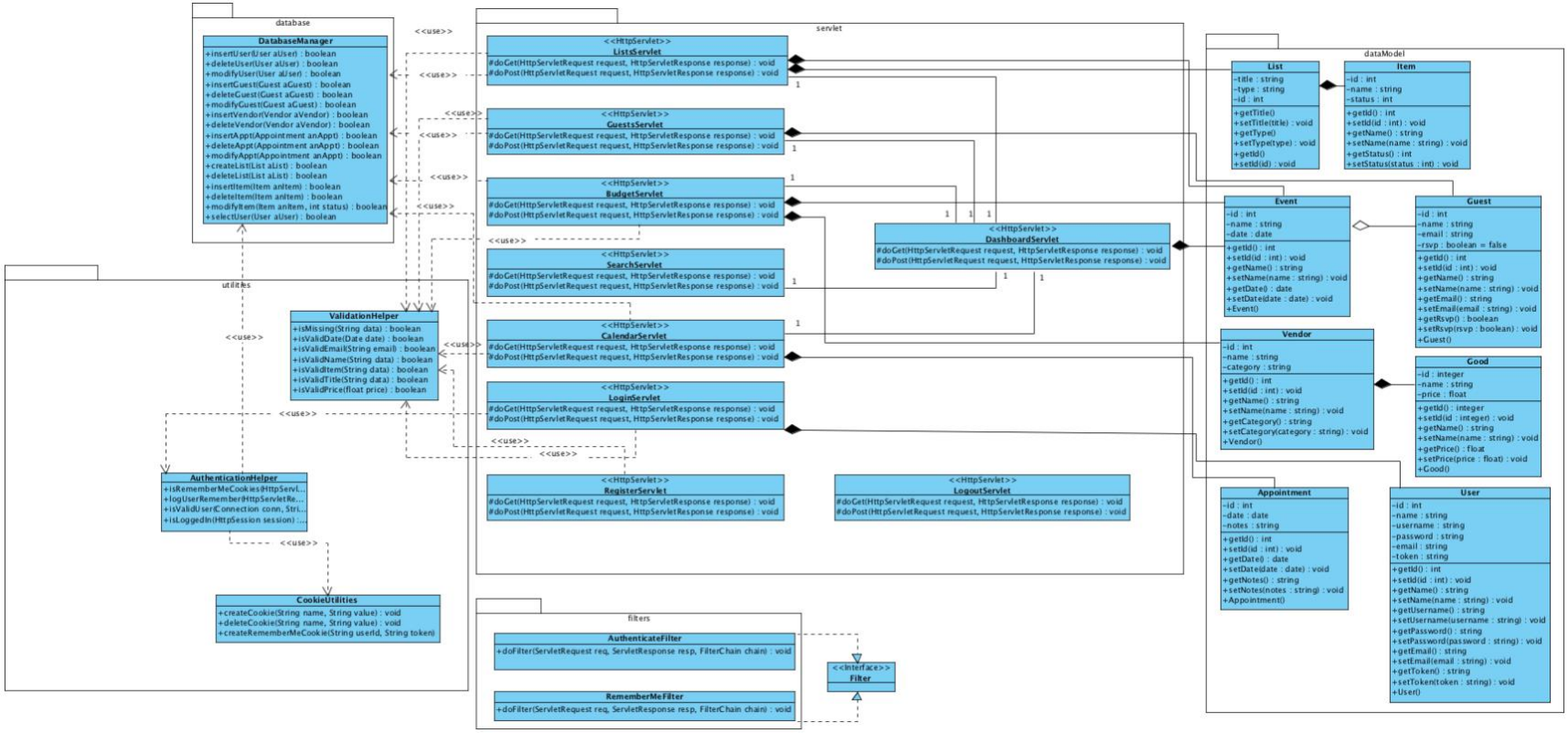

Figure 32: UML Class Diagram

3.4 Process Modelling

3.4.1 Data Flow Diagram

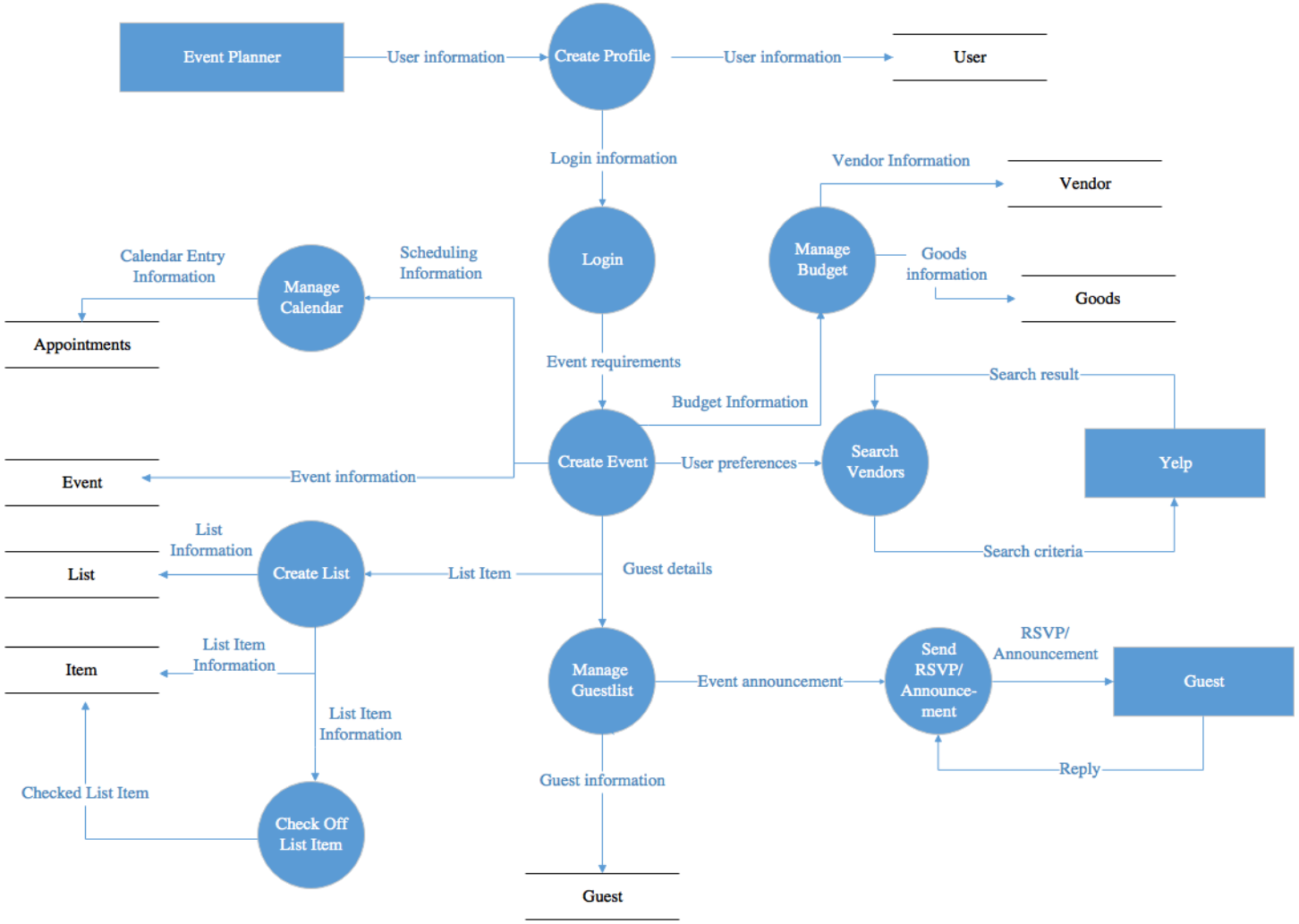

Figure 33: Data Flow Diagram

4. NON-FUNCTIONAL REQUIREMENTS

NFR1 The system shall comply with https, by having a trusted SSL Certificate.

NFR2 The guests shall receive announcement emails, at most, 5 minutes after the event planner sends them.

NFR3 The system shall be available 24 hours a day, 7 days a week, except when maintenance is needed.

NFR4 The system shall meet or exceed 99% uptime.

NFR5 The system shall store errors raised in log files.

NFR6 The system shall have a housekeeping mechanism to delete old log files.

NFR7 The system shall support 30,000 active users (figure valid up to December 2018).

NFR8 The system shall hold a maximum of 500 guests per event.

NFR9 The system shall hold a maximum of 3 events per user.

NFR10 The interface of the system shall be responsive.

5. LOGICAL DATABASE REQUIREMENTS

Our project will be utilizing a SQL relational modelled database. The following are the logical requirements for its implementation:

1. Based on the event details inputted by the user, we will be utilizing character, numeric, and date data formats.
2. Event details will be stored until the user decides to delete the information.
3. Primary and foreign key constraints will be implemented. The table below contains the primary and foreign keys of each table:

Table Name	Primary Key	Foreign Key
User	userId	
Event	eventId	userId
Guest	guestId	
Guest_Event	guestId, eventId	guestId, eventId
Vendor	vendorId	eventId
Good	goodId	vendorId
List	listId	eventId
Item	itemId	listId
Appointment	apptId	userId

6. APPROVAL

We approve the project as described above, and authorize the team to proceed.

Project Role	Name	Signature	Date
Back-End Web Developer	Aline N. Alencar	Aline N. Alencar	Nov 3, 2017
Front-End Web Developer	Kie Ogiya	Kie Ogiya	Nov 3, 2017
Back-End Web Developer	M. Alyssa Villacete	M. Alyssa Villacete	Nov 3, 2017
Database Administrator	Princess Ilasin	Princess Ilasin	Nov 3, 2017
Project Consultant	Anjana Shah	Anjana Shah	Nov 3, 2017