SOEN343
SOFTWARE ARCHITECTURE AND DESIGN

Community Service App
(Phase 2)

Team C: ArchiTECH

Instructor:
Dr. Joumana Dargham

1. Imran Ahmed (40172931) ~ Leader
2. Daniel Soldera (40168674)
3. Joe El-Khoury (40173108)
4. Titouan Sablé (40179062)
5. Ali Alp Erdinc (40172910)
6. Julien Fadel (40002473)
7. Kunal Shah (40153500)
8. Rohan Das (40177213)
9. Arsany Fahmy (40157267)

Date: 11/16/2022

Table of Contents

1. Summary of the project

The system to be developed will be a community service application that will target teenagers and young adults. The purpose of this application is to get the younger generation of our society to get to know their community and improve their lifestyle, but also help them get work experience early on in their career. The system developed will be the solution to solve this issue. In more details, it will be a full-stack application using an Object Oriented Programming language in the backend. The system will contain 4 types of users : User, Employer, Organiser and Admins. Users will be able to sign up for accounts and be able to participate in activities, go to social networking events. They will also apply for volunteering or job opportunities, which will be posted by the Employers. The application will also contain a request centre section, where users can get notifications in the form of newsletter and or reminders, and post comments in the forum. Admin handles any request made by employers and users and in general, manages the system. Additional features of the system are discussed in more details in the use cases and sequence diagrams found in the report.

2. System architecture

Table 1 : Description of the UI layer of our system

UI Layer	**Description**
Description	The User Interface of the system will be a Graphical User Interface using HTML, CSS, JavaScript and additonal frameworks. The UI will be composed of a homepage (containing navbar and footer), login and registration pages, account profile, a forum page, donations page and a dedicated page for each of the four services that the system offers. (Request centre section, Volunteering and humanitarian section, Activities section, and Social network section).
When used	The UI is executed and displayed at all times during the execution of the system
Advantages	Most Software developers, including members of our team are familiar with HTML, CSS and JavaScript. Thus, the implementation of the user interface will not be very difficult.
Disadvantages	HTML code gets very large and hard to understand with complex web design. This makes it very hard to maintain the code and debug it.

Above is a table that describes the UI layer used for our system, including the advantages and disadvantages of the UI used.

Table 2: Description of the application layer of the system

Application layer	**Description**
Description	All data belonging to the system and their opeartions will be fetched from the backend, which is composed of Java Spring and the Database (SQL). The data will be fetched through an API call in the front end. Session states of the system can be acquired using Java Servlets. The 'HttpSessionInterface' in Java Servlets allow the get information about session [1]. The page transitions of the system will be dependent on CSS and any frameworks that enhance the trasition, such as Bootstrap. The transformation of data will all be done in the backend in the Java REST API. Each of the API endpoints in the REST API (which will be called through an action from the user in the front-end) will be responsible for doing any modifications and transformation of the data.
When used	The application layer will be used throughout the entire execution of the system and will play a much important role when the user initiates a request to the backend.
Advantages	Java Spring is very used in the industry and has many advantages than other languages as it uses design patterns such as dependency injection, its performance is very good, fast and uses POJO (Plain Old Java Object), making it easier for developers to code [2]
Disadvantages	Java Spring is very complex and requires a lot of practice to be comfortable. Even if most members of our team are familiar with Java, there is a possibility that Java Spring can be complex/difficult to understand for some of the members of the team as not many of us were exposed to such frameworks before.

Above is a table that describes the Application layer used for our system, including the advantages and disadvantages of the Application layer used.

Table 3 : Description of the Foundations and domain objects

Foundation and domain objects	**Description**
Description	The Database (SQL) that will be used for the system will be located in this layer. The Database will include tables such as user/account information, tables for activity information, tables containing all the jobs offered and their information, etc. All of the operations done on these tables will be done in this layer, such as READ, WRITE, DELETE, UPDATE of the data. The system will be compatible on most modern Operating Systems such as Windows, Mac OS, Linux, IOS and Android.
When used	At all times during the execution of the system
Advantages	SQL is very simple and easy to use. The learning curve is not complex. All users have devices that run on Windows, Mac OS, Linux, IOS or Android, thus running the application will not be difficult for the general consumer. They are also very reliable Operating Systems and very secure.
Disadvantages	Some of the GUI for running SQL can be very complex to understand, making it hard for any new programmer who is trying to learn the language [3].

Above is a table that describes the Foundation layer used for our system, including the advantages and disadvantages of the Foundation layer used.

3. Use Cases Diagrams

The Community Platform has many functionalities for different types of users. These functionalities can be divided into 3 sections, Functionalities for the Request centre section, for the volunteering and humanitarian section, and Activities sections.

For the request centre section, the user is able to apply for memberships and is able to share comments, reviews and complaints for activities and events. In addition, another functionality is the lottery, which is held monthly where users who have donated upto 100$ can enter the lottery and win prizes. The administrators are responsible for performing the lottery and notifying the winners.

For the Volunteering and humanitarian section, the user is able to apply for a job/volunteering position and is able to make donations to the community service. As for the employers, they are able to post their jobs and volunteering positions into the system, if they are hiring.

For the activities seciton, the user is able to search and filter activities and events, is able to participate in events they are interested in, and is able to request reminders for them. The Organisers, who are people who organise the events, are able to post events into the system.

In the following page, use cases and additional details for each of these functionalities will be discussed.

1) Request centre section

For the Request centre section, the three uses cases below are considered

a) System for applying for letter news and membership [ID: 1]

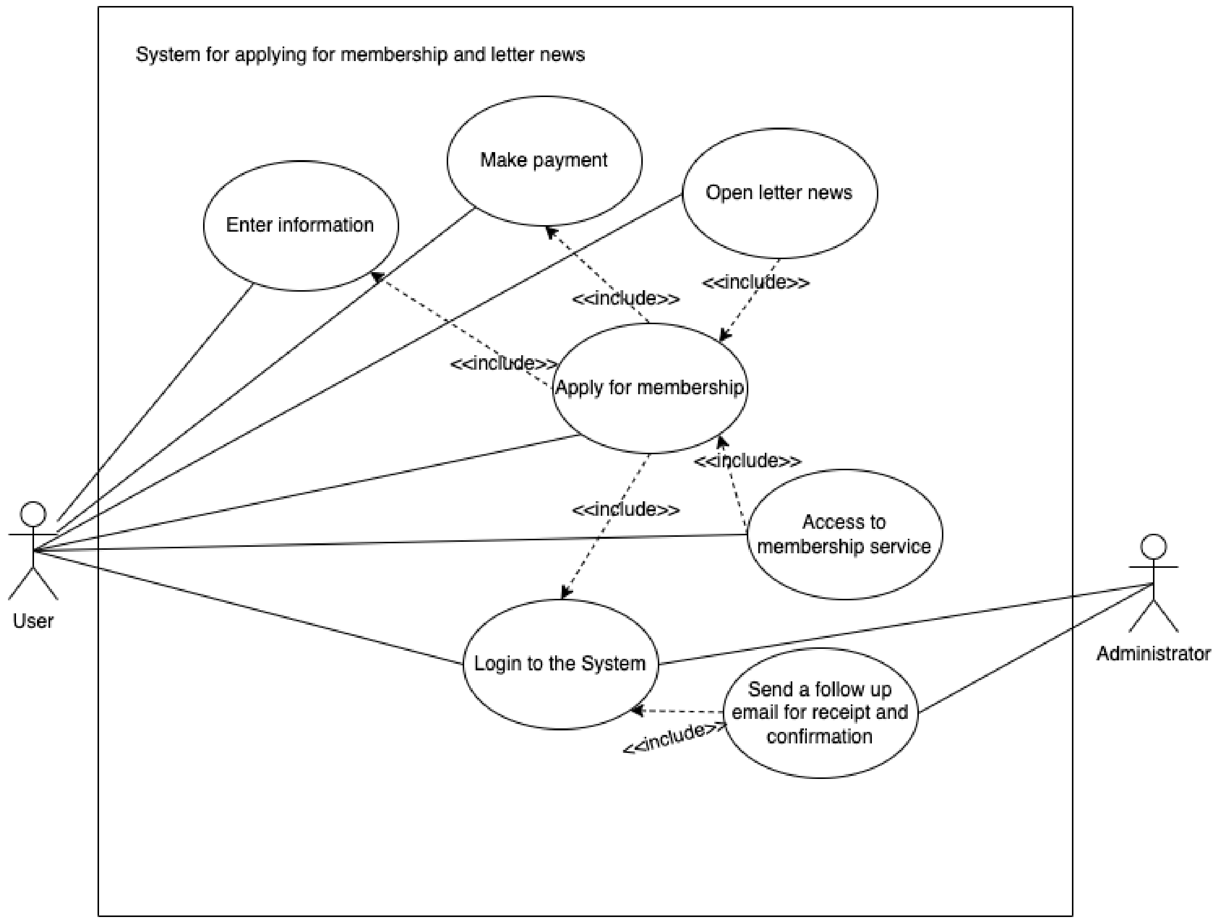

Figure 1: Use case diagram of User applying for membership and letter news

The user is the primary actor in this diagram. The user is able to login to the system and apply for membership. After applying for the membership, the user receives a confirmation email sent by the administrator. The user then has access to the letter news and to the services

b) System for sharing comments/reviews/complaints [ID: 2]

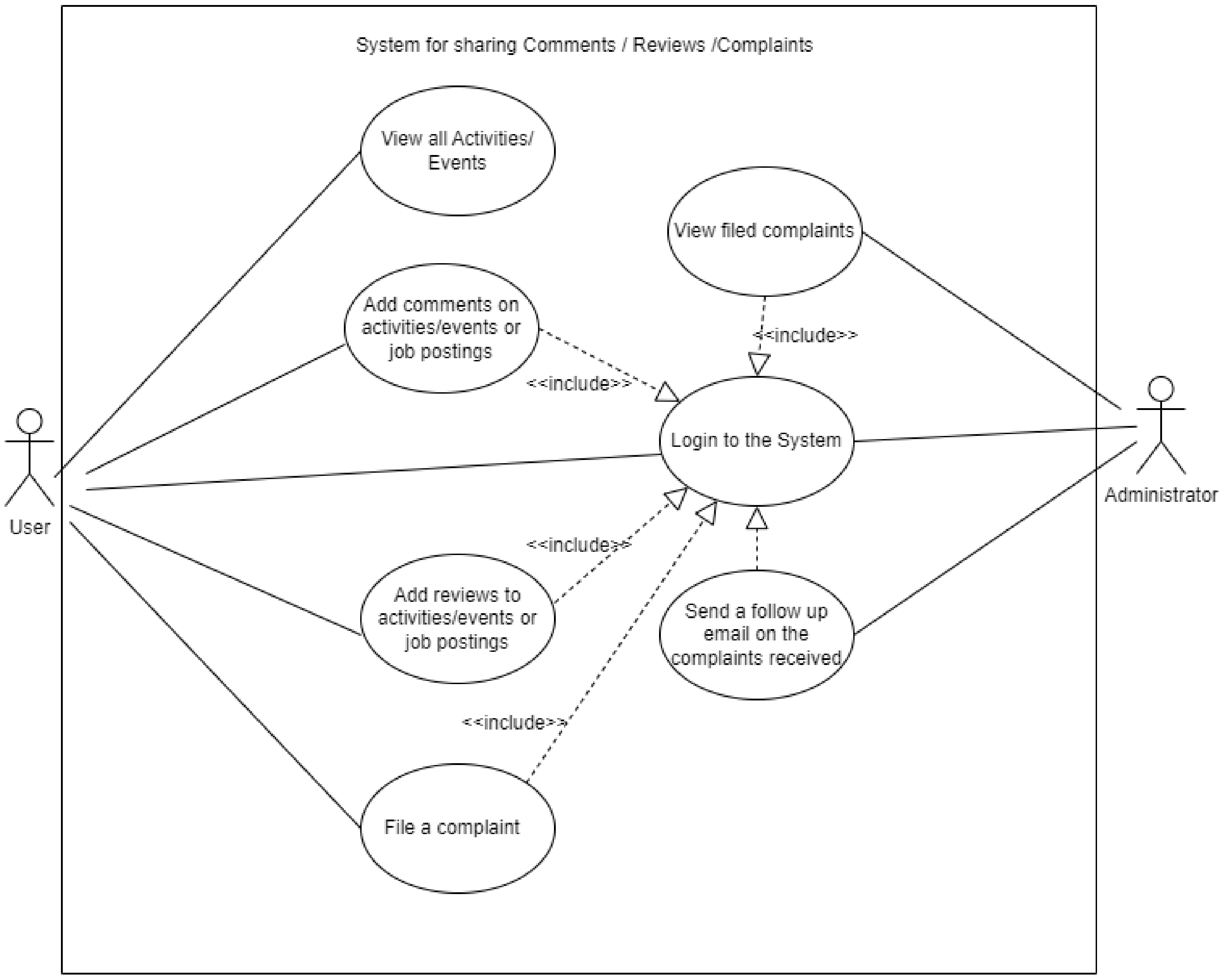
System for sharing Comments / Reviews /Complaints
View all Activities/ Events
View filed complaints
Add comments on activities/events or job postings
<<include>>
<<include>>
Login to the System
User
Administrator
<<include>>
Add reviews to activities/events or job postings
Send a follow up email on the complaints received
<<include>>
File a complaint

Figure 2 : The use case of the sharing comment/reviews/complaints feature

The primary actor for this system is the User. The user is able to login to the app to view all the activities and events, add comments, reviews and file a complaint .The secondary actor is the admin who can view the complaints received and send a follow up email to the users.

c) System for lottery and lottery results notifications [ID: 3]

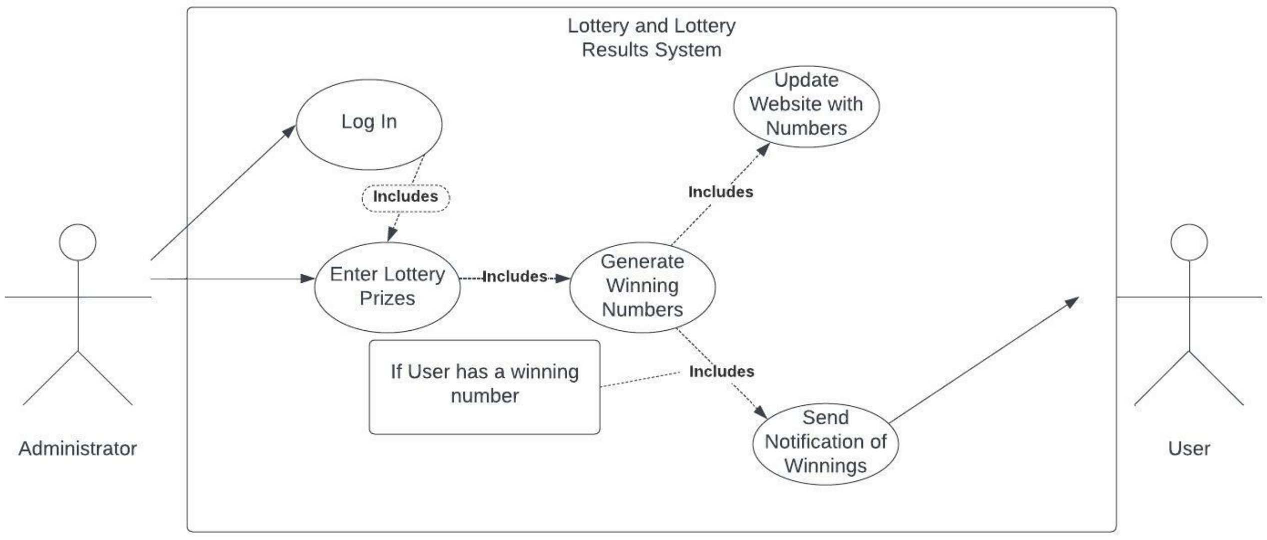
Lottery and Lottery Results System
Log In
Includes
Enter Lottery Prizes
Includes
Generate Winning Numbers
Includes
Update Website with Numbers
If User has a winning number
Includes
Send Notification of Winnings
Administrator
User

Figure 3 : The lottery and lottery results notifications feature

The Administrator is the primary actor of this system. After the lottery has been performed, the admin can enter the lottery prizes (and the quantity of each prize) into the website. Each user who enters the lottery will have a unique lottery ticket number. The system will randomly choose winning numbers from the list of people who are in the lottery. These numbers will determine who won the lottery. After it has been performed, It then updates the website with these new winning numbers and notifications are sent to every user who won the lottery (user who’s lottery ticket number correspond to a winning number)

2) Volunteering and humanitarian section

For the Volunteering and humanitarian section, the three uses cases below are considered

a) System for User to Apply to Job/Volunteering Position [ID: 4]

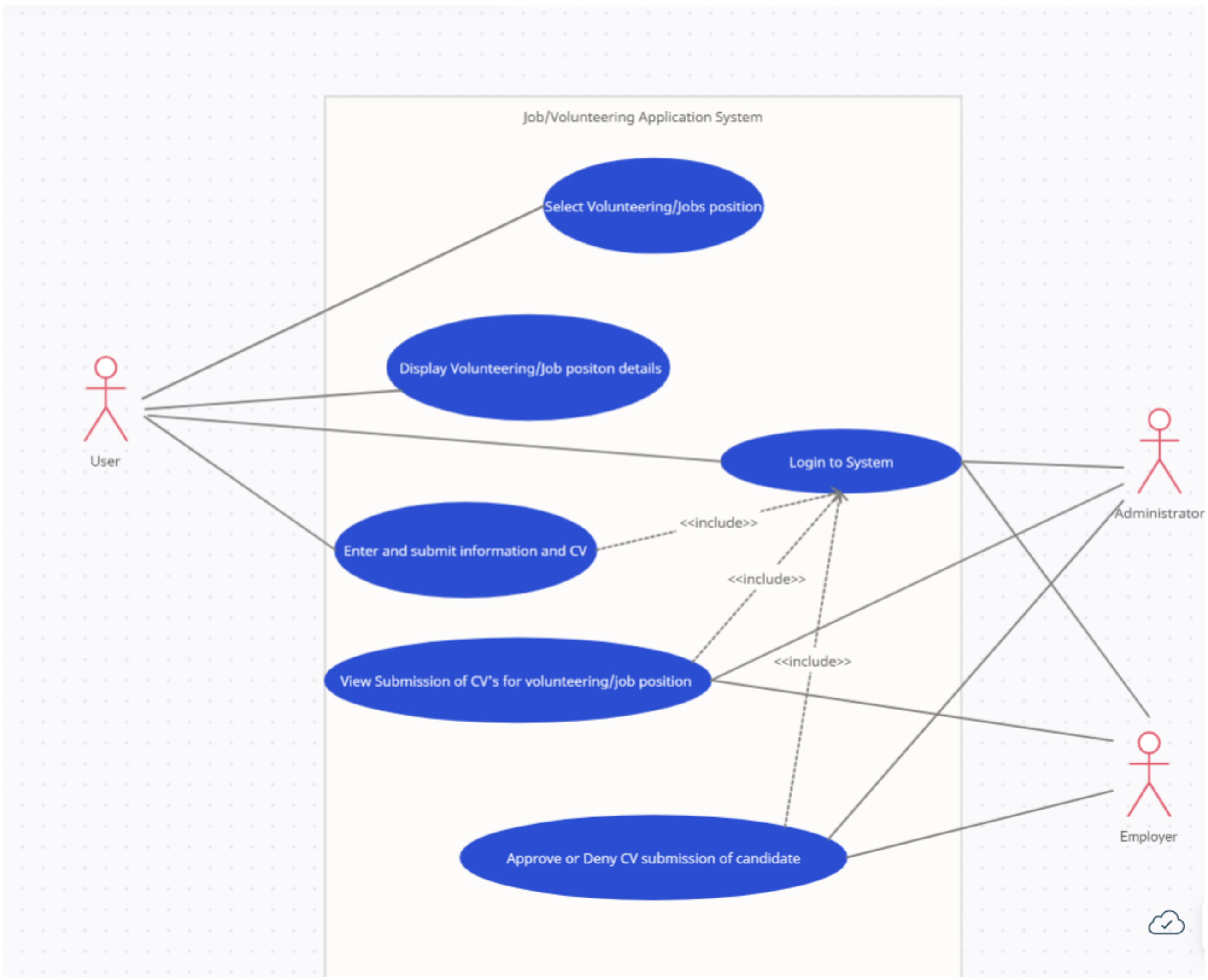
Job/Volunteering Application System
Select Volunteering/Jobs position
Display Volunteering/Job positon details
User
Login to System
Administrator
<<include>>
Enter and submit information and CV
<<include>>
<<include>>
View Submission of CV's for volunteering/job position
Employer
Approve or Deny CV submission of candidate

Figure 4 : The use case of the applying to a job/volunteer position feature

There the user is the primary actor of this use case. The user is able to Select the jobs, display more details about it and can submit their CV if interested.The employer is able to look at the submission of CV's of potential candidates and approve or deny any CV's submission. The same can be done for the administrator, If necessary.

b) System for Employers to Approve Job/Volunteering position [ID: 5]

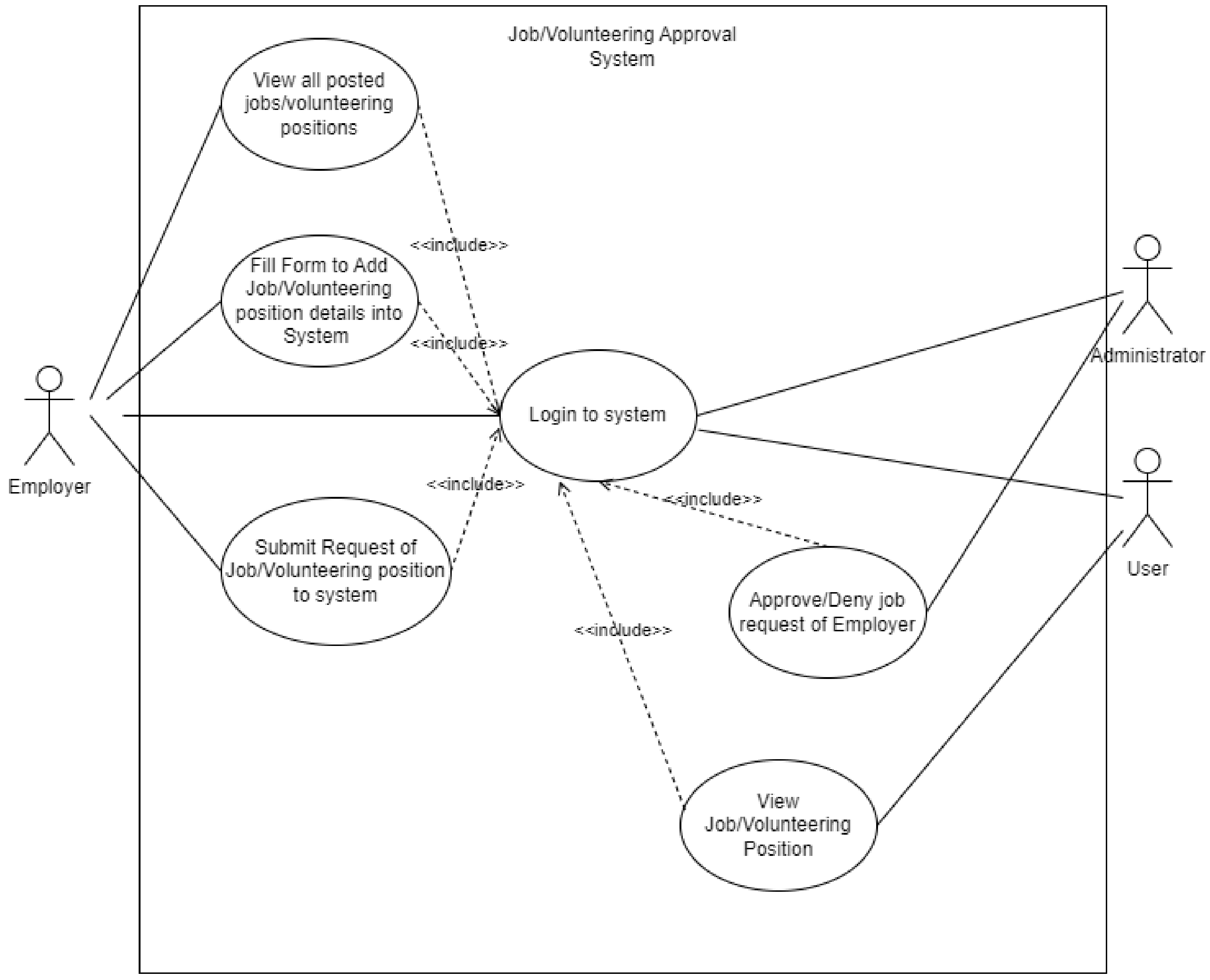
Job/Volunteering Approval System
View all posted jobs/volunteering positions
Fill Form to Add Job/Volunteering position details into System
Submit Request of Job/Volunteering position to system
Login to system
Approve/Deny job request of Employer
View Job/Volunteering Position
<<include>>
<<include>>
<<include>>
<<include>>
<<include>>
Employer
Administrator
User

Figure 5 : The use case diagram of Employer submitting a job/volunteering position into the system

The primary actor of this system is the employer. The employer is able to view the jobs he/she posted, add new jobs into the system, by adding all the necessary information, and submit a request. The secondary actor is the User and Admin. The user will be able to view the job posted by the employer and the admin is able to approve or deny requests of jobs from employers. In order for all the functionalities to execute, each of the users have to login.

c) Donations Transaction system [ID: 6]

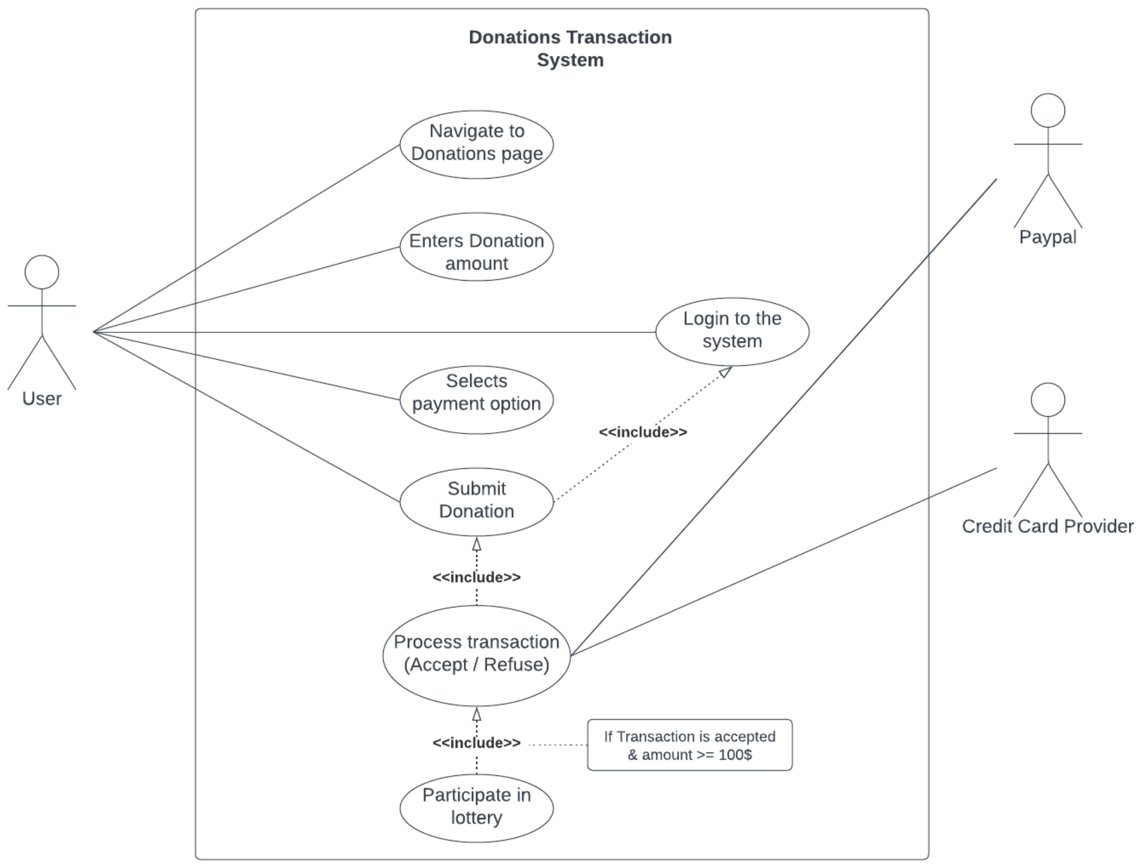

Figure 6 : The use case diagram of User submitting a donation into the system

The primary actor is the user. In order to submit a donation, the user needs to log into the system. The user can navigate to the donations page, enter a donation amount, select a donation option (credit card or PayPal) then submit the donation. The transaction is then processed by either PayPal API or the credit card provider. If the transaction is accepted, the user receives a confirmation message as well as a confirmation email. If the user inputs an amount greater or equal to 100$, the user automatically participates in a lottery and an additional email containing the lottery information and the lottery ticket number gets sent to the user.

3) Activities section

For the activities section, the four uses cases below are considered

a) Event booking System [ID:7]

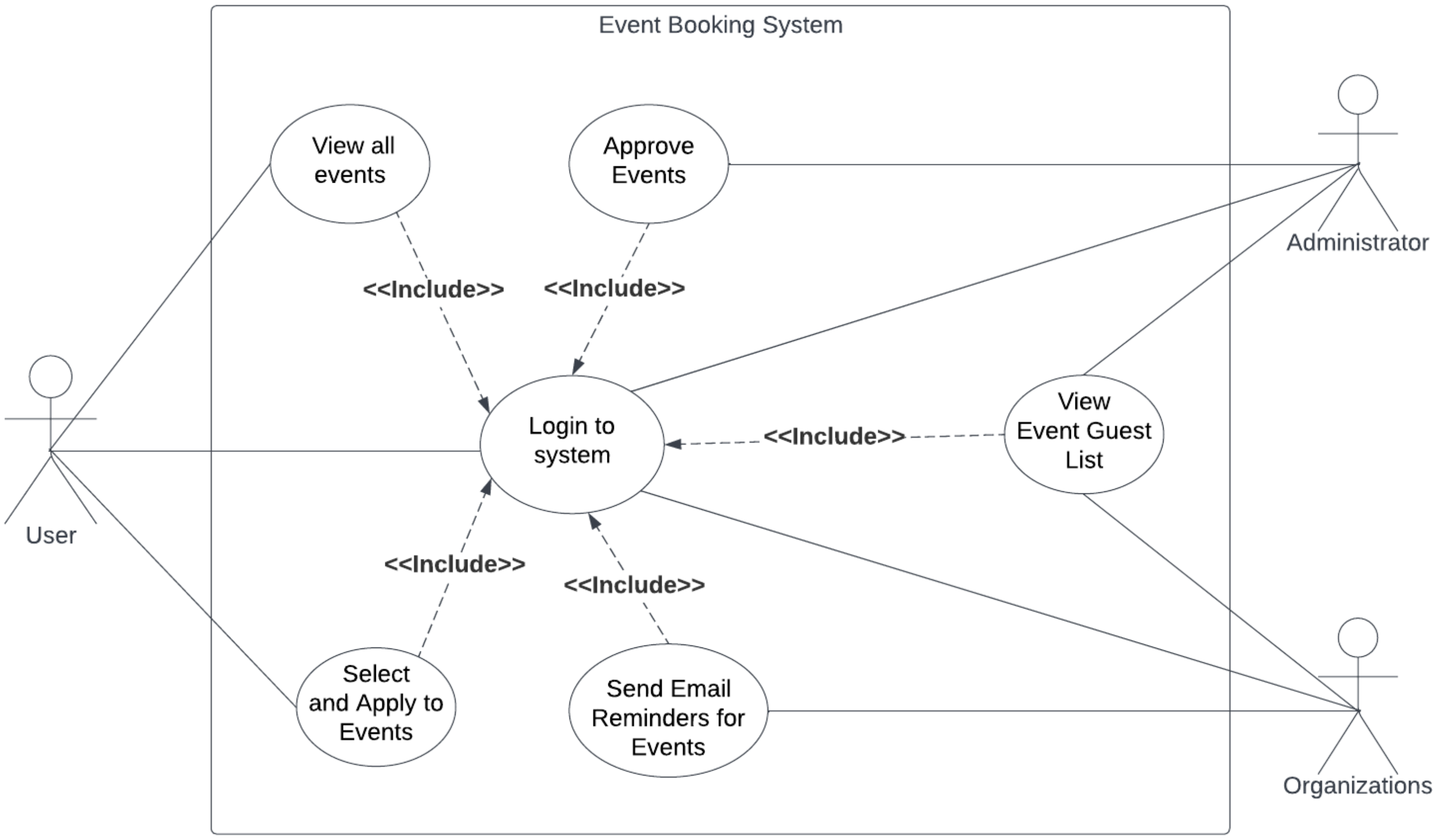

Figure 7 : The use case diagram of a User booking events that organisations post on the website

The primary actor of this system is the user. The user can navigate to the events page to view all the events posted by different organisations. The user can select events and book them if they are interested. The secondary actors are the administrators and organisations. Organisations can post events by sending a request and admins can approve them if they conform to the website's criterias. Organisations can see a list of the people who have booked for specific events as well as the admins. Importantly, all of these actions can only be done when the actors of this use case have logged in to their respective account.

b) Reminders system [ID: 8]

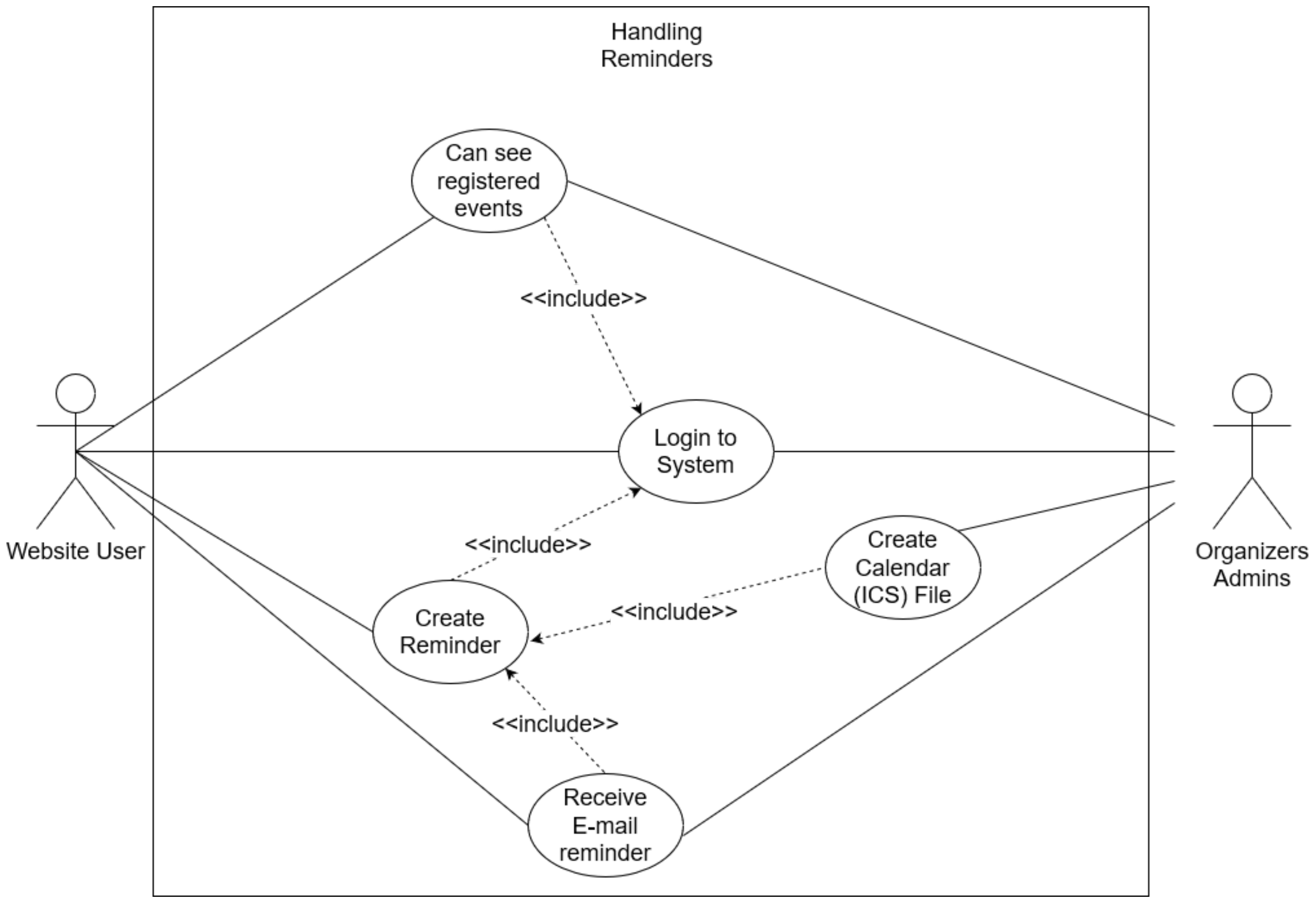

Figure 8 : The use case diagram of how reminders are made/handled

The primary actor in this use case is any one participating in events, seminars, workshops, meetings, in the near future. In other words, it is the user. The event organisers and administrators are responsible for providing reminders for events by either providing a calendar file (ics) for download or by emailing users. In both formats, they must provide the following information of the event:

- Name;
- Date & Time;
- Location;
- Brief Description of Event;

Users need to be logged in, and must be on the events page to select events they would like to participate in. For each event selected, a message dialog pops up on their screen that allows the user to choose how they would like to be reminded of the event: email, or calendar.

If a user chooses email, then a confirmation email will be sent immediately to their email address. The user is then reminded of the event the day before it occurs.

If the user chooses the calendar option, then the calendar file (ics) is downloaded onto their device. When they open the file, the calendar app on their device is opened. Afterwards, they can choose when to be reminded of the event.

c) List of available seasonal activities (seminars, workshops, courses, sports, etc.) [ID: 9]

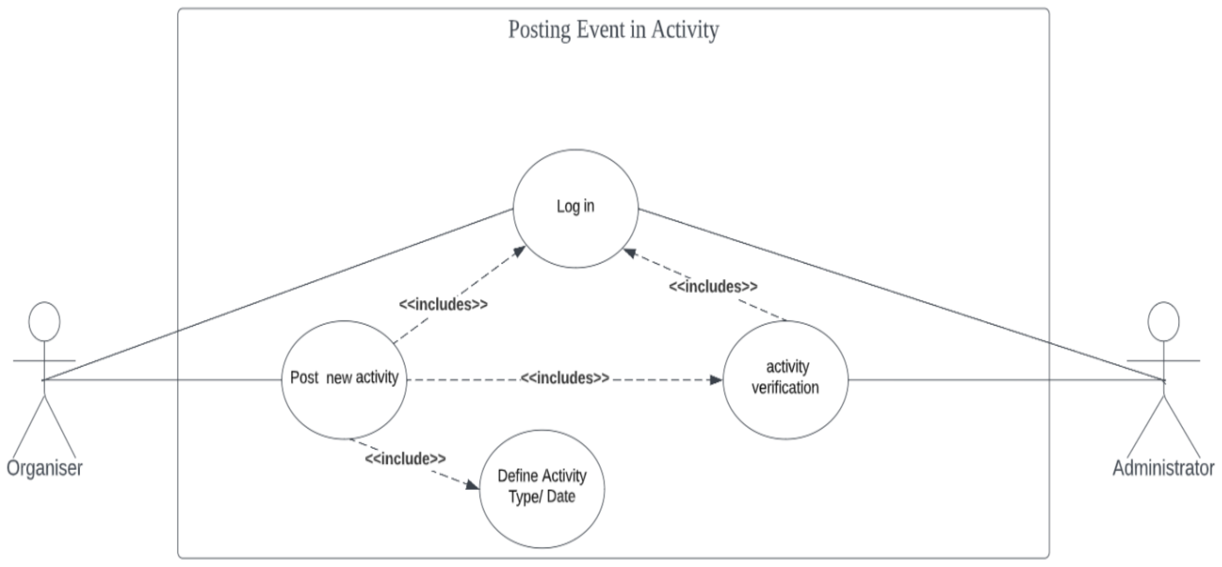

Figure 9 : The use case diagram of the posting an event system

The primary actor of this use case is the organiser. The organiser is able to post an activity and enter all the necessary information. The admin approves or denies the request of the activity. All actors need to be logged in in order for this functionality to execute.

d) System for Searching and filtering activities [ID: 10]

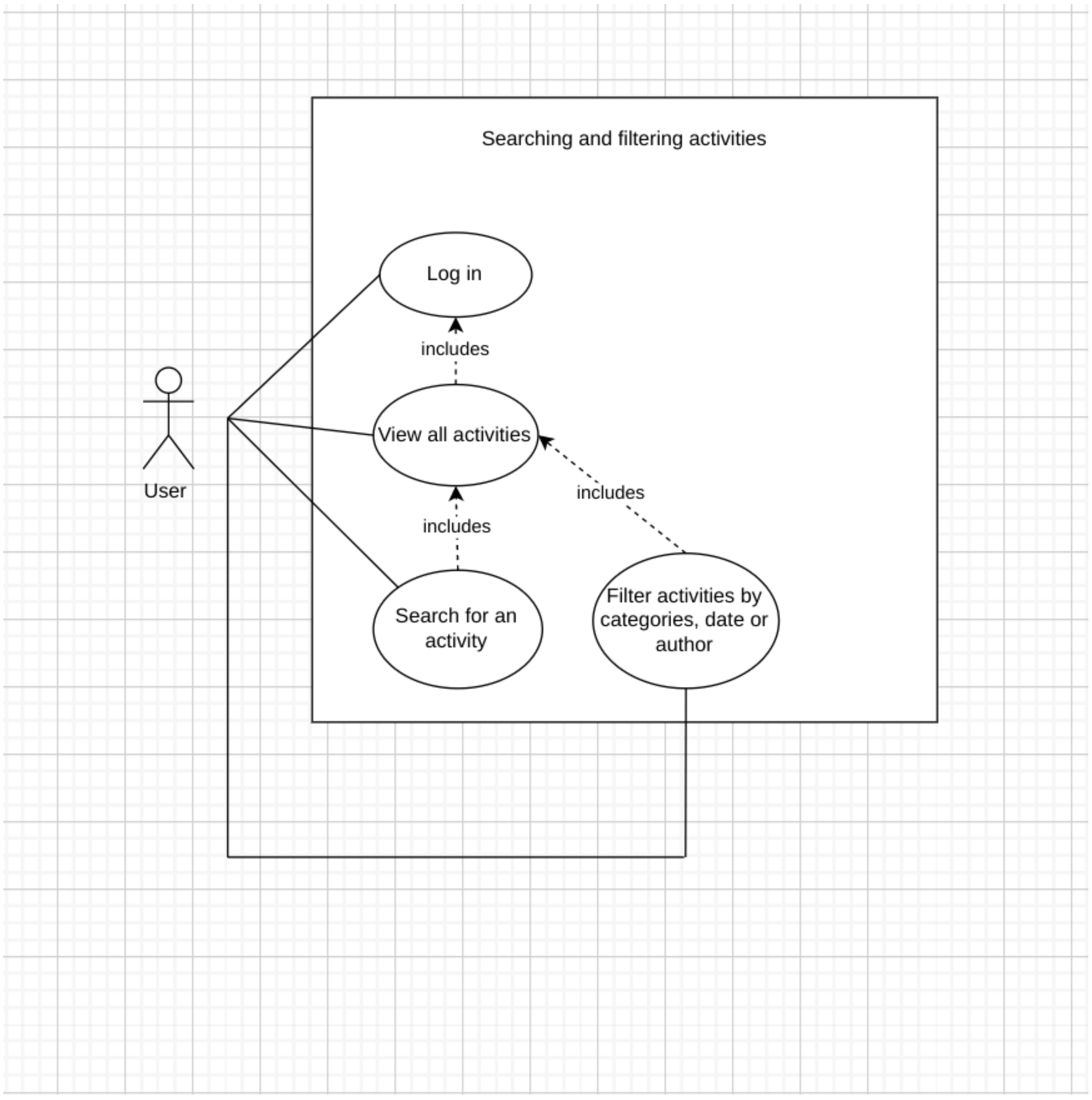
Searching and filtering activities
Log in
includes
View all activities
User
includes
includes
Search for an activity
Filter activities by categories, date or author

Figure 10 : The use case diagram for User searching and filtering activities

The primary actor of this system is the User. The user can view the activity and can search and filter the activities.

In this seciton, the use cases scenarios of each of the use cases are presented. Each of the scenarios are linked to one of the use cases from the previous section. The ID number determines the use case diagram in question.

1) Request centre section

For the request centre section, 3 use case scenarios are considered

Table 4 : Use case scenario of system applying for memberships and letter news

ID	1
Title	Applying for membership and letter news.
Description	The user is able to register for memberships and sign up for newsletters.
Primary Actor	User
Preconditions	User already has an account registered on the platform
Postconditions	User has access to newsletters and membership access to a community service
Inputs	- Payment
Outputs	- Confirmation message - Email receipt - Access to newsletter and membership
Main Success Scenario	- The user login to the platform - The user clicks the membership button - The user enters the necessary information - The user pays the necessary amount - The user receives confirmation message and receipt through email - The user has access to newsletters and membership

Table 5 : Use case scenario of system for sharing comments/reviews/complaints

ID	2
Title	Sharing comments, reviews, complaints, etc.
Description	The user is able to share comments, reviews and submit complaints.
Primary Actor	User
Preconditions	The user has to have an account and be logged in to the system.
Postconditions	The comments, reviews and complaints are added into the system and in the UI. Any user or employer is able to see them.
Inputs	Comments, reviews and complaints
Outputs	- Confirmation message - Complaint number via email
Main Success Scenario	-The user login to his account - The user adds comments on events/activities - The user adds review for activities - The user adds complaints for activities or of the app in general. - A confirmation message appears letting the user know that their comment/review/complaint was added to the system. - User receives the complaint number via email.

Table 6 : System for lottery and lottery results notifications

ID	3
Title	Lottery and lottery results notifications
Description	The lottery draw is performed by the request center, once the draw is completed everyone that received a winning number would be sent a notification about what they won and instructions on how to receive the prizes with their ID and the confirmation notification as proof.
Primary Actor	Administrator

Preconditions	User has an account, enrolled in the lottery and is given a unique lottery ticket number
Postconditions	The lottery results are posted to the website, and all the winners have been notified via email on how they will receive their prizes
Inputs	List of prizes the center has to give away
Outputs	- Confirmation messages - Posted lottery numbers
Main Success Scenario	- The admin puts in the names of the prizes, and the quantity - The site generates a winning number for each of the prizes - The site is updated to show the new winning numbers in the lottery info section - Confirmation messages are sent to all the winners (uses who have a winning number)

2) Volunteering and humanitarian section

For the volunteering and humanitarian section, the 3 use case scenarios are considered.

Table 7 : Use case scenario of Job/Volunteering Application system

ID	4
Title	The user applies for a job/volunteering position found in the system
Description	The user is able to view the many jobs/volunteering opportunities that are given by employers in the system. If the user is interested in one of these jobs, he/she can apply for the job and submit all the necessary information required.
Primary Actor	User
Preconditions	The user has to have an account and be logged in to the system before initiating the first action

Postconditions	- The user successfully applied for the job - The user is no longer able to apply again for the same position.
Inputs	User information and CV
Outputs	Confirmation message of successful application and message about a potential interview, if necessary
Main Success Scenario	- The user login to his account - The user selects a job/volunteering position that he/she is interested - The user enters all necessary information and adds CV - The user submits application correctly - A confirmation message appears letting the user know that application was sent out successfully - The employer view's the submission of the user - The employer accepts or rejects the submission and sends out a message to the user

Table 8 : Use case scenario of Job/Volunteering Request system

ID	5
Title	The employers posts a job/volunteering position into the system
Description	If the employer has a new position for their company and would like to hire some people, he/she can submit the position into the system and interview candidates.
Primary Actor	Employer
Preconditions	The employer has to have an account.
Postconditions	The job is added into the system and in the UI. Any user is able to see the job details.
Inputs	Job/Volunteering position details from the Employer
Outputs	- Confirmation message of successful submission of job. - Awaiting Confirmation message letting

	employer know that the job has been added to the system
Main Success Scenario	- The employer login to the account - The employer selects the option to add a job/volunteering position - The employer adds all the information of the job/volunteering position - The employer submits the application - A confirmation message is sent to the employer letting them know that the application was sent out successfully - The admin gets a message about a job request from the employer. - Admin accepts the job request - Employer gets a message letting them know that the job has been approved and is added into the system

Table 9 : Use case scenario of Donations Transaction system

ID	6
Title	The user donates to the community service
Description	Any website user can choose to donate to the organisation to help reduce the price of the activities for the youth. By submitting a donation higher than 100$, the user will automatically enter an annual lottery with a prize of 100 000$
Primary Actor	User
Preconditions	- User has to have an account and be logged in - User needs to have a valid payment method
Postconditions	The user will receive a confirmation message from the system that the payment has successfully been processed as well as a confirmation email. Additionally, if the donation amount is 100$ or more, an additional email with the lottery ticket number will be sent.
Inputs	Payment method (PayPal or credit card), card number, user info
Outputs	Confirmation message, confirmation email and lottery ticket number sent via email if user eligible for the lottery
Main Success Scenario	- User logs into their account

	- User navigates to the donation page - User selects donation amount - User enters the correct payment information - The service provider (Paypal / Credit Card) accepts the transaction - Website shows a confirmation message - User receives a confirmation email - User receives a lottery ticket number by email (if donation at least 100 $)

3) Activities section

For the activities section, the four use cases scenarios are considered.

Table 10 : Use case scenario of Event booking System

ID	7
Title	User booking an event
Description	A user books an event and enters that event's guest list if they are interested in participating in it.
Primary Actor	User
Preconditions	The user needs to have an account and needs to be logged into that account before booking an event.
Postconditions	The user is added to the guest list of that event. The user receives a confirmation email. The user receives reminders about the event (optionally).
Inputs	List of events the user booked for.
Outputs	Confirmation email confirming the user has been added to the guest list.
Main Success Scenario	- User logs into their account - User navigates to the events page - User selects an event and clicks on book event - Website sends confirmation email to the user - User confirms booking in the email sent - User is redirected to the website - Website displays confirmation message letting the user know that he has been booked for the event

Table 11 : Use case scenario of the Reminders system

ID	8
Title	Creating/Handling Reminders
Description	The user is able to create reminders for organised events taking place in the near future.
Primary Actor	Website User
Preconditions	End-user must have access to the system (website) and must be logged in with an account.
Postconditions	User has the choice to cancel the reminder or remove it after it has been created.
Inputs	Select the activities/events that you want a reminder for
Outputs	Confirmation Email, or downloaded calendar file.
Main Success Scenario	- The user logs in to the platform - The user clicks the membership button - The user clicks on future events - The user chooses events they would like to participate in. - A message dialog appears, asking the user if they would like to download a calendar (ics) file, or receive emails to remind them of the event. - ICS File - The user selects a calendar file. The event information is extracted [from a database or from the website] and used to generate the file. - The file is downloaded onto their device (phone) - The user opens the file, which opens the device’s main calendar application. - The user chooses when the reminder will occur. - Emails - The user selects email. - Assuming the user has already entered the email, they will receive a confirmation email - The user will be alerted shortly before events occur.

Table 12 : Use case scenario of the Posting events system

ID	9
Title	Posting an event in activity
Description	An event organiser posts an official event to be displayed on the website and the administrator accepts or denies its posting.
Primary Actor	Event Organiser
Preconditions	The Organizer is logged in and clicks add events on the activities page of the website.
Postconditions	A new page on the website displaying the event, its details and allowing booking.
Inputs	A public event with its date duration and price.
Outputs	A new page on the website detailing event.
Main Success Scenario	← The organiser enters their login information → The system verifies their identity through the database ← The organiser presses the add event button on the Activities page → The website redirects them to the add event page ← The organiser details the event, its location, date and duration. They then press the post button. → The system stores this information in the database however, it does not post it until administrator approval. ← An admin logs in and approves the event. → A new page is added to the website detailing the event and its cost.
Alternate scenario	← The organiser enters their login information → The system verifies their identity through the database ← The organiser presses the add event button on the Activities page → The website redirects them to the add event page ← The organiser details the event, its location, date and duration. They then press the post button. → The system stores this information in the database however, it does not post it until administrator approval. ← An admin logs in and rejects the event due to faulty advertising or breaks in code of conduct. → The organiser is alerted via email that their event is rejected

Table 13 : Use case scenario of the User searching and filtering activities

ID	10
Title	Searching and filtering activities
Description	A user, after logging in, can search for different activities on the website and filter them depending on the filters requested
Primary Actor	User
Preconditions	The User is logged in and clicks see or search for activities using, if requested, search filters.
Postconditions	The page displays the requested activities depending on the requested query. It filter the activities depending on the date, category and author
Inputs	A string requesting an activity type.
Outputs	A new page on the website detailing activities.
Main Success Scenario	← The user enters their login information → The system verifies their identity through the database → The website displays all the activities ← The user can view, filter, search for activities

4. Sequence diagram

Below are the sequence diagrams of each of the use case scenarios that were defined in the previous section. The ID number determines the use case scenario in question for each of the sequence diagrams.

1) Request Center Section

Sequence diagram for Use Case Scenario ID 1:

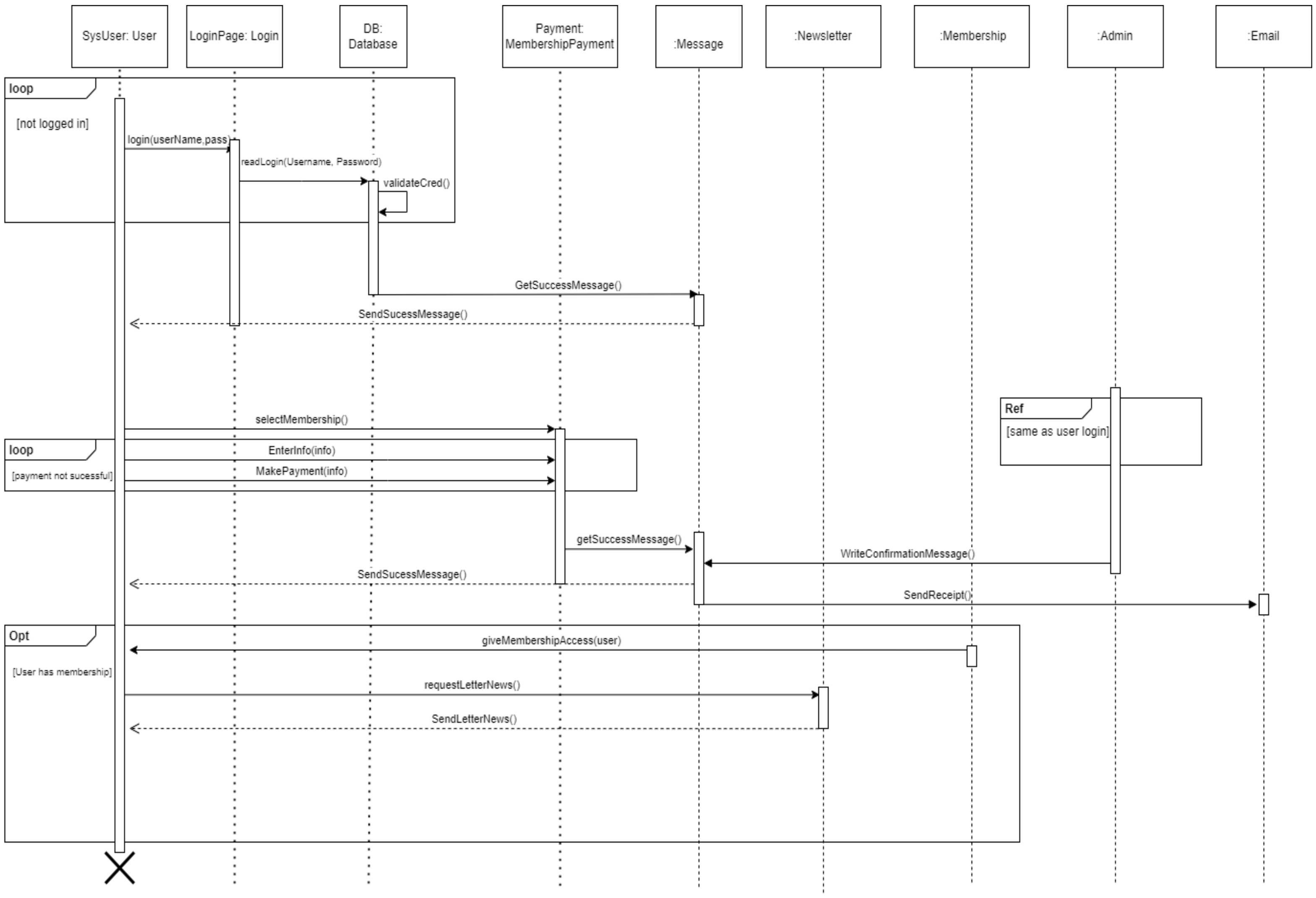

Figure 11 : Sequence diagram of User applying for Membership [ID: 1]

In this sequence diagram, some of the classes have responsibilities. The Database class is the information expert because it has the necessary information to verify the credentials for user login.The Login Class is a pure fabrication class in which we assign the responsabilities in getting the input from the user. The message class is also a purely fabricated class in which we assign the responsibilities of sending success or error messages to the users when triggering an event. The email is a pure fabrication that is assigned to send emails to the user. The membership class is a purely fabricated class which has the responsibilities to assign membership access to users.

Sequence diagram for Use case scenario ID 2:

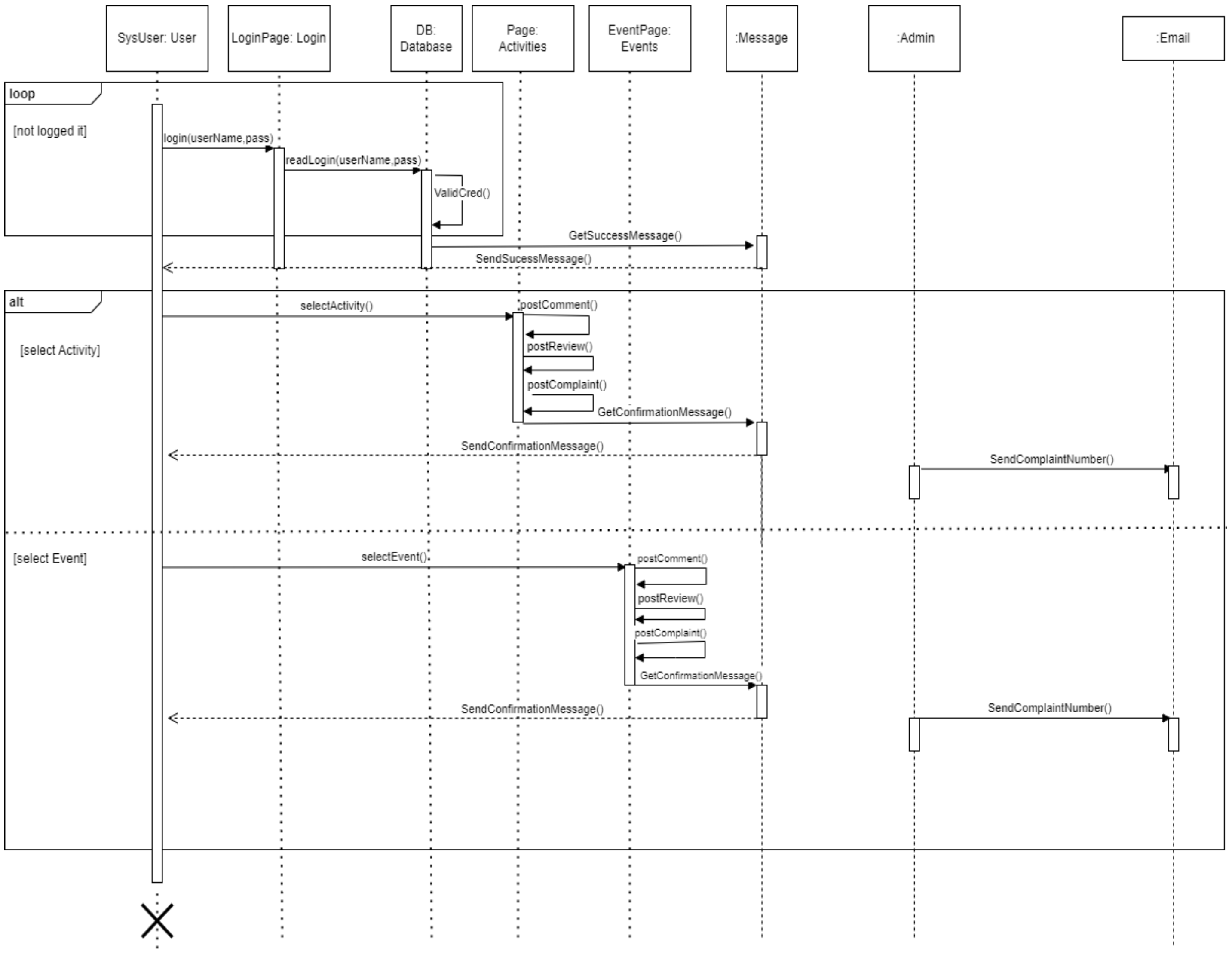

Figure 12: Sequence diagram for applying for posting reviews/complaint [ID: 2]

A new class that was not defined in the previous sequence diagram and not in the domain model is Events class. This class uses the information expert pattern as it has the necessary information to display the activities and events and process any request from the users.

Sequence diagram for Use case scenario ID 3 :

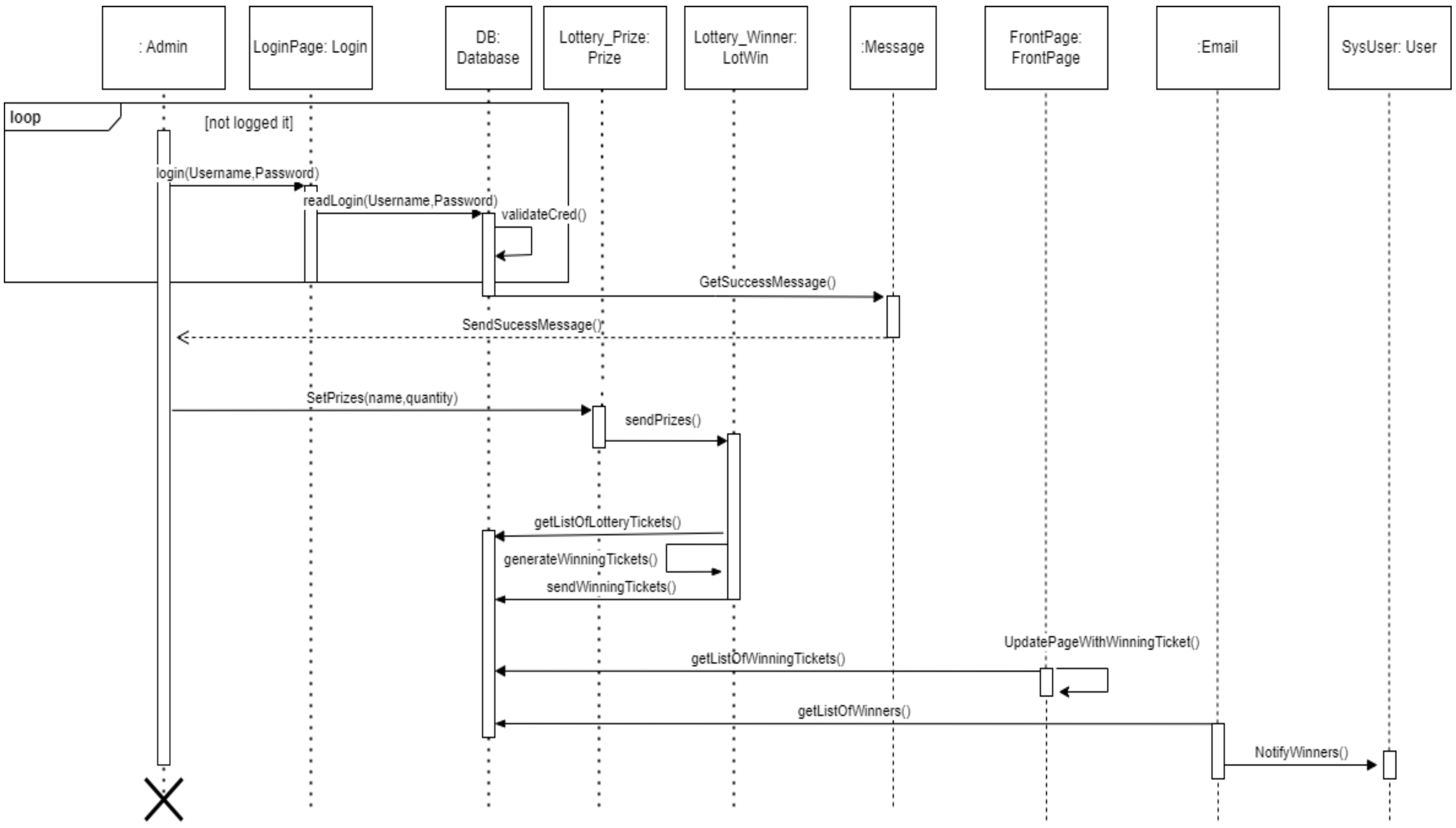

Figure 13 : Sequence diagram of Generating winners of the lottery [ID: 3]

In this diagram, the newly added classes with responsibilities are Prize and LotWin and FrontPage. The class Prize is a purely fabricated class which sends the list of prizes for the lottery to the LotWin class. The LotWin is the information expert class since it contains the list of the lottery prizes as well as the selected randomised lottery winner. The FrontPage class is a purely fabricated class which has the responsibilities of displaying information such as winning lottery tickets into the page.

2) <u>Volunteering and humanitarian section</u>

Sequence diagram for Use case scenario 4:

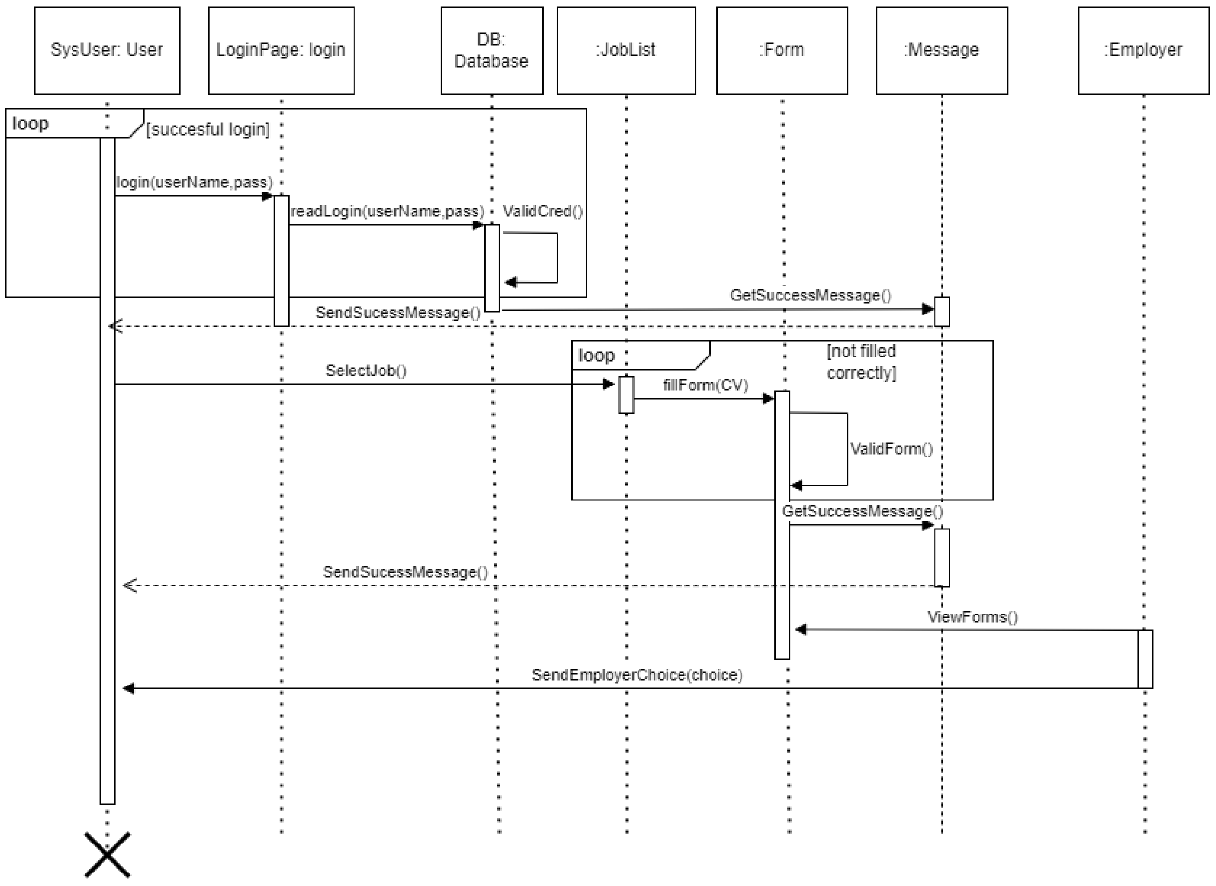

Figure 14 : Sequence diagram of applying for job/volunteering position [ID :4]

The new classes that were added in the sequence diagram are Login, Database, Form and Message. The Database Class is an Information Expert for the User credentials. The Database has all the information needed to authenticate the User. The Form class uses the Creator Pattern. It has the responsibility of creating the job application of the user and sending it to the Database. The Login Class is a pure fabrication class in which we assign the responsabilities in getting the input from the user. The message class is also a purely fabricated class in which we assign the responsibilities of sending success or error messages to the users when triggering an event.

Sequence diagram for Use case scenario 5:

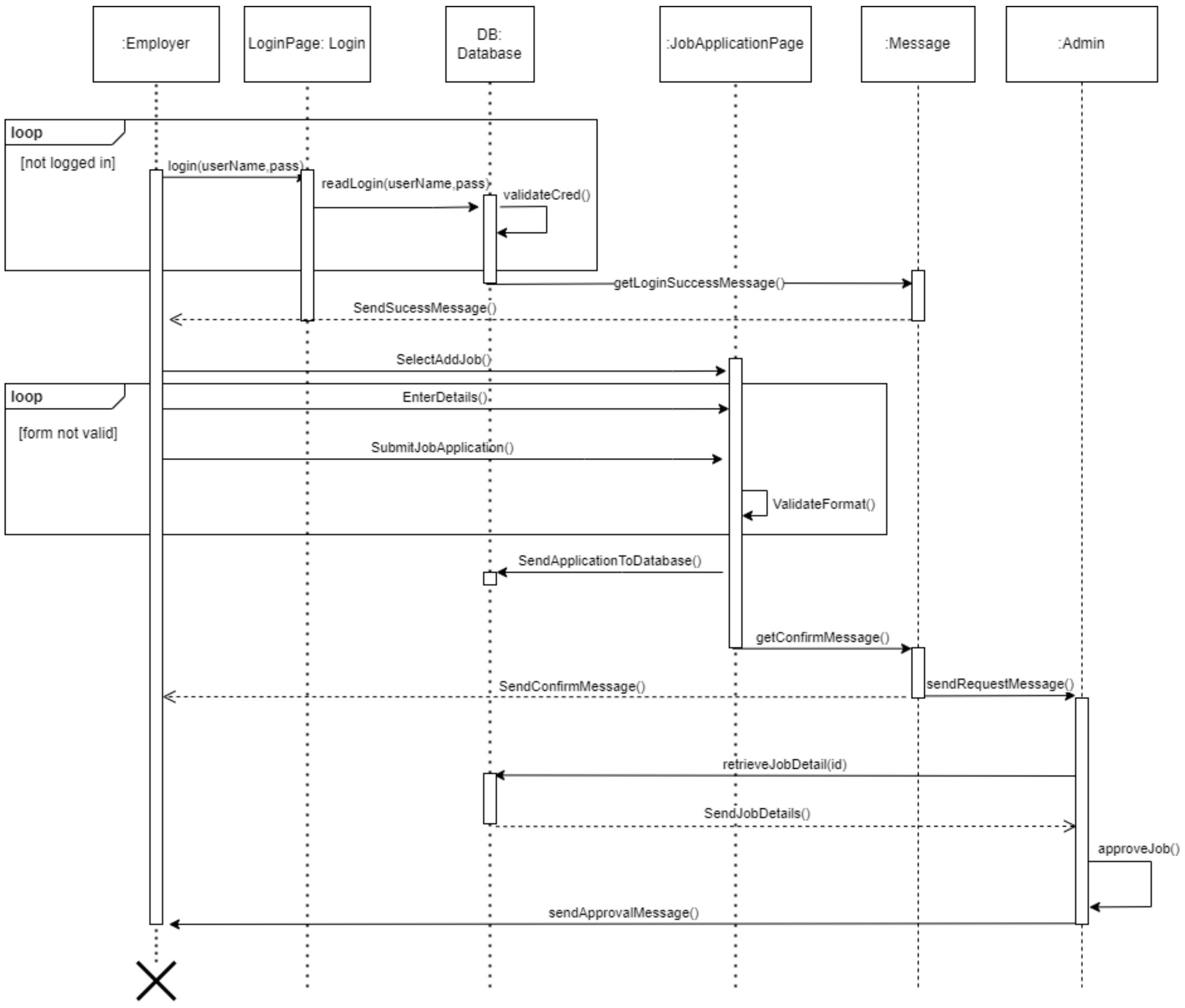

Figure 15 : Sequence diagram of applying for requesting a job/volunteering position [ID :5]

The new classes that were added in the sequence diagram are Login, Database, Job ApplicationPage and Message. The Login, Database and Message patterns are the same as in the previous use case scenario (ID 4) (Information Expert, Pure Fabrication and Pure Fabrication respectively). The newly added class that was not found in previous sequence diagrams and in the domain model is the JobApplicationPage. This class uses the creator pattern as this class will be responsible for creating a job request application to the database so that the admin can approve or deny the request. It is also an information expert class as it has all the information of all jobs in the system.

Sequence diagram for use case scenario 6:

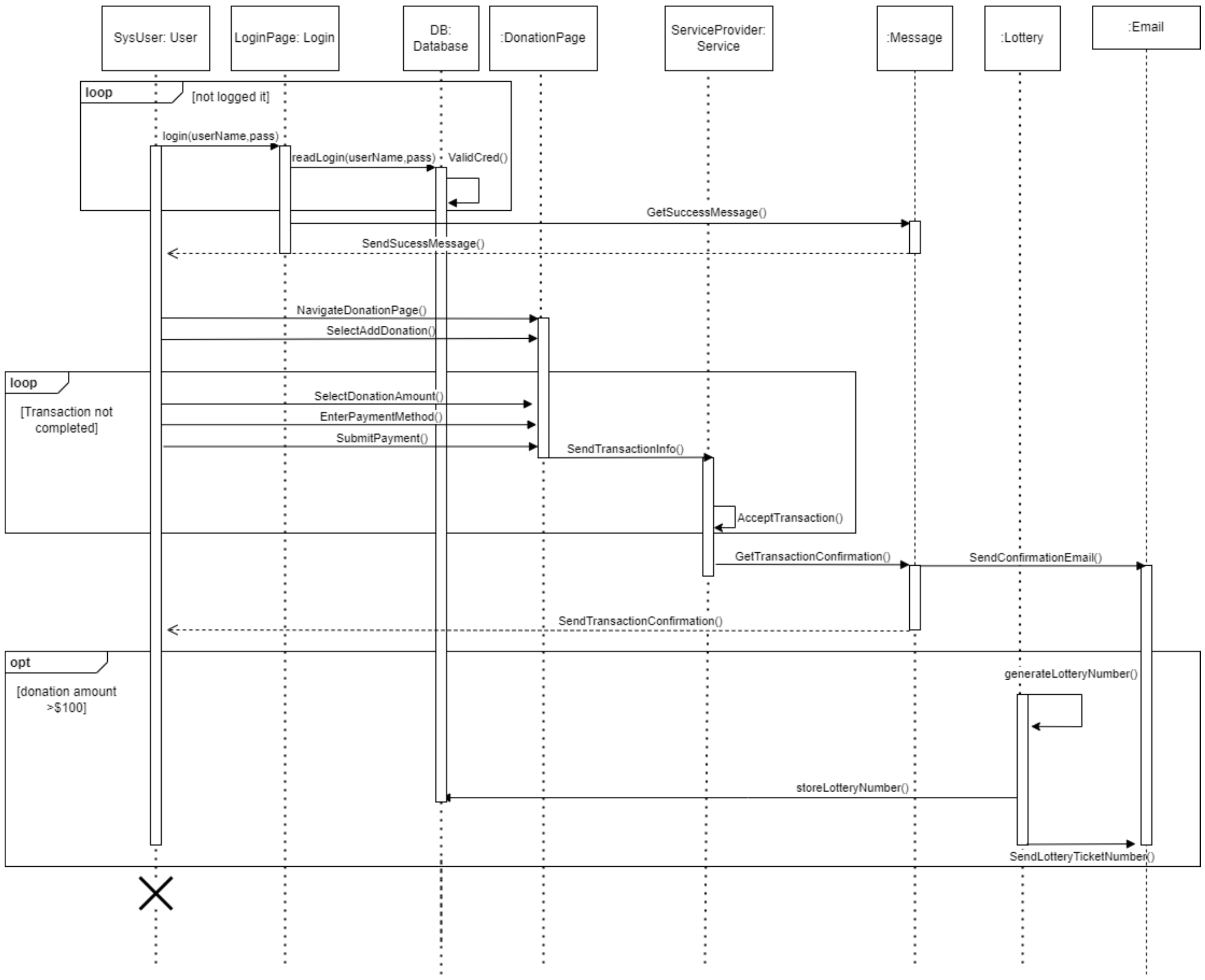

Figure 16 : Sequence diagram for submitting donations [ID: 6]

For this sequence diagram, the newly added classes that were not found in previous sequence diagrams are: DonationPage, Lottery and ServiceProvider. The DonationPage class is the information expert since it holds all the needed information related to the donation, it also uses the creator pattern which initiates an instance of the class Transaction. The Lottery class is a purely fabricated class which has the only function of generating a unique lottery ticket and sending it to the database.The ServiceProvider class is also a purely fabricated class responsible for accepting or refusing the transaction.

3) <u>Activities section</u>

Sequence diagram for use case scenario 7:

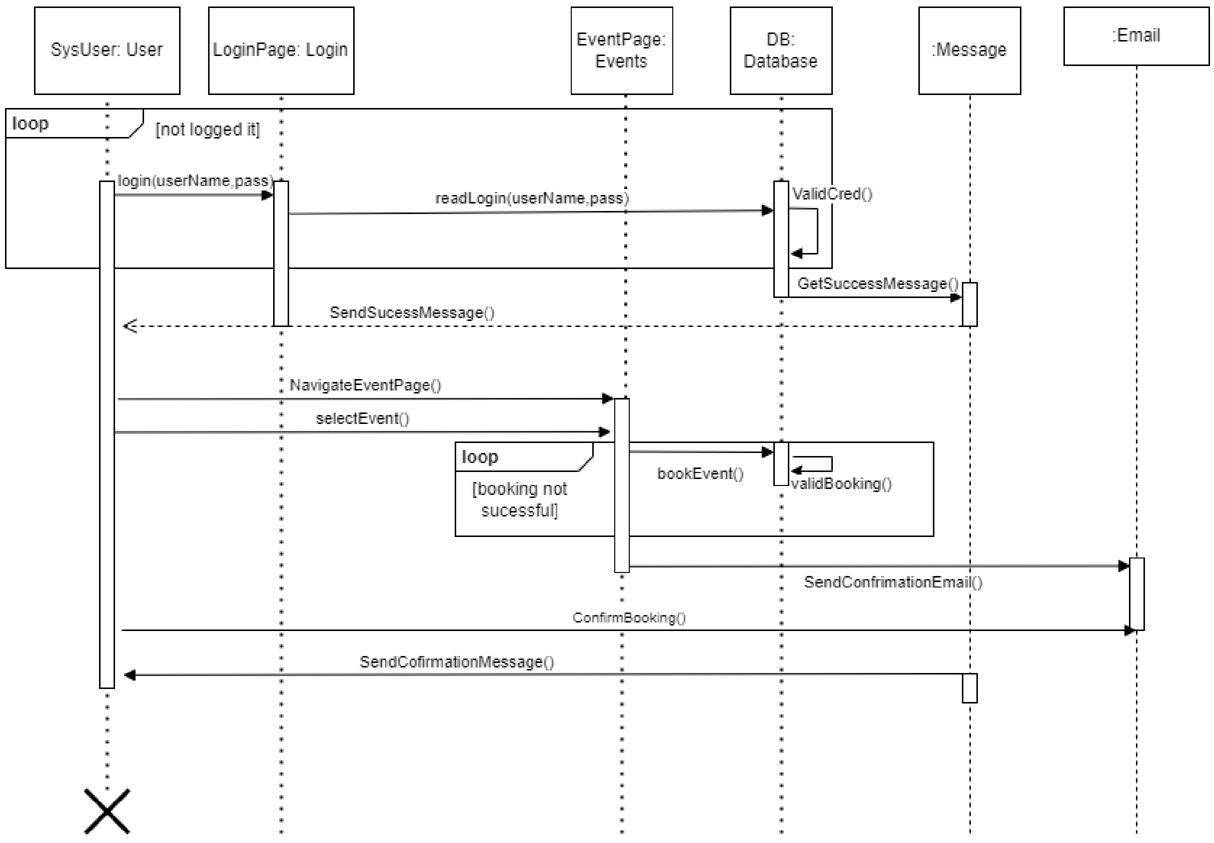

Figure 17 : Sequence diagram of user booking for a event [ID: 7]

For this sequence diagram, the newly added class which wasn't found in the previous sequence diagrams is the EventPage class. This class is an information expert class which holds all the necessary information about the different types of events available.

Sequence diagram for use case scenario 8:

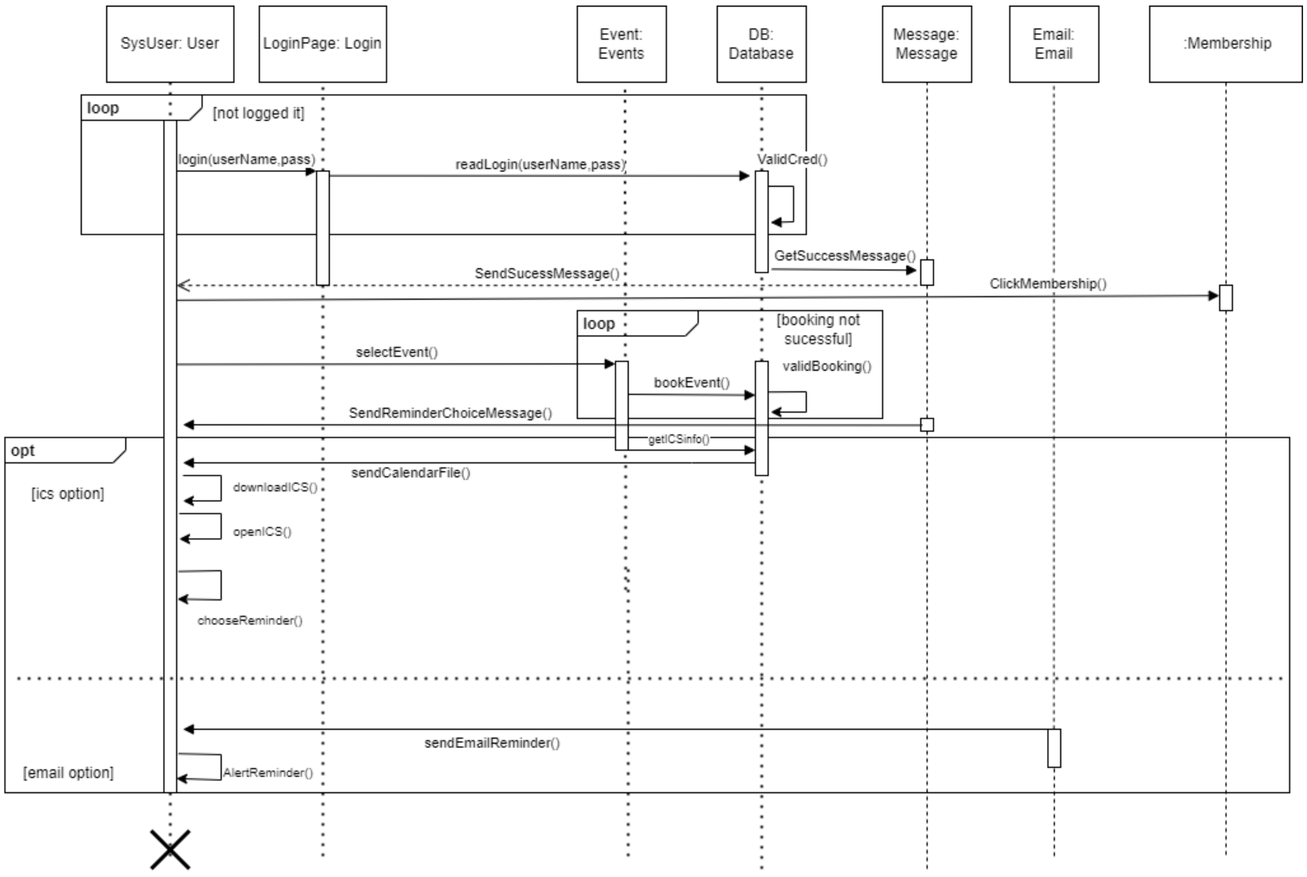

Figure 18 : Sequence diagram for reminders in events [ID: 8]

This sequence diagram is very similar to the previous one (ID 7). The only new added class is the Membership class which is purely fabricated to simply validate if the given user has a membership for the website or not and other responsibilities.

Sequence diagram for use case scenario 9:

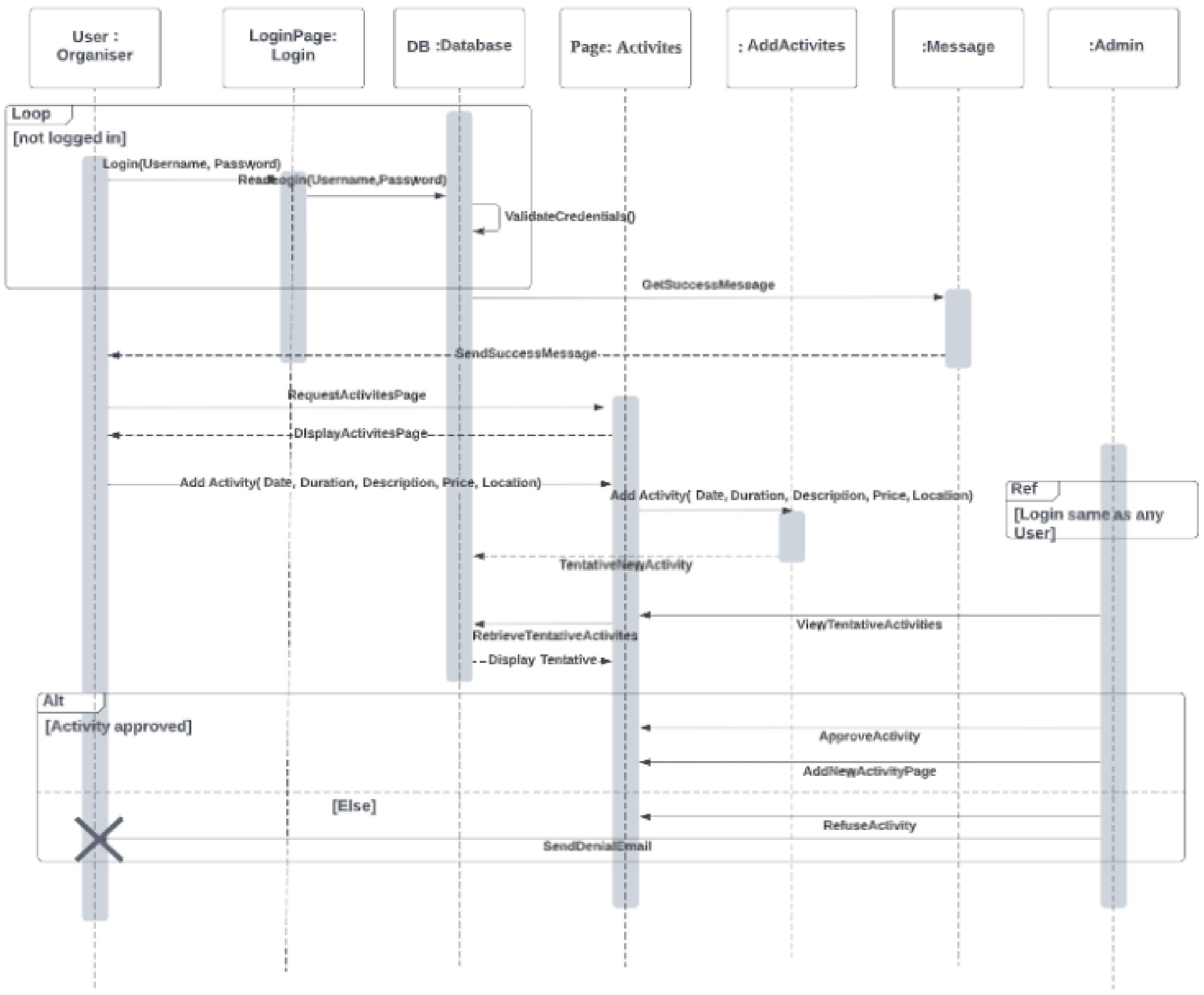

Figure 19 : Sequence diagram of adding activity [ID: 9]

For this sequence diagram, the newly added classes AddActivities. The AddActivites class is a purely fabricated class which adds a certain given activity to the database. It is also a creator pattern as it creates the activity and sends it to the database.

Sequence diagram for use case scenario 10:

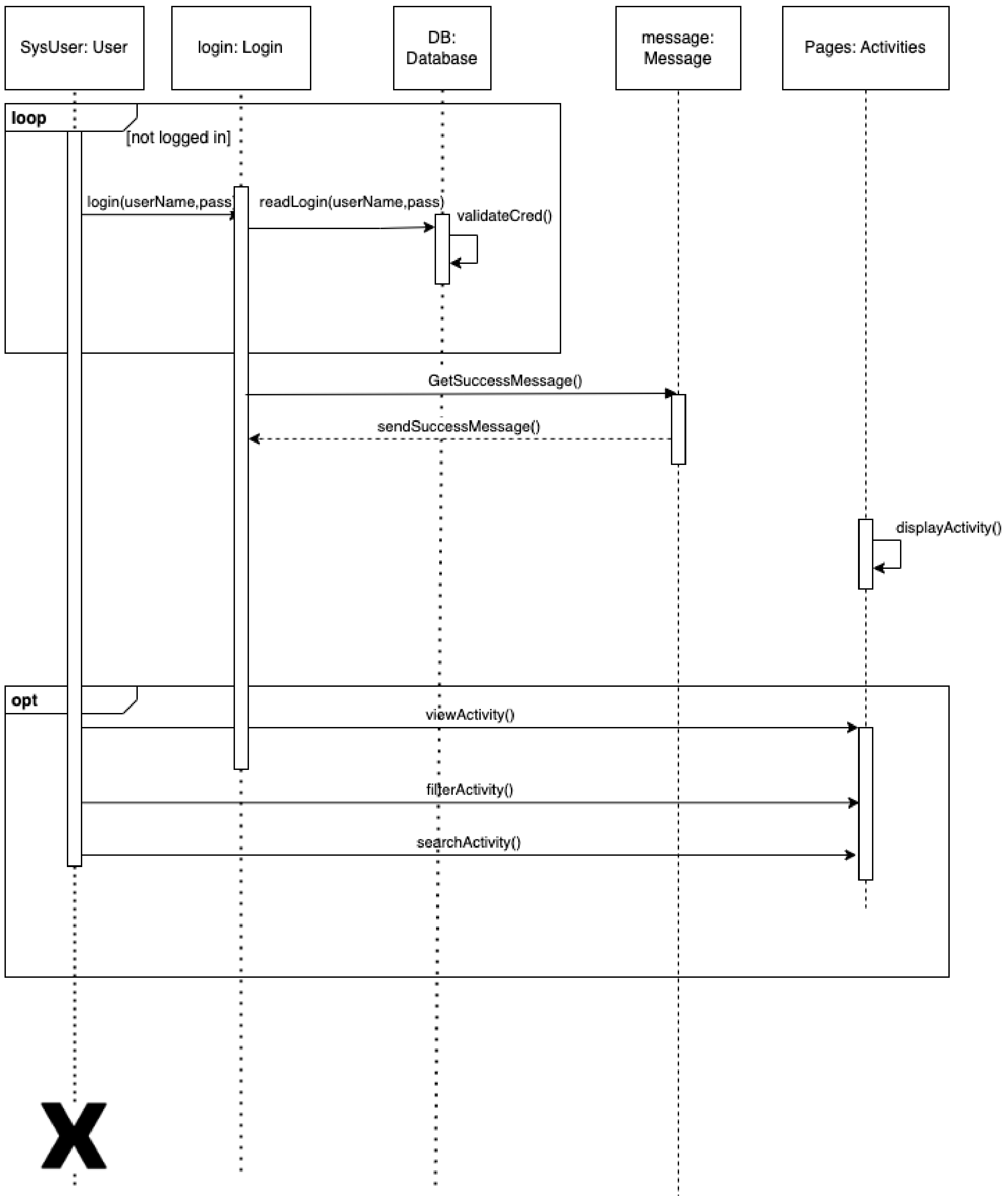

Figure 20 : Sequence diagram for viewing, searching, filtering activities [ID:10]

This sequence diagram represents the searching for activities scenario where a user accesses the activities page and starts searching for a specific activity among the available activities. This sequence diagram does not add any new classes other than the ones previously discussed.

5. Class diagram

Below is the class diagram for the whole Community Service App that we are developing.

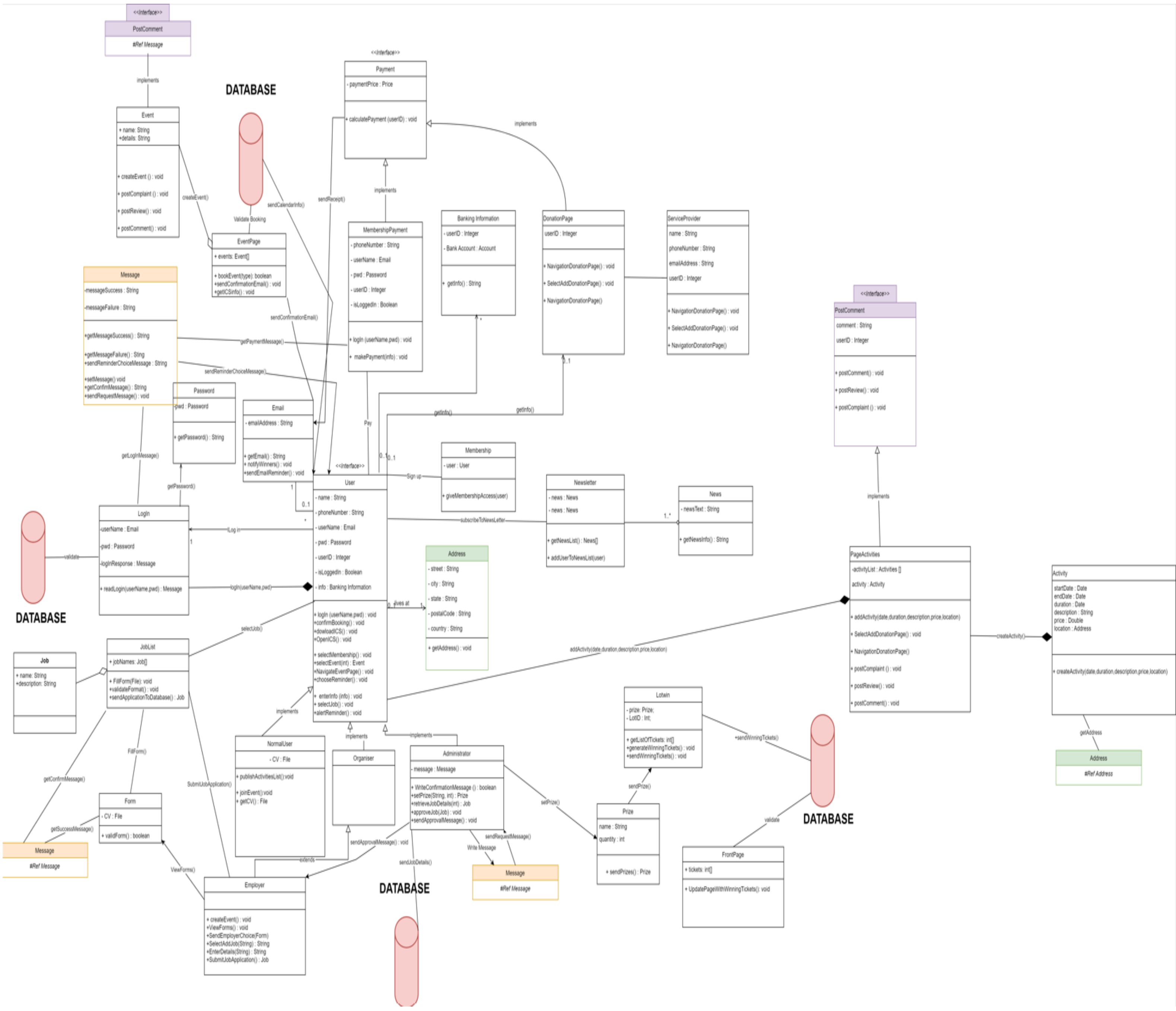

Figure 21 : Class Diagram of the Community Service App

Gang Of Four patterns

The factory method is one of the gang of four patterns that was applied in in our class diagram

<u>Factory Method</u>

Problem : The problem is that each type of user of the system has similar functionalities between each other. We are instantiating each of the users and these functionalities every time, which is leading to a wastage of resources. If we want to add another new type of user in our system, we would need to make a whole new class again.

Solution : The solution is to have an abstract class that each type of user will implement to extend all the common functionalities. We will give the responsibilities to the sub classes to instantiate their own object. This allows reusability of the functionalities, but also allows different behaviours for each of the users.

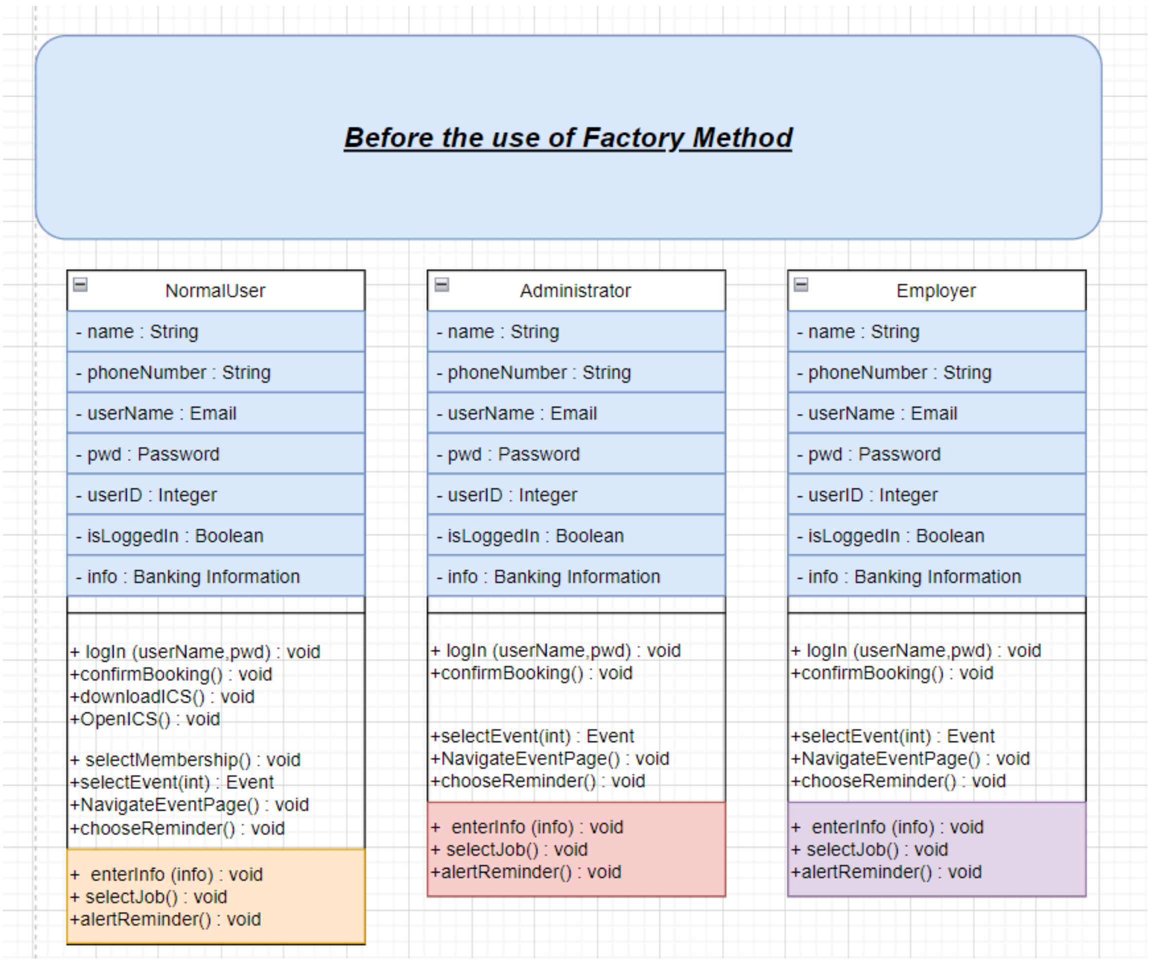

Figure 22 : Users Class Diagrams before applying the factory method

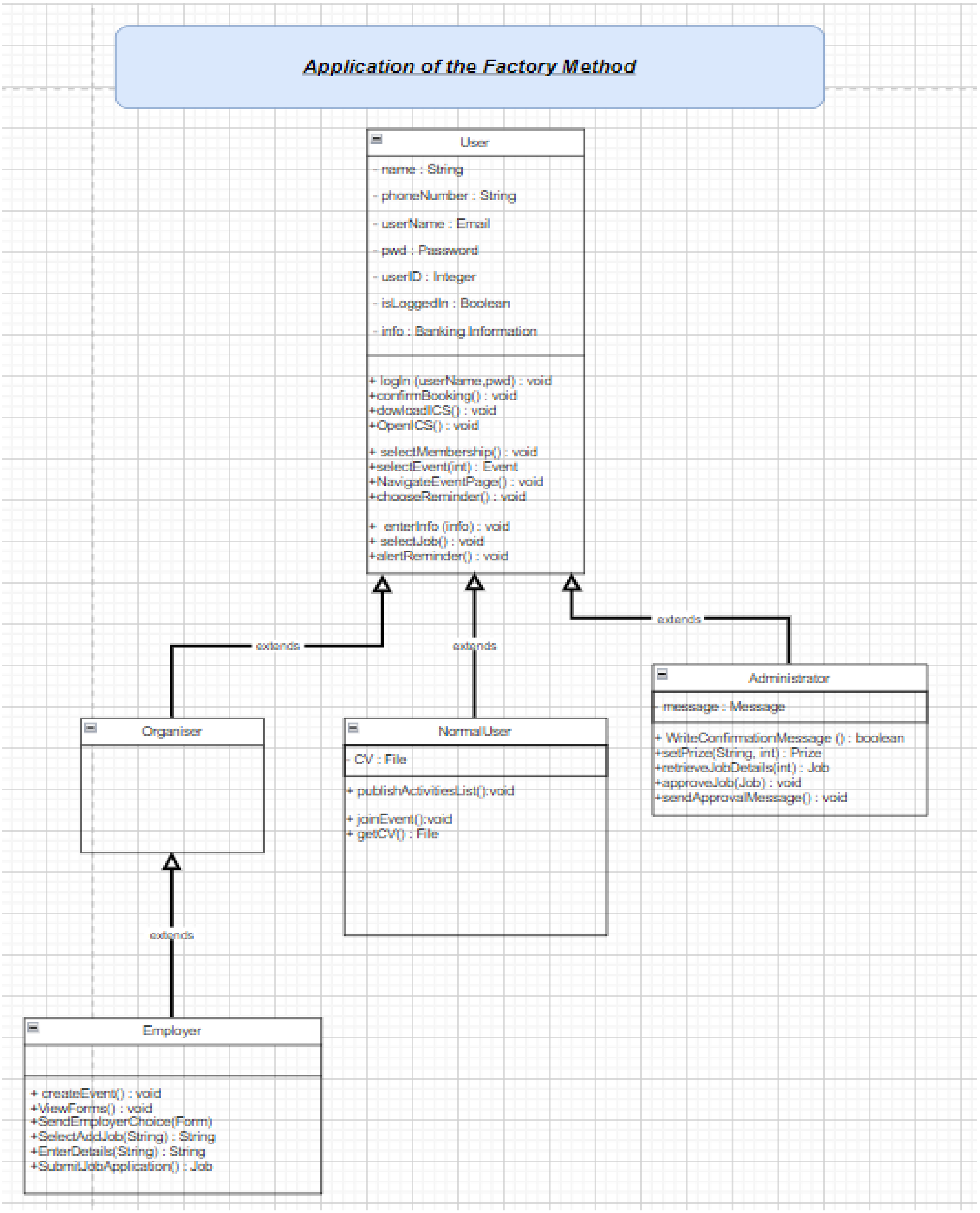
Application of the Factory Method
User
- name : String
- phoneNumber : String
- userName : Email
- pwd : Password
- userID : Integer
- isLoggedIn : Boolean
- info : Banking Information
+ logIn (userName,pwd) : void
+confirmBooking() : void
+dowloadICS() : void
+OpenICS() : void
+ selectMembership() : void
+selectEvent(int) : Event
+NavigateEventPage() : void
+chooseReminder() : void
+ enterInfo (info) : void
+ selectJob() : void
+alertReminder() : void
extends
extends
extends
Organiser
NormalUser
- CV : File
+ publishActivitiesList():void
+ joinEvent():void
+ getCV() : File
Administrator
- message : Message
+ WriteConfirmationMessage () : boolean
+setPrize(String, int) : Prize
+retrieveJobDetails(int) : Job
+approveJob(Job) : void
+sendApprovalMessage() : void
extends
Employer
+ createEvent() : void
+ViewForms() : void
+SendEmployerChoice(Form)
+SelectAddJob(String) : String
+EnterDetails(String) : String
+SubmitJobApplication() : Job

Figure 23 : The application of the Factory Method

The other gang of four pattern used is the strategy pattern

Strategy Pattern

Problem : The problem is that for the methods posting reviews, comments and complaints, the user can use these methods in events and activities classes. We have these common functionalities that are found in both activities and events class, but their implementation and behaviour is different

Solution : The solution is to have an interface that will have the common methods, in which the activities and events class will implement this interface and inherit the methods (postComment(), postReview(), postComplaint()). This will allow the classes to have different implementation and behaviours, even though the method signature is the same.

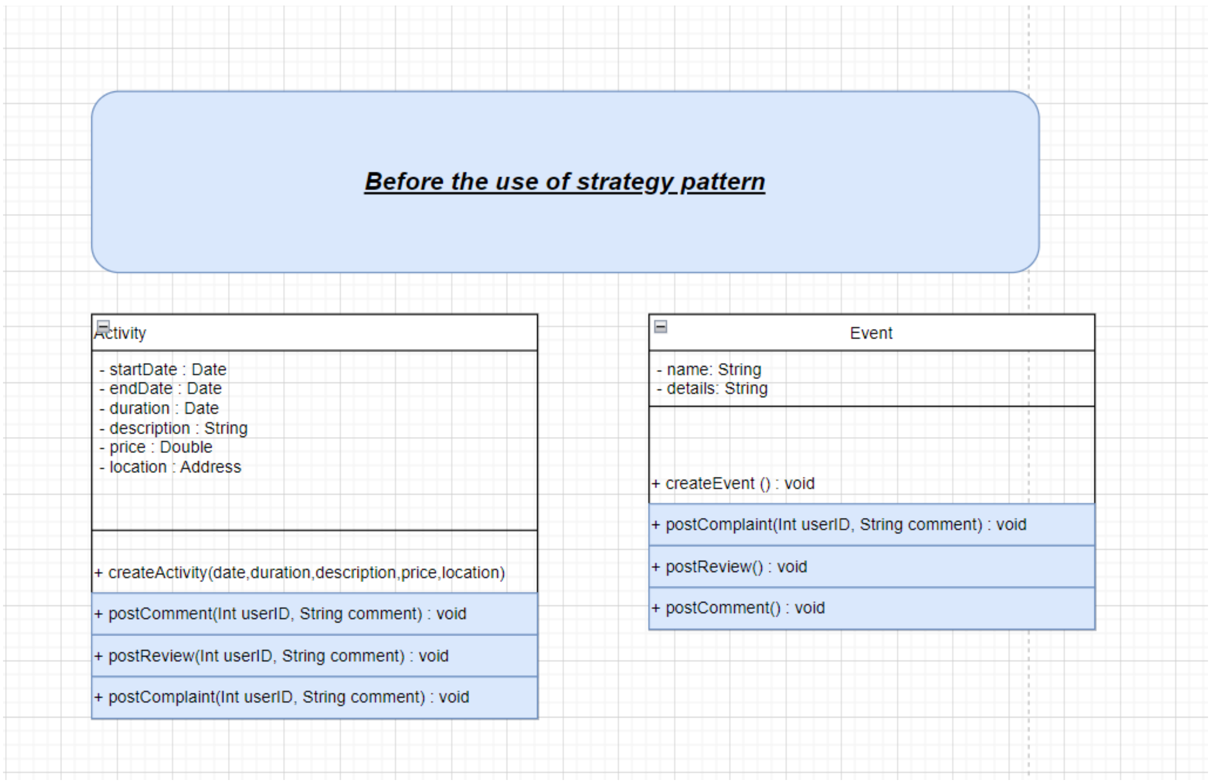
Before the use of strategy pattern
Activity
- startDate : Date
- endDate : Date
- duration : Date
- description : String
- price : Double
- location : Address
+ createActivity(date,duration,description,price,location)
+ postComment(Int userID, String comment) : void
+ postReview(Int userID, String comment) : void
+ postComplaint(Int userID, String comment) : void
Event
- name: String
- details: String
+ createEvent () : void
+ postComplaint(Int userID, String comment) : void
+ postReview() : void
+ postComment() : void

Figure 22 : Activity and Event Class diagram before applying the strategy pattern

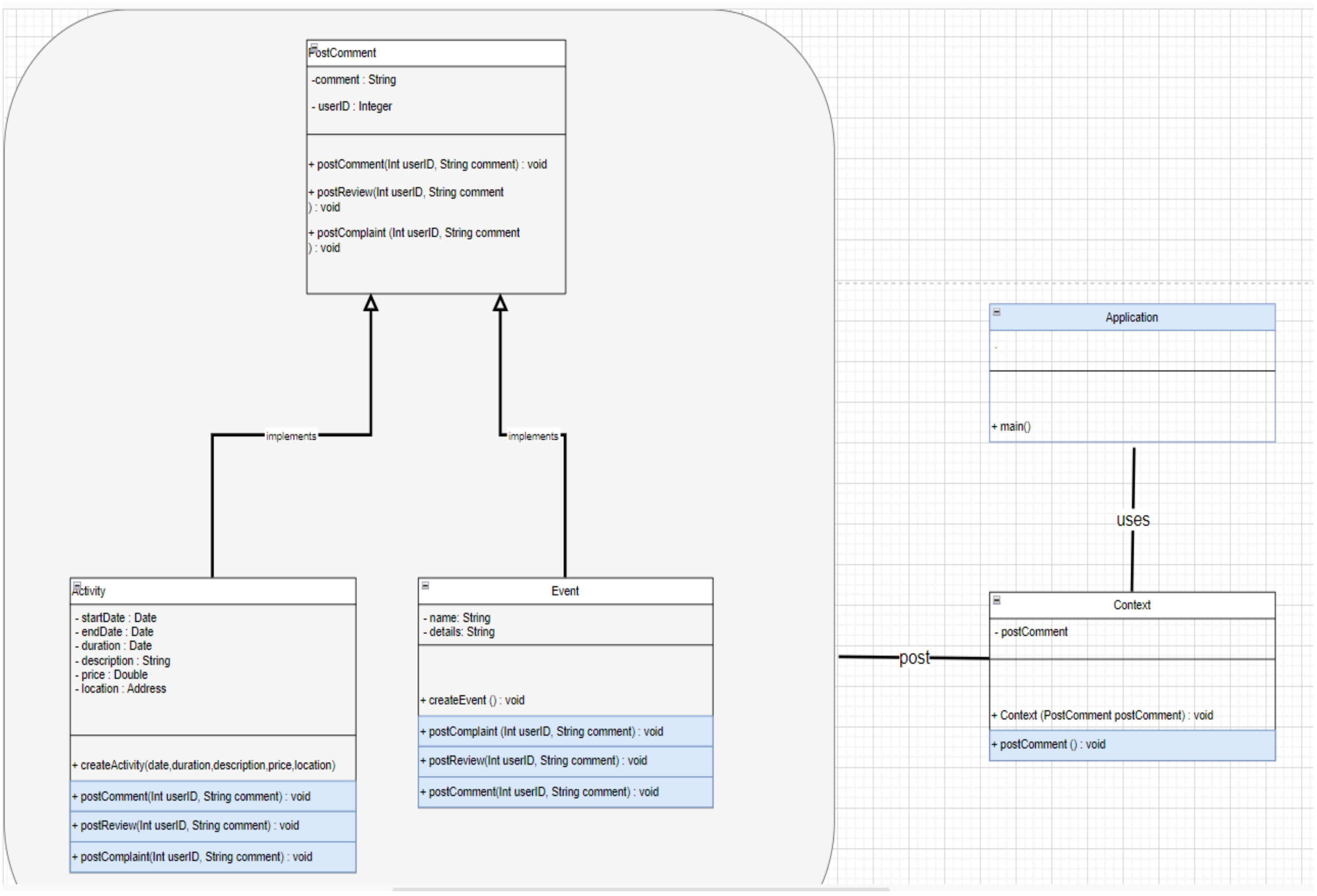
PostComment
-comment : String
- userID : Integer
+ postComment(Int userID, String comment) : void
+ postReview(Int userID, String comment) : void
+ postComplaint (Int userID, String comment) : void
implements
implements
Activity
- startDate : Date
- endDate : Date
- duration : Date
- description : String
- price : Double
- location : Address
+ createActivity(date,duration,description,price,location)
+ postComment(Int userID, String comment) : void
+ postReview(Int userID, String comment) : void
+ postComplaint(Int userID, String comment) : void
Event
- name: String
- details: String
+ createEvent () : void
+ postComplaint (Int userID, String comment) : void
+ postReview(Int userID, String comment) : void
+ postComment(Int userID, String comment) : void
Application
+ main()
uses
Context
- postComment
+ Context (PostComment postComment) : void
+ postComment () : void
post

Figure 23 : Activity and Event Class Diagram after applying Strategy Pattern

6. Contributions of Members

Name	**Task Done**
Imran Ahmed	Summary of the Project + Worked on System Architecture with Titouan + Use case 4 and Use case 5 + Use case scenario 4 and Use case scenario 5 + Worked on Sequence Diagram 1 to 8 as a team.
Titouan Sablé	System Architecture with Imran + Use Case Diagram 7 + Use Case Scenario 7
Joe El-Khoury	Use Case Scenario 8 and Use Case 8 Diagram
Kunal H. Shah	Use Case Scenario 2 + Use Case Diagram 2 + worked on Sequence Diagram(2,7,8)
Ali Alp Erdinc	Worked on Use case diagram 9 + Use case Scenario 9 +Use case 9's sequence diagram + coordinated meeting for review + wrote meeting minutes on github + asked team questions to project coordinator and explained needed fixes to team
Daniel Soldera	Worked on Use Case diagram 3 + Use Case Scenario 3 + Class Diagram
Rohan Das	Use Case scenario 1 + Use case diagram 1 + sequence diagram(1,5, 6, 10)
Arsany Fahmy	Use Case scenario 6 + Use case diagram 6 + worked on sequence diagrams (6, 7, 8, 9)
Julien Fadel	Use Case diagram and scenario number 10. The class diagram with Daniel. Two examples of Gang Of four

7. References

[1]
https://www.edureka.co/blog/session-in-java/#:~:text=Session%20In%20Java,-The%20time%20interval&text=In%20simpler%20terms%2C%20a%20session,technologies%20that%20implement%20session%20tracking.

[2] https://www.javatpoint.com/java-spring-pros-and-cons

[3] https://unstop.com/blog/advantages-and-disadvantages-of-sql