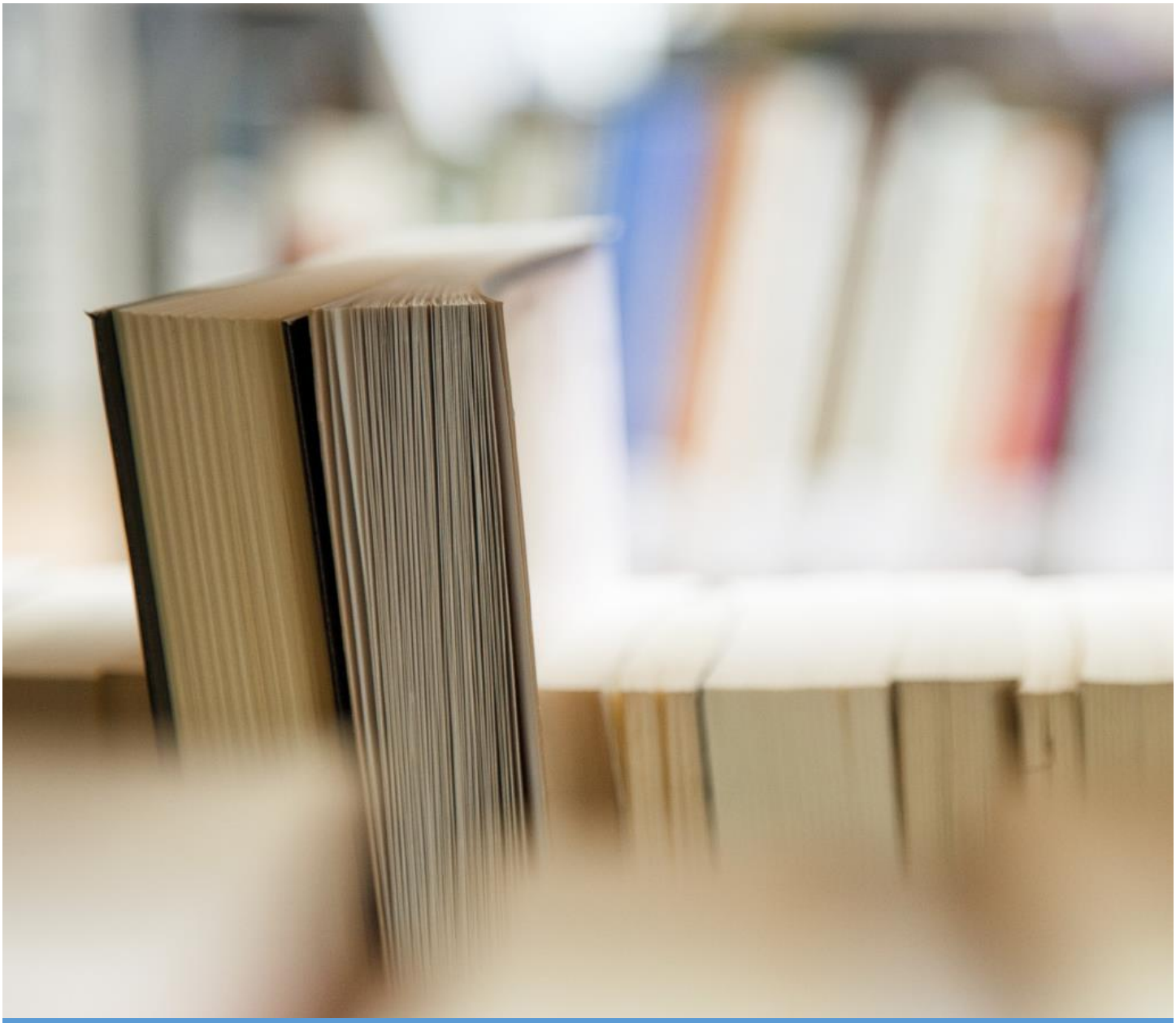


SRS report on: “Expenditure Management System”

11/17/16

Management Information System

Contents

Letter of Transmittal

Dr. Md. Mahbubul Alam Joarder

Professor

and

Amit Seal Ami

Lecturer,

Institute of Information Technology,

University of Dhaka

November 17, 2016

Honorable Sir,

We have submitted the enclosed report on Software Requirements Specification and Analysis of our proposed "Expenditure Management System" for your approval. This report includes all the requirements we have acquired for developing the project.

The primary purpose of this document is creating SRS report for the project what we are doing. This report includes the details of each steps we have followed to collect the requirements.

Sincerely yours,

Md. Abu Bakar Siddique (BSSE 0609)

M.A. Nur Quraishi (BSSE 0615)

Feroze Al Mamun (BSSE 0634)

Munsi Toufiqur Rahman (BSSE 0636)

Md. Nazmul Haque (BSSE 0637)

Institute of Information Technology

University of Dhaka

Enclosure: Software Requirement Specification and Analysis Report

Exclusive Summary

The purpose of our proposed project is to reduce all the manual process for Expenditure Management System. As our proposed system is implemented as a web based portal, it will provide an easy way to access this site from anywhere via internet thus it will reduce time and cost.

Acknowledgement

We are expressing our heartiest gratitude to Almighty ALLAH to complete the designated SRS report in time and without hassles.

We are grateful to our honorable sir Dr. Md. Mahbubul Alam Joarder and Amit Seal Ami for their supervision throughout the working time. They helped us a lot by sharing their invaluable knowledge with us.

Chapter 1

Introduction

1.1 Purpose

This document is the software Requirement Specification (SRS) for "Expenditure Management System". In this document, there contains functional, non-functional and supported requirements that establishes a requirements baseline for developing our proposed project. The requirements having this document are independent, uniquely numbered and embodied by topic. The SRS serves as specifying user requirements to the developer by means of communication and provides a common reference point for both the developer team and stakeholder community. The SRS will evolve enough time as users and developers work together to validate, clarify and expand its contents.

1.2 Intended Audience

This SRS is intended for several audiences including the customers as well as the project managers, designers, developers, and testers.

The customer will use this SRS to verify that the developer team will produce a product which will meet all the requirements specified and approved by the customer.

This SRS document helps the project manager to plan milestones and a delivery date, and ensuring that the developing team is on the right track during developing the system.

The designers will use this SRS as a basis for designing the system. The designers will continually roll back to this SRS to ensure that the system they are designing, will fulfill the customer's requirements

The necessity of using this SRS documentation for the developer is in order to provide a basic for developing the system's functionality. The developers will link the requirements defined in this SRS so that they ensure that they are producing a software which meets all the requirements defined in SRS as satisfying the customer requirements.

SRS document provides an easy way for the tester to perform their task with more flexibility because in SRS document requirements must be clear and specific with no uncertainty, requirements should be measurable in terms of specific values, requirements should be testable having some evaluation criteria for each requirement, and requirements should be complete, without any contradiction.

Chapter 2

Inception

2.1 Introduction

The goal of the inception phase is to achieve concurrence among all stakeholders on the lifecycle objectives for the project. The inception phase is of significance primarily for new development efforts, in which there are significant business and requirements risks which must be addressed before the project can proceed.

Inception is the beginning phase of requirements engineering. It defines how does a software project get started and what is the scope and nature of the problem to be solved. To establish the groundwork, we have worked with the following factors related to the inception phases:

I. Identifying Stakeholders
II. Recognizing multiple viewpoints
III. Working towards collaboration
IV. Asking the First Questions

2.1.1 Identifying stakeholders

Stakeholders refer an individual, group, or organization, who may affect, be affected by, or perceive itself to be affected by a decision, activity, or outcome of a project. Stakeholders include end-users who interact with the system and everyone else in an organization that may be affected by its installation. To identify the stakeholders, we consulted with an employee and asked her following questions:

- Who is paying for the project?
- Who will be using the project outcomes?
- Who gets to make the decisions about the project (if this is different from the money source)?
- Who has resources I need to get the project done?
- Whose work will my project affect? (During the project and also once the project is completed).

By analyzing the requirements, we select two stakeholders for our proposed “Expenditure Management System” project. These stakeholders are teachers and the director.

1. Director: The director will add specific sectors of expense along with field names of expense. He also assigns each teacher as the sub admin of the system.
2. Teacher: Each teacher proposes an amount of expense corresponding to the assigned fields of the assigned sectors.

2.1.2 Asking the first question

We predefine a set of context-free questions focuses on the customer and other stakeholders, overall project goals and benefits. The questions are mentioned above. These questions have helped us to identify all stakeholders, measurable benefit of the successful implementation and possible alternatives to custom software development. Next set of questions are helped us to gain a better understanding of problem and allows the customer to voice his or her perception about the solution. The final set of question focused on the effectiveness of the communication activity itself.

2.1.3 Recognizing multiple viewpoints

We gather these view points by discussing with our selected stakeholders including Course Teacher and student.

1) Director
    i) Accessing system from any device that has internet access
    ii) Restricting access to functionality of the system
    iii) Approving or rejecting the proposal of the sub-admins
    iv) Generating annual report
    v) Controlling the budget

2) Teacher
    i) Proposing an amount of expense
    ii) Easy access to the system

2.1.4 Working towards collaboration

Every stakeholder has their own requirements. We followed following steps to merge these requirements:

1. Identify the common and conflicting requirements
2. Categorize the requirements
3. Take priority points for each requirement from stakeholders and on the basis of this voting prioritize the requirements
4. Make final decision about the requirements

Common Requirements:

- Web-Based Interfaces.
- The application can be accessed from any computer that has Internet access.
- Easy Access.
- Maintain a database of all information.

Conflicting Requirements:

- Easy access and Strong Authentication.
- Don't allow access to any authenticated user.

Final Requirements:

We finalized following requirements for the system by categorizing and prioritizing the requirements:

- Error free system (Level of tolerance 5% that means maximum 5% error may be considerable).
- Web-based interfaces.
- Allow valid users to login and logout.
- Restrict access to functionality of the system based upon user roles.
- Allow administrators of the system to change users and configure parameters of the system.
- Maintain a database of all information.

2.2 Conclusion

Inception phase helped us to establish basic understanding about our proposed "Expenditure Management System" project; identify the people who will be benefited if the expenditure management process becomes automated, define the nature of the "Expenditure Management System" software and establish a preliminary communication with our stakeholders.

Chapter 3

Elicitation

3.1 Introduction

Requirement elicitation is the process of discovering, reviewing, documenting, and understanding the user's needs and constraints for the system. Elicitation is a step of requirement engineering that helps the customer to define what is required. To complete the elicitation step we face many problems like problems of scope, problems of volatility and problems of understanding. However, this is not an easy task. To overcome these problems, we have worked with the Eliciting requirements activity in an organized and systematic manner.

3.2 Eliciting requirements

Unlike inception where Q&A (Question and Answer) approach is used, elicitation makes use of a requirements elicitation format that combines the elements of problem solving, elaboration, negotiation, and specification. It requires the cooperation of a group of end-users and developers to elicit requirements. To elicit requirements, we completed following four works.

- Collaborative Requirements Gathering
- Quality Function Deployment
- Usage Scenarios
- Elicitation work products

3.3 Collaborative requirements gathering

Many different approaches to collaborative requirements gathering have been proposed. Each makes use of a slightly different scenario. We completed following steps to accomplish this task.

- The meetings were conducted with Teachers. He was questioned about the requirements and expectations from the automated “Expenditure Management system”.
- He was asked about the problems he is facing with the current manual system. At last we selected our final requirement list from covering the meetings.

3.4 Quality function deployment

Quality Function Deployment (QFD) is a technique that translates the needs of the customer into technical requirements for software. It concentrates on maximizing customer satisfaction from the Software engineering process. With respect to our project the following requirements are identified by a QFD**.**

3.4.1 Normal requirements

Normal requirements consist of objectives and goals that are stated during the meeting with the customers. Normal requirements of our project are: -

- Allow valid user to log in and log out to the system.
- Restrict access to functionality of the system based upon user roles
- Allow valid users that log in to renew items, reserve items, and view the items
- The application is accessible from any computer via internet.

3.4.2 Expected Requirements

- Selectable fields wherever possible instead of fields that require the user to type in data.
- Easy operation of the system

3.4.3 Exciting requirements

These requirements go beyond the customer's expectations and prove to be very satisfying when present.

- The user interface should provide appropriate error messages for invalid input as well as tool-tips and online help.
- The user interface should follow standard web practices such that the web interface is consistent with typical internet applications.
- Offer log in with smart phone and tablets.
- The system's configuration shall be documented and updated as changes to the system are made due to patches, new releases, new Changes etc.

3.5 Usage Scenario

Expenditure Management System (EMS)

Expenditure Management System is a system for IIT, University of Dhaka to manage and record each transaction related to the expense in various sectors. The system is administrated by the Director of IIT, University of Dhaka. The system is also driven by a set of teachers who is assigned by the Director in one or many sectors of expense.

First, the Director needs to add specific sectors of expense. Then, he/she adds various field names of expense under each sector with a unique field access key and estimated amount collected from the annual budget report. At the beginning of the year, the Director, super admin of the system assigns each teacher as the sub admin of the system with a unique username, password, one or

many sector names. The director also needs to add one or many field access keys under each sector name.

After successful addition of the sub admins, each sub admin proposes an amount of expense corresponding to the assigned fields of the assigned sectors. Then their proposal is sent to the Director for approval. If the Director approves it then, the amount will be added to the proposed field of the proposed sector. If he/she rejects the proposal, then the proposal will be discarded and a notification is sent to the sub admin about the rejected proposal.

At the end of the year, the system will produce an annual report to the director mentioning the spent amount of each fields of each sector. If the total spent amount is below 75% or over 100% of the total estimated amount then, a special report is generated to the Director by the system mentioning the over spent fields of each sectors. The director can remove the sub admins in order to prepare the system again.

[The Director must be authenticated too]

3.6 Elicitation work product

The output of the elicitation task can vary depending on size of the system or product to be built. Our elicitation work product includes:

- A statement of our requirements for automated Online Student Registration and Exam system.
- A bounded statement of scope for our proposed system.
- A list of customers, users and other stakeholders who participated in requirement specification.
- Set of usage scenarios.
- Description of the system's technical environment.

Chapter 4

Scenario Based Modeling

This chapter describes scenario based modeling of Expenditure Management System.

4.1 Definition of Use Case

A use case is a software and system engineering term that describes how a user uses a system to accomplish a particular goal. A use case acts as a software modeling technique that defines the features to be implemented and the resolution of any errors that may be encountered.
Use cases define interactions between external actors and the system to attain to particular goals.
There are three basic elements that make up a use case:
1. Actors: Actors are the type of users interact with the system.
2. System: Use cases capture functional requirements that specify the intended behavior of the system.
3. Goals: Use cases are typically initiated by a user to fulfill goals describing the activities and variants involved in attending the goal.

Use cases are modeled using unified modeling language and are represented by ovals containing the names of the use case. Actors are represented using lines with the name of the actor written below the line. But here, we use the combination of lines and a circle to represent each actor. To represent an actor's participation in a system, a line is drawn between the actor and the use case. Boxes around the use case represent the system boundary.
There are two types of actor:
1. Primary actor.
2. Secondary actor.

Primary actor: Primary actor refers who is directly involved with the system in order to achieve required function and benefit from the system. They interact directly and frequently with the software.
In our proposed system, students and teachers are primary actor.
Secondary actor: Secondary actor refers who is indirectly involved with the system but necessary to support the system so that system can perform its functionality without any hinder. Secondary actor either produces or consumes information.
In our proposed system, there is no secondary actor.

4.2 Use Case Diagrams

Use case diagram provides the non-technical view of the system. A use case diagram is a graphic depiction of the interactions among the elements of a system. A use case is a methodology used in system analysis to identify, clarify and organize system requirements.

4.2.1 Level-0 of the use case:

By analyzing the usages scenario, we select two primary actors. Figure 4.2.1 indicates that these two actors are directly interacted with the hole system. Basically the whole system directly depends on these two actors. But we cannot find any secondary actor by analyzing the usages scenario.
Here, these two actors who are directly involved with our proposed system:
1. Director
2. Teacher

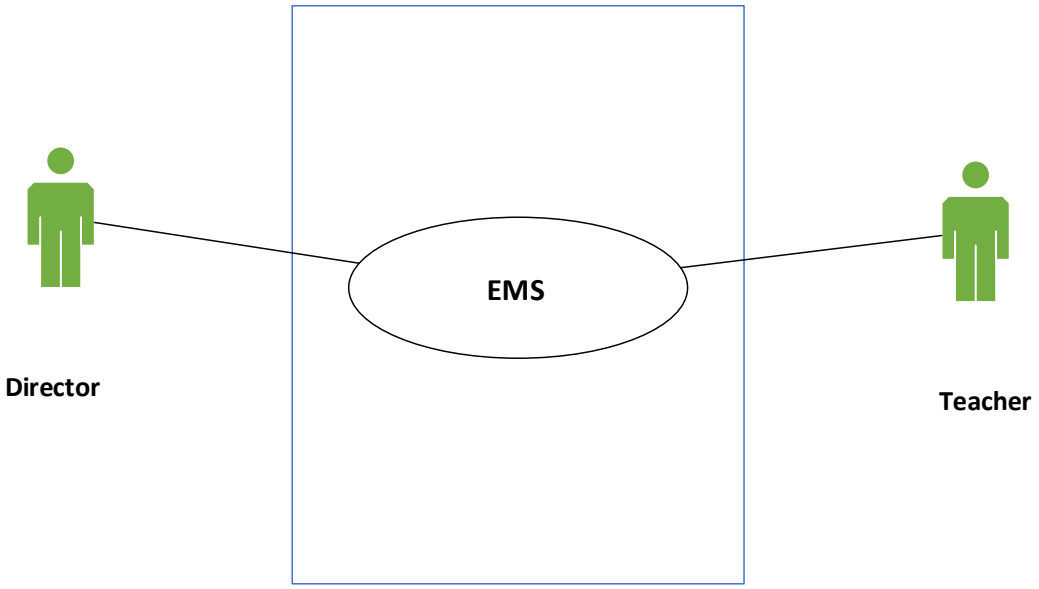


Figure 4.2.1: Level 0 of the use case

4.2.2 Level-1 of the use case:

The proposed system is divided into five subsystems. These subsystems are registration, authentication, addition, expense management and report. Here expense management, addition and report are the heart of this proposed project and authentication subsystem is used only for protecting this system from unauthorized user.

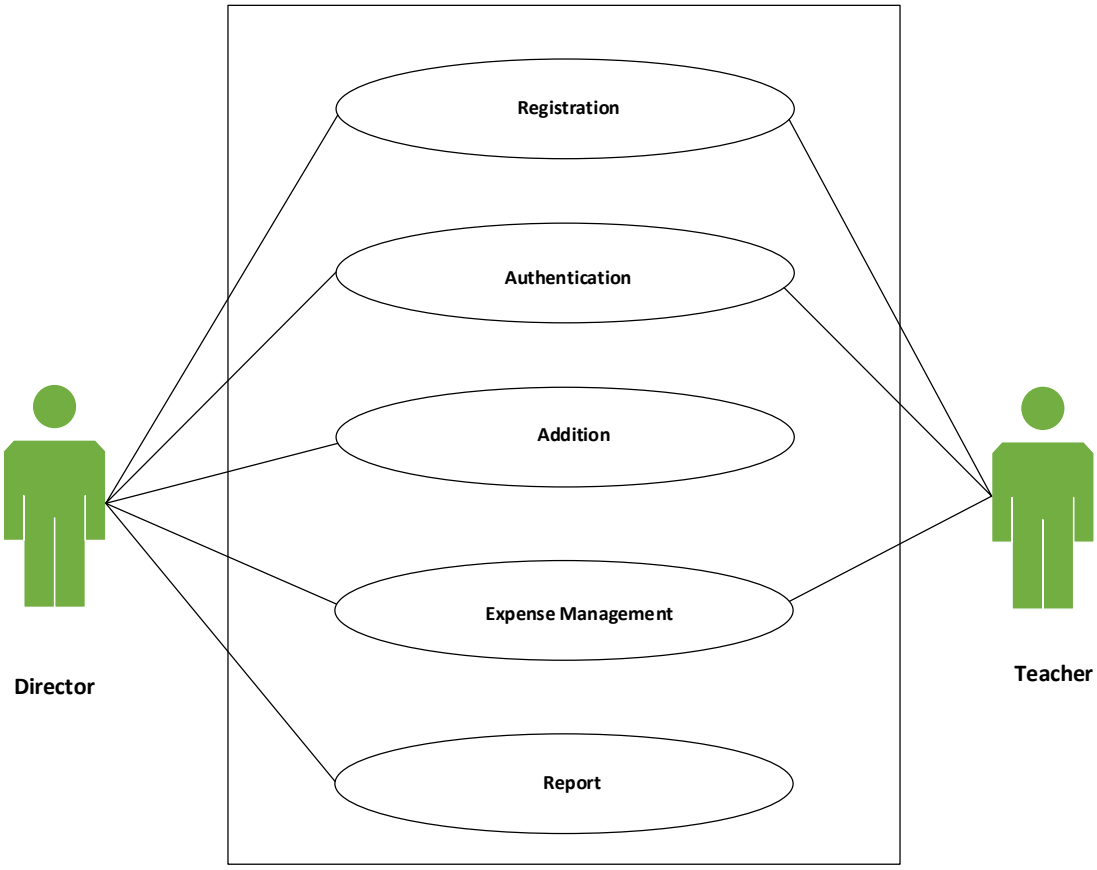


Figure 4.2.2: Level-1 of the use case

4.2.3.1 Level-2 of the “Registration” subsystem:

The registration subsystem is separated into two actions. These actions are registered as teacher and registered as director. These actions are registered as teacher and registered as director. Teachers are involved with the second action and director is interacted with the first action. Action registered as teacher allows any user to fill up form as a teacher of the proposed system. Registered as director allows the director to fill up form as a director.

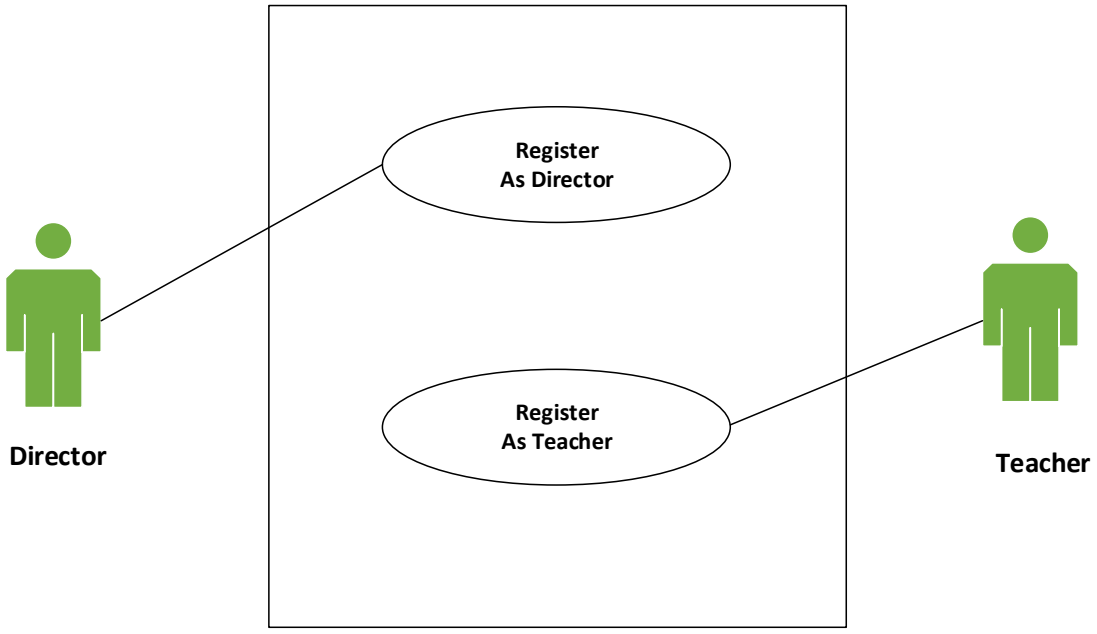


Figure 4.2.3.1: Level-2 of the “Registration" subsystem

4.2.3.2 Level-2 of the “Authentication” subsystem:

The authentication subsystem is divided into three actions. These actions are log in, log out and maintain profile. Both teachers and director can be interacted with all for actions specified at authentication subsystem. Action log in is used to enter the system, action log out is used to exit from the system and maintain profile is to change the profile information.

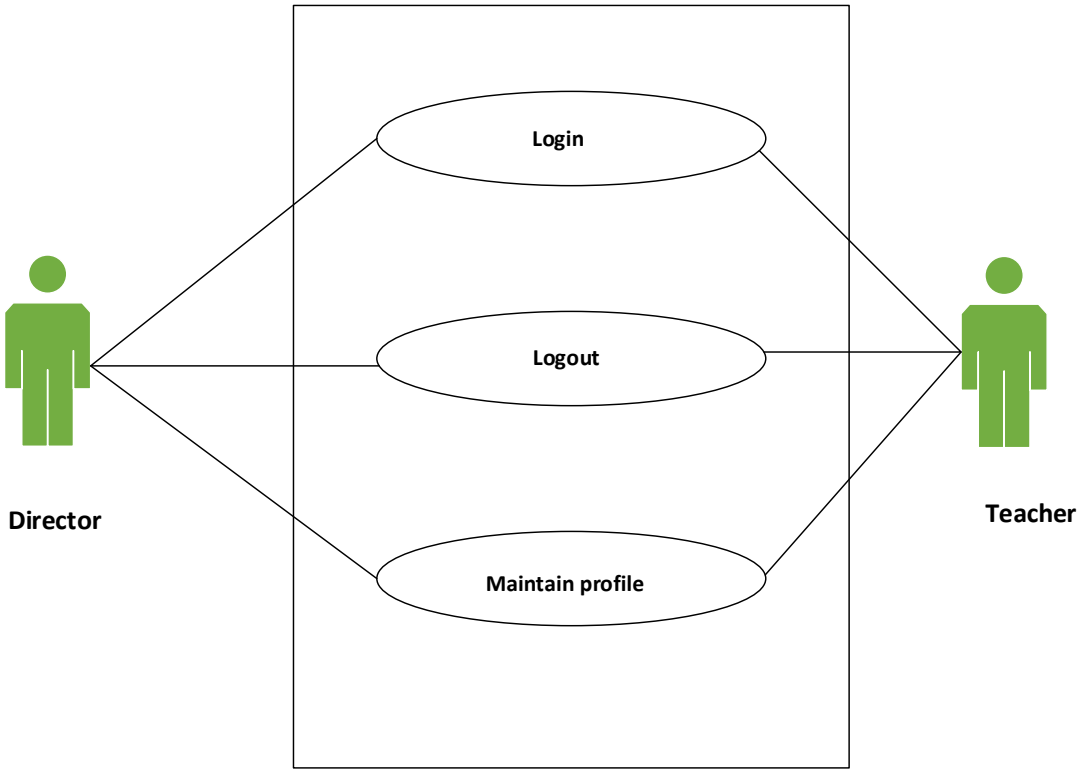


Figure 4.2.3.2: Level-2 of the “Authentication” subsystem

4.2.3.3 Level-2 of the “Addition” subsystem:

This subsystem is divided into two parts. The first one is adding sector of expenses and adding sub-admin. A number of sectors will be added in the expense sectors. Teachers will be added as sub-admin of the subsystem.

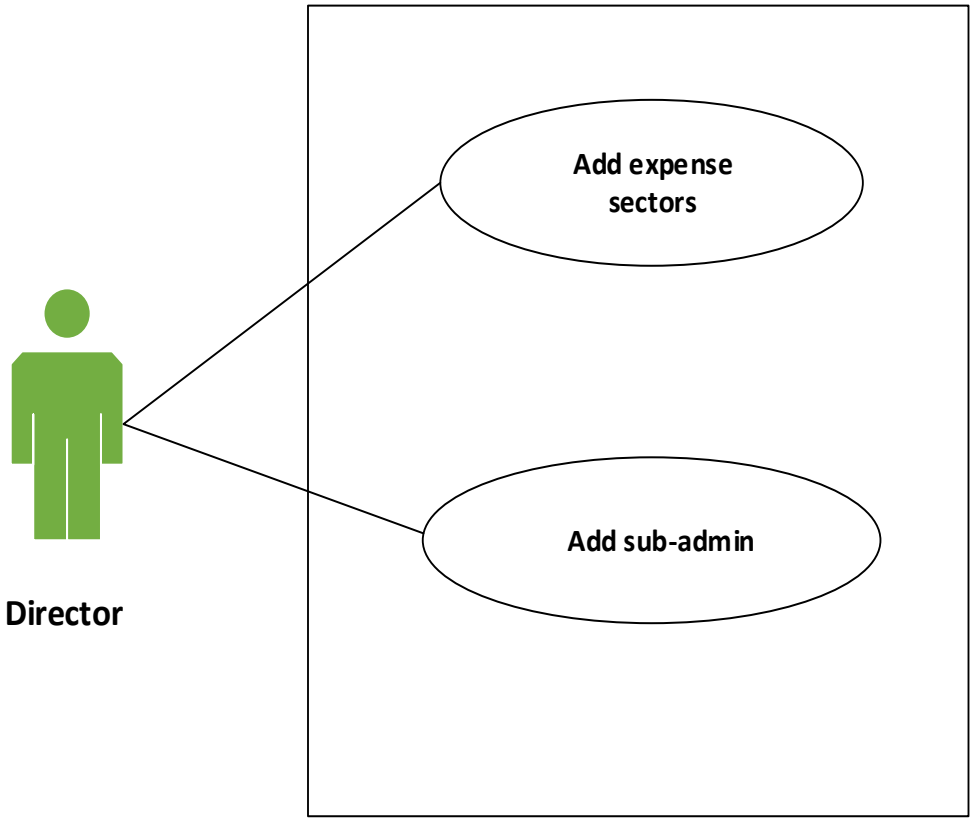


Figure 4.2.3.3: Level-2 of the “Addition” subsystem

4.2.3.4 Level-2 of the “Expense management” subsystem:

This subsystem is divided into three parts. Estimation, decision and notification. The sub-admins proposed a certain amount in the estimation. Director may approve or reject the proposal which is illustrated in decision part. A notification will be sent to the sub-admin if the proposal is rejected.

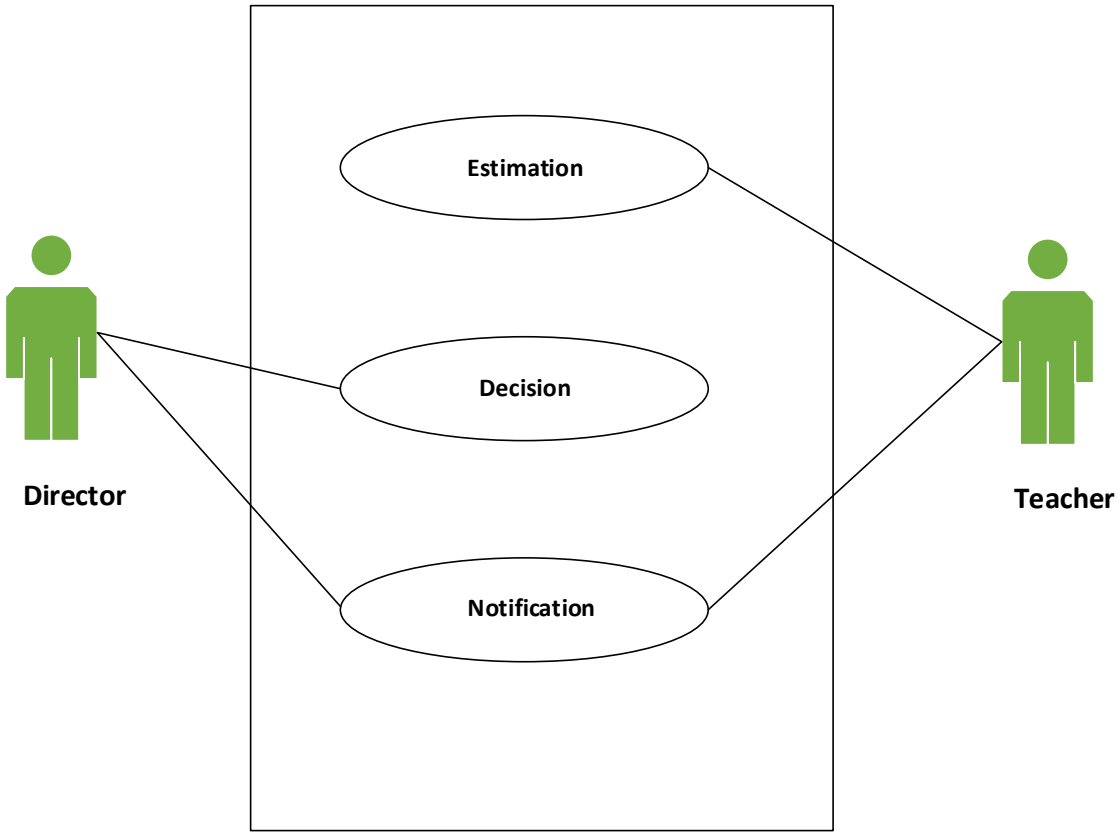


Figure 4.2.3.4: Level-2 of the “Expenditure management” subsystem

4.2.3.5 Level-2 of the “Report” subsystem:

This subsystem is divided into three parts. Expense record, general report and special report. If a notification has to be sent to the admin upon the expense, then a special report is generated.

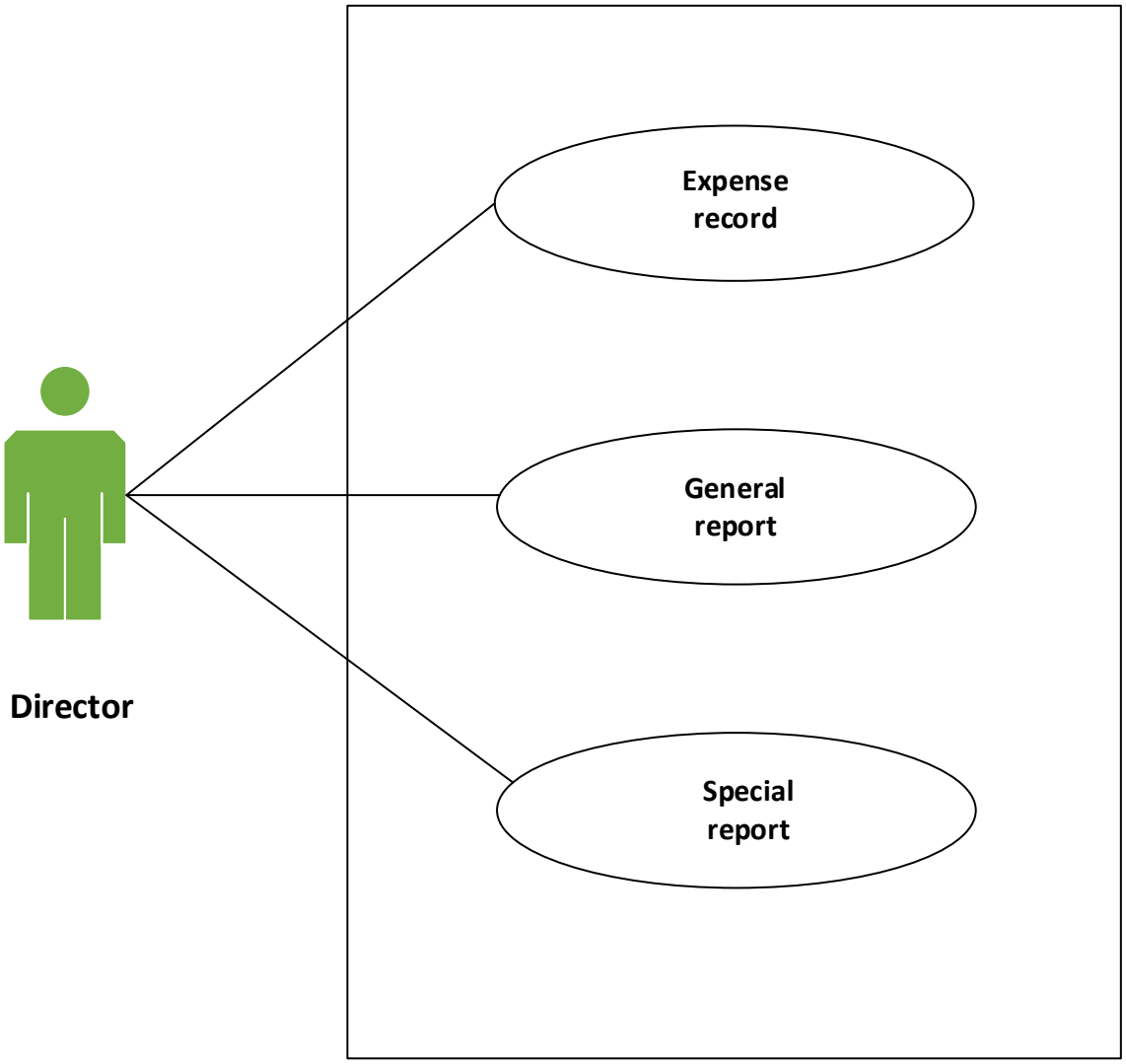


Figure 4.2.3.5: Level-2 of the “Report” subsystem

4.2.4.1 Level-3 of the “Add expense sectors” subsystem:

This subsystem is divided into two parts. Add field names and add field access keys.

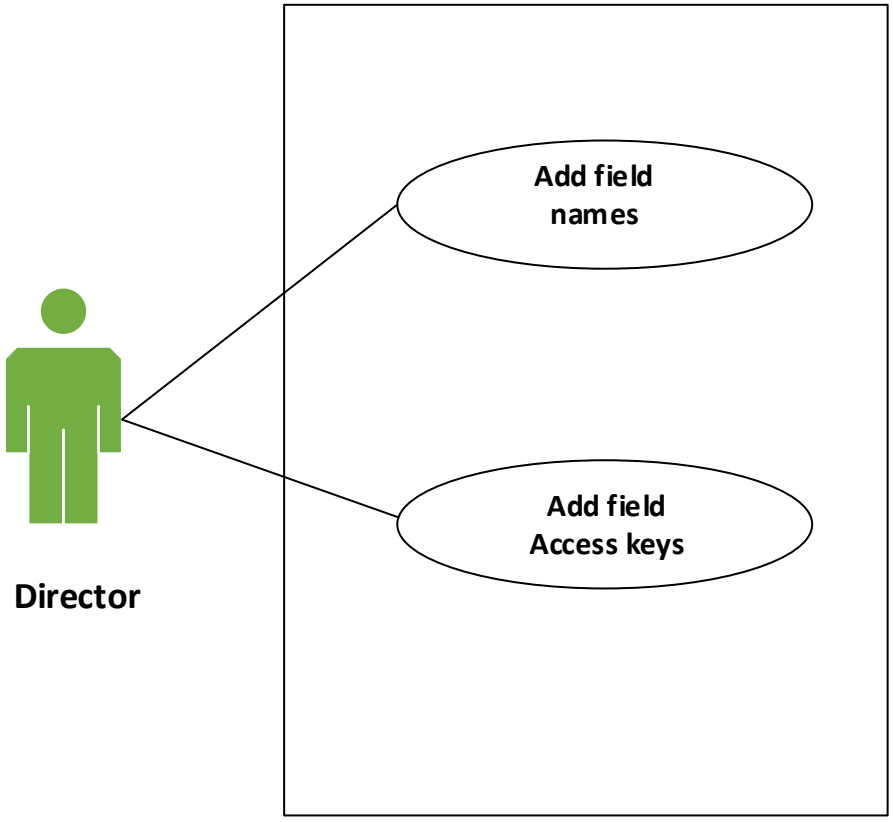


Figure 4.2.4.1: Level-3 of the “Add expense sectors” subsystem

4.3 Activity and Swim Lane Diagram

Activity diagram shows the technical view of the system for every use case from which we can understand how the system actually works and how the actors interact with the system.
Here is the activity diagram for "Registration" use case:

Use case: Registration

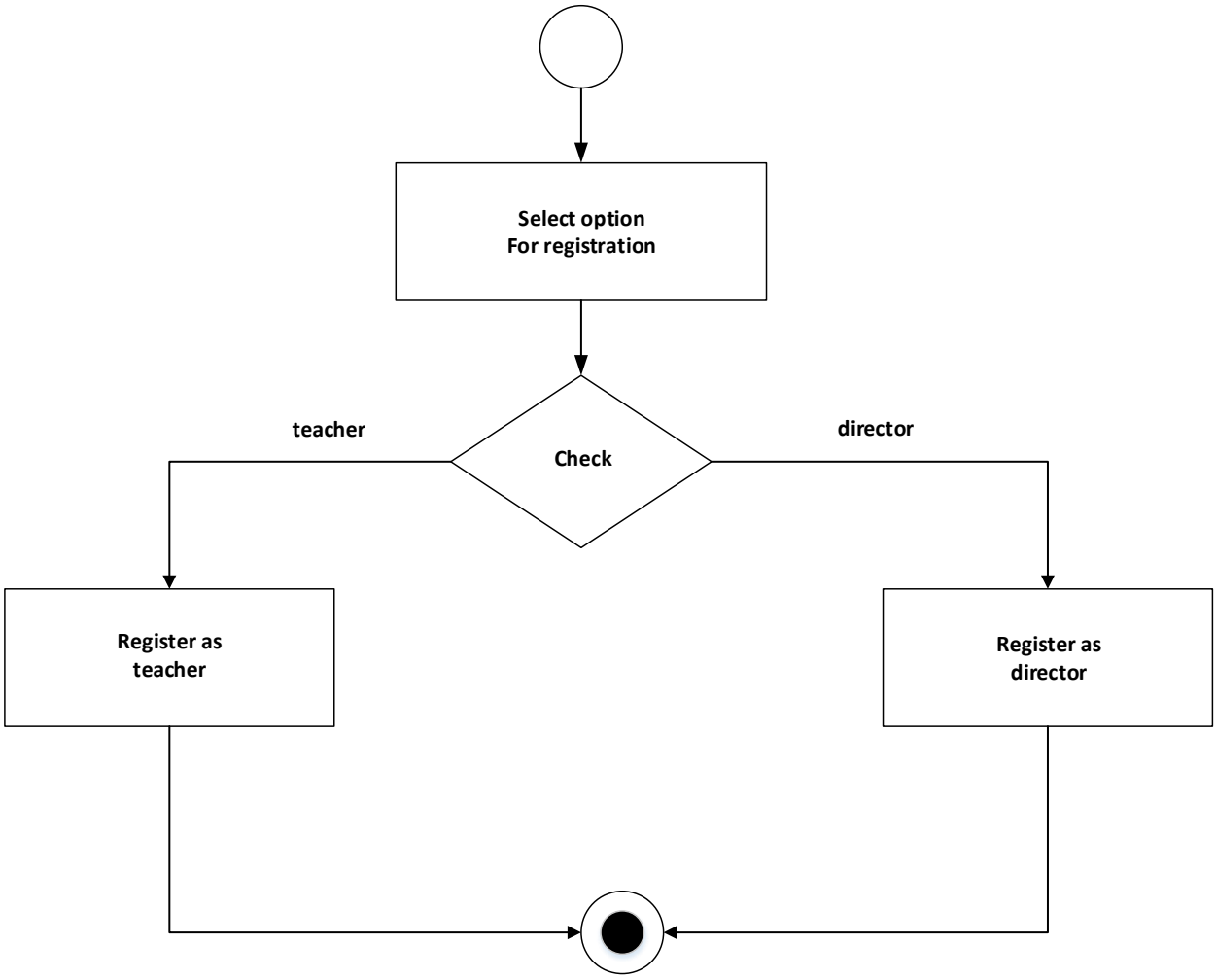


Figure 4.3.1: Activity Diagram for "Registration" use case

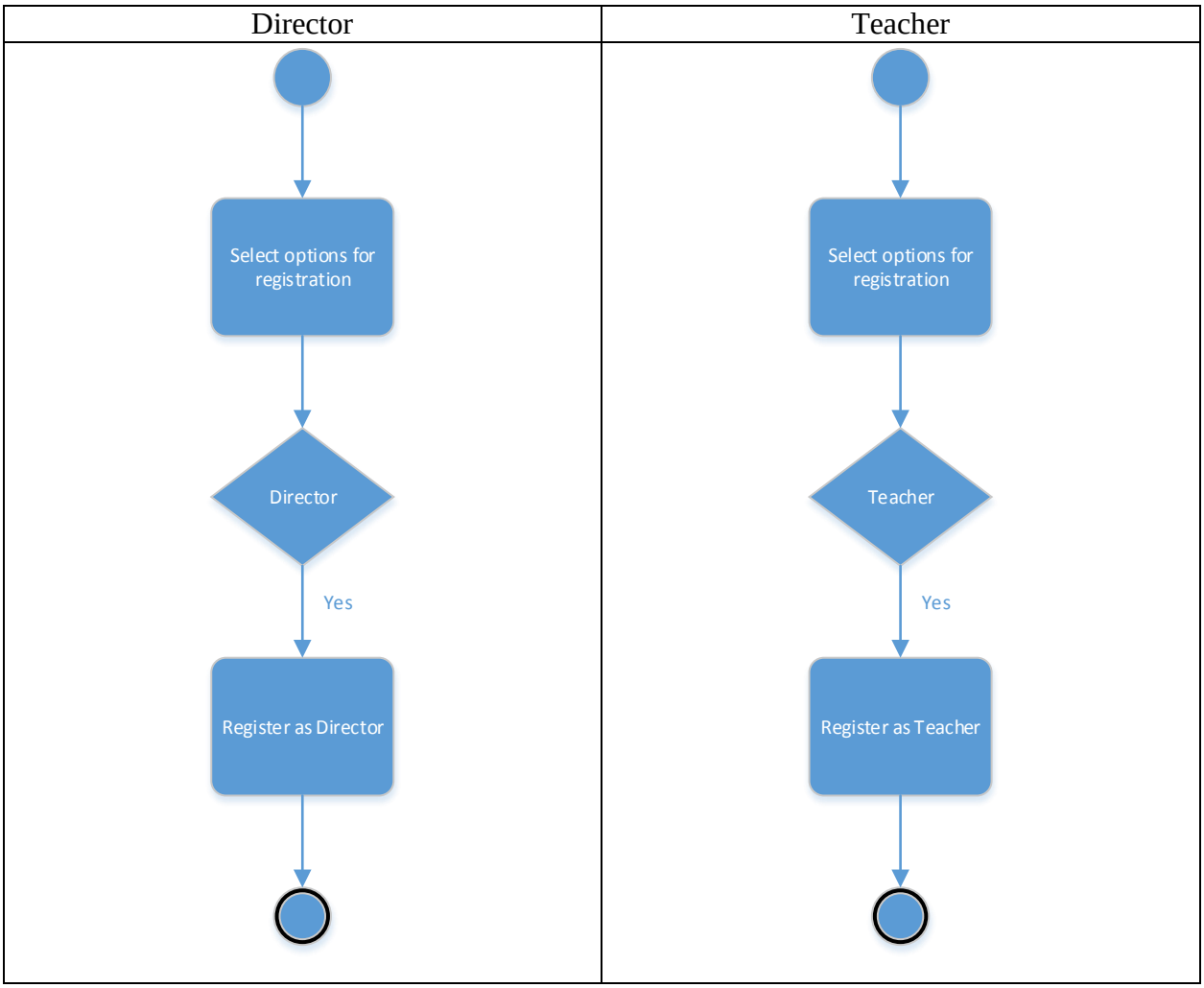


Figure 4.3.2: Swim Lane Diagram for “Registration” use case

Use Case: Log in

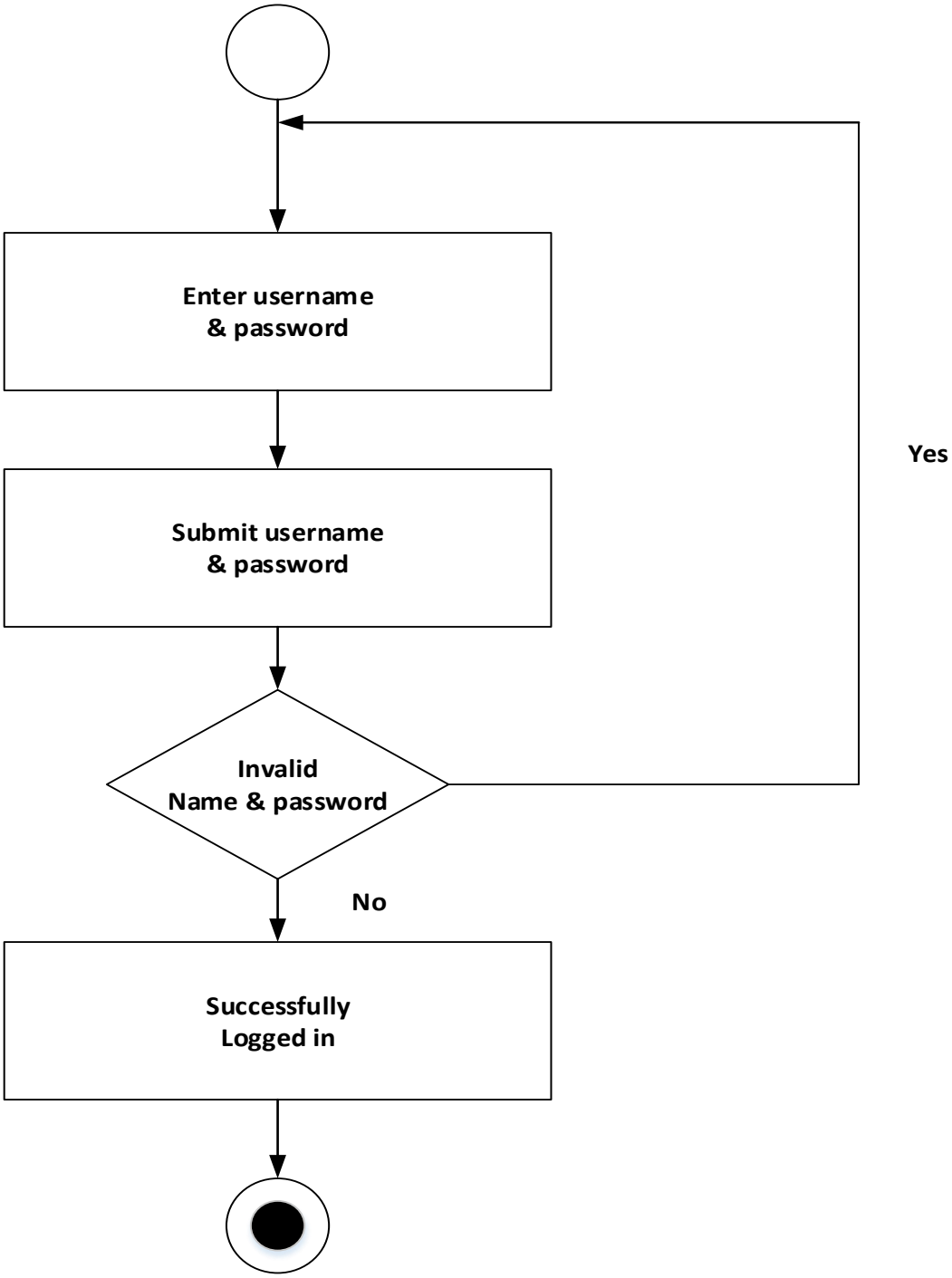


Figure 4.3.3: Activity Diagram for “Log in” use case

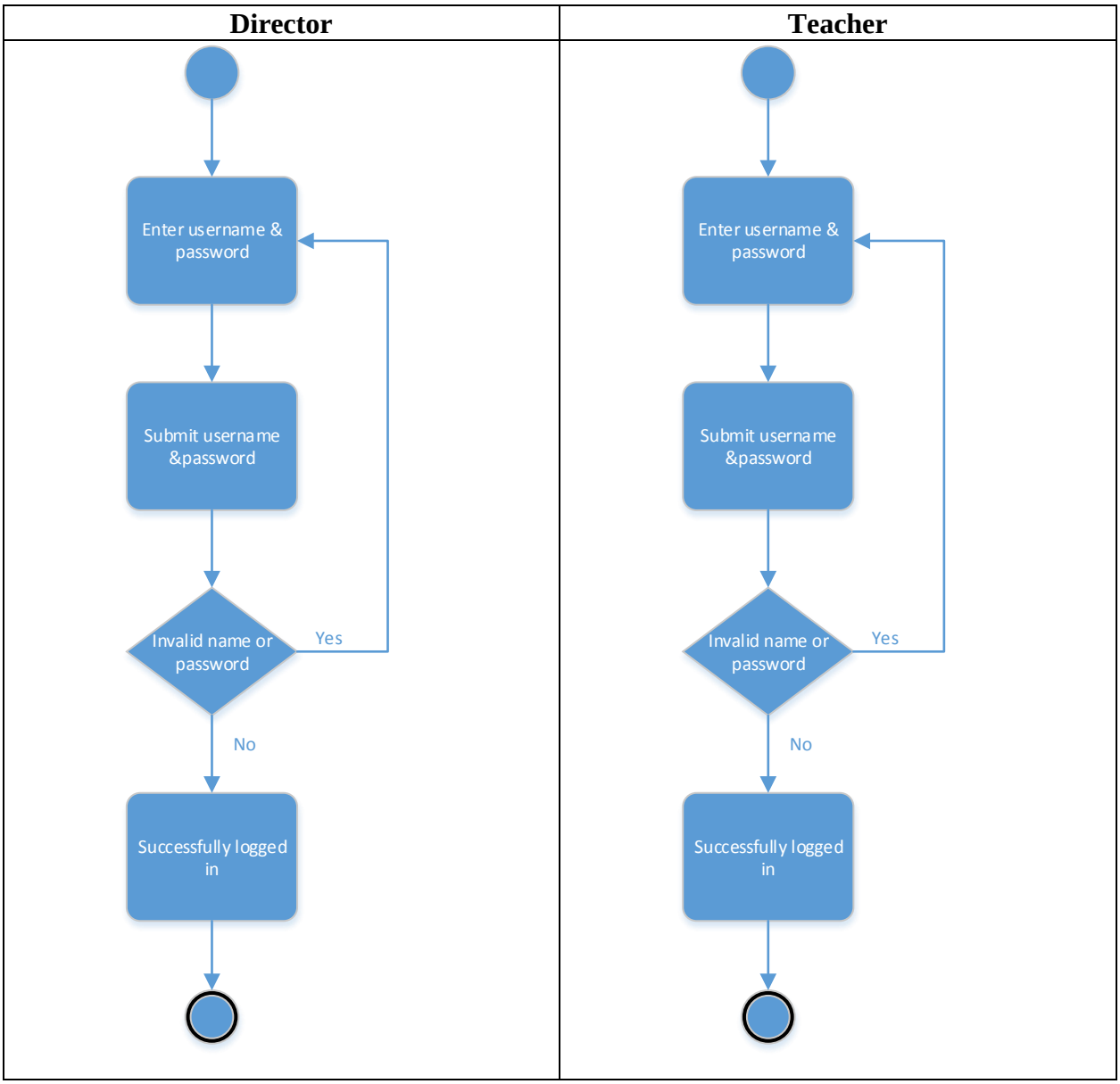


Figure 4.3.4: Swim Lane Diagram for “Log in” use case

Use Case: Log out

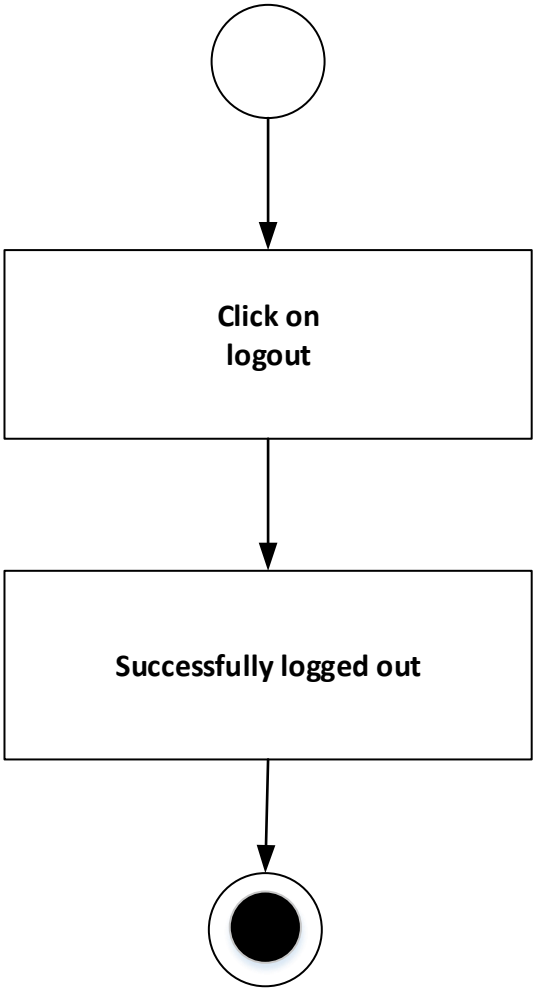


Figure 4.3.5: Activity Diagram for “Log out” use case

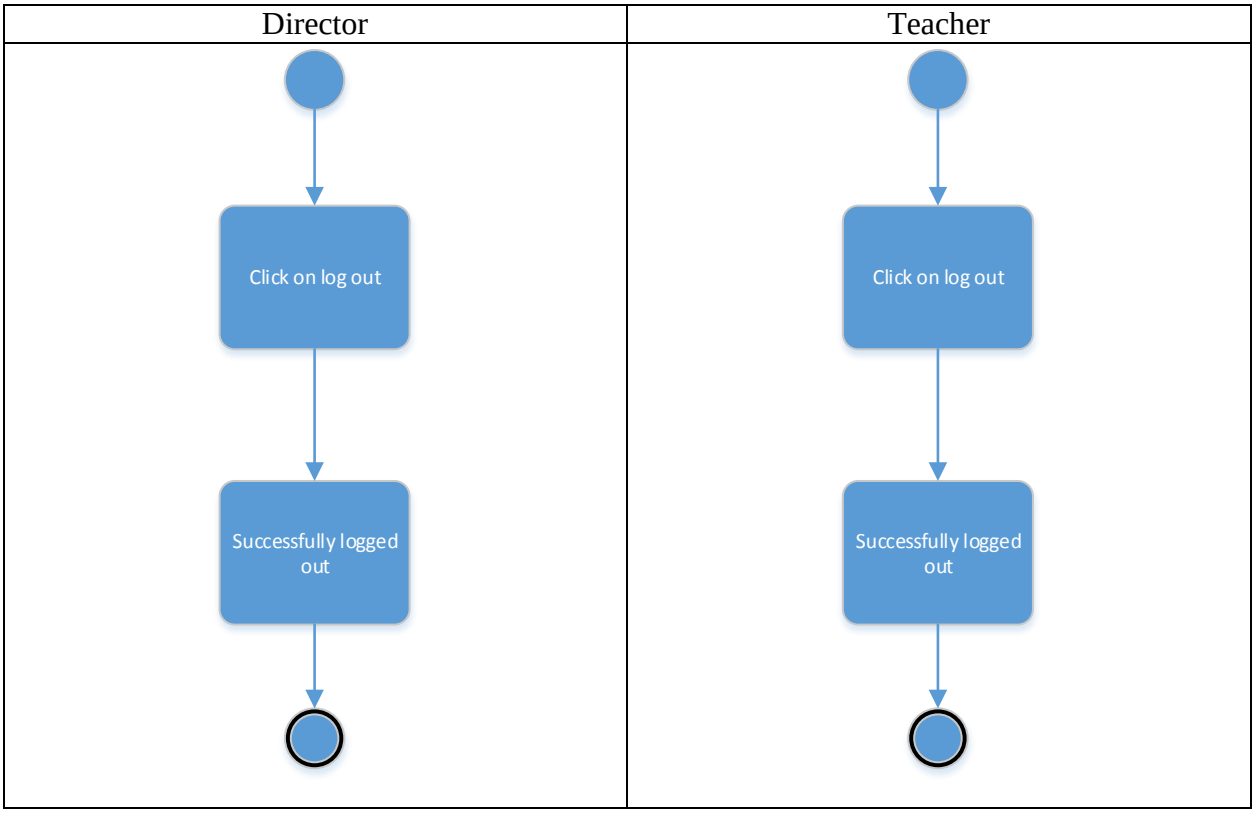


Figure 4.3.6: Swim Lane Diagram for “Log out” use case

Use Case: Maintain profile

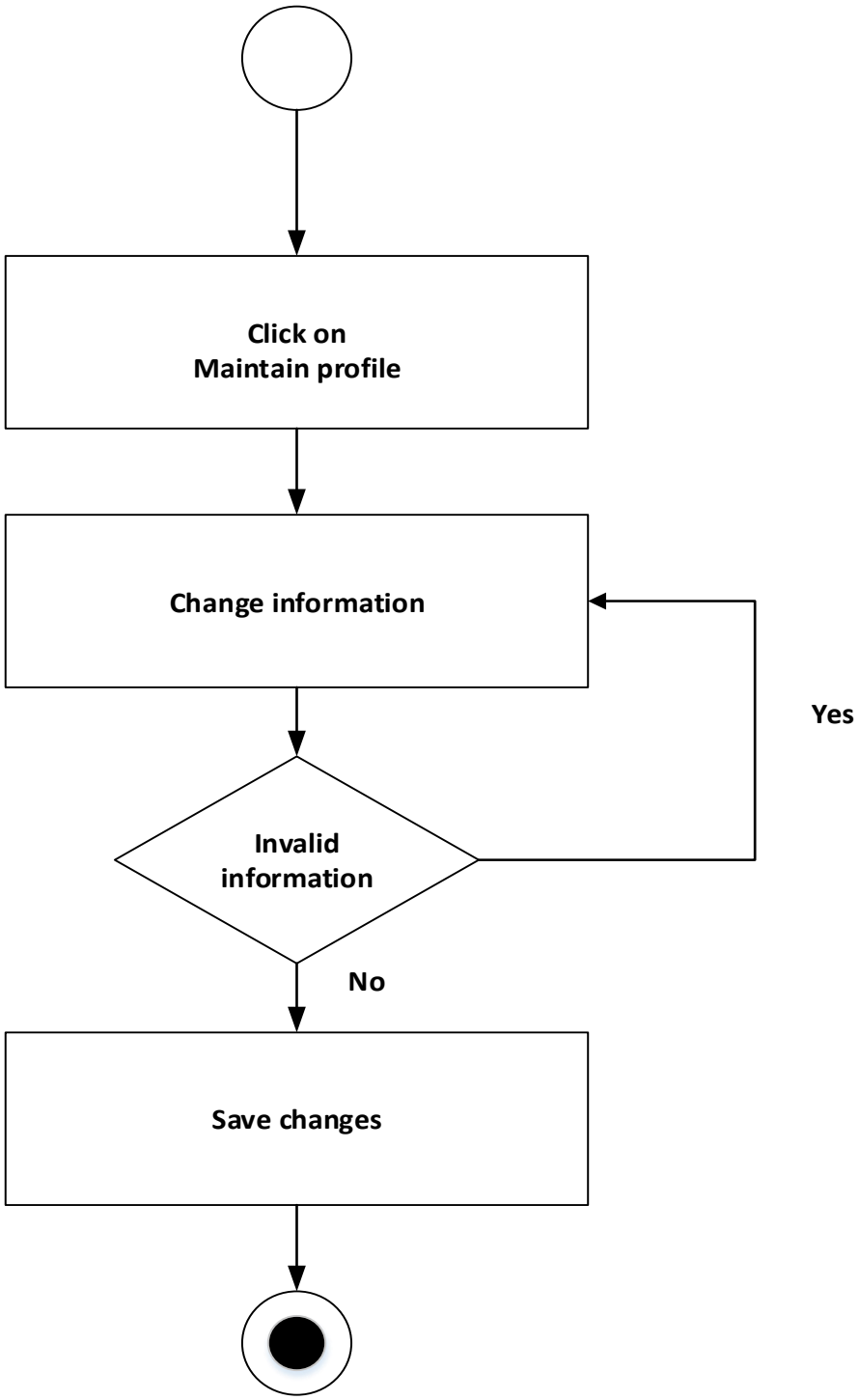


Figure 4.3.7: Activity Diagram for “Maintain profile” use case

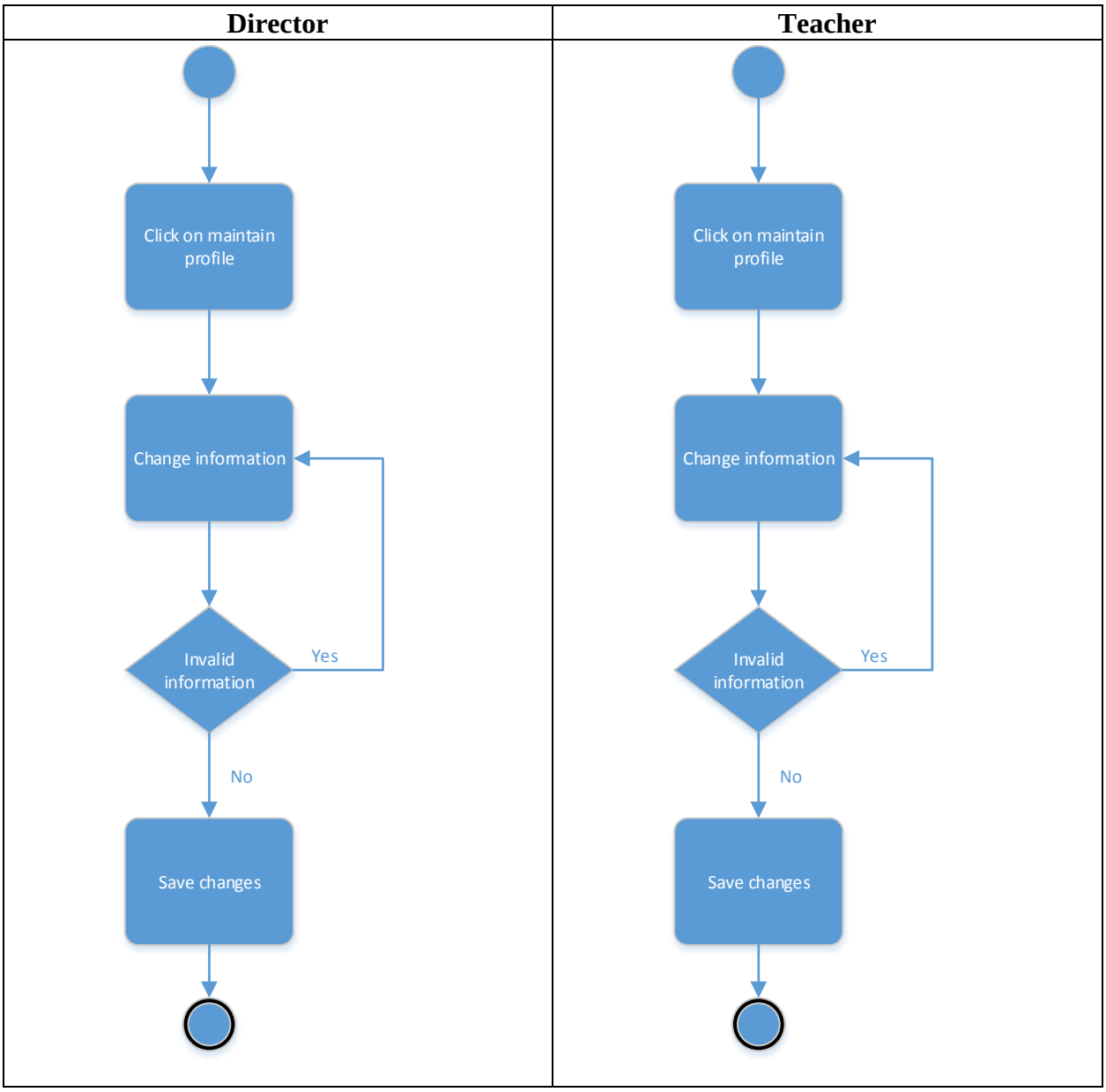


Figure 4.3.8: Swim Lane Diagram for “Maintain profile” use case

Use Case: Add expense sector

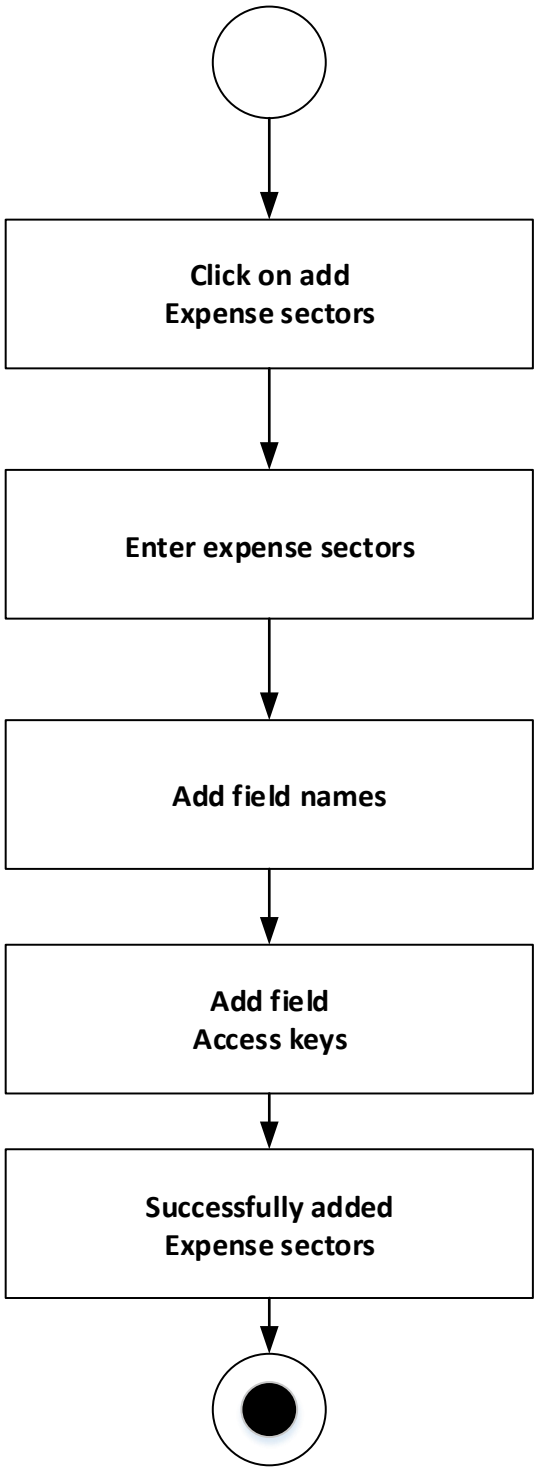


Figure 4.3.9: Activity Diagram for “Add expense sector” use case

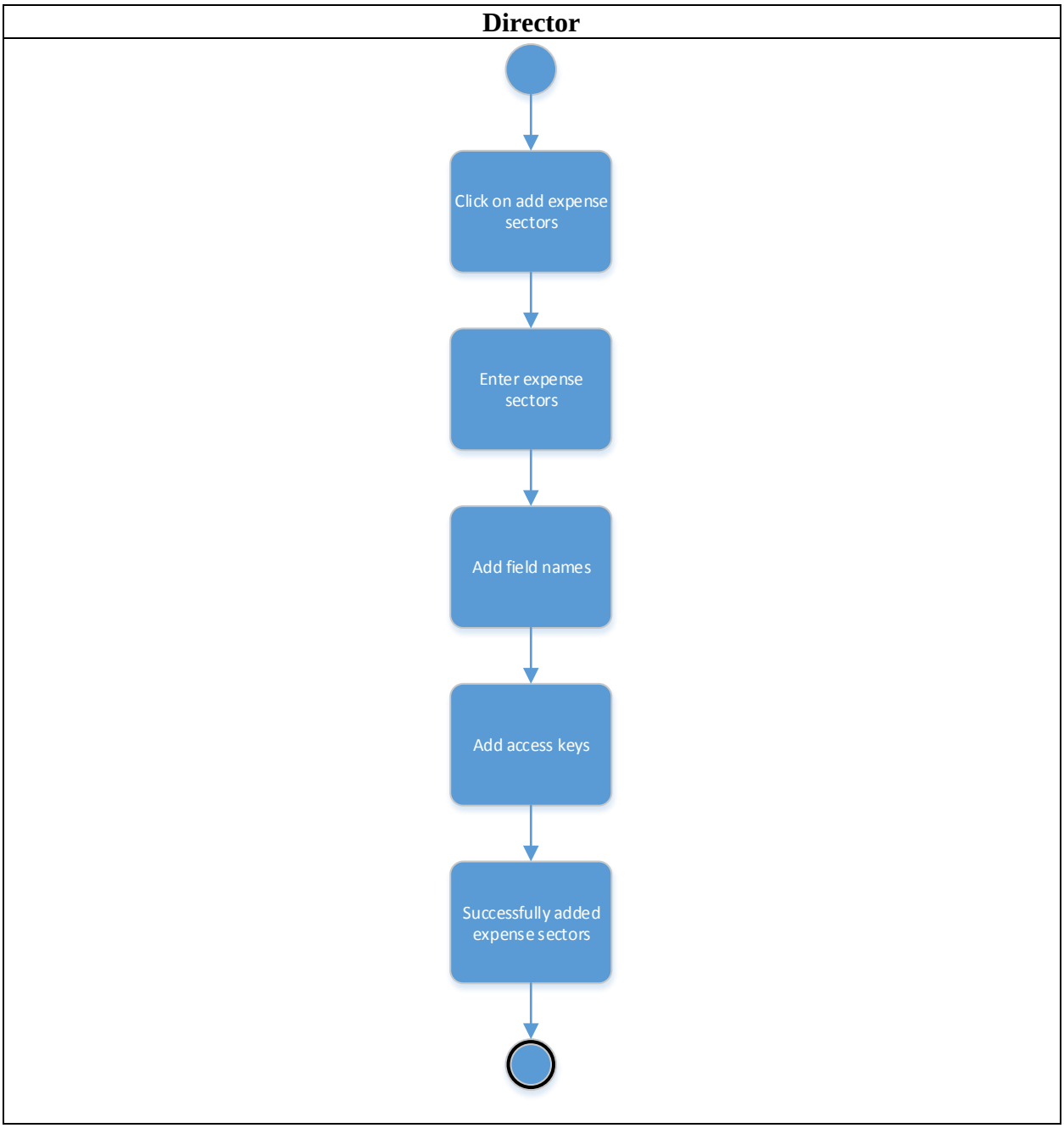


Figure 4.3.10: Swim Lane Diagram for “Add expense sector” use case

Use Case: Add sub-admin

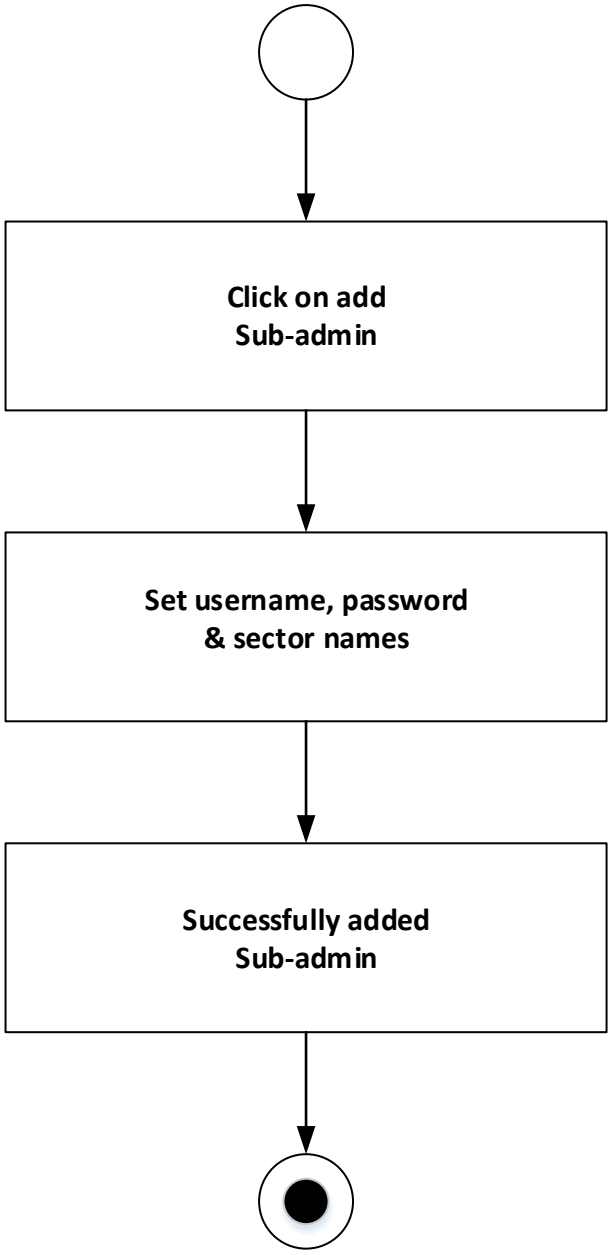


Figure 4.3.11: Activity Diagram for “Add sub-admin” use case

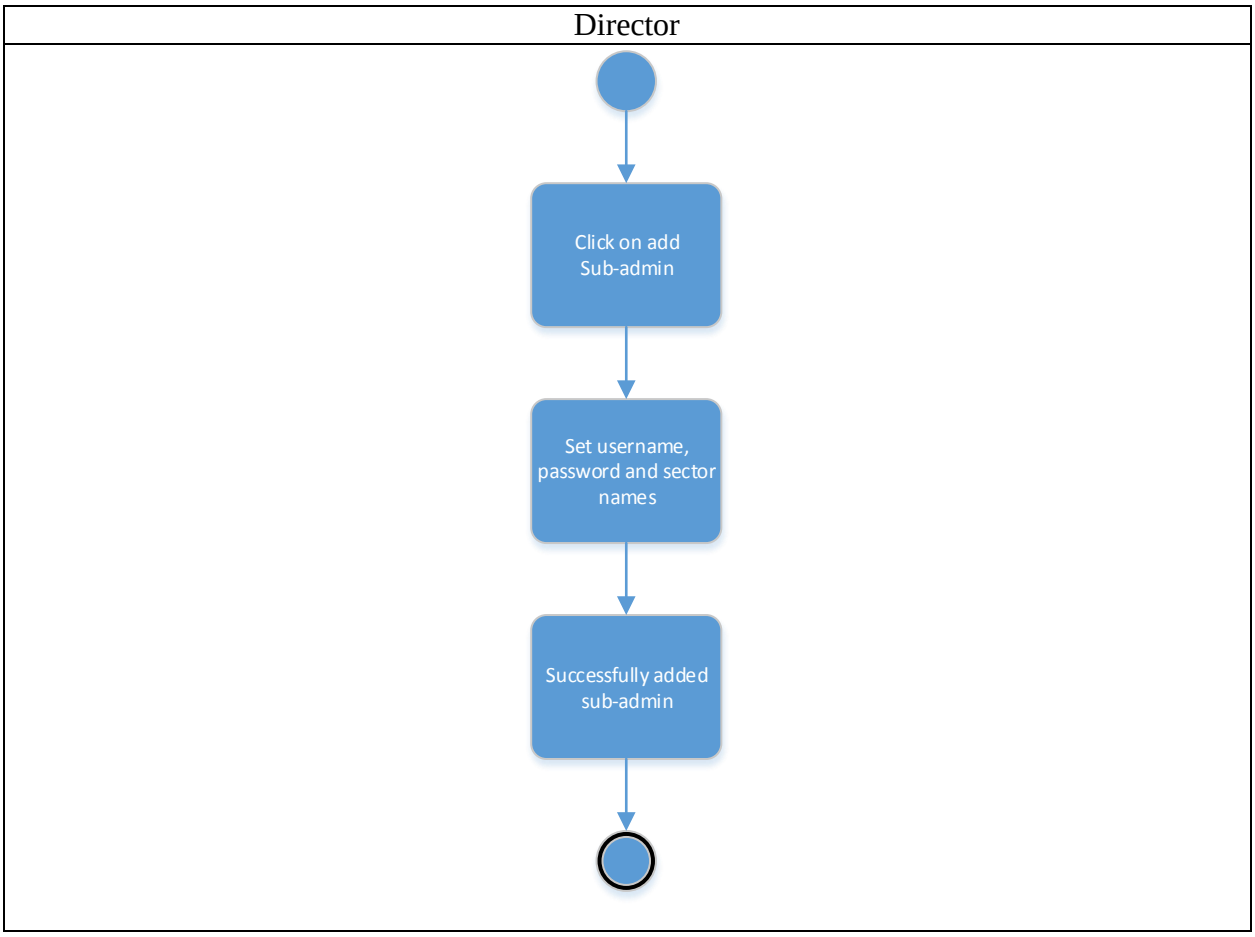


Figure 4.3.12: Swim Lane Diagram for “Add sub-admin” use case

Use Case: Expenditure management

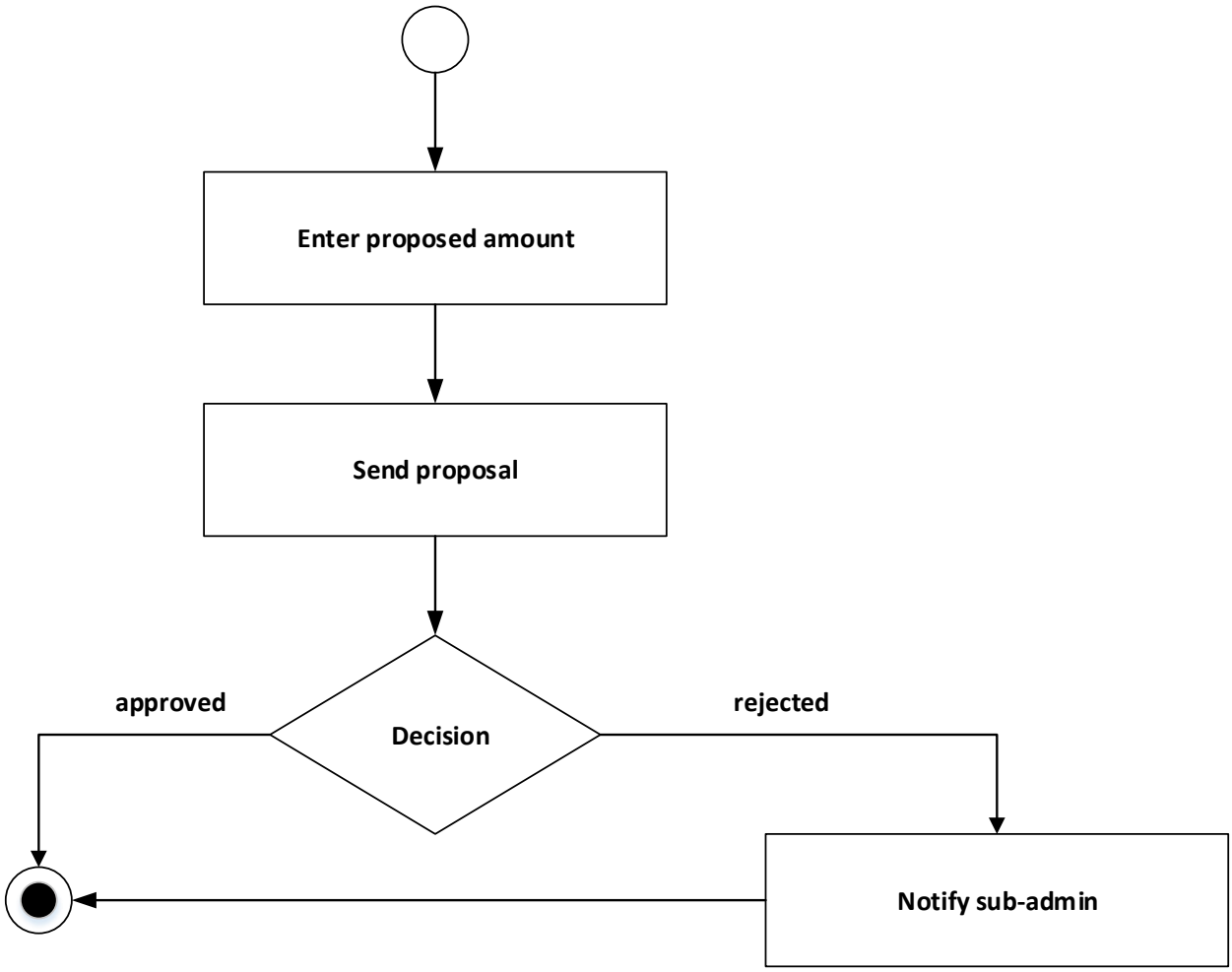


Figure 4.3.13: Activity Diagram for “Expenditure management” use case

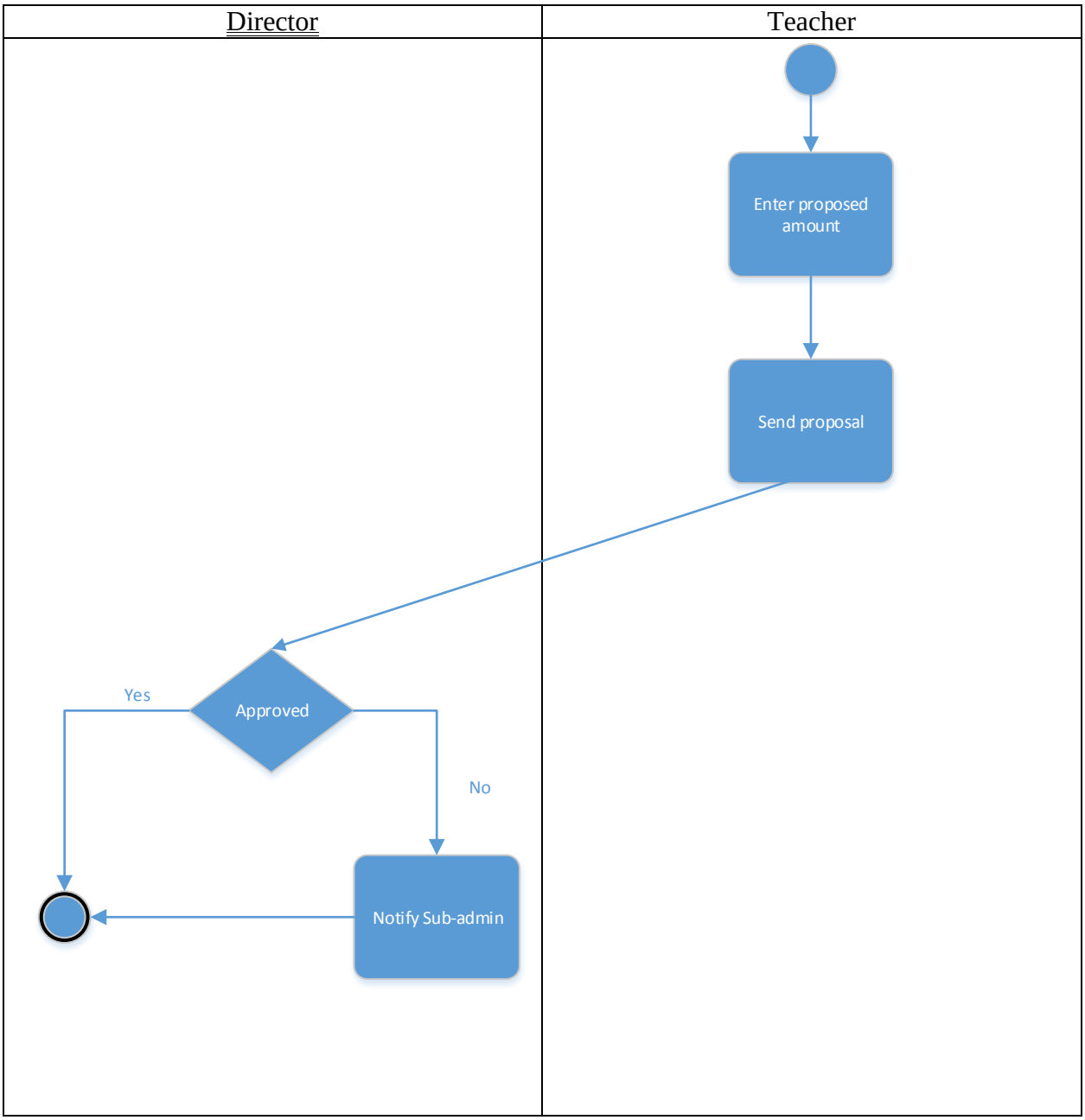


Figure 4.3.14: Swim Lane Diagram for “Expenditure management” use case

Use Case: Report

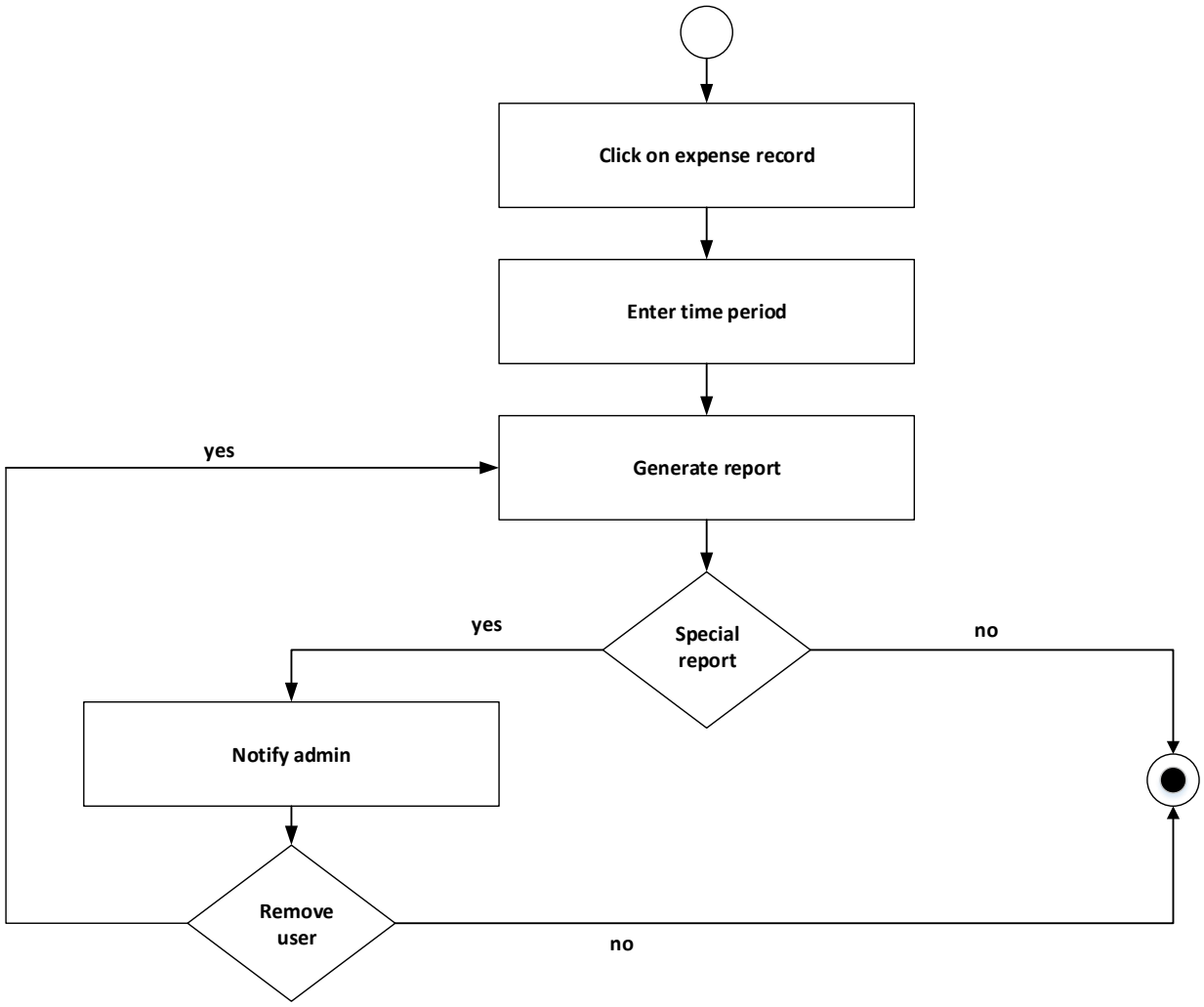


Figure 4.3.15: Activity Diagram for “Report” use case

Chapter 5

Data Model

In this chapter, we discuss the data model of our proposed system.

5.1 Data Modeling Concept:

A data model visually represents the nature of data, business rules governing the data, and how it will be organized in the database. A data model is comprised of two parts- logical design and physical design. Data Models are created in either Top-Down Approach or Bottom-Up Approach. In Top-Down Approach, data models are created by understanding and analyzing the business requirements. In Bottom-Up Approach, data models are created from existing databases. For our proposed system, we use Top-Down Approach.

5.2 Data Objects:

A database object in a relational database is a data structure used to either store or reference data. The most common object that people interact with is the table. Other objects are indexes, stored procedures, sequences, views and many more.

5.2.1 Grammatical Parsing (Noun Identify)

We have identified all nouns by parsing from our proposed user scenario and each noun may be in either problem space or solution space. A noun which is in solution space can be attribute or data object.

5.2.1.1 Table: Noun identification

NID	Noun	Problem/Solution Space	Attributes
1.	Expenditure Management System	P	-
2.	IIT	P	-
3.	University of Dhaka	P	-
4.	Transaction	P	-
5.	Sectors	S	8, 9, 10, 15, 19
6.	Director of IIT (Super Admin)	S	Director id, Username, Password
7.	Teacher (Sub Admin)	S	Teacher id, Username, Password
8.	Field name	S	-
9.	Field access key	S	-
10.	Estimated amount	S	-

11.	Annual budget report	P	-
12.	Amount of expense	S	-
13.	Approval	S	-
14.	Proposal	P	-
15.	Date of approval	S	-
16.	Proposed Sector	S	Proposed sector id, 12, 13, 17, 18
17.	Assigned field name	S	-
18.	Assigned sector name	S	-
19.	Spent amount	S	-
20.	Special report	P	-
21.	Spent fields	P	-

5.2.2 Identify Data Object:

Nouns having attributes are selected as data object. So, the data objects are:

1. ***Sector:**
   i. field_name
   ii. field_access_key
   iii. estimated_amount
   iv. date_of_approval
   v. spent_amount

* Format of creating dynamic data object

2. **Director:**
   i. director_id
   ii. user_name
   iii. password

3. **Teacher:**
   i. teacher_id
   ii. user_name
   iii. password

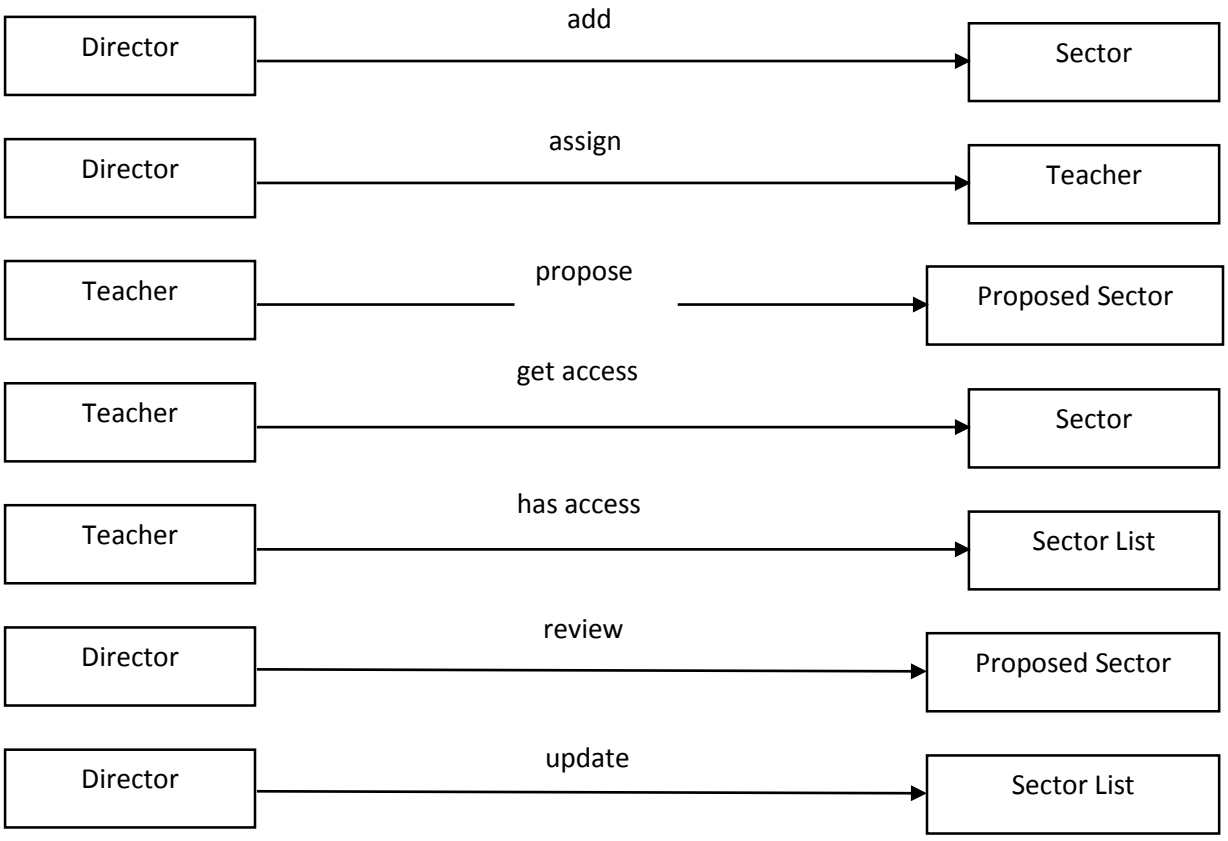
4. **Proposed Sector:**
   i. proposed_sector_id
   ii. amount_of_expense
   iii. approval

iv. assigned_field_name
v. assigned_sector_name

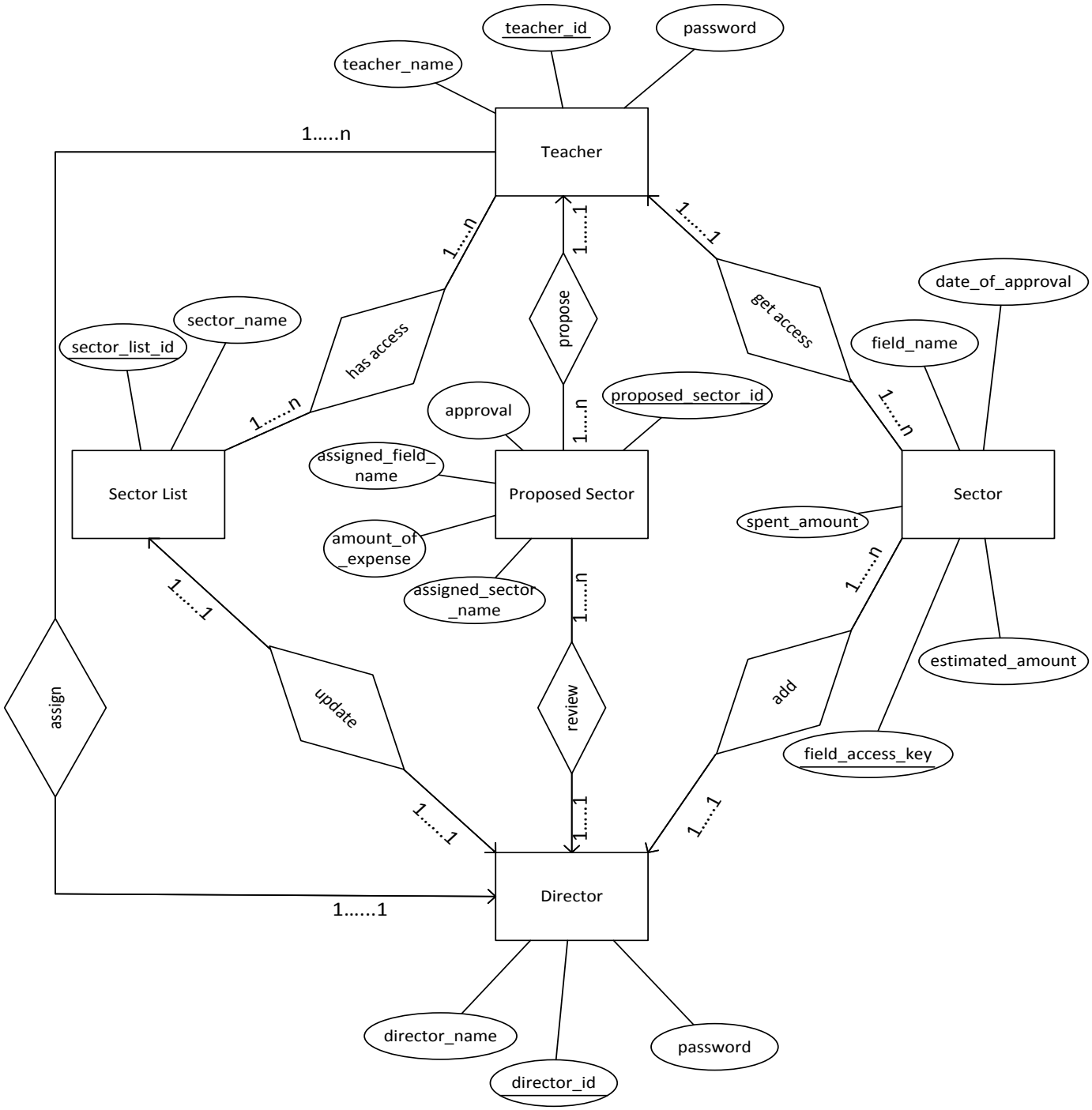
5. ***Sector List:**
   i. sector_list_id
   ii. sector_name

*Infrastructural Data Object

5.2.3 Pair Relations:



5.2.4 ER-Diagram:



5.2.1.1 Figure: ER Diagram

5.2.5 Database Table:

Director	
Attribute	Type (Size)
director_id	Integer (4)
user_name	Varchar2 (20)
password	Varchar2 (20)

Teacher	
Attribute	Type (Size)
teacher_id	Integer (4)
user_name	Varchar2 (20)
password	Varchar2 (20)
director_id	Integer (4)

Sector List	
Attribute	Type (Size)
sector_list_id	Integer (4)
sector_name	Varchar2 (20)
director_id	Integer (4)

Sector	
Attribute	Type (Size)
field_access_key	Integer (4)
field_name	Varchar2 (20)
estimated_amount	Integer (10)
spent_amount	Integer (10)
date_of_approval	Date
teacher_id	Integer (4)
director_id	Integer (4)

Proposed Sector	
Attribute	Type (Size)
proposed_sector_id	Integer (4)
assigned_field_name	Varchar2 (20)
assigned_sector_name	Varchar2 (20)

amount_of_expense	Integer (10)
approval	Integer(1)
teacher_id	Integer (4)
director_id	Integer (4)

Teacher-Sector List	
Attribute	Type (Size)
teacher_id	Integer (4)
sector_list_id	Integer (4)

Chapter 6

Class Based Model

In this chapter, we describe class based model of our proposed project "Expenditure Management System".

6.1 Class Based Modeling Concept:

Class-based modeling represents the objects that the system will manipulate, the operations that will applied to the objects, relationships between the objects and the collaborations that occur between the classes that are defined.

6.2 General Classification:

6.2.1 Table: Properties of general classification

Property Name	ID (G.C)
External entity	1
Things	2
Occurrences	3
Roles	4
Organizational units	5
Places	6
Structures	7

6.3 Selection Criteria:

6.3.1 Table: Properties of selection criteria

Property Name	ID (S.C)
Retained information	1
Needed service	2
Multiple attributes	3
Common attributes	4
Common operations	5
Essential requirements	6

6.4 Class Identification:

NID	Noun (Potential class)	Problem/solution space	General Classification	Selection Criteria
1.	Expenditure Management System	P	-	-
2.	IIT	P	-	-
3.	University of Dhaka	P	-	-
4.	Transaction	P	-	-
5.	*Sector	S	2, 3, 7	1, 3, 4, 6
6.	*Director of IIT (Super Admin)	S	4, 5	1, 2, 3, 4, 5, 6
7.	*Teacher (Sub Admin)	S	4, 5	1, 2, 3, 4, 5, 6
8.	Field name	S	-	-
9.	Field access key	S	-	-
10.	Estimated amount	S	-	-
11.	Annual budget report	P	-	-
12.	Amount of expense	S	-	-
13.	Approval	S	-	-
14.	Proposal	P	-	-
15.	Date of approval	S	-	-
16.	*Proposed Sector	S	2, 3, 7	1, 3, 4, 6
17.	Assigned field name	S	-	-
18.	Assigned sector name	S	-	-
19.	Spent amount	S	-	-
20.	Special report	P	-	-
21.	Spent fields	P	-	-

Star (*) sign indicates that there are selected for class. So, the classes are:

1. Director
2. Teacher
3. Sector
4. Proposed Sector
5. *Sector List
6. *Database

*Infrastructure classes

6.5 Subject, Verb, Object/ Predicate Identification:

Subject	Verb	Object/Predicate
Director	add	sectors of expense
Director	add	field names, field access key and estimated amount
Director	assign	teachers with username, password, sector names, field access keys.
Teacher	propose	amount of expense corresponding to the assigned fields of the assigned sectors
Teacher	send	proposal to director
Director	approve/reject	proposal
Director	send	notification
Teacher	view	notification
Director	view	annual report/special report
Director	remove	assigned teachers

6.6 Class Card:

1. Director	
Attributes	Methods
1. directorId 2. userName 3. password	1. authenticate() 2. register() 3. addSector() 4. viewSectors() 5. addField() 6. viewFields() 7. assign() 8. viewAssignees() 9. viewProposal() 10. review() 11. send() 12. remove() 13. viewReport()
Responsibilities	Collaborator Classes
1. Authentication as super admin 2. Registration as super admin 3. Addition of sector 4. Addition of field of expense in a sector 5. Viewing existing sectors 6. Viewing existing fields of a sector	1. Database 2. Database 3. Sector List, Database 4. Sector, Database 5. Sector List, Database 6. Sector, Database

7. Assignment of teachers in specific fields of a specific sector 8. Viewing the assignee list 9. Viewing existing proposals 10. Reviewing a proposal 11. Sending notification 12. Removing assignee from the assignee list 13. Viewing generated reports	7. Teacher, Sector, Database 8. Teacher, Database 9. Proposed Sector, Database 10. Proposed Sector, Database 11. Proposed Sector, Database 12. Teacher, Database 13. Database

2. Teacher	
Attributes	Methods
1. teacherId 2. userName 3. password 4. directorId 5. accessable<sectorId, listOfFields>	1. authenticate() 2. register() 3. propose() 4. send() 5. view()
Responsibilities	Collaborator Classes
1. Authentication as super admin 2. Registration as super admin 3. Proposing a new amount of expense 4. Sending proposal to the director 5. Viewing notification or feedback	1. Database 2. Database 3. Proposed Sector, Database 4. Proposed Sector, Director, Database 5. Proposed Sector, Database

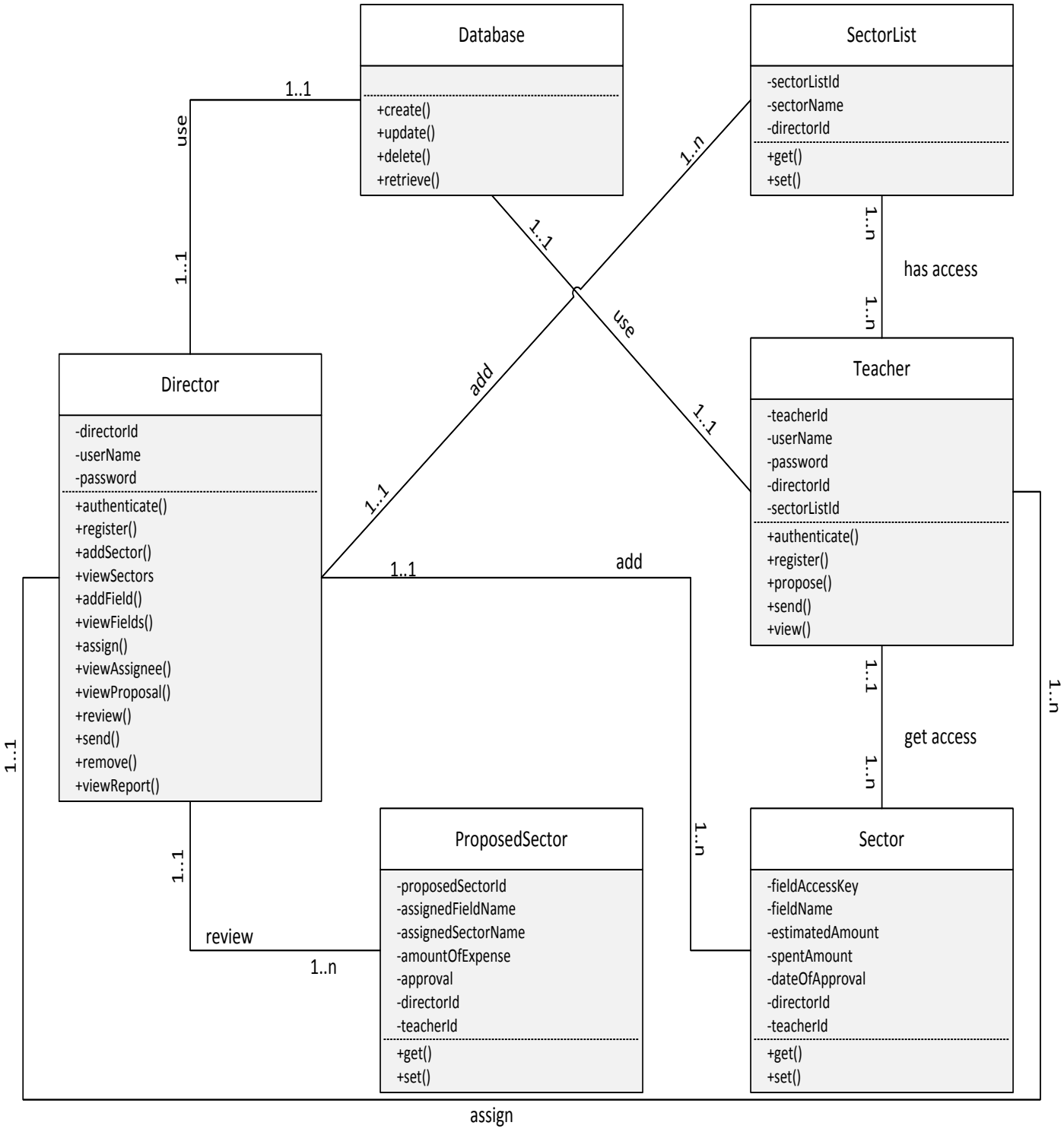
3. Sector List	
Attributes	Methods
1. sectorListId 2. sectorName 3. directorId	1. get() 2. set()
Responsibilities	Collaborator Classes
1. Getting attributes from database 2. Setting attributes to database	1. Database 2. Database

4. Sector	
Attributes	Methods
1. fieldAccesKey 2. fieldName 3. estimatedAmount 4. spentAmount 5. dateOfApproval 6. teacherId 7. directorId	1. get() 2. set()
Responsibilities	Collaborator Classes
1. Getting attributes from database 2. Setting attributes to database	1. Database 2. Database

5. Proposed Sector	
Attributes	Methods
1. proposedSectorId 2. assignedFieldName 3. assignedSectorName 4. amountOfExpense 5. approval 6. teacherId 7. directorId	1. get() 2. set()
Responsibilities	Collaborator Classes
1. Getting attributes from database 2. Setting attributes to database	1. Database 2. Database

6. Database	
Attributes	Methods
----	1. create() 2. update() 3. delete() 4. retrieve()
Responsibilities	Collaborator Classes
1. Creating an element 2. Updating the element 3. Deleting an element 4. Retrieving element(s)	1. --- 2. --- 3. --- 4. ---

6.7 CRC Diagram:



6.7.1 Figure CRC Diagram

Chapter 7

Flow Oriented Model

This chapter is intended to describe “Flow Oriented Model” of our proposed system.

7.1 Introduction

Low models focus on the flow of data objects as they are transformed by processing functions. Derived from structured analysis, flow models use the data flow diagram, a modeling notation that depicts how input is transformed into output as data objects move through the system. Each software function that transforms data is described by a process specification or narrative. In addition to data flow, this modeling element also depicts control flow. Data flow oriented modeling is the most widely used analysis notation. Flow oriented modeling focuses on structured analysis and design, follows a top to down methodology and uses a graphical technique depicting information flows and the transformations that are applied as data moves from input to output.

7.2 Data Flow Diagram (DFD)

A data flow diagram (DFD) is a significant modeling technique for analyzing and constructing information processes. DFD literally means an illustration that explains the course or movement of information in a process. DFD illustrates this flow of information in a process based on the inputs and outputs. A DFD can be referred to as a Process Model.

7.2.1 Level – 0 of DFD

By analyzing the requirements and user scenario, we have found two inputs and one output. Theses inputs are –

I. Admin information
II. Annual budget report

And output is –

I. Report

Level - 0 of the DFD diagram indicates these inputs and the output interacted with the system.

Level – 0 of the DFD of our proposed system is given below -

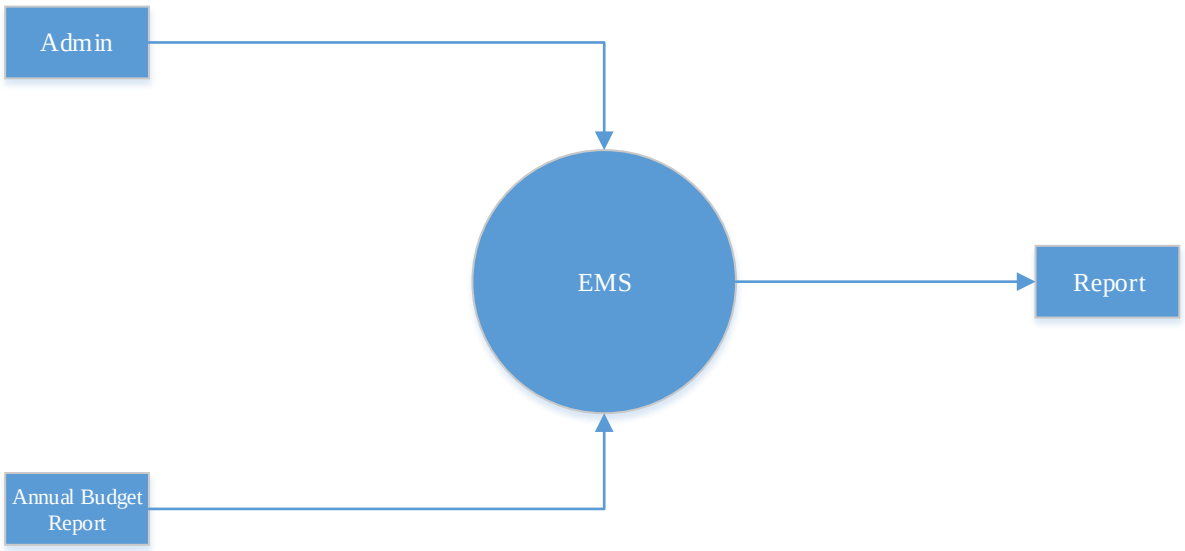


Figure 7.2.1: Level – 0 of the DFD

7.2.2 Level – 1 of DFD

Here, level-1 of the DFD shows how these inputs flow and store in database or produce output of the system.

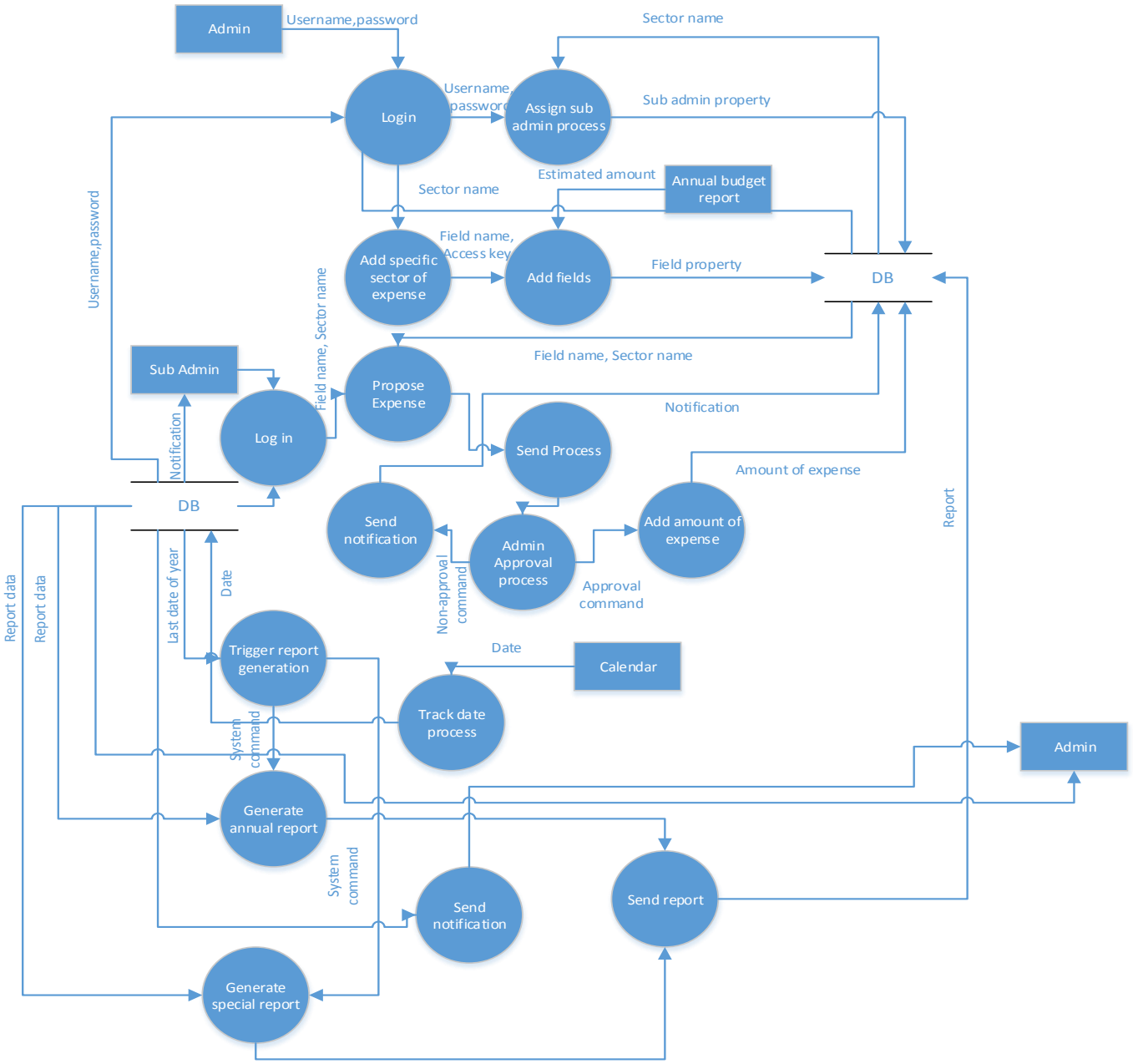


Figure 7.2.2: Level – 1 of DFD

Chapter 8

Behavioral Model

The behavioral model indicates how software will respond to external events.

8.1 State Transition Diagram:

State diagram represents active states for each class the events (triggers). For performing this task, we have identified all the events, their initiators and collaborators.

Event List:

1. add specific sectors of expense
2. add various field names of expense under each sector with a unique field access key and estimated amount
3. assigns each teacher with a unique username, password, one or many sector names includes one or many field access keys
4. propose an amount of expense corresponding to the assigned fields of the assigned sectors
5. send proposal to the Director for approval.
6. review (accept/reject) the proposal
7. send notification to teacher
8. view annual/special report

Identifying the initiator and collaborator:

Event No.	Initiator	Collaborators
1.	Director	Sector List
2.	Director	Sector
3.	Director	Teacher, Sector List, Sector
4.	Teacher	Proposed Sector
5.	Teacher	Proposed Sector, Director
6.	Director	Proposed Sector
7.	Director	Teacher
8.	Director	-

State Transition Diagrams (STD):

1.Director:

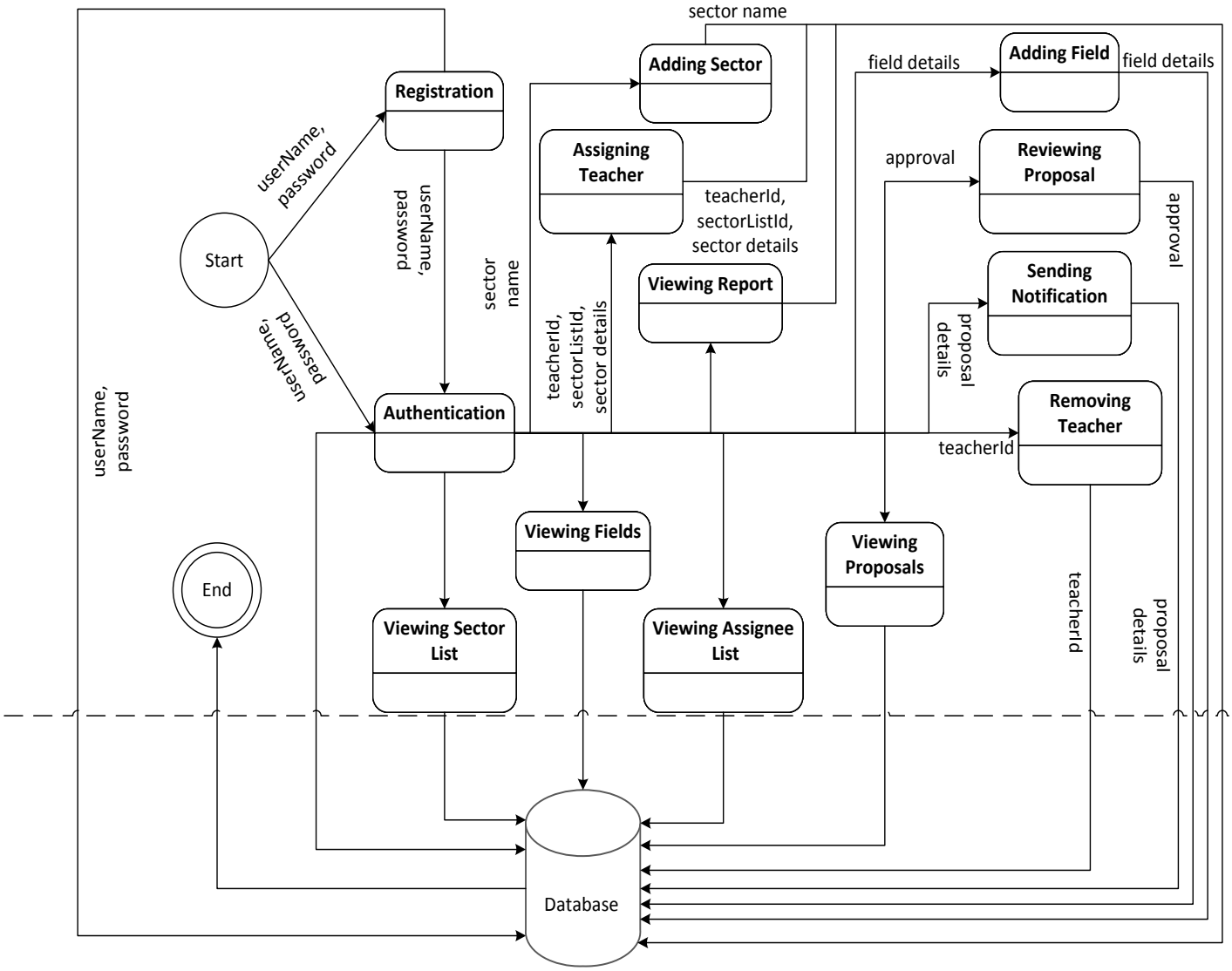


Figure: STD for “Director” class

2.Teacher:

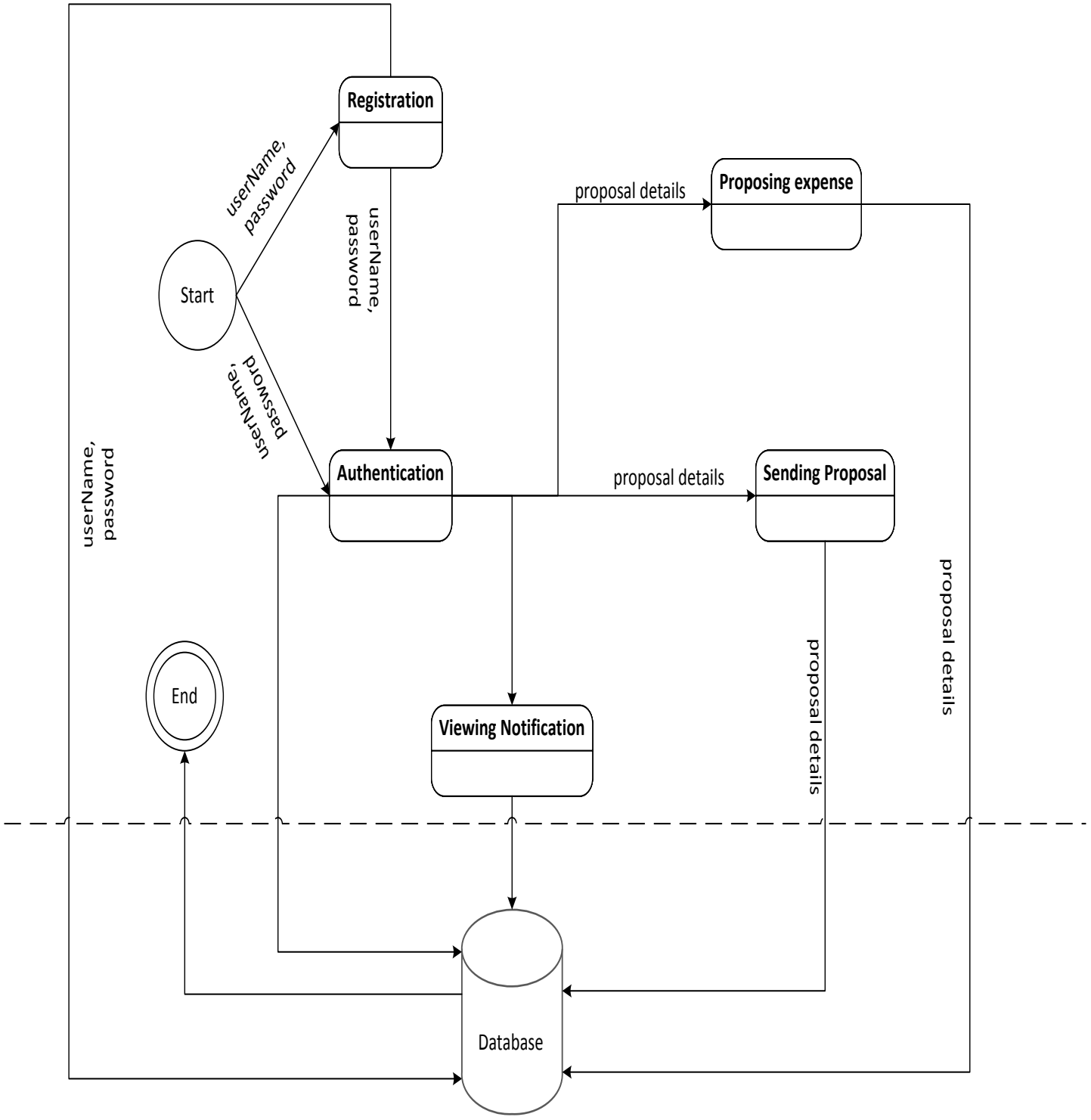


Figure: STD for "Teacher" class

8.2 Sequence Diagram:

Sequence diagram indicates how events cause transitions from object to object.

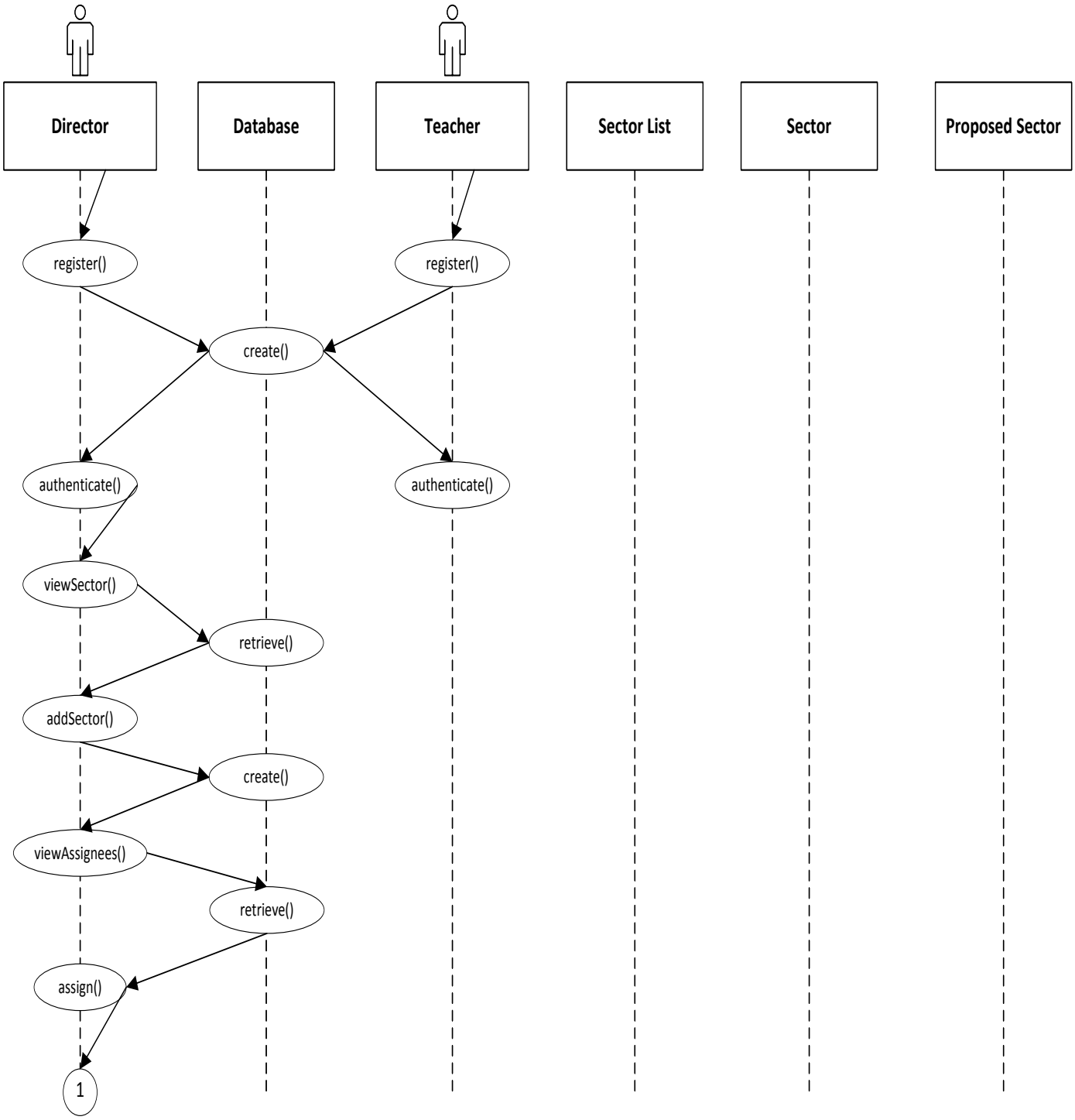
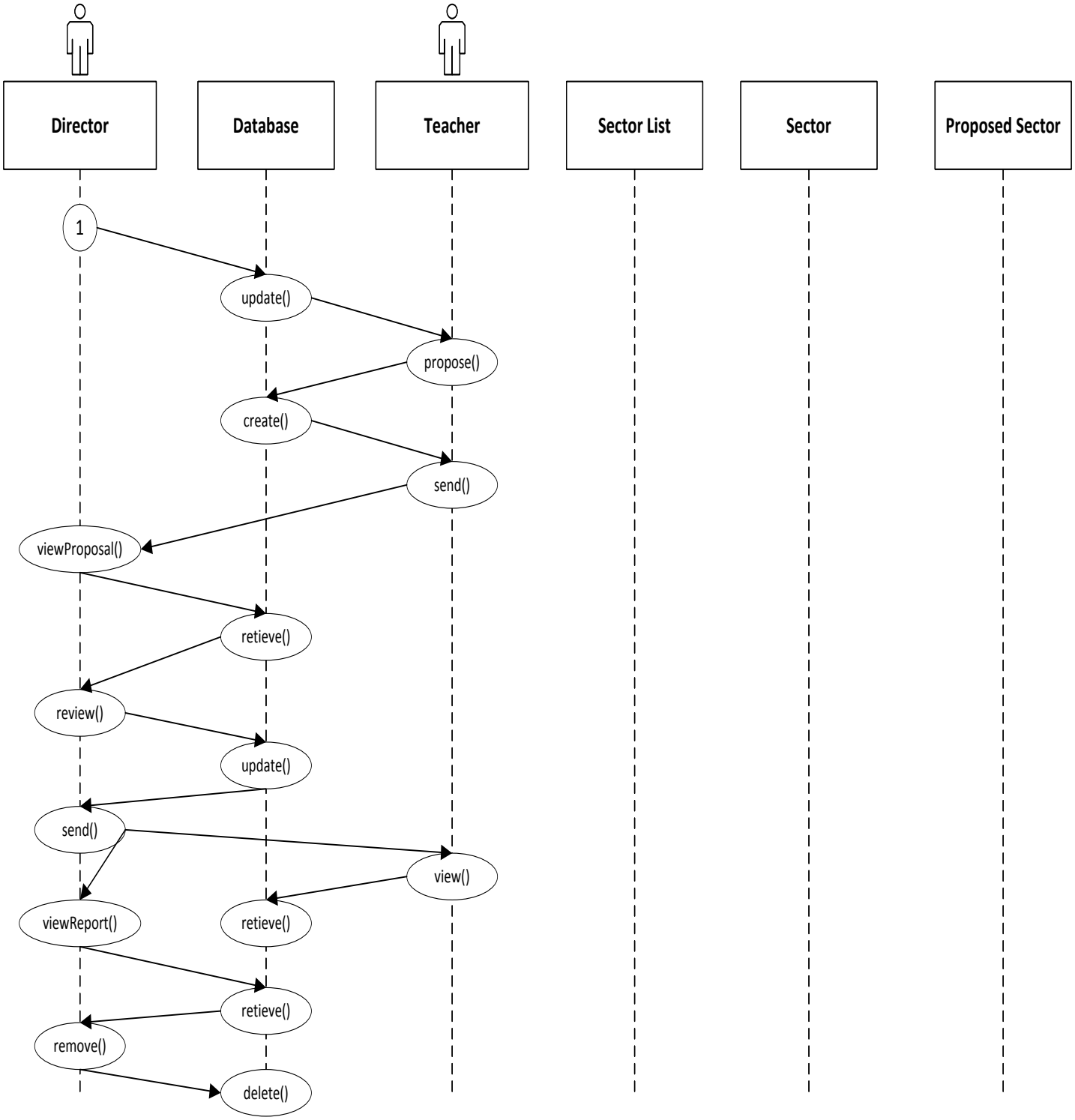


Figure: Sequence Diagram (Part-1)



8.2.2 Figure: Sequence Diagram (Part-2)