Abstract

This document contains the requirements, analysis and design artifacts for the Employee Scheduling System (ESS) software system. ESS is a personnel scheduling system that facilitates the employee submission and subsequent supervisor approval or denial of time off requests.

The rest of this document is structured as follows: Chapter 1 contains the introduction. This chapter presents a brief description of the system. Chapter 2 outlines the functional requirements of the system. In addition, Chapter 2 contains use case diagrams and use case descriptions for all use cases involved in ESS. Lastly, the requirement analysis will be outlined in Chapter 2 on next revision of RAD document. Chapter 3 illustrates key GUI screen mockups for the Employee Scheduling System. Chapter 4

1 INTRODUCTION

1.1 SCOPE OF SYSTEM

The Employee Scheduling System (ESS) is a system used to provide simple and efficient means for an employee to request time off and for appointed supervisors to administrate, approve, or deny those requests. ESS has an internal database with authorized users and their password hashes. Employees can submit requests for time off, which are stored in the database. Supervisors are then able to see the contents of the time off requests, the employee that initiated it, and the reason for the request. Once the Supervisor responds to a request, it is removed from the Supervisor's queue and the database. The system includes secure login, logout functionality in addition to the primary scheduling applications.

1.2 OVERVIEW OF DOCUMENT

The rest of the document is structured as follows: Chapter 2 outlines the functional requirements of the system, then the use case diagram. Individual detailed use case descriptions are then listed. Chapter 3 depicts several individual user interface mockups. Chapter 4?

2.3 USE CASE DESCRIPTIONS

Use case name	Login
Participating actors	Initiated by Employee or Supervisor
Flow of events	1. Employee enters their user ID in User ID field and Password in Password field. 2. **ESS responds by authenticating the entered user ID and Password.**
Entry condition	
Exit condition	User ID and password are authenticated via SQL query.
Security requirements	The password must be hashed at all times. The dialogue boxes that handle username and password must be shielded against code execution and SQL injections. Password policy must be used to eliminate malicious input. Windows shortcut-key exploits must be disabled to avoid accessing a shell or forcing an exploit. Only <ENTER> will be recognized for acknowledgement of the message/dialogue box.

Figure 2.2: Login: valid login

Event #2 is now incomplete.
The correction from the previous
version was the numbering of the events
and not to remove the descriptors.

Use case name	Logout
Participating actors	Initiated by Supervisor
Flow of events	3. Supervisor presses the logout button on Request Response menu. 4. **System closes Response Request menu and returns user to the login screen.**
Entry condition	Supervisor is logged in to the ESS system.
Exit condition	Supervisor is logged out and returned to the login interface.
Security requirements	Resources allocated to the session must be terminated properly to ensure there are no bugs in the software.

Figure 2.4: Logout: RRMenu

Use case name	Logout
Participating actors	Initiated by Supervisor
Flow of events	5. Supervisor presses the logout button on Supervisor menu. 6. **System closes Response Request menu and returns user to the login menu.**
Entry condition	Supervisor is logged in to the ESS system.
Exit condition	Supervisor is logged out and returned to the login menu.
Security requirements	Resources allocated to the session must be terminated properly to ensure there are no bugs in the software.

Figure 2.4: Logout: SuperMenu

is not a use case

Use case name	SuperMenu
Participating actors	~~Initiated by~~ Supervisor
Flow of events	1. Supervisor selects Request Response button 2. ESS responds by opening Request Response Menu
Entry condition	
Exit condition	
Security requirements	

Figure 2.5: SuperMenu: RRMenu

Use case name	SuperMenu
Participating actors	Initiated by Supervisor
Flow of events	3. Supervisor selects Time Off Request button 4. ESS responds by opening Time Off Request Menu
Entry condition	
Exit condition	
Security requirements	

Figure 2.6: SuperMenu: TOR Menu

Use case name	TimeOffRequest
Participating actors	Initiated by Supervisor or Employee
Flow of events	1. The Employee or Supervisor select a date, time and a reason per request. 2. **2. ESS receives the form and pushes following fields to the database tables: Employee/Supervisor name, request date, request time, and request reason.**
Entry condition	The Supervisor selects Request Time Off button from Supervisor Menu Form.
Exit condition	The employee's time off request is reflected in the appropriate employee and supervisor queues.
Security requirements	

Figure 2.9: TimeOffRequest

Event #2 is incomplete.
Same issue as on page 7.

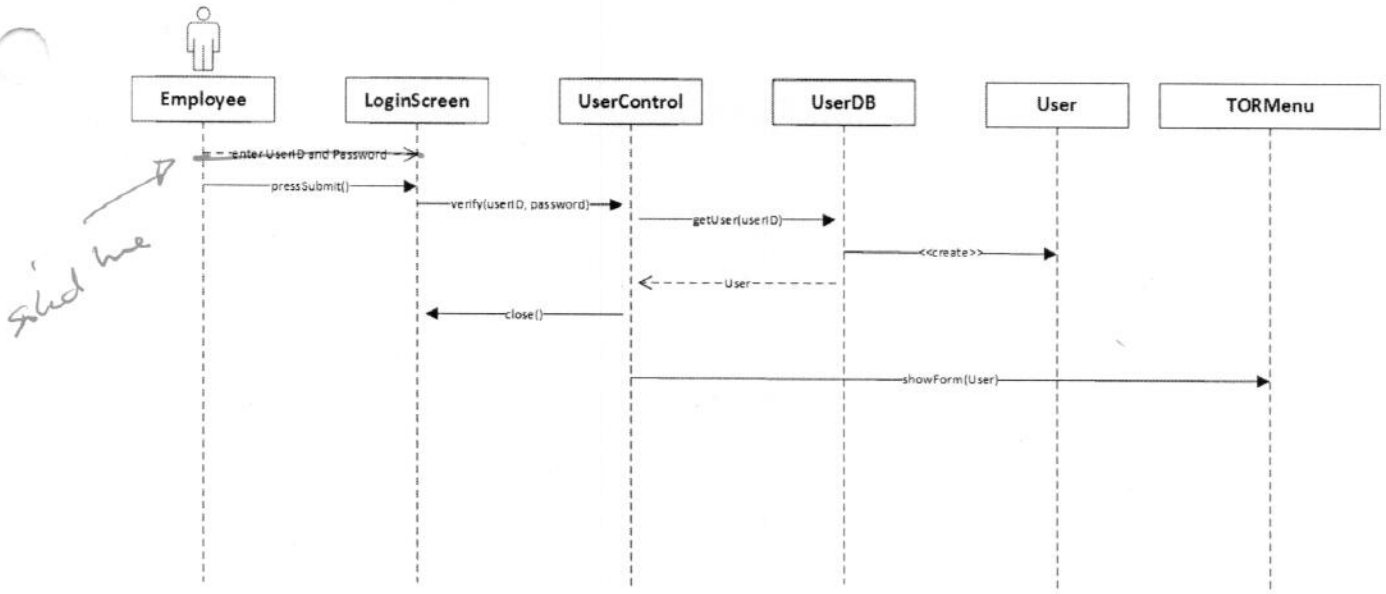

Figure 2.10: Login: valid login (non-supervisor) sequence

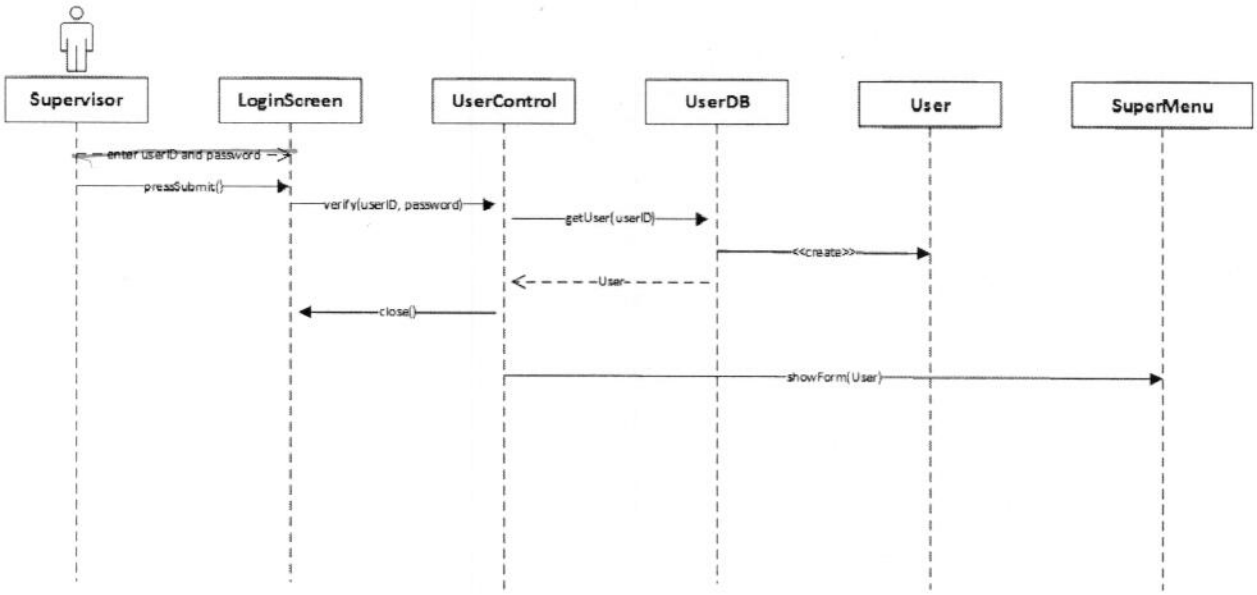

Figure 2.11: Login: valid login (Supervisor) sequence

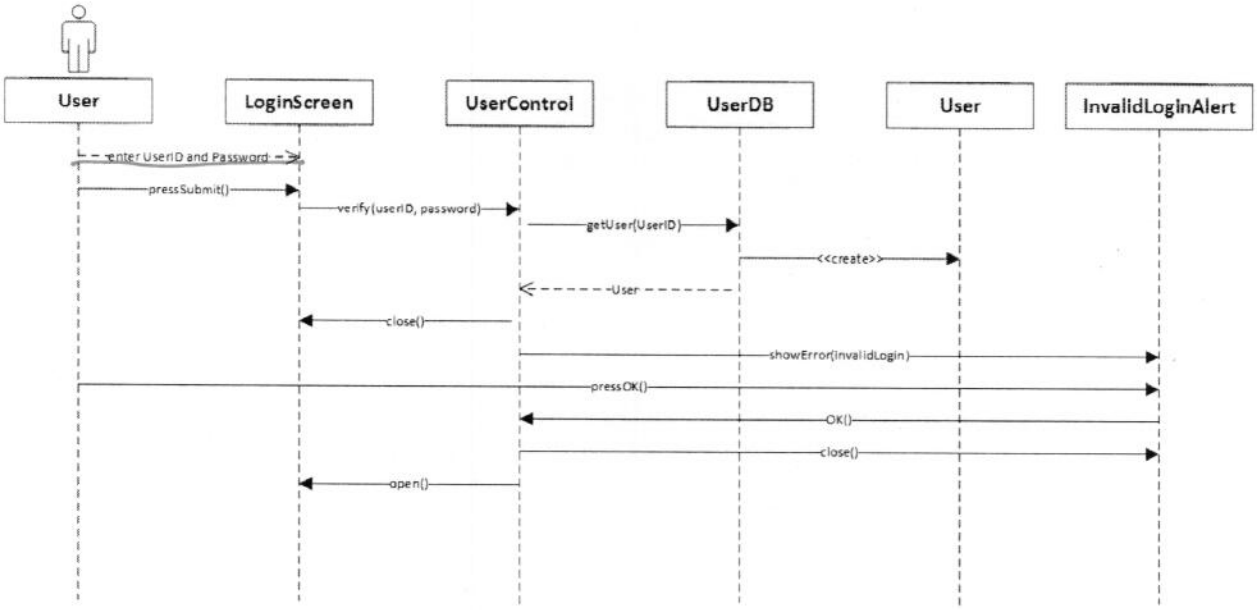

Figure 2.12: Login: invalid sequence

Use case name	Logout
Participating actors	Initiated by User
Flow of events	1. User presses the logout button on the Time Off Request menu. 2. **System closes Time Off Request menu and returns user to the login screen.**
Entry condition	User is logged in to the ESS system.
Exit condition	User is logged out and returned to the login interface.
Security requirements	Resources allocated to the session must be terminated properly to ensure there are no bugs in the software.

Figure 2.4: Logout: TORMenu

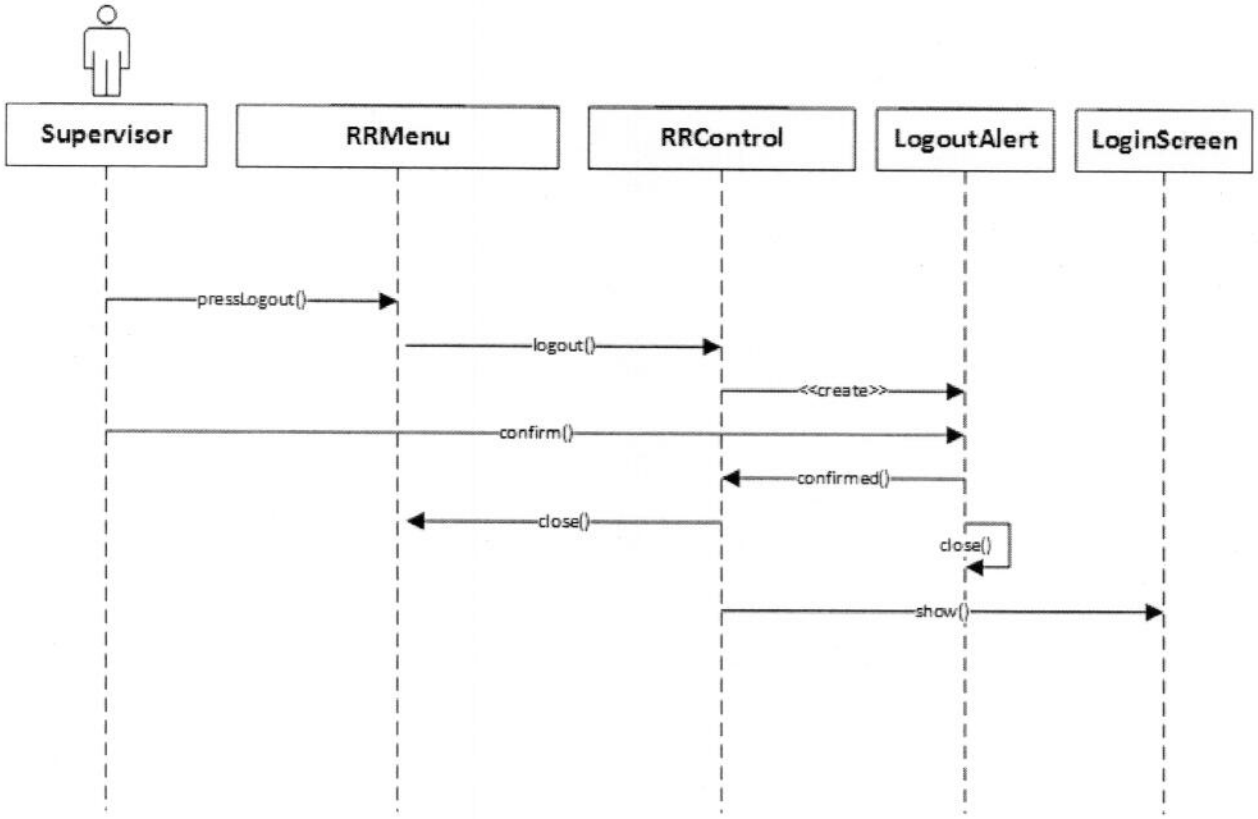

Figure 2.14: Logout: RRMenu sequence

There is no sequence diagram for fig 2.4

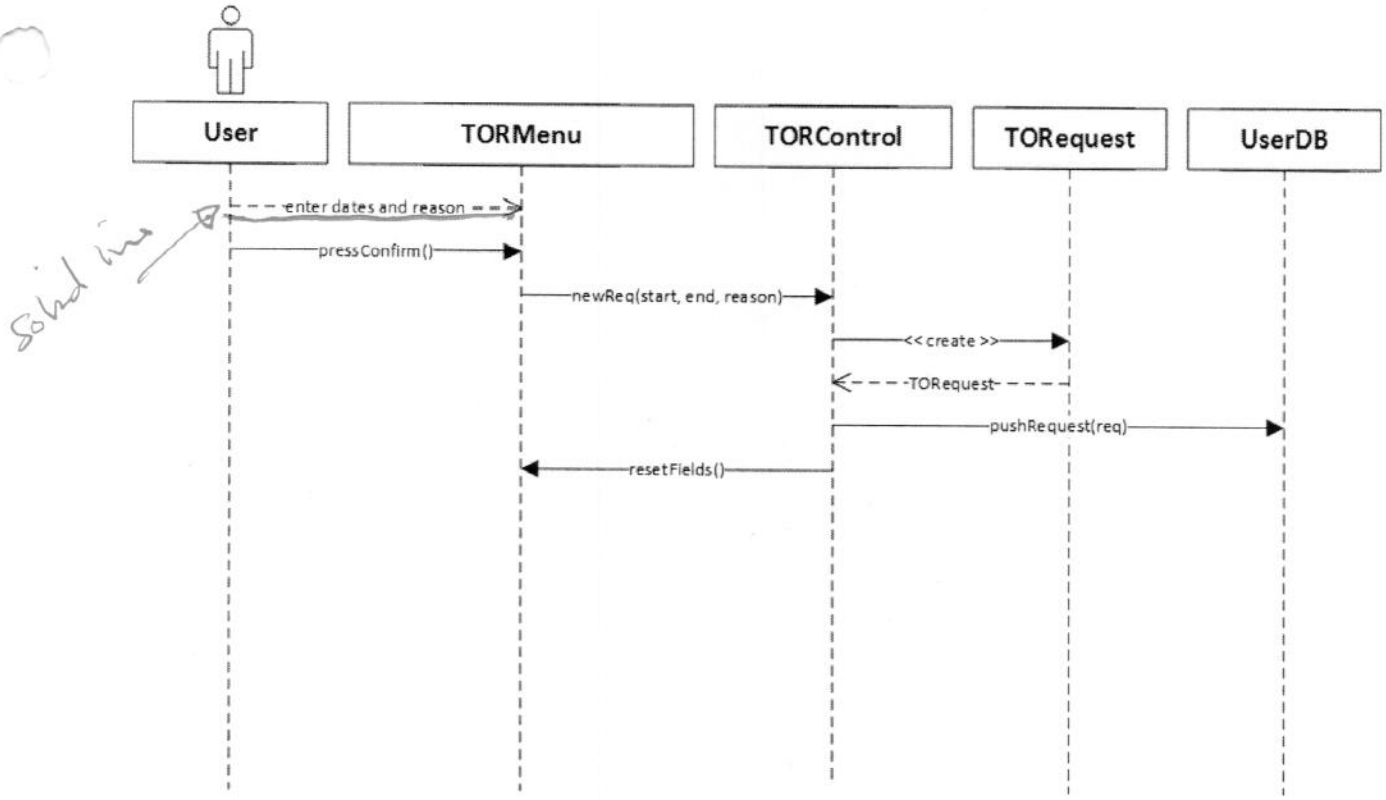

Figure 2.15: TOR(Time Off Request) sequence

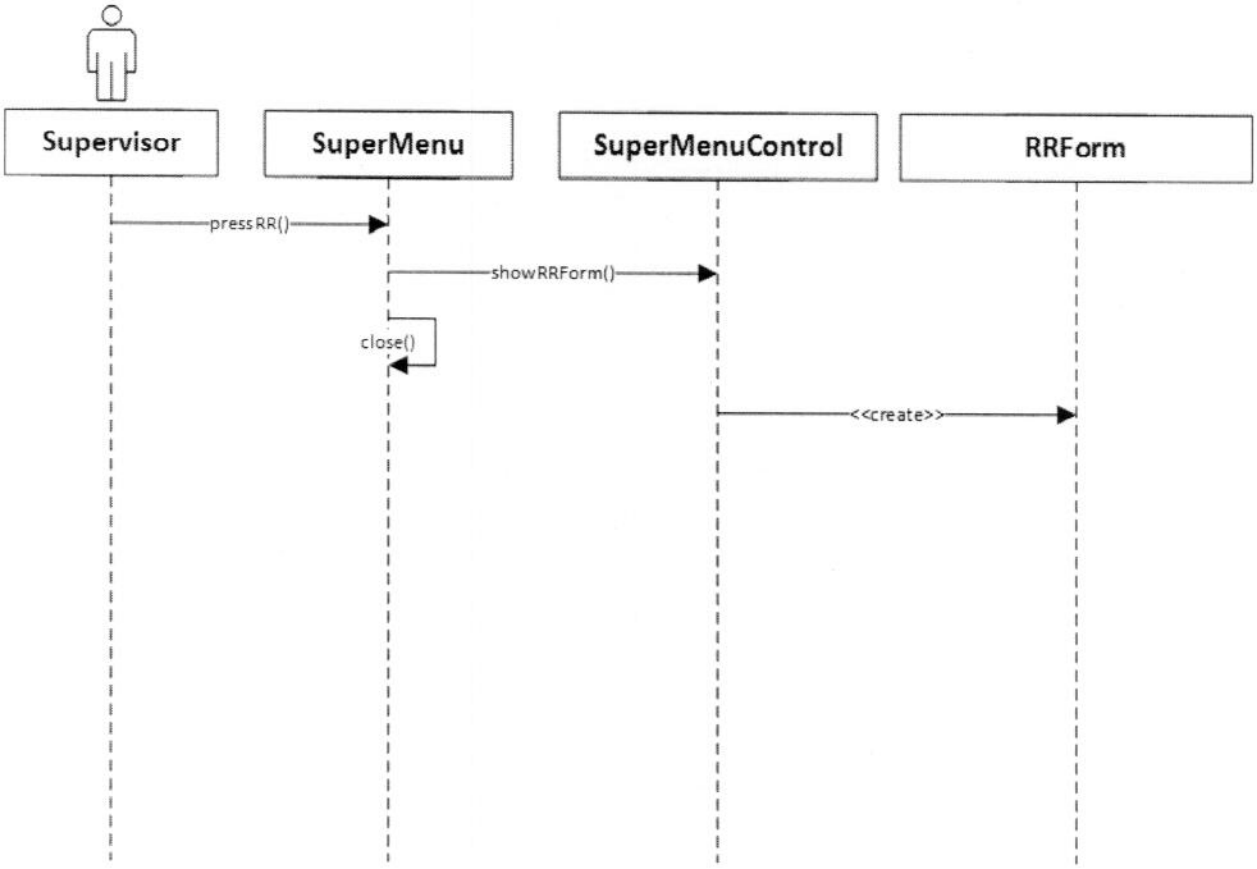

Figure 2.16: SuperMenu: RRMenu sequence

Use case name is not a use case (fig 2.5)

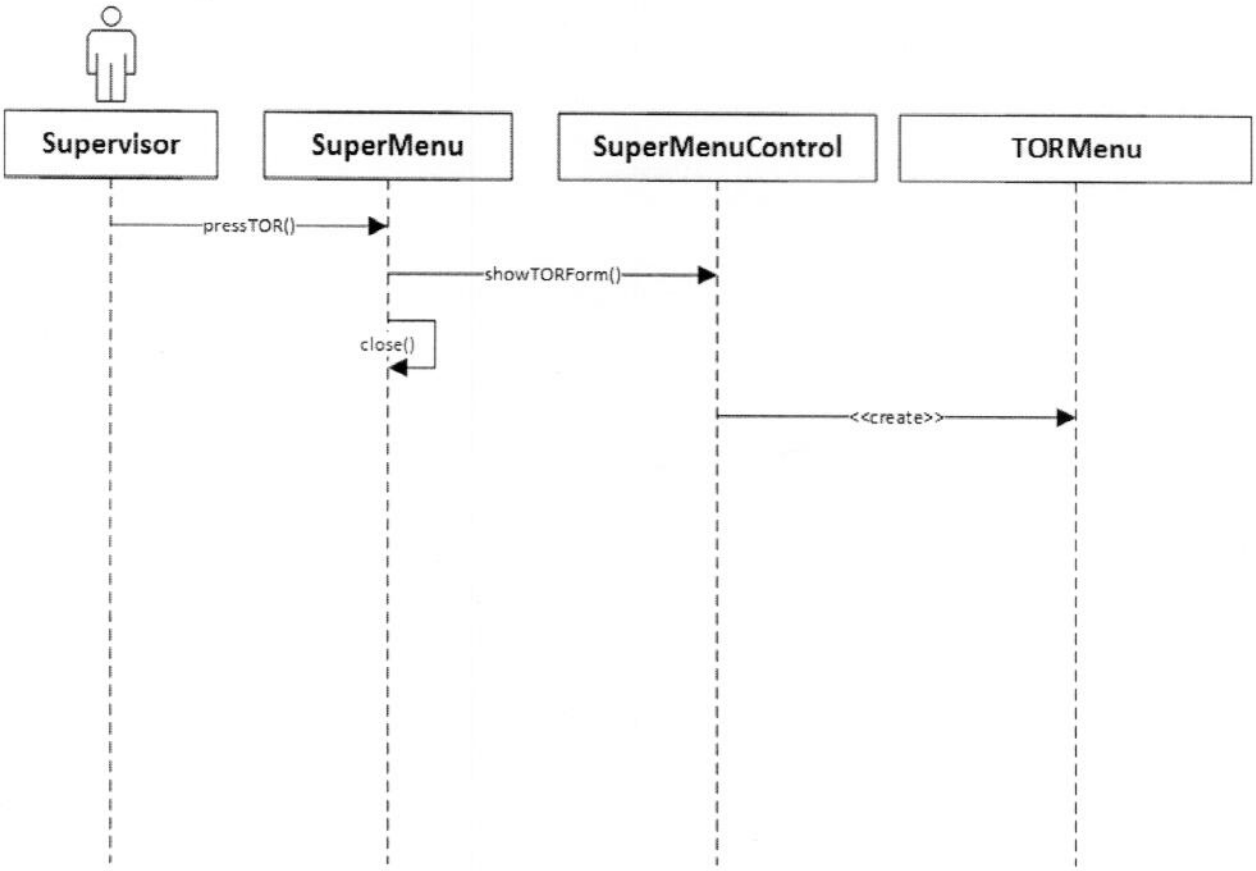

Figure 2.17: SuperMenu: TORMenu sequence

use case name
is not a use case (fig 2.5)

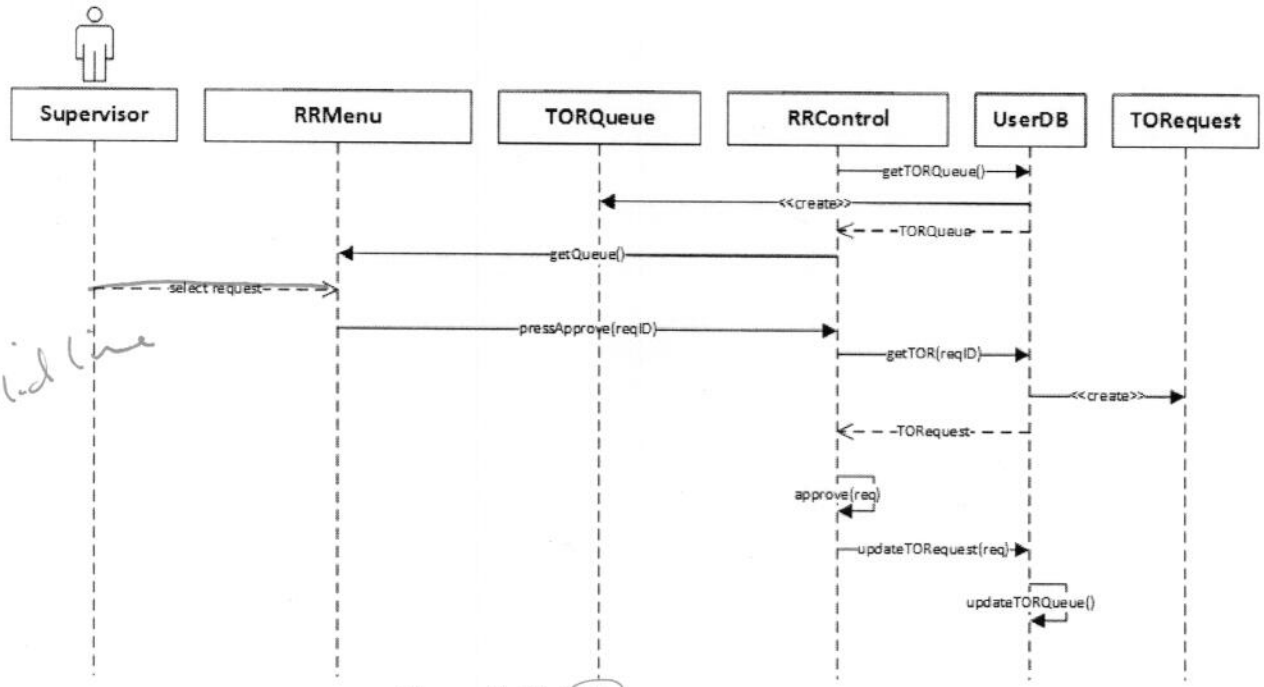

Figure 2.18: RR: approve sequence

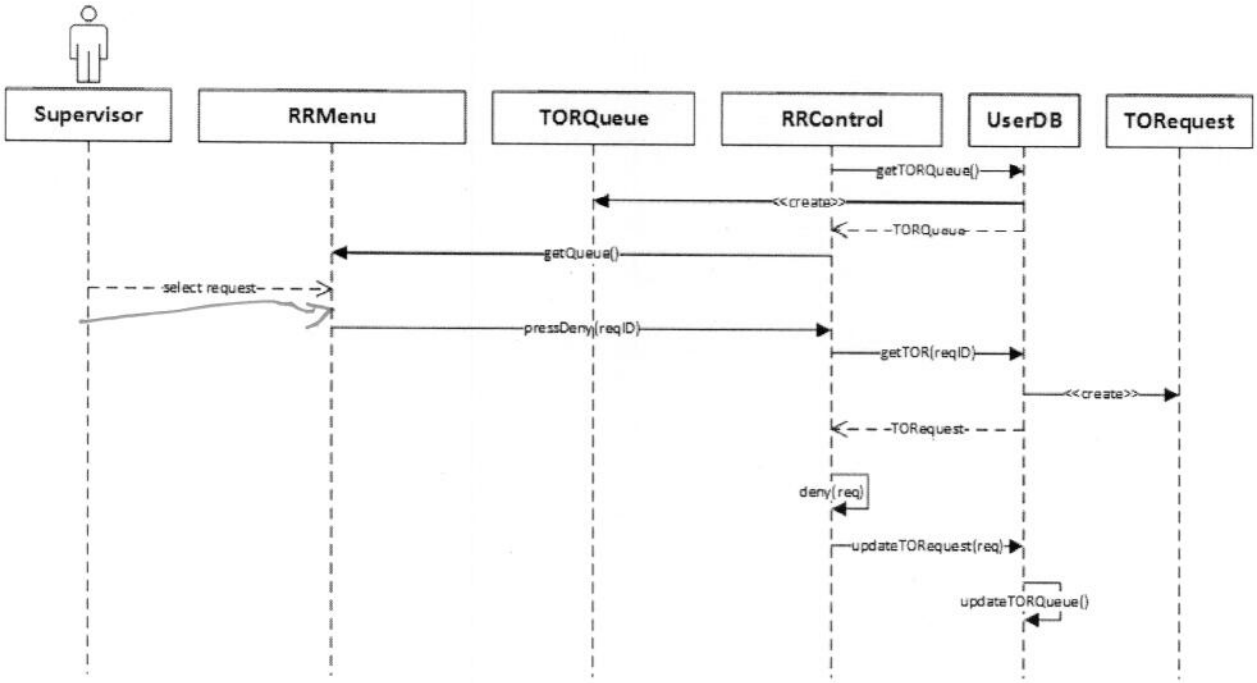

Figure 2.19: RR: deny sequence

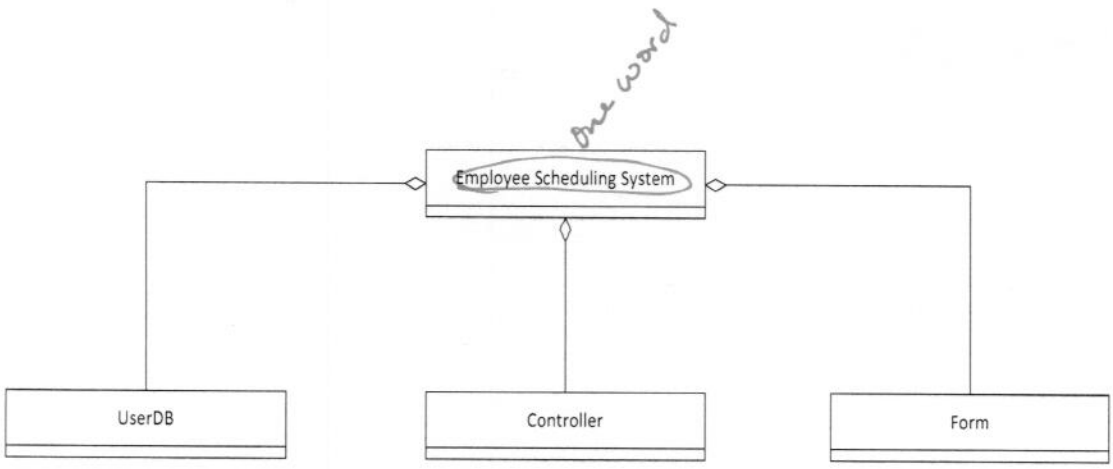

Figure 4.5: Class Diagram

4.2 DETAILED CLASS DESIGN

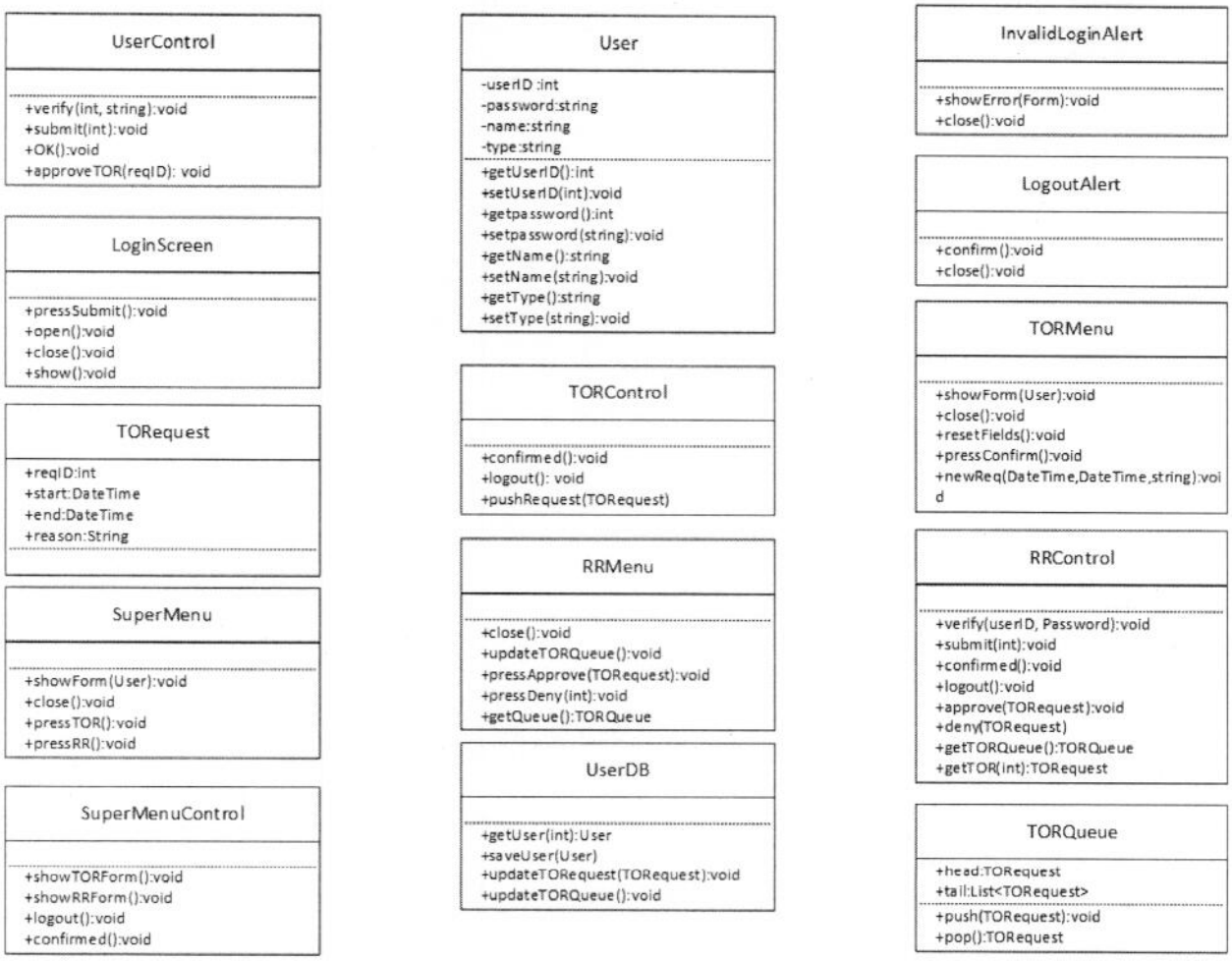

Figure 4.6: Class Diagram

It is not clear how you would capture the association in Fig. 4.3

Forms? Controller?