Software Design and Analysis

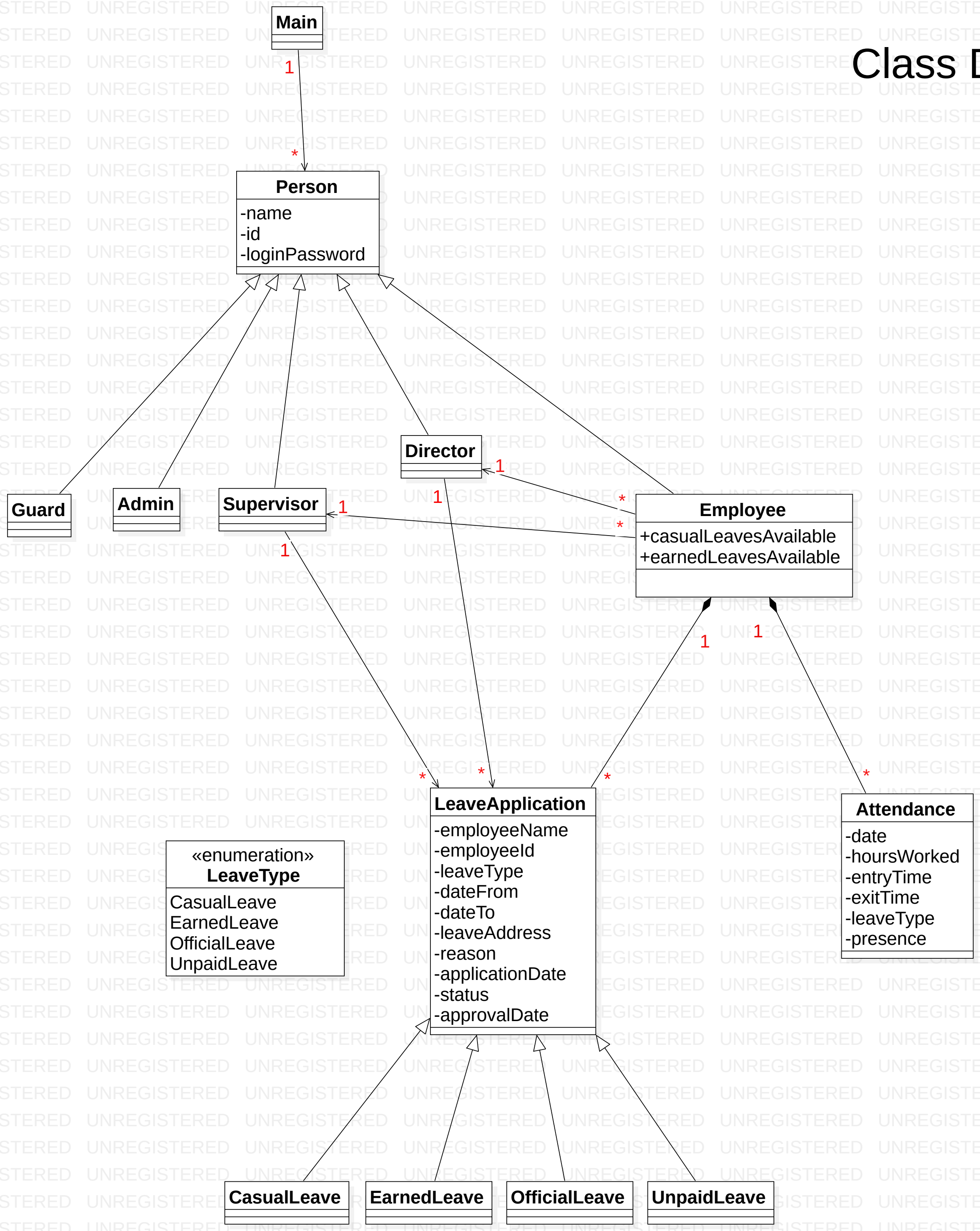
Assignment 3

Group Members:

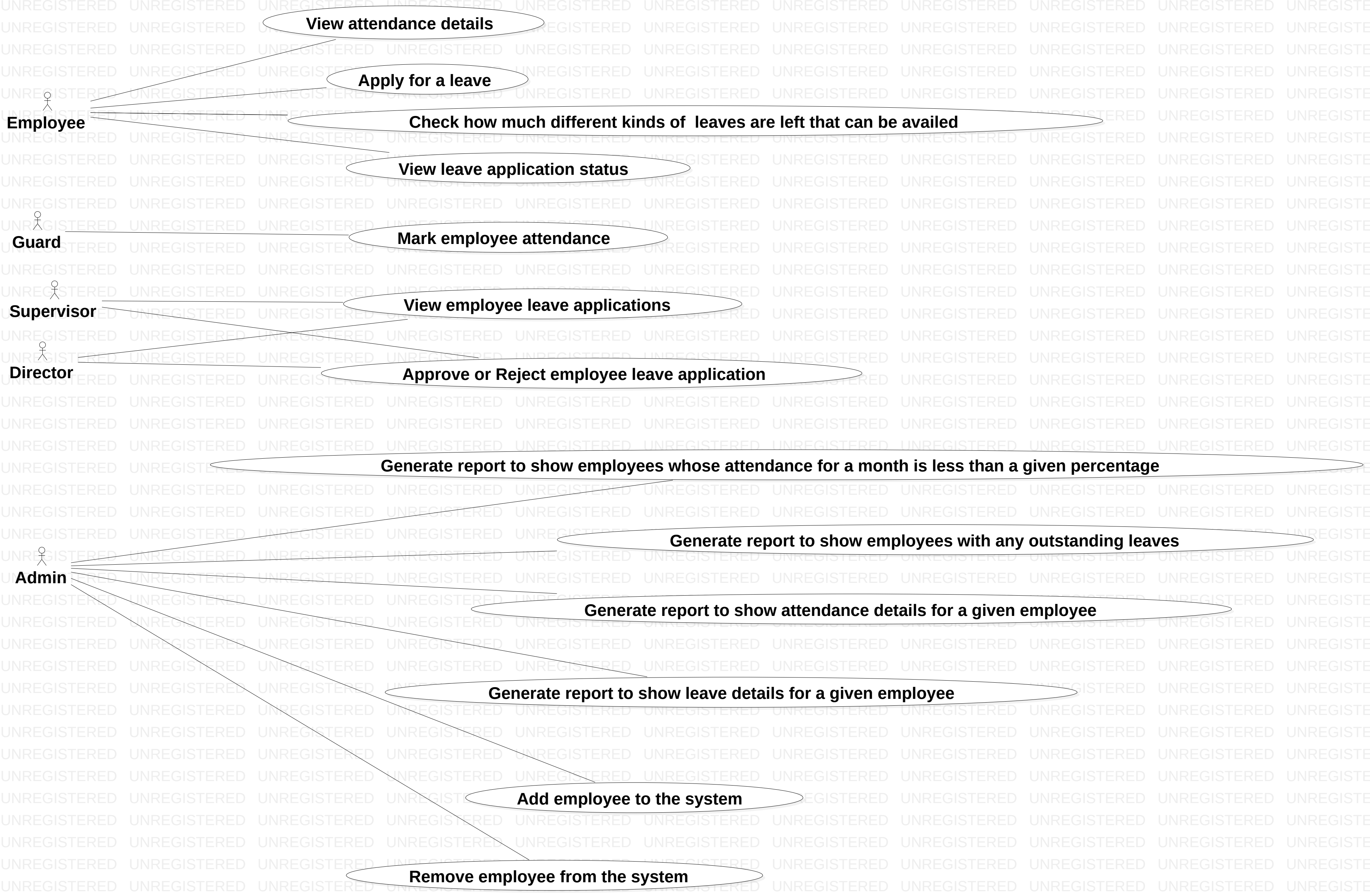
Faizan Shabir 22L-6552

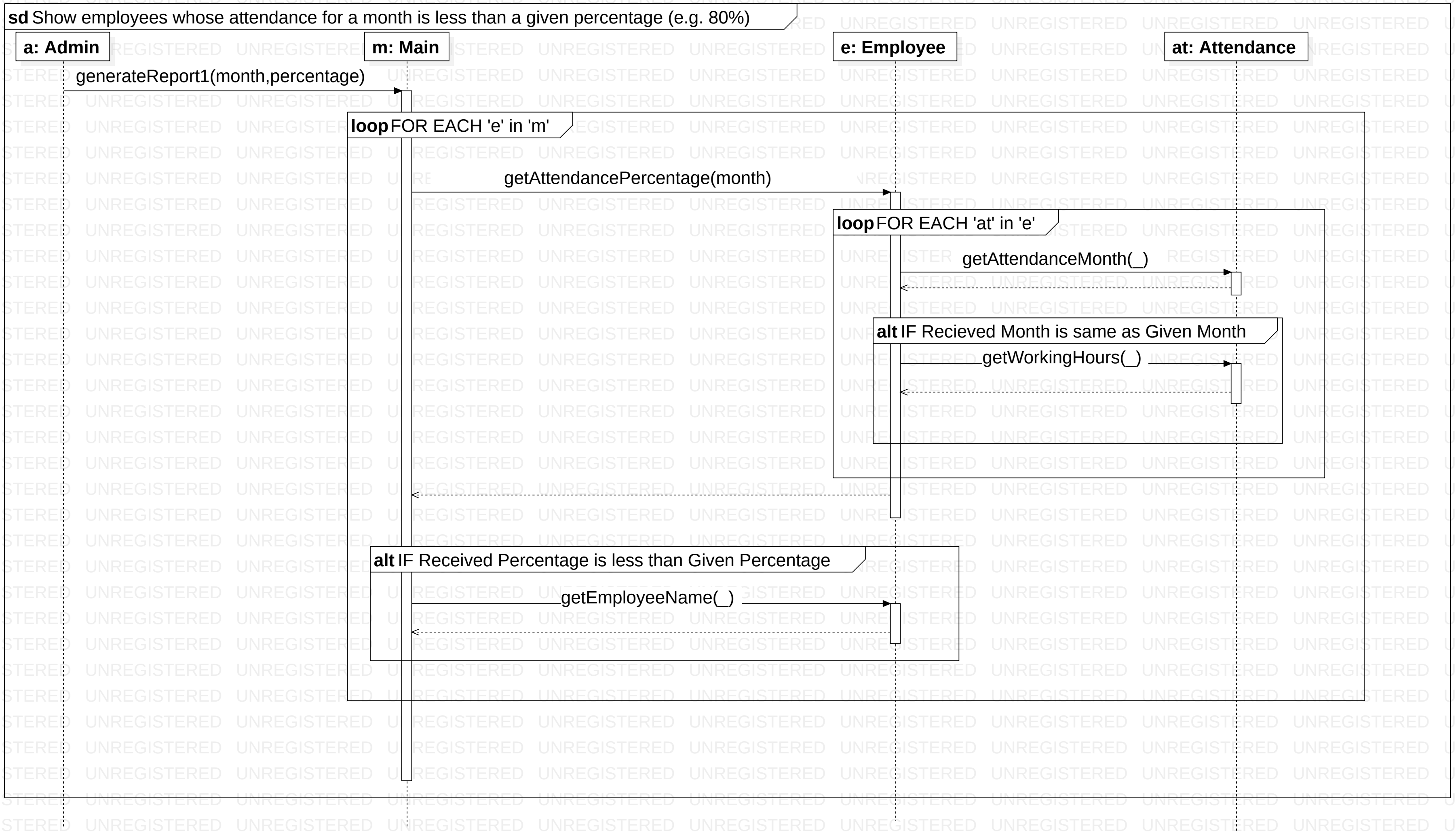
Ibaad Hussain 21L-1827

Class Diagram



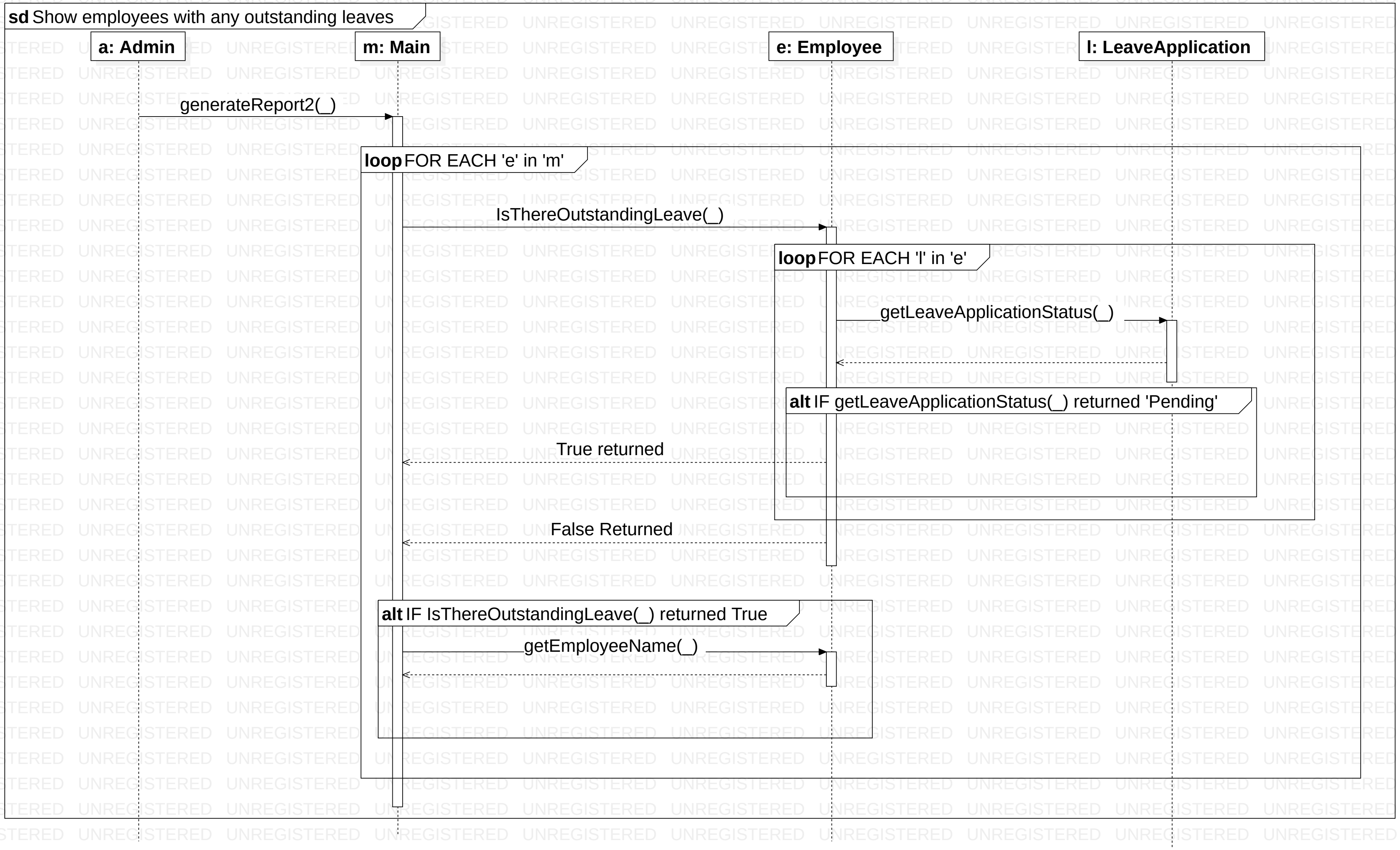
Use Case Diagram


Employee
View attendance details
Apply for a leave
Check how much different kinds of leaves are left that can be availed
View leave application status
Guard
Mark employee attendance
Supervisor
View employee leave applications
Director
Approve or Reject employee leave application
Generate report to show employees whose attendance for a month is less than a given percentage
Admin
Generate report to show employees with any outstanding leaves
Generate report to show attendance details for a given employee
Generate report to show leave details for a given employee
Add employee to the system
Remove employee from the system


sd Show employees whose attendance for a month is less than a given percentage (e.g. 80%)
a: Admin
m: Main
e: Employee
at: Attendance
generateReport1(month,percentage)
loop FOR EACH 'e' in 'm'
getAttendancePercentage(month)
loop FOR EACH 'at' in 'e'
getAttendanceMonth(_)
alt IF Recieved Month is same as Given Month
getWorkingHours(_)
alt IF Received Percentage is less than Given Percentage
getEmployeeName(_)

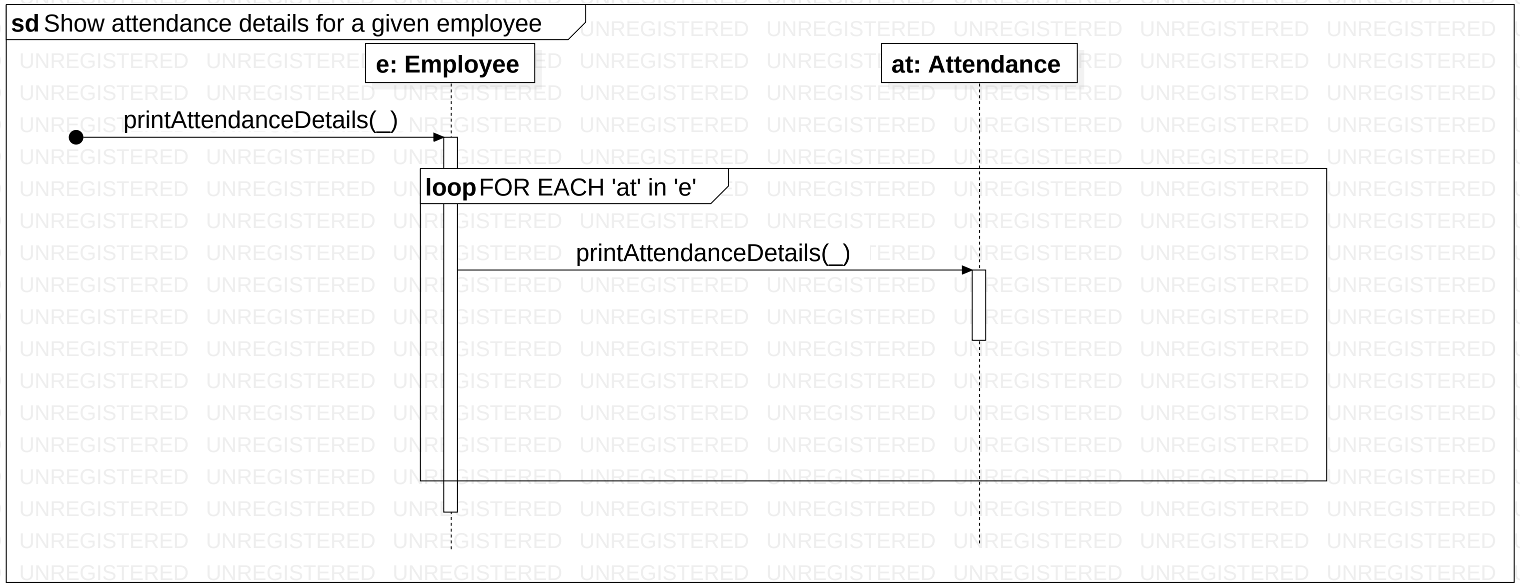
Sequence Diagram 1 (Heading of the sequence diagram specifies the use case covered)

Admin calls generateReport1(..) function of Main and specifies the month and percentage for which to generate report. Main calls getAttendancePercentage(..) function of all its stored employees and specifies the month for which to calculate attendance percentage. Main prints the employee's name if employee returns a percentage less than required. Employee calls getAttendanceMonth(_) function of all its Attendance objects. If the received month is same as the required month the working hours for that attendance are noted for percentage calculation.


sd Show employees with any outstanding leaves
a: Admin
m: Main
e: Employee
l: LeaveApplication
generateReport2(_)
loop FOR EACH 'e' in 'm'
IsThereOutstandingLeave(_)
loop FOR EACH 'l' in 'e'
getLeaveApplicationStatus(_)
alt IF getLeaveApplicationStatus(_) returned 'Pending'
True returned
False Returned
alt IF IsThereOutstandingLeave(_) returned True
getEmployeeName(_)

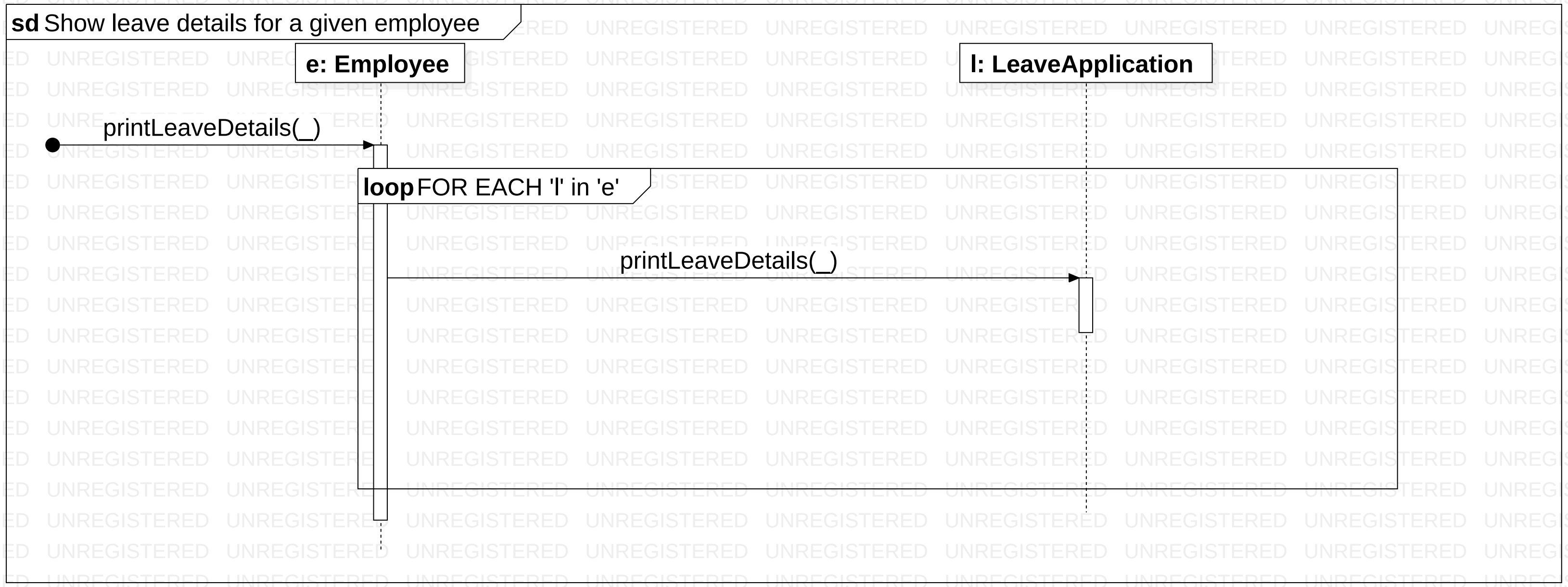
Sequence Diagram 2 (Heading of the sequence diagram specifies the use case covered)

Admin calls generateReport2(_) function of Main. Main calls IsThereOutstandingLeave(_) of all its employees, if true is returnd, Main prints the name of that employee. Employee calls getLeaveApplicationStatus(_) for all its leave applications, if any one leave application returns a pending status, the employee returns True to Main, otherwise returns False


sd Show attendance details for a given employee
e: Employee
at: Attendance
printAttendanceDetails(_)
loop FOR EACH 'at' in 'e'
printAttendanceDetails(_)

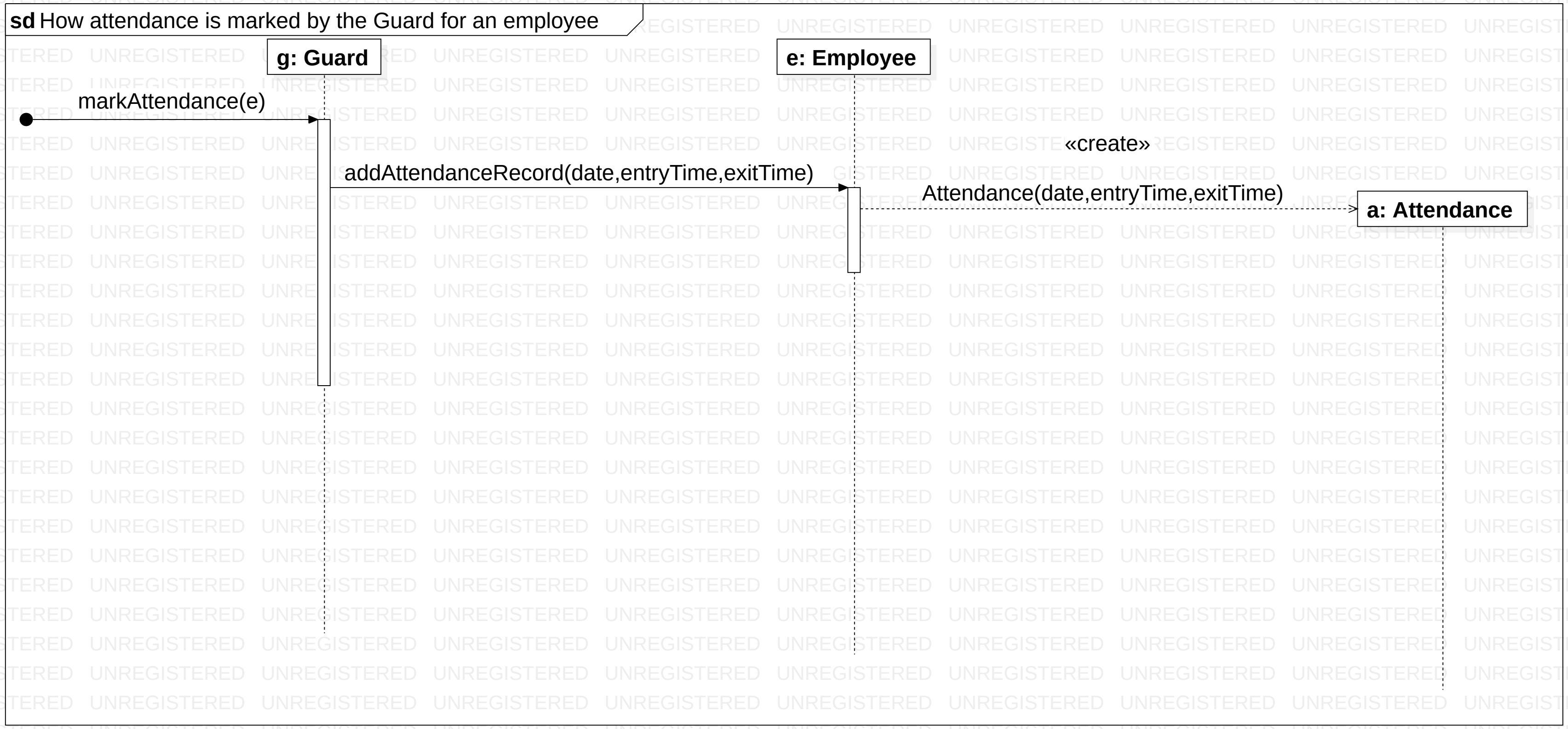
Sequence Diagram 3 (Heading of the sequence diagram specifies the use case covered)

Employee has a function printAttendanceDetails(_), this function calls printAttendanceDetails(_) of all the Attendance objects in the Employee



Sequence Diagram 4 (Heading of the sequence diagram specifies the use case covered)

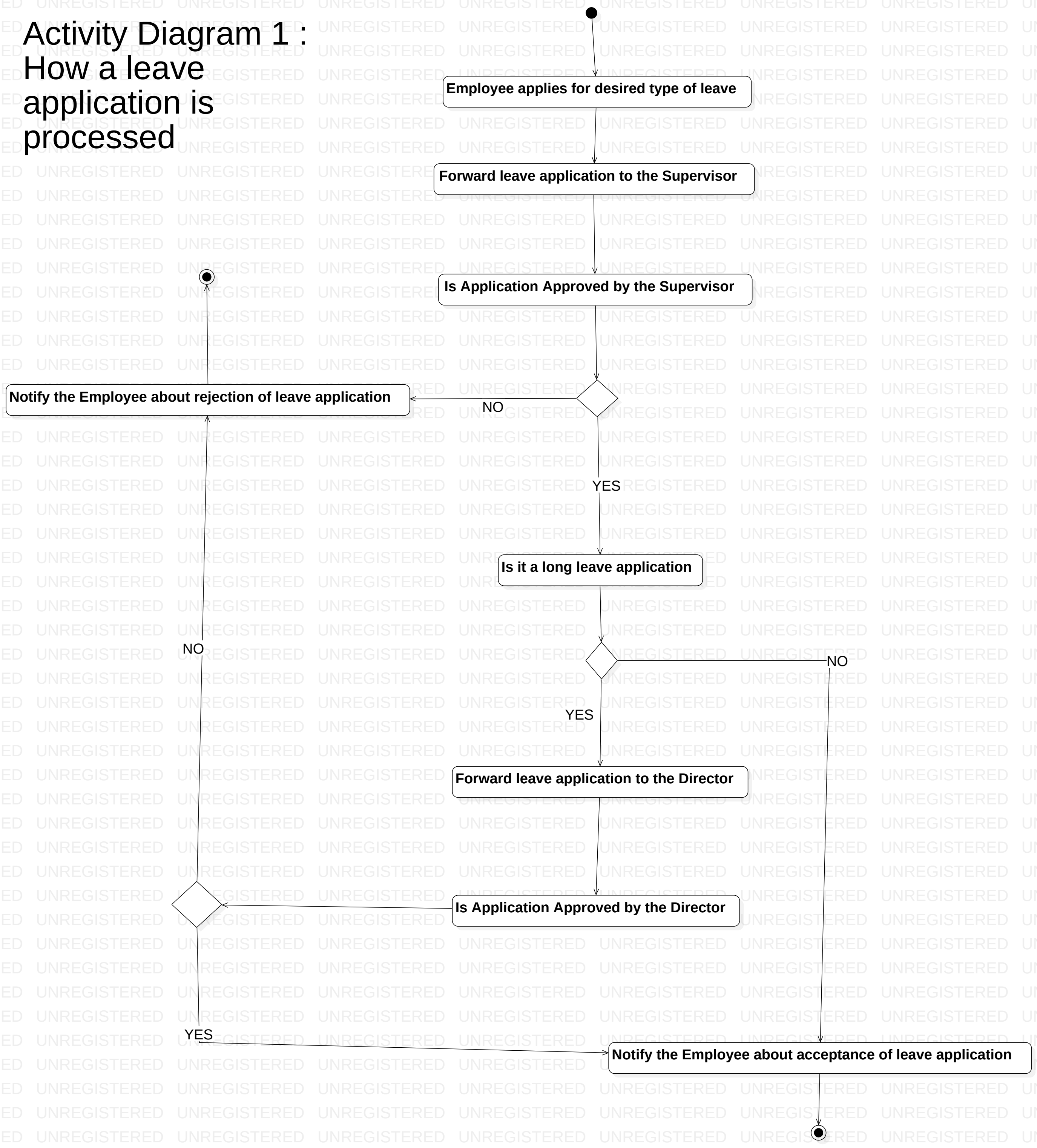
Employee has a function printLeaveDetails(_), this function calls printLeaveDetails(_) of all the LeaveApplication objects in the Employee. After the loop is finished, the function also prints the leave balance of the employee


sd How attendance is marked by the Guard for an employee
g: Guard
e: Employee
markAttendance(e)
addAttendanceRecord(date,entryTime,exitTime)
«create»
Attendance(date,entryTime,exitTime)
a: Attendance

Sequence Diagram 5 (Heading of the sequence diagram specifies the use case covered)

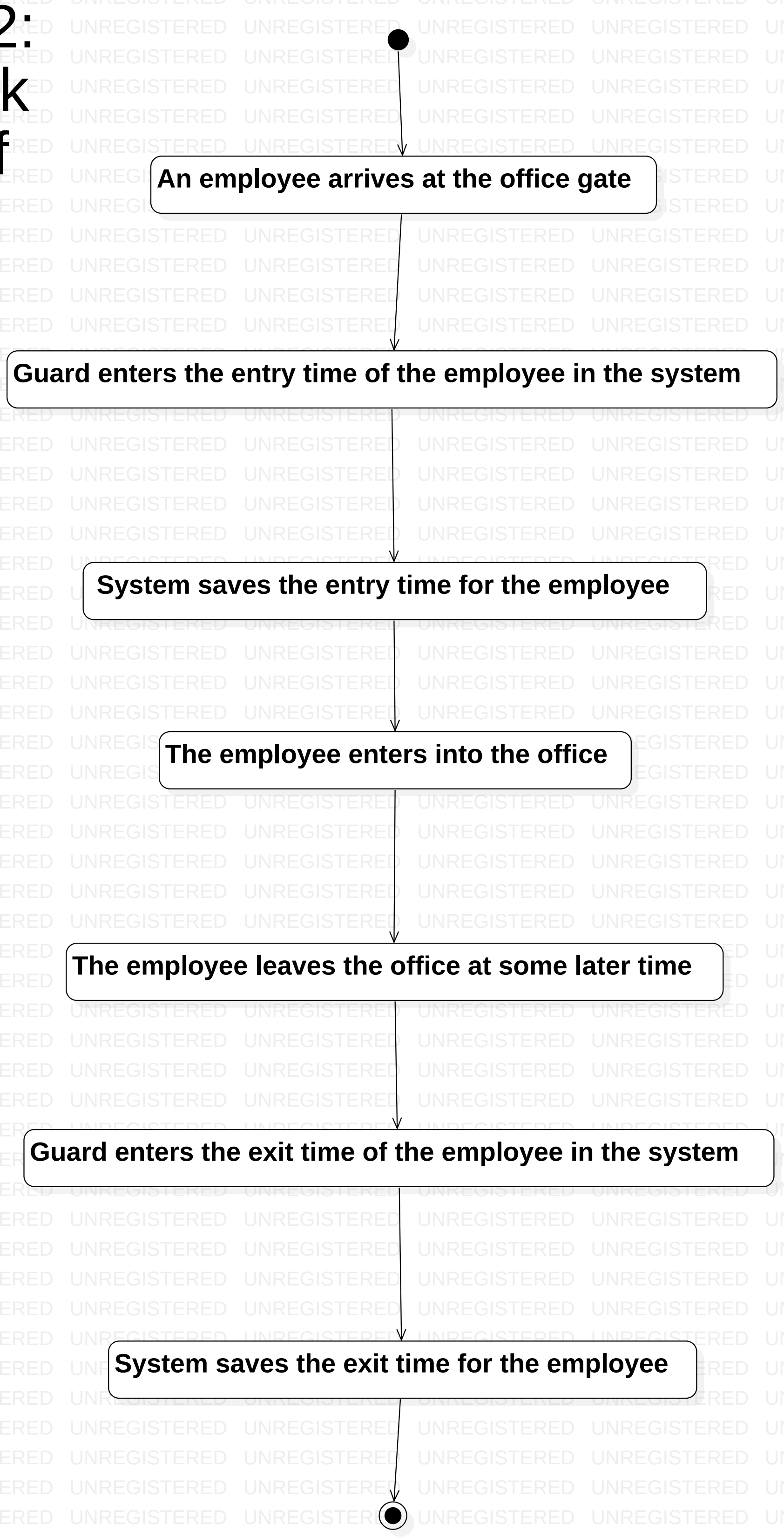
markAttendance(..) function of guard is called with Employee object as parameter. Guard object calls addAttendanceRecord(..) function of the Employee object with date, entryTime, and exitTime as parameters. The Employee object creates a new Attendance object by calling its constructor and saves this new Attendance object in its array of attendance records.

Activity Diagram 1 : How a leave application is processed

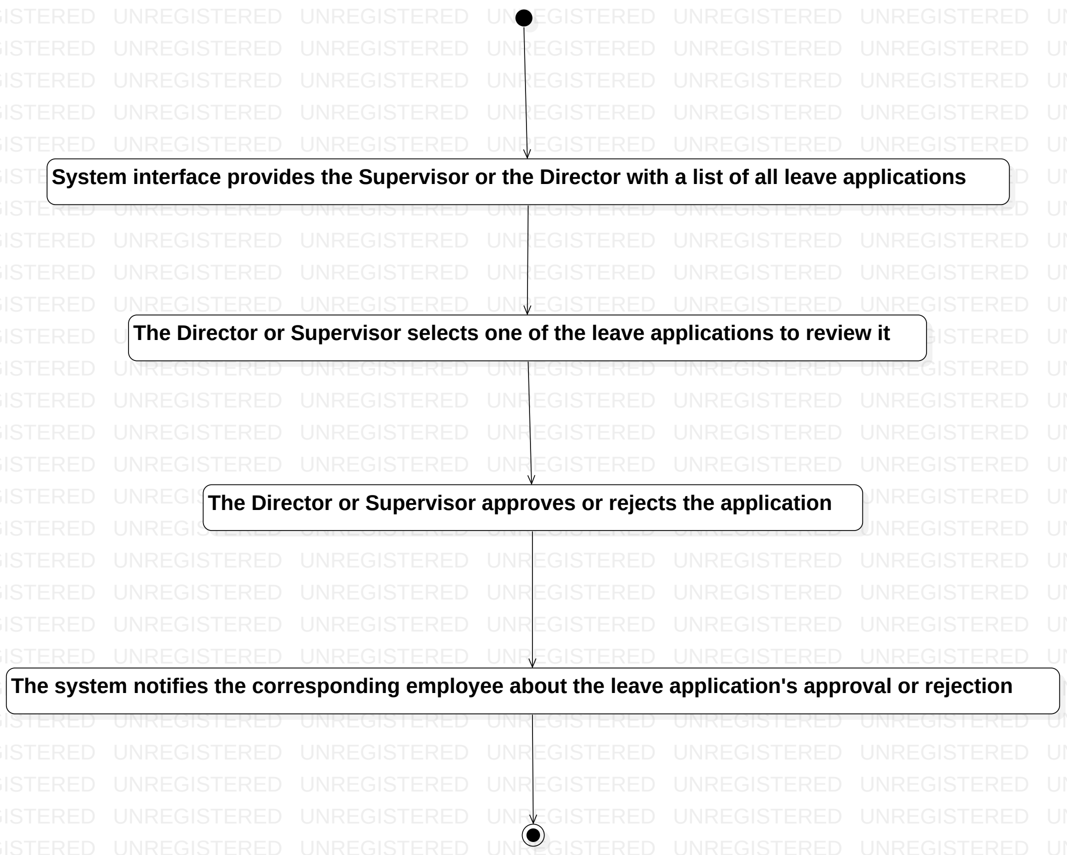


Employee applies for desired type of leave
Forward leave application to the Supervisor
Is Application Approved by the Supervisor
NO
Notify the Employee about rejection of leave application
YES
Is it a long leave application
NO
YES
Forward leave application to the Director
NO
Is Application Approved by the Director
YES
Notify the Employee about acceptance of leave application

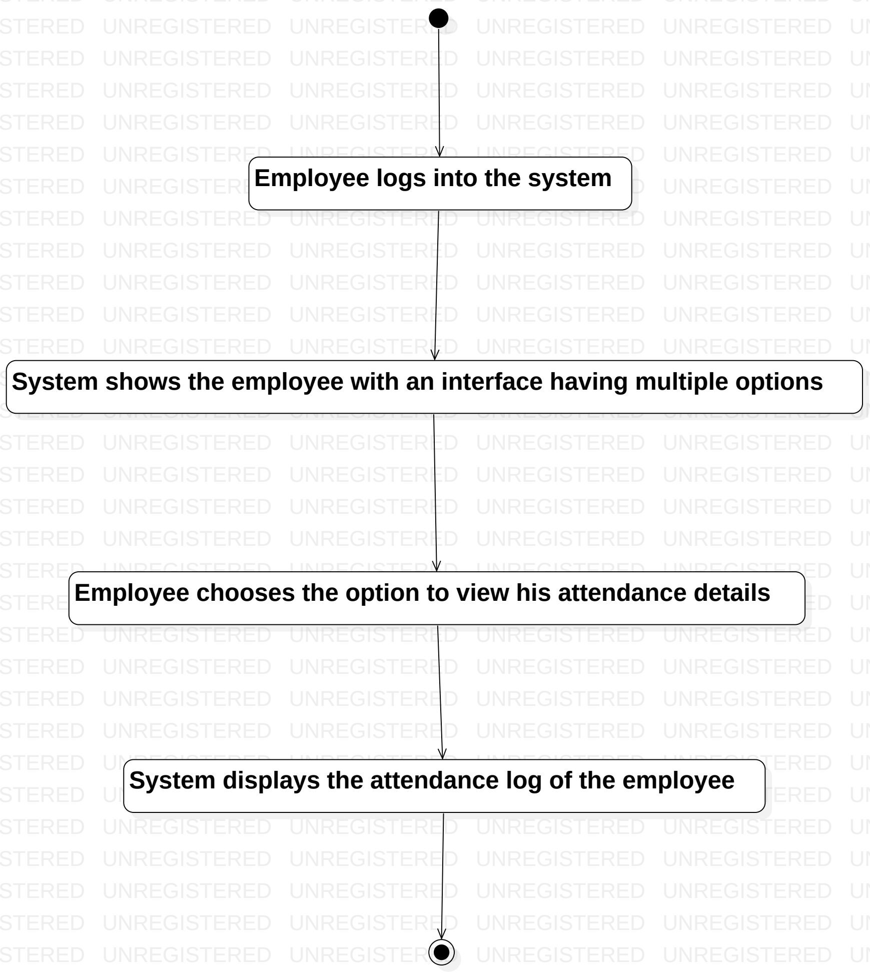
Activity diagram 2: How Guards mark the attendance of employees


An employee arrives at the office gate
Guard enters the entry time of the employee in the system
System saves the entry time for the employee
The employee enters into the office
The employee leaves the office at some later time
Guard enters the exit time of the employee in the system
System saves the exit time for the employee

Activity Diagram 3: How leave applications are approved or rejected

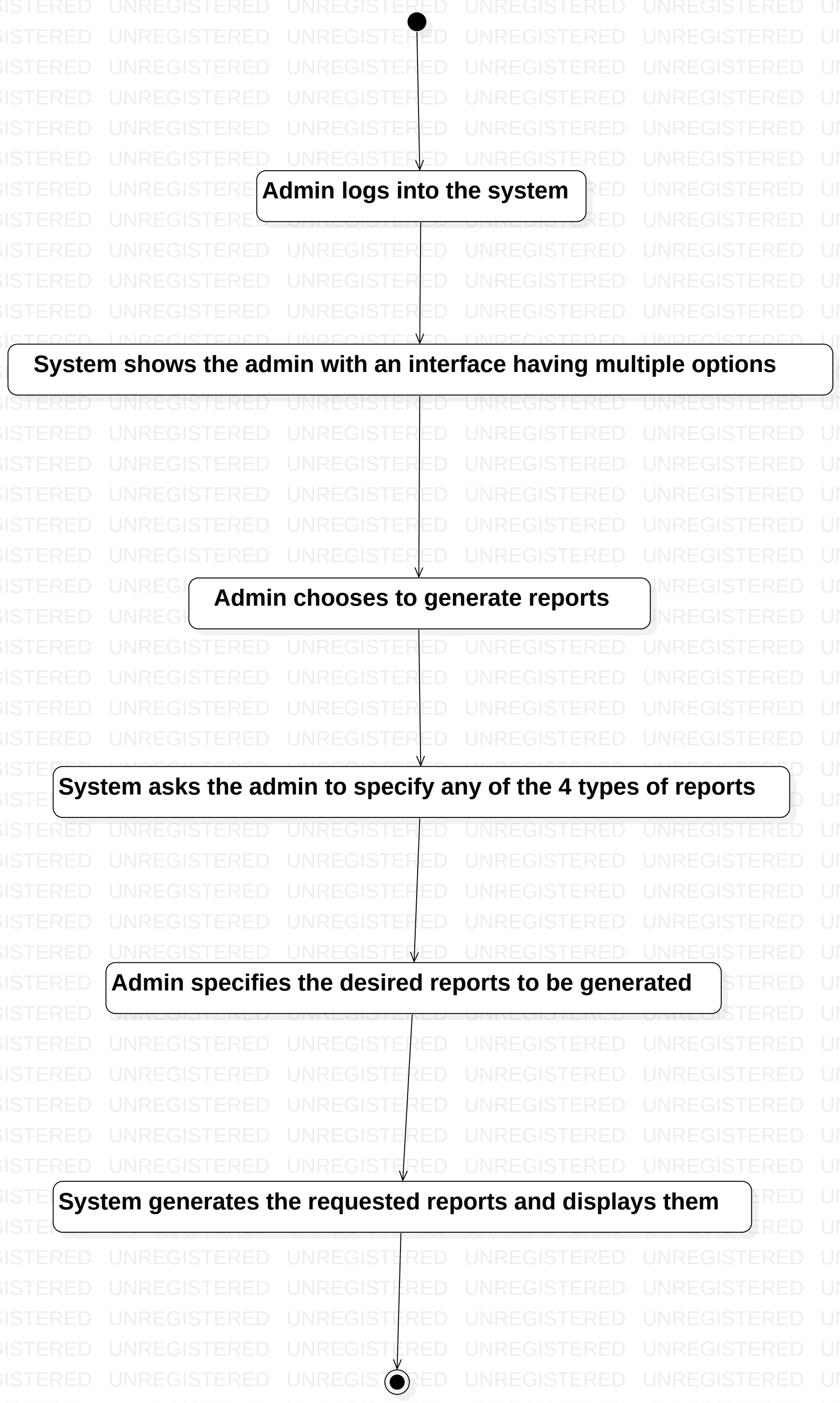

System interface provides the Supervisor or the Director with a list of all leave applications
The Director or Supervisor selects one of the leave applications to review it
The Director or Supervisor approves or rejects the application
The system notifies the corresponding employee about the leave application's approval or rejection

Activity Diagram 4: How an employee views his attendance



Employee logs into the system
System shows the employee with an interface having multiple options
Employee chooses the option to view his attendance details
System displays the attendance log of the employee

Activity Diagram 5: How Admin generates reports


Admin logs into the system
System shows the admin with an interface having multiple options
Admin chooses to generate reports
System asks the admin to specify any of the 4 types of reports
Admin specifies the desired reports to be generated
System generates the requested reports and displays them