Project Part 2

Team: Ashwath Gundepally
Omkar Prabhu
Vipraja Patil

Title: Workforce Management System

Project Summary: A web based application that helps Management and Employees with schedule management, initiating and approving shift change requests and time-offs, and managing payment details for an organization.

Project Requirements

USER REQUIREMENTS			
ID	**Actor**	**Requirement**	**Priority**
US-01	Employee, Manager	As a user, I should be able to login	High
US-02	Manager	As a manager, I should be able to create employee accounts	High
US-03	Manager	As a manager, I should be able to create employee schedules	High
US-04	Manager	As a manager, I should be able to edit employee schedules	High
US-05	Employee	As an employee, I should be able to view public profiles of other employees	Medium
US-06	Employee	As an employee, I should be able to request to exchange my shifts	Medium
US-07	Employee	As an employee, I should be able to request to drop my shifts	Medium
US-08	Manager	As a manager, I should be able to approve of leave/ shift exchange requests	Medium
US-09	Manager	As a manager, I should be able to edit details of the employees	High
US-10	Employee	As an employee, I should be able to view payment details	Low
US-11	Employee, Manager	As a user, I should be able to clock-in and clock-out	High
US-12	Employee, Manager	As a user, I should be able to change the passwords	High
US-13	Employee	As an employee, I should be able to submit my availability for the upcoming work week.	Medium
US-14	Employee, Manager	As a user, I should be able to view schedules	High
US-15	Manager	As a manager, I can view requests for shift exchange, and timeoff	High
US-16	Manager	As a manager, I can review Employee's performance	Low

BUSINESS REQUIREMENTS		
ID	**Requirements**	**Priority**
BR-01	Maximum 100 employee accounts can be created	High

FUNCTIONAL REQUIREMENTS		
ID	**Requirements**	**Priority**
FR-01	Email the employee the link to reset their password	High

FR-02	Email the employees about holidays	Low
FR-03	An alert should be send to the manager if the employee doesn't turn up for their shift	Medium
FR-04	Email the employees about their performance status	Low

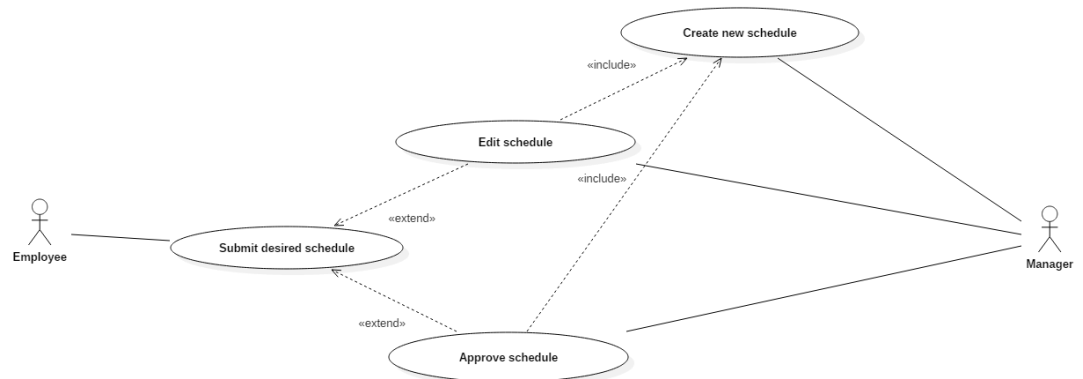| NON-FUNCTIONAL REQUIREMENTS		
ID	**Requirements**	**Priority**
NFR-01	Store salted hash of the password in the database	Medium
NFR-02	If the attempts to login exceeds more than 5 times, the system will block the user account	Low
NFR-03	The system should be able to handle 20 concurrent employee logins	Low

Use cases

a. Use Case Overview
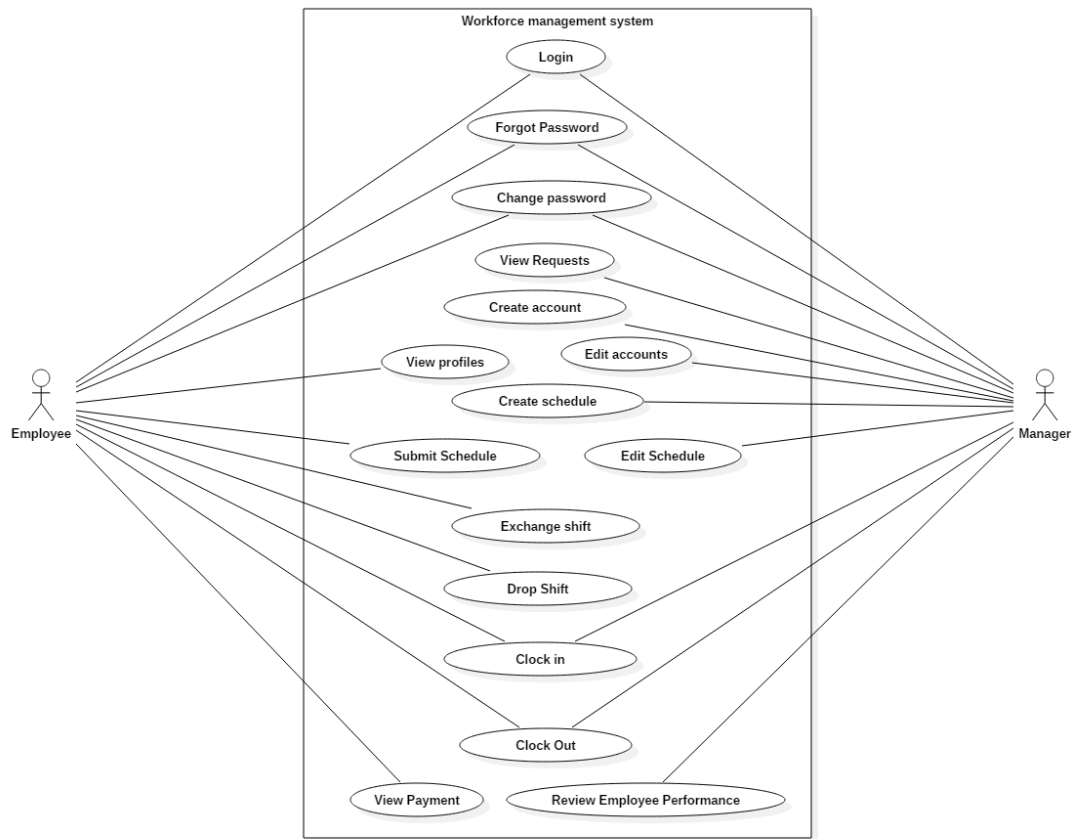
i.   Sub-diagram

Creation of new schedule

Exchange or Drop Shifts
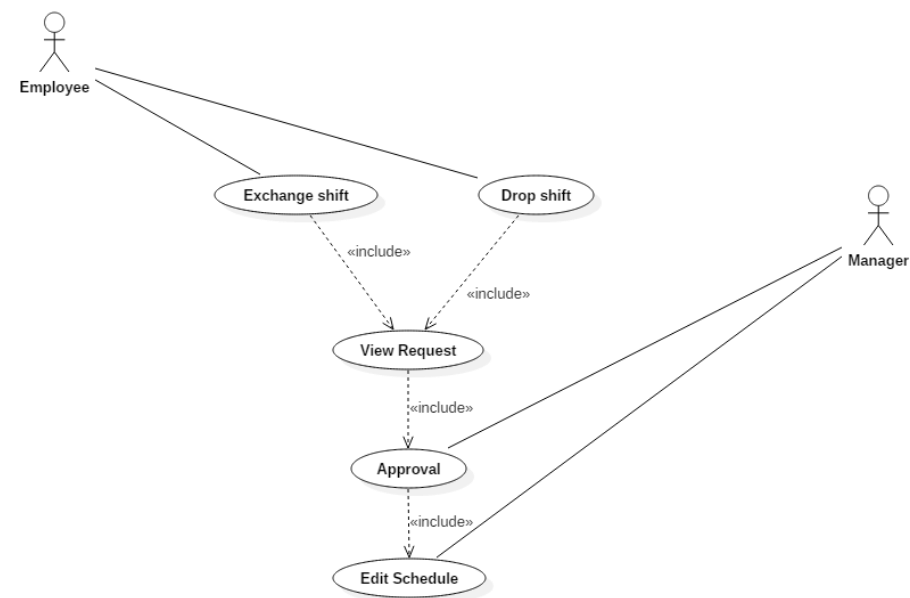
b. Use case documents

Use Case ID:	UC 01
Use Case Name:	Create Schedule
Description:	Let's the manager create a new schedule

Primary actor	Manager		
Secondary actor	Employee		
Precondition	Employee should exist in the database		
Postcondition	New schedule is created		
Frequency of use	Weekly		
Flow of events		Actor action	System response
	1	Employee submits the desired schedule	
	2	Manager will edit this submitted schedule as per organizational needs	
	3	Create a new schedule for the employee	Update schedule on the portal
Variations	If the employee fails to submit the schedule, Manager will automatically create the schedule.		
Exceptions	None		
Developer's notes	None		

Use Case ID:	UC 02
Use Case Name:	Drop Shift
Description:	Employees can request to drop their shifts

Primary actor	Employee		
Secondary actor	Manager		
Precondition	Employee request should exist in the database		
Postcondition	Drop request is denied/approved		
Frequency of use	Random		
Flow of events		Actor action	System response
	1	Employee requests for dropping shift	System will create a request notification on portal
	3	Manager will view request	
	4	Manager approves request	Update employee schedule on the

		portal
Variations	Manager could deny this request	
Exceptions	None	
Developer's notes	None	

Use Case ID:	UC 03
UCase Name:	Exchange Shift
Description:	Employees can request to exchange their shifts

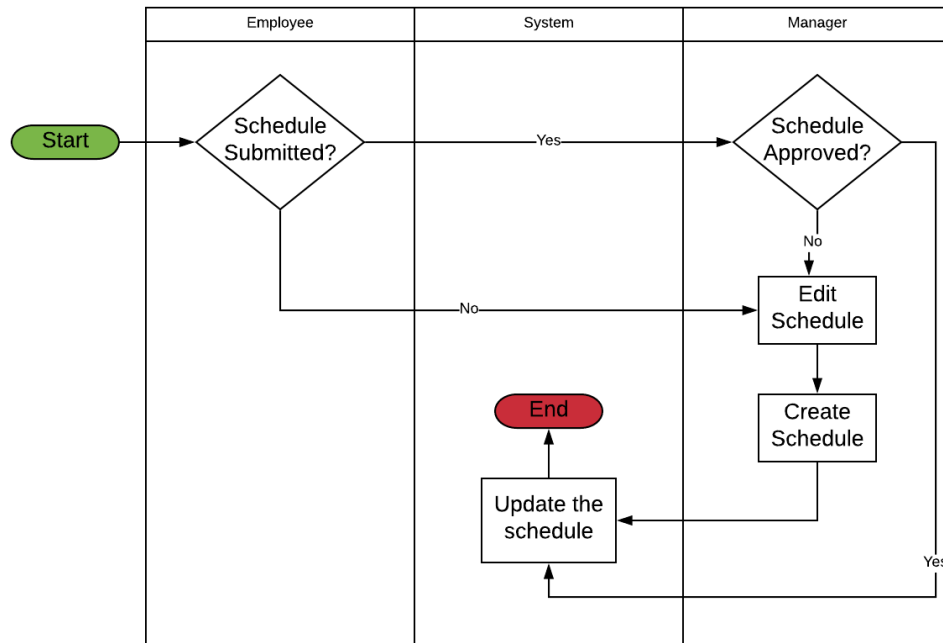| Primary actor	Employee		
Secondary actor	Manager		
Precondition	Employee should exist in the database		
Postcondition	Exchange shift request is denied/approved		
Frequency of use	Random		
Flow of events		Actor action	System response
	1	Employee requests for exchanging shift	System will create a request notification on portal
	3	Manager will view request	
	4	Manager approves request	Update employee schedule on the portal
Variations	Manager could deny this request		
Exceptions	None		
Developer's notes	None		

Activity Diagram

a. Activity diagram for creating new schedule

b. Activity diagram for Exchange shift
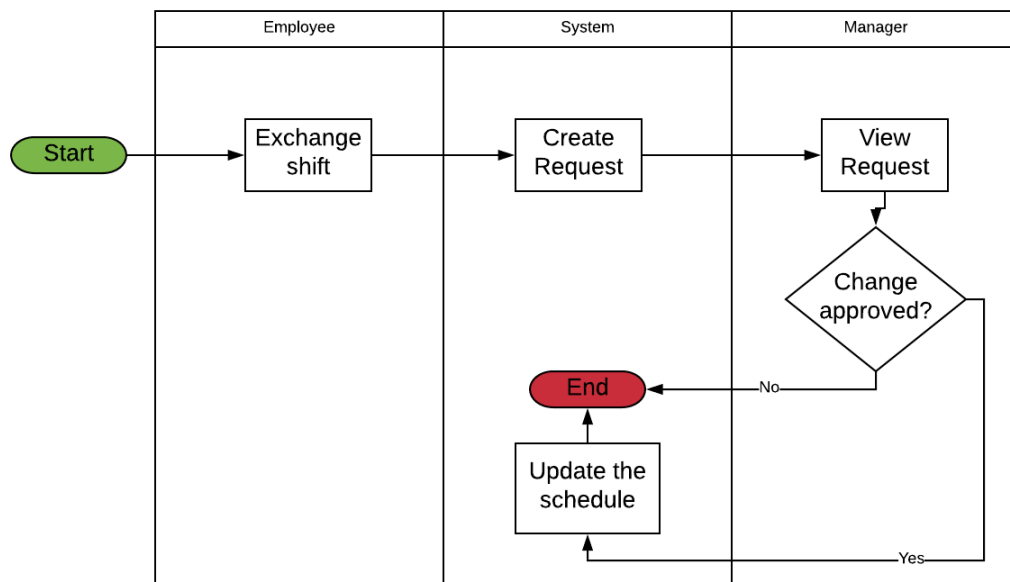
c. Activity diagram for Drop shift

Employee	System	Manager

Start → Drop shift → Create Request → View Request

Change approved?

No → End

Yes → Update the schedule → End

UI Mockup

   a.   Login Page

b. Employee Dashboard

VIOA   Profile
Welcome, Adam   Sign Out

My Dashboard
Employee Search
My Schedule   Clock In   Clock Out
Exchange Shift
Drop Shift
View Payment

c. Employee – Employee search

VIOA   Profile
Welcome, Adam   Sign Out

My Dashboard   Employee Search
Employee Search   Name: [   ]
My Schedule
Exchange Shift   Search Result:
Drop Shift
View Payment

d. Employee - View Schedule

VIOA   Profile

Welcome, Adam   Sign Out

my Dashboard		Schedule for this week		
Employee Search				
my Schedule		Date	Day	Time
Exchange Shift		01/12/18	Monday	9-12pm
Drop Shift		01/13/18	Tuesday	2-3 pm
View Payment		01/15/18	Thursday	8-12 am

e. Manager Dashboard

VIOA   Profile
Welcome, Manager   Sign Out

My Dashboard
Edit accounts
View Schedules   (clock IN)   (Clock Out)

[Requests]
Adam: Drop Shift
Liza : Exchange Shift
Peter: Drop Shift
Amy : Exchange Shift

f. Manager - Drop shift page

VIOA   Profile

Welcome, Adam   Sign Out

My Dashroom	Drop Shift			
Employee Search				
My Schedule	Date	Day	Time	Drop
Exchange Shift	01/12/18	Monday	9-12pm	X
Drop Shift	01/13/18	Tuesday	2-3pm	X
View Payment	01/15/18	Thursday	8-12am	X

[okay]

User Interactions

Requirement ID: US-03
Use Case ID: UC-01
Use Case name: Create Schedule
Description: Every week the employee will submit a desired schedule, the manager will either approve the schedule or edit the given schedule according to the organizational needs.
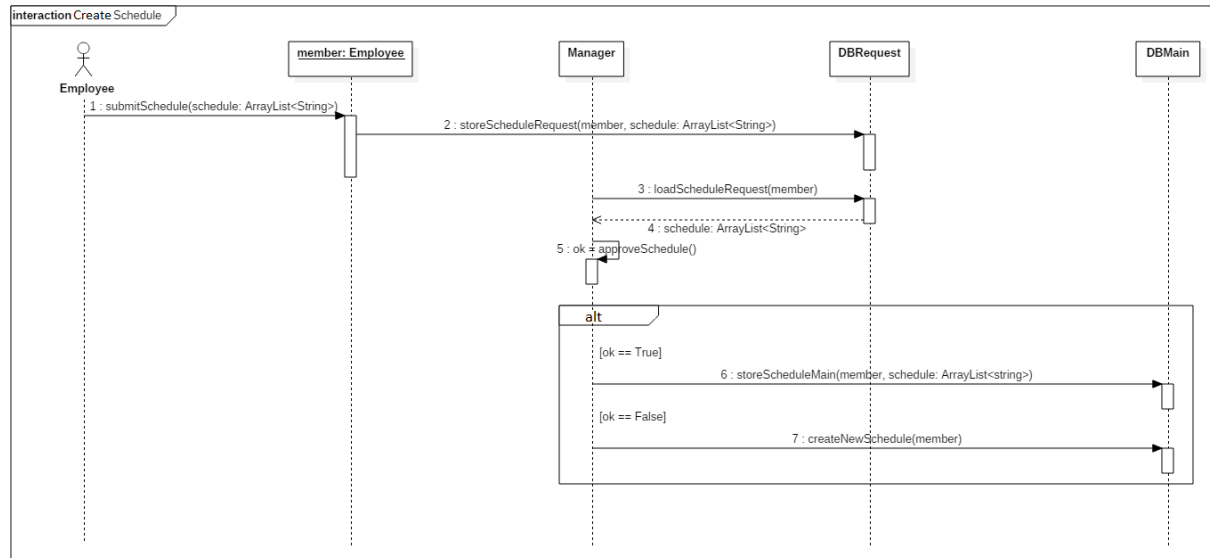
Requirement ID: US-08
Use Case ID: UC-07
Use Case name: Drop Shift
Description: Employee can request to drop shift

interaction Drop Shift

Employee → member: Employee → :Request → :Manager → :DBRequest → :DBMain

1 : dropShift(schedule: ArrayList<String>)
2 : createRequest(member)
3 : storeScheduleRequest(member, schedule: ArrayList<String>)
4 : viewRequest(member)
5 : loadScheduleRequest(member)
6 : schedule: ArrayList<String>
7 : ok = approveSchedule()
8 : [ok]storeScheduleMain(member, schedule: ArrayList<String>)
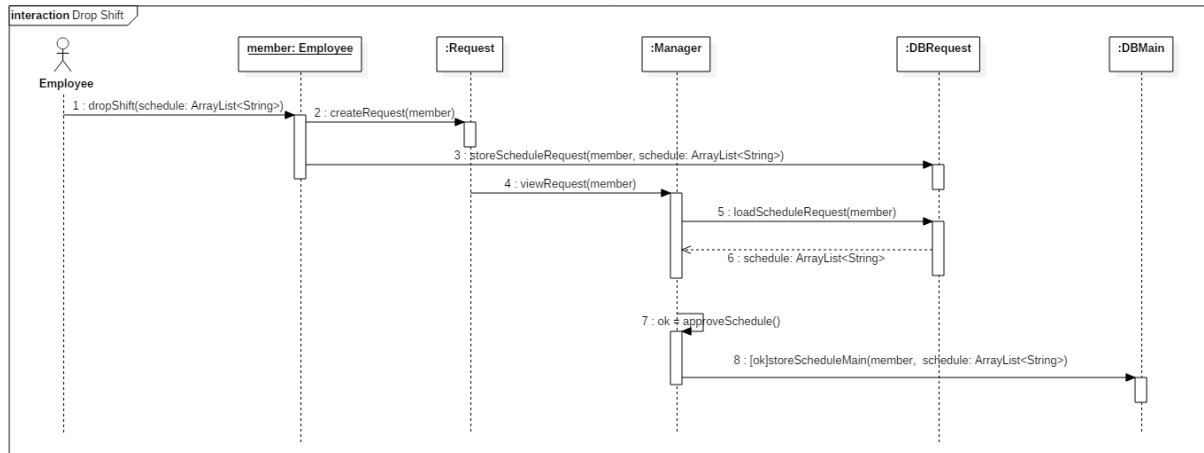
Requirement ID: US-08
Use Case ID: UC-06
Use Case name: Drop Shift
Description: Employee can request to exchange shift

interaction Exchange shift

Employee → member: Employee → :Request → :Manager → :DBRequest → :DBMain

1 : exchangeShift(schedule: ArrayList<String>)
2 : createRequest(member)
3 : storescheduleRequest(member, schedule: ArrayList<String>)
4 : viewRequest(member)
5 : loadScheduleRequest(member)
6 : schedule: ArrayList<string>
7 : ok = approveSchedule()
8 : [ok]storeScheduleMain(member, schedule: ArrayList<String>)

Class Diagram

User
- userName: String
- userPwd: String
- UserDetails: PersonalDetails
- UserRole: String

+login(userName: String, userPwd: String): Boolean
+logout(): Boolean
+clockIn(userName: User): Time
+clockOut(userName: User): Time
+getUserName(): String
+setUserName(userName: String): void
+getUserPwd(): String
+setUserPwd(userPwd: String): void
+forgotPwd(): void
+changePwd(newPwd: String): Boolean

«interface» ViewProfile
+viewProfile(): void
+viewPayment(): void
+viewSchedule(): void

TimeSheet
+storeClockInTime(employeeName: Employee): void
+storeClockOutTime(employeeName: Employee): void
+loadClockInTime(): int
+loadClockOutTime(): int
+CalculateTotalHours(int Hours, int TotalHours): void
+LoadTotalHours(): int
+CalculateHours(Time timein): int

PersonalDetails
- name: String
- dateOfBirth: Date
- phoneNumber: long
- personalEmailID: String

+getName(): String
+setName(name: String): void
+getDateOfBirth(): Date
+setDateOfBirth(dateOfBirth: Date): void
+getPhoneNumber(): int
+setPhoneNumber(phoneNumber: long): void
+getPersonalEmailID(): String
+setPersonalEmailID(personalEmailID: String): void
+displayPublicProfile(): void

Notification
+holidayAler(date: Date): void
+sendAlert(alertMsg: String): void
+sendEmployeeAlert(alertMsg: String, employeeName: Employee): void

Employee
- EmployeePoints: int
- PayRate: float
- EmploymentStatus: String
- schedule: ArrayList<String>
- timeIn: int
- totalHours: int
- timeOut: int

+submitSchedule(schedule: ArrayList<String>): void
+exchangeShift(schedule: ArrayList<String>): void
+dropShift(schedule: ArrayList<String>): void
+getTimeIn(): int
+getTimeOut(): int

Request
+createRequest(Employee employeeName): void

Manager
+createNewSchedule(): void
+approveSchedule(): Boolean
+editSchedule(): void
+createNewAccount(newName: String): Boolean
+viewRequest(employeeName: Employee): void
+giveEmployeeReviews(employeeName: Employee, intEmployeePoints): EmployeePoints
+editProfile(employeeName: Employee): void

DBRequest
+loadScheduleRequest(employeeName: Employee): ArrayList<String>
+storeScheduleRequest(employeeName: Employee, schedule: ArrayList<String>): void

DBMain
+loadScheduleMain(employeeName: Employee): ArrayList<String>
+storeScheduleMain(employeeName: Employee, schedule: ArrayList<String>): void

1..*