Shift Management System

Date: April 10th, 2024

Table of Contents

Introduction/Overview - Document Information

Document Authors

Linh Nguyen Le

Document Purpose

The purpose of this document is to organize the team and provide a detailed breakdown of system features, UX and systems design, and use case solutions.

Indented Audience

The intended audience is the users using and experiencing this application as well as development team for future reference.

Project Overview

Project proposal

As the business is thriving, ABC business is getting more and more line workers and thus its schedule system is becoming more and more outdated. As the result, the company is looking for how to optimize the working process and when it comes to workload optimization it is not as advanced and well organized as the other fields. Dozens of workers are facing the issue of quickly finding someone to replace them for the upcoming shift. As well as managing their tasks and deadlines. This means that there is a demand for an application that can handle employees scheduling and allows seamless communication between line workers and managers .

Problem Statement

The Problem of:	The current systems to manage workers shifts are overly complex and slow.
Affects:	Shift workers, employee managers
The impact of which is:	Empty shifts cannot be filled quickly enough. Last-minute holes in a schedule cannot be filled by other employees before the scheduled shift has already started, and employees with open schedules cannot easily find openings.
A successful solution would:	Would allow employees to drop and take shifts with oversight or approval from their managers and enable efficient shift scheduling and fulfillment.

Product Vision

For	ABC Business
Who	Need a system to efficiently manage employee shifts.
Workplace Shift Management Tool	Is a web application.
That	Allows for easy shift management, employee/employer communication, and more.
Unlike	Current day systems, which require multiple separate applications to function.
Our product	Will combine the functionality of current day systems into one cohesive package.

Stakeholders and Users

Stakeholder Name/Identifier	Category
PRJ666 Professor	Sponsor
Business that purchases our software.	ABC Business
Low-level employees (line workers, cooks, etc.)	User
General managers, managers	Manager
System administrators/support	Administration
Project Leader	Developers
Project developers	Developers

Functional Requirements

1. The system shall allow login for existing users, both administrators and for regular employees
2. The system shall allow different privileges based on user type.
    2.1. The system shall allow managers to modify their own schedules and tasks.
    2.2. The system shall allow managers to modify all schedules and tasks of all employees working under them, as well as the creation or deletion of tasks.
3. The system shall allow users to create or remove other users based on user privileges.
    3.1. The system shall allow managers to remove or create workers profiles.
    3.2. The system shall allow administrators to remove or create managers.
4. The system shall allow task interaction:
    4.1. The system shall allow managers to create tasks with specified details for employees.
        4.1.1. The system shall allow the specification of time for tasks.
        4.1.2. The system shall allow the specification of task deadlines.
        4.1.3. The system shall allow for the setting of repeated tasks.
            4.1.3.1. The system shall allow repeated tasks to be set for each day.
        4.1.4. The system shall allow for setting task descriptions.
        4.1.5. The system shall allow for tracking of task completion status.
    4.2. The system shall allow managers to update details of tasks

4.3. The system shall allow employees to confirm task completion.
4.4. The system shall allow managers to delete tasks.
4.5. The system shall allow managers to assign tasks to individual employees.

5. The system shall allow scheduled interaction.
    5.1. The system shall allow managers to assign shifts manually/automatically to employees.
    5.2. The system shall allow managers to modify the timeslots for shifts to be assigned to
    5.3. The system shall allow employees to claim open shifts/tasks.
6. The system will have an open shifts list, where any shift can be picked up by an employee with matching status and availability for a given shift.
7. The system will have a schedule generator for the selection of employees.
8. The system will generate reports of:
    8.1. Missed shifts with option to re-schedule them.
    8.2. Uncompleted tasks with the option to send a reminder or assign new employee(s).
    8.3. Performance based on time spent on tasks and general workload (being late, spent too much time on lunch, total number of shifts this month) for each employee.

Nonfunctional Requirements

1. Operational Requirements
    a. The system will be written and served as a web application, accessible via web browsers.
2. Performance Requirements
    a. The system will load and function on desktop web browsers.
3. Security Requirements
    a. The system will only allow employees to access employee-related functionality.
4. Data Integrity
    a. The system will only collect basic employee information.
    b. Data regarding employees and shift management details will be encrypted and stored in a secure database.
5. Localization Requirements
    a. The web application will only be localized for English.

Project Scope

Project Goals/Objectives:

The goal of this project is to create a singular application that can implement all the functionalities required for managing employees, as opposed to current methods which require handling multiple disparate applications that add up to what is necessary.

- Employers will be able to efficiently manage employees.
- Employees will be able to easily communicate ability to take shifts.
- Remove need for multiple applications to achieve same results.
- Remove need to submit identical materials on multiple platforms, thus reducing chance of materials not being submitted to necessary locations

Project Boundaries

Within scope:

- Create a web app that implements a system for handling employees and employee shift details.
- Employers can create new accounts and remove accounts for new or prior employees respectively and set available employees to open shifts.
- Employees can specify which shifts they will be available to take, as well as communicate with employers and other employees regarding shift details and needs.
- Notifications on shift detail updates will be sent to appropriate parties via system notifications and emails.
- Provide tutorials and documentation on system use.

Project Deliverables:

1. Front-end web application
2. Back-end application
3. Database
4. Cloud Storage

Success Criteria:

- Employers and employees can successfully manage and communicate details regarding management updates and shift changes.

Project Assumptions:

- The team is responsible for the application launch, with operational systems and processes.

- The application will provide intuitive control.

System Risks

Risk	Response
This web app system will allow a lot of users within one organization with multiple features for every employee and employer. Thus, performance might be a big concern and risk.	The team will choose the most efficient architecture with the best optimization as well as the programming language and database combination. These steps would ensure the best response time and overall performance.
Organizational risks. The team had a hard start and still has issues with communication.	The team needs to meet all together and consider changing the approach.
Development risk. Team members may refuse to work or deliver features late	Have a stand-up every even day to report status. If the action does not improve team members' performance, professor will be notified

Operating Environment

Operating environment for the Workplace Shift Management Tool system will consist of:

- Server Operating System: Linux/UNIX
- SQL based distributed database with necessary data integrity constraints
- Supported Browsers:
    - Google Chrome
    - Firefox
    - Safari
- Platform:
    - Programming Framework: React
    - API: NodeJS
    - Language: JavaScript, TypeScript

Process and Data Modeling

Business Rules

Business Rule Number	Business Rule Description	Related UC
BR01	Users must be logged in to access resources	All
BR02	Accounts can only be created with a lower privilege than their creator.	UC04
BR03	Shifts must be unique	UC03
BR04	Shifts cannot be assigned to multiple people at once	UC01
BR05	Shifts cannot overlap on one person	UC01, UC06
BR06	Only managers can assign shifts to other employees	UC01
BR07	Email address attached to users must be unique	UC04
BR08	Generated schedules should not be unreasonable*	UC06

System Use Case Diagrams

Use Case Name: Add Shift to Schedule	ID: 1	Importance Level: High
Primary Actor: Manager	Use Case Type: Detail, Essential	
Stakeholders and Interests: Manager – wants to set a shift to a specific timeframe Employees – want shifts to take place during manageable time windows		
Brief Description: This use case describes how a manager-level employee can assign shifts to the company schedule.		
Trigger: Manager wants to assign a shift to an unused time window Type: External		
Relationships: Association: Manager Include: Extend: Generalization:		
Normal Flow of Events: 1. Retrieve current schedule 2. Select unused window of time in the schedule 3. Retrieve list of employee tasks 4. Select tasks to be assigned to new shift 5. Add shift to schedule		

Subflows:
Alternate/Exceptional Flows: 1. Manager cancels the add shift sequence

Use Case Name: Delete Shift from Schedule	ID: 2	Importance Level: High
Primary Actor: Manager	Use Case Type: Detail, Essential	
Stakeholders and Interests: Manager – wants to remove a shift from the current schedule		
Brief Description: This use case describes how a manager-level employee can delete shifts from the company schedule.		
Trigger: Manager wants to remove an unnecessary shift from the schedule Type: External		
Relationships: Association: Manager Include: Extend: Generalization:		
Normal Flow of Events: 1. Retrieve current schedule 2. Select shift that is to be deleted 3. Delete selected shift		
Subflows:		
Alternate/Exceptional Flows: 1. Manager cancels the delete shift sequence		

Use Case Name: Edit Existing Shift	ID: 3	Importance Level: High
Primary Actor: Manager	Use Case Type: Detail, Essential	
Stakeholders and Interests: Manager – wants to edit a shift on the current schedule Employees – may need shift details altered		
Brief Description: This use case describes how a manager-level employee can edit existing shifts that are currently on the company schedule		
Trigger: Manager wants to edit a shift that requires alteration(s) Type: External		
Relationships: Association: Manager Include: Employee Extend: Generalization:		

Normal Flow of Events: 1. Retrieve current schedule 2. Select shift that is to be edited 3. Enter new timeframe for current shift 4. Select tasks to remove from current shift 5. Select tasks to add to current shift 6. Save shift to schedule
Subflows:
Alternate/Exceptional Flows: 1. Manager cancels the edit shift sequence

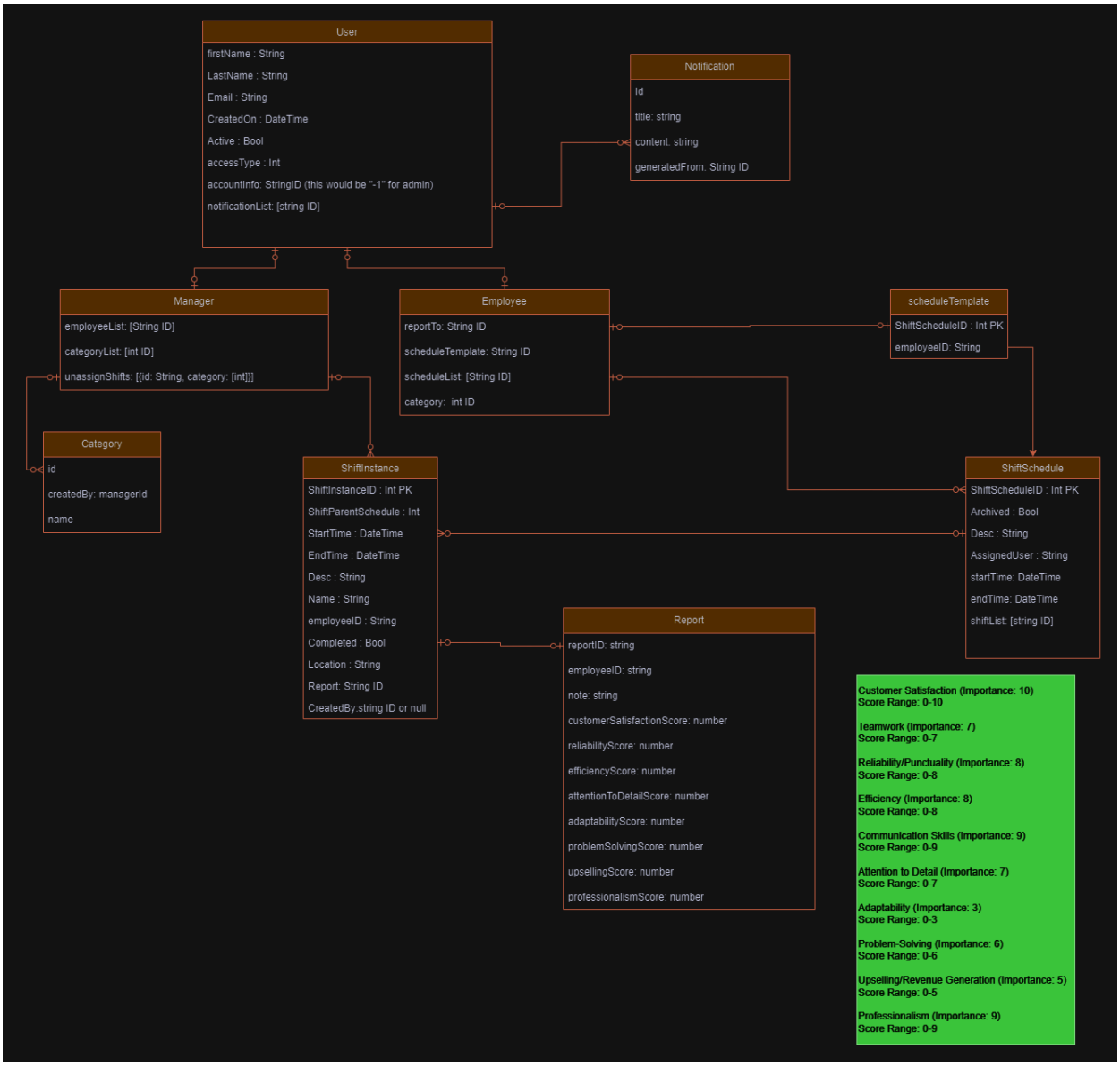
Use Case Name: Manager Registers New Employee	ID: 4	Importance Level: High
Primary Actor: Manager	Use Case Type: Detail, Essential	
Stakeholders and Interests: Manager – Wants to register a new employee Employee – Wants to be able to log in to system		
Brief Description: This use case describes how a manager registers a new employee into the system.		
Trigger: Manager wants to register a new employee Type: External		
Relationships: Association: Manager Include: Employee Extend: Generalization:		
Normal Flow of Events: 1. Generate employee ID 2. Enter employee data 3. Save new employee to database		
Subflows:		
Alternate/Exceptional Flows: 1. Manager cancels the employee registration		

Use Case Name: Log In	ID: 5	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: User – Wants to log in		
Brief Description: This use case describes how a user (manager, employee, etc.) will initially log in to the system		

Trigger: User – Wants to log in to system Type: External
Relationships: Association: User Include: Extend: Generalization:
Normal Flow of Events: 1. Enter user credentials 2. User is authenticated
Subflows:
Alternate/Exceptional Flows: 1. User cancels log-in 2. User fails authentication

Use Case Name: Generate Schedule	ID: 6	Importance Level: Medium
Primary Actor: Employee	Use Case Type: Detail, Essential	
Stakeholders and Interests: Employee – Wants to create a schedule quickly		
Brief Description: This use case describes an employee creating a schedule		
Trigger: User – Wants to log in to system Type: External		
Relationships: Association: Employee Include: Extend: Generalization:		
Normal Flow of Events: 1. Generate a list of possible schedules 2. Select schedule 3. Save new schedule to employee		
Subflows:		
Alternate/Exceptional Flows: 1. Employee cancels		

Database



Retrospective

Problem

- Project Scope: Project scope was too big for a small team with most people having the goal of passing the course instead of creating something great and lasting.
- Teamwork:
    - Assumption can be deadly: In the beginning, we assumed that everyone had the required programming skills as well as a great mindset and a desire to

contribute, to make the project reach its minimum delivery state. However, this proves to be wrong as issues popped up in the very first sprint.

- Daily communication is not enough: The balance between keeping track of team members without too much intervention proved to be difficult as teammates refused to communicate on time as well as report accurate status to get support or to modify project scope.

Potential solutions

- Assess teammate skillset carefully before generating project scope.
- Understand individual's goals to adjust the project scope accordingly.
- Daily stand-up may require more interaction such as showing evidence or describing status in live meeting instead of chat messaging.