EPMS

Employee Management System

Dwayne Roark

Table of Contents

Table of Contents Cont.

Proposal

Problem statement:

Efficient employee management is an important goal for any company. Methods such as manual record-keeping or spreadsheets have their downfalls and are prone to input errors. They can also be time-consuming and hard to scale.

Objective:

This proposal aims to implement a solution that will streamline employee management with an easily scalable system that also implements a system of checks to help enhance data accuracy and provides a comprehensive solution for managing employee information.

System Requirements:

Tie together employee and employment attributes in a user-friendly interface. Employee attributes include things like name, sex, birth date, SSN, contact number, address, and email. Employment attributes include things like ID, hire date, location, and pay rate.

- This would allow HR to enter new employees quickly and correctly.
    - Database field constraints can be used to ensure data types and ranges.
- As well as allowing HR or managers to search for employees by different attributes.
    - Such as name or SSN.

Customer Base:

The customer base would be anyone performing HR/managerial functions in businesses, government, and educational institutions, ranging from small with around 10 employees to very large with 1000s or more.

Hardware Requirements:

A web server hosting a SQL database with network connectivity between end-user PCs and the web server.

Software:

The user interface (front end) will be a simple HTML/CSS web page with an Apache PHP back end communicating with a MariaDB database.

Network:

A typical company internal network of 100mbps or 1gbps is more than sufficient, with standard 99.99% SLA uptime.

Project/Development Plan:

Week 1-2: Server setup and installation, ensuring communication between front-end, back-end, and database. Database tables are established with attributes.

Week 3-4: General front-end design established and plan for communication through backend established.

Week 5-6: Authentication system established.

Week 7-8: Working test product and demo recorded for midterm.

Week 9-11: Implement improvements based on customer feedback.

Week 12-14: Writing test cases. Finalizing touches.

Week 15: Demo for final.

Customer Problem Statements & System Requirements

Customer Problem Statement:

Employee management is the act of logging employee personal information and company-specific information in a singular location. Employee attributes include things like name, sex, birth date, SSN, contact number, address, and personal email. Employment attributes include things like employee ID, hire date, location, and pay rate.

Efficient employee management is an important goal for any company. Methods such as manual record-keeping or spreadsheets have their downfalls and are prone to input errors. They can also be time-consuming and hard to scale. A proper employee management system (EMS) would include an easy-to-use interface that allows for data entry, employee lookup, entry modification, and record deletion.

Glossary of Terms:

EMS – Employee Management System

EID – Employee ID

SSN – Social Security Number

Pay Rate – List of predefined pay bands showing the current salary range of that employee.

Functional System Requirements:

Req Number	Priority Weight	Description
FR1	High	Data entry fields should be first name, last name, sex, birth date, SSN, contact number, address, email, EID, hire date, location, and pay rate.
FR2	High	First and last name fields should be limited to character-based input.
FR3	High	Sex input should be limited to M or F.
FR4	High	Birth date and hire date should be formatted DATE mm/dd/yyyy.
FR5	High	SSN should be formatted as 9-digit INT.
FR6	High	Contact number should be formatted as 10-digit INT.
FR7	High	Address and email should be formatted as VARCHAR.
FR8	High	EID should be formatted as 5-digit INT.
FR9	High	Location should be selected from a list of predefined company locations.

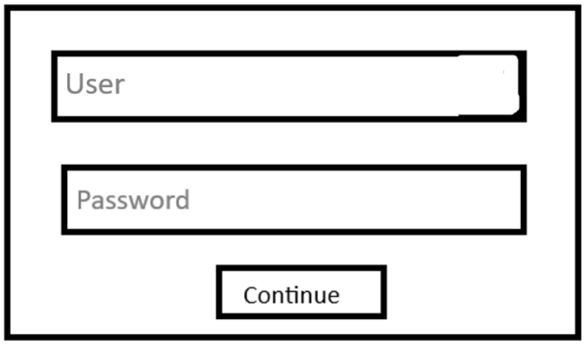
FR10	High	Pay rate should be selected from a list of predefined pay rates.
FR11	Med	The system should have 2 levels of authentication. Read and modify.
FR12	High	Search functionality that enables HR administrators to quickly locate and retrieve employee records based on any of the specified fields for efficient data access.

Non-Functional System Requirements:

Req Number	Priority Weight	Description
NR1	High	Scalability: The system should be scalable to accommodate an increasing number of employee records and adapt to the growing needs of the organization
NR2	Med	Interoperability: Ensure compatibility with existing HR systems, allowing seamless data exchange and integration with other enterprise applications.
NR3	High	User Interface Design: Design an intuitive and user-friendly interface for both HR administrators and employees, promoting ease of use and reducing the learning curve.
NR4	High	Data Integrity: Maintain high levels of data integrity, ensuring that employee information is accurate, consistent, and reliable.

User Interface Requirements:

Login (Med requirement): A simple username and password prompt referencing an internal database of users with access and their rights. In the real world, this would most likely be domain-based authentication or a much larger database of access roles, but that development is way outside of the scope of this simple example project.



Page layout (High requirement): I am looking to make a simple field of input style design that adds users to a table that is visible below the entry points. A search field will simply either skip to that table entry or reduce the size of the table to just the corresponding entries.

Search

EID	First Name	Last Name	Sex	Birth Date	SSN	Phone #	Address	Email	Hire date	Location|▼|	Pay Rate|▼|
Input	Input	Input	Input	Input	Input	Input	Input	Input	Input	Input	Input

10254	Tom	Laff	M	1986/ 07/18	999999999	9371541234	123 Main St, Town, OH	blah@yahoo.com	2019/05/ 12	Lewisburg	E5
19482	Dave	Jones	M	1999/ 05/20	888888888	7653450194	567 Water St, Richmond , IN	blah@gmail.com	2020/04/ 15	Eaton	E9

Field Examples (Med Requirement): Each input field will have a grey text that will give an example of the expected input format.

11111	Name	Name	M Or F	yyyy/ mm/dd	111111111	1111111111	123 Main Dayton, OH	blah@yahoo.com	yyyy/ mm/dd

Messages (Med Requirement): Missing information or incorrect input information would show those fields as red after clicking submit if they do not match the correct format and a message to the user that says to check data fields. Or there will be a green success message for additions, edit, and deletions.

Error: Please verify format of attempted data entry. User was not added.

User added successfully.

User updated successfully.

User deleted successfully.

Functional Requirements Specification

Stakeholders:

- HR
- Payroll
- Managers
- Employees
- IT Administrators

Actors and Goals:

- Primary Actors
  - HR – The primary users of an employee management system would be the HR department. This would be a central database that can be used to store all the corresponding information for an employee. They will be responsible for add, deletions, and edits.
  - Payroll – The central database could be used by the payroll coordinator to track who works where, for how many hours, and in what pay band.
  - Managers – Could use this system to track who works in their departments and see their HR information.

- Secondary Actors
  - Employees – It is their data being tracked so ultimately; they have a hand in the system but it would be indirectly through HR.
  - IT– Could also add, delete, and edit if needed and troubleshoot issues with the system.
  - System – is responsible for holding the information entered by the HR department.

Use Cases:

HR (10)

-Add employee: To add a new employee to the system (2)

-Delete employee: Remove employee from the system (2)

-Edit employee: Change the information of an employee already in the database (2)

-Search for employee: Using attributes to search for an employee (2)

-Troubleshoot issues: Input or standardization issues can be looked at by HR (2)

Manager/payroll (2)

-Search for employee: Using one of the main attributes to search for an employee (2)

System (8)

-Authenticate login: Verify the user has access rights to information (2)

-Display search: If a search is done the system needs to return the correct information (2)

-Display login: Default landing page for username and password (2)

-Display information: Default page once logged into the system (2)

IT (14)

-Add a new location: If a new site were to open IT would be needed to add a new location to the database (2)

-Modify pay bands: If the scale were to change IT would be needed to modify the list of pay bands available (2)

-New user creation: If a new user needs access to the system IT would need to create this user (2)

-Troubleshooting: If any database issue were to occur IT would need to do this (2)

-Add employee: To add a new employee to the system if HR cannot or as part of troubleshooting (2)

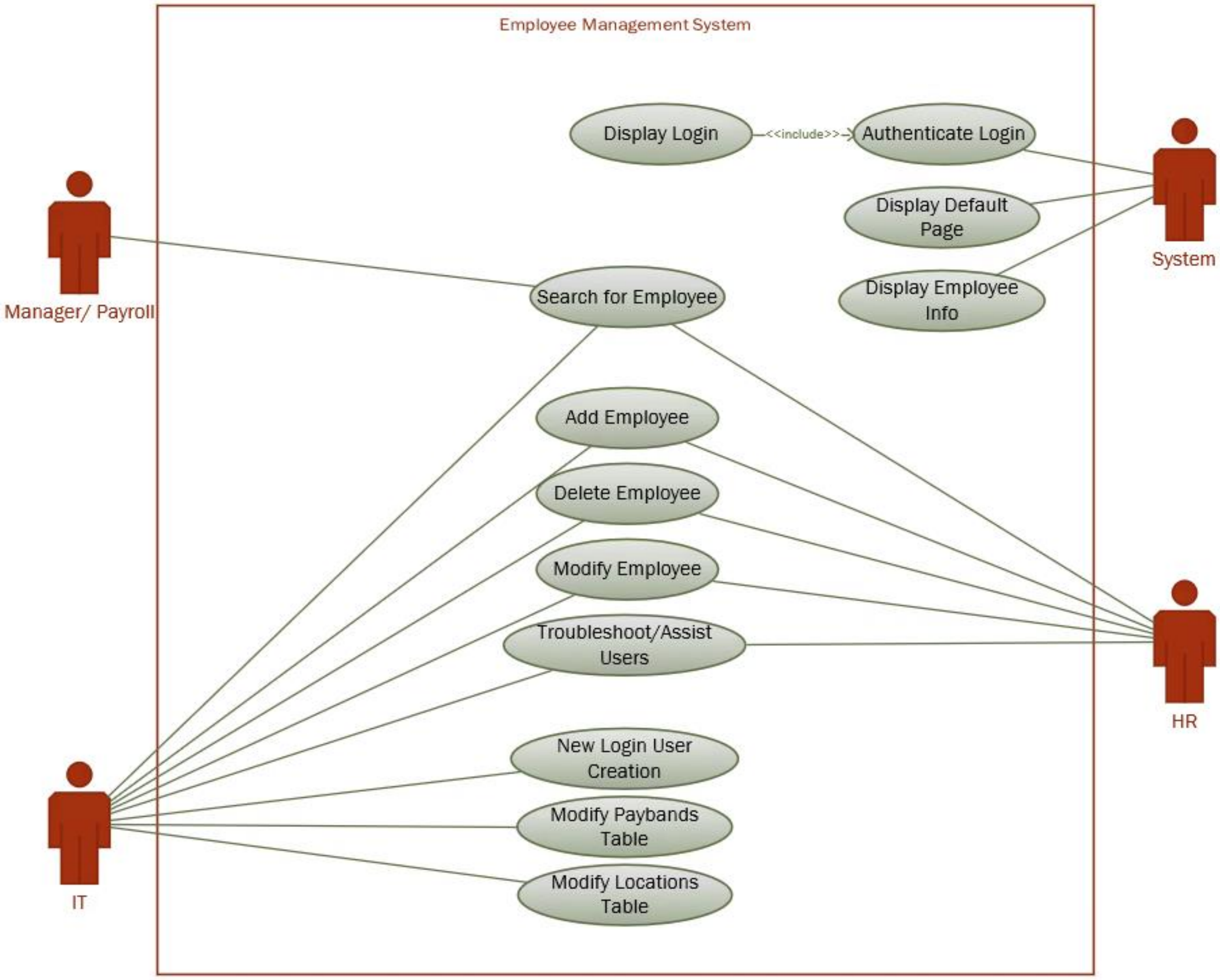
-Delete employee: Remove employee from the system if HR cannot or as part of troubleshooting (2)

-Edit employee: Change the information of an employee already in the database if HR cannot or as part of troubleshooting (2)

-Search for employee: Using attributes to search for an employee if HR cannot or as part of troubleshooting (2)

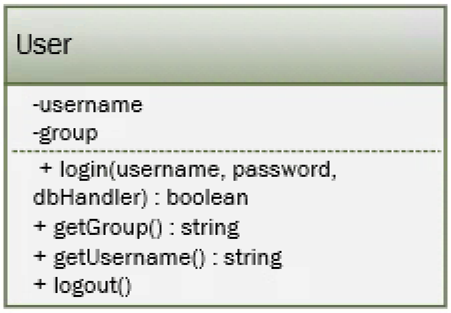
Use Case Diagram:

Display login information is included in the authentication.



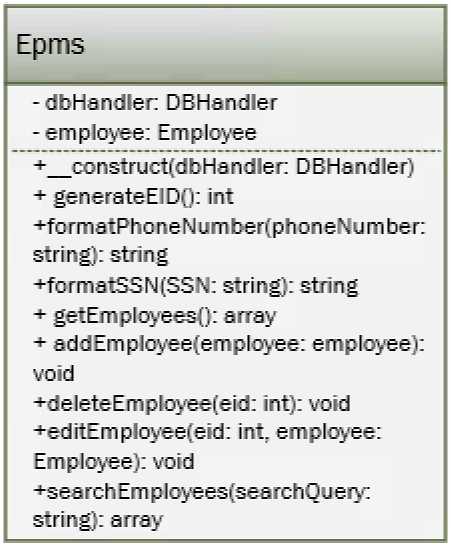
<u>Class Diagram:</u>

User - This class handles the authentication for login to the system



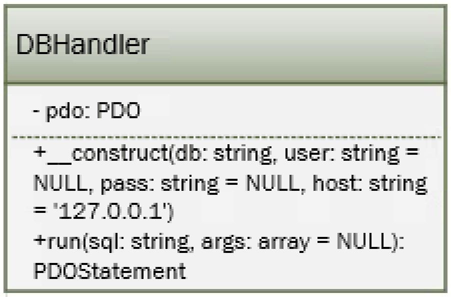
```
User
-username
-group
-----------------------------
 + login(username, password,
dbHandler) : boolean
+ getGroup() : string
+ getUsername() : string
+ logout()
```

Epms - This is the primary class in the system. It contains the functions that will add, remove, edit, and search for employees in the database.



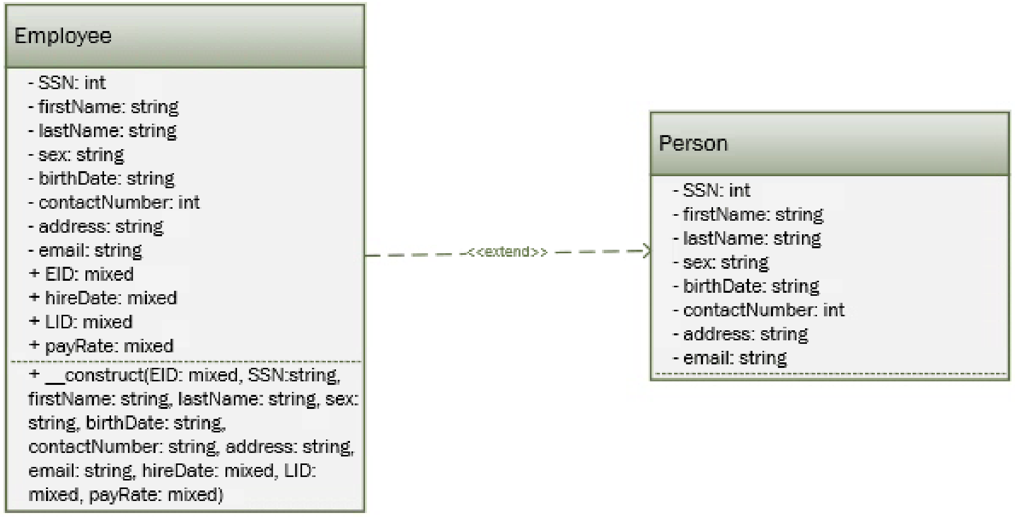
```
Epms
- dbHandler: DBHandler
- employee: Employee
-----------------------------
+__construct(dbHandler: DBHandler)
+ generateEID(): int
+formatPhoneNumber(phoneNumber:
string): string
+formatSSN(SSN: string): string
+ getEmployees(): array
+ addEmployee(employee: employee):
void
+deleteEmployee(eid: int): void
+editEmployee(eid: int, employee:
Employee): void
+searchEmployees(searchQuery:
string): array
```

DBHandler - This class is responsible for taking user input from the Epms class and transferring it to the database as well as performing queries on the database.

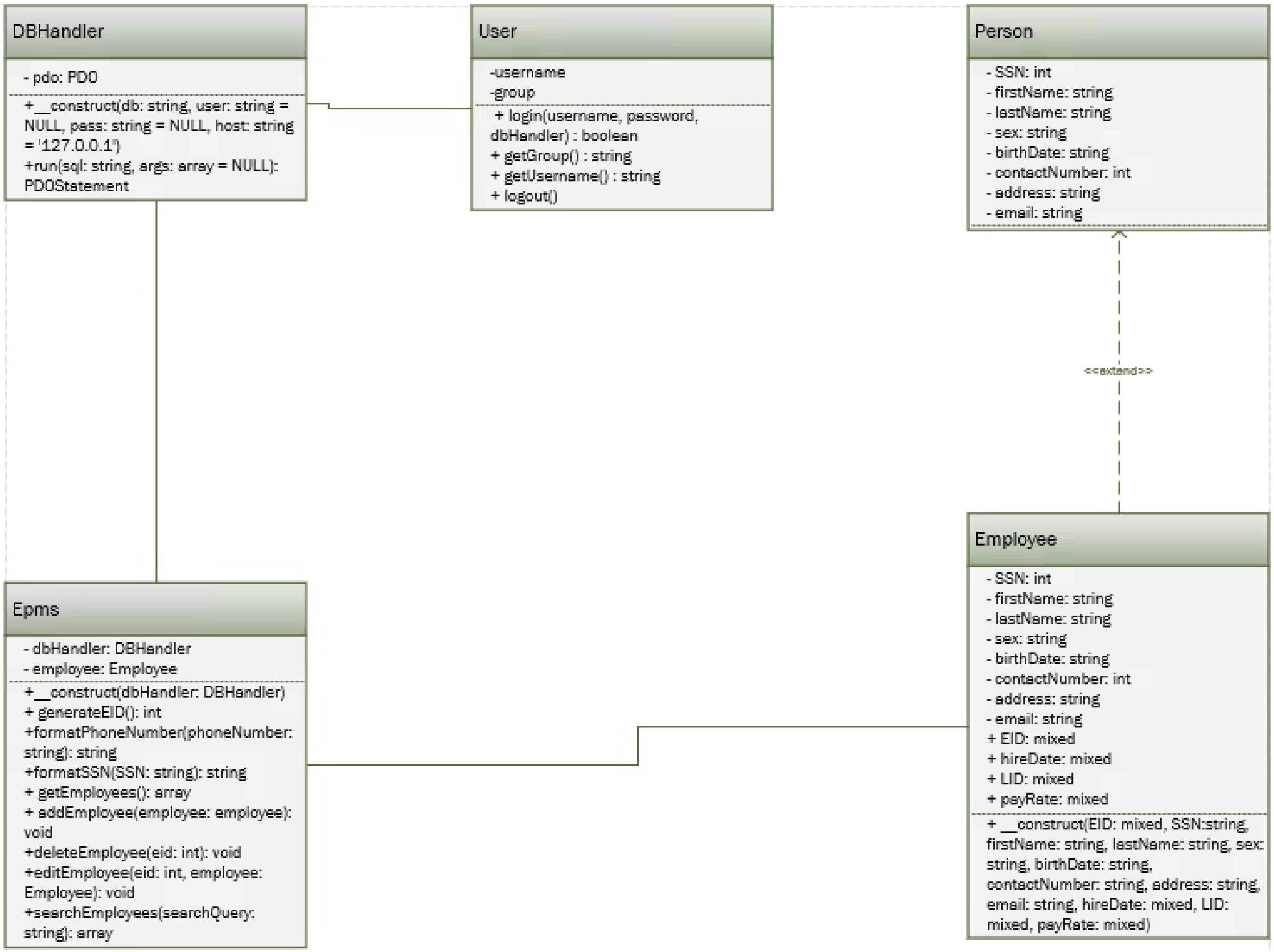


```
DBHandler
- pdo: PDO
-----------------------------
+__construct(db: string, user: string =
NULL, pass: string = NULL, host: string
= '127.0.0.1')
+run(sql: string, args: array = NULL):
PDOStatement
```

Employee/Person - The Employee class is an extension of the Person class which is a general template for personal information. The Employee class is a more specific template that includes personal details plus additional information related to someone's job in the company.



Final


DBHandler
- pdo: PDO
+__construct(db: string, user: string = NULL, pass: string = NULL, host: string = '127.0.0.1')
+run(sql: string, args: array = NULL): PDOStatement
User
-username
-group
+ login(username, password, dbHandler) : boolean
+ getGroup() : string
+ getUsername() : string
+ logout()
Person
- SSN: int
- firstName: string
- lastName: string
- sex: string
- birthDate: string
- contactNumber: int
- address: string
- email: string
<<extend>>
Epms
- dbHandler: DBHandler
- employee: Employee
+__construct(dbHandler: DBHandler)
+ generateEID(): int
+formatPhoneNumber(phoneNumber: string): string
+formatSSN(SSN: string): string
+ getEmployees(): array
+ addEmployee(employee: employee): void
+deleteEmployee(eid: int): void
+editEmployee(eid: int, employee: Employee): void
+searchEmployees(searchQuery: string): array
Employee
- SSN: int
- firstName: string
- lastName: string
- sex: string
- birthDate: string
- contactNumber: int
- address: string
- email: string
+ EID: mixed
+ hireDate: mixed
+ LID: mixed
+ payRate: mixed
+ __construct(EID: mixed, SSN:string, firstName: string, lastName: string, sex: string, birthDate: string, contactNumber: string, address: string, email: string, hireDate: mixed, LID: mixed, payRate: mixed)

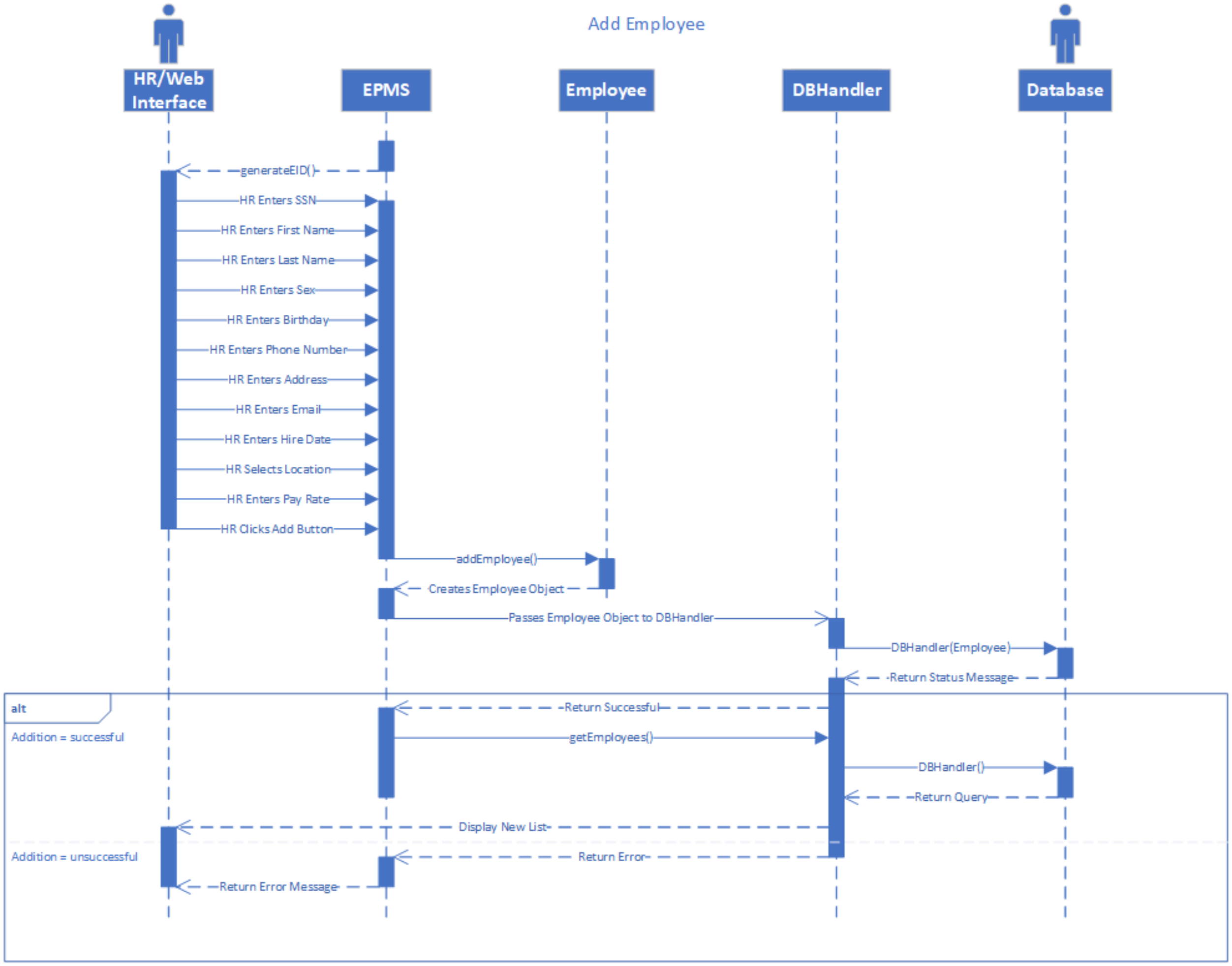
System Sequence Diagram and Activity Diagram

System Sequence Diagrams

Add Employee

Actor: HR
Objects: Interface, EPMS, Employee, DBHandler, Database

- HR enters employee information
- Employee object created
- Employee object passed to database using DBHandler
- If the employee is added the list refreshes and shows the employee
- If addition fails reason code is given



Delete Employee

Actor: HR

Objects: Interface, EPMS, Employee, DBHandler, Database

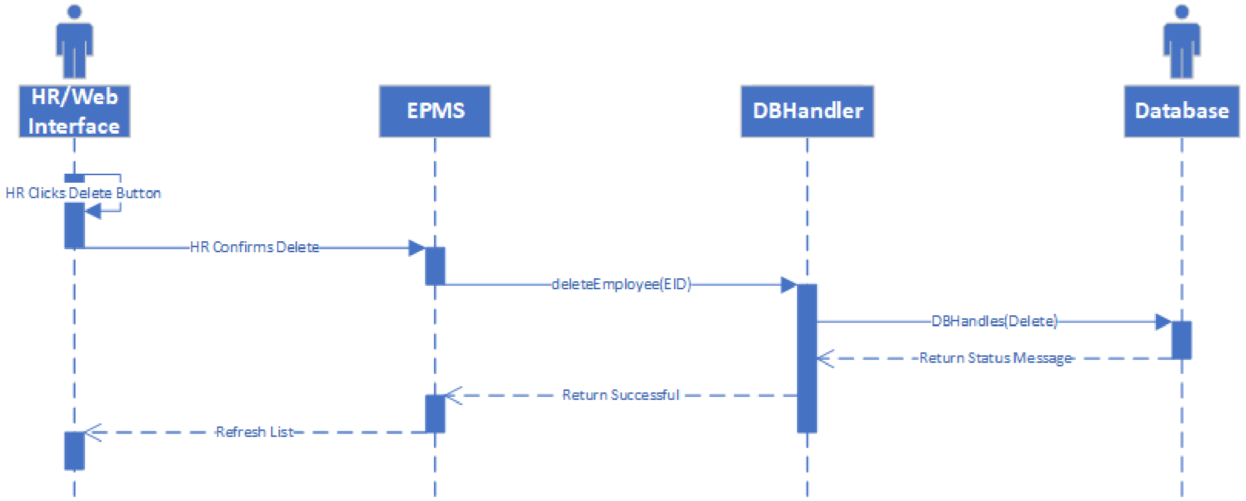
-HR clicks the delete button

-Page prompts confirmation

-EID sent for deletion from the database by DBHandler

-List refresh with employee missing from it

Delete Employee



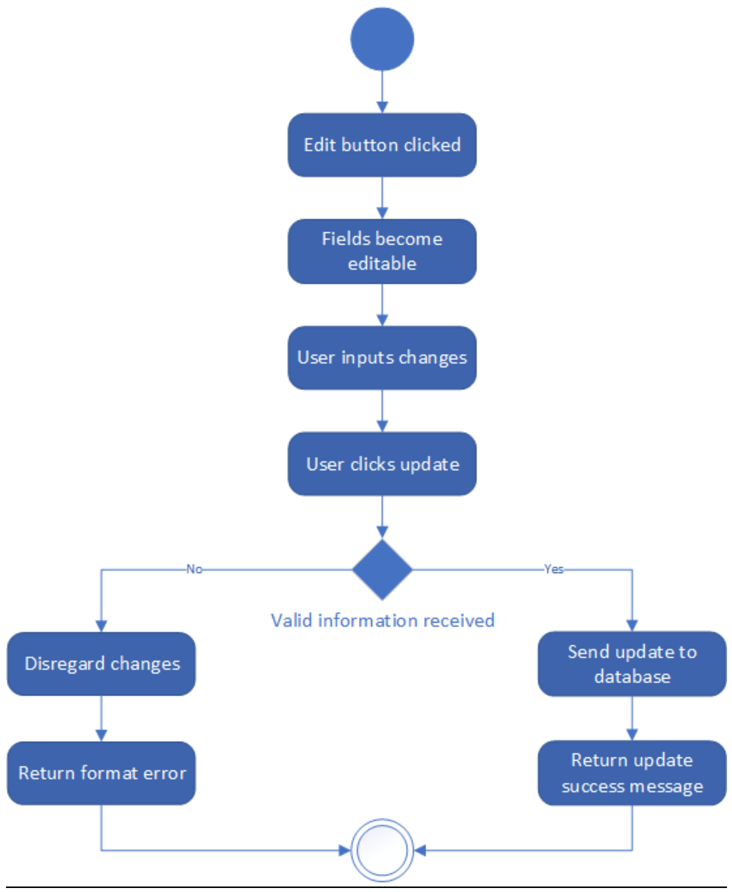
Activity Diagrams

Edit Employee

States:
-Initial State: Original record exists in the database
-Final State: Existing entry is updated with new information

Actions:
HR clicks the edit button on an employee entry. The record is loaded in a modifiable state. Fields are updated by HR. The record is saved to the database in an updated fashion, or a format error is returned.

Edit Employee



Search Employee

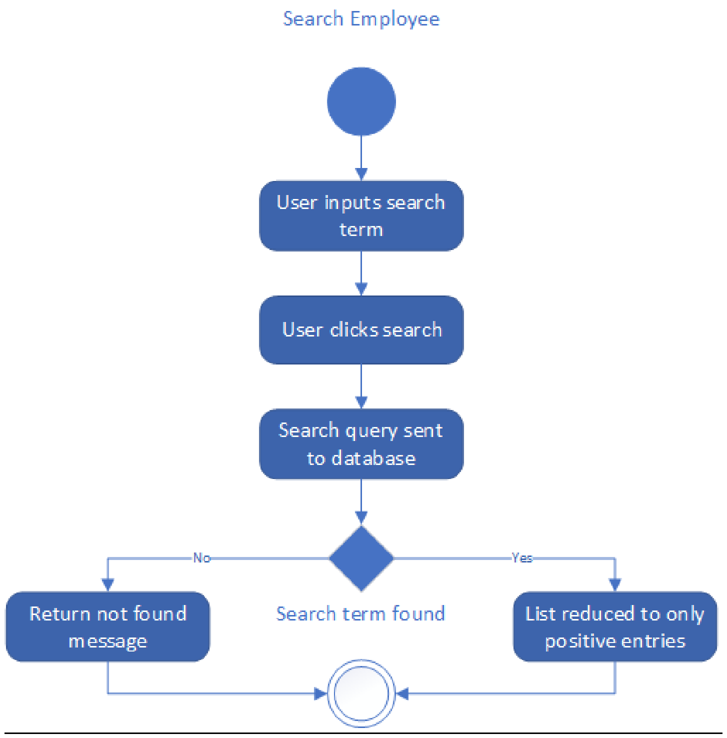
States:

-Initial State: A full list of employee records is shown

-Final State: Record with matching information is shown

Actions:

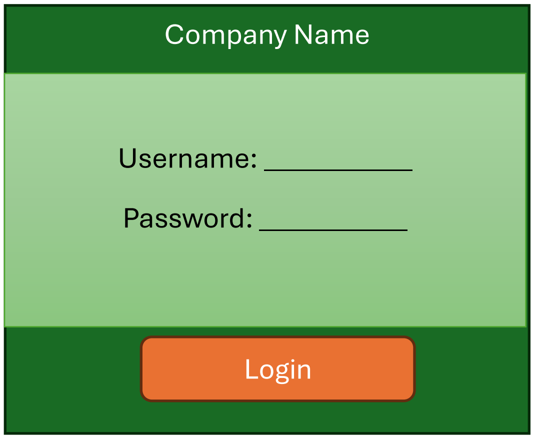
HR enters key information into the search bar. HR clicks the search button. The list is reduced to show just the entries matching the key results. If no results match the list disappears showing a message about no matching results.



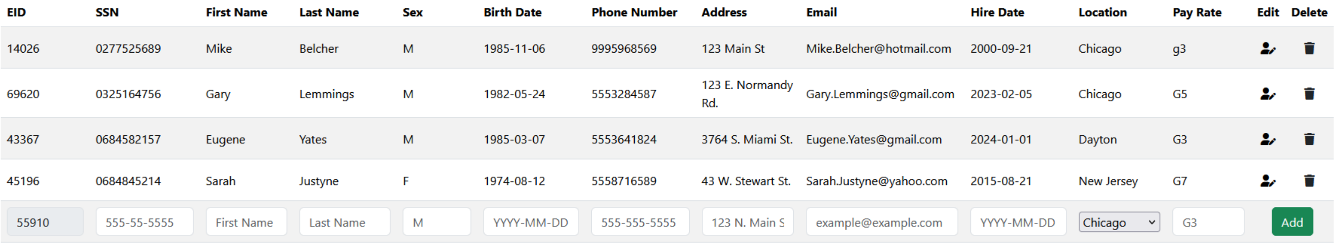
User Interface Specifications

Preliminary Design

Use Case: Login



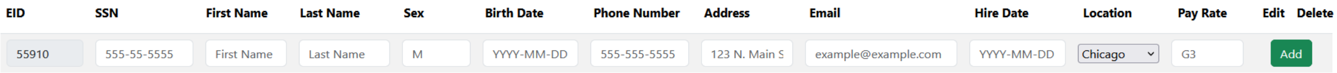
Initial login page. The user enters login information, clicks "Login", and is redirected to the main EPMS page. →



EID	SSN	First Name	Last Name	Sex	Birth Date	Phone Number	Address	Email	Hire Date	Location	Pay Rate	Edit	Delete
14026	0277525689	Mike	Belcher	M	1985-11-06	9995968569	123 Main St	Mike.Belcher@hotmail.com	2000-09-21	Chicago	g3	👤	🗑
69620	0325164756	Gary	Lemmings	M	1982-05-24	5553284587	123 E. Normandy Rd.	Gary.Lemmings@gmail.com	2023-02-05	Chicago	G5	👤	🗑
43367	0684582157	Eugene	Yates	M	1985-03-07	5553641824	3764 S. Miami St.	Eugene.Yates@gmail.com	2024-01-01	Dayton	G3	👤	🗑
45196	0684845214	Sarah	Justyne	F	1974-08-12	5558716589	43 W. Stewart St.	Sarah.Justyne@yahoo.com	2015-08-21	New Jersey	G7	👤	🗑
55910	555-55-5555	First Name	Last Name	M	YYYY-MM-DD	555-555-5555	123 N. Main S	example@example.com	YYYY-MM-DD	Chicago	G3	Add	

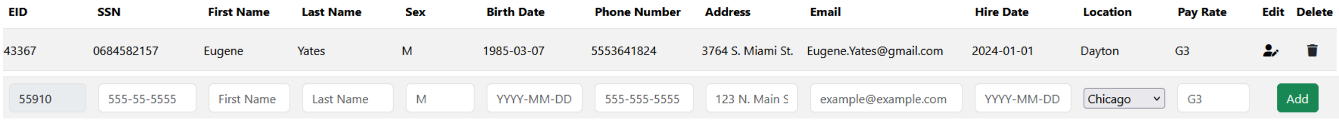
If that user has edit rights this would be a sample screen. If not, it will look similar but the icons for edit and delete will be greyed out.

Use Case: Add Employee



EID	SSN	First Name	Last Name	Sex	Birth Date	Phone Number	Address	Email	Hire Date	Location	Pay Rate	Edit	Delete
55910	555-55-5555	First Name	Last Name	M	YYYY-MM-DD	555-555-5555	123 N. Main S	example@example.com	YYYY-MM-DD	Chicago	G3	Add	

The user is presented with the above prompt. Entering format matching data and clicking add returns 2 options. →



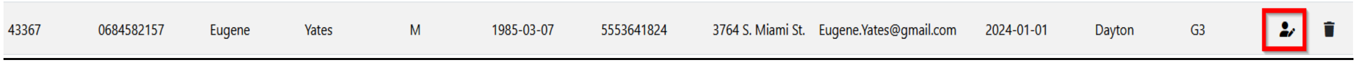
EID	SSN	First Name	Last Name	Sex	Birth Date	Phone Number	Address	Email	Hire Date	Location	Pay Rate	Edit	Delete
43367	0684582157	Eugene	Yates	M	1985-03-07	5553641824	3764 S. Miami St.	Eugene.Yates@gmail.com	2024-01-01	Dayton	G3	👤	🗑
55910	555-55-5555	First Name	Last Name	M	YYYY-MM-DD	555-555-5555	123 N. Main S	example@example.com	YYYY-MM-DD	Chicago	G3	Add	

Successful additions will populate the list with the information supplied.

Error: Please verify the format of the attempted data entry. The employee was not added.

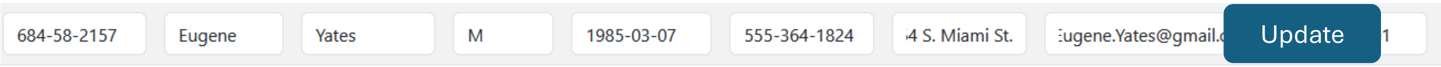
Failed additions will return an error message stating what was missing (format mismatch, empty prompt box, etc.)

Use Case: Edit Employee



43367	0684582157	Eugene	Yates	M	1985-03-07	5553641824	3764 S. Miami St.	Eugene.Yates@gmail.com	2024-01-01	Dayton	G3

Clicking the edit button turns the fields of an entry into populated text boxes →

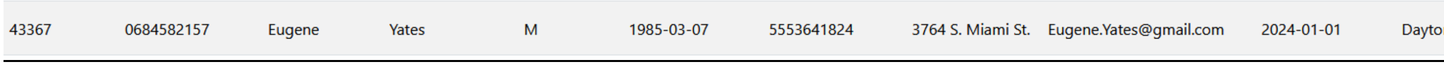


684-58-2157	Eugene	Yates	M	1985-03-07	555-364-1824	4 S. Miami St.	:ugene.Yates@gmail.c	1

Error: Please verify the format of the attempted data entry.

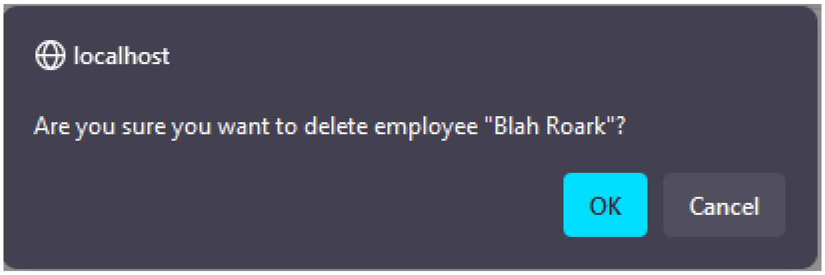
Clicking update updates the employee in the database or rejects the update if the format does not match.

Use Case: Delete Employee



43367	0684582157	Eugene	Yates	M	1985-03-07	5553641824	3764 S. Miami St.	Eugene.Yates@gmail.com	2024-01-01	Dayto

Clicking the delete button prompts the user for confirmation →

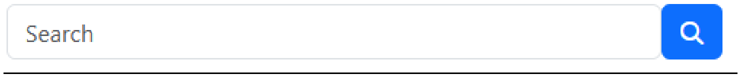


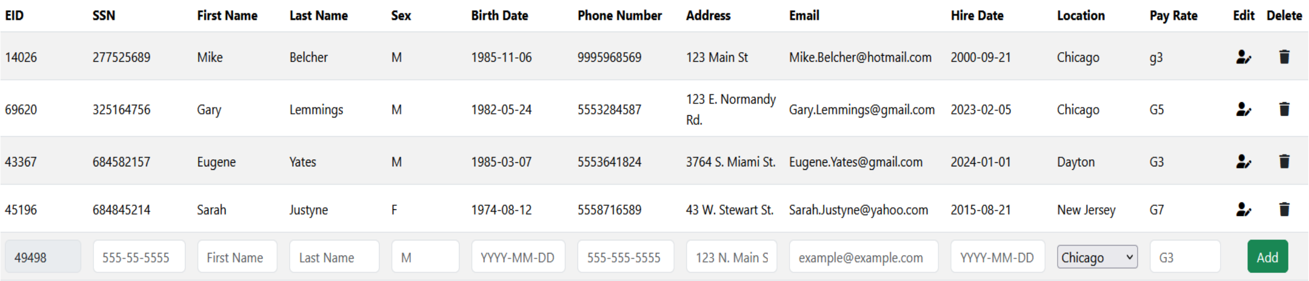
localhost

Are you sure you want to delete employee "Blah Roark"?

Clicking "Yes" removes the employee, and clicking No cancels the action.

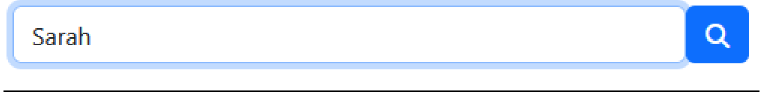
Use Case: Search for Employee

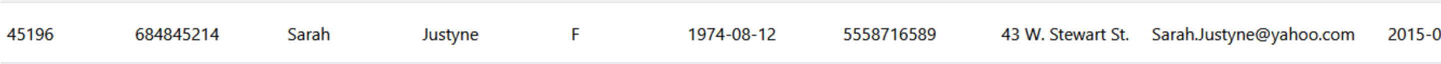




EID	SSN	First Name	Last Name	Sex	Birth Date	Phone Number	Address	Email	Hire Date	Location	Pay Rate	Edit	Delete
14026	277525689	Mike	Belcher	M	1985-11-06	9995968569	123 Main St	Mike.Belcher@hotmail.com	2000-09-21	Chicago	g3	✎	🗑
69620	325164756	Gary	Lemmings	M	1982-05-24	5553284587	123 E. Normandy Rd.	Gary.Lemmings@gmail.com	2023-02-05	Chicago	G5	✎	🗑
43367	684582157	Eugene	Yates	M	1985-03-07	5553641824	3764 S. Miami St.	Eugene.Yates@gmail.com	2024-01-01	Dayton	G3	✎	🗑
45196	684845214	Sarah	Justyne	F	1974-08-12	5558716589	43 W. Stewart St.	Sarah.Justyne@yahoo.com	2015-08-21	New Jersey	G7	✎	🗑
49498	555-55-5555	First Name	Last Name	M	YYYY-MM-DD	555-555-5555	123 N. Main S	example@example.com	YYYY-MM-DD	Chicago	G3	Add	

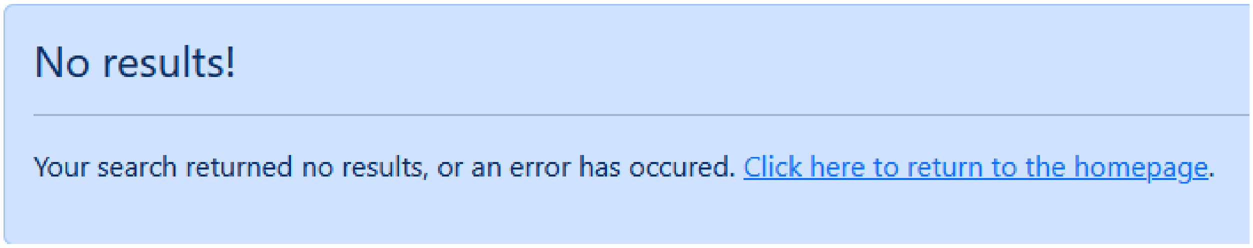
Typing a search term in the text box and clicking search reduces the list shown to just entries that match the search or list no matching entries →





45196	684845214	Sarah	Justyne	F	1974-08-12	5558716589	43 W. Stewart St.	Sarah.Justyne@yahoo.com	2015-0

-or-



No results!

Your search returned no results, or an error has occured. Click here to return to the homepage.

Clearing search returns the full list.

User Effort Estimation

Use Case	Clicks	Keystrokes
Use Case: Login	1	<50
Use Case: Add	1	<50
Use Case: Edit	2	<50
Use Case: Delete	2	0
Use Case: Search	1	<10