Payroll System
CSI 3140 Term Project Phase 3
July 23, 2023

Jeremy Fang - 300014214
Jaleelah Ammar - 300174391
Shannon Noah - 300163898

Source Code

Link to our public github repository with our source code:
https://github.com/ShannonNoah/CSI-3140-Term-Project

Status Report

- We estimated that building the front-end application using the React Framework and making all of the pages with styling would take about 1 week, which was an accurate estimate.
- We estimated that converting the E-R diagrams into the database schema would take 1-2 weeks, which was accurate as it took about 2 weeks.
- We estimated that developing the backend, including creating the backend server and creating routes for the API would take about 1 week, and it ended up taking a bit longer, roughly 2 weeks, to account for debugging the small errors that came up.
- Overall, we ended up having to make some small adjustments to the timeframe of our original plan in some areas, though we were still always on track by communicating regularly with one another. In the end, we were able to successfully complete all components of the project on time.

Finalized Schedule

- 6 weeks

Week 1 (June 12 - 18)	Made E-R diagram + Class Diagram
Week 2 (June 19 - 25)	Made Schema + Component Diagram
Week 3 (June 26 - July 2)	Experimented with using React and Express to build familiarity through self-learning.
Week 4 (July 3 - 9)	Built a skeleton of the backend and frontend.
Week 5 (July 10- 16)	Finalized the frontend application and the backend as well as made the connection to the database component.
Week 6 (July 17 - 24)	Tested many cases and debugged errors that came up such as the search functionality not displaying the record table.

Summary

Our team was able to successfully complete this project due to our efforts to communicate with one another regularly. In phase one, we selected the scrum method for the development of this project, which includes weekly sprints. We made sure to update one another consistently on what were were working on, ask eachother for guidance if we ran into problems, and collaborate to come up with solutions through many group meetings. We also made sure to divide up the different components of the project to specific team members and communicate the timeline of when different sections need to be completed by. While we sometimes had to make modifications to our original schedule, we worked with one another to stay on track and be able to complete all deliverables of the project on time.

Project Scope

Initial Project Scope Statement:

The project will be a web application including authorization and different features for the two main users of the application: human resource employees and standard employees. For human resources, the system will allow users to easily create, modify and close employee information records, attendance records, and payroll transaction records. As for other employees, the system will allow them to access similar information for their personal employee profiles.

Revised Project Scope

As was mentioned in our first deliverable, there was a fairly strict time constraint for the delivery of our project, as well as limited resources (man-power). In order to accommodate this, we decided that we would focus on the features that would bring the most value to our stakeholders, namely: record creation, modification, deletion and searching. Furthermore, we decided that an emphasis on design and process would lead to a stronger foundation to the project, thus making development more efficient in the future. This meant we spent a lot of time designing diagrams, prototyping, and making sure to follow design standards, which resulted in us having to cut out some of the previously proposed requirements. Included in the SRS section of the report, cut requirements have been struck through.

Revised Project Estimates:

In our initial project estimate, we estimated that we would have enough time/resources to fully complete the requirements proposed. However, we did not accurately estimate the time required to design and prototype for the project, resulting in an overall increase in project cost.

Time, cost, performance trade-off assessment

As mentioned above, we focused on building a foundation for the future of the system, which results in a greater upfront cost, as well as time. However, the time and cost spent will result in better performance in the future due to reduced redundancy, separation of concerns, and modularity of the project as a whole.

Work Breakdown Structure

Level	ID	Description
1	**1**	**Planning**
2	**1.1**	**Identify Problem**
3	1.1.1	Create user stories to better understand pain points
3	1.1.2	Identify user goals
2	**1.2**	**Devise Solution**
3	1.2.1	Form requirements from user goals
3	1.2.2	Design compliant Software Requirements Specification
2	**1.3**	**Decide on Project Scope**
3	1.3.1	Estimate time requirements from SRS
3	1.3.2	Label and sort requirements based on priority
3	1.3.3	Filter out requirements based on time constraint estimates
1	**2**	**Design**
2	**2.1**	**Select Technology Stack**
3	2.1.1	Research common technology stacks in domain
3	2.1.2	Read documentation to familiarize technologies
2	**2.2**	**Perform Domain Analysis**
3	2.2.1	Create E-R diagrams for backend
3	2.2.2	Convert E-R diagrams into database schema
3	2.2.2	Create Component diagrams for web application
3	2.2.3	Create Activity diagrams for user flow control

1	**3**	**Developing**
2	**3.1**	**Develop frontend**
3	3.1.1	Create pages for frontend
3	3.1.2	Create styling for pages
3	3.1.3	Create router for routing between pages
2	**3.2**	**Develop backend**
3	3.2.1	Create backend server
4	3.2.1.1	Create routes for API
4	3.2.1.2	Create controllers for API
3	3.2.2	Create Database
4	3.2.2.1	Create models for entities
3	3.2.3	Connect backend server to database
4	3.2.3.1	Connect server to database
4	3.2.3.2	Connect individual API endpoints to database and perform intended function
2	**3.3**	**Integrate backend and frontend**
3	3.3.1	Create API calls from web application to backend server
1	**4**	**Testing**
2	4.1	Test backend API
2	4.2	Test Validation and Verification

Retrospective

Problems Encountered:

Problem 1: Relational vs Document Driven Databases

Initially we designed the E-R diagrams and database schema with a relational database in mind. However, upon further research we found that mongoDB is a document driven database, and thus it does not support many features that a relational database like SQL would. One example of this was how mongoDB does not support referential keys or primary keys. In order to resolve this problem, we needed to add additional checks in the API controller functions to make sure that the referenced ids were valid.

Problem 2: Bad User Behaviour

We wanted to minimize the need for specialized pages and tedious specifications on the front end. It became clear when setting up API calls that this approach left lots of room for invalid user input to cause problems in the database. Users could assume that they did not need to enter employees dates of birth. They could also assume that it was not important to specify what field they wanted to use to search for employees or transactions. We solved this issue by limiting the options of users and blocking them from submitting forms with incomplete information. We used simple HTML solutions like dropdown menus and required fields to ensure validation.

Problem 3: Integrating the Front and Back Ends

We encountered issues with getting information back and forth between the website to the database. We used a REST API system to ensure standardization, but integrating this came with challenges. The biggest issue was getting around the default CORS policies preventing the express server and the mongo database from communicating. Even when we made allowances for the specific ports on which the front and back ends were listening to communicate, POST requests were blocked by the policy. We discovered that our server was sending HTTP requests through port 80 (not the port it was running on) because it was emulating the way it would behave in production. We solved the issue by allowing requests from any port.

Management Issues:

Main Issue: Github Collaboration

Since each of the team members worked independently on a different component of the project (Shannon on the front end, Jeremy on the back end, and Jaleelah on the integration), collaborating via Github proved to be a challenge. Conflicts between changes made it difficult to test the system while it was still in development. We solved this issue by using several branches in our repository and communicating which changes were important enough to be manually ported over to other members' builds.

Conclusion

In conclusion, our team was able to successfully collaborate with one another to create a web application that handles payroll processing by allowing users to modify and close employee information records, attendance records, and payroll transaction records. We were able to successfully complete all components of the project, including building the front end application, developing the backend server, and connecting it to our database. We also scheduled time to test the backend API and the validation and verification features, as well as debug small errors we encountered.

Additional Included Figures

ER Diagram:

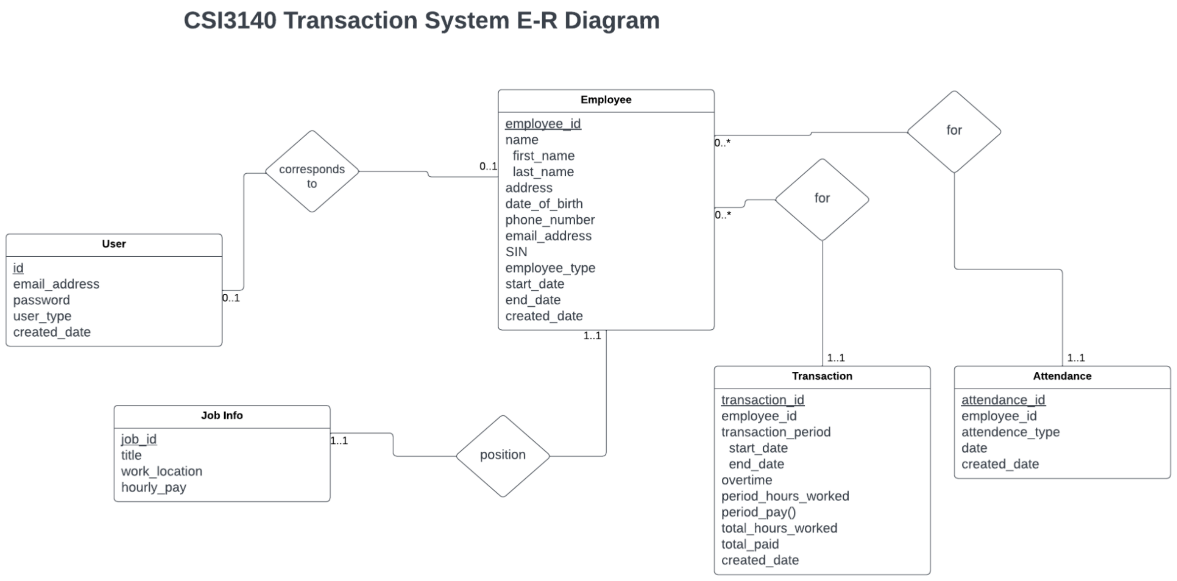


Figure 1. E-R diagram for backend entities

Class Diagram:

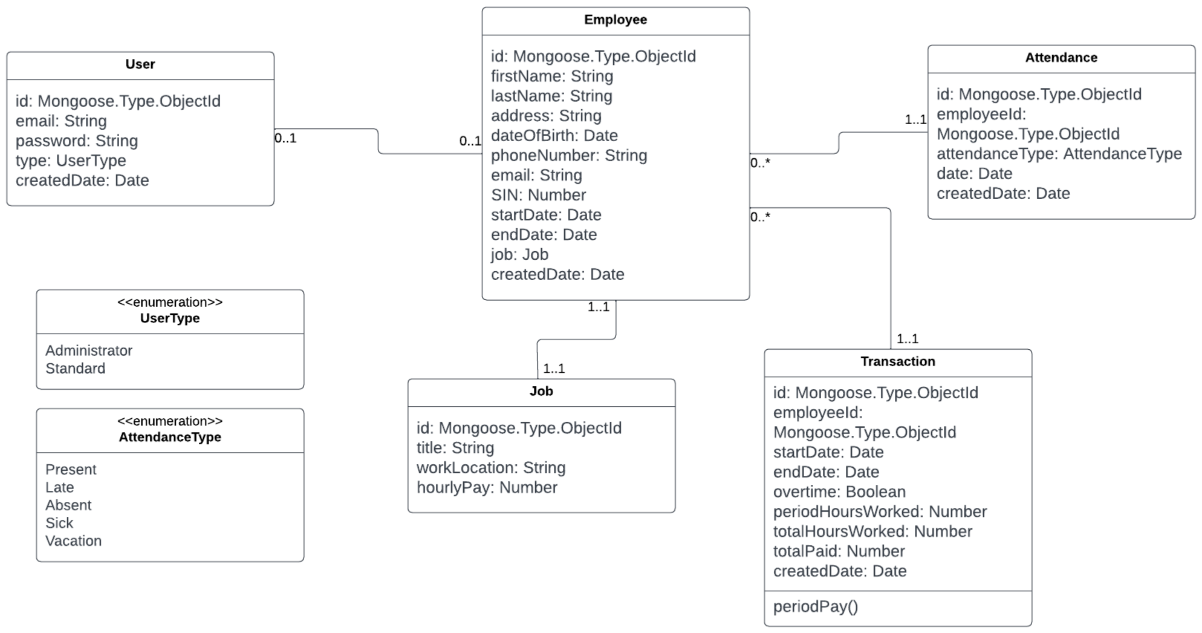
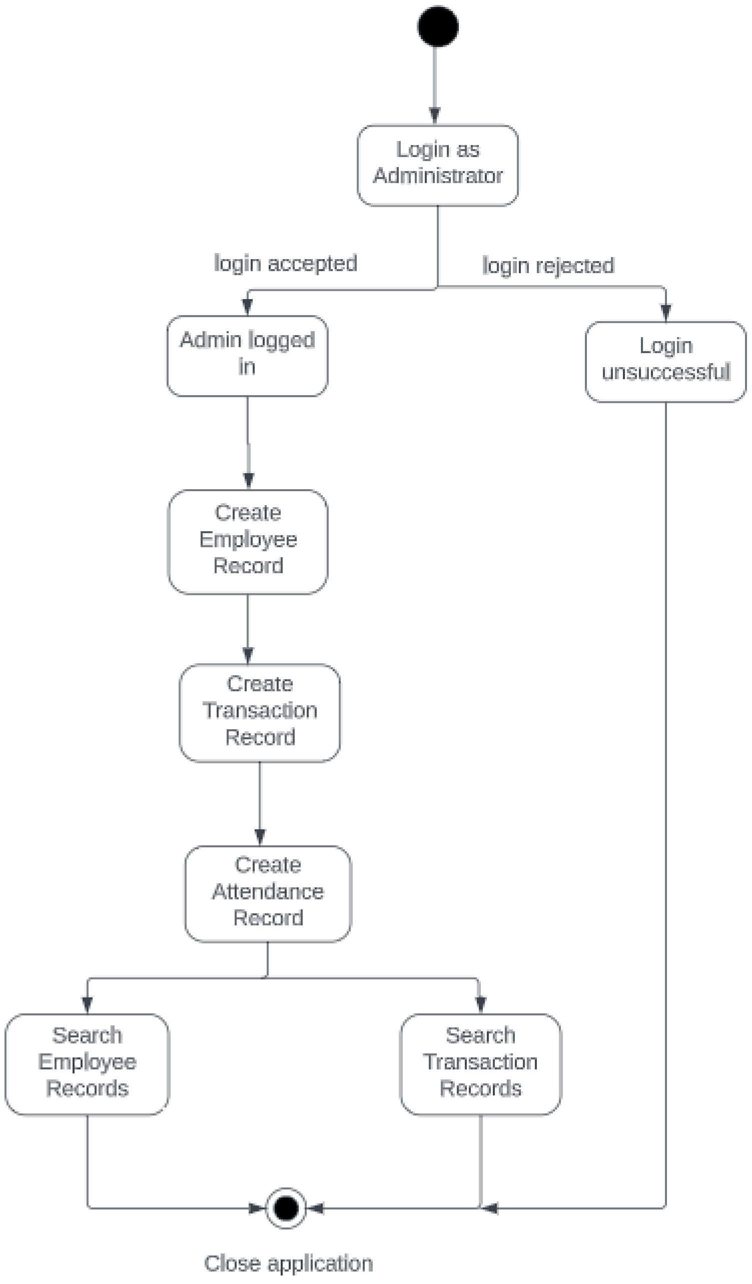
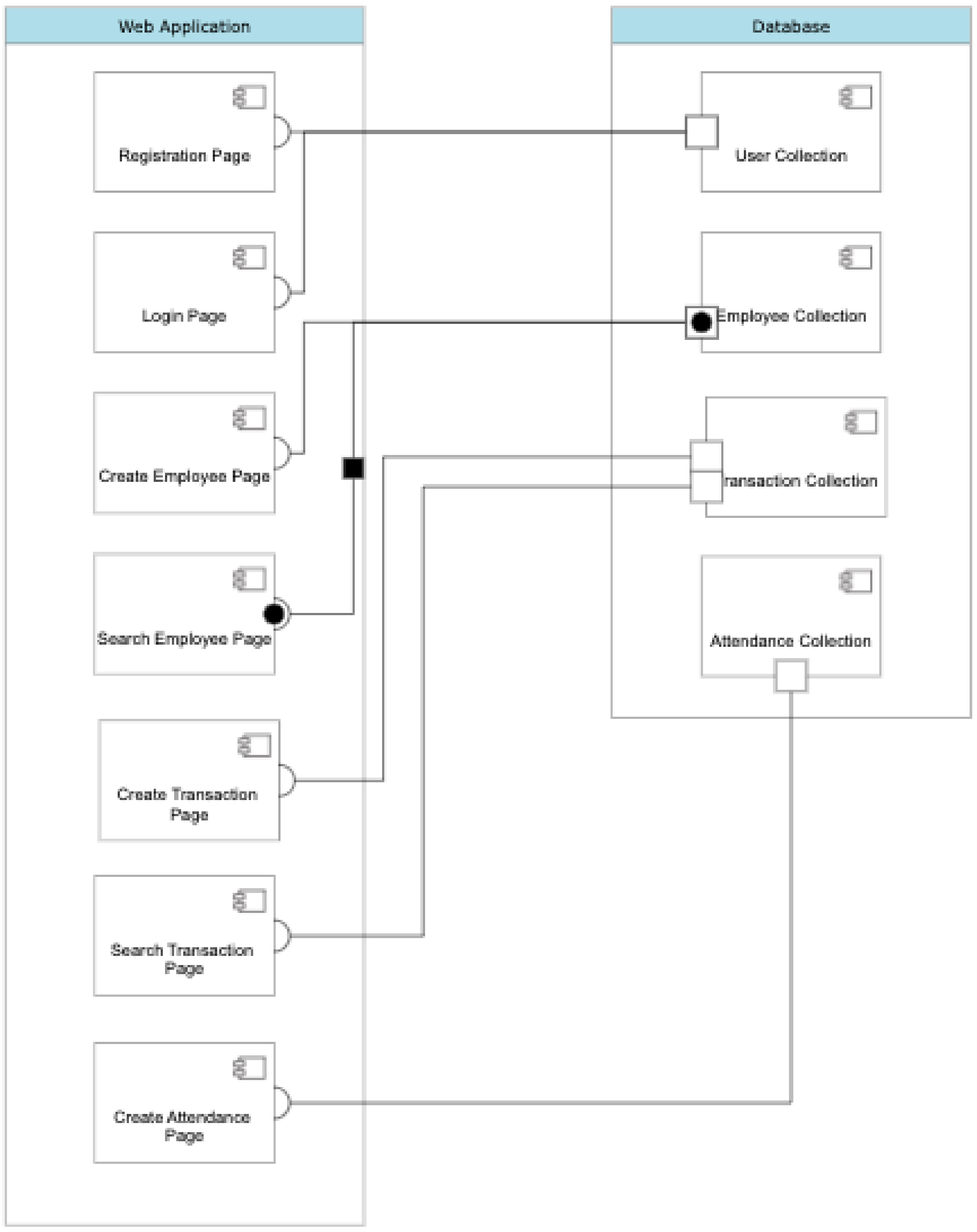


Figure 2. UML classes for backend entities

Activity Diagram:



Component Diagram:



Revised Software Requirements:

Goals:

ID	Stakeholder	Goal
GOAL-01	Human Resources	To more efficiently enter transaction information regarding employee payments
GOAL-02	Human Resources	Create, modify and close employee information records
GOAL-03	Human Resources	Easily keep track of employee attendance
GOAL-04	Human Resources	Search the system for information regarding specific employees
GOAL-05	Employees	Easily access personal attendance and payment information

Functional Requirements:

ID	Requirement	Priority	Goal	Status
F-REQ-01	The system shall allow HR to create employee records	High	GOAL-02	Complete
F-REQ-02	The system shall allow HR to modify employee information in their records	High	GOAL-02	Complete
F-REQ-03	The system shall allow HR to close employee records	High	GOAL-02	Complete
F-REQ-04	The system shall require personal information when creating an employee record including the following: full name, address, phone number, SIN, birth date	High	GOAL-02	Complete
F-REQ-05	The system shall require job information when creating an employee record including the following: job title, work location, start date, and salary	High	GOAL-02	Complete
F-REQ-06	The system shall optionally allow HR to enter the following when creating an employee record: email address, and job end date	High	GOAL-02	Complete
F-REQ-07	The system shall require form validation and verification when creating an employee record	Medium	GOAL-02	Complete
F-REQ-08	The system shall allow HR to create employee	High	GOAL-01	Complete

	payroll transactions			
F-REQ-09	The system shall allow HR to close incorrectly entered transactions, reversing the operations	High	GOAL-01	Complete
F-REQ-10	The system shall keep track of the total hours worked and total amount paid to an employee on their records	Medium	GOAL-02	Complete
F-REQ-11	The system shall require payroll transactions to contain at least the following information: employee id, pay period, period hours worked, pay rate, and period pay	High	GOAL-01	Complete
F-REQ-12	The system shall optionally allow HR to mark a transaction as having been overtime work.	Medium	GOAL-01	Complete
F-REQ-13	The system shall require employee records to have unique IDs	High	GOAL-02	Complete
F-REQ-14	The system shall require transaction records to have unique IDs	High	GOAL-1	Complete
F-REQ-15	The system shall allow HR to create records recording an employees attendance status for a date	High	GOAL-3	Complete
F-REQ-16	The system shall only allow attendance records to contain the following attendance statuses: present, late, absent, sick, and vacation.	High	GOAL-3	Complete
F-REQ-17	The system shall allow HR to change the attendance status of an attendance record	High	GOAL-3	Complete
F-REQ-18	The system shall allow HR to delete attendance records	High	GOAL-3	Complete
F-REQ-19	The system shall display a full list of employee records when no filter has been applied	High	GOAL-4	Complete
F-REQ-20	The system shall allow HR to filter employee records based on employee ID	High	GOAL-4	
F-REQ-21	The system shall allow HR to filter employee records based on employee name	High	GOAL-4	
~~F-REQ-22~~	~~The system shall allow HR to filter employee records based on job title~~	~~Medium~~	~~GOAL-4~~	
~~F-REQ-23~~	~~The system shall allow HR to filter employee records based on start date at the company~~	~~Low~~	~~GOAL-4~~	

~~F-REQ-24~~	~~The system shall allow HR to filter employee records based on last day worked at the company~~	~~Low~~	~~GOAL-4~~	
F-REQ-25	The system shall allow base users to view payroll transactions filtered by their personal employee IDs	Medium	GOAL-5	
F-REQ-26	The system shall allow base users to view attendance records filtered by their personal employee IDs	Medium	GOAL-5	
F-REQ-27	The system shall allow for two types of users: administrators, and standard users	Medium		complete
F-REQ-28	The system shall ensure that admin users have access to all features mentioned in the functional requirements	High		
F-REQ-29	The system shall ensure that base users only have access to search features for records matching their employee IDs	High		
~~F-REQ-30~~	~~The system shall allow base users to filter transactions based on pay periods~~	~~Medium~~	~~GOAL-5~~	
F-REQ-31	The system shall allow base users to filter attendance records based on date	Medium	GOAL-5	
F-REQ-32	The system shall allow HR to filter transaction records based on employee IDs	High	GOAL-4	
F-REQ-33	The system shall allow HR to filter transaction records based on name	High	GOAL-4	
F-REQ-34	The system shall allow users to sign up as either admin or base user	High	GOAL-5	
F-REQ-35	The system shall allow users to log in to the system using existing accounts	High	GOAL-5	

*note: requirements that have been struck through have been considered out-of-scope due to time constraints

Non-Functional Requirements

ID	Requirement	Priority	Type
NF-REQ-01	The system shall allow users to access records at least 99.5% of the time	High	Reliability
NF-REQ-02	The system shall update viewable records in less than 30 seconds after an admin user has made a change	High	Performance
NF-REQ-03	The system shall lock an account after 5 failed login attempts over a 30 minute interval.	High	Security