南京信息工程大学

本科生毕业论文(设计)

Undergraduate Final Project

题目(Title)

Payroll Calculator Web Application Using Django Framework

学生姓名(Name)	Valetta Jesslyn (苏彦儒)
学号(Student ID)	20175344002
学院(College)	School of Computer and Software
专业(Major)	Software Engineering
指导教师(Supervisor)	Prof. Wang Haibin(王海彬)

二〇二一年五月二十日

Table of Contents

List of Figures

List of Tables

ABSTRACT

No matter how small or big a company is, it must have someone responsible for calculating the payroll of the company's employees. At times, payroll may be the most significant indirect cost of a company. As the solution to this problem, a payroll system can be used. The payroll system automates payroll calculation so that it saves time and can ensure the payment of the employees is paid in time. This payroll calculator web application is targeted to help small businesses in calculating their employee's salaries. It is intended for small business that employs piece-rate pay system. The system can be used in a device that has internet and browser application, such as Google Chrome, Firefox, Internet Edge, and Opera Browser. It is developed using the Django web framework.

Keywords: *Web Application；Payroll；Django*

1. Introduction

1.1. Overview

Payroll processing is essential in a company as it involves the payment of the company's workforce. Payroll processing calculates the salary, allowance, and deductions of employees that vary across different companies[1]. A business may encounter several payroll processing challenges, such as paying workers on time and accurately. In manual payroll processing, calculating payroll is time-consuming and tedious, especially in companies with a large number of employees. As the demand for timely, efficient, and reliable payroll raises, it increases automated processing systems' needs[2].

This project does exactly that; it automates payroll calculation. Although the web application can be used to calculate employees who have fixed daily salaries, it is developed mainly to calculate piece-rate workers' payments. In brief, the employer or anyone in charge of calculating the payroll should input all of the data, such as the employee's personal information, salary per day, attendance record, allowances, deductions, and others, into the web application. Then, to get the salary of all the employees, all we need to do is input the range of the date and click on the "Fetch" button. The system will calculate all employee salaries of that particular date range for us. The results can be viewed directly from the web applications or in a Portable Document Format (PDF) file, and the PDF file can be downloaded.

1.2. Scope

This payroll web application focuses on calculating the salary of employees at a given range of dates. This project is intended for small business that employs piece-rate pay system. It calculates the salary per day, allowance, deductions, and the completed processes of each employee. The details of each employee's salary are also generated. The web application is developed using the Django web framework and its default database, SQLite. The front-end design uses the combination of

HTML, CSS, JQuery, and Bootstrap framework. This project works on any device with internet and a browser application, such as Google Chrome, Firefox, Opera Browser, and Microsoft Edge.

1.3. Motivation

No matter how small or big a company is, it must have someone responsible for calculating the payroll of the company's employees. At times, payroll may be the most significant indirect cost of a company as the salary calculation process can be complicated[3]. Especially in a company that implements piece-rate payment, manually calculating each employee's salary often causes errors, resulting in a wrong paycheck. This project's motivation is to save time and money by increasing the efficiency of calculating employees' salaries, mainly in small manufacturing enterprises where piece-rate payment is employed. By using the payroll web application, an employer only has to input the data (employee's information, daily salary, attendance record, and other data) into the system. Then payroll system will automatically calculate each employee's salaries, thus reducing the possibility of wrong paychecks resulting from human errors.

1.4. Aim, Objectives, and Goals of the Project

Aim:

To develop an easy-to-use web application that automatically calculates each employee's salaries accurately, mainly for employees that are paid using a piece-rate pay system.

Objectives:

- Study how piece-rate payment is calculated
- Ensure the calculations of the system is accurate
- Design a user-friendly interface web application

Goals:

- Reduce the time of calculating employees' salaries
- Minimize the possibility of wrong paychecks resulting from manual calculation

- Keeps employee’s information and attendance record
- Keeps all order records and their corresponding processes
- Monitors each employees performance easily

1.5. Definitions and Acronyms

ASGI: Asynchronous Server Gateway Interface. Its documentation defines ASGI as “a spiritual successor to WSGI, intended to provide a standard interface between async-capable Python web servers, frameworks, and applications” [4].

Django: The official Django website defines *Django* as “a high-level Python Web framework that encourages rapid development and clean, pragmatic design” [5].

SQLite: The official SQLite website defines *SQLite* as “an in-process library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine” [6].

HTML: Hypertext Markup Language

CSS: Cascading Style Sheets

PDF: Portable Document Format

MVC: Model-View-Controller

MTV: Model-Template-View

MSME: Micro, Small, and Medium Enterprise

URL: Uniform Resource Locator

WSGI: Web Server Gateway Interface. Its documentation stated that WSGI is “a specification that describes how a web server communicates with web applications, and how web applications can be chained together to process one request” [7].

2. Literature Review

2.1. Introduction to Payroll and Piece-rate

The Cambridge Academic Content Online Dictionary defines the term payroll as “a list of a company’s employees and the amount each earns, or the total earnings a business gives to its employees” (Cambridge University Press, 2021)[8].

Generally, there are two categories of how payrolls are calculated: fixed wages and piece-rate pay [9]. Fixed wages refer to the employees that are paid fixed amounts. The company may calculate their payments either by fixed hourly rate or fixed salary. Hourly workers are paid based on the hours they spend working; for example, if an employee worked for 40 hours, then the employee's payment is equal to 40 multiplied by the agreed hourly rate[10]. While fixed salary workers are paid the same amount despite how long they work. Suppose the annual salary of an employee is $80,000 (eighty thousand dollars), and he/she is paid biweekly. Then his/her biweekly pay is around $3,077 gotten from $80,000 divided by 26 biweekly pay periods[11].

In comparison, piece rate means "a way of paying for work that is based on a fixed rate for a particular amount done rather than the time it takes to do the job" (Cambridge University Press, 2021) [12]. Piece-rate workers are paid based on the pieces or units they produce or did. The more they produce, the more they are paid [13].

The modern manufacturing industry in developed countries has gradually got rid of the piece-rate system. For example, in the US manufacturing industry, piece-rate wages had fallen from the primary form of remuneration a century ago to less than 5% in 2003 [14]. Regardless of the decline in developed countries, piece-rate wages are still very common in developing countries. For instance, according to a study on 3010 randomly picked home workers across six provinces in Indonesia, 90% of the interviewed respondents stated that they are paid by piece-rate pay system [15]. Several other studies also found that the majority of Micro, Small, and Medium Enterprises (MSMEs) employ piece-rate pay [16][17][18][19]. Hence, piece-rate pay still represents important payment mechanisms in industries where the output can be measured easily, such as in the construction and manufacturing industries (e.g., garments).

2.2. Current Research Progress

2.2.1. Indonesia’s Research Progress

According to the Central Bureau of Statistics (Badan Pusat Statistik (BPS)) in Indonesia, the number of MSMEs reaches 64 million [20]. Moreover, 99% of the economic structure in Indonesia comes from MSMEs [21]. Within those 64 million MSMEs, there are still a lot of MSMEs who do not exercise the application of accounting. According to research on the city of Banjarmasin, the application of the accounting system in small and medium enterprises is low [22].

Another research on Jember regency, which consists of 31 sub-districts, found that the understanding of accounting in MSMEs is still low [23]. According to other research made on a city called Salatiga, only 39.22% of 51 MSMEs make records of the salary [24]. Several other researches also show that the use of digital technology in MSMEs is low [25][26]. The use of digital technology in MSMEs is essential as it can increase the productivity of the enterprise.

The most popular payroll system in Indonesia can only calculate fixed wages (hourly, monthly, or weekly). This might be one of the reasons why small businesses that employ piece-rate pay (or piecework) still count the salary manually and do not use any payroll system.

2.2.2. International Research Progress

According to a report made by the International Labour Organization (ILO) published in October 2019, MSMEs are responsible for two-thirds of all jobs worldwide[27]. As the business continues to grow, they often find that their current operations struggle to accommodate that growth. One of the most common operations to feel that pressure is the payroll management process. Based on research, the global payroll software market is expected to grow at a Compound annual growth rate (CAGR) of more than 9% during the forecast period 2019-2025[28]. However, there has been a lack of software developed to address the needs of MSMEs. Surveys performed by Deloitte in 2018 found that 21% of companies in the world are planning to use a cloud-based system to process payroll

[29]. Therefore there is an increase in demand for cloud-based payroll software.

Payroll software providers usually offer the software in two ways, an integrated payroll system and a standalone payroll system. Generally, the payroll system is integrated as a part of accounting software or human resource management (HRM) software. This kind of software may make the system complicated because it has many functions. Another type of payroll is the standalone payroll system, this kind of system is usually less complicated than the former one. Furthermore, not all payroll software provides support for calculating the piece-rate pay system. Most of them can only calculate fixed wages.

2.3. Introduction to Django Framework

Django is a high-level Python web application framework that allows fast development of web applications. Django accomplishes this intent with an efficient and clean design [30]. A little bit about Django's history, it was first developed in 2003 by the World Online (a newspaper Web operation) developers named Adrian Holovaty and Simon Willison. For around two years, they were constantly adding improvements to this framework. In 2005, they released the first public version, named after one of the best guitarists of all time, Django Reinhardt [31].

Django is now running under the supervision of the Django Software Foundation (DSF). It has more than 3,000 packages dedicated to the Django framework and over 1,000 contributors [32]. Django is a back-end framework that can be utilized to solve connection problems with databases, handling content management, user authentication, and much more. Subsequently, web developers do not need to write the same code for similar website modules, such as database connection and administration interface. Compared to other frameworks, Django is more effortless to use. Therefore it is favoured among web developers.

All functions that appear in the Django framework are in the form of a web application. These applications only need to be imported. Hence we can focus more on the website's unique applications without dealing with all these back-end issues[30].

2.3.1. Django’s Feature

a) Great documentation: Compared with other open source technologies, Django offers one of the best documentation in the market. It is well organized, so any desired function can be searched with ease.
b) Remarkably fast: Django aims to help developers transform applications from idea to completion as fast as possible.
c) Completely loaded: Django contains numerous features that can be utilized to deal with common web development tasks. Django is capable of handling content management, user authentication, data validation, sessions, sending emails, and many other tasks.
d) Secure: Django attaches great importance to security and helps developers evade many common security faults, such as cross-site scripting (XSS), cross-site request forgery (CSRF), SQL injection, and clickjacking. Django’s user authentication system gives a secure way to manage user accounts and passwords.
e) Exceptionally scalable: Django is flexibly scalable; thus, it can satisfy the world’s heaviest traffic demands. It uses a “shared nothing” architecture, which implies you can add hardware on any level (web/application server, cache server, or database server). The framework neatly separates various components, such as its database layer and application layer. It also comes with a simple but powerful caching framework.
f) Astonishingly versatile: Governments, organizations, and companies have adopted Django to develop all kinds of things, from social networks to content management systems to scientific computing platforms.
g) Implemented in Python: Python is easy to learn; therefore Django framework is easier to learn than other frameworks.

3. System Architecture Design

Django architecture is similar to Model-View-Controller (MVC) architecture[33], although there are some differences. Django calls its architecture Model-Template-View (MTV) [31]. A Django model is the sole, ultimate source of information about the project's data. Usually, each model maps to one database table. Each model is a Python class, and each attribute represents one database field [34].

As stated in the Django documentation website, the "view" specifies the data presented to the user, not the user interface, as in the view of MVC architecture. In Django's architecture interpretation, the view is not necessarily the appearance of the data but the data to be displayed. It describes which data we see, not how the data is viewed [31]. The view (also known as view function) is "a Python function that takes a Web request and returns a Web response"[35]. The response can be anything, for example, an image, HTML contents of a web page, an XML document, or a redirect. The view itself contains the logic needed to return the response [35].

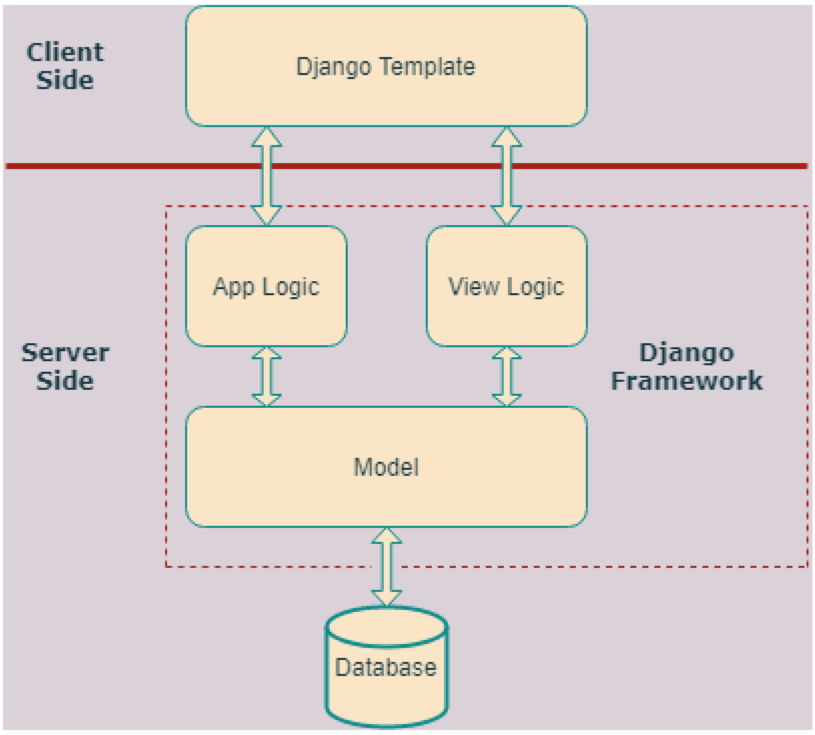
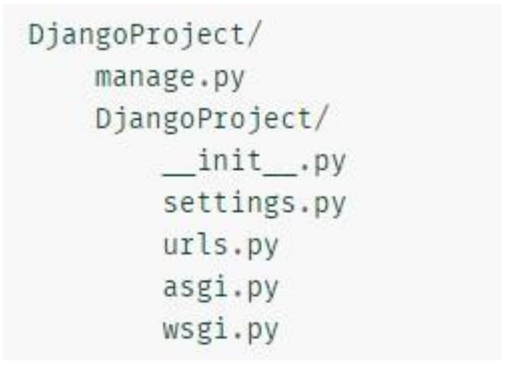


Figure 2.3.1 Simplified Django architecture based on The Django book, "Django's Structure – A Heretic's Eye View." https://djangobook.com/mdj2-django-structure/ (accessed Apr. 17, 2021).

The template is the one that describes how the data is presented. It is the one that handles the user interface part. The view performs the logical function and interacts with the model to obtain data and then modify the template accordingly [36]. A template comprises the static part of the requested HTML output and some special syntax, which describes how to insert the dynamic content. Django templates are text documents or Python strings that are marked up with the Django template language. The template engine can recognize and interpret certain constructs. The main ones are variables and tags. As stated in Django documentation, “tags provide arbitrary logic in the rendering process”[37]. Tags are enclosed by {% and %}, for example, {% if condition %} Output something {% endif %}. Variables are enclosed by "{{" and "}}", for example: {{ myVar }}.

In Django, the “controller” in MVC is probably the framework itself, the one that sends the request to the appropriate view according to the Django URL configuration [31][36]. The simplified Django architecture is summarized in Figure **2.3.1**[38].

3.1. Django Project



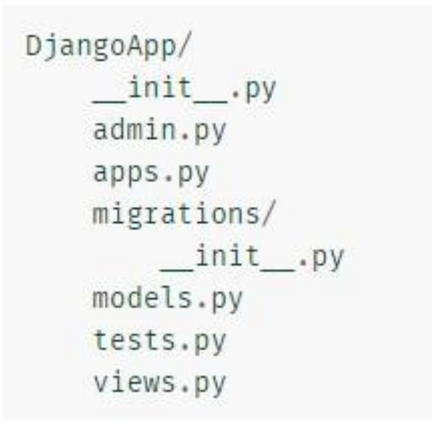
```
DjangoProject/
    manage.py
    DjangoProject/
        __init__.py
        settings.py
        urls.py
        asgi.py
        wsgi.py
```

Figure 3.1.1 Automatically generated Django project folder

When a Django project is created, it will automatically create a manage.py file and five other python files inside a folder with the project’s name, i.e., __init__.py, asgi.py, settings.py, urls.py, and wsgi.py. For instance, if we create a Django project named “DjangoProject”, then the files automatically created by Django are shown in Figure **3.1.1**. The explanations of each file are listed below[39][36]:

- The outer DjangoProject/ root directory is the project’s container, and it can be renamed to whatever we like.
- The manage.py file is a command-line utility that allows us to interact with the Django project in numerous ways.

- The inner DjangoProject/ directory is the actual Python package for the project.
- The ***__init__.py*** file is an empty file that informs Python that this folder (the inner DjangoProject/ directory) is a Python package.
- The *asgi.py* file is the entry point for ASGI-compatible webservers to serve the Django project.
- The *settings.py* file contains the settings/configuration of the Django project. Every Django project needs to have a settings file.
- The *urls.py* file consists of the URL declarations of the Django project. It is the "table of contents" of the website. By default, it includes the URL pattern of the administrator.
- The *wsgi.py* file is the entry point for WSGI-compatible webservers to serve the Django project.



```
DjangoApp/
    __init__.py
    admin.py
    apps.py
    migrations/
        __init__.py
    models.py
    tests.py
    views.py
```

Figure 3.1.2 Automatically generated Django app folder

3.2. Django Application

In Django, a project is a combination of configurations and applications for a specific website. A project can consist of multiple applications. An application can be in numerous projects. Similar to when a Django project is created, when a Django app is created, it will also automatically generate the basic directory structure like in Figure **3.1.2**. The explanations are listed below[38][40]:

- The migrations directory is where Django stores the changes of the project's database.

- The *admin.py* file is the file to register the application’s models among the Django admin application.
- The *apps.py* file is a configuration file standard for all Django applications.
- The *models.py* file contains the application’s models.
- The *tests.py* consists of the test procedures that are run when testing the application.
- The *views.py* file consists of the views of the application.

4. System Analysis and Design

4.1. System Requirements

4.1.1. Functional Requirements

Functional requirements are the main requirements that specify the system. It defines how the system should operate and its behavior under specific inputs or conditions. It can be data manipulation, the behavior of the system, output, or functional change after an action. A functional requirement specifies how the system should perform under some action [41]. The functional requirements of this project are listed below:

a) Authorization:

- Only the authorized users (the ones who calculated the employee salary and/or anyone who is permitted to access the web application by the employer or someone who has the authority to do so) can access or login into the web application.
- The employees should not be able to access the web application.
- Logged in user can log out of the web application.

b) Employee data:

- The authorized users should be able to view all employees’ personal information that has been inserted into the database.

- The authorized users can filter the employees either by their first name, last name, phone number, email, address, position, employment type, notes, or date hired.
- The authorized users should be able to add/create new employee's personal information into the database as long as all required field (first name) has been entered.
- The hired date of an employee is automatically recorded when a new employee's personal information is added.
- The authorized users should be able to update all employees' personal information in the database.
- The authorized users should be able to delete one or more employee's personal information from the database.

c) Position:

- The authorized user should be able to view all positions that have been inserted into the database.
- The authorized user should be able to add/create a new position as long as the position's name doesn't exist in the database (the position's name must be unique).
- The authorized user should be able to edit the position name in the database.
- The authorized user should be able to delete one or more positions from the database.

d) Employment Type:

- The authorized user should be able to view all employment types that have been inserted into the database.
- The authorized user should be able to add/create a new employment type as long as the employment type name doesn't exist in the database (the employment type name must be unique), description field can be empty.
- The authorized user should be able to edit employment types name and descriptions in the database.

- The authorized user should be able to delete one or more employment types from the database.

e) Order:

- The authorized users should be able to view all orders that have been inserted into the database.
- The authorized user can filter the orders either by their code, name, description, or date created.
- The authorized user should be able to add/create new order into the database as long as all required fields (order code, name, and quantity) have been entered correctly.
- The newly added order code must not exist in the database (order code must be unique).
- The quantity order must be an integer greater or equal to 0.
- The date created of an order is automatically set when a new order is added.
- The authorized user should be able to edit order information in the database.
- The authorized user should be able to delete one or more orders from the database.

f) Process:

- The authorized users should be able to view all processes that have been inserted into the database.
- The authorized user can filter the processes either by their order code, process name, description, or price.
- The authorized user should be able to add/create a new process into the database as long as all required fields (order code, process name, price, and quantity) have been entered correctly.
- The price must be greater or equal to 0.
- The quantity must be an integer greater or equal to 0. If there is a completed process of the current edited process, then the quantity of the edited process can’t be less than the total completed process.

- The authorized user should be able to edit process information in the database.
- The authorized user should be able to delete one or more processes from the database.

g) ***Completed Process:***

- The authorized users should be able to view all completed processes that have been inserted into the database.
- The authorized user can filter the completed processes either by their process, employee, or date recorded.
- The authorized user should be able to add/create new completed processes into the database as long as all required fields (process id, employee id, and quantity) have been entered correctly.
- The quantity must be an integer greater or equal to 1 and less or equal to the process quantity minus the total completed process quantity all employees have done.
- The date recorded of a completed process is automatically set when a new completed process is added.
- The authorized user should be able to edit completed process information in the database.
- The authorized user should be able to delete one or more completed processes from the database.

h) ***Daily Salary:***

- The authorized users should be able to view all daily salaries that have been inserted into the database.
- The authorized user should be able to add/create a new daily salary into the database as long as all required fields (employee id and daily salary) have been entered correctly.
- Each employee id can only have one record of daily salary.
- The daily salary must be greater or equal to 0.

- The last-modified date and time are automatically updated every time change is made to the daily salary.
- The authorized user should be able to edit daily salary information in the database.
- The authorized user should be able to delete one or more daily salary records from the database.

i) Salary:

- The authorized users should be able to view employees' salary after inputting the date range of the salary that want to be calculated.
- The authorized users should be able to view the employees' salaries in a PDF file.
- The authorized users should be able to download the PDF file of the employees' salary.
- The authorized user should be able to see each employee's salary details (includes all completed processes, daily salary, allowances, and deductions of the employee) of the given date range.
- The authorized users should be able to view employee salary details in a PDF file.
- The authorized users should be able to download the PDF file of the employee salary details.

j) Attendance:

- The authorized users should be able to view all employees' attendance records that have been inserted into the database.
- The authorized user can filter the attendance records either by their employee id, date, or percentage.
- The authorized user should enter the employee id and the date range of the attendance records he/she wants to input before they can add any attendance record. After inputting the employee id and the date range, the authorized user

can add the attendance records as long as all required fields (date and attendance percentage) have been entered correctly.

- Each employee can only have one record of attendance on the same day.
- The attendance percentage must be greater or equal to 0 and less or equal to 100 (0≤ attendance percentage ≥100). Attendance percentage zero means the person is absent, attendance percentage less than 100 means the person is late, and attendance percentage of 100 means the person is on time.
- The authorized user should be able to edit the attendance records in the database.
- The authorized user should be able to delete one or more attendance records from the database.

k) Allowance:

- The authorized users should be able to view all employees' allowances that have been inserted into the database.
- The authorized user can filter the allowance records either by their employee id, date, or description.
- The authorized user should be able to add/create employee's allowance into the database as long as all required fields (employee, amount, and date) have been entered correctly.
- The allowance amount must be greater than 0.
- The authorized user should be able to edit the allowance record in the database.
- The authorized user should be able to delete one or more allowance records from the database.

l) Deduction:

- The authorized users should be able to view all employees' deductions that have been inserted into the database.
- The authorized user can filter the deduction records either by their employee id, date, or description.

- The authorized user should be able to add/create an employee’s deduction into the database as long as all required fields (employee, amount, and date) have been entered correctly.
- The deduction amount must be greater than 0.
- The authorized user should be able to edit the deduction record in the database.
- The authorized user should be able to delete one or more deduction records from the database.

4.1.2. Non-Functional Requirements

Non-functional requirements define the criteria to assess the system’s operation and not its specific behaviors [42]. It indicates the system’s quality properties. It can assist in assessing the overall function’s quality and the system from a user viewpoint[41]. The non-functional requirements of this project are listed below:

a) Security

The web application can only be accessed by the people who have the username and password to prevent it from unauthorized users. In other words, the web application can only be accessed by the authorized user. By default, Django does not store raw passwords. Django only stored the hashed password. So, even if someone somehow manages to view or obtain the user database, they still will not know the users’ password.

b) Reliability

The web application should function without any failure, and it should not give the wrong calculation of the salary. It has to make sure the data inserted is correct; for instance, quantity must be an integer or date filed must be a date.

c) Availability

The web application should be available for use anytime, any day (24 hours a day, 7 days a week).

d) Usability

The web application should be easy to use, even without any prior training of the system. Also, the design should be simple.

e) Portability

The web application can be used across different devices, such as phones, laptops, or tablets, of different operating systems, like Windows, Android, iOS, macOS, Linux, as long as the devices have internet and a browser application, such as Google Chrome, Firefox, Opera Browser, and Microsoft Edge.

4.2. Project’s Use Case Diagram

The use case diagram below describes the process that the authorized user can do with the system. Manageemployeesuse case means that the authorized user can view, add, edit, and delete employee data. Managepositionsuse case means that the authorized user can view, add, edit, and delete positions. Manage employment types use case means that the authorized user can view, add, edit, and delete employment types. Manage orders use case means that the authorized user can view, add, edit, and delete orders. Manage processes use case means that the authorized user can view, add, edit, and delete processes. Manage completed processes use case means that the authorized user can view, add, edit, and delete completed processes. Manage daily salaries use case means that the authorized user can view, add, edit, and delete daily salary. View salary use case describes when the authorized user view a specific date range of salary, but they must first input the date range. They also can view the employee’s salary detail. Manage attendances use case means that the authorized user can view, add, edit, and delete attendance records. The authorized user must first input the date range and employee ID before they can add attendance record. Manage allowances use case means that the authorized user can view, add, edit, and delete allowances. Manage deductions use case means that the authorized user can view, add, edit, and delete deductions.

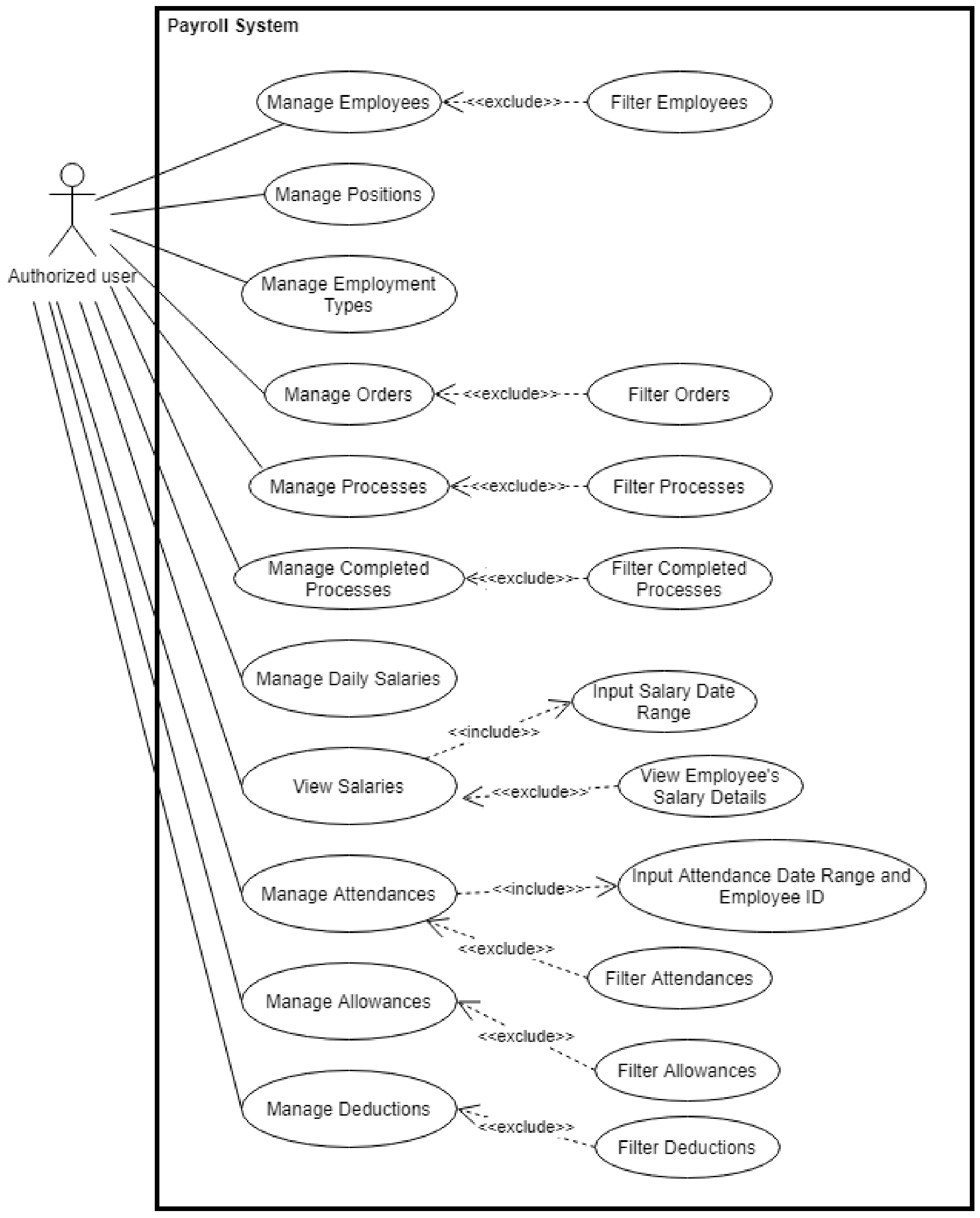


Figure 4.2.1 The project's use case diagram

All of the filter use cases mean that the authorized user can filter the data for specific attributes (each filter has different attributes it can filter). The payroll web application use case diagram is shown in Figure **4.2.1**.

4.3. Project's Activity Diagram

Several activity diagrams of this project are shown below:

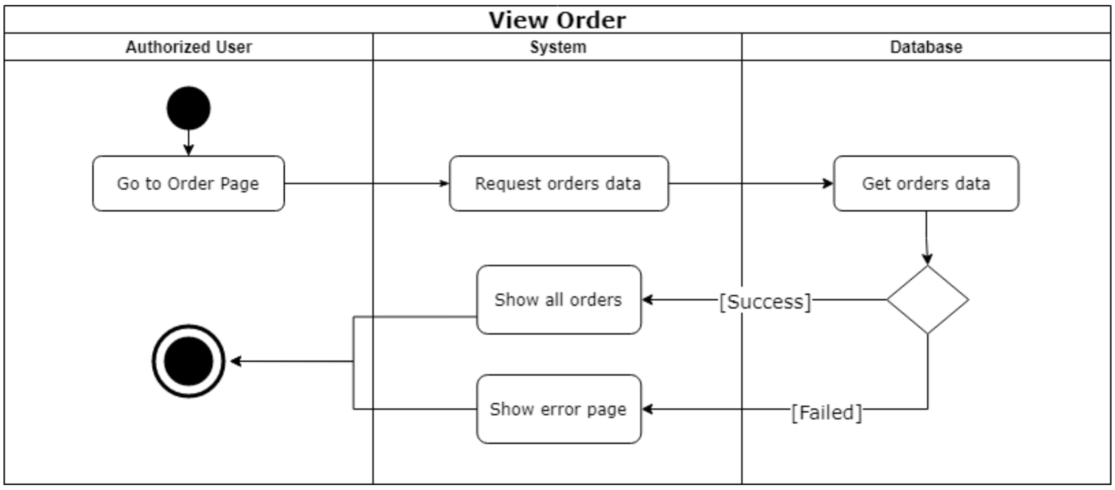


Figure 4.3.1 Activity diagram of view order

After the user logins into the system, they can view employees' personal information, positions, employment types, orders, processes, completed processes, allowances, deductions, attendance records, employees' daily salary, and employees' salary that are stored in the database. Except for employees' salaries, the others have a similar process. The activity diagram above (Figure **4.3.1**) describes the actions that the authorized user, system, and database do when viewing order. The system will get the data from the database and show it to the authorized user.

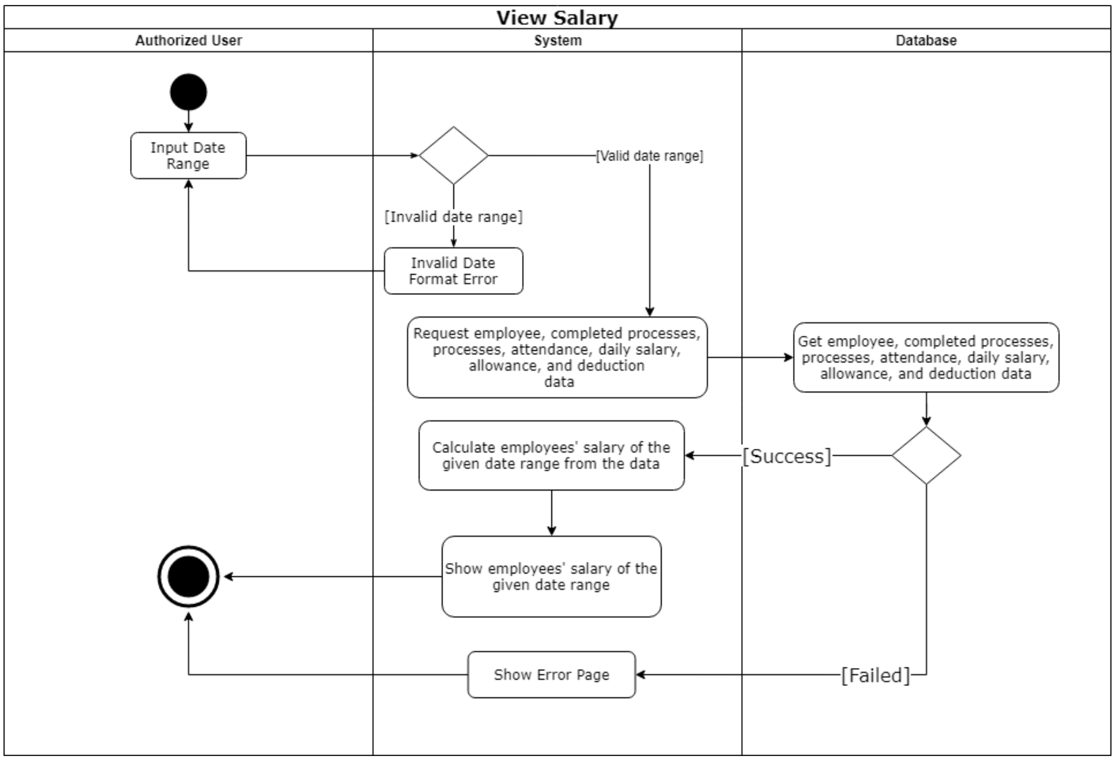
The actions that the authorized user, system, and database do when viewing salary are described in Figure **4.3.2**. To view the employees' salary, the user must input the date range of the salary to be calculated. The system will get all employee data that still work in the company, then automatically calculate each of their salaries by multiplying each of the employee's completed processes in the given date range with the corresponding process payment per unit (piece rate payment). Then add them with the total attendance percentage of the given date range divided by 100 and multiplied by the employee's daily salary, plus the allowance amount, and minus by the deduction amount that the employee may have. The formula to calculate one employee' salary is shown below:

$$Piece\ rate\ payment = \sum Completed\ process\ quantity \times its\ payment/unit$$

Note: Piece-rate payment calculates all completed processes that the employee has done within the given date range.

$$Salary = Piece\ rate\ payment + \left(Daily\ salary \times \frac{Total\ attendance\ percentage}{100}\right) + \sum Allowance - \sum Deduction$$

Note: Daily salary is the daily salary of the employee. Total attendance percentage is the total attendance percentage of the employee within the given date range. $\sum Allowance$ means all allowances the employee has within the given date. $\sum Deduction$ means all deductions the employee has within the given date.



View Salary
Authorized User
System
Database
Input Date Range
[Valid date range]
[Invalid date range]
Invalid Date Format Error
Request employee, completed processes, processes, attendance, daily salary, allowance, and deduction data
Get employee, completed processes, processes, attendance, daily salary, allowance, and deduction data
Calculate employees' salary of the given date range from the data
[Success]
Show employees' salary of the given date range
Show Error Page
[Failed]

Figure 4.3.2 Activity diagram of view salary

After the user logins into the system, they can also add new employees' personal information, positions, employment types, orders, processes, completed processes, allowances, deductions, attendance records, and employees' daily salary to the database. Each of them has different required fields and constraints. Required fields must be filled, and constraints must be satisfied before they can be added to the database.

Generally, the system will get the form and show it to the user. If the user has input all required fields correctly, then they will be added to the database. Otherwise, the system will give an error message telling the user which fields of the form are inputted incorrectly. Figure **4.3.3** describes the actions that the authorized user, system, and database do when adding a completed process.

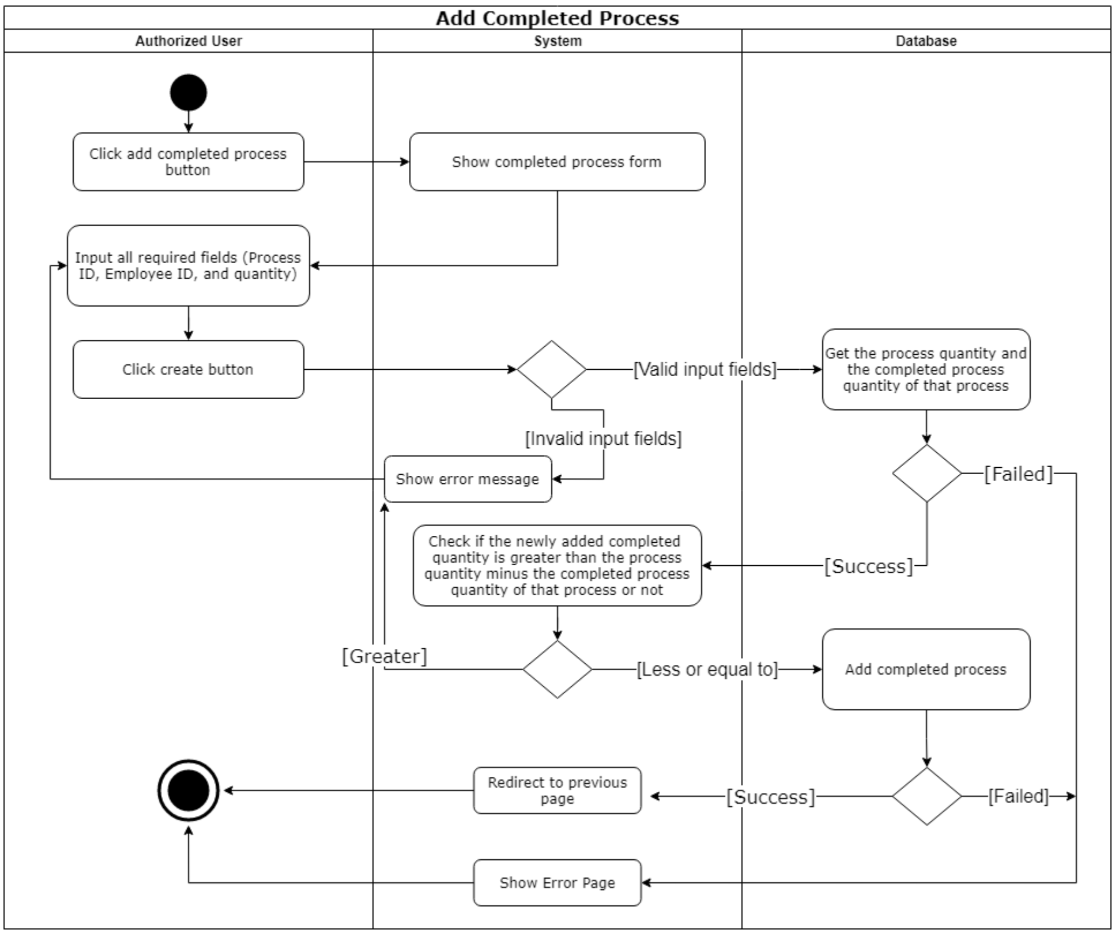

Add Completed Process
Authorized User
System
Database
Click add completed process button
Show completed process form
Input all required fields (Process ID, Employee ID, and quantity)
Click create button
[Valid input fields]
Get the process quantity and the completed process quantity of that process
[Invalid input fields]
Show error message
[Failed]
[Success]
Check if the newly added completed quantity is greater than the process quantity minus the completed process quantity of that process or not
[Greater]
[Less or equal to]
Add completed process
Redirect to previous page
[Success]
[Failed]
Show Error Page

Figure 4.3.3 Activity diagram of add completed process

The authorized user can also edit employees' personal information, positions, employment types, orders, processes, completed processes, allowances, deductions, attendance records, and employees' daily salary that are stored in the database. Each of them has a different form and field constraint they must satisfy to be able to update the data in the database. The activity diagram of the editing process is shown in Figure **4.3.4**.

Generally, the user has to click on the ID, and then the system will request the data with that ID from the database, and it will display the edit form with its data to the user. The user can then edit the fields and click on the update button. The system will check whether the fields satisfy the constraints. If it does, the system will update the data in the database. If it does not, then the system will give the error message to the user, so the user knows which field is inputted incorrectly.

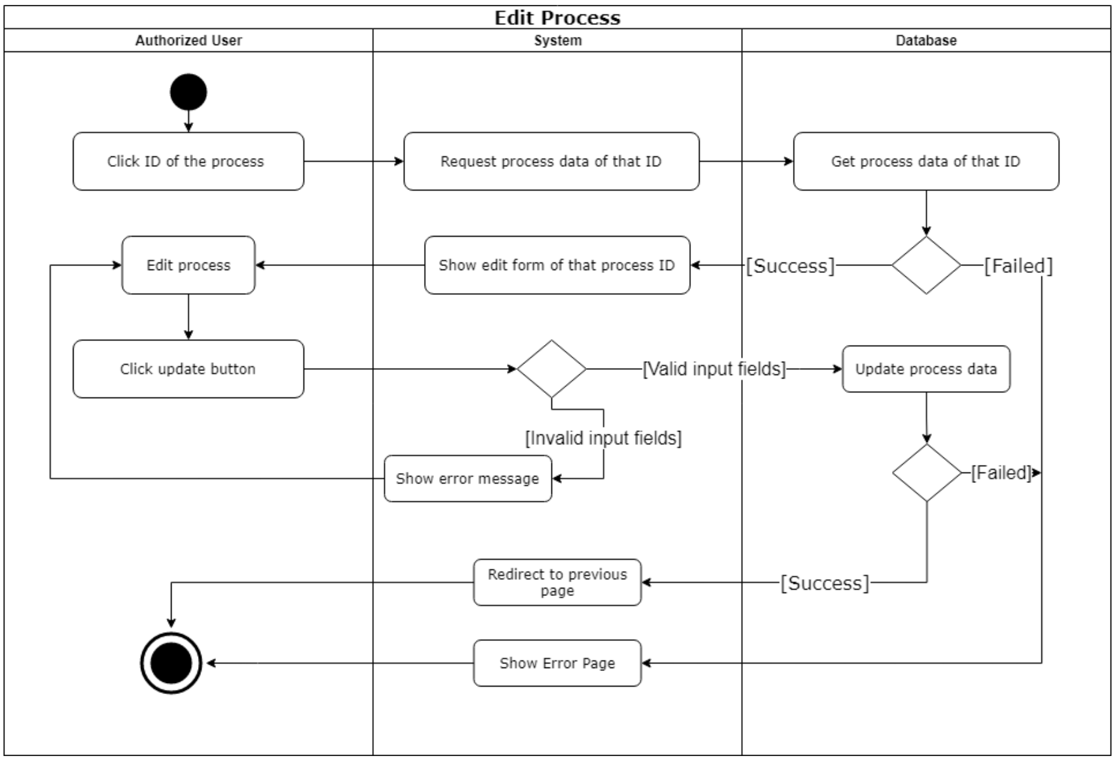


Figure 4.3.4 Activity diagram of edit process

The authorized user can also delete employees' personal information, positions, employment types, orders, processes, completed processes, allowances, deductions, attendance records, and employees' daily salary that are stored in the database. Their processes are all similar. Basically, the user only has to tick the checkbox (located on the left side) of the row that wants to be deleted, then click on the delete button. There will be a confirmation pop-up to ensure that the user does not click the delete button unintentionally. The system will delete the selected data from the database after the user clicks on the delete button in the confirmation pop-up. The activity diagram of the deleting process is shown in Figure **4.3.5**.

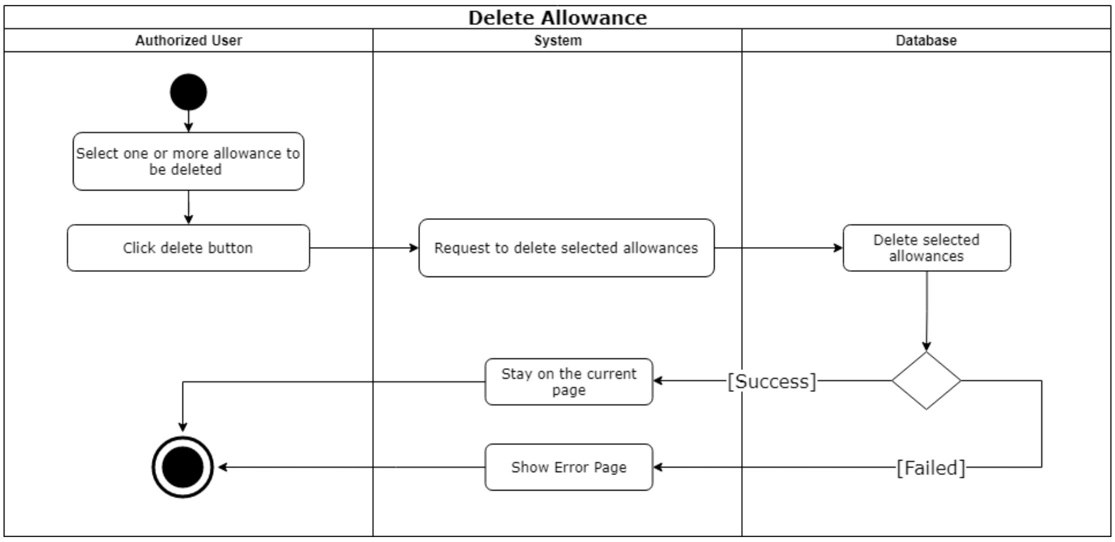


Figure 4.3.5 Activity diagram of delete allowance

The authorized user can also filter employees' personal information, orders, processes, completed processes, allowances, deductions, and attendance records that are stored in the database. Basically, the user has to click on the filter button, then input the fields to be filtered, and click the search button. The system will request the data that matches the filtered request from the database, and it will show the filtered data back to the user. The activity diagram of the filtering attendance records is shown in Figure **4.3.6**.

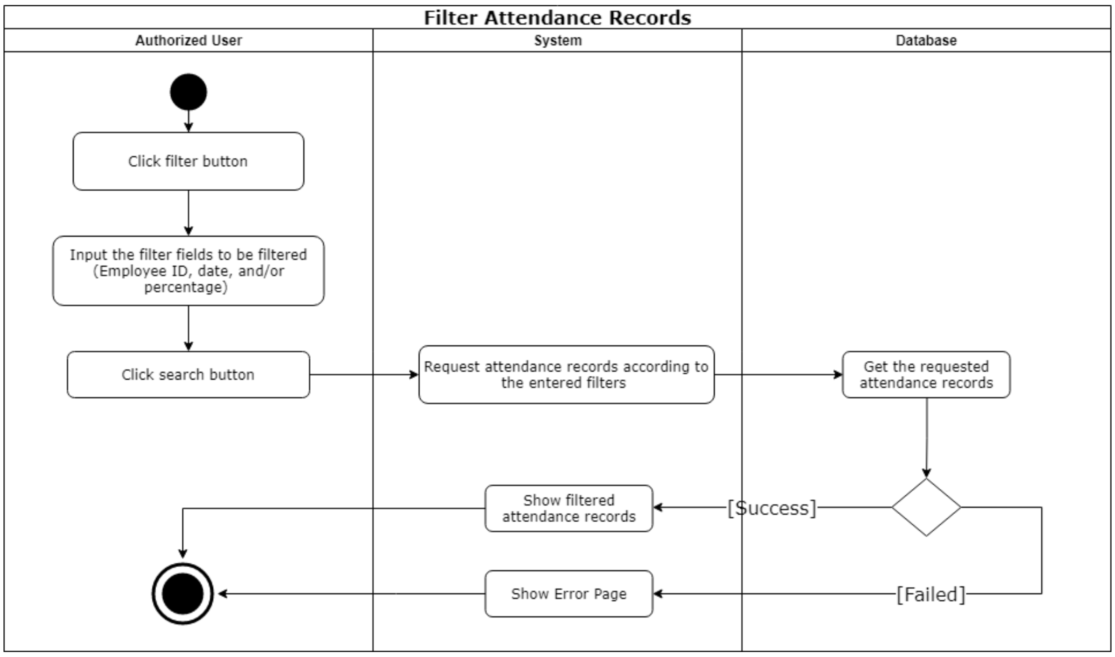

Filter Attendance Records
Authorized User
System
Database
Click filter button
Input the filter fields to be filtered (Employee ID, date, and/or percentage)
Click search button
Request attendance records according to the entered filters
Get the requested attendance records
Show filtered attendance records
[Success]
Show Error Page
[Failed]

Figure 4.3.6 Activity diagram of filter attendance records

4.4. Project’s Database Design

The project’s entity relationship diagram (ERD) is shown in Figure **4.4.1**. It is also shown at the end of this thesis with a bigger size.

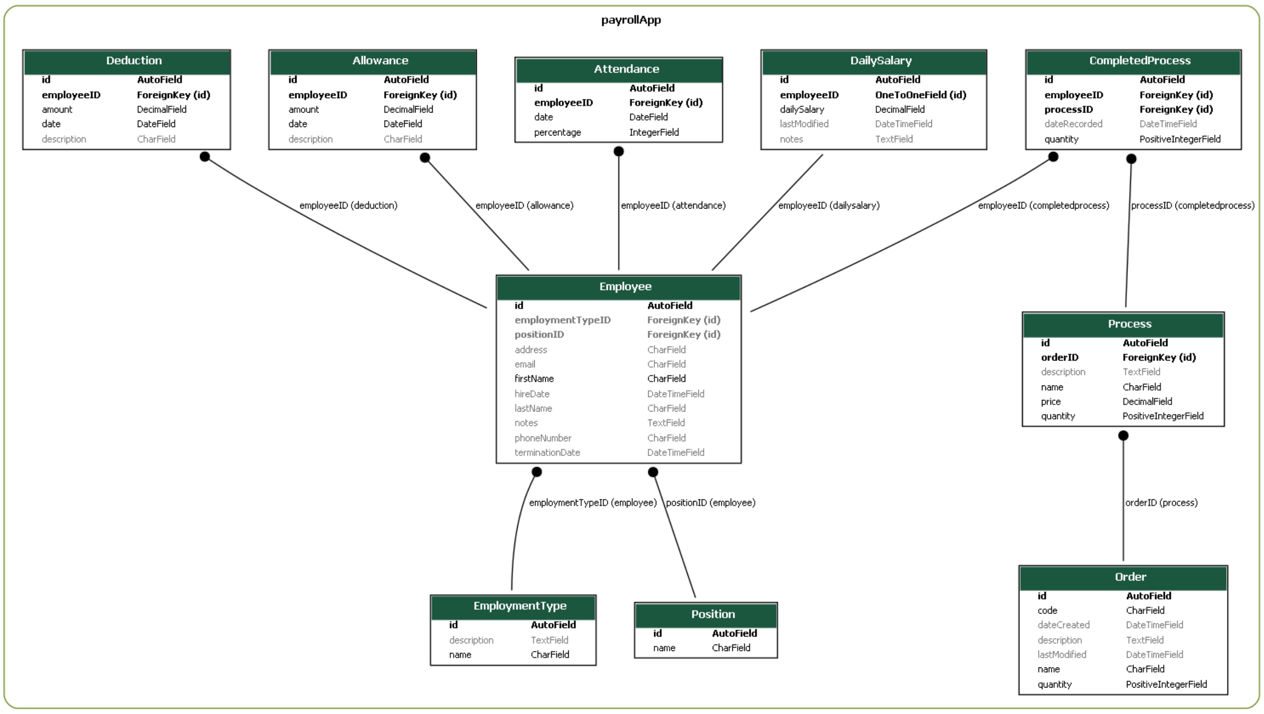


Figure 4.4.1 Project’s ERD

5. System Implementation

5.1. Development Tools

The development tools used during the development of the system are as follows:

1. Text editor: Visual Studio Code (Version 1.55.0)
2. Virtual environment: Python virtual environment (venv module)
3. Browser application: Google Chrome and Firefox
4. Diagram design: Online diagram maker (diagrams.net)
5. Other tools: GitHub and Heroku

All of the codes are written in Visual Studio Code, including the HTML, CSS, JQuery, and Python files. The diagrams in the thesis are made using an online diagram maker called diagrams.net. The virtual environment is used to isolate the project’s files from the computer directories so it can have its own independent installed packages to avoid conflict with other packages [43][44]. Windows PowerShell is used to create the virtual environments, the Django project, and the Django app. It is also used to install python packages needed in this project. GitHub is used to keep changes in the project, so the codes can be reverted if there are unknown bugs that are usually hard to find. The branch in GitHub is used to isolate development work, so it allows us to fix bugs and safely experiment with new ideas and not messing up with the master codes [45]. This project is deployed on Heroku (URL: https://payroll-calculator-system.herokuapp.com).

5.2. Development Specification

5.2.1. Hardware Specification

The hardware specifications used in the development of the web applications are as follows:

- Device type: Laptop computer
- Device brand: ASUS X550Z
- Processor: AMD FX-7500 Radeon R7, 10 Compute Cores 4C+6G
- System type: 64-bit Operating System, x64-based processor
- Memory: 4GB

5.2.2. Software Specification

The software specifications used in the development of the web applications are as follows:

- Operating system: Windows 10 Home Single Language
- Programming language: Python (Version 3.8.3)
- Command-line: Windows PowerShell
- Web framework: Django (Version 3.1.7)

5.3. The System's Django Application

5.3.1. Models

The *models.py* file contains the application's models. Each Django model class usually maps to a single database table, and each attribute of the model maps to a database column. An instance of the Django model class represents one record in the database table. For example, the employee model of the system is shown below:

```
class Employee(models.Model):
  firstName  = models.CharField(max_length=30)
  lastName   = models.CharField(max_length=30, blank=True, null=True)
  phoneNumber= models.CharField(max_length=15, blank=True, null=True)
  email     = models.CharField(max_length=100, blank=True, null=True)
  address   = models.CharField(max_length=200, blank=True, null=True)
  positionID = models.ForeignKey('Position', default=None, null=True,
 on_delete=models.SET_DEFAULT, blank=True)
  employmentTypeID = models.ForeignKey('EmploymentType', default=None
, null=True, on_delete=models.SET_DEFAULT, blank=True)
  hireDate        = models.DateTimeField(auto_now_add=True)
  terminationDate = models.DateTimeField(blank=True, null=True)
  notes           = models.TextField(blank=True, null=True)

  def __str__(self):
    if self.lastName:
      return str(self.id)+" | "+self.firstName.capitalize()+" "+self.
lastName.capitalize()
    else:
      return str(self.id)+" | "+self.firstName.capitalize()
```

The employee model above has 10 columns, namely firstName, lastName, phoneNumber, email, address, positioned, employmentTypeID, hireDate, terminationDate, and notes.

```
lastName   = models.CharField(max_length=30, blank=True, null=True)
```

Each field class type determines the column type; it tells the database the kind of data to be stored. The line of code above is an example of a single field labeled *lastName*, the field class type *models.CharField* specifies the column type string, and it is given three arguments:

- max_length=30 -- specifies the maximum length of the field in characters
- blank=True -- meaning lastName field is allowed to be empty
- null=True -- meaning Django will store empty values as NULL in the database

5.3.2. URLs

The *urls.py* file consists of the URL declarations of the Django project. It is the “table of contents” of the website. The URL patterns of the system are too long; thus, only a few of them will be displayed.

```
urlpatterns = [
    path('', views.home_view, name="home"),
    path('login/', auth_views.LoginView.as_view(template_name='accoun
t/login.html'), name="login"),
    path('logout/', auth_views.LogoutView.as_view(next_page='/'), nam
e = 'logout'),

    path('salary/date/', views.inputDateSalary_view, name="inputDate-
salary"),
    path('salary/', views.salary_view, name="salary"),
    path('salary/pdf_view/', views.ViewSalaryPDF.as_view(), name="vie
w-salary-pdf"),
    path('salary/pdf_download/', views.DownloadSalaryPDF.as_view(), n
ame="download-salary-pdf"),
    path('salary/details/<employee_id>/', views.salary_of_employee_de
tails_view, name="salary-details-employee"),
    path('salary/details/pdf/view', views.ViewSalaryOfEmployeeDetails
PDF.as_view(), name="view-salary-of-employee-details-pdf"),
    path('salary/details/pdf/download/', views.DownloadSalaryOfEmploy
eeDetailsPDF.as_view(), name="download-salary-of-employee-details-
pdf"),
    . . .
    . . .
    ]
```

As stated before, Django contains a lot of built-in features for common web development tasks, the login and logout of this system utilize Django built-in user authentication system.

```
path('salary/', views.salary_view, name="salary")
```

Django will run through all URL patterns in order then stops at the first one that matches the requested URL. Once it finds a matching URL pattern, Django will import and calls the Python functions (the views) given in the argument. In the example above, when Django will call the *salary_view* function. If none of the URL patterns matches the requested URL, Django invokes the error-handling view. The URL patterns can also be named; in the example above, it is named salary. So now we can use {% url ‘salary’ %} instead of hard coding the URL in the sidebar. This way, even when we want to change the URL patterns (salary/) to something else, we only need to change the *urls.py* file.

5.3.3. Views

The *views.py* file consists of the views of the application. Django view function, or simply view, is a Python function that takes a web request and returns a web response. It contains the logic necessary to return the response. The response can be an image, or the HTML contents, or a 404 error, or a redirect, and much more. There is a lot of view in the system, but the main part of the system is to calculate the salary so only the view of the system's salary page will be shown here.

```
@login_required(login_url='login')
def salary_view(request):
  context = getSalary(request)
  return render(request, 'salary/salaries.html', context)
```

The *@login_required(login_url='login')* is a decorator that checks if the user is logged in or not. It makes sure that only the signed in user can view the salary. If the user is logged in, the view is executed normally. However, if the user is not logged in, it will redirect to the login page.

The salary view will return a HttpResponse object with the rendered text, which is the combination of the template 'salary/salaries.html' with the given context dictionary. Because the code for calculating the salary will be used not only in the salary view but also in the view salary PDF part so, in order to not have redundant code, the code is put in a function named *getSalary()*. The *getSalary()* function is shown below:

```
def getSalary(request):
  start = request.session.get("startDate")
  end = request.session.get("endDate")

  endPlus1 = datetime.strptime(end, "%Y-%m-%d")
  endPlus1 = endPlus1 + timedelta(days=1)
  completedProcesses = CompletedProcess.objects.filter(dateRecorded__
range=[start, endPlus1])
  employees          = Employee.objects.all().exclude(terminationDate
__isnull=False) #exclude employee that has been terminated
  attendances        = Attendance.objects.filter(date__range=[start,
endPlus1])
  allowance          = Allowance.objects.filter(date__range=[start, e
ndPlus1])
  deduction          = Deduction.objects.filter(date__range=[start, e
ndPlus1])
```

```
  salaries = []
  i = 0
  for employee in employees:
    pieceRate = 0
    currCompletedProcesses = completedProcesses.filter(employeeID=emp
loyee.id).values('processID', 'quantity')
    for currCompletedProcess in currCompletedProcesses:
      qty = currCompletedProcess['quantity']
      processPrice = Process.objects.get(id=currCompletedProcess['pro
cessID'])
      pieceRate = pieceRate + (qty*getattr(processPrice, 'price'))

    currAttendances = attendances.filter(employeeID=employee.id).valu
es('percentage')
    attendancePercentage = 0
    for i in range(len(currAttendances)):
      currPercentage = currAttendances[i]['percentage']
      attendancePercentage = attendancePercentage + currPercentage
    if attendancePercentage > 0:
      attendancePercentage = attendancePercentage/100

    try:
      dailySalaryObj = DailySalary.objects.get(employeeID=getattr(emp
loyee, 'id'))
      salary = float(getattr(dailySalaryObj, 'dailySalary'))*attendan
cePercentage

    except:
      salary = 0

    currAllowances = allowance.filter(employeeID=employee.id).values(
'amount')
    allowanceAmt = 0
    for i in range(len(currAllowances)):
      currAmt = currAllowances[i]['amount']
      allowanceAmt = allowanceAmt + currAmt

    currDeductions = deduction.filter(employeeID=employee.id).values(
'amount')
    deductionAmt = 0
    for i in range(len(currDeductions)):
      currAmt = currDeductions[i]['amount']
      deductionAmt = deductionAmt + currAmt

    salaries.append(Salary(employee, salary, pieceRate, allowanceAmt,
 deductionAmt))
    i = i+1
  flag   = True
  if not completedProcesses and not attendances and not allowance and
 not deduction:
    flag=False

  total = 0
  for salary in salaries:
    total = total + salary.total

  context = {
    'salaries':salaries,
    'flags':flag,
    'startDate': start,
    'endDate': end,
    'total': total
  }
  return context
```

The *getSalary()* function takes the start and end date of the salary to be calculated from the session. The start and end date session is set when the user inputs the date range, which will be the page before the salary page is shown. Then the completed processes, attendances, allowances, and deductions record within these dates are taken from the database. All of the employees that are still within the company, meaning he/she has not been terminated, will also be obtained from the database. After all of the required data are obtained, it will compute the piece-rate pay, fixed salary, allowances, and deductions of each employee if there are any. To calculate the piece-rate pay of each employee, the system will iterate through all the completed process data which the employee does, get the quantity of that completed process, and multiply it with the price per unit of the process. The second thing the system will compute is the fixed salary of each employee. It will first filter the attendance records and calculate the total attendance percentage of the employee. Then the attendance percentage will be divided by 100 and multiplied by the employee's daily salary if there is any. Some employee may not have a daily salary, which means his/her fixed salary will always be zero (0). Next, the system iterates through allowances and deductions data. After calculating each employee's completed processes, attendances, allowances, and deductions, the *Salary* (shown below) class will be created and put into a list named salaries. The

Salary class will automatically calculate the total salary of each employee.

```
class Salary:
  def __init__(self, employeeID, salary, pieceRate, allowance, deduct
ion):
    self.employeeID = employeeID
    self.salary = salary
    self.pieceRate = pieceRate
    self.allowance = allowance
    self.deduction = deduction
    self.total = float(salary)+float(pieceRate)+float(allowance)-
float(deduction)
```

If there are no records in the database within the start and end date, the variable *flag* will be set to false. The variable *flag* will be used in the templates to mark that there is no salary to be calculated in the date range. Lastly, the total salary of all the employees will be calculated.

5.4. User Interface

The screenshots of the project's web application are shown below:

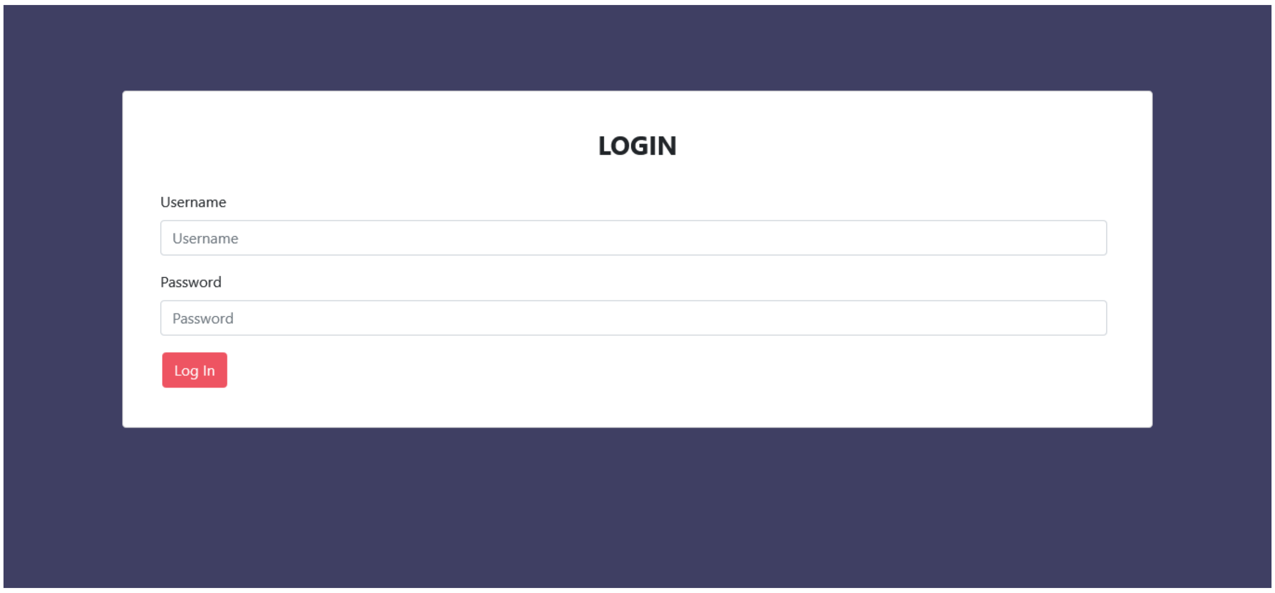
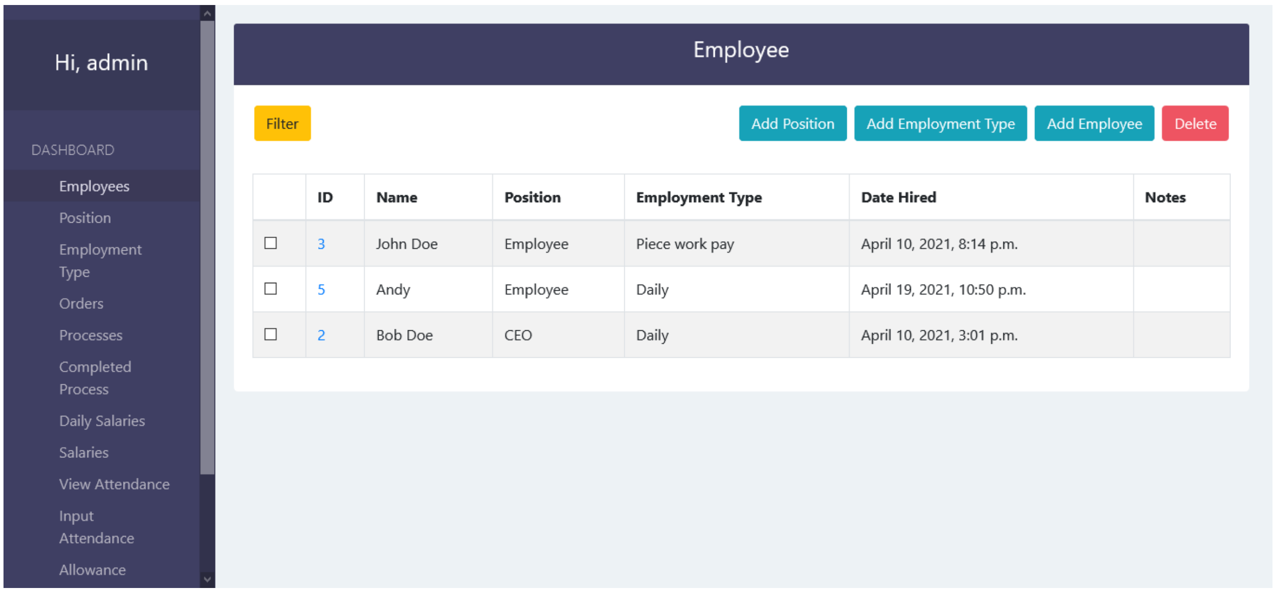
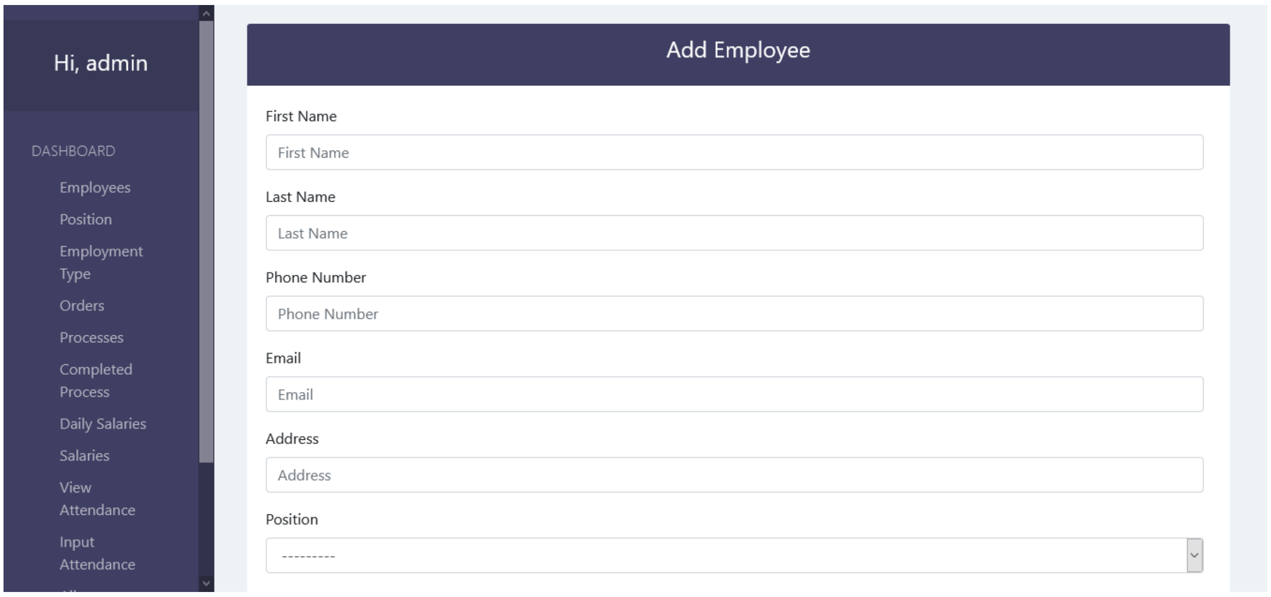


Figure 5.4.1 Login page



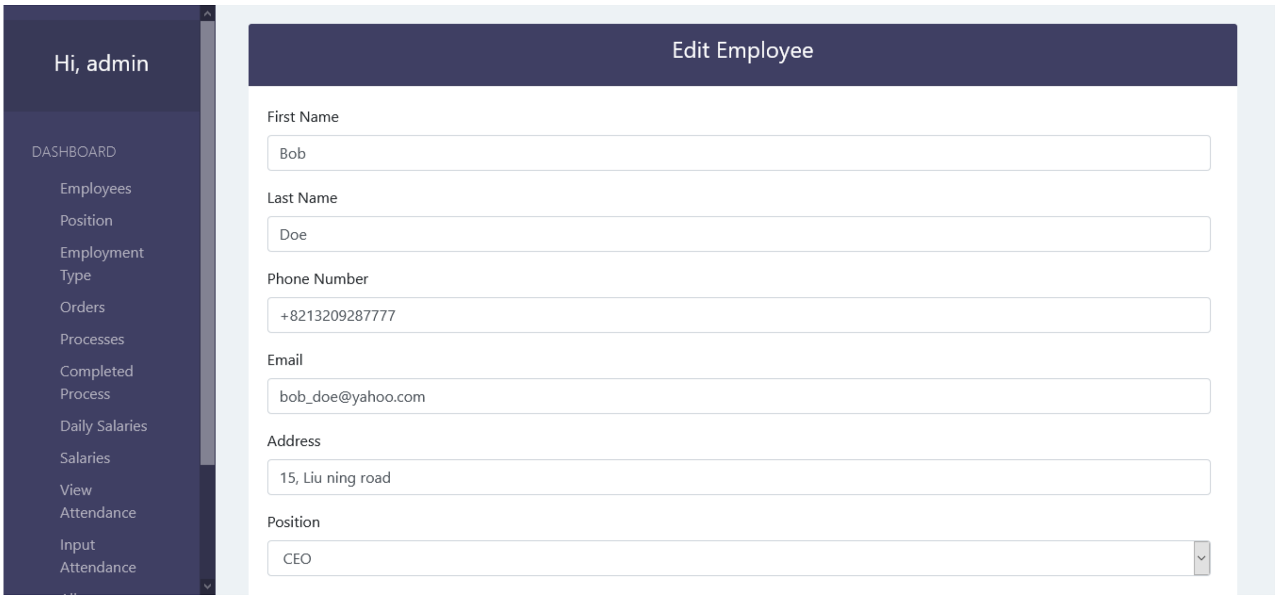
	ID	Name	Position	Employment Type	Date Hired	Notes
☐	3	John Doe	Employee	Piece work pay	April 10, 2021, 8:14 p.m.	
☐	5	Andy	Employee	Daily	April 19, 2021, 10:50 p.m.	
☐	2	Bob Doe	CEO	Daily	April 10, 2021, 3:01 p.m.	

Figure 5.4.2 View employee page



First Name	First Name
Last Name	Last Name
Phone Number	Phone Number
Email	Email
Address	Address
Position	---------

Figure 5.4.3 Add employee page



Edit Employee

First Name: Bob

Last Name: Doe

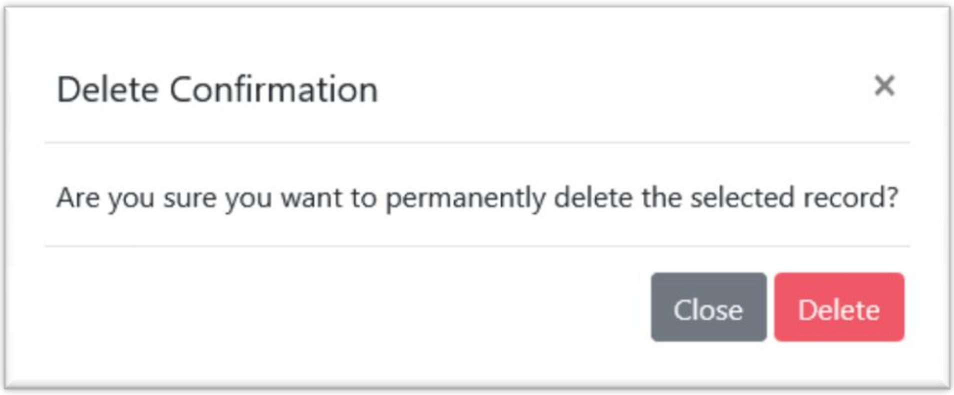
Phone Number: +8213209287777

Email: bob_doe@yahoo.com

Address: 15, Liu ning road

Position: CEO

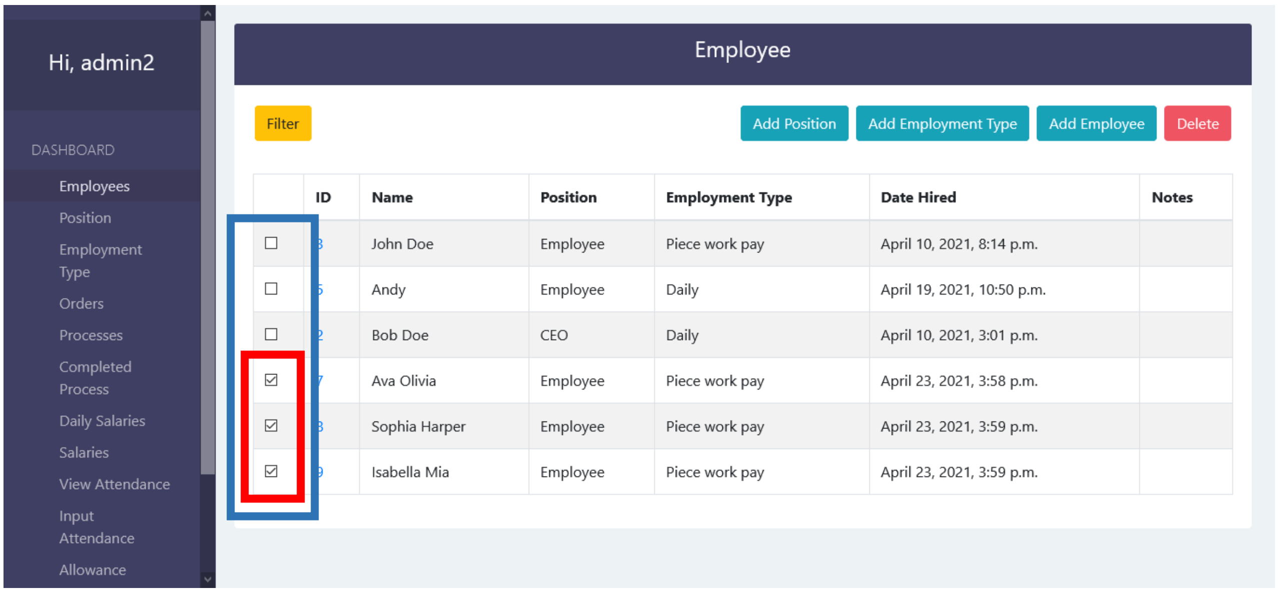
Figure 5.4.4 Edit employee page



Delete Confirmation

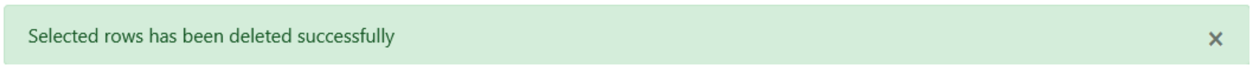
Are you sure you want to permanently delete the selected record?

Figure 5.4.5 Delete confirmation pop up



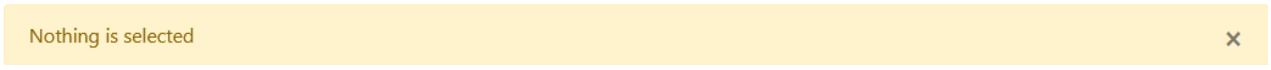
	ID	Name	Position	Employment Type	Date Hired	Notes
☐	3	John Doe	Employee	Piece work pay	April 10, 2021, 8:14 p.m.	
☐	5	Andy	Employee	Daily	April 19, 2021, 10:50 p.m.	
☐	2	Bob Doe	CEO	Daily	April 10, 2021, 3:01 p.m.	
☑	7	Ava Olivia	Employee	Piece work pay	April 23, 2021, 3:58 p.m.	
☑	8	Sophia Harper	Employee	Piece work pay	April 23, 2021, 3:59 p.m.	
☑	9	Isabella Mia	Employee	Piece work pay	April 23, 2021, 3:59 p.m.	

Figure 5.4.6 Checkbox



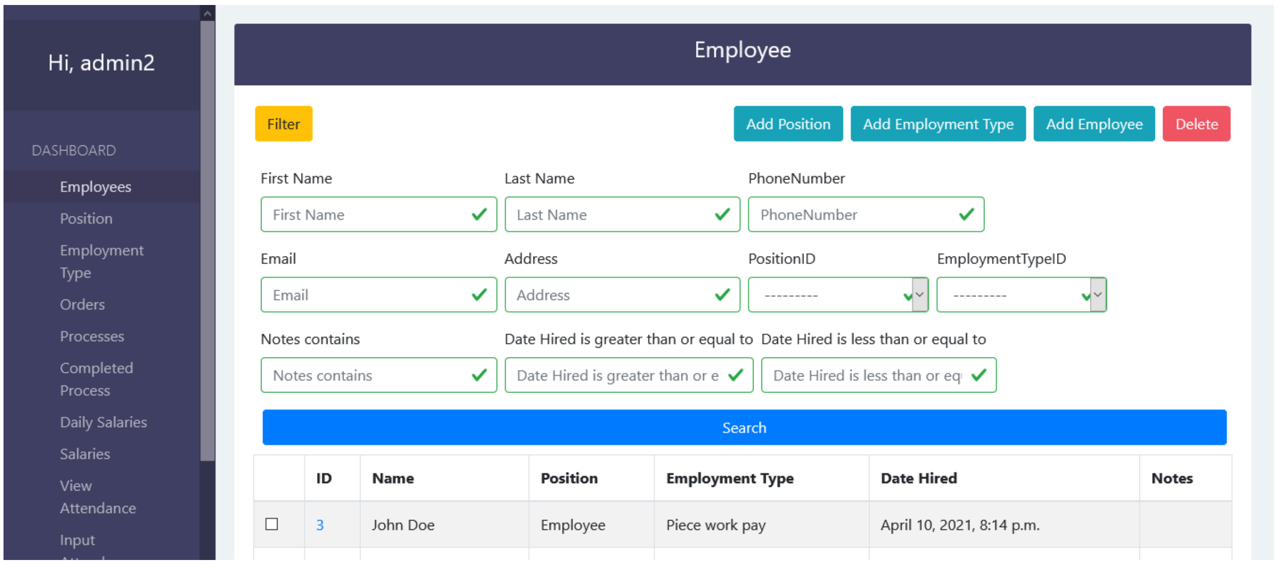
Selected rows has been deleted successfully ×

Figure 5.4.7 Delete successful message



Nothing is selected ×

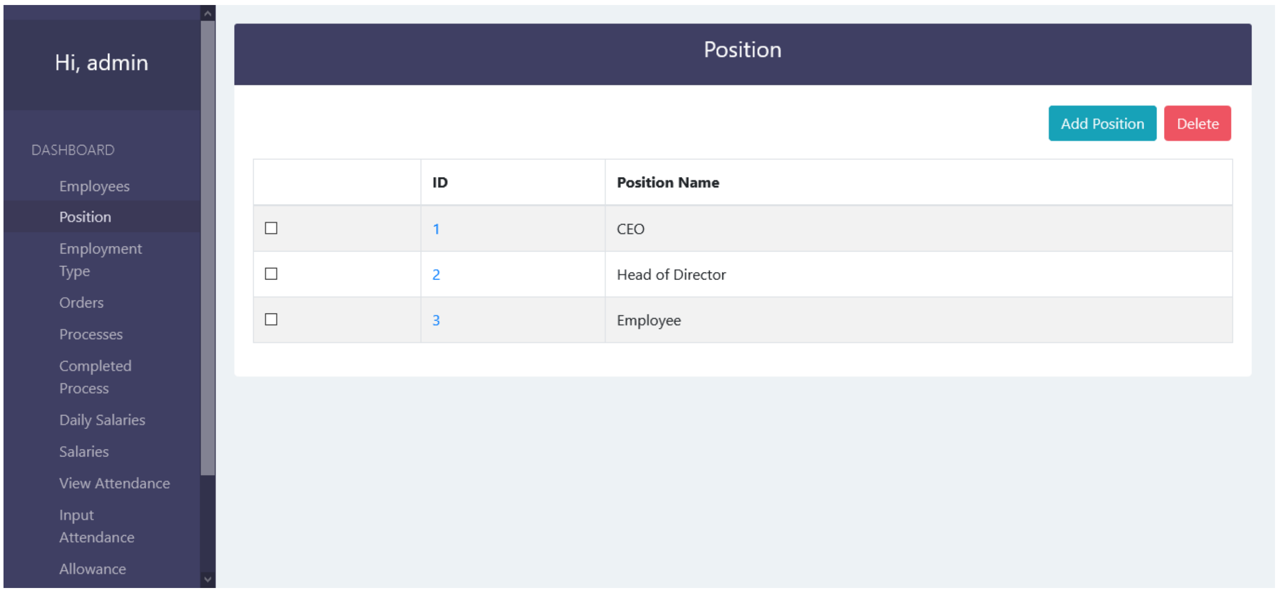
Figure 5.4.8 Nothing is selected message



First Name: First Name
Last Name: Last Name
PhoneNumber: PhoneNumber
Email: Email
Address: Address
PositionID: ---------
EmploymentTypeID: ---------
Notes contains: Notes contains
Date Hired is greater than or equal to: Date Hired is greater than or e
Date Hired is less than or equal to: Date Hired is less than or eq

	ID	Name	Position	Employment Type	Date Hired	Notes
☐	3	John Doe	Employee	Piece work pay	April 10, 2021, 8:14 p.m.	

Figure 5.4.9 Employee page with the filter part



	ID	Position Name
☐	1	CEO
☐	2	Head of Director
☐	3	Employee

Figure 5.4.10 View position page

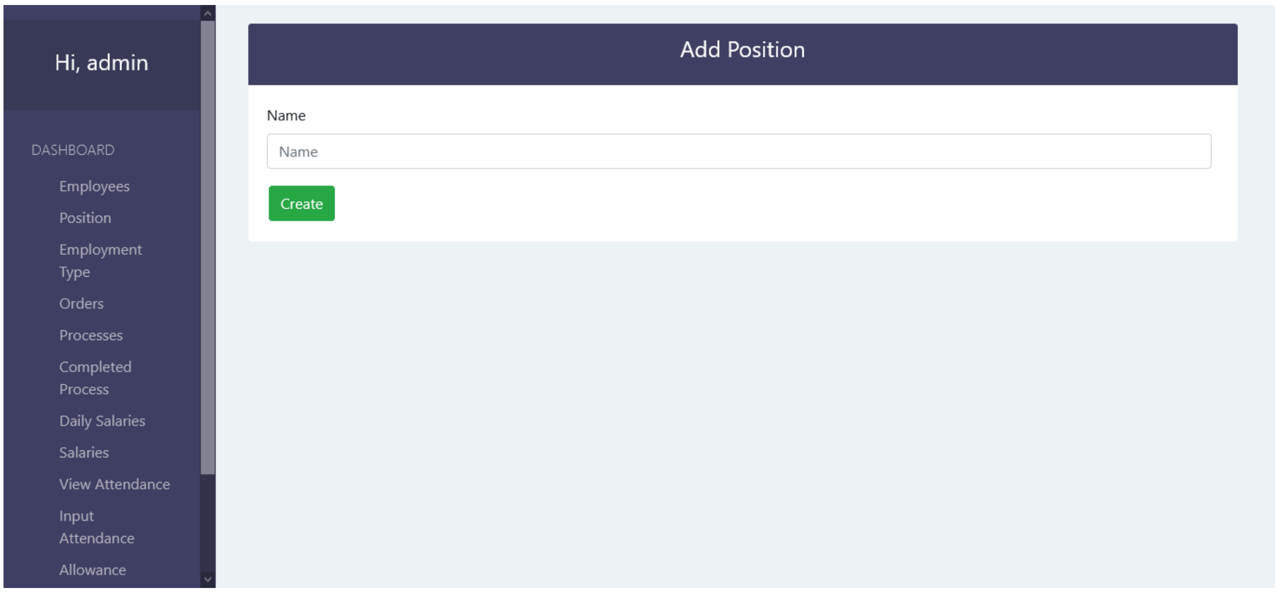


Figure 5.4.11 Add position page

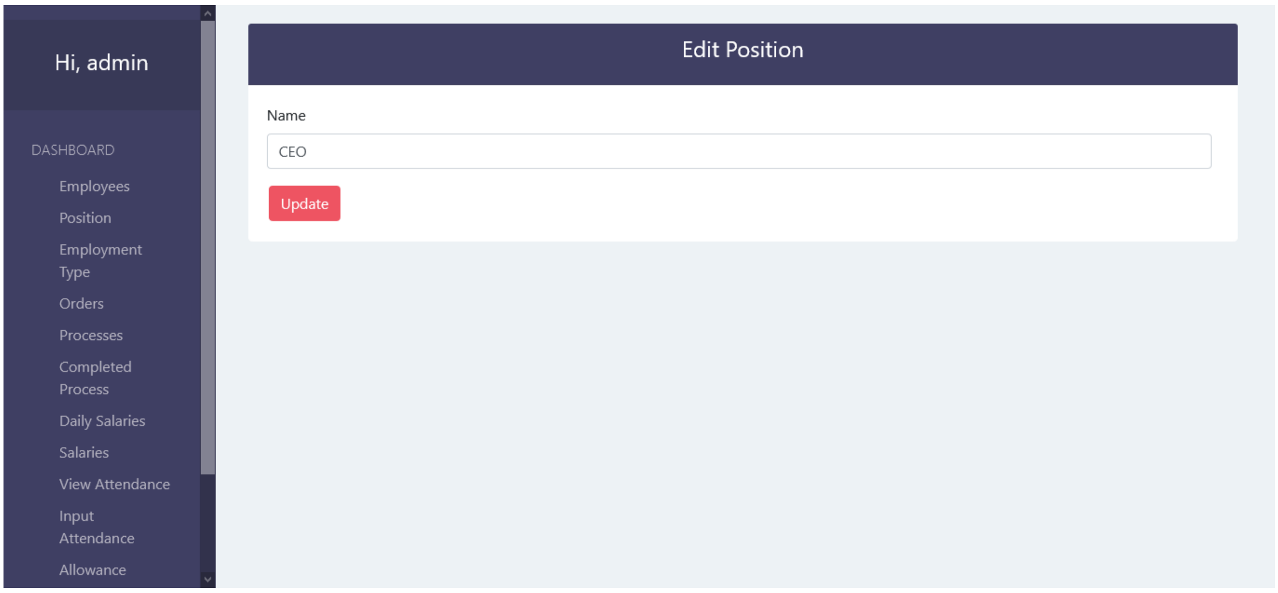
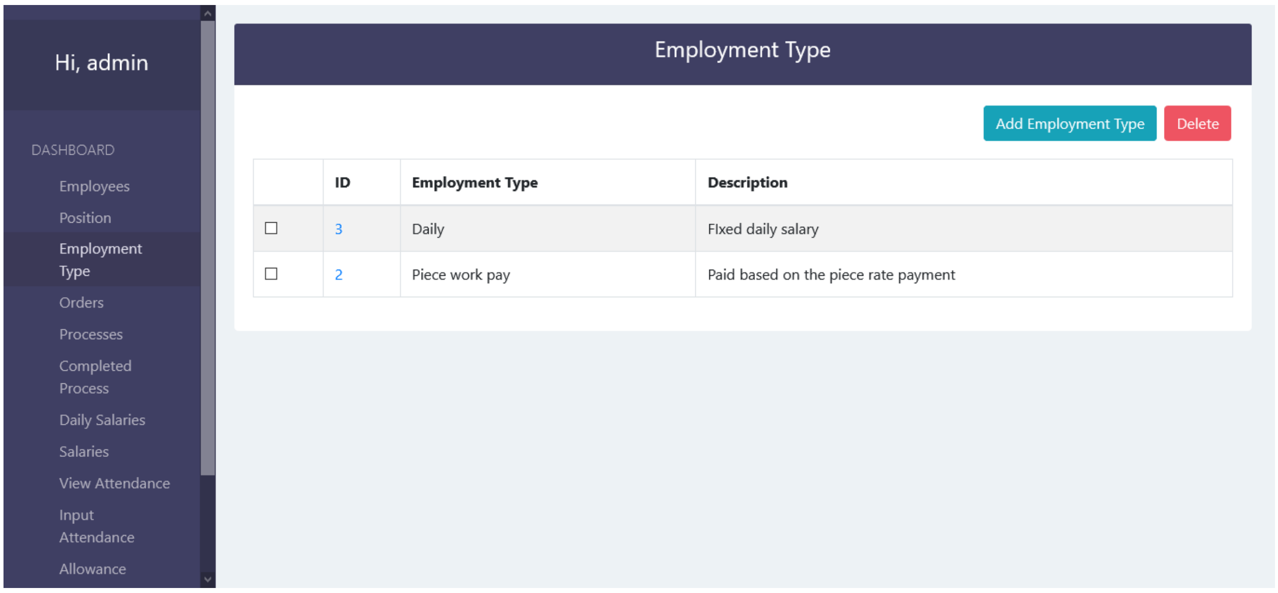


Figure 5.4.12 Edit position page

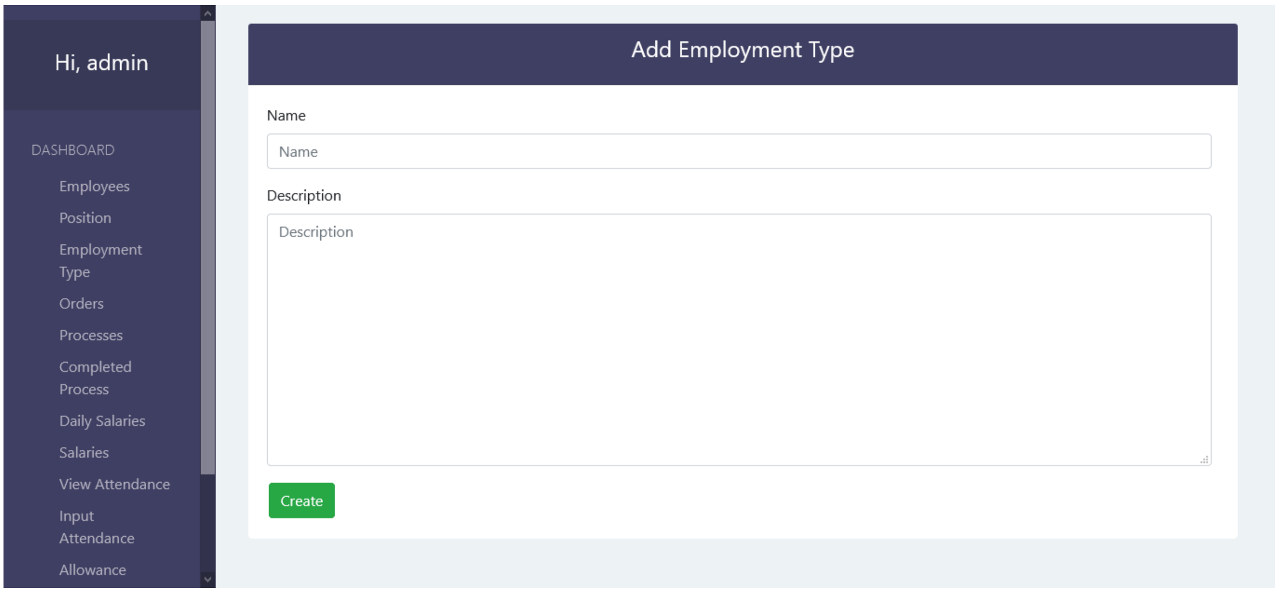


Employment Type

Add Employment Type | Delete

	ID	Employment Type	Description
☐	3	Daily	FIxed daily salary
☐	2	Piece work pay	Paid based on the piece rate payment

Figure 5.4.13 View employment type page



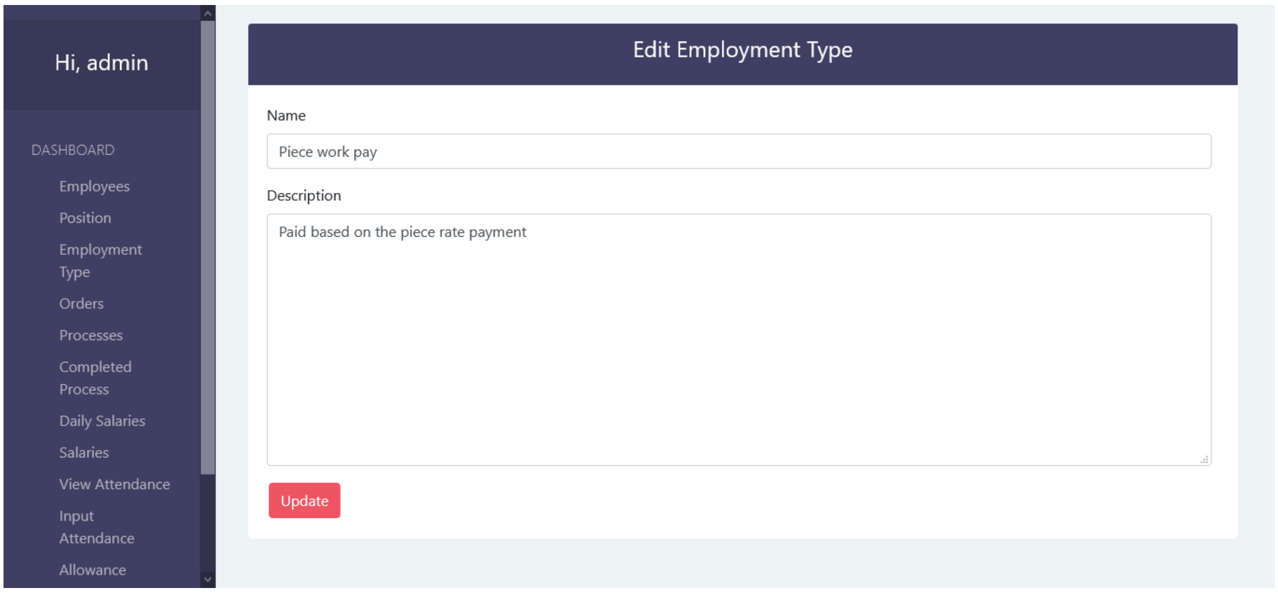
Add Employment Type

Name: Name

Description: Description

Create

Figure 5.4.14 Add employment type page



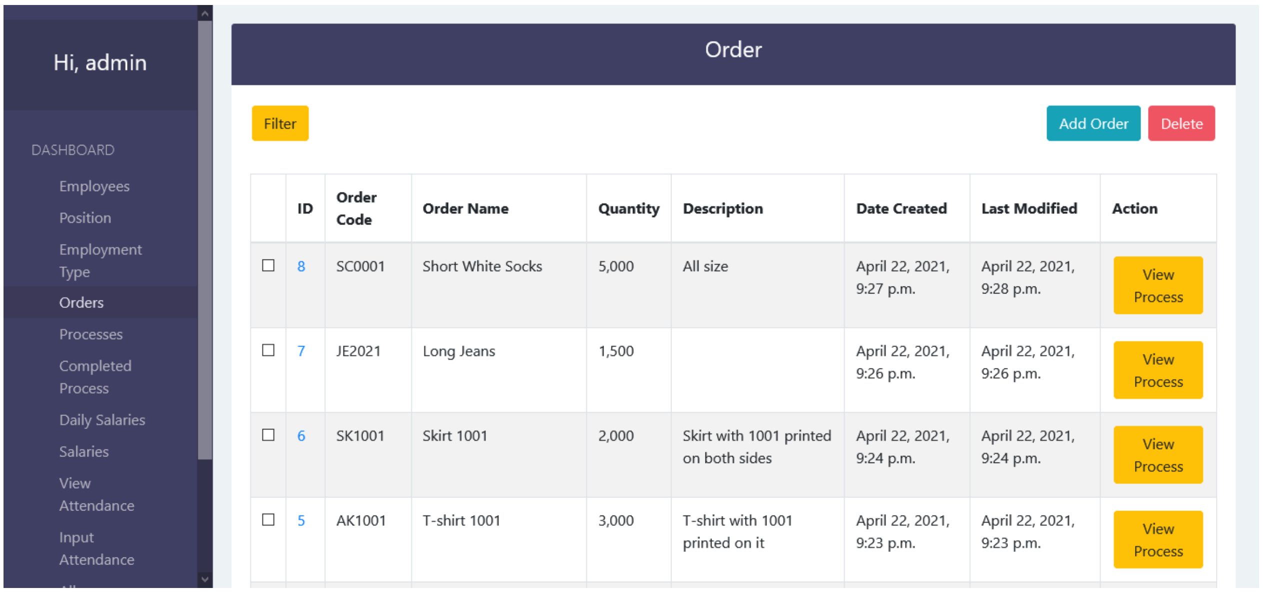
Edit Employment Type

Name: Piece work pay

Description: Paid based on the piece rate payment

Update

Figure 5.4.15 Edit employment type page



Order

	ID	Order Code	Order Name	Quantity	Description	Date Created	Last Modified	Action
☐	8	SC0001	Short White Socks	5,000	All size	April 22, 2021, 9:27 p.m.	April 22, 2021, 9:28 p.m.	View Process
☐	7	JE2021	Long Jeans	1,500		April 22, 2021, 9:26 p.m.	April 22, 2021, 9:26 p.m.	View Process
☐	6	SK1001	Skirt 1001	2,000	Skirt with 1001 printed on both sides	April 22, 2021, 9:24 p.m.	April 22, 2021, 9:24 p.m.	View Process
☐	5	AK1001	T-shirt 1001	3,000	T-shirt with 1001 printed on it	April 22, 2021, 9:23 p.m.	April 22, 2021, 9:23 p.m.	View Process

Figure 5.4.16 View order page

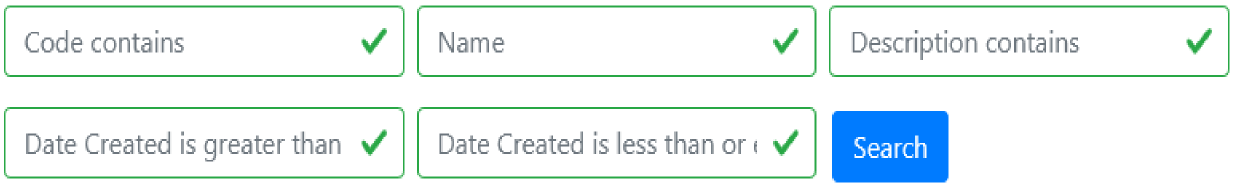
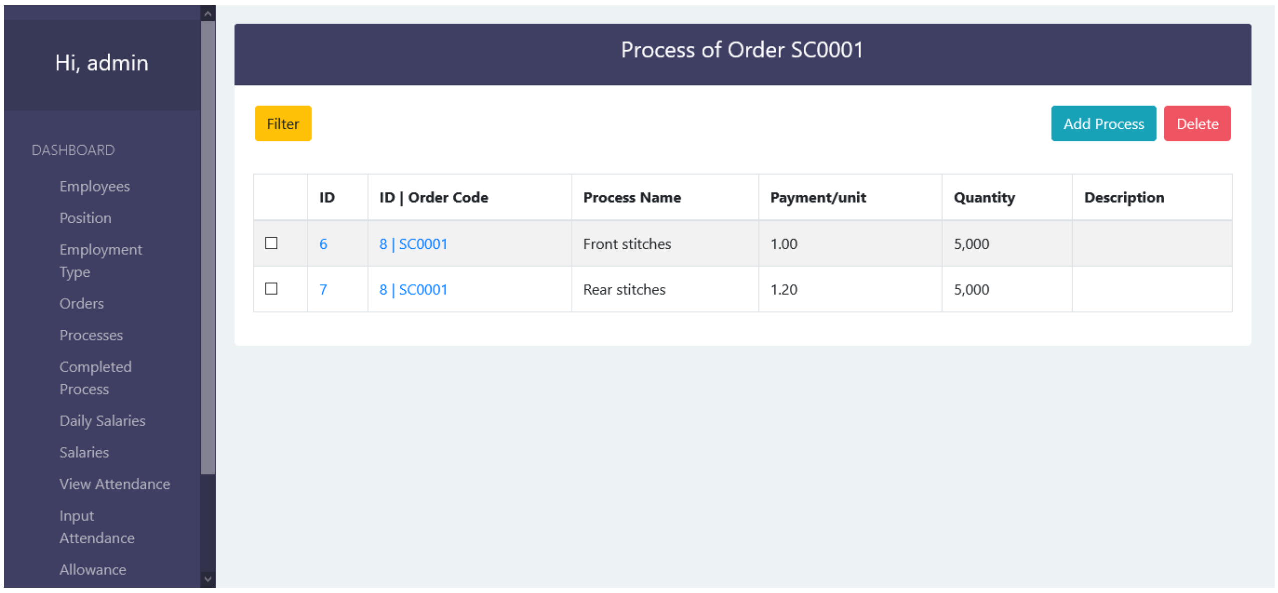


Figure 5.4.17 Order filter part


Hi, admin

DASHBOARD

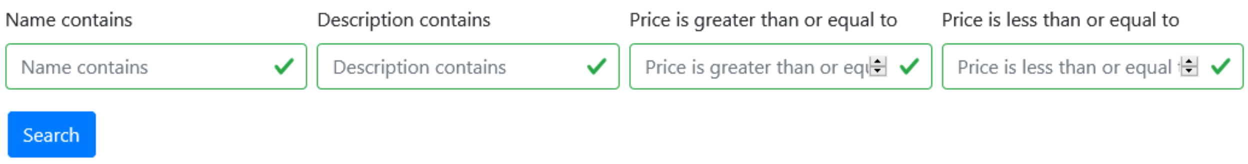
- Employees
- Position
- Employment Type
- Orders
- Processes
- Completed Process
- Daily Salaries
- Salaries
- View Attendance
- Input Attendance
- Allowance

Process of Order SC0001

Filter · Add Process · Delete

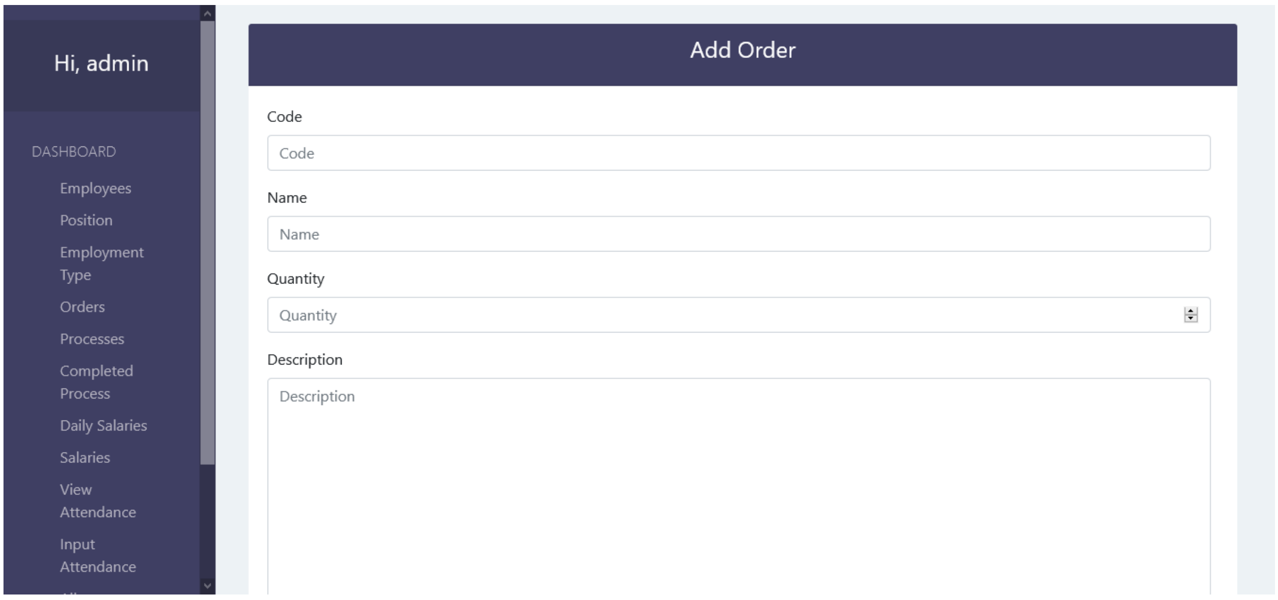
	ID	ID | Order Code	Process Name	Payment/unit	Quantity	Description
☐	6	8 | SC0001	Front stitches	1.00	5,000	
☐	7	8 | SC0001	Rear stitches	1.20	5,000	

Figure 5.4.18 View process of order page


Name contains	Description contains	Price is greater than or equal to	Price is less than or equal to
Name contains ✓	Description contains ✓	Price is greater than or eq ✓	Price is less than or equal ✓

Search

Figure 5.4.19 Process of order filter part


Hi, admin

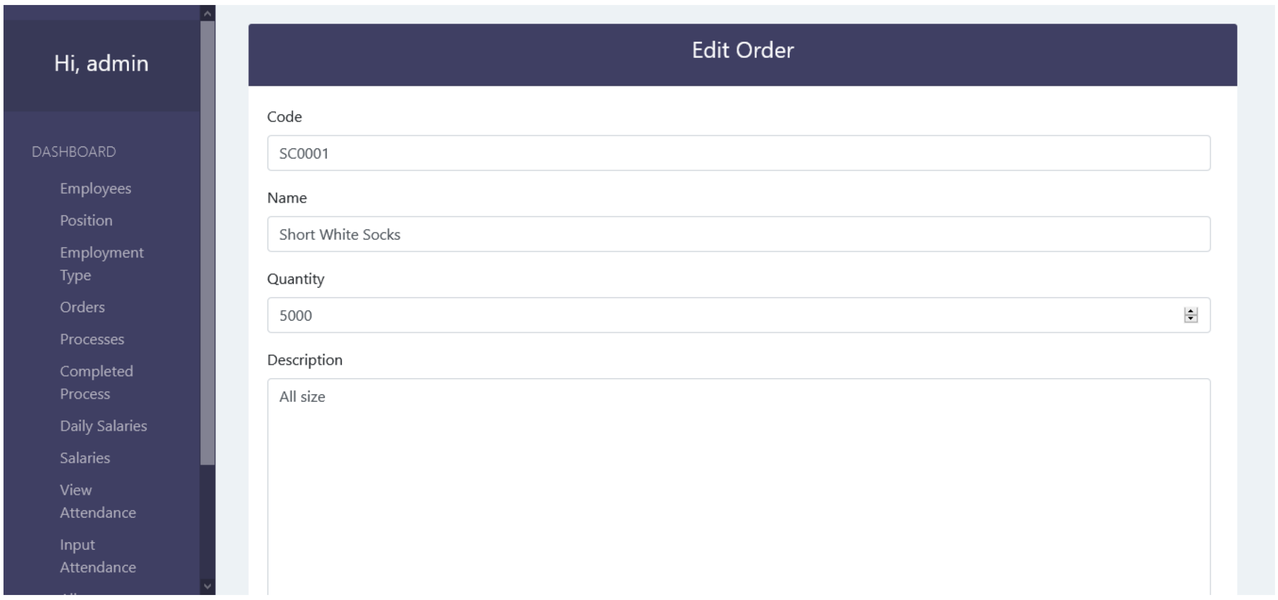
DASHBOARD

- Employees
- Position
- Employment Type
- Orders
- Processes
- Completed Process
- Daily Salaries
- Salaries
- View Attendance
- Input Attendance

Add Order

Code	Code
Name	Name
Quantity	Quantity
Description	Description

Figure 5.4.20 Add order page



Edit Order

Code

SC0001

Name

Short White Socks

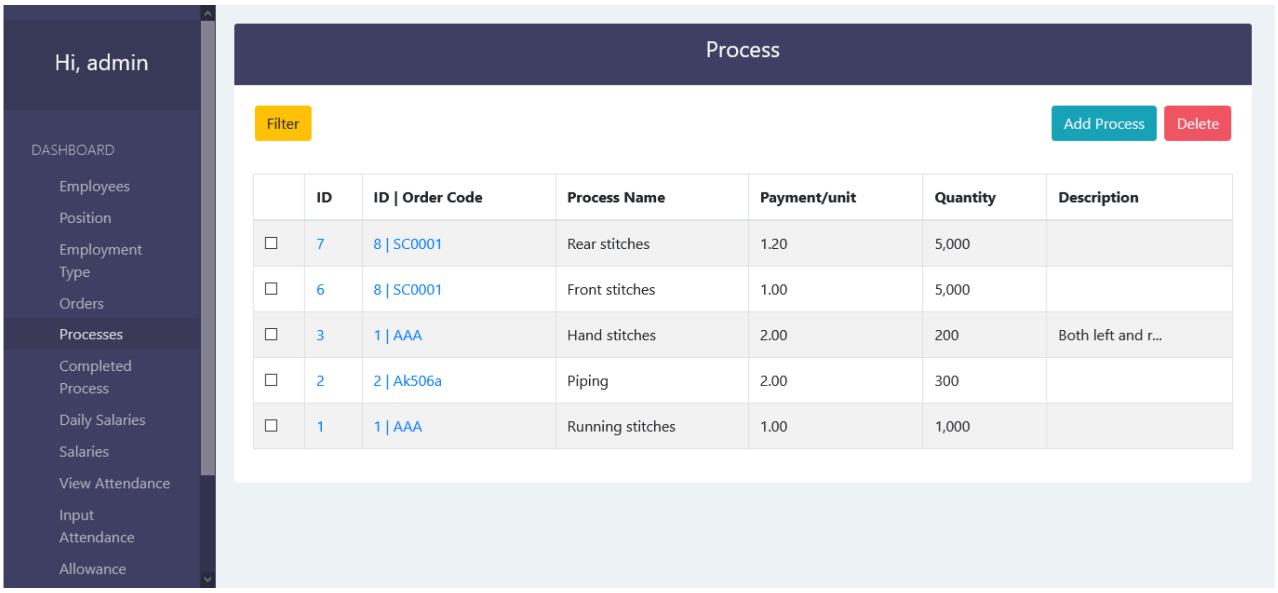
Quantity

5000

Description

All size

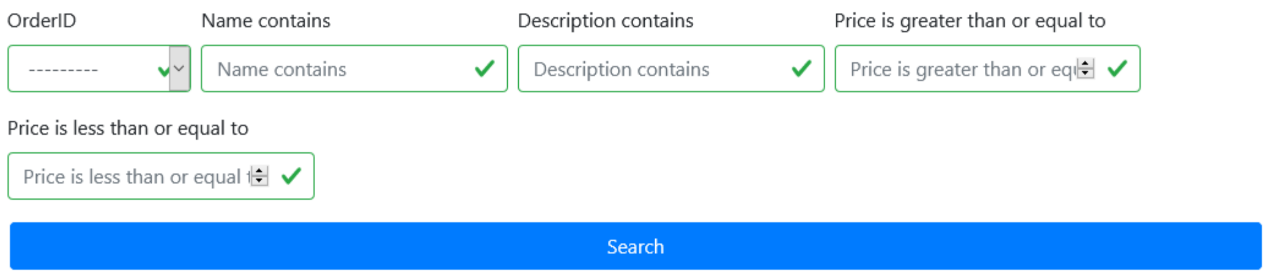
Figure 5.4.21 Edit order page



Process

	ID	ID | Order Code	Process Name	Payment/unit	Quantity	Description
☐	7	8 | SC0001	Rear stitches	1.20	5,000	
☐	6	8 | SC0001	Front stitches	1.00	5,000	
☐	3	1 | AAA	Hand stitches	2.00	200	Both left and r...
☐	2	2 | Ak506a	Piping	2.00	300	
☐	1	1 | AAA	Running stitches	1.00	1,000	

Figure 5.4.22 View process page



OrderID: ---------

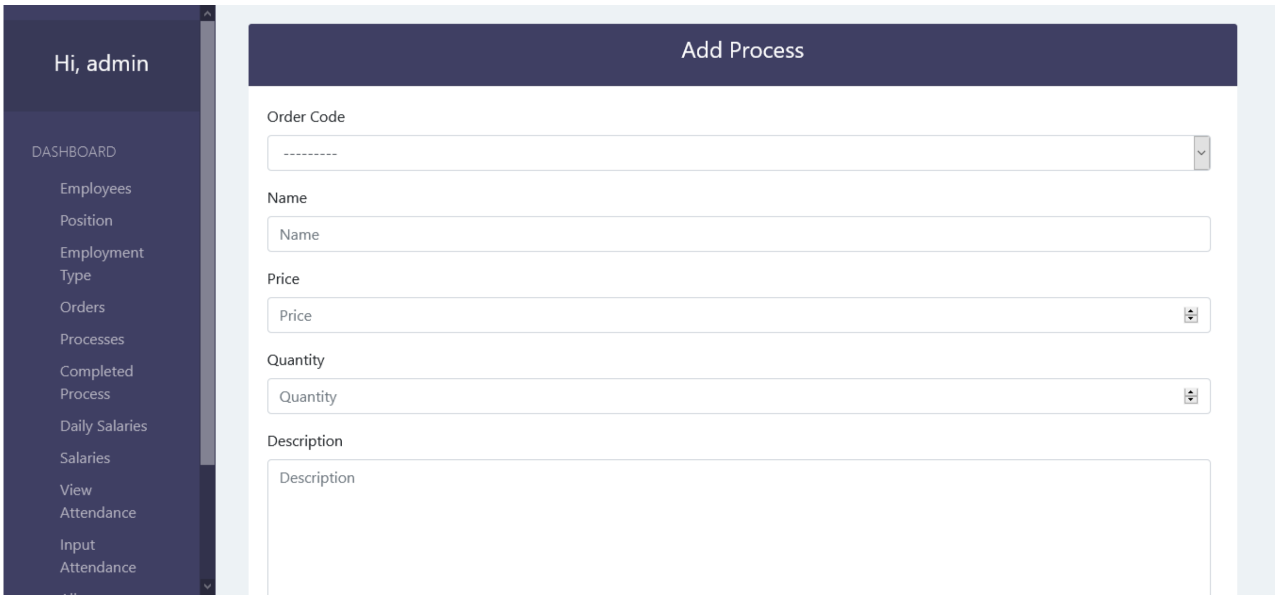
Name contains: Name contains

Description contains: Description contains

Price is greater than or equal to: Price is greater than or eq

Price is less than or equal to: Price is less than or equal

Figure 5.4.23 Process filter part



Add Process

Order Code: ---------

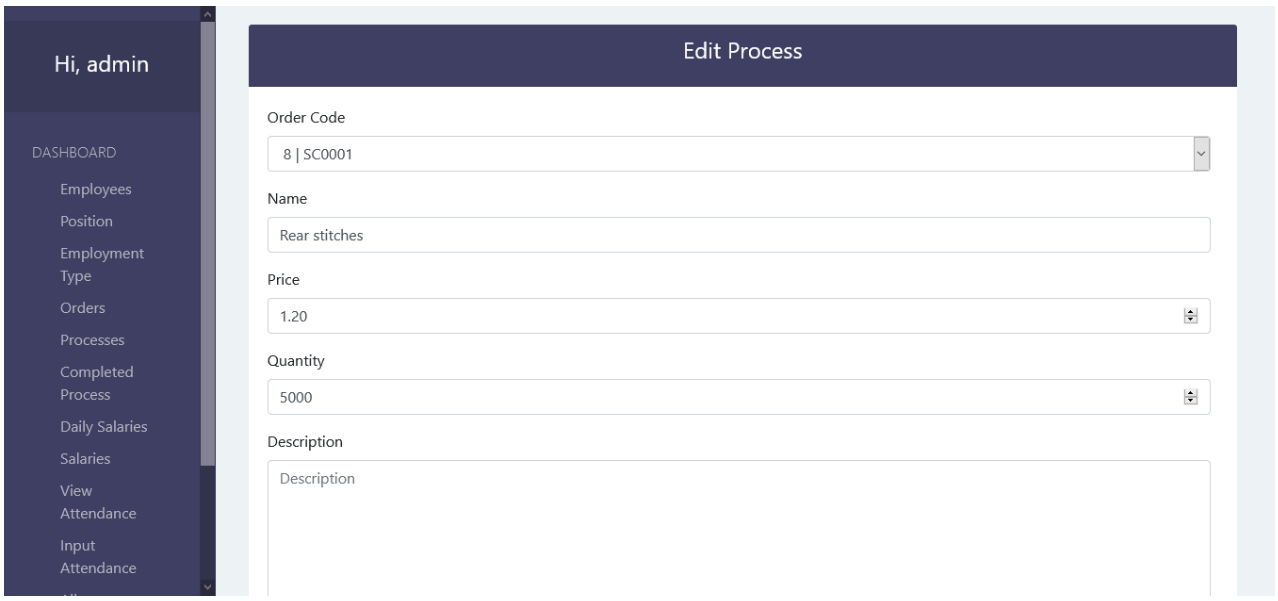
Name: Name

Price: Price

Quantity: Quantity

Description: Description

Figure 5.4.24 Add process page



Edit Process

Order Code: 8 | SC0001

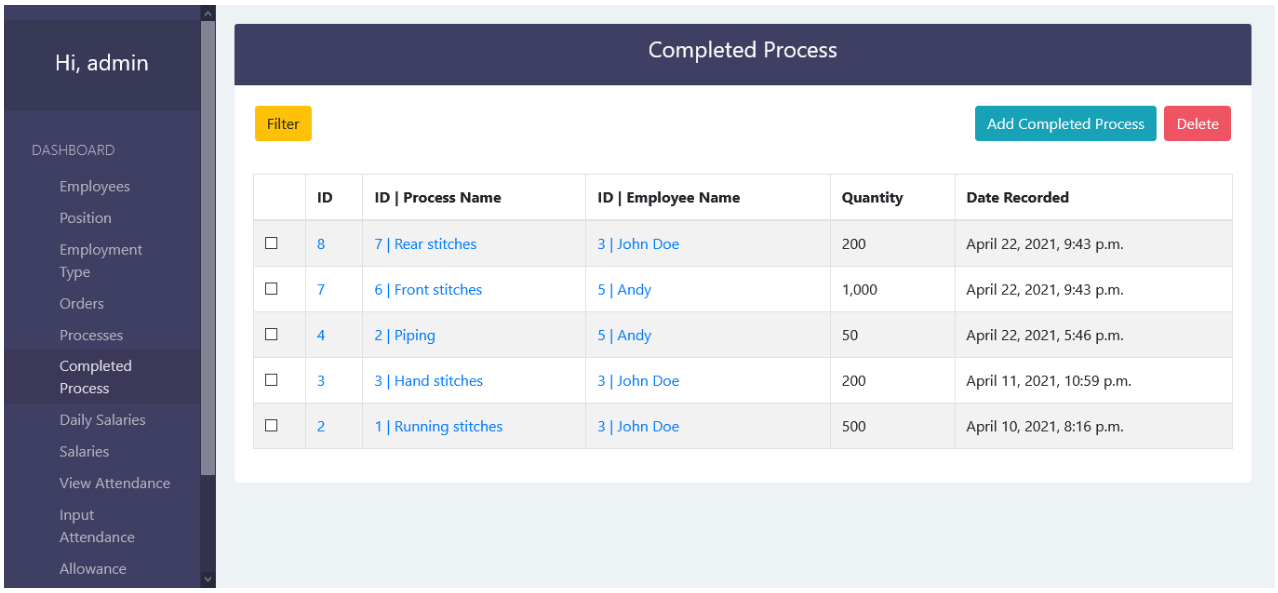
Name: Rear stitches

Price: 1.20

Quantity: 5000

Description: Description

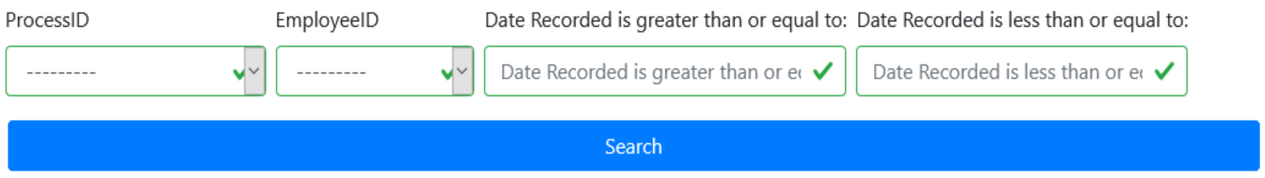
Figure 5.4.25 Edit process page



Completed Process

	ID	ID | Process Name	ID | Employee Name	Quantity	Date Recorded
☐	8	7 | Rear stitches	3 | John Doe	200	April 22, 2021, 9:43 p.m.
☐	7	6 | Front stitches	5 | Andy	1,000	April 22, 2021, 9:43 p.m.
☐	4	2 | Piping	5 | Andy	50	April 22, 2021, 5:46 p.m.
☐	3	3 | Hand stitches	3 | John Doe	200	April 11, 2021, 10:59 p.m.
☐	2	1 | Running stitches	3 | John Doe	500	April 10, 2021, 8:16 p.m.

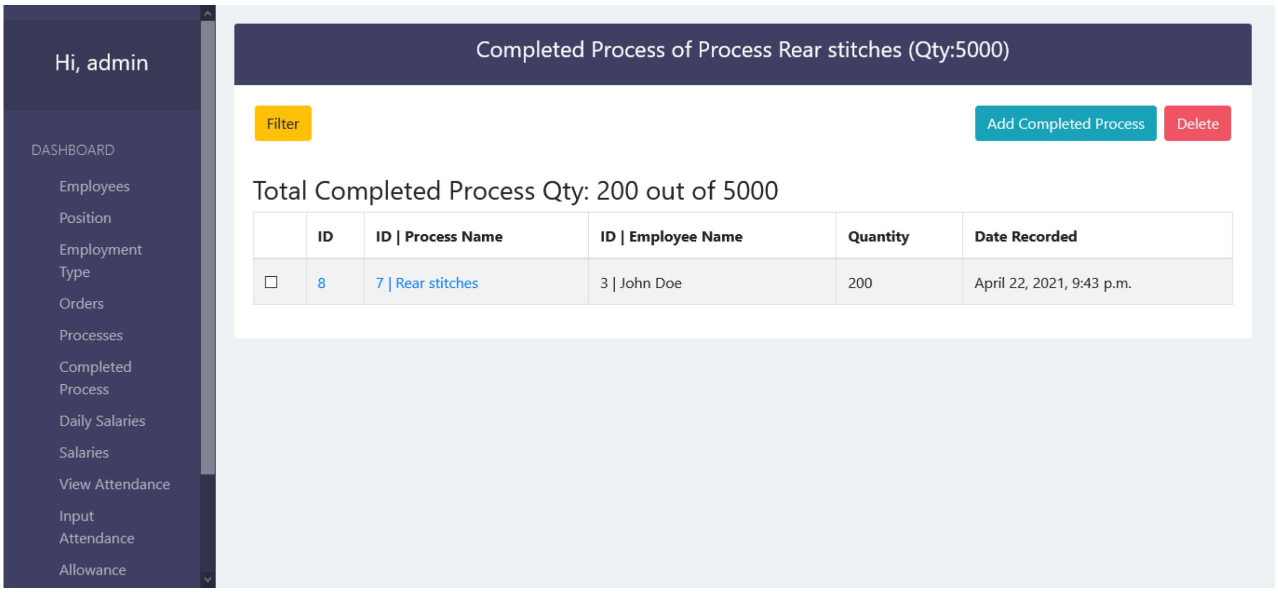
Figure 5.4.26 View completed process page


ProcessID

EmployeeID

Date Recorded is greater than or equal to:
Date Recorded is greater than or e
Date Recorded is less than or equal to:
Date Recorded is less than or e
Search

Figure 5.4.27 Completed process filter part

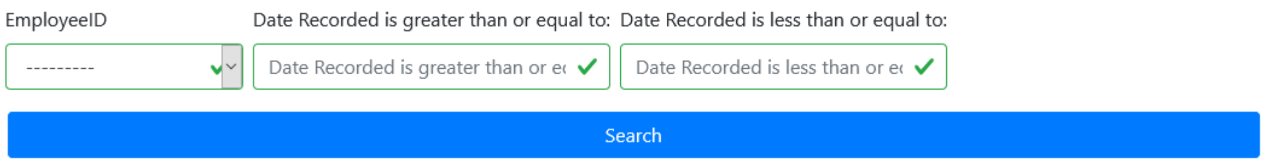


Completed Process of Process Rear stitches (Qty:5000)

Total Completed Process Qty: 200 out of 5000

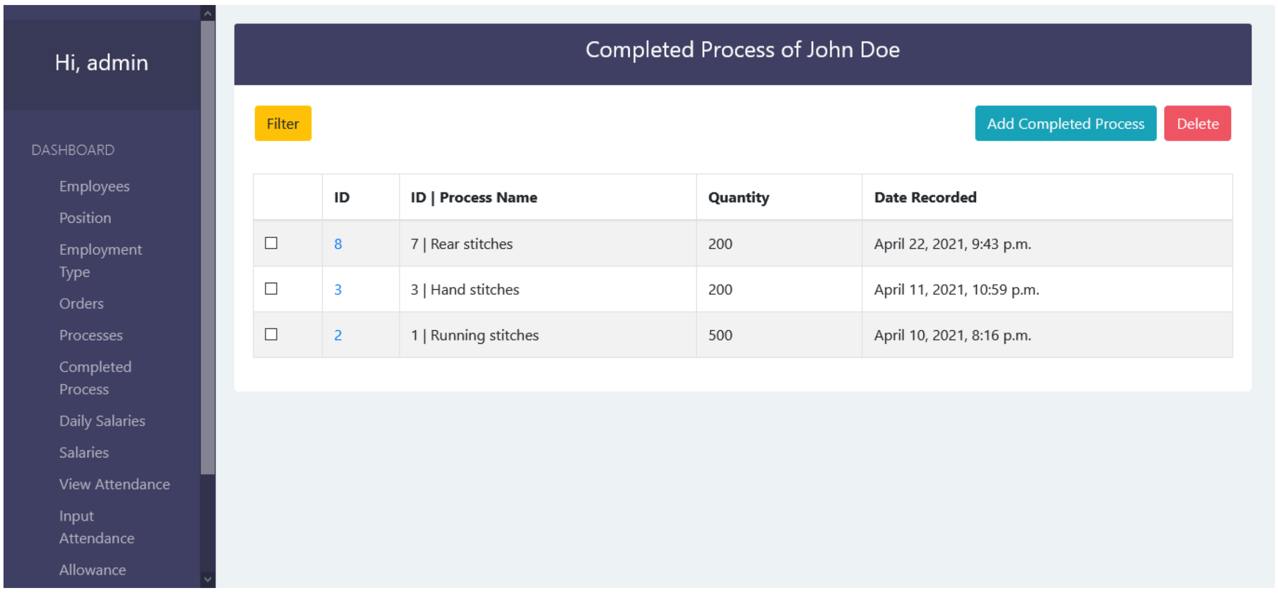
	ID	ID | Process Name	ID | Employee Name	Quantity	Date Recorded
☐	8	7 | Rear stitches	3 | John Doe	200	April 22, 2021, 9:43 p.m.

Figure 5.4.28 View completed process of a particular process page


EmployeeID

Date Recorded is greater than or equal to:
Date Recorded is greater than or e
Date Recorded is less than or equal to:
Date Recorded is less than or e
Search

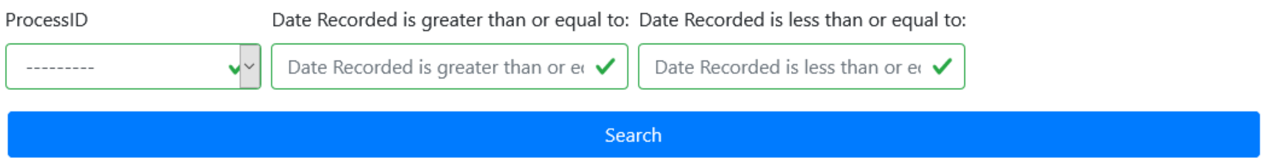
Figure 5.4.29 Completed process of a particular process filter part



Completed Process of John Doe

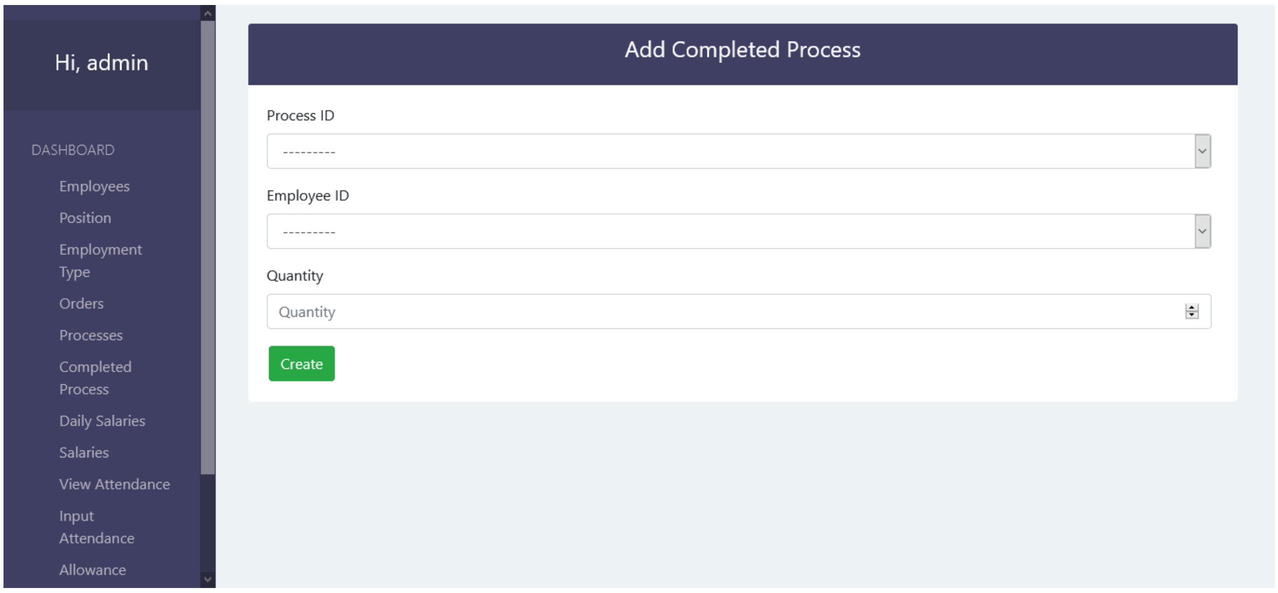
	ID	ID | Process Name	Quantity	Date Recorded
☐	8	7 | Rear stitches	200	April 22, 2021, 9:43 p.m.
☐	3	3 | Hand stitches	200	April 11, 2021, 10:59 p.m.
☐	2	1 | Running stitches	500	April 10, 2021, 8:16 p.m.

Figure 5.4.30 View completed process of an employee page



ProcessID	Date Recorded is greater than or equal to:	Date Recorded is less than or equal to:
---------	Date Recorded is greater than or e	Date Recorded is less than or e

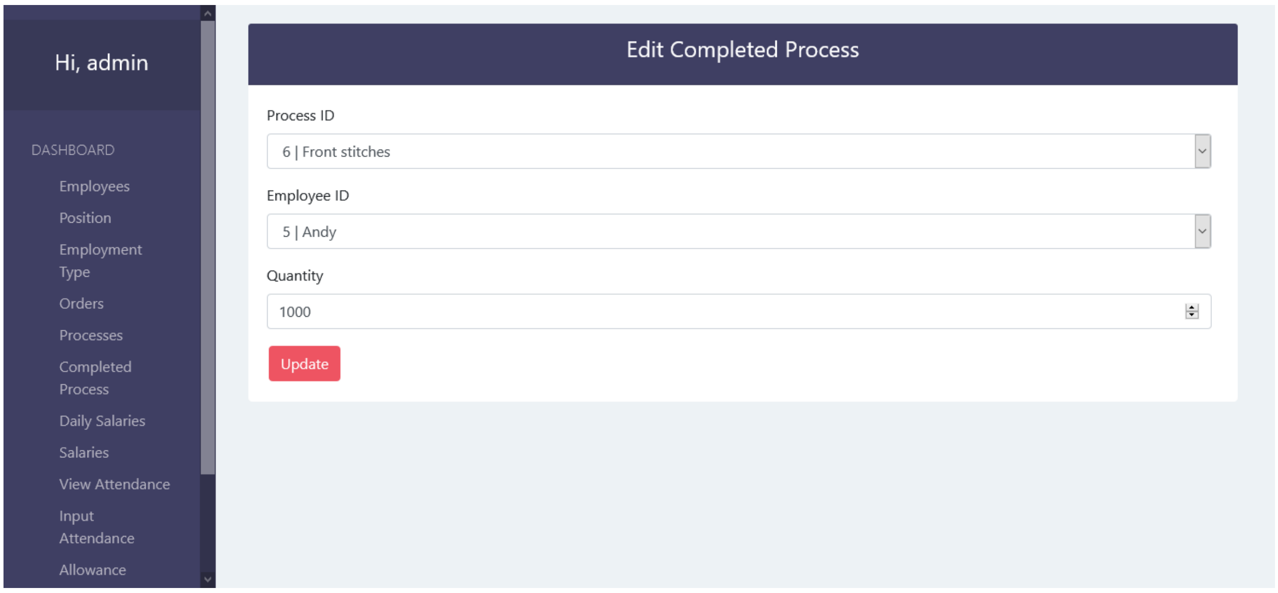
Figure 5.4.31 Completed process of an employee filter part



Add Completed Process

Process ID	---------
Employee ID	---------
Quantity	Quantity

Figure 5.4.32 Add completed process page



Edit Completed Process

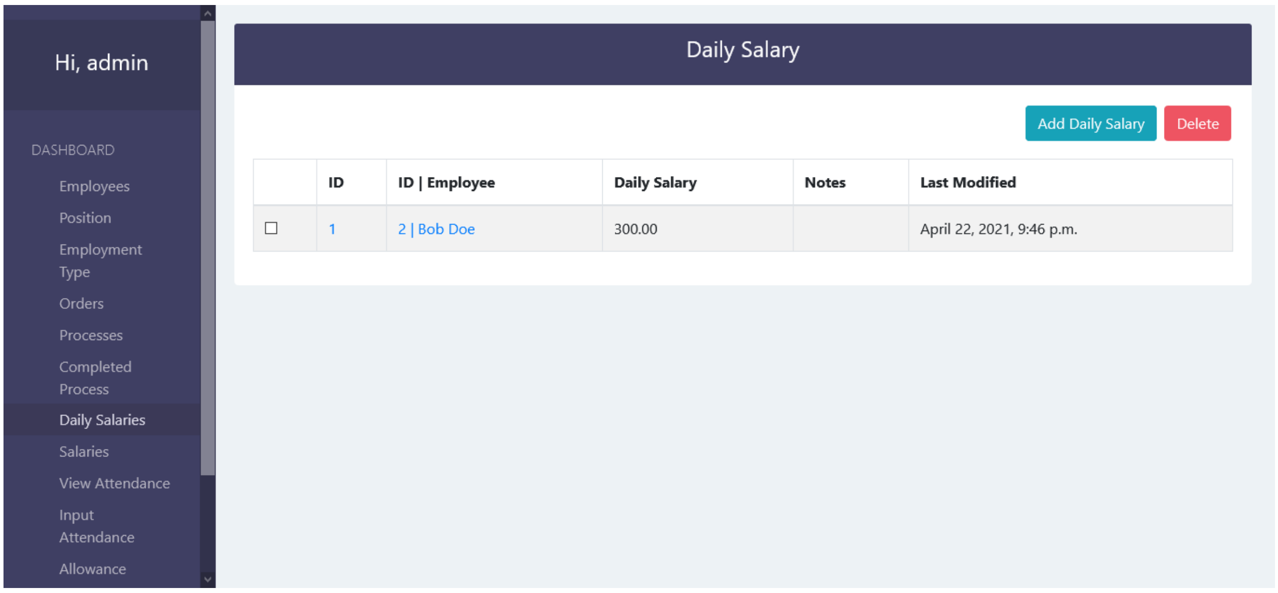
Process ID: 6 | Front stitches

Employee ID: 5 | Andy

Quantity: 1000

Update

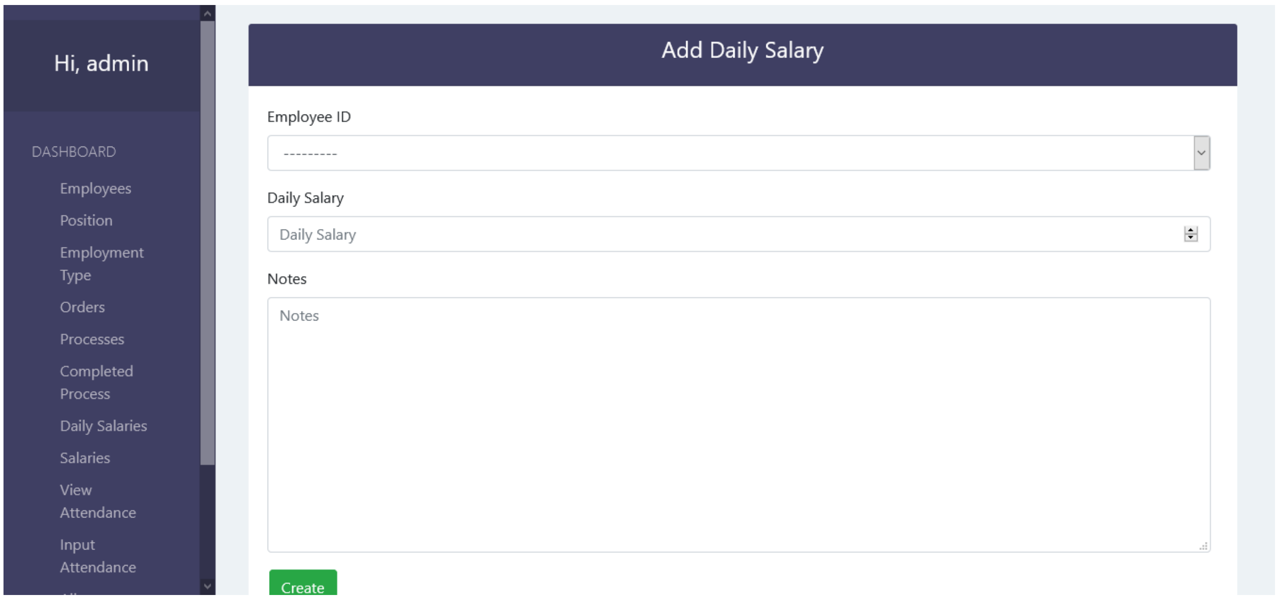
Figure 5.4.33 Edit completed process page



Daily Salary

	ID	ID | Employee	Daily Salary	Notes	Last Modified
☐	1	2 | Bob Doe	300.00		April 22, 2021, 9:46 p.m.

Figure 5.4.34 View daily salary page



Add Daily Salary

Employee ID

Daily Salary

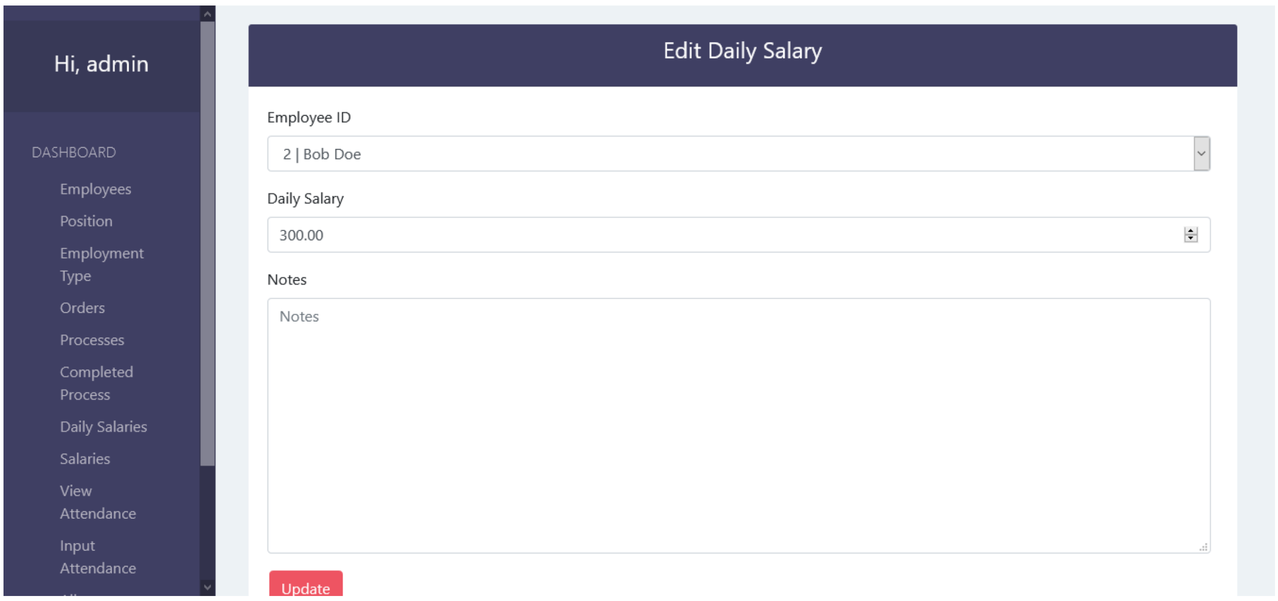
Daily Salary

Notes

Notes

Create

Figure 5.4.35 Add daily salary page



Edit Daily Salary

Employee ID

2 | Bob Doe

Daily Salary

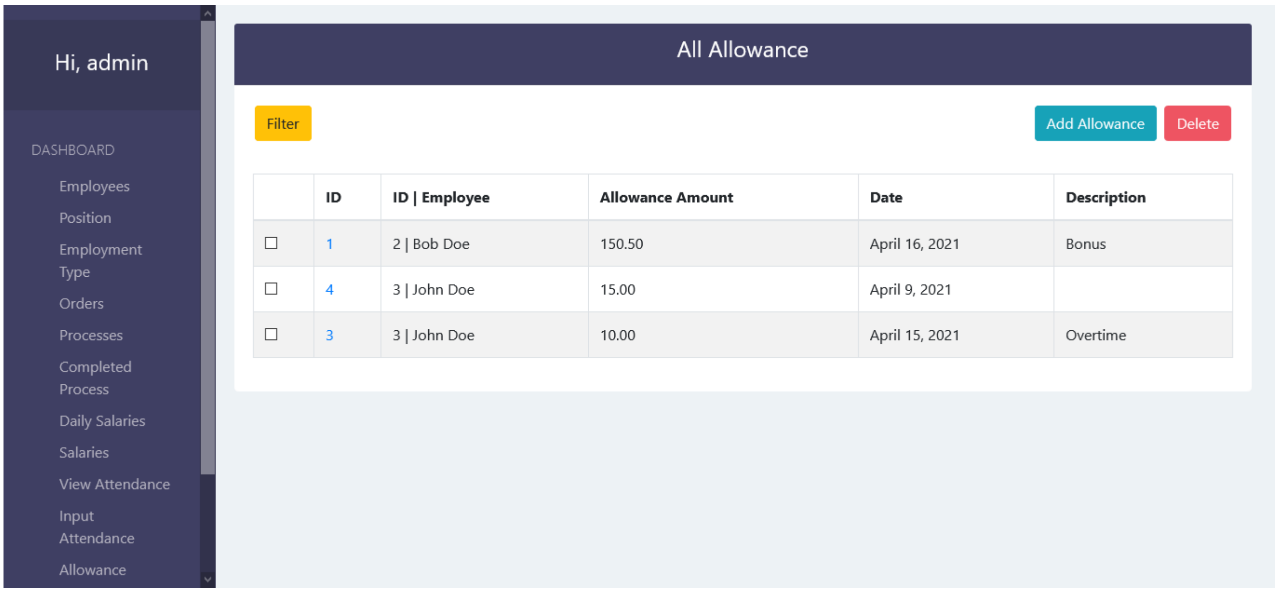
300.00

Notes

Notes

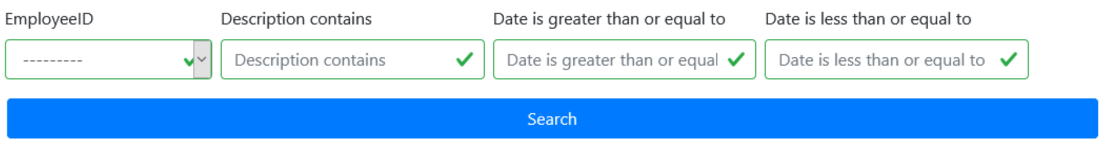
Update

Figure 5.4.36 Edit daily salary page



	ID	ID | Employee	Allowance Amount	Date	Description
☐	1	2 | Bob Doe	150.50	April 16, 2021	Bonus
☐	4	3 | John Doe	15.00	April 9, 2021	
☐	3	3 | John Doe	10.00	April 15, 2021	Overtime

Figure 5.4.37 View allowance page


EmployeeID

Description contains
Description contains
Date is greater than or equal to
Date is greater than or equal
Date is less than or equal to
Date is less than or equal to
Search

Figure 5.4.38 Allowance filter part

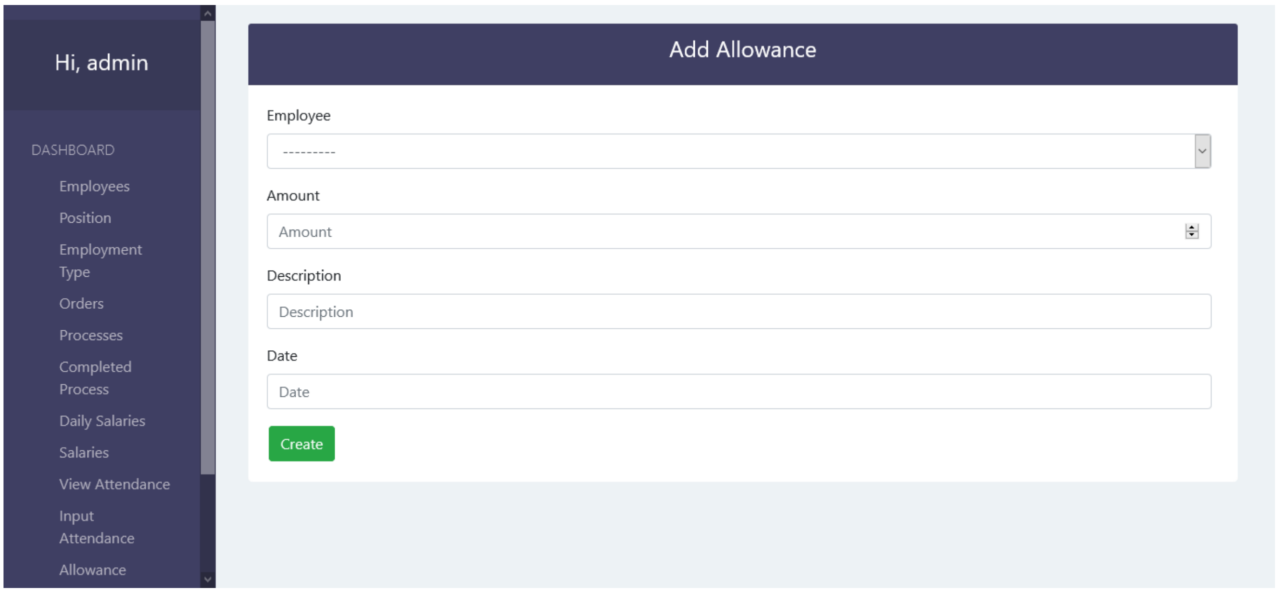
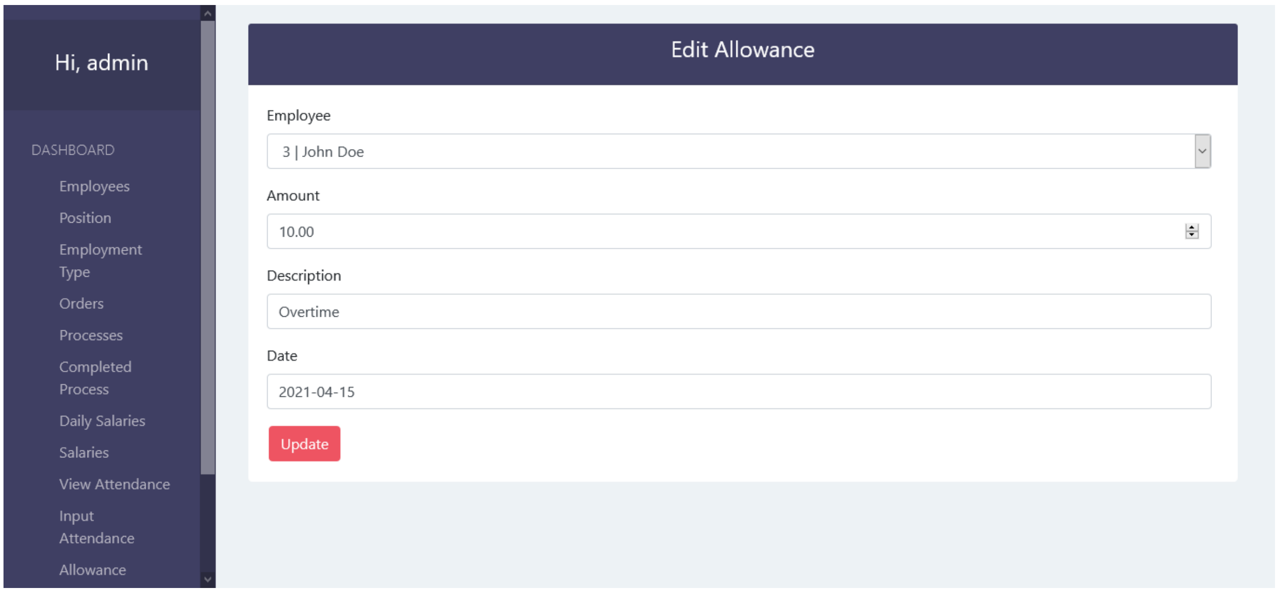


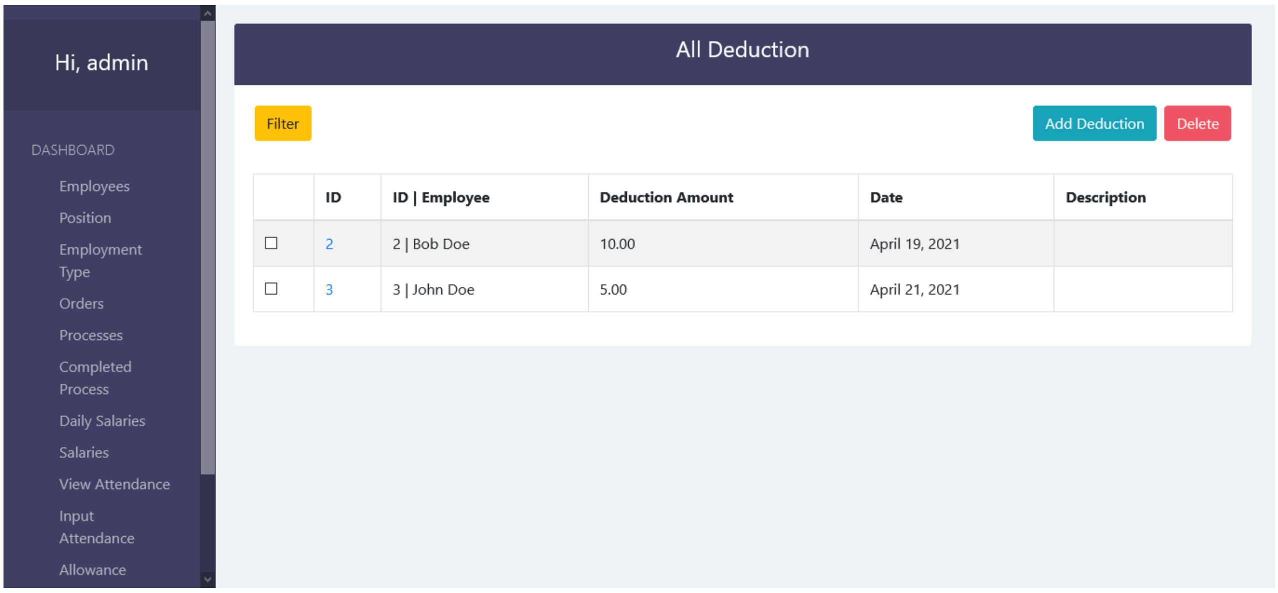
Figure 5.4.39 Add allowance page



Edit Allowance

Employee	3 | John Doe
Amount	10.00
Description	Overtime
Date	2021-04-15

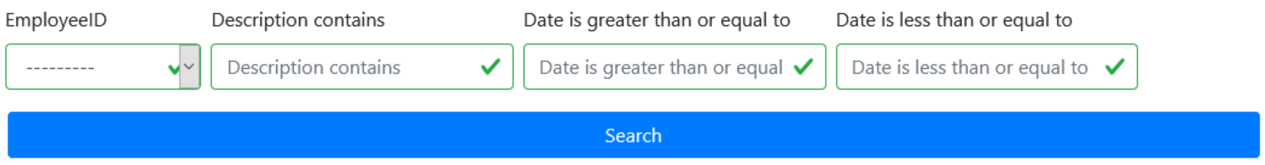
Figure 5.4.40 Edit allowance page



All Deduction

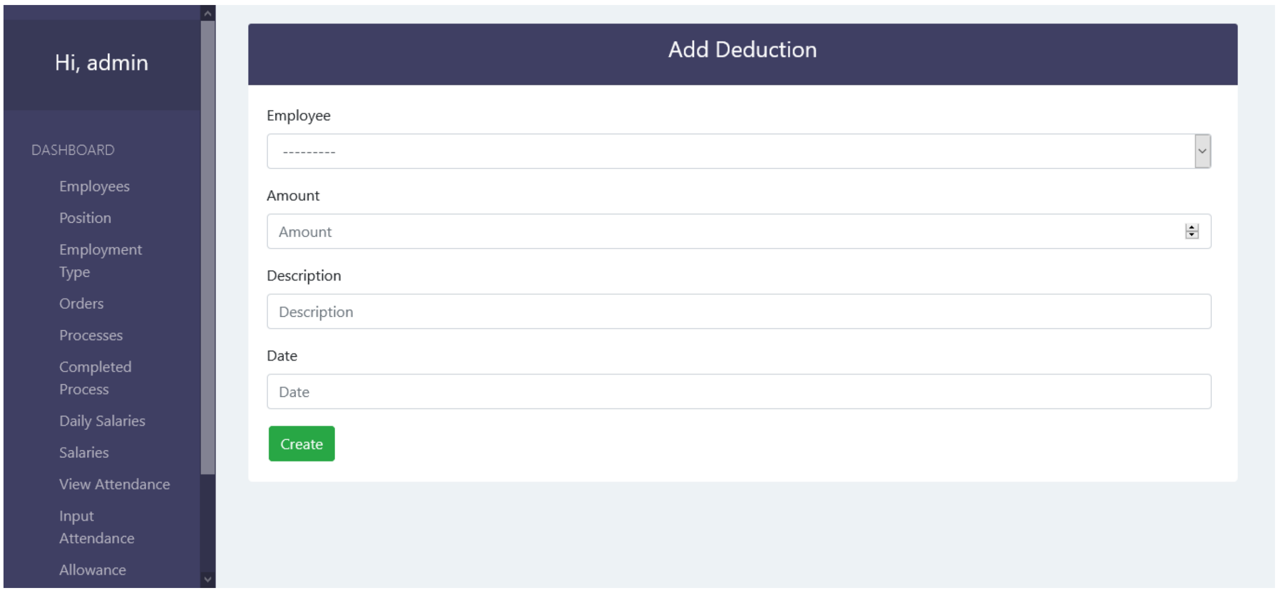
	ID	ID | Employee	Deduction Amount	Date	Description
☐	2	2 | Bob Doe	10.00	April 19, 2021	
☐	3	3 | John Doe	5.00	April 21, 2021	

Figure 5.4.41 View deduction page



EmployeeID	Description contains	Date is greater than or equal to	Date is less than or equal to
---------	Description contains	Date is greater than or equal	Date is less than or equal to

Figure 5.4.42 Deduction filter part



Add Deduction

Employee

Amount

Amount

Description

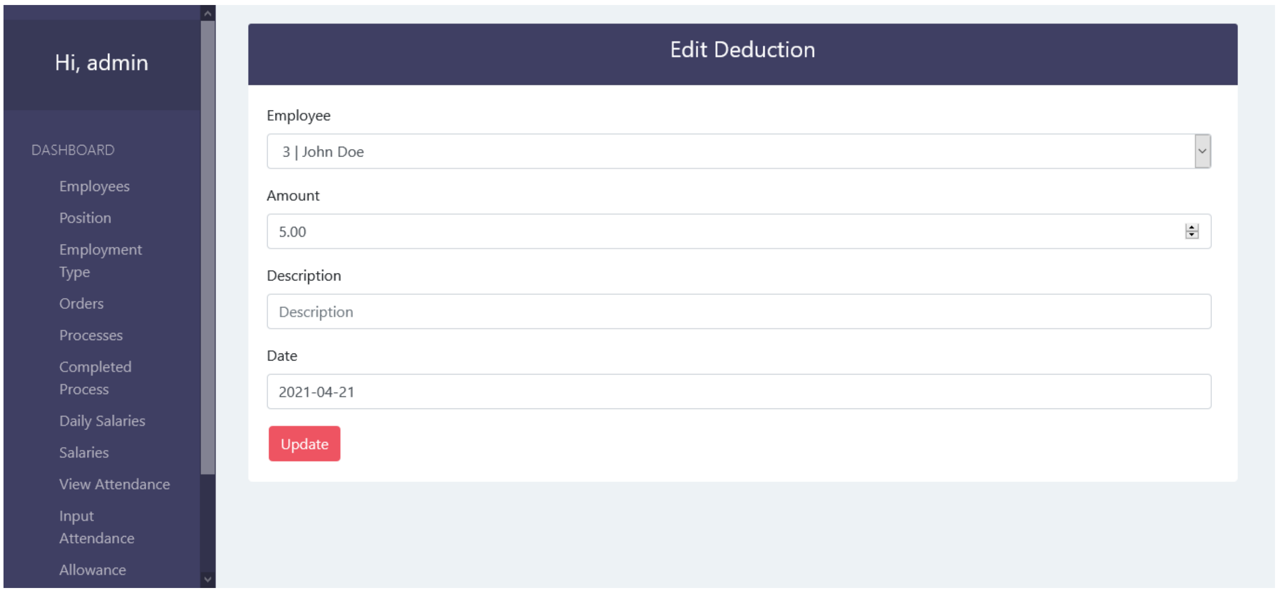
Description

Date

Date

Create

Figure 5.4.43 Add deduction page



Edit Deduction

Employee

3 | John Doe

Amount

5.00

Description

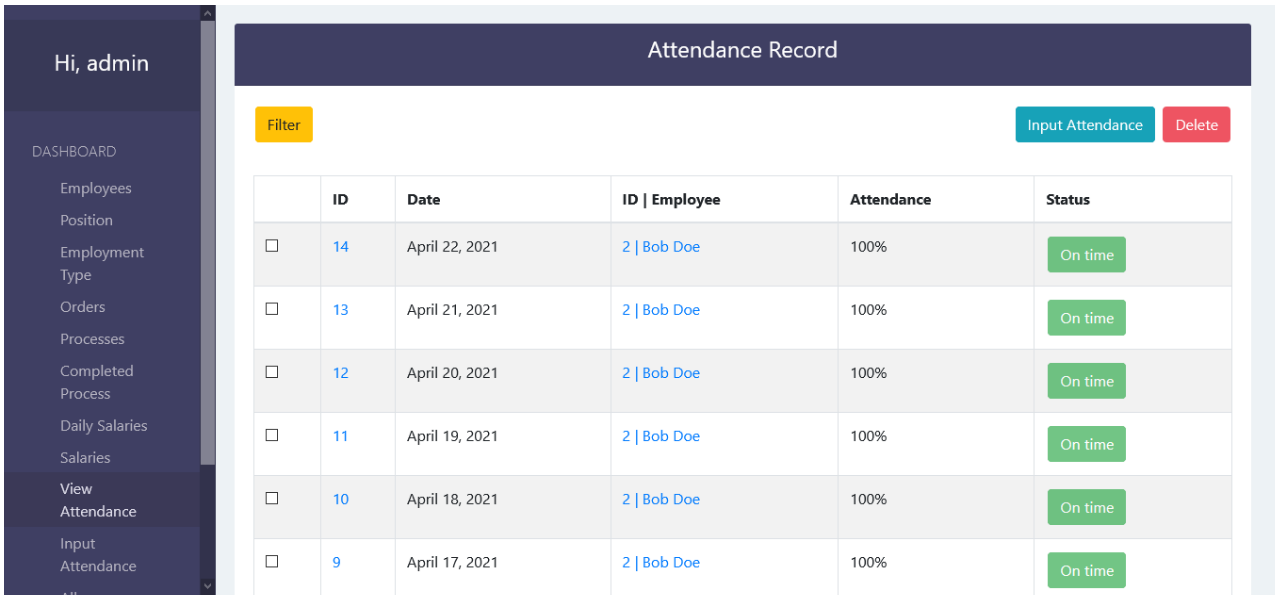
Description

Date

2021-04-21

Update

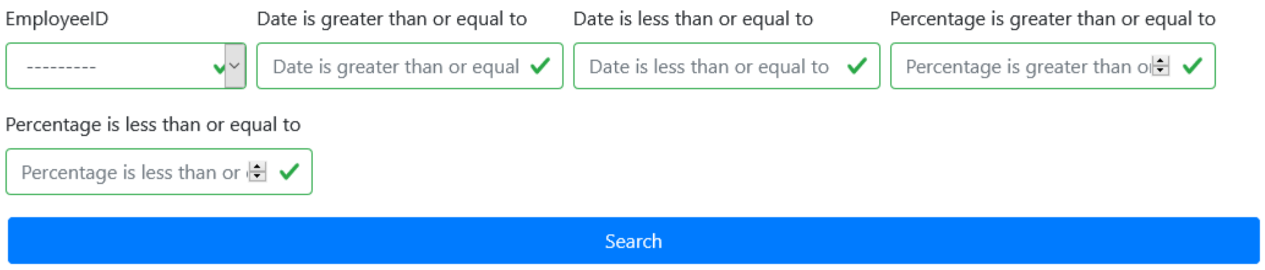
Figure 5.4.44 Edit deduction page



Attendance Record

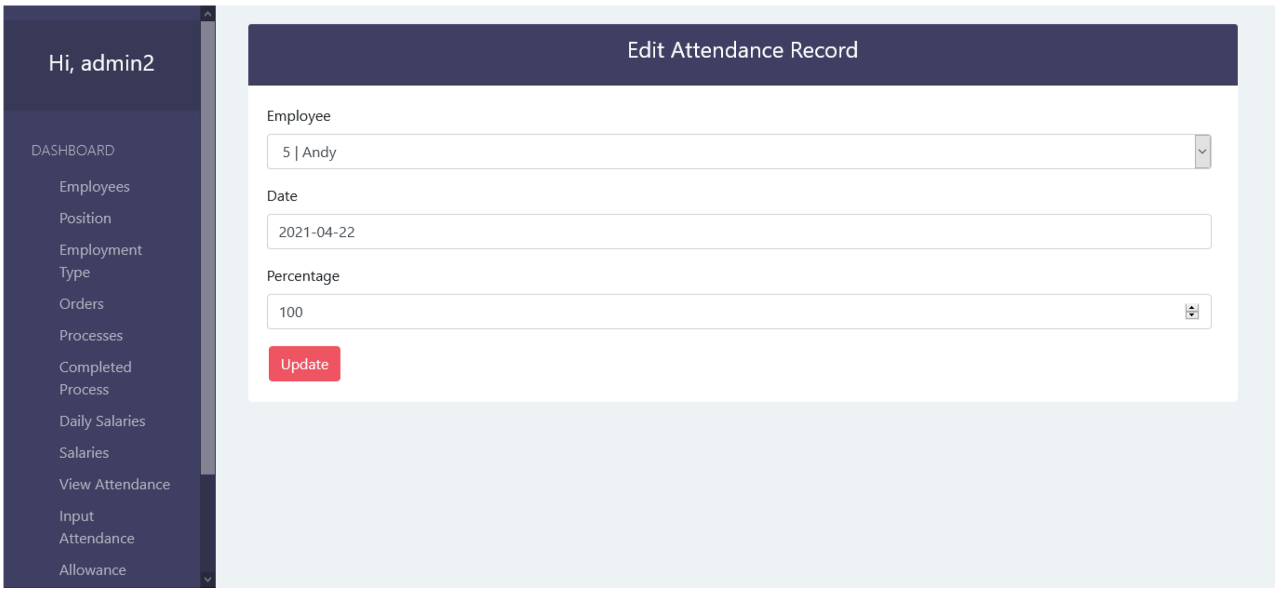
	ID	Date	ID | Employee	Attendance	Status
☐	14	April 22, 2021	2 | Bob Doe	100%	On time
☐	13	April 21, 2021	2 | Bob Doe	100%	On time
☐	12	April 20, 2021	2 | Bob Doe	100%	On time
☐	11	April 19, 2021	2 | Bob Doe	100%	On time
☐	10	April 18, 2021	2 | Bob Doe	100%	On time
☐	9	April 17, 2021	2 | Bob Doe	100%	On time

Figure 5.4.45 View attendance page



EmployeeID	---------
Date is greater than or equal to	Date is greater than or equal
Date is less than or equal to	Date is less than or equal to
Percentage is greater than or equal to	Percentage is greater than o
Percentage is less than or equal to	Percentage is less than or

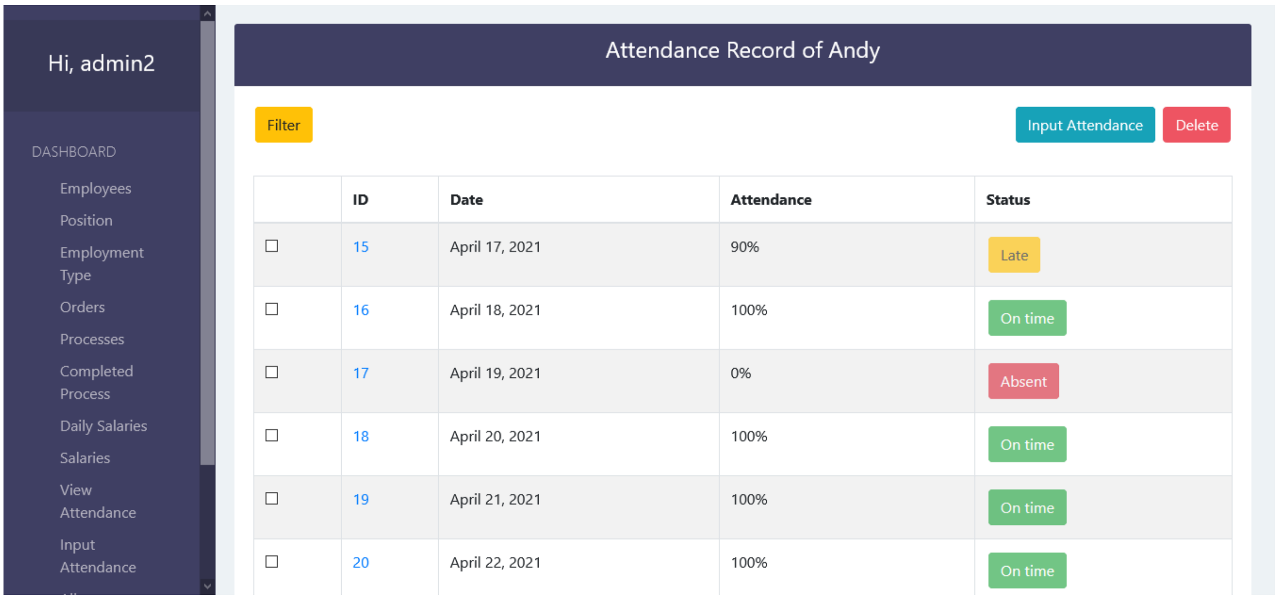
Figure 5.4.46 Attendance filter part



Edit Attendance Record

Employee	5 | Andy
Date	2021-04-22
Percentage	100

Figure 5.4.47 Edit attendance page



Attendance Record of Andy

	ID	Date	Attendance	Status
☐	15	April 17, 2021	90%	Late
☐	16	April 18, 2021	100%	On time
☐	17	April 19, 2021	0%	Absent
☐	18	April 20, 2021	100%	On time
☐	19	April 21, 2021	100%	On time
☐	20	April 22, 2021	100%	On time

Figure 5.4.48 View attendance of a particular employee page

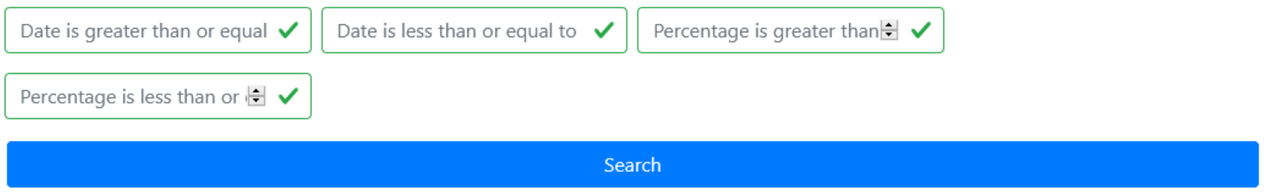
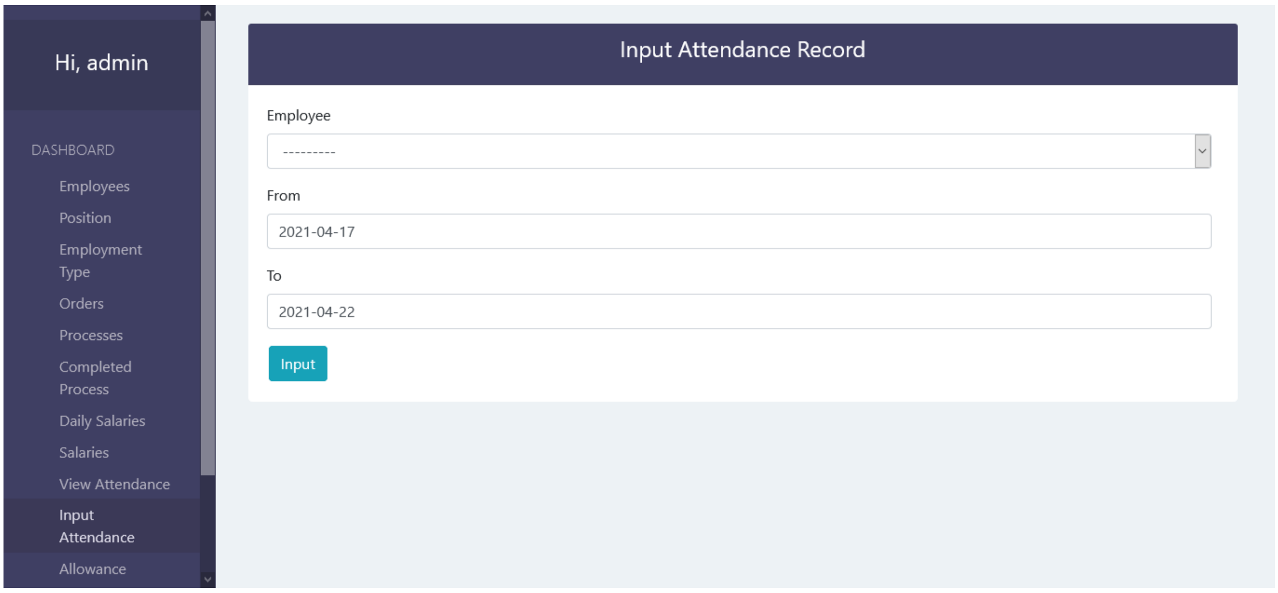

Date is greater than or equal
Date is less than or equal to
Percentage is greater than
Percentage is less than or
Search

Figure 5.4.49 Attendance of a particular employee filter part



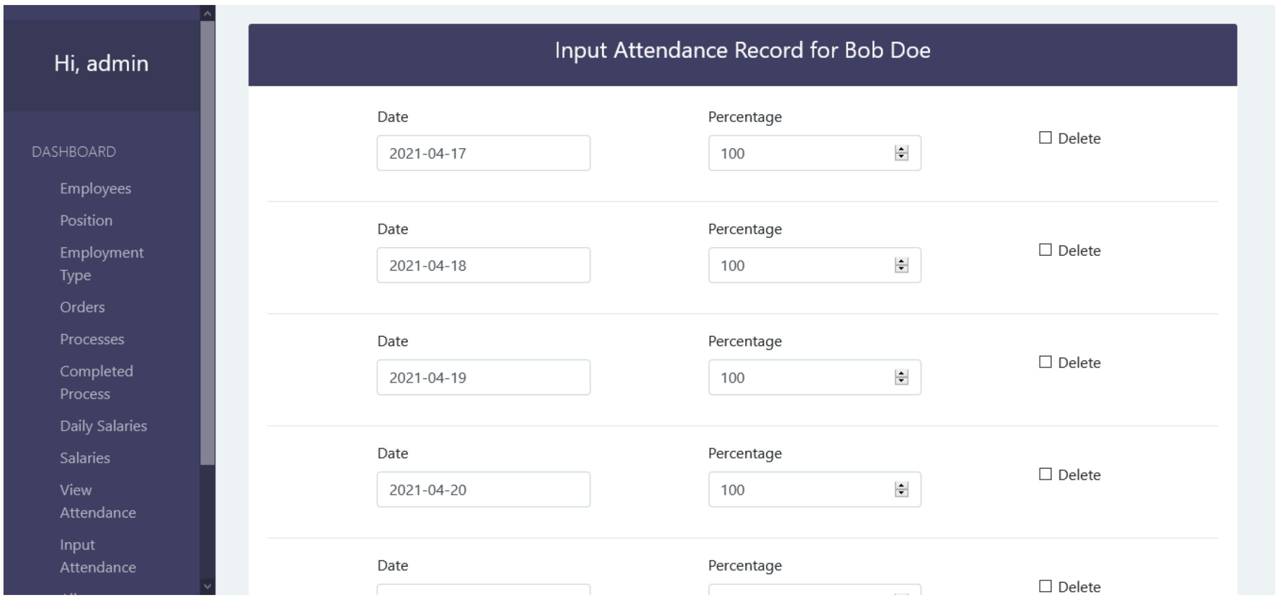
Input Attendance Record

Employee: ---------

From: 2021-04-17

To: 2021-04-22

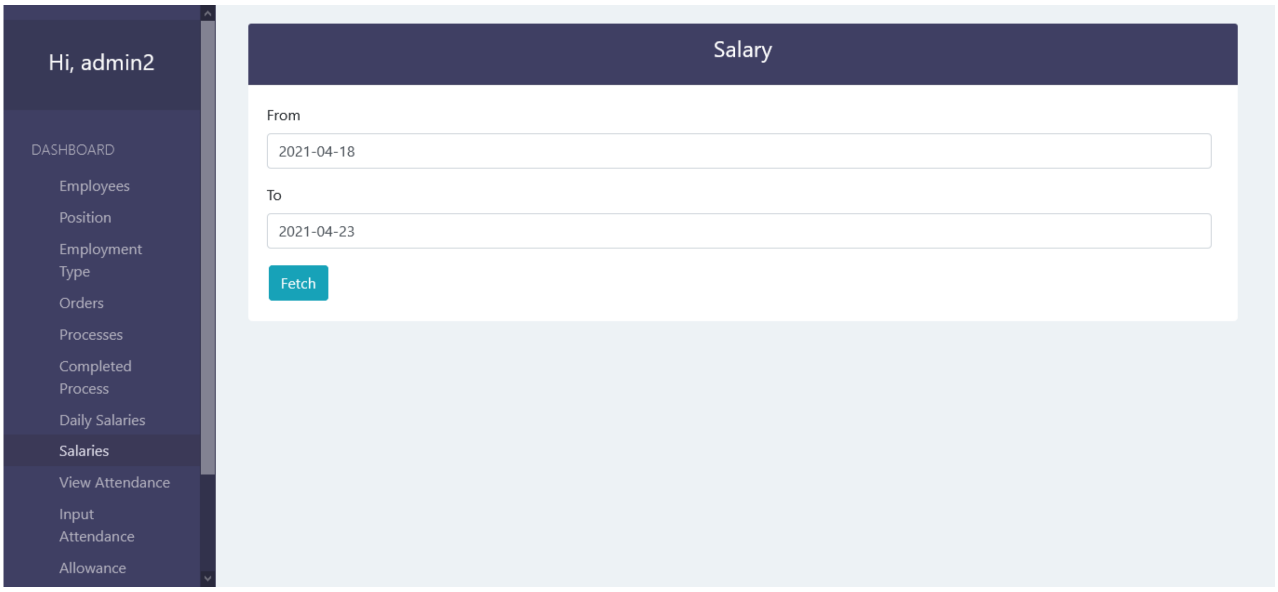
Figure 5.4.50 Input employee and attendance date page



Input Attendance Record for Bob Doe

Date	Percentage	
2021-04-17	100	☐ Delete
2021-04-18	100	☐ Delete
2021-04-19	100	☐ Delete
2021-04-20	100	☐ Delete
		☐ Delete

Figure 5.4.51 Add attendance page

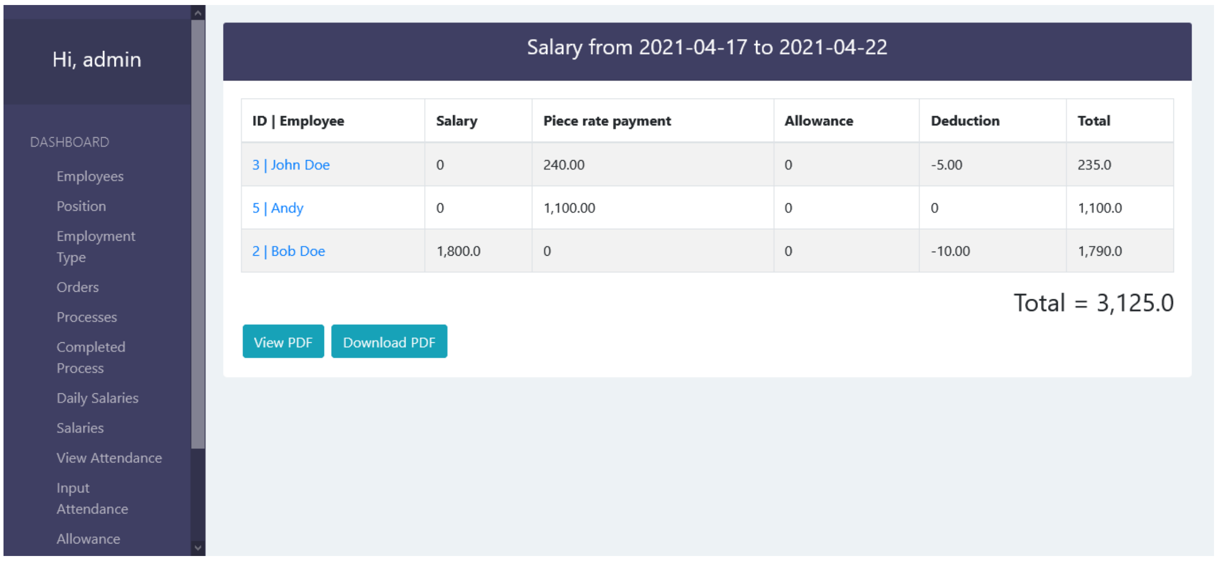


Salary

From: 2021-04-18

To: 2021-04-23

Figure 5.4.52 Input salary date range page

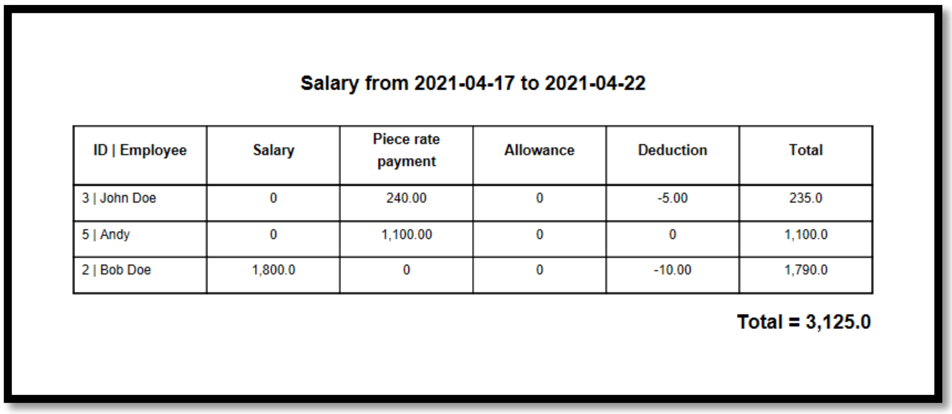


Salary from 2021-04-17 to 2021-04-22

ID | Employee	Salary	Piece rate payment	Allowance	Deduction	Total
3 | John Doe	0	240.00	0	-5.00	235.0
5 | Andy	0	1,100.00	0	0	1,100.0
2 | Bob Doe	1,800.0	0	0	-10.00	1,790.0

Total = 3,125.0

Figure 5.4.53 View salary page

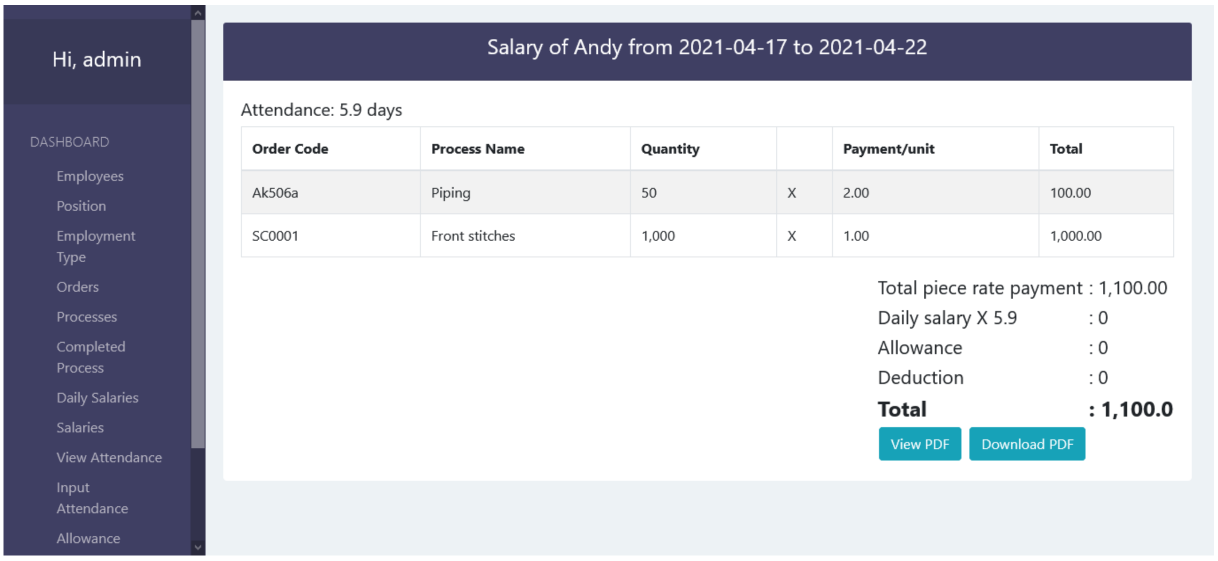


Salary from 2021-04-17 to 2021-04-22

ID | Employee	Salary	Piece rate payment	Allowance	Deduction	Total
3 | John Doe	0	240.00	0	-5.00	235.0
5 | Andy	0	1,100.00	0	0	1,100.0
2 | Bob Doe	1,800.0	0	0	-10.00	1,790.0

Total = 3,125.0

Figure 5.4.54 Salary PDF example



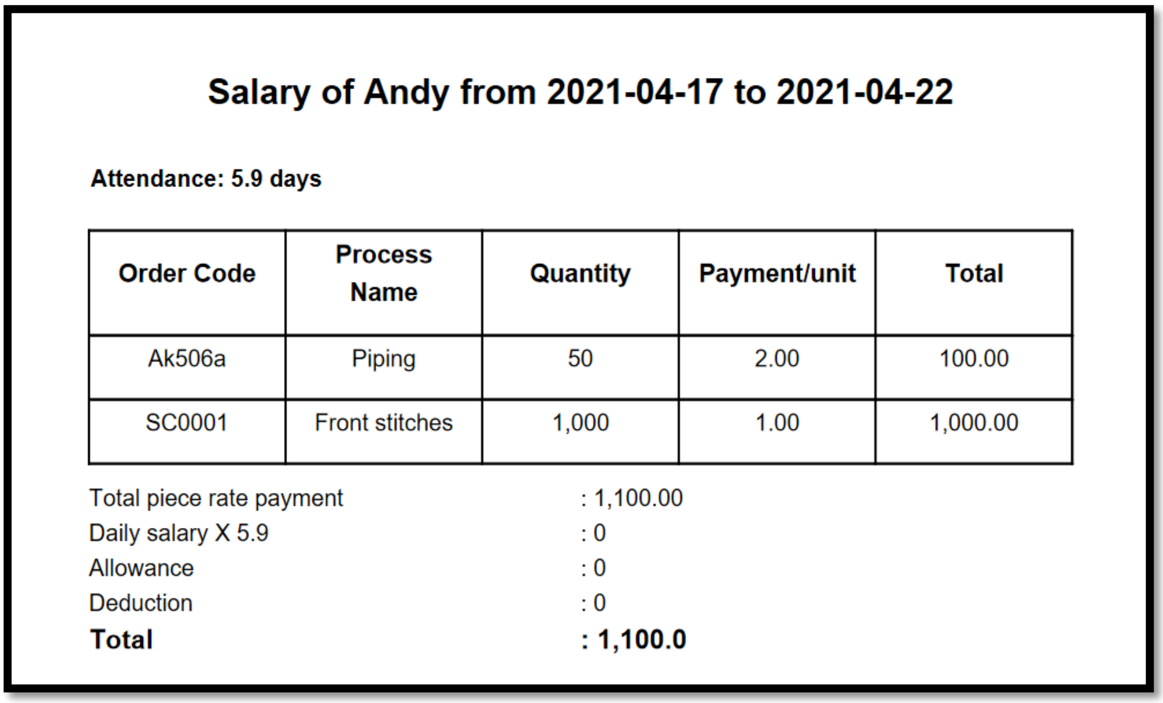
Salary of Andy from 2021-04-17 to 2021-04-22

Attendance: 5.9 days

Order Code	Process Name	Quantity		Payment/unit	Total
Ak506a	Piping	50	X	2.00	100.00
SC0001	Front stitches	1,000	X	1.00	1,000.00

Total piece rate payment	: 1,100.00
Daily salary X 5.9	: 0
Allowance	: 0
Deduction	: 0
Total	**: 1,100.0**

Figure 5.4.55 View salary details of an employee page



Salary of Andy from 2021-04-17 to 2021-04-22

Attendance: 5.9 days

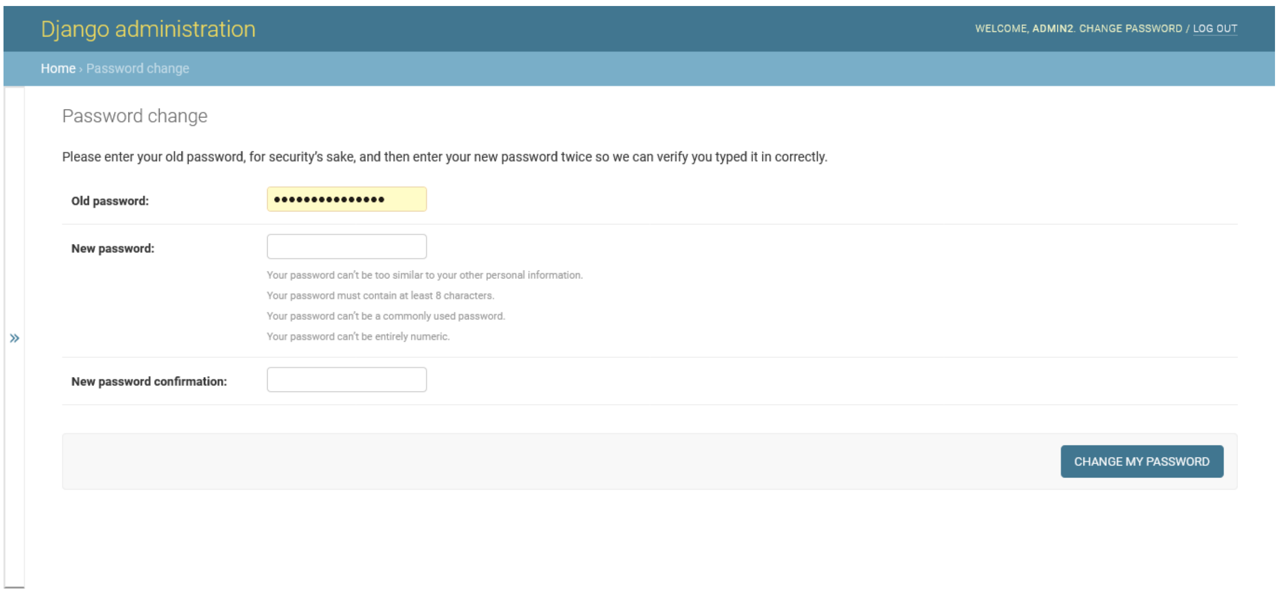
Order Code	Process Name	Quantity	Payment/unit	Total
Ak506a	Piping	50	2.00	100.00
SC0001	Front stitches	1,000	1.00	1,000.00

Total piece rate payment	: 1,100.00
Daily salary X 5.9	: 0
Allowance	: 0
Deduction	: 0
Total	**: 1,100.0**

Figure 5.4.56 Salary details PDF file example

- ***Change password page:***

  Authorized users with staff and superuser status can change the password of a user by going to the Django admin dashboard. The change password page is shown in Figure **5.4.57**.



Django administration

WELCOME, ADMIN2. CHANGE PASSWORD / LOG OUT

Home › Password change

Password change

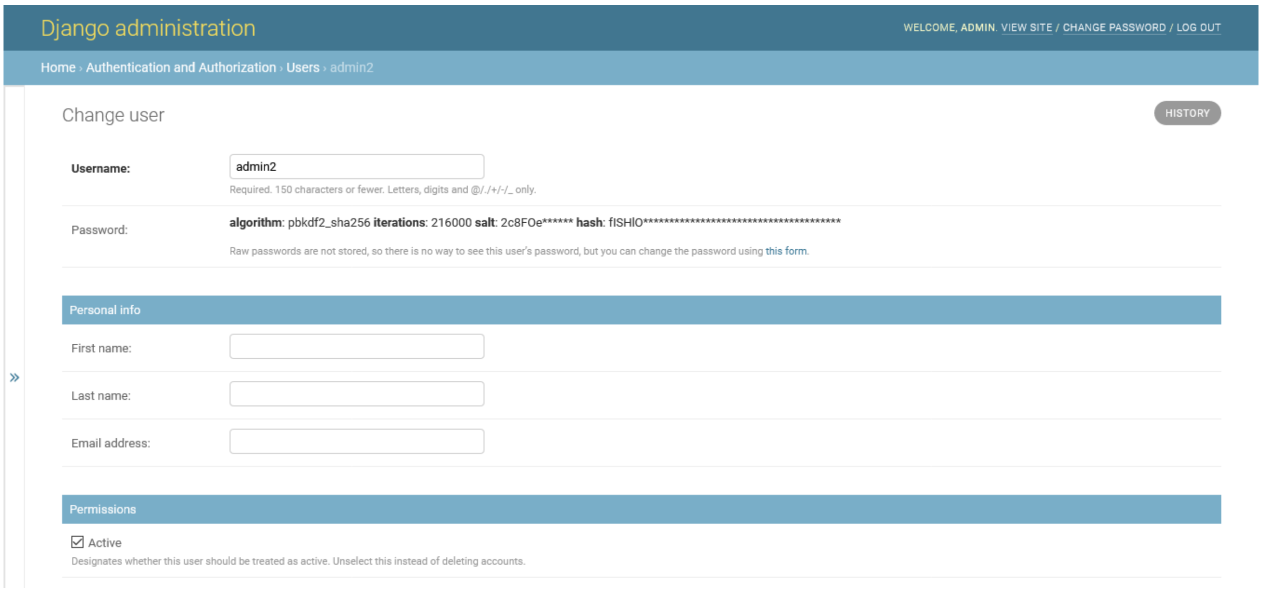
Please enter your old password, for security's sake, and then enter your new password twice so we can verify you typed it in correctly.

Old password:	•••••••••••••••
New password:	Your password can't be too similar to your other personal information. Your password must contain at least 8 characters. Your password can't be a commonly used password. Your password can't be entirely numeric.
New password confirmation:	

Figure 5.4.57 Change password page

- ***Edit user page:***

  Authorized users with staff and superuser status can change the users' information by going to the Django admin dashboard. The change user page is shown in Figure **5.4.58**.


Django administration
WELCOME, ADMIN. VIEW SITE / CHANGE PASSWORD / LOG OUT
Home › Authentication and Authorization › Users › admin2
HISTORY
Personal info
Permissions

Change user

Username: admin2

Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Password: **algorithm**: pbkdf2_sha256 **iterations**: 216000 **salt**: 2c8FOe****** **hash**: fISHlO**************************************

Raw passwords are not stored, so there is no way to see this user's password, but you can change the password using this form.

First name:

Last name:

Email address:

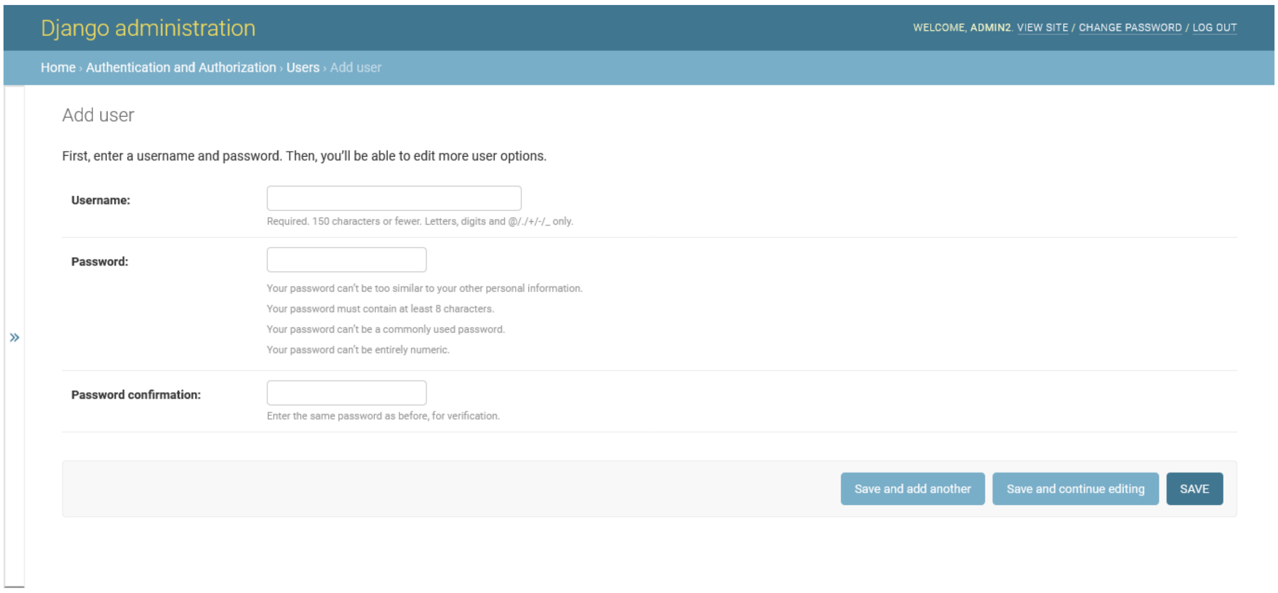
☑ Active

Designates whether this user should be treated as active. Unselect this instead of deleting accounts.

Figure 5.4.58 Change user page

➢ ***Change password page:***

Authorized users with staff and superuser status can change the add user by going to the Django admin dashboard. The add user page is shown in Figure **5.4.59**.


Django administration
WELCOME, ADMIN2. VIEW SITE / CHANGE PASSWORD / LOG OUT
Home › Authentication and Authorization › Users › Add user
Save and add another
Save and continue editing
SAVE

Add user

First, enter a username and password. Then, you'll be able to edit more user options.

Username:

Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Password:

Your password can't be too similar to your other personal information.
Your password must contain at least 8 characters.
Your password can't be a commonly used password.
Your password can't be entirely numeric.

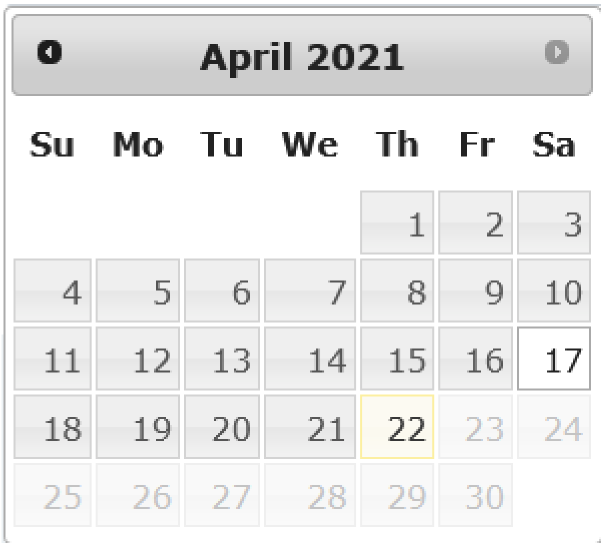
Password confirmation:

Enter the same password as before, for verification.

Figure 5.4.59 Add user page

➢ ***Choose date user interface:***

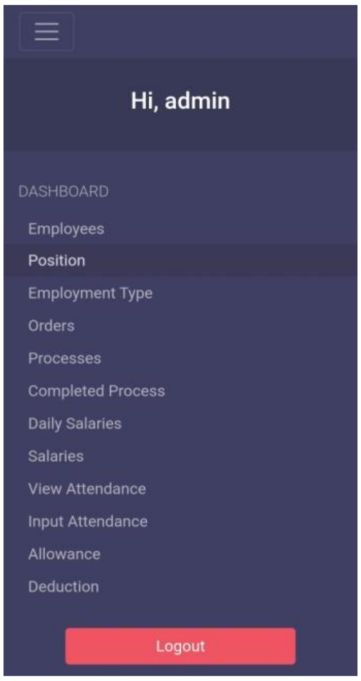
The user interface of choosing the date is shown below (Figure **5.4.60**):

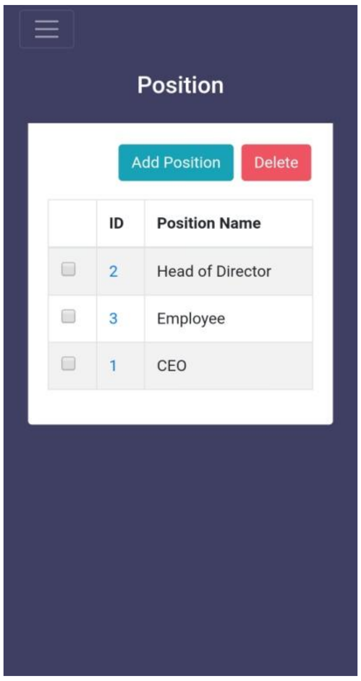


				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

Figure 5.4.60 Choose date user interface

➢ ***Several system screenshots on a smartphone:***





	ID	Position Name
☐	2	Head of Director
☐	3	Employee
☐	1	CEO

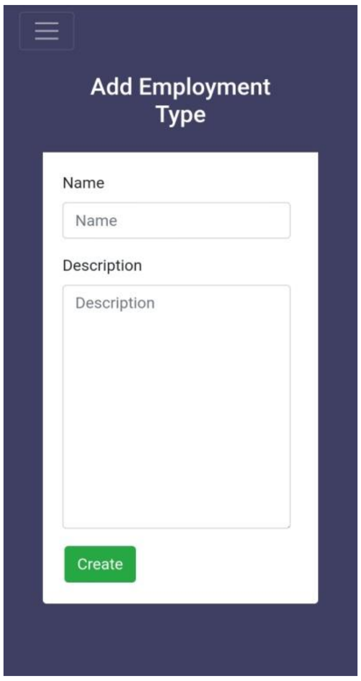
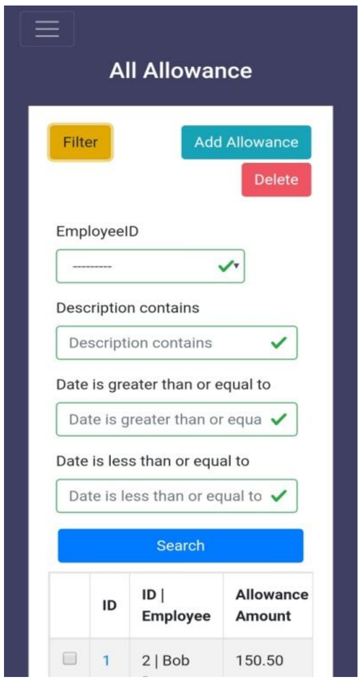


Figure 5.4.63 Sidebar view on phone

Figure 5.4.62 Position page on phone

Figure 5.4.61 Add employment page on phone


All Allowance

Filter

Add Allowance

Delete

EmployeeID

Description contains

Description contains

Date is greater than or equal to

Date is greater than or equa

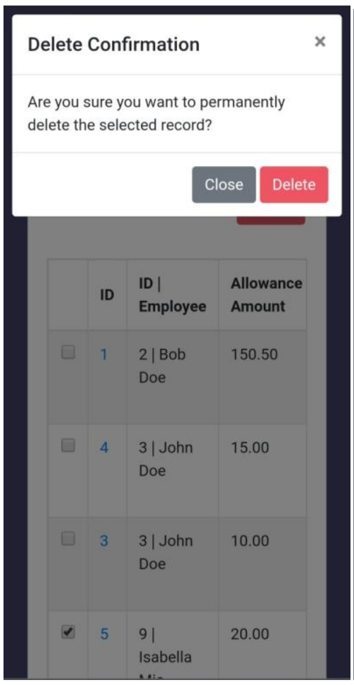
Date is less than or equal to

Date is less than or equal to

Search

	ID	ID | Employee	Allowance Amount
☐	1	2 | Bob	150.50

Figure 5.4.64 Allowance page with filter part on phone

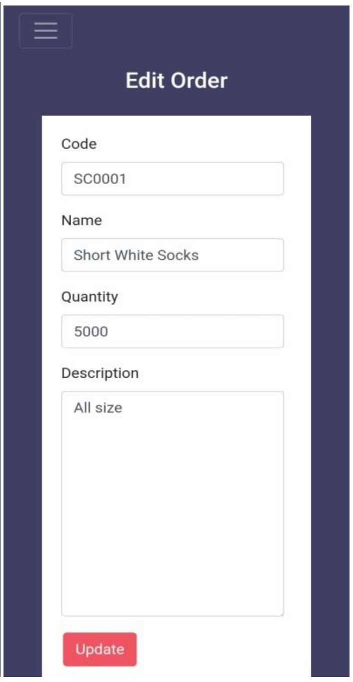

Delete Confirmation ×

Are you sure you want to permanently delete the selected record?

Close Delete

	ID	ID | Employee	Allowance Amount
☐	1	2 | Bob Doe	150.50
☐	4	3 | John Doe	15.00
☐	3	3 | John Doe	10.00
☑	5	9 | Isabella	20.00

Figure 5.4.65 Delete confirmation pop up on phone


Edit Order

Code: SC0001

Name: Short White Socks

Quantity: 5000

Description: All size

Update

Figure 5.4.66 Edit order page on phone

6. System Testing

6.1. Test Method

Generally, software testing technology can be divided into white-box testing and black-box testing [46]:

White-box testing, also known as structural testing, is the thorough examination of the code's structure and the code's internal logic. Because of this testing's characteristics, the tester must have complete knowledge and access to the source code. So, a qualified tester is expected to do this test. Consequently, this test is very costly and time-consuming. However, through this testing, the maximum coverage is achieved when writing the test scenario. It can also uncover hidden code and reveal implementation errors. White box testing can be applied at system, integration, and unit levels of the software testing process [47].

On the other hand, black-box testing, also known as functional testing, is testing without the need to know the internal structure of the code. It only inspects the software's fundamental aspects [47]. It has nothing to do with the inner mechanism of the system; the tester only focuses on the output produced in response to the inputs and execution conditions [46]. Thus, it is usually less time-consuming and has a more uncomplicated test perception compared with white box testing [47]. In the black-box testing technique, the tester designs the test cases according to the information in the specification. The tester knows the expected output and conducts tests to ensure that the system outputs what should be outputted [48].

6.2. Test Constraint

All the tests are done in two environments, Heroku and localhost. The localhost environment is also the development environment of the system. Python virtual environment (venv module) is used for running the server in the localhost.

6.3. Test Cases

Three representative test cases are shown below:

6.3.1. Authentication

Table 6.3.1 Authentication related test cases

Test Case ID	Test Content	Expected result	Test Result
Auth-1	Login with a correct username and password	User can log in to the payroll web app and access it	Achieve the expected result
Auth-2	Login with a non-existent account	The error message is displayed to the user	Achieve the expected result
Auth-3	Logout of the system	The user is logged out of the system	Achieve the expected result
Auth-4	User goes to any page before logging in	The user is either directed to the login page, or an error message is shown	Achieve the expected result
Auth-5	User with staff and superuser and status login to Django admin dashboard	The user is logged in to the Django admin dashboard	Achieve the expected result
Auth-6	User without staff and	User can not log in to the	Achieve the expected

	superuser and status login to Django admin dashboard	Django admin dashboard	result
Auth-7	User with staff and superuser and status change user password	User password is changed	Achieve the expected result
Auth-8	User with staff and superuser and status edit user information	User information is changed	Achieve the expected result

6.3.2. Completed Process

Required fields: Process ID, employee ID, and quantity

Data constraints:

- Quantity must be greater than 1 and less than its process quantity minus the existing completed process quantity of that process (the total completed process must be less or equal to its process quantity)

Table 6.3.2 Completed process related test cases

Test Case ID	Test Content	Expected result	Test Result
CPro-1	View the completed process page	User can view the completed process page	Achieve the expected result
CPro-2	Add completed process (all required fields are filled correctly)	A new completed process is added to the database	Achieve the expected result
CPro-3	Try to add completed process without filling all required fields or filling it incorrectly	An error message is displayed	Achieve the expected result
CPro-4	Edit completed process information with quantity less or equal to its process' quantity minus the existing completed	Completed process information is updated in the database	Achieve the expected result

	process quantity of that process		
CPro-5	Edit completed process information with quantity higher than its process' quantity minus the existing completed process quantity of that process	An error message is displayed	Achieve the expected result
CPro-6	Click the delete button on the completed process page and in the delete pop-up window without selecting anything	Nothing is deleted from the database, and the "Nothing is selected" message (**Figure 5.4.8**) is displayed	Achieve the expected result
CPro-7	Delete one or more completed processes	The selected completed processes are deleted from the database, and a success message (**Figure 5.4.7**) is displayed	Achieve the expected result
Pro-8	Filter completed processes	The completed processes in the completed process page are filtered correctly according to the user's input	Achieve the expected result
Pro-9	Click the process name on the completed process page	The system redirects the page to completed processes of that process page	Achieve the expected result
Pro-10	Filter completed processes of a particular process	The completed processes in the completed process of a particular process page are filtered correctly according to the user's	Achieve the expected result

		input	
Pro-11	Click the process name on the completed process of process page	The user is redirected to the edit process page	Achieve the expected result
Pro-12	Click the employee name on the completed process page	The system redirects the page to completed processes of that employee page	Achieve the expected result
Pro-13	Click the ID in the completed process of employee page	The user is redirected to the edit completed process page of that ID	Achieve the expected result
Pro-14	Filter completed processes of employee	The completed processes in the completed process of employee page are filtered correctly according to the user's input	Achieve the expected result

6.3.3. Salary

Table 6.3.3 Salary related test cases

Test Case ID	Test Content	Expected result	Test Result
Sal-1	Go to the salary page after entering the date range	User can view all employees' salary	Achieve the expected result
Sal-2	Go to the salary page before entering the date range	An error page is displayed	Achieve the expected result
Sal-3	Click "View PDF" on the salary page	A new tab with the PDF of the salary page is opened	Achieve the expected result
Sal-4	Click "Download PDF" on the salary page	A PDF file is downloaded	Achieve the expected result
Sal-5	Click the employee on the salary page	The system redirects the page to salary details page of that employee and	Achieve the expected result

		shows the attendance day, the completed processes, the employee’s daily salary, allowances, and deductions that the employee has within the date range the user entered	
Sal-6	Click “View PDF” on the salary details page	A new tab with the PDF of the salary details of the employee page is opened	Achieve the expected result
Sal-7	Click “Download PDF” on the salary details page	Salary details of the employee PDF file is downloaded	Achieve the expected result

7. Conclusion and Future Work

7.1. Conclusion

The web application defined in this thesis has been developed and is working correctly. The conclusions are as follows:

- The web application has all of the requirements specified in the system requirements chapter of this thesis.
- According to the testing that has been done, all of the functions in the system works with no problem whatsoever.

7.2. Future Works

In the future, the system can be developed further. Some suggestions are as follows:

- Add more account type, such as employee account and HR account where each of them has different permission in the system, for example, employee account can only look at their own salary.
- Automatically add an employee account when a new employee is added with a default username and password, which can be changed by the

employee.

- Send salary details email to all employees after the salary has been paid.

References

[1] S. M. Bragg, *Accounting for Payroll: A Comprehensive Guide*. 2004.

[2] F. H. Rusly, A. Ahmi, Y. Y. A. Talib, and K. Rosli, “Payroll system: A bibliometric analysis of the literature,” in *AIP Conference Proceedings*, Sep. 2018, vol. 2016, no. 1, p. 020124, doi: 10.1063/1.5055526.

[3] K. McBeth, “What is Payroll? A 2021 guide to Processing Payroll | QuickBooks.” https://quickbooks.intuit.com/r/payroll/what-is-payroll/ (accessed Apr. 13, 2021).

[4] ASGI, “ASGI Documentation.” https://asgi.readthedocs.io/en/latest/ (accessed Apr. 18, 2021).

[5] “The Web framework for perfectionists with deadlines | Django.” https://www.djangoproject.com/ (accessed Apr. 15, 2021).

[6] “About SQLite.” https://www.sqlite.org/about.html (accessed Apr. 15, 2021).

[7] WSGI, “What is WSGI?,” *WSGI.org*. https://wsgi.readthedocs.io/en/latest/what.html (accessed Apr. 18, 2021).

[8] “PAYROLL | definition in the Cambridge English Dictionary.” https://dictionary.cambridge.org/us/dictionary/english/payroll (accessed Apr. 13, 2021).

[9] B. Shearer, “Piece Rates, Fixed Wages and Incentives: Evidence from a Field Experiment,” 2004. Accessed: Apr. 14, 2021. [Online]. Available: https://academic.oup.com/restud/article-abstract/71/2/513/1590219.

[10] “What Is a Fixed Hourly Rate of Pay?” https://work.chron.com/notify-employer-change-availabilty-3322.html (accessed Apr. 15, 2021).

[11] G. Ferguson, “How do I Calculate a Salary?” https://smallbusiness.chron.com/calculate-salary-3605.html (accessed Apr. 15, 2021).

[12] “PIECE RATE | definition in the Cambridge English Dictionary.” https://dictionary.cambridge.org/us/dictionary/english/piece-rate (accessed Apr.

14, 2021).

[13] C. Houston, “The Ins and Outs of Paying Piece Rate - Stratus.hr®.” https://stratus.hr/2018/09/24/piece-rate/ (accessed Apr. 13, 2021).

[14] S. Helper, M. Kleiner, and Y. Wang, “Analyzing Compensation Methods in Manufacturing: Piece Rates, Time Rates, or Gain-Sharing?,” Cambridge, MA, Nov. 2010. doi: 10.3386/w16540.

[15] I.-M. Project, “Homeworkers in Indonesia Results from the homeworker: mapping study in North Sumatra, West Java, Central Java, Yogyakarta, East Java and Banten,” 2015. Accessed: Apr. 14, 2021. [Online]. Available: http://www.ilo.org/jakarta/whatwedo/publications/WCMS_438252/lang--en/index.htm.

[16] F. Borino, “Piece rate pay and working conditions in the export garment sector,” Dec. 2018. Accessed: Apr. 14, 2021. [Online]. Available: www.ilo.org/publns.

[17] R. Irwansyah, K. Heryanda, and N. M. D. A. Mayasari, “No Title,” *Glob. Sci. Journals*, vol. 8, no. 5, 2020, [Online]. Available: https://www.globalscientificjournal.com/researchpaper/DEMAND_OF_LABOUR_AT_MSMES_IN_BANGLI_.pdf.

[18] H. Sandee, B. Isdijoso, and S. Sulandjari, “SME clusters in Indonesia: an analysis of growth dynamics and employment conditions,” 2003. Accessed: Apr. 14, 2021. [Online]. Available: http://www.ilo.org/jakarta/whatwedo/publications/WCMS_123971/lang--en/index.htm.

[19] M. Mawardi, “The Survival of Micro, Small and Medium Enterprises (MSMEs) in Indonesian Industrial Clusters: A Case Study of the Furniture and Footwear Industrial Cluster of East Java Province, Indonesia,” *Univ. Wollongong Thesis Collect. 1954-2016*, Jan. 2014, Accessed: Apr. 14, 2021. [Online]. Available: https://ro.uow.edu.au/theses/4222.

[20] T. Santia, “Berapa Jumlah UMKM di Indonesia? Ini Hitungannya - Bisnis Liputan6.com,” Sep. 04, 2020.

https://www.liputan6.com/bisnis/read/4346352/berapa-jumlah-umkm-di-indonesia-ini-hitungannya (accessed Apr. 15, 2021).

[21] "UU Cipta Kerja Buka Ruang Konsolidasi Data Tunggal KUMKM," *Kementrian koperasi dan UKM*. http://www.depkop.go.id/read/uu-cipta-kerja-buka-ruang-konsolidasi-data-tunggal-kumkm (accessed Nov. 26, 2020).

[22] E. Sri, J. Asyikin, and O. Sari, "Penerapan Sistem Akuntansi Dasar pada Usaha Kecil Menengah di kota Banjarmasin," *media.neliti.com*, vol. 6, no. 2, pp. 81–91, Sep. 2016, Accessed: Apr. 15, 2021. [Online]. Available: https://media.neliti.com/media/publications/163609-ID-penerapan-sistem-akuntansi-dasar-pada-us.pdf.

[23] S. Fatimah, "Faktor–Faktor Yang Mempengaruhi Terkendalanya Praktik Akuntansi Pada UMKM Di Kabupaten Jember," Universitas Muhammadiyah Jember, 2020.

[24] E. Penti Kurniawati, P. Ika Nugroho, and dan Chandra Arifin, "Penerapan Akuntansi pada Usaha Mikro Kecil dan Menengah (UMKM)," *J. Manaj. dan Keuang.*, vol. 10, no. 2, Jan. 2015, Accessed: Apr. 15, 2021. [Online]. Available: http://jurnal.darmajaya.ac.id/index.php/jmk/article/view/332.

[25] Y. A. Indra, "Penerapan Sebelum dan Sesudah Sistem Informasi Akuntansi Syariah Dengan Menggunakan Aplikasi Software Accounting Dalam Penyusunan Laporan Keuangan Bagi UMKM," *Al-Intaj J. Ekon. dan Perbank. Syariah*, vol. 6, no. 2, pp. 77–87, Sep. 2020, Accessed: Apr. 15, 2021. [Online]. Available: ejournal.iainbengkulu.ac.id.

[26] R. Aribowo and Z. Zulkifli, "Evaluasi Penerapan Sistem Akuntansi Penjualan Tunai dan Penerimaan Kas Pada UMKM (Studi Kasus Pada Waza-Waza Angkringan Bento)," STIE Widya Wiwaha, 2017.

[27] International Labour Organization (ILO), "The power of small: Unlocking the potential of SMEs - InfoStories," Oct. 2019. https://ilo.org/infostories/en-GB/Stories/Employment/SMEs (accessed Apr. 15, 2021).

[28] "Payroll Software Market By Component, By Organization Size, By

Deployment Type, By Verticals and By Region - Global Forecast up to 2025," Nov. 2019. https://www.researchandmarkets.com/reports/4866495/payroll-software-market-by-component-by (accessed Apr. 15, 2021).

[29] "Executive summary: Explore the results of the 2018 Payroll Operations Survey," 2018. Accessed: Apr. 15, 2021. [Online]. Available: https://www2.deloitte.com/content/dam/Deloitte/us/Documents/human-capital/us-2018-payroll-operations-survey.pdf.

[30] Data Flair, "Django Tutorial for Beginners - Learn the Core Aspects of Django Framework." https://data-flair.training/blogs/django-tutorial/ (accessed Apr. 16, 2021).

[31] Django, "FAQ: General | Django documentation | Django," *Django Software Foundation*. https://docs.djangoproject.com/en/3.2/faq/general/ (accessed Apr. 16, 2021).

[32] D. Rubio, *Beginning Django*, 1st ed. Berkeley, CA: Apress, 2017.

[33] J. Plekhanova, "Evaluating web development frameworks: Django, Ruby on Rails and CakePHP," vol. 20, p. 2009, Sep. 2009, Accessed: Apr. 18, 2021. [Online]. Available: www.ibit.temple.edu.

[34] Django, "Models | Django documentation | Django," *Django Software Foundation*. https://docs.djangoproject.com/en/3.1/topics/db/models/ (accessed Apr. 17, 2021).

[35] Django, "Writing views | Django documentation | Django," *Django Software Foundation*. https://docs.djangoproject.com/en/3.2/topics/http/views/ (accessed Apr. 18, 2021).

[36] P. Priya, "Django Architecture | Working System Of Django MVT Architecture." https://www.educba.com/django-architecture/ (accessed Apr. 17, 2021).

[37] Django, "Templates | Django documentation | Django," *Django Software Foundation*. https://docs.djangoproject.com/en/3.2/topics/templates/ (accessed Apr. 18, 2021).

[38] The django book, "Django's Structure – A Heretic's Eye View."

https://djangobook.com/mdj2-django-structure/ (accessed Apr. 17, 2021).

[39] Django, “Writing your first Django app, part 1 | Django documentation | Django,” *Django Software Foundation*. https://docs.djangoproject.com/en/3.2/intro/tutorial01/ (accessed Apr. 18, 2021).

[40] T. Ko, “A Quick Glance of Django,” *Medium,* Sep. 10, 2017. https://medium.com/@timmykko/a-quick-glance-of-django-for-beginners-688bc6630fab (accessed Apr. 18, 2021).

[41] M. Patel, “Functional Requirements Vs. Non-functional Requirements,” *Kody Technolab,* Jul. 13, 2020. https://kodytechnolab.com/functional-requirements-vs-non-functional-requirements (accessed Apr. 19, 2021).

[42] V. Ng, “Non-Functional Requirement of the Mobile Development system,” Mar. 10, 2019. https://medium.com/@vishwasng/non-functional-requirement-of-the-mobile-development-system-e0ed98f2a872 (accessed Apr. 21, 2021).

[43] Python, “venv — Creation of virtual environments — Python 3.8.9 documentation,” *Python Software Foundation*. https://docs.python.org/3.8/library/venv.html (accessed Apr. 22, 2021).

[44] Python, “12. Virtual Environments and Packages — Python 3.9.4 documentation,” *Python Software Foundation*. https://docs.python.org/3/tutorial/venv.html (accessed Apr. 22, 2021).

[45] Github, “About branches - GitHub Docs,” *Github Inc*. https://docs.github.com/en/github/collaborating-with-issues-and-pull-requests/about-branches (accessed Apr. 22, 2021).

[46] S. Nidhra, “Black Box and White Box Testing Techniques - A Literature Review,” *Int. J. Embed. Syst. Appl.*, vol. 2, no. 2, pp. 29–50, 2012, doi: 10.5121/ijesa.2012.2204.

[47] M. E. Khan and F. Khan, “A Comparative Study of White Box, Black Box and Grey Box Testing Techniques,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 3, no. 6, pp. 22–25, 2012, [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.685.1887&rep=rep1

&type=pdf#page=22.

[48] P. Mitra, S. Chatterjee, and N. Ali, “Graphical analysis of MC/DC using automated software testing,” in *2011 3rd International Conference on Electronics Computer Technology*, Apr. 2011, pp. 145–149, doi: 10.1109/ICECTECH.2011.5941819.

ACKNOWLEDGEMENT

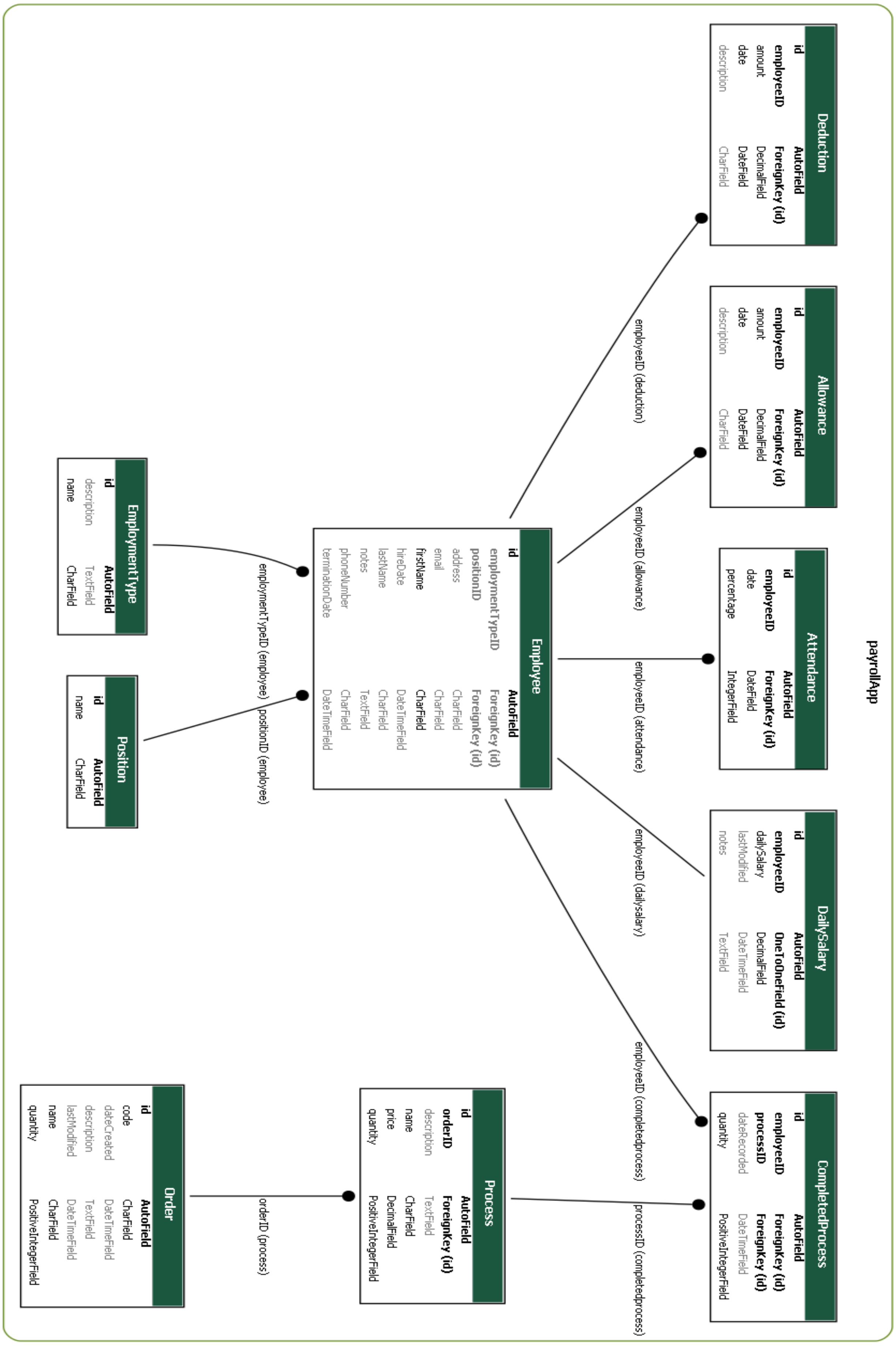
I want to take this opportunity to express my deepest gratitude to all the people who made this thesis possible. First of all, I would like to thank God that he has given me perseverance, patience, guidance, and the capabilities to finish this thesis.

I am deeply grateful to Professor Wang Haibin (王海彬) as my supervisor, who has been helpful in providing guidance, suggestions, and supporting me with care and patience in finishing this project and thesis. I also would like to express my appreciation to my family, who has given me so much support, encouragement, and love.

I would like to extend my sincere thanks to Bernardy Ho, Turdialiev Bekhzod, and my fellow classmates for their encouragement and those who have given their time to help me on some aspects of the thesis and throughout the years of my study. I also would like to express my gratitude to all of the teachers who have taught us and to the staff of the College of International Students (CIS), who always took care of the international students at Nanjing University of Information Science and Technology.

May 2021

Valetta Jesslyn


payrollApp
Deduction
id AutoField
employeeID ForeignKey (id)
amount DecimalField
date DateField
description CharField
Allowance
id AutoField
employeeID ForeignKey (id)
amount DecimalField
date DateField
description CharField
Attendance
id AutoField
employeeID ForeignKey (id)
date DateField
percentage IntegerField
DailySalary
id AutoField
employeeID OneToOneField (id)
dailySalary DecimalField
lastModified DateTimeField
notes TextField
CompletedProcess
id AutoField
employeeID ForeignKey (id)
processID ForeignKey (id)
dateRecorded DateTimeField
quantity PositiveIntegerField
employeeID (deduction)
employeeID (allowance)
employeeID (attendance)
employeeID (dailysalary)
employeeID (completedprocess)
processID (completedprocess)
Employee
id AutoField
employmentTypeID ForeignKey (id)
positionID ForeignKey (id)
address CharField
email CharField
firstName CharField
hireDate DateTimeField
lastName CharField
notes TextField
phoneNumber CharField
terminationDate DateTimeField
Process
id AutoField
orderID ForeignKey (id)
description TextField
name CharField
price DecimalField
quantity PositiveIntegerField
employmentTypeID (employee)
positionID (employee)
orderID (process)
EmploymentType
id AutoField
description TextField
name CharField
Position
id AutoField
name CharField
Order
id AutoField
code CharField
dateCreated DateTimeField
description TextField
lastModified DateTimeField
name CharField
quantity PositiveIntegerField