

รายงาน

เรื่อง การพัฒนาเว็บไซต์สำหรับจ่ายเงินเดือนพนักงาน

โดย

613020574-6 นางสาวชนกนันท์ สมฝัน

เสนอ

อาจารย์ปัญญาพล หอระตะ

รายงานนี้เป็นส่วนหนึ่งของรายวิชา SC313303 การพัฒนาโปรแกรมประยุกต์สำหรับองค์กร

ภาคเรียนที่ 2 ปีการศึกษา 2563

สาขาวิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์

มหาวิทยาลัยขอนแก่น

(เดือนพฤษภาคม พ.ศ.2564)

อธิบายการทำงานของ Code

1. Configure

package me.dorin.payroll.configure เป็น package เกี่ยวกับการกำหนดการเข้าถึงเว็บไซต์ ตั้งแต่ขั้นตอนการตรวจสอบ token ไปจนถึงการกำหนด UserSecurityObject เช่น การสร้าง Class SecurConfig โดยขยายความสามารถของ WebSecurityConfigurerAdapter และ Override Method configure และใช้ @EnableWebSecurity เพื่อเปิดการใช้งาน Web Security ดังภาพ

```
@Configuration
@EnableWebSecurity
@ComponentScan("me.dorin.payroll.configure.security")
@EnableGlobalMethodSecurity(prePostEnabled = true)
@AllArgsConstructor
public class SecurityConfig extends WebSecurityConfigurerAdapter {
    private final CustomAuthenticationProvider authProvider;
    private final JwtAuthenticationEntryPoint jwtAuthenticationEntryPoint;
    private final JwtRequestFilter jwtRequestFilter;

    @Override
    public void configure(HttpSecurity http) throws Exception {
        http.cors().and().csrf().disable()
            .authorizeRequests().antMatchers("/login", "/error", "/").permitAll()
            .anyRequest().authenticated()
            .and().exceptionHandling().authenticationEntryPoint(jwtAuthenticationEntryPoint)
            .and().sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS);

        http.addFilterBefore(jwtRequestFilter, UsernamePasswordAuthenticationFilter.class);
    }
}
```

ใน package security จะทำงานเกี่ยวกับการตรวจสอบความถูกต้องของผู้ใช้โดยจะทำการโหลดข้อมูลของผู้ใช้จาก Username และทำการตรวจสอบว่ามีในฐานข้อมูลหรือไม่ ดังภาพ

```
package me.dorin.payroll.configure.security;

import ...

@Component
@AllArgsConstructor
public class CustomAuthenticationProvider implements UserDetailsService {
    private final EmployeeRepository employeeRepository;

    @Override
    public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {
        Optional<Employee> user = employeeRepository.findByUsername(username);
        if (user.isEmpty()) {
            throw new UsernameNotFoundException(username);
        }

        return new SecurityUserDetail(user.get());
    }
}
```

2. Exception

package me.dorin.payroll.exception เป็น package เกี่ยวกับการ Handling Errors เราจะทำการสร้าง Custom Exception มาช่วยในการจัดการ Errors Cases ซึ่งในแต่ละ Exception จะระบุข้อความที่จะส่งกลับหาผู้ใช้ เพื่อให้ผู้ใช้อ่านแล้วสามารถเข้าใจในทันที เช่น Class DuplicatedUsername คือกรณีที่ผู้ใช้สร้าง username ตรงกับที่มีอยู่แล้วในระบบ จากนั้นทำการใช้ @ResponseStatus เพื่อกำหนดให้ Class DuplicatedUsername ทำการ response Http Status เป็น Bad Request ดังภาพ

```
package me.dorin.payroll.exception;

import ...

@ResponseStatus(code = HttpStatus.BAD_REQUEST)
public class DuplicatedUsername extends RuntimeException {
    public DuplicatedUsername(String username) {
        super(String.format("Username = %s duplicated", username));
    }
}
```

3. Web

ใช้สำหรับเก็บโค้ดตัวโปรแกรม

3.1 Controller

คือส่วนของการเริ่มการทำงาน และรับคำสั่งมาจากผู้ใช้งานผ่าน View โดย View จะส่งข้อมูลมายัง Controller และ Controller จะประมวลผล โดยบางคำสั่งอาจจะต้องติดต่อกับ Model เพื่อทำการประมวลผล ข้อมูลอย่างถูกต้องเรียบร้อยแล้วก็จะส่งไปยัง View เพื่อแสดงผลตามคำสั่งที่ผู้ใช้งานร้องขอ เช่น Class EmployeeController

- @RequestMapping(path = "/employee") ใช้ระบุว่าหากมีการเรียกใช้ path /employee ให้เข้ามาตรวจสอบที่ Class นี้
- @RestController ใช้เพื่อระบุว่า Class นี้เป็น Controller
- @AllArgsConstructor เป็น Annotation ของ Lombok ใช้สร้าง constructor แบบมี args ครบ

- @GetMapping, @PostMapping, @PatchMapping และ @DeleteMapping ใช้เพื่อบอก spring ว่าหากมีการส่งข้อมูลตาม Http Method ที่ระบุไว้ให้เรียกใช้ Java Method นั้น ๆ
- @RequestBody คือการระบุว่าให้รับ Parameter นี้มาจาก Body
- @PathVariable คือการระบุว่าให้รับ Parameter นี้มาจาก Path
- ในทุก ๆ Method จะส่งข้อมูลที่รับมาไปประมวลผลต่อที่ Service เสมอเพื่อความเรียบร้อย

ดั่งภาพ

```

@RestController
@RequestMapping(path = "/employee")
@AllArgsConstructor
public class EmployeeController {
    private final EmployeeService employeeService;

    @GetMapping
    public Page<Employee> list(Pageable pageable) {
        return employeeService.getAllEmployee(pageable);
    }

    @GetMapping(path =("/{id}")
    public Employee getEmployeeById(@PathVariable Long id) { return employeeService.getEmployeeById(id); }

    @PostMapping
    @Transactional
    public Employee createEmployee(@RequestBody EmployeeCreateRequest request) throws DuplicatedUsername {
        return employeeService.createEmployee(request);
    }

    @PatchMapping(path =("/{id}")
    @Transactional
    public Employee updateEmployee(@PathVariable("id") Long id, @RequestBody EmployeeUpdateRequest request) {
        return employeeService.updateEmployee(id, request);
    }

    @DeleteMapping(path =("/{id}")
    @Transactional
    public Employee deactivateEmployee(@PathVariable("id") Long id) { return employeeService.disableEmployee(id); }

    @PatchMapping(path =("/{id}", params = "a=reactivate")
    @Transactional
    public Employee reactivateEmployee(@PathVariable("id") Long id) { return employeeService.reactivateEmployee(id); }

    @PostMapping(path =("/{id}", params = "a=updateSalary")
    @Transactional
    public EmployeeSalary updateEmployeeSalaryAndRole(@PathVariable("id") Long id,
        @RequestBody EmployeeSalaryUpdateRequest request) {
        return employeeService.updateEmployeeSalary(id, request);
    }
}

```

3.2 Model

ใช้สำหรับเก็บ DAO หรือ Data Access Object ไว้ โดยในขั้นตอนนี้จะระบุ Relation ของ Data แต่ละชุดไว้ชัดเจนเพื่อให้ Spring สามารถ Auto generate ขึ้นมาได้ เช่น Class Employee

- @Entity ใช้เพื่อบอกว่า Class นี้เป็น DAO หรือ Data Access Object
- @Table(name = "employee") ใช้ระบุว่า Class นี้จะต้องตั้งชื่อตารางว่าอะไร
- @Data เป็น Annotation Lombok ใช้ Generate Getter, Setter equals toString ให้อัตโนมัติ
- @Id ใช้บอกว่า Field นี้เป็น Id
- @NotNull เพื่อบอกว่า Field นี้ต้องไม่เป็น Null
- @Valid เพื่อให้ตรวจสอบ Input ที่รับเข้ามา
- @JsonIgnore เพื่อไม่ให้ส่งข้อมูลนี้ออกไปขณะเรียกใช้ข้อมูลบนหน้าเว็บ
- @Column ใช้เพื่อบอกว่าต้องไปหาค่าของตัวแปรนี้ที่ Field ไหน
- @ElementCollection + @CollectionTable ใช้เพื่อให้ JPA สร้าง Relation แบบ oneToMany แบบพิเศษให้เอง

ดังภาพ

```
@Data
@Entity
@Table(name = "employee")
public class Employee {
    @Id
    @NotNull
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Setter(AccessLevel.PRIVATE)
    private Long id;

    @NotNull
    @Valid
    private String thaiName;

    @NotNull
    @Valid
    private String englishName;

    @NotNull
    @Valid
    private String socialId;

    @NotNull
    @Valid
    private String address;

    @ElementCollection
    @CollectionTable(name = "salary", joinColumns = @JoinColumn(name = "employee_id"))
    private List<EmployeeSalary> salaryHistory;

    private boolean hasPayrollAccess;
```

3.3 Repository

คลังสำหรับเก็บคำสั่งการ Query ข้อมูล ในลักษณะของการ Query ข้อมูลจาก Entity Class เช่น Class EmployeeSalaryRepository

- extends CrudRepository<Employee, Long> ซึ่ง CrudRepository เป็น interface ที่เก็บคำสั่งเบื้องต้นไว้ เช่นคำสั่ง findAll(), findOne(), save(), count(), delete() เป็นต้น ซึ่งเราสามารถเรียกใช้งานได้เลยโดยที่เราไม่ต้องเขียนขึ้นมาเอง และมันยังเป็น interface ที่กำหนดการดำเนินการพื้นฐานสำหรับฐานข้อมูล Create, Read, Update, Delete ซึ่งมี Method ที่สำคัญเอาไว้ให้ spring framework มาอ่านเอาข้อมูลไปใช้ในการสร้างคำสั่ง SQL อีกด้วย โดยวิธีการนั้นจะใช้เทคนิคที่เรียกว่า Java Reflection
- @Repository เพื่อบอกว่า Class นี้เป็น Interface ที่ใช้ Access DB
- @Query ใช้ในกรณีที่ต้องการระบุ HQL เอง

ดั่งภาพ

```
package me.dorin.payroll.web.repository;

import ...

@Repository
public interface EmployeeRepository extends CrudRepository<Employee, Long> {
    List<Employee> findByDeactivated(boolean deactivated);
    Page<Employee> findByDeactivated(boolean deactivated, Pageable pageable);
    Optional<Employee> findByUsername(String username);

    @Query("SELECT e FROM Employee e WHERE e.deactivated = false")
    List<Employee> findAllByActive();
}
```

3.4 Scheduler

package me.dorin.payroll.web.scheduler เป็น package ใช้สำหรับเก็บ Class scheduler ไว้ ซึ่งมีแค่ไฟล์เดียวคือ CutOffScheduler

- @Scheduled(cron = "0 0 0 1 1/1 ?") จะระบุให้ Spring รู้ว่า Method นี้จะต้องถูกรันทุกวัน ที่ 1 ตอนเวลาเที่ยงคืน ของทุกๆเดือน

- @Transactional เพื่อให้การทำงานเป็นแบบ Transactional หากมี Row ใดไม่สามารถทำต่อได้จะต้องยกเลิกทั้งหมด

ดั่งภาพ

```
package me.dorin.payroll.web.scheduler;

import ...

@Component
@Log
@AllArgsConstructor
public class CutOffScheduler {
    private final EmployeeRepository employeeRepository;
    private final PayoutService payoutService;

    @Scheduled(cron = "0 0 0 1 1/1 ?")
    @Transactional
    public void processCutOff() {
        log.info(msg: "Start processing salary payout");

        List<Employee> employees = employeeRepository.findAllByActive();

        employees.forEach(employee -> {
            PayoutLogCreateRequest request = new PayoutLogCreateRequest();
            request.setMonth(new Date());
            request.setRemark("Auto cut off");

            payoutService.createPayout(employee, request);
        });

        log.info(msg: "Finish processing salary payout");
    }
}
```

3.5 Service


ทำหน้าที่ในการ Create, Read, Update และ Delete Employee และ PayoutLog และสร้าง Class ที่ implements interface Service ที่เราเขียนไว้ไปใช้งาน โดยจะเก็บไว้ที่ package me.dorin.payroll.web.Service.implement โดยในส่วนนี้จะมีการใช้ Annotation @Service ซึ่งเราจะสามารถเรียกใช้งานผ่าน spring framework ได้ เช่น หากใน Class EmployeeController มีการเรียกใช้คำสั่ง getAllEmployee() ใน Class EmployeeServiceImpl ซึ่ง Implements จาก Class EmployeeService จะไปดึงข้อมูลจาก Repository ออกมาแสดงผล ดังภาพ

```
public class EmployeeServiceImpl implements EmployeeService {
    private final EmployeeRepository employeeRepository;
    private final PasswordEncoder passwordEncoder;

    @Override
    public Page<Employee> getAllEmployee(Pageable pageable) {
        return employeeRepository.findByDeactivated(deactivated: false, pageable);
    }
}
```

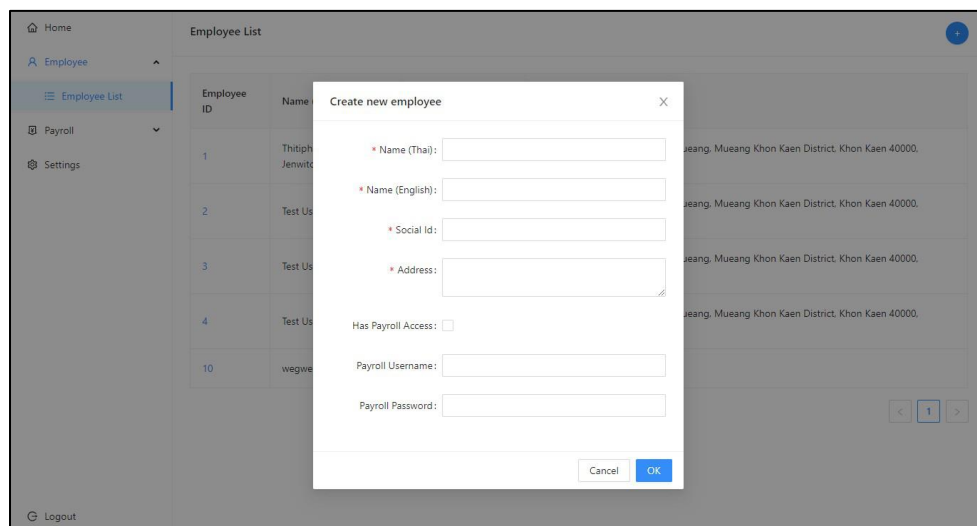
ตัวอย่างหน้าเว็บไซต์

1. หน้าสำหรับแสดง Employee ทั้งหมดที่มีอยู่ในระบบ



| Employee ID | Name (English) | Name (Thai) | Address |
|-------------|------------------------|--------------------------|------------------------------------------------------------------------------------------|
| 1 | Thitiphat Jenwichuwong | นายฐิติพันธ์ เจนวิฑูวงศ์ | 70 Thanon Klang Mueang, Nai Mueang, Mueang Khon Kaen District, Khon Kaen 40000, Thailand |
| 2 | Test User | ยูเซอร์ทดสอบ | 70 Thanon Klang Mueang, Nai Mueang, Mueang Khon Kaen District, Khon Kaen 40000, Thailand |
| 3 | Test User 2 | เทสยูเซอร์ 1 | 70 Thanon Klang Mueang, Nai Mueang, Mueang Khon Kaen District, Khon Kaen 40000, Thailand |
| 4 | Test User3 | ยูเซอร์ทดสอบ3 | 70 Thanon Klang Mueang, Nai Mueang, Mueang Khon Kaen District, Khon Kaen 40000, Thailand |
| 10 | wegweg | wtewegweg | gwegwegweg |

2. หน้าสำหรับสร้าง Employee ใหม่ หาก Employee ดังกล่าวมีสิทธิ์ในการเข้าถึงเว็บไซต์นี้ ให้กดเลือก “Has Payroll Access” และสร้าง Username และ Password สำหรับ Login ให้กับ Employee ดังกล่าว



Create new employee

* Name (Thai):

* Name (English):

* Social Id:

* Address:

Has Payroll Access: ☐

Payroll Username:

Payroll Password:

Cancel OK

3. หน้าสำหรับแสดงข้อมูลส่วนตัวของ Employee รวมถึงข้อมูลเงินเดือนที่ได้รับ

Home

Employee

Payroll

Settings

Logout

Thitiphat Jenwichuwong นายฐิติพัฒน์ เจนวิฑูวงศ์

Edit

Fired this employee

User Id: 1Social Id: 1449900444432

Address: 70 Thanon Klang Mueang, Nai Mueang, Mueang Khon Kaen District, Khon Kaen 40000, Thailand

Creation Time: Fri, April 23, 2021, 8:07:48 AMUpdate Time: Wed, May 5, 2021, 2:59:30 AM

Deactivated: falseDeactivate Time: No deactivated date

Permission

Has Payroll Access: truePayroll Username: manneaber

Employee Salary

Timestamp

Salary

Role

Auditor

Wed, May 5, 2021, 2:59:30 AM

500000

CEO

1

Wed, May 5, 2021, 2:55:40 AM

300000

CEO

1

Wed, May 5, 2021, 2:55:24 AM

150000

CEO

1

Tue, May 4, 2021, 3:45:46 PM

70000

CFO

1

4. หน้าสำหรับบอกสถานการณ์จ่ายเงินสำหรับ Employee แต่ละคน

Home

Employee

Payroll

Settings

Logout

Thitiphat Jenwichuwong - #18

Paylog

Timestamp

status

Auditor

remark

May 9, 2021, 12:29:43 AM

CREATE

1

May 9, 2021, 12:33:01 AM

AUDIT

1

Test Update

May 9, 2021, 12:33:06 AM

PAYMENT

1

Test Update

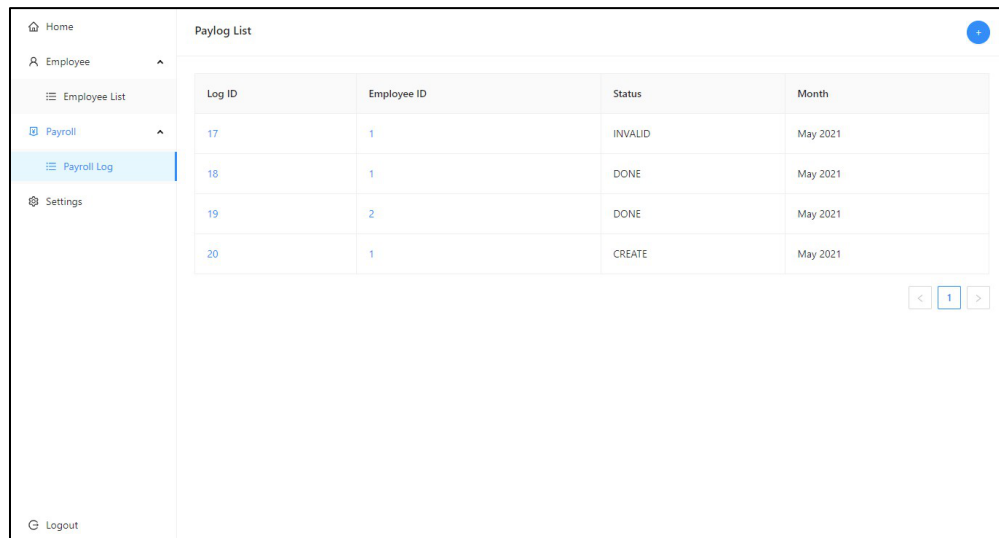
May 9, 2021, 12:33:11 AM

DONE

1

Test Update

5. หน้าสำหรับแสดงสถานการณ์จ่ายเงินของ Employee ทุกคน

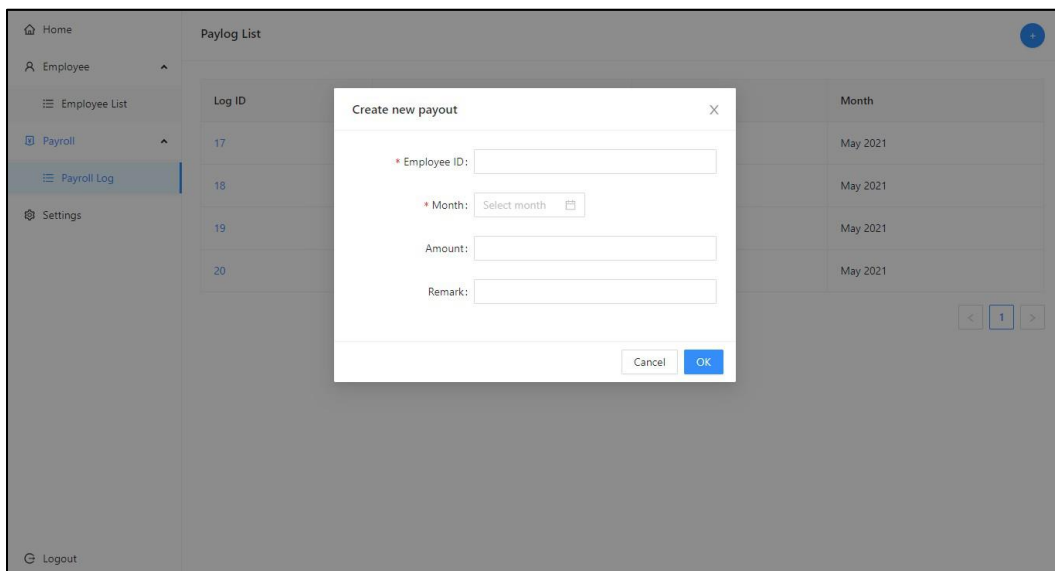


The screenshot shows a web application interface with a sidebar on the left containing navigation links: Home, Employee, Employee List, Payroll, Payroll Log (selected), and Settings. The main area is titled 'Paylog List' and contains a table with the following data:

| Log ID | Employee ID | Status | Month |
|--------|-------------|---------|----------|
| 17 | 1 | INVALID | May 2021 |
| 18 | 1 | DONE | May 2021 |
| 19 | 2 | DONE | May 2021 |
| 20 | 1 | CREATE | May 2021 |

At the bottom right of the table, there are pagination controls showing '< 1 >'. A 'Logout' link is visible in the bottom left corner of the sidebar.

6. หน้าสำหรับสร้างการจ่ายเงินใหม่



The screenshot shows the same 'Paylog List' interface as before, but with a modal form titled 'Create new payout' open in the center. The modal form contains the following fields:

- * Employee ID:
- * Month: (with a calendar icon)
- Amount:
- Remark:

At the bottom of the modal, there are 'Cancel' and 'OK' buttons. The background of the interface is dimmed.

7. หน้าสำหรับการอัปเดตสถานการณ์จ่ายเงินเดือน

The screenshot shows a web application interface for managing payroll. A modal dialog titled "Update payroll status" is open, allowing a user to update the status of a payroll entry. The dialog contains a "Status" dropdown menu and a "Remark" text input field. Below these fields are "Cancel" and "OK" buttons. The background shows the employee profile for "Thitiphath Jenwichuwong - #18" and a table of payroll entries.

| Timestamp | Status | Amount | Remark |
|--------------------------|---------|--------|-------------|
| May 9, 2021, 12:29:43 AM | | | |
| May 9, 2021, 12:33:01 AM | | | Test Update |
| May 9, 2021, 12:33:06 AM | PAYMENT | 1 | Test Update |
| May 9, 2021, 12:33:11 AM | DONE | 1 | Test Update |

8. หน้าสำหรับอัปเดตเงินเดือนในกรณีที่เงินเดือนมีการเปลี่ยนแปลง โดยจะยังคงแสดงข้อมูลเงินเดือนก่อนหน้านี้จะถูกอัปเดตเอาไว้

The screenshot shows a web application interface for managing employee salaries. A modal dialog titled "Update Salary" is open, allowing a user to update the salary of an employee. The dialog contains a "Salary" text input field and a "Role" dropdown menu. Below these fields are "Cancel" and "OK" buttons. The background shows the employee profile for "Thitiphath Jenwichuwong" and a table of employee salary history.

| Timestamp | Salary | Role | Auditor |
|------------------------------|--------|------|---------|
| Wed, May 5, 2021, 2:59:30 AM | 500000 | CEO | 1 |
| Wed, May 5, 2021, 2:55:40 AM | 300000 | CEO | 1 |
| Wed, May 5, 2021, 2:55:24 AM | 150000 | CEO | 1 |
| Tue, May 4, 2021, 3:45:46 PM | 70000 | CFO | 1 |

9. หากต้องการไล่ Employee ออกให้กดปุ่ม “Fired this employee”

