

คู่มือมาตรฐานการพัฒนาซอฟต์แวร์

Software Development Standards (SDS)

เวอร์ชัน 2.2.0

ทีมพัฒนาซอฟต์แวร์ระบบ ทีมที่ 4

System Development Team 4

คณะวิทยาการสารสนเทศ มหาวิทยาลัยบูรพา



ทีมบริหารจัดการคุณภาพ (Quality Management)

ปรับปรุงล่าสุด พฤศจิกายน 2564

สารบัญ

	หน้า
ส่วนที่ 1 มาตรฐานการเขียนโปรแกรม (Coding Standards).....	1
1 การตั้งชื่อไฟล์และคลาส.....	1
1.1 การตั้งชื่อไฟล์และคลาสของ Controller	1
1.2 การตั้งชื่อไฟล์และคลาสของ Model	2
1.3 การตั้งชื่อไฟล์ View	3
2 การตั้งชื่อฟังก์ชัน.....	3
2.2 การตั้งชื่อฟังก์ชันของ Model	5
3 การตั้งชื่อตัวแปร	7
3.1 ตัวแปรแทน Object ของ Model	7
3.2 ตัวแปรที่รับค่ามาจากฐานข้อมูล.....	7
3.3 ตัวแปรสำหรับรับค่าจาก Fetch Array และ Object.....	8
3.4 ตัวแปร Array	8
3.5 ตัวแปร JSON	8
3.6 ตัวแปรนับรอบของลูป	8
4 การจัดทำมาตรฐานเกี่ยวกับฐานข้อมูล	9
4.1 การตั้งชื่อฐานข้อมูล.....	9
4.2 การตั้งชื่อตาราง.....	9
4.3 การตั้งชื่อฟิลด์.....	9
4.4 การเขียน Comment ของตารางและฟิลด์.....	10

5 การตั้งชื่อตัวแปรของ Config.....	11
6 การตั้งชื่อฟังก์ชันของ Helper	12
7 การเขียน Comments	13
7.1 Comment คลาสของ Controller และ Model.....	13
7.2 Comment ฟังก์ชันใน Controller, Model และ Helper	14
7.3 Comment ส่วนของ View.....	16
7.4 Comment บรรทัดเดียว หรือตัวแปรต่างๆ.....	18
7.5 Comment ระบุขอบเขตส่วนการทำงานนั้นๆ.....	18
ส่วนที่2 มาตรฐานส่วนติดต่อผู้ใช้ (UI Standards).....	20
1 การแสดงสีปุ่ม (Button Color)	20
2 การจัดวางตำแหน่งปุ่ม (Button Position)	21
3 การแสดงกล่องข้อความยืนยัน (Confirm Box)	22
4 การแสดงผลอื่นๆ	22

ส่วนที่ 1 มาตรฐานการเขียนโปรแกรม (Coding Standards)

มาตรฐานการเขียนโปรแกรมนี้เป็นมาตรฐานที่กำหนดขึ้นในบริบทของการพัฒนาระบบโดยใช้ CodeIgniter Framework ซึ่งเป็นสถาปัตยกรรมแบบ MVC (Model-View-Controller) โดยส่วนที่มีการกำหนดมาตรฐาน ประกอบด้วยไฟล์ในโฟลเดอร์ controllers, models, views, config และ helpers ดังนั้นจึงมีการกำหนดมาตรฐานการเขียนโปรแกรมแบ่งตามหัวเรื่อง และ MVC รวมถึง config, helpers และมาตรฐานเกี่ยวกับฐานข้อมูลดังนี้

1 การตั้งชื่อไฟล์และคลาส

เกี่ยวกับชื่อไฟล์ Controller, Model, View รวมทั้ง ชื่อคลาสของ Controller และ Model

1.1 การตั้งชื่อไฟล์และคลาสของ Controller

หลักการตั้งชื่อไฟล์และคลาส

1. ตั้งชื่อไฟล์ขึ้นต้นด้วยพินิจใหญ่เท่านั้น ถ้ามีหลายคำคั่นคำด้วยเครื่องหมายขีดล่าง (_) ได้เช่น Project.php
2. การตั้งชื่อคลาสต้องเป็นชื่อเดียวกันกับชื่อไฟล์และขึ้นต้นด้วยตัวอักษรพินิจใหญ่ เช่น Project
3. ควรตั้งชื่อตามโมดูล หรืองานของระบบนั้นๆ เช่น การจัดการหลักสูตร ควรตั้งชื่อว่า course_management
4. คอนโทรลเลอร์หลักของระบบ ควรตั้งชื่อด้วย ชื่อย่อระบบ_controller เช่น emeeting_controller
5. ส่วนของข้อมูลพื้นฐานของระบบ ควรตั้งชื่อลงท้ายด้วย _base เช่น ข้อมูลพื้นฐานของระบบ การจัดการประชุม ใช้ชื่อว่า meeting_base
6. ส่วนของรายงานของระบบ ควรตั้งชื่อลงท้ายด้วย _report เช่น project_report

7. คอนโทรลเลอร์สำหรับการเชื่อมโยงข้อมูลกับระบบสารสนเทศอื่น ควรตั้งชื่อด้วย ชื่อระบบหรือ งาน_service เช่น emeeting_service
8. คอนโทรลเลอร์สำหรับการรับ - ส่งค่าในรูปแบบของ AJAX ควรตั้งชื่อด้วยชื่อระบบหรือ งาน_ajax เช่น emeeting_ajax

ข้อยกเว้น

1. กรณีที่ต้องใช้ตัวเลขเป็นส่วนหนึ่งของชื่อไฟล์หรือคลาส ให้ตั้งอยู่ตำแหน่งท้ายสุด เช่น section1, section2 เป็นต้น
2. กรณีสร้างโฟลเดอร์ย่อย base, report, service หรือ ajax ไม่ต้องตั้งชื่อลงท้ายด้วย _base, _report, _service หรือ _ajax ตามลำดับ

ข้อห้าม

1. ห้ามตั้งชื่อขึ้นต้นด้วย c_ เช่น c_project, con_project, controller_project
2. ห้ามตั้งชื่อขึ้นต้นด้วย ชื่อระบบ_ ยกเว้นคอนโทรลเลอร์หลักของระบบเท่านั้น

1.2 การตั้งชื่อไฟล์และคลาสของ Model

หลักการตั้งชื่อไฟล์และคลาส

1. โมเดลต้องประกอบด้วย 2 ไฟล์คือ Da และ M
2. ตั้งชื่อไฟล์ขึ้นต้นด้วยพิมพ์ใหญ่เท่านั้น ถ้ามีหลายคำคั่นคำด้วยเครื่องหมายขีดล่าง (_) เช่น Da_ppm_project.php, M_ppm_project.php
3. การตั้งชื่อคลาสต้องเป็นชื่อเดียวกันกับชื่อไฟล์และขึ้นต้นด้วยตัวอักษรพิมพ์ใหญ่ เช่น Da_ppm_project, M_ppm_project
4. ควรตั้งชื่อตามชื่อตารางในฐานข้อมูลเท่านั้น
5. โมเดลหลักของระบบ ควรตั้งชื่อด้วย ชื่อระบบ_model เช่น emt_model

1.3 การตั้งชื่อไฟล์ View

หลักการตั้งชื่อไฟล์

1. ชื่อไฟล์ต้องขึ้นต้นด้วย v_ เช่น v_project.php
2. ตั้งชื่อไฟล์ด้วยตัวอักษรพิมพ์เล็กเท่านั้น และคั่นคำด้วยเครื่องหมายขีดล่าง (_) ได้เช่น v_project_detail.php
3. ชื่อ View ต้องสอดคล้องกับชื่อฟังก์ชัน หรือคลาส เช่น ฟังก์ชันชื่อ projecttype_input ควร ตั้งชื่อไฟล์ว่า v_projecttype_input.php

ข้อยกเว้น

1. กรณี 1 ฟังก์ชันเรียกมากกว่า 1 view ให้ขยายชื่อไฟล์จากชื่อเดิมได้ เช่น ฟังก์ชันชื่อ project_input() เรียกไฟล์ v_project_input_section1.php และ v_project_input_section2.php
2. กรณี 1 view ถูกเรียกจากหลายฟังก์ชัน ให้ตั้งชื่อตามชื่อฟังก์ชันแรกที่ใช้เรียกใช้

2 การตั้งชื่อฟังก์ชัน

หลักการตั้งชื่อฟังก์ชัน

1. ตั้งชื่อฟังก์ชันด้วยตัวอักษรพิมพ์เล็กเท่านั้น และคั่นคำด้วยเครื่องหมายขีดล่าง (_) ได้เช่น project_input()

หมวดของฟังก์ชัน

1. ฟังก์ชันสำหรับแสดงหน้าจอการบันทึกหรือแก้ไขข้อมูล

- 1.1 หน้าจอบันทึกข้อมูลอย่างเดียว หรือทั้งบันทึกและแก้ไข ตั้งชื่อลงท้ายด้วย `_input` หรือ ตั้งชื่อเป็น `input` เช่น `projecttype_input()`, `input()`
- 1.2 หน้าจอการแก้ไขข้อมูลอย่างเดียว ตั้งชื่อลงท้ายด้วย `_edit` หรือตั้งชื่อเป็น `edit` เช่น `projecttype_edit()`, `edit()`
2. ฟังก์ชันสำหรับการบันทึกหรือแก้ไขฐานข้อมูล
 - 2.1 สำหรับบันทึกอย่างเดียว ตั้งชื่อลงท้ายด้วย `_insert` หรือตั้งชื่อเป็น `insert` เช่น `projecttype_insert()`, `insert()`
 - 2.2 สำหรับแก้ไขอย่างเดียว ตั้งชื่อลงท้ายด้วย `_update` หรือตั้งชื่อเป็น `update` เช่น `projecttype_update()`, `update()`
 - 2.3 สำหรับบันทึกและแก้ไข ตั้งชื่อลงท้ายด้วย `_save` หรือตั้งชื่อเป็น `save` เช่น `projecttype_save()`, `save()`
3. ฟังก์ชันสำหรับการลบข้อมูลในฐานข้อมูล ตั้งชื่อลงท้ายด้วย `_delete` หรือตั้งชื่อเป็น `delete` เช่น `projecttype_delete()`, `delete()`
4. ฟังก์ชันสำหรับการแสดงผล
 - 4.1 หน้าหลักสำหรับแสดงข้อมูล ตั้งชื่อลงท้ายด้วย `_show` หรือตั้งชื่อเป็น `show` เช่น `projectlist_show()`, `show()`
 - 4.2 หน้าสำหรับแสดงข้อมูลแบบลงรายละเอียด ตั้งชื่อลงท้ายด้วย `_detail` หรือตั้งชื่อเป็น `detail` เช่น `projectlist_detail()`, `detail()`
5. ฟังก์ชันสำหรับการนำเข้าและอ่านข้อมูลจากไฟล์โดยเฉพาะ (ไฟล์ Excel) ตั้งชื่อลงท้ายด้วย `_import` หรือตั้งชื่อเป็น `import` เช่น `person_import()`, `import()`
6. ฟังก์ชันสำหรับส่งออกข้อมูลในรูปแบบต่างๆ
 - 6.1 ส่งออกข้อมูลรูปแบบไฟล์ Excel ตั้งชื่อลงท้ายด้วย `_excel` หรือตั้งชื่อเป็น `excel` เช่น `actionplan_excel()`, `excel()`

- 6.2 ส่งออกข้อมูลรูปแบบไฟล์ Word ตั้งชื่อลงท้ายด้วย `_word` หรือตั้งชื่อเป็น word เช่น `actionplan_word()`, `word()`
- 6.3 ส่งออกข้อมูลรูปแบบไฟล์ PDF ตั้งชื่อลงท้ายด้วย `_pdf` หรือตั้งชื่อเป็น pdf เช่น `actionplan_pdf()`, `pdf()`
- 6.4 ส่งออกข้อมูลรูปแบบตัวอย่างก่อนพิมพ์ตั้งชื่อลงท้ายด้วย `_print` หรือตั้งชื่อเป็น print เช่น `actionplan_print()`, `print()`
- 6.5 ส่งออกข้อมูลหลายรูปแบบในฟังก์ชันเดียว ตั้งชื่อลงท้ายด้วย `_export` หรือตั้งชื่อเป็น export เช่น `actionplan_export()`, `export()`
7. ฟังก์ชันสำหรับแสดง Popup ตั้งชื่อลงท้ายด้วย `_popup` หรือตั้งชื่อเป็น popup เช่น `projecttype_insert_popup()`, `projecttype_save_popup()`, `popup()`
8. ฟังก์ชันสำหรับการเชื่อมโยงข้อมูลกับระบบสารสนเทศที่เกี่ยวข้อง
 - 8.1 การรับข้อมูล ตั้งชื่อขึ้นต้นด้วย `get_service_` หรือตั้งชื่อเป็น `get_service` เช่น `get_service_person`, `get_service()`
 - 8.2 การส่งข้อมูล ตั้งชื่อขึ้นต้นด้วย `post_service_` หรือตั้งชื่อเป็น `post_service` เช่น `post_service_person`, `post_service()`
9. ฟังก์ชันสำหรับรับ - ส่งค่าในรูปแบบ AJAX ตั้งชื่อลงท้ายด้วย `_ajax` หรือตั้งชื่อเป็น ajax เช่น `projecttype_ajax`, `ajax()`

2.2 การตั้งชื่อฟังก์ชันของ Model

ฟังก์ชันในไฟล์ Da

1. ประกอบด้วย ฟังก์ชันหลัก 4 ฟังก์ชัน เท่านั้น ได้แก่ `insert()`, `update()`, `delete()`, `get_by_key()`
2. ฟังก์ชันสำหรับอัปเดตบางฟิลด์

- 2.1 กรณีอัปเดต 1 - 2 ฟิลด์ให้ตั้งชื่อว่า update_ชื่อฟิลด์ที่ต้องการอัปเดต เช่น update_firstname(), update_prefix_firstname()
- 2.2 กรณีอัปเดตมากกว่า 2 ฟิลด์ให้ตั้งชื่อว่า update_ชื่อการทำงานนั้นๆ เช่น update_person_retire() คือการอัปเดตฟิลด์สถานะของบุคลากรและวันที่ออก
3. ฟังก์ชันสำหรับลบ โดยไม่อ้างอิง PK
 - 3.1 ให้ตั้งชื่อว่า delete_by_ชื่อฟิลด์ เช่น delete_by_dept_id(), delete_by_dept_id_and_pos_id()

ฟังก์ชันในไฟล์ M

1. ฟังก์ชันอื่นๆ นอกเหนือจากฟังก์ชันในไฟล์ Da เช่น การคิวรีข้อมูลต่างๆ
2. ตั้งชื่อฟังก์ชันด้วยตัวอักษรพิมพ์เล็กเท่านั้น และคั่นคำด้วยเครื่องหมายขีดล่าง (_) ได้เช่น get_all()
3. โครงสร้างของชื่อฟังก์ชัน action_data_by_condition(for_something)
 - 3.1 action คือ การกระทำ ตัวอย่างเช่น get, search, count, update
 - 3.2 data คือ ข้อมูลที่ต้องการ ตัวอย่างเช่น project, projectname, projecttype
 - 3.3 by_condition คือ เงื่อนไขการค้นหา เช่น by_pjid, by_pjname
 - 3.4 for_something คือ ถูกเรียกใช้เพื่อฟังก์ชัน โมดูล หรือเงื่อนไข โดยเฉพาะ (ถ้าสำคัญ) เช่น for_mission, for_ajax

หมวดของฟังก์ชัน

1. ฟังก์ชันสำหรับคิวรีดึงข้อมูล
 - 1.1 สำหรับดึงข้อมูลทั่วไป ไม่มีการค้นหา หรือค้นหาแบบมีเงื่อนไขไม่ซับซ้อน ได้แก่ดึง ข้อมูลทั้งหมด (get_all) ข้อมูลที่ขึ้นต่อกัน เช่น

สาขาขึ้นอยู่กับคณะที่เลือก เป็นต้น ให้ ขึ้นต้นด้วย get_ เช่น
get_project()

1.2 สำหรับดึงข้อมูลที่มีการค้นหาแบบมีเงื่อนไขที่ซับซ้อน ให้ขึ้นต้นด้วย
search_ เช่น search_cousestr_by_csname_and_dpid

2. ฟังก์ชันสำหรับคิวรีโดยเรียกใช้ SQL function

2.1 ให้ตั้งชื่อขึ้นต้นด้วย SQL function เช่น count_person(),
sum_salary(), max_salary(), avg_salary()

3. ฟังก์ชันสำหรับคิวรีเพื่อตรวจสอบข้อมูล

3.1 สำหรับ return ค่า เป็น binary เช่น 0,1 TRUE, FALSE Y,N ให้ตั้ง
ชื่อขึ้นต้นด้วย check เช่น check_active_person()

3 การตั้งชื่อตัวแปร

3.1 ตัวแปรแทน Object ของ Model

ขึ้นต้นด้วย m ต่อด้วยชื่อย่อของตาราง เช่น M_cdms_customer ใช้ชื่อตัวแปร
ว่า m_cus

3.2 ตัวแปรที่รับค่ามาจากฐานข้อมูล

หลักการตั้งชื่อตัวแปร

1. ให้ตั้งชื่อตัวแปรเป็นตัวอักษรพิมพ์เล็กทั้งหมด
2. กรณีรับค่าหลาย record ให้ใช้ขึ้นต้นด้วย arr_ ชื่อย่อหรือชื่อเต็มของข้อมูล
เช่น \$arr_ps, \$arr_person
3. กรณีรับค่า record เดียว ให้ใช้ขึ้นต้นด้วย obj_ ชื่อย่อหรือชื่อเต็มของข้อมูล
เช่น \$obj_ps, \$obj_person
4. กรณีรับค่าฟิลด์เดียว หรือจากฟังก์ชันของ SQL ให้ตั้งชื่อให้สื่อความหมาย
เช่น รับค่าจาก ฟังก์ชัน sum_salary() ให้ตั้งชื่อว่า \$sum_salary

3.3 ตัวแปรสำหรับรับค่าจาก Fetch Array และ Object

ความแตกต่างระหว่าง array และ object และต้องตั้งชื่อตัวแปรรับค่าคนละแบบ

หลักการตั้งชื่อตัวแปร

1. ให้ตั้งชื่อตัวแปรด้วยตัวอักษรพิมพ์เล็กทั้งหมด
2. กรณีรับค่าจากการ Fetch array
3. ค่า Key ให้ตั้งชื่อว่า key_ข้อมูลนั้นๆ เช่น \$key_ps
4. ค่า Value ให้ตั้งชื่อว่า val_ข้อมูลนั้นๆ เช่น \$val_ps
5. กรณีรับค่าจากการ Fetch object ให้ตั้งชื่อว่า obj_ข้อมูลนั้นๆ เช่น \$obj_ps

3.4 ตัวแปร Array

ชื่อตัวแปรที่บ่งบอกว่าเป็นชุดของ Array ให้ขึ้นต้นด้วย arr ตัวอย่างเช่น \$arr_ps, \$arr_dp

3.5 ตัวแปร JSON

กรณีที่ต้องการส่งข้อมูลผ่านตัวแปรกลับมาในรูปแบบ JSON ให้ขึ้นต้นด้วย json ตัวอย่างเช่น json_data, json_message

3.6 ตัวแปรนับรอบของลูป

กรณีต้องการตั้งตัวแปรเพื่อใช้นับรอบของลูป สามารถใช้ตัวแปรในรูปแบบ Single ได้แต่ต้องสื่อ ความหมาย เช่น

1. ใช้ตัวแปร \$i เพื่อนับบรรทัดของลูป
2. ใช้ตัวแปร \$i, \$j, \$k หรือ \$m, \$n หรือ \$x, \$y, \$z ร่วมกัน กรณีมีลูปมากกว่า 1 ลูปได้ตามความเหมาะสม

4 การจัดทำมาตรฐานเกี่ยวกับฐานข้อมูล

4.1 การตั้งชื่อฐานข้อมูล

ข้อบังคับ

1. ต้องเป็นตัวอักษรพิมพ์เล็กทั้งหมด
2. ต้องคั่นด้วยเครื่องหมายขีดล่าง (_)

หลักการตั้งชื่อฐานข้อมูล

1. ชื่อย่อของระบบ เช่น hr, emt, spms
2. ลงท้ายด้วยคำว่า db ตัวอย่างเช่น hr_db, emt_db, buu_db

4.2 การตั้งชื่อตาราง

ข้อบังคับ

1. ต้องเป็นตัวอักษรพิมพ์เล็กทั้งหมด
2. ต้องคั่นด้วยเครื่องหมายขีดล่าง (_)

หลักการตั้งชื่อตาราง

1. ขึ้นต้นด้วยชื่อย่อของระบบในฐานข้อมูล ความยาวไม่เกิน 4 ตัวอักษร เช่น hr, emt, spms
2. ตามด้วยชื่อโมดูลการทำงาน หรือบ่งบอกว่าใช้เก็บข้อมูลนั้นๆ ตัวอย่างเช่น hr_person, hr_province, emt_agenda

4.3 การตั้งชื่อฟิลด์

ข้อบังคับ

1. ต้องเป็นตัวอักษรพิมพ์เล็กทั้งหมด
2. ต้องคั่นด้วยเครื่องหมายขีดล่าง (_)

หลักการตั้งชื่อฟิลด์

1. ต้องขึ้นต้นด้วยชื่อย่อของตาราง ความยาวไม่เกิน 4 ตัวอักษร (ไม่รวมชื่อฐานข้อมูล) เช่น ตาราง hr_person ชื่อย่อเป็น ps หรือตาราง hr_admin ชื่อย่อเป็น am เป็นต้น
2. หลังชื่อย่อของตาราง ให้ระบุชื่อฟิลด์นั้น ๆ โดยมีชื่อฟิลด์ที่ต้องบังคับใช้ในรูปแบบเดียวกัน ดังนี้
 - 2.1. ชื่อฟิลด์ที่เป็นคีย์หลัก ต้องลงท้ายด้วย id เช่น ps_id
 - 2.2. ชื่อฟิลด์ที่เป็นความหมายหรือข้อมูลหลักของตาราง ต้องลงท้ายด้วย name เช่น ps_name, ps_first_name, ps_last_name
 - 2.3. ชื่อฟิลด์ที่บ่งบอกถึงลำดับของข้อมูล ต้องลงท้ายด้วย seq เช่น am_seq, dp_seq
 - 2.4. ชื่อฟิลด์ที่บ่งบอกถึงลำดับชั้น ต้องลงท้ายด้วย level เช่น dp_level
 - 2.5. ชื่อฟิลด์ FK จากตารางอื่น ให้ใช้ชื่อเดิมมาต่อท้าย เช่น ps_pf_id, ps_dp_id

ข้อยกเว้น

1. ชื่อย่อของตารางมีความยาวมากกว่า 4 ตัวอักษรได้กรณีที่ไม่สามารถลดคำให้น้อยลงได้ (ถ้าจำเป็น)

4.4 การเขียน Comment ของตารางและฟิลด์

ทุกตารางและทุกฟิลด์ต้องมีการคอมเมนต์หรือนิยามความหมายกำกับไว้ให้ครบถ้วน ไม่มีข้อยกเว้น

หลักการเขียนคอมเมนต์

1. ตารางให้นิยามความหมายว่า ตารางเก็บข้อมูลอะไร หรือใช้สำหรับทำอะไร เช่น ตาราง emt_agenda คือ ตารางเก็บข้อมูลระเบียบวาระ

2. ฟิลด์ให้นิยามความหมายว่าใช้เก็บข้อมูลอะไร เช่น
 - 2.1. agd_id คือ รหัสระเบียบวาระ
 - 2.2. agd_name คือ ชื่อระเบียบวาระ
 - 2.3. agd_seq คือ ลำดับที่ในการแสดงผล
3. การระบุตัวอย่างของข้อมูล หากฟิลด์นั้นมีตัวอย่างของข้อมูลชัดเจน ให้ใส่ต่อท้ายใน เครื่องหมายวงเล็บด้วย เช่น
 - 3.1. agd_status คือ สถานะของระเบียบวาระ (Y=ใช้งาน, N=ไม่ใช้งาน)
4. ฟิลด์ที่อ้างอิงจากฟิลด์อื่น (FK) ให้อธิบายความหมายเดียวกันกับตารางตั้งต้น (PK) และต่อท้าย ด้วยว่ามาจากตารางไหนและ/หรือฐานข้อมูลไหน (ระบุชื่อฐานข้อมูลด้วย หากอยู่คนละ ฐานข้อมูล) เช่น
 - 4.1. agd_person_id คือ รหัสบุคลากร (ตาราง hr_person)
 - 4.2. agd_mt_id คือ รหัสการประชุมย่อย (ตาราง emt_meeting)

5 การตั้งชื่อตัวแปรของ Config

ข้อบังคับ

1. ต้องเป็นตัวอักษรพิมพ์เล็กทั้งหมด
2. ต้องคั่นด้วยเครื่องหมายขีดล่าง (_)

หลักการตั้งชื่อฟิลด์

1. ขึ้นต้นด้วยชื่อย่อของระบบ (สอดคล้องกับชื่อไฟล์คอนฟิก) เช่น hr, emt, spms
2. แล้วตามด้วยชื่อข้อมูลนั้นๆ ตัวอย่างเช่น
 - 2.1. โฟลเดอร์ของระบบ ใช้คำว่า folder เช่น \$config["hr_folder"], \$config["emt_folder"]

2.2. ที่อยู่ไฟล์ที่อัปโหลดของระบบ ใช้คำว่า upload_path เช่น

```
$config["hr_upload_path"]
```

2.3. ที่ตั้งไดเรกทอรีของระบบ ใช้คำว่า root_path เช่น

```
$config["hr_root_path"]
```

2.4. ชื่อฐานข้อมูลของระบบ ใช้คำว่า db_name เช่น

```
$config["hr_db_name"]
```

2.5. ที่อยู่รูปภาพต่างๆ ของระบบ ใช้คำว่า image_ ชื่อข้อมูลนั้นๆ เช่น

```
$config["hr_image_header"]
```

2.6. ที่อยู่ไอคอนต่างๆ ของระบบ ใช้คำว่า icon_ ชื่อข้อมูลนั้นๆ เช่น

```
$config["hr_icon_add"], $config["hr_icon_edit"],
```

```
$config["hr_icon_delete"]
```

ข้อควรระวัง

- ห้ามตั้งชื่อคอนฟิกร่วมกับระบบอื่น หากต้องการเรียกคอนฟิกรวบรวมอื่นสามารถเรียกใช้ได้ เลย (ถ้ามี) หรือหากต้องการตั้งชื่อคอนฟิกร่วมกับระบบอื่นเอง ให้ตั้งชื่อคอนฟิกร่วมกันด้วยชื่อ ระบบของตัวเองก่อนเสมอ เพื่อป้องกันการเขียนทับคอนฟิกรวบรวมอื่น เช่น ระบบบุคลากร ต้องการมีคอนฟิกร่วมชื่อ โฟลเดอร์ระบบ UMS ให้ตั้งชื่อว่า \$config["hr_ums_folder"] เป็นต้น

6 การตั้งชื่อฟังก์ชันของ Helper

ข้อบังคับ

- ต้องเป็นตัวอักษรพิมพ์เล็กทั้งหมด
- ต้องขึ้นด้วยเครื่องหมายขีดล่าง (_)

หลักการตั้งชื่อไฟล์

1. ชื่อฟังก์ชันการทำงานนั้นๆ ตัวอย่างเช่น ฟังก์ชันอ่านไฟล์ของระบบ ตัวอย่าง
`read_file()`

ข้อห้าม

1. ห้ามตั้งชื่อฟังก์ชันซ้ำกับ `function_helper` ที่มีอยู่แล้ว
2. ห้ามเพิ่ม/ลบ/แก้ไขเกี่ยวกับฟังก์ชันในไฟล์ `function_helper` โดยพลการ ยกเว้นมีข้อสรุป จากหัวหน้าทีมทุกทีมแล้ว
3. ห้ามตั้งชื่อฟังก์ชันซ้ำกับระบบอื่น หากต้องการเรียกฟังก์ชันจากระบบอื่น สามารถเรียกใช้ได้เลย (ถ้ามี) หรือหากต้องการคัดลอกฟังก์ชันจากระบบอื่น มาใช้ระบบตัวเอง ให้ตั้งชื่อฟังก์ชันขึ้นต้น ด้วยชื่อระบบของตัวเองก่อนเสมอ เพื่อป้องกันการเขียนทับ Helper ของระบบอื่น เช่น ระบบ บุคลากรต้องการมี ฟังก์ชันดึงปีงบประมาณปัจจุบันเหมือนระบบกำกับงบบฯ ให้ตั้งชื่อว่า `hr_get_bgpy()` เป็นต้น
4. หลีกเลี่ยงการใช้ตัวเลขในการตั้งชื่อฟังก์ชัน สำหรับฟังก์ชันที่การทำงาน คล้ายกัน แต่ต้องการตั้ง ชื่อให้แตกต่างกันโดยใช้ตัวเลข ควรตั้งชื่อฟังก์ชันให้ สู่ถึงการทำงาน

7 การเขียน Comments

7.1 Comment คลาสของ Controller และ Model

ในคลาสของ Controller และ Model ให้เขียนคอมเมนต์รูปแบบเดียวกัน

ข้อบังคับ

1. ให้เขียนคอมเมนต์คลาสกำกับในไฟล์คลาสทุกไฟล์ไม่มีข้อยกเว้น
2. เขียนคอมเมนต์คลาสไว้ข้างบนของ Class Controller
3. เขียนคอมเมนต์คลาสด้วยตัวอักษรภาษาอังกฤษเท่านั้น

หลักการเขียนคอมเมนต์คลาส

1. บรรทัดที่ 1 ใช้เครื่องหมายเปิดคอมเมนต์คือ /*
2. บรรทัดที่ 2 ระบุชื่อคลาส เช่น Baseposition
3. บรรทัดที่ 3 ระบุการทำงานของคลาสแบบคร่าวๆ เช่น Base Data of Position Management
4. บรรทัดที่ 4 ระบุชื่อผู้สร้างไฟล์คลาส หลังหัวข้อ @author เช่น @author Somchai
5. บรรทัดที่ 5 ระบุวันที่สร้างไฟล์คลาส ในรูปแบบ ปี พ.ศ. - เดือน - วัน หลังหัวข้อ @Create Date เช่น @Create Date 2558-10-26
6. บรรทัดที่ 6 ใช้เครื่องหมายปิดคอมเมนต์คือ */

หมายเหตุ :

1. แต่ละบรรทัด ให้ใส่เครื่องหมาย * เว้นวรรค 1 ครั้งนำหน้าเสมอ (ยกเว้นบรรทัดที่ 1 และ 6)
2. ข้อความคอมเมนต์หลังหัวข้อ @author ให้เคาะ tab 1 ครั้ง

ตัวอย่างการคอมเมนต์คลาส

```
/*
 * Baseposition
 * Base Data of Position Management
 * @author Somchai
 * @Create Date 2558-10-26
 */
```

7.2 Comment ฟังก์ชันใน Controller, Model และ Helper

ในฟังก์ชันของ Controller, Model และ Helper ให้เขียนคอมเมนต์รูปแบบเดียวกัน

ข้อบังคับ

1. ให้เขียนคอมเมนต์ฟังก์ชันกำกับทุกฟังก์ชัน ไม่มีข้อยกเว้น
2. เขียนคอมเมนต์ฟังก์ชันไว้ด้านบน ก่อนประกาศฟังก์ชันนั้น ๆ
3. เขียนคอมเมนต์ฟังก์ชันด้วยตัวอักษรภาษาอังกฤษเท่านั้น

หลักการเขียนคอมเมนต์ฟังก์ชัน

1. บรรทัดที่ 1 ใช้เครื่องหมายเปิดคอมเมนต์คือ /*
2. บรรทัดที่ 2 ระบุชื่อฟังก์ชัน เช่น position_insert
3. บรรทัดที่ 3 ระบุการทำงานของฟังก์ชันแบบคร่าวๆ เช่น Insert position in database after form add
4. บรรทัดที่ 4 ระบุข้อมูลเข้า (Input) หลังหัวข้อ @input กรณีไม่มีให้ใส่เครื่องหมายขีด (-) เช่น @input position_name_th, position_name_en, position_seq
5. บรรทัดที่ 5 ระบุข้อมูลที่ส่งกลับคืน (Output) หลังหัวข้อ @output กรณีไม่มีให้ใส่ เครื่องหมายขีด (-) เช่น @output The last insert id (pos_id)
6. บรรทัดที่ 6 ระบุชื่อผู้สร้างฟังก์ชัน หลังหัวข้อ @author เช่น @author Somchai
7. บรรทัดที่ 7 ระบุวันที่สร้างฟังก์ชัน ในรูปแบบ ปีพ.ศ. - เดือน - วัน หลังหัวข้อ @Create Date เช่น @Create Date 2558-10-26
8. บรรทัดที่ 8 ใช้เครื่องหมายปิดคอมเมนต์คือ */

หมายเหตุ :

1. แต่ละบรรทัด ให้ใส่เครื่องหมาย * เว้นวรรค 1 ครั้งนำหน้าเสมอ (ยกเว้นบรรทัดที่ 1 และ 8)

2. ข้อความคอมเมนต์หลังหัวข้อ @input, @output, @author ให้เคาะ tab 1 ครั้ง

ตัวอย่างการคอมเมนต์ฟังก์ชัน

```
/*
* position_insert
* Insert position in database after form add
* @input postion_name_th, postion_name_en, position_seq
* @output The last insert id (pos_id)
* @author Somchai
* @Create Date 2558-10-26
*/
```

7.3 Comment ส่วนของ View

ข้อบังคับ

1. ให้เขียนคอมเมนต์ส่วนของ View กำกับทุกไฟล์ไม่มีข้อยกเว้น
2. เขียนคอมเมนต์ส่วนของ View ไว้บรรทัดแรกของไฟล์
3. เขียนคอมเมนต์ส่วนของ View ด้วยตัวอักษรภาษาอังกฤษเท่านั้น
4. ระบุคอมเมนต์ส่วนของวิวให้อยู่ในรูปแบบของ PHP ยกเว้นไฟล์วิวเป็น HTML ให้ใช้การคอมเมนต์แบบ HTML ได้

หลักการเขียนคอมเมนต์ส่วนของ View

1. บรรทัดที่ 1 ใช้เครื่องหมายเปิดคอมเมนต์คือ /*
2. บรรทัดที่ 2 ระบุชื่อไฟล์วิว เช่น v_position_input
3. บรรทัดที่ 3 ระบุการทำงานของวิวแบบคร่าวๆ เช่น Display input form of position for add

4. บรรทัดที่ 4 ระบุข้อมูลเข้า (Input) หลังหัวข้อ @input กรณีไม่มีให้ใส่เครื่องหมายขีด (-) เช่น @input Array of headings (arr_head)
5. บรรทัดที่ 5 ระบุข้อมูลที่ส่งกลับคืน (Output) หลังหัวข้อ @output กรณีไม่มีให้ใส่ เครื่องหมายขีด (-) เช่น @output Input form of position
6. บรรทัดที่ 6 ระบุชื่อผู้สร้างฟังก์ชัน หลังหัวข้อ @author เช่น @author Somchai
7. บรรทัดที่ 7 ระบุวันที่สร้างฟังก์ชัน ในรูปแบบ ปีพ.ศ. - เดือน - วัน หลังหัวข้อ @Create Date เช่น @Create Date 2558-10-26
8. บรรทัดที่ 8 ใช้เครื่องหมายปิดคอมเมนต์คือ */

หมายเหตุ :

1. แต่ละบรรทัด ให้ใส่เครื่องหมาย * เว้นวรรค 1 ครั้งนำหน้าเสมอ (ยกเว้นบรรทัดที่ 1 และ 8)
2. ข้อความคอมเมนต์หลังหัวข้อ @input, @output, @author ให้เคาะ tab 1 ครั้ง

ตัวอย่างการคอมเมนต์ส่วนของ View

```
/*
* v_position_input
* Display input form of position for add
* @input Array of headings (arr_head)
* @output Input form of position
* @author Somchai
* @Create Date 2558-10-26
*/
```

7.4 Comment บรรทัดเดียว หรือตัวแปรต่างๆ

กรณีต้องการคอมเมนต์เพื่อนิยามความหมายของตัวแปร หรือส่วนการทำงาน บรรทัดนั้น ๆ หรือคอมเมนต์เพื่อระบุวันที่แก้ไข ผู้แก้ไข หรือหมายเหตุสำหรับกรณีที่มีการ ปรับแก้หรือเพิ่มเติมโปรแกรม (ไม่บังคับ)

หลักการเขียนคอมเมนต์

1. ให้คอมเมนต์ด้านบนก่อนประกาศตัวแปร หรือก่อนบรรทัดนั้นๆ
2. ขึ้นต้นด้วยเครื่องหมายคอมเมนต์คือ //
3. เว้นวรรค 1 ครั้ง ตามด้วยคอมเมนต์ที่ต้องการ โดยสามารถคอมเมนต์เป็น ภาษาไทยหรือ อังกฤษได้ตามความเหมาะสม

ตัวอย่างการคอมเมนต์

```
// ตั้งค่าเริ่มต้นของ i  
$i = 0;
```

7.5 Comment ระบุขอบเขตส่วนการทำงานนั้นๆ

กรณีต้องการคอมเมนต์ส่วนของการทำงานที่มีคำสั่งมากกว่า 1 บรรทัด เพื่อระบุ ขอบเขตการทำงานนั้น ๆ (ไม่บังคับ)

ข้อบังคับ

1. เขียนคอมเมนต์ด้วยตัวอักษรภาษาอังกฤษเท่านั้น
2. ต้องระบุคอมเมนต์ไว้ทั้ง 2 ส่วน คือ ส่วนบนและส่วนท้ายของการทำงาน นั้น ๆ

หลักการคอมเมนต์ส่วนบน

1. เปิด - ปิดคอมเมนต์ไว้ด้านบนก่อนเริ่มการทำงานนั้นๆ โดยใช้เครื่องหมายคอมเมนต์แบบ PHP คือ /* และ */ ตามลำดับ หรือเครื่องหมายคอมเมนต์ใน HTML ก็ได้
2. ส่วนของข้อความให้เว้นวรรค 1 ครั้ง แล้วขึ้นต้นด้วย Start แล้วตามด้วยอธิบายส่วนการทำงานนั้นๆ

หลักการคอมเมนต์ส่วนท้าย

1. เปิด - ปิดคอมเมนต์ไว้ด้านล่างสุดหลังการทำงานนั้นๆ โดยใช้เครื่องหมายคอมเมนต์แบบ PHP คือ /* และ */ ตามลำดับ หรือเครื่องหมายคอมเมนต์ใน HTML ก็ได้
2. ส่วนของข้อความให้เว้นวรรค 1 ครั้ง แล้วขึ้นต้นด้วย End แล้วตามด้วยอธิบายส่วนการทำงานนั้นๆ

ตัวอย่างการคอมเมนต์

```
/* Start Form input of position */
..
...
/* End Form input of position */
```

ส่วนที่2 มาตรฐานส่วนติดต่อผู้ใช้ (UI Standards)

มาตรฐานส่วนติดต่อผู้ใช้นี้ใช้ธีมเพลตแบบ Bootstrap Framework เป็นต้นแบบในการกำหนดมาตรฐาน ซึ่งเป็นรูปแบบของธีมเพลตที่ใช้พัฒนากันในหลายระบบ โดยรูปแบบของการแสดงผลจะมีลักษณะเหมือนกัน ดังนั้น จึงจัดทำมาตรฐานนี้ขึ้น เพื่อให้การแสดงผลส่วนติดต่อผู้ใช้ (User Interface) เป็นไปตามมาตรฐานเดียวกัน

1 การแสดงสีปุ่ม (Button Color)

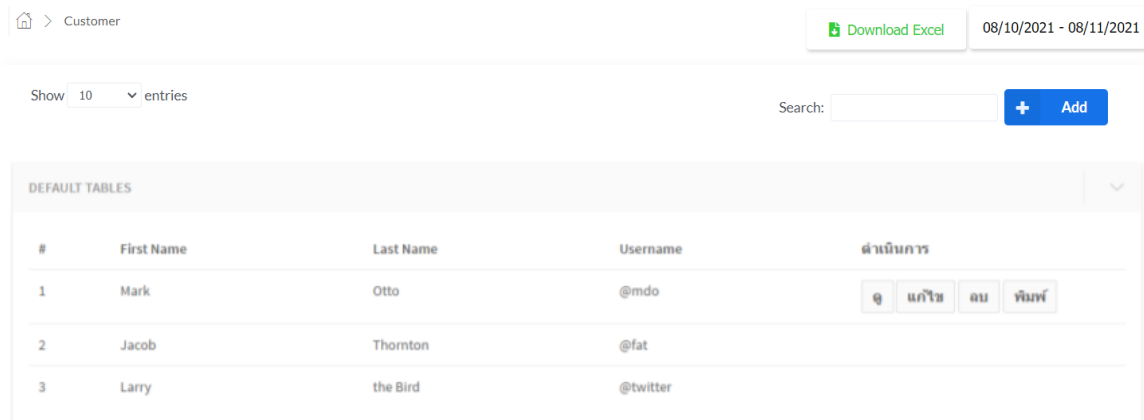
Operations

บันทึก	Save, Normal	พิมพ์	download	Export
แก้ไข	Warning	ยกเลิก	เคลียร์	Clear, Cancel, Back, Close
ลบ	Denger	เพิ่ม		Others

รูปภาพที่ 2-1 แสดงรายการปุ่ม และสีปุ่ม (Button Color)

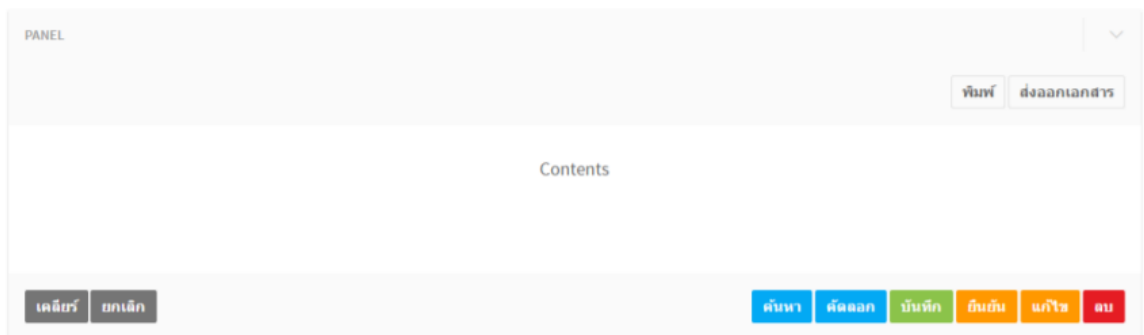
- ปุ่มบันทึก แสดงเป็น สีเขียว
- ปุ่มยืนยัน, แก้ไข แสดงเป็น สีส้ม
- ปุ่มลบ แสดงเป็น สีแดง
- ปุ่มพิมพ์, ส่งออกเอกสาร แสดงเป็น สีขาว
- ปุ่มเคลียร์, ยกเลิก แสดงเป็น สีเทา
- ปุ่มเพิ่ม และอื่นๆ แสดงเป็น สีนํ้าเงิน, ฟ้า

2 การจัดวางตำแหน่งปุ่ม (Button Position)



รูปภาพที่ 2-2 แสดงการจัดวางตำแหน่งปุ่ม (Button Position)

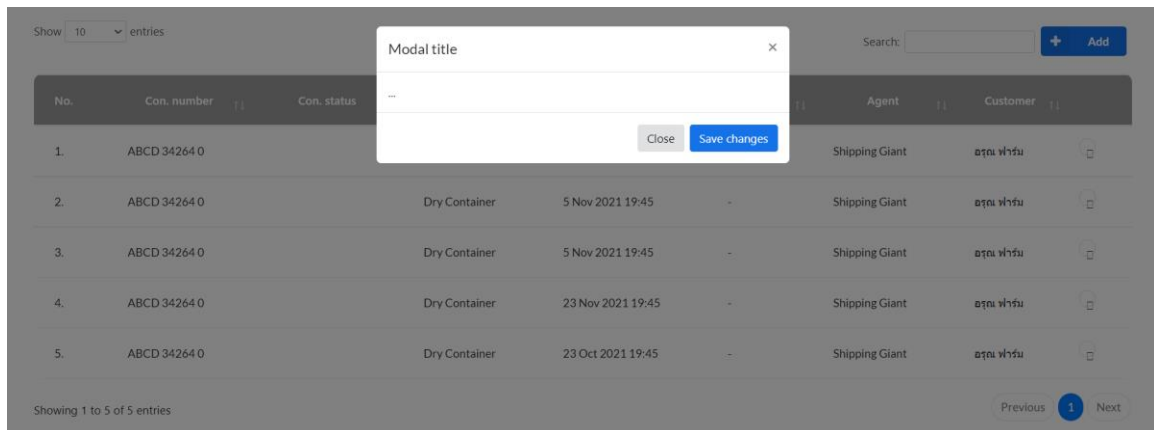
- ปุ่มพิมพ์, ส่งออกเอกสาร จัดวางตำแหน่งขวาบน
- ปุ่มเพิ่ม จัดวางตำแหน่งขวาบน
- ไอคอน หรือรูปภาพแทนการดำเนินการ จัดเรียงตามลำดับดังรูปภาพที่ 2-2



รูปภาพที่ 2-3 แสดงการจัดวางตำแหน่งปุ่ม (Button Position)

- ปุ่มพิมพ์, ส่งออกเอกสาร จัดวางตำแหน่งขวาบน
- ปุ่มเคลียร์, ยกเลิก จัดวางตำแหน่งซ้ายล่าง
- ปุ่มค้นหา, คัดลอก, บันทึก, ยืนยัน, แก้ไข, ลบ จัดวางตำแหน่งขวาล่าง

3 การแสดงกล่องข้อความยืนยัน (Confirm Box)



- ปุ่มตกลง แสดงเป็นสีน้ำเงิน, ฟังก์ชัน จัดวางตำแหน่งขวาล่าง
- ปุ่มยกเลิก แสดงเป็นสีเทา จัดวางตำแหน่งซ้ายล่าง
- กรณีปุ่มตกลง สามารถแสดงเป็นปุ่มบันทึก, ยืนยัน, แก้ไข, ลบ หรือปุ่มดำเนินการอื่นๆ ที่ต้องการใช้ให้ สอดคล้องกับงาน โดยแสดงสีปุ่มตามดังรูปภาพที่ 2-1

4 การแสดงผลอื่นๆ

ประเด็นเกี่ยวกับมาตรฐานส่วนติดต่อผู้ใช้ที่ต้องกำหนดรูปแบบกันภายในทีมพัฒนา ให้เป็นมาตรฐานเดียวกัน ภายในระบบ หรือไซด์เดียวกัน มีดังนี้

4.1. การแสดงข้อความแจ้งเตือน Form Validation รูปแบบการแสดงผล
แล้วแต่ธีมเพลต แต่ควรเป็น รูปแบบเดียวกันทั้งระบบ หรือไซด์

4.2. การแสดง Tooltip ใช้สำหรับอธิบายรายละเอียดของปุ่ม (button) หรือลิงค์ (link) ซึ่งควรมีการอธิบาย รายละเอียดให้ชัดเจน กรณีต้องการคำอธิบายเพิ่มเติม และรูปแบบการแสดงผลแล้วแต่ธีมเพลต แต่ควร เป็นรูปแบบเดียวกันทั้งระบบ หรือไซด์

4.3. การแสดง Placeholder ใช้สำหรับอธิบายการกรอกข้อมูลในฟิลด์โดยแสดงเป็นข้อความพื้นหลังในฟิลด์ ซึ่งควรกำหนดว่าให้แสดงข้อความเป็นตัวอย่างข้อมูล ชื่อฟิลด์ข้อมูล หรืออื่นๆ โดยขึ้นอยู่กับความ เหมาะสมของงาน

- 4.4. การแสดง Icon หรือรูปภาพแทนการดำเนินการ เลือกใช้ตามรูปแบบ และ
ความเหมาะสมของธีมเพลต แต่ควรเป็นรูปแบบเดียวกันทั้งระบบ หรือไซต์
- 4.5. การแสดง Datepicker และรูปแบบวันที่ (Date Format) ควรเป็นรูปแบบ
เดียวกันทั้งระบบ หรือไซต์