คู่มือมาตรฐานการพัฒนาซอฟต์แวร์

Software Development Standards (SDS)

เวอร์ชัน 2.2.4

ทีม 1

สาขาวิชาวิศวกรรมซอฟต์แวร์ Software Engineering

คณะวิทยาการสารสนเทศ มหาวิทยาลัยบูรพา

การควบคุมการเปลี่ยนแปลงเอกสาร

ตารางที่ 0-1 การควบคุมการเปลี่ยนแปลงเอกสาร

เวอร์ชันปัจจุบัน		2.2.4		
วันที่แก้ไขเวอร์ชันล่าสุด		กันยายน 2564		
ผู้จัดทำเอกสารนี้		ฝ่ายพัฒนาซอฟต์แวร์ทีม 1		
เวอร์ชัน	**วันที่**	**รายการที่แก้ไข**	**ผู้แก้ไข**	**หมายเหตุ**
1.0.0	30 สิงหาคม 2563	สร้างเทมเพลต	ธนเกียรติ หอมหวล	ต้นฉบับ
2.0.0	20 สิงหาคม 2564	สร้างเทมเพลต และแก้ไขเนื้อหา	พงศ์ธร ,ชัญญาพัชญ์	
2.1.0	9 กันยายน 2564	ปรับแก้ แพลตฟอร์ม	อภิญญา	
2.2.1	15 กันยายน 2564	ปรับแก้เนื้อหา	ชัญญาพัชญ์ , อภิญญา	
2.2.2	17 กันยายน 2564	ปรับแก้เนื้อหา	อภิญญา	
2.2.3	18 กันยายน 2564	ปรับแก้เนื้อหา	อภิญญา	
2.2.4	20 กันยายน 2564	แก้ไขคำผิด	อภิญญา	

สารบัญ

หน้า

สารบัญรูปภาพ

ภาพที่ หน้า

สารบัญตาราง

ตารางที่ หน้า

ส่วนที่ 1 มาตรฐานการเขียนโปรแกรม (Coding (Standards)

มาตรฐานการเขียนโปรแกรมนี้เป็นมาตรฐานที่กำหนดขึ้นในบริบทของการพัฒนาระบบโดยใช้ CodeIgniter Framework ซึ่งเป็นสถาปัตยกรรมแบบ MVC (Model View และ Controller) โดยส่วนที่มีการกำหนดมาตรฐาน ประกอบด้วยไฟล์ในโฟลเดอร์ Controller, Model, View, Config และ Helpers ดังนั้นจึงมีการกำหนดมาตรฐานการเขียนโปรแกรมแบ่งตามหัวเรื่องและ MVC รวมถึง Config, helpers และมาตรฐานเกี่ยวกับฐานข้อมูล ดังนี้

1. การตั้งชื่อไฟล์และคลาส

เกี่ยวกับชื่อไฟล์ Controller, Model และ View รวมทั้งชื่อคลาสของ Controller และ Model

1.1. การตั้งชื่อไฟล์และคลาสของ Controller

หลักการตั้งชื่อไฟล์และคลาส

1.1.1 ตั้งชื่อไฟล์ด้วยอักษรพิมพ์เล็กเท่านั้น และคั่นคำด้วยเครื่องหมายขีดล่าง (_) เช่น project.php

1.1.2 การตั้งชื่อคลาสต้องเป็นชื่อเดียวกันกับไฟล์ และขึ้นต้นด้วยอักษรพิมพ์ใหญ่เช่น Project

1.1.3 ควรตั้งชื่อตามโมดูล หรืองานของระบบนั้นๆ เช่น การจัดการบัญชีผู้ใช้งาน ควรตั้งชื่อ _management

1.1.4 ส่วนของข้อมูลพื้นฐานของระบบ ควรตั้งชื่อลงท้ายด้วย _base เช่น ข้อมูลพื้นฐานของระบบการจัดการประชุม ใช้ชื่อว่า M_base

1.1.5 ส่วนของรายงานของระบบ ควรตั้งชื่อลงท้ายด้วย _report เช่น project_report

1.1.6 คอนโทรลเลอร์สำหรับการเชื่อมโยงข้อมูลกับระบบสารสนเทศอื่น ควรตั้งชื่อด้วยชื่อระบบหรืองาน_service เช่น login_service

1.1.7 คอนโทรลเลอร์สำหรับการรับ - ส่งค่าในรูปแบบของ AJAX ควรตั้งชื่อด้วย ชื่อระบบหรืองาน_ajax เช่น login_ajax

ข้อยกเว้น

1.1.8 กรณีที่ต้องใช้ตัวเลขเป็นส่วนหนึ่งของชื่อไฟล์หรือคลาส ให้ตั้งอยู่ตำแหน่งท้ายสุด เช่น section1, section2 เป็นต้น

1.1.9 กรณีสร้างโฟลเดอร์ย่อย base, report, service หรือ ajax ไม่ต้องตั้งชื่อลงท้ายด้วย _base, _report, _service หรือ _ajax ตามลำดับ

1.1.10 กรณี CodeIgniter เวอร์ชัน 3 ตั้งชื่อไฟล์ขึ้นต้นด้วยตัวอักษรพิมพ์ใหญ่

ข้อห้าม

1.1.11 ห้ามตั้งชื่อขึ้นต้นด้วยตัวอักษร c เช่น c_index, con_index และ controller_index

1.1.12 ห้ามตั้งชื่อขั้นต้นด้วย ชื่อระบบ_ ยกเว้นคอนโทรลเลอร์หลักของระบบเท่านั้น

1.2. การตั้งชื่อไฟล์และคลาสของ Model

หลักการตั้งชื่อไฟล์และคลาส

1.2.1 โมเดลต้องประกอบด้วย 2 ไฟล์คือ da และ m

1.2.2 ตั้งชื่อไฟล์ด้วยตัวอักษรพิมพ์เล็กเท่านั้น และคั่นคำด้วยเครื่องหมายขีดล่าง (_) ได้เช่น da_dqs_project.php, m_dqs_project.php

1.2.3 การตั้งชื่อคลาสต้องเป็นชื่อเดียวกันกับชื่อไฟล์และขึ้นต้นด้วยตัวอักษรพิมพ์ใหญ่ เช่น Da_dqs_project, M_dqs_project

1.2.4 ควรตั้งชื่อตามชื่อตารางในฐานข้อมูลเท่านั้น

1.2.5 โมเดลหลักของระบบ ควรตั้งชื่อด้วย ชื่อระบบ_model เช่น dqs_model

ข้อยกเว้น

1.2.6 กรณี CodeIgniter เวอร์ชัน 3 ตั้งชื่อไฟล์ขึ้นต้นด้วยตัวอักษรพิมพ์ใหญ่

1.3. การตั้งชื่อไฟล์ View

หลักการตั้งชื่อไฟล์

1.3.1 ตั้งชื่อไฟล์ด้วยตัวอักษรพิมพ์เล็กเท่านั้น และคั่นคำด้วยเครื่องหมายขีด (-) ในกรณีที่ชื่อมีไฟล์มีหลายพยางค์ เช่น modal-add-member.blade.php

1.3.2 ชื่อไฟล์ View ที่เป็นการทำงานของหน้าหลักควรตั้งชื่อไฟล์ว่า index.blade.php

1.3.3 ชื่อไฟล์ต้องขึ้นต้นด้วย v_ เช่น v_project.php

2. การตั้งชื่อฟังก์ชัน

เกี่ยวกับฟังก์ชันของ Controller และ Model

2.1. การตั้งชื่อฟังก์ชัน Controller

หลักการตั้งชื่อฟังก์ชัน

2.1.1 ตั้งชื่อฟังก์ชันด้วยตัวอักษรพิมพ์เล็กเท่านั้น และคั่นคำด้วยเครื่องหมายขีดล่าง (_) ได้เช่น project_input()

หมวดของฟังก์ชัน

2.1.2 ฟังก์ชันสำหรับแสดงหน้าจอการบันทึกข้อมูลหรือแก้ไขข้อมูล

1) หน้าจอการบันทึกข้อมูลอย่างเดียว หรือทั้งบันทึกและแก้ไข ตั้งชื่อลงท้ายด้วย _input หรือตั้งชื่อเป็น input เช่น projecttype_input(), input()

2) หน้าจอการแก้ไขข้อมูลอย่างเดียว ตั้งชื่อลงท้ายด้วย _edit หรือตั้งชื่อเป็น edit เช่น projecttype_edit(), edit()

2.1.3 ฟังก์ชันสำหรับการบันทึกหรือแก้ไขลงฐานข้อมูล

1) สำหรับบันทึกอย่างเดียว ตั้งชื่อลงท้ายด้วย _insert หรือตั้งชื่อเป็น insert เช่น projecttype_insert(), insert()

2) สำหรับการแก้ไขอย่างเดียว ตั้งชื่อลงท้ายด้วย _update หรือตั้งชื่อเป็น update เช่น projecttype_update(), update()

3) สำหรับบันทึกและแก้ไข ตั้งชื่อลงท้ายด้วย _save หรือตั้งชื่อเป็น save เช่น projecttype_save(), save()

2.1.4 ฟังก์ชันสำหรับการลบข้อมูลในฐานข้อมูล ตั้งชื่อลงท้ายด้วย _delete หรือตั้งชื่อเป็น delete เช่น projecttype_delete(), delete()

2.1.5 ฟังก์ชันสำหรับการแสดงผล

1) หน้าหลักสำหรับแสดงข้อมูลตั้งชื่อลงท้ายด้วย _show หรือตั้งชื่อเป็น show เช่น projectlist_show(), show()

2) หน้าสำหรับแสดงข้อมูลแบบลงรายละเอียด ตั้งชื่อลงท้ายด้วย _detail หรือตั้งชื่อเป็น detail เช่น projectlist_detail(), deailt()

2.1.6 ฟังก์ชันสำหรับการนำเข้าและอ่านข้อมูลจากไฟล์ ตั้งชื่อลงท้ายด้วย _import หรือตั้งชื่อเป็น import เช่น person_import(), import()

2.1.7 ฟังก์ชันสำหรับส่งออกข้อมูลในรูปแบบต่าง ๆ

1) ส่งออกข้อมูลในรูปแบบไฟล์ Excel ตั้งชื่อลงท้ายด้วย _excel หรือตั้งชื่อเป็น excel เช่น plan_excel() , excel()
2) ส่งออกข้อมูลในรูปแบบไฟล์ pdf ตั้งชื่อลงท้ายด้วย _pdf หรือตั้งชื่อเป็น pdf เช่น plan_pdf(), pdf()
3) ส่งออกข้อมูลหลายรูปแบบในฟังก์ชันเดียว ตั้งชื่อลงท้ายด้วย _export หรือตั้งชื่อเป็น export เช่น plan_export(), export()

2.1.8 ฟังก์ชันสำหรับแสดง Popup ตั้งชื่อลงท้ายด้วย _popup หรือตั้งชื่อเป็น popup เช่น projecttype_insert_popup(), popup()

2.1.9 ฟังก์ชันสำหรับรับ - ส่งค่าในรูปแบบ AJAX ตั้งชื่อลงท้ายด้วย _ajax หรือตั้งชื่อเป็น ajax เช่น projecttype_ajax(),

2.2. การตั้งชื่อฟังก์ชัน Model

<u>ฟังก์ชันในไฟล์ da</u>

2.2.1 ประกอบด้วยฟังก์ชันหลัก 4 ฟังก์ชันเท่านั้น ได้แก่ insert(), update(), delete(), get_by_key()

<u>ฟังก์ชันในไฟล์ m</u>

2.2.2 ฟังก์ชันอื่น ๆ นอกเหนือจากฟังก์ชันในไฟล์ da เช่น หารคิวรี่ข้อมูลต่าง ๆ การอัปเดตบางฟิลด์ การลบโดยไม่อ้างคีย์หลัก เป็นต้น

2.2.3 ตั้งชื่อฟังก์ชันด้วยอักษรพิมพ์เล็กเท่านั้น และคั่นคำด้วยเครื่องหมายขีดล่าง (_) ได้ เช่น get_all()

2.2.4 โครงสร้างของชื่อฟังก์ชัน action_data_by_condition(for_something)

1) action คือ การกระทำ ตัวอย่างเช่น get, search, count, update
2) data คือ ข้อมูลที่ต้องการ ตัวอย่างเช่น project, projectname, projecttype
3) by_condition คือ เงื่อนไขการค้นหา เช่น by_id, by_name
4) for_something คือ ถูกเรียกใช้เพื่อฟังก์ชัน มอดูล หรือเงื่อนไขโดยเฉพาะ (ถ้าสำคัญ) เช่น for_mission, for_ajax

<u>หมวดของฟังก์ชัน</u>

2.2.5 ฟังก์ชันสำหรับคิวรี่ดึงข้อมูล

1) สำหรับดึงข้อมูลทั่วไป ไม่มีการค้นหา หรือค้นหาแบบมีเงื่อนไขไม่ซับซ้อน ได้แก่ ดึงข้อมูลทั้งหมด (get_all) ข้อมูลที่ขึ้นต่อกัน เช่น สาขาขึ้นอยู่กับคณะที่เลือก เป็นต้น ให้ขึ้นต้นด้วย get_ เช่น get_project()

2) สำหรับดึงข้อมูลที่มีการค้นหาแบบมีเงื่อนไขที่ซับซ้อน ให้ขึ้นต้นด้วย search_ เช่น search_cousestr_by_csname_and_dpid

2.2.6 ฟังก์ชันสำหรับคิวรี่โดยเรียกใช้ SQL function

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

2.2.7 ฟังก์ชันสำหรับคิวรี่เพื่อตรวจสอบข้อมูล

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

2.2.8 ฟังก์ชันสำหรับอัปเดตบางฟิลด์

1) กรณีอัปเดต 1 – 2 ฟิลด์ ให้ตั้งชื่อว่า update_ชื่อฟิลด์ที่ต้องการอัพเดต เช่น update_firstname(), update_prefix_firstname()

2) กรณีอัปเดตมากกว่า 2 ฟิลด์ ให้ตั้งชื่อว่า update_ชื่อการทำงานนั้น ๆ เช่น update_person_retire() คือ การอัปเดตฟิลด์สถานะของบุคลากร และวันที่ออก

2.2.9 ฟังก์ชันสำหรับลบ โดยไม่อ้างอิง PK

1) ให้ตั้งชื่อว่า delete_by_ชื่อฟิลด์ เช่น delete_by_dept_id(), delete_by_dept_id_and_pos_id()

3. การตั้งชื่อตัวแปร

3.1. ตัวแปรสำหรับรับค่าจาก Fetch Array และ Object

ความแตกต่างระหว่าง array และ object และต้องตั้งชื่อตัวแปรรับค่าคนละแบบ

<u>หลักการตั้งชื่อตัวแปร</u>

3.1.1 ให้ตั้งชื่อตัวแปรด้วยตัวอักษรพิมพ์เล็กทั้งหมด

3.1.2 กรณีรับค่าจากการ Fetch array

1) ค่า Key ให้ตั้งชื่อว่า key_ข้อมูลนั้นๆ เช่น $key_ps

2) ค่า Value ให้ตั้งชื่อว่า val_ข้อมูลนั้นๆ เช่น $val_ps

3.1.3 กรณีรับค่าจากการ Fetch object ให้ตั้งชื่อว่า row_ข้อมูลนั้นๆ เช่น $row_ps

3.2. ตัวแปรแทน Object ของ Model

ขึ้นต้นด้วย m ต่อด้วยชื่อย่อของตาราง เช่น m_hr_person ใช้ชื่อตัวแปรว่า mps

3.3. ตัวแปรที่รับค่ามาขากฐานข้อมูล

หลักการตั้งชื่อตัวแปร

3.3.1 ให้ตั้งชื่อตัวแปรเป็นตัวอักษรพิมพ์เล็กทั้งหมด

3.3.2 กรณีรับค่าหลาย record ให้ใช้ขึ้นต้นด้วย rs_ชื่อย่อหรือชื่อเต็มของข้อมูล เช่น $rs_ps, $rs_person

3.3.3 กรณีรับค่า record เดียว ให้ใช้ขึ้นต้นด้วย qu_ชื่อย่อหรือชื่อเต็มของข้อมูล เช่น $qu_ps,$qu_person

3.3.4 กรณีรับค่าฟิลด์เดียว หรือจากฟังก์ชันของ SQL ให้ตั้งชื่อให้สื่อความหมาย เช่น รับค่าจากฟังก์ชัน sum_salary() ให้ตั้งชื่อว่า $sum_salary

3.3.5 กรณีรับค่าเป็น Array ใช้สำหรับ drop down list ต้องขึ้นต้นด้วย opt_ เช่น $opt_provinc

3.4. ตัวแปรทั่วไปหรือ Array

3.4.1 ให้ตั้งชื่อตัวแปรด้วยรูปแบบ Camel Case

3.4.2 ใช้ชื่อตัวแปรให้ตรงกับข้อมูล เช่น $productAll และ $productDetail เป็นต้น

3.5. ตัวแปรนับรอบของลูป

3.5.1 ใช้ตัวแปร $i เพื่อนับบรรทัดของลูป

3.5.2 ใช้ตัวแปร $i, $j และ $k หรือ $x, $y และ $z ร่วมกัน กรณีมีลูป มากกว่า 1 ลูป ได้ตามความเหมาะสม

4. การจัดทำมาตรฐานเกี่ยวกับฐานข้อมูล

4.1. การตั้งชื่อฐานข้อมูล

ให้ตั้งชื่อฐานข้อมูลเป็นอักษรพิมพ์เล็กทั้งหมดตามด้วยเครื่องหมายขีดล่าง (_) ตัวอย่างเช่น hr_buu เป็นต้น

หลักการตั้งชื่อฐานข้อมูล

4.1.1 ขึ้นต้นด้วยชื่อย่อของไซต์ เช่น regis

4.1.2 ตามด้วยชื่อระบบ หรือชื่อย่อของระบบ เช่น dqs

4.1.3 ลงท้ายด้วยคำว่า db เช่น regis_dqs

4.2. การตั้งชื่อตาราง

ข้อบังคับ

4.2.1 ต้องเป็นตัวอักษรพิมพ์เล็กทั้งหมด

4.2.2 ต้องคั่นด้วยเครื่องหมายขีดล่าง (_) เป็นต้น

หลักการตั้งชื่อตาราง

4.2.3 ขึ้นต้นด้วยชื่อย่อของระบบในฐานข้อมูล ความยาวไม่เกิน 6 ตัวอักษร เช่น report,year

4.2.4 ตามด้วยชื่อมอดูลการทำงาน หรือบ่งบอกว่าใช้เก็บข้อมูลนั้น ๆ เช่น report_person

4.3. การตั้งชื่อฟิลด์

ข้อบังคับ

4.3.1 ต้องเป็นตัวอักษรพิมพ์เล็กทั้งหมด

4.3.2 ต้องคั่นด้วยเครื่องหมายขีดล่าง (_)

หลักการตั้งชื่อฟิลด์

4.3.3 ต้องขึ้นต้นด้วยชื่อย่อของตาราง ความยาวไม่เกิน 4 ตัวอักษร (ไม่รวมชื่อฐานข้อมูล) เช่น ตาราง hr_person ชื่อย่อเป็น ps หรือตาราง hr_admin ชื่อย่อเป็น am เป็นต้น

4.3.4 หลังชื่อย่อของตาราง ให้ระบุชื่อฟิลด์นั้นๆ โดยมีชื่อฟิลด์ที่ต้องบังคับใช้ในรูปแบบเดียวกัน ดังนี้

1) ชื่อฟิลด์ที่เป็นคีย์หลัก ต้องลงท้ายด้วย id เช่น dqs_id
2) ชื่อฟิลด์ที่เป็นความหมายหรือข้อมูลหลักของตาราง ต้องลงท้ายด้วย name เช่น dqs_name, dqs_first_name, dqs_last_name
3) ชื่อฟิลด์ที่บ่งบอกถึงลำดับของข้อมูล ต้องลงท้ายด้วย seq เช่น am_seq, dp_seq
4) ชื่อฟิลด์ที่บ่งบอกถึงลำดับชั้น ต้องลงท้ายด้วย level เช่น dp_level
5) ชื่อฟิลด์FK จากตารางอื่น ให้ใช้ชื่อเดิมมาต่อท้าย เช่น ps_pf_id, ps_dp_id
6) ชื่อฟิลด์FK ตารางตารางเดียวกันและบ่งบอกความสัมพันธ์แม่ลูก ต้องลงท้ายด้วย
7) parent_id เช่น dp_parent_id

ข้อยกเว้น

4.3.5 ชื่อย่อของตารางมีความยาวมากกว่า 4 ตัวอักษรได้กรณีที่ไม่สามารถลดคำให้น้อยลงได้ (ถ้าจำเป็น)

4.4. การเขียนคอมเมนต์ (Comment) ของตารางและฟิลด์

ทุกตาราง และทุกฟิลด์ต้องมีการคอมเมนต์หรือนิยามความหมายกำกับไว้ให้ครบถ้วน ไม่มีข้อยกเว้น

หลักการเขียนคอมเมนต์

4.4.1 ตาราง ให้นิยามความหมายว่า ตารางเก็บข้อมูลอะไร หรือใช้สำหรับทำอะไร ตัวอย่างเช่น ตาราง dqs_report คือ ตารางสรุปรายงาน

4.4.2 ฟิลด์ ให้นิยามความหมายว่าใช้เก็บข้อมูลอะไร ตัวอย่างเช่น

dqs_id คือ รหัสสมาชิก

dqs_name คือ ชื่อสมาชิก

4.4.3 การระบุตัวอย่างของข้อมูล หากฟิลด์นั้นมีตัวอย่างของข้อมูลชัดเจน ให้ใส่ต่อท้ายในเครื่องหมายวงเล็บด้วย ตัวอย่างเช่น dqs_status คือ สถานะของแผนก (Y=ใช้งาน, N=ไม่ใช้งาน)

4.4.4 ฟิลด์ที่อ้างอิงจากฟิลด์อื่น (FK) ให้อธิบายความหมายเดียวกันกับตารางตั้งต้น (PK) และต่อท้ายด้วยว่ามาจากตารางไหน หรือฐานข้อมูลไหน (ระบุชื่อฐานข้อมูลด้วย หากอยู่คนละฐานข้อมูล) ตัวอย่างเช่น dqs_personId คือ รหัสสมาชิก (ตาราง dqs_persons)

5. การตั้งชื่อตัวแปรของ Config

ข้อบังคับ

5.1. ต้องเป็นตัวอักษรพิมพ์ใหญ่ทั้งหมด

5.2. ต้องขั้นด้วยเครื่องหมายขีดล่าง (_)

หลักการตั้งชื่อฟิลด์

5.3. ขึ้นต้นด้วยชื่อย่อของระบบ (สอดคล้องกับชื่อไฟล์คอนฟิก)

5.4. ต่อท้ายด้วยชื่อที่ข้อมูลต้องการตั้ง เช่น URL_IMAGE, DB_NAME และ DE_PASSWORD

1) โฟลเดอร์ของระบบ ใช้คำว่า folder เช่น $config["ps_folder"], $config["ps_folder"]

2) ที่อยู่ไฟล์ที่อัพโหลดของระบบ ใช้คำว่า upload_path เช่น $config["hr_upload_path"]

3) ที่ตั้งไดเรกทอรี่ของระบบ ใช้คำว่า root_path เช่น $config["hr_root_path"]

4) ชื่อฐานข้อมูลของระบบ ใช้คำว่า db_name เช่น $config["hr_db_name"]

5) ที่อยู่รูปภาพต่างๆ ของระบบ ใช้คำว่า image_ชื่อข้อมูลนั้นๆ เช่น $config["hr_image_header"]

6) ที่อยู่ไอคอนต่างๆ ของระบบ ใช้คำว่า icon_ชื่อข้อมูลนั้นๆ เช่น $config["hr_icon_add"], $config["hr_icon_edit"], $config["hr_icon_delete"]

ข้อห้าม

5.5. ห้ามตั้งชื่อ Config ซ้ำกับชื่อที่มีอยู่แล้ว

5.6. ห้ามแก้ไขหรือลบ Config โดยพลการ หรือหากต้องการตั้งชื่อคอนฟิกเกี่ยวกับระบบอื่นเอง ให้ตั้งชื่อคอนฟิกขึ้นต้นด้วยชื่อระบบของตัวเองก่อนเสมอ เพื่อป้องกันการเขียนทับคอนฟิกของระบบอื่น

6. การตั้งชื่อฟังก์ชันของ Helper

ข้อบังคับ

6.1. ต้องเป็นตัวอักษรพิมพ์เล็กทั้งหมด

6.2. ต้องคั่นด้วยเครื่องหมายขีดล่าง (_)

หลักการตั้งชื่อฟิลด์

6.3. ขึ้นต้นด้วยชื่อย่อของระบบ (สอดคล้องกับชื่อไฟล์ Helper) เช่น tm, vc และ hr

6.4. ตามด้วยชื่อฟังก์ชันการทำงานนั้น ๆ ตัวอย่างเช่น hr_read_file()

ข้อห้าม

6.5. ห้ามตั้งชื่อฟังก์ชันซ้ำกับฟังก์ชันที่มีอยู่แล้วใน Helper

6.6. ห้ามตั้งชื่อฟังก์ชันซ้ำกับฟังก์ชันอื่น ๆ ที่มีในระบบทั้งหมด

6.7. ห้ามเพิ่ม/แก้ไข/ลบ ฟังก์ชันใน Helper โดยพลการ ต้องทำการปรึกษาหัวหน้าทีมพัฒนาก่อน

6.8. ห้ามตั้งชื่อฟังก์ชัน ซ้ำกับระบบอื่น หากต้องการเรียกฟังก์ชันจากระบบอื่นสามารถเรียกใช้ได้เลย (ถ้ามี) หรือหากต้องการคัดลอกฟังก์ชันจากระบบอื่นมายังระบบตัวเองให้ตั้งชื่อฟังก์ชันขึ้นต้นด้วยชื่อจากระบบอื่นมายังระบบตัวเอง

6.9. หลีกเลี่ยงการใช้ตัวเลขในการตั้งชื่อฟังก์ชัน สำหรับฟังก์ชันที่การทำงานคล้ายกันแต่ต้องการตั้งชื่อให้แตกต่างกันโดยใช้ตัวเลข ควรตั้งชื่อให้สื่อถึงการทำงาน

7. การเขียนคอมเมนต์

7.1. คอมเมนต์คลาสของ Controller และ Model

ในคลาสของ Controller และ Model ให้เขียนคอมเมนต์รูปแบบเดียวกัน

ข้อบังคับ

7.1.1 ให้เขียนคอมเมนต์ทุกไฟล์ ไม่มีข้อยกเว้น

7.1.2 ให้เขียนคอมเมนต์ฟังก์ชันกำกับทุกฟังก์ชัน ไม่มีข้อยกเว้น

7.1.3 เขียนคอมเมนต์คลาสไว้บรรทัดแรกของไฟล์

7.1.4 เขียนคอมเมนต์คลาสด้วยภาษาอังกฤษ หรือภาษาไทยเท่านั้น

หลักการเขียนคอมเมนต์ส่วนของ Class

7.1.5 บรรทัดที่ 1 ใช้เครื่องหมายเปิดคอมเมนต์ คือ /*

7.1.6 บรรทัดที่ 2 ระบุชื่อคลาส เช่น Member

7.1.7 บรรทัดที่ 3 ระบุชื่อคลาสและข้อความเกี่ยวกับการทำงานคร่าว ๆ เช่น Member Model of Member

7.1.8 บรรทัดที่ 4 ระบุชื่อผู้สร้างไฟล์คลาสหลังหัวข้อ @author เช่น @author : Somsak

7.1.9 บรรทัดที่ 5 ระบุวันที่สร้างไฟล์คลาสหลังหัวข้อ @Create Date เช่น @Create Date : 2563-8-25

7.1.10 บรรทัดที่ 6 ใช้เครื่องหมายปิดคอมเมนต์ คือ */

หมายเหตุ

แต่ละบรรทัดให้ใส่เครื่องหมาย * และเว้นวรรค 1 ครั้งก่อนเขียนข้อความเสมอ (ยกเว้นบรรทัดที่ 1 และบรรทัดที่ 6)

ตัวอย่างการคอมเมนต์คลาส

```
/*
* Member
* Member Model of Member
* @author : Somsak
* @Create Date : 2563-8-25
*/
```

7.2 คอมเมนต์ฟังก์ชันใน Controller, Model,Helper และ View

ในฟังก์ชันของ Controller, Model, Helper, View (ส่วนที่เป็น Javascript) ให้เขียนคอมเมนต์รูปแบบเดียวกัน

ข้อบังคับ

7.2.1 เขียนคอมเมนต์กำกับทุกฟังก์ชัน ไม่มีข้อยกเว้น

7.2.2 เขียนคอมเมนต์ฟังก์ชันไว้ด้านบน ก่อนประกาศฟังก์ชันนั้น ๆ

7.2.3 เขียนคอมเมนต์ฟังก์ชันด้วยภาษาอังกฤษ หรือภาษาไทยเท่านั้น

หลักการเขียนคอมเมนต์ส่วนของคลาส

7.2.4 บรรทัดที่ 1 ใช้เครื่องหมายเปิดคอมเมนต์ คือ /*

7.2.5 บรรทัดที่ 2 ระบุฟังก์ชัน เช่น create()

7.2.6 บรรทัดที่ 3 ระบุชื่อข้อความเกี่ยวกับการทำงานคร่าว ๆ เช่น create member

7.2.7 บรรทัดที่ 4 ระบุข้อมูลที่ระบบต้องรับเข้ามาหลังหัวข้อ @input เช่น @input : numberOfMember

7.2.8 บรรทัดที่ 5 ระบุข้อมูลที่ระบบแสดงหรือข้อมูลส่งออกมา หลังหัวข้อ @output เช่น @output : UserID

7.2.9 บรรทัดที่ 6 ระบุชื่อผู้สร้างไฟล์คลาสหลังหัวข้อ @author เช่น @author : Jackson

7.2.10 บรรทัดที่ 7 ระบุวันที่สร้างไฟล์คลาสหลังหัวข้อ @Create Date เช่น @Create Date : 2564-9-17

7.2.11 บรรทัดที่ 8 ใช้เครื่องหมายปิดคอมเมนต์ คือ */

หมายเหตุ :

7.2.12 แต่ละบรรทัดให้ใส่เครื่องหมาย * และเว้นวรรค 1 ครั้งก่อนเขียนข้อความเสมอ (ยกเว้นบรรทัดที่ 1 และบรรทัดที่ 8)

7.2.13 กรณีไม่มีข้อมูลรับเข้าให้ใส่เครื่องหมายขีด (-)

ตัวอย่างการคอมเมนต์ส่วนของฟังก์ชัน

```
/*
* create()
* create member
* @input : numberOfMember
* @output : StudentID
* @author : Somsak
* @Create Date : 2563-8-25
*/
```

7.3 คอมเมนต์ส่วนของ View

ข้อบังคับ

7.3.1 ให้เขียนคอมเมนต์ทุกไฟล์ View ไม่มีข้อยกเว้น

7.3.2 หากมีฟังก์ชันในหน้า ต้องเขียนคอมเมนต์ฟังก์ชันกำกับทุกฟังก์ชันไม่มีข้อยกเว้น

7.3.3 เขียนคอมเมนต์ส่วนของ View ไว้บรรทัดแรกของไฟล์

7.3.4 เขียนคอมเมนต์ส่วนของ View ด้วยภาษาอังกฤษ หรือภาษาไทยเท่านั้น

หลักการเขียนคอมเมนต์ส่วนของ View

7.3.5 บรรทัดที่ 1 ใช้เครื่องหมายเปิดคอมเมนต์ คือ /*

7.3.6 บรรทัดที่ 2 ระบุชื่อไฟล์ View เช่น index.php

7.3.7 บรรทัดที่ 3 ระบุชื่อข้อความเกี่ยวกับการทำงานคร่าว ๆ เช่น Show member

7.3.8 บรรทัดที่ 4 ระบุข้อมูลที่ระบบต้องรับเข้ามาหลังหัวข้อ @input เช่น @input : numberOfMember

7.3.9 บรรทัดที่ 5 ระบุข้อมูลที่ระบบแสดงหรือข้อมูลส่งออกมา หลังหัวข้อ @output เช่น @output : UsertID

7.3.10 บรรทัดที่ 6 ระบุชื่อผู้สร้างไฟล์คลาสหลังหัวข้อ @author เช่น @author : Jackson

7.3.11 บรรทัดที่ 7 ระบุวันที่สร้างไฟล์คลาสหลังหัวข้อ @Create Date เช่น @Create Date : 2564-9-17

7.3.12 บรรทัดที่ 8 ใช้เครื่องหมายปิดคอมเมนต์ คือ */

หมายเหตุ

7.3.13 แต่ละบรรทัดให้ใส่เครื่องหมาย * และเว้นวรรค 1 ครั้งก่อนเขียนข้อความเสมอ (ยกเว้นบรรทัดที่ 1 และบรรทัดที่ 8)

7.3.14 กรณีไม่มีข้อมูลรับเข้าให้ใส่เครื่องหมายขีด (-)

ตัวอย่างการคอมเมนต์ส่วนของ View

```
/*
* index.blade.php
* Show member
* @input : numberOfMember
* @output : StudentID
* @author : Somsak
* @Create Date : 2563-8-25
*/
```

7.4 คอมเมนต์บรรทัดเดียวหรือตัวแปรต่าง ๆ

กรณีต้องการคอมเมนต์เพื่อนิยามความหมายของตัวแปร หรือส่วนการทำงานบรรทัดนั้น ๆ หรือคอมเม้นท์เพื่อระบุวันที่แก้ไข ผู้แก้ไข หรือหมายเหตุสำหรับกรณีที่มีการปรับแก้หรือเพิ่มเติมโปรแกรม (ไม่บังคับ)

หลักการเขียนคอมเมนต์

7.4.1 ขึ้นต้นด้วยเครื่องหมายคอมเมนต์ก่อนที่จะประกาศตัวแปร 1 บรรทัด เช่น //

7.4.2 เว้นวรรค 1 ครั้ง ตามด้วยคอมเม้นท์ที่ต้องการ โดยสามารถคอมเม้นท์เป็นภาษาไทยหรืออังกฤษได้ตามความเหมาะสม

7.4.3 ให้คอมเม้นท์ด้านบนก่อนประกาศตัวแปร หรือก่อนบรรทัดนั้นๆ

ตัวอย่างการคอมเมนต์

```
// กำหนดค่าเริ่มต้นของ i โดยใช้สำหรับการนับรอบ
$i=0;
// ส่วนของ Input Password
<input type="password" name="password">
```

7.5 คอมเมนต์สำหรับการแก้ไขไฟล์หรือฟังก์ชัน Controller, Model และ View

กรณีต้องการคอมเม้นต์ส่วนของการทำงานที่มีคำสั่งมากกว่า 1 บรรทัด เพื่อระบุขอบเขตการทำงานนั้นๆ (ไม่บังคับ)

ข้อบังคับ

7.5.1 เขียนคอมเม้นต์ด้วยตัวอักษรภาษาอังกฤษเท่านั้น และขึ้นต้นด้วยอักษรตัวพิมพ์ใหญ่

7.5.2 ต้องระบุคอมเม้นท์ไว้ทั้ง 2 ส่วน คือ ส่วนบนและส่วนท้ายของการทำงานนั้น ๆ

หลักการคอมเม้นท์ส่วนบน

7.5.3 เปิด - ปิดคอมเม้นท์ไว้ด้านบนก่อนเริ่มการทำงานนั้นๆ โดยใช้เครื่องหมายคอมเม้นท์แบบ PHP คือ /* และ */ ตามลำดับ หรือเครื่องหมายคอมเม้นท์ใน HTML ก็ได้

7.5.4 ส่วนของข้อความให้เว้นวรรค 1 ครั้ง แล้วขึ้นต้นด้วย Start แล้วตามด้วยอธิบายส่วนการทำงานนั้น ๆ

หลักการเขียนคอมเมนต์

7.5.5 ให้เขียนต่อจากคอมเมนต์ในส่วนของหัวไฟล์หรือฟังก์ชันนั้น โดยระบุครั้งที่แก้ไข วันที่ทำการแก้ไข ชื่อผู้แก้ไข (กรณีไม่ใช่ผู้สร้างไฟล์ หรือฟังก์ชันนั้น) และอธิบายสิ่งที่ทำการแก้ไข เช่น @Update Date 2: 2564-9-17 Jackson แก้ไขส่วนของปุ่มสมัครสมาชิก

หลักการคอมเม้นต์ส่วนท้าย

7.5.6 เปิด – ปิดคอมเม้นต์ไว้ด้านล่างสุดหลังการทำงานนั้น ๆ โดยใช้เครื่องหมายคอมเม้นต์แบบ PHP คือ /* และ */ ตามลำดับ หรือเครื่องหมายคอมเม้นท์ใน HTML ก็ได้

7.5.7 ส่วนของข้อความให้เว้นวรรค 1 ครั้ง แล้วขึ้นต้นด้วย End แล้วตามด้วยอธิบายส่วนการทำงานนั้น ๆ

ตัวอย่างการคอมเมนต์สำหรับการแก้ไขไฟล์หรือฟังก์ชัน

```
/* Show view login */

Welcome to Document QRcode System

End Show view login*/
```

ส่วนที่ 2 มาตรฐานส่วนติดต่อผู้ใช้ (UI STANDARDS)

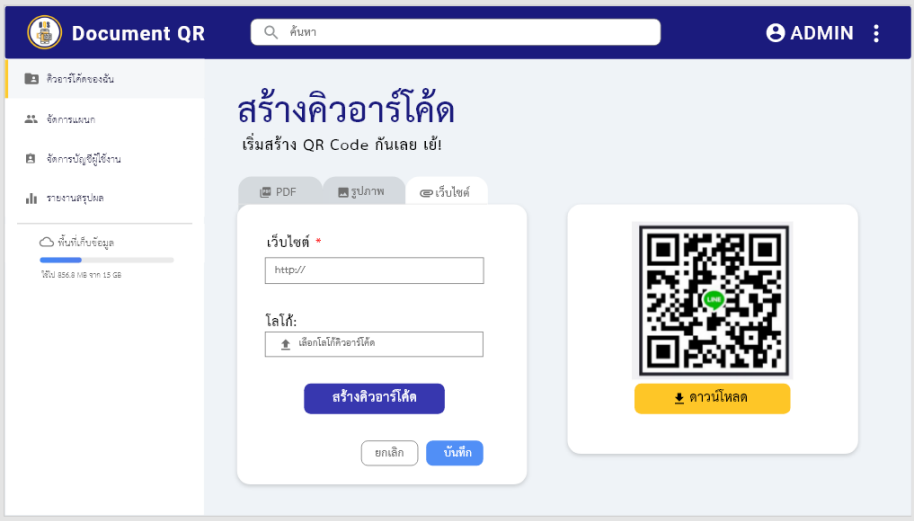
มาตรฐานส่วนติดต่อผู้ใช้เป็นส่วนเทมเพลต Bootstrap Framework เป็นต้นแบบในการกำหนดมาตรฐานที่ใช้ในการพัฒนากันอย่างแพร่หลายและมีลักษณะเหมือนกัน ดังนั้น จึงจัดมาตรฐานนี้ เพื่อให้การแสดงผลส่วนติดต่อผู้ใช้งาน (User Interface) เป็นไปตามมาตรฐานเดียวกัน

1. การแสดงสีปุ่ม (BUTTON COLOR)

ภาพที่ 2-1 แสดงรายการปุ่ม และสีปุ่ม (Button Color)

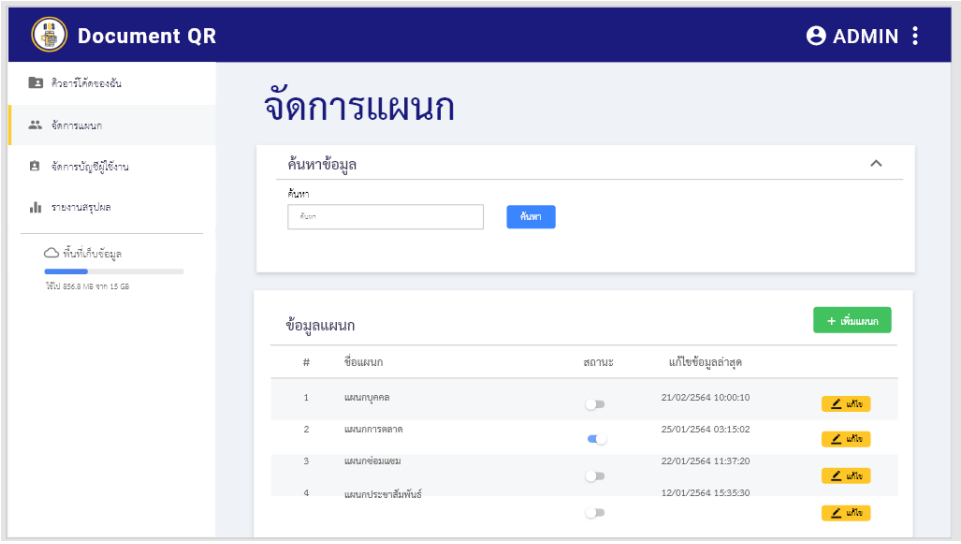
- ปุ่ม**สร้างคิวอาร์โค้ด และแก้ไขชื่อไฟล์** แสดงเป็น สีน้ำเงิน
- ปุ่ม**ค้นหา, ย้าย และบันทึก** แสดงเป็น สีฟ้า
- ปุ่ม**ยืนยัน, เพิ่มแผนก** แสดงเป็น สีเขียว
- ปุ่ม**สร้าง และดูสถิติการเข้าใช้งาน** แสดงเป็น สีขาว
- ปุ่ม**ลบ และยกเลิก** แสดงเป็น สีแดง
- ปุ่ม**ดาวน์โหลด** แสดงเป็น สีเหลือง

2. การจัดวางตำแหน่งปุ่ม (Button Position)



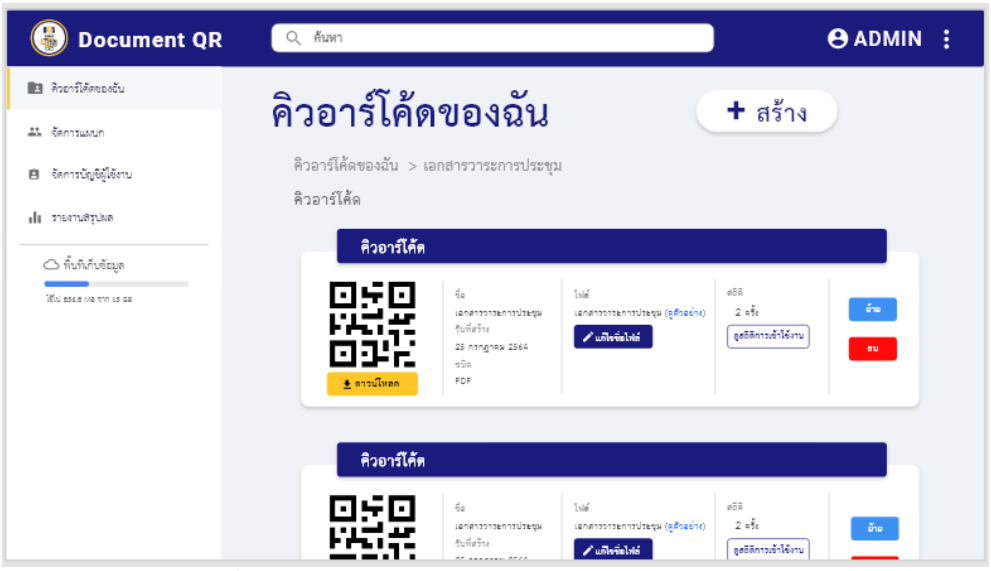
ภาพที่ 2-2 แสดงการจัดวางตำแหน่งปุ่ม (Button Position)

- ปุ่ม**สร้างคิวอาร์โค้ด** จัดวางตำแหน่งกลาง
- ปุ่ม**บันทึก** จัดวางตำแหน่งกลางล่าง และถัดจากปุ่มยกเลิก
- ปุ่ม**ยกเลิก** จัดวางตำแหน่งกลางล่าง
- ปุ่ม**ดาวน์โหลด** จัดวางตำแหน่งขวาล่าง



ภาพที่ 2-3 แสดงการจัดวางตำแหน่งปุ่ม (Button Position)

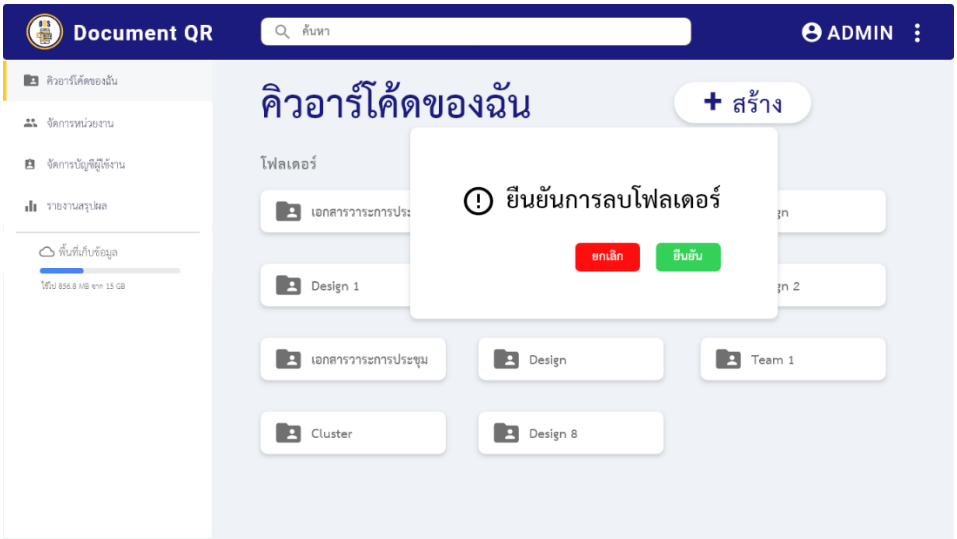
- ปุ่ม**ค้นหา** จัดวางตำแหน่งกลางบน
- ปุ่ม**เพิ่มแผนก** จัดวางตำแหน่งขวากลาง
- ปุ่ม**แก้ไข** จัดวางตำแหน่งขวา



ภาพที่ 2-4 แสดงการจัดวางตำแหน่งปุ่ม (Button Position)

- ปุ่ม**สร้าง** จัดวางตำแหน่งขวาบน
- ปุ่ม**ย้าย** จัดวางตำแหน่งขวา
- ปุ่ม**ลบ** จัดวางตำแหน่งขวา

3. การแสดงกล่องข้อความยืนยัน (Confirm Box)



ภาพที่ 2-5 แสดงการแสดงกล่องข้อความยืนยัน (Confirm Box)

- ปุ่ม**ยืนยัน** จัดวางตำแหน่งขวา
- ปุ่ม**ยกเลิก** จัดวางตำแหน่งซ้าย

- กรณีปุ่มตกลง สามารถแสดงเป็นปุ่มบันทึก, ยืนยัน, แก้ไข, ลบ หรือปุ่มดำเนินการอื่น ๆ ที่ต้องการใช้ให้สอดคล้องกับงาน โดยแสดงสีปุ่มตามดังรูปที่ 1

4. การแสดงผลอื่น ๆ

กรณีเกี่ยวกับมาตรฐานในส่วนติดต่อผู้ใช้ที่กำหนดรูปแบบกันในทีมพัฒนา ให้เป็นรูปแบบเดียวกันภายในระบบ มีดังนี้

การแสดงข้อความแจ้งเตือน Form Validation รูปแบบการแสดงผลจะเป็นในลักษณะของ Alert Box ทั้งระบบ

การแสดง Tooltip จะมีคำอธิบายเพิ่มเติมในส่วนเงื่อนไขต่าง ๆ ที่จำเป็นในการกรอกข้อมูลลงแบบฟอร์มซึ่งจะเป็นในลักษณะเดียวกันทั้งระบบ

การแสดง Placeholder ใช้ในการแสดงตัวอย่างของข้อความที่จำเป็นต้องกรอกภายในฟิลด์นั้น ๆ ซึ่งข้อความที่แสดงขึ้นมานั้นจะเป็นในลักษณะข้อความพื้นหลังของฟิลด์

การแสดงไอคอนหรือรูปภาพแทนการดำเนินการ เลือกใช้ในสิ่งที่เหมาะสมกับเทมเพลตของระบบ ซึ่งจะมีลักษณะเดียวกันทั้งระบบ

การแสดงวันที่ (Date Format) จะเป็นการแสดง วัน เดือน ปี และเวลา ซึ่งจะมีรูปแบบเดียวกันทั้งระบบ