

เอกสารการออกแบบระบบ
การออกแบบและพัฒนาซอฟต์แวร์เพื่อการบริหารจัดหาอาจารย์สอนพิเศษ
Tutorium (ฉบับแก้ไข)



เสนอ

รองศาสตราจารย์ ดร. ธาราทิพย์ สุวรรณศาสตร์

ผู้จัดทำ

กนกพล	ธงไชยเจริญศิริ	5731001821
ชนาธิป	เกรียงไกรเพชร	5730106921
ณัฐภัทร	บุญประคอง	5731036821
ธีรโชติ	บุญประภากร	5730271721
นนทิววัฒน์	วิสุทธิไกรสิทธิ์	5730282621
พีรวุฒิ	เหลือ่องเรืองโรจน์	5731083221
ภาคภูมิ	ทวิสิทธิชาติ	5731087821
อธิป	อินทรภิมย์	5731111121

ผู้จัดการโครงการ

พีรวุฒิ เหลือ่องเรืองโรจน์

ส่งมอบวันที่ 9 พฤศจิกายน 2560

โครงการนี้เป็นส่วนหนึ่งของรายวิชา 2110423 วิศวกรรมซอฟต์แวร์ (Software Engineering)

ภาคการศึกษาต้น ปีการศึกษา 2560 ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

สารบัญ

บทที่ 1 บทนำ	1
1.1 วัตถุประสงค์ของระบบ	1
1.2 วัตถุประสงค์ของการออกแบบระบบ	1
1.3 นิยามศัพท์.....	2
บทที่ 2 สถาปัตยกรรมซอฟต์แวร์	3
2.1 ภาพรวมของการออกแบบสถาปัตยกรรม	3
2.2 การแบ่งส่วนประกอบของระบบ (Subsystem Decomposition).....	4
2.3 การเชื่อมโยงระหว่างซอฟต์แวร์และฮาร์ดแวร์	5
2.4 โครงสร้างของระบบ	6
2.5 การออกแบบด้านความปลอดภัย.....	10
บทที่ 3 การออกแบบส่วนต่อประสานกับผู้ใช้	19
3.1 ระบบสำหรับบัญชีผู้ใช้ทั่วไป (non-administrator)	19
3.1.1 ระบบย่อยการสมัครสมาชิก	19
3.1.2 ระบบย่อยบัญชีนักเรียน.....	21
3.1.3 ระบบย่อยบัญชีผู้ให้บริการ (Tutor)	28
3.2 ระบบสำหรับบัญชีผู้ดูแลระบบ (administrator).....	33
บทที่ 4 รหัสเทียมสำหรับฟังก์ชันที่สำคัญในระบบ	40
4.1 การลงทะเบียนผู้ใช้.....	40
4.2 การลงชื่อเข้าสู่ระบบ	41
4.3 การส่งคำร้องขอสมัครเป็นผู้ให้บริการ.....	42
4.4 การมอบสถานะผู้ให้บริการ	42
4.5 การเพิ่มเวลาว่างสอน	43
4.6 การจองเวลาเรียน	44
4.7 การยกเลิกการจองเวลาเรียน.....	45
4.8 การค้นหาผู้ให้บริการ.....	46

4.9 การค้นหาผู้รับบริการ	46
4.10 การส่งคำร้องขอรับบริการ.....	46
4.11 การสร้างข้อเสนอคอร์สเรียน.....	47
4.12 การลบข้อเสนอคอร์สเรียน	48
4.13 การส่งข้อเสนอคอร์สเรียน.....	48
4.14 การแก้ไขข้อเสนอคอร์สเรียน.....	49
4.15 การชำระค่าบริการ.....	50
ภาคผนวก ก แผนภาพกรณีใช้งานบาวนด์รี.....	52

สารบัญตาราง

ตารางที่ 1 ตารางควบคุมการเข้าถึงแสดงสิทธิ์ในการเข้าถึงหน้าต่าง ๆ ในระบบของผู้ใช้ในแต่ละบทบาท.....	11
ตารางที่ 2 ตารางควบคุมการเข้าถึงแสดงสิทธิ์การเข้าถึงเมท็อดต่าง ๆ ของผู้ใช้ในแต่ละบทบาท	18

สารบัญภาพ

ภาพที่ 1 แผนภาพส่วนประกอบ (Component Diagram)	4
ภาพที่ 2 แผนภาพการปรับใช้ (Deployment Diagram)	5
ภาพที่ 3 แผนภาพคลาสเอนทิตี (Entity Class Diagram)	6
ภาพที่ 4 แผนภาพคลาสควบคุม (Control Class Diagram)	7
ภาพที่ 5 แผนภาพคลาสบาวนด์ารี (Boundary Class Diagram)	8
ภาพที่ 6 แผนภาพคลาสรวมทั้งระบบ	9
ภาพที่ 7 หน้าแรกของระบบสำหรับผู้เยี่ยมชม	19
ภาพที่ 8 หน้าสมัครสมาชิกสำหรับผู้เยี่ยมชม	20
ภาพที่ 9 หน้าลงชื่อเข้าใช้ระบบสำหรับผู้เยี่ยมชม	20
ภาพที่ 10 หน้าแกระบบสำหรับนักเรียน แสดงรายการหล่นลง (Dropdown) ที่เกี่ยวกับบัญชีผู้ใช้	21
ภาพที่ 11 หน้าแกระบบสำหรับนักเรียน แสดงรายการหล่นลง (Dropdown) สำหรับคอร์สเรียน	22
ภาพที่ 12 หน้าข้อมูลส่วนตัวของผู้ใช้	23
ภาพที่ 13 หน้ารายงานปัญหา	24
ภาพที่ 14 หน้าจัดการคอร์สเรียนสำหรับนักเรียน	25
ภาพที่ 15 หน้าจัดการข้อเสนอคอร์สเรียน	26
ภาพที่ 16 หน้าจอผุดขึ้น (Pop-up) แสดงข้อมูลของตัวเตอร์จากหน้าจัดการข้อเสนอคอร์สเรียน	27
ภาพที่ 17 หน้าสมัครเป็นตัวเตอร์	28
ภาพที่ 18 หน้าแกระบบสำหรับตัวเตอร์	29
ภาพที่ 19 หน้ากรอกข้อมูลการสอนของตัวเตอร์	30
ภาพที่ 20 หน้าจัดการคอร์สสำหรับตัวเตอร์	31
ภาพที่ 21 หน้าคั่นหน้านักเรียนสำหรับตัวเตอร์	32
ภาพที่ 22 หน้าลงชื่อเข้าใช้สำหรับผู้ดูแลระบบ	33
ภาพที่ 23 หน้าแสดงภาพรวมของระบบ	34
ภาพที่ 24 หน้าจัดการผู้ใช้	35
ภาพที่ 25 หน้าจัดการคำขอเป็นตัวเตอร์	36
ภาพที่ 26 หน้าแสดงรายงาน	37
ภาพที่ 27 หน้าแสดงบัญชีผู้ใช้ที่ถูกระงับ	38

ภาพที่ 28 หน้าแสดงคำขอลบบัญชีผู้ใช้.....	39
ภาพที่ 29 แผนภาพกรณีใช้งานบารันดารี.....	52

บทที่ 1 บทนำ

1.1 วัตถุประสงค์ของระบบ

ซอฟต์แวร์เพื่อการบริหารจัดการอาจารย์สอนพิเศษ Tutorium เป็นระบบเว็บแอปพลิเคชันที่ถูกพัฒนาขึ้นเพื่ออำนวยความสะดวกให้กับนักเรียนและติวเตอร์ในจับคู่ระหว่างผู้เรียนกับผู้สอน การนัดหมายเวลาเรียน และการชำระเงิน รวมถึงช่วยอำนวยความสะดวกให้กับนักเรียนในการหาติวเตอร์ที่ต้องการเรียนด้วย และอำนวยความสะดวกให้กับติวเตอร์ที่ต้องการหารายได้จากการสอนพิเศษผ่านระบบนายหน้าที่สามารถเชื่อถือได้

1.2 วัตถุประสงค์ของการออกแบบระบบ

เมื่อทีมผู้พัฒนาได้ทำการวิเคราะห์ความต้องการของผู้ใช้ออกมาเป็นแบบจำลองการวิเคราะห์ (analysis model) ที่กล่าวถึงในเอกสารข้อกำหนดซอฟต์แวร์ (software requirement specification) แล้ว ทีมผู้พัฒนาจะนำแบบจำลองการวิเคราะห์มาทำการออกแบบระบบให้ได้เป็นแบบจำลองการออกแบบ (design model) โดยคำนึงถึงสถานะความเป็นจริงของขั้นตอนการทำงาน โครงสร้างข้อมูล ซอฟต์แวร์ และฮาร์ดแวร์ เพื่อที่จะนำไปใช้ในขั้นตอนการพัฒนาจริง (implementation) ต่อไป

การออกแบบระบบของทีมผู้พัฒนาจะนำความต้องการเชิงคุณภาพของผู้ใช้ (non-functional requirement) เข้ามาพิจารณา โดยสามารถแบ่งวัตถุประสงค์ของการออกแบบระบบได้ดังต่อไปนี้

- 1) มีความง่ายต่อการใช้งาน (usability)
 - (1) ระบบแสดงผลเป็นภาษาไทย ซึ่งเป็นภาษาหลักของผู้ใช้ระบบ
 - (2) ระบบสามารถทำได้บนเว็บเบราว์เซอร์ได้ทั้งรูปแบบคอมพิวเตอร์ตั้งโต๊ะและอุปกรณ์เคลื่อนที่ โดยรองรับกับเว็บเบราว์เซอร์ Internet Explorer 9 หรือใหม่กว่า
- 2) มีความสามารถในการสนับสนุน (supportability)
 - (1) ระบบมีการออกแบบเพื่อให้ผู้ใช้สามารถใช้งานระบบได้บนคอมพิวเตอร์ส่วนตัวและอุปกรณ์เคลื่อนที่ทุกเครื่อง ผ่านทางเบราว์เซอร์ที่ระบบรองรับ
- 3) มีความมั่นคง (security)
 - (1) ผู้ใช้สามารถยืนยันตัวตนผู้เข้าใช้งานได้จากบัญชีผู้ใช้เฟซบุ๊ก (facebook) หรือบัญชีผู้ใช้ไลน์ (line)
 - (2) ระบบสามารถเก็บข้อมูลย้อนหลังได้อย่างน้อย 1 ปี

1.3 นิยามศัพท์

- 1.3.1 ผู้เยี่ยมชม (External user) หมายถึง ผู้ใช้ที่ไม่ได้เป็นสมาชิกที่ลงทะเบียนไว้กับระบบ
- 1.3.2 ผู้รับบริการ หรือนักเรียน (Student) หมายถึง ผู้ใช้ที่ลงทะเบียนเป็นสมาชิกผู้ใช้งานระบบ เพื่อใช้บริการค้นหาและจองการบริการ
- 1.3.3 ผู้ให้บริการ, ผู้สอน หรือติวเตอร์ (Tutor) หมายถึง ผู้รับบริการที่ทำการยืนยันตัวตน เพื่อรับสิทธิ์การยื่นข้อเสนอวิชาให้กับผู้รับบริการ
- 1.3.4 ผู้ดูแลระบบ (Administrator หรือ Admin) หมายถึง กลุ่มบุคคลที่ทำหน้าที่ดูแลระบบ ยืนยันตัวตนให้กับผู้ให้บริการ รับข้อร้องเรียนและจัดการบัญชีผู้ใช้
- 1.3.5 ผู้พัฒนาระบบ หมายถึง กลุ่มบุคคลซึ่งทำหน้าที่พัฒนาระบบ ดูแล และแก้ไขปัญหาทางเทคนิคของระบบ
- 1.3.6 ข้อเสนอวิชา (Course Offer) หมายถึง ข้อเสนอของผู้ให้บริการที่ส่งให้กับผู้รับบริการ เพื่อแสดงความจำนงค์ต้องการจะเสนอการสอน
- 1.3.7 คอร์ส (Course) หมายถึง กิจกรรมการเรียนการสอนระหว่างผู้ให้บริการและผู้รับบริการ
- 1.3.8 การเริ่มคอร์ส หมายถึง วันเริ่มต้นของกิจกรรมการเรียนการสอน
- 1.3.9 คำร้องขอรับบริการ (Request) หมายถึง ข้อเสนอที่ผู้รับบริการแสดงความจำนงค์ขอรับบริการจากผู้ให้บริการ

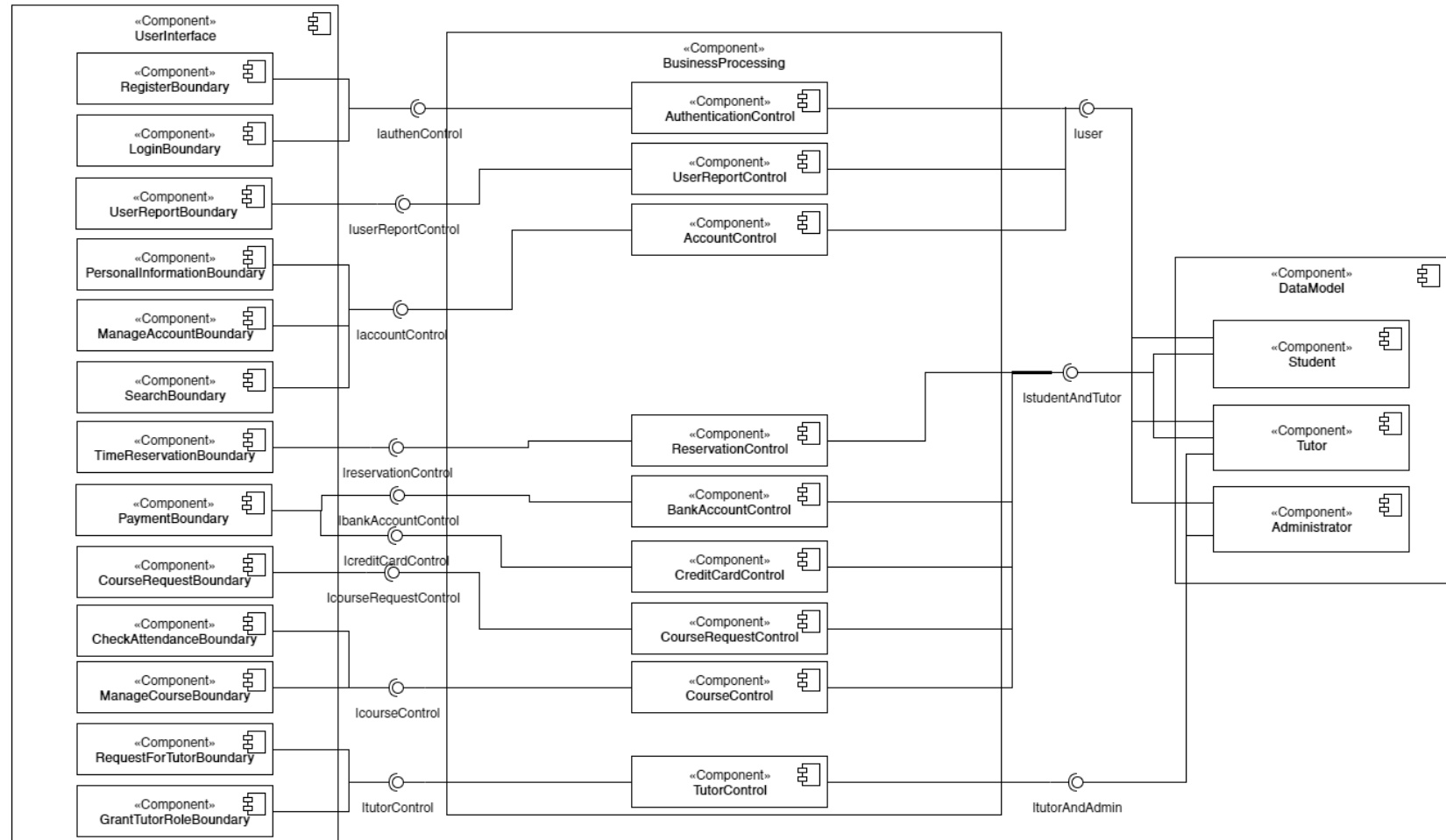
บทที่ 2 สถาปัตยกรรมซอฟต์แวร์

2.1 ภาพรวมของการออกแบบสถาปัตยกรรม

ซอฟต์แวร์เพื่อการบริหารจัดหาอาจารย์สอนพิเศษ Tutorium ออกแบบโดยใช้สถาปัตยกรรมแบบไคลเอนต์ - เซิร์ฟเวอร์ (Client-Server) โดยฝั่งไคลเอนต์จะติดต่อกับผู้ใช้ผ่านทางหน้าเว็บ และไคลเอนต์จะติดต่อกับเซิร์ฟเวอร์ผ่านเครือข่ายอินเทอร์เน็ต

2.2 การแบ่งส่วนประกอบของระบบ (Subsystem Decomposition)

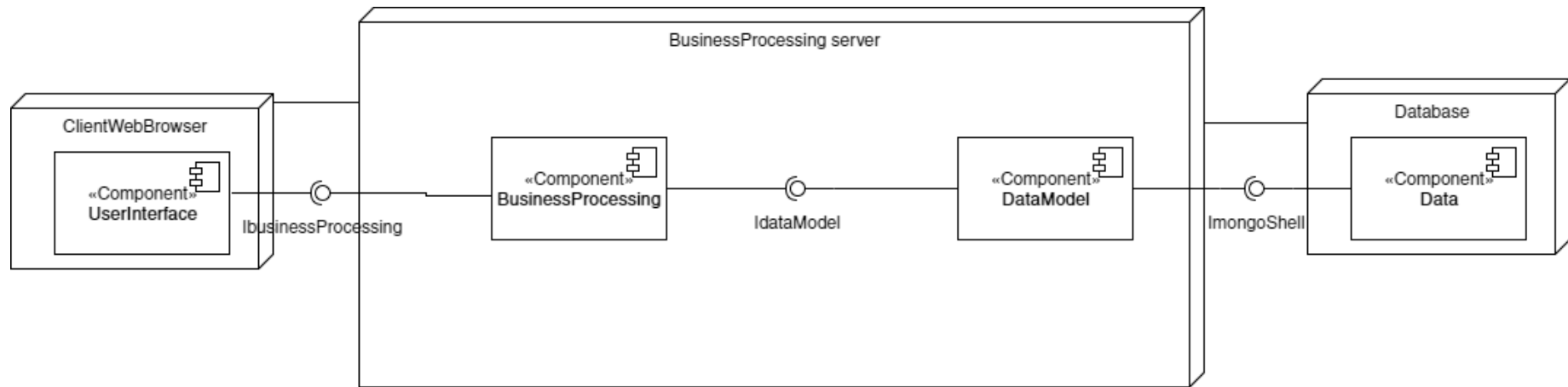
อธิบายด้วยแผนภาพส่วนประกอบ (Component Diagram) ดังต่อไปนี้



ภาพที่ 1 แผนภาพส่วนประกอบ (Component Diagram)

2.3 การเชื่อมโยงระหว่างซอฟต์แวร์และฮาร์ดแวร์

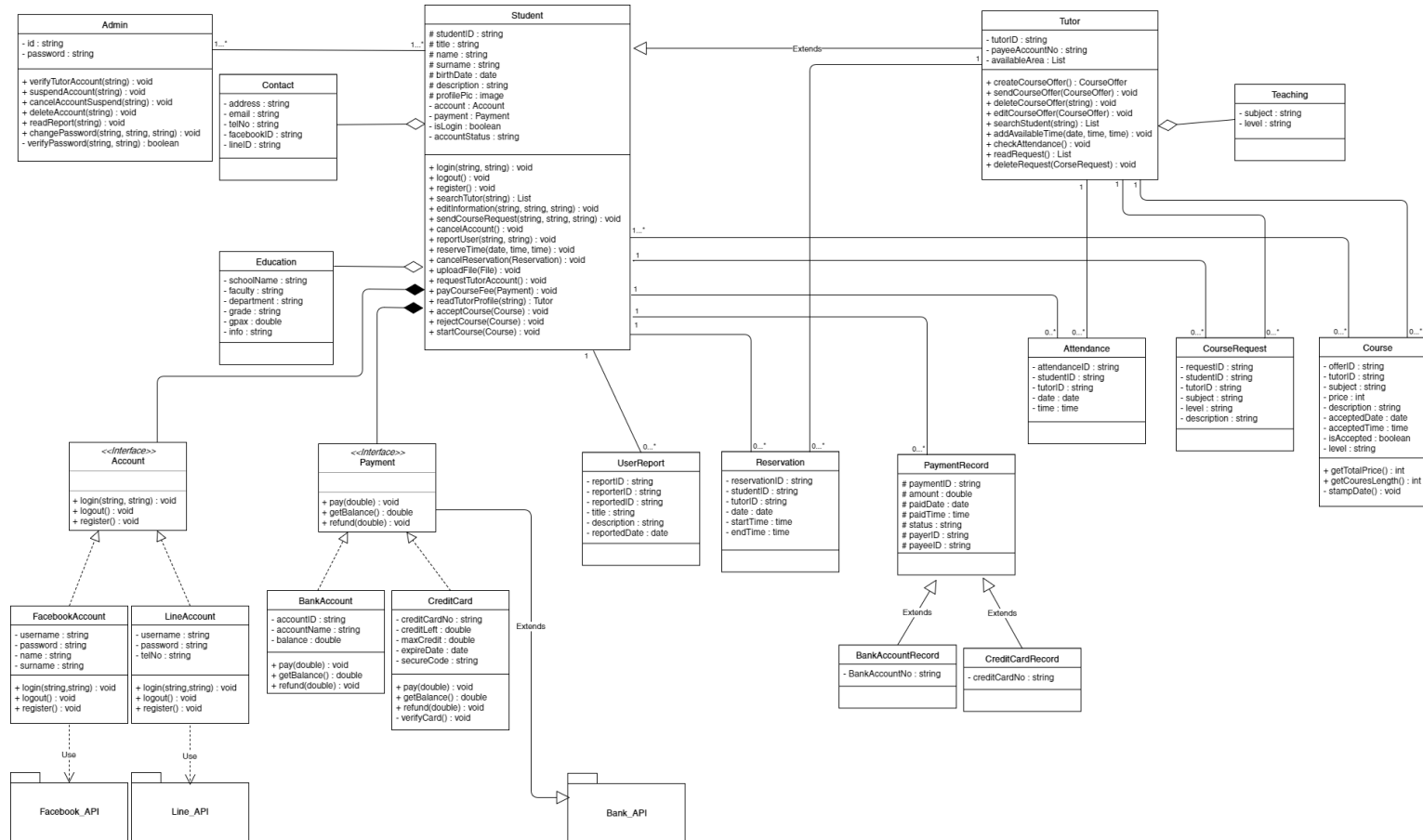
อธิบายด้วยแผนภาพการปรับใช้ (Deployment Diagram) ดังต่อไปนี้



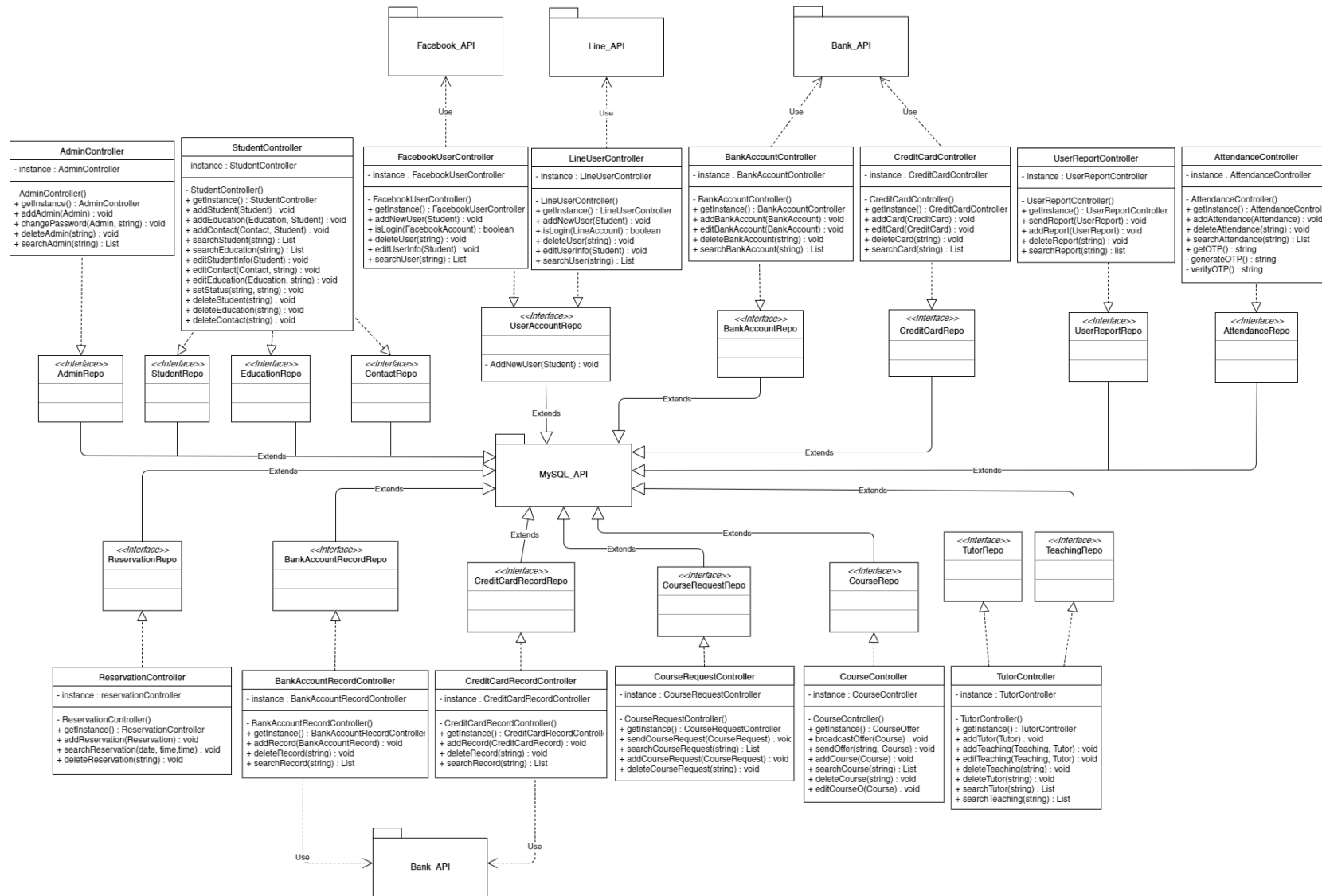
ภาพที่ 2 แผนภาพการปรับใช้ (Deployment Diagram)

2.4 โครงสร้างของระบบ

ทีมผู้พัฒนาได้ทำการออกแบบคลาสที่ประกอบไปด้วย คลาสเอนทิตี (Entity Class), คลาสควบคุม (Control Class) และคลาสบาว์ดารี (Boundary Class) โดยอธิบายด้วยแผนภาพคลาสเอนทิตี, แผนภาพคลาสควบคุม, แผนภาพคลาสบาว์ดารี และแผนภาพคลาสรวมทั้งระบบ ดังภาพที่ 3, 4, 5 และ 6 ตามลำดับ

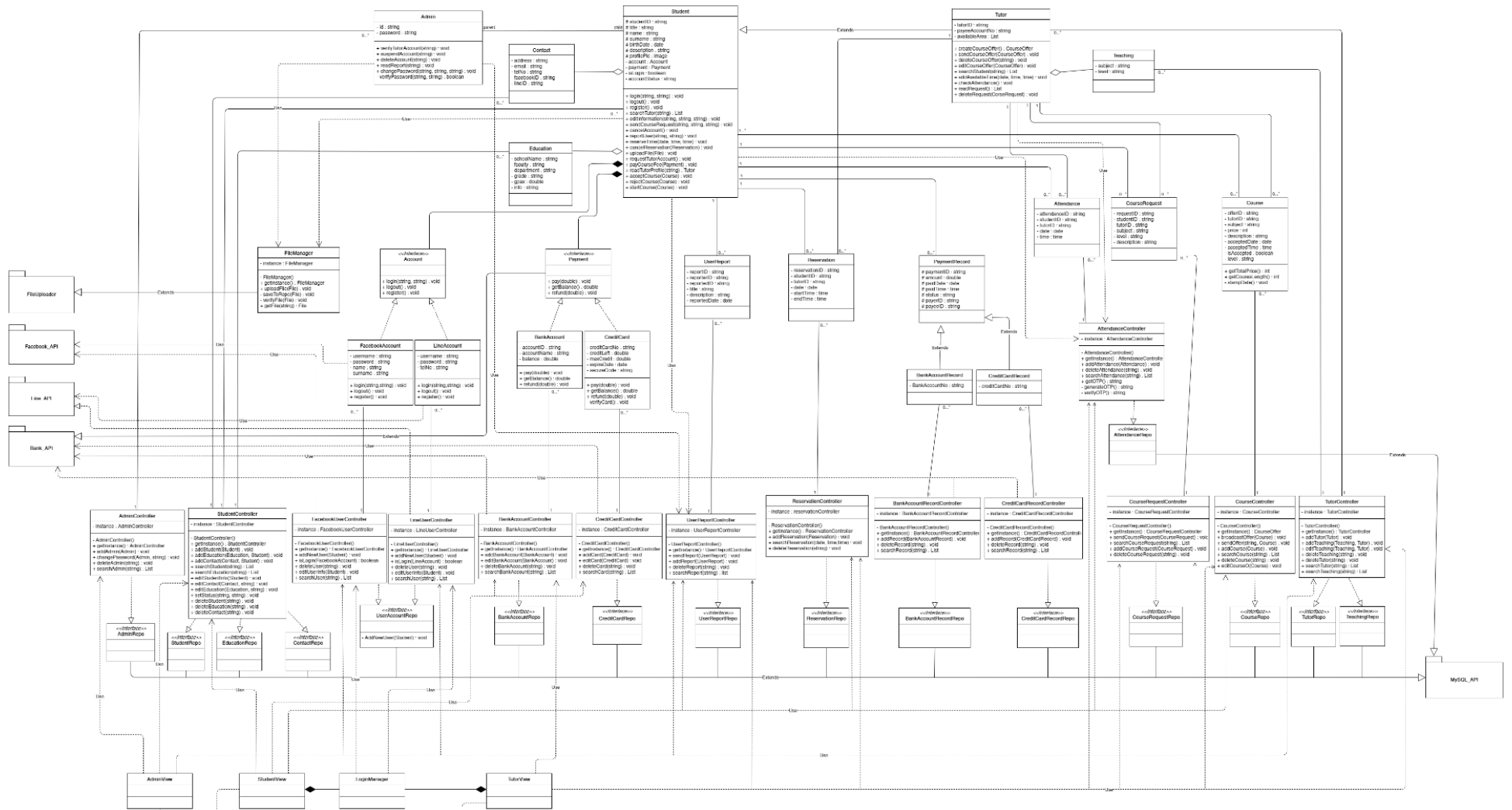


ภาพที่ 3 แผนภาพคลาสเอนทิตี (Entity Class Diagram)



ภาพที่ 4 แผนภาพคลาสควบคุม (Control Class Diagram)

ภาพที่ 5 แผนภาพคลาสบาวนด์ดารี (Boundary Class Diagram)



ภาพที่ 6 แผนภาพคลาสรวมทั้งระบบ

2.5 การออกแบบด้านความปลอดภัย

ระบบ Tutorium มีการควบคุมการใช้งานให้เป็นไปตามตารางการควบคุมการเข้าถึง (Access Control Table) ดังตารางที่ 1 ที่กล่าวถึงสิทธิ์การเข้าถึงหน้าเว็บต่าง ๆ และตารางที่ 2 ที่กล่าวถึงสิทธิ์การเข้าถึงเมนูที่ต่าง ๆ แยกตามผู้ใช้ในแต่ละบทบาท

ลำดับ	ชื่อหน้า	กลุ่มผู้ใช้งาน		
		Student	Tutor	Administrator
1	หน้าสมัครสมาชิก	✓	✓	
2	หน้าเข้าสู่ระบบ (ผู้เยี่ยมชม)	✓	✓	
3	หน้าเข้าสู่ระบบ (ผู้ดูแล)			✓
4	หน้าแรก (ผู้เยี่ยมชม)	✓	✓	
5	หน้าแรก (นักเรียน)	✓		
6	หน้าแรก (ติวเตอร์)		✓	
7	หน้าข้อมูลส่วนตัว	✓	✓	
8	หน้ารายงานการกระทำผิด	✓	✓	
9	หน้าจัดการคอร์สเรียน	✓	✓	
10	หน้าเสนอคอร์สเรียน		✓	
11	หน้าคั่นหน้านักเรียน		✓	
12	หน้ารายละเอียดคอร์สเรียน		✓	
13	หน้าสมัครเป็นผู้สอน	✓		
14	หน้าภาพรวมระบบ			✓
15	หน้าลบบัญชีผู้ใช้			✓
16	หน้ารับรายงานการกระทำผิด			✓

ลำดับ	ชื่อหน้า	กลุ่มผู้ใช้งาน		
		Student	Tutor	Administrator
17	หน้าลงทะเบียนผู้ใช้			✓
18	หน้ายืนยันการเป็นติวเตอร์			✓
19	หน้าจัดการผู้ใช้			✓

ตารางที่ 1 ตารางควบคุมการเข้าถึงแสดงสิทธิ์ในการเข้าถึงหน้าต่าง ๆ ในระบบของผู้ใช้ในแต่ละบทบาท

ลำดับ	เมทอด	กลุ่มผู้ใช้งาน		
		Student	Tutor	Administrator
1	Student			
1.1	login()	✓	✓	
1.2	logout()	✓	✓	
1.3	register()	✓	✓	
1.4	searchTutor()	✓	✓	
1.5	editInformation()	✓	✓	
1.6	sendCourseRequest()	✓	✓	
1.7	cancelAccount()	✓	✓	
1.8	reportUser()	✓	✓	
1.9	reserveTime()	✓	✓	
1.10	cancelReservation()	✓	✓	
1.11	uploadFile()	✓	✓	
1.12	requestTutorAccount()	✓	✓	

ลำดับ	เมทอด	กลุ่มผู้ใช้งาน		
		Student	Tutor	Administrator
1.13	payCourseFee()	✓	✓	
1.14	readTutorProfile()	✓	✓	
1.15	acceptCourse()	✓	✓	
1.16	rejectCourse()	✓	✓	
1.17	startCourse()	✓	✓	
2	Tutor			
2.1	createCourseOffer()		✓	
2.2	sendCourseOffer()		✓	
2.3	deleteCourseOffer()		✓	
2.4	editCourseOffer()		✓	
2.5	searchStudent()		✓	
2.6	addAvailableTime()		✓	
2.7	checkAttendance()		✓	
2.7	readRequest()		✓	
2.8	deleteRequest()		✓	
3	Admin			
3.1	verifyTutorAccount()			✓
3.2	suspendAccount()			✓
3.3	deleteAccount()			✓

ลำดับ	เมทอด	กลุ่มผู้ใช้งาน		
		Student	Tutor	Administrator
3.4	readReport()			✓
3.5	changePassword()			✓
4	FileManager			
4.1	getInstance()	✓	✓	✓
4.2	uploadFile()	✓	✓	✓
4.3	getFile()	✓	✓	✓
5	Course			
5.1	getTotalPrice()	✓	✓	
5.2	getCourseLength()	✓	✓	
6	AdminController			
6.1	getInstance()			✓
6.2	addAdmin()			✓
6.3	changePassword()			✓
6.4	deleteAdmin()			✓
6.5	searchAdmin()			✓
7	StudentController			
7.1	getInstance()	✓	✓	
7.2	addStudent()	✓	✓	
7.3	addEducation()	✓	✓	

ลำดับ	เมทอด	กลุ่มผู้ใช้งาน		
		Student	Tutor	Administrator
7.4	addContact()	✓	✓	
7.5	searchStudent()	✓	✓	
7.6	searchEducation()	✓	✓	
7.7	editStudentInfo()	✓	✓	
7.8	editContact()	✓	✓	
7.9	setStatus()	✓	✓	
7.10	deleteStudent()	✓	✓	
7.11	deleteEducation()	✓	✓	
7.12	deleteContact()	✓	✓	
8	FacebookUserController			
8.1	getInstance()	✓	✓	✓
8.2	addnewUser()	✓	✓	✓
8.3	isLogin()	✓	✓	✓
8.4	deleteUser()	✓	✓	✓
8.5	editUserInfo	✓	✓	✓
8.6	searchUser()	✓	✓	✓
9	FacebookAccount			
9.1	login()	✓	✓	✓
9.2	logout()	✓	✓	✓

ลำดับ	เมทอด	กลุ่มผู้ใช้งาน		
		Student	Tutor	Administrator
9.3	register()	✓	✓	✓
10	LineUserController			
10.1	getInstance()	✓	✓	✓
10.2	addNewUser()	✓	✓	✓
10.3	isLogin()	✓	✓	✓
10.4	deleteUser()	✓	✓	✓
10.5	editUserInfo()	✓	✓	✓
10.6	searchUser()	✓	✓	✓
11	LineAccount			
11.1	login()	✓	✓	✓
11.2	logout()	✓	✓	✓
11.3	register()	✓	✓	✓
12	BankAccountController			
12.1	getInstance()	✓	✓	
12.2	addCard()	✓	✓	
12.3	editCard()	✓	✓	
12.4	deleteCard()	✓	✓	
12.5	searchCard()	✓	✓	
13	BankAccount			

ลำดับ	เมทอด	กลุ่มผู้ใช้งาน		
		Student	Tutor	Administrator
13.1	pay()	✓	✓	
13.2	getBalance()	✓	✓	
13.3	refund()	✓	✓	
14	CreditCard			
14.1	getInstance()	✓	✓	
14.2	addCard()	✓	✓	
14.3	editCard()	✓	✓	
14.4	deleteCard()	✓	✓	
14.5	searchCard()	✓	✓	
15	CreditCard			
15.1	pay()	✓	✓	
15.2	getBalance()	✓	✓	
15.3	refund()	✓	✓	
15.4	verifyCard()	✓	✓	
16	ReservationController			
16.1	getInstance()	✓	✓	
16.2	addReservation()	✓	✓	
16.3	searchReservation()	✓	✓	
16.4	deleteReservation()	✓	✓	

ลำดับ	เมทอด	กลุ่มผู้ใช้งาน		
		Student	Tutor	Administrator
17	CourseRequestController			
17.1	getInstance()	✓	✓	
17.2	sendCourseRequest()	✓	✓	
17.3	searchCourseRequest()	✓	✓	
17.4	addCourseRequest()	✓	✓	
17.5	deleteCourseRequest()	✓	✓	
18	CourseController			
18.1	getInstance()	✓	✓	
18.2	broadcastOffer()	✓	✓	
18.3	sendOffer()	✓	✓	
18.4	addCourse()	✓	✓	
18.5	searchCourse()	✓	✓	
18.6	deleteCourse()	✓	✓	
18.7	editCourse()	✓	✓	
19	TutorController			
19.1	getInstance()		✓	✓
19.2	addTutor()		✓	✓
19.3	addTeaching()		✓	✓
19.4	editTeaching()		✓	✓

ลำดับ	เมทอด	กลุ่มผู้ใช้งาน		
		Student	Tutor	Administrator
19.5	deleteTeaching()		✓	✓
19.6	deleteTutor()		✓	✓
19.7	searchTutor()		✓	✓
19.8	searchTeaching		✓	✓
20	UserReportController			
20.1	getInstance()	✓	✓	✓
20.2	sendReport()	✓	✓	✓
20.3	addReport()	✓	✓	✓
20.4	deleteReport()	✓	✓	✓
20.5	searchReport()	✓	✓	✓

ตารางที่ 2 ตารางควบคุมการเข้าถึงแสดงสิทธิ์การเข้าถึงเมทอดต่าง ๆ ของผู้ใช้ในแต่ละบทบาท

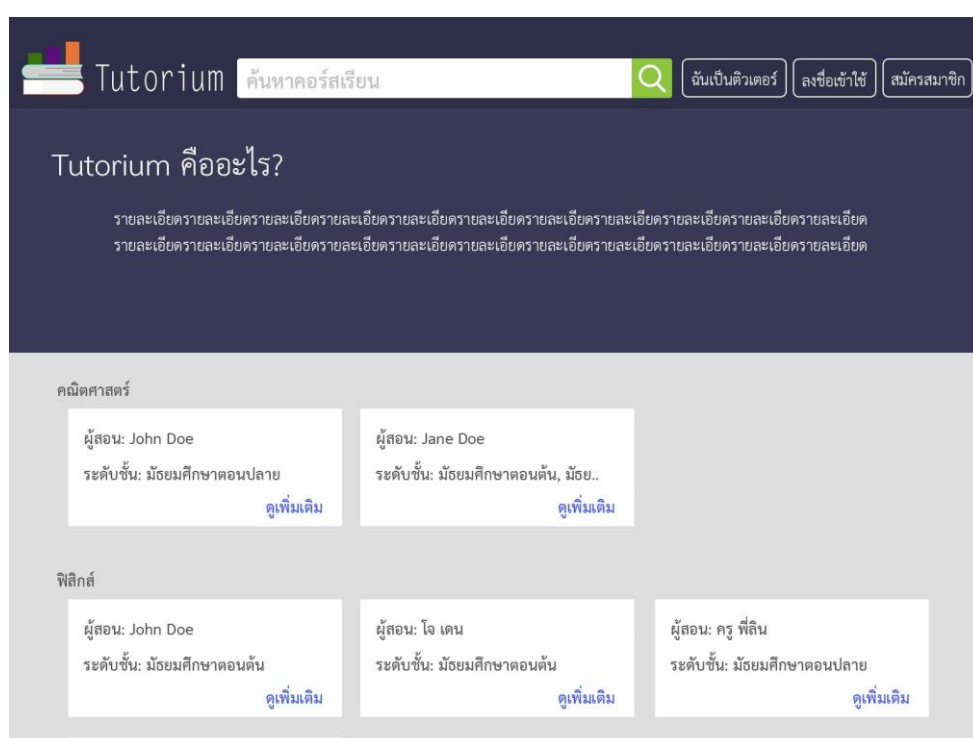
บทที่ 3 การออกแบบส่วนต่อประสานกับผู้ใช้

ทีมผู้พัฒนาได้แบ่งระบบออกเป็น 2 ส่วน คือ ระบบสำหรับบัญชีผู้ใช้ทั่วไป (non-administrator) และระบบสำหรับบัญชีผู้ดูแลระบบ (admin)

รูปแบบการออกแบบในเว็บจะใช้ Card-based Design การออกแบบไม่เน้นความสวยงามแบบหรูหรา แต่เน้นความเรียบง่าย(Minimal) เพื่อให้ผู้ใช้งานสามารถทำความเข้าใจวิธีการใช้งานระบบได้ง่ายมากที่สุด

3.1 ระบบสำหรับบัญชีผู้ใช้ทั่วไป (non-administrator)

3.1.1 ระบบย่อยการสมัครสมาชิก



ภาพที่ 7 หน้าแรกของระบบสำหรับผู้เยี่ยมชม

หน้าแรกของระบบสำหรับบัญชีผู้ใช้ทั่วไป จะแสดงหน้าจอสำหรับผู้ที่ยังไม่ได้ทำการลงชื่อเข้าใช้ โดยออกแบบให้มีส่วนแสดงข้อมูลเบื้องต้นเกี่ยวกับเว็บไซต์และบริการของ Tutorium มีลักษณะเป็นแถบขนาดใหญ่และเน้นตัวอักษรให้มีขนาดใหญ่เพื่อดึงดูดความสนใจของผู้ใช้ และแสดงรายชื่อตัวเตอร์ในส่วนถัดมาให้ผู้ใช้ได้ค้นหาคอร์สเรียนเพื่อเป็นตัวช่วยในการตัดสินใจใช้บริการ



ภาพที่ 8 หน้าสมัครสมาชิกสำหรับผู้เยี่ยมชม

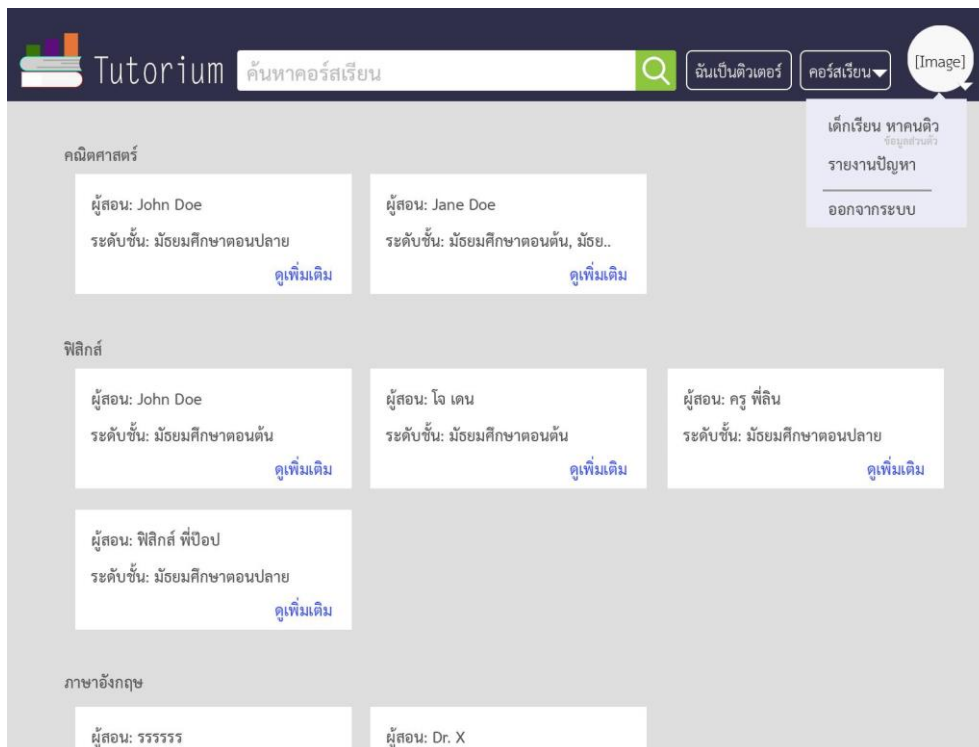
ออกแบบให้มีปุ่มขนาดใหญ่เพียงสองปุ่มเพื่อดึงความสนใจของผู้ใช้ให้เลือกวิธีสมัครสมาชิกด้วย Facebook หรือ Line และมีปุ่มขนาดเล็กทางด้านล่างสำหรับผู้ที่已经是สมาชิกอยู่แล้วเพื่อนำทางไปยังหน้าลงชื่อเข้าใช้



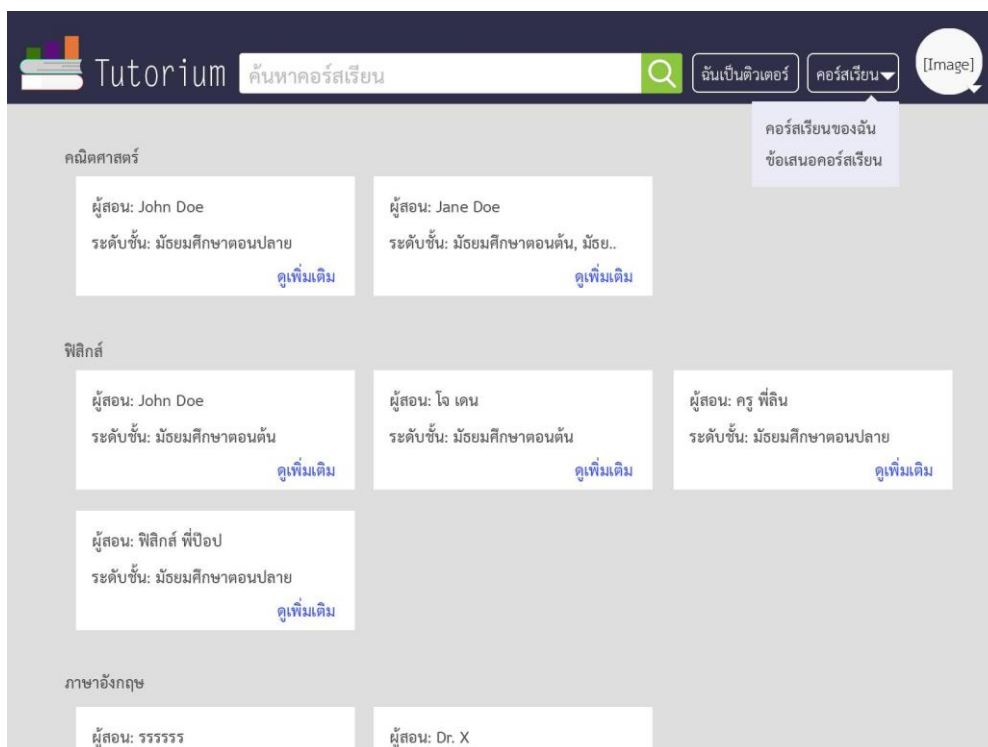
ภาพที่ 9 หน้าลงชื่อเข้าใช้ระบบสำหรับผู้เยี่ยมชม

ออกแบบให้มีปุ่มขนาดใหญ่เพียงสองปุ่มเพื่อดึงความสนใจของผู้ใช้ให้เลือกวิธีลงชื่อเข้าใช้ผ่าน Facebook หรือ Line และมีปุ่มขนาดเล็กทางด้านล่างสำหรับผู้ที่ยังไม่ได้เป็นสมาชิกเพื่อนำทางไปยังหน้าสมัครสมาชิก

3.1.2 ระบบย่อยบัญชีนักเรียน



ภาพที่ 10 หน้าแรกของระบบสำหรับนักเรียน แสดงรายการหล่นลง (Dropdown) ที่เกี่ยวกับบัญชีผู้ใช้



ภาพที่ 11 หน้าแรกของระบบสำหรับนักเรียน แสดงรายการหล่นลง (Dropdown) สำหรับคอร์สเรียน

การออกแบบหน้าแรกของระบบสำหรับนักเรียน แบ่งสาเหตุการออกแบบได้ดังนี้

1. ออกแบบปุ่มต่าง ๆ ให้อยู่ในรูปแบบรายการหล่นลง (Dropdown menu) เพื่อป้องกันไม่ให้เกิดความสับสนในหมวดหมู่ของรายการต่าง ๆ
2. ออกแบบให้แสดงรายชื่อติวเตอร์ในหน้าแรกเพื่ออำนวยความสะดวกให้แก่ผู้ใช้ที่เป็นนักเรียนให้ง่ายต่อการค้นหาติวเตอร์

ภาพที่ 12 หน้าข้อมูลส่วนตัวของผู้ใช้

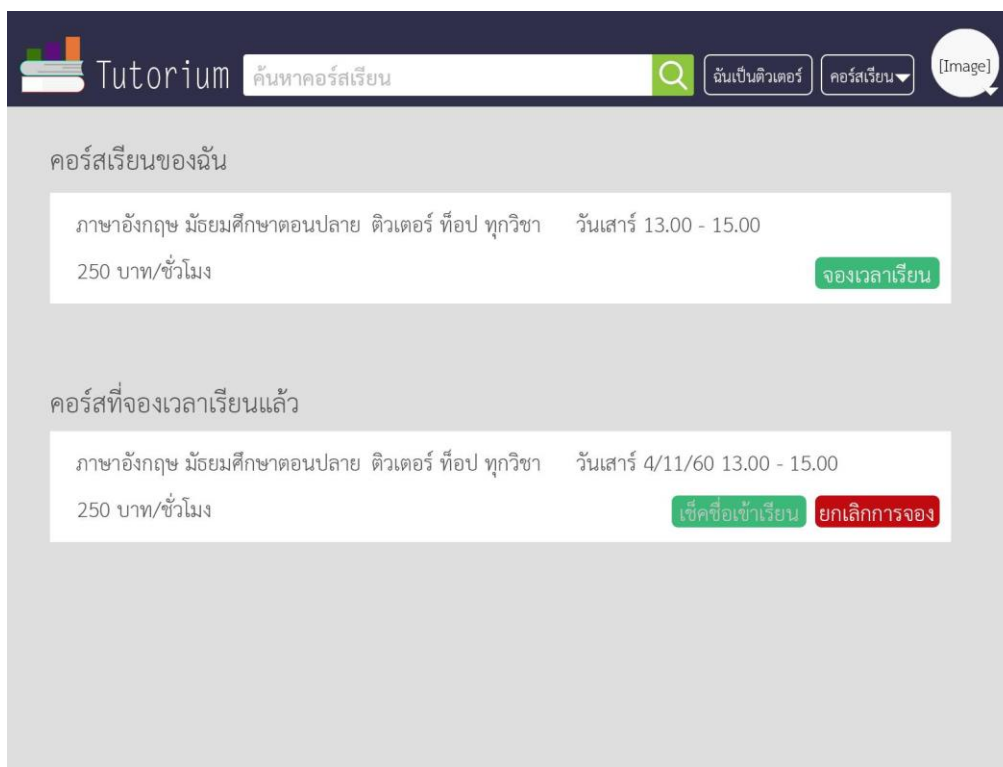
การออกแบบหน้าข้อมูลส่วนตัวของผู้ใช้ แบ่งสาเหตุการออกแบบได้ดังนี้

1. ออกแบบส่วนสำหรับข้อมูลทั่วไปให้เป็นช่องสำหรับกรอกหรือเลือก และมีปุ่ม “บันทึก” สำหรับยืนยันการแก้ไขข้อมูล โดยปุ่มข้อความ “บันทึก” บนปุ่มจะเป็นสีเทาเมื่อข้อมูลไม่มีการเปลี่ยนแปลง (Inactive) และจะเปลี่ยนเป็นสีขาวเมื่อข้อมูลมีการเปลี่ยนแปลง
2. ส่วนข้อมูลเกี่ยวกับการเรียน ออกแบบให้มีลักษณะเป็นปุ่มโดยกดปุ่ม “+” เพื่อเพิ่มข้อมูล และปุ่ม “x” เพื่อลบข้อมูลที่อยู่ภายในกรอบเดียวกัน
3. ออกแบบให้ปุ่มมีสีที่ต่างกันเพื่อให้ผู้ใช้เห็นถึงระดับความสำคัญที่ต่างกัน

ภาพที่ 13 หน้ารายงานปัญหา

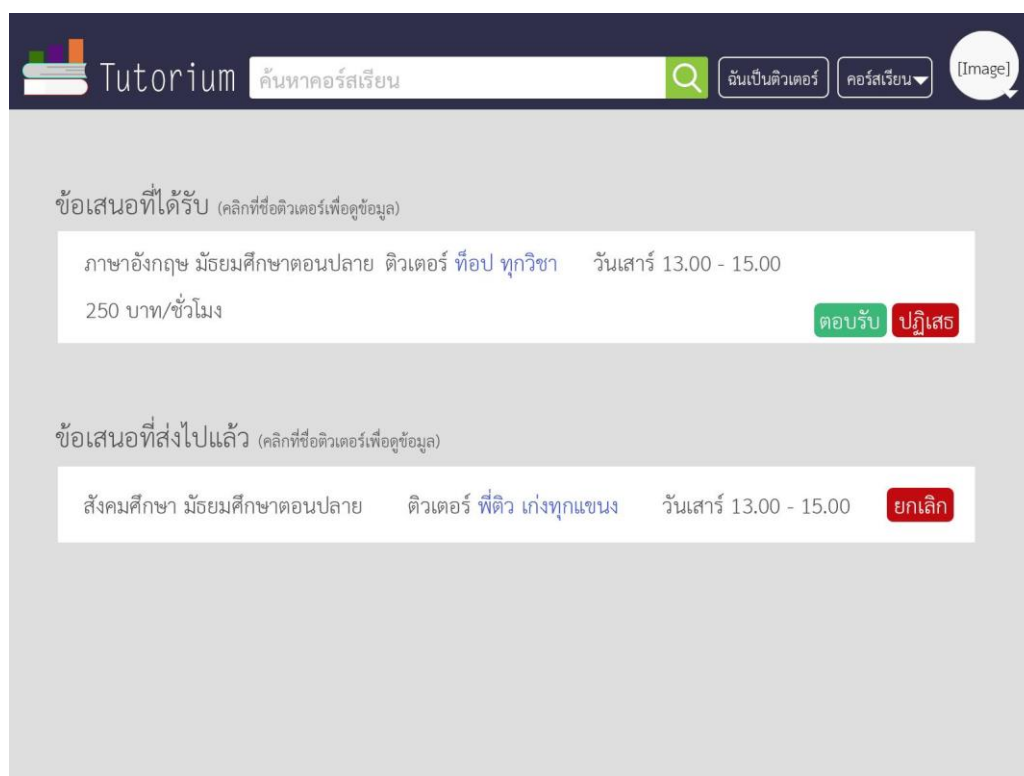
การออกแบบหน้ารายงานปัญหา แบ่งสาเหตุการออกแบบได้ดังนี้

1. ออกแบบให้มีส่วนประเภทของปัญหาเพื่อให้ผู้ใช้เลือกประเภทของปัญหาที่จะรายงาน
2. ออกแบบให้ส่วน “ชื่อผู้ใช้ที่ต้องการรายงาน” มีลักษณะเป็นช่องค้นหาเพื่อค้นหาเฉพาะรายชื่อที่มีในระบบ และจะแสดงส่วนนี้เมื่อประเภทของปัญหาคือ “รายงานผู้ใช้” เท่านั้น
3. ออกแบบให้ข้อความบนปุ่มบันทึกเป็นสีเทาและไม่สามารถกดปุ่มได้ (Inactive) เมื่อผู้ใช้กรอกข้อมูลไม่ครบ โดยข้อความจะเปลี่ยนเป็นสีขาวและสามารถกดได้ (Active) เมื่อกรอกข้อมูลครบแล้วเท่านั้น



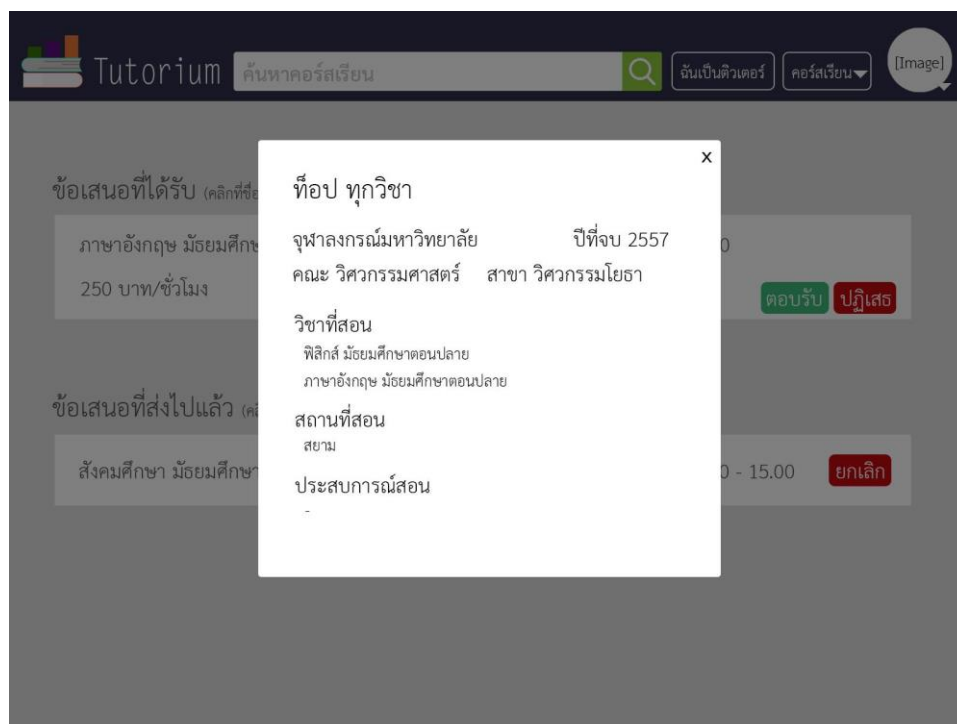
ภาพที่ 14 หน้าจัดการคอร์สเรียนสำหรับนักเรียน

ออกแบบให้แบ่งออกเป็นสองส่วนคือส่วนของคอร์สเรียนที่ได้ทำการตกลงกับติวเตอร์ไว้แล้ว โดยจะมีปุ่มให้จองเวลาเรียน ซึ่งสามารถจองได้เฉพาะวันและเวลาที่ตกลงกันไว้เท่านั้น และส่วนสำหรับคอร์สเรียนที่ได้จองเวลาไว้แล้ว จะมีปุ่มสำหรับยกเลิกการจอง และปุ่มเช็คชื่อเข้าเรียนที่ Inactive และจะ Active หลังจากถึงเวลาเรียนแล้วเท่านั้น



ภาพที่ 15 หน้าจัดการข้อเสนอคอร์สเรียน

ออกแบบให้แบ่งออกเป็นสองส่วนได้แก่ส่วนข้อเสนอที่ได้รับ ซึ่งจะแสดงข้อเสนอทั้งหมดที่ได้รับจากติวเตอร์ นักเรียนสามารถเลือกตอบรับหรือปฏิเสธข้อเสนอเหล่านี้ได้ และส่วนแสดงข้อเสนอที่นักเรียนเป็นผู้ส่งไปหาติวเตอร์



ภาพที่ 16 หน้าจอฟุดขึ้น (Pop-up) แสดงข้อมูลของติวเตอร์จากหน้าจัดการข้อเสนอคอร์สเรียน

ออกแบบให้เป็นหน้าจอฟุดขึ้น (Pop-up) มีพื้นหลังสีขาว เพื่อดึงความสนใจของผู้ใช้และแสดงว่าเป็นหน้าจอป๊อปอัพ โดยหน้าจอดีเดิมจะเป็นสีเทาเพื่อระบุว่าเป็นหน้าที่ Inactive

3.1.3 ระบบย่อยบัญชีผู้ให้บริการ (Tutor)

ขั้นตอนการสมัครเป็นติวเตอร์

1. ตรวจสอบข้อมูลส่วนตัว
2. กรอกข้อมูลเพิ่มเติม* และอัปโหลดหลักฐานการยืนยันตัวตน
(บัตรประจำตัวประชาชน หรือ บัตรประจำตัวนักเรียน / นักศึกษา หรือ พาสปอร์ต)
3. เลือกบัญชีธนาคารสำหรับรับค่าบริการ
4. ยืนยันและรอการตอบรับจากผู้ดูแลระบบ
* สามารถแก้ไขได้ในภายหลัง

ข้อมูลส่วนตัว

ชื่อ นามสกุล
 เพศ ระดับการศึกษา
 Facebook
 Line ID
 อีเมล
 โทรศัพท์

ข้อมูลด้านการศึกษา (กรอกถ้ายังศึกษาอยู่ในกรอบปี พ.ศ. ที่คาดว่าจะจบ)

โรงเรียน / มหาวิทยาลัย ปีที่จบ คณะ สาขา
 + เพิ่มข้อมูลด้านการศึกษา

วิชาที่สามารถสอนได้

วิชา ระดับชั้น

 + เพิ่มวิชา

ประสบการณ์สอนโดยสังเขป

สถานที่ที่สามารถสอนได้

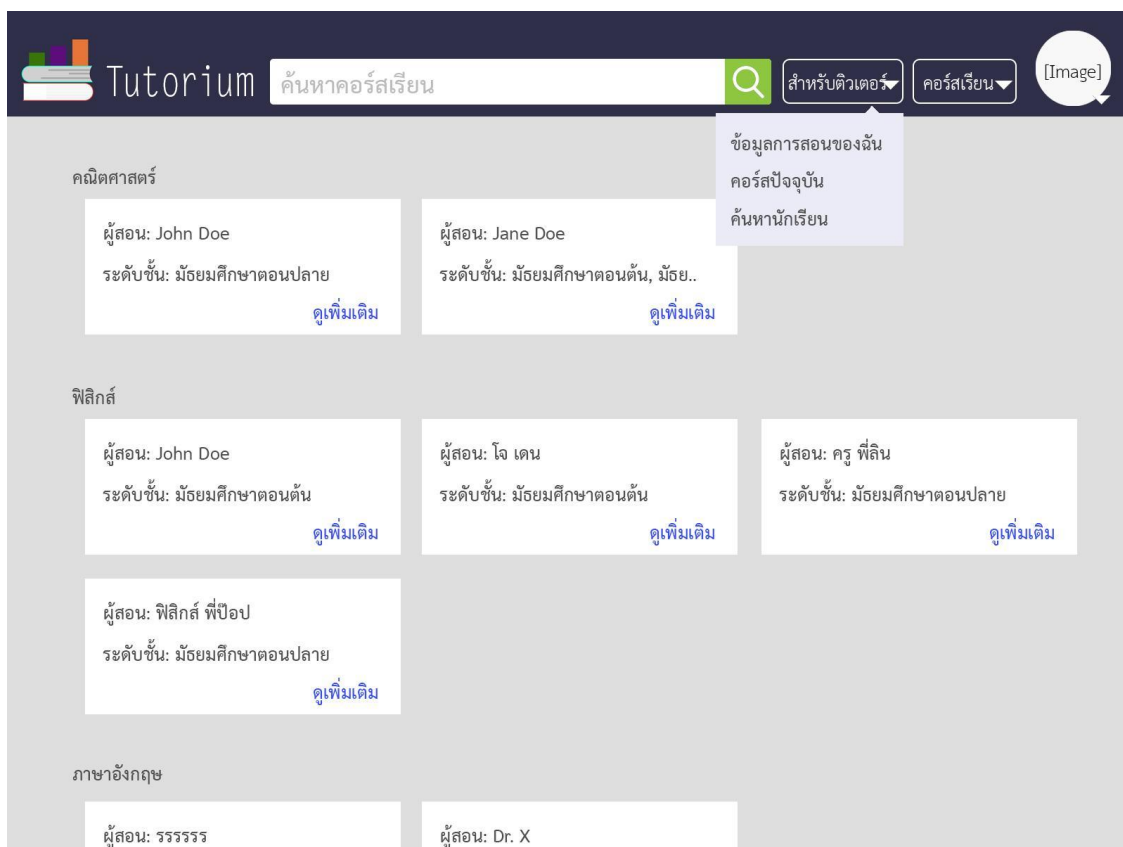
หลักฐานการยืนยันตัวตน

บัญชีธนาคารสำหรับรับค่าบริการ

☐ ฉันได้อ่านและยอมรับเงื่อนไขการให้บริการแล้ว

ภาพที่ 17 หน้าสมัครเป็นติวเตอร์

ออกแบบให้มีส่วนกรอกข้อมูลเพิ่มเติม ได้แก่ ข้อมูลด้านการศึกษา วิชาที่สามารถสอนได้ ประสบการณ์สอน สถานที่ที่สามารถสอนได้ หลักฐานการยืนยันตัวตน และบัญชีธนาคารสำหรับรับค่าบริการ เพื่อนำข้อมูลบางส่วนไปแสดงให้กับนักเรียนประกอบการตัดสินใจยื่นข้อเสนอให้ติวเตอร์



ภาพที่ 18 หน้าแรกของระบบสำหรับติวเตอร์

ออกแบบให้มีการเปลี่ยนแปลงจากปุ่ม “ฉันเป็นติวเตอร์” เป็น “สำหรับติวเตอร์” เพื่อแสดงรายการสำหรับติวเตอร์

Tutorium ค้นหาครุสเรียน

ข้อมูลการสอนของฉัฉน

ข้อมูลด้านการศึษา (หากกัฉฉศึกษาอยู่ใถ้กรอกปีพ.ศ. ที่คัฉฉจะจบ)

โรงเรียน / มหาวิทยาลัย ปีที่จบ คณะ สาขา

จุฬาลงกรณ์มหาวิทยาลัย 2557 วิศวกรรมศาสตร์ วิศวกรรมโยธา

+ เพิ่มข้อมูลด้านการศึษา

วิชาที่สัาสามารถสอนได้

วิชา ระดับชั้น

ฟิสิกส์ มัธยมศึษาตอนปลาย -

ภาษาอังกฤษ มัธยมศึษาตอนปลาย -

+ เพิ่มวิชา

ประสบการณ์สอนโดยสัาขเป

สถานที่ที่สัาสามารถสอนได้

สยาม X +

หลักฐานการยืนยันตัวฉฉ Browse ..

idcard.jpg x

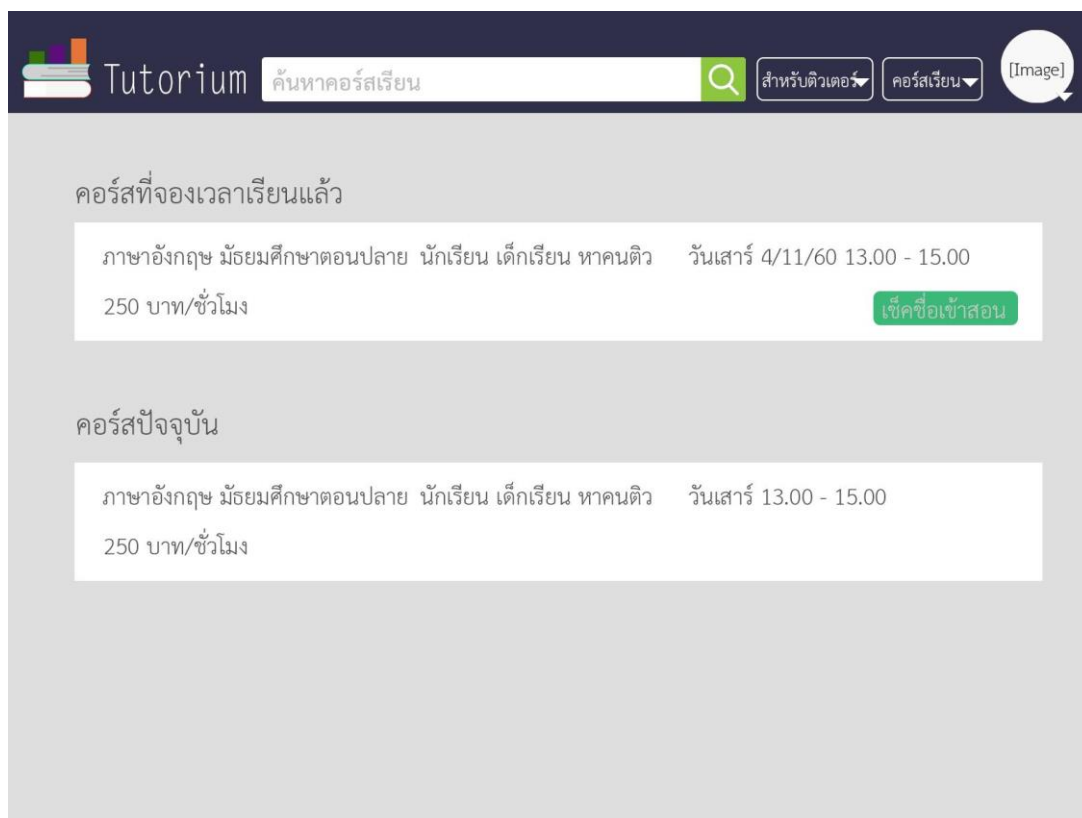
บัญชีธนาคารสำหรับรับค่าบริการ

(เลือกจากบัญชีธนาคารที่มีอยู่แล้ว)

+ เพิ่มบัญชี

ภาพที่ 19 หน้ากรอกข้อมูลการสอนของตัวเตอรื

หน้าสำหรับแก้ไขข้อมูลการสอนของตัวเตอรื ออกแบบให้ปุ่มบันทึกมีลักษณะเป็น Inactive เมื่อข้อมูลไม่มีการแก้ไข และจะเปลี่ยนเป็น Active เมื่อข้อมูลถูกแก้ไข



ภาพที่ 20 หน้าจัดการคอร์สสำหรับตัวต่อ

ออกแบบให้แบ่งเป็นสองส่วน ได้แก่ส่วน “คอร์สปัจจุบัน” แสดงรายการของคอร์สทั้งหมดที่ได้ทำการตกลงกับนักเรียนไว้ และส่วน “คอร์สที่จองเวลาเรียนแล้ว” แสดงรายการคอร์สทั้งหมดที่นักเรียนได้ทำการจองเวลาไว้ โดยมีปุ่ม “เช็คชื่อเข้าสอน” ในลักษณะ Inactive และจะเปลี่ยนเป็น Active หลังจากถึงเวลาสอนแล้วเท่านั้น

Tutorium ค้นหาคอร์สเรียน

ค้นหานักเรียน

ข้อเสนอที่ได้รับ

เด็กเรียน หาคณิศว (ชาย)
 วิชา ภาษาอังกฤษ มัธยมศึกษาตอนปลาย
 วันเสาร์ 13.00 - 15.00
 สถานที่ สยาม

ผลการค้นหา

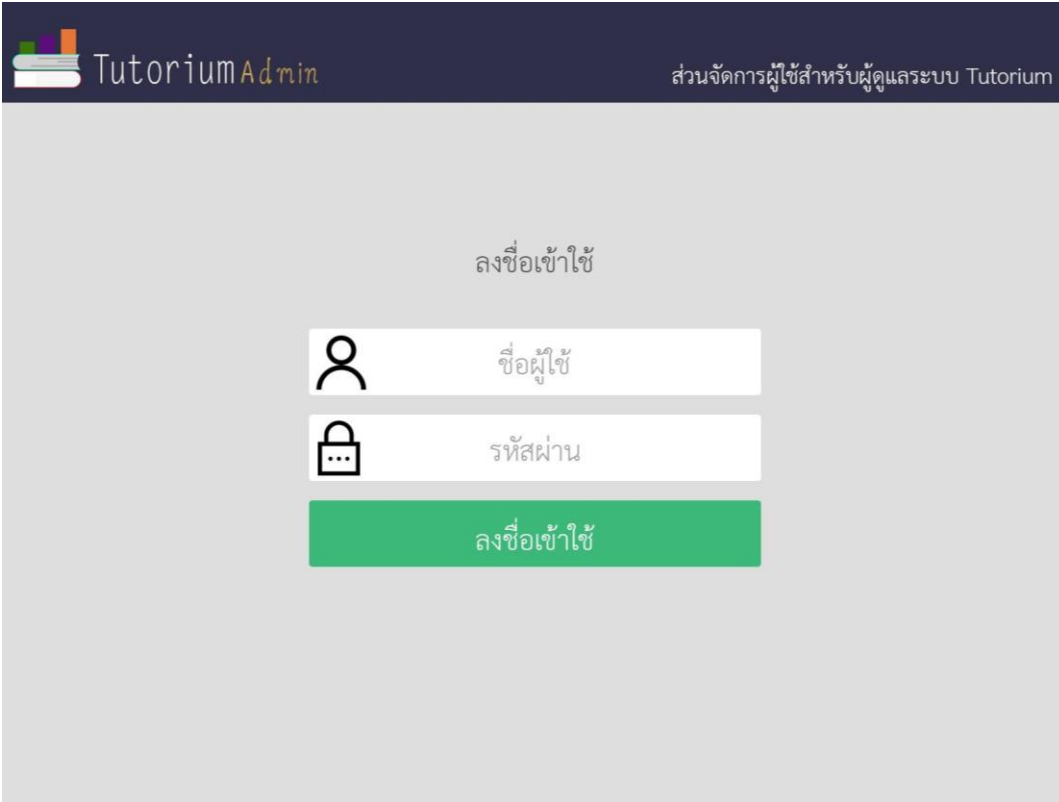
มานะ รักการเรียน (ชาย)
 วิชา คณิตศาสตร์ มัธยมศึกษาตอนต้น
 วันพุธ 17.00 - 19.00
 สถานที่ เซ็นทรัลพระราม 2

Kwan Suk (หญิง)
 วิชา วิทยาศาสตร์ ประถมศึกษาตอนปลาย
 วันอาทิตย์ 10.00 - 12.00
 สถานที่ บางนา

ภาพที่ 21 หน้าคั่นหานักเรียนสำหรับติวเตอร์

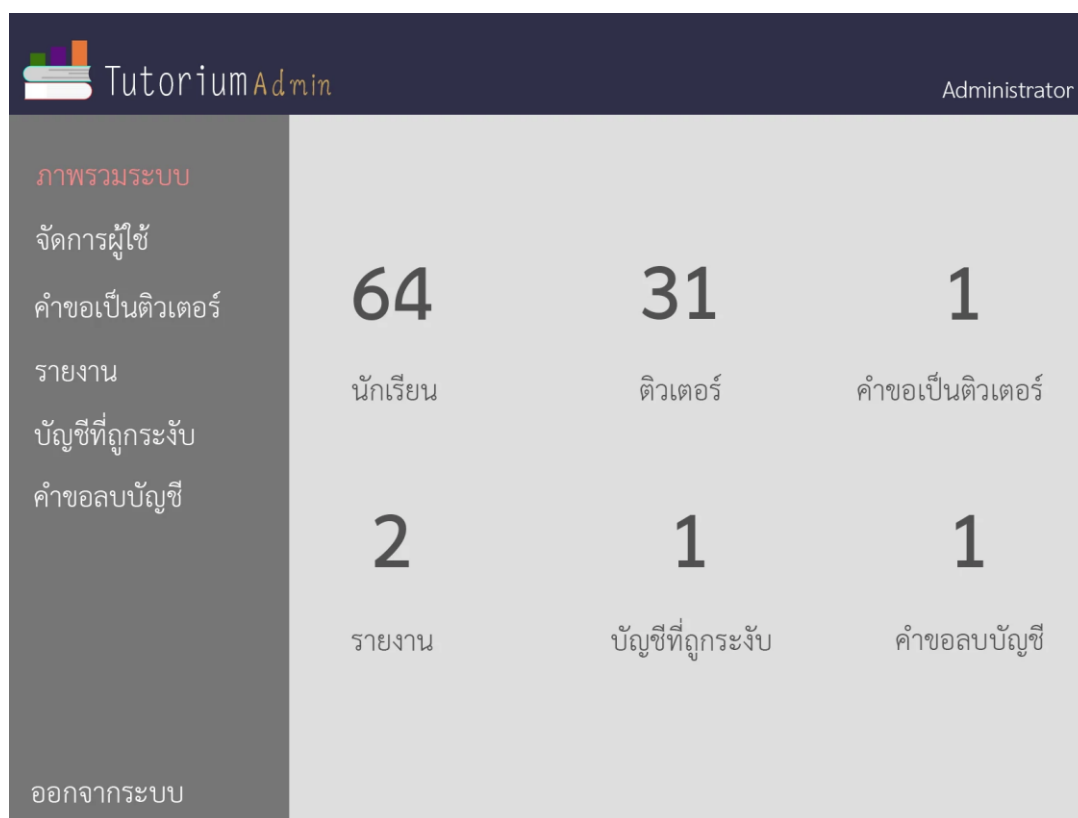
ออกแบบให้มีช่องสำหรับคั่นหานักเรียน มีส่วนแสดงข้อเสนอขอเรียนที่ได้รับจากนักเรียน และมีส่วนแสดงรายการนักเรียนทั้งหมดที่ตรงกับคำค้น ติวเตอร์สามารถยื่นข้อเสนอให้กับนักเรียนได้ผ่านทางหน้านี้

3.2 ระบบสำหรับบัญชีผู้ดูแลระบบ (administrator)



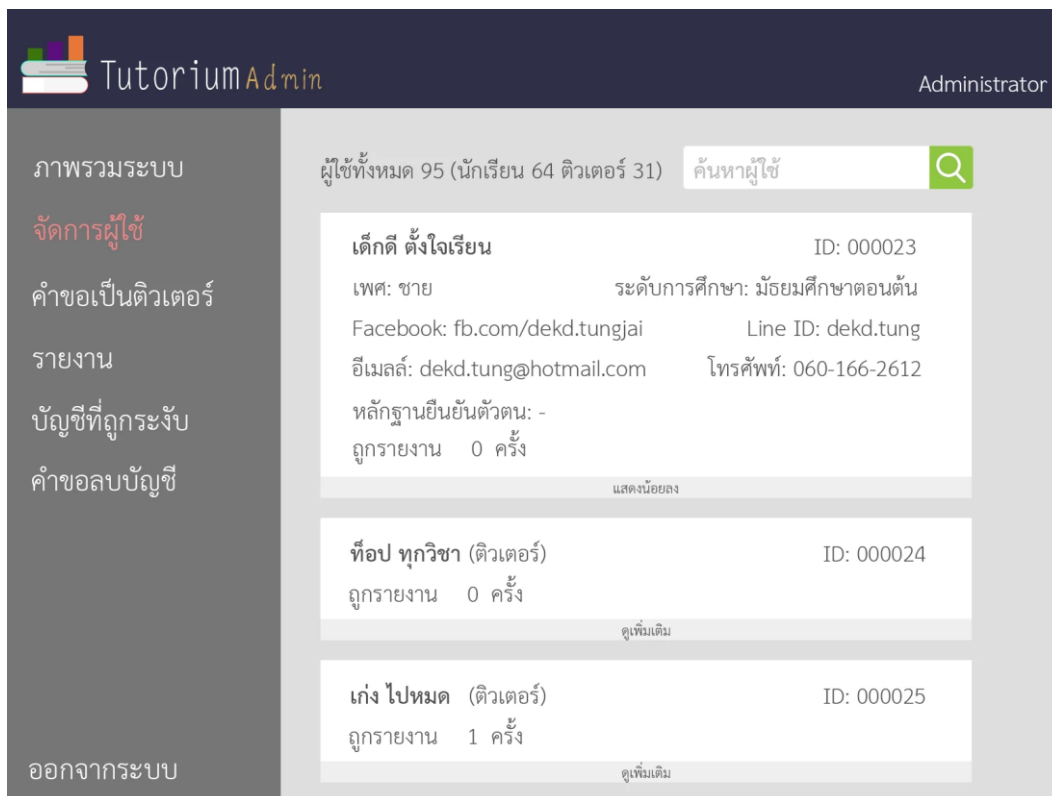
ภาพที่ 22 หน้าลงชื่อเข้าใช้สำหรับผู้ดูแลระบบ

ออกแบบเน้นความเรียบง่าย และความรวดเร็วในการเข้าสู่ระบบ มีช่องสำหรับกรอกชื่อผู้ใช้ รหัสผ่าน และปุ่มสำหรับเข้าระบบอย่างชัดเจน



ภาพที่ 23 หน้าแสดงภาพรวมของระบบ

ออกแบบเน้นความเข้าใจง่ายของภาพรวมระบบ โดยแสดงเป็นตัวเลข และมีป้ายกำกับด้านล่างของตัวเลข



TutoriumAdmin Administrator

ภาพรวมระบบ

จัดการผู้ใช้

คำขอเป็นติวเตอร์

รายงาน

บัญชีที่ถูกกระชับ

คำขอลบบัญชี

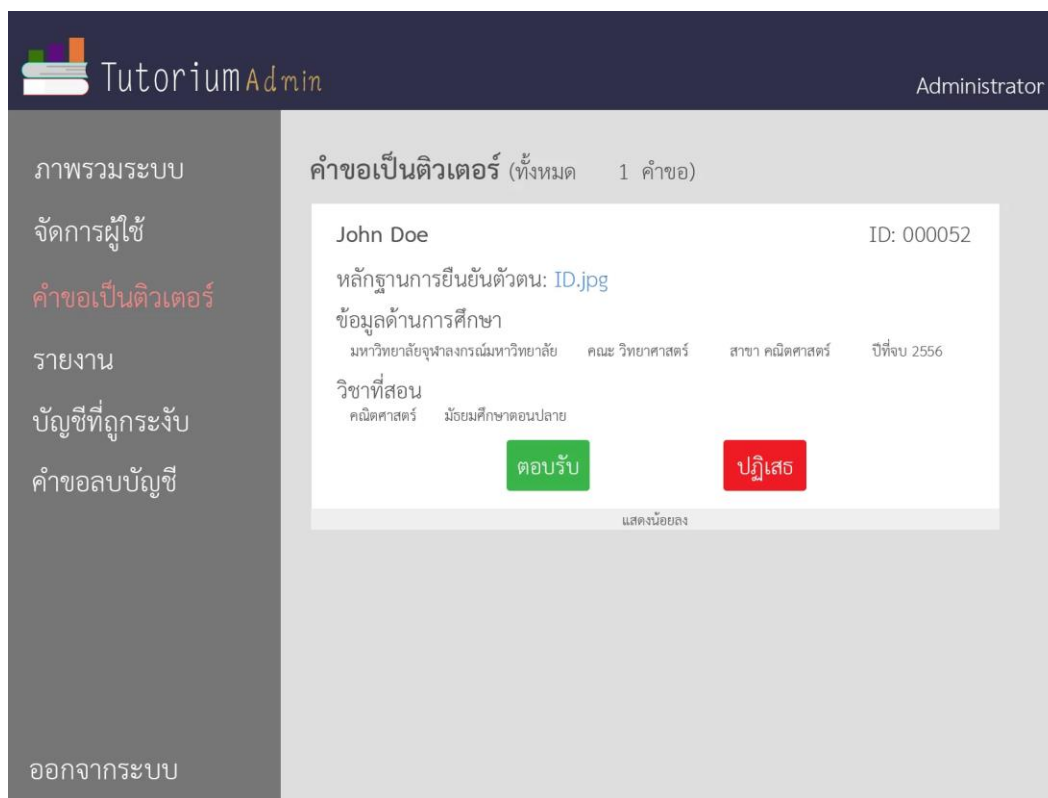
ออกจากระบบ

ผู้ใช้ทั้งหมด 95 (นักเรียน 64 ติวเตอร์ 31) ค้นหาผู้ใช้

เด็กดี ตั้งใจเรียน เพศ: ชาย Facebook: fb.com/dekd.tungjai อีเมลล์: dekd.tung@hotmail.com หลักฐานยืนยันตัวตน: - ถูกรายงาน 0 ครั้ง	ID: 000023 ระดับการศึกษา: มัธยมศึกษาตอนต้น Line ID: dekd.tung โทรศัพท์: 060-166-2612
แสดงน้อยลง	
ท็อป ทูริชา (ติวเตอร์) ถูกรายงาน 0 ครั้ง	ID: 000024
ดูเพิ่มเติม	
เก่ง ไปหมด (ติวเตอร์) ถูกรายงาน 1 ครั้ง	ID: 000025
ดูเพิ่มเติม	


ภาพที่ 24 หน้าจัดการผู้ใช้

ออกแบบให้แสดงรายการของผู้ใช้ทั้งหมดที่ตรงกับคำค้นในช่องค้นหา โดยแสดงในรูปแบบการ์ดที่สามารถขยายเพื่อดูข้อมูลเพิ่มเติมของผู้ใช้ได้ โดยเริ่มต้นจะแสดงในรูปแบบย่อเพื่อความกระชับและเข้าใจง่าย



ภาพที่ 25 หน้าจัดการคำขอเป็นติวเตอร์

หน้าแสดงคำขอเป็นติวเตอร์ ออกแบบให้แต่ละคำขออยู่ในรูปของการ์ดที่สามารถย่อหรือขยายได้ ภายในการ์ดจะแสดงข้อมูลเกี่ยวกับการสอนและหลักฐานการยืนยันตัวตนของผู้ส่งคำขอเพื่อประกอบการตัดสินใจตอบรับคำขอ และมีปุ่มสองปุ่ม โดยใช้สีเขียวเพื่อแสดงถึงการอนุมัติสำหรับปุ่มตอบรับ และสีแดงเพื่อเตือนให้ระวังในการกดปุ่มสำหรับปุ่มปฏิเสธ


TutoriumAdmin
Administrator

ภาพรวมระบบ
จัดการผู้ใช้
คำขอเป็นติวเตอร์
รายงาน
บัญชีที่ถูกระงับ
คำขอลบบัญชี

ออกจากระบบ

รายงาน (ทั้งหมด 2 รายงาน)

ไม่สามารถอัปโหลดหลักฐานได้ - Jane Doe (000028) 18/11/2017 13.24 (รายงานทั่วไป)

รายละเอียด: อัปโหลดหลักฐานไม่ได้ค่ะ [ดูเพิ่มเติม](#)

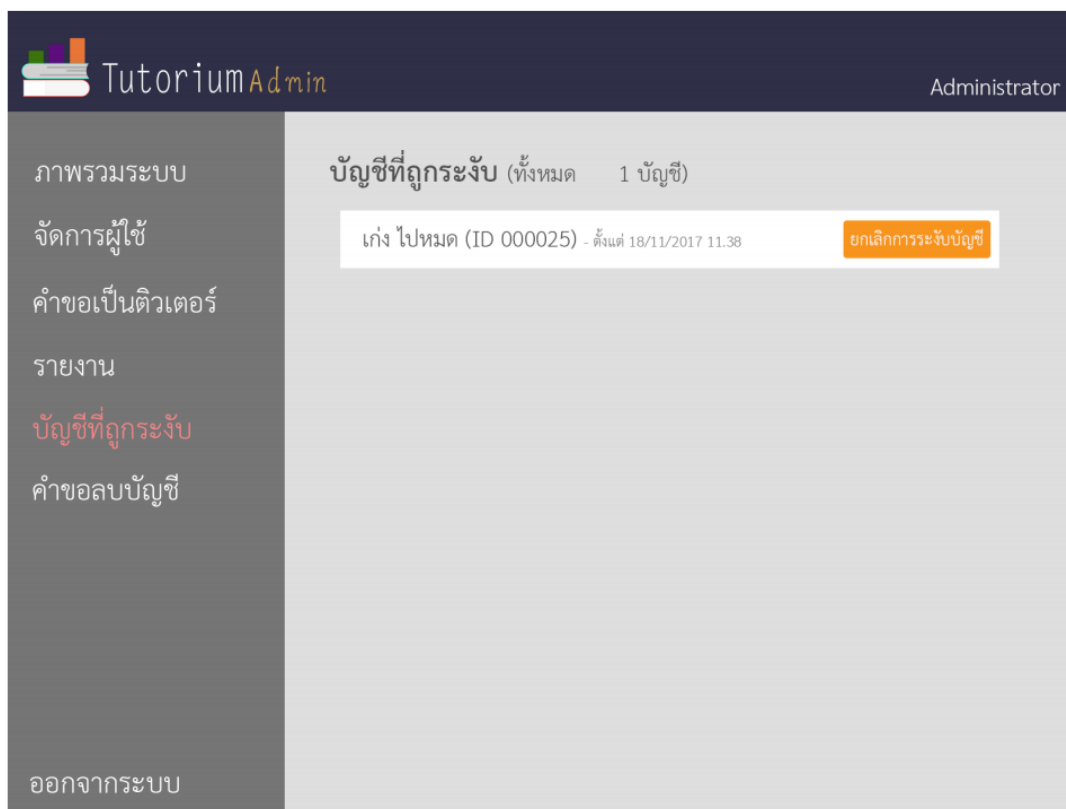
ติวเตอร์ไม่มาสอนครับ - เด็กเรียน หาคณิศว (000026) 17/11/2017 17.06 (รายงานผู้ใช้)

ผู้ใช้ที่ถูกรายงาน: เก่ง ไปหมด (ID 000025)
รายละเอียด: ครั้งที่ 2 แล้วครับ [ดูเพิ่มเติม](#)

ระงับบัญชีผู้ใช้ เก่ง ไปหมด (ID 000025) (โปรดใช้อย่างระมัดระวัง)

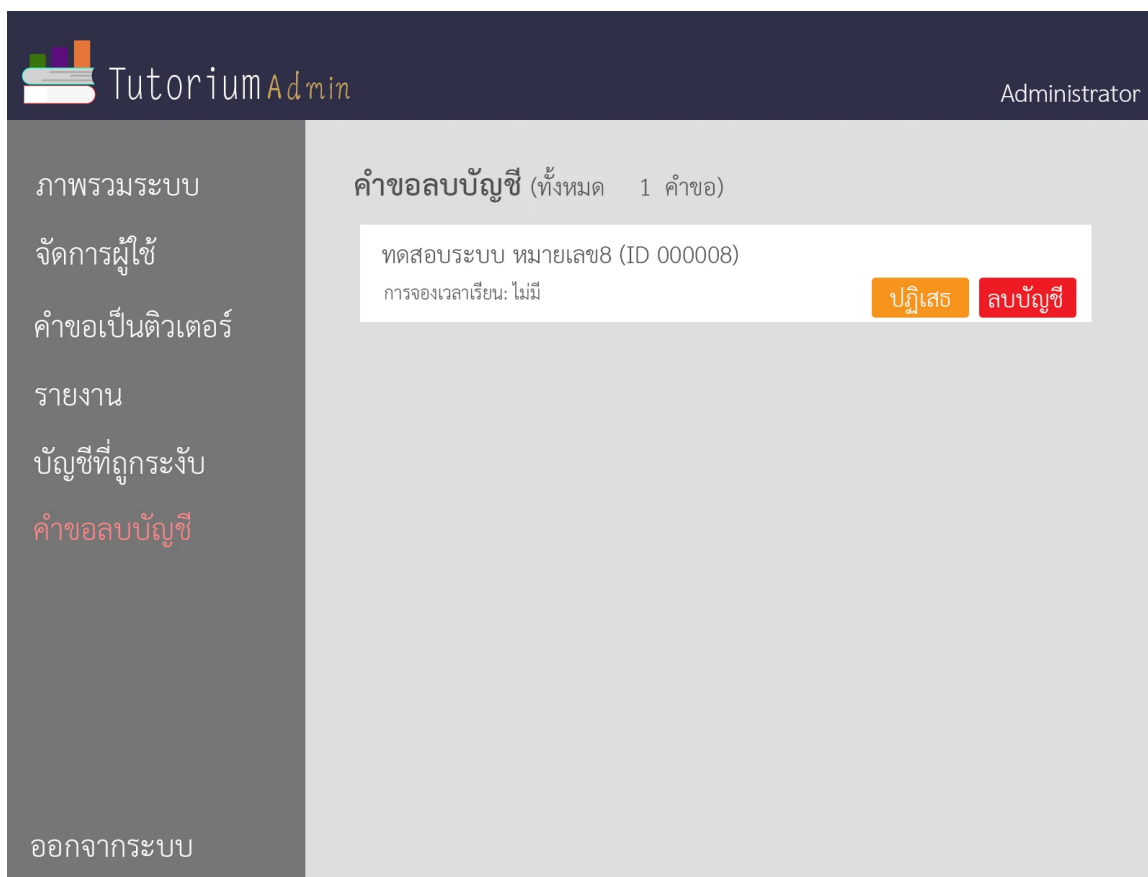
ภาพที่ 26 หน้าแสดงรายงาน

ออกแบบให้แต่ละรายงานอยู่ในรูปแบบของการ์ด เพื่อความประชับและเข้าใจง่าย โดยมีปุ่ม “ดูเพิ่มเติม” เพื่อใช้อ่านรายละเอียดเพิ่มเติมของรายงาน สำหรับรายงานประเภทรายงานผู้ใช้ จะมีปุ่มระงับบัญชีผู้ใช้ ออกแบบให้มีสีแดงและมีข้อความ “โปรดใช้อย่างระมัดระวัง” เพื่อเตือนให้ระวังในการใช้ปุ่ม



ภาพที่ 27 หน้าแสดงบัญชีผู้ใช้ที่ถูกระงับ

ออกแบบให้แสดงแต่ละบัญชีที่ถูกระงับในรูปแบบของการ์ด โดยแสดงเฉพาะชื่อผู้ใช้ รหัสผู้ใช้ และวันที่ที่ถูกระงับเพื่อความกระชับและเข้าใจง่าย มีปุ่ม “ยกเลิกการระงับบัญชี” อยู่ด้านข้างใช้เพื่อยกเลิกการระงับบัญชี ให้มีสีเหลืองเพื่อเตือนให้ระวัง เพราะผู้ใช้อย่างนี้อาจสร้างปัญหาซ้ำ ๆ อีกได้



ภาพที่ 28 หน้าแสดงคำขอลบบัญชีผู้ใช้

ออกแบบให้แสดงแต่ละคำขอในรูปแบบของการ์ด โดยจะแสดงข้อมูลชื่อผู้ใช้ รหัสผู้ใช้ และจำนวนคอร์สเรียนที่มีการจองอยู่ในขณะนั้น โดยมีปุ่ม “ปฏิเสธ” ใช้สำหรับปฏิเสธคำขอ มีสีเหลืองเพื่อเตือนให้ระวังในการใช้ปุ่ม และมีปุ่ม “ลบบัญชี” ใช้สำหรับลบบัญชี มีสีแดงเพื่อเตือนให้ระวังที่สุด เพราะไม่สามารถกู้คืนหรือย้อนกลับได้

บทที่ 4 รหัสเทียมสำหรับฟังก์ชันที่สำคัญในระบบ

ในบทนี้ ทีมผู้พัฒนาจะยกรหัสเทียมของฟังก์ชันที่สำคัญในระบบดังต่อไปนี้ เพื่อให้ผู้อ่านเห็นภาพของการออกแบบการทำงานของเมทอด ทั้งนี้ ทีมผู้พัฒนาจะแทนฟังก์ชันในการส่งข้อความแสดงข้อผิดพลาด (Error Message) ด้วยฟังก์ชัน print ในรหัสเทียมที่จะนำเสนอต่อไปนี้

4.1 การลงทะเบียนผู้ใช้

รหัสเทียมของเมทอด register

เมทอดนี้ถูกเรียกใช้เมื่อผู้เยี่ยมชม (External User) ลงทะเบียนผู้ใช้ระบบผ่านทางบัญชีผู้ใช้เฟสบุ๊คหรือบัญชีผู้ใช้ไลน์เพื่อเปลี่ยนสถานะเป็นผู้รับบริการ (Student)

```
func register(conditionAccept, registerType, title, name, surname, birthdate, description) {
    if conditionAccept == false {
        print("กรุณายอมรับเงื่อนไขการใช้งานก่อนสมัครใช้งาน")
        return
    }

    if registerType = 'Line' {
        accountID = lineLogin()

        registerSuccess = INSERT INTO 'Student'
                               VALUES(AccountType = 'registerType',
                                       AccountID = 'accountID',
                                       title = 'title',
                                       name = 'name',
                                       surname = 'surname',
                                       birthdate = 'birthdate',
                                       description = 'description')

    } else {
        accountID = facebookLogin()
```

```

registerSuccess = INSERT INTO 'Student'
                    VALUES(AccountType = 'registerType',
                             AccountID = 'accountID',
                             title = 'title',
                             name = 'name',
                             surname = 'surname',
                             birthdate = 'birthdate',
                             description = 'description')
}

if registerSuccess == false {
    print("การลงทะเบียนล้มเหลว โปรดลองใหม่อีกครั้ง")
    return
}

return
}

```

4.2 การลงชื่อเข้าสู่ระบบ

รหัสเทียมของเมทอด login

เมทอดนี้ถูกเรียกใช้เมื่อผู้รับบริการ (Student) ซึ่งเป็นผู้ใช้ในระบบต้องการลงชื่อเข้าสู่ระบบผ่านทางบัญชีผู้ใช้เฟซบุ๊กหรือบัญชีผู้ใช้ไลน์

```

func login(loginType, username, password) {
    if loginType == 'facebook' {
        token, id = performFacebookLogin()
    } else if loginType == 'line' {
        token, id = performLineLogin()
    } else {
        id= SELECT username
            FROM 'Admin'
    }
}

```

```

WHERE username = 'username'
AND password = 'password'
}

SESSIONS['token'], SESSION['id'] = token, id
return
}

```

4.3 การส่งคำร้องขอสมัครเป็นผู้ให้บริการ

รหัสเทียมของเมทอด requestTutorAccount

เมทอดนี้ถูกเรียกใช้เมื่อผู้ให้บริการ (Student) ที่ไม่มีสถานะเป็นผู้ให้บริการ (Tutor) ต้องการเปลี่ยนสถานะเป็นผู้ให้บริการ โดยส่งคำร้องขอไปให้ผู้ดูแลระบบ (Admin) ทำการพิจารณาอนุมัติต่อไป

```

func requestTutorAccount(studentID, detail, proofDocument) {
  if proofDocument == empty {
    print("กรุณาแนบเอกสารยืนยันก่อนสมัครเป็นติวเตอร์")
    return
  }

  request = INSERT INTO 'TutorRequest'
    VALUES(studentID = 'studentID',
      detail = 'detail'
      proof = 'proofDocument')

  return
}

```

4.4 การมอบสถานะผู้ให้บริการ

รหัสเทียมสำหรับเมทอด grantTutorRole

เมื่อก่อนนี้ถูกเรียกใช้เมื่อผู้ดูแลระบบ (Admin) ต้องการอนุมัติหรือปฏิเสธคำร้องขอเป็นผู้ให้บริการ (Tutor) ของผู้รับบริการ (Student)

```
func grantTutorRole(studentID) {
    tutorRequest = SELECT *
                      FROM 'TutorRequest'
                      WHERE studentID = 'studentID'

    tutor = SELECT *
             FROM 'Tutor'
             WHERE studentID = 'studentID'

    if tutorRequest == empty {
        print("ผู้ใช้งานคนดังกล่าวไม่ได้ส่งคำขอยืนยันสมัครเป็นติวเตอร์")
        return
    } else if tutor != empty {
        print("ผู้ใช้งานคนดังกล่าวมีสถานะเป็นติวเตอร์อยู่แล้ว")
    }

    grantTutor= INSERT INTO 'Tutor'
                VALUES(studentID='studentID')

    return
}
```

4.5 การเพิ่มเวลาว่างสอน

รหัสเทียมสำหรับเมื่อก่อน addAvailableTime

เมื่อก่อนนี้ถูกเรียกใช้เมื่อผู้ให้บริการ (Tutor) ต้องการเพิ่มเวลาว่างสำหรับสอนในตารางเวลาของตน

```
func addAvailableTime(tutorID, date, startTime, endTime) {
    reservation = SELECT *
                  FROM 'Reservation'
                  WHERE tutorID = 'tutorID'
                    AND date = 'date'
```



```

        AND startTime = 'startTime'
        AND endTime = 'endTime'
    if reservation != empty {
        print("คุณมีนัดหมายกับนักเรียนในช่วงเวลานี้แล้ว กรุณาเลือกช่วงเวลาอื่น")
        return
    }

    addAvailableTime = INSERT INTO 'AvailableTime'
        VALUES (tutorID = 'tutorID',
            date = 'date',
            startTime = 'startTime',
            endTime = 'endTime')

    return
}

```

4.6 การจองเวลาเรียน

รหัสเทียมสำหรับเมทอด reserveTime

เมทอดนี้ถูกเรียกใช้เมื่อผู้รับบริการ (Student) ต้องการจองเวลาเรียนในคอร์สที่ตนได้จับคู่ไว้แล้ว นอกจากนี้ เมื่อผู้รับบริการจองเวลาเรียนจะต้องมีการชำระค่ามัดจำเป็นร้อยละ 30 ของค่าบริการทั้งหมดที่คิดตามจำนวนชั่วโมงที่ได้จองเวลาเรียนไป

```

func reserveTime(studentID, tutorID, date, startTime, endTime) {
    if !(date in available.date && startTime in available.startTime && endTime in
available.endTime) {
        print("คิวเตอร์ไม่ว่างในช่วงเวลาที่ท่านเลือก")
        return
    }

    paySuccess = payCourseFee()

    if paySuccess == false {

```

```

    print("การชำระเงินมัดจำไม่สมบูรณ์ กรุณาทำการใหม่อีกครั้ง")
    return
}

reservation = INSERT INTO 'Reservation'
              VALUES(tutorID = 'tutorID',
                      studentID = 'studentID',
                      date = 'date',
                      startTime = 'startTime',
                      endTime = 'endTime')

notifyTutor(tutorID)

return
}

```

4.7 การยกเลิกการจองเวลาเรียน

รหัสเทียมของเมทอด cancelReservation

เมทอดนี้ถูกเรียกใช้เมื่อผู้รับบริการ (Student) ต้องการยกเลิกการจองเวลาเรียนที่ตนได้จองไว้

```

func cancelReservation(studentID, reserveID) {
    reservation = SELECT *
                  FROM 'Reservation'
                  WHERE reserveID = 'reserveID'
                  AND studentID = 'studentID'

    if reservation == empty {
        print("คุณไม่มีการนัดหมายในช่วงเวลานี้")
        return
    }

    deleteReservation = DELETE FROM 'Reservation'

```

```

WHERE reserveID = 'reserveID'
AND studentID = 'studentID'

return
}

```

4.8 การค้นหาผู้ให้บริการ

รหัสเทียมของเมทอด searchTutor

เมทอดนี้ถูกเรียกใช้เมื่อผู้รับบริการ (Student) ต้องการสืบค้นผู้ให้บริการ (Tutor)

```

func searchTutor(filter) {
    courses = querySelectCourses(filter)
    return courses
}

```

4.9 การค้นหาผู้รับบริการ

รหัสเทียมของเมทอด searchStudent

เมทอดนี้ถูกเรียกใช้เมื่อผู้ให้บริการ (Tutor) ต้องการสืบค้นผู้รับบริการ (Student)

```

func searchTutor(filter) {
    students = querySelectStudent(filter)
    return students
}

```

4.10 การส่งคำร้องขอรับบริการ

รหัสเทียมของเมทอด sendCourseRequest

เมทอดนี้ถูกใช้เมื่อผู้รับบริการ (Student) ต้องการส่งคำร้องขอรับบริการจากผู้ให้บริการ (Tutor)

```

func sendCourseRequest(studentID, tutorID, subject, level) {
    courseRequest = INSERT INTO 'CourseRequest'

```

```

VALUES(studentID = 'studentID',
        tutorID = 'tutorID',
        subject = 'subject',
        level = 'level')

if courseRequest != empty {
    notifyTutor(studentID, tutorID)
}

return
}

```

4.11 การสร้างข้อเสนอคอร์สเรียน

รหัสเทียมของเมทอด createCourseOffer

เมทอดนี้ถูกเรียกใช้เมื่อผู้ให้บริการ (Tutor) ต้องการสร้างข้อเสนอคอร์สเรียน

```

func createCourseOffer(tutorID, subject, price, description, level) {
    if tutorID == empty || subject == empty || price == empty || description == empty ||
level = empty {
        print("ข้อมูลที่กรอกไม่ถูกต้อง กรุณากรอกข้อมูลให้ครบถ้วน")
        return
    } else if price < 150 {
        print("ข้อมูลที่กรอกไม่ถูกต้อง กรุณากรอกราคามากกว่า 150 บาท/ชั่วโมง")
        return
    }

    createCourse = INSERT INTO 'Course'
        VALUES(tutorID = 'tutorID',
                subject = 'subject',
                price = 'price',
                description = 'description',
                level = 'level')
}

```

```

return
}

```

4.12 การลบข้อเสนอคอร์สเรียน

รหัสเทียมของเมทอด deleteCourseOffer

เมทอดนี้ถูกเรียกใช้เมื่อผู้ให้บริการ (Tutor) ต้องการลบข้อเสนอคอร์สเรียน

```

func deleteCourseOffer(tutorID, courseID) {
    course = SELECT *
        FROM 'Course'
        WHERE CourseID = 'courseID'

    if course == empty {
        print("ไม่มีคอร์สเรียนนี้ในระบบ")
        return
    }

    deleteCourse = DELETE FROM 'Course'
        WHERE CourseID = 'courseID'
        AND tutorID = 'tutorID'

    return
}

```

4.13 การส่งข้อเสนอคอร์สเรียน

รหัสเทียมของเมทอด sendCourseOffer

เมทอดนี้ถูกเรียกใช้เมื่อผู้ให้บริการ (Tutor) ต้องการส่งข้อเสนอคอร์สเรียนไปยังผู้รับบริการ (Student)

```

func sendCourseOffer(courseID, studentID) {

```

```

course= SELECT *
        FROM 'Course'
        WHERE CourseID = 'courseID'

notifyOfferToStudent(course, studentID)

return
}

```

4.14 การแก้ไขข้อเสนอคอร์สเรียน

รหัสเทียมของเมทอด editCourseOffer

เมทอดนี้ถูกเรียกใช้เมื่อผู้ให้บริการ (Tutor) ต้องการแก้ไขข้อมูลภายในข้อเสนอคอร์สเรียนที่ตนได้สร้างไว้

```

func editCourseOffer(tutorID, courseID, subject, price, description, level) {
    course = SELECT *
            FROM 'Course'
            WHERE CourseID = 'courseID'

    if course == empty {
        print("ไม่มีคอร์สเรียนนี้ในระบบ")
        return
    } else if tutorID == empty || subject == empty || price == empty || description == empty
    || level == empty {
        print("ข้อมูลที่กรอกไม่ถูกต้อง กรุณากรอกข้อมูลให้ครบถ้วน")
        return
    } else if price < 150 {
        print("ข้อมูลที่กรอกไม่ถูกต้อง กรุณากรอกราคามากกว่า 150 บาท/ชั่วโมง")
        return
    }

    updateCourse = UPDATE 'Course'
                  SET subject = 'subject',

```

```

        price = 'price',
        description = 'description',
        level = 'level'
    WHERE CourseID = 'courseID'
        AND tutorID = 'tutorID'

    return
}
```

4.15 การชำระค่าบริการ

รหัสเทียมของเมท็อด payCourseFee

เมท็อดนี้ถูกเรียกใช้เมื่อผู้รับบริการทำการจองเวลาเรียน (มีการเรียกเมท็อด reserveTime) และเมื่อผู้รับบริการทำการเช็คชื่อเข้าเรียนครั้งแรก (มีการเรียกเมท็อด checkAttendance) โดยสามารถเลือกชำระค่าบริการผ่านทางบัญชีธนาคารหรือบัตรเครดิต

```

func payCourseFee(studentID, paymentMethod, price) {
    if paymentMethod == 'bankTransfer' {
        bankAccountNo = SELECT 'bankAccountNo'
                        FROM 'BankAccountPayment'
                        WHERE studentID = 'studentID'

        success = bankTransferPaymentRequest(bankAccountNo, price)
    } else {
        creditCardNo = SELECT 'creditCardNo'
                      FROM 'CreditCardPayment'
                      WHERE studentID = 'studentID'

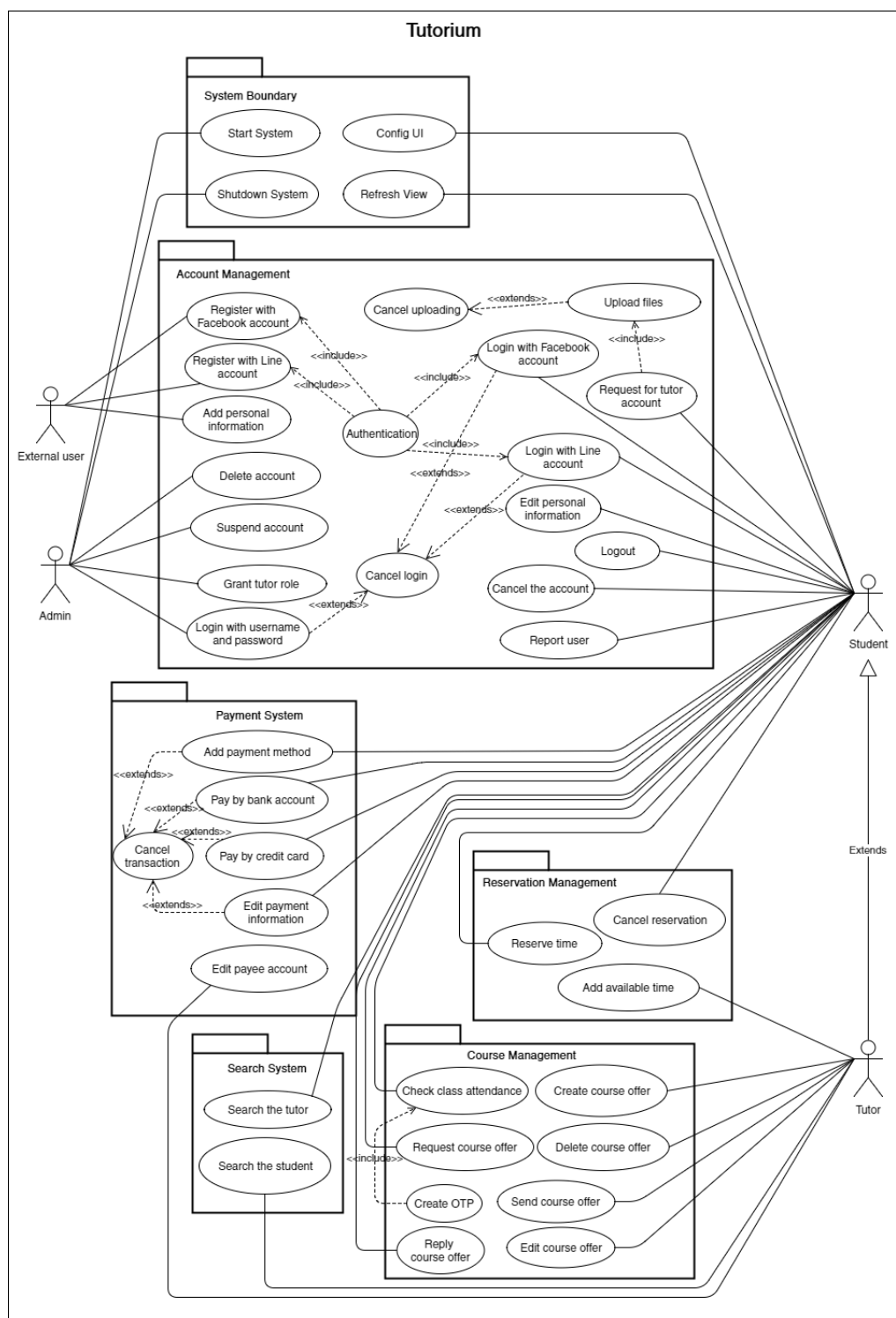
        success = creditCardPaymentRequest(creditCardNo, price)
    }

    if success == false {
        print("เกิดข้อผิดพลาด กรุณาชำระเงินใหม่อีกครั้ง")
    }
}
```

```
}  
  
return  
}
```


ภาคผนวก ก แผนภาพกรณีใช้งานบาว์นดารี

ในขั้นตอนการออกแบบระบบ ทีมผู้พัฒนาได้ออกแบบแผนภาพกรณีใช้งานที่คำนึงถึงการออกแบบ โดยมีการรวมกรณีใช้งานบาว์นดารี (Boundary Use Case) เข้ามาในระบบ



ภาพที่ 29 แผนภาพกรณีใช้งานบาว์นดารี