**รายงานโครงงาน**

เรื่อง Application KMITL News

**นำเสนอ**

อ. ปริญญา เอกปริญญา

**โดย**

63010366 นาย ถิรวัฒน์ จิตรานุเคราะห์

63010422 นาย ธนภูมิ ชัยพรรณา

63010487 นาย นพสิทธิ์ จันทรวีระกุล

63010519 นาย นันทวัฒน์ รัตนเรืองวิมาณ

63010584 นาย ปัญญวัฒน์ ธนทัตธาดา

63010633 นาย พนธกร สุจิตชวานนท์

63010656 นาย พศิน แก้วนิล

โครงงานนี้เป็นส่วนหนึ่งของการศึกษา

วิชา 01076024 SOFTWARE ARCHITECTURE AND DESIGN ปีการศึกษา 2565

**บทคัดย่อ**

**ชื่อโครงงาน** KMITL News

**ผู้จัดทำโครงการ** 63010366 นาย ถิรวัฒน์ จิตรานุเคราะห์

63010422 นาย ธนภูมิ ชัยพรรณา

63010487 นาย นพสิทธิ์ จันทรวีระกุล

63010519 นาย นันทวัฒน์ รัตนเรืองวิมาณ

63010584 นาย ปัญญวัฒน์ ธนทัตธาดา

63010633 นาย พนธกร สุจิตชวานนท์

63010656 นาย พศิน แก้วนิล

**อาจารย์ที่ปรึกษา** อ.ปริญญา เอกปริญญา

**ปีการศึกษา** 2565

----------------------------------------------------------------------------------

โครงงาน Application KMITL News มีแนวคิดมาจากปัญหาในชีวิตประจำวันที่ทางผู้พัฒนาได้พบเจอ ในปัจจุบันหากมีความต้องการติดตามข่าวสารจากทางสถาบันได้อย่างครบถ้วน นักศึกษามีความจำเป็นที่ต้องติดตามเพจเฟซบุ๊กมากกว่า 10 เพจเพื่อตามข่าวได้ครบซึ่งในแต่ละเพจประกอบด้วยโพสหลากหลายเรื่องต่างกันออกไปและ content ในเพจนั้นประกอบทั้งที่นักศึกษาอยากทราบและไม่อยากทราบ อีกทั้งในบางครั้งข่าวสารที่มาจากอาจารย์ของแต่ละคณะนั้นไม่ได้มีการโพสลงไปในเพจแต่ถูกโพสลงไปในเว็บไซต์ของทางสถาบัน ถ้าหากทางนักศึกษาจำเป็นต้องเปิดเข้าไปดูด้วยตัวเองซึ่งสร้างคววามยุ่งยากและอาจตกหล่นข่าวสารที่จำเป็นได้

ทางผู้พัฒนาได้เล็งเห็นถึงปัญหาเหล่านี้ ด้วยความต้องการที่จะเพิ่มความสะดวกสบายในการใช้งานและแก้ไขปัญหาให้นักศึกษาได้รับข่าวสารอย่างครบถ้วนโดยพัฒนา application KMITL News ที่จะรวบรวมข่าวสารจากทั้งอาจารย์และสถาบันมารวมกันในที่เดียวและยังทำให้ผู้ใช้งานสามารถกรองข่าวที่ตัวเองสนใจได้ โดยข่าวสารนั้นถูกโพสได้ทั้งแอดมินและผู้ใช้งานแต่ผู้ใช้งานจะต้องผ่านการคัดกรองเพื่อให้ข่าวภายในแอพมีความถูกต้องมากที่สุด

**สารบัญ**

|  |  |
| --- | --- |
| **เรื่อง** | **หน้า** |
| **บทที่ 1 บทนำ** | **1** |
| **1.1 ที่มาของโครงการ** |  |
| **1.2 จุดประสงค์ของโครงการ** |  |
| **บทที่ 2 ภาพรวมและการออกแบบระบบ** | **2** |
| **2.1 ภาพรวมขั้นตอนการทำงานของระบบ** |  |
| **2.1.1 ฝั่งผู้ใช้งาน** |  |
| **2.1.1.1 Login/Register** |  |
| **2.1.1.2 OTP** |  |
| **2.1.1.3 Filter** |  |
| **2.1.1.4 Tags display page** | **3** |
| **2.1.1.5 Profile page** |  |
| **2.1.1.5.1 Profile ตัวเอง** |  |
| **2.1.1.5.1.1 แก้ไขโปรไฟล์** |  |
| **2.1.1.5.2 Profile คนอื่น** |  |
| **2.1.2 ฝั่งแอดมิน** | **4** |
| **2.1.2.1 แอดมิน Display page** |  |
| **2.1.2.1.1 Admin Filter** |  |
| **2.1.2.2 แอดมิน Post approve** |  |
| **2.1.2.3 แอดมิน Report** |  |
| **2.1.2.4 แอดมิน Verified** |  |
| **2.2 การออกแบบหน้า** | **5** |
| **2.2.1 ออกแบบหน้าผู้ใช้งาน** |  |
| **2.2.1.1 ออกแบบหน้า Login/Register** |  |
| **2.2.1.2 ออกแบบหน้า Login** | **6** |
| **2.2.1.3 ออกแบบหน้า Register** | **7** |
| **2.2.1.4 ออกแบบหน้า Display feed** | **8** |
| **2.2.1.5 ออกแบบหน้า Filter** | **9** |
| **2.2.1.6 หน้า Profile** | **10** |
| **2.2.1.7 ออกแบบส่วนเสริม** | **12** |

|  |  |
| --- | --- |
| **เรื่อง** | **หน้า** |
| **2.2.2 ออกแบบหน้าแอดมิน** | **13** |
| **2.2.2.1 ออกแบบ Display feed admin** |  |
| **2.2.2.2 ออกแบบหน้า Filter ของแอดมิน** | **14** |
| **2.2.2.3 ออกแบบหน้า Post approve ของแอดมิน** | **15** |
| **2.2.2.4 ออกแบบหน้า report ของแอดมิน** | **16** |
| **2.2.2.5 ออกแบบหน้า verified account ของแอดมิน** | **17** |
| **บทที่ 3 การประยุกต์บทเรียน** | **18** |
| **3.1 Event Storming** |  |
| **3.1.1 Collect Domain Events** |  |
| **3.1.2 Refine Domain Events** | **19** |
| **3.1.3 Track cause รวมกับการใส่ aggregates และ จัดการโยงลำดับ** | **20** |
| **3.2 UML** | **21** |
| **3.2.1 UML Component** |  |
| **3.2.2 domain model** | **22** |
| **3.2.2.2 ContentContext** |  |
| **3.2.2.3 AdministrationContext** | **23** |
| **3.3 Quality Attribute** | **24** |
| **3.3.1 Availability** |  |
| **3.3.1.1 สามารถ Display feed** |  |
| **3.3.1.2 ความสามารถในการเชื่อมต่อ Database** |  |
| **3.3.1.3 แสดง Tags** | **25** |
| **3.3.1.4 จัดการ content** |  |
| **3.3.2 Integrability** |  |
| **3.3.2.1 Pub.dev** |  |
| **3.3.3 Modifiability** | **26** |
| **3.3.3.1 การแก้ไข code ในส่วน Backend** |  |
| **3.3.3.2 การแก้ไข code ในส่วน frontend** |  |
| **3.3.4 Performance** | **27** |
| **3.3.4.1 การส่ง request ข้อมูล** |  |

|  |  |
| --- | --- |
| **เรื่อง** | **หน้า** |
| **3.4 Software Architectural Style** | **28** |
| **3.4.1 MVC** |  |
| **3.4.2 Plug-in** |  |
| **3.4.3 Representational State Transfer (REST)** |  |
| **3.5 Design pattern** | **29** |
| **3.5.1 Façade** |  |
| **3.5.2 Iterator** | **30** |
| **บทที่ 4 การพัฒนา** | **31** |
| **4.1 Frontend** |  |
| **4.1.1 ภาษาที่ใช้พัฒนา Frontend** |  |
| **4.1.2 Main Page** |  |
| **4.1.2.1 userdisplay.dart** |  |
| **4.1.2.2 post\_page.dart** | **32** |
| **4.1.2.3 editprofilepage.dart** |  |
| **4.1.2.4 othersprofilepage.dart** | **33** |
| **4.1.2.5 userprofilepage.dart** |  |
| **4.1.2.6 home.dart** | **34** |
| **4.1.2.7 login.dart** |  |
| **4.1.2.8 register.dart** | **35** |
| **4.1.2.9 postapprove.dart** | **36** |
| **4.1.2.10 postreport.dart** | **37** |
| **4.1.2.11 userreport.dart** | **38** |
| **4.1.2.12 userverified.dart** |  |
| **4.1.3 Widget** | **39** |
| **4.1.3.1 circle\_button.dart** |  |
| **4.1.3.2 create\_post\_contrainer.dart** |  |
| **4.1.3.3 icbutton.dart** | **40** |
| **4.1.3.4** **profile\_container.dart** |  |

|  |  |
| --- | --- |
| **เรื่อง** | **หน้า** |
| **4.1.3.5 navigation\_drawer.dart** | **41** |
| **4.1.3.6 post\_contrainer.dart** |  |
| **4.1.3.7 tag\_button.dart** | **42** |
| **4.1.3.8** **emailfield.dart** |  |
| **4.1.3.9 passwordfield.dart** | **43** |
| **4.1.3.10 bottom\_banner\_ad.dart** |  |
| **4.1.3.11 tags\_field.dart** | **44** |
| **4.2 Back-end** | **45** |
| **4.2.1 ภาษาที่ใช้ในการพัฒนา backend** |  |
| **4.2.2 Data** |  |
| **4.2.2.1 DataContext.cs** |  |
| **4.2.3 Controller** | **46** |
| **4.2.3.1 AdminController.cs** |  |
| **4.2.3.2 AdvertiserController.cs** |  |
| **4.2.3.3 LoginController.cs** |  |
| **4.2.3.4 PostController.cs** | **47** |
| **4.2.4 Model** | **48** |
| **4.2.4.1 Advertiser.cs** |  |
| **4.2.4.2 Post.cs** |  |
| **4.2.4.3 Posts\_Users.cs** |  |
| **4.2.4.4 Tags\_Follows.cs** | **49** |
| **4.2.4.5 Tags\_Posts.cs** |  |
| **4.2.4.6 User.cs** |  |
| **4.2.4.7 User\_Follows.cs** | **50** |
| **4.2.4.8 Users\_SharedPosts.cs** |  |
| **4.2.5 Program.cs** |  |
| **4.3 Github** | **51** |
| **4.3.1 Frontend** |  |
| **4.3.2 Backend** |  |
|  |  |
| เรื่อง | หน้า |
| **บทที่ 5 การแบ่งงาน** | 52 |
| **5.1 ฝ่ายในการทำงาน** |  |
| **5.1.1 ผู้ดูแลฝ่าย Backend** |  |
| **5.1.2 ผู้ดูแลฝ่าย Frontend** |  |
| **บทที่ 6 สรุปการทำงาน** | 53 |

**สารบัญรูปภาพ**

|  |  |
| --- | --- |
| **รูป** | **หน้า** |
| **รูปที่ 1 ออกแบบหน้า home** | **5** |
| **รูปที่ 2 ออกแบบหน้า login** | **6** |
| **รูปที่ 3 ออกแบบหน้า register** | **7** |
| **รูปที่ 4 ออกแบบหน้า Display feed** | **8** |
| **รูปที่ 5 ออกแบบหน้า Filter** | **9** |
| **รูปที่ 6 หน้าโปรไฟล์ผู้ใช้งานท่านอื่น** | **10** |
| **รูปที่ 7 ออกแบบหน้าโปรไฟล์ตัวเอง** | **11** |
| **รูปที่ 8 ออกแบบส่วนเสริม** | **12** |
| **รูปที่ 9 ออกแบบหน้า Display ของแอดมิน** | **13** |
| **รูปที่ 10 ออกแบบหน้า Filter ของแอดมิน** | **14** |
| **รูปที่ 11 ออกแบบหน้า Post approve ของแอดมิน** | **15** |
| **รูปที่ 12 ออกแบบหน้า report ของแอดมิน** | **16** |
| **รูปที่ 13 ออกแบบหน้า verified account ของแอดมิน** | **17** |
| **รูปที่ 14 Collect Domain Events** | **18** |
| **รูปที่ 15 Refine Domain Events** | **19** |
| **รูปที่ 16 Track cause** | **20** |
| **รูปที่ 17 component diagram** | **21** |
| **รูปที่ 18 UserContext** | **22** |
| **รูปที่ 19 ContentContext** | **22** |
| **รูปที่ 20 AdministrationContext** | **23** |
| **รูปที่ 21 import font awesome** | **28** |
| **รูปที่ 22 Façade ใน DataContext** | **29** |
| **รูปที่ 23 Iterator ใน database** | **30** |
| **รูปที่ 24 ส่วนหนึ่งของ code ใน userdisplay.cs** | **31** |
| **รูปที่ 25 ส่วนหนึ่งของ code ใน post\_page.dart** | **32** |
| **รูปที่ 26 ส่วนหนึ่งของ code ใน editprofilepage.dart** | **32** |
| **รูปที่ 27 ส่วนหนึ่งของ code หน้า othersprofilepage** | **33** |

|  |  |
| --- | --- |
| **รูป** | **หน้า** |
| **รูปที่ 28 ส่วนหนึ่งของ code หน้า userprofilepage** | 33 |
| **รูปที่ 29 ส่วนหนึ่งของ code หน้า home.dart** | 34 |
| **รูปที่ 30 ส่วนหนึ่งของ code หน้า login.dart** | 34 |
| **รูปที่ 31 ส่วนหนึ่งของ code หน้า register.dart** | 35 |
| **รูปที่ 32 ส่วนหนึ่งของ code หน้า postapprove.dart** | 36 |
| **รูปที่ 33 ส่วนหนึ่งของ code หน้า postreport.dart** | 37 |
| **รูปที่ 34 ส่วนหนึ่งของ code หน้า userreport.dart** | 38 |
| **รูปที่ 35 ส่วนหนึ่งของ code หน้า userverified.dart** | 38 |
| **รูปที่ 36 code ใน circle\_button.dart** | 39 |
| **รูปที่ 37 ส่วนหนึ่งของ code ใน create\_post\_contrainer.dart** | 39 |
| **รูปที่ 38 code ในการสร้างปุ่ม icbutton** | 40 |
| **รูปที่ 39 ส่วนหนึ่งของ code ใน profile\_container.dart** | 40 |
| **รูปที่ 40 ส่วนหนึ่งของ code ใน navigation\_drawer** | 41 |
| **รูปที่ 41 ส่วนหนึ่งของ code ใน post\_contrainer** | 41 |
| **รูปที่ 42 code ใน tag\_button** | 42 |
| **รูปที่ 43 ส่วนหนึ่งของ code ใน emailfield** | 42 |
| **รูปที่ 44 ส่วนของ code ใน passwordfield** | 43 |
| **รูปที่ 45 ส่วนหนึ่งของ code ใน widget bottom\_banner\_ad** | 43 |
| **รูปที่ 46 ส่วนหนึ่งของ code ใน widget tags\_field** | 44 |
| **รูปที่ 47 ตัวอย่าง code DataContext.cs** | 45 |
| **รูปที่ 48 code ดู post ที่โดน report** | 46 |
| **รูปที่ 49 code ดึง data ของ ads** | 46 |
| **รูปที่ 50 code ในการ register** | 47 |
| **รูปที่ 51 code ในการเพิ่ม tags ลงไปในโพส** | 47 |
| **รูปที่ 52 code Advertiser.cs** | 48 |
| **รูปที่ 53 code Post.cs** | 48 |
| **รูปที่ 54 code Posts\_Users.cs** | 48 |
| **รูปที่ 55 code Tags\_Follow.cs** | 49 |
| **รูปที่ 56 code Tags\_Posts.cs** | 49 |
| **รูปที่ 57 code User.cs** | 49 |

|  |  |
| --- | --- |
| **รูป** | **หน้า** |
| **รูปที่ 58 code User\_Follows.cs** | 50 |
| **รูปที่ 59 code Users\_SharedPosts.cs** | 50 |
| **รูปที่ 60 ส่วนหนึ่งของ code ใน Program.cs** | 50 |

**บทที่ 1**

**บทนำ**

**1.1 ที่มาของโครงการ**

โครงงาน Application KMITL News มีแนวคิดมาจากปัญหาในชีวิตประจำวันที่ทางผู้พัฒนาได้พบเจอ ในปัจจุบันหากมีความต้องการติดตามข่าวสารจากทางสถาบันได้อย่างครบถ้วน นักศึกษามีความจำเป็นที่ต้องติดตามเพจเฟซบุ๊กมากกว่า 10 เพจเพื่อตามข่าวได้ครบซึ่งในแต่ละเพจประกอบด้วยโพสหลากหลายเรื่องต่างกันออกไปและ content ในเพจนั้นประกอบทั้งที่นักศึกษาอยากทราบและไม่อยากทราบ อีกทั้งในบางครั้งข่าวสารที่มาจากอาจารย์ของแต่ละคณะนั้นไม่ได้มีการโพสลงไปในเพจแต่ถูกโพสลงไปในเว็บไซต์ของทางสถาบัน ถ้าหากทางนักศึกษาจำเป็นต้องเปิดเข้าไปดูด้วยตัวเองซึ่งสร้างคววามยุ่งยากและอาจตกหล่นข่าวสารที่จำเป็นได้

จากเหตุผลข้างต้นด้วยความต้องการที่จะเพิ่มความสะดวกสบายในการใช้งานและแก้ไขปัญหาให้นักศึกษาได้รับข่าวสารอย่างครบถ้วนโดยพัฒนา application KMITL News ที่จะรวบรวมข่าวสารจากทั้งเพจเฟซบุ๊กของสถาบัน อาจารย์ และในเว็บไซต์ของถามสถาบันหรือภาควิชานั้นๆ มารวมกันในที่เดียวและยังทำให้ผู้ใช้งานสามารถกรองข่าวที่ตัวเองสนใจได้และสามารถค้นหาและติดตามได้ผ่าน tags โดยข่าวสารนั้นถูกโพสได้ทั้งแอดมินและผู้ใช้งานแต่ผู้ใช้งานจะต้องผ่านการคัดกรองเพื่อให้ข่าวภายในแอพมีความถูกต้องมากที่สุด

**1.2 จุดประสงค์ของโครงการ**

พัฒนาแอพพลิเคชั่นที่สามารถแก้ปัญหาการที่นักศึกษาจำเป็นต้องติดตามหลายช่องทางเพื่อที่สามารถติดตามข่าวสารได้อย่างครบถ้วน โดยแอพพลิเคชั่นที่พัฒนาขึ้นจะทำให้นักศึกษาสามารถติดตามหัวข้อที่ตัวเองสนใจหรือติดตามผู้ใช้งานอาจารย์ที่ต้องการได้ อีกทั้งยังมีการจัดหมวดหมู่ติด tags เพื่อที่สามารถให้ผู้ใช้งานตามได้โดยง่าย และมีการคัดกรองข่าวสารภาพในแอพพลิเคชั่นให้มีความถูกต้องอยู่ตลอดเสมอ และยังสามารถให้ third party เข้ามาติดต่อเพื่อขอให้แสดงโฆษณาลงภายในแอพพลิเคชั่น

**บทที่ 2**

**ภาพรวมและการออกแบบระบบ**

**2.1 ภาพรวมขั้นตอนการทำงานของระบบ**

**2.1.1 ฝั่งผู้ใช้งาน**

**2.1.1.1 Login/Register**

เป็นหน้าแรกเมื่อผู้ใช้งานเปิดเข้าแอพพลิเคชั่นจะแบ่งออกเป็น 2 แบบคือผู้ใช้เก่าและผู้ใช้ใหม่ หากผู้ใช้งานเป็นผู้ใช้งานเก่าจะสามารถทำการ login แต่หากเป็นผู้ใช้งานใหม่จะต้องทำการ register เพื่อสร้างบัญชีผู้ใช้งานก่อนซึ่งการ register จำเป็นต้องใส่ข้อมูลประกอบด้วย ชื่อ-สกุล เบอร์โทรศัพท์ อีเมล และ password เพื่อใช้ในการ login ต่อไป

**2.1.1.2 Display feed**

เป็นหน้าหลักที่จะมีการแสดงโพสทุกอย่างทั้งที่ติดตามและ random ขึ้นมาซึ่งในหน้าหลักนี้ผู้ใช้งานสามารถโพสได้เช่นกันโดยจะมีกล่องให้สร้างโพสอยู่ทางด้านบนของหน้า feed และด้านบนใน AppBar จะมีปุ่มให้เปิดหน้าต่าง slide เพื่อดูเมนูในการเปลี่ยนหน้าไปยังหน้าที่มีให้เลือก เพิ่มความสะดวกในการใช้งานแก่ผู้ใช้งาน

**2.1.1.3 Filter**

หน้า filter จะเกิดจากปุ่มเปิด menu filter ขึ้นมาเพื่อให้สามารถเปลี่ยนหน้าไปยังหน้าต่างๆ โดยสามารถเลือกได้เป็น Home page, For you page, Tags ซึ่งแบ่งเป็น Tags search และ Following tags, Profile page, Request Verified page สำหรับเพื่อยืนยันตัวตนทำให้สามารถโพสข่าวได้โดยไม่ต้องรอการ verify สิ่งที่ต้องการโพสจากแอดมินและด้านล่างสุดจะมีปุ่มให้ logout ออกจากระบบ

**2.1.1.4 Tags display page**

เป็นหน้าที่จะเข้ามาดูโพสด้วย tag ที่ผู้ใช้งานค้นหาเมื่อ search tag ที่ต้องการหน้า Tags display จะแสดงทุกโพสที่เกี่ยวกับ tag นั้นๆ ออกมาทั้งหมดไม่ว่าจะเป็นผู้ใช้งานท่านใดโพสก็ตามและยังสามารถกดติดตาม tag นั้นๆ ได้อีกด้วย

**2.1.1.5 Profile page**

**2.1.1.5.1 Profile ตัวเอง**

เป็นหน้าโปรไฟล์ตัวเองที่แสดงรูปภาพตัวเองและชื่อของผู้ใช้งานตัวเองที่สามารถเข้าไปเพื่อแก้ไขโปรไฟล์ตัวเองอีกทั้งยังมีการ display สิ่งที่ผู้ใช้งานได้ทำการโพสไปทั้งหมด ซึ่งตรงโพสนั้นๆ สามารถที่จะกดลบโพสได้

**2.1.1.5.1.1 แก้ไขโปรไฟล์**

สามารถแก้ไขชื่อและรูปที่จะ display ออกไป

**2.1.1.5.2 Profile คนอื่น**

เป็นหน้าที่ผู้ใช้งานเข้าไปดูหน้า profile ผู้ใช้งานคนอื่นโดยจะแสดงรูปและชื่อของผู้ใช้งานคนนั้นๆ และมีปุ่มให้ติดตามผู้ใช้งานคนนั้นเพื่อที่จะได้ไม่พลาดสิ่งที่ผู้ใช้งานคนนั้นโพส ซึ่งจำเป็นมากๆ ผู้ใช้งานที่ต้องการติดตามเป็นอาจารย์ เช่น ต้องการติดตามทุกโพสที่อาจารย์ปริญญาเป็นคนโพส เป็นต้น อีกทั้งยังมีปุ่ม report เพื่อใช้ในกรณีที่ผู้ใช้งานนั้นๆ มีการ display ชื่อไม่เหมาะสมหรือผู้ใช้งานที่ได้รับการ verify โพสสิ่งที่ไม่เป็นความจริงมากจนเกินไปหรือเกิดการ spam อีกทั้งยังมีการแสดงโพสที่ผู้ใช้งานคนนั้นเคยโพสไปและยังสามารถ report โพสที่ผู้ใช้งานคนนั้นได้ทำการโพสไปได้

**2.1.2 ฝั่งแอดมิน**

**2.1.2.1 แอดมิน Display page**

หน้า display feed จะเหมือนกันกับฝั่ง user แต่มีการเปลี่ยนจากปุ่ม report เปลี่ยนเป็นปุ่มลบโพสเพื่อทำให้แอดมินสามารถลบโพสที่เห็นบนหน้า feed ได้ทันทีไม่ต้องรอผู้ใช้งานคนอื่นๆ มา report

**2.1.2.1.1 Admin Filter**

หน้า Admin Filter จะแตกต่างจากของผู้ใช้งานทั่วไปคือแทนที่ request verified เปลี่ยนเป็น admin panel ซึ่งประกอบด้วย post นั้นคือ การที่แอดมินจะทำการ verify โพสต่างๆ ที่ผู้ใช้งานทำการโพสลงหน้าฟีดคล้ายกับกลุ่มในเฟซบุ๊กreport คือการดูว่ามีการ report โพสอะไรบ้างแล้วทำการลบหรือปัดตกคำร้องนั้นไป verified คือการยืนยันตัวตอนผู้ใช้งานที่ทำการขอ verified account ของตัวเองมา และด้านล่างยังมีปุ่ม logout เพื่อออกจากระบบ

**2.1.2.2 แอดมิน Post approve**

เป็นหน้ารวมโพสที่ต้องการขอ approve เพื่อให้โพสนั้นถูกโพสลงไปใน feed ได้อย่างเสร็จสมบูรณ์ ซึ่งแอดมินจะเห็นโพสที่รอการ approve และมีปุ่มให้แอดมินกดว่าผ่านหรือไม่ผ่าน

**2.1.2.3 แอดมิน Report**

เป็นหน้าที่แอดมินเห็นโพสหรือโปรไฟล์ที่ถูก report ส่งมาจากผู้ใช้งานคนอื่นๆ และจะมีปุ่มให้กดว่าจะลบโพสนี้หรือไม่หากไม่ก็ถูกปล่อยผ่านไป

**2.1.2.4 แอดมิน Verified**

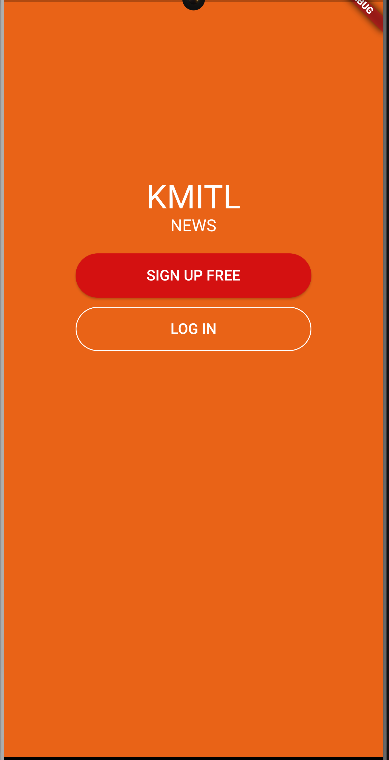
เป็นหน้าที่จะเห็นข้อมูลของผู้ใช้งานคนนั้นๆ และทำการกดปุ่มให้ว่าผ่านการ verified หรือไม่ โดยช้อมูลที่ส่งมาเพื่อ verified จะประกอบด้วยชื่อ-สกุล อีเมล และเบอร์โทร

**2.2 การออกแบบหน้า**

ออกแบบ UX/UI บน figma เพื่อนำเป็นแบบในการทำงานจริง

**2.2.1 ออกแบบหน้าผู้ใช้งาน**

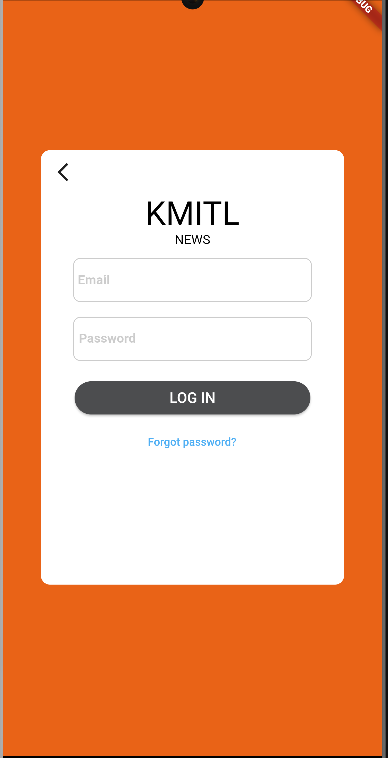
**2.2.1.1 ออกแบบหน้า Login/Register**



**รูปที่ 1 ออกแบบหน้า home**

**2.2.1.2 ออกแบบหน้า Login**

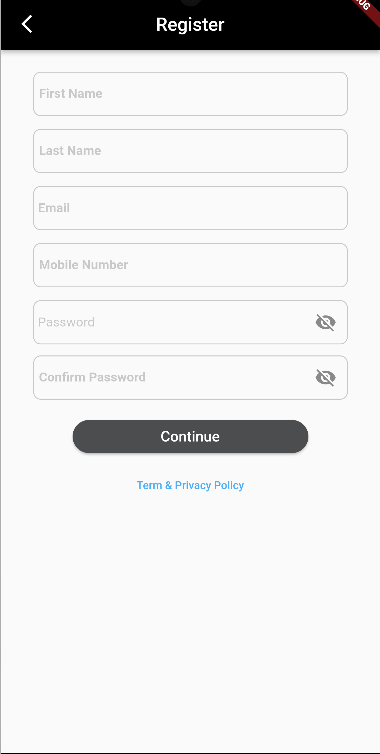
ให้ผู้ใช้งานกรอกอีเมลและ password ก่อน login



**รูปที่ 2 ออกแบบหน้า login**

**2.2.1.3 ออกแบบหน้า Register**

ให้ผู้ใช้งานกรอก ชื่อจริง นามสกุลจริง อีเมล เบอร์โทรศัพท์ password และ ยืนยัน password อีกครั้งเพื่อให้ชัวร์ว่าผู้ใช้งานกรอกรหัสถูกต้อง



**รูปที่ 3 ออกแบบหน้า register**

**2.2.1.4 ออกแบบหน้า Display feed**

เป็นหน้าที่แสดงโพสทั้งที่ผู้ใช้งานกดติดตามและ random ขึ้นมา



**รูปที่ 4 ออกแบบหน้า Display feed**

**2.2.1.5 ออกแบบหน้า Filter**

หน้าที่สามารถกดแล้วสามารถเปลี่ยนหน้าไปยังหน้าต่างๆ ภายในแอพพลิเคชั่น

****

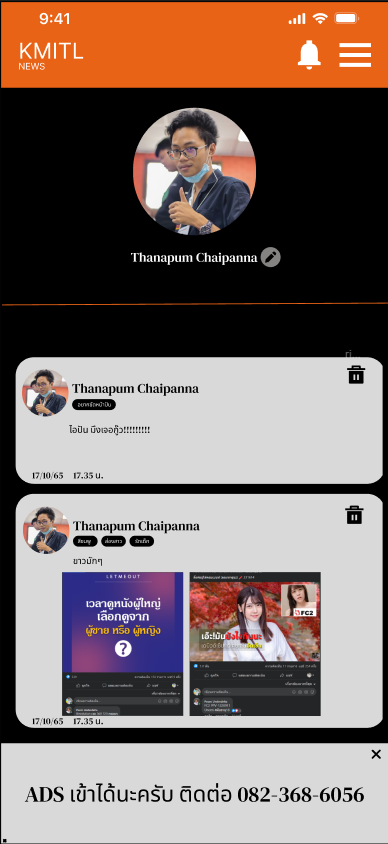
**รูปที่ 5 ออกแบบหน้า Filter**

**2.2.1.6 หน้า Profile**

แสดงหน้า profile ของผู้ใช้งานเองและผู้ใช้งานท่านอื่น



**รูปที่ 6 หน้าโปรไฟล์ผู้ใช้งานท่านอื่น**



**รูปที่ 7 ออกแบบหน้าโปรไฟล์ตัวเอง**

**2.2.1.7 ออกแบบส่วนเสริม**

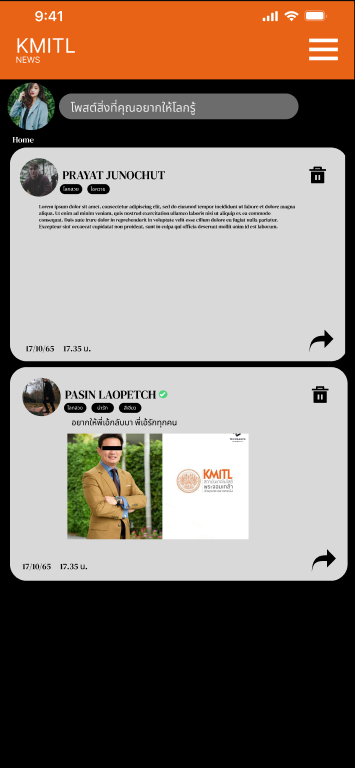
เป็นส่วนที่เกิดขึ้นเมื่อกดการโพส ใส่ tags



**รูปที่ 8 ออกแบบส่วนเสริม**

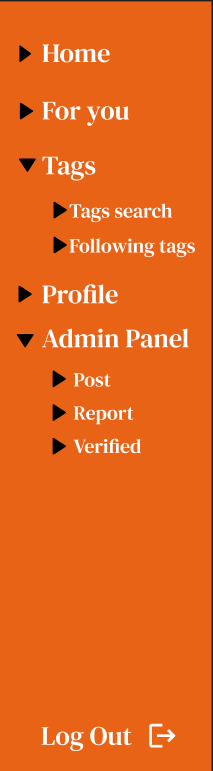
**2.2.2 ออกแบบหน้าแอดมิน**

**2.2.2.1 ออกแบบ Display feed admin**

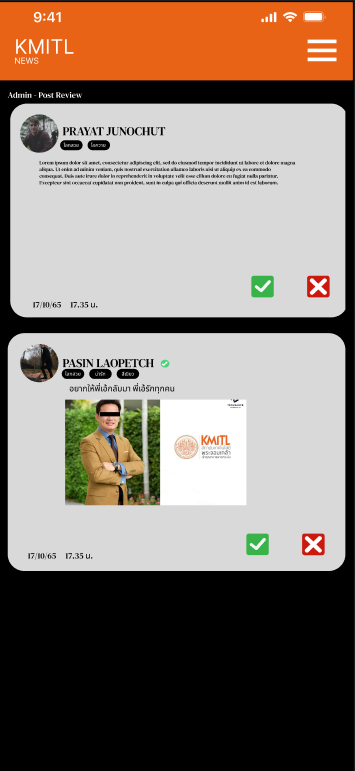
****

**รูปที่ 9 ออกแบบหน้า Display ของแอดมิน**

**2.2.2.2 ออกแบบหน้า Filter ของแอดมิน**

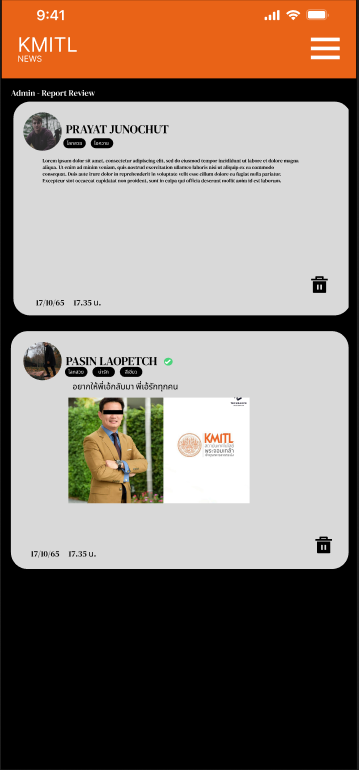
****

**รูปที่ 10 ออกแบบหน้า Filter ของแอดมิน**

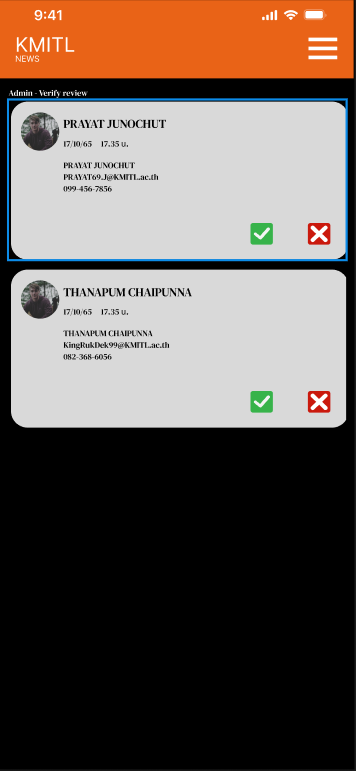
** 2.2.2.3 ออกแบบหน้า Post approve ของแอดมิน**

**รูปที่ 11 ออกแบบหน้า Post approve ของแอดมิน**

**2.2.2.4 ออกแบบหน้า report ของแอดมิน**

****

**รูปที่ 12 ออกแบบหน้า report ของแอดมิน**

** 2.2.2.5 ออกแบบหน้า verified account ของแอดมิน**

**รูปที่ 13 ออกแบบหน้า verified account ของแอดมิน**

**บทที่ 3**

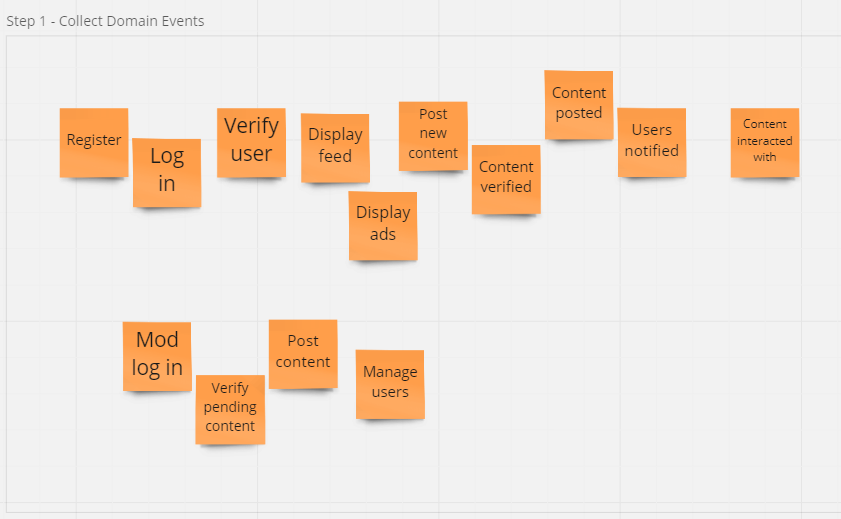
**การประยุกต์บทเรียน**

**3.1 Event Storming**

[Event Storming, Online Whiteboard for Visual Collaboration (miro.com)](https://miro.com/app/board/uXjVPQ-22Vo=/)

Event Storming แบ่งเป็น 3 steps คือ

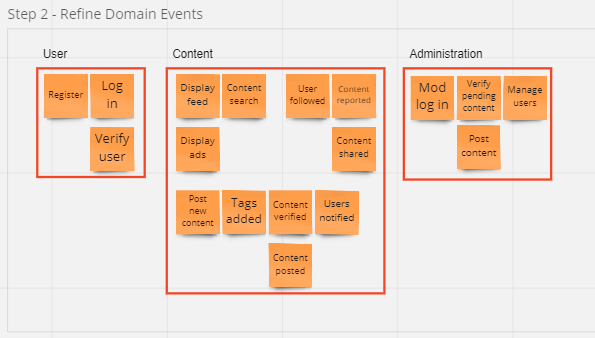
**3.1.1 Collect Domain Events**

 เป็นการนำ Event หลักต่างๆ มาวางเอาไว้บน board ด้วย post-it สีส้ม

**รูปที่ 14 Collect Domain Events**

ในรูปจะประกอบด้วย Domain หลักที่เป็นการทำงานหลักของแอพพลิเคชั่นโดยแบ่งเป็น 2 สายคือ user และ admin โดยฝั่ง user ประกอบไปด้วย Register, Login, Verify user, Display feed, Display ads, Post new content, content verified, Content posted, Users notified, Content interacted with และฝั่งadmin ประกอบด้วย Admin login, Verify pending content, Post content, manage users.

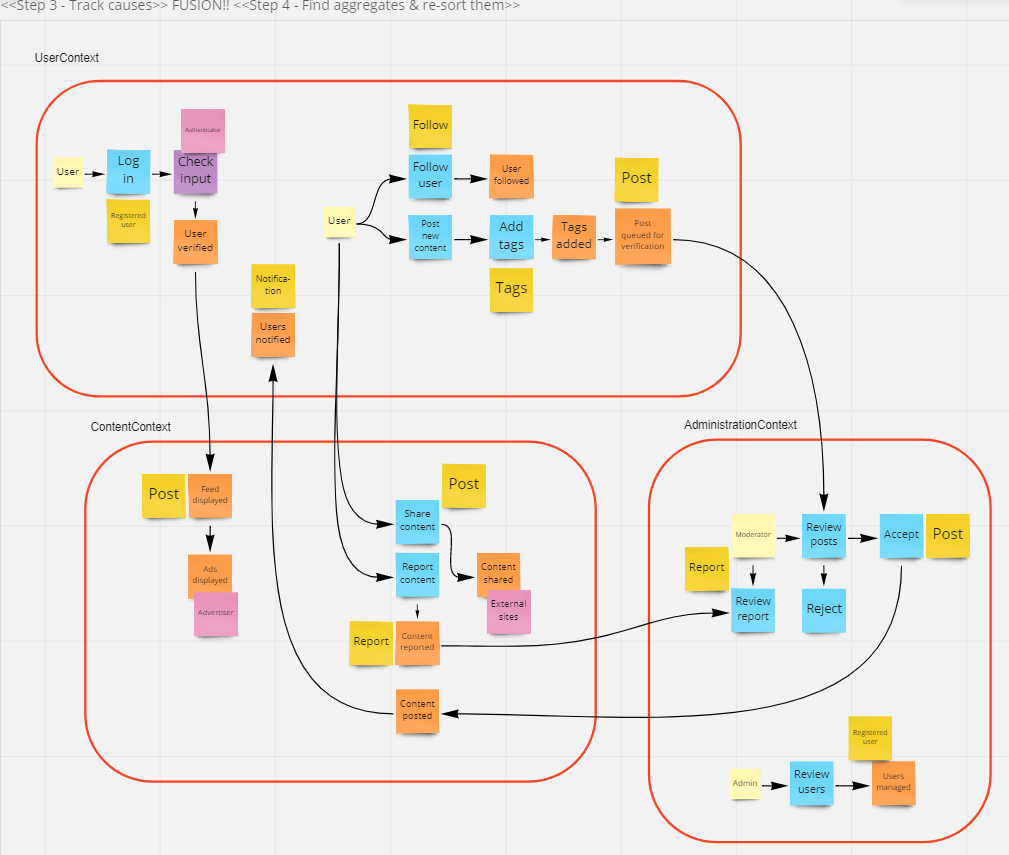
**3.1.2 Refine Domain Events**

เป็นการแบ่งกรอบ domain ที่มีความเกี่ยวข้องกันเข้าไปอยู่ด้วยกัน ในแอพพลิเคชั่น KMITL News แบ่งออกเป็น 3 กรอบนั้นคือ User, Content และ Administration

**รูปที่ 15 Refine Domain Events**

**3.1.3 Track cause รวมกับการใส่ aggregates และ จัดการโยงลำดับ**

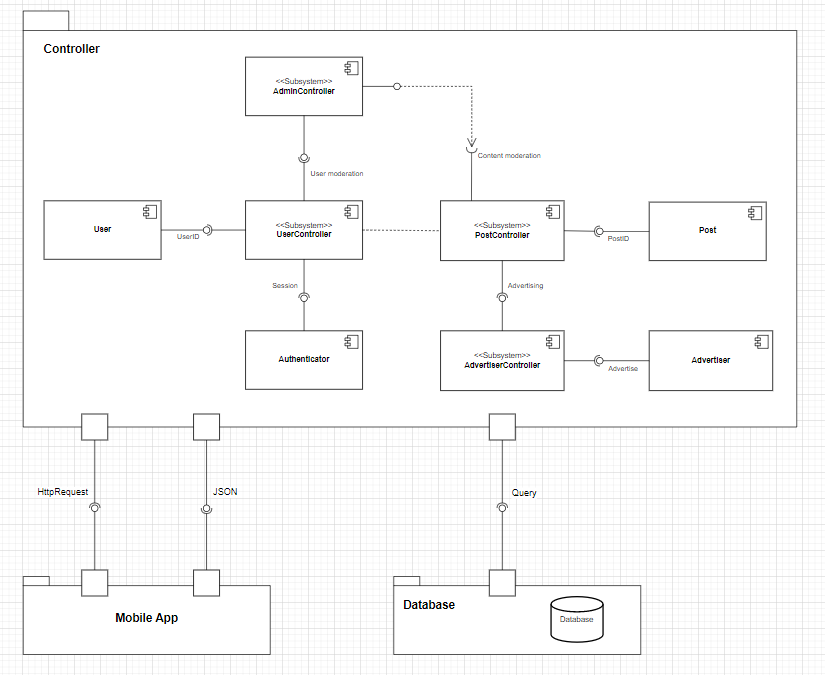
เป็นส่วนในการใส่การทำงานการโยงข้อมูลการทำงานเข้าด้วยกันเป็นลำดับเหตุการณ์ โดยแบ่งสีเป็น สีเหลืองอ่อนเป็น user หรือ actor สีครามคือสิ่งที่ user ต้องการทำ (command) สีเหลืองเข้มคือ aggregate สีม่วงคือ logic และข้อมกำหนดต่างๆ สีชมพูคือ service จากภายนอก และมีลูกศรชี้โยงการทำงานได้ดัง รูปที่() และทำการแบ่งออกเป็น bounded context ซึ่งได้เป็น 3 bounded คือ UserContext, ContentContext และ AdministrationContext

**รูปที่ 16 Track cause**

**3.2 UML**

[SAD\_Diagram.drawio - diagrams.net](https://app.diagrams.net/#G1_QBtbKABu9dVxuOQ1W4NnBFdNKxFF0aj)

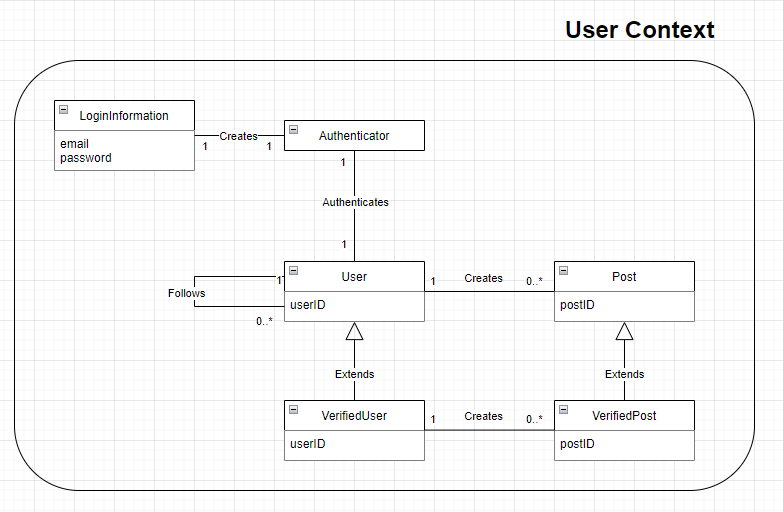
**3.2.1 UML Component**

****

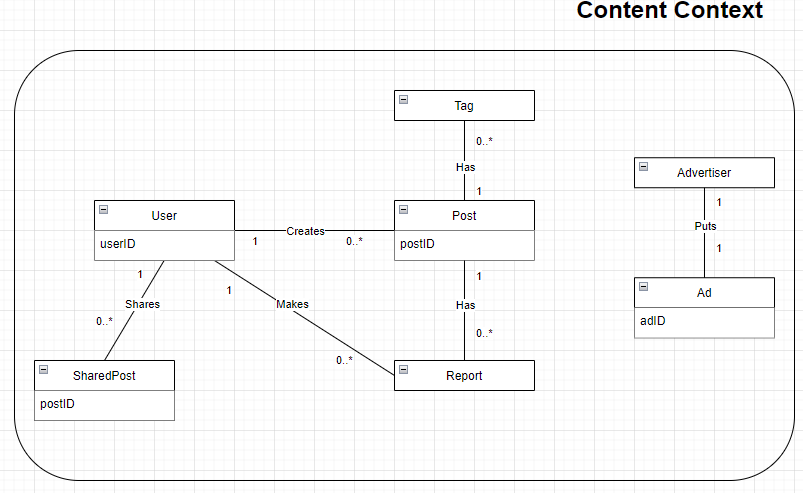
**รูปที่ 17 component diagram**

**3.2.2 domain model**

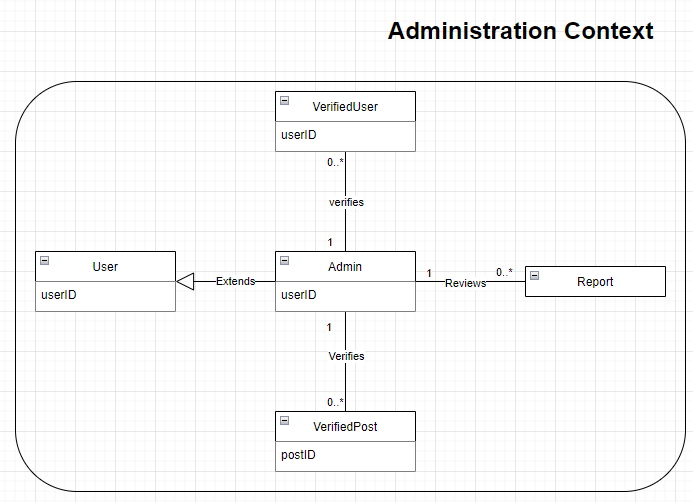
แบ่งออกเป็น 3 bounded context

**3.2.2.1 UserContext**

**รูปที่ 18 UserContext**

**3.2.2.2 ContentContext**

**รูปที่ 19 ContentContext**

** 3.2.2.3 AdministrationContext**

**รูปที่ 20 AdministrationContext**

**3.3 Quality Attribute**

**3.3.1 Availability**

โปรแกรมสามารถใช้งานได้ตามจุดประสงค์

**3.3.1.1 สามารถ Display feed**

หน้าแอพพลิเคชั่นสามารถแสดงผล feed ข่าวตามที่ผู้พัฒนาต้องการตามจุดประสงค์

|  |  |
| --- | --- |
| Source of stimulus | Users, Developers |
| Stimulus | Crash, incorrect response |
| Artifacts | Application |
| Environment | Normal Operation, Startup |
| Response | Display error window |
| Response measure | 99.99% Availability |

**3.3.1.2 ความสามารถในการเชื่อมต่อ Database**

ในการเชื่อมต่อกับ database สามารถเชื่อมต่อได้ตลอดเวลาเนื่องจาก database ทีเลือกใช้เป็นแบบ Local ซึ่งทำให้สามารถเปิดและปิดตามความต้องการได้อีกด้วย

|  |  |
| --- | --- |
| Source of stimulus | Users, Developers |
| Stimulus | Connecting to Backend |
| Artifacts | Application |
| Environment | Normal Operation, Startup |
| Response | Able to connect and communicate with Backend |
| Response measure | 24-hour connection |

**3.3.1.3 แสดง Tags**

เมื่อทำการค้นหา tags และสามารถแสดงโพสทุกโพสที่มี tags นั้นอยู่ออกมา

|  |  |
| --- | --- |
| **Source of stimulus** | Users, Developers |
| **Stimulus** | Tags search |
| **Artifacts** | Application |
| **Environment** | Normal Operation |
| **Response** | Display only posts containing the searched tag |
| **Response measure** | Response time < 1 second |

**3.3.1.4 จัดการ content**

ทำให้แอดมินสามารถคัดกรองเนื้อหาภายในแอพพลิเคชั่นให้มีความถูกต้อง

|  |  |
| --- | --- |
| **Source of stimulus** | Users, Developers |
| **Stimulus** | Content Management |
| **Artifacts** | Application |
| **Environment** | Normal Operation |
| **Response** | Admin is able to review and remove content |
| **Response measure** | Response time < 1 second |

**3.3.2 Integrability**

**3.3.2.1 Pub.dev**

สามารถเพิ่มเครื่องมือใช้งานจากภายนอกเข้ามาใช้ร่วมกับ code ที่มีอยู่ได้ เนื่องจากการเลือกใช้ภาษา dart ทำให้มีความ flexible ในการเพิ่ม plugin เข้ามา

|  |  |
| --- | --- |
| **Source of stimulus** | Developers |
| **Stimulus** | Add new plugin |
| **Artifacts** | Entire code |
| **Environment** | Development |
| **Response** | Plugin is successfully added and ready to use |
| **Response measure** | New plugin |

**3.3.3 Modifiability**

**3.3.3.1 การแก้ไข code ในส่วน Backend**

สามารถแก้ไขได้ง่ายเนื่องจากทางผู้พัฒนาได้เลือกใช้ Architectural pattern แบบ MVC

|  |  |
| --- | --- |
| **Source of stimulus** | Developers |
| **Stimulus** | Add/ Delete/ Modify functionality |
| **Artifacts** | Code |
| **Environment** | Build complete time |
| **Response** | Build complete |
| **Response measure** | Build time |

**3.3.3.2 การแก้ไข code ในส่วน frontend**

สามารถแก้ไขได้ง่ายเนื่องจาก code ส่วนมากมีการแยกออกมาเป็น Widget ทำให้สามารถหาและแก้ไขเพื่อส่งผลกับทุกจุด ทำให้ไม่เสียเวลาในการแก้ไข

|  |  |
| --- | --- |
| **Source of stimulus** | Developers |
| **Stimulus** | Add/ Delete/ Modify functionality and widget |
| **Artifacts** | Code |
| **Environment** | Build complete time |
| **Response** | Build complete, Make/Deploy modification |
| **Response measure** | Widget is able to be used in other classes |

**3.3.4 Performance**

**3.3.4.1 การส่ง request ข้อมูล**

ในการเชื่อมต่อและส่ง request ไปยัง database เพื่อขอข้อมูลจำนวนมาก ประกอบด้วยโพส โปรไฟล์ โฆษณา เป็นต้นโดยใช้เวลาในการ response ไม่มาก

|  |  |
| --- | --- |
| **Source of stimulus** | Users, Developer |
| **Stimulus** | User request |
| **Artifacts** | Application |
| **Environment** | Normal Operation |
| **Response** | Request response |
| **Response measure** | Latency < 1 seconds |

**3.4 Software Architectural Style**

**3.4.1 MVC**

ใช้ในส่วนของ backend ใช้จัดการกับส่วนของ code component เพื่อให้กลายเป็นโครงสร้างให้เรียกใช้งานได้งาน สะอาด ไม่มีการเรียกที่ทับซ้อนกันไม่ให้มีการเรียกเป็นทอดๆ เช่น ไม่จำเป็นต้องเรียก A แล้วไปหา B แต่สามารถเรียก B ได้โดยตรง

**3.4.2 Plug-in**

ใช้ในส่วนของ Frontend ร่วมกับ flutter ใช้ในการตกแต่งเพิ่มความสวยงามของแอพพลิเคชั่น ซึ่งทางผู้พัฒนาเลือกใช้ Font Awesome ซึ่งเป็นเว็บที่รวม icon ต่างๆ ในเลือกและเป็นเว็บที่ creator dev designer ส่วนมากเลือกใช้

**รูปที่ 21 import font awesome**

**3.4.3 Representational State Transfer (REST)**

ในส่วนของ frontend มีการใช้ Restful API เพื่อติดต่อกับ Backend โดยการยิง HTTP Methods เช่น GET.POST GET**.**DELETE เป็นต้น

**3.5 Design pattern**

**3.5.1 Façade**

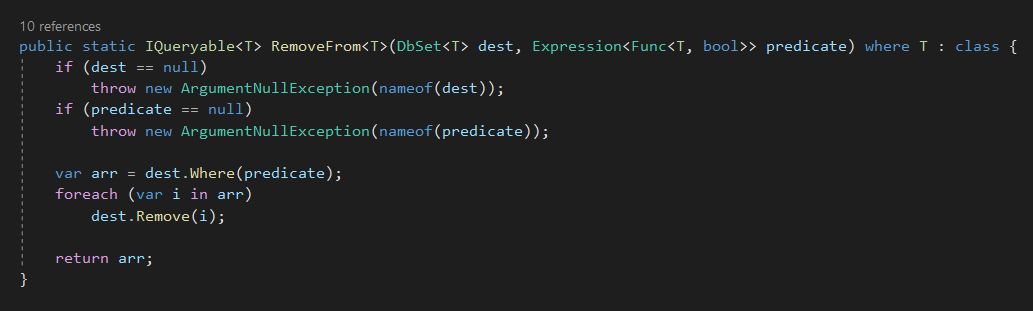
ใช้ในส่วนของ DataContext ซึ่งเอาไว้เชื่อมกับทาง database



**รูปที่ 22 Façade ใน DataContext**

**3.5.2 Iterator**

ใช้เพื่อจัดการกับ listdata เพื่อให้ดูข้อมูลเป็นลำดับจาก database



**รูปที่ 23 Iterator ใน database**

**บทที่ 4**

**การพัฒนา**

**4.1 Frontend**

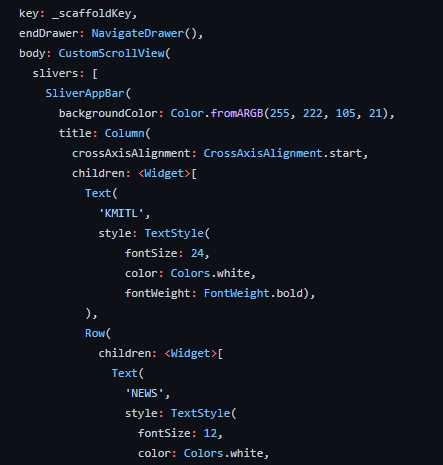
**4.1.1 ภาษาที่ใช้พัฒนา Frontend**

ในการพัฒนาแอพพลิเคชั่น KMITL News ผู้พัฒนาเลือกใช้ Flutter ในการพัฒนา mobile application นี้ ซึ่ง flutter เป็นเครื่องมือในการพัฒนา mobile application ที่สามารถทำงานข้าม platform ได้ในทั้ง IOS และ Android ใน flutter นั้นจะเป็นภาษา **Dart** ซึ่งถูกพัฒนาโดย google ออกมาเป็น open source ให้ใช้ฟรี

ภาษา dart ที่ใช้ใน Flutter ซึ่งจะมีความคล้ายกับภาษา Java เนื่องจาก dart เป็นภาษาที่รองรับ OOP และมีแนวคิดของ class และ inheritance เช่นเดียวกับภาษา Java

4**.1.2 Main Page**

**4.1.2.1 userdisplay.dart**

 เป็นหน้าที่แสดง feed หลักของผู้ใช้งาน โดยจะแสดงโพสที่มีใน database

**รูปที่ 24 ส่วนหนึ่งของ code ใน userdisplay.cs**

**4.1.2.2 post\_page.dart**

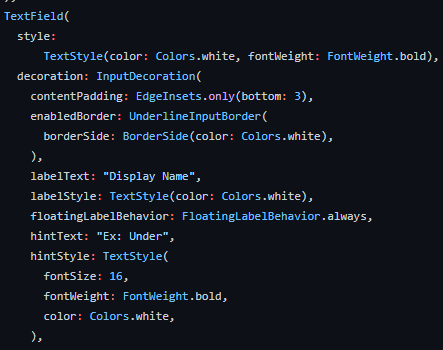
เป็นส่วนหนึ่งของหน้าหลักผู้ใข้งานแต่ทำหน้าที่ตอนที่ผู้ใช้งานทำการสร้างโพส



**รูปที่ 25 ส่วนหนึ่งของ code ใน post\_page.dart**

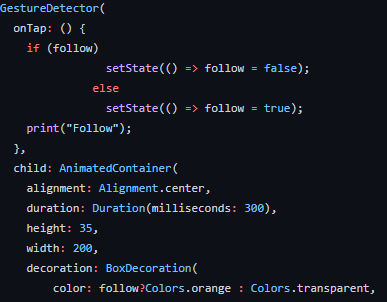
**4.1.2.3 editprofilepage.dart**

เป็นหน้าที่ทำหน้าที่ให้ผู้ใช้งานแก้ไขโปรไฟล์ตัวเองไม่ว่าจะเป็นแก้ไขรูปหรือชื่อที่ display ให้ผู้ใช้งานคนอื่นเห็น



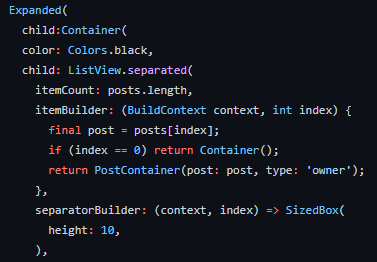
**รูปที่ 26 ส่วนหนึ่งของ code ใน editprofilepage.dart**

**4.1.2.4 othersprofilepage.dart**

เป็นหน้าโปรไฟล์ของผุ้ใช้งานคนอื่นเมื่อเข้าไปในในหน้าโปรไฟล์คนอื่นสามารถทำการกด follow ผู้ใช้งานอื่นได้ และสามารถเห็นโพสที่ผู้ใช้งานคนนั้นได้โพสไปอีกทั้งสามารถ report โปรไฟล์และโพสได้

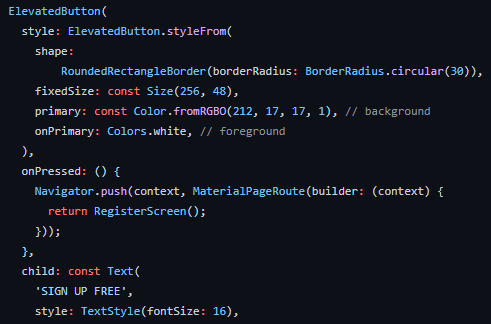
**รูปที่ 27 ส่วนหนึ่งของ code หน้า othersprofilepage**

**4.1.2.5 userprofilepage.dart**

เป็นหน้าโปรไฟล์ของผู้ใช้งานเอง ในหน้านี้ผู้ใช้งานสามารถกดแก้ไขโปรไฟล์ตัวเองได้โดยจะเข้าไปที่หน้า editprofilepage และสามารถเห็นโพสที่ตัวเองได้โพสไปอีกทั้งสามารถลบโพสที่โพสไปได้

**รูปที่ 28 ส่วนหนึ่งของ code หน้า userprofilepage**

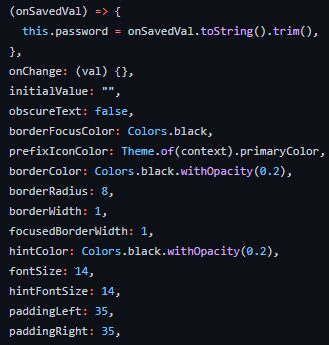
**4.1.2.6 home.dart**

เป็นหน้าแรกที่ผู้ใช้งานเข้าแอพพลิเคชั่น โดยให้ผู้ใช้งานเลือกระหว่าง Login หรือ Register

**รูปที่ 29 ส่วนหนึ่งของ code หน้า home.dart**

**4.1.2.7 login.dart**

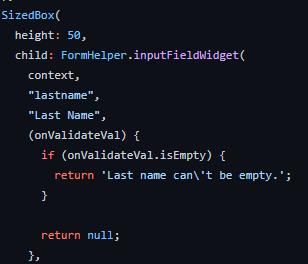
สืบเนื่องจากหน้า home หากผู้ใช้งานเป็นผู้ใช้งานเก่าหรือผู้ใช้งานที่มีบัญชีอยู่แล้วหน้า login เป็นหน้าที่ให้ผู้ใช้งานกรอกอีเมลและ password ให้ถูกต้องแล้วจึงสามารถเข้าใช้งานแอพพลิเคชั่นได้



**รูปที่ 30 ส่วนหนึ่งของ code หน้า login.dart**

**4.1.2.8 register.dart**

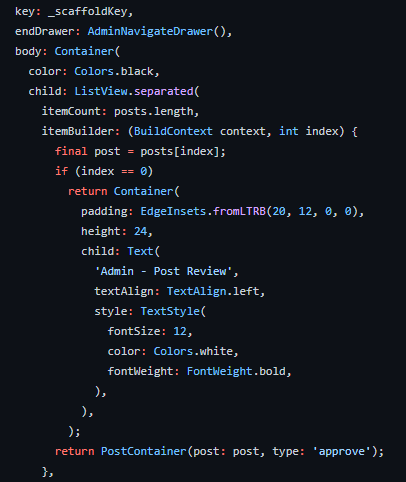
สืบเนื่องจากหน้า home หากผู้ใช้งานเป็นผู้ใช้งานใหม่และไม่มีบัญชี ผู้ใช้งานจำเป็นต้องสร้างบัญชีโดยการกด Register ซึ่งในหน้า register นี้ผู้ใช้งานจำเป็นต้องกรอกชื่อจริงนามสกุลจริงอีเมล password และ confirm password ให้ตรงกันจึงถือว่า register เสร็จสมบูรณ์



**รูปที่ 31 ส่วนหนึ่งของ code หน้า register.dart**

**4.1.2.9 postapprove.dart**

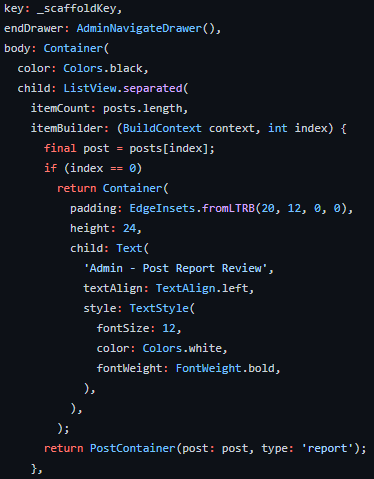
ในหน้านี้เป็นส่วนประกอบของฝั่งแอดมินหากผู้ใช้งานที่ทำการ login เข้ามาไม่ได้เป็นบัญชีแอดมิน ผู้ใช้งานจะไม่สามารถเห็นและเข้ามาในหน้านี้ได้ ในหน้า postapprove เป็นหน้าที่จะแสดงโพสที่รอการยืนยันให้สามารถโพสได้จากแอดมิน



**รูปที่ 32 ส่วนหนึ่งของ code หน้า postapprove.dart**

**4.1.2.10 postreport.dart**

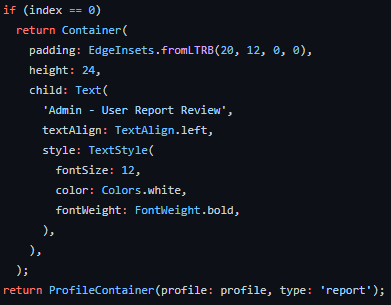
ในหน้านี้เป็นส่วนประกอบของฝั่งแอดมินหากผู้ใช้งานที่ทำการ login เข้ามาไม่ได้เป็นบัญชีแอดมิน ผู้ใช้งานจะไม่สามารถเห็นและเข้ามาในหน้านี้ได้ ในหน้า postreport จะรวมโพสที่ถูก report เอาไว้ให้แอดมินได้ทำการตรวจสอบอีกครั้งว่าสมควรถูกลบออกหรือไม่



**รูปที่ 33 ส่วนหนึ่งของ code หน้า postreport.dart**

**4.1.2.11 userreport.dart**

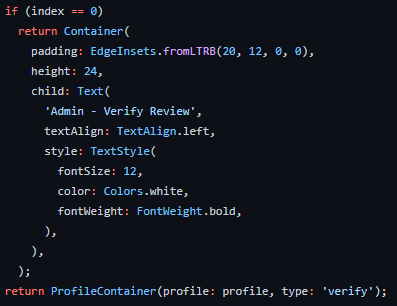
ในหน้านี้เป็นส่วนประกอบของฝั่งแอดมินหากผู้ใช้งานที่ทำการ login เข้ามาไม่ได้เป็นบัญชีแอดมิน ผู้ใช้งานจะไม่สามารถเห็นและเข้ามาในหน้านี้ได้ในหน้า userreport จะแสดงผู้ใช้งานที่โดน report ให้แอดมินได้ทำการตรวจสอบหากเป็นผู้ใช้งานที่ประพฤติตนไม่เหมาะสมหรือเป็นผู้ใช้งานปลอมแอดมินสามารถทำการลบบัญชีผู้ใช้งานได้



**รูปที่ 34 ส่วนหนึ่งของ code หน้า userreport.dart**

**4.1.2.12 userverified.dart**

ในหน้านี้เป็นส่วนประกอบของฝั่งแอดมินหากผู้ใช้งานที่ทำการ login เข้ามาไม่ได้เป็นบัญชีแอดมิน ผู้ใช้งานจะไม่สามารถเห็นและเข้ามาในหน้านี้ได้ในหน้า userverified เป็นหน้าที่แอดมินจะทำการตรวจสอบว่าผู้ใช้งานที่ขอการ verified มาเป็นผู้ใช้งานคนนั้นจริงๆ สามารถพิสูจน์ได้



**รูปที่ 35 ส่วนหนึ่งของ code หน้า userverified.dart**

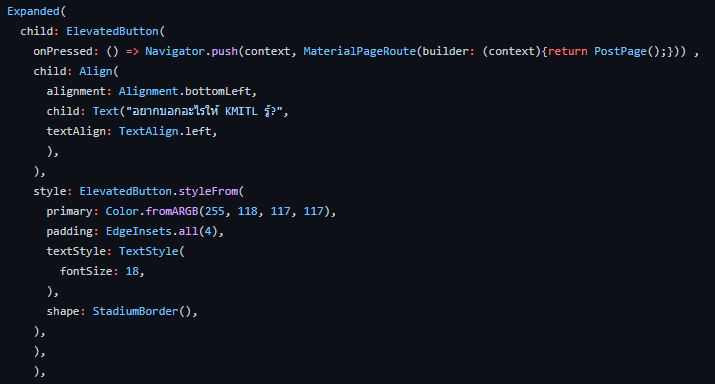
**4.1.3 Widget**

**4.1.3.1 circle\_button.dart**

ใช้ช่วยในการสร้างปุ่มเป็นวงกลม

**รูปที่ 36 code ใน circle\_button.dart**

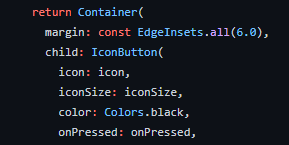
**4.1.3.2 create\_post\_contrainer.dart**

**** ใช้ในการสร้างกล่องสำหรับโพส

**รูปที่ 37 ส่วนหนึ่งของ code ใน create\_post\_contrainer.dart**

**4.1.3.3 icbutton.dart**

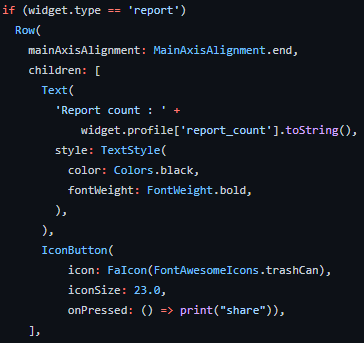
ใช้สร้างปุ่มเพื่อเอาไปใช้ในหน้าอื่นๆ

****

**รูปที่ 38 code ในการสร้างปุ่ม icbutton**

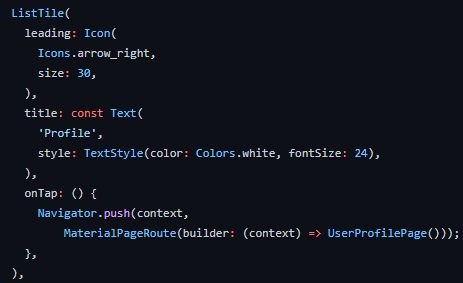
**4.1.3.4** **profile\_container.dart**

สร้างพื้นที่ในการใส่ข้อความเอาไปใช้ในส่วนของการ verified ผู้ใช้งานและดูการถูก report ซึ่งอยู่ในส่วนของแอดมิน



**รูปที่ 39 ส่วนหนึ่งของ code ใน profile\_container.dart**

**4.1.3.5 navigation\_drawer.dart**

Widget เสริมในการสร้าง slide menu ในการข้ามไปยังหน้าต่างๆ ประกอบด้วย Home page, For you page, Tags, Profile page, Request Verified, Logout เพื่อเอาไปใช้ในทุกๆ หน้าในแอพพลิเคชั่น ไม่ว่าจะเป็นหน้า home หน้า profile ตัวเองหรือคนอื่น

**รูปที่ 40 ส่วนหนึ่งของ code ใน navigation\_drawer**

**4.1.3.6 post\_contrainer.dart**

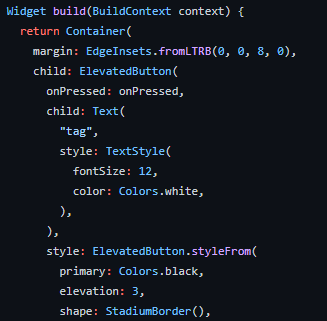
ใช้ส่วนที่เอาไว้สร้างโพสเพื่อเอาไป display ให้หน้า display feed

****

**รูปที่ 41 ส่วนหนึ่งของ code ใน post\_contrainer**

**4.1.3.7 tag\_button.dart**

ใช้ในการใส่เพื่อให้ tags ในโพสสามารถกดได้



**รูปที่ 42 code ใน tag\_button**

**4.1.3.8** **emailfield.dart**

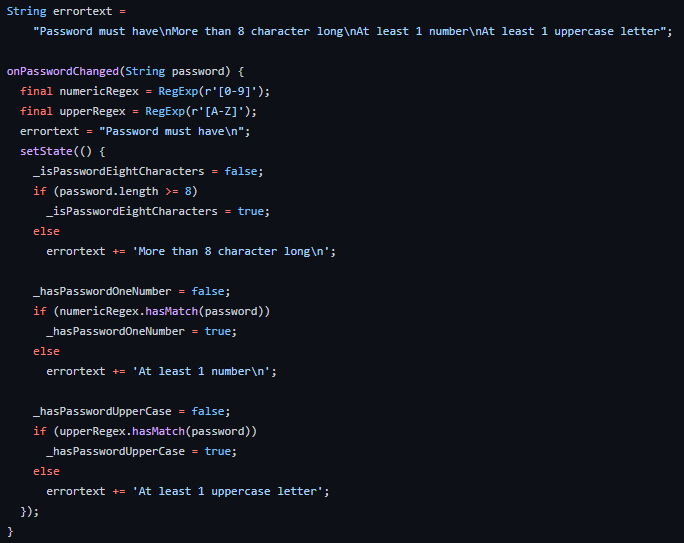
Widget ในการใส่อีเมลและเช็ตว่าที่ใส่ลงไปนั้นถูกหรือไม่

****

**รูปที่ 43 ส่วนหนึ่งของ code ใน emailfield**

**4.1.3.9 passwordfield.dart**

Widget การใส่ password และ ตรวจสอบว่า password ผ่านหรือไม่โดย password ต้องประกอบ 1 ตัวเลข ตัวอักษรใหญ่ 1 ตัว และความยามอย่างน้อย 8 ตัวอักษร



**รูปที่ 44 ส่วนของ code ใน passwordfield**

**4.1.3.10 bottom\_banner\_ad.dart**

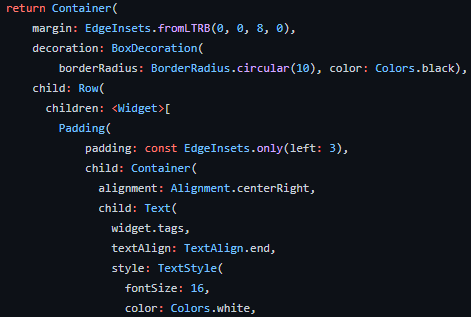
เป็น Widget ที่ใช้ในการแสดงผลโฆษณาที่ด้านล่างของแอพพลิเคชั่นที่สามารถกดเข้าเพื่อไปยังหน้าเว็บของโฆษณานั้นๆ ผ่านลิ้งก์ที่ผูกเอาไว้



**รูปที่ 45 ส่วนหนึ่งของ code ใน widget bottom\_banner\_ad**

**4.1.3.11 tags\_field.dart**

มีเป็น widget ที่ใช้ในส่วนของการสร้างโพส ทำหน้าที่คล้ายกับ tags\_button เพียงแต่วิธีใช้และจุดประสงค์เวลาที่ใช้ต่างกัน

****

**รูปที่ 46 ส่วนหนึ่งของ code ใน widget tags\_field**

**4.2 Back-end**

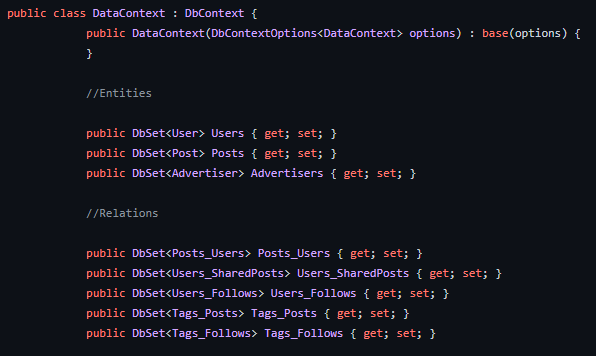
**4.2.1 ภาษาที่ใช้ในการพัฒนา backend**

ในการพัฒนา database ของแอพพลิเคชั่น KMITL News ทางผู้พัฒนาได้เลือกใช้ภาษา C# เป็นภาษาหลักในการพัฒนา ซึ่งเลือกใช้ C# เนื่องจากภาษา C# มีความเอกประสงค์ในการใช้งาน ง่ายสำหรับผู้เริ่มต้น มีความคล้ายกับภาษาอื่นซึ่งทำให้ง่ายต่อการทำความเข้าใจ ง่ายต่อการ maintain และทำงานได้อย่างรวมเร็ว

**4.2.2 Data**

**4.2.2.1 DataContext.cs**

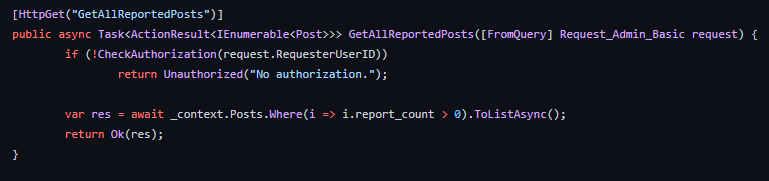
ใช้เชื่อมกับ database เป็นการตั้งชื่อ schema และเป็นการบอก database ให้ทราบถึงโครงสร้าง



**รูปที่ 47 ตัวอย่าง code DataContext.cs**

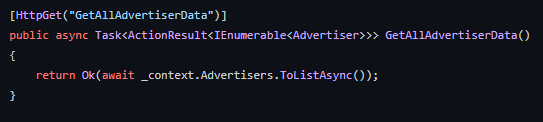
**4.2.3 Controller**

**4.2.3.1 AdminController.cs**

จัดการ action ที่เกี่ยวกับ admin ทั้งหมด โดยควบคุมการ verify post, verify user ดู post ที่ยังไม่ถูก verify ดู user ที่โดน report และการลบ post

**รูปที่ 48 code ดู post ที่โดน report**

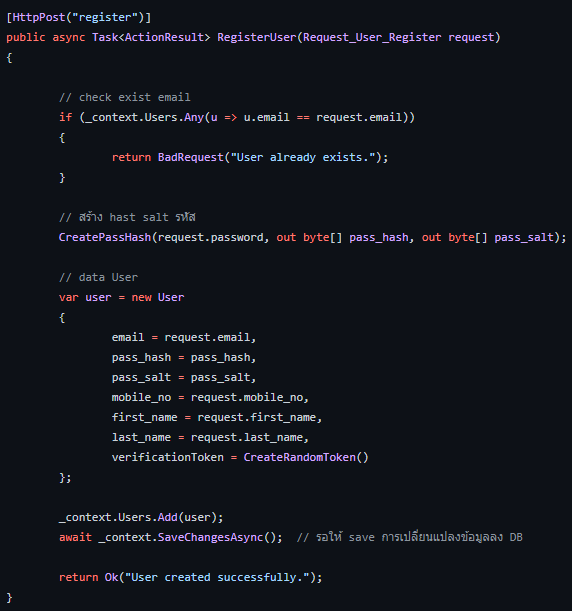
**4.2.3.2 AdvertiserController.cs**

 ใช้ในการดึงโฆษณาเพื่อมาแสดงใน frontend

**รูปที่ 49 code ดึง data ของ ads**

**4.2.3.3 LoginController.cs**

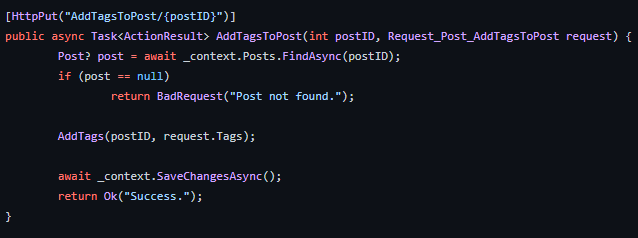
ใช้ในการจัดการในหน้า login และการแก้ไขโปรไฟล์ สามารถทำได้ครอบคลุมทั้งการหา id ว่ามีอยู่แล้วในระบบหรือไม่ในการ login การ check password ว่าตรงกับอีเมลที่ใส่ในการ login หรือไม่ ครอบคลุมการ register เพื่อสร้าง account ขึ้นมา การ check อีเมลว่ามีการใช้ซ้ำกันหรือไม่ การ reset password การ reset จำนวนในการ report และนับจำนวนที่โดน report ในฝั่งของ admin



**รูปที่ 50 code ในการ register**

**4.2.3.4 PostController.cs**

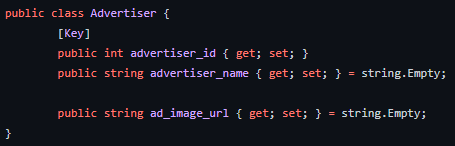
ใช้เกี่ยวกับโพสทั้งหมดไม่ว่าจะเป็นการดึงโพสทั้งหมดใน database และสามารถดูได้ว่า post นั้นๆ โดน report ไปกี่ครั้งและ reset จำนวนที่โดน report อีกทั้งเป็นตัวควบคุมในการสร้างโพสพร้อมกับการใส่ tags และยังมี function ในการ share โพสนั้นๆ อีกทั้งยังสามารถดูได้ว่ามีโพสไหนบ้างที่ถูก verified ไปแล้วบ้าง สามารถดูได้ว่าผู้ใช้งานคนอื่นๆ ได้ share อะไรไปแล้วบ้างดูโพสทั้งหมดของผู้ใช้งานนั้นๆ สามารถแสดงโพสที่มี tags ตามที่ต้องการได้ สามารถดู followers



**รูปที่ 51 code ในการเพิ่ม tags ลงไปในโพส**

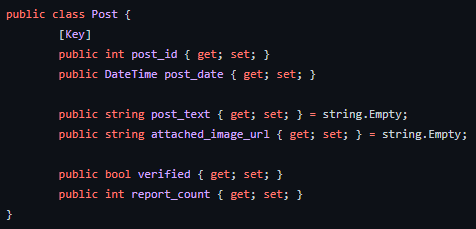
**4.2.4 Model**

**4.2.4.1 Advertiser.cs**

เป็นส่วนประกอบของโครงสร้างข้อมูลของ advertiser

**รูปที่ 52 code Advertiser.cs**

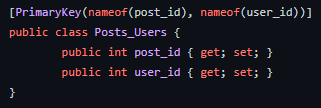
**4.2.4.2 Post.cs**

**** เป็นส่วนประกอบของโครงสร้างข้อมูลของโพส

**รูปที่ 53 code Post.cs**

**4.2.4.3 Posts\_Users.cs**

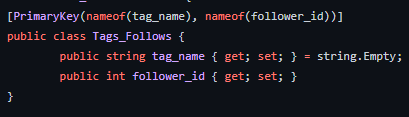
เป็นส่วนประกอบของโครงสร้างข้อมูลของโพสที่ผู้ใช้งานได้ทำการโพส

****

**รูปที่ 54 code Posts\_Users.cs**

**4.2.4.4 Tags\_Follows.cs**

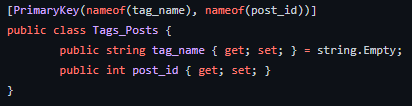
เป็นส่วนประกอบของโครงสร้างข้อมูลของ tags ที่สามารถบอกได้ว่ามีผู้ใช้งานใดติดตาม tags บ้าง

****

**รูปที่ 55 code Tags\_Follow.c**

**4.2.4.5 Tags\_Posts.cs**

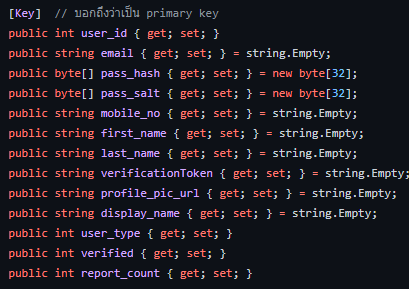
เป็นส่วนประกอบของโครงสร้างข้อมูลของ tags ที่สามารถบอกได้ว่ามีโพสใดมีการใส่ tags บ้าง

****

**รูปที่ 56 code Tags\_Posts.cs**

**4.2.4.6 User.cs**

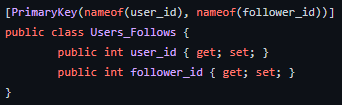
เป็นส่วนประกอบของโครงสร้างข้อมูลของผู้ใช้งาน

****

**รูปที่ 57 code User.cs**

**4.2.4.7 User\_Follows.cs**

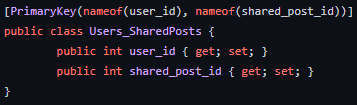
เป็นส่วนประกอบของโครงสร้างข้อมูลของผู้ใช้งานที่สามารถบอกได้ว่ามีผู้ใช้งานใดติดตามผู้ใช้งานคนใดบ้าง



**รูปที่ 58 code User\_Follows.cs**

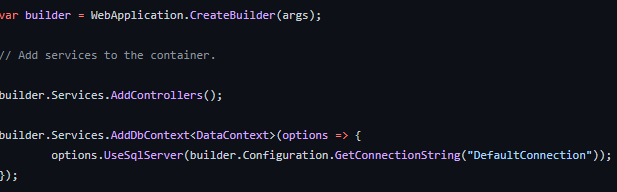
**4.2.4.8 Users\_SharedPosts.cs**

เป็นส่วนประกอบของโครงสร้างข้อมูลของผู้ใช้งานที่สามารถบอกได้ว่ามีผู้ใช้งานใดทำการ share โพสอะไรไปบ้าง

****

**รูปที่ 59 code Users\_SharedPosts.cs**

**4.2.5 Program.cs**

เป็นส่วนหลักเพื่อใช้ในการ run โปรแกรม มี services ต่างๆ เมื่อทำการเริ่มเรียกใช้แล้วจะทำการดึง services อื่นๆ ต่อกันไปเรื่อยๆ

**รูปที่ 60 ส่วนหนึ่งของ code ใน Program.cs**

4.3 Github

**4.3.1 Frontend**

[underpoom/softarchfinalprojectflutter (github.com)](https://github.com/underpoom/softarchfinalprojectflutter)

**4.3.2 Backend**

[Natashi/KMITL\_SoftArch\_KMITLNews\_Backend: Backend Web API for my Software Architectures class group project (github.com)](https://github.com/Natashi/KMITL_SoftArch_KMITLNews_Backend)

**บทที่ 5**

**การแบ่งงาน**

**5.1 ฝ่ายในการทำงาน**

ในการทำงานแบ่งออกเป็น 2 ฝ่ายคือ Backend และ Frontend

**5.1.1 ผู้ดูแลฝ่าย Backend**

ประกอบด้วยทั้งหมด 4 ท่าน ได้แก่

1. นาย ถิรวัฒน์ จิตรานุเคราะห์

2. นาย ธนภูมิ ชัยพรรณา

3. นาย นันทวัฒน์ รัตนเรืองวิมาณ

4. นาย ปัญญวัฒน์ ธนทัตธาดา

**5.1.2 ผู้ดูแลฝ่าย Frontend**

ประกอบด้วยทั้งหมด 3 ท่าน ได้แก่

1. นาย นพสิทธิ์ จันทรวีระกุล

2. นาย พนธกร สุจิตชวานนท์

3. นาย พศิน แก้วนิล

**บทที่ 6**

**สรุปการทำงาน**

ในการทำแอพพลิเคชั่น KMITL News ทางทางทีมผู้พัฒนาได้ทำการแบ่งงานให้แต่ละคนในทีมทำ แต่ถึงจะมีการแบ่งงานอย่างชัดเจน ทางทีมเน้นการทำงานแบบช่วยเหลือซึ่งกันและกัน ในการทำงานหากมีคนใดในทีมติดปัญหาในการทำงาน คนในทีมพร้อมที่จะเข้าไปช่วยร่วมแก้ปัญหาไม่ว่าจะเป็นในฝ่าย frontend หรือ backend

ในการทำงาน coding อาจพบเจอปัญหาในการทำงานอยู่บ้างซึ่งเกิดจากปัญหาเรื่องความชำนาญของภาษา เนื่องจากบางคนในทีมเริ่มใช้ภาษา dart และ C# เป็นภาษาแรก ทำให้ต้องใช้เวลาในการเรียนรู้ เมื่อเริ่มดำเนินงานความเชี่ยวชาญเพิ่มมากขึ้นทำให้งานสามารถดำเนินได้รวมเร็วมากยิ่งขึ้นส่งผลให้สามารถพัฒนาแอพพลิเคชั่นออกมาได้ตามที่ตั้งเป้าหมายเอาไว้และสามารถพัฒนาจนเสร็จภายในเวลาที่กำหนดพร้อมกับประยุกต์ร่วมกับบทเรียน

การประยุกต์ร่วมกับบทเรียนเข้ามาใช้ทั้ง MVC, Client server & N-tier เข้ามาใช้ในการจัดการใน backend เพื่อให้สามารถแก้ไขได้ง่ายและเชื่อมต่อกับ backend อย่างมีประสิทธิภาพ ซึ่งส่งเสริมได้ quality attribute ทั้ง Availability Performance และ Modifiability อีกทั้งยังมีการนำ Plugin จากภายนอกเข้ามาเพื่อให้หน้า frontend มีความสามารถมากยิ่งขึ้น ซึ่งส่งเสริมเรื่อง integrability เพื่อให้แอพพลิเคชั่นที่พัฒนาขึ้นมาสามารถใช้งานได้อย่างมีประสิทธิภาพและง่ายต่อผู้ใช้งาน