

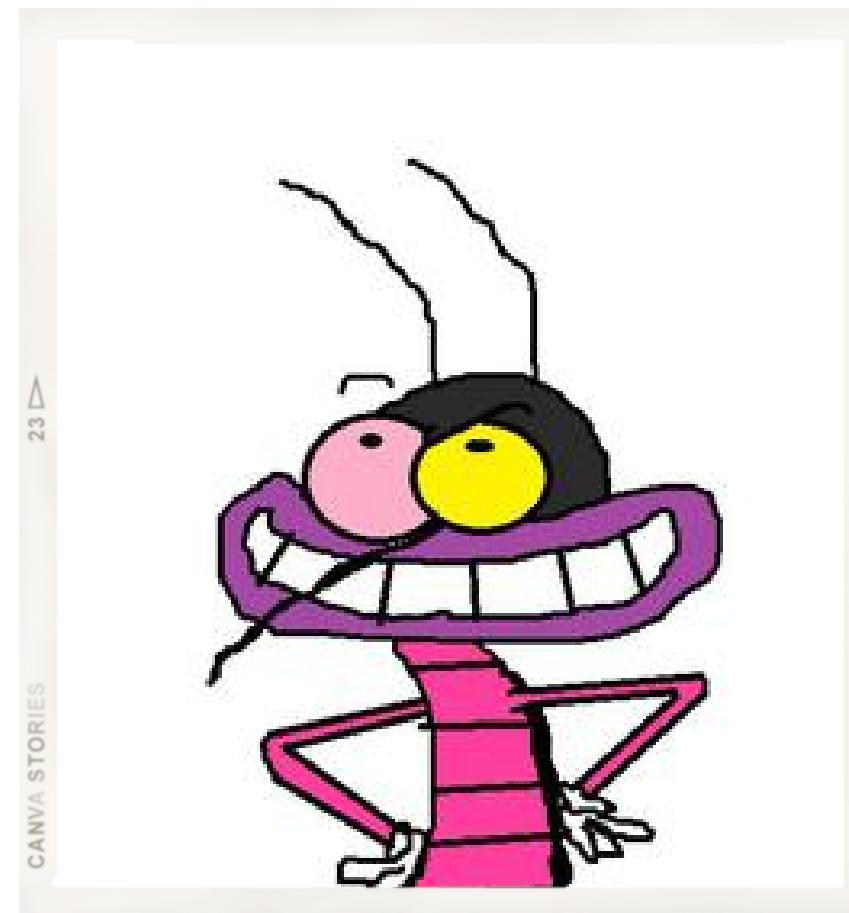


KKU BLOG

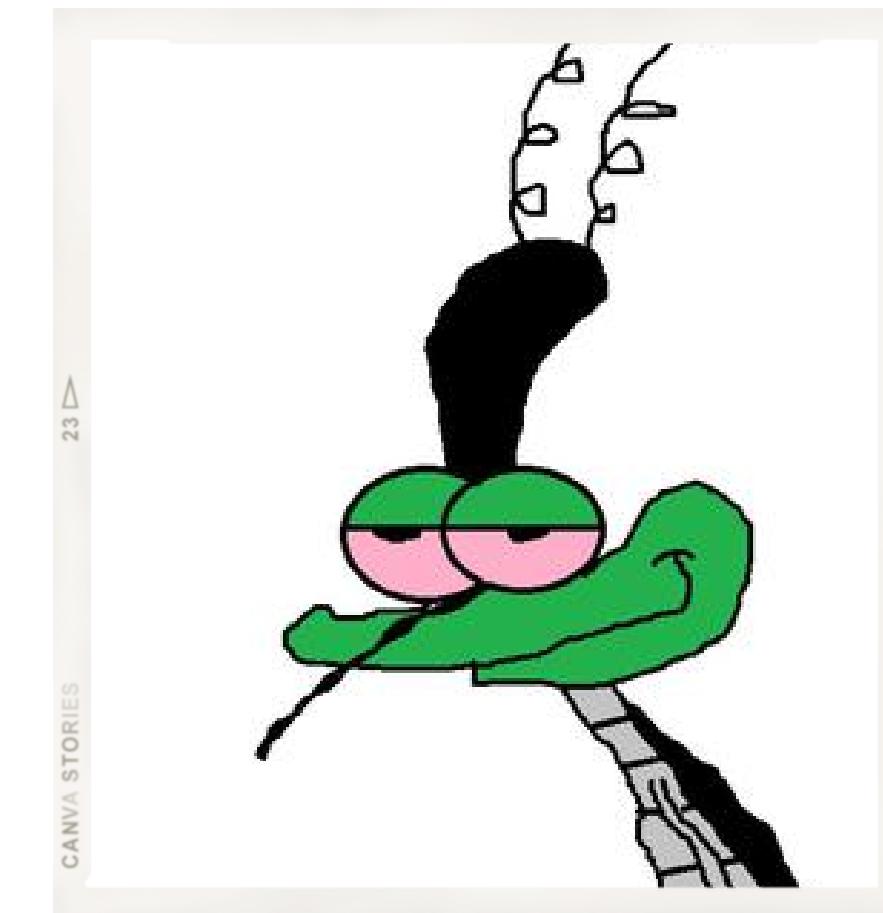
เว็บไซต์แลกเปลี่ยนความคิดเห็น KKU



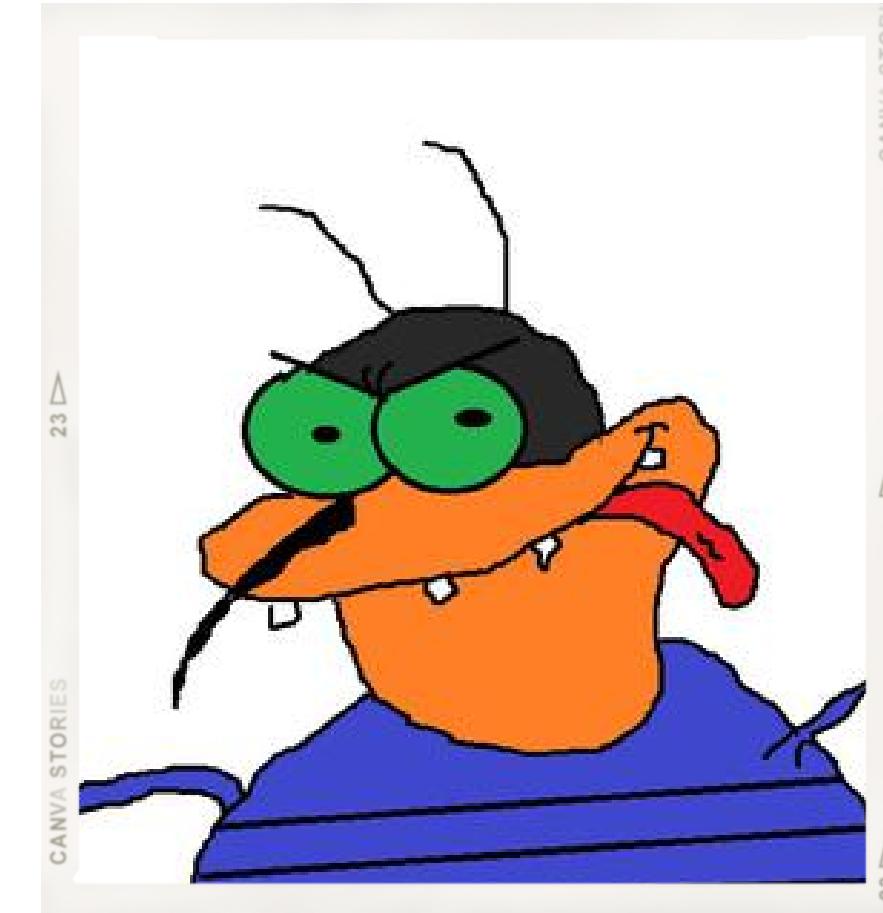
សមាជិក



643020045-6
นายលោនវ៉ាណ៍ ហមាយបុណ្យ



643021233-0
นายវិរភ័ទ សែងចិត្ត



643021239-8
นายពន្លេវុនី វរមេះ



ที่มาและความสำคัญ

ในปัจจุบันมหาวิทยาลัยขอนแก่นมีบุคลากรจำนวนมากทั้งนักศึกษา อาจารย์ พนักงาน และบุคคลภายนอก ทำให้การพูดคุยเกี่ยวกับหัวข้อต่างๆ มีมากตามไปด้วย ทั้งการตั้งคำถาม ซื้อขายของใช้ ประชาสัมพันธ์ และการพูดคุยส่วนใหญ่นี้จะอยู่ในรูปแบบของแอปพลิเคชันโซเชียลมีเดีย ซึ่งมีมากมายต่างกันออกไป

จึงมีแนวคิดในการทำเว็บไซต์ที่รวบรวมหัวข้อต่างๆที่เกี่ยวข้องกับมหาวิทยาลัย ขอนแก่น ที่สามารถพูดคุยตั้งกระทู้หัวข้อต่างๆเพื่อแลกเปลี่ยนความคิดเห็นมาไว้ที่เดียว เพื่อให้เกิดความสะดวกในการพูดคุยมากขึ้น



วัตถุประสงค์

1. เพื่อใช้เป็นสื่อในการศึกษาให้กับผู้ที่สนใจ เรื่อง Spring Boot ในการเขียน Web Blog
2. เพื่อพัฒนาการเขียน Spring Boot ใน Pattern ต่างๆ
3. เพื่อสร้าง Web Blog ในการใช้แลกเปลี่ยนความคิดเห็นของนักศึกษามหาลัยขอนแก่น



ประโยชน์ที่คาดว่าจะได้รับ

1. เพิ่มความสะดวกในการ สือสารพูดคุยเกี่ยวกับหัวข้อต่าง ๆ และแบ่งปันข้อมูลที่เกี่ยวกับมหาวิทยาลัยของนักศึกษา
2. นักศึกษาและบุคลากรสามารถมาพูดคุย และแลกเปลี่ยนความคิดเห็นที่มีประโยชน์ เพื่อพัฒนามหาวิทยาลัยและส่งเสริมความร่วมมือในการแก้ปัญหา
3. การประชาสัมพันธ์กิจกรรมต่างๆ ของมหาวิทยาลัย มีความทั่วถึงมากขึ้น



ทฤษฎีที่เกี่ยวข้อง

Spring Boot Spring Boot คือ Framework ใน Spring อันหนึ่ง สามารถช่วยทำให้สร้าง Web application หรือ Web service ได้ง่ายขึ้น เพราะ Spring Boot มี Auto Configuration ซึ่งช่วยลดความยุ่งยากในการกำหนดค่าต่างๆ และสามารถใช้งานได้ทันที เนื่องจาก Spring Boot มี Java Web Server ที่ built-in มาให้แล้ว



ทฤษฎีที่เกี่ยวข้อง

Strategy Pattern เป็นการสร้างการทำงานหลาย ๆ แบบ แยกออกเป็น class และมี interface ร่วมกัน ยกตัวอย่างเช่นการจัดเรียงข้อมูล ไม่ว่าจะเป็น merge sort, bubble sort, quick sort เราสามารถนำมาประยุกต์ใช้งานได้โดยการสร้างเพียง method เดียวและทำการกำหนด signature ให้เหมือนกัน

```
Config
  CustomSuccessHanler.java
  SpringScecurityConfig.java
Controller
  AdminController.java
  DashboardControll.java 2
  GuestController.java
  LoginController.java
  RegisterController.java
> DTO
> Model
Repository
  CategoryRepository.java
  CommentReplyRepository..
  CommentRepository.java
  ImageRepository.java
  PostRepository.java
  ProfileRepository.java
  RoleRepository.java
  UserRepository.java
Service
  AdminService.java 1
  CommentService.java
  ImageService.java
  PostService.java
  userDetailsService.java
  UserlistService.java
  UserService.java
  UserServiceID.java
  UserServiceImpl.java
  DemoApplication.java
```



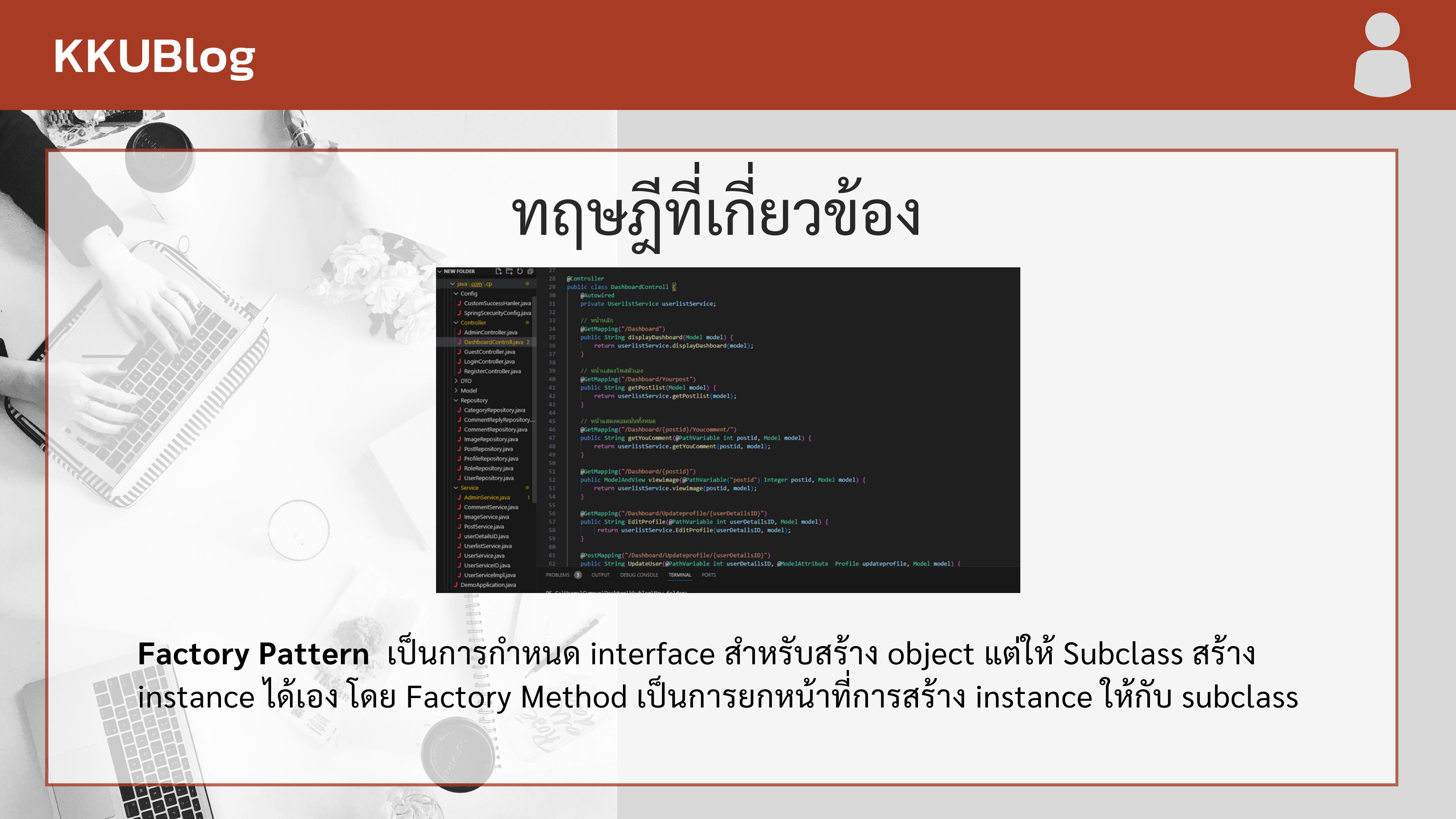
ทฤษฎีที่เกี่ยวข้อง

- ⌄ Repository
 - J CategoryRepository.java
 - J CommentReplyRepository..
 - J CommentRepository.java
 - J ImageRepository.java
 - J PostRepository.java
 - J ProfileRepository.java
 - J RoleRepository.java
 - J UserRepository.java

Repository Pattern เป็นการวางแผนสร้างของโค้ดรูปแบบหนึ่งเพื่อใช้แยก Logic สำหรับการเข้าถึงแหล่งข้อมูล (Data Source) ออกจาก Business layer โดยที่มี Repository Interface ทำหน้าที่เป็นตัวกลางในการติดต่อระหว่าง Business layer กับ Data source โดยจะดึงข้อมูลจาก Data source และแปลงข้อมูลให้อยู่ในรูปที่ทางผู้ดูแล Business layer สามารถนำไปใช้งานได้ทันที



ทฤษฎีเกี่ยวข้อง



A screenshot of a Java code editor showing a file named `DashboardController.java`. The code implements several methods for handling dashboard-related requests, such as displaying the dashboard, getting post lists, and viewing comments. It uses annotations like `@Controller`, `@GetMapping`, and `@Service`. The code editor interface includes a sidebar with project structure, a code editor with syntax highlighting, and a terminal tab at the bottom.

```
27 @Controller
28 public class DashboardController {
29     @Autowired
30     private UserlistService userlistService;
31
32     // หน้าหลัก
33     @GetMapping("/Dashboard")
34     public String displayDashboard(Model model) {
35         return userlistService.displayDashboard(model);
36     }
37
38     // หน้าแสดงโพสต์ของฉัน
39     @GetMapping("/Dashboard/Yourpost")
40     public String getPostList(Model model) {
41         return userlistService.getPostList(model);
42     }
43
44     // หน้าแสดงคอมเม้นท์ของฉัน
45     @GetMapping("/Dashboard/{postid}/Youcomment/")
46     public String getYouComment(@PathVariable int postid, Model model) {
47         return userlistService.getYouComment(postid, model);
48     }
49
50     @GetMapping("/Dashboard/{postid}")
51     public ModelAndView viewImage(@PathVariable("postid") Integer postid, Model model) {
52         return userlistService.viewImage(postid, model);
53     }
54
55     @GetMapping("/Dashboard/Updateprofile/{userDetailsID}")
56     public String EditProfile(@PathVariable int userDetailsID, Model model) {
57         return userlistService.EditProfile(userDetailsID, model);
58     }
59
60     @PostMapping("/Dashboard/Updateprofile/{userDetailsID}")
61     public String UpdateUser(@PathVariable int userDetailsID, @ModelAttribute Profile updateprofile, Model model) {
62 }
```

Factory Pattern เป็นการกำหนด interface สำหรับสร้าง object แต่ให้ Subclass สร้าง instance ได้เอง โดย Factory Method เป็นการกำหนดว่าที่การสร้าง instance ให้กับ subclass



ทฤษฎีที่เกี่ยวข้อง

Spring Dependency Injection เป็นแนวคิดหนึ่งใน Spring Framework ที่ช่วยให้เราสามารถเรียกใช้ function ที่ไม่มีในคลาสที่อยู่ แต่สามารถใช้ function จากคลาสอื่นเข้ามาใช้งานได้ โดยการใช้ `@Autowired` ซึ่งช่วยในการลดความผูกพันของคลาสต่าง ๆ และทำให้ระบบหรือคลาสมีความยืดหยุ่นมากขึ้น



เครื่องมือที่เกี่ยวข้อง

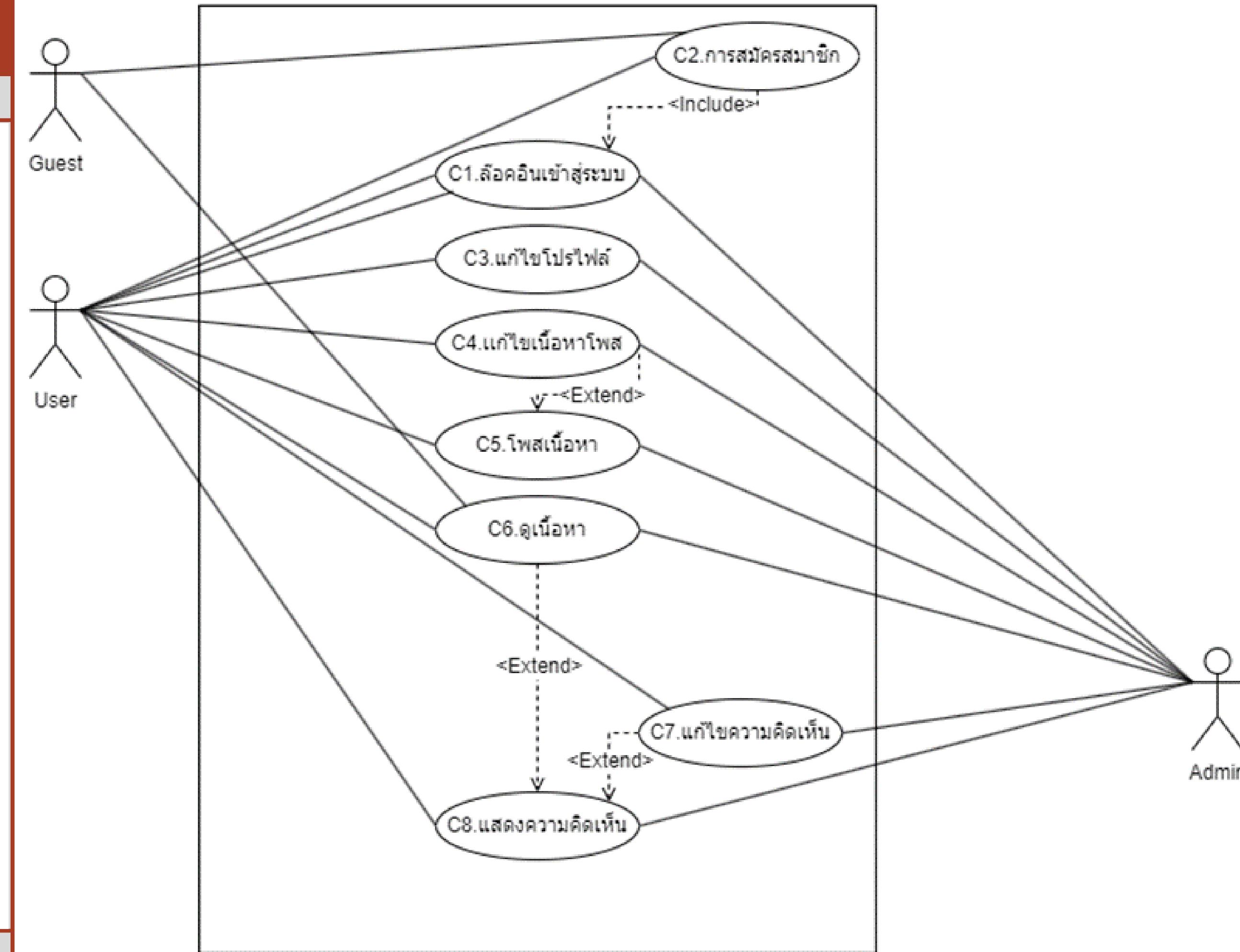
Visual studio code เป็นโปรแกรมแก้ไขโค้ดที่มีขนาดเล็กแต่มีความสามารถมาก สามารถใช้งานได้กับหลายภาษา ใช้ในการพัฒนาหน้าเว็บไซต์ html และเชื่อมฐานข้อมูลเพื่อทำงานและแก้ไข java

Mysql Workbench เป็นเครื่องมือการจัดการฐานข้อมูล MySQL ช่วยในการออกแบบฐานข้อมูล, จัดการฐานข้อมูล, และเชื่อมต่อกับฐานข้อมูล MySQL ใช้สร้างฐานข้อมูลเพื่อใช้งานกับหน้าเว็บไซต์

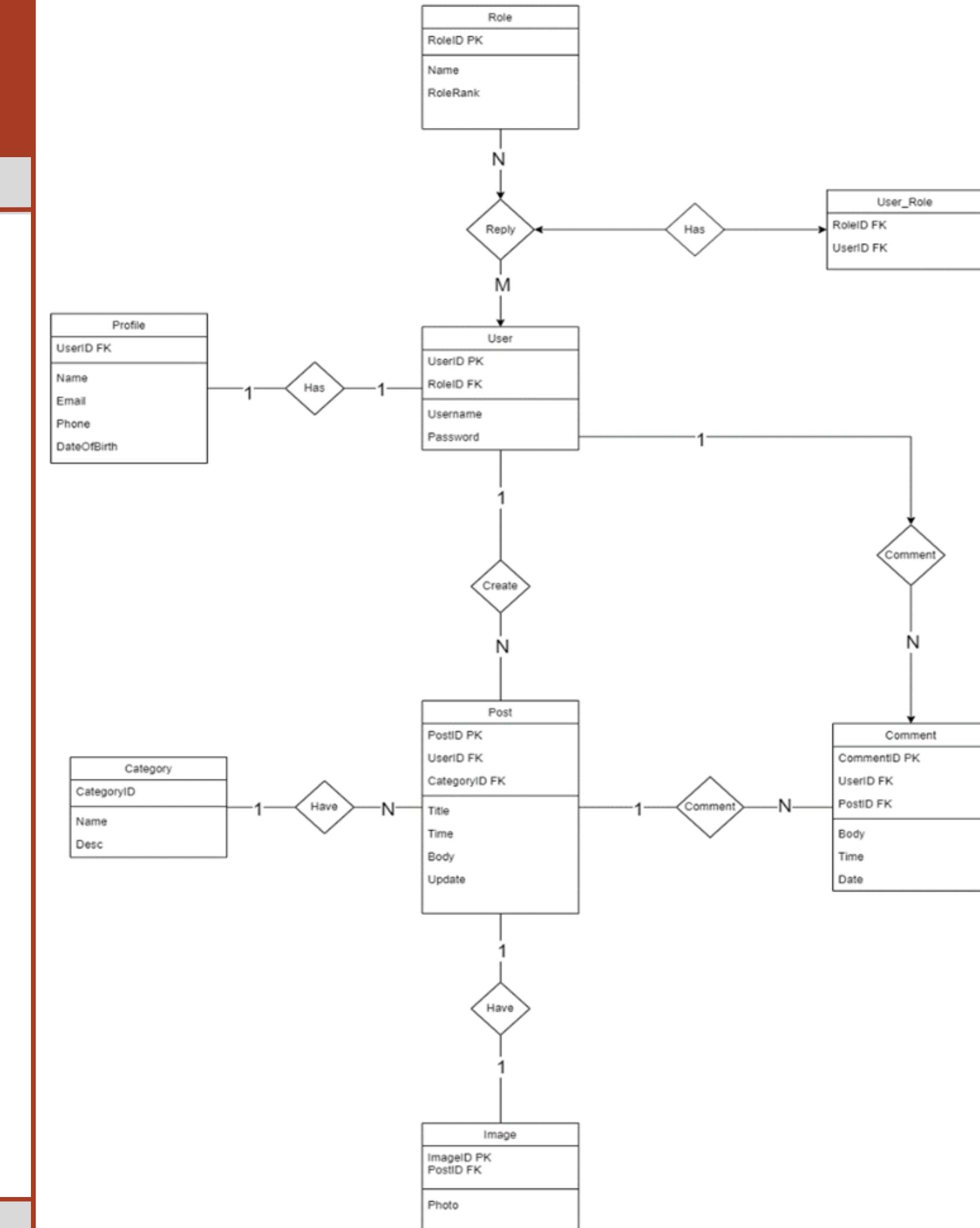
Microsoft Word ใช้สร้างเอกสาร เป็นเครื่องมือที่มีประสิทธิภาพในการเขียนและจัดรูปเอกสาร รวมถึงการจัดรูปแบบข้อความ, รูปภาพ, ตาราง, และอื่น ๆ

USE CASE DIAGRAM

Use Case Diagram Web KKU BLOG

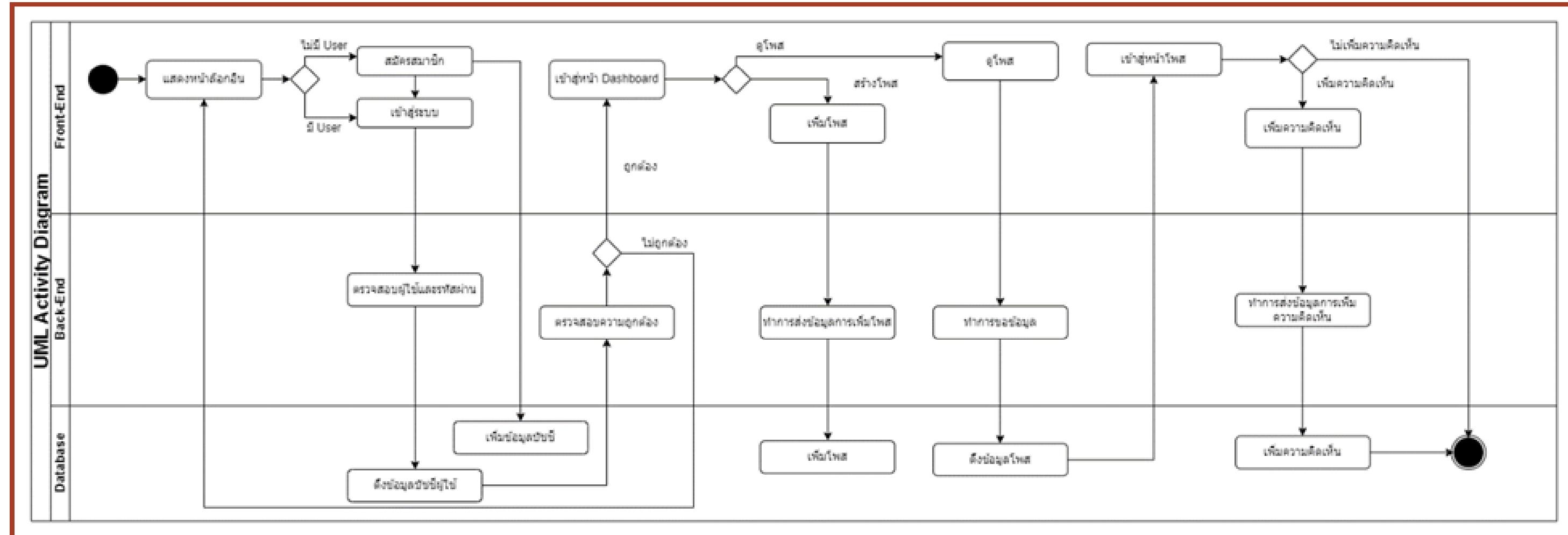


ER-DIAGRAM



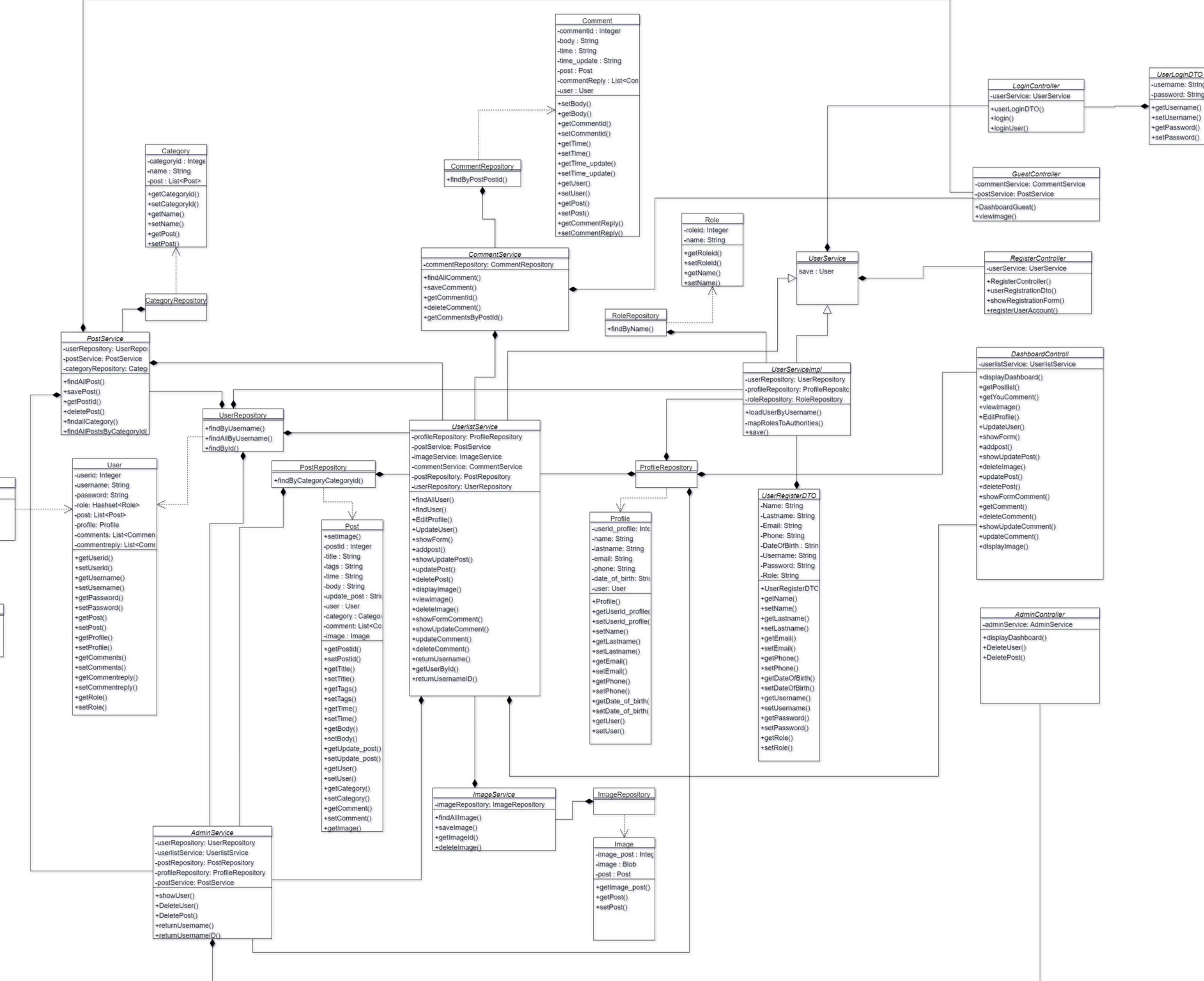


UML ACTIVITY DIAGRAM



KKUBlog

CLASS DIAGRAM





**DEMO เว็บไซต์
KKU BLOG**

THANK
YOU!

