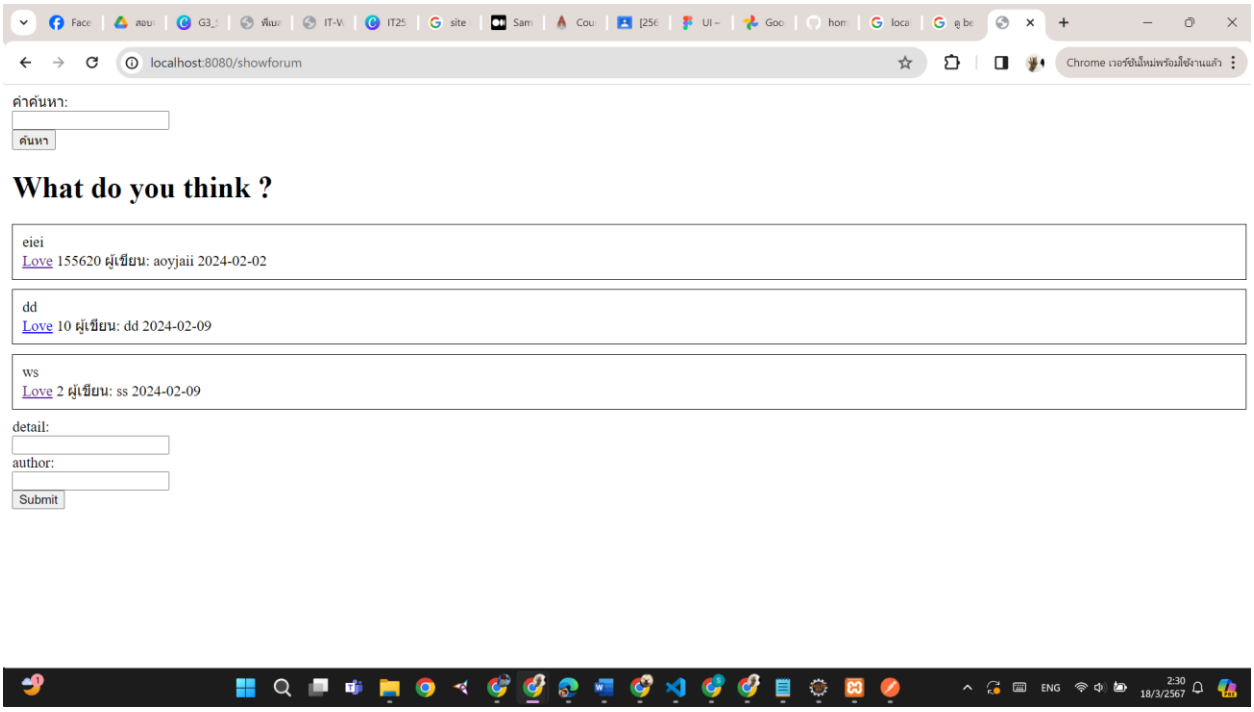
LAB



ForunController

```java
package com.example.eljstl.controller;

import java.util.Date;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;

import com.example.eljstl.model.Forum;
import com.example.eljstl.model.ForumRepository;

@Controller
public class ForumController {


	@Autowired
	ForumRepository repo;

	@GetMapping("/forumssss")
	public String show(Model model) {
		Forum f1 = new Forum();
		f1.setDetail("11");
		return "Forum";
	}
```

```java
	@GetMapping("/addforum")
	public String addPost(
			@RequestParam(value = "detail",defaultValue = "not
found")String dt,
			@RequestParam(value = "author",defaultValue = "not
found")String author,
			@RequestParam(value = "love",defaultValue = "0")Integer
love
			) {
		Forum f1 = new Forum();
		f1.setDetail(dt);
		f1.setAuthor(author);
		f1.setLove(love);
		f1.setPost_date(new Date()); //ทำเวลาให้เป็นปัจจุบัน
		repo.insert(f1);

		return "redirect:/showforum";
	}

	@GetMapping("/like/{id}")
	public String like(@PathVariable Integer id,Model model) {
		Forum Like = repo.findById(id); //เอามาแค่ findbyid ที่อยู่ใน repo
		Like.setLove(Like.getLove() + 1);

		repo.save(Like);
	    return "redirect:/showforum";
	}

//ลบ

	@GetMapping("/dforum/{id}")
	public String delete(@PathVariable Integer id, Model model) {
	    repo.delete(id);
	    return "redirect:/showforum";
	}


	@GetMapping("/showforum")
    public String showAllForum(Model model) {
        List<Forum> forumList = repo.kk();
        model.addAttribute("forums", forumList);
        return "Forum"; // ชื่อของ JSP ที่จะแสดงผล
    }
//ค้นหาอันนี้ไม่มีในแลป
	@GetMapping("/searchforum")
	public String searchForum(@RequestParam(value = "keyword") String
keyword, Model model) {
		List<Forum> searchResult = repo.search(keyword); // เรียกใช้เมธอดค้นหาใน
ForumRepository
		model.addAttribute("searchResult", searchResult); // เพิ่มผลลัพธ์การค้นหาลงในโมเดล
		return "Forum"; // ชื่อของหน้าที่จะแสดงผลผลลัพธ์การค้นหา
	}

}
```

ForumRepository

```
package com.example.eljstl.model;

import java.util.List;

import org.springframework.stereotype.Repository;


import jakarta.persistence.EntityManager;
import jakarta.persistence.PersistenceContext;
import jakarta.persistence.Query;
import jakarta.transaction.Transactional;

@Repository
public class ForumRepository {

	@PersistenceContext
	private EntityManager em;

//เพิ่ม

	@Transactional
	public Forum insert(Forum m) {
		em.persist(m);
		return m;
	}

//ดึงข้อมูลforumมาโชว์ทั้งหมด
	public List<Forum> kk() {
		Query data = em.createQuery("from Forum");
		return data.getResultList();

	}

//หาไอดี
	public Forum findById(Integer id) {
		return em.find(Forum.class, id);
	}

//แก้ไข
	@Transactional
	public Forum save(Forum f) {
```

```
            em.persist(f);
            return f;
        }
//ลบ
        @Transactional
        public void delete(Integer id) {
          Forum forum = findById(id);
          if (forum != null) {
            em.remove(forum);
          }
        }

        public Forum getForum(Integer id) {
          return findById(id);
        }
//ค้นหา
public List<Forum> search(String keyword) {
          Query query = em.createQuery("SELECT f FROM Forum f WHERE f.detail LIKE :keyword");
          query.setParameter("keyword", "%" + keyword + "%");
          return query.getResultList();
        }


}
```

Model

```
package com.example.eljstl.model;

import java.sql.Date;

import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;

@Entity
public class Forum {
        @Id
        @GeneratedValue(strategy = GenerationType.IDENTITY)
        private int id;
        private String detail;
        private String author;
```

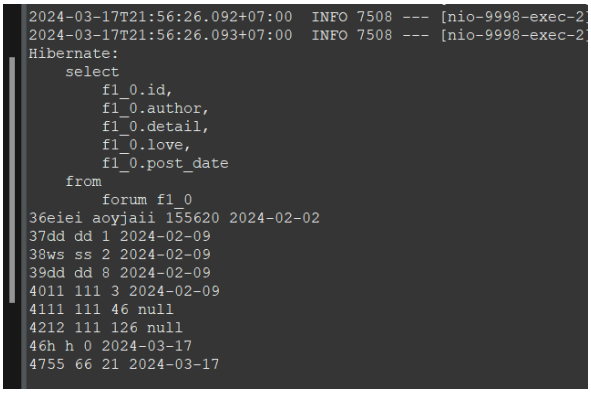
```
	private int love;
	private Date post_date;
	public int getId() {
		return id;
	}
	public void setId(int id) {
		this.id = id;
	}
	public String getDetail() {
		return detail;
	}
	public void setDetail(String detail) {
		this.detail = detail;
	}
	public String getAuthor() {
		return author;
	}
	public void setAuthor(String author) {
		this.author = author;
	}
	public int getLove() {
		return love;
	}
	public void setLove(int love) {
		this.love = love;
	}
	public Date getPost_date() {
		return post_date;
	}
	public void setPost_date(java.util.Date post_date) {
	    this.post_date = new Date(post_date.getTime()); //ทำเวลาให้เป็นปัจจุบัน
	  }
}
```

JSP //มีค้นหาแล้วให้แสดงผลลัพธ์ที่หน้านี้ด้วย

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
<style type="text/css">
```

```
        .forum-box {
            border: 1px solid #000;
            padding: 10px;
            margin-bottom: 10px;
        }
</style>

<body>
<form action="/searchforum" method="get">
    <label for="keyword">คำค้นหา:</label><br>
    <input type="text" id="keyword" name="keyword"><br>
    <input type="submit" value="ค้นหา">
</form>
<h1>What do you think ?</h1>

<c:forEach items="${forums}" var="forum">
<div class="forum-box">
${forum.detail} <br /> <a href="/like/${forum.id}">Love</a>
${forum.love}
ผู้เขียน: ${forum.author} ${forum.post_date}<br />
            <a href="/dforum/${forum.id}" onclick="return confirm('Are you
sure you want to delete this forum?')">Delete</a>

</div>
</c:forEach>

<c:if test="${not empty searchResult}">
    <h2>Search Result:</h2>
    <c:forEach items="${searchResult}" var="forum">
        <div class="forum-box">
            ${forum.detail} <br /> <a href="/like/${forum.id}">Love</a>
            ${forum.love}
            ผู้เขียน: ${forum.author} ${forum.post_date}<br />
            <a href="/dforum/${forum.id}" onclick="return confirm('Are you
sure you want to delete this forum?')">Delete</a>

        </div>
    </c:forEach>
</c:if>

<form action="/addforum" method="get">
            <label for="detail">detail:</label><br> <input type="text"
                id="detail" name="detail"><br> <label
for="author">author:</label><br>
            <input type="text" id="author" name="author"><br> <input
                type="submit" value="Submit">
      </form>
</body>
</html>
```

/อันนี้แลปที่เพิ่มข้อมูลแบบ set ค่าที่ โค้ดเลย แล้วก็โชว์ข้อมูลที่ console ถ้าค้นหาในเว็บ /showforum จะหน้าขาว



Repository

```
package com.javaweb.springjpa.model;


import java.util.List;


import org.springframework.stereotype.Repository;


import jakarta.persistence.EntityManager;
import jakarta.persistence.PersistenceContext;
import jakarta.persistence.Query;
import jakarta.transaction.Transactional;



@Repository
public class ForumRepository {

	@PersistenceContext
		private EntityManager em;

		@Transactional
		public Forum insert(Forum m) {
			em.persist(m);
```

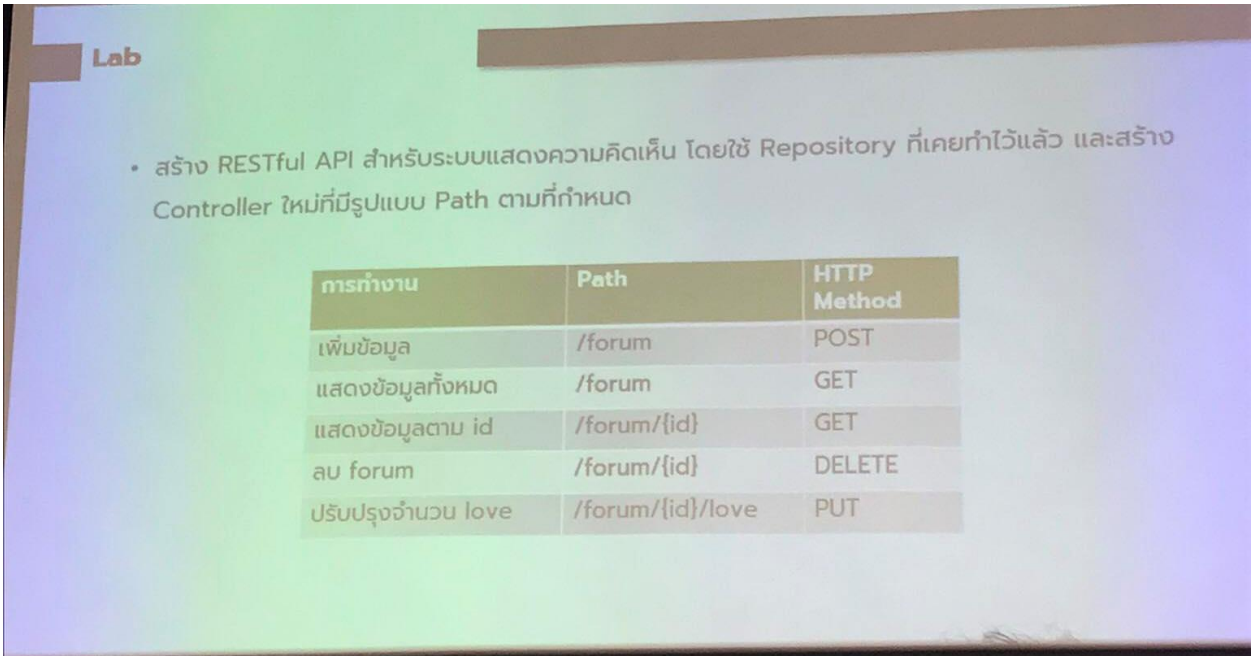
```
                return m;
        }

        public List<Forum> kk() {
                Query data = em.createQuery("from Forum");
                return data.getResultList();

        }

}
```

Controller

```
package com.javaweb.springjpa.controller;

import java.util.Date;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RestController;


import com.javaweb.springjpa.model.Forum;
import com.javaweb.springjpa.model.ForumRepository;


@RestController
```

```
public class ForumController {

		@Autowired
		ForumRepository repo;

		@GetMapping("/addforum") //เพิ่ม
		public void addforum() {
		 Forum m1 = new Forum();
			m1.setDetail("Sasi");
			m1.setAuthor("aoyaoy");
			m1.setLove(1552147);
			m1.setPost_date(new Date());
			repo.insert(m1);

			System.out.println("Save Success");

		}

		@GetMapping("/showforum") //โชว์ข้อมูลที่ comsole
		public void showallforum() {
			List<Forum>  forumList = repo.kk();
			for(Forum m:forumList) {
				System.out.println(m.getId() + ' ' + m.getDetail() + ' ' + m.getAuthor()
+ ' ' + m.getLove()  + ' ' + m.getPost_date());
			}
		}
}
```

แลปAPI



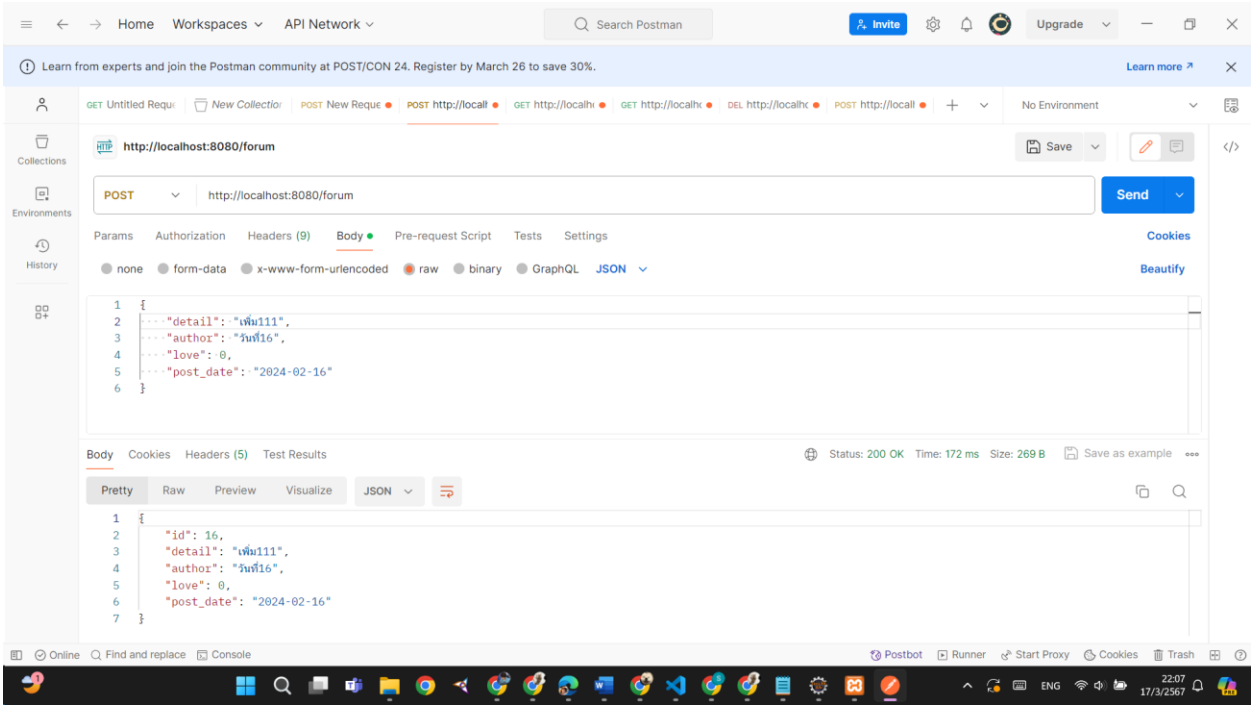
Model

```java
package com.example.eljstl.model;

import java.util.List;

public class ForumData {

	private List<Forum> forum;

	public List<Forum> getForum() {
		return forum;
	}

	public void setForum(List<Forum> forum) {
		this.forum = forum;
	}

	public ForumData(List<Forum> forum) {
		super();
		this.forum = forum;
	}


	public ForumData() {
	}


}
```

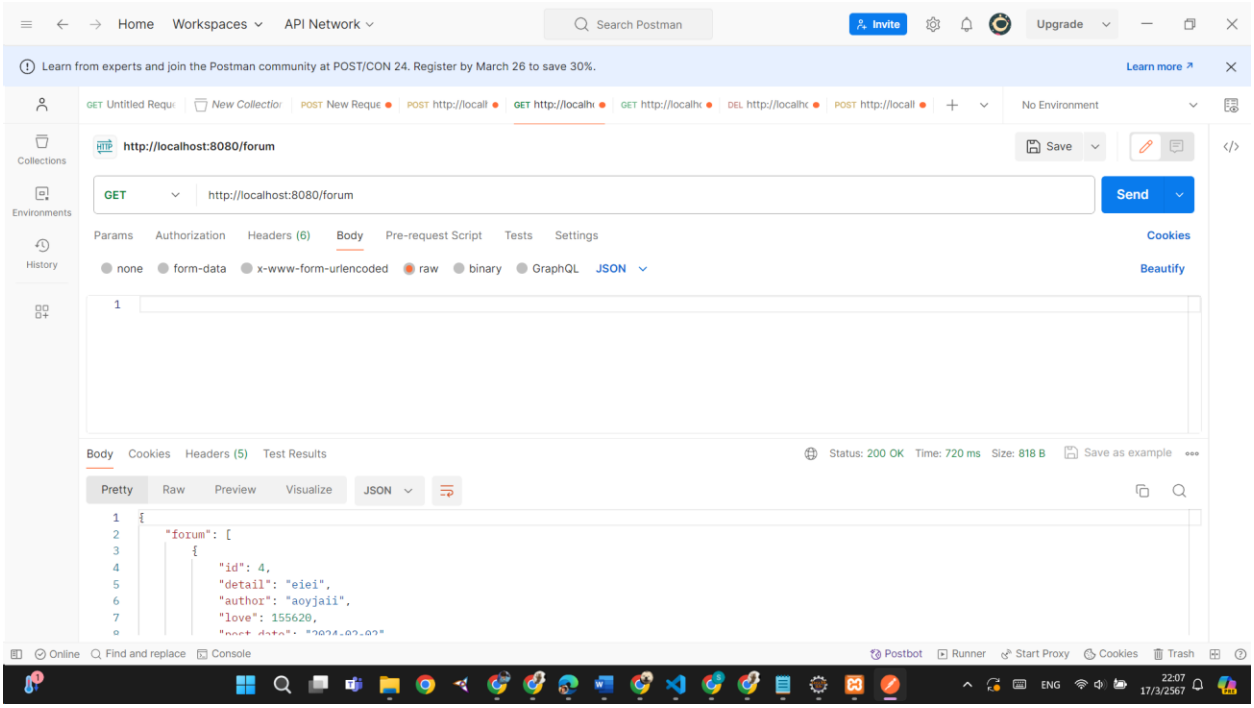
Controller

```
package com.example.eljstl.controller;

import java.util.Collections;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RestController;

import com.example.eljstl.model.Forum;
import com.example.eljstl.model.ForumData;
import com.example.eljstl.model.ForumRepository;


@RestController
public class ForumAPI {

    @Autowired
    ForumRepository repo;

    @PostMapping("/forum")
```

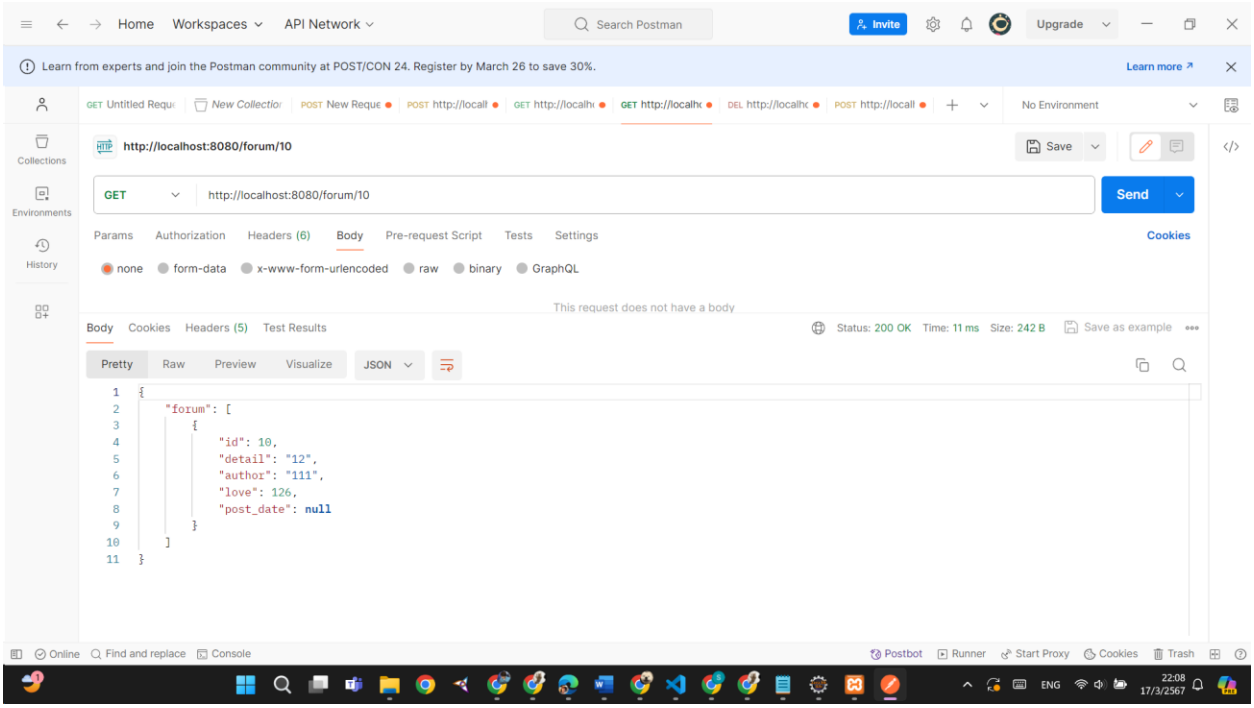
```
public Forum add(@RequestBody Forum forum) {
        return repo.insert(forum);
}
```



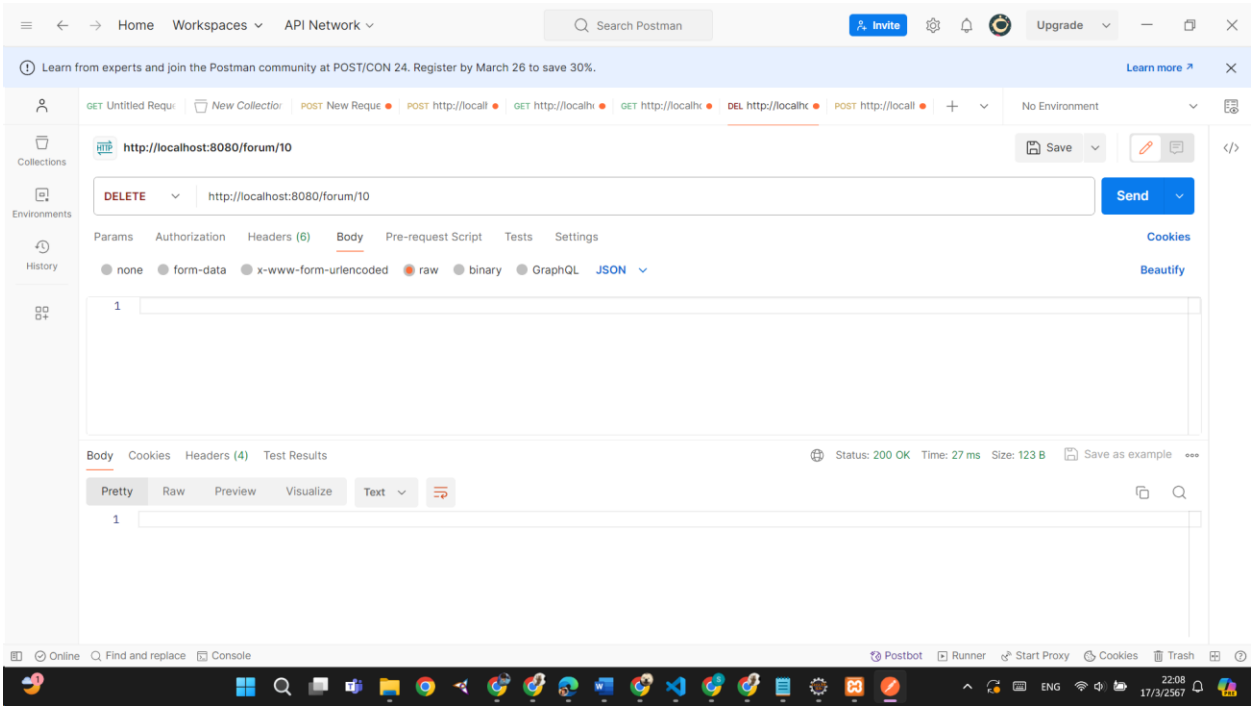

```
@GetMapping("/forum")
public ForumData getAllForum() {
        List<Forum> data = repo.kk();
        return new ForumData(data);
}
```



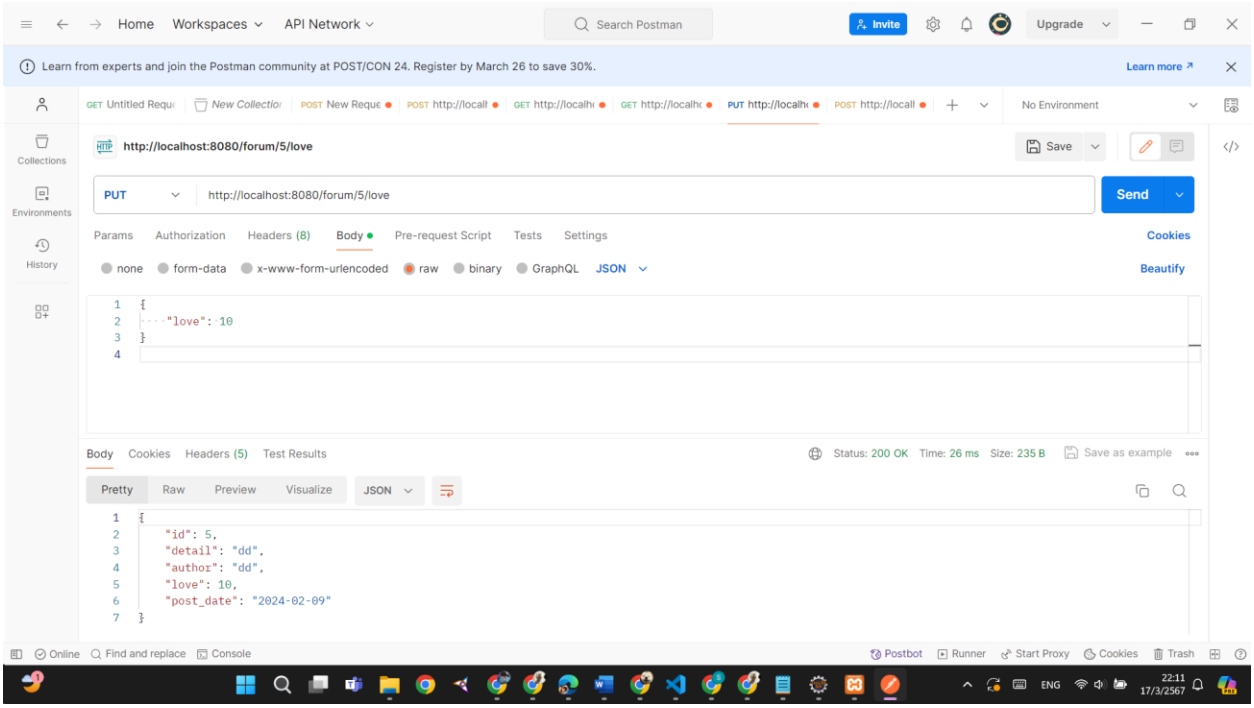

```
@GetMapping("/forum/{id}")
public ForumData getForum(@PathVariable("id") Integer id) {
   Forum forum = repo.findById(id);
   if (forum != null) {
      return new ForumData(Collections.singletonList(forum));
   } else {
      return null;
   }
}
```




```
@DeleteMapping("/forum/{id}")
public void deleteForum(@PathVariable("id") Integer id) {
        repo.delete(id);
        }
```




```
@PutMapping("/forum/{id}/love")
public Forum editForum(@PathVariable("id") Integer id, @RequestBody Forum forum) {
   Forum editForum = repo.findById(id);
   editForum.setLove(forum.getLove());
   return repo.save(editForum);
}
```



}

Repository ใช้อันเดียวกันกับอันบน

```
package com.example.eljstl.model;

import java.util.List;

import org.springframework.stereotype.Repository;


import jakarta.persistence.EntityManager;
import jakarta.persistence.PersistenceContext;
import jakarta.persistence.Query;
import jakarta.transaction.Transactional;

@Repository
```

```
public class ForumRepository {

	@PersistenceContext
	private EntityManager em;

	@Transactional
	public Forum insert(Forum m) {
		em.persist(m);
		return m;
	}

	public List<Forum> kk() {
		Query data = em.createQuery("from Forum");
		return data.getResultList();

	}

	public Forum findById(Integer id) {
		return em.find(Forum.class, id);
	}

	@Transactional
	public Forum save(Forum f) {
		em.persist(f);
		return f;
	}

	@Transactional
	public void delete(Integer id) {
```

```
        Forum forum = findById(id);
        if (forum != null) {
            em.remove(forum);
        }
    }

    public Forum getForum(Integer id) {
        return findById(id);
    }
    public List<Forum> search(String keyword) {
        Query query = em.createQuery("SELECT f FROM Forum f WHERE f.detail LIKE :keyword");
        query.setParameter("keyword", "%" + keyword + "%");
        return query.getResultList();
    }

}
```

Pom ไฟล์

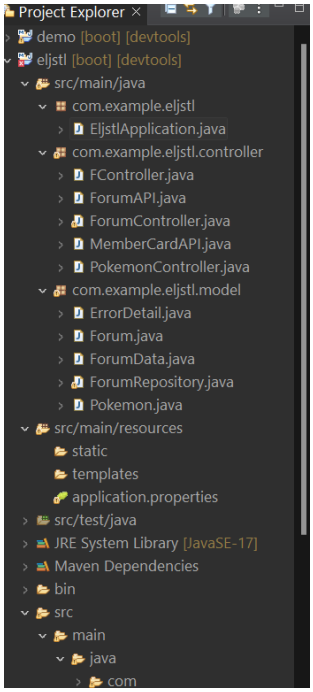
```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
	<modelVersion>4.0.0</modelVersion>
	<parent>
		<groupId>org.springframework.boot</groupId>
		<artifactId>spring-boot-starter-parent</artifactId>
		<version>3.2.2</version>
		<relativePath/> <!-- lookup parent from repository -->
	</parent>
	<groupId>com.example</groupId>
	<artifactId>eljstl</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<name>eljstl</name>
	<description>Demo project for Spring Boot</description>
	<properties>
```

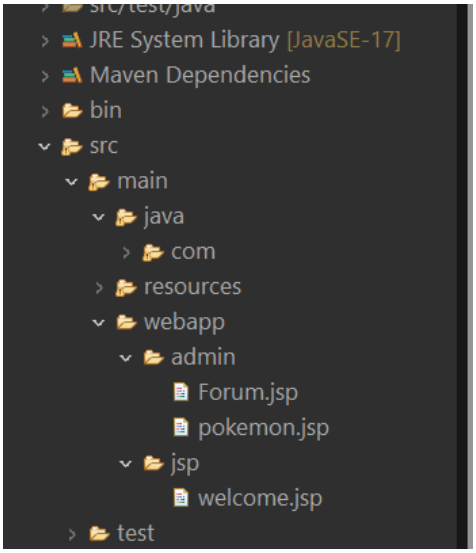
```
            <java.version>17</java.version>
      </properties>
      <dependencies>
            <dependency>
                  <groupId>org.springframework.boot</groupId>
                  <artifactId>spring-boot-starter-data-jpa</artifactId>
            </dependency>
            <dependency>
                  <groupId>org.springframework.boot</groupId>
                  <artifactId>spring-boot-starter-web</artifactId>
            </dependency>

            <dependency>
                  <groupId>org.springframework.boot</groupId>
                  <artifactId>spring-boot-devtools</artifactId>
                  <scope>runtime</scope>
                  <optional>true</optional>
            </dependency>
            <dependency>
                  <groupId>com.mysql</groupId>
                  <artifactId>mysql-connector-j</artifactId>
                  <scope>runtime</scope>
            </dependency>
            <dependency>
                  <groupId>org.springframework.boot</groupId>
                  <artifactId>spring-boot-starter-test</artifactId>
                  <scope>test</scope>
            </dependency>

            <dependency>
    <groupId>org.apache.tomcat.embed</groupId>
    <artifactId>tomcat-embed-jasper</artifactId>
    <version>10.1.18</version>
</dependency>

<dependency>
<groupId>jakarta.servlet.jsp.jstl</groupId>
<artifactId>jakarta.servlet.jsp.jstl-api</artifactId>
<version>3.0.0</version>
</dependency>

<dependency>
<groupId>org.glassfish.web</groupId>
<artifactId>jakarta.servlet.jsp.jstl</artifactId>
<version>3.0.0</version>
</dependency>

<!-- https://mvnrepository.com/artifact/org.glassfish.web/jstl-impl -->
<dependency>
    <groupId>org.glassfish.web</groupId>
    <artifactId>jstl-impl</artifactId>
    <version>1.2</version>
    <scope>runtime</scope>
</dependency>

      </dependencies>
```

```
	<build>
		<plugins>
			<plugin>
				<groupId>org.springframework.boot</groupId>
				<artifactId>spring-boot-maven-plugin</artifactId>
			</plugin>
		</plugins>
	</build>

</project>
```

App.properties

```
#server.port=9787
spring.mvc.view.prefix=/admin/
spring.mvc.view.suffix=.jsp

spring.datasource.url = jdbc:mysql://localhost/jpa?characterEncoding=utf-8
# characterEncoding=utf-
8&useUnicode=true&useJDBCCompliantTimezoneShift=true&useLegacyDatetimeCode=fa
lse&serverTimezone=UTC

spring.datasource.username=root
spring.datasource.password=

spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true
hibernate.generate_statistics = true
spring.jpa.hibernate.ddl-auto = update
# spring.jpa.properties.hibernate.dialect =
org.hibernate.dialect.MySQLDialect
```





Project Web นอย

```
server.port=8899

spring.mvc.view.prefix=/jsp/
spring.mvc.view.suffix=.jsp

spring.datasource.url = jdbc:mysql://localhost:/blog?characterEncoding=utf-8
spring.datasource.username = root
spring.datasource.password =

spring.jpa.show-sql = true
spring.jpa.properties.hibernate.format_sql=true
spring.jpa.hibernate.ddl-auto = update
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL8Dialect
```

Controller

//Auth

```
package com.java.demo.controller;
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import com.java.demo.model.User;
import com.java.demo.repository.UserRepository;
import jakarta.servlet.http.HttpServletRequest;
@Controller
public class AuthController {
        @Autowired
  private UserRepository userRepository;
        @GetMapping("/register")
  public String showRegistrationForm(Model model) {
    model.addAttribute("user", new User());
    return "register";
  }
  @PostMapping("/register")
  public String registerUser(User user, Model model) {
    // ตรวจสอบว่าชื่อผู้ใช้ซ้ำหรือไม่
    if (userRepository.existsByUsername(user.getUsername())) {
      model.addAttribute("error", "ชื่อผู้ใช้นี้มีอยู่แล้ว กรุณาเลือกชื่อผู้ใช้อื่น");
      return "register";
    }
    // บันทึกผู้ใช้
    userRepository.save(user);
```

```
    return "redirect:/login";
}
        @GetMapping("/login")
          public String showLoginForm(Model model) {
            model.addAttribute("user", new User()); // เพิ่ม user เพื่อให้มีข้อมูลในกรณีที่เกิด error
            return "login";
          }
        @PostMapping("/login")
        public String loginUser(User user, Model model, HttpServletRequest request) {
          // ตรวจสอบ user ในฐานข้อมูล
          List<User> users = userRepository.findByUsername(user.getUsername());

          if (!users.isEmpty()) {
              User foundUser = users.get(0);
              if (foundUser.getPassword().equals(user.getPassword())) {
                // การเข้าสู่ระบบสำเร็จ ส่งชื่อผู้ใช้ไปยังหน้า dashboard
                model.addAttribute("username", foundUser.getUsername());
                request.getSession().setAttribute("loggedInFirstname", foundUser.getFirstname());
                request.getSession().setAttribute("loggedInLastname", foundUser.getLastname());

               // เก็บชื่อผู้ใช้ใน session
                request.getSession().setAttribute("loggedInUsername", foundUser.getUsername());
                return "redirect:/dashboard";
              }
            }
          // การเข้าสู่ระบบไม่สำเร็จ ใส่ข้อความผิดพลาดลงใน model
          model.addAttribute("error", "ชื่อผู้ใช้หรือรหัสผ่านไม่ถูกต้อง");
          return "login";
        }
        }
```

//FirstController

```
package com.java.demo.controller;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;

@Controller
public class FirstController {

@GetMapping("/")
public String welcome(Model model) {
```

```
model.addAttribute("message", "Civilized Land");
return "/home";
}
}
```

//PostController

```
package com.java.demo.controller;

import java.util.Date;
import java.util.HashSet;
import java.util.List;
import java.util.Set;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestParam;

import com.java.demo.model.Comment;
import com.java.demo.model.Post;
import com.java.demo.repository.PostRepository;

import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpSession;
import jakarta.transaction.Transactional;

@Controller
public class PostController {
	@Autowired PostRepository repo;


	@GetMapping("/dashboard")
  public String showPost(Model model, HttpSession session) {
    // ดึงชื่อผู้ใช้จาก session
    String loggedInUsername = (String) session.getAttribute("loggedInUsername");

    // หรือดึงชื่อผู้ใช้จาก model attribute
    // String loggedInUsername = (String) model.getAttribute("username");

    if (loggedInUsername != null) {
```

```
        List<Post> allpost = repo.kk();
        model.addAttribute("posts", allpost);
        model.addAttribute("loggedInUsername", loggedInUsername); // ส่งชื่อผู้ใช้ไปยังหน้า dashboard


        return "dashboard";
    } else {
        // ไม่มีชื่อผู้ใช้ที่เข้าสู่ระบบ, ส่งไปยังหน้า login หรือทำการ redirect
        return "redirect:/login";
    }
        }
//      ไปหน้า เพิ่ม
        @GetMapping("/addpost")
        public String addpost(Model model) {

                return "addpost";
                }

        @GetMapping("/addpostself")
        public String addpostself(Model model) {

                return "addpostself";
                }


        // คลาส Controller
        @PostMapping("/addpost")
        public String addPost(
            @RequestParam String detail,
            @RequestParam String category,
            @RequestParam(value = "love", defaultValue = "0") Integer love,
            @RequestParam(required = false) String sourcePage, // เพิ่มพารามิเตอร์ sourcePage
            HttpSession session, Model model ,HttpServletRequest request) {

          // ประมวลผลข้อมูลฟอร์มตามที่ต้องการ
          // ในที่นี้คือการเพิ่มข้อมูลลงใน repository หรือฐานข้อมูล
          Post post = new Post();
          post.setCategory(category);
          post.setDetail(detail);

          // ดึงชื่อผู้ใช้จาก session
```

```
    String loggedInUsername = (String) session.getAttribute("loggedInUsername");

    if (loggedInUsername != null) {
        post.setAuthor(loggedInUsername);
    } else {
        return "redirect:/login";
    }
    post.setTimestamp(new java.sql.Date(new java.util.Date().getTime()));
    post.setLove(love);
    repo.insert(post);

   // ตรวจสอบ sourcePage และส่งผู้ใช้กลับไปยังหน้าที่ถูกต้อง
    if ("profile".equals(sourcePage)) {
        return "redirect:/profile";
    } else if ("dashboard".equals(sourcePage)) {
        return "redirect:/dashboard";
    } else {
        // หากไม่ระบุ sourcePage หรือไม่ถูกต้องให้ส่งผู้ใช้กลับไปยังหน้า dashboard เป็นค่าเริ่มต้น
        return "redirect:/dashboard";
    }
}

@PostMapping("/addpostself")
public String addPostself(
        @RequestParam String detail,
        @RequestParam String category,
        @RequestParam(value = "love", defaultValue = "0") Integer love,
        @RequestParam(required = false) String sourcePage, // เพิ่มพารามิเตอร์ sourcePage
        HttpSession session, Model model ,HttpServletRequest request) {

    // ประมวลผลข้อมูลฟอร์มตามที่ต้องการ
    // ในที่นี้คือการเพิ่มข้อมูลลงใน repository หรือฐานข้อมูล
    Post post = new Post();
    post.setCategory(category);
    post.setDetail(detail);

    // ดึงชื่อผู้ใช้จาก session
    String loggedInUsername = (String) session.getAttribute("loggedInUsername");

    if (loggedInUsername != null) {
        post.setAuthor(loggedInUsername);
    } else {
```

```
            return "redirect:/login";
        }
        post.setTimestamp(new java.sql.Date(new java.util.Date().getTime()));
        post.setLove(love);
        repo.insert(post);

    // ตรวจสอบ sourcePage และส่งผู้ใช้กลับไปยังหน้าที่ถูกต้อง
        if ("profile".equals(sourcePage)) {
            return "redirect:/profile";
        } else if ("dashboard".equals(sourcePage)) {
            return "redirect:/dashboard";
        } else {
            // หากไม่ระบุ sourcePage หรือไม่ถูกต้องให้ส่งผู้ใช้กลับไปยังหน้า dashboard เป็นค่าเริ่มต้น
            return "redirect:/dashboard";
        }
    }



    @GetMapping("/logout")
    public String logout(HttpSession session) {
        // ลบ session ทั้งหมดที่เกี่ยวข้องกับผู้ใช้
        session.invalidate();

        // ส่งไปยังหน้า login หรือหน้าที่คุณต้องการให้ผู้ใช้ไปหลังจาก logout
        return "redirect:/login";
    }
    @GetMapping("/like/{id}")
    public String like(@PathVariable Integer id, Model model, HttpSession session,
HttpServletRequest request) {
        String loggedInUsername = (String) session.getAttribute("loggedInUsername");

        if (loggedInUsername != null) {
            // เรียก method เพื่อดึงข้อมูลการไลค์ของผู้ใช้จากฐานข้อมูล
            Set<Integer> likedPostIds = repo.findLikedPostIdsByUsername(loggedInUsername);

            if (!likedPostIds.contains(id)) {
                Post post = repo.findById(id);

                if (post != null) {
                    if (!post.getLikedUsernames().contains(loggedInUsername)) {
```

```
                    // ยังไม่ได้ไลค์, กดไลค์และเพิ่ม ID ของโพสต์เข้าไปในฐานข้อมูล
                    post.setLove(post.getLove() + 1);
                    post.getLikedUsernames().add(loggedInUsername);
                    repo.save(post);
                  }
                }
              } else {
                // ไลค์ไปแล้ว, ทำการ Unlike
                repo.unlikePost(id, loggedInUsername);
              }
            }

            // ดึง URL ของหน้านั้นก่อนที่จะทำการไลค์
            String referer = request.getHeader("Referer");

            // ทำการ redirect ไปที่ URL ของหน้าที่ผู้ใช้เคยเข้ามาก่อนที่จะทำการไลค์
            return "redirect:" + referer;
        }

        //ค้นหาโพส
                @GetMapping("/search")
                  public String searchPosts(@RequestParam(required = false) String keyword,
                                            @RequestParam(required = false) String category, Model model) {
                    List<Post> allPosts;

                    if ((keyword != null && !keyword.isEmpty()) || (category != null &&
!category.isEmpty())) {
                        // If keyword or category is provided, perform search
                        allPosts = repo.searchPosts(keyword, category);
                      } else {
                        // Otherwise, fetch all posts
                        allPosts = repo.kk();
                      }

//                    if (keyword != null && !keyword.isEmpty()) {
//                      // If keyword is provided, perform search
//                      allPosts = repo.searchPosts(keyword);
//                    } else {
//                      // Otherwise, fetch all posts
//                      allPosts = repo.kk();
//                    }
```

```
                model.addAttribute("posts", allPosts);
                model.addAttribute("keyword", keyword);


                return "dashboard";


            }


//              @GetMapping("/searchcate")
//              public String searchPosts(@RequestParam(required = false) String keyword,
//                      @RequestParam(required = false) String category,
//                      Model model) {
//                List<Post> allPosts;
//
//                if ((keyword != null && !keyword.isEmpty()) || (category != null &&
!category.isEmpty())) {
//                  // If keyword or category is provided, perform search
//                  allPosts = repo.searchPosts(keyword, category);
//                } else {
//                  // Otherwise, fetch all posts
//                  allPosts = repo.kk();
//                }
//
//                model.addAttribute("posts", allPosts);
//                model.addAttribute("keyword", keyword);
//
//                return "dashboard";
//              }


              //แก้ไขโพส

                    // แสดงหน้าฟอร์มแก้ไขโพสท์
              @GetMapping("/editpost/{id}")
              public String editPostForm(@PathVariable Integer id, Model model, HttpSession
session) {
                Post post = repo.findById(id);

                // Check if the logged-in user is the author of the post
                if (post != null && isLoggedInUserAuthor(session, post.getAuthor())) {
                  model.addAttribute("post", post);
                  return "editpost";
```

```
        } else {
            // If not authorized, you can redirect to an error page or handle it accordingly
            return "redirect:/dashboard";
        }
    }

    @PostMapping("/editpost/{id}")
    public String editPost(@PathVariable Integer id, @RequestParam String detail,
HttpSession session) {
        Post post = repo.findById(id);

        // Check if the logged-in user is the author of the post
        if (post != null && isLoggedInUserAuthor(session, post.getAuthor())) {
            post.setDetail(detail);
            repo.save(post);
            return "redirect:/dashboard";
        } else {
            // If not authorized, you can redirect to an error page or handle it accordingly
            return "redirect:/dashboard";
        }
    }

    @GetMapping("/delete/{id}")
    public String deletePost(@PathVariable Integer id, HttpSession
session,HttpServletRequest request) {
        Post post = repo.findById(id);
        String referer = request.getHeader("Referer");
        // Check if the logged-in user is the author of the post
        if (post != null && isLoggedInUserAuthor(session, post.getAuthor())) {
            repo.deleteById(id);

                return "redirect:" + referer;
        } else {
            // If not authorized, you can redirect to an error page or handle it accordingly

                return "redirect:" + referer;
        }
    }

    // Helper method to check if the logged-in user is the author of the post
    private boolean isLoggedInUserAuthor(HttpSession session, String postAuthor) {
        String loggedInUsername = (String) session.getAttribute("loggedInUsername");
        return loggedInUsername != null && loggedInUsername.equals(postAuthor);
    }
```

//คอมเม้นต์

```
@Transactional
  @PostMapping("/addComment")
  public String addComment(
        @RequestParam Integer postId,
        @RequestParam String comment,
        HttpSession session
  ) {
     String loggedInUsername = (String) session.getAttribute("loggedInUsername");
     repo.addComment(postId, loggedInUsername, comment);

     return "redirect:/dashboard";
  }

//โปรไฟล์
 @GetMapping("/profile")
 public String showProfile(Model model, HttpSession session) {
   // ดึงชื่อผู้ใช้จาก session
   String loggedInUsername = (String) session.getAttribute("loggedInUsername");

   // หรือดึงชื่อผู้ใช้จาก model attribute
   // String loggedInUsername = (String) model.getAttribute("username");

   if (loggedInUsername != null) {
     List<Post> userPosts = repo.findByAuthor(loggedInUsername);
     model.addAttribute("posts", userPosts);
     model.addAttribute("loggedInUsername", loggedInUsername); // ส่งชื่อผู้ใช้ไปยัง
หน้า dashboard
     return "profile";
   } else {
     // ไม่มีชื่อผู้ใช้ที่เข้าสู่ระบบ, ส่งไปยังหน้า login หรือทำการ redirect
     return "redirect:/login";
   }
 }
}
```

//ProfileController

```
package com.java.demo.controller;

import java.security.Principal;
```

```
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;

import com.java.demo.model.Post;
import com.java.demo.repository.PostRepository;

import jakarta.servlet.http.HttpSession;

@Controller
public class ProfileController {
	@Autowired PostRepository repo;

 // ใน Controller ที่แสดงโปรไฟล์
	@GetMapping("/profile1")
	public String showProfilePage(Model model, HttpSession session) {
	  String loggedInUsername = (String) session.getAttribute("loggedInUsername");

	  if (loggedInUsername != null) {
        List<Post> allpost = repo.kk();
        model.addAttribute("posts", allpost);
        model.addAttribute("loggedInUsername", loggedInUsername); // ส่งชื่อผู้ใช้ไปยังหน้า dashboard
	    return "profile";
	  } else {
	    // ไม่มีชื่อผู้ใช้ที่เข้าสู่ระบบ, ส่งไปยังหน้า login หรือทำการ redirect
	    return "redirect:/login";
	  }
	}
}
```

Model

//comment.java

```
package com.java.demo.model;

import java.util.Date;

import com.fasterxml.jackson.annotation.JsonIgnore;

import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
```

```
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import jakarta.persistence.ManyToOne;

@Entity
public class Comment {
	@Id
	  @GeneratedValue(strategy = GenerationType.IDENTITY)
	  private int id;

	  private String author;
	  private String content;
	  private Date timestamp;

	  @ManyToOne
	  @JsonIgnore
	  private Post post;

	  public Comment() {
	    // constructor ตามปกติ
	  }

	  public Comment(String author, String content, Date timestamp, Post post) {
	    this.author = author;
	    this.content = content;
	    this.timestamp = timestamp;
	    this.post = post;
	  }

		public int getId() {
			return id;
		}

		public void setId(int id) {
			this.id = id;
		}

		public String getAuthor() {
			return author;
		}

		public void setAuthor(String author) {
			this.author = author;
		}
```

```
	public String getContent() {
		return content;
	}

	public void setContent(String content) {
		this.content = content;
	}

	public Date getTimestamp() {
		return timestamp;
	}

	public void setTimestamp(Date timestamp) {
		this.timestamp = timestamp;
	}

	public Post getPost() {
		return post;
	}

	public void setPost(Post post) {
		this.post = post;
	}
}
```

//Post.java

```
package com.java.demo.model;

import java.sql.Date;
import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Set;
import jakarta.persistence.CascadeType;
import jakarta.persistence.Column;
import jakarta.persistence.ElementCollection;
import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import jakarta.persistence.JoinColumn;
import jakarta.persistence.OneToMany;
import jakarta.persistence.Temporal;
```

```
import jakarta.persistence.TemporalType;

@Entity
public class Post {

  @Id
  @GeneratedValue(strategy = GenerationType.IDENTITY)
  private int id;

  private String category;
  private String detail;
  private String author;
  private int love;
  private Date timestamp;

  @ElementCollection
  @Column(name = "liked_username")
  private Set<String> likedUsernames = new HashSet<>();

  @OneToMany(cascade = CascadeType.ALL, orphanRemoval = true)
  @JoinColumn(name = "post_id")
  private List<Comment> comments = new ArrayList<>();

  public int getId() {
    return id;
  }

  public Date getTimestamp() {
    return timestamp;
  }

  public void setTimestamp(Date timestamp) {
    this.timestamp = timestamp;
  }

  public Set<String> getLikedUsernames() {
    return likedUsernames;
  }

  public void setLikedUsernames(Set<String> likedUsernames) {
    this.likedUsernames = likedUsernames;
  }

  public List<Comment> getComments() {
```

```
    return comments;
}

public void setComments(List<Comment> comments) {
    this.comments = comments;
}

public void addComment(Comment comment) {
    comments.add(comment);
    comment.setPost(this);
}

public void unlikePost(String username) {
    if (likedUsernames.remove(username)) {
        setLove(getLove() - 1);
    }
}

public void setId(int id) {
    this.id = id;
}

public String getDetail() {
    return detail;
}

public void setDetail(String detail) {
    this.detail = detail;
}

public String getAuthor() {
    return author;
}

public void setAuthor(String author) {
    this.author = author;
}

public int getLove() {
    return love;
}

public void setLove(int love) {
    this.love = love;
```

```
    }

    public String getCategory() {
        return category;
    }

    public void setCategory(String category) {
        this.category = category;
    }
}
```

//User.java

```
package com.java.demo.model;

import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;

@Entity
public class User {

        @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

        private String firstname;
        private String lastname;
    private String username;
    private String password;
        public int getId() {
                return id;
        }
        public void setId(int id) {
                this.id = id;
        }
        public String getUsername() {
                return username;
        }
        public void setUsername(String username) {
                this.username = username;
        }
        public String getPassword() {
```

```
		return password;
	}
	public void setPassword(String password) {
		this.password = password;
	}
	public User(int id, String username, String password) {
		super();
		this.id = id;
		this.username = username;
		this.password = password;
	}

	public String getFirstname() {
		return firstname;
	}
	public void setFirstname(String firstname) {
		this.firstname = firstname;
	}
	public String getLastname() {
		return lastname;
	}
	public void setLastname(String lastname) {
		this.lastname = lastname;
	}
	public User() {
		super();
	}
}
```

Repository

//PostRepository

```
package com.java.demo.repository;

import java.util.ArrayList;

import java.util.Date;

import java.util.HashSet;
import java.util.List;
import java.util.Set;
import org.springframework.stereotype.Repository;
import com.java.demo.model.Comment;
import com.java.demo.model.Post;
import jakarta.persistence.EntityManager;
import jakarta.persistence.PersistenceContext;
```

```
import jakarta.persistence.Query;
import jakarta.transaction.Transactional;

@Repository
public class PostRepository {

  @PersistenceContext
  private EntityManager em;

  // เพิ่ม
  @Transactional
  public Post insert(Post m) {
    em.persist(m);
    return m;
  }

  // ดึงข้อมูลมาโชว์
  public List<Post> kk() {
    Query data = em.createQuery("from Post");
    return data.getResultList();
  }

  public Post findById(Integer id) {
    return em.find(Post.class, id);
  }

  @Transactional
  public Post save(Post post) {
    // ตรวจสอบว่าโพสต์นี้ยังไม่ได้ถูกกดไลค์โดย username ที่ระบุ
    if (!postIsLikedByUser(post.getId(), post.getAuthor())) {
      // ถ้ายังไม่ถูกกดไลค์, ก็ดำเนินการบันทึกข้อมูล
      em.merge(post); // ใช้ merge แทน persist เพื่ออัปเดตข้อมูลโพสต์ที่มี id นี้
      return post;
    }
    // ถ้าโพสต์ถูกกดไลค์แล้ว ไม่ต้องทำอะไร
    return null;
  }

  private boolean postIsLikedByUser(Integer postId, String username) {
    Query query = em.createQuery("FROM Post WHERE id = :postId AND love > 0 AND author =
:username")
        .setParameter("postId", postId)
```

```
            .setParameter("username", username);

    List<Post> resultList = query.getResultList();

    return !resultList.isEmpty();
  }
  @Transactional
  public void unlikePost(Integer postId, String username) {
    Post post = em.find(Post.class, postId);
    if (post != null) {
      // ตรวจสอบว่าโพสต์นี้ได้รับไลค์จากผู้ใช้นี้หรือไม่
      if (post.getLikedUsernames().contains(username)) {
        post.setLove(post.getLove() - 1);
        post.getLikedUsernames().remove(username);
        em.merge(post);
      }
    }
  }

  // เพิ่ม method เพื่อเรียกข้อมูลการไลค์โพสต์จากฐานข้อมูล
  public Set<Integer> findLikedPostIdsByUsername(String username) {
    Query query = em.createQuery("SELECT p.id FROM Post p WHERE :username MEMBER OF
p.likedUsernames")
            .setParameter("username", username);

    List<Integer> resultList = query.getResultList();

    return new HashSet<>(resultList);
  }

 //ค้นหา
  public List<Post> searchPosts(String keyword) {
    Query query = em.createQuery("from Post where detail like :keyword or author like :keyword");
    query.setParameter("keyword", "%" + keyword + "%");
    return query.getResultList();
  }



  public List<Post> searchPosts(String keyword, String category) {
    // ตรวจสอบว่ามีการระบุ category หรือไม่
    if (category != null && !category.isEmpty()) {
```

```
        Query query = em.createQuery("from Post where detail like :keyword and category =
:category");
        query.setParameter("category", category);
        query.setParameter("keyword", "%" + keyword + "%");
        return query.getResultList();
      } else {
        // ถ้าไม่ระบุ category, ให้ค้นหาโพสท์ทั้งหมด
        Query query = em.createQuery("from Post where detail like :keyword");
        query.setParameter("keyword", "%" + keyword + "%");
        return query.getResultList();
      }
    }

    //ลบ
    @Transactional
    public void deleteById(Integer id) {
      Post post = em.find(Post.class, id);
      em.remove(post);
    }



//คอมเม้นต์
 // เพิ่ม method สำหรับเพิ่มคอมเม้นต์
    @Transactional
    public void addComment(Integer postId, String username, String comment) {
      Post post = em.find(Post.class, postId);

      if (post != null) {
        Comment newComment = new Comment();
        newComment.setAuthor(username);
        newComment.setContent(comment);
        newComment.setTimestamp(new Date());
        newComment.setPost(post); // ตั้งค่า post ให้กับ comment

        post.getComments().add(newComment);
        em.merge(post);
      }
    }

    public List<Post> findByAuthor(String author) {
      Query query = em.createQuery("from Post where author = :author");
```

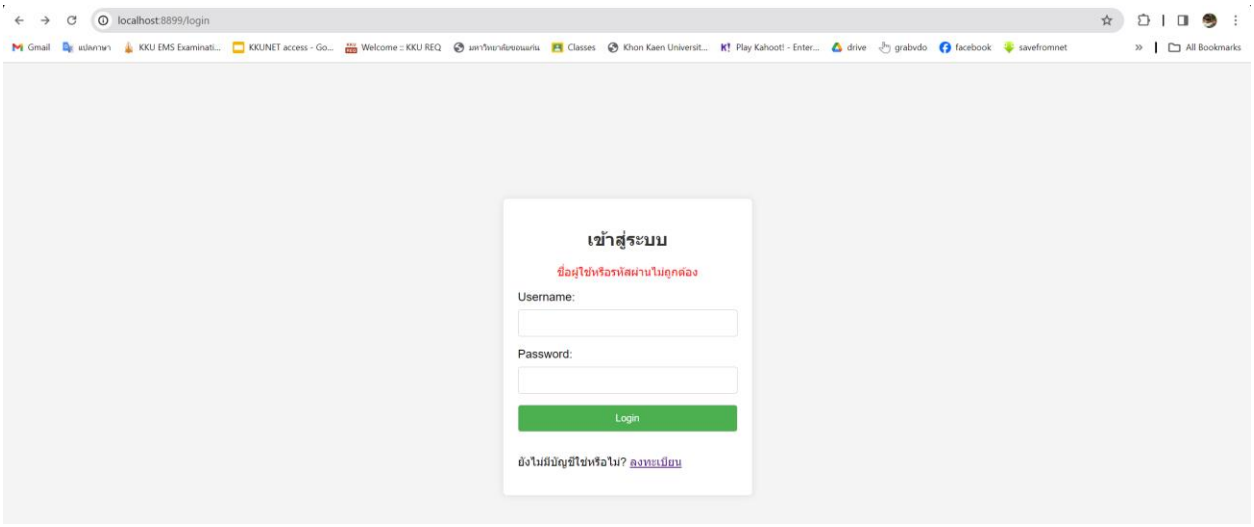
```
        query.setParameter("author", author);
        return query.getResultList();
    }



}
```

//UserRepository สร้างเป็น interface

```
package com.java.demo.repository;

import java.util.List;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;

import com.java.demo.model.User;

public interface UserRepository extends JpaRepository<User, Integer> {
        List<User> findByUsername(String username);

        @Query("SELECT CASE WHEN COUNT(u) > 0 THEN true ELSE false END FROM User u
WHERE u.username = :username")
    boolean existsByUsername(@Param("username") String username);
}
```

JSP

//login

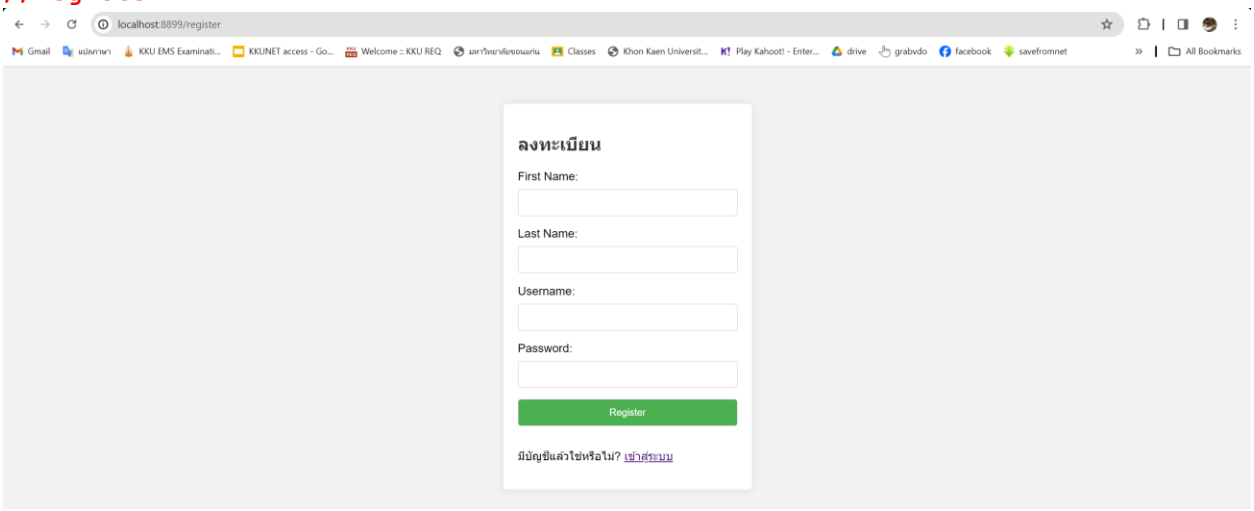



```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
pageEncoding="utf-8" %>
<html>
<head>
    <title>Login</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 0;
            display: flex;
            align-items: center;
            justify-content: center;
            height: 100vh;
        }

        h2 {
            color: #333;
            text-align: center;
        }

        form {
            background-color: #fff;
            padding: 20px;
            border-radius: 5px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            width: 300px;
        }

        label {
            display: block;
            margin-bottom: 8px;
        }

        input {
            width: 100%;
            padding: 10px;
            margin-bottom: 15px;
```

```
            border: 1px solid #ccc;
            border-radius: 4px;
            box-sizing: border-box;
        }

        input[type="submit"] {
            background-color: #4caf50;
            color: white;
            cursor: pointer;
        }

        input[type="submit"]:hover {
            background-color: #45a049;
        }

        div.error {
            color: red;
            margin-bottom: 15px;
            text-align: center;
        }

        form.register-form {
            margin-top: 20px;
            text-align: center;
        }

        form.register-form input[type="submit"] {
            background-color: #2196F3;
        }

        form.register-form input[type="submit"]:hover {
            background-color: #0b7dda;
        }
    </style>
</head>
<body>
    <form action="${pageContext.request.contextPath}/login" method="post">
        <h2>เข้าสู่ระบบ</h2>

        <!-- แสดงข้อความผิดพลาด (ถ้ามี) -->
        <c:if test="${not empty error}">
            <div class="error">
                ${error}
            </div>
        </c:if>

        <label for="username">Username:</label>
        <input type="text" id="username" name="username" required>
        <br>
        <label for="password">Password:</label>
        <input type="password" id="password" name="password" required>
        <br>
        <input type="submit" value="Login">
        <div class="register-link">
            <p>ยังไม่มีบัญชีใช่หรือไม่? <a href="/register">ลงทะเบียน</a></p>
        </div>
```

```
    </form>
</body>
</html>
```

//register

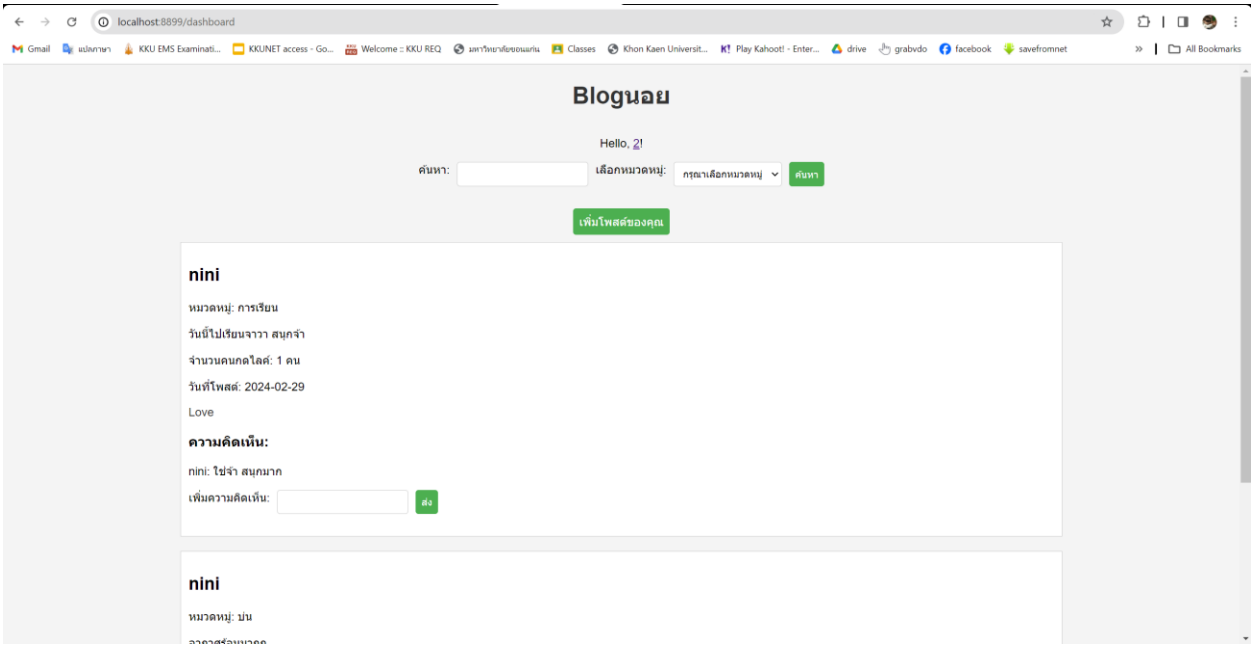



```
<%@ page language="java" contentType="text/html; charset=utf-8"
pageEncoding="utf-8" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<html>
<head>
    <title>Register</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f2f2f2;
            margin: 0;
            padding: 0;
        }

        h2 {
            color: #333;
        }

        form {
            background-color: #fff;
            padding: 20px;
            border-radius: 5px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            width: 300px;
            margin: 50px auto;
        }

        label {
            display: block;
            margin-bottom: 8px;
        }

        input {
            width: 100%;
```

```
            padding: 10px;
            margin-bottom: 15px;
            border: 1px solid #ccc;
            border-radius: 4px;
            box-sizing: border-box;
        }

        input[type="submit"] {
            background-color: #4caf50;
            color: white;
            cursor: pointer;
        }

        input[type="submit"]:hover {
            background-color: #45a049;
        }

        div.error {
            color: red;
            margin-bottom: 15px;
        }

        form.login-form {
            margin-top: 20px;
        }
    </style>
</head>
<body>
    <form action="/register" method="post">
    <h2>ลงทะเบียน</h2>

    <!-- แสดงข้อความผิดพลาด (ถ้ามี) -->
    <c:if test="${not empty error}">
        <div class="error">
            ${error}
        </div>
    </c:if>

    <label for="firstname">First Name:</label>
<input type="text" id="firstname" name="firstname" required>

<label for="lastname">Last Name:</label>
<input type="text" id="lastname" name="lastname" required>


    <label for="username">Username:</label>
    <input type="text" id="username" name="username" required>

    <label for="password">Password:</label>
    <input type="password" id="password" name="password" required>

    <input type="submit" value="Register">

    <div class="register-link">
        <p>มีบัญชีแล้วใช่หรือไม่? <a href="/login">เข้าสู่ระบบ</a></p>
    </div>
```

```
</form>

</body>
</html>
```

//dashboard




```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
<head>
    <title>หน้า Home</title>
    <style type="text/css">
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 0;
            display: flex;
            flex-direction: column;
            align-items: center;
        }

        h1 {
            color: #333;
        }

        form {
            display: flex;
            margin-bottom: 20px;
        }
```

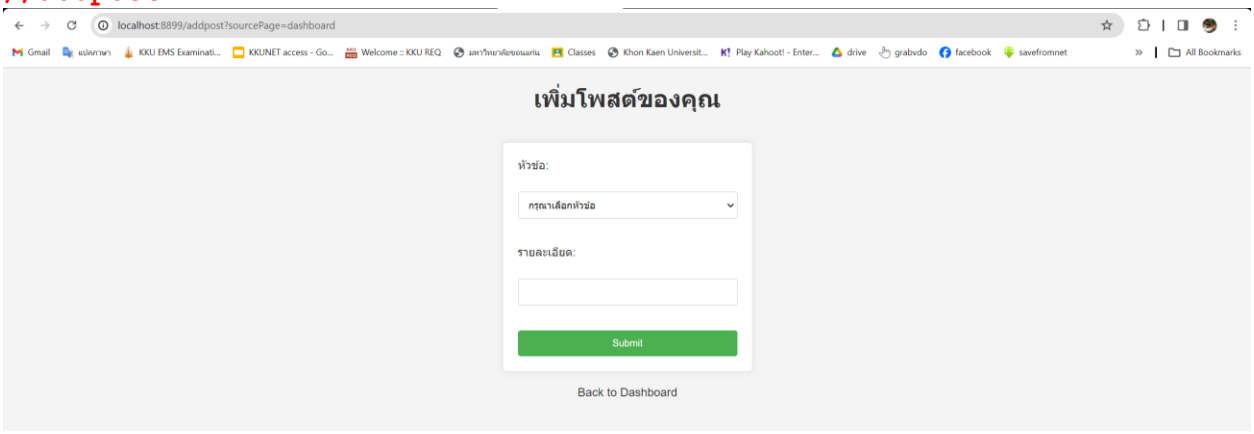
```css
label {
    margin-right: 10px;
}

select, input[type="text"] {
    padding: 8px;
    border: 1px solid #ccc;
    border-radius: 4px;
    margin-right: 10px;
}

button {
    background-color: #4caf50;
    color: white;
    padding: 8px;
    border: none;
    border-radius: 4px;
    cursor: pointer;
}

button:hover {
    background-color: #45a049;
}

.box {
    border: 1px solid #ccc;
    padding: 10px;
    margin-bottom: 10px;
    width: 70%;
    margin: 10px auto;
    background-color: #fff;
}

.post-header {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-bottom: 10px; /* เพิ่มขึ้นมาเพื่อให้มีระยะห่างระหว่าง header กับข้อความ */
}

a.edit-link, a.delete-link {
    text-decoration: none;
    color: #333;
}

.like-link {
    color: #333;
    text-decoration: none;
    cursor: pointer;
}

.like-link:hover {
    text-decoration: underline;
}

a.add-post-link {
```

```
            text-decoration: none;
            color: #fff;
            background-color: #4caf50;
            padding: 8px;
            border-radius: 4px;
            margin-top: 10px;
            display: inline-block;
        }

        p.no-posts {
            text-align: center;
            font-style: italic;
        }

        a.logout-link {
            background-color: red;
            padding: 8px;
            border-radius: 4px;
            margin-top: 10px;
            color: white;
            display: inline-block;
            text-decoration: none;
        }

    </style>
</head>
<body>

<h1>Blogนอย</h1>

<!-- ในส่วนที่แสดง Hello, ${loggedInUsername}! -->
<p>Hello, <a
href="${pageContext.request.contextPath}/profile">${loggedInUsername}</a>!</p
>


<form action="${pageContext.request.contextPath}/search" method="GET">
    <label for="keyword">ค้นหา:</label>
    <input type="text" id="keyword" name="keyword" value="${keyword}"/>
    <label for="category">เลือกหมวดหมู่:</label>
    <select id="category" name="category">
        <option value="" disabled selected>กรุณาเลือกหมวดหมู่</option>
        <option value="บ่น">บ่น</option>
        <option value="นอย">นอย</option>
          <option value="การเรียน">การเรียน</option>
        <option value="การทำงาน">การทำงาน</option>
        <option value="ครอบครัว">ครอบครัว</option>
    </select>
    <button type="submit">ค้นหา</button>
</form>

<a class="add-post-link"
href="${pageContext.request.contextPath}/addpost?sourcePage=dashboard">เพิ่มโพสต์ของ
คุณ</a>
```

```
<c:if test="${not empty posts}">
    <c:forEach var="post" items="${posts}">
        <div class="box">
            <div>
       <c:if test="${post.author eq loggedInUsername}">
    <a class="edit-link"
href="${pageContext.request.contextPath}/editpost/${post.id}">Edit</a>
    <a class="delete-link" href="javascript:void(0);"
onclick="confirmDelete(${post.id})">Delete</a>
</c:if>
                <h2>${post.author}</h2>
                <p>หมวดหมู่: ${post.category}</p>
                <p>${post.detail}</p>
                <c:url var="likeUrl"
value="${pageContext.request.contextPath}/like/${post.id}" />

                <p>จำนวนคนกดไลค์: ${post.love} คน</p>
                   <p>วันที่โพสต์: ${post.timestamp}</p>
                <a class="like-link" id="likeButton-${post.id}"
href="${likeUrl}" onclick="likePost(${post.id},
'${post.likedUsernames.contains(loggedInUsername)}')">Love</a>
            </div>
             <div class="comments">
    <h3>ความคิดเห็น:</h3>
    <c:forEach var="comment" items="${post.comments}">
        <p>${comment.author}: ${comment.content}</p>
    </c:forEach>

    <!-- Form for adding a comment -->
    <form action="${pageContext.request.contextPath}/addComment"
method="post">
        <input type="hidden" name="postId" value="${post.id}" />
        <label for="comment-${post.id}">เพิ่มความคิดเห็น:</label>
        <input type="text" id="comment-${post.id}" name="comment" required/>
        <button type="submit">ส่ง</button>
    </form>
</div>
        </div>

    </c:forEach>
</c:if>

<c:if test="${empty posts}">
    <p class="no-posts">ไม่พบโพสต์ที่ค้นหา</p>
</c:if>

<a class="logout-link" href="${pageContext.request.contextPath}/logout">ออกจาก
ระบบ</a>
<script>
    function likePost(postId, isLiked) {
        var likeButton = document.getElementById('likeButton-' + postId);

        if (isLiked) {
            // ถ้าไลค์แล้ว, ทำการ Unlike
```

```
                likeButton.style.color = '#333'; // เปลี่ยนสีเป็นค่าเริ่มต้น
            } else {
                // ถ้ายังไม่ไลค์, กดไลค์และเปลี่ยนสี
                likeButton.style.color = 'red'; // เปลี่ยนสีเป็นสีแดง
            }
        }
    </script>
    <script>
        function confirmDelete(postId) {
            var confirmDelete = confirm("คุณต้องการลบโพสต์นี้ใช่หรือไม่?");
            if (confirmDelete) {
                // กระทำการลบโพสต์ (สามารถเปลี่ยนไปยัง URL หรือการทำ AJAX request ได้)
                window.location.href =
"${pageContext.request.contextPath}/delete/" + postId;
            }
        }
    </script>

    </body>
    </html>
```

//addpost

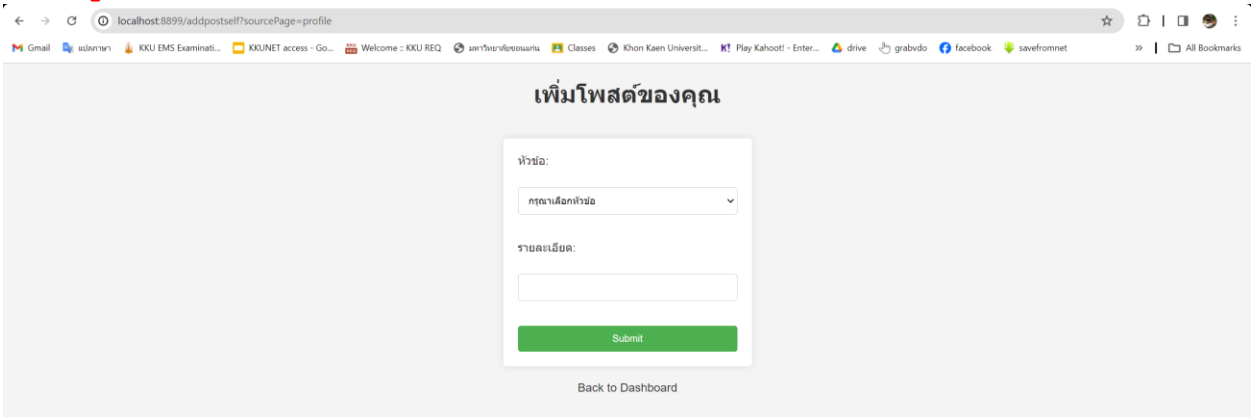



```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <title>เพิ่มโพสต์</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 0;
            display: flex;
            flex-direction: column;
            align-items: center;
        }

        h1 {
            color: #333;
```

```
        }

        form {
            background-color: #fff;
            padding: 20px;
            border-radius: 5px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            width: 300px;
            margin: 20px auto;
            display: flex;
            flex-direction: column;
        }

        label {
            margin-bottom: 8px;
            color: #333;
        }

        select, input[type="text"] {
            padding: 10px;
            margin-bottom: 15px;
            border: 1px solid #ccc;
            border-radius: 4px;
            box-sizing: border-box;
        }

        input[type="submit"] {
            background-color: #4caf50;
            color: white;
            padding: 10px;
            border: none;
            border-radius: 4px;
            cursor: pointer;
        }

        input[type="submit"]:hover {
            background-color: #45a049;
        }

        .error-message {
            color: red;
            font-size:14px;
            margin-top: 1px;
        }
           a {
            text-decoration: none;
            color: #333;
        }
    </style>
</head>
<body>

<h1>เพิ่มโพสต์ของคุณ</h1>

<form action="/addpost?sourcePage=dashboard" method="post" accept-
charset="UTF-8">
```

```
    <label for="category">หัวข้อ:</label><br>
    <select id="category" name="category">
        <option value="" disabled selected>กรุณาเลือกหัวข้อ</option>
        <option value="บ่น">บ่น</option>
        <option value="นอย">นอย</option>
        <option value="การเรียน">การเรียน</option>
        <option value="การทำงาน">การทำงาน</option>
        <option value="ครอบครัว">ครอบครัว</option>
    </select>
      <span class="error-message" id="category-error"></span><br>
    <label for="detail">รายละเอียด:</label><br>
    <input type="text" id="detail" name="detail"><br>
    <input type="submit" value="Submit">
</form>
    <!-- เพิ่มส่วนนี้เพื่อแสดงข้อความ error -->
    <c:if test="${not empty error}">
        <div style="color: red;">
            ${error}
        </div>
    </c:if>

    <a href="/dashboard">Back to Dashboard</a>

<!-- ใช้ JavaScript เพื่อแสดงข้อความ error ถ้าหลังจาก Submit แล้วไม่ได้เลือกหัวข้อ -->
<script>
    document.querySelector('form').addEventListener('submit', function(event)
{
        var category = document.getElementById('category');
        var categoryError = document.getElementById('category-error');
        if (category.value === '') {
            categoryError.textContent = 'กรุณาเลือกหัวข้อ';
            event.preventDefault();
        } else {
            categoryError.textContent = ''; // ลบข้อความ error ถ้ามีการเลือกหัวข้อ
        }
    });
</script>
</body>
</html>
```

//addpostself

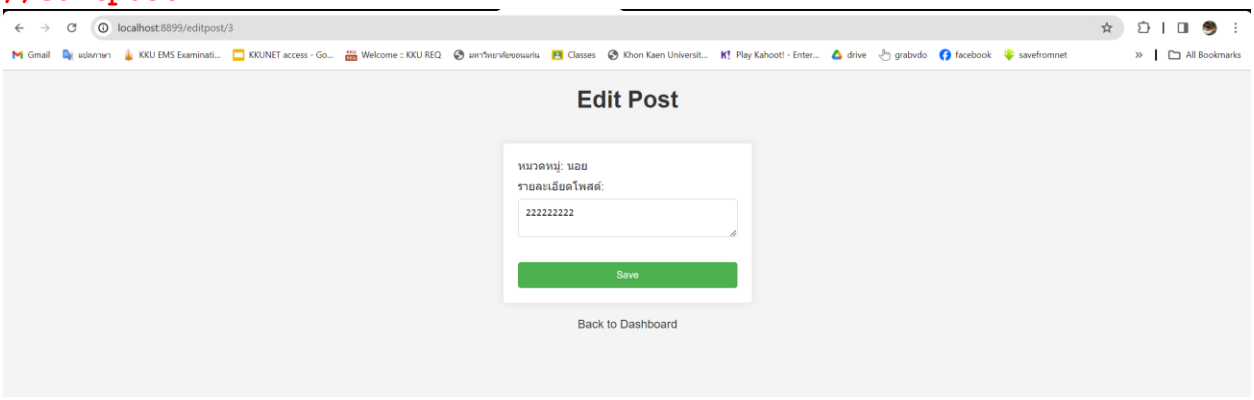


```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <title>เพิ่มโพสต์</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 0;
            display: flex;
            flex-direction: column;
            align-items: center;
        }

        h1 {
            color: #333;
        }

        form {
            background-color: #fff;
            padding: 20px;
            border-radius: 5px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            width: 300px;
            margin: 20px auto;
            display: flex;
            flex-direction: column;
        }

        label {
            margin-bottom: 8px;
            color: #333;
        }

        select, input[type="text"] {
            padding: 10px;
            margin-bottom: 15px;
            border: 1px solid #ccc;
            border-radius: 4px;
            box-sizing: border-box;
        }

        input[type="submit"] {
            background-color: #4caf50;
            color: white;
            padding: 10px;
            border: none;
            border-radius: 4px;
            cursor: pointer;
        }

        input[type="submit"]:hover {
            background-color: #45a049;
        }
```

```
        .error-message {
            color: red;
            font-size:14px;
            margin-top: 1px;
        }
           a {
            text-decoration: none;
            color: #333;
        }
    </style>
</head>
<body>

<h1>เพิ่มโพสต์ของคุณ</h1>

<form action="/addpostself?sourcePage=profile" method="post" accept-
charset="UTF-8">

    <label for="category">หัวข้อ:</label><br>
    <select id="category" name="category">
        <option value="" disabled selected>กรุณาเลือกหัวข้อ</option>
        <option value="บ่น">บ่น</option>
        <option value="นอย">นอย</option>
        <option value="การเรียน">การเรียน</option>
        <option value="การทำงาน">การทำงาน</option>
        <option value="ครอบครัว">ครอบครัว</option>
    </select>
      <span class="error-message" id="category-error"></span><br>
    <label for="detail">รายละเอียด:</label><br>
    <input type="text" id="detail" name="detail"><br>
    <input type="submit" value="Submit">
</form>
    <!-- เพิ่มส่วนนี้เพื่อแสดงข้อความ error -->
    <c:if test="${not empty error}">
        <div style="color: red;">
            ${error}
        </div>
    </c:if>

    <a href="/dashboard">Back to Dashboard</a>

<!-- ใช้ JavaScript เพื่อแสดงข้อความ error ถ้าหลังจาก Submit แล้วไม่ได้เลือกหัวข้อ -->
<script>
    document.querySelector('form').addEventListener('submit', function(event)
{
        var category = document.getElementById('category');
        var categoryError = document.getElementById('category-error');
        if (category.value === '') {
            categoryError.textContent = 'กรุณาเลือกหัวข้อ';
            event.preventDefault();
        } else {
            categoryError.textContent = ''; // ลบข้อความ error ถ้ามีการเลือกหัวข้อ
        }
```

```
    });
</script>
</body>
</html>
```

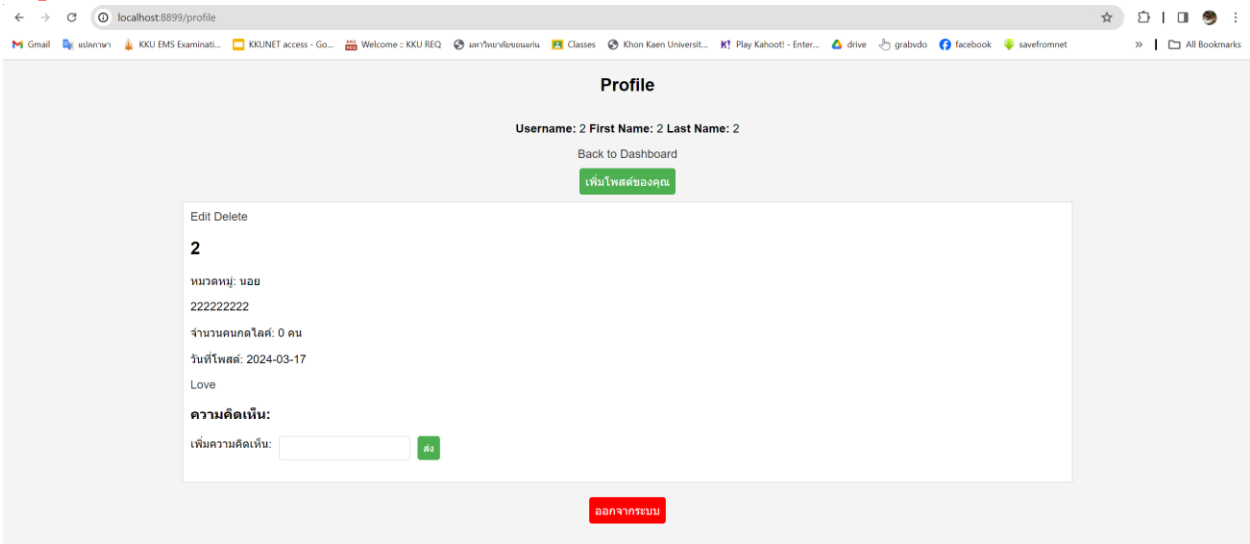
//editpost




```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <title>Edit Post</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 0;
            display: flex;
            flex-direction: column;
            align-items: center;
        }

        h1 {
            color: #333;
        }

        form {
            background-color: #fff;
            padding: 20px;
            border-radius: 5px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            width: 300px;
            margin: 20px auto;
            display: flex;
            flex-direction: column;
        }

        label {
            margin-bottom: 8px;
            color: #333;
        }

        textarea {
```

```
            padding: 10px;
            margin-bottom: 15px;
            border: 1px solid #ccc;
            border-radius: 4px;
            box-sizing: border-box;
            resize: vertical; /* ทำให้สามารถปรับขนาดตามแนวดิ่งได้ */
        }

        input[type="submit"] {
            background-color: #4caf50;
            color: white;
            padding: 10px;
            border: none;
            border-radius: 4px;
            cursor: pointer;
        }

        input[type="submit"]:hover {
            background-color: #45a049;
        }

        a {
            text-decoration: none;
            color: #333;
        }

        a:hover {
            text-decoration: underline;
        }
    </style>
</head>
<body>

<h1>Edit Post</h1>

<form method="post" action="/editpost/${post.id}">
<label for="category">หมวดหมู่: ${post.category}</label>

    <label for="detail">รายละเอียดโพสต์:</label>
    <textarea id="detail" name="detail">${post.detail}</textarea>
    <br>
    <input type="submit" value="Save">
</form>

<a href="/dashboard">Back to Dashboard</a>

</body>
</html>
```

//profile




```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Profile</title>
        <style type="text/css">
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 0;
            display: flex;
            flex-direction: column;
            align-items: center;
        }

        h1 {
            color: #333;
        }

        form {
            display: flex;
            margin-bottom: 20px;
        }

        label {
            margin-right: 10px;
        }

        select, input[type="text"] {
            padding: 8px;
            border: 1px solid #ccc;
            border-radius: 4px;
            margin-right: 10px;
        }
```

```css
button {
    background-color: #4caf50;
    color: white;
    padding: 8px;
    border: none;
    border-radius: 4px;
    cursor: pointer;
}

button:hover {
    background-color: #45a049;
}

.box {
    border: 1px solid #ccc;
    padding: 10px;
    margin-bottom: 10px;
    width: 70%;
    margin: 10px auto;
    background-color: #fff;
}

.post-header {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-bottom: 10px; /* เพิ่มขึ้นมาเพื่อให้มีระยะห่างระหว่าง header กับข้อความ */
}

a.edit-link, a.delete-link {
    text-decoration: none;
    color: #333;
}

.like-link {
    color: #333;
    text-decoration: none;
    cursor: pointer;
}

.like-link:hover {
    text-decoration: underline;
}

a.add-post-link {
    text-decoration: none;
    color: #fff;
    background-color: #4caf50;
    padding: 8px;
    border-radius: 4px;
    margin-top: 10px;
    display: inline-block;
}

p.no-posts {
    text-align: center;
```

```
            font-style: italic;
        }

        a.logout-link {
            background-color: red;
            padding: 8px;
            border-radius: 4px;
            margin-top: 10px;
            color: white;
            display: inline-block;
            text-decoration: none;
        }
         a {
            text-decoration: none;
            color: #333;
        }
    </style>
</head>
<body>
    <h2>Profile</h2>
   <p><b>Username:</b> ${loggedInUsername}  <b>First Name:</b>
${sessionScope.loggedInFirstname} <b>Last Name:</b>
${sessionScope.loggedInLastname}</p>

   <a href="/dashboard">Back to Dashboard</a>

  <!--<form action="${pageContext.request.contextPath}/search" method="GET">
    <label for="keyword">ค้นหา:</label>
    <input type="text" id="keyword" name="keyword" value="${keyword}"/>
    <label for="category">เลือกหมวดหมู่:</label>
    <select id="category" name="category">
        <option value="" disabled selected>กรุณาเลือกหมวดหมู่</option>
        <option value="บ่น">บ่น</option>
        <option value="นอย">นอย</option>
          <option value="การเรียน">การเรียน</option>
        <option value="การทำงาน">การทำงาน</option>
        <option value="ครอบครัว">ครอบครัว</option>
    </select>
    <button type="submit">ค้นหา</button>
</form>-->

<a class="add-post-link"
href="${pageContext.request.contextPath}/addpostself?sourcePage=profile">เพิ่มโพสต์
ของคุณ</a>

<c:if test="${not empty posts}">
    <c:forEach var="post" items="${posts}">
        <div class="box">
            <div>
       <c:if test="${post.author eq loggedInUsername}">
    <a class="edit-link"
href="${pageContext.request.contextPath}/editpost/${post.id}">Edit</a>
    <a class="delete-link" href="javascript:void(0);"
onclick="confirmDelete(${post.id})">Delete</a>
</c:if>
```

```
                <h2>${post.author}</h2>
                <p>หมวดหมู่: ${post.category}</p>
                <p>${post.detail}</p>
               <!--<c:url var="likeUrl"
value="${pageContext.request.contextPath}/like/${post.id}" />  -->

                <p>จำนวนคนกดไลค์: ${post.love} คน</p>
                    <p>วันที่โพสต์: ${post.timestamp}</p>
                <a class="like-link" id="likeButton-${post.id}"
href="${likeUrl}" onclick="likePost(${post.id},
'${post.likedUsernames.contains(loggedInUsername)}')">Love</a>
            </div>
             <div class="comments">
    <h3>ความคิดเห็น:</h3>
    <c:forEach var="comment" items="${post.comments}">
        <p>${comment.author}: ${comment.content}</p>
    </c:forEach>

    <!-- Form for adding a comment -->
    <form action="${pageContext.request.contextPath}/addComment"
method="post">
        <input type="hidden" name="postId" value="${post.id}" />
        <label for="comment-${post.id}">เพิ่มความคิดเห็น:</label>
        <input type="text" id="comment-${post.id}" name="comment" required/>
        <button type="submit">ส่ง</button>
    </form>
</div>
        </div>

    </c:forEach>
</c:if>

<!--<c:if test="${empty posts}">
    <p class="no-posts">ไม่พบโพสต์ที่ค้นหา</p>
</c:if>-->

<a class="logout-link" href="${pageContext.request.contextPath}/logout">ออกจาก
ระบบ</a>

<script>
    function confirmDelete(postId) {
        var confirmDelete = confirm("คุณต้องการลบโพสต์นี้ใช่หรือไม่?");
        if (confirmDelete) {
            // กระทำการลบโพสต์ (สามารถเปลี่ยนไปยัง URL หรือการทำ AJAX request ได้)
            window.location.href =
"${pageContext.request.contextPath}/delete/" + postId;
        }
    }
</script>

</body>
</html>
```