Họ tên: Đỗ Thành Đạt

Mã sinh viên: 2021602433

Nhóm: 4

Câu 2:

Quản lý user:

```
public interface IUserService {  2 usages  1 implementation  ThanhCao18 +1
    UserResponse createUser(UserCreateRequest userCreateRequest);  1 usage  1 implementation  ThanhCao18
    UserResponse updateUser(int id, UserUpdateRequest userUpdateRequest);  1 usage  1 implementation  ThanhCao18
    String deleteUser(int id);  1 usage  1 implementation  ThanhCao18
    List<UserResponse> getAllUsers();  1 usage  1 implementation  ThanhCao18
    UserResponse getUserbyId(int id);  1 usage  1 implementation  Datne03
}
```

Quản lý Review:

```
public interface IReviewService {  2 usages  1 implementation  ThanhCao18 *
    ReviewResponse addReviewToProduct(int user_id, int product_id, ReviewDTO review);  1 usage
    List<Review> getAllReviews();  1 usage  1 implementation  new *
    Optional<ReviewResponse> getReviewById(int id);  no usages  1 implementation  new *
    void deleteReviewById(int id);  1 usage  1 implementation  new *
    ReviewResponse updateReviewById(int id, ReviewDTO review);  1 usage  1 implementation  new *
}
```

Quản lý Message:

```
public interface IMessageService {  2 usages  1 implementation  Datne03 +1
    List<MessageDTO> getAllMessages();  1 usage  1 implementation  Datne03
    MessageDTO sendMessage(MessageDTO MessageDTO);  1 usage  1 implementation  Datne03
    MessageDTO updateMessage(int id, MessageDTO MessageDTO);  1 usage  1 implementation  Datne03
    void deleteMessage(int id);  1 usage  1 implementation  Datne03
}
```

Các hàm xử lý chính:

1. User:

1.1. Login

```
2.  String createToken(User user) {
        JWSHeader header = new JWSHeader(JWSAlgorithm.HS256);
        JWTClaimsSet jwtClaimsSet = new JWTClaimsSet.Builder()
                .subject(user.getEmail())
                .issuer("Ananas")
                .issueTime(new Date())
                .expirationTime(new Date(
                        Instant.now().plus(100, ChronoUnit.HOURS).toEpochMilli()
                ))
```

```
                .claim("scope", buildScopeToRoles(user))
                .build();
        Payload payload = new Payload(jwtClaimsSet.toJSONObject());
        JWSObject jwsObject = new JWSObject(header, payload);

        try {
            jwsObject.sign(new MACSigner(KEY_SIGN.getBytes()));
            return jwsObject.serialize();
        } catch (JOSEException e) {
            log.error("cant create token", e);
            throw new RuntimeException();
        }
    }

    public AuthenticationResponse authenticationResponse(AuthenticationRequest authenticationRequest) {
        var user = userRepository.findByUsername(authenticationRequest.getUsername())
                .orElseThrow(() -> new AppException((ErrException.USER_NOT_EXISTED)));
        boolean checked = passwordEncoder.matches(authenticationRequest.getPassword(), user.getPassword());
        if (!checked) {
            throw new AppException(ErrException.USER_NOT_EXISTED);
        }
        var token = createToken(user);
        return AuthenticationResponse.builder()
                .token(token)
                .check(true)
                .build();
    }
```

1.2. Đăng ký:

```
@Override 1 usage  Datne03
public UserResponse createUser(UserCreateRequest userCreateRequest) {
    if (userRepository.existsByUsername(userCreateRequest.getUsername())) {
        throw new AppException(ErrException.USER_EXISTED);
    }
    if (userRepository.existsByEmail(userCreateRequest.getEmail())) {
        throw new AppException(ErrException.EMAIL_EXISTED);
    }
    User user = userMapper.toUser(userCreateRequest);
    user.setPassword(passwordEncoder.encode(user.getPassword()));
    HashSet<String> roles = new HashSet<>(); // HashSet đảm bảo rằng mỗi vai trò của người dùng là duy nhất,tối ưu hóa các thao tác
    roles.add(Role.User.name()); // cho phep user them nguoidungmoi
    user.setRoles(roles);
    if (userCreateRequest.getEmail() != null && !userCreateRequest.getEmail().isEmpty()) {
        String subject = "Welcome to our service";
        String text = "Dear "+userCreateRequest.getUsername()+","+userCreateRequest.getEmail()+","+userCreateRequest.getPassword();
        emailService.sendMessage(userCreateRequest.getEmail(), subject, text);
    }
    return userMapper.toUserResponse(userRepository.save(user));
}
```

1.3. Sửa

```
@Override 1 usage  Datne03
public UserResponse updateUser(int id, UserUpdateRequest userUpdateRequest) {
    User user = userRepository.findById(id)
            .orElseThrow(() -> new AppException(ErrException.USER_NOT_EXISTED));
    userMapper.updateUser(user, userUpdateRequest);
    user.setPassword(passwordEncoder.encode(user.getPassword()));
    return userMapper.toUserResponse(userRepository.save(user));
}
```

1.4. Xóa

```java
@PostAuthorize("hasRole('Admin')") 1 usage  Datne03
public String deleteUser(int id) {
    User userDelete = userRepository.findById(id)
            .orElseThrow(() -> new AppException(ErrException.USER_NOT_EXISTED));
    userRepository.delete(userDelete);
    return "xoa thanh cong";
}
```

1.5. Lấy user

```java
@Override 1 usage  Datne03
public List<UserResponse> getAllUsers() {
    return userRepository.findAll().stream() Stream<User>
            .map(userMapper::toUserResponse) Stream<UserResponse>
            .collect(Collectors.toList());
}

@Override 1 usage  Datne03
public UserResponse getUserbyId(int id) {
    User user = userRepository.findById(id)
            .orElseThrow(() -> new AppException(ErrException.USER_NOT_EXISTED));
    return userMapper.toUserResponse(user);
}
```

1.6. Quên mật khẩu:

```java
// Quên mật khẩu - Gửi qua email
public void forgotPassword(String email) {  1 usage  Datne03
    User user = userRepository.findByEmail(email)
            .orElseThrow(() -> new UsernameNotFoundException("User not found"));

    String subject = "Dear, "+user.getUsername()+", your account has been reset password successfully.";
    String resetPassword = generateRandomPassword( length: 6);
    emailService.sendMessage(email, subject , text: "Your new password: "+resetPassword);
    user.setPassword(passwordEncoder.encode(resetPassword));
    userRepository.save(user);
}

// Hàm để tạo chuỗi ngẫu nhiên dài `length` ký tự
private String generateRandomPassword(int length) {  1 usage  Datne03
    String allowedChars = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";
    Random random = new Random();
    StringBuilder password = new StringBuilder();

    for (int i = 0; i < length; i++) {
        int index = random.nextInt(allowedChars.length());
        password.append(allowedChars.charAt(index));
    }

    return password.toString();
}
```

1.7. Đổi mật khẩu

```java
public boolean changePassword(ChangePasswordRequest request) {  1 usage  Datne03
    User user = userRepository.findByUsername(request.getUsername())
            .orElseThrow(() -> new RuntimeException("User not found"));
    if (user != null && passwordEncoder.matches(request.getOldPassword(), user.getPassword())) {
        user.setPassword(passwordEncoder.encode(request.getNewPassword()));
        userRepository.save(user);
        return true;
    }
    return false;
}
```

2. Message

2.1. Xem danh sách Message

```java
@Override  1 usage  Datne03
public List<MessageDTO> getAllMessages() {
    List<Messages> messages = messageRepository.findAll();
    return messages.stream()  Stream<Messages>
            .map(messageMapper::toMessageDTO)  Stream<MessageDTO>
            .collect(Collectors.toList());
}
```

2.2. Gửi tin nhắn:

```java
@Override  1 usage  Datne03
public MessageDTO sendMessage(MessageDTO messageDto) {
    User sender = userRepository.findById(messageDto.getSenderId())
            .orElseThrow(() -> new RuntimeException("Sender not found"));
    User receiver = userRepository.findById(messageDto.getReceiverId())
            .orElseThrow(() -> new RuntimeException("Receiver not found"));
    Messages newMessage = messageMapper.toMessages(messageDto,sender,receiver);
    return messageMapper.toMessageDTO(messageRepository.save(newMessage));
}
```

2.3. Sửa xóa tin nhắn

```java
@Override  1 usage  Datne03
public MessageDTO updateMessage(int id, MessageDTO newMessage) {
    Messages messages = messageRepository.findById(id).orElse( other: null);
    messageMapper.updateMessageDTO(messages, newMessage);
    return messageMapper.toMessageDTO(messageRepository.save(messages));
}

@Override  1 usage  Datne03
public void deleteMessage(int id) {
    messageRepository.deleteById(id);
}
```

3. Review

3.1. Thêm review

```
public ReviewResponse addReviewToProduct(int user_id, int product_id, ReviewDTO review) {  1 usage  Datne03
    Review currentReview = new Review();
    if (userRepository.existsById(user_id)) {
        if (productRepository.existsById(product_id)) {
            currentReview.setRating(review.getRating());
            currentReview.setComment(review.getComment());
            currentReview.setProduct(productRepository.findById(product_id).get());
            currentReview.setUser(userRepository.findById(user_id).get());
        }
    }
    return reviewMapper.toReviewResponse(reviewRepository.save(currentReview));
}
```

3.2. Xem review

```
public List<Review> getAllReviews() {  1 usage  Datne03
    return reviewRepository.findAll().stream().toList();
}

public Optional<ReviewResponse> getReviewById(int id) {  no usages  Datne03
    return reviewRepository.findById(id).map(reviewMapper::toReviewResponse);
}
```

3.3. Xóa sửa review

```
public void deleteReviewById(int id) {  1 usage  Datne03
    reviewRepository.deleteById(id);
}

public ReviewResponse updateReviewById(int id, ReviewDTO review) {  1 usage  Datne03
    Optional<Review> optionalReview = reviewRepository.findById(id);
    if (optionalReview.isPresent()) {
        reviewMapper.updateReview(optionalReview.get(), review);
        review.setRating(review.getRating());
        review.setComment(review.getComment());
    }
    return reviewMapper.toReviewResponse(reviewRepository.save(optionalReview.get()));
}
```

4. Email

```java
public class EmailService {
    JavaMailSender mailSender;
    public void sendMessage(String to, String subject, String text) {  2 usages  Datne03
        SimpleMailMessage message = new SimpleMailMessage();
        message.setTo(to);
        message.setSubject(subject);
        message.setText(text);
        mailSender.send(message);
    }
}
```