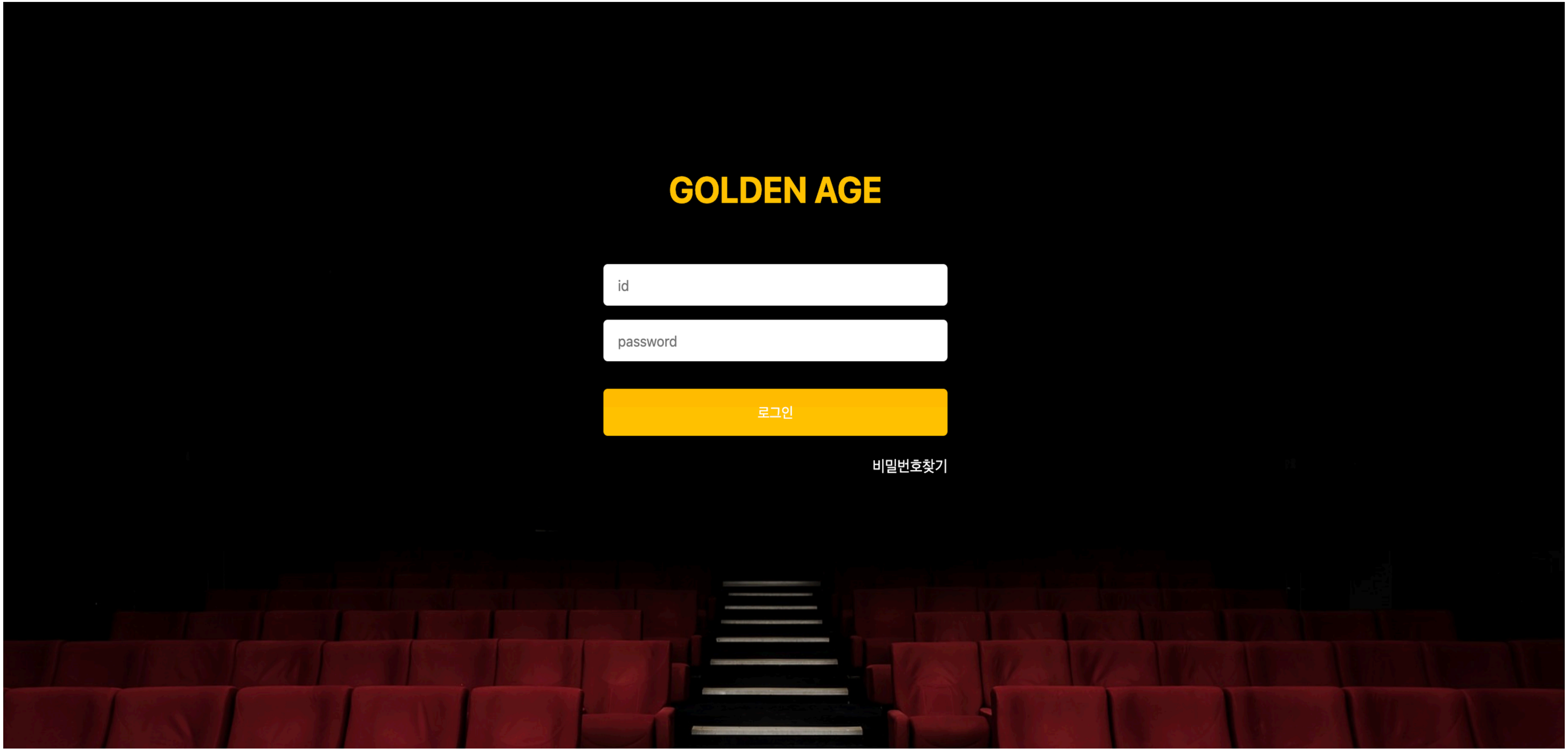
로그인



Spring-security conifg 설정

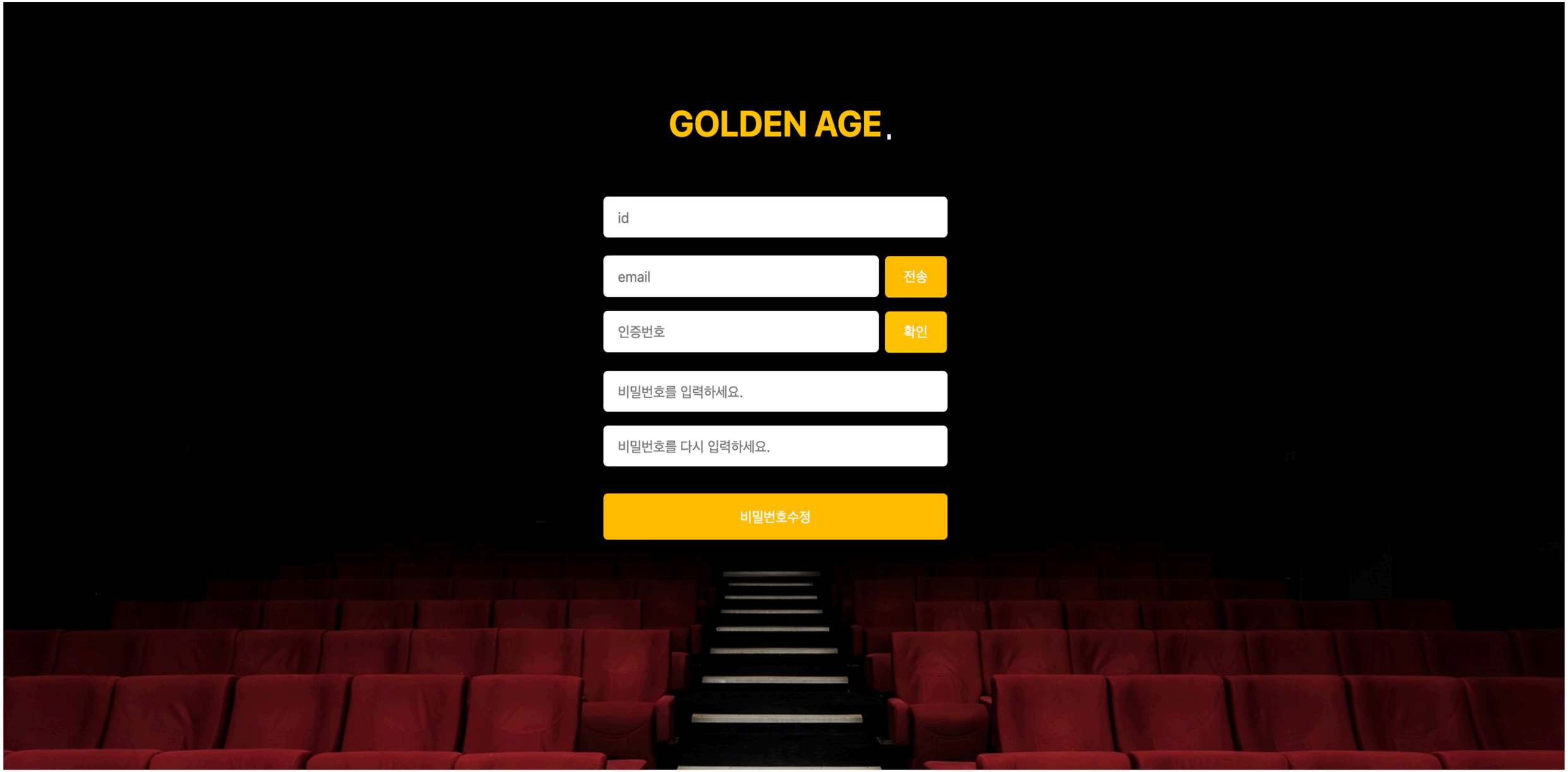
```java
@Configuration
@EnableWebSecurity
public class SpringSecurityConfig extends WebSecurityConfigurerAdapter {

    1 usage
    @Autowired
    AdminServiceImpl as;

    intectme12
    @Override
    protected void configure(AuthenticationManagerBuilder auth) throws Exception {

        auth.userDetailsService(as).passwordEncoder(passwordEncoder());
    }

    intectme12
    @Override
    public void configure(WebSecurity web){

        web.ignoring().antMatchers( ...antPatterns: "/css/**", "/js/**", "/images/**", "/lib/**");
    }
```

```java
    intectme12
    @Override
    protected void configure(HttpSecurity http) throws Exception {
        http.csrf().disable() HttpSecurity
                .authorizeRequests() ExpressionUrlAuthorizationConfigurer<...>.ExpressionInterceptUrlRegistry
//                  .antMatchers("/product/**").hasRole("USER")
                .antMatchers( ...antPatterns: "/admin/login").hasRole("ADMIN")
                .anyRequest().permitAll()
            .and() HttpSecurity
                .formLogin() FormLoginConfigurer<HttpSecurity>
                .loginPage("/admin/loginPage")
                .loginProcessingUrl("/login")
                .defaultSuccessUrl("/")
                .failureUrl( authenticationFailureUrl: "/admin/failure")
            .and() HttpSecurity
                .logout() LogoutConfigurer<HttpSecurity>
                .logoutUrl("/logout")
                .logoutSuccessUrl("/")
                .deleteCookies( ...cookieNamesToClear: "JSESSIONID")
                .invalidateHttpSession(true)
            .and() HttpSecurity
                .exceptionHandling() ExceptionHandlingConfigurer<HttpSecurity>
                .accessDeniedPage( accessDeniedUrl: "/admin/failure");
    }

    1 usage  intectme12
    @Bean
    public PasswordEncoder passwordEncoder() { return new BCryptPasswordEncoder(); }
}
```

ServiceImpl

```java
@Override
public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {

    AdminDTO adminDTO = adminMapper.selectAdminDTO(username);
    List <AuthDTO> authDTO = adminMapper.selectAuthDTO();

    if(adminDTO == null){
        throw new UsernameNotFoundException("admin not authorized.");
    }

    List<GrantedAuthority> authorities = new ArrayList<~>();

    if(authDTO != null){

        for(int i = 0; i < authDTO.size(); i++) {

            if(adminDTO.getAuthCode() == authDTO.get(i).getAuthCode()){

                authorities.add(new SimpleGrantedAuthority(authDTO.get(i).getAuthName()));
            }
        }
    }

    AdminDTOImpl admin = new AdminDTOImpl(adminDTO.getAdminId(), adminDTO.getAdminPwd(), authorities);
    admin.setDetails(adminDTO);
    System.out.println("admin : " +authorities);
    System.out.println("admin : " +admin);
    return admin;
}
```

Email 인증을 통한 비밀번호 수정



Mail conifg 설정

```
    @Bean
    public JavaMailSender javaMailService() {

        JavaMailSenderImpl javaMailSender = new JavaMailSenderImpl();

        javaMailSender.setHost("smtp.naver.com");
        javaMailSender.setUsername("[redacted]");
        javaMailSender.setPassword("[redacted]");

        javaMailSender.setPort(465);

        javaMailSender.setJavaMailProperties(getMailProperties());

        return javaMailSender;
    }

    1 usage   intectme12
    private Properties getMailProperties() {

        Properties properties = new Properties();
        properties.setProperty("mail.transport.protocol", "smtp");
        properties.setProperty("mail.smtp.auth", "true");
        properties.setProperty("mail.smtp.starttls.enable", "true");
        properties.setProperty("mail.debug", "true");
        properties.setProperty("mail.smtp.ssl.trust","smtp.naver.com");
        properties.setProperty("mail.smtp.ssl.enable","true");

        return properties;
    }
}
```

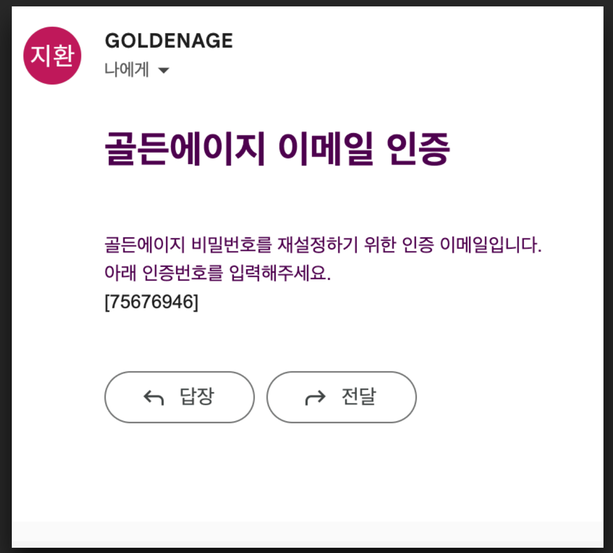
Controller

```java
@PostMapping("/email")
@ResponseBody
public Object matchIdEmail(@RequestBody HashMap<String, String> adminMap) throws MessagingException, UnsupportedEncodingException {

    System.out.println("adminMap : " + adminMap);

    String id = adminMap.get("id");
    String email = adminMap.get("email");

    AdminDTO ckEmail = emailService.selectEmail(id);
    System.out.println("ckEmail : " + ckEmail);

    int result = 0;

    if(ckEmail != null){

        String dbEmail = ckEmail.getAdminEmail();
        System.out.println("dbEmail : " + dbEmail);

        if(email.equals(dbEmail)) {

            //인증번호생성
            AuthKey authKey = new AuthKey();
            String authNum = authKey.makeAuthKey();
            System.out.println("authNum : " + authNum);

            //인증번호 업데이트(id, email 기준)
            result = emailService.updateAuthNum(id, dbEmail, authNum);
            System.out.println("result udateAuthNum : " + result);
```

```java
2 usages   intectme12 *
public class AuthKey {

    //인증번호 랜덤 생성
    1 usage   intectme12
    public String makeAuthKey(){

        Random ran = new Random();
        StringBuffer sb = new StringBuffer();

        do {

            int num = ran.nextInt( bound: 10);

            sb.append(num);

        } while(sb.length() < 8);

        return sb.toString();
    }
}
```

Controller - MailHandler

```
            //인증메일 발송
            if(result > 0) {

                MailHandler mailHandler = new MailHandler(mailSender);
                mailHandler.setSubject("골든에이지 인증코드 발송 이메일입니다.");
                mailHandler.setText(
                        "<h1>골든에이지 이메일 인증</h1>" +
                                "<br>골든에이지 비밀번호를 재설정하기 위한 인증 이메일입니다." +
                                "<br>아래 인증번호를 입력해주세요." +
                                "<br>[" + authNum + "]"
                );
                mailHandler.setFrom( email: "intectme@naver.com",  name: "GOLDENAGE");
                mailHandler.setTo(dbEmail);
                mailHandler.send();
            }

        } else if(!email.equals(dbEmail)){

            result = 0;
        }

    } else {

        result = 0;
    }

    System.out.println("result : " + result);

    Map<String, Object> resultAjax = new HashMap<>();
    resultAjax.put("result", result);
    System.out.println("result : " + result);

    Map<String, Object> resultAjax = new HashMap<>();
    resultAjax.put("result", result);

    return resultAjax;
}
```



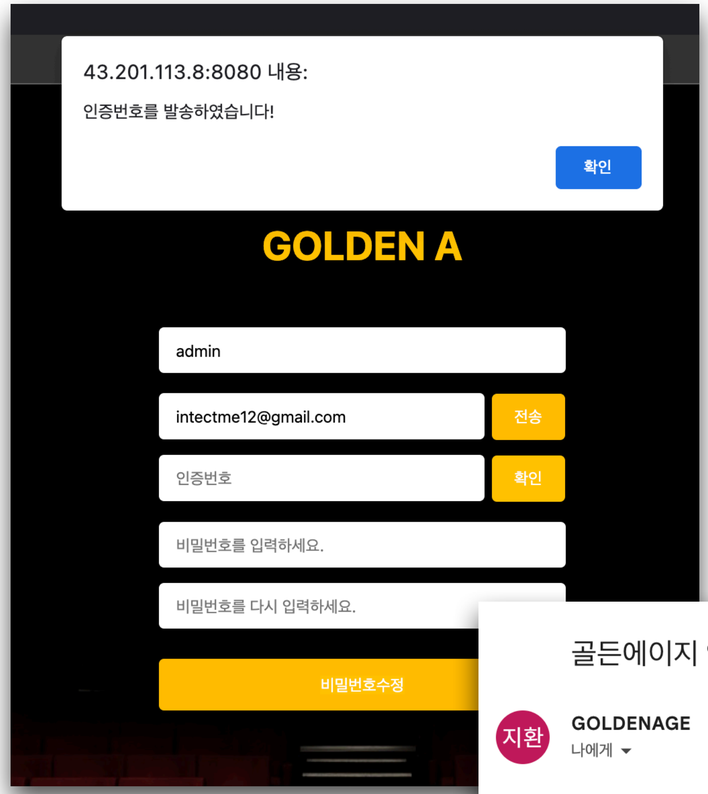

```
@Component
public class MailHandler {

    3 usages
    private JavaMailSender mailSender;
    3 usages
    private MimeMessage message;
    6 usages
    private MimeMessageHelper messageHelper;

    1 usage  intectme12
    @Autowired
    public MailHandler(JavaMailSender mailSender) throws MessagingException{

        this.mailSender = mailSender;
        message = this.mailSender.createMimeMessage();
        messageHelper =new MimeMessageHelper(message,  multipart: true,  encoding: "UTF-8");
    }

    // 제목
    1 usage  intectme12
    public void setSubject(String subject) throws  MessagingException{
        messageHelper.setSubject(subject);
    }

    // 내용
    intectme12
    public void setText(String htmlContent) throws MessagingException{
        messageHelper.setText(htmlContent,  html: true);
    }

    // 발송자
    1 usage  intectme12
    public void setFrom(String email, String name) throws UnsupportedEncodingException, MessagingException{
        messageHelper.setFrom(email, name);
    }

    // 수신자
    1 usage  intectme12
    public void setTo(String email) throws MessagingException{
        messageHelper.setTo(email);
    }

    intectme12
    public void addInline(String contentId, DataSource dataSource) throws MessagingException{
        messageHelper.addInline(contentId, dataSource);
    }

    // 보내기
    1 usage  intectme12
    public void send() { mailSender.send(message); }

}
```

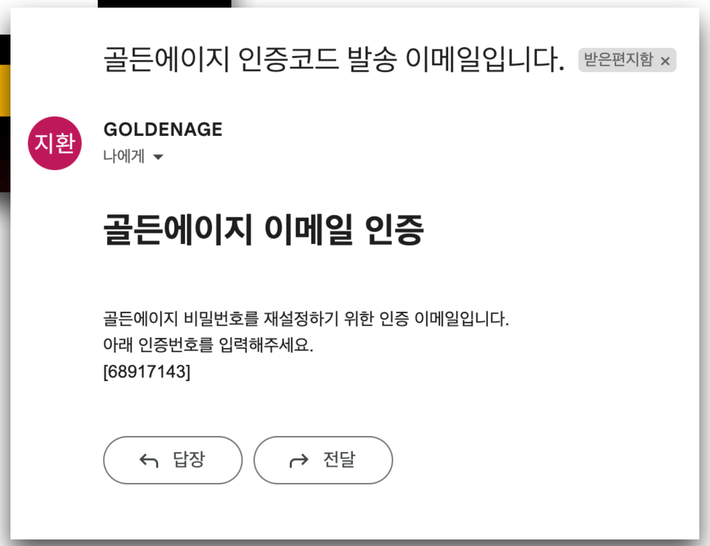
Controller

```
intectme12 *                                                                    8
@PostMapping("/authNum")
@ResponseBody
public Object matchAuthNum(@RequestBody HashMap<String, String> adminMap){

    System.out.println("adminMap" + adminMap);

    String id = adminMap.get("id");
    String email = adminMap.get("email");
    String authNum = adminMap.get("authNum");

    AdminDTO adminDTO = emailService.matchAuthNum(id, email);

    int result = 0;

    if(adminDTO != null){

        String dbAuthNum = adminDTO.getAuthNum();

        if(authNum.equals(dbAuthNum)){

            result = 1;

        } else {

            result = 2;
        }
    }

    System.out.println("result : " + result);

    Map<String, Object> resultAjax = new HashMap<>();
    resultAjax.put("result", result);

    return resultAjax;
}
```

Id, Email 확인 AJAX

```
<script>
    $("#email-submit").click('on', function (){

        let id = document.getElementById("id").value;
        let email = document.getElementById("email").value;

        console.log(id);
        console.log(email);

        let sendData = { 'id' : id, 'email' : email};
        console.log(sendData);
        console.log(JSON.stringify(sendData));

        $.ajax({

            url:'/submit/email',
            type: 'post',
            data: JSON.stringify(sendData),
            contentType : 'application/json',
            dataType : 'json',
            async : false,
            success: function (data){

                JSON.stringify(data);
                let result = data.result:
                    alert("인증번호를 발송하였습니다!")
                    $("#authNum").removeAttr("readonly");
                } else {

                    alert("아이디 또는 이메일이 잘못되었습니다.")
                }
            },
            error: function (data){

            }
        })
    })
```





인증번호 확인 AJAX

```
$("#auth-submit").click('on', function (){

    let id = document.getElementById("id").value;
    let email = document.getElementById("email").value;
    let authNum = document.getElementById("authNum").value;

    console.log(id);
    console.log(email);
    console.log(authNum);

    let sendData = { 'id' : id, 'email' : email, 'authNum' : authNum };

    console.log(sendData);
    console.log(JSON.stringify(sendData));

    $.ajax({

        url:'/submit/authNum',
        type: 'post',
        data: JSON.stringify(sendData),
        contentType : 'application/json',
        dataType : 'json',
        async : false,
        success: function (data){
            JSON.stringify(data);
            let result = data.result;

            if(result == 1){

                alert("인증번호가 확인 되었습니다.")
                $("#id").attr('readonly', true);
                $("#email").attr('readonly', true);
                $("#email-submit").attr('readonly', true);
                $("#auth-submit").attr('readonly', true);
                $("#authNum").attr('readonly', true);
                $("#password-one").removeAttr('readonly');
                $("#password-two").removeAttr('readonly');
            } else if(result == 2){

                alert("인증번호가 틀렸습니다.")
            } else {

                alert("아이디 또는 이메일이 잘못되었습니다.")
            }
        },
        error: function (data){

        }
    })
})
```