

1-4. (React, Vue)

: / :

Page No. 150011909

Launch Release No. 7.3.500.20250722 / Latest Release No.

: (React, Vue) SSO '1-0.'

```
> AUD7
> Spring Boot() AUD7 , React Vue() UI
> , block
  (top , parent postMessage AUD )
```

```
> AP PORTAL ( ) AUD7 PORTAL Domain name .
> POTAL AP Cookie bimatrix_ap_accessToken .
```

1.

1-1) application.yml (Vue, React)

```
>
: {}/src/main/resources/application.yml
```

```

1  ∨ spring:
2  ∨   thymeleaf:
3      prefix: classpath:/templates/
4      check-template-location: true
5      suffix: .html
6      mode: HTML
7      cache: false # default true, 개발 시에는 false로 두는 것이 좋음
8
9  ∨ server:
10     port: 9995
11     #ssl:
12     # key-store: classpath:keystore.p12
13     # key-store-type: PKCS12
14     # key-store-password: 123456
15
16     # servlet:
17     #   session:
18     #     cookie:
19     #       http-only: true
20     #       secure: false
21
22  ∨ aud:
23     server-url: http://aud-local.bimatrix.com:8087/matrix
24     x-ap-update-addr: aud-local.bimatrix.com
25     x-aud-ap-id: matrix
26     x-aud-ap-secret-key: C558A12512744C80A3795354427B47C8
27     ssh-key-path: D:\\document\\업무\\15. claude 활용\\spring-embedded\\private_key.p
28

```

- < 1-1. application.yml >
- port:
 - server-url: AUD7
 - x-ap-update-addr: AUD7
 - x-aud-ap-id: '1-0. '
 - x-aud-ap-secret-key: '1-0. ' Secret Key
 - ssh-key-path: '1-0. ' private_key.pem

1-2)

> SHARE_DOMAIN AUD7 .
: {}/frontend/src/main.[jsx|js] (React jsx, Vue js)

```

1  import './assets/imatrix_header.css'
2
3  import { createRoot } from 'react-dom/client'
4  import DivViewer from './DivViewer.jsx'
5  import IFrameViewer from './IFrameViewer.jsx'
6
7  // -----
8  // 공통 초기화: 전역 변수 등록 & 토큰 쿠키 설정
9  // (mode에 관계없이 React 앱 로드 전에 선행 실행)
10 // -----
11 const audServerUrl = window.IIT_DATA.audServerUrl
12 const SHARE_DOMAIN = 'bimatrix.com'
13 const AUD7_PATH = audServerUrl + '/AUD/500'
14
15 window.AUD7_PATH = AUD7_PATH
16 window.AUD7_SETTING_PATH = AUD7_PATH
17 window.SHARE_DOMAIN = SHARE_DOMAIN
18 window.gvWebRootName = audServerUrl
19
20 // 인증 토큰 쿠키 등록
21 const exdate = new Date()
22 exdate.setDate(exdate.getDate() + 1)
23 document.cookie = `bimatrix_ap_accessToken=${window.IIT_DATA.token}; expires=${exdate.toUTCString()}; path=/; domain=${
24
25 // -----
26 // mode에 따라 컴포넌트 분기
27 //   div   → App.jsx
28 //   iframe → IFrameViewer.jsx
29 // -----
30 const mode = window.IIT_DATA.mode || 'div'
31 const Root = mode === 'iframe' ? IFrameViewer : DivViewer
32
33 // 주의: React 18 StrictMode는 useEffect를 개발 모드에서 두 번 실행시키므로,
34 // AUD 외부 스크립트가 중복 로드되는 부작용이 있어 본 임베디드 시나리오에서는 사용하지 않는다.
35 createRoot(document.getElementById('root')).render(<Root />)
36

```

< 1-2. main.jsx >

- SHARE_DOMAIN: AUD7 .
 ex) AUD7 : aud.example.com, : embedded.example.com
 SHARE_DOMAIN = 'example.com'

1-3)

> .
 : {}/frontend/src/{DivViewer.[jsx|vue]} or {IFrameViewer.[jsx|vue]} (React jsx, Vue js) ((div / iframe) jsx)

```

1  import { useEffect, useRef } from 'react'
2  import './layout.css'
3
4  // IIT_DATA 추출
5  const audServerUrl = window.IIT_DATA.audServerUrl
6
7  // 보고서 목록 (데모용 - 실제 보고서 코드로 교체)
8  const reportList = [
9    { name: 'i-AUD 보고서 호출', code: 'REP4CE0125E54214D8396F788D30F335583', module: 'SD', isShow: true },
10   { name: 'i-AUD 보고서 호출2', code: 'REP236AB97070714FA3AC22F9DFD00AFFF3', module: 'SD', isShow: false },
11 ]
12
13 // 컴포넌트 외부 모듈 스코프에 두는 가변 상태
14 // (외부 AUD JS 가 직접 DOM 을 조작하므로 React 리렌더링이 필요 없어 useState 대신 plain object 사용)
15 const state = {
16   reportInfo: null,
17   rName: '',
18   folderCode: '',
19   description: '',
20   moduleCode: '',
21   TemplateCode: '',
22   reportId: '',
23 }
24
25 // React 18 StrictMode 미사용이지만, 혹시 모를 중복 마운트에 대비한 가드
26 let initialized = false
27
28 function DivViewer() {
29   const errorRef = useRef({ show: false, message: '' })
30
31   useEffect(() => {
32     if (initialized) return
33     initialized = true
34
35     try {
36       loadExternalResources(audServerUrl)
37       winResizer()

```

< 1-3. DivViewer.jsx >

- name:
- code: AUD7
- module:
- isShow:

1-4)

> .(Vue, React)
: {}/src/main/java/com/aud/embedded/AudAction.java

```

22 public class AudAction {
23     private final String audSecretKey;
24
25     public AudAction(@Qualifier("audWebClient") WebClient audWebClient
26         , @Value("${aud.x-aud-ap-id}") String audApId
27         , @Value("${aud.ssh-key-path}") String audApSecretFilePath
28         , @Value("${aud.x-ap-update-addr}") String apUpdateAddr
29         , @Value("${aud.x-aud-ap-secret-key}") String audSecretKey) {
30         this.audWebClient = audWebClient;
31         this.audApId = audApId;
32         this.audApSecretFilePath = audApSecretFilePath;
33         this.apUpdateAddr = apUpdateAddr;
34         this.audSecretKey = audSecretKey;
35     }
36
37     public String getAccessToken() throws Exception {
38         AtomicReference<String> apToken = new AtomicReference<>(initialValue: "");
39         // Application 공통 계정으로 토큰 발급하여 공통으로 사용할 경우는 로그인 유저의 코드를 따로 전달하지 않
40         // 해당 부분은 실제 인증을 받고 나서 외부 포탈로 로그인한 사용자를 기준으로 토큰 발급을 원할 경우에 처
41         // 로그인 유저는 백엔드에서 세션에서 확인 후 각 사이트별로 관리하는 방법으로 확인합니다.
42         String loginUserCode = "matrix";
43
44         try {
45             // aud7 플랫폼에서 발급받은 secret key를 ssh의 private key로 서명하여 전달한다.
46             PrivateKey privateKey = loadPrivateKey(audApSecretFilePath);
47             // aud7 secret key 서명 생성
48             String signedMessage = signMessage(audSecretKey, privateKey);
49
50             ResponseEntity<String> res =
51                 audWebClient.post()
52                     .uri("/api/auth/sign/ap/token")
53                     .headers(headers -> {
54                         headers.set("X-AUD-AP-Id", audApId);
55                         headers.set("X-AUD-AP-Secret-SSH", signedMessage);
56                         headers.set("X-AP-UPDATE-ADDR", apUpdateAddr);
57                         headers.set("X-AUD-USER", loginUserCode);
58                     })
59                     .body(apToken);
60         }
61     }
62 }

```

< 1-4. AudAction.java >

```

- loginUserCode:
:   loginUserCode .
:   loginUserCode .
( AUD7 . )

```

2.

```

1) {}/frontend npm run build .
2) {}/frontend/dist/assets/ Spring .
   -frontend/dist/assets/index.js src/main/resources/static/assets/index.js
   -frontend/dist/assets/index.css -> src/main/resources/static/assets/index.css

```

3.

```

> IntelliJ IDE SpringThymeleafTestApplication .
: {}/src/main/java/com/aud/embedded/SpringThymeleafTestApplication.java

```

4.

```

> URL .
div : http://{port}/aud/sitePortalAUD7EmAudDemo/div
iframe : http://{port}/aud/sitePortalAUD7EmAudDemo/iframe

```

i-AUD 보고서 호출

i-AUD 보고서 호출2

STM_310

	FieldName	Employees1	Employees2	Employees3
		id	id	id
		first_name	(Null)	first_name
		last_name	last_name	(Null)
		email	(Null)	email
		phone_number	phone_number	(Null)
		hire_date	hire_date	hire_date
		job_id	(Null)	(Null)
		salary	salary	salary
		department_id	(Null)	(Null)
		manager_id	(Null)	(Null)



AUD AP API

- AUD AP API AUD Application Application JWT .

-
- 1
- . (key = bimatrix_ap_accessToken)

URL		Header		
{ A U D } / a p i / a u t h / s i g n / a p / t o k e n	POST	API Request Header		String
			AUD SSH Private.pem Secret Key Header Secret Key AUD . JWT AUD	

application.yml (Vue, React)

```
spring:
  thymeleaf:
    prefix: classpath:/templates/
    check-template-location: true
    suffix: .html
    mode: HTML
    cache: false # default true,    false

server:
  port: 9995
  #ssl:
  #  key-store: classpath:keystore.p12
  #  key-store-type: PKCS12
  #  key-store-password: 123456

#  servlet:
#    session:
#      cookie:
#        http-only: true
#        secure: false

aud:
  server-url: {AUD7  }
  x-ap-update-addr: {AUD7  }
  x-aud-ap-id: {'1-0. '  }
  x-aud-ap-secret-key: {'1-0. '  Secret Key }
  ssh-key-path: {'1-0. '  private_key.pem  }
```


main.jsx (React)

```
import './assets/imatrix_header.css'

import { createRoot } from 'react-dom/client'
import DivViewer from './DivViewer.jsx'
import IframeViewer from './IframeViewer.jsx'

//
// : &
// (mode React )
//
const audServerUrl = window.IIT_DATA.audServerUrl
const SHARE_DOMAIN = { AUD7 }
const AUD7_PATH = audServerUrl + '/AUD/500'

window.AUD7_PATH = AUD7_PATH
window.AUD7_SETTING_PATH = AUD7_PATH
window.SHARE_DOMAIN = SHARE_DOMAIN
window.gvWebRootName = audServerUrl

//
const exdate = new Date()
exdate.setDate(exdate.getDate() + 1)
document.cookie = `bimatrix_ap_accessToken=${window.IIT_DATA.token}; expires=${exdate.toUTCString()};
path=/; domain=${SHARE_DOMAIN}`

//
// mode
// div App.jsx
// iframe IframeViewer.jsx
//
const mode = window.IIT_DATA.mode || 'div'
const Root = mode === 'iframe' ? IframeViewer : DivViewer

// : React 18 StrictMode useEffect ,
// AUD .
createRoot(document.getElementById('root')).render(<Root />)
```

main.js (Vue)

```
import './assets/imatrix_header.css'

import { createApp } from 'vue'
import DivViewer from './DivViewer.vue'
import IFrameViewer from './IFrameViewer.vue'

//
// : &
// (mode Vue )
//
const audServerUrl = window.IIT_DATA.audServerUrl
const SHARE_DOMAIN = 'bimatrix.com'
const AUD7_PATH = audServerUrl + '/AUD/500'

window.AUD7_PATH = AUD7_PATH
window.AUD7_SETTING_PATH = AUD7_PATH
window.SHARE_DOMAIN = SHARE_DOMAIN
window.gvWebRootName = audServerUrl

//
const exdate = new Date()
exdate.setDate(exdate.getDate() + 1)
document.cookie = `bimatrix_ap_accessToken=${window.IIT_DATA.token}; expires=${exdate.toUTCString()};
path=/; domain=${SHARE_DOMAIN}`

//
// mode
// div DivViewer.vue
// iframe IFrameViewer.vue
//
const mode = window.IIT_DATA.mode || 'div'
const Root = mode === 'iframe' ? IFrameViewer : DivViewer

createApp(Root).mount('#app')
```

DivViewer.jsx (React)

```
import { useEffect, useRef } from 'react'
import './layout.css'

// IIT_DATA
const audServerUrl = window.IIT_DATA.audServerUrl

// ( - )
const reportList = [
  { name: { }, code: {AUD7 }, module: { }, isShow: { } },
  { name: 'i-AUD 2', code: 'REP236AB97070714FA3AC22F9DFD00AFF3', module: 'SD', isShow: false },
]

//
// ( AUD JS DOM React useState plain object )
const state = {
  reportInfo: null,
  rName: '',
  folderCode: '',
  description: '',
  moduleCode: '',
  TemplateCode: '',
  reportId: '',
}

// React 18 StrictMode ,
let initialized = false
```

```

function DivViewer() {
  const errorRef = useRef({ show: false, message: '' })

  useEffect(() => {
    if (initialized) return
    initialized = true

    try {
      loadExternalResources(audServerUrl)
      winResizer()
      if (window.GFN_AUTHORITY && window.GFN_AUTHORITY.UpdateSession) {
        window.GFN_AUTHORITY.UpdateSession()
      }

      const handleResize = () => {
        winResizer()
        if (window.AUD && window.AUD.GetMainViewer) {
          const mainViewer = window.AUD.GetMainViewer()
          if (mainViewer && mainViewer.ViewerSizeChanged) mainViewer.ViewerSizeChanged()
        }
      }
      window.addEventListener('resize', handleResize)
      return () => window.removeEventListener('resize', handleResize)
    } catch (e) {
      handleError(' ', e)
    }
  }, [])

  function handleError(message, error) {
    errorRef.current = { show: true, message: `${message} (${error.message || error})` }
    console.error(message, error)
  }

  return (
    <>
      <div className="top_panel"></div>
      <div className="left_panel">
        <ul>
          {reportList.map(report => (
            <li key={report.code}>
              <div
                className="rep_div"
                onClick={() => openReport(report.code, report.isShow, report.module)}
              >
                {report.name}
              </div>
            </li>
          ))}
        </ul>
      </div>
      <div className="main_group VisibleFrame">
        <div className="titlebg" id="titlebg_main" style={{ display: 'none' }}>
          <div className="title_area">
            <ul>
              <li><span id="dvReportName"></span></li>
            </ul>
          </div>
          <div className="bookmark" id="bookmarkIcon" style={{ display: 'none' }}></div>
          <div className="location" style={{ display: 'none' }}></div>
          <div className="topbtn_group"></div>
        </div>
        <div id="AUDview" className="istudio-common-viewer"></div>
      </div>
      <div className="foot_panel"></div>
    </>
  )
}

//
// (AUD JS/CSS)
//

```

```

function loadExternalResources(serverUrl) {
  const head = document.head
  const scripts = [
    '/AUD/500/js/lib/audframework/debug/bimatrix.lib.audframework.js',
    '/AUD/500/js/lib/audframework/debug/bimatrix.module.audframework.js',
    '/AUD/500/js/lib/rsa/jsbn.js',
    '/AUD/500/js/lib/rsa/prng4.js',
    '/AUD/500/js/lib/rsa/rng.js',
    '/AUD/500/js/lib/rsa/rsa.js',
    '/portal/js/jquery-3.6.0.min.js',
    '/portal/js/Base64.js',
    '/portal/js/jquery.portal.common.js',
    '/portal/js/jquery.cookie.js',
    '/portal/js/authorityCheck_em.jsp',
    '/portal/js/portal.message.jsp',
    '/portal/js/portal.option.data.jsp',
    '/portal/js/portal.content.top.js',
    '/portal/js/matrix.script.comm.js',
    '/portal/js/matrix.script.content.em.js',
    '/portal/js/portal.content.bookmark.js',
    '/portal/js/portal.content.condition.js',
    '/extention/AUD/customscript.jsp',
    '/extention/portal/customscript.jsp',
  ]
  const styles = [
    '/AUD/500/theme/skin-default/ko/css/bimatrix.module.audframework.css',
    '/AUD/500/theme/skin-default/ko/css/ion.rangeSlider.css',
    '/extention/AUD/bimatrix.custom.audframework.css',
  ]

  styles.forEach(path => {
    const link = document.createElement('link')
    link.rel = 'stylesheet'
    link.href = serverUrl + path
    head.appendChild(link)
  })
  scripts.forEach(path => {
    const script = document.createElement('script')
    script.src = serverUrl + path
    script.async = false
    // GFN_OPTION portal CSS
    if (path.endsWith('portal.option.data.jsp')) {
      script.onload = () => loadPortalThemeCss(serverUrl)
    }
    head.appendChild(script)
  })
}

function loadPortalThemeCss(serverUrl) {
  const opt = window.GFN_OPTION
  if (!opt || !opt.PORTAL_THEME_CSS_PATH) return
  const root = opt.WEB_ROOTNAME || ''
  const baseUrl = (root && serverUrl.endsWith(root))
    ? serverUrl.slice(0, -root.length)
    : serverUrl
  const link = document.createElement('link')
  link.rel = 'stylesheet'
  link.href = `${baseUrl}${opt.PORTAL_THEME_CSS_PATH}/theme.css`
  document.head.appendChild(link)
}

//
//
//
function openReport(code, isTitle, module) {
  state.reportId = code
  state.moduleCode = module

  const PARAM_ARR = [{ KEY: 'VS_TEST', VALUE: 'VS_TEST1_VAL' }]
  if (window.AUD) window.AUD.SetCustomParams(PARAM_ARR)
}

```

```

    if (window.GFN_AUTHORITY && window.GFN_AUTHORITY.searchReportInfo) {
        state.reportInfo = window.GFN_AUTHORITY.searchReportInfo(code)
        state.rName = state.reportInfo.name
        state.folderCode = state.reportInfo.option.FolderCode
        state.description = state.reportInfo.desc
    }

    if (window.GFN_AUTHORITY && window.GFN_AUTHORITY.USER_AUTH_INFO) window.GFN_AUTHORITY.USER_AUTH_INFO()

    if (isTitle) {
        const btnType = (window.GFN_OPTION && window.GFN_OPTION.OP04_VIEW_BTN === 'TEXT') ? 'text_type' :
'img_type'

        if (window.$ && typeof window.$('.topbtn_group').option_top === 'function') {
            window.$('.topbtn_group').option_top('view_btn', {
                btn_type: btnType,
                callbackFn: settingTitle,
                embedded: true,
            })
        }
    } else {
        document.querySelector('.titlebg').style.display = 'none'
        winResizer()
    }

    if (module === 'SD') {
        if (window.AUD && window.AUD.Init) window.AUD.Init(AudOpenReport)
    } else if (module === 'SX') {
        window.AUD.ShellModuleCode = 'SX'
        if (window.AUD && window.AUD.Init) window.AUD.Init(MetaOpenReport)
    }
}

function AudOpenReport() {
    if (window.AUD && window.AUD.SetFileDialogCallback) window.AUD.SetFileDialogCallback()
    if (window.AUD) window.AUD.LoadDocument('AUDview', state.reportId, 2)
}

function MetaOpenReport() {
    const metaInfo = {
        META_CODE: state.reportId,
        NAME: state.rName,
        DESC: state.description,
        PARENT: state.folderCode,
        TYPE: state.moduleCode,
    }
    window.AUD.MetaViewManager.IsMetaFileView = true
    window.AUD.mViewerId = 'AUDview'
    window.AUD.LoadMetaDocument('AUDview', state.reportId, metaInfo)
}

//
// /   (option_top )
//
function settingTitle() {
    document.querySelector('.titlebg').style.display = ''
    const info = state.reportInfo
    if (!info) return
    if (window.setReportInfo) window.setReportInfo(
        info.code,
        info.name,
        info.desc,
        info.module,
        info.path,
        '',
        { FolderCode: info.option.FolderCode }
    )
    if (window.menuVisible) window.menuVisible(
        info.code,
        info.module,
        { AuthNo: info.option.AuthNo },

```

```

        false
    )
    const buttons = ['btnEdit', 'btnSaveAs', 'btnExport']
    buttons.forEach(id => {
        const el = document.getElementById(id)
        if (el) el.style.display = 'none'
    })
    settingReportInfo(info.name)
    winResizer()
}

function settingReportInfo(name) {
    const el = document.querySelector('.titlebg #dvReportName')
    if (!el) return
    el.innerText = name
    el.classList.add('title_bullet')
    const opt = window.GFN_OPTION || {}
    const root = opt.WEB_ROOTNAME || ''
    const baseUrl = (root && audServerUrl.endsWith(root))
        ? audServerUrl.slice(0, -root.length)
        : audServerUrl
    const imgUrl = `${baseUrl}${opt.PORTAL_THEME_IMG_PATH}/tree/tree_iaud.png`
    el.style.background = `url('${imgUrl}') 0px 0px no-repeat`
}

//
//
//
function winResizer() {
    const win_w = window.innerWidth
    const win_h = window.innerHeight

    const topPanel = document.querySelector('.top_panel')
    const leftPanel = document.querySelector('.left_panel')
    const footPanel = document.querySelector('.foot_panel')
    const titleBg = document.querySelector('.titlebg')
    const viewPanel = document.querySelector('.view_panel')
    const audView = document.getElementById('AUDview')

    const top_panel_height = (topPanel && topPanel.offsetHeight) || 0
    const left_panel_width = (leftPanel && leftPanel.offsetWidth) || 0
    const foot_panel_height = (footPanel && footPanel.offsetHeight) || 0

    if (leftPanel) {
        leftPanel.style.height = `${win_h - top_panel_height}px`
        leftPanel.style.top = `${top_panel_height}px`
    }

    const mainGroup = document.querySelector('.main_group')
    if (mainGroup) {
        mainGroup.style.height = `${win_h - top_panel_height - foot_panel_height}px`
        mainGroup.style.top = `${top_panel_height}px`
        mainGroup.style.left = `${left_panel_width}px`
        mainGroup.style.width = `${win_w - left_panel_width}px`
    }

    let title_panel_height = (titleBg && titleBg.offsetHeight) || 0
    const isHidden = titleBg && getComputedStyle(titleBg).display === 'none'
    if (isHidden) {
        title_panel_height = 0
    } else {
        settingReportTitleWidth()
    }

    if (viewPanel) {
        viewPanel.style.height = `${win_h}px`
        viewPanel.style.width = `${win_w}px`
    }

    if (audView) {
        audView.style.top = `${title_panel_height}px`
    }
}

```

```

        audView.style.height = `${win_h - top_panel_height - foot_panel_height - title_panel_height}px`
    }
}

function settingReportTitleWidth() {
    const titleEl = document.querySelector('#titlebg_main')
    const topArea = titleEl ? titleEl.querySelector('.topbtn_area') : null
    const bullet = titleEl ? titleEl.querySelector('.title_bullet') : null

    if (!titleEl || !topArea || !bullet) return

    const titlebg_w = titleEl.offsetWidth
    const title_topbtn_w = topArea.offsetWidth
    const title_reportName_w = titlebg_w - title_topbtn_w

    bullet.style.width = `${title_reportName_w}px`
}

export default DivViewer

```

DivViewer.vue (Vue)

```

<script setup>
import { onMounted, onBeforeUnmount, ref } from 'vue'
import './layout.css'

// IIT_DATA
const audServerUrl = window.IIT_DATA.audServerUrl

// ( - )
const reportList = [
    { name: 'i-AUD ', code: 'REP4CE0125E54214D8396F788D30F335583', module: 'SD', isShow: true },
    { name: 'i-AUD 2', code: 'REP236AB97070714FA3AC22F9DFD00AFFF3', module: 'SD', isShow: false },
]

// AUD JS DOM Vue
const state = {
    reportInfo: null,
    rName: '',
    folderCode: '',
    description: '',
    moduleCode: '',
    TemplateCode: '',
    reportId: '',
}

const errorMessage = ref('')
const showError = ref(false)

let handleResize = null

onMounted(() => {
    try {
        loadExternalResources(audServerUrl)
        winResizer()
        if (window.GFN_AUTHORITY && window.GFN_AUTHORITY.UpdateSession) {
            window.GFN_AUTHORITY.UpdateSession()
        }

        handleResize = () => {
            winResizer()
            if (window.AUD && window.AUD.GetMainViewer) {
                const mainViewer = window.AUD.GetMainViewer()
                if (mainViewer && mainViewer.ViewerSizeChanged) mainViewer.ViewerSizeChanged()
            }
        }
        window.addEventListener('resize', handleResize)
    } catch (e) {

```

```

        handleError('  .', e)
    }
})

onBeforeUnmount(() => {
    if (handleResize) window.removeEventListener('resize', handleResize)
})

function handleError(message, error) {
    showError.value = true
    errorMessage.value = `${message} (${error.message || error})`
    console.error(message, error)
}

//
// (AUD JS/CSS)
//
function loadExternalResources(serverUrl) {
    const head = document.head
    const scripts = [
        '/AUD/500/js/lib/audframework/debug/bimatrix.lib.audframework.js',
        '/AUD/500/js/lib/audframework/debug/bimatrix.module.audframework.js',
        '/AUD/500/js/lib/rsa/jsbn.js',
        '/AUD/500/js/lib/rsa/prng4.js',
        '/AUD/500/js/lib/rsa/rng.js',
        '/AUD/500/js/lib/rsa/rsa.js',
        '/portal/js/jquery-3.6.0.min.js',
        '/portal/js/Base64.js',
        '/portal/js/jquery.portal.common.js',
        '/portal/js/jquery.cookie.js',
        '/portal/js/authorityCheck_em.jsp',
        '/portal/js/portal.message.jsp',
        '/portal/js/portal.option.data.jsp',
        '/portal/js/portal.content.top.js',
        '/portal/js/matrix.script.comm.js',
        '/portal/js/matrix.script.content.em.js',
        '/portal/js/portal.content.bookmark.js',
        '/portal/js/portal.content.condition.js',
        '/extention/AUD/customscript.jsp',
        '/extention/portal/customscript.jsp',
    ]
    const styles = [
        '/AUD/500/theme/skin-default/ko/css/bimatrix.module.audframework.css',
        '/AUD/500/theme/skin-default/ko/css/ion.rangeSlider.css',
        '/extention/AUD/bimatrix.custom.audframework.css',
    ]

    styles.forEach(path => {
        const link = document.createElement('link')
        link.rel = 'stylesheet'
        link.href = serverUrl + path
        head.appendChild(link)
    })
    scripts.forEach(path => {
        const script = document.createElement('script')
        script.src = serverUrl + path
        script.async = false
        // GFN_OPTION portal CSS
        if (path.endsWith('portal.option.data.jsp')) {
            script.onload = () => loadPortalThemeCss(serverUrl)
        }
        head.appendChild(script)
    })
}

function loadPortalThemeCss(serverUrl) {
    const opt = window.GFN_OPTION
    if (!opt || !opt.PORTAL_THEME_CSS_PATH) return
    const root = opt.WEB_ROOTNAME || ''
    const baseUrl = (root && serverUrl.endsWith(root))
        ? serverUrl.slice(0, -root.length)

```



```

        : serverUrl
const link = document.createElement('link')
link.rel = 'stylesheet'
link.href = `${baseUrl}${opt.PORTAL_THEME_CSS_PATH}/theme.css`
document.head.appendChild(link)
}

//
//
//
function openReport(code, isTitle, module) {
    state.reportId = code
    state.moduleCode = module

    const PARAM_ARR = [{ KEY: 'VS_TEST', VALUE: 'VS_TEST1_VAL' }]
    if (window.AUD) window.AUD.SetCustomParams(PARAM_ARR)

    if (window.GFN_AUTHORITY && window.GFN_AUTHORITY.searchReportInfo) {
        state.reportInfo = window.GFN_AUTHORITY.searchReportInfo(code)
        state.rName = state.reportInfo.name
        state.folderCode = state.reportInfo.option.FolderCode
        state.description = state.reportInfo.desc
    }

    if (window.GFN_AUTHORITY && window.GFN_AUTHORITY.USER_AUTH_INFO) window.GFN_AUTHORITY.USER_AUTH_INFO()

    if (isTitle) {
        const btnType = (window.GFN_OPTION && window.GFN_OPTION.OP04_VIEW_BTN === 'TEXT') ? 'text_type' :
'img_type'

        if (window.$ && typeof window.$('.topbtn_group').option_top === 'function') {
            window.$('.topbtn_group').option_top('view_btn', {
                btn_type: btnType,
                callbackFn: settingTitle,
                embedded: true,
            })
        }
    } else {
        document.querySelector('.titlebg').style.display = 'none'
        winResizer()
    }

    if (module === 'SD') {
        if (window.AUD && window.AUD.Init) window.AUD.Init(AudOpenReport)
    } else if (module === 'SX') {
        window.AUD.ShellModuleCode = 'SX'
        if (window.AUD && window.AUD.Init) window.AUD.Init(MetaOpenReport)
    }
}

function AudOpenReport() {
    if (window.AUD && window.AUD.SetFileDialogCallback) window.AUD.SetFileDialogCallback()
    if (window.AUD) window.AUD.LoadDocument('AUDview', state.reportId, 2)
}

function MetaOpenReport() {
    const metaInfo = {
        META_CODE: state.reportId,
        NAME: state.rName,
        DESC: state.description,
        PARENT: state.folderCode,
        TYPE: state.moduleCode,
    }
    window.AUD.MetaViewManager.IsMetaFileView = true
    window.AUD.mViewerId = 'AUDview'
    window.AUD.LoadMetaDocument('AUDview', state.reportId, metaInfo)
}

//
// /    (option_top )
//

```

```

function settingTitle() {
    document.querySelector('.titlebg').style.display = ''
    const info = state.reportInfo
    if (!info) return
    if (window.setReportInfo) window.setReportInfo(
        info.code,
        info.name,
        info.desc,
        info.module,
        info.path,
        '',
        { FolderCode: info.option.FolderCode }
    )
    if (window.menuVisible) window.menuVisible(
        info.code,
        info.module,
        { AuthNo: info.option.AuthNo },
        false
    )
    const buttons = ['btnEdit', 'btnSaveAs', 'btnExport']
    buttons.forEach(id => {
        const el = document.getElementById(id)
        if (el) el.style.display = 'none'
    })
    settingReportInfo(info.name)
    winResizer()
}

function settingReportInfo(name) {
    const el = document.querySelector('.titlebg #dvReportName')
    if (!el) return
    el.innerText = name
    el.classList.add('title_bullet')
    const opt = window.GFN_OPTION || {}
    const root = opt.WEB_ROOTNAME || ''
    const baseUrl = (root && audServerUrl.endsWith(root))
        ? audServerUrl.slice(0, -root.length)
        : audServerUrl
    const imgUrl = `${baseUrl}${opt.PORTAL_THEME_IMG_PATH}/tree/tree_iaud.png`
    el.style.background = `url('${imgUrl}') 0px 0px no-repeat`
}

//
//
//
function winResizer() {
    const win_w = window.innerWidth
    const win_h = window.innerHeight

    const topPanel = document.querySelector('.top_panel')
    const leftPanel = document.querySelector('.left_panel')
    const footPanel = document.querySelector('.foot_panel')
    const titleBg = document.querySelector('.titlebg')
    const viewPanel = document.querySelector('.view_panel')
    const audView = document.getElementById('AUDview')

    const top_panel_height = (topPanel && topPanel.offsetHeight) || 0
    const left_panel_width = (leftPanel && leftPanel.offsetWidth) || 0
    const foot_panel_height = (footPanel && footPanel.offsetHeight) || 0

    if (leftPanel) {
        leftPanel.style.height = `${win_h - top_panel_height}px`
        leftPanel.style.top = `${top_panel_height}px`
    }

    const mainGroup = document.querySelector('.main_group')
    if (mainGroup) {
        mainGroup.style.height = `${win_h - top_panel_height - foot_panel_height}px`
        mainGroup.style.top = `${top_panel_height}px`
        mainGroup.style.left = `${left_panel_width}px`
        mainGroup.style.width = `${win_w - left_panel_width}px`
    }
}

```

```

    }

    let title_panel_height = (titleBg && titleBg.offsetHeight) || 0
    const isHidden = titleBg && getComputedStyle(titleBg).display === 'none'
    if (isHidden) {
        title_panel_height = 0
    } else {
        settingReportTitleWidth()
    }

    if (viewPanel) {
        viewPanel.style.height = `${win_h}px`
        viewPanel.style.width = `${win_w}px`
    }

    if (audView) {
        audView.style.top = `${title_panel_height}px`
        audView.style.height = `${win_h - top_panel_height - foot_panel_height - title_panel_height}px`
    }
}

function settingReportTitleWidth() {
    const titleEl = document.querySelector('#titlebg_main')
    const topArea = titleEl ? titleEl.querySelector('.topbtn_area') : null
    const bullet = titleEl ? titleEl.querySelector('.title_bullet') : null

    if (!titleEl || !topArea || !bullet) return

    const titlebg_w = titleEl.offsetWidth
    const title_topbtn_w = topArea.offsetWidth
    const title_reportName_w = titlebg_w - title_topbtn_w

    bullet.style.width = `${title_reportName_w}px`
}
</script>

<template>
    <div class="top_panel"></div>
    <div class="left_panel">
        <ul>
            <li v-for="report in reportList" :key="report.code">
                <div
                    class="rep_div"
                    @click="openReport(report.code, report.isShow, report.module)"
                >
                    {{ report.name }}
                </div>
            </li>
        </ul>
    </div>
    <div class="main_group VisibleFrame">
        <div class="titlebg" id="titlebg_main" style="display: none;">
            <div class="title_area">
                <ul>
                    <li><span id="dvReportName"></span></li>
                </ul>
            </div>
            <div class="bookmark" id="bookmarkIcon" style="display: none;"></div>
            <div class="location" style="display: none;"></div>
            <div class="topbtn_group"></div>
        </div>
        <div id="AUDview" class="istudio-common-viewer"></div>
    </div>
    <div class="foot_panel"></div>
</template>

```

IframeViewer.jsx (React)

```
import { useEffect, useRef } from 'react'
```

```

import './layout.css'

const audServerUrl = window.IIT_DATA.audServerUrl
const webRoot = window.IIT_DATA.webRoot

// ( - )
const reportList = [
  { name: { }, code: {AUD7 }, module: { }, isShow: { } },
  { name: 'i-AUD 2', code: 'REP236AB97070714FA3AC22F9DFD00AFFF3', module: 'SD', isShow: false },
]

let initialized = false

function IFRAMEVIEWER() {
  const iframeRef = useRef(null)
  const formRef = useRef(null)
  const loaded = useRef(false)

  useEffect(() => {
    if (initialized) return
    initialized = true

    winResizer()

    const handleResize = () => {
      winResizer()
      sendResize()
    }
    window.addEventListener('resize', handleResize)
    return () => window.removeEventListener('resize', handleResize)
  }, [])

  // iframe resize
  function sendResize() {
    try {
      if (iframeRef.current && iframeRef.current.contentWindow) {
        iframeRef.current.contentWindow.postMessage(JSON.stringify({ type: 'resize' }), '*')
      }
    } catch (e) { /* cross-origin */ }
  }

  // form POST iframe
  function submitForm(code, isTitle, module) {
    const form = formRef.current
    if (!form) return
    form.innerHTML = ''
    form.target = 'REPORT_AUD'
    form.action = webRoot + '/iaud_main_view'

    const fields = { audServerUrl, id: code, isTitle: String(isTitle), mCode: module }
    Object.entries(fields).forEach(([name, value]) => {
      const input = document.createElement('input')
      input.type = 'hidden'
      input.name = name
      input.value = value
      form.appendChild(input)
    })
    form.submit()
    loaded.current = true
  }

  // (: form POST / : fnOpen )
  function openReport(code, isTitle, module) {
    if (!iframeRef.current) return

    if (loaded.current) {
      try {
        const w = iframeRef.current.contentWindow
        const viewer = (w.AUD && w.AUD.GetMainViewer) ? w.AUD.GetMainViewer() : null
        if (viewer && viewer.Dispose) try { viewer.Dispose() } catch (e) {}
      }
    }
  }
}

```

```

        if (w.AUD && w.AUD.SetCustomParams) w.AUD.SetCustomParams([{ KEY: 'VS_TEST', VALUE:
'VS_TEST1_VAL' }])
        w.fnOpen(code, isTitle, module)
        return
    } catch (e) {
        console.log('iframe reuse failed:', e.message)
    }
}
submitForm(code, isTitle, module)
}

// (titlebg iframe )
function winResizer() {
    const win_w = window.innerWidth
    const win_h = window.innerHeight

    const topPanel = document.querySelector('.top_panel')
    const leftPanel = document.querySelector('.left_panel')
    const footPanel = document.querySelector('.foot_panel')
    const audView = document.getElementById('AUDview')

    const top_panel_height = (topPanel && topPanel.offsetHeight) || 0
    const left_panel_width = (leftPanel && leftPanel.offsetWidth) || 0
    const foot_panel_height = (footPanel && footPanel.offsetHeight) || 0

    if (leftPanel) {
        leftPanel.style.height = `${win_h - top_panel_height}px`
        leftPanel.style.top = `${top_panel_height}px`
    }

    const mainGroup = document.querySelector('.main_group')
    if (mainGroup) {
        mainGroup.style.height = `${win_h - top_panel_height - foot_panel_height}px`
        mainGroup.style.top = `${top_panel_height}px`
        mainGroup.style.left = `${left_panel_width}px`
        mainGroup.style.width = `${win_w - left_panel_width}px`
    }

    if (audView) {
        audView.style.height = `${win_h - top_panel_height - foot_panel_height}px`
        sendResize()
    }
}

return (
    <>
    <div className="top_panel"></div>
    <div className="left_panel">
        <ul>
            {reportList.map(report => (
                <li key={report.code}>
                    <div
                        className="rep_div"
                        onClick={() => openReport(report.code, report.isShow, report.module)}
                    >
                        {report.name}
                    </div>
                </li>
            ))}
        </ul>
    </div>
    <div className="main_group VisibleFrame">
        <iframe
            ref={iframeRef}
            id="AUDview"
            name="REPORT_AUD"
            className="istudio-common-viewer"
            frameBorder="0"
            scrolling="no"
            style={{
                border: 'none',

```

```

        display: 'block',
        position: 'absolute',
        left: 0,
        top: 0,
        width: '100%',
        height: '100%',
      })
    />
    <form ref={formRef} method="post" style={{ display: 'none' }} />
  </div>
  <div className="foot_panel"></div>
</>
)
}

export default IframeViewer

```

IframeViewer.vue (Vue)

```

<script setup>
import { onMounted, onBeforeUnmount, ref } from 'vue'
import './layout.css'

const audServerUrl = window.IIT_DATA.audServerUrl
const webRoot = window.IIT_DATA.webRoot

// ( - )
const reportList = [
  { name: 'i-AUD ', code: 'REP4CE0125E54214D8396F788D30F335583', module: 'SD', isShow: true },
  { name: 'i-AUD 2', code: 'REP236AB97070714FA3AC22F9DFD00AFFF3', module: 'SD', isShow: true },
]

const iframeRef = ref(null)
const formRef = ref(null)
const loaded = ref(false)

let handleResize = null

onMounted(() => {
  winResizer()

  handleResize = () => {
    winResizer()
    sendResize()
  }
  window.addEventListener('resize', handleResize)
})

onBeforeUnmount(() => {
  if (handleResize) window.removeEventListener('resize', handleResize)
})

// iframe resize
function sendResize() {
  try {
    if (iframeRef.value && iframeRef.value.contentWindow) {
      iframeRef.value.contentWindow.postMessage(JSON.stringify({ type: 'resize' }), '')
    }
  } catch (e) { /* cross-origin */ }
}

// form POST iframe
function submitForm(code, isTitle, module) {
  const form = formRef.value
  if (!form) return
  form.innerHTML = ''
  form.target = 'REPORT_AUD'
  form.action = webRoot + '/iaud_main_view'

```

```

const fields = { audServerUrl, id: code, isTitle: String(isTitle), mCode: module }
Object.entries(fields).forEach(([name, value]) => {
  const input = document.createElement('input')
  input.type = 'hidden'
  input.name = name
  input.value = value
  form.appendChild(input)
})
form.submit()
loaded.value = true
}

// (: form POST / : fnOpen )
function openReport(code, isTitle, module) {
  if (!iframeRef.value) return

  if (loaded.value) {
    try {
      const w = iframeRef.value.contentWindow
      const viewer = (w.AUD && w.AUD.GetMainViewer) ? w.AUD.GetMainViewer() : null
      if (viewer && viewer.Dispose) try { viewer.Dispose() } catch (e) {}
      if (w.AUD && w.AUD.SetCustomParams) w.AUD.SetCustomParams([{ KEY: 'VS_TEST', VALUE:
'VS_TEST1_VAL' }])
      w.fnOpen(code, isTitle, module)
      return
    } catch (e) {
      console.log('iframe reuse failed:', e.message)
    }
  }
  submitForm(code, isTitle, module)
}

// (titlebg iframe )
function winResizer() {
  const win_w = window.innerWidth
  const win_h = window.innerHeight

  const topPanel = document.querySelector('.top_panel')
  const leftPanel = document.querySelector('.left_panel')
  const footPanel = document.querySelector('.foot_panel')
  const audView = document.getElementById('AUDview')

  const top_panel_height = (topPanel && topPanel.offsetHeight) || 0
  const left_panel_width = (leftPanel && leftPanel.offsetWidth) || 0
  const foot_panel_height = (footPanel && footPanel.offsetHeight) || 0

  if (leftPanel) {
    leftPanel.style.height = `${win_h - top_panel_height}px`
    leftPanel.style.top = `${top_panel_height}px`
  }

  const mainGroup = document.querySelector('.main_group')
  if (mainGroup) {
    mainGroup.style.height = `${win_h - top_panel_height - foot_panel_height}px`
    mainGroup.style.top = `${top_panel_height}px`
    mainGroup.style.left = `${left_panel_width}px`
    mainGroup.style.width = `${win_w - left_panel_width}px`
  }

  if (audView) {
    audView.style.height = `${win_h - top_panel_height - foot_panel_height}px`
    sendResize()
  }
}
}
</script>

<template>
  <div class="top_panel"></div>
  <div class="left_panel">
    <ul>

```

```

        <li v-for="report in reportList" :key="report.code">
            <div
                class="rep_div"
                @click="openReport(report.code, report.isShow, report.module)"
            >
                {{ report.name }}
            </div>
        </li>
    </ul>
</div>
<div class="main_group VisibleFrame">
    <iframe
        ref="iframeRef"
        id="AUDview"
        name="REPORT_AUD"
        class="istudio-common-viewer"
        frameborder="0"
        scrolling="no"
        style="border: none; display: block; position: absolute; left: 0; top: 0; width: 100%; height:
100%;"
    />
    <form ref="formRef" method="post" style="display: none;"></form>
</div>
<div class="foot_panel"></div>
</template>

```

AudAction.java (Vue, React)

```

package com.aud.embedded;

import com.fasterxml.jackson.core.JsonProcessingException;
import com.fasterxml.jackson.databind.ObjectMapper;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Service;
import org.springframework.web.reactive.function.client.WebClient;
import org.springframework.web.reactive.function.client.WebClientRequestException;
import org.springframework.web.reactive.function.client.WebClientResponseException;
import org.thymeleaf.util.StringUtils;
import java.security.*;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.security.spec.PKCS8EncodedKeySpec;
import java.util.*;
import java.util.concurrent.atomic.AtomicReference;

@Service
public class AudAction {
    private final WebClient audWebClient;
    private final String audApId;
    private final String audApSecretFilePath;
    private final String apUpdateAddr;
    private final String audSecretKey;

    public AudAction(@Qualifier("audWebClient") WebClient audWebClient
        , @Value("${aud.x-aud-ap-id}") String audApId
        , @Value("${aud.ssh-key-path}") String audApSecretFilePath
        , @Value("${aud.x-ap-update-addr}") String apUpdateAddr
        , @Value("${aud.x-aud-ap-secret-key}") String audSecretKey) {
        this.audWebClient = audWebClient;
        this.audApId = audApId;
        this.audApSecretFilePath = audApSecretFilePath;
        this.apUpdateAddr = apUpdateAddr;
        this.audSecretKey = audSecretKey;
    }

    public String getAccessToken() throws Exception {

```



```

AtomicReference<String> apToken = new AtomicReference<>("");
// Application
//
//
String loginUserCode = {AUD7 };

try {
    // aud7 secret key ssh private key
    PrivateKey privateKey = loadPrivateKey(audApSecretFilePath);
    // aud7 secret key
    String signedMessage = signMessage(audSecretKey, privateKey);

    ResponseEntity<String> res =
        audWebClient.post()
            .uri("/api/auth/sign/ap/token")
            .headers(headers -> {
                headers.set("X-AUD-AP-Id", audApId);
                headers.set("X-AUD-AP-Secret-SSH", signedMessage);
                headers.set("X-AP-UPDATE-ADDR", apUpdateAddr);
                headers.set("X-AUD-USER", loginUserCode);
            })
            .retrieve()
            .toEntity(String.class)
            .block();
    if (res == null) throw new Exception("");
    res.getHeaders().get("bimatrix_ap_accessToken").stream().findFirst().ifPresent(apToken::set);
    if (StringUtils.isEmpty(apToken.get()))
        throw new Exception("token empty");
} catch (WebClientRequestException e) {
    e.printStackTrace();
    // throw
} catch (WebClientResponseException e) {
    if (HttpStatus.UNAUTHORIZED == e.getStatusCode()) {
        e.printStackTrace();
        //
        // throw
    } else {
        e.printStackTrace();
        // throw
    }
}

} catch (Exception e){
    e.printStackTrace();
}

return apToken.get();
}

//
private static PrivateKey loadPrivateKey(String path) throws Exception {
    // ( )
    System.out.println("Private Key path: " + path);
    String keyPEM = new String(Files.readAllBytes(Paths.get(path)))
        .replaceAll("-----BEGIN PRIVATE KEY-----", "")
        .replaceAll("-----END PRIVATE KEY-----", "")
        .replaceAll("\\s", ""); //

    byte[] keyBytes = Base64.getDecoder().decode(keyPEM);
    PKCS8EncodedKeySpec keySpec = new PKCS8EncodedKeySpec(keyBytes);
    KeyFactory keyFactory = KeyFactory.getInstance("RSA");
    return keyFactory.generatePrivate(keySpec);
}

//
private static String signMessage(String message, PrivateKey privateKey) throws Exception {
    Signature signature = Signature.getInstance("SHA256withRSA");
    signature.initSign(privateKey);
    signature.update(message.getBytes());

    byte[] signedBytes = signature.sign();

```

```

        return Base64.getEncoder().encodeToString(signedBytes);
    }

    public String getAudMainJsonParam(Map<String, String[]> paramMap) {
        final ObjectMapper mapper = new ObjectMapper();
        String json = "";
        List<Map<String, String>> paramList = new ArrayList<>();
        for (String key : paramMap.keySet()) {
            if(!key.startsWith("VS") && !key.startsWith("VN") && !key.startsWith("V_")) continue;
            paramList.add(
                new HashMap<>() {{
                    put("KEY", key);
                    Arrays.stream(paramMap.get(key)).findFirst().ifPresent(value -> {
                        put("VALUE", value);
                    });
                }}
            );
        }
        try {
            json = mapper.writeValueAsString(paramList);
        } catch (JsonProcessingException e) {
            //throw new Exception(e.getMessage());
        }
        return json;
    }
}

```

 iFrame



spring-embedded-react.zip



spring-embedded-vue.zip