금융권 VDE 워크스페이스 표준 이미지 설계서 (개선본)

버전: 1.1
작성일: 2026-01-06
적용 대상: 금융권 폐쇄망(VDE) 환경에서 code-server(Web IDE) 기반 개발 워크스페이스

---

1. 목적 및 설계 원칙

1.1 목적

본 문서는 금융권 폐쇄망(VDE) 환경에서 **웹 기반 code-server IDE 워크스페이스**를 안정적·재현 가능·감사 가능하게 제공하기 위한 **워크스페이스 표준 이미지 설계 기준**를 정의한다.

- 사용자별 임의 설치를 최소화하고, 승인된 개발환경을 즉시 제공
- AI 기능(Tab 자동완성, OpenCode 에이전트, Chat, Code RAG)은 **외부 서비스 컨테이너로 분리**
- 워크스페이스 컨테이너는 **경량 CPU 기반**으로 운영

1.2 핵심 설계 원칙

- **분리 원칙**: Workspace(IDE) ≠ Model/AI Runtime
- **단일 진입점**: 모든 AI 호출은 AI Gateway 를 통해서만 허용
- **불변 이미지**: 개발 중인 워크스페이스는 이미지 업데이트로 변경하지 않음
- **강통제 기본값**: 확장/패키지/네트워크는 기본 차단 후 허용

---

2. 전체 아키텍처 개요

- VDE 사용자 → Web Portal → Workspace 생성/접속
- Workspace Container
    - code-server (Web IDE)
    - OpenCode CLI
    - 승인된 VS Code Extensions (사전 설치)
- 외부 서비스 컨테이너(공용)
    - Tabby Autocomplete
    - OpenCode Model Service
    - Chat LLM Service
    - Code RAG Service

- AI Gateway (단일 AI 진입점)
    - 인증/인가
    - 요청 라우팅
    - 감사 로그
    - DLP/PII 필터링

---

3. AI Gateway 설계

3.1 목적 및 역할

AI Gateway 는 워크스페이스(code-server)에서 발생하는 모든 AI 호출을 단일 경로로 수렴시키는 **보안·감사·정책 집행 계층**이다.

- **단일 Egress 종점**: Workspace 는 AI Gateway 외부로 AI 관련 트래픽을 직접 송신하지 않는다.
- **정책 라우팅**: 요청 유형(autocomplete/agent/chat/rag)에 따라 백엔드 서비스를 분기한다.
- **인증/인가**: 사용자/워크스페이스/프로젝트 단위 접근 제어를 수행한다.
- **감사 및 추적성**: 모든 요청에 Correlation ID 를 부여하고, 감사 이벤트를 중앙 수집한다.
- **데이터 보호**: DLP/PII/비밀키 탐지 및 마스킹/차단 정책을 집행한다.

3.2 배치 및 네트워크 경계

- AI Gateway 는 내부망 서비스로 배치하며, 외부(사용자)에서 직접 접근하지 않는다.
- Workspace(Container)에서 허용하는 AI 관련 egress 는 **AI Gateway 로만 제한**한다.
- AI 백엔드(Tabby/OpenCode Model/Chat LLM/RAG)는 Gateway 뒤에 위치하며, Workspace 에서 직접 호출 불가하다.

3.3 인터페이스(Endpoint) 표준

AI Gateway 는 최소 아래 Endpoint 를 제공한다.

1) **Autocomplete**

- `POST /v1/autocomplete` (또는 WebSocket/SSE 기반 스트리밍)
- 라우팅: Tabby Autocomplete

2) **Agent (OpenCode)**

- `POST /v1/agent` (스트리밍 권장)
- 라우팅: OpenCode Model Service
- 사용 맥락: 코드 변경 제안/패치 생성/리팩토링

3) **Chat**

- `POST /v1/chat` (스트리밍 권장)
- 라우팅: Chat LLM Service
- 사용 맥락: 질의응답, 문서화, 설계 검토

4) **Code RAG**

- `POST /v1/rag/query`
- `POST /v1/rag/index` (운영 정책에 따라 별도 인덱서로 분리 가능)
- 라우팅: Code RAG Service

주의: RAG 인덱싱은 일반적으로 리소스/권한 영향이 커서, Workspace 에서 직접 호출을 허용할지(또는 CI/인덱서 전용 계정만 허용할지) 정책으로 분리한다.

3.4 인증·인가(금융권 필수)

3.4.1 신원 전달

- 사용자 인증은 Web Portal/SSO Proxy 에서 수행하고, AI Gateway 는 내부망에서 토큰 기반으로 검증한다.
- 표준 헤더(예):
  - `X-User-Id`, `X-Tenant-Id`, `X-Workspace-Id`, `X-Project-Id`
  - `X-Correlation-Id` (없으면 Gateway 에서 생성)

3.4.2 권한 모델(예시)

- `tenant` → `project` → `workspace` 스코프
- 정책 예:
  - Autocomplete: workspace 단위 허용
  - Agent: 특정 프로젝트/브랜치에서만 허용(권한 높은 작업)
  - Chat: 프로젝트 범위 컨텍스트 허용 여부 정책화
  - RAG Query: project 범위 읽기 권한 필요
  - RAG Index: 인덱서 전용 서비스 계정만 허용(권장)

3.5 정책 엔진 및 라우팅

- 요청의 `tool_type` 또는 endpoint 경로 기반으로 라우팅한다.
- 모델 선택은 Workspace 가 직접 지정하지 않고, Gateway 정책으로 결정한다.
  - 예: `agent` 트래픽은 `opencode-model` 로, `chat` 트래픽은 `chat-llm` 로 강제
- 레이트리밋/쿼터는 사용자·프로젝트·워크스페이스 단위로 적용한다.

3.6 데이터 보호(DLP/PII/Secret) 정책

- 입력 프롬프트/코드 조각에 대해 아래 정책을 적용한다.
  - 비밀키(토큰/인증서/SSH 키) 패턴 탐지 및 마스킹/차단

    - 주민번호/계좌번호 등 PII 탐지(정규식 + 룰 기반)
    - 사내 중요정보 분류 태그(예: CONFIDENTIAL) 기반 차단
- 차단/마스킹 이벤트는 감사 로그로 남긴다.

3.7 감사 로그(최소 필드)

AI Gateway 는 각 요청에 대해 최소 아래 메타데이터를 감사 이벤트로 기록한다.

- 공통: `timestamp`, `correlation_id`, `tenant_id`, `project_id`, `workspace_id`, `user_id`, `client_ip(내부)`, `endpoint`, `status_code`, `latency_ms`
- 사용량: `prompt_tokens`, `completion_tokens` (가능한 경우)
- 정책: `route_target`, `policy_version`, `dlp_action(allow/mask/block)`
- 변경작업(Agent) 추가: `repo_id`, `branch`, `changed_files(list or hash)`, `diff_hash`

원문 프롬프트/응답 저장 여부는 금융권 규정에 따라 별도 정책으로 분리하며, 기본은 **저장 최소화**를 권장한다.

3.8 가용성 및 장애 격리

- AI Gateway 는 stateless 로 구성하고 수평 확장(HPA)을 적용한다.
- 백엔드 서비스 장애 시:
    - Autocomplete/Chat/Agent 를 상호 격리하여 부분 장애로 제한
    - 표준 에러 코드/재시도 정책 정의

3.9 구성 및 배포

- 권장 구성요소:
    - Reverse Proxy(예: NGINX/Envoy) + App(FastAPI/Go) 또는 통합 게이트웨이
    - mTLS(내부망 서비스 간) 또는 서비스 메시 정책(조직 표준에 따름)
- 설정은 ConfigMap/Secret 로 분리하고, 버전(Policy Version)을 명시한다.

---

4. 베이스 이미지 및 실행 사용자

베이스 이미지 및 실행 사용자

3.1 Base OS

- Ubuntu 24.04 LTS (또는 금융권 내부 Hardened 이미지)
- 패키지 소스: **사내 APT 미러만 허용**

3.2 실행 사용자 정책

- 기본 사용자: `coder` (UID/GID: 1000:1000)

- root 실행 금지
- 볼륨 마운트 호환을 위해 UID 고정

---

4. 포함 패키지 목록

4.1 공통 필수 유틸리티

- ca-certificates, curl, wget
- git, openssh-client
- bash, jq
- zip, unzip, tar
- tzdata (Asia/Seoul), locales (ko_KR.UTF-8)

4.2 개발 도구 (공통 베이스)

- build-essential
- python3, python3-venv, pipx
- nodejs LTS + corepack
- openjdk-17-jdk

  언어별 요구가 큰 경우, `workspace-base` + `workspace-java`, `workspace-python` 등 파생 이미지로 분리한다.

---

5. code-server 구성

5.1 설치 및 실행

- 설치 위치: `/opt/code-server/`
- 실행 파일: `/opt/code-server/bin/code-server`

5.2 설정 경로

- 설정 파일: `~/.config/code-server/config.yaml`
- 데이터/확장 경로: `~/.local/share/code-server/`

5.3 보안 설정 기본값

- telemetry 비활성화
- auto-update 비활성화
- `auth: none` 사용 조건:
  - 외부 Ingress/Proxy 에서 **SSO 인증 강제**

- code-server Pod 는 내부 네트워크에서만 접근 가능

---

6. VS Code 확장(Extensions) 관리 정책

6.1 기본 정책 (강통제)

- 사용자의 런타임 확장 설치 금지
- 모든 확장은 **이미지 빌드 시 사전 설치**
- 외부 마켓 접근 차단

6.2 확장 설치 방식

- VSIX 파일을 `/opt/vsix/`에 포함
- 이미지 빌드 단계에서:

```
code-server --install-extension /opt/vsix/<extension>.vsix
```

6.3 확장 경로 정책

- `~/.local/share/code-server/extensions` 는 이미지에 포함
- 런타임에서는 읽기 전용(파일시스템 권한 또는 정책으로 제어)

---

7. OpenCode(opencode) 구성

7.1 설치 원칙

- OpenCode CLI 는 **워크스페이스 컨테이너 내부**에 설치
- 설치 위치: `/usr/local/bin/opencode`
- 실행 권한: 사용자 `coder`

7.2 설정 파일 (정정 사항)

- 설정 형식: **JSON**
- 설정 경로:

```
~/.config/opencode/opencode.json
```

OpenCode 전역 설정은 JSON 형식을 사용하며, TOML 형식은 사용하지 않는다.

7.3 설정 내용 원칙

- LLM Provider/Endpoint: **AI Gateway URL 만 지정**
- 모델 선택/라우팅은 Gateway 정책으로 통제

- 토큰/비밀정보는 설정 파일에 직접 포함하지 않음

7.4 시크릿 관리

- Kubernetes Secret 또는 런타임 마운트 방식 사용
- 예: `/run/secrets/ai_gateway_token` (0400)

---

8. 경로 및 권한 표준

구분	경로	권한
워크스페이스	`/workspace/<project>`	coder:coder 750
code-server 설정	`~/.config/code-server/`	700
code-server 확장	`~/.local/share/code-server/extensions`	755 (읽기)
OpenCode 설정	`~/.config/opencode/opencode.json`	600
VSIX 저장소	`/opt/vsix/`	root 644
시크릿	`/run/secrets/*`	root 0400

9. 네트워크 정책

9.1 Egress 허용 대상

- AI Gateway (단일 엔드포인트)
- 사내 Git / Artifact Registry
- 사내 패키지 미러

9.2 차단 대상

- 외부 인터넷 전체
- 직접적인 LLM/Model 서비스 접근

---

10. 헬스체크 및 운영 주의사항

10.1 헬스체크

- code-server `/healthz` 사용
- readiness 판단 시 `status=alive` 여부 확인 권장

10.2 운영 주의사항

- `/healthz` 가 expired 상태여도 IDE 세션이 즉시 장애는 아닐 수 있음
- 자동 재시작 정책은 신중히 적용

---

11. 업데이트 및 배포 전략

11.1 버전 관리

- 이미지 버전: `workspace-base:YYYY.MM.DD-buildN`
- code-server, opencode, 확장 VSIX 버전 고정

11.2 롤아웃 방식

- 신규 워크스페이스만 신규 이미지 사용
- 기존 워크스페이스는 유지
- 카나리 또는 블루/그린 전략 권장

11.3 인플레이스 업데이트 금지

- 개발 중 세션/상태 손상 방지
- 감사/재현성 확보 목적

---

12. 검증 및 감사 항목 (필수)

- code-server 기동 및 로그인 검증
- 승인 확장 로딩 여부
- opencode CLI 실행 및 AI Gateway 연동
- 샘플 Repo 에서 diff 생성/적용/롤백
- 네트워크 egress 정책 검증
- 사용자/워크스페이스/요청 간 Correlation ID 로깅

---

13. 결론

본 표준 이미지는 금융권 VDE 환경에서 요구되는 **보안성, 재현성, 감사 가능성**을 충족하면서도, AI 기반 개발 생산성을 제공하기 위한 최소·필수 구성이다.

- Workspace 는 경량·불변 이미지
- AI/모델은 외부 서비스로 분리
- 모든 통제는 Gateway 와 이미지 버전으로 일원화

본 설계를 기준으로 조직 내 표준 Workspace 이미지를 정의·운영한다.

부록 B. 젠스파크 채용 항목 통합 반영 (v1.2)

본 설계서는 v1.1 개선본을 기준으로 젠스파크 의견 중 **채용 가치가 검증된 항목만 선별 통합**하여 v1.2 로 확정한다.

B.1 신규 통합 항목 요약

- Web IDE 선택 근거 표(code-server vs VS Code Server)
- AI Gateway 이후 운영 설계(모니터링/비용 추적/DR)
- 규제 준수 매핑(전자금융감독규정, 개인정보보호법)
- 감사로그 확장: Diff 해시 기반 변경 추적

(※ OpenCode 설정 포맷 오류(opencode.toml)는 전면 배제하고 JSON 기준 유지)

B.2 Web IDE 옵션 비교 (제출용 핵심)

평가 항목	code-server	VS Code Server
소스코드 감사	가능(MIT)	불가(비공개)
폐쇄망 운영	완전 가능	제한적
외부 계정 의존	없음	MS 계정 일부 필요
텔레메트리 통제	완전 차단	부분 제한

결론: 금융권 보안 심사·망분리·감사 요건 충족을 위해 code-server 를 표준 Web IDE 로 채택한다.

B.3 AI Gateway 운영 확장 설계

B.3.1 모니터링 및 비용 추적

- Prometheus Metrics:
  - `ai_gateway_requests_total{service, project_id}`
  - `ai_gateway_latency_seconds{service}`
  - `ai_gateway_tokens_total{project_id, model}`
- 목적: 프로젝트별 비용 통제, 이상 사용 탐지

B.3.2 재해 복구 및 백업

- Workspace PVC: Velero + CSI Snapshot
- Gateway 장애 시:
    1) 캐시 응답
    2) DR 센터 Gateway DNS 절체
    3) AI 기능 비활성화(IDE 기본 기능 유지)

B.4 규제 준수 매핑 (요약)

전자금융감독규정

- 제 15 조(해킹 방지): 폐쇄망, 단일 AI Gateway, DLP
- 제 13 조(전산자료 보호): 프로젝트별 데이터 격리, 암호화된 PVC

개인정보보호법

- 제 29 조(안전조치): 접근통제, 감사로그, 접속기록 보관

부록 C. 제출용 요약본(Executive Summary)

C.1 설계 개요 (1 페이지 요약)

- Web IDE: code-server 기반
- Workspace: 경량·불변 컨테이너
- AI 기능: 외부 GPU 서비스 + AI Gateway 단일 통제
- 보안 핵심: 망분리, egress 통제, 감사로그

C.2 보안/운영 핵심 포인트

- 사용자 임의 설치 불가(확장/모델)
- 모든 AI 호출은 Gateway 경유
- 변경 이력은 Diff 해시로 추적 가능

부록 D. 심사위원 예상 Q&A

Q1. 왜 code-server 를 선택했습니까?

A. 소스코드 감사 가능, 외부 계정 의존 없음, 텔레메트리 완전 차단이 가능하여 금융권 폐쇄망 요건을 충족합니다.

Q2. AI 가 외부로 데이터 유출할 위험은 없습니까?

A. Workspace 는 AI Gateway 만 접근 가능하며, Gateway 에서 DLP/PII 필터링과 정책 라우팅을 수행합니다.

Q3. AI 서비스 장애 시 개발 업무는 중단됩니까?

A. 아닙니다. AI 기능만 단계적으로 비활성화되며 IDE 기본 기능은 유지됩니다.

Q4. AI 가 생성한 코드 변경은 어떻게 추적합니까?

A. 모든 변경은 Diff 해시로 기록되어 감사 로그 및 SIEM 에서 추적 가능합니다.

Q5. 비용 폭증이나 오남용은 어떻게 통제합니까?

A. 프로젝트별 토큰 사용량을 실시간 모니터링하고, 이상 징후 발생 시 정책적으로 제한합니다.

※ 본 문서는 금융권 보안·감사·운영 심사를 위한 최종 통합본(v1.2)으로 사용 가능하다.