

은행업무 프로그램

사람 희현 은빈 hyoeun jeong (백엔드)배근호 GY K

SmartBankSystem 팀 프로젝트 가이드 (외부 저장 + 암호화 적용)

팀 구성: 총 6명

모두가 각자 하나의 기능 클래스를 맡아 개발합니다!

데이터 저장: JSON 형식으로 `.json` 파일

비밀번호/인증서 암호화 적용

모든 기능은 `BankSystem` 또는 `main.py` 에서 통합 호출

1 로그인 및 공인인증서 관리 (`LoginManager`)

기능 요약:

- 공인인증서 발급 및 비밀번호 암호화 저장
- 로그인/로그아웃 기능 구현

데이터 파일: `certs.json`

함수 목록:

- `issue_certificate(user_id, password)`
 - 입력한 비밀번호를 단방향 암호화(`hashlib`) 후 저장
- `verify_password(user_id, input_password)`
 - 입력 비밀번호를 암호화해 기존값과 비교
- `logout_user()`
 - 현재 로그인 정보 초기화

암호화 사용: ☒ 단방향 (비교만 가능)

2 계좌 개설 및 해지 (`AccountManager`)

기능 요약:


- 새 계좌 개설 시 고유 번호 부여

- 계좌 해지는 잔액 0일 경우에만 가능

 데이터 파일: `accounts.json`

 함수 목록:

- `create_account(user_name, resident_id)`
 - 사용자 이름과 주민번호 저장
 - 계좌번호는 자동으로 생성 (예: 1001, 1002, ...)
- `close_account(account_no)`
 - 잔액이 0인지 확인하고 삭제
- `verify_account(account_no)`
 - 해당 계좌번호가 존재하는지 확인
- `validate_resident_id(resident_id)`
 - 주민번호 형식 검증 (예: `YYYYMMDD-XXXXXXX`)

 암호화 사용: ❌ (개인정보는 기본 저장)

3 입금 및 출금 기능 (`TransactionManager`)


 기능 요약:

- 사용자의 계좌에 입금/출금
- 잔액은 `balances.json` 에 저장 및 갱신

 데이터 파일: `balances.json`

 함수 목록:

- `deposit(account_no, amount)`
 - 해당 계좌의 잔액을 증가시킴
- `withdraw(account_no, amount)`
 - 잔액이 부족할 경우 오류 출력
- `get_balance(account_no)`
 - 잔액 조회 (없으면 0으로 처리)

 암호화 사용: ❌ (금액은 암호화 불필요)

4 계좌 이체 기능 (`TransferManager`)

📌 기능 요약:

- 계좌 간 이체 수행
- 이체 시 1일 한도, OTP 인증 포함

📁 데이터 파일: `transfers.json` , `otp.json` (임시 저장)

🔧 함수 목록:

- `transfer(from_acc, to_acc, amount)`
 - 보낸 사람 계좌에서 차감, 받는 사람 계좌에 추가
- `verify_recipient(account_no)`
 - 수신 계좌 유효성 검사
- `generate_otp()`
 - 6자리 OTP 코드 생성
- `verify_otp(user_input)`
 - 입력 OTP와 일치 여부 확인
- `record_transfer_history(...)`
 - 날짜, 시간, 금액, 출/입금 계좌 기록

🧠 암호화 사용: 🔑 OTP는 간단한 난수 생성, 복호화는 불필요

📁 5 거래 내역 조회 기능 (`HistoryManager`)

📌 기능 요약:

- 입출금 및 이체 내역 확인
- 날짜별 필터링, 최근 5건, CSV 내보내기

📁 데이터 파일: `history.json` , `export_history.json`

🔧 함수 목록:

- `get_all_history(account_no)`
 - 해당 계좌의 전체 거래 조회
- `get_recent_history(account_no, count=5)`
 - 최근 `count` 개 거래 출력
- `search_by_date(account_no, start_date, end_date)`

- 시작 ~ 종료 날짜 범위 내 거래만 출력
- `export_csv(account_no)`
 - 거래내역을 CSV형식으로 내보내기 (json 확장자 사용)

🧠 암호화 사용: ❌ (기록 공개 가능)

👤 6 사용자 정보 관리 기능 (`UserManager`)

📌 기능 요약:

- 사용자 등록 및 개인정보 수정
- 비밀번호는 양방향 암호화 저장 후 복호화 가능

📁 데이터 파일: `users.json` , `user_passwords.json`

🔧 함수 목록:

- `register_user(name, phone, email, pw)`
 - 비밀번호는 양방향 암호화하여 저장
- `change_password(user_id, new_pw)`
 - 암호화된 비밀번호 갱신
- `update_user_info(user_id, phone, email)`
 - 전화번호와 이메일만 수정 가능 (이름은 고정)

🧠 암호화 사용: ✅ 양방향 암호화 (복호화 가능, `Fernet` 사용)

📦 공통 저장 규칙 및 파일 위치

| 파일명 | 내용 |
|----------------------------------|----------------------------|
| <code>certs.json</code> | 공인인증서 비밀번호 (단방향 암호화) |
| <code>accounts.json</code> | 사용자 계좌 정보 (이름, 주민번호, 계좌번호) |
| <code>balances.json</code> | 계좌번호 → 잔액 |
| <code>transfers.json</code> | 이체 내역 (보낸사람, 받는사람, 금액, 시간) |
| <code>history.json</code> | 거래 기록 (입출금/이체 포함) |
| <code>users.json</code> | 사용자 이름, 전화번호, 이메일 등 |
| <code>user_passwords.json</code> | 로그인 비밀번호 (양방향 암호화) |

💡 각 팀원은 하나의 `.py` 파일을 만들고,

`main.py` 또는 `BankSystem.py` 에서 클래스를 모두 통합해 실행하게 하세요!

원하시면 각 클래스별 템플릿도 바로 만들어드릴게요 😊

어떤 파트부터 개발하고 싶으신가요?