OH:SY -FINANCIAL-

오경택 | 홍훈의 | 송가람 | 유승경

A Table of contents.

Part 1,

팀원 소개

Team name: OH:SY -FINANCIAL-

팀원 소개

오경택

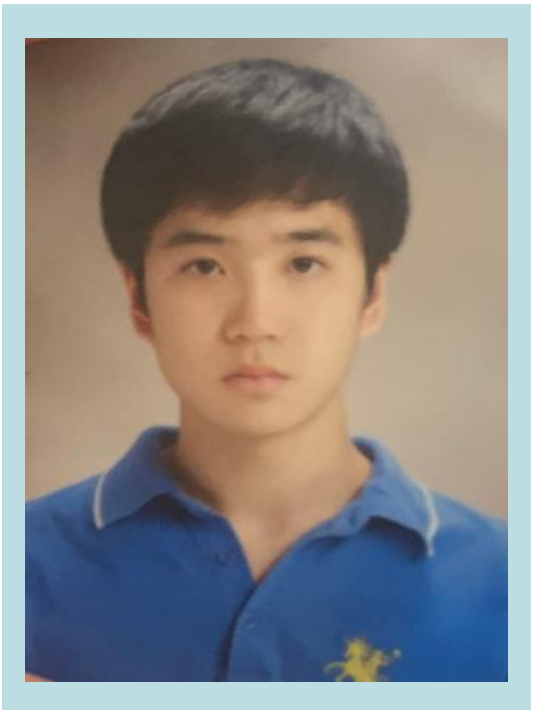
gyeongtaek.dev@gmail.com

홍훈의

asdfzxcv5621@.comnaver

송가람

sjjsy9634@naver.com



유승경

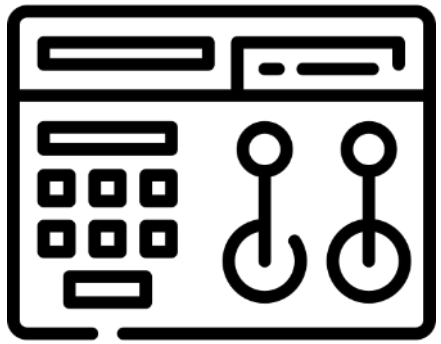
tmdrud7766@gmail.com

[intel] Edge AI SW 아카데미 - 6기 : 리눅스프로그래밍

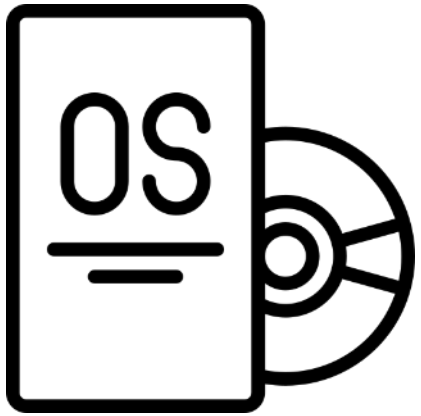
Part 2,

초록

프로젝트 개요

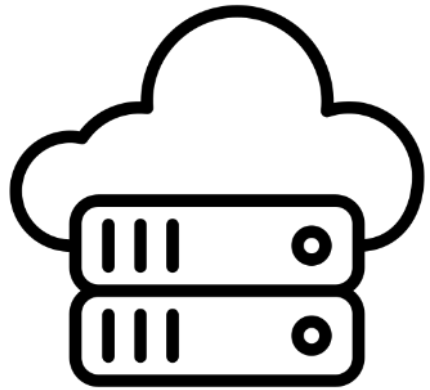


입력 장치
터치 스크린



운영 체제
Linux 기반 Raspberry Pi

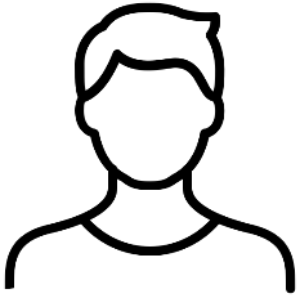
디스플레이
SPI 기반 LCD (fbcp- ili9341)



데이터 베이스
Maria DB (My SQL기반)

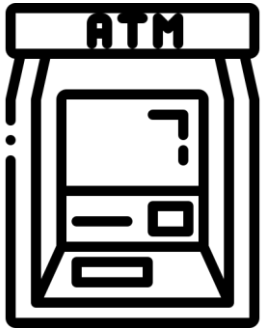
프로젝트 목표

01



사용자 친화적인 UI 및 보안 기능 강화

02



소형 임베디드 시스템에서 작동 가능한 ATM

03

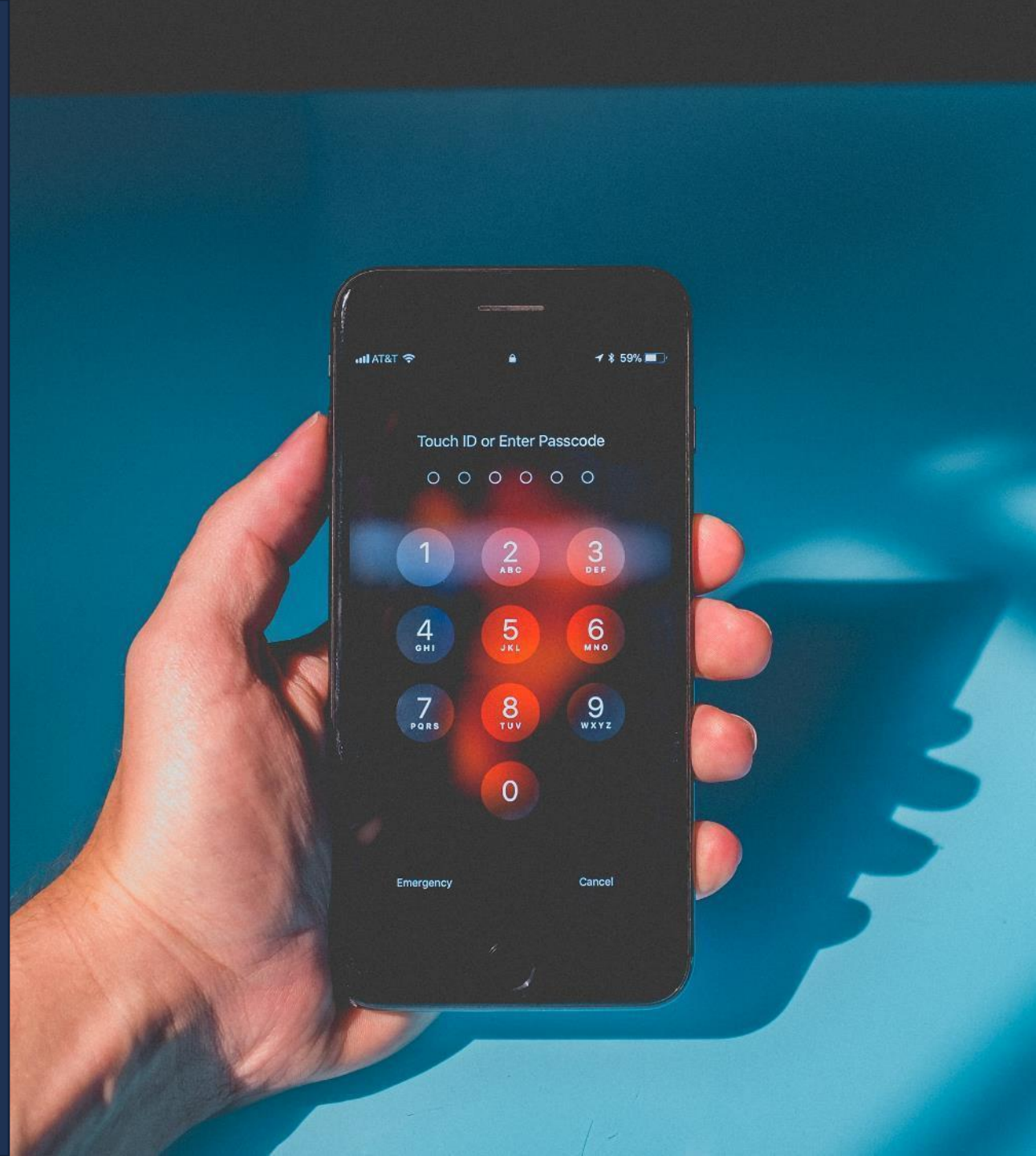
Raspberry Pi 기반의 효율적인 ATM솔루션 구현

04

실시간 데이터베이스 연동을 통한 금융 서비스 제공

Part 3,

제품 소개


AT&T
59%
Touch ID or Enter Passcode
1
2 ABC
3 DEF
4 GHI
5 JKL
6 MNO
7 PQRS
8 TUV
9 WXYZ
0
Emergency
Cancel

제품 소개

품목	전자기기	제조사	OHSY FINANCIAL
모델명	터치터치 ATM	가격	100,000₩

특장점

5세 이상의 어린이를 위한 ATM기 장난감

최신식 터치 방식 탑재

CCTV 탑재

DB서버 구현

주요 기능

금융서비스

인가

보안

프로토타입 시연(로그인)

프로토타입 시연(입금)

프로토타입 시연(출금)

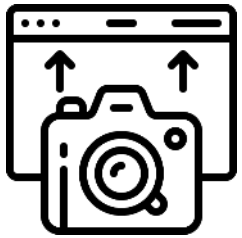
프로토타입 시연(송금)

프로토타입 시연(잔액 확인)

프로토타입 시연(뒤로 가기)

제품 비교군

01



실시간 카메라 사용 및 영상 녹화 기능

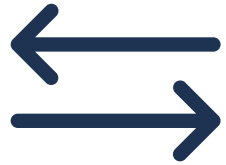
02

LCD 터치로 재미를 극대화

03

실사화를 통한 교육 가능

04



실제 데이터를 주고 받을 수 있음

Part 4,

프로젝트 진행 상황

프로젝트 진행 상황

2025.1.31
ATM UI/UX 구현
기본적인 은행 UI 구현

2025.2.1
DB 구현
Maria DB 를 이용한 DB 구현

2025.2.2
리눅스 서버 연결
DB 서버 연동

2025.2.3
UI 화면 연동
입,출금,메인 화면 끼리 연동

2025.2.4
카메라 연동
라즈베리 파이에 카메라 연동

2025.2.5
터치 연동
좌표를 받아와 버튼에 적용

2025.2.6
인자 값 서버 전송
터치해서 받은 값을 서버에 전송

2025.2.7
카메라 서버에 전송
영상을 서버에 저장

2025.2.8
프로토타입 개발
개발 및 테스트

Part 5,

개발 결과

시스템 구조

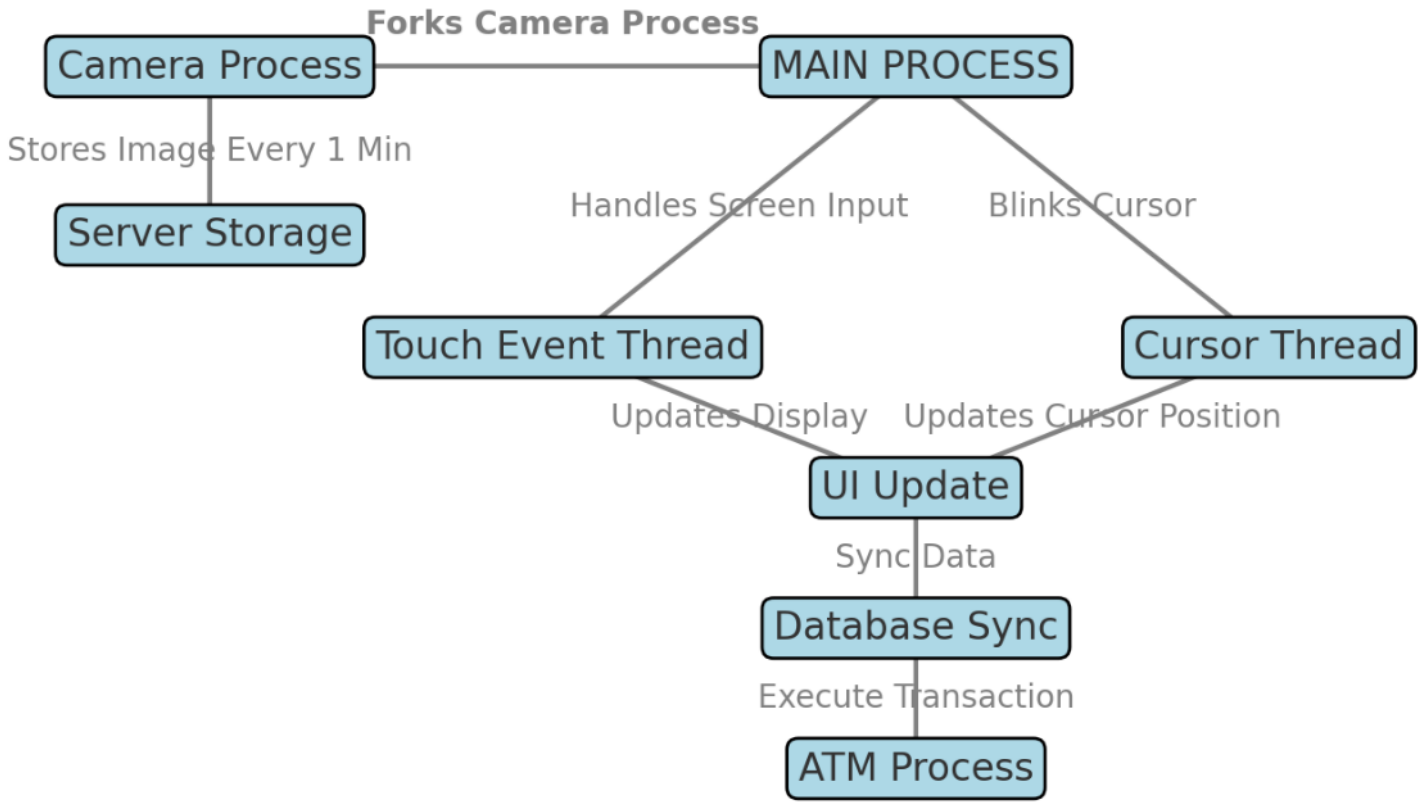
파일 구성 및 기능

파일명	주요기능
intel_bank_atm_login.c	사용자 로그인 처리 초기 화면 UI
intel_bank_atm_main.c	기능 선택 인터페이스 작업 분기 처리
intel_bank_atm_deposit.c	입금 처리
intel_bank_atm_withdrawal.c	출금 처리
intel_bank_atm_transfer.c	계좌이체 처리 수취인 계좌 확인
intel_bank_atm_check_balance.c	잔액 조회
intel_bank_atm_end.c	종료 화면 처리 초기화면 복귀
intel_bank_atm_camera.c (&back)	CCTV 촬영 영상 저장
intel_bank_atm_header.h	공통 헤더 정의
intel_bank_atm_client.c (&back)	클라이언트 모니터링 파일 전송 서버 연동

시스템 구성 요소

구성 요소	세부 내용
UI 시스템	- 디스플레이 갱신 (프레임 버퍼) - 터치스크린 인터페이스 - 사용자 입력 처리
데이터베이스	- TCP/IP 기반 MariaDB 연동 - 계좌 관리 및 동기화
보안 시스템	- DB를 통한 사용자 인증 - CCTV 녹화
하드웨어 인터페이스	- 터치스크린 제어 - 카메라 디바이스 - 디스플레이 출력
네트워크	- 서버 클라이언트 TCP/IP 통신 - 실시간 데이터 전송 - 모니터링 시스템
프로세스 관리	- 개별 프로세스를 실행하여 화면, 기능 분리 execl() 을 사용 - 멀티 스레딩 활용

코드 관계도



메인 프로세스

- 카메라 프로세스 실행
- 커서 동작 제어 (터치)
- 화면 입력 처리

카메라 프로세스

- 포크되어 독립적으로 실행
- 1분마다 영상 저장
- 클라이언트 실행 파일을 통해 서버로 영상 전송

사용자 인터페이스

- 커서 스레드 터치의 위치 정보를 계속 해서 업데이트
- 터치 이벤트 스레드 사용자의 터치 입력 처리와 디스플레이 업데이트

데이터의 처리 흐름

- UI 업데이트 화면 갱신
- 데이터베이스 동기화
- ATM 프로세스 거래 실행 담당

핵심 소스코드(UI)

UI 시스템의 핵심 구조와 특징

- 프레임버퍼, 비트맵 기반 그래픽 시스템
- **하드웨어 직접 제어**
  - 디스플레이 하드웨어에 직접 접근
  - 중간 레이어 없이 메모리에 직접 쓰기
  - 낮은 지연 시간과 빠른 응답속도
- **리소스 효율성**
  - X 추가 라이브러리
  - 메모리 매핑을 통한 효율적인 리소스 관리
  - 시스템 리소스 사용 최소화
- 저희의 핵심 주제인 교육적인 장난감의 취재에 맞게 성능이 낮은 기계에서도 효율적으로 작동 가능

- **간편한 사용**
- Draw_rect()
- Draw_rounded_rect()
- Draw_circle()
- Draw_letter()
- Draw_number()
- Draw_text()
- 위와 같은 단순한 구조를 사용하여 사용자는 빠른 시간에 원하는 구조의 UI를 픽셀 단위로 완성시킬 수 있습니다.

- **멀티 스레드**
  - 커서의 깜박임 애니메이션 스레드
  - 터치의 입력 처리 스레드

핵심 소스코드(DB와 기능)

(1) 로그인 처리 (`intel_bank_atm_login.c`)

- 기능:
  - 사용자가 ATM에 로그인 (ID, 비밀번호 입력)
  - 입력 값이 DB의 사용자 정보와 일치하는지 확인
- 사용된 기술:
  - ✅ 터치스크린 입력 처리 (`handle_touch_event()`)
  - ✅ 데이터베이스 연동 (`mysql_query()`)
  - ✅ 멀티스레딩 (`pthread_create()`)
  - ✅ 화면 UI 업데이트 (`draw_alert_box()`)

(2) 잔액 조회 (`intel_bank_atm_check_balance.c`)

- 기능:
  - 현재 계좌의 잔액을 확인
  - 잔액이 0 이하일 경우 에러 메시지 표시
- 사용된 기술:
  - ✅ SQL 조회 (`SELECT money FROM tbl_bank_account`)
  - ✅ SPI 기반 LCD 화면 표시 (`draw_alert_box()`)
  - ✅ 프레임버퍼 공유 메모리 (`/dev/fb0`)

핵심 소스코드(DB와 기능)

(3) 입금 (`intel_bank_atm_deposit.c`)

- 기능:
  - 특정 금액을 계좌에 추가
  - 현재 잔액을 표시
- 사용된 기술:
  - ✅ SQL 업데이트 (`UPDATE tbl_bank_account SET money = money + X`)
  - ✅ SPI 기반 LCD 화면 업데이트 (`draw_alert_box()`)

(4) 출금 (`intel_bank_atm_withdrawal.c`)

- 기능:
  - 출금 요청 시 계좌 잔액 확인
  - 잔액 부족 시 오류 메시지 출력
- 사용된 기술:
  - ✅ SQL 업데이트 (`UPDATE tbl_bank_account SET money = money - X`)
  - ✅ 잔액 부족 검출 (`mysql_affected_rows()`)
  - ✅ LCD 경고창 출력 (`draw_alert_box()`)

핵심 소스코드(DB와 기능)

(5) 계좌 이체 (`intel_bank_atm_transfer.c`)

- 기능:
  - A 계좌 → B 계좌로 금액 전송
  - 잔액 변경 후 화면에 반영
- 사용된 기술:
  - ✅ SQL 트랜잭션 (`UPDATE sender, UPDATE receiver`)
  - ✅ UI 터치 입력 (`handle_touch_event()`)
  - ✅ 멀티스레딩 (`pthread_create()`)

결론 및 정리

- ✅ POSIX Thread를 활용한 ATM UI & 터치 이벤트 처리
- ✅ MariaDB와의 연동을 통한 금융 거래 지원
- ✅ SPI 기반 디스플레이 처리로 UI 출력 최적화
- ✅ 멀티프로세싱 (`execl()`)을 활용한 개별 기능 분리

핵심 소스코드(카메라)

카메라의 주요 구성

- 디바이스 /dev/video0 (V4L2 카메라)
- 해상도 176x144 (YUYV 포맷)
- 프레임 레이트 10 FPS
- 녹화 시간 60초 단위 연속 녹화

Int capture_to_raw_yuv()

- cctv_image_file 폴더를 생성 capture_to_fb() 에서 변환된 yuv 파일을 저장

Int capture_and _encode()

- yuv 파일을 FFMPEG 를 이용하여 MP4 파일로 저장
- 저장하는 과정에서 포크를 하고 파이프를 통해 읽고 쓰도록 하였습니다.

Int generate_filename()

- 변환하는 과정에서 파일 이름을 생성

카메라 구현 기능

1. 프레임버퍼를 통한 화면 출력
2. YUV to RGB 변환 처리
3. YUV 포맷으로 임시 저장
4. FFMPEG으로 MP4 변환
   (영상 스트림을 다양한 형태로 기록하고 변환하는 오픈소스 프로그램)
5. 날짜/시간 기반 파일명 생성
6. ATM 기기별 구분 저장
   (cam_ATM1_YYYYMMDD_HHMMSS.mp4)
7. YUV 파일은 계속해서 녹화 중이니까 쌓이지 않게 1분마다 지워주었습니다.

핵심 소스코드(서버와 클라이언트)

서버의 주요 구성

- 메인 스레드 port: 26000 통해 새 클라이언트가 접속 할 때마다 전용 스레드를 생성
- 접속된 스레드 데이터 수신, 파일 저장 처리
- 정리 스레드 60초 간격으로 디렉토리 검사 120초 이상 된 파일 자동 삭제

클라이언트 주요 구성

- 디렉토리 모니터링 지정된 디렉토리 60초 간격 검사 -> 새 파일 발견 시 전송 시작
- 파일 전송 파일명 전송, 데이터 청크 전송, 전송 후 로컬 파일 삭제

[Step 1: 파일명 전송]
클라이언트 → 서버: {파일명}₩n
서버 → 클라이언트: "OK"

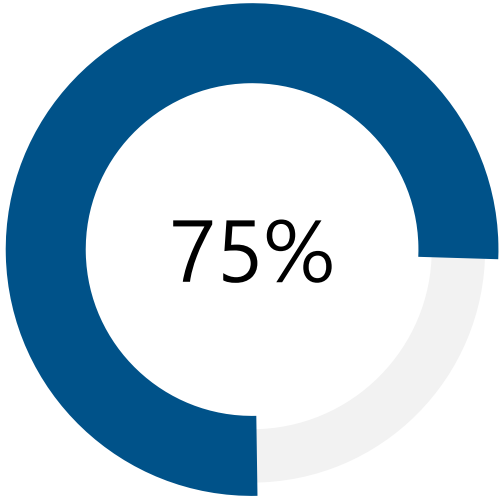
[Step 2: 파일 데이터 전송]
클라이언트 → 서버: [1024바이트 청크]
(반복 전송)

[Step 3: 전송 완료]
클라이언트: shutdown(SHUT_WR)
서버 → 클라이언트: "OK" or "ERROR"

Part 6,

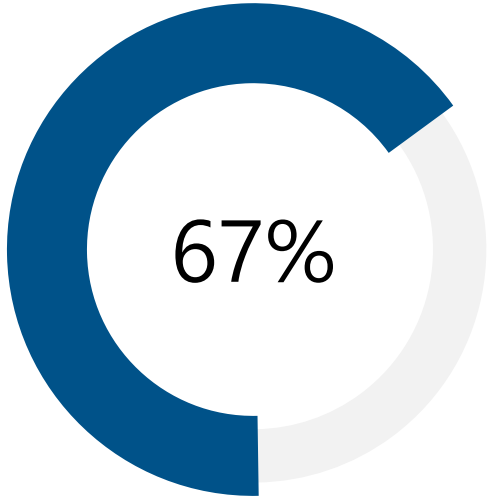
개발 결과 검토

현재 진행 상황



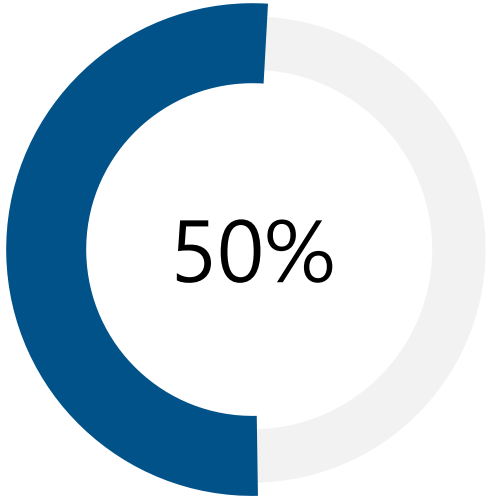
ATM 개발

전체 진행률



서버 구축 강화

서버 구축 및 보안 진행률



제품화

제품화 진행률

추후 개발 요소

휴대화

배터리 내장으로 휴대성 향상

게임 추가

은행 시뮬레이션 게임으로의 발전

장치 추가

I2C ,SPI를 이용하여 카드리더기,
적외선 센서 등의 하드웨어 추가

Part 7,

개발 후기

개발 후기

S

- LCD터치 구현을 통한 터치 작동 원리 이해와 리눅스 개발 경험
- 파트 별로 나눈 작업이 원활히 진행됨

Y

- 하드웨어의 동작 원리와 LCD 터치의 좌표 값을 이용한 구역 설정 이해
- DB 연동에 대한 이해와 리눅스의 사용 용도를 복기하는 계기가 됨

O

- 모든 팀원이 각자의 역할을 잘 수행하여 큰 문제 없이 마무리할 수 있었음
- 포크, 스레드와 같은 프로세스 제어의 활용처에 대해 좀 더 이해

H

- 스레드에 대한 개념과 통신에 대한 개념을 이해
- 코드 리팩토링에 대한 중요성을 깨달음

감사합니다.

Q.