## II. 상세 요구사항

아래 사항을 참고하여, 상세 요구사항을 분석해 보시오.

- 카드, 버스, 검사원 ID는 다음과 같은 포맷으로 사용한다.
  카드ID : CARD_001, CARD_002, CARD_003, …
  버스ID : BUS_001, BUS_002, BUS_003, …
  검사원 ID : INSP_001, INSP_002, INSP_003, …
  Validator ID : 이 시스템에서는 고려하지 않는다.

- 모든 입력은 콘솔로 받는다.

- 검사원의 ID와 Password가 들어있는 INSPECTOR.TXT파일은 이미 Server로부터 전송 받았다고 가정한다. '../CLIENT/ 폴더'에 저장되어 있는 파일을 사용한다 (상대경로를 사용하여 File을 읽어야 함). ID는 원본 그대로, Password는 SHA256 해시값으로 변환되어 저장되어 있다.
  INSPECTOR.TXT에 저장되어 있는 내용은 다음과 같다. (PASSWORD 원본은 들어있지 않음)
  ID와 Password는 고정길이이며 빈 칸 하나로 구분되어 있다.

| 검사원 ID (8BYTE) | Password HASH값 (64BYTE) | Password 원본 |
|---|---|---|
| INSP_001 | 03AC674216F3E15C761EE1A5E255F067953623C8B388B4459E13F978D7C846F4 | 1234 |
| INSP_002 | E12E115ACF4552B2568B55E93CBD39394C4EF81C82447FAFC997882A02D23677 | ABCD |
| INSP_003 | F08B6EDAB3B27AA05F3AE28397A6C1E21FE023484E0390ED6DD69A991CF190AE | AB1234 |
| INSP_004 | DB2E7F1BD5AB9968AE76199B7CC74795CA7404D5A08D78567715CE532F9D2669 | 4567 |
| INSP_005 | E0BEBD2281999342581486 6B62701E2919EA26F1370499C1037B53B9D49C2C8A | ABC123 |
| INSP_006 | EE415ECB101105C32411C0CDD984B1473DE0CC351DB45613E518C584ACB793BD | PASS1234 |
| INSP_007 | 435C554A2E9CD54D2D3431B8AF2B5D7BA740C64F1DCA92B7AF8A76B05D484EF3 | QWERTY |
| … | … | … |

- Login ID와 Password를 입력 받아서, 위 파일에 ID와 Password 쌍이 존재하면 Login 성공이고 그렇지 않으면 실패이다.

- Login 성공 시 'LOGIN SUCCESS'를 출력하고 다음 입력을 대기한다. 실패 시 'LOGIN FAIL'을 출력하고 계속 입력을 받는다.

- Login 성공 후 '버스ID'를 입력 받는다. 버스ID를 입력 받은 시점을 검사 시작 시각으로 한다.

- 버스ID를 입력 받은 후에는 Card 정보를 입력 받고, 해당 정보를 분석하여 결과를 파일로 저장한다. Card 정보는 검사 완료 전 까지 계속 입력이 가능하게 한다.
  • Validator에서 교통카드를 인식시켰을 경우의 Data를 콘솔화면에서 다음과 같은 포맷으로 입력한다. (고객 인터뷰 표1 참고)
  [카드ID(8)][버스ID(7)][승차/하차 코드(1)][최근 승차시각(14)]
  ex) CARD_001BUS_001N20171019093610
  CARD_002BUS_003F20171019143610

• 검사 결과 파일은 다음 폴더와 파일 이름으로 저장한다. (상대경로 사용)
폴더 : ../검사원ID/ (폴더가 없으면 생성해야 함)
파일명 : 검사원ID_검사시작시각.TXT
ex) ../INSP_001/INSP_001_20171023164832.TXT
• 검사 결과 파일의 내용은 다음과 같이 저장한다.
[검사원ID]#[버스ID]#[카드데이터]#[검사 결과 코드]#[검사시각]
ex) INSP_001#BUS_001# CARD_001BUS_001N20171019093610#R1#20171023164905
INSP_001#BUS_001# CARD_001BUS_001N20171019093610#R2#20171023165015
• 검사 결과 코드값은 다음과 같이 사용한다. (검사 판정 우선순위는 R1, R2, R3, R4 순서임)
R1 : 정상
R2 : 버스ID 정보가 다름
R3 : 승차 정보 없음
R4 : 승차 시각(3시간) 초과

- 해당 버스에서의 검사가 완료 되었다는 의미로 카드데이터 대신 'DONE'을 입력하여 검사를 완료한다. 새로운 버스에 탑승하여 검사를 할 수 있도록 다시 '버스ID'를 입력 받고 검사를 진행할 수 있다.

- 'DONE'을 입력하여 검사를 완료한 상태에서 새로운 버스ID를 입력하는 대신 'LOGOUT'을 입력하면 결과 파일을 Server로 전송하고 Validator 프로그램을 종료한다.

- Validator와 Server는 Socket (TCP/IP) 통신을 한다.
• LOGOUT시점에 Validator에서 저장한 결과 파일들의 파일명과 내용을 Server로 전송한 후에, 전송 완료한 파일들을 '../BACKUP'폴더로 이동시킨다.
• Server에서는 Validator로부터 전달받은 파일명을 사용하여 파일 내용을 '../SERVER/'폴더에 저장한다. (상대경로 사용)

- Manager 단말기와 Server는 HTTP 통신을 한다. 각 Manager 단말기에는 Manager ID가 부여되어 있다.
• Manager 단말기에서 'REPORT'를 요청하면 Server가 Validator로부터 수신한 파일들 중 요청 날짜의 파일들을 분석하여 Report를 작성하고 Manager 단말기로 Report를 전송한다. Report 작성 시 Report ID를 생성한다.
• Server는 Manager 단말기의 요청에 따라 Report를 전송한 후 Manager 단말기로부터 처리완료('FINISH') 신호를 기다린다. 만약 Timeout 시간동안 처리완료 신호가 오지 않는 경우, Report가 정상적으로 처리되지 않았다고 판단하고 해당 Report를 파일로 저장하고 전송 실패 LOG를 남긴다. Report 파일명은 'TIMEOUT_ReportID.TXT'의 형태로 하고, '../SERVER/REPORT'폴더에 저장한다.
• Report의 내용은 다음과 같다. 검사원ID 순서로 출력하고, 각 필드의 구분은 빈 칸으로 한다.
* [검사원ID] [검사 카드 수] [비정상 카드 수]
ex) INSP_001 20 11
INSP_002 30 8
INSP_003 5 3
…

• Manager 단말기와 Server의 통신 프로토콜은 다음과 같다.
(Manager 단말기는 ManagerTerminal.exe로 주어지고, 활용법은 구현 & Test 문서 참조)

※ REPORT 요청

- URI : GET http://127.0.0.1:8081/REPORT/<Manager Id>/<Date>
- 동작 :
  1) <Date>에 해당하는 Report를 생성하여 응답
  2) Timeout시간(5초) 동안 'REPORT 처리완료' 혹은 'REPORT 처리 실패' 신호를 기다림.
  3) Log파일에 Report 정보 추가 : YYYY-MM-DD HH:MM:SS.FFF | Manager ID | "REPORT" | Report ID
- 응답 Body : JSON 문자열 형식
  1) 응답할 Report가 존재할 경우
  {"Result":"Ok","ReportID":"<Report ID>","Report":"<Report>"}
  2) 응답할 Report가 존재하지 않을 경우 (<Date>에 해당하는 Report 생성이 불가능할 경우)
  {"Result":"No Report"}

※ REPORT 처리 완료

- URI : POST http://127.0.0.1:8081/FINISH/
- 요청 Body : JSON 문자열 형식 {"ManagerID":<Manager ID" ,"Report ID":<Report ID>}
- 동작 :
  1) Report를 파일로 저장 : "₩SERVER₩REPORT₩FINISH_[Report ID].TXT"
  2) Log파일에 처리완료 추가 : YYYY-MM-DD HH:MM:SS.FFF | Manager ID | "FINISH" | Report ID
- 응답 Body : {"Result":"Ok"}

※ REPORT 처리 실패

- URI : POST http://127.0.0.1:8081/FAIL/
- 요청 Body : JSON 문자열 형식 {"ReportID":<Report ID>,"ManagerID":<Manager ID"}
- 동작 :
  1) Report를 파일로 저장 : "₩SERVER₩REPORT₩FAIL_[Report ID].TXT"
  2) Log파일에 처리 실패 라인 추가 : YYYY-MM-DD HH:MM:SS.FFF | Manager ID | FAIL | Report ID
- 응답 Body : {"Result":"Ok"}

- REPORT 요청 처리 후 Timeout 시간동안 Manager 단말기로부터 FINISH, FAIL이 전송되지 않으면 다음과 같이 처리
  - 동작 :
    1) Report를 파일로 저장 : "₩SERVER₩REPORT₩TIMEOUT_[Report ID].TXT"
    2) Log파일에 Timeout 라인 추가 : YYYY-MM-DD HH:MM:SS.FFF | Manager ID | "TIMEOUT" | Report ID

  LOG파일은 다음 형태로 출력한다. (분단위로 파일 생성)
  - 파일 경로/이름 : ".₩SERVER₩LOG₩LOG_HHMM.TXT"
- Server 콘솔에서 'QUIT'을 입력하면 Server 프로그램을 종료한다.
- Server에 여러 Validator가 동시에 접속하는 상황은 고려하지 않는다.
- Server에 여러 Manager 단말기가 동시에 접속하는 상황을 고려한다.
- 입력값은 모두 정확하다고 가정한다.