**3. Specific Requirements**

* 1. **Functional Requirements**
     1. **Data Dictionary**

[Table 1] User

| **Field** | **Key** | **Constraint** | **Description** |
| --- | --- | --- | --- |
| id | PK | Not Null | User id |
| name |  | Not Null | User’s name |
| school\_number |  |  | User’s school number |
| position |  | Not null | User’s position  (student / instructor) |

[Table 2] Lecture

| **Field** | **Key** | **Constraint** | **Description** |
| --- | --- | --- | --- |
| id | PK | Not Null | Lecture id |
| type | FK | Not Null | Ref. Seat arrangement type |
| name |  | Not Null | Lecture name |
| instructor |  | Not Null | Instructor name |
| time |  | Not Null | Lecture time |
| place |  | Not Null | Lecture place |

[Table 3] Bus

| **Field** | **Key** | **Constraint** | **Description** |
| --- | --- | --- | --- |
| id | PK | Not Null | Bus number |
| type | FK | Not Null | Ref. Seat arrangement type |
| starting\_point |  | Not Null | Starting point of bus |
| departure\_time |  | Not Null | Departure time of bus |

[Table 4] Seat\_Arrangement

| **Field** | **Key** | **Constraint** | **Description** |
| --- | --- | --- | --- |
| type | PK | Not Null | Seat arrangement type |
| arrangement |  | Not Null | Seat arrangement position |

[Table 5] Bus\_Reservation\_Status

| **Field** | **Key** | **Constraint** | **Description** |
| --- | --- | --- | --- |
| id | PK/FK | Not Null | Ref. / Bus |
| type | FK | Not Null | Ref. Seat arrangement |
| arrangement | FK | Not Null | Ref. Seat arrangement |
| status |  | Not Null | Seat reservation status |

[Table 6] Lecture\_Reservation\_Status

| **Field** | **Key** | **Constraint** | **Description** |
| --- | --- | --- | --- |
| id | PK/FK | Not Null | Ref. Lecture |
| type | FK | Not Null | Ref. Seat arrangement |
| arrangement | FK | Not Null | Ref. Seat arrangement |
| status |  | Not Null | Seat reservation status |

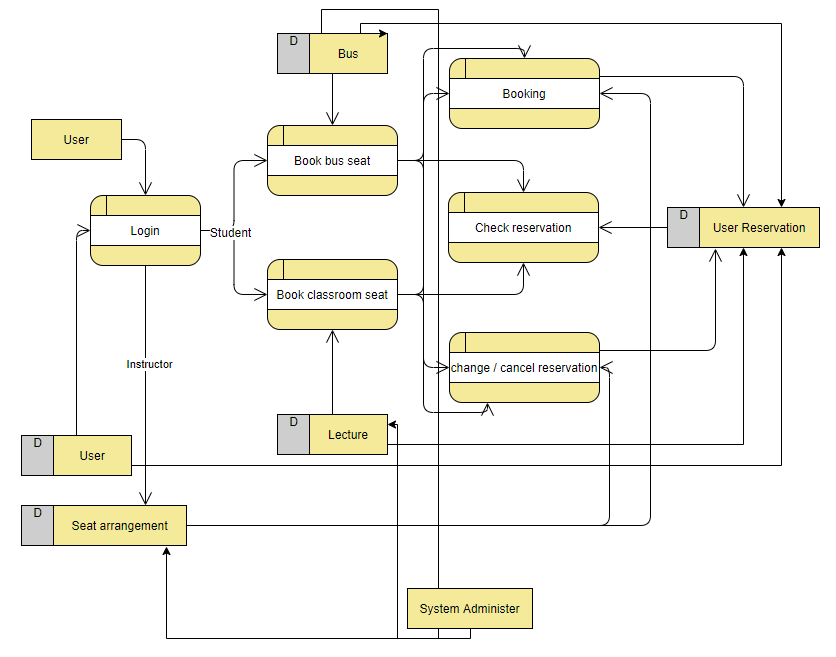
[Table 7] Lecture\_Taken

| **Field** | **Key** | **Constraint** | **Description** |
| --- | --- | --- | --- |
| id | PK/FK | Not Null | Ref. User |
| lecture\_id | FK | Not Null | Ref. Lecture |
| time | FK | Not Null | Ref. Lecture |
| place | FK | Not Null | Ref. Lecture |

[Table 8] User\_Reservation

| **Field** | **Key** | **Constraint** | **Description** |
| --- | --- | --- | --- |
| id | PK/FK | Not Null | Ref. User |
| lecture\_bus\_id | FK | Not Null | Ref. Lecture and Bus |
| seat\_position |  | Not Null | Reserved seat position list |

* + 1. **Data Flow Diagram**

[Figure 3] Data flow diagram

* 1. **Performance Requirements**

다음의 requirements는 예상을 기반으로 작성되었으며 향후에 변동될 수 있습니다.

**3.3.1. Static numerical requirement**

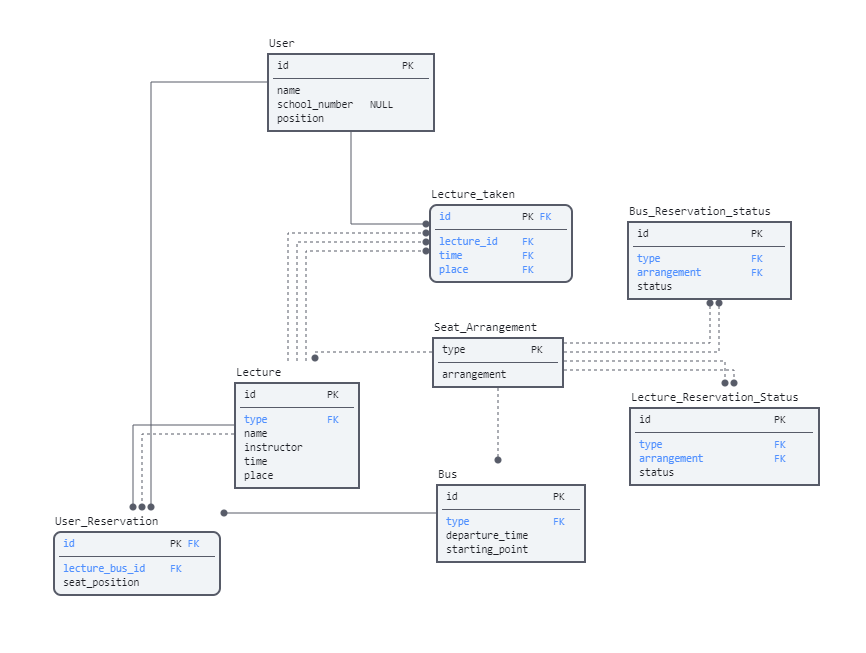
* System은 하나의 terminal만 지원합니다.
* 한 기기에서 한 명의 user만 system을 이용할 수 있습니다. 여러 기기에서 동시 연결은 불가능합니다.
* System은 500명의 user가 동시에 접속해도 원활하게 동작해야 합니다. System은 동시에 30000개의 데이터를 처리할 수 있어야 합니다.
* 시스템이 원활하게 동작하기 위한 모바일 최소 사양은 1GB RAM과 512MB의 저장공간, Android 6.0입니다.

**3.3.2. Dynamic numerical requirement**

* Reservation의 변동이 있다면 10초 이내에 database가 update되어야 합니다.
* User의 reservation 가능 여부는 database와 비교하여 5초 이내에 확인되어야 합니다.
* Login 과정은 5초 이내에 완료되어야 합니다.
* 좌석 배치 UI는 3초 이내에 load되어야 합니다.
* Reservation 현황은 3초 이내에 load되어야 합니다.
  1. **Logical Database Requirements**

모든 data는 Firebase의 Firestore를 이용하여 저장됩니다. Data는 json objects로 저장되며 objects에 들어가는 Data type은 Integer, String, List가 있습니다.

저장된 data는 예약, 예약 확인, 예약 변경 및 취소 등에 이용됩니다.

****[Figure 2] Entity Relationship Diagram

* 1. **Design Constraint**

Firebase가 Apache lisence를 따르고 있으므로 system 역시 Apache lisence를 따라야 합니다. Apache liscence의 경우 source code의 공개 의무나 상업적 이용에 제한을 두고 있지 않으므로 system이 apache liscence를 갖고 있음을 밝혀야 합니다.

* 1. **Standards compliance**

System 내부의 모든 program에서 사용되는 file name, variable name, method name, class name 등과 source code structure는 Google developers에서 배포하는 Kotlin Style Guide를 따릅니다. Firestore를 이용할 때 data structure는 평면적으로 구성합니다.