Application Specification Draft

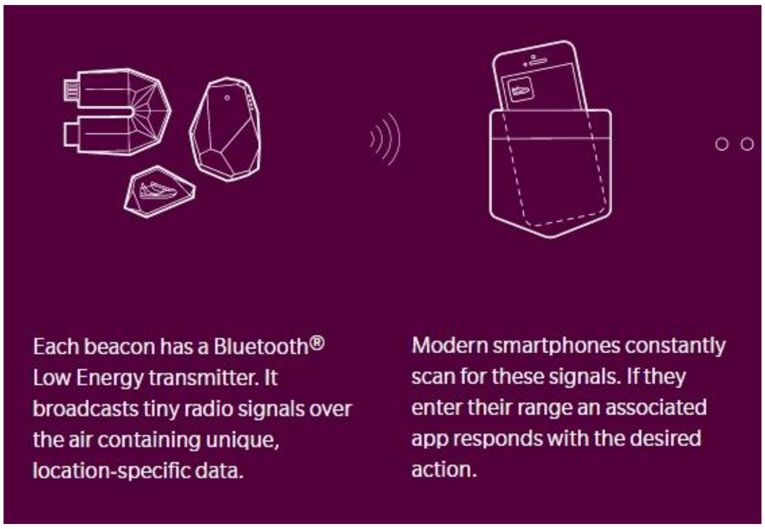
(arranged our thoughts so far)

1) Aims

- Automatic attendance checking mobile application using BLE beacon
- Name : yet to be discussed
- Users : SNU students and instructors
- Operating systems : Android and iOS
- Development tool : Fuse

2) Backend Details

1. Beacon


Each beacon has a Bluetooth® Low Energy transmitter. It broadcasts tiny radio signals over the air containing unique, location-specific data.
Modern smartphones constantly scan for these signals. If they enter their range an associated app responds with the desired action.

<from estimate.com>

- Use a phone device(either android or apple) as a signal transmitter

  How to build an android beacon transmitter:

  a. Open source code from Android Beacon Library
     https://github.com/AltBeacon/android-beacon-library/blob/master/src/main/java/org/altbeacon/beacon/BeaconTransmitter.java

  b. An app called 'Locate Beacon' from Google Play
     https://play.google.com/store/apps/details?id=com.radiusnetworks.locate&hl=en

- Different identifiers mean different courses

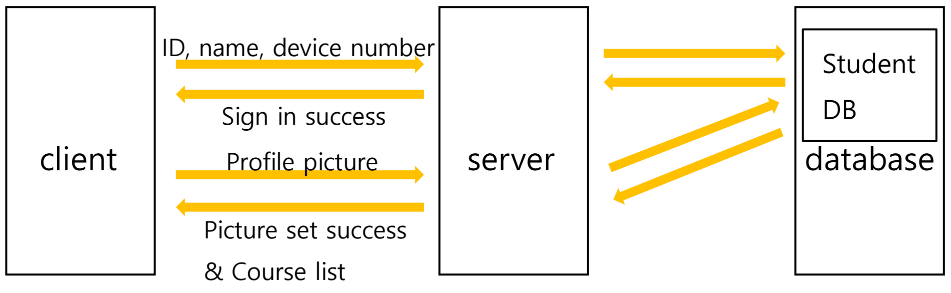
We just use a single phone device as a beacon transmitter. We can experiment whether the app works correctly for different courses & classrooms by changing the identifiers(minor bits) of a beacon. If needed, use more devices.

- Bluetooth automatically on without executing the app??

2. Server

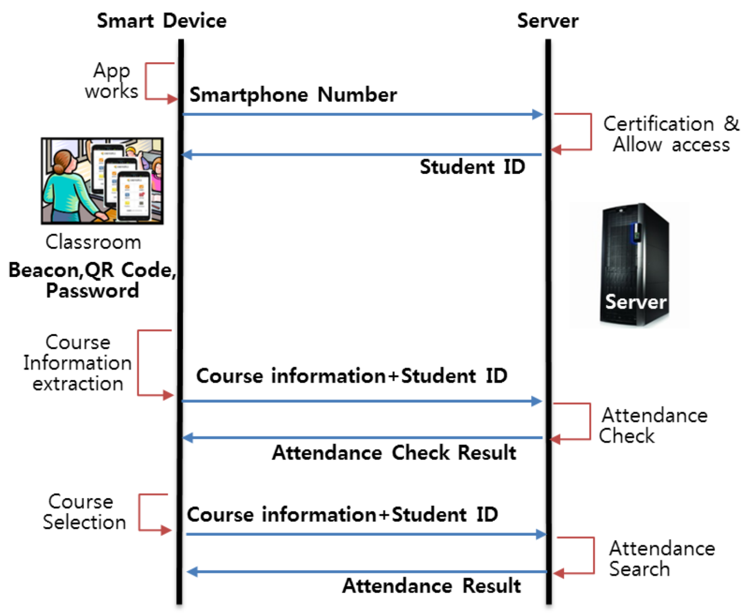
- Use Django(Django REST framework)/Couchbase server/…
- What a server does:

a. Authentication



Determine whether user is an SNU member or not by ID and name.
Save user's profile picture.
Load course list from student DB.

b. Attendance checking



<from: International Journal of Multimedia and Ubiquitous Engineering>

Extract course information from a beacon signal by accessing beacon DB.

Scan the student list of that course and find out whether the student exists in DB.

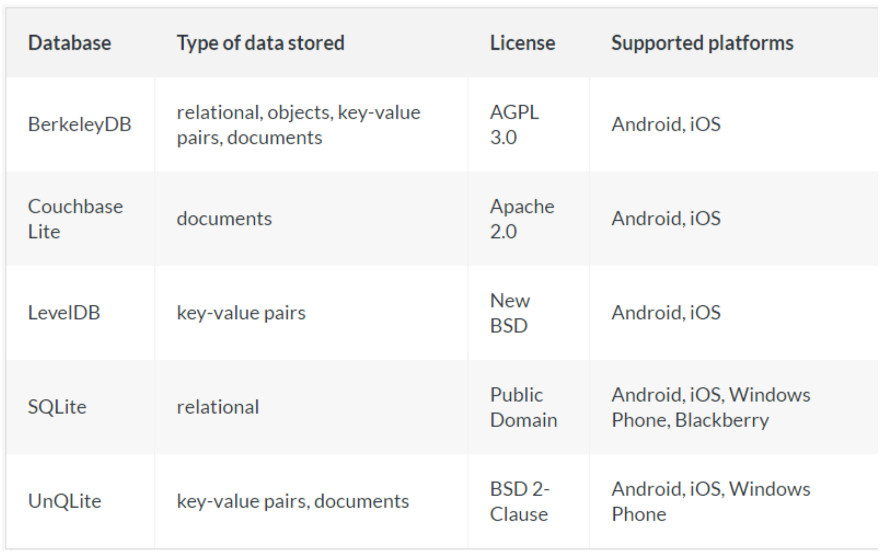
Send attendance results (attendance + time) to the client.

c. Return attendance history

Access to student DB and attendance DB of that student.

3. Database

- Should be decided which database system to use



Database	Type of data stored	License	Supported platforms
BerkeleyDB	relational, objects, key-value pairs, documents	AGPL 3.0	Android, iOS
Couchbase Lite	documents	Apache 2.0	Android, iOS
LevelDB	key-value pairs	New BSD	Android, iOS
SQLite	relational	Public Domain	Android, iOS, Windows Phone, Blackberry
UnQLite	key-value pairs, documents	BSD 2-Clause	Android, iOS, Windows Phone

<5 popular databases for mobile: from https://www.developereconomics.com/five-popular-databases-for-mobile>

- Course DB

  5 to 10 courses (This is an initial aim and can be changed)

  Fields : course name(key), classroom, student list, time

- Student/Instructor DB

  Fields : ID(key), name, course list, (device number), attendance history

- Attendance DB

  Each student's attendance history

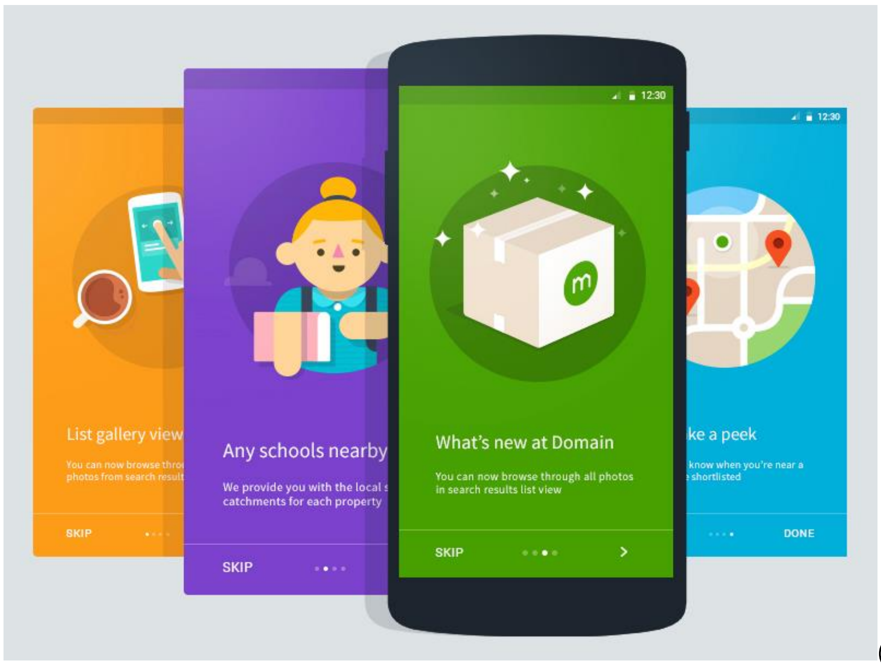
  Fields : course name(key), date, attendance, IN/OUT time

- Beacon DB

  Fields : beacon ID(key), course name, classroom

3) General flow and implementation

1. Introduction page: Show briefly how to use this app


List gallery view
You can now browse thro
photos from search result
SKIP
Any schools nearby
We provide you with the local s
catchments for each property
SKIP
12:30
What's new at Domain
You can now browse through all photos
in search results list view
SKIP
ke a peek
know when you're near a
shortlisted
12:30
DONE
(example)

- App on -> Name page -> Introduction starts

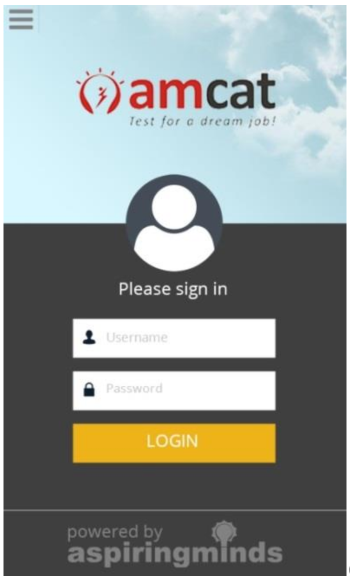
- **Page flow**
  a. Load your course list by sign up
  b. Go into your classroom with your phone and it's done!
  c. Check your attendance history
  d. See the details by clicking the course button
  e. Get started

- **Implementation**

  Use fuse tool example

  https://www.fusetools.com/examples/onboarding-with-pagecontrol

2. Sign in page: Users get authentication



(example)

- Click 'Get started' button -> Sign in page shows up
- There's no specific sign-up process. User can just sign in with student/instructor ID and name. The number of the device which sends user-inputted information to the server will implicitly work like a password.

  (Or, since there are some sign up frameworks on the internet, it will not be hard to implement sign up and sign in process separately. In this case, user signs up with student/instructor ID, name, username and password and signs in with username and password. Device number will also be sent to the server.)

  (Or email authentication)

  학교 메일로 인증/ login + password
- Once the user signs in, the app stays signed in unless the user manually signs out.
- If a user who's not a member of SNU(not in student or instructor DB) tries to sign in, an error message pops up.
- If user signs out before the class finishes, record the time he/she signs out as the OUT time.
- When a user signs out and wants to sign in again, it will be the same process.
- The user is not allowed to sign in with different IDs on the same device. This is to prevent some proxy attendance(대리 출석) for friends.

  (Or, since it's processed to be OUT when user signs out, we may not need to deal with this problem.)

- **Authentication flow**

  a. User inputs his/her student/instructor ID and name.
     We use name to prevent non-SNU member arbitrarily inputs a student/instructor ID.

  b. The information(ID, name + device number) goes to the server.

  c. The user-inputted ID and name are compared with IDs and names in DB.

  d. If there's an ID matching with the user-inputted one, store the device number with it. Else pop up an error message.

     d-1) If a device number already exists under the ID, find out whether the number exists for another ID.

     If not, replace the number with a new one.(the user changed his/her phone)

     If so, pop up an error message.(to prevent a proxy attendance)

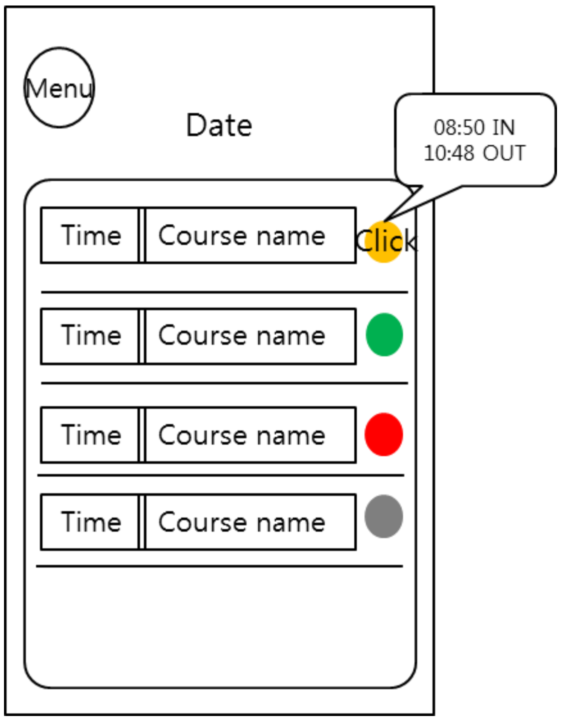
  e. If the matched ID is a student's, execute in student mode. Else, instructor mode.

  f. Pop up succeed message and load user's course list.

- **Implementation**

3. Main page(Student): Show user's own daily attendance and attendance history

- **Daily page**



a. Shows today's courses and attendances

b. Use colors to represent the attendance(출석 현황).

Like red for absent, yellow for late, green for attended, gray for classes that have not started yet.

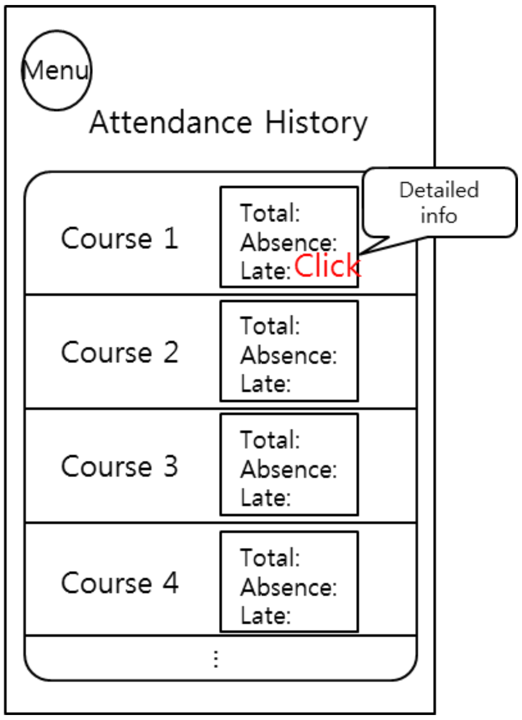
(+ More creative idea!)

c. If the user clicks the circle that

represents his/her attendance, it will show the time he/she gets in and out.

d. Yesterday and tomorrow(yet to be discussed)

- **History page**



a. Shows user's attendance history

total number of classes, number of absences, number of lates, …

b. If the user clicks the course button, he/she can check more detailed information.

When he/she was absent, when he/she was late…

This can be implemented differently by using tabs per courses.

- **Attendance check flow**

a. A beacon is installed inside the classroom.

b. When user gets into the range of the beacon,

The app catches the beacon signal and sends the identifier numbers to the server.

c. Server extracts course information from the signal using beacon DB.

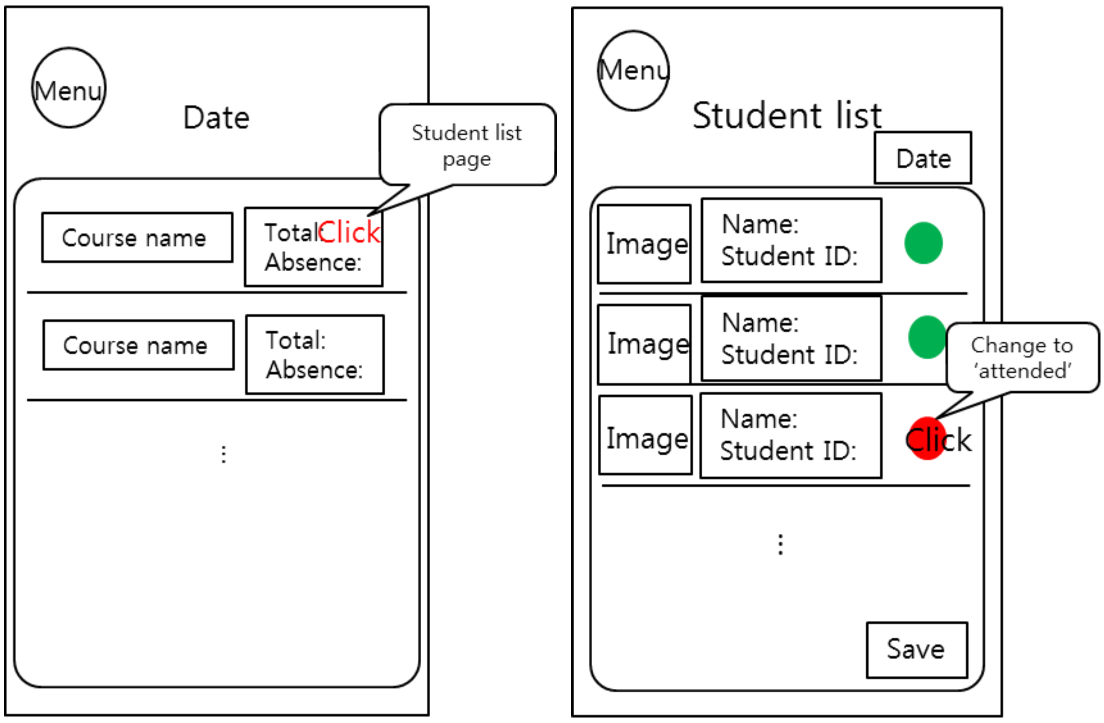
d. Server updates requesting client's attendance DB and return.

e. When it comes to OUT…

일정 거리 이상 되면 out처리

4. **Main page(Instructor): Show student list with their attendance(student profile picture) and overall history**

- **Daily page**

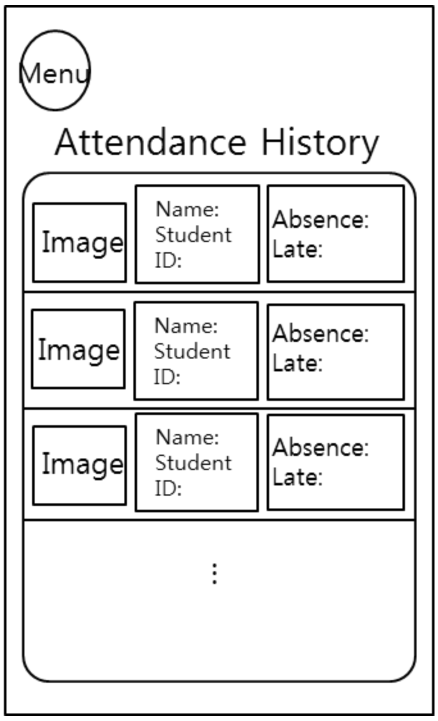


a. Show today's courses and students' attendance (total number of students, number of absent students, …).

b. If user clicks the course button, it will lead to another page in which user can see the list of students who are taking the course. Each student is represented with a picture that he/she set when signing in, name, and student ID. User can check students' attendance by seeing the color of the circle.

   Here, user can manually change a student's attendance by clicking the circle and saving. If user clicks the circle, it will toggle its color and change its state. In the same way, user can manually check the attendance of the whole students on this page.

- **History page**



a. Show each student's attendance history. Like student list page from the daily one, students are represented by their picture, name and student ID. Number of absences and lates by the time user checks this page are seen on the page.

b. 내보내기 기능?

- Implementation