Smart Brochure
(paperless brochure using beacon)

Hyeonsu Lim
Information System in HYU
Email: tatto_hs@naver.com

Jaemook Kang
Information System in HYU
Email: kjm8475@naver.com

Kiseong Kim
Information System in HYU
Email: rltjd1231@naver.com

***Abstract*—When you go to the art museum, you will find out several things to help you see the exhibition better, such as brochure, program books, and audio-guide, etc. For the big size of exhibitions supported by big art gallery or of famous artist, there would be no problem to prepare the goods mentioned before. However, there are a lot of artists who are trying to open an exhibition in small art gallery and students who are preparing the exhibition for the graduation, and they have a lot of problems to possess those kinds of goods. The purpose of this software project is to help them. We can provide many kinds of IoT services, such as the explanation of the exhibition, explanations of each art, audio-guide, and so on, by using beacon and the mobile application.**

Keywords — beacon; bluetooth; iot; museum; brochure; smart phone; application;

Roles	Name	Task description and etc.
User	Kiseong Kim	Use the mobile application and get some data about exhibition
Customer	Kiseong Kim	Purchase this software service and offer the data about exhibition
Software Developer	Jaemook Kang	Develop the mobile application and back-end server
Development manager	Hyeonsu Lim	Manage team and project and make the every plan of the process of project

I. Introduction

If you have only a little interest, you will find many art galleries and museums easily. There are over 100 exhibition halls in Seoul, which means that a lot of exhibitions we can enjoy are opening every day. In order to help the audiences enjoy these exhibitions better and feel better, those exhibitions are providing many items that will help people understand the exhibition better. The audiences need to pay extra costs to buy or rent those goods.

However, not all the exhibitions provide those services. In the case of famous artists and big art museums, a lot of people will visit there and see the works, so there will be many people who will pay extra costs to get the chance to inspect better. Therefore, famous artists and big art museums can make extra incomes by making lots of guide goods and sell them. On the other hand, lets think about obscure artists and the university students who are preparing the exhibition for the graduations. They can hardly provide those items for their audiences. The first reason is the cost. The cost to make audio guide, brochure, and program books are not low, so the obscure artists and the students cannot afford it.(about 2,000,000 won)

The second reason is the difference of peoples interests. Practically, it is really hard for the small exhibition halls to have peoples attention. Even though they spend money more and produce the goods for their exhibitions, there will not be that many people who will pay extra money to buy or rent the goods.

We thought that people who are trying to open small exhibition and students who are preparing an exhibition doesnt want the extra income but the interests of people. We wanted to provide them more chances for the amateur artists to introduce themselves and to appeal their works to people. Therefore, we want to fulfill their requirements through SMART BROCHURE. Artists can provide good service to their audiences with less cost, and audiences dont have to pay extra money and get the chance to use many services that artists want to provide, only by installing an application. Though there is a similar application, named Jeonsi bogo, this service is only for the big exhibitions, so we still need to develop new system.

II. Index

III. Requirements

A. Requirement for The Environment for Using Beacon

1. Bluetooth module

- Users need to have Bluetooth 4.0 or higher module.

2. Smartphone OS

1) ios 7 or higher

2) Android 4.3 or Higher

3) OSX mavericks 10.9

B. Requirement for Server

1. How to send information

1) The beacon installed in the gallery will send the id code to the smartphone which has the application, and the smartphone will send that id code and the customer information to the server.

2) Lastly, the server will send the appropriate data, which is decided by combining the gallerys information and the customer information, to the smartphone.

2. Needs

We need to develop a back-end server that stores the information and judge the id code sent by beacon.

C. Requirement for User

1. Accuracy : Users want more precise sensor when they use beacon technology

2. Unimportant data : Users dont want information which is not necessary

3. Push alarm

1) Push alarm is popped up on the users smartphone when passes by an exhibition.

2) The user can get some information by push the yes button.

4. Exhibition list

- If push the button, you can see list of exhibitions you watched. Latest exhibition is located in top of the list.

5. Map

1) Map provides a course how to see the exhibition.

2) If users push one of the mutton, Smart Brochure gives users the map of the exhibition.

6. Location of work in the exhibition

- In the map, users can see some buttons which indicates work name and where works are.

7. Work button

- If users push button on the map, they can get screen which has information about the work.

8. Work picture

- In the information screen, picture of the work is located left-top.

- Users can check on whether explanation corresponds to the work by picture.

9. Text box1

- Text box1 is located next to work picture. There are work name, artist name, and techniques in the box.

10. Text box2

- Text box2 has explanation of the work. If explanation is so long, uses can use scroll technique.

11. Voice button

- In the text box2, users can use voice button. If users push the button, they can hear explanation of the work.

12. Previous exhibition

- This application can saved data about previous exhibition.

13. Delete function

- If users want to delete previous exhibition information, they can delete the data.

14. Reporting problem

- When use Smart Brochure, users can find some problem. In this situation, users can report this problem to developer.

IV. Development Environment

A. Choice of software development platform

1. Which platform and why? (e.g., Windows, Linux, Web, or etc.)

1) Windows for android application developing

2) Mac OS for iOS application developing.

2. Which programming language and why?

1) java for android application developing

Java is a general-purpose computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. As of 2015, Java is one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by James Goslingat Sun Microsystems (which has since been acquired by Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

2) objective-C and Swift for iOS application developing.

Objective-C is a general-purpose, object-oriented programming language that adds Smalltalk-style messaging to the C-programming language. It is the main programming language used by Apple for the OS X and iOS operating systems, and their respective application programming interfaces (APIs), Cocoa and Cocoa Touch. Objective-C's features often allow for flexible, and often easy, solutions to programming issues. Delegating methods to other objects and remote invocation can be easily implemented using categories and message forwarding. Swizzling of the isa pointer allows for classes to change at runtime. Typically used for debugging where freed objects are swizzled into zombie objects whose only purpose is to report an error when someone calls them. Swizzling was also used in Enterprise Objects Framework to create database faults. Swizzling is used today by Apple?s Foundation Framework to implement Key-Value Observing.

Swift is a multi paradigm, compiled programming language created by Apple Inc. for iOS and OS X development Swift is designed to work with Apple's Cocoa and Cocoa Touch frameworks and the large body of existing Objective-C code written for Apple products. Swift is intended to be more resilient to erroneous code ("safer") than Objective-C, and also more concise. It is built with the LLVM compiler framework included in Xcode6, and uses the Objective-C runtime, allowing C, Objective-C, C++ and Swift code to run within a single program, but its proprietary nature may hinder Swift's adoption outside the Apple ecosystem.

3) JSON

JSON is an open standard format that uses human-readable text to transmit data objects consisting of attribute?value pairs. It is used primarily to transmit data between a server and web application, as an alternative to XML. Although originally derived from the JavaScript scripting language, JSON is a language-independent data format.

B. Provide a cost estimation for your built. (including any purchase of software/hardware)

1. cost for server : 1 year for free. And after 1year, there will be additional prices. We predict maybe about 1,000 people will use our service, and DAU(Daily Activity User) will be 300 around. So we will use t1micr instance (AWS), and its prices are about $30 per month. So maybe there will be additional 360$ per year.

2. cost for beacon : We will use the RECO beacon. Reco beacon is authorised by iBeacon. Its prices are 229,000 (10 pieces).

3 cost for developer :

C. Provide clear information of your development environment. (e.g., version of software, OS version, your computer resources)

1. iOS develop

1) Mac : OS X Yosemite ver 10.10.1

2) iOS : iOS 8.3

2. Android develop
1) Windows : Windows7 ultimate
2) Android OS : Android 4.4 kitkat

D. Using any commercial cloud platform (e.g., Amazons EC2) is definitely a BONUS.

1. We will use Apache Web Server because it is the world's most widely used web server software. As of June 2013, Apache was estimated to serve 54.2percent of all active website and 53.3percent of the top servers across all domains. (The most important reason is we have already apache web server.)

Apache Web Server has some features:
- easy and fast customizing using module.
- It can handle many traffic easily.
- It can control web server more delicately
- It is tested enough, so it is very stable.

2. Software in use
Any existing software or algorithm in use? (doing a similar task as your proposal; provide a proper reference if there is any)

1. Trello :

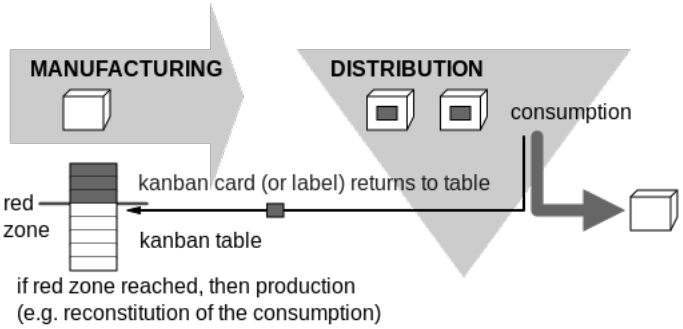

Fig. 1. Kanban Paradigm

Trello is a free web-based project management application. Trello uses the kanban paradigm for managing project. Kanban is a scheduling system

Fig. 2. trello

for lean and just-in-time (JIT) production.

Projects are represented by boards, which contain lists (corresponding to task lists). Lists contain cards (corresponding to tasks). Cards are supposed to progress from one list to the next (via drag-and-drop), for instance mirroring the flow of a feature from idea to implementation. Users can be assigned to cards. Users and boards can be grouped into organizations.

There is a little similar software in Korea, GO But this software service is only for big exhibition so small exhibition artist or students cant use that services.

3. Task distribution (If you want, you can provide this later at the next phase - design)

Which member is responsible for what?

V. Specifications

A. Modeling for Specifications

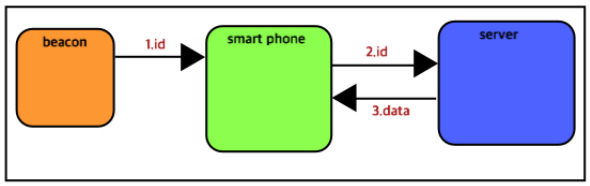

Fig. 3. basic structure

Beacon sends a specific ID value to the smart phone, when smart phone comes into it?s signal area. Then smart phone application recognizes this ID value and sends this value to server. Server which has this ID value check the location of beacon. After that, server sends the information or data about exhibition to smart phone.

B. Prototype for Specifications

Our application is divided into two parts. One is server side with Amazon Web Service EC2 and Ruby on Rails. The other part is client side acting at the smart phone. Client side, smart phone application, is structured by objective-C (iOS application), and java(Android application)

C. Specification for front-end application pages

0) BLE seraching outside the application

Fig. 4. Information Page02

[BLE]

In our application, there is the service class [SearchBLE.java] which is searching the [BLE]. This service searches the BLE, if the bluetooth module on the smartphone is on. BLE(Bluetooth low energy) is a wireless personal area network(PAN) technology. It is designed and marketed for applications in the healthcare, fitness, security, home entertainment industries, and [beacon].

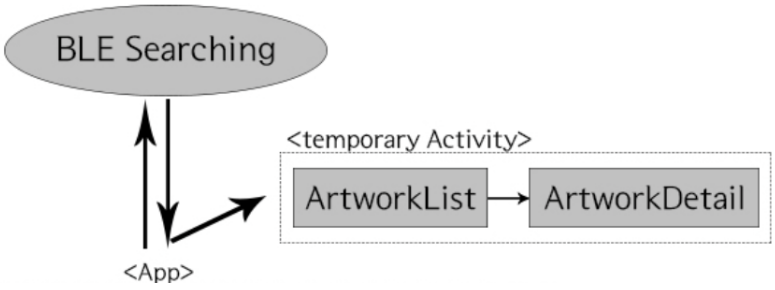

Fig. 5. how temporary activity is operating

If user let bluetooth module [on], smartphone will be searching BLE automatically. That is, it is searching beacon signal. If it finds beacon signal, smartphone get [beacon id]. And smartphone sends this beacon id to server, then server sends to smartphone the all data about exhibition which is stored at that beacon id. After, every data about exhibition is downloaded to smartphone. This downloaded data is showed through [Temporary

Activity].

The first page has the brief explains for exhibition and the list of artworks (similar with [My History Page02]). Among these artworks, if you pick one, you can show the details of that artworks such like photos and detail explanations (similar with [My History Page 03]).

You have to know this [Temporary Activity] is totally different page with main application. This is [only] temporary. This activity cannot access to main application, and also main application cannot access this activity neither. This activity is only for showing the data from the server. And in main application, not every data showed at temporary activity is stored. Only the name of exhibition and the downloaded date are stored.

1) My history page

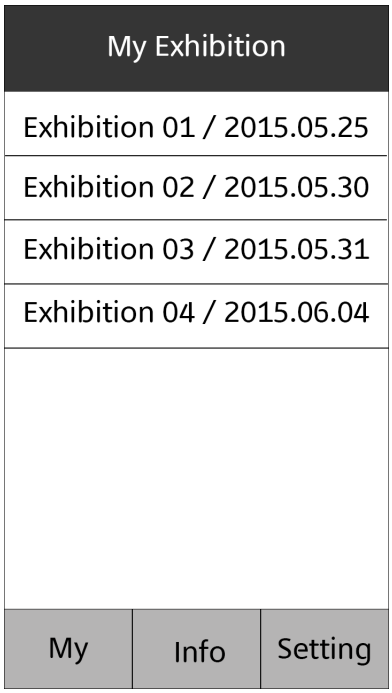

Fig. 6. BLE

[My History Page01]
This is the first page of application. And you can access this page by tab menu under the display. There is the list of exhibitions which user have already seen.

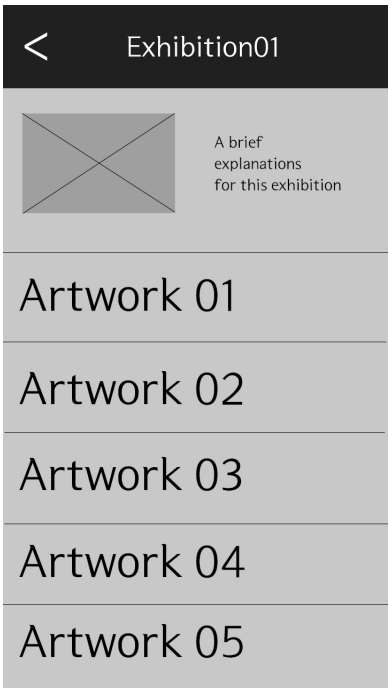

Fig. 7. My History Page02

[My History Page02]
This is the detail page of the exhibition01. To access this page, applications has to communicate with the back-end server. Server will send the data of Exhibition 01 which the user already downloaded by beacon communication at that Exhibition. On Navigation bar, there is the name of the exhibition. Under the navigation bar, the left side of the first cell, there is main image of exhibition. And Next to the main image, the right side of the first cell, there is the brief explanation for this exhibition. There will be the information of this exhibition such like the theme of exhibition, the name of exhibition center, address of exhibition center. Below First cell, there is the list of artwork which is displayed in this exhibition. user can see the detail information about the artwork such as image of artwork, text explanation of artist, or voice-audio explanation.

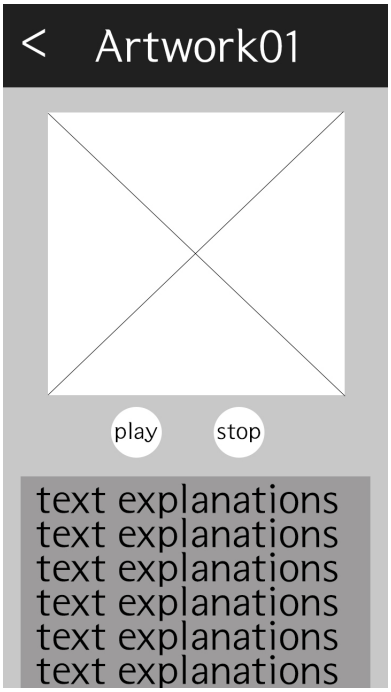

Fig. 8. My History Page03

[My History Page03]

This is the detail page of artwork. On the Navigation bar, there is the name of artwork. Under the Navigation bar, there is the Image of the artwork(If user touch the small image, the pop up window will appear, and user can see the big size image). Below the artwork image, there is the play and stop button. This button is for voice-audio explanation. Voice-audio explanation is not for every artwork. We offer the voice-audio explanation only for the artwork that artist want, and artwork that artist offer the voice-audio explanation data. So if there is the voice-audio explanation, there will be play and stop buttons. And if there is no voice-audio explanation, the play and stop buttons will not exist. Under the Image and buttons, there is the text explanations that artist offer.

2) Information page

Fig. 9. Information Page01

[Information Page01]

user can access the Information page by touching tab menu button under the screen. When user touch the [Info] button under the display, every data is downloaded from server. This page is for noticing the exhibition. There is the list of the exhibitions which is on going now or which will be started. user can check the detail information about the exhibition that user like by touching the name of the exhibition.

Fig. 10. Information Page02

[Information Page02]

This page shows the detail information of the exhibition. user can move to the exhibition list page by back button on the navigation bar. On navigation bar, there is the name of the exhibition. Below the navigation bar, user can find the brief information about the exhibition such as the main image of the exhibition, theme of the exhibition, and map or address of exhibition.

3) Setting page

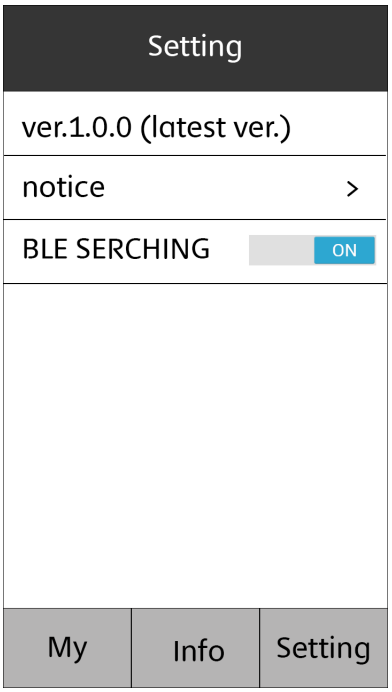

Fig. 11. Setting Page

[Setting Page]

In Setting page, there will be the additional function. For example, there will be the on/off button that control the searching BLE. If the user makes the button-state [on], smartphone will search beacon signal and get beacon id (explained and [0)BLE searching outside the application]). And if user makes the button-state [off], smartphone will not search any beacon signal outside the application. And there will be another board for inform the latest version or notice etc.

D. Specifications for Server

1) Filezilla

We use the [filezilla] to upload the data such like image, and text for exhibition. FileZilla is free-open source cross platform. It consists of filezilla client and filezilla server. It can used at windows, mac os, and linux.

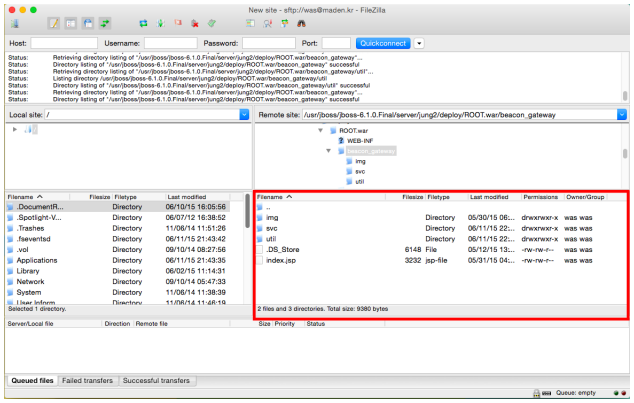

Fig. 12. filezilla01

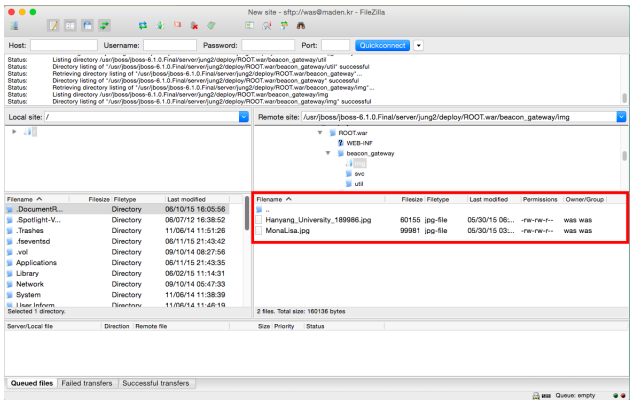

Fig. 13. filezilla02

If artist want to upload the data of his exhibition, he needs to contact us and send the data to us first. After receiving data from artist, we can upload the exhibition data to our server. Then, data is stored at our web server, and wail the signal calling it.

2) Pgadmin

We use [pgadmin] for make and control the database.

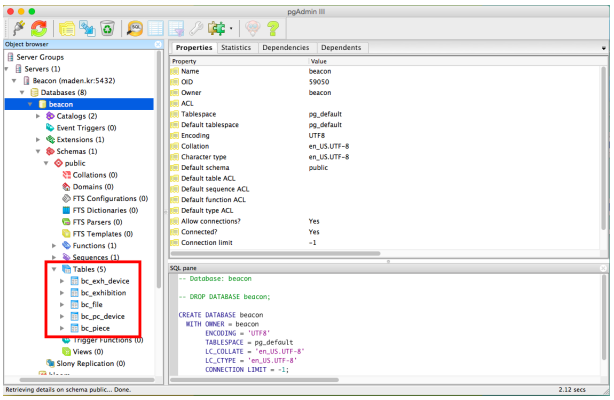

Fig. 14. pgadmin

In first picture, there are tables of our database. We made 5 tables.

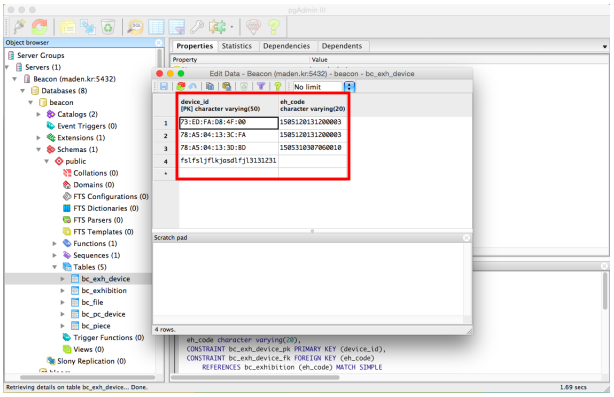

Fig. 15. pgadmin table01

First, [bc.exh.device] table is for beacon id [PK](This beacon is at the entrance of the exhibition). At this table, we make the exhibition code and match to beacon id. And we make the exhibition code to.

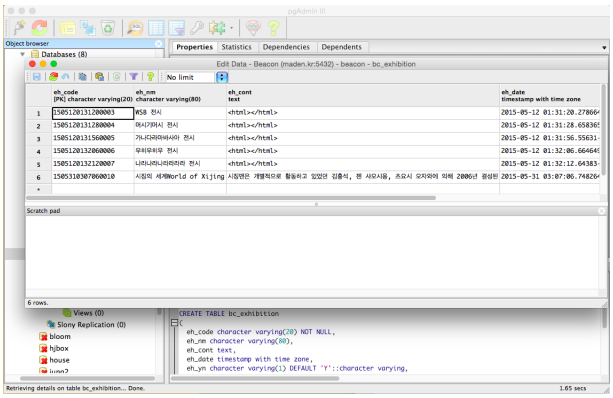

Fig. 16. pgadmin table02

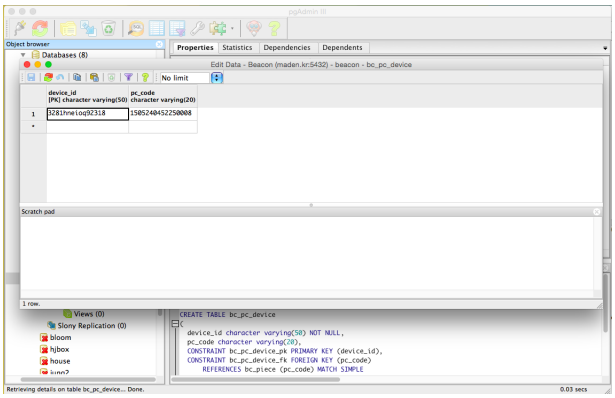

Fig. 18. pgadmin table04

Second, [bc.exhibition] table is for brief information about exhibition. Exhibition code is [PK]. This table is connected to [bc.exh.device]table.

[bc.pc.device] table is for beacon id[PK]. But this beacon is located nearby artwork. If user goes to artwork, then application recognize the beacon and send to server, this table.

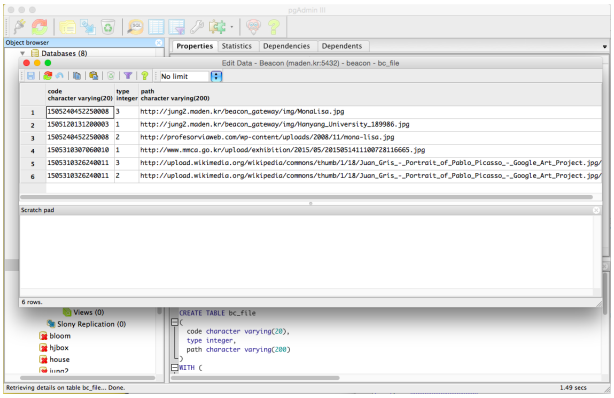

Fig. 17. pgadmin table03

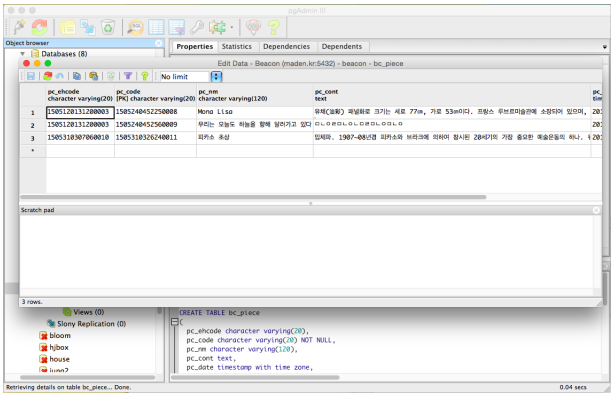

Fig. 19. pgadmin table05

[bc.file] table is for image files. There is the 3 type integers. Type integer 1 is for exhibition main image. It is used at My history page02, and Information page02. Type integer 2 is for thumbnail image of artwork. It is used at My history page 02. Type integer 3 is for the image of artwork. It is used My history page 03.

[bc.piece] table is for detail information about artwork. Artwork has its own code[PK].

VI. Architecture Design and Implementation

A. Overall architecture

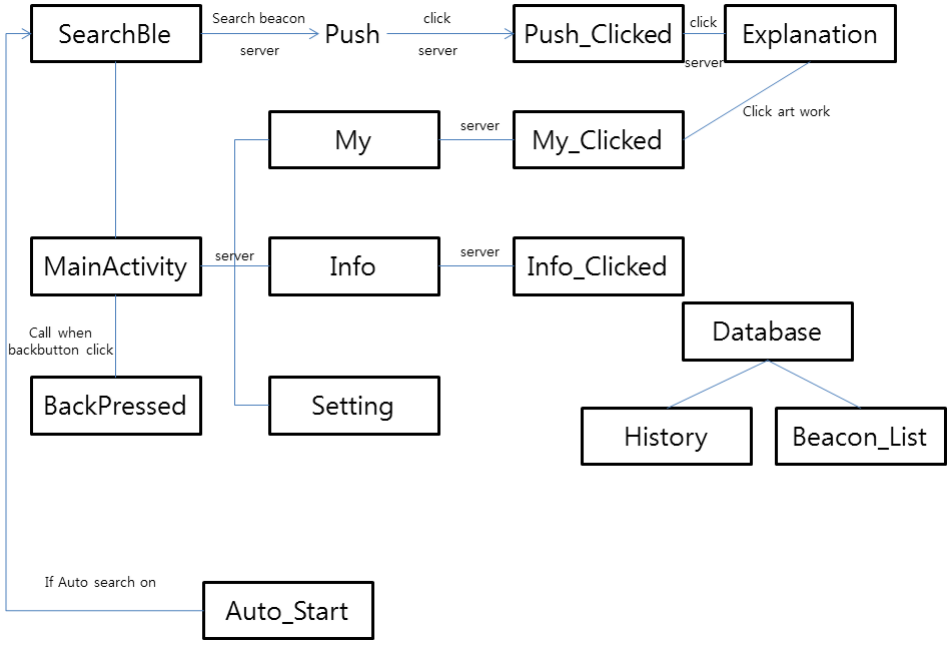

Fig. 20. Overall architecture

B. Directory organization

directory	File names	Module name in use	Etc
SmartBrochure/app/src/java/com/example/jay/smart_brochure	Splash.java	class Splash	
SmartBrochure/app/src/java/com/example/jay/smart_brochure	SearchBLE.java	class SearchBLE	
SmartBrochure/app/src/java/com/example/jay/smart_brochure	MainActivity.java	class MainActivity	
SmartBrochure/app/src/java/com/example/jay/smart_brochure	My.java	class My	
SmartBrochure/app/src/java/com/example/jay/smart_brochure	My_Clicked.java	class My_Clicked	
SmartBrochure/app/src/java/com/example/jay/smart_brochure	Push_Clicked.java	class Push_Clicked	
SmartBrochure/app/src/java/com/example/jay/smart_brochure	Explanation.java	class Explanation.	
SmartBrochure/app/src/java/com/example/jay/smart_brochure	Info.java	class Info	
SmartBrochure/app/src/java/com/example/jay/smart_brochure	Info_Clicked.java	class Info_Clicked	
SmartBrochure/app/src/java/com/example/jay/smart_brochure	Setting.java	class Setting	
SmartBrochure/app/src/java/com/example/jay/smart_brochure	Database.java	class Database	
SmartBrochure/app/src/java/com/example/jay/smart_brochure	History.java	class History	

Fig. 21. Directory organization

C. Code analysis

1) Splash

-purpose : To show a cover page during the program is loading on the device.

-functionality : Splash is a kind of loading screen. The function of splash is gaining time to get data which is necessary for program. Additionally, we can promote our application name during splash time.

-location of source code : Smart-Brochure/appsrc/java/com/example/jay/smart_brochure/Splash.java

-class components

a. 'onCreate' method is implemented firstly in the activity class like main() method in JavaSE. In the onCreate, setContentView shows basic layout which is used to compose main page. By using postDelayed in handler, this method can do function how long loading page appear.

b. onCreateOptionsMenu method does initializing option menu of activity.

c. In onOptionsItemSelected method, Handle action bar item clicks here. The action bar will automatically handle clicks on the Home/Up button, so long as you specify a parent activity in AndroidManifest.xml.

-how/why you used it : If you want to implement some program, the program needs time to get source which is vital for program. Splash does function which gains time getting data.

2) SearchBle

-purpose : To search beacon which uses Bluetooth Low Energy(BLE).

-functionality : SearchBLE is implemented on application background. Usually almost classes inherit Activity class, but SearchBLE inherits Service class because searching function isnt seen in the smartphone screen but implemented background. This service searches the BLE devices around the user and check if the BLE address is in our database or not. If the address is in our database and it is not the one that user already visited, this service will send the BLE address to the server to get the information of the exhibition and give the push alarm to the user to get the brochure.

-location of source code : Smart-Brochure/appsrc/java/com/example/jay/smart_brochure/SearchBle.java

-class components

a. BroadcastReceiver uses the onReceive() method to get the state of the Bluetooth. If the devices Bluetooth is on, this starts the Searching BLE service, by using Timer instance, which makes the period of searching time and so on. If the Bluetooth becomes off, this stops searching BLE devices by calling cancel() method in the Timer instance and scanLeDevice() method by giving the parameter as false. It is used to realize operation when application completed specific task.

b. Timer is basic class in Android. By using Timer class, we can set schedule when searching starts and how long search beacon. When we use Timer instance, Android appreciate '1 = 1 millisecond'. Therefore, if we want to set searching time 3 seconds, we have to put value 3*1000. In our code, we set it as schedule(search, 10*1000, 20*1000); which meanse it will start searching after 10 secs, and for 20secs, and it will rotate.

c. onLeScan's main function is using Handler, which means it will use the Threads. In the run() method, which is working on the thread, the application will check the database if there is a valid address that our service is using, and if two are matched, this will check whether the user has visited to this exhibition by searching the History table in the Database class. Only when there is no history about the beacon address, it will send the address and the code to the server to get the information about the exhibition and will give the

device push-alarm which will help the user to get the brochure. When we search beacon, there are so many beacons even if we don't need. If we dont figure out whether it is valid or not, we have to send all beacon data which are not necessary to server. That can provoke wasting of data use. By using this method, we can send appropriate beacon data to server. Additionally, if an appropriate beacon data sent to server, the method will save the address in the temporary ArrayList¡String¿ variable to prevent sending same address to the server over and over again. If we dont do this, push alarm will come continuously while audiences watch the exhibition.

d. sendId() method is the core part of this service which sends the data to the server and get the data from server to the application. Using HashMap¡String, Object¿ type of variable, we put the beacon id and make it to JSONObject, which server wants as the data type to get from the device. After that, by using DefaultHttpClient, the device will get the data from the server as JSONArray type, so we will cast the datatype that we want to use, and make the push alarm by using NotificationCompat class.

-how/why you used it : By using Auto_Start class, we made SearchBLE service start when the devices booting is completed, only if the auto-search option is on. Therefore, if the user set up the option as on, the user doesnt need to turn on the application to get the brochure. By only turning on the Bluetooth, this service will search the beacons around the user and send the data to the server and will notify the user by push alarm. This is one of the core concept of IoT..

3) MainActivity

-purpose : To show main page and constitute tab view which consist of My, Info, and Setting for customer.

-functionality : MainActivity is the body of SmartBrochure. Most classes are used in this class. The main function is showing main page which is made up of several components.

-location of source code : Smart-Brochure/appsrc/java/com/example/jay/smart_brochure/MainActivity.java

-class components

a. turnOnBT method instructs whether to turn on or turn off Bluetooth to customers by dialogue. If Bluetooth is turned on before starting application, notification dialogue will not occur on the device.

b. getServiceTaskName checks if the SearchBLE service is now running or not. If it is not running, we will start SearchBLE service in order to search the beacons if the Bluetooth is on.

c. We overrode onDestroy() method. When the application destroys, we need to turn of the SearchBLE service too, if the auto-searching option is off. Therefore, we opened database in this method to check if the option is turn on or off.

-how/why you used it : MainActivity is foundation of application. We build the tab activity to make the tabs for the pages, My, Info, Setting. We made this class inherit TabActivity, but we will fix this to FragmentActivity because TabActivity is deprecated.

4) My

-purpose : My_Clicked shows the list of art work. The list includes name and image of each art work.

-functionality : First tab button on the main page and if you execute SmartBrochure, you can see my page firstly. If you click one of the list, list of the exhibition work appears.

-location of source code : Smart-Brochure/appsrc/java/com/example/jay/smart_brochure/My.java

-class components

a. sendId method does the same function as the sendId method in the SearchBLE. We get the

exhibition code and the BLE address that is already saved in the Database when the user visitied the exhibition, and send them to the server to get the exhibition brochure from the server. After server gets the beacon data and the valid exhibition code, the servier will send the exhibition data (exhibition name, list of work, image and so on) to SmartBrochure and we go to the My_Clicked class to show the brochure.

b. customAdapter's main function is giving data to list view. In the customAdapter, getView method put the data to the list.

c. We made this listview get the OnItemClick-Listner, so when user touch the list, the page will go to the My_Clicked page which gets the data of the exhibition.

-how/why you used it : My page gets the list of history from the Database and put them to the listview. We thought that listview is the best way to show the list to the user. Between My and My_Clicked, the data communication between the device and server occurs.

5) My_Clicked

-purpose : My_Clicked shows the list of art work of the exhibition that the user visited before. Most of the function is as same as Push_Clicked.

-functionality : My_Clicked class must need to communicate with server in order to get the brochure from the server. In the My page we sent the id and exhibition code to the server, we will get the brochure data from the server in My_Clicked page and will show the brochure.

-location of source code : Smart-Brochure/appsrc/java/com/example/jay/smart_brochure/My_Clicked.java

-class components

a. ImageLoaderConfigurationbition is used to show the image from the image URL we get from the server. In the application, image file is not anywhere. However, in order to use work image, we can use server. When we need some image file, we get URL which has image from server. In other words, just borrow image file when we need. If we go out of the page which has image, image file is removed from application.

b. We use customAdapter and OnClickListener, and so on, which is used in other pages also.

-how/why you used it : We communicate with the server here again, because we dont want to save so many imagefiles to the application database, which will cause some problems of the storage of the device. Instead, we chose to communicate with the sever again, so we gets all of the data we use here from the server.

6) Push_Clicked

-purpose : Push_Clicked also shows the list of art work. The list includes name and image of each art work.

-functionality : Push_Clicked class must need to communicate with server in order to get the brochure from the server. In the Push page we sent the to the server, we will get the exhibition code and brochure data from the server in Push_Clicked page and will show the brochure when user clicked the push alarm button which was sent before.

-location of source code : Smart-Brochure/appsrc/java/com/example/jay/smart_brochure/Push_Clicked.java

-class components

a. ImageLoaderConfigurationbition is used to show the image from the image URL we get from the server. In the application, image file is not anywhere. However, in order to use work image, we can use server. When we need some image file, we get URL which has image from server. The image URL or the Image file that we used here will never be stored in the database.

b. In the onCreate method, we get the data of

the brochure. Here, we need to save the beacons id and exhibition code in the database in order to keep the history of users exhibiting. We used History class to save the exhibition title, name, and the BLE address. After that, we use the addHistory method to save it in the database with SQLite.

-how/why you used it : Push_Clicked is very similar with My_Clicked. However, they are different in sending id value to server. My_Clicked is focused on existing exhibition information. In contrast, Push_Clicked is focused on new information from beacon search. While My_Clicked page sent the beacons id and exhibition code, in Push_Clicked page, we send only the beacons id that SearchBLE service has searched to the server.

7) Explanation

-purpose : Explanation class has function which gives the detailed description about art work. Users can read the detailed image and the explanation of the specific work that the user wants to see the detail.

-functionality : When you click one of the art works in the list(in the My_clicked or Push_Clicked), Explanation class is implemented. In this process, communication with server is very essential like My_clicked or Push_Clicked. Explanation also use image by using URL from server.

-location of source code : Smart-Brochure/appsrc/java/com/example/jay/smart_brochure/Explanation.java

-class components

a. In the Explanation class, onCreate method's main function is setting ImageView and TextView which will be used after getting data from the server. They are given from server. At the end of this method, we call the setExplanation() method to set the data.

b. setExplanation method uses some function like DisplayImageOptions and ImageLoaderConfiguration. By using them, we set the image from the imageURL, and the explanation of the artwork.

-how/why you used it : Explanation class is the most informative section in our application. Like My_Clicked, we use server to use adequate image, because image data is very important in exhibition information.

8) Info

-purpose : Info notifies several exhibitions information which is what is exhibiting now.

-functionality : Second tab button on the main page so if you execute SmartBrochure, you can't see Info page firstly. If you click Info button, list of several exhibition which are exhibition now appears. We get this information from the server, which means, if the user click on the Info tab button, the device will communicate with the server to get the information.

-location of source code : Smart-Brochure/appsrc/java/com/example/jay/smart_brochure/Info.java

-class components

a. In the Info class, sendId method is the important method because uses server like several other classes. Here, the device will just send the url Id and then the server will send the list of the exhibitions only.

b. Here, we also used ListView to show the list of the exhibitions which users can go to and get the brochure and see the artworks now.

-how/why you used it : This is just a basic page of ListView, to show the list of exhibitions our service is providing. We will get the list from the server when user tries to go to the info page. If the Internet is not on on the device, this page will show nothing beacause it will not get any data from the server. The reason why we made this page is that we thought that users might need the information of the exhibitions and can choose the exhibition that they want to go. They will choose

one or several of the exhibitions from the list and when they go to the exhibitions, they can get the brochure on their device.

9) Info_Clicked

-purpose : Info_Clicked shows exhibition name, location, date and brief explanation.

-functionality : Info_Clicked class also must need communication with server. If application sends the id value which is in the database, after that server sends information about the exhibition which is correct with id to application.

-location of source code : Smart-Brochure/appsrc/java/com/example/jay/smart_brochure/Info_Clicked.java

-class components

a. In setInformation, there are so many function to make Info_Clicked class. To show title, explanation, date and address, use text view. ImageLoaderConfigurationbition is also used for implanting image like My_Clicked. In the application, image file doesn't store anywhere. However, in order to use work image, we can use server. When we need some image file, we get URL which has image from server. In other words, just borrow image file when we need. If we go out of the page which has image, image file is removed from application.

-how/why you used it : Info_Clicked knows us information about exhibition that we can go. People who are interested in art want to kwon exhibition information which they can go nowadays. Also we add some data like exhibition image, brief explanation and so on. These information will satisfy user's needs.

10) Setting

-purpose : Setting is used to set automatic search by using Bluetooth and show the version of the application.

-functionality : By setting, users can choose whether searching the beacon devices automatically or not. If you want to get exhibition data when you hang around outside, ust set 'automatic search' on in the setting.

-location of source code : Smart-Brochure/appsrc/java/com/example/jay/smart_brochure/Setting.java

-class components

a. In the onCreate method, auto_search find signal from beacon which is around user. btn_onoff set that user turn on/off the automatic search function. At first, we get the on/off data from the database to show the user the current setting.

b. In the onCreate method, auto_search find signal from beacon which is around user. btn_onoff set that user turn on/off the automatic search function. At first, we get the on/off data from the database to show the user the current setting.

-how/why you used it : If user always turn on Bluetooth and turn on auto_search function, user's smartphone battery will be consumed rapidly. Therefore, we thought it is very important to set these functions on/off. This function can save users smartphone battery and satisfy the users preference.

11) Database

-purpose : To store the lists of beacons that our service is using/ will use, the history of the users own exhibitions that he/she has visited and got the brochure from it, and save the setting of automatic beacon searching preference.

-functionality : In the services and pages of our service, especially SearchBLE, My, and Setting, we need to use the database to keep the users preferences and the history of visiting the exhibitions.

-location of source code : Smart-Brochure/appsrc/java/com/example/jay/smart_brochure/Database.java

-class components

a. First, the database class inherits SQLiteOpenHelper to use the database with SQLite.

b. setExplanation method uses some function like DisplayImageOptions and ImageLoaderConfiguration. By using them, we set the image from the imageURL, and the explanation of the artwork.

c. init() method - when the user first downloaded this application, the device doesnt have any database, but we need the list of the beacons that we use. Therefore, when the application was on for the first time, call init() method in Database.java and this will make the list of the beacons and set the ONOFF preference 0, which means off, as default.

d. getOnoff() method - search the CheckOnOff table and get ONOFF row and return in order to check if the onoff setting is on or off.

e. editOnoff() method - get String value as the parameter and edit the ONOFF row in the CheckOnOff table.

f. getList() method - return the list of the S-Brochure table.

g. updateBeacons() method - using Cursor class and ContentValues class, update the beacons that out service use.

h. getBeaconse() method - return the list of the beacons that we are using, to identify the BLE address that is valid for this application.

i. addHistory() method - using the History class, add the beacons address and the name and code of the exhibition that user visited to the S_Brochure table.

j. getHistoryName(), getHistoryAddress(), and getHistoryCode() methods - three of these methods searches the DB and return the beacons address and the name and code of the exhibition that user visited with ArrayList¡String¿ type.

k. searchHistory() method - with the parameter of String type, search the DB to find if there is the same address in the DB.

-how/why you used it : We developed this database with SQLite. If some classes(activities) needs the database information, they can instantiate the Database class and use the methods that we have made so that they will be able to access to the Database and get the information they need and edit the database if it is needed. The database is essential to our Smart Brochure service, since we need to manage the history of the users visiting exhibitions; if they want to read the previous brochure, they dont need to go to the exhibition again if they have already visited. Also, the application needs to have the list of the beacons that it should search and send the information to the server, because there are a lot of BLE devices around us. Therefore, we had to give the application the list of beacons that it should send the information if one of them is sensed. Lastly, we have the function of auto-searching setting, so the database has the setting and can figure if the setting is on or off.

12) History

-purpose : To make it easier and flexible to keep the database.

-functionality : There are several methods to get the data and give the date to the caller. Other classes will instantiate this class and save data in the object, and will send this object to the database.

-location of source code : Smart-Brochure/appsrc/java/com/example/jay/smart_brochure/History.java

-class components

a. This class has getAddress(), setAddress(), getName(), setName(), getCode(), setCode() method to make the flexibility higher.

-how/why you used it : We just made some

methods to get and set the data to and from the database. We will use this class by making it instance in other classes. Save the data in the instance and transfer the instance to the Database. Without this method, it becomes more complex to keep the database.

.

VII. Use Cases

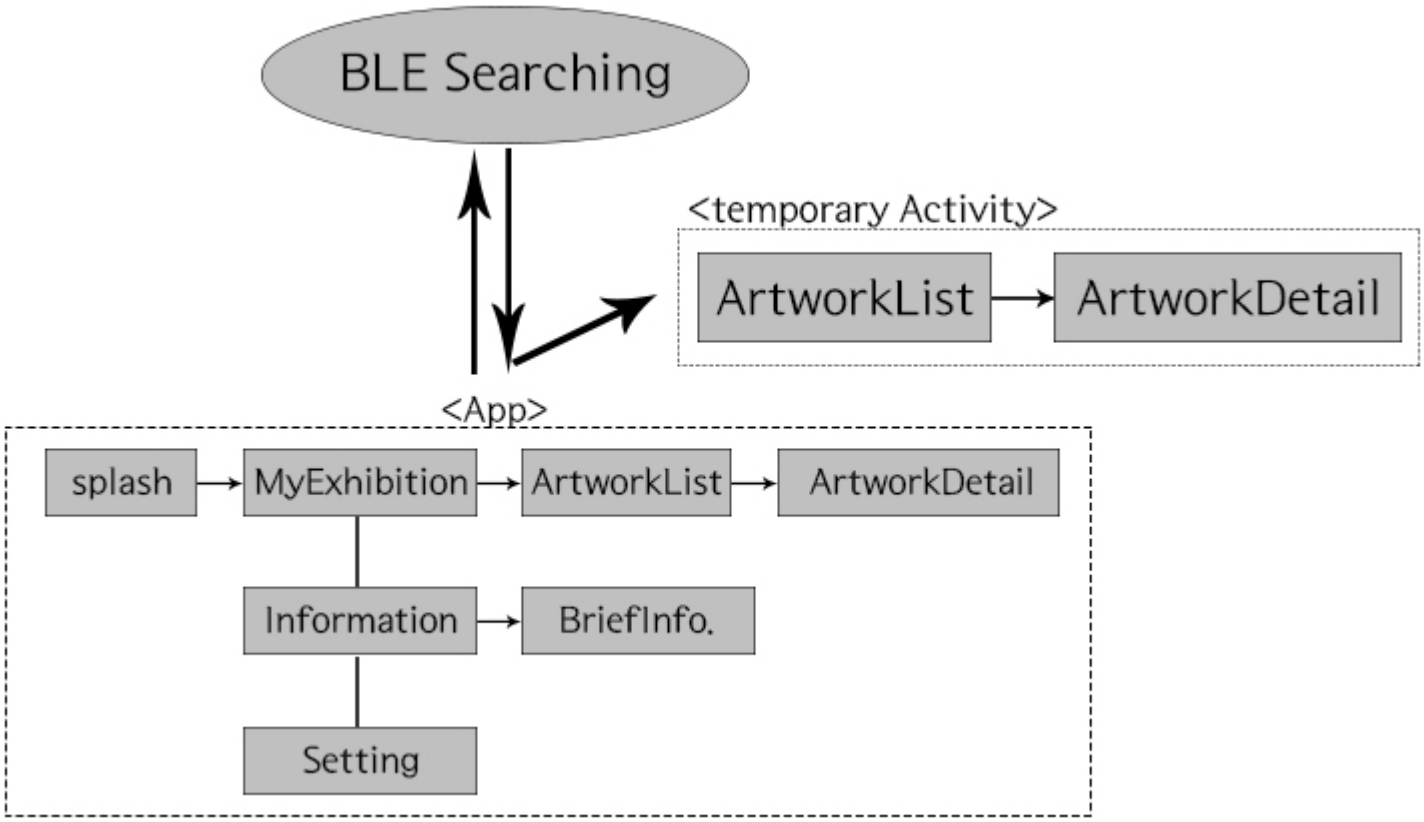

Fig. 22. Application Flowchart

.

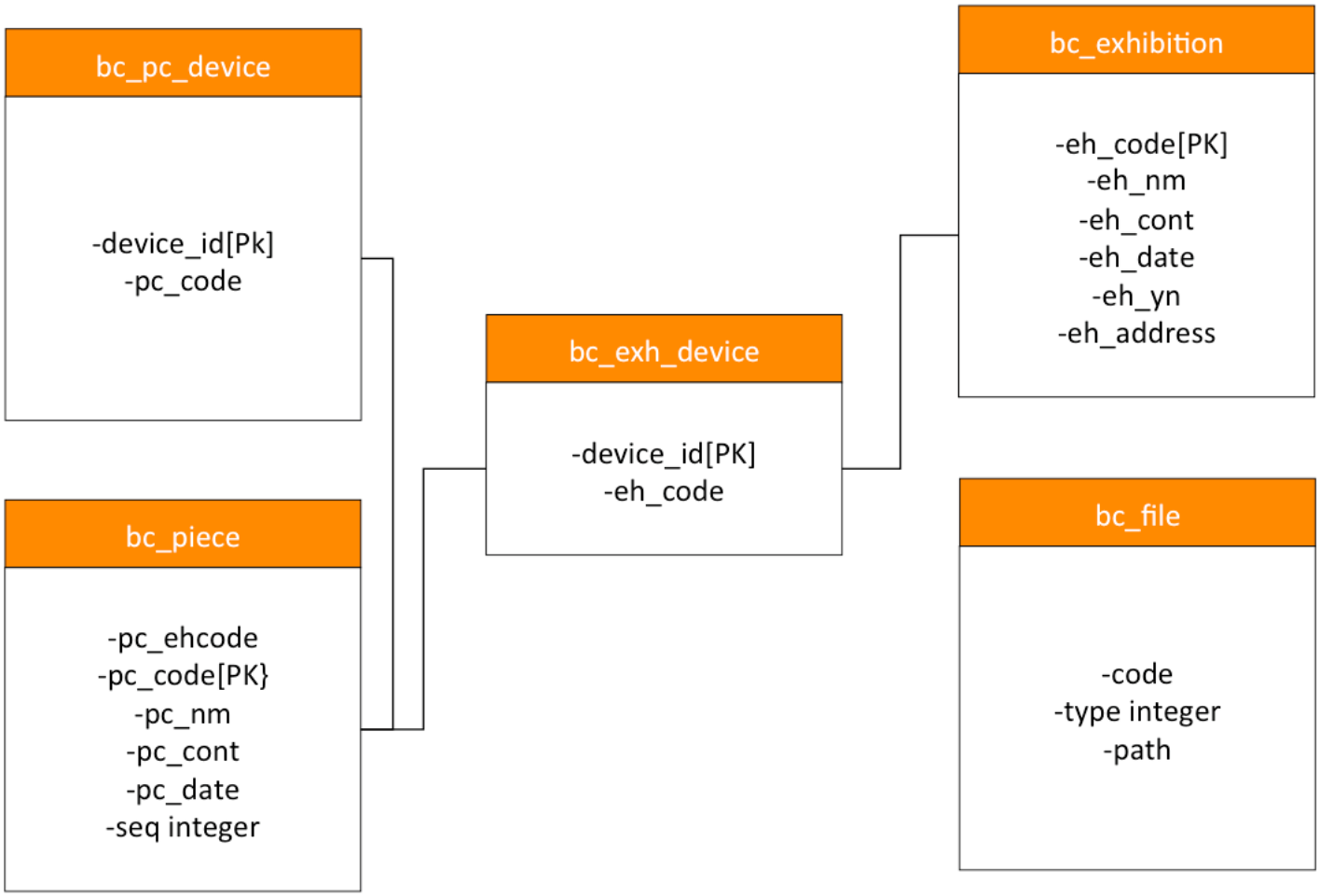

Fig. 23. Server Flowchart

.

6-1) BLE searching and push notice

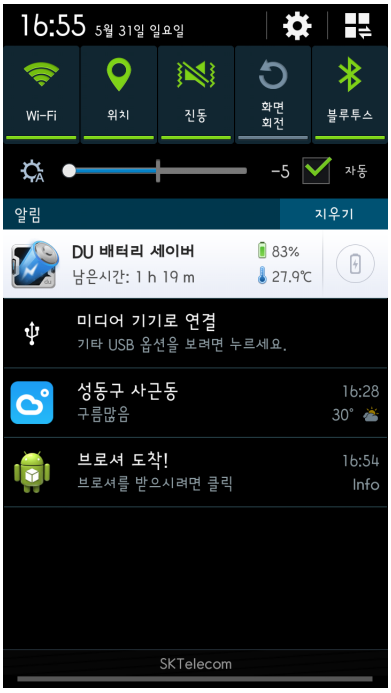

Fig. 24. BLE searching and push notice

If User turn on the bluetooth module, smartphone finds the BLE signal, and then Server sends to notice that information data about Exhibition is downloaded. If user touch the push notice, then temporary activity is open.

6-2) Temporary Activity01

This is the page when temporary page is open. It shows the information received from server. Upper-side of the page, there are the name and main image of exhibition. Under, there are the artwork lists which are displayed at the exhibition now user is seeing.

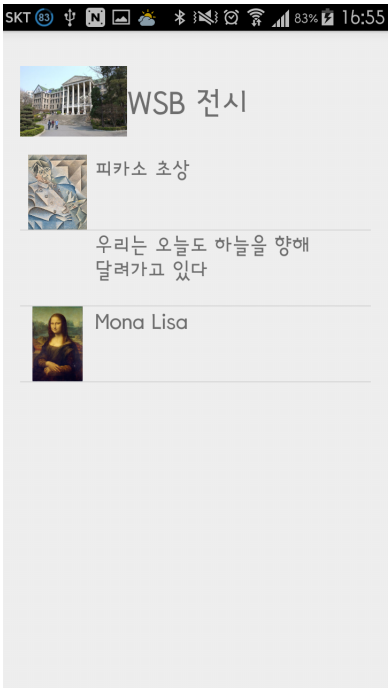

Fig. 25. Temporary Activity 01

6-2) Temporary Activity02

If the user touch one artwork, he can watch the detail of artwork. Of course every data is received from server. There are image, name, and detail descriptions of the artwork.

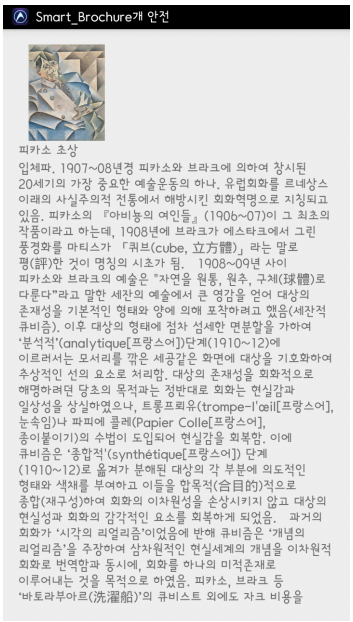

Fig. 26. Temporary Activity02

6-3) Main Application - My History Page01

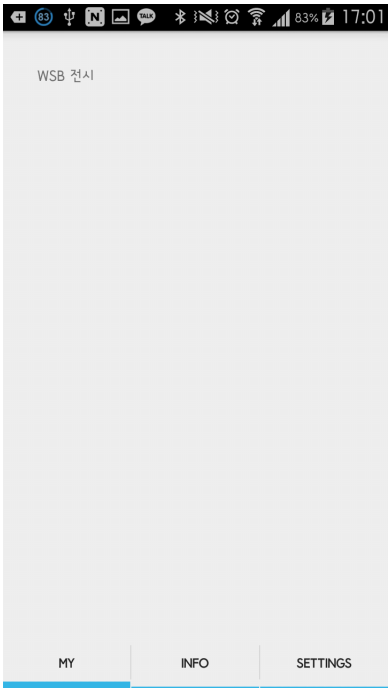

Fig. 27. My History01

User will see this page when he run the application. It is the first page of main application. First, there is tab menu under the screen. By this tab menu, user can move to page he want to see. This page is for user own. The data which he received from server and saw through temporary activity is stacked at this page. But this page only store the name of exhibition. Because the memory of smartphone. If user touch some exhibition which want to see again, then application send to the server the [eh.code] and get the data of that exhibition.

6-3) Main Application - My History Page02

This page shows the detail information about the exhibition. User can see the brief details about the exhibition theme, the Artist, and the list of artwork which is displayed on that exhibition.

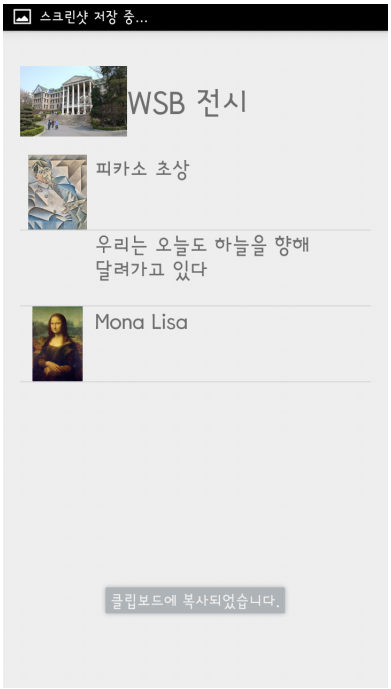

Fig. 28. My History02

6-3) Main Application - My History Page03

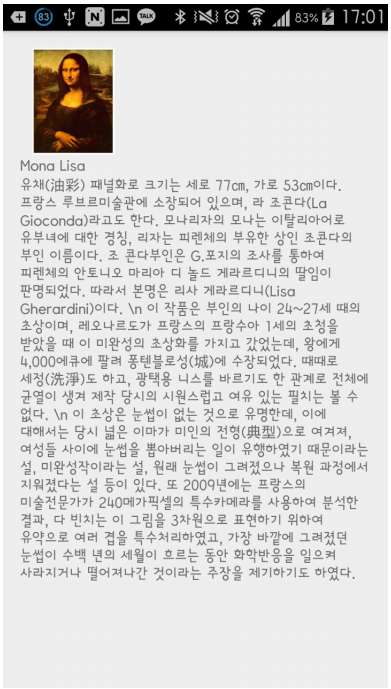

Fig. 29. My History03

If user touch any artwork at the My History Page01, user can see the detail information about that artwork like this page.

6-3) Main Application - Information Page01

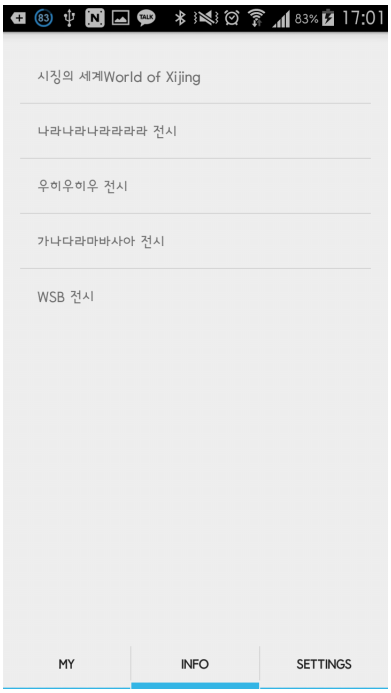

Fig. 30. Information Page01

This is the second menu page of the application. User can move to this page from the other using tab menu under the screen. This page is for inform about the other exhibition. On the screen, there will be the list of exhibition that user has not seen yet. If user touch the INFO menu from other page, or swipe down the list, then application connect to the server([bc.exhibition]) and get the data of the list of the exhibition. It means, user can refresh the list whenever he wants.

6-3) Main Application - Information Page02

Fig. 31. Information Page02

If user select one exhibition, he can see the detail information about that exhibition such like the main image of the exhibition, information about the artist, the address of the exhibition and so on.

6-3) Main Application - Setting Page

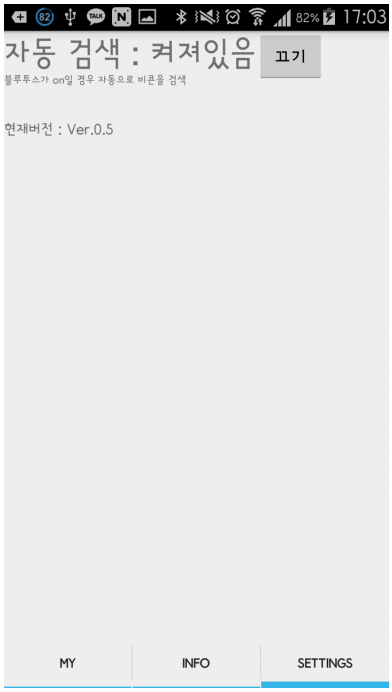

Fig. 32. Setting Page01

User can turn on or off the function searching the BLE. - Turn on

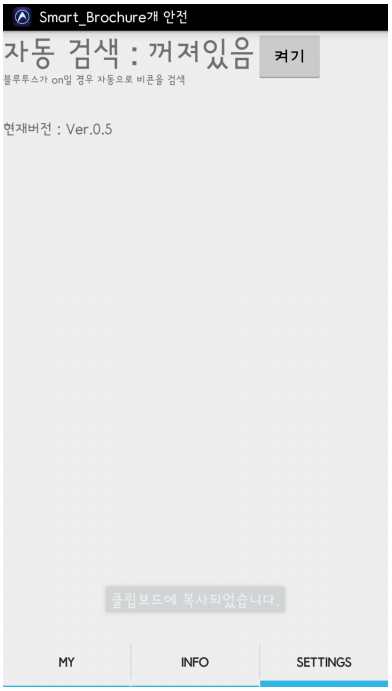

Fig. 33. Setting Page02

User can turn on or off the function searching the BLE. - Turn off

VIII. Software Installation Guide

8.1 How to install?

Enter the Google-Playstore or Appstore, and Search the [SMART BROCHURE] And you can get it.

IX. Disccusion

First of all, it was really hard for us to make the device communicate with beacons. Since we didn't have any SDK that works with our beacons, we had to develop the code from nothing. Therefore, we searched about it in the website, which gives developers android APIs that it supports. In the Bluetooth category, we were able to find the information about Bluetooth Low Energy(BLE). We had to study about the threads in the android, how to use Bluetooth in our application, and make it possible to run the service on the background of the device. Also we have some trouble to use Latex and Github. We were very confused when professor told us that we have to use them. After a lot of try, we could get used to use them. From this class, we could deal with so many unfamiliar sofrtwares and hardwares and learn from these process. Finally, we think it is better that doing usecase at first because it would be better for us to write the documentation.