**Requirements Specification Document**

**for**

**Empty Lecture Room Project**

2조

김남웅 김재범 박로빈 박승수 박지호 양태성

Table of Contents2

**0. Version History**3

**1. Product Description3**

1.1 Functional Requirements3

1.2 Non-Functional Requirements4

1.3 Constraints4

**2. Design Sketches5**

**3. Use Cases6**

**4. Process Description6**

4.1 Software Tools6

4.2 Job Descriptions and Work Process6

4.3 Roles and Schedule6

4.4 Risk Summary7

**5. Remarks8**

**0. Version History**

1.0.0 First Draft

1.0.1 Added Table of Contents, page number. Version history relocated

1.1.0 Functional Requirements change.

**1. Product Description.**

This project’s main objective is to create an application that searches the university (Chung-Ang University) to look for empty lecture room that may be usable in vague situations, such as in between lectures, space required for very short amount of time, etc.

* 1. Functional Requirements

1. Able to find empty lecture rooms based on specific criteria

1.1) Building: can find lecture rooms that is empty based on specified input. Results show empty lecture room based on building only (exclude other buildings).

1.2) Time: can find lecture rooms based on time specification

regardless of building, show lecture rooms available based purely on time (user location not specified)

1.3) Multiple Criteria: can select multiple criteria for search (ex: building 309, time 1300)

All above: as much as results should be shown.

1. Timetable Insertion: database insertion, user timetable

2.1) Database Insertion: Excluded. Updated via App Store for sufficient application size.

2.2) User timetable: User can input their timetable to the application to manage their timetable with the application. Not required but needed for another feature.

3) Room Recommendation: When user’s timetable is inserted, recommendation feature is usable. The application can search through the university timetable to locate an empty lecture room that may be suitable for the user.

3.1) Application tries to find the nearest lecture room available, based on where the user is most likely at based on the timetable.

3.2) Search first on the building floor, then move on to neighboring floors. Then move out to neighboring buildings, and so on.

* 1. Non-Functional Requirements

1. Usability Requirement

1.1) Executable for Android Gingerbread and over

1.2) First display will be user timetable. May show empty timetable if not inserted.

1.3) Tab layouts for each menu. (Tab layout: switches between menus by touching/sliding)

1. Efficiency Requirement

2.1) Data parsing time should be within one (1) minute.

2.2) Database search for categorized search should be within eight (8) seconds.

2.3) Application space should be within 10 MB for application only

2.4) No restraints for database space(TBD)

1. Reliability Requirement

3.1) Search based on categories should match the school timetable at all times

3.2) Search based on recommendation should match the school timetable at all times

3.3) Other reliability issues are ignored (explained on constraints)

1. Implement Requirement

4.1) Android SDK plugin version 2.3.3

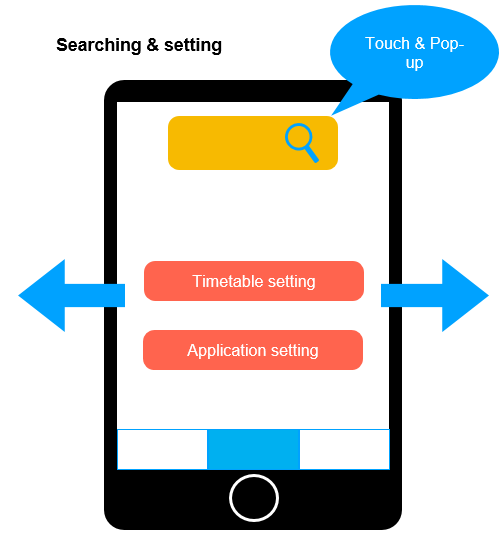
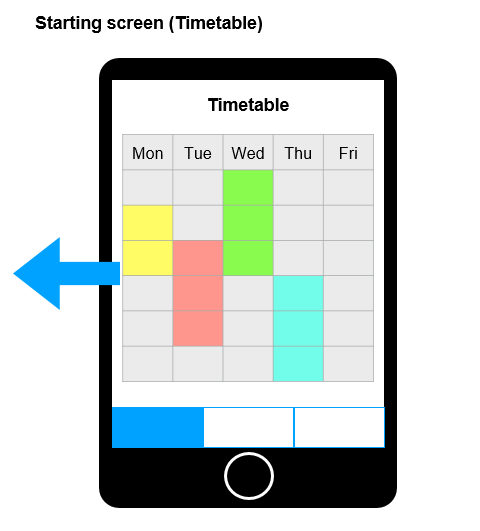
4.2) Python Beautiful Soup library is used for xtml parsing (for university timetable data)

4.3) Timetable data is searched from Everytime©.

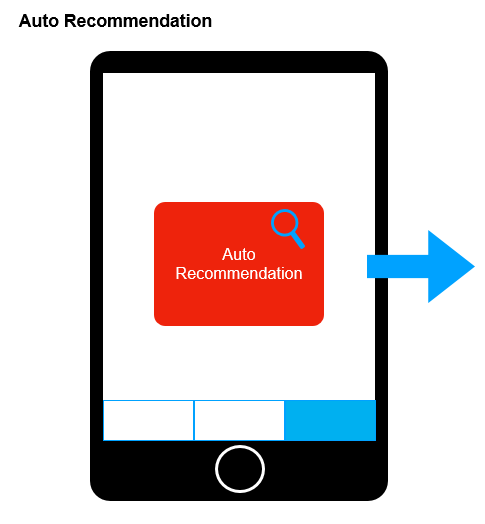
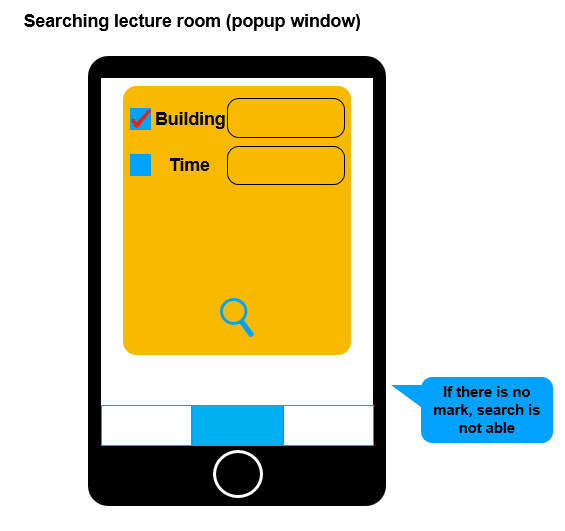
* 1. Constraints

1. School schedule that is not on the timetable cannot be searched
2. More constraints TBD

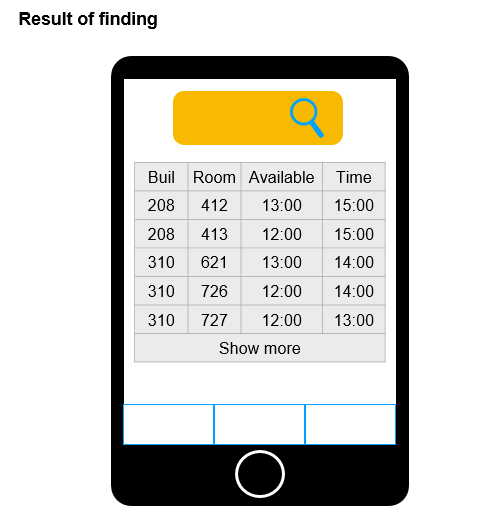
**2. Design Sketches**



**Rejected**



**Rejected**



**Rejected**

**3. Use Cases (scenarios)**

3.1 Scenario 1

Two lectures A and B are in different buildings, within less than 1-hour interval. The Recommendation function will search for an empty lecture room closer to the latter lecture. Recommendation will look up to your timetable to identify the next lecture to be.

3.2 Scenario 2

To have short meetings for students, you may select the category of “Building”, and “Time” to specify where you might want to meet. This search function would be more helpful if students are apart from each other.

3.3 Scenario 3

The case you may want to wait for any reason (fellow, peer) at a certain building. You can select the category “Building”, to specify your needs.

**4. Process Description**

4.1 Software Tools

Python3: used for timetable crawling

Android SDK: for overall design and integration on an Android Smartphone

Mysql: managing database

More TBD

4.2 Job Descriptions and Work Process

1) Timetable crawling from Everytime© with pyhton3

2) Establish database with collected data

3) Prototype by Android SDK without heavy design

4) Testing (tool TBD)

* 1. Roles and Schedules

4.3.1 Roles

Team Representative – 박지호

Application Basic Design – 박지호 양태성

Component Assembly – 박지호 양태성

Documentation, Presentation Preparation – 양태성

Database Architecture Design – 김재범

Technology Implementation – 김남웅 박지호

Testing Team – 박로빈 박승수

4.3.2 Schedules

~ 4/9 Finish first draft of requirement specification

~ 4/15 Have customer meeting, improve requirements specification

~ 5/15 (expected) Component prototyping

Layout prototyping

Parsing data prototyping

Database SQL Design/prototyping

Have regular meetings with customer

~ 5/18 Test individual component

~ 6/9 Component assembly

~ 6/10 Finalize before release

Be sure to have regular meetings with customer to prevent larger problems.

Date may be changed due to lecture schedules.

4.4 Risk Summary

Risk Level

Level 3: Can occur frequently during development

Level 2: May occur but not frequently

Leve 1: Occurrence is very unlikely

* + 1. External Factor (Developer)

1. When the SW differs with customer demand (Level 3)

Solution: Have regular meetings with the customer to ensure development is on course, keep up with consistent prototyping

1. When one developer is unable to participate temporarily (Level 2)

Solution: Contact the remainder of team members for efficient division of remaining work

1. When the development time becomes longer than predicted (Level 2)

Solution: Contact and discuss with customer to delay the deadline for the project, with cost responsibility to developers

1. When one developer becomes unable to participate permanently (Level 1)

Solution: Reduce the project scale, or delay the deadline with discussion with the customer

* + 1. External Factor (Customer)

1. When factors that are demanded by the customers take too much time or impossible (Level 3)

Solution: Have discussion together with the developers to identify the factor and try to find alternatives

1. When customers are unavailable for meeting (Level 2)

Solution: Try finding contact methods online (KakaoTalk for instance), if not available email the progress

1. When customers are unreachable for contact on development error (Level 2)

Solution: Email the error with suggestion to solve the error

1. When the cost is over what has been predicted (Level 1)

Solution: Have meeting to compare most expansive factor to modify or remove demands

* + 1. Internal Factor (Software)

1. When error occurs on testing before component assembly (Level 3)

Solution: Find the error from the component (with frequent prototyping)

1. When error occurs after component assembly

Solution: The error would have occurred from assembly process, try finding the error during assembly

**5. Remarks**

All above may be altered due to customer meetings, schedule change, team member insufficiency.