

COMP4971F - Independent Work Final Report

Multi-device web applications suite for API-less online services

KU Chun Kit

Advised by Dr. David Rossiter

Department of Computer Science and Engineering
The Hong Kong University of Science and Technology

09 May 2015

Table of Contents

1.	Motivation.....	1
2.	Deliverables.....	1
3.	Features.....	2
	3.1 HKUST "break in" (UST 爆房).....	2
	3.2 HKUST room booking (UST 卜房)	2
	3.3 Public API: liba_lib.....	3
	3.4 Cross-platform compatibility	3
	3.5 Mobile optimized interface.....	4
4.	Demonstrations.....	4
	4.1 HKUST "break in" (UST 爆房).....	5
	4.2 HKUST room booking (UST 卜房)	8
	4.3 Public API: liba_lib.....	13
	4.4 Cross-device compatibility	15
5.	Competitive analysis	16
6.	Implementation	17
	6.1 HKUST "break in" (UST 爆房).....	17
	6.2 HKUST room booking (UST 卜房)	17
	6.3 Public API: liba_lib.....	18
7.	User comments	19
	7.1 Benoit Travers (BBA, year 3, using iOS Safari).....	19
	7.2 Ivan Fung (CS & Math, year 3, using Android 2.3)	20
8.	Improvements.....	20
	8.1 HKUST "break in" (UST 爆房).....	20
	8.1.1 Location-specific suggestions.....	20
	8.1.2 Time-specific suggestions.....	20
	8.2 HKUST room booking (UST 卜房)	21
	8.2.1 Time-specific suggestions.....	21
	8.2.2 Speed boost.....	21

9. Conclusion.....22

1. Motivation

Despite a growing trends for online service providers such as Twitter, Google and Facebook to provide API for third-party developers to integrate these services into their own applications, many developers and operators of other online services are reluctant to provide or open APIs. However, chances are these online services are providing suboptimal user experiences and have not made full use of the application's potentials.

To improve this, this project investigates different techniques of utilizing these online services even without publicly accessible APIs. Moreover, third-party developers could connect several online services into a suite of powerful application that achieves outcomes impossible when these online services are working independently. The experiences gathered from this project can be extended to utilizing other API-less online services by a third-party developer.

2. Deliverables

A room break-in application for locating available empty classroom at HKUST will be created by scraping the open online course catalogue.

A HKUST library room booking application for finding and booking available HKUST library study rooms without date limitation. (Currently, library restricts booking to within 2 weeks only).

Both of these 2 applications are built without relying on the APIs of the original HKUST web services because even if they exist, they are not publicly accessible. To enable third-party developers to make use of the web services, APIs to the applications built for this project will also be opened for public access.

3. Features

3.1 HKUST "break in" (UST 爆房)

To many HKUST students, it is hard to find a place to study or discuss in the crowded campus. Even though group study rooms and some common study places are available, they are normally over-crowded. Meanwhile there is an abundance of empty classrooms un-used, un-noticed and under-utilized.

This application first parses the publicly accessible [HKUST course catalogue](#) for availability of classrooms and generate a set of available rooms at every timeslot over the week. The parsed and processed data are reusable by storing them into the database, indexed by weekdays and timeslots.

The application is served as web page. Not much logic is involved in terms of retrieving data from the database. Each page refresh delivers information about one empty classroom at the time. The application conveniently allows user to share the room information via *WhatsApp* directly from the web page. Other than the page content, other assets such as images and style sheets are cached by the browser as frequent page refresh is a common place when using this application.

3.2 HKUST room booking (UST 卜房)

Used by thousands of HKUST students, the [HKUST library room booking](#) system is a popular yet rather simple web service. It allows for students to book rooms with only the restrictions of time (one cannot book room beyond 2 weeks in the future) and room availability. Compared to *HKUST "break in"* which is making use of static data, *HKUST room booking* contains multiple dynamic components: parser to check for room availability and request forger to create HTTP requests for room booking. These operations are to be carried out in real time by sending forged requests to the library's system.

This application uses HTML parser to extract data about room availability. Then it collect necessary information from user such as date, room of choice, ITSC user name and password before forging a booking request to library system.

Forging requests to trick the system into believing it is interacting with the actual user can be done only because the library's system does not attempt to check whether the user involved is actually a human using techniques like CAPTCHA (reverse Turing test).

The application is a web app with the logic handled by JavaScript. It intelligently suggest a room currently available for booking through a user-friendly interface that reassembles some of the booking apps in the market.

Most importantly, this application help users to break the 2-week restriction of booking by internally scheduling the booking to the date when booking becomes possible, then the application can book the room on behalf of the user. This shows how open API can add value to an existing system by enabling some features unavailable without it.

3.3 Public API: *liba_lib*

Although the HKUST library room booking service investigated in this project does not provide open API (application programming interface) for developers to probe into the system, wrapper library can still be created by parsing their HTML outputs and forge HTTP requests to mimic an ordinary server-browser interaction and fake the service into believing it is interacting with users via designated means, while it is actually exchanging information with another server.

A PHP library *liba_lib* is developed precisely for the purpose of wrapping the HKUST library room booking system and thus provide a public API as if the developer is communicating directly with the library's system.

The code are organized into a package [available on GitHub](#) and can easily be embedded into other projects using package manager *Composer*.

3.4 Cross-platform compatibility

To deliver the application to the most number of users possible, the applications are built to be compatible with platforms ranging from ordinary desktop computers to mobile devices. By doing so, users are able to interact with the application without having to install it beforehand. Also, there will be only one codebase to maintain in contrast to multiple codebase for non-cross-platform applications.

3.5 Mobile optimized interface

Mainly serving as utility applications, the two applications are built to optimize performance in mobile devices. In recent years mobile devices have become the most common devices people use every day and it would totally miss the point as utility applications if they are not optimized for mobile. Therefore, while the applications are totally usable on desktop computers, their layouts and architectures are optimized for mobile. This is done by using responsive layout, caching application assets in the browser and adding message indicating the application is being loaded.

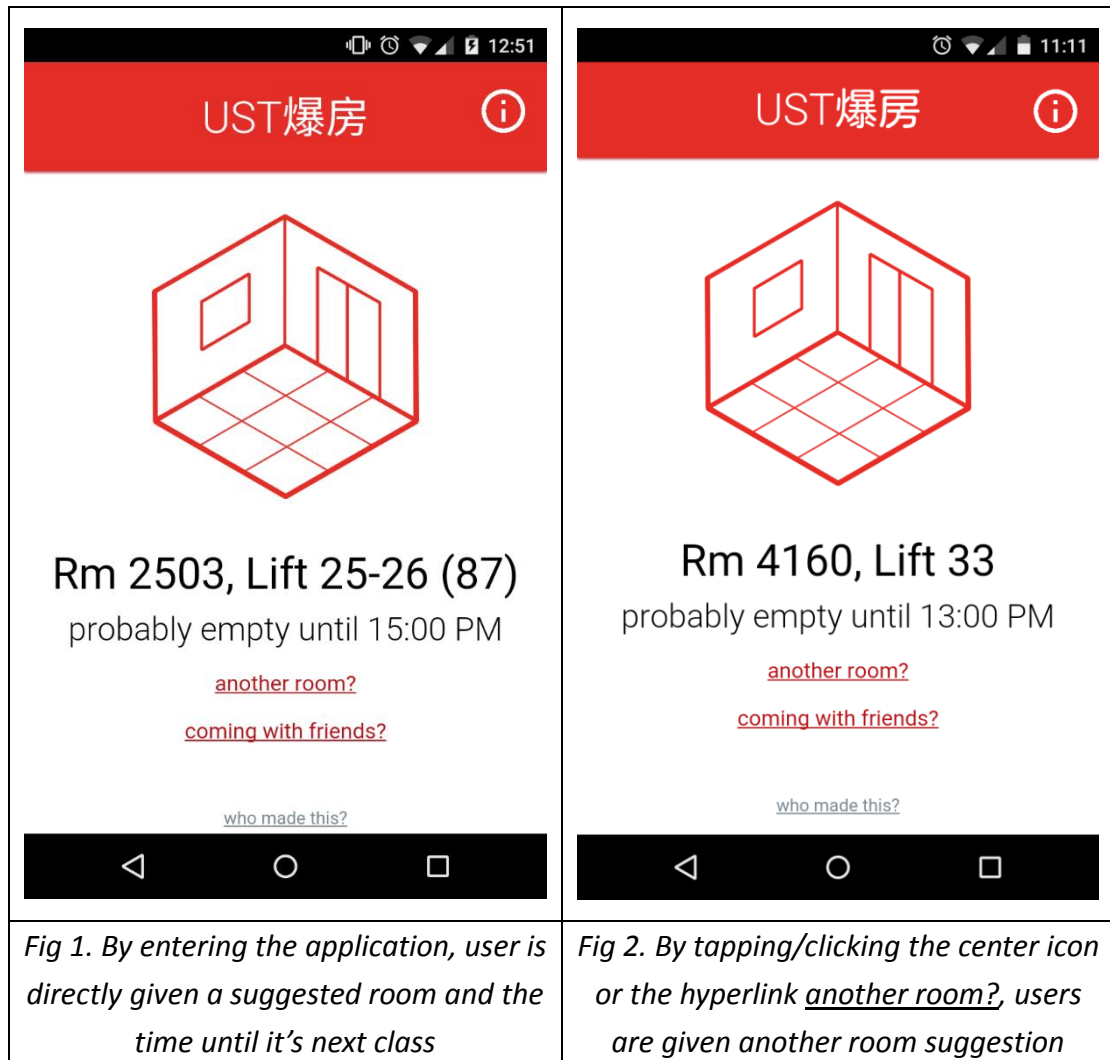
4. Demonstrations

For each application/API demonstrated, all of the screenshots are taken on a *LG Nexus 4* smartphone running *Android 5.0* unless specified otherwise.

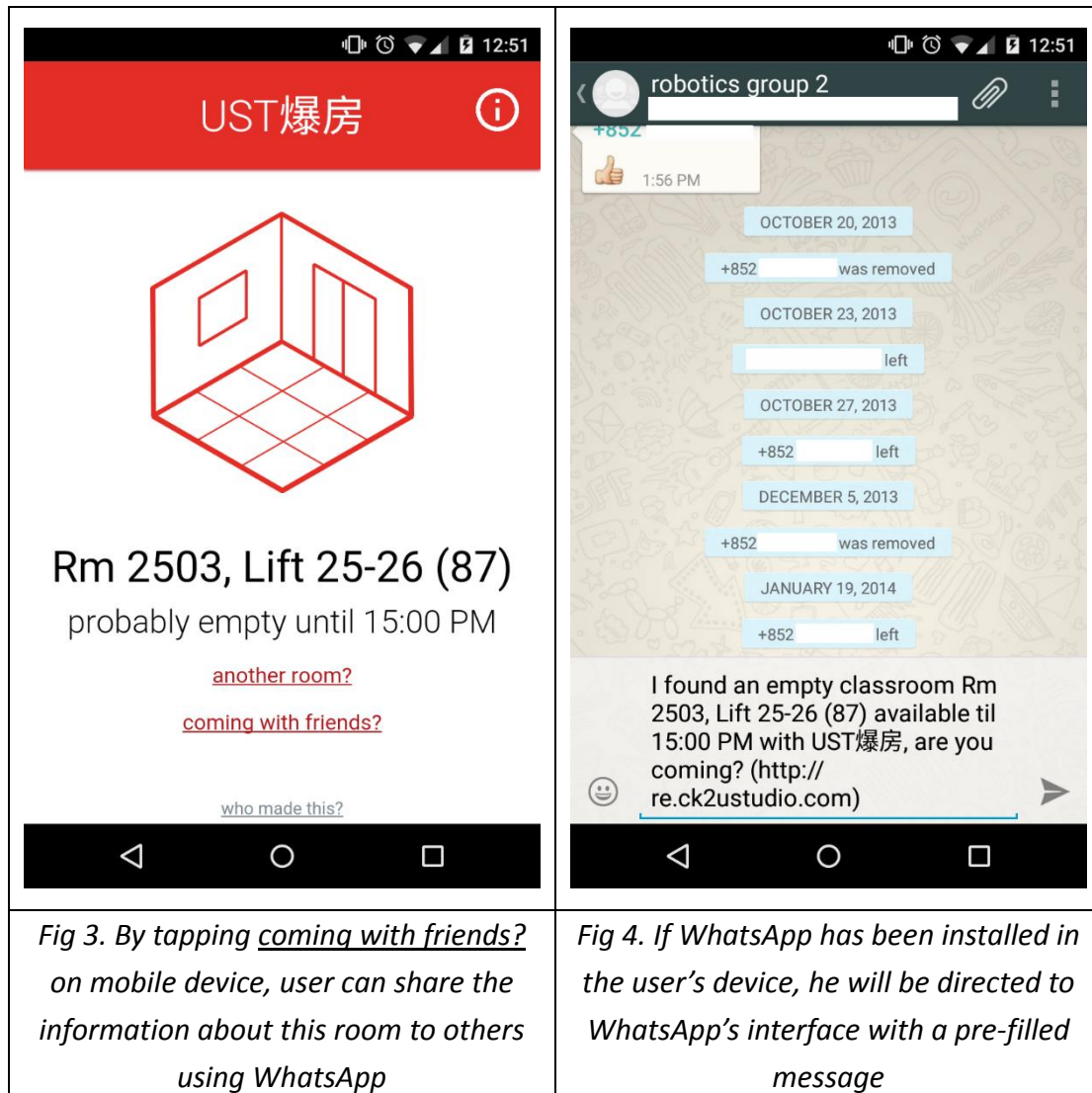
(The demonstration starts from next page)

4.1 HKUST "break in" (UST 爆房)

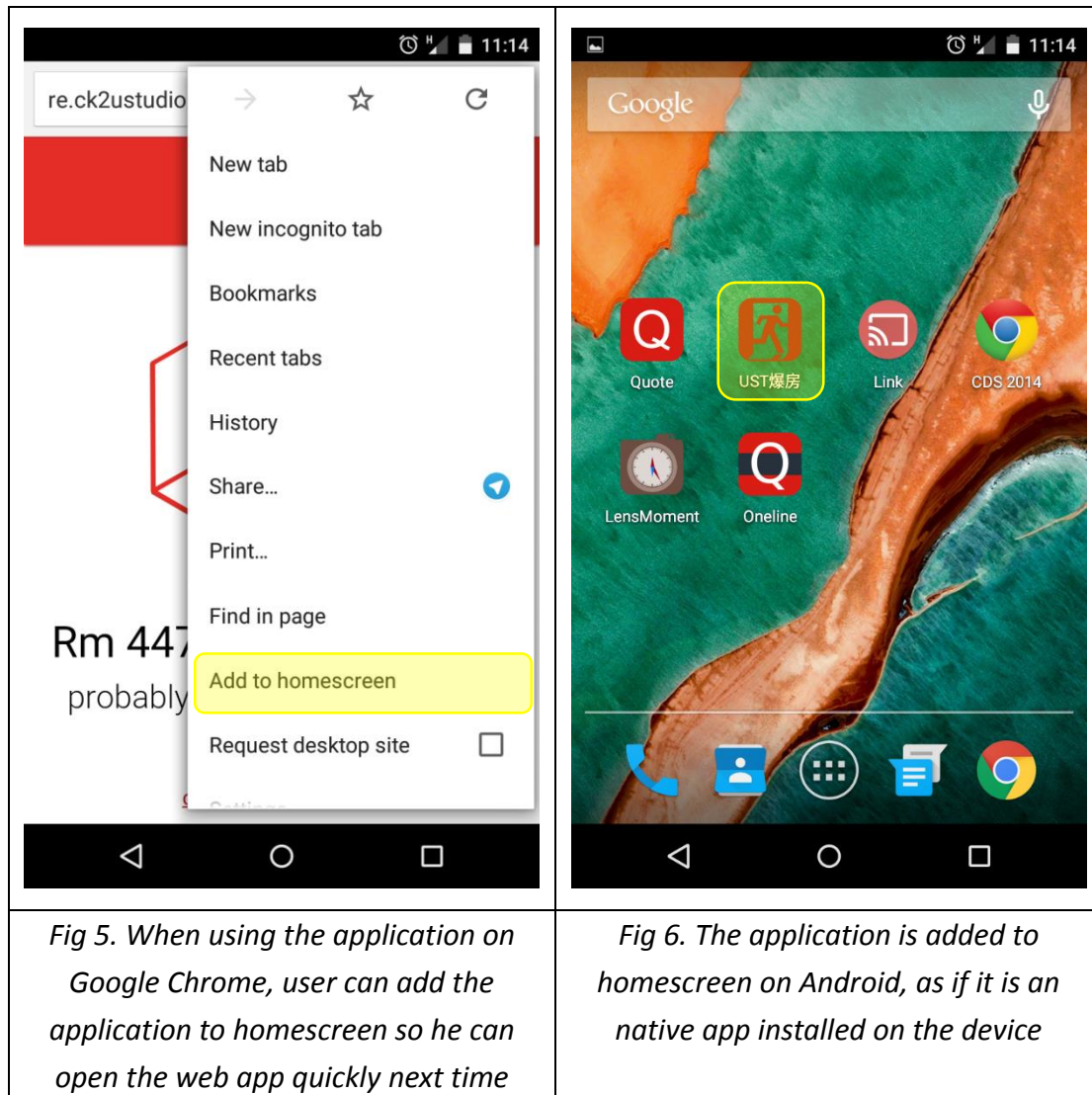
Getting a room



Share to WhatsApp



Add to homescreen



4.2 HKUST room booking (UST 卜房)

Choosing an area

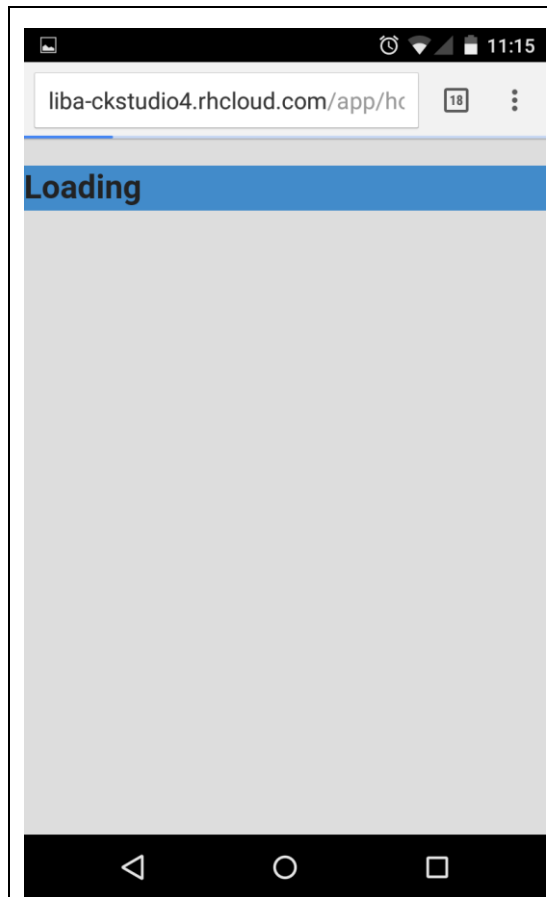


Fig 7. Before all of the application assets are downloaded, user is shown a loading screen assuring them the application is working

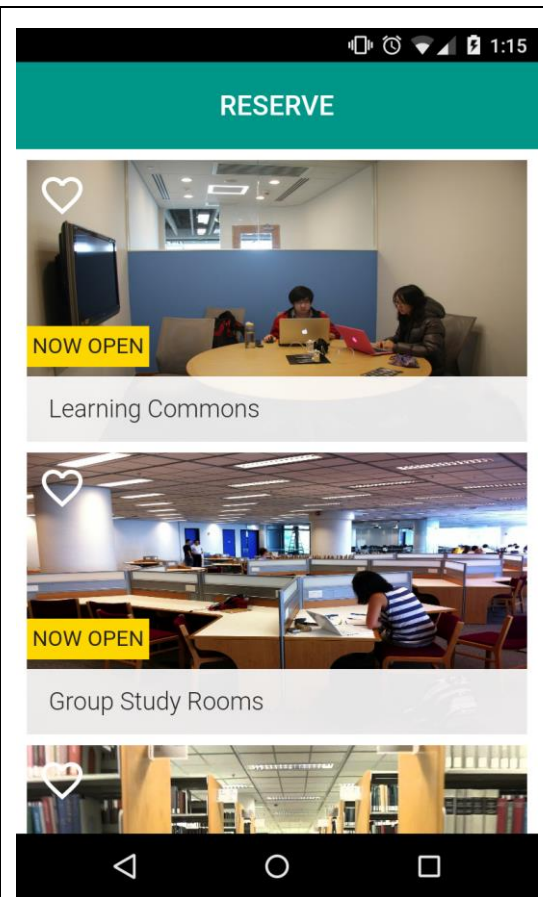
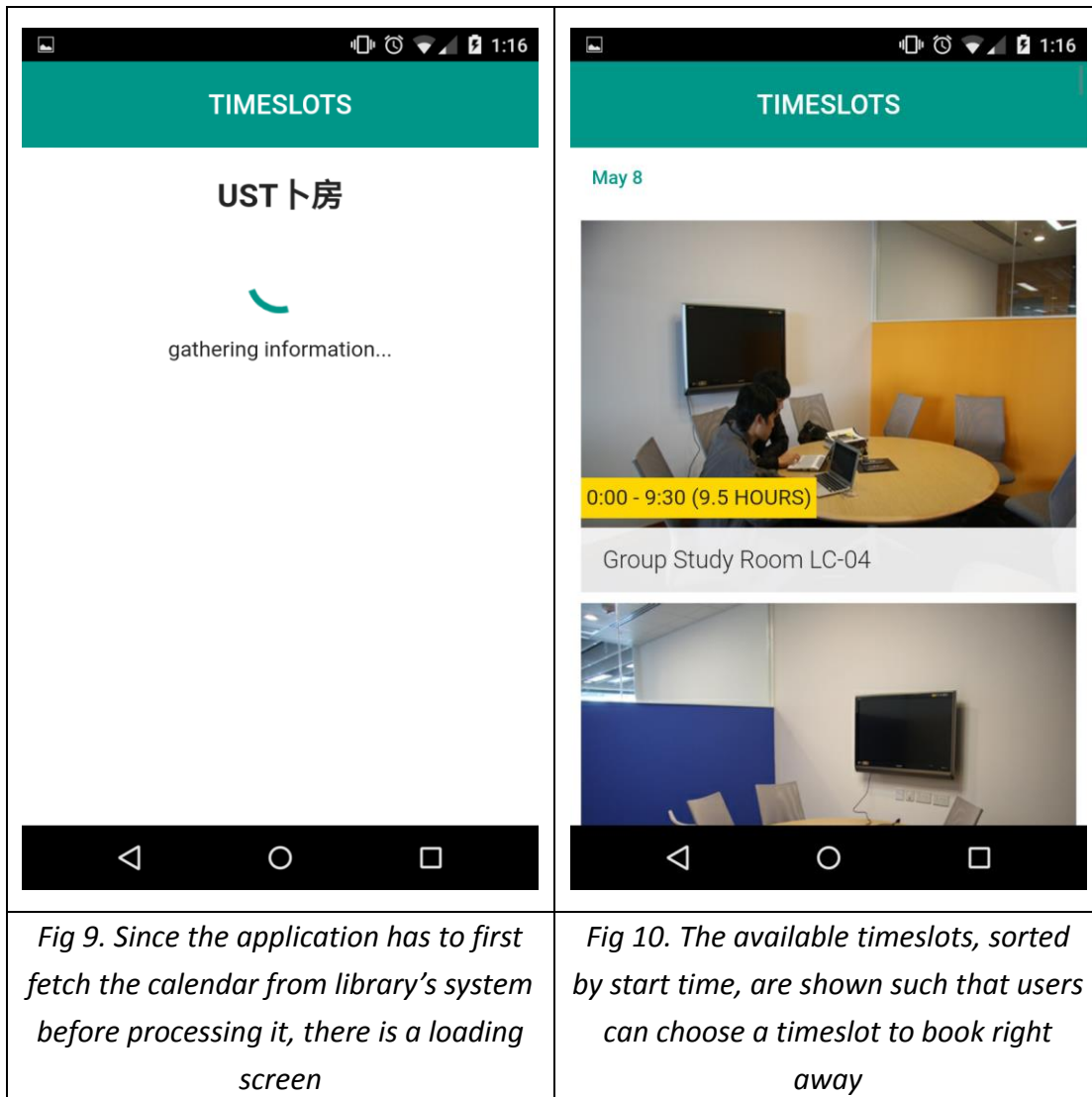


Fig 8. User is shown a list of library areas available booking, they can choose an area to see available timeslots

Searching for available rooms



Searching for available rooms (cont'd)

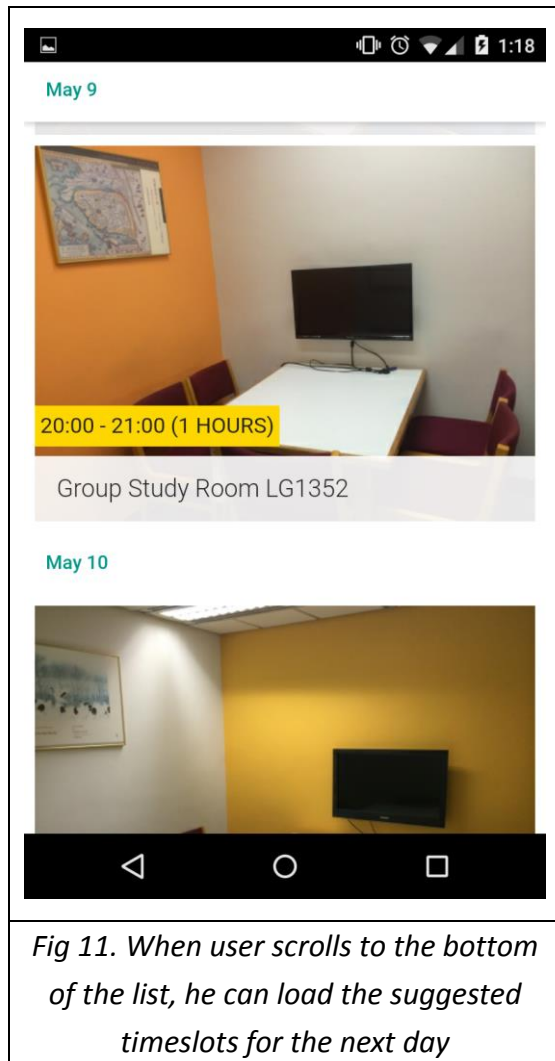


Fig 11. When user scrolls to the bottom of the list, he can load the suggested timeslots for the next day

Viewing room particulars

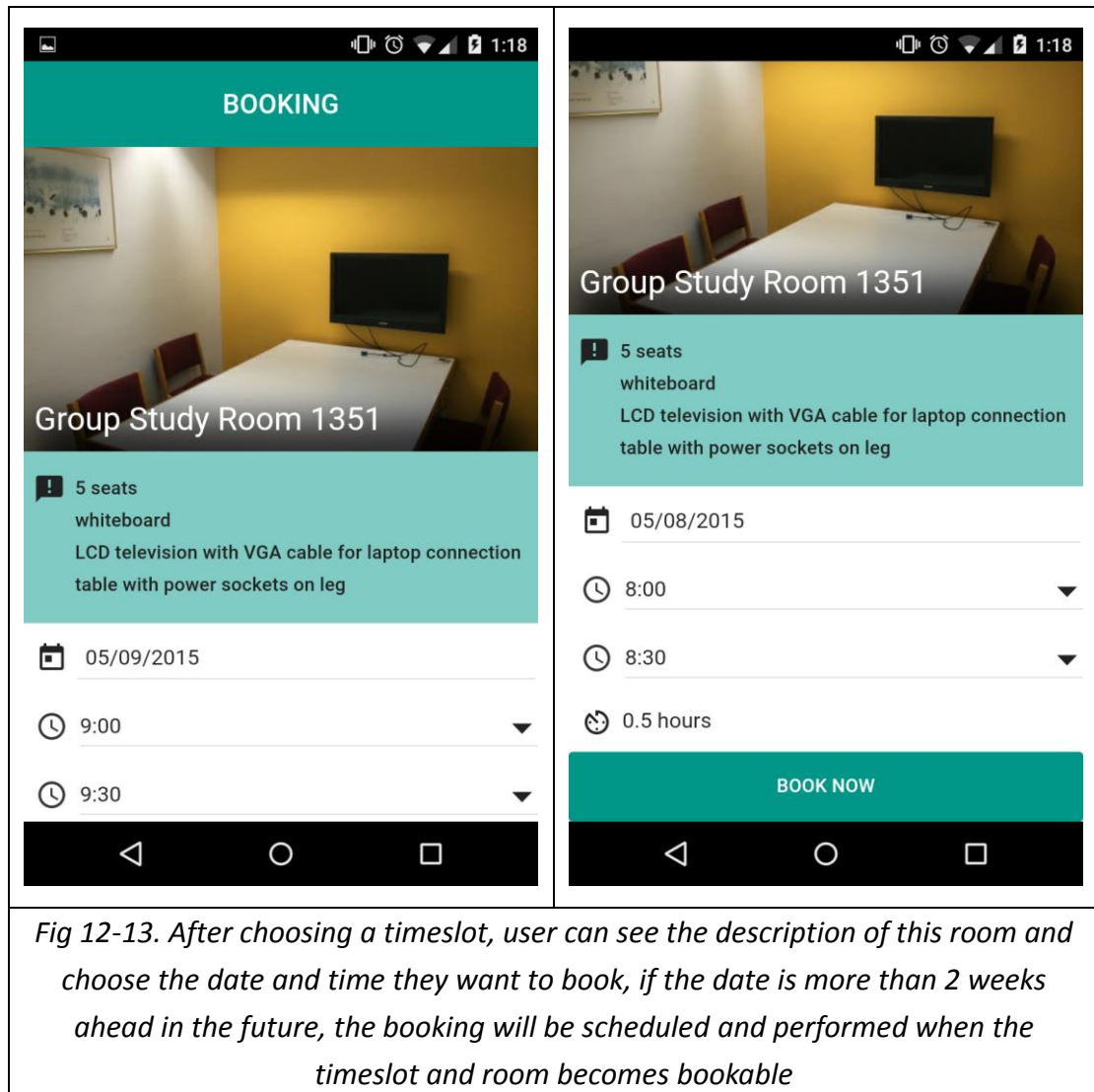
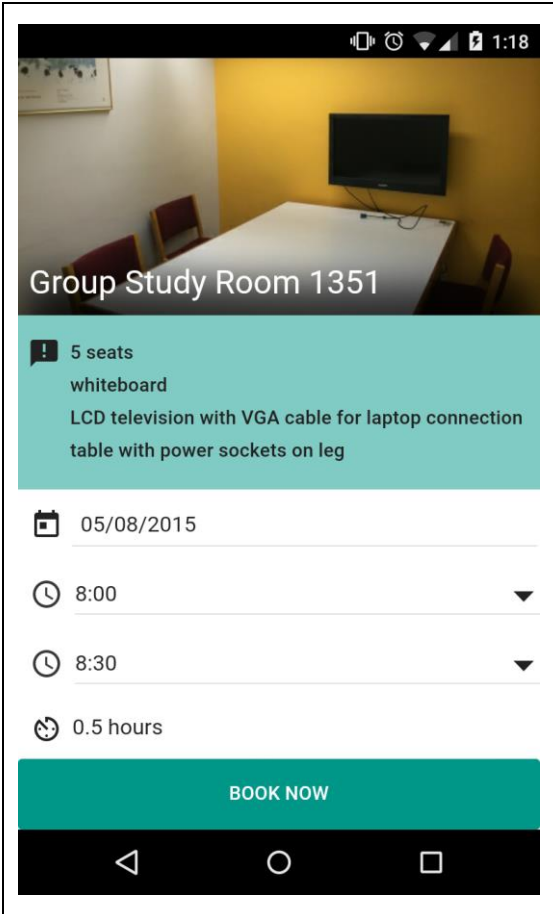
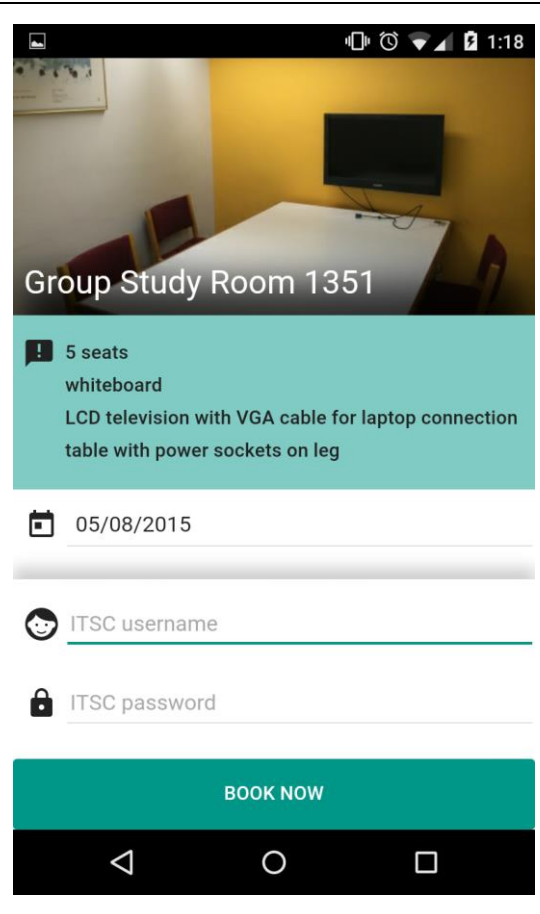


Fig 12-13. After choosing a timeslot, user can see the description of this room and choose the date and time they want to book, if the date is more than 2 weeks ahead in the future, the booking will be scheduled and performed when the timeslot and room becomes bookable

Booking a room

 <p>Group Study Room 1351</p> <ul style="list-style-type: none">5 seatswhiteboardLCD television with VGA cable for laptop connectiontable with power sockets on leg <p>05/08/2015</p> <p>8:00</p> <p>8:30</p> <p>0.5 hours</p> <p>BOOK NOW</p>	 <p>Group Study Room 1351</p> <ul style="list-style-type: none">5 seatswhiteboardLCD television with VGA cable for laptop connectiontable with power sockets on leg <p>05/08/2015</p> <p>ITSC username</p> <p>ITSC password</p> <p>BOOK NOW</p>
<p><i>Fig 14. After choosing a booking date and time, the duration field is updated automatically</i></p>	<p><i>Fig 15. By tapping <u>BOOK NOW</u> user are prompted to insert his ITSC user name and password to complete the booking process</i></p>

4.3 Public API: liba_lib

This API is a wrapper for HKUST library room booking system, it sends forged requests to the system pretending to be initiating the transaction from a browser. This library enables third-party developers to easily access library room booking service using this well-documented and thoroughly tested module. Users primarily utilize public methods of two classes: *Schedule* and *Booking*.

class LibaAPI\Schedule

```
void __construct($_bAuth_user = NULL, $_bAuth_pass = NULL)
```

Create a new Schedule class instance

```
array getAvailable($date, $area=3, $cover=1, $limit=50)
```

Get a list of available room at the designated date time

```
boolean isAvailableWithoutLogin($start, $end, $area, $room)
```

Check if a room is available without the need to login

class LibaAPI\Booking

```
void __construct($_bAuth_user = NULL, $_bAuth_pass = NULL)
```

Create a new Booking class instance

```
boolean book($start, $end, $area, $room)
```

Book a room

```
stdClass isBookable($start, $end, $area, $room)
```

Check if a room is bookable by calling the library system itself

With the two classes described above, third-party developers are empowered to access to the API-less HKUST library room booking system. They can build their own applications regardless of the low-level implementation of this wrapper. Furthermore, they could utilize or extend the *Parser* class for even more functionalities.

class LibaAPI\Parser

```
static getInstance()
```

Enforce singleton design pattern


```
static array parseSchedule($_bAuth_user = NULL, $_bAuth_pass = NULL)
```

Parse available room schedule for the next 2 week for all area

```
static array parseDay($_bAuth_user = NULL, $_bAuth_pass = NULL, $date)
```

Parse for available room schedule on one day, all area, all room

```
static array parseRoom($_bAuth_user = NULL, $_bAuth_pass = NULL, $date, $area,  
$room)
```

Parse for available room schedule on one day, one area, one room

```
static array parseArea($_bAuth_user = NULL, $_bAuth_pass = NULL, $date, $area)
```

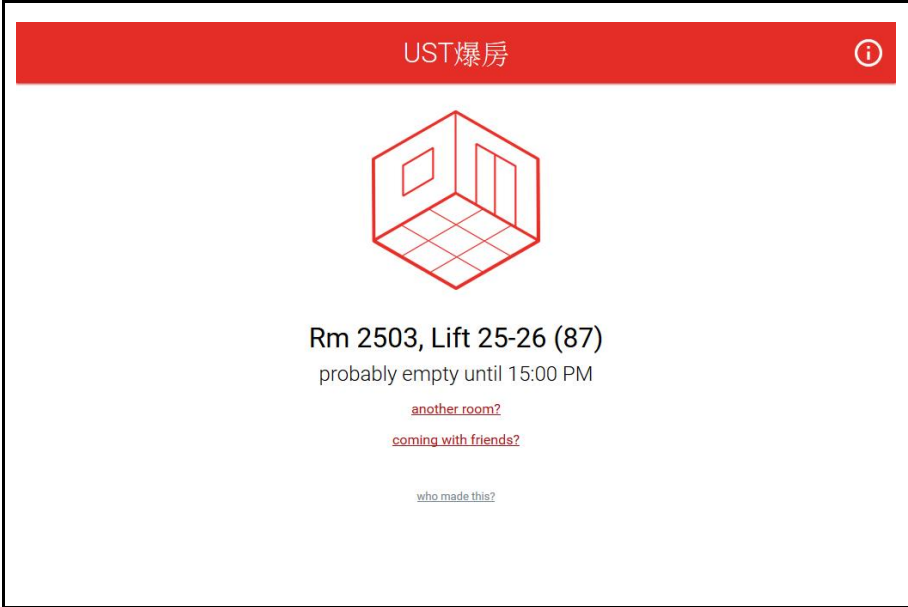
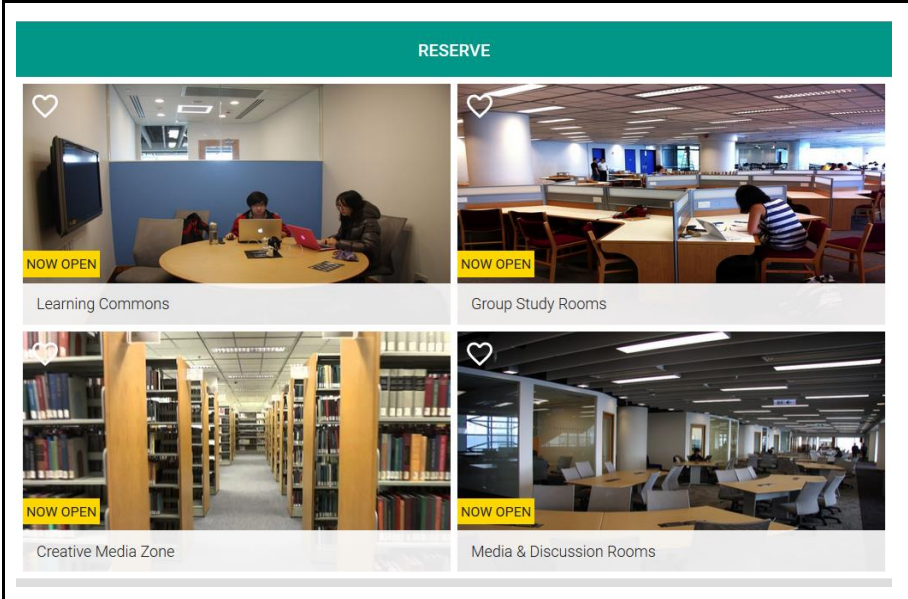
Parse for available room schedule on one day, one area, all room

The source code of this public API is open sourced and is hosted on GitHub. One can declare this library as a dependency of his project using *Composer*.

4.4 Cross-device compatibility

The interfaces of *HKUST “break in”* and *HKUST room booking* are both made to be responsive. That is, their layouts are adaptive to window size. When the window size is larger than a certain value, the layout will switch from single-column layout to double-column in order to optimize the use of space.

Since they are both web apps, users can use them on desktop or mobile with minimum restriction except that *HKUST “break in”* share to WhatsApp feature is unavailable on desktop.

	<p><i>Fig 16.</i> Firefox/Win 8.1</p>
	<p><i>Fig 17.</i> Firefox/Win 8.1</p>

5. Competitive analysis

This section compares *HKUST room booking* with the original HKUST library room booking system.

Feature Platform	Auto-suggest bookable room	Booking beyond 2 weeks	Delete booking	Load speed
HKUST room booking (UST 卜房)	Yes, shows only bookable room ordered by availability	Yes, via scheduled booking	No	Slow, may take 10 seconds to load suggested timeslots
HKUST library room booking system	No	No	Yes	Fast, usually finished loading within several seconds

The major competitive advantage of *HKUST room booking (UST 卜房)* is the ability to suggest bookable room and book beyond 2 weeks. In terms of speed, *HKUST room booking (UST 卜房)* is slower because it needs to fetch and parse data from the library system first.

6. Implementation

6.1 HKUST "break in" (UST 爆房)

This application fetch and parse the HKUST course catalogue to create a list of available classroom for each 30-minute timeslots of the week. Python library *Scrapy* provides convenient functions to parse and extract HTML content using XPath and regular expression.

The parser generates a set of room-timeslot pairs, which are then fed into an $O(n^2)$ sorting program that converts data into timeslot-available room format.

The front end merely serves value stored in the database. Since users may refresh frequently to get the most suitable suggestion, this application leverage browser caching of application assets including images and fonts to optimize loading speed.

6.2 HKUST room booking (UST 卜房)

HKUST library room booking system wrapper *liba_lib* is heavily used to provide some core features of this application in the back end. PHP framework *Laravel's* scheduling feature enabled for booking beyond 2 weeks.

The back end provides RESTful web service to manipulate 2 resources – timeslots and bookings. Users can request for timeslots and post bookings via HTTP requests. The PHP script running on the server can recognize which route and which HTTP method the client is calling, then invoke the respective controller to handle the request. Should a booking request attempt to book a room more than 2 weeks later, the requests will be stored in MySQL database that using *Laravel's* scheduling feature can retrieve and perform the booking in the future.

The front end makes use of *AngularJS* framework and a user interface library *Material Angular* to keep the code organized and provide an experience reassembling that of a native app. Each view of the front end client has its respective controller and HTML template such that different parts of the application are well decoupled, even in local level.

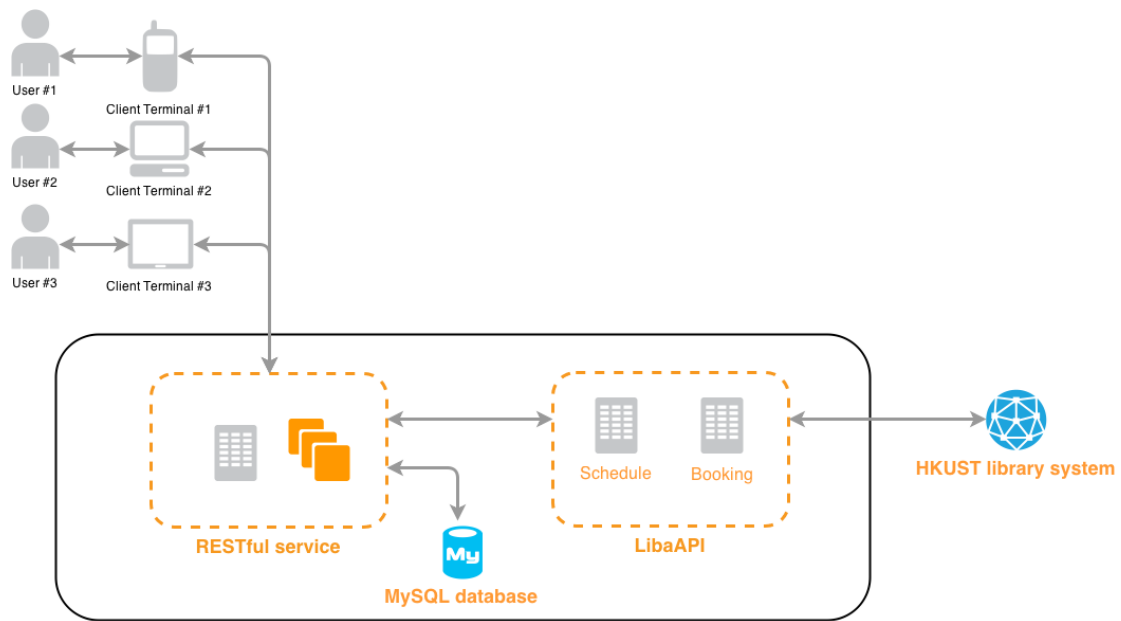


Fig 18. System architecture

6.3 Public API: liba_lib

The aim of developing *liba_lib* is to investigate the feasibility of wrapping an API-less web service to provide interface as if a native API was provided by the target service. In this case, *liba_lib* serves as a wrapper to HKUST library room booking system's schedule display and booking functions. Fig. 19 in the next page is a class diagram explaining the implementation of this class.

Instead of using native CURL functions, *rmccus/requests* is used to more conveniently forging HTTP requests. To parse HTML responses from library system, *ganon* is used. The classes are organized under namespace *LibaAPI* with a set of test cases for other developers to understand the usage and ensure integrity when they further develop the library.

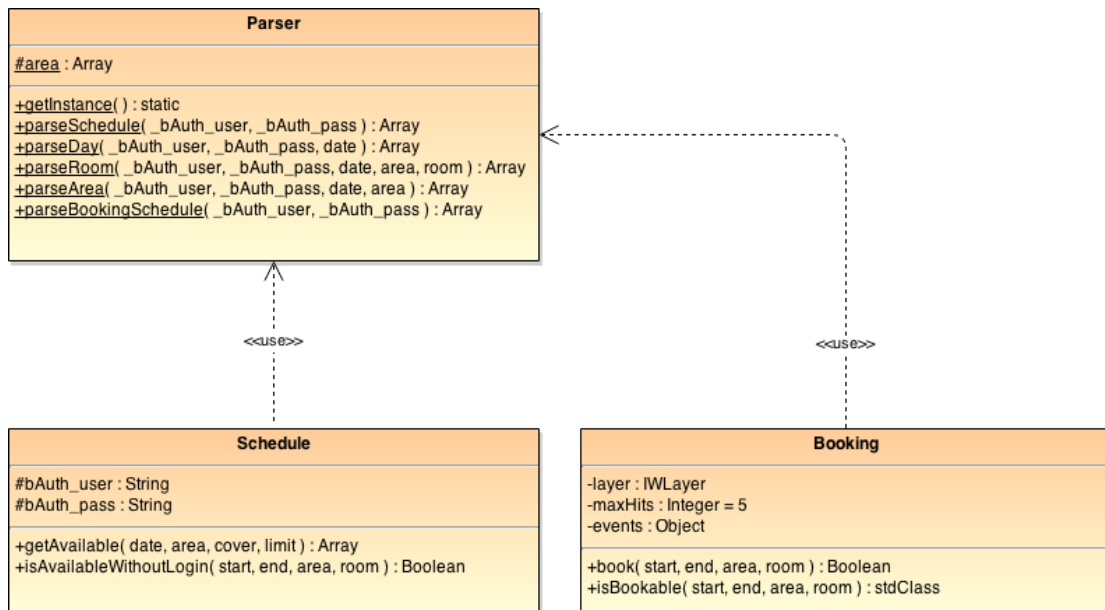


Fig 19. Class diagram

7. User comments

7.1 Benoit Travers (BBA, year 3, using iOS Safari)

On HKUST "break in" (UST 爆房):

When my group members and I are planning for a meeting, it seems strange to tell them we are going to have a meeting on that day at that time, but I can only tell you the venue 30 minutes before the meeting. This application only suggests a room available at the moment when I am using it, so I cannot use it to plan for a meeting. It would be better if I can input a time and get a suggestion.

On HKUST room booking (UST 卜房):

The user interface looks like a native mobile and the design is quite modern. The transition animations could be a bit smoother though.

Usually when we book a library room we are booking it for a couple days later. Currently in this application I need to scroll down hundreds of timeslots in order to reach the day I want to book room on. There should be a date filter for users to see timeslots on their date of choice directly.

7.2 Ivan Fung (CS & Math, year 3, using Android 2.3)

On HKUST "break in" (UST 爆房):

This is a handy application for me to locate an empty room at any time I want. However, the application sometimes tells me to go to LSK building which is far away from where I am. There ought to be a search-by-location/lift number function so I can easily locate an empty classroom nearby.

On HKUST room booking (UST 卜房):

I do not trust in this application when it asks for my ITSC user name and password lest should it record my password for malicious activities. It should instead redirect me to the library system's booking interface.

8. Improvements

8.1 HKUST "break in" (UST 爆房)

8.1.1 Location-specific suggestions

Originally the application has no knowledge on the location of its user, which forces them to refresh the page several time to get an empty classroom nearby.

To improve, classrooms, lifts and critical places such as library, canteen and atrium can be organized into interconnected nodes in a graph with weights assigned to paths between nodes. By applying Dijkstra's algorithm to each node, when user input his current location the application can find the closest classroom from pre-calculated graphs. If used with API from [The HKUST Localization Project](#), user does not even have to input his address manually, this will greatly enhance the user experience.

8.1.2 Time-specific suggestions

Currently to prevent competitors from cloning the database, the application only show users available classrooms at the time. To loosen this restriction it is as simple as opening an API for this function because all data are stored in database – users' requests are readily fulfilled by database query.

8.2 HKUST room booking (UST 卜房)

8.2.1 Time-specific suggestions

In response to a critical comment, user should be able to select which day he is asking timeslots for. This change does not introduce any API change and only client-side interface and logic need to change.

8.2.2 Speed boost

One unavoidable performance bottleneck is *liba_lib* because it requests and parses resource from library system. While it is not possible to cast any change on the external library system, internally the parsing engine may be implemented in Python using *Scrapy* instead of PHP *ganon* library. The drawback is rather uncertain because then the backend will have to implemented in a 2-tier manner. (Fig. 18-19) Nevertheless, there is no guarantee that communication overhead between web server and python server would offset the speed boost from using Python to parse HTML.

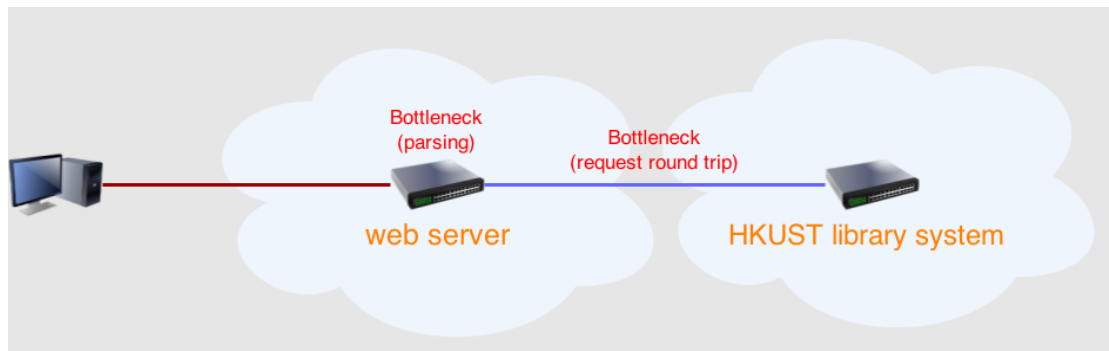


Fig 18. Current design

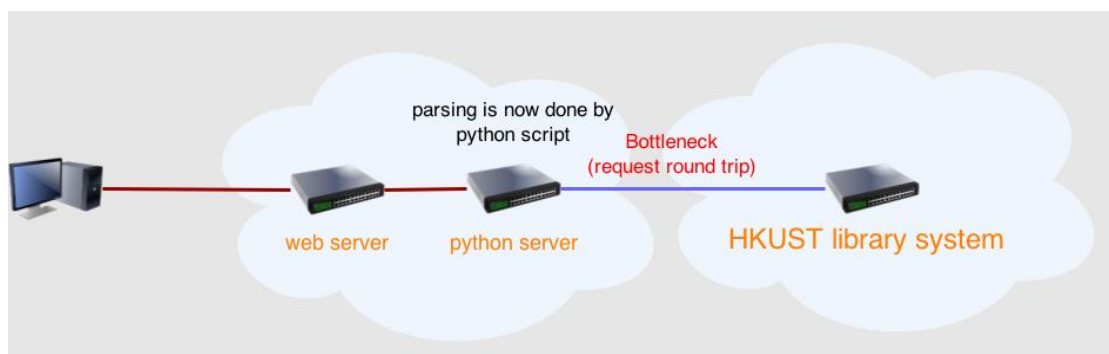


Fig 19. Design using Python to parse responses from library system

Although caching current calendar of HKUST library system seems to be a

promising solution, booking condition is constantly changing inside library system so each call has to be done in real-time. Therefore caching is not a feasible solution.

9. Conclusion

This project aims at investigating techniques to utilize API-less online services. As a result, two proof-of-concept applications, namely *HKUST "break in"* (UST 爆房) and *HKUST room booking* (UST 卜房), and a public API *liba_lib* are created. From which this project gathered experience in parsing static data set for another usage, develop wrapper to a service by mimicking server-browser interaction and using a wrapper library to develop third-party application to an online service that provides extra functionalities as an alternative to the original service.