STUDY MATE

-- Project report

Yongbing Yang

Role: Website design, Database design

Zhihan Mei

Role: Android app design, Server setup

10 Jun 2020
Version 1.0

BACKGROUND AND RATIONALE

A common phenomenon among students at the University of Auckland is that lots of students are having trouble with checking the daily class timetable. Most students choose to take a screenshot of the timetable and set it as wallpaper. This occupies the ability of students' personal choice, and triggered us to build a project that could provide another option for the students.

Besides, students at the University of Auckland is hard to get an objective appraisal for courses, most students only allowed ask for more detail at the student center or have an appointment with the taught teacher, even the students will not receive any announcements about the courses changes, in case compsci 399, I noticed this course by looking through the whole course list. Therefore, our team is keen to build a course comments area for students to feel free to leave their advice for other students who will take the same course in the future.

SPECIFIC AIMS

1. A portable mobile app that provides students with convenient timetable service.
2. A platform that allows students to post comments and ratings about UoA courses.

PROJECT APPROACH

1. Looking for an appropriate IDE to build an Android application.

   By research, Android app developers are mainly switching to Android Studio from Eclipse in recent years. Hence the Android Studio is selected with Java due to the reason that Java is covered previously in our COMPSCI courses

2. Design user interface and database for the application.

   Before starting coding the project, we use Adobe XD to design the user interface and user experience for the application, we draw the low-level prototype to discuss the layout and reactive for each page or content jump. Also, we use 'draw.io' to design the Entity-relationship diagram for our database to build the best database we can, like a good relationship between each table and how many attributes should be contained. In addition, we tried the Marvel before, then we found Adobe XD is more suitable for our requirement, Marvel is good with the high-level prototype but lower control for users. Adobe XD allowed the user to design all the things as the user exactly thought.

3. Building Timetable crawler

   Based on student's have their own unique course timetable, we decide to download the user's timetable from the university website after the first time the user logs into our application. Compared with other program languages, we choose python which is the lightweight yet powerful scripting language to build the timetable crawler for two reasons. Firstly, Python is the most familiar language with me because we learned python from Compsci 101, in addition, Python is good for web scraping, Scrapy and Beautiful Soup are among the widely used frameworks based on Python that makes scraping using this language such an easy route to take. Firstly we analyze the web elements for the web page by using the inspect function within Chrome, after we noticed the timetable page is the dynamic page, we decide to use Phantomjs which is the WebDriver to simulate a browser and waiting for the web page completely loaded then we start command crawler to acquire information. There are other good web drivers, why we choose Phantomjs is because Phantomjs is the headless web driver and we use Chrome(web driver) for testing code, we can see what happens after every single command.

4. Building multiple level menus for the Comment page

   After a few days of learning and discovering the basic Android structure, we decided to build the multiple level menu of the comment page first. This menu would allow users to switch between different courses in different subjects. With another week of learning the ListView and RecyclerView which are parts of Android components, we discovered that it is not easy to process this huge amount of course information so we decided to achieve the comment function with a form of website service.

5. Switching Comment page to web-based

   Implementing the comment function on the web not only avoids the technical difficulties but also increases the usage and frequency of use of this function. We have to consider it for those students who do not want to install our software or who are not convenient to use mobile phones, because we hope More students join us to view or leave precious comments or opinions. Building the website, we use the basic three technologies, HTML, CSS, and JS(javascript), also we use PHP to connect databases. The local environment is Wampserver 64. Of course, we uploaded the web page to the server after the local test was completed and published. In the construction of web pages, we put the user experience first and strive to make the operation simple and clear, and at the same time give users more "control" to a certain extent. Based on these we use a three-level menu to show the whole courses and we allow the user to change the rating at any time, moreover, users are able to see their comments immediately after each submitting.

6. Hosting the webpage on the server

   We chose the Google Cloud Platform (GCP) as our server hosting provider. GCP provides stable service with a free trial for 1 year. After searching and reading for relative articles, we deployed WordPress on our GCP server.

7. Using FTP to connect the server

   As we are trying to display our own website on the server, WordPress does not meet our requirements and we need to upload our own files to the server. By searching for tutorials we decide to use FileZilla to connect to the server through FTP.

8. Building MySQL database with Python

   Firstly, we are not using Mysql for the first time, SQLite is more lightweight which we thought it might be more suitable for our application. Then, when projects need to connect to the server to put everything on the cloud, we found SQLite is not supported to put on the server and website easier to connect to Mysql. We use ‘Navicat’ as our database visualization tool. Using Python because our target is to enable all students to use our application, we don’t want to only support limited faculty students. Then we needed all course information from the university website, so we did another Python crawler to import all course information from the website to Mysql. We build two databases, one at the localhost for testing another is on server to avoid database lose or crash accidents during website building.

9. Launching the timetable crawler to the App

   This is the most time-consuming part of our project. We found that the biggest challenge to us is that it will be much more complex if we wish to transfer our Python crawler with a webdriver to our Android app and run it locally. At this point, we chose to design another crawler with Java which fits the Android developing environment and would achieve the same function as Python crawler that could scrape dynamic websites. After a week of learning the JSoup module in Java, we noticed that a simple HTTP request sent by JSoup will not reach the university website and without a dynamic webdriver we will never be able to reach the goal.

10. Deploying the Python HTTP server combined with crawler to GCP

    With another week of struggling and researching, we chose to achieve this function on the server. It will be much easier if we run our crawler script in an environment that supports Python and transfer the timetable data to our app. We constructed an HTTP server using Python and combined it with our crawler that runs successfully at an earlier time. We used Nginx to link the internal HTTP server to our server's external IP address. After that, we are able to send requests and receive the raw data on the App side.

11. Transferring timetable data to a real timetable

    After we successfully obtain the timetable data from the server, the last thing we should do is to transfer these data to a real, visible timetable. Previously, a demo of TableLayout was set, after a few days of research, we found that doing dynamic changes to current layout requires much more sets of conditions. Hence we decide to generate the whole tab using our raw data. The final product that generates 5 tabs at the same time by pressing the "Display" button is designed for this.

TESTING

1. Crawler testing

   Crawler testing is mainly focused on achieving the function that scrapes timetable information on the SSO page with the user's upi and password.

   1st version: Selenium using PhantomJS on Python. Python script runs successfully but individually. By inputting the user upi and password, the crawler will return with a dictionary that stores the timetable information.

   2nd version: JSoup on Java. A version coded in Java using the JSoup module. After getting upi and password from user input, JSoup sends http requests to the server and tries to get timetable information.

3rd version: Python crawler combined with http server. Using python simple http server to set a response method for post requests and return the timetable dictionary as response body.

2. Comments function testing

   Comments function testing is testing the announcement should be a drop-down menu, and the rating star can be selected, changed, the comments can able to display

   1st version: Design a menu with Android built-in components to show whole university courses.

   2ed version: the rating stars can be changed from 0 to 5, the color of a selected star is orange, and unselected one should be empty, the mouseover should change the empty star to black but selected stars should not change.

   3rd version: the input enable enter information, click the submit button will despair the information and successfully update the database, at the same time, the past comments and current one should display at below, the latest one should list at the top.

3. Server testing

   Server testing is mainly focused on providing a stable container for our website and Python http server.

   1st version: Both website and Python http server are deployed successfully.

4. Timetable Display testing

   Timetable testing is aiming to achieve the functionality of displaying timetables using raw data from the server.

   1st version: Using TableLayout, data sets could not be easily imported dynamically.

   2nd version: After switching to GridLayout, the same problem of display still remains.

   3rd version: Using Tablayout with viewpager, we are able to import all the data to different tabs one by one with a much better visual representation that does not screw all labels in one screen.

CONTINUOUS IMPROVEMENT

1. Timetable crawler

   1st version deprecated: Not able to fit in Android's Java environment.

   2nd version deprecated: Since the login page of the university sets JavaScript on the "Login" button, a simple scraper is not able to perform this function and it is deprecated.

   3rd version adopted: Firstly, the http server built by Python is able to receive http requests and send responses successfully. Besides,

2. Comment function

   1) because of the large data, the courses list function is not fast enough, the web page might get slower when large data requested, we should finger out a more efficient way to let student found they're except course

   2) the rating star need be deleted if the user selected by mistakes

   3) the comments should able to delete by the user who made it

   4) users need to be able to rate useful comments, the highest rating one set as the recommend one and list at the top of the comments area

   5) Furthermore, we might consider linking users account to comments function, such that the enroll courses will be pre-loaded within the user's account

3. Timetable function

   1st and 2nd version deprecated: Difficult to add timetables to a specific cell dynamically.

   3rd version adopted: With a LinkedHashMap we are able to import all timetables by order and provide users with a much better experience.

PROJECT PLAN

Our project plan is not strictly followed. We spent a variety of time doing research and watching relative tutorials. The base framework of our application was built around week 5 and it is the time when other tests come. Based on only two people in our groups, restricted by communication (Covid-19). We are hard to work together and we both are beginners in application development, we need more time to solve the problems the main methods help users to solve the problem are google, youtube, or ask friends. Also, we got problems from our source, the university website. Because our project is based on the website, most of the information should be from the website. Firstly, because the Covid-19, University of Auckland decided to change to online teaching mode, so that the room location changes to online 'student service online', it is directly leading our team to cancel location function. From the original plan, the timeTable should be downloaded automatic but the current application has to ask the user to click 'refresh' to update classTable. Then as we mentioned before, the difficulty of building a three-level menu by java, we moved our course review function to the web views.