

國立中興大學科技管理研究所
碩士學位論文

MyNext：利用集體智慧開發適用於台灣研究所
招生考試之社群查榜系統

MyNext: A Collective Intelligence Enabled
System for Cross-Checking Entrance
Exam Results

指導教授：沈培輝博士 Dr. John Sum
研究生：蔡智強 Chih-Chiang Tsai

中華民國一百零一年七月

國立中興大學科技管理研究所
碩士學位論文

MyNext：利用集體智慧開發適用於台灣研究所
招生考試之社群查榜系統

MyNext: A Collective Intelligence Enabled
System for Cross-Checking Entrance
Exam Results

指導教授：沈培輝博士 Dr. John Sum
研究生：蔡智強 Chih-Chiang Tsai

中華民國一百零一年七月

National Chung Hsing University
Graduate Institute of Technology Management
Master Thesis

MyNext: A Collective Intelligence Enabled
System for Cross-Checking Entrance
Exam Results

Advisor: Dr. John Sum

Student: Chih-Chiang Tsai

Date: July, 2012

國立中興大學 科技管理 研究所

碩士學位論文

題目： MyNext: A Collective Intelligence Enabled System for Cross-Checking
Entrance Exam Results

姓名： 蔡智強 學號： 7099026101

經 口 試 通 過 特 此 證 明

論文指導教授

沈培輝

論文考試委員

陳同孝

沈培輝

張啟昌

中華民國 101 年 7 月 17 日

中興大學博碩士論文授權書

本授權書所授權之論文為立書人在 **中興大學管理學院 科技管理研究所**，100學年度第2學期取得碩士學位之論文。

論文題目：**MyNext：利用集體智慧開發適用於台灣研究所招生考試之社群查榜系統**

指導教授：**沈培輝**

授權事項：

- 一、立書人 **蔡智強** 同意無償授權 **中興大學** 將上列論文全文資料之以微縮、數位化或其他方式進行重製作為典藏及網際網路公開傳輸之用。 **中興大學** 在上述範圍內得再授權第三者進行重製或網際網路公開傳輸等其他利用。
- 二、立書人 **蔡智強** 同意無償授權教育部指定送繳之圖書館及 **中興大學** 將上列論文全文資料之紙本為學術研究之目的予以重製， **同意全本重製**，惟每人以一份為限。
- 三、立書人 **蔡智強** 同意 **中興大學** 以有償方式再授權資料庫廠商 將前條典藏之資料收錄於廠商之資料庫，並以電子形式透過單機、網際網路、無線網路或其他公開傳輸方式授權用戶進行檢索、瀏覽、下載、傳輸、列印等。立書人可選擇之權利金方案如下列二種擇一：
 - *由立書人收取有償授權之權利金，其權利金額度由 **中興大學** 與再授權之廠商議定之。
 - *由立書人將有償授權之權利金捐贈 **中興大學** 校務基金。
- 四、 **中興大學** 得將第三條之權利再授權予其他第三人進行網際網路公開傳輸等其他增值利用。
- 五、前四條授權均為非專屬授權，立書人仍擁有上述授權著作之著作權。立書人擔保本著作為立書人所創作之著作，有權依本授權書內容進行各項授權，且未侵害任何第三人之智慧財產權。如有侵害他人權益及觸犯法律之情事，立書人願自行負責一切法律責任，被授權人一概無涉。

論文紙本於 **中興大學** 圖書館內公開陳列上架時間：三年後公開

有償授權條件：享有權利金的回饋，權利金捐贈校務發展基金。

論文全文上載網路公開時間：三年後公開

立書人：**蔡智強**

簽名：



中華民國 101 年 8 月 9 日

誌謝

很快的兩年過去了，本論文也終於可以順利完成！在此要感謝許許多多促成本論文順利完成的人。首先就是最辛苦的 John。雖然我常讓你找不到人，電話也很難通，讓您很擔心，但是您還是每次都很有耐心的給予我不同的建議和新的研究方向。對我來說，您不但是位好老師，更是一位可以無所不談的好朋友。此外，也感謝擔任口試委員陳同孝老師以及張啟昌老師。因為你們的許多寶貴建議與指導，讓本論文可以更加完善。

謝謝跟我同班了兩年的同學們：一樣愛 Apple 的史蒂芬、愛耍帥的居米、像媽媽一樣的莊莊、講話很吵的恬恬、驚喜不斷的一節、上課最認真的 Joy、有 model 身材的小沾、來自泰國的王子 Ty、沒去到慕谷慕魚的洪書緯、吃不胖的賣口、脾氣很好的羅倫斯、還有籃球超強的冠軍隊長廖仲強。因為大家的幫忙和互相漏氣求進步，一起趕報告到天亮或者一起出遊玩樂，讓我兩年的研究所生活更加精采！另外還要感謝我的好學弟 Luke。我這個學長實在是很不盡責，一直到我快畢業前兩三個月才慢慢跟你混熟。但是你真的幫了我很多，可以讓我放心的衝刺論文。口試當天你也抽空出來幫忙我佈置場地、處理一些 paperwork 等等。另外還要謝謝美莉姊，雖然您很愛虧我，但是還好有您的細心提醒才不會讓我忘記選課而無法畢業。還有 Anna、Lili、元達、鴻康、Sony、W 以及其他科管所的夥伴們，因為有你們讓我在台中的生活能夠過得更順利。謝謝我的工作伙伴沈士棋這段時間的包容。謝謝 AB 的一些建議還有屎豪的幫忙測試。還有我許許多多的好朋友們，感謝你們一路上的關心與幫助。

再來要感謝的是一路都很支持我的 Jennie。雖然你當初為了我而在台北找了工作，可是我卻很不爭氣的考上台中的研究所。但是這兩年來，你總是在我身邊陪伴著我。不管是一同分享著喜悅或者是聽著我抱怨，有時候更會晚上一起陪著我忙到半夜。感謝這段時間你的包容和支持，讓我可以順利完成研究所的學業。

最後，我要感謝最愛我的家人們。感謝大舅舅讓我在台中念書的這兩年不用擔心住的問題。感謝同樣在中興念書的妹妹在學校的時候會不時的跑來關心我。感謝我的爸爸、媽媽，您們的支持讓我可以毫無後顧之憂的完成我人生階段中的一個小小里程碑。

謹此致上我最誠摯的謝意，謝謝。

Denny 中華民國一零一年七月於台中

摘要

在台灣，每年的春天是研究所考試的季節。由於各校分別舉行考試，因此考生通常會報考多所學校來增加自己錄取的機會。放榜時，表現好的學生便會出現錄取超過一間學校的重榜情形。對於備取的考生來說，交叉查榜是一個非常重要的工作。目前已經有一些線上服務可以提供備取生線上查榜。但是這些現有的服務都有著一些共同的缺點像是版面設計不良以及無法提供備取生額外的協助。本研究基於現有服務的一些缺失，開發了一套更先進的交叉查榜服務 - MyNext。MyNext 提供了一個全新設計的查榜介面讓考生更容易使用。運用 Facebook 社群網路，MyNext 可以讓考生更容易的與他們的朋友互相連結來取得查榜上面的協助。考生們和其他的使用者可以在 MyNext 上面交換意見、進行上榜預測、提供建議，備取生可以從這些資訊中來更加準確的推測上榜的機會。運用了集體智慧的精神，一些在現有系統難以解決的問題如考生同名問題，也可以藉由考生們的過濾來加以辨識。藉由與 Facebook 的整合及運用集體智慧的概念，考生們可以以合作的方式更有效率的交叉查榜，並且建立一個研究所考生的線上社群網路。

關鍵字：網路應用程式、研究所入學考試、交叉查榜、線上社群、集體智慧

Abstract

In Taiwan, graduate school entrance exams are conducted on Spring term each year and results are released shortly after the exams have finished. Students could thus browse the corresponding school websites for the releases. In recent years, some online cross-checking systems have been developed for one-stop checking. However, these systems are usually badly designed and they do not provide information on if a short-listed student will give up the offer. In this regard, we present in this paper an advanced cross-checking system called MyNext. MyNext provides three major functions. First, it provides an interface for a student to cross-check the lists. Second, by connecting with Facebook, an interface has been developed to let the students seek advices from their friends. The interface allows their friends leaving comments and giving recommendations. This collective intelligence (combining comments, recommendations and other advices) would definitely be beneficial to a student on justifying how likely he/she will be admitted to the applied programs. Besides, friends can tell if the applied programs are really suitable for the student. Finally, owing to resolve the problem of two or more students with the same name, a simple function has been developed to let a student to cross out the programs he/she has not applied. In such case, the system could distinguish who is who on the list and thus make cross-checking more effective and reliable.

Keywords: Web application, graduate school entrance exam, cross-checking, online social network, collective intelligence

Contents

誌謝	i
摘要	ii
Abstract	iii
Contents	iv
List of Figures	vii
List of Tables	viii
List of Listings	ix
1 Introduction	1
1.1 Project Goals	3
1.2 Organization of the Thesis	5
2 Background	6
2.1 Cross-Checking System (CCS)	6
2.2 Collective Intelligence	7
2.2.1 Wiki	8
2.2.2 Social Bookmarking	9
2.2.3 Recommendation	10
2.2.4 Online Dating	10
2.3 Social Networking Service (SNS)	11
3 Services of MyNext	12
3.1 The Use-Case of MyNext	12
3.2 Workflow of the Use-Case	14
3.2.1 Browsing Lists by Program	14
3.2.2 Searching for a Name	15

3.2.3	Voting for Possible Outcome	15
3.2.4	Filtering Common Name	17
3.2.5	Leaving Comment to a Program	17
4	System Architecture	19
4.1	System Design	19
4.1.1	Administrative Operation	19
4.1.2	Three-Tier Architecture of MyNext	19
4.2	Infrastructure	21
4.2.1	Hardware	21
4.2.2	Operating System	22
4.2.3	Web Server	22
4.2.4	Database Server	22
4.2.5	Web Application Framework	23
4.2.6	Client-Side Scripting	23
4.2.7	The MyNext Stack	23
4.3	Data Models	24
4.3.1	User model	24
4.3.2	FBAccount model	24
4.3.3	School model	26
4.3.4	Program model	26
4.3.5	Record model	26
4.3.6	Vote model	26
4.3.7	Filter model	26
4.3.8	Correction model	26
4.3.9	Comment model	27
4.3.10	Underlying Datastore	27
4.4	Program Design	27
4.4.1	Parsers	27
4.4.2	Object Generator	28
4.4.3	Batch Update Module	28
4.4.4	List Browsing Module	29
4.4.5	Multiple Offer Getter Module	29
4.4.6	Search Module	30
4.4.7	Watchlist Module	30
4.4.8	Prediction Voting Module	30
4.4.9	Comment Module	31
4.4.10	Common Name Filtering Module	31

4.4.11	Intelligent Prediction Module	33
4.4.12	Facebook Authentication Module	33
5	User Interface	36
5.1	Homepage	36
5.2	Login Page	37
5.3	School Page	38
5.4	Program Page	38
5.5	Search Results Page	41
5.6	Admin Console	41
5.7	Usage Examples	41
5.7.1	Searching for a Student	42
5.7.2	Using the Intelligent Prediction	43
6	Conclusions	45
	References	48

List of Figures

1.1	Activity Diagram of the Fill-Up Process.	2
1.2	Popular Online Cross-Checking Systems.	4
2.1	Duplicate Results of Daso.	7
2.2	Distracting Banners on iCross.	8
2.3	A Common Name Example.	9
2.4	Examples of Modern Web applications with Collective Intelligence.	10
3.1	Use-Case Diagram of MyNext.	13
3.2	Sequence Diagram of Browsing Lists by Program.	15
3.3	Sequence Diagram of Searching for a Name.	16
3.4	Sequence Diagram of Voting for Possible Outcome.	16
3.5	Sequence Diagram of Filtering Common Name.	17
3.6	Sequence Diagram of Leaving Comment to a Program.	18
4.1	Flow Chart of the Administrative Operation.	20
4.2	The Three-Tier Architecture of MyNext.	21
4.3	The MyNext Stack.	24
4.4	Class Diagram of MyNext.	25
5.1	Screenshot of the Homepage.	36
5.2	Screenshot of the Watchlist.	37
5.3	Screenshot of the Login Page.	37
5.4	Screenshot of the Facebook Login Page.	38
5.5	Screenshot of the School Page.	39
5.6	Screenshot of the Program Page.	40
5.7	Screenshot of the Search Results Page.	41
5.8	Screenshot of the Admin Console.	42
5.9	Illustration of Searching for a Student.	42
5.10	Illustration of Using the Intelligent Prediction.	44

List of Tables

4.1	System Infrastructure of MyNext.	22
4.2	List of the Programs.	28
6.1	Feature Comparison Table of MyNext, Daso and iCross.	46

List of Listings

4.1	Pseudocode of Parsers.	28
4.2	Pseudocode of Object Generator.	29
4.3	Pseudocode of Batch Update Module.	29
4.4	Pseudocode of List Browsing Module.	29
4.5	Pseudocode of Multiple Offer Getter Module.	30
4.6	Pseudocode of Search Module.	30
4.7	Pseudocode of Watchlist Module.	31
4.8	Pseudocode of Prediction Voting Module.	31
4.9	Pseudocode of Comment Module.	32
4.10	Pseudocode of Common Name Filtering Module.	32
4.11	Pseudocode of Intelligent Prediction Module.	34
4.12	Pseudocode of Facebook Authentication Module.	35

Chapter 1

Introduction

In Taiwan, graduate schools take new admissions in a very unique way. There are no standardized exams like the Graduate Record Exam (GRE) or the Graduate Management Admission Test (GMAT) (Keith-Spiegel & Wiederman, 2000). Instead, entrance exams are held independently by different schools. The exams are usually conducted during the Spring term each year and students can take any number of exams in accordance with their interests and time schedules. The results of the exams are released shortly after in the format of short-lists and waiting-lists. For students on the short-list, they will be requested to indicate if they accept or turn down the offer before a specific deadline set by each school. If a student declines the offer, the school would then fill up the vacancy by the student on the top of the waiting-list. Same question will be asked (usually over the phone) by the school. If the fill-up student declines the offer, the next student on the waiting-list will be asked for their willingness to take the offer. The process is repeated until either (i) the quota of student intakes has been filled or (ii) there are no more students on the waiting-list. Figure 1.1 is a diagram illustrates the fill-up process.

During the fill-up process, some schools will update the lists to reflect the latest changes. In such cases, students on the waiting-list can follow the status of the fill-up process. However, only a few schools will provide these updates and even if they provide updates to the lists, the frequency of updates is usually unpredictable. Normally, the fill-up process can take months, and in some extreme cases it can even last till just a few days before the new school year begins. During this indefinite period of time, students on the waiting-list could only wait and pray for the best. For those students do not want to just wait and pray, they could check out the students on the lists and see if they get offers from other programs. By gathering this information, students can make assumptions on how likely they can be admitted. The process of gathering information on students with multiple offers is called

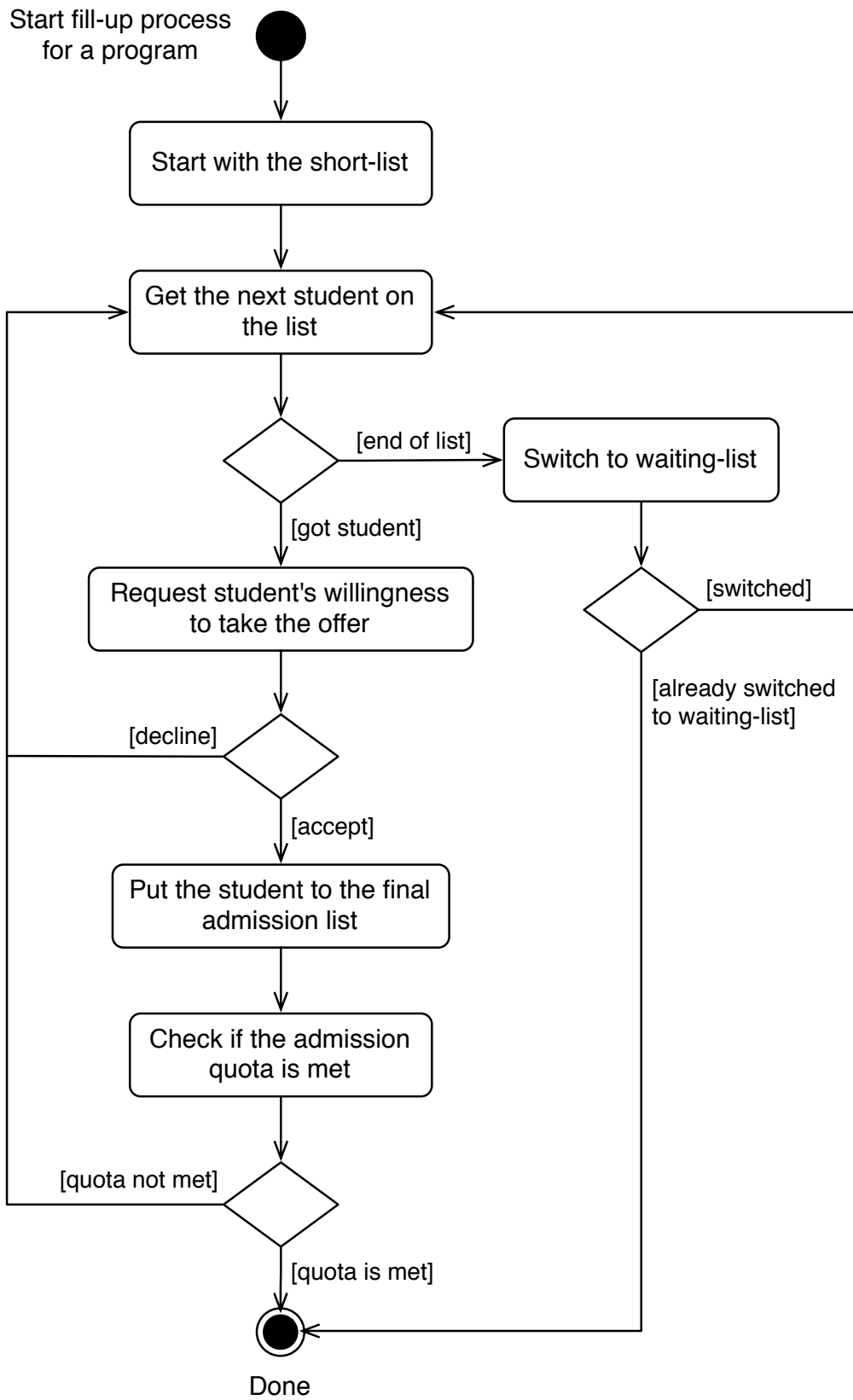


Figure 1.1: Activity Diagram of the Fill-Up Process.

cross-checking. Cross-checking is like searching over a big tree structure with the student himself/herself as the root. Clearly, it is a tedious and time consuming task as students have to check from each individual school to collect all the information. Add that to the anxiety of the students, the process is more stressful than one could possibly imagine.

To relief the pain of cross-checking, some online cross-checking systems (CCS) have been built and make available for the anxious students. Two of the most popular CCSs, Daso and iCross, are shown in Figure 1.2. However, these systems share some common shortcomings including (i) inadequate page layouts, (ii) unable to distinguish common names and (iii) provide little information to help students on waiting-lists to predict how likely they will get admitted. The details of these systems and their shortcomings will be discussed in the next chapter.

1.1 Project Goals

In view of the shortcomings of current CCSs, we are going to implement a more powerful CCS called MyNext. MyNext is able to (i) provide the necessary functions for cross-checking, (ii) provide user-friendly and simple interface, (iii) able to resolve common names and other disambiguations and (iv) provide assistance for students to determine their chances of getting admitted. To accomplish to first two goals, existing web technologies can be readily applied. To accomplish the last two goals, the concept of collective intelligence is applied. Collective intelligence is a shared or group intelligence formed when collaborating and making decisions. By letting students to form a collective intelligence on exam results, cross-checking is no longer an one-man activity, but rather a collaborative effort. However, none of the existing systems have embraced the advantage of collective intelligence.

Online social networking services (SNS) have forever changed how we communicate and interact. This is especially true for college students. By integrating with the most popular SNS, Facebook, friends of the students on lists can add comments, make recommendations, and vote for the programs that are the best for the students on the lists. These pieces of information can be regarded as the collective intelligence of exam results formed by students and their friends which can use to provide further assistance on cross-checking. The concept of collective intelligence will be further explained in the next chapter as well.

Teso Group 大碩研究所 www.daso.com.tw

全國研究所線上交叉查榜。

101年研究所榜單線上快速交叉查榜，提供您全國最快研究所交叉查榜！

最新研究所考試資訊

新功能：Smart全方位交叉查榜
如曾直接查詢考取學校系之全部考生姓名，或相關考生的所有考取學校，可點選右側連結做快速交叉查榜！

手機版上線了！

依學校查詢(一般生/在職專班)
依學校查詢(甄試)
依學校查詢(轉學考)

依學校查詢(一般生/在職專班) | 依學校查詢(甄試) | 依學校查詢(轉學考)

依關鍵字查詢原始榜單(初試及複試榜單)
選擇年度：2012
研究所(一般生和在職專班)
甄試生 轉學考
關鍵字： 查詢

原始查榜系統採純文字原始榜單比對查榜

功能說明：
1. 可查詢研究所及轉學考各類原始榜單。
2. 原始呈現個人考取正取、備取記錄。
3. 輸入關鍵字，即可查榜。
(可輸入姓名、准考證、學校、系所...等)

依考生資料查詢榜單(錄取榜單)
研究所(一般生和在職專班) 研究所甄試生
轉學考
准考證號碼：
應考者姓名： 陳冠宇 查詢

線上查榜系統採資料庫快速查榜

功能說明：
1. 預設為本年度交叉查詢榜單。
2. 依查榜的資料，可直接點選整個系所完整的榜單，不必重複查詢。
3. 依查榜的資料，點選系所榜單後，可直接點選榜單裡其他考生之姓名、准考證查詢正取、備取記錄，以供擷測上榜機率。

大碩研究所 8,520 likes

如果您覺得本系統方便好用，請按讚給我們一點鼓勵，謝謝！

公告日期	年	考試類別	學校首頁連結	初/複試	原始榜單連結
2012/07/09	2012	[備大] 一般生	國立台北藝術大學	複試	正/備取名單
2012/07/09	2012	[備大] 一般生	國立新竹教育大學	複試	正/備取名單
2012/06/18	2012	[研] 在職專班	私立中山醫學大學	複試	正/備取名單
2012/06/18	2012	[研] 一般生	私立南開科技大學	複試	正/備取名單

YSE! 洋碩美語
最HOT的 TOEIC 金色證書
2012含金量最高的外語課程
等你來報名!

填表報名六項課程送 研究所口試 無敵錦囊電子書
進修研究所，讓你一次學會三種超能力

(a) Daso

台一國際投資顧問股份有限公司 全國第一家 獲內政部移民註冊登記證
http://www.visaworld.com.tw 移民資訊免費索取 請進入...

台一移民留學 TEL:02-25082431 E-mail:teic@visaworld.com.tw

101年最新全國勞工就業培訓 年滿16歲之中華民國國民均可申請，多種免費電腦課程網路報名中，www.konnet.com.tw
碩博士論文指導-全國考選 專精論文寫作指導、英文文獻檢索指導、作業報告、統計分析、經驗豐富、歡迎諮詢 www.ckpaper.com.tw
OhtStudy 留學打工 研究所、大學、學院、語言課程、一週讀書一週打工、免費代辦、服務好禮，www.oht Google 學術的寶典

101學年度碩士班考試

國立台灣大學	台灣聯合大學系統	國立交通大學	國立清華大學
國立成功大學	國立中正大學	國立中央大學	國立高雄大學
國立陽明大學	國立嘉義大學	國立政治大學	國立中興大學
國立中山大學	國立東華大學	國立台南大學	國立台北大學
國立宜蘭大學	國立聯合大學	國立暨南國際大學	國立台灣海洋大學
國立台灣藝術大學			
逢甲大學	淡江大學	東吳大學	東海大學
靜宜大學	中原大學	元智大學	銘傳大學
世新大學	大同大學	輔仁大學	長庚大學
文化大學	台北醫學大學	中國醫藥大學	高雄醫學大學
國立台灣科技大學	國立台北科技大學	國立高雄第一科技大學	國立雲林科技大學
國立高雄應用科技大學	國立虎尾科技大學	國立台中科技大學	國立屏東科技大學
國防醫學院	國立台北護理健康大學		
國立台灣師範大學	國立高雄師範大學	國立彰化師範大學	國立台北教育大學
國立新竹教育大學	台北市立教育大學	國立台中教育大學	國立屏東教育大學

Mac 运行缓慢?
使用 MacKeeper, 速度可加快 25% 以上

加快 Mac 运行速度

(b) iCross

Figure 1.2: Popular Online Cross-Checking Systems.

1.2 Organization of the Thesis

The rest of this thesis is divided into five chapters. Chapter 2 will elaborate more about CCSs and their shortcomings. The concept of collective intelligence will be introduced and how social networking services facilitate collective intelligent activities are discussed. Chapter 3 will elucidate the services delivered by MyNext. Its potential users are identified. To ease the presentation, the services will be described in term of use-cases. Their workflows are elaborated by using sequence diagrams. The system architecture including the system design, the infrastructure supporting the services and the data model will be presented in Chapter 4. Chapter 5 will describe the user interface of MyNext. Accompanied by screenshots, how students can use the system will be described. Most importantly, how the system can facilitate collective intelligence will be illustrated. Finally, Chapter 6 will draw some conclusions of the project, and discuss about some possible future developments of the system.

Chapter 2

Background

In this chapter, we will first describe some of the existing cross-checking systems. Their shortcomings will be highlighted. Second, the concept of collective intelligence will be introduced. Finally, how social networking services can facilitate collective intelligent activities will be described.

2.1 Cross-Checking System (CCS)

An online cross-checking system is a system that can help students to cross-check. These systems work by storing all the lists in the databases and joining the results together by students' names. Once a student has queried for exam results of a specific program, the system not just returns the short-list and waiting-list of the requested program, but also lists out all the multiple offers for each student on the lists. Two of the most popular CCSs are Daso and iCross. Daso is also the name of the famous cram school who built the system. While these systems have facilitated the students searching for the lists, they share some common shortcomings.

The first shortcoming is that their table-based layouts are not simple enough to read. For Web applications, a clean and uncluttered user interface is preferred over messy and overwhelming ones (Friedman & Lennartz, 2009). One of the conventional CCS, Daso, has a confusing layout that seems to just pile a bunch of things together. Also, instead of just providing the results from the database, they also allow students to search from the raw lists. So the returned results are actually duplicated sometimes. Figure 2.1 shows the duplicate searching of Daso. iCross, on the other hand, contains some really big banner advertisements at the top of each page so students have to scroll down for the content that they are interested in. There are also advertisements placed in the middle of the page as well, which can further interfere with browsing. The problem with iCross is illustrated in Figure 2.2.

Figure 2.1: Duplicate Results of Daso.

Furthermore, both CCSs list the results in a table format which can make the list look terribly long and overwhelming.

The second shortcoming is that these systems are unable to distinguish common names. People with same names are very frequent in Chinese language while some names are more likely to appear than others. These “Market Names” (Taiwanese slang term) or common names can cause confusion when grouping related records by students’ names. They would just list all of the records together under the same name without considering if they are actually belong to different students. Figure 2.3 shows a common name with more than 60 records but clearly belong to more than one person due to the diversity of programs.

The third shortcoming is that these systems provide little information to help a student on the waiting-list to predict how likely he/she will get admitted. They just merely repeat the results from the original lists with a single feature added, cross-checking. They do not provide any help on how likely a student on the short-list would give up an offer. What students on the waiting-list want to know from cross-checking are their chances of being admitted and these systems fail to provide assistance. So after cross-checking, students on the waiting-lists still have to figure out on their own about their chances. This makes cross-checking not very efficient.

2.2 Collective Intelligence

Collective intelligence is a shared or group intelligence formed when collaborating and making decisions (United Nations, 2008). Traditionally collective intelligence refers to groups of individuals acted collectively in ways that seem intelligent. It is a concept of solving problems collaboratively. Nowadays, collective intelligence



Figure 2.2: Distracting Banners on iCross.

can be best shown by the rise of modern Web applications. Many modern Web applications are built with collective intelligence in mind (O'Reilly, 2007). By applying collective intelligence, modern websites not only let their visitors consume the information on the site, but also let them share and contribute. The following are some forms of the collective intelligence found on modern web applications.

2.2.1 Wiki

A wiki is a web application that all of its content is created and modified by its users (Wagner, 2004). The content of wikis can vary from knowledge to note-taking. There are also several expert-moderated wikis on different areas such as medical wikis (Barsky & Giustini, 2007). Wikipedia is currently the largest wiki encyclopedia in the world. Visitors from all around the world have created hundreds of thousands of pages containing valuable knowledge to just about everything.

您現在所在的位置:[首頁](#)>[查榜系統](#)>[資料庫查榜](#)>

准考證號碼：
應考生姓名：陳冠宇

年	准考證號	姓名	學校	科系(系所榜單)	組別	類別	初(複)試	錄取別
2012	515000365	陳冠宇	國立臺灣科技大學	資訊工程系碩士班		一般生	複試	備取 94
2012	21810157	陳冠宇	國立政治大學	國家發展研究所	選考經濟學	一般生	複試	備取 8
2012	201441210005	陳冠宇	國立高雄大學	統計學研究所	統計組	一般生	複試	備取 12
2012	05090238	陳冠宇	國立清華大學	化學工程學系		一般生	複試	正取
2012	321010297	陳冠宇	國立中央大學	化學工程與材料工程學系	甲組	一般生	複試	正取
2012	522000718	陳冠宇	國立中央大學	資訊工程學系		一般生	複試	備取 213
2012	201222210044	陳冠宇	國立高雄大學	政治法律學系	公法組	一般生	複試	正取 2
2012	724010073	陳冠宇	國立中央大學	法律與政府研究所	法律組	一般生	複試	備取 11
2012	429000152	陳冠宇	國立中央大學	會計研究所		一般生	複試	備取 32
2012	539150003	陳冠宇	國立臺灣大學	醫學工程學研究所碩士班	丁組	一般生	複試	正取 5
2012	431330020	陳冠宇	國立中山大學	電機工程學系碩士班	丙組	一般生	複試	正取
2012	502100052	陳冠宇	國立臺灣科技大學	電子工程系碩士班	甲組	一般生	複試	正取 22
2012	502300196	陳冠宇	國立臺灣科技大學	電子工程系碩士班	乙二組	一般生	複試	備取 6
2012	5413045	陳冠宇	國立成功大學	電腦與通信工程研究所	甲組	一般生	複試	備取 22
2012	D7100733	陳冠宇	國立中興大學	資訊科學與工程學系		一般生	複試	備取 98

Figure 2.3: A Common Name Example.

2.2.2 Social Bookmarking

Social bookmarking services such as Del.icio.us are also a form of collective intelligence known as the folksonomy system. Folksonomy is a system of classification derived from the tags created by users to categorize and organize different pieces of information (mainly different websites and links) (Peters, 2009). By using social bookmarking, not only users can access their favorite websites from anywhere, but also users are able to discover similar websites more easily due to the power of folksonomy (Halpin, Robu, & Shepherd, 2007).

2.2.3 Recommendation

By collecting users opinions and comments, recommendations are commonly found on electronic commerce websites. One of the strongest points of Amazon, the largest online store in the world, is its powerful recommendation system. The review system let its customers to share the experiences they have with the products. These reviews can become the determining factor when a potential customer makes his/her buying decisions (Linden, Smith, & York, 2003).

2.2.4 Online Dating

Online dating can be considered as the earlier form of the current social networking services. Online dating services utilize data mining techniques to categorize their users into different groups (Heino, Ellison, & Gibbs, 2010). Unmoderated matchmaking is the key feature provided by online dating services. The system can match people with similar interests together by the information gathered from user profiles. Some of the leading online dating services include AOL Personals, Yahoo! Personals and Match.com.

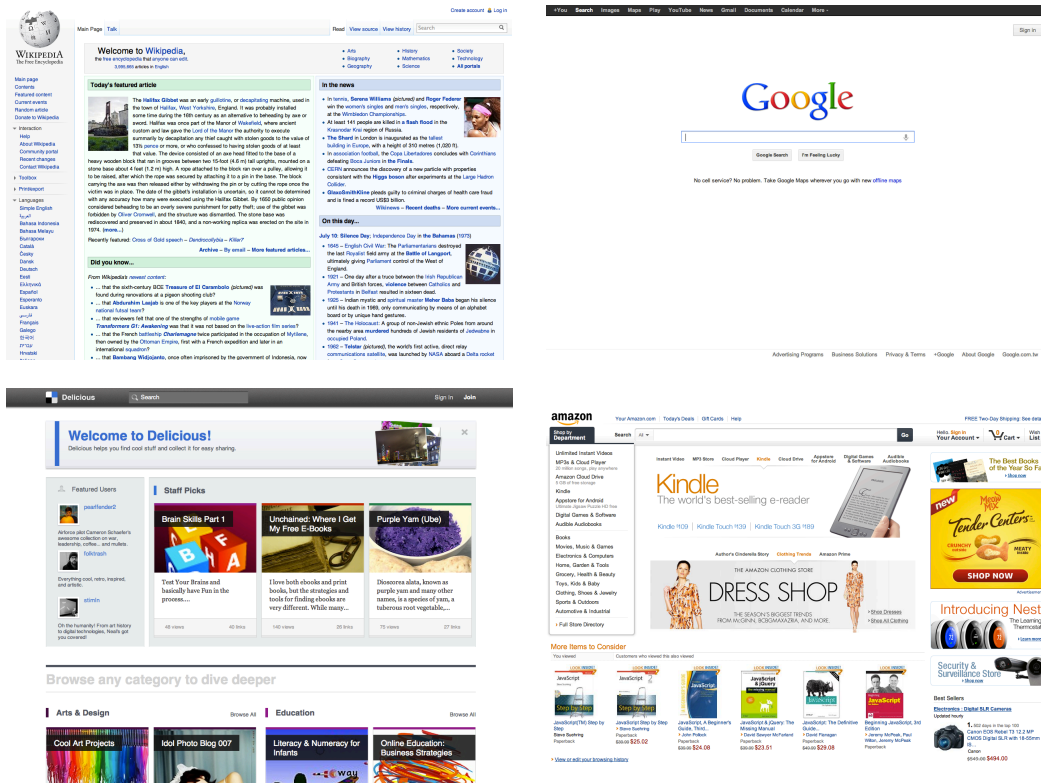


Figure 2.4: Examples of Modern Web applications with Collective Intelligence.

2.3 Social Networking Service (SNS)

A social networking service is an online platform that focuses on building social relationships among people who share interests, activities, backgrounds, or real-life connections. SNSs can also be considered as a form of collective intelligence. Typically, a SNS consists several features such as (i) categorizing users into different groups by their regions, former schools, or working environments, (ii) recommending new friends based on the classification mentioned and (iii) providing a systematic way for users to share their thoughts and generate content by using text, images, videos and other types of media. Facebook is the most widely used SNS in the world. It consists of over 901 million active users all around the world. It is also the most popular SNS among students (Pempek, Yermolayeva, & Calvert, 2009). Because of its wide adoption, Facebook is also considered the most powerful platform to promote new ideas and spread information. By bringing SNS to education, students can gather information faster, and their voices can be heard. There are already many applications of using Facebook on education such as building discussion forums and study groups.

Chapter 3

Services of MyNext

3.1 The Use-Case of MyNext

A CCS is necessary to have the ability to (i) provide the necessary functions for cross-checking, (ii) provide user-friendly and simple interface, (iii) able to resolve common names and other disambiguations and (iv) provide assistance for students to determine their chances of getting admitted. MyNext is designed to accomplish all the above goals. There are three main groups of MyNext users: students, friends, and administrators. Figure 3.1 illustrates the use-case of the system.

MyNext provides students to browse the lists of programs offered by each school and search for names in all lists. These basic functions do not need students to login. In order to have access to more advanced functions of the system, students are required to login using their Facebook accounts. After students logged in, the system will then get their friends list and check if their friends are also using the system. Students then gain access to functions like personalization, suggestions, and list corrections. Suggestions is the main feature that stands out from other CCSs because it provides a prediction of how likely a student on the waiting-list can get the offer. List corrections is also another use of the collaborative effort to correct the lists since occasionally there are typos or missing characters in the names. Students can also vote down potential common names to make the multiple offers display more meaningful and realistic.

Students' friends can use MyNext to help the students and also help the system to make more accurate and meaningful suggestions. The systems provides several functions for the friends of the students. First, they can vote for the best outcome they suggest for the students. For example, if student A has got multiple offers from five different programs, his friends can vote among those programs which they think is the best for him. Second, they can vote for the possible outcomes for other

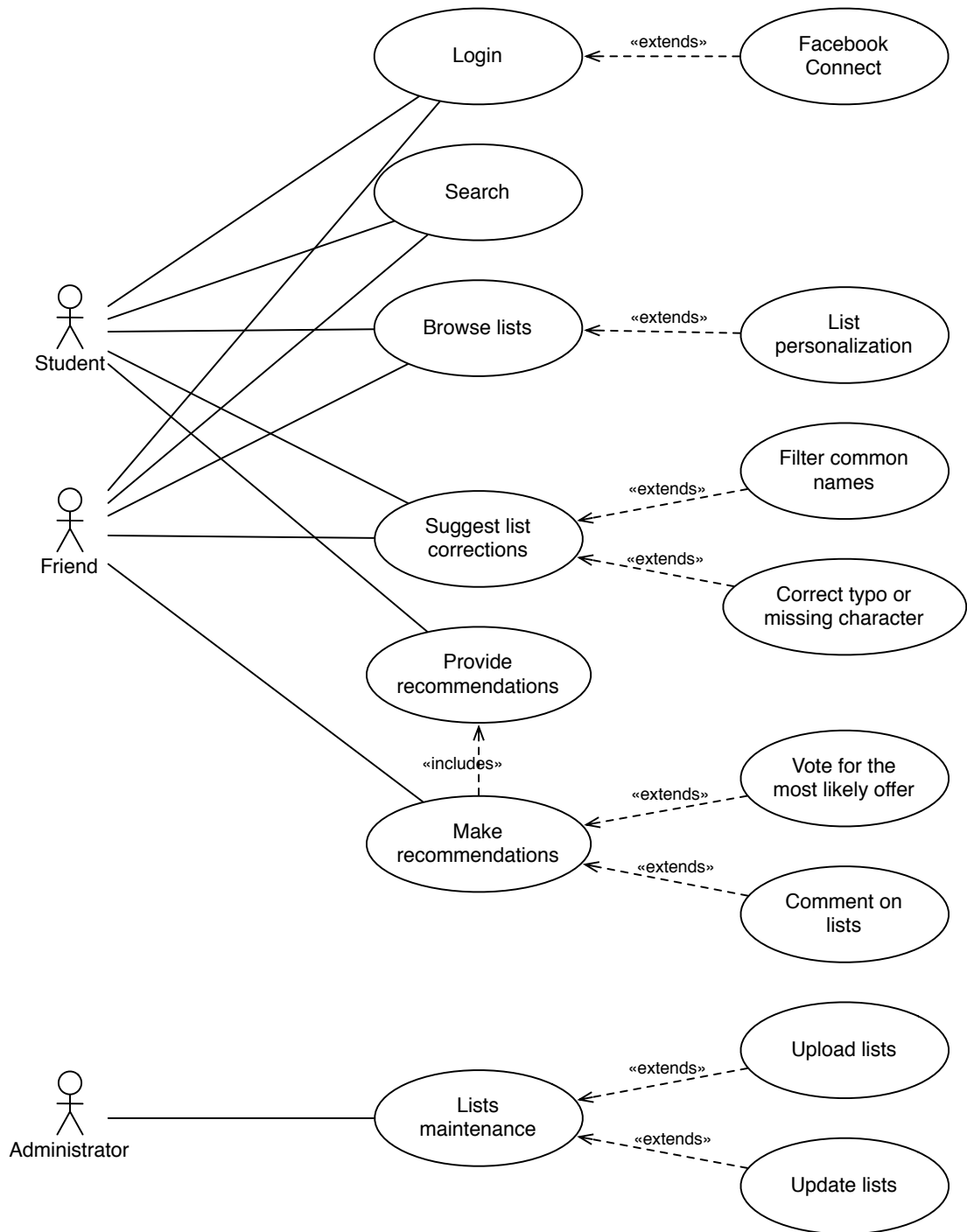


Figure 3.1: Use-Case Diagram of MyNext.

students on the list. Continuing from the example above, based on the observations of the lists, personal opinions, and probably some known facts about the multiple offers of other students on the list, friends of student can vote for their predictions on the possible outcomes of other students on the list. This can also help student A to eliminate other students that are before him on the list. Finally, friends can also help to improve the lists by making corrections and filtering common names.

The third group of MyNext's users are the administrators. Due to the fact that there are just too many formats for the results (mostly one for each school), some manual processes are needed to import the lists into the system. An administrator needs to obtain the lists released by each school in PDF format and upload those files to the system. Next, he/she must observe the list format patterns and input the patterns to the system. Finally, he/she runs the command to parse and generate the records into objects and store them in the system datastore from the admin console.

The detail information of each function and program will be explained in the following chapters.

3.2 Workflow of the Use-Case

In this section, we will describe the main functions of MyNext in detail. The main functions of the system are (i) browsing lists for a program, (ii) searching for a student's name, (iii) voting for possible outcome for a student, (iv) filtering common names and (v) leaving comments to a program. The implementations and designs of specific programs mentioned in this section will be discussed in the later chapters.

3.2.1 Browsing Lists by Program

The sequence diagram of this function is illustrated in Figure 3.2. An user (whether a student or a friend) can request lists for a specific program from the website by clicking on the program link (user interface). The interface will then process the user interaction, and send out the request to the application programs. The application programs will tell the Record entity to get a list of records whose program is the requested program. The Record entity will return the records requested. The application programs will format the records into a list and pass it back to the interface. The interface will then display the records on the page, and the user will finally see the lists for the program.

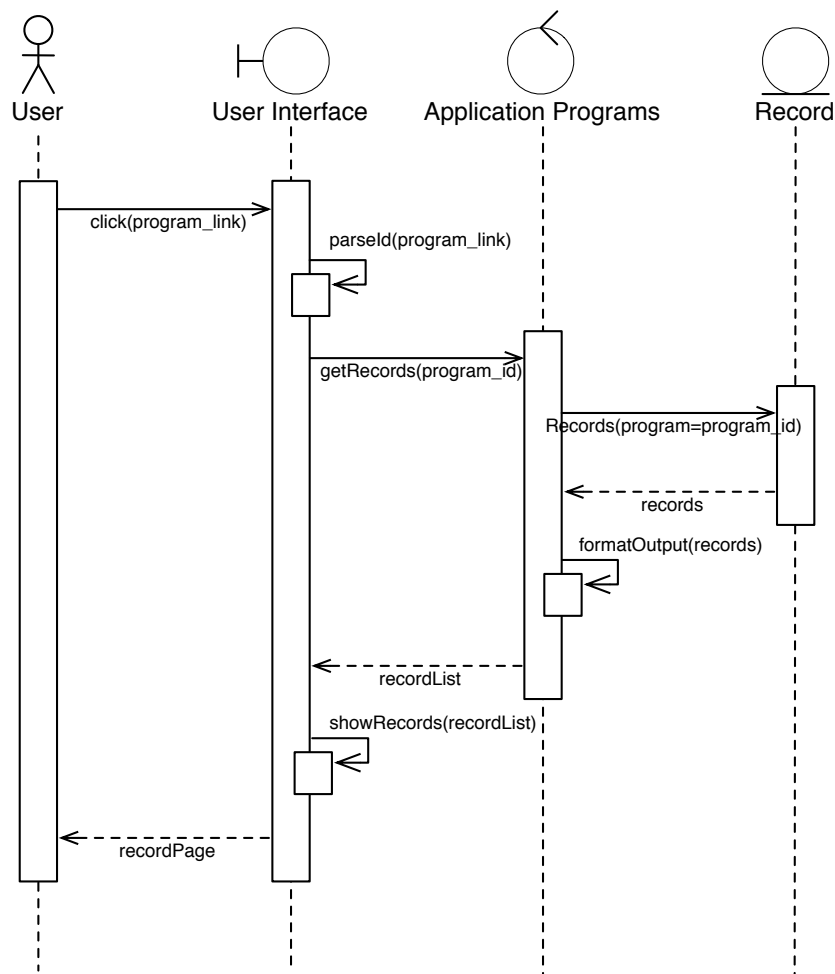


Figure 3.2: Sequence Diagram of Browsing Lists by Program.

3.2.2 Searching for a Name

The sequence diagram of this function is shown in Figure 3.3. An user (whether a student or a friend) input a name into the search field. The interface will send a search query with the name entered by the user to the application programs. The application programs will request a list of records whose name is the requested name from the Record entity. The Record entity will return the matching records. The application programs will format the records into a list and pass it back to the interface. The interface will then display the records on a search results page for the user.

3.2.3 Voting for Possible Outcome

The sequence diagram of this function is displayed in Figure 3.4. An user (whether a student or a friend) casts a vote on his/her prediction of possible outcome for a

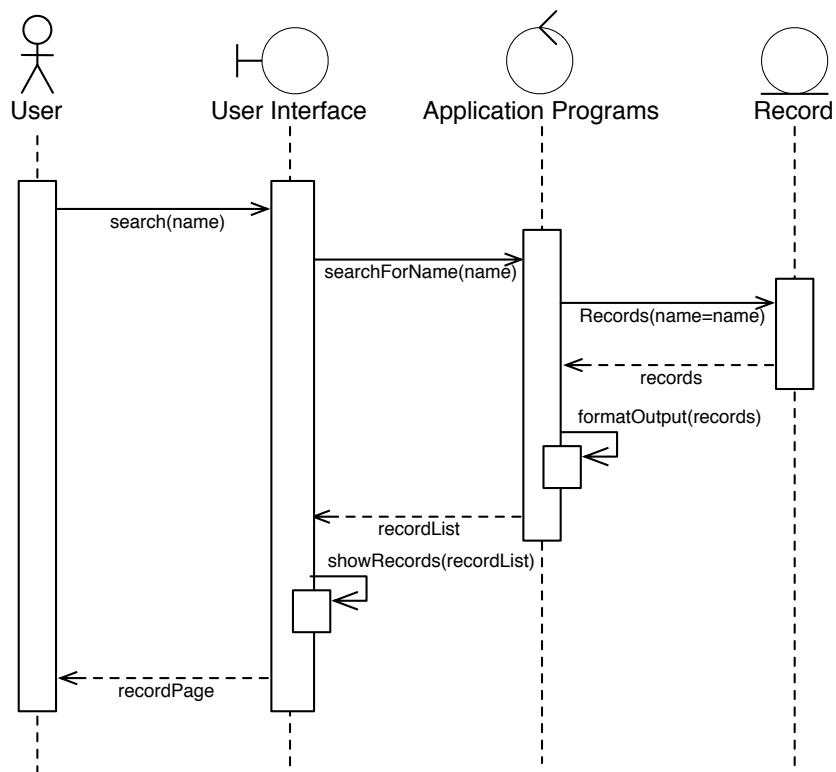


Figure 3.3: Sequence Diagram of Searching for a Name.

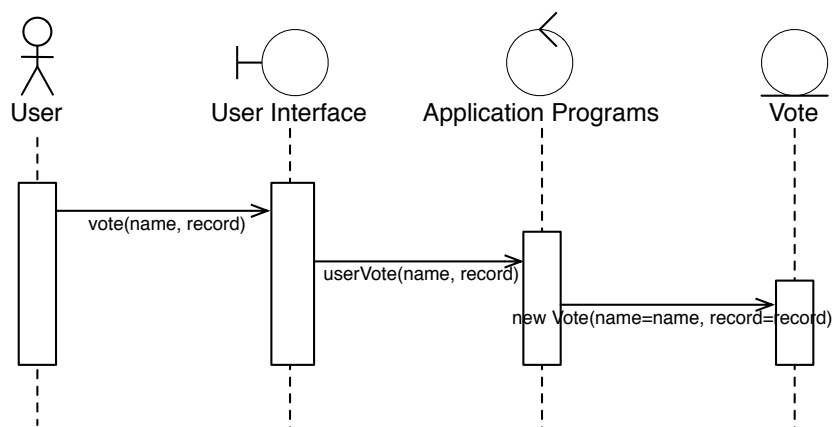


Figure 3.4: Sequence Diagram of Voting for Possible Outcome.

student. The interface will send a request with the user's vote to the application programs. The application programs will create a new Vote containing the name of the student and the outcome voted by the user. Since this function is done asynchronously, there are no return values.

3.2.4 Filtering Common Name

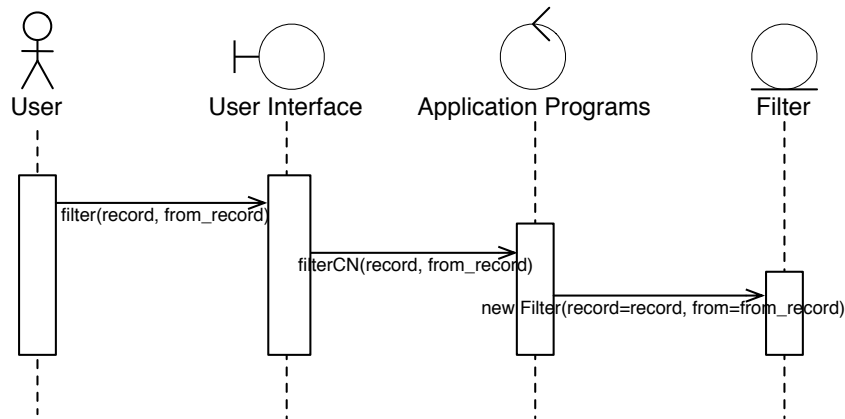


Figure 3.5: Sequence Diagram of Filtering Common Name.

The sequence diagram of this function is illustrated in Figure 3.5. An user (whether a student or a friend) filters out a record from a student's multiple offers. The interface will send the filter with the filtered record and the source record to the application programs. The application programs will create a new Filter containing the filtered record and the source record that it was filtered from. Similar to the voting function, this function is also done asynchronously.

3.2.5 Leaving Comment to a Program

The sequence diagram of this function is shown in Figure 3.6. An user (whether a student or a friend) leaves a message in the comment section of a program. The interface will send out a request with the message and the program to the application programs. The application programs will create a new Comment with the message and the program that the message was left on. This function is also asynchronous, so there are no return values.

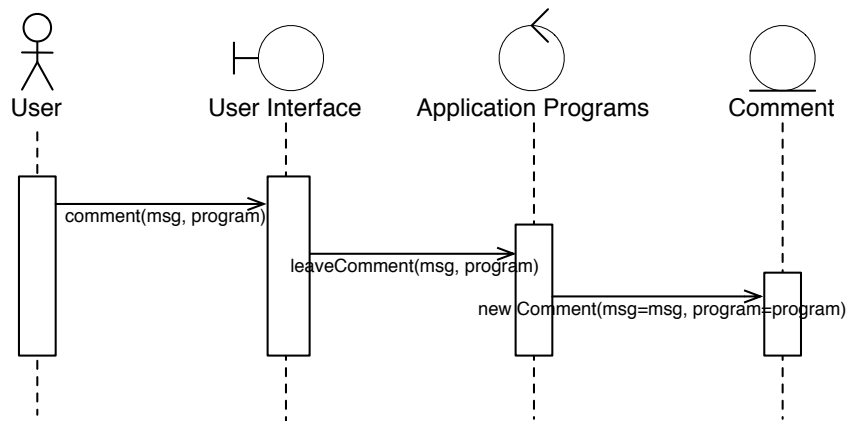


Figure 3.6: Sequence Diagram of Leaving Comment to a Program.

Chapter 4

System Architecture

4.1 System Design

The operations of MyNext can be split up into two parts: administrative operation and the main three-tier architecture of website operation. Administrative operation is mainly list maintenance such as uploading new lists or batch update lists. The main website operation follows the three-tier architecture model.

4.1.1 Administrative Operation

Data maintenance is the main objective of administrative operation. Administrators need to manually download the lists released by schools from their respective websites for the system to process. The lists are usually released in the Portable Document Format (PDF), while some of them are just plain webpages (HTML). The raw lists are then processed by parsers with different patterns that are specifically written for a type of lists (which is usually lists released by one school). The processed lists are stored in a comma-separated values (CSV) file that will be feed to the object generator in the next step. The parsing processes of different lists are grouped into a batch update module that can be called to mass update the lists. Object generator takes the formatted CSV file, generates data objects for the list records and finally stores them into the datastore. The entire administrative operation is illustrated in Figure 4.1.

4.1.2 Three-Tier Architecture of MyNext

Three-tier architecture is a client-server architecture in which the user interface, process logic, data storage and data access are developed as independent modules (Eckerson, 1995). The three tiers are (i) presentation tier, (ii) logic tier and (iii) data

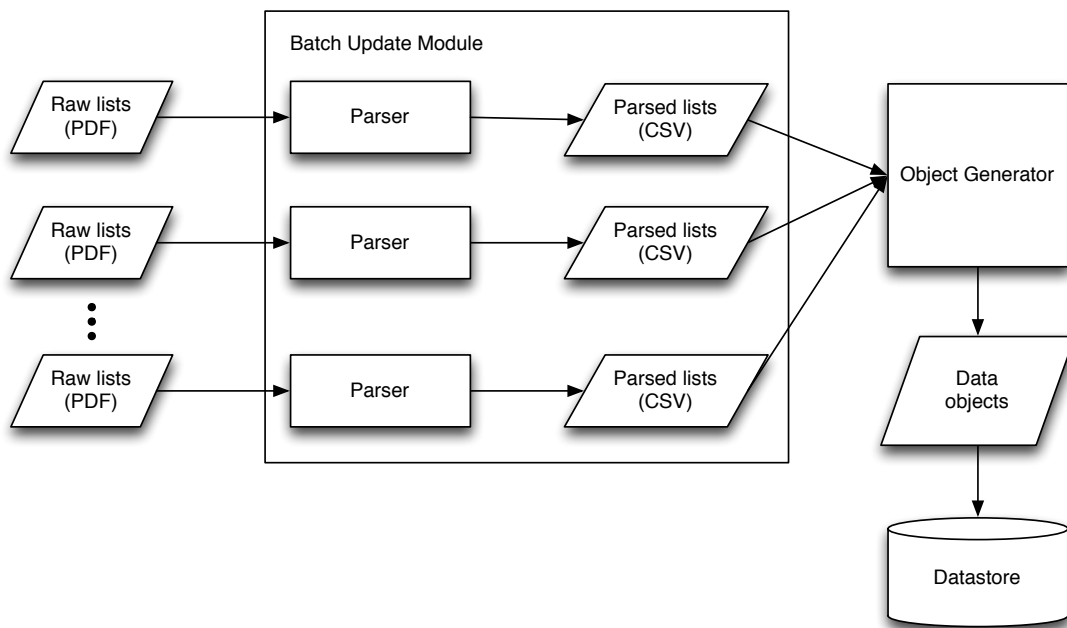


Figure 4.1: Flow Chart of the Administrative Operation.

tier. The website user interface is considered as the presentation tier, application programs developed to process the requests belong to the logic tier, and finally the data models and the backend datastore are regarded as the data tier. Figure 4.2 shows the three-tier architecture of MyNext.

The user interface of MyNext is in the presentation tier which is the closest to the users. It is designed with responsiveness in mind to let users have the best experience. By utilizing the latest Web technologies, the user interface is both simple and intuitive. The user interface will be explained in great detail in Chapter 5.

Logic tier contains all the application programs of MyNext. The role of these application programs is to process the requests from the user and also retrieve data from the data tier. The application programs are further divided into three sections: list query section, interaction section and social network integration section. The list query section contains the core programs that can be use to retrieve list records from the datastore based on different criteria specified by users. The interaction section consists of programs that enable user to vote, leave comments, and do list corrections. Programs in the social network integration section are in charge of connecting to Facebook and retrieving details of users.

Finally, the data tier is where all the records are stored. MyNext uses data models as the abstraction to store data in an object-oriented way. Therefore, the underlying database engine only serves as the datastore of the data models and objects. Data models also provide interfaces and methods for the logic tier to retrieve and update

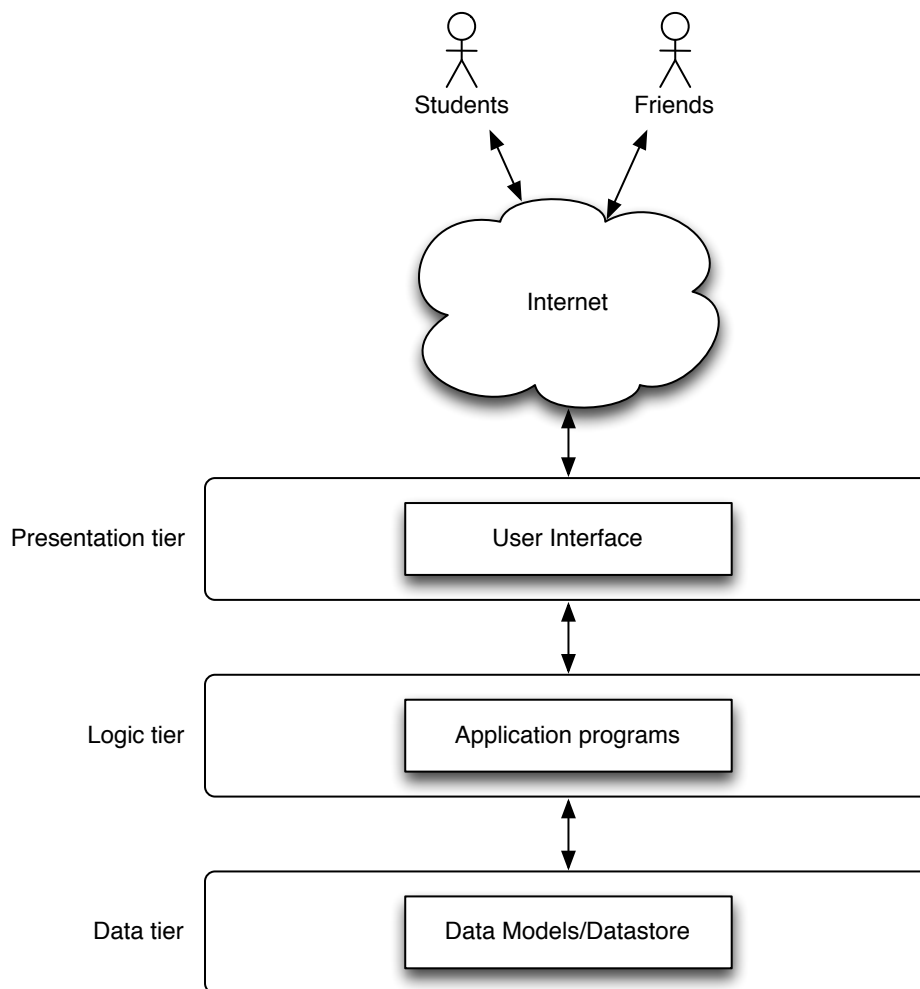


Figure 4.2: The Three-Tier Architecture of MyNext.

records. The details of the implementation of each program are provided later in the chapter.

4.2 Infrastructure

In this section, each component of the MyNext will be described. The infrastructure of MyNext is summarized in Table 4.1.

4.2.1 Hardware

MyNext is powered by a virtual private server (VPS). Using a VPS instead of a traditional dedicated machine is more cost efficient and more flexible. A VPS is a virtual machine running on a more powerful server. It has its own resources and

Table 4.1: System Infrastructure of MyNext.

Component	Technology/Implementation
Hardware	Virtual Private Server
Operating System	Linux
Web Server	Apache
Database Server	PostgreSQL
Web Application Framework	Django
Client-side Scripting	JavaScript & AJAX

hard disk space just like a real machine. Since service providers can run multiple VPSs on a single physical server, the cost of VPSs is only a fraction of the cost of a dedicated machine. The configuration of VPSs can also be upgraded or downgraded at any time with minimum system downtime. This enables MyNext to scale up or down if needed with almost no effort.

4.2.2 Operating System

For the best stability and compatibility, MyNext has chosen Linux as its underlying operating system. Linux is a proven server platform for Web applications. As of August 2011, over 60% of the Web servers were running Linux (Q-Success, n.d.). As an interesting fact, over 90% of the supercomputers in the world are running customized Linux distributions, too. This means that whenever stability is required, Linux is the number one choice.

4.2.3 Web Server

Just like Linux, Apache is often coupled with Linux as the de facto standard of Web servers. Over 65% of the websites are served by Apache (Netcraft Ltd, n.d.). Since it is so widely used, there is a wide variety of extensions to choose from to further extend the functionality of Apache. MyNext uses `mod_wsgi` to run all of its application programs and other server-side code written in Python.

4.2.4 Database Server

PostgreSQL is an open source relational database management system (RDBMS). It is served as the underlying database engine for MyNext data models. One very important feature that makes MyNext chose PostgreSQL over other RDBMSs is the support of UTF-8 4-byte characters. Because there are just way to many characters exist in Chinese, MyNext needs to store some of the extremely rare characters

found in some names that are UTF-8 4-byte encoded. Out of all stable versions of the well-known RDBMSs, only PostgreSQL fully supports UTF-8 4-byte characters. It is also the recommended RDBMS by Django, the Web application framework that MyNext is built on.

4.2.5 Web Application Framework

Due to the need of rapid development, Web application frameworks are the trend of Web development in the past few years. Django is a Web application framework for perfectionists with deadlines (Kaplan-Moss & Holovaty, 2007). It has covered everything that is difficult and tedious such as maintaining database connections and parsing HTTP requests so developers can focus on writing the application itself. The applications built on Django follow a Model-Template-View (MTV) paradigm which makes them very easy to maintain. Some of the well-known applications built on Django are DISQUS, Bitbucket and Instagram. MyNext uses Django to built all of its application programs and some parts of the admin console.

4.2.6 Client-Side Scripting

JavaScript and Asynchronous JavaScript and XML (AJAX) are really what make the entire Web application business possible (Paulson, 2005). Instead of serving content page-by-page, JavaScript and AJAX enabled applications behave just like desktop applications where updates only take place to the part of the page that needed them. Because the communication is asynchronous, users do not have to wait for server's reply before doing other actions. This generally makes Web applications more responsive and user friendly. MyNext uses JavaScript and AJAX throughout to improve the responsiveness of the user interface.

4.2.7 The MyNext Stack

A technology stack refers to the different layers of components that are used to provide a system. Figure 4.3 illustrates the entire MyNext stack. At the bottom is the operating system, Linux. Running on top of Linux are the web server and the database server, Apache and PostgreSQL respectively. Python is running on top of Apache using mod_wsgi extension. Django is running on top of mod_wsgi and facilitates PostgreSQL as its underlying datastore. Finally, all the application programs and the admin console are built on top of Django.

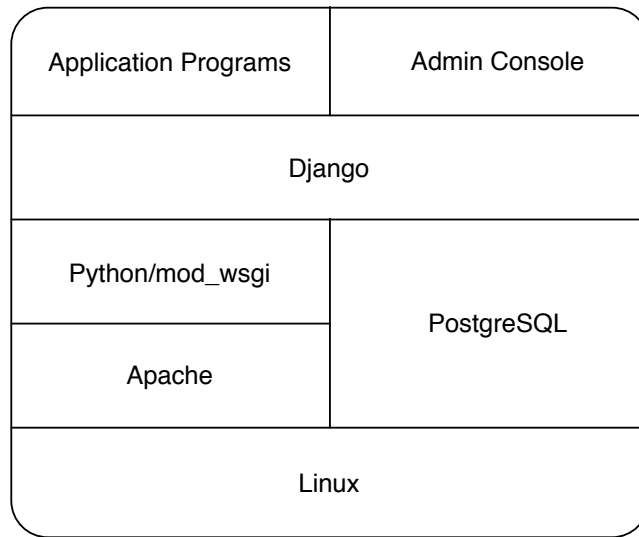


Figure 4.3: The MyNext Stack.

4.3 Data Models

Data models are the abstraction layer on top of the datastore. Data is stored as an object. There are several data models in MyNext. Data models also provide interface and methods for the application programs to interact with data. Figure 4.4 gives an overview of all data models in the form of a class diagram.

4.3.1 User model

This model stores the users of MyNext. Users are those who have logged in with their Facebook account. Therefore an User object also has a FBAccount object. User model is also in charge of keeping track of user comments, votes, corrections, and common name filters. The base User model is provided by the Django `django.contrib.auth` package.

4.3.2 FBAccount model

This model is one-to-one related to the User model because one User object can only link to one FBAccount. This model has interfaces to Facebook that can retrieve additional information such as friends list. It also handles the authentication of Facebook account and logs the user in using the local User object.

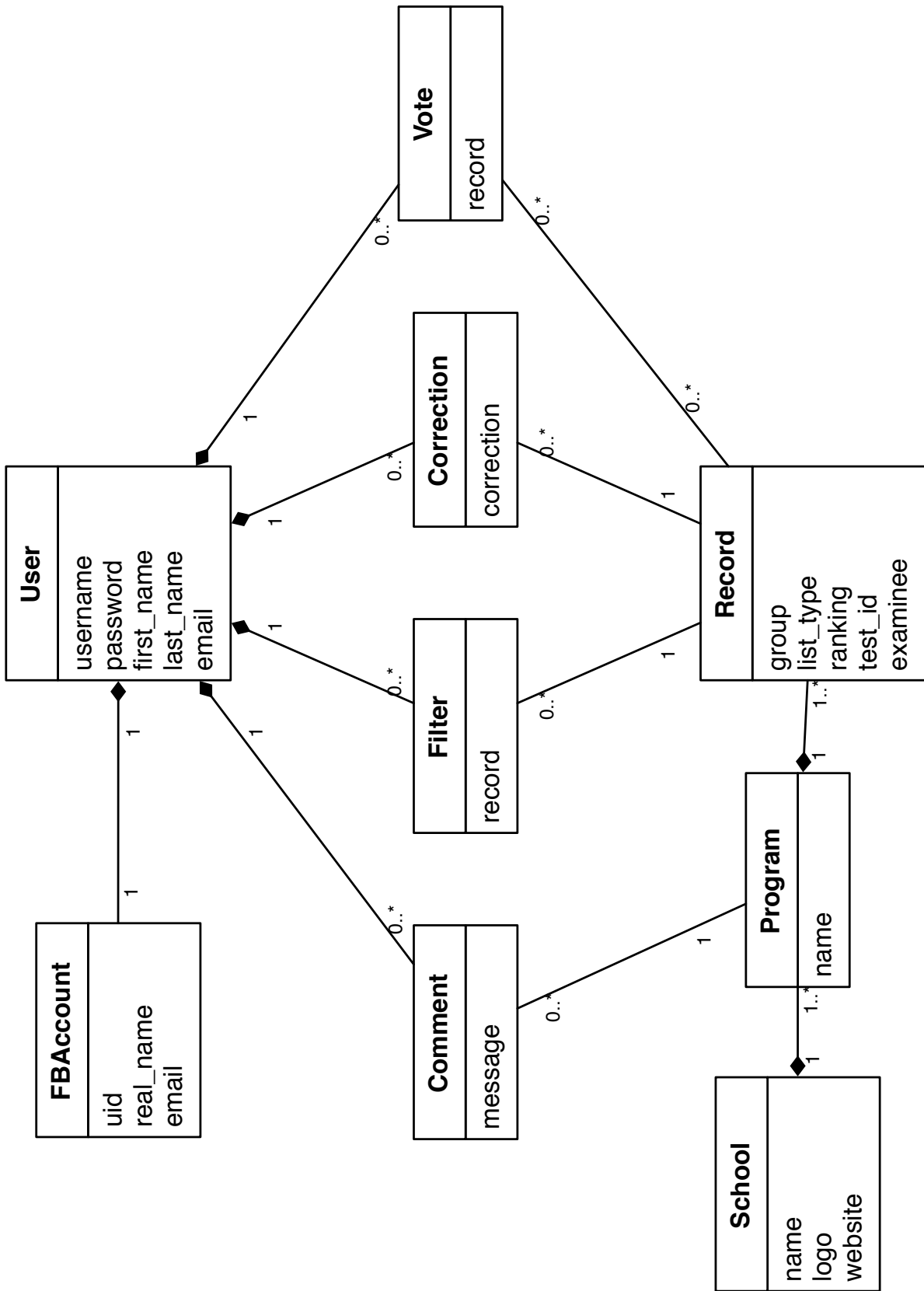


Figure 4.4: Class Diagram of MyNext.

4.3.3 School model

This model stores the basic information of a graduate school. It has interfaces to retrieve list of programs of the school. Basically a list of School objects is displayed on MyNext's home page for browsing purposes.

4.3.4 Program model

This model stores the information about a program. A Program object is owned by a School object. It has interfaces to retrieve the lists of the program, and also the list of comments left by the users about the program.

4.3.5 Record model

This model stores all the records of the exam results. The Record objects are owned by the Program object that they belong to. This is the core model with all the important interfaces such as querying for list of records, searching through records, retrieving corrections and common name filters, and aggregating the votes of each record.

4.3.6 Vote model

This model stores the votes for possible outcome predicted by the users. The Vote objects are owned by the User object who casted them and are many-to-many related to Record objects. It has interfaces to casting votes and preventing duplicate votes for a student. It also can retrieve a list of votes by all users or just by user's friends.

4.3.7 Filter model

This model stores the common name filters done by the users on multiple offer records. Filter objects are owned by the User object who created them and are many-to-one related to a Record object. It has similar interfaces to the Vote model because Filter objects behave very much like Vote objects.

4.3.8 Correction model

This model stores the correction made by users on a record. Correction objects are owned by the User object who made them and are many-to-one related

to a Record object. The model has interfaces for the admin console to review the corrections and apply them if the corrections are approved by administrators.

4.3.9 Comment model

This model stores the comments made by users on a program. Comment objects are owned by the User object who left them and are many-to-one related to a Program object. It has interfaces for users who left the comment to remove the comment. Also has interfaces to enable comment moderation in the admin console.

4.3.10 Underlying Datastore

Although the data models are object-oriented, the underlying storage is relational. Therefore an object-relational mapping (ORM) is needed to map the data objects to database records. Fortunately, Django provides a powerful ORM that perfectly abstracts the datastore away. Only the ORM was used to manipulate with all the data in the datastore, this makes MyNext portable to any underlying database engine if a switch of underlying database is ever needed. MyNext chooses the database server recommended by Django as its underlying datastore, which is PostgreSQL.

4.4 Program Design

MyNext is coded entirely in Python, which is a dynamic programming language developed by Guido van Rossum. MyNext chooses Python because of its clean syntax, it makes use of indentations and code blocks, and most importantly the ability to build on top of Django.

In this section, we describe the implementation detail of application programs and other modules of MyNext. The summary of the programs in both admin console and application programs subsystems is presented in Table 4.2.

All the pseudocode snippets are written using Python syntax. Functions used in the pseudocode snippets might not exist in real Python. Most of them are just made up to illustrate the logic.

4.4.1 Parsers

Parsers are in charge of converting and parsing the raw lists. Because the raw lists are usually in PDF format, text contents are first extracted out from the PDF. Then the text is parsed using a few regular expressions with the patterns written

Table 4.2: List of the Programs.

Admin Console	Application Programs
Parsers	List browsing module
Object generator	Multiple offer getter module
Batch update module	Search module
	Watch list module
	Prediction voting module
	Comment module
	Common name filtering module
	Intelligent prediction module
	Facebook authentication module

specifically for the list format. Finally the parsed data are formatted in comma-separated values and save to an external file. Listing 4.1 shows the pseudocode of parsers.

```

input_filename = 'nchu.pdf'
output_filename = filename_without_ext(input_filename) + '.csv'
input_pdf = open(input_filename)
input_txt = pdf.extract_text(input_pdf)
patterns = [
    # Patterns with the format of the specific list
    ...
]

for pattern in patterns:
    matches = re.findall(pattern, input_txt)

    if matches:
        for match in matches:
            csv.write(output_filename, match)

```

Listing 4.1: Pseudocode of Parsers.

4.4.2 Object Generator

Object generator focuses on generating data objects from CSV files. It loops through CSV files in a location and generate data objects out of them. Listing 4.2 shows the pseudocode of object generator.

4.4.3 Batch Update Module

The batch update module is based on the parsers and the object generator. This module can be used to batch update the lists. The module will loop through all the

```
for csv_file in csv_directory:
    reader = csv.read(csv_file)
    for row in reader:
        record = generate_object(row)
        record.save()
```

Listing 4.2: Pseudocode of Object Generator.

PDF files, run parser for each PDF file, and finally run object generator to generate data objects from the parsed CSV files. Listing 4.3 shows the pseudocode of batch update module.

```
for parser in parsers:
    return_code = run_parser(parser)
    if return_code:
        # Error occurred, log the error list and move on to the next list
        ...

run_object_generator()
```

Listing 4.3: Pseudocode of Batch Update Module.

4.4.4 List Browsing Module

This module belongs to the list query section of the application programs. It will show a list of programs if a school is received from the user, or show the lists of the program if a program is received from the user. When there is nothing received from the user, a list of graduate schools is displayed. Listing 4.4 shows the pseudocode of this module.

```
query = request.POST.get('query')
if query is not None:
    if isinstance(query, School):
        return Program.objects.filter(school=query)
    elif isinstance(query, Program):
        return Record.objects.filter(program=query)
else:
    return School.objects.all()
```

Listing 4.4: Pseudocode of List Browsing Module.

4.4.5 Multiple Offer Getter Module

This module belongs to the list query section of the application programs. After a list of records for a program is obtained, multiple offers are checked one-by-one

for each record on the list using the multiple offer getter module. The module uses the interface provided by the Record model to get a list of multiple offers by that record. If multiple records are found, they are cached to a non-persistent property of the record. Listing 4.5 shows the pseudocode of multiple offer getter module.

```
for record in records:
    # Interface provided by Record model
    offers = record.get_multiple_offers()

    record._multiple_offers = offers
```

Listing 4.5: Pseudocode of Multiple Offer Getter Module.

4.4.6 Search Module

This module belongs to the list query section of the application programs. This module takes the name or test ID entered by the user and queries the Record model using the interface it provides. It returns a list of results if any. Listing 4.6 shows the pseudocode of search module.

```
query = request.POST.get('query')
if query is not None:
    results = list(Record.objects.filter(examinee__name=query))
    results.append(Record.objects.filter(test_id=query))
    return results
else:
    return None
```

Listing 4.6: Pseudocode of Search Module.

4.4.7 Watchlist Module

This module belongs to the interaction section of the application programs. This module can create, delete, and return a list of items in an user's watchlist. Listing 4.7 shows the pseudocode of watchlist module.

4.4.8 Prediction Voting Module

This module belongs to the interaction section of the application programs. This module is used for users to cast votes of their prediction for students with multiple offers. The module uses interfaces provided by the Vote model to create and delete votes. Before casting a vote for a student, the module needs to check for duplicate

```

def create(program):
    w = WatchItem(user=request.user, program=program)
    w.save()

def remove(program):
    w = WatchItem.objects.get(user=request.user, program=program)
    if w:
        w.delete()

```

Listing 4.7: Pseudocode of Watchlist Module.

votes of the user on that student. Listing 4.8 shows the pseudocode of prediction voting module.

```

def cast_vote(name, record):
    # Check for existing vote first
    v = Vote.objects.get(user=request.user, name=name)
    if v:
        v.delete()

    v = Vote(user=request.user, name=name, record=record)
    v.save()

def remove_vote(name, record):
    v = Vote.objects.get(user=request.user, name=name, record=record)
    if v:
        v.delete()

```

Listing 4.8: Pseudocode of Prediction Voting Module.

4.4.9 Comment Module

This module belongs to the interaction section of the application programs. This module is in charge of user comments. It allows users to add comments and delete their own comments. It also enables administrators to delete comments. Listing 4.9 shows the pseudocode of comment module.

4.4.10 Common Name Filtering Module

This module belongs to the interaction section of the application programs. This module provides several functions such as adding and removing the common name filter for records from a source record and demoting the position of the filtered records when displaying multiple offers. Filtered records are moved to the bottom of the list and are slightly faded out. Listing 4.10 shows the pseudocode of common name filtering module.

```

def add_comment(message):
    c = Comment(user=request.user, message=message)
    c.save()

def delete_comment(id):
    # Check if the comment belongs to the user
    c = Comment.objects.get(user=request.user, pk=id)
    if c:
        c.delete()

def admin_delete_comment(id):
    # Check if the user is an administrator
    if request.user in administrators:
        c = Comment.objects.get(pk=id)
        c.delete()

```

Listing 4.9: Pseudocode of Comment Module.

```

def add_filter(record, from_record):
    # get_or_create is a shortcut method that will only create
    # the object if it isn't already existed
    f, created = get_or_create(Filter, user=request.user,
                               record=record, from_record=from_record)

def remove_filter(record, from_record):
    f = Filter.objects.get(user=request.user,
                           record=record, from_record=from_record)
    if f:
        f.delete()

def demote_offers(records):
    # Get filters of the user that are filtered from these records
    filters = Filter.objects.get(user=request.user,
                                  from_record__in=records).values('record')

    for record in records:
        for offer in record._multiple_offers:
            # Demote the record to the last of the multiple offer list
            if offer in filters:
                move_to_bottom(offer)

```

Listing 4.10: Pseudocode of Common Name Filtering Module.

4.4.11 Intelligent Prediction Module

This module belongs to the interaction section of the application programs. This module makes predictions based on the collective intelligence of MyNext users and their friends. It will determine which program each student with multiple offers would go based on users' votes. By default, only votes from friends of the user are counted, but an user can also choose to use all votes. The module will calculate the position on the waiting-list for the requested student. If the position is 0, then it means that the student will get the offer for sure. Listing 4.11 shows the pseudocode of intelligent prediction module.

4.4.12 Facebook Authentication Module

This module belongs to the social network integration section of the application programs. This module provides necessary functions to talk to the Facebook servers and authenticate users using Facebook API. It will be coupled with the JavaScript API on the client-side to log in to Facebook and record down the access token for users. Then the system can use this access token to access to the information on users Facebook later on such as getting users' friends lists. Listing 4.12 shows the pseudocode of Facebook authentication module.

```

def predict(record, friends_only=True, from_record=None):
    # Student on short-list always return 0
    if record.list_type == 1:
        return 0

    # Current position of the requested student
    position = record.ranking

    # Get all the records in front of the requested student
    # Part 1: all the students on short-list
    # Part 2: all the students on waiting-list with ranking less than
    # the requested student
    records_infront = list(Record.objects.filter(
        program=record.program, list_type=1))
    records_infront.append(list(Record.objects.filter(
        program=record.program, list_type=2, ranking_lt=position)))

    # Cache the multiple offers
    get_multiple_offers(records_infront)

    for r in records_infront:
        # Total offers: multiple offers + current offer
        total_offers = r._multiple_offers.append(r)
        # Remove the from record to prevent infinite loops when
        # calling recursively
        if from_record is not None:
            total_offers.remove(from_record)
        # Get all votes for all offers
        votes = Vote.objects.filter(record_in=total_offers)

        # Filter votes to only casted by friends
        if friends_only:
            votes = votes.filter(
                user_in=request.user.fbaccount.get_friends())

        # Count votes for each record and get the highest voted record
        highest_voted_record = votes.Count('record').Max()

        # Call predict recursively to predict the highest voted record
        # Bump up position if highest voted record is admitted for sure
        # in another program
        if highest_voted_record != r:
            highest_pos = predict(highest_voted_record,
                friends_only, record)
            if highest_pos == 0:
                position = position - 1

                # If requested position after bumped up is 0, then
                # the requested student is admitted for sure.
                if position == 0:
                    break

    # Return the calculated position for the requested student
    return position

```

Listing 4.11: Pseudocode of Intelligent Prediction Module.

```

def fb_auth():
    # Use the JavaScript API to show login window
    launch_js_api_login()

    # Wait for client to finish login, client will send
    # auth code after finished
    auth = wait_for_client_auth()

    # Login failed
    if not auth:
        return False

    # Get account details using Facebook API
    profile = fb_api.get_profile(auth)

    # Check if a local user existed for the account
    fbaccount = FBAccount.objects.get(auth=auth)

    # If the account has already been linked with a local user,
    # log the user into the system
    if fbaccount:
        fbaccount.user.login()
        return True

    # Otherwise create a new local user and link the account
    # to the user
    else:
        user = User(
            username=profile.username,
            first_name=profile.first_name,
            last_name=profile.last_name,
            email=profile.email)
        user.save()
        fbaccount = FBAccount(user=user, auth=auth)
        fbaccount.save()
        return True

```

Listing 4.12: Pseudocode of Facebook Authentication Module.

Chapter 5

User Interface

The user interface of MyNext is designed with responsiveness and user friendliness in mind. By utilizing JavaScript and AJAX, MyNext is striving to bring students best browsing experience. In this chapter, all components of the user interface will be discussed in detail accompanied with screenshots.

5.1 Homepage

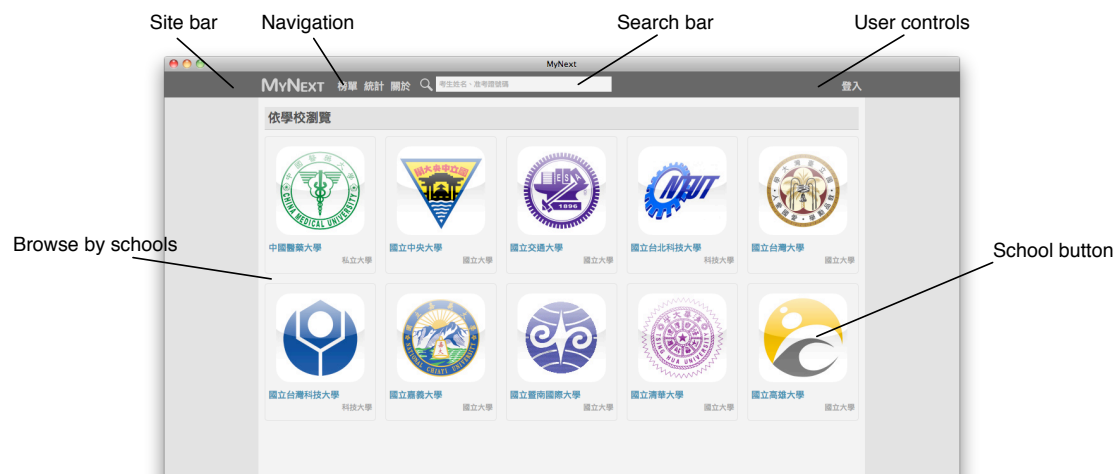


Figure 5.1: Screenshot of the Homepage.

Figure 5.1 shows the key components of MyNext homepage. First of all, all the pages in MyNext consist of an universal site bar at the top. This bar allows students to quickly go back to the homepage, access the navigation menu, using the search bar, and login or logout of MyNext. The rest of the homepage is a list of all the graduate schools displayed in a grid format along with their logos. Nothing else is on the page to bring further confusion or messiness. It is just a clean, straight

forward school list for students to quickly find what they want. There is however one thing that is missing from the screen which is the watchlist because in the screen we haven't logged in yet with a Facebook account. Once a student has logged in, he/she shall see the screen like Figure 5.2. The watchlist is basically a customized list of items that the user is interested in. It is similar to the "bookmarks" of the browser.



Figure 5.2: Screenshot of the Watchlist.

5.2 Login Page



Figure 5.3: Screenshot of the Login Page.

Before a student can use all the features of MyNext, he/she would need to login using his/her Facebook account. Currently MyNext does not manage its own membership system, so it is entirely tied to the Facebook for authentication. It has the advantage of letting the students to quickly get started on all the features on MyNext. The initial login page is shown in Figure 5.3. After a student clicks the Facebook login button, a pop up window like in Figure 5.4 will appear asking him/her to login to Facebook. Finally, a permission request page will come up asking for permissions that are needed for MyNext to function properly. After a student

has accepted to give permissions to MyNext, they will not be asked again the next time they login.

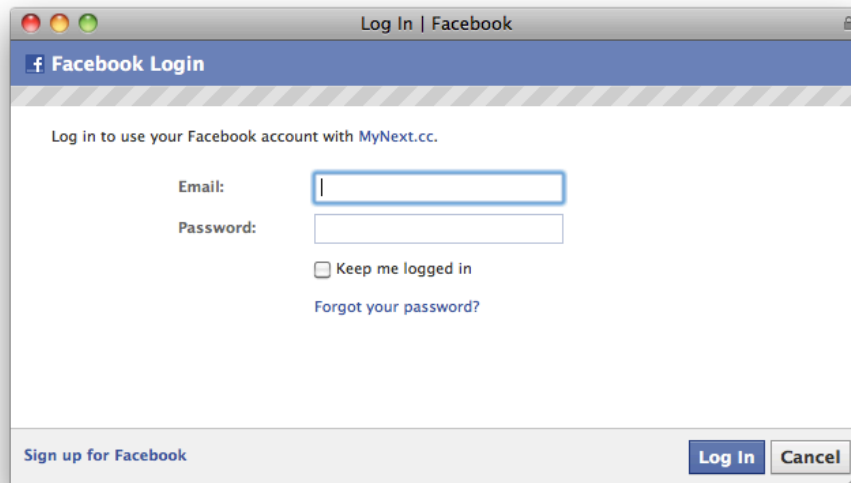


Figure 5.4: Screenshot of the Facebook Login Page.

5.3 School Page

After a student has clicked on a school from the homepage, he/she will then be linked to the school page of the school he/she clicked. As in Figure 5.5, the school page has several components. At the top of the school page, the information about the school is displayed such as its name, English name, website, results release date, total number of students on lists, etc. At the bottom of the school information is a list of all programs offered by the school. There is a little statistics bar at the bottom of each program that shows the summary of the results of that program. At the bottom right corner there is a button for students to add the program to their watchlists. Because the interface is AJAX enabled, all the little widgets and buttons fire immediately after they are clicked, and the changes are updated instantly. The school page provides a quick glance of the status of the school and its programs.

5.4 Program Page

The program page is probably where most of the cross-checking and interactions take place. The program page is divided to two sides. The left side are the short-lists and waiting-lists for the program, and the right side is the sidebar which provides

MyNEXT 考生姓名、准考證號碼 Denny Tsai 登出

國立中興大學
National Chung Hsing University

類型 國立大學
網站 www.nchu.edu.tw
放榜日期 無口試 03/16 初試 03/16 複試 03/30
放榜系所 67
上榜數合計 4211

系所列表

資訊管理學系 ■2 ■35 ■109 ■100 ◁162	食品暨應用生物科技學系 ■4 ■50 ■93 ■92 ◁76	機械工程學系 ■6 ■72 ■171 ■180 ◁279
土木工程學系 ■6 ■86 ■84 ■106 ◁206	化學工程學系 ■3 ■49 ■85 ■113 ◁179	材料科學與工程學系 ■3 ■63 ■97 ■104 ◁223
數學教學中等教師進修班 ■1 ■24 ■2 ■12 ◁0	中國文學系 ■2 ■35 ■23 ■41 ◁39	歷史學系 ■2 ■26 ■5 ■16 ◁21
圖書資訊學研究所 ■2 ■8 ■4 ■5 ◁7	財務金融學系 ■1 ■21 ■103 ■107 ◁172	國家政策與公共事務研究所 ■2 ■38 ■41 ■31 ◁19
國際政治研究所 ■2 ■44 ■28 ■35 ◁52	企業管理學系 ■3 ■20 ■67 ■73 ◁117	會計學系 ■1 ■16 ■52 ■68 ◁73
行銷學系 ■1 ■15 ■40 ■48 ◁28	運動與健康管理研究所 ■2 ■10 ■13 ■15 ◁29	農藝學系 ■5 ■20 ■9 ■13 ◁8

Figure 5.5: Screenshot of the School Page.

several functions such as quickly jump through different groups in the program, changing display options, toggling votes display, and a comment section for the program.

On each record, there is a “multiple offers” list with all other records with the same name of the student. There are several actions students can do in the “collaborative cross-checking” column. First is voting, once a student has logged in, he/she can cast a vote on whether he/she thinks the student would most likely get the offer from which one of the multiple offers. Second is the common name filtering, students can mark one or more records from the multiple offers list if they think those records do not refer to the same student. The system will then move those records to the bottom of the multiple offers list and show them in a fade-out effect. Third is the list correction button, a pop up dialog will appear to let students make correction suggestions to a record. Finally, the badge indicating the list type and the ranking is actually a button itself, and a very important one, too. This button will initiate the system to do an intelligent prediction of that student’s calculated position on the list. This is especially useful for those students on the waiting-list. The system can check whether they can be bumped up or even get admitted based on the votes collected by the system. Although this is not an accurate prediction, it can at least provide some pointers as to how much chance the student can get

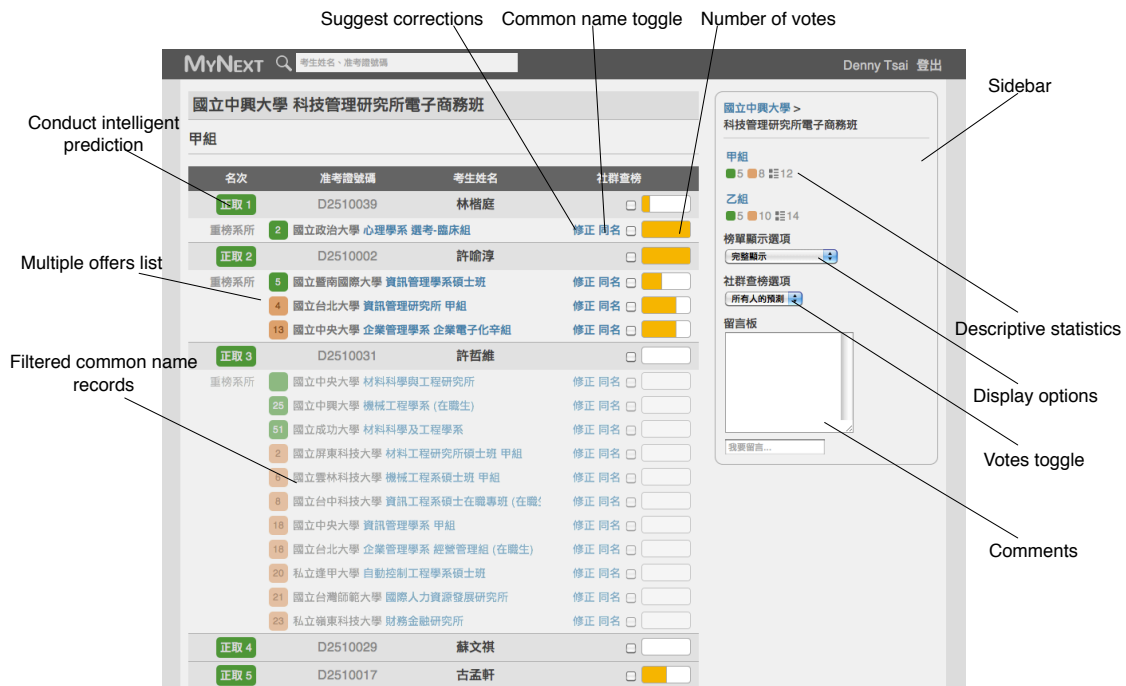


Figure 5.6: Screenshot of the Program Page.

admitted to the program.

On the sidebar, there are some assistant features as well. Since the sidebar stays statically on the page even if the page is scrolled, it can be seen as the dashboard of the program. First, it has a descriptive statistics section about the program. It shows the number of students on the short-list, waiting-list, and who got multiple offers in each group of the program if there are any. The display options let students to toggle between several different views. This can help to reduce the number of records displayed on the list so only the records that the students are interested in are shown. Next is the toggle for votes. Because sometimes students only want to know about their friends suggestions or sometimes they just can't trust other people to provide useful information, the toggle can let students to toggle the votes to include all votes or only the votes by their friends. This toggle will also affect the intelligent prediction results as well. At the bottom of the sidebar, there is a comment section. This place is for students to communicate and share their thoughts and insights about the program. Students can leave some insights about the fill-up process of the program or some concrete information they got. Generally, it is just an additional communication tool to aid the students when they are cross-checking.

MyNEXT 考生姓名、准考證號碼			
搜尋結果：宗翰			
1	101M27D005	何宗翰	私立弘光科技大學 職業安全與防災研究所碩士班
2	35000043	何宗翰	私立淡江大學 保險學系保險經營碩士班
5	605005	何宗翰	私立逢甲大學 機械與電腦輔助工程學系機械工程碩士班 熱流組
6	M164000001	何宗翰	私立世新大學 經濟學系
2	61731012	余宗翰	國立台灣師範大學 機電科技學系 精密機械組
3	43230005	余宗翰	私立中原大學 機械工程學系 乙組
3	1229014	余宗翰	私立元智大學 機械工程學系碩士班 乙組
54	438340017	余宗翰	國立中山大學 機械與機電工程學系碩士班 丁組
1	26200004	劉宗翰	私立淡江大學 化學工程與材料工程學系碩士班 B 組
3	1304015	劉宗翰	私立元智大學 化學工程與材料科學學系碩士在職專班 (在職生)
3	1304015	劉宗翰	私立元智大學 化學工程與材料科學學系碩士在職專班 (在職生)
17	141010008	劉宗翰	國立東華大學 國際企業學系碩士班
1	16032009	劉宗翰	私立大同大學 化學工程研究所 乙組
3	14113-1005	劉宗翰	國立雲林科技大學 環境與安全衛生工程學系碩士班 環安組(英文+微積分(1)+化學)
6	土K008	劉宗翰	國立高雄應用科技大學 土木工程與防災科技所碩士班 甲組

Figure 5.7: Screenshot of the Search Results Page.

5.5 Search Results Page

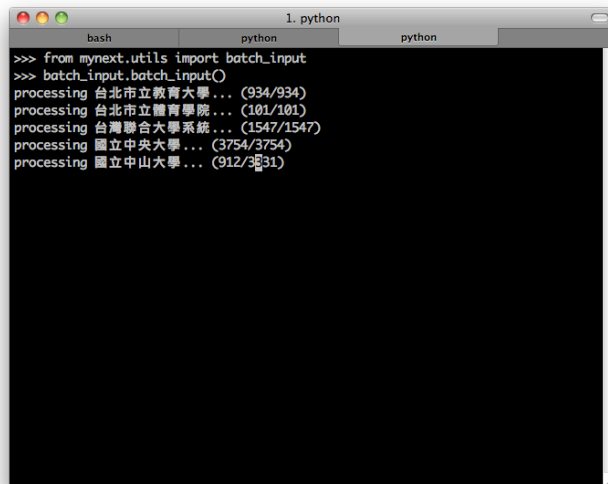
The search results page is pretty straight forward. This page displays the results of a search like in Figure 5.7. The search can be conducted on the search bar located at the site bar at the top of every page. It is stuck to the top of the window so the search function is always available.

5.6 Admin Console

The admin console is built for system administrators to perform administrative operation. The console is based on the Python interactive shell. It has several commands available the can batch update the lists, make corrections to the list, moderate the comments, etc. The admin console is a command line based environment. A graphical user interface was not built because the admin console isn't designed for public use.

5.7 Usage Examples

In this section, we provide two usage examples on how students can use MyNext to improve their cross-checking experience. The first example is just a search for



```
>>> from mynext.utils import batch_input
>>> batch_input.batch_input()
processing 台北市立教育大學... (934/934)
processing 台北市立體育學院... (101/101)
processing 台灣聯合大學系統... (1547/1547)
processing 國立中央大學... (3754/3754)
processing 國立中山大學... (912/3831)
```

Figure 5.8: Screenshot of the Admin Console.

the name of a student. The second example is a more sophisticated application of intelligent prediction.

5.7.1 Searching for a Student

Student A wants to know how well did his friend, Student B, did on the entrance exams. After the results were released, Student A log on to MyNext. He then typed in Student B's name in the search bar. The search results page is returned with records of with Student B's name. The example is illustrated in Figure 5.9.



Figure 5.9: Illustration of Searching for a Student.

5.7.2 Using the Intelligent Prediction

The above example is just a simple search. Now consider the following situation: student C is on the waiting-list for a program, she does have any other offers. Therefore she is very anxious about whether or not she can get the offer. She is on the number four spot on the waiting-list. The three students before her all have multiple offers, but she still wants to at least have some indication about her chances of getting admitted. She logs on to MyNext, and asks her friends also to check out the list for her. Her friends make some suggestions for her by voting. She then uses the intelligent prediction function and finds out that her chance of getting admitted is very likely. Additionally, there are some comments from the comment section that say the first two students on the waiting-list have already turned down the offer according to the department office. Thus, she gains more confidence on her exam results. Figure 5.10 shows using the intelligent prediction and the comments section.

The diagram illustrates the process of using intelligent prediction in a ranking system. It shows a transition from a list of candidates with their current ranks to a predicted list where the top candidates' ranks are updated. A chat window at the bottom shows a discussion about the prediction results.

Rank	ID	Name
備取 1	423030083	涂謹滢
重榜系所	8	國立交通大學 資訊管理研究所 甲組
備取 2	423030015	黃學惇
重榜系所	3	國立交通大學 資訊管理研究所 甲組
	13	國立清華大學 資訊系統與應用研究所
備取 3	423030096	彭煥淇
重榜系所	2	國立台灣師範大學 資訊教育研究所 資訊科技教育組
	3	國立交通大學 生物資訊及系統生物研究所 乙組
	3	國立交通大學 資訊管理研究所 乙組
	3	國立政治大學 資訊管理學系 商管組
	14	國立中興大學 資訊管理學系
	1	國立成功大學 資訊管理研究所 乙組
	3	國立清華大學 服務科學研究所 乙組(服務系統組)
備取 4	423030010	謝志優

Rank	ID	Name
預測 0	423030083	涂謹滢
重榜系所	8	國立交通大學 資訊管理研究所 甲組
預測 0	423030015	黃學惇
重榜系所	3	國立交通大學 資訊管理研究所 甲組
	13	國立清華大學 資訊系統與應用研究所
備取 3	423030096	彭煥淇
重榜系所	2	國立台灣師範大學 資訊教育研究所 資訊科技教育組
	3	國立交通大學 生物資訊及系統生物研究所 乙組
	3	國立交通大學 資訊管理研究所 乙組
	3	國立政治大學 資訊管理學系 商管組
	14	國立中興大學 資訊管理學系
	1	國立成功大學 資訊管理研究所 乙組
	3	國立清華大學 服務科學研究所 乙組(服務系統組)
預測 0	423030010	謝志優

留言板

Mark王:
聽說丙組前兩個已經確定不去囉...

Roy Chen:
剛剛去系辦看了 現在確定備到3了

謝謝各位大大!

Figure 5.10: Illustration of Using the Intelligent Prediction.

Chapter 6

Conclusions

In this thesis, we have built a new CCS called MyNext. It has several breakthrough advancements such as integrating with SNS, Facebook and applying the concept of collective intelligence. In the traditional cross-checking aspect, MyNext is an improvement over existing CCSs by providing a better and more user friendly user interface. By grasping the concept of collective intelligence, MyNext lets students and their friends to collaborate on different aspects of cross-checking and makes the process more interesting. MyNext turns a tedious work that was usually done alone by students themselves to a group activity. Cross-checking using MyNext is no longer just blindly browsing through lists and trying to figure out the possible outcome with no assistance. Using MyNext, students can help out each other by voting for their predictions and leaving comments. Students' friends can also help out by making recommendations that they think are the best for the students. Common name filtering can make the list simpler to interpret. And most importantly, one of the best features of MyNext is the ability to intelligently predict the outcome for a student based on the votes and recommendations made by all the users of MyNext. This is collective intelligence in full swing.

Feature Comparison with Existing CCSs

Table 6.1 shows the feature comparison of MyNext, Daso, and iCross. Only basic features are provided by Daso and iCross while MyNext also has the advanced features. One thing to note is that only MyNext does not have any advertisement banners while the other two have massive advertisements that can greatly hinder the browsing experience.

Table 6.1: Feature Comparison Table of MyNext, Daso and iCross.

Features	CCS		
	MyNext	Daso	iCross
Basic			
Basic cross-checking	✓	✓	✓
Searching	✓	✓	✓
Interaction			
Watchlist	✓	✗	✗
Outcome prediction voting	✓	✗	✗
Common name filtering	✓	✗	✗
Comments	✓	✗	✗
Corrections	✓	✗	✗
Intelligent prediction	✓	✗	✗
SNS integration			
Login with Facebook	✓	✗	✗
Friends based interactions	✓	✗	✗
Others			
Provide descriptive statistics	✓	✗	✗
Customized display options	✓	✗	✗
Other personalizations	✓	✗	✗
Banner advertisements	✗	✓	✓

Breakthroughs and Contributions

MyNext is an advanced CCS that not only let students do cross-checking, but it most importantly focuses on waiting-listed students. Waiting-listed students are the ones that needed cross-checking the most because of the uncertainty of their outcome. MyNext focuses to become a true assistant for waiting-listed students by providing useful information provided from students' friends and other users. Waiting-listed students can figure out their possible directions more easily after using MyNext.

Potential Future Improvements

Due to the nature of collective intelligence, garbage data exists as well as valuable information. One of the potential improvements could be implementing a trust system. A lot of Web applications with collective intelligence have some sort of self-governing mechanisms. For example, Wikipedia has numerous functions such as undo edits so the junk or spam can be removed quickly by anyone. If an user's

edits are removed regularly, the quality of that user's contribution is questionable, and in some cases, users with questionable reputations might even lose the ability to contribute. Therefore, by implementing a trust system. Users can vote for a user's credibility based on their behalf and their Facebook friends can also authenticate an user's true identity. If user identities can be identified, further concrete information can be provided by those authenticated students and further improve the reliability of MyNext.

Another possible improvement is to improve the algorithm of prediction. Since right now the predictions are based solely on students input, if a trusted source of school or program ranking can be found, then the rankings of the schools could probably help the prediction. Additionally, a lot more hidden information can be collect from the system that can probably help the prediction, also. Hit count of a program, number of comments, number of votes, and probably some other metrics that can be found and collected from the Web server logs can also help to determine the popularity of different programs. User comments could be another source of information. By using data mining, knowledge could be formed from extracting key words from comments. The knowledge gained from comments is also a form of collective intelligence and may even further help the predictions. By combining statistical models and human opinions, the accuracy of MyNext predictions can be greatly improved (Nagar & Malone, 2011).

By bringing in the latest concept of Web development as well as the latest Web technologies, we hope MyNext can revolutionize the current CCSs and bring online CCSs to the next level.

References

- Barsky, E., & Giustini, D. (2007, December). Introducing web 2.0: wikis for health librarians. *Journal of the Canadian Health Libraries Association*, 28(4), 147–150.
- Brandon, D. M. (2008). *Software engineering for modern web applications: Methodologies and technologies*. IGI Global.
- Eckerson, W. W. (1995). Three tier client/server architecture: Achieving scalability, performance, and efficiency in client server applications. *Open Information Systems*, 10(1).
- Friedman, V., & Lennartz, S. (2009). *The smashing book*. Smashing Media GmbH.
- Halpin, H., Robu, V., & Shepherd, H. (2007). The complex dynamics of collaborative tagging. In *Proceedings of international conference on world wide web*.
- Heino, R. D., Ellison, N. B., & Gibbs, J. L. (2010). Relationshipshopping: Investigating the market metaphor in online dating. *Journal of Social and Personal Relationships*, 27, 427–447.
- Kaplan-Moss, J., & Holovaty, A. (2007). *The definitive guide to django: Web development done right (expert's voice in web development)*. Apress.
- Keith-Spiegel, P., & Wiederman, M. W. (2000). *The complete guide to graduate school admission: Psychology, counseling, and related professions*. Psychology Press.
- Linden, G., Smith, B., & York, J. (2003). Amazon.com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1), 76–80.
- Nagar, Y., & Malone, T. (2011, March). *Combining human and machine intelligence for making predictions* (Tech. Rep.). MIT Center for Collective Intelligence.
- Netcraft Ltd. (n.d.). *March 2012 web server survey*. Available from <http://news.netcraft.com/archives/2012/03/05/march-2012-web-server-survey.html> (Online; accessed 29-Mar-2012)
- O'Reilly, T. (2007). What is web 2.0: Design patterns and business models for the next generation of software. *Communications & Strategies*(1), 17–37.
- Paulson, L. D. (2005, 10). Building rich web applications with ajax. *Computer*, 38(10), 14–17.

- Pempek, T. A., Yermolayeva, Y. A., & Calvert, S. L. (2009). College students' social networking experiences on Facebook. *Journal of Applied Developmental Psychology, 30*(3), 227–238.
- Peters, I. (2009). *Folksonomies: Indexing and retrieval in web 2.0*. De Gruyter.
- Q-Success. (n.d.). *Usage of operating systems for websites*. Available from http://w3techs.com/technologies/overview/operating_system/all (Online; accessed 26-Mar-2012)
- United Nations. (2008). *2008 state of the future*. Millennium Project.
- Wagner, C. (2004). Wiki: A technology for conversational knowledge management and group collaboration. *Communications of the Association for Information Systems, 13*, 265–289.