Abstract

Constructing their own study plan is always one of the greatest challenges for first-year students. Currently, universities assign teaching staff members to be academic advisors to assist students to consolidate their own study plan, yet academic advisors can hardly spare any capacity to understand strengths and personality of each student. AI Student Advisor learns from students’ previous grades to understand their strengths to provide academic advices accordingly. This paper describes the design and implementation of AI Student Advisor, a mobile application that consist of an ML grade prediction engine and an academic advising community platform. This paper introduces the process and consideration throughout the development. Besides, this project also discuss the development process of the grade prediction module.
Acknowledgement

We would like to express our greatest gratitude to the following people for their help. Without their help, this project might not be able to be completed.

Dr. T. W. Chim: We would like to thank Dr. Chim for being our project supervisor. He was always supportive and gave us timely and helpful advice.

Dr. S. M. Yiu: We would like to thank Dr. Yiu for giving us timely reminder about the difficulty in training the grade predication module. We have made improvement to our grade prediction approach according to his advice.

The staff of the department: We would like to thank the staff, especially staff of student office for providing past students’ academic performances to us. Without the data, our grade prediction module could not be trained.

HKU CS students: We would like to thank every HKU CS student who had contributed in our project by providing valuable real-world data of past academic performance and user feedback. Without the data, our project is not achievable.
# Table of Contents

Abstract ......................................................................................................................................................... 1
Acknowledgement ............................................................................................................................................. 2
List of Figures .................................................................................................................................................. 5
List of Tables .................................................................................................................................................... 6
Abbreviations .................................................................................................................................................. 7

1 Introduction .................................................................................................................................................. 8
  1.1 Background ............................................................................................................................................. 8
  1.2 Previous Works ...................................................................................................................................... 8
  1.3 Objectives ............................................................................................................................................. 9
  1.4 Scope .................................................................................................................................................... 9
  1.5 Deliverables ......................................................................................................................................... 10
  1.6 Outline of Report ................................................................................................................................ 10

2 Methodology .............................................................................................................................................. 11
  2.1 Development Approach ......................................................................................................................... 11
  2.2 System Architecture ............................................................................................................................... 12
    2.2.1 iOS .................................................................................................................................................... 12
    2.2.2 Android .......................................................................................................................................... 13
    2.2.3 Web Service ................................................................................................................................... 14
  2.3 Programming Languages, Frameworks and Services ............................................................................ 16
    2.3.1 Frontend ......................................................................................................................................... 16
    2.3.2 Backend ....................................................................................................................................... 17
  2.4 Functions Design .................................................................................................................................. 18
    2.4.1 Authentication ................................................................................................................................. 18
    2.4.2 Grade Prediction .............................................................................................................................. 19
    2.4.3 Post Sorting and Filtering (My Post / Bookmarked / Hot / Chronological) .................................... 20
    2.4.4 Create Post ..................................................................................................................................... 22
    2.4.5 Bookmark Post ............................................................................................................................... 22
    2.4.6 Add Comment ................................................................................................................................. 23
  2.5 Data Collection Methods ....................................................................................................................... 25
  2.6 Project Milestones ................................................................................................................................... 25

3 Results ....................................................................................................................................................... 26
  3.1 User Interface ....................................................................................................................................... 26
List of Figures

Figure 1. The six major procedures in a Agile Software Development Lifecycle, which starts from “meet” then go through the loop for a serval times before it ends [7]. ........................................11
Figure 2. The interaction between the three objects in the MVC architecture [9].................................12
Figure 3. The interaction between the three objects in the MVP architecture [11]............................13
Figure 4. The interaction between the three objects in the MVVM architecture [12]........................13
Figure 5. A REST architecture diagram showing the connection between components [17]. 14
Figure 6. A demonstration on the use of DragRefreshAndLoadMoreTable ................................16
Figure 7. The system sequence diagram of the Authentication function ........................................18
Figure 8. The system flow of the grade prediction function .........................................................19
Figure 9. The system flow of the function of changing the sorting / filtering methods ........20
Figure 10. Algorithm of calculating hot posts [20] ........................................................................21
Figure 11. The system flow of Create Post function ....................................................................22
Figure 12. The system flow of the Bookmark Post function .......................................................22
Figure 13. The system flow of the Add Comment function .......................................................23
Figure 14. Navigation drawer (left) & bottom navigation (right) layout ......................................26
Figure 15. Layout of iOS version Facebook app [22] .................................................................27
Figure 16. Navigation drawer of AI Student Advisor .................................................................27
Figure 17. Home screen of the app showing the newest posts ...................................................28
Figure 18. User interface of creating a post ................................................................................29
Figure 19. User interface of adding a tag ..................................................................................29
Figure 20. A tag “COMP9999” being added .............................................................................29
Figure 21. User interfaces of viewing comments and adding comments ..................................30
Figure 22. User interface of editing a post ...............................................................................31
Figure 23. User interface of logging in ......................................................................................32
Figure 24. User interfaces of registration ..................................................................................32
List of Tables

Table 1. Project milestones. ......................................................................................................................25
Abbreviations
1 Introduction

1.1 Background

At the beginning of each semester, university students are most likely to ask questions like, which course suit strength or interest. Students often find that the courses they choose are not suitable for them due to the lack of understanding to their own strength or the requirements of the courses. The withdraw rate varies among different courses, while a university lecturer, Ray Archer, claimed for a rate of 10% in introductory courses [1]. Besides, a recent study conducted by the Australia government found that one third of the students drop out of the first subject they select owing to their rough consideration before making their decision [2].

Currently, many students try to obtain aids from online resources or advice from senior students, however the information obtained hardly suits the needs of them because their strengths and personalities are not considered. The best assistance to be sought is academic advisors assigned by faculties. They provide assistances for students in making their own study plan. Nonetheless, it is over demanding for academic advisors to understand strengths and personalities of every student due to the lack of human resources. In short, university students are currently receiving insufficient support in constructing their own study plan, while providing assistances by means of academic advisors can be improved to provide more timely advices to students.

1.2 Previous Works

Artificial Intelligence (AI) can learn the strengths and personalities of each student and substantially replace the work of academic advisors. Especially by machine learning (ML), it is believed that the relationship among the performances of students in different courses can be found, and eventually, grades of students can be projected from their previous performances before they actually take a particular course. The concept of applying ML technologies in grade prediction has been proved by two researches conducted by George Mason University (GMU) and Information Technology University (ITU) respectively. The research of ITU aimed to identify students with special needs with their historical performances [3]. ITU has implemented three grade prediction modules with ML techniques of Collaborative Filtering (CF), Matrix Factorization (MF), and Restricted Boltzmann Machines (RBM) respective, and then compare their accuracy in grade prediction. The result shows that RBM is the most accurate technique applied. On the contrary, the research of GMU took a more human-oriented approach, which aimed to identify the relationship between instructors’ characters and student’s performances [4]. GMU compared different techniques of ML and found that a hybrid of Factorization Machines (FM), Random Forests (RF), and Personalized Multi-Linear Regression (PMLR) techniques gives the most accurate result. However, none of the research has been implemented into a usable user-product for the use of students and tutors.
There is a user-product in the market using AI technologies to learn about grade distribution of each course and provide service online to help students to project their grades in different courses [5]. The application does not consider students’ ability but only present the grade distribution to users. Yet, it is believed that performance of a student does not only depend on probabilities, but strengths of them. Personality should, as well, be considered when selecting a suitable major of study.

Previous development of AI academic advising tools demonstrates the possibility of it, yet, it can be further developed to learn more dimensions of a student to give more personal advises to them.

1.3 Objectives

This project aims to develop a mobile application using existing ML technologies to provide academic advices to students. The mobile application includes two major features: a grade prediction module implementing suitable machine learning technologies which help students evaluate their performances in different courses easily and accurately; and an online community for course selection related topics which allow students to obtain courses information. The first feature learns students’ strengths by their historical performances to project the grades they will obtain in future courses. The second feature allows students to evaluate the suitability of the course content to their interest by acquiring human advices. In addition, the online community includes stimulating social networking and gamified elements to engage university students with the application.

1.4 Scope

The project is scoped within HKU CS in the extension of this project. Building a grade prediction module with ML techniques requires a huge number of samples (i.e. students’ transcript). To ensure that the number of samples collected is sufficient, only HKU CS students’ course performances will be considered in this project. Their historical performances are relatively easier to be obtained within the team’s capacity. To further limit the variance, only courses offered by HKU CS department will be considered since they all hold a subset of the programme learning outcomes of HKU CS leading to a simpler classification.

The mobile application will support iOS and Android platform in the scope of this project. These two mobile OS share over 96% of the market worldwide while over 99% of the market in Hong Kong [6]. Deploying on these two platforms meets the industrial standard, moreover, acceptably caters the needs of the targeted user group.
1.5 Deliverables

The final product of the project is a mobile application providing grade prediction function and hosting an online community for academic planning related topics. The mobile application shall support both iOS and Android platform.

The grade prediction function allows students to predict the grade they will obtain in any courses they may take in the future. Students are only required to provide their performances in the courses they have already taken for obtaining an accurate result in the grade prediction. Students shall be able to re-visit the result of previous grade prediction or run a new grade prediction at any time.

The online community operate in the way of a forum. Students shall be able to ask and to answer questions related to academic planning. The online community consist of features which help students to access the information of a specific course easily to enhance its educative purpose.

1.6 Outline of Report

The following parts of the report summarise the development process of the application. Section 2 will first talk about the development approach of the project and justify engineering choices including the system architecture in the 2 OS and the backend server. Next, it will talk about the designs of functions in the mobile application and the grade prediction module. Moving into section 3, it will present the user interfaces of the mobile application as well as the training and testing result of the grade prediction module. In addition, it will sum up the difficulties the team have encountered throughout the whole process. Finally, in chapter 4, it will summarise the contribution of the project and foresee the further work in this area of study.
2 Methodology

2.1 Development Approach

![Image of Agile Software Development Lifecycle]

*Figure 1. The six major procedures in a Agile Software Development Lifecycle, which starts from “meet" then go through the loop for a serval times before it ends [7].*

An Agile software development cycle has been adopted as the development approach in this project. The Agile is an iterative which focuses in accommodating changes and a quick delivery of products [8]. The Agile is an incremental development approach, which the project is broken down into smaller parts and modules. Requirements and development plan are set at the beginning of each iteration and ends by installing the product of that cycle into the final product. Comparing with traditional waterfall model, waterfall model follows definite phases and fixed requirements. Each stage of the development is long while the flexibility is low. Product will not be available at the early stage of development causing it very difficult to consult the supervisor and make adjustment according to his requirement. Features of the Agile is important to the development of this project for better partnership with the supervisor and understanding the feasibility of the concept at the earlier stage of the development.

In this project, the online community has been completed in the early stage of the development since it is less dependent on the development progress of other modules; while the grade prediction module and other minor modules are added to the application in the later stage of the development since they have more prerequisites including users’ data.
2.2 System Architecture

2.2.1 iOS

A Model-View-Controller architecture has been adopted as the system architecture of the iOS application in this project. The architecture is highly supported by the Apple Inc. and the iOS community. As a result, MVC has performed the best among all architectures in the iOS platform [10].

However, there is a common issue of the MVC architecture, which the controller is overweight. Developers tend to allocate all logics which do not belong to the view or the model into the controller, making a massive controller. In such manner, the team is implementing a web service layer and interface builders to subdivide the controller into modules to maintain the balance between three objects in the MVC architecture. For details of the implementation of the iOS application, please refer to Mr. KWOK Cheuk Lum’s report on the same project.
2.2.2 Android

A Model-View-Controller architecture has been adopted as the system architecture of the Android application in this project. It is a traditional architecture supported by the Android community, as well as most of the system engineering developing GUI systems [13]. The architecture aims to separate logics of an application into three models so that each model only needs to handle their respective codes and communicate with their neighbouring model (see Figure 2). The MVC architecture produces well organised and high maintainability systems.

The MVC architecture consists of a model, a view and a controller [14]. View is the object which users interact with. It is also responsible for rendering the user interface. In the Android application, it includes all the static views which users see and touch. It passes users’ interactions to the controller. In general, the controller responds to users’ interactions and handles the logic between the view and the model. In the android application, it includes different event listeners, logics to update the view, network handling and the interactions with the database. Lastly, the model is independent of the view. It principally handles data and
business logics. In the Android application, most of the data of the online community is stored in the network server, which is not considered as the model in the scope of the Android application. The model of the Android application stores users’ information and network data (e.g. cookies).

In recent years, there is a debate among the Android community on the best architecture for developing Android application. The issue of following the MVC architecture is the overweight of the controller [15]. All the logics which do not belong to the view or the model will be arranged into the controller causing it terribly difficult to maintain the application. The use of Model-View-Presenter (MVP) and Model-View-ViewModel (MVVM) architectures is broadly supported by different Android developers [16]. In short, the two new architectures allocate the work of updating the user interface, which is the responsibility of the controller, to the view (see Figure 3 and Figure 4). The Android application in this project is not adopting any of the newly proposed architecture because most of the complicated logics are implemented on the backend server. The controller in the application only retrieves data and updates the display. If the view shares workload of the controller, it will cause overweight of the view while underweight of the controller (or Presenter / ViewModel).

2.2.3 Web Service

A REST architecture has been adopted as the system architecture of the web service in this project. The REST architecture enables the two versions (i.e. iOS and Android) of the application to access resources host on the web. The application can access the web service with uniform and predefined set of stateless operations, which is JSON in this project (see Figure 5). The online community is hosted on the web service; therefore, the application sends JSON requests to the REST architecture every time when it performs activities in the forum. The REST architecture is adopted because of its scalable which enable the project to be developed in a faster pace while preventing the system from any suspension of service as
it grows in the future. For more details of the architecture of the web service, please refer to Mr. CHAN Lap Kiu and Mr. CHAU Chun Wing’s report on the same project.
2.3 Programming Languages, Frameworks and Services

2.3.1 Frontend

The mobile application is implemented in native languages on iOS and Android respectively. Objective-C and Java are chosen as the programming languages to be used in the development of the two platforms respectively. These two languages are the most typical language in their corresponding platform; hence their community is large facilitating the development process. Newer languages like Swift and Kotlin are not preferable because their communities are relatively small [18], making obstacles for the team which is fresh to mobile app development in the development process. In addition, Swift is a new language under development. There may be unexpected compatibility or compilation problems occasionally in the development process. Hybrid languages, including React Native and Flutter, have also been considered but not adopted due to the nature of the application. The application implements its most complicated logics at the backend while the front end focuses on advancing users’ experience. The proportion of reusable logics is insignificant while owing to different users’ habits, localisation should be made to the UI in the two versions of the application, thus the reusability of the UI resources is unsubstantial either. The development process of the project cannot benefit from the better reusability of hybrid languages. Besides, the application may utilise machine dependent hardware feature which may not be supported by hybrid languages. As a result, native languages are chosen in this project.

![Figure 6](image)

*Figure 6. A demonstration on the use of DragRefreshAndLoadMoreTable*

To smooth the development process, third-party frameworks and libraries are included in the application. AFNetworking and OkHttp are applied to handle HTTP requests. They simplify the work to manage request / respond headers and cookies. Besides, libraries are also employed in the design of the user interfaces. DragRefreshAndLoadMoreTable and Ultra-Pull-to-Refresh with Load-More are the libraries responsible for creating the layout of the
forum (see Figure 6). They divide the forum into pages to minimise the size of a single HTTP requests, which allows users to get the latest post sooner. They also enable the use of gestures to gain full control of updating and back dating in the forum.

2.3.2 Backend

The web service in this project is implemented in Node.js environment with MongoDB database. Node.js’s asynchronous feature benefits the multi-user system in this project by improving the efficiency in responding to users’ request. In other respects, MongoDB database is a NoSQL database which does not require predefined structure producing a more scalable data structure. It is more suitable for rapid developed projects and start-up projects covering this project. Over and above, Node.js and MongoDB are chosen as a pair because it is a common industrial practice to consider them as a combination.

The backend of the project is host online on Heroku and MongoDB Cloud which provide PaaS. Online host are suitable for small projects containing low sensitivity data like this project. PaaS providers manage the platform in their environments, thus save the team’s effort in setting up a server but to focus in the core features of the project.
2.4 Functions Design

The application consists of six major functions. This section demonstrates how each function utilise system resources and how data flows among different domains in the perspective of the mobile application. This section also shows how user interact with all the functions. Detail design of the grade prediction will be discussed in Section Error! Reference source not found..

2.4.1 Authentication

The authentication function is not related to the objective of the project, but it is the prerequisite for users to access to access all the below functions. The project is in form of a member-based system. User’s identity should be verified for distinguishing each user in the online community as well as in the grade prediction function. By differentiating each user, the application can remember the preference of each user, hence simplify the way how users interact with the application (e.g. bookmarks, votes, etc). Moreover, remembering users’ record in grade prediction can reduce the effort which user need when inputting their grades. It enables users to refer back to their last result and quick run another grade prediction test simply by updating with a few new records.

For the first time a user launches the mobile application, they are required to register an account. Users should provide their email address as their username and create a password
for their account. They may provide their nickname for the communication purpose when using the online community. They can reset their passwords with their email address if they have forgotten them.

When a user submits a request to create an account, the application pass all users’ information (as mentioned above) to the system backend (see Figure 7). The backend checks the provided email address among all users account to avoid replication. If no previous record found in the database, the backend encrypts the password and creates a temporary account in the database. The backend instructs the email server to send a token to the user’s email address. The user should enter the token to the application to activate. If the token is not entered, the temporary account will be removed after a period of time. After the activation of the account at the backend, it will send acknowledge to the application. The users can then perform login activity.

When the password is sent to the backend, it is hashed to avoid hackers from retrieving it. Bcrypt is adopted in this project to hash passwords because it is specifically designed for password hashing [19]. It adds 128-bit salt to the hashes and allows modifiable number of iterations in hashing to defence rainbow attacks and brute force attacks.

When a user perform login, the application performs the first step of data validation by checking the format of the email address and the length of the password before passing the user’s information to the backend for verification. The user’s information will be checked against the record in the database. If record is found, the backend declares that the login is successful while the application will direct the user into the online community. Else if the record is not found, the backend replies with an unsuccessful message. The application will indicate that the login information is incorrect on the login screen.

2.4.2 Grade Prediction

![Figure 8. The system flow of the grade prediction function](image)
Grade prediction is one of the key features of the project. The grade prediction module generates a predicted grade to all the courses that the user has not taken yet. Users only need input the grades of all the courses they have taken to the mobile application. The minimum number of courses is not set for the grade prediction, however the more record input by users, the more accurate of the result of grade prediction will be.

When a user request to generate a report, the application will then pass their data to the grade prediction modules for report generation (see Figure 8) (details of the implementation of the grade prediction module are in Section Error! Reference source not found.). The grade prediction module then generates a report and then responds to the users’ application. The report is shown in the application when it receives the respond. Users can always refer back to their reports in the application.

2.4.3 Post Sorting and Filtering (My Post / Bookmarked / Hot / Chronological)

![Figure 9. The system flow of the function of changing the sorting / filtering methods](image)

When users are logged into the application, they can access to the posts in the forum. Right after they have logged in, the application automatically send request to the database to retrieve 5 posts. These posts are defaulted to be sorted in chronological order. Users can view more post by sending another request to the server to retrieve 5 more post. Only 5 post will be delivered over the network at any instance to limit the data transfer time over the network.

Users can view posts in different categories, including posts created by the user (please refer to Section 2.4.4 for how to create a post) and posts bookmarked by the user(please refer to Section 2.4.5 for how to bookmark a post). Users can also sort posts in accordance of their hotness or freshness (default soring). Sorting help users to obtain the type information they are the most interested in and trace back the posts they have visited already.

The application send request to the database when changing the sorting / filtering method (see Figure 9). The logics of sorting and filtering are implemented at the backend; thus, the database will put the post in the requested order in its responds. The database always responds with a maximum number of 5 posts to each request.
The design of the algorithm for calculating the hotness of posts has taken references to the implementation of Reddit on community, one of the most popular forums around the world. Reddit calculate hotness of post by their number of up-vote received, reduced by a negative effective of down-vote received and a decaying effect of time (see Figure 10). The project has adopted a similar approach but with amendments to fit into the constrains of the online community to calculate hotness of the post. Number of up-vote in the Reddit’s algorithm is changed to number of users who have bookmarked. On the other hand, the effect of bookmarks in the online community have the same importance as that of the up-votes in Reddit, which the first 10 bookmarks made have the same affect in the hotness of the post as the following 100 bookmarks made.

The up-vote-down-vote system is not adopted in the project due to the nature of the online community. The online community is not for social networking purpose where people can express their opinion by giving votes. The online community serves the purpose of academic advising which allows students raise question in this area. People are not likely to express their agreement or disagreement on questions but on answers or statements. As a result, the bookmark system is adopted to allow users to recap the answer given previously or look for answers of questions which had not been answered at the time they last visit. Up-vote-down-vote is implemented in comment sections where answers most like be.
2.4.4 Create Post

Since the online community is for academic advising purpose, students are encouraged to raise related questions in the forum. All users can create posts in the online community. Each post shall have a title, and content. In addition, users can add multiple tags to the post to catch the attention of their targeted audience and to make the post easily to be searched. Users are encouraged to put course related information including course codes, topics, and related concepts in tags.

Once the user has completed the creation of a post, the application passes the post data together with the user’s information to the database (see Figure 11). The database acknowledges the application after the successful creation of the post. The database organises all posts in a table making them available to all users. Users can access all the post in the community by sending request to the database (please refer to Section 2.4.3 for details).

2.4.5 Bookmark Post

Figure 11. The system flow of Create Post function

Figure 12. The system flow of the Bookmark Post function
The online community is expected to be a question-based forum for academic advising related topics. Bookmark is an important function for users to interact with the questions in the forum. Users are suggested to use the bookmark function when they would like to remember the answer of a specific question or the answer has not been given to a specific question which they may want to explore again at later time.

When a user makes a bookmark on a post, the application passes “post id” and “user id” to the back end to specify that particular user have bookmarked that particular post (see Figure 12). The database saves users’ identity and bookmark count for the purpose of sorting and filtering (see Section 2.4.3). After the record of making a bookmark has been taken, the database acknowledges the user’s application. Users can refer to the posts they have bookmarked by filtering the posts (see Section 2.4.3) or identifying the bookmark field in each post.

2.4.6 Add Comment

![Diagram](image)

*Figure 13. The system flow of the Add Comment function*

The add comment function is designed for users to reply to the questions raised by other users in form of posts in the online community. All comments are in the format of plain text. All users can reply to any post for unlimited times. This is the only direct channel provided for users to interact.

When a user creates a comment, the users’ application send the comment content, the “post id” of the post which the comment refer to, and the user’s information to the database (see Figure 13). The database acknowledges the users’ application when the record of comment is received. The database organises all comments in a manner that is accessible by all users under their corresponding post.

Users can view comments in the “comments” field of any post. Users can give up-votes or down-vote to comments to indicate their judgement on the quality of the answer given by other users. Each up-vote is weighted for 1 point while each down-vote is weighted for -1
point. The sum of points is shown in each comment to give users a point of reference on the accuracy of the answer given in that comment.
2.5 Data Collection Methods

The grade prediction module in this project is implemented with ML algorithm. Data of past students’ performances in different courses is needed to train the module. Since the scope of the project focus on HKU CS department, only data of CS students in CS subjects are needed. Two methods are employed for collecting the required data.

The HKU CS department is providing transcripts of past students to the project. These records are records from graduated students, so the record will be relatively completed. With sufficient number of courses taken in the past, a relationship can be observed easily. These data also provide reference points on advanced courses which may not be found in current students. However, these transcripts are from students who graduated in the same year which may give bias to the data with records in a single year. Courses taken by these graduates may have changes in their contents, instructors, or replaced by other courses.

The team is publishing a questionnaire to collect information from all HKU CS students non-selectively. The questionnaire is uploaded and receive responds online. The questionnaire is released via mail and in-class announcement. These records reflect the performance of all students across the years. The record is relative timelier than those which are provided by the department because the audience of the questionnaire is current students. However, this project focuses on CS courses which most of the students only enrol them in year 2 or above, data collected via questionnaire may not contain enough past performance references for observing the relationship among courses.

2.6 Project Milestones

<table>
<thead>
<tr>
<th>Date</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sept</td>
<td>Deliverables of Phase 1</td>
</tr>
<tr>
<td>Oct</td>
<td>Development of mobile app (Basic UI, GPA calculator, Personality test)</td>
</tr>
<tr>
<td>Nov-Dec</td>
<td>Development of mobile app (Forum feature)</td>
</tr>
<tr>
<td>Jan</td>
<td>Deliverables of Phase 2</td>
</tr>
<tr>
<td>Jan-Feb</td>
<td>Data Collection (Questionnaire) +AI model research + training + mobile app testing</td>
</tr>
<tr>
<td>Feb-April</td>
<td>AI model research + training + migration with mobile app</td>
</tr>
<tr>
<td>April</td>
<td>Deliverables of Phase 3</td>
</tr>
</tbody>
</table>

Table 1. Project milestones.
3 Results

3.1 User Interface

Layout design is crucial to a mobile application for there is no user manual nor tutorial provided to users. Users use the application according to what they perceive from the UI. There are norms in UI design, such that people are used to the meaning of those representations [21]. For example, a button with three horizontal stripes means navigation drawer, and a bar at the bottom means switching functions (see Figure 14). Both representations are widely used for switching between functions and exhausted the choices of representation for such purpose. The application implemented the navigation drawer layout.

Navigation drawer layout is a layout which shows all of its functions in the navigation drawer, typically on the left side of the screen. This layout is commonly seen in mobile applications developed by Google. The development strategy of Google in developing mobile application is to scale down the scope of each application so that the focus of the applications is clearly defined. Although there are different functions listed in the navigation drawer, they share similar layout.
Bottom navigation layout is a layout which shows the application’s major functions in the bottom bar which allows users to switch to different functions by one simple click. Most mobile applications adopt this layout style. This layout is suitable for mobile applications with multiple featured functions and the users are always switching between functions. The functions usually have very different layout, which users can identify the significant differences when switching between functions. Taking the iOS version of Facebook application as an example, the bottom bar enables users to switch to different modules, including homepage, friend requests, massages and notifications (see Figure 15), which have completely different layout designs and nature of function.

Navigation drawer layout is implemented in AI Student Advisor to simplify the UI of the application. The application has a single focusing function. It features the online academic advising community where the grade prediction engine will be integrated to. Although there are functions for switching the methods of reading posts in the online community, all functions share similar layout. Users can be more focus in the main function with the current UI design.

AI Student Advisor implements the navigation drawer layout (See Figure 16). All functions are presented in the navigation drawer in the left. The top part of the navigation drawer shows users’ information including their icon (had not been set by the user in Figure 16), nickname (“Ben”), and cumulative grade point average (CGPA) (“4.3”). The menu in the navigation drawer is first followed by sorting methods of the online community, then followed by other functions of the mobile app. Functions can be launched by selecting the respective links in the application drawer. Details of the layout of each specific function will be discussed in the following sections. Users can close the navigation drawer by 3 ways: pressing the back
button (<) on the top left corner, pressing at the diminished area on the right of the navigation drawer and swiping to the left of the screen. In addition, Android version supports the fourth way to close the navigation drawer, which is by pressing the back button on mobile devices.

3.1.1 Post Sorting and Filtering (My Post / Bookmarked / Hot / Chronological)

Figure 17. Home screen of the app showing the newest posts.

### demonstrated the basic interface of a post. Taking the top post as an example, each post includes a title (“Mobile app”) in the first row, content (“Is the mobile app course a good grade course?”) in the second row, tags (“Mobile App”) in the third row and number of comments (“0” at the bottom right corner next to the comment icon (```))). There is also a bookmark button (```) and comment icon (```) in the bottom row.

Pressing the bookmark button performs the “adding to bookmarked” function to posts which have not been bookmarked but “removing from bookmarked” function to bookmarked posts. Bookmarked posts show a highlighted bookmark (```) (“Which course should I take?” in Figure 17) while posts which have not been bookmarked show a normal bookmark (```) (“Mobile app” and “Which common core course to take?” in Figure 17). All bookmarked posts are shown in the list of “bookmarked” automatically when choosing “Bookmarked” in the navigation drawer.

Pressing on any area (except the bookmark button) of a post shows the details of the post. The app will start a new screen showing the details of the post, all comments and allowing users to add new comments. Details of the comments screen will be discussed in Section 3.1.3.

The user interface provides four types of view orders and filters, including self-created posts, viewing bookmarked posts, viewing hot posts and viewing posts in chronological order. The
home screen of the app is defaulted to show the newest posts in the online community. User can select other sorting methods of the posts in the navigation drawer (see Figure 16), which can be opened by clicking the menu button (≡) on the top left corner of the screen (see Figure 17). In the navigation drawer (see Figure 16), users can select “My Posts” to display posts in the online community which are created (see Section 3.1.2) by the user in chronological order. By selecting “Bookmarked”, users can view posts which are marked as bookmarked by the user. Last but not least, users can select “Hot” to display posts in descending order of hotness. If users would like to restore the default order (i.e. chronological order) of the forum, they can select “Forum” in the navigation drawer.

Viewing posts in different orders or filters does not affect the layout of the online community, but only the content and order of the posts. Viewing orders and filtering methods can be changed in the same way as in the home screen of the app in any other screens listed above.

3.1.2 Create Post

![Figure 18. User interface of creating a post.](image1)

![Figure 19. User interface of adding a tag.](image2)

![Figure 20. A tag “COMP9999” being added.](image3)

When a user clicks on the add post button (+) at the top right corner on any page of the online community (see Figure 17), the app will create a new screen of creating a post (see Figure 18).

Figure 18 shows the user interface when users create a post. Users need to enter a post title in the top text box (with the placeholder “Post Title”). The larger text box is for entering the content of a post. A tag can be added by clicking the “Add” button in blue next to “Tags”, then a dialog box (see Figure 19) will pop up. Users can set the text to be displayed in a single tag by typing in the text box of the dialog box (with the placeholder “Please enter a tag”). Users can complete adding a tag by clicking “Add” or return to the previous screen without creating a tag by clicking “Cancel” in the dialog box. Tags successfully created will
be listed below “Tags” (see Figure 20). Tags can be deleted by clicking on the created tags. Multiple tags can be created by repeating the above procedures. A post can also be created without any tag. After filling all the required fields, users can create the post by clicking “Post” in the top right corner or cancel the creation and return to the community screen by clicking the return button (<) at the top left corner on the creating post screen (see Figure 18 and Figure 20).

3.1.3 Add / View Comment

![Figure 21. User interfaces of viewing comments and adding comments.](image)

After a user pressed on any area of a post in the home screen, the app shows the screens of viewing and adding comments (see Figure 21). This page shows the details of the post. Details include creator of the post (“ben”), title of the post (“Mobile App”), content of the post (“Is the mobile app course a good grade course?”) and number of comments (“1” in the bottom right corner next to the comment icon (лежащий)) (see Figure 21a). There is also a bookmark button (лежащий) which performs the same function as the bookmark button (лежащий) in Figure 21.

Under the details of the post are the comments (see Figure 21a). No comments are shown in this area if there are no comments. Each comment shows its creator (“Victor”) and content (“Yes, best course ever!”) (see Figure 21a). New comments can be created by tapping at the edit text area which is docked at the bottom of the screen. The keyboard pops up after tapping the edit text box and push it up on top of the keyboard (see Figure 21b). Users can enter the content of the comment in the edit text box with the virtual keyboard and submit the comment using the send button (лежащий). The send button is dimmer in colour and disabled until the user enters a string that contains characters other than spaces or newlines. After submitting the comment, the virtual keyboard is hidden, and the newly created comment is
added at the bottom of the list (see Figure 21a). The screen will automatically scroll to the bottom to display the newest comment if it was not.

If the user is the creator of the post, an edit button (✔️) is shown in the top right corner (see Figure 21a). Users can edit the post by pressing the edit button. An edit post screen will be launched. Details of the edit post screen will be discussed in the next section.

3.1.4 Edit Post

![Image](image_url)

*Figure 22. User interface of editing a post.*

Users can edit a post in the edit post screen (see Figure 22) navigated from the comments screen. The screen is nearly the same as the create post screen so that users do not need to adapt to the UI again. The only difference is that it automatically fills in the original post title (“Mobile app”) in the smaller text box, the original post content (“Is the mobile app course a good grade course?”) in the bigger text box, and the original tag(s) at the bottom. Users can edit the post title, content and tags just like how they create a post (see Section 3.1.2). After editing the post, users can post the amendment by clicking “Post” at the top right corner to overwrite the original post or cancel the amendment by clicking the return button (←) at the top left corner.
3.1.5 Login / Register

![Figure 23](image1.png)

**Figure 23. User interface of logging in.**

The app shows the login screen (see Figure 23) when users launch the app for the first time or after logged out. Users can login by pressing “Login” button after entering their username and password.

![Figure 24](image2.png)

**Figure 24. User interfaces of registration.**

If a user does not have an account, he or she can click “Register” in blue at the bottom of the screen (see Figure 23). Then a registration screen is shown (See Figure 24a). After filling in his or her email and password, he or she can click “Continue” in blue at the bottom of the
screen to navigate to the next step (see Figure 24b). The server generates a one-time token via email after receiving the create-account request to verify that the email address the user provided is a valid email address. The user has to enter the token into the text box and click “Continue” in blue under the text box. They can click “Resend token” in blue at the bottom of the screen if they have not received the token. After verification, a personality test is shown (see Figure 24c). Users can complete the test and login to the app or skip it for now by pressing “skip” in blue at the top. Then the registration process is completed, and the application redirects the user to the home screen.
3.2 Module Training

128 records have been collected via transcripts provided by the CS department and questionnaire. Only records with 10 or above CS courses taken are modelled for training to ensure that a relationship among performance in different courses can be identified. Responses with less than indicative number of courses may lower the accuracy of the prediction by giving extreme relationships. 100 records are shortlisted for training the module.

The training is separated into 2 parts — with 50 records trained each time. The first training only focuses on inputting data into the model, while the second training test verify the result of the first training meanwhile inserting record into the model.

The result after two parts of the training are similar, which is an error range of ± 0.6. The training engine return a data model to be hosted on the database. The backend search for the most similar record and return the record as the predicted grade. Users can access the grade prediction via the application.

3.3 Difficulties

3.3.1 Lack of Data

The scope of the project is specific and well defined making the variation of data to be small. However, a small scope also come along with a small user group. Collection of data is extremely difficult because the number of graduates in the past few is much less than the number of students nowadays. On the other hand, data provided by junior students do not make much contribution to the project since there are a lot of CS electives, but the number of CS courses they have taken is not significant to estimate a relationship among performances in courses. It is recommended that the next phase of development should extend the scope to majors with many students, but with a low variation in courses. Other programmes offered by the engineering faculty may be good choices since they have much less elective comparing with CS major.

3.3.2 Variation of Data Types

At the propose-stage of the project, the team had considered using human feedback in the online community to improve the accuracy of the grade prediction model. However, it is found impossible because of the variation in human activities. it is beyond the capacity of the team to handle various data format and data type. During this project, the importance of integration of different data format and human interaction is observed. It is believed that further studies in this field may give great contribution to the development of ML.
4 Conclusion

This project has proven that there is a relationship among students’ performances in different courses. The training result of the grade prediction model is satisfactory. HKU CS students can now access to AI Student Advisor to seek advices from the grade prediction module as well as advices from other students in the online community. Students can select electives according to their strength and interest.

In the development process, a significant effect of human factor is observed in students’ historical grades in courses giving great difficulties in giving accurate results. Although the error range is small, the range shows limitation of ML in predicting human behaviour. There is no evidence found from the project, but the error range of $\pm 0.6$ may be caused by the difference of effort students put in different courses. I wish there are future studies which can either prove or disprove this hypothesis.
References


