

---

# COMP2043.GRP Final Group Report

## Social Language Learning Application

---

UNIVERSITY OF NOTTINGHAM NINGBO CHINA

APRIL 19, 2018

TEAM ACE

SUPERVISED BY DAVE TOWEY



University of  
**Nottingham**

UNITED KINGDOM • CHINA • MALAYSIA

ALBERT MILLAN – 20027140

SEN YANG – 16518733

SHIYING YU – 16518736

TINGTING ZHU – 16518749

WENPENG CHENG – 16518305

ZHENGCHUN ZHAO – 16518746

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Background</b>	<b>2</b>
2.1	Initial Problem statement . . . . .	2
2.2	Current system . . . . .	3
2.3	Proposed solution . . . . .	5
<b>3</b>	<b>Research</b>	<b>6</b>
3.1	From Mechanical Repetition to Interactive Learning . . . . .	6
3.2	Development tools for Language Learning Applications . . . . .	7
3.3	Market Analysis . . . . .	8
3.3.1	Duolingo . . . . .	8
3.3.2	HelloTalk . . . . .	8
3.3.3	Payment System . . . . .	8
<b>4</b>	<b>Software Requirements Engineering</b>	<b>9</b>
4.1	Stakeholder Profiles . . . . .	9
4.2	Requirements Elicitation Procedure . . . . .	10
4.3	Requirements Modification . . . . .	11
<b>5</b>	<b>Software Design</b>	<b>12</b>
5.1	Architectural Design Decisions . . . . .	12
5.1.1	Ionic Framework . . . . .	12
5.1.2	Angular . . . . .	13
5.1.3	XAMPP & LAMP . . . . .	13
5.1.4	RESTful PHP API . . . . .	14
5.2	Implementation Design Decisions . . . . .	15
5.2.1	Project Structure & Naming Convention . . . . .	15
5.2.2	User Authentication & Validation . . . . .	15
5.2.3	Real-time Chat System . . . . .	17
5.2.4	Posting in the Forum . . . . .	17
5.2.5	Calendar Library & Date Formatting . . . . .	18
5.3	Testing . . . . .	19
5.3.1	End-to-End . . . . .	19
5.3.2	Metamorphic Testing . . . . .	19
5.3.3	User Acceptance Testing . . . . .	19
<b>6</b>	<b>User Interface</b>	<b>21</b>
6.1	Registration . . . . .	22
6.2	Forum . . . . .	22
6.3	Chat . . . . .	23
6.4	Calendar . . . . .	23
6.5	Account . . . . .	24
<b>7</b>	<b>Supporting Software Development Tools</b>	<b>25</b>
7.1	Common File Management Tool: Github . . . . .	25
7.2	Task Management Tool: Teambition . . . . .	25
7.3	Agile Methodology . . . . .	25
<b>8</b>	<b>Requirements Evaluation</b>	<b>27</b>
8.1	Pending Requirements . . . . .	27

<b>9</b>	<b>Reflection</b>	<b>28</b>
9.1	Authentic Software Engineering Experience . . . . .	28
9.2	Lack of Experience . . . . .	28
9.2.1	Appropriate Leadership Style . . . . .	29
9.2.2	Inefficient Task Management . . . . .	29
9.2.3	Missleading Planning . . . . .	30
9.3	Cross-cultural Communication Issues . . . . .	31
9.4	Clearly Defined Roles . . . . .	31
9.5	Inefficient Exploitation of Supporting Tools . . . . .	32
<b>10</b>	<b>Conclusion</b>	<b>33</b>
10.1	State of the project . . . . .	33
10.2	Future Work . . . . .	33
10.2.1	Authentication . . . . .	33
10.2.2	Chat . . . . .	34
<b>11</b>	<b>Appendices</b>	<b>41</b>
A	Support Documents . . . . .	41
B	Details of the survey . . . . .	42
C	Software Requirements . . . . .	43
C.1	<b>Initial Functional Requirements:</b> . . . . .	43
C.2	Modified Functional Requirements: . . . . .	46
C.3	Non-Functional Requirements: . . . . .	46
D	Testing . . . . .	47
D.1	E2E Testing . . . . .	47
D.2	Metamorphic Testing . . . . .	48
D.3	User Acceptance Testing . . . . .	49
E	Minutes . . . . .	51
E.1	Minutes 1 . . . . .	51
E.2	Minutes 2 . . . . .	51
E.3	Minutes 3 . . . . .	52
E.4	Minutes 4 . . . . .	53
E.5	Minutes 5 . . . . .	54
E.6	Minutes 6 . . . . .	55
E.7	Minutes 7 . . . . .	56
E.8	Minutes 8 . . . . .	57
E.9	Minutes 9 . . . . .	57
E.10	Minutes 10 . . . . .	58
E.11	Minutes 11 . . . . .	58
E.12	Minutes 12 . . . . .	59
F	User-Manual . . . . .	59

---

# 1 Introduction

Over the last year, a team of six computer science students have developed a hybrid social language learning mobile and web application<sup>1</sup> on behalf of Vis-à-Vis (VAV) organisation, Language Center (LC) [1] and Maria<sup>2</sup> from University of Nottingham Ningbo China (UNNC). These parties, key stakeholders, envisioned the Online Language Learning Exchange (OLLE) Nottingham Advantage Award<sup>3</sup> (NAA) project as an opportunity to foster language learning within Nottingham university and, in the long run, to provide a secure language learning platform for all other users.

This paper aims to explain the app development process from the software engineering team's point of view. It is divided in the following manner. Background research is highlighted in **Chapter 2**. This includes an explanation of, and modifications done to, the initial problem statement. It also contains a clarification of the current system as well as potential areas of improvement. **Chapter 3** shows an analysis of the literature regarding mobile learning tools, and the market of the app. It is followed by a description of the system requirements in **Chapter 4**. Software design and implementation decisions, as well as software testing tools, are presented in **Chapter 5**. The user interface is shown in **Chapter 6**. Supporting team management and file sharing tools, and development methodology are explained in **Chapter 7**. An evaluation of the accomplished system requirements is outlined in **Chapter 8**. Our thoughts and reflections on the experience of the project are highlighted in **Chapter 9**. In the final chapter, we comment on the current state of the project and provide a guideline to improve it further.

---

<sup>1</sup>Hybrid app: application that contains both native and web elements, and can be executed in multiple platforms.

<sup>2</sup>Maria: codename of third stakeholder.

<sup>3</sup>NAA module: a cross-campus award supported by employers and included on the degree transcript of Nottingham University students.

---

## 2 Background

In this chapter, we present a description of the initial problem statement, as well as the modifications done to the document after negotiation with stakeholders. The current system is described next, highlighting its limitations and proposing an alternative solution

### 2.1 Initial Problem statement

According to the project description (see **Appendix A**), we were asked by VAV, the LC, and Maria to create a mobile platform that “supports and augments existing online language learning at UNNC”. It must be freely open to all Nottingham and partner University students. External participants would be able to use the app if they have paid a fee in advance.

Upon meeting with the stakeholders, information that was not present on the original document was revealed. The application is a two year long project and must have two separate functions. On the one hand, provide an educational platform that enables VAV and the LC to grade university students enrolled on the NAA language modules, enable students to share their language learning strategies, and to inform them about upcoming language learning events. These students, have to complete and submit weekly tasks via a *forum*, as well as remaining active in a *chat-room* system. VAV must manage the events on a *calendar* system, visible to the students.

On the other hand, all users should have a space where they can communicate with their friends, exchange language learning resources (multi-media, books, articles, etc.) and check a news feed<sup>4</sup> which would be updated weekly by VAV. The app should distinguish between external participants and students from Nottingham or partner universities, and charge them a fee accordingly. In other words, include a one-to-many chat system that supports file-sharing, a news feed, user-authentication and a billing system.

The team and the stakeholders have agreed to implement the former function this year while leaving the latter for the team that follows up on the project next year. We believe that completing the two functions within the one year time constraint is unreasonable, given the team member’s lack of prior mobile development experience. Thus, on the upcoming chapters of the report, the issues with regard the implementation of the second function are ignored, focusing exclusively on the NAA function.

---

<sup>4</sup>news feed: space containing specific information, and is updated regularly.

---

## 2.2 Current system

Currently, the stakeholders use a combination of resources to mimic the functionality of the forum, calendar and chat. The first two are readily available on Nottingham University’s *Moodle*<sup>5</sup> website [2]. The chat functionality is obtained using *WeChat*<sup>6</sup> groups [3]. This is a primary source of concern to the stakeholders, claiming that users are confused due to the lack of integration between the forum, calendar and chat system.

The forum enables administrators/stakeholders to add, delete or modify forum topics, in which NAA students can reply to the given topic. There are seven topics — Spanish, Korean, Japanese, Italian, French, English and Open Learning Log<sup>7</sup> (OLL) — yet only the latest one is used. The reason for this is that students are assessed on their participation in the OLL, but not in the other topics. We believe that the *Moodle* webpage is not user-friendly, making it hard for stakeholders to delete unused topics. It is assumed that they intended to use them at the start of the year, and were unable to delete them once they realized nobody was participating.

Stakeholders use of a calendar in the *Moodle* webpage to set submission deadlines and remind students about language learning events, nevertheless, functionality is limited. Students can only see a screen-shot of an outdated calendar month, having no room for interaction. Should they want to see future events, they need to download an Excel document<sup>8</sup>.

According to the stakeholders, the aim of the chat is to rapidly inform NAA students about upcoming events, get students to practice the language they strive to learn, enable students to ask questions and to get students to know each other. On a weekly basis, stakeholders change the chat topic so as to encourage communication within the chat groups.

The stakeholders agree that the aforementioned systems have failed to perform as expected. They claim that there are too many forum topics, complicating interaction for students, who often get lost and disinterested in the module. Similarly, chat participation falls soon after a new topic is posted. And the fact that the calendar is outdated and static<sup>9</sup> exacerbates the problem. The following are growing concerns of the stakeholders that our implementation aims to tackle:

- **System lacks integration.** Resources are on *Moodle*, while social interaction is carried out through *WeChat*. Students that do not interact on *WeChat* are not able to follow the forum discussions, and viceversa. Consequently, students are unable to complete their tasks, losing interest in the module.

---

<sup>5</sup>Moodle: educational software designed to manage university modules.

<sup>6</sup>WeChat: Tencent’s instant messaging application. Includes one-to-many communication.

<sup>7</sup>OLL: Forum topic in which users comment on the language learning strategy they use and why do they find it useful.

<sup>8</sup>Excel: software to store, manipulate, calculate and analyze data in cells.

<sup>9</sup>static: it is not responsive to user input.

- 
- **Increased work for stakeholders.** Low participation rates lead to students missing out on important information. Stakeholders receive more emails from students as a result, considerably increasing the stakeholders' workload.
  - **Erroneous chat purpose.** Students' purpose of the chat does not match with that intended by stakeholders. *WeChat* is perceived as an app used to have fun through social interactions [4]. This is ingrained in *WeChat* users, including the students. On the other hand, stakeholders claim that groups should support the learning of a given language, conversing on related topics. Should they use their own chat system, the stakeholders could define the purpose of the chat and, thereby, change the student's perception of the chat. For instance, *Slack's* purpose is to chat about work related issues in a simple, agile and productive fashion rather than having fun [5]. This proves that the chat's purpose can be modified.
  - **Chats have low participation rates.** Stakeholders claim that people have several groups on *WeChat*, and it is not possible to respond effectively to all of them. If the app only included the OLLE groups, students would focus their attention on them, potentially increasing participation rates.
  - **Low-level calendar implementation.** The calendar was originally designed using Microsoft Excel, rather than built-in web functions. On the website, a link that enables the downloading of the excel calendar document is displayed, followed by an outdated image of the October month, as shown in **Figure 1**. Such design strategy impedes user-interaction and hinders system maintenance duties. On the one hand, users cannot display the events from other months within the website. Users who do not regularly check for updates in the excel spreadsheet miss-out on important information. On the other, administrators have not updated the calendar since the start of the academic year, suggesting that maintenance is tedious. In turn, event updates seldom occur, aggravating the user miss-information problem.

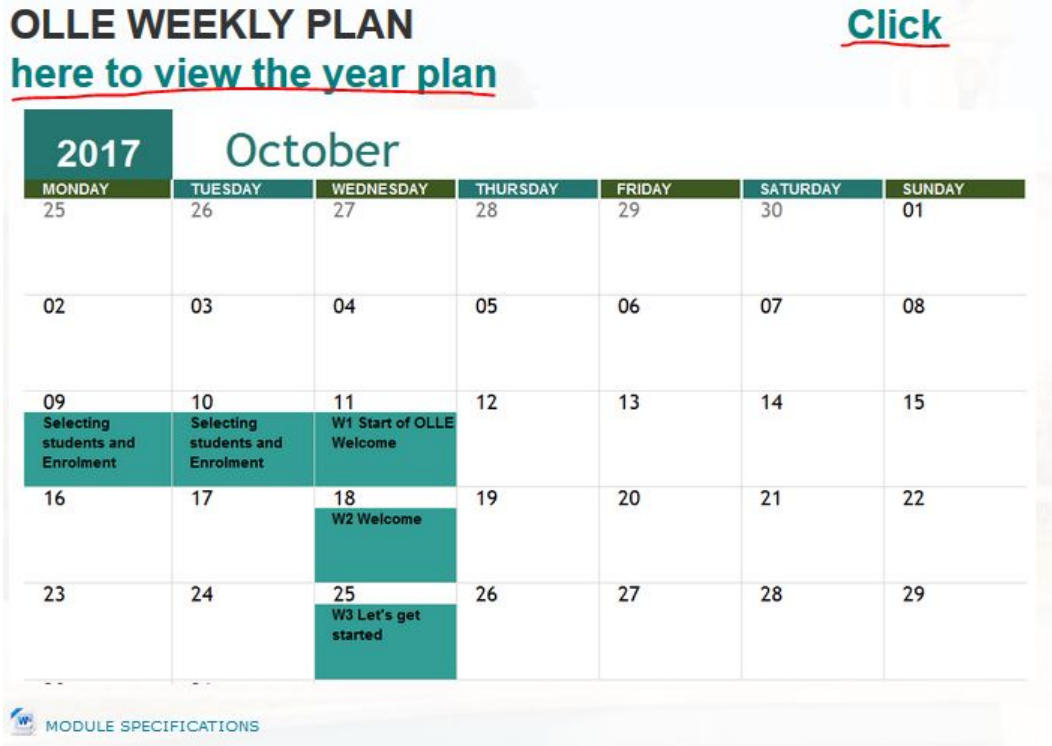


Figure 1: Current Calendar System from *Moodle* as per April 1, 2018

## 2.3 Proposed solution

Our proposed solution integrates the three functions — forum, chat, and calendar — into a single platform, as well as including user authentication to verify users and distinguish one user type from the others. Such integration will enable NAA users to keep track of new information with ease. Given that Android and iOS mobile operating systems have a combined market share of 95% as per February 2018 [6], implementing a hybrid application is the best solution.

The app will have three types of users, including administrators/stakeholders, NAA, and external users. By getting users to select the language they aim to learn and assigning them into their corresponding language chat group, any sort of confusion that arose from having multiple Wechat groups should disappear. Fewer confused users would result in stakeholders receiving less inquiries via email, reducing the workload. Stakeholders will also be able to redefine the purpose of the app should they use an in-app chat-group rather than Wechat groups. Lastly, the calendar user interface and functionality will be redesigned, ensuring that the functions flow smoothly and can be navigated with ease.



---

## 3 Research

This chapter provides background information concerning the project, assessing the effectiveness of language learning methodologies in mobile platforms and the existing cross-platform development solutions. It also presents a market analysis, including two language learning and mobile transaction processing applications.

### 3.1 From Mechanical Repetition to Interactive Learning

Technologies, mobile or otherwise, are instrumental in language instruction. They are not themselves language instructors, but rather instructional tools. Therefore, learning languages through your mobile appears to be less effective than traditional face-to-face methods. Reasons for this include the fact that apps tend to teach vocabulary in isolated units rather than in relevant contexts; apps minimally adapt to suit the skill sets and learning goals of individual learners; and apps rarely offer explanatory corrective feedback to learners [7, 8]. This is due to human beings differing in terms of language learning skills, strategies and motivations, and the current technology’s inability to detect and provide language learning environments that best adapt to each individual learner’s needs [9]. In other words, there is no technology that can dynamically adopt the instructional features that best suit the learning environment of each potential user, making mobile platforms inefficient learning tools compared with traditional face-to-face methods.

On the other end lie those apps that use social, language learning tools. Connecting a language learner with an instructor enables for the co-creation of a context that enhances learning, supported and developed by the interaction between both individuals [10, 11, 12]. The exposure to a more realistic context fosters the retention of recently acquired vocabulary [9], and boosts the confidence levels of the interacting individuals, particularly, when using the new language [13]. An example is *HelloTalk* [14], a language learning app that connects instructors with users, encourages them to chat with others, and provides them with tools to enhance the language skills of their peers.

Combining interactive features in an instructional tool poses several advantages. The features of shared tasks and real-time interaction not only enhance the efficiency of group learning, but also improve the quality of interaction during language learning [15, 16]. In addition, interaction between users enables them to shape their own context given the boundaries of the task, co-creating the language learning environment that best suits them [17]. Hence, in light of the stakeholder’s view and the associated benefits, our app combines these two methodologies.

---

## 3.2 Development tools for Language Learning Applications

The increasing fragmentation of mobile devices in terms of operating systems, models and target users has created the problem of supporting all the possible mobile platforms to reach the highest number of potential users. Considering a paid language learning application, a wider customer-base means higher earnings. In this part we analyze the pros and cons of three current cross-platform development approaches, namely *native*, *web* and *hybrid* applications [18, 19, 20].

*Web apps* are browser-based applications in which the software is downloaded from the web. On the one hand, the app is developed only once and distributed to multiple platforms via the browser. They do not require installation nor subsequent upgrades, and cannot be downloaded from the app store. This means that they do not occupy space in the mobile storage, but also cannot be used when there is no internet connection. On the other, web apps have limited access to the device's hardware and native features, hindering the user experience in mobile platforms [18]. Extra costs and time are incurred to download the web page from the internet and render it on the screen.

*Native apps* are developed using a platform specific Integrated Development Environment (IDE). That is, source code for one operating system is not reusable in others. Since all the current mobile operating systems do not share any IDE or programming language, supporting all the available operating systems can increase development costs, in terms of required time and programming skills [19]. Nevertheless, they provide the richest user experience. Performance of the app is fast, has a consistent look, and has full access to the platform's hardware features. Native apps are downloaded from the app store, and can be executed without internet connection.

*Hybrid apps* combine the advantages of web and native apps. The application code is developed only once, using a framework specific language e.g. Javascript, and is then deployed to all operating systems supported by the framework [20]. They are installed on the device, and can thereby be downloaded from any app store, as well as access the device's hardware through specialized Application Programming Interface<sup>10</sup> (API). Nevertheless, hybrid applications have higher energy consumption, and can have different performances depending on the platform where the application is deployed [19].

---

<sup>10</sup>API: set of routines, each of which executes certain lines of code.

---

### 3.3 Market Analysis

**Duolingo's** [21] and **HelloTalk's** characteristics are analyzed. The app will be commercialized during the second phase of the project. Thus, transaction processing tools are also presented.

#### 3.3.1 Duolingo

*Duolingo* uses mainstream teaching tools in the industry. It divides language into sections of increasing difficulty. Each section has a set of exercises to complete. Exercises are multiple-choice questions composed of images, creating a user-friendly experience. New sections can be learned upon completing the previous ones. In addition, it sends daily notifications to the user's phone, and emails them if they fall behind in their lessons. Duolingo also has a social feature, enabling users to create/join clubs of people learning the same language. There they can post their latest progress and make friends.

#### 3.3.2 HelloTalk

*HelloTalk* differs significantly from the mainstream language learning applications. *HelloTalk* does not provide teaching resources for its users. Instead, it connects a learner user with one that already knows the language. Then, it supports the interaction of the connected users through a chat system, which includes tools to correct grammar mistakes from other users.

#### 3.3.3 Payment System

*WeChat Pay* [22] and *Alipay* [23] are payment applications predominantly used in China. Outside China, *Paypal* [24] is the most popular option, accounting for 74% of the payments market share [25]. The apps synchronize the user's bank account details, processing cashless transactions. In addition, *Paypal* can process transactions in multiple currencies. Payments can be received in Renmibi<sup>11</sup> currency despite being paid in other currencies. *WeChat*, *Alipay* and *Paypal* provide an online user-manual to implement their services, containing APIs and a step-by-step procedure.

---

<sup>11</sup>Renmibi: Chinese currency.

---

## 4 Software Requirements Engineering

Software Requirements Engineering is the process of designing and agreeing upon system rules with the stakeholders [26]. The project was supported by three different stakeholders, each of whom had different requirements and expressed them in different ways. Our priority has been to get them to agree on their requirements.

This chapter highlights the software requirements engineering process, identifying the stakeholders' profiles, the requirements elicitation strategy, and the changes made by stakeholders to the requirements document once they had been defined.

### 4.1 Stakeholder Profiles

**Vis-a-Vis (VAV).** The stakeholder group that has shown the greatest interest in the development of the project. They provided most input regarding aesthetics and functionality, and were eager to meet frequently. Meetings were held every two weeks during the development phase to monitor progress, providing valuable feedback. VAV always took into consideration our input to the project, suggesting requirements within the boundaries of our coding capabilities.

**Language Center.** It is the most important stakeholder, yet the one who committed less time to the project. She got promoted to Director of the Language Center (DoLC). Managing the OLLE project was not her original idea, but as DoLC, it became her responsibility. There are, presumably, many new duties that come with being DoLC, and most are more important than the project. In turn, she delegated requirements' decisions to other stakeholders, did not reply to emails and had little time to meet with us. The funding required to purchase the infrastructure of the app was provided by the Language Center.

**Maria.** This stakeholder took a side-role in the requirements gathering phase. She provided ideas for the project, but we did not consider her requirements as they conflicted with those proposed by other stakeholders. These requirements were oriented towards later stages of the project, addressing the needs of external users. Maria refused to bargain for requirements that better fitted our skills and the time constraints of the project. Moreover, living abroad in a country seven hours behind Chinese local time did not facilitate the arrangement of meetings.

---

## 4.2 Requirements Elicitation Procedure

To better conceptualize the stakeholders’ ideas, a one-to-one meeting was held with each stakeholder. They strongly differed on the core purpose of the app. While VAV and the LC were in favor of a ‘peer-supporting’ platform that enabled students to share language learning strategies with others, Maria preferred a platform that provided language learning media tools to students. At this stage, the Language Center imposed their authority, claiming to be the main stakeholder and, thereby, having the final decision concerning how should the team proceed. Afterwards, a fourth interview was organized with both VAV and the LC. Grouping them together was particularly effective, drawing out a set of requirements to which both stakeholders agreed.

Throughout the process, complicated decisions were backed-up by the provision of documents/user testing. For instance, stakeholders were arguing among themselves about the platform the app should be developed for, on the assumptions that the app had to evoke native feeling<sup>12</sup> and that there was not enough time to develop a native app for both Android and iOS platforms. A survey asking students for the mobile platform they use was carried out. Further details are given in **Appendix B**. Given the results in **Figure 2**, a *hybrid* app is the best possible solution.

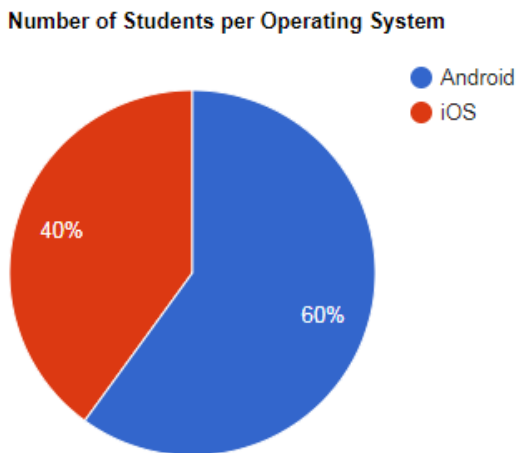


Figure 2: Survey results

Similarly, irrelevant requirements were screened out using Leal et al.’s *Devil’s Advocate* approach [27]. During the meetings, upon the selection of a new requirement, one member would ask a question that leads others to argue against the usefulness of the given requirement. Should they follow the lead and argue against it, the requirement was classified as irrelevant. On the other side, should they come up with other

---

<sup>12</sup> “The app must feel like any other mobile app, rather than a web app” – VAV

---

arguments that supported the need for the requirement, the requirement was classified as valid. Having more arguments for/against particular requirements better prepared the team for stakeholder meetings, enhancing our ability to influence the stakeholders' decisions.

### **4.3 Requirements Modification**

The stakeholders of our project, on the 10 March 2018, modified functional requirement 3, which states that 'The app must autonomously process the validation and verification of NAA user accounts'. The new requirement states 'All NAA accounts will be created by administrator account users'. VAV argued that a language fair will be taking place at the start of the next academic year, gathering the required user details to create the accounts.

---

## 5 Software Design

This chapter presents the architectural and implementation design decisions of the app, underlining the programming languages and algorithms used, as well as an explanation of the user interface. It also describes the testing methodologies used.

### 5.1 Architectural Design Decisions

The *Architectural Design* explains how the system is organized. Somerville claims that *compactness*, *performance*, *availability*, *security* and *maintainability* are the key factors to consider when making design decisions [26]. *Compactness* is the extent to which the functions/tools of the product fit in optimally with each other, and how well they fulfill the purpose they were designed for. *Performance* consists of the set of criteria that shape the behaviour of the system, including run-time and functional goals. *Availability* considers the frequency of development of new tools for the system, as well as their ease of implementation. *Maintainability* refers to the methodology employed to maximize the lifecycle of the product. *Security* examines how the critical data is stored, transferred and protected from external attacks. Based on this, we explain the key design decisions of the system:

#### 5.1.1 Ionic Framework

The *Ionic Framework* is an open-source, bundle of tools (*Cordova* [28], *Angular* [29] & *Ionic*) that allow for the development of *hybrid* mobile applications [30]. Ionic apps are built using web development technologies such as *Angular* (*HTML5*, *SCSS* & *Typescript*). The web code is then wrapped using *Cordova* native containers, enabling the deployment of the app in multiple-platforms (*Android*, *iOS*, *Windows Phone* & *Web browser*) with a single code base. In addition to this, Cordova plugins provide access to native device features, replicating development features from proprietary languages including iOS and Android.

The *Ionic Framework* also provides Ionic-specific development tools, which are appropriately integrated with Angular web development technologies, and effectively enhance the developer's coding capability. For example, an extensive array of libraries<sup>13</sup> are at our disposal, provided by creators of the framework. Compared with similar cross-platform mobile development tool Framework7 [31], Ionic is supported by a large community of developers, having lots of user-created content, including libraries, tutorials and sample projects [32]. In other words, the Ionic Framework is compact, plenty of resources are available, and optimally aligns with the performance expectations of the stakeholders.

---

<sup>13</sup>library: a set of pre-compiled routines that a program can use.

---

### 5.1.2 Angular

*Angular* is a JavaScript Framework that allows for the development of reactive Single-Page-Applications (SPAs). These contain a single *Hypertext-Markup-Language (HTML)* file with *Javascript* code that loads content when the app is executed and displays it dynamically<sup>14</sup> [33]. As a result, the app response times are faster and provides a more reactive user experience, compared to ‘on-click’ loading of new pages in traditional server-client web development communication. Faster rendering-times result in better *performance*.

Another advantage of Angular is its project structure. The app composed of several angular *components*. These components contain the code of a specific page URL/function. Therefore, errors in a component only affect the given component rather than the entire project. Errors are thereby isolated and detected, facilitating *maintenance* duties.

### 5.1.3 XAMPP & LAMP

*XAMPP* [34] is an open source cross-platform web server package consisting of *Apache HTTP Server* [35], *MySQL* [36] database and *Hypertext Preprocessor (PHP)* [37]. This tool is particularly useful to set up and manage a local web server. A live server was only rented upon the completion of the app. The reason is that outsourced servers are scalable, having no need to migrate to new servers in case the user-base grew exponentially. The final software bundle used, thus, is LAMP [38]. An explanation of the key advantages of each of the tools in LAMP is provided:

- **Linux Operating System (L).** Compared to Windows, Linux servers often run for years without failure nor requiring hardware updates, granting stability and long-lasting server life-cycle [39]. Linux servers’ *security* levels are high, given the way account privileges are assigned. Users do not have administrator privileges, and thus have limited access within the system [40, 41]. In contrast, Windows does not distinguish between administrator and guest accounts, enabling both to freely access data. These characteristics support our decision of taking *CentOS* [42] Linux server.
- **Apache HTTP Server (A).** Web servers transfer the data from the website’s host to the browser in the computer or the mobile device. *Apache* is open source, and the source code can be modified to suit individual needs [43].
- **MySQL (M).** It is an open-source relational database management system (RDBMS) [36]. Data can be inserted, read, updated or deleted from/in tables, whose attributes are represented by columns.

---

<sup>14</sup>display content dynamically: the app shows certain features, such as buttons or events when needed



---

Attribute data types are predefined and relationships can be created between them, allowing for the linking of multiple tables.

Desired functionality played a significant role in the selection of the database model. MySQL can securely and efficiently process transaction payments [44, 45], a core function in the next phase of the project. The reason for this lies in MySQL capability to keep the integrity of the transaction, revoking the transaction if it fails to pass the transaction rules.

- **PHP (P).** PHP is a server-side programming language designed for web development. The team had previous experience working with PHP. Concepts and best practices were already known at the start of the project.

#### 5.1.4 RESTful PHP API

The *Representational State Transfer (REST)* Application Programming Interface<sup>15</sup> (API) provides a coherent service structure to develop the server side code. It allows to keep all routines within one single file. An incoming HTTP request contains a unique addressable Uniform Resource Identifier<sup>16</sup> (URI) that, in turn, invokes a specific routine within the file. In simpler terms, the RESTful API contains the universal set of ‘actions’, yet only one ‘action’ is executed when a new ‘instruction’ is received. This way, back-end errors can be easily detected and corrected upon the execution of the method, considerably easing maintenance duties.

REST structure is effective as an API because it is stateless [46]. That is, it does not recall previous calls in a given sequence of interactions with the user, having a high performance. It allows the server to quickly release computational resources, and the implementation is further simplified because the server does not have to deal with ‘conversational’ resource usage across requests [47]. This, and its resemblance and familiarity of the team with Object Oriented Programming<sup>17</sup> (OOP), supports the team’s design decision of using RESTful API.

---

<sup>15</sup>API: set of routines, each of which executes certain lines of code.

<sup>16</sup>URI: a string of characters used to identify a resource.

<sup>17</sup>OOP: a programming paradigm based on the concept of ‘objects’, which may contain data, in the form of fields, often known as attributes; and code, often known as methods.

---

## 5.2 Implementation Design Decisions

### 5.2.1 Project Structure & Naming Convention

Group coding can be an arduous task if there is a lack of convergence regarding the naming of files. Errors can arise every time a member's project folder is imported by other teammates. The team followed a strict file naming convention. The project was divided in four sections, the chat, forum, calendar and authentication, each of which consisted of a set of components. The naming convention was:

[section\_name]-[component\_name]

e.g. [chat]-[new-form]

Regarding the filing system, all the components representing a screen-page are filed within the 'src/pages' folder. Generic methods, such as http request to establish client-server communications, are filed within the 'src/providers' folder. External components loaded within a screen-page file are stored in 'src/components'. Generic styling variables are stored in 'src/theme/variables.scss', albeit almost all page-components have their own '.scss' styling files. Lastly, all components are loaded in 'src/app/app.module.ts'.

### 5.2.2 User Authentication & Validation

Authentication is the process of verifying the identity of users and authorization is the process of verifying that a given user has the right to perform some action [48]. There are two user groups: stakeholders and students. The app must be able to identify the user account type, and provide functional rights accordingly.

A *Role Based Authentication System (RBAS)* was used to identify users [49]. This system assigns roles to specific users, each of which has different access rights. Higher roles includes access rights of lower roles. There are up to four roles: *registered*, *validated*, *admin* and *master*. *Registered* role is an intermediate state in which user credentials have been stored in the database but the user has still got to validate the account by inputting the validation code sent to their email account. Upon completion, the role switches to *validated*, and has read rights<sup>18</sup>. This step ensures that both the user and the email exist. *Admin* and *master* accounts have read-write rights<sup>19</sup>, although master has super-admin rights, enabling to delete all the data stored in the database.

This super-user functionality was achieved maximizing angular's advantage — dynamic rendering of elements. The same template component is used by all roles, but renders extra elements for *admin* and *master*

---

<sup>18</sup>read right: status that only allows the user to see the content displayed, but prevents them from creating, modifying or deleting content.

<sup>19</sup>read-write rights: status that allows the user to see, create, edit and delete content

roles. This method is more efficient than having similar multiple-components because less components are loaded when the app is executed.

We added security layer to the system by including JSON Web Token (JWT). JWT is a JSON-based security token encoding that enables identity and security information to be shared across security domains [50]. Upon login, the user identifier (userID) is encrypted in the server to generate a unique JWT, and is then sent back to the client, as shown in **step 1** from Figure 3. Thereafter, any time the user requires resources from the server, both the user identifier and the token are sent as parameters, shown in **step 2**. Again, the server encodes the user identifier and compares it with the received token, validating the operation if both tokens are the same.

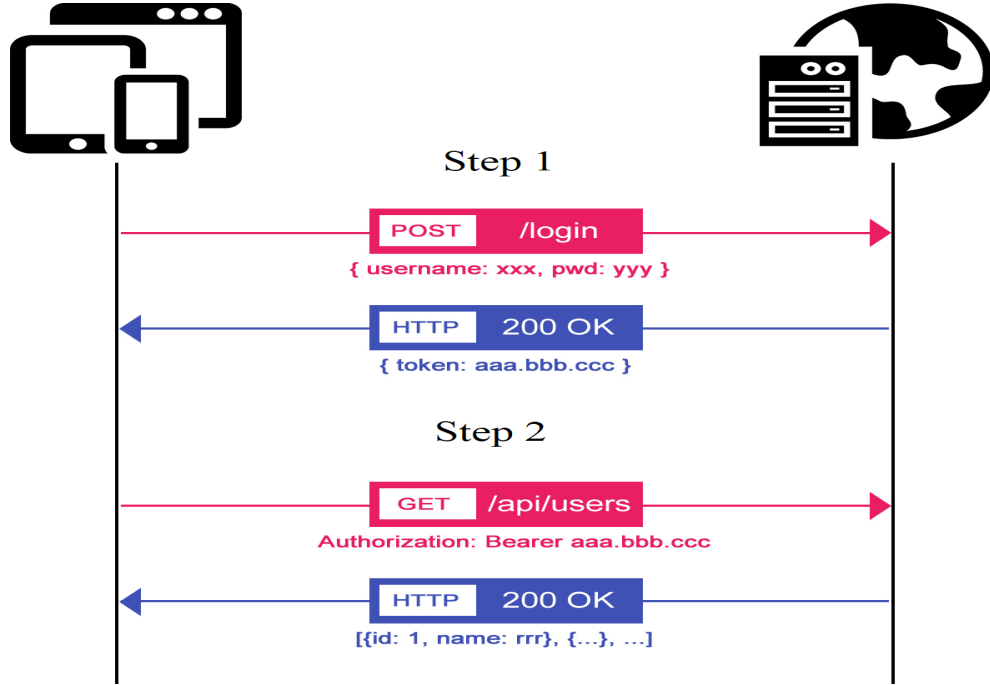


Figure 3: JSON Web Token operation. Source: [51]

Passwords must be encrypted to prevent other individuals from accessing an existing account. A hash function<sup>20</sup>, *SHA-256* [52], has been used for one-way encryption. This means that the algorithm encodes the password, but it cannot be decoded back by applying the same algorithm. Password verification is then carried out by ensuring that the encoded inputted password matches with that one stored in the database.

<sup>20</sup>hash function: cryptographic primitive widely used to provide services of data integrity and authentication issues.

---

### 5.2.3 Real-time Chat System

Instant messaging chats have real-time functionality. That is, users receive incoming messages from others without having to refresh the chat page. A *polling* [53, 54] technique has been used. In *polling*, the receiver periodically checks for new message updates in the server. Each request queries whether new messages have been stored any time after the most recent message displayed on the screen was sent. This process is illustrated in **Figure 4**.

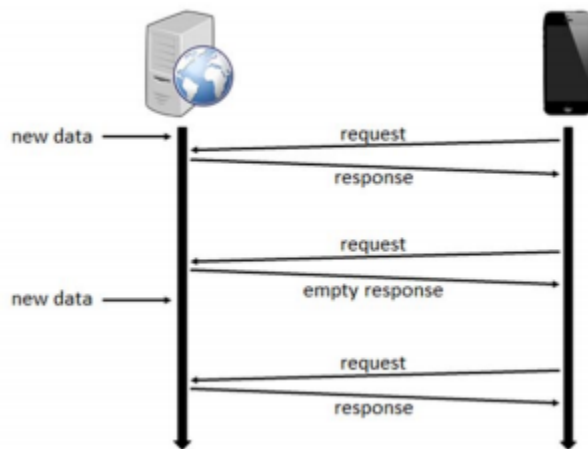


Figure 4: Client Polling sequence diagram. Source: [54]

We acknowledge this solution as purely functional yet highly inefficient. Despite the ease of implementation, its characteristics of perpetually interacting with the data source tends to (i) create a lot of header data that could adversely affect a mobile operator's network, and (ii) significantly deteriorate the battery life between charges of mobile devices [53]. We propose an alternative, more efficient solution to the problem in **Chapter 10**. Technical complexity and time constraints prevented the team from implementing such solution.

### 5.2.4 Posting in the Forum

The hurdle that needed to be overcome in the forum is to identify whether a given user post is original or a reply to another post. The former case answers directly to the forum topic and creates a new entry for which replies can be made. The latter, consist of the replies made to the new entry or perhaps to an already existing reply to the entry. This problem was solved by using a *Tree* data structure [55], and improved using *Union-Find* algorithm [55, 56].

---

A tree is an abstract data type (ADT)<sup>21</sup> that simulates a hierarchical tree structure, with a root value and subtrees of children with a parent node, represented as a set of linked nodes [57]. The Union-Find algorithm works for a set of size 'n' where each element has its own value. Then, the union of these elements is represented as a tree where one points to the other [55], having a parent and a child node. The parent node, under the union of a new element to the parent node, would have two child nodes. If the element is added to the child node, the child node would have its own child, being this the added element. This way, the Union-Find algorithm creates associations between nodes and shapes the tree structure.

In our project, each post in the database has a unique identifier called postID. A given post also has a parentID, pointing towards the post it is replying. Should the postID and parentID be the same, the post can be categorized as original. Otherwise, it is a reply of the post whose postID equals the reply post's parentID. This way, when the client requests the reply posts of a given post, the replies can be obtained by searching those posts whose parentIDs match the given post's postID, being the result the child nodes of the current post.

### 5.2.5 Calendar Library & Date Formatting

A library displays the calendar on the screen, loading the events that are already stored in the database. To do so, events are ordered in chronological order, and the time format is modified to match that one required by the library.

---

<sup>21</sup>ADT: a logical description or a specification of components of the data and the operations that are allowed, that is independent of the implementation.

---

## 5.3 Testing

Three types of testing methodologies have been used to test our app, including *End-to-End (E2E)* [58], *Metamorphic* [59] and *User Acceptance testing* [60]. Further information of the test conditions and results is shown in **Appendix D**.

### 5.3.1 End-to-End

*End-to-End (E2E)* testing was used for this purpose, analyzing whether the components of the application interact with each other as originally designed, and ensuring that the right information is transferred accordingly. Tests were undertaken using Jasmine [61], a framework to test Javascript code, and Protractor [62], a built-in AngularJS testing API. Due to the wide array of components in the system, a multi-stage testing procedure has been followed. Several sub-systems, each of which consists of a component and those with which it interacts, have been individually tested. Upon completion, a test evaluated the app from start to finish, testing this way the entire system.

### 5.3.2 Metamorphic Testing

*Metamorphic testing (MT)* was used to detect errors that were not detected by E2E tests. In E2E testing, the tester must establish a predefined desired outcome that is then compared with that one produced by the test case execution, passing the test if the two outcomes are the same. Such tests are subject to the *oracle problem*: situations where the tester cannot decide whether the outcomes of the test case executions are correct [63]. Unlike E2E testing, MT provides a solution to the oracle problem, checking the program against two or more selected *metamorphic relations*<sup>22</sup> (MR) rather than focusing on the correctness of each individual output [64]. For example, administrator users must be able to perform all the actions that a registered user can perform. If this property is not met after logging-in with both accounts, an error is identified. Thus, MT enables to find errors that we thought never existed, while having to develop no code for it, saving us time.

### 5.3.3 User Acceptance Testing

*User Acceptance Testing* [65] is the process in which a software application is tested by users. Human beings differ from each other, performing certain actions and procedures when using software that leads to finding errors that we would not have found otherwise. On top of that, we used a *System Usability Scale (SUS)* to

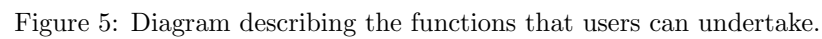
---

<sup>22</sup>Metamorphic relationship: the existing associations between the known properties of a problem domain.

---

obtain feedback from users. SUS is a questionnaire where users can quantify their experience with the app. During the acceptance testing, users were given the app without explanation of its functions, encouraging them to discover them by themselves, and comment on the unappealable features of the app. Based on these results, improvements were made to the app. Further details of the testing is provided in **Appendix D**.

This chapter presents the user interface, highlighting how users can interact with the app to perform the desired actions. The set of actions that users can undertake is shown in **Figure 5**.





## 6.1 Registration

To register, users must create an account, filling-up their details. On completion, they must input a validation code sent to their email. Upon validation or login, users will access the forum.

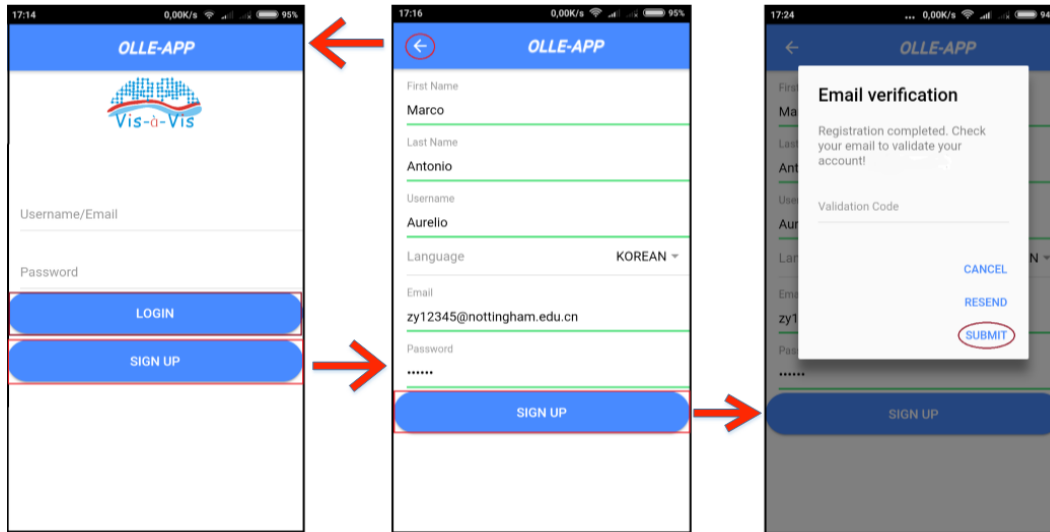


Figure 6: From left to right — login, register and validation pages.

## 6.2 Forum

Weekly topics are displayed in the forum page. Administrators can create new topics. All users can access a given topic, enabling them to view, or reply to, comments within the topic.

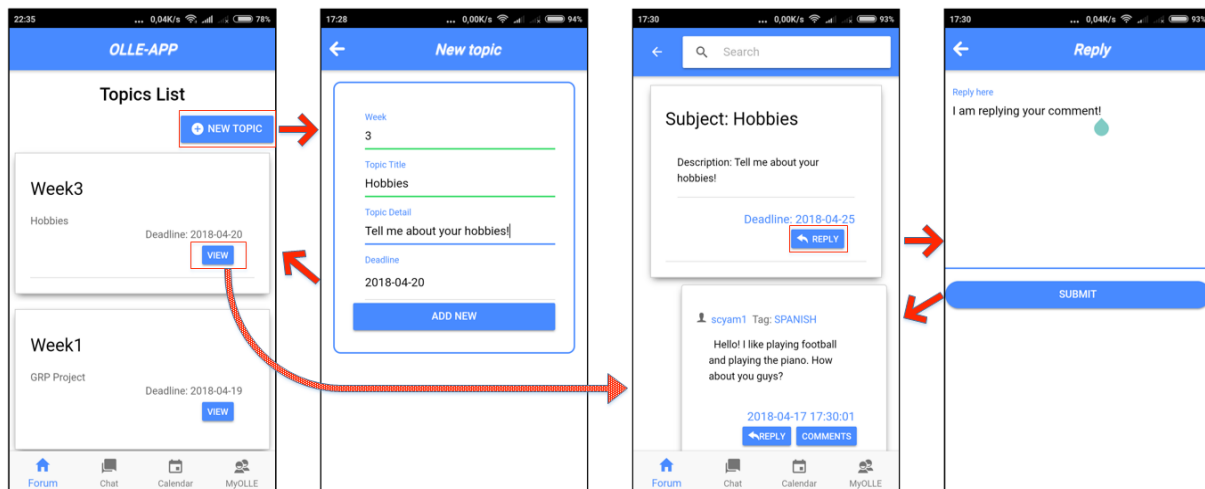


Figure 7: Forum pages.

## 6.3 Chat

The set of chat rooms is listed in the chat page. New chats can be created, edited or deleted by administrators. All users can join any chat room at any point in time and send messages. Your own messages are displayed at the right side of the screen, whereas others' messages are located at the left side.

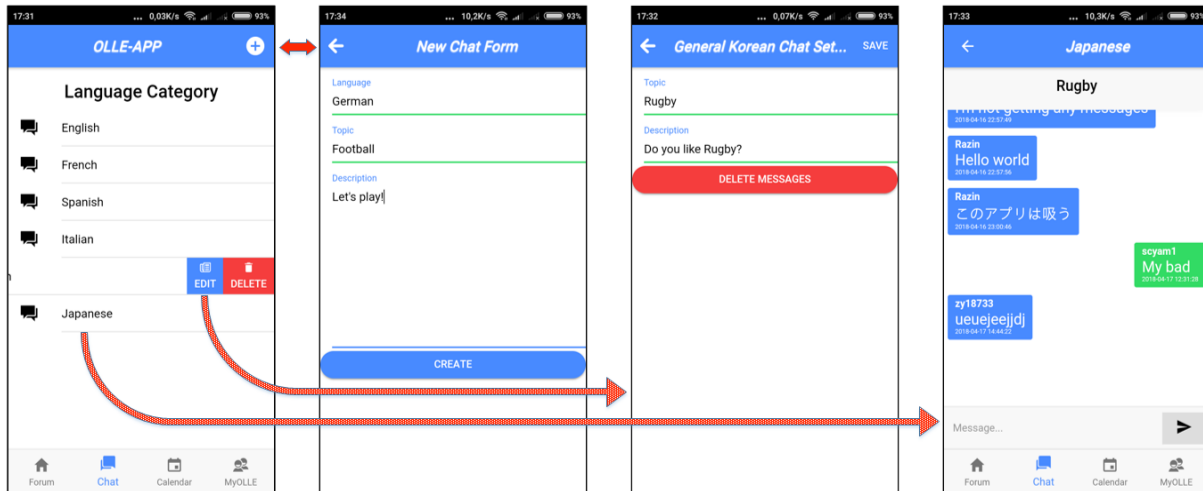


Figure 8: Chat pages.

## 6.4 Calendar

Calendar events are shown in the date in which they take place. Administrators can create, edit or delete a given event. NAA users can only view events. Upon selecting a date, the list of events taking place on that date are displayed at the bottom. If selected, more information about the event is provided.

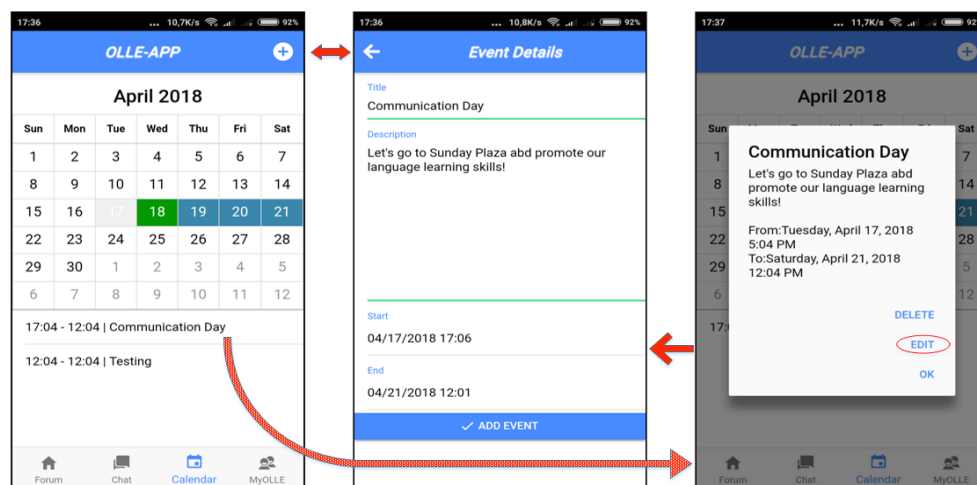


Figure 9: Calendar pages.

---

## 6.5 Account

User details are displayed in the account page. Users can edit their name, surname, username and password. They also logout in the account page.

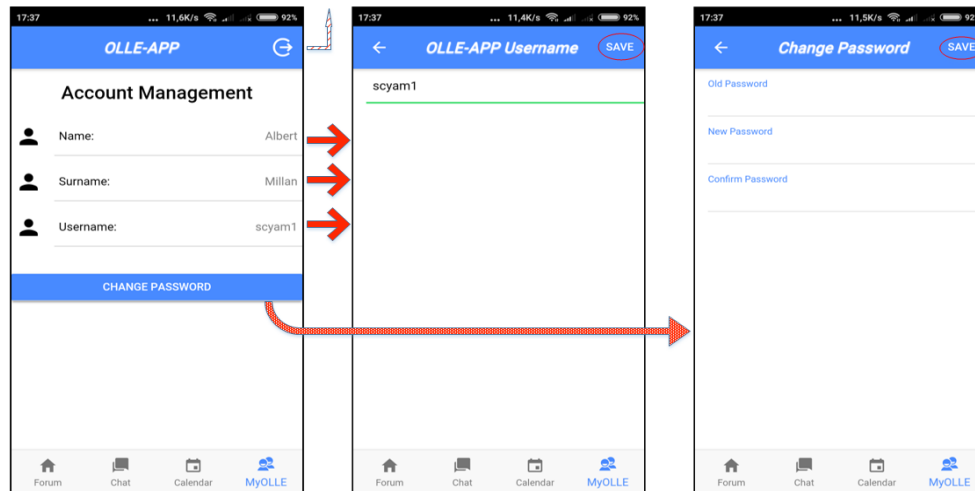


Figure 10: Account pages.

---

## 7 Supporting Software Development Tools

The software management and file sharing tools used have been key drivers to our accomplishments. This chapter explains these tools, emphasizing how they enabled us to achieve better results.

### 7.1 Common File Management Tool: Github

*GitHub* [66] is a code hosting platform for open-source and private software projects. The primary features are version control and ease of distributed development. *GitHub's* version control keeps track of file changes, comparing the content of files and the time they were uploaded into the common repository<sup>23</sup>. The developer can always retrieve a previously uploaded version of the code. It is itself a back-up in case there is a crash or the code is lost.

Github uses *Git* [67] to manage files. Git synchronizes all files in Github's online repositories with your local computer files. This way, the process of uploading, downloading and, particularly, deleting files can be rapidly undertaken. We acknowledge that the team has not effectively exploited this tool. More information is provided in the **Chapter 9**.

### 7.2 Task Management Tool: Teambition

*Teambition* [68] was used to manage tasks and meetings, as well as being the team's communication resource. The interface is user-friendly, providing tools to create and assign tasks, as well as deadlines to specific teammembers. Comments can be provided on specific tasks, facilitating the provision of feedback to other teammates. Furthermore, meetings can be arranged, specifying a time and location. It also includes a chat room for the team members. The team set a rule in which all project related conversations should be undertaken in this chat. All team management duties have been carried out via *Teambition*.

### 7.3 Agile Methodology

We have taken an *Agile* [69] approach for the development of the project. Such method is characterized for the realization of incremental solutions to software specification, development and delivery. The agile methodology is defined by a set of principles, specified in the *Agile Manifesto* [70]. The core principles include the prioritization of people over software processes, emphasizing face-to-face communication and the creation of an environment that motivates individuals [71]; working software over comprehensive documentation;

---

<sup>23</sup>repository: place where files are stored.

---

customer collaboration over contract negotiation, in particular, to jointly determine system features across each development cycle [72]; and responding to change over following a plan, as close interaction with stakeholders makes requirements more volatile [73].

---

## 8 Requirements Evaluation

In this chapter, we analyze our accomplishments based on the system requirements, shown in Appendix B. Only those requirements the team was unable to finish are shown, highlighting the reasons that impeded their completion. Completed requirements are assessed in **Chapter 10**.

### 8.1 Pending Requirements

**Requirement 2.** The app functions in both Android & iOS operating systems, yet it cannot be downloaded from their corresponding app stores. We struggled to purchase the developer licenses required to release the app into the market. Many documents from the organization (Nottingham University) were required for the purchase of the license. We asked stakeholders to facilitate them, but there was no time left by the time they were provided to us.

**Requirement 3.** The team was unable to meet such demand on time. At the time it was modified, a high-level implementation of the user authentication system had already been developed, and other sections such as the forum were lagging behind.

**Requirements 4, 6, 8 & 18.** These requirements were not fulfilled due to time constraints.

**Requirements 9 & 14.** Technical difficulties and lack of experience impeded the team from completing these requirements. We underestimated the complexity of the procedure required to implement push notifications/reminders. Setting up Android and iOS certificates<sup>24</sup> is necessary [74], yet we were unaware of this until the end of the project.

**Requirements 20–26.** These correspond to the second phase of the project to be completed next year. We included them in the report to provide a reference of what needs to be done to the team that takes over the project the coming year, although we did not commit to completing them this year.

---

<sup>24</sup>Certificates: allows the developer to access specific native features of a given operating system.

---

## 9 Reflection

This chapter highlights both technical and team management aspects of the project, underlining the struggles and accomplishments of the team.

### 9.1 Authentic Software Engineering Experience

The project allowed the team to gain first-hand experience regarding real-world, software engineering project development. The team members had no previous experience working in a project of such scale, forcing us to face unprecedented situations and to explore unfamiliar solutions to problems. Dealing with stakeholders, group coding a sizeable solution or understanding team dynamics, are the acquired skills that cannot be taught by a lecturer, but rather learned by doing. Learning and appreciating these skills throughout the development of the project was a key driver towards its success, keeping the team motivated, and overwhelmingly satisfied upon the completion of the project.

Another aspect of the authenticity of the experience was provided forming, consolidating and managing the team throughout the development process. Team members perceive that trust and commitment was built and sustained by overcoming Tuckman's team evolution phases: *forming*, *storming*, *norming* and *performing* [75]. During the *forming* phase, team members got to know each other and conflict was avoided. Meetings were peaceful and easygoing. During *storming* phase, team members started voicing out their opinions and conflict arose as a result this. Discussions in the meetings got tense and uncertainty remained regarding the procedure the team should follow. *Norming* and *performing* followed. We were under time pressure, being behind the time plan we had set at the start of the project. We realized that members needed to cooperate with each other to accomplish common goals. Trust built between team members, considerably increasing the output in a minimal amount of time. Despite the challenges, conflict was necessary to enhance cooperation and commitment towards the accomplishment of shared objectives, playing a key role in the success of the project.

### 9.2 Lack of Experience

The team was set back in terms of team management, task assignation, and planning, due to lack of previous experience. This was particularly evident during early phases of the project, where team management decisions were improvable.

---

### 9.2.1 Appropriate Leadership Style

The leader followed a decentralized leadership style, yet it was not appropriate for the context the team was settled-in. Decentralized leadership style characterizes for the delegation and transfer of decision making power to the members of the team. Although it is particularly effective to foster positive relationships between teammembers, it can be counterproductive if they lack information at the time of making decisions [76]. The leader, who had access to everyone's output and thus knew about the strengths of each member, was in a better position to make reasoned decisions when conflict arose. However, decentralization enabled for voting, in turn, selecting the first option, which resulted in delays (having to correct low quality output) and stress due to time-pressure.

We believe that such a stressful and conflicting atmosphere could have been avoided should the leader had imposed a centralized leadership style<sup>25</sup>, encouraging the team members to forgo their their beliefs in light of a potentially beneficial future outcome.

### 9.2.2 Inefficient Task Management

The scope of the tasks created was often broad, lacking elaboration of potential sub-tasks. Failure to establish specific tasks results in team members becoming confused, pulling apart, and reverting to mediocre performance [77]. For example, initially, the team was divided into three groups, one for each system (calendar, chat and forum). After reviewing progress later on, improvements were minimal considering the effort and time spent in order to achieve them. Having such broad tasks, some teammembers were unable to understand the required design architecture for the system they were developing (e.g. database tables, functional features, etc.). In turn, anxiety built within the team as deadlines were getting closer. Progress was not as originally expected.

Nevertheless, changes in the team's execution of operations enhanced the team's productivity. *Teambition* was vital to organize and distribute tasks across teammbmembers, but also to raise awareness regarding the importance of simplifying existing tasks into smaller subtasks, as shown in **Figure 11**. Confused teammembers did not have to think about the entire system's architecture, but rather about the functionality of the specific subtask at hand at one time. Output increased significantly as a result, forming a solid architecture when all the subtasks of the system where effectively combined.

---

<sup>25</sup>centralization: concentration of decision making power under a single authority.



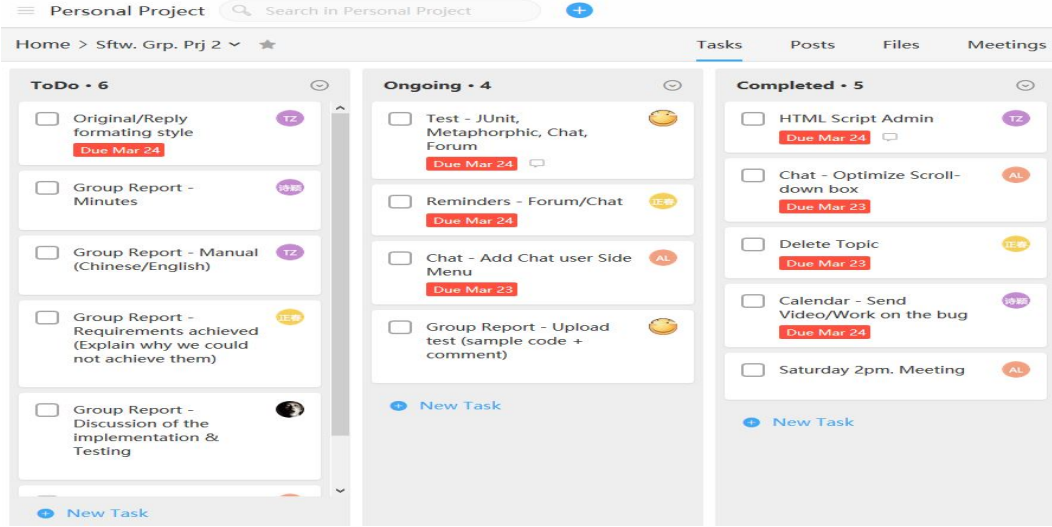


Figure 11: *Teambition's* taskflow sample. Tasks assigned to members and listed in one of the three columns.

### 9.2.3 Misleading Planning

The original plan in our interim report was misleading. The assumptions regarding task completion times were misguided, often underestimating not only the time taken to complete a given task, but also the dates in which the tasks were to be started, as shown in **Figure 12**. Having no valid plan to rely on hindered the task of keeping track of progress and deadlines. Should we be given the chance to go back in time, we would have definitely started coding earlier and would have been stricter in terms of following the task deadlines.

This challenges the agile development approach followed, characterized for the delivery of working software in short time periods [78, 79, 80, 81, 82]. However, agile methodologies also stress the individual ability of the software developers' skills as an enabler to complete projects under an agile approach [83, 84, 85, 86], being these the variety of programming languages mastered and length of experience [72]. We argue that, given the lack of exposure to multiple programming languages and short coding experience, student software projects must initiate the development in earlier stages of the project. The reason for this is to familiarize with the tools and make up for the shortfall of skills.



Figure 12: Self-made Gant chart comparing the expected (black) and actual (orange) project plan workflow.

### 9.3 Cross-cultural Communication Issues

Communication between team members plays an important role to achieve the desired performance goals. Teams tend to perform better when there are no barriers worsening communication between team members [77]. In our case, a language barrier existed. Members had to communicate using English rather than Chinese. English is not the mother tongue of any teammate. Consequently, misunderstandings were frequent and time-consuming. We were conscious about the situation, having members ready to translate to both Chinese and English when necessary. Patience and persistence was the key to solve communication issues, ensuring that every member could follow the discussions in the meetings.

### 9.4 Clearly Defined Roles

A crucial factor towards the team's success was that everyone clearly understood their role in the team. Similar to other student teams, we had to decide on leadership and the team responsibilities each member was to undertake [87, 88]. Two leadership styles were used, although clear and strong leadership yielded better results than the soft approach. Under a clear designated leader, there was more pressure to fulfill the expectation, increasing the commitment and efforts put onto the project. In addition, clear leadership that effectively understands the capabilities/skillset of each of the teammates can more effectively assign tasks that best suit such skills. **Table 1** shows the roles in the team:

---

Name	Role	Description
Albert	Team Leader	Manage & coordinate the team's actions.
	Editor	Ensure quality of written reports.
	Product Owner	Primary contact with stakeholders, assign tasks.
	Leading Developer (1st Semester)	Design the architecture of the app. Lead coding efforts.
Tingting	Leading Developer (2nd Semester)	Modify the architecture of the app. Lead coding efforts.
	Editor assistant	Help editor writing the reports.
	Implementer	Turns ideas into code. Organized work to be done.
Shiying	Researcher	Gathered potential solutions to be used by the team.
	Assistant	Prepared the minutes.
Sen	Q&A Tester	Generate tests.
	Server Specialist	Researched, configured and maintained the server.
	Repository Master	Manage github repository
Wenpeng	Devil's advocate	Question established methodologies seeking for improvement.
	Researcher	Background research and provide alternative solutions.
Zhengchun	Front-end Developer	Style the app.
	Web Master	Maintain and manage project website

Table 1: Team Roles

## 9.5 Inefficient Exploitation of Supporting Tools

The team claims that it was unable to make the most from *Github's Git*. Git considerably facilitates the creation, modification and deletion of files within a local project folder, and eases the synchronization with the online common repository. Failing to use this tool caused team members to focus on their section/task to be developed, creating multiple online repositories (one for every section) rather than implementing changes into a single, common repository. In turn, each member worked from the last version of their repository, that is, their own section.

---

## 10 Conclusion

This chapter summarizes the current state of the project, emphasizing our accomplishments. We also discuss some areas of the project where improvements can be made, aiming to provide a guideline for those computer science students taking on the project next year.

### 10.1 State of the project

During the first phase of the project, we have set robust infrastructural foundations to sustain, and facilitate the development of, the second phase of the project. We have implemented all the core functional components specified in requirement 1, being these the authentication, chat, forum and calendar systems, as well as the majority of system features, including those specified in requirements 5, 7, 10, 11, 12, 13, 15, 16, 17 and 19. Security is optimal. Confidential data is encrypted, and users must overpass the JWT security layer to perform any actions. Lastly, we have rented Tencent’s scalable<sup>26</sup> server<sup>27</sup> [89], which can support increasing volumes of traffic associated with a large user-base. Overall, the infrastructure meets the functional requirements specified by the stakeholders. We thereby encourage next year’s team to continue the project were we left it.

### 10.2 Future Work

While we were able to implement the desired functional behaviour of the authentication, chat, forum, and calendar systems, many opportunities remain for further improvements, particularly the authentication and the chat.

#### 10.2.1 Authentication

User validation codes generated upon registration are not encoded. The program is coded in such a way that even in the case where a user’s validation code was stolen, no thief could impersonate the given user. However, it is good practice to encrypt sensible data before it is transferred across the network [49]. Reason for this ‘secure’ behaviour lies on the second problem. Should a user close the app upon entering the user details required for registry and before completing email validation, it will not be allowed to complete registration successfully. This area needs an urgent solution as any phone crash or accidental closure of the app at the given time would impede users from accessing the app.

---

<sup>26</sup>scalable server: can support increasing volumes of traffic without failure

<sup>27</sup>server: a computer that provides data to other computers. It is where all the data of the app is stored.

---

An optimal solution to the first problem would involve sending a link to the user’s email. The link would be composed of a generic *Uniform Resource Locator*<sup>28</sup> (URL) alongside a unique validation code stored in the database. Upon clicking the link, the server would compare the validation code stored in the database with the one embedded to the URL, validating the account. The second problem can be addressed by storing in the database the time at which the user registered. After a certain period of time, if no validation occurred, the database entry of the user could be automatically deleted, enabling the user to register again.

### 10.2.2 Chat

As we discussed previously, the current system can significantly increase the amount of traffic through the network connection, while simultaneously damaging the battery life of the mobile device. We suggest using a *pushing* [53] technique to recreate real-time functionality. The *pushing* technique has the application’s server deliver notification messages to the receivers upon learning of new updates, avoiding the more costly periodic checks of the *polling* technique in the process. The pushing approach relies on an exchange channel between the application’s server and receivers’ application. Websockets can be used to create a bi-directional communication, so both sides can send data at any time and retrieve it almost instantaneously [90]. The performance improvements are lower latency, less network traffic and less CPU usage in the server and client.

---

<sup>28</sup>URL: a reference to a web resource that specifies its location on a computer network and a mechanism for retrieving it.

---

## References

- [1] <https://www.nottingham.edu.cn/en/language-centre/index.aspx>
- [2] <https://moodle.nottingham.ac.uk/login/index.php>
- [3] <https://web.wechat.com/>
- [4] <https://www.tencent.com/en-us/system.html>
- [5] <https://slack.com/intl/es/about>
- [6] <https://gs.statcounter.com/os-market-share/mobile/worldwide>
- [7] Heil, C.R., Wu, J., Joey, J., and Schmidt, T., (2016). A Review of Mobile Language Learning Applications: Trends, Challenges, and Opportunities. *The EuroCALL Review*, 24(2), pp. 32.
- [8] Morales, R., Igler, B., Bhm, S., and Chitchaipoka, P., (2015). Context-Aware Mobile Language Learning. *Kasetsart Journal of Social Sciences*, 56, pp. 82-87.
- [9] Mohamad Nor, N. and Ab Rashid, R., (2017). A review of theoretical perspectives on language learning and acquisition. *Procedia Computer Science*, pp. 1-7.
- [10] Vygotsky, L.S., (1978). Interaction between learning and development. *Mind in society: The development of higher psychological processes*(pp 79-91). Cambridge, MA: Harvard University Press.
- [11] Kukulska-Hulme, A. (2009). Will mobile learning change language learning? *ReCALL*, 21(2), pp. 157-165.
- [12] Wong, L.H., and Looi, C.K., (2011). What seams do we remove in mobile assisted seamless learning? A critical review of the literature. *Computers & Education*, 57(4), pp. 2364-2381.
- [13] Ab Rashid, R., Mohamed, S.B., Rahman, M.F., and Shamsuddin, S.N.W. (2017). Developing speaking skills using virtual speaking buddy. *International Journal of Emerging Technologies in Learning*, 12(5), pp. 195-201.
- [14] <https://hellotalk.com/>
- [15] Lan, Y.J, Sung, Y.T., and Chang, K.E., (2007). A mobile-device-supported peer-assisted learning system for collaborative early EFL reading. *Language Learning & Technology*, 11(3), pp. 130-151.

- 
- [16] Zurita, G., and Nussbaum, M., (2004). Computer supported collaborative learning using wirelessly interconnected handheld computers. *Computers & Education*, 42(3), pp. 289-314.
- [17] Kiernan, P.J., and Aizawa, K.. (2004). Cell phones in task based learning: Are cell phones useful language learning tools? *ReCALL*, 16(1), pp. 11-84.
- [18] Xanthopoulos, S., and Xinogalos, S., (2013) A Comparative Analysis of Cross-platform Development Approaches for Mobile Applications. *ACM International Conference Proceeding Series*, pp. 213-220.
- [19] Ciman, M., and Gaggi, O., (2017) An empirical analysis of energy consumption of cross-platform frameworks for mobile development. *Pervasive and Mobile Computing*, 39, pp. 214-230.
- [20] El-Kassas, W.S., Abdullah, B.A., Yousef, A.H., and Wahba, A.M., (2017) Taxonomy of Cross-Platform Mobile Applications. *Ain Shams Engineering Journal*, 8(2), pp. 163-190.
- [21] <https://www.duolingo.com>
- [22] <https://pay.weixin.qq.com/index.php/public/wechatpay>
- [23] <https://intl.alipay.com/>
- [24] <https://www.paypal.com/us/webapps/mpp/home>
- [25] [www.datanyze.com/market-share/payments/](http://www.datanyze.com/market-share/payments/). Accessed 15 April, 2018
- [26] Somerville, Ian. (2011). *Software Engineering*. 9th ed. Addison-Wesley.
- [27] Leal, S., Vrij, A., Mann, S., and P.Fisher, R., (2010). Detecting true and false opinions: The Devil's Advocate approach as a lie detection aid. *Acta Psychologica*, 134(3), pp. 323-329.
- [28] <https://cordova.apache.org/>
- [29] <https://angular.io/>
- [30] <https://ionicframework.com/framework>
- [31] <http://framework7.io/>
- [32] stackshare, <https://stackshare.io/stackups/framework7-vs-ionic-vs-xamarin>, 8 April, 2018
- [33] TJVanToll, (2017). <https://developer.telerik.com/topics/web-development/what-is-angular/>, 15 April, 2018
-

- 
- [34] <https://www.apachefriends.org/es/index.html> [http://www.pcworld.com/article/204423/why\\_linux\\_beats\\_windows\\_for\\_security.html](http://www.pcworld.com/article/204423/why_linux_beats_windows_for_security.html), 8 April, 2018.
- [35] <http://www.apache.org/>
- [36] <https://www.mysql.com/>
- [37] <http://php.net/>
- [38] [https://en.wikipedia.org/wiki/LAMP\\_\(software\\_bundle\)](https://en.wikipedia.org/wiki/LAMP_(software_bundle))
- [39] Katherine Noyes, (2010) pcworld, [http://www.pcworld.com/article/202452/why\\_linux\\_is\\_more\\_secure\\_than\\_windows.html](http://www.pcworld.com/article/202452/why_linux_is_more_secure_than_windows.html), 8 April, 2018.
- [40] Katherine Noyes, (2010) pcworld, [http://www.pcworld.com/article/202452/why\\_linux\\_is\\_more\\_secure\\_than\\_windows.html](http://www.pcworld.com/article/202452/why_linux_is_more_secure_than_windows.html), 8 April, 2018.
- [41] edgehosting, <http://www.edgehosting.com/blog/2017/06/amazon-linux-versus-centos/>, 8 April, 2018.
- [42] centos, <http://www.centos.org/>, 8 April, 2018
- [43] Denise Sullivan, techwalla, <http://www.techwalla.com/articles/advantages-of-apache-web-server/>, 8 April, 2018.
- [44] fredosaurus, <http://www.fredosaurus.com/notes-db/transactions/acid.html>, 8 April, 2018.
- [45] BetterSQL, (29 July, 2014 ) clustrix, <http://www.clustrix.com/bettersql/acid-compliance-means-care/>, 8 April, 2018.
- [46] Garriga, M., Mateos, C., Flores, A., Cechich, A., and Zunino, A., (2016). RESTful service composition at a glance: A survey. *Journal of Network and Computer Applications*, 60, pp. 32-53.
- [47] Fielding, R., (2000). Architectural styles and the design of network-based software architectures. *Ph.D. dissertation* University of California, CA, USA;
- [48] Boyd, R., (2011). *Getting started with OAuth 2.0*. O'Reilly Media, Inc.
- [49] Stallings, W., and Brown, L., (2012). *Computer Security*. 2nd Ed., Pearson.
- [50] Campbell, B., Mortimore, C., and Jones, M., (2012). *JSON Web Token (JWT) bearer token profiles for OAuth 2.0*.
-



- 
- [51] Actualit ekino, (2015) ekino, <http://www.ekino.com/introduction-aux-json-web-tokens/>, 8 April, 2018.
- [52] Algreto-Badillo, I., Feregrino-Urbe, C., Cumplido, R., and Morales-Sandoval, M., (2012). FPGA-based implementation alternatives for the inner loop of the Secure Hash Algorithm SHA-256. *Microprocessors and Microsystems*, 37(6), pp. 750-757.
- [53] Ronglon, S., and Arpnikanondt, C., (2016) Signal: An open-source cross-platform universal messaging system with feedback support. *The Journal of Systems & Software*, 117(2016), pp. 30-54.
- [54] Somogyi, A.A, Lewis, D.D., Cohen, A.F., Flockhart, D.A., Ferro, A., Loke, Y.K. and Ritter, J.M. Available at: <https://bpubs.onlinelibrary.wiley.com/doi/full/10.1111/bcp.12317> (Accessed: 10 April 2018).
- [55] Sedgewick, R., and Wayne, K., (2011). *Algorithms*. 4th ed. Addison-Wesley.
- [56] users, <http://www.users.csbsju.edu/~lziegler/CS162/UnionFind.html>, 8 April, 2018.
- [57] wikipedia, [http://en.wikipedia.org/wiki/Tree\\_\(data\\_structure\)](http://en.wikipedia.org/wiki/Tree_(data_structure)), 8 April, 2018.
- [58] tutorialspoint, [https://www.tutorialspoint.com/software\\_testing\\_dictionary/end\\_to\\_end\\_testing.html](https://www.tutorialspoint.com/software_testing_dictionary/end_to_end_testing.html). 26 March, 2018.
- [59] Chen, Y.C., Kuo, Ma, W., Susilo, W., F., Towey, D., Voas, J., and Zhou, Z.Q., (2016) Metamorphic Testing for Cybersecutiry. *IEEE Computer*, 49(6) pp. 48-55.
- [60] Stati S, <http://www.softwaretestinghelp.com/what-is-user-acceptance-testing-uat/>, 14 April, 2018.
- [61] jasmine, <http://jasmine.github.io>, 8 April, 2018.
- [62] protractortest, <http://www.protractortest.org>, 27 March, 2018.
- [63] Weyuker, E.J., (1982) On testing non-testable programs. *The Computer Journal*, 25(4) pp. 465-470.
- [64] Chen, Y.C, Kuo, F., Towey, D., and Zhou, Z.Q., (2012) Metamorphic Testing: Applications and Integration with Other Methods. *Proceedings of the IEEE 12th International Conference on Quality Software 2012*, pp. 285-288.
- [65] Usability.gov, <https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html>, 14 April, 2018.
- [66] <https://github.com/>
-

- 
- [67] <https://git-scm.com/>
- [68] <https://www.teambition.com/>
- [69] Krista Trapani, cprime, <http://www.cprime.com/2016/03/what-is-agile-product-development/>, 8 April, 2018.
- [70] Alex Lichtenberger, (27 Aug, 2014 ) blog, <http://blog.itil.org/2014/08/allgemein/what-it-service-management-can-learn-from-the-agile-manifesto-and-vice-versa/>, 8 April, 2018.
- [71] Cockburn, A. and Highsmith, J., (2001) Agile software development: the people factor. *IEEE Computer*, 34(11) pp. 131-133.
- [72] Chan, F.K.Y, and Thong, J.Y.L., (2009) Acceptance of agile methodologies: A critical review and conceptual framework. *Decision Support Systems*, 46, pp. 803-814.
- [73] Reifer, D.J, (2002) How good are agile methods?. *IEEE Software*, 19(4), pp. 16-18.
- [74] Simon, (April 26, 2016) devdactic, <http://devdactic.com/ionic-push-notifications-guide/>, 8 April, 2018.
- [75] Tuckaman, B.W. (1965). Developmental sequence in small groups. *Psychological Bulletin*, 63(6), pp. 384-399.
- [76] Argote, L., McEvily, B., and Reagans, R., (2003). Managing knowledge in organizations: an integrative framework and review of emerging themes. *Management Science*, 49(4), pp. 571-582.
- [77] Katzenbach, J.R. & Smith, D.K. (1993). The Discipline of Teams. *Harvard Business Review*, 71(2), pp. 111-120.
- [78] Huo, M., Verner, J., Zhu, L., and Babar, M.A., (2004). Software quality and agile methodods. *Proceedings of the 28th Annual International Computer Software and Applications Conference*, pp. 520-525.
- [79] Greening, J., (2001). Launching extreme programming at a process-intensive company. *IEEE Software*, 36, pp. 57-66.
- [80] Murru, O., Deias, R., and Mugheddue, G. (2003). Assessing XP at a European Internet company. *IEEE Software*, 20, pp. 37-43.
- [81] Rasmussen, J., (2003). Introducing XP into Greenfield Projects: lessons learned. *IEEE Software*, 20, pp. 21-28.
-

- 
- [82] Schuh, P., (2001). Recovery, redemption, and extreme programming. *IEEE Software*, 18, pp. 34-41.
- [83] Ceschi, M., Sillitti, A., Succi, G., and Panfilis, S.D., (2005). Project management in plan-based and agile companies. *IEEE Software*, 22(3), pp. 21-27.
- [84] Nerur, S., Mahapatra, R., and Mangalaraj, G., (2005). Challenges of migrating to agile methodologies. *Communications of the ACM*, 48(5), pp. 73-78.
- [85] Cohn, M., and Ford, D., (2003). Introducing an agile process to an organization. *IEEE Software*, 36(6), pp. 74-78.
- [86] McManus, J., (2003). Team agility. *Computer Bulletin*, 45(5), pp. 26-27.
- [87] Towey, D., Foster, D., Gilardi, F., Martin, P., White, A., Jiang, Y., Pan, Y. & Qu, Y. (2017). Developing an Open Educational Resource: Reflections on a Student-staff Collaboration. *Proceedings of the IEEE 41st Annual Computer Software and Application Conference* 2017, pp. 711-717.
- [88] Towey, D., Foster, D., Gilardi, F., Gorla, C., Martin, P. & White, A. (2015). Researching and Supporting Student Note-taking: Building a Multimedia Note-taking App. *Proceedings of the IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)* 2015, pp. 54-58.
- [89] <https://cloud.tencent.com/?lang=en>. 9 April, 2018.
- [90] Srinivasan, L., Scharnagl, J., and Schilling, K., (2013) Analysis of WebSockets as the New Age Protocol for Remote Robot Tele-operation. *3rd IFAC Symposium on Telematics Applications*, pp. 83-88.
- [91] Bangor, Aaron, Kortum, P., and Miller, J., (2009). Determining what individual SUS scores mean: adding an adjective rating scale. *Journal of Usability Studies*, 4(3), pp. 114-123.

---

## 11 Appendices

### A Support Documents

#### Requirement:

A client (the UNNC Language Centre) has requested a software application to support and augment existing online language learning at UNNC

#### Background:

The Peer supported online language learning exchange (OLLE) is a Nottingham Advantage Award module jointly created in 2013, managed and delivered by the Language Centre and the vis-à-vis student organisation, through which over 240 students at the University of Nottingham have learnt to learn how to improve their language learning skills together by making leverage of the affordances of online learning and tools. The aim of this GRP project is to create a UNNC app that will be freely open to all students of the University of Nottingham and to partner University students and, perhaps for a small fee, to external participants worldwide.

#### Project Outline:

The selected students will form a software engineering team (initially called Team2017-02), and work together to complete a full cycle (maybe several full cycles) of the software engineering process, resulting in delivery of a tool (and accompanying artefacts), as requested by the client.

The SE team will need to go through a complete requirements engineering process to identify the exact SE project requirements. Part of this will involve applying an appropriate requirements elicitation methodology.

The team will need to make informed decisions about which SE process approaches or methodologies to apply to this project. (Experiences from previous SE teams may prove useful.)

Over the course of the project, the SE team will need to produce several artefacts, including (but not limited to): a report on the current situation, a system requirements specification, design documentation, team management and planning documentation, prototypes, progress reports, verification and validation plans, code, code documentation, and instructions manual.

The target goal of the SE project will be to deliver, on time, a tool/platform that can support online language exchange. An informal vision of this, from the client's perspective, has been proposed as follows:

"A social networking language learning APP in the form of a semi-open educational resource within the University of Nottingham and its partner universities; with a fee paying version/aspect for all other external users."

---

Figure 13: Initial Project Description

## B Details of the survey

The survey was conducted to help stakeholders decide the platform the app should be developed for. This survey only consisted of two questions that were students' ID and the mobile platform students who enrolled in OLLE module currently use. The questionnaire was distributed by VAV, requiring no ethical approval consent on our behalf. They put this questionnaire in the Wechat group of OLLE students and the project team collected the final result.

According to the statistics, there were 40 students answered this questionnaire in total. Sixty percent of students' mobile platforms were iOS and the rest of students chose Android

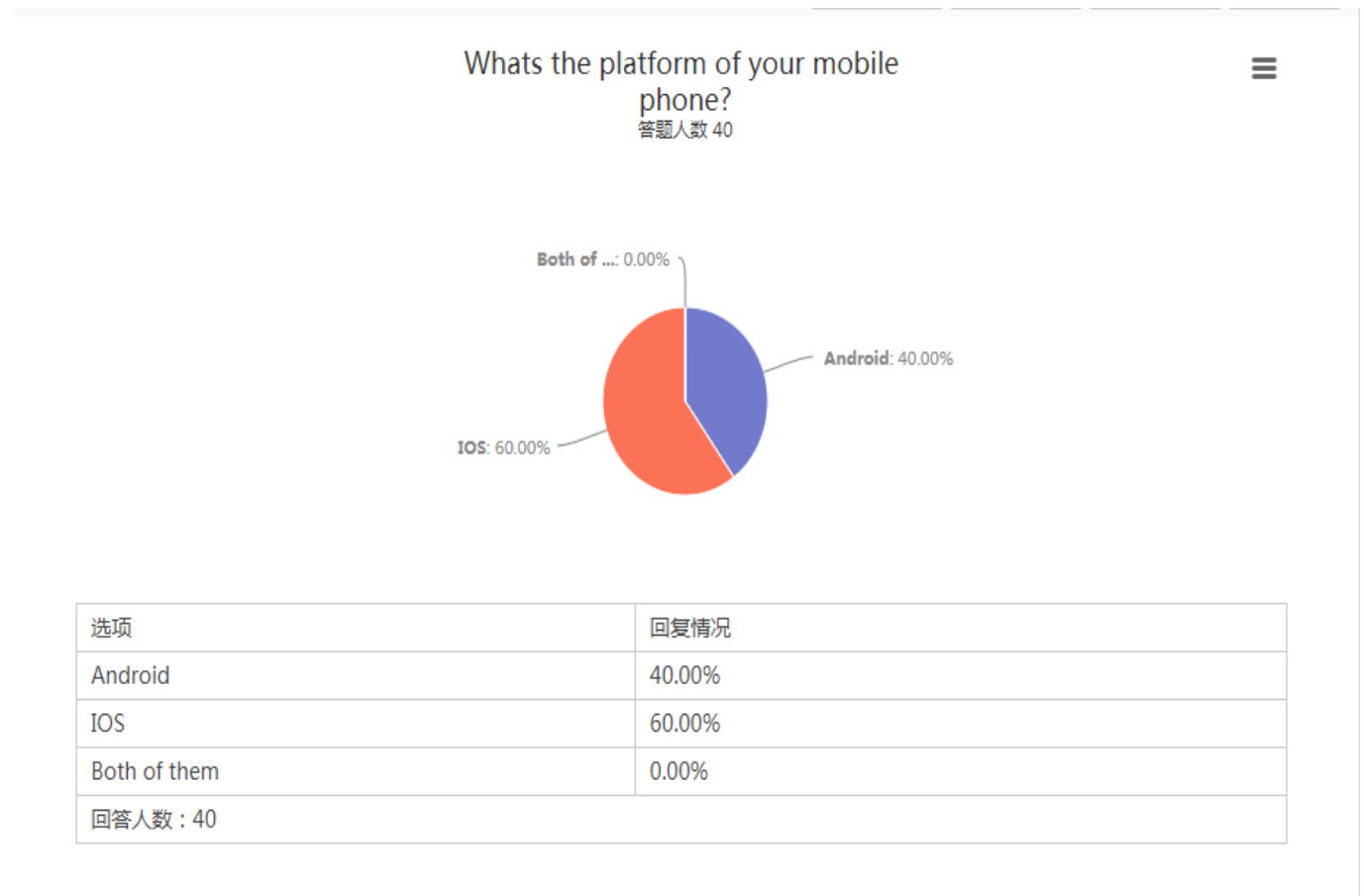


Figure 14: Survey result

---

## C Software Requirements

### C.1 Initial Functional Requirements:

1. The app integrates a Forum, a Calendar and six public chat rooms.
  - (a) Data will be stored in databases.
2. The app will work on both Android and iOS, downloading from their corresponding App stores.
  - (a) Cordova plugins will enable the use of phone features, and to distribute it through the app stores.
  - (b) Result will be an hybrid-app.
3. The app must autonomously process the validation and verification of NAA user accounts.
  - (a) User creates an account typing their own username and password.
  - (b) A query is made to the database to check whether the email is a Nottingham university email and whether the username is in use or not.
  - (c) Validation code sent to user's email.
  - (d) User data stored in the Database.
4. An administrator can delete any user account.
  - (a) Administrator makes a query to the database to filter by username.
  - (b) Administrator selects a user.
  - (c) Administrator confirms a consent message to delete account.
  - (d) Account information is deleted from the database.
5. An administrator can modify the discussion topic of a given chat room.
  - (a) Administrator taps on top of the current topic.
  - (b) Administrator writes the new topic.
  - (c) The HTML tag is replaced by the new String typed by the Administrator.
6. Administrator must be able to check individual user participation in a chat room.
  - (a) Each listed user in the database of a given chat room will have a variable that counts the times they send a message.

- 
7. An administrator can create/modify/access/delete a forum topic, archiving the previous topic.
    - (a) Administrator taps on top of the create button.
    - (b) Administrator writes the topic description and name.
    - (c) Previous topic, including all its posts, is archived into 'Previous Weekly Tasks folder'.
  8. An administrator can filter those users who have not completed the weekly task (open learning log) in the forum.
    - (a) Administrator selects the 'Not submitted' option from a scrollable menu.
    - (b) Query sent to database.
    - (c) Users who have not completed the task appear in the administrator's mobile screen.
  9. An administrator can send a reminder to those users who have not completed the weekly task.
    - (a) Administrator selects the 'Not submitted' option from a scrollable menu.
    - (b) Administrator taps on 'Select all' button.
    - (c) Administrator taps on 'Send reminder'.
    - (d) Cordova Plugin command to send notification to the phone.
  10. An administrator can change the task submission deadline in the forum.
    - (a) Administrator types the string of the new deadline.
    - (b) Reminder sent to all NAA users.
  11. An administrator can create/modify/access/delete an event in the Calendar.
    - (a) Administrator inputs the string of the name and description of the event.
  12. A NAA user can post/reply/modify a comment *only for the current topic*.
    - (a) Title and body user input required to submit the comment to the forum.
    - (b) Comment stored in the forum's database.
  13. NAA user and Administrators can view all comments from all topics in the forum.
  14. NAA user will receive a reminder by the app if they fail to post a comment during the week's topic.
-

- 
- (a) Administrator changes/creates a new topic.
  - (b) Reminder sent using ID stored in the database.
15. NAA user can only view the description of an event in the Calendar.
- (a) A query is sent to the database to ask for the description.
16. NAA user can enter/exit a language chat room at any point in time.
- (a) Username and ChatID are added to the given chat's database.
17. NAA user can send/view instant messages in the chat room.
- (a) Encryption occurs using user ChatID.
  - (b) Message stored in the Chat database.
18. NAA user can view all the other users in a given chat room.
- (a) Query to the chat's database for all users.
  - (b) New screen will display the results.
19. NAA user can change their password and username in the settings screen.
- (a) User input is modified in the database through a query.
20. Non-NAA user can use the app without the approval from administrators.
21. Non-NAA user must pay an in-app fee if they are not tri-campus students.
22. A survey, designed by the stakeholders, is completed during registration.
23. Non-NAA users can add/delete friends.
24. Non-NAA users can make chat one-to-one or in groups with friends.
25. A matching system suggests friends to Non-NAA users based on the score on the survey.
26. Non-NAA users will receive weekly news/events developed by Administrators.
-



---

## **C.2 Modified Functional Requirements:**

Functional Requirement 3 was changed during development phase to the following requirement:

3. All NAA accounts will be created by administrator accounts users.
  - (a) Administrator collect user information at the start of the academic year.
  - (b) User information is loaded in an excel file.
  - (c) Excel file is uploaded into the app, creating all the user accounts.
  - (d) Email notifies user regarding the activation of their account.

## **C.3 Non-Functional Requirements:**

1. App response time to user commands must be no more than 2 seconds.
2. Only Administrator can update the database of NAA users.
3. The database must be able to support up to 300 users for the initial year of the project.
4. The app must support cross-platform usage.

---

## D Testing

### D.1 E2E Testing

**Figure 15** shows the set of E2E test cases performed to the app. The tested page component is shown in the left-most column. A description of the test is also provided. The results from protractor indicated that almost all main functions meet expected condition.

Functions	Test cases	Description(Expected outcome)	Result
Log in page	1	should direct to the login page	Passed
	2	should log in failed if not enter the right identification	Passed
	3	should log in successfully	Passed
	4	should return to log in page after logout	Passed
Forum page	1	should go to Forum page	Passed
	2	should show the first week	Passed
	3	should show the first week topic	Passed
	4	should show the deadline of first topic	Passed
	5	should create a new topic and view detail of it	Passed
	6	should be able to edit topic	Passed
	7	should be able to delete topic	Passed
Chat page	1	should go to Chat page	Passed
	2	should be able to create new chat room	Passed
	3	should be able to send messages	Passed
	4	should be able to check detail of chat room	failed
Calendar page	1	should go to Calendar page	Passed
	2	should create new event	Passed
	3	should be able to check detail of event	Passed
Account page	1	should go to MyOLLE page	Passed
	2	should be able to change name	Passed
	3	should be able to change password	Passed

Figure 15: Collated output from protractor.

---

## D.2 Metamorphic Testing

Figure 16 shows the set of metamorphic test cases performed to the app. The source and follow-up tests are presented under the Input column, highlighting the metamorphic relationship between these.

Test case	Input	Metamorphic Relationships	Result
1	Two different types of user( Admin&student)login	Both admin and NAA students should all direct to forum page after login successfully	passed
2	User 1 send a message in English chat room, User 2 send a message in English chat room too.	The message sent by User 1, and the message sent by User 2, should be displayed at the right side of their respective phone screens. Similarly, the message received by User 1, and the message received by User 2, should be displayed at the left of their respective screens.	passed
3	Admin 1 added new event. Admin edited this event.	Both adding and editing event functions should load and render the create new event page component.	passed
4	Two different types of user( Admin&student)login	Admin should have new topic button in forum, edit/delete button of topic, new chat form button and edit/delete button in chat room, add new event button and edit/delete button in calendar. The above buttons shouldn't be seen in student interface.	passed
5	User 1 posted a reply to a topic, Then user 2 posted a reply to user 1's reply	For both User 1 and User 2, whenever they submit the reply of the respective topic/reply, the subsequent reply should be displayed below the topic/reply	passed
6	User 1 continually send 10 messages in English chat room.	Messages send by user should be displayed with time order, with a increase sequence from top to bottom.	passed
7	User 1 try to change his username to that one of User 2. User 2 tries to change his username to that one of User 1.	No two users can have the same username.	passed


Figure 16: Metamorphic testing results

---

### D.3 User Acceptance Testing

**Figure 17** shows the ten questions contained in the System Usability Scale questionnaire. Users tested the app, without our guidance and then completed the survey. Responses are quantified, with an input range from one to five. The survey was completed through the participant's phone, so we were not present when it was being filled-up. It was completed by fourteen participants.

### System Usability Scale Questionnaire



1. I think that I would like to use this app frequently \*

Strongly disagree ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 Strongly agree

2. I found the app unnecessarily complex \*

Strongly disagree ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 Strongly agree

3. I thought the app was easy to use \*

Strongly disagree ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 Strongly agree

4. I thought that I would need the support of a technical person to be able to use this app \*

Strongly disagree ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 Strongly agree

5. I found the various functions in this app were well integrated \*

Strongly disagree ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 Strongly agree

6. I thought there was too much inconsistency in this app \*

Strongly disagree ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 Strongly agree

7. I would imagine that most people would learn to use this app very quickly \*

Strongly disagree ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 Strongly agree

8. I found the app very cumbersome to use \*

Strongly disagree ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 Strongly agree

9. I felt very confident using the app \*

Strongly disagree ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 Strongly agree

10. I needed to learn a lot of things before I could get going with this app \*

Strongly disagree ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 Strongly agree

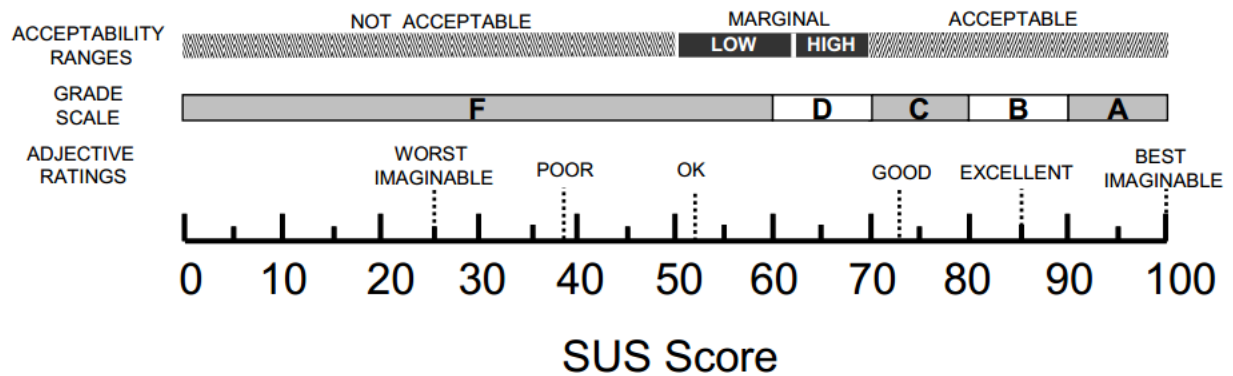
Figure 17: SUS questionnaire

Figure 19 presents the associated score that each participant obtained for each question. The calculation method is: minus 1 point for odd term's score, use 5 to minus original score for even term. Then add all processed scores together and multiply by 2.5. The result is the individual SUS score. Because we chose 14 volunteers to test our app and excepted one unusual score, so we have to add rest 13 SUS scores and get the average of it. Shown below, the average score of our app is 68.27.

Question No.	1	2	3	4	5	6	7	8	9	10	SUS Score
User 1	5	1	5	1	3	1	5	1	5	1	95(X)
User 2	2	3	4	1	2	4	4	5	2	1	50
User 3	4	2	4	2	4	2	4	3	4	2	72.5
User 4	5	4	3	4	5	4	3	4	3	5	45
User 5	4	1	5	1	4	2	5	1	4	1	90
User 6	4	2	4	4	3	3	4	3	4	2	62.5
User 7	5	5	5	4	3	3	3	3	5	3	57.5
User 8	5	2	5	2	4	2	5	2	5	2	85
User 9	3	2	5	1	3	3	5	1	5	1	82.5
User 10	3	2	4	1	5	1	5	1	5	1	90
User 11	3	1	4	1	3	2	5	1	5	1	85
User 12	4	2	4	3	4	2	4	1	3	3	70
User 13	1	4	1	4	1	2	3	5	2	4	22.5
User 14	2	1	4	1	3	1	4	2	3	1	75
Abandon the highest score											Average score:68.27

Figure 18: 14 user's SUS score

From Bangor's [91] research(shown on figure 19), we compared our score with it. Therefore, we have a clear evaluation of our app.



**Figure 4.** A comparison of the adjective ratings, acceptability scores, and school grading scales, in relation to the average SUS score

Figure 19: Bangor's SUS relationships

---

## **E Minutes**

### **E.1 Minutes 1**

Form: Informal meeting

Time: 2017.10.17 PM 13:30-14:10

Place: PB ground floor

Members: All

Summary: We should discuss with supervisor about:

1. Use of open source.
2. Developing language/environment language of the server.
3. Access of the Nottingham system.
4. Team work division.

Ask the client about:

1. Website or computer/phone app, run on Android/IOS/Windows?
2. Only Student/student-teacher?

### **E.2 Minutes 2**

Form: Formal meeting

Time: 2017.10.19 PM 12:00-12:20

Place: NTB ground floor

Members: All except for Zhengchun Zhao (attending LCF meeting)

Things to do:

1. Download ethic form from Moodle.
2. Send Consent form Information sheet to supervisor and arrange the interview.
3. Contact the teacher first: with no programming knowledge, but expert in language teaching
4. Prepare questions for the interview in detail (not beyond capability) , rehearse interview for several times(maybe create prototypes and personas first)

---

### E.3 Minutes 3

Form: Informal meeting

Time: 2017.10.24 PM 13:30-15:20

Place: PB ground floor

Members: All

Notes:

1. Who is going to attend the interview? When?
2. Teacher/students the same account?
3. if different account not enough account cant judge if they need to pay
4. Teacher be assigned or applied by students Decide by: Do the stakeholder provide learning materials?  
If No, ranking system to judge everybody: become teacher?
5. Social network : how to communicate
6. Meeting scheduling tool? Forum(for public)? chat(for private)?
7. Content of the forum(asking questions make new friends)?
8. People post things irrelevant? Report/inspection
9. How many languages to display? All, with filter option
10. Result: Keep simple first, chat material uploading personal profile

---

#### **E.4 Minutes 4**

Form: Formal meeting

Time: 2017.10.26 PM 12:00-13:00

Place: Third Place

Members: All

About the interview:

1. Prepare: One person asks, others act as stakeholder (make it hard).
2. Start searching for all kinds of information relevant before interview, make record of the result.
3. After interview: ask for permission to keep in contact(meet regularly to fulfill the specifications)
4. Ask for sensitive information(Email/password) about students and teachers.

Group work:

1. Start learning Latex (for research hand in 2nd of Nov.)
2. Search materials of assigned aspects.
3. Sen: Payment system.
4. Tingting: Chat system.
5. Albert/Roy: Android vs IOS.
6. Eddie: Creating social networks.
7. Jane: Process of making an app.



---

## E.5 Minutes 5

Form: Formal meeting

Time: 2017.12.2 PM 12:00-14:30

Place: Mr. Duck Restaurant

Members: All

Notes:

1. Result of interview: stakeholder wants : a forum + chat(contains group function)
2. Hard point: no resources needed, but realize a synchronized forum of moodle (every log posted, every respond etc on the app should also be on moodle page automatically.) Only authorized teacher can start a topic in forum, students can only create subtitle :no need to encourage. (NAA credit support)
3. Problems to be solved: low viscosity of users. (hard to solve)
4. Report hand in : no need to be formal, but with reference.
5. Information about other stakeholders: Former manager, now in Oxford and is a VAV staff.
6. Evaluate their requirement and negotiate to remove the unrealistic ones.
7. After final prototype created: show three stakeholders and check: they may not agree.
8. Two ways for group chat: friend list(add familiar people) motivation? Or set official groups
9. Problem remains: should only skilled people able to create group chat?
10. About pay system: not needed now, but still learn it, in case stakeholder change their minds. (consider situation in different countries: for all three campuses)
11. Account and password: no access to schools, but can use the email: set our own database, costs money.

---

## E.6 Minutes 6

Form: Formal meeting

Time: 2018.3.6 PM 7:30

Place: SSB 214

Members: All

Achieved:

1. Usage of XAMPP and Mysql.
2. Able to insert data into database.
3. Generic functions of sending and retrieving information to the back end.
4. Divide members into three groups: chatcalendar and forum functions and start working.

To be solved:

1. Environment settings of ionic gets tons of errors.
2. Not frequent enough usage of Github (updating code) and Teambition (reporting progress).

---

## E.7 Minutes 7

Form: Formal meeting

Time: 2018.3.13 PM 6:10

Place: Third place

Members: All

Achieved:

1. Chat system operates quite well.
2. Able to show interface on smart phones by Phonegap.

To be solved:

1. Users unable to sign up and log in on their own PCs.
2. Forum gets inconsistent when applying the reply function.
3. Difficulties in finding materials of API.

---

## **E.8 Minutes 8**

Form: Informal meeting

Time: 2018.3.17 PM 2:00-6:00

Place: SSB 214

Members: All

Achieved:

1. Solved problems in log in system.
2. Able to edit personal information.
3. Learned functions of calendar on Youtube.
4. Successful implementation of key board plugins.

To be solved:

1. Error exist in sliding of chatting interface.

## **E.9 Minutes 9**

Form: Formal meeting

Time: 2018.3.20 PM 6:00

Place: Third place

Members: All

Achieved:

1. Minor problems of chatting system are solved.
2. Adding of administrator to add and delete chatting topics.

To be solved:

1. Errors exist in Calendar, interface is not showing.
2. Trying to realize the reminder.

---

### **E.10 Minutes 10**

Form: Informal meeting

Time: 2018.3.24 PM 2:00

Place: SSB 410

Members: All

Achieved:

1. Calendar functions work fine now.
2. Rewriting of back end and solved some relevant mistakes.

To be solved:

1. Database is not applied in some portions of code.

### **E.11 Minutes 11**

Form: Formal meeting

Time: 2018.3.28 AM 10:30

Place: SEB 432

Members: All

Achieved:

1. Event can be permanently stored in calendar now.
2. Adding the function of Event description and cancel adding an event.
3. Database tables of all functions are completed.

To be solved:

1. Should include a super administrator to control access of administrators.
2. Add style sheet to the whole app.
3. Write a user manual.

---

### **E.12 Minutes 12**

Form: Formal meeting

Time: 2018.4.3 PM 6:30

Place: Daves home

Members: All

Achieved:

1. Minutes of this semester.
2. CSS settled and applied.
3. Some part of final report.

### **F User-Manual**

---

# User Manual

## Social Language Learning Application

---



---

# About This Manual

## Purpose

This manual has been written to help you to understand and use the Social Language Learning application. It displays the functional capabilities and operational details of the Forum, Calendar and Chats and contains the procedures that you should know for using the application.

The database maintenance tasks have not been covered in this manual.

## Intended Users

This manual is primarily intended for normal users who are students enrolled in OLLE and an administrator of VAV.

## Organization of the Manual

This manual is organized as follows:

Chapter	Description
<b>Introduction</b>	It provides an overview of the application. It also provides the details of the hardware and software requirements, and its interfaces with other systems.
<b>Getting started</b>	It provides a brief introduction about the general working features of the application that will assist you while learning languages.
<b>Reflective learning log</b>	It provides the description for an administrator to manage tasks and steps for normal users to be followed to finish weekly tasks.
<b>Calendar</b>	It gives the description for an administrator to manage events in calendar and explains how normal users use it.
<b>Chats</b>	It offers the description for an administrator to manage chat rooms and how normal users interact with others in chat rooms.
<b>Operating Environment</b>	It provides the details of the operating environment used in the application.

Table 1: Organization of this user manual



---

## 1 Introduction

### 1.1 User Roles and Access Rights

The user roles are limited to the types of user accounts – normal users or administrators. Access of the functionalities will be presented based on the user roles.

The following are the users of the application:

- Normal user: Student enrolled in OLLE module
- Administrator: User from Vis a vis

### 1.2 Features

The application is envisaged to improve overall efficiency and effectiveness for students in learning languages and for staffs in OLLE in managing students. Activities of Forum, Calendar and Chats can be divided in the following sub-groups:

- Writing a learning log regarding to a topic —Normal user and Administrator
- Manage topics (create, delete and modify topics) —Administrator
- Check impending events in the calendar —Normal user and Administrator
- Manage calendars events (create) —Administrator
- Share learning experiences with other students in chat rooms –Normal users, Administrator
- Manage chats rooms including create and delete chat rooms and modify topics —Administrator

## 2 Getting started

The section includes the details of logging on process and accessing the basic required functionality of the device.

### 2.1 Logging in

Access to the application is limited to tri-campus students. To use it, first log on to the app using your email or username and password.

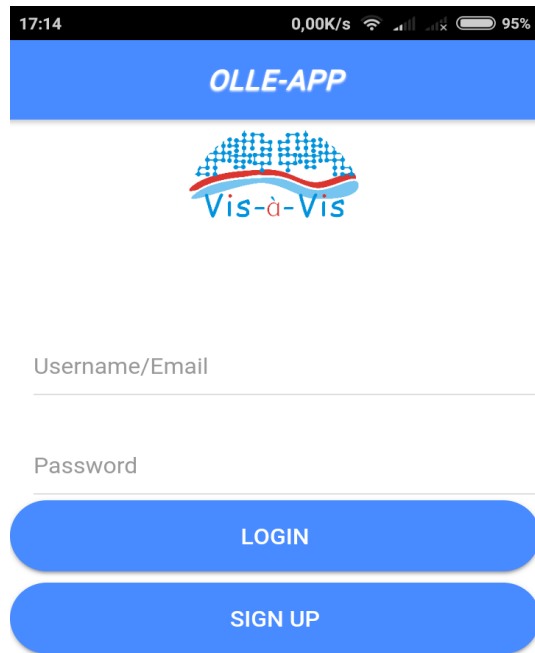


Figure 1: Log in

#### 2.1.1 How to create an account

1. Click **SIGN UP** in the log in page
2. You have to fill in a registration form using a university email and you are strongly recommended to use a password differs from the university emails password for a safety reason.

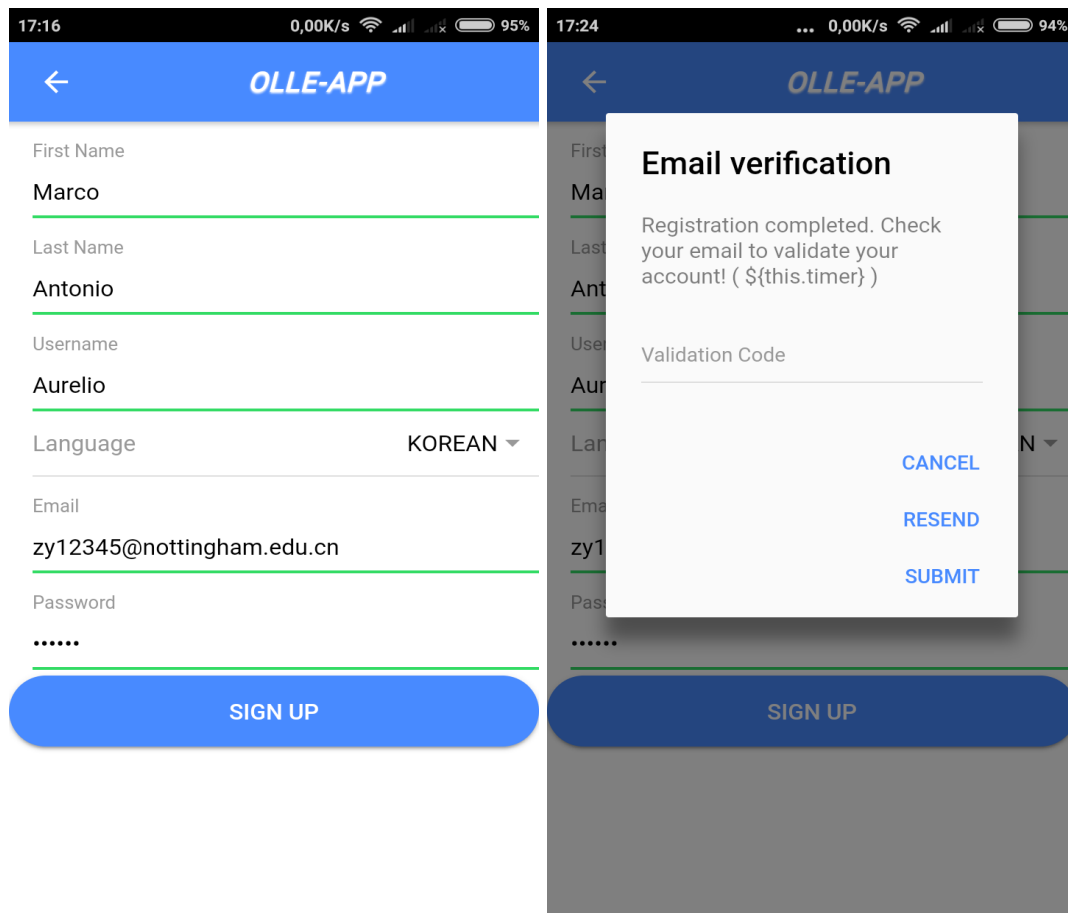


Figure 2: Sign up

3. Once you submit a correct registration form, you are prompted to provide the validation code that has been sent to your university's email address. Then, you have access to the application as a normal user.

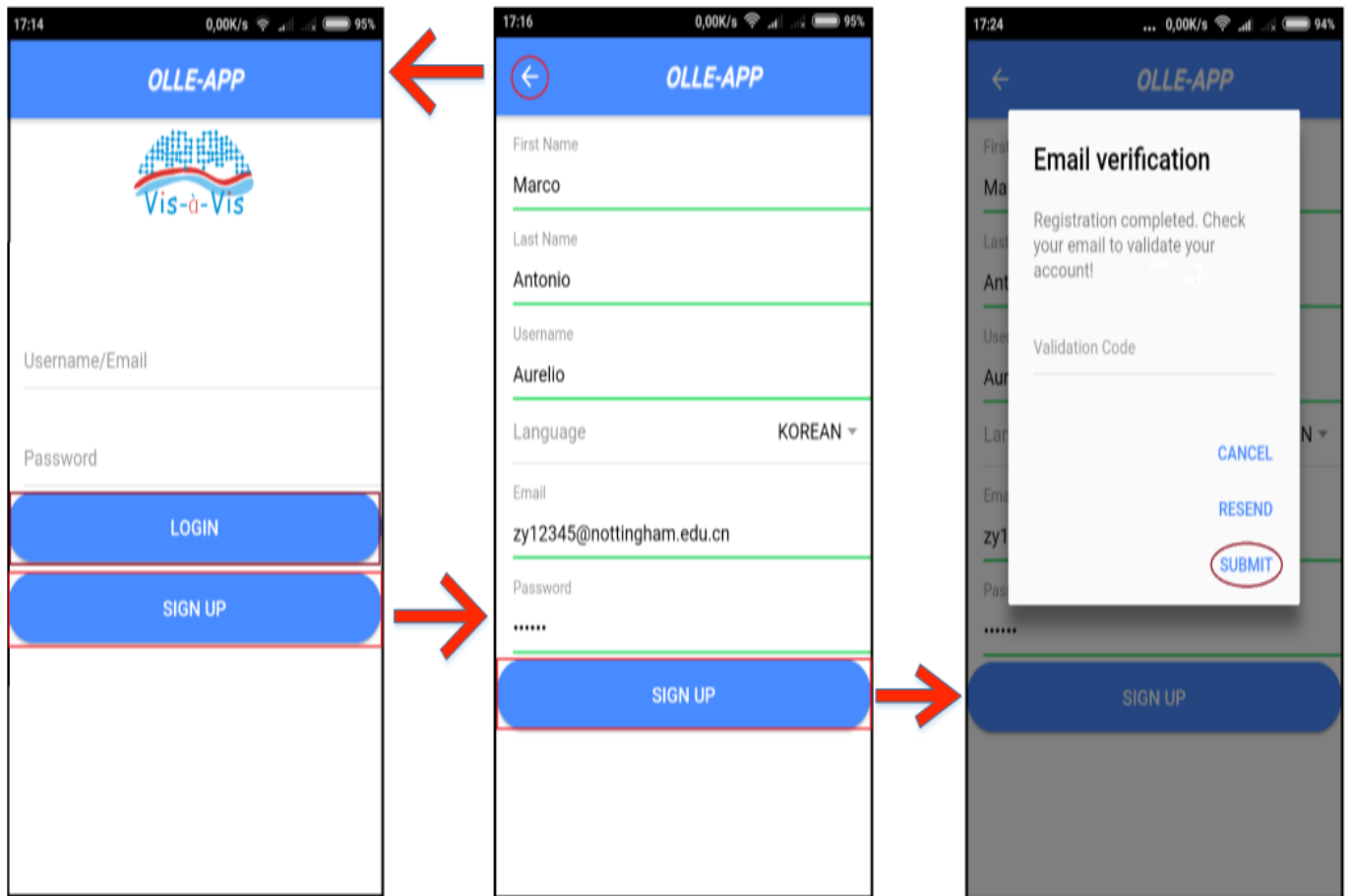


Figure 3: Overall process of registration

**NOTE:**

If the registration is not successful (email address is invalid, username already exists, required fields did not fill up and validation code is incorrect etc.), you will get a toast with corresponding message.

If you are an administrator, here is an username and password:

- Username: scyam1
- Password: 123456

### 2.1.2 Change password

To change your password:

1. Click **MyOLLE** and go to the **Account Management** page.
2. Click **CHANGE PASSWORD**.
3. Edit your password and click **SAVE** to finish changing your password.

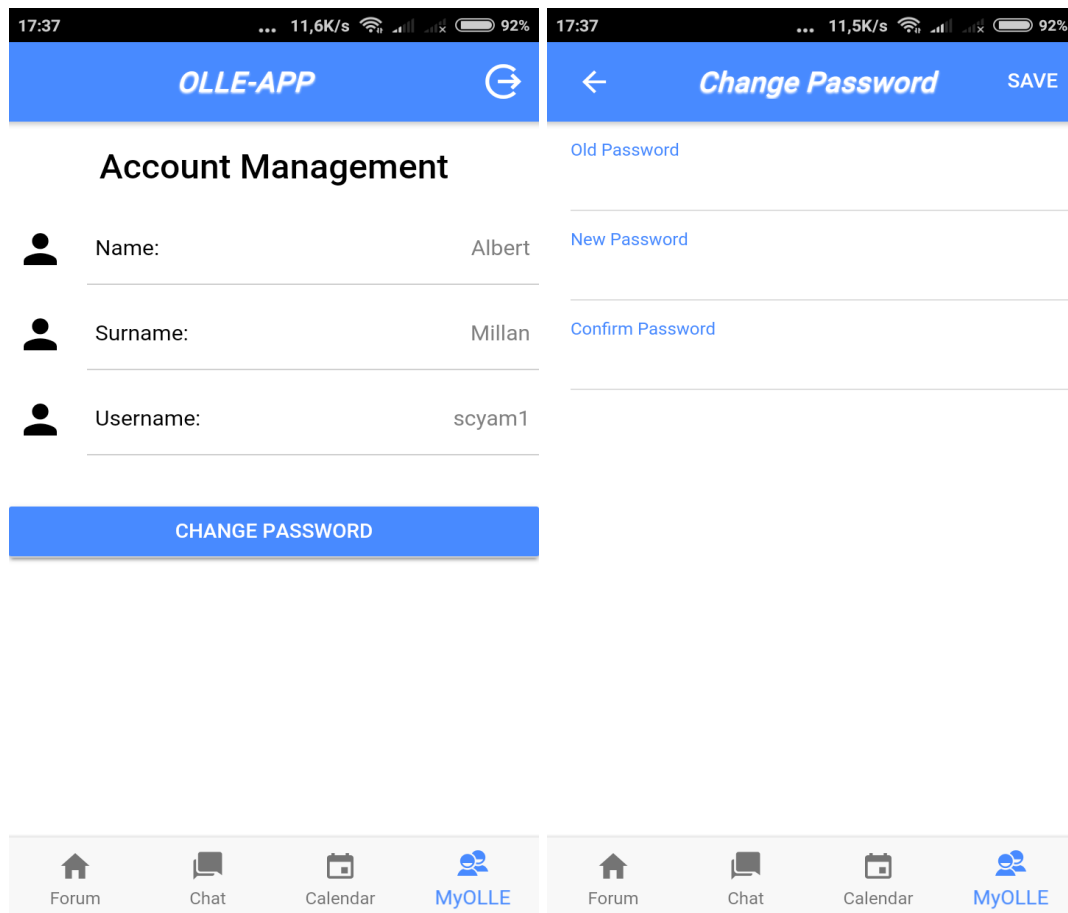


Figure 4: Change password

### 2.1.3 Change username

To change your username:

1. Click **MyOLLE** and go to the **Account Management** page.
2. Click **Username**.
3. Edit your username and click **SAVE** to finish changing your username.

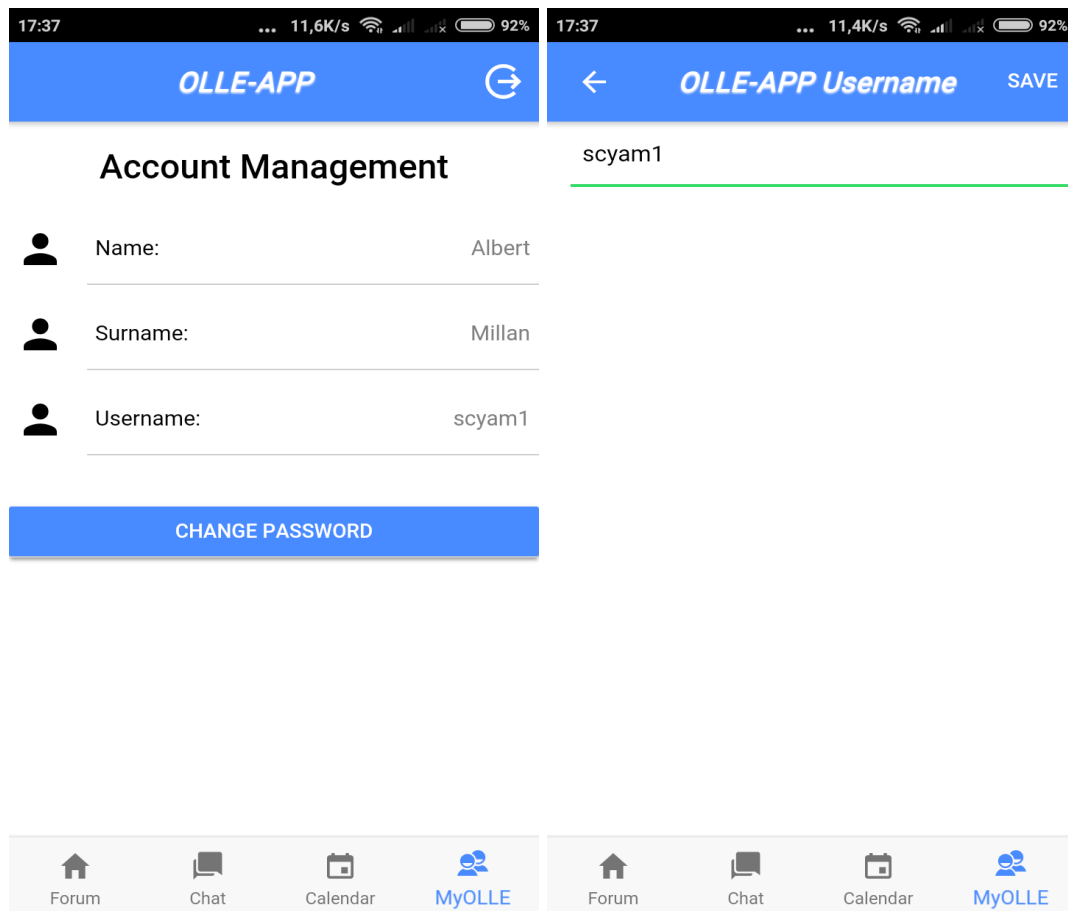


Figure 5: Change username

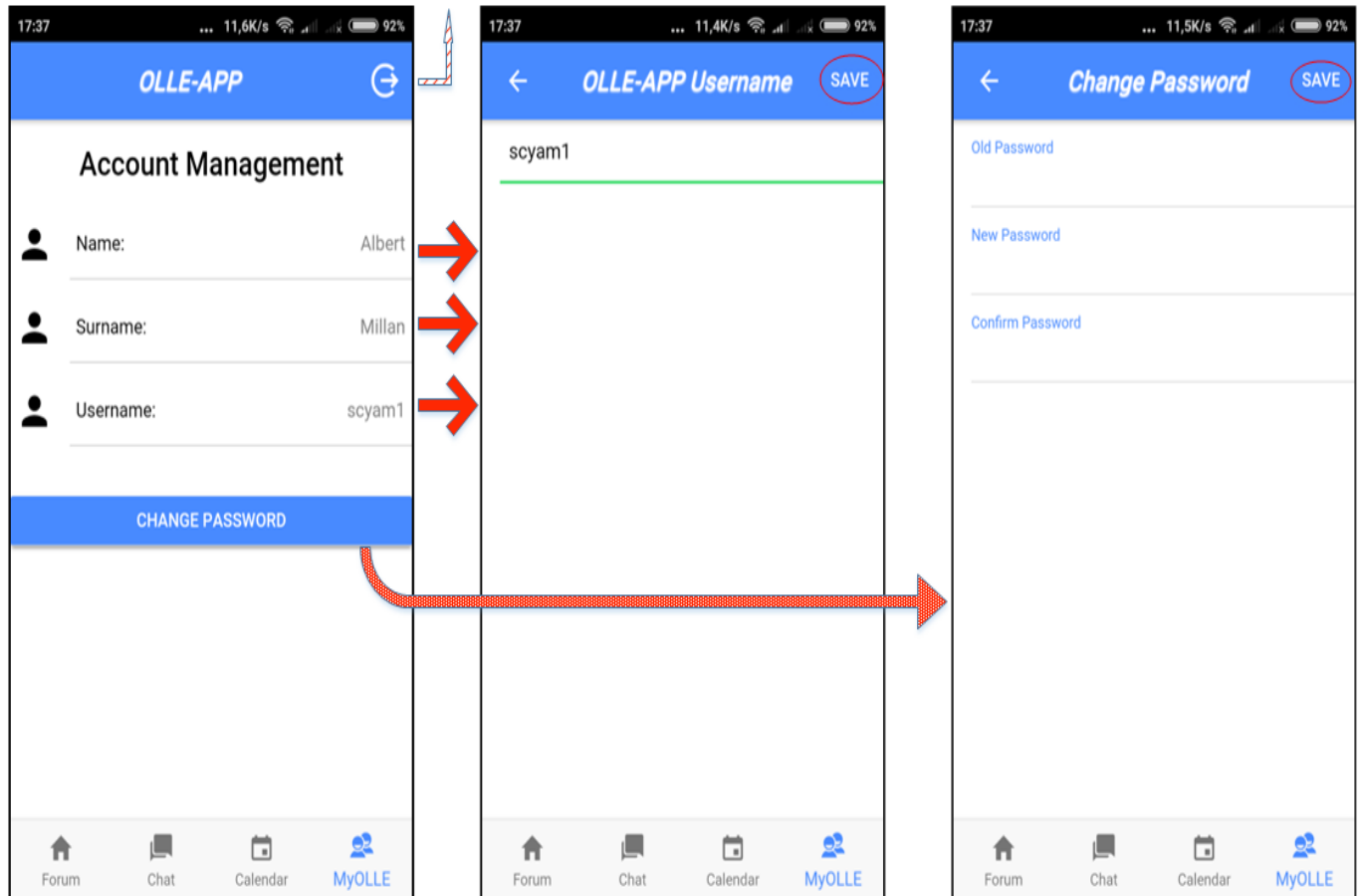


Figure 6: Overall process of changing username and password

### 3 Forum

This chapter includes the details of how an admin user manages the forum and how a normal user interacts with the forum.

#### 3.1 Create a weekly topic (Administrator)

To create a weekly topic:

1. Click **NEW TOPIC**. The new form appears.

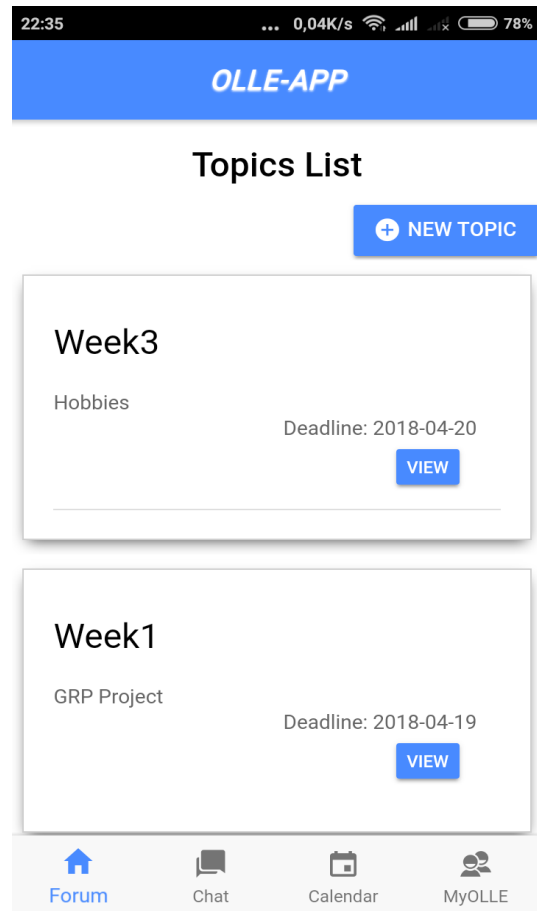


Figure 7: Create a topic

2. Fill in the form.
  - Week number is a positive integer



---

A screenshot of a mobile application interface for creating a new topic. At the top, a blue header bar contains a white back arrow and the text 'New topic'. Below this is a white rounded rectangle containing the form fields. The fields are: 'Week' with the value '3', 'Topic Title' with the value 'Hobbies', 'Topic Detail' with the value 'Tell me about your hobbies!', and 'Deadline' with the value '2018-04-20'. Each field has a green underline. At the bottom of the form is a blue button with the text 'ADD NEW'. The status bar at the very top shows the time '17:28', network speed '0,00K/s', and battery level '94%'.

Figure 8: Caption

3. A new topic should be presented in the forum page.

**NOTE:**

If the creation of a new topic is not successful, a toast with corresponding error message should be prompted.

### 3.2 Delete a topic (Administrator)

To delete a topic:

1. Slide the topic card to left and click **DELETE**.
2. Once an alert dialog appears, confirm the deletion by click **CONFIRM**, otherwise, click **CANCEL**.

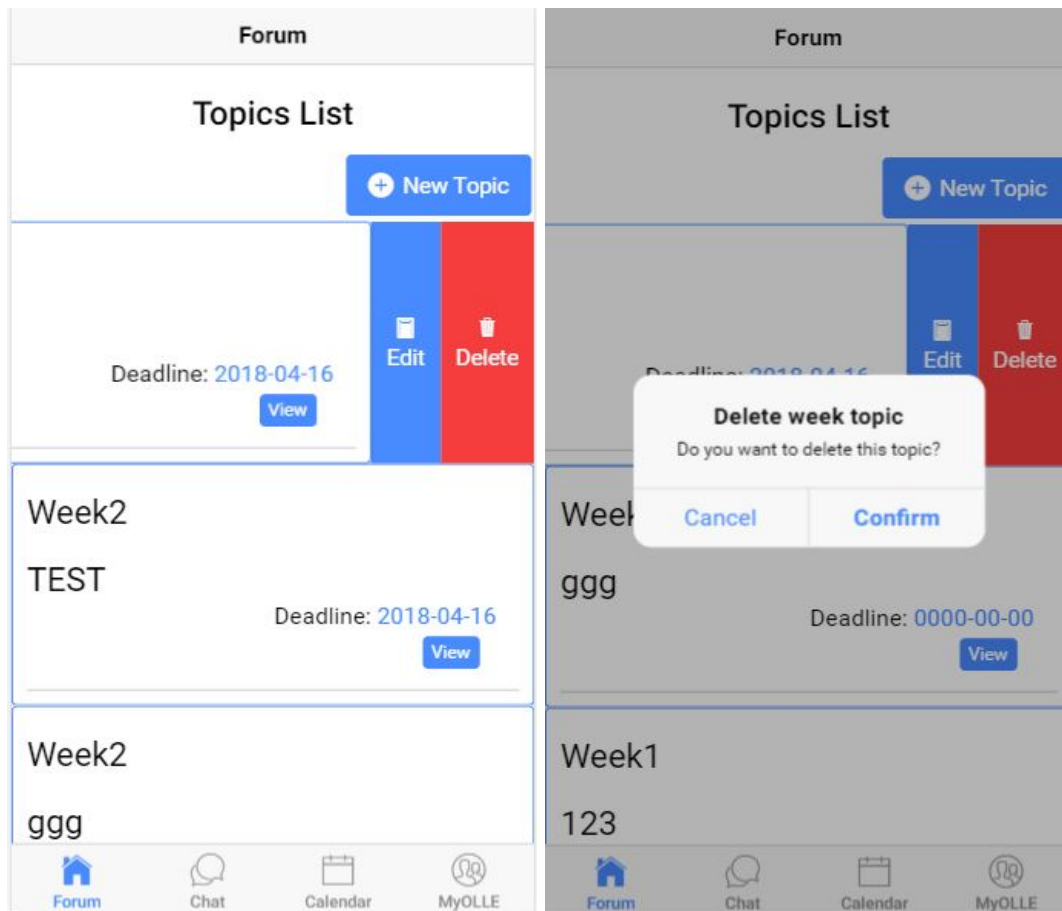


Figure 9: Delete a topic

### 3.3 Modify a topic (Administrator)

To modify a topic:

1. Slide the topic card to left and click **Edit** and a new form should be presented.
2. Fill up the new topic form.
3. Submit the form with correct topic information.

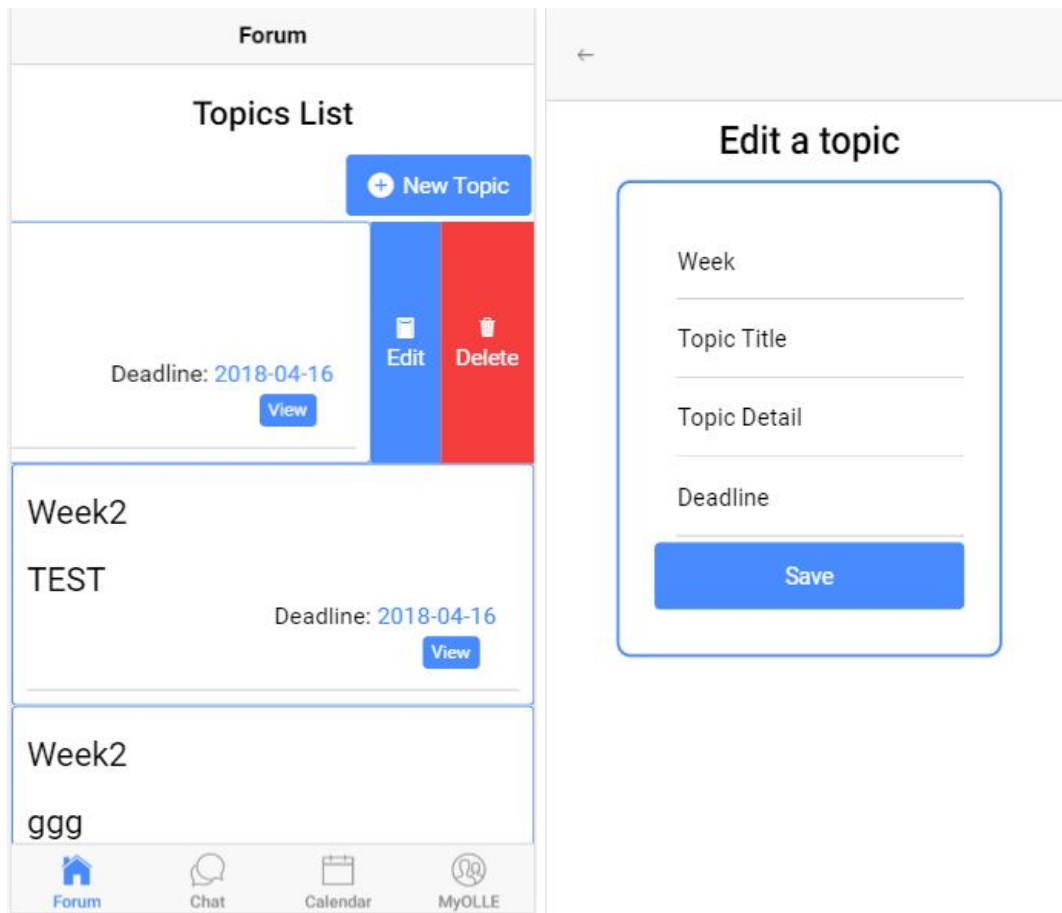


Figure 10: Modify a topic

### 3.4 View detail of a topic (Administrator and normal user)

To look through detail of a topic:

1. Click **VIEW**. The topic detail page with related others learning logs appear.

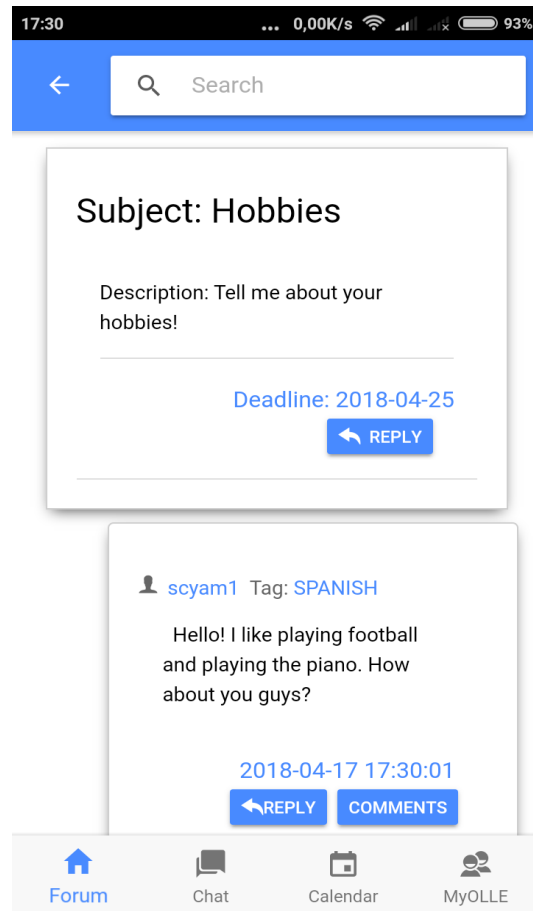


Figure 11: View details

### 3.5 Write a reflective learning log (Administrator and normal user)

To write a learning log:

1. Click **REPLY** within topic detail. The Reply page appears.
2. Write down your log and click **SUBMIT**.

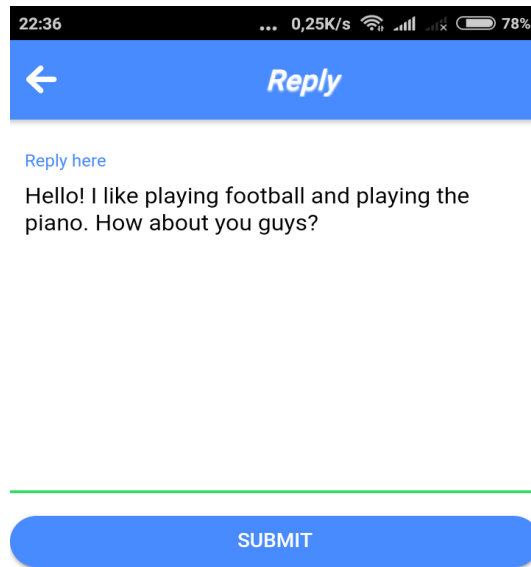


Figure 12: Write a log

3. On successful submission, your log should be presented in the topic detail page.

**NOTE:**

The content of your log should not be empty, otherwise, a toast with related message will be prompted.

### 3.6 Reply to others learning logs (Administrator and normal user)

To comment on others learning logs:

1. Click **REPLY** within someones learning log. The **Reply** page appears.
2. Write down your comment and click **SUBMIT**.
3. On successful submission, to see your comment, click **COMMENTS** and your comment will be presented below.

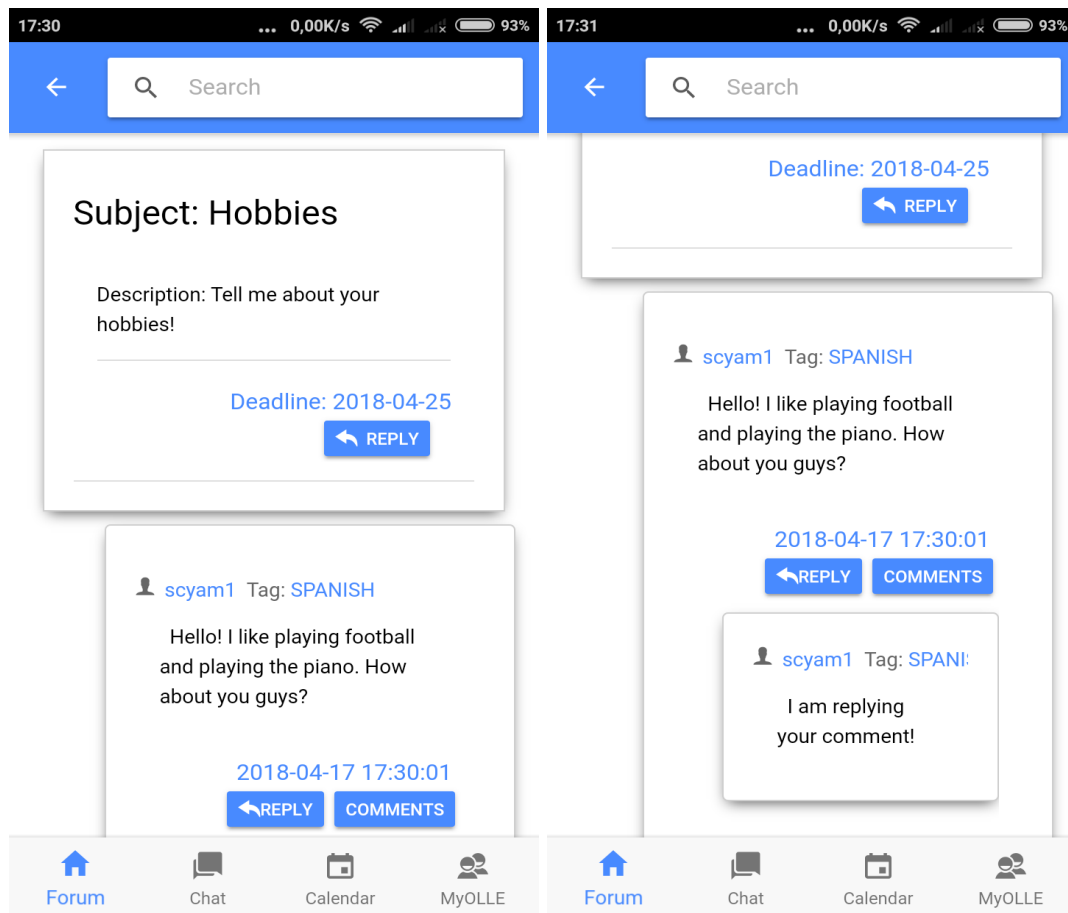


Figure 13: Comment

### 3.7 Search learning logs by a language

To search learning logs by ones target learning language:

1. Input a language (English, Mandarin, Spanish, French, Italian, German, Japanese or Korean).
2. Every learning logs whose posters language match the key word will be presented, and others logs will be invisible.

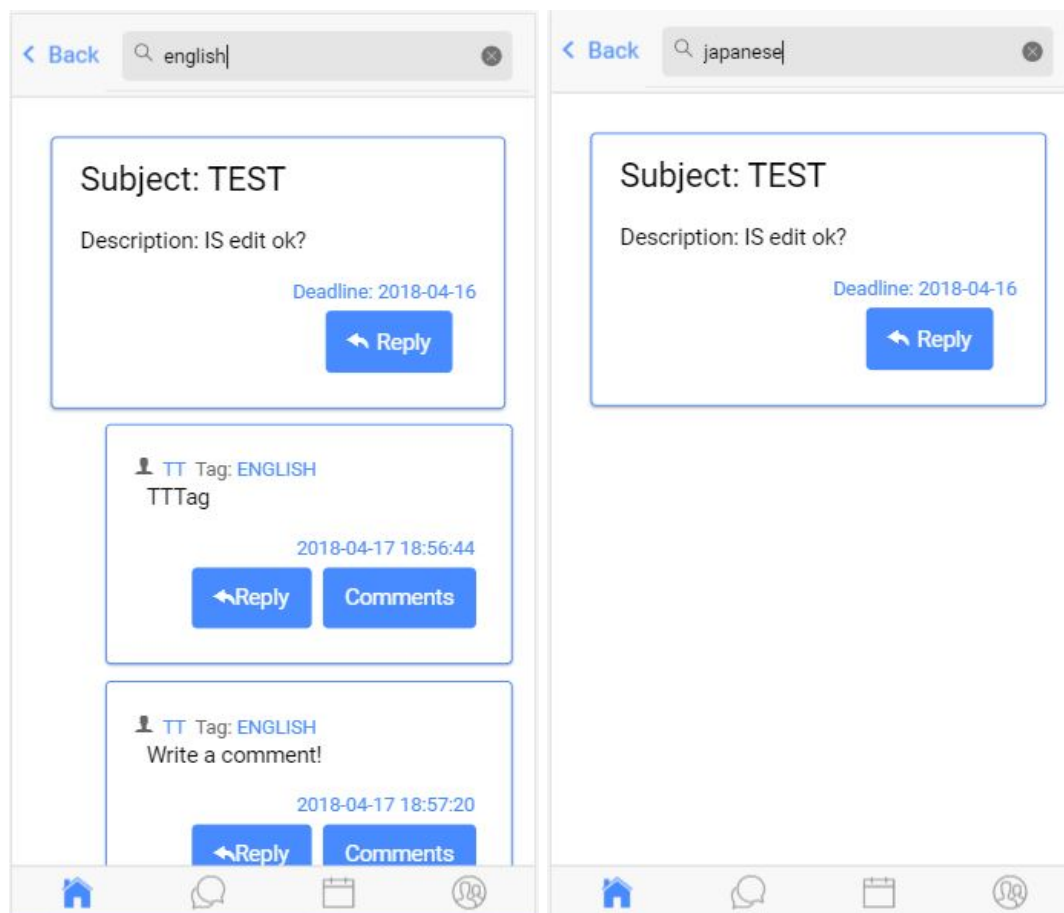


Figure 14: Search

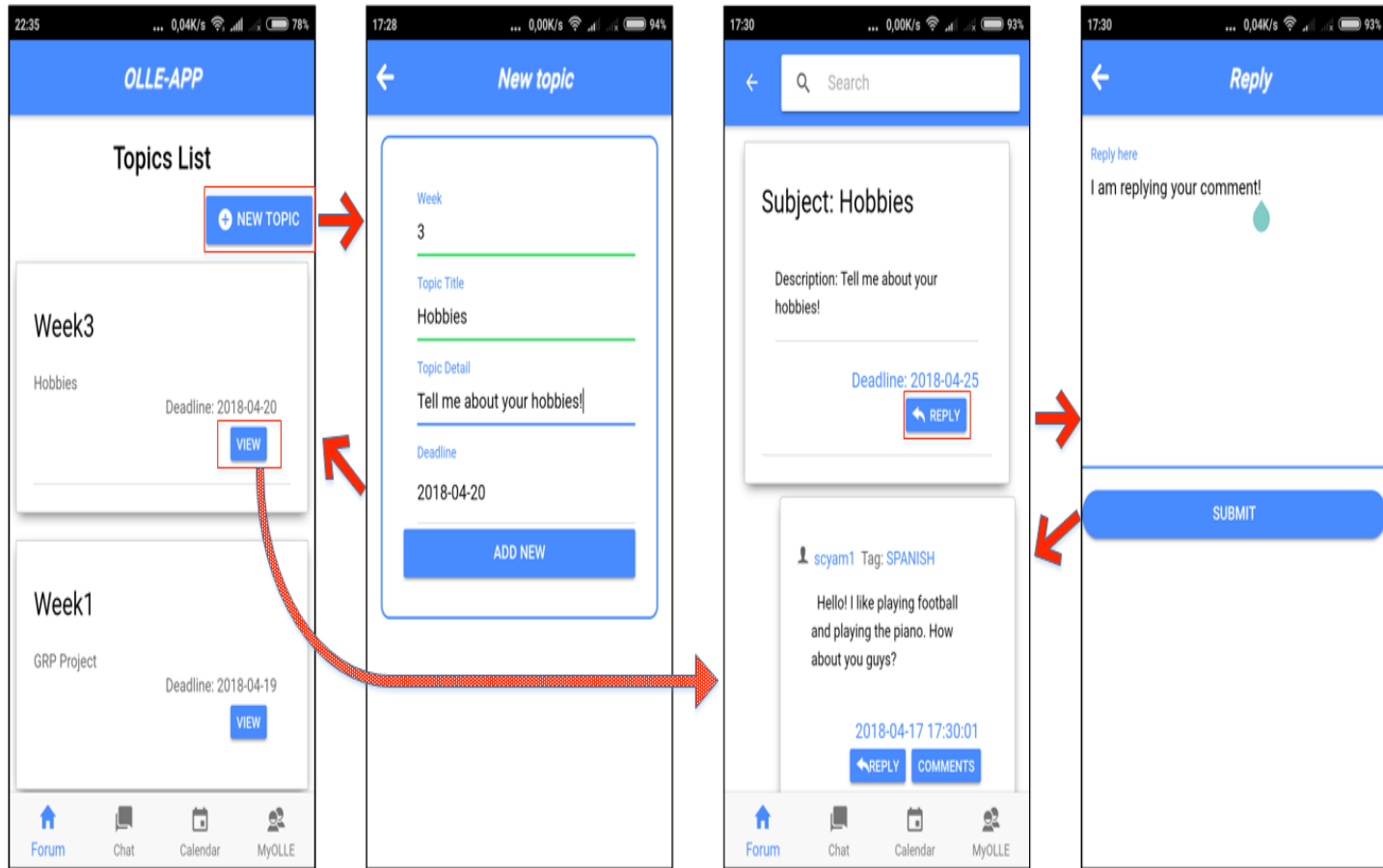


Figure 15: Overall view

## 4 Calendar

### 4.1 Check an event (Administrator, normal user)

To check an event:

1. Click a date.
2. The detail of that day should be presented below the calendar.

#### NOTE:

A date which has an event is marked in blue.



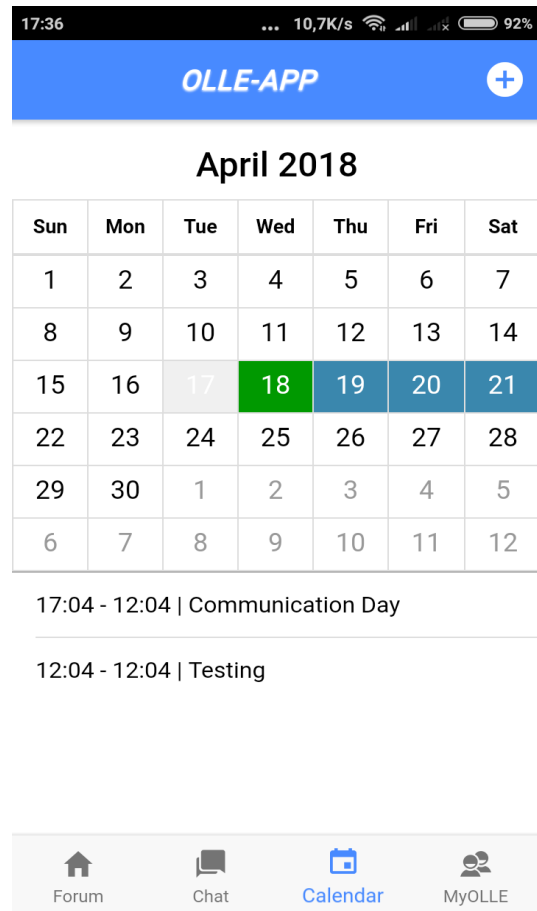


Figure 16: Calendar

## 4.2 Create an event (Administrator)

To create a new event:

1. Click the **add** button and the **New Chat Form** page appears.
2. Fill the detail of an event in the text field.

---

The screenshot shows a mobile application interface for creating an event. At the top, a status bar displays the time 17:36, network speed 10.8K/s, and battery level 92%. Below this is a blue header bar with a white back arrow and the text 'Event Details'. The form contains three sections: 'Title' with the value 'Communication Day', 'Description' with the text 'Let's go to Sunday Plaza abd promote our language learning skills!', and 'Start' and 'End' times. The 'Start' time is '04/17/2018 17:06' and the 'End' time is '04/21/2018 12:01'. At the bottom is a blue button with a white checkmark and the text 'ADD EVENT'.

17:36 10.8K/s 92%

← *Event Details*

Title  
Communication Day

Description  
Let's go to Sunday Plaza abd promote our language learning skills!

Start  
04/17/2018 17:06

End  
04/21/2018 12:01

✓ ADD EVENT

Figure 17: Create an event

3. Select a start time, then select an end time.
4. Click **OK** and then click **OK** to proceed after an alert dialog appears.
5. A new event should be created if you check that days event (Please refer to 4.1).

#### 4.2.1 Modify an event

To modify an event:

1. Click an event and the detail page should be presented.
2. Click **Edit**.

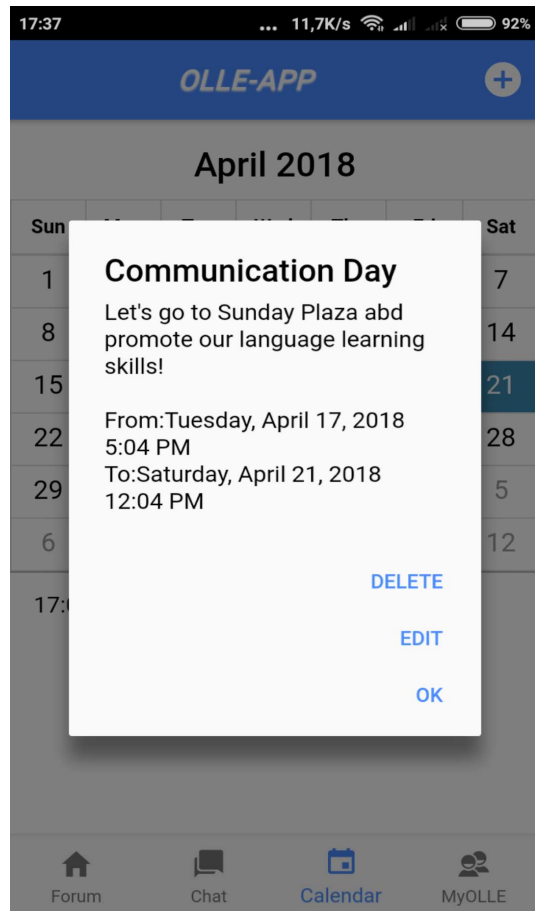


Figure 18: Modify an event

3. Fill in the forum.

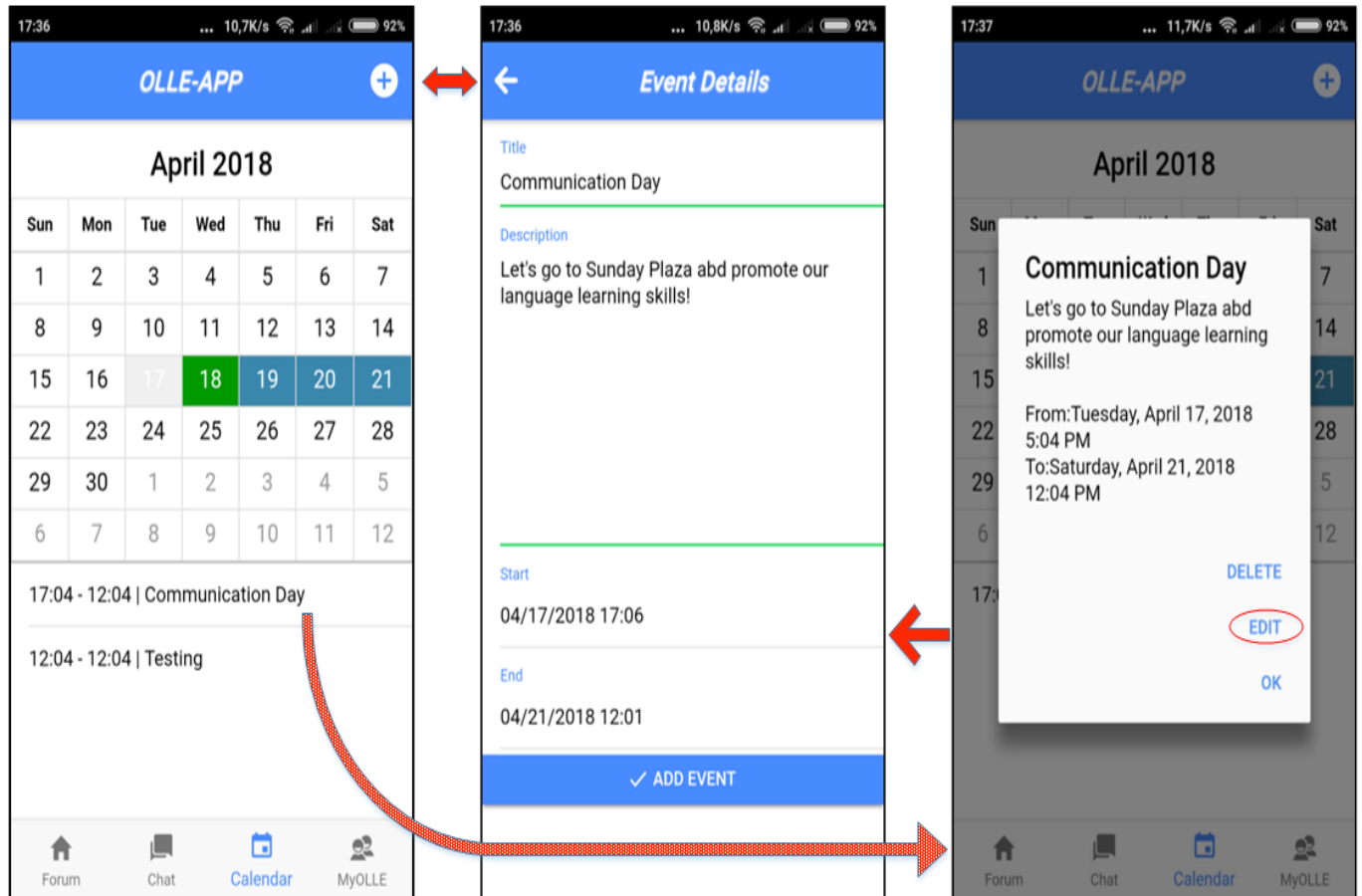


Figure 19: Overall view

## 5 Chat

This chapter includes the details of how an admin user manages chat rooms and how a normal user user chat rooms to chat with others.

### 5.1 Create a chat room (Administrator)

To create a new chat room:

1. Click the **add** button and the **New Chat Form** page appears.
2. Fill in the form.
3. A new chat room should be presented in the chat page.

---

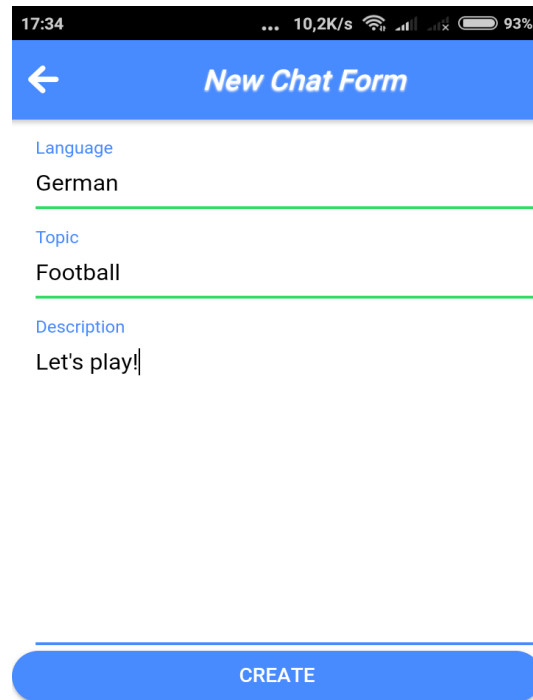


Figure 20: Create a chatroom

## 5.2 Delete a chat room (Administrator)

To delete a chat room:

1. Slide a chat room to left and click **Delete**.
2. Once an alert dialog appears, confirm the deletion by click **Proceed**, otherwise, click **Cancel**.
3. On the successful deletion, a toast with successful message should be presented.

**NOTE:**

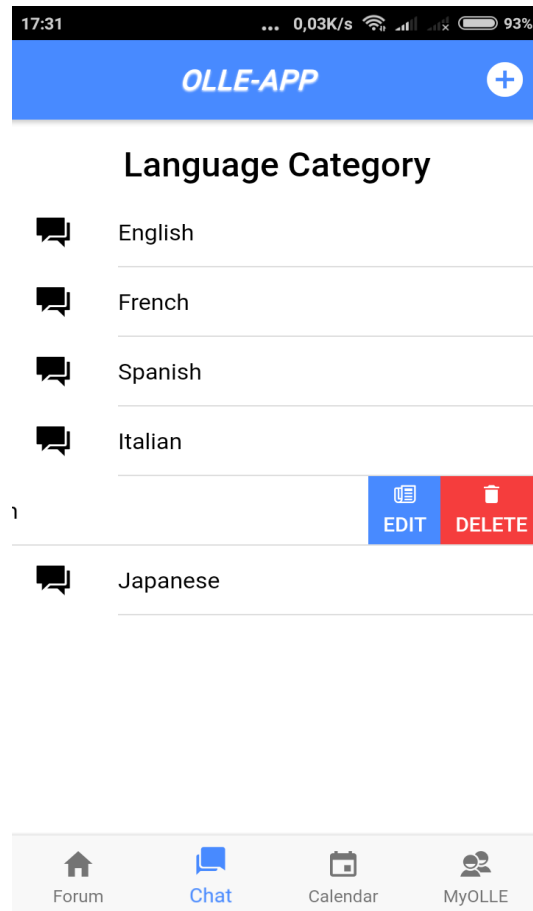


Figure 21: Delete a topic

If deletion a chat room is not successful, a toast with error message should be presented.

### 5.3 Modify a topic in a chat room (Administrator)

To modify a chat room:

1. Slide a chat room to left and click **Edit** and a **Chat Settings** page should be presented.
2. Fill in the form.
3. Click **Save** to confirm the modification.

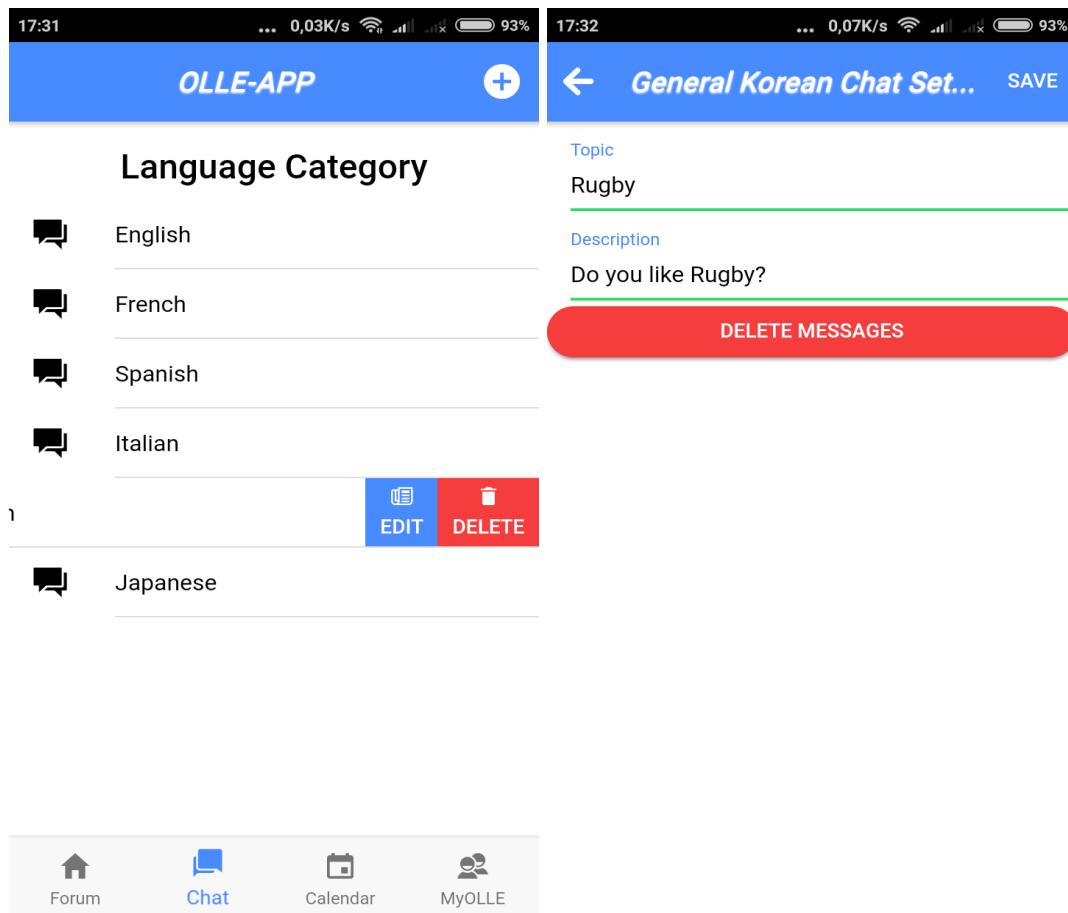


Figure 22: Modify a topic

#### 5.4 Delete all messages (Administrator)

To delete messages:

1. Slide a chat room to left and click **Edit** and a **Chat Settings** page should be presented.
2. Click **Delete Messages**.
3. Click **Proceed** to Proceed

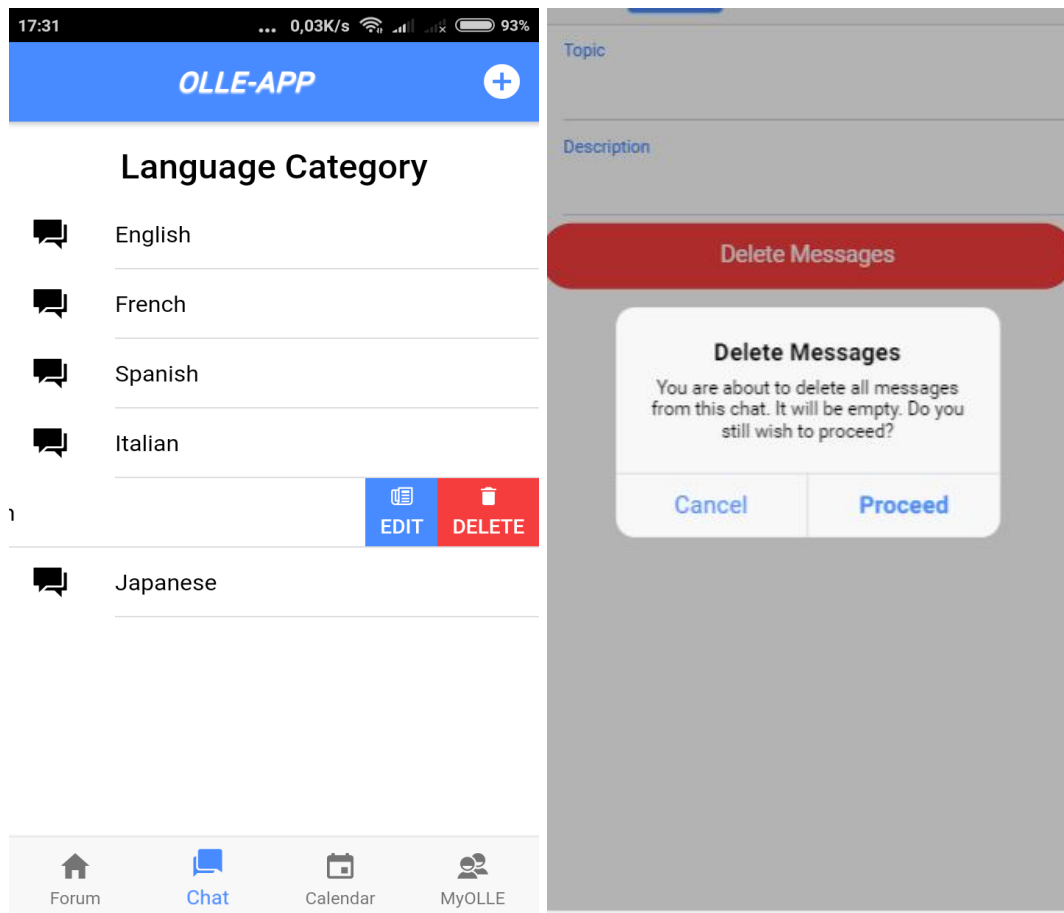


Figure 23: Delete messages

### 5.5 Send messages (Administrator, normal user)

1. Select a language group and enter it.
2. Scroll down the drawer, and a chat topic should be displayed.
3. Write a text in the text filed.
4. Click send icon to send a message.



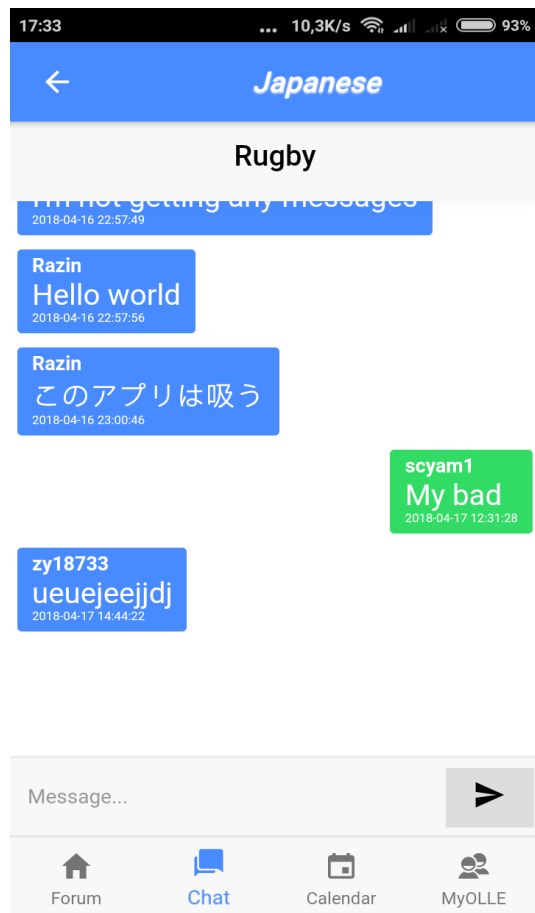


Figure 24: Send a message

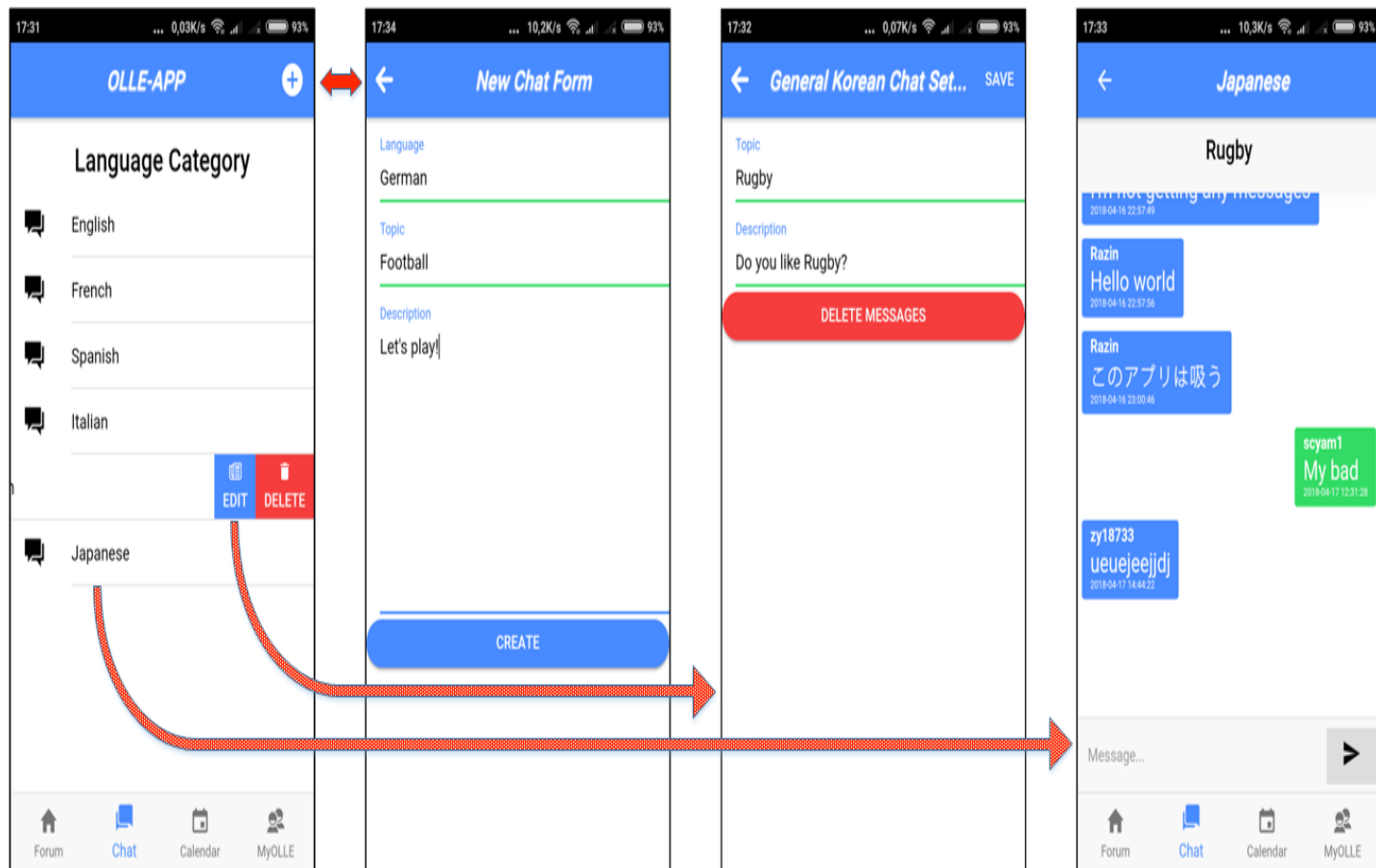


Figure 25: Overall view

## 6 Software environment

The following are the software environments used in the app.

<b>Operating Platform</b>	iOS, Android
<b>Server</b>	Tecent Cloud
<b>Other software</b>	PhoneGap

Table 2: Software environments used in the app