TOWARDS STREAMLINED CLUB MANAGEMENT: INTEGRATING COMMUNICATION AND MANAGEMENT IN A CENTRALIZED APPLICATION

Zhongshun Rao[1], Victor Phan[2]

[1]Tarbut V'Torah Community Day School, 5 Federation Way,
Irvine, CA 92603
[2]Computer Science Department, California State Polytechnic University,
Pomona, CA 91768

ABSTRACT

This project explored a solution to help with club management [1]. Club management has been a complex task for many people [2]. The solution to this problem is to create a centralized management application that allows all management and communication to be done on it [3]. The application is made with FlutterFlow and google firestore. It consists of two main parts, the communication and the management. Possible challenges are the users' experiences in features like chat and task management, but these will be improved in the future through updates. The application can be applied to all kinds of clubs like sport club or school clubs, providing communication and management for members. It is hoped that this application could provide users a convenient and effective way to organize their clubs.

KEYWORDS

club management, organizational efficiency, comprehensive platform, Mobile applications

1. INTRODUCTION

Club management has always been a complex task for many people. It requires club leadership to organize club events and communicate with other members. In order to make it easy, leaders use different apps to help them out from management. For example, using reminders as a communication platform while using cloud services to help manage tasks [4]. But this is not enough: leaders and members often have to switch between different applications and platforms which could be unorganized and inefficient. This problem is important because it could lead to serious consequences such as missing deadlines, forgetting about important events, having ineffective communications, and creating less cohesive clubs. This could make it harder for leaders and members to stay organized in the long run. According to a survey from the United States Census, 29% of girls and 24% of boys were involved in clubs in 2020, and that number is still increasing, showing that club management is a growing task for people [5].

Method A is a centralized management and communication platform system. The method the system used is the Kanban approach [6]. This approach provides customized experience for users. One disadvantage of this system is the user's experience. Also, it doesn't have a communication system. The second method is also a centralized club management system. One disadvantage of the system is that the system doesn't have a group chat system, which limits its communication.

David C. Wyld et al. (Eds): CMLA, GRAPH-HOC, CIoT, DMSE, ArIT, WiMoNe, CSEIT – 2023
pp. 123-131, 2023. CS & IT - CSCP 2023 DOI: 10.5121/csit.2023.1310011

The last method is a management system intended for Cricket Clubs. It helps with many things for a cricket club. However, it doesn't help with other types of clubs' management. My project improved on these works by combining the management part and the communication part into a single application, creating a real centralized platform that ultimately helped with club management.

My solution to this problem is to make an application that combines all the necessary features club leaders need for management. This will solve the problem because people no longer need various softwares for different purposes in club management which could be unorganized and inefficient. Instead, one platform with all the functionalities that are necessary for club management helped people stay organized. Currently there exists some other ways to help club management. For example, many people use Gmail to serve as an announcement tool [7]. Discord is also a good app for communications of the club, and there are other things similar to them. But one problem with these softwares is they only served for one or two purposes in club management. Meanwhile, my solution will solve the problem from its root and bring more convenience for club leaders to manage their clubs.

In my experiment, I set up a survey and invited some people to reflect on their experience after using the app. The four questions explored users' experience on communication, management, and navigation. It also asked if the user would recommend this to a friend. The average score of the survey is 2.83. The highest score for the chat feature is 3, and the lowest score is 2, this indicates that the app's chat feature could be improved given more refinements. The highest score for navigating is 4, and the lowest is 3. This result shows that the app has a clean interface. The average score of users' interest in recommending the app to other people is 3. The average score for management is 4, and the lowest is 1. This indicates that the app could potentially help people with their management. Overall, it is a decent result for a new application and shows space for further improvement.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Differentiate People's Messages

Chat components could have some challenges when implemented. One challenge of the chat component is that it is difficult to differentiate people's messages. For example, the program won't know if the message is sent by the user or other people. To solve this, we can have each user maintain a document collection in our database of their own messages with their own IDs. Another challenge is how to get an invite system working. Our implementation of the invite system would use multiple collections: one for the chat itself, one that maintains references to chats that specific users accept, and one that maintains outgoing invites sent to users [10].

2.2. Permission System

Another challenge is related to the permission system [8]. Obviously, not everyone should be able to edit the club and invite people. Only trusted members of a club's leadership board should be able to do that. A permission system would be a good solution to this challenge. Basically, we create some lists for each permission, such as a list for making announcements, meetings, and more. When the club creator gives someone a new permission, the person's name is added to the corresponding list. Finally, the app will display the appropriate controls to the club page

according to the lists. This system would allow for granularity with how much control a given person has over a club.

2.3. The Club Invitation Component

The club invitation component is also very important. We want to find an easy and fast way for users to invite others into their club. There should not be too many procedures to invite someone to the club. The solution to this challenge is quite simple. Since every user has their unique email when registering, the app can search through the profile document and check if any email matches up. After the user submits the email of the user they want to invite, the system will display the user's information on the screen if the account exists and the user can invite the person into their club. This implementation is simple and easy to use.

3. SOLUTION

The user will first log in to the app. Then, they will be navigated to a homepage in which they can choose to visit a club, edit their profile, or access the chat. If users decide to chat, they will have a chat selector in which they can choose the chat they want to go to or request chat members, or accept chat invitations. If a user decides to visit a club, they can pick which club they want to go to and they will be at the homepage of the club. There people can choose to see announcements, upcoming meetings, other members, and tasks. If users are managers or other leadership roles, they can post meetings and announcements, assign tasks, invite people to the club, and also manage permissions for members. The 3 major parts of the program are 1) Chat features, 2) Club Creation, and 3) Club Management. These components are linked to each other closely and create a friendly and convenient environment for leaders and members, providing them with everything they need for club management. I used FlutterFlow as my platform to create this program.

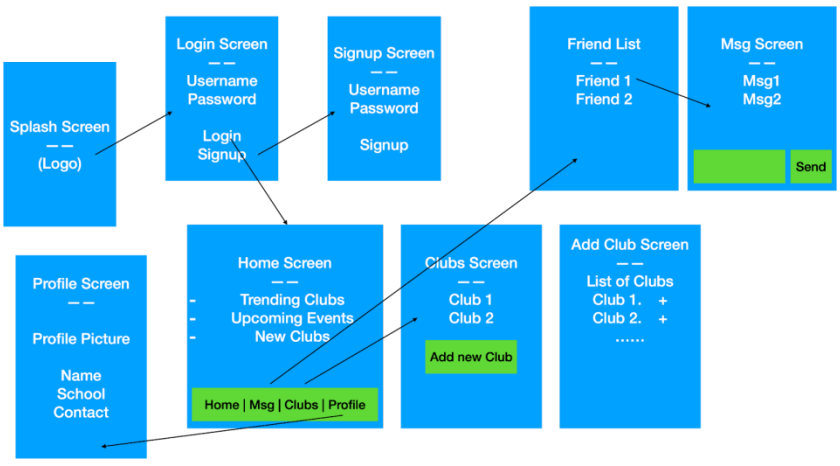

Figure 1. Overview of the solution

The purpose of the chat feature is to provide communications among club members [9]. The chat feature relies on firestore documents to work. Firestore documents are sets for fields [12]. They are very useful when it comes to storing data. In this case, the chat feature utilized firestore documents for storing users' group chats and private chats.

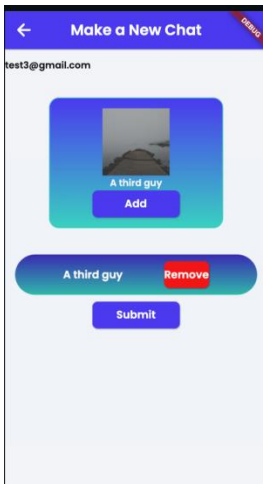

Figure 2. Screenshot of making a new chat

```
onPressed: () async {
  final chatsCreateData = {
    'members': FFAppState().groupmembers,
  };
  var chatsRecordReference = ChatsRecord.collection.doc();
  await chatsRecordReference.set(chatsCreateData);
  _model.createdDocument = ChatsRecord.getDocumentFromData(
      chatsCreateData, chatsRecordReference);
  final chatsUpdateData = {
    'members': FieldValue.arrayUnion([currentUserReference]),
  };
  await _model.createdDocument!.reference.update(chatsUpdateData);
  final invitesCreateData = {
    ...createInvitesRecordData(
      sender: currentUserReference,
      chatRef: _model.createdDocument!.reference,
    ),
    'recipients': FFAppState().groupmembers,
    'members': FFAppState().groupmembers,
  };
  var invitesRecordReference = InvitesRecord.collection.doc();
  await invitesRecordReference.set(invitesCreateData);
  _model.createdinvite = InvitesRecord.getDocumentFromData(
      invitesCreateData, invitesRecordReference);
  final invitesUpdateData = {
    'members': FieldValue.arrayUnion([currentUserReference]),
  };
  await _model.createdinvite!.reference.update(invitesUpdateData);
  final userChatsCreateData = createUserChatsRecordData(
    chatref: _model.createdDocument!.reference,
    user: currentUserReference,
  );
  await UserChatsRecord.collection.doc().set(userChatsCreateData);
  context.pop();
  setState(() {});
},
```

Figure 3. Screenshot of code 1

This is the action chart of sending chat invitations to users. Action one will create a chats document in the chats collection. Then it will set users in the group members list, which is a list of references to the user's profile documents. Action two will get a reference to the document made in Action one and add the creator of the chat into the group members list. The creator will add a reference to their profile document. Action three will create a document in the invite collection. The sender is the reference to the creator's Profile documents. The recipients, and members of the invite are the group members. Chat_ref is set to the document reference of the chat document. Action four will access the document made in Action three a d add the creator to the member list. All the members in the member list will have a reference to their profile document. Action 5 creates a document in the user chat collection, and chat_ref will become a reference of the document. This document contains people who accepted the invitation. In this case it will automatically add the creator of the invitation into the document and make the user set to the creator's profile document. The process is complete, so we navigate back to a screen.

The club creation component's purpose is to allow users to create their own clubs. The club creation also relies on firestore documents to work. Firestore documents are sets for fields. They are very useful when it comes to storing data. In this case, it is the information of the clubs that people create.

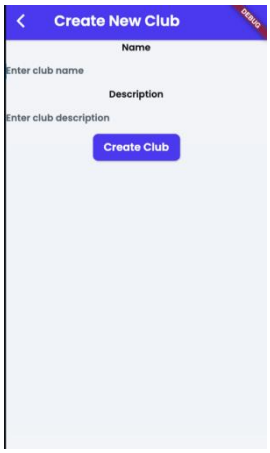

Figure 4. Screenshot of creating new club

```
onPressed: () async {
  final clubsCreateData = {
    ...createClubsRecordData(
      name: _model.textController1.text,
      description: _model.textController2.text,
    ),
    'members': [currentUserReference],
    'canMakeAnouncement': [currentUserReference],
    'canMakeTask': [currentUserReference],
    'canMakeMeetings': [currentUserReference],
    'canAddPermission': [currentUserReference],
    'canInvitePeople': [currentUserReference],
  };
  var clubsRecordReference = ClubsRecord.collection.doc();
  await clubsRecordReference.set(clubsCreateData);
  _model.clubDocument = ClubsRecord.getDocumentFromData(
      clubsCreateData, clubsRecordReference);
  final profilesUpdateData = {
    'clubs': FieldValue.arrayUnion([_model.clubDocument!.reference]),
  };
  await currentUserReference!.update(profilesUpdateData);
  context.pushNamed(
    'ClubPage',
    queryParams: {
      'club': serializeParam(
        _model.clubDocument,
        ParamType.Document,
      ),
    }.withoutNulls,
    extra: <String, dynamic>{
      'club': _model.clubDocument,
    },
  );
  setState(() {});
},
```

Figure 5. Screenshot of code 2

The club creation page has three actions. Action one will create a firestore document. It stores it in a collection of clubs. The document will include fields such as name as the club name, description, and members. Members is a list of profile references. We add the creator's profile reference to it. We also set privileges, which are canMakeAnnouncement, canMakeTask, canMakeMeetings, canAddPermission, canInvitePeople. All of these are lists of user profile references. For all of these we add the creator's profile reference to it. Action 2 will get a reference to the document we previously made. Then we will change the creator's list of clubs to include this reference so that the creator can access this club. The last action will navigate the creator's page to the club page. Then it will load the page according to the club document. This way the creator can immediately see the new club.

To invite new users to a club, club managers only need to concern themselves with the emails of desired users. Every user that signs up through the application has a unique email that managers can use to search for specific users. This approach relies on looking through profile records that are automatically generated when new users sign up.

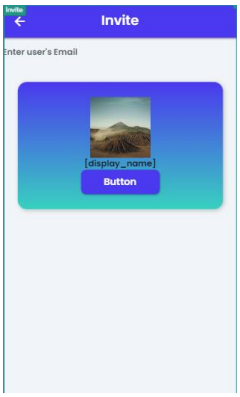

Figure 6. Screenshot of inviting

Figure 7. Screenshot of code 3

Figure 8. Screenshot of code 4

One of the important club management features is the club invitation. The club leadership will input an email of the person that they want to invite to their club in a textfield. The email will be saved into a variable called inviteduser. Then, the code will query a single document from the profiles collection. These profile documents are automatically made whenever a user signs up. Then, there is going to be a filter that will check if the email matches the email inside the invited user's profile documents. When the email matches, the app will return a document of that user and it will be unique since there should only be one user with that email. Then, we will display the profile picture and name from the document. If the invited user accepts the invitation. Their information will also be added to the member list of the club.

4. EXPERIMENT

The program's utility is limited if its target audience cannot use it and does not enjoy using it either. It is important, because this being a social media app requires that its user base is able to understand its functionalities and effortlessly go through the menus to maintain their clubs.

My experiment plan is to have people complete a survey that reflects their opinions and experience on using the app. The four questions are:

How easy is it to navigate the app?
Would you recommend this to a friend?
How easy is it to manage a club using this app?
How easy is it to set up a conversation with someone else?

The first question and second reflect how easy it is to use the app, which is related to the experience of users. It also shows how satisfying the user is after using the app. The third question tested how effective the club managing functions, and the last question tested how effective is the communication function of the app.

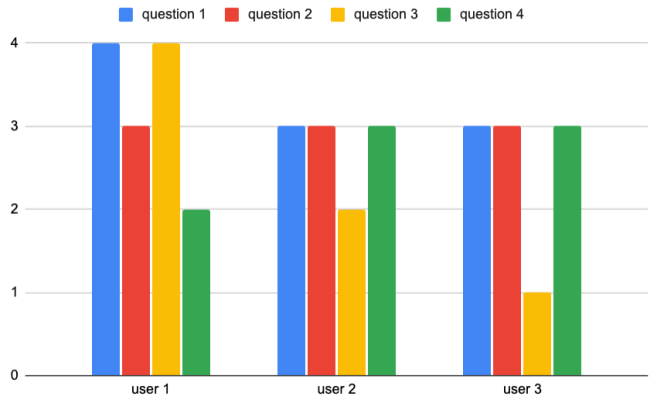

Figure 8. Figure of experiment 1

The average score for the first question was a 3.33. The highest score was a 4. The lowest was a 3, possibly because there is no home button for easy access so it is not easy to access another club. The average score for the second question was a 3.0. The highest score was a 3, which indicates some interest that could be improved given more refinements. The average score for the third question was a 2.33. The highest score was a 4, and the lowest was a 1. This is likely because the chat feature needs more refining compared to popular texting applications like Discord or WeChat. The average score for the fourth question was a 2.67. The highest score was a 3, and the

lowest score was a 2, likely because as previously mentioned, the chat feature may not be completely refined. This is an initial version of the application, however, so as a start this is generally an indicator of consumer interest. More updates to the app that give it more features over time would likely cause these figures to increase.

5. RELATED WORK

One method A. Rahmat and N. M. Hanifiahtried did is to create a centralized management and communications platform system for clubs by using the Kanban approach [13]. This approach allows clubs to use their own methods for management. According to their testing, the system is easy to use. However, the user's experience in using the system could be challenging. Also, this management system does not have the communication component of a club management system, which is a crucial part of a centralized club managing platform.

Another method Noraini binti Joharitried is to create a centralized management system for clubs because it is difficult to store and manage the information manually [14]. By creating a club management System, students are able to get information faster and more efficiently. The system also includes features like forum, message, and messages boards.These features further helped students' club with their communication. Their solution is similar to mine, But one difference is that their system has a forum feature instead of group chats.

One method R. Vidya Pradosh, R.Saran, and S.Vasanth created a Cricket Club Management System [15]. The system helps the user with booking ground, applying membership, and registering for training batches. The system helps admins maintain the database for the club and make it more effective. One difference between their method and mine is that my method focuses more on the general club management and provides communication, which is an essential part of club management systems.

6. CONCLUSIONS

Some limitations to my project is that it needs some additional features to further improve users' club management experience. For example, many clubs send out surveys to collect club members' information. This is a crucial part of club communication and it will be implemented into my project in future updates. Another feature that will be implemented in the future is the check-in module, which is an important element of member management. Club leaders can then take attendance for club events [11]. One other feature that could potentially be implemented is a resource page. A resource page can help navigate members to important contents and organized files too. I would use flutterflow to update these features to my project. The surveys and resource page will be accessible through the users' club home page, and the check-in page will be accessible through the events in the meeting page.

Clubio is a solution to the problem that it could be difficult to organize the club using different applications. It merges all the component club management in one single platform, making it easy for users to manage their clubs and communicate between members.

REFERENCES

[1] Ibrahim, Mohd Shukri, et al. "Information technology club management system." Acta Electronica Malaysia 2.2 (2018): 01-05.

[2] Thiel, Ansgar, and Jochen Mayer. "Characteristics of voluntary sports clubs management: A sociological perspective." European sport management quarterly 9.1 (2009): 81-98.

[3] Wright, Sue Ellen, and Gerhard Budin, eds. Handbook of terminology management: application-oriented terminology management. Vol. 2. John Benjamins Publishing, 2001.
[4] Siegemund, Frank. "A context-aware communication platform for smart objects." Lecture notes in computer science (2004): 69-86.
[5] Nam, Charles B., and E. Walter Terrie. "Measurement of socioeconomic status from United States census data." Measures of socioeconomic status. Routledge, 2021. 29-42.
[6] Al-Baik, Osama, and James Miller. "The kanban approach, between agility and leanness: a systematic review." Empirical Software Engineering 20 (2015): 1861-1897.
[7] Chen, Mia Xu, et al. "Gmail smart compose: Real-time assisted writing." Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2019.
[8] Fragkaki, Elli, et al. "Modeling and enhancing android's permission system." Computer Security–ESORICS 2012: 17th European Symposium on Research in Computer Security, Pisa, Italy, September 10-12, 2012. Proceedings 17. Springer Berlin Heidelberg, 2012.
[9] Kramer, Gerhard, Ivana Marić, and Roy D. Yates. "Cooperative communications." Foundations and Trends® in Networking 1.3–4 (2007): 271-425.
[10] Paulauskiene, Justina, et al. "The cost-effectiveness analysis of cervical cancer screening using a systematic invitation system in Lithuania." International journal of environmental research and public health 16.24 (2019): 5035.
[11] Beech, John, Simon Horsman, and Jamie Magraw. "Insolvency events among English football clubs." International Journal of Sports Marketing and Sponsorship 11.3 (2010): 53-66.
[12] Varshney, Heena, Ali S. Allahloh, and Mohammad Sarfraz. "Iot based ehealth management system using arduino and google cloud firestore." 2019 International Conference on Electrical, Electronics and Computer Engineering (UPCON). IEEE, 2019.
[13] Rahmat, Azizah, and Nur Aisyah Mohd Hanifiah. "Usability Testing in Kanban Agile Process for Club Management System." 2020 6th International Conference on Interactive Digital Media (ICIDM). IEEE, 2020.
[14] Johari, Noraini. Design and implementation of society and club management system in UiTM. Diss. Universiti Teknologi MARA, 2006.
[15] Sharma, Gaurav. Web Application Development for San Diego Cricket Club. Diss. San Diego State University, 2012.