AN INTELLIGENT AND DATA-DRIVEN MOBILE VOLUNTEER EVENT MANAGEMENT PLATFORM USING MACHINE LEARNING AND DATA ANALYTICS

Serena Wen[1] and Yu Sun[2]

[1]Lewis and Clark High School, 521 W 4th Ave, Spokane, WA 99204
[2]California State Polytechnic University, Pomona, CA, 91768

ABSTRACT

In Lewis and Clark High School's Key Club, meetings are always held in a crowded classroom. The system of event sign-up is inefficient and hinders members from joining events. This has led to students becoming discouraged from joining Key Club and often resulted in a lack of volunteers for important events. The club needed a more efficient way of connecting volunteers with volunteering opportunities. To solve this problem, we developed a VolunteerMatch Mobile application using Dart and Flutter framework for Key Club to use. The next steps will be to add a volunteer event recommendation and matching feature, utilizing the results from the research on machine learning models and algorithms in this paper.

KEYWORDS

Event management platform, volunteer support, machine learning, cloud computing

1. INTRODUCTION

Club meetings for LC Key Club [15] have always been held in a crowded classroom. The upcoming volunteer events would usually be written on the whiteboard, and members always had to jostle our way around each other just to reach the single event sign-up sheet. If someone missed a meeting, there was a low chance that they could find out what the events were or even get a chance to sign-up for them.

Members missing meetings often resulted in low volunteer numbers for certain opportunities that needed as many people helping out as possible. Key Club needed a more efficient system that would allow members to easily view and sign-up for whichever volunteering opportunities were coming up. It would benefit both the events our club members volunteered at by providing more volunteers and also giving club members greater opportunities.

One of the techniques that was proposed to solve the issue of missing meetings and missing the chance to sign-up for an event was the use of the Remind messaging app [16]. Remind allows school organizations to mass communicate with all their members. Each club or organization, such as Key Club, creates a channel with the club officers as the channel managers. Channel managers could send out separate messages on Remind, regarding club meeting time updates, and also the dates and details of volunteering opportunities. The issue with this system, unfortunately, was that when one club officer sent a message, it created a different message channel. So, with multiple messages regarding volunteer events showing up on different message channels, members could never see them all clearly in one place. When the pandemic interfered with Key Club's in-person meetings, club officers resorted to using the method of Remind to update members about events [1]. The confusing nature of multiple message channels further discouraged sign-ups.

In this paper, we present a new approach for the Key Club volunteer opportunity sign-up system. Our goal is to streamline the process of signing up for a volunteering event [2]. So, we have developed an application where, once they have signed up for an account, Key Club members can easily view volunteering opportunity details and sign-up for the upcoming events.

There are several other important, and useful features of this application. First of all, not only will club members be able to view the date, time, location, and other details of a volunteer event, they will also be able to see what other Key Club members have signed up for that event. Another feature of the app is the limit to the number of sign-ups for each volunteering event, which prevents any confusion from too many volunteers signing up. The application also features the ability for members to invite each other to specific volunteering events. Each club member has an Invites page where they can either choose to ignore or accept any pending invites to a volunteer opportunity. Finally, this application has two different types of user roles - admin and member. The admin role is assigned to Key Club officers, who have additional features to add, delete, or edit volunteering opportunities. This new approach is much better than the previous methods our club used because now, club members can see and sign-up for events all in one place. With this application, there will be no need to scrounge around various message channels or jostle around a crowded classroom just to find or put your name down on an opportunity. Therefore, we believe that this application will increase the number of Key Club members signing up for events because club members can do it wherever is convenient for them.

Our related experiment evaluates the factors needed in the process of building a volunteering event recommendation engine for the application users. This recommendation engine is part of a future feature of the application that will recommend certain opportunities based on the user's profile.

The rest of the paper is organized as follows: Section 2 gives details on the challenges that we met during the design and development of the application and a subsequent experiment to determine details of a future recommendation feature of the app; Section 3 focuses on the details of our solutions corresponding to the challenges that we mentioned in Section 2; Section 4 presents the relevant details about the experiment we did, following by presenting the related work in Section 5. Finally, Section 6 gives the concluding remarks as well as pointing out the future work of this project.

2. Challenges

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

2.1. Deciding the logistics of the application's features

The first challenge of the application development was figuring out the specific logistics of features, such as the invite other members feature [4]. The initial idea was quite simple: members would be able to search up other members and invite them to a volunteer event. As we delved further into coding this feature, however, we realized there were some important questions that needed to be answered. Would users only be able to invite their friends to an event? Would this mean we needed to implement a "friends" system on the app? We also had to consider the complexity of the solutions, as ideas that involved more extensive coding would take much longer to implement, and we were in a bit of a time crunch [3]. Eventually, after discussing with other Key Club officers, we came up with a simpler solution to the invite feature. Users could invite any of the other users to an event, and users that got invited would be able to view who invited them. Other questions regarding the logistics that still need to be considered involve how the app will be reused in future, subsequent years, especially since the roster of Key Club members changes each year.

2.2. Designing the UI

Another challenge we struggled with was designing the UI to be the most user-friendly for each of the features the app was meant to include [18]. We had to learn the Dart programming language and understand how we could use various objects provided by Dart to make the screens look like how we wanted them to look [17]. Whenever we got an idea for a page's design, research had to go into seeing which Dart widgets could be used to produce that design.

It was a challenge to design the Home page, especially because it contained the most important information - the list of volunteer opportunities. The Home page design planned for a section each for all the events and for the user's events, but an obstacle we had to overcome was deciding which way to present the information that would make it most user-friendly.

There was lots of debugging that went into the UI code. For example, when designing the bottom navigation bar to be different for users under the role of admin, the Dart code became finicky and it took us a while to work through the problem [5].

However, UI design is important because it has to be user-friendly and make those who download the application want to actually use it.

2.3. Connecting each screen to the database

A third challenge we encountered while developing the application was when we had to connect each of the application screens to the back-end database [7]. This was a significant challenge to overcome because it was key to making the app function correctly. The app needed to be able to communicate data with the database and also update important sections when actions were taken on the front end. The database had two main sections of User data and Events data, and their interconnectedness had to be taken into account [6]. For instance, whenever a user signs up for an event, the event has to be added to the User data and the Events data has to be updated with the user's info. There were also many Null pointer errors and such that came up in the process of coding and testing that had to be understood and addressed. Additionally, important data had to be saved/sent whenever the application page was switched. For example, from the Home Page to the Details page of a certain event, we had to ensure that the information about which specific event it was, was sent to the next screen.

3. SOLUTION

The LC Key Club Application provides an efficient system of updating, viewing, and signing up for Key Club members. Its main features are on the Home page, Event Details page, Invite and Pending Invites page, and the additional Edit/Add Events pages for accounts assigned to the Administrative role.

When users first download the app, they need to create an account by simply inputting some quick personal info and their intended username and passcode. Once logged in or signed up, club members can immediately view the upcoming volunteering opportunities on the Home page. In the event list on the Home page, the most important information regarding events - date and time- are already displayed. Additionally, the number of volunteer spots filled is shown.

From the Home page, Users can easily navigate to a specific opportunity's Details page. On the details page, more information is displayed, like the location of the volunteering event and also which other Key Club members have signed up for the event. Once a user has decided to sign-up for an opportunity, they can now invite other fellow Key Club members. Their name will also be added to the volunteer list and the event will now show up on the Homepage in the My Events section.

If a user has been invited to a volunteering event, a notifying icon will show up in the Invites tab on the bottom navigation bar. Here is where club members can view the volunteering opportunities that they have been invited to along with who has invited them. Users can choose to accept or ignore the invite and the pending event will be removed from the page.

One final significant component of the application is the ability of users assigned the Admin role to add new volunteering events and edit or delete current opportunities. The following diagram illustrates the connections and relationships between each screen of the application.

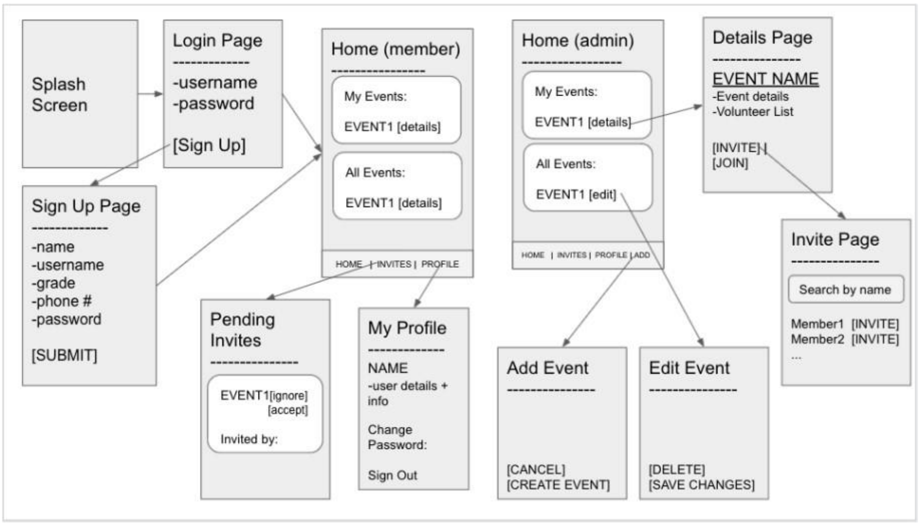
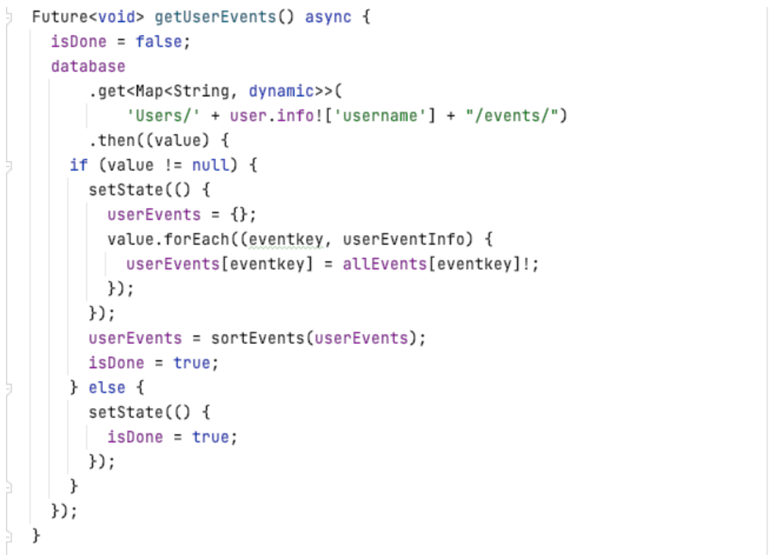


Figure 1. The overview of the project

In the next section, the code implementation of each component is discussed.

Component 1 - Home Page

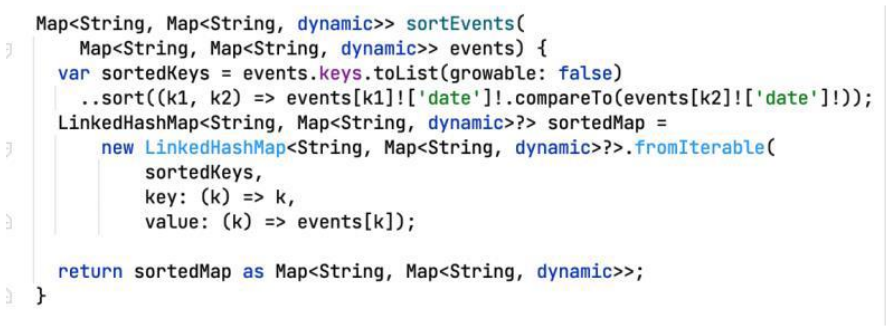
The most important code of the Home Page component was the retrieval and organization of volunteering events data from the database. The getAllEvents() method first gets the information of each event from the database. From there, the event data is stored in a map that is later iterated through in a ListBuilder to display on the application screen. The getUserEvents() method takes the information from the newly created allEvents map to gather a list of events that the user is currently signed up for. Rather than going through a complicated process of iterating through all the allEvents items and searching if the user is listed as one of the volunteers, this function only iterates through the specific user's data. Then, the event keys that are listed in the user's current events are assigned to a new userEvents map item, with the associated allEvents map information. This allows for efficient database searching. We already know that whichever events the user is signed up for will be in allEvents, so it is much easier to iterate through the user's events.



```
Future<void> getUserEvents() async {
  isDone = false;
  database
      .get<Map<String, dynamic>>(
          'Users/' + user.info!['username'] + "/events/")
      .then((value) {
    if (value != null) {
      setState(() {
        userEvents = {};
        value.forEach((eventkey, userEventInfo) {
          userEvents[eventkey] = allEvents[eventkey]!;
        });
      });
      userEvents = sortEvents(userEvents);
      isDone = true;
    } else {
      setState(() {
        isDone = true;
      });
    }
  });
}
```

Figure 2. Code of getUserEvents()

In both the getAllEvents() and getUserEvents() functions there contains the sortEvents() function. This important piece of code, as its name suggests, sorts the events, specifically in order of date with the events coming up the closest being higher on the list.

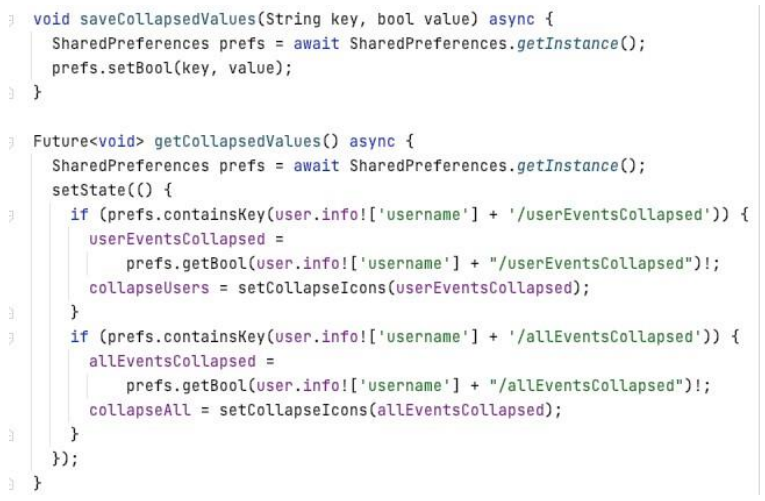


```
Map<String, Map<String, dynamic>> sortEvents(
    Map<String, Map<String, dynamic>> events) {
  var sortedKeys = events.keys.toList(growable: false)
    ..sort((k1, k2) => events[k1]!['date']!.compareTo(events[k2]!['date']!));
  LinkedHashMap<String, Map<String, dynamic>?> sortedMap =
      new LinkedHashMap<String, Map<String, dynamic>?>.fromIterable(
          sortedKeys,
          key: (k) => k,
          value: (k) => events[k]);

  return sortedMap as Map<String, Map<String, dynamic>>;
}
```

Figure 3. The code of sortEvents()

A challenge of the Home page UI was mentioned in Section 2, regarding the collapse function of the containers where the User Events and All Events are displayed. To solve this issue, we used the SharedPreferences object. Now, whenever a user switches away from the Home page tab and back, the collapsed or un-collapsed state of the MyEvents and AllEvents containers is saved and maintained. The code for the functions used can be seen in the figure below.



```
void saveCollapsedValues(String key, bool value) async {
  SharedPreferences prefs = await SharedPreferences.getInstance();
  prefs.setBool(key, value);
}

Future<void> getCollapsedValues() async {
  SharedPreferences prefs = await SharedPreferences.getInstance();
  setState(() {
    if (prefs.containsKey(user.info!['username'] + '/userEventsCollapsed')) {
      userEventsCollapsed =
          prefs.getBool(user.info!['username'] + "/userEventsCollapsed")!;
      collapseUsers = setCollapseIcons(userEventsCollapsed);
    }
    if (prefs.containsKey(user.info!['username'] + '/allEventsCollapsed')) {
      allEventsCollapsed =
          prefs.getBool(user.info!['username'] + "/allEventsCollapsed")!;
      collapseAll = setCollapseIcons(allEventsCollapsed);
    }
  });
}
```

Figure 4. Code of saveCollapsedValues()

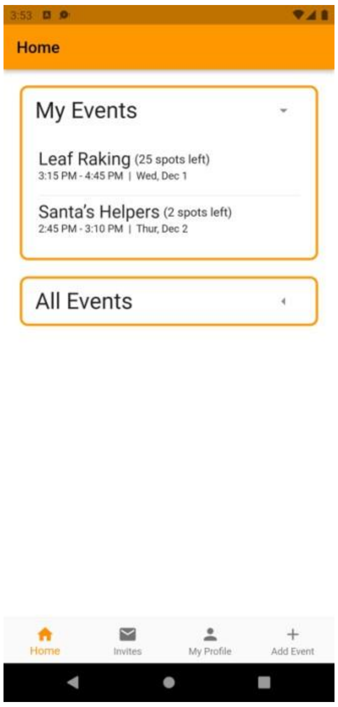

3:53
Home
My Events
Leaf Raking (25 spots left)
3:15 PM - 4:45 PM | Wed, Dec 1
Santa's Helpers (2 spots left)
2:45 PM - 3:10 PM | Thur, Dec 2
All Events
Home
Invites
My Profile
Add Event

Figure 5. Screenshot of App (1)

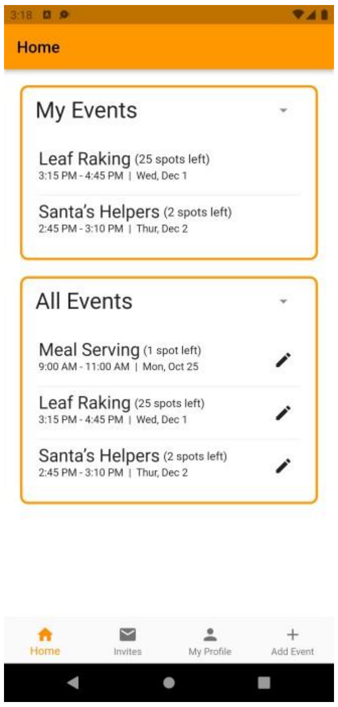
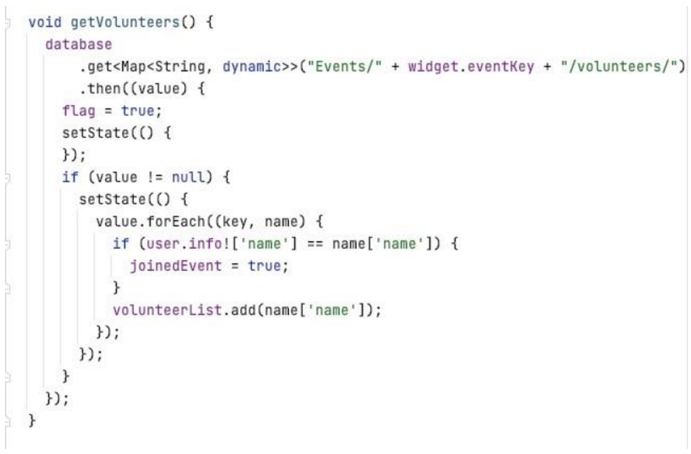

3:18
Home
My Events
Leaf Raking (25 spots left)
3:15 PM - 4:45 PM | Wed, Dec 1
Santa's Helpers (2 spots left)
2:45 PM - 3:10 PM | Thur, Dec 2
All Events
Meal Serving (1 spot left)
9:00 AM - 11:00 AM | Mon, Oct 25
Leaf Raking (25 spots left)
3:15 PM - 4:45 PM | Wed, Dec 1
Santa's Helpers (2 spots left)
2:45 PM - 3:10 PM | Thur, Dec 2
Home
Invites
My Profile
Add Event

Figure 6. Screenshot of App (2)

In the application screenshots above, the UI design of the Home page can be seen, where users can choose to collapse or uncollapse the My Events and All Events section. To reach the Details page for each event, users simply need to click on the event bar item. The clickable bar item was utilized by wrapping each event Container object with an InkWell, which allowed an “onTap()” function to be utilized.

Component 2 - Event Details Page

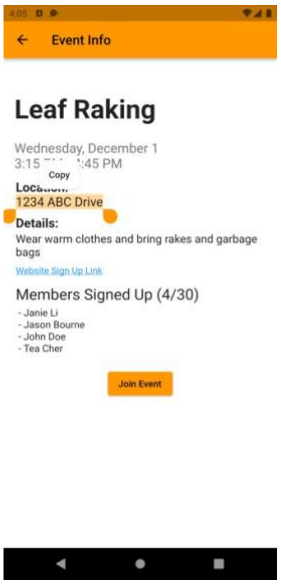
An important code component of the Event Details page was the function that retrieved the event’s list of volunteers from the database. The figure below shows how each volunteer name retrieved is placed into an outside volunteerList variable. In the UI build code, the volunteerList is iterated through and displayed as a list on the screen.



```
void getVolunteers() {
  database
      .get<Map<String, dynamic>>("Events/" + widget.eventKey + "/volunteers/")
      .then((value) {
    flag = true;
    setState(() {
    });
    if (value != null) {
      setState(() {
        value.forEach((key, name) {
          if (user.info!['name'] == name['name']) {
            joinedEvent = true;
          }
          volunteerList.add(name['name']);
        });
      });
    }
  });
}
```

Figure 7. Code of getVolunteers()

While the Event Details page displays which other volunteers are signed up, it also displays important additional information about an event, including location, items to bring, and an outside website sign-up link if needed.



Event Info

Leaf Raking

Wednesday, December 1
3:15 ... :45 PM
Copy
Loca...
1234 ABC Drive
Details:
Wear warm clothes and bring rakes and garbage bags
Website Sign Up Link

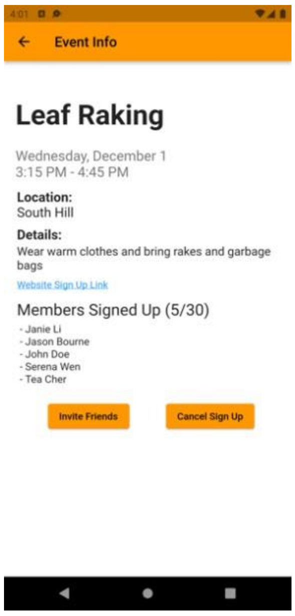
Members Signed Up (4/30)
- Janie Li
- Jason Bourne
- John Doe
- Tea Cher

Join Event

Figure 8. Screenshot of event info (1)

The text for the location is wrapped in a SelectableText, allowing users to copy-paste the address rather than having to type it out letter-by-letter when they are searching for directions. The website sign-up link is very useful for certain events that require its volunteers to sign-up officially using the organization's website. The clickable link only appears if a url has been inputted when the event was first created. A URL_launcher package was used to implement this.

On this page, users can sign-up for an event, invite friends or cancel their sign-up. Their name is added to the "Members Signed Up" list, and immediately displayed on the screen when they join.



Leaf Raking

Wednesday, December 1
3:15 PM - 4:45 PM

Location:
South Hill

Details:
Wear warm clothes and bring rakes and garbage bags

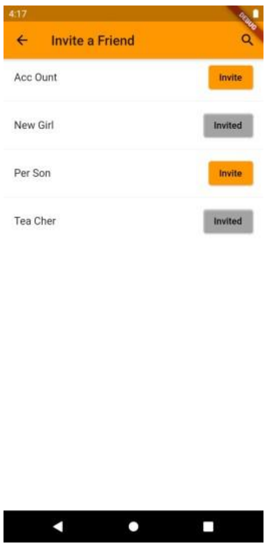
Website Sign Up Link

Members Signed Up (5/30)

- Janie Li
- Jason Bourne
- John Doe
- Serena Wen
- Tea Cher

Figure 9. Screenshot of event info (2)

The Invite Friends button leads to the Invite Friends screen page, where the getNames() function retrieves the list of Key Club members and then filters the list into only members that are not yet signed up for the specific event. Once a user invites another user using the “INVITE” button, the text changes to “INVITED”.



Acc Ount	Invite
New Girl	Invited
Per Son	Invite
Tea Cher	Invited

Figure 10. Screenshot of Invite a friend

Component 3 - Event Invites Page

In the second tab on the bottom navigation bar, there is the Pending Invites page. Here, as you can see in the screenshot image below, the events a user has been invited to is listed out with the options to ignore or accept the invitation. Users can also view who their inviters are (if there are multiple).

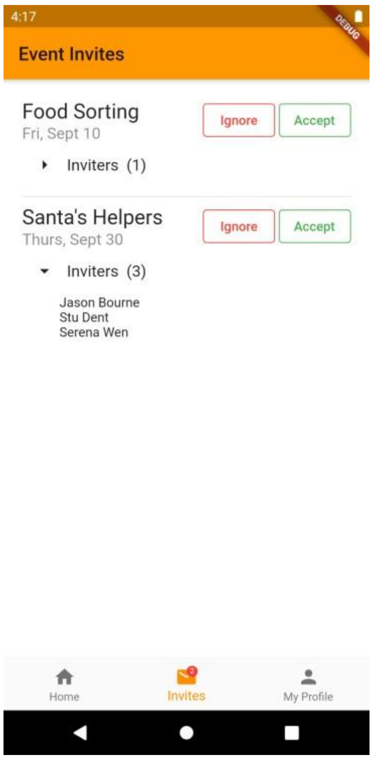


Figure 11. Screenshot of event invites

By pressing “Ignore”, the user activates a pop-up dialog box to confirm the ignore. If they choose to ignore, then the event will be removed from their Invite box. “Accept” will direct them to the Event Details page. We are working on implementing a function that will automatically be able to identify if an event is full and notify any users with pending invites. This will prevent confusion with invited users if they see they have been invited to an event that is already full.

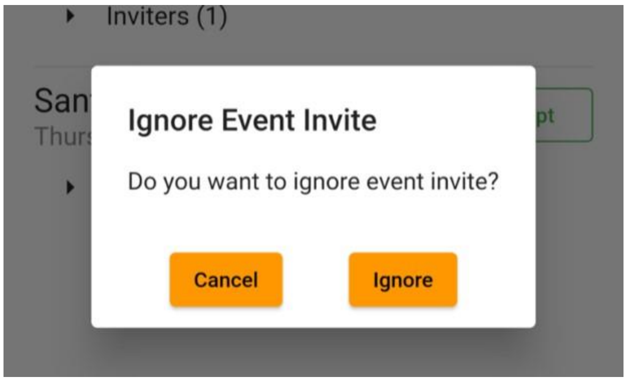


Figure 12. Screenshot of Ignore Event Invite dialog box

Component 4 – Administrative Role

Users with the admin role (reserved for Key Club officers), have the additional features to create a new event to add to the “All Events” section for every single user or edit the details of an event. The inputs include the event name, location, date, time, volunteer limit, and any additional details. The UI for inputting date and time is an easy-to-use selection input that doesn’t require any typing or specific formatting. Admins have a reserved tab on the BottomNavigationBar that directs them to the Create Events page [Figure 14]. There is also the Edit Event Info page that is accessed from the admin's Home page as seen in Figure 13.

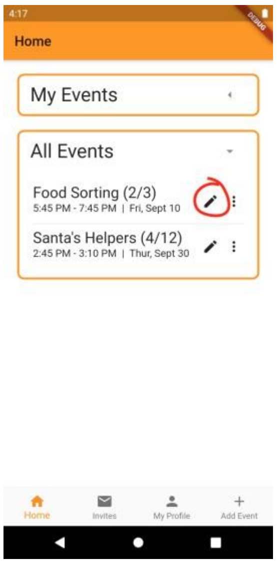


Figure 13. Screenshot of Admin Home page

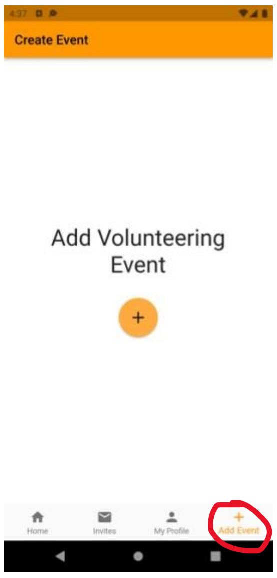
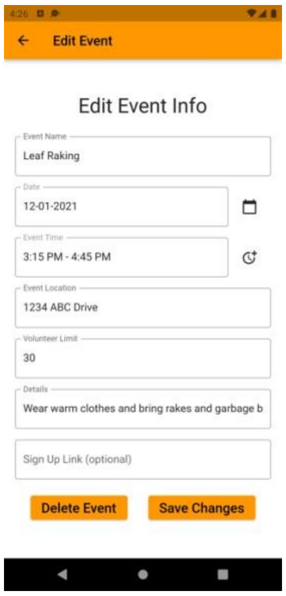


Figure 14. Screenshot of Admin Add Event page

The Edit Event Info pages and Create Events are very similar in content. In the screenshot of the Edit Event Info page below, you can see that the various attributes for an event also include the optional Sign-Up Link. Not all events will require an outside link to sign-up. If the box is left empty, an option to click a URL simply does not show up on the Event Details page.



Edit Event

Edit Event Info

Event Name: Leaf Raking

Date: 12-01-2021

Event Time: 3:15 PM - 4:45 PM

Event Location: 1234 ABC Drive

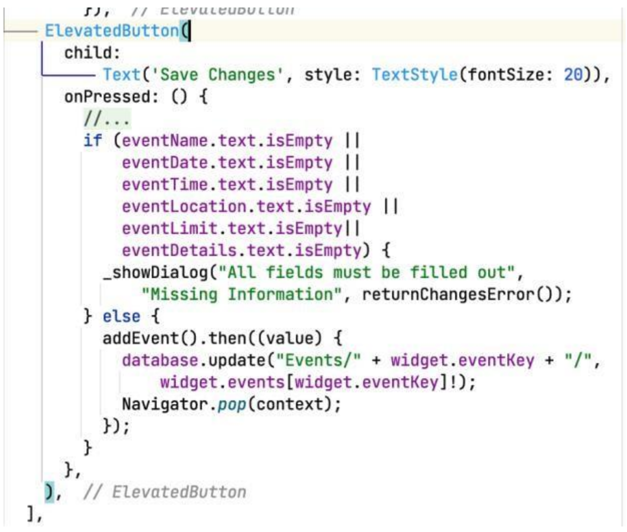
Volunteer Limit: 30

Details: Wear warm clothes and bring rakes and garbage b

Sign Up Link (optional)

Figure 15. Screenshot of Edit Event Info

Before any changes can be saved, we have implemented code that verifies that all required text fields have information in them (aside from the optional sign-up link)



```
    }), // ElevatedButton
  ElevatedButton(
    child:
      Text('Save Changes', style: TextStyle(fontSize: 20)),
    onPressed: () {
      //...
      if (eventName.text.isEmpty ||
          eventDate.text.isEmpty ||
          eventTime.text.isEmpty ||
          eventLocation.text.isEmpty ||
          eventLimit.text.isEmpty||
          eventDetails.text.isEmpty) {
        _showDialog("All fields must be filled out",
            "Missing Information", returnChangesError());
      } else {
        addEvent().then((value) {
          database.update("Events/" + widget.eventKey + "/",
              widget.events[widget.eventKey]!);
          Navigator.pop(context);
        });
      }
    },
  ), // ElevatedButton
],
```

Figure 16. The code of “Save Changes” button onPressed()

From the editing screen, admins can also use the DELETE EVENT button in case of an event cancellation [8]. One important thing to note is that the event data is not completely erased from the database. Instead, it is transferred to a “DeletedEvents” section of the database, just in case the event was not meant to be deleted.

4. EXPERIMENT

Overall, the purpose of the 4 experiments performed was to answer the question of how we should utilize the machine learning models to recommend volunteering events to users based on their profile. First, we decided which attributes (inputs) would be used to determine which volunteer opportunity to recommend to the user. The 5 attributes are gender, grade, main interest/hobby, and personal availability (based on morning, afternoon, evening). There were 6 possible output values for the preferred event type.

We then generated random dummy input data and partially random dummy output data in order to test the accuracy of each machine learning classification algorithm model [12]. The 4 models we used were Support Vector Machine (SVM), Random Forest Classifier (RFC), Gaussian NaiveBayes (GNB), and Stochastic Gradient Descent Classification (SGDC).

The main question we sought to answer with these experiments was how we should recommend events to application users based on their profile

4.1. Experiment 1 - Which machine learning algorithm provides the highest learning accuracy?

Answering this question of the experiment was necessary to decide which machine learning algorithm was the best to use. We tested each model with a dummy dataset size of 1000. For the RFC machine learning model, we used an RF depth of 10. The datasets were divided into training and testing data with which the models were fit and tested. The score was calculated for each model's accuracy and we gathered the average score data from 20 trials of dummy data. The graph below compares the average accuracy score for each model.

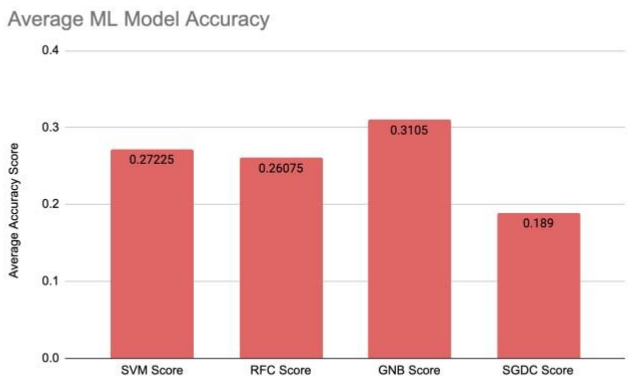


Figure 17. Average ML Model Accuracy

As shown in the figure, the most accurate was the Gaussian NB classification model with a score of 0.315 in comparison to the lowest score (from the SGDC model) of 0.189. Therefore, for the development of the recommendation engine in the future, the Gaussian Naive-Bays machine learning algorithm should be used to analyze the inputs.

4.2. Experiment 2 - Which algorithm parameters provide the highest accuracy?

While the previous experiment already showed the best learning accuracy in the Gaussian NB algorithm, We used the Random Forest Classifier model for this experiment, as it had the RF_max_depth parameter to work with [20]. We, again, used dummy data set sizes of 1000, split the data into training and testing, fit the data to the model, then calculated the score of 20 trials for each RF_max_depth. The values used were 1,2,5,10,15, and 30 and the results can be seen in the graph below.

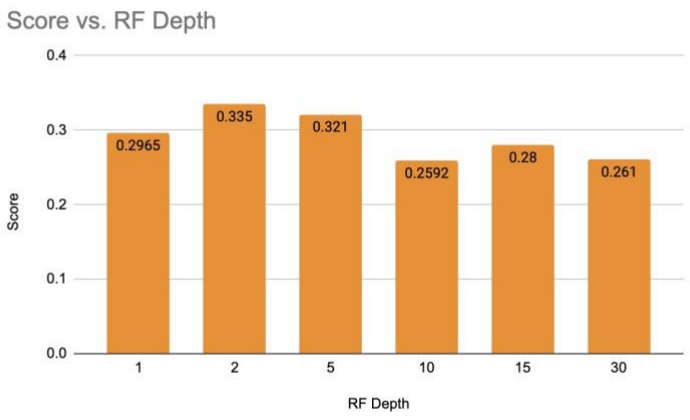


Figure 18. Score vs. RF Depth

Surprisingly, the RF depth of 2 provided the highest accuracy out of the trials performed. Interestingly, the learning accuracy appears to decrease as the max depth increases. This could be due to the fact that the partially random output data generation used a relatively simple line of reasoning to determine the output, and the RFC model did not need the greater depths.

4.3. Experiment 3 - How does the size of the dataset affect the accuracy?

The purpose of answering this experiment question was to further understand the factors that lead to the highest accuracy in training and testing the machine learning models. In this case, we studied how the factor of data-set size affected accuracy. We varied the data-set size with values ranging from 50 to 1000 and tested it for each of the machine learning algorithms. After going through the same process of generating dummy input and output data and training and testing it, we recorded the average accuracy score for each set size for each model from trials of 20.

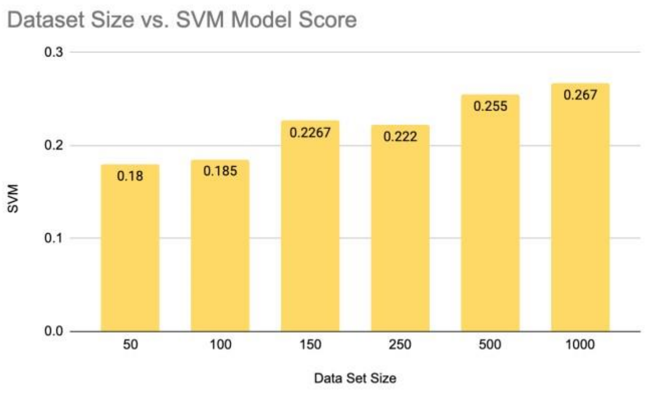


Figure 19. Data-set Size vs. SVM Model Score

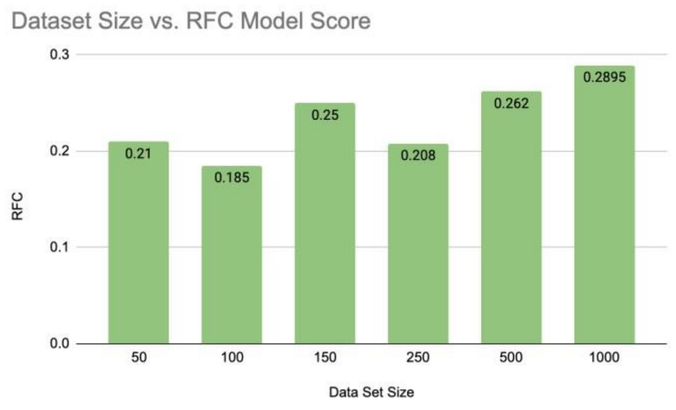


Figure 20. Data-set Size vs. RFC Model Score

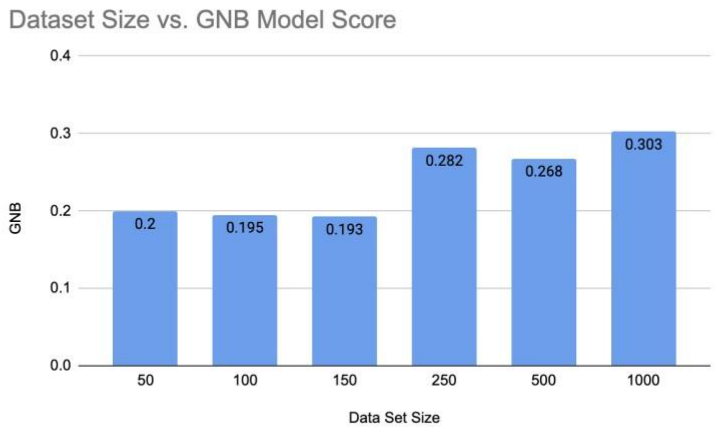


Figure 21. Data-set Size vs. GNB Model Score

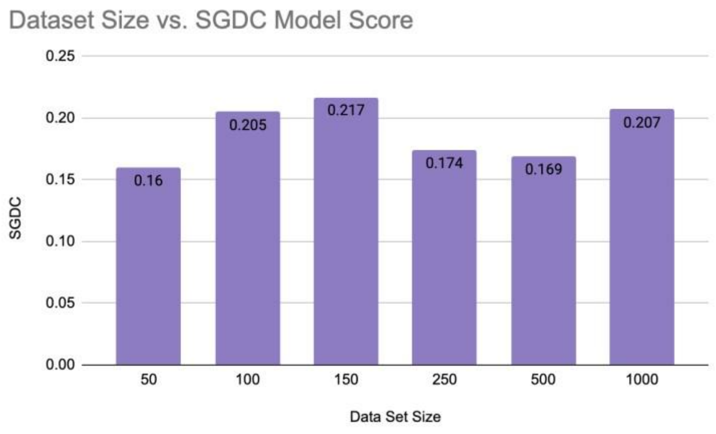


Figure 22. Data-set Size vs. SGDC Model Score

For the SVM model, its highest score was 0.267 at a data set size of 1000. For the RFC model, it had a score of 0.2895 and the GNB algorithm had a high score of 0.303, both occurring with the data-set size of 1000. Only the SGDC Model was out of pattern, with its highest learning score of

0.217 coming from the data set size of 150. The resulting figures above show that in general, the larger the data set, the higher the learning accuracy, with the highest accuracy being at a data set size of 1000.

4.4. Experiment 4 - What attribute input combinations will provide the highest accuracy?

To test this question, we used 6 different attribute combinations. One combination included all 4 attributes (Gender, Grade, Main interest/Hobby, Availability) and the others were a combination of 3 different attributes. The final combination was simply the interest attribute, as it held the highest weight in the partially random output data generator we had initially coded.

We utilized the Gaussian NB model since it had overall produced the highest learning score in previous experiments. The data-set size was 1000 and the average of 20 trials was taken.

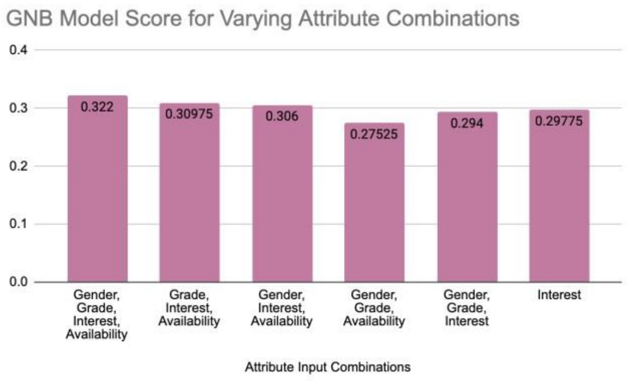


Figure 23. GNB model score for varying attribute combinations

As seen in the graph above, the combination that included all 4 of the input attributes produced the highest score of 0.322, though the other combinations were not too far behind. The lowest model score was 0.27525 with the combination that did not include the main interest/hobby. The single attribute combination of main interest produced a higher score of 0.29775. This follows along with the logic we took in the partially random generation code for output data, as we made it so the main interest input held the most weight. This is significant because it shows that the machine learning algorithm was able to recognize the pattern.

Drawing from the results of each experiment performed, the classification machine learning algorithm that would produce the highest accuracy for future development of a recommendation engine is the Gaussian Naive-Bayes model [19]. The model should be used with data sets of size 1000 and with inputs of all four of the attributes of a user: gender, grade, main interest/hobby, and availability.

5. RELATED WORK

Butgereit, L presented research on developing a virtual volunteering platform, specifically focusing on the characteristics that made up a successful volunteer platform [9]. Her research was based around the idea of an online tutoring platform that would open up volunteer opportunities for those who cannot always donate their time to volunteer physically.

Butgeriet's work is similar to the research we did here, as we are both trying to solve a problem related to lack of access to volunteering events/opportunities. One of the seven key characteristics the author discusses is "Easy Scheduling", which is also interconnected with the purpose of the application we built - to provide a simple platform where volunteers can view and sign-up for events.

Reuter, C, et al presented their research on the use of social media during a specific disaster, the 2013 European floods [10]. The paper's goal was to discuss how they used the research to develop a "cross-platform" social media application called XHELP. XHELP will help organizations coordinate volunteers and relief aid during disasters by compiling relevant data and information from social media in one place.

Both of our work involves understanding and helping coordination of volunteer activities but Reuter, C et al. is more about gathering information that would help the coordination.

Gudmunds, Emma etal.'s paper discusses their research on the barriers that prevent/hinder people from starting volunteering or "continue[ing] engagement" with it [11]. The paper proposed an application be developed that would provide a platform would-be volunteers could use to start their community service. Gudmunds' development of the app was focused on the user experience that would encourage them to search for opportunities.

Both of our research and methodology involve an application that lowers the barriers to volunteering opportunities, but this paper focuses on motivating potential volunteers to take action, while mine provides a means for them to take that action.

6. CONCLUSIONS

In this paper, we proposed a solution designed to address the barriers to taking part in volunteering opportunities provided at Lewis and Clark High School's Key Club. A mobile application has been developed that provides Key Club members easy access to the details and sign-up areas of all the upcoming events. Key Club officers can also conveniently add and edit the current volunteer events that appear. The application will also, in future updates, be able to further encourage volunteer sign-rups by recommending relevant events to users based on their profile. In our experimental research, we studied the accuracy of specific machine learning algorithms in developing this recommendation engine, focusing on which combination of factors would produce the highest learning accuracy in recommending an event. Through the experiment, we discovered that the Gaussian Naive-Bays machine learning model produced the highest accuracy in predicting which volunteering event types a user would prefer.

The application has a couple of current limitations that we seek to resolve in future work. One feature we plan to add is a wait-list system for various opportunities when the volunteer limit has been reached. It is common for a user to want to sign-up for an event when it is full, and they never find out if someone already signed up decides to cancel, leaving a spot open. A wait-list would be a great way to notify members of this.

Another limitation of the current app is it is just a view and sign-up system with no form of reminding volunteers of an upcoming event they are signed up for, or if there are any changes to the event. We plan to implement a push notifications system within the application to overcome this limitation.

We also plan to work more in the future on improving the user-friendliness of the application, such as enhancing the UI design and implementing the recommendation engine that was a part of our research [13]. This would enhance the user experience on the app and motivate them to continue using it.

Finally, we plan to expand the application, so it can be used for all types of volunteering organizations or clubs [14]. This will be done by making the app into a template that can be customized and then utilized by different groups to help them coordinate events and sign-ups.

REFERENCES

[1] Morens, David M., Gregory K. Folkers, and Anthony S. Fauci. "What is a pandemic?." (2009): 1018-1021.

[2] Thiele, Marco R. "Streamline simulation." 6th International Forum on Reservoir Simulation. Society of Petroleum Engineers: Schloss Fuschl, Austria, 2001.

[3] Gimenez-Nadal, Jose Ignacio, and Almudena Sevilla-Sanz. "The time-crunch paradox." Social indicators research 102.2 (2011): 181-196.

[4] Murphy, Paul Regis, and Donald F. Wood. Contemporary logistics. Vol. 9. Upper Saddle River: Prentice Hall, 2004.

[5] Montello, Daniel R. Navigation. Cambridge University Press, 2005.

[6] Zatta, Claudia. "Interconnectedness." (2019).

[7] Castano, Silvana, et al. Database Securi. Addison—Wesley, 1995.

[8] Voevodsky, Vladimir. "Cancellation theorem." arXiv preprint math/0202012 (2002).

[9] L. Butgereit, "Seven characteristics of a successful virtual volunteering platform," 2011 IST-Africa Conference Proceedings, 2011, pp. 1-8.

[10] Reuter, Christian, et al. "XHELP: Design of a cross-platform social-media application to support volunteer moderators in disasters." Proceedings of the 33rd annual ACM conference on human factors in computing systems. 2015.

[11] Gudmunds, Emma, and Lovisa Tegelberg. "The Volunteer App-Increasing Volunteer Engagement Through User Experience and Behavioral Design." (2021).

[12] Diebold, Francis X., and Robert S. Mariano. "Comparing predictive accuracy." Journal of Business & economic statistics 20.1 (2002): 134-144.

[13] Backer, Thomas E. "Assessing and Enhancing." Reviewing the behavioral science knowledge base on technology transfer 155 (1995): 21.

[14] Biere, Armin. "Resolve and expand." International conference on theory and applications of satisfiability testing. Springer, Berlin, Heidelberg, 2004.

[15] Build character with key club through service and leadership. Key Club. (n.d.). Retrieved September 28, 2021, from https://www.keyclub.org/.

[16] Remind. (n.d.). Retrieved September 28, 2021, from https://www.remind.com/.

[17] Bracha, G. (2015). *The Dart programming language*. Addison-Wesley Professional.

[18] Vlachoudis, V. (2009, May). FLAIR: a powerful but user friendly graphical interface for FLUKA. In *Proc. Int. Conf. on Mathematics, Computational Methods & Reactor Physics (M&C 2009), Saratoga Springs, New York* (Vol. 176).

[19] Ontivero-Ortega, M., Lage-Castellanos, A., Valente, G., Goebel, R., & Valdes-Sosa, M. (2017). Fast Gaussian Naïve Bayes for searchlight classification analysis. *Neuroimage*, *163*, 471-479.

[20] Rodriguez-Galiano, V. F., Ghimire, B., Rogan, J., Chica-Olmo, M., & Rigol-Sanchez, J. P. (2012). An assessment of the effectiveness of a random forest classifier for land-cover classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, *67*, 93-104.