A SMART COMMUNITY-DRIVEN TUTORING MOBILE PLATFORM USING ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Haoyun Yang[1], Yu Cao[2]

[1]Lutheran High School of Orange County, 2222 North Santiago Boulevard, Orange, CA 92867
[2]Computer Science Department, California State Polytechnic University, Pomona, CA 91768

ABSTRACT

Our application aims to assist students with class materials by providing a platform with multiple-choice and free-response questions for them to practice [13]. We built this application with Flutter and Firebase which provides compatibility through different mobile platforms and secure data storage [14]. To provide an effective and user-friendly interface, we allow everyone to create their own quizzes based on their focus area and explore quizzes others created. We also developed a hint system where the user can access the hint if they struggle to find the correct solution. Moreover, the user can access their recent quiz attempts to further review the concepts and materials they missed. We validated the effectiveness of the application by conducting an experiment on the user's quiz score over time given continuous practice. We also conducted another experiment on the impact of hints on users' understanding of materials which concluded a mixed results in their accuracy. Throughout our tests, we proved the effectiveness of the application and its success in fulfilling its purpose.

KEYWORDS

Education, Academic, Mobile Application, Practice

1. INTRODUCTION

The problem that the app intends to solve is the struggles school students may have during their academic careers. Studies show that high school students benefit from at least two hours of school-related studies per night [1]. Throughout the stages of primary and secondary school, children may encounter difficulties in their coursework. If left unattended, students may fall behind without parental acknowledgment, causing a potential dent in their future opportunities. These difficulties are generally resolved through external practice, to which this app provides.

The first application that we found attempts to create a speciation between the instructor and the student. Only instructor accounts were able to develop quizzes, which although can increase the legitimacy of the quizzes, lack a peer-to-peer connection and are not much different than traditional study [15]. Our project allows all users to write quizzes, where students can pinpoint the difficult ideas that were taught and create meaningful hints and questions that can support other students.

The second application tries to accomplish something similar to us but lacks features that can heavily benefit the users in their studies. The quiz portion of the app only contained true or false options, limiting creators to certain questions and consumers to have a 50% chance of obtaining the point through guessing. Our application introduces customizable choices in our multiple-choice questions, where instructors and students can create any amount of questions they wish. Users are also able to select multiple correct answers, decreasing the motivation to guess when stuck.

The final application that we reviewed did not contain quizzes for studying, instead uses flashcards to solve a similar problem. The app allows the user to create and review their flashcards to prepare for exams but lacks evaluation of their progress. Quizzes are more helpful in this aspect as they tell the user how well they understand the material, ensuring confidence on the day before the examination.

The application provides a platform for students to practice the concepts they learned at school. Through prolonged practice on the app, users may see an increase in their academic performance [2]. The solution is effective because it allows teachers to customize the quizzes based on their lectures, creating an environment which perfectly aligns the course material with the students' review. The application also allows students to create their own quizzes which is more self guided and can allow students to focus on the concepts that they need help with [4]. General practice at home proves less effective than this application, as the content being reviewed isn't governed by the instructor. Students may find themselves researching topics and attaining answers differing from what was taught.

In our experiments, our goal was to evaluate the app's effectiveness in helping the students grasp the learning materials. Through giving a group of students quizzes on the same concept and a time barrier of one week in between. The experiment yielded a direct positive relationship between the number of tries the students had and the accuracy of their quizzes. Therefore, we concluded that the application was effective in raising the users' scores with each quiz attempt. It also suggests that continuous reinforcement in memory helps students with previously learned materials.

We also studied the impact of hints on user performance on quizzes. A quiz was again given to a group of students, but they were split into two groups with one group having access to hints, and the other group not. Throughout their two attempts, the students who were not given hints showed a more consistent score on their next quiz than those who did. There was not a significant difference in mean between the two scores, concluding that quizzes with hints may only be beneficial to users who utilize them in a proper way.

2. CHALLENGES

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

2.1. The lack of past analysis

A problem that arose in my project was the lack of past analysis. Users could take any quiz they wanted but observe their results only once. Being able to save results is important to students as they can look back upon their progress and improvement, which will also give them a sense of how the app is developing their learning abilities. To resolve this problem, I would use Firebase Firestore to save any quiz that the consumers take. Once on the cloud, it can be retrieved any time and will include every attempt the user has made.

2.2. Check for the questions

When the users are creating quizzes, it is crucial to check for the questions the user created as for example every question must have at least one correct answer. We also need to check on the number of questions that exist in a quiz as ideally a quiz would consist of multiple questions. To address this, we can check those conditions when the user clicks the button to submit the quiz they created. If one of those conditions doesn't match, we can display a snackbar or a popup to inform the user about the rules of creating quizzes.

2.3. The lack of question support

Another challenge that was present in the application was the lack of question support. When students struggle with a question, they are left with no choice but to guess, which doesn't benefit their learning and can cause false readings in their scores. Some students may also resort to using outside resources for their quizzes, which may only prove to give them the direct answer. To attack this problem, I would implement a hint feature in which the user selects to be given slight nudges to the correct answer. If the quiz creator did not provide a hint, eliminating a few wrong answers can also help the student arrive at the correct selection without completely nullifying their learning.

3. Solution

User Input: The application receives input in a variety of ways. This can be displayed when the user creates a quiz, answers questions, or navigates around the app. User input can include typing, gestures, and tapping.

Processing: Most of the data on the application is stored on Google Firebase. Using their Firebase Firestore, the application can save quizzes, user history, and more. The data is retrieved whenever students open the app, bringing an updated homepage consistently.

Firestore is also used in assessing the students' answers on submission. When the user submits their quiz, an algorithm is utilized in which the choices are compared with the correct selections. Correct selections are always up to date, as the app requests from Firestore upon completion. Once the score has been determined in our code, it is returned to Firestore for retainment and also shown to the user through their current page.

Output: Using Flutter's vast components, the app displays a complex user interface. Widgets like AnimatedContainer are present throughout the app to create smooth transitions and overall improve the user's experience.

When the code needs to output data to the user, StatefulWidgets and Providers are useful for updating the page in small quantities. This way, the user interface remains precise and uses minimal resources to display.

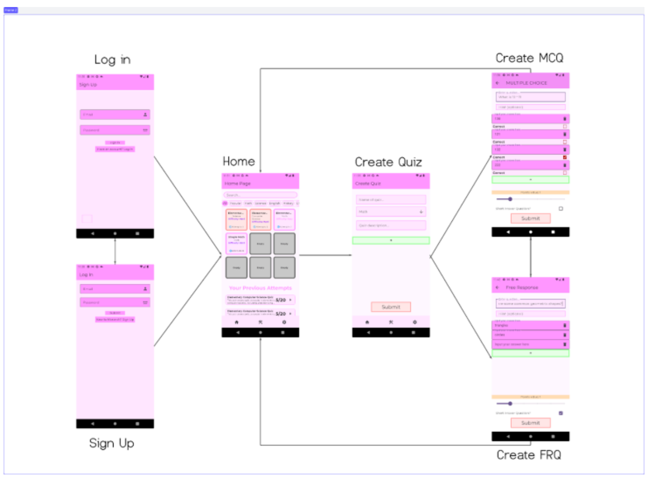


Figure 1. Overview of the solution

User Input

User input is important to nearly all the applications on the internet [12]. They help the app function in ways specific to the consumers. The purpose of user input in this particular app lies in the creation and taking of quizzes.

Authentication is implemented into the user input to ensure security and personalization in the application. Each user has their own account, allowing compatibility over multiple devices. The authentication component also creates an environment in which the user can be assured that their quiz attempt history is saved and hidden from potential attackers.

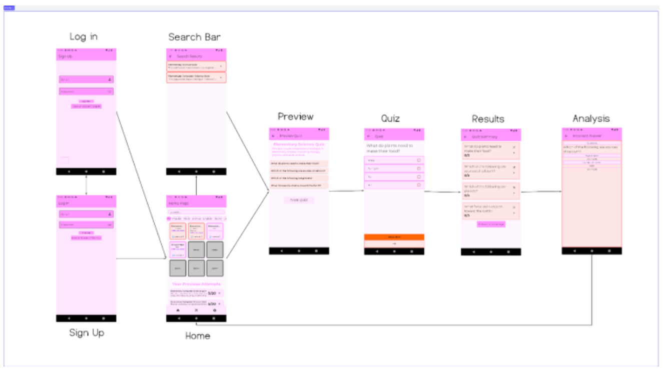
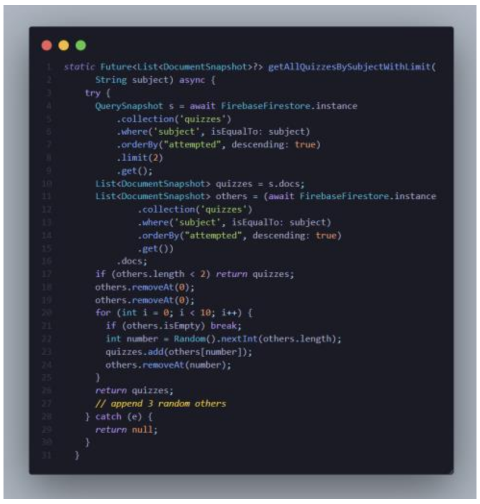


Figure 2. Screenshot of the process

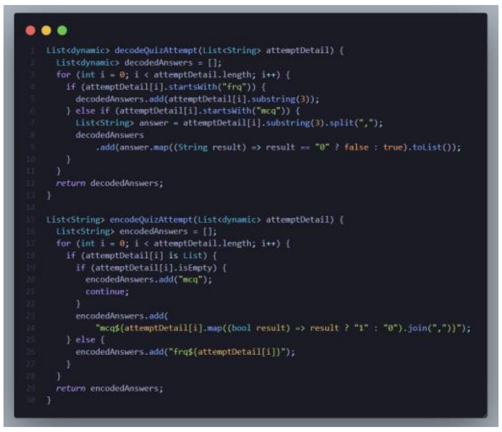


```dart
static Future<List<DocumentSnapshot>?> getAllQuizzesBySubjectWithLimit(
      String subject) async {
    try {
      QuerySnapshot s = await FirebaseFirestore.instance
          .collection('quizzes')
          .where('subject', isEqualTo: subject)
          .orderBy("attempted", descending: true)
          .limit(2)
          .get();
      List<DocumentSnapshot> quizzes = s.docs;
      List<DocumentSnapshot> others = (await FirebaseFirestore.instance
              .collection('quizzes')
              .where('subject', isEqualTo: subject)
              .orderBy("attempted", descending: true)
              .get())
          .docs;
      if (others.length < 2) return quizzes;
      others.removeAt(0);
      others.removeAt(0);
      for (int i = 0; i < 10; i++) {
        if (others.isEmpty) break;
        int number = Random().nextInt(others.length);
        quizzes.add(others[number]);
        others.removeAt(number);
      }
      return quizzes;
      // append 3 random others
    } catch (e) {
      return null;
    }
  }
```

Figure 3. Screenshot of code 1

The code above displays the algorithm which sorts out which quizzes to display to the user. The first consideration that the code goes through is the selected subject. The function getAllQuizzesBySubjectWithLimit() accounts for this and returns a list of applicable quizzes. These quizzes are in order of attempts taken as the two most popular ones have been sorted out.

Two quizzes are removed from the start of the list as they are guaranteed to be displayed, being the ones with the highest popularity. The program then utilizes the Random().nextInt() to sort out random quizzes from the remaining pile and show them to the user respectively.



```dart
List<dynamic> decodeQuizAttempt(List<String> attemptDetail) {
  List<dynamic> decodedAnswers = [];
  for (int i = 0; i < attemptDetail.length; i++) {
    if (attemptDetail[i].startsWith("frq")) {
      decodedAnswers.add(attemptDetail[i].substring(3));
    } else if (attemptDetail[i].startsWith("mcq")) {
      List<String> answer = attemptDetail[i].substring(3).split(",");
      decodedAnswers
          .add(answer.map((String result) => result == "0" ? false : true).toList());
    }
  }
  return decodedAnswers;
}

List<String> encodeQuizAttempt(List<dynamic> attemptDetail) {
  List<String> encodedAnswers = [];
  for (int i = 0; i < attemptDetail.length; i++) {
    if (attemptDetail[i] is List) {
      if (attemptDetail[i].isEmpty) {
        encodedAnswers.add("mcq");
        continue;
      }
      encodedAnswers.add(
          "mcq${attemptDetail[i].map((bool result) => result ? "1" : "0").join(",")}");
    } else {
      encodedAnswers.add("frq${attemptDetail[i]}");
    }
  }
  return encodedAnswers;
}
```

Figure 4. Screenshot of code 2

The code above displays the algorithm for tracking the users' previous quiz attempts. Since there are both short answer questions and multiple choice questions with potentially multiple correct answers, we decided to use an encoded string to represent the result of the quiz attempt. The beginning three letters of the encoded string denote the type of the question: "mcq" for multiple choice questions and "frq" for free response questions.

When the user's quiz attempt comes to be stored, the encodeQuizAttempt is used. In the case of a multiple-choice question, each possible answer is mapped with 1 or 0, 1 being the choices the user selected and 0 being empty. For free response, only one answer may be inputted. Therefore, we simply append the user's text input of the free response question to the array.

The code for quiz decryption has a similar structure. Using the identifiable 3 letters at the beginning of each string, questions are categorized by their type and decoded separately. The split() function is utilized for multiple choice questions, converting the string into a list which will include all possible selections. Again, for free response, the sole input is appended.

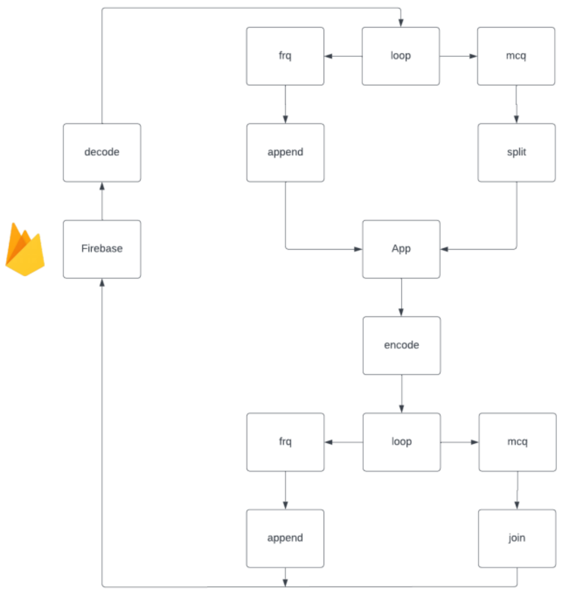
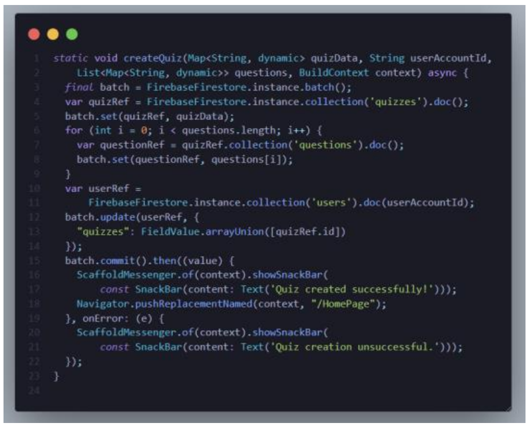


Figure 5. Screenshot of the input



```
static void createQuiz(Map<String, dynamic> quizData, String userAccountId,
    List<Map<String, dynamic>> questions, BuildContext context) async {
  final batch = FirebaseFirestore.instance.batch();
  var quizRef = FirebaseFirestore.instance.collection('quizzes').doc();
  batch.set(quizRef, quizData);
  for (int i = 0; i < questions.length; i++) {
    var questionRef = quizRef.collection('questions').doc();
    batch.set(questionRef, questions[i]);
  }
  var userRef =
      FirebaseFirestore.instance.collection('users').doc(userAccountId);
  batch.update(userRef, {
    "quizzes": FieldValue.arrayUnion([quizRef.id])
  });
  batch.commit().then((value) {
    ScaffoldMessenger.of(context).showSnackBar(
        const SnackBar(content: Text('Quiz created successfully!')));
    Navigator.pushReplacementNamed(context, "/HomePage");
  }, onError: (e) {
    ScaffoldMessenger.of(context).showSnackBar(
        const SnackBar(content: Text('Quiz creation unsuccessful.')));
  });
}
```

Figure 6. Screenshot of code 3

This code displays the "createQuiz" function which will upload the quiz to Firebase Firestore. In order to store the data in Firestore, the questions (currently a List) must be split and stored in the documents separately. The risk of making separate requests to put the data is that if one of the requests fails the others will still be executed which could cause discrepancy between data. As a result, we used Firebase's batch write to avoid this problem.

All the operations in the batch write are atomic, meaning that the operations are either fully processed or not processed at all. Therefore, we don't have to worry about any discrepancy that may occur within the data or rolling back the data ourselves.

After the batch has been committed, the user is notified of the status of the creation with the SnackBar widget. Lastly, if successful, the navigator moves the current display back to the home page.

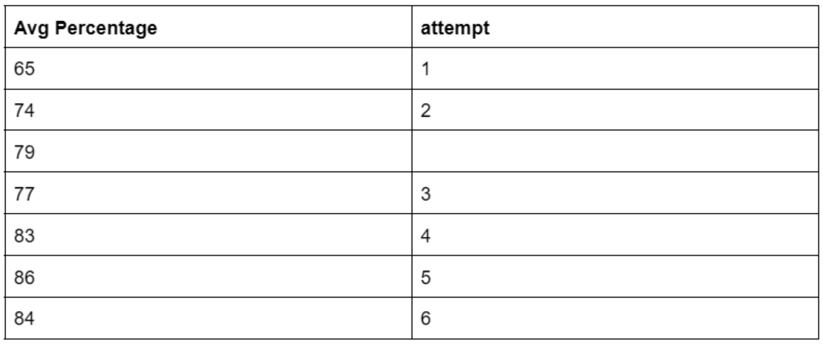
4. EXPERIMENT

4.1. Experiment 1

This experiment will determine whether the users retain the information in attempting the quizzes on the application. This is a critical criterion because this shows the effectiveness of the app in assisting the user's academics. When students take a lecture immediately, they can generally recall most of the knowledge due to short-term memory [3]. However, this short-term memory fades away over time. Therefore, students need practices like homework to better retain their knowledge. To examine our app's effectiveness in helping the user memorize and understand the materials, we designed an experiment to address this aspect.

By sending the application to a variety of users, we can gather retainment data through their attempts on the same quiz. We request them to take a quiz with a time difference of 1 week between tries. Their results from the quizzes can then be plotted on a graph. If a positive slope is seen between their attempts chronologically and their score, it verifies that the application is helpful in its purpose of supporting students in their studies.

Figure 7. Table of experiment 1



Avg Percentage	attempt
65	1
74	2
79	
77	3
83	4
86	5
84	6

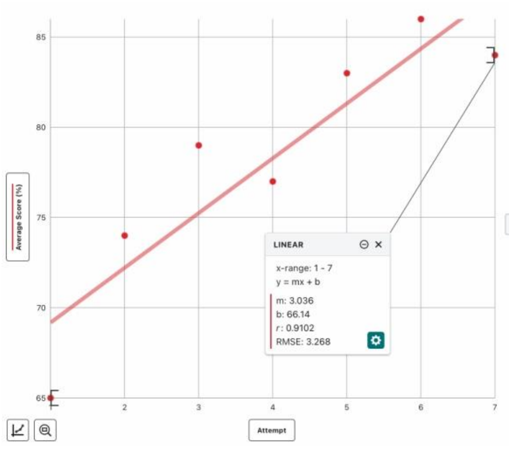


Figure 8. Figure of experiment 1

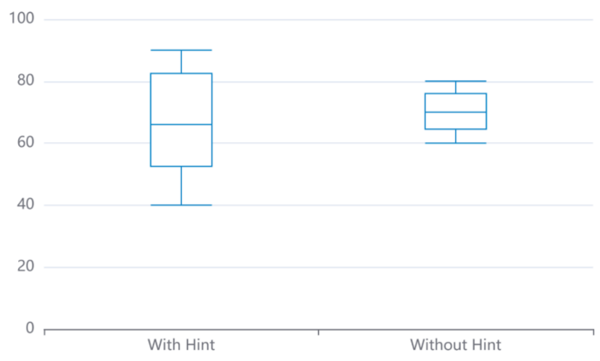
The slope in the shown graph was 3.036, expressing a positive relationship between the attempt number and average score, measured chronologically. The correlation coefficient was 0.9102, indicating that the positive relationship is highly valid and represents the positive effect of our practices on the user’s ability to retain knowledge. Despite a few outliers, the general scoring of the average users increased each time they took the same quiz. The data from this experiment supports our application in its goal to increase the student’s knowledge in their coursework. A different study also shares a similar result, supporting the fact that performance is boosted through quiz learning [5]. From this, instructors will also have a better time teaching the concepts in class [6].

4.2. Experiment 2

For Experiment B, we decided to investigate the effectiveness of hints on the students’ accuracy in their future attempts. We initially believed that hints may hinder the users’ ability to understand and process the question/concepts by pushing them too close to the answer. As the app is intended to help students learn the concepts, this experiment is vital in understanding the effects of hints so that we may improve the feature to help consumers.

Design: For this experiment, we created two groups of 10 users with one group having access to a hint for each question and another group without hints. After 1 week elapsed, we gave each group a quiz with similar concepts, both without hints, to test their impacted differences.

Shown: Average score of the two groups on their second attempt, in percentage.



With Hint	Without Hint
max: 90	max: 80
Q3: 83	Q3: 76
median: 66	median: 70
Q1: 53	Q1: 65
min: 40	min: 60

Figure 9. Figure of experiment 2

Analysis:

Through this experiment, we claim that the group of students who were given hints on their first attempt had a larger range and standard deviation in their score distribution. This could show that some struggled on their blind second attempt, potentially due to the lack of processing and understanding on their first try. While some utilize the hint and understand the materials better.

The second group that were not given hints had stabler results on their second quiz. The standard deviation was lower, showing that students had more predictable quiz results in studying without a hint. The median of the second group was also shown to be higher than the students who received the hint, signifying a better understanding of the given materials.

Overall, the impact of hints on practice quizzes can depend on the quality of the hints and the activeness of how students interact with them [7].

5. RELATED WORK

Aaboud's quiz application contains a student and an admin interface, which can be helpful when the type of questions requires a level of expertise to create [8]. The separate interface also helps the users navigate the app easier. However, distinguishing administrators from students may become a limitation, as peer-to-peer studying has proven to be more beneficial than resources provided by teachers. Our app allows all users to create quizzes, in which when students have resolved their struggles, they are able to develop unique questions that tackle the issue, supporting other users in their studies.

Reddy and Anjum's quiz app contain a sharing feature that allows users to jump in on questions that their friends may send them [9]. Although potentially effective, the application is limited in its True/False category. The developers focused on the payment aspect instead of maximizing the consumers' learning experience. Through the variety of questions on our app Monarch, students have a wider range of selections and learning opportunities. For example, our app provides free-response applicability to accurize the user's scoring and limit guessing on questions.

Huynh's application tackles the problem of student learning through flashcards [10]. Although they are beneficial in preparing for exams and building memory, this app is unable to evaluate the student's progress and the effectiveness of their learning. Due to this, students will not be able to be confident in themselves with clear numbers that they have successfully understood the concepts. Furthermore, flashcards usually contain specific, divisionalized ideas that may not be present on the exam. Quizzes on the other hand follow test format and can simulate the user's performance in class.

6. CONCLUSIONS

One limitation of this project is that students are unable to directly communicate with their classmates. The class as a whole may share similar struggles due to one consistent instructor, and these problems should be resolved as a team. To improve on this aspect, a chat and share function may be implemented into the application in which students taking similar classes can speak to each other. A tutor function can also be added, where users can voluntarily set up a tutor account and provide support to students who haven't done well on their recent quiz attempts.

Another limitation of the application is the lack of data feedback the quiz creator can view. Usually, instructors, these creators can benefit from data including the average score of their quizzes, which may indicate a lack of clarity in certain questions, difficulty measurement, or the effectiveness of the quiz [11]. I would create a dashboard for the quiz writers in which they are notified of all completed attempts on their quiz. They can then use the data to their advantage, creating questions that can accelerate the consumers' learning even better.

To conclude, our app provides users a platform to practice and reinforce the materials learned in the day through multiple-choice and free response questions. However, there are still many areas we can further develop to improve the user's learning experience, such as shareable quizzes, tutor sections, and data for quiz creators. In the future, we will continue developing a more interactive and user-friendly application.

REFERENCES

[1] Brabeck, Mary, J. Jeffrey, and Sara Fry. "Practice for knowledge acquisition (not drill and kill)." American Psychological Association (2011).
[2] Kirtman, Lisa. "Online versus in-class courses: An examination of differences in learning outcomes." Issues in teacher education 18.2 (2009): 103-116.
[3] Cowan, Nelson. "What are the differences between long-term, short-term, and working memory?." Progress in brain research 169 (2008): 323-338.
[4] Gamage, Sithara HPW, et al. "Optimising Moodle quizzes for online assessments." International journal of STEM education 6 (2019): 1-14.
[5] Hillman, Jennifer. "The impact of online quizzes on student engagement and learning." Introduction to Abnormal Psychology (2012): 1-6.
[6] Price, Thomas W., Rui Zhi, and Tiffany Barnes. "Hint generation under uncertainty: The effect of hint quality on help-seeking behavior." Artificial Intelligence in Education: 18th International Conference, AIED 2017, Wuhan, China, June 28 – July 1, 2017, Proceedings 18. Springer International Publishing, 2017.
[7] Aaboud, Maroua Hibatollah. "Developing a quiz application for medical students: case: pill game." (2023).
[8] Huynh, Tung. "A flashcard mobile application development with Flutter." (2021).

[9] Cheon, Yoonsik, and Carlos Chavez. "Converting Android native apps to Flutter cross-platform apps." 2021 International conference on computational science and computational intelligence (CSCI). IEEE, 2021.
[10] Hatziapostolou, Thanos, and Iraklis Paraskakis. "Enhancing the impact of formative feedback on student learning through an online feedback system." Electronic Journal of E-learning 8.2 (2010): 111-122.
[11] Shafiq, Farah, and Aishah Siddiquah. "EFFECT OF CLASSROOM QUIZZES ON GRADUATE STUDENTS'ACHIEVEMENT." International Journal of Academic Research 3.5 (2011).
[12] Dong, Jianming, Shirley Martin, and Paul Waldo. "A user input and analysis tool for information architecture." CHI'01 extended abstracts on Human factors in computing systems. 2001.
[13] Juliana, H. D. R., et al. "Evecurate – A Smart Event Management App Using Flutter and Firebase." International Journal of Scientific Research & Engineering Trends 7.4 (2021): 2519-2524.
[14] Zhou, Zhibin, and Dijiang Huang. "Efficient and secure data storage operations for mobile cloud computing." 2012 8th international conference on network and service management (cnsm) and 2012 workshop on systems virtualiztion management (svm). IEEE, 2012.
[15] Xiao, Li, Yunhao Liu, and Lionel M. Ni. "Improving unstructured peer-to-peer systems by adaptive connection establishment." IEEE Transactions on Computers 54.9 (2005): 1091-1103.