AN INTELLIGENT MOBILE APP TO ASSIST IN FINDING PERSONALIZED STUDENT OPPORTUNITIES USING ARTIFICIAL INTELLIGENCE AND FIREBASE

Zhengtan Jin [1], Han Tun Oo [2]

[1] Beckman High School, 3588 Bryan Ave, Irvine, CA 92602
[2] California State Polytechnic University, Pomona, CA, 91768

ABSTRACT

This paper presents an AI-driven mobile application that helps students discover personalized opportunities, such as internships, volunteering, activism events, and mental health resources [1]. Students often face challenges navigating a vast number of options, further complicated by discrimination and eligibility barriers. The app uses Firebase for data storage and authentication, and ChatGPT to evaluate and rank opportunities based on user-provided profile data using a structured scoring prompt [7]. Two experiments were conducted to test its effectiveness. The first assessed the consistency of AI rankings using prompts of varying specificity, showing that a clear and hyper-structured prompt produced the most reliable results, with up to 74.4% consistency. The second experiment tested different search keywords for locating mental health providers, revealing that specific terms like "mental health" returned more accurate results than general terms like "therapy." These findings support the app's ability to deliver consistent, personalized, and relevant recommendations, ultimately simplifying the process for students to access valuable opportunities.

KEYWORDS

Student opportunities, AI recommendation, Personalized ranking, Mental health access

1. INTRODUCTION

Many students struggle with finding help and figuring out what they want to do. Internships, volunteering, activism, and mental health are all very important aspects to one's academic career. Participating in these opportunities not only helps them gain valuable experiences but also helps build their identity and resume. There is such a wide range, and it is often difficult for students to find the right ones for them. Their own gender, race, and beliefs can make it even hard to find certain opportunities that they want. According to Pew Research Center studies, nearly half of Black men and over a third of Black women report experiencing discrimination or unfair treatment by employers due to their race [2]. Additionally, about 42% of working women say they have faced discrimination on the job because of their gender, including unequal pay and being passed over for important assignments. Discrimination makes finding these opportunities even harder, and students often go through a lot of trouble in order to find the right ones for them [3].

This project is built upon existing research aimed at improving AI consistency and reducing bias. Cao et al. (2024) attempted to reduce prompt bias by subtracting inherent biases detected from blank prompts, but their method struggled with open-ended tasks [8]. Our approach improves this by using a structured scoring prompt to minimize open-ended interpretation. Wei et al. (2022) used Chain-of-Thought prompting to guide AI reasoning step by step, which helped in factual tasks but not in subjective ranking. Instead, our project uses a weighted scoring system to make the AI score specific criteria, reducing ambiguity. Tang et al. (2023) addressed inconsistent AI rankings by averaging results from multiple shuffled runs, but their method was time-consuming. Our app achieves similar consistency by using a single, fixed scoring prompt that assigns numeric scores to each item, making the process faster and more efficient. These improvements enable our app to deliver more consistent and personalized results without requiring repeated model calls. An AI-powered app that provides opportunities perfectly tailored based on the student's information. By using AI, the students would be saving a lot of time and trouble as all they have to do is plug in their information. It will find opportunities that match the students'needs, and they no longer have to worry about discrimination or inability. Using AI to help filter through the opportunities not only saves time but could also potentially discover opportunities the student has never known before. LinkedIn's 2023-2024 data reports showed that their AI job recommender has helped increase job application rates by nearly 30-40% compared to non-personalized searches. This shows how accurate and helpful AI can be when it comes to finding personalized opportunities that match the person's needs.

In experiment 1, we tested the AI's inconsistency with ranking opportunities by changing the scoring prompt complexity from simple to hyper-specific. Keeping the same user data, each prompt was run multiple times to compare the overlap in the top results. The key finding was that the most structured prompt with the clearest directions produced the highest consistency. More details did not always mean more consistent, and clarity in the prompt matter actually more. Experiment 2 tested with the accuracy of mental health provider results from Google Maps by using different keywords. The key finding was that the most specific and less broad keywords such as "mental health" and "psychologist" had the most consistency. How specific the keyword greatly affected the accuracy of the search engine. Both experiments showed that clarity and specificity had the biggest effect on reliable and consistent outputs.

2. Challenges

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

2.1. Structured Prompts for Consistent AI Ranking

The AI algorithm prompt is a major component of the program. The prompt has to be just right or else the AI would often give inaccurate rankings based on its own bias. ChatGPT has its own randomness and bias that make creating consistent and accurate rankings of opportunities difficult. To solve these issues, I could create a very specific algorithmic prompt rather than just feeding it the opportunities and telling it to rank it based on the user's information. By creating things such as a point system that assigns points to how well each aspect of the opportunity matches with the user's information, it takes out much of the AI's bias and creates much more consistent lists.

2.2. Improving Matches with Detailed Input Data

Information is also another major component of the program. There needs to be enough information from both the opportunities and the user or else the AI could struggle with finding accurate opportunities and could show unrelated results. To fix this problem, I could create a very detailed page where the user can put in a lot of their personal information. The more information the AI has, the more accurate it can be in finding perfect matching opportunities for that student. The opportunities could also be more detailed and we could create multiple fields for each opportunity so the AI will have a deeper understanding of that opportunity and thus provide more accurate results.

2.3. Optimizing Performance with Async Scoring and API Management

Another core component of the program is the database. The program needed to handle real-time data while also integrating the AI's algorithmic scoring system. Each opportunity required an API call to generate a personalized score based on the user's profile. This can create big performance issues and long wait times. To solve this, I could use the async, future, and await features to handle asynchronous operations in an efficient manner. The AI's scoring system could also be handled sequentially so we can avoid API rate limits, which would create a much more smooth experience.

3. SOLUTION

The system is structured around three core components: the user profile, the database, and the AI-based opportunity ranking system. The application begins by allowing users to log in or sign up through Firebase Authentication [9]. Once authenticated, users are directed to a profile page where they provide detailed personal information such as gender, ethnicity, grade level, location, academic interests, and hobbies. This information is stored securely in Firebase Firestore.

The second component, the database, houses all opportunity listings, including internships, volunteering events, activism programs, and mental health resources. Each entry is organized with relevant attributes such as eligibility criteria, location, subject focus, and type.

The final component is the AI ranking engine, which utilizes ChatGPT [10]. When a user visits a specific opportunity category, the system retrieves all relevant opportunities and combines them with the user's profile. A custom scoring prompt is sent to ChatGPT, which evaluates how well each opportunity aligns with the user's data. It assigns a compatibility score from 0 to 1, allowing the app to rank and display the results accordingly.

This entire workflow is built using Flutter for the frontend, Firebase for data handling and user management, and OpenAI's ChatGPT for AI processing [11]. The seamless integration of these technologies enables a smooth and personalized experience for users, helping them discover the most relevant and meaningful opportunities efficiently.

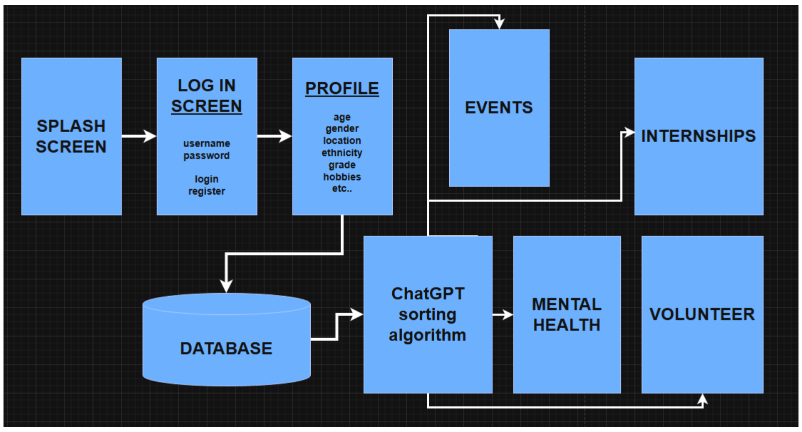
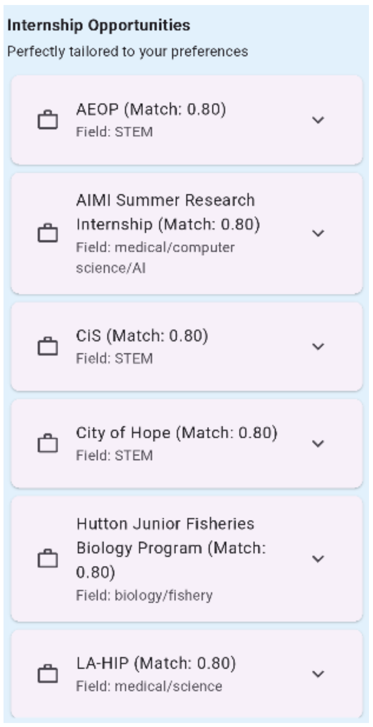


Figure 1. Overview of the solution

The AI-based recommendation system is responsible for evaluating and ranking all available opportunities based on each user's profile. This component uses ChatGPT to analyze how well each opportunity aligns with the user's background, interests, and eligibility. A customized scoring prompt is used to reduce AI bias and improve consistency, assigning a score between 0 and 1 for each opportunity [12]. These scores determine the ranking order shown in the app. To ensure accuracy, the AI prompt is designed with a fixed structure that evaluates key criteria such as subject fit, location, and user qualifications using a weighted system.


Internship Opportunities

Perfectly tailored to your preferences

AEOP (Match: 0.80)
Field: STEM

AIMI Summer Research Internship (Match: 0.80)
Field: medical/computer science/AI

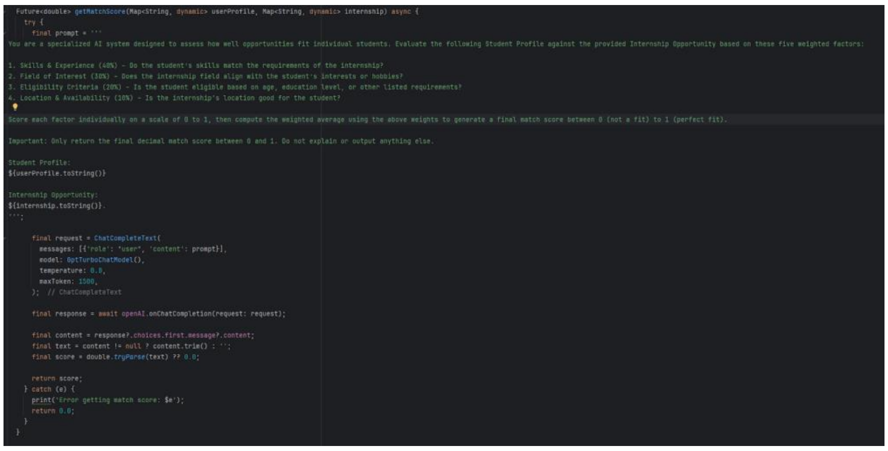
CIS (Match: 0.80)
Field: STEM

City of Hope (Match: 0.80)
Field: STEM

Hutton Junior Fisheries Biology Program (Match: 0.80)
Field: biology/fishery

LA-HIP (Match: 0.80)
Field: medical/science

Figure 2. Screenshot of internship opportunities



```
  Future<double> getMatchScore(Map<String, dynamic> userProfile, Map<String, dynamic> internship) async {
    try {
      final prompt = '''
You are a specialized AI system designed to assess how well opportunities fit individual students. Evaluate the following Student Profile against the provided Internship Opportunity based on these five weighted factors:

1. Skills & Experience (40%) - Do the student's skills match the requirements of the internship?
2. Field of Interest (30%) - Does the internship field align with the student's interests or hobbies?
3. Eligibility Criteria (20%) - Is the student eligible based on age, education level, or other listed requirements?
4. Location & Availability (10%) - Is the internship's location good for the student?

Score each factor individually on a scale of 0 to 1, then compute the weighted average using the above weights to generate a final match score between 0 (not a fit) to 1 (perfect fit).

Important: Only return the final decimal match score between 0 and 1. Do not explain or output anything else.

Student Profile:
${userProfile.toString()}

Internship Opportunity:
${internship.toString()}.
''';

      final request = ChatCompleteText(
        messages: [{'role': 'user', 'content': prompt}],
        model: GptTurboChatModel(),
        temperature: 0.0,
        maxToken: 1500,
      );  // ChatCompleteText

      final response = await openAI.onChatCompletion(request: request);

      final content = response?.choices.first.message?.content;
      final text = content != null ? content.trim() : '';
      final score = double.tryParse(text) ?? 0.0;

      return score;
    } catch (e) {
      print('Error getting match score: $e');
      return 0.0;
    }
  }
```

Figure 3. Screenshot of code 1

The getMatchScore function is triggered whenever the user navigates to a category page such as internships, volunteering, activism, or mental health resources. This function sends a structured prompt to ChatGPT that includes the user's profile data along with all available opportunities. The prompt instructs the AI to evaluate each opportunity based on four main criteria: user eligibility, interests, skills, and location.
Within the function, a variable called prompt is created, containing a scoring guide that clearly outlines how the AI should assign scores. Each opportunity is processed individually, and the AI returns a compatibility score ranging from 0 to 1. These scores are stored in a variable called score.

The scores are then used to sort the list of opportunities in descending order, ensuring that the most relevant results are shown first. This process minimizes randomness and enhances the accuracy of recommendations by providing consistent, structured input to the AI.

The user profile component is responsible for collecting and storing personal information that is essential for generating accurate recommendations. This includes data such as age, gender, ethnicity, location, academic interests, and hobbies. The component uses multiple text controllers within the Flutter framework to capture user input. Once the user submits their information, the data is uploaded to Firebase Firestore, where it becomes accessible for the AI sorting algorithm. This structured and detailed data allows the AI to make more informed and relevant matches when ranking opportunities.

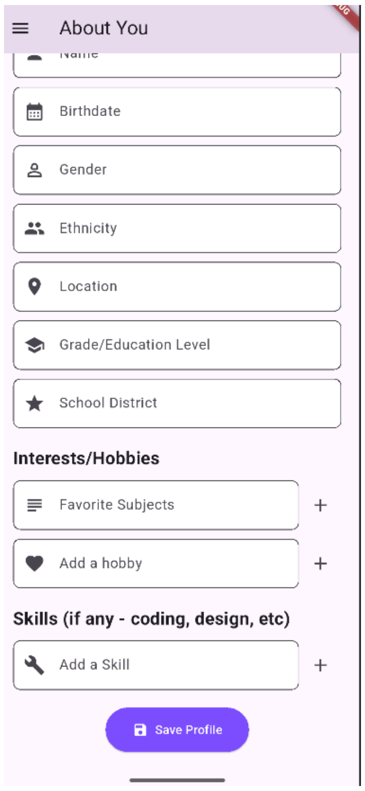
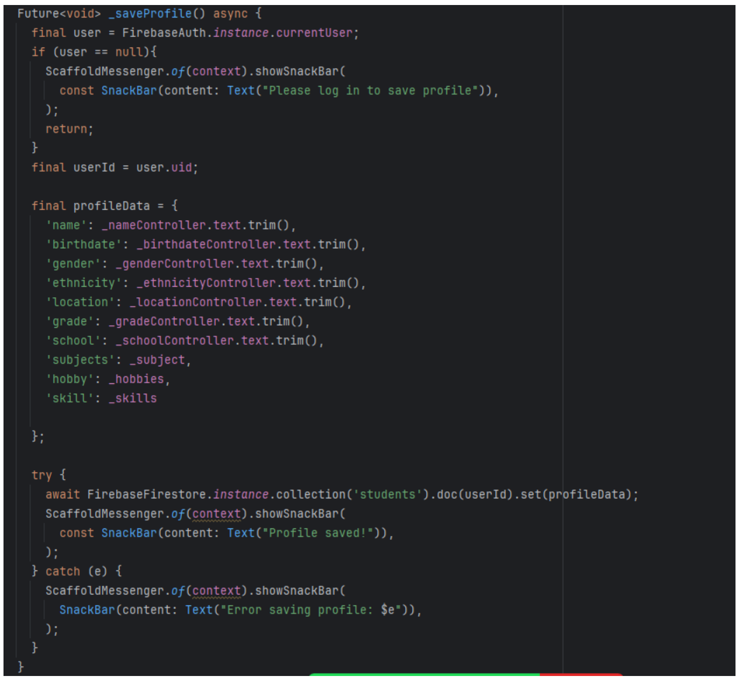


Figure 4. Screenshot of profile page



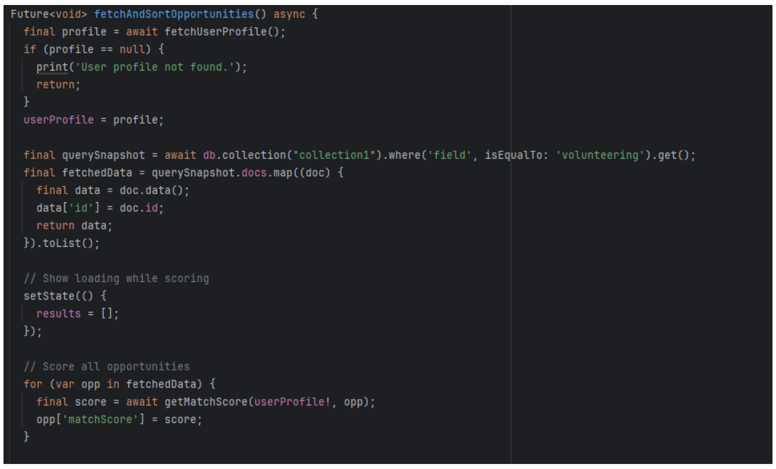
```
Future<void> _saveProfile() async {
  final user = FirebaseAuth.instance.currentUser;
  if (user == null){
    ScaffoldMessenger.of(context).showSnackBar(
      const SnackBar(content: Text("Please log in to save profile")),
    );
    return;
  }
  final userId = user.uid;

  final profileData = {
    'name': _nameController.text.trim(),
    'birthdate': _birthdateController.text.trim(),
    'gender': _genderController.text.trim(),
    'ethnicity': _ethnicityController.text.trim(),
    'location': _locationController.text.trim(),
    'grade': _gradeController.text.trim(),
    'school': _schoolController.text.trim(),
    'subjects': _subject,
    'hobby': _hobbies,
    'skill': _skills

  };

  try {
    await FirebaseFirestore.instance.collection('students').doc(userId).set(profileData);
    ScaffoldMessenger.of(context).showSnackBar(
      const SnackBar(content: Text("Profile saved!")),
    );
  } catch (e) {
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text("Error saving profile: $e")),
    );
  }
}
```

Figure 5. Screenshot of code 2

The user profile data is saved through a function called _saveProfile, which is executed when the user taps the “Save” button after completing their profile. Each text input field is linked to a controller that retrieves the user's responses. The function collects these values and formats them into a structured data object.

This object is then uploaded to Firebase Firestore, where it is stored under the authenticated user's unique ID. This stored profile becomes a key input for the AI recommendation engine, ensuring that every opportunity is evaluated based on accurate and personalized user data. By using Firebase's real-time database and Flutter's state management, the system ensures seamless data handling and updates [13].

The database component serves as the central repository for all user profiles and opportunity listings, including internships, volunteering, activism events, and mental health resources. This component is built using Firebase Firestore, which allows for scalable, real-time data storage and retrieval. Each opportunity entry contains detailed fields such as eligibility criteria, subject focus, location, and type. These structured entries make it easier for the AI to evaluate and score opportunities accurately. The database also supports API calls from the AI sorting function, enabling personalized opportunity matching for each user.



```
Future<void> fetchAndSortOpportunities() async {
  final profile = await fetchUserProfile();
  if (profile == null) {
    print('User profile not found.');
    return;
  }
  userProfile = profile;

  final querySnapshot = await db.collection("collection1").where('field', isEqualTo: 'volunteering').get();
  final fetchedData = querySnapshot.docs.map((doc) {
    final data = doc.data();
    data['id'] = doc.id;
    return data;
  }).toList();

  // Show loading while scoring
  setState(() {
    results = [];
  });

  // Score all opportunities
  for (var opp in fetchedData) {
    final score = await getMatchScore(userProfile!, opp);
    opp['matchScore'] = score;
  }
```

Figure 6. Screenshot of code 3

The database functionality is triggered when the app loads a specific opportunity category page. At that point, the system fetches all relevant documents from the appropriate Firestore collection such as "volunteering" or "internships." The data is stored in a temporary list called fetchedData, which contains all available opportunities for that category.

The program then loops through each entry in fetchedData and uses the getMatchScore function to evaluate its compatibility with the user's stored profile. Each opportunity is assigned a numeric score based on AI evaluation, and these scores are used to sort the results in descending order. This ensures that users see the most relevant and personalized options first. By combining asynchronous data fetching with AI-based scoring, the system delivers efficient, real-time recommendations [15].

4. Experiment

4.1. Experiment 1

A potential blind spot in the program would be the AI's consistency and accuracy. It is important this works well because we need to ensure the user gets the best choices and rankings from their information. If it is inconsistent and the rankings are different every time, then that could cause the user to struggle finding ones that are actually tailored to them.

We can run an experiment using 4 different versions of the scoring prompt, each increasing in structure and details. These include: a very general prompt, a basic structured prompt, a moderately detailed prompt, and a hyper specific prompt. The independent variable is the level of prompt complexity, and the dependent variable is the consistency of results across multiple runs. I will use the same user profile and opportunities, running each prompt 4-5 times, and then comparing the average percentage overlap of the top 5 opportunities from each run to see how consistent it is. This helps us get a deep idea of what helps reduce bias and randomness from the AI model.

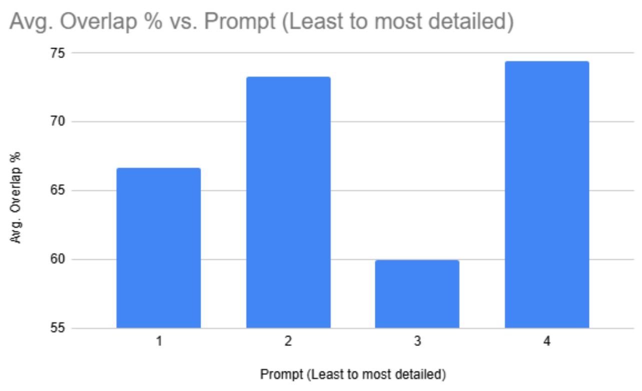


Figure 7. Figure of experiment 1

The mean of the overlap percentage across all prompts is 68.6%, and the median is 70%. The lowest value is 60% from prompt 3, and the highest is 74.4% from prompt 4. What was surprising was the huge drop in consistency for prompt 3 despite it being more specific than 1 and 2. This suggests that introducing extra details and complexity without clarifying with exact instructions (such as a grading scale that was added in prompt 4), may have confused the model and caused more variability in its results. More detail isn’t always better and what matters more is how clearly that detail is presented. The biggest effect on my results came from how precisely and consistently the prompt given to ChatGPT helped guide the scoring process, and not just how much more details was added to the prompt.

4.2. Experiment 2

Another potential blind spot is the accuracy of the mental health map. It needs to be reliable in providing the user with actual mental health providers and not give back general doctors or just random clinics.

We can run an experiment using different 4-5 different google map search keywords. These keywords, “therapy”, “mental health”, “psychiatrist”, “counseling”, and “psychologist”, will serve as the independent variable. I will collect the locations from each keyword and evaluate whether each result is actually relevant. The dependent variable will be the percentage of results that were actually related to mental health. This helps us determine which keyword is most accurate and consistent in giving the user the actual mental health resources they need.

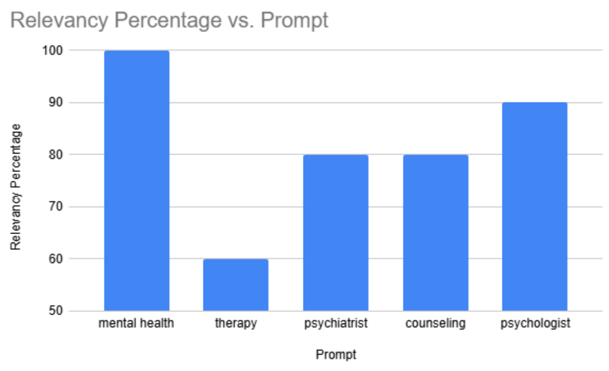


Figure 8. Figure of experiment 2

The mean of the relevance scores is 82%, and the median is 80. The lowest value is 60% for the therapy keyword, and highest is mental health with 100% relevancy. The most surprising result was how poorly the keyword “therapy” performed. This is probably because it is a very broad term that could relate to many things such as sports and just general health. The biggest effect on my results came from how detailed the terms were. More specific terms like “mental health” and “psychologist” came back with highly relevant results compared to the broader terms. The more specific the keyword is, the less likely for random clinics unrelated to mental health to show up.

5. Related Work

Cao et al. (2024) made a method to reduce bias in AI outputs by identifying and subtracting bias linked to prompt phrasing [4]. They analyze how the model responds to a “blank” prompt, or a prompt that has major key words removed, to reveal its built in biases. They then subtract these biases from the actual prompt embedding before generating a response. However, this method works mainly for fact-based tasks and struggles with open-ended prompts. Our project improves on this by using a hyper specific scoring prompt to reduce as much open-ended thinking as possible in order to boost consistency.

Wei et al. (2022) used Chain-of-Thought prompting to reduce randomness in AI models [5]. By making the model reason step-by-step through its own answers, it becomes more focused and less likely to provide inconsistent answers. It improved reproducibility, but only in fact-based tasks and not so much open-ended reasoning. Our project builds on this by using a hyper specific weighted scoring system that makes the AI grade four criteria for each individual opportunity, and then averages out the four scores to provide a final match score between 0 and 1. This tries to reduce as much open-ended thinking as possible.

Tang et al. (2023) used permutation self-consistency to solve inconsistent ranking outputs of AI models by shuffling and re-ranking the same list multiple times, then averaging the results to create a ranking that was 7-18% more consistent [6]. A limitation of this method is that it requires multiple model calls, making it time consuming as well as computationally expensive. Our app improves on this by using a fixed weighted scoring prompt that assigns numeric scores to each item based on consistent criteria. These scores are then averaged to make a more consistent final ranking without having to go through multiple iterations.

6. CONCLUSIONS

One limitation of the project is the randomness and inconsistency of ChatGPT's outputs when ranking the opportunities. Even when using a hyper specific prompt, we still only get around a 74% consistency rate. This is because large language models like ChatGPT have built-in randomness and biases that help it be more creative and understanding. If I had more time, I would experiment more with ChatGPT's API and prompts to try and reduce the randomness [14]. Things such as prompt chaining, where you extract keywords and features from each opportunity and then send another prompt to grade those features, could potentially make it more accurate and consistent. Running the AI multiple times, storing its rankings, and then averaging out the scores could also potentially boost our consistent rate.

In conclusion, this app helps users find opportunities and mental health resources by using AI. It allows users to make easier decisions without the frustration of having to look manually online. By promoting this app, it can help save tons of effort for people while providing reliable and personalized guidance.

REFERENCES

[1] Schaeffer, Katherine. "Black workers' views and experiences in the US labor force stand out in key ways." (2023).

[2] Shastri, Andrey. "Gender inequality and women discrimination." IOSR Journal of Humanities and social science 19.11 (2014): 27-30.

[3] Hinings, Bob, Thomas Gegenhuber, and Royston Greenwood. "Digital innovation and transformation: An institutional perspective." Information and organization 28.1 (2018): 52-61.

[4] Xu, Ziyang, et al. "Take care of your prompt bias! investigating and mitigating prompt bias in factual knowledge extraction." arXiv preprint arXiv:2403.09963 (2024).

[5] Wei, Jason, et al. "Chain-of-thought prompting elicits reasoning in large language models." Advances in neural information processing systems 35 (2022): 24824-24837.

[6] Tang, Raphael, et al. "Found in the middle: Permutation self-consistency improves listwise ranking in large language models." arXiv preprint arXiv:2310.07712 (2023).

[7] Welsby, Philip, and Bernard MY Cheung. "ChatGPT." Postgraduate Medical Journal 99.1176 (2023): 1047-1048.

[8] Hoang, Yen Nhi, et al. "Consistency and accuracy of artificial intelligence for providing nutritional information." JAMA network open 6.12 (2023): e2350367-e2350367.

[9] Moroney, Laurence. "Using authentication in firebase." The Definitive Guide to Firebase: Build Android Apps on Google's Mobile Platform. Berkeley, CA: Apress, 2017. 25-50.

[10] Rathore, Bharati. "Usage of AI-powered marketing to advance SEO strategies for optimal search engine rankings." Eduzone: International Peer Reviewed/Refereed Multidisciplinary Journal 5.1 (2016): 30-35.

[11] Roumeliotis, Konstantinos I., and Nikolaos D. Tselikas. "Chatgpt and open-ai models: A preliminary review." Future Internet 15.6 (2023): 192.

[12] Challen, Robert, et al. "Artificial intelligence, bias and clinical safety." BMJ quality & safety 28.3 (2019): 231-237.

[13] Moroney, Laurence. "The firebase realtime database." The Definitive Guide to Firebase: Build Android Apps on Google's Mobile Platform. Berkeley, CA: Apress, 2017. 51-71.

[14] Rawashdeh, Ahmad, Omar Rawashdeh, and Mohammad Rawashdeh. "ChatGPT and ChatGPT API: An Experiment with Evaluating ChatGPT Answers." Proceedings of the Future Technologies Conference. Cham: Springer Nature Switzerland, 2024.

[15] Ferrara, Steve, and Saed Qunbar. "Validity arguments for AI-based automated scores: Essay scoring as an illustration." Journal of Educational Measurement 59.3 (2022): 288-313.