GOLFGLOW: AN OFFLINE-FIRST MOBILE APP FOR AMATEUR GOLF TOURNAMENT MANAGEMENT AND PERSONAL PROGRESS TRACKING

Xilai Zhang[1], Mirna Shabo [2]

[1] Horizon High School, 10393 Seidel Rd, Winter Garden, FL 34787
[2] California State Polytechnic University, Pomona, CA, 91768

ABSTRACT

GolfGlow is a mobile application designed to help golfers manage tournaments, track player rankings, and log personal growth through an accessible, offline-first platform [1]. Developed using Flutter and relying on local storage via Shared Preferences, the app eliminates the need for internet connectivity while maintaining a user-friendly interface [2]. It features three core systems: a tournament management system for scheduling events, a ranking system for recording and updating player scores, and a profile management system for storing basic user information [3]. To evaluate its effectiveness, a survey-based experiment involving five users was conducted, yielding an overall satisfaction score of 4.12 out of 5. The results highlighted strong usability, particularly in event creation and ease of navigation, while also identifying app responsiveness as an area for improvement. Compared to existing web-based or hardware-dependent golf tools, GolfGlow offers a lightweight and inclusive solution ideal for amateur players. Its offline capability and simplicity make it a practical alternative for self-guided golf engagement and progress tracking.

KEYWORDS

Offline mobile application, Golf tournament management, Player ranking system, User experience evaluation

1. INTRODUCTION

The development of the GolfGlow application was inspired by a personal interest in golf and the recognition of a gap in accessible, user-friendly digital tools to support the sport. While golf continues to grow in popularity—evidenced by the PGA Tour's revenue surpassing 1.8 billion U.S. dollars in 2023 and over 238,000 U.S. high school students participating in the sport—there remains a lack of mobile applications tailored to everyday golfers for managing tournaments, tracking progress, and maintaining performance data. This issue primarily affects amateur golfers, fans, and local tournament organizers who often lack streamlined tools for recording schedules, rankings, or personal development metrics. Without such tools, individuals may lose motivation to engage in the sport or struggle to improve due to limited tracking mechanisms. Fans might miss key events, and organizers may find it difficult to communicate tournament logistics effectively. Addressing this problem is essential to fostering a more inclusive and supportive environment for golf enthusiasts of all skill levels. By offering a mobile solution that operates

David C. Wyld et al. (Eds): COMIT, AISO, CRBL, WIMNET, EDUPT– 2025
pp. 97-108, 2025. CS & IT - CSCP 2025 DOI: 10.5121/csit.2025.151208

offline and emphasizes simplicity, GolfGlow aims to democratize access to golf management tools and enhance user engagement, both in casual play and in structured competitive settings.

Three external methodologies were analyzed to assess how GolfGlow compares to existing technological solutions in the golfing domain. One study focused on the behavioral effects of smartphone golf app usage among professional players, particularly regarding performance metrics like swing accuracy and distance [4]. While informative, it did not introduce a practical application, highlighting GolfGlow's advantage as a fully functional, user-ready tool. Another approach involved a web-based league management system designed to coordinate events and track leaderboards [5]. Although effective in formal settings, it required internet connectivity and lacked mobile and offline capabilities, limiting its accessibility. A third method introduced a wearable device that provided biomechanical feedback on golf swings using motion sensors. While offering detailed analytics, this solution was hardware-dependent and catered primarily to high-level athletes.

In contrast, GolfGlow is a mobile, offline-first solution that supports a wide user base without the need for internet access or specialized equipment. By focusing on manual input, local data storage, and ease of use, GolfGlow fills a critical gap between highly specialized technologies and casual user needs. Its emphasis on accessibility and simplicity makes it a more versatile and inclusive alternative for golfers seeking self-directed progress tracking and tournament management.

GolfGlow is a mobile application designed to assist golfers in managing tournament events, tracking player rankings, and logging personal growth. The app was developed with the goal of providing an accessible, user-centered platform that functions entirely offline [6]. Users can manually input data related to golf tournaments, including names, dates, times, and descriptions, as well as maintain profiles and monitor skill development through logged games and practice sessions. The design prioritizes simplicity and ease of use, targeting both amateur and professional users who may not have access to complex or internet-dependent golf tools. Data is stored locally on the device using SharedPreferences, ensuring that users retain access to their information without relying on backend infrastructure or internet connectivity.

This approach directly addresses the core problem of limited accessibility in golf-related digital tools by eliminating barriers such as cost, connectivity, or technical complexity. Unlike generic productivity apps such as calendars or notes, GolfGlow is tailored specifically to the needs of golfers, incorporating features that are directly relevant to the sport, such as ranking and tournament tracking. Furthermore, its offline capability makes it especially appealing for users in areas with limited connectivity. Compared to existing tools that focus on either elite-level performance analytics or league administration, GolfGlow provides a personalized, lightweight, and effective alternative for everyday golf engagement.

To evaluate the user experience of the GolfGlow application, a survey-based experiment was conducted with the primary objective of assessing the app's usability, clarity, and functional effectiveness across various features. The experiment involved the distribution of a 10-question survey, with each question rated on a scale from 1 to 5. Five participants were selected to use the application and respond to the survey based on their experience. The questions covered essential areas such as ease of navigation, event and ranking creation, responsiveness, usefulness, and overall user satisfaction.

The results indicated a strong overall performance, with an average rating of 4.12 out of 5 across all questions and participants. Users particularly praised the event creation feature, user-friendliness, and the likelihood of recommending the app to others, each receiving average scores of 4.4. The only noted area for improvement was application responsiveness, which received the lowest average score at 3.4. This suggests the need for enhanced feedback mechanisms or UI responsiveness in future iterations. Overall, the experiment was successful in validating the core features of GolfGlow while identifying clear directions for refinement based on user input.

2. CHALLENGES

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

2.1. Tournament Management System Concern

A potential concern regarding the tournament management system within GolfGlow is the reliability of storing all data locally on the user's device. Skeptics may question how users can trust that their tournament data will remain safe and accessible if it is not backed up to a cloud-based service. This concern is addressed by utilizing SharedPreferences, which provides a persistent storage mechanism that retains data even when the application is closed or the device is restarted. Although this approach does not include multi-device syncing or automatic cloud backups, it ensures that users maintain control over their data without dependency on an internet connection. The decision to use local storage supports GolfGlow's commitment to offline accessibility, simplicity, and lightweight performance. Furthermore, should user needs evolve, the application can be expanded in the future to include optional cloud synchronization features, providing greater flexibility while maintaining its core functionality.

2.2. Ranking Management System Concern

A common skepticism regarding the ranking management system in GolfGlow pertains to the manual input of player scores rather than using automated scoring tools integrated with golf hardware. Critics may argue that relying on user-entered data introduces the potential for inaccuracy and inefficiency. However, the manual entry system was intentionally designed to emphasize user engagement and offline functionality. By allowing users to input and manage scores directly, the application fosters a sense of ownership and active participation in tracking performance. This also eliminates the need for expensive or specialized equipment, making the application accessible to a broader range of users. The ranking system dynamically updates based on the manually entered data and is stored locally via SharedPreferences, ensuring persistence without reliance on network connectivity. While the current implementation prioritizes simplicity and inclusivity, the application is designed with scalability in mind; future updates may introduce optional integrations with digital golf tools for those seeking automated tracking.

2.3. Profile Management System

One concern that may arise regarding the profile management system in GolfGlow is the security and reliability of storing personal user data locally rather than through a secured cloud service. Critics might argue that this approach limits data protection and could pose risks in the event of device failure or app deletion. However, the decision to store profile data locally using SharedPreferences aligns with the application's offline-first design philosophy. This method ensures that all data remains confined to the user's device, thereby minimizing exposure to external security threats and eliminating dependency on network access. It also enhances performance and simplifies the user experience, particularly for those in areas with limited or unreliable internet connectivity. While the current local storage strategy prioritizes autonomy and lightweight operation, future development plans include the possibility of integrating secure cloud storage and authentication options. This would allow users to choose between local-only storage or cloud-based data synchronization, depending on their preferences and security needs.

3. SOLUTION

Upon launching the GolfGlow application, users are presented with a home screen that features a welcome message and a display of previously created tournaments. This interface also includes a persistent navigation bar at the bottom of the screen, which facilitates access to five primary modules: Home, Events, Ranking, Growth, and Profile. Each module is designed to support a specific aspect of user engagement. The Home module serves as an overview hub, summarizing existing tournaments. The Events module enables users to create and manage new tournament entries by manually inputting relevant information such as the tournament name, date, time, and description. Once submitted, these tournaments are stored locally and reflected both in the Events module and on the Home screen. The Ranking module allows for manual entry of player names and corresponding scores; the system then dynamically updates player rankings based on cumulative scores and game participation. The Growth module offers functionality for users to document personal development activities, including games played, equipment utilized, and practice sessions completed. Lastly, the Profile module facilitates the creation and modification of user profiles, capturing essential details such as name, age, and gender. All user data is stored locally on the device through SharedPreferences, ensuring full offline capability. The user interface is responsive to data changes, automatically updating to reflect modifications made within any of the modules. This design prioritizes accessibility, simplicity, and autonomy, allowing users to engage with the application without reliance on internet connectivity or cloud-based services [10].

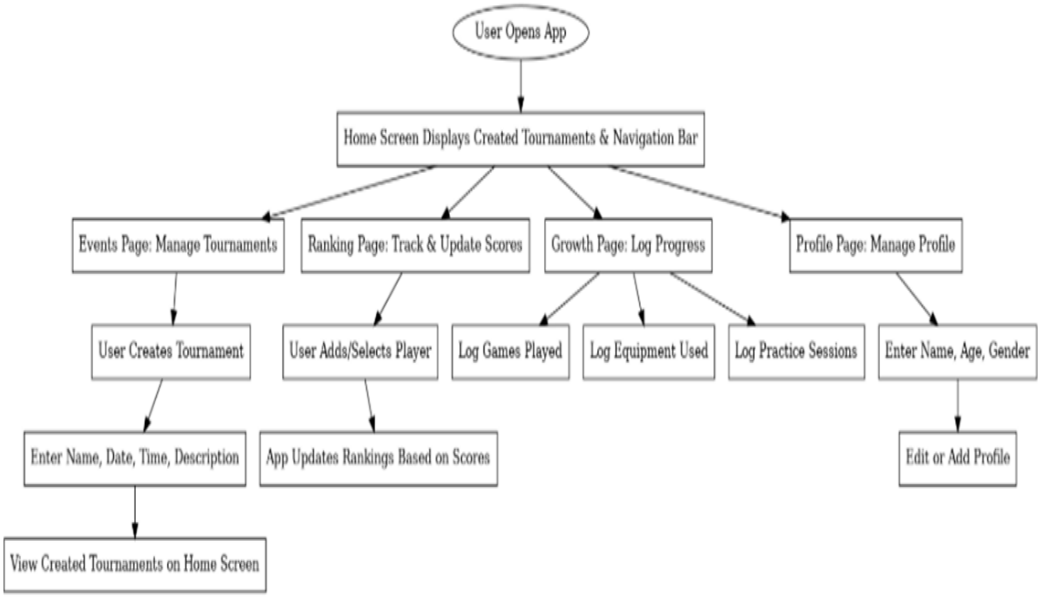

User Opens App
Home Screen Displays Created Tournaments & Navigation Bar
Events Page: Manage Tournaments
Ranking Page: Track & Update Scores
Growth Page: Log Progress
Profile Page: Manage Profile
User Creates Tournament
User Adds/Selects Player
Log Games Played
Log Equipment Used
Log Practice Sessions
Enter Name, Age, Gender
Enter Name, Date, Time, Description
App Updates Rankings Based on Scores
Edit or Add Profile
View Created Tournaments on Home Screen

Figure 1. Overview of the solution

The first component is Tournament Management System. The tournament management system allows the user the creation and management of tournaments by manually inputting tournament’s names, dates, and description. The system stores the data locally, for users to access and view on the homescreen. The system ensures the user can access their tournament information at any time.

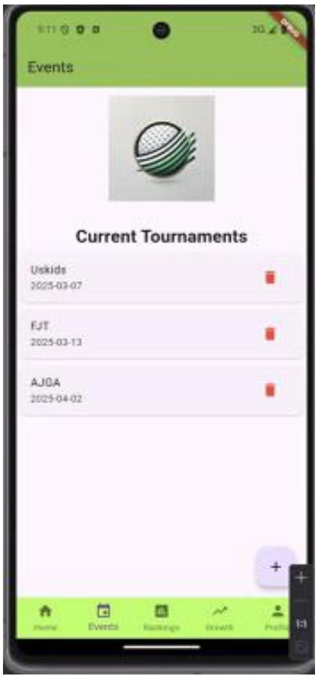
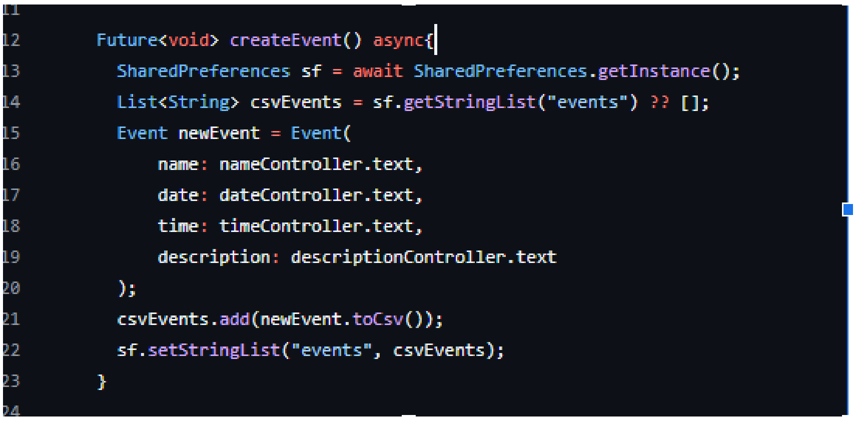


Figure 2. Screenshot of the APP 1

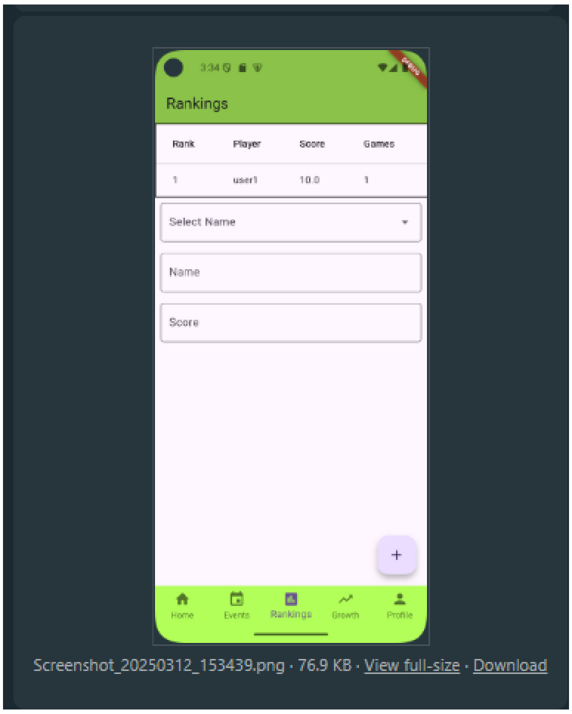


```
    Future<void> createEvent() async{
      SharedPreferences sf = await SharedPreferences.getInstance();
      List<String> csvEvents = sf.getStringList("events") ?? [];
      Event newEvent = Event(
          name: nameController.text,
          date: dateController.text,
          time: timeController.text,
          description: descriptionController.text
      );
      csvEvents.add(newEvent.toCsv());
      sf.setStringList("events", csvEvents);
    }
```

Figure 3. Screenshot of code 1

The core functionality of this code segment is to facilitate the creation and local storage of golf tournament events within the application. This is accomplished using the SharedPreferences API, which enables lightweight key-value data storage on the user’s device. The process begins with the asynchronous retrieval of the SharedPreferences instance, ensuring that stored data can be accessed and modified reliably. Next, the code attempts to load any previously saved events from local storage using the key "events". If no data exists under that key, it initializes an empty list, thereby maintaining data consistency and preventing runtime errors. The user-generated tournament data—comprising the tournament name, date, time, and description—is then encapsulated into an Event object. This object is subsequently serialized into a CSV-like format and appended to the list of existing events. Finally, the updated list is written back to local storage using sf.setStringList("events", csvEvents), which persists the data across app sessions. This design choice ensures that all tournament information remains available to the user even when the application is closed or offline. The entire process is optimized for simplicity and offline functionality, reinforcing the application's core design principles of accessibility and ease of use without reliance on external databases or backend services.

The second component is the ranking management system. The purpose of the ranking management system is to enable users to manually add player scores and dynamically update ranking based on inputted scores. The user is also able to keep track of the ranking and scores. The players are ranked without an internet connection, causing it to be more efficient.



Rankings

Rank	Player	Score	Games
1	user1	10.0	1

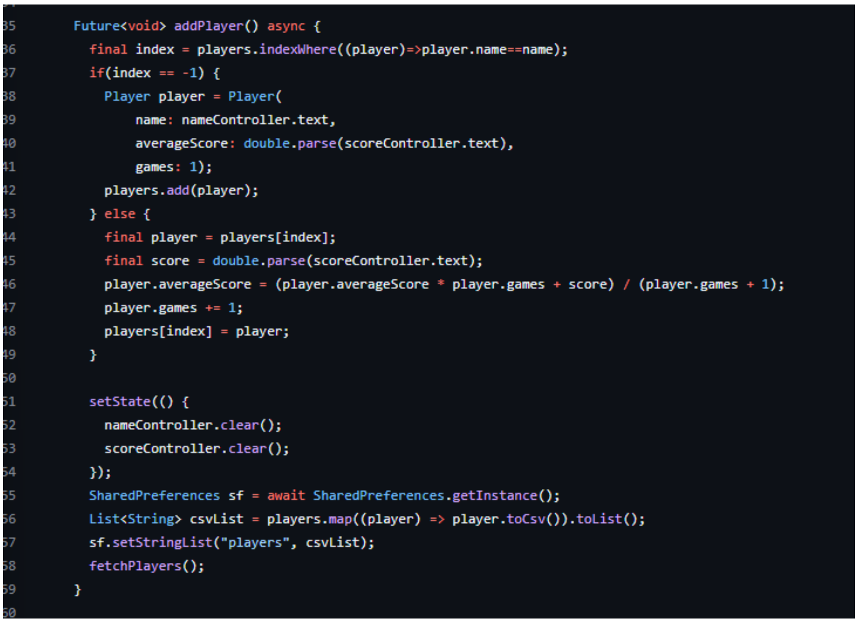
Select Name

Name

Score

Screenshot_20250312_153439.png · 76.9 KB · View full-size · Download

Figure 4. Screenshot of APP 2



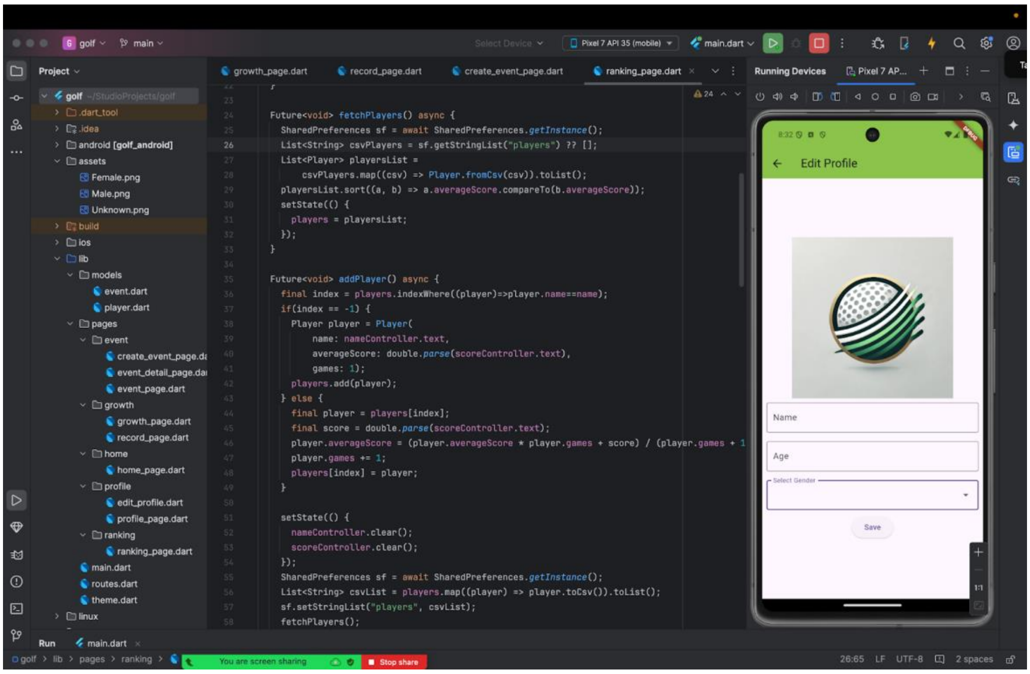
```
    Future<void> addPlayer() async {
      final index = players.indexWhere((player)=>player.name==name);
      if(index == -1) {
        Player player = Player(
            name: nameController.text,
            averageScore: double.parse(scoreController.text),
            games: 1);
        players.add(player);
      } else {
        final player = players[index];
        final score = double.parse(scoreController.text);
        player.averageScore = (player.averageScore * player.games + score) / (player.games + 1);
        player.games += 1;
        players[index] = player;
      }

      setState(() {
        nameController.clear();
        scoreController.clear();
      });
      SharedPreferences sf = await SharedPreferences.getInstance();
      List<String> csvList = players.map((player) => player.toCsv()).toList();
      sf.setStringList("players", csvList);
      fetchPlayers();
    }
```

Figure 5. Screenshot of code 2

The primary function of this code segment is to manage the addition and updating of player rankings within the application. Utilizing the SharedPreferences API, the code ensures that all ranking data is stored locally on the device, allowing the app to function without internet connectivity. The process begins by checking whether the player already exists in the rankings list. This is achieved using the indexWhere() method, which searches for an existing entry matching the player's name. If no such entry is found, a new player object is instantiated with the user-inputted name and score and added to the rankings list. If the player already exists, the function recalculates the player's average score to reflect the new input and updates the existing object accordingly. Once the player data is modified or added, the updated rankings list is saved back to SharedPreferences, ensuring persistence of the changes. The function concludes by invoking fetchPlayers(), which refreshes the rankings displayed in the user interface. This local, dynamic ranking system enables users to maintain and monitor player performance manually and efficiently, aligning with the application’s offline-first design philosophy and emphasizing user autonomy and simplicity.

The third component is the profile management system [9]. The purpose of this system is allowing users to create and input profile details such as name, age, and gender. The profile management system uses SharedPreferences to save data locally on the user’s device. The system does not require any backend or internet to be functional.

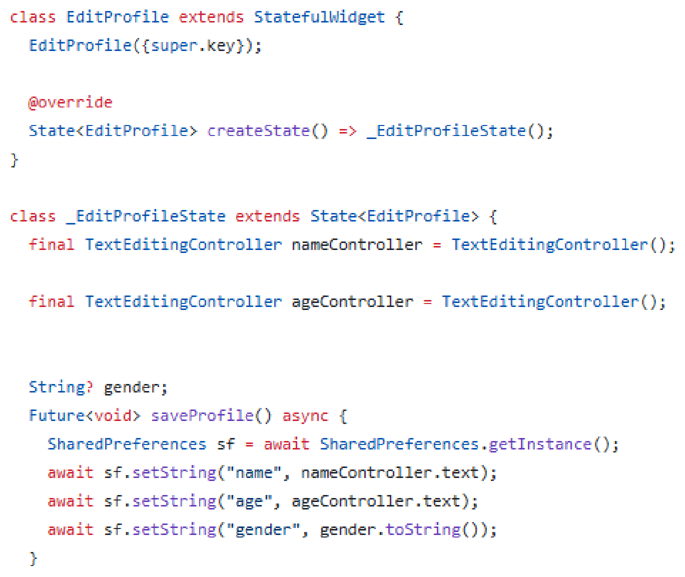


```
  Future<void> fetchPlayers() async {
    SharedPreferences sf = await SharedPreferences.getInstance();
    List<String> csvPlayers = sf.getStringList("players") ?? [];
    List<Player> playersList =
        csvPlayers.map((csv) => Player.fromCsv(csv)).toList();
    playersList.sort((a, b) => a.averageScore.compareTo(b.averageScore));
    setState(() {
      players = playersList;
    });
  }

  Future<void> addPlayer() async {
    final index = players.indexWhere((player)=>player.name==name);
    if(index == -1) {
      Player player = Player(
          name: nameController.text,
          averageScore: double.parse(scoreController.text),
          games: 1);
      players.add(player);
    } else {
      final player = players[index];
      final score = double.parse(scoreController.text);
      player.averageScore = (player.averageScore * player.games + score) / (player.games + 1
      player.games += 1;
      players[index] = player;
    }

    setState(() {
      nameController.clear();
      scoreController.clear();
    });
    SharedPreferences sf = await SharedPreferences.getInstance();
    List<String> csvList = players.map((player) => player.toCsv()).toList();
    sf.setStringList("players", csvList);
    fetchPlayers();
```

Figure 6. Screenshot of APP 3



```
class EditProfile extends StatefulWidget {
  EditProfile({super.key});

  @override
  State<EditProfile> createState() => _EditProfileState();
}

class _EditProfileState extends State<EditProfile> {
  final TextEditingController nameController = TextEditingController();

  final TextEditingController ageController = TextEditingController();


  String? gender;
  Future<void> saveProfile() async {
    SharedPreferences sf = await SharedPreferences.getInstance();
    await sf.setString("name", nameController.text);
    await sf.setString("age", ageController.text);
    await sf.setString("gender", gender.toString());
  }
```

Figure 7. Screenshot of code 3

This section of code is responsible for managing user profile data within the application, allowing users to input and store personal information such as name, age, and gender [7]. The profile management system is implemented using a stateful widget, which enables real-time updates to the user interface based on the state of the profile form. The widget's state is managed through the EditProfileState class, which handles the logic for capturing and saving user inputs. Text input fields are controlled using TextEditingController instances for both name and age, allowing the application to track changes and access the text content efficiently. The selected gender is stored as a nullable String, capturing the user's choice from a predefined list of options. When the user initiates the save action, the saveProfile() function is triggered. This function uses SharedPreferences.getInstance() to access the device's local storage and persist the user's input. Key-value pairs are saved using methods such as sf.setString(), storing the name, age, and gender securely on the local device. By eliminating the need for internet connectivity or cloud services, this approach ensures that user data remains private and accessible even in offline environments [8]. The local storage design also aligns with the app's goal of being lightweight and highly usable for individual golfers without reliance on backend infrastructure.

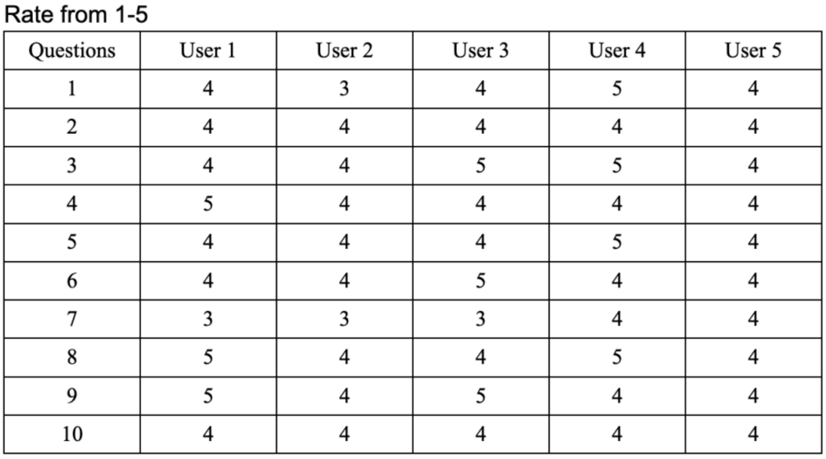
4. EXPERIMENT

This experiment was designed to assess the application's usability, functionality, and overall user experience across a general audience. Conducting this experiment is critical for understanding how effectively the application meets user expectations and identifying areas in need of improvement. Additionally, by gathering direct input from users, the developer can obtain valuable insights and suggestions that can inform future updates and iterations of the application. This method is particularly beneficial for applications like GolfGlow that prioritize user engagement, offline access, and practical features over backend complexity.

This experiment was designed to assess the application's usability, functionality, and overall user experience across a general audience. Since the app does not rely on artificial intelligence or perform algorithmic predictions, a survey-based assessment allowed for the collection of qualitative feedback directly from users. Conducting this experiment is critical for understanding how effectively the application meets user expectations and identifying areas in need of

improvement. Additionally, by gathering direct input from users, the developer can obtain valuable insights and suggestions that can inform future updates and iterations of the application. This method is particularly beneficial for applications like GolfGlow that prioritize user engagement, offline access, and practical features over backend complexity.
Survey Questions:

How easy was it to get the purpose of the application?
How easy was it to navigate the application?
How straightforward was the event creation process?
How straightforward was the ranking creation process?
How straightforward was the growth creation process?
How useful did you find the application?
How responsive was the application when using different features?
Would you recommend the application to anyone?
Was the application user friendly and easy to use?
How comfortable did you feel when using the application?



Rate from 1-5

Questions	User 1	User 2	User 3	User 4	User 5
1	4	3	4	5	4
2	4	4	4	4	4
3	4	4	5	5	4
4	5	4	4	4	4
5	4	4	4	5	4
6	4	4	5	4	4
7	3	3	3	4	4
8	5	4	4	5	4
9	5	4	5	4	4
10	4	4	4	4	4

Figure 8. Table of experiment

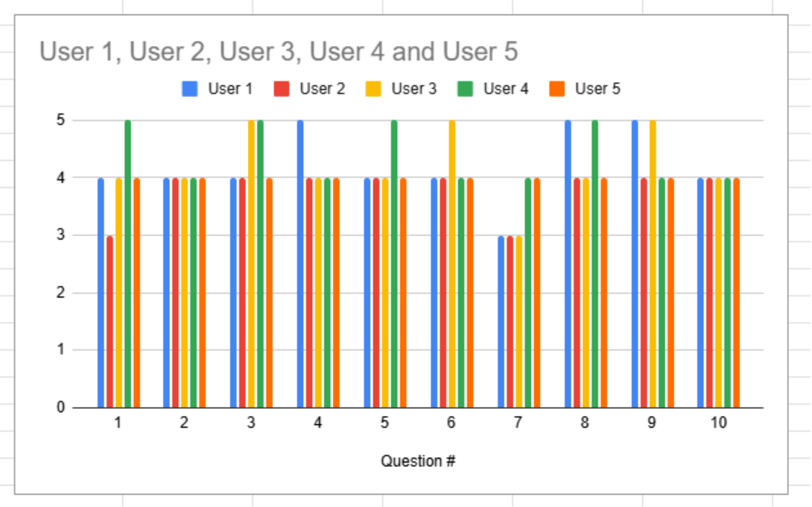


Figure 9. Figure of experiment

The user survey conducted as part of the evaluation process yielded insightful results regarding the overall functionality and user experience of the GolfGlow application. Each of the ten questions was rated on a scale from 1 to 5 by five participants. The average scores for each question were as follows: Q1 － 4.0, Q2 － 4.0, Q3 － 4.4, Q4 － 4.2, Q5 － 4.2, Q6 － 4.2, Q7 － 3.4, Q8 － 4.4, Q9 － 4.4, and Q10 － 4.0. The cumulative average across all responses was 4.12 out of 5, indicating that users generally found the application effective, intuitive, and easy to use. The highest-rated items were related to event creation, user-friendliness, and willingness to recommend the app, suggesting that GolfGlow’ s core features are well-received by users. In contrast, the lowest-scoring item pertained to the app’ s responsiveness, with an average rating of 3.4. This highlights a key area for improvement in future updates, such as implementing more dynamic UI feedback or optimizing screen transitions. Overall, the experiment was successful in identifying both the strengths and limitations of the application, offering actionable data to guide future enhancements aimed at improving user satisfaction and performance.

5. RELATED WORK

“The Effect of Smartphone App-Use Patterns on the Performance of Professional Golfers” by Jea Woog Lee, Jae Jun Nam, Kyung Doo Kang, and Doug Hyun Han explores how the usage of smartphone-based golf applications impacts player performance metrics such as swing quality, distance accuracy, and shot precision [11]. The study centers on professional golfers and aims to evaluate the behavioral and performance effects of using existing commercial apps. While the paper contributes valuable insights into the influence of technology on elite athletic performance, it does not propose or develop a new technological solution. In contrast, GolfGlow is a purpose-built mobile application that directly addresses the need for accessible, offline tools tailored to both amateur and professional golfers. It provides tournament tracking, ranking management, and personal growth logging without reliance on external systems. Unlike the research, which is observational and theoretical, GolfGlow offers a fully functional, lightweight system that empowers users to manage their golf experience independently. This practical orientation, combined with offline capability and user-centered design, positions GolfGlow as a more inclusive and actionable alternative for a broader audience.

The article “Development of a Player-Friendly Web-Based Golf League Management Application” by Devin Staton and Jeremy Bellah presents a web-based platform designed to assist golf league organizers in scheduling events and tracking leaderboard data [12]. The system prioritizes administrative functions for managing formal leagues and relies on online connectivity for operation. In comparison, GolfGlow is a mobile-first application tailored to individual users—particularly amateurs—who wish to manage personal tournaments, track scores, and monitor growth without depending on internet access. While both systems offer event scheduling features, GolfGlow differentiates itself through its offline functionality, simplified interface, and broader accessibility. Additionally, GolfGlow includes a personalized ranking system and growth-tracking module, which are absent in the web-based league manager. By focusing on casual players and prioritizing usability over formal league infrastructure, GolfGlow addresses a wider user base and offers greater flexibility in varied environments. Its mobile, offline design makes it especially effective for users seeking on-the-go, self-managed golf tracking without the constraints of web-only tools.

The article “A Wearable Device for Measuring Golf Swing Biomechanics”, developed by Jessica Rose with licensing contact David Mallin, introduces a hardware-based solution designed to analyze golf swing biomechanics in real time. This wearable device provides biomechanical feedback to users with the goal of enhancing their technique through detailed motion tracking [13]. Unlike GolfGlow, which is purely software-based and focuses on self-managed tournament scheduling, ranking, and growth tracking, the wearable device is centered on performance analytics through motion sensors. While the device offers advanced data for improving swing mechanics, it requires specialized hardware and is most beneficial to users with access to such technology. In contrast, GolfGlow is a cost-effective, offline mobile application that allows users to manually log scores and activities without the need for external devices. Its accessibility, simplicity, and broader focus make it more suitable for casual golfers or those without access to biomechanical tools. By removing hardware and connectivity barriers, GolfGlow provides a scalable and user-friendly alternative for tracking golf progress and engagement.

6. CONCLUSIONS

While GolfGlow successfully provides offline functionality and user-centered golf tracking tools, several limitations remain in its current implementation. First, all data—including tournament events, rankings, and profile information—is stored locally on the device using SharedPreferences. Although this supports offline use, it also means that users risk permanent data loss if the application is deleted or the device is compromised, due to the absence of a backup or cloud synchronization mechanism [14]. Second, the application lacks real-time features such as push notifications, automatic reminders, or live updates, which could enhance user engagement and functionality. Third, the user interface does not yet provide visual confirmation feedback (e.g., pop-up notifications) when actions such as saving data or creating events are completed, potentially leading to confusion about whether tasks were successful.

Given additional development time, several enhancements would be prioritized. These include redesigning the user interface to improve visual clarity, adding support for profile pictures, and implementing optional cloud backup via Firebase or a similar backend service [15]. Expanding language options for global users and integrating with wearable fitness devices such as the Apple Watch would further increase the app's utility. If starting the project anew, greater attention would be placed on optimizing the Growth module to ensure it delivers meaningful insights for skill development. These improvements would strengthen GolfGlow's reliability, scalability, and appeal to a wider audience while preserving its offline-first design philosophy.

REFERENCES

[1] Hoehle, Hartmut, and Viswanath Venkatesh. "Mobile application usability." MIS quarterly 39.2 (2015): 435-472.

[2] Casini, Marco, Domenico Prattichizzo, and Antonio Vicino. "The automatic control telelab: A user-friendly interface for distance learning." IEEE Transactions on education 46.2 (2003): 252-257.

[3] Zainal, Nurezayana. "The Development of Santai Esport Tournament Management System." Applied Information Technology And Computer Science 5.2 (2024): 1103-1120.

[4] Lee, Jea Woog, et al. "The effect of smartphone app-use patterns on the performance of professional golfers." Frontiers in Psychology 12 (2021): 678691.

[5] Staton, Devin, and Jeremy Bellah. "Development of Player-Friendly Web-Based Golf League Management Application." Journal of Software Engineering and Applications 15.3 (2022): 59-74.

[6] García-Holgado, Alicia, et al. "User-centered design approach for a machine learning platform for medical purpose." Iberoamerican Workshop on Human-Computer Interaction. Cham: Springer International Publishing, 2021.

[7] Fallatah, Khalid U., Mahmoud Barhamgi, and Charith Perera. "Personal data stores (PDS): a review." Sensors 23.3 (2023): 1477.

[8] Chen, Ruichuan, et al. "Towards statistical queries over distributed private user data." 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12). 2012.

[9] Kannan, P. "Semantic-based Researcher Profile Management System: Case Study on VIVO." Retrieved February 16 (2015): 2019.

[10] Babu, Shaik Masthan, A. Jaya Lakshmi, and B. Thirumala Rao. "A study on cloud based Internet of Things: CloudIoT." 2015 global conference on communication technologies (GCCT). IEEE, 2015.

[11] Lee, Jea Woog, et al. "The effect of smartphone app-use patterns on the performance of professional golfers." Frontiers in Psychology 12 (2021): 678691.

[12] Staton, Devin, and Jeremy Bellah. "Development of Player-Friendly Web-Based Golf League Management Application." Journal of Software Engineering and Applications 15.3 (2022): 59-74.

[13] Kim, Sung Eun, et al. "Validation of inertial measurement units for analyzing golf swing rotational biomechanics." Sensors 23.20 (2023): 8433.

[14] Choi, Hyoung Yong, and Sang Soo Kim. "How is satisfaction with online-to-offline app formed? Importance of confirmation through offline experience." Sage Open 12.4 (2022): 21582440221134899.

[15] Chougale, Pankaj, et al. "Firebase-overview and usage." International Research Journal of Modernization in Engineering Technology and Science 3.12 (2021): 1178-1183.