

Software Engineering Department

ORT Braude College

Capstone Project Phase A - 61998

**System for smart common sport grounds management**

Obied Haddad-206989295

[Obied.Haddad@e.braude.ac.il](mailto:Obied.Haddad@e.braude.ac.il)

Ebrahem Enbtawe-208047357

[Ebrahem.Enbtawe@e.braude.ac.il](mailto:Ebrahem.Enbtawe@e.braude.ac)

Advisor: Zeev Frenkel

GIT Link: <https://github.com/obiedh/final-project.git>

2024

[**Abstract 3**](#_pa49156ashs2)

[**1. Introduction 4**](#_3pj1aypa1rqo)

[**2. Background and Related Work 5**](#_2wy9iss8bhis)

[2.1 The Most Popular Existing Solutions: 5](#_5cewwdwe87d8)

[Table 1: Comparative Analysis of Sports Field Reservation Systems 6](#_vlayn9qa9ktq)

[**Data Structure 7**](#_chpvpiqc1uay)

[**3 Algorithms 10**](#_57jjastcox7m)

[3.1 Greedy Algorithm for Field Matching: 10](#_ggp4a1j2mhue)

[3.2 Collaborative Filtering Recommendation System: 12](#_nn5aks5gierx)

[3.3 Automated Conflict Resolution: 14](#_kkmsyd91sj85)

[**4. Expected Achievements 17**](#_m4f3epgl9k3u)

[4.1 Outcomes: 17](#_svdei09yjr00)

[4.1.1 Optimized Reservation Matching Algorithm: 17](#_qfngw2naog8m)

[4.1.2 Real-time Conflict Resolution and Notification System: 17](#_74a1ybm18s3b)

[4.1.3 Dynamic User Experience through Collaborative Filtering: 17](#_687gtlx0rlgo)

[4.2 Unique Features: 18](#_a88ql9oxibju)

[4.2.1 Intelligent Reservation Matching Algorithm: 18](#_gmguivnfq7hp)

[4.2.2 Automated Real-Time Reservation Monitoring and Notifications: 18](#_rhwhfx77oi3h)

[4.2.3 Dynamic User Experience through Collaborative Filtering: 18](#_3v5lglmg4kux)

[4.3 Criteria for Success: 19](#_g6qwfsvl8123)

[4.3.1 System performance: 19](#_k77bd4pd9aep)

[4.3.2 Efficiency Metrics: 19](#_x10l6kf18fhy)

[4.3.3 Security and Privacy: 19](#_eghpedlcp4z3)

[4.3.4 User-Friendly Interface: 19](#_qpu4e3r3a9dq)

[**5 The Process 20**](#_py1raohcr90c)

[5.1 Research – Field Reservation Management 20](#_sujndudxho0y)

[5.1.1 Constraints and Challenges – Field Reservation Management: 21](#_3q09r6bwkhr7)

[5.1.1.1 Proximity: 21](#_4wfkednswl2f)

[5.1.1.2 Availability: 21](#_i1u3lckpyfdi)

[5.2 Methodology and Development Process 22](#_o9sy4iiy0vr8)

[5.2.1 Planning and Preparation: 22](#_o37epj1rehdo)

[5.2.2 Implementation: 22](#_lzrlr2ia3ua1)

[5.2.3 Iterative Development: 22](#_vbr3t93gx89k)

[**6 The Product: 23**](#_qydo2670uucx)

[6.1 Requirements: 23](#_se9fbb3hld0g)

[Table 2: System Requirements for the Sports Facility Management System 23](#_ydzj3xrcphp8)

[6.2 UML Diagrams 25](#_lhab0yphz25q)

[6.2.1 Use Case Diagram 25](#_aegvl3ccxqh3)

[6.2.2 Activity Diagram - Field Reservation Flow 26](#_mc842rqtvp5j)

[6.2.3 Class Diagram - Entities 26](#_5vdsz3e0mak5)

[6.3 Draft of GUI graphical user interface. 27](#_bsqcprolcs6c)

[**7. Verification and Evaluation 44**](#_pcvd441p7j29)

[7.1 Table 3:Unit Tests 44](#_5bkfo3uowbtc)

[**8. References 45**](#_wxo9j7xv46i)

# 

# Abstract

Our system is an advanced solution that simplifies the process of reserving tennis courts and fields. By utilizing state-of-the-art algorithms and real-time tracking features, the app creates a seamless connection between field managers and sports enthusiasts. Managers can efficiently handle reservations through a user-friendly interface, while users can easily discover and book fields that meet their needs.

The app's development aims to unify and synchronize the fragmented landscape of existing reservation solutions. By addressing the limitations of current systems, our application provides a comprehensive and integrated approach to sports field reservation management. This includes optimizing data structures and implementing a robust database structure for efficient organization and storage, ensuring seamless access to reservation-related information.

Furthermore, our app ensures accessibility for field managers and sports enthusiasts alike, enhancing the overall sports field reservation experience. It offers a more intuitive, efficient, and enjoyable platform for all stakeholders involved. We invite managers to explore the future of field reservations with our app.

# 

# 

# 1. Introduction

Efficient sports field reservation for players can be a complex task, particularly for businesses without dedicated management systems, affecting a wide range of sports enthusiasts. The struggle to find suitable fields and the potential waste of resources in navigating various reservation platforms pose significant challenges. Traditional reservation solutions are often costly and cumbersome, exacerbating the difficulties faced by both field managers and sports aficionados.

Connecting field managers and sports enthusiasts in real-time, our app streamlines the reservation process and helps users select the best available options based on factors such as field availability, proximity, and cost. By integrating state-of-the-art technologies and algorithms, such as machine learning for optimized field matching, our solution enhances the efficiency and personalization of the reservation experience.

The core of our application harnesses powerful algorithms such as the "Greedy Algorithm for Field Matching" and the "Collaborative Filtering Recommendation System," details of which will be elaborated in the Algorithms section. The front end is developed using Flutter, ensuring dynamic user interfaces and seamless interactions, while the backend leverages Flask-SQLAlchemy for robust data handling. These technologies enable real-time tracking, efficient management of reservation conflicts, and sophisticated data processing. Additionally, the integration of machine learning algorithms allows the app to deliver personalized field recommendations, tailored according to individual user preferences and historical data.

In our app, users will quickly set up their accounts by entering basic information such as their name, phone number, and location. This streamlined approach allows our system to suggest sports fields that not only meet their specific needs but also enhance their overall booking experience. This feature simplifies the process of finding and reserving sports facilities, making it more efficient and user-friendly.

The technologies incorporated into the app will include secure communication protocols such as OTP for user authentication via SMS. Additionally, the app will leverage cloud-based services for scalable and reliable performance, ensuring that both managers and users can access the platform without any downtime or performance issues.

By implementing these advanced technologies and algorithms, our app will revolutionize the sports field reservation experience, making it more accessible, efficient, and enjoyable for all users and managers involved.

# 2. Background and Related Work

The landscape of sports field reservation services has witnessed the emergence of diverse approaches and technologies aimed at streamlining the reservation process. Various reservation platforms and applications have been developed, each offering its own set of features and functionalities. However, these solutions often face limitations in scope and encounter challenges in efficiently connecting users with available sports fields.

The variation of existing solutions and technologies has been a driving force behind the development of our application. By recognizing the fragmented nature of the current reservation landscape, we are committed to unifying and synchronizing such businesses through our comprehensive reservation management app. This approach will address the limitations of existing solutions and provide a more integrated and seamless experience for both field managers and sports enthusiasts.

## 2.1 The Most Popular Existing Solutions:

1. **TIMIFY:** offers a feature-rich reservation platform that caters to various service industries, including hospitality, beauty, and travel agencies. Its benefits include personalized customer experiences and increased customer loyalty. However, its limitations lie in the need for continuous system improvement based on feedback and performance metrics. TIMIFY is popular in Europe and North America and boasts over 45,000 users as of 2024. [More information can be found on their official website.](https://www.timify.com/en/)
2. **Omnify:** provides a sophisticated reservation system with omni-channel appointment booking software. It offers seamless integration into websites and social media pages, providing convenience and reliability for both customers and businesses. Omnify is widely used in the United States, Canada, and Australia, serving approximately 30,000 active users. [Visit Omnify’s official site for more details.](https://www.getomnify.com/)
3. **Zoho Bookings:** is recognized for its user-friendly interface and ease of use, making it a popular choice among small-scale businesses. However, it may not fully meet the needs of larger enterprises due to its basic in-house reservation systems, which lack some advanced functionalities essential for managing complex operations and high volumes of reservations. Zoho Bookings has a strong presence in India and the United States, with over 20,000 businesses using its services. [Learn more about Zoho Bookings here](https://www.zoho.com/bookings/).

As discussed in the "Background and Related Work" section, our application's comparative advantages are outlined in [table 1](#_vlayn9qa9ktq) below. This table contrasts our app with existing solutions, highlighting core features, key advantages, and unique benefits specifically designed for managers.

### Table 1: Comparative Analysis of Sports Field Reservation Systems

| **Aspect** | **TIMIFY** | **Omnify** | **Zoho Bookings** | **Your Application** |
| --- | --- | --- | --- | --- |
| **Core Features** | Customizable booking forms, and online payments. | Omni-channel booking integrates with social media. | Simple, user-friendly interface. | Real-time tracking, and personalized field recommendations. |
| **Key Advantage** | Enhances customer loyalty with personalized experiences. | Provides convenience through seamless integrations. | Ideal for small businesses with its ease of use. | Optimizes reservation process with machine learning. |
| **Integration** | Integrates with multiple business services. | Uses cloud-based solutions for scalability. | Utilizes basic in-house tech for ease of use. |  |
| **Technology Use** | Advanced scheduling algorithms. | Uses cloud-based solutions for scalability. | Utilizes basic in-house tech for ease of use. | Incorporates machine learning for field recommendations. |
| **Benefits for Managers** | Automated scheduling reduces managerial workload. | Dashboard for tracking bookings and customer interactions. | Dashboard for tracking bookings and customer interactions. | Enhanced decision-making through comprehensive reporting and real-time monitoring. |
| **Why Choose Our App** | Efficient but lacks customization for complex needs. | Versatile integration but can be overwhelming. | Great for simplicity but limited in advanced features. | Unparalleled personalization and efficiency in field reservation management. |

# 

# Data Structure

In the server part of our application, we keep a database that is designed around several key tables to facilitate the management of user profiles, payments, reservations, fields, and favorites fields.

**User Table:**

* Columns:
  + uid (UUID): Unique identifier for the user.
  + username (String): User's username.
  + password (String): User's password (hashed).
  + phonenum (String): User's phone number.
  + preferences\_id: UUID - Link to the Preferences table.
* Relations:
  + One-to-Many with Payments: A user can have multiple payment records.
  + One-to-Many with Favorites: A user can have multiple favorite fields.
  + One-to-Many with Reservations: A user can have multiple reservations.
  + One-to-One with Preferences: Each user has one set of preferences.
  + One-to-Many with Ratings: Each user can give multiple ratings.

**Payments Table:**

* Columns:
  + id: Unique identifier for the payment.
  + holder\_id: ID of the user who made the payment.
  + card\_number: Credit card number used for the payment.
  + digit\_code: Security code of the credit card.
  + month: Expiry month of the credit card.
  + year: Expiry year of the credit card.
* Relations:
  + Many-to-One with User: Many payments can be associated with one user.

**Reservations Table:**

* Columns:
  + id: Unique identifier for the reservation.
  + field\_id: ID of the field reserved.
  + date: Date of the reservation.
  + interval\_time: Time interval for the reservation.
  + status: Status of the reservation (e.g., confirmed, canceled).
  + due\_date: Due date for the reservation.
  + due\_time: Due time for the reservation.
  + user\_uuid: UUID of the user who made the reservation.
  + price: Price of the reservation.
  + field\_name: Name of the field reserved.
  + location: The location of the field is reserved.
  + image URL: The image URL of the field is reserved.
* Relations:
  + Many-to-One with User: Many reservations can be associated with one user.
  + Many-to-One with Field: Many reservations can be associated with one field.

**Fields Table:**

* Columns:
  + id: Unique identifier for the field.
  + name: Name of the field.
  + location: Location of the field.
  + latitude: Latitude coordinate of the field.
  + longitude: Longitude coordinate of the field.
  + sport\_type: Type of sport the field supports.
  + image URL: Image URL of the field.
  + facilities: Comma-separated list of facilities available at the field.
  + average\_rating: Average rating of the field.
  + ratings\_id: UUID - Link to the Rating table.
* Relations:
  + One-to-Many with Reservations: A field can have multiple reservations.
  + One-to-Many with Ratings: Each field can receive multiple ratings.

**Favorites Table:**

* Columns:
  + uid: Unique identifier for the favorite.
  + user\_id: ID of the user who favorited the field.
  + field\_id: ID of the favorite field.
* Relations:
  + Many-to-One with User: Many favorites can be associated with one user.
  + Many-to-One with Field: Many favorites can be associated with one field.

**Ratings Table:**

* Columns:
  + uid: Unique identifier for the rating.
  + User\_id: of the user who gave the rating.
  + field\_id: UUID ID of the field being rated.
  + rating: Numerical rating given by the user.
  + review: Text review of the field.
  + timestamp: Date and time when the rating was submitted.
* Relations:
  + Many-to-One with User: Many ratings can be associated with one user.
  + Many-to-One with Fields: Many ratings can be associated with one field.

**Preferences Table:**

* Columns:
  + id: UUID - Unique identifier for the preference.
  + user\_id: UUID - ID of the user.
  + sport\_type: Preferred sport type.
  + preferred\_location: Preferred location.
  + amenities: Preferred amenities (comma-separated list)
* Relations:
  + One-to-One with User: Each set of preferences is associated with one user.

These tables and their relationships form the backbone of the application's data model, allowing for the management of users, payments, reservations, fields, favorites, Preferences, and ratings.

# 

# 3 Algorithms

We identified several challenges inherent in the current landscape of sports field reservation systems, such as inefficiencies in field allocation, and difficulties in handling peak load reservations. Our project aims to showcase a range of algorithms designed specifically to address these challenges and optimize the sports field reservation process. By leveraging these algorithms, we aspire to offer a versatile and user-centric reservation management app that goes beyond the limitations of existing solutions.

## 3.1 Greedy Algorithm for Field Matching:

**Objective:** Efficiently match user preferences with available sports fields.

**How it works:** The algorithm selects the best available field for each reservation based on certain criteria, such as proximity, facilities, and user ratings. It makes locally optimal choices at each step, resulting in a solution that is acceptable for most cases.

**Formulas:**

1. Proximity Score:

Where distance (A, B) calculates the distance between locations A and B.

**distance (A, B) = 0**   
The distance between two locations is typically not exactly 0, but in some cases, such as when the locations are identical or very close together, the distance may effectively be 0.

In this case, we set the distance(A, B) to be 1.

1. Facilities Score:

Where n is the number of facilities, is the weight assigned to each

facility, and Facility\_match (A, B) is a function that determines the match between the

reservation and field facilities.

1. Ratings Score:

Where average\_rating(R) calculates the average rating from the set of ratings R and exp(-x) is the exponential function.

**Pseudocode:**

function selectBestField(reservation, availableFields):  
 Initialize bestField to None  
 Initialize bestScore to negative infinity  
  
 for each field in availableFields:  
 Calculate totalScore as the sum of:  
 - Proximity score between reservation location and field location  
 - Facilities score based on matching reservation and field facilities  
 - Ratings score using the average user ratings of the field  
  
 if totalScore is greater than bestScore:  
 Update bestField to current field  
 Update bestScore to totalScore  
  
 return bestField

## 

## 3.2 Collaborative Filtering Recommendation System:

**Objective:** Generate personalized recommendations for sports fields based on user preferences and historical booking data using collaborative filtering techniques.

**How it works:**

1. Data Collection:

* Collect comprehensive historical booking data, including user preferences (e.g., preferred sports, location, amenities), booking dates, and booked sports fields.
* Store this data in a structured format in the application's database for further analysis.

1. User-Item Matrix Creation:

* Construct a user-item matrix where rows represent users, columns represent sports fields, and each cell denotes a user's booking history (e.g., 1 if the user has booked the field, 0 otherwise).
* Incorporate additional features such as user ratings, field amenities, and booking frequency into the user-item matrix to enrich the recommendation process.

1. Similarity Calculation:

* Compute the similarity between users or sports fields based on the user-item matrix using various similarity measures such as cosine similarity, and Pearson correlation coefficient.
* Adjust similarity calculations to handle sparse data and mitigate the impact of outliers.

1. Recommendation Generation:

* Generate personalized recommendations for each user or sports field by aggregating the preferences of all users or fields.
* Implement strategies to address the cold start problem for new users or fields by leveraging item-based collaborative filtering, and content-based filtering.
* Utilize techniques such as weighted averaging or matrix factorization to enhance recommendation accuracy and diversity.
* Incorporate post-filtering mechanisms to remove irrelevant or unpopular recommendations before presenting them to users.

Formulas:

1. Cosine Similarity:

Where u and v are vectors representing the preferences of two users or sports fields.

1. Pearson Correlation Coefficient:

Where and are the means of the preferences of users u and v respectively.

**Pseudocode:**

| function generateRecommendations(user, userItemMatrix):  Initialize recommendations as an empty dictionary  Calculate similarities between user and all other users using cosine similarity    for each otherUser in userItemMatrix:  if otherUser is not the current user:  Compute similarity score between user and otherUser    for each field in otherUser's preferences:  if field is not already booked by user:  Accumulate weighted preference for field based on similarity score   Sort and return fields by their accumulated preference scores |
| --- |

## 

## 

## 3.3 Automated Conflict Resolution:

**Objective:** Automatically resolve conflicts in reservation requests to optimize field utilization and prevent double bookings.

**How it works:**

1. Availability Check:

* Upon receiving a reservation request, the system checks the availability of the requested field for the specified date and time.
* If the requested time slot is available, the reservation is confirmed, and the system updates the field's booking schedule accordingly.
* If the requested time slot is already booked or conflicts with an existing reservation, the system initiates an automated conflict resolution process.

1. Conflict Resolution Process:

* The conflict resolution algorithm prioritizes fairness and efficiency, considering factors such as reservation priority, proximity, and user preferences.
* Based on predefined rules and parameters, the algorithm selects the most suitable resolution strategy.
* Resolution strategies may include rescheduling conflicting reservations, suggesting alternative time slots, or reallocating resources to accommodate multiple requests simultaneously.

1. Notification and Update:

* Once a resolution is reached, the system notifies the involved parties (both users and managers) of the outcome and updates the reservation status accordingly.
* Continuous monitoring and optimization ensure that conflicts are swiftly addressed, maximizing field utilization while minimizing disruptions to users' booking experience.

Formulas and Parameters:

1. Reservation Priority Formula:

t: Reservation time | : Current time | : Rate of priority decay

1. Proximity Score Formula:

: Proximity weight factor | Distance: Distance between user location and field

1. User Preference Score Formula:

: Weight factor for user ratings | OtherPreferences: Other relevant user preferences

Predefined Rules and Parameters:

* Rules for Rescheduling: Lower-priority reservations may be rescheduled to accommodate higher-priority requests.
* Parameters for Reallocation: Priority may be given to users with higher ratings or those who have previously faced booking disruptions.

Maximizing Field Utilization:

The algorithm balances maximizing field utilization with minimizing disruptions to users' booking experience by optimizing time slot allocation, adjusting reservation durations, and offering incentives for off-peak bookings

**Pseudocode:**

| function resolveConflict(reservationRequest):  if field is available:  confirmReservation()  else:  resolveUsingAlternatives(reservationRequest)  function resolveUsingAlternatives(request):  alternative = findBestAlternative(request)  applyAlternativeResolution(alternative)  function findBestAlternative(request):  # Determine the best alternative based on availability, user preferences, and proximity  return bestAvailableOption  function applyAlternativeResolution(alternative):  # Apply the chosen alternative resolution strategy  updateSystem(alternative)  notifyStakeholders() |
| --- |

# 

# 4. Expected Achievements

## 4.1 Outcomes:

In our Sports Facility Management System project, we anticipate achieving significant advancements and delivering valuable results for both users and field managers. These expected achievements align with the diverse goals of our project and aim to enhance the overall experience of sports facility reservations.

#### 4.1.1 Optimized Reservation Matching Algorithm:

Objective: Implement an intelligent algorithm to efficiently match user preferences with available sports fields.

How it works: This achievement utilizes the "Greedy Algorithm for Field Matching" to make locally optimal choices based on user preferences, field availability, and location. This algorithm is crucial for ensuring that the system recommends the most suitable fields, thus enhancing user satisfaction and operational efficiency. For a detailed explanation of how this algorithm operates, please refer to Section 3.1 on the Greedy Algorithm for Field Matching.

#### 4.1.2 Real-time Conflict Resolution and Notification System:

Objective: Implement a system to automatically resolve booking conflicts and notify users and managers in real-time, enhancing the responsiveness and reliability of the reservation process. How it works: This feature is driven by the Automated Conflict Resolution algorithm. It continuously monitors the booking schedule to detect and resolve conflicts as they arise. By ensuring conflicts are managed efficiently, the system maintains accurate reservation statuses and immediately informs all parties of any necessary updates. This capability is crucial during peak reservation periods and helps optimize field utilization while ensuring a smooth experience for both field managers and users.

#### 4.1.3 Dynamic User Experience through Collaborative Filtering:

Objective: Provide a dynamic and user-centric interface that adapts to individual preferences and enhances user engagement through personalized recommendations.

How it works: Utilizing the Collaborative Filtering Recommendation System, this feature personalizes the user interface based on individual user behaviors and preferences. It analyzes past interactions and booking patterns to tailor the available options, presenting users with field choices that best match their preferences. This not only improves user satisfaction but also encourages more frequent use of the app by making the reservation process more intuitive and aligned with user needs.

## 

## 4.2 Unique Features:

#### 4.2.1 Intelligent Reservation Matching Algorithm:

Our project will introduce a sophisticated algorithm to optimize the matching of user preferences with available sports fields. This will ensure efficient and personalized reservation assignments

#### 4.2.2 Automated Real-Time Reservation Monitoring and Notifications:

the Automated Conflict Resolution algorithm, which keeps all parties continuously informed about the status of their reservations, ensuring a smooth and transparent process.

#### 4.2.3 Dynamic User Experience through Collaborative Filtering:

The interface of our reservation system adapts to user behaviors and preferences using the Collaborative Filtering Recommendation System. This personalization enhances user engagement by providing tailored field recommendations and remembering user history

## 

## 4.3 Criteria for Success:

Our project's success will be evaluated based on key criteria that encompass system performance, user satisfaction, efficiency, security, and privacy:

#### 4.3.1 System performance:

The server part should be able to work with the database correctly, including adding, editing, and reading the data. The GUI of the user’s and manager’s sides of the application should work properly. Options from the GUI should be correctly connected to the server.

#### 4.3.2 Efficiency Metrics:

We will measure the efficiency of the reservation process, including time and cost savings for users and field managers compared to traditional methods.

#### 4.3.3 Security and Privacy:

Rigorous measures will be implemented to ensure the security and privacy of user data, addressing non-trivial requirements and building trust among users.

**Example:**

Encryption: All sensitive data will be encrypted both in transit and at rest to prevent unauthorized access.

Access Control: Strict access controls will be enforced, ensuring that only authorized personnel have access to sensitive information.

Regular Security Audits: Regular audits will be conducted to identify and address any potential security vulnerabilities.

#### 4.3.4 User-Friendly Interface:

The success criteria include the development of a user-friendly interface that caters to the needs of both sports enthusiasts and field managers, fostering a positive user experience.

# 

# 5 The Process

## 5.1 Research – Field Reservation Management

As we continue to develop our Field Reservation Management application, our research has centered around key questions essential to its refinement:

* How can we efficiently assign a customer the closest football field or tennis court during the reservation process?
* What factors are pivotal in persuading sports facilities to adopt our reservation system?
* What considerations significantly influence a customer's decision to book a field through our platform?

To address these inquiries, we've engaged in thorough research,

Consultation with Field Managers and Users:

* We conducted interviews and surveys with field managers to understand their challenges, preferences, and expectations regarding reservation systems. Additionally, we engaged with users of existing reservation platforms to gather feedback on their experiences and identify areas for improvement.

Review of Existing Systems and Feedback:

* We thoroughly examined feedback from users of existing reservation systems, including online reviews, forums, and social media platforms. This analysis helped us understand common pain points, likes, and dislikes among users. We also studied the features, strengths, and weaknesses of competing reservation systems to identify opportunities for differentiation and improvement.

Exploration of Academic and Industry Resources:

* Our research extended to industry reports, and case studies related to sports facility management and reservation systems. By delving into these resources, we gained insights into emerging trends, best practices, and innovative solutions in the field.

Following this research phase, we convened to distill our findings and identify crucial aspects to integrate into the application's development.

Among the insights garnered, we've recognized the importance of incorporating a post-reservation evaluation questionnaire within the application. This questionnaire will seek

feedback on:

* Overall satisfaction with the field reservation app.
* User-friendliness of the application.
* Most appreciated features or functionalities within the app.
* Suggestions for additional features or enhancements in future updates.
* Reliability and stability of the app during usage.
* Ease of tracking field reservations through the app.
* Perception of the app's speed and efficiency in facilitating reservations.

While the application is still under development, we aim to leverage the insights gleaned from the questionnaire to iterate and enhance the effectiveness of our field reservation management application before its release.

### 5.1.1 Constraints and Challenges – Field Reservation Management:

In our system, selecting the most appropriate field for a reservation can pose challenges due to various factors that need to be taken into account. Some of these factors may include:

#### 5.1.1.1 Proximity:

The distance between the customer and the available fields can affect the convenience of the reservation and the time required to access the field.

#### 5.1.1.2 Availability:

The availability of the fields (such as whether they are currently booked or have the capacity for additional reservations) can influence their suitability for a particular booking.

To tackle these challenges, our system comprehensively considers these factors when recommending the most suitable field for a reservation. This necessitates access to detailed information about the fields (such as their location, amenities, and current availability) and the capability to weigh these factors against each other to make informed decisions.

## 5.2 Methodology and Development Process

Following a thorough research phase, we opted for the Agile approach due to its iterative nature, enabling ongoing planning and continuous learning. Our methodology involves stages of requirement gathering, design and prototyping, development and testing, and deployment with a focus on user feedback. This structured approach ensures the creation of a sports field reservation app that exceeds user expectations, offering a seamless experience for both sports enthusiasts and field managers.

Our work process for the sports fields reservation app is divided into three main parts:

### 5.2.1 Planning and Preparation:

In this phase, we held meetings to create a detailed requirements document for the system. We developed diagrams to visualize the desired outcome and identified the tools and technologies required for development. Specifically, we decided to utilize the Framework of Flutter, The code is written in Dart, for front-end development, the Framework of Flask-sqlalchemy for back-end development code is written in Python, and PostGress (Database) for managing the data of our interactive Application.

### 5.2.2 Implementation:

This phase involves using inner service to connect to an SMS API and send confirmation messages to users. Additionally, we defined and completed initial tasks for the project, such as creating the main screens for users and establishing the basic infrastructure of the application.

### 5.2.3 Iterative Development:

As we progress and learn, we will continue to develop and enhance the system in stages until we reach the final product. This iterative approach entails ongoing planning and adjustments based on lessons learned, ensuring that the application evolves to meet user needs and expectations effectively.

# 

# 6 The Product:

In software development, functional requirements (FR) define what a system is supposed to do, while non-functional requirements (NFR) specify how the system should perform. Both types of requirements are essential for ensuring that a software system meets user needs and quality standards.

As outlined below in [Table 2](#_ydzj3xrcphp8), our system's requirements are categorized into functional and non-functional to highlight their roles in achieving system effectiveness and efficiency.

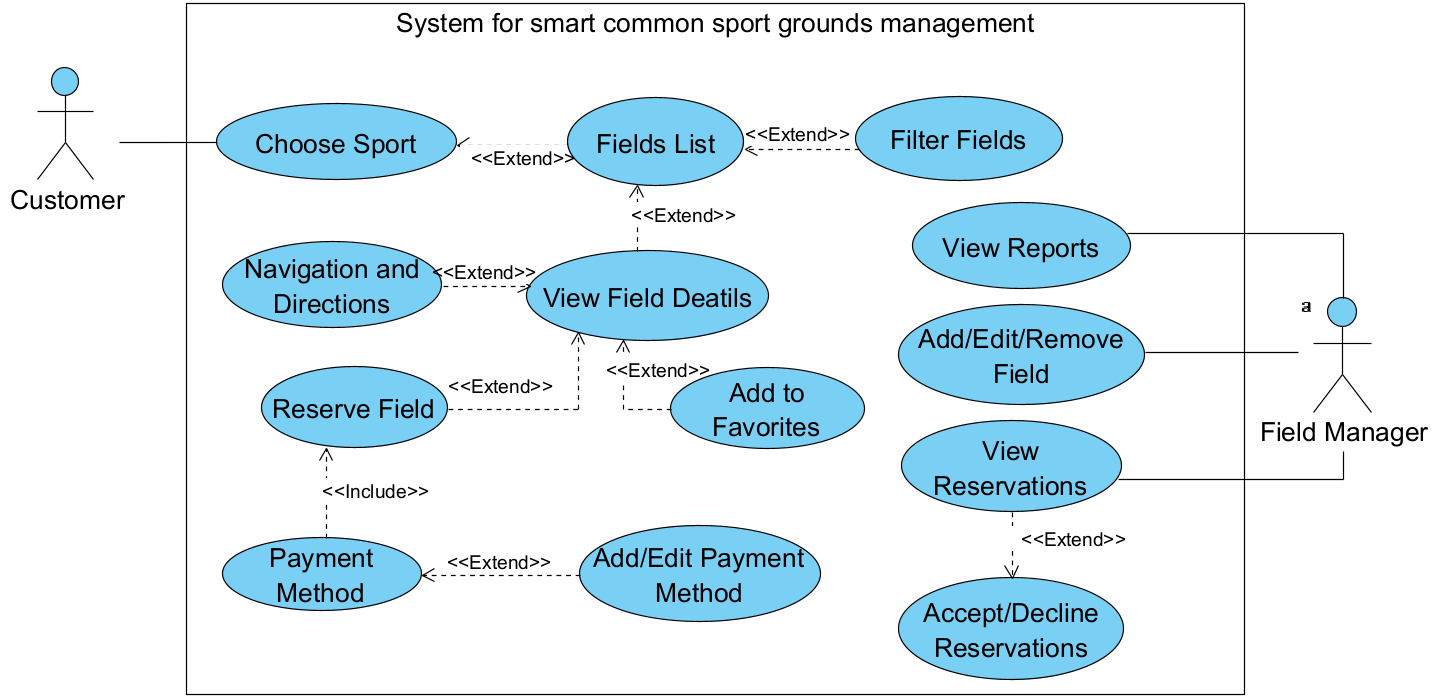
## 6.1 Requirements:

#### Table 2: System Requirements for the Sports Facility Management System

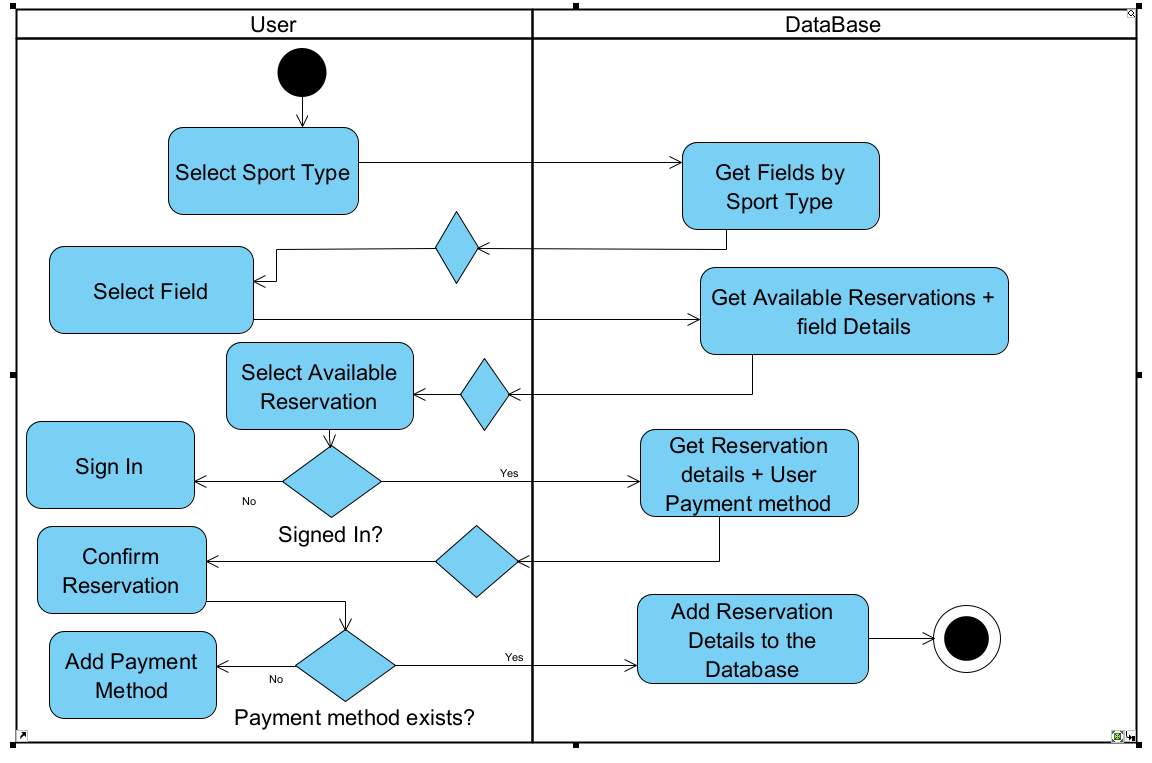
| **Req.**  **number** | **Requirements description** | **Reqtype (FR or NFR)** |
| --- | --- | --- |
| **1** | The system allows users to identify by username and password | FR |
| **2** | The username is unique for each user | NFR |
| **3** | The system allows identifying the user type immediately after logging in | FR |
| **4** | The user types are, Customers, Field Managers | NFR |
| **5** | The system allows for the Field Managers to add new Fields | FR |
| **6** | Each Stadium has its unique attributes{location, available dates, pricing} | NFR |
| **7** | The system allows Field Managers to view the reservation of their fields | FR |
| **8** | The system allows the Field Managers to adjust the field attributes | FR |
| **10** | The reservation can be canceled only before a specific time of the reservation | FR |
| **11** | The system allows customers to view Pending Reservations | FR |
| **12** | The System allows for customers to view Accepted Reservations | FR |
| **13** | The system allows each customer to pay | FR |
| **15** | The system allows users to view Stadiums | FR |
| **16** | The system will allow customers to add a payment method | FR |
| **17** | The system allows for Customers to remove payment method | FR |
| **18** | The customer can filter for specific location/time/date of the field | FR |
| **19** | The system allows for Field Managers to remove Fields | FR |
| **20** | The system allows Field Managers to remove/block users | FR |
| **21** | The status of each reservation can be: {Pending, Accepted, Canceled} | NFR |

## 6.2 UML Diagrams

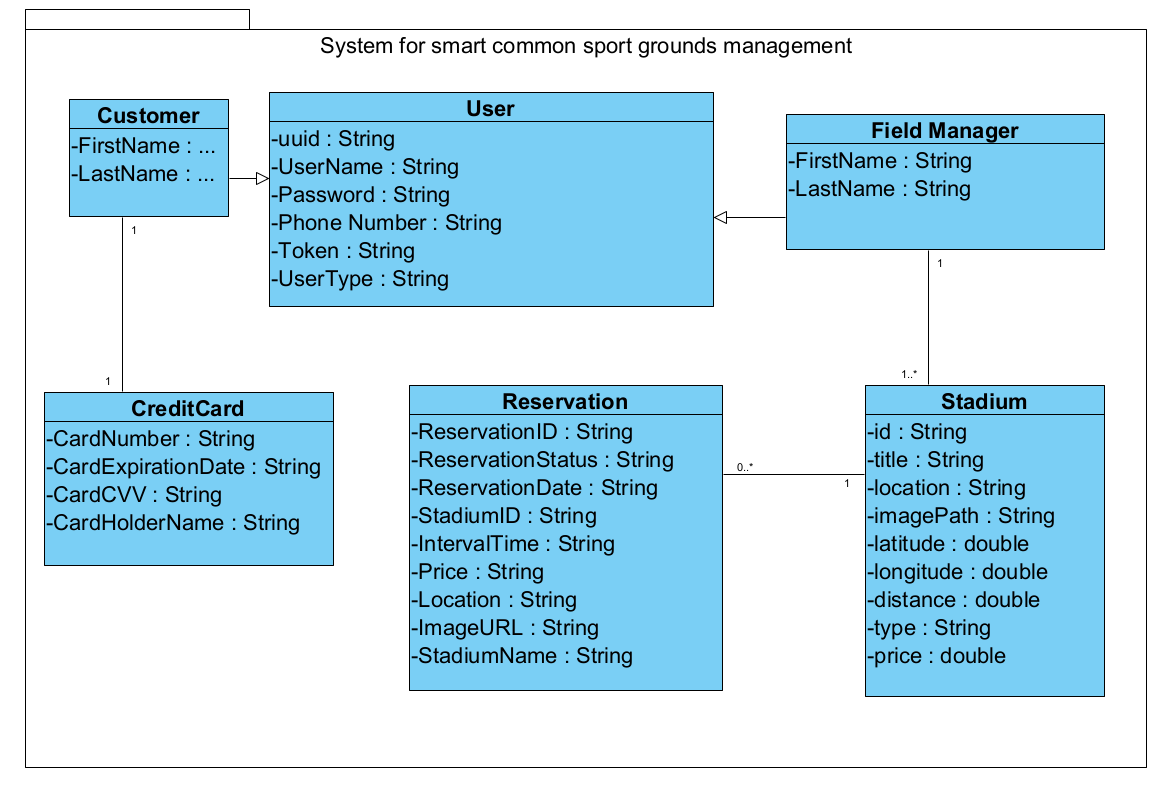
### 6.2.1 Use Case Diagram

****

### 6.2.2 Activity Diagram - Field Reservation Flow

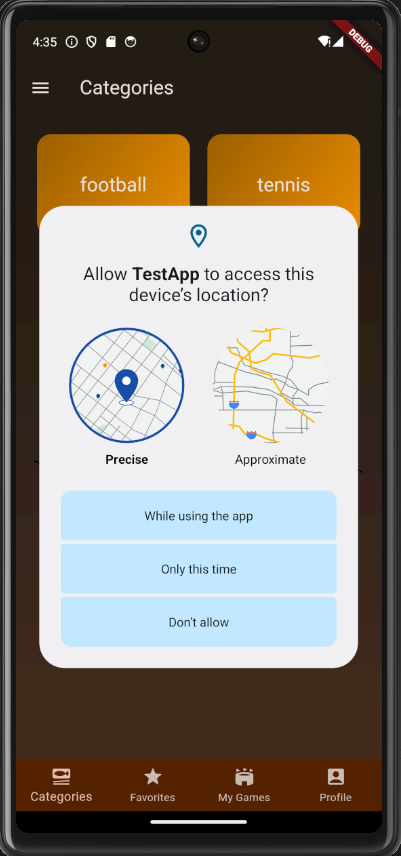
****

### 6.2.3 Class Diagram - Entities



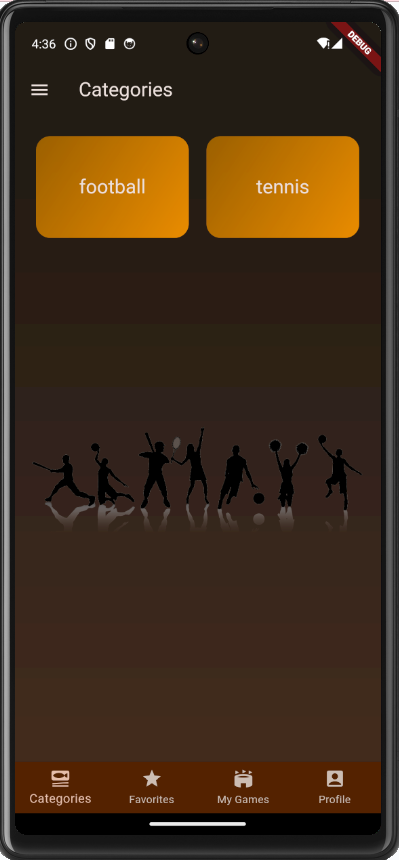
## 6.3 Draft of GUI We will explore the graphical user interface (GUI) of our application, offering a detailed analysis of each screen's purpose and functionality.

First Screen:



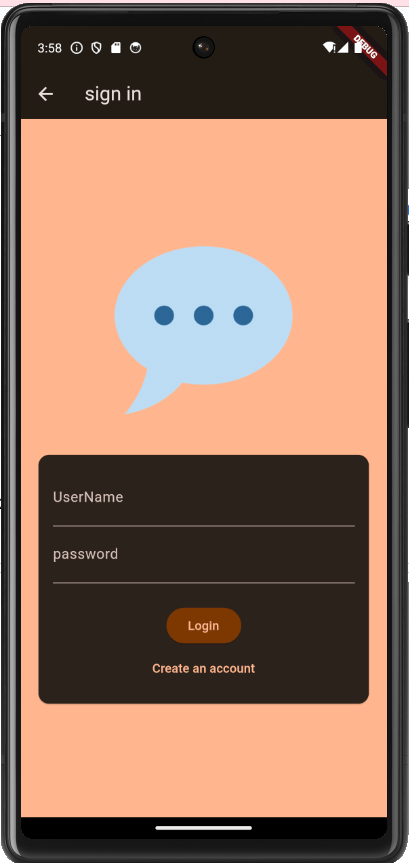
**Fig1.** This is the first screen that the user will see, he will decide whether to allow access to his location or not.

Home Screen:



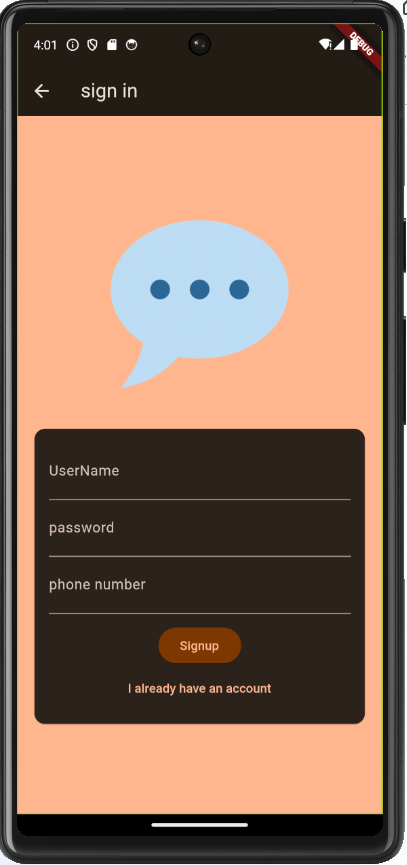
**Fig. 2.**  That is the first screen that the customer will be able to interact with, once he’s logged in he can navigate to several tabs, like “Favorites”, “profile”, and “My games” and also from the Hamburger bar on the top left he can access the Login screen, his Credit cards etc…

Login:



**Fig. 3.** This is the sign-in page, each user can join with his private username and password, and the system will recognize the type of the user (Customer/Field Manager) and open the relevant page. For the new users who want to join our system, can sign up by clicking on the Create an account Text.

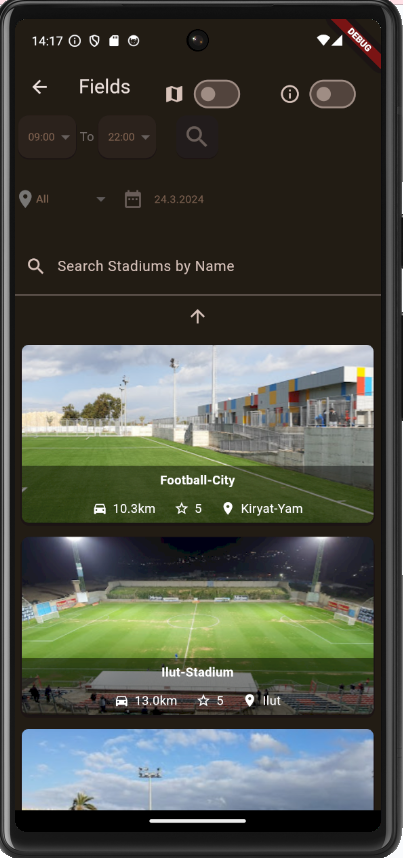
Sign Up



**Fig. 4.** Once clicking the “Create an account” Text, We will see those fields required for signing up, if the user wants to go back to the sign-in, he will click “ I already have an account”

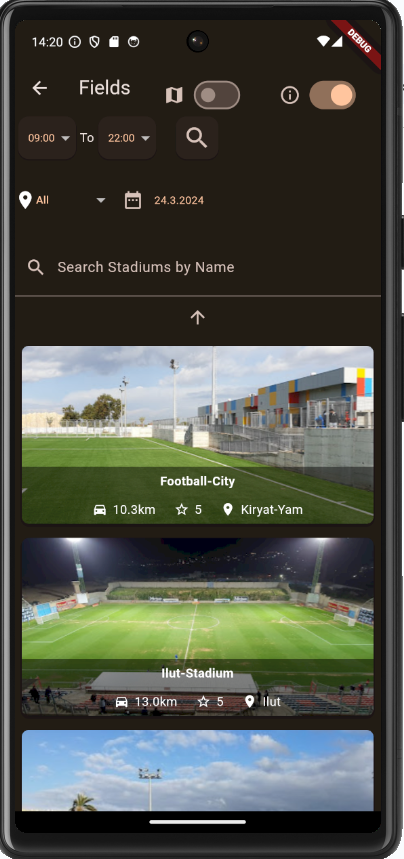
**Reserving a Field Process:**

Filtering for Stadium:



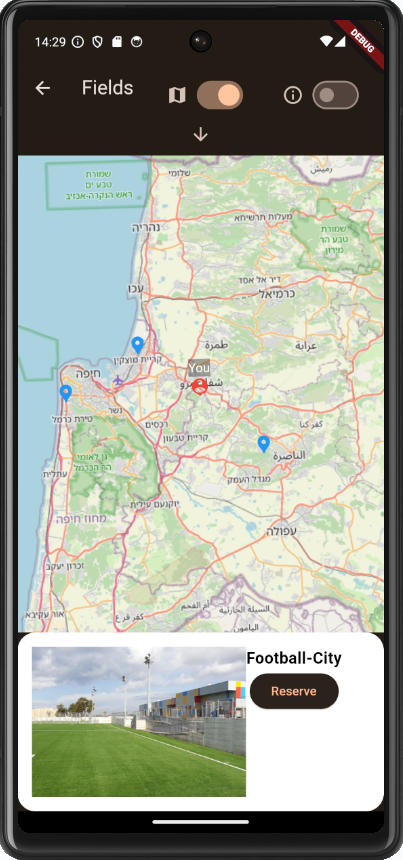
**Fig. 5.**  After selecting a category from Fig. 2, you'll be taken to a screen where you can search for fields based on your preferences. You can filter by proximity, date, time, location, or the field's name.

The filters can be turned on/off:



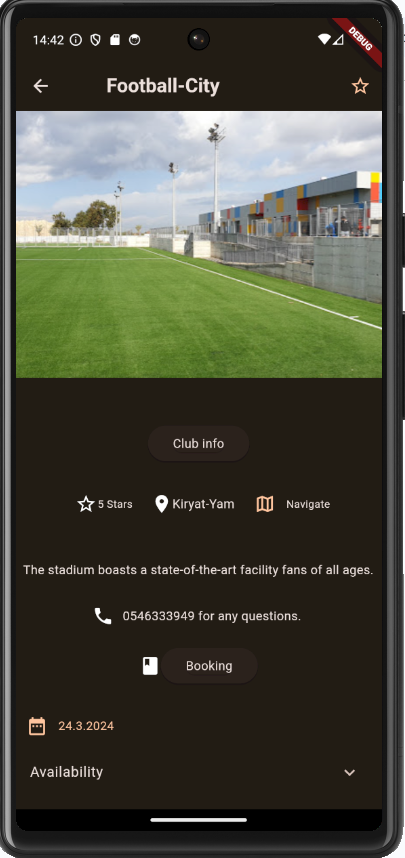
**Fig. 6.** Here the filters are turned on, and once the customer chooses his desired filters, he can click the search Icon to get results according to his filters.

Map Can be turned on as well



**Fig. 7.** The customer can also search for stadiums throughout the MAP, the red ICON is the location of the user, and the blue Icons are the fields.   
Once one of the blue Icons is clicked the user can see a small overview of the fields and the option to reserve it.

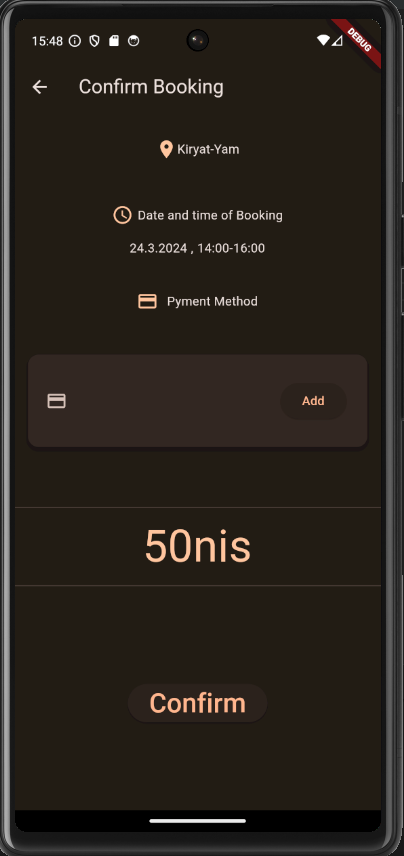
Stadiums Details Screen



**Fig. 8.** Here the customer can see a few details about the fields, he can click the “Navigate” button in order to use Google Maps to navigate to the field, also to see the field availability, he can click on the drop-down list “Availability” in order to see the available times he can reserve

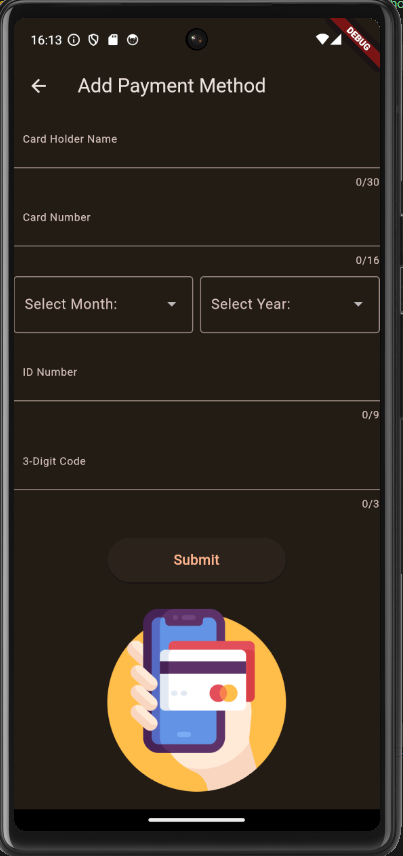
The customer can add the field to the favorites.

Confirm Reservation:



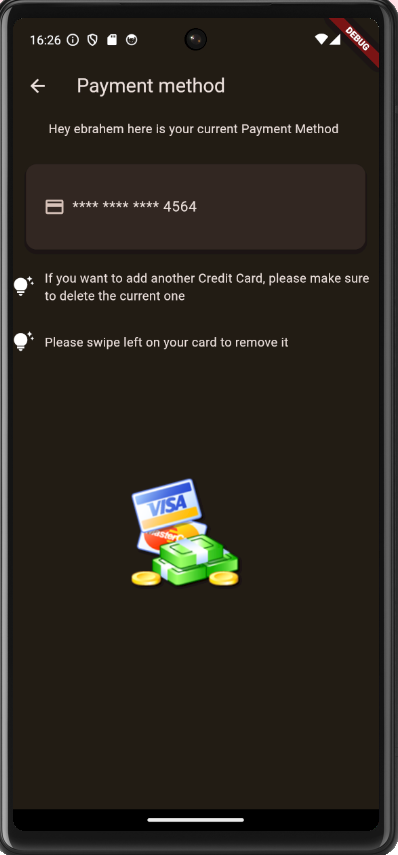
**Fig. 9.** After the user has clicked on one of the available times listed in Fig8 he will be navigated to this screen, which will show all the details of the reservation, the system will not allow the user to confirm the reservation unless he adds a Payment-Method by clicking on the Add button or adding it from the home screen in Fig2 (from the Hamburger Bar)

Adding payment method:



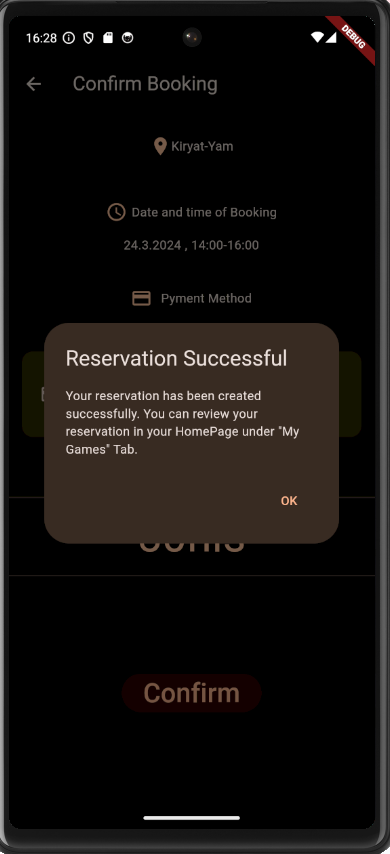
**Fig. 10.** This Screen will allow the user to add a new payment method, the user can add one payment method at a time.

Payment Method:



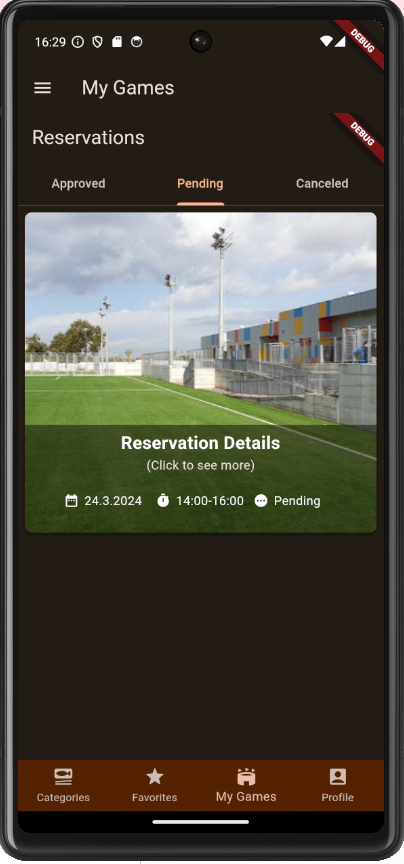
**Fig. 11.** The customer can see his Payment Method on this Screen.

Confirming Reservation:



**Fig. 12.** After Adding the payment method, and clicking Confirm, there will be a confirmation text.

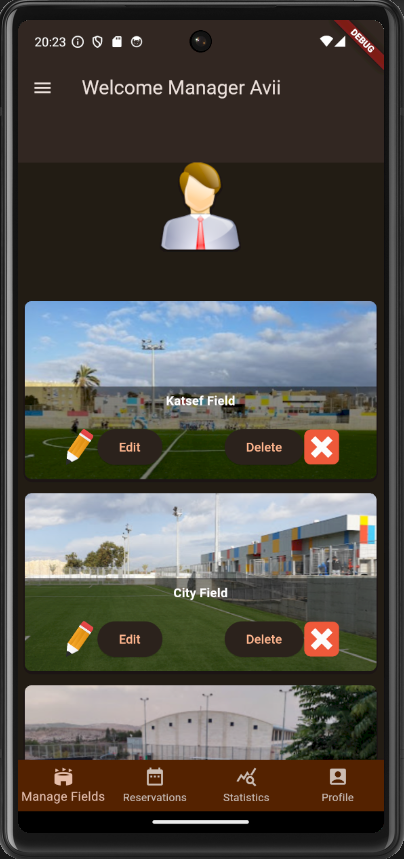
My Games Tab:



**Fig. 13.** Here the customer can view his Pending/Approved/Canceled Reservations, he can click on the reservation to see its details.

The manager decides whether to accept/deny the reservation.

Manager HomePage:



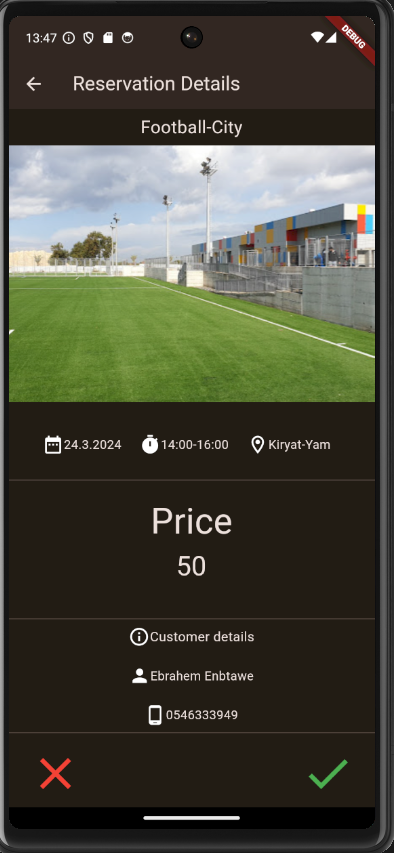
**Fig. 14.** Once a user logs in and he’s identified as a Field Manager he will be presented with the following screen (Fig. 14) that the Manager Fields screen, as he will be able to Edit/Delete his fields to his liking.

Manager Reservations:



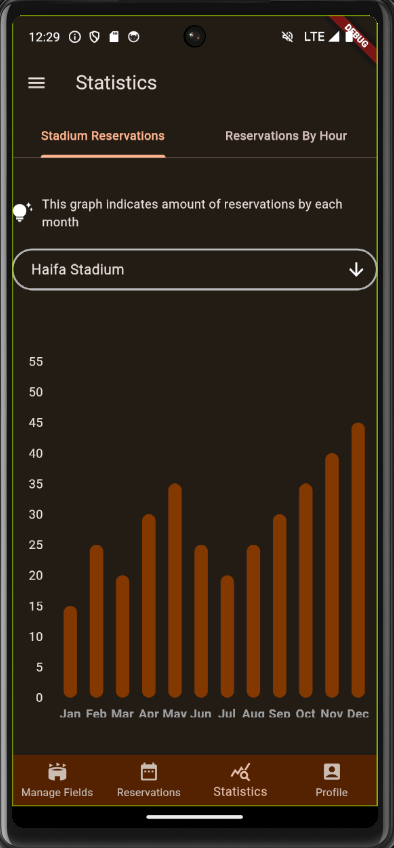
**Fig. 15.** Similar to the reservations of a certain customer (Fig. 13) The manager has the same view, only what's different is he can see all the reservations of his fields only.

Accept/Reject Reservations:



**Fig. 16.** After clicking one of the reservations from (Fig. 15), The manager will be presented with all of the reservation’s details including the details of the customer that has reserved the field, and the manager has the ability to accept/reject the reservation.

Statistics:



**Fig. 17.** Here the managers can view statistics, for example in Fig. 17 we can see the number of reservations each month for the stadium “Haifa Stadium”.

# 7. Verification and Evaluation

## 7.1 Table 3:Unit Tests

| Test No# | Test Subject | Expected Result | Details |
| --- | --- | --- | --- |
| 1 | User Registration and Login | Successful registration and login with valid credentials; error messages for invalid attempts. | Tests both the registration process and login functionality to ensure user data is correctly handled, including validation for incorrect credentials. |
| 2 | Field Creation and Validation | Successful field creation with valid details; error messages for incomplete or invalid field data. | Verifies that new sports fields can be added correctly and that the system checks for data completeness and validity. |
| 3 | Reservation Booking and Cancellation | Successful booking for available fields and correct handling of double bookings and cancellations. | Ensures that reservations are managed correctly, including tests for conflicts and cancellation processes for both valid and non-existent bookings. |
| 4 | User Profile Updates | Successful updates with valid data; error handling for invalid data updates. | Checks the system’s ability to correctly update and validate user profile information. |
| 5 | Database CRUD Operations | Add/Edit/Remove/Search operations function correctly for all database entities. | Comprehensive testing of database operations to ensure all data interactions are performed correctly and efficiently. |
| 6 | GUI Functionality | All GUI elements function correctly and respond appropriately to user interactions. | Evaluates the usability and responsiveness of all graphical user interface elements under various scenarios. |
| 7 | Connection Stability | The system maintains stable connectivity and handles network interruptions gracefully. | Tests the system's robustness in maintaining stable connections and its capability to recover from network issues. |

# 

# 8. References

[1] Greedy Algorithm for Field Matching: Kleinberg, J., & Tardos, E. (2006). Algorithm Design. Pearson Education. H. D. Lasswell's work in "The Theory of Political Propaganda" (1927)

[2] Collaborative Filtering Recommendation System: Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. Computer, 42(8), 30-37.

[3] Automated Conflict Resolution: Z. Michalewicz and D. B. Fogel, “How to Solve It: Modern Heuristics,” 2nd ed., Springer, Berlin, Germany, 2004.

[4] Machine Learning Algorithms: James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning. Springer.

[5] User Interface Design and Development with Flutter: The Flutter Community. (2022). Flutter Documentation. Retrieved from <https://flutter.dev/docs>.

Official documentation and tutorials from the Flutter website were referenced for front-end development,

[6] Flask-SQLAlchemy Usage: Grinberg, M. (2018). Flask Web Development: Developing Web Applications with Python. O'Reilly Media.

[7] Real-time Data Handling and Processing: Fisher, D., & Dorner, R. (2019). Stream Processing with Apache Flink. O'Reilly Media.

[8] AI Tools and Techniques:

OpenAI's ChatGPT, Grammarly, Perplexity AI, BART, Pegasus AI