SponsorIN

Brandon Tiong
Emanuel Macias
Ege Keser

https://github.com/twosize/SponsorIN
https://youtu.be/hkl0gRBrgX0?feature=shared

Table of Contents

1. Project Definition

Background

Before 2021, the college athletes around the United States were not able to receive any financial support from their supporters or find a sponsorship because of the NIL (Name, Image and Likeness) restrictions of NCAA (National Collegiate Athletic Association). Due to the recent laws passed in many states, college athletes had the chance to get paid by their supporters as well as started to find sponsorships to build their brands using their own name. However, according to our research, there is no platform that would help college athletes to promote themselves and find a possible sponsorship connection. It's easier for companies to find talented athletes and sponsor them due to their giant budgets. Unfortunately, that is not the case for athletes.

What is SponsorIN and how will it help?

SponsorIN will come into play when college athletes want to promote themselves and find sponsorships easily. SponsorIN is a web application to help college athletes and companies to find each other. SponsorIN will provide college athletes and companies an environment to build network, stay connected, and possibly, build a business relationship. College athletes will be able to promote themselves and look out for sponsorship, while companies are looking for athletes with great potential and help them grow their own brand under their sponsorship.

How will it be developed?

We will start developing SponsorIN by developing basic functions that most of the web applications would normally have. These functions include user registration, sign in – sign out and basic search. Then, our team will focus on developing the user specific functions such as sending-accepting-negotiating offers, athlete pool viewing-filtering-sorting, user verification and user editing-deletion. Once we have all the functions up and running, we will connect those functions with associated front-end and database sections so that we will have a complete web application that aims to accomplish our goals.

2. Project Requirements

2.1 Functional Requirements

- **User Types**: Three different types of users—Admin, Company, and Athlete—with distinct permissions and functionalities.

Common Features for All Users

- **Login & Authentication**: Secure login process for Admins, Companies, and Athletes.
- **Profile Creation & Editing**: Customizable profiles to add and edit pertinent details.
- **Messaging System**: In-app secure messaging for communication between all user types.
- **User Search**: Ability to search for other users based on relevant criteria.

Admin-Specific Features

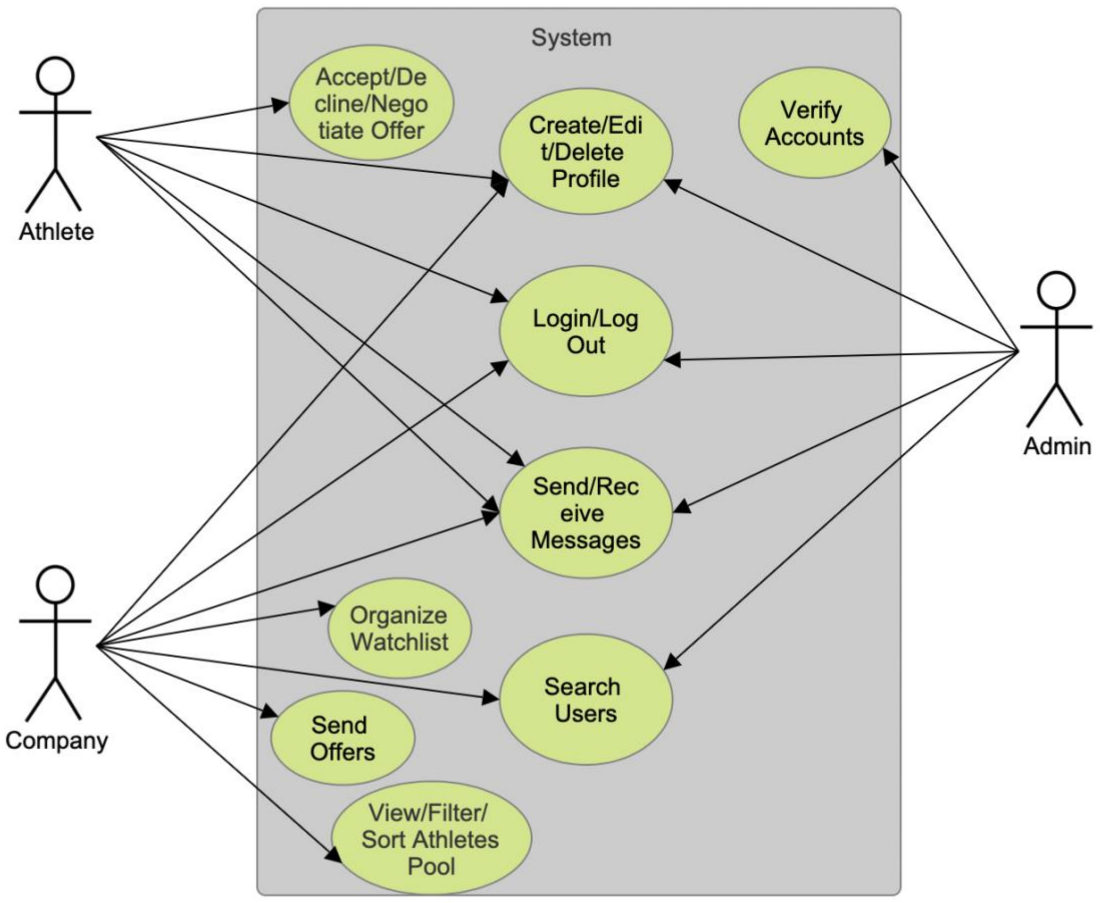
- **Profile Verification**: Exclusive rights to verify the profiles of Companies and Athletes.
- **User Editing & Deletion**: Capability to edit or delete any user profiles.

Company-Specific Features

- **Send Offers**: The ability to send sponsorship offers to Athletes.
- **Organized Watchlist**: A feature to save and organize a list of Athletes they are interested in.
- **Filtered Athlete Pool**: Advanced search and filters specific to finding suitable Athletes.

Athlete-Specific Features

- **Accept/Decline/Counter Offers**: The ability to interact with offers from Companies by accepting, declining, or sending a counteroffer.



2.2 Usability Requirements

- **User Interface**
    - **Responsive Design**: The web app should be accessible and usable on various devices including smartphones, tablets, and desktops.
    - **Intuitive Navigation**: Menus and sub-menus should be easy to understand and navigate.
    - **Information Architecture**: Efficient categorization of information for easy retrieval and action.
- **Performance**
    - **Load Time**: The web application should load within 3 seconds to provide a seamless user experience.
    - **Scalability**: The system should be able to handle a growing number of users and data.
    - **Error Handling**: Graceful error messages and solutions should be provided for common issues.

2.3 System Requirements

- **Hardware**
- Since the application is web-based, the hardware requirements are minimal; users need a device with internet connectivity. On the server side, robust cloud servers may be used for better scalability and performance.

- **Software**
  - **Back-End**: Python Flask for server-side logic.
  - **Front-End**: HTML, CSS, JavaScript and Bootstrap for UI/UX.

- **Database**
  - **Database Engine**: PostgreSQL
  - **Data Backup**: Regular automated backups.
  - **Data Integrity**: ACID compliance to ensure data integrity.

- ***Security Requirements***
  - **Data Encryption**: Data should be encrypted during transit and at rest.
    - This will be done by HTTPS, and SSL certificate
  - **Authorization**: Role-based access control for different types of users.
  - **Secure APIs**: All APIs should be secure and only accessible through authorized methods.

By addressing these requirements, the project will provide an efficient and secure experience for both athletes and companies.

3. Project Specification

- **Area of Focus**
  The primary focus of the application is to create a web application that connects college athletes and sponsoring companies, in compliance with the newly updated NCAA regulations. Our target audience includes college athletes seeking sponsorships and companies interested in sponsoring athletes to promote their products or service.
- **Libraries, Frameworks and Development Environment**
  The web application will primarily be developed using Python Flask for the backend, HTML, CSS, and JavaScript for the frontend, and PostgreSQL for the relational database. The framework will include React to enhance the frontend.
- **Platform**
  The primary platform will be web-based for desktop browsers.
- **Genre**
  The application SponsorIN falls under the Business Networking and Sports Management genre.
- **Responsibilities**
  1- Brandon Tiong: Full-Stack Developer, mainly backend.
  2- Emanuel Macias: Full-Stack Developer, mainly frontend.
  3- Ege Keser: Full-Stack Developer, mainly data, documentation and support for backend and frontend.

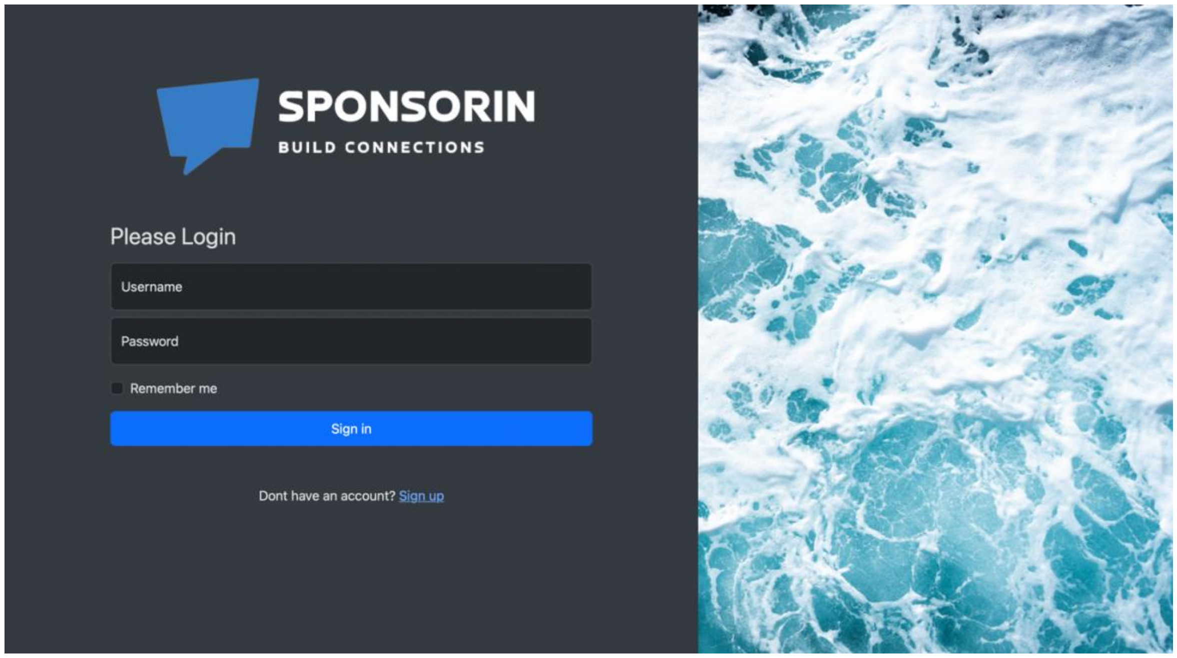
4. System – Design and Analysis Perspective

4.1 Subsystem Functional - Architectural Design

4.1.1 User Management Subsystem:

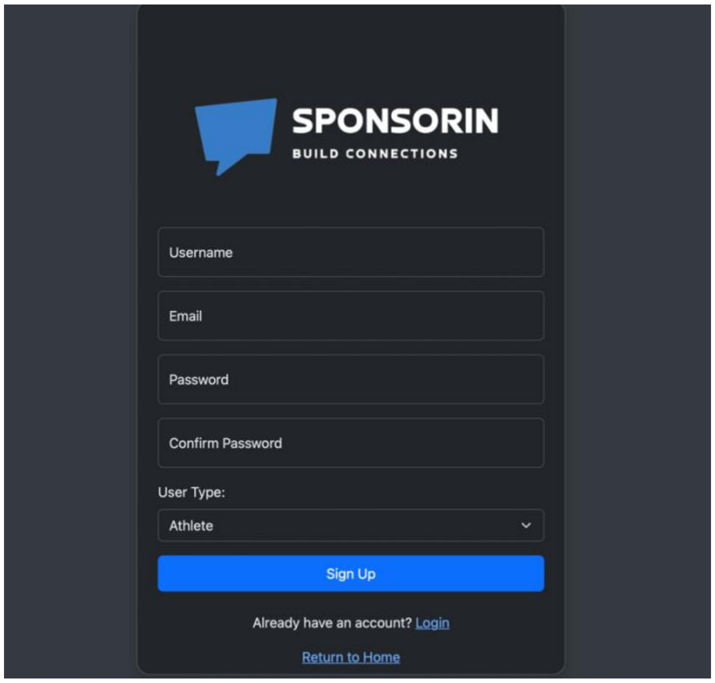
- **Description**: Handles user registration, authentication, profile management, and Users (Admin, Company, Athlete).
- **Components**:
- Registration & Sign In/Out.
- User-Based Access Control.
- Profile Creation & Editing.

Sign-in:



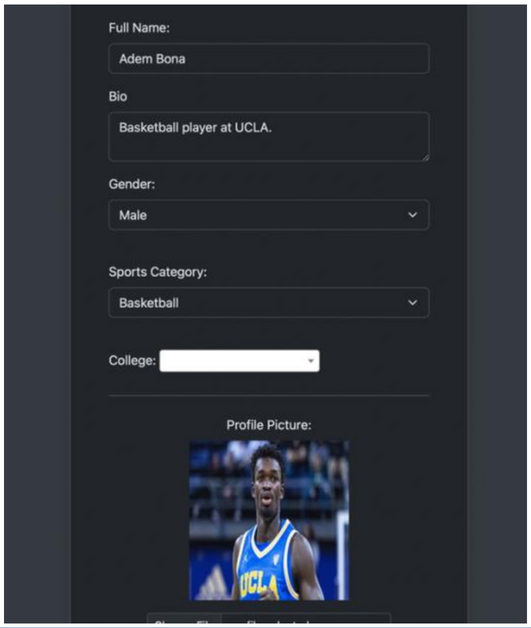
If the users are registered before, they will be able to sign in using their credentials which is the username and the password.

Register:



Companies or College Athletes who wants to use SponsorIN are required to register with their information to access the full functionalities of the software. Users will be able to register for SponsorIN by entering a username, email, password, and password once again one more time. At the end, user will be prompted to choose their type. In that case, they can either register as an Athlete or as a Company. No user can register as an admin since admins are the only people who can create other admin users.

Create – Edit Profile:



Full Name: Adem Bona

Bio: Basketball player at UCLA.

Gender: Male

Sports Category: Basketball

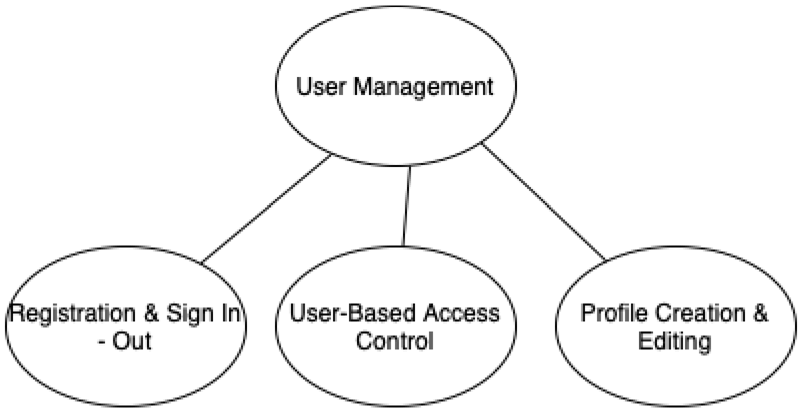
College:

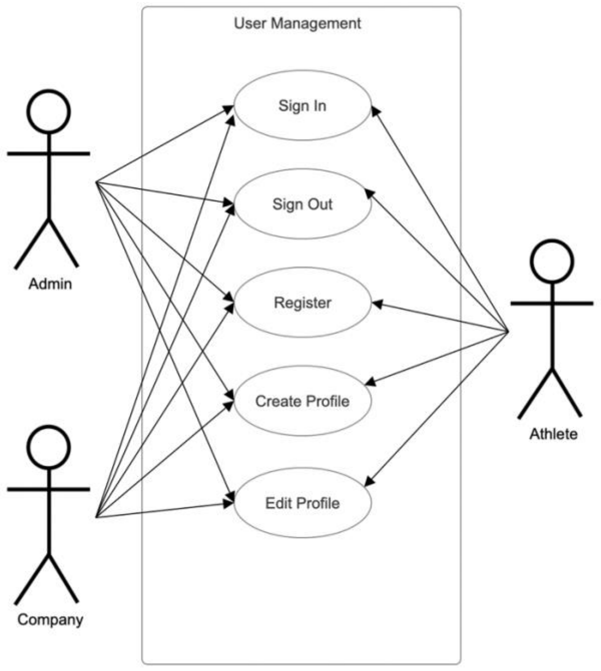
All athletes and companies registered to our system are required to create a profile for themselves. If they register as an athlete, they will need to create a profile that includes their full name, biography, gender, the sports they play, the college they attend and their profile picture. Company will only have biography, company name, full name for company's representative who uses the account and their logo.

Data Dictionary:

Table	Variable	Key	Type	Description
appuser				
	userid	Primary	Integer	ID # of the user
	username	No	Character	Username of the user
	password	No	Character	Password of the user
	usertype	No	usertype	Type of the user
	email	No	Character	Email of the user
profile				
	profileid	Primary	Integer	ID # of the profile
	userid	Foreign	Integer	ID # of the user associated with the profile
	fullname	No	Character	Full name of the user
	bio	No	Text	Biography of the user
	profilepicture	No	Text	Profile picture of the user
	verifiedstatus	No	Boolean	Status of the user that shows if the user is verified or not
athleteprofile				
	athleteprofileid	Primary	Integer	ID # of the athlete profile

	profileid	Foreign	Integer	ID # of the profile
	gender	No	gendertype	Gender of the user
	sportscategory	No	sportscategorytype	Sport the athlete plays
	collegeid	Foreign	Integer	ID # of the college athlete attends
companyprofile				
	companyprofileid	Primary	Integer	ID # of the company profile
	profileid	Foreign	Integer	ID # of the profile
	companyname	No	Character	Name of the company
	companylogo	No	Text	Profile Picture of the company

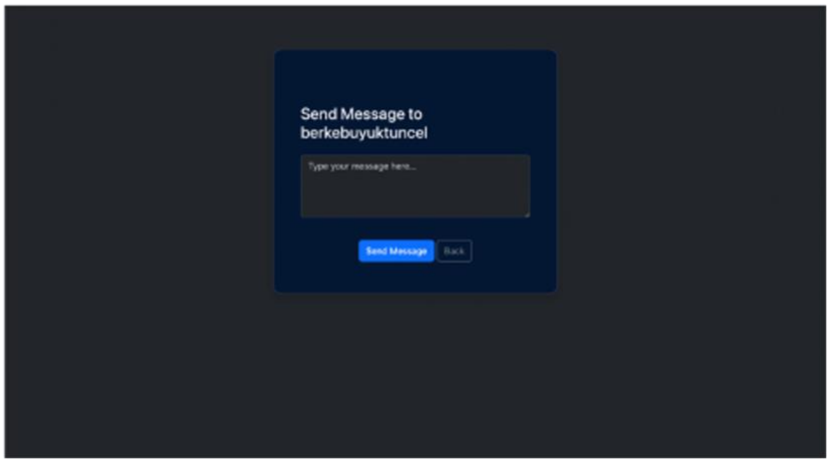

User Management
Registration & Sign In - Out
User-Based Access Control
Profile Creation & Editing



4.1.2 *Communication Subsystem:*

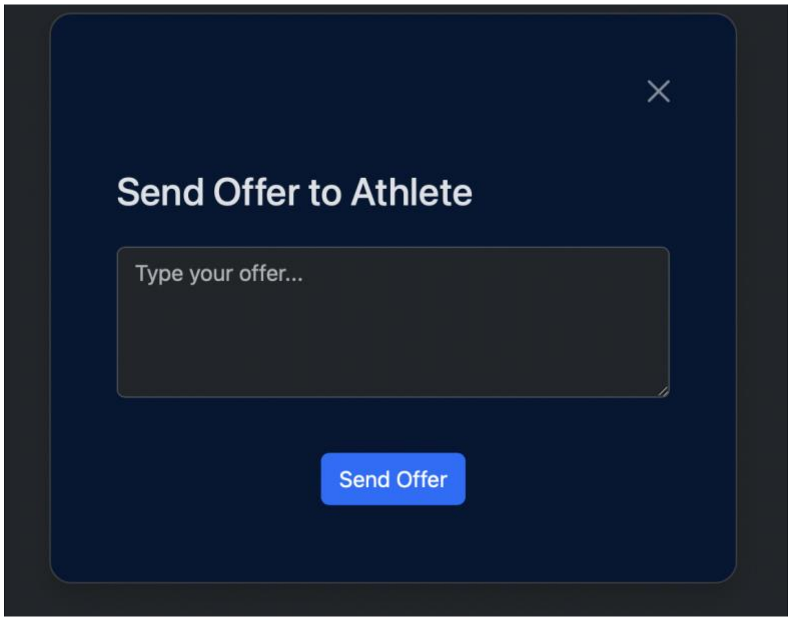
- **Description**: Allows in-app messaging between companies and athletes, including offer negotiations.
- **Components**:
- Messaging System.
- Offer Management (Create, Accept, Decline, Counter).

Send Message:



Users will be able to send messages to each other to build connections or if they need any type of communication or technical support from admins. Athletes will be able to send messages to admins and companies, but they will not be able to communicate with each other. Admins are allowed to send messages to both companies and athletes.

Send Offer:

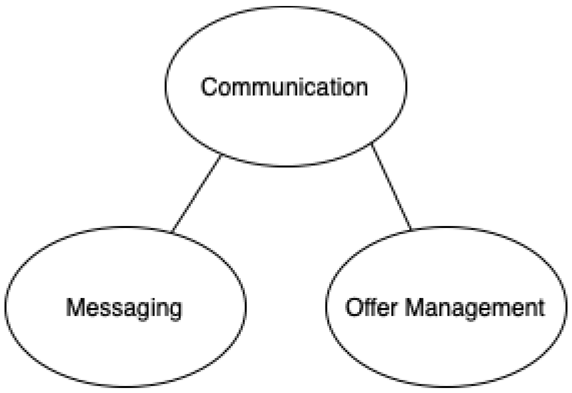


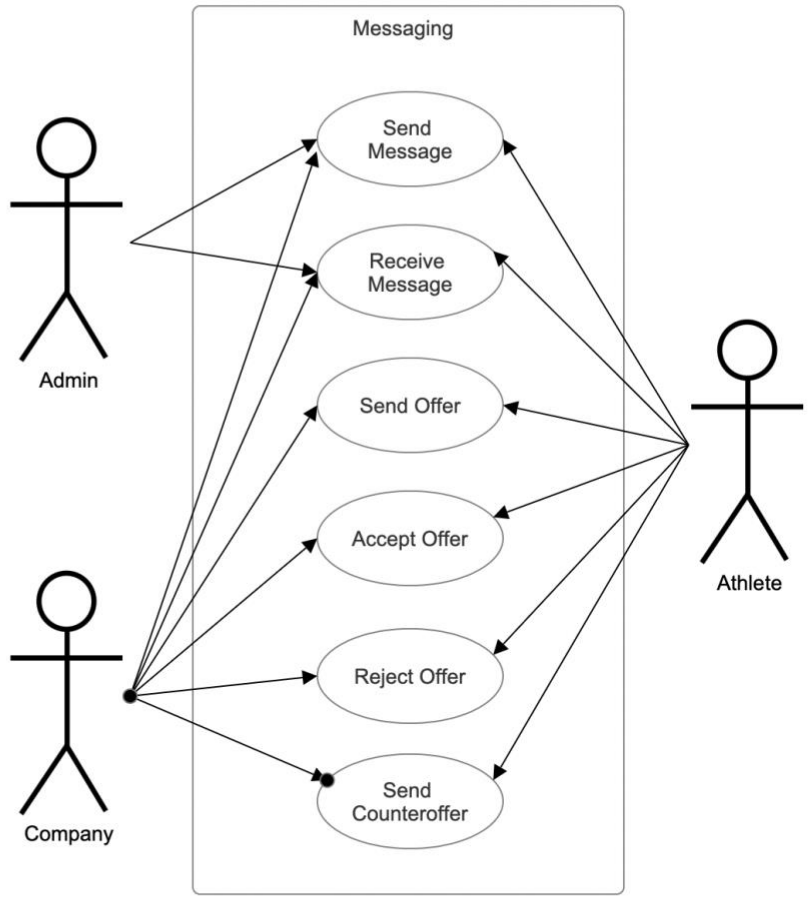
Users are allowed to send sponsorship offers to each other. First offer must be sent by a company to an athlete. Then, athletes will have the option to accept, decline or provide a counteroffer. There is no upper limit for counter limits between athletes and companies. Once an offer is sent by a company, they will have the ability to modify or retract the offer.

Data Dictionary:

Table	**Variable**	**Key**	**Type**	**Description**
message				
	messageid	Primary	Integer	ID # of the message
	senderid	Foreign	Integer	ID # of the user sent the message
	receiverid	Foreign	Integer	ID # of the user received the message
	content	No	Text	Message itself
offer				
	offerid	Primary	Integer	ID # of the offer
	companyid	Foreign	Integer	ID # of the company who gave the offer

	athleteid	Foreign	Integer	ID # of the athlete who received the offer
	details	No	Text	Details of the offer
	status	No	offerstatus	Status of the offer
	last_updated_by	No	updatedbytype	User who last updated the offer

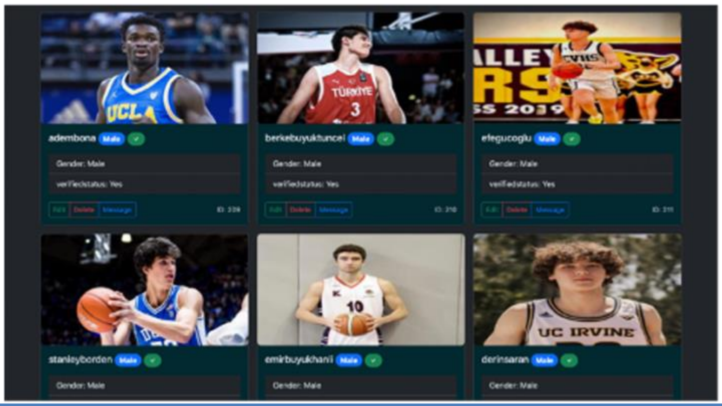




4.1.3 *Admin Control Subsystem:*

- **Description**: Provides control for administrators to oversee, validate, and intervene when necessary.
- **Components**:
- Profile Verification.
- User Editing & Deletion.

Profile Verification:


adembona Male ✓
Gender: Male
verifiedstatus: Yes
ID: 209

berkebuyuktuncel Male ✓
Gender: Male
verifiedstatus: Yes
ID: 210

efegucoglu Male ✓
Gender: Male
verifiedstatus: Yes
ID: 211

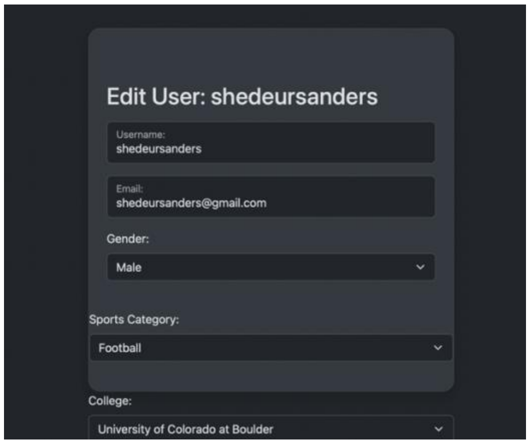
stanleyborden Male ✓
Gender: Male

emirbuyukhanli Male ✓
Gender: Male

derinsaran Male ✓
Gender: Male

Each user must have their profile verified and there is only one way of getting the profile verified. Once a user registered for SponsorIN and created their profile, the profile will come up as created but not verified. If a profile is not verified, no one will be able to see their profile except admins. If an athlete created their profile but it is not verified yet, companies will not be able to see this athlete. It's exactly the same situation for the companies too. Only users who can verify profiles are admins. There is no such a thing called self-verification.

Edit User:



Edit User: shedeursanders

Username: shedeursanders

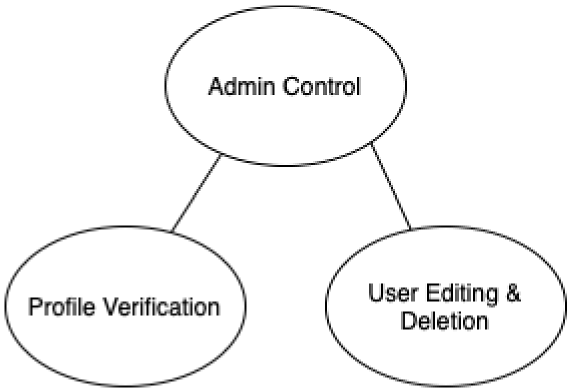
Email: shedeursanders@gmail.com

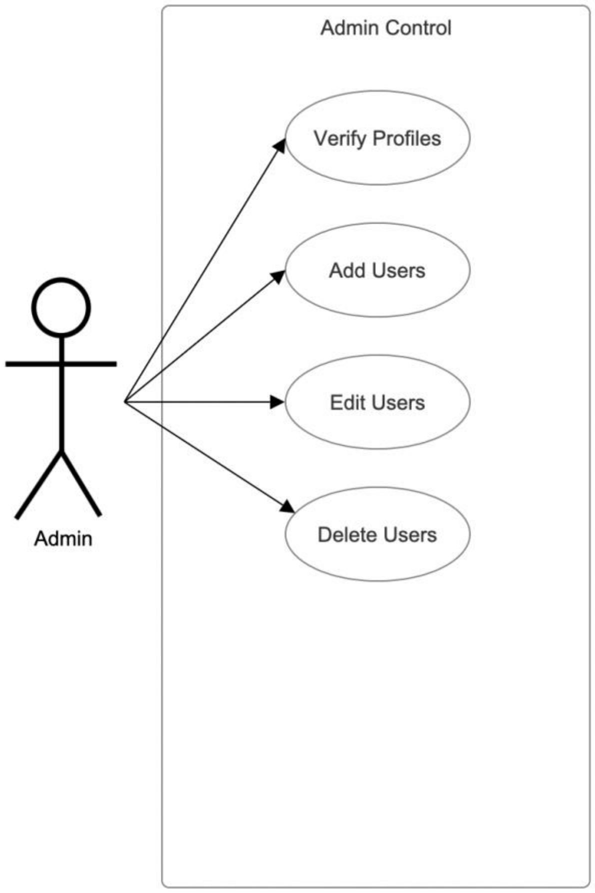
Gender: Male

Sports Category: Football

College: University of Colorado at Boulder

Admins will be able to edit any type of data belongs to the user profile of companies and athletes.

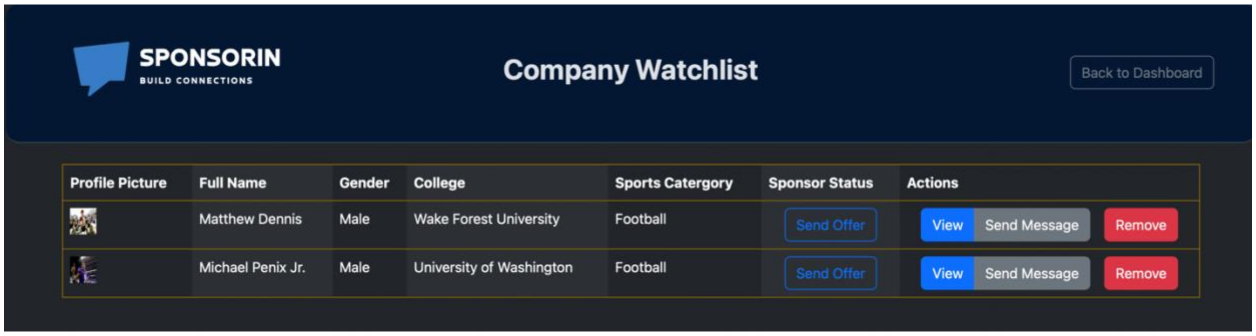




4.1.4 Company Subsystem:

- **Description**: For companies to find, save, and interact with potential athlete profiles.
- **Components**:
- Filtered Athlete Pool.
- Send Offers to Athletes.
- Organized Watchlist.

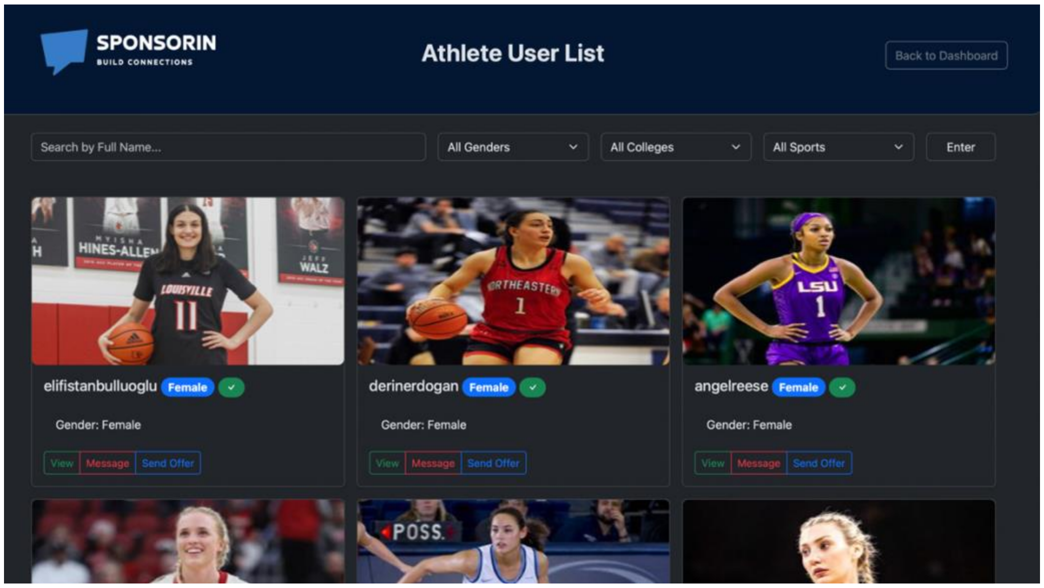
Organized Watchlist:



Profile Picture	Full Name	Gender	College	Sports Catergory	Sponsor Status	Actions
	Matthew Dennis	Male	Wake Forest University	Football	Send Offer	View Send Message Remove
	Michael Penix Jr.	Male	University of Washington	Football	Send Offer	View Send Message Remove

Companies will be able to create a watchlist for themselves to keep only the athletes they are interested in sponsoring. Using the watchlist, companies can send offers and send messages to athletes. They will always have the ability to remove an athlete from the watchlist.

Filtered Athlete Pool:



elifistanbulluoglu Female
Gender: Female

derinerdogan Female
Gender: Female

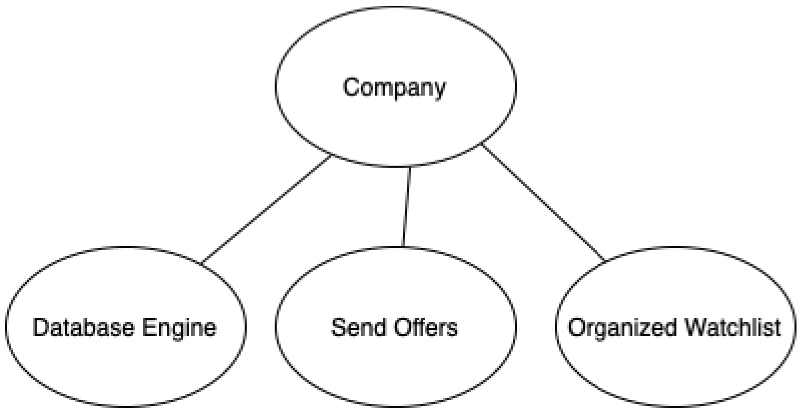
angelreese Female
Gender: Female

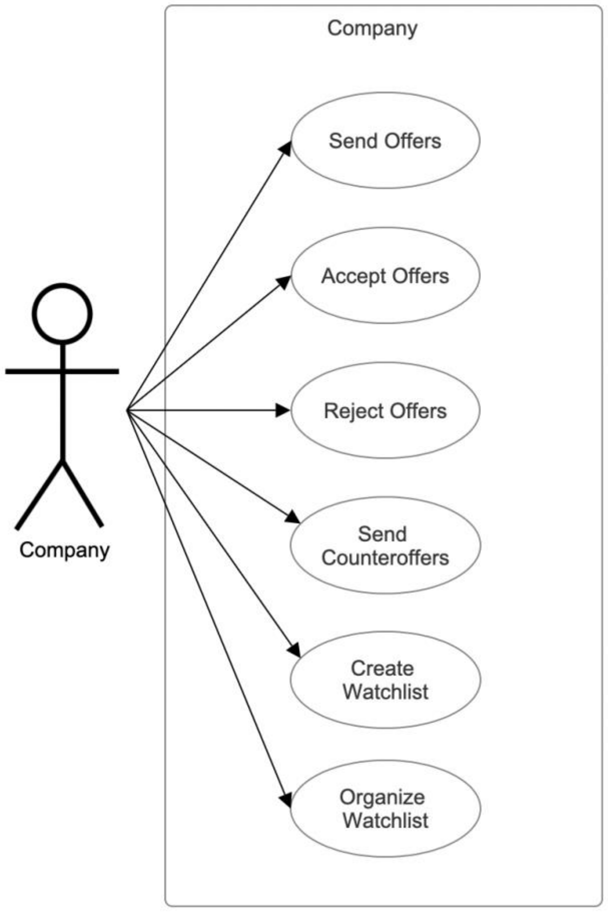
Companies are able to look at the large database of players from different categories. If they are looking for specific categories and choices, they have the option to use filtering and see the athletes related to their filters. Companies can use sports filtering and pick basketball, soccer, and football. They can also filter based on the colleges and gender.

Data Dictionary:

Table	Variable	Key	Type	Description
watchlist				
	watchlistid	Primary	Integer	ID # of the watchlist
	companyid	Foreign	Integer	ID # of the company who created the watchlist
	athleteid	Foreign	Integer	ID # of the athlete who got added to watchlist

	sportscategory	No	sportscategorytype	Type of sport this athlete plays
offer				
	offerid	Primary	Integer	ID # of the offer
	companyid	Foreign	Integer	ID # of the company who gave the offer
	athleteid	Foreign	Integer	ID # of the athlete who received the offer
	details	No	Text	Details of the offer
	status	No	offerstatus	Status of the offer
	last_updated_by	No	updatedbytype	User who last updated the offer

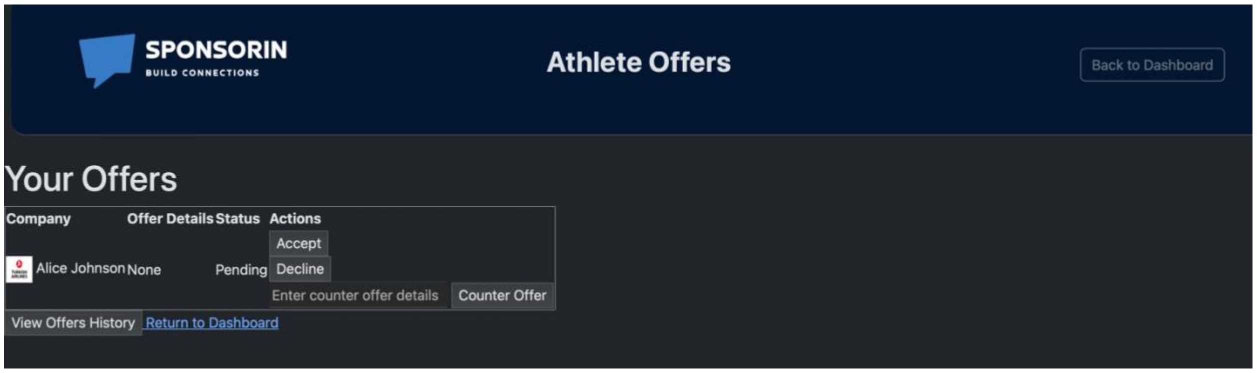




4.1.5 Athlete Subsystem:

- **Description**: Enables athletes to market themselves effectively and interact with companies.
- **Components**:
- Profile Customization & Showcase.
- Accept/Decline/Counter Offers.

Accept / Decline / Counter Offers:



Athlete Offers

Your Offers

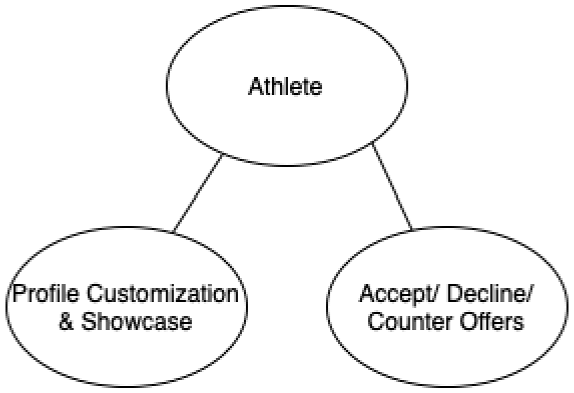
Company	Offer Details	Status	Actions
Alice Johnson	None	Pending	Accept Decline Enter counter offer details Counter Offer

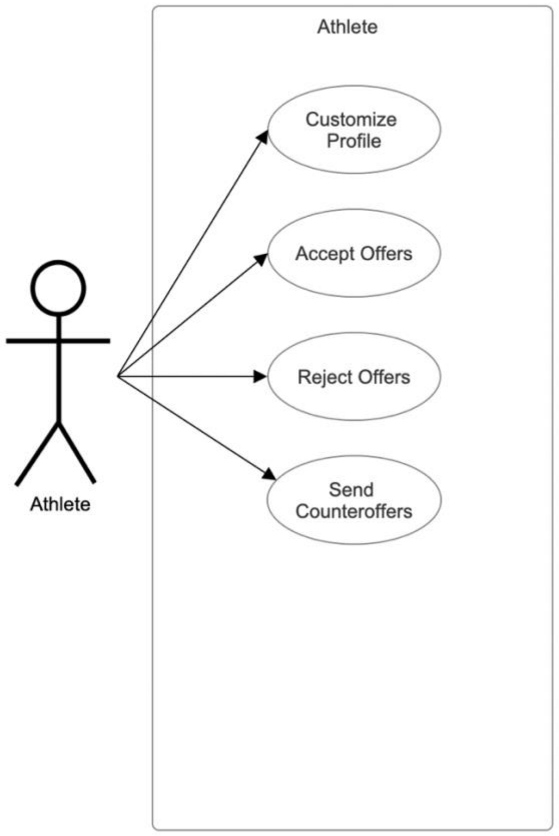
View Offers History Return to Dashboard

Athletes will be able to accept, decline or give a counteroffer and ask for better conditions. However, they will never be able to be the one who gave the first offer.

Data Dictionary:

Table	Variable	Key	Type	Description
athleteprofile				
	athleteprofileid	Primary	Integer	ID # of the athlete profile
	profileid	Foreign	Integer	ID # of the profile
	gender	No	gendertype	Gender of the user
	sportscategory	No	sportscategorytype	Sport the athlete plays
offer				
	offerid	Primary	Integer	ID # of the offer
	companyid	Foreign	Integer	ID # of the company who gave the offer
	athleteid	Foreign	Integer	ID # of the athlete who received the offer
	details	No	Text	Details of the offer
	status	No	offerstatus	Status of the offer
	last_updated_by	No	updatedbytype	User who last updated the offer

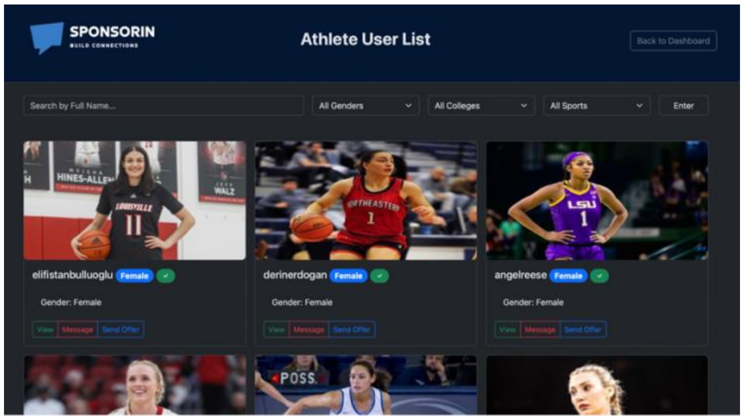

Athlete
Profile Customization & Showcase
Accept/ Decline/ Counter Offers



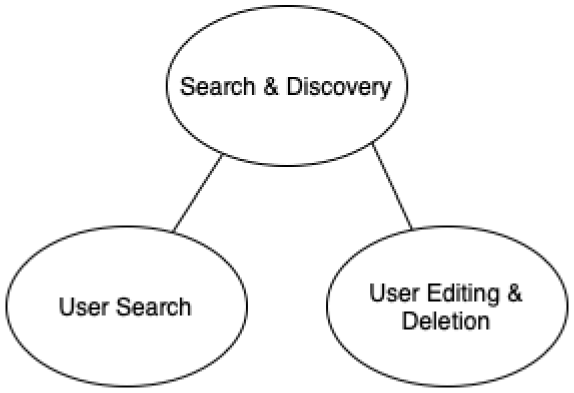
4.1.6 Search & Discovery Subsystem:

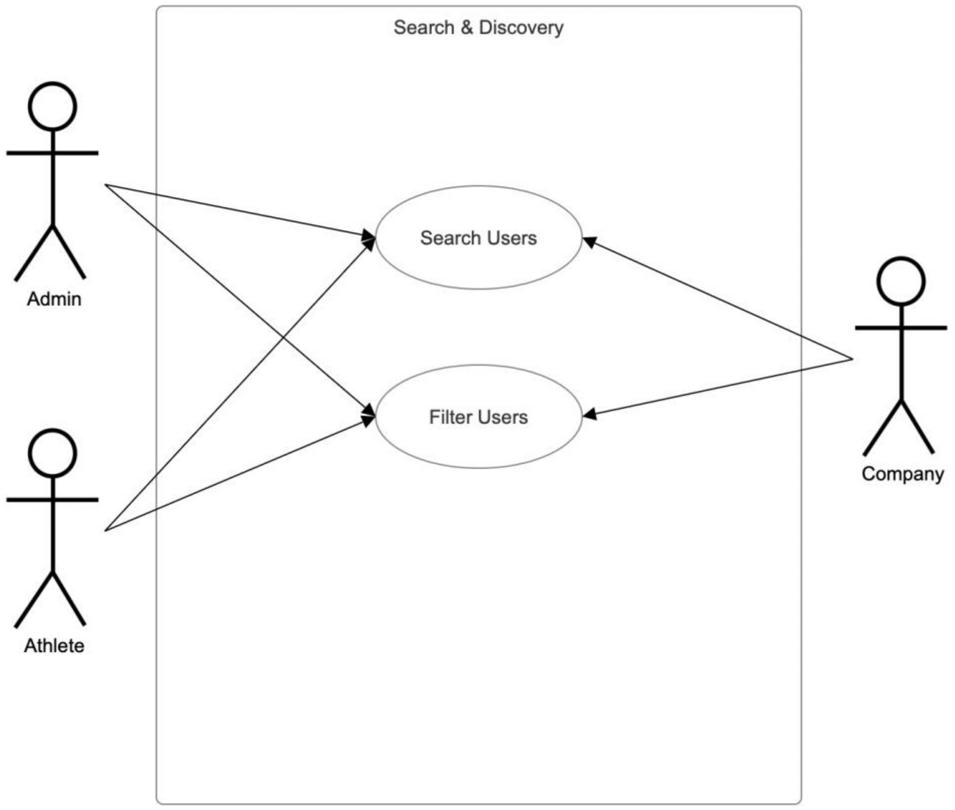
- **Description**: Provides a search capability for both companies and athletes.
- **Components**:
- User Search
- Advanced Filtering Options

User Search – Advanced Filtering:



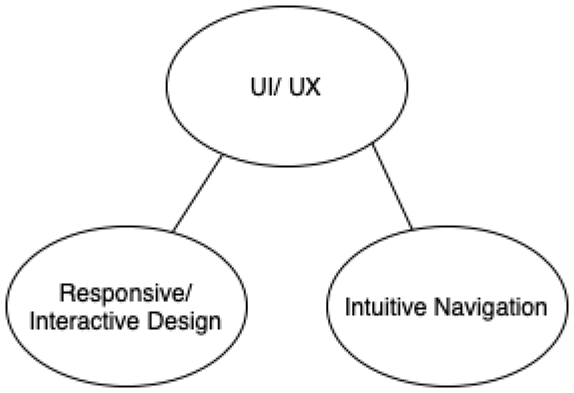
Admins, companies, and athletes are all allowed to search for certain type of users. They also have the ability of using filtering options to match a user with their potential requests and needs.





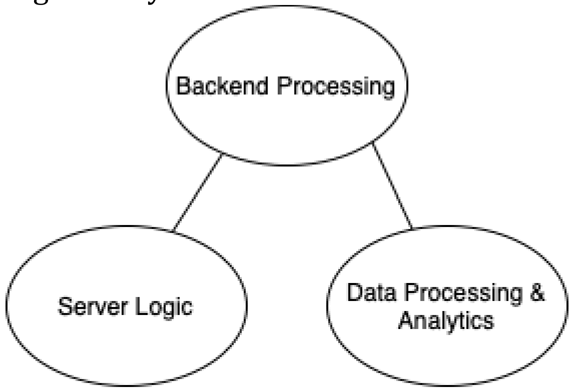
4.1.7 UI/UX Subsystem:

- **Description**: Focuses on delivering an optimal user experience across devices.
- **Components**:
- Responsive - Interactive Design.
- Intuitive Navigation.



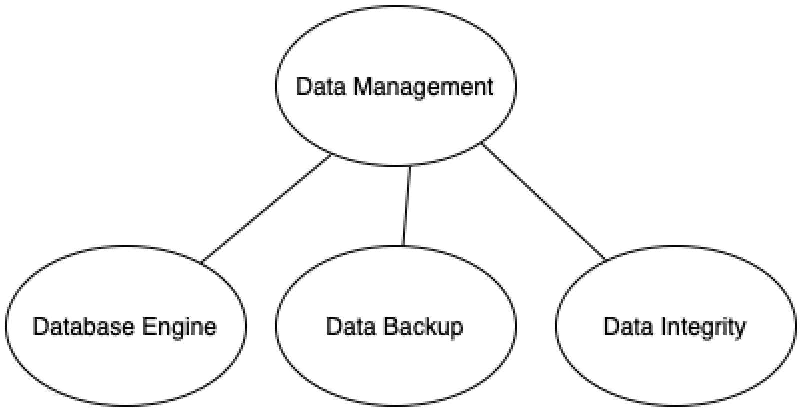
4.1.8 Backend Processing Subsystem:

- **Description**: Deals with the server-side logic, data processing, and other computations.
- **Components**:
- Server Logic (Python Flask).
- Data Processing & Analytics.



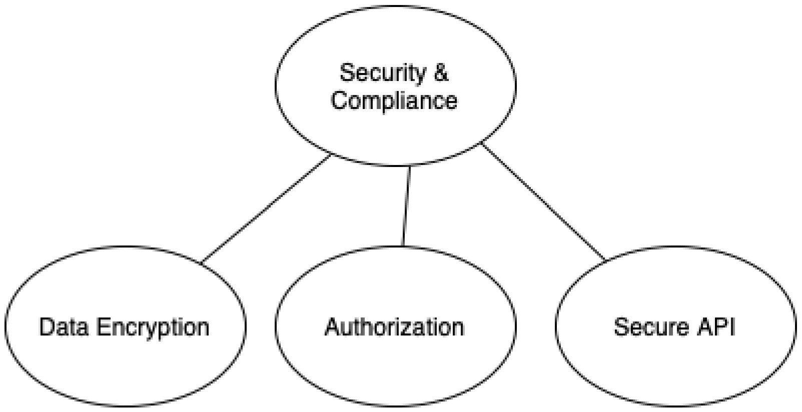
4.1.9 Data Management Subsystem:

- **Description**: Ensures secure, efficient, and reliable data storage and retrieval.
- **Components**:
- Database Engine (PostgreSQL).
- Data Backup & Recovery.
- Data Integrity.



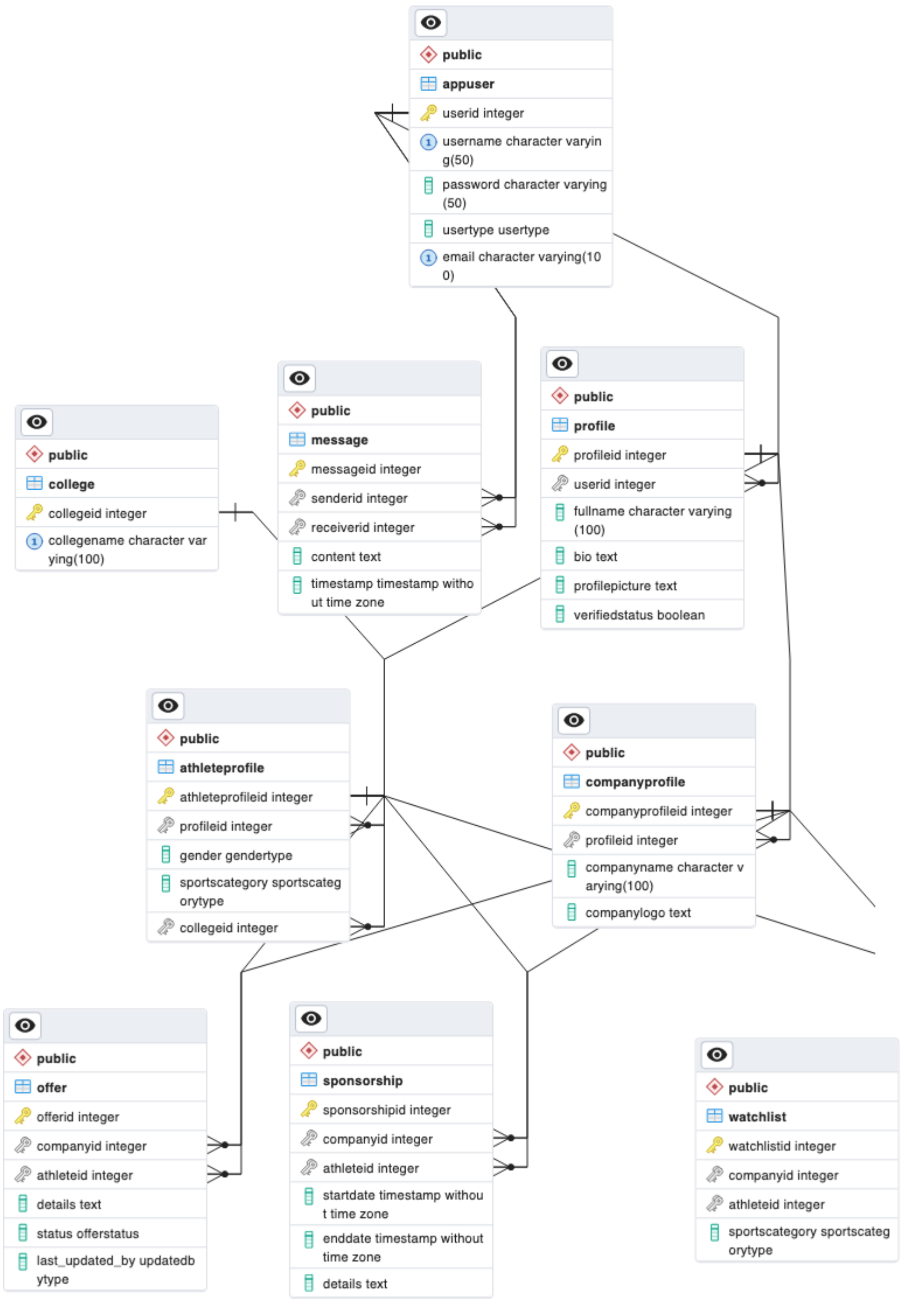
4.1.10 Security & Compliance Subsystem:

- **Description**: Protects user data, ensures secure communication, and enforces platform integrity.
- **Components**:
- Data Encryption (HTTPS, SSL).
- Authorization & Role-based Permissions.
- Secure APIs.



4.2 Database Design – Entity Relationship Model and Data Dictionary

4.2.1 Entity Relationship Model



4.2.2. Database Design

SponsorIN's data architecture is running on PostgreSQL for our database engine. To complement it, we utilize pgAdmin 4, an intuitive GUI that makes database interactions more user-friendly for our team. Our database schema structured into nine distinct tables, each serving a specific function within our platform's ecosystem. We created custom enum types, providing a set of predefined values that reflect the various roles, offer statuses, and sport categories. These types ensure data consistency and enhance the readability of our database logic.

Enums:

UserType: This enum defines the roles within our community— Admin, Company, and Athlete which helped us creating three distinct dashboards and allowed us to declare which user type can access which functionalities.

GenderType and SportsCategoryType: These enums categorize our athletes, ensuring we have a diversity with users and sports.

OfferStatus: This enum tracks the progression of negotiations from pending to accepted, declined, and counteroffered, providing clear communication channels and transparency.

Tables:

AppUser: This table is used when a user is signed up on our platform. It securely holds usernames, passwords, and email addresses, along with each user assigned a distinct role such as if they are an admin, athlete, or company.

Profile: Extending from AppUser, the Profile table adds more details about the users such as the usernames, biographical information, and even a profile picture.

College: This table stores all the colleges in the United States, so athletes are able to select which college they are currently attending.

AthleteProfile and CompanyProfile: These tables branch out from Profile, tailored to the specific needs of our two main user type. AthleteProfile has gender type, sports category, and the college they are attending. CompanyProfile focuses on corporate entities like company name and a field to upload company logo.

Message: This table is used to help create our messaging system in our platform as it stores the messages sent between users and creating a unique primary message id for each new message created. senderID references appuser table to see which user sent the message. ReceiverID shows which user is receiving the message. A timestamp field is also added to store the time and date when the message was sent.

Offer: This table is used to help create our offer system to capture and manage the sponsorship proposals as it serves a record system of the ongoing engagement between companies and athletes. Each record in the Offer table is uniquely identified by an OfferID, ensuring every transaction is traceable and distinct. The CompanyID column establishes a direct link to the CompanyProfile table, identifying the company user, extending the sponsorship. AthleteID connects to an AthleteProfile and pinpointing the athlete under consideration for sponsorship. Details column provides a field to describe the offer and the amount being offered. Finally, we have a Status column which tracks the progression of the offer through predefined stages which works with our predefined enum type offerStatus. Status column is initially set to 'Pending'. It reflects the current state of negotiations transitioning from pending to acceptance, decline, or even counteroffer.

Watchlist: Watchlist table is used to create the functionality for company users as they create their own list of athletes who they consider sponsoring.

Sponsorship: This table is used to store and finalize sponsorship deals between the athletes and companies.

4.2.3. Data Dictionary

Table	Variable	Key	Type	Description
appuser				
	userid	Primary	Integer	ID # of the user
	username	No	Character	Username of the user
	password	No	Character	Password of the user
	usertype	No	usertype	Type of the user
	email	No	Character	Email of the user
college				
	collegeid	Primary	Integer	ID # of the college
	collegename	No	Character	Name of the college
message				
	messageid	Primary	Integer	ID # of the message
	senderid	Foreign	Integer	ID # of the user sent the message

	receiverid	Foreign	Integer	ID # of the user received the message
	content	No	Text	Message itself
	timestamp	No	timestamp	Time message was sent
profile				
	profileid	Primary	Integer	ID # of the profile
	userid	Foreign	Integer	ID # of the user associated with the profile
	fullname	No	Character	Full name of the user
	bio	No	Text	Biography of the user
	profilepicture	No	Text	Profile picture of the user
	verifiedstatus	No	Boolean	Status of the user that shows if the user is verified or not
athleteprofile				
	athleteprofileid	Primary	Integer	ID # of the athlete profile
	profileid	Foreign	Integer	ID # of the profile
	gender	No	gendertype	Gender of the user
	sportscategory	No	sportscategorytype	Sport the athlete plays
	collegeid	Foreign	Integer	ID # of the college athlete attends
companyprofile				
	companyprofileid	Primary	Integer	ID # of the company profile

	profileid	Foreign	Integer	ID # of the profile
	companyname	No	Character	Name of the company
	companylogo	No	Text	Profile Picture of the company
offer				
	offerid	Primary	Integer	ID # of the offer
	companyid	Foreign	Integer	ID # of the company who gave the offer
	athleteid	Foreign	Integer	ID # of the athlete who received the offer
	details	No	Text	Details of the offer
	status	No	offerstatus	Status of the offer
	last_updated_by	No	updatedbytype	User who last updated the offer
sponsorship				
	sponsorshipid	Primary	Integer	ID # of the sponsorship
	companyid	Foreign	Integer	ID # of the company who sponsors athlete
	athleteid	Foreign	Integer	ID # of the athlete who gets sponsored
	startdate	No	timestamp	Start date of the sponsorship
	enddate	No	timestamp	End date of the sponsorship
	details	No	Text	Details of the sponsorship
watchlist				

	watchlistid	Primary	Integer	ID # of the watchlist
	companyid	Foreign	Integer	ID # of the company who created the watchlist
	athleteid	Foreign	Integer	ID # of the athlete who got added to watchlist
	sportscategory	No	sportscategorytype	Type of sport this athlete plays

4.3 System Design Process and Technical Preferences

The development process began with the design of the database schema. This foundational step involved defining how data would be organized, how different entities would relate to each other, and ensuring the structure supported the platform's functional requirements.

Database Schema Design

- **Identifying Entities and Relationships:** The primary entities were identified as Admin, Athlete, and Company, each having unique attributes and roles within the system. Relationships between these entities were established to reflect real-world interactions.
- **Schema Creation:** Using tools like ER diagrams, a detailed database schema has been created. This included tables for user information, profiles, messaging, offers, and any other relevant data.

Language and Framework Selection

- **Choosing Python and Flask:** Flask has been used as the main framework due to its simplicity and modular design. Flask's flexibility was key in building lightweight, service-oriented web applications.
- **PostgreSQL as Database Engine:** PostgreSQL was selected for its robustness and compatibility with SQLAlchemy, an ORM tool used for database interactions.
- **pgAdmin 4 for Database Management:** To manage and interact with the PostgreSQL database, pgAdmin 4 was chosen for its comprehensive GUI, facilitating easier database administration and querying.

Frontend and Backend Development

- **Frontend Development with Bootstrap and JavaScript:** Bootstrap and JavaScript has been used as the main frontend tools. This decision aligned with Flask's clear separation of backend and frontend development.
- **Backend Structure in app.py:** The core application, app.py, served as the central routing mechanism, directing user requests appropriately.

Building Functionalities

- **Developing Distinct Dashboards:** The team first developed three distinct dashboards for Admin, Athlete, and Company users, focusing on their unique functionalities.
- **Coding Shared Functionalities:** After establishing the distinct features, shared functionalities like messaging, profile viewing, and search were implemented.
- **Testing with Basic Templates:** Initially, basic templates were used to test functionalities. Once confirmed working, these were enhanced with Bootstrap and JavaScript for a better user experience.

Data Population and Finalization

- **Real Data Integration:** With the dashboards and functionalities in place, the system was populated with real data to emulate actual usage scenarios. Real data has been collected through NCAA and companies' website and combined with a profile picture for a better-looking profile. Using the collected information, Excel spreadsheets has been created to store the information in a more convenient way for our developers. Then, all the accounts and profile has been created one by one and manually to make sure every information is accurate and functional.
- **Final Design and Testing:** Final design touches were applied to the application, ensuring the application was both functional and aesthetically pleasing.
- **Testing and Iteration:** The entire system underwent thorough testing, with feedback loops in place for continuous improvement and refinement.

This approach ensured a clear division of labor and a structured development process, enabling the team to efficiently build a robust platform catering to the specific needs of college athletes, companies, and admins.

Flask Libraries and Functions Used

- **User Authentication:** Flask-Login has been integrated for managing user sessions, a pivotal functionality for a secure user experience and handling user types. Routes like /login and /logout helps managing the user authentication by ensuring that only authorized users can access sensitive areas of the platform.
- **Database Interaction:** SQLAlchemy has been integrated for interacting with PostgreSQL database, allowing us to manage complex data models. Database schema definition process, data transaction handling process became possible,

and atomicity, consistency, isolation, and durability compliances have been ensured by using SQLAlchemy.

- **User Experience:** Backend is engineered to provide feedback to the user through Flask's flash messaging function. With flash messaging functionality implemented, users got real time updates based on the success or failure of their actions like signing up and signing in.

4.4 Refinements of Design

Revised Overview of Database Schema Refinement

Initial Model

The initial database schema for the SponsorIN began with five primary tables: appuser, profile, message, offer, and watchlist. This setup was intended to cover the basic functionalities such as user management, communication, offer management, and interest tracking.

Reasons for Refinement

- **Pro - Enhanced Specificity:** The need to address specific requirements for athlete and company profiles, not fully supported by the initial model.
- **Pro - Comprehensive Sponsorship Management:** The initial schema lacked a dedicated mechanism to manage sponsorship details effectively.
- **Con - Limited College Data Representation:** Absence of a dedicated structure to handle crucial college-specific information for athlete profiles.
- **Con - Data Segregation and Management:** The initial model's approach to profile management led to potential data redundancy and inefficiencies.

Changes from Initial Model

- **Introduction of AthleteProfile and CompanyProfile Tables:** Separated the profile table into athleteprofile and companyprofile for more distinct and effective data management.
- **Addition of College Table:** Introduced a college table to store and manage college-specific information linked to athletes.
- **Introduction of Sponsorship Table:** Added a sponsorship table to enhance the handling of sponsorship agreements between athletes and companies.
- **Rearrangement of Watchlist Table:** Maintained the watchlist table from the initial model but modified its role and relations due to the introduction of new tables.

Refined Model Analysis

The refined database schema provided a more specialized and effective structure. By introducing specific tables for athlete and company profiles, and a separate table for college information, the schema became more aligned with the real-world data model of the SponsorIN platform. The addition of the sponsorship table allowed for a more comprehensive management of sponsorships, addressing a key aspect of the platform's functionality.

Refined Design

The refined database schema can be summarized as follows:

- **User Table (**appuser**):** The central table for common user data.
- **Profile (**profile**):** Main table for common profile data.
- **Athlete Profile (**athleteprofile**):** Specifically for athlete-related data.
- **Company Profile (**companyprofile**):** Tailored for company-related data.
- **College (**college**):** Dedicated to storing college-related information.
- **Message (**message**):** For user-to-user communication.
- **Offer (**offer**):** To handle sponsorship offers.
- **Watchlist (**watchlist**):** Enabling companies to track athletes of interest.
- **Sponsorship (**sponsorship**):** For managing sponsorship agreements.

This revised schema better accommodates the complexities of the SponsorIN platform, offering an improved structure for data management and functionality. The separation of user profiles into athlete and company categories, along with the addition of college and sponsorship details, greatly enhanced the system's efficiency and usability.

4.5 Subsystem Algorithm Analysis

Employing foundational if statements and singular for loops, SponsorIN's architecture aligns with the principles of simplicity and efficiency. The use of for loops, operating at a time complexity of O(n), strategically retrieves data from our database and seamlessly integrates it into the frontend. This emphasis on O(n) efficiency underscores our commitment to delivering a high-performing system, capable of handling diverse data loads with optimal speed. SponsorIN's design philosophy centers on user-friendly simplicity while simultaneously prioritizing scalable efficiency, resulting in a platform that is not only intuitive but also robust in its ability to adapt to varying demands.

5. Final Product

- **Source Code**
  Source code for the SponsorIN project can be accessed through the link below:
  https://github.com/twosize/SponsorIN
- **Introduction and Demo Video**
  Introduction and demo video can be accessed through the link below:
  https://youtu.be/hkl0gRBrgX0?feature=shared
- **Team Members**
  1- Brandon Tiong: https://github.com/twosize
  2- Emanuel Macias: https://github.com/DarkNeo777
  3- Ege Keser: https://github.com/egekeser