UA Sports Hub
Sports App/GUI

By: Mark Roche, Jovan Vance
Shabeeb Shah

ECE 373

Problem Statement

With the evolution of technology granting us easier access to information, we have hundreds, even thousands, of places to find information that we are looking for. Sometimes we are presented with too much information, and as a result cannot find what we are looking for. This may be a frustrating experience for some people who may be looking for something specific and may need it quickly, or just people who are in need of information. This problem also extends into the sports journalism and information domain.

There are hundreds of sources to get the latest and breaking sports news. These sources come from established heavyweights such as ESPN and CBSSports, as well as up and coming sources such as Bleacher Report and theScore. There are also popular blogs one can visit, articles written by newspaper beat writers, as well as all of the news coming from the sports club itself. With all of these sources available at the flick of a finger, there is a ton of information to parse through to find what you are looking for. Websites like ESPN and Bleacher Report allow users to follow specific sports and teams, although this is restricted to information coming from that specific website. Any die hard fan of their team isn't going to want the latest information from just one source, they will want any relevant information from every source possible, and this is what the UA Sports Hub sets out to do.

The UA Sports Hub is designed to get all of the desired information in one place. With an account in the UA Sports Hub, a user will get any information about the teams and sports the user wants to follow, all in one central location. The user will be able to see articles from multiple sources through the UA Sports Hub. In addition to viewing articles from outside sources, users will also have the ability to read blogs written by other users on the UA Sports Hub. Users can also comment on articles and blogs, which gives the sports hub a social networking aspect.

In Summary, the UA Sports Hub is designed to bring all of the news and information regarding U of A sports into one central location. Users can view articles from sites such as

ESPN and Grantland through the Hub, and will also be able to socialize with other users. Users will have easy access to see the teams and their players through the app as well. Basically, the app is a one-stop-shop for anything Arizona sports.

Program Boundaries

Given the nature of the project, the group wanted to make an app like interface to create the Sports Hub. We mainly wanted have a way for a user to easily create a customizable account and to get a unique experience in the Hub. We wanted the user to create an account and gain access to the Hub, where the user can then do multiple things. Once the user has access to the Hub, one of the things they can do is quickly view the teams currently on the hub (not all of the teams are currently in the Hub, but its off to a good start), and then proceed to view the players on the team. Another thing the user is able to do is write blogs and comment on blogs. User written blogs will be viewable by all users and all users will be able to discuss the blog through use of comments on the blog. The user can also view articles from other sites, however they will not be able to comment on the articles. Lastly, users have quick access to the five most recent games on the home screen of the Hub.

As it stands, most of the buttons on the pages do nothing. There is one button on each page that does work so that there is enough to show of the functionality of the program. All of the buttons on the main home screen are functional, besides the big middle button which serves as a picture. Clicking the "Teams" button will take you to a page with eight buttons representing eight different teams. All of these buttons are not functional except for the mens basketball team photo, which upon being clicked, takes you to a page with all of the players and coaches on the basketball team. Only the button with "Aaron Gordon" is function, which upon clicked will take you to his player profile. All the other buttons on the home screen are fully functional. Users

have full ability to view articles, recent blogs, recent games, and their profile. In summary, everything is functional besides full view of every player and team.

Domain Analysis

The current implementation of the program is based on 13 classes: People, Athlete, Coach, User, Admin, Team, Blog, Article, Media, Venue, Sport, Game, and Comment. These classes are essentially the backbone of the program as they contain all of the information that is to be displayed through the Hub.

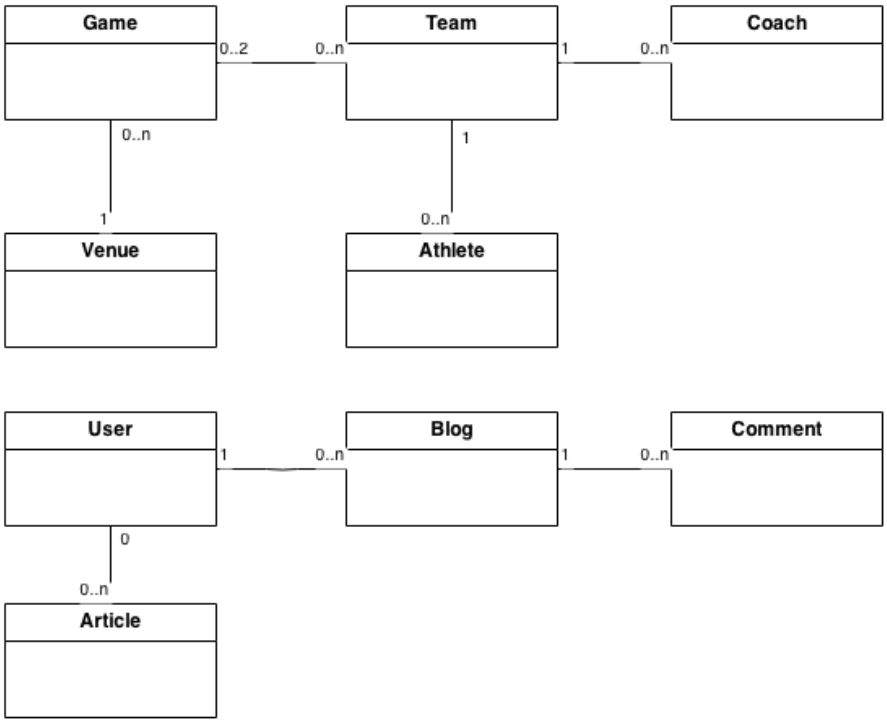


Figure 1: High Level UML of the classes that make up the structure of the UA Sports Hub

Although these classes form the base of the program, they were the most simple to write as well. The majority of the classes just had a lot of instance variables that pertained to what we wanted to be on the program. The large majority of the methods in the classes ended up being getters and setters. In addition to these base 13 classes, we have an additional 19 classes that all pertain to the graphical user interface. Most of the work ended up going into these 19 classes as these determined how the program would look and behave.

The classes above also had to communicate with each other a fair amount. On a high-level view, the Hub contains many Teams, Games, Athletes, Coaches, Users, Admins, Articles,

Blogs, Venues, and Comments. Teams have a list of Athletes and a list of Coaches. Games have a Home Team and an Away Team, as well as a venue. Users can have a list of Blogs they have written and a list of Comments they have submitted. Blogs have a list of Comments associated with them. Venues contain a list of teams that play in that venue, as well as a list of games that are played in that venue.

In terms of the program hierarchy, the SportsHub class contains every other class. It has a list of everything added to the Hub, for example a list of every athlete in the Hub. The Person class is abstract, reason being that we didn't want Person to actually be instantiated, as well as it contains information that Athletes, Coaches, and Users have. Going along that line, Athlete, Coach, and User extended Person, again because Person contained variables that these classes also should have. Going down one more level, the Admin class extended User because admins are essentially users with special privileges on the Hub. Media was another abstract class that we made. The Media class contained a title variable and a body variable. Both the Article and Blog class extended the Media class because both classes needed a title and body, but the way authors are handled in the classes warranted them being different classes.

Our initial domain analysis yielded that we would need 3 packages: Event, People, and Media. In the Event package would be Game, Sport, Team, and Venue. In the People package would be Person, Athlete, Coach, User, and Admin. Lastly, in the Media package would be Article, Media, and Blog. After a more thorough analysis, we added a Comment class to the Media package and added a Software package. The software package was not included in the initial analysis because the software package was to contain all of the GUI classes, which we did not come up with until work on the user interface started. Although the Venue and Sport classes are in the current implementation, they are not used. Ideally these would be used in future implementations, as will be discussed in the extensibility section.

Interaction Analysis

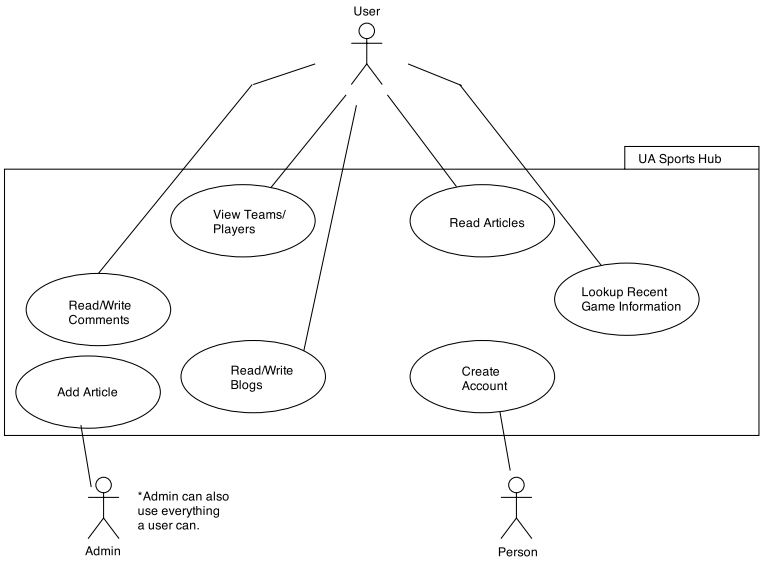


Figure 2: Use Case diagram for UA Sports Hub. Describes what different kind of people have the ability to do when accessing the UA Sports Hub.

I. Use Case

Figure 1 shows the Use Case diagram for a user, admin, and person. An account is needed to access the UA Sports Hub. A person who is not a user or admin can create an account, at which point they will become a user. Users have access to the UA Sports Hub. Once on the app, users can view teams and the player profiles of the players on that team, read/write comments, look up game information, read user written blogs, write blogs, and read related articles. Admins can do everything that a user can, but admins are the ones responsible for adding new articles to the app.

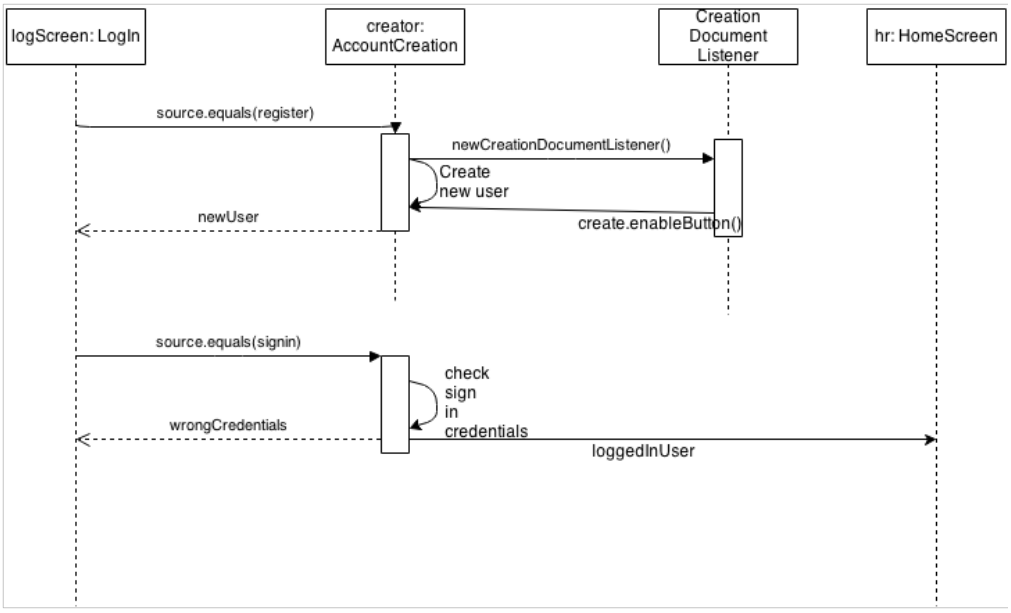


Figure 3: Sequence diagram for logging on to the UA Sports Hub

II. Boundary Cases

The boundary cases arise based on what kind of person is trying to use the app. If a person without account is trying to access the app, they will not be able to access the app. If they perhaps tried to enter account information from another website, such as ESPN, the program should not let the person access the UA Sports Hub, unless the username and password are registered in the app. Another boundary case arises in determining who can add articles to the site. We wanted it organized so that not every user can add articles to the website so that the articles can be as relevant as possible. The admin class was made so that it would have all the functionality of a user, as well as the ability to add articles. When an admin logs on, there is a button present in the upper left hand corner of the UI that says ”Add Article”. Clicking this button brings up a window that prompts the admin to enter the article title and source URL. This button is only visible to admins. When a regular user logs in, they will not be able to see the button, therefore are restricted from adding new articles. They can, however, see the newest articles that the admins uploaded.

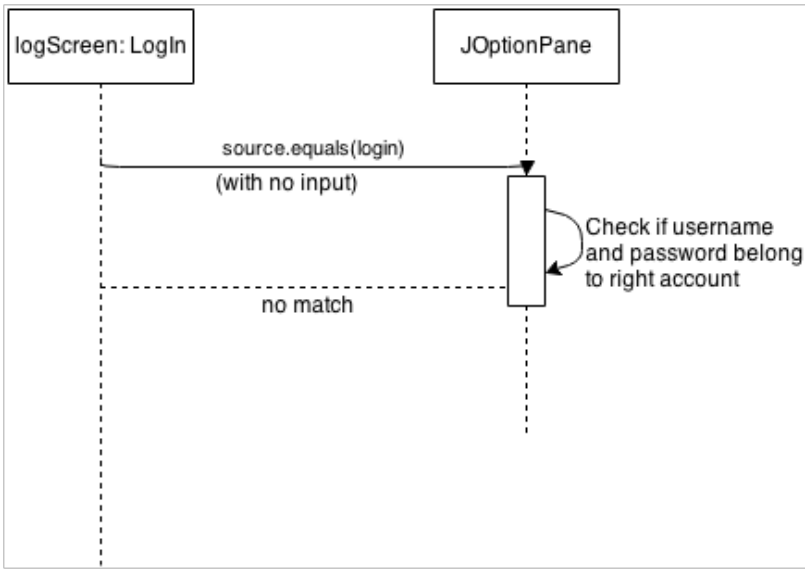


Figure 4: Boundary case sequence diagram for when a user logs in with incorrect log in credentials.

Implementation

The final version of the program that is being turned in is a partially finished product. However, there is enough in the app so that the functionally of the whole app can be seen. Upon clicking the “Teams” button from the home screen, the user is presented with eight buttons representative of eight different Arizona athletic teams. However, only one of the buttons actually performs an action. Clicking the picture of the men’s basketball team takes the user to a page that shows all of the players and coaches on the team. Again, only one of the buttons on this page actually performs an action. Clicking the button for ‘Aaron Gordon” takes the user to a short player profile page for Aaron Gordon. From this page, the user can only go backwards to the main home screen as this is how far we intended the user to go. The player profile page is incomplete as well; given more time we wanted to add more information such as seasonal stats for the player. Although not all of the buttons on the pages are functional, there are enough functional so that the functionality of the program can be properly seen.

The program gets all of its initial information from running driver1. Driver1 creates three users and an admin, as well as creates eight different Arizona athletic teams, five different athletic games, five blogs, five articles, and a basketball team that includes sixteen players and three coaches. There is another driver, “driver2”, which loads the information made from driver1 and creates the sports hub based off the information. Throughout various parts of the program, such as adding an article or blog, the sports hub is saved so that it can save the changes made and load them up next time driver2 is run.

As we had planned, most of the hard work was done in the GUI classes and not in the foundational classes described earlier. The program didn't require any complex algorithms, but we wanted the program to function similar to how a website functioned. We wanted the app to run mainly off of one frame, so one complicated part was navigating through the program using just one frame. This was done by using a card layout on the main panel. Action listeners were

added to the buttons, and when they were pressed, the card layout made visible the panel that the button was linked to. Once the button took us to the desired panel, we made a back button on the top left of every linked panel would take us back to the previous panel we came from. This essentially gave us the browser-like functionality that we desired.

For some of the elements of the program, we had to look online for additional resources. For example, the articles pulled from the internet were initially displayed poorly, but readable based on the resources from the java libraries. Upon further research, we discovered a created library that provided a relatively easy way to embed a browser into the application, thus allowing for a perfect display of the article and also allowing the user to freely roam around the linked website, which the user was not able to do with the initial way the articles were implemented. The libraries used were DJNativeSwing and DJNativeSwing-SWT. A class named SimpleWebBrowser was made to create the browser to display the article. This implementation also required us to download the SWT library from the eclipse website for our specific operating system, which essentially makes the program dependent on the operating system. Different archive files were made to support Windows, OSX, and Linux.

Testing and Results

To test the Sports Hub we kept a general rule that before another feature was added, each part of the existing application should function properly. The feature of the application that was tested the most was making sure that a majority of the application existed inside one frame and that when a button was pressed, the frame would switch the panel currently being displayed. This functionality was desired from the very beginning of the project process and it was very important to the group that most of the app not open too many different windows. For example, when a user selects one of the game buttons from the bottom of the homepage the current panel would switch to a game information panel drawn from the Game class. On the

game panel there is also a back button that then takes the user back to the applications main page. This functionality also exists when a user selects the "Teams" button from the main page, and then again when the user selects both an individual Team and then a member of that team. The "My Profile" button will also demonstrate similar behavior.

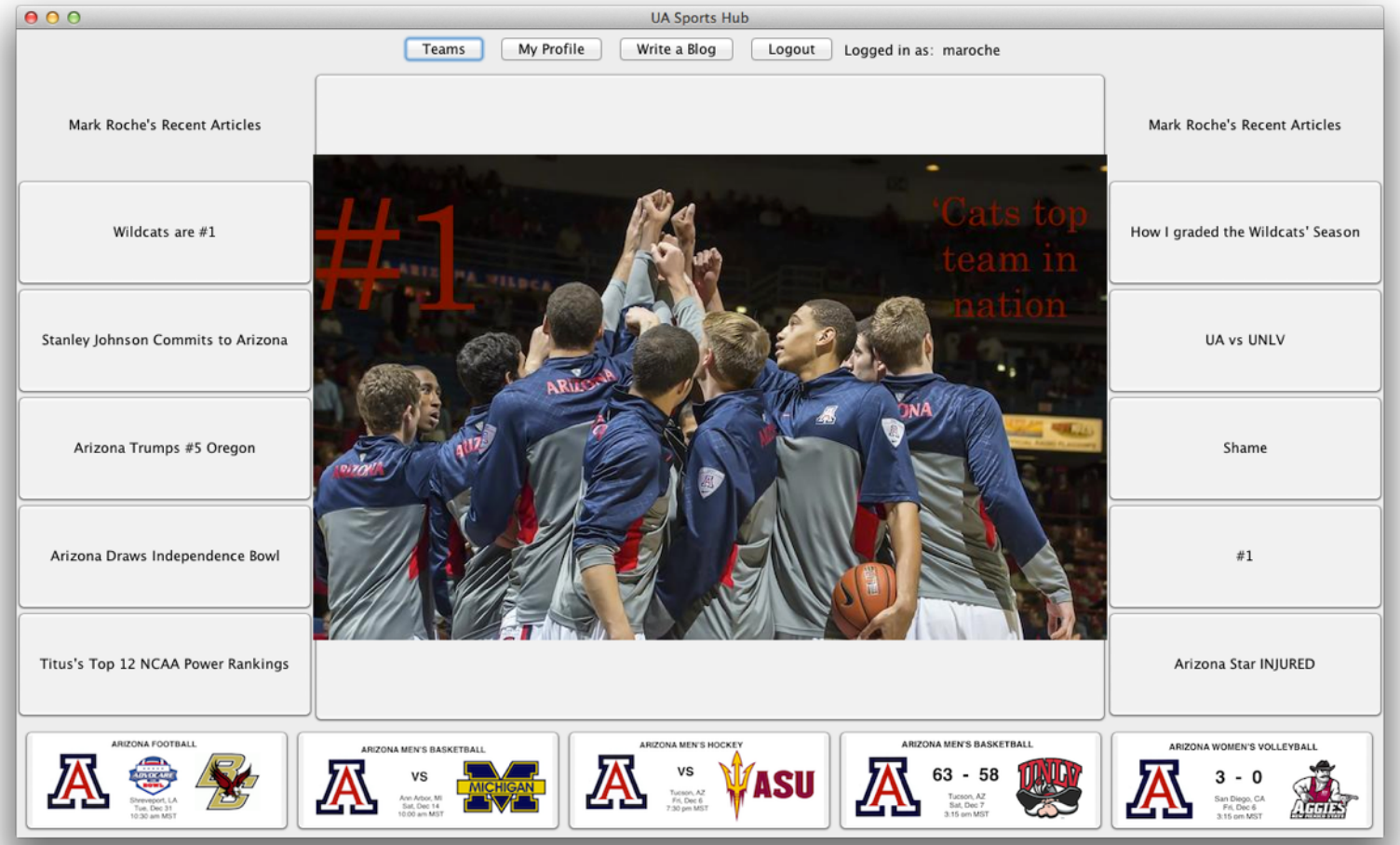


Figure 5 (Above): Here the main page is displayed with each of the buttons being displayed.

Figure 6 (Below): A game page is selected, changing the panel on the frame. The back button will take the user back to the main page.

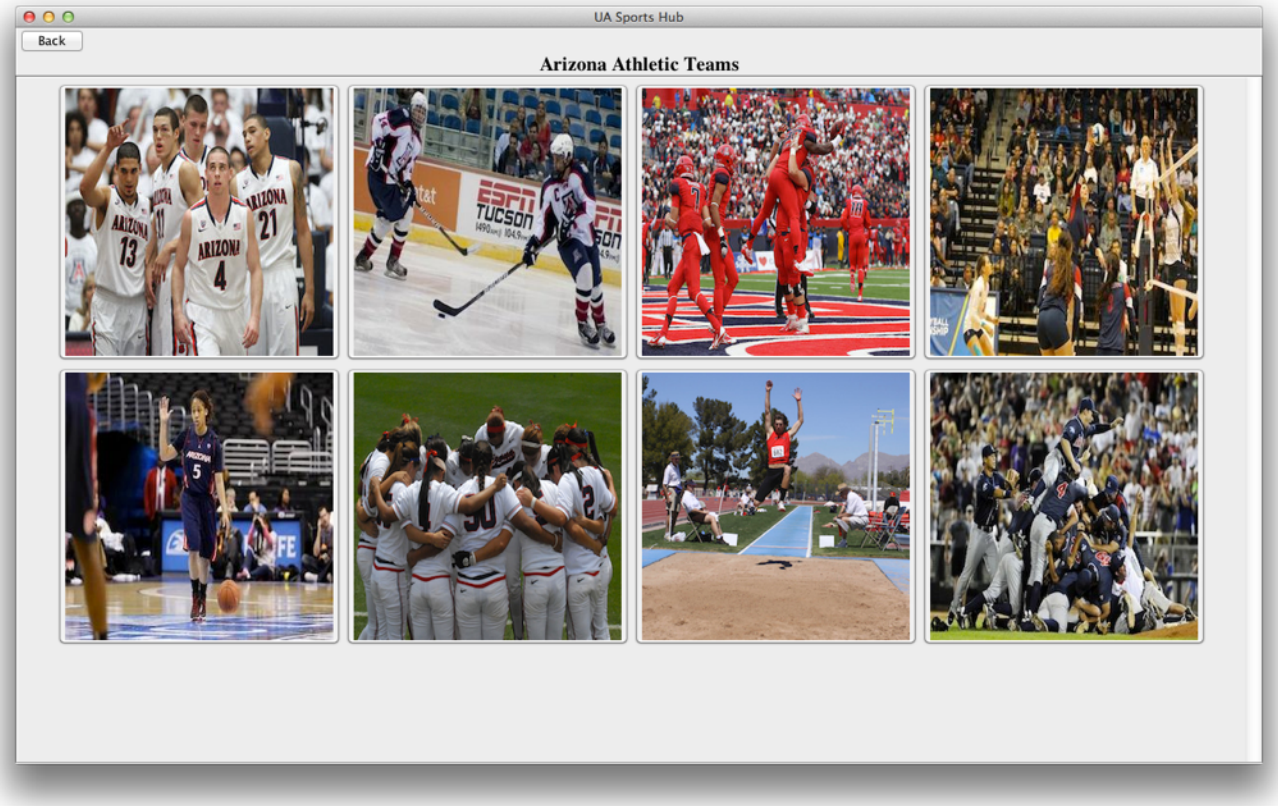
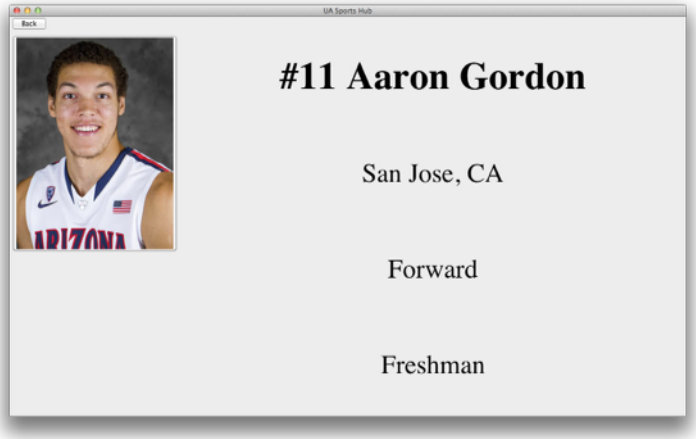


Figure 7 (Above): The user has selected the teams page from the main page. The user can then select a team (current only basketball) or go back to the main panel.

Figure 8 (Below): The user has selected the basketball team, and can then look at an individual's information (currently only Aaron Gordon).



Some features were not able to obey this principle and to work around them we created a way for the user to keep their attention on a a single frame at a time. When a user selects a blog

page, a new small frame will open with the blog that the user selected. Should a user select another blog, the frame that opened will simply be disposed and replaced by the new frame.

Another area that was tested was the ability to navigate through blog posts by pressing the "Previous Blog" and "Next Blog" buttons in the blog interface. A main reason that the blogs were implemented in a new frame is due to when opened in a new panel on the existing frame, the previous and next blog functionality no longer worked properly. A final function that was tested with the blogs themselves was the function of having a new blog appear in the main page after the user had written the blog. Dynamically changing the homepage proved to be tricky however we were able to implement it.

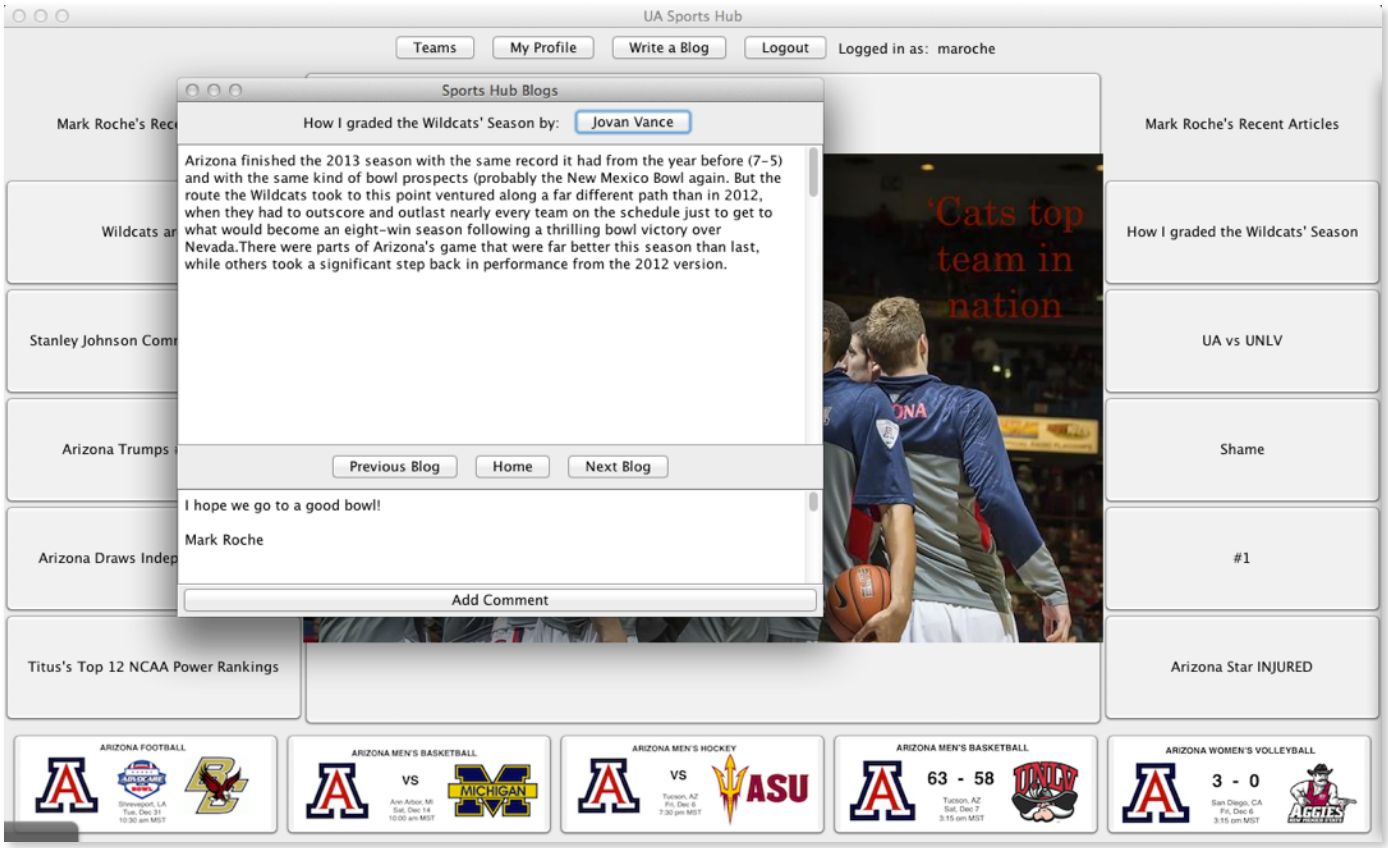


Figure 9: A blog is being displayed.

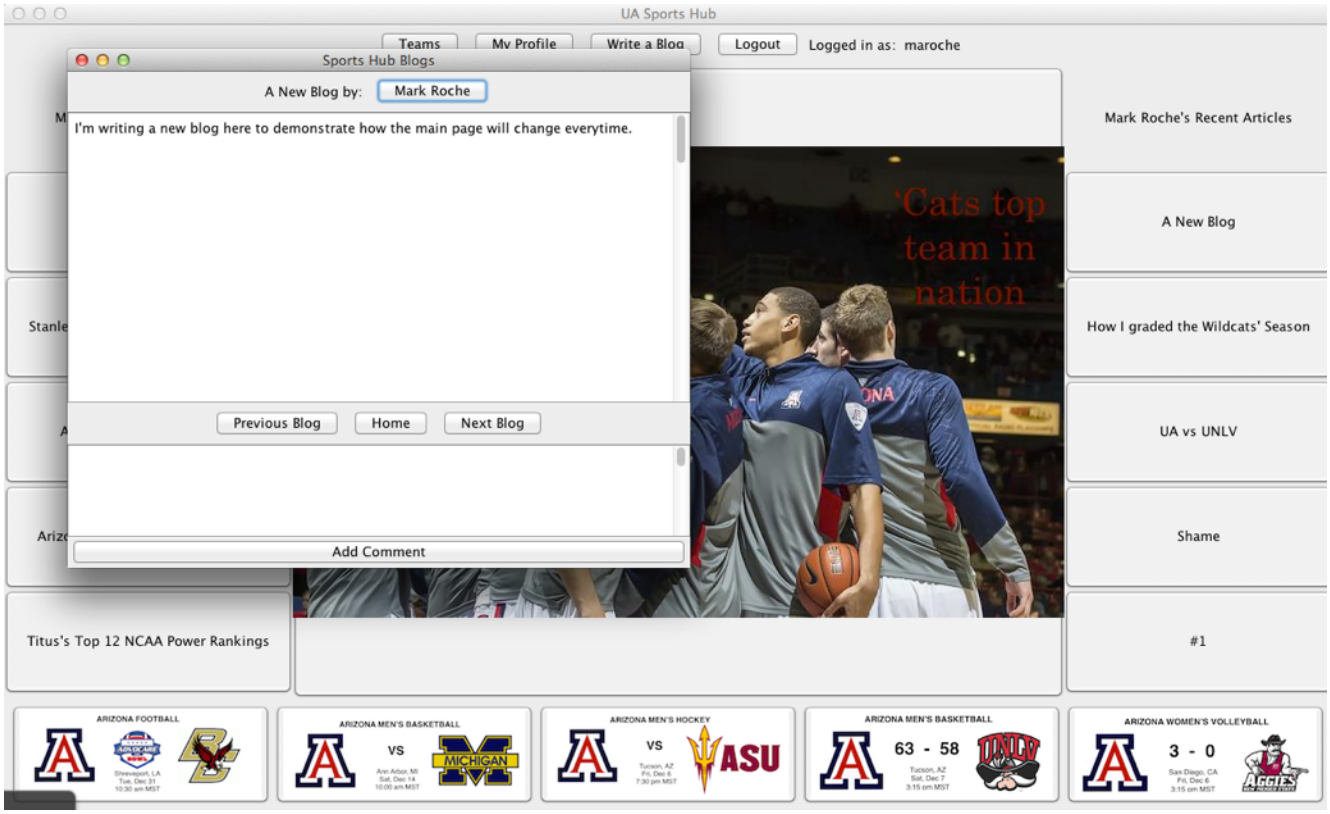
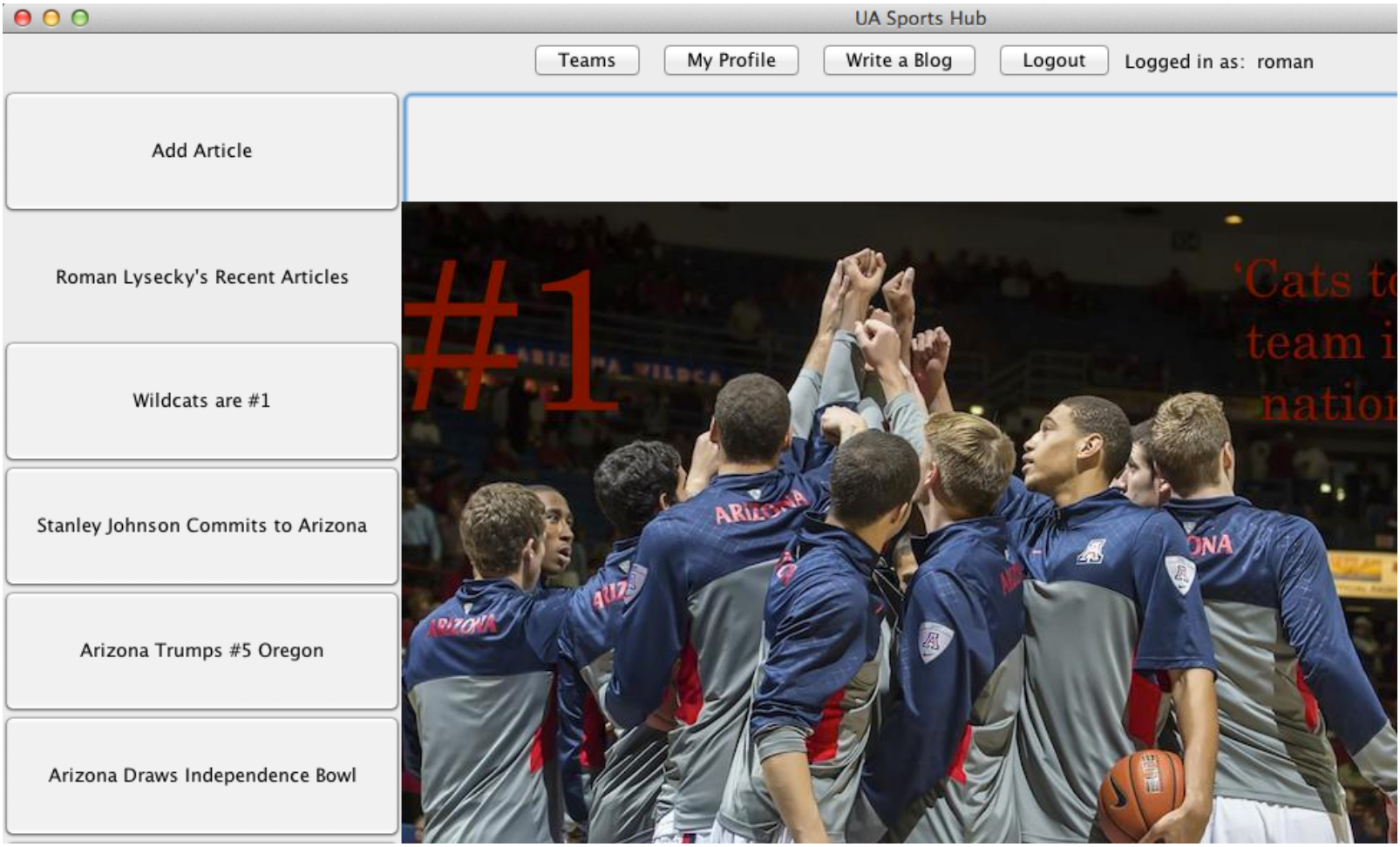


Figure 10: A new blog is added, and the main page has bee updated accordingly.

Another area where blogs were tested was in regard to adding comments to blogs. Each blog had its own ArrayList of comments that a user could add to. To test this we added comments to the blog and would log out of the application, log back in as another user, and then verify that the comment added was still there and this functionality worked.

The article was the next tested aspect of the project as this feature opened web articles in a new frame that behaved similarly to a web browser. When a user selected an article, the app would open a new frame and link the user to that specified article. The articles were also able to be dynamically changed on the home screen similar to how the blogs behave. However, only admins are able to add new articles to the application.

Figure 11 (Below): An admin user can add articles to the main page.



The article feature when tested would actually crash the application from time to time if the user opened too many articles. If the application was run for the first time as well, sometimes the articles would not work, the application would simply need to be refreshed and it would function properly.

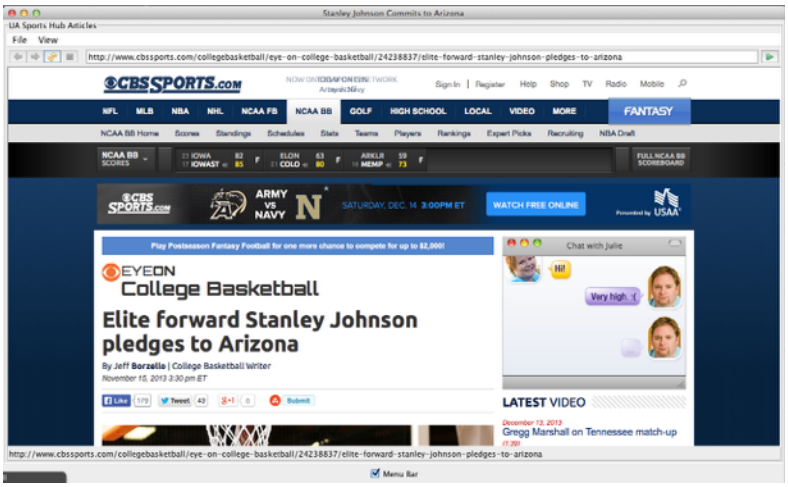


Figure 12: An article being opened by the application

The final area of testing was the actual account creation. When a user would create an account, the application would not allow the user to finish the process until all of the necessary information was entered. After accounts had been created, we were able to test them by logging in with the new user's username and password.

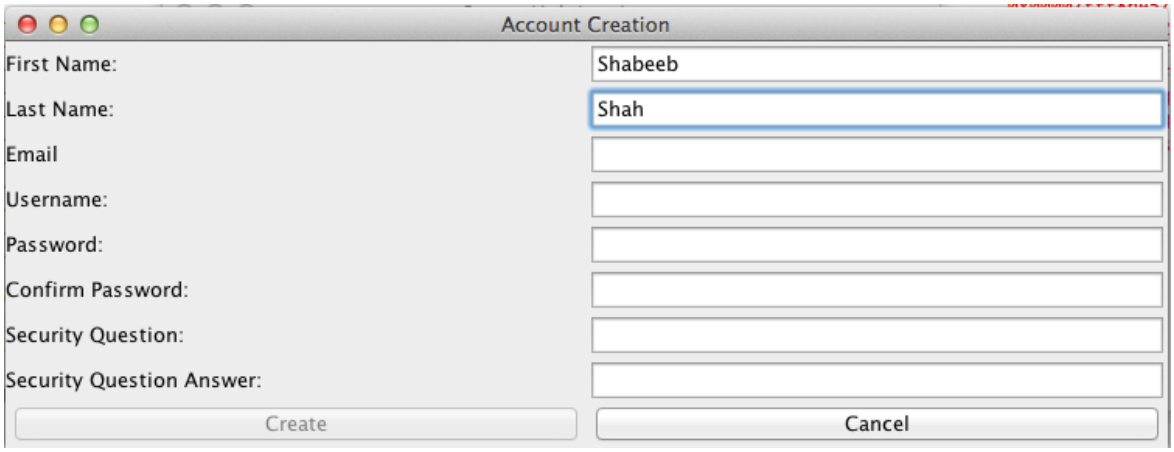


Figure 13: The create field has been disabled because the user has not entered all of the proper information.

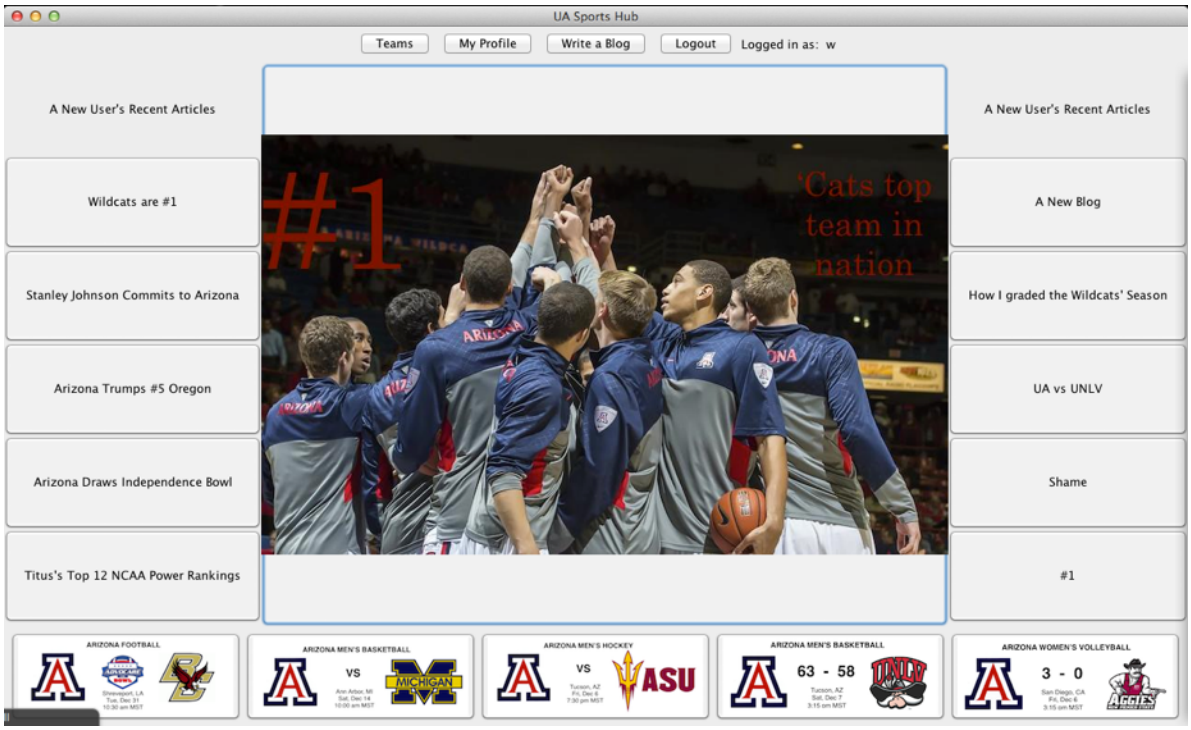


Figure 14: A new user's login screen

Extensibility

As previously mentioned the entire app is not a completely finished product as it would take an immense amount of time to create individual pages for each individual athlete on each team. There does exist a few venues of extensibility we could implement if given some more time. The first one for example would be to have each individual page working for each athlete on each team. In the account creation system, we also could add few more fields so that a new user's profile could have all of the information necessary to display it on that user's profile. Another place where the project could be improved would be to have articles, and player statistics be pulled from the internet so that the application would be updated with new and relevant information for all of the users. A great feature would be to have live game statistics and game information in real time so users can check the scores of their favorite University of Arizona teams on the application. The live game statistics could also pull information from the Sport and Venue classes so the user can learn more about the specific venue and sport currently being viewed. Another place that could be improved would be to have a networking functionality so that multiple users could be logged in at a time from different locations and interact with each other on the app much like many of the social networks people use today.