CSCI5448 – Object Oriented Analysis and Design
Class Project - Part #5

Team Members:
- Chris Pillion
- John Zavidniak
- Akshat Kambli

Title:

Entertainment and Sports Program – Next Generation (ESP-NGen)

Project Summary:

This effort will produce an interactive sports tracking program that allows users to obtain news and information about their favorite sports, leagues, teams, and athletes. Users will be able to create an account, access different aspects of sports information and news, save preferences, and receive customer support. System Admins will be responsible for keeping the statistical information up to date and providing support to other application users. Lastly, external journalists will be allowed to create an account as a reporter, submit stories and articles for approval, and receive support, if needed. The primary objective is to create a single sports related resource for the sports enthusiast application user.

Project Evaluation:

1. **List the features that were implemented (table with ID and title).**

   Requirements Met:
   The project requirements were broken into the four following categories:
   Business Requirements, User Requirements, Functional Requirements, and **Non-functional Requirements**
   Below are the features from Part II that were implemented in the final version of the project.

Business Requirements				
ID	**Requirement**	**Topic Area**	**User**	**Priority**
BR-001	Users can only create an account with a verified email address	Authentication	- Primary - Journalist	High
BR-002	Users can only create at most one primary user account and one journalist account per email address	Authentication	- Primary - Journalist	High
BR-003	Admins must approve all externally submitted news articles and information	Updates	Admin	Med

User Requirements				
ID	**Requirement**	**Topic Area**	**User**	**Priority**
UR-001	Primary user shall be able to view sports content related (but not limited) to league, team, and player information	User Content	Primary	Critical
UR-002	Admin shall be able to approve and update submitted news stories from the Journalist	User Content	Primary	Critical
UR-003	Primary user shall be able to access preferences, such as favorites, that will be saved to their account	Preferences	Primary	Low
UR-004	Primary user and journalists shall be able to delete their account at any time	Preferences	- Primary - Journalist	Med
UR-005	Primary user and journalists shall be able to log into and out of their account	Authentication	- Primary - Journalist	High
UR-006	Journalists shall be able to submit news articles to be posted through the application	User Content	Journalist	High
UR-007	Primary users shall be able to view news articles and updates	User Content	Primary	High
UR-008	Primary users and journalists shall be able to receive customer support from a system admin	Support	- Primary - Journalist - Admin	Med

Functional Requirements				
ID	Requirement	Topic Area	User	Priority
FR-01	The System must verify that the username is unique and password meets the set of guidelines at the time of registration	Authentication	-Primary -Journalist	High
FR-02	The System should differentiate between a primary user and a journalist	Authentication	-Primary -Journalist	High
FR-04	Registration will involve two factor authentication through E-mail	Authentication	-Primary -Journalist	High

Non-Functional Requirements				
ID	**Requirement**	**Topic Area**	**User**	**Priority**
NFR-01	Application should work on adequate internet connection	Performance	-Primary -Journalist	High
NFR-02	Application should respond to user action within 2 seconds	Usability	-Primary -Journalist	Medium

Use Cases Met:

ID	Use Case Name
UC-01	View News
UC-02	View League Info
UC-03	View Team Info
UC-04	Submit News
UC-06	Select Favorites
UC-07	Delete Account
UC-08	Receive Customer Support
UC-09	Logout of Account
UC-10	Journalist Login
UC-11	Create Journalist Account
UC-12	Create User Account
UC-13	Verify Email
UC-14	Verify Profession

2. **List the features were not implemented from Part 2 (table with ID and title).**

Requirements Not Met:
Below are the remaining requirements that were not implemented or verified for the final application.

Business Requirements				
ID	**Requirement**	**Topic Area**	**User**	**Priority**
BR-004	Journalists must create an account with an email from a verified news source domain name	Authentication	Journalist	High

Functional Requirements				
ID	Requirement	Topic Area	User	Priority
FR-03	System should detect and display new users in real time	User content	-Primary	Medium

Non-Functional Requirements				
ID	**Requirement**	**Topic Area**	**User**	**Priority**
NFR-03	Application should handle at-least 5 users at a time	Scalability	-Primary	High

Use Cases Not Met:

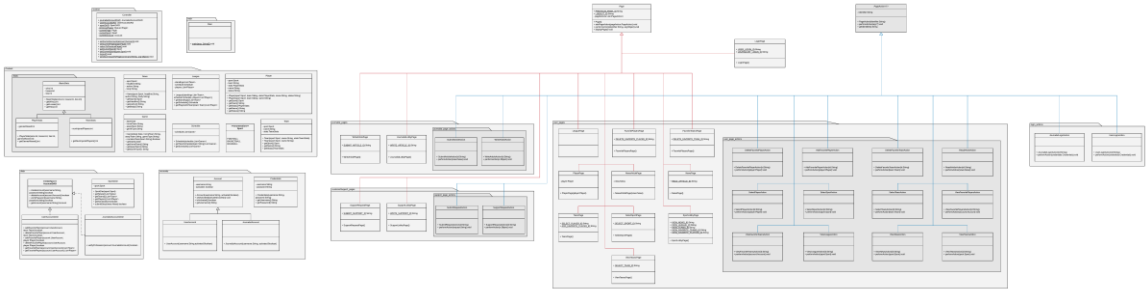
ID	Use Case Name
UC-05	Update News

Explanation:

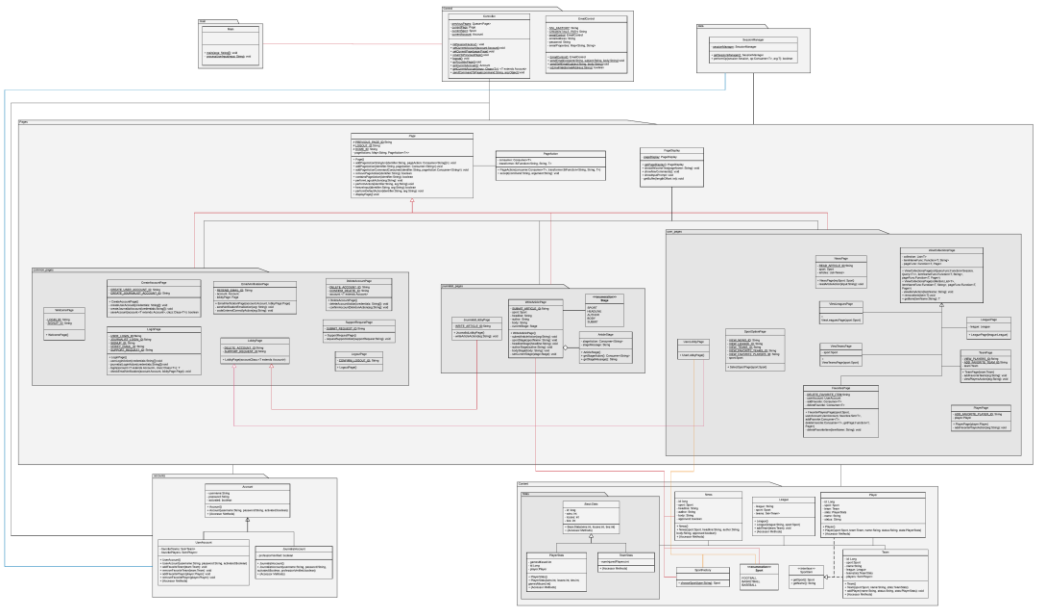
- For BR-004, Journalists still need to be approved, but they do not require a "verified news source domain name." The approval is done at the admins discretion, but the important point is that an approval system was implemented.
- FR-03 was not completed for small-scale single application testing. This would be created later when the application is closer to wide scale use.
- Again, NFR-03 is more for a larger scale on a shared server. This application currently runs on the local machine and local server, so testing of this requirement was not performed.
- UC-05 was not implemented because of a streamlined design that removed the grunt work for the admin. Once an admin approves a news story, the system automatically posts the news for viewers to see. There is no need for the admin to go in and manually add it to the webpage.

3. **Show your Part 2 class diagram and your final class diagram. What changed? Why? If it did not change much, then discuss how doing the design up front helped in the development.**

Part 2 Class Diagram:



Final Class Diagram:



- Our part 2 class diagram had considerable floating classes with missing relationships in between them. We also had created separate classes for performing single actions. These errors are rectified in the new class diagram by making it more organized with well established relationships between the classes.

- We have decided to use Model View Controller (MVC) pattern for designing the class diagram. The MVC pattern with the separation of pattern feature provided the multiple views of the same data at same time which led to faster implementation of our project.

- In our final class diagram, we have not made any changes in the primary functions of our project, however the internal design of classes is slightly changed.

4. Did you make use of any design patterns in the implementation of your final prototype? If so, how? Show the classes from your class diagram that implements each design pattern (each design pattern as a separate image in the .PDF). If not, where could you make use of design patterns in your system? Show a class diagram of how you could implement each design pattern and compare how it would change from your current class diagram.

1. **State Design Pattern:**
   - Controller class has CurrentPage instance variable which is following state design pattern.
   - CurrentPage is the instance variable of the class which stores the current page of the application being displayed.
   - We change the state of the application by calling the setcurrentPage(page Page) which changes the current page of the application.

2. **MVC Design Pattern:**
   - We are following MVC design pattern in our project by having Controller class as a controller of classes.
   - All the subclasses of Page viz. CreateAccountPage, LoginPage, LogoutPage, WelcomePage are defined as views and League, News, Team etc. as models. Basically, that enabled separation of concerns in our project.
   - The database interaction of our project is handled by the model classes which are completely independent of the view and controller classes.

3. **Factory Design Pattern**
   - SportsFactory class is following factory design pattern. The class has a method chooseSport(String type) which expects a parameter sportsType, and returns the appropriate Enum Type by comparing it various sports types.
   - This class is being used in WriteArticlePage().

4. **Strategy design pattern**
   - Our project makes use of the strategy design pattern through the Page abstract class, and the PageAction class.

- The Controller class has a Page instance variable which refers to various page of the application. It allows us to dynamically change the page of the class.
- Every Page has a set of PageActions that define the Page's behavior. These actions can by dynamically added and removed by the Page at runtime.

5. What have you learned about the process of analysis and design now that you have stepped through the process to create, design and implement a system?

- We have learnt about the software development cycle through the practical implementation of our project. Building up the project from the scratch is a very important task while simultaneously following the deadline and keeping in mind the dynamic nature of the process and requirements. It is very essential for the team members to work on their individual parts while keeping entire project in consideration and the mutual compatibility of their modules as well.

- Object-oriented approach is very helpful for us in determining following ways such as:

1. Reusability of programming and models
2. System understanding
3. Reduced development risks
4. Flexible approach
5. Reduced cost with any changes

- The second part of the project in which we have developed activity diagrams, user requirements, case diagram and sequence diagrams made us to understand and determine the flow and structure of the project.

- Additionally, after the practical implementation of the requirements, we could learn to detect the areas having smelly codes and corrected them with various refactoring methods taught in the class.