

Project Report

Online Prescription Systems for Doctor.

By

Md. Mineuddin Ahmed Dipu

Professional Master’s in computer Science

Department of Computer Science and Engineering

October 2022

Declaration:

Acknowledgment:

I would like to give special thanks to following people who have made the completion of this project possible:

My parents, for their continued support and generous funding.

Dr. Md. Ezharul Islam (My supervisor), for giving me the opportunity to undertake this project with her as well as for his support and guidance.

Md. Rahul (My friend), for his unyielding knowledge on anything computer or math related and his willingness to always help.

Arob Hossain (Younger Brother), for his unrested help to make the error vanished and also to the coding.

Abstract:

This document is the final report for the project online prescription system for doctors. The key objective of this project is to provide a high tech online prescription with full of medicine list, investigation list and patient/doctor satisfaction. We will release the newer version day by day and always try to make it smooth, easy and fast loader.

Besides we will create more opportunities for new doctors so they can connect, share and learn more from our prescription.

Table of Contents

1. Introduction
   1. Purpose ………………………………………………………..9
   2. Scope of project ……………………………………………….9
   3. Existing System ……………………………………………….10
   4. Proposed System ………………………………………………10
2. Software Requirement Specification (SRS)
   1. System Environment……………………………………………11
   2. Functional Requirement Specification …………………………12
      1. User Characteristics………………………………………12
   3. Non-Functional Requirements…………………………………12
   4. Detailed Requirement Specification……………………………13
      1. Functional Requirement………………………………….13
      2. Non-functional Requirement……………………………..19
   5. Logical Structure of the Data……………………………………19
   6. Security………………………………………………………….23
3. Design
   1. User Interface Design……………………………………………24
      1. Introduction………………………………………………..24
      2. Interface Analysis………………………………………….24
      3. User Analysis………………………………………………24
      4. Task Analysis………………………………………………26
   2. UML
      1. Use-case Diagram………………………………………….27
      2. Activity Diagram…………………………………………..28
   3. System Design Environment…………………………………….29
      1. Frontend Interface Design …………………………………….29
      2. Backend Design……………………………………………...29
      3. Frontend and Backend Programming…………………………..29
      4. Backend Database……………………………………………29
   4. System Running Environment…………………………………29
   5. Interface Design Steps…………………………………………29
4. Implementation and Deployment
   1. Define interface objects and actions…………………………..42
5. Testing
   1. Overview………………………………………………………47
   2. Scope…………………………………………………………..47
   3. Appendix………………………………………………………47
6. Conclusion…………………………………………………………51

List of figures

1. Figure 2.1 System Environment……………………………………11
2. Figure 2.2 Logical Structure of Article manager data……………...20
3. Figure 3.1 Use-case diagram……………………………………….27
4. Figure 3.2 Activity Diagram………………………………………..28
5. Figure 3.3 Menu- Home……………………………………………30
6. Figure 3.4 Menu- Football………………………………………….31
7. Figure 3.5 Menu- contact ………………………………………….32
8. Figure 3.6 Menu- Registration………………………………………33
9. Figure 3.7 Menu- your post…………………………………………34
10. Figure 3.8 Admin home…………………………………………….35
11. Figure 3.9 Admin Country insert……………………………………36
12. Figure 3.10 Admin City insert………………………………………37
13. Figure 3.11 Admin Registration…………………………………….38
14. Figure 3.12 Admin Category……………………………………….39
15. Figure 3.13 Admin News………………………………………….40
16. Figure 3.14 Admin Payment………………………………………41
17. Figure 4.1 Home page Interface……………………………………42
18. Figure 4.2 Contact page Interface………………………………….43
19. Figure 4.3 Registration page Interface………………………………44
20. Figure 4.4 Your Post page Interface…………………………………45
21. Figure 4.5 News page Interface………………………………………46

List of tables

1. Table 2.1 Search News………………………………………………13
2. Table 2.2 Communicate…………………………………………….14
3. Table 2.3 Add Author………………………………………………14
4. Table 2.4 Update Person…………………………………………….15
5. Table 2.5 Update News Status………………………………………15
6. Table 2.6 Enter communication…………………………………….16
7. Table 2.7 Check Status…………………………………………….17
8. Table 2.8 Send Communication……………………………………17
9. Table 2.9 Publish News …………………………………………….18
10. Table 2.10 Remove News……………………………………………19
11. Table 2.5.1 User Data Entity……………………………………………20
12. Table 2.5.2 News Entry………………………………………………21
13. Table 2.5.3 Category Entry……………………………………………21
14. Table 2.5.4 Country Entry…………………………………………….21
15. Table 2.5.5 City Entry…………………………………………………22
16. Table 2.5.6 Payment Entry…………………………………………….22
17. Table 2.5.7 Comment Entry…………………………………………….22
18. Table 5.3.1 User Login System…………………………………………47
19. Table 5.3.2 Category Insert…………………………………………….49
20. Table 5.3.3 Country Insert…………………………………………….49
21. Table 5.3.4 City Insert……………………………………………….50
22. Table 5.3.5 News Insert……………………………………………….50
23. Table 5.3.6 User Sign Up…………………………………………….50
24. Introduction
    1. Purpose:

The purpose of this document is to present a detailed description about online prescription project. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for both the stakeholders and the developers of the system.

* 1. Scope of Project:

By this project doctors and patient both can be benefited. Doctors can easily prescribe patient. They can keep record of their patient and patient also get email of the prescription. So they also can track their health history. They don’t need to preserve any piece of paper at their home. Moreover a piece of paper may destroy at any time. On the other hand if the patient will come after a long time then the doctor can easily track this patient previous record. So it will easy to find the previous health history of any patient. A doctor also can research on the data he has. He also can share his experience with other practitioners and medical student for their study purpose. Overall it can play an incredible role in our health care sector.

* 1. Existing System:

There are a few existing systems in our marketplace now. There are more then 7-8 prescription systems in Bangladesh. We have some added features what is going to amaze doctors to take our systems and also our system is not only design for doctors but also design for patient and organization like hospital and diagnostic center.

* 1. Proposed System:

The proposed system is designed for an online prescription system with doctor appointment and payment. Where patient can take serial of any doctor, confirm appointment through payment and doctor prescribe the patient through our prescription system. The patient will get the prescription through the email along with printed copy.

1. Software Requirement Specification (SRS)

2.1 System Environment

Admin

Doctor

Appointment

Prescription

Print

Payment

Figure 2.1: Online Prescription system Environment

The Admin accesses the Appointment and payment option. Doctor will create the prescription and print the prescription.

2.2 Functional Requirements Specification

This section outlines the use cases for each of the active doctors separately while the admin is main actor in this system.

2.2.1 User Characteristics

* The Reader is expected to be Internet literate and be able to use a search engine. The main screen of the Online Journal Website will have the search function and a link to “Author Information.”
* The Author is expected to be Internet literate and to be able to use email with attachments.
* The Editor is expected to be Windows literate and to be able to use button,

pull-down menus, and similar tools.

Moreover the specifications are….

Post Article, Update Article, Delete Article, Search Article, Add User, Check Status, Give Payment, User Data Entity, Category Entity, News Data Entity, Comment Data Entity, Payment Data Entity.

2.3 Non-Functional Requirements

The Online Magazine will be on a cloud server. We will use either google cloud or Amazon cloud. There will not be any physical server. So this will be very efficient and there won’t be any risk of server crash or something like this.

The Article Manager will run on the editor’s PC and will contain Mysql database. Mysql is already installed on this computer and is a Windows operating system.

Easy Load, Superfast, Communication, Good font, Cloud server, Concurrency, Real time feedback.

2.4 **Detailed** **Requirement Specification**

2.4.1 Functional Requirements

The Logical Structure of the Data is contained in Section .

2.1 Search News

|  |  |
| --- | --- |
| **Trigger** | The Reader assesses the Online sports news Website |
| **Precondition** | The Web is displayed with grids for searching |
| **Basic Path** | 1. The Reader chooses how to search the Web site. The choices are by Author, by Category, and by Keyword. 2. If the search is by Author, the system creates and presents an alphabetical list of all authors in the database. In the case of an article with multiple authors, each is contained in the list. 3. The Reader selects an author. 4. The system creates and presents a list of all articles by that author in the database. 5. The Reader selects an article. 6. The system displays the Abstract for the article. 7. The Reader selects to download the article or to return to the article list or to the previous list. |
| **Alternative Paths** | In step 2, if the Reader selects to search by category, the system creates and presents a list of all categories in the database.   1. The Reader selects a category. 2. The system creates and presents a list of all articles in that category in the database. Return to step 5.   In step 2, if the Reader selects to search by keyword, the system presents a dialog box to enter the keyword or phrase.   1. The Reader enters a keyword or phrase. 2. The system searches the Abstracts for all articles with that keyword or phrase and creates and presents a list of all such articles in the database. Return to step 5. |
| **Postcondition** | The selected article is downloaded to the client machine. |
| **Exception Paths** | The Reader may abandon the search at any time. |
| **Other** | The categories list is generated from the information provided when article are published and not predefined in the Online Journal database. |

2.2 Communicate

|  |  |
| --- | --- |
| **Trigger** | The user selects a *mailto* link. |
| **Precondition** | The user is on the *Communicate* page linked from the Online Journal Main Page. |
| **Basic Path** | This use case uses the *mailto* HTML tag. This invokes the client email facility. |
| **Alternative Paths** | If the user prefers to use his or her own email directly, sufficient information will be contained on the Web page to do so. |
| **Postcondition** | The message is sent. |
| **Exception Paths** | The attempt may be abandoned at any time. |
| **Other** | None |

2.3 Add Author

|  |  |
| --- | --- |
| **Trigger** | The Editor selects to add a new author to the database. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The system presents a blank grid to enter the author information. 2. The Editor enters the information and submits the form. 3. The system checks that the name and email address fields are not blank and updates the database. |
| **Alternative Paths** | If in step 2, either field is blank, the Editor is instructed to add an entry. No validation for correctness is made. |
| **Postcondition** | The Author has been added to the database. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | The author information includes the name mailing address and email address. |

2.4 Update Person

|  |  |
| --- | --- |
| **Trigger** | The Editor selects to update a user to author and the person is already in the database. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The Editor selects Author. 2. The system creates and presents an alphabetical list of people in the category. 3. The Editor selects a person to update. 4. The system presents the database information in grid form for modification. 5. The Editor updates the information and submits the form. 6. The system checks that required fields are not blank. |

2.5 Update News Status

|  |  |
| --- | --- |
| **Trigger** | The Editor selects to update the status of an article in the database. |
| **Precondition** | The Editor has accessed the Article Manager main screen and the article is already in the database. |
| **Basic Path** | 1. The system creates and presents an alphabetical list of all active articles. 2. The Editor selects the article to update. 3. The system presents the information about the article in grid format. 4. The Editor updates the information and resubmits the form. |
| **Alternative Paths** | In step 4, the use case *Enter Communication* may be invoked. |
| **Postcondition** | The database has been updated. |
| **Exception Paths** | If the article is not already in the database, the use case is abandoned. In addition, the Editor may abandon the operation at any time. |
| **Other** | This use case can be used to add categories for an article, to correct typographical errors, or to remove a reviewer who has missed a deadline for returning a review. It may also be used to allow access to the named use case to enter an updated article or a review for an article. |

2.6 Enter Communication

|  |  |
| --- | --- |
| **Trigger** | The Editor selects to add a document to the system. |
| **Precondition** | The Editor has accessed the Article Manager main screen and has the file of the item to be entered available. |
| **Basic Path** | 1. The Editor selects the article using the *3.2.6, Update Article Status* use case. 2. The Editor attaches the file to the grid presented and updates the respective information about the article. 3. When the Editor updates the article status to indicate that a review is returned, the respective entry in the Reviewer table is updated. |
| **Alternative Paths** | None |
| **Postcondition** | The article entry is updated in the database. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | This use case extends *3.2.6, Update Article Status* |

2.7 Check Status

|  |  |
| --- | --- |
| **Trigger** | The Editor has selected to check status of all active articles. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The system creates and presents a list of all active articles organized by their status. 2. The Editor may request to see the full information about an article. |
| **Alternative Paths** | None. |
| **Postcondition** | The requested information has been displayed. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | The editor may provide an enhanced list of status later. At present, the following categories must be provided:   1. Received but no further action taken 2. Reviewers have been assigned but not all reviews are returned (include dates that reviewers were assigned and order by this criterion). 3. Reviews returned but no further action taken. 4. Recommendations for revision sent to Author but no response as of yet. 5. Author has revised article but no action has been taken. 6. Article has been accepted and copyright form has been sent. 7. Copyright form has been returned but article is not yet published.   A published article is automatically removed from the active article list. |

2.8 Send Communication

|  |  |
| --- | --- |
| **Trigger** | The editor selects to send a communication to an author. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The system presents an alphabetical list of authors. 2. The Editor selects an author. 3. The system invokes the Editor’s email system entering the author’s email address into the *To:* entry. 4. The Editor uses the email facility. |
| **Alternative Paths** | None. |
| **Postcondition** | The communication has been sent. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | The standard copyright form will be available in the Editor’s directory for attaching to the email message, if desired. |

2.9 Publish News

|  |  |
| --- | --- |
| **Use Case Name** | Publish Article |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The system creates and presents an alphabetical list of the active articles that are flagged as having their copyright form returned. 2. The Editor selects an article to publish. 3. The system accesses the Online Database and transfers the article and its accompanying information to the Online Journal database. 4. The article is removed from the active article database. |
| **Alternative Paths** | None. |
| **Postcondition** | The article is properly transferred. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | Find out from the Editor to see if the article information should be archived somewhere. |

2.10 Remove News

|  |  |
| --- | --- |
| **Trigger** | The Editor selects to remove an article from the active article database. |
| **Precondition** | The Editor has accessed the Article Manager main screen. |
| **Basic Path** | 1. The system provides an alphabetized list of all active articles. 2. The editor selects an article. 3. The system displays the information about the article and requires that the Editor confirm the deletion. 4. The Editor confirms the deletion. |
| **Alternative Paths** | None. |
| **Postcondition** | The article is removed from the database. |
| **Exception Paths** | The Editor may abandon the operation at any time. |
| **Other** | Find out from the Editor to see if the article and its information information should be archived somewhere. |

* + 1. Non-Functional Requirements

2.5 Logical Structure of the Data

The logical structure of the data to be stored in the internal Article Manager database is given below.

Review

Reviewer/Admin

Article

Author

writes

sent to

writes

has

Figure .2 - Logical Structure of the Article Manager Data

The data descriptions of each of these data entities is as follows:

* + 1. **User Data Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Type** | **Description** | **Comment** |
| Id | Integer | User Id |  |
| Name | varchar | User Name | Single reviewer |
| Email Address | varchar | User Email |  |
| User Type | varchar | Access level |  |
| Password | varchar |  |  |
| Join Date | datetime | Joining date |  |
| City Id | Integer |  |  |
| Gender | Char |  |  |

* + 1. **News Data Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Type** | **Description** | **Comment** |
| Id | Integer | Name of Article |  |
| Title | varchar | Author entity | Name of principle author |
| Summery | varchar | Other authors is any; else null | Not a pointer to an Author entity |
| Description | varchar | Reviewer entity | Will be several |
| Status | Integer | Review entity | Set up when reviewer is set up |
| Admin Id, Category Id, User Id | Integer | Body of article | Contains Abstract as first paragraph. |

* + 1. **Category Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Type** | **Description** | **Comment** |
| Id | Integer | Name of Article |  |
| Name | varchar | Name of one Author | May be several |
| Description | varchar | Abstract of article | Used for keyword search |

* + 1. **Country Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Type** | **Description** | **Comment** |
| Id | Integer |  |  |
| Name | varchar | Name of the country | May be several |
| Description | varchar | Abstract of article | Used for keyword search |

* + 1. **City Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Type** | **Description** | **Comment** |
| Id | Integer | Name of Article |  |
| Name | varchar | Name of one Author | May be several |
| Country Id | varchar | Abstract of article | Used for keyword search |

* + 1. **Payment Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Type** | **Description** | **Comment** |
| Id | Integer | Name of Article |  |
| User Id | Integer | Name of one Author | May be several |
| Datetime | dstetime | Abstract of article | Used for keyword search |
| Purpose | varchar |  |  |

**2.5.7 Comment Entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Item** | **Type** | **Description** | **Comment** |
| Id | Integer | Name of Article |  |
| User Id | Integer | Name of one Author | May be several |
| Datetime | dstetime | Abstract of article | Used for keyword search |
| News id | Integer |  |  |
| Publish status | Integer |  |  |

* 1. Security

The server on which the online sports Magazine resides will have its own security to prevent unauthorized *write*/*delete* access. There is no restriction on *read* access. The use of email by an Author or Reviewer is on the client systems and thus is external to the system.

The PC on which the Article Manager resides will have its own security. Only the Editor will have physical access to the machine and the program on it. There is no special protection built into this system other than to provide the editor with *write* access to the Online Journal to publish an article.

1. Design

3.1 User Interface Design:

3.1.1 Introduction

User interface design creates an effective communication medium between a human and a computer. The interface has to be right because it models a user’s perception of the software. As we know that a key tenet of all software engineering process models is “understand the problem before you attempt to design a solution”, we analysis the interface before starting the design steps.

3.1.2 Interface Analysis

We divide interface analysis into following parts:

i. User Analysis

ii. Task Analysis

3.1.3 User Analysis

In this part we follow two steps:

a. Identify user

b. Know user

**Identify user**

From the requirements specification we have identified following four user categories.

1. Authorized User

2. User

3. Reader

4. Admin

**Know user**

We collect following information about the users.

**Authorized User**

Age: Any

Work type: Writer

Skills: Average

Domain expert: Yes

Office hour: Any time

Frequency of use: Very frequently

Consequence of a mistake: High

General computer experience: Yes

**User**

Age: Any

Skills: Average

Frequency of use: Occasionally

Consequence of a mistake: Low

General computer experience: Yes

**Reader**

Age: Any

Frequency of use: Occasionally

Consequence of a mistake: Low

General computer experience: Yes

3.1.3 Task Analysis

In this step we identify and analyze the tasks of every users separately.

*Authorized User*: He/She has following tasks.

1. Issue

Goal: Issue the requested item

Precondition:

 User must be eligible for posting news.

Sub-task:

i. Check user comment

ii. Update new news

*User and Reader:*

1. Search

Goal: Search any news.

2. Comment

Goal: comment each news.

Precondition:

 Logged in as valid user for user.

Sub-task: Logged in

Admin:

1. Create Category for new news.

Goal: post news category wise.

Precondition:

 Valid category.

Sub-task:

 Search category

3. Change user type

Goal: Change the user type

Precondition:

 Valid User

Sub-task:

 Update user status.

* 1. UML 🡪Activity Diagram

3.2.1 Use case Diagram:

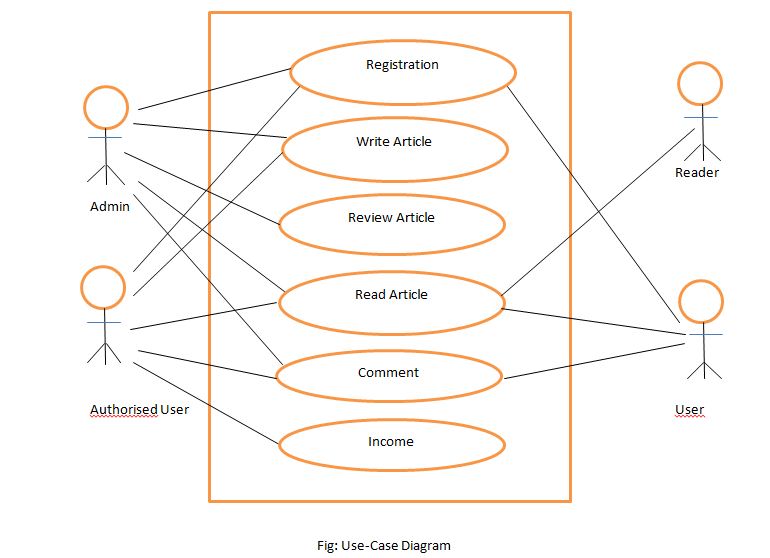


Figure 3.1: use-case diagram

3.2.2 Activity Diagram:

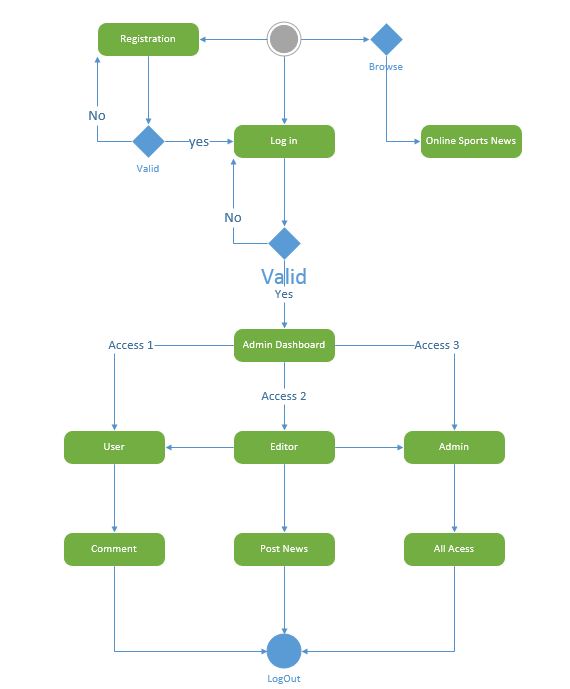


Figure 3.2: Activity Diagram

3.3 System Design Environment

3.3.1 Frontend Interface Design:

* HTML5.0, CSS3, Javascript and Jquery,Bootstrap.

3.3.2 Backend Design:

* HTML5.0, Bootstrap and CSS3

3.3.3 Frontend and Backend Programming:

* Php Language(OOP) using Model View Controller platform.

3.3.4 Backend Database

* Mysql database.
  1. System Running Environment

This system is tested and running in windows operating system. This is completely ok for Mozilla firefox, google chrome, opera, safari and also definitely internet explorer operating system.

3.5 Interface Design Steps

We follow the following steps to design the Online sports newspaper’s user interface.

i. Define interface objects and actions

ii. Define events that will cause the state of the user interface to change

3.5.1 Define interface objects and actions

We identified following objects and actions for the user interface. This is the prototype for the system.

3.5.1.1 Home (Menu)

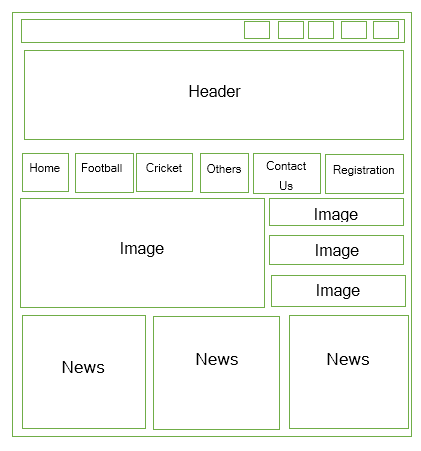


Figure 3.3: UI home design

3.5.1.2 Football (Menu)

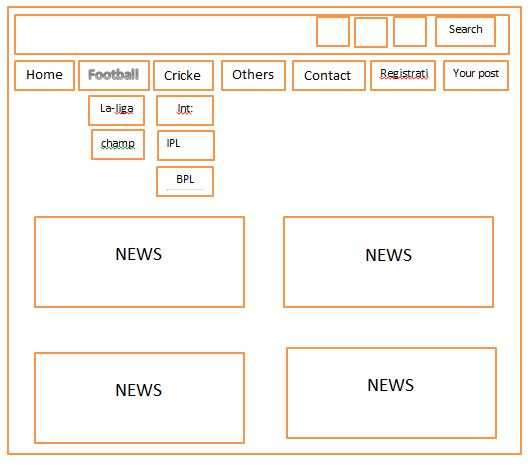


Figure 3.4: UI football page design

3.5.1.3 Contact Page

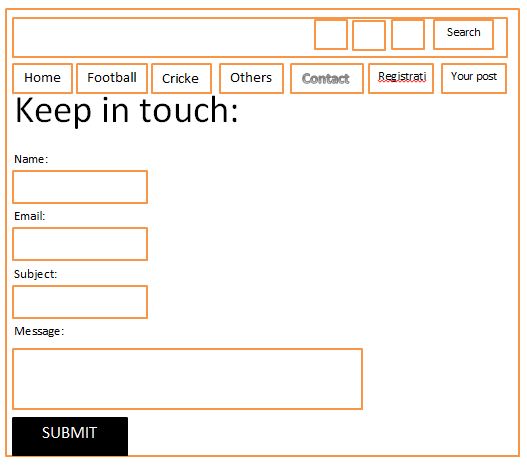


Figure 3.5: UI contact page design

3.5.1.4 Registration Page:

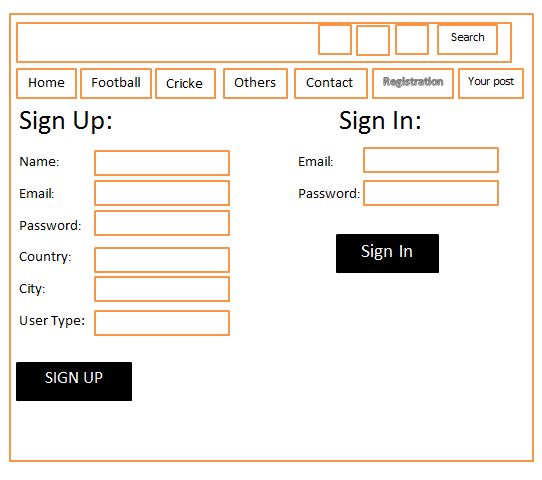


Figure 3.6: UI registration page design

3.5.1.5 News Post Page:



Figure 3.7: UI user post page design

3.5.2 Internal/Dashboard Interface Prototype:

3.5.2.1 Dashboard**:**



Figure 3.8: Admin Dashboard home design

3.5.2.3 Add Country Page:



Figure 3.9: Admin Dashboard country insert page design

3.5.2.4 Add City Page:



Figure 3.10: Admin Dashboard city add page design

3.5.2.5 Add User Page:

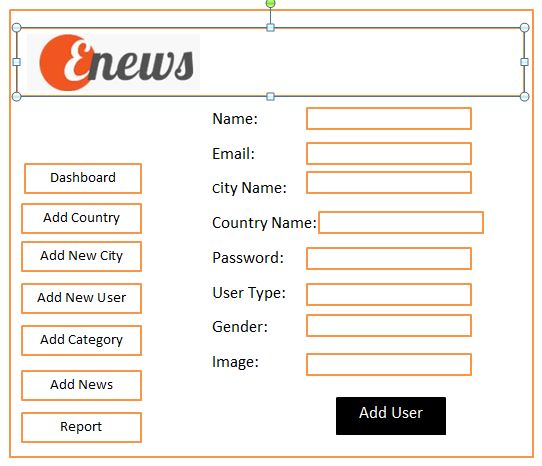


Figure 3.11: Admin Dashboard user registration design

3.5.2.6 Category add Page:

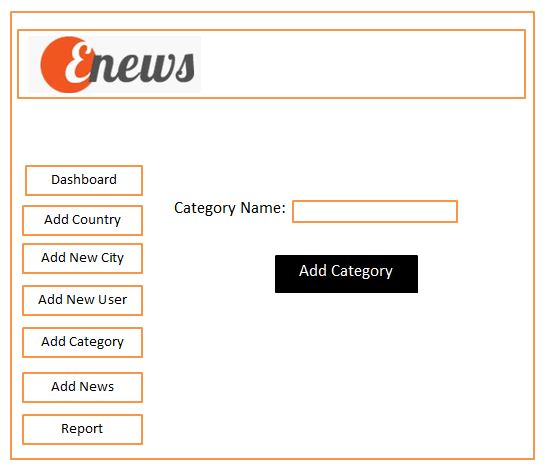


Figure 3.12: Admin Dashboard category add page design

3.5.2.7 News add Page:



Figure 3.13: Admin Dashboard news add page design

3.5.2.8 Add Payment Page:



Figure 3.14: Admin Dashboard payment add design

1. Implementation & Deployment

Depict each interface state as it look to end user.

4.1 Define interface objects and actions

We identified following objects and actions for the user interface. This is not the prototype for the system. This is the real interface what the end user will see.

4.1.1 **Home:**

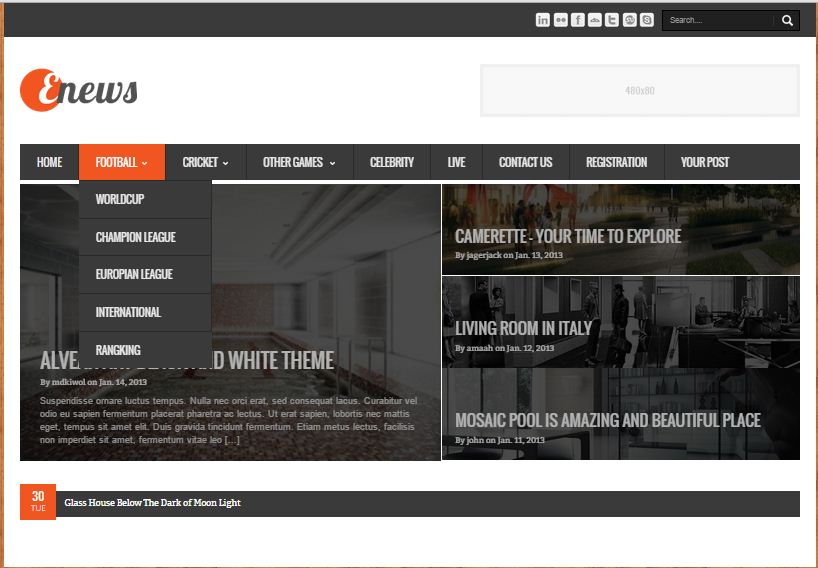
****

Figure 4.1: Home page interface for end user

4.1.2Contact Page:

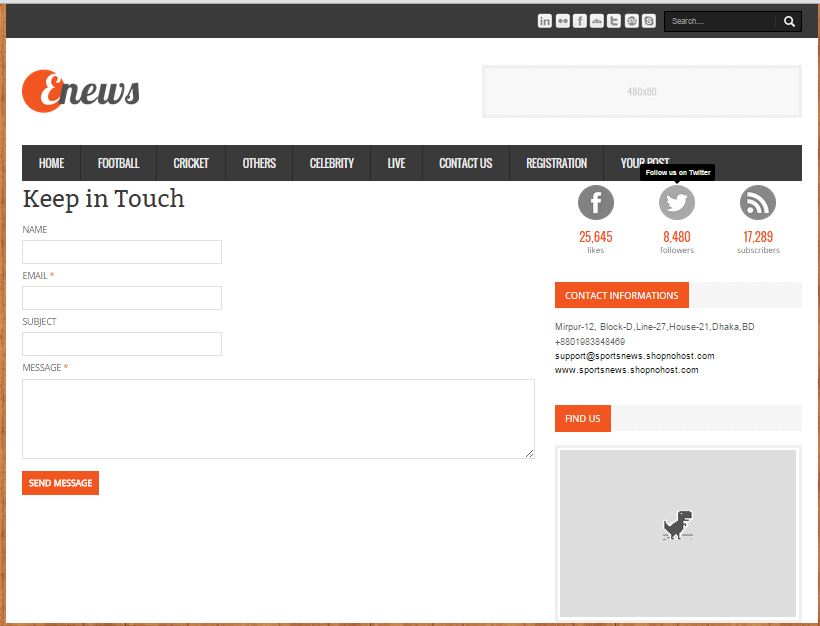
****

Figure 4.2: Contact page interface for end user

4.1.3 Registration:

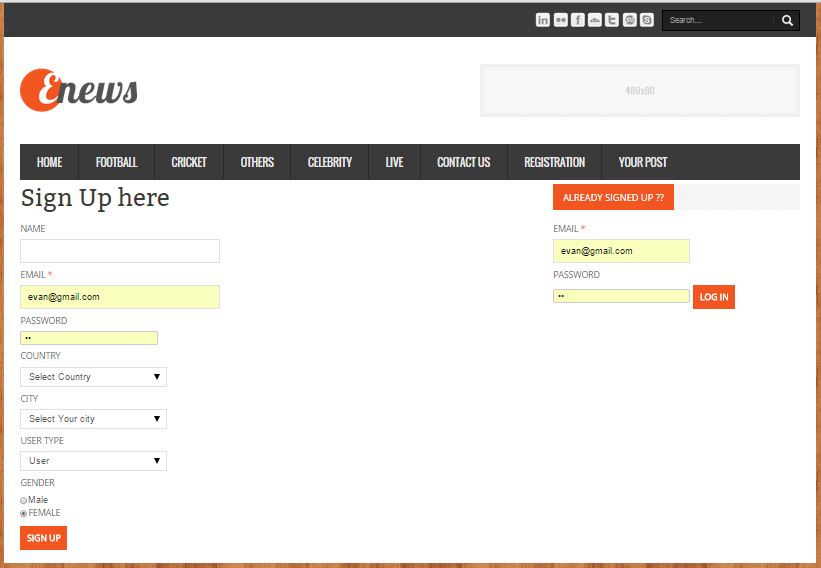


Figure 4.3: Registration page interface for end user

4.1.4 You’r Post:

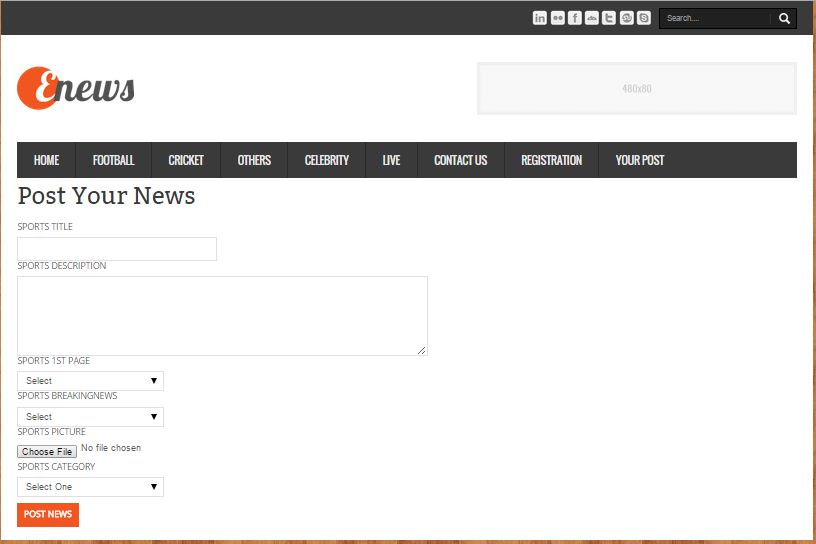
****

Figure 4.4: Home page interface for end user

4.1.5 News:

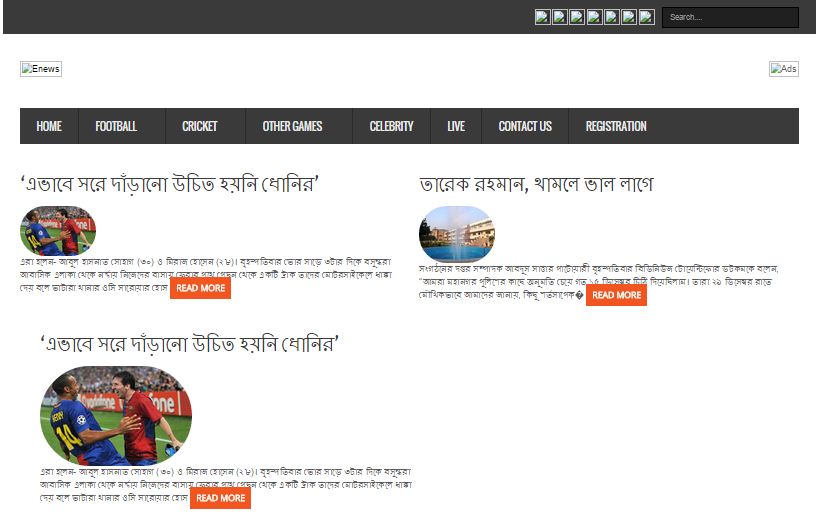
****

Figure 4.5: Home page interface for end user

1. Testing

**5.1 Overview**

Testing is the process of evaluating a system or its component(s) with the intent to find that whether it satisfies the specified requirements or not. This activity results in the actual, expected and difference between their results. In simple words testing is executing a system in order to identify any gaps, errors or missing requirements in contrary to the actual desire or requirements.

**5.2 Scope**

Testing depends on the source code but reviewing requirements and developing test cases is independent from the developed code. Finding bugs in the Software is the task of testers. Developers are only responsible for the specific component or area that is assigned to them but testers understand the overall workings of the software, what the dependencies are and what the impacts of one module on another module are. This section of the document describes the overall testing activities with test case.

**5.3 Appendix:**

**White box testing:**

5.3.1 **user login system:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Serial No:** | **Inputs:** | **Click Button:** | **Expected result:** | **Pass/fail** |
| 01. | Null, Null | Login | Message display **Error** | Pass |
| 02. | Null, 123456789 | Login | Message display **Error** | Pass |
| 03. | 123456789, Null | Login | Message display **Error** | Pass |
| 04. | 123456789, admin | Login | Message display **Error** | Pass |
| 05. | admin,Null | Login | Message display **Error** | Pass |
| 06. | Null, admin | Login | Message display **Error** | Pass |
| 07. | 123456789, 123456789 | Login | Message display **Error** | Pass |
| 08. | @$%#\*&!~(){}[], admin | Login | Message display **Error** | Pass |
| 09. | @$%#\*&!~(){}[], @$%#\*&!~(){}[] | Login | Message display **Error** | Pass |
| 10. | admin, @$%#\*&!~(){}[] | Login | Message display **Error** | Pass |
| 11. | @$%#\*&!~(){}[], 123456789 | Login | Message display **Error** | Pass |
| 12. | 123456789, @$%#\*&!~(){}[] | Login | Message display **Error** | Pass |
| 13. | admin, @$%#\*&!~(){}[] | Login | Message display **Error** | Pass |
| 14. | admin, 123456789 | Login | Message display **Success** | Pass |
| 15. | admin, admin123456789 | Login | Message display **Success** | Pass |
| 16. | admin, 123456789admin | Login | Message display **Success** | Pass |
| 17. | ADMIN, 123456789 | Login | Message display **Success** | Pass |
| 18. | ADMIN, admin123456789 | Login | Message display **Success** | Pass |
| 19. | ADMIN, 123456789admin | Login | Message display **Success** | Pass |
| 20. | ADMIN, ADMIN | Login | Message display **Success** | Pass |
| 21. | admin, ADMIN 123456789 | Login | Message display **Success** | Pass |
| 22. | admin, 123456789ADMIN | Login | Message display **Success** | Pass |
| 23. | ADmin, AdminN | Login | Message display **Success** | Pass |

Table 4: white box testing of "**login system**"

5.3.2 **Category insert system:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Serial No:** | **Inputs:** | **Click Button:** | **Expected result:** | **Pass/fail** |
| 01. | Null, Category name | add | Message display **Error** | Pass |
| 02. | Null, Null, Null, captcha | add | Message display **Error** | Pass |
| 03. | Null, Null, category name cd, | add | Message display **Error at least 3 character** | Pass |
|  |  |  |  |  |

5.3.3 **Country insert system:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Serial No:** | **Inputs:** | **Click Button:** | **Expected result:** | **Pass/fail** |
| 01. | Null, Country name | add | Message display **Error** | Pass |
| 02. | Null country | add | Message display **Error** | Pass |
| 03. | Country name 2 character | add | Message display **Error at least 3 character** | Pass |

5.3.4 **City insert system:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Serial No:** | **Inputs:** | **Click Button:** | **Expected result:** | **Pass/fail** |
| 01. | Null, country | add | Message display **Error** | Pass |
| 02. | Null city | add | Message display **Error** | Pass |
| 03. | city and country name less than 3 character | add | Message display **Error at least 3 character** | Pass |

5.3.5 **Insert News system:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Serial No:** | **Inputs:** | **Click Button:** | **Expected result:** | **Pass/fail** |
| 01. | Null title | create | Message display **Error** | Pass |
| 02. | Null, Description | create | Message display **Error** | Pass |
| 03. | Null 1st page | create | Message display **Error** | Pass |
| 04. | Null breakingnews | create | Message display **Error** | Pass |
| 08. | Null Picture | create | Message display **Error** | Pass |

5.3.6 **user sign up:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Serial No:** | **Inputs:** | **Click Button:** | **Expected result:** | **Pass/fail** |
| 01. | Null name | Submit | Message display **Error** | Pass |
| 02. | Null, Email | Submit | Message display **Error** | Pass |
| 03. | Null password or password at least 6 character | Submit | Message display **Error** | Pass |
| 04. | Null address | Submit | Message display **Error** | Pass |
| 05. | Null country select | Submit | Message display **Error** | Pass |
| 07. | Null Gender | Submit | Message display **Error** | Pass |
| 08. | Null contact | Submit | Message display **Error** | Pass |

1. **Conclusion**:

I am pleased to submit the final Software documentation report on Online sports newspaper. From this, the readers will get a clear and easy view of online sports newspaper. To improve newspaper efficiency, online newspaper is very essential. An online newspaper system is more effective than paper based manual system. This document can be used effectively to maintain software development cycle. It will be very easy to conduct the whole project using it. Hopefully, this document can also help our junior BSSE batch students. We tried our best to remove all dependencies and make effective and fully designed document. We believe that reader will find it in order.