SOFTWARE REQUIREMENTS SPECIFICATION
FOR
“SPORTSBUZZ”

phaneendra.c13@iiits.in,ankitraj.g13@iiits.in,daradevi.k13@iiits.in, kranthi.n13@iiits.in

24 September, 2015.

1. Introduction

This document is a Software Requirement Specification (SRS) for the SPORTSBUZZ Web Based project. This document is prepared by following IEEE conventions for software requirement specification.

The main purpose of the project is to provide a platform to generate the news and updates of most popular games.

1.1.Purpose

The aim of this document is to specify complete description of the Platform to be developed. It is basis for agreement between suppliers and customers about the product to be developed. The document will describe all the functional and non-functional requirements, functionalities, external interfaces and all those features that will be facilitated in the end product.

The Intended Audience for the document are the Multi-sports lovers and developers who tend to add value to it.

1.2.Scope of the Project

This Project is intended to make the user to get the details of all the sports of which he likes on a common platform. Basically, the Project is focussing on those users who are obsessed to use different kinds of applications for different sports.

In this way, an user can get accessed to all kinds of sports easily,efficiently and fastly.

1.3.Overview

The main focus of this document, is to describe the application from the user's perspective. In the following section, the product's perspective is taken to indicate the user characteristics, product functions, assumptions, etc.. The next section deals with specific system requirements, external interface requirements, performance requirements, data storage requirements etc..

2.Overall Description

This section gives detailed information about the requirements from the Product's Perspective, User's Perspective and the whole functionalities of the service in brief. This section will describe all those features which will affect the final product.

2.1.Product Perspective

This product is eventually developed for all kinds of sports lovers who don’t want to waste their valuable time to access different applications. Product will be launched for all popular sports under one app.There will be a remote server where session data will be stored.

The Product extensively. Then, if he wants to know the details and updates, it requires the users to log into the system before they can choose the type of sports needed. Once chosen, The user can now select features and choose what he wants as the final product.

2.2.Product Functions

This subsection deals with the main functions of the product. The Product can generate automatic news through artificial intelligence and can help customize them, gives alert notifications by dynamic xml databases, can store information and session progress of every user and can update new news over time.

2.3.User Characteristics

The Users of the Product are mainly the multi sport lovers of different regions who doesn’t have an certain organised system on a single platform.The user set can also be those who cannot spare their time to watch through television.

3.Specific Requirements

This Section deals with all the software requirements – both functional and non-functional which will be delivered to the user as an end product. All the Requirements are categorized into external interface requirements, functional requirements and non-functional requirements.

3.1.External Interface Requirements

In this sub-section, External Interface Requirements can be divided into

1. Application's Main Interface
2. Generic User Profile Interface
3. Customising Portal Interface

4.Database Manager Interface.

3.1.1. Application's Basic Interface

Name: Basic Interface
Purpose: To provide the user with a basic idea of what the application can do and to provide the user an option to signup.
Source of Input: This is ths main home page of the application.
Units of Measure: Information about the different sports that are available .

Softwares needed: html, css, javascript, jquery

3.1.2.Generic User Profile Interface

Name : Generic User Profile Interface
Purpose: To provide the user with flexibility by automatic generation of details of the last commit or more frequent commit he did.
Source of Input: Once the user registers, he gets to access this page.
Validity: Until the remote cloud server is working
Units of Measure: Usability, Attractive and must update data by itself.

Softwares needed: Javascript, xml, ajax.

3.1.3.Customising Website Interface

Name: Customising website Interface
Purpose: For those where application interface is not available,backend will semantically gets the data.
Source of Input: The users choice of redirecting to interface will be based on the weblink he gors for.
Validity:Once he finishes his work,his session will be stored in the databases and makes to generate updates based on these queries.
Units of Measure: Extra Features that can be added, Usability, Ease

Softwares needed: Jquery DragDrop Plugin, xml, javascript

3.1.4.Database Management Interface

Name: Database Management Interface
Purpose: Once the scores are generated, the system admin can update/insert/delete/query the database.
Source of Input: Once the user clicks on the refresh button,which is a user-interface where he can see all the databases of insert/update/delete/query them.
Units of Measure: Ease of use, easy to manage databases

4.System Features

This section deals with the functional and non-functional requirements and basically portrays what the system actually does. The section is organised in terms of system features stating the functional requirements in each. Later on, the non-functional requirements are explicitly stated.

4.1.User Management

The Product will be launched as a web application and application for Android and IOS platform and can be accessed in a web browser. The user has to fill the signup form in order to register for the service. His email will be verified and he can therefore have a unique account based on the email id. He can also signup through his Gmail / Facebook accounts.

Once registered, the user can login to his account and all his progress thereafter is saved in a cloud server. If the login information does not match with the stored values, an error dialog is shown and redirected to the login page.

4.1.1 Register

Primary Actor: User
Overview: The User Registers into the system.

1)**Sign Up**
As a User,
→ It can four ways to signup into the site.
A.Self registration.
B.Fb SignUp.
C.Gmail SignUp.
D.Twitter SignUp.
After this User,need to verify the registerd E-mail Id.
As a server,
Fetch data from fb/gmail or store the data provided for registration to the database.

2)**Verification**

As a user,
I should verify my correctness.I should login from the unique link sent to me from the server to my registered mail.
As server,
I should send a Verification link to the email of the user to the test user's correctness.

3)**Forgot Password**

As a user,
If I forget my password I should be able to reset my password.
As a server
I should create a unique Password reset link will be sent to the email id provided for the registration,So that no one else can change my password.

4)**Login**

As a User,
I should be able to login.
As a Server,
I should verify login password and id, they should match,checks from the database.

4)**Select a Sport**

As a User,
They can select the sport which they likes.In the sport they can see the News,Info of the team/Player and Live scorces.
As a Server,
If a user selects the Info,It's a static data I will linkup to the database.
If a user selects the News or Scores,It's a dynamic data I will linkup to the PHP server(Active Server).

Note:-we are trying to keep the 5 sports into this application.Same above schema works for all the sports.

5)**Feedback of the App**

As a user,
Gives the fedback of the app condition.
As a server,
Stores into the databases.

Main Flow:

1. The user sees examples on the website of different portals that can be created.
2. If the user wants to create a portal, then he clicks on the “make a portal” button when he gets redirected to a login/signup page
3. If not registered, he is asked to signup or login with facebook or gmail.
4. If he chooses to signup, he fills the signup form and enters a valid email address.
5. An email verification link is sent to his gmail inbox which on clicked confirms his validity.
6. Once registered, he can login into the system with that username and password.

Alternate Flow:

3. If registered already, he can login directly to the system with his username an password
4. If he chooses to signin using gmail or facebook, he is redirected to a page requesting access to profile information, on accepting he can login to the system.

4.1.2 Login

Primary Actor: User
Overview: The user logs into the system and all his progress is stored
Main Flow:

1. The user logs into the system with the username and password.
2. He gets to choose the type of portal he wants to design for the university
3. He starts customising the portals.
4. When he logs out, all of his progress is stored in his profile and he can retrieve that from the account whenever he wants.

REQ1: The System shall provide a User interface to see examples of portals generated.

REQ2: The System shall provide a registration page.

REQ3: The System shall send a verification link to the user's email for validation.

REQ4: The System shall store the user's details in a database and show them in the user's profile page.

REQ5: The System shall support customising portals and saving their progress.

REQ6: The System shall also provide a login option through facebook or gmail.

4.2. Generating Uniform sets of Portals

Primary Actor: System
Overview: One major problem most universities face is that there is no uniformity in their portals. If it has 10 portals, all of them have different types of user-interfaces in the front end and also diferent data retrieval mechanisms, different types of databases; even when they have a lot of features in common.This problem can be overcomed by providing a set of pre-defined common functions and also pre-defining the variant features.

Main Idea: To generate ” uniform “ but customisable portals with same script for forms, same script for animations, same script for data retrieval etc..

REQ7: The System must have scripts for forms, for animations, for data retrieval, for certain common features and for display which can be linked with diferent databases to generate different kinds of portals.

4.3. Portal Customisation

Once the user logs in and selects the type of portal he wants to create, he gets a portal

with some common and variant features already embedded in it. He can then customise this portal to reach his requirements.

4.3.1. Choosing the type

Primary Actor: User
Overview: The User chooses the type of portals he wants to create
Pre-condition: The user must login to his account.
Main Flow:

1. The user sees the type of portals that can be made for his university. For example, he sees types like social , academic, sports, etc.. under which are different portals like mess portal, library portal, teaching-assistant portal, Courses portal, sharing portal and so on.
2. He chooses a type from them and he is redirected to an interface wherein he can see the portals with some basic functionalities (a combination sub-set from common and variant features)
3. He can thereafter generate the portal directly or customise them.

4.3.2. Customising Portals

Primary Actor: User
Overview: The User can customise and generate portals
Pre-condition: The user must login and select the type of portal he wants to generate.
Main Flow:

1. The user sees many common and variant features already existing in the portal.
2. This workspace has an edit option which when clicked gives a list of additional features by the side.
3. If the user likes any feature and wants to add it into the portal, he can just drag and drop the feature (ex: live chat with logged in members) and it gets added to the portal.
4. He can have a preview with the newly added features before generating the final product.
5. Once satisfied, he generates the final product.

REQ8: The System must provide portals with a set of common and variant features already existing.

REQ9: The System must provide a drag-drop option to add features to the portals

REQ10: The System must provide a delete feature option in case the user does not need some features

REQ11: The System must be able to automatically allocate databases for every feature

added.

REQ12: The System shall provide a zip folder with complete databases and javascripts, once a portal is generated.

5. NonFunctional Requirements

5.1. Performance Requirements

1. The Application uses AJAX to retreive data from the databases so there is no need to reload webpage everytime the user updates the database.
2. The Application must be able to store cookies to remember the user's login details or to prevent restarting the server if the browser crashes.

5.2. Security Requirements

1. The user's details are encrypted and stored in the cloud server to prevent any hacking attacks.