**Group Members :**

Farrukh alam virk:l1f18bscs0424

Sohaib ahmad:l1f18bscs0411

Ather mehmood:l1f18bscs0413

**library management system of research paper**

* **Introduction and Background**

With the increase in the number of researchers and readers, it has become difficult to publish your paper, even more when other publishing services are costing you much, so we planned to design a Research Article Management System.

A nonprofit scholarly networking and publishing platform is being planned as an alternative to for-profit platforms such as ResearchGate and Academia.edu. We named this Product ScholarIN. It will help many scholars to get their document verified and publish their documents on ScholarIn if they cannot pay Money to publish it by them own. It will help many researchers financially. Our Document Verification Department will Process the Articles and state it. Using ScholarIn other users can also read verified and non Verified Articles with name of Author. And users can Search through many VANV articles. While Authors can know who visited their Document.

This Project will be funded by well known rich companies, Macrosoft and Apple.

* **Product (Problem Statement)**

The Purpose of the Product to Provide a site Where researcher can publish their document easily. The main purpose of this system to maintain a open source repository where other can upload Documents after verification and even without verification,then other users can filter out articles from verified and unverified articles. Other purpose Is to design it this way that authors can keep track that who visited his Published Articles.

* **Background**

Article Library is a Website where researchers register and get their document verified and publish to public. All known website are paid. Many of those companies gather your data and sells to other companies, This way they generate more Money from you.

* **Scope**
* On ScholarIn Users Reputation is managed on certain activities, then user can be ranked or removed accordingly.
* Readers can provide feedback to Author after reading his Article.
* ReSearchers can send their article to verify after getting their Profile Verified. Only verified Profile can request article verification.
* *Researcher can manage visibility setting of his/her article.*
* **Objective(s)/Aim(s)/Target(s)**
* Reader and Researchers can access their Profile from any device with Browser Installed.
* Reference are managed made by a Author to Other Author.
* Readers can Search about their concerned articles using search Engine within Site.
* Admin can manage every activity and can remove any document or person from site.
* System should be able to handle 100 requests per second.
* It should work 24/7 a week.
* International Users can also Access this Network.
* **Challenges**

The Only challenge is to motivate user to use ScholarIN instead of other Websites

* **Completeness Criteria**

Since this is Very large Project which will be used by million international users, we can its state complete only when It passes all Tests and Products stops receiving issues from testing users of 500 members.

* **Business Goals**

*It is a Non-Profit Product funded by Some well known companies, MacroSoft and MangoSoft. It doesn’t has any Business Related Intentions. Although if they stop funding, then we will generate revenue using Ads(selling).*

* **Document Conventions**

***ResearchGate and Academia.ed*** *:* Document Publishing Websites.

***ScholarIn*** *:* Our Product Name.

**Overall Description**

* **Product Features**
* It help you to publish your own papers easily
* It will help many researchers financially
* Every Author can know who visited their document.
* It maintain a open source repository where everyone can upload documents after verification
* other users can filter out articles
* Users Reputation is managed on certain activities, then user can be ranked or removed accordingly.
* *Researcher can manage visibility setting of his/her article.*
* Reader and Researchers can access their Profile from any device with Browser Installed.
* Researcher can provide feedback to Author after reading his Article.
* Readers can Search about their concerned articles using search Engine within Site.
* It should work 24/7 a week.
* International Users can also Access this Network.
* It can generate revenue using Ads.
* System should be able to handle 100 requests per second.
* **User Classes and Characteristics**

The various user classes that you anticipate will use this product are following:

* Different Aurthors use this product to publish their papers.
* Reserchers use this product for reserch purpose.

* Students use this product for learning purpose.
* Or Every normanl person can use this product for information purpose or collecting data.
* **Operating Environment**

The software environment required to operate are following :

Front end:

• Android developer tool

• Advance java

Back end:

• MySQL

The Hardware environment required to operate are following :

* Android version 2.3 ginger bread(minimum, android user’s)
* 2GB ram

1.2 GHz processor

Intel i5

* Windows 7/8/8.1/10
* **Design and Implementation Constraints**

Register

* Description : First the user will have to register/sign up. There are two different type of users.
* The library manager/head : The manager have to provide details about the name of library ,address, phone number, email id.
* Regular person/student : The user have to provide details about his/her name of address, phone number, email id

Sign up

* Input: Detail about the user as mentioned in the description.
* Output: Confirmation of registration status and a membership number and password will be generated and mailed to the user.
* Processing: All details will be checked and if any error are found then an error message is displayed else a membership number and password will be generated.

Login

* Input: Enter the membership number and password provided.
* Output : User will be able to use the features of software.

In Hardware the user required atleast 1.2 GHz processor , Android version 2.3 ginger bread , Windows 7, 2GB RAM.

In Software the user required Android developer tool and advance java for front end and MySQL for back end.

* **Assumptions and Dependencies**

The project could be affected if these assumptions are incorrect :

* **project** member's availability.
* **project** member's performance.
* **project** member's skills.
* Vendor delivery times.
* Vendor performance issues.
* Accuracy of the **project** schedule dates.
* 3) Functional Requirements

**Use Case Diagram:**



**Table 1: UC-1**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-1 | |
| **Purpose** | | To get register in Article Library | |
| **Priority** | | Medium | |
| **Pre-conditions** | | Must be Researcher | |
| **Post-conditions** | | Successfully registered.  Able to post the articles. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Click on Registration button | | Ask to Register as normal user or as an appropriate Scholar |
| **2** | Click on suitable option. | | Open appropriate page |
| **3** |  | |  |
| **…** |  | |  |
| **Alternate Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | User click mistakenly on a page he doesn’t want to open | | System give him/her option to return back |
| **2** |  | |  |
| **3** |  | |  |
| **…** |  | |  |

**Table 1: UC-2**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-2 | |
| **Purpose** | | Sign in | |
| **Priority** | | High | |
| **Pre-conditions** | | Must be registered | |
| **Post-conditions** | | Successfully signing.  Able to perform activities according to rights. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Click on sign button | | Ask to enter email or password, whatever as pre requirement |
| **2** |  | |  |
| **3** |  | |  |
| **…** |  | |  |
| **Alternate Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | If user forget his/her user name or password, then click on forget password option. | | System open page to recover password and user name by using link. |
| **2** | If user not registered | | Get user to registration page |
| **3** |  | |  |
| **…** |  | |  |

**Table 1: UC-3**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-3 | |
| **Purpose** | | For researcher to post his/her article | |
| **Priority** | | High | |
| **Pre-conditions** | | Must be Researcher  Must be registered in article library as researcher  Must be sign in | |
| **Post-conditions** | | Successfully post article | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Click on Post article button | | Ask to post article as verified or non-verified. |
| **2** | Click on suitable option. | | Open appropriate page. |
| **3** | Drag/drop article to post. | | Show msg on post of article. |
| **…** |  | |  |
| **Alternate Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Not registered as researcher | | Get him/her to registration page |
| **2** |  | |  |
| **3** |  | |  |
| **…** |  | |  |

**Table 1: UC-4**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-4 | |
| **Purpose** | | To verify the article | |
| **Priority** | | Medium | |
| **Pre-conditions** | | Must be Researcher  Must be verified profile | |
| **Post-conditions** | | Successfully get verify. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Click on verify article | | Check that his/her verified profile, if yes verify the article by verification department, if not show in dialogue “Your profile is not verified”. |
| **2** |  | |  |
| **3** |  | |  |
| **…** |  | |  |
| **Alternate Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | User try to verified the article, if he/she has not verified profile | | System display message according to scenario |
| **2** |  | |  |
| **3** |  | |  |
| **…** |  | |  |

**Table 1: UC-5**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-5 | |
| **Purpose** | | For researcher to check the response of user to his/her article | |
| **Priority** | | Medium | |
| **Pre-conditions** | | Must be Researcher | |
| **Post-conditions** | | Successfully check the activities towards his/her article that; how many downloads/read/views | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Click on track audience | | System get him/her to the page of tracking where he can see all activities related track audience |
| **2** |  | |  |
| **3** |  | |  |
| **…** |  | |  |
| **Alternate Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Not a researcher | | System give him/her option to return back |
| **2** |  | |  |
| **3** |  | |  |
| **…** |  | |  |

**Table 1: UC-6**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-6 | |
| **Purpose** | | To manage the visibility of article | |
| **Priority** | | Medium | |
| **Pre-conditions** | | Must be Researcher | |
| **Post-conditions** | | Successfully manage visibility. | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Click on visibility (If article exist) | | Give options that on/off visibility |
| **2** |  | |  |
| **3** |  | |  |
| **…** |  | |  |
| **Alternate Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Try to access visibility mode (Article not exist) | | System display message according to scenario |
| **2** |  | |  |
| **3** |  | |  |
| **…** |  | |  |

**Table 1: UC-7**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-7 | |
| **Purpose** | | To rate the Author | |
| **Priority** | | Medium | |
| **Pre-conditions** | | Must be reader | |
| **Post-conditions** | | Successfully rate the author (about his/her article). | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Click on feedback | | Open the page of feedback |
| **2** | Fill form and submitted | | System display message according to scenario (Submitted or an error) |
| **3** |  | |  |
| **…** |  | |  |
| **Alternate Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | User try to rate if he/her not read that article | | System display message according to scenario |
| **2** |  | |  |
| **3** |  | |  |
| **…** |  | |  |

**Table 1: UC-8**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-8 | |
| **Purpose** | | To manage users (remove) | |
| **Priority** | | Medium | |
| **Pre-conditions** | | Must be Admin | |
| **Post-conditions** | | Successfully manage the users | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Click on Remove user | | Open the page where all defaulter user’s info according to algorithm set |
| **2** |  | |  |
| **3** |  | |  |
| **…** |  | |  |
| **Alternate Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Try to remove users if he/she not be the admin | | System display message according to scenario |
| **2** |  | |  |
| **3** |  | |  |
| **…** |  | |  |

**Table 1: UC-9**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-9 | |
| **Purpose** | | To rate the article | |
| **Priority** | | Medium | |
| **Pre-conditions** | | Must signing | |
| **Post-conditions** | | Successfully able to read that article | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | Click on read article | | Open the page of all authors to be selected |
| **2** | Select author | | Open selected author articles |
| **3** | Select article | | Open in readable form |
| **…** |  | |  |
| **Alternate Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** | User try to search/read article that doesn’t exist | | System display message according to scenario |
| **2** |  | |  |
| **3** |  | |  |
| **…** |  | |  |

* **4 - Nonfunctional Requirements**

**4.1 - Performance Requirements:-**

Performance Requirements is; **how fast or responsive your software is**. How fast would a particular peice react to any of the user’s actions under certain workload.

There are many classes of Performance Requirements.For Instance; **Throughput (**how many requests the framework could handle**).**

So, here are some Performance Requirements I think our research paper guide i.e. **ScholarIn** should go under.

* **Response Time:-**

Depending on the context and the workload, **response time** is how fast the requests are being processed. Each particular case has a response time. Response time should be in seconds if accessing any webpage or research article but a tenure of 20-30 mins is okay for a big batch job.

In our case, the **ScholarsIn,** we will provide a response time where a user could access their research document in seconds and the verfication and publishing wouldn’t take much time, so that the publishers can publihs their documents or research papers ASAP!

For example; the requirement for response time for a web API endpoint to be under 200ms.

* **Throughput:-**

Throughput tells about the work load on the framework and is measured in **operation per time period.** Throughput is the rate at which incoming requests are processed or accepted.

In our case, the **ScholarsIn,** we will provide a throughput as responsive as it could be for the users and publishers to go with the flow.

* **Scalability:-**

It is how to assess the highest workloads under which the framework/system will meet the performance requirements.

In our case, the **ScholarsIn,** the scalability method would start with **google recommendations for the regular web pages.** Since, google is very sensitive with its web pages, therefore, we’ll provide the speed insights or probably the google’s speed insights for speed check.

* **Portability:-**

Portability defines how a framwork can be launched on one or another environment. It includes other using platforms or any other hardware or software. It gives the best performance time for one platform to perform over another.

**4.2 - Safety Requirements:-**

Safety non functional requiremtent of a software is a batch of system safety. The software cannot be allowed to function independently.

Any loss, damage or harm can be maintained by keeping some safety requirements such as keeping the software up-to-date, enforcing a strong password policy, keeping website clean, backing up the data and also hiring a security expert would make a lot of difference.

* **4.3 - Security Requirements:-**

This non functional requirement assures that the system and its data is safe from malware or any other viscous attacks or any unauthorized access.

In our case, the **ScholarsIn,** therer’ll be every safety requirements. Here’s a catch! To protect your panel or admin panel from any malware, we’ll provide users with login flow and different user roles as system behaviour or user actions.

Our software will be provided with the specific threats the system will be protected from, such as, the details should be considered under what circumstances the unauthorized access or any malware has attacked. This software will also have the comprehensive authorization scheme each actor.

* **4.4 Additional Software Quality Attributes:-**

It facilitates the performance measurements of the product by software testing profressionals. Some attributes, such as; availability, reliability, testability, portability etc.

It is helpful in designing the software architecture for the new product.

An example of quality attributes is; a product must be able to stream video content to 0.5-1M concurrent users 24/7 all over the world.

Its easy of use over ease of learning is that with the quality attributes, it is easy to maintain the system and correcting defects. **Usability** can be measured in terms of ease of use.

Here are some **ease of uses** of quality attributes;

* **Function Suitability:-**

Degree to which a framework provides functions that meet stated needs when used under specified conditions. It is only concerned with the implied needs. Such as; Functional completness, functional correctness, functional appropriatness.

* **Portability:-**

Portability defines how a framwork can be launched on one or another environment. It includes other using platforms or any other hardware or software. It gives the best performance time for one platform to perform over another. It also includes adaptability, installability, replaceability.

* **Maintainability:-**

Degree of effectiveness and efficiency with which a product or system can be modified by the intended maintainers. It can include corrections, improvements etc. It includes those carried out by specialized support staff. It also includes installation and updates of the software.

* **Usability:-**

This can be measured in terms of ease of use. The app should be user friendly, should be easy to learn and navigating should be simple.

* **Reliability:-**

The working of the project under different working environments and different conditions is reliable on product reliability.

* **Testability:-**

It makes the system easy to find defects and to test them. Can be divided into different modules for testing.

* **Other Requirements**

Some other requirements are:-

* **External Interface Requirements:-**

According to Richard Thayer (2002), **“External interface requirements specify hardware, software or database elements with which a system or component must interface…”**

* **Hardware Interfaces:-**

It defines the characteristics of each interface between hardware and software components of the system. This may include the controlling interactions between software and hardware or any other supported devices.

* **Software Interfaces:-**

Control items exchanged between the software components.

Describes the services needed by the external software components. Identifies that the data will be shared all over the software components.

* **Communication Interfaces:-**

Describes the requirements for any comm. functions that the product will use, including mails, browsing, online communication etc. All the e-forms are controlled by this interface.

* **Data Base Requirements:-**

In developing, the concept of database is well understood. The data needed to store or to retrieve is done by data base.

* **Widespread Programming Language Support:-**

To simply integration efforts, languages frequently features standard APIs that can interact directly with database. Well known examples are the Java database connectivity ;API for java, ADO.NET framework.

Providing direct access from within the application code via APIs is one way to achieve database integration.

* **Data Access via SQL:-**

The real power of databases is the retrieval and analysis of the data. Hence, developers use SQL because it allows them to interact with the databases while providing the same capability to business users. Examples of SQL integration are; C/C++, Clojure, Microsoft.NET, Node, Objective-C etc.

**Reference:**

From the article of Alexander Podelko

From the article of Oracle framework.

From Academia.edu or Wikipedia