GRST Programming Challenge Aggregator

Prepared by: Matthew Griffin, Ashish Rastogi, Kiritbhai Soheliya, Martin Tiernan

Software Engineering, Fort Hays State University

Table of Contents

Customer Statement of Requirements (CSR) 3

Glossary of Terms 5

System Requirements 6

Project Management 10

1. **Customer Statement of Requirements (CSR)**

**1.1 Problem Statement**

There are many websites that offer programming challenges. Typically, these sites are used by an interviewee preparing for a job interview, or a computer science student furthering their education. These sites vary in usability and focus. If you are looking to practice and hone your programming skills across a wide variety of topics for potential upcoming interviews or to help study for upcoming classes, you face a variety of problems. A typical sequence if you are looking for an appropriate challenge or challenges is as follows:

1. **Navigate the web to an appropriate site**
2. **Learn the site U.I. and navigation**
3. **Find a challenge that suits your interests**
4. **Repeat 1-3 until sufficiently prepared**

This current process leaves a lot to be desired and that forces the user to spend an unnecessary amount of time on web browsing and learning various site navigation. The potential solution should be one that eliminates the tedium that comes from having to repeat steps one and two of this process and save the users valuable time that they could use to prepare for an interview or an upcoming test.

**1.2 Solution**

The proposed solution should present these challenges from multiple sites and present them in a unified design that has a centralized location. Ideally the new sequence should be:

1. **Download/Access the app/site**
2. **Learn the site U.I. and navigation**
3. **Find a challenge that suits your interests**
4. **Repeat 3 until sufficiently prepared**

Users should no longer have to find challenge sites on their own, and they will be able to view challenges from other sites without having to learn a new interface. Thus, eliminating the repletion in steps one and two and ultimately saving the user time. Finally, by pulling from many sources the user will be offered with the broadest possible selection of challenges that they can find at a single location.

The proposed solution must be able to gather and store adequate information about each challenge. This information should be reliable, and a user should not have to worry about a challenge “disappearing” and be forced to scour the web. It should gather enough information that a user should be able to complete a challenge without the need of going to the source website, however they should be able to find the source website if desired.

To ensure that users can adequately locate and complete challenges, a serviceable interface must be provided so they can find challenges to their liking. Showing every problem from all the included sites could overwhelm potential users. Users must be able to use a menu to search and filter the database to limit the quantity of entries to a more manageable form. They may only be interested in problems from a single or several source sites, or only looking for problems of an easy difficulty. Finally, the results must be presented in a way that users can view and find the complete text and other challenge specific information.

A secondary goal of the project should be a more personalized user experience. This includes allowing the user to upload their own personal challenges. Also, a user may want to be to create a list of favorite challenges and save them for later. Users may want to be able to toggle the status of challenges between solved, viewed but not solved, not viewed, skipped, and newly downloaded. A user may wish to add text or attach files associated with the challenges, so they can revisit their implementations or, so they can review past solutions. These functions would allow a user a unique view of the site and provide additional advantages if they do continually use the product over time.

The only true way that this project solves the problem if a user finds it easier to locate challenges than manually going each site. Great care must be taken when creating the interface. The time to learn the created interface must be significantly lower than the time to learn the sum of the websites interfaces. Additionally, if searching the database is slow and unreliable, then the original process of visiting each site might be more convenient to the user. It is not enough to simply present the user with access to all challenges. A user **must** save time when comparing with the traditional method to be a true solution.

1. **Glossary of Terms**

**Challenge**- A programming challenge commonly found on the internet or in computer science coursework. Typically focuses on learning a specific concept over practical application.

**User-** A person that is using the application.

**Database –** the storage of all the collected challenges and text associated with the challenges and accompanying websites.

**Web Scraper-** A program that gathers publicly available data from the internet and typically stores it in a database.

**User Interface-** The interface that lets the user interact with and search within the database.

**Search/Query-** A user filters or limits the results using certain parameters such as site origin.

**Results-** The list of challenges that is presented to the user after a search query.

1. System Requirements Engineering
   1. Enumerated Front-end Functional Requirements

|  |  |  |
| --- | --- | --- |
| Identifier | Requirement | PW |
| REQ1 | The system shall allow user to search challenges using keywords | 1 |
| REQ2 | The system shall allow user to filter the searched result | 3 |
| REQ3 | The system shall allow user to mark challenges as complete | 5 |
| REQ4 | The system shall allow user to register | 4 |
| REQ5 | The system shall allow user to login and manage profile | 4 |
| REQ6 | The system shall allow user to save challenges into user profile | 5 |
| REQ7 | The system shall allow user to view challenges | 2 |

The key objective for this system is to allow users to search the challenges from the consolidated database. So, REQ1 has the highest priority, because emphasis is that users shall be able to search with no string attached.

Going with the natural flow of user expectations, users shall be able to view challenge/s that makes it a Second priority (REQ7) requirement.

User interaction can be made impactful if advanced search and filter capability is provided to user, that brings us to define the REQ2 and third highest priority requirement.

Generally, users use the challenge websites and resources at events where they need to appear for an interview or brush-up the skillset, with that in mind users don’t really want to create username and password if they had to use resources for limited period of time.

It’s an optional requirement for a user to register as a new user, login with registered user and manage profile to view history on search results, change password, change avatar etc., REQ4-5 will serve the purpose.

REQ3 & REQ6 are the lower priority requirements, serves as next step in using the resources from such a site

* 1. Enumerated back-end functional Requirements.

|  |  |
| --- | --- |
| Identifier | Requirement |
| REQ1 | The system shall run periodical web-scrapping queries from all listed website. |
| REQ2 | The system shall record result of web-scrapping queries in local database |
| REQ3 | The system shall allow administrator to initiate web-scrapping queries |
| REQ4 | The system shall allow administrator to add or remove challenges. |

The basic idea of this system is to gather various challenges from several websites and provide a one-stop tool to users. The primary requirement REQ1 is to allow the system to run automated web-scrapping queries periodically.

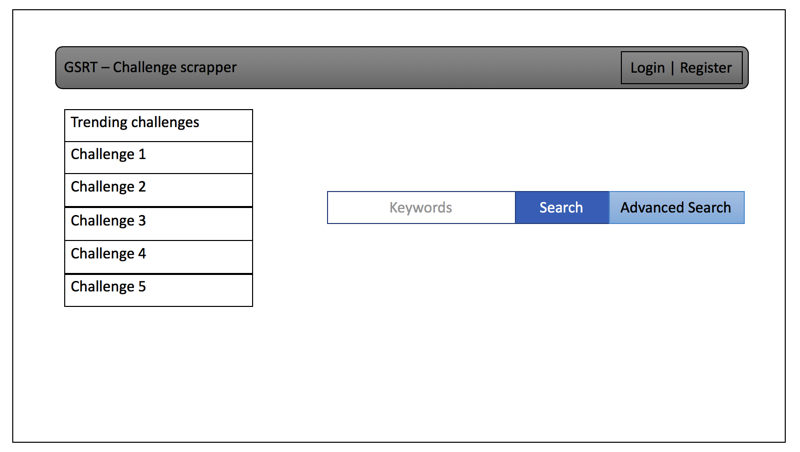
System will have local database to store all challenges, so that system doesn’t have to rely on external storage which can induces network related issues for the users, It is covered in REQ2.

The system will also allow administrator/s to run ad-hoc queries as needed.

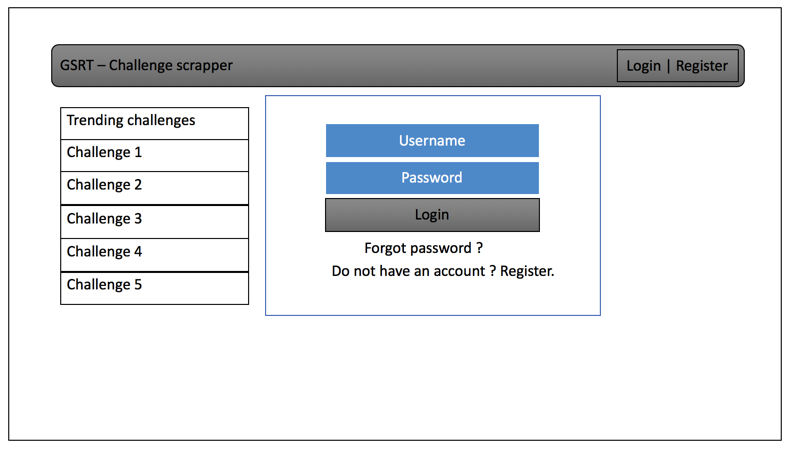
System will also provide administrative capability to add or remove challenges on need basis.

On-Screen Appearance Requirements

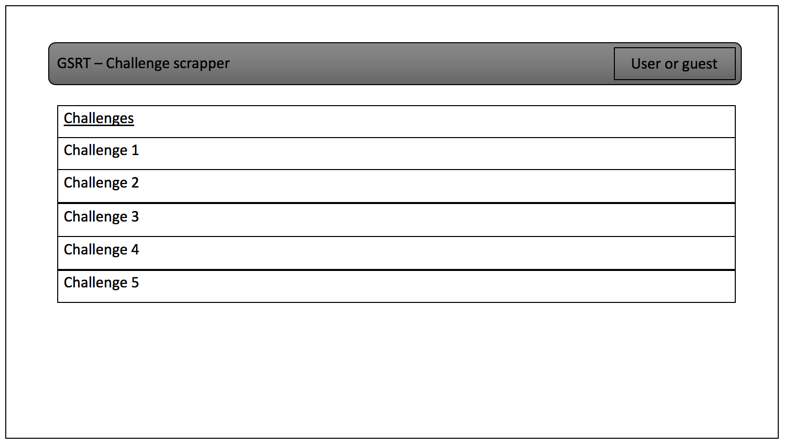
Page 1. This is the landing page of the website.



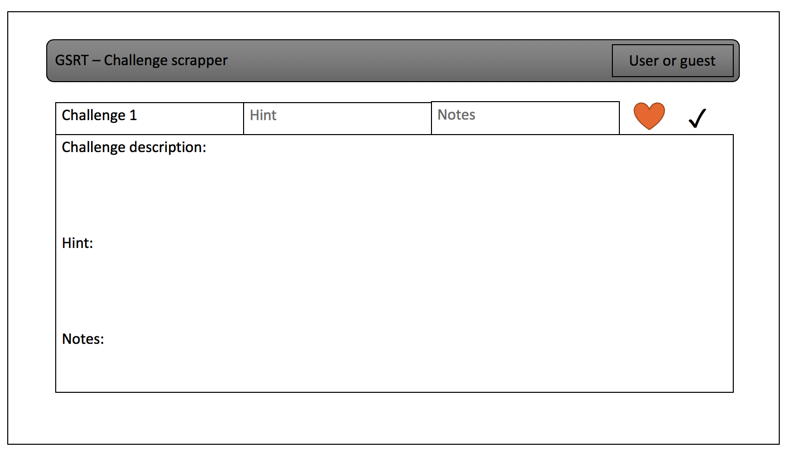
Page 2. User login page of the website, register page will have the same view. It’s a minimalistic design.



Page 3. It will display the challenges from the search criteria.



Page 4. Individual search results or challenges can be view separately, making notes capability is provided and will have the hint section too.



6. Project Management

Above is a Gantt chart showing all required documentation that needs to be completed for Report 1. The in-depth description of events can be obtained from Dr. Marsic’s website and will bot be duplicated here. Longer term documentation goals are described in the syllabus and will be documented in depth until when are closer. The hierarchy of prerequisites for actually creating the project is still be examined. When that is complete, at the end of Report 1, an additional Gantt chart for the project itself will be created.

Breakdown of Responsibilities:

Report 1 Part 1:

Matthew Griffin: Project Management Section, Gantt Chart

Ashish Rastogi: Functional and Non-functional Requirements

Kiritbhai Soheliya: On-Screen Appearance Requirements

Martin Tiernan: CSR Problem Statement and Glossary