A picture containing text, sign

Description automatically generated

College of Computing

Computer Science Department

CS3141 Team Software Project

Fall

**Image Huntr (Subject to change)**

Section: R02

Team #: 5

| Name | Role |
| --- | --- |
| Rhys Brockenshire | Scrum Master |
| Luke Rogin | Developer |
| Tom Dolan | Developer |
| Chris Torrey | Developer |
| Grant Willison | Developer |

Instructor:

Serein AL-Ratrout

# **Abstract**

(***Abstract*** *is one paragraph that summarizes your project, and describes the content and scope of the project objective, methodology, findings, and conclusion. So, you need to write one paragraph that gives an abstract idea about the entire project, the aim of the project, the process model you used, the tools, what you have done, the results, and your conclusion. If you think the project is worth extending to a Final Year Project (FYP) by you or other students or can be adopted and extended by industry/market, then mention that here and add it also as future work.)*

Example of abstract

In this project a student registration web application for university students and staff was developed, the aim of this application is to provide a simple set-up of programs for student enrolment, improve efficiencies and eliminate unnecessary paperwork. The system mainly has two modules: students and staff. Students can create an account then sign in/out, add, update, delete, and modify their data and schedule. Staff can also create accounts and then sign in, add, update, delete, and modify their data. The scrum agile process model is followed during project development and the implementation was realized by the use of object-oriented PHP, HTML, MYSQL, and Dreamweaver technologies.

It has been found that the final system was simple and user-friendly with an easy user interface, hence the end-users do not need to undergo extensive training or require any special skills. It was also secure, and reliable.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# **Table of Contents**

# **Table of Figures**

Chapter1

Specification

# **1.1 Introduction**

We are creating a web scraping software that will pull images from the internet based on a web query we do, with a user interface. For example, if you type in ‘cats’ and ‘150’, 150 images of cats will appear. Also, you will be able to apply multiple filters to get more specific results.

# **1.2 Problem Statement**

The problem we are trying to solve is not getting specific enough results when searching images on google. We want to solve this so user can get more specific image results so that they can get what they want faster. For instance, if a user wants an image of a black cat sleeping in a tree, they may find a few images, but our search result will bring as many images of said situation to the user as they request. This will also mean the user doesn’t have to sift through tons of images to find what they’re looking for.

Problems:

1) Not getting the proper images to appear in a search.

2) Not being able to search for a super specific category.

3) Not being able to choose how many images appear on the screen

# **1.3 Aim and Objectives**

Aim:

The aim of this project is to develop a web application to retrieve images from a user input to allow users to access more images at a time from the internet than they would otherwise be able to do. This will be achieved by utilising high precision filtration and algorithms.

Objectives:

* Accurate search results
* Fast image retrieval
* minimise data loss
* visually appealing interface

# **1.4 Stakeholders**

Stakeholders consist of us, as developers, and those who may use our web application.

# **1.5** Methodology

Our team will follow the scrum software development process utilising JIRA to keep track of tasks. the basic structure of tasks we will be following is as follows develop idea

* Project Initiation: Define the project scope, objectives, and requirements
* Planning: Create a project plan outlining tasks, timelines, and resources
* Requirements Gathering: Elicit and document detailed functional and non-functional requirements.
* Design: Create architectural designs and system diagrams.
* Development: Write code according to the design specifications.
* Testing: Conduct various levels of testing, including functional, integration, and system testing.
* Deployment: Prepare the software for deployment to a production environment.
* User Acceptance Testing: Involve end-users to validate that the software meets their needs.
* Documentation: Create user manuals, technical documentation, and release notes.
* Training: Train end-users, administrators, and support staff on how to use and maintain the software.
* Maintenance and Support: Provide ongoing support for the software, including bug fixes and updates.
* Project Closure: Conduct a post-project review to evaluate what went well and areas for improvement.

# **1.6 Tools**

- Bun

- ReactJS

- IDE of Choice

- SQL (or other database system)

- Online hosting service

# **1.7 High-Level Business Requirements**

## **Functional Requirements**

* Search bar to add constraints
* display and sort images
* Access to Download images that the user request

## **Non-functional requirements**

* acquire images quickly
* easily usable interface
* store images efficiently for user sort through and browse

# **1.8 Product backlog**

| **Priority** | **User Story** | **Tasks** | **Estimated effort** | **Sprint** |
| --- | --- | --- | --- | --- |
| \*\*\* | As a user I want to be able to apply filters to my search so that I can get more specific results | **Design and implement interface** | **2 H** | **1** |
| **Remove non-applicable images** | **2.5 H** |
| **Show resulting images** | **2 H** |
| **\*\*** | As a user I want to have a simple user interface so I can use the software easily | **Input fields and buttons/tabs** | **1 H** | **1** |
| **Choose aesthetically pleasing colors** | **.5 H** |
| **Set up correct layouts for device types** | **1.5 H** |
| \* | As a user I want quick results so that I will get my information quickly | **Research appropriate algorithms and methodology** | **1 H** | **1** |
| **Set up high-precision filtration and algorithms** | **4 h** |

* 1. **Security Requirements engineering practice(s)**

***[you can apply one or more of the following practices]***

1. **Establish Security and Privacy Requirements**
   * **Identify key milestones and deliverables**
   * **Assign security experts**
   * **Define minimum security and privacy criteria for an application**
   * **Deploying a security vulnerability tracking system**
2. **Create Quality Gates/Bug Bars**
   * **Define minimum acceptable levels of security and privacy quality**
3. **Perform Security and Privacy Risk Assessments**
   * **Identify the need for threat modeling and security design reviews**

Chapter 2

Analysis and Design

For chapter 2 analysis and design,

Analysis specifies what the system should do, and design is the process of planning a new system or replacing an existing system by defining its components or modules to satisfy specific requirements.

In this chapter, you need to draw the following UML diagram + textual format to explain them  
  
UML use case diagram + textual formal to explain it  
  
UML class diagram +  textual formal to explain it  
  
UML sequence diagram + textual formal to explain it  
  
You can keep revisiting and updating this chapter (updating the design for the proposed system) during project development since the scrum software process allows this.

* **Security Design practice(s)**

***[you can apply one or more of the following practices ]***

1. Establish Design Requirements
   * Address security and privacy concerns early to reduce a project's expense.
   * Validate all design specifications against a functional specification
2. Perform Attack Surface Analysis/Reduction
   * The points attacker can use to break your system
   * Analyze the overall attack surface (points)
   * disable or restrict access to system services
   * apply the principle of least privilege
   * Employing layered defenses
3. Use Threat Modeling

* Simulate attack scenarios.

Chapter 3

Implementation

For the implementation section, you add screenshots for your implementation, and interfaces of your product/ project with simple statements to explain how users will use them.

Keep revisiting and updating this chapter during project development since the scrum software process allows this. (updating with screenshots for the new increments/components )

**Security implementation practice(s)**

***[you can apply the following practice]***

1. Deprecate Unsafe Functions

* Analyze all project functions to ban unsafe ones
* Replace the unsafe functions with safer alternatives

Chapter 4

## Validation

**For Chapter 4 (Validation)**: here you need to write about the process of checking that your software system meets specifications and requirements so that it fulfils its intended purpose, and to confirm or prove the accuracy of your project.

Write about your testing and validation; **level of testing** you had, unit testing, integration testing, validation testing, and acceptance testing.   Did you have **manual or automated** testing or both? specify the part(s) that have automated testing and part(s) that have manual testing, and **What is your oracle?**

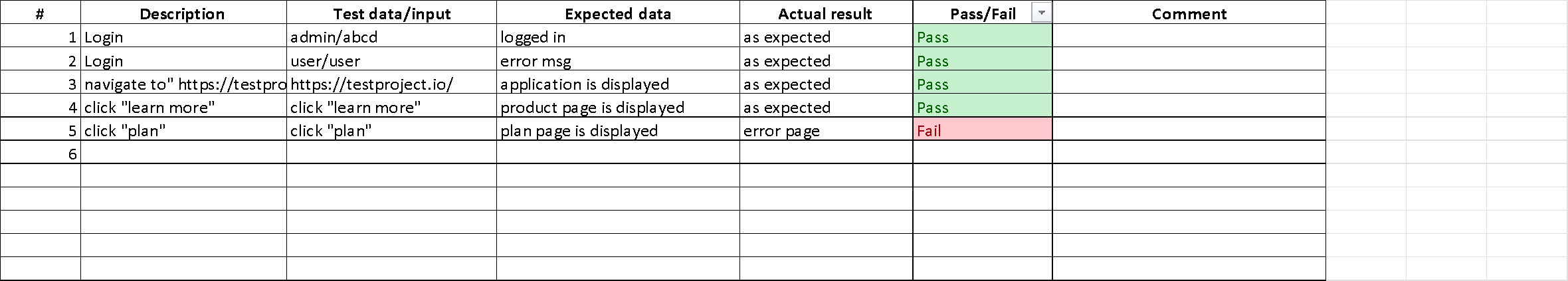
**Write the** **test cases** for valid and invalid **input** (please see Week3 Automated Testing/ slide 11),

then confirm that no errors in the code that the application is able to operate in the required condition (OS, web browsers) and that you have created the code correctly.

For validation and acceptance testing write who tested your system. MTU students? computer science student? other department students? your group only? other college students? public users? How many students/users? How many times? could they use it easily or did they make mistakes?

You keep revisiting and updating this chapter during project development since the scrum software process allows this. (updating with test cases/ test suites you have done for the new increments/components)

Create a **testing document** with a ***comprehensive list of actions to be performed and the expected results of those actions***



**Security validation practice(s)**

***[you can apply one or more of the following practices]***

1. Perform Dynamic Analysis
   * Perform run-time verification of the software
   * Penetration test.
2. Perform Fuzz Testing: an automated software testing technique that attempts to find hackable software bugs by randomly feeding invalid and unexpected inputs and data into a computer program in order to find coding errors and security loopholes.
3. Conduct Attack Surface Review

* Review attack surface upon code completion
* Ensure that any new attack vectors created as a result of the changes have been reviewed and mitigated including threat models

Chapter 5

## Limitations and Future Work

**For Chapter 5**

Limitations: address everything that the project left,  if some project backlog items/ features/ requirements have not been implemented then mention them in this part with an explanation/justification why you couldn't implement them (Time constraints the time was not enough, some developers were unavailable, because of COVID19, or  tool limitation ....etc.). Many students tend to feel that presenting the limits of their work makes work weaker. on the contrary, approaching this section shows maturity for the academic universe, and writing about them actually strengthens your work by identifying any problems before reviewers/readers find them.

Future work : if the limitations can be addressed in the future then add this in here in future work, moreover, if you believe this project can be extendable (add more features/ more parts) that the project is worth extending to a Final Year Project (FYP) by you or other students or can be adopted and extended by industry as a product so you can give directions for that in future work.

Chapter 6

Conclusion

**For  Chapter 6 (Conclusion),**: write what you have concluded.

Examples:

I solved many problems in the project…

This application/project/system was applied to improve the learning process.

The results of this project showed that the system significantly facilitated the students' learning process.

The system is useless, acceptable, usable, beneficial, or maybe enjoyable and why do you believe that?

# References

[1] *React*. (n.d.). Retrieved October 1, 2023, from https://react.dev/

[2] <https://tspt5f23.atlassian.net/jira/software/projects/T5/boards/1>

[3] Bun — A fast all-in-one JavaScript runtime. (n.d.). Bun. Retrieved October 1, 2023, from <https://bun.sh/>

[4] Vercel: Develop. Preview. Ship. For the best frontend teams. (n.d.). Retrieved October 1, 2023, from https://vercel.com/

# Appendix:

**Security Release practice(s)**

1. Create an Incident Response Plan

* Prepare an Incident Response Plan to address new threats that can emerge over time
* identify appropriate security emergency contacts
* establish security servicing from other groups

1. Conduct Final Security Review

* Review all security activities that were performed to ensure software release readiness. Examine threat models, tools outputs,and performance against the quality gates and bug bars defined during the Requirements Phase

1. Certify Release and Archive

* Certify software prior to a release to ensure security and privacy requirements were met
* Archive all specifications, source code, binaries, private symbols, threat models, documentation, emergency response plans, and license and servicing terms for any third-party software.