

**CS 320 Course Project Final Report**

**for**

**Python Onboarding**

**For Incoming Students**

**(POFIS)**

**Version 0.1**

**Prepared by**

**Group Name: Saab**

|  |  |  |
| --- | --- | --- |
| **Rebecca Daniel** | **11585986** | **rebecca.daniel@wsu.edu** |
| **Andrew Cornish** | **11571297** | **andrew.cornish@wsu.edu** |
| **Samuel Dunn** | **11453733** | **samuel.i.dunn@wsu.edu** |
| **Abdi Vicenciodelmoral** | **11554779** | **a.vicenciodelmoral@wsu.edu** |

**Date:** **12/13/2019**

Contents

[1 Introduction 1](#_Toc26214193)

[TO DO: Please provide a brief introduction to your project.> 1](#_Toc26214194)

[1.1 Project Overview 1](#_Toc26214195)

[1.2 Definitions , Acronyms and Abbreviations 1](#_Toc26214196)

[1.3 References and Acknowledgments 1](#_Toc26214197)

[2 Design 2](#_Toc26214198)

[2.1 System Modeling 2](#_Toc26214199)

[2.2 Interface Design 2](#_Toc26214200)

[3 Implementation 3](#_Toc26214201)

[3.1 Development Environment 3](#_Toc26214202)

[3.2 Task Distribution 3](#_Toc26214203)

[3.3 Challenges 3](#_Toc26214204)

[4 Testing 4](#_Toc26214205)

[4.1 Testing Plan 4](#_Toc26214206)

[4.2 Test for Functional Requirements 4](#_Toc26214207)

[4.3 Test for Non-functional Requirements 4](#_Toc26214208)

[4.4 Hardware and Software Requirements 4](#_Toc26214209)

[5 Analysis 5](#_Toc26214210)

[6 Conclusion 6](#_Toc26214211)

[Appendix A – Group Log 7](#_Toc26214212)

# Introduction

## Project Overview

POFIS aims to become a standalone educational assistance asset for WSU-V. Its primary target audience is incoming transfer students with exposure to programming, but a lack of experience with the Python programming language. POFIS will contain useful reference pages, code snippets, a cookbook for more common design practices, links to established python modules, and an interactive tutorial. While it will be self-sufficient, it is not a substitute of education, and therefore will not aim to teach programming, merely standards and best practices for Python. Instructors may refer students to POFIS if they feel a student would benefit from the assets contained within POFIS.

## 1.2 Definitions, Acronyms and Abbreviations

POFIS – Python Onboarding For Incoming Students

WSGI - Web Server Gateway Interface

WSU-V – Washington State University, Vancouver

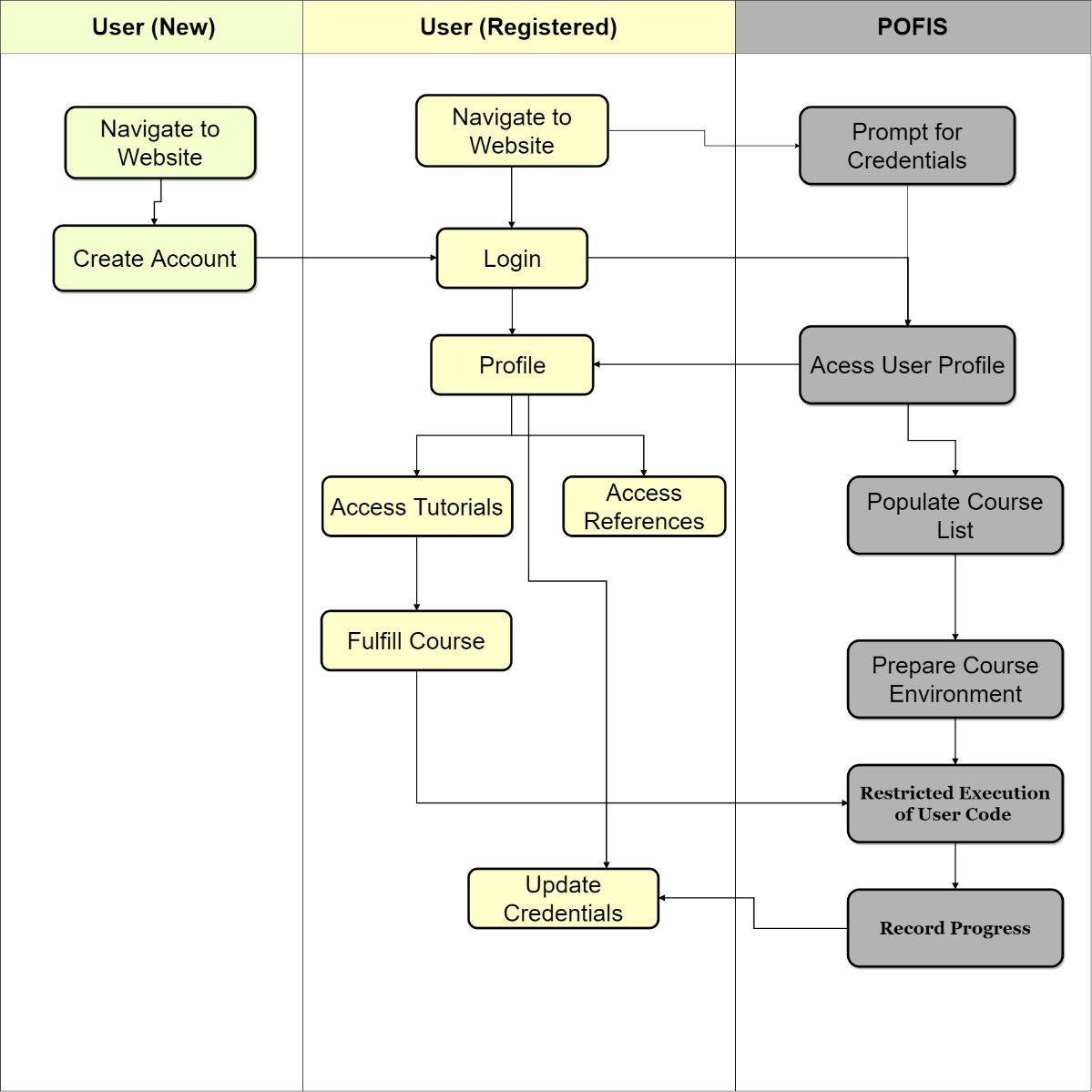
## References and Acknowledgments

*IEEE Guide to Software Requirements Specifications*, 1st ed., The Institute of Electrical and Electronics Engineers, Inc, New York, NY, 1984, pp.1-26

# Design

## System Modeling

During the implementation of POFIS, some modifications were necessary from the previously anticipated system modeling. Furthermore, it was determined that the following diagram provided a better representation of the process in a more concise manner.



## Interface Design

The following screenshots are representative of each subsystem of POFIS.

NEED STUFF

# Implementation

## Development Environment

The specific development environment varied throughout the implementation of POFIS throughout the process. Most commonly, Linux was utilized as the preferred operating system, although the specific Linux distributions varied. Sublime3 was commonly used as a text editor throughout the project, although other simple text editing programs were also employed. Sublime3 and IntelliJ were commonly utilized for editing Python, JavaScript, HTML, Jinja2 and Sphinx code. In addition, GitHub was used extensively for collaboration and version control.

## Task Distribution

Throughout the design and implementation process, all team members contributed to every aspect of the system. The following table highlights the specific tasks which were spearheaded by each member.

|  |  |
| --- | --- |
| Rebecca Daniel | Authored Sphinx restructured text pages. |
| Andrew Cornish | Report editor, Authored Sphinx restructured text pages |
| Samuel Dunn | Sphinx implementation and mentor, |
| Abdi Vicenciodelmoral | UML and diagram designer, User account procedure |

## Challenges

As the project progressed, the team faced many challenges during the entire implementation process. The most notable challenge faced was safely executing user generated code.

* + 1. **Safe execution of user generated code**

It was decided that pure lexical analysis of user input would be prone to logical gaps and errors. To avoid this, it was determined that it was necessary to execute user generated code. The decision to execute user-generated code posed many security threats. For instance, malicious users could gain access to the servers file system and potentially delete files or execute malicious code. In order to remove this security risk, it was determined that the user generated code must be executed in a controlled environment. This was accomplished in POFIS by substituting the interpreters core functionality with unittest.mock.MagicMock instances, which will prevent the user from performing actions such as, opening files, importing modules, reading from stdin, and other actions deemed unsafe.

* + 1. **Infinite loop**

Although the most hazardous functionality a user could access was mitigated by the previous solution, there remained a method which the user could submit code which could cause an infinite loop. If a user, maliciously or accidently, submitted code which caused an infinite loop the result could render the service inoperable. To mitigate this possibility, user generated code is evaluated in a separate, killable, process from the webserver. The webserver will periodically check for completion of execution and evaluation and reply to the user with results when they are made available. If the execution time of user generated code exceeds 500ms, the server will kill the evaluation process and inform the user of the timeout failure.

# Testing

<This section is a summary of your testing report>

## Testing Plan

User account creation: The user account creation shall be tested with multiple accounts using various length names. Attempts will be made to ensure only the proper syntax is allowed in each field.

User login: User login shall be tested using multiple accounts.

Home page: Ensure the home page works on listed browsers.

Tutorial pages: Multiple pages will be tested for functionality. As a standard practice, each addition to a tutorial will be fully tested prior to implementation on the live server

## Test for Functional Requirements

<Describe your test results for the functional requirements.

TODO: Provide a list of use cases or functions you have tested, as well as the testing results (whether or not the system passed the tests).>

## Test for Non-functional Requirements

<Similar to the Section 4.2, but this section is for the non-functional requirements. >

Under normal operating conditions, the POFIS system shall be capable of the following:

1. Home and tutorial selection pages shall not take longer than 15 seconds to load.
2. User account creation shall not take longer than 10 seconds to confirm or refuse.
3. Initial session loading of any tutorial shall not take longer than 15 seconds.
4. Further steps of a tutorial should not take longer than 5 seconds to load.
5. Code submitted by a user shall take no longer than 10 seconds to receive feedback.
   1. Exceptions to this rule may occur if tutorial has extensive coding involved.
   2. If an exception is warranted, the instructions must indicate that the processing time may be longer than the norm.

**Safety and Security Requirements**

The system is to be used as an educational and informational reference only. Use of the system is not intended to be a requirement, therefore PII is not necessary for account creation. The use of passwords is still deemed necessary and will follow the following guidelines:

1. Password must be 8 to 12 characters in length
2. Passwords must contain at least one numeric character
3. Passwords may contain the following special characters: @, # and $
4. Certain passwords (such as “passw0rd”) will be refused

## Hardware and Software Requirements

Tests shall be performed using Windows, Macintosh and Linux operating systems using common web browsers for the various systems. If possible, the listed operating systems shall be tested using Mozilla Firefox v.68+ and Google Chrome v.7+. Furthermore, the Windows system will be further tested using Microsoft Edge v.44+, and tests using, Macintosh Safari v.11+ will be conducted on the Macintosh operating system.

# Analysis

<In this Section you need to analyze the effort that has been put on this project.

TODO: Describe how many hours (approximately) each team member spent on the project, for each milestone, which milestone takes the most effort and why. >

# Conclusion

<Conclude the document with what you have learned through working on the project.>

# Appendix A – Group Log

< Describe how frequently the group members meet during the semester, and how effective the communication is. This is optional for one-person projects.>