Program Showcase

Proposal, Technical Project
Plan #1
to be presented on or before June 9th, 2018

Student: Melissa Gonzalez
In support of Bachelor of Computer Science
Western Washington University

Client: Dr. Jagodzinski
Email: Filip.Jagodzinski@wwu.edu

Advisor: Dr. Jagodzinski
Email: Filip.Jagodzinski@wwu.edu

Table of Contents

1. Introduction

Programmers often write code in different languages and they execute their programs in a variety of different ways. Executing code may depend on their preference of a GUI or a simple command line interface however, it may also depend on the specific language they are working on. The process of writing, editing, compiling and executing code requires special developer tools that someone outside the programming scope may not fully understand. For this reason, sharing a code's output depends on whether the person whom they are sharing the output with knows how to use certain tools to execute the code.

Program showcase will create an environment where users of all skill levels will be able to share their Python program's output. This website will allow users to create an account where they can upload their files to then share with anyone without the need to download additional compiling tools. Users will be able to search for their programs once they are uploaded and execute them right on the website's emulator. Because users will only be able to see their program's output, they will not need to worry about the privacy of their code. When another user searches for projects to execute, they will only be able to view the output.

1.1 Problem Statement

My client wants software to facilitate sharing programs. Specifically, for a group of young programmers that during a summer camp, will want to share their work with family members who do not know how to compile and execute Python programs. Therefore, this website will facilitate the task of searching and viewing the desired output while keeping the actual code private.

1.2 Needs Statement

This project will allow sharing only the output of a program that a user has worked on with friends or family members outside of the computer science scope. If they know a specific username, they will be able to look up the files the user has uploaded, and then execute them. Users can upload without the risk of plagiarism because their actual code will not be public.

1.3 Objective

The complete project will be a tool that will be utilized by users who write Python code. The website will provide an environment where they will be allowed to run their programs across all web browsers without depending on external compiling tools or worrying about installing the programming language themselves.

2. Proposed Technical Approach

The website will be hosted on a server provided by Western Washington University's computer science support. The website 's interface will handle all user input therefore it is important to create a process where the front end interface accepts input that is then processed in the backend or sent to the database if needed. Once an action is complete, the request from the user will be displayed on the screen.

2.1 Requirements

Requirement	Solution
1) Account creation	The site shall contain a section dedicated to creating an account. Once a user fills in the required username, email and password (encrypted when stored in database) information on the page and it is correctly formatted, it will load their newly created account within 5 seconds of them clicking the 'submit' button. Anyone with a unique email will be able to create an account with a storage limit of 200MB. Within this account they will upload Python files that they can later search for to show what it outputs.
2) Email verification	The site shall verify the input for the 'password' field in the create account section by checking if it follows the standard email format (contains '@' and '.').
3) Password recovery	In the case that a user forgets their password, they will click a link that will invoke an input box asking for their email where their password will be sent to.
4) Logging in	The page shall load within 4 seconds upon a user's 'log in' button click. A pop up will display missing/incorrect information or it will load their account page within 4 seconds if their information matched data in our database. A spinning cursor will indicate the page is still loading.
5) Upload Python file	The site shall allow users to upload Python files in an 'upload' section within their account page. Once a user fills in the title and description text boxes and selects a Python file, it shall upload within 5 seconds them clicking the 'upload' button. Otherwise, it will display in red text the missing/incorrect information.
6) Check for malicious code	The site shall only upload secure code. It will determine if the Python file is safe to upload by parsing it and checking for possible Python code injection it may contain.
7) Execute in emulator	The Python programs that are uploaded onto an account will be searched and executed by anyone who knows the username of a user's account. They can select a program to run that such user has uploaded. When they click on a program to run, the website will take them to an emulator page where they can input arguments in a text box at the top of the emulator. When the user hits the 'run' button, the program will start running. In the case of an error, there will be a pop up on the page that displays the error to the user.

2.2 Implementation Design

Program showcase will take input from users in order to create accounts. Once an account is created a user will have uploading privileges. As long as they choose a Python file (.py extension) and a parser checks it free of malicious code, it will be stored onto their account. Queries for usernames will make other users able to find a specific username to then run a Python program they've uploaded. Only users logged into their accounts will be able to view the actual code their files contained, others users will only be able to execute their program on the site's emulator. Once a user enters the execution page, depending on what the program selected requires, they will be able to modify arguments or interact with the program by inputting text. This process will repeat for each program a user executes.

2.3 Quality Assurance Plan

Risk #	Risk:	Risk Outcome:	Risk Remediation:
1	Code Injection	If the site does not handle malicious code injection, it could lead to manipulation or destruction of the database and site overall.	Parsing Python files before they uploaded can prevent from malicious actions. A check for specific Python code injection will keep the secure.
2	Website does not meet user needs	If the product does not meet customer needs, our product will become obsolete and development will have been wasted on an irrelevant product.	Continuous communication with the customer will allow both parties to be aware of any changes or concerns. The earlier changes in product requirements can be identified, the easier it will be to solve them.
3	AthenaHealth API or schema changes.	Changes in AthenaHealth's API or database schema may break our final product as our programmed database calls are no longer appropriate to pull the required data.	If we ensure the software is easy to update, any changes to the API or database schema could easily be applied to accommodate future development requirements.

3. Expected Project Results

Upon completion of the project, our client can expect the following results:

1. A user will be able to input information to create an account. The information will be saved onto a database so a user will also be able to log in.
2. Python files will be able to be uploaded and stored to a user account.
3. Other users will be able to search for usernames and execute on an emulator a program that a user has uploaded.

3.1 Measures of Success

Our project will be considered a success if the following criteria are met:

1. Account information is successfully saved in ordered to be queried from our database.
2. Users can upload Python files onto their accounts and search for these files when they want to share its output.

3. Python programs will be executed through an emulator that allows for active participation if the program is written to do so.

4. Schedule

Deliverable	**Due Date**
Interface for each page created.	4/18/2018
Ability to create accounts.	4/25/2018
Python files stored in user accounts.	5/15/2018
Programs executed through emulator	5/28/2018
Get approval from client	6/1/2018
Bugs worked out and project finalized.	12/8/2018
Final project handed off to client.	12/15/2018