Saint Augustine’s College, Sydney

**Saint Augustine’s College, Sydney**

**Software Engineering Year 11: Object Oriented Programming**

Example Project Documentation

Table of Contents

[Planning 3](#_Toc150638067)

[Task Definition 3](#_Toc150638068)

[Hardware and Software Requirements 4](#_Toc150638069)

[Storyboard 5](#_Toc150638070)

[Class Diagrams 6](#_Toc150638071)

[Data Dictionary 7](#_Toc150638072)

[Algorithm Design 8](#_Toc150638073)

[GANTT Chart 9](#_Toc150638074)

[Implementation 10](#_Toc150638075)

[GitHub Repository URL 10](#_Toc150638076)

[Testing 11](#_Toc150638077)

# Planning

## Task Definition

I have been assigned the task of developing a Python application to address a common cybersecurity issue: weak passwords. The application, named "LockSmithy", is designed to generate strong, secure passwords for users. The effectiveness of a password is often undermined by its simplicity and predictability, making it vulnerable to breaches. "LockSmithy" aims to mitigate this risk by creating passwords that are difficult to decipher.

The three simple, core functionalities of the application are as follows:

* Secure password generation: My tool will produce passwords that are a mix of alphanumeric characters and symbols to ensure complexity and strength.
* Customizable length: Users will have the option to select the length of their generated password, allowing for flexibility and adherence to different security requirements.
* Save to clipboard: Users can copy the generated password directly to their clipboard for easy use when creating or updating their accounts.

## Hardware and Software Requirements

In order to develop LockSmithy, my development environment will need to meet the following hardware and software requirements.

|  |  |
| --- | --- |
| Software | Requirements |
| Operating System | Any OS compatible with Python; Windows, macOS, or Linux |
| Python Version | Python 3.x |
| GooeyPie Framework | Latest compatible version with Python 3.x |
| Python Package Manager | PIP for installing GooeyPie and any other dependencies |
| IDE | Visual Studio Code |
| Version Control | Git for version control and GitHub for repository hosting |

|  |  |
| --- | --- |
| Hardware | Requirements |
| Processor | Modern CPU capable of efficient multitasking |
| Memory | 4GB RAM or higher for optimal performance |
| Storage | Minimum of 1GB free space for project and software |

## Storyboard

This storyboard presents a preliminary design of the user interface planned for development utilizing Python 3.x and the GooeyPie framework. The envisioned application is streamlined, featuring input fields for character count and password, along with a selection of checkboxes that allow for the inclusion of various character types in the password generation process. It will include a button to regenerate the password and another to copy it to the clipboard. The help screen is designed to offer concise information about the software, highlighting its principal functions.

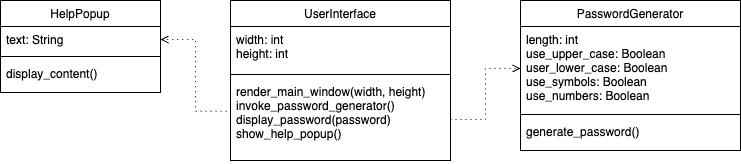
A screenshot of a computer

Description automatically generated *This diagram was generated using* [*.drawio*](https://www.drawio.com/)*, a link to the template can be found* [*here*](https://drive.google.com/file/d/1v-T5jTqqZkWZ4OXoLb7UGfcRfovvmecz/view?usp=sharing)*.*

## Class Diagrams

The application will be structured into three key classes:

* UserInterface: Manages GUI elements and interactions.
* PasswordGenerator: Handles password creation logic.
* HelpPopup: Controls help information display.



The UserInterface class depends on the PasswordGenerator class to generate a password. Furthermore, it depends on the HelpPopup class to provide help content to the user. These dependencies indicate that UserInterface does not function independently and requires the services of these two classes to fulfill its responsibilities.

## 

## Algorithm Design

The provided pseudocode delineates the method for the generatePassword method. It constitutes the program's most intricate algorithm, involving string operations and iterative constructs to assemble the password according to specified parameters.

#### generate\_password

BEGIN generate\_password(length, use\_lowercase, use\_uppercase, use\_numbers, use\_special)

SET characters = ''

SET password = ''

IF use\_lowercase THEN

APPEND 'abcdefghijklmnopqrstuvwxyz' TO characters

ENDIF

IF use\_uppercase THEN

APPEND 'ABCDEFGHIJKLMNOPQRSTUVWXYZ' TO characters

ENDIF

IF use\_numbers THEN

APPEND '0123456789' TO characters

ENDIF

IF use\_special THEN

APPEND '!@#$%^&\*()\_+-=[]{}|;:,.<>/?' TO characters

ENDIF

FOR i FROM 1 TO length

SET randomIndex = RANDOM INTEGER BETWEEN 1 AND LENGTH(characters)

APPEND characters[randomIndex] TO password

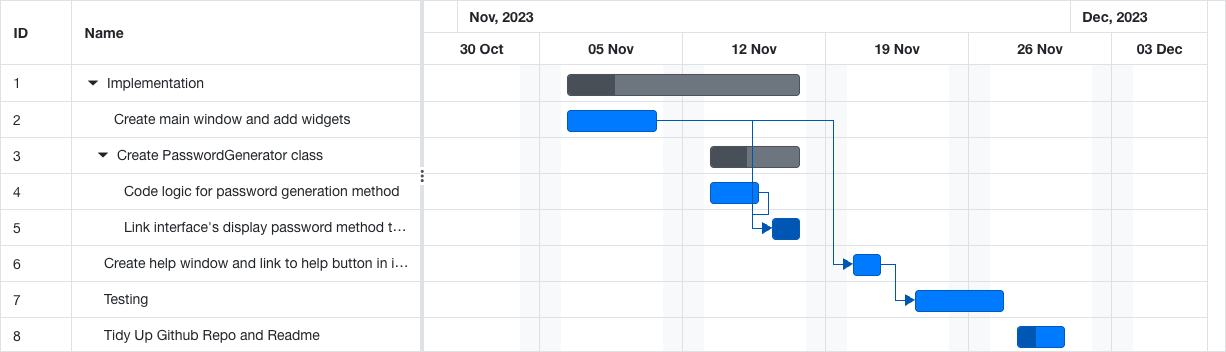
ENDFOR

RETURN password

END generate\_password

## GANTT Chart

The following GANTT chart was created at the beginning of my project and includes predicted timescales for each of the main tasks surrounding implementation, testing and release of my software.



*I created this GANTT chart using the free GANTT chart creator* [*https://www.onlinegantt.com/#/gantt*](https://www.onlinegantt.com/#/gantt)

# Implementation

## GitHub Repository URL

# Testing