Software Documentation for

SNUB

Southern New (Hampshire) University Database

CS114 Introduction to Software Engineering

Tim, Ben, Joe, Max

3/23/2023

Table of Contents

[Revision History 3](#_Toc35542547)

[1 Introduction 3](#_Toc35542548)

[1.1 Project Scope and Purpose 3](#_Toc35542549)

[1.2 Intended Audience 3](#_Toc35542550)

[2 Description 3](#_Toc35542551)

[2.1 Product Description 3](#_Toc35542552)

[2.2 Operating Environment 3](#_Toc35542553)

[2.3 User Classes 3](#_Toc35542554)

[2.3.1 Class 1 3](#_Toc35542555)

[2.3.2 Class 2 3](#_Toc35542556)

[2.3.3 Class 3 3](#_Toc35542557)

[3 Features 3](#_Toc35542558)

[3.1 Feature 1 3](#_Toc35542559)

[3.2 Feature 2 4](#_Toc35542560)

[3.3 Feature 3 4](#_Toc35542561)

[4 Interface 4](#_Toc35542562)

[4.1 User Interface 4](#_Toc35542563)

[4.2 Hardware Interface 4](#_Toc35542564)

[4.3 Software Interface 4](#_Toc35542565)

[5 Other Requirements 4](#_Toc35542566)

# Revision History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Responsible Person** | **Version** |
|  |  |  |  |  |
|  |  |  |  |  |

# 1 Introduction Ben

## 1.1 Project Scope and Purpose

<This is where you introduce the project briefly. A more detailed description will follow>

## 1.2 Intended Audience

<Who is intended audience. What purpose will this project serve.>

# 2 Description Joe

## 2.1 Product Description

SNUB, or the SNHU user database. Is a program that allows students and faculty to make notes, publish information/tips and keep up to date with the student body through an interactive, networked app.

## 2.2 Operating Environment

Because of our unique server-client model, we can develop a single monolithic server app, and have many different clients such as:

* Local winforms
* Local wpf
* Web
* Mobile

## 2.3 User Classes

## Server Side

On the server side we have many *endpoints.* Endpoints being networked locations where you can access or modify data, these include endpoints such as *[classes](https://github.com/KenwoodFox/SNUB/blob/0bc09f5cc8a7b65d68468260ae2ac1ef15a49a7e/Server/app/app.py" \l "L49)*, *[class\_notes](https://github.com/KenwoodFox/SNUB/blob/0bc09f5cc8a7b65d68468260ae2ac1ef15a49a7e/Server/app/app.py" \l "L56)* and *[version](https://github.com/KenwoodFox/SNUB/blob/0bc09f5cc8a7b65d68468260ae2ac1ef15a49a7e/Server/app/app.py" \l "L29).*

## Client Side

On the client sides we have many important classes and methods to decode and encode server endpoints, the most prevalent of witch are *[serverConnect](https://github.com/KenwoodFox/SNUB/blob/0bc09f5cc8a7b65d68468260ae2ac1ef15a49a7e/Client/serverConnect.cs" \l "L13)*, with methods like *[getValues](https://github.com/KenwoodFox/SNUB/blob/0bc09f5cc8a7b65d68468260ae2ac1ef15a49a7e/Client/serverConnect.cs" \l "L31)* and *[getClassNotes](https://github.com/KenwoodFox/SNUB/blob/0bc09f5cc8a7b65d68468260ae2ac1ef15a49a7e/Client/serverConnect.cs" \l "L51)*. As well as our [main class](https://github.com/KenwoodFox/SNUB/blob/0bc09f5cc8a7b65d68468260ae2ac1ef15a49a7e/Client/Program.cs" \l "L12).

# 3 Features

<Here you will talk about features of your software>

As of 3/23/2023 the software contains 4 main features. Users may check their connection to the webserver through a connect button. After initiating connection, the connection status and server version are displayed to the user. If the connection was successful users may select a class from a drop-down menu. After successfully selecting a class users may select a note to view from a list box. Notes are displayed in the format [Date - Author]. After successfully selecting a note from the list box the content of the note is displayed in a multiline text box. Users may also add notes. After successfully selecting a class users may input text into a text box and select the add button to upload their note to the server.

## 3.1 Connect to Network

Users must check network connection before any of the following features are available. After clicking the connect button a prompt is sent to the server to retrieve the current version. If the server returns a null value, the user is notified of the failed connection status. If the server returns a value, it is displayed to the user as the software version. This must be completed before using any other features.

## 3.2 Select Class

Users may select a class from the “class” drop down menu. When a class is selected, a prompt containing the class ID is sent to the server. If the connection is successful, the server returns a list of notes in the form [Date, Author, Note]. The returned value is parsed, and the Date and Author are displayed in a list box in the form [Date - Author].

## 3.3 Select and Read Notes

Users may select and read notes displayed in the list box. When a user selects an index in the list box a prompt containing the class ID and index ID is sent to the server. If the connection is successful the server returns a list of notes in the form [Date, Author, Note]. The returned value is parsed, and the content of the note is displayed in a multiline text box.

## 3.4 Add Notes

Users may input text in two separate textboxes, one corresponding to the author and one corresponding to the note. An add button can be used to upload the text box information to the server. If either box is null, the user is notified that null values were not uploaded to the server. If both boxes contain values then the data is formatted sent to the server, available immediately for use.

# 4 Interface

## 4.1 User Interface

The main interface of our program was developed using Visual Studio’s WinForms template, which allowed us to use buttons, textboxes, and menu strips to provide a simple and efficient interface which demonstrates the features of our project. In its infancy, our team is also developing a more pleasing interface using Visual Studio’s WPF formatting, which utilizes HTML and CSS to implement neat, user friendly visuals and a pleasing interface. The WPF interface is under development inside of a separate branch of our project and will be worked on and improved in the following weeks.

## 4.2 Hardware Interface

Our program will operate on any typical desktop system that is running Windows OS. It is possible to allow our project to be implemented into other hardware interfaces like smartphones, such as iPhone or Android, but will not be considered for the foreseeable future.

## 4.3 Software Interface

Unless we make the program executable in our future development, our interface will require the Windows operating system or an emulator of Windows. The interface is found within Visual Studio at the moment, in the form of a WinForms project.

# 5 User Document – How To

<These are essentially help files telling users how to use the software>

Our team’s project

# 6 Conclusions and Lessons Learned

<What have you learned from this project? What else can we include?>