Instructor: Rakeshkumar Shukla
Course: CSCI 5335-01E
23 March 2021

Requirements Elicitation
Better Notes

Team 01E 5: Accidental Consequences
Shane Casiano - Email: sc10196@georgiasouthern.edu; Eagle ID: 901014912
Jade Spahr - Email: Js21914@georgiasouthern.edu; Eagle ID: 901007178
Jacob French - Email: jf07042@georgiasouthern.edu; Eagle ID: 900965033
Alexis Jones - Email: aj06542@georgiasouthern.edu; Eagle ID: 900963115
Seth Aitcheson - Email: sa11637@georgiasouthern.edu; Eagle ID: 901194817

Table of Contents

1. Problem Statement
2. Objectives
3. Functional Requirements (Actors, Scenarios, Relationships)
    a. Actors
    b. Scenarios
    c. Relationships
    d. Requirements
    e. Use case diagram
4. Nonfunctional Requirements
    a. Nonfunctional requirements table
5. Target Environment / Build Architecture

1. Problem Statement

With the improvement of technology and the internet they have provided ways of assisting people with daily challenges in their lives. The abilities to set reminders for certain events, add alarms for certain times, being able to talk to people all over the country, and overall make the daily task of life seem less daunting.

However, regarding people with memory difficulty, such as older people or people who are mentally challenged, while technology has benefited these people in many different ways, it is still lacking certain assistance for their needs. Although we do have software for helping memory such as notepad software where, they can make notes to remember information, not all these notepads will remind these people of the notes they made, or have these notes in a way that is more than just words on a paper; that alone does not help with memory recovery.

2. Objectives

The objective of our BetterNotes project is to provide software in which users with memory difficulties can record information easier and have a way to remember that information in a convenient way. We will make a software that stores information into a notepad application and can insert the appropriate images that associate with the information given. We also plan to provide a reminder setting for people to set to remind the person that they made the notes in each format as well as a way that will allow users to store these notes in other formats such as PDF. We will also offer a feature that will allow users to use speech to text options for those who have difficulties typing and a text to speech option for those that may struggle to read the included font types. The primary goal is to create an application conducive to simplifying the lives of those that specifically struggle with memory capacity, specifically, those with memory loss issues. The program could provide structure and a “home base” of sorts that can be checked regularly, the program will also be capable of specifically aiding this group by sending reminders (if set by the user) for those who may still be adjusting to checking the program regularly.

3. Functional Requirements (Actors, Scenarios, Relationships)

3.a. Actors

[User] The person using the system.
[System] The device the user is using.

3.b. Scenarios

[Actor: user]

[OpenNote] [OpenReminder]

[Summary] Opens the selected note from the home page / file explorer.

[Flow of Events]

1. User either opens file from file explorer, or uses an OpenFileDialog to open a file from the program
2. Program will unarchive note and save it to metadata folder
3. Program will read metadata of note

[AddNote]

[Summary] Creates a new note by creating an object and placing metadata

[Flow of Events]

1. User clicks new
2. User inserts pertinent requested information
3. New note object and metadata is created

[AddReminder]

[Summary] Creates a new reminder by creating note object and placing metadata

[Flow of Events]

1. User creates new note
2. User provides note information
3. User selects reminder option
4. User provides reminder information
5. New note object and metadata is created

[SwitchToResourcesTab]

[Summary] Open the resources tab to the user's currently opened notes / reminder.

[Flow of Events]

1. User selects resources
2. User can now perform SearchImage, SearchVideo, SetReminder, STT and TTS

[SearchImage]

[Summary] Based on text the user provided in a textbox located in the image section of the resource tab, this will search for images and list their preview.

[Flow of Events]

1. User highlights a string in a textbox, or entered a search term manually in the resources tab
2. User can now InsertImage

[InsertImage]

[Summary] Insert the currently selected image to the current note.

[Flow of Events]

1. User clicks on image they want to insert into current cursor location on textbox
2. Image is downloaded and saved to note file
3. Image is inserted

[SearchVideo]

[Summary] Based on text the user provided in a textbox located in the video section of the resource tab, this will search for video and list their preview based on text the user provided in a textbox.

[Flow of Events]

1. User highlights a string in a textbox, or entered a search term manually in the resources tab
2. User can now InsetVideo

[InsertVideo]

[Summary] Insert the currently selected video to the current note.

[Flow of Events]

1. User clicks on video they want to insert into current cursor location on textbox
2. Image is embedded into the textbox

[TextToSpeech]

[Summary] Converts text from the textbox located in the TTS section of the resource tab to the current note.

[Flow of Events]

1. User highlights text or clicks text to speech in resources tab
2. User receives audio (either system or recorded) that user can play

[SpeechToText]

[Summary] Using the System's microphone, the detected audio will be converted to text and inserted to the current note.

[Flow of Events]

1. User clicks speech to text in resources tab
2. User clicks record
3. Recording is then processed and returned as string
4. String is inserted in user selected area

[SetReminder]

[Summary] Set the time for the reminder.

[Flow of Events]

1. User selects reminder option
2. User provides reminder information (time, date, type)
3. Metadata is edited to reflect changes

[SaveNote]

[Summary] Saves the current note, the associated reminder (if any), and the resources such as images and videos by archiving them into a format (.bnot), and the system will update the reminder data based on the note's reminder.

[Flow of Events]

1. User clicks save
2. SaveFileDialog opens requesting location of file (saves a dummy file)
3. The note is archived, resulting archive is save at the path retrieved from SaveFileDialog

[ConvertPDF]

[Summary] Converts the current note to a PDF file.

[Flow of Events]

1. User clicks convertPDF
2. Note calls virtual printer ("Microsoft Print to PDF")
3. User completes SaveFileDialog to save the PDF

[Actor: system]

[Alert]

[Summary] Send SMS / email or toast notification to the user

[Flow of Events]

1. System checks if it is time to remind
2. If it is time to remind:
    a. System checks what kind of notification to send from metadata (Toast, Email, Phone)
    b. System sends notification to requested zones
3. Else, system waits and repeats.

3.c. Relationship

For the scenario of Alert has many relationships to go through. For the user to create an alert sent there first has to be a note that a notification can be attached to. This would mean that a note has to be opened which includes opening saved notes and creating new notes. Once a note exists a reminder can then be added to it as an included functionality. The reminder that can be set to a note would then be saved or stored in the system. The system would then send out an alert when the stored notification time comes to be in the present.

Both notes and reminders will be able to be saved once a note is created which means there also needs to be a way to open them as well. With this being said, when the notepad is opened there will be an option to view stored reminders and saved notes. When dealing with an open note there are included functionalities of inserting videos and images into the note, and using outside resources on the note. The insert images and video functionality will extend to an outside resource to search for the image and/or video desired by the user. Additional resources that will be included are text-to-speech to read content, inserting text with speech-to-text, and converting the note into pdf format.

3.d. Requirements

- The user will be able to insert images into a note
- The user will be able to insert videos into a note
- The user will be able to convert selected text to speech
- The user will be able to convert speech to inline text
- The user will be able to convert a note to PDF
- The user will be able to set up reminders via Windows Toast Notifications
- The user will be able to set up reminders via Email
- The user will be able to set up reminders via text message.
- The user will be able to open and view recent notes
- The user will be able to insert returned text to speech to the document
- The user will be able to save the note to a chosen directory.
- The program will display relative short summaries to user of each note (creation date, other metadata values)
- The program will keep relative metadata (name/username)

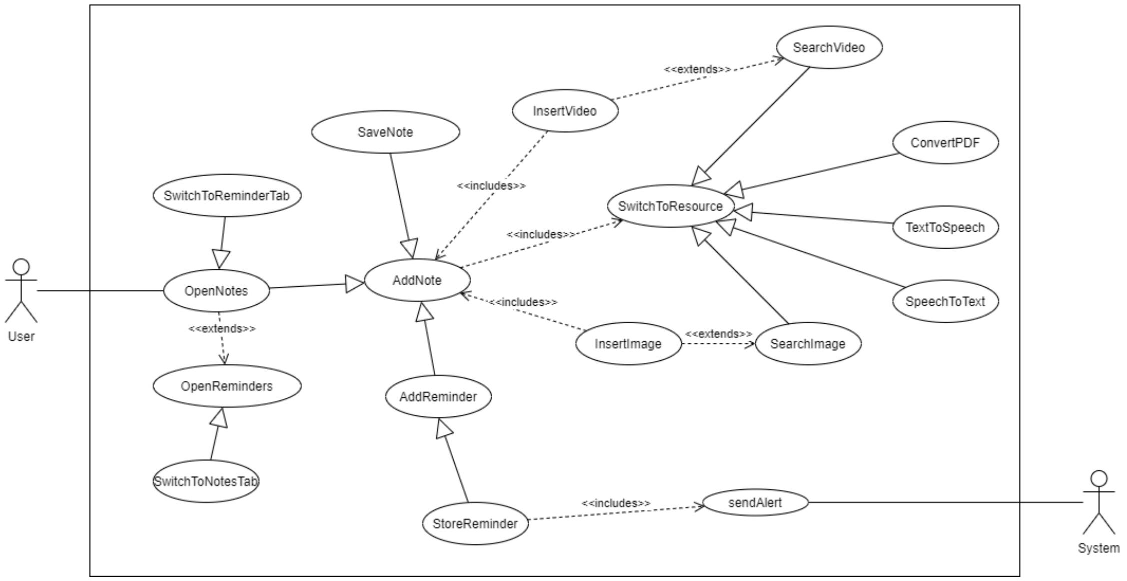

Figure 3.e: Use Case Diagram for proposed functionality of Better Notes

4. Nonfunctional Requirements

Category	Nonfunctional Requirement
Usability	● Program will display relative short summaries to user of each note (creation date, other metadata values)
Reliability	● Program will offer consistent recent items view ● Program will function on minimum Windows hardware requirements for the target environment
Performance	● Program will not interfere with operation of other programs on a users computer ● Program will clean up work areas (extracted notes, temp directories)
Implementation	● Program will be usable on any version of Windows 10 past target environment build
Operation	● Program will be usable by visually impaired (if user has appropriate software installed, not required for operation)
Integration	● Program will allow for reminders to be sent
Data	● Program will keep relative metadata (name/username)

Table 4.a: Table containing the nonfunctional requirements of BetterNotes

5. Target Environment / Build Architecture

Our target environment for Better Notes will be Windows 10 version 2004 and 20H2 (consumer editions). Our minimum required version will be: Windows 10 Build 14393 using .NET framework 4.7.2 as target. This program will be almost if not entirely created in C#, but may include some Batch or Python scripts depending on the needs of certain functions. For this, we would primarily use Visual Studio and Visual Studio Code (for Batch and Python scripting). Other included possibilities include PyCharm for more distinct Python scripting requiring class or function definition and detailed imports. The program will be designed as a WPF application and may have limited networking features to share information, this would likely run with existing infrastructure (such as email sharing) to prevent the need for implementation of costly network based features.