Emergency Backup

Mattia Carlino, Andrea Zenotto, Dario Bagnara

February 15, 2025

Contents

1 Project Overview

1.1 Description

The application, developed by the Group11, allows backing up a chosen folder to a USB device without using the presumed non-functional monitor, through specific figures drawn with the mouse. Specifically, the command is activated by drawing a rectangle that, starting from the top-left corner of the monitor, follows all four edges of the screen in full-screen mode. The user is guided through voice commands that suggest the confirmation and completion commands for the backup.

1.2 User manual & Main Features

- At the start of the application, by launching the emergency_backup file located in the emergency_backup/release/[target_OS] path, a configuration window will open. In this window, the folder to be backed up must be specified, along with, optionally, the file extensions of interest for the backup. This configuration will be saved, so it won't need to be entered each time the application is launched. At this point, when a backup needs to be performed, the user must move the mouse along the entire perimeter of the screen, starting from the top-left corner in a counterclockwise direction. Once this is done, a voice command will suggest drawing a horizontal line with the mouse to confirm. The backup will then start, and confirmation will be provided through the command panel window and a voice notification. It will then be possible to proceed with a new backup, even in the event of failure. If no external device is connected for the backup, the user will be informed through a voice notification, and the program will remain on standby.
- The application is designed to run in the background; therefore, once opened, it will not be possible to close the configuration GUI using the standard X button in the corner. This ensures its availability when needed.
- The application does not contain any malicious software. All configuration and backup data are protected by privacy regulations in compliance with the European GDPR standards.
- The application is portable, ensuring its functionality on macOS, Windows, and Linux.

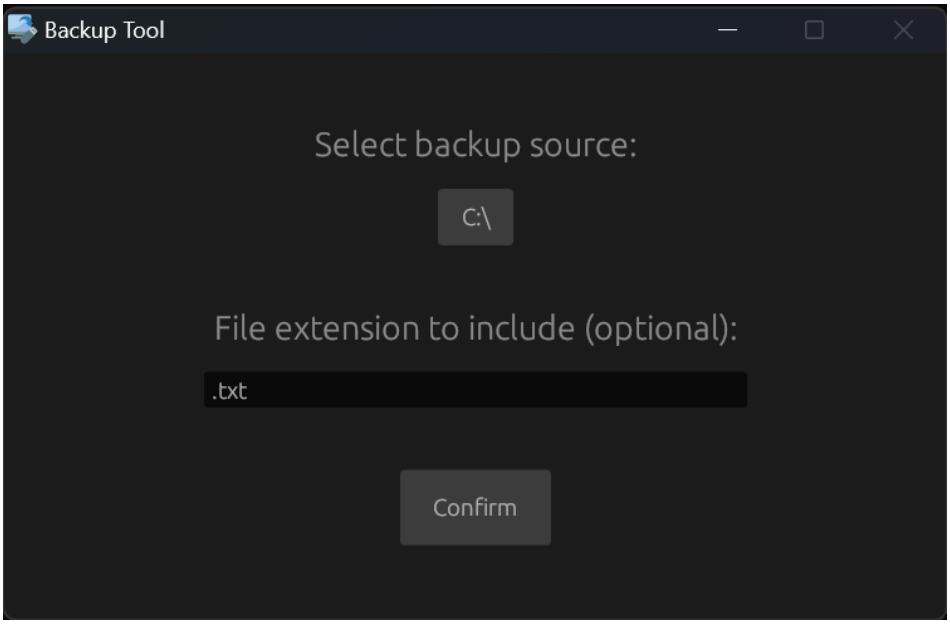

Figure 1: Configuration GUI

2 Structure

2.1 src

- main.rs: it is the application's entry point. It launches the configuration GUI and creates the thread that will handle mouse movements and initiate the backup.
- backup.rs: it searches for a USB device to perform the backup, discarding those that do not have enough space to contain the desired folder (the size of which is calculated). At this point, it recursively copies the relevant files to the found USB device, triggering appropriate errors in case any of the previous operations fail.
- event_manager.rs: It handles what follows a mouse movement: once the new coordinate is received, through an appropriate function (see input.rs), it triggers the command activation handlers, confirms the command (i.e., the backup), or continues sampling while waiting for a complete shape.
- gui.rs: in this file, the configuration window is created, through which the folder to perform the backup is defined. The new path and desired extension are then set at the graphical level and either loaded or saved.
- input.rs: it captures the new mouse coordinates, and through a logic, determines if the new point belongs to a known shape: in that case, it returns the completed shape or continues sampling at the event manager.
- lib.rs

2.2 Cargo.toml

- chrono: a library for date and time management in Rust.
- eframe: a framework for creating graphical user interface (GUI) applications in Rust
- egui: a library for creating graphical user interfaces. It's lightweight and focused on simplicity and responsiveness.
- epi: it manages the lifecycle of an application and provides a basic structure for initializing, updating, and displaying the application window.
- gui: custom library that handles the GUI of the application.
- image: a library for image manipulation and processing in Rust
- rdev: a library for interacting with input devices. It can be used to detect input events
- rfd: a library to simplify interaction with the file system, such as opening and saving files via dialog windows. It provides an interface for opening and selecting files in a cross-platform way.
- rodio: a library for audio processing in Rust
- serde: A library for serializing and deserializing data in Rust
- serde_json : An extension of serde specific to JSON handling.
- sysinfo : A library for collecting system information, such as CPU load, available memory, disks.

2.3 release

In the release folder, there are three separate directories, each containing the executables for MacOS, Windows, and Linux, along with the two folders required for images and audio tracks.

3 GUI

The first function called from main is represented by start_gui. This function configures the application window and launches MyApp through the eframe framework with the title "Backup Tool". MyApp, using the egui framework, through the update method and using the FileDialog component, allows the creation of a graphical window where the user can choose the folder of interest and any extensions, thus defining the state of the application, i.e., the user's configuration, called AppConfig.

4 Mouse Tracking

The goal is to capture the position of a point and determine if that position lies inside a specific area, and if so, detect when a shape is complete. To do this, the area is defined by the coordinates of the top-left and bottom-right corners, and it includes a contains method to check if a point is inside it. A tracker is defined, which has several states: InsideCurrentArea, InsideNextArea, ActivationShapeCompleted, ConfirmationShapeCompleted, and OutsideBoundaries.

The tracker contains an update method that is called every time the mouse moves, and a handle_shape_completion method, which together with the necessary checks, return one of the previous states. Finally, the generate_path function generates the path to be followed by the tracker: a path is defined by a set of rectangular areas, which, depending on the value of the shape parameter, can either be the entire screen frame (left vertical line, bottom horizontal, right vertical, and top horizontal) or a horizontal line in the middle of the screen, used as the confirmation command for the backup.

5 Backup

The main function is perform_backup(). This function loads the saved configuration, which includes the directory containing the file(s) to be backed up and the file extension. It then calculates the space needed to store the backup using the get_dir_size function, and searches for a USB device capable of holding this backup. Finally, using the copy_dir_recursive function, it copies all files, recursively if a subfolder is found. The saved folder is renamed by adding a timestamp of when the backup is performed.

6 Audio

Audio files are played by the play_sound function. This function creates a sink, which is an object that handles audio playback, and, depending on whether the application is running in debug or release mode, it searches for the file in a specific directory. The file is then opened and decoded to be played by the previously created Sink, waiting for its completion through the sleep_until_end function.