# Teilnehmer/innen des Teams:

|  |  |
| --- | --- |
| Klasse:  BI 22a | Team:  Rino & Aleks |

# Anforderungsdefinition (Meilenstein A)

|  |  |
| --- | --- |
| „Prozessverwalter“ | |
| **Fachlicher Inhalt:**  (Allgemeine Beschreibung) | Nutzen: Mit dem Programm sollen Prozesse gesucht werden können und abhängig, ob sie bereits am laufen sind oder nicht kann man den Prozess starten und killen.**Automation:**  Das Programm automatisiert den Startup oder kill Prozess von Prozessen die man  auswählt (e.g. Suchfeld)  **Details:**   * Konfiguration: keine * Integration: Starten per PowerShell * Administration: bei der "kill-Process" Funktion kommt ein Fail Safe im Sinn von  "Sind Sie sicher?" Fenster * Sicherheitsaspekte: nein   (Grid mit "Listbox" ausführen)  (Skizze / Mockup)    **Erkenntnisse aus der Machbarkeitsabklärung in Windows Powershell:**  Folgende Features sind vorab untersucht worden und das Suchfenster für die verschiedenen Prozesse mit der Auswahl werden wahrscheinlich die grössten Herausforderungen sein. |
| **MUSS**  **Kriterien:**  (Konkrete Features, die umzusetzen sind) | **Folgende Features sollen implementiert werden, um einen produktiven Ablauf sicherzustellen:**   * Grid Fenster mit Auswahlfunktion * Kill/Restart Button * Quit-Button |

|  |  |
| --- | --- |
| **KANN**  **Kriterien:**  (Konkrete Features, die optional sind) | **Folgende Features können zusätzlich implementiert werden: (Varianten, Kreativität)**   * Bei der Kill Option ein Fail Safe in Form von einem extra Formular, wo man für Bestätigung gefragt wird. * Bestätigung das der Prozess erfolgreich gestartet/Gekillt wurde oder nicht * Listbox Inhalt ist sortiert und schön formatiert |

## Planung Meilensteine (LB1 / LB2)

|  |  |  |  |
| --- | --- | --- | --- |
| *MS* | *Tätigkeit / Abgabe* | *Soll-Datum* | *Ist-Datum* |
| A | ProjektstartTeam Bildung, Kollaborationsplattform, GitHub Repos \*, LehrerzugangWahl / Ausarbeitung der Anforderungsdefinition Abnahme Anforderungsdefinition durch Lehrperson | 09.01.2024 | 09.01.2024 |
| B | Teamaufgabe 1:Abgabe: Lösungsdesign  (Funktionsmodell / GUI / PAP / Storyboard) | 09.01.2024 | 09.01.2024 |
| B2 | Teamaufgabe 2: (Nur LB2)Abgabe: Testvorschrift und Testfälle | 30.01.2024 | 30.01.2024 |
| C | Einzelaufgabe 2 (LB1) / 3 (LB2):Abgabe Programmcode und DokumentationFachgespräch Projektabnahme | 30.01.2024 | 09.01.2024 |
| C2 | Einzelaufgabe 4: (Nur LB2)Abgabe: Ausgefüllter Systemtest | 30.01.2024 12:00 | 30.01.2024 11:47 |

\*) Öffentliche GitHub-URLs im Ablageordner auf dem BSCW ablegen! (pro Team)

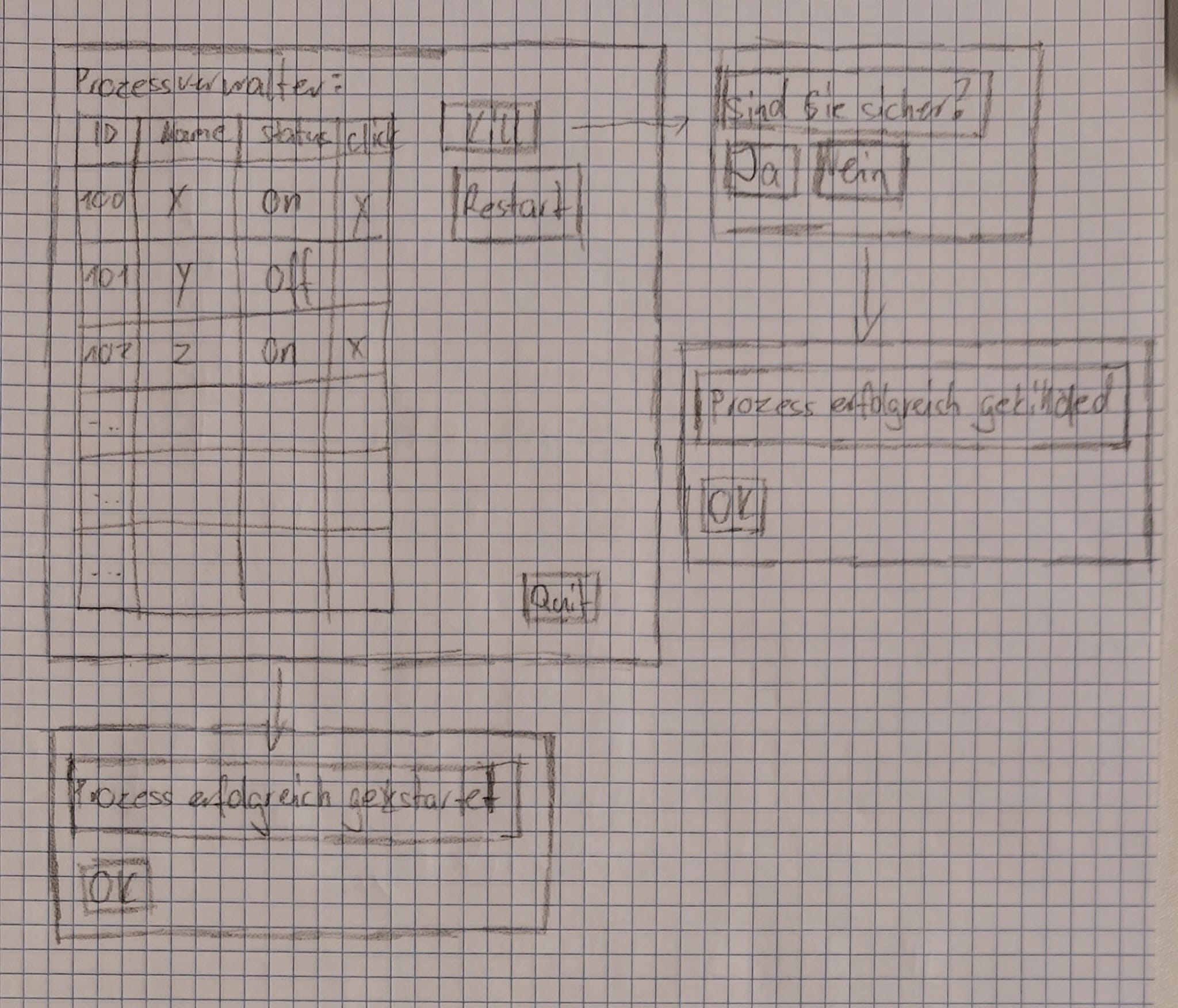
Namenskonvention URL: **M122\_Klasse\_Thema\_Name\_Name**

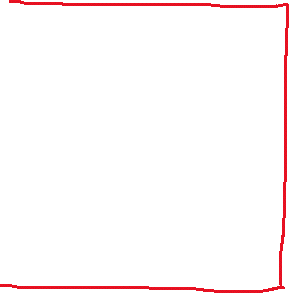
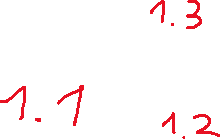
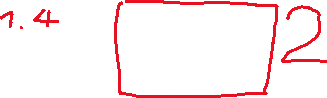
# Lösungsdesign (Meilenstein B: Teamaufgabe 1)

Anhand der Analyse wurde folgendes Lösungsdesign entworfen:

## Schematische Darstellung der Funktionalität, sog. Funktionsmodell

Im Folgenden ist die erwartete Funktionalität dargestellt und erklärt:

  
  
Legende:

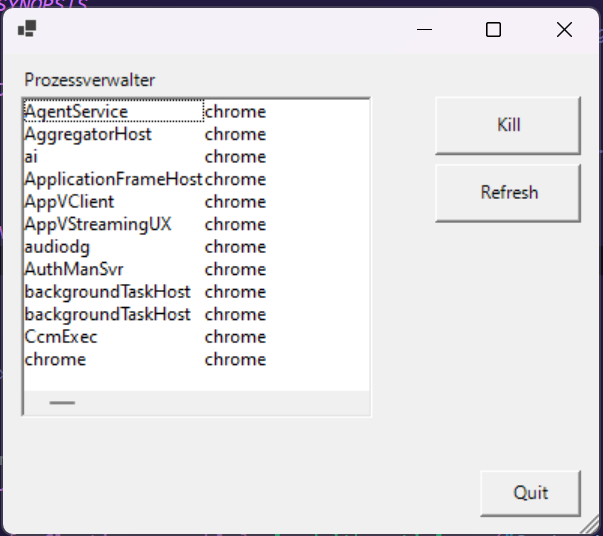


1. Hauptfenseter mit 3 geplanten Buttons und somit Funktionen.   
   1.1 Listbox mit allen Prozessen die laufen. Clickbar sortiert.  
   1.2 Quit-Button, der es ermöglicht das Programm zu schliessen  
   1.3 Restart-Button der nicht laufende Prozesse "restartet"  
   1.4 Kill-Button der laufende Prozesse beendet
2. "Confirmation"-Fenster, aufrage zum bestätigen  
   2.1 Nein-Button, zum den Kill Prozess aufhalten  
   2.2 Ja-Button, bestätigen und somit den Prozess force killen
3. Bestätigungsfenster  
   3.1 OK-Button zum Ablauf abschliessen und zurück zum Skript kommen.
4. Bestätigungs-Meldung, dass der Prozess erfolgreich restartet, wurde  
   4.1 OK-Button zum Ablauf abschliessen und zurück zum Skript kommen

(**Funktionsmodell**: Skizze, Bild, Pictogramm, Mindmap, Blockdiagramm, UseCase (API) zur obigen Anforderungsdefinition **mit Legende**)

## Graphische Benutzer Schnittstelle (GUI) zur Konfiguration des Ablaufs

Das zu erwartende GUI ist hier dargestellt:





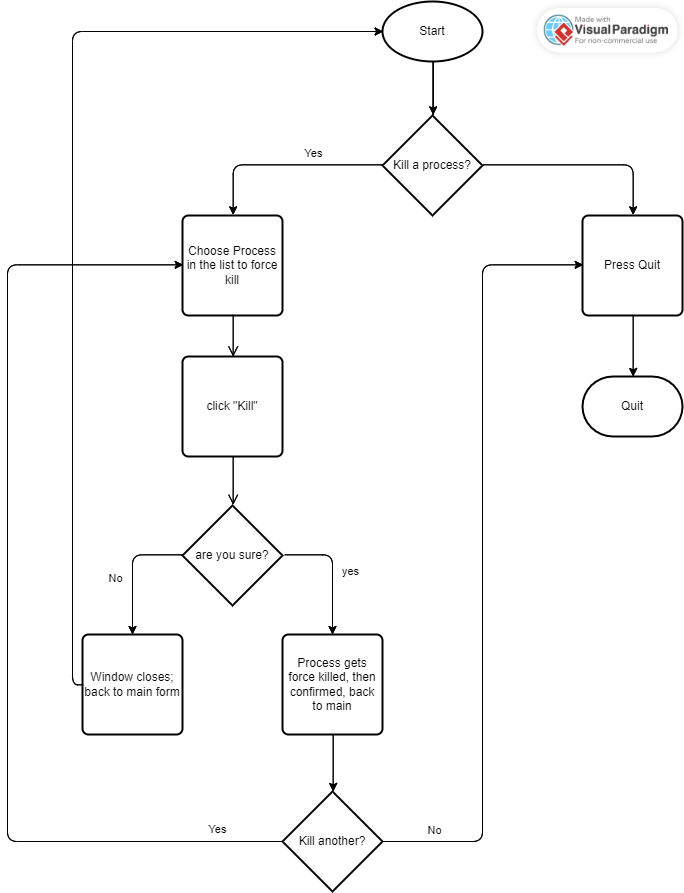
1. Kill Button zum "killen" vom ausgewählten Prozess (force kill)
2. Refresh Button, um die Liste zu refreshen (nach z.B. erfolgreichem Killen von einem Prozess)
3. Ausgewählter Prozess in der Prozessliste (Textbox)
4. Prozessliste mit allen laufenden Prozessen auf dem PC
5. Quit Button um das Programm beenden

## Ablauf der Automation

Aus Benutzersicht ist folgender Ablauf des Programms zu erwarten:

Man wählt zuerst den Prozess aus den man Beenden will. Sobald man diesen Prozess dann in der Liste ausgewählt hat, auch "kill" klicken. Beim Bestätigung Fenster auf "Ja" klicken, wenn man fortfahren möchte und sich sicher ist, den gewählten Prozess zu beenden. Ansonsten auf "nein" drücken und zurück zum Prozessmanager kehren. Sobald der Prozess erfolgreich beendet wurde, erscheint ein Bestätigungsfenster, dort auf "OK" drücken und somit ist der Prozessbeendet. Falls man nochmals einen Prozess beenden möchte, auf "Refresh" klicken, um die Liste zu aktualisieren und dann diesen Prozess wiederholen.

Aus Administratorsicht ist folgender Ablauf des Programms zu erwarten:



(Flussdiagramm (PAP / APIs: UML Aktivitätsdiagram) / Storyboard)

# Testvorschrift (LB2 Meilenstein B2: Teamaufgabe 2)

Testbeschrieb und vorbereitetes Testprotokoll siehe Dokument   
***M122\_LB2\_Testvorschrift\_MS-B2\_Namen.docx***

# Testprotokoll (LB2 Meilenstein C2: individuelle Aufgabe 4)

Ausgefülltes Testprotokoll siehe Dokument   
***M122\_LB2\_Testvorschrift\_MS-C2\_Namen.docx***

# Systemdokumentation (Meilenstein C: individuelle Aufgabe 3)

Die erstellten Projekt-WPS-Scripts sind hier abgelegt und für Entwickler dokumentiert:

Öffentliche GitHub-URLs im Ablageordner auf dem BSCW ablegen! (pro Team)

Namenskonvention URL: **M122\_Klasse\_Thema\_Name\_Name**

*à Ein* ***Branch*** *und separater Doku-Ordner pro Teammitglied erstellen*

## Umfang / Abgrenzung / Änderungen gegenüber Design

Aufgrund unten beschriebener Umstände sind Anpassungen des ursprünglichen Lösungsdesigns gemacht worden:

Da ein "Restart" Botton nicht ersichtlich war, wurde dieser Button mit einem "Refresh" Button ersetzt. Somit besteht die Möglichkeit einen Prozess aus dem Skript-Fenster her zu restarten nicht. Jedoch besteht nun die neue Funktion nach einem Erfolgreichen Kill die Prozessliste zu aktualisieren.

Umstände / Anpassungen / Veränderungen

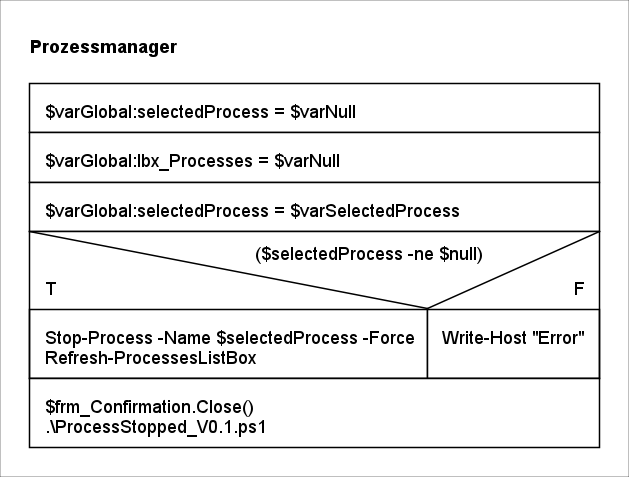
## Funktionalität der Implementation.

Zusätzlich zu der Inline-Dokumentation sind hier folgende Funktionen / Eventhandler detailliert beschrieben:

Beischreibung im Inlinecode der jeweiligen Skripten.

Ausführliche Beschreibung der internen Funktionen (Eventhandler wie z.B: $btn\_ok\_Click), der Parameter und der Rückgabewerte

Die wichtigsten Variablen und Funktion ist so strukturiert:



**Struktogramm** (BET / SYS) oder **UML-Aktivitätsdiagramm** (API)

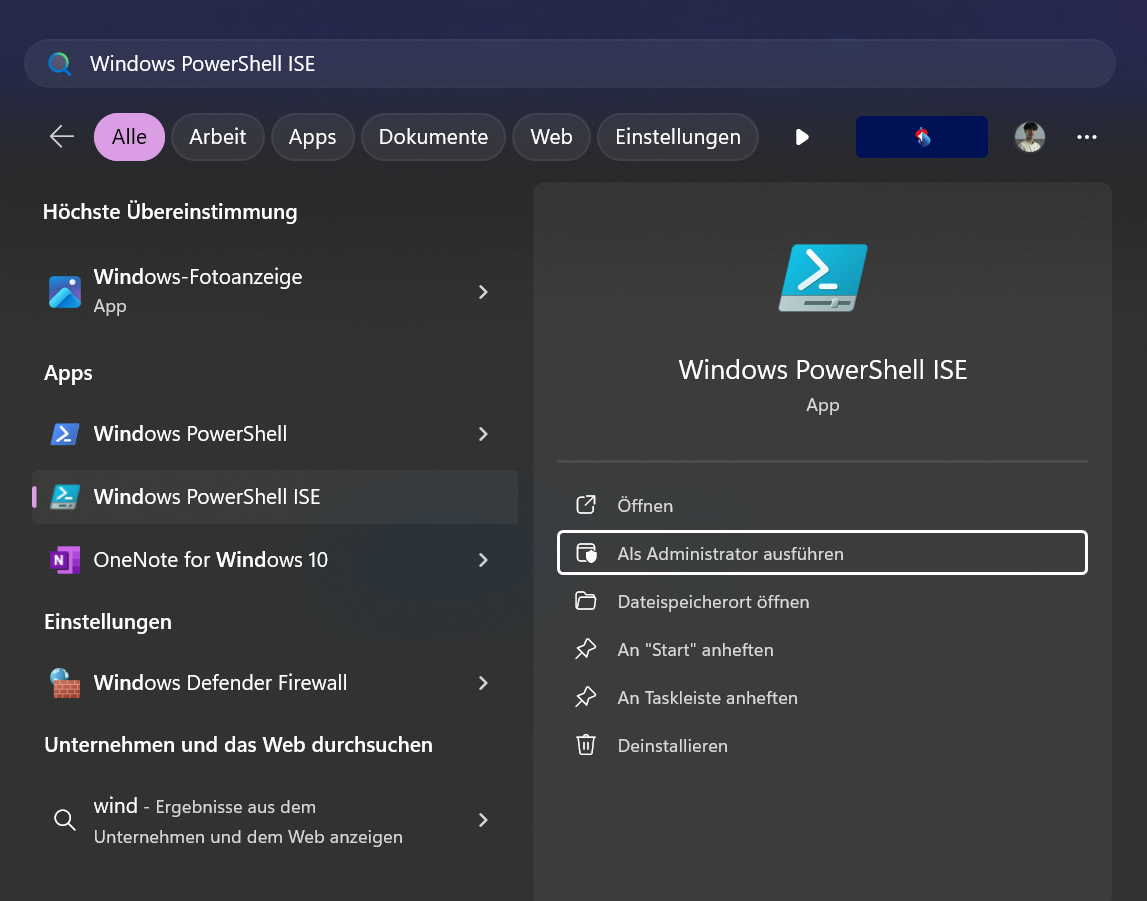
# Betriebsdokumentation (Meilenstein C: individuelle Aufgabe 3)

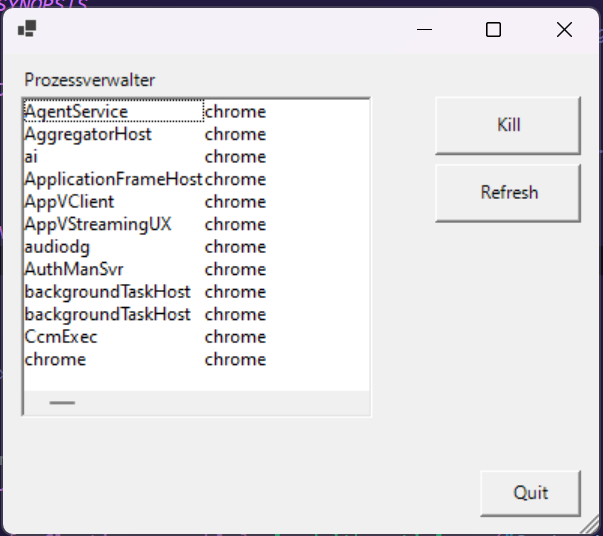
Für Administrator und Benutzer wird folgende Anleitung ausgeliefert

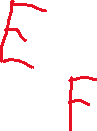
## Installationsanleitung für Administratoren

Skript herunterladen. Im Directory die ZIP-Datei entpacken. Ordner am besten per VS-Code oder ähnlicher Applikation öffnen und dann die Datei "Processmanager" öffnen. Theoretisch möglich die Datei als z.B. Batch Datei zu speichern und so diese Datei auch ohne VS-Code zu öffnen.

## Bedienungsanleitung für Benutzer

Das Programm ist folgendermassen zu bedienen ...  
  
  
1. Das Skript sollte am besten als Admin gestartet werden  






Legende:

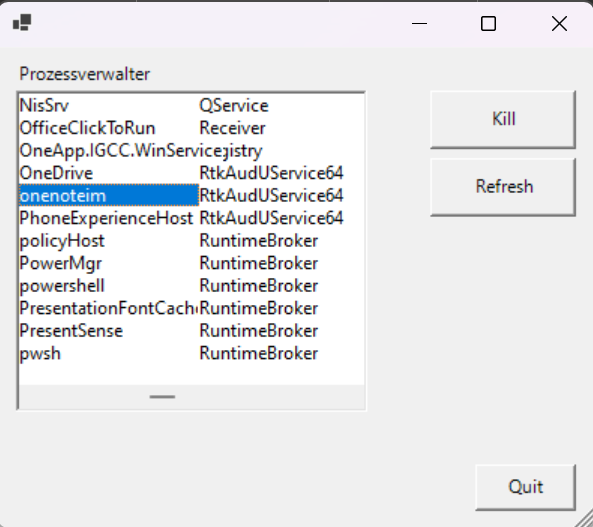
A. Möglichkeiten für folgende Funktionen (von links nach rechts):  
Minimieren, Vollbild, Schliessen  
B. Kill Button zum "killen" vom ausgewählten Prozess (force kill)

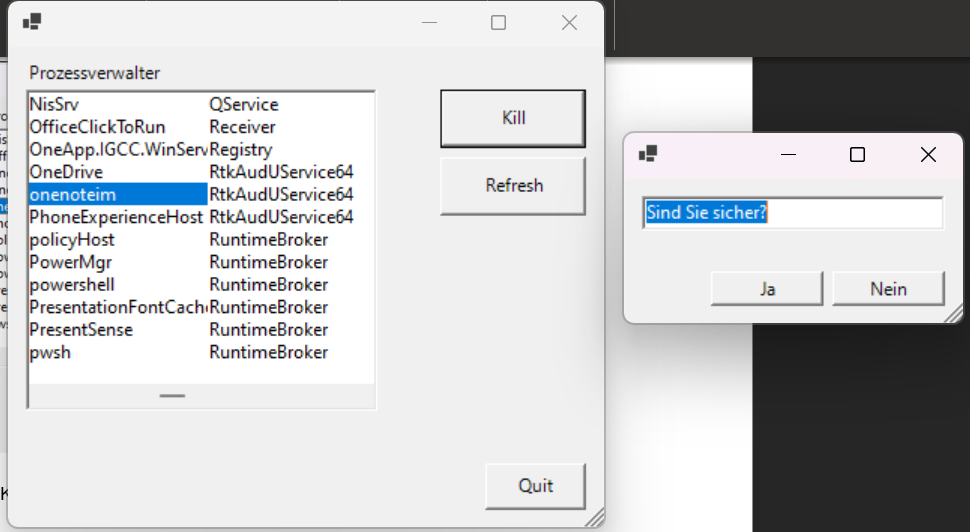
C. Refresh Button, um die Liste zu refreshen (nach z.B. erfolgreichem Killen von einem Prozess)

D. Ausgewählter Prozess in der Prozessliste (Textbox)

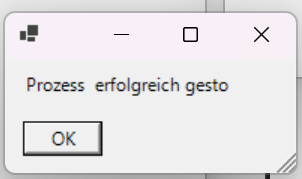
E. Prozessliste mit allen laufenden Prozessen auf dem PC

F. Quit Button um das Programm beenden

2. Prozess auch der Prozessliste auswählen  


3. "Kill" klicken  


4. Auf "ja" klicken



5. Prozess wurde erfolgreich beendet