# Versiune pentru public – paragrafe clare

## 2.2 Funcționalități Principale

Sistemul urmărește continuu procesele importante din server și detectează rapid când ceva nu merge bine. Când găsește o problemă, încearcă automat să repornească serviciul afectat, folosind reguli configurate în prealabil. Pentru a evita reporniri în buclă, are un mecanism de protecție care oprește temporar încercările, iar pentru fiecare proces se pot defini verificări de sănătate după repornire. Toate acțiunile sunt înregistrate în fișiere de jurnal, iar administratorii pot lucra din linia de comandă.

## 3.1 Considerații de Securitate

Datele de acces la baza de date sunt păstrate în fișiere temporare și se folosesc timp limitat, iar conexiunile expiră dacă durează prea mult. Contul din baza de date are drepturi minime, strict cât are nevoie. Evenimentele importante sunt raportate în sistemul de jurnalizare al serverului.

## 3.2 Considerații de Performanță

Conexiunile la baza de date sunt optimizate pentru a răspunde repede și a consuma puține resurse. Mecanismul de protecție împotriva repornirilor repetate previne supraîncărcarea. Ritmul verificărilor poate fi ajustat, iar rotația automată a jurnalelor menține sistemul agil.

## 3.3 Considerații de Scalabilitate

Fiecare proces poate fi configurat separat, astfel încât sistemul se adaptează ușor la nevoi diferite. Există mai multe moduri de repornire și o structură modulară care permite adăugarea de procese și tipuri noi pe viitor.

### 4.1.1 Monitorizare Continuă

Sistemul verifică la intervale configurabile dacă procesele esențiale funcționează normal. Când un proces intră în stare de alarmă, evenimentul este detectat automat, indiferent de tipul serviciului monitorizat.

### 4.1.2 Restart Automat

Repornirea poate fi făcută în diferite moduri: prin serviciul de sistem, direct pe proces sau prin comenzi personalizate. Înainte de repornire se pot rula comenzi pregătitoare, iar după repornire se verifică automat dacă totul este din nou funcțional, cu posibilitatea de a reîncerca de câteva ori.

### 4.1.3 Circuit Breaker Pattern

Dacă repornirile eșuează de prea multe ori, sistemul intră într-o stare de protecție și oprește încercările o perioadă. Pragurile și timpii pot fi configurați, iar stările sunt înregistrate pentru analiză.

### 4.1.4 Logging și Monitorizare

Jurnalele se rotesc automat când devin mari, iar evenimentele critice se trimit și către sistemul de jurnalizare al serverului. Nivelul de detaliu poate fi setat, de la diagnostic detaliat la mesaje de eroare.

### 4.2.1 Performanță

Sistemul răspunde rapid la probleme și folosește eficient resursele. Conexiunile către baza de date sunt gestionate astfel încât să nu încetinească funcționarea.

### 4.2.2 Fiabilitate

Funcționează fără a necesita supraveghere constantă. Erorile sunt tratate corect, iar după o problemă, sistemul se recuperează singur acolo unde este posibil.

### 4.2.3 Mentenabilitate

Configuratorul pe fișiere INI face sistemul ușor de adaptat. Jurnalele detaliate ajută la depanare, iar documentația oferă ghidaj clar.

### 4.2.4 Securitate

Datele sensibile sunt tratate cu atenție, configurațiile sunt verificate la pornire, iar accesul este limitat la strictul necesar.

### 4.3.1 Baza de Date

Sistemul folosește MySQL/MariaDB pentru a salva stările proceselor. Structura tabelelor este indexată pentru căutări rapide și performanță constantă.

### 4.3.2 Procese și Servicii

Fiecare proces are setările lui: cum se repornește, cât se așteaptă, ce verificări se fac după și ce limite de timp se aplică. Astfel, comportamentul e precis și previzibil.

### 4.3.3 Logging

Putem stabili cât de mari pot deveni fișierele de jurnal și câte copii de rezervă păstrăm, pentru a controla spațiul și claritatea istoricului.

### 5.1 Structura Codului

Programul are o buclă principală care monitorizează procesele, funcții care citesc alarmele din baza de date, repornesc procesele după reguli, verifică starea lor post-restart și aplică mecanismul de protecție. Configurațiile sunt citite și validate la pornire, iar jurnalizarea are funcții dedicate pentru scriere și rotație.

### 5.2 Strategiile de Restart

Repornirea la nivel de serviciu folosește unelte standard ale sistemului pentru a verifica și restarta serviciile. Repornirea la nivel de proces se face direct, identificând și oprind procesele problematice. Pentru cazuri speciale, se pot defini comenzi personalizate, inclusiv pași de verificare.

### 5.3 Circuit Breaker

Mecanismul are trei stări: normal, blocat și reset. În funcție de câte eșecuri la repornire apar și în cât timp, sistemul decide dacă să continue, să oprească încercările sau să reîncerce după o pauză.

### 6.1 Scriptul de Testare

Există un script care simulează alarme și acoperă toate metodele de repornire, inclusiv mecanismul de protecție, pentru a verifica că totul funcționează corect.

### 6.2 Scenarii de Testare

Se testează situații reale: procese care cad, reporniri care eșuează repetat, activarea protecției și rotația jurnalelor, pentru a asigura stabilitatea sistemului.

### 7.1 Instalarea Sistemului

Fișierele sunt copiate într-o locație standard a serverului, baza de date se pregătește cu un script, iar serviciul este înregistrat în sistem pentru a porni corect.

### 7.2 Configurarea Inițială

Se completează fișierul de configurare cu valorile potrivite mediului, se adaugă credențialele bazei de date și lista proceselor de urmărit.

### 7.3 Operaționalizare

Serviciul se pornește prin comanda specifică a sistemului, se urmăresc jurnalele și se poate seta pornirea automată la fiecare restart al serverului.

### 8.1 Logurile Sistemului

Jurnalul principal se află într-un fișier dedicat, iar la evenimente importante se raportează și în jurnalul sistemului. Rotația automată menține fișierele curate și mici.

### 8.2 Comenzi de Administrare

Administratorii pot verifica starea, pot forța o repornire atunci când e necesar și pot accesa rapid ajutorul pentru utilizare.

### 8.3 Mentenanța Configurației

Pe măsură ce apar procese noi sau cerințe noi, configurațiile se pot actualiza ușor pentru a ține pasul cu schimbările.

### 9.1 Automatizarea Completă

Intervențiile manuale sunt rare, timpul de reacție la probleme este scurt, iar serviciile rămân disponibile mai mult timp.

### 9.2 Flexibilitatea Configurării

Sistemul se adaptează ușor la tipuri diferite de procese și nevoi, datorită strategiilor și verificărilor configurabile.

### 9.3 Robustețea Sistemului

Mecanismele de protecție, tratarea corectă a erorilor și recuperarea automată fac sistemul stabil în fața problemelor.

### 9.4 Scalabilitatea

Se pot adăuga mai multe procese sau cerințe fără schimbări majore, fiecare cu setările lui.

### 10.1 Limitări Tehnice

Sistemul depinde de MySQL/MariaDB și, pentru reporniri, are nevoie uneori de drepturi de administrator. De asemenea, opțiunile de repornire sunt limitate la cele definite.

### 10.2 Considerații de Securitate

Folosirea fișierelor de configurare pentru credențiale cere atenție, iar anumite operații necesită permisiuni ridicate. În jurnale pot apărea informații sensibile dacă nu sunt filtrate corect.

### 10.3 Considerații de Performanță

Verificările frecvente către baza de date și numărul mare de procese pot încărca sistemul, dar setările corecte atenuează aceste efecte.

### 11.1 Îmbunătățiri Planificate

Pe viitor, se dorește suport pentru mai multe baze de date, o interfață web, alerte prin email sau SMS și integrare cu alte sisteme de monitorizare.

### 11.2 Extensii Posibile

Se are în vedere suportul pentru containere, monitorizarea resurselor, integrarea cu orchestratoare și operarea pe mai multe servere.

### 12.1 Realizările Proiectului

A fost construit un sistem complet care monitorizează, repornește inteligent, se protejează de erori repetate, ține jurnale clare și se configurează ușor.

### 12.2 Beneficiile Implementării

Timpul de nefuncționare scade mult, munca manuală se reduce, infrastructura devine mai fiabilă, costurile scad și încrederea în sistem crește.

### 12.3 Valoarea Adăugată

Soluția automatizează remedierile, folosește bune practici din domeniu, oferă bază solidă pentru extindere și este o alternativă accesibilă la produse comerciale.