|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | | | |  | |
|  | escola  superior  de tecnologia  e gestão  POLITÉCNICO  DO PORTO | | | | |  | |
|  | | | LEI |  |  | | |
| lICENCIATURA Engenharia INFORMÁTICA  SISTEMAS OPERATIVOS | |  |
|  | | | Trabalho Prático  Grupo A  José Magalhães – 8210125  Miguel Carvalho – 8210130  01/2024 | | |  |
|
|
|  | | | | | | | |

Índice

[Índice 3](#_Toc154853349)

[Índice de Figuras 4](#_Toc154853350)

[1 Introdução 5](#_Toc154853351)

[1.1 Âmbito / Contextualização 5](#_Toc154853352)

[1.2 Objetivos 5](#_Toc154853353)

[1.3 Estrutura do documento 6](#_Toc154853354)

[2 Manual de compilação, configuração e utilização da aplicação 7](#_Toc154853355)

[3 Descrição das funcionalidades implementadas 8](#_Toc154853356)

[3.1 Módulo Main 8](#_Toc154853357)

[4 Descrição e justificação da utilização de mecanismos de sincronização e comunicação entre módulos 9](#_Toc154853358)

[4.1 Módulo Main 9](#_Toc154853359)

[5 Funcionalidades adicionais 10](#_Toc154853360)

[5.1 10](#_Toc154853361)

Índice de Figuras

1. Introdução

O desenvolvimento de sistemas operativos eficientes e robustos é essencial para uma gestão eficaz dos recursos computacionais em ambientes críticos, como é o caso de satélites. Este relatório documenta o trabalho prático realizado na unidade curricular de Sistemas Operativos, cujo objetivo é simular um sistema operativo para um satélite, incorporando conceitos de multiprocessamento, comunicação e sincronização.

* 1. Âmbito / Contextualização

No âmbito desta disciplina, o trabalho proposto abrange a simulação de um sistema operativo de um satélite que gere duas unidades de computação, o CPU e a MEM, com ênfase na sincronização e comunicação entre os diferentes componentes. O propósito desta simulação está na necessidade de assegurar que os sistemas computacionais em satélites satisfaçam requisitos críticos, tais como segurança, robustez, fiabilidade e a capacidade de comunicação em tempo real. Ao enfrentar estes desafios, tivemos a oportunidade de aplicar conceitos teóricos adquiridos durante aulas práticas.

* 1. Objetivos

O principal objetivo deste projeto foi desenvolver um simulador de sistema operativo para um satélite, utilizando linguagem Java e aplicando conceitos já mencionados. O trabalho visa proporcionar uma compreensão prática da matéria abordada na unidade curricular de Sistemas Operativos, focando-se em situações reais de festão de recursos computacionais em tempo real.

* 1. Estrutura do documento

Este relatório é composto por vários capítulos e subcapítulos de modo a facilitar a leitura e a interpretação dos conteúdos. A estrutura compreende os seguintes capítulos:

* **Introdução** – Este capítulo apresenta uma visão geral do trabalho.
* **Manual de compilação, configuração e utilização da aplicação** – Neste capítulo serão fornecidas orientações detalhadas sobre como compilar, configurar e utilizar a aplicação desenvolvida.
* **Descrição das funcionalidades implementadas** – Aqui, será realizada uma análise detalhada das funcionalidades desenvolvidas no sistema operativo. Cada componente, destacando as suas responsabilidades e contribuições para o bom funcionamento da aplicação.
* **Descrição e justificação da utilização de mecanismos de sincronização e comunicação entre módulos** – Este capítulo explora os mecanismos de sincronização e comunicação implementados no projeto. Serão identificados e explicados os métodos escolhidos para garantir a coordenação eficiente entre as diferentes partes do sistema.
* **Funcionalidades adicionais** – Neste último capítulo, serão apresentadas as funcionalidades adicionais propostas e implementadas para enriquecer o sistema operativo.

1. Manual de compilação, configuração e utilização da aplicação

Este trabalho foi desenvolvido na linguagem *JAVA* e utilizado o *Visual Studio Code*.

Recomendamos a instalação e configuração prévia dessas ferramentas para facilitar a utilização da aplicação. O código-fonte completo está disponível no repositório do GitHub “<https://github.com/SOGrupoA/TP_SO>”. Qualquer documentação adicional ou informações relevantes também podem ser encontradas no mesmo repositório.

Para compilar e executar o projeto, é necessário seguir os seguintes passos:

1. Abrir o terminal na pasta do “projeto”.
2. Executar os seguintes comandos, um de cada vez:
   * “javac -cp ./lib/json.jar;./lib/jfreechart.jar;./lib/jcommon.jar -d ./classes -encoding UTF-8 \*.java”
   * “java -cp .;./lib/json.jar;./lib/jfreechart.jar;./lib/jcommon.jar;./classes Kernel”

**NO FIM, COLOCAR EM PDF**

Estes passos garantem a compilação e execução adequadas do projeto.

Outra opção, em vez de executar os comandos manualmente, é simplesmente abrir o arquivo de configuração “*start.bat*”. Este arquivo já contém os comandos necessários para compilar e executar o projeto de forma automatizada. **\*\*\*\*\***

**MULTIPROCESSAMENTO, MULTIPLAS THREADS**

1. Descrição das funcionalidades implementadas
   1. Módulo Main

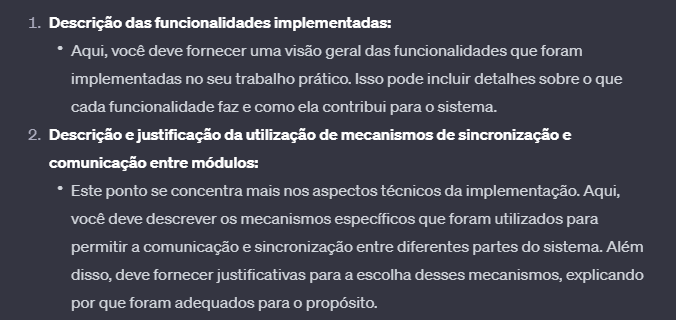
O módulo “Main”, representado pela classe “*kernel*” é o núcleo essencial do sistema, com a finalidade de processar, controlar e validar as etapas fundamentais do sistema operativo de um satélite. Este módulo fica encarregue das verificações e controlo, desempenhando um papel crucial ao tomar as decisões essenciais para garantir o bom funcionamento do sistema.

A função principal do “*kernel*” é iniciar a aplicação e exibir a página de Login. Utilizando a classe *“SwingUtilities”*, garantimos que a criação da interface gráfica ocorre de forma segura, proporcionando assim uma boa experiência ao utilizador.

A função *“handleLogin”* é essencial para autenticar utilizadores. Ao receber informações de nome de utilizador e password, o “*kernel*” lê um ficheiro JSON que contém dados de utilizador. Se o nome de utilizador corresponder a um registo existente e a palavra-passe estiver correta, o utilizador é autenticado com sucesso, resultando na abertura de uma nova página do menu. Caso contrário, é fornecida uma mensagem de erro.

Além disso, são utilizadas funções *“exportMessages”* e *“exportLogs”* que permitem exportam mensagens e logs para arquivos CSV, respetivamente. Este processo é facilitado pelo uso da biblioteca *Swing*.

1. Descrição e justificação da utilização de mecanismos de sincronização e comunicação entre módulos
   1. Módulo Main



1. Funcionalidades adicionais

Além das funcionalidades solicitadas, foram adicionadas funcionalidades extra à aplicação. Este capítulo destaca essas adições, contribuindo para uma experiência mais completa.