CENTRO UNIVERSITÁRIO LUTERANO DE PALMAS
CURSO DE CIÊNCIA DA COMPUTAÇÃO

RODRIGO ALVAREZ CUNHA DONATO

ATUALIZAÇÃO E APRIMORAMENTO DO MÓDULO DE SIMULAÇÃO DOS ALGORITMOS DE ESCALONAMENTO DE PROCESSOS DA PLATAFORMA WEB OSLIVE.

PALMAS – TO
2023

Rodrigo Alvarez Cunha Donato

ATUALIZAÇÃO E APRIMORAMENTO DO MÓDULO DE SIMULAÇÃO DOS ALGORITMOS DE ESCALONAMENTO DA PLATAFORMA WEB OSLIVE.

Projeto Tecnológico II elaborado e apresentado como requisito parcial para obtenção do título de bacharel em ciência da computação pelo Centro Universitário Luterano de Palmas (CEULP/ULBRA).

Orientador: Prof. M.e Madianita Bogo Marioti.

Palmas – TO

2023

Rodrigo Alvarez Cunha Donato

ATUALIZAÇÃO E APRIMORAMENTO DO MÓDULO DE SIMULAÇÃO DOS ALGORITMOS DE ESCALONAMENTO DA PLATAFORMA WEB OSLIVE.

Projeto Tecnológico II elaborado e apresentado como requisito parcial para obtenção do título de bacharel em Ciência da Computação pelo Centro Universitário Luterano de Palmas (CEULP/ULBRA).

Orientador: Prof. M.e Madianita Bogo Marioti.

Aprovado em: _____/_____/_____.

BANCA EXAMINADORA

_____

Prof. M.e Madianita Bogo Marioti

Orientador

Centro Universitário Luterano de Palmas – CEULP

_____

Prof. Douglas Aquino Moreno

Centro Universitário Luterano de Palmas – CEULP

_____

Profa. Fernanda Pereira Gomes

Centro Universitário Luterano de Palmas – CEULP

Palmas – TO

2022

RESUMO

DONATO, Rodrigo Alvarez Cunha. **Atualização e aprimoramento do módulo de simulação dos algoritmos de escalonamento da plataforma web OSLive.** 2023. Trabalho de Conclusão de Curso (Graduação) – Curso de Ciência da Computação, Centro Universitário Luterano de Palmas, Palmas/TO, 2010.

Este trabalho de conclusão de curso objetiva aprimorar a compreensão e o aprendizado sobre gerência de processos em Sistemas Operacionais (SOs), por meio da plataforma OSLive utilizada na faculdade CEULP/ULBRA. Para garantir um ensino efetivo, é importante que os estudantes possam explorar e entender como diferentes algoritmos de escalonamento operam e como afetam o desempenho geral do sistema. Entretanto, apesar do seu potencial, o módulo de simulação dos Algoritmos de Escalonamento de Processos da plataforma OSLive pode estar defasado perante as tecnologias recentes, possuindo ainda erros que limitam a experiência do usuário. Isso evidencia a necessidade de um aperfeiçoamento que permita aos estudantes um aprendizado mais eficiente. Com isso em mente, a investigação propõe aprimorar a simulação dos algoritmos de escalonamento, alinhando-os à atual arquitetura da plataforma e implementando melhorias de conteúdo. Isso inclui a inserção de explicações detalhadas para proporcionar um contexto mais profundo ao material estudado. Além disso, o trabalho foca na identificação e correção de erros e necessidades de melhorias, que atualmente limitam a execução do módulo a certos navegadores e podem influenciar negativamente a confiabilidade dos resultados obtidos. Corrigir essas falhas é fundamental para proporcionar uma experiência do usuário mais fluida e aumentar a versatilidade da ferramenta. Desta forma, esta pesquisa contribui para o aprimoramento do ensino de sistemas operacionais, fornecendo uma ferramenta de aprendizado mais atualizada, eficiente e confiável, que pode auxiliar os alunos a entender melhor o gerenciamento de processos e o funcionamento interno dos SOs.

Palavras-chave: Sistemas Operacionais; OSLive; Simulação; Escalonamento de Processo .

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

CEULP/ULBRA	Centro Universitário Luterano de Palmas
CPU	Unidade Central de Processamento
CEULP	Centro Universitário Luterano de Palmas
E/S	Entrada e Saída
FIFO	First In First Out
RR	Round Robin
SJF	Shortest Job First
SO	Sistema Operacional

SUMÁRIO

1 INTRODUÇÃO

Sistemas Operacionais (SOs) desempenham uma função essencial em sistemas computacionais, atuando como uma ponte essencial entre o *hardware* e o *software* aplicativo, e a disciplina que estuda esses sistemas é obrigatória ou sugerida nas matrizes curriculares nacionais dos cursos superiores da área de Computação e Informática no Brasil, conforme o Ministério da Educação - MEC (BRASIL, 2012). Os SOs controlam e gerenciam a maioria das operações de um computador, desde a inicialização até a execução de aplicativos. Uma das funções mais complexas de um SO, conforme Silva de Oliveira et al. (2001), a gerência de processos em sistemas operacionais envolve a administração dos diversos processos executados por um computador, a alocação eficiente de recursos garante que cada processo receba o tempo de CPU necessário.

O gerenciamento de processos inclui o escalonamento de processo, que disciplina o uso da CPU, escolhendo qual processo vai executar a cada momento, dependendo do algoritmo de escalonamento selecionado para o sistema, assim ele cria uma fila de processos aptos a entrar na CPU. Esse gerenciamento é conduzido pelo escalonador, um processo do SO que utiliza critérios diversos, baseados em algoritmos de escalonamento, para decidir a ordem e a prioridade de execução dos processos.

Existem vários algoritmos de escalonamento, sendo os quatro básicos (Oliveira, 2001): *First-In, First-Out* (FIFO); *Shortest-Job-First* (SJF); *Round-Robin* (RR) e Prioridade. O FIFO, é um algoritmo de paginação simples que mantém uma lista de páginas na memória, removendo a mais antiga durante uma falha para adicionar uma nova página ao final. SJF, é um método não preemptivo que prioriza a execução de processos de acordo com seus tempos de execução. RR, esse algoritmo possui os processos que têm um tempo designado, o quantum; se ainda estiverem em execução ao término desse tempo, a CPU é cedida para outro processo. No algoritmo de Prioridade, em cada processo recebe uma prioridade e o processo executável com a prioridade mais alta pode ser executado.

Além disso, existe também o escalonamento de múltiplas filas, que permite que os processos sejam divididos em diferentes categorias ou "filas", nas múltiplas filas usa-se um ou mais algoritmos para criar uma política de escalonamento e que é isso que os SOs fazem na prática (Silberschatz, Galvin e Gagne, 2013).

O entendimento dos algoritmos relacionados ao escalonamento de processos é fundamental para os estudantes da disciplina de Sistemas Operacionais. Compreender esse conceito pode ser desafiador para os alunos, já que vai além de entender apenas a teoria. É

preciso também conseguir abstrair como os processos são organizados e controlados num ambiente dinâmico e que não pode ser visualizado por isso, o projeto OSLive foi desenvolvido pelo Departamento de Computação do CEULP/ULBRA. O OSLive é uma plataforma online concebida pelos estudantes do Centro Universitário Luterano de Palmas. Seu propósito é fornecer suporte ao ensino e aprendizagem dos conteúdos da disciplina de Sistemas Operacionais (SO) nos cursos de computação.

O módulo de simulação dos algoritmos de escalonamento de processos do OSLive oferece uma experiência interativa, possibilitando aos usuários simular um escalonamento de processos. No módulo, podem ser selecionados diferentes modelos de escalonamento, criar processos individuais e, em seguida, observar os resultados gerados em uma tabela, juntamente com uma animação que representa a fila e o processamento.

Contudo, foi identificado que o módulo de escalonamento de processos do OSLive precisa de melhorias. A necessidade de atualização se deve a fatores relevantes: a plataforma precisa se adaptar a nova estrutura do OSLive, que foi criada por Carlos Bruno Freitas Sardinha e é descrita em detalhes no trabalho "Desenvolvimento de um Módulo para a Resolução de Exercícios Sobre Paginação Por Demanda para o Ambiente Oslive" (2022); foram identificados erros de execução, tanto por alunos quanto pela professora durante o uso na disciplina; além disso, a avaliação realizada por Ana Carolina da Costa Marinho, intitulada "OSLIVE: avaliação da qualidade pedagógica da ferramenta com base na taxonomia TARDA" (Marinho, 2022), evidenciou áreas de melhoria na plataforma.

Deste modo, o objetivo central deste projeto foi desenvolver uma nova versão do Módulo de Simulação dos Algoritmos de Escalonamento do OSLive, não apenas para adequar-se à mais recente arquitetura da plataforma, mas também implementar mensagens explicativas para enriquecer o aprendizado, necessidade identificada na avaliação realizada usando a taxonomia TARDA; e identificar e corrigir os problemas existentes, identificados pelo uso na disciplina.

2 REFERENCIAL TEÓRICO

2.1 Sistemas Operacionais

Sistemas operacionais são componentes essenciais dos sistemas computacionais, responsáveis por gerenciar os recursos de *hardware* e *software* e fornecer uma interface para os usuários (Flynn & McHoes, 2017). Este referencial teórico abordará quatro áreas-chave dos sistemas operacionais: gerência de arquivos, gerência de entrada e saída, gerência de memória e principalmente gerência de processos.

A gerência de arquivos é uma função essencial dos sistemas operacionais, responsável por organizar, armazenar e recuperar informações nos dispositivos de armazenamento (Laudon & Laudon, 2016). Ela gerencia a estrutura de diretórios, metadados e permissões de acesso, garantindo a integridade dos dados e facilitando o gerenciamento dos arquivos pelos usuários e aplicativos. Além disso, a gerência de arquivos também lida com a alocação de espaço em disco e a recuperação de espaço livre, otimizando o uso do espaço de armazenamento disponível (Coulouris *et al*., 2011).

Os sistemas operacionais oferecem uma interface de alto nível para a interação do usuário e dos aplicativos com o sistema de arquivos, ocultando a complexidade das operações de baixo nível envolvidas no gerenciamento de arquivos (Stallings, W., 2012). Essa interface permite que os usuários e os aplicativos criem, leiam, modifiquem e excluam arquivos, bem como manipulem diretórios e gerenciem as permissões de acesso aos arquivos (Flynn & McHoes, 2017). A gerência de arquivos também desempenha um papel fundamental na implementação de mecanismos de segurança, como criptografia de arquivos e sistemas de detecção de intrusão (Bishop, 2002).

A gerência de entrada e saída (E/S) é uma parte dos sistemas operacionais que lida com a coordenação e o controle dos dispositivos de *hardware* externos ao processador e à memória principal, como teclados, mouses, monitores, impressoras, discos rígidos e outros periféricos (Patterson & Hennessy, 2013). Essa área é fundamental para o funcionamento eficiente do sistema, visto que permite a interação do usuário com o computador e a troca de dados entre diferentes componentes de *hardware.*

Para gerenciar as operações de E/S de maneira eficiente, os sistemas operacionais utilizam abstrações, como fluxos de E/S e *buffers*, que facilitam a interação entre os aplicativos e os dispositivos de *hardware* (Tanenbaum & Bos, 2014). Os fluxos de E/S permitem que os aplicativos leiam e escrevam dados de maneira consistente e independente do tipo de dispositivo, enquanto os *buffers* ajudam a otimizar o desempenho das operações de

E/S, armazenando temporariamente os dados antes de serem transferidos entre os aplicativos e os dispositivos (Silberschatz *et al*., 2013).

Os sistemas operacionais gerenciam as operações de E/S por meio de *drivers* de dispositivo e rotinas de interrupção (Brookshear & Brylow, 2014). *Drivers* de dispositivo são programas especializados que atuam como intermediários entre o sistema operacional e o *hardware* específico, traduzindo comandos e dados entre os dois e garantindo a compatibilidade entre os dispositivos e o sistema. As rotinas de interrupção, por sua vez, são mecanismos que permitem que o *hardware* sinalize ao processador quando uma operação de E/S está completa ou quando um evento ocorre, como a chegada de dados de um teclado (Patterson & Hennessy, 2013). Isso permite que o sistema operacional gerencie as operações de E/S de forma assíncrona e eficiente, garantindo que os recursos do processador sejam utilizados de forma otimizada e evitando atrasos desnecessários.

A gerência de memória é uma função essencial dos sistemas operacionais que lida com a alocação e desalocação de memória para os processos e programas em execução, garantindo o uso eficiente e seguro dos recursos de memória disponíveis (Smith & Nair, 2005). Uma das principais tarefas da gerência de memória é garantir que cada processo tenha acesso suficiente à memória e que os processos não interfiram uns nos outros, evitando problemas como a corrupção de dados e a violação de segurança (Comer, 2008).

Existem várias técnicas utilizadas pelos sistemas operacionais para gerenciar a memória, incluindo particionamento, paginação e segmentação (Deitel & Deitel, 2003). O particionamento divide a memória em blocos de tamanho fixo ou variável, que são alocados aos processos conforme necessário. A paginação, por outro lado, divide a memória em blocos de tamanho fixo chamados páginas e permite que os processos sejam divididos em páginas menores que podem ser alocadas em áreas não contíguas da memória, facilitando a alocação eficiente e o gerenciamento de memória virtual (Engineering Libretexts, s/d). A segmentação divide os processos em segmentos de tamanho variável com base em unidades lógicas, como funções ou estruturas de dados, proporcionando uma abordagem mais granular e flexível para o gerenciamento de memória (Laudon & Laudon, 2016).

Além dessas técnicas, os sistemas operacionais também implementam políticas de substituição de páginas e algoritmos de alocação de memória, como o algoritmo de substituição do mínimo tempo de residência (LRU) e o algoritmo de alocação de memória de primeiro ajuste (FFA), para otimizar o uso da memória e garantir o desempenho eficiente dos processos em execução (Flynn & McHoes, 2017). A gerência de memória também é essencial para a implementação de mecanismos de proteção e isolamento, garantindo que os processos

não acessem indevidamente a memória de outros processos ou do próprio sistema operacional (Wai Chan *et al*., 2005).

2.1.1 Gerência de Processos

A gerência de processos é um aspecto central dos sistemas operacionais, responsável por coordenar e controlar a execução de programas e processos em um sistema computacional. A gerência de processos engloba a criação, o gerenciamento, a sincronização e a terminação de processos, garantindo que os recursos do sistema sejam compartilhados de maneira justa e eficiente entre os diversos processos em execução (Comer, 2008).

Um processo é uma instância de um programa em execução, que inclui o código executável, os dados e a pilha de execução, além de informações de estado, como registradores e ponteiros de instrução (Smith & Nair, 2005). Cada processo possui um espaço de endereçamento isolado e recursos alocados, como memória, arquivos e conexões de rede. Os sistemas operacionais gerenciam processos por meio de uma estrutura chamada *Process Control Block* (PCB), que armazena informações importantes sobre cada processo, como o estado atual, o espaço de endereçamento, o contador de programa, os registradores e as prioridades de execução (Deitel & Deitel, 2003).

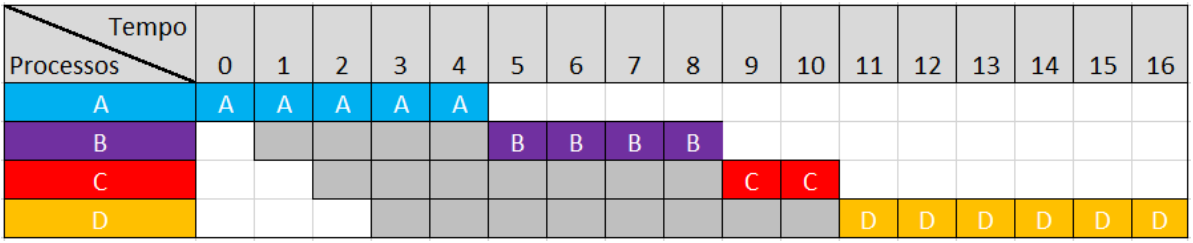
Para garantir o compartilhamento eficiente dos recursos de CPU entre os processos, os sistemas operacionais implementam algoritmos de escalonamento, que determinam a ordem e a duração das execuções dos processos na CPU (Flynn & McHoes, 2017). Existem vários algoritmos de escalonamento, como *First-In, First-Out* (FIFO), *Shortest Job First* (SJF), *Round Robin* (RR) e Prioridade, cada um com suas próprias características e *trade-offs* em termos de desempenho, eficiência e justiça (Laudon & Laudon, 2016). Para cada um deles será apresentado um exemplo, de forma que todos os exemplos utilizam os processos informados na Figura 1.

Figura 1: Tabela de Processos

Processos	T. Chegada	T. Execução	Prioridade
A	0	5	1
B	1	4	2
C	2	2	3
D	3	6	0

First-in, First-out (FIFO): Neste algoritmo, os processos são executados na ordem em que chegam à fila de prontos. O processo que chega primeiro é executado primeiro e assim por diante. É um algoritmo simples, mas pode levar a longos tempos de espera se um processo longo estiver no início da fila. A Figura 2 apresenta um exemplo da utilização do algoritmo FIFO, para os processos indicados na Figura 1.

Figura 2: Escalonamento com Algoritmo FIFO



Como demonstrado na Figura 2, o processo 'A' foi iniciado no instante 0, sendo o primeiro a ser executado, com um tempo de execução de 5 unidades. Enquanto 'A' estava em execução, os processos 'B', 'C' e 'D' chegaram na fila de prontos nos tempos 1, 2 e 3, respectivamente, cada um com seus respectivos tempos de execução - 'B' com 4 unidades, 'C' com 2 unidades e 'D' com 6 unidades. Ao final da execução do processo 'A', o processo 'B' foi o próximo a ser atendido pela CPU. Posteriormente, após a finalização do processo 'B', o processo 'C' foi executado. Finalmente, o processo 'D' foi processado apenas depois que o processo 'C' terminou sua execução e liberou a CPU.

Portanto, a ordem de execução dos processos, seguindo a política FIFO, foi 'A', 'B', 'C' e 'D', respeitando a ordem de chegada e o tempo de execução de cada um. Isso é um exemplo prático de como funciona o escalonamento FIFO em sistemas operacionais.

Shortest Job First (SJF): O processo com o menor tempo de execução estimado é escolhido para ser executado a seguir. Este algoritmo minimiza o tempo médio de espera, mas é difícil de implementar na prática, pois requer conhecimento prévio do tempo de execução de cada processo. A Figura 3 apresenta um exemplo da utilização do algoritmo SJF, para os processos indicados na Figura 1.

Figura 3: Escalonamento com Algoritmo SJF

Tempo / Processos	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
A	A	A	A	A	A												
B								B	B	B	B						
C						C	C										
D												D	D	D	D	D	D

Na Figura 3, é observada a aplicação da política de escalonamento SJF (*Shortest Job First*), também conhecida como Menor Trabalho Seguinte, em uma versão não preemptiva.

Inicialmente, o processo 'A' chega no tempo 0 e imediatamente inicia sua execução. Por ser um algoritmo não preemptivo, a execução de 'A' continua até ser concluída, mesmo que outros processos cheguem durante seu tempo de execução. Deste modo, 'A' completa sua execução no tempo 4. Ao término da execução de 'A', os processos 'B', 'C' e 'D' já estão presentes na fila de prontos. Seguindo a política de escalonamento SJF, a CPU escolherá o processo com o menor tempo de execução restante. Assim, o processo 'C', que tem um tempo de execução de 2 unidades, será o próximo a ser executado. Depois de 'C' concluir sua execução no tempo 6, restam os processos 'B' e 'D' na fila. Deles, 'B', que demanda 4 unidades de tempo de execução, é escolhido para ser executado a seguir, pois requer menos tempo de CPU. Finalmente, após 'B' terminar sua execução no tempo 10, 'D' é o único processo que ainda não foi executado. Portanto, ele é executado em seguida, terminando no tempo 16.

Em resumo, seguindo a política de escalonamento SJF não preemptiva, a sequência de execução dos processos será 'A', 'C', 'B' e 'D'. Esse exemplo ilustra como funciona o escalonamento SJF não preemptivo em sistemas operacionais, priorizando sempre os processos com menor tempo de execução e não permitindo interrupções na execução de um processo até que este seja finalizado.

Round Robin (RR): Cada processo é atribuído a um intervalo de tempo fixo, chamado quantum, para executar na CPU. Quando o quantum de um processo expira, ele é movido para o final da fila e o próximo processo na fila começa a ser executado. Este algoritmo é justo e adequado para sistemas interativos, mas pode ser ineficiente se o quantum for muito grande ou muito pequeno. A Figura 4 apresenta um exemplo da utilização do algoritmo RR, para os processos indicados na Figura 1.

Figura 4: Escalonamento com Algoritmo RR

Tempo / Processos	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
A	A	A	A									A	A				
B				B	B	B								B			
C							C	C									
D									D	D	D				D	D	D

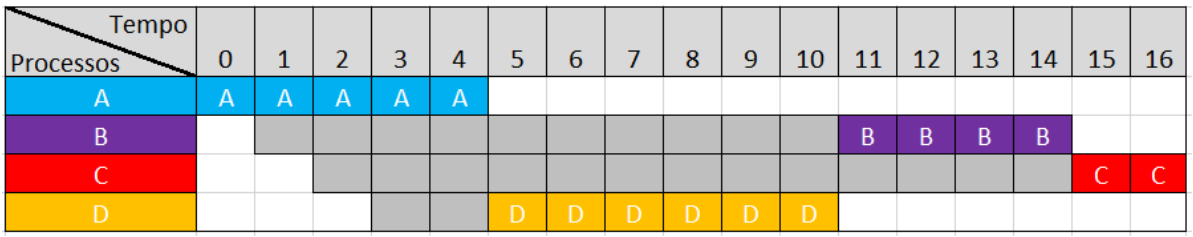
Na Figura 4, a política de escalonamento adotada é a RR (*Round Robin*), com um *quantum* de tempo definido como 3 unidades. No escalonamento RR, cada processo na fila de prontos é alocado para executar por um período fixo de tempo, conhecido como *quantum*. Se um processo não concluir sua execução dentro desse quantum, ele é interrompido e enviado ao final da fila de prontos, onde aguarda sua próxima vez de executar. Inicialmente, 'A' é o primeiro processo a chegar e, portanto, o primeiro a ser executado. Como o quantum é 3, 'A' é executado por 3 unidades de tempo, sendo então interrompido e enviado para o final da fila. Nesse momento, 'A' ainda tem 2 unidades de tempo restantes para executar. Em seguida, a CPU se volta para o próximo processo na fila, que é 'B'. 'B' tem um tempo de execução total de 4 unidades, porém, será interrompido após as 3 unidades de tempo devido ao quantum. 'B', então, ainda tem 1 unidade de tempo restante para executar. O processo seguinte na fila é 'C', que tem um tempo de execução total de 2 unidades. Como 2 é menor que o quantum de 3, 'C' é capaz de concluir sua execução na primeira vez que é alocado para executar, liberando espaço na fila de prontos. Em seguida, 'D' é alocado para executar. Com um tempo de execução total de 6 unidades, 'D' será interrompido após o quantum de 3 unidades, restando ainda 3 unidades de tempo para completar sua execução. Nesse ponto, todos os processos já tiveram ao menos uma chance de executar e a CPU retorna ao início da fila, onde 'A' aguarda para continuar sua execução. 'A' executa as 2 unidades de tempo restantes e conclui sua execução. Em seguida, 'B' e 'D' são alocados para executar o tempo restante, um após o outro, completando o ciclo de execução.

Assim, a sequência de execução, sob a política de escalonamento RR com um quantum de 3, será 'A', 'B', 'C', 'D', 'A', 'B', 'D'. Isso ilustra como o escalonamento RR permite uma partilha de tempo mais justa entre os processos, garantindo que todos tenham uma oportunidade de executar em um período de tempo razoável.

Prioridade: Os processos são atribuídos com prioridades e o escalonador seleciona o processo com a maior prioridade para executar a seguir. As prioridades podem ser estáticas ou dinâmicas, e o algoritmo pode ser preemptivo ou não preemptivo. Este algoritmo é flexível e pode ser adaptado a várias situações, mas pode levar à inanição se os processos de baixa prioridade forem continuamente preteridos por processos de alta prioridade.

As Figura 5 e 6 apresentam exemplos da utilização dos algoritmos de prioridade não preemptivo e preemptivo, respectivamente, considerando os processos da Figura 1.

Figura 5 - Escalonamento com Algoritmo de Prioridade não Preemptivo.



Na Figura 5, é aplicado o algoritmo de escalonamento por prioridade não preemptiva. No escalonamento por prioridade, cada processo recebe um nível de prioridade e a CPU é alocada ao processo com a prioridade mais alta (sendo 0 a prioridade mais alta e números maiores representando prioridades mais baixas). Em uma versão não preemptiva deste algoritmo, uma vez que um processo inicia sua execução, ele não é interrompido até que termine, independentemente das prioridades dos outros processos que possam chegar. 'A' é o primeiro processo a chegar e, portanto, inicia a execução imediatamente. Apesar de 'B' e 'C' chegarem enquanto 'A' ainda está em execução, eles não podem interromper 'A' porque o escalonamento por prioridade não preemptiva está sendo utilizado. 'A' completa sua execução no instante 4. Quando 'A' conclui a execução, 'B', 'C' e 'D' estão na fila de prontos. A CPU é então alocada ao processo com a prioridade mais alta, que neste caso é 'D' com prioridade 0. 'D', apesar de ser o último a chegar, é executado a seguir devido à sua alta prioridade. 'D' inicia sua execução no tempo 4 e conclui no tempo 10. Após a conclusão de 'D', 'B' e 'C' estão esperando na fila. Entre eles, 'B', com uma prioridade de 2, é o próximo na ordem de prioridades e, portanto, é o próximo a ser executado, começando no tempo 10 e terminando no tempo 14. Por último, 'C', que tem a menor prioridade de todos os processos (3), é executado depois de todos os outros terem concluído. 'C' inicia sua execução no tempo 14 e termina no tempo 16.

Assim, a sequência de execução dos processos é 'A', 'D', 'B' e 'C', de acordo com suas prioridades e a aplicação do algoritmo de escalonamento por prioridade não preemptiva. Este exemplo demonstra como o escalonamento por prioridade pode ser útil para garantir que os processos mais críticos (ou seja, aqueles com maior prioridade) sejam executados antes dos processos de menor prioridade.

Figura 6: Escalonamento com Algoritmo de Prioridade Preemptivo

Tempo / Processos	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
A	A	A	A							A	A						
B												B	B	B	B		
C																C	C
D				D	D	D	D	D	D								

Na Figura 6, é utilizado o algoritmo de escalonamento por prioridade preemptiva. 'A' começa a execução imediatamente, pois é o único processo na fila. No entanto, quando 'D' chega no instante 3, interrompe 'A' devido à sua prioridade superior e assume a CPU. 'D' prossegue sem interrupções até a conclusão no instante 9. Com 'D' concluído, 'A' retoma sua execução (já que tem prioridade maior que os processos restantes 'B' e 'C'), terminando no instante 10. Em seguida, 'B' inicia sua execução, tendo uma prioridade mais alta que 'C', e termina no instante 14. Finalmente, 'C', com a menor prioridade, é executado de 14 a 16. Portanto, sob o escalonamento de prioridade preemptiva, a sequência de execução dos processos é 'A' (parcialmente), 'D', 'A' (conclusão), 'B' e 'C'.

A gerência de processos é um aspecto fundamental do sistema operacional que envolve a manutenção e gerenciamento dos processos que estão sendo executados em um sistema (Tanenbaum & Bos, 2015). A gerência de processos não se limita apenas à alocação de CPU aos processos, mas também lida com sincronização, comunicação e compartilhamento de recursos entre os processos. Esses conceitos são essenciais para o funcionamento eficiente e seguro de um sistema operacional e, por consequência, de todo o sistema computacional.

A sincronização de processos dependentes precisam ser coordenados para evitar condições de corrida e garantir a consistência dos dados (Silberschatz, Galvin & Gagne, 2018). Mecanismos de sincronização de processos, como semáforos e monitores, são usados para garantir que os processos acessem recursos compartilhados de forma organizada e sem conflitos.

A comunicação entre processos (*Inter-Process Communication* - IPC) é outro componente importante da gerência de processos (Tanenbaum & Bos, 2015). Os processos frequentemente precisam se comunicar uns com os outros para completar tarefas que exigem colaboração. O IPC pode ocorrer através de vários métodos, incluindo tubos (*pipes*), *sockets*, filas de mensagens e memória compartilhada.

O compartilhamento de recursos, como a CPU e a memória, é fundamental para a eficiência e a performance do sistema (Silberschatz, Galvin & Gagne, 2018). No caso da CPU, os algoritmos de escalonamento são usados para determinar a ordem e a duração da

alocação da CPU aos processos. Os recursos podem ser compartilhados de forma justa (por exemplo, através do escalonamento *round-robin*) ou com base na prioridade (por exemplo, através do escalonamento por prioridades). Os sistemas operacionais também implementam mecanismos para prevenir a condição de impasse (*deadlock*) onde dois ou mais processos estão impossibilitados de prosseguir porque cada um está esperando que o outro libere um recurso.

Para garantir o compartilhamento seguro e eficiente de recursos, o sistema operacional também implementa políticas e mecanismos de proteção (Tanenbaum & Bos, 2015). Isso é especialmente importante em sistemas operacionais multiusuário e multiprogramados, onde o acesso não autorizado ou o uso indevido de recursos pode levar a problemas de segurança ou degradação do desempenho.

2.2 OSLive

O OSLive é uma plataforma online criada pelos estudantes do Centro Universitário Luterano de Palmas (CEULP/ULBRA) com o objetivo de oferecer suporte ao ensino e aprendizado dos conteúdos da disciplina de Sistemas Operacionais (SO) nos cursos de computação. Na página inicial do OSLive, estão disponíveis opções para acessar os diversos módulos, como mostrado na Figura 7 abaixo. Para acessar esses módulos, basta utilizar o link http://ulbra-to.br/oslive/.

Figura 7: Tela Inicial do OSLive

Fonte: http://ulbra-to.br/oslive/.

No módulo dedicado à área de gerenciamento de processos, encontra-se uma série de exercícios e simulações que abordam o funcionamento do escalonamento de processos. Esse recursos oferece uma oportunidade para os alunos explorarem e compreenderem melhor os aspectos fundamentais dos sistemas operacionais. No módulo de gerenciamento de memória, a ferramenta proporciona simulações de algoritmos de paginação, incluindo paginação por demanda, e explora o algoritmo de segmentação. Essas simulações são valiosas para os estudantes, pois permitem uma compreensão prática e visual desses conceitos essenciais em sistemas operacionais.

2.2.1 Simulação De Escalonamento De Processo De Múltiplas Filas

A simulação é uma das ferramentas de análise mais poderosas disponíveis para os responsáveis por projetos e operações de processos ou sistemas complexos. Em um mundo competitivo, a simulação tornou-se uma ferramenta eficaz para o planejamento, projeto e controle de sistemas. Não é mais considerada a abordagem de "último recurso", é hoje vista como uma metodologia de resolução de problemas indispensáveis para engenheiros, designers e gerentes (Shannon, 1992).

De acordo com a definição de De Jong (1991), a simulação computacional é a replicação por meio de um programa de computador de um fenômeno, processo, sistema ou aparato. Nesse processo de replicação, a saída do programa é derivada dos dados de entrada fornecidos pelo usuário.

A aplicação desse recurso se estende por várias esferas do conhecimento, possibilitando a realização de testes em ambientes simulados para validar resultados antes da implementação no mundo real. Essa abordagem não apenas evita desperdício de recursos ao antecipar os desfechos, mas também oferece a oportunidade de visualizar e compreender o funcionamento de sistemas que, de outra forma, permaneceram invisíveis. Essa compreensão aprofundada pode ser explorada em contextos didáticos, científicos ou profissionais, enriquecendo a compreensão dos fenômenos em questão.

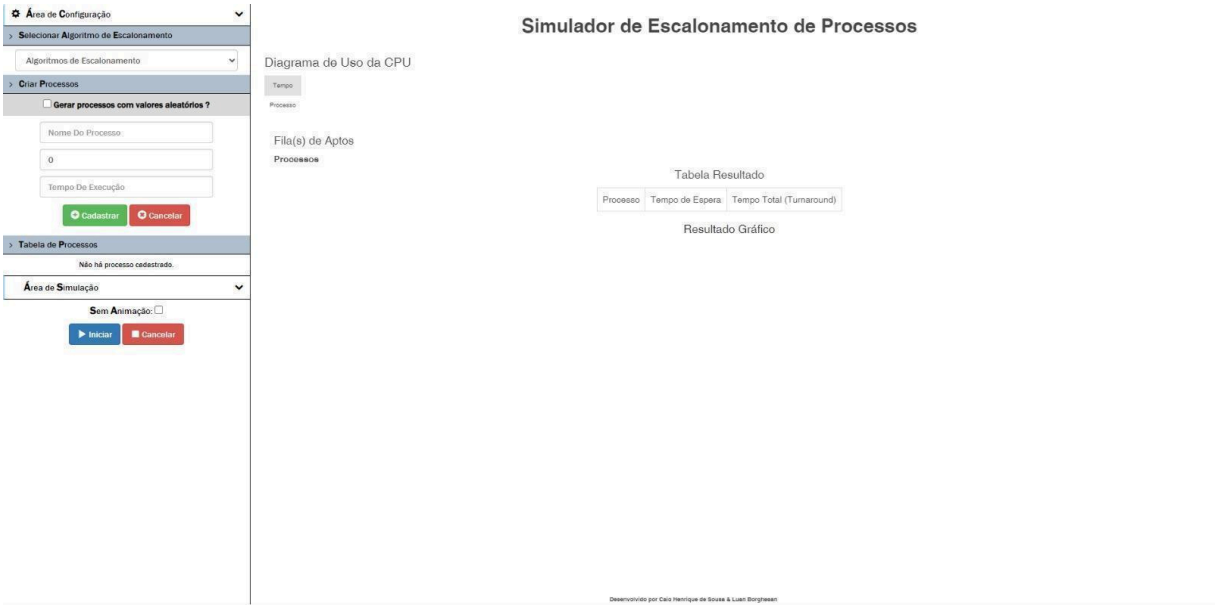
Dada a importância sobre os recursos de aplicação que se estende em ambientes de gerenciamento e conhecimento, o OSLive é composto por vários módulos, um dos módulos implementado é a Simulação de escalonamento de processos, que oferece a simulação dos quatro algoritmos básicos de escalonamento. Com isso, os sistemas operacionais aplicam suas políticas de escalonamento ao utilizar uma combinação de algoritmos de escalonamento,

formando o que é conceitualmente conhecido como algoritmo de múltiplas filas. Contudo, analisar individualmente a simulação dos quatros algoritmos fundamentais não proporciona uma compreensão completa das políticas de escalonamento dos sistemas operacionais, as quais se fundamentam nos algoritmos de múltiplas filas.

Essa estrutura combinada emprega os algoritmos básicos de acordo com a demanda específica do sistema. Portanto, torna-se evidente a relevância da abordagem simplificada que oferece uma visão de como as políticas de escalonamento dos sistemas operacionais empregam os algoritmos em conjunto, permitindo os alunos compreenderem como os algoritmos interagem, e como os sistemas operacionais realizam o escalonamento de processos.

De acordo com Silva (2010), no módulo de simulação de escalonamento os alunos têm a oportunidade de compreender a dinâmica dos algoritmos de escalonamento, os quais desempenham o papel de atribuir o processador a um determinado processo entre diversos existentes no sistema. Além disso, podem experienciar de forma prática conceitos, como preempção, prioridade, fila de processos aptos, tempo de espera, tempo total e tempo de execução. Isso permite não apenas a compreensão teórica, mas também a vivência concreta desses elementos-chave na gestão e na execução dos processos em um ambiente computacional.

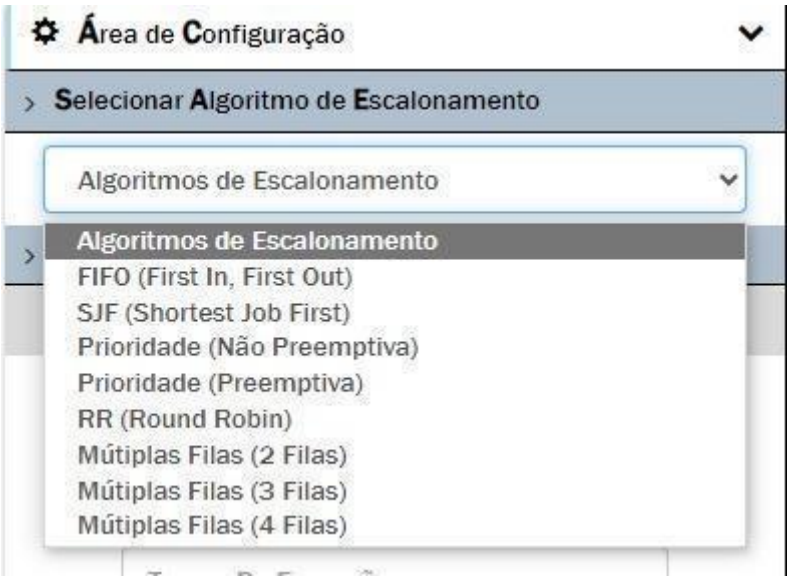
Figura 8: Tela Inicial do Simulador de Escalonamento



Fonte: : https://ulbra-to.br/oslive/simulacao/

Quando há um ou mais processos prontos para a execução, o sistema operacional enfrenta a decisão de determinar a ordem de execução, utilizando um dos algoritmos de escalonamento disponíveis. A simulação desse processo se inicia com a seleção do algoritmo a ser utilizado na Área de Configuração. Nessa etapa, estão listados os algoritmos disponíveis, conforme apresentado na Figura 9. Essa escolha define o comportamento e a lógica específica adotada para a execução dos processos em questão.

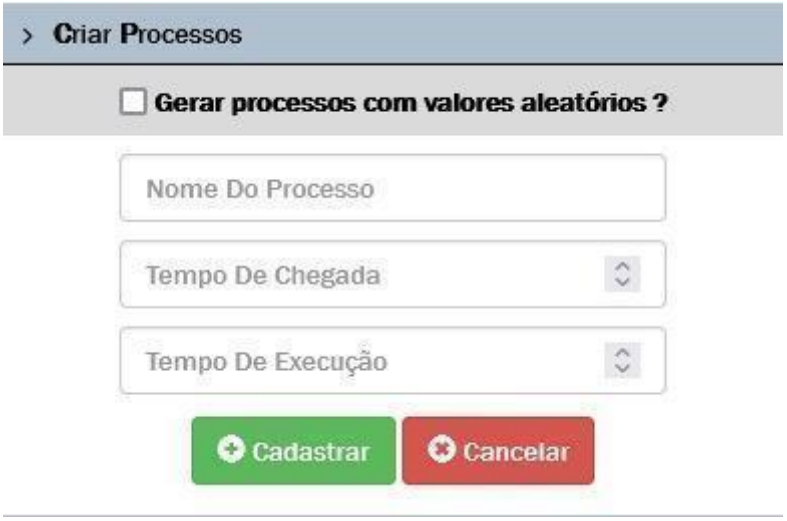
Figura 9: Área de Configuração: Seleção do Algoritmo de Escalonamento



Fonte: https://ulbra-to.br/oslive/simulacao/

Uma vez selecionado o algoritmo de interesse, o próximo passo é gerar os processos para que o sistema operacional possa gerenciar sua execução, aplicando o algoritmo escolhido. Esse processo de geração pode ser conduzido de duas maneiras: gerando processos com valores aleatórios ou inserindo-os manualmente. É importante observar que ao inserir manualmente, a ferramenta destaca a necessidade de atribuir o tempo de chegada do primeiro processo como sendo igual a 0, como evidenciado na Figura 10. Essa consideração inicial é necessária para a correta execução e simulação do funcionamento do algoritmo selecionado.

Figura 10: Área de Configuração: Criação dos processos manualmente



Fonte: : https://ulbra-to.br/oslive/simulacao/

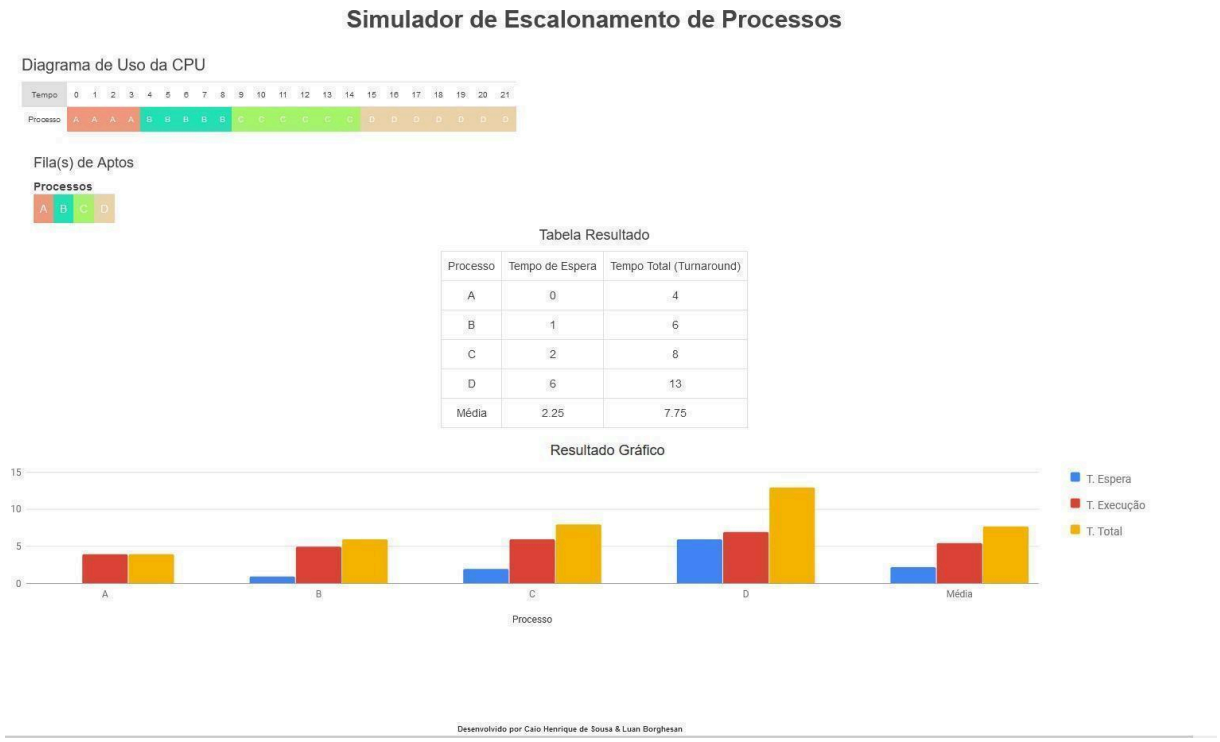
Os processos foram gerados com valores aleatórios. Com isso feito, é viável dar início à simulação do funcionamento do algoritmo de escalonamento selecionado, neste caso, o *First in, First out* - FIFO. Você pode optar por visualizar o resultado com ou sem animação ao marcar a caixa de seleção.

Figura 11: Área de Configuração: Criação dos Processos com Valores Aleatórios

Fonte: https://ulbra-to.br/oslive/simulacao/

Após iniciar a simulação, a área de visualização do escalonamento de processos exibe o funcionamento do algoritmo selecionado, conforme mostrado na Figura 11. No diagrama de utilização da CPU, é possível observar os processos em execução, seus tempos de chegada à CPU e a duração de suas execuções até a conclusão. A fila de processos aptos mostra aqueles que aguardam para serem executados, enquanto a tabela de resultados apresenta informações sobre os tempos de espera (o período que cada processo aguardou na fila de aptos para execução) e o tempo total (o tempo de espera somado ao tempo de execução). Adicionalmente, há um gráfico que ilustra visualmente os tempos destacados na tela de resultados, incluindo o tempo médio de execução dos processos conforme o algoritmo de escalonamento escolhido.

Figura 12: Área de Simulação: Resultado do Escalonamento



Fonte: https://ulbra-to.br/oslive/simulacao/

Concluído esse processo, há a finalização da criação do exercício. Na ferramenta OSLive, disponibiliza-se o campo da tabela de resultados onde os valores gerados podem ser visualizados e manipulados para análise adicional ou para a resolução de problemas específicos dentro do contexto do escalonamento de processos.

3 METODOLOGIA

Neste segmento, serão detalhados os recursos, instrumentos e técnicas adotadas para a realização deste estudo, cujo objetivo foi desenvolver uma nova versão do Módulo de Simulação dos Algoritmos de Escalonamento de Processos, que será integrado ao ambiente OSLive.

3.1 Materiais

Esta seção apresenta as ferramentas Angular, *Bootstrap* e *HTML5* associando sua utilidade no trabalho e a arquitetura do *software*, visando esclarecer o *design* do projeto.

3.1.1 Ferramentas Utilizadas

A camada de *interface* com o usuário, conhecida como *Front-end*, é a porção visível e interativa do desenvolvimento de *software*. Ela se torna a ponte entre o usuário e o sistema, oferecendo as informações e as ferramentas que facilitam o engajamento do usuário com a plataforma.

No desenvolvimento do *Front-end* deste projeto, foi utilizado o *Angular*, uma plataforma de desenvolvimento robusta. Escrito em *TypeScript*, o *Angular* é um *framework* de desenvolvimento voltado para a construção de aplicações *web* dinâmicas e responsivas (Angular, 2022).

Complementando o conjunto de ferramentas, o *Bootstrap*, um *framework* de código aberto idealizado em 2010 pela equipe da rede social *Twitter*. Ele foi projetado para ser uma referência em estilização de interfaces *web* (Bootstrap, 2021), facilitando a criação de um *design* consistente e adaptável.

O *CSS*, que manipula a apresentação visual dos elementos da *web*, foi empregado para refinar a estética do site. Quando combinado com o *Bootstrap*, o *CSS* permite criar uma experiência de usuário amigável, proporcionando maior fluidez e navegabilidade. Além disso, o *HTML5*, uma linguagem de marcação de textos essencial para a construção de páginas *web*, foi usado para estabelecer a estrutura básica das páginas da plataforma. A escolha do HTML5 se deu por sua versatilidade, acessibilidade e amplo suporte nos navegadores modernos.

3.2 Desenho de Estudo

Os processos utilizados para o desenvolvimento desse projeto foram realizados seguindo etapas. Na Figura 13, a seguir é possível ver a sequência das etapas.

Figura 13: Sequência do trabalho

Inicialmente, foi realizado um encontro com a especialista da área, onde foram discutidas as necessidades do módulo de Simulação de Escalonamento de Processo e as fontes para a identificação de necessidades de melhoria e de erros do módulo de simulação. A etapa seguinte consistiu em uma investigação dos conceitos de Sistema Operacional (SO) relacionados ao tópico de escalonamento de processos, bem como estudar as tecnologias que seriam utilizadas no *Front-End*, que são definidas na descrição da arquitetura descrita no trabalho do Sardinha (2022).

Na fase subsequente, como o objetivo do trabalho foi desenvolver uma nova versão adaptada à nova arquitetura do projeto, foi realizado a fase de identificação de erros e necessidades de melhorias, essa identificação ocorreu da seguinte maneira:

- Primeiramente, as reuniões, com a orientadora e coordenadora do projeto do OSLive, Madianita Bogo Marioti, que também é professora da disciplina Sistemas Operacionais, foram fundamentais para prover *feedbacks* e sugestões que foram documentados através de discussões aprofundadas sobre o progresso e as áreas necessitadas de refinamento;
- Em seguida, a identificação de erros e necessidades de melhorias:
    - a partir do *feedback* da avaliação da ferramenta na taxonomia TARDA (Marinho, 2022), foi possível identificar erros e necessidades de melhorias na

plataforma OSLive como um todo, e assim definir o que foi implementado no módulo de simulação de escalonamento de processos;

- a observação do uso da simulação no ambiente de sala de aula resultou no levantamento uma relação de necessidades de melhoria nesta etapa, houve interação direta com alunos e a professora da disciplina, que os orientou a informarem problemas e sugestões de melhorias do módulo de simulação de escalonamento de processos da plataforma OSLive. As melhorias e correções identificadas foram: verificação de cadastro dos processos, necessidade de responsividade, ordenar a tabela de processo, execução em navegadores diferentes.

A quarta fase, a implementação do módulo, incluiu o desenvolvimento, os testes e a verificação das soluções apresentadas na terceira fase, conforme orientações da orientadora e especialista da área. Essa fase seguiu da seguinte forma:

1) No decorrer do desenvolvimento, um conjunto de melhorias foram previamente selecionadas a serem integradas, resultando em uma nova versão da simulação de escalonamento de processos da plataforma OSLive, destinada às fases de teste pelo desenvolvedor. O objetivo era apresentar essa versão à especialista para validação e, se necessário, indicação de ajustes;

2) As fases de testes envolveram a verificação de entradas de dados, mensagens explicativas, cores para representar componentes, fluxo de dados, estrutura do projeto e mensagens de erro. A especialista também testou a funcionalidade do aplicativo, verificou as informações de entrada e saída, analisou a representação precisa do funcionamento dos algoritmos e a precisão dos resultados;

3) Na fase de validação, foram realizadas reuniões alternadas apresentando o trabalho realizado e a especialista testando o sistema de forma independente e avaliando correções e ajustes necessários. Se uma funcionalidade fosse

aprovada, outra seria escolhida para desenvolvimento até que todos os requisitos fossem atendidos.

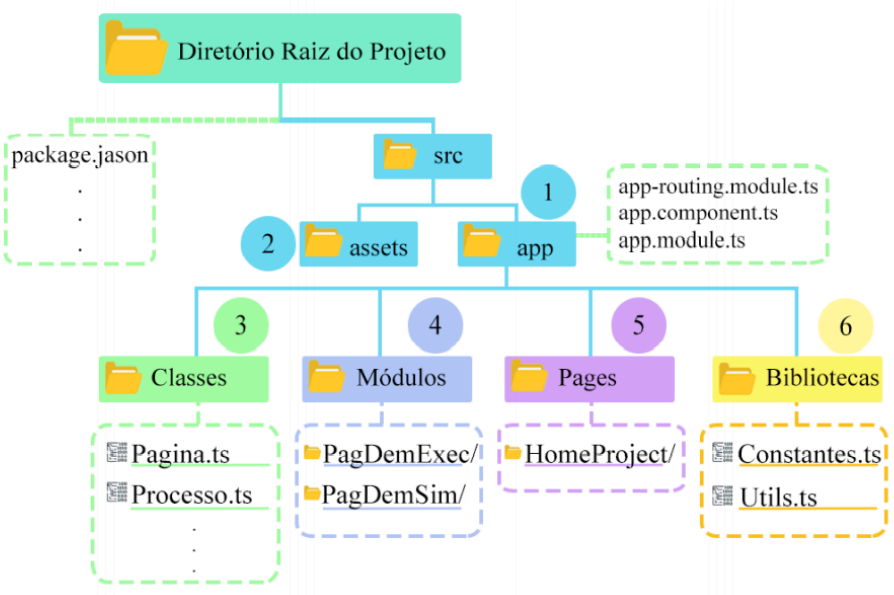
Finalmente, na última etapa, foi feita a entrega da nova versão, com a apresentação das melhorias. Nesse encontro, foi dado destaque às modificações implementadas no módulo de simulação de escalonamento de processo. Foram exploradas áreas de aprimoramento, mas também possíveis erros presentes no sistema, cada falha compreendida foi corrigida. A correção de erros visou resolver não somente as questões imediatas, mas também garantir que o módulo de simulação estivesse otimizado para aprimorar a experiência do usuário. Sendo assim, consolidou as melhorias na nova versão do módulo de simulação de escalonamento da plataforma OSLive.

3.2.1 Estrutura Atual da Plataforma OSLive

A estrutura do OSLive recebeu uma atualização significativa, originada do trabalho de Sardinha (2022). A reformulação estratégica da arquitetura foi analisada e, também, proposta por Carlos Bruno Freitas Sardinha em seu trabalho. Sardinha propôs uma nova estrutura para o OSLive. Esta proposta enfatiza a integração dos módulos e a padronização dos processos, estabelecendo um ambiente em que os recursos podem ser reutilizados e as funcionalidades podem ser compartilhadas entre diferentes projetos. O objetivo é claro: melhorar a eficiência operacional eliminando redundâncias, promover a consistência no desenvolvimento e uso das aplicações.

Esta iniciativa, como proposta por Sardinha (2022), não é apenas uma melhoria na gestão atual do OSLive, mas uma medida preventiva, antecipando as necessidades de escalabilidade e aprimoramento do sistema no futuro. Ele reconhece que para o OSLive continuar a evoluir e acomodar novas inovações, uma estrutura padronizada, integrada e eficiente não é apenas benéfica, mas essencial, destacando a importância da reutilização de código e da uniformidade na estruturação dos projetos para o sucesso contínuo e sustentável do OSLive. A Figura 14 detalha a nova macroestrutura.

Figura 14: Macroestrutura do OSLive



Fonte: Sardinha (2022)

A Figura 14 apresenta uma estrutura interna coesa e unificada para o OSLive. Esta configuração foi projetada para promover a integração e a eficiência, simplificando a reutilização de recursos dentro do ambiente. Os componentes da figura são explicados seguindo uma ordem lógica que reflete sua funcionalidade e interdependência:

1. Pasta "Módulos" (4): representa o epicentro funcional do sistema, servindo como depósito para todos os módulos individuais. Cada módulo, criado como uma entidade operacional autônoma, é mantido aqui, pronto para ser integrado no ecossistema mais amplo.

2. Pasta "app" (1): age como o alicerce estrutural do projeto, hospedando o arquivo *app-routing.module.ts*. Este arquivo é imperativo, pois orienta a navegação global, estipulando a página inicial e direcionando para os módulos localizados na pasta "módulo".

3. Pasta *"assets"* (2): opera como um armazém central, essencial para a administração eficaz dos ativos estáticos utilizados pelos módulos e outras facetas do sistema. Recursos como imagens e outros documentos são armazenados aqui para um acesso e reutilização facilitados.

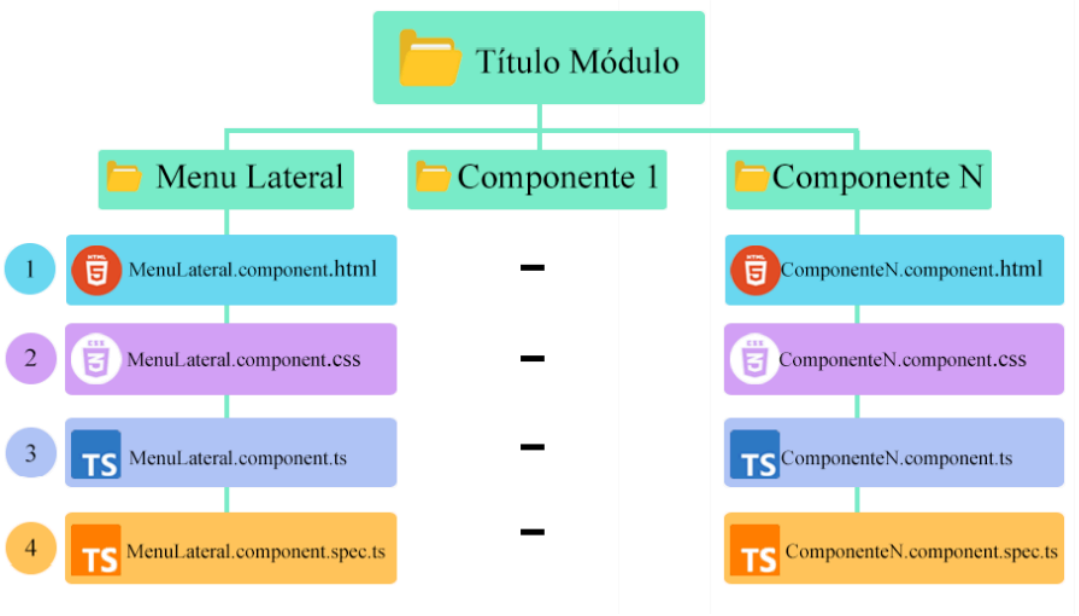
4. Pasta "Pages" (5): dedicada a conteúdos que forneçam informações adicionais ou contextuais, esta área é reservada para hospedar páginas como a homepage, detalhes sobre a equipe do projeto e informações sobre colaboradores, distinta dos módulos operacionais.

5. Pastas "Classes e Funções" (3) & “Bibliotecas“ (6): constituem os pilares reutilizáveis que suportam a operacionalidade dos módulos. As classes (3) especificam

estruturas e métodos comuns, enquanto as funções (6) são utilizadas para tarefas específicas. Adicionalmente, constantes parametrizáveis (também no ponto 6) oferecem uma configuração versátil e uniforme através dos vários módulos.

A estrutura proposta por Sardinha (2022), visa o aumento da eficácia no desenvolvimento, a simplificação da manutenção e a uniformidade na experiência do usuário, promovendo a reutilização de código e a padronização. Esta abordagem potencializa sua robustez, flexibilidade e escalabilidade. Na Figura 15 é apresentado como foi estruturado o módulo de escalonamento de processos.

Figura 15: Estrutura dos Módulos do OSLive



Fonte: Sardinha (2022)

Seguindo o modelo criado por Sardinha (2022), os módulos do OSLive seguiram o modelo proposto, com o menu lateral e uma pasta para cada componente, deixando assim de fácil compreensão e modificação.

4 RESULTADOS

O propósito deste trabalho foi criar uma nova versão do Módulo de Simulação dos Algoritmos de Escalonamento na plataforma *web* OSLive. O escopo abrangeu desde a identificação de erros encontrados na plataforma durante sua utilização por alunos e pela professora como apoio na disciplina de Sistemas Operacionais nos cursos da áreas da tecnologia da informações do CEULP/ULBRA até a necessidade de reconstruir o código de acordo com a atual estrutura do projeto OSLive, com as necessidades de melhorias e correções.

A implementação seguiu conforme a atual arquitetura do projeto OSLive desenvolvida pelo Sardinha (2022), que foi disponibilizada na plataforma de repositórios GitLab. Onde foi criado um novo módulo, para suportar a simulação de escalonamento, que por sua vez seguiu o modelo do outro modo já existente no repositório, com um componente para o menu lateral e outro para área de simulação. Também foi adicionado no repositório novas classes necessárias para a simulação de processos ocorrer corretamente.

4.1 Identificação de Necessidades de Melhorias do Módulo de Escalonamento de Processos da Plataforma OSLive

OSLive foi lançado em 2020, ao longo do tempo foram identificados alguns erros em sua estrutura. Esta descoberta aponta para a necessidade de revisão e aprimoramento em alguns módulos do sistema. Neste trabalho, foi realizado um levantamento de necessidades de melhorias no Módulo de Simulação de Escalonamento de Processos, que foram identificadas durante a utilização em sala de aula nos cursos de Sistemas Operacionais. Além disso, foram coletados *feedbacks* dos alunos, conforme apresentado no projeto tecnológico: OSLIVE: avaliação da qualidade pedagógica da ferramenta com base na taxonomia TARDA (Marinho, 2022). Por meio desses *feedbacks,* foram levantadas as necessidades apresentadas na Tabela 1.

Tabela 1: Lista de Necessidade de Melhorias no Módulo de Simulação Escalonamento de Processos

	Melhorias no Módulo
1	Verificação de Cadastro dos Processos
2	Falta de Responsividade
3	Ordenar a Tabela de Processo

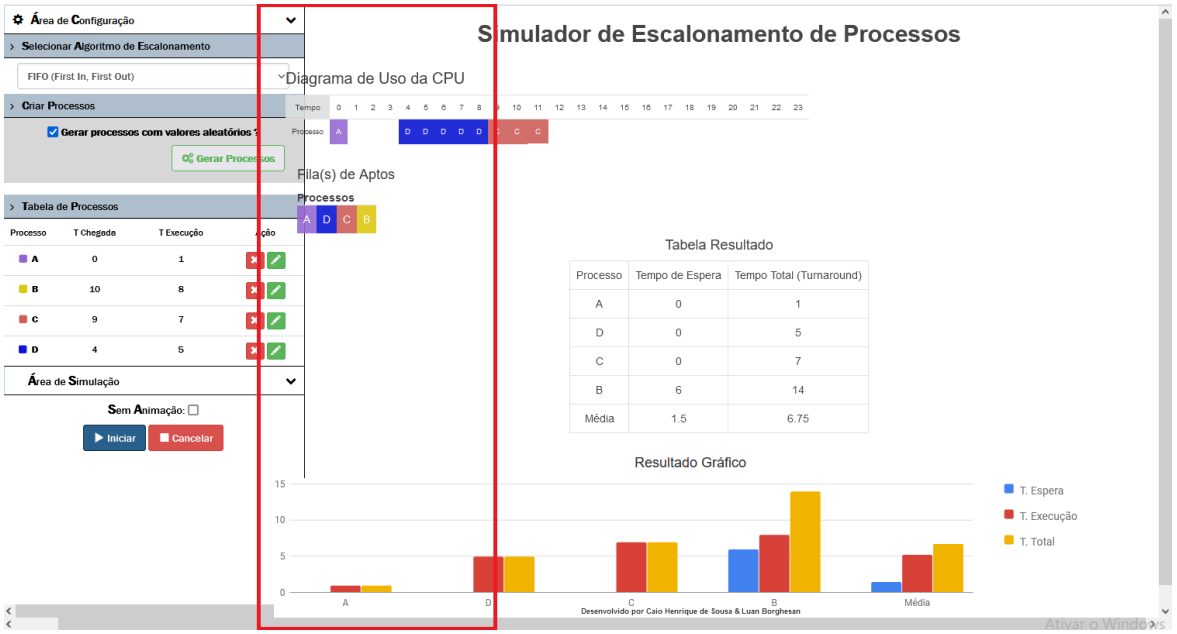
4	Execução em Navegadores Diferentes
5	Ícones de Informações

A funcionalidade de verificação no cadastro dos processos não permitirá a adição de dois processos com mesmo tempo de chegada como mostra na Figura 16.

Figura 16: Processos com o Mesmo Tempo de Chegada

Quanto a estrutura, foi sugerida a correção das representações gráficas da plataforma, que não possuía responsividade, como demonstrado na Figura 17.

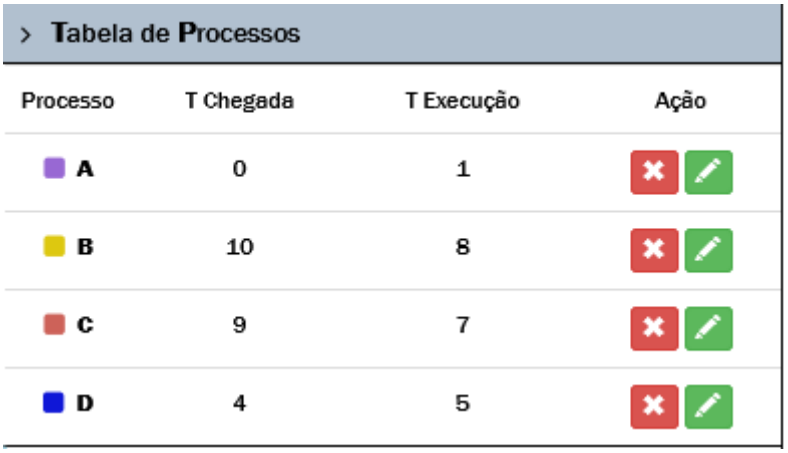
Figura 17: Erro de Responsividade no Módulo de Escalonamento de Processos



A responsividade das representações gráficas é um aspecto fundamental para garantir uma experiência de usuário consistente e eficiente. Foi identificado um problema específico relacionado à responsividade, ao arrastar para o lado na barra inferior para direita, observou-se a sobreposição inadequada de blocos, o que prejudica a visualização e compreensão das informações apresentadas. Esta falha impacta a usabilidade da ferramenta, tornando essencial a correção desse erro para garantir uma representação gráfica precisa e livre de conflitos, permitindo aos usuários uma fluidez compreensível com a plataforma.

A próxima necessidade de melhoria identificada no módulo está relacionada à ordenação dos processos na tabela, visando organizá-los em ordem crescente de acordo com o tempo de chegada, que é o momento de transição do processo para a fila de aptos, como costuma ser apresentado na representação dos livros didáticos. Como é possível ver na Figura 18, que não está ordenado por tempo de chegada.

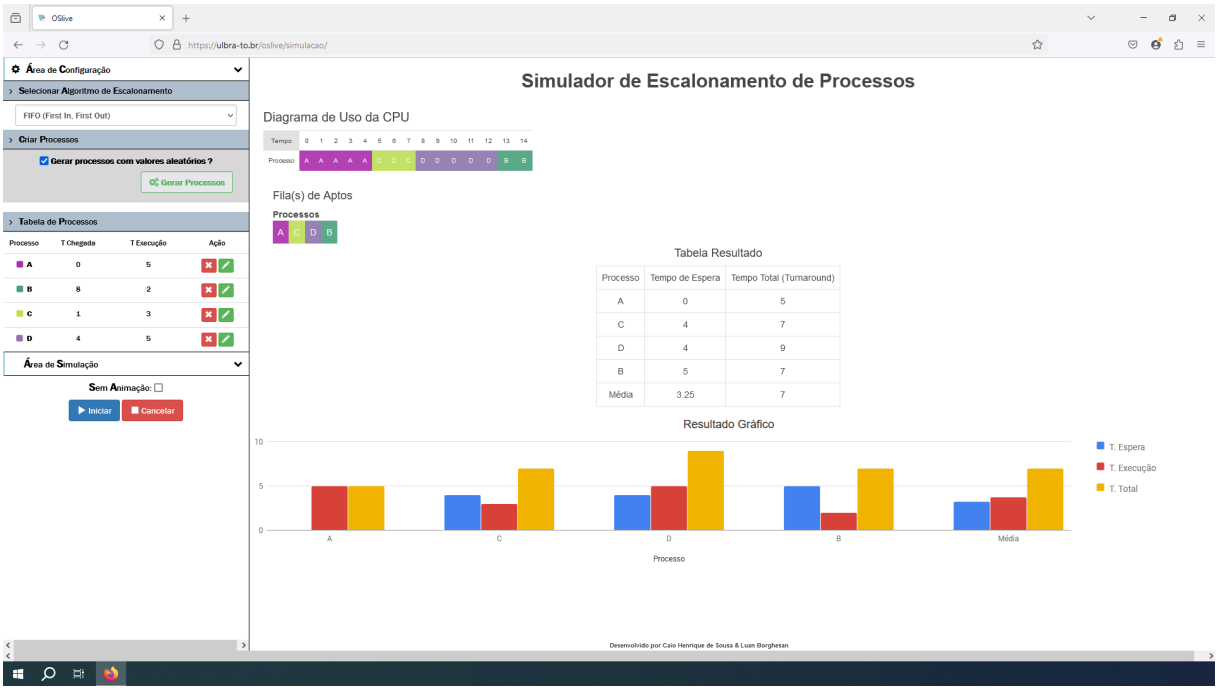
Figura 18: Ordenar a Tabela de Processos



Ao ordenar os processos de forma crescente em relação ao tempo de chegada, os usuários terão uma visualização mais clara e lógica da sequência temporal, facilitando a compreensão e evitando que o usuário se confunda no momento da análise dos processos por não perceber que a ordem de chegada está trocada.

Outra melhoria identificada foi a necessidade de tornar o módulo de escalonamento de processos compatível com uma variedade de navegadores e essa limitação identificada tem impacto significativo na acessibilidade e na utilidade para usuários que preferem ou dependem de outros navegadores para suas atividades cotidianas. Na Figura 18 é possível observar que a simulação de escalonamento de processos ocorrendo de forma adequada, foi apresentada corretamente no navegador *Firefox*.

Figura 19: Execução no Navegador Firefox



Na Figura 20 é possível observar que a simulação não foi apresentada corretamente em outro navegador, no caso o *Microsoft Edge,* esse problema também se repete em outros navegadores como no *Chrome* e *Opera*. Como pode ser observado na figura 19, o escalonamento não ocorre de forma correta, e acaba deixando uma lacuna no diagrama de uso da CPU, o processo "A" tem o tempo de chegada igual a 0, ou seja, deveria ter aparecido no tempo 0 do diagrama, mas ele só aparece no final da simulação.

Figura 20: Execução no Navegador Microsoft Edge

Expandir a compatibilidade para navegadores como *Chrome, Safari* e *Edge* e outros foi essencial para garantir uma experiência aos usuários, independente do navegador que utilizem, possibilitando uma utilização mais ampla e acessível da ferramenta de simulação de escalonamento de processos oferecida pelo OSLive.

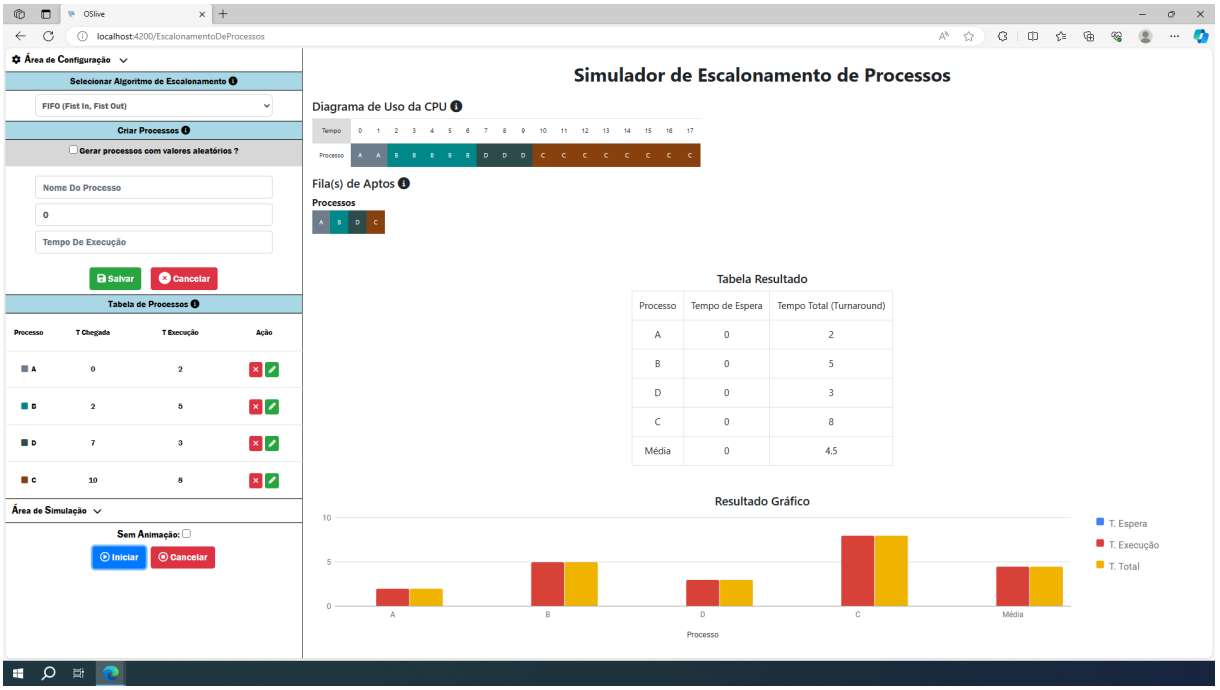
Por fim, um ponto de melhoria identificado na avaliação realizada no TCC (Marinho, 2022), na interface gráfica, foi identificada a necessidade de apresentar informações sobre cada área de importância do módulo de simulação de escalonamento de processo, essa melhoria será apresentada nas próximas seções. A partir dos pontos identificados foi implementada a nova versão do Módulo de Simulação de Escalonamento de Processos. Na próxima seção serão apresentados detalhes das melhorias propostas e desenvolvidas para que os usuários da plataforma utilizem um ambiente mais completo.

4.2 Melhorias de Responsividade da Interface Gráfica no Módulo de Simulação de Escalonamento de Processos

Para aprimorar a responsividade da interface gráfica no módulo de simulação de escalonamento de processos, foram implementadas melhorias no *layout*, garantindo a capacidade da apresentação dos elementos da simulação se adaptar à página, oferecendo experiência ao usuário, o que a plataforma não apresentava. Por isso, o princípio de *design* responsivo envolveu a criação de *layouts* flexíveis, uso adequado de recursos como grades fluidas e media queries, e o ajuste dinâmico de elementos da interface para se adequarem ao tamanho da tela.

Na implementação da versão anterior, foram utilizados *pixels* e *bootstrap* para dividir a tela, o que para criar *layouts* responsivos pode ser problemático, especialmente quando se trata do mal uso do *grid* do *framework Bootstrap* ou de qualquer sistema *grid* responsivo. O principal problema está na falta de flexibilidade que os *pixels* oferecem em tamanhos diferentes de tela, pois, quando são definidos tamanhos fixos em *pixels* para dividir a tela, os elementos da *interface* podem não se ajustar adequadamente e resultar em *layouts* quebrados, sobreposição de elementos. Para sanar esse problema e propor melhorias ao módulo foi utilizado um sistema de *grid* flexível do *Bootstrap*, conforme pode ser observado na Figura 21, a tela responsiva e dinâmica do OSLive.

Figura 21: Nova versão com Responsividade Corrigida



Para garantir a responsividade, a melhoria implementada focalizou principalmente a otimização do sistema de *grid* do *Bootstrap* na versão 4.1, de forma que foi utilizado o *grid* flexível do *Bootstrap* baseado em porcentagem e unidades relativas, em vez de *pixels* fixos, garantindo que os elementos se adaptam dinamicamente ao tamanho da tela. Essa melhoria incluiu reorganização de conteúdo, redimensionamento de gráficos e adaptação da navegação para facilitar o uso em telas menores, o que promoveu uma distribuição adequada dos espaços destinados à *sidebar* e à área de simulação, evitando a sobreposição indesejada dos componentes. Além disso, houve uma refatoração do código *CSS* para garantir uma experiência responsiva em telas que variam de *tablets* e *desktops*, estabelecendo uma base sólida para a adaptabilidade da interface em diferentes dispositivos.

Portanto, essas mudanças não apenas melhoraram a aparência visual do módulo, mas também reforçaram a execução, proporcionando uma experiência agradável para os usuários.

4.3 Adição de Verificação no Cadastro de Processos

Com o intuito de impossibilitar o cadastro de processos com o mesmo tempo de chegada, foi adicionada a funcionalidade de verificação dos cadastros de processos, que consiste em verificar se já existe um processo com o mesmo tempo de chegada passado pelo processo atual, assim barrando o seu cadastro, para que a simulação não seja executada da forma incorreta.

Figura 22: Verificação do tempo de chegada no processo

Na Figura 22, apresenta a funcionalidade de verificação da entrada de um processo no Simulador de Escalonamento de Processos. A interface impede a entrada de múltiplos processos com tempos de chegada idênticos, assegurando a integridade da simulação. Um aviso visual é apresentado na parte inferior da tela, alertando o usuário sobre a tentativa de registro de um processo com tempo de chegada já existente. O sistema requer que cada processo tenha um tempo de chegada único para prosseguir com o cadastro, uma etapa para evitar conflitos e garantir a precisão da simulação do algoritmo de escalonamento.

4.4 Ordenação da Tabela de Processos de Valores

No processo de ordenação de valores aleatórios na tabela de processos, após o usuário clicar no botão “Gerar Processo”, o sistema gerará uma certa quantia de processos dependendo da quantidade de filas que o usuário selecionar. Ou seja, o sistema só irá gerar mais de 4 processos aleatórios caso o usuário selecione os escalonamentos de múltiplas filas. Para os demais algoritmos o sistema gerará como padrão 4 processos aleatórios. Depois que todos os processos são gerados, o sistema utiliza uma função que usa o tempo de chegada dos processos para ordenar a lista em ordem crescente.

Todavia, se o usuário for adicionar os processos manualmente, o sistema, após cada criação de processo irá fazer a verificação na lista com a função que usa o tempo de chegada dos processos para ordenar a lista, diferentemente da opção de geração aleatória, que faz a ordenação só depois de todos os processos são criados. Na Figura 23 pode ser observado que os processos estão ordenados por tempo de chegada.

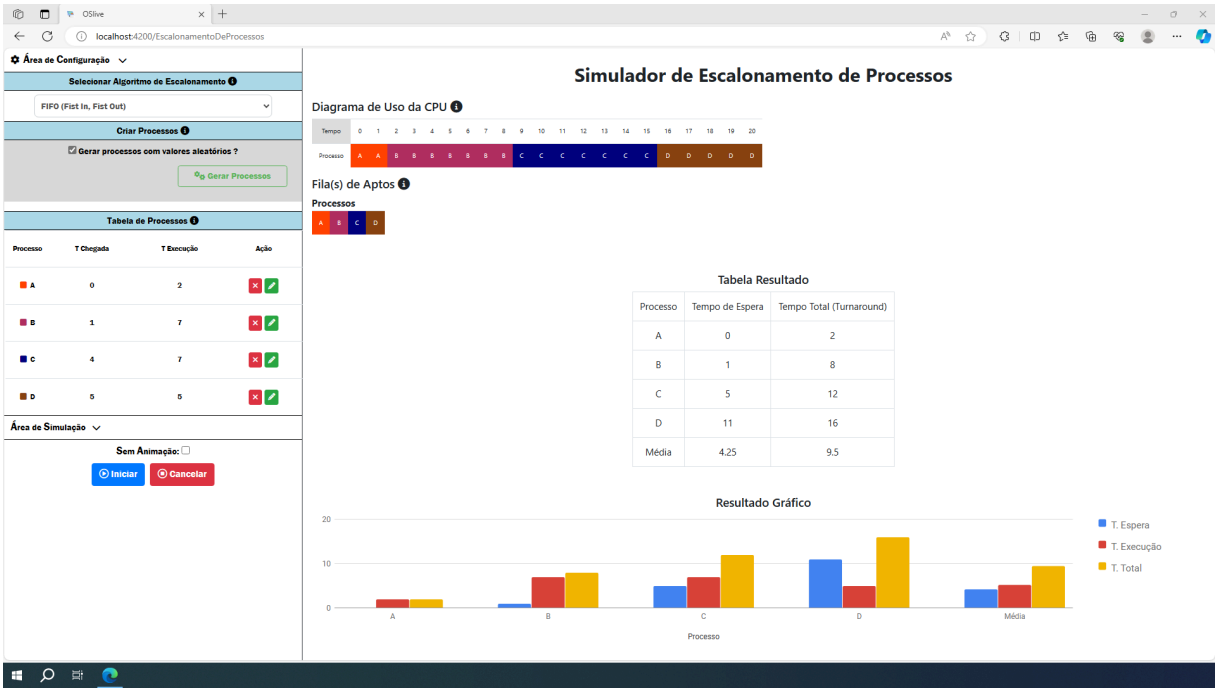
Figura 23: Nova Tabela de Processos Ordenada

A Figura 23 apresenta uma tabela de processos ordenada pelo tempo de chegada, que foi gerada na execução da nova versão do módulo de simulação. Essa modificação irá facilitar ao usuário compreender a simulação de escalonamento dos processos, pois proporciona uma visualização clara e sequencial das etapas de processamento. Isso permite acompanhar a ordem em que os processos são executados e entender como as decisões de escalonamento afetam o desempenho e a eficiência do sistema. Além disso, ao observar a tabela, o usuário pode identificar padrões e otimizar estratégias de gerenciamento de processos, tornando a experiência mais compreensível.

4.5 Navegação Multiplataforma do OSLive

Com o objetivo de solucionar o desafio enfrentado pela versão original do OSLive, que se limitava à execução da simulação somente no navegador *Firefox,* foi realizada uma refatoração completa do código para a adoção da nova arquitetura do projeto. Isso envolveu a atualização das versões do *Angular* e do *Bootstrap*, permitindo a execução da simulação em diversos navegadores, ampliando significativamente a compatibilidade do OSLive. A versão 14 do *Angular* é em *Typescript* diferente da versão anterior do projeto que era em *Javascript*, com isso foi necessária a mudança de lógica da implementação de cada algoritmo de simulação.

Figura 24: OSLive Executando em Outro Navegador (*Microsoft Edge*)



A Figura 24 exibe a versão atualizada do OSLive, apresentando a execução no navegador *Microsoft Edge* e exibindo a simulação do algoritmo de escalonamento SJF (*Shortest Job First*) corretamente. A nova versão foi executada várias vezes e com entradas diversas em outros navegadores, para observar se sua execução estava de acordo com o esperado, obtendo-se sucesso em todos os testes realizados.

Os resultados dos testes realizados em diferentes navegadores mostraram que o módulo agora é capaz de executar a simulação dos algoritmos e processo de escalonamento de maneira consistente. Consequentemente, a atualização representa não apenas uma resolução para a limitação anterior de execução exclusiva no Firefox, mas também um avanço significativo na usabilidade do OSLive.

4.6 Adição do Ícone de Informação

A fim de acrescentar mais informações sobre cada área relevante do módulo de simulação de escalonamento de processos, foi adicionado um ícone de “info” ao lado de vários elementos do módulo de simulação, com o intuito de minimizar dúvidas sobre aquela seção, como apresentado na Figura 25.

Figura 25: Ícone de Informações

Ao oferecer um ícone de informação em áreas relevantes do módulo de simulação de escalonamento de processos é possível observar que a informação sobre o conceito daquela área foi apresentado, tendo em vista que esse conceito ajuda a compreender os algoritmos e seu funcionamento, assim explicitamente exibe os elementos apresentados na tela o que resulta a melhor compreensão sobre a simulação.

5 CONSIDERAÇÕES FINAIS

O OSLive é uma plataforma interativa desenvolvida com o objetivo de facilitar o ensino e a aprendizagem de conceitos fundamentais de sistemas operacionais. Este ambiente virtual proporciona uma experiência prática e engajadora para os alunos, permitindo a simulação e a visualização de processos operacionais em tempo real. A plataforma destaca-se por sua capacidade de demonstrar conceitos complexos de maneira intuitiva e acessível, contribuindo significativamente para o entendimento prático dos tópicos abordados em cursos de Ciência da Computação e áreas relacionadas.

Dentre os vários módulos disponíveis no OSLive, um dos mais relevantes é o módulo de simulação de algoritmos de escalonamento. Este módulo permite aos usuários simular e analisar diferentes algoritmos de escalonamento de processos, um conceito crucial no estudo de sistemas operacionais. Ao oferecer uma ferramenta visual e interativa, o módulo facilita a compreensão dos estudantes sobre como diferentes algoritmos influenciam a execução e a priorização de processos em um sistema operacional.

Recentemente, este módulo passou por atualizações significativas, motivadas pela necessidade de aprimorar a experiência do usuário e corrigir problemas identificados. As atualizações foram baseadas em feedbacks recebidos de usuários e em uma análise detalhada das funcionalidades do módulo. Essas melhorias incluem a implementação de verificações aprimoradas no cadastro de processos, a ordenação da tabela de processos para facilitar a visualização, melhorias na responsividade da interface e na navegação multiplataforma.

O trabalho no OSLive abre caminho para pesquisas e desenvolvimentos futuros. Integrar novas tecnologias, como uma fila de aptos animada para todos os tipos de algoritmos de escalonamento, pode agregar mais funcionalidades à plataforma ou uma pesquisa sobre a eficiência das mudanças implementadas neste projeto. Em resumo, este projeto não apenas resolveu desafios técnicos, mas também estabeleceu uma base para inovações futuras, que está acessível no Gitlab do projeto.

REFERÊNCIAS

Angular.io. What is Angular. 2022. Disponível em: <https://angular.io/guide /what-is-angular>. Acesso em: 07 maio 2023.

BISHOP, M. Computer Security: Art and Science. Boston, MA: Addison-Wesley, 2002.

BOOTSTRAP. About bootstrap. 2021. Disponível em: <https://getbootstrap.com/docs /4.1/about/overview>. Acesso em: 07 maio 2023.

BRASIL. Ministério da Educação. Conselho Nacional de Educação. Parecer CNE/CES nº 136/2012. Diretrizes Curriculares Nacionais para os cursos de graduação em Computação. Relator: Paulo Monteiro Vieira Braga Barone. Diário Oficial da União: seção 1, Brasília, DF, aprovado em 9 mar. 2012.

BROOKSHEAR, J.G.; BRYLOW, D. Computer Science: An Overview. Harlow, England: Pearson, 2014.

CHAN, Wai; SHING, Eusden; HUNG, Jonathan; HUANG, Zhichao. Operating System Privilege: Protection and Isolation. [S.l.], 5 abr. 2005. Notas de aula da CS 111, Harvard University. Disponível em: <https://read.seas.harvard.edu/~kohler/class/05s-osp/notes/notes9.html>.

COMER, D. Operating System Design: The XINU Approach. Boca Raton, FL: CRC Press, 2008.

COULOURIS, G.; DOLLIMORE, J.; KINDBERG, T. Distributed Systems: Concepts and Design. Harlow, England: Pearson, 2011.

DEITEL, H.M.; DEITEL, P.J. Operating Systems. Harlow, England: Pearson, 2003.

DE JONG, Ton. Learning and instruction with computer simulations. Education and Computing, v. 6, n. 3-4, 1991.

ENGINEERING LIBRETEXTS. Memory Partitioning. Disponível em: <https://eng.libretexts.org/Courses/Delta_College/Introduction_to_Operating_Systems/12%3 A_Memory_Management/12.04%3A_Memory_Partitioning>.

FLYNN, I.; MCHOES, A. Understanding Operating Systems. Boston, MA: Cengage Learning, 2017.

LAUDON, K.C.; LAUDON, J.P. Management Information Systems: Managing the Digital Firm. Harlow, England: Pearson, 2016.

MARINHO, Ana C. OSLIVE: avaliação da qualidade pedagógica da ferramenta com base na taxonomia TARDA. 2022. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação). Centro Universitário Luterano de Palmas, Palmas, Tocantins, 2022. Disponível em: <http://ulbra-to.br/bibliotecadigital/publico/home/documento/3739>.

OLIVEIRA, Rômulo S.; CARISSIMI, Alexandre S.; TOSCANI, Simão S. Sistemas operacionais. V11 (Livros didáticos informática UFRGS). Grupo A, 2001.

PATTERSON, D.A.; HENNESSY, J.L. Computer Organization and Design: The Hardware/Software Interface. Waltham, MA: Morgan Kaufmann, 2013.

SARDINHA, C. B. F. DESENVOLVIMENTO DE UM MÓDULO PARA A RESOLUÇÃO DE EXERCÍCIOS SOBRE PAGINAÇÃO POR DEMANDA PARA O AMBIENTE OSLIVE. 2022. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação). Centro Universitário Luterano de Palmas, Palmas, Tocantins, 2022. Disponível em: <http://ulbra-to.br/bibliotecadigital/publico/home/documento/2481>.

SHANNON, Robert E. INTRODUCTION TO SIMULATION. Texas - Tx: Department Of Industrial Engineering Texas A&M Univerity College Station, 1992.

SILBERSCHATZ, A.; GALVIN, P.B.; GAGNE, G. Fundamentos de Sistemas Operacionais. Hoboken, NJ: Wiley, 2013.

SMITH, J.E.; NAIR, R. Virtual Machines: Versatile Platforms for Systems and Processes. Amsterdam, The Netherlands: Elsevier, 2005.

STALLINGS, W. Operating Systems: Internals and Design Principles. Harlow, England: Pearson, 2012.

TANENBAUM, A.S.; BOS, H. Modern Operating Systems. Harlow, England: Pearson, 201