

Organização Geral do Projeto:

Na pasta inicial está localizado o index.html, a pagina inicial onde o usuário terá o primeiro contato com o site. Ademais, dentro dessa pasta existem outras cujas especificações serão apresentadas agora:

CSS: Essa pasta contém todos os arquivos css de todas as páginas.

Style: css da página index

Forms: css das páginas adicionar, cadastro e login.

Noticia: css da página de notícia.

Inicio: css da página de início do site.

HTML: Pasta que contém todos os arquivos html do site, com exceção do index.

Inicio: pagina inicial onde as noticias serão mostradas para o usuário escolher uma para ler.

Noticia: onde a noticia é mostrada com mais profundidade ao usuário.

Cadastro: página de cadastro, essencial para que seja possível adicionar noticias

Login: login de usuário.

Adicionar: página de adicionar para notícias.

Imagens: Aqui todas as imagens do site estão armazenadas.

JavaScript: contém um único arquivo contendo todas as funcionalidades do site, as quais serão explicadas a seguir.

Transição suave entre páginas

Essa funcionalidade é ativada quando o documento HTML é carregado (DOMContentLoaded). Ela adiciona a classe visível ao corpo da página ao concluir o carregamento, permitindo que a transição de aparência ocorra de maneira fluida. Efeito de scroll na página início.html. Essa função observa o movimento de rolagem do usuário para aplicar efeitos visuais. O comportamento é dividido em duas etapas:

Monitoramento de rolagem:

O evento scroll verifica se o usuário está rolando para baixo ou para cima comparando a posição atual da rolagem (scrollTop) com a última posição registrada (scrollTop). Caso a rolagem seja para baixo, a função effectScroll() é chamada.

Aplicação do efeito:

A função effectScroll() identifica os elementos com a classe oculto-2 e, com um intervalo de 300ms entre cada elemento, adiciona a classe visível-2. Esse processo permite que os elementos apareçam de maneira gradual e sequencial, criando um efeito dinâmico e atraente para o usuário.

Essas funcionalidades aprimoram a experiência de navegação, tornando-a mais imersiva e esteticamente agradável.

Página de Cadastro:

1- Identificação do usuário logado:

A variável `usuarioLogado` recupera as informações do usuário atualmente autenticado no sistema, armazenadas no `localStorage` sob a chave `'usuarioLogado'`. Essas informações são convertidas de texto para objeto utilizando `JSON.parse()`.

2. Gerador de IDs únicos:

A função `geradorID()` retorna um identificador único baseado no timestamp atual (`Date.now()`), garantindo que cada novo usuário cadastrado tenha um ID exclusivo.

3. Verificação de login antes de acessar páginas de autenticação:

Se um usuário já está logado e tenta acessar as páginas de cadastro (`cadastro.html`) ou login (`login.html`), o sistema exibe um alerta informando que é necessário sair da conta antes de prosseguir. Em seguida, o usuário é redirecionado para a página inicial (`inicio.html`), e o código restante é interrompido com `return`.

4. Cadastro de usuário:

Caso o usuário não esteja logado e esteja acessando a página de cadastro (`cadastro.html`), o sistema habilita o formulário de cadastro. A seguir estão os detalhes dessa funcionalidade:

Validação do formulário:

Ao submeter o formulário, a função verifica se os campos nome e senha foram preenchidos. Caso contrário, um alerta é exibido.

Verificação de duplicidade:

O sistema busca no localStorage a lista de usuários existentes e verifica se o nome já está cadastrado. Se o nome já existe, o cadastro é bloqueado e uma mensagem de alerta é exibida.

Cadastro de novos usuários:

Um novo objeto de usuário é criado contendo um ID único (id), o nome e a senha fornecidos. Esse objeto é adicionado à lista de usuários e armazenado novamente no localStorage. Após o cadastro, o sistema informa o sucesso da operação, redefine o formulário e redireciona o usuário para a página de login (login.html).

Essas funções garantem um processo de autenticação seguro e bem estruturado, promovendo uma experiência consistente para o usuário.

Página de Login:

1. Verificação de login em página de login:

Quando o usuário acessa a página de login (login.html) e não está logado (não há dados no localStorage), o formulário de login é configurado para capturar os dados de login. A função realizarLogin() é chamada ao enviar o formulário, e a validação dos campos é feita antes de prosseguir.

2. Validação dos campos de login:

A função `realizarLogin()` valida se os campos `nome-login` e `senha-login` não estão vazios. Caso algum campo esteja vazio, um alerta é mostrado e o login não é processado.

3. Autenticação do usuário:

O sistema busca no `localStorage` todos os usuários cadastrados e procura pelo nome fornecido no campo de login.

Caso o nome seja encontrado, ele verifica se a senha fornecida corresponde à senha cadastrada.

Se os dados estiverem corretos, o usuário é considerado autenticado, e as informações do usuário são armazenadas no `localStorage` sob a chave `'usuarioLogado'`. O usuário é redirecionado para a página inicial (`inicio.html`), e um alerta de sucesso é exibido.

Caso a senha esteja incorreta, um alerta informa o erro. Se o nome do usuário não for encontrado, o sistema alerta que o usuário não existe.

Logout do Usuário

1. Verificação do link de logout:

A variável `verifyLogout` captura o link de logout (caso esteja presente na página) e adiciona um evento de clique. Quando o link é clicado, a função `logout()` é chamada.

2. Função de logout:

A função `logout()` remove os dados de login armazenados no `localStorage` utilizando `localStorage.removeItem('usuarioLogado')`. Após isso, um

alerta informa que o usuário foi deslogado, e ele é redirecionado para a página de login (login.html).

Essas funcionalidades garantem que o usuário possa fazer login e logout de forma segura, com o sistema sempre verificando se as credenciais são válidas e armazenando ou removendo as informações conforme necessário.

Página de Adicionar notícias:

Verificação de Login e Permissão de Acesso:

Obter o usuário logado:

A função começa verificando se há um usuário logado, recuperando os dados do localStorage usando `JSON.parse(localStorage.getItem('usuarioLogado'))`.

Exibição do Pop-Up:

O código também lida com a exibição de um pop-up. O pop-up pode ser aberto ao clicar no botão identificado por `open-popup` e fechado clicando no botão identificado por `close-popup`. Além disso, se o usuário clicar fora do conteúdo do pop-up, ele será fechado.

Verificação se o usuário está logado e se tem permissão para adicionar notícias:

Se um usuário estiver logado e seu nome for "adm" (o administrador), o sistema permite o acesso à funcionalidade de adicionar notícias.

Caso contrário, se o usuário não for o administrador, um alerta informa que ele não tem permissão para adicionar notícias.

Se o usuário não estiver logado, o sistema alerta que é necessário estar logado e o redireciona para a página de login.

Adição de Notícia

Formulário de Adição de Notícia:

Se o usuário for o administrador, a função adiciona um evento ao formulário de adicionar notícia, que chama a função `addMateria()` quando o formulário é enviado.

Função `addMateria()`:

Dentro da função `addMateria()`, os valores dos campos do formulário são coletados para compor uma nova notícia. A função realiza as seguintes verificações e ações:

Imagem: O código verifica se o usuário selecionou uma imagem para a notícia. Caso a imagem não tenha sido escolhida, um alerta é mostrado e o redirecionamento para a página inicial ocorre.

Conversão da Imagem para Base64: Se a imagem foi selecionada, ela é convertida para um formato Base64 usando a função `convertImageToBase64()`. Isso permite que a imagem seja armazenada como uma string em `localStorage`.

Criação da Nova Notícia: A nova notícia é composta pelos dados do formulário, incluindo título, explicação, subtítulos, textos e a imagem convertida, além do link da notícia.

Armazenamento das Notícias: A nova notícia é adicionada a uma lista de notícias (armazenada no localStorage) e a lista é atualizada.

Redirecionamento e Limpeza de Formulário: Após a notícia ser adicionada com sucesso, o formulário é limpo e o usuário é redirecionado para a página inicial (inicio.html).

Função para Conversão da Imagem (Base64):

A função `convertImageToBase64()` utiliza a API `FileReader` para ler a imagem selecionada como um arquivo e converte-la para uma string em formato Base64, que é passada para a função de callback. Essa string pode ser armazenada e exibida no site sem necessidade de utilizar um servidor externo para hospedagem de imagens.

Página Início:

Verificação da Página e Recuperação das Notícias

Verificação se está na página inicio.html: A função começa verificando se a página atual é inicio.html

utilizando `window.location.pathname.includes('inicio.html')`.

Recuperação das Notícias do localStorage: O código tenta obter as notícias armazenadas no localStorage usando `JSON.parse(localStorage.getItem('noticias'))`. Caso não haja notícias no localStorage, ele define `noticias` como um array vazio.

Seleção do Contêiner de Notícias: A constante `container` armazena o elemento HTML que irá conter todas as notícias, acessado pelo ID `noticias-container`.

Criação dos Elementos HTML para Cada Notícia

Iteração sobre as Notícias: O código utiliza `forEach` para iterar sobre cada notícia armazenada em `noticias` e criar os elementos HTML correspondentes.

Criação da Div de Linha Divisória: Para cada notícia, uma `div` com a classe `linha-divisoria` é criada e adicionada ao contêiner de notícias (`container`), servindo como separador entre as notícias.

Criação da Div da Notícia: Para cada notícia, é criada uma `div` com a classe `oculto-2` e o ID `container2`. Essa `div` representa a estrutura principal da notícia e tem um fundo de imagem definido pela URL da imagem da notícia.

Criação do Título da Notícia: Dentro da `div` da notícia, é criada uma `div` com o ID `title-2`, e um `h1` é adicionado com o título da notícia. O título é retirado da propriedade `título` da notícia.

Criação do Texto de Apresentação da Notícia: Uma nova `div` com o ID `texto-apresentacao2` é criada, contendo um parágrafo (`p`) com a explicação da notícia (`explicacao`). Esse texto apresenta um resumo ou introdução da notícia.

Criação do Botão "Ver Mais": Dentro de `textoApresentacaoDiv`, é criada uma nova `div` para os botões, denominada `button-container`. Em seguida, um link (`a`) é criado para direcionar o usuário à página de detalhes da notícia (`noticia.html`), passando o ID da notícia como parâmetro na URL (`?id=${noticia.id_noticia}`).

Dentro deste link, é criado um botão (`button`) com o texto "Veja Mais", o qual será clicado para abrir a página de detalhes da notícia.

Composição da Estrutura da Notícia: A div de título e a div de texto com o botão "Ver Mais" são agrupadas dentro da div da notícia (noticiaDiv). Depois, esta div é adicionada ao contêiner de notícias (container).

Página Notícias:

Verificação da Página e Processamento Inicial

Verificação da Página noticia.html: O código começa verificando se a página atual é noticia.html com `window.location.pathname.includes('noticia.html')`.

Carregamento e Processamento da Página: O evento `window.addEventListener('load', () => { processarPagina(); })` é usado para chamar a função `processarPagina` assim que a página for totalmente carregada. Isso garante que o processamento dos dados da notícia aconteça após o carregamento completo.

Função processarPagina

A função processarPagina é responsável por:

- Recuperar o usuário logado e as notícias do `localStorage`.
- Extrair o ID da notícia a partir dos parâmetros da URL.
- Encontrar a notícia correspondente ao ID.
- Se o usuário estiver logado, a função `gerenciarBotoes` será chamada para gerenciar a visibilidade de botões específicos.

A função `atualizarPaginaComNoticia` é chamada para exibir o conteúdo da notícia na página.

Função `atualizarPaginaComNoticia`:

- Esta função é responsável por atualizar os elementos da página com os dados da notícia:
- Vídeo: A URL do vídeo do YouTube é embutida em um `iframe` (com base no link da notícia).
- Subtítulos e Texto: Os subtítulos e o conteúdo textual da notícia são atualizados nos elementos correspondentes da página (`title1`, `title2`, `title3`, `texto`, `texto2`, `texto3`).

Função `gerenciarBotoes`

Esta função gerencia a visibilidade de um botão (de exclusão da notícia) com base no usuário logado:

Se o `id` do usuário logado for igual ao `id_user` da notícia, o botão de exclusão (`button2`) terá a classe `visivel2`, tornando-o visível na interface.

Exclusão de Notícia

Exclusão da Notícia: A exclusão é feita quando o usuário clica no botão de exclusão (`button2`). A função `excluir()` é chamada para:

- Verificar se o parâmetro `id` da notícia está presente na URL.
- Buscar a notícia correspondente no `localStorage`.
- Remover a notícia da lista de notícias.
- Atualizar o `localStorage` para refletir a remoção.

- Exibir um alerta informando que a notícia foi excluída e redirecionar o usuário para a página inicio.html.

Verificação do ID: Se o ID da notícia não for encontrado na URL ou se a notícia não existir no localStorage, o código exibe um alerta e impede a exclusão.