Documentação – Gerenciador de tarefas

Teste prático EasyJur

Criado por:
Matheus Teixeira
(35) 9 9835-4552
matheus.silva35235@gmail.com

Sobre o sistema:
O sistema Gerenciador de Tarefas cuida do gerenciamento das tarefas, permitindo criar, editar e excluir tarefas, gerenciar permissões de usuários e concluir tarefas prontas.

Tecnologias em uso:
Linguagens de Programação: HTML, CSS, PHP 8 e JavaScript.

Frameworks e Bibliotecas: Bootstrap 5.3.0, jQuery 3.7.1 e DataTables 1.10.25.

Banco de Dados: MySQL 10.4.0.

Outras Ferramentas: Select2, SweetAlert2 e bibliotecas adicionais para suporte de exportação de dados em PDF, Excel e impressão.

Instalando:

Recomenda-se que utilize o Xampp para apache e BD.

1. Configure o seu banco de dados na porta ‘3306’ e com um usuário com nome “root” sem senha.
2. Necessário a utilização de um servidor apache, PHP 8 e um servidor MySQL +8.0
3. Acesse o GitHub https://github.com/Matheusedus/Tarefas_manager realize o download dos os arquivos e adicione na pasta que aponta ao apache.
4. Na pasta irá encontrar um arquivo chamado task_manager.sql, entre no seu MySQL e crie uma database (schema) com o nome de task_manager e realize a importação do arquivo citado.
5. Abra o navegador e insira a URL referente ao diretório do sistema.
6. O usuário administrador para efetuar o login é admin@admin.com -> Senha: Admin@12

Utilizando a aplicação

Instruções para usar o sistema

Página de Login
Nesta página, o usuário precisa informar seu email e senha para entrar no

sistema. Se não tiver uma conta, o usuário pode clicar em “Cadastre-se” para fazer um novo registro.

Página de Registro
Na página de registro, o usuário precisa fornecer as seguintes informações:
Nome completo
Telefone
Email único (será usado como nome de usuário)
Senha seguindo os critérios de uma senha segura.

Permissões de usuário
Ao fazer um novo registro, o usuário terá somente a permissão de leitura (visualização) no sistema. As outras permissões serão definidas pelo administrador.

Página inicial – pendencias
Depois de entrar no sistema, o usuário será direcionado para a página inicial, que mostrará as tarefas pendentes. Nessa página, o usuário poderá ver as tarefas pendentes e mudar o status de uma tarefa para “Concluído”.

Permissões das tarefas
Existem diferentes permissões relacionadas às tarefas:
Ler(VISUALIZAR/READ/R): permite ver e marcar uma tarefa como concluída.
Atualizar(EDITAR/UPDATE/U): permite alterar as informações de uma tarefa.
Excluir(DELETAR/DELETE/D): permite remover uma tarefa.
Imprimir(IMPRIMIR/PRINT/P): permite gerar relatórios em formato Excel e PDF.
Criar(INSERIR/CREATE/C): permite criar novas tarefas.
Admin(ADMIN/A): permite criar novos usuários, modificar permissões dos usuários e reverter uma tarefa como “Pendente”.

Criar nova tarefa
Na página inicial, há a opção de criar uma nova tarefa. Ao selecionar essa opção, um modal será aberto, onde o usuário poderá digitar o nome da tarefa e uma descrição. Ao salvar, a tarefa será adicionada na tabela sem a necessidade de atualizar a página.

Tabela de tarefas
Na tabela de tarefas, serão mostradas as seguintes colunas:
Nome da tarefa
Descrição
Data de criação
Data de conclusão
Status (verde para concluído e amarelo para pendente)
Coluna “Concluído” com uma caixa de seleção para indicar a conclusão da tarefa. Coluna “Ações” com botões de “Excluir” e “Editar”.

Cabeçalho

No cabeçalho da página, existem as seguintes opções:
Todas as pendencias: redireciona para uma página parecida com a de tarefas pendentes, mas mostra todas as tarefas, incluindo as concluídas.
Gerenciar Usuários: acesso restrito aos usuários com permissão de admin. Nessa página, é possível editar informações do usuário, como senha, telefone e nome, além de gerenciar as permissões do usuário, adicionando ou removendo.
Botão “Sair”: encerra a sessão e faz o logout do usuário.

Observação: As informações de permissões são armazenadas na sessão quando o usuário entra no sistema, permitindo recuperá-las e fazer as verificações necessárias durante o uso da aplicação.

O fluxo de dados na aplicação segue está sequência

O usuário interage com a interface do usuário, realizando ações como criar, editar ou excluir uma tarefa.

O arquivo JavaScript correspondente à ação desejada é ativado. Por exemplo, ao clicar no botão de criação de uma tarefa, o arquivo home.js é executado.

O arquivo JavaScript usa a biblioteca jQuery para fazer uma solicitação AJAX para o arquivo do roteador apropriado para a ação. Por exemplo, ao criar uma tarefa, a solicitação é enviada para o arquivo taskManagerRouter.php.

O roteador, responsável por encaminhar as solicitações, analisa a solicitação recebida e identifica a ação desejada.

O roteador cria uma instância da classe do controlador apropriado para lidar com a ação. Por exemplo, ao criar uma tarefa, o controlador TaskManagerController é instanciado.

O controlador recebe a solicitação e executa a lógica de negócios necessária. Ele pode interagir com a classe do modelo correspondente para buscar ou modificar os dados no banco de dados. Por exemplo, o controlador TaskManagerController pode chamar o método createTask() do modelo TaskManagerModel.

O modelo se comunica com o banco de dados usando as configurações especificadas no arquivo database.php para buscar ou modificar os dados solicitados. Por exemplo, o modelo TaskManagerModel pode executar uma consulta SQL para inserir uma nova tarefa no banco de dados.

O modelo retorna os dados solicitados para o controlador.

Com base nos dados retornados pelo modelo, o controlador decide qual visualização deve ser renderizada. Por exemplo, após criar uma tarefa com sucesso, o controlador pode decidir renderizar a visualização index.php na pasta home.

A visualização é renderizada com os dados fornecidos pelo controlador e retornada como resposta para a solicitação AJAX.

O arquivo JavaScript, ao receber a resposta AJAX, processa os dados retornados e atualiza a interface do usuário para refletir as alterações realizadas.

Esse fluxo de dados permite uma interação suave entre a interface do usuário, o JavaScript, os controladores e modelos da aplicação, garantindo a manipulação correta e a atualização dos dados no banco de dados, bem como a exibição dos resultados na interface do usuário.

Descrição das APIs e funcionalidades do sistema

API de Controle de Acesso (/app/routers/accessControlRouter.php)
Esta é uma API em PHP para controle de acesso que permite realizar operações relacionadas a usuários e permissões. A classe AccessControlController é responsável por receber as solicitações e chamar os métodos apropriados da classe AccessControlModel. Ela verifica as permissões antes de executar determinadas operações.

Alguns dos principais métodos da classe AccessControlController são:

getUsers(): Retorna todas as informações dos usuários, juntamente com suas permissões.

getUserById(): Retorna as informações de um usuário com base no ID fornecido.

check(): Verifica se um usuário possui permissão com base no ID fornecido.

editUser(): Edita as informações de um usuário com base nos dados fornecidos.

getUserPermissions(): Retorna as permissões do usuário atualmente logado.

getPermissionsByUser(): Retorna as permissões de um usuário com base no ID fornecido.

editUserPermissions(): Edita as permissões de um usuário com base nos dados fornecidos.

A classe AccessControlModel realiza operações de consulta, atualização e exclusão no banco de dados relacionadas a usuários e permissões. Ela usa a classe Connection para estabelecer a conexão com o banco de dados.

Essa API permite visualizar, criar, editar e excluir usuários, bem como gerenciar suas permissões. Também é possível verificar as permissões de um usuário e editar suas permissões.

API do Gerenciador de Tarefas (/app/routers/taskManagerRouter.php)
Essa é uma API em PHP que implementa um sistema de gerenciamento de tarefas. A classe TaskManagerModel é responsável por interagir com o banco de dados e possui métodos para obter tarefas, criar tarefas, editar tarefas, excluir tarefas, visualizar tarefas e atualizar o status de uma tarefa. A classe TaskManagerController é responsável por receber as solicitações e chamar os

métodos apropriados da classe TaskManagerModel. Ela também realiza verificação de permissões antes de executar determinadas operações.

Principais métodos da classe TaskManagerController

getTasks(): Retorna todas as tarefas ou apenas as tarefas pendentes, dependendo do parâmetro callGetTasks passado na requisição.

createTask(): Cria uma nova tarefa com base nos dados fornecidos na requisição.

editTask(): Edita uma tarefa existente com base nos dados fornecidos na requisição.

deleteTask(): Exclui uma tarefa com base no ID fornecido na requisição.

viewTask(): Retorna os detalhes de uma tarefa com base no ID fornecido na requisição.

updateTaskStatus(): Atualiza o status de uma tarefa com base no ID e no status fornecidos na requisição.

A classe TaskManagerModel realiza operações de consulta, inserção, atualização e exclusão no banco de dados. Ela usa a classe Connection para estabelecer a conexão com o banco de dados e possui métodos para obter todas as tarefas, obter tarefas pendentes, criar tarefa, editar tarefa, excluir tarefa, visualizar tarefa e atualizar o status da tarefa.

Ambas as classes dependem do arquivo routes.php para importar as configurações de rota e localização dos arquivos necessários.

Essa API implementa um sistema simples de gerenciamento de tarefas, onde é possível realizar operações como criar, editar, excluir, visualizar e atualizar o status das tarefas.

API de Login (loginRouter.php): A API de Login é responsável por lidar com as funcionalidades de login e registro de usuários. Ela segue uma arquitetura MVC (Model-View-Controller) para separar as responsabilidades e facilitar a manutenção do código.

A classe LoginController é responsável por receber as solicitações relacionadas ao login e registro de usuários. Ela chama os métodos apropriados na classe LoginModel para executar as operações necessárias.

Principais métodos da classe LoginController

login(): Realiza o processo de login do usuário. Ele recebe o email e a senha fornecidos, valida os dados e verifica se o usuário existe no banco de dados. Se o login for bem-sucedido, as permissões do usuário são armazenadas na sessão.

register(): Realiza o processo de registro de um novo usuário. Ele recebe os dados do formulário, como nome, email, telefone e senha. Antes de registrar o usuário, ele verifica se o email já está em uso. Se a senha atender aos critérios de segurança definidos, o usuário é registrado no banco de dados.

checkLogin(): Verifica se o usuário está logado. Ele é chamado em determinadas rotas para garantir que o usuário esteja autenticado antes de acessar determinadas páginas ou executar determinadas ações.

A classe LoginModel é responsável por lidar com as operações de banco de dados relacionadas ao login e registro de usuários. Ela utiliza a classe Connection para estabelecer a conexão com o banco de dados.

Principais métodos da classe LoginModel

register(): Registra um novo usuário no banco de dados. Ele verifica se o email fornecido já está em uso e, em seguida, insere os dados do usuário na tabela correspondente. Também é realizada uma transação para garantir a consistência dos dados.

login(): Realiza a autenticação do usuário. Ele verifica se o usuário existe com o email fornecido e, em seguida, verifica se a senha fornecida está correta. Se o login for bem-sucedido, as permissões do usuário são retornadas.

checkExistingEmail(): Verifica se um email já está em uso por outro usuário. Ele realiza uma consulta no banco de dados para verificar se há algum registro com o email fornecido.

Essa API permite que os usuários realizem o login, registrem-se como novos usuários e verifiquem se estão autenticados. Ela garante a segurança das senhas armazenando-as de forma criptografada no banco de dados.

A aplicação Task Manager segue a arquitetura MVC (Model-View-Controller), que é um padrão de design de software amplamente utilizado para separar as preocupações relacionadas à lógica de negócio, à interface do usuário e ao gerenciamento de dados. Essa arquitetura proporciona uma organização eficiente e uma divisão clara das responsabilidades, facilitando o desenvolvimento, a manutenção e o teste da aplicação.

O banco de dados chamado “Task Manager” (versão do servidor 10.4.27) é composto por quatro tabelas principais:

permissions: Esta tabela armazena as permissões disponíveis. Ela possui três colunas: ‘id’ (um identificador único para cada permissão), ‘description’ (uma descrição detalhada da permissão) e ‘acronym’ (uma sigla representando a permissão).

tasks: Esta tabela guarda informações sobre as tarefas. Ela contém colunas para ‘id’ (um identificador único para cada tarefa), ‘user_id’ (o ID do usuário responsável pela tarefa), ‘name’ (o nome da tarefa), ‘description’ (uma descrição detalhada da tarefa), ‘created_at’ (a data e hora de criação

da tarefa), ‘finished_at’ (a data e hora de conclusão da tarefa, que pode ser nula) e ‘status’ (o status atual da tarefa).

user_permission: Esta tabela mapeia os usuários às suas permissões. Ela tem três colunas: ‘id’ (um identificador único para cada relação usuário-permissão), ‘user_id’ (o ID do usuário) e ‘permission_id’ (o ID da permissão).

users: Esta tabela armazena informações sobre os usuários. Ela inclui colunas para ‘id’ (um identificador único para cada usuário), ‘name’ (o nome do usuário), ‘email’ (o email do usuário), ‘password’ (a senha do usuário), ‘phone’ (o número de telefone do usuário), ‘created_at’ (a data e hora de criação do usuário), ‘updated_at’ (a data e hora da última atualização do usuário) e ‘status’ (o status atual do usuário).

Além disso, existem relações entre essas tabelas. A tabela “tasks” tem uma chave estrangeira (‘user_id’) que se refere à tabela “users” (‘id’). A tabela “user_permission” tem duas chaves estrangeiras: ‘user_id’, que se refere à tabela “users”, e ‘permission_id’, que se refere à tabela “permissions”.

Essa estrutura permite que a aplicação gerencie as permissões dos usuários, acompanhe as tarefas atribuídas a eles e armazene informações sobre as permissões disponíveis, as tarefas e os usuários.