Serviço Storage - Ionic

[Serviço Storage - Ionic - Documentos Google]

O que foi feito:

Adicionamos o serviço `Storage` do Ionic ao projeto para permitir o armazenamento local de dados no dispositivo do usuário.
No momento do login do usuário, o "idpessoa" é recuperado a partir do resultado do login e armazenado no `Storage`.
Em seguida, ao acessar a página de cadastro de higienização, o "idpessoa" armazenado no `Storage` é recuperado e preenche automaticamente o campo "Responsável" no formulário de cadastro.

Por que foi feito:

A necessidade surgiu porque você desejava que, após o usuário fazer o login, o "idpessoa" dele fosse utilizado para preencher o campo "Responsável" em um formulário de cadastro posterior.
Para realizar isso, era importante armazenar o "idpessoa" em um local persistente (que persiste entre as sessões do aplicativo) para que pudesse ser acessado em diferentes páginas e momentos.
O `Storage` do Ionic fornece uma solução prática para o armazenamento local de dados no dispositivo do usuário.

Como foi feito:

Instalação do pacote `@ionic/storage-angular`:
Primeiramente, você instalou o pacote `@ionic/storage-angular` no seu projeto. Esse pacote é uma biblioteca que fornece acesso ao `Storage` do Ionic de forma simplificada.
Importação e Inicialização do `IonicStorageModule`:
No arquivo `app.module.ts`, você importou o `IonicStorageModule` do pacote `@ionic/storage-angular` e o iniciou com a função `forRoot()`. Essa inicialização cria o banco de dados do `Storage` e torna-o pronto para uso em todo o aplicativo.
Armazenamento do "idpessoa" no `Storage`:
No arquivo `tela-inicio.page.ts`, após o login do usuário, o "idpessoa" foi armazenado no `Storage` usando a função `set()` disponibilizada pelo serviço `Storage`. Dessa forma, o "idpessoa" ficou disponível para ser recuperado em outras partes do aplicativo.
Recuperação do "idpessoa" e preenchimento do campo "Responsável":
No arquivo `add-higienizacao-ambiente.page.ts`, na função `ngOnInit()`, você recuperou o "idpessoa" armazenado no `Storage` usando a função `get()` do serviço `Storage`. Em seguida, o "idpessoa" foi atribuído à variável `fk_funcionario_resp_higienizacao`, que preenche automaticamente o campo "Responsável" no formulário de cadastro.

Utilização do "idpessoa" no cadastro de higienização:
No mesmo arquivo `add-higienizacao-ambiente.page.ts`, na função `cadastrar()`, o "idpessoa" recuperado do `Storage` foi usado como valor para o campo "Responsável" no objeto `dados` antes de ser enviado para a API.

Com essas etapas, o `Storage` do Ionic foi corretamente utilizado para armazenar o "idpessoa" do usuário logado e preencher automaticamente o campo "Responsável" no formulário de cadastro de higienização. Isso possibilitou que você mantivesse esse valor persistente entre diferentes páginas e sessões do aplicativo.

Espero que esta explicação detalhada tenha sido útil para compreender o que foi feito, por que foi feito e como tudo funciona. Se tiver mais alguma dúvida ou precisar de mais esclarecimentos, sinta-se à vontade para perguntar! Estou aqui para ajudar.

Para que serve o serviço Storage ?

O serviço `Storage` no Ionic é uma ferramenta que permite armazenar dados localmente no dispositivo do usuário, facilitando o armazenamento e recuperação de informações entre sessões ou após o fechamento do aplicativo.

O `Storage` é útil para:

Armazenamento local de dados: Ele permite salvar informações relevantes, como configurações, preferências do usuário ou tokens de autenticação, diretamente no dispositivo. Isso torna possível acessar esses dados sem a necessidade de uma conexão de rede, melhorando a experiência do usuário e a eficiência do aplicativo.
Persistência de dados: Os dados armazenados no `Storage` permanecem disponíveis mesmo após o usuário fechar o aplicativo ou reiniciar o dispositivo. Isso possibilita que o aplicativo lembre-se das preferências do usuário e do estado anterior quando ele voltar a usá-lo.
Cache de dados: O `Storage` pode ser usado como um cache local para armazenar temporariamente dados frequentemente acessados pelo aplicativo. Isso ajuda a reduzir a quantidade de solicitações à API e melhora o desempenho, especialmente em condições de conexão instável.

O `Storage` é uma maneira conveniente e segura de armazenar dados localmente, especialmente dados que não precisam ser compartilhados com o servidor ou que são acessados com frequência. Existem duas opções principais de `Storage` no Ionic.