

Oi, time da Pipo! Espero que estejam bem.

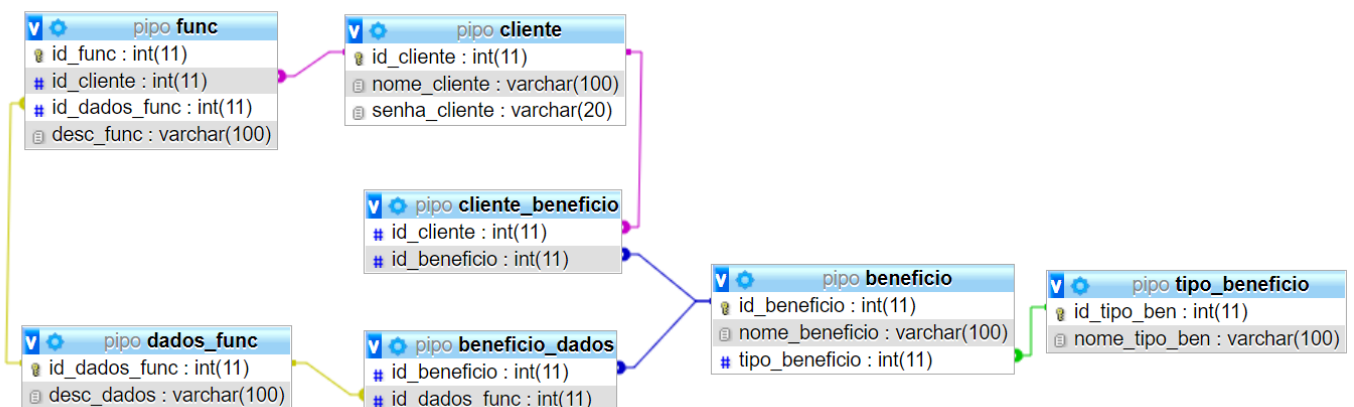
Neste documento vou explicar as minhas escolhas e meu raciocínio para desenvolver a solução do Exercício de Engenharia.

A minha primeira decisão foi qual linguagem eu usaria para desenvolver e se eu focaria apenas no backend ou se dividiria meu tempo com o frontend também. Pensei inicialmente em Python para desenvolver apenas o backend, uma vez que é a linguagem de programação mais próxima de mim no momento, já que desenvolvi meu mestrado em Python e faz 2 anos que não estudo outra linguagem.

Assim, comecei a desenvolver toda a lógica da solução em Python, porém não estava feliz com algumas soluções que eu estava propondo. Lembrei que para o desenvolvimento do meu TCC no técnico de informática, em 2015, eu utilizei PHP e HTML, juntamente com MySQL. Decidi visualizar alguns arquivos para lembrar e acabei optando por realizar a solução do exercício dessa maneira. Por não estudar a linguagem há 7 anos, fiquei receosa por estar desatualizada demais para desenvolver. Por isso fui pesquisando e me divertindo a cada coisa que eu lembrava. Agradeço por vocês terem me proporcionado isso :)

Portanto, desenvolvi a solução em PHP, HTML e MySQL, melhorando um pouquinho o frontend com Bootstrap (mas meu foco não foi o frontend).

Primeiramente, criei tabelas no banco de dados para representar todas as informações e planilhas apresentadas. Tentei criar de uma maneira que a inclusão de novos benefícios e clientes fosse facilitada. Segue abaixo a imagem do banco (pipo.sql):



A tabela benefício tem seu código, nome e o tipo do benefício (Plano de Saúde, Plano Odontológico ou Plano de Saúde Mental), além de cada benefício ter os dados necessários para a inclusão de um novo funcionário (tabela benefício_dados) e também os clientes da Pipo que oferecem aquele benefício (tabela cliente_beneficio). Cada cliente tem seu código, nome e senha, além dos funcionários (tabela func) e os benefícios oferecidos. Cada funcionário tem seu código, a empresa em que trabalha, quais informações estão cadastradas (tabela dados_func) e a informação em si.

Ao começar a desenvolver as páginas PHP, decidi que começaria a desenvolver desde o primeiro acesso do cliente para iniciar a inclusão de um novo funcionário. Por isso a página inicial (inicio.php) requer o nome do cliente e, em seguida (entrar.php), um código de acesso ou senha daquele cliente (senha.php), para que não haja funcionários sendo cadastrados em empresas erradas.

Selecione a sua empresa:

<input type="text" value="Digite o nome da sua empresa"/>	
Acme Co	<input type="button" value="Selecionar empresa"/>
Tio Patinhas Bank	<input type="button" value="Selecionar empresa"/>

inicio.php

Página para incluir funcionário nos benefícios

Sua empresa:

Acme Co

Informe seu código de acesso ou senha:

<input type="text"/>
<input type="button" value="Entrar"/>

entrar.php

Após verificar se a senha ou código de acesso do cliente está correta, a página para cadastrar um novo funcionário (cadastro.php) mostra no canto esquerdo quais os benefícios oferecidos por aquela empresa e também os campos a serem preenchidos para o cadastro do novo funcionário.

Para que os dados não se repetissem, eu criei tabelas que indicam quais informações são necessárias para quais benefícios. Assim, ao saber a empresa que está cadastrando um novo funcionário, eu sei quais benefícios ela oferece. Sabendo os benefícios, eu sei quais informações são necessárias. No PHP, coloquei os códigos dessas informações em um array e depois selecionei os valores únicos, para retirar os dados duplicados e o cliente não precisar digitar os dados mais de uma vez, evitando possíveis erros que ocorriam com planilhas separadas e diferentes para cada benefício.

Como as tabelas indicam quais os dados necessários para cada benefício, além de não ter dados duplicados, cada benefício tem acesso apenas aos dados que precisam.

Empresa: **Acme Co**

Os benefícios oferecidos são:

Norte Europa


Dental Sorriso

Para incluir um novo funcionário, informe os dados abaixo:

Nome

CPF (apenas numeros)

Data admissao

Email

Peso

Altura

cadastro.php

Nesta página, os dados já podem ser tratados antes de serem cadastrados no banco, como, por exemplo: o campo data de admissão aceitar apenas datas, ser obrigatório o símbolo “@” no campo do e-mail, o campo peso aceitar somente números, o campo nome aceitar apenas letras e o campo CPF verificar se o é um número válido de acordo com as regras de CPF e não aceitar letras. Também é importante não deixar que o usuário cadastre qualquer campo em branco.

A última página (dados_func.php) apresenta a possibilidade de cadastrar outro funcionário ou sair. Neste momento, fiquei com vontade de desenvolver outras páginas para um CRUD completo de funcionários dos clientes, porém decidi focar no que o exercício pedia: facilitar o processo atual de inclusão dos novos funcionários.

O que você gostaria de fazer agora?

Cadastrar outro funcionário

Sair

dados_func.php

Agradeço a oportunidade e estou à disposição para esclarecer qualquer dúvida!