

## Pset 7: CC50 Finanças

### Objetivos.

- ♦ Conectá-lo a Web!
- ♦ Introduzir você a HTML, CSS, PHP e SQL.
- ♦ Ensiná-lo a se ensinar novas linguagens.

### Leitura recomendada.

- <http://www.w3schools.com/html/>
- <http://www.w3schools.com/css/>
- <http://www.w3schools.com/php/>
- <http://www.w3schools.com/sql/>

### NOTIFICAÇÃO.

Para este Pset, você é convidado e incentivado a consultar "recursos externos", incluindo livros, a Web e amigos, para aprender mais sobre HTML, CSS, PHP e SQL, desde que o seu trabalho em geral seja seu próprio. Em outras palavras, ainda continua a haver uma linha, mesmo que não precisamente definida, entre aprender com os outros e falar que o trabalho de outras pessoas é seu.

Você pode adotar ou adaptar trechos de código escritos por outras pessoas (encontrados em algum livro, online, ou em outro local), contanto que você cite (na forma de comentários) a origem dos mesmos.

E você pode aprender dos seus colegas, contanto que pedidos de ajuda não se transformem em "me mostre o seu código" ou "escreva isso para mim". Você não pode, para ficar claro, examinar o código dos colegas. Em caso de dúvida quanto à esses quesitos, manda um e-mail para [ajuda@cc50.com.br](mailto:ajuda@cc50.com.br).

## **Honestidade Acadêmica.**

Todo o trabalho feito no sentido do cumprimento das expectativas deste curso deve ser exclusivamente seu, a não ser que a colaboração seja expressamente permitida por escrito pelo instrutor do curso. A colaboração na realização de Psets não é permitida, salvo indicação contrária definida na especificação do Set.

Ver ou copiar o trabalho de outro indivíduo do curso ou retirar material de um livro, site ou outra fonte, mesmo em parte e apresentá-lo como seu próprio constitui desonestidade acadêmica, assim como mostrar ou dar a sua obra, mesmo em parte, a um outro estudante. Da mesma forma é desonestidade acadêmica apresentação dupla: você não poderá submeter o mesmo trabalho ou similar a este curso que você enviou ou vai enviar para outro. Nem poderá fornecer ou tornar as soluções disponíveis para os Psets para os indivíduos que fazem ou poderão fazer este curso no futuro.

Você está convidado a discutir o material do curso com os outros, a fim de melhor compreendê-lo. Você pode até discutir sobre os Psets com os colegas, mas você não pode compartilhar o código. Em outras palavras, você poderá se comunicar com os colegas em Português, mas você não pode comunicar-se em, digamos, C. Em caso de dúvida quanto à adequação de algumas discussões, entre em contato com o instrutor.

Você pode e deve recorrer à Web para obter referências na busca de soluções para os Psets, mas não por soluções definitivas para os problemas. No entanto, deve-se citar (como comentários) a origem de qualquer código ou técnica que você descubra fora do curso.

Todas as formas de desonestidade acadêmica são tratadas com rigor.

## **Licença.**

Copyright © 2011, Gabriel Lima Guimarães.

O conteúdo utilizado pelo CC50 é atribuído a David J. Malan e licenciado pela Creative Commons Atribuição-Uso não-comercial-Compartilhamento pela mesma licença 3.0 Unported License.

Mais informações no site:

<http://cc50.com.br/index.php?nav=license>

**Notas:**

Seu trabalho neste Pset será avaliado em três quesitos principais:

*Exatidão.* Até que ponto o seu código é consistente com as nossas especificações e livre de bugs?

*Design.* Até que ponto o seu código é bem escrito (escrito claramente, funcionando de forma eficiente, elegante, e / ou lógica)?

*Estilo.* Até que ponto o seu código é legível (comentado e indentado, com nomes de variáveis apropriadas)?

## Começando

- ☐ Só faltam dois Psets agora!
- ☐ Para este Pset, considere fazer o download de programas como o **Firefox**, **Firebug**, **Live HTTP Headers** e **Web Developer** (nessa ordem), cada um dos quais está disponível no site do curso em **Software**. Uma vez instalados, Firebug, Live HTTP Headers, e Web Developer irão aparecer como opções no menu de **Ferramentas** do Firefox. Veja se você pode descobrir como cada um funciona simplesmente tentando mexer. É provável que você achará todos esses programas úteis (mas não necessários para este conjunto de problemas).
- ☐ O seu trabalho deve funcionar da mesma forma em pelo menos dois desses navegadores:
  - ☐ Chrome 5.x
  - ☐ Firefox 3.x
  - ☐ Internet Explorer 8.x
  - ☐ Opera 10.x
  - ☐ Safari 5.x

Certifique-se, então, que o seu trabalho funciona em pelo menos dois navegadores. Está tudo bem se você notar algumas pequenas diferenças estéticas entre os dois navegadores. Certifique-se de escrever (na forma de comentário) em qual navegador o seu site é melhor visualizado, para a correção.

- ☐ Para hospedar sites na internet, você vai precisar de um servidor, basicamente um computador que hospeda o seu site. Os seguintes guias mostram a você como transformar o seu próprio computador em um servidor LAMP (Linux ,Apache, MySQL e PHP).

<http://interessespessoais.com/sistemasoperativos/instalar-lamp-ubuntu-1-linha-comando/>

<http://www.lucaskreutz.com.br/blog/2011/07/11/instalando-um-servidor-lamp-no-ubuntu-11-04/>

ou ainda um último guia que mostra a instalação de forma mais “manual”, mas que também funciona

<http://www.gustavomarttos.com/blog/como-instalar-o-lamp-no-ubuntu/>

Agora vamos explicar para que serve tudo isso.

**Linux** é, como você já sabe, o Sistema Operacional. **Apache** é o Servidor em si que possibilita você a colocar sites na internet. **MySQL** é o Banco de Dados, que é simplesmente um gerenciador de dados que ficam guardados no servidor. Em um banco de dados você pode armazenar informações como o login e a senha dos usuários do seu site. **PHP** é a linguagem usada para comunicação com o servidor. É essa linguagem que vocês vão usar para programar o conteúdo dinâmico nos seus sites. Por último, o programa PHPMyAdmin é muito útil para

gerenciar o seu banco de dados utilizando uma tela bem amigável, ao invés de uma interface de comando.

Todos esses programas são Open-source e estão entre os mais utilizados pelos grandes servidores da internet.

Agora que o seu computador está funcionando como um servidor, você pode entrar no seu browser e digitar na URL

`http://localhost/`

e os arquivos na sua pasta `/var/www/` serão lidos pelo browser. Agora tente

`http://localhost/phpmyadmin`

Para acessar o seu banco de dados recém criado!

- ☐ Não passe desse ponto se você não conseguiu acessar o seu servidor em `http://localhost/` ou o seu phpMyAdmin em `http://localhost/phpmyadmin/`! Muitas vezes é preciso reiniciar o Apache para que o phpMyAdmin comece a funcionar e se você tiver alguma dúvida poste o seu problema no forum do CC50!
- ☐ Vá em frente e tente criar uma pasta em `/var/www/` chamada `pset7`.

É provável que você não conseguirá, pois, por padrão, essa pasta não pertence a você, mas ao usuário `root` do computador. Basta então entrar em um terminal e escrever o seguinte comando, onde `usuario` é o nome do seu usuário do Ubuntu (o `usuario` que aparece em `usuario@computador:~$` no seu terminal).

```
sudo chown usuario /var/www
```

Isso vai fazer com que a pasta pertença a você (Eba!). Agora você já pode copiar a pasta `pset7` que veio com esse documento para dentro de `/var/www`.

Agora entre dentro da pasta `includes` que fica dentro da `pset7` que você acabou de copiar e abra o arquivo `constants.php` (com um editor de texto como o Gedit). Observe que o valor da variável `DB_PASS` está faltando. Coloque a senha que você criou para o banco de dados nessa variável, e saiba que o nome de usuário do seu banco de dados será o padrão para administradores: **root**.

Para que você tenha um lugar para armazenar os portfolios dos usuários, nós tomamos a liberdade de criar um banco de dados MySQL só para você para esse Pset.<sup>1</sup> Nós até já o preenchemos com uma tabela! Mas esse banco de dados que nós criamos está salvo em um arquivo, e você vai ter que fazer o upload dele para o seu servidor. Siga para:

---

<sup>1</sup> MySQL é um banco de dados gratuito e de código aberto que o CC50, o Facebook e vários outros sites usam.

`http://localhost/phpmyadmin/`

E forneça o nome de usuário (root) e a senha que você criou ao instalar o servidor LAMP no seu computador. Você finalmente se encontrará na página principal do phpMyAdmin.

Na tela inicial do phpMyAdmin, clique em **Import** e depois escolha o arquivo **finance.sql** que veio junto com esse documento e clique em **Go**.

Finalmente digite a seguinte URL no seu browser

`http://localhost/pset7/`

Você deve encontrar-se na página de login da sua própria cópia do CC50 Finanças! Se alguma coisa parece quebrada, poste o seu problema no fórum do CC50. Não tente registrar ou logar no site ainda!

### Home, sweet home... page.

- ☐ É hora de fazer uma home page! Vá para `/var/www/`. Crie um arquivo chamado `index.html` nesse diretório (delete-o e crie outro, se um já existir) e preencha esse arquivo com HTML válido (ou XHTML).<sup>1</sup> Faça uma home page para você. Obras de arte, embora encorajadas, não são necessárias. Eu não mostrei uma veia artística muito rigorosa nas aulas, afinal. Desde que o HTML seja válido, a sua home page pode conter tanto ou tão pouco conteúdo quanto você queira.

Quando estiver pronto para examinar a sua obra-prima (ou trabalho em andamento), salve o arquivo e, em seguida, siga para a URL.

`http://localhost/`

Uau, essa página é feia. (Ok, talvez não seja.) Mas o ponto é que você agora está na Web! Faça todas as melhorias que você quiser em `index.html`. Você certamente pode, mas não precisa, empregar CSS. Toda vez que você salvar as alterações, não se esqueça de recarregar a página no seu navegador, e sempre que você alterar o CSS de uma página, pode ser que aquela alteração não apareça de imediato pois os browsers costumam armazenar os arquivos `.css` em Cache, para diminuir o tempo de carregamento futuro do mesmo site, por isso você sempre deve limpar o Cache do seu browser para ver as alterações feitas em um arquivo `.css` surtirem efeito.

Em última análise, tenha certeza que o seu HTML é válido de acordo com o W3C Markup Validation Service:

`http://validator.w3.org/`

Não tem problema se esse validador te mostrar alguns avisos (Warnings), mas erros (Errors) identificados por ele não serão perdoados.

---

<sup>1</sup> Qualquer versão do HTML ou XHTML.

## Yahoo!

- ☐ Não se esqueça de ajuda@cc50.com.br !
- ☐ Se você não tem certeza do que significa comprar e vender ações (ações de uma empresa), vá para a URL abaixo e leia um pouco sobre o assunto.

<http://www.mundotrade.com.br/aprendizado/o-que-sao-acoes>

Você está prestes a implementar o CC50 Finanças, uma ferramenta na Web com a qual você pode gerenciar carteiras de ações. Não só essa ferramenta permitirá que você verifique os preços de ações reais em um determinado momento, mas também permitirá que você compre (ok, "compre") e venda (bem, "venda"), as ações!

- ☐ Permitam-me compartilhar uma mensagem da minha caixa de Spam com você.

Prezado investidor:

A5 Laboratories (AFLB.OB) acaba de anunciar o que poderia ser uma das maiores descobertas médicas do século 21.

Esse pode ser um dos mais importantes avanços na medicina em 80 anos.

AFLB.OB pode ter feito o maior avanço em tecnologia da medicina dessa geração.

NÃO PERCA essa oportunidade.

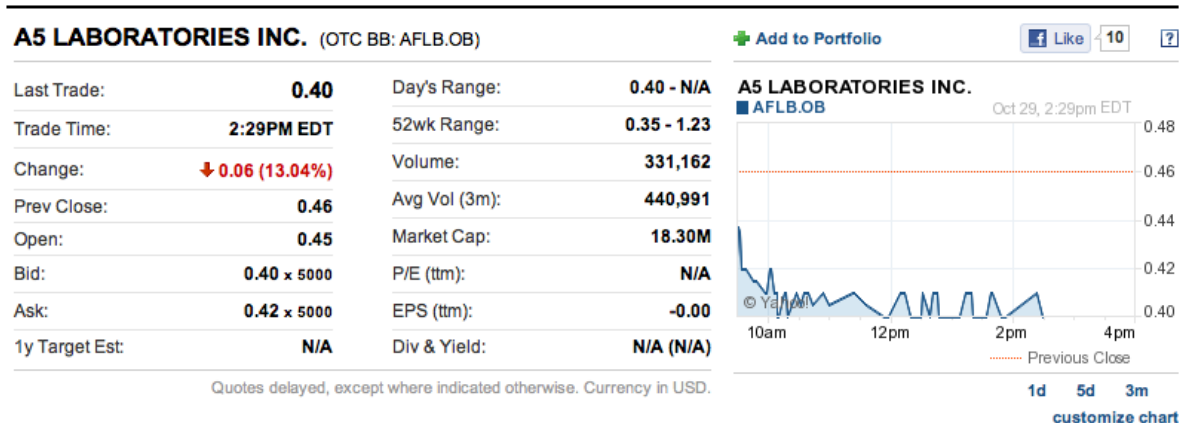
Aqui é para investir solidamente.

Tim Fields,  
Untapped Wealth

- ☐ Uau, que cara legal esse Tim, me dando uma ótima dica de investimento. O Gmail foi muito estúpido para chamar isso de spam! Vamos entrar nessa oportunidade agora. Vá para o Yahoo Finanças na URL abaixo.

<http://finance.yahoo.com/>

Digite o símbolo para A5 Laboratories., AFLB.OB, no campo superior esquerdo dessa página e clique em **"GET QUOTE"**. Você deve ver uma tabela como a abaixo.



Wow, parece que essa ação custa apenas alguns centavos! O que deve ser uma coisa boa. De qualquer modo, observe como Yahoo relata não apenas o preço mais recente de uma ação (**Last Trade**), mas também quando ela foi trocada de mãos pela última vez (**Trade Time**), a porcentagem pela qual o preço da ação mudou ao longo do dia mais recente (**Change**), o preço mais recente de abertura (**Open**), o valor mais alto e o mais baixo do dia (**Day's Range**) e muito mais.

Além disso, olhe um pouco abaixo disso e você deverá ver uma caixa de ferramentas como essa.

#### Toolbox

[Set Alert for AFLB.OB](#)

[Download Data](#)

[Updates on your phone](#)

[Add Quotes to Your Web Site](#)

Add AFLB.OB Headlines to My Yahoo!

Acontece que o Yahoo permite que você baixe todos esses dados sobre as ações. Vá em frente e clique em **Download Data** para baixar um arquivo no formato CSV (valores separados por vírgula). Abra o arquivo no Excel ou qualquer editor de texto (Gedit) e você deve ver uma "linha" de valores, todos extraídos da tabela. Acontece que o link que você acabou de clicar levou à URL abaixo.

<http://download.finance.yahoo.com/d/quotes.csv?s=AFLB.OB&f=s1l1d1t1c1ohgv&e=.csv>

Observe como o símbolo da A5 é incorporado nessa URL (como o valor do parâmetro HTTP chamado *s*) que é como o Yahoo sabe quais dados deve retornar. Observe também o valor do parâmetro HTTP chamado *f*, é um pouco críptico (e sem documentação oficial), mas o valor desse parâmetro informa ao Yahoo quais campos de dados ele deve retornar para você. Se você está curioso para saber como ele faz isso, siga para a URL abaixo.

<http://www.gummy-stuff.org/Yahoo-data.htm>

É interessante notar que um monte de sites que integram dados de outros sites fazem-no através de "screen scraping", um processo que requer escrever programas que analisam HTML



procurando dados de interesse (tarifas aéreas, preços de ações , *etc.*). Escrever um screen scrapper para um site tende a ser um pesadelo, pois o código HTML de um site é geralmente uma bagunça, e se o site altera o formato de suas páginas da noite para o dia, você precisa re-escrever o seu programa.<sup>1</sup>

Felizmente, já que o Yahoo fornece dados em CSV, o CC50 Finanças não precisará fazer nenhum screen scraping. Ele fará o download e a análise dos arquivos CSV. Fique ainda mais feliz, nós escrevemos esse código para você!

Vamos agora voltar a nossa atenção para o código que você recebeu.

- Vá agora para `/var/www/` e abra `index.php` (com algo como o Gedit e não com o seu browser). Você verá uma página HTML bem simples, a mesma página que você visitou mais cedo ao testar o seu servidor (um pouco antes de ser redirecionado para `login.php`). Observe como nós estruturamos a página, com três elementos `div` dentro de `body`, cada qual com um único atributo `id`. Observe as referências a `styles.css` e `logo.gif`. Esses arquivos podem ser encontrados em `pset7/css/` e `pset7/images/`, respectivamente. Nós colocamos os dois arquivos em subdiretórios para manter a pasta `pset7/` mais arrumada. À medida que você avança ao implementar o CC50 Finanças, você é bem vindo a colocar arquivos adicionais em qualquer diretório.

Vá em frente e abra `styles.css`. Esse arquivo define os estilos para vários dos elementos HTML do site, de modo que não temos que incluir atributos `style` para os mesmos elementos em cada arquivo PHP. Não há necessidade de dominar CSS para esse conjunto de problemas, mas saiba que você não pode ter mais de um elemento `div` por página com o mesmo atributo `id`. Um `id` deve ser exclusivo. Em qualquer caso, você está convidado a modificar `styles.css` como achar melhor.

Agora observe que `index.php` "inclui" (*copia e cola*) um arquivo chamado `common.php` que pode ser encontrado em `pset7/includes/`. Qualquer arquivo PHP que você criar para esse Pset que se destina a ser visitado por um usuário também deve conter, antes de qualquer outra coisa, essa mesma linha.

```
require_once("includes/common.php");
```

Note que se você decidir colocar quaisquer arquivos PHP dentro de subdiretórios de `pset7/`, você pode precisar especificar um caminho diferente para `common.php` (*por exemplo* `../includes/common.php`).

Vamos dar uma olhada no código que estamos copiando via `require_once`. Navegue para `pset7/includes/` e abra `common.php`. Cada linha desse arquivo será copiada para `index.php` através do comando `require_once`, cada uma das linhas desse arquivo será executada antes de qualquer outra coisa em `index.php`. As primeiras linhas de `common.php` garantem que você seja informado de erros no seu próprio código através do browser. A função `session_start` garante que você terá acesso a `$_SESSION`, uma variável "superglobal" através da qual nós vamos

---

<sup>1</sup>Veja [http://wiki.cs50.net/Screen\\_Scraping](http://wiki.cs50.net/Screen_Scraping) se estiver curioso sobre como fazer, de qualquer forma.

lembrar quando um usuário estiver conectado<sup>1</sup> As próximas linhas copiam mais três arquivos; nós vamos voltar a eles aos poucos. As próximas linhas de código garantem que os usuários serão obrigados a fazer login para acessar a maioria das páginas. As últimas linhas de código garantem que você esteja conectado ao seu banco de dados, onde você vai armazenar as carteiras de ações dos seus usuários.

Tudo bem, agora abra `constants.php` novamente. Nesse arquivo nós definimos algumas constantes globais. Todos os seus `phps` incluirão `common.php`, que, por sua vez, inclui `constants.php`, portanto você terá acesso a esses valores globais em cada um dos seus `phps`. Observe a URL do Yahoo! Finanças, que deve ser bastante familiar (embora nós tenhamos alterado os seus parâmetros de modo que `s` seja passado). Nessa URL, porém, está faltando um valor para esse parâmetro `s`. Vamos ver por que isso acontece.

Abra `stock.php`, e você verá algo que se assemelha a uma `struct` de C. Na verdade, esse código define uma estrutura ("classe" em PHP) chamada Stock (Ação), cujo objetivo é encapsular dados relacionados a uma ação.<sup>2</sup> Embora o Yahoo ofereça mais campos do que aqueles armazenados por essa estrutura, a nossa classe fornece apenas o básico.

Agora dê uma olhada em `helpers.php`. Você não precisa entender como todo esse código funciona, mas certifique-se de entender o que as funções declaradas nesse arquivo fazem, lendo pelo menos, os comentários delas. Note, particularmente, como `lookup` recebe, como seu único argumento, o símbolo de uma ação, que ele acrescenta a `YAHOO` usando o concatenador de strings do PHP (um simples ponto `."`), a fim de baixar o CSV certo.

Finalmente, dê uma olhada em `apology.php`. Esse arquivo serve como um "modelo" para `apologize` em `helpers.php` para que, através de apenas uma função, você possa pedir desculpas ao usuário devido a todos os tipos de problemas.

Tudo bem, vamos examinar apenas mais três arquivos, todos terminados em `.php`. Navegue de volta para `pset7/` e abra `login.php`. Lembre-se que você foi redirecionado para essa página, quando você tentou acessar `index.php` com o seu navegador. Note que esse arquivo não contém muito código. Na verdade, ele é muito parecido com `index.php`, e é quase inteiramente formado por HTML. Mas é esse HTML que implementa a página de login que você viu. Note que nós criamos um formulário usando uma tabela.<sup>3</sup> Dê uma olhada na seguinte linha:

```
<form action="login2.php" method="post">
```

---

<sup>1</sup> Mesmo que o HTTP seja um protocolo "stateless", onde os browsers se desconectam dos servidores assim que eles terminam de fazer download das páginas, "cookies" deixam que o browser lembre ao servidor quem ele (ou, na verdade, você) é em pedidos de conteúdo subsequentes. PHP usa "cookies de sessão" para dar `$_SESSION` a você, um array associativo no qual você pode guardar qualquer dado que você quiser ter acesso durante a visita inteira de um usuário. No momento que o usuário termina a sua sessão (ao fechar o seu browser), os dados armazenados em `$_SESSION` são perdidos para aquele usuário especificamente, pois na sua próxima visita, o usuário receberá um novo cookie!

<sup>2</sup> Por convenção, nomes de classes começam com letra maiúscula.

<sup>3</sup> Muitos Web Developers não gostam de usar tabelas para o layout de um site, mas elas são úteis em algumas situações. Na verdade, a utilização de tabelas, ao invés de CSS para o layout de sites é atualmente considerada ultrapassada.

Essa linha instrui o seu browser a “mandar” os dados inseridos no formulário para `login2.php` via POST. `login2.php` deve ser, então, quem lida com a autenticação dos usuários. Vamos verificar. Abra `login2.php`.

Acontece que `login2.php` não é muito longo. Sua primeira linha, assim como as de `index.php` e de `login.php`, inclui aquele arquivo `common.php`. Logo depois o input do usuário é “escapado” por fins de segurança usando `mysql_real_escape_string`, para que o banco de dados do CC50 Finanças não caia vítima de um “ataque de injeção de SQL”, no qual um usuário envia código SQL em vez de um nome de usuário e/ou uma senha ao efetuar o login. Veja [http://www.php.net/mysql\\_real\\_escape\\_string](http://www.php.net/mysql_real_escape_string) para referência.

A próxima linha de código prepara uma string de SQL da seguinte forma.

```
$sql = "SELECT uid FROM users WHERE username='$username' AND password='$password'";
```

Para ser claro, suponha que o Presidente Skroob tente logar no CC50 Finanças com seu nome de usuário e sua senha.<sup>1</sup> Essa linha de código irá atribuir a `$sql` o valor abaixo.

```
SELECT uid FROM users WHERE username='pskroob' AND password='12345'
```

Talvez seja desnecessário dizer, mas a próxima linha de `login2.php` executa esse `SELECT` no banco de dados com a função `mysql_query`, armazenando o “conjunto de resultados” (linhas retornadas) em uma variável chamada `$result`. Somente se o `username` e a `password` do Skroob estiverem corretos, o banco de dados retornará de fato um registro. E, assim, se `mysql_num_rows` retorna 1, Skroob foi autenticado com êxito. Nosso código “lembra” disso armazenando o seu ID numérico (`uid`) em `$_SESSION`; Então o nosso caro Presidente é redirecionado para `index.php`, onde o seu portfolio (carteira de ações) o aguarda (assim que você implementá-lo).<sup>2</sup> Se, no entanto, a senha ou o nome de usuário inseridos forem inválidos, ele é informado do fato.

Aliás, por que esse redirecionamento de volta para `index.php`, após a autenticação bem-sucedida, não resulta em um loop infinito? Bem, lembre-se que `index.php` requer `common.php`, que contém o seguinte código.

```
if (!preg_match("/:?log(?:in|out)|register)\d*\.", $_SERVER["PHP_SELF"]))
{
    if (!isset($_SESSION["uid"]))
        redirect("login.php");
}
```

Embora seja um pouco assustador (eu prefiro “elegante”), esse código simplesmente pergunta se `$_SESSION["uid"]` tem algum valor atribuído (por exemplo o ID do Skroob). Se não, deve ser porque ninguém está logado e assim redirecionamos o tráfego para `login.php` (chamando `redirect`, uma função definida em `helpers.php`). Se, porém, `$_SESSION["uid"]` tem um valor definido (provavelmente atribuído por `login2.php`), não vamos redirecionar, vamos, ao contrário, deixar o usuário logado onde ele está. É claro que, se o usuário não logado está em

---

<sup>1</sup> <http://en.wikipedia.org/wiki/Spaceballs#Spaceballs>

<sup>2</sup> Por razões de eficiência, nós não “lembramos” dos usuários pelos seus nomes de usuário, mas sim pelos seus IDs (inteiros) que identificam unicamente cada usuário.

`login.php`, `logout.php` ou `register.php`, esse código não irá redirecioná-lo para que ele não entre em um loop infinito de redirecionamentos, graças à "expressão regular" que passamos a `preg_match`.<sup>1</sup>

Agora, como vamos permitir ao Skroob deslogar do site? Bem, perceba que nós incluímos em `index.php` um link para `logout.php`! Dê uma olhada nesse último. Note que ele chama `logout`, uma função definida em `helpers.php` (nós chamamos essa mesma função no topo de `login.php`). Como alternativa, o Skroob pode simplesmente sair do seu navegador, já que `$_SESSION` estará perdido nesse caso também.

- ☐ Ufa, isso foi muito. Vá fazer um lanche.
- ☐ Tudo bem, vamos falar sobre esse banco de dados que nós sempre mencionamos. Dirija-se agora para o phpMyAdmin e clique no link para `finance` (ao lado da qual há um 1 entre parênteses, o que confirma que uma tabela já espera por você). Na página que aparece, você verá a tabela chamada `users`, clique em Browse nessa tabela para ver o seu conteúdo. Ah, algumas pessoas familiares. De fato, o nome de usuário e a senha do Presidente Skroob estão aqui!

Agora clique na guia rotulada **Structure**. Hmm, alguns campos familiares. Lembre-se que `login2.php` gera consultas como a abaixo.

```
SELECT uid FROM users WHERE username='pskroob' AND password='12345'
```

Como o phpMyAdmin deixa claro, essa tabela chamada `users` contém três campos: `uid` (cujo tipo é `UNSIGNED INT`), juntamente com `username` e `password` (cujos tipos são `VARCHAR`). Parece que nenhum desses campos pode ser `NULL`, e o comprimento máximo de `username` e `password` é 255. Uma característica interessante de `uid`, entretanto, é que ele tem a opção auto-incremento habilitada: ao inserir um novo usuário na tabela, você não precisa especificar um valor para `uid`, o usuário receberá o próximo `INT` disponível. Perceba, por fim, a caixa **Indexes**. Parece que a chave `PRIMARY` dessa tabela é `uid`, o que quer dizer que (como esperado) dois usuários não podem compartilhar o mesmo ID. Claro, `username` também deve ser exclusivo entre os usuários, e por isso nós também o definimos assim. Na verdade, poderíamos ter definido `username` como a chave primária dessa tabela. Mas, em nome da eficiência, a abordagem mais convencional é usar um `INT` como `uid`. Aliás, esses campos são chamados de "índices", porque os bancos de dados tendem a utilizar esses "índices" para encontrar registros mais rapidamente.

Faz sentido? Ok, volte para

`http://localhost/pset7/login.php`






e tente fazer login como Presidente Skroob (cujo nome de usuário e senha você já deve saber). Se tudo ocorreu corretamente, você deve se encontrar em `index.php`, onde (no momento) muito pouco o aguarda.

---

<sup>1</sup> [http://en.wikipedia.org/wiki/Regular\\_expression](http://en.wikipedia.org/wiki/Regular_expression)

- Vá de volta para o phpMyAdmin e clique na aba **Structure** da tabela `users`. Vamos dar a cada um de seus usuários algum dinheiro. No meio dessa página há um espaço onde você pode adicionar um campo (**Add Field**). Simplesmente clique em **Go**.

Através do formulário que aparece, defina um campo do tipo DECIMAL chamado `cash` usando as configurações descritas abaixo, clique em **Save**.

Field	cash
Type 	DECIMAL 
Length/Values <sup>1</sup>	65,4
Default <sup>2</sup>	As defined:  0.0000
Collation	
Attributes	UNSIGNED 
Null	<input type="checkbox"/>
AUTO_INCREMENT	<input type="checkbox"/>
Comments	

Se você der uma olhada na documentação do MySQL em

<http://dev.mysql.com/doc/refman/5.1/en/numeric-types.html>

você verá que os dados do tipo DECIMAL são usados para "armazenar valores numéricos exatos". Um comprimento de 65, 4 para um DECIMAL diz que os valores de `cash` não podem ter mais do que 65 dígitos no total, quatro dos quais podem estar à direita do ponto decimal. (Ooo, frações de centavos. Parece *Office Space*)

Ok, volte para a aba **Browse** e dê a todos os usuários \$10.000,00 manualmente.<sup>1</sup> A maneira mais fácil é clicar em **Check All** e depois, na mesma linha, no ícone que se parece com um lápis. Na página que aparece, mude o 0,0000 para 10000,0000 para cada um dos seus usuários, e em seguida, clique em **Go**. Eles não vão ficar felizes!?

- É chegada a hora de escrever algum código! Vamos deixar que novos usuários se cadastrem.

Volte a sua atenção para `/var/www/pset7/`. Copie agora o arquivo `login.php` e cole-o com o nome `register.php`. Faça a mesma coisa copiando `login2.php` e colando, chamando o novo arquivo de `register2.php`.<sup>2</sup>

Abra `register.php` e mude o `title` em `head` para o que você quiser. Em seguida, altere o valor do atributo `action` do formulário de `login2.php` para `register2.php`. Em seguida adicione uma linha adicional na tabela HTML contendo um novo campo chamado `password2` (queremos que os usuários digitem suas senhas duas vezes para diminuir a chance de erros).

<sup>1</sup> Em teoria, nós poderíamos definir o valor default de `cash` como 10000.000, mas, em geral, é melhor colocar tais configurações no código e não no banco de dados, de forma que elas fiquem mais fáceis de ser alteradas.

<sup>2</sup> Você com certeza pode, particularmente se estiver entre aqueles mais confortáveis, desviar dessas convenções de nomes de arquivos e estruturar o seu site como você achar melhor, desde que os outros requerimentos básicos sejam cumpridos.

Finalmente, altere o botão `submit` de `Log In` para `Registrar` e faça aquele link lá embaixo apontar para `login.php` (para que os usuários possam navegar para outra página, se eles já tiverem uma conta).

Tudo bem, vamos dar uma olhada no seu trabalho! Siga para

`http://localhost/pset7/login.php`

e clique no link dessa página para `register.php`. Se a nova página parece quebrada (ou talvez simplesmente feia), sintá-se livre para fazer alguns ajustes, salvar suas alterações e depois recarregar a página.

Assim que a página parecer ok, volte para seu editor de textos e abra `register2.php`. Nem precisamos dizer que é necessário substituir o código de lá para que ele realmente registre usuários. Permita-nos oferecer algumas sugestões.

- i. Se `$_POST["username"]` ou `$_POST["password"]` foram mandados em branco ou se `$_POST["password"]` não equivale a `$_POST["password2"]`, você vai querer mandar o usuário a uma página que pede desculpas, explicando, pelo menos, um dos problemas.
- ii. Para inserir um novo usuário em seu banco de dados, você pode querer passar à função `mysql_query` uma string como  

```
INSERT INTO users (username, password, cash) VALUES('$username', '$password', 10000.00)
```

apesar de deixarmos para você decidir quanto dinheiro o seu código deve dar a novos usuários. Também tenha certeza de escapar qualquer entrada do usuário (por exemplo `$username` e `$password`) com a função `mysql_real_escape_string` primeiro.
- iii. Saiba que `mysql_query` irá retornar `FALSE` se o `INSERT` falhar (como pode acontecer se, digamos, o `username` já existe).<sup>1</sup> Claro, se você não pode fazer um `INSERT`, certamente deverá pedir desculpas ao usuário, por exemplo chamando `apologize`.
- iv. Se, porém, o seu `INSERT` foi um sucesso, saiba que você pode descobrir qual `uid` foi designado para aquele usuário através da função `mysql_insert_id` logo após `mysql_query`.<sup>2</sup>
- v. Se o registro foi bem sucedido, você deveria efetuar o login do seu novo usuário ("lembrando" o seu `uid` em `$_SESSION`), depois disso simplesmente redirecione-o para `index.php`.

Terminou tudo isso? Pronto para testar? Siga de volta para

`http://localhost/pset7/register.php`

e tente registrar um novo usuário. Se você chegar a `index.php`, é provável que você tenha tido sucesso! Confirmar isso retornando ao phpMyAdmin, clicando uma vez mais naquela aba rotulada **Browse** da tabela `users`. Dê uma olhada se o novo usuário está por lá. Se não, é hora de debugar!

Você deve ter notado, aliás, que `helpers.php` fornece uma função chamada `dump` que cospe para o seu navegador o(s) valor(es) de qualquer variável que você passar à ela, usando

---

<sup>1</sup> Veja [http://www.php.net/mysql\\_query](http://www.php.net/mysql_query) para referência.

<sup>2</sup> Veja [http://www.php.net/mysql\\_insert\\_id](http://www.php.net/mysql_insert_id) para referência.

pset7/includes/dump.php como seu modelo. Por exemplo, se você gostaria de ver o conteúdo inteiro do array `$_POST`, basta adicionar

```
dump($_POST);
```

temporariamente ao seu código onde você quiser, assim como você costumava usar `printf` para debugar os seus programas em C. Note que `dump` foi feito somente para te ajudar a resolver os seus problemas, e não para pedir desculpas aos usuários.

Certifique-se, aliás, que qualquer HTML gerado por `register.php` é válido, verificando-o em <http://validator.w3.org/>!

- ☐ Tenha em mente que você é bem-vindo a mexer e aprender com a nossa implementação, disponível na URL abaixo

<http://cc50.com.br/finance/>

Você está de fato convidado a registrar quantos nomes de usuários (fakes) você quiser, a fim de jogar. E você está convidado a ver o HTML e o CSS das nossas páginas (vendo o código através do seu próprio navegador), de modo que você possa aprender com ou aperfeiçoar os nossos próprios projetos. Se desejar, fique à vontade para adotar o nosso HTML e CSS inteiramente para o seu projeto.

Mas não sinta que você deve copiar o nosso design. Na verdade, você pode modificar cada um dos arquivos que foram dados a você para satisfazer os seus próprios gostos, bem como incorporar suas próprias imagens e muito mais. Na verdade, gostaríamos muito de ver que a sua versão do CC50 Finanças é mais agradável do que a nossa!

- ☐ Agora vamos capacitar os usuários a checar os preços das ações. Vá em frente e crie mais dois arquivos, nomeados `quote.php` e `quote2.php`. Cabe a você se você quer criá-los a partir do zero ou baseá-los em seus arquivos para logins e registros. Em última análise, porém, `quote.php` deve apresentar aos usuários um formulário (que manda as informações digitadas para `quote2.php`) que espera um símbolo de uma ação em um campo de texto. Se esse símbolo for válido, `quote2.php` deve informar ao usuário o preço atual da ação representada por esse símbolo.

E agora? Bem, lembre-se da função chamada `lookup` dentro de `helpers.php`. Por que não chamá-la com um código como o abaixo?

```
$s = lookup($_POST["symbol"]);
```

Assumindo que `$_POST["symbol"]` é não-nulo e contém um símbolo para um ação real, `lookup` irá retornar um "objeto" do tipo `stock` (lembre-se que `stock` foi definido em `stock.php`). Se você pensar em `$s` como um ponteiro (uma "referência" em PHP), você pode acessar (ou, melhor ainda imprimir) campos individuais do objeto com código como o abaixo.

```
print($s->price);
```

Aliás, lembre-se que o seu código PHP não precisa aparecer só no topo dos arquivos `php`. Na verdade, você pode intercalar PHP e HTML, como no exemplo abaixo, supondo que `$s` já tenha recebido o valor de retorno de `lookup` em outro ponto desse arquivo (por exemplo no topo).

```
<div style="text-align: center">
  A ação <? print($s->name); ?> atualmente custa $<? print($s->price); ?>.
</div>
```

Na verdade, se tudo o que você quer fazer é imprimir algum(ns) valor(es), você pode usar essa sintaxe mais concisa em vez disso:

```
<div style="text-align: center">
  A ação <?= $s->name ?> atualmente custa $<?= $s->price ?>.
</div>
```

Claro, se o usuário enviou um símbolo inválido (quando `lookup` retorna `NULL`), certifique-se de dar as devidas desculpas, explicando o problema! Certifique-se, também, que qualquer HTML gerado por `quote.php` e `quote2.php` é válido, verificando-o em <http://validator.w3.org/>.

- ☐ Agora é a hora de se concentrar um pouco em design. Atualmente, o seu banco de dados só guarda os usuários, mas ele não tem uma forma de guardar as carteiras deles.<sup>1</sup> Não faz muito sentido adicionar mais campos à tabela `users` para lembrar das ações que cada um tem (usando, digamos, um campo diferente por companhia). É melhor manter esses dados em uma nova tabela, com o objetivo de não impor restrições às carteiras ou gastamos espaço com campos que talvez nem serão utilizados.

Que tipo de informações nós precisamos guardar nessa tabela exatamente para “lembrar” das carteiras dos usuários? Bem, nós provavelmente queremos um campo para identificar o usuário por seu ID (`uid`) contido em `users`. Nós provavelmente queremos guardar as ações de cada usuário através dos seus símbolos, que são menores que os nomes (e mais eficientemente guardados na memória).<sup>2</sup> E nós ainda precisamos saber quantas ações cada usuário tem de uma certa companhia. Em outras palavras, uma tabela com três campos (`uid`, `symbol` e `shares`) pode ser uma boa ideia, mas você é bem vindo a prosseguir com suas próprias ideias de design. De qualquer forma, volta para o `phpMyAdmin` e crie essa nova tabela, colocando o nome que você achar mais interessante. Para criar uma nova tabela vá para a página `structure` do banco de dados `finance` (não de alguma tabela) e procure o campo **Create Table**. Especifique o nome e a quantidade de campos e depois clique em **Go**. Na nova janela que aparece defina (em qualquer ordem) cada um dos seus campos.

Se você decidiu utilizar aqueles três campos (`uid`, `symbol` e `shares`), perceba que `uid` não deve ser definido como a chave primária dessa tabela, se fosse assim cada usuário só poderia ter ações de uma única companhia (já que o seu `uid` não poderia aparecer em mais de uma linha). Perceba também que você não pode deixar um `uid` e um `symbol` aparecerem juntos (na mesma linha)

<sup>1</sup> Com “carteira” nós queremos dizer um conjunto de ações que algum usuário possui.

<sup>2</sup> Com certeza você poderia lembrar de um ID numérico, ao invés de um símbolo, para cada companhia. Mas nesse caso você precisaria manter o seu próprio banco de dados de companhias, construído com o tempo a partir de dados coletados, por exemplo, do Yahoo. Então é provavelmente melhor (e com certeza mais simples) lembrar das ações através dos seus respectivos símbolos.



mais de uma vez. É melhor atualizar a quantidade de ações (`shares`) que o usuário tem de uma companhia ao invés de criar dois registros diferentes para o mesmo usuário e a mesma companhia. Um jeito interessante de impor essa restrição na hora de criar a tabela é definir uma “joint primary key” colocando um índice PRIMARY tanto para `uid` quanto para `symbol`. Dessa forma, `mysql_query` vai retornar `FALSE` se você tentar inserir mais do que uma linha com o mesmo par `uid` e `symbol`. Nós deixamos, porém, a decisão dos tipos dos seus campos para você.<sup>1</sup> Quando terminar de definir a sua tabela, clique em **Save!**

- ☐ Antes de deixar que os usuários comprem e vendam ações, vamos dar algumas ao Skroob e aos seus amigos de graça. Dirija-se, no phpMyAdmin, a sua tabela `users` e tome nota dos IDs dos seus usuários. Depois vá para a sua nova tabela (das carteiras dos usuários) e siga para a aba **Insert**. Através dessa interface, “compre” algumas ações para os seus usuários inserindo manualmente algumas novas linhas nessa tabela (Você pode querer voltar à página do Yahoo! Finance para procurar alguns símbolos). Não há a necessidade de retirar o preço das ações do `cash` deles na tabela `users`; seja bonzinho e dê essas ações de graça mesmo.

Agora que seus usuários já tem algumas ações, vamos ver o que você fez. Clique na aba **SQL** e execute a seguinte linha, onde `tbl` representa o nome da sua nova tabela.

```
SELECT * FROM tbl WHERE uid = 4
```

Assumindo que 4 é o ID do Skroob, isso deve retornar todas as linhas de `tbl` que representam ações do presidente. Se os únicos campos na tabela são, digamos, `uid`, `symbol` e `shares`, saiba que a linha acima é a mesma coisa que a seguinte.

```
SELECT uid, symbol, shares FROM tbl WHERE uid = 4
```

Se, por outro lado, você quisesse checar somente a quantidade de ações do Skroob da companhia A5, você poderia tentar algo como

```
SELECT shares FROM tbl WHERE uid = 4 AND symbol = 'AFLB.OB'
```

Se você comprou alguma ação dessa companhia para o Skroob, essa consulta deve retornar uma linha com uma coluna, representando o número de ações. Se você não deu a ele nenhuma ação dessa companhia, a consulta retornará um conjunto vazio.

Acontece que, através dessa aba **SQL**, você também poderia ter inserido aquelas “compras” com o operador `INSERT`. Mas a GUI do phpMyAdmin te salvou desse trabalho.

Ok, vamos começar a usar todo esse conhecimento. Está na hora de deixar os usuários utilizarem as suas carteiras! Mexa em `index.php`, de forma que ele reporte as informações sobre cada uma das ações que o usuário tem, incluindo o número de ações e o valor atual, além do dinheiro atual do usuário. Você é bem vindo, mas não obrigado, a usar os outros dados guardados pela classe `stock`. Nem é necessário dizer que `index.php` vai precisar chamar `lookup` assim como

---

<sup>1</sup> Se você incluir `uid` nessa tabela, saiba que o seu tipo deve ser igual ao tipo de `uid` em `users`. Mas não especifique `auto_increment` para esse campo, pois você só quer essa opção de incrementação quando os IDs dos usuários forem criados (por `register2.php`). E não chame a sua tabela de `tabela`.

`quote2.php` o fez, mas talvez mais de uma vez. Saiba que o script PHP pode certamente chamar `mysql_query` várias vezes, mesmo que, até agora, nós só tenhamos visto essa função sendo usada uma vez por arquivo no máximo. De forma similar, você pode chamar `mysql_fetch_array` várias vezes, particularmente dentro de loops.

Por exemplo, se o seu objetivo é simplesmente mostrar, digamos, as ações do Presidente Skroob, uma por linha, em uma tabela HTML, você pode gerar as linhas com um código como o seguinte.

```
<?
$result = mysql_query("SELECT symbol, shares FROM tbl WHERE uid = 4");
while ($row = mysql_fetch_array($result))
{
    $s = lookup($row["symbol"]);
    print('<tr>');
    print('<td>' . $s->name . '</td>');
    print('<td>' . $row["shares"] . '</td>');
    print('</tr>');
}
?>
```

Alternativamente, você pode fazer isso sem o operador de concatenação (.) através de uma sintaxe como a seguinte:

```
<?
$result = mysql_query("SELECT symbol, shares FROM tbl WHERE uid = 4");
while ($row = mysql_fetch_array($result))
{
    $s = lookup($row["symbol"]);
    print("<tr>");
    print("<td>{$s->name}</td>");
    print("<td>{$row[\"shares\"]}</td>");
    print("</tr>");
}
?>
```

Note que, na segunda versão, nós utilizamos aspas duplas nas linhas de HTML, ao invés de aspas simples de forma que as variáveis lá dentro (`$n->name` e `$row["shares"]`) sejam interpoladas (substituídas por seus valores) pelo interpretador do PHP; variáveis entre aspas simples não são interpoladas. Nós também colocamos chaves em volta dessas variáveis para que o PHP perceba que elas são variáveis; variáveis com sintaxes mais simples (por exemplo `$foo`) não precisam de chaves para a interpolação.<sup>1</sup>

---

<sup>1</sup> Está tudo bem usar aspas duplas dentro daquelas chaves, mesmo que nós já tenhamos usado aspas duplas em volta do argumento inteiro de `print`.

De qualquer forma, apesar de ser comum, gerar HTML através da função `print` não a forma mais elegante possível. Um outro jeito, ainda talvez um pouco deselegante, se parece com o seguinte:

```
<? $result = mysql_query("SELECT symbol, shares FROM tbl WHERE uid = 4"); ?>

<? while ($row = mysql_fetch_array($result)): ?>

    <? $s = lookup($row["symbol"]); ?>

    <tr>
        <td><?= $s->name ?></td>
        <td><?= $row["shares"] ?></td>
    </tr>

<? endwhile ?>
```

Para saber como deve ser o HTML final gerado, dê uma olhada em

<http://cc50.com.br/finance/>

a procura de inspiração ou dicas. Mas não se sinta obrigado a copiar o nosso design. Faça esse website da forma que você quiser! Qualquer HTML e PHP que você escrever deve estar nos padrões comuns de estilo (indentado), mas não tem problema se alguma das suas linhas tiver mais do que 80 caracteres de tamanho. Qualquer HTML que você gerar dinamicamente (através da função `print` do PHP) não precisa seguir nenhum padrão de estilo.

Aliás, apesar de nós sempre usarmos o Presidente Skroob nos nossos exemplos, o seu código deve funcionar para qualquer usuário que estiver logado.

- Agora é a hora de implementar a venda de ações em, digamos `sell.php` e `sell2.php`. Nós deixamos o design dessas páginas para você. Mas saiba que, em `sell2.php`, você pode deletar uma linha da sua tabela (em nome do Skroob, por exemplo) com a seguinte string SQL:

```
DELETE FROM tbl WHERE uid = 4 AND symbol = 'AFLB.OB'
```

Nos deixamos para você projetar o que exatamente uma string como essa deve fazer. Você pode, com certeza, testar as suas strings SQL na aba **SQL** do phpMyAdmin. Agora que o usuário não tem mais as ações, devemos dar a ele o dinheiro referente a elas. Então vender uma ação não envolve somente atualizar a tabela que contém a carteira do seu usuário, mas a tabela dos usuários também. Nós deixamos para você determinar como computar quanto dinheiro o usuário deve receber. Mas assim que você souber a quantia (digamos \$500), SQL como o seguinte deve fazer o depósito (para o Skroob, por exemplo):<sup>1</sup>

```
UPDATE users SET cash = cash + 500 WHERE uid = 4
```

---

<sup>1</sup> Com certeza, se o banco de dados ou o servidor web cair entre esse `DELETE` e esse `UPDATE`, o Skroob pode perder todo esse dinheiro. Você não precisa se preocupar com esses casos! Também é possível, com o uso de multi processamento e condições de corrida, que um Skroob bem esperto engane o seu site para pagá-lo mais de uma vez. Você também não precisa se preocupar com esses casos! Bem, mas se você está muito disposto, pode utilizar MyISAM ou InnoDB com transações SQL para resolver esses problemas. Veja <http://dev.mysql.com/doc/refman/5.1/en/innodb.html> para referência.

Está tudo bem, por simplicidade, deixar somente que os usuários vendam todas as suas ações de uma companhia de uma vez, ao invés de somente algumas. Nem é necessário dizer isso mas teste o seu código logando no site como algum usuário e vendendo algumas ações. Você sempre pode “comprá-las” de volta manualmente com o phpMyAdmin.

Como sempre, tenha certeza que o seu HTML é válido!

- ☐ Agora falta deixar os usuários de fato comprarem alguma coisa. Implemente isso em, digamos, `buy.php` e `buy2.php`.<sup>1</sup> Você é inteiramente responsável pelo design dessa responsável, mas, como antes, sinta-se livre para dar uma olhada em

`http://cc50.com.br/finance/`

em busca de inspiração ou dicas. Com certeza você vai precisar garantir que um usuário não pode gastar mais dinheiro do que ele ou ela possui no momento. E você vai querer ter certeza que os usuários só podem comprar quantidades inteiras (e nunca frações) de ações. Para isso saiba que

```
preg_match("/^\d+$/", $_POST["shares"])
```

retorna `TRUE` se e somente se `$_POST["shares"]` contem um inteiro não negativo, devido ao uso de uma expressão regular. Dê uma olhada em [http://www.php.net/preg\\_match](http://www.php.net/preg_match) para checar alguns detalhes. Sempre peça desculpas ao usuário se você precisar rejeitar o input dele ou dela por algum motivo. Em outras palavras, tenha certeza que você está validando todo o input do seu usuário rigorosamente (nós deixamos para você determinar o que precisa ser validado!).

Aliás, se você implementou a sua tabela das carteiras como nós instruímos (com aquela `joint primary key`), saiba que um SQL como o seguinte (o qual, infelizmente, não coube em uma única linha) vai inserir uma nova linha na tabela a não ser que o par especificado de `uid` e `symbol` já exista em alguma outra linha. Nesse caso o número de ações (`shares`) naquela linha vai simplesmente aumentar (por exemplo em 10).

```
INSERT INTO table (uid, symbol, shares) VALUES(4, 'AFLB.OB', 10)
ON DUPLICATE KEY UPDATE shares = shares + VALUES(shares)
```

Como sempre tenha certeza que o HTML final é válido!

---

<sup>1</sup> Como sempre, você não precisa se preocupar com interrupções do serviço ou condições de corrida.

- ☐ E agora o seu Gran Finale. Os seus usuários tem o poder de comprar e vender ações, e até de checar o valor das suas carteiras. Mas eles não tem um modo de ver o seu histórico de transações.

Mexa nas suas implementações da compra e venda de ações de uma forma que você comece a guardar as transações feitas. As seguintes informações são importantes

- ☐ Se a operação foi de venda ou de compra.
- ☐ O símbolo da companhia.
- ☐ O número de ações vendidas ou compradas.
- ☐ O preço de uma ação no momento da operação.
- ☐ A data e a hora da operação.

Agora, através de um arquivo chamado `history.php`, deixe que os usuários vejam os seus próprios históricos de transações, formatados como você achar melhor. Coloque um link para essa página em algum lugar em `index.php`. Tenha certeza que o HTML gerado é válido!

- ☐ Ufa. Olhe agora para `index.php` e, se isso ainda não está lá, coloque links para, pelo menos, `buy.php`, `history.php`, `logout.php`, `quote.php`, e `sell.php` (ou os seus equivalentes) de forma que cada um está a apenas um clique de distância da carteira de ações do usuário!

- ☐ E agora a cereja do bolo. Só mais um recurso para terminar, mas agora você pode escolher. Implemente pelo menos um (1) dos recursos abaixo. Você pode interpretar cada um deles da forma que quiser; nós deixamos todas as decisões de design para você. Só seja claro em relação a qual recurso você escolheu (por exemplo colocando um link nomeado apropriadamente em `index.php`). E tenha certeza que o HTML da sua página final é válido.

- ☐ Deixe que os usuários troquem suas senhas.
- ☐ Deixe que os usuários que esqueceram as suas senhas recebam lembretes via email.
- ☐ Mande um email com informações sobre a transação do usuário, toda vez que ele ou ela comprar ou vender ações.
- ☐ Deixe que os usuários depositem fundos adicionais.

- ☐ Quando você achar que terminou, chame os seus amigos ou tente você mesmo quebrar o seu site, como um bom adversário faria. Sob nenhuma circunstância você (ou nós) deve conseguir fazer o seu código quebrar (fazendo aparecer, por exemplo, algum aviso ou erro do interpretador do PHP). Você deve sempre identificar e se desculpar por qualquer erro que o input de um usuário, malicioso ou não, pode induzir. E, como uma boa medida de segurança, você deve “limpar” qualquer input do usuário em todas as páginas do seu site, por exemplo com `mysql_real_escape_string`!

- ☐ Não se esqueça que o seu trabalho deve funcionar da mesma forma em pelo menos dois grandes browsers!

### Checando...

Antes de considerar esse Pset terminado, melhor perguntar a si mesmo algumas coisas e depois voltar e melhorar o seu código conforme o necessário! Não considere essa lista como uma lista de todas as nossas expectativas, mas apenas como alguns lembretes úteis. As caixas que vieram antes representam a lista completa! Para ser claro, considere as perguntas abaixo retóricas. Não há a necessidade de respondê-las por escrito para nós, pois todas as suas respostas devem ser "sim!"

- ☐ O HTML da sua homepage é válido de acordo com `validator.w3.org`?
- ☐ O HTML gerado por todos os seus arquivos PHP é válido de acordo com `validator.w3.org`?
- ☐ As suas páginas detectam e cuidam de inputs inválidos de forma correta?
- ☐ Você está guardando os históricos das transações dos usuários de forma correta?
- ☐ Você implementou pelo menos um (1) recurso adicional?
- ☐ Todas as suas páginas (exceto as de login, logout e registro) precisam de autenticação?
- ☐ Você escolheu tipos de dados apropriados para os campos das tabelas do seu banco de dados?

Como sempre, se você não respondeu “sim” a um ou mais dessas perguntas porque você está tendo alguma dificuldade, mande um email para `ajuda@cc50.com.br`!

- ☐ Esse foi o Pset 7!