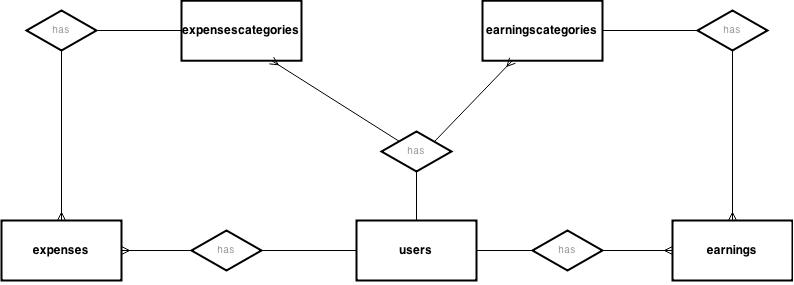
**Documentación de Accounting Book**

**Información general**

* **Lenguaje de programación**: PHP 5.0
* **Framework**: Laravel 5.0
* **Repositorio**: <https://github.com/sanmen1593/AccountingBook>

**Información técnica**

**Diagrama de base de datos**



**Tabla users**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributo** | **Tipo** | **Nulleable** | **Unique** |
| Id (PK) | int |  | x |
| name | string |  |  |
| lastname | string |  |  |
| username | string |  |  |
| email | string |  |  |
| currency | string |  |  |

**Tabla earnings**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributo** | **Tipo** | **Nulleable** | **Unique** |
| Id (PK) | int |  | x |
| amount | string |  |  |
| date | datestring |  |  |
| timestamps | datetime |  |  |
| earningsCategory\_id (FK) | int |  |  |
| user\_id (FK) | int |  |  |

**Tabla expenses**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Atributo** | **Tipo** | **Nulleable** | **Unique** | **Description** |
| Id (PK) | int |  | x |  |
| amount | double |  |  |  |
| description | string |  |  |  |
| date | datestring |  |  | AAAAMMDD |
| timestamps | datetime |  |  |  |
| user\_id (FK) | int |  |  |  |
| expensesCategory\_id (FK) | int |  |  |  |

**Tabla earningscategories**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributo** | **Tipo** | **Nulleable** | **Unique** |
| Id (PK) | int |  | x |
| slug | string |  |  |
| superior\_cat | int | x |  |
| user\_id (FK) | int |  |  |

**Tabla expensescategories**

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributo** | **Tipo** | **Nulleable** | **Unique** |
| Id (PK) | int |  | x |
| slug | string |  |  |
| superior\_cat | int | x |  |
| user\_id (FK) | int |  |  |

**Sesiones**

La autenticación a la aplicación se hace mediante la funcionalidad que ofrece Laravel. Para este caso, el usuario se autentica con su usuario (**username**) y contraseña (**password**). Todas las páginas están “prohibidas” al menos que el usuario ya se haya autenticado. En caso tal de que no, simplemente será redirigido a la página de login.

Para manejar las sesiones (login, logout) utilizamos **SessionsController**, que posee 3 funciones:

* **Create**: Esta función se encarga simplemente de retornar la vista **login.blade.php,** que es la vista donde se encuentra el formulario para hacer login. En caso de que el usuario ya esté autenticado, le hace una redirección al dashboard del usuario.
* **Store**: Esta función reciba los parámetros para iniciar sesión (username y password). Después de verificar que ambos datos vengan, realiza un intento de autenticación. Si es correcta, lo reenvía al dashboard de usuario autenticado. Sino, lo reenvía a la página de login con un mensaje de “Login failed!”.
* **Destroy**: Esta función no recibe ningún parámetro. Simplemente cierra la sesión del usuario que se encuentre autenticado, y lo reenvía a la página de login.

**Usuarios**

Se pueden registrar usuarios, actualizar los ya existentes, eliminarse (soft-deleting, no se borran sus datos, pero si el acceso) y por supuesto, consultar su información. Para esto, está la clase UserController, con las siguientes funciones:

* **Index**: Esta función simplemente retorna la vista del perfil.
* **Create**: Retorna la vista donde se encuentra el formulario de registro.
* **Store**: Recibe los datos del formulario de registro, y se validan los datos utilizando Validator de Laravel. Si falla, se retorna al formulario de registro y se muestran los distintos errores que presenta el formulario. Si es exitoso, se crea el usuario en la base de datos y se redirige a la página de login con un mensaje de registro exitoso.
* **Show**:
* **Update**:
* **Edit**:
* **Destroy**:

**Categorías**

Estas categorías están para que el usuario pueda tener un control de sus ingresos y gastos, de manera que sepa en qué categoría están los mayores gastos o los mayores ingresos. Las **categorías son personalizadas** por cada usuario, de manera que puede crearlas, editarlas y eliminarlas.

Las categorías pueden ser **categorías superiores** o **subcategorías** de una categoría superior. Esta funcionalidad solo llega hasta un nivel, es decir, una subcategoría no puede tener otras subcategorías.

Además, cada categoría tiene un tipo: Si es una categoría para los ingresos o si es una categoría para los egresos. Esta diferenciación está más que todo para los freelancers, para quienes sus ingresos vienen de distintas fuentes.

Para este “modulo”, utilizamos el controlador **CategoriesController**, que contiene las siguientes funciones:

* **Index**: Primero busca todas las categorías correspondientes al usuario, y luego las divide en dos arreglos: Uno de categorías superiores, y uno de un arreglo de categorías inferiores. El de categorías inferiores, el índice de cada arreglo de categorías inferiores corresponde al id de la categoría superior.  
  Luego se mandan estos dos arreglos a la vista de **indexcategories**, donde se muestra cada categoría superior con sus correspondientes categorías inferiores.
* **Create**:
* **Store**:
* **Show**:
* **Update**:
* **Edit**:
* **Destroy**: