DOKUMENTASI ATURDUIT.ID
Dr. Umi Widyastuti, SE, M.E dan Team

Nucleus Web Development

JASA WEB DESIGN
Jl Kalibaru Raya, Cilodong
Kota Depok, Indonesia
Phone : +62 85885881998
Email : cs@membuat.web.id
TENTANG WEB

ATURDUIT.ID MERUPAKAN WEB APPLICATION YANG BERFUNGSI UNTUK PERENCANAAN DAN PENGELOLAAN KEUANGAN SECARA REALTIME.

FEATURE
- PENCATATAN PEMASUKAN
- PENCATATAN PENGELUARAN
- PEMASUKAN VS PENGELUARAN
- REPORT TO PDF & EXCEL
- GRAFIK DATA AKTIFITAS
- MANAJEMEN ROLE USER

TECHNOLOGY WEB
- LARAVEL FRAMEWORK
- CLOUD VPS SERVER
- SSL DENGAN LET’S ENCRYPT
- MYSQL DATABASE
- CYBERPANEL

INFORMASI DETAIL WEB

NAMA DOMAIN : ATURDUIT.ID
SERVER IP ADDRESS : 103.172.205.187

TEKNOLOGI SERVER
SECURITY SERVER : OPEN LITESPEED

CSF FIREWALL
```python
# Code

# Funcions

# Main
```

Gambar 2
<?php

namespace App\Http\Controllers\Auth;
use Illuminate\Http\Request;
use App\Http\Controllers\Controller;
use Illuminate\Support\Facades\Session;
use Illuminate\Foundation\Auth\AuthenticatesUsers;
use Illuminate\Support\Facades\Auth;
use DB;

class LoginController extends Controller
{
    use AuthenticatesUsers;
    protected $redirectTo = '/home';
    public function __construct()
    {
        $this->middleware('guest')->except('logout');
    }
    public function logout()
    {
        Auth::logout();
        return redirect('/login');
    }
    public function authenticate(Request $request)
    {
        $email = $request->input('email');
        $password = $request->input('password');
        $this->validate($request, [
            'email' => 'nullable',
            'phone' => 'nullable',
            'password' => 'required'
        ]);  
        if ($email != null) {
            if (Auth::attempt(['email' => $email, 'password' => $password, 'status' => 'Active'])) {
                $user = DB::table('users')->where('email', $email)->get();
                //data = DB::table('roleaccess')->where('userid',$user[0]->userid)->get();
            }  
        }
    }
}
} else if ($phone != null) {
    if (Auth::attempt(['phone' => $phone, 'password' => $password, 'status' => 'Active'])) {
        $user = DB::table('users')->where('phone', $phone)->get();
    }
}

return redirect()->back()->withErrors(["this->username() => Lang::get('auth.failed'),
]);

/**
 * get application settings
 *
 * @return object
 */
public function getApplication()
{
    $data = DB::table('settings')->where('settingsid', '1')->get();
    if ($data) {
        $res['success'] = true;
        $res['data'] = $data;
        $res['message'] = 'list data';
        return response($res);
    }
}

public function showLoginForm()
{
    return view('auth.login');
}
<?php

namespace App\Http\Controllers\Auth;

use App\User;
use Illuminate\Http\Request;
use App\Http\Controllers\Controller;
use Illuminate\Support\Facades\Validator;
use Illuminate\Foundation\Auth\RegistersUsers;
use DB;
use Illuminate\Support\Facades\Auth;

class RegisterController extends Controller
{
    use RegistersUsers;
    protected $redirectTo = '/home';
    public function __construct()
    {
        $this->middleware('guest');
    }
    protected function validator(array $data)
    {
        return Validator::make($data, [
            'name' => 'required|string|max:255',
            'email' => 'string|email|max:255|unique:users',
            'phone' => 'required|max:15|min:9',
            // 'role' => 'required|string',
            'password' => 'required|string|min:6|confirmed',
        ]);}
    protected function create(array $data)
    {
        return User::create(
            'name' => $data['name'],
            'email' => $data['email'],
            'phone' => $data['phone'],
            // 'role' => $data['role'],
            'password' => bcrypt($data['password']),
        );
    }
    public function showRegistrationForm()
    {
        return view('auth.register');
    }
}
public function daftar(Request $request)
{
    $name  = $request->input('name');
    $email = $request->input('email');
    $phone = $request->input('phone');
    $password = bcrypt($request->input('password'));
    $time  = date("Y-m-d h:i:s");
    $access = $request->input('dataroles');

    $phonecheck = DB::table('users')
        ->where('phone', '=', $phone)
        ->first();
    if ($phonecheck) {
        return redirect()->back()->with('gagal', 'Nomor Handphone sudah terdaftar di Database kami!');
    } else {
        $data = array('name' => $name, 'email' => $email, 'password' => $password, 'phone' => $phone, 'role' => 'Staff', 'status' => 'Active', 'updated_at' => $time, 'created_at' => $time);
        $insert = DB::table('users')->insert($data);
        if ($insert) {
            $lastid = DB::pdo()->lastInsertId();
            for ($i = 2; $i < 15; $i++) {
                $insertrole = DB::table('roleaccess')->insert(
                    [
                        'userid' => $lastid,
                        'roleid' => $i
                    ]
                );
            }
            return redirect()->route('login');
        }
    }
}
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Yajra\Datatables\Datatables;
use App\AccountModel;
use App\SettingModel;
use App\Http\Controllers\TraitSettings;
use DB;
use App;
use Auth;

class AccountController extends Controller
{
    use TraitSettings;

    public function __construct()
    {
        $data = $this->getApplications();
        $lang = $data[0]->languages;
        App::setLocale($lang);
        $this->middleware('auth');
    }

    // show default data
    public function index()
    {
        if (Auth::user()->isRole("5") )
            return view( 'account.index' );
        else{
            return redirect('home');
        }
    }

    public function detail($id)
    {
        if (Auth::user()->isRole("5") )
            return view( 'account.detail',compact('id') );
        else{
            return redirect('home');
        }
    }

    public function getdata()
    {
        $account = AccountModel::select(["accountid", "name", "balance", "description" ]);

        return Datatables::of($account)
        ->addColumn('balance', function ($single ) {
            $setting = DB::table('settings')-
            where('settingsid', '1')->get();
            return $setting[0]->currency.number_format($single->balance, 2);
        })
        ->addColumn('action', function ($accountsingle ) {
        })
    }
return '<a href="account/detail/\$.accountsingle->accountid." id="btneidit" customdata=\$.accountsingle->accountid.' class="btn btn-sm btn-success">" class="ti-check-box"></i>'.
</a>
trans('lang.detail').'</a>

<a href="#" id="btneidit" customdata=\$.accountsingle->accountid.' class="btn btn-sm btn-primary" data-toggle="modal" data-target="#edit">" class="ti-pencil"></i>'.
</a>
trans('lang.edit').'</a>

<a href="#" id="btndelete" customdata=\$.accountsingle->accountid.' class="btn btn-sm btn-danger" data-toggle="modal" data-target="#delete">" class="ti-trash"></i>
</a>
trans('lang.delete').'</a>

</a>
</a>

}); make( true);

/**
    * Get account data detail
    *
    * @return object
    */
public function getdatadetail($id){

    $data = DB::select("SELECT p.accountnumber, p.name, COALESCE(a.amount, 0) as income, COALESCE(b.amount, 0) as expense, COALESCE(p.balance+(COALESCE(a.amount, 0)-
    COALESCE(b.amount, 0)), 0) as balance from account as p left join (select accountid,sum(amount) as amount
    from transaction where type=1 group by accountid) as a on a.accountid = p.accountid
left join (select
    accountid,sum(amount) as amount from transaction where type=2 group by accountid) as b on b.accountid
    = p.accountid
    where p.accountid = $id group by p.accountid");

    $res['accountnumber'] = $data[0]->accountnumber;
    $res['name'] = $data[0]->name;
    $res['balance'] = number_format($data[0]->balance);
    return response($res);

}

/**
    * Get account balance by month
    *
    * @return object
    */
public function accountbalancebymonth($id){

    $year = date('Y');
    $sjian = DB::select("SELECT p.accountnumber, p.name, COALESCE(a.amount, 0) as income,
    COALESCE(b.amount, 0) as expense,
    COALESCE((COALESCE(a.amount, 0)-
    COALESCE(b.amount, 0)), 0) as balance
    from account as p
    left join (select accountid,
    sum(amount) as amount from transaction
}
where type=1 and
month(transactiondate)=01 and year(transactiondate)=\$year group by accountid) as a on a.accountid = p.accountid
left join (select accountid,sum(amount) as amount
from transaction where type=2 and
month(transactiondate)=01 and year(transactiondate)=\$year group by accountid) as b on b.accountid = p.accountid
where p.accountid = \$id group by
p.accountid";)

$feb = DB::select("SELECT p.accountnumber,
p.name,COALESCE(a.amount,0) as income,
COALESCE(b.amount,0) as expense,
COALESCE((COALESCE(a.amount,0)-
COALESCE(b.amount,0)),0) as balance
from account as p
left join (select accountid,
sum(amount) as amount
from transaction
where type=1 and
month(transactiondate)=02 and year(transactiondate)=\$year group by accountid) as a on a.accountid = p.accountid
left join (select accountid,sum(amount) as amount
from transaction where type=2 and
month(transactiondate)=02 and year(transactiondate)=\$year group by accountid) as b on b.accountid = p.accountid
where p.accountid = \$id group by
p.accountid";)

$imar = DB::select("SELECT p.accountnumber,
p.name,COALESCE(a.amount,0) as income,
COALESCE(b.amount,0) as expense,
COALESCE((COALESCE(a.amount,0)-
COALESCE(b.amount,0)),0) as balance
from account as p
left join (select accountid,
sum(amount) as amount
from transaction
where type=1 and
month(transactiondate)=03 and year(transactiondate)=\$year group by accountid) as a on a.accountid = p.accountid
left join (select accountid,sum(amount) as amount
from transaction where type=2 and
month(transactiondate)=03 and year(transactiondate)=\$year group by accountid) as b on b.accountid = p.accountid
where p.accountid = \$id group by
p.accountid";)

$sapr = DB::select("SELECT p.accountnumber,
p.name,COALESCE(a.amount,0) as income,
COALESCE(b.amount,0) as expense,
<table>
<thead>
<tr>
<th>Statement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>COALESCE(b.amount, 0), 0) as balance</code></td>
<td>From account as p left join (select accountid, sum(amount) as amount from transaction where type=1 and</td>
</tr>
<tr>
<td><code>month(transactiondate)=04 and year(transactiondate)=\$year group by accountid) as a on a.accountid = p.accountid</code></td>
<td></td>
</tr>
<tr>
<td><code>amount</code></td>
<td>left join (select accountid, sum(amount) as amount)</td>
</tr>
<tr>
<td><code>month(transactiondate)=04 and year(transactiondate)=\$year group by accountid) as b on b.accountid = p.accountid</code></td>
<td>from transaction where type=2 and where p.accountid = $id group by p.accountid&quot;;</td>
</tr>
</tbody>
</table>
| $\text{simay}$ = DB::select("SELECT p.accountnumber, p.name, COALESCE(a.amount, 0) as income, COALESCE(b.amount, 0) as expense, COALESCE((COALESCE(a.amount, 0)-
| `COALESCE(b.amount, 0)), 0) as balance`                                                                          | From account as p left join (select accountid, sum(amount) as amount from transaction where type=1 and |
| `month(transactiondate)=05 and year(transactiondate)=\$year group by accountid) as a on a.accountid = p.accountid` |                                                                                                   |
| `amount`                                                                                                          | left join (select accountid, sum(amount) as amount)                                               |
| `month(transactiondate)=05 and year(transactiondate)=\$year group by accountid) as b on b.accountid = p.accountid` | from transaction where type=2 and where p.accountid = \$id group by p.accountid";               |
| $\text{sijun}$ = DB::select("SELECT p.accountnumber, p.name, COALESCE(a.amount, 0) as income, COALESCE(b.amount, 0) as expense, COALESCE((COALESCE(a.amount, 0)-
| `COALESCE(b.amount, 0)), 0) as balance`                                                                          | From account as p left join (select accountid, sum(amount) as amount from transaction where type=1 and |
| `month(transactiondate)=06 and year(transactiondate)=\$year group by accountid) as a on a.accountid = p.accountid` |                                                                                                   |
| `amount`                                                                                                          | left join (select accountid, sum(amount) as amount)                                               |
| `month(transactiondate)=06 and year(transactiondate)=\$year group by accountid) as b on b.accountid = p.accountid` | from transaction where type=2 and where p.accountid = \$id group by p.accountid";               |
| p.accountid"
| $jul = DB::select("SELECT p.accountnumber, p.name, COALESCE(a.amount,0) as income, COALESCE(b.amount,0) as expense, COALESCE((COALESCE(a.amount,0)-COALESCE(b.amount,0)),0) as balance
| from account as p
| left join (select accountid, sum(amount) as amount
| from transaction
| where type=1 and
| month(transactiondate)=07 and year(transactiondate)=$year group by accountid) as a on a.accountid = p.accountid
| amount
| left join (select accountid, sum(amount) as amount
| from transaction where type=2 and
| month(transactiondate)=07 and year(transactiondate)=$year group by accountid) as b on b.accountid = p.accountid
| where p.accountid = $id group by
| p.accountid"
| $iags = DB::select("SELECT p.accountnumber, p.name, COALESCE(a.amount,0) as income, COALESCE(b.amount,0) as expense, COALESCE((COALESCE(a.amount,0)-COALESCE(b.amount,0)),0) as balance
| from account as p
| left join (select accountid, sum(amount) as amount
| from transaction
| where type=1 and
| month(transactiondate)=08 and year(transactiondate)=$year group by accountid) as a on a.accountid = p.accountid
| amount
| left join (select accountid, sum(amount) as amount
| from transaction where type=2 and
| month(transactiondate)=08 and year(transactiondate)=$year group by accountid) as b on b.accountid = p.accountid
| where p.accountid = $id group by
| p.accountid"
| $isep = DB::select("SELECT p.accountnumber, p.name, COALESCE(a.amount,0) as income, COALESCE(b.amount,0) as expense, COALESCE((COALESCE(a.amount,0)-COALESCE(b.amount,0)),0) as balance
| from account as p
| left join (select accountid, sum(amount) as amount
| from transaction
| where type=1 and
| month(transactiondate)=09 and year(transactiondate)=$year group by accountid) as a on a.accountid = p.accountid
|
| Amount | Left join (select accountid, sum(amount) as left join (select accountid, sum(amount) as
| Month(transactiondate)=09 and year(transactiondate)=year group by accountid) as a on a.accountid = p.accountid |
|        | from transaction where type=2 and |
|        | where p.accountid = $id group by |
|        | $iokt = DB::select("SELECT p.accountnumber, |
|        | p.name, COALESCE(a.amount, 0) as income, |
|        | COALESCE(b.amount, 0) as expense, |
|        | COALESCE((COALESCE(a.amount, 0)- |
|        | COALESCE(b.amount, 0)), 0) as balance |
|        | from account as p |
|        | left join (select accountid, sum(amount) as amount |
|        | from transaction where type=1 and |
|        | left join (select accountid, sum(amount) as |
|        | from transaction where type=2 and |
|        | where p.accountid = $id group by |
|        | $inov = DB::select("SELECT p.accountnumber, |
|        | p.name, COALESCE(a.amount, 0) as income, |
|        | COALESCE(b.amount, 0) as expense, |
|        | COALESCE((COALESCE(a.amount, 0)- |
|        | COALESCE(b.amount, 0)), 0) as balance |
|        | from account as p |
|        | left join (select accountid, sum(amount) as amount |
|        | from transaction where type=1 and |
|        | left join (select accountid, sum(amount) as |
|        | from transaction where type=2 and |
|        | where p.accountid = $id group by |
|        | $ides = DB::select("SELECT p.accountnumber, |
|        | p.name, COALESCE(a.amount, 0) as income, |
|        | COALESCE(b.amount, 0) as expense, |
|        | COALESCE((COALESCE(a.amount, 0)- |
|        | COALESCE(b.amount, 0)), 0) as balance |
|        | from account as p |
|        | left join (select accountid,
`sum(amount) as amount` from transaction
where type=1

`month(transactiondate)=12 and year(transactiondate)=$year group by accountid` as a on a.accountid = p.accountid
left join (select accountid,sum(amount) as amount from transaction where type=2 and
month(transactiondate)=12 and year(transactiondate)=$year group by accountid) as b on b.accountid = p.accountid

```
$res['ijan'] = $ijan[0]->balance;
$res['ifeb'] = $ifeb[0]->balance;
$res['imar'] = $imar[0]->balance;
$res['iapr'] = $iapr[0]->balance;
$res['imay'] = $imay[0]->balance;
$res['ijun'] = $ijun[0]->balance;
$res['ijul'] = $ijul[0]->balance;
$res['iags'] = $iags[0]->balance;
$res['isep'] = $isep[0]->balance;
$res['iokt'] = $iokt[0]->balance;
$res['inov'] = $inov[0]->balance;
$res['ides'] = $ides[0]->balance;
```

return response($res);

```
public function save( Request $request ) {
    $name = $request->input( 'name' );
    $balance = $request->input( 'balance' );
    $description = $request->input( 'description' );
    $accountnumber = $request->input('accountnumber ');

    $data = array( 'name'=>$name, 'balance'=>$balance,
    'description'=>$description,'accountnumber'=>$accountnumber );
    $insert = DB::table('account')->insert( $data );

    if ( $insert ) {
        $res['success'] = true;
        $res['message'] = 'Account has been added';
        return response( $res );
    }
}
```

```
public function savedit( Request $request ) {
    $id = $request->input('id');
    $name = $request->input('name');
    $balance = $request->input('balance');
    $description = $request->input('description');
    $accountnumber = $request->input('accountnumber ');
```
```php
$update = DB::table('account')->where('accountid', $id)
    ->update(
        ['name' => $name,
        'balance' => $balance,
        'description' => $description,
        'accountnumber' => $accountnumber
    ];

if ($update) {
    $res['success'] = true;
    $res['message'] = 'Account has been updated';
    return response($res);
}

public function delete(Request $request) {
    $id = $request->input('iddelete');

    $delete = DB::table('account')->where('accountid', $id)->delete();
    $delete=transaction = DB::table('transaction')->where('accountid', $id)->delete();
    if ($delete) {
        $res['success'] = true;
        $res['message'] = 'Account has been deleted';
        return response($res);
    }

    public function getedit(Request $request) {
        $id = $request->input('id');

        $data = DB::table('account')->where('accountid', $id)->get();

        if ($data) {
            $res['success'] = true;
            $res['message'] = $data;
            return response($res);
        }

    public function getaccounttransaction($id) {
        $data = DB::table('transaction')->select(['transaction.*', 'category.name as category',
            'subcategory.name as subcategory', 'DB::raw("IFNULL(a.amount,"-" ) as income, IFNULL(b.amount,"-" ) as expense")'])
            ->leftJoin('DB::raw("(select transactionid,amount from transaction where type=1) as a"), function($join){
                $join->on("a.transactionid","=","transaction.transactionid");
```


```php
});

->leftJoin(DB::raw("(select transactionid, amount from transaction where type=2) as b"), function($join){
    $join->on("b.transactionid", ",=", "transaction.transactionid");
})
->leftJoin('subcategory', 'subcategory.subcategoryid', '=', 'transaction.categoryid')
->leftJoin('category', 'category.categoryid', '=', 'subcategory.categoryid')
->leftJoin('account', 'account.accountid', '=', 'transaction.accountid')
->where('transaction.accountid', "$id")
->orderBy('transactiondate', 'desc');

return Datatables::of($data)
->addColumn('income', function($single){
    $setting = DB::table('settings')->where('settingsid', '1')->get();
    $return = "";
    if($single->income == '-1'){
        $return = '1';
    }else {
        $return = '<p class="text-success">'. $setting[0]->currency_number_format($single->income, 2). '</p>';}

    return $return;
});

->addColumn('expense', function($single){
    $setting = DB::table('settings')->where('settingsid', '1')->get();
    $return = "";
    if($single->expense == '-1'){
        $return = '1';
    }else {
        $return = '<p class="text-danger">'. $setting[0]->currency_number_format($single->expense, 2). '</p>';}

    return $return;
});

->addColumn('transactiondate', function($single){
    $setting = DB::table('settings')->where('settingsid', '1')->get();
    return date(DB::table('settings')->where('settingsid', '1')->get())->format('Y-m-d H:i:s');
});

->rawColumns(['expense', 'income'])
->make(true);
```
YUK, KONSULTASI DENGAN KAMI

Nucleus Web Development

Jl Kalibaru Raya, Cilodong
Kota Depok, Indonesia

Phone: +62 85166039256
Email: cs@membuat.web.id
Web: www.membuat.web.id