**YII INSTALLATION GUIDE (yii stand for “YES IT IS”)**

**Yii installation**

* First step
  + Open cmd and select root folder of your project
  + Write php init in cmd , It will shows two options (production and development ) , select 0
  + Or if it give you error type:

c:\xampp\php\php.exe init

* + php init (it will generates files )

now configuration files are generated also with db files , change db creditionals

* open cmd and type yii migrate, it will ask confirmation type y to confirm
* yii installation is completed

**Add New Panel**

・ make a new copy of backend panel

・ replace all the namespaces from backend\ to the copied panel's name

・ add another alias in the **common/config/bootstrap.php** file is the root directory

Yii::setAlias('dispatcher', dirname(dirname(\_\_DIR\_\_)) . '/dispatcher');.

・ Config/main.php, assets/appAssets.php also replace other occurances for no confliction

・ Controllers/siteController change the namespace

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**//http://www.bsourcecode.com/yiiframework2/yii-2-user-login-from-database/**

**Separate session and login**

Add these lines after user in 3 files and also create a directory **tmp**

**frontend/config/main.php,**

'session' => [

'name' => 'PHPFrontSESSID',

'savePath' => \_\_DIR\_\_ . '/../tmp',

],

**backend/config/main.php,**

'session' => [

'name' => 'PHPBACKSESSID',

'savePath' => \_\_DIR\_\_ . '/../tmp',

],

**company/config/main.php**  
 'session' => [

'name' => 'PHPComapnySESSID',

'savePath' => \_\_DIR\_\_ . '/../tmp',

],

Also change the model 'identityClass'

'user' => [

'identityClass' => ‘common\models\User’,

'enableAutoLogin' => true,

],

**Change to this**

'identityClass' => 'dispatcher\models\DispatcherModel',

Then go to dispatcher\models\DispatcherModel.php

And open DispatcherModel.php

Then write these lines

**First include this line in the header**

1. **use yii\web\IdentityInterface**;

implement interface

change this line

**class DispatcherModel extends \yii\db\ActiveRecord**

to

**class DispatcherModel extends \yii\db\ActiveRecord implements IdentityInterface**

**and alsoe add these functions to the class**

/\*\*

\* @inheritdoc

\*/

public static function findIdentity($id)

{

return static::findOne(['id' => $id, 'status' => self::STATUS\_ACTIVE]);

}

/\*\*

\* @inheritdoc

\*/

public static function findIdentityByAccessToken($token, $type = null)

{

throw new NotSupportedException('"findIdentityByAccessToken" is not implemented.');

}

/\*\*

\* Finds user by username

\*

\* @param string $username

\* @return static|null

\*/

public static function findByUsername($username)

{

return static::findOne(['username' => $username, 'status' => self::STATUS\_ACTIVE]);

}

/\*\*

\* Finds user by password reset token

\*

\* @param string $token password reset token

\* @return static|null

\*/

public static function findByPasswordResetToken($token)

{

if (!static::isPasswordResetTokenValid($token)) {

return null;

}

return static::findOne([

'password\_reset\_token' => $token,

'status' => self::STATUS\_ACTIVE,

]);

}

/\*\*

\* Finds out if password reset token is valid

\*

\* @param string $token password reset token

\* @return boolean

\*/

public static function isPasswordResetTokenValid($token)

{

if (empty($token)) {

return false;

}

$expire = Yii::$app->params['user.passwordResetTokenExpire'];

$parts = explode('\_', $token);

$timestamp = (int) end($parts);

return $timestamp + $expire >= time();

}

/\*\*

\* @inheritdoc

\*/

public function getId()

{

return $this->getPrimaryKey();

}

/\*\*

\* @inheritdoc

\*/

public function getAuthKey()

{

return $this->ID;

}

/\*\*

\* @inheritdoc

\*/

public function validateAuthKey($authKey)

{

return $this->getAuthKey() === $authKey;

}

/\*\*

\* Validates password

\*

\* @param string $password password to validate

\* @return boolean if password provided is valid for current user

\*/

public function validatePassword($password)

{

return Yii::$app->security->validatePassword( $password, Yii::$app->security->generatePasswordHash($this->password) );

}

/\*\*

\* Generates password hash from password and sets it to the model

\*

\* @param string $password

\*/

public function setPassword($password)

{

$this->password\_hash = Yii::$app->security->generatePasswordHash($password);

}

/\*\*

\* Generates "remember me" authentication key

\*/

public function generateAuthKey()

{

$this->auth\_key = Yii::$app->security->generateRandomString();

}

/\*\*

\* Generates new password reset token

\*/

public function generatePasswordResetToken()

{

$this->password\_reset\_token = Yii::$app->security->generateRandomString() . '\_' . time();

}

/\*\*

\* Removes password reset token

\*/

public function removePasswordResetToken()

{

$this->password\_reset\_token = null;

}

1. **copy login-form form common/models/login-form**
2. **and paste it dispacture/models/login-form**
3. **and open dispatcher/controllers/SiteController.php change this**

**use common\models\LoginForm;**

**to**

**use dispatcher\models\LoginForm;**

**Model foriegn key data show in gridview**

<?= GridView::widget([

'dataProvider' => $dataProvider,

'filterModel' => $searchModel,

'columns' => [

['class' => 'yii\grid\SerialColumn'],

'name',

'username',

'email',

[

'attribute' => 'company\_id',

'label' => 'Company',

'value' => function( $model )

{

return $model->company->name;

// return Html::a( $data->name, Url::to( [ "company/view", "id" => $data->ID ] ) );

},

'format' => 'raw'

],

['class' => 'yii\grid\ActionColumn'],

],

]); ?>

**Model foriegn key search data show in gridview**

Add these lines for search to DriversearchModel

$CompanyModel = new CompanyModel();

$query->andFilterWhere([

'ID' => $this->ID,

'company\_id' => $CompanyModel->name

]);

$CompanyModel = CompanyModel::find();

$query->leftJoin( [ 'tbl\_company' => $CompanyModel ], "tbl\_company.ID=tbl\_driver.company\_id" );

$query->andFilterWhere(['like', 'tbl\_company.name', $this->company\_id])

**Check email exists code**

**Add this line to in your model for email unique**

['email', 'unique','message'=>"The email isn't correct"],

/\*\*

\* @inheritdoc

\*/

public function rules()

{

return [

[['name', 'email', 'username', 'password', 'address', 'status', 'image'], 'string', 'max' => 255],

[['name', 'email', 'username', 'password', 'address', 'status', 'company\_id'], 'required'],

[['company\_id'], 'integer'],

['email', 'email','message'=>"The email isn't correct"],

['email', 'unique','message'=>'Email already exists!'],

['username', 'unique','message'=>'username already exists!'],

];

}

**Get login user id and data**

Use this to get login user company id

Yii::$app->user->identity->company\_id;

User this to get loged user id

Yii::$app->user->getId();

**Add company id add to database without form field**

Add these line in actions create

$company\_id = Yii::$app->user->identity->company\_id;

$model->company\_id =$company\_id;

**Datetime picker**

<http://demos.krajee.com/widget-details/datepicker>

paste in Dispatcher/view/taxi /\_form

**use kartik\datetime\DateTimePicker;**

<?php

// Usage with model and Active Form (with no default initial value)

echo $form->field($model, 'luggage')->widget(DateTimePicker::classname(), [

'options' => ['placeholder' => 'Enter event time ...'],

'pluginOptions' => [

'autoclose' => true

]

]);

**Add dropdown in gridview search**

Paste this code in your view

Dispatcher/view/taxi/index.php

I am using this code for car model dropdown

**Dynamic**

**First add this code in your model class**

Mycase dispatcher/models/ Taximodel\_Models.php

public static function dropdown() {

$models = static::find()->all();

foreach ($models as $model) {

$dropdown[$model->name] = $model->name;

}

return $dropdown;

}

and then add this code in your index.php file

gridview   
  
 Dispatcher/view/taxi/index.php

use dispatcher\models\Taximodel\_Models;

[

'attribute' => 'model',

'label' => 'Model',

'filter' => Taximodel\_Models::dropdown(),

'value' => function($data11, $index, $dataColumn) {

$roleDropdown = Taximodel\_Models::dropdown();

// echo "<pre>"; print\_r($roleDropdown); exit;

return $roleDropdown[$data11->tModel->name];

},

],

**And gridview like this**

<?=

GridView::widget([

'dataProvider' => $dataProvider,

'filterModel' => $searchModel,

'columns' => [

['class' => 'yii\grid\SerialColumn'],

// 'ID',

'plates',

/\* [

'attribute' => 'model',

'label' => 'Model',

'value' => function( $data ) {

return $data->tModel->name;

// return $model->name;

// return Html::a( $data->name, Url::to( [ "company/view", "id" => $data->ID ] ) );

},

//'format' => 'raw'

],

\*/

[

'attribute' => 'model',

'label' => 'Model',

'filter' => Taximodel\_Models::dropdown(),

'value' => function($data11, $index, $dataColumn) {

// more optimized than $model->role->name;

$roleDropdown = Taximodel\_Models::dropdown();

// echo "<pre>"; print\_r($roleDropdown); exit;

return $roleDropdown[$data11->tModel->name];

},

],

[

'attribute' => 'Year',

'label' => 'Year',

'filter' => array\_combine(range(date("Y"), 1980), range(date("Y"), 1980)),

'value' => function($data112, $index, $dataColumn) {

// more optimized than $model->role->name;

$roleDropdown2 = array\_combine(range(date("Y"), 1980), range(date("Y"), 1980));

// echo "<pre>"; print\_r($data112); exit;

return $roleDropdown2[$data112->Year];

},

],

'capacity',

// 'speed',

// 'luggage',

// 'image',

// 'company\_id',

'status',

['class' => 'yii\grid\ActionColumn'],

],

]);

?>

**Ajax base pagination**

Before gridview add this line

<?php \yii\widgets\Pjax::begin(['timeout' => 10000, 'clientOptions' => ['container' => 'pjax-container']]); ?>

and after gridview add this

<?php \yii\widgets\Pjax::end(); ?>

**MENU LINKS SETTINGS**

**Open backend/views/layouts/main.php**

if (Yii::$app->user->isGuest) {

$menuItems[] = ['label' => 'Login', 'url' => ['/site/login']];

} else {

$menuItems = [

['label' => 'Driver', 'url' => ['/driver/index']],

['label' => 'Taxi', 'url' => ['/taxi/index']],

['label' => 'Logout (' . Yii::$app->user->identity->name . ')','url' => ['/site/logout'], 'linkOptions' => ['data-method' => 'post']],

];

}

**Change pagination size**

Put these lines in your data provider in search model

Backend/models/TaxiSearch.PHP

'pagination' => [

'pageSize' => 10,

],

**Create new component**

[**https://accounts.google.com/DisplayUnlockCaptcha**](https://accounts.google.com/DisplayUnlockCaptcha)

**google analytics checker api**

[**https://ga-dev-tools.appspot.com/explorer/**](https://ga-dev-tools.appspot.com/explorer/)

You can use your class to implement in yii

First create your class and paste in common directory

taximobility\common\components

in my case I am implement a google anyalitics class

and copy this file to taximobility\common\components

gapi.php

<?php

/\*\*

\* GAPI - Google Analytics PHP Interface

\*

\* http://code.google.com/p/gapi-google-analytics-php-interface/

\*

\* @copyright Stig Manning 2009

\*

\* This program is free software: you can redistribute it and/or modify

\* it under the terms of the GNU General Public License as published by

\* the Free Software Foundation, either version 3 of the License, or

\* (at your option) any later version.

\*

\* This program is distributed in the hope that it will be useful,

\* but WITHOUT ANY WARRANTY; without even the implied warranty of

\* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

\* GNU General Public License for more details.

\*

\* You should have received a copy of the GNU General Public License

\* along with this program. If not, see <http://www.gnu.org/licenses/>.

\*

\* @author Stig Manning <stig@sdm.co.nz>

\* @version 1.3

\*

\*/

namespace common\components;

use Yii;

use yii\base\Component;

class gapi extends Component

{

const http\_interface = 'auto'; //'auto': autodetect, 'curl' or 'fopen'

const client\_login\_url = 'https://www.google.com/accounts/ClientLogin';

const account\_data\_url = 'https://www.google.com/analytics/feeds/accounts/default';

const report\_data\_url = 'https://www.google.com/analytics/feeds/data';

const interface\_name = 'GAPI-1.3';

const dev\_mode = false;

private $auth\_token = null;

private $account\_entries = array();

private $account\_root\_parameters = array();

private $report\_aggregate\_metrics = array();

private $report\_root\_parameters = array();

private $results = array();

/\*\*

\* Constructor function for all new gapi instances

\*

\* Set up authenticate with Google and get auth\_token

\*

\* @param String $email

\* @param String $password

\* @param String $token

\* @return gapi

\*/

public function test($email, $password, $token=null)

{

if($token !== null)

{

$this->auth\_token = $token;

}

else

{

$this->authenticateUser($email,$password);

}

}

/\*\*

\* Return the auth token, used for storing the auth token in the user session

\*

\* @return String

\*/

public function getAuthToken()

{

return $this->auth\_token;

}

/\*\*

\* Request account data from Google Analytics

\*

\* @param Int $start\_index OPTIONAL: Start index of results

\* @param Int $max\_results OPTIONAL: Max results returned

\*/

public function requestAccountData($start\_index=1, $max\_results=20)

{

$response = $this->httpRequest(gapi::account\_data\_url, array('start-index'=>$start\_index,'max-results'=>$max\_results), null, $this->generateAuthHeader());

if(substr($response['code'],0,1) == '2')

{

return $this->accountObjectMapper($response['body']);

}

else

{

throw new Exception('GAPI: Failed to request account data. Error: "' . strip\_tags($response['body']) . '"');

}

}

/\*\*

\* Request report data from Google Analytics

\*

\* $report\_id is the Google report ID for the selected account

\*

\* $parameters should be in key => value format

\*

\* @param String $report\_id

\* @param Array $dimensions Google Analytics dimensions e.g. array('browser')

\* @param Array $metrics Google Analytics metrics e.g. array('pageviews')

\* @param Array $sort\_metric OPTIONAL: Dimension or dimensions to sort by e.g.('-visits')

\* @param String $filter OPTIONAL: Filter logic for filtering results

\* @param String $start\_date OPTIONAL: Start of reporting period

\* @param String $end\_date OPTIONAL: End of reporting period

\* @param Int $start\_index OPTIONAL: Start index of results

\* @param Int $max\_results OPTIONAL: Max results returned

\*/

public function requestReportData($report\_id, $dimensions, $metrics, $sort\_metric=null, $filter=null, $start\_date=null, $end\_date=null, $start\_index=1, $max\_results=30)

{

$parameters = array('ids'=>'ga:' . $report\_id);

if(is\_array($dimensions))

{

$dimensions\_string = '';

foreach($dimensions as $dimesion)

{

$dimensions\_string .= ',ga:' . $dimesion;

}

$parameters['dimensions'] = substr($dimensions\_string,1);

}

else

{

$parameters['dimensions'] = 'ga:'.$dimensions;

}

if(is\_array($metrics))

{

$metrics\_string = '';

foreach($metrics as $metric)

{

$metrics\_string .= ',ga:' . $metric;

}

$parameters['metrics'] = substr($metrics\_string,1);

}

else

{

$parameters['metrics'] = 'ga:'.$metrics;

}

if($sort\_metric==null&&isset($parameters['metrics']))

{

$parameters['sort'] = $parameters['metrics'];

}

elseif(is\_array($sort\_metric))

{

$sort\_metric\_string = '';

foreach($sort\_metric as $sort\_metric\_value)

{

//Reverse sort - Thanks Nick Sullivan

if (substr($sort\_metric\_value, 0, 1) == "-")

{

$sort\_metric\_string .= ',-ga:' . substr($sort\_metric\_value, 1); // Descending

}

else

{

$sort\_metric\_string .= ',ga:' . $sort\_metric\_value; // Ascending

}

}

$parameters['sort'] = substr($sort\_metric\_string, 1);

}

else

{

if (substr($sort\_metric, 0, 1) == "-")

{

$parameters['sort'] = '-ga:' . substr($sort\_metric, 1);

}

else

{

$parameters['sort'] = 'ga:' . $sort\_metric;

}

}

if($filter!=null)

{

$filter = $this->processFilter($filter);

if($filter!==false)

{

$parameters['filters'] = $filter;

}

}

if($start\_date==null)

{

$start\_date=date('Y-m-d',strtotime('1 month ago'));

}

$parameters['start-date'] = $start\_date;

if($end\_date==null)

{

$end\_date=date('Y-m-d');

}

$parameters['end-date'] = $end\_date;

$parameters['start-index'] = $start\_index;

$parameters['max-results'] = $max\_results;

$parameters['prettyprint'] = gapi::dev\_mode ? 'true' : 'false';

$response = $this->httpRequest(gapi::report\_data\_url, $parameters, null, $this->generateAuthHeader());

//HTTP 2xx

if(substr($response['code'],0,1) == '2')

{

return $this->reportObjectMapper($response['body']);

}

else

{

throw new Exception('GAPI: Failed to request report data. Error: "' . strip\_tags($response['body']) . '"');

}

}

/\*\*

\* Process filter string, clean parameters and convert to Google Analytics

\* compatible format

\*

\* @param String $filter

\* @return String Compatible filter string

\*/

protected function processFilter($filter)

{

$valid\_operators = '(!~|=~|==|!=|>|<|>=|<=|=@|!@)';

$filter = preg\_replace('/\s\s+/',' ',trim($filter)); //Clean duplicate whitespace

$filter = str\_replace(array(',',';'),array('\,','\;'),$filter); //Escape Google Analytics reserved characters

$filter = preg\_replace('/(&&\s\*|\|\|\s\*|^)([a-z]+)(\s\*' . $valid\_operators . ')/i','$1ga:$2$3',$filter); //Prefix ga: to metrics and dimensions

$filter = preg\_replace('/[\'\"]/i','',$filter); //Clear invalid quote characters

$filter = preg\_replace(array('/\s\*&&\s\*/','/\s\*\|\|\s\*/','/\s\*' . $valid\_operators . '\s\*/'),array(';',',','$1'),$filter); //Clean up operators

if(strlen($filter)>0)

{

return urlencode($filter);

}

else

{

return false;

}

}

/\*\*

\* Report Account Mapper to convert the XML to array of useful PHP objects

\*

\* @param String $xml\_string

\* @return Array of gapiAccountEntry objects

\*/

protected function accountObjectMapper($xml\_string)

{

$xml = simplexml\_load\_string($xml\_string);

$this->results = null;

$results = array();

$account\_root\_parameters = array();

//Load root parameters

$account\_root\_parameters['updated'] = strval($xml->updated);

$account\_root\_parameters['generator'] = strval($xml->generator);

$account\_root\_parameters['generatorVersion'] = strval($xml->generator->attributes());

$open\_search\_results = $xml->children('http://a9.com/-/spec/opensearch/1.1/');

foreach($open\_search\_results as $key => $open\_search\_result)

{

$report\_root\_parameters[$key] = intval($open\_search\_result);

}

$account\_root\_parameters['startDate'] = strval($google\_results->startDate);

$account\_root\_parameters['endDate'] = strval($google\_results->endDate);

//Load result entries

foreach($xml->entry as $entry)

{

$properties = array();

foreach($entry->children('http://schemas.google.com/analytics/2009')->property as $property)

{

$properties[str\_replace('ga:','',$property->attributes()->name)] = strval($property->attributes()->value);

}

$properties['title'] = strval($entry->title);

$properties['updated'] = strval($entry->updated);

$results[] = new gapiAccountEntry($properties);

}

$this->account\_root\_parameters = $account\_root\_parameters;

$this->results = $results;

return $results;

}

/\*\*

\* Report Object Mapper to convert the XML to array of useful PHP objects

\*

\* @param String $xml\_string

\* @return Array of gapiReportEntry objects

\*/

protected function reportObjectMapper($xml\_string)

{

$xml = simplexml\_load\_string($xml\_string);

$this->results = null;

$results = array();

$report\_root\_parameters = array();

$report\_aggregate\_metrics = array();

//Load root parameters

$report\_root\_parameters['updated'] = strval($xml->updated);

$report\_root\_parameters['generator'] = strval($xml->generator);

$report\_root\_parameters['generatorVersion'] = strval($xml->generator->attributes());

$open\_search\_results = $xml->children('http://a9.com/-/spec/opensearch/1.1/');

foreach($open\_search\_results as $key => $open\_search\_result)

{

$report\_root\_parameters[$key] = intval($open\_search\_result);

}

$google\_results = $xml->children('http://schemas.google.com/analytics/2009');

foreach($google\_results->dataSource->property as $property\_attributes)

{

$report\_root\_parameters[str\_replace('ga:','',$property\_attributes->attributes()->name)] = strval($property\_attributes->attributes()->value);

}

$report\_root\_parameters['startDate'] = strval($google\_results->startDate);

$report\_root\_parameters['endDate'] = strval($google\_results->endDate);

//Load result aggregate metrics

foreach($google\_results->aggregates->metric as $aggregate\_metric)

{

$metric\_value = strval($aggregate\_metric->attributes()->value);

//Check for float, or value with scientific notation

if(preg\_match('/^(\d+\.\d+)|(\d+E\d+)|(\d+.\d+E\d+)$/',$metric\_value))

{

$report\_aggregate\_metrics[str\_replace('ga:','',$aggregate\_metric->attributes()->name)] = floatval($metric\_value);

}

else

{

$report\_aggregate\_metrics[str\_replace('ga:','',$aggregate\_metric->attributes()->name)] = intval($metric\_value);

}

}

//Load result entries

foreach($xml->entry as $entry)

{

$metrics = array();

foreach($entry->children('http://schemas.google.com/analytics/2009')->metric as $metric)

{

$metric\_value = strval($metric->attributes()->value);

//Check for float, or value with scientific notation

if(preg\_match('/^(\d+\.\d+)|(\d+E\d+)|(\d+.\d+E\d+)$/',$metric\_value))

{

$metrics[str\_replace('ga:','',$metric->attributes()->name)] = floatval($metric\_value);

}

else

{

$metrics[str\_replace('ga:','',$metric->attributes()->name)] = intval($metric\_value);

}

}

$dimensions = array();

foreach($entry->children('http://schemas.google.com/analytics/2009')->dimension as $dimension)

{

$dimensions[str\_replace('ga:','',$dimension->attributes()->name)] = strval($dimension->attributes()->value);

}

$results[] = new gapiReportEntry($metrics,$dimensions);

}

$this->report\_root\_parameters = $report\_root\_parameters;

$this->report\_aggregate\_metrics = $report\_aggregate\_metrics;

$this->results = $results;

return $results;

}

/\*\*

\* Authenticate Google Account with Google

\*

\* @param String $email

\* @param String $password

\*/

protected function authenticateUser($email, $password)

{

$post\_variables = array(

'accountType' => 'GOOGLE',

'Email' => $email,

'Passwd' => $password,

'source' => gapi::interface\_name,

'service' => 'analytics'

);

$response = $this->httpRequest(gapi::client\_login\_url,null,$post\_variables);

//Convert newline delimited variables into url format then import to array

parse\_str(str\_replace(array("\n","\r\n"),'&',$response['body']),$auth\_token);

if(substr($response['code'],0,1) != '2' || !is\_array($auth\_token) || empty($auth\_token['Auth']))

{

throw new Exception('GAPI: Failed to authenticate user. Error: "' . strip\_tags($response['body']) . '"');

}

$this->auth\_token = $auth\_token['Auth'];

}

/\*\*

\* Generate authentication token header for all requests

\*

\* @return Array

\*/

protected function generateAuthHeader()

{

return array('Authorization: GoogleLogin auth=' . $this->auth\_token);

}

/\*\*

\* Perform http request

\*

\*

\* @param Array $get\_variables

\* @param Array $post\_variables

\* @param Array $headers

\*/

protected function httpRequest($url, $get\_variables=null, $post\_variables=null, $headers=null)

{

$interface = gapi::http\_interface;

if(gapi::http\_interface =='auto')

{

if(function\_exists('curl\_exec'))

{

$interface = 'curl';

}

else

{

$interface = 'fopen';

}

}

if($interface == 'curl')

{

return $this->curlRequest($url, $get\_variables, $post\_variables, $headers);

}

elseif($interface == 'fopen')

{

return $this->fopenRequest($url, $get\_variables, $post\_variables, $headers);

}

else

{

throw new Exception('Invalid http interface defined. No such interface "' . gapi::http\_interface . '"');

}

}

/\*\*

\* HTTP request using PHP CURL functions

\* Requires curl library installed and configured for PHP

\*

\* @param Array $get\_variables

\* @param Array $post\_variables

\* @param Array $headers

\*/

private function curlRequest($url, $get\_variables=null, $post\_variables=null, $headers=null)

{

$ch = curl\_init();

if(is\_array($get\_variables))

{

$get\_variables = '?' . str\_replace('&amp;','&',urldecode(http\_build\_query($get\_variables)));

}

else

{

$get\_variables = null;

}

curl\_setopt($ch, CURLOPT\_URL, $url . $get\_variables);

curl\_setopt($ch, CURLOPT\_RETURNTRANSFER, true);

curl\_setopt($ch, CURLOPT\_SSL\_VERIFYPEER, false); //CURL doesn't like google's cert

if(is\_array($post\_variables))

{

curl\_setopt($ch, CURLOPT\_POST, true);

curl\_setopt($ch, CURLOPT\_POSTFIELDS, $post\_variables);

}

if(is\_array($headers))

{

curl\_setopt($ch, CURLOPT\_HTTPHEADER,$headers);

}

$response = curl\_exec($ch);

$code = curl\_getinfo($ch,CURLINFO\_HTTP\_CODE);

curl\_close($ch);

return array('body'=>$response,'code'=>$code);

}

/\*\*

\* HTTP request using native PHP fopen function

\* Requires PHP openSSL

\*

\* @param Array $get\_variables

\* @param Array $post\_variables

\* @param Array $headers

\*/

private function fopenRequest($url, $get\_variables=null, $post\_variables=null, $headers=null)

{

$http\_options = array('method'=>'GET','timeout'=>3);

if(is\_array($headers))

{

$headers = implode("\r\n",$headers) . "\r\n";

}

else

{

$headers = '';

}

if(is\_array($get\_variables))

{

$get\_variables = '?' . str\_replace('&amp;','&',urldecode(http\_build\_query($get\_variables)));

}

else

{

$get\_variables = null;

}

if(is\_array($post\_variables))

{

$post\_variables = str\_replace('&amp;','&',urldecode(http\_build\_query($post\_variables)));

$http\_options['method'] = 'POST';

$headers = "Content-type: application/x-www-form-urlencoded\r\n" . "Content-Length: " . strlen($post\_variables) . "\r\n" . $headers;

$http\_options['header'] = $headers;

$http\_options['content'] = $post\_variables;

}

else

{

$post\_variables = '';

$http\_options['header'] = $headers;

}

$context = stream\_context\_create(array('http'=>$http\_options));

$response = @file\_get\_contents($url . $get\_variables, null, $context);

return array('body'=>$response!==false?$response:'Request failed, fopen provides no further information','code'=>$response!==false?'200':'400');

}

/\*\*

\* Case insensitive array\_key\_exists function, also returns

\* matching key.

\*

\* @param String $key

\* @param Array $search

\* @return String Matching array key

\*/

public static function array\_key\_exists\_nc($key, $search)

{

if (array\_key\_exists($key, $search))

{

return $key;

}

if (!(is\_string($key) && is\_array($search)))

{

return false;

}

$key = strtolower($key);

foreach ($search as $k => $v)

{

if (strtolower($k) == $key)

{

return $k;

}

}

return false;

}

/\*\*

\* Get Results

\*

\* @return Array

\*/

public function getResults()

{

if(is\_array($this->results))

{

return $this->results;

}

else

{

return;

}

}

/\*\*

\* Get an array of the metrics and the matchning

\* aggregate values for the current result

\*

\* @return Array

\*/

public function getMetrics()

{

return $this->report\_aggregate\_metrics;

}

/\*\*

\* Call method to find a matching root parameter or

\* aggregate metric to return

\*

\* @param $name String name of function called

\* @return String

\* @throws Exception if not a valid parameter or aggregate

\* metric, or not a 'get' function

\*/

public function \_\_call($name,$parameters)

{

if(!preg\_match('/^get/',$name))

{

throw new Exception('No such function "' . $name . '"');

}

$name = preg\_replace('/^get/','',$name);

$parameter\_key = gapi::array\_key\_exists\_nc($name,$this->report\_root\_parameters);

if($parameter\_key)

{

return $this->report\_root\_parameters[$parameter\_key];

}

$aggregate\_metric\_key = gapi::array\_key\_exists\_nc($name,$this->report\_aggregate\_metrics);

if($aggregate\_metric\_key)

{

return $this->report\_aggregate\_metrics[$aggregate\_metric\_key];

}

throw new Exception('No valid root parameter or aggregate metric called "' . $name . '"');

}

}

/\*\*

\* Class gapiAccountEntry

\*

\* Storage for individual gapi account entries

\*

\*/

class gapiAccountEntry

{

private $properties = array();

public function \_\_construct($properties)

{

$this->properties = $properties;

}

/\*\*

\* toString function to return the name of the account

\*

\* @return String

\*/

public function \_\_toString()

{

if(isset($this->properties['title']))

{

return $this->properties['title'];

}

else

{

return;

}

}

/\*\*

\* Get an associative array of the properties

\* and the matching values for the current result

\*

\* @return Array

\*/

public function getProperties()

{

return $this->properties;

}

/\*\*

\* Call method to find a matching parameter to return

\*

\* @param $name String name of function called

\* @return String

\* @throws Exception if not a valid parameter, or not a 'get' function

\*/

public function \_\_call($name,$parameters)

{

if(!preg\_match('/^get/',$name))

{

throw new Exception('No such function "' . $name . '"');

}

$name = preg\_replace('/^get/','',$name);

$property\_key = gapi::array\_key\_exists\_nc($name,$this->properties);

if($property\_key)

{

return $this->properties[$property\_key];

}

throw new Exception('No valid property called "' . $name . '"');

}

}

/\*\*

\* Class gapiReportEntry

\*

\* Storage for individual gapi report entries

\*

\*/

class gapiReportEntry

{

private $metrics = array();

private $dimensions = array();

public function \_\_construct($metrics,$dimesions)

{

$this->metrics = $metrics;

$this->dimensions = $dimesions;

}

/\*\*

\* toString function to return the name of the result

\* this is a concatented string of the dimesions chosen

\*

\* For example:

\* 'Firefox 3.0.10' from browser and browserVersion

\*

\* @return String

\*/

public function \_\_toString()

{

if(is\_array($this->dimensions))

{

return implode(' ',$this->dimensions);

}

else

{

return '';

}

}

/\*\*

\* Get an associative array of the dimesions

\* and the matching values for the current result

\*

\* @return Array

\*/

public function getDimesions()

{

return $this->dimensions;

}

/\*\*

\* Get an array of the metrics and the matchning

\* values for the current result

\*

\* @return Array

\*/

public function getMetrics()

{

return $this->metrics;

}

/\*\*

\* Call method to find a matching metric or dimension to return

\*

\* @param $name String name of function called

\* @return String

\* @throws Exception if not a valid metric or dimensions, or not a 'get' function

\*/

public function \_\_call($name,$parameters)

{

if(!preg\_match('/^get/',$name))

{

throw new Exception('No such function "' . $name . '"');

}

$name = preg\_replace('/^get/','',$name);

$metric\_key = gapi::array\_key\_exists\_nc($name,$this->metrics);

if($metric\_key)

{

return $this->metrics[$metric\_key];

}

$dimension\_key = gapi::array\_key\_exists\_nc($name,$this->dimensions);

if($dimension\_key)

{

return $this->dimensions[$dimension\_key];

}

throw new Exception('No valid metric or dimesion called "' . $name . '"');

}

}

Now open your taximobility\common\config\main.php

And paste this line in component array

'gapi' => [

'class' => 'common\components\gapi'

],

Please check these three linies in your class headers

namespace common\components;

use Yii;

use yii\base\Component;

Now final result is this

'components' => [

'cache' => [

'class' => 'yii\caching\FileCache',

],

'gapi' => [

'class' => 'common\components\gapi'

],

],

Now create a controller just simple using gii

Name in my case is GapiController.php

backend\controllers\GapiController

and now create db table analytics settings

CREATE TABLE `tbl\_analytics\_setting` (

`ID` int(11) NOT NULL AUTO\_INCREMENT,

`email` varchar(255) DEFAULT NULL,

`password` varchar(255) DEFAULT NULL,

`profile\_id` int(11) DEFAULT NULL,

`date\_created` datetime DEFAULT NULL,

PRIMARY KEY (`ID`)

) ENGINE=InnoDB AUTO\_INCREMENT=2 DEFAULT CHARSET=utf8mb4;

And create **model** and then **crud**

Now go to your **views**

In my case i used Gapicontroller

So my folder name is **gapi**

so my link Is

backend\views\gapi\index.php

and paste this code

<?php

//html helpers

use yii\helpers\Html;

//your model link

use company\models\AnalyticsSetting;

//model query

$analytics = AnalyticsSetting::find()->one();

$ga = Yii::$app->gapi;

//passing value test function which is in your calss

$ga->test( $analytics->email, $analytics->password );

//getting data requestReportData function which is in your calss

$ga->requestReportData( $analytics->profile\_id, [ 'browser', 'browserVersion' ], [ 'pageviews', 'visits' ] );

?>

<table>

<tr>

<th>Browser &amp; Browser Version</th>

<th>Pageviews</th>

<th>Visits</th>

</tr>

<?php

foreach( $ga->getResults() as $result ):

?>

<tr>

<td><?php echo $result ?></td>

<td><?php echo $result->getPageviews() ?></td>

<td><?php echo $result->getVisits() ?></td>

</tr>

<?php

endforeach

?>

</table>

<table>

<tr>

<th>Total Results</th>

<td><?php echo $ga->getTotalResults() ?></td>

</tr>

<tr>

<th>Total Pageviews</th>

<td><?php echo $ga->getPageviews() ?>

</tr>

<tr>

<th>Total Visits</th>

<td><?php echo $ga->getVisits() ?></td>

</tr>

<tr>

<th>Results Updated</th>

<td><?php echo $ga->getUpdated() ?></td>

</tr>

</table>

**Remove delete and view button form grid view**

Paste this code in your gridview

[

'class' => 'yii\grid\ActionColumn',

'template' => '{update}'

],

**Count query**

$tracks = backend\models\TracksModel::find()->where('1 = 1')->count();

Add JavaScript live link in view

echo yii\helpers\Html::jsFile("http://ajax.googleapis.com/ajax/libs/jquery/1.6.1/jquery.min.js");

echo yii\helpers\ Html::jsFile("http://maps.google.com/maps/api/js?sensor=false");

Add custom JavaScript link in view

$this->registerJsFile( Yii::getAlias( '@web' ) . '/js/loadgpx.js', [ 'position' => \yii\web\View::POS\_BEGIN ] );

**Add multiple grid view in views**

<h3>Photos</h3>

<?php

use yii\data\ActiveDataProvider;

$dataProvider = new ActiveDataProvider([

'query' => backend\models\TracksPhoto::find()->where(" track\_id='" . $model->id . "'"),

'pagination' => [

'pageSize' => 5,

],

]);

echo GridView::widget([

'dataProvider' => $dataProvider,

'columns' => [

['class' => 'yii\grid\SerialColumn'],

// Simple columns defined by the data contained in $dataProvider.

// Data from the model's column will be used.

[

'attribute' => 'photo\_link',

'format' => 'html',

'value' => function($data) {

//return $data->imageurl;

return Html::img("../../../" . $data->photo\_link, ['width' => '60']);

},

],

['class' => 'yii\grid\ActionColumn', 'template' => '{delete}'],

],

]);

?>

**Show image in grid view**

[

'attribute' => 'pic',

'label' => 'Picture',

'format' => 'html',

'filter' => false,

'value' => function($data) {

//return $data->imageurl;

if (!empty($data->pic)) {

return Html::img("../../../" . $data->pic, ['width' => '60']);

} else {

return Html::img("../../../upload/noimage/no-image.jpg", ['width' => '60']);

}

},

],

**Upload image in yii 2 advanced**

**First open model and paste this code**

**In my case I open**

**backend/models/sponsersModel.php**

public function uploadImage() {

// get the uploaded file instance. for multiple file uploads

// the following data will return an array (you may need to use

// getInstances method)

// image is taken from server

$image = UploadedFile::getInstance($this, 'image');

// if no image was uploaded abort the upload

if (empty($image)) {

return false;

}

// store the source file name

//$this->filename = $image->name;

$ext = end((explode(".", $image->name)));

// generate a unique file name

//$this->image is ur db image name field

$this->image = Yii::$app->security->generateRandomString().".{$ext}";

// the uploaded image instance

return $image;

}

And then open controller

In my case

**backend/sponsers/SponsesController.php**

**Action create**

Paste this code in action create

use yii\web\UploadedFile;

$image = $model->uploadImage();

//$model->image = $image->name;

if ($image !== false) {

$path = "../../../upload/sponsers/" . $model->image;

$model->image = "upload/sponsers/" . $model->image;

$image->saveAs($path);

}

///////////////////////////////**full code actioncreate example**/////////////////////////////////

public function actionCreate() {

$model = new SponsersModel();

if ($model->load(Yii::$app->request->post())) {

$image = $model->uploadImage();

//$model->image = $image->name;

if ($image !== false) {

$path = "../../../upload/sponsers/" . $model->image;

$model->image = "upload/sponsers/" . $model->image;

$image->saveAs($path);

}

$model->save();

Yii::$app->session->setFlash('success', 'Updoad Success fully');

return $this->redirect(['view', 'id' => $model->id]);

} else {

return $this->render('create', [

'model' => $model,

]);

}

}

///////////////////////////////////////////////////////////////////////////////////

Action Update

Paste this code in your action update

use yii\web\UploadedFile;

$image = $model->uploadImage();

if ($image !== false) {

$path = "../../../upload/sponsers/" . $model->image;

$model->image = "upload/sponsers/" . $model->image;

unlink("../../../" . $old\_file);

//echo "<pre>"; print\_r($model); exit;

$image->saveAs($path);

} else {

$model->image = $old\_file;

}

/////////////////////////////// **full code actionUpate example**/////////////////////////////////

public function actionUpdate($id) {

$model = $this->findModel($id);

$old\_file = $model->image;

if ($model->load(Yii::$app->request->post())) {

$image = $model->uploadImage();

if ($image !== false) {

$path = "../../../upload/sponsers/" . $model->image;

$model->image = "upload/sponsers/" . $model->image;

unlink("../../../" . $old\_file);

//echo "<pre>"; print\_r($model); exit;

$image->saveAs($path);

} else {

$model->image = $old\_file;

}

$model->save();

return $this->redirect(['view', 'id' => $model->id]);

} else {

return $this->render('update', [

'model' => $model,

]);

}

}

/////////////////////////////////////////////////////////

**Add date time in filter**

Paste this code in your gridview

[

'attribute' => 'add\_date',

'value' => 'add\_date',

'filter' => \yii\jui\DatePicker::widget([

'model' => $searchModel,

'attribute' => 'add\_date',

'clientOptions' => [

'autoclose' => true,

'format' => 'MM-dd-yyyy',

'todayHighlight' => true,

],

'dateFormat' => 'MM-dd-yyyy',

]),

'format' => 'html',

],

**//////////////////////////////////////////////////**

**Off Debug mode in yii2 advanced**

Go to backend/config/main-local.php

and comment these line

$config['bootstrap'][] = 'debug';

$config['modules']['debug'] = 'yii\debug\Module';

Or open cmd and write php init

//////////////////////////////////////////////////////

**Mobile number validation in yii2 advanced**

[['mno'], 'number', 'numberPattern' => '/^\s\*[-+]?[0-9]\*[.,]?[0-9]+([eE][-+]?[0-9]+)?\s\*$/'],

**//////////////////////////////////////////////////////////**

**Auth all pages in yii2**

Go to your controller directory and open your controllere.g sponsersController.php

And enter code in behaviors functions

'access' => [

'class' => AccessControl::className(),

'rules' => [

[

'actions' => ['login', 'error'],

'allow' => true,

],

[

'actions' => ['logout', 'index', 'create', 'update', 'delete', 'view'],

'allow' => true,

'roles' => ['@'],

],

],

],

////////////////////// **full code like this** ////////////////////////////////////////////

public function behaviors() {

return [

'access' => [

'class' => AccessControl::className(),

'rules' => [

[

'actions' => ['login', 'error'],

'allow' => true,

],

[

'actions' => ['logout', 'index', 'create', 'update', 'delete', 'view'],

'allow' => true,

'roles' => ['@'],

],

],

],

'verbs' => [

'class' => VerbFilter::className(),

'actions' => [

'delete' => ['post'],

],

],

];

}

**Show dropdown static list in form**

<?= $form->field($model, 'status')->dropDownList(['Active' => 'Active', 'Inactive' => 'Inactive'], [ 'prompt' => 'Select Status']) ?>

**Show dynamic dropdown list in form**

<?php

$users = UsersModel::find()->all();

$users\_list = array();

foreach ($users as $key => $user) {

$users\_list[$user->id] = $user->fname . " " . $user->lname;

}

echo $form->field($model, 'user\_id')->dropDownList($users\_list, ['prompt' => 'Select User']);

?>

**Register inline js in yii2 advanced**

<?php

$regijs = <<< JS

function CONVERTOtestasdfa(id, textval, type) {

alert("asdfads");

/\*if (type == 'KM') {

var TOTAL = textval/3.6;

$("#duration\_number\_" + id).html(TOTAL);

}

if (type == 'MILES') {

var TOTAL = textval/2.23694;

$("#duration\_number\_" + id).html(TOTAL);

} \*/

}

JS;

$this->registerJs($regijs);

?>

**Database access into your local file services**

$dataBase = new DataBase();

$db = include\_once 'advanced/common/config/main-local.php';

$db = $db["components"]["db"];

preg\_match('/^mysql:host=(.\*?);dbname=(.\*?)$/', $db["dsn"], $params);

$dbhost = $params[1];

$dbname = $params[2];

$dbuser = $db["username"];

$dbpass = $db["password"];

unset($params);

$dataBase->dbhost = $dbhost;

$dataBase->dbname = $dbname;

$dataBase->dbuser = $dbuser;

$dataBase->dbpass = $dbpass;

$dataBase->Connect\_Database();

**For pretty url in yii (url rewriting)**

'urlManager' => [

'class' => 'yii\web\UrlManager',

'enablePrettyUrl' => true,

'showScriptName' => true,

],

**For creating menu**

Include

views/layouts/main.php

use yii\widgets\Menu;

<?php

echo Menu::widget([

'items' => [

['label' => 'Home', 'url' => ['site/index']],

['label' => 'About', 'url' => ['site/about']],

],

'labelTemplate' =>'{label} Label',

'linkTemplate' => '<a href="{url}"><span>{label}</span></a>',

]);

?>

See tutorial

[**http://www.bsourcecode.com/yiiframework2/menu-widget-in-yii-framework-2-0/**](http://www.bsourcecode.com/yiiframework2/menu-widget-in-yii-framework-2-0/)

**validation**

<http://www.bsourcecode.com/yiiframework2/validation-rules-for-model-attributes-in-yiiframework-2-0/#yii2_compare_password_validation>

**URL RE-WRITING**

**Removing index.php**

Go to backend/config/main.php and write thiscode

'session' => [

'name' => 'backendSession',

'savePath' => \_\_DIR\_\_ . '/../../tmp/',

],

'urlManager' => [

'class' => 'yii\web\UrlManager',

'enablePrettyUrl' => true,

'showScriptName' => false

],

**'showScriptName' => false,** used for removing index.php

**Create htaccss file in /backend/web/.htaccess directory**

RewriteEngine on

# If a directory or a file exists, use it directly

RewriteCond %{REQUEST\_FILENAME} !-f

RewriteCond %{REQUEST\_FILENAME} !-d

# Otherwise forward it to index.php

RewriteRule . index.php

**Removing web folder from url**

Copy all the files and folder from **/backend/web/** directory into one up folder directory **(/backend/)**

**Change the index.php in /backend/ according because in up directory change occur**

**Remove** (../)

**Delete /web/** **folder**

**Add dropdown in your form**

**First write this code in your model action for e.g**

**Users->create**

**use backend\models\**JobCategory**;**

$JobType = JobCategory::find()->all();

$list = ArrayHelper::map( $JobType,'ID','name');

And then pass into create view like this

return $this->render('create', [

'model' => $model,

'list' => $list,

]);

AND then pass into \_form

And put these code into forms

<?= $form->field($model, 'categoryID')->dropDownList($list,['prompt'=>'Select Category']) ?>

**Add new panel in yii like backend**

**Here is new panel name is provider**

1. Take a copy of backend into **provider** (new panel name)

2. Add provider in common/config/bootstrip.php

3. Rename every backend text(occurences) into provider in /provider/ directory

4. UserProvider.php

4.1 Take copy of User.php into UserProvider.php in /common/models/ directory

4.2 Rename the table name from {{%user}} into {{users}}

4.3 Add Constant 'const STATUS\_TYPE = 'serviceProvider';' after 'const STATUS\_ACTIVE = 'active';'

4.4 add condition ''userType' => self::STATUS\_TYPE,' in 'function findIdentity($id)'

5. LoginFormProvider.php

5.1 Take copy of LoginForm.php into LoginFormProvider.php in /common/models/ directory

5.2 change '$this->\_user = User::findByUsername($this->username);' into '$this->\_user = UserProvider::findByUsername($this->username);' in **public function getUser()**

6. Change the identity class to 'User' in /provider/config/main.php into 'UserProvider'

7. SiteController.php

7.1 Change the use of **'use common\models\LoginForm;'** into 'use **common\models\LoginFormProvider;'** in /provider/controllers/SiteController.php

7.2 Create model by '$model = new LoginFormProvider();' instead of '$model = new LoginForm();' in /provider/controllers/SiteController.php

8. Creating ServiceProvider as same as backend user

public function actionCreate()

{

$model = new Users();

if ($model->load(Yii::$app->request->post())){

$model->password\_hash = Yii::$app->security->generatePasswordHash($model->password\_hash);

$model->auth\_key = Yii::$app->security->generateRandomString();

// $model->password\_reset\_token = Yii::$app->security->generateRandomString() . '\_' . time();

if($model->save()){

return $this->redirect(['view', 'id' => $model->ID]);

}

else {

return $this->render('create', [

'model' => $model,

]);

}

}

else {

return $this->render('create', [

'model' => $model,

]);

}

}

**9. Note: The users table must contain all the fields of user**

**10.**

**How to disable yii2 csrf token when u post data.**

1. In your desired controller create this variable:

**public $enableCsrfValidation;**

2. In your dsired action put this line

**$this->enableCsrfValidation = false;**

**Change backend panel theme very easily (updated 8/21/2015)**

**First create directory in backend**

**Backend/themes/mytheme**

**and then move your view folder in mytheme folder**

**and then go to config/main.php**

**'layout'=>'@webroot/themes/mytheme /views/layouts/main.php',**

**puth this code after this**

return [

'id' => 'app-backend',

**And also put this code in components section**

**'view' =>[**

**'theme' => [**

**'pathMap' => ['@app/views' => '@webroot/themes/mytheme /views'],**

**'baseUrl' => '@web/themes/botheme'**

**],**

**],**