



Yii2. Registration

sd@freematiq.com

<https://github.com/freematiq/php>



Настройка БД

Файл настроек - /config/db.php

```
'class' => 'yii\db\Connection',  
'dsn' => 'pgsql:host=localhost;dbname=yii2db',  
'username' => 'postgres',  
'password' => '',  
'charset' => 'utf8',
```

```
// Schema cache options (for production environment)  
// 'enableSchemaCache' => true,  
// 'schemaCacheDuration' => 60,  
// 'schemaCache' => 'cache',
```



Миграции

- migrate	Manages application migrations.
migrate/create	Creates a new migration.
migrate/down	Downgrades the application by reverting old migrations.
migrate/history	Displays the migration history.
migrate/mark	Modifies the migration history to the specified version.
migrate/new	Displays the un-applied new migrations.
migrate/redo	Redoes the last few migrations.
migrate/to	Upgrades or downgrades till the specified version.
migrate/up (default)	Upgrades the application by applying new migrations.

```
public function safeUp() {}  
public function safeDown() {}
```



Auth_key

В МИГРАЦИИ

```
$this->addColumn('{{%user}}', 'auth_key', $this->string(32));
```

User.php

```
public function generateAuthKey()
{
    $this->auth_key = \Yii::$app->security->generateRandomString();
}
```

RegisterForm.php

```
$user = new User();
$user->username = $this->username;
$user->email = $this->email;
$user->setPassword($this->password);
$user->generateAuthKey();
```



Создание таблиц

./yii migrate/create addUserTable

```
$this->createTable('users', [  
    'id' => $this->primaryKey()->comment('Первичный ключ'),  
    // 'имя поля' => $this->тип()->notNull()->comment('коммент');  
    'name' => $this->string('200')->notNull()->comment('Имя'),  
]);
```

->unique()

->defaultValue()



Редактирование полей

Добавление колонки

```
$this->addColumn('таблица', 'колонка',  
    'Команда (TIMESTAMP WITH TIME ZONE NOT NULL DEFAULT now())'  
);  
$this->addColumn($table, $column, $this->integer());  
$this->addCommentOnColumn('таблица', 'колонка', 'комментарий');
```

Удаление колонки

```
$this->dropColumn($table, $column);
```



Работа с записями

Вставка

```
$this->insert('table', [  
    'field' => 'value',  
    ...,  
]);
```

Обновление

```
$this->update('table', [  
    'field' => 'condition',  
    ...,  
], [  
    'field' => 'value',  
    ...,  
]);
```



SQL - Запросы

Прямой запрос в БД

```
$this->execute('update ... where ...');
```




URL

```
'urlManager' => [  
    'enablePrettyUrl' => true,  
    'showScriptName' => false,  
    'rules' => [  
        '<controller:\w+>/<id:\d+>' => '<controller>/view',  
        '<controller:\w+>/<action:\w+>/<id:\d+>' => '<controller>/<action>',  
        '<controller:\w+>/<action:\w+>' => '<controller>/<action>',  
        'customer/<id:\d+>' => 'customer/view',  
        'about' => 'site/about',  
    ],  
],
```



Class User

```
/**
 * User model
 *
 * @property integer $id
 * @property string $username
 * @property string $password_hash
 * @property string $email
 * @property integer $status
 * @property string $password write-only password
 */

class User extends \yii\db\ActiveRecord implements
\yii\web\IdentityInterface
```



ActiveRecord

```
extends \yii\db\ActiveRecord
```

```
public static function findIdentity($id)
{
    return static::findOne(['id' => $id, 'status' => self::STATUS_ACTIVE]);
}
```

```
public static function findByUsername($username)
{
    return static::findOne(['username' => $username,
        'status' => self::STATUS_ACTIVE]);
}
```

::findOne(); - поиск объекта в базе

```
public static function tableName()
{
    // http://www.yiiframework.com/doc-2.0/guide-db-dao.html#quoting-table-and-column-names
    return '{{%user}}';
}
```



Form rules

```
public function rules()
{
    return [
        ['username', 'trim'],
        ['username', 'required'],
        ['username', 'unique', 'targetClass' => '\app\models\User', 'message' => 'This
username has already been taken.'],
        ['username', 'string', 'min' => 2, 'max' => 255],
        ['email', 'trim'],
        ['email', 'required'],
        ['email', 'email'],
        ['email', 'string', 'max' => 255],
        ['email', 'unique', 'targetClass' => '\app\models\User', 'message' => 'This email
address has already been taken.'],
        ['password', 'required'],
        ['password', 'string', 'min' => 6],
    ];
}
```



Register method

```
public function register()
{
    if (!$this->validate()) {
        return null;
    }

    $user = new User();
    $user->username = $this->username;
    $user->email = $this->email;
    $user->setPassword($this->password);

    return $user->save() ? $user : null;
}
```



Вывод формы

```
public function actionRegister()
{
    $model = new \app\models\RegisterForm();

    if ($model->load(Yii::$app->request->post())) {
        if ($user = $model->register()) {
            if (Yii::$app->getUser()->login($user)) {
                return $this->goHome();
            }
        }
    }

    return $this->render('register', [
        'model' => $model,
    ]);
}
```



Вывод формы

register.php

```
<?= $form->field($model, 'username')->textInput(['autofocus' => true]) ?>
```

```
<?= $form->field($model, 'email')->textInput() ?>
```

```
<?= $form->field($model, 'password')->passwordInput() ?>
```



Домашнее задание

- Подключить БД к проекту
- Написать миграцию для создания пользователя
- Создать модель пользователя
- Создать форму регистрации
- Реализовать регистрацию
- Учитывать миграционность!