实现文档

本项目是基于 Yii2 开发的宠物寄养管理系统，采用标准的 MVC（Model-View-Controller）设计模式。系统包含前台 frontend 和后台 backend 两个应用模块。

一、Model 层（模型层）

Model 层负责数据处理、业务逻辑和数据库交互。

1.1 核心模型设计

模型文件位于 `common/models/` 目录下，所有实体类继承自 `yii\db\ActiveRecord`，实现了 Active Record 模式。

1.1.1 主要实体模型

示例：OrderEmployee 模型

代码块

```php
// common/models/OrderEmployee.php
namespace common\models;

class OrderEmployee extends \yii\db\ActiveRecord
{
    // 指定数据库表名
    public static function tableName()
    {
        return 'order_employee';
    }

    // 数据验证规则
    public function rules()
    {
        return [
            [['OrderID', 'EmployeeID'], 'required'],
            [['OrderID', 'EmployeeID'], 'integer'],
            [['OrderID', 'EmployeeID'], 'unique', 'targetAttribute' => ['OrderID', 'EmployeeID']],
            [['EmployeeID'], 'exist', 'skipOnError' => true, 'targetClass' => Employee::className()],
            [['OrderID'], 'exist', 'skipOnError' => true, 'targetClass' => Fosterorder::className()],
        ];
    }

    // 属性标签（用于表单显示）
    public function attributeLabels()
    {
        return [
            'OrderID' => '订单编号',
            'EmployeeID' => '员工编号',
        ];
    }

    // 关联查询：一对一关系
    public function getEmployee()
    {
        return $this->hasOne(Employee::className(), ['EmployeeID' => 'EmployeeID']);
    }

    public function getOrder()
    {
        return $this->hasOne(Fosterorder::className(), ['OrderID' => 'OrderID']);
    }
}
```

其他主要模型：

- **Pet**：宠物基础信息（PetID, PetName, PetSpecies, PetAge, PetGender）
- **Dog**：狗类宠物扩展信息（继承Pet，增加 DogBreedType, TrainingLevel）
- **Cat**：猫类宠物扩展信息（继承Pet，增加 CatBreedType, EyeColor）
- **Customer**：客户信息（CustomerID, UserID, CustomerName, Gender, PhoneNumber 等）
- **Employee**：员工信息（EmployeeID, UserID, EmployeeName, Gender, Salary 等）
- **Fosterorder**：寄养订单（OrderID, CustomerID, PetID, 时间、价格等）
- **Fosterservice**：寄养服务（ServiceID, ServiceName, ServicePrice, ServiceContent）
- **User**：用户认证（UserID, username, password, role）

1.2 搜索模型（Search Model）

搜索模型位于 `backend/models/` 目录下，继承自对应的实体模型，专门用于处理列表页的搜索和过滤功能。

示例：OrderEmployeeSearch

```php
// backend/models/OrderEmployeeSearch.php
namespace backend\models;

use yii\data\ActiveDataProvider;

class OrderEmployeeSearch extends OrderEmployee
{
    public function search($params)
    {
        // 创建查询对象
        $query = OrderEmployee::find();

        // 配置数据提供者
        $dataProvider = new ActiveDataProvider([
            'query' => $query,
        ]);

        // 加载搜索参数
        $this->load($params);

        if (!$this->validate()) {
            return $dataProvider;
        }

        // 添加过滤条件
        $query->andFilterWhere([
            'OrderID' => $this->OrderID,
            'EmployeeID' => $this->EmployeeID,
        ]);

        return $dataProvider;
    }
}
```

1.3 Model 层职责总结

- **数据验证**：通过 `rules()` 定义字段验证规则（必填、类型、唯一性、外键约束等）
- **数据库映射**：通过 `tableName()` 指定对应的数据表
- **关联关系**：通过 `hasOne()` 和 `hasMany()` 定义模型间的关联
- **业务逻辑**：封装复杂的数据处理逻辑
- **数据查询**：提供 ActiveDataProvider 进行分页、排序、过滤

二、Controller 层（控制器层）

Controller 层负责处理用户请求、调用 Model 处理数据、选择合适的 View 进行渲染。

2.1 控制器结构

控制器文件位于 `backend/controllers/` 目录下，继承自 `yii\web\Controller`。

示例：OrderEmployeeController

代码块

```php
// backend/controllers/OrderEmployeeController.php
namespace backend\controllers;

use yii\web\Controller;
use yii\web\NotFoundHttpException;
use yii\filters\AccessControl;

class OrderEmployeeController extends Controller
{
    // 行为配置
    public function behaviors()
    {
        return [
            // 访问控制
            'access' => [
                'class' => AccessControl::className(),
                'rules' => [
                    [
                        'allow' => true,
                        'roles' => ['@'],
                        'matchCallback' => function ($rule, $action) {
                            $userRole = Yii::$app->user->identity->role ??
null;
                            return $userRole === 'admin';  // 仅管理员可访问
                        },
                    ],
                ],
                'denyCallback' => function ($rule, $action) {
                    throw new ForbiddenHttpException('您没有权限访问此资源。');
                },
            ],
            // HTTP 方法过滤
            'verbs' => [
                'class' => VerbFilter::className(),
                'actions' => [
                    'delete' => ['POST'],  // 删除操作只允许 POST 请求
                ],
            ],
        ];
    }

    // 列表页
    public function actionIndex()
    {
        $searchModel = new OrderEmployeeSearch();
        $dataProvider = $searchModel->search(Yii::$app->request->queryParams);

        return $this->render('index', [
            'searchModel' => $searchModel,
            'dataProvider' => $dataProvider,
        ]);
    }

    // 详情页
    public function actionView($OrderID, $EmployeeID)
    {
        return $this->render('view', [
            'model' => $this->findModel($OrderID, $EmployeeID),
        ]);
    }

    // 创建
    public function actionCreate()
    {
        $model = new OrderEmployee();

        if ($model->load(Yii::$app->request->post()) && $model->save()) {
            return $this->redirect(['view',
                'OrderID' => $model->OrderID,
                'EmployeeID' => $model->EmployeeID
            ]);
        }

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

    // 更新
    public function actionUpdate($OrderID, $EmployeeID)
    {
        $model = $this->findModel($OrderID, $EmployeeID);

        if ($model->load(Yii::$app->request->post()) && $model->save()) {
            return $this->redirect(['view',
                'OrderID' => $model->OrderID,
                'EmployeeID' => $model->EmployeeID
            ]);
        }

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

    // 删除
    public function actionDelete($OrderID, $EmployeeID)
    {
        $this->findModel($OrderID, $EmployeeID)->delete();
        return $this->redirect(['index']);
    }

    // 辅助方法：查找模型
    protected function findModel($OrderID, $EmployeeID)
    {
        if (($model = OrderEmployee::findOne(['OrderID' => $OrderID,
'EmployeeID' => $EmployeeID])) !== null) {
            return $model;
        }

        throw new NotFoundHttpException('请求的页面不存在。');
    }
}
```

2.2 其他主要控制器

- **CustomerController**：客户管理（增删改查）
- **EmployeeController**：员工管理
- **PetController**：宠物管理
- **DogController**：狗类宠物管理
- **CatController**：猫类宠物管理
- **FosterorderController**：寄养订单管理
- **FosterserviceController**：寄养服务管理
- **SiteController**：站点通用页面（登录、首页等）

2.3 Controller 层职责总结

请求处理：接收和解析用户请求参数

权限控制：通过 AccessControl 实现基于角色的访问控制（RBAC）

业务协调：调用 Model 层进行数据操作

响应生成：选择合适的 View 并传递数据

异常处理：处理 404、403 等异常情况

重定向：操作成功后跳转到相应页面

三、View 层（视图层）

View 层负责数据展示和用户交互界面的渲染。

3.1 视图结构

视图文件位于 `backend/views/` 目录下，每个控制器对应一个视图目录。

OrderEmployee 视图目录结构：

代码块

```
backend/views/orderemployee/
├── index.php        # 列表页
├── view.php         # 详情页
├── create.php       # 创建页
├── update.php       # 更新页
├── _form.php        # 表单组件（create 和 update 共用）
└── _search.php      # 搜索表单组件
```

3.2 列表视图（index.php）

代码块

```php
<?php
use yii\helpers\Html;
use yii\grid\GridView;
use yii\widgets\Pjax;

$this->title = '订单-员工';
$this->params['breadcrumbs'][] = $this->title;
$role = Yii::$app->user->identity->role ?? 'guest';
$isAdmin = $role === 'admin';
?>
```

```php

<div class="order-employee-index">
    <h1><?= Html::encode($this->title) ?></h1>

    <?php if ($isAdmin): ?>
        <p>
            <?= Html::a('新增关联', ['create'], ['class' => 'btn btn-success']) ?>
        </p>
    <?php endif; ?>

    <?php Pjax::begin(); ?>

    <?= GridView::widget([
        'dataProvider' => $dataProvider,
        'filterModel' => $searchModel,
        'columns' => [
            ['class' => 'yii\grid\SerialColumn'],
            'OrderID',
            'EmployeeID',
            $isAdmin ? [
                'class' => 'yii\grid\ActionColumn',
                'buttons' => [
                    'view' => function($url,$model,$key){
                        return Html::a('查看',['view','OrderID'=>$model->OrderID,'EmployeeID'=>$model->EmployeeID]);
                    },
                    'update' => function($url,$model,$key){
                        return Html::a('编辑',['update','OrderID'=>$model->OrderID,'EmployeeID'=>$model->EmployeeID]);
                    },
                    'delete' => function($url,$model,$key){
                        return Html::a('删除',['delete','OrderID'=>$model->OrderID,'EmployeeID'=>$model->EmployeeID],[
                            'data'=>['confirm'=>'确认删除该关联? ','method'=>'post']
                        ]);
                    },
                ],
            ] : [
                'class' => 'yii\grid\ActionColumn',
                'template' => '{view}',
            ],
        ],
    ]); ?>

    <?php Pjax::end(); ?>
```

```
</div>
```

关键组件说明：

GridView：数据表格组件，支持排序、分页、过滤

Pjax：Ajax 局部刷新，无需整页加载

Html::a()：生成超链接

权限判断：根据用户角色显示不同操作按钮

3.3 表单视图（_form.php）

代码块

```
<?php
use yii\helpers\Html;
use yii\widgets\ActiveForm;
?>

<div class="order-employee-form">
    <?php $form = ActiveForm::begin(); ?>

    <?= $form->field($model, 'OrderID')
        ->textInput(['readonly' => true, 'style' => 'background-color: #f5f5f5;'])
        ->label('订单编号') ?>

    <?= $form->field($model, 'EmployeeID')
        ->textInput(['readonly' => true, 'style' => 'background-color: #f5f5f5;'])
        ->label('员工编号') ?>

    <div class="form-group">
        <?= Html::submitButton($model->isNewRecord ? '创建' : '保存', ['class' => 'btn btn-success']) ?>
    </div>

    <?php ActiveForm::end(); ?>
</div>
```

表单特性：

ActiveForm：自动生成表单并绑定模型

字段验证：前端自动生成验证规则（基于 Model 的 rules）

CSRF 防护：自动添加 CSRF token

Ajax 验证：支持实时字段验证

3.4 详情视图（view.php）

使用 DetailView 组件展示单条记录的详细信息：

代码块

```php
<?php
use yii\helpers\Html;
use yii\widgets\DetailView;

$this->title = $model->OrderID;
?>

<div class="order-employee-view">
    <h1><?= Html::encode($this->title) ?></h1>

    <p>
        <?= Html::a('编辑', ['update', 'OrderID' => $model->OrderID, 'EmployeeID' => $model->EmployeeID], ['class' => 'btn btn-primary']) ?>
        <?= Html::a('删除', ['delete', 'OrderID' => $model->OrderID, 'EmployeeID' => $model->EmployeeID], [
            'class' => 'btn btn-danger',
            'data' => [
                'confirm' => '确认删除此记录？',
                'method' => 'post',
            ],
        ]) ?>
    </p>

    <?= DetailView::widget([
        'model' => $model,
        'attributes' => [
            'OrderID',
            'EmployeeID',
        ],
    ]) ?>
</div>
```

3.5 View 层职责总结

数据展示：使用 GridView、DetailView 等组件展示数据

表单渲染：使用 ActiveForm 生成表单

用户交互：提供按钮、链接等交互元素

权限控制：根据用户角色显示不同内容

XSS 防护：使用 `Html::encode()` 转义输出

布局应用：继承统一的布局模板

四、MVC 协作流程

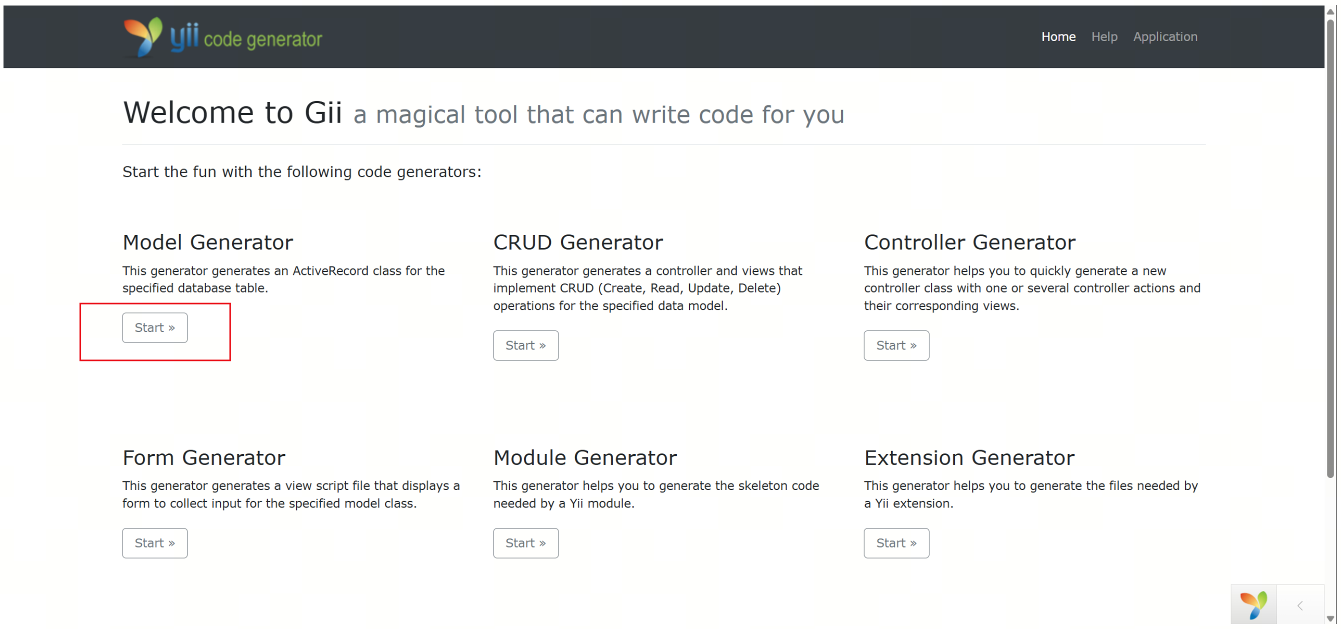
以"查看订单-员工列表"为例：

代码块

```
用户请求
  ↓
1. 路由解析：/orderemployee/index → OrderEmployeeController::actionIndex()
  ↓
2. 权限验证：AccessControl 检查用户是否为 admin 角色
  ↓
3. Controller 创建 SearchModel 和 DataProvider
  ↓
4. Model 执行数据库查询，应用过滤条件
  ↓
5. Controller 调用 render() 方法，传递数据到 View
  ↓
6. View 使用 GridView 渲染数据表格
  ↓
7. 返回 HTML 响应给用户
```

五、Yii生成Model、CRUD等过程

进入yii的主页，点击strart按钮开始自动生成。前提是必须制作好数据库中的数据表。



Welcome to Gii a magical tool that can write code for you

Start the fun with the following code generators:

Model Generator
This generator generates an ActiveRecord class for the specified database table.
Start »

CRUD Generator
This generator generates a controller and views that implement CRUD (Create, Read, Update, Delete) operations for the specified data model.
Start »

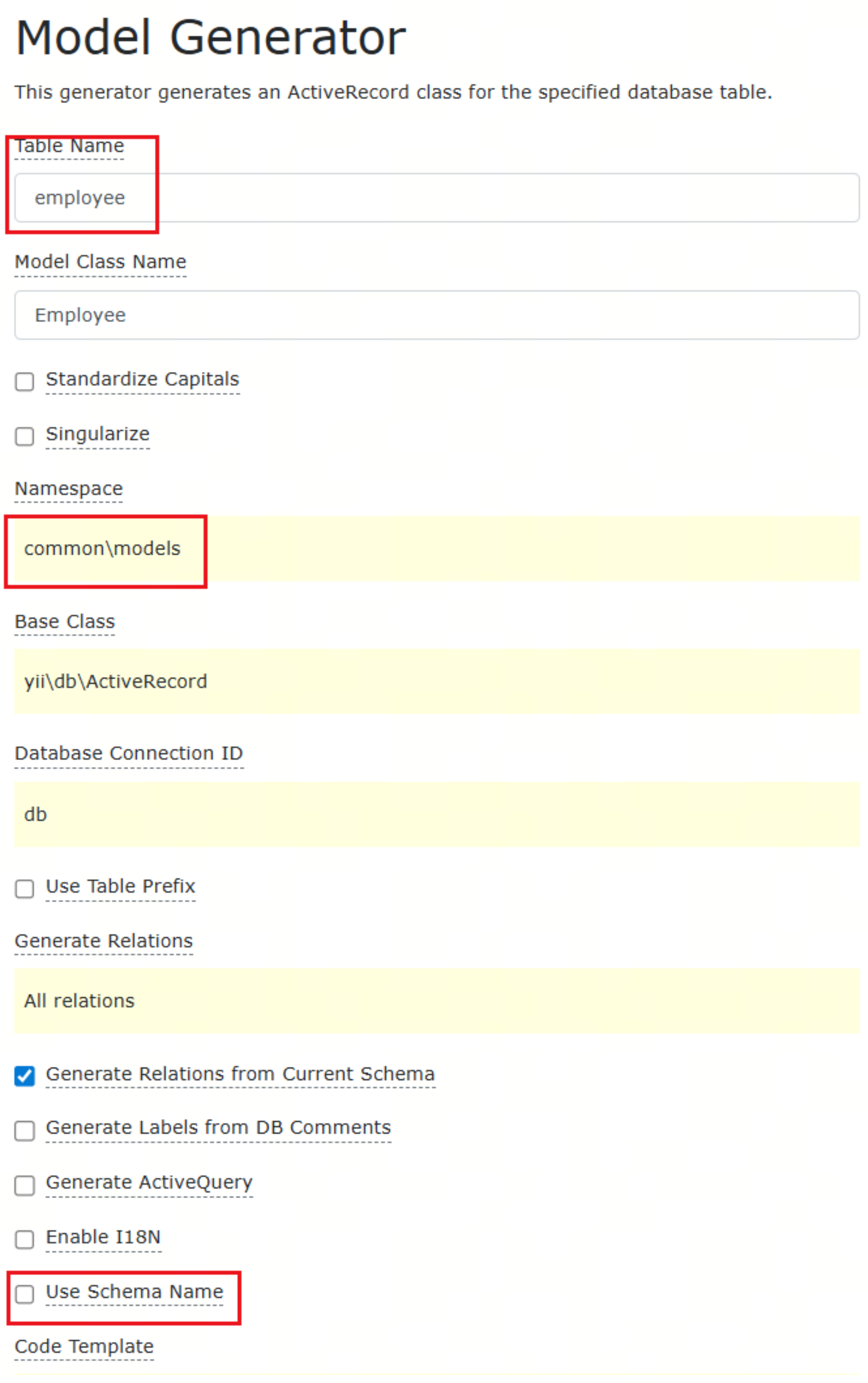
Controller Generator
This generator helps you to quickly generate a new controller class with one or several controller actions and their corresponding views.
Start »

Form Generator
This generator generates a view script file that displays a form to collect input for the specified model class.
Start »

Module Generator
This generator helps you to generate the skeleton code needed by a Yii module.
Start »

Extension Generator
This generator helps you to generate the files needed by a Yii extension.
Start »

进入Model Generator的内部，选中



Model Generator

This generator generates an ActiveRecord class for the specified database table.

Table Name: employee

Model Class Name: Employee

[ ] Standardize Capitals

[ ] Singularize

Namespace: common\models

Base Class: yii\db\ActiveRecord

Database Connection ID: db

[ ] Use Table Prefix

Generate Relations: All relations

[✓] Generate Relations from Current Schema

[ ] Generate Labels from DB Comments

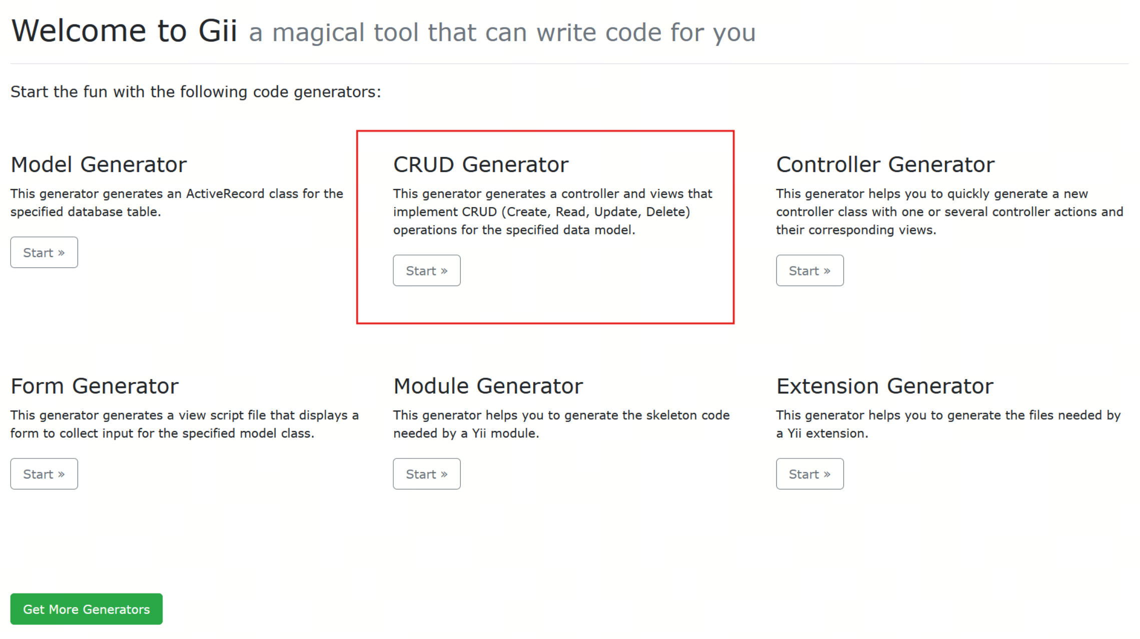
[ ] Generate ActiveQuery

[ ] Enable I18N

[ ] Use Schema Name

Code Template

然后进行CRUD的生成，如下图：



Welcome to Gii a magical tool that can write code for you

Start the fun with the following code generators:

Model Generator

This generator generates an ActiveRecord class for the specified database table.

Start »

CRUD Generator

This generator generates a controller and views that implement CRUD (Create, Read, Update, Delete) operations for the specified data model.

Start »

Controller Generator

This generator helps you to quickly generate a new controller class with one or several controller actions and their corresponding views.

Start »

Form Generator

This generator generates a view script file that displays a form to collect input for the specified model class.

Start »

Module Generator

This generator helps you to generate the skeleton code needed by a Yii module.

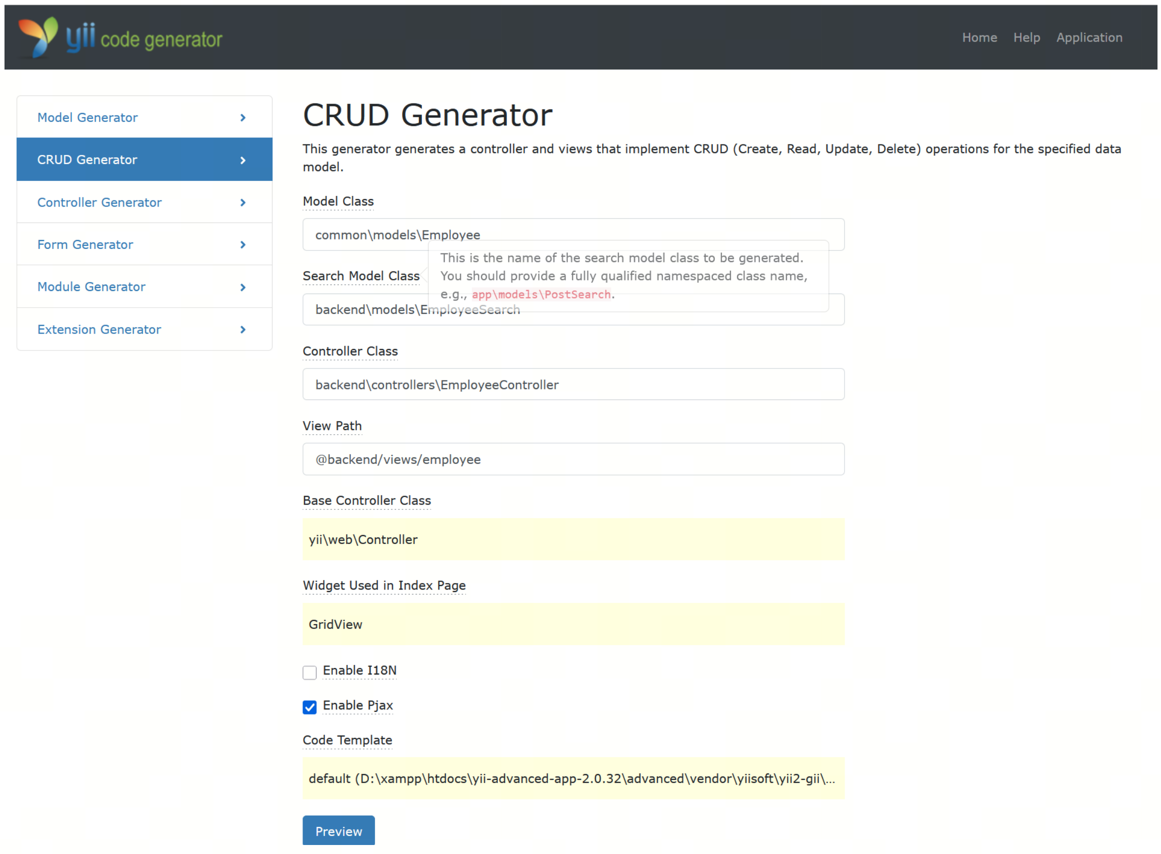
Start »

Extension Generator

This generator helps you to generate the files needed by a Yii extension.

Start »

Get More Generators



CRUD Generator

This generator generates a controller and views that implement CRUD (Create, Read, Update, Delete) operations for the specified data model.

Model Class: common\models\Employee

This is the name of the search model class to be generated. You should provide a fully qualified namespaced class name, e.g., `app\models\PostSearch`.

Search Model Class: backend\models\EmployeeSearch

Controller Class: backend\controllers\EmployeeController

View Path: @backend/views/employee

Base Controller Class: yii\web\Controller

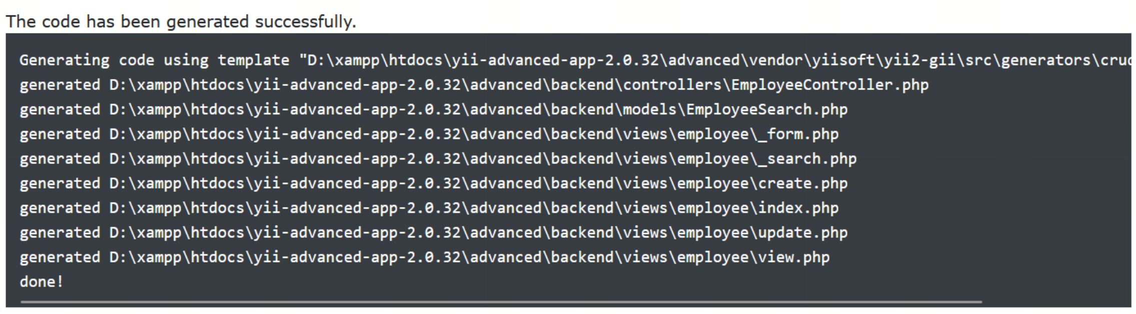
Widget Used in Index Page: GridView

☐ Enable I18N

☑ Enable Pjax

Code Template: default (D:\xampp\htdocs\yii-advanced-app-2.0.32\advanced\vendor\yiisoft\yii2-gii\...

Preview



The code has been generated successfully.

```
Generating code using template "D:\xampp\htdocs\yii-advanced-app-2.0.32\advanced\vendor\yiisoft\yii2-gii\src\generators\cru
generated D:\xampp\htdocs\yii-advanced-app-2.0.32\advanced\backend\controllers\EmployeeController.php
generated D:\xampp\htdocs\yii-advanced-app-2.0.32\advanced\backend\models\EmployeeSearch.php
generated D:\xampp\htdocs\yii-advanced-app-2.0.32\advanced\backend\views\employee\_form.php
generated D:\xampp\htdocs\yii-advanced-app-2.0.32\advanced\backend\views\employee\_search.php
generated D:\xampp\htdocs\yii-advanced-app-2.0.32\advanced\backend\views\employee\create.php
generated D:\xampp\htdocs\yii-advanced-app-2.0.32\advanced\backend\views\employee\index.php
generated D:\xampp\htdocs\yii-advanced-app-2.0.32\advanced\backend\views\employee\update.php
generated D:\xampp\htdocs\yii-advanced-app-2.0.32\advanced\backend\views\employee\view.php
done!
```