目前我们在详情页提交的评论，提交之后评论会直接在评论列表中进行显示。我们需要为评论添加审核功能。新添加的评论只有审核通过之后才会在评论列表进行显示。而审核不通过的评论，我们可以编辑评论的内容再次进行审核，直到审核通过。

评论的审核流程我们可以定义为一个状态机。回到项目，首先我们来修改Comment类，在Comment类中我们需要添加一个属性来保存当前评论对象的状态。打开控制台，输入`symfony console make:entity`，类名我们输入Comment，属性名称我们叫做status，类型这里选择string类型，长度255，在数据库中可以为空吗？这里选择yes。回车要退出命令行。

我们查看一下命令行的更改，在Comment类中添加了$status属性，状态机同时只能保存一个状态，这里使用string类型没有问题，我们修改对应的get方法和set方法，在setStatus()方法中我们添加第二个参数$context。

```
#src/Entity/Comment.php

class Comment
{
    // ...

    /**
    * @ORM\Column(type="string", length=255, nullable=true)
    */
    private $status;

    public function getStatus(): ?string
    {
        return $this->status;
    }

    public function setStatus(?string $status, $context = []): self
    {
        $this->status = $status;

        return $this;
    }
    // ...
}
```

我们还需要创建migration文件，打开控制台，复制命令行，粘贴。执行数据库的更改，粘贴输入yes。现在数据库comment表中就添加了status列，现在我们就可以定义评论的工作流了。

打开workflow.yaml配置文件，在workflows配置下，我们定义新的工作流名称叫做`comment_check`。我们复制之前定义好的工作流，然后进行修改，粘贴，类型这里我们选择状态机`status_machine`。这里需要调整一下对齐，属性仍然是status，当前状态机所对应的类是Comment类，我们定一些状态。

首先是新的评论，然后是检查状态checking。如果审核不通过的话，我们需要编辑，然后是拒绝状态rejected，然后是approved。初始化的状态是new状态，我们定一些状态的转换。首先是`start_check`，`from: new`，`to: checking`。然后是审核未通过`check_rejected`，`from: checking`，`to: rejected`。然后是`re_edit`重新编辑，`from:rejected`，`to: edit`。然后是重新审核`re_check`，这里

from: edit, to: checking。最后是审核通过check_approved, from: checking, to: approved。

```
#config/packages/workflow.yaml

framework:
    workflows:
        comment_check:
            type: 'state_machine'
            audit_trail:
                enabled: true
            marking_store:
                type: 'method'
                property: 'status'
            supports:
                - App\Entity\Comment
            initial_marking: new
            places:
                - new
                - checking
                - edit
                - rejected
                - approved
            transitions:
                start_check:
                    from: new
                    to: checking
                check_rejected:
                    from: checking
                    to: rejected
                re_edit:
                    from: rejected
                    to: edit
                re_check:
                    from: edit
                    to: checking
                check_approved:
                    from: checking
                    to: approved
```

这样我们就定义好了评论的工作流，我们打开控制台生成一下评论工作流的图表，我们使用symfony console workflow:dump comment_check，我们使用dot库，图片的类型这里输入SVG，输出文件的名称，我们叫做comment_check.svg。

```
php bin/console workflow:dump comment_check | dot -Tsvg -o
comment_check.svg
```

回车，我们查看项目根目录，在根目录下就生成了SVG文件。查看一下图片，新提交的评论初始化为new状态。当点击开始检查之后，转换为checking状态，如果检查失败的话，我们需要重新编辑，重新编辑成功之后开始检查，如果检查成功的话，状态会修改approved。这就是我们期望的状态机的流程。

我们打开vendor目录查看一下工作流的源码。在symfony目录下，我们打开workflow目录。在workflow根目录下有个Workflow.php文件，我们打开这个文件，Workflow类就是我们工作流的服务类。然后对应的还有个StateMachine类，这个类就是状态机的类，我们查看一下状态机类的源码。状态机类它继承于Workflow类，在类中只对构造方法有个修改，在构造方法中唯一的不同就是状态机同时只能保存一个状态，其他的方法和工作流是一样的。

这里我们就不再详细的介绍了，如果有兴趣的话，我们可以查看Symfony官方提供的一个案例，案例的代码在github上进行了开源，我们已经定义了文章操作的工作流。

当有新的文章发表时，对应角色的用户对文章进行审核。我们可以在新的文章创建时，可以向对应的角色用户发送邮件，通知他们来完成相应的工作。在下节课，我们先来安装邮件组件，使用邮件组件来发送邮件。