# 第6章 实现留言功能



一个完整的博客怎么能缺少留言功能呢，当然我们可以用第三方社会化评论插件，既然我们有了数据库，我们不妨把留言保存到自己的数据库里。

我们设定：只有在文章页面才会显示留言板。

打开 post.js ，修改 `Post.prototype.save` 中要存入的文档为：

```
var post = {
    name: this.name,
    time: time,
    title:this.title,
    post: this.post,
    comments: []
};
```

我们在文档里增加了 comments 键（数组），用来存储此文章上的留言（一个个对象）。为了也让留言支持 markdown 语法，我们将 `Post.getOne` 函数里的 `doc.post = markdown.toHTML(doc.post);` 修改为：

```
if(doc){
  doc.post = markdown.toHTML(doc.post);
  doc.comments.forEach(function (comment) {
    comment.content = markdown.toHTML(comment.content);
  });
}
```

接下来我们在 models 下新建 comment.js 模型文件，添加如下代码：

```
var mongodb = require('./db');

function Comment(name, day, title, comment) {
  this.name = name;
  this.day = day;
  this.title = title;
  this.comment = comment;
}

module.exports = Comment;

//存储一条留言信息
Comment.prototype.save = function(callback) {
  var name = this.name,
      day = this.day,
      title = this.title,
      comment = this.comment;
  //打开数据库
  mongodb.open(function (err, db) {
    if (err) {
      return callback(err);
    }
    //读取 posts 集合
    db.collection('posts', function (err, collection) {
      if (err) {
        mongodb.close();
        return callback(err);
      }
      //通过用户名、时间及标题查找文档，并把一条留言对象添加到该文档的 comments 数组里
      collection.update({
        "name": name,
        "time.day": day,
        "title": title
      }, {
```

```
        $push: {"comments": comment}
      } , function (err, result) {
          mongodb.close();
          callback(null);
      });
    });
  });
};
```

修改 index.js ，在 `Post = require('../models/post.js')` 后添加一行代码：

```
Comment = require('../models/comment.js');
```

这里我们创建了 comment 的模型文件，用于存储新的留言到数据库，并在 index.js 中引入以作后用。

接下来我们创建 comment 的视图文件，在 views 文件夹下新建 comment.ejs ，添加如下代码：

```
<br />
<% post.comments.forEach(function (comment, index) { %>
  <p><a href="<%= comment.website %>"><%= comment.name %></a>
  <span class="info"> 回复于 <%= comment.time %></span></p>
  <p><%- comment.content %></p>
<% }) %>

<form method="post">
<% if (user) { %>
  姓名：<input type="text" name="name" value="<%= user.name %>" /><br />
  邮箱：<input type="text" name="email" value="<%= user.email %>" /><br />
  网址：<input type="text" name="website" value="/u/<%= user.name %>" /><br />
<% } else { %>
  姓名：<input type="text" name="name" /><br />
  邮箱：<input type="text" name="email" /><br />
  网址：<input type="text" name="website" value="http://" /><br />
<% } %>
  <textarea name="content" rows="5" cols="80"></textarea><br />
  <input type="submit" value="留言" />
</form>
```

注意：这里根据用户登录状态的不同，让 网址 一项显示不同的提示信息。还需注意的一点是，未登录的用户在留言的时候， 网址 这一项需要加上 `http://` 前缀，否则在生成连接的时候会基于当前 url （本地是 localhost:3000）。

打开 article.ejs ，在 `<%- include footer %>` 前添加一行代码：

```
<%- include comment %>
```

这样我们就在文章页面引入了留言模块。

最后，修改 index.js ，注册留言的 POST 响应，在 `app.get('/u/:name/:day/:title')` 后添加如下代码：

```
app.post('/u/:name/:day/:title', function(req,res){
  var date = new Date(),
      time = date.getFullYear() + "-" + (date.getMonth()+1) + "-" + date.getDate() + " " + date.getHours() + ":" + date.getMinutes
  var comment = {
      name: req.body.name,
      email: req.body.email,
      website: req.body.website,
      time: time,
      content: req.body.content
  };
  var newComment = new Comment(req.params.name, req.params.day, req.params.title, comment);
  newComment.save(function (err) {
    if (err) {
      req.flash('error', err);
      return res.redirect('/');
    }
    req.flash('success', '留言成功!');
    res.redirect('back');
  });
});
```

注意：这里我们使用 `res.redirect('back');` ，即留言成功后返回到该文章页。

现在，我们就给博客添加了留言的功能。

Last edited by nswbmw, a month ago