这是一个基于SSM（Spring + SpringMVC + MyBatis）框架的博客项目。以下是对项目结构的简要说明：

controller目录：

admin目录： 包含管理后台相关的控制器。

home目录： 包含前台（博客首页等）相关的控制器。

dto目录： 存放数据传输对象（DTO）。

entity目录： 存放实体类。

enums目录： 存放枚举类型。

interceptor目录： 存放拦截器。

mapper目录： 存放MyBatis的Mapper接口。

service目录： 存放服务接口及其实现类。

impl目录： 存放服务接口的实现类。

util目录： 存放工具类。

resources目录：

mapper目录： 存放MyBatis的XML映射文件。

mybatis目录： 存放MyBatis的配置文件。

spring目录： 存放Spring框架的配置文件。

webapp目录：

resource目录：

assets目录： 存放前端资源文件，如CSS、JS等。

WEB-INF目录：

view目录： 存放JSP视图文件，按照功能划分为Admin和Home两个部分。

test目录：

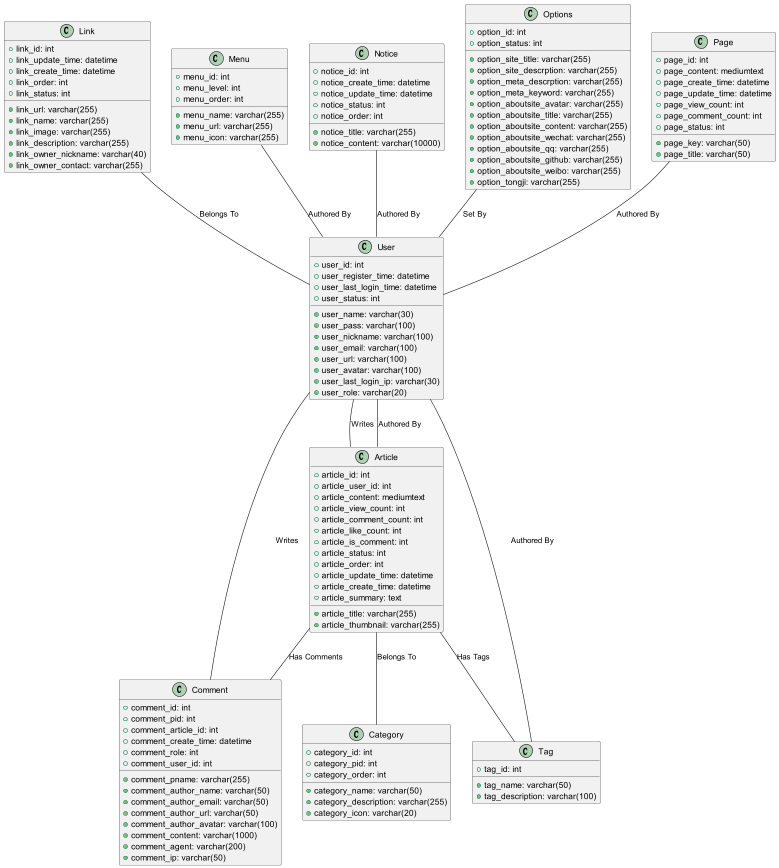
java目录：

com/Mingtc/ssm/blog目录：

mapper目录： 存放测试用的Mapper接口。

service目录： 存放测试用的服务接口。

项目的主要模块包括管理后台（Admin）和前台（Home），每个模块下又根据功能 划分了不同的包，包括文章、分类、评论、链接、菜单、公告、选项、页面、 标签、用户等。前端资源主要存放在webapp/resource/assets目录下，包括CSS、 JS等文件。 MyBatis 的映射文件存放在resources/mapper目录下。



用户实体（User Entity）：

Writes（写入）：表示用户与文章（Article表）和评论（Comment表）之间的关系。这意味着User表中的user\_id是Article表和Comment表的外键，将它们与创建它们的用户关联起来。

文章实体（Article Entity）：

Belongs To（属于）：表示Article表与Category表之间的关系。它表示一篇文章属于某个特定的分类。Article表中的category\_id是一个外键，引用了Category表的主键（category\_id）。

Has Tags（拥有标签）：表示Article表与Tag表之间的多对多关系。它暗示了存在一个名为article\_tag\_ref的连接表（在图中未显式显示），用于使用它们的相应ID将文章与标签关联起来。

Has Comments（拥有评论）：表示Article表与Comment表之间的关系。它表示一篇文章可以有多个相关的评论。Comment表中的comment\_article\_id是一个外键，引用了Article表中的article\_id。

Authored By（由...撰写）：表示文章由用户撰写。Article表中的article\_user\_id是一个外键，引用了User表中的user\_id。

链接实体（Link Entity）：

Belongs To（属于）：表示链接与用户之间的关系。Link表中的user\_id是一个外键，引用了User表中的user\_id。

菜单实体（Menu Entity）：

Authored By（由...撰写）：表示Menu表与User表之间的关系。它表示一个菜单由用户撰写。Menu表中的user\_id是一个外键，引用了User表中的user\_id。

公告实体（Notice Entity）：

Authored By（由...撰写）：表示一个公告由用户撰写。Notice表中的user\_id是一个外键，引用了User表中的user\_id。

选项实体（Options Entity）：

Set By（由...设置）：表示Options表与User表之间的关系。它表示站点选项由用户设置。Options表中的user\_id是一个外键，引用了User表中的user\_id。

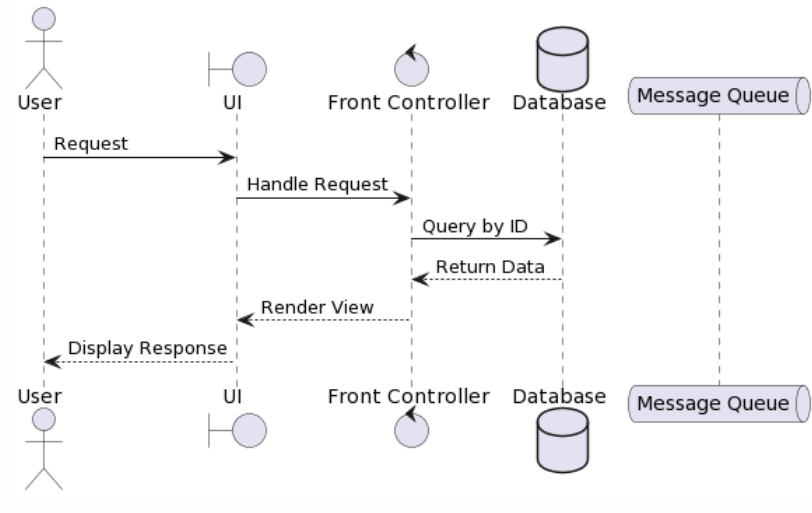
页面实体（Page Entity）：

Authored By（由...撰写）：表示一个自定义页面由用户撰写。Page表中的user\_id是一个外键，引用了User表中的user\_id。

评论实体（Comment Entity）：

Authored By（由...撰写）：表示一个评论由用户撰写。Comment表中的comment\_user\_id是一个外键，引用了User表中的user\_id。

|  |
| --- |
| article表：  作用：存储文章信息。  字段解释：  article\_id: 文章ID（主键）。  article\_user\_id: 用户ID，表示文章的作者。  article\_title: 文章标题。  article\_content: 文章内容。  article\_view\_count: 文章访问量。  article\_comment\_count: 文章评论数。  article\_like\_count: 文章点赞数。  article\_is\_comment: 是否允许评论。  article\_status: 文章状态。  article\_order: 文章排序值。  article\_update\_time: 文章更新时间。  article\_create\_time: 文章创建时间。  article\_summary: 文章摘要。  article\_thumbnail: 文章缩略图。  article\_category\_ref表：  作用：文章与分类的关联表。  字段解释：  article\_id: 文章ID（外键）。  category\_id: 分类ID（外键）。  article\_tag\_ref表：  作用：文章与标签的关联表。  字段解释：  article\_id: 文章ID（外键）。  tag\_id: 标签ID（外键）。  category表：  作用：存储文章分类信息。  字段解释：  category\_id: 分类ID（主键）。  category\_pid: 分类父ID，表示分类的层次关系。  category\_name: 分类名称。  category\_description: 分类描述。  category\_order: 分类排序值。  category\_icon: 分类图标。  comment表：  作用：存储评论信息。  字段解释：  comment\_id: 评论ID（主键）。  comment\_pid: 上级评论ID。  comment\_pname: 上级评论名称。  comment\_article\_id: 文章ID（外键）。  comment\_author\_name: 评论人名称。  comment\_author\_email: 评论人邮箱。  comment\_author\_url: 评论人个人主页。  comment\_author\_avatar: 评论人头像。  comment\_content: 评论内容。  comment\_agent: 浏览器信息。  comment\_ip: IP地址。  comment\_create\_time: 评论时间。  comment\_role: 角色，是否管理员。  comment\_user\_id: 评论用户ID。  link表：  作用：存储友情链接信息。  字段解释：  link\_id: 链接ID（主键）。  link\_url: 链接URL。  link\_name: 链接名称。  link\_image: 链接头像。  link\_description: 链接描述。  link\_owner\_nickname: 所属人名称。  link\_owner\_contact: 联系方式。  link\_update\_time: 更新时间。  link\_create\_time: 创建时间。  link\_order: 排序号。  link\_status: 链接状态。  menu表：  作用：存储菜单信息。  字段解释：  menu\_id: 菜单ID（主键）。  menu\_name: 菜单名称。  menu\_url: 菜单URL。  menu\_level: 菜单等级。  menu\_icon: 菜单图标。  menu\_order: 菜单排序值。  notice表：  作用：存储公告信息。  字段解释：  notice\_id: 公告ID（主键）。  notice\_title: 公告标题。  notice\_content: 公告内容。  notice\_create\_time: 创建时间。  notice\_update\_time: 更新时间。  notice\_status: 公告状态。  notice\_order: 排序值。  options表：  作用：存储系统配置选项。  字段解释：  option\_id: 配置选项ID（主键）。  其他字段表示各种配置选项，如站点标题、站点描述、统计信息等。  page表：  作用：存储自定义页面信息。  字段解释：  page\_id: 自定义页面ID（主键）。  page\_key: 页面标识。  page\_title: 页面标题。  page\_content: 页面内容。  page\_create\_time: 创建时间。  page\_update\_time: 更新时间。  page\_view\_count: 页面访问量。  page\_comment\_count: 页面评论数。  page\_status: 页面状态。  tag表：  作用：存储标签信息。  字段解释：  tag\_id: 标签ID（主键）。  tag\_name: 标签名称。  tag\_description: 标签描述。  user表：  作用：存储用户信息。  字段解释：  user\_id: 用户ID（主键）。  user\_name: 用户名。  user\_pass: 用户密码。  user\_nickname: 用户昵称。  user\_email: 用户邮箱。  user\_url: 用户个人主页。  user\_avatar: 用户头像。  user\_last\_login\_ip: 用户最后登录IP。  user\_register\_time: 用户注册时间。  user\_last\_login\_time: 用户最后登录时间。  user\_status: 用户状态。  user\_role: 用户角色。 |



项目逻辑十分简单，运行顺序如下：

1. 用户 (User) 发送请求到用户界面 (UI)。
2. 用户界面 (UI) 将请求传递给前端控制器 (Front Controller)。
3. 前端控制器 (Front Controller) 发送查询请求给数据库 (Database)。
4. 数据库 (Database) 执行查询操作并返回数据给前端控制器 (Front Controller)。
5. 前端控制器 (Front Controller) 将数据渲染到用户界面 (UI)。
6. 用户界面 (UI) 显示响应给用户 (User)。

该项目是一个使用 Spring 框架的传统三层架构(MVC 架构)，包括控制层（controller）、服务层（service）、持久层（mapper）。数据库访问采用了 MyBatis 框架。项目还包含拦截器用于处理请求，以及前端资源和视图用于构建用户界面。

controller 包下包含 admin 和 home 两个子包，分别表示后台管理和前台页面的控制器。

service 包用于存放服务层的代码，包含 impl 子包表示服务层的实现。

DAO 模式和持久层操作：

mapper 包用于存放 MyBatis 的映射器接口，实现了数据访问对象（DAO）模式。

resources/mapper 存放 MyBatis 的 XML 映射文件，用于配置 SQL 映射关系。

拦截器：

interceptor 包用于存放拦截器，这些拦截器可能用于处理请求前后的逻辑，例如登录验证、权限控制等。

DTO（数据传输对象）：

dto 包用于存放数据传输对象，它们可能用于在服务层和控制器层之间传递数据。

工具类和枚举：

util 包用于存放工具类，这些类可能包含一些通用的工具方法。

enums 包用于存放枚举类，定义了一些常量值。

前端资源和视图：

webapp 包下的 resource/assets 存放了前端资源，如 CSS、JS、图片等。

WEB-INF/view 存放了视图层的代码，按照模块划分，包括 Admin（后台管理页面）和 Home（前台页面）两个主要模块。

测试：

test 目录下存放了测试相关的代码，包括 mapper 和 service 两个子包，用于单元测试。