

产品源代码整体架构说明

本文档基于当前仓库代码结构整理，用于快速理解 LSERP-MES / WebErp 后端工程的模块边界、请求调用链、数据访问方式和主要扩展点。

1. 工程概览

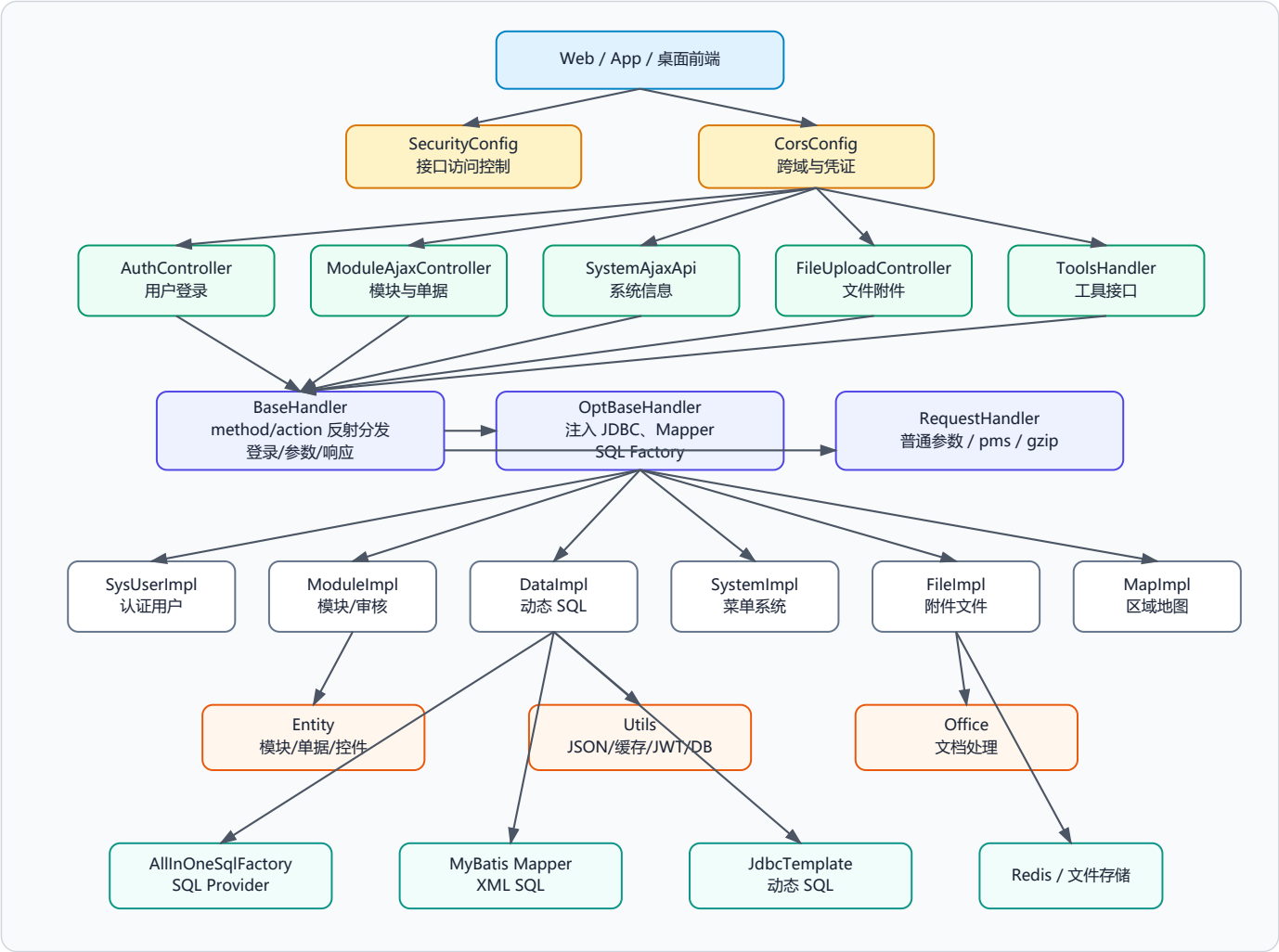
本项目是一个基于 Spring Boot 的 Java 后端服务，代码主体位于 `WebErp/weberp`。整体采用 Maven 父子模块结构，当前父工程 `WebErp` 下只有一个可执行子模块 `weberp`。

层级	路径	说明
仓库根目录	<code>.</code>	README、历史构建产物和项目根文件
Maven 父工程	<code>WebErp/pom.xml</code>	<code>packaging=pom</code> ，聚合 <code>weberp</code> 子模块，统一部分依赖版本
应用子模块	<code>WebErp/weberp</code>	Spring Boot 可执行 JAR 模块，主类为 <code>org.example.WebErpApplication</code>
主源码	<code>WebErp/weberp/src/main/java/org/example</code>	Controller、Handler、Impl、Entity、Utils 等后端源码
配置资源	<code>WebErp/weberp/src/main/resources</code>	<code>application.properties</code> 、MyBatis 配置、Mapper XML、语言包
测试代码	<code>WebErp/weberp/src/test/java/org/example</code>	单元/回归测试入口

关键技术栈：

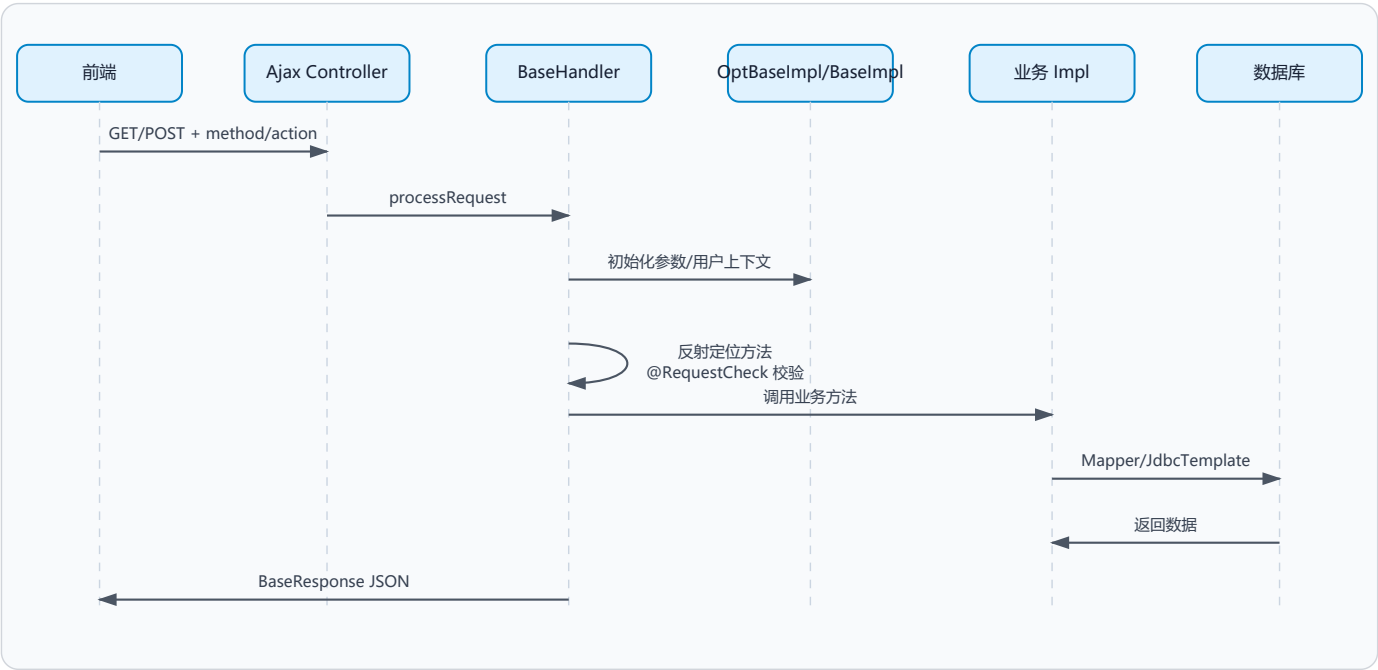
- Spring Boot Web 3.4.3：HTTP 服务和依赖注入。
- Spring Security 3.4.3：接口访问控制。
- MyBatis 3.5.17 / mybatis-spring-boot-starter 3.0.4：Mapper XML 数据访问。
- JdbcTemplate / NamedParameterJdbcTemplate：大量动态 SQL 和存储过程调用。
- HikariCP：数据库连接池。
- PageHelper：分页插件，当前方言配置为 `dm`。
- JWT：Token 生成、刷新和校验辅助。
- Redis：缓存或会话相关基础设施。
- Aspose / Spire / iText / JavaCV / ZXing：Office、PDF、音视频、二维码等文件处理能力。

2. 总体架构图



3. 请求处理架构

项目的 HTTP 入口不是标准的 “一个 URL 对应一个业务方法” 的 REST 风格，而是多个 Ajax 入口统一接收请求，再由 `BaseHandler` 根据请求参数中的 `method` 或 `action` 反射调用同名业务方法。



主要机制：

- `Controller` 只提供统一入口，例如 `/Api/ModuleAjaxApi/`、`/Api/SystemAjaxApi/`、`/Api/SysUserAjaxApi/**`。
- `BaseHandler#getMethod()` 从 `method` 或 `action` 参数获取业务方法名。
- `BaseHandler#processRequest()` 负责参数初始化、登录检查、`@RequestCheck` 校验、缓存判断、反射调用、日志记录、Token 刷新和响应输出。
- `RequestHandler` 兼容普通 `query/form` 参数、`pms` 加密参数、`gzip/deflate` 压缩请求体。
- `OptBaseHandler` 在 `BaseHandler` 基础上注入 `JdbcTemplate`、`CRMapper`、`DMCrmMapper` 和 `AllInOneSqlFactory`，并创建 `OptBaseImpl` 作为业务上下文。

4. 源码包职责

包	主要职责
<code>org.example</code>	Spring Boot 启动入口 <code>WebErpApplication</code>
<code>Api</code>	公共请求处理、统一响应、登录状态、日志、单用户控制
<code>Auth</code>	登录、验证码、密码、Token、安全辅助
<code>Config</code>	CORS 配置
<code>SystemApi</code>	系统信息、系统菜单、账套/系统版本等接口
<code>ModuleApi</code>	核心模块 Ajax 接口、模块 DTO、MyBatis Mapper
<code>FileUploadApi</code>	文件上传、下载、附件权限与附件记录维护
<code>Impl</code>	主要业务实现层，包含模块、数据、用户、系统、文件、地图、短信、更新等实现
<code>Impl.Sql</code>	多数据库 SQL Provider 和工厂，按数据库类型生成 SQL 或调用对应 Mapper
<code>Service</code>	业务接口定义，例如 <code>ModuleImplService</code> 、 <code>AuthService</code> 、 <code>IModuleEvent</code>
<code>Entity</code>	响应对象、模块配置对象、单据对象、控件对象、审核对象、异常对象等领域模型
<code>Enums</code>	系统枚举、参数方向、任务类型、登录状态等
<code>Utils</code>	通用工具，包括配置读取、JSON、缓存、JWT、文件、压缩、加密、数据库操作等
<code>Office</code>	Office/Word 文档生成与转换相关工具
<code>PageBreaksApi</code>	分页符相关 Mapper

5. 核心业务模块

5.1 认证与用户

入口：`AuthController`，路径为 `/Api/SysUserAjaxApi`。

核心实现：

- `SysUserImpl implements AuthService`：登录、手机号登录、切换账套、重置密码、验证码、登录状态维护。
- `JwtHelp` / `JwtUtils`：Token 创建、刷新、缓存和解析辅助。
- `SingleUserHandler`：单用户登录控制和会话互斥。
- `SafetyUtil`：安全辅助逻辑。

典型方法：

- `Login`
- `CheckLogin`
- `LoginOut`
- `ChangeServer`
- `ResetPwd`
- `GenerateCaptcha`
- `SendPhoneCode`

5.2 系统基础信息

入口：`SystemAjaxApi`，路径为 `/Api/SystemAjaxApi`。

核心实现：

- `SystemImpl`：系统列表、系统信息、登录信息、菜单、数据库服务器、Web 更新信息等。

典型方法：

- `GetSystems`
- `GetSystemInfo`
- `GetSysMenus`
- `GetSystemLoginInfo`
- `GetProSysType`
- `GetDbServer`
- `GetWebUpdateInfo`

5.3 动态模块与单据

入口：`ModuleAjaxController`，路径为 `/Api/ModuleAjaxApi`。

这是项目最核心的业务入口，围绕“模块配置驱动”的 ERP/MES 页面和单据能力展开。前端传入 `ModuleId`、`MenuId`、`method` 等参数后，后端从系统配置表和业务表中组装模块元数据、字段、列表、明细、审核、附件、右键菜单、桌面数据等。

核心实现：

- `ModuleImpl implements ModuleImplService`：模块初始化、字段数据、模块数据、单据保存、审核、附件、权限、桌面配置。
- `DataImpl`：动态 SQL、表结构、字段、主键、存储过程、附件权限、模块配置数据读取。
- `ModuleEventImpl`：模块数据加载、保存、状态变化、附件上传等事件扩展点。
- `MapImpl`：区域和地图相关接口实现。

典型方法：

- `GetModuleIniParams`
- `GetModuleData`
- `GetFieldData`
- `GetBillIniParams`
- `AddOrUpd`
- `Delete`
- `SaveBill`
- `GetModuleCfg`
- `GetModuleDetailsData`
- `GetAuditHistory`
- `GetRoles` / `SaveRoles`
- `GetBSDesktopData`

5.4 文件与附件

入口：`FileUploadController`，路径为 `/Api/FileUploadApi`。

核心实现：

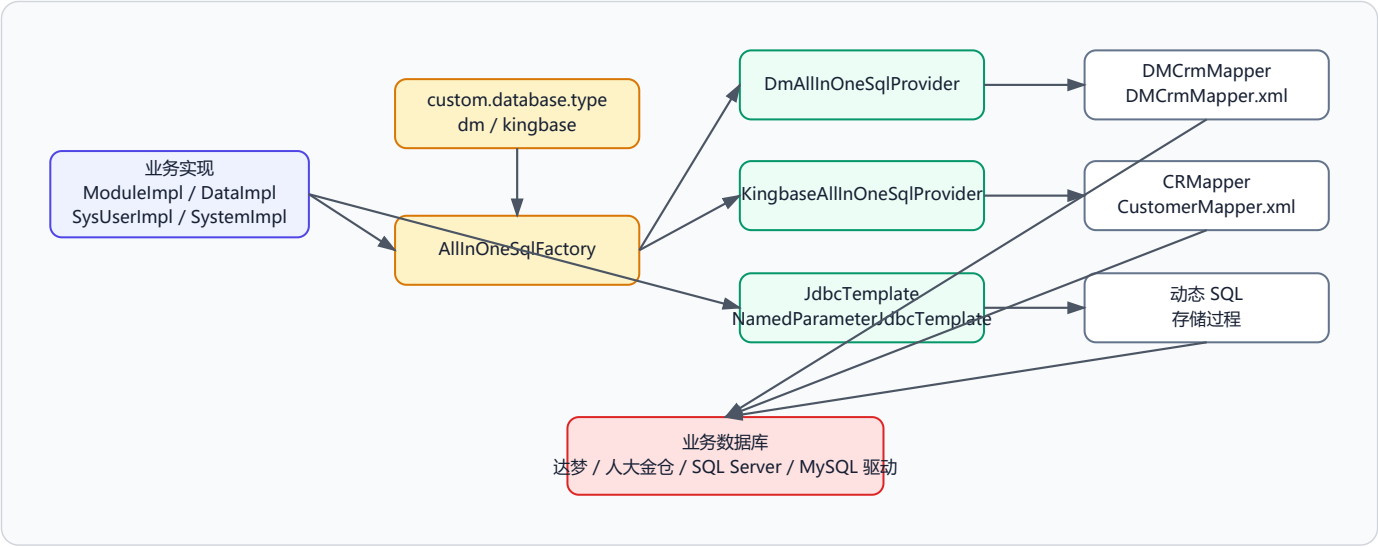
- `FileImpl`：文件保存、校验、附件基础记录。
- `FileUtil` / `PathUtil` / `ZipUtil`：路径处理、实际文件保存/删除、批量压缩。
- `ModuleImpl` / `DataImpl`：附件权限检查、附件记录入库、模块关联。
- `OfficeUtil` / `CreateWordUtil`：文档处理、预览或转换相关能力。

典型方法：

- `DoWebUpload`
- `DownloadFiels`
- `DoDelete`
- `SaveViewModule`

6. 数据访问架构

项目同时使用 MyBatis Mapper XML 和 Spring `JdbcTemplate`。



数据访问特点:

- `application.properties` 通过 `custom.database.type` 指定当前数据库类型，当前配置为 `dm`。
- `AllInOneSqlFactory#createProvider()` 根据数据库类型创建 `DmAllInOneSqlProvider` 或 `KingbaseAllInOneSqlProvider`。
- 达梦数据库主要走 `DMCrmMapper` 和 `DMCrmMapper.xml`。
- 人大金仓或通用查询主要走 `CRMapper` 和 `CustomerMapper.xml`。
- 大量复杂业务仍直接使用 `JdbcTemplate` 或 `NamedParameterJdbcTemplate` 拼装动态 SQL。
- `mybatis-config.xml` 配置了 `PageHelper` 插件，方言为 `dm`。

7. 配置与运行边界

主要配置文件: `WebErp/weberp/src/main/resources/application.properties`。

关键配置项:

- `server.port=8088` : 服务端口。
- `custom.database.type=dm` : 当前 SQL 方言/Provider 选择。
- `spring.datasource.*` : 数据库连接、驱动和 Hikari 连接池配置。
- `mybatis.mapper-locations=classpath:mapper/*.xml` : Mapper XML 位置。
- `pagehelper.helper-dialect=dm` : 分页方言。
- `spring.data.redis.*` : Redis 连接配置。
- `language=Language_CN` : 语言包选择。
- `jwt.expiration`、`SingleUser`、`RestInitPwd`、`LockErrPwd` 等: 登录和安全相关开关。

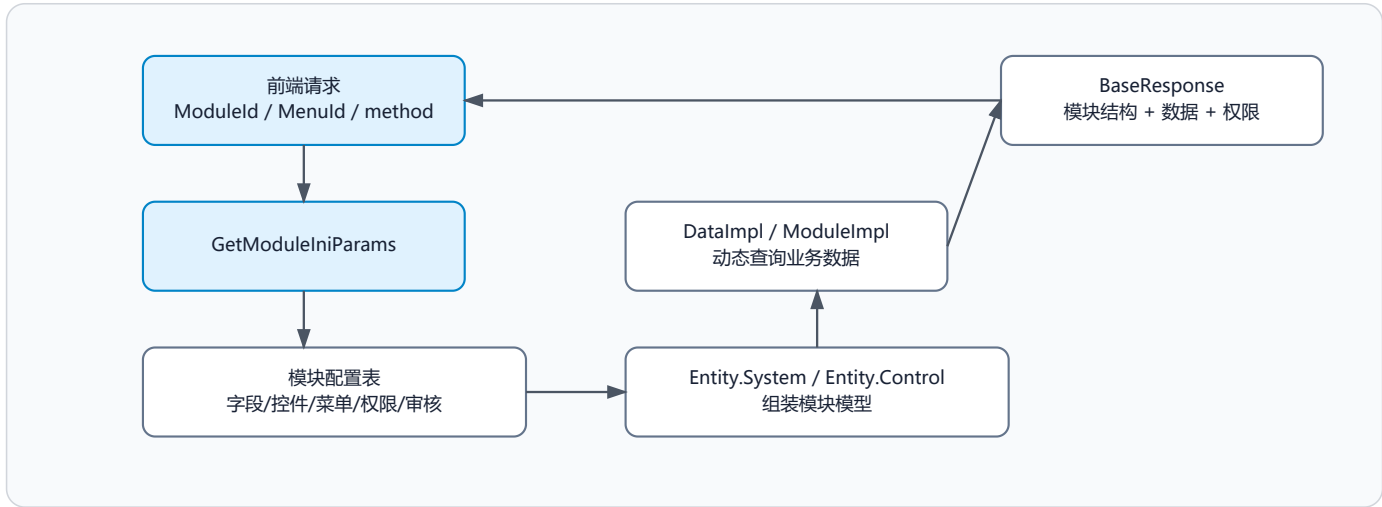
安全边界:

- `SecurityConfig` 当前只放行指定 `/Api/*` 入口，其余请求默认拒绝。
- `CorsConfig` 明确列出允许的前端来源，并允许携带凭证。
- Controller 入口放行后，业务级登录校验主要由 `BaseHandler` 和 `@RequestCheck` 决定。

注意: 配置文件中包含数据库、Redis 等环境信息，生产环境建议通过环境变量、外部配置中心或部署平台密钥管理注入，避免将敏感连接信息写入源码仓库。

8. 模块配置驱动模型

从代码结构看，系统大量业务并非通过固定 Java DTO 和固定 SQL 完成，而是依赖数据库中的模块配置表、字段配置、控件配置、菜单配置和权限配置动态组装页面与数据。



这种架构的影响：

- 新增业务模块时，可能更多依赖数据库配置和元数据，而不是新增独立 Controller。
- `ModuleId` 是贯穿模块初始化、数据查询、附件、审核、权限的核心参数。
- `Entity.Control` 下的 `Field`、`GridPanel`、`TreePanel`、`ComboBox` 等对象承担前端控件描述能力。
- `BaseModule`、`BillModule`、`ModuleBaseEntity` 等模型承载基础模块和单据模块配置。

9. 扩展点

常见扩展方式：

1. 新增 Ajax 方法：在对应 Controller 中新增 public 无参方法，通过前端 `method` 或 `action` 调用，并按需添加 `@RequestCheck`。
2. 新增模块能力：优先检查是否能通过模块配置表、字段配置、菜单配置实现，再考虑修改 `ModuleImpl` 或 `DataImpl`。
3. 新增数据库方言：实现 `AllInOneSqlProvider`，并在 `AllInOneSqlFactory` 中按新的 `custom.database.type` 分支返回 Provider。
4. 新增 Mapper 查询：在 Mapper 接口中定义方法，并在 `resources/mapper/*.xml` 中补充对应 SQL。
5. 新增文件处理能力：扩展 `FileUploadController`、`FileImpl`、`FileUtil` 或 `Office` 工具类。
6. 新增业务事件：通过 `ModuleEventImpl` 的加载、保存、状态变化、附件上传等事件机制接入。

10. 维护建议

- 优先理解 `BaseHandler -> OptBaseHandler -> OptBaseImpl/BaseImpl` 这条公共链路，再看具体业务方法。
- 排查接口问题时，先确认请求路径、`method/action`、`ModuleId/MenuId`、`@RequestCheck` 参数校验和登录状态。
- 修改数据访问逻辑时，先确认当前数据库类型和实际使用的是 Mapper XML、SQL Provider 还是 JdbcTemplate 动态 SQL。

- 对涉及附件、审核、权限、单据保存的改动，应同步验证 `ModuleImpl`、`DataImpl` 和数据库配置表的联动。
- 生产配置建议外置，尤其是数据库地址、用户名、密码、Redis 地址、JWT 相关配置。