核心功能：

- 用户管理（学生、老师、管理员多角色）
- 课程管理（视频上传、章节管理）
- 订单支付
- 直播
- 作业提交与批改

第一周：项目架构设计与基础搭建

学习目标

- 掌握Spring Boot项目快速搭建方法
- 理解分层架构设计思想
- 学会规范的数据库建模方法
- 建立统一的代码规范意识

课程内容

理论讲解

- 在线教育平台架构分析与技术选型
- Spring Boot生态系统介绍
- 数据库设计原则与最佳实践
- RESTful API设计规范

实践操作

1. Spring Boot项目初始化

环境要求：

```
- JDK 17+
- Maven （生成项目）
- MySQL （存储关系数据：用户信息、老师信息、课程信息等）
- Redi  （缓存：订单支付的临时信息）
- Git
- Postman
```

- 使用Spring Initializr创建项目
  - Dependencies：
    - Spring Web：支持构建Web应用
    - Spring Data JPA：操作数据库的框架
    - MySQL Driver：支持MySQL数据库连接
    - Redis：支持Redis缓存
- 集成MySQL、Redis、Spring Security依赖
- 配置application.yml多环境配置

2. 数据库设计与建模

- 设计用户表（users）、角色表（roles）、课程表（courses）
- 建立实体关系（用户-角色多对多，课程-用户多对多）
- 使用JPA注解创建实体类

3. 统一返回格式设计

- 设计Result通用返回类
- 实现ResultCode枚举定义状态码
- 创建ResponseUtils工具类

4. 全局异常处理机制

- 实现@ControllerAdvice全局异常处理器
- 定义业务异常类BusinessException

项目架构设计

分层架构说明

```
Controller: 接收请求，参数校验，调用Service
Service: 业务逻辑处理，事务控制（数据库调用、作业提交）
Repository: 数据访问，与数据库交互
Entity: 数据实体，映射数据库表
```

目录结构设计

```
src/main/hava/com/edu/platform/
- EduPlatformApplication.java       # 启动层
- config/
  - RedisConfig.java
  - WebConfig.java
- controller/                       # 控制层
  - UserController.java
  - CourseController.java
  - OrderController.java
- service/                          # 服务层
  - UserService.java
  - CourseService.java
  - OrderService.java
- repository/                       # 数据访问层
  - UserRepository.java
  - CourseRepository.java
  - OrderRepository.java
```

```
- entiy
  - User.java
  - Course.java
  - Order.java
- dto/                                  # 数据传输对象
  - request/
  - response/
- common/                               # 公共类
  - result/
  - exception/
  - constants/
  - utils/
```

数据库设计与建模

核心表设计

```
-- 用户表
CREATE TABLE edu_user (
  id BIGINT AUTO_INCREMENT PRIMARY KEY COMMENT '用户id',
  username VARCHAR(50) NOT NULL UNIQUE COMMENT '用户名',
  password ****,
  email ****,
  phone ****,
  nickname ****,
  role ENUM(*****) DEFAULT ****,
  created_time ****,
  updated_time ****
)

-- 课程表
CREATE TABLE edu_course (
  id ****,
```

```
  title ****,
  description ****,
  time ****,
  price ****,
  teacher_id ****,
  categori_id ****,
  view_count ****,
  student_count ****,
  created_time ****,
  updated_time ****,
  INDEX idx_teacher_id (teacher_id),
  INDEX idx_category_id (category_id)
)

-- 订单表
CREATE TABLE edu_order (
  id ****,
  order_no ****,
  user_id ****,
  course_id ****,
  amount ****,
  pay_time ****,
  created_time ****,
  updated_time ****,
  INDEX idx_user_id (user_id),
  INDEX idx_course_id (course_id)
  INDEX idx_order_no (order_no)
)
```

实体类设计

```
@Entity
@Table(name = 'edu_user')
```

```java
@Data
@NoArgsConstructor
@AllArgsConstructor
public class User {
  @Id
  @GeneratedValue(strategy = GenerationType.IDENTITY)
  private Long id;

  @Colum(unique = true, nullable = false, length = 50)
  private String username;

  @Column(***)
  private String password;

  @Column(***)
  private String email;

  @Column(***)
  private String phone;

  @Column(***)
  private String nickname;

  @Column(***)
  private UserRole role = *****;

  @CreationTimestamp
  @Column(***)
  private LocalDateTime createdTime;

  @UpdateTimestamp
  @Column(***)
  private LocalDateTime updatedTime;
```

```
}

public enum UserRole {
  STUDNET, TEACHER, ADMIN
}
```

统一返回格式设计

统一响应结果类

```
@Data
@NoArgsConstructor
@AllArgsConstructor
public class Result<T> {
  private Integer code;
  private String message;
  private T data;
  private Long timestamp;

  public static <T> Result<T> success() {
    return success(null);
  }

  public static <T> Result<T> success(T data) {
    Result<T> result = new Result<>();
    result.setCode(200);
    result.setMessage("操作成功");
    result.setData(data);
    result.setTimestamp(System.currentTimeMillis());
    return result;
  }

  public static <T> Result<T> error(String message) {
```

```
    }

}

public enum ResultCode {
    SUCCESS(200, "操作成功"),
    PARAM_ERROR(400, "参数错误"),
    未授权***,
    禁止访问 ****,


    // 业务错误
    USER_NOT_FOUND(1001, "用户名不存在");
    PASSWORD_ERROR(1002, "密码错误");
    USER_DISABLED(1003, "用户已被禁用";
                  )
}
```

https://github.com/macrozheng/mall-learning

https://github.com/201206030/novel-plus

https://github.com/elunez/eladmin

https://docs.spring.io/spring-boot/documentation.html

课后作业

作业1：项目初始化

1. 使用Spring Initializr创建项目
2. 配置数据库连接和Redis连接
3. 创建标准的目录结构
4. 提交代码到GitHub

作业2：数据库设计

1. 根据课程设计创建数据库表
2. 编写对应的实体类
3. 创建Repository接口
4. 测试数据库连接

作业3：基础框架搭建

1. 实现统一返回格式
2. 添加全局异常处理
3. 编写一个简单的Hello World接口
4. 测试异常处理机制