苍穹外卖day12

1. 工作台

1. 需求分析与接口设计

工作台是系统运营的数据看板，并提供快捷操作入口，可以有效提高商家的工作效率。

工作台展示的数据：

- 今日数据
- 订单管理
- 菜品总览
- 套餐总览
- 订单信息

接口设计：

今日数据接口

订单管理接口

菜品总览接口

套餐总览接口

订单搜索（已完成）

各个状态的订单数量统计（已完成）

今日数据的接口设计：

基本信息

Path： /admin/workspace/businessData

Method： GET

接口描述：

请求参数

返回数据

名称	类型	是否必须	默认值	备注	其他信息
code	integer	必须			format: int32
data	object	必须			
├─ newUsers	integer	必须		新增用户数	format: int32
├─ orderCompletionRate	number	必须		订单完成率	format: double
├─ turnover	number	必须		营业额	format: double
├─ unitPrice	number	必须		平均客单价	format: double
├─ validOrderCount	integer	必须		有效订单数	format: int32
msg	string	非必须			

需求分析和设计

订单管理的接口设计：

基本信息

Path： /admin/workspace/overviewOrders

Method： GET

接口描述：

请求参数

返回数据

名称	类型	是否必须	默认值	备注	其他信息
code	integer	必须			format: int32
data	object	必须			
├─ allOrders	integer	必须		全部订单	format: int32
├─ cancelledOrders	integer	必须		已取消数量	format: int32
├─ completedOrders	integer	必须		已完成数量	format: int32
├─ deliveredOrders	integer	必须		待派送数量	format: int32
├─ waitingOrders	integer	必须		待接单数量	format: int32
msg	string	非必须			

菜品总览的接口设计：

基本信息

Path: /admin/workspace/overviewDishes

Method: GET

接口描述:

请求参数

返回数据

名称	类型	是否必须	默认值	备注	其他信息
code	integer	必须			format: int32
data	object	必须			
├─ discontinued	integer	必须		已停售菜品数量	format: int32
├─ sold	integer	必须		已启售菜品数量	format: int32
msg	string	非必须			

套餐总览的接口设计：

基本信息

Path: /admin/workspace/overviewSetmeals

Method: GET

接口描述:

请求参数

返回数据

名称	类型	是否必须	默认值	备注	其他信息
code	integer	必须			format: int32
data	object	必须			
├─ discontinued	integer	必须		已停售套餐数量	format: int32
├─ sold	integer	必须		已启售套餐数量	format: int32
msg	string	非必须			

2. 代码导入

C:\baidunetdiskdownload\资料\day12\工作台模块功能代码

sky-server/src/main/java/com/sky/controller/admin/WorkSpaceController.java

sky-server/src/main/java/com/sky/service/WorkspaceService.java

sky-server/src/main/java/com/sky/service/impl/WorkspaceServiceImpl.java

DishMapper.java

```java
    /**
     * 根据条件统计菜品数量
     * @param map
     * @return
     */
    Integer countByMap(Map map);
```

DishMapper.xml

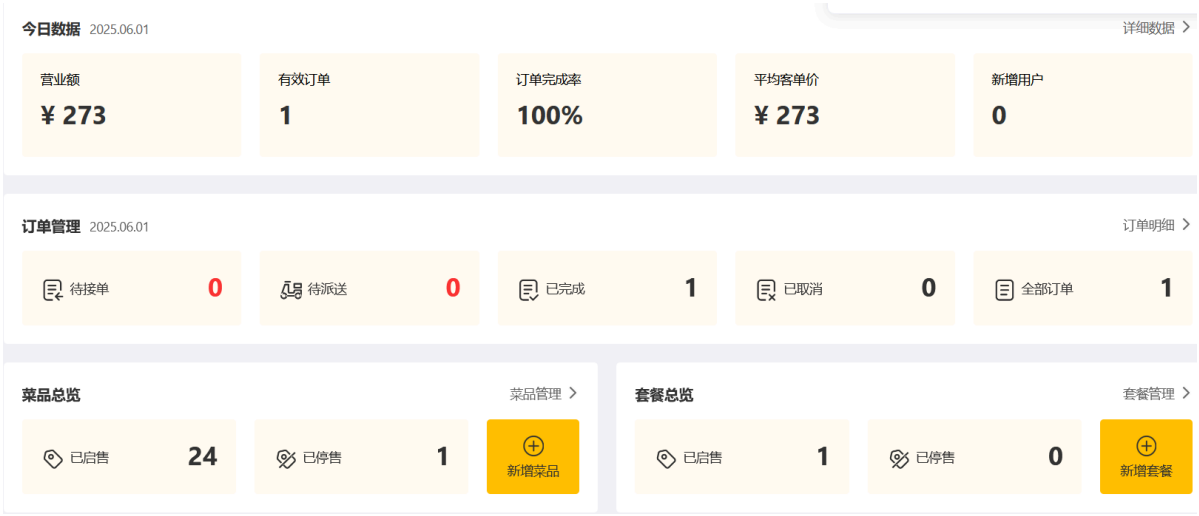
```xml
    <select id="countByMap" resultType="java.lang.Integer">
        select count(id) from dish
        <where>
            <if test="status != null">
                and status = #{status}
            </if>
            <if test="categoryId != null">
                and category_id = #{categoryId}
            </if>
        </where>
    </select>
```

SetmealMapper.java

```java
    /**
     * 根据条件统计套餐数量
     * @param map
     * @return
     */
    Integer countByMap(Map map);
```

SetmealMapper.xml

```xml
    <select id="countByMap" resultType="java.lang.Integer">
        select count(id) from setmeal
        <where>
            <if test="status != null">
                and status = #{status}
            </if>
            <if test="categoryId != null">
                and category_id = #{categoryId}
            </if>
        </where>
    </select>
```



3. Apache POI

Apache POI 是一个处理Miscrosoft Office各种文件格式的开源项目。简单来说就是，我们可以使用 POI 在 Java 程序中对Miscrosoft Office各种文件进行读写操作。

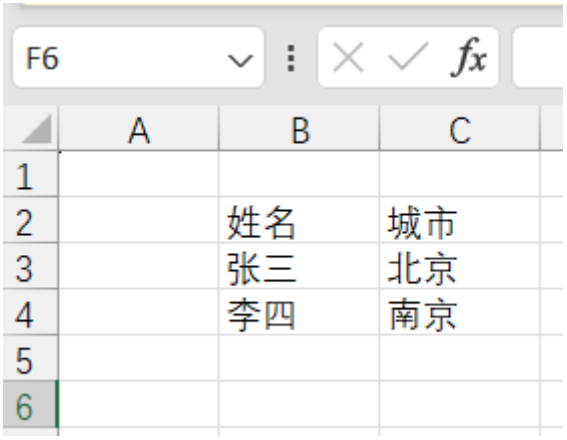
一般情况下，POI 都是用于操作 Excel 文件。

sky-server/src/main/java/com/sky/test/POITest.java

```java
package com.sky.test;

import org.apache.poi.xssf.usermodel.XSSFCell;
import org.apache.poi.xssf.usermodel.XSSFRow;
import org.apache.poi.xssf.usermodel.XSSFSheet;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
import org.apache.xmlbeans.impl.xb.xmlconfig.Extensionconfig;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.function.BiFunction;
import java.util.function.Function;

/**
 * 使用POI操作excel文件
 */
public class POITest {

    /**
     * 通过POI创建excel文件写入文件内容
     */
    public static void write() throws Exception {
        //在内存中创建excel文件
        XSSFWorkbook excel = new XSSFWorkbook();
        //在excel文件中创建sheet页
```

```
        XSSFSheet sheet = excel.createSheet("info");
        //在sheet中创建行对象，rownum从0开始
        XSSFRow row = sheet.createRow(1);
        //创建单元格并写入文件内容
        row.createCell(1).setCellValue("姓名");
        row.createCell(2).setCellValue("城市");

        //创建一个新行
        row=sheet.createRow(2);
        row.createCell(1).setCellValue("张三");
        row.createCell(2).setCellValue("北京");

        row=sheet.createRow(3);
        row.createCell(1).setCellValue("李四");
        row.createCell(2).setCellValue("南京");

        //通过输出流将内存中的excel文件写入到磁盘
        FileOutputStream fileOutputStream = new FileOutputStream(new
File("C:\\baidunetdiskdownload\\资料\\info.xlsx"));
        excel.write(fileOutputStream);

        //关闭资源
        fileOutputStream.close();
        excel.close();
    }

    public static void read() throws Exception {
        //读取磁盘已经存在的excel文件
        FileInputStream fileInputStream = new FileInputStream(new
File("C:\\baidunetdiskdownload\\资料\\info.xlsx"));
        XSSFWorkbook excel = new XSSFWorkbook(fileInputStream);
        //读取第一个sheet页
        XSSFSheet sheet = excel.getSheetAt(0);
        //获取sheet中最后一行的行号
        int lastRowNum = sheet.getLastRowNum();
        for (int i = 0; i <= lastRowNum; i++) {
            //获得某一行
            XSSFRow row = sheet.getRow(i);
            if (row != null) {//避免空行
                //获得单元格对象
                BiFunction<XSSFRow,Integer,String> func=(XSSFRow xssfRow,
Integer cellIndex)->{
                    XSSFCell cell = xssfRow.getCell(cellIndex);
                    return cell != null? cell.getStringCellValue(): "";
                };//避免空单元格
                String cellValue1 = func.apply(row,1);
                String cellValue2 = func.apply(row,2);
                System.out.println(cellValue1+" "+cellValue2);
            }
        }

        //关闭资源
        fileInputStream.close();
        excel.close();
    }

```

```
81      public static void main(String[] args) throws Exception{
82          //write();
83          read();
84      }
85  }
```



2. 导出运营数据Excel报表

导出的Excel报表格式：

运营数据报表					
概览数据					
营业额		订单完成率		新增用户数	
有效订单		平均客单价			
明细数据					
日期	营业额	有效订单	订单完成率	平均客单价	新增用户数

业务规则：

- 导出Excel形式的报表文件
- 导出最近30天的运营数据

接口设计：

基本信息

Path： /admin/report/export

Method： GET

接口描述：

请求参数

返回数据

注意：当前接口没有返回数据，因为报表导出功能本质上是文件下载，服务端会通过输出流将Excel文件下载到客户端浏览器

实现步骤：

- 设计Excel模板文件
- 查询近30天的运营数据
- 将查询到的运营数据写入模板文件
- 通过输出流将Excel文件下载到客户端浏览器

复制"C:\baidunetdiskdownload\资料\day12\运营数据报表模板.xlsx"

到sky-server/src/main/resources/template/运营数据报表模板.xlsx

ReportController.java

```
1     /**
2      * 导出运营报表
3      *
4      * @param resp
5      */
6     @GetMapping("/export")
7     @ApiOperation("导出运营报表")
8     public void exportExcel(HttpServletResponse resp) {//获得输出流
9         reportService.exportBusinessData(resp);
10    }
```

ReportService.java

```java
    /**
     * 导出运营报表
     * @param resp
     */
    void exportBusinessData(HttpServletResponse resp);
```

ReportServiceImpl.java

```java
    @Autowired
    private WorkspaceService workspaceService;

        /**
     * 导出运营报表
     * @param resp
     */
    @Override
    public void exportBusinessData(HttpServletResponse resp) {
        //1. 查询数据库，获取营业数据--查询最近30天的运营数据
        LocalDate dateBegin = LocalDate.now().minusDays(30);
        LocalDate dateEnd = LocalDate.now().minusDays(1);

        //查询概览数据
        BusinessDataVO businessDataVO = workspaceService.getBusinessData(
                LocalDateTime.of(dateBegin, LocalTime.MIN),
                LocalDateTime.of(dateEnd, LocalTime.MAX)
        );

        //2. 通过POI写入到excel文件中
        InputStream in =
this.getClass().getClassLoader().getResourceAsStream("template/运营数据报表模板.xlsx");//从类路径下读取资源

        //基于模板文件创建新的excel文件
        try {
            XSSFWorkbook excel=new XSSFWorkbook(in);
            //获取sheet标签页
            XSSFSheet sheet1 = excel.getSheet("Sheet1");
            //填充时间
            sheet1.getRow(1).getCell(1).setCellValue("时
间："+dateBegin+"至"+dateEnd);

            //填充第四行
            XSSFRow row = sheet1.getRow(3);
            row.getCell(2).setCellValue(businessDataVO.getTurnover());

 row.getCell(4).setCellValue(businessDataVO.getOrderCompletionRate());
            row.getCell(6).setCellValue(businessDataVO.getNewUsers());

            //填充第五行
            row=sheet1.getRow(4);

 row.getCell(2).setCellValue(businessDataVO.getValidOrderCount());
            row.getCell(4).setCellValue(businessDataVO.getUnitPrice());
```

```
41
42              //填充明细数据
43              for(int i=0;i<30;i++){
44                  LocalDate date=dateBegin.plusDays(i);
45                  BusinessDataVO businessData =
workspaceService.getBusinessData(
46                          LocalDateTime.of(date, LocalTime.MIN),
47                          LocalDateTime.of(date, LocalTime.MAX)
48                  );
49                  row=sheet1.getRow(7+i);//获得某一行
50                  row.getCell(1).setCellValue(date.toString());
51                  row.getCell(2).setCellValue(businessData.getTurnover());
52
 row.getCell(3).setCellValue(businessData.getValidOrderCount());
53
 row.getCell(4).setCellValue(businessData.getOrderCompletionRate());
54                  row.getCell(5).setCellValue(businessData.getUnitPrice());
55                  row.getCell(6).setCellValue(businessData.getNewUsers());
56
57              }
58
59              //3. 通过输出流下载到客户端浏览器
60              ServletOutputStream outputStream = resp.getOutputStream();
61              excel.write(outputStream);
62
63              //4. 关闭资源
64              excel.close();
65              outputStream.flush();
66              outputStream.close();
67
68          } catch (IOException e) {
69              throw new RuntimeException(e);
70          }finally {
71              try {
72                  in.close();
73              } catch (IOException e2) {
74                  e2.printStackTrace();
75              }
76          }
77
78
79      }
```

运营数据报表

时间：2025-05-02至2025-05-31

概览数据

营业额	¥2,333.00	订单完成率	70.00%	新增用户数	1
有效订单	7	平均客单价	¥333.29		

明细数据

日期	营业额	有效订单	订单完成率	平均客单价	新增用户数
2025-05-02	¥0.00	0	0.00%	¥0.00	0
2025-05-03	¥0.00	0	0.00%	¥0.00	0
2025-05-04	¥0.00	0	0.00%	¥0.00	0
2025-05-05	¥0.00	0	0.00%	¥0.00	0
2025-05-06	¥0.00	0	0.00%	¥0.00	0
2025-05-07	¥0.00	0	0.00%	¥0.00	0
2025-05-08	¥0.00	0	0.00%	¥0.00	0
2025-05-09	¥0.00	0	0.00%	¥0.00	0
2025-05-10	¥0.00	0	0.00%	¥0.00	0
2025-05-11	¥0.00	0	0.00%	¥0.00	0
2025-05-27	¥0.00	0	0.00%	¥0.00	0
2025-05-28	¥606.00	1	100.00%	¥606.00	0
2025-05-29	¥362.00	1	50.00%	¥362.00	0
2025-05-30	¥1,365.00	5	83.33%	¥273.00	0
2025-05-31	¥0.00	0	0.00%	¥0.00	0