一、预约下单

1、生成订单后处理逻辑-封装短信接口

操作service_msm模块

（1）service_msm引入依赖

```
<dependency>
    <groupId>com.atguigu</groupId>
    <artifactId>rabbit_util</artifactId>
    <version>0.0.1-SNAPSHOT</version>
</dependency>
```

（2）service_msm添加配置

```
#rabbitmq地址
spring.rabbitmq.host=192.168.44.165
spring.rabbitmq.port=5672
spring.rabbitmq.username=guest
spring.rabbitmq.password=guest
```

（3）在model模块封装短信实体

```
@Data
@ApiModel(description = "短信实体")
public class MsmVo {

    @ApiModelProperty(value = "phone")
    private String phone;

    @ApiModelProperty(value = "短信模板code")
    private String templateCode;

```

```
    @ApiModelProperty(value = "短信模板参数")
    private Map<String,Object> param;
}
```

(4) service_msm封装service接口和实现

```
//发送短信接口
boolean send(MsmVo msmVo);

//发送短信实现
@Override
public boolean send(MsmVo msmVo) {
    if(!StringUtils.isEmpty(msmVo.getPhone())) {
        //String code = (String)msmVo.getParam().get("code");
        //仅为了测试
        String code = RandomUtil.getFourBitRandom();
        return this.sendMsmDx(msmVo.getPhone(),code);
    }
    return false;
}
```

(5) 封装mq监听器

```
@Component
public class SmsReceiver {
    @Autowired
    private MsmService msmService;

    @RabbitListener(bindings = @QueueBinding(
            value = @Queue(value = MqConst.QUEUE_MSM_ITEM, durable = "true"),
            exchange = @Exchange(value = MqConst.EXCHANGE_DIRECT_MSM),
            key = {MqConst.ROUTING_MSM_ITEM}
    ))
    public void send(MsmVo msmVo, Message message, Channel channel) {
        msmService.send(msmVo);
    }
}
```

2、生成订单后处理逻辑-更新排班数量

操作模块：service_hosp

（1）service_hosp引入依赖

```
<!--rabbitmq消息队列-->
<dependency>
    <groupId>com.atguigu.yygh</groupId>
    <artifactId>rabbit-util</artifactId>
    <version>1.0</version>
</dependency>
```

（2）添加配置

```
#rabbitmq地址
spring.rabbitmq.host=192.168.44.165
spring.rabbitmq.port=5672
spring.rabbitmq.username=guest
spring.rabbitmq.password=guest
```

（3）在model模块封装更新排班实体

```
@Data
@ApiModel(description = "OrderMqVo")
public class OrderMqVo {

    @ApiModelProperty(value = "可预约数")
    private Integer reservedNumber;

    @ApiModelProperty(value = "剩余预约数")
    private Integer availableNumber;

    @ApiModelProperty(value = "排班id")
```

```
    private String scheduleId;

    @ApiModelProperty(value = "短信实体")
    private MsmVo msmVo;
}
```

(4) 封装更新排班方法

操作ScheduleService

```
/**
     * 修改排班
     */
void update(Schedule schedule);

//实现方法
@Override
public void update(Schedule schedule) {
    schedule.setUpdateTime(new Date());
    //主键一致就是更新
    scheduleRepository.save(schedule);
}
```

(5) 封装mq监听器

```
@Component
public class HospitalReceiver {

    @Autowired
    private ScheduleService scheduleService;

    @Autowired
    private RabbitService rabbitService;

    @RabbitListener(bindings = @QueueBinding(
            value = @Queue(value = MqConst.QUEUE_ORDER, durable = "true"),
            exchange = @Exchange(value = MqConst.EXCHANGE_DIRECT_ORDER),
            key = {MqConst.ROUTING_ORDER}
```

```
    ))
    public void receiver(OrderMqVo orderMqVo, Message message, Channel channel) throw
        //下单成功更新预约数
        Schedule schedule = scheduleService.getById(orderMqVo.getScheduleId());
        schedule.setReservedNumber(orderMqVo.getReservedNumber());
        schedule.setAvailableNumber(orderMqVo.getAvailableNumber());
        scheduleService.update(schedule);
        //发送短信
        MsmVo msmVo = orderMqVo.getMsmVo();
        if(null != msmVo) {
            rabbitService.sendMessage(MqConst.EXCHANGE_DIRECT_MSM, MqConst.ROUTING_MS
        }
    }
}
```

3、生成订单后处理逻辑-完善订单接口

操作service_orders

（1）引入依赖

```
<dependency>
    <groupId>com.atguigu</groupId>
    <artifactId>rabbit_util</artifactId>
    <version>0.0.1-SNAPSHOT</version>
</dependency>
```

（2）添加配置

```
#rabbitmq地址
spring.rabbitmq.host=192.168.44.165
spring.rabbitmq.port=5672
spring.rabbitmq.username=guest
spring.rabbitmq.password=guest
```

(3) 修改OrderServiceImpl类下单方法

```
@Autowired
private RabbitService rabbitService;

//生成预约挂号订单
@Override
public Long saveOrders(String scheduleId, Long patientId) {

   ..................................

        //排班可预约数
        Integer reservedNumber = jsonObject.getInteger("reservedNumber");
        //排班剩余预约数
        Integer availableNumber = jsonObject.getInteger("availableNumber");
        //发送mq信息更新号源和短信通知
        OrderMqVo orderMqVo = new OrderMqVo();
        orderMqVo.setScheduleId(scheduleId);
        orderMqVo.setReservedNumber(reservedNumber);
        orderMqVo.setAvailableNumber(availableNumber);

        //短信提示
        MsmVo msmVo = new MsmVo();
        msmVo.setPhone(orderInfo.getPatientPhone());
        String reserveDate =
            new DateTime(orderInfo.getReserveDate()).toString("yyyy-MM-dd")
            + (orderInfo.getReserveTime()==0 ? "上午": "下午");
        Map<String,Object> param = new HashMap<String,Object>(){{
            put("title", orderInfo.getHosname()+"|"+orderInfo.getDepname()+"|"+orderI
            put("amount", orderInfo.getAmount());
            put("reserveDate", reserveDate);
            put("name", orderInfo.getPatientName());
            put("quitTime", new DateTime(orderInfo.getQuitTime()).toString("yyyy-MM-d
        }};
        msmVo.setParam(param);

        orderMqVo.setMsmVo(msmVo);
        rabbitService.sendMessage(MqConst.EXCHANGE_DIRECT_ORDER, MqConst.ROUTING_ORDE

    } else {//下单失败
        System.out.println("下单失败");
        throw new YyghException(20001,"下单失败");
```

```
41      }
42      //返回订单号
43      return orderInfo.getId();
44  }
```