**第7章-投保试算-理财类（实战-阅读源码）**

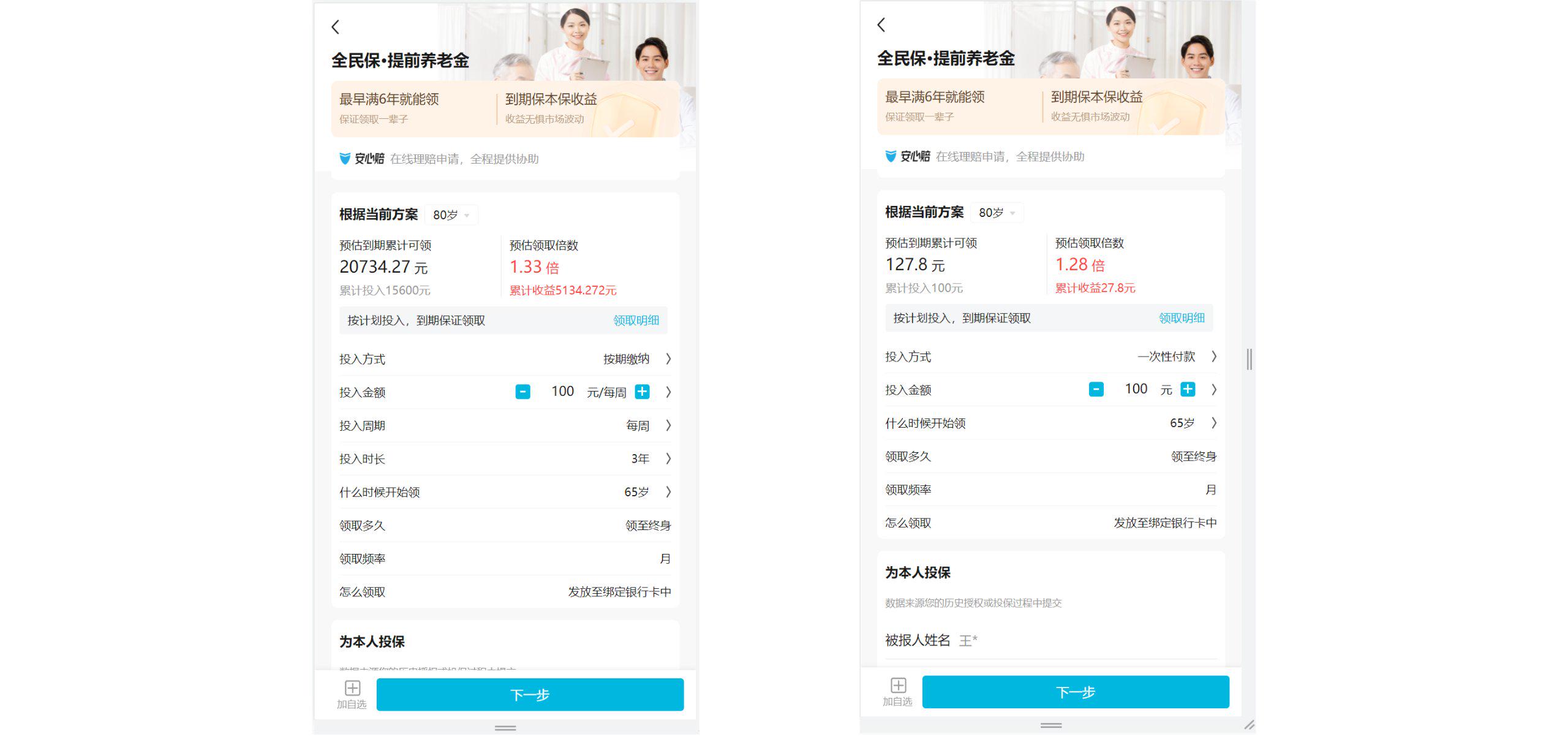
**学习目标**

* 开发并能阐述理财试算的整个业务处理流程
* 使用策略模式和门面模式完成收益试算的处理

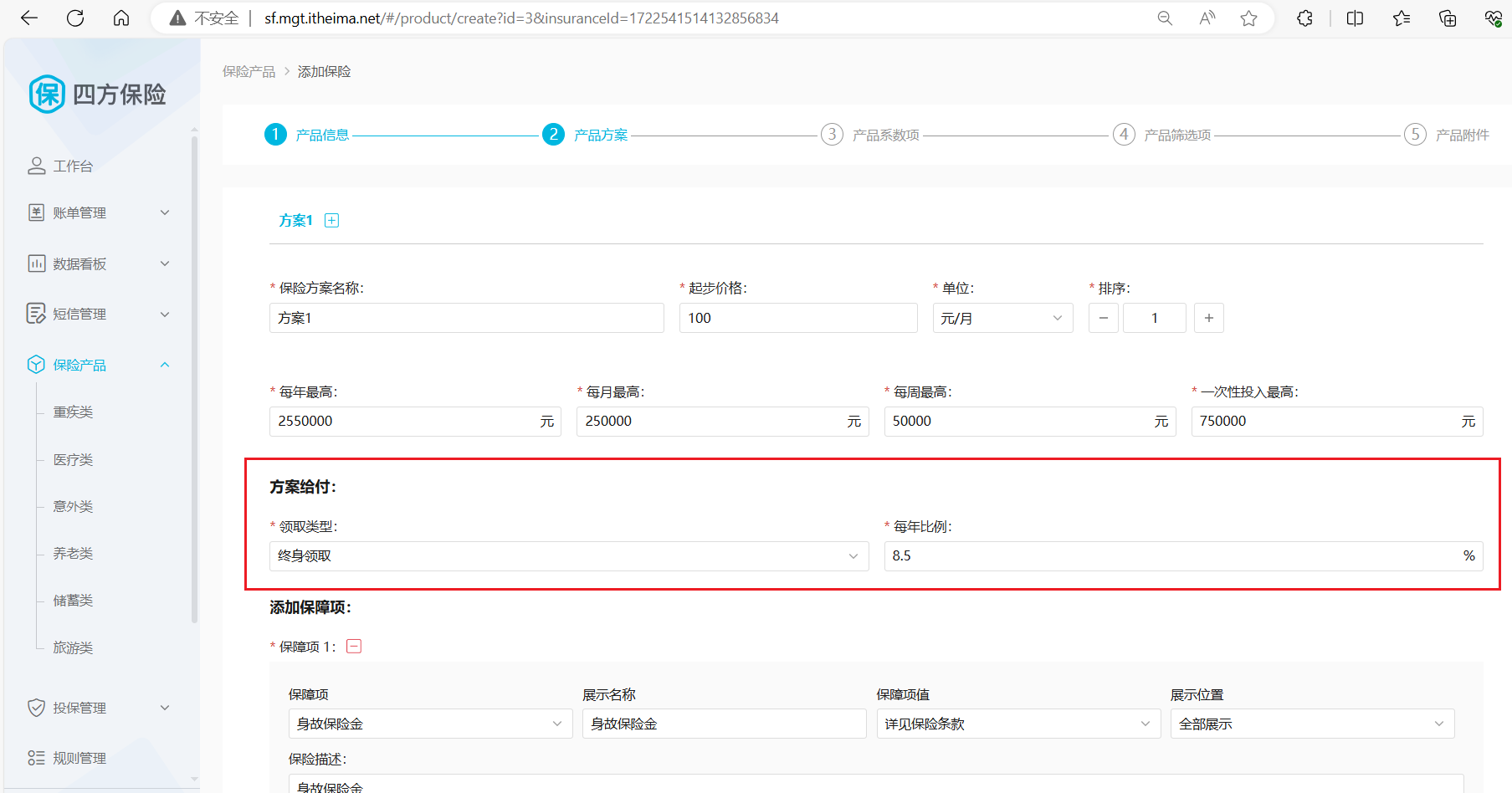
**1、收益试算分析**

**1.1、需求分析**

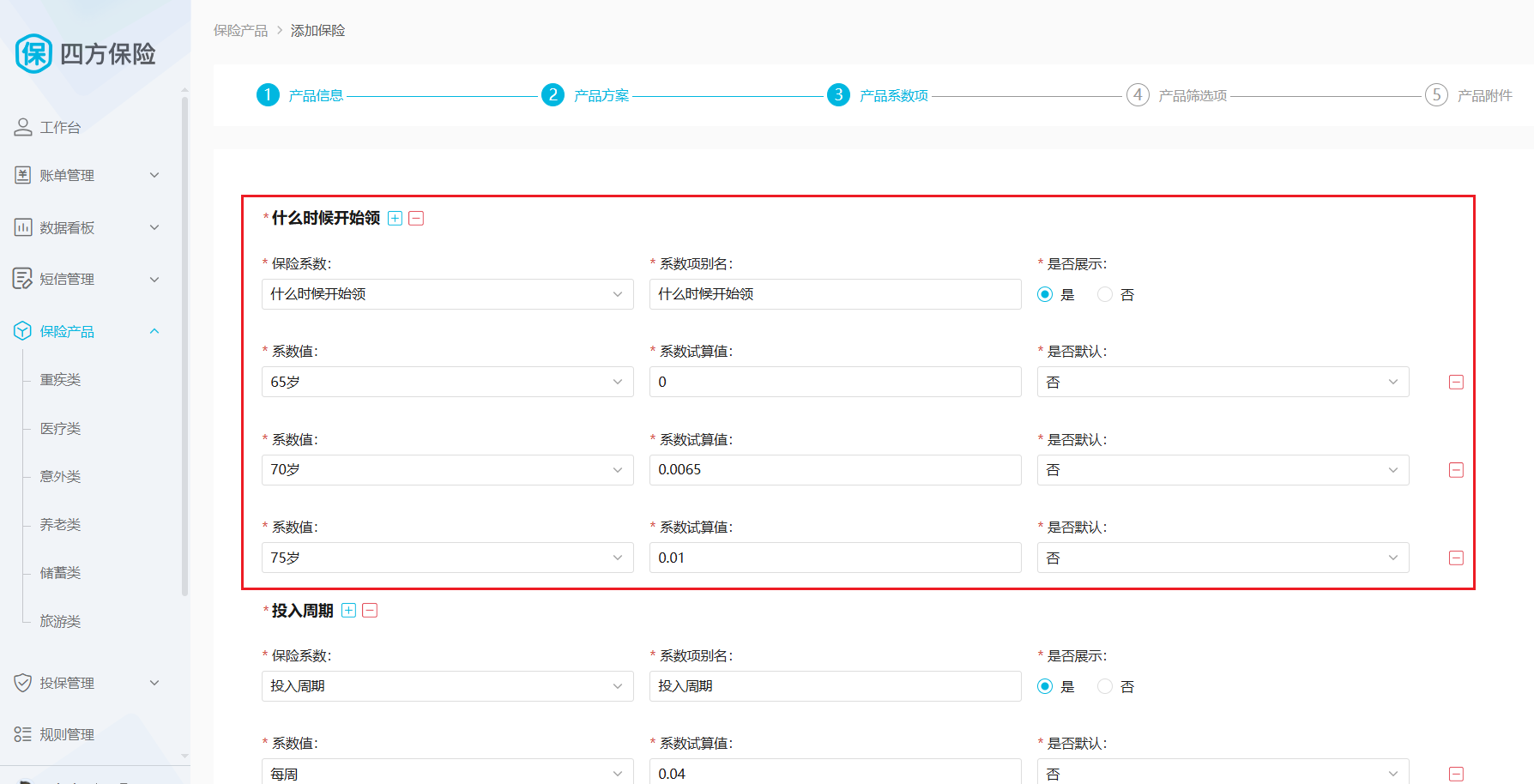
收益试算主要是依据用户的投入及领取方式然后整体计算投入及收益情况，主体需求就是：通过产品投入及方案给付计划来计算整体投入及未来收益情况，来我们看下功能效果图：



从上图分析发现：同一个理财保险选择不同的投入方式，获得的最终收益不同，而理财收益比例是根据在发布理财保险时制定的方案给付计划是有关联的



除此之外系数也会影响整个的计算结果，例如：领取的时间越晚收益越高，投入时间越长收益也越高，还有其他的系数都会影响最终的收益金额，这些要看具体的设置情况



**1.2、接口分析**

在app端产品详情页面中，选择了不同的系数之后会发起收益的计算；如图：



Swagger接口地址：http://localhost:7065/doc.html#/default/%E4%BF%9D%E9%99%A9%E6%8A%95%E4%BF%9D/doEarningsUsingPOST

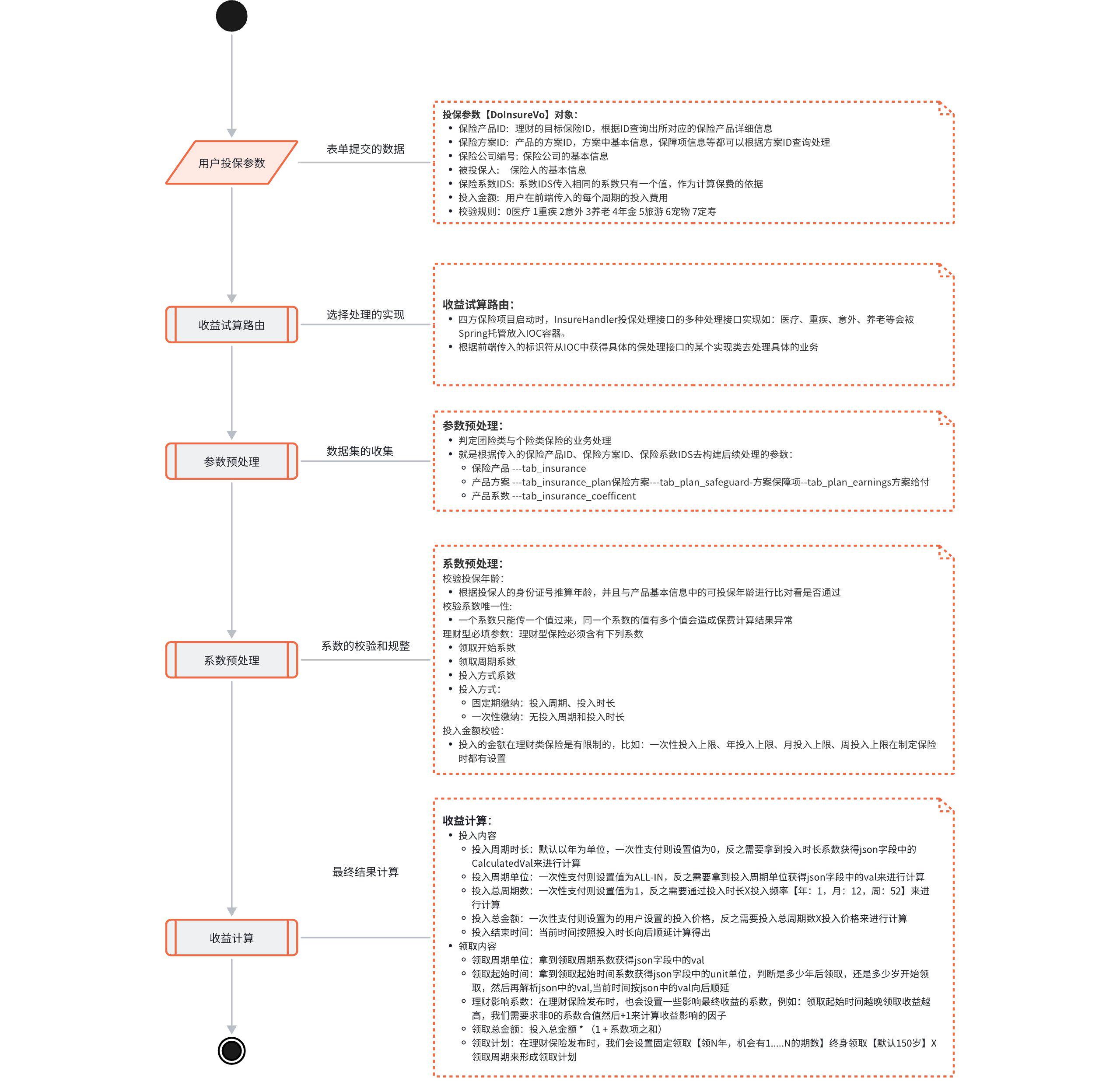
建议：从浏览器中跟踪接口信息如下：

|  |  |
| --- | --- |
| 接口描述 | 说明 |
| **请求地址** | http://sf.app.itheima.net/api/insurance-app/insure/do-earnings |
| **请求方式** | POST |
| **请求参数** | {  "checkRule": "1", # 路由key；*0医疗 1重疾 2意外 3养老 4储蓄 5旅游*  "companyNo": "100000001", # 企业编号  "insuranceId": "1722449243278430209", # 保险产品id  "price": 100, # 投入金额  "insuranceCoefficentIds": [ # 保险系数项id集合  "1726792497633431553",  "1726792497666985985",  "1726792497650208770",  "1726792497658597381",  "1726792497658597383"  ],  "insurancePlanId": "1727575594796875778", #保险方案id  "customerRelationIds": [ # 被投保人id集合  "1225092536525656054"  ]  } |
| **响应结果** | {  "premium": 100, #投入金额  "premiums": 100, #投入总金额  "periods": 1, #投入期数  "periodicUnit": "ALL-IN", #投入方式  "periodic": "0", #投入周期时长  "putInEndTime": "2024-02-02 20:49:17", #投入截止时间  "actualGetStartTime": "2048-02-02 20:49:17", #领取开始时间  "actualGetPeriodicUnit": "MONTH", #领取频率  "periodicVos": [ #每期详情  {  "periodic": "第1期",  "receivedAmount": 0.71000000,  "actualGetStartTime": "2048-02-02 20:49:17"  },  {  "periodic": "第2期",  "receivedAmount": 0.71000000,  "actualGetStartTime": "2048-03-02 20:49:17"  },  .....  ],  "receivedAmounts": 127.80, #领取总金额  "accumulatedEarnings": 27.80000000, #累加收益  "multiple": 1.28 #累计倍数 } |

**1.3、计算流程**

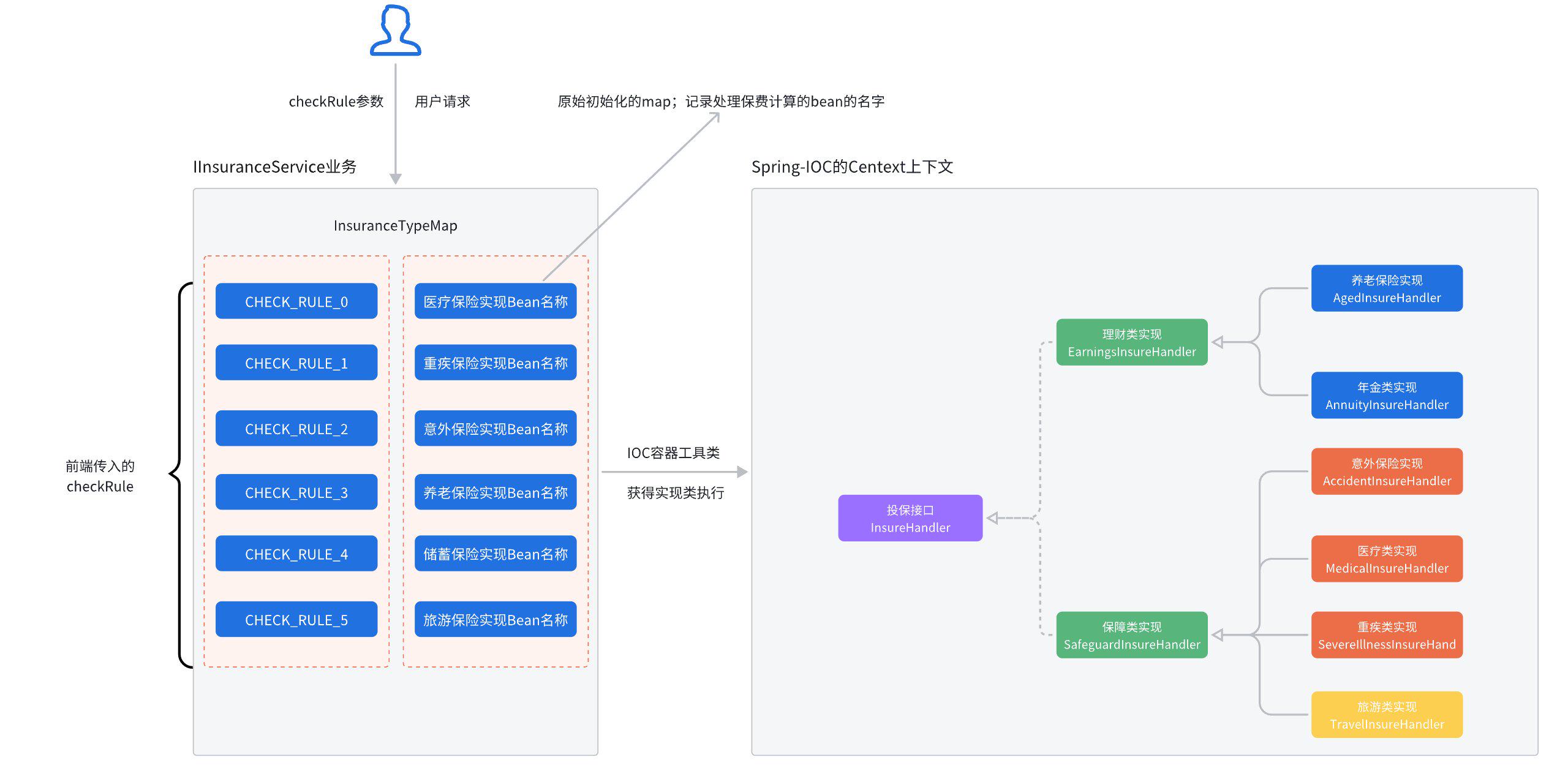
我们梳理一下整个收益试算的业务流程，用户提交所选择的保险参数后，后台程序经过下列4个步骤：

* 收益试算路由：保险的产品有很多类，各自有自己需要处理的业务逻辑，根据传入计算标识选择对应的计算方式
* 参数预处理：关键性数据需要从后端数据库再次查询，查询后组织成需要的数据格式备用
* 系数预处理：参与系数的校验、最终收益金额与领取方式、领取频率、投入方式等有关
* 理财收益计算：按照个险类保险保费、团险类保险保费公式计算出用户最终的投保保费



**2、收益试算策略**

在四方保险中不同类型从产品，对应不同的保费计算；所以针对这些类型都设计了收益计算处理类InsureHandler；并应用策略模式初始化到map中；再根据前端传入的标识符从map中获取对应的处理器进行收益计算。如下图：



* **项目启动阶段：**

项目启动时，InsureHandler投保处理接口实现如：医疗、重疾、意外、旅游、养老、年金等会被Spring托管放入IOC容器。

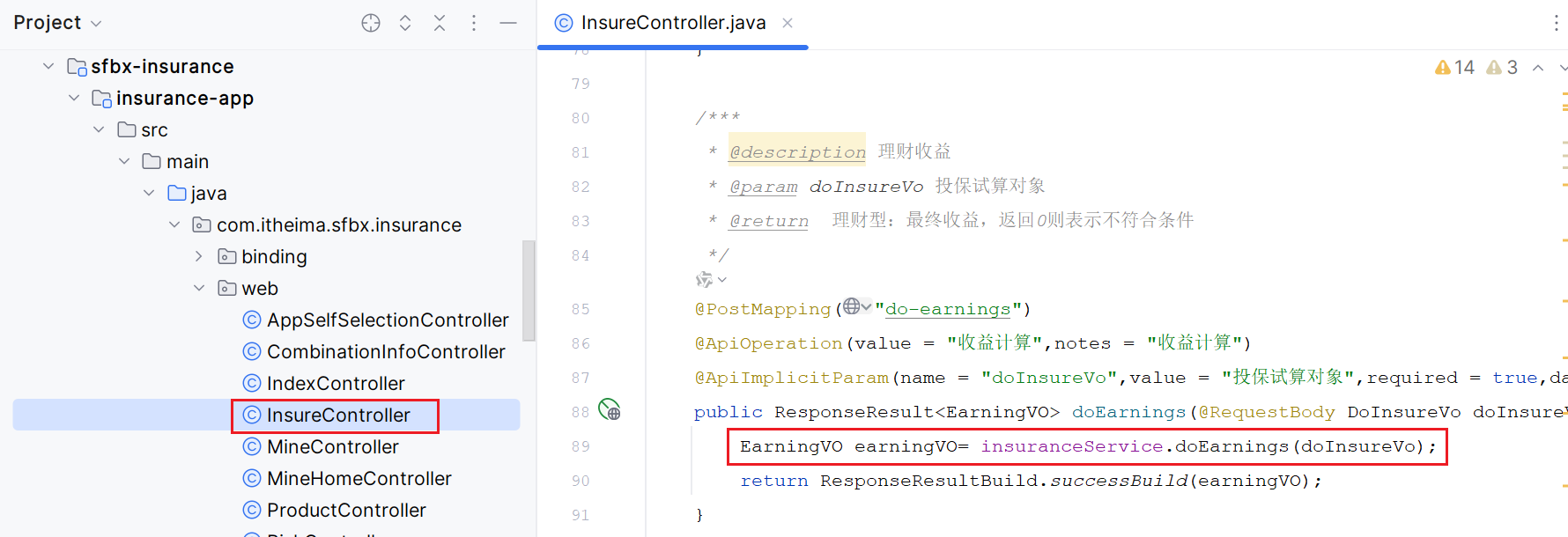
静态变量InsuranceTypeMap被构建，并设置insuranceTypeMap中的内容：K：checkRule参数，V：投保实现类bean名称

* **业务请求阶段：**

传入checkRule参数从insuranceTypeMap中依据K值找到V值，也就是拿到处理类的bean名称

依据bean名称通过IOC容器工具类从IOC中获得具体的保处理接口的某个实现类去处理具体的业务

根据接口信息；具体代码中跟踪如下：



业务方法中；应用策略模式获取对应的保费处理器：



**3、参数预处理**

**3.1、流程**

在上一步确定了收益处理器之后；在收益处理器中；首先进行的是参数预处理：流程如下：



**3.2、回顾产品模型**

参数预处理就是根据产品id查询相关的所有数据；那产品关联的数据有哪些？我们通过概念模型直观的查看一下：



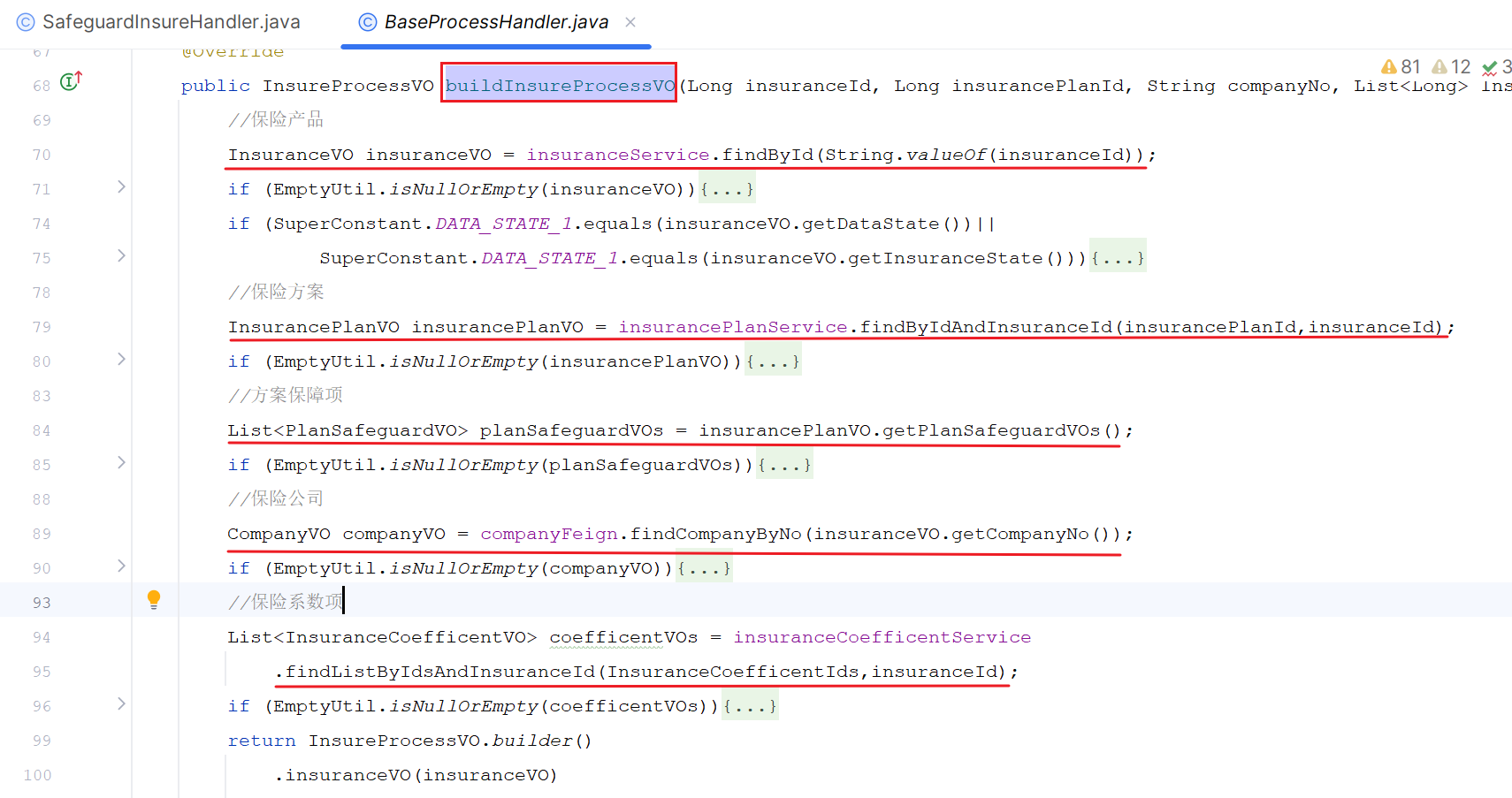
**3.3、功能实现**

了解保险产品的关系之后；代码中查询如下：



具体查询的代码：

com.itheima.sfbx.insurance.handler.impl.BaseProcessHandler#buildInsureProcessVO



从代码中我们可以看出；最终查询出来的结果全部都封装到了 InsureProcessVO 也就是；在实现过程中，查询到的数据都往这个返回结果中设置即可。

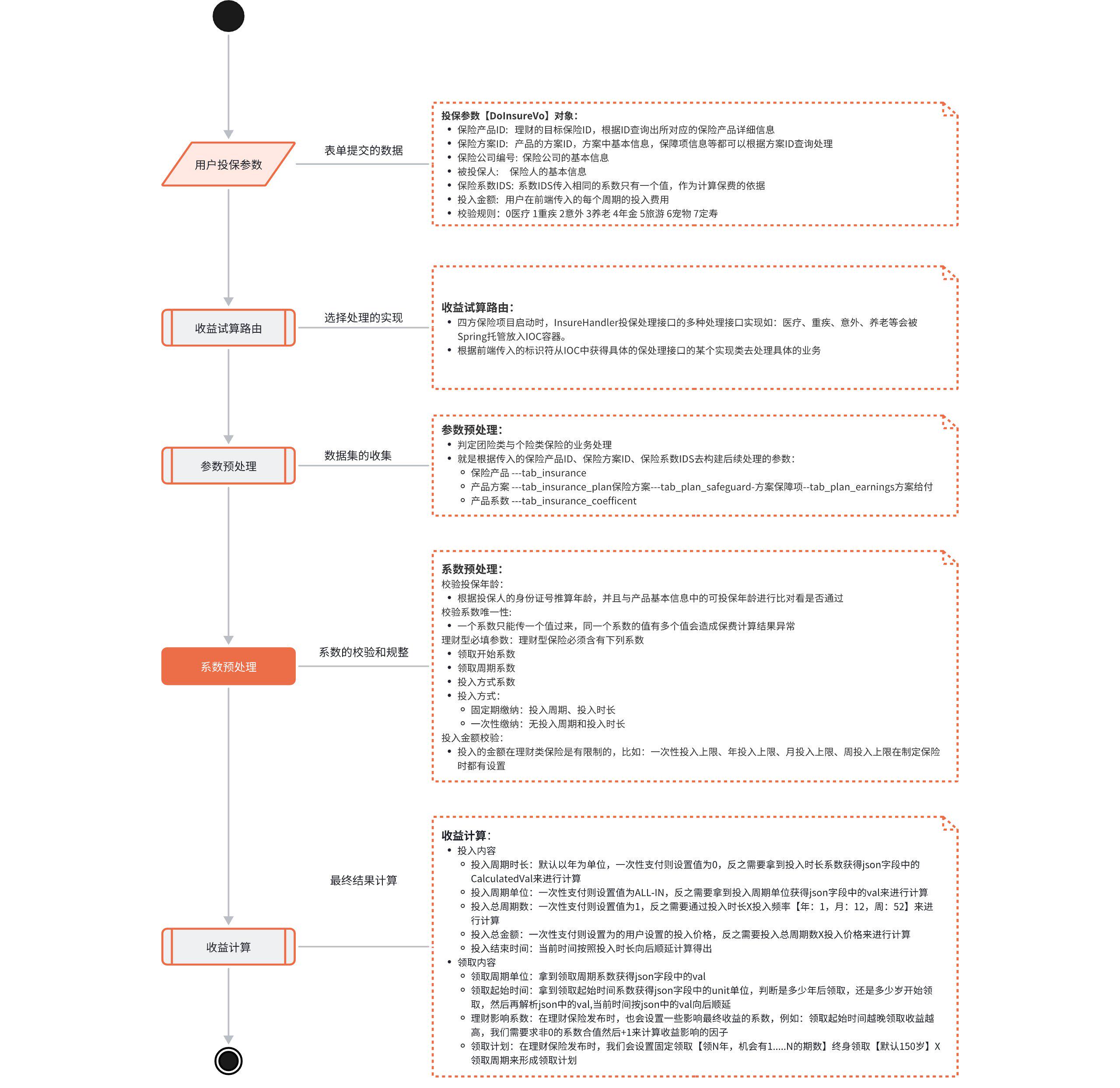
参数的预处理：根据出入的保险产品ID、保险方案ID、保险系数IDS去构建后续的处理参数，在做业务处理时候需要各位注意的是：

* 保险产品的ID能查询出保险产品，且保险产品状态必须是上线状态
* 保险方案必须是属于当前保险产品，且同时查询出方案保障项、方案给付方案
* 按传入的保险系数IDS查询保险系数项，且保险系数属于当前保险产品

**4、系数预处理**

**4.1、流程**

在参数预处理完后；再根据如下流程需要处理系数项：



**4.2、功能分析**

**目标**：

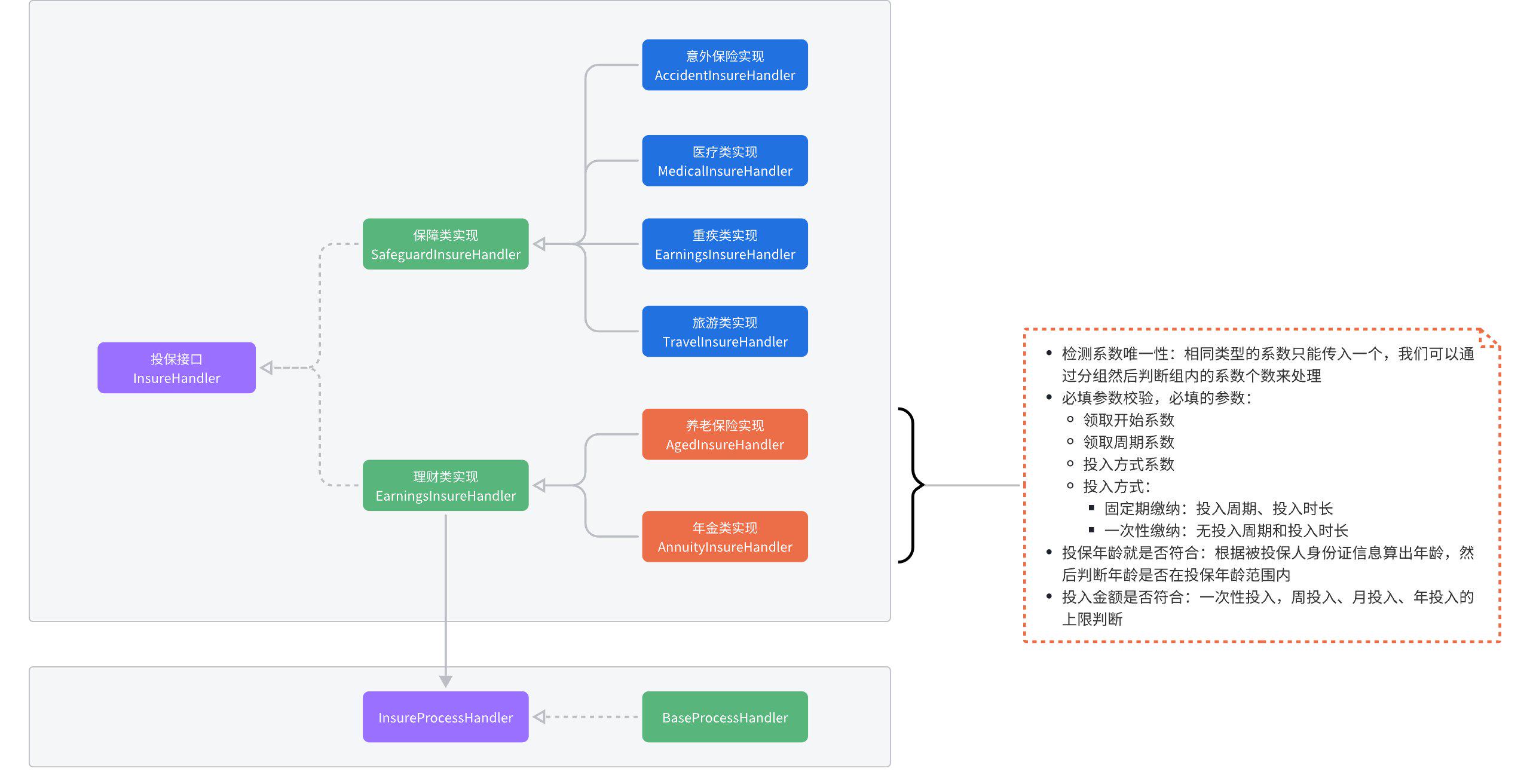
最终收益的内容与各系数有关系；所以要确定下来详情页面中选择的系数项及隐含的一些系数获取并处理。

也就是说：



|  |
| --- |
| 上述的系数项查询是根据前端选择的系数项id数组从数据库中查询出来的系数项列表 |

这些从前端传递过来的系数项是否符合标准；根据如下要求进行校验：



|  |
| --- |
| * 检测系数唯一性：相同类型的系数只能传入一个，我们可以通过分组然后判断组内的系数个数来处理 * 必填参数校验，必填的参数： * 领取开始系数 * 领取周期系数 * 投入方式系数 * 投入方式： * 固定期缴纳：投入周期、投入时长 * 一次性缴纳：无投入周期和投入时长 * 投保年龄就是否符合：根据被投保人身份证信息算出年龄，然后判断年龄是否在投保年龄范围内 * 投入金额是否符合：一次性投入，周投入、月投入、年投入的上限判断 |

在管理后台中对保险产品设置的时候：



而在前端app中输入的投入金额是变化且经过网络传输；所以需要校验。

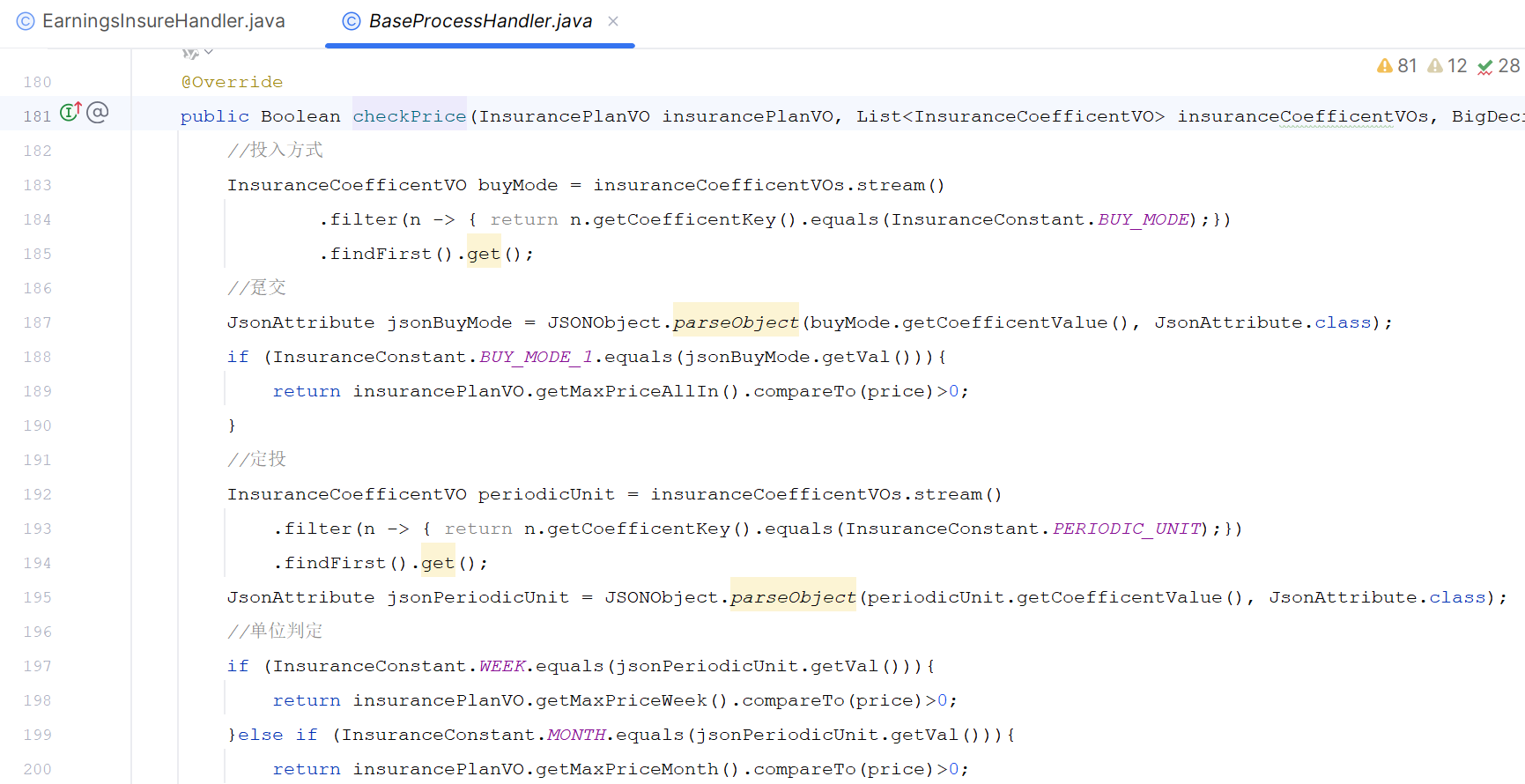


**4.3、功能实现**

理财类的系数与前面保障类的系数校验唯一性、年龄的做法是一样的，代码实现也相同。

下面看看理财类中对投保金额 与产品方案中设置的进行校验：





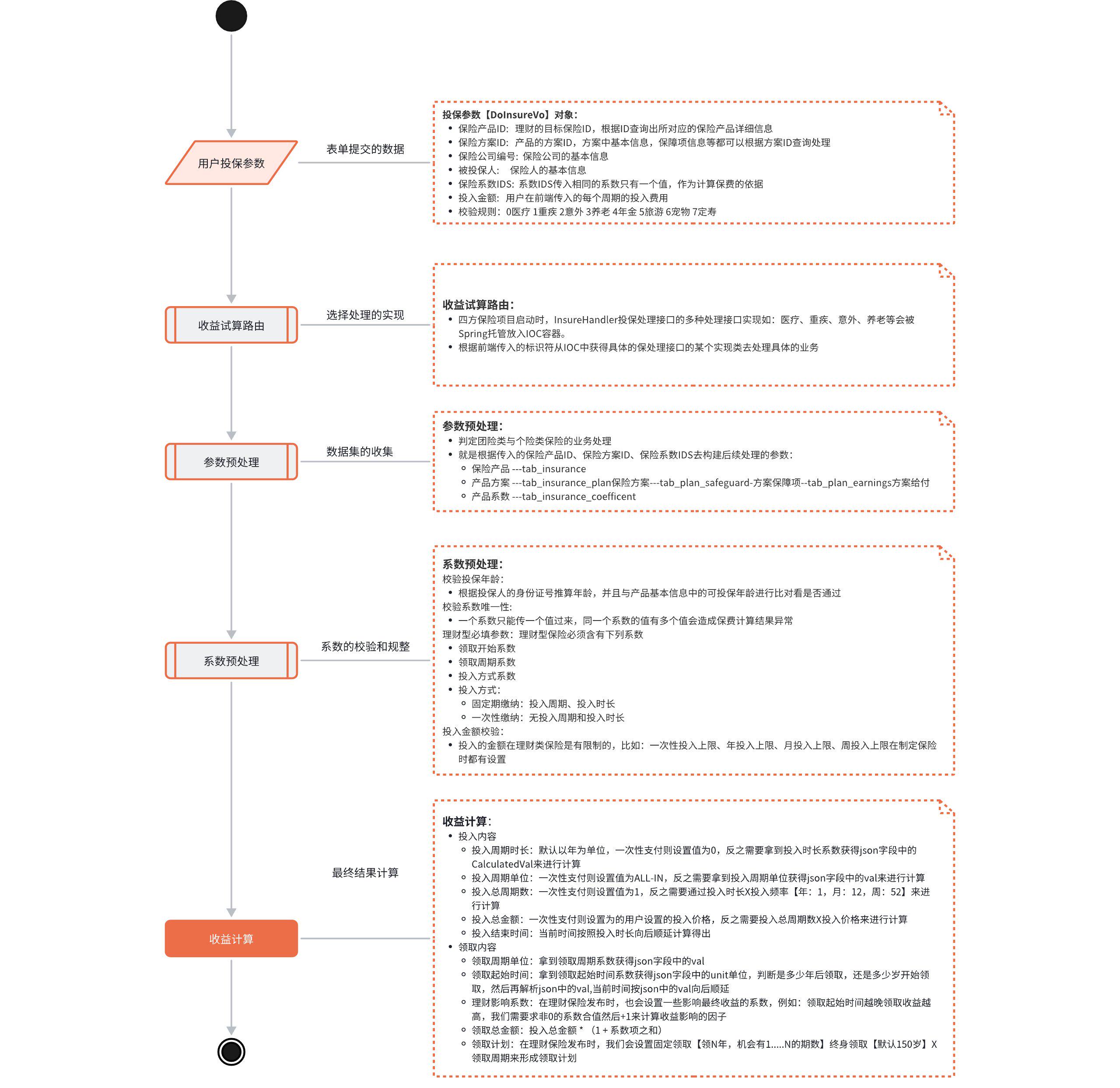
**5、收益试算**

**5.1、收益分析**

返回给前端的数据结构如下：

|  |
| --- |
| Java {  "premium": 100, #前端传的；投入金额  "premiums": 100, #后端计算的投入总金额  "periods": 1, #投入期数  "periodicUnit": "ALL-IN", #投入方式  "periodic": "0", #投入周期时长  "putInEndTime": "2024-02-02 20:49:17", #投入截止时间  "actualGetStartTime": "2048-02-02 20:49:17", #领取开始时间  "actualGetPeriodicUnit": "MONTH", #领取频率  "periodicVos": [ #每期详情  {  "periodic": "第1期",  "receivedAmount": 0.71000000,  "actualGetStartTime": "2048-02-02 20:49:17"  },  {  "periodic": "第2期",  "receivedAmount": 0.71000000,  "actualGetStartTime": "2048-03-02 20:49:17"  },  .....  ],  "receivedAmounts": 127.80, #领取总金额  "accumulatedEarnings": 27.80000000, #累加收益  "multiple": 1.28 #累计倍数 } |

上述所有的数据可以通过如下计算：



所以具体分为投入内容、领取内容两部分；接下来针对这两部分进行分析与实现说明。

**5.2、投入内容**

**5.2.1、分析**

投入内容全部都由前端传入的系数项id决定了；再次说明和查询它的意义在于：

* 前端传递过程中数据的安全性问题，所以最好还是根据id到后端查询一遍；
* 然后返回之后的内容在app端的 领取明细 中需要展示。

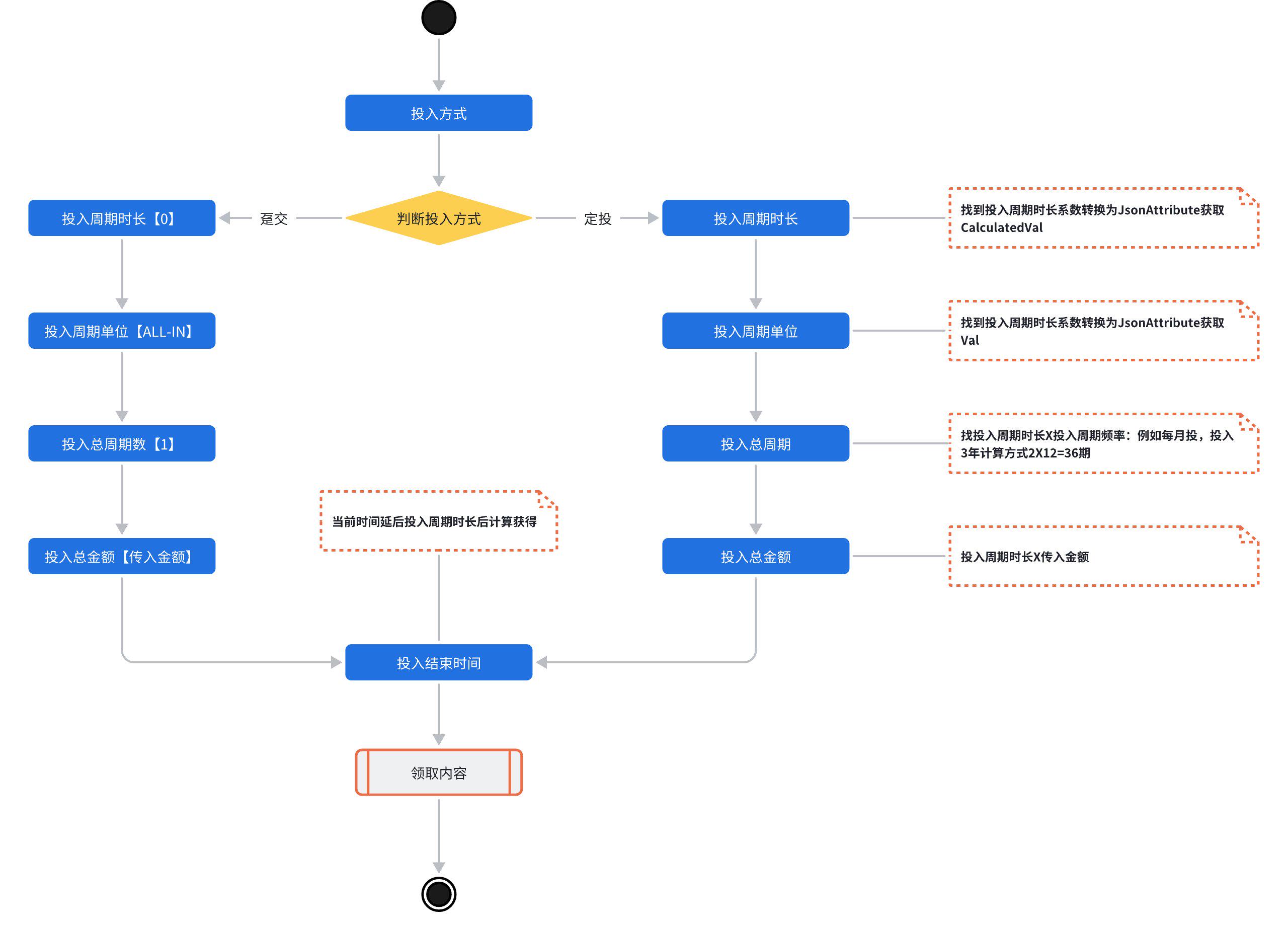




所以具体投入内容的获取或者说查询可以如下逻辑：

|  |
| --- |
| * 投入内容 * 投入时长：默认以年为单位，一次性支付则设置值为0，反之需要拿到投入时长系数获得json字段中的CalculatedVal来进行计算 * 投入周期单位：一次性支付则设置值为ALL-IN，反之需要拿到投入周期单位获得json字段中的val来进行计算 * 投入总周期数：一次性支付则设置值为1，反之需要通过投入时长X投入频率【年：1，月：12，周：52】来进行计算 * 投入总金额：一次性支付则设置为的用户设置的投入价格，反之需要投入总周期数X投入价格来进行计算 * 投入结束时间：当前时间按照投入时长向后顺延计算得出 |

流程展示如下：



**5.2.2、实现**





**5.3、领取内容**

**5.3.1、分析**

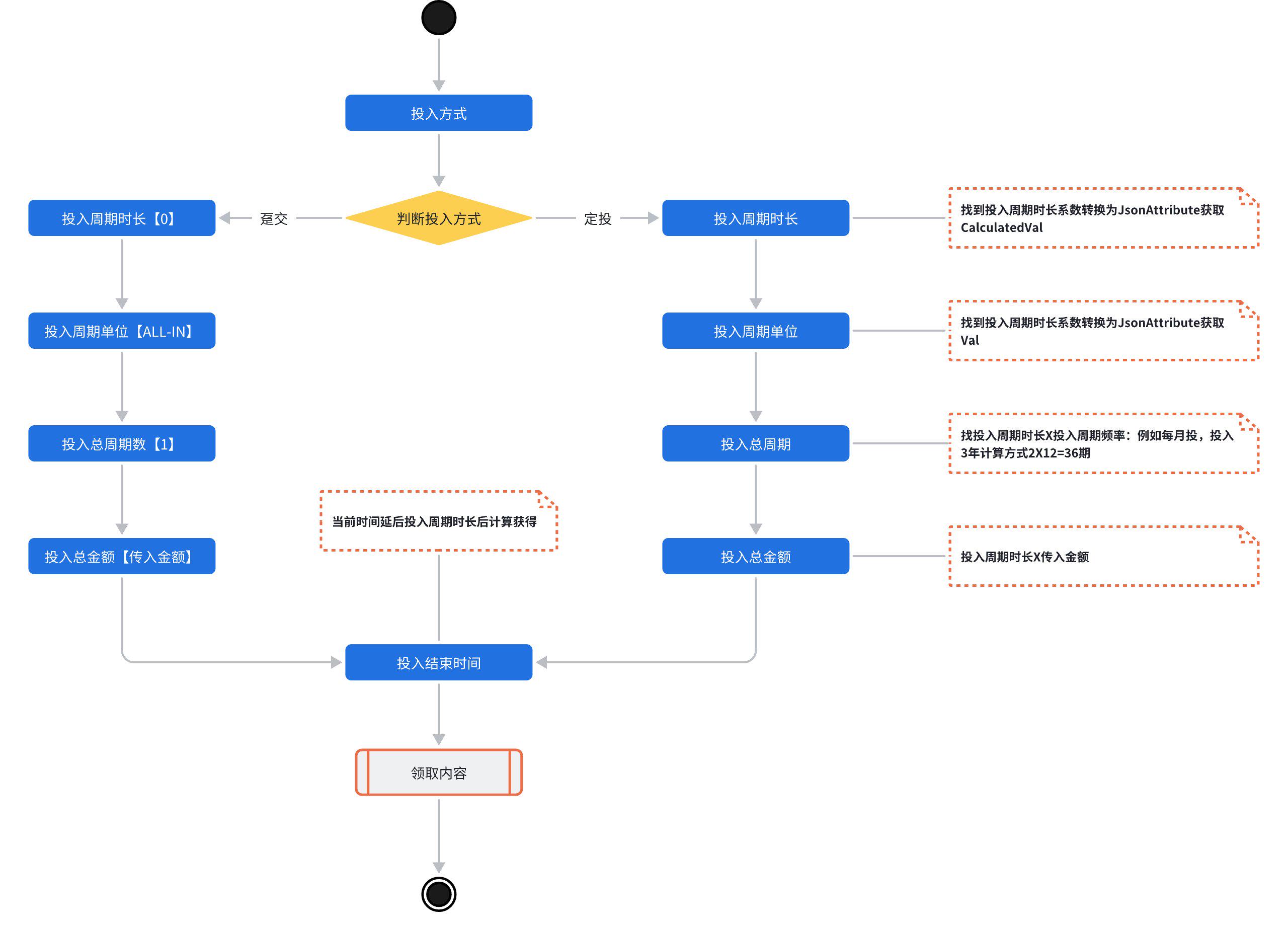
界面中需要展示领取的内容：



分析领取内容的计算如下：

|  |
| --- |
| * 领取内容 * 领取周期单位：拿到领取周期系数获得json字段中的val * 领取起始时间：拿到领取起始时间系数获得json字段中的unit单位，判断是多少年后领取，还是多少岁开始领取，然后再解析json中的val,当前时间按json中的val向后顺延 * 领取计划：在理财保险发布时，我们会设置固定领取【领N年，机会有1.....N的期数】终身领取【默认150岁】X领取周期来形成领取计划 |

流程展示如下：



**5.3.2、实现**



**6、问题**

* 理财类保险产品有哪些？
* 说一下投保试算的流程？
* **简历职责-候选**：独立完成在app端中用户选择理财类保险产品保险方案及系数项之后的收益试算功能；

**源码阅读实战**：

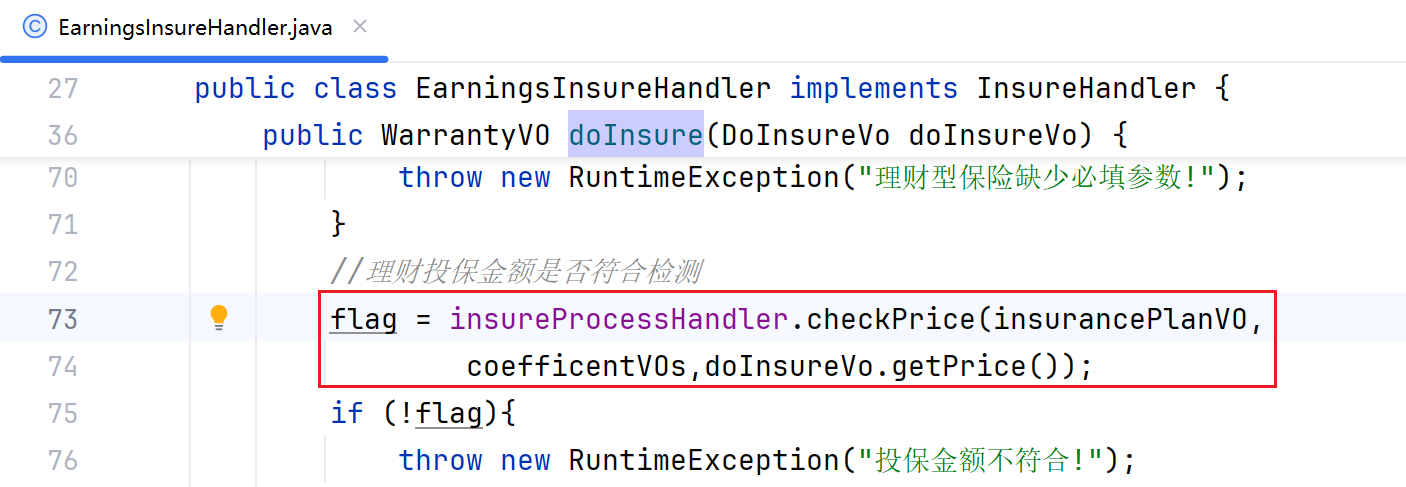
时间：16点

内容：

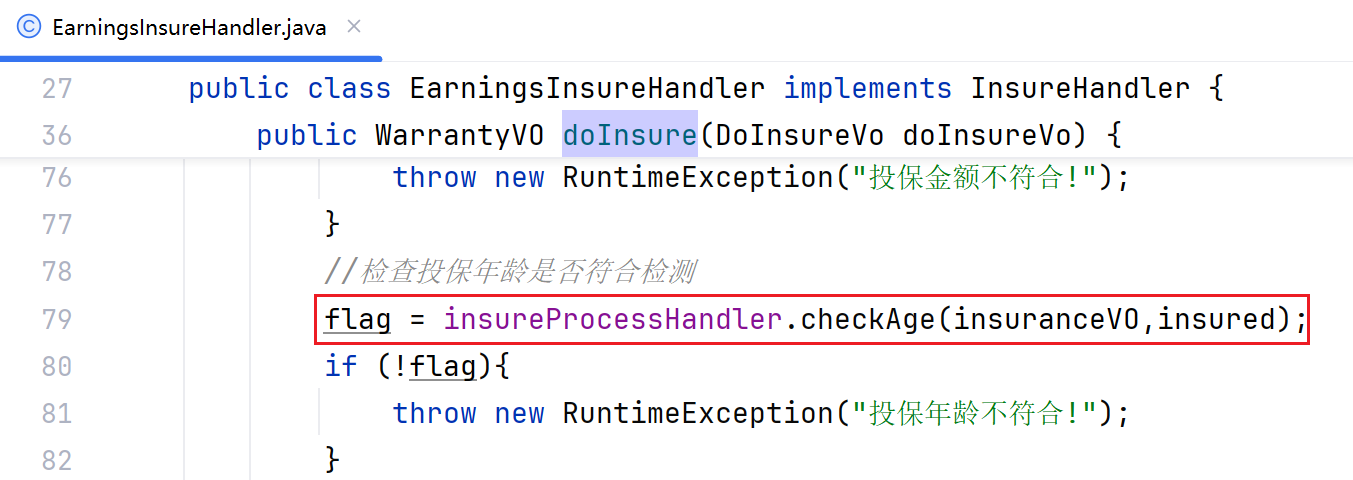
* 流程图；每组一人上来讲解理财类的投保试算流程
* 在投保试算的过程中涉及的SQL语句
* 讲说如下方法
* 1、com.itheima.sfbx.insurance.handler.impl.EarningsInsureHandler#doInsure中的第68行



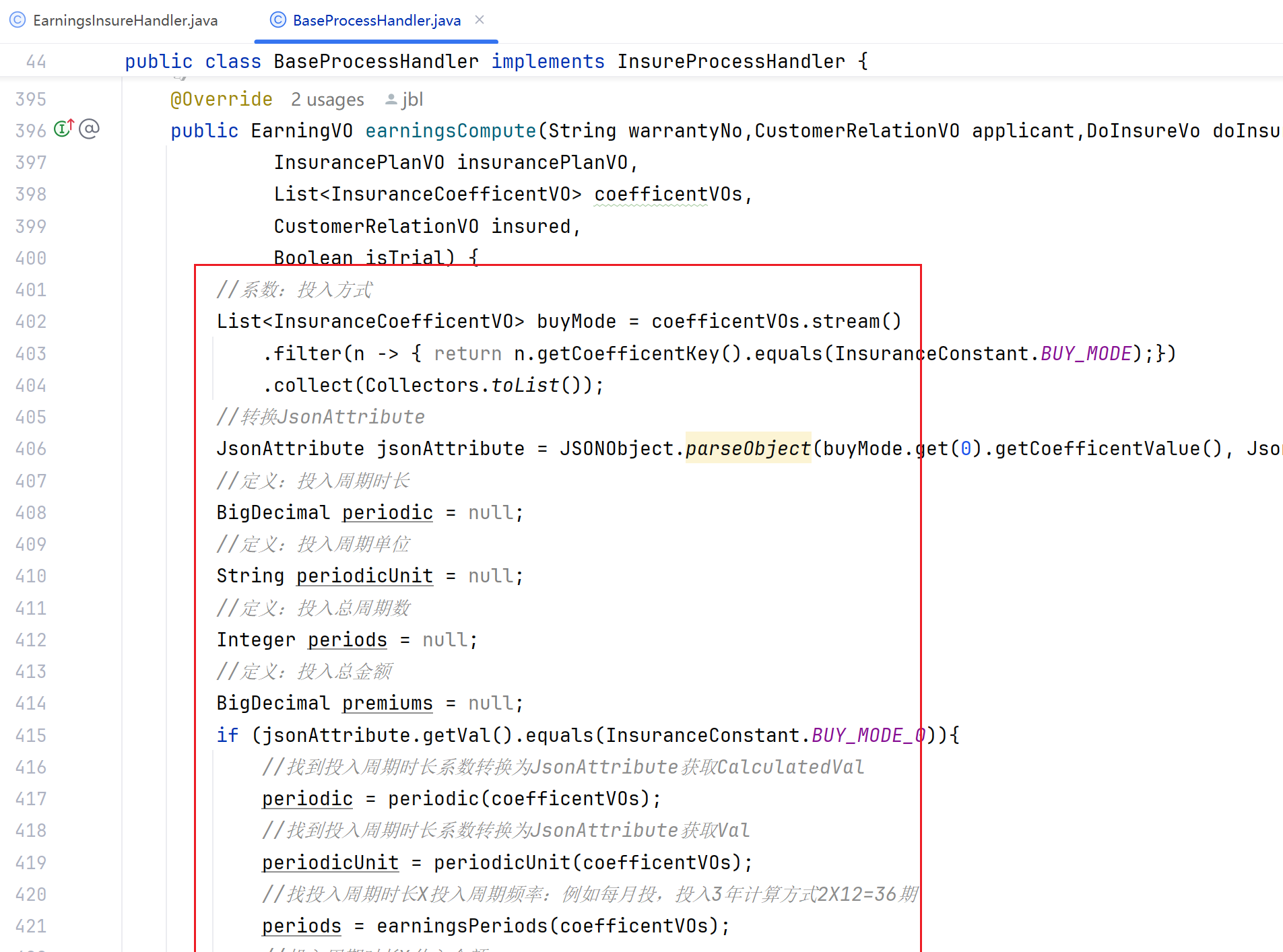
* 2、com.itheima.sfbx.insurance.handler.impl.EarningsInsureHandler#doInsure中的第73行



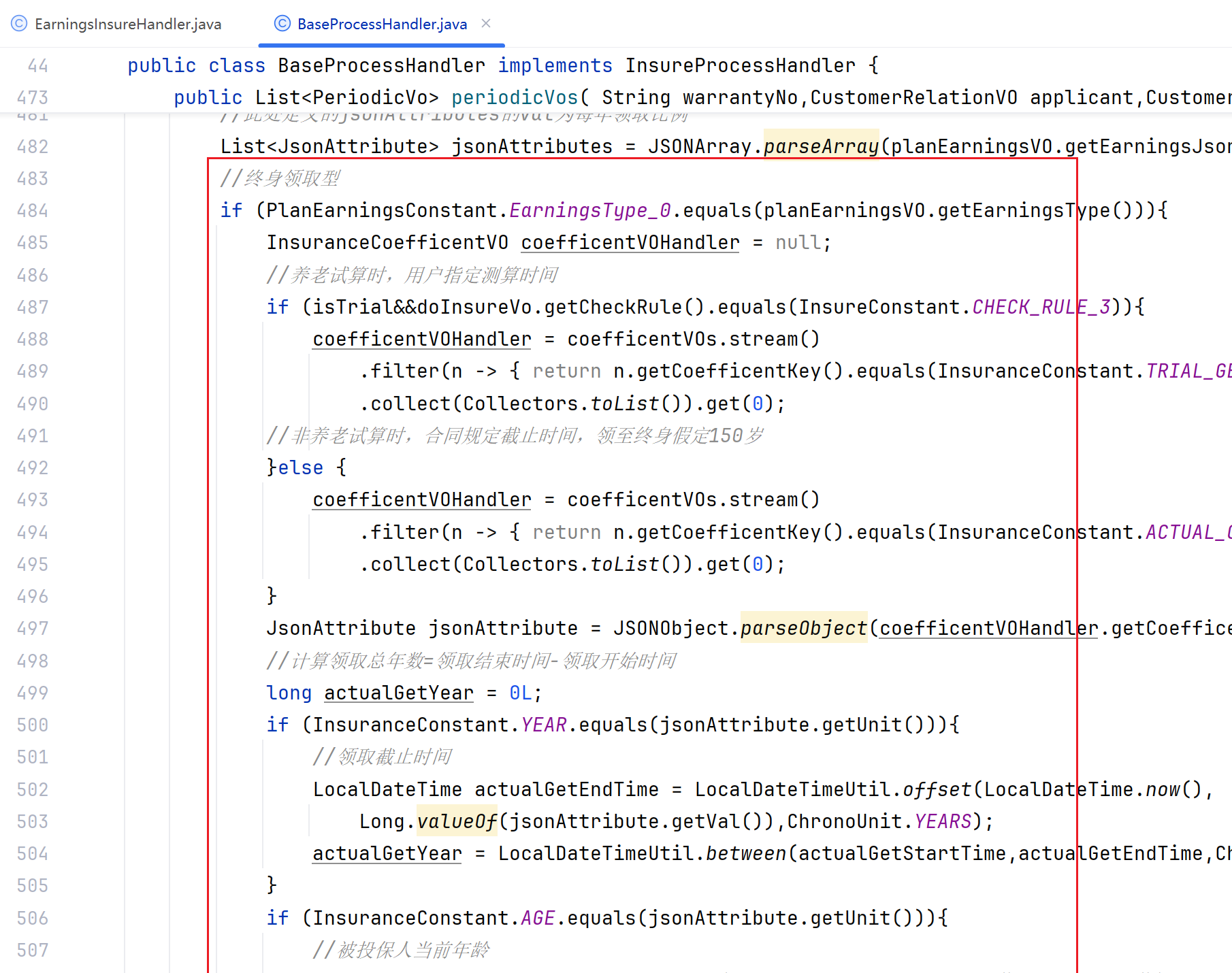
* 3、com.itheima.sfbx.insurance.handler.impl.EarningsInsureHandler#doInsure中的第79行



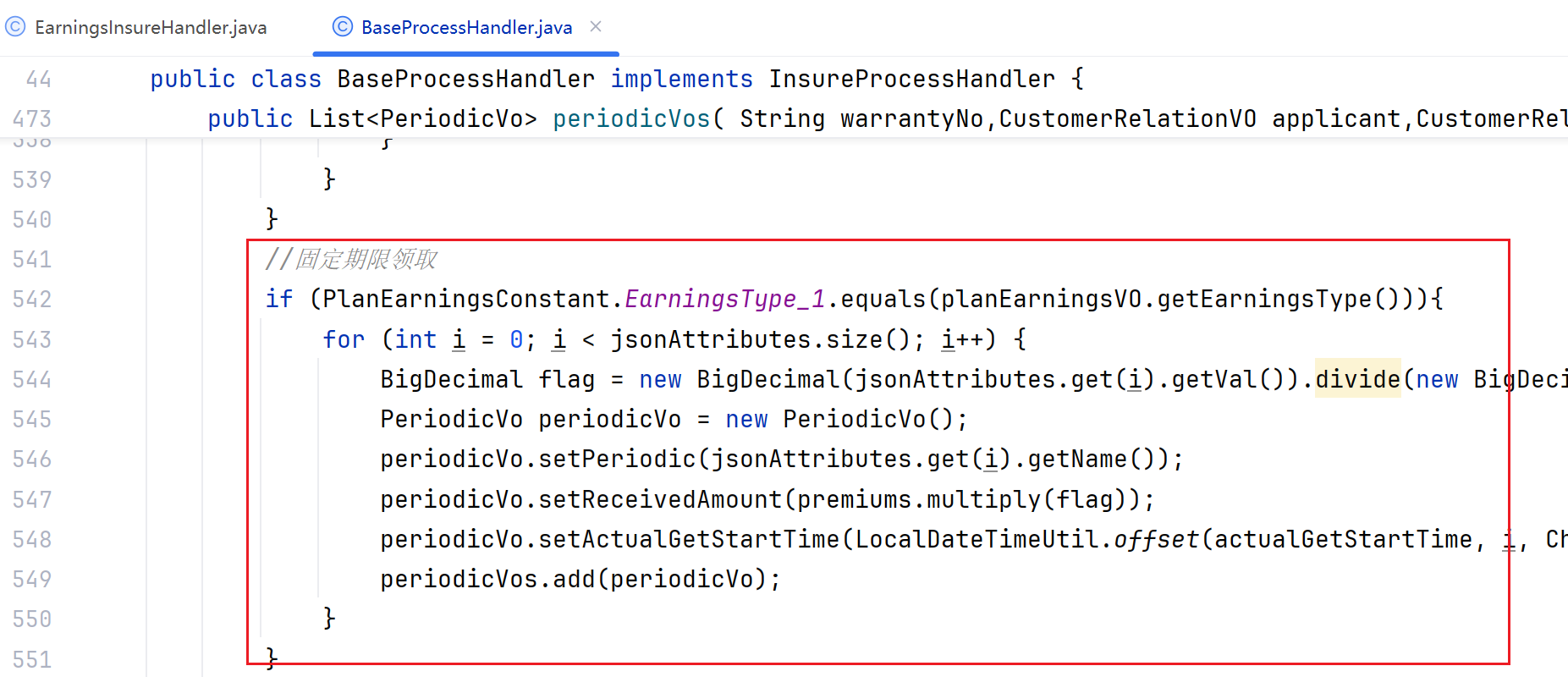
* 4、com.itheima.sfbx.insurance.handler.impl.BaseProcessHandler#earningsCompute 中的第402 - 429行



* 5、com.itheima.sfbx.insurance.handler.impl.BaseProcessHandler#periodicVos 中的第484行终身领取



* 6、com.itheima.sfbx.insurance.handler.impl.BaseProcessHandler#periodicVos 中的第541行固定领取



人员：每组最多2个人上台演讲