## 人工客服对接流程

1. 接收客户需求：

客户将设计需求提交给人工客服，并陈述自己的期望和要求。

1. 确认需求：

客服收到客户的需求后，需要再次确认并核对客户的期望，以确保自己对需求的理解是准确的。

1. 分配设计师：

客服将需求和客户的期望传达给设计师。由于不同的客户需要不同类型的设计方案，因此，客服应根据设计类型、客户要求、设计师风格等因素来匹配合适的设计师。

1. 沟通需求细节：

设计师与客服合作，分析客户的设计需求和期望。设计师开始考虑如何最好地实现客户的要求，同时，客服在整个项目周期内始终与客户保持联系，收集反馈和评论。在这个过程中，客服要始终保持开放性和透明度，并在保护客户和设计师的隐私的同时，不断努力满足客户的需求。

1. 提供初步设计方案：

设计师提交初步的设计方案给客服，客户则接收到来自客服的反馈。客服向客户演示设计方案，并询问是否满意。如果客户有任何意见或建议，客服将把意见反馈给设计师，并调整设计方案。客户只需要和客服沟通即可，不需要与设计师直接交流。

1. 审核设计方案：

客服审核修改后的方案，并根据客户的反馈，将改动反馈给设计师。设计师将继续修改并最终确定设计方案。

1. 交付设计方案：

设计师将设计方案交给客服。客服将设计方案展示给客户，以确认设计方案是否符合客户的要求和期望，同时解决可能存在的问题。如果客户对设计方案不满意，则客服将重新联系设计师，以制定针对新需求的方案。

1. 交付和支付：

在设计方案最终确定后，客户按协议支付费用，然后才能接收设计源文件。

**成熟的对接流程**

明确合作意向 → 合作合同（含时间表/付款安排/交付内容/违约细则）→ 预付款 →初稿 → 中期款 → 改稿 → 定稿 → 尾款 → 交付设计源文件

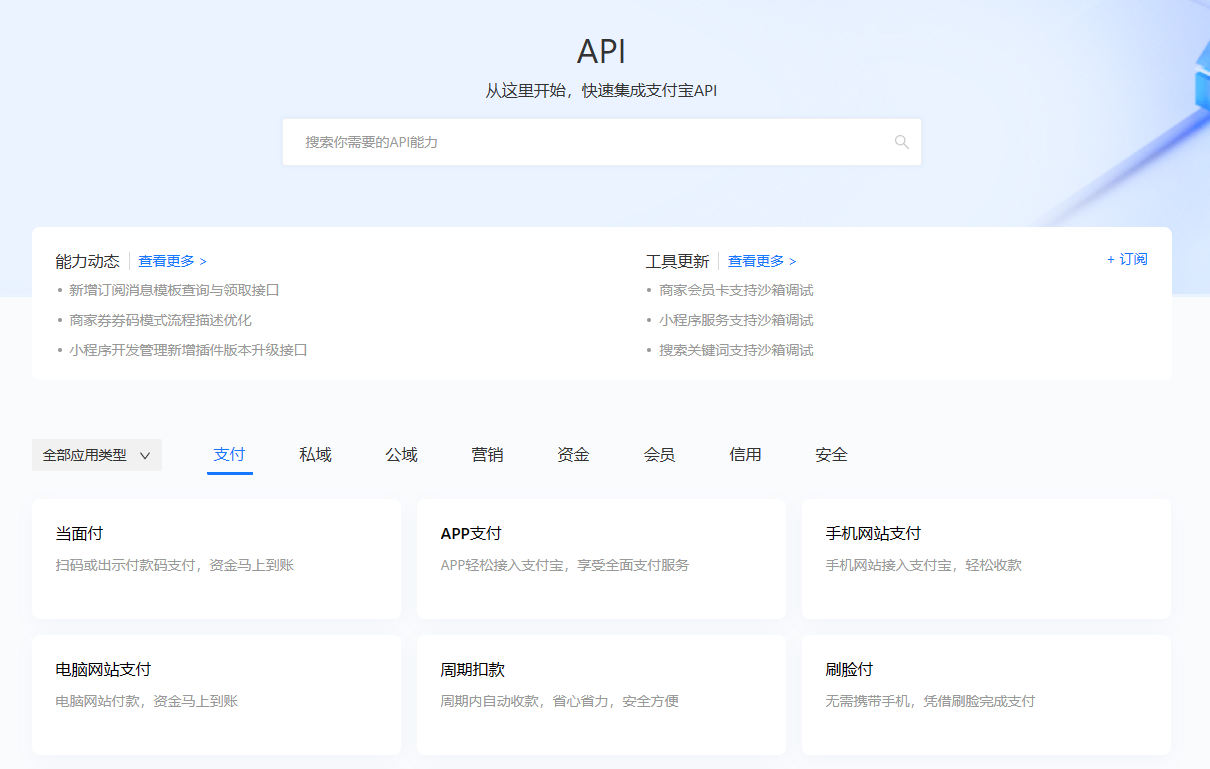
不收到订金不开始做，订金一般收取总费用的30%左右。没结尾款不交付源文件

**交易支付**

实现交易支付功能一般需要考虑以下几个方面：安全性、可实现性和性能

常见的技术方案为调用第三方支付服务：使用支付宝、微信支付、Paypal等第三方支付服务，这种方式相对简单，只需要调用相应的API接口即可，因为支付流程由第三方支付服务处理，开发不需要考虑支付的流程，只需要验证支付结果即可

下图为[支付宝开放平台 (alipay.com)](https://open.alipay.com/api)



采用第三方支付服务

如果采用第三方支付服务，开发者可以使用相应支付服务提供的开发者工具包或API接口来进行开发。不同的支付服务提供商可能会提供不同的工具包或API接口，但是一般来说，开发需要掌握以下技术：

HTTP和HTTPS协议：支付服务提供商的API接口通常使用HTTP或者HTTPS协议来传输数据，开发者需要了解如何使用这些协议来发送请求和接收响应。

RESTful API设计：大多数支付服务提供商的API接口是基于RESTful架构的，开发者需要了解如何设计RESTful API，包括如何定义资源、状态码、请求方法、认证方式等。

OAuth2认证：支付服务提供商通常需要开发者进行OAuth2认证，以获得对API接口的访问权限，开发者需要了解如何进行OAuth2授权码流程或者客户端凭证流程等认证方式。

数据加密和签名：为了保证支付过程的安全性，支付服务提供商一般会对传输的信息进行加密和签名。开发者需要了解如何对数据进行加密和签名，并且如何进行解密和验签。

Web框架和数据库操作：为了实现支付功能，开发者可能需要使用Web框架和数据库操作工具，常见的Web框架有Spring、Flask、Django等，常见的数据库操作工具有JDBC、Hibernate、Mybatis等。

采用JAVA进行交易支付处理

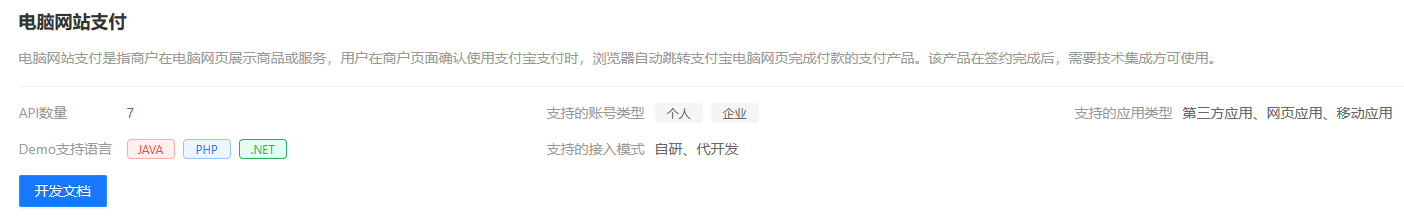
如果使用Java开发支付功能，以下为大致的开发流程：

**1．选择支付服务提供商**：根据自己的需求选择一个支付服务提供商，例如支付宝、微信支付等。

**2．注册开发者账号：**在支付服务提供商的官网上注册一个开发者账号，启用开发者模式获取相应的AppID、AppKey、AppSecret、商户号等信息。



**3．熟悉API文档：**在支付服务提供商的官网上查看API文档，了解不同支付模式下相应的API接口和参数信息，理解API请求和响应的格式。



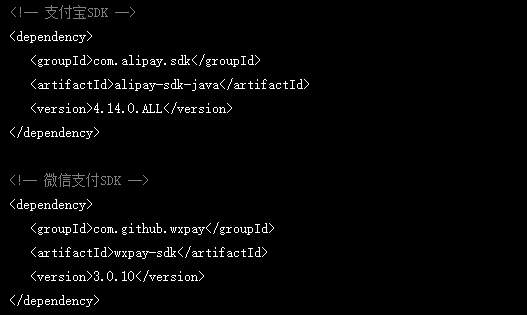


**4．搭建开发环境：**在自己的开发环境中搭建相应的Java开发环境，例如Java Development Kit（JDK）或使用集成开发工具，例如Eclipse、IntelliJ IDEA等。

**5．集成支付SDK：**根据支付服务提供商提供的SDK和API接口，选择合适的Java库和依赖，完成SDK的集成，并且根据SDK提供的接口实现支付功能。

*注：SDK，即Software Development Kit，是一个包含API、工具、文档等一系列开发资源的工具包，它可以帮助开发者更快速、更方便地使用第三方服务的功能。在集成支付功能时，支付服务提供商会提供相应的SDK和API接口，开发者可以将SDK整合进自己的应用程序中，直接调用SDK提供的API接口来实现支付功能。*

以下给出Java集成支付SDK大致的实现过程：

1. 在开发环境中安装Maven或Gradle等依赖管理工具，并且创建一个新的Java项目。
2. 在pom.xml（或build.gradle）文件中添加相关依赖，例如：
3. 按照支付服务提供商的说明，编写Java代码，调用SDK提供的API接口，实现支付功能。例如，支付宝提供的Java SDK可以参考以下代码：
4. 运行Java程序，在本地环境或者线上环境中测试支付功能，确保支付功能的稳定性和安全性。

**6．数据加密和签名：**根据支付服务提供商提供的安全规范，对支付数据进行加密和签名处理，保证支付过程中数据的安全性。

以下给出大致流程：

1. 加密

在支付请求数据中敏感信息需要进行加密处理，采用对称加密算法，例如AES、DES等，通过对称算法将信息用密钥加密，可以保证信息在传输过程中不被截获并泄漏。在数据传送之前，需要商户后台使用相对应的算法对交易信息进行加密。加密工作如下：

1. 将商户请求报文中的所有内容整理在一起，例如商户订单号、金额、商品信息等，形成明文串。
2. 使用加密算法对明文串进行加密处理，生成密文串。
3. 将密文串添加到商户请求报文中相应的字段中。
4. 将商户请求报文发送给支付宝支付网关。
5. 签名

为了保证支付请求数据在传输过程中不被篡改，需要对数据进行签名处理，采用非对称加密算法，例如RSA、DSA等，确保报文在完整的基础上进行签名。签名工作如下：

1. 将商户请求报文中的所有内容整理在一起，例如商户订单号、金额、商品信息等，形成字符串型的原始内容串。
2. 使用商户私钥进行签名处理，生成签名。
3. 将签名添加到商户请求报文中相应的字段中。
4. 将商户请求报文发送给支付宝支付网关。
5. 验签

当支付宝支付网关收到商户请求报文并返回支付结果后，需要商户后台验证返回的应答是否合法，以保证数据的一致性。验签工作如下：

1. 获取请求报文中的所有参数，包括支付宝返回的支付结果、加密字符串、商户号等信息。
2. 将这些参数按照约定的顺序进行排序，并将参数进行拼接组成字符串。
3. 使用支付宝公钥对该字符串进行解密。
4. 将解密后的字符串拆分为键值对，并判断支付结果是否正确。

**7．接口测试和调试：**在本地搭建相应的测试环境，对接口进行测试和调试，保证支付功能正常运行。

**8．部署线上系统：**将支付功能集成到线上系统中，在线上环境中进行相应的测试和验证，确保支付功能的稳定性和安全性。