数字遗产交付者

（Digital Inheritance Deliverer）

项目介绍文档

时间：2023年10月14日

目录

1. 项目介绍

1.1. 项目背景

随着互联网技术的不断发展，数字遗产成为了越来越多人的关注焦点然而，由于缺乏有效的保护措施和传承机制，数字遗产的保护意识仍然存在不足。数字遗产包括个人数据、照片、视频、音频、文档等数字文件，以及网络账户、虚拟货币、社交账号等虚拟财产。这些遗产不仅代表着个人的重要资产，也承载着许多人的情感和回忆需求

数字资产/遗产保护意识不足与传承问题：

1. 缺乏相关法律法规：目前，我国在数字资产保护方面的法律法规尚不完善，导致许多人在面临数字遗产继承问题时无法可

依。

2. 缺乏教育宣传：数字遗产保护意识在公众中缺乏普及，许多人对数字遗产的认识不足，缺乏保护和传承的意识和能力
3. 数字遗产类别众多： 随着数字时代的到来，越来越多的人拥有数字资产，如加密货币、社交媒体账户、电子邮件和云存储。如果死者未能妥善考虑这些数字资产的分配，可能会导致问题。

数字遗产的继承问题涉及到多个方面，如法律、技术、情感等。如何确保数字遗产的合法继承，避免纠纷和矛盾，是当前亟待解决的问题。

现实资产/遗产未能妥善继承：

1. 无效或缺乏遗嘱： 很多人未能在生前制定有效的遗嘱，或者干脆没有遗嘱。这会导致遗产分配时根据法定继承法进行，而不是按照死者的真实意愿。
2. 家庭纷争： 家庭成员之间可能存在争议，导致无法达成共识。这些争议可能源于遗嘱的内容、资产分配或家庭关系紧张。
3. 多次婚姻： 如果死者曾经有多次婚姻，家庭结构变得复杂，涉及多个配偶和子女。这可能导致继承权的争议，特别是如果没有明确的遗嘱规定。
4. 执行问题： 遗产分配可能受到执行问题的困扰。执行遗嘱的人（执行者）可能不诚实或不胜任，导致遗产分配不按照死者意愿进行。

5. 遗产税： 部分国家/地区对继承的遗产征收税款，这可能减少继承人最终获得的遗产金额。死者未能采取适当的财务规划措施来降低遗产税可能导致不按意愿分配的问题。

现实社会中遗嘱等和继承相关问题将牵扯到各方各面，步骤繁琐，容易引发人际关系紧张等问题。

相关法律背景：

1.法律默认继承顺序：在缺乏明确遗嘱的情况下，财产通常会按照法律规定的继承链条传递。这可能会导致不同的财产分配，与去世者的意愿不符。这样的情况可能引发家庭内的争议和纠纷，甚至导致亲戚之间的关系破裂。

2.处理失踪人口存在缺陷：当人口失踪并被宣告死亡后，存在较长的时间（2-4年）间隔，期间失踪人的财产可能变得无人管理，容易被不法分子滥用。这使得财产的合法继承和维护失踪人的意愿变得非常复杂。

相关法律见下页：

2、我国《民法通则》第21条规定：

失踪人的财产由他的配偶、父母、成年子女或者关系密切的其他亲属、朋友代管。代管有争议的，或者没有以上规定的人或者以上规定的人无能力代管的，由人民法院指定的人代管。

即失踪人的财产由他人代为保管，并只能为失踪人的利益而妥善处理(如清偿债务等)，否则将承担相应的赔偿责任。

3、如果被宣告死亡，在法律上等同于自然死亡，依法对其财产，则作为遗产开始继承。

依据《继承法》规定，继承开始后，如果遗嘱有效，则按遗嘱继承。否则开始法定继承，继承法第10条规定，遗产按照下列顺序继承：

第一顺序：配偶、子女、父母。

第二顺序：兄弟姐妹、祖父母、外祖父母。

一、宣告失踪后多长时间可宣告死亡

《中华人民共和国民法典》第四十六条规定，公民有下列情形之一的，利害关系人可以向人民法院申请宣告他死亡：下落不明满四年的；因意外事故下落不明，从事故发生之日起满二年的。战争期间下落不明的，下落不明的时间从战争结束之日起计算。

二、宣告死亡申请人有顺序吗

宣告死亡申请人有顺序限制，前顺序人不同的，后顺序人不能申请宣告死亡。

须经利害关系人申请。申请人包括：

（一）配偶；（二）父母、子女；（三）兄弟姐妹、祖父母、外祖父母、孙子女、外孙子女；（四）其他有民事权利义务关系的人。必须按此顺序申请，顺序在先的申请人有排他效力，有在先顺序的排除在后顺序，同顺序的权力平等。

数字遗产交付者：责任与挑战

随着互联网的发展，个人数据和隐私、数字资产成为了人们日益关注的问题。互联网的普及使得个人信息更容易被泄露。个人数据可能被非法收集、滥用。在数字资产方面，广泛存在盗窃、欺诈等问题。作为数字资产/遗产，也存在继承不当，未能按照死者意愿继承等问题。

数字遗产交付者面临着许多挑战，也承担着相应的责任。

随着互联网用户的增加，数字遗产的继承问题也逐渐凸显。用户的数字资产（如数字货币、社交媒体账户、云存储文件等）可能因用户离世而无人继承，导致数字遗产的流失。

数字遗产交付者需要建立用户的信任，以确保用户数据的隐私和安全。数字遗产交付者作为连接用户与数字遗产的桥梁，需要承担起保护用户数据和隐私的责任。需要了解数字遗产的特点和价值，制定合理的保护措施，确保数字遗产的安全和完整性。同时，还需要积极探索合适的传承方式，确保数字遗产能够得到合法、公正的继承。同时，数字遗产交付者需要关注法律、技术、道德等多个方面的问题，以确保用户数字遗产的安全和继承。为了应对这些挑战，数字遗产交付者需要不断创新和完善服务，以建立用户的信任和满意度。

1.2. 典型社会问题和痛点

基本问题： 去世者未能及时制定遗嘱，但在生前希望规划遗产分配。

矛盾问题： 去世者想要合理分配遗产，包括制定遗嘱和公证，但可能会因潜在的不公平分配而受到不当待遇。

举例：父亲公开的遗嘱给予哥哥100万元，弟弟1万元，父亲可能遭到弟弟的不公平对待。我们的项目有助于维护去世者的遗愿隐私。

升级问题： 抵御不法分子对法律漏洞的攻击和利用，捍卫去世者的意愿，确保财产能够顺利继承。

举例： 在电影《消失的她》中，男主角A为了继承遗产与女主角B骗婚。后来，A在东南亚杀害了B并将其尸体扔入海中。由于B的亲朋好友无法找到她的尸体，即使怀疑A，法律也无法制裁A。B现在被视为失踪，法律在没有遗嘱的情况下，通常将配偶定为第一继承顺序。幸运的是，电影中B的朋友最终找到了她的尸体，并制裁了A。但如果找不到尸体，结局将会十分痛苦。如果B使用了我们的项目并设定了区块链遗嘱，那么就可以减轻损失，甚至避免这场悲剧的发生。

1.3. 项目功能

我们将根据用户的意愿，通过区块链技术，确保在发生意外情况（尤其是去世）时，将用户上传的数字财产/遗嘱交付给用户已经绑定的人。将用户的遗产交付给用户希望交付给的人。

本系统（数字遗产交付者）中用户均匿名，支持继承关系的绑定（通过互通地址与公钥的方式），在绑定关系的基础上，可以实现遗产内容的定向编写/制定，在遗产制定者账号活跃期过后，被分配到遗产的人可以对遗产进行解密。为保护某个帐户免受类似DDOS的攻击，我们提供了屏蔽每个地址的功能。

我们的项目可以在本地打开多个网页并登录不同的账号，实现模拟的遗产继承全过程。

我们的系统/项目有以下的技术特点：

- 基于区块链技术支持
- 由智能合约保障安全
- 基于密码学技术加密遗产信息
- 可支持加密货币
- 去中心化系统，不受他人控制

我们的系统/项目实现了以下几个目标：

✔ 简便的去中心化系统：我们提供了一个允许人们提前制定遗嘱的去中心化平台

因为当人们发生意外（尤其是去世）时，已经来不及制定遗嘱。通过区块链的去中心化特性，用户的遗产分配得以保障，不受单一实体的控制。

✔ 隐私保护：我们确保在去世前不公开用户的资产分配。

避免因公开的不均等分配而受到不公平现实的对待。我们使用智能合约来确保遗产交付的安全性和可靠性。使用密码学技术对遗产信息进行加密，以保护用户的隐私。

✔ 财产继承保障：我们保护逝者的愿望，实现财产的顺利继承。

抵御不法分子攻击和利用法律漏洞的行为。允许用户使用加密货币进行遗产分配和交付（例如上传加密货币账号密码的方式），提供更多的灵活性。

2. 技术开发方案

2.1. 技术架构

前端技术：

利用 Bootstrap 和 Sass 实施前端界面设计，以确保用户界面在外观和响应性方面达到最佳标准。

JavaScript 实现前端逻辑，处理用户界面的各种操作，用到了web3库与以太坊网络进行通信，使其能够与区块链互动。

区块链技术：

Solidity 是我们选择的智能合约编程语言，专用于定义合约的行为和规则。主要用于管理用户信息、关系绑定和数字遗产的信息，提供了对这些数据的存储、获取和操作功能。需要特别强调的是，合约还具备根据具体需求添加更多功能和安全性检查的灵活性。

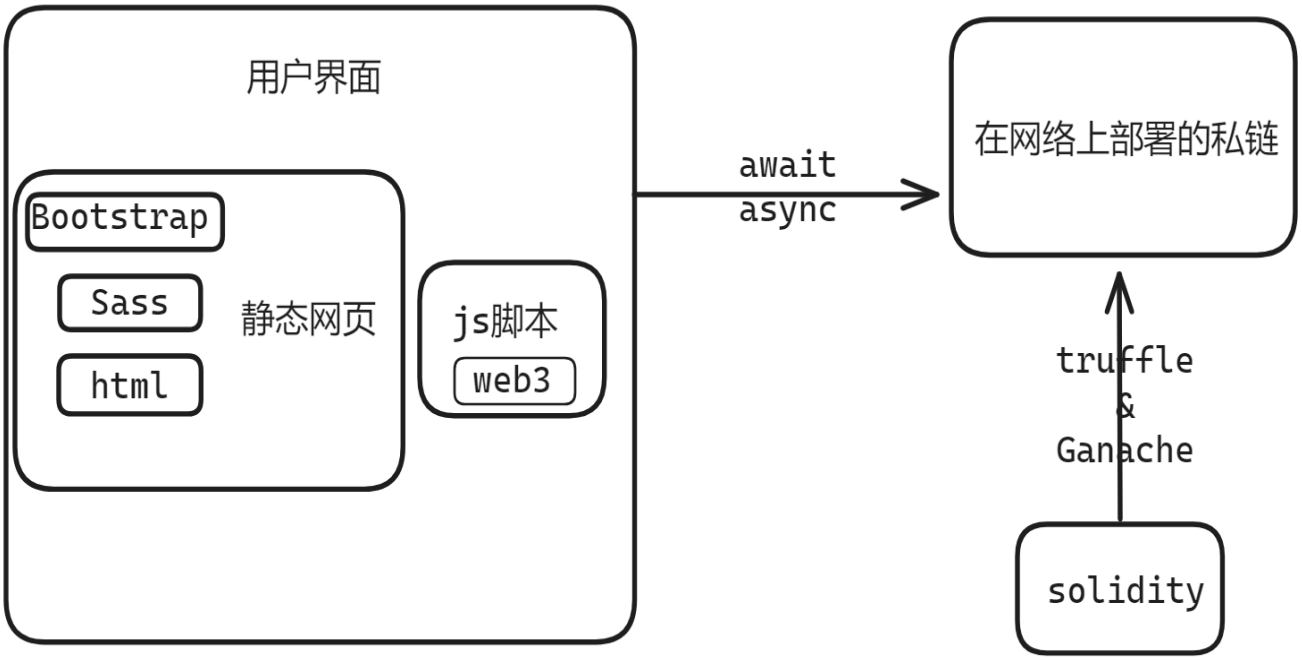
Truffle 框架在合约的部署、测试和开发工作流程方面发挥着关键作用，确保合约的有效性和可维护性。

Ganache 作为区块链模拟器，为我们提供了在开发和测试阶段快速迭代和调试合约的能力，同时避免消耗实际以太坊网络资源。这对于优化合约的性能和功能非常重要。

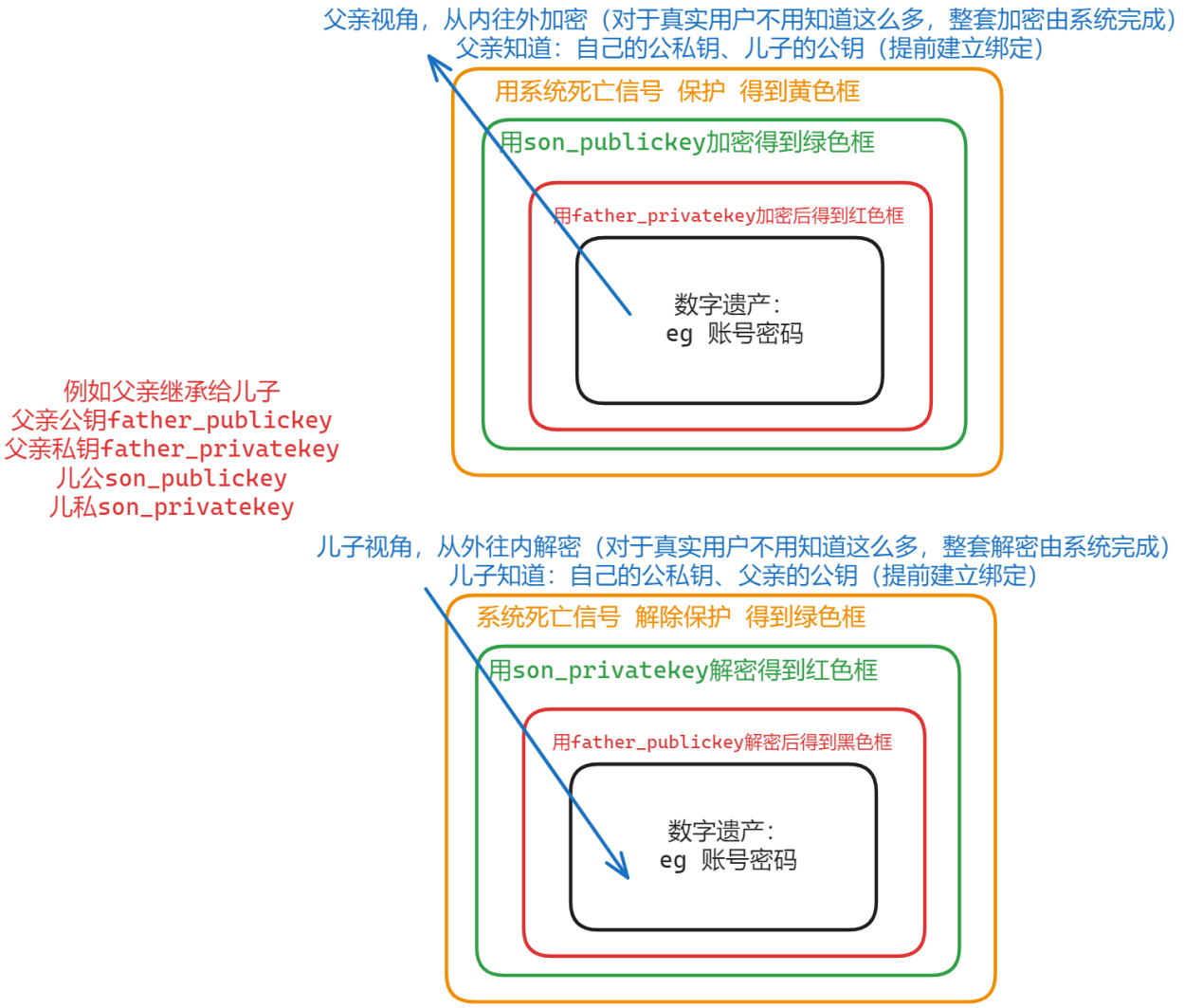
2.2. 项目开发流程

- 需求和架构定义
- 前端界面开发，包括Bootstrap、Sass的集成
- Js与前端界面的结合
- 智能合约的开发和测试
- 使用 Web3.js 库来与局域网上的私链进行通信和交互
- 使用Truffle部署智能合约到Ganache模拟器进行本地测试和调试
- 使用Webpack构建web应用程序
- 集成测试，确保前端与智能合约的协同工作

2.3. 项目系统结构



2.4. 用户遗产内容在链上的加密与解密



2.5. 技术介绍

Bootstrap：

简介：Bootstrap是一个流行的前端框架，由Twitter开发，用于构建现代、响应式的网站和web应用程序。提供了一套现成的HTML、CSS和JavaScript组件，可用于创建一致、美观的用户界面。支持响应式设计，使网站能够适应不同屏幕尺寸。

用途：用于构建网页的布局、按钮、表单、导航、模态框等常见UI元素，可大幅度提高开发效率。

Sass：

简介：Sass（Syntactically Awesome Stylesheets）是一种CSS的扩展语言，它提供了更多的功能和工具来组织和编写CSS代码。支持变量、嵌套规则、混合（Mixins）、函数等，使CSS更加模块化和可维护。

用途：用于编写复杂的CSS样式，尤其适用于大型项目，可以提高代码的可读性和可维护性。

JavaScript：

简介：JavaScript是一种高级的、解释性的编程语言，用于在web开发中添加交互性和动态功能。支持面向对象编程，事件驱动模型，异步编程等特性，使其非常适合前端开发。

用途：用于处理网页上的用户交互、验证表单、处理事件、与服务器通信等。

Solidity：

简介：Solidity是以太坊区块链上的智能合约编程语言，用于定义合约的行为和规则。支持面向对象编程，具有状态变量、函数、事件等构建块，可以用于创建去中心化应用的智能合约。

用途：主要用于以太坊区块链上编写智能合约，控制数字资产和去中心化应用的逻辑。

Truffle：

简介：Truffle是一个以太坊智能合约开发框架，用于构建、测试和部署智能合约。提供了一整套工具，包括编译、部署、测试和交互智能合约的能力，使以太坊开发更加高效。

用途：主要用于以太坊区块链上的DApp开发，有助于简化合约的管理和开发工作流程。

Ganache：

简介：Ganache是一个以太坊区块链模拟器，用于本地开发、测试和调试以太坊智能合约。模拟以太坊网络，不需要真正的以太坊币，提供快速开发和调试环境。

用途：用于在开发和测试阶段模拟以太坊网络，以确保智能合约的正确性和性能。

web3：

简介：web3.js是一种用于与以太坊区块链进行交互的JavaScript库。提供了一组API，用于从JavaScript代码中执行交易、查询区块链数据，与以太坊网络进行通信。

用途：主要用于DApp开发，使开发者能够创建与以太坊区块链进行交互的前端应用

2.6. Javascript接口设计

App 对象和初始化

1. `start`（无参数）：
   - 用途：初始化应用程序，包括连接到Web3提供程序，获取网络ID，并初始化智能合同。
2. `LogIn`（无参数）：
   - 用途：登录用户。需要用户输入帐户地址和密码，验证登录，如果成功，则绑定帐户并重定向到用户协议页面。

用户管理

1. `Deth`（无参数）：
   - 用途：标记用户为已故状态，将用户的状态更改为“死亡”。
2. `QueryMyBinding`（无参数）：
   - 用途：查询当前用户的绑定关系，列出与用户绑定的其他帐户以及它们的关系。
3. `AddBinding`（需要参数`Address`和`Relation`）：
   - 用途：创建新的绑定关系，需要另一个帐户的地址和关系类型。如果绑定成功，将更新用户的绑定关系。

遗产管理

1. `QueryMyAlloc`（无参数）：
   - 用途：查询用户分配的遗产，列出分配给用户的遗产ID、另一个帐户的地址、关系和数据。
2. `AddOneAlloc`（需要参数`Address`和`Content`）：
   - 用途：创建新的遗产分配，需要另一个帐户的地址和数据。如果分配成功，将添加新的遗产分配。
3. `DeleteOneAlloc`（需要参数`ID`）：
   - 用途：删除一个特定的遗产分配，需要提供遗产ID以删除指定的分配。
4. `CheckMyInher`（无参数）：
   - 用途：检查用户可以继承的遗产，列出遗产ID、赠与人地址、关系和状态。
5. `GetMyInher`（需要参数`ID`）：
   - 用途：获取特定遗产的详细信息，需要提供遗产ID，并在页面上显示数据。

用户界面导航

1. `ShowAddr`（无参数）：
   - 用途：在页面上显示用户的地址和公钥。
2. `Intro`（无参数）：
   - 用途：重定向到首页。
3. `User`（无参数）：
   - 用途：重定向到用户协议页面。
4. `Account`（无参数）：
   - 用途：重定向到用户账户页面。
5. `Designate`（无参数）：
   - 用途：重定向到遗产分配页面。
6. `Carry`（无参数）：
   - 用途：重定向到继承遗产页面。

隐私设置

1. `pingbi`（需要参数`pingbi`）：
   - 用途：屏蔽特定的绑定关系，需要提供要屏蔽的帐户地址，将该绑定关系标记为已忽略，不再显示。

2.7. Solidity接口设计

结构体

UserInfo（用户信息）

- isAlive（是否存活）：布尔类型
- 描述：用于指示用户是否存活。

ac（用户账户）

- account（账户地址）：地址类型
- index（索引）：无符号整数（uint256）

Binding（关系绑定）

- addr1（地址1）：地址类型
- addr2（地址2）：地址类型
- relationship（关系）：字符串
- isIgnored（是否忽略）：布尔类型

Inheritance（继承关系）

- addr1（地址1）：地址类型
- addr2（地址2）：地址类型
- data（数据）：字符串
- AllocID（分配ID）：无符号整数（uint256）

功能

用户管理

- 添加用户账户：
- 函数名称：addAc(address _account, uint256 _index)
- 获取账户地址：
- 函数名称：getacAccount(uint256 i)
- 获取账户索引：
- 函数名称：getacIndex(uint256 i)
- 获取用户账户总数：
- 函数名称：getacCount()
- 设置用户信息（是否存活）：
- 函数名称：setUserInfo(address addr, bool _isAlive)

- 获取用户信息（是否存活）：
- 函数名称：getUserInfo(address addr)

关系绑定

- 获取地址1（关系绑定）：
- 函数名称：getBindingAddress_1(uint256 index)
- 获取地址2（关系绑定）：
- 函数名称：getBindingAddress_2(uint256 index)
- 获取关系（关系绑定）：
- 函数名称：getBindingRelationship(uint256 index)
- 获取关系绑定总数：
- 函数名称：getBindingCount()
- 设置是否忽略（关系绑定）：
- 函数名称：setBindingIgnored(uint256 index, bool _isIgnored)
- 获取是否忽略（关系绑定）：
- 函数名称：getBindingIgnored(uint256 index)

继承关系

- 获取地址1（继承关系）：
- 函数名称：getInheritanceAddress_1(uint256 index)
- 获取地址2（继承关系）：
- 函数名称：getInheritanceAddress_2(uint256 index)
- 获取数据（继承关系）：

- 函数名称：getInheritanceData(uint256 index)
- 获取继承关系总数：
- 函数名称：getInheritanceCount()
- 获取分配ID（继承关系）：
- 函数名称：getInheritanceID(uint256 index)
- 添加绑定关系：
- 函数名称：addBinding(address addr1, address addr2, string memory relationship)
- 移除绑定关系：
- 函数名称：removeBinding(uint256 index)
- 添加继承关系：
- 函数名称：addInheritance(address addr1, address addr2, string memory data)
- 移除继承关系：
- 函数名称：removeInheritance(uint256 index)

合约初始化

- 获取初始化状态：
- 函数名称：get_initial()
- 设置初始化状态：
- 函数名称：set_initial(bool _isInitial)

3. 最终成果

数字遗产继承的全过程：

第一步：绑定（Bind）

描述：资产给予者与资产继承者之间通过地址与公钥，建立继承关系。

详细说明：在这一步中，用户将使用目标继承者的地址和公钥信息来建立继承关系。这样，可以明确指定将数字遗产传递给哪些人。

第二步：指定（Designate）

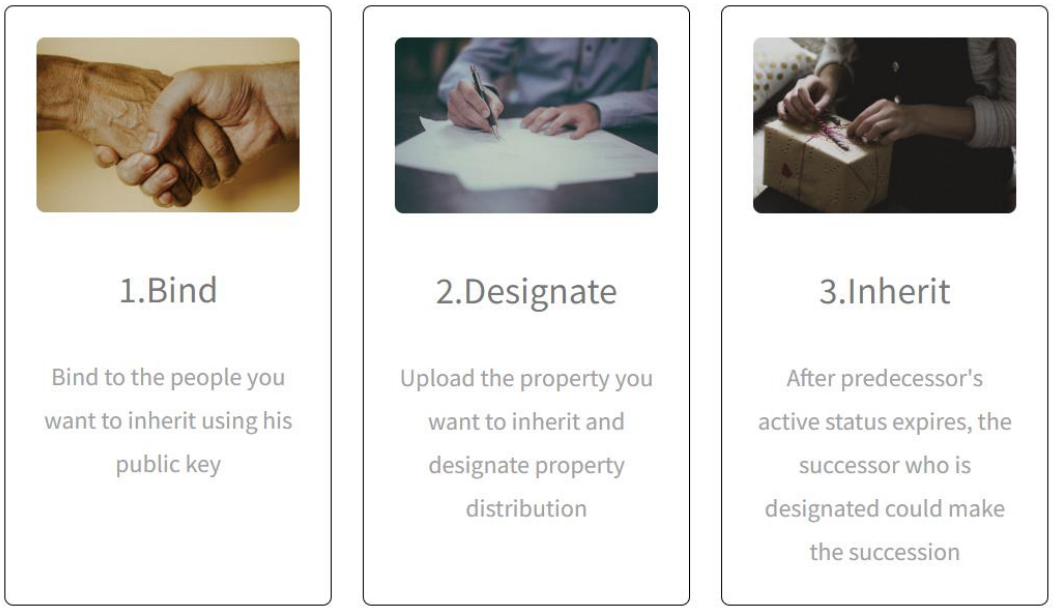
描述：上传希望继承的资产，指定资产的分发对象。

详细说明：在此步骤中，可以上传您希望继承的数字资产，并指定对谁分发这些资产。

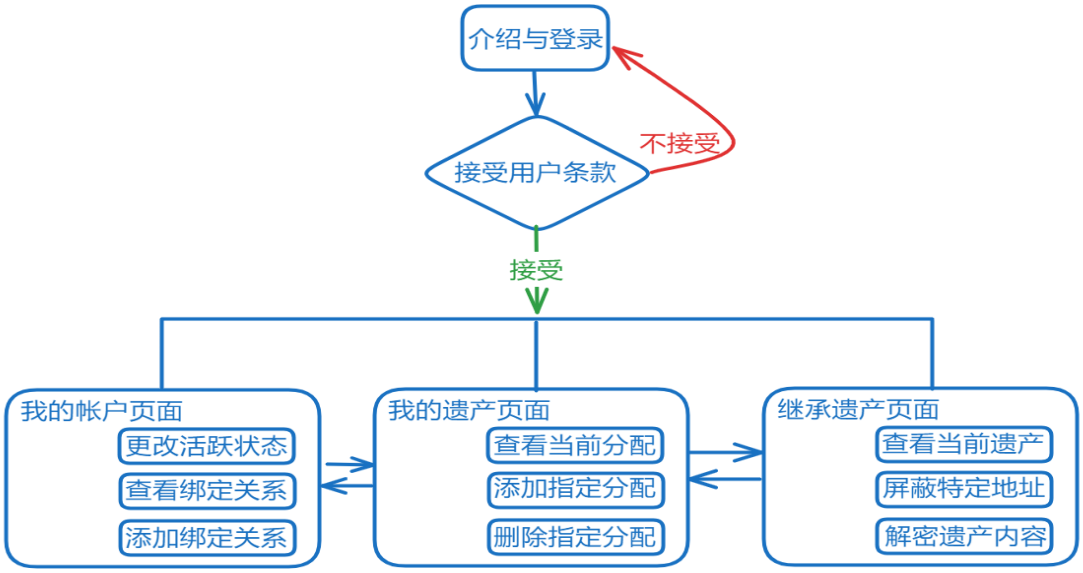
第三步：继承（Inherit）

描述：在资产给予者的帐户活跃状态结束后，指定的继承者可以进行继承。

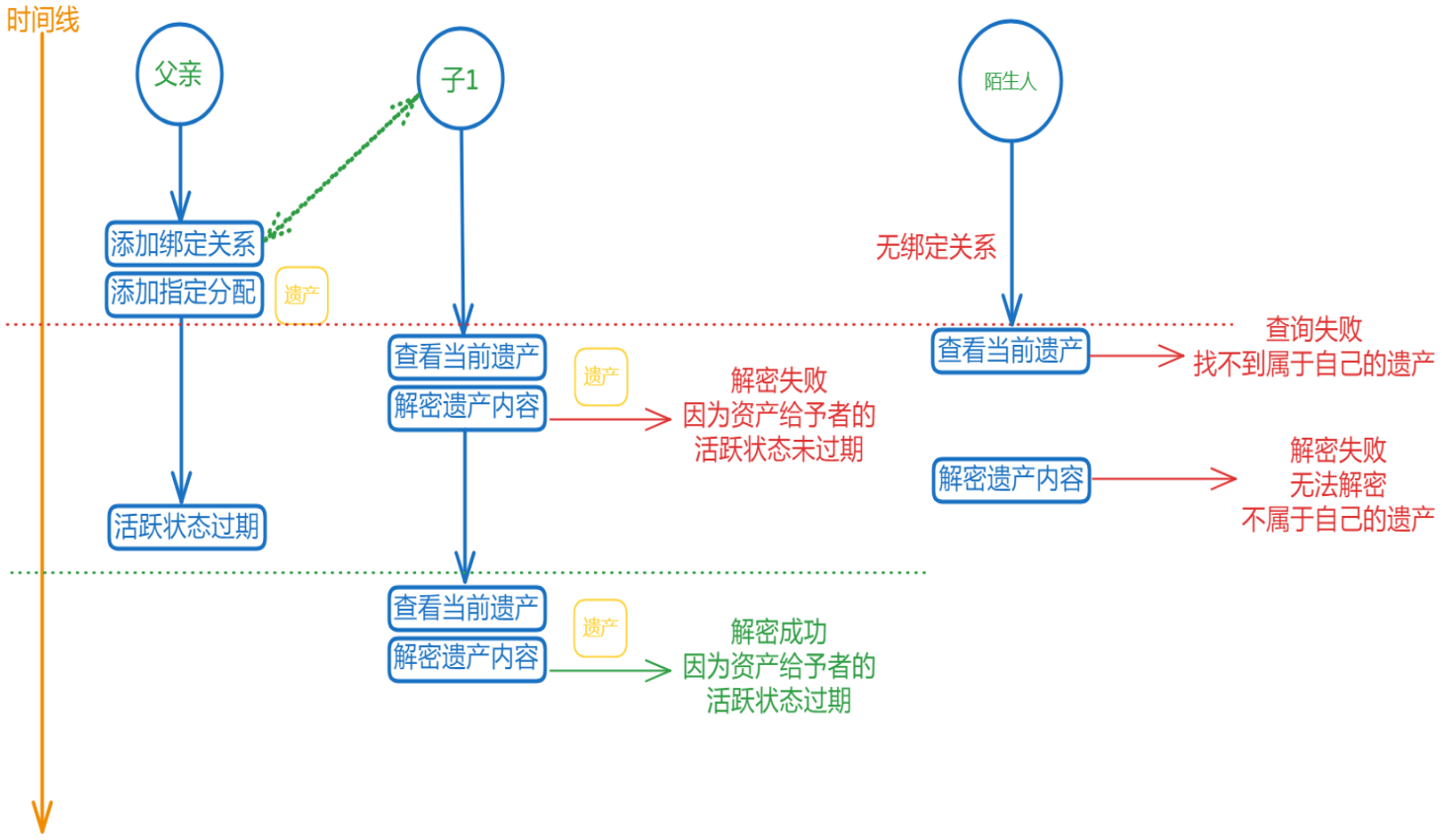
详细说明：当资产给予者的帐户活跃状态过期后，被指定为继承者的人将能够执行数字遗产的继承，



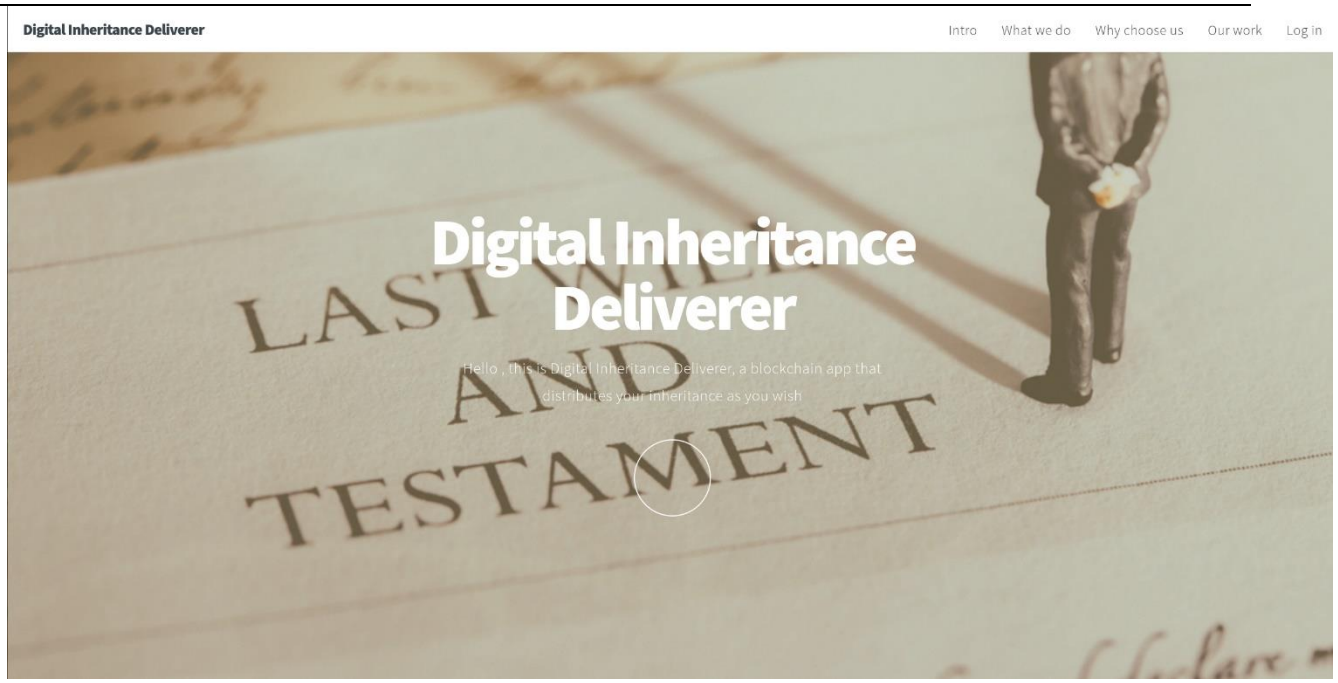
网页结构如下图所示：用户通过登录并同意用户条款之后，会来到三个页面，可以分别进行关系绑定，奉陪遗产和继承遗产



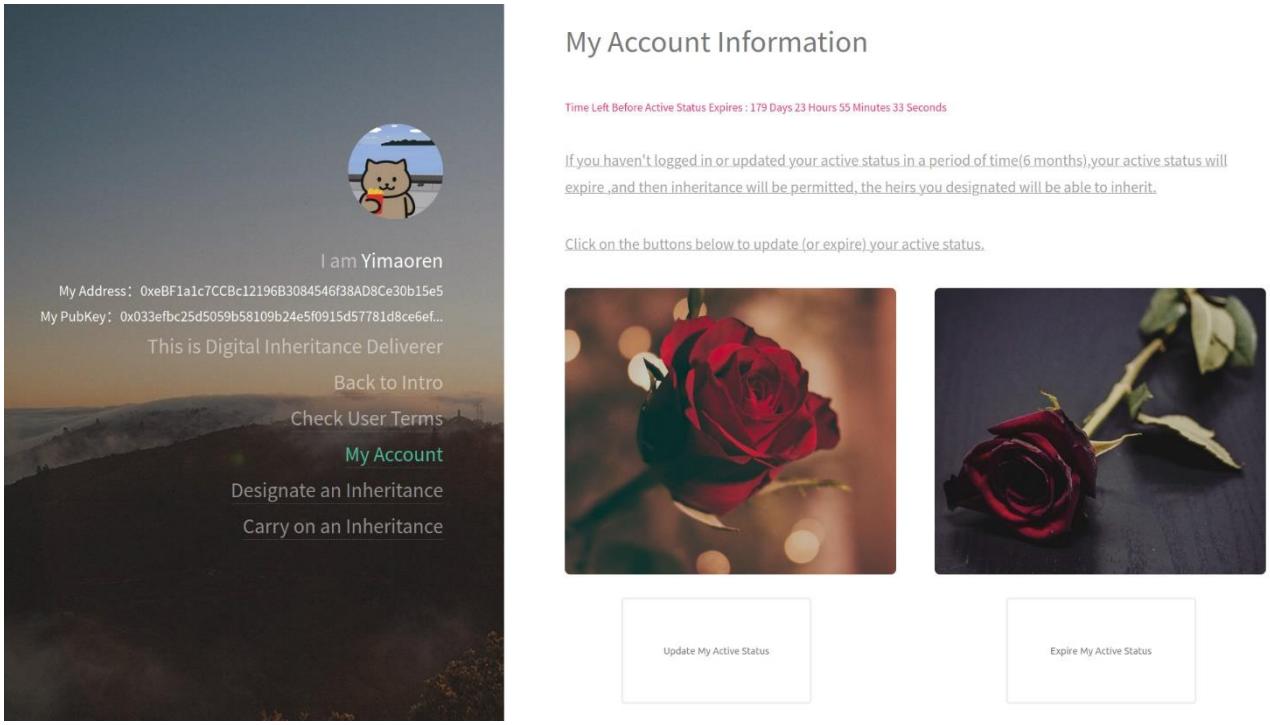
简单的遗产继承过程如图：例如父亲通过添加绑定关系、指定财产分配后，儿子可以查看到当前这份遗产的ID（不能解密此份遗产的内容），在父亲的活跃状态过期时，儿子便可以成功解密遗产内容，实现遗产继承。在此过程中，陌生人始终无法参与遗产的继承过程。


时间线
父亲
子1
陌生人
添加绑定关系
添加指定分配
遗产
无绑定关系
查看当前遗产
解密遗产内容
遗产
解密失败
因为资产给予者的
活跃状态未过期
查看当前遗产
查询失败
找不到属于自己的遗产
解密遗产内容
解密失败
无法解密
不属于自己的遗产
活跃状态过期
查看当前遗产
解密遗产内容
遗产
解密成功
因为资产给予者的
活跃状态过期

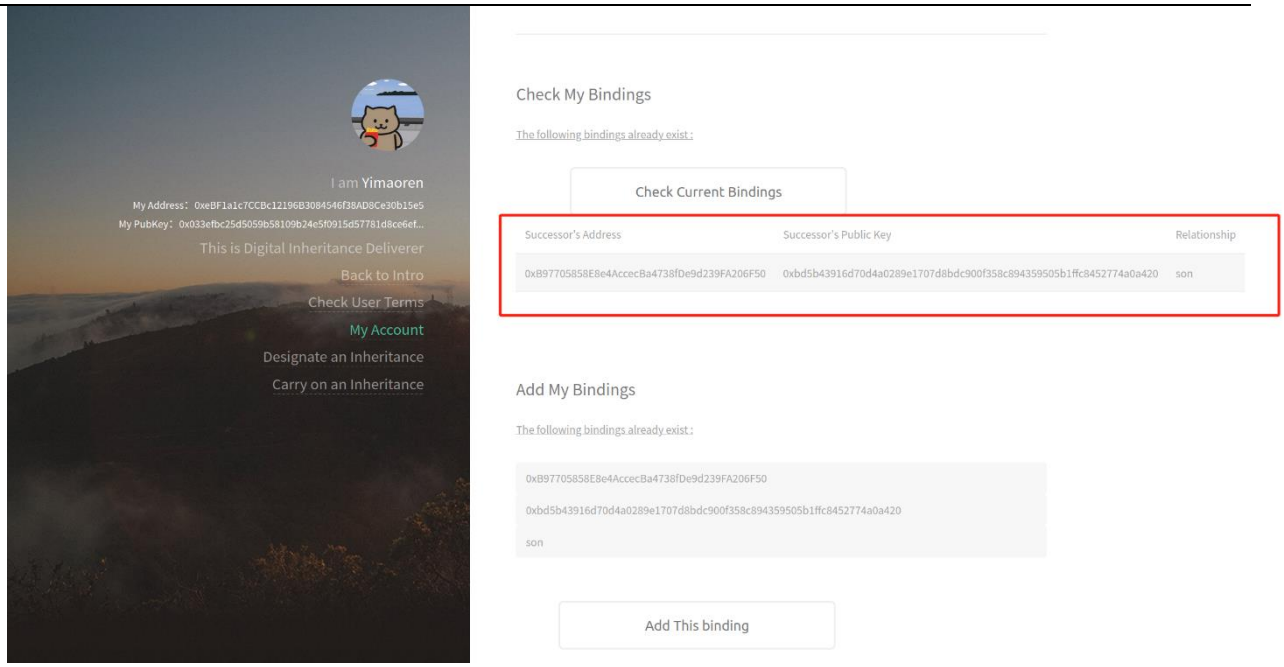
部分实现界面展示如下：



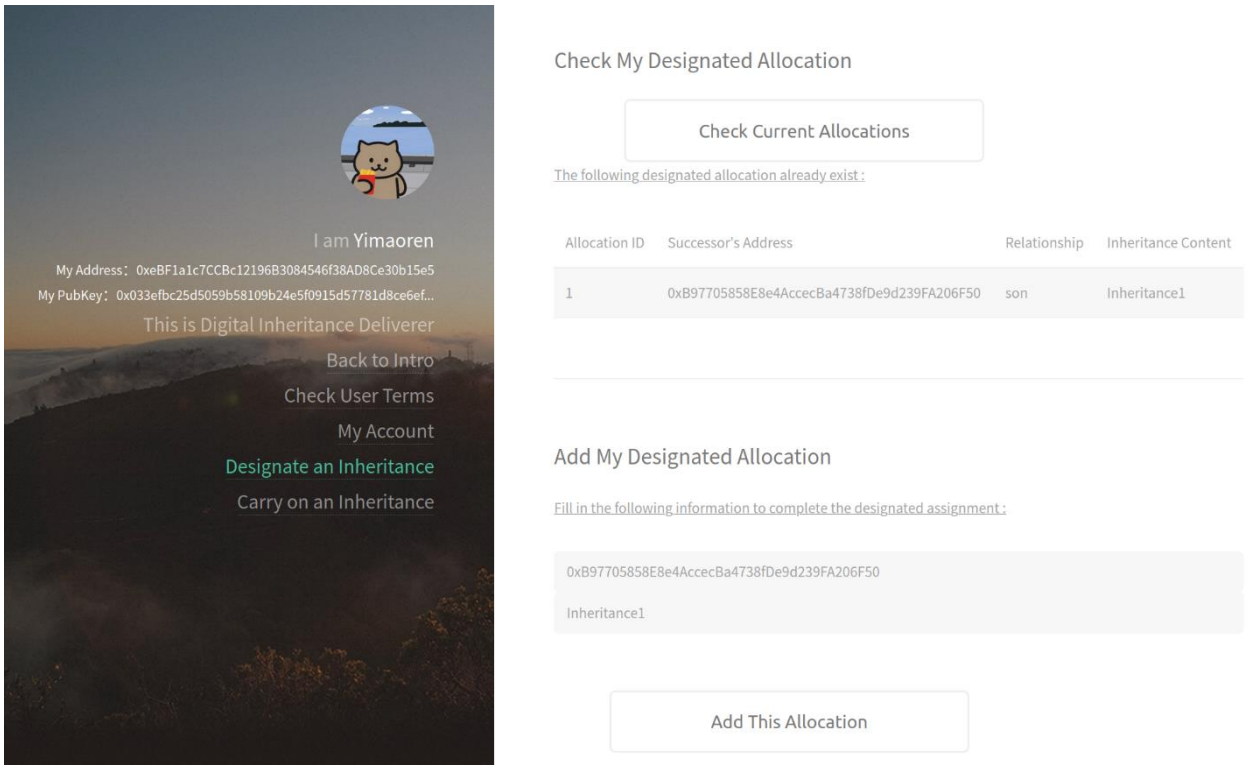
介绍与登录图（打开封面图）



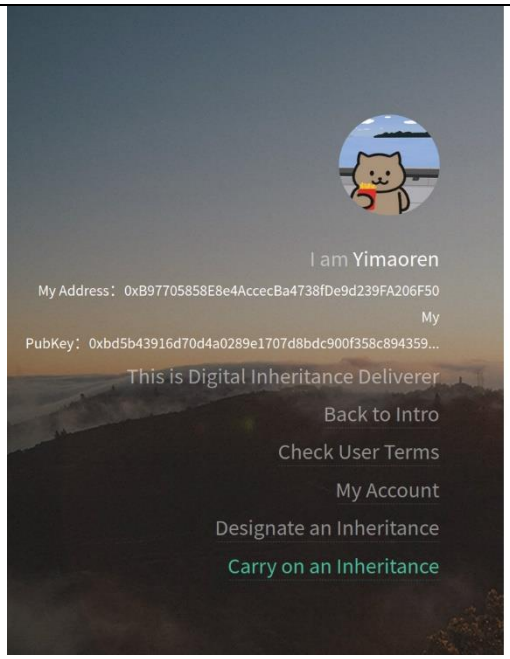
我的帐户界面（活跃状态更改）

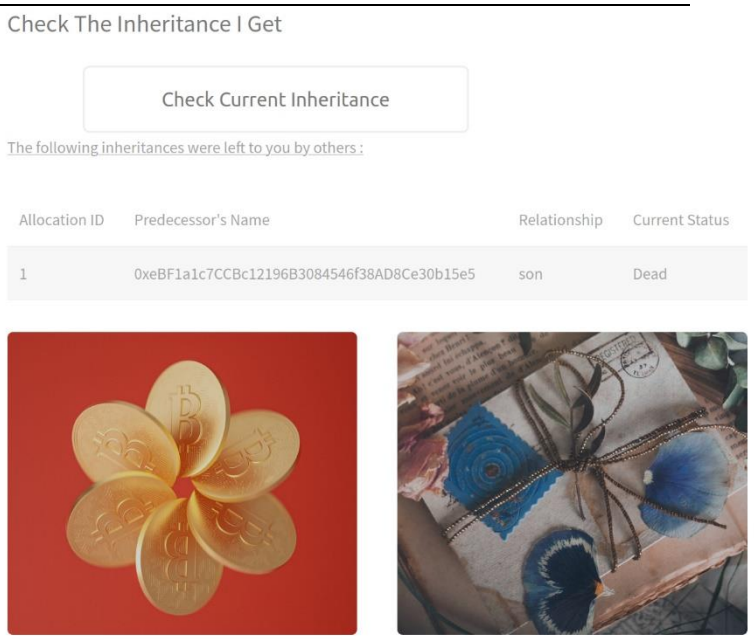


我的帐户界面（绑定关系的确立）

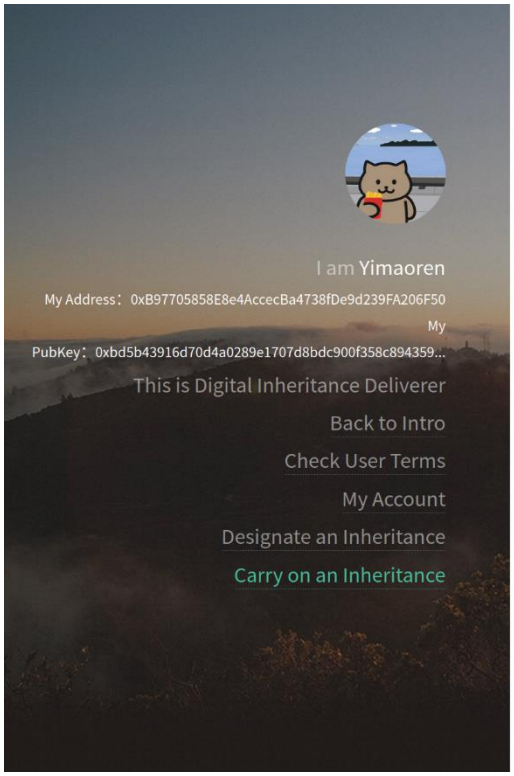


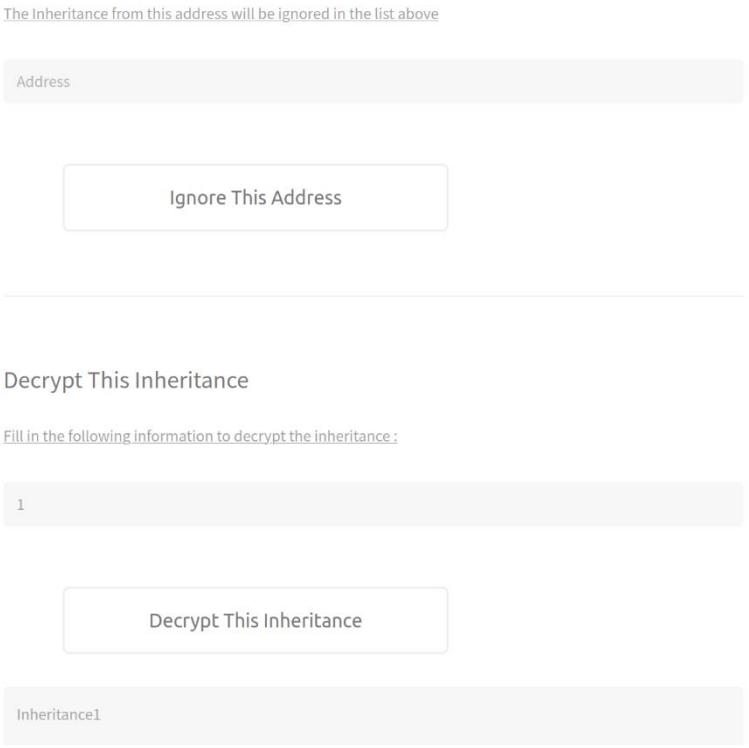
分配指定财产界面（添加并查看指定分配的财产/遗产）





继承财产界面（查看可以继承的遗产）





继承财产界面（屏蔽某个地址或是解密继承的遗产）

4. 后期改进思路

1. 用户教育和认知障碍： 区块链技术可能对一般用户来说相对陌生，项目需要提供广泛的用户教育和培训，以确保他们能够正确使用该系统。此外，新用户可能会感到不适应匿名性和加密技术。我们要建立广泛的用户教育，以解释数字遗产继承的概念、安全性和法律方面的信息。清楚地解释智能合约的作用和工作原理。
2. 社区建设： 建立一个积极的用户社区，以促进互动和支持。社区可以为用户提供帮助和答疑，同时增加项目的知名度。
3. 法律和监管合规性： 涉及财产分配和继承的项目通常会涉及到法律和监管问题。需要确保项目符合当地和国际法律，这可能需要法律顾问的协助。遗产继承可能涉及复杂的法律问题，需要妥善处理。
4. 用户友好性： 改进用户界面，使其更加友好和直观，以吸引更多的用户。考虑开发移动应用程序，以增加可访问性。确保用户界面是直观和易于导航的。提供清晰的指导，以帮助用户完成绑定、指定和继承步骤。考虑创建向导式的界面，以逐步引导用户完成整个过程。
5. 社交互动和支持： 创建社交互动渠道和支持体系，以帮助用户解决问题和获得支持。这有助于建立社区和用户满意度。
6. 法律合规和合约制定： 提供法律合规和遗嘱制定方面的服务或建议，以帮助用户了解法律要求，尤其是在涉及加密货币的情况下。
7. 多样性的加密货币支持： 考虑支持多种加密货币，以提高灵活性，因为不同用户可能喜欢使用不同的数字资产。这将提高项目的灵活

性，以满足各种用户需求。

8. 安全审计： 尽管项目强调了安全性，但仍然需要不断改进，特别是在加密和智能合约的方面。确保智能合约没有漏洞并且是安全的至关重要。定期进行智能合约和系统的安全审计，以确保没有潜在的漏洞或威胁。这有助于保护用户的数字财产。
9. 透明性： 提供透明度，允许用户在合理的范围内跟踪绑定、指定和继承的进展，适当看到一些不涉及隐私的信息，以增加用户信任感。
10. 多平台支持： 考虑开发移动应用程序，以使项目在不同平台上可访问。这将扩大用户基础。
11. 合作伙伴关系： 寻求合作伙伴关系，尤其是与法律和金融机构，以增加项目的信誉和可信度。
12. 定期更新和维护： 确保系统和智能合约得到定期更新和维护，以应对新的技术和安全挑战。

综上所述，数字遗产交付者项目具有很大的潜力，但迫切需要注重用户教育、安全性和法律合规性。通过不断改进用户体验和提高项目的可信度，这样才能实现项目的长期成功。