**课程报告**

|  |  |
| --- | --- |
| **课程名称：** | **企业软件项目实训** |
| **学生姓名：** | **张耀** |
| **学生学号：** | **201630666387** |
| **学生专业：** | **软件工程** |
| **开课学期：** | **2018-2019第二学期** |

**软件学院**

**2019年6月**

1. **区块链技术原理**

区块链是一个只能不断增长的记录列表，称为块，它们使用加密技术进行链接和保护；是一种分布式数据存储、点对点传输、共识机制、加密算法等计算机技术的新型应用模式。区块链涉及的知识体系甚广，包含金融知识、分布式体系、博弈论、密码学、数据库原理等。可见，区块链技术具有相当大的广度和深度，但究其本质，区块链技术并不是一种单一的、全新的技术，而是多种现有技术（如加密算法、P2P文件传输等）整合的结果，这些技术与数据库巧妙地组合在一起，形成了一种新的数据记录、传输、存储与呈现的方式。

区块链具有去中心化、防篡改、匿名性、开放性和自治性等特性。系统是去中心化的，由于使用分布式核算和存储，体系不存在中心化的管理机构，任意节点的权利和义务都是均等的，系统中的数据块由系统中具有维护功能的节点来共同维护。系统具有防篡改的能力，一旦信息经过验证并添加至区块链中，就会被永久的存储起来，除非能够同时控制系统中超过51%的节点，否则单个节点上对数据库的修改是无效的，因此区块链的数据稳定性和可靠性极高。系统具有匿名性，节点之间的交换遵循固定的算法，其数据交互是无需信任的，因此交易对方无须通过公开身份的方式让对方对自己产生信任，对信用的累积非常有帮助。系统是开放的，除了交易各方的私有信息被加密外，区块链的数据对所有人公开，任何人都可以通过公开的接口查询区块链数据和开发相关的应用，因此整个系统信息高度透明。系统具有自治性，区块链采用基于协商一致的规范和协议使得整个系统中的所有节点能够在去信任的环境自由安全的交换数据，使得对人的信息改成了对机器的信任，任何人为的干预都不起作用。

区块链的核心数据结构是区块，区块按时间顺序先后产生，每个区块包含一段时间内产生的交易数据；把相关的数据汇总再计算摘要，进行汇总的完整性正确性证明；每个区块计算摘要时，把前一个区块的摘要作为一个数据计算在内，构成了数据链；最新区块包含了所有数据链的完整性证明，整个链上的任何数据改动都会破坏数据链的相关性。

区块链的存储是分布式结构的，区块链中每一笔新交易的传播都采用分布式的结构，根据p2p网络层协议，消息从单个节点广播到网络上的其他节点。区块链让所有数据均存储在每个节点中，并实时更新。区块链的核心引擎是共识机制，这是一种多方协作机制，用于协调多参与方达成共同接受的唯一结果，且保证此过程难以被欺骗，且持续稳定运行。

在区块链系统中，所有权验证机制的基础是非对称加密算法。非对称加密是指加密和解密使用不同密钥的加密算法。因此在非对称加密算法中，存在一组密钥对，它们满足两个条件：对信息用其中一个密钥加密后，只能用另外一个密钥才能解开；不能根据任意的一个密钥计算出另一个密钥。所以即使区块链上的数据是共享的公开的，但是还是可以实现只有在数据拥有者授权的情况下才能访问私密数据。

区块链系统中引入一种脚本，称为智能合约。智能合约是在区块链上达成协议的一种方式，智能合约并没有超越传统合约的功能，但它能最小化达成协议所需的信任。智能合约不仅是一个可以自动执行的程序，它自己就是一个区块链参与者。它对接收到的交易进行回应，它可以接收和储存价值，也可以向外发送信息和价值。有了智能合约的存在，区块链技术可以使系统有机会去处理一些无法预见的交易模式，增加了技术的实用性。

区块链技术原理的来源可归纳为一个数学问题：拜占庭将军问题。拜占庭将军问题延伸到互联网生活中来，其内涵可概括为：在互联网大背景下，当需要与不熟悉的对手方进行价值交换活动时，人们如何才能防止不会被其中的恶意破坏者欺骗、迷惑从而做出错误的决策。进一步将拜占庭将军问题延伸到技术领域中来，其内涵可概括为：在缺少可信任的中央节点和可信任的通道的情况下，分布在网络中的各个节点应如何达成共识。区块链技术解决了闻名已久的拜占庭将军问题——它提供了一种无需信任单个节点、还能创建共识网络的方法。

1. **联盟链和公有链的异同**

联盟链是由某个群体内部指定多个预选的节点为记账人，每个块的生成由所有的预选节点共同决定，其他接入节点可以参与交易，但不过问记账过程，其他任何人可以通过该区块链开放的API进行限定查询。联盟链的各个节点通常有与之对应的实体机构组织，通过授权后才能加入与退出网络。各机构组织组成利益相关的联盟，共同维护区块链的健康运转。

公有链允许世界上任何个体或者团队发送交易，且交易能够获得该区块链的有效确认，任何人都可以参与共识过程。公有链上的各个节点可以自由加入和退出网络，并参加链上数据的读写，读写时以扁平的拓扑结构互联互通，网络中不存在任何中心化的服务端节点。像大家熟悉的比特币和以太坊，都是一种公有链。

联盟链和公有链都是使用区块链的技术，区别在于参与角色和权限不同。由于公有链的高度去中心化，其共识速度，吞吐都是比较低的。公有链也是有很多风险，如可能存在的暗网交易、洗钱及恐怖主义活动风险、监管缺位导致的消费者财务风险、无主体承担法律责任等。此外，现有的公有链也并非为大众所有。如比特币，前10大矿池算力合计超过93%；以太坊中，前10大矿池算力合计超过87%。

联盟链由于节点准入门槛较高，因此相较于公有链，去中心化程度较低。取而代之的是节点间共识速度和吞吐的提高。联盟链可实现对节点的监管，进行身份认证，去代币等。联盟链作为支持分布式商业的基础组件，更能满足分布式商业中的多方对等合作与合规有序发展要求。

1. **信任链的建立**

首先信任密码学算法。区块链是用算法达成信任的，其中最重要的算法之一，就是密码学。区块链中最基本的密码学应用是hash摘要、对称加密和非对称加密算法，以及相关的签名验签算法。

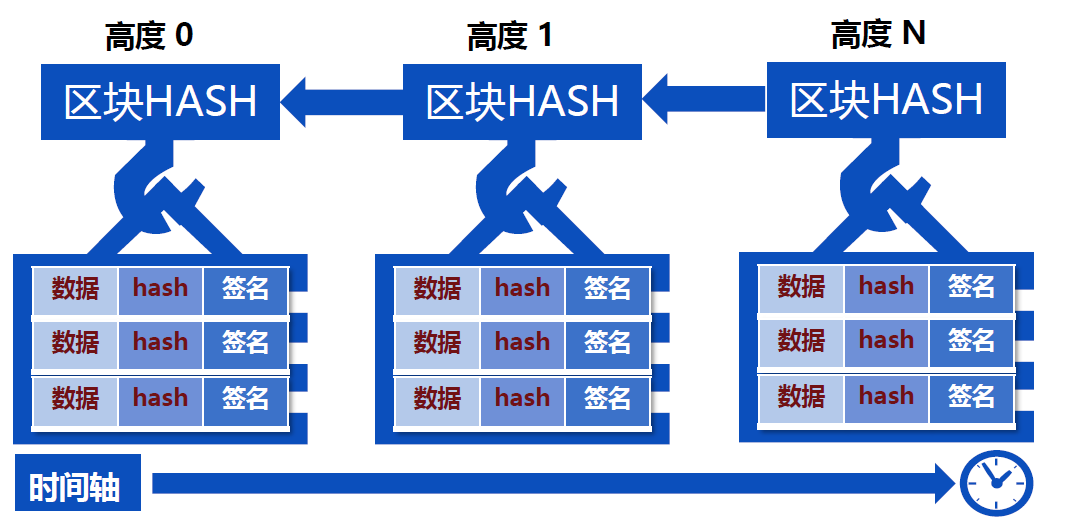
其次是信数据。区块链的数据结构是区块和链。新区块将自己的区块高度、交易列表，和上一个区块的hash，共同再生成一个hash做为新区块的标识，如此循环，形成了一个环环相扣的数据链。这个链条里的任何一个字节甚至一个bit被修改，都会因为hash算法的特性被校验发现。同时，区块数据被广播给全网所有参与者，参与者越多，规模效应越强。少数人即使强行修改、删剪自己的区块数据，也很容易被其他人校验出异常并不予采纳，只有多数人认可的数据得以留存和流传。也就是说，数据是大家人盯人的形态盯着的，且存在多份副本，一旦落地，只要链还在，数据就可以永远留存。基于容易验证的链式数据结构、群体冗余保存、共同鉴证，区块链数据是“难以篡改”的，所有人拿到的数据也都是一致的，信息公开透明，公共知识得以彰显和固化。

再者是信博弈论。区块链中最玄妙的部分是共识算法。共识算法的定义是在一个群体中，用一种机制协调大家共同或轮流记账，得出无争议的、唯一性的结果，且保证这个机制可以持续下去。

然后是信智能合约。一个智能合约是一套以数字形式定义的承诺，包括合约参与方可以在上面执行这些承诺的协议。简单地说，可以理解为纸质合约的电子版，用代码实现，无差别地运行在区块链网络的每一个节点上，在共识的作用下执行既定的合约规则。只不过，代码总是会有bug的，但是作为合约是公开的，意味着代码需要接受公众的审查，和严格的测试。所以这里的信任是辨证的，有条件的。

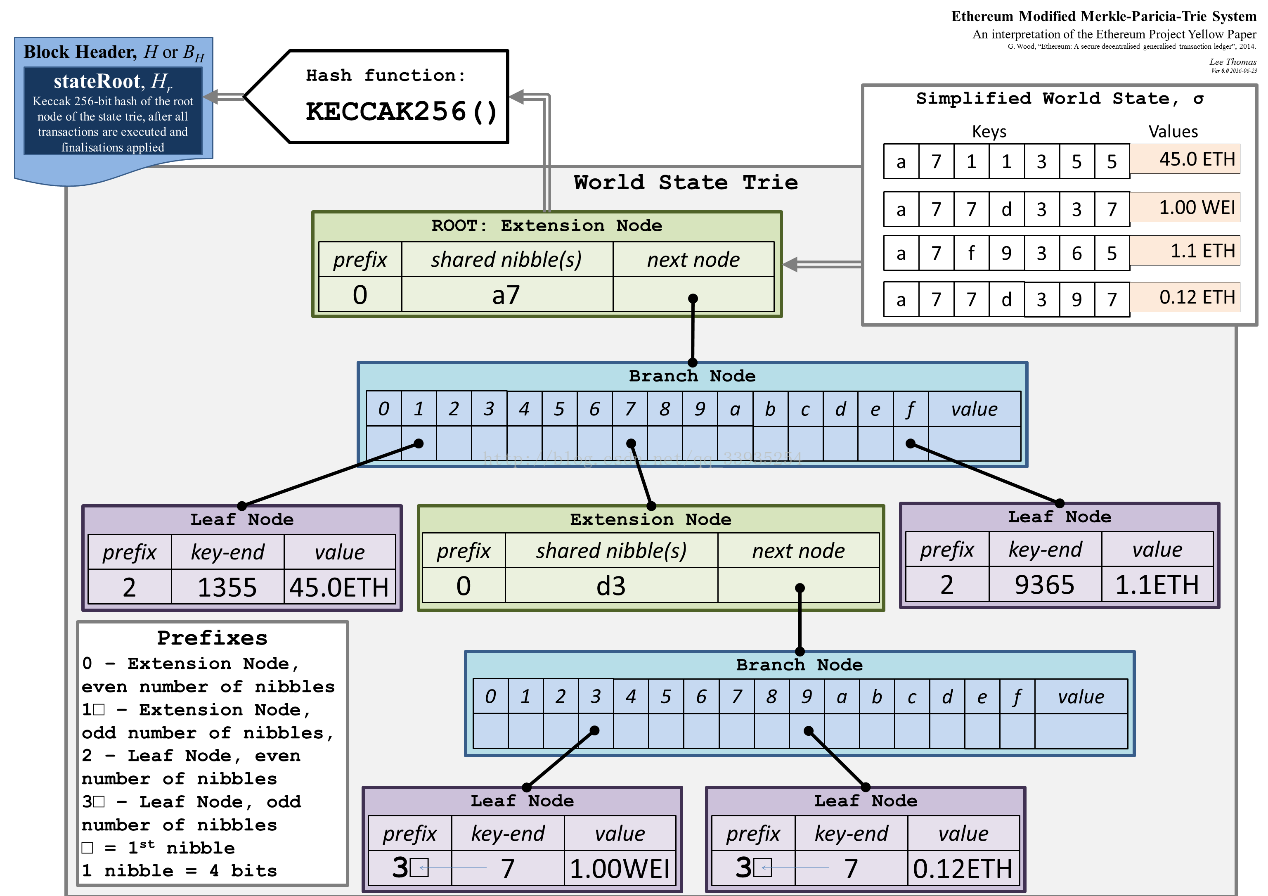
1. **链式存储和MPT存储**

区块链中每个区块包含一段时间内产生的交易数据，把相关的数据汇总计算摘要，进行汇总的完整性正确性证明。每个区块计算摘要时，把前一个区块的摘要作为一个数据计算在内，构成了数据链。最新区块包含了所有数据链的完整性证明，整个链条上的任何数据改动都会破坏数据链的相关性。



MPT的全称是Merkle Patricia Tree，是Merkle Tree + Patricia Tree。MPT树中一个重要的概念是十六进制前缀编码，用来对key进行编码。因为字母表是16进制的，所以每个节点可能有16个孩子。因为有两种[key,value]节点(叶节点和扩展节点)，引进一种特殊的终止符标识来标识key所对应的是值是真实的值，还是其他节点的hash。如果终止符标记被打开，那么key对应的是叶节点，对应的值是真实的value。如果终止符标记被关闭，那么值就是用于在数据块中查询对应的节点的hash。无论key奇数长度还是偶数长度，HP都可以对其进行编码。最后我们注意到一个单独的hex字符或者4bit二进制数字，即一个nibble。

HP编码很简单。一个nibble被加到key前，对终止符的状态和奇偶性进行编码。最低位表示奇偶性，第二低位编码终止符状态。如果key是偶数长度，那么加上另外一个nibble，值为0来保持整体的偶特性。

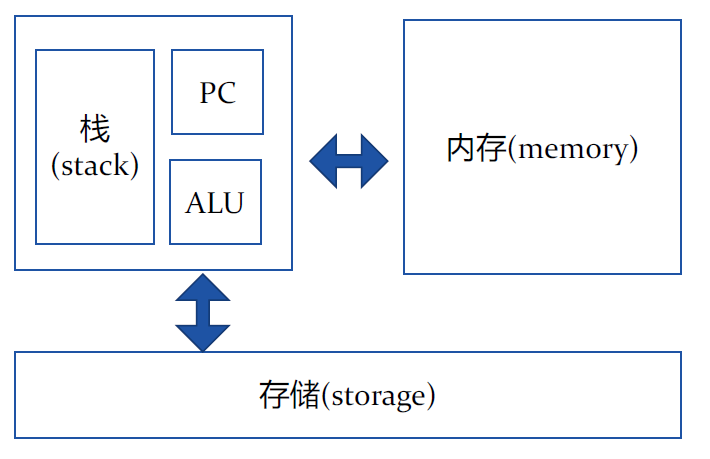


1. **Gas在智能合约中的作用**

Gas就是让智能合约在区块链中可以执行下去所需的花费。交易是按照既定的合约一步步执行，每执行一个命令就有一定数额的Gas被消耗。Gas的存在一方面是作为区块链中各个节点的回报，以此鼓励节点积极去打包交易。另一方面Gas的存在也可以限制恶意的或者不正确的智能合约的运行，其保证了区块链网络的稳定。假如合约中存在死循环的代码，当Gas的消耗达到上限时，事务将被回滚，这样避免区块链网络所有节点的瘫痪。

1. **EVM中的数据存储结构**

EVM的核心是大小为256bit的寄存器，称为PC，PC总是指向某条EVM指令，从EVM启动开始直到EVM停机，ALU一直不停地执行PC指向的指令，再更新PC指向下一条指令。



EVM中的栈的每个元素大小为256bit，栈的最大深度为1024。内存(memory)是临时存储设备，在ALU执行时，用来存放智能合约和智能合约处理的数据。内存是一个线性的整数数组，每个整数有唯一的地址，大小为256bit。存储(storage)是账户拥有持久的存储空间，它是key-value结构，key和value均为256bit的二进制串，在不借助外力的情况下，存储是不可遍历的，智能合约只能读写本账户的存储，无法读写其他账户的存储。

1. **群组架构的好处**

采用群组架构的网络中，根据业务场景的不同，可存在多个不同的账本，区块链节点可以根据业务关系选择群组加入，参与到对应账本的数据共享和共识过程中。

各群组独立执行共识流程，由群组内参与者决定如何进行共识，一个群组内的共识不受其他群组影响，各群组拥有独立的账本，维护自己的交易事务和数据，使得各群组之间解除耦合独立运作，可以达成更好的隐私隔离；

机构的节点只需部署一次，通过群组设置即可参与到不同的多方协作业务中，或将一个业务按用户、时间等维度分到各群组，群组架构可快速地平行扩展，在扩大了业务规模同时，极大简化了运维复杂度，降低管理成本。

1. **分布式存储的优势**

FISCO-BCOS对分布式数据存储的支持，节点可将数据存储在远端分布式系统中，克服了本地化数据存储的诸多限制。分布式存储可以将计算和数据隔离，节点故障不会导致数据异常；数据在远端存储，数据可以在更安全的隔离区存储。分布式存储不仅支持Key-Value形式，还支持SQL方式，使得业务开发更为简便；世界状态的存储从原来的MPT存储结构转为分布式存储，避免了世界状态急剧膨胀导致性能下降的问题。

1. **当前区块链实施的难度**

区块链的去中心特性，增加了监管的难度。而区块链最初是从比特币中脱离出来的，之后的绝大多数应用也与数字货币有关，由于人们对这个新兴技术认知不足，盲目跟随，导致财产损失严重，负面新闻层出，区块链也因此给部分人留下了不好的印象。

在某些业务场景下，比如企业自营的业务，中心化的系统比去中心化的系统更有优势。一方面，中心化的系统能应付高并发、高效率、高吞吐的场景，而这些是区块链无法媲美的。另一方面，在这些场景下，尽管企业拥有极高的控制权，但企业已经建立了能让消费者完全信任的征信机制。

而对于多方合作的业务而言，现有的刚需必然已经存在有效的且已经成熟的解决方案。虽然区块链技术可以十分完美的解决这类业务问题，但是项目的落地有可能会损害到现存体制的核心利益，所以是不会一帆风顺的。

区块链技术仍然需要时间去不断发展革新，无论是在人们的认知上，还是技术的发展，还是相关的监管政策的出台，都需要各方的努力，这些难题不解决，区块链技术的发展也必将受到限制，也就不可能真正为我们所用。

1. **音链项目的开发**
2. **需求概述**

随着区块链技术应用的推广，该技术在越来越多的领域得到了应用，特别是在企业服务方向的应用，给越来越多的企业和政府部门带来了一定程度效率上的提高，数据不可篡改和去中心化的特性帮助企业之间建立更加完备的信任机制，减少了在谈判和不信任的情况下的花销。

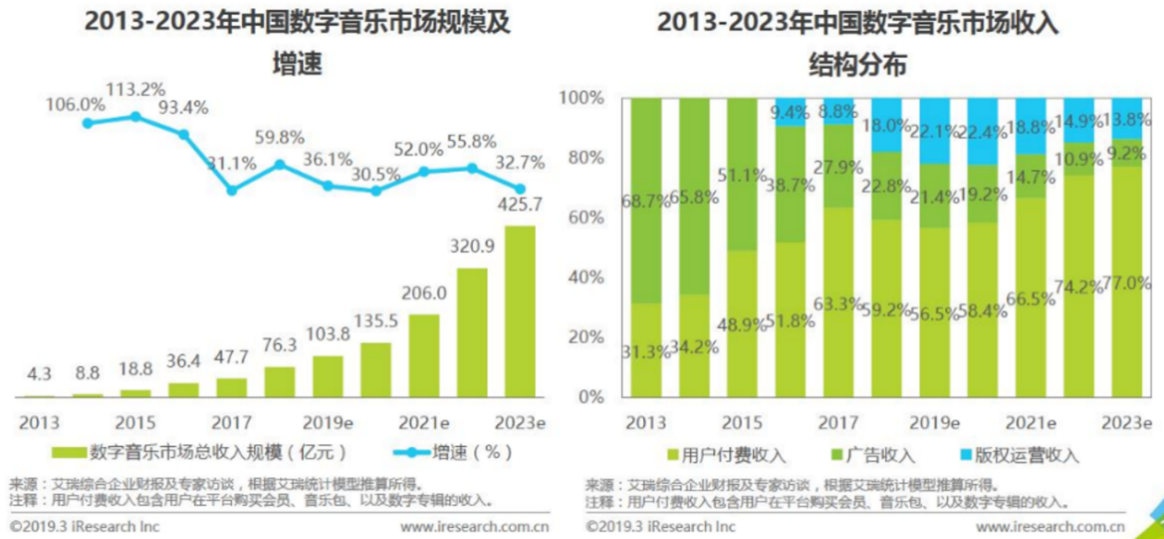
在调研中，我们发现中国的音乐市场正在快速发展，高速增长的音乐市场规模和巨大的音乐市场体量吸引了越来越多的人和企业加入到这个战场中。自2015年中国音乐版权的觉醒以来，涌现了众多数字音乐的授权平台，如100Audio、V.Fine Music等，他们一方面保护着音乐人的音乐版权不受侵犯，另一方面为众多的音乐人提供音乐宣传推广的平台，提高音乐人的收入。

尽管如此，中国的音乐市场相较欧美国家仍然有较大的差距。我国严厉打击网络侵权以来，据不完全统计，仍然有6900多起关于音乐版权纠纷的案件发生。在音乐授权的平台上，音乐人将音乐上传到平台后，音乐的宣传力度是有限的，同时，他们所获得的收益仅为40%至70%不等。音乐人在中国音乐著作权协会中登记音乐版权后，他们的收益到账速度较慢，需要等待较长的周期才能完成收益的分配。当用户希望获得在中国音乐著作权协会中登记了版权的音乐的使用权时，需要通过协会的申请，其中需要较多次经过协会才能完成的手续。

由此，针对音乐版权纠纷问题、音乐人权益保障、音乐授权使用效率的提高等问题，我们提出了基于区块链的解决方案。为音乐人、版权企业、使用音乐的普通用户、使用音乐的企业和仲裁机构提供统一化的平台。此外，企业用户可参与到区块链系统中，并基于该系统打造自己的平台。

1. **需求分析**
2. **市场分析**

从艾瑞咨询的音乐行业市场报告来看，中国的音乐市场在未来五年间，仍然有庞大的增长空间，而当前音乐市场的总收入达到了近百亿元的规模。



1. **国内音乐授权平台分析**

AGM平台：数字音乐素材授权、音乐定制、授权查询、音乐人合作。

100Audio：数字音乐素材授权、音乐人合作。

V.Fine Music 平台：数字音乐素材授权、音乐人合作、音乐定制、授权查询。

特点：形式均为音乐商城、音乐库规模大、模式成熟。

商业模式：专注于版权音乐授权的电商平台。

主要商品：数字音乐素材。

特点：授权快捷；使用范围、时间、价格、数量统一，减少双方谈判沟通的成本，避免劣币驱逐良币；提供快速寻找合适的音乐素材，提供定制功能。

1. **中国音乐著作权协会分析**

中国音乐著作权协会是国内唯一的音乐著作权集体管理组织，针对海量音乐使用者出现的侵权情况，登记并管理音乐作品及其版权资料，提供著作许可服务并收取费用，提供法律支持。

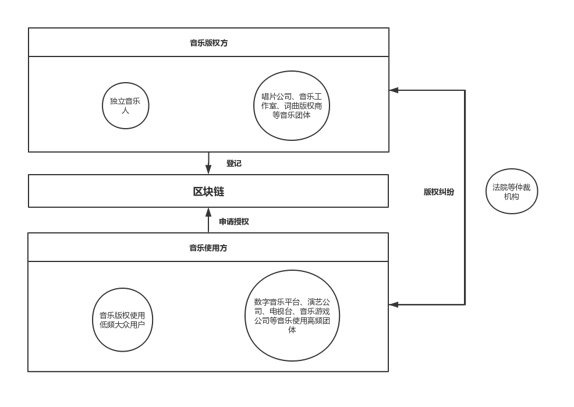
协会中,拥有版权的人可以进行：版权注册、版权交易、作品修改、作品注销等功能。但是从版权注册的流程来看，申请版权注册需要45+30+邮费接近100元的费用，版权人的作品分配授权费用时需要至少两个月的时间才能够到账。

对于音乐版权的使用者来说，协会为这类用户提供音乐的复制权、表演权、广播权、信息网络传播权的申请使用。

但是其中的问题有：需提前明确使用的歌曲及其信息；通过协会报备、审核；办理许可、出具发票；收费标准由协会统一商定。这整个过程都会需要通过音乐协会才能进行，且整个流程的周期较长。

1. **用户角色分析**

用户角色从整体来看，分为3种类型，音乐产生者，音乐使用者，纠纷仲裁机构。



1. 独立音乐人

如，网易云音乐中的独立音乐人。从调研来看，音乐人的主要需求在于音乐版权的管理，以及音乐的授权问题。音乐人能够在我们的平台上登记他们的音乐版权，能够对版权进行转让、授权，同时在必要时能够对音乐版权的注销。他们通过平台直接与版权使用方进行版权的交易，从而提高他们的收入，加快收入的到账速度。

2. 版权所有企业

如，环球音乐、索尼音乐、华纳音乐等这样大型的音乐公司。对于版权所有企业来说，最重要的目标是通过版权的售卖和授权使用能够以最大效率来获得收益。因此，一方面需要为他们提供一个能够快速授权的平台，另一方面，能够为他们授予一个保护他们版权的平台。

3. 版权使用企业

如，网易云音乐、QQ音乐等大型数字音乐平台以及网易游戏这样需要常常使用音乐素材制作游戏的公司。版权使用企业可能存在大量地使用音乐的情况，也可能存在各个小部门会需要使用少量音乐的情况。大量使用音乐时，企业之间可以通过区块链进行合作来共享音乐；当企业只需要使用少量音乐时，可以直接进行单曲授权，从而快速获得版权的授予。

4. 版权使用普通用户

如，抖音平台、直播平台中的网红、中小型的游戏制作方、音乐改造爱好者等普通用户。对于普通用户来说，他们使用音乐往往在数量较少，能够快速获得音乐的授权对这类用户群体来说显得更加重要。

5. 仲裁机构

仲裁机构是在处理版权纠纷时，能够快速地查询并对版权的信息和历史记录进行取证，快速判案，通过区块链中的数据提高解决案件的效率。

1. **系统用户划分**

在系统中，用户设计为四种类型：普通用户、音乐人、企业、仲裁机构。以下为各种类型的用户对应的功能，其中，普通用户、企业、音乐人考虑到了他们会申请其他音乐使用权的授权，因此都提供了申请授权功能。而音乐人、企业、仲裁机构三种类型的用户需要进行认证才可确定对应的身份类型。此外，考虑到企业会购买版权、雇佣音乐人来制作音乐等问题，直接让企业版权的拥有方和使用方具有相同的功能，在使用时，他们根据自己的身份和需要进行使用即可。

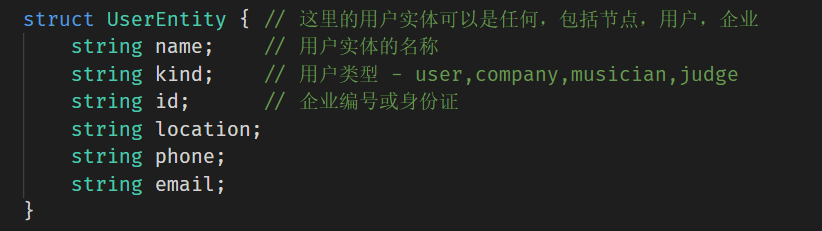
1. **功能性需求**

|  |  |  |  |
| --- | --- | --- | --- |
| **功能名称** | **用户角色** | **描述** | **备注** |
| 账号注册 | 普通用户 | 用户输入姓名与手机号码注册，获取系统返回的公钥和私钥。 |  |
| 登录 | 普通用户、音乐人、企业用户、仲裁机构 | 各类用户使用自己的私钥进行登录。 |  |
| 音乐人认证申请 | 音乐人 | 音乐人填写姓名、身份证号码、所在地、联系电话、电子邮箱等信息，申请认证音乐人。 |  |
| 企业认证申请 | 企业用户 | 企业用户填写姓名、企业编码、所在地、联系电话、电子邮箱等信息，申请认证企业用户。 |  |
| 仲裁机构认证申请 | 仲裁机构 | 仲裁机构填写机构名称、所在地、联系电话、电子邮箱等信息，申请认证仲裁机构。 |  |
| 版权信息查询 | 任何用户 | 任何用户都可在音乐区输入音乐作品的名称、作者名字即可进行查询音乐作品的版权人信息。 |  |
| 版权授权申请 | 普通用户、音乐人、企业用户 | 用户在音乐区查询音乐后，点击申请授权，填写授权需要的购买用户、授权地域、授权期限、其他说明、授权人姓名、联系电话等信息后付款即将授权申请单提交给版权人。 |  |
| 版权详细信息查询 | 仲裁机构 | 仲裁机构登录后，输入作品名称和作者名字即可查询作品的详细信息。 |  |
| 版权历史记录查询 | 仲裁机构 | 仲裁机构登录并输入作品名称和作者名字后可查看作品交易的详细记录。 |  |
| 仲裁机构信息查看 | 仲裁机构 | 仲裁机构登录后，可查看仲裁机构的名称、所在地、联系电话等信息 |  |
| 用户信息查看 | 普通用户、音乐人、企业用户 | 用户登录后，可查看用户名称、编码/身份证号码、所在地、联系电话、电子邮箱等信息 | 普通用户仅有名字和联系电话信息 |
| 授权订单查看 | 普通用户、音乐人、企业用户 | 用户登录后，可进入授权订单中，查看用户申请的授权，当授权被同意后，电子授权证明可使用。 |  |
| 电子授权证明查看 | 普通用户、音乐人、企业用户 | 用户登录后，可进入授权订单中，查看已被授权的作品的电子授权证明。 |  |
| 版权转让 | 音乐人、企业用户 | 用户登陆后，可在音乐版权列表中，将对应的音乐进行转让给其他用户，填写其他用户的公钥后，即可将版权所属权转让给公钥对应的用户。 |  |
| 版权注销 | 音乐人、企业用户 | 用户登录后，可在音乐版权列表中，将对应的某首作品进行注销。 |  |
| 版权登记 | 音乐人、企业用户 | 用户登录后，在个人中心中，使用版权登记功能，填写作品的信息后即可申请作品的版权登记。 |  |
| 授权申请判定 | 音乐人、企业用户 | 用户登录后，用户在授权申请列表中，对应申请授权的授权单进行确定同意与否。 |  |
| 版权共享 | 企业用户 | 企业用户登录后，针对已经确定需要进行音乐版权共享的情况，将音乐版权与其他企业或用户进行共享，给予他们相应的使用权力。 |  |

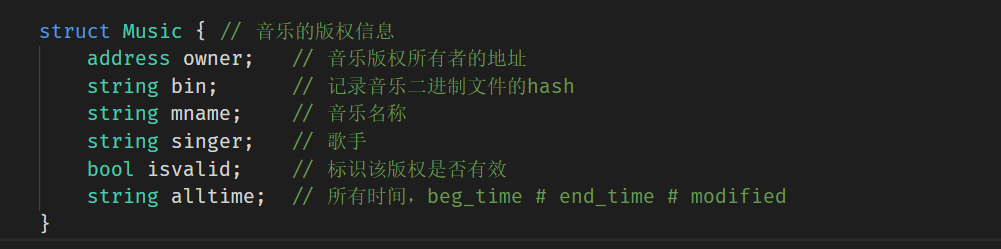
1. **智能合约设计**
2. **数据结构**

在合约层面，不论是平台还是大众用户，均被视作用户实体(UserEntity)，且拥有一个地址(账户)；在链上会存储每个地址(账户)的信息，通过 mapping 实现；用户可以更新自己的信息；用Music[]存储所有音乐，而不是mapping(address=>Music[])，这里可能会有性能上的问题，但是实现简单；授权记录的实现方式描述和上述一样；信息查询方式，仅支持多个关键词全字匹配，这是基于需求的考虑。

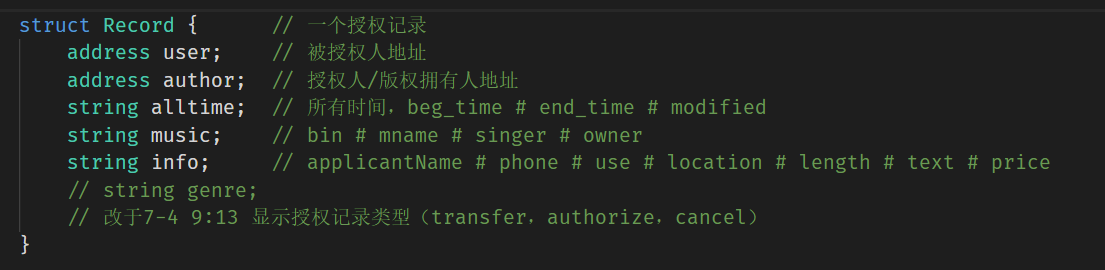
参考用户角色分析，用户可以分为普通用户、音乐人、企业和仲裁机构，前者通过申请认证功能申请成为不同的用户类型。普通用户通过姓名和手机号码进行注册，生成UserEntity实例，其中的id、地址、邮箱属性在用户进行认证时接收用户对应的输入。



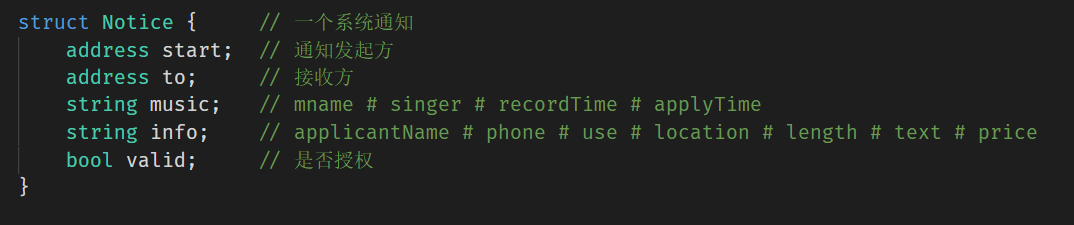
用户可以通过登记版权功能将自己的音乐上传到音链上，因此需要基本的Music架构记录用户所上传的音乐。Music通过唯一的二进制hash来对每一首不同的音乐进行标识，mname、alltime、singer等属性在用户调用register函数时输入。



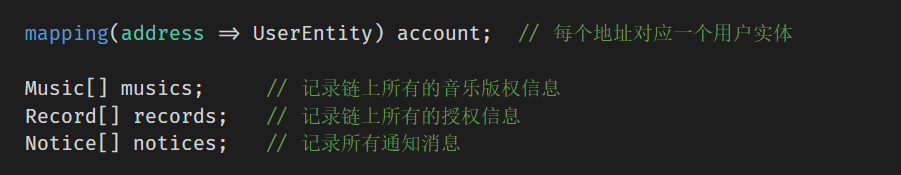
基本架构Record用以记录用户执行的授权操作，在用户执行同意授权、版权转让和注销音乐操作时会生成相应的Record记录用户的操作，方便用户进行查询，genre记录用户操作的类型，user和author分别记录该操作涉及的两个用户的地址，music、info、alltime记录与Record生成的音乐内容、Record内容和时间信息。



基本架构Notice在申请方向版权方提出授权申请时生成，在版权方登陆账号查看时提示版权方进行申请处理。start、to分别记录通知发起方和接收方的地址，music和info储存该通知涉及的音乐信息和申请人信息，最后用valid记录申请受理状况。

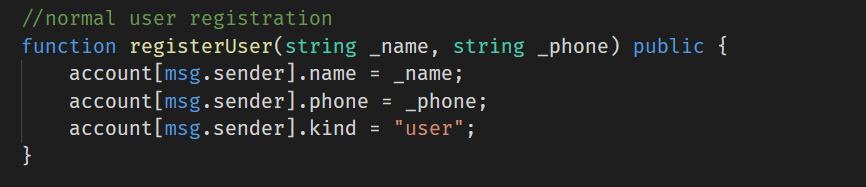


利用mapping将用户地址和UserEntity一一对应，另外用数组储存Music、Record和Notice记录。

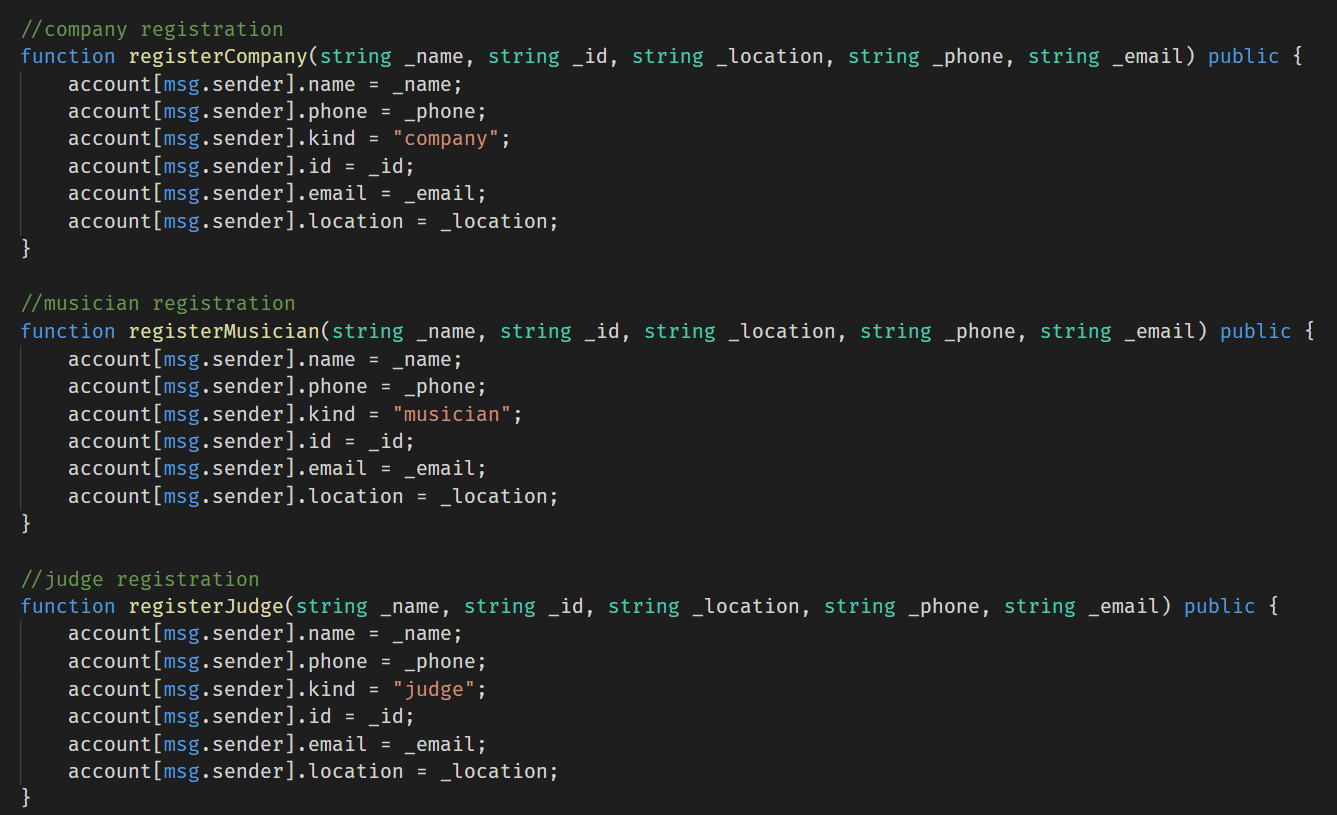


1. **功能实现**

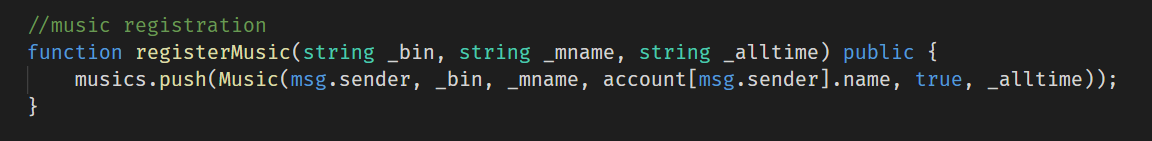
用户首次注册通过调用registerUser，输入姓名和电话，生成与其地址唯一对应的UserEntity。



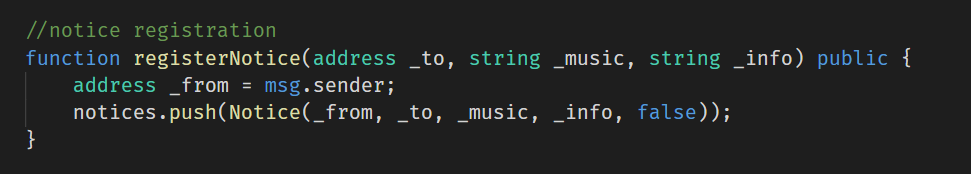
用户通过registerCompany、registerMusician、registerJudge函数分贝申请认证为企业、音乐人、仲裁机构，并在申请时完善id、地址、邮箱等信息。



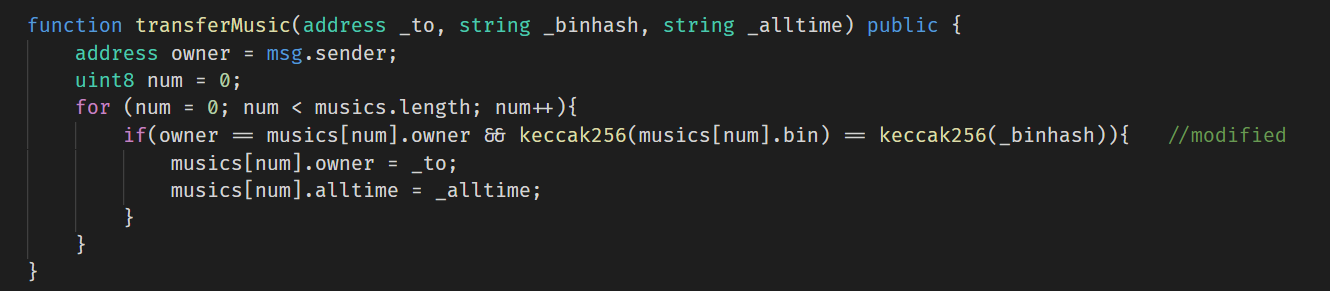
用户在上传音乐时需要调用registerMusic函数，函数从后台接收binhash、时间信息alltime以及用户输入的音乐名称，将生成的音乐存储添加进musics数组中。



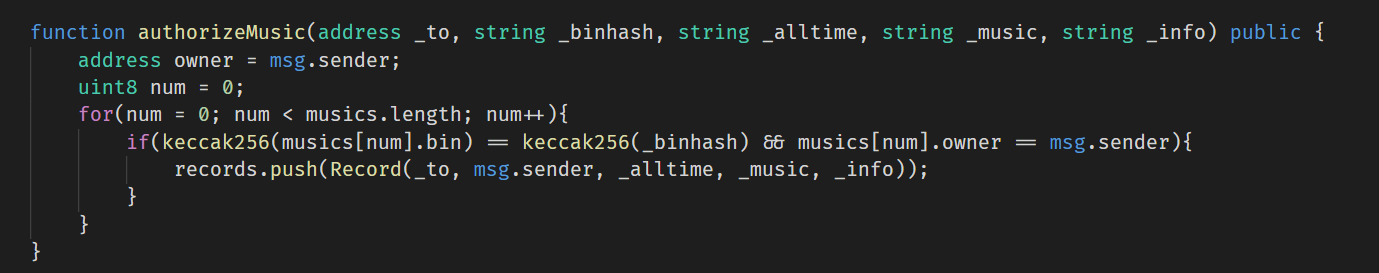
申请方在填入数据并点击申请时需要调用registerNotice函数，该函数接收版权方地址、音乐信息以及申请信息作为输入，生成一条notice并加入到数组中，当版权方登陆时，在授权申请界面将会看到这条申请。



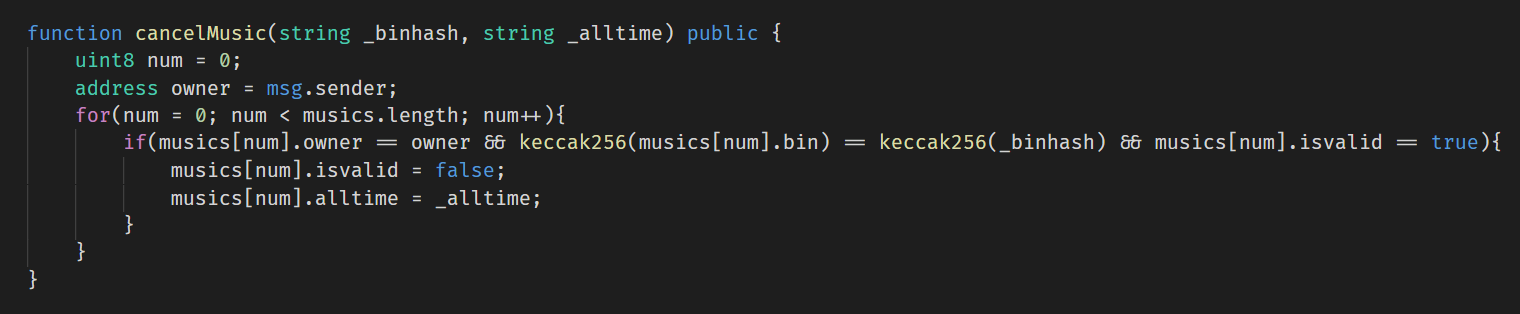
transferMusic实现了版权转让的功能，函数接收目标用户地址、音乐hash码和时间信息为输入，通过修改该music对象的owner地址和时间信息，记录下新的版权拥有者和版权起始、结束时间等信息，最后通过生成一条record记录下这首音乐的版权转让操作，方便用户的后续查询。



在版权方接收到申请方的notice后，他可以查看其中的申请内容，如果版权方同意将版权授予申请方，那么他在点击同意时将会调用authorizeMusic函数，该函数从后台接收申请方地址、音乐hash码、时间信息、音乐信息和申请方信息，并生成一条record记录这首音乐的授权操作。



cancelMusic对应版权撤销功能，当这首音乐的版权拥有者点击确定注销以后，cancelMusic将会更改这首音乐的valid标识符，同时生成一条record记录下版权拥有者对这首音乐的版权撤销操作。



1. **后端设计**
2. **架构设计**



后端的整体架构大约分为3层：数据接口、服务层和Controller层，结构如上图所示。

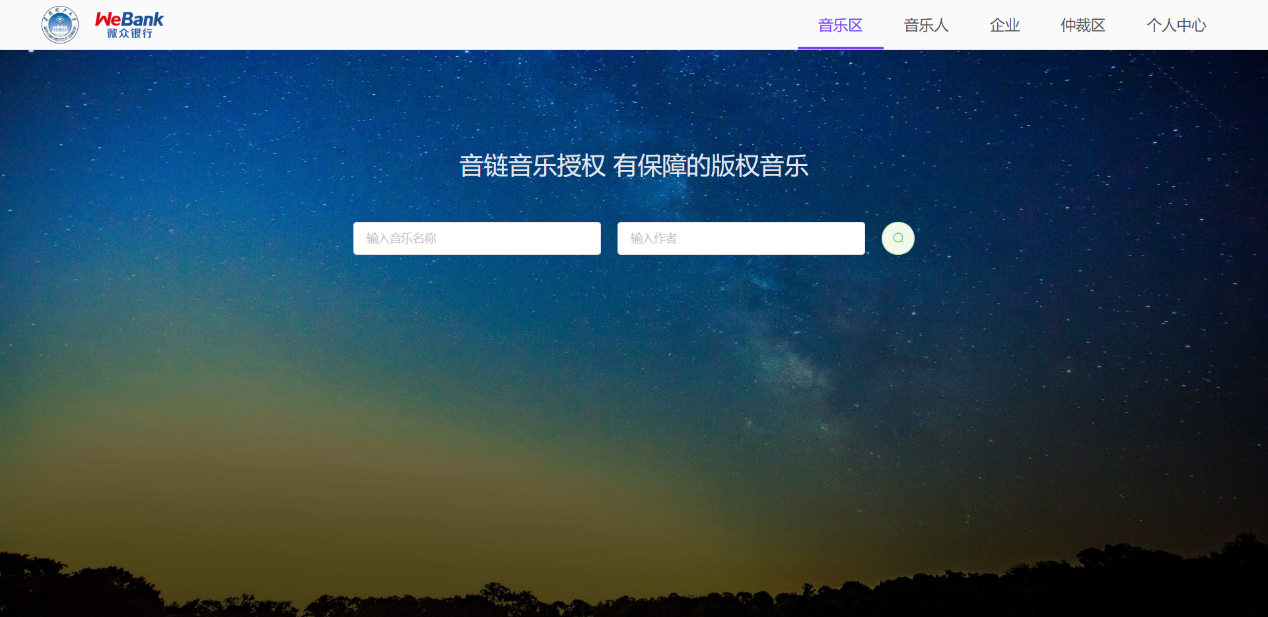
（1）数据接口：在智能合约转化成java文件以后，对其暴露的函数接口进行封装，使接口更加美观，方便服务层进行调用。

（2）服务层：利用数据接口实现需求，使其可以被Controller层所调用。

（3）Controller：响应前端的请求，并返回所需的数据。

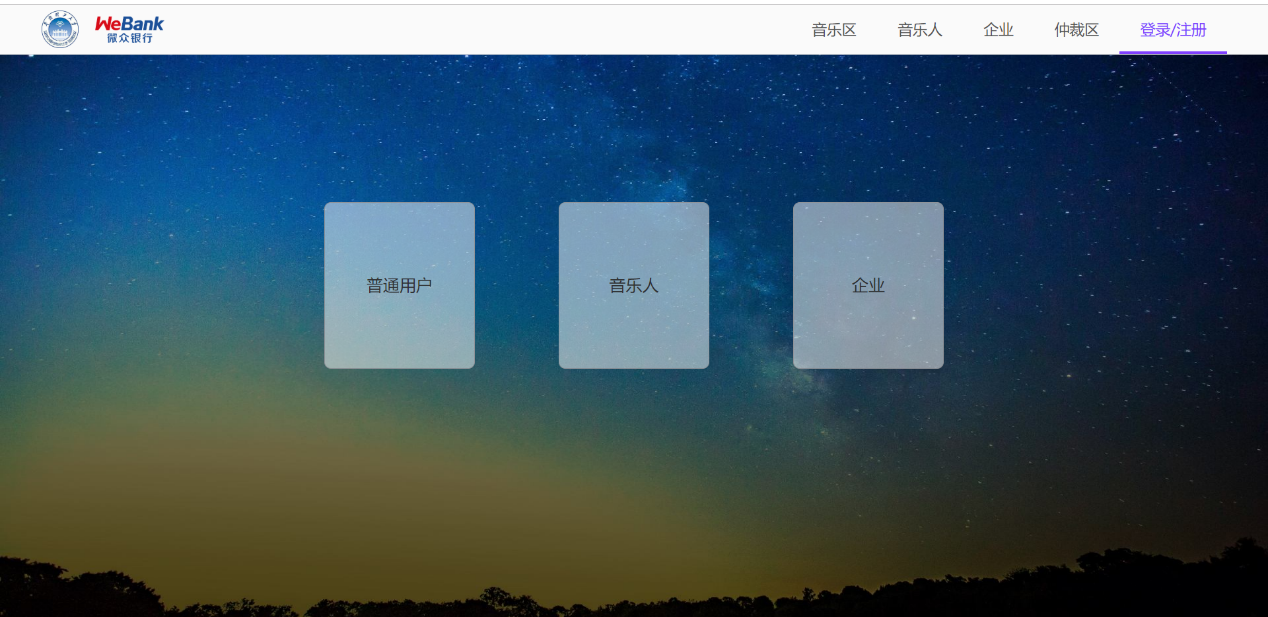
1. **前端设计**

前端项目使用了Vue框架，搭配Element UI实现。总共包含10个页面，以及数个操作流。以下展示部分图：



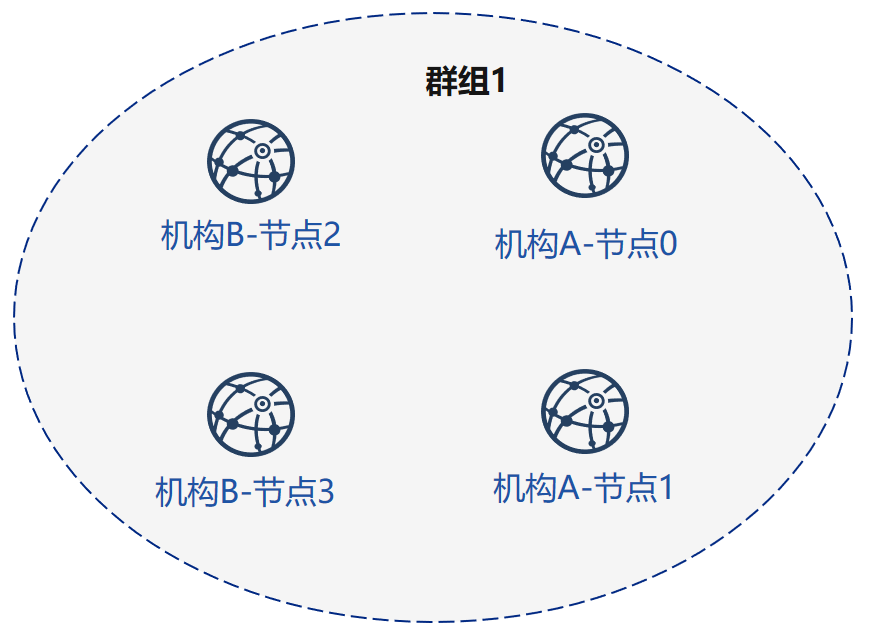




****

Vue是一个组件式开发的框架，因此，我按照分组件的思路将前端实现切分为三个模块，分别是顶栏、各个展示页以及个人中心功能页。个中的展示页，包含了数个对话框模块，用于与用户交互且具有用户友好性，此类功能包括申请授权、登录、注册等。

1. **区块链部署**
2. **组网模式**



对于需求的再考虑，我们觉得搭建多群组架构暂无必要，因此，我们采用多机构单群组的组网模式。

1. **机构、节点、群组、地址信息配置**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 机构 | 节点 | 所属群组 | P2P地址 | RPC/channel监听地址 |
| 机构A | 节点0 | 群组1、2 | 127.0.0.1:30300 | 0.0.0.0:8545/:20200 |
| 机构A | 节点1 | 群组1、2 | 127.0.0.1:30301 | 0.0.0.0:8546/:20201 |
| 机构B | 节点2 | 群组1 | 127.0.0.1:30302 | 0.0.0.0:8547/:20202 |
| 机构B | 节点3 | 群组1 | 127.0.0.1:30303 | 0.0.0.0:8548/:20203 |

为了便于协作开发后端，我们决定事先将区块链部署上云，并且向外网开放channel端口，以此省去开发后端时对区块链的搭建过程，也保证了在多人协作开发时的区块链配置一致。除此之外，也能让智能合约的开发和调试能同步进行且互不干扰，极大减少开发时对环境的依赖。

1. **课程回顾**

在上这门实训之前，我对区块链的认识仅停留在百度百科上的浅显的描述而已。第一节课开始，许老师便开始给我们讲区块链的基础知识，从区块链的前世今生，到密码学基础。刚开始，对于非对称加密，数字签名，数字证书等知识的理解还十分懵懂，特别是对应的例子，听起来也特别的绕。后来反复的看资料上的例子，才将其梳理清楚。在非对称加密算法中，一组密钥对对于公私钥的区分应该是看场景的。

接下来便是区块链的实操，从搭链开始，到通过控制台去部署合约，查看区块等。至此，我对区块链有了进一步的认识，心里会想原来区块链是这样一个“实物”，夹杂着一些潜意识的认知，我轻易的将其理解为一种数据库，一种分布式的可靠的数据保密的数据库。后来，听完企业导师的课之后，从build\_chain脚本的解析，到自己去尝试通过RPC访问区块链，自己对区块链技术或者说是FISCO-BCOS这个平台，也有了一定的认识，最起码能把 FISCO-BCOS的逻辑架构图看明白。

有了这样的知识基础，在接下来的项目选题和实现上，我们能最大程度的贴合区块链。于是乎，诞生了音链这样一个项目。在两周不到的时间内，我们基本完成了整个项目。尽管最后，我们项目中还有一些需求没有完成且部分bug没有解决。这个课程让我收获颇丰，这次项目经历也令我印象深刻。

**参考文献**

[1]看懂区块链的技术原理 <http://www.360doc.com/content/18/0221/20/16022863_731276817.shtml>

[2]说信任区块链时究竟在信任什么？<https://blog.csdn.net/fiscobcos/article/details/88877530>

[3]以太坊MPT原理，你最值得看的一篇<https://blog.csdn.net/itleaks/article/details/79992072>

[4]FISCO-BCOS官方文档

<https://fisco-bcos-documentation.readthedocs.io/zh_CN/latest/docs/introduction.html>

[5]从音乐作品大数据探著作权侵权与保护<http://blog.sina.com.cn/s/blog_7813a0670102xc7u.html>

[6]Element UI官方文档 <https://element.eleme.cn/#/zh-CN>

[7]Vue.js官方文档 <https://cn.vuejs.org/>