Bachelorarbeit

Blockpass — Detailed Academic Records on an Ethereum Blockchain

Verfasser

Barakura Thierry Tuyishime

angestrebter akademischer Grad

Bachelor of Science (BSc)

Wien, 2018

Studienkennzahl lt. Studienblatt:	A 033 521
Fachrichtung:	Informatik - Medieninformatik
Betreuer:	Univ.-Prof. Dipl.-Ing. Dr. Wolfgang Klas

Acknowledgements

I would like to thank Alexander Garber for providing me technical support and tips on how to go about it, during the development of this project.

Contents

Abstract

This thesis explores the viability of creating a blockchain based system, that can be used by universities to manage the complete life cycle of education records. With the goal being that students should be able to get a record detailing every task's contribution to their academic studies. Research related to this topic as well as a detailed look at how this project tries to tackle the problem is presented. Future work and a conclusion summarising all findings is given at the end.

1 Introduction

With Bitcoin becoming as widely known as it is today, many industries have now caught on to some of its hype. As the financial market has started to embrace or at least acknowledge it[12], building on the basic technologies it has popularized. One of them is called the blockchain, which simplified to its core is a database that is virtually tamper-proof. Combined with a network where everyone shares the same database and a consensus algorithm, which ensures everyone agrees on the database's state, this becomes a powerful tool. Bitcoin adds a scarce resource, which is used to represent value to effectively become a decentralised money exchange network

Some of the people trying to do even more, are the ones behind the Ethereum Foundation[2]. One of its founders, Vitalik Buterin, is the creator of Ethereum, which is another cryptocurrency that builds on blockchain technology. In contrast to Bitcoin however, Buterin made the blockchain Turing complete.[13] This allows others to develop programs for the blockchain — so called distributed applications, abbreviated as Apps. They run on the Ethereum Virtual Machine, which essentially represents a globally shared distributed computer and leverage smart contracts that can also save state.

Naturally the possibilities of running programs on a globally available machine have led to much speculation about what smart contracts are capable of. Not only can you use that technology to create your own cryptocurrency, you can create almost anything you want. A prediction market like Gnosis[5], a digital advertising platform using a token that is exchanged on the ethereum network[3], a voting system[6] or many others.

Some believe smart contracts could change the internet itself, to make it revolve around values instead of information. Since they can be used to create autonomous contracts, envisioning them to be used in place of law or other regulations does not seem so far fetched either. Whether code can indeed become law like Lessig might have envisioned for the internet, however, is something that still needs to be explored.[23]

Today different entities are trying to find out how smart contracts can be used for their own purposes, since a secure shared database is a desirable technology to have. There are however a few considerations that need to be made in order to decide whether a blockchain is something that is suitable for a specific application or not. Therefore we are going to explore the usefulness and feasibility of applying smart contracts to a particular field – the educational system.

More specifically using the blockchain to record a student's complete history of study activities and achievements. The goal being to have a complete picture of any kind of study related data at arbitrary granularity.

To that end we will firstly look into different applications of blockchain technology in the educational sector as well as other unique uses of smart contracts. We will delve into common problems and go over how some of those applications are implemented. Then a detailed look at our solutions to these problems will be provided, as well as an overview of our application specific design and realisation. Following that is a discussion of achievements, lessons learned and a look at possible future work. A setup guide, which can be used for recreating the work presented in this thesis, will be included in the appendix section.

2 Related Work

There have only been a few applications of blockchain technology in the educational sector. Some approaches that have been made, relate to the issuance and verification of certificates. At the Massachusetts Institute of Technology, a platform for digital certificates geared towards universities was developed as part of a project by the Media Lab Learning Initiative.[4] The system uses the Bitcoin blockchain to store the hash of a "well-structured digital certificate"[25] in a transaction, whose output is assigned to the recipient. This makes it possible to verify the issuer, recipient and the validity of a particular certificate. It also allows easy transfer of learning experiences and their corresponding certificates to other systems used for university admissions or even future employers.[31] Their work has been made publicly available for collaboration under the name Blockcerts.[1, 32]

A literature review by Ølnes[26] on Bitcoin infrastructure for e-government use cases, has shown that there has not been sufficient research in this field yet. They have found that Bitcoin technology or more generally blockchain technology research, has mostly been done in technology, economy and regulation fields. In their opinion, the public sector is one of those that does have great potential for blockchain technology because of the possibility to extend the storage of certificates to other types of documents. With respect to the general lack of research on blockchain technologies in the public sector, they argue that academic research should try to find arguments as to why there has not been much development.

An article by M. E. Peck tries to answer a question directly related to that, namely whether a blockchain is really needed for a specific use case.[28] It is argued that if a traditional database can be used to meet the needs, then it is unwise to think about a blockchain based application. The article goes into detail about how specific use cases are not fit for the blockchain despite constant efforts of well-intentioned people. Among other aspects that have to be looked at, throughput, privacy and governance are some of the more important ones when considering public blockchains. Although there are alternatives in permissioned ledgers, where trust and throughput are not that much of a problem

for example, they too do not represent a solution for all problems. Despite this, there are genuine use cases where sacrificing some aspects like speed and cost to gain the benefits of a blockchain is favourable. Some of those being applications in environments where censorship and universal access might be a problem.

Additional analysis of blockchain technology and its uses in governmental processes provides a critical view on benefits of blockchains and its implications. It highlights the incompatibility of governance requirements and blockchain technology as well as its immaturity and the need to shift away from a technology-driven approach. The administrative processes need to benefit from the technology in the same way that blockchain applications need to fit their requirements.[27]

There has also been research on blockchain technology and education at a lower level. The educational certificate blockchain by Xu et al.[38] aims to create a blockchain that is tailored to efficient querying of histories of certificates. Their consensus mechanism uses peers to create blocks in an attempt to reduce latency and increase throughput. For efficient querying they use a special tree structure called MPT-Chain, which also supports querying for historical transactions of one specific account.

Research that does not occupy itself solely with certificates, has been done by Turkanovič et al.[35] They have developed an education credit platform named EduCTX, which builds directly on the concept of the European Credit Transfer and Accumulation System — ECTS. It uses the Ark Blockchain Platform to create a unified global credit and grading system. Higher education institutions act as peers in this network managing ECTX, which are tokens handed out to students for course completions, similar to ECTS points.

We can see that although research has been made regarding blockchain technologies in the educational system, there still seems to be uncertainty on its best use. This means that "real world" applications are only few and far between. At ANN Polytechnic there is currently a system in use that utilises the blockchain for verifying the authenticity of diploma certificates.[30] Sony Global Education has also announced a project for using the blockchain in education.[33, 20] Their system records educational achievements and records in a manner that makes it possible to reference data and transcripts from multiple education institutions. In addition to that it controls access to that information and can be used to reliably share information with authorised third parties. It was planned to be used in the 5th Global Math Challenge held in 2017. Details however, are not available, which is symptomatic for many current applications of blockchain technology in education.

If we look at other industries, there have been a few examples of commercial applications that do provide background information on motivation, considerations and platform design. Share&Charge is one of those examples.[21] It is a sharing platform for electric vehicle charging stations that implements smart contracts and a token that is backed by Euros. Users can share their charging poles to get rewarded in tokens, which can be used for other services on the platform or traded in for cash. Built on the Ethereum blockchain, it uses some design philosophies for smart contracts that have also been found to be useful in

this thesis. One of them being the use of a manager contract for discoverability of other contracts and updatability of the system.

Another project that was created at the Bosch IoT Lab by researchers from ETH Zrich and University of St. Gallen is an odometer fraud prevention system using the blockchain to record mileage and GPS data.[10] This way users own and control the data, while at the same time making it easily verifiable without a 3rd party. It uses a dongle to read out the odometer and record the position to then transfer the data to an application on a laptop that also resides in the car. After salting and hashing the dataset, its fingerprint is published on the Ethereum blockchain by signing the transaction locally before submitting it. The actual data is encrypted and then uploaded to a cloud database. To verify the data a supplementary app is used that requests raw data from the database and verifies the hashes with those on the blockchain. Although the blockchain offers many advantages for this kind of use case, the researchers worry about the blockchain future as well. They have also done research on sharing platforms similar to Share&Charge.[9]

Record keeping and identities on the blockchain, in the context of education and other fields, has also seen some exploration. Accenture and Microsoft have developed a prototype of a digital ID network using blockchain technology.[8] Estonia has collaborated with Bitnation to provide notarisation services on the blockchain as an addition to their e-residency program.[34] Others have looked into putting medical records in the patient's hand by using blockchain technology.[19] Mercy College researchers have explored the use of the blockchain for creating a system that provides proofs of different kinds of personal information.[11] Their system allows tagging information with proofs that are verified by trusted bodies. In the case of education, those proofs may be college degrees or other academic achievements. Similar explorations have also been made for a data bank, where information resides with individuals instead of governmental entities or bigger organisations, like it currently is. The data bank that is based on blockchain technology would also allow for continuous access to more specific information like outside curricular activities or community work participation instead of only basing assessments on snapshots like test results.[29]

Blockchains and their supposed fit for record keeping has also garnered a bit of skepticism. Lemieux's paper[22] about trusting records explores whether blockchain technology can serve as a solution to creating and storing digital records. In using current digital preservation standards as a basis, a blockchain application for a land registry system is developed. The paper concludes that although information integrity benefits from blockchain technology, reliability of information is not guaranteed. Even though there is a lot of optimism because of the potential for increased efficiency, reduced costs and transparency,[39] their legal enforceability and adaptability to the current legal framework of regular contracts is still not answered.[14] However researchers suggest that the question of "whether smart contracts are a smart idea, [...] will soon be tested in the courts."[14, p. 11]

Overall there have been quite a few applications of blockchain technology in

different industries. The research done in the educational sector particularly, has been focused mainly on certificates and their verifiability. This thesis tries to have a look at other possibilities with regard to blockchains. Since certificates have to be based on prior achievements, one can extend the use of blockchain technology to every activity in the educational lifecycle. Record every assignment submission, every test and all courses a student takes. Automate assigning ECTS points for course completion, based on parameters that have been defined by instructors beforehand. Whether this is too ambitious is something that has not been researched yet.

3 Blockchain-related Design Considerations

The Share&Charge platform as well as the odometer project, highlight some of the problems you have think about when developing for the blockchain. Since it is practically impossible to change information that is already on the chain, you cannot easily update your deployed contracts. Another consideration that needs to be made, is how you want to sign transactions and handle user data. Generally one should strive to not be in possession of any user credentials. This however makes it relatively problematic to have mobile users. Mobile in the sense of different and possibly shared devices. Since all transactions have to be signed by a user's key, they have to be stored somewhere. There are however a few other design decisions that need to be made for the blockchain. We will discuss and also present our solutions to these problems.

3.1 User Management

To send transactions on the Ethereum blockchain you need to have an address you are in control of. Since addresses are derived from private keys you will first have to create your own[13, p. 22] and secure it with a password through one of two ways:

1. Wallet Application: Use an Ethereum wallet application to create your keys and addresses. Keys will usually be saved locally using this method.
2. Wallet Service: Use a web service to create your keys and addresses. May provide a way to create printable paper wallets of your keys. Paper wallets have a QR code which can be used to collect Ethereum payments. To access them however you will have to import your wallet into a wallet application.

The use of a wallet application comes with the obvious bonus that you are the only one at any one time that is in possession of your keys. Should you lose them by having your computer stolen or because of a hard drive failure, you will forever have lost access to that account. Unless you back them up, which is highly recommended.

Wallet services however let you create your credentials without having to install anything. In addition to that, if you implement a way to backup encrypted keys in your wallet service, a user is now not bound to a specific machine. However, this requires trust, which goes against what the cryptocurrency movement is trying to change.[13, p. 24] To achieve something similar with offline keys, you would have to always move your private keys to the device that you want to access the blockchain from.

Despite the problems of owning credentials you should not be in possession of, regardless of the fact that they are encrypted, we still chose a design where we create the users wallet for them and save them online. The way this improves the user experience is something that is very helpful for this project. Though it needs to be stated that this is only possible because our use case already requires some level of trust from our users.

3.2 Updates

The blockchain represents a record of transactions. Once a transaction has been included on the blockchain it cannot be taken back. For that to happen everyone that is participating would have to agree on rolling back a certain transaction. This is something that is not going to happen very often except for larger scale attacks or other kinds of problematic scenarios.[13, p. 42]

Since deploying a smart contract and practically everything that changes the state of the blockchain is a transaction, it is not really possible to change what is already publicly available. Updating software now becomes a difficult task. You can't really ask everyone that runs a node to roll back the deployment of a contract to upload an updated one. In order to still have a way to provide bug fixes or new features you need a system that allows changes or redirects to updated components. Implementing a contract provider[24] similar to the one implemented in the Share&Charge platform allows updates to other parts of the system. The contract provider holds references to other smart contracts, which can then be changed to point to newer versions of those contracts.

3.3 Restrictions

Smart contract development is in many ways just like general software development. There is however one aspect that is even more important when writing smart contracts than normally. Since you cannot change what is already out, you have to keep your smart contracts as simple as possible if you do not want to run into problems caused by bugs later on. Additionally the Solidity language — the currently supported programming language for the Ethereum virtual machine — does not support all the features one might expect as a developer these days. In some ways this restricts what you can do in a positive way. Though it cannot be ruled out that this makes development and possibly also your business logic more complicated than it needs to be.

To mitigate some of these problems the contract provider introduced earlier is used to mediate messages between contracts and also allows each contract to

have a very specific function, making the system modular. Additionally you will possibly have to implement your own data structures if basic arrays and maps are not sufficient.[17, 18, 16]

4 Outline of Design and Implementation

There are a number of details in the implementation that have big implications for a blockchain based application. These decisions relate to nodes in the system, access to it, as well as how smart contracts behave. This section will provide an overview of the overall application design and its realisation. A more detailed look at specific interactions and implementations can be found in the documentation diagrams that accompany this project's code.

4.1 System Layout

On a high level, the project was implemented with a Model-View-Controller architecture in mind. In terms of what technology was used for each one, the view is a relatively simple web browser based interface, while the controller and model aspects are handled by smart contracts. However, that distinction is not perfectly met all the time since account management is handled by another party. There are also instances where the view-layer needs to access the data-layer because of shortcomings in the solidity programming language. A PHP server with a database connection is used in addition to the controllers implemented in solidity.

4.2 Entities

To get a basic understanding of the system, we need to have a look at the entities that interact with each other (cf. Figure 1).

First, we have students. A student takes courses, tests and submits assignments, possibly in collaboration with other students. Supervisors create and oversee courses as well as their tests and assignments. In addition to that, they grade submissions made by students. Once this has been done a student can now go on to view their records and get an overview of how they performed on each individual test and assignment for each given course.

4.3 Model-View-Controller Pattern

4.3.1 Model

One part of our model, the one that represents the data that is acted upon, is implemented in smart contracts.

In Figure 2 we can see the contracts that are part of our model. They are part of the data package and have names that end in "DB" as a way to symbolise that they themselves and the data they hold, was modelled with databases and normalization in mind. Since solidity doesn't support the notion of packages like

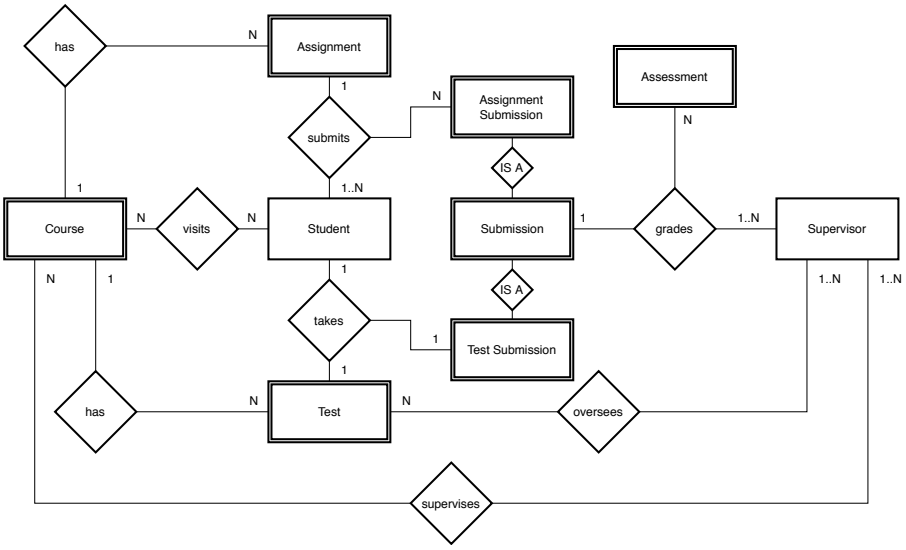

Figure 1: Entity relationship model detailing the system and its participants.

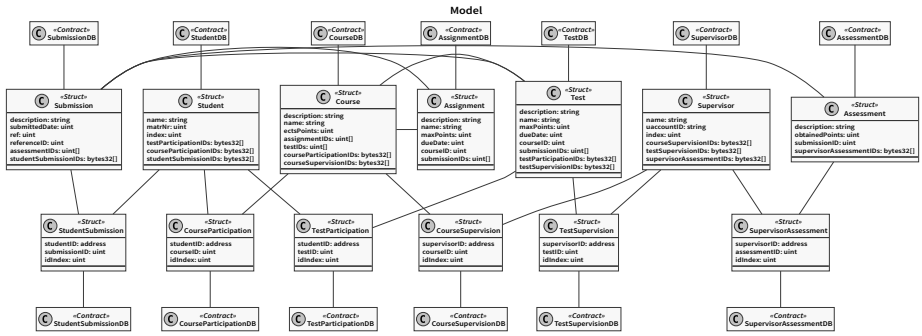

Figure 2: *DB contracts.

Java does, the use of the word 'package' here is meant to represent structuring of code and separation of concerns.

Column	Type	Null	Constraint
matrikelnr	int(16)	No	Primary
address	varchar(42)	No	Unique
pwhash	varchar(255)	No	-

Table 1: Student table.

The second part of our model is made up of a database with two tables for students and supervisors (cf. Table 1, Table 2). To keep track of all the

Column	Type	Null	Constraint
uaccointid	varchar(128)	No	Primary
address	varchar(42)	No	Unique
pwhash	varchar(255)	No	-

Table 2: Supervisor table.

users in the system and provide authentication without requiring manual credential management, a system for registration and authorization needs to be in place. A database with tables for the different kinds of actors, holds the relevant information like their ethereum address, user id as well as data related to their password. Since special care needs to be taken with passwords, only their stretched and salted hashes are saved.

4.3.2 View

Our view-layer is made with JavaScript as well as HTML5. JavaScript is needed to talk to the controllers and HTML5 was chosen because it provides basic input validations on data types like numbers as well as text. The Model-View-Controller pattern became a little bit problematic here because on certain calls the view-layer calls directly into data located in the model. This is because of limitations inherited by the solidity programming language used to implement the model-layer.

4.3.3 Controller

Finally our controllers are also implemented as smart contracts.

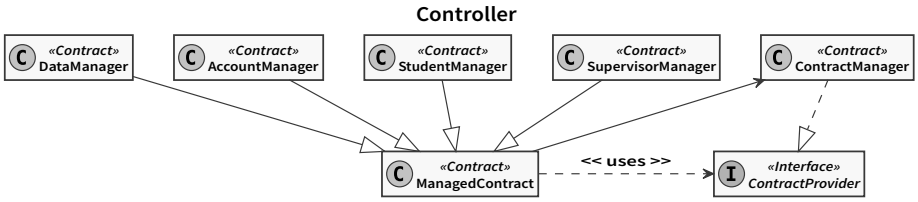

Figure 3: Manager contracts.

The contracts that have controller responsibilities can be seen in Figure 3. They are part of the manager package and have names that end in "manager". These are the contracts that the view-layer interacts with. When the view needs to present data, it is a controller contract that fetches the information from the model, acts upon it if necessary and passes it to the view.

4.4 Nodes

The business logic that was implemented in our system is run by smart contracts, who in turn need to be deployed on the blockchain. When deployed these smart contracts run on nodes in the blockchain network. A node can refer to a participant in a network as well as a special program that makes it possible to talk to the blockchain like in our case. There are a number of different tools available. Parity was chosen for the main duration of development. It provides an interface for smart contracts in addition to an easy way to set up a development chain. For running the completed system, Geth is used.

4.5 Account Management

To interact with the blockchain every user needs to have an account there. Giving them direct access to the nodes in the system, as well as having them run their own node, are some of the options that allow them to talk to the blockchain on their own. To do that most of them would also have to use some sort of wallet application, which manages their credentials and transactions for them. Others could use command line interfaces to achieve the same goals. However, these are not desirable solutions since they would require a lot of know-how from the users. It would also restrict mobility for the non-technically minded users, since we cannot guarantee that every device has the right applications pre-installed.

To solve that, a separate mechanism that stores login details and facilitates authorization of actions is implemented. A database keeps track of users credentials including blockchain related account information that is generated upon registration. PHP and its password hashing API is used for passwords and credentials are cached in a local session when a user logs in. The decision to cache credentials locally was made to reduce friction when interacting with the blockchain, since a user already needs to wait some time for their actions to be completed and recognized by the chain.

4.6 Smart Contracts

As evidenced by the section on the Model-View-Controller pattern, our system's core is made up of smart contracts. These smart contracts are organized and designed in a way that separates each contract's responsibilities from others.

4.6.1 Manager Contracts

The manager contracts provide all the functionality our system is capable of. There is a StudentManager contract that makes it possible for students to submit assignments and test. It has functions for signing up students for courses as well as functions to determine a students performance.

Supervisors interact with the SupervisorManager contract. They use it to create courses, assignments, tests and also to grade students.

Then there is the DataManager contract. This one is used by all the actors in the system and can to some extent also be used by the public. It is responsible

for providing information on all courses and tests as well as information on the ones a user is registered or responsible for. It can also show data about a specific submission and the grade associated with it.

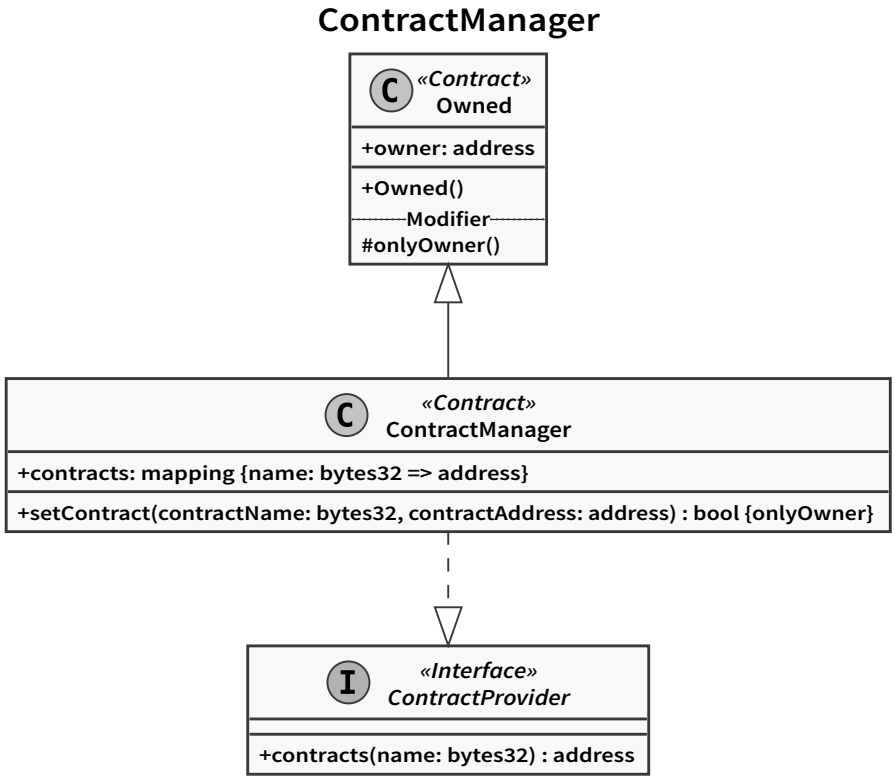

Figure 4: ContractManager contract.

The smart contract that ties everything together and makes the system modular as well as upgradeable is the ContractManager (cf. Figure 4). The ContractManager is used to keep track of all the other contracts and can be asked to provide a reference to a specific contract. This makes it possible to call functions of other contracts inside a contract. The way this is achieved is by having a mapping that maps a contracts name to an address. For this to be possible the contract that is being mapped, has to have special properties. It needs to extend a contract called ManagedContract, which holds a reference to the ContractManager itself. This way it always knows where its manager is and the manager knows where all other contracts are. Additionally, Managed-Contract provides some minimal access control for functions through the use of modifiers. Its permission modifier checks that the caller of a function, is actually the one that is allowed to call it. It does that by asking the ContractManager for the address of the contract that is allowed to call the function and compares

it to the caller's address.

4.6.2 *DB Contracts

To save all the values the system uses, some of the contracts that have been implemented practically only concern themselves with saving and providing values. Each contract is responsible for a different kind of dataset. They are designed in a way that allows data to be accessed sequentially as well as through direct access with some sort of identifier.

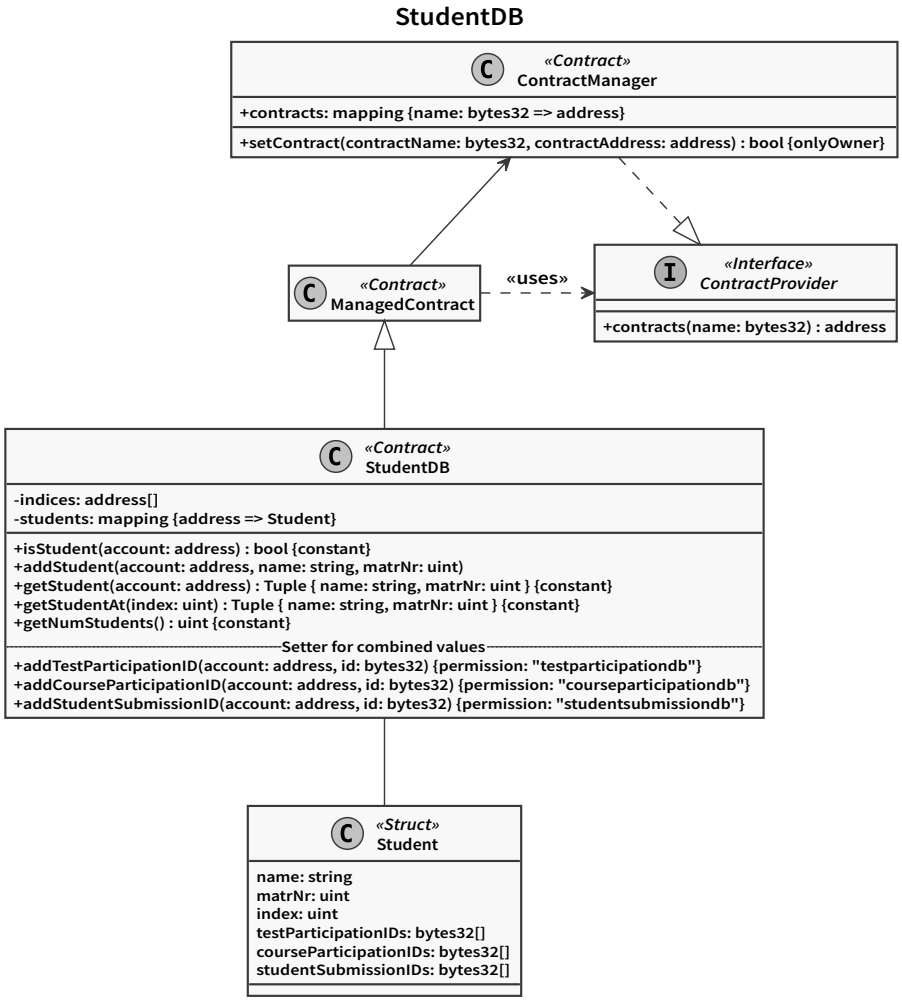

Figure 5: StudentDB contract.

The StudentDB contract for example, implements a mapping of address to

students and keeps track of the keys for the mapping in a separate array (cf. Figure 5). Additionally a student object also knows where its key is located to keep referential integrity. This provides a way for delete functionality to be implemented if it is needed down the road.

4.6.3 Utility Contracts

Then we have utility contracts that are not instantiated but only every extended. One such contract is the ManagedContract that was mentioned earlier. The other one used in this system is the Owned contract. It has functionality that transfers ownership to the creator of a contract. This way the functions that are tagged accordingly with modifiers are only callable by the owner of the contract.

4.7 Interface

Our user interface is mostly made of HTML5 web pages that handle user input, call relevant functions of our contracts and display data. It is hosted by a PHP server which also provides functions for interacting with the user database. This allows us to implement features like user search without having to change our smart contracts and also makes them leaner as a result. JavaScript is used to tie everything together. It calls functionally in PHP scripts through AJAX calls, makes it possible to talk to our node through the web3 JavaScript library and is also used for setting up the complete system.

4.7.1 System Setup

In order to make our user interface work, we need to deploy our contracts to the blockchain first. For this, a custom solution was built that can be used to deploy all the contracts. It also creates the necessary bindings for our interface and compiles these in a file that is then shipped and used as the API. It contains the contracts' ABI as well as the addresses they were deployed to. For debugging purposes all contracts are included and callable. This gives us access to the *DB contracts to inspect their data without having to resort to blockchain explorers.

4.8 Use Cases

The system created during this project set out to replicate the way universities operate at a basic level. Students attend courses to complete assignments and tests handed out by supervisors, with supervisors grading the students' performances. To keep track of the goals, a few use cases were defined in advance.

4.8.1 Summaries

U0 Specify course details
A supervisor defines the details of a course (see u:find[7])

U1 Register for course
A student registers themselves for a currently offered course

U2 Binding participation in a course
A student signs a binding agreement to participate in a previously selected course

U3.1 Hand out assignment
The course's supervisors give out an assignment

U3.2 Hand in assignment
A student completes an assignment that is part of a course

U4 Take test
A student takes a test that is part of a course

U5 Grade submission
A supervisor grades submissions according to assessment criteria

Optional

U6 Search academic records
A student searches their academic records for specific items

U7 Create report of academic records
A student creates a detailed report of their academic records

4.8.2 Details

Although most use cases have been replicated in some form, they have also changed a little bit as the project took shape. An error that was made during the project's conception for example, was that use case U7 (cf. Section 4.8.1) was defined as optional, which is not correct since that was supposed to be the original goal of this work. Therefore it also makes sense to look at this use case in detail.

The way this use case was implemented, is through combinations of processing on the view side in the JavaScript interface as well as on the controller side through the respective manager contracts (cf. Figure 6). The user interface requests a list of the students courses and parses through all of them to generate a detailed report of passed assignments and tests. It also aggregates the grades a student has received and displays a summary of all their academic records. This is made possible, by having the manager compute and verify all the data like checking whether a student has successfully handed in every assignment necessary and checking for the best grade they have received on one. This is done by the smart contracts and is essentially free since no state needs to be changed and therefore no transaction needs to be created. To see how all contracts interact with each other and how the data is presented to the view, a more detailed sequence diagram is provided in the appendix section.

5 Discussion

This project was a first foray into the use of a blockchain and smart contracts for an education specific application. As such it merits discussion of its accomplishments as well as shortcomings.

5.1 Achievements

Interacting with the blockchain requires users to have a comparatively better understanding of credentials than traditional computing paradigms demand. To exchange money for example, you need the address of the recipient. Since addresses are basically random 40 character long strings in Ethereum's case, this becomes relatively hard to remember compared to traditional user ids. With that in mind, this project set out to try to abstract the blockchain nature of the application away. It tried to provide a user experience that resembles applications that are used today. To achieve that goal, a web application was implemented that doesn't rely on plug-ins or other programs installed on a user's device. It was designed to provide mobility similar to systems that are currently used by universities. This was made possible by tying a user id to the Ethereum address that is generated upon registering with the service. As a side effect other users now become discoverable because they can be looked up by their easily rememberable id.

To sidestep the problem of making users need to install additional programs to interact with the blockchain, a choice to expose transaction signing capabilities through the node was made. The web application talks to the node and authorizes every transaction with a user's Ethereum address and password for their key. To not have the user input their data on for every transaction they want to make, their credentials are cached in the browser's web storage API. This is done locally on login and only for the duration of one session. The data cannot be accessed by other tabs or domains, which also prevents data from hopping from a secure HTTPS connection to a non-secure HTTP only one. In the real world, sessions and cookies would probably be a better fit than local caching of sensitive data. A different approach would be to have every user authorize every action with their password.

By leveraging these two mechanisms the system is able to function without plug-ins or additional programs and mostly works like traditional web services. However, this approach also has a few disadvantages. One of them being, that having every user's credentials in the cloud is problematic from a decentralisation standpoint since they lose control of their data. This is partly mitigated by the nature of our application. Universities already require some trust by students and other actors, so centralisation is not that much of a problem.

This project was also able to implement almost all of the specified use cases. Use cases where some of their description was not precise enough, was mostly handled in a way that favours students in their use of the system. For example, assessments made by supervisors can only be corrected by awarding more points since a user is always shown their best grade on a submission. This is mostly

because the system currently doesn't support changing or removing data.

5.2 Lessons Learned

The novelty of blockchain technology and smart contracts has led to a few problems during the development of the system. Many tools that are used during smart contract development are still in development themselves. Some got discontinued and completely replaced by different applications. The only currently supported integrated development environment is the web application called 'remix' which is also known as browser-solidity. It can be used offline and also to edit local files in conjunction with another tool called 'remixd'. Setting it up and operating it reliably can be problematic since remixd is still in alpha and could cause data loss without sufficient backups. It can be worthwhile however, since remix provides a relatively powerful debugger that can freely move in time.

Testing and debugging smart contracts in general, can be a time consuming task. This is because of the work flow inherent to smart contract development. You start by writing the contract, deploy it and then get the information like its ABI and the address it has. It is only then, that you can start to interact with it. If you are working in remix you can use its debugger to step through the functions of your smart contract. Otherwise you are bound to do debugging by using events as a substitute for exceptions and setting up a listener for those that you want to capture. Testing multiple contracts now requires deploying it to the blockchain, setting it up for interaction, debugging it and doing that again for every single contract. Additionally, every change that is made requires these steps to then be repeated as a whole for every affected contract.

Another consideration that needs to be made when developing a system that uses the blockchain, is the time it takes for a transaction to be included on the chain. During development this is not a problem since a chain can be set up in such a way that transactions happen instantly. On a production chain however, this cannot be guaranteed. Transactions in a proof of work based blockchain usually take a few seconds or minutes to be incorporated. Therefore the application needs to be designed to take these delays into account. Listeners and asynchronous programming paradigms help in these kinds of scenarios.

Gas cost is another detail that needs to be kept in mind. More generally this refers to the complexity and the amount of work a contract does. In Ethereum, smart contracts run on the virtual machine that is run by everyone in the network. Since computing power isn't free, the cost of running smart contracts needs to be calculated in some way and paid for. This is where gas costs come in. Every instruction in the solidity programming language has costs associated with it, called gas. Therefore costs for every function as well as the creation of a contract can be quantified. In Ethereum gas is paid for by offering ether in exchange for executing a transaction on someone's behalf. The system employed in this thesis is made in way that no one has ether, which means no one can really pay for transactions. To mitigate this problem somewhat, transactions are made free in our system by setting the gas costs to zero for every node — or more accurately, miner. This allows transactions to go through without

actually having to pay for the computing that is done. However there is still a problem. A function can only be so complex, as in consume a fixed amount of gas, before it simply gets rejected by the network even if the gas was set to not cost anything.

How do you implement complex business logic given these restrictions? The first thing that has to be done, is to look at what needs to be on the blockchain and what doesn't. If it's possible to delegate tasks to different systems, that is something that should be considered since it lessens the amount of work and therefore the gas a contract needs. Another thing that can be done is to try to flatten out logic by avoiding loops. Loops that operate on arrays that grow over time can become quite costly if not accounted for. Additionally complex contracts make programming trickier since you can run into problems with the compiler or other tools used during development. For this reason making use of all the functions solidity provides becomes key. You can avoid writing unnecessary code by leveraging automatically created getters for public variables for example.

Lastly, time management becomes especially important with all these considerations that need to be made for blockchain applications. Realizing the shortcomings of different tools as well as getting a better understanding of their capabilities and how they compare to each other, is something that needs to be factored in during a project's conception or design phase.

5.3 Outlook

The future promises a lot of potential for growth for smart contracts and the Ethereum Foundation itself still has a few milestones left that will significantly impact smart contract development. Though this project has only implemented a basic proof of concept and many things in Ethereum are constantly changing, there are a few key areas in this project that could be improved upon even now. The first one concerns updatability and extendibility. Some work has already been done here as evidenced by the contract manager. However, the contracts themselves need to be made updatable as well. The *DB contracts for example need to be extended in a way that makes it possible to export and import their data to and from newer versions of themselves. There should also be more use of events in state changing functions of smart contracts in general. This would make it possible to get a better understanding of the current status of a transaction and would also make error handling easier.

From a feature standpoint, a search function like the one described in U6 (cf. Section 4.8.1) needs to be implemented in the interface. The user interface should also be designed in a way, that authorization of transactions with passwords becomes more transparent. It should be left up to the user to decide, whether they want to authorize requests manually, authorize all of them during given time frame by making the system remember the password or maybe even organize and authorize them in groups. One could also look into how transactions can be created more easily, without having to query for gas costs first. Although it would require local keys, a way to create offline transactions that

would be sent pre-signed, could be preferable to the approach employed here.

With respect to the goal of the project in general, the smart contracts employed here could also be made smarter. The possibility of providing more decisions and more tunable parameters during the setup phase should be explored. This would make it possible to have much more automated contracts. In addition to that, the system could also be extended by features like certificate issuance and verification, or others similar to those that were talked about in the related work section. Ultimately, there is also going to be a need for research on sustainability of current proof of work based chains and alternatives. [36, 37, 15]

6 Conclusion

This thesis set out to explore the possibilities of creating a blockchain application for the educational sector that gives students the ability to get a detailed report of all their academic records. Although there are many design considerations that need to be made when creating such a system, it was shown that it is possible. Furthermore the analysis of related work suggests that there hasn't been a lot of research on this kind of use case yet. This unique position and approach to the general problem this project was trying to tackle, has also caused problems to come to light that are associated with the tools that were employed. In general it can be said that trying to solve problems with blockchain technology and smart contracts is a high effort undertaking. Through the duration of the project it has become clear that future work in the area of applying blockchain technology to the educational sector as well as work on the technology itself is still needed.

References

[1] About blockcerts. Retrieved November 18, 2017 from `http://www.blockcerts.org/about.html`.

[2] About the ethereum foundation. Retrieved November 18, 2017 from `https://www.ethereum.org/foundation`.

[3] Basic attention token. Retrieved November 18, 2017 from `https://basicattentiontoken.org/`.

[4] Digital certificates project. Retrieved November 18, 2017 from `http://certificates.media.mit.edu/`.

[5] Gnosis. Retrieved November 18, 2017 from `https://gnosis.pm/`.

[6] Horizon state. Retrieved November 18, 2017 from `https://horizonstate.com`.

[7] u:find. Retrieved January 26, 2018 from `https://ufind.univie.ac.at/de/index.html`.

[8] Accenture and microsoft add blockchain tech to biometrics id platform. *Biometric Technology Today 2017*, 7 (2017), 12.

[9] Bogner, A., Chanson, M., and Meeuw, A. A decentralised sharing app running a smart contract on the ethereum blockchain. In *Proceedings of the 6th International Conference on the Internet of Things* (New York, NY, USA, 2016), IoT'16, ACM, pp. 177–178.

[10] Chanson, M., Bogner, A., Wortmann, F., and Fleisch, E. Blockchain as a privacy enabler: An odometer fraud prevention system. In *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers* (New York, NY, USA, 2017), UbiComp '17, ACM, pp. 13–16.

[11] Chen, Z., and Zhu, Y. Personal archive service system using blockchain technology: Case study, promising and challenging. In *2017 IEEE International Conference on AI Mobile Services (AIMS)* (June 2017), pp. 93–99.

[12] Crosby, M., Pattanayak, P., Verma, S., and Kalyanaraman, V. Blockchain technology: Beyond bitcoin. *Applied Innovation 2* (2016), 6–10.

[13] Dannen, C. *Introducing Ethereum and Solidity: Foundations of Cryptocurrency and Blockchain Programming for Beginners*. Apress, 2017.

[14] Giancaspro, M. Is a 'smart contract' really a smart idea? insights from a legal perspective. *Computer Law & Security Review* (2017).

[15] Herlihy, M., and Moir, M. Blockchains and the logic of accountability: Keynote address. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science* (New York, NY, USA, 2016), LICS '16, ACM, pp. 27–30.

[16] Hitchens, R. Enforcing referential integrity in ethereum smart contracts. Retrieved November 6, 2017 from `https://medium.com/@robhitchens/enforcing-referential-integrity-in-ethereum-smart-contracts-a9ab1427ff42`, April 2017.

[17] Hitchens, R. Solidity CRUD: Part 1. Retrieved November 6, 2017 from `https://medium.com/@robhitchens/solidity-crud-part-1-824ffa69509a`, February 2017.

[18] Hitchens, R. Solidity CRUD: Part 2. Retrieved November 6, 2017 from `https://medium.com/@robhitchens/solidity-crud-part-2-ed8d8b4f74ec`, February 2017.

[19] Hoy, M. B. An introduction to the blockchain and its implications for libraries and medicine. *Medical Reference Services Quarterly 36*, 3 (2017), 273–279. PMID: 28714815.

[20] IBM. Sony and sony global education develop a new system to manage students' learning data, built on ibm blockchain. Retrieved November 8, 2017 from `https://www-03.ibm.com/press/us/en/pressrelease/52970.wss`, August 2017.

[21] Jentzsch, S. Share&charge smart contracts: the technical angle. Retrieved November 8, 2017 from `https://blog.slock.it/share-charge-smart-contracts-the-technical-angle-58b93ce80f15`, April 2017.

[22] Lemieux, V. L. Trusting records: is blockchain technology the answer? *Records Management Journal 26*, 2 (2016), 110–139.

[23] Meitinger, T. H. Smart contracts. *Informatik-Spektrum 40*, 4 (Aug 2017), 371–375.

[24] Monax. Tutorials — Solidity 1: The five types model. Retrieved November 6, 2017 from `https://monax.io/docs/solidity/solidity_1_the_five_types_model/`, 2017.

[25] Nazaré, J., Hamilton, K., and Schmidt, P. What we learned from designing an academic certificates system on the blockchain. Retrieved November 18, 2017 from `https://medium.com/mit-media-lab/what-we-learned-from-designing-an-academic-certificates-system-on-the-blockchain-34ba5874f196`, June 2016.

[26] Ølnes, S. *Beyond Bitcoin Enabling Smart Government Using Blockchain Technology*. Springer International Publishing, Cham, 2016, pp. 253–264.

[27] Ølnes, S., Ubacht, J., and Janssen, M. Blockchain in government: Benefits and implications of distributed ledger technology for information sharing. *Government Information Quarterly* (2017).

[28] Peck, M. E. Blockchain world - do you need a blockchain? this chart will tell you if the technology can solve your problem. *IEEE Spectrum 54*, 10 (October 2017), 38–60.

[29] Raju, S., Rajesh, V., and Deogun, J. S. The case for a data bank: An institution to govern healthcare and education. In *Proceedings of the 10th International Conference on Theory and Practice of Electronic Governance* (New York, NY, USA, 2017), ICEGOV '17, ACM, pp. 538–539.

[30] Rohaidi, N. Using blockchain for student certificates slashes admin costs. Retrieved November 18, 2017 from `https://govinsider.asia/digital-gov/patrice-choong-ngee-ann-polytechnic-campus-ecosystem/`, June 2017.

[31] Schmidt, P. Certificates, reputation, and the blockchain. Retrieved November 18, 2017 from `https://medium.com/mit-media-lab/certificates-reputation-and-the-blockchain-aee03622426f`, October 2015.

[32] Schmidt, P. Blockcerts—an open infrastructure for academic credentials on the blockchain. Retrieved November 18, 2017 from `https://medium.com/mit-media-lab/blockcerts-an-open-infrastructure-for-academic-credentials-on-the-blockchain-899a6b880b2f`, October 2016.

[33] Sony Global Education. Sony global education develops technology using blockchain for open sharing of academic proficiency and progress records. Retrieved November 8, 2017 from `https://www.sony.net/SonyInfo/News/Press/201602/16-0222E/index.html`, February 2016.

[34] Sullivan, C., and Burger, E. E-residency and blockchain. *Computer Law & Security Review 33*, 4 (2017), 470 – 481.

[35] Turkanović, M., Hölbl, M., Košič, K., Heričko, M., and Kamišalić, A. EduCTX: A blockchain-based higher education credit platform. *ArXiv e-prints* (Oct. 2017).

[36] Vranken, H. Sustainability of bitcoin and blockchains. *Current Opinion in Environmental Sustainability 28*, Supplement C (2017), 1 – 9.

[37] Vukolić, M. *The Quest for Scalable Blockchain Fabric: Proof-of-Work vs. BFT Replication*. Springer International Publishing, Cham, 2016, pp. 112–125.

[38] Xu, Y., Zhao, S., Kong, L., Zheng, Y., Zhang, S., and Li, Q. *ECBC: A High Performance Educational Certificate Blockchain with Efficient Query*. Springer International Publishing, Cham, 2017, pp. 288–304.

[39] Yermack, D. Corporate governance and blockchains. *Review of Finance 21*, 1 (2017), 7–31.

Appendix

A Setup Guide

Requirements

1. The project's code as well as its documentation.
2. Access to an Ethereum node that exposes web3, eth and personal APIs over the HTTP endpoint.
3. A suitable genesis block with sufficiently high block gas limit (check gas costs in doc folder)
4. A Solidity compiler.
5. A PHP server.
6. An SQL database.

Setup

1. Create an account on your node if you don't have one already
   `geth account new`
2. Lower gas cost to 0
   `miner.setGasPrice(0)`
3. Start miner
   `miner.start()`
4. Set up database with correct tables

```
CREATE TABLE ‘student‘ (
    ‘matrikelnr‘ int(16) NOT NULL,
    ‘address‘ varchar(42) NOT NULL,
    ‘pwhash‘ varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE ‘supervisor‘ (
  ‘uaccountid‘ varchar(128) NOT NULL,
  ‘address‘ varchar(42) NOT NULL,
  ‘pwhash‘ varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

ALTER TABLE ‘student‘
  ADD PRIMARY KEY (‘matrikelnr‘),
  ADD UNIQUE KEY ‘address‘ (‘address‘);

ALTER TABLE ‘supervisor‘
  ADD PRIMARY KEY (‘uaccountid‘),
  ADD UNIQUE KEY ‘address‘ (‘address‘);
```

5. Fill in the database's credentials in sampleconfig.ini and rename it to .config.ini
6. Compile solidity code with the helper script to a combined json file
   `./compile.sh`
7. Serve all the files in the ui folder to localhost on port 8000
8. Open the setup web page and deploy contracts to create the API file (use the previously created account to set everything up)
   `localhost:8000/blockpass/admin/setup.html`
9. Copy the API file with the helper script to the ui folder
   `./init.sh`
10. Register all contracts on the contract manager by using the register page (use the previously created account to set everything up)
    `localhost:8000/blockpass/admin/manager.html`
11. Stop serving the ui folder on localhost
12. Publish all files in the ui folder on the PHP server
13. Finally, use the system by registering students and supervisors.

B Extra Diagrams

Since it was one of the main goals of this project, this appendix section has been used to include the simplified diagram for how academic reports are generated. However, the project's source files include additional documentation and diagrams that give a more detailed look of the system's inner workings.

Follow the directions in the source's README files if you have trouble finding them.

Academic Report Generation [Simplified]

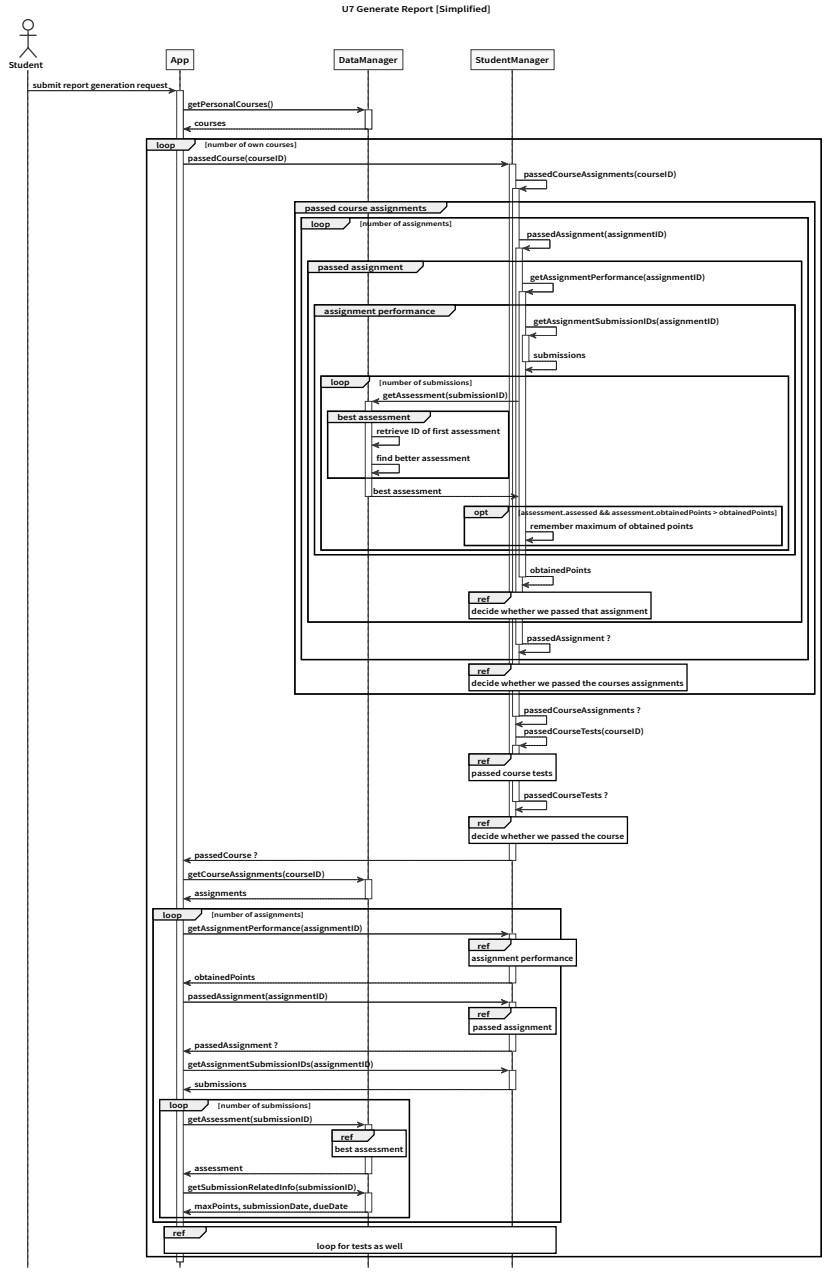

Figure 6: Simplified sequence diagram for generating a detailed report of academic achievements and activities.