[>>> Go to the german version](https://docs.google.com/document/d/1Vd-k94S423wP7_CafUJaCyJ7oz1vMCcThof2rWmFr9k/edit#)

Documentation of the Gradido Platform

English Version, Status: 03.09.2020

[General considerations](#_otmtx4tgnbpv)

[Requirements](#_il41wig21jtb)

[Avoiding multiple creation](#_6nesiw498lqk)

[Open Source only later](#_nf4rwry49flo)

[Architecture](#_jnigu22jujkx)

[Approach](#_rgd2bceuzvs8)

[Layers](#_brcy161optbs)

[Layer 1: Transaction protocol](#_lkgsv5up9fu3)

[Layer 2: Blockchain](#_fve20pw1jkj8)

[Layer 3: User accounts](#_nyzv9dqczblc)

[Layer 4: User interface](#_8buv8zinvixp)

[Software Components](#_k1n0pnqjlehl)

[Community- Server (Group Server)](#_v5omkeev7iec)

[GDT Server](#_4zbjeg6r4pmn)

[Login Server](#_x4ocorj40uxc)

[Node Server](#_79x5archfrxk)

[Illustration of Server Connections](#_tn71c2d69i3a)

# 

# 

# General considerations

## Requirements

* **Currency platform for the Gradido model** - threefold money creation 3x1000 GDD per month (Active Basic Income, Public Budget, Equalization and Environmental Fund), 50% decay per year
* **Easy to use for everyone** - use must be as easy as PayPal, i.e. e-mail address or username. Therefore private keys must be stored encrypted in the login server.
* **Worldwide scalability** - The system should be scalable worldwide, so that it can be used for financial transactions all over the world. This is achieved by dividing the system into sub-networks, each of which belongs to a group of people, e.g. a community.
* **Security** - The system must meet the security requirements for financial transactions.
* **Gradual development in milestones and sprints** - the system already runs as MVP1 on a small central platform. The next step is to run it with limited functionality on the DLT platform Hedera Hashgraph (MVP2). The entire range of functions will be developed step by step.

## Avoiding multiple creation

The Active Basic Income is created by payments to the community. This requires a manageable size of the communities (called "groups" in the source code). A community member must credibly prove the hours he or she has worked. Since the lifetime is limited, multiple creation is less critical with the active basic income than for example with an unconditional basic income.

When Gradido becomes the official currency, identity is ensured by the regional or local authorities

In addition, the development of a global digital identity is being driven forward by both the public and private sectors. That is not our task.

## Open Source only later

At a later point in time the code is to be disclosed. In order to make the initial development as easy and undisturbed as possible, we are initially developing in a closed environment.

# 

# Architecture

## Approach

* Structure in several layers, similar to the OSI model
* Use of public-key signatures for transactions similar to Bitcoin
* Use of the Hedera Hashgraph Consensus Service for the chronological sorting of transactions
* Use of google protobuf to exchange transactions with each other platform independent
* Use of C++ based servers for the critical system areas for maximum security and performance
* Use of php and javascript for good usability and faster development of new comfort functions.

### Layers

#### Layer 1: Transaction protocol

* defined as google protocol buffers
* is stored like a block chain
* should not contain personal data
* based on the Hedera Hashgraph transaction format
* signed with ed25519 signatures

#### Layer 2: Blockchain

* temporal classification of the transaction in a chain per group, e.g. by Hedera
* verify, store and make publicly available the transactions
* secured against subsequent modification, deletion and loss due to server failure

#### Layer 3: User accounts

* encrypted storage of private keys for the users
* only with the correct e-mail-password combination the private keys can be decrypted
* Account recovery/change password using a passphrase from which the private key can be calculated
* (Optional) Store the user's passphrase encrypted with Server Admin Key(s) if the user did not write down his passphrase when opening the account. Security risk!
* Signing of user transactions

#### Layer 4: User interface

* Caching and displaying transactions belonging to the user
* Create transactions
* Comfort functions such as friend lists for fast transfers

## Software Components

### Community- Server (Group Server)

Layer 4 - User interface

We aim for a decentralized network of communities, from small communities (a few 100 users) to very large communities. To make it easy for the small communities, the community server should also run on simple webspace. Therefore it does not contain any particularly security-relevant functions and is developed in CakePHP. The community admins do not need high qualifications.

To send the transactions themselves via Hedera, the community server would need the php module for grpc. This would make it more difficult for users to install the community server on a simple web space.

**Properties of the community server:**

* CakePHP
* modern
* User-friendly
* Communicates a lot with the login server
* One instance per group / community
* Can also be hosted on a simple webspace with php and mysql

### GDT Server

During the transition phase for project financing (layer 4 - user interface)

Membership fees and grants paid via Elopage or Digistore are transferred to the GDT server, where they are rewarded with a multiple of GradidoTransform (GDT). GDT are not subject to decay and can be converted to GDD at a later date.

**Properties of the GDT server:**

* CakePHP
* Connection to Elopage and Digistore
* Manages GDT for those who support the Gradido project
* Communicates with the community server via json-ajax-requests
* Currently only one instance
* Should not play a role in the fully established system later

### Login Server

Layer 3, user accounts and layer 2, block chain

Several communities can access one login server. Besides the login, it also fulfills other complex and security-relevant functions, e.g.

* Storage of private keys
* Signing and routing transactions is the main function of the login server.

For security and performance reasons the login server is developed in C++.

**Properties of the login server:**

* C++ with Poco
* safety critical
* Communicates via json-ajax-request with the community server
* Communicates via jsonrpc-request with the Node server.
* Maximum one login server per group
* can manage multiple groups
* Hosting requires ssh-access so a VPS or root server
* Server admin must be reliable and should know something about server security
* Sends signed transactions to the Hedera network

### Node Server

Layer 2, Blockchain

* C++ with Poco
* performance critical
* One node server can manage many groups
* stores the block chain of each group in a separate folder
* Connection to the Hedera network as mirror node
* Receives transactions from the Hedera network
* Provides transaction data for login and community servers
* There should be more than one per group in case one fails.
* Should be able to be used for a Supernode, which receives, checks and stores all transactions of the world (currently about 40 transactions per ms, about 650 GByte per day)

### Illustration of Server Connections

