**Requirements**

**Functional Requirements**

* GenoDist is a two-level database management system which is based on a distributed architecture. On the top level, blockchains, a distributed database, stores growing lists of records. These records provide information about how to access to data and summary of data. Each user must have access to blockchains. On the lower level, a distributed system stores the actual data on servers which must be hidden from users.
* Depending on the location of the request and content of the requested data, the most suitable server, in terms of distance and availability of data, must respond to the request.
* During an erroneous situation on a server, the flow of data continues operating via other servers.
* GenoDist encrypts the data to increase security of the system.

**Non-functional Requirements**

* *Availability*: The amount of time the system is operational shall be increased with the distributed architecture compared to centralized architecture via distribution and replication of data to servers and quick recovery procedure when a failure is detected.
* *Continuity*: The system shall be able to handle major interruptions such as power outages of a server by continuing operation on other services.
* *Portability*: The system shall be usable in different environments.
* *Recoverability*: The system shall restore the data in the event of corruption or loss by copying the replicated data from other servers.
* *Response time:* Via distributed architecture, response time shall be decreased compared to centralized architecture by locating the request location and sending the data to requester from nearest server.
* *Reusability*: The system shall be reused across multiple products to store and retrieve the data.
* *Robustness*: The system shall be able to cope with the errors during execution time.
* *Scalability*: The system shall be practical and efficient when applied to large input data by adding new resources.
* *Security*: Security problem of the distributed systems shall be solved by encrypting the data.
* *Transparency*: The block chain part of the system shall be transparent while the resources where actual data is stored shall be hidden from user.
* Resource Constraints:
  + Processor speed:?
  + Memory:?
  + Network Bandwidth:?