D7001D: Mini Project

Mirjalol Aminov, Nadir Arfi, Diana Mustakhova, Chandan Singh, Brianna Swan

24 October 2022

With the support of the Erasmus+ Programme of the European Union

EMJMD GENIAL

GREEN NETWORKING & CLOUD COMPUTING

UNIVERSITÉ DE LORRAINE

LEEDS BECKETT UNIVERSITY

LULEÅ UNIVERSITY OF TECHNOLOGY

Summary

- Methodology and Approach
- Design of System
- Performance Evaluation

Methodology and Approach

- First step, set up basic networking functionalities
- Define a chunk format suitable for encryption
- Solve the encryption decryption functionalities
- Define similarity computation function
- Decide on a language → Python
- Goals:
  - Design a distributed networking system
  - Implement a publish-subscribe functionality
- Requirements
  - Anonymously communication
  - Hide both origin and destination of data

Design of System

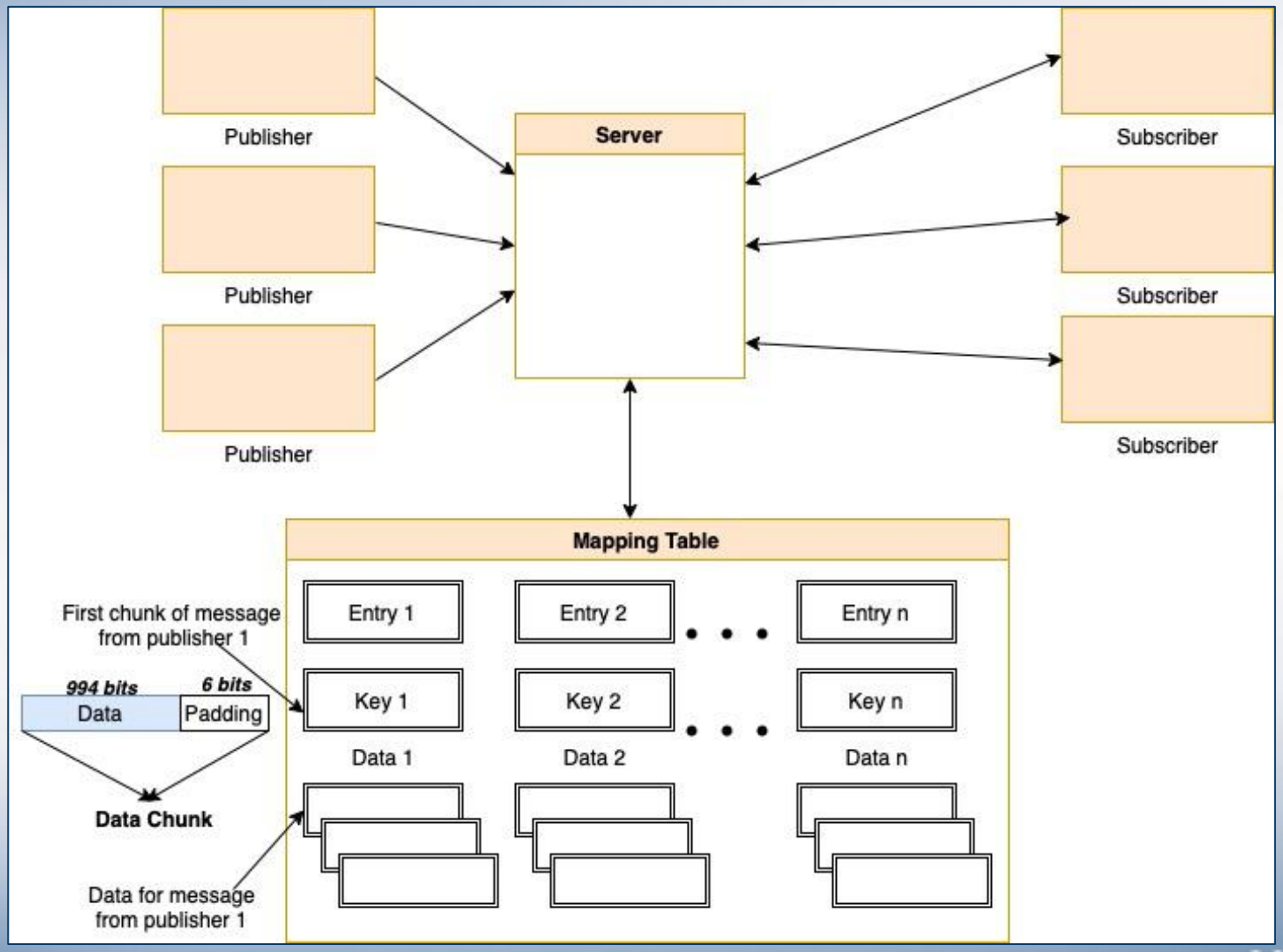
- **Client data originator (Publisher)**
  - A random ID vector of -1 and +1 with dimension 1000 is generated for the Publisher
  - Generated ID is published on the client's webpage
  - A user can input data to be published
  - Destination ID vector is embedded into data chunk itself
  - Encrypted data is sent to the known IP address of the server

- **Client-recipient (Subscriber)**
  - Knows the ID of the Publisher
  - Sends ID to Server to REQUEST all data with this label
  - Receives encrypted data
  - Decrypts the data

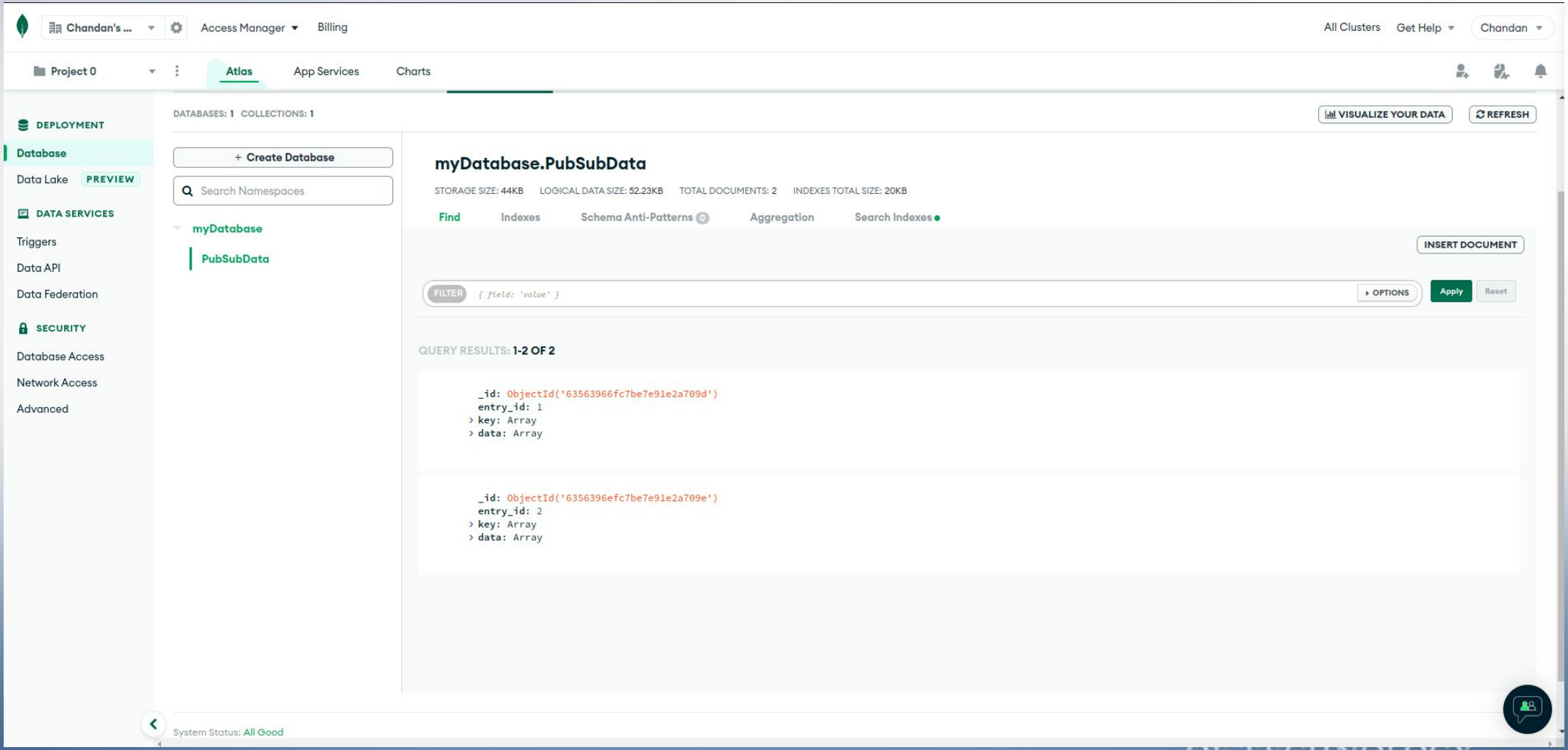
Design of System

- **Pub/Sub System (Server)**
  - Receive encrypted data
  - Store encrypted data chunks based on a similarity measure
  - Maintains **mongoDB database** to store the encrypted data
    - Initially empty
    - Maintains N buffers in memory for storing data of N users
    - if table is empty
      - Save first chunk as key and data
    - else
      - Performs associative search
      - Compute dot product with all entries
        - If result is larger than a threshold value (parameter)
        - Select entry with maximum value of dot product
        - Stores received chunk the buffer linked to this entry
  - Search for an ID and its corresponding data
  - Replies back to subscriber's request

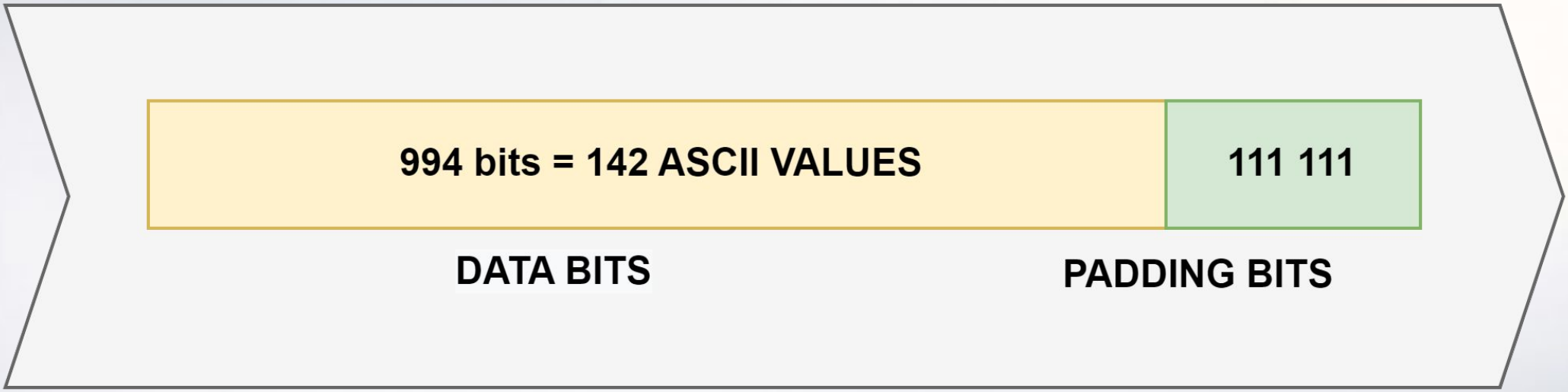
Design of System



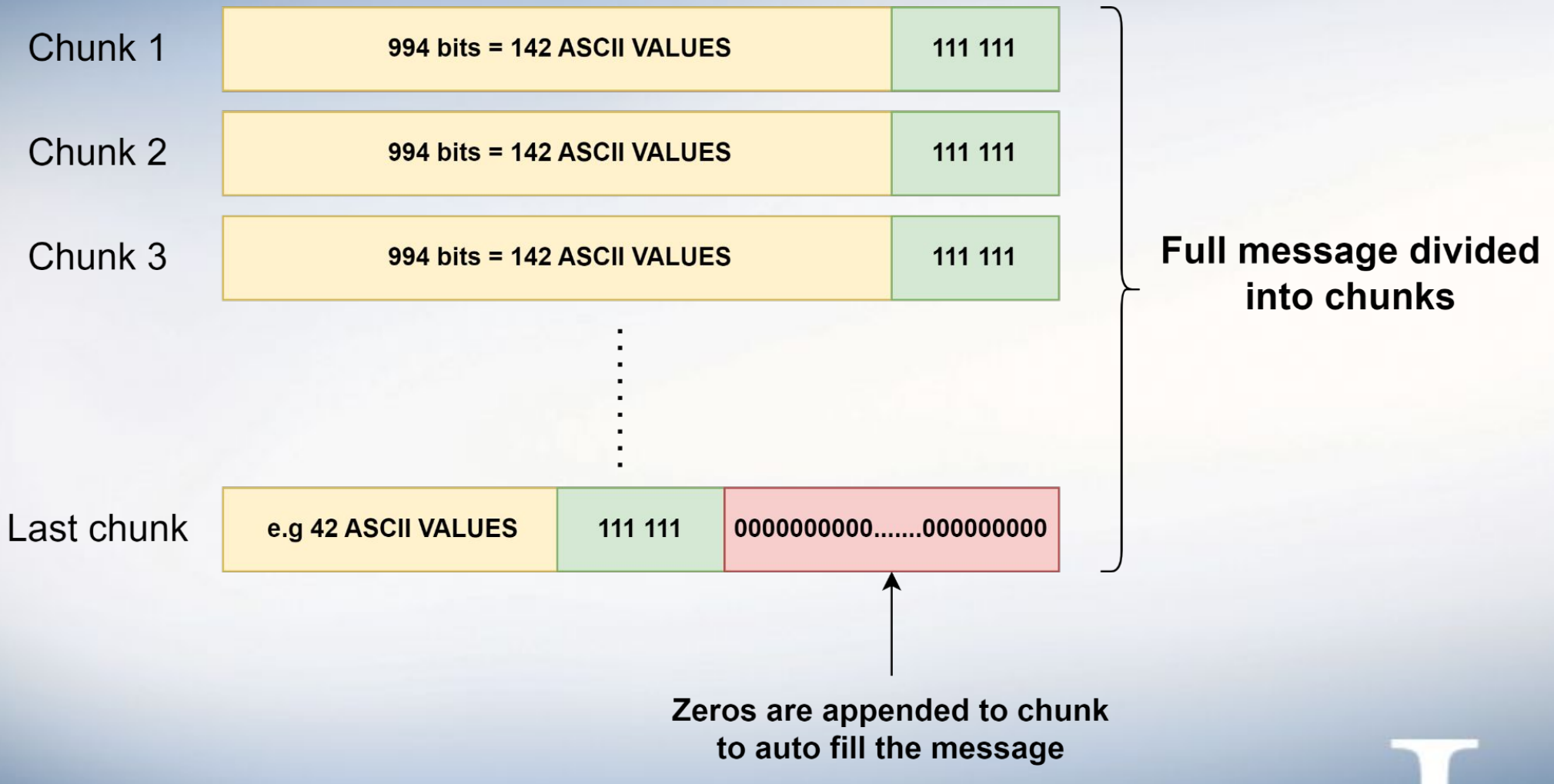
MongoDB


Chandan's ...
Access Manager
Billing
All Clusters
Get Help
Chandan
Project 0
Atlas
App Services
Charts
DEPLOYMENT
Database
Data Lake
PREVIEW
DATA SERVICES
Triggers
Data API
Data Federation
SECURITY
Database Access
Network Access
Advanced
DATABASES: 1 COLLECTIONS: 1
VISUALIZE YOUR DATA
REFRESH
+ Create Database
Search Namespaces
myDatabase
PubSubData
myDatabase.PubSubData
STORAGE SIZE: 44KB
LOGICAL DATA SIZE: 52.23KB
TOTAL DOCUMENTS: 2
INDEXES TOTAL SIZE: 20KB
Find
Indexes
Schema Anti-Patterns
Aggregation
Search Indexes
INSERT DOCUMENT
FILTER
{ field: 'value' }
OPTIONS
Apply
Reset
QUERY RESULTS: 1-2 OF 2
_id: ObjectId('63563966fc7be7e91e2a709d')
entry_id: 1
key: Array
data: Array
_id: ObjectId('6356396efc7be7e91e2a709e')
entry_id: 2
key: Array
data: Array
System Status: All Good

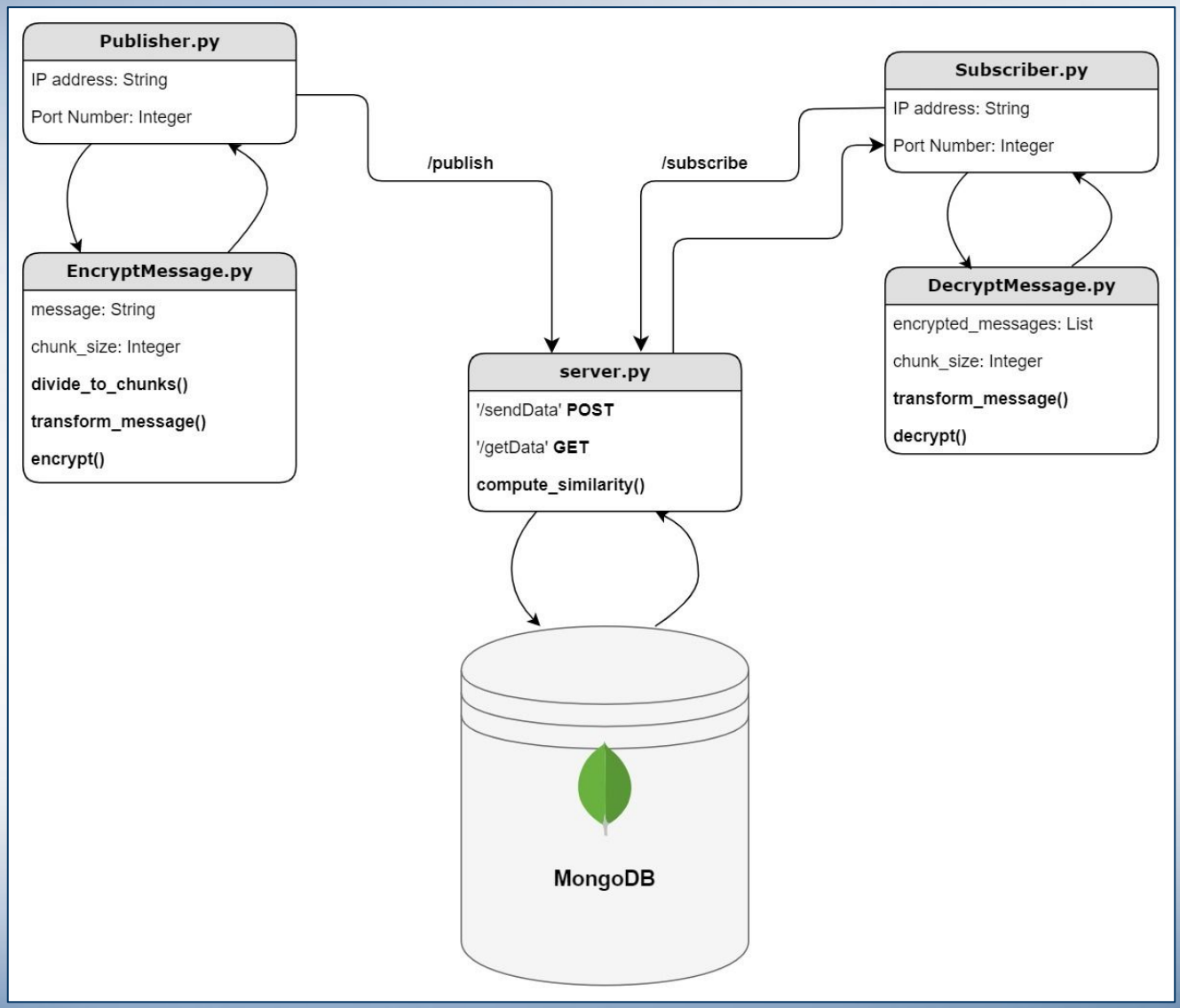
Data chunk format



Last chunk Problem

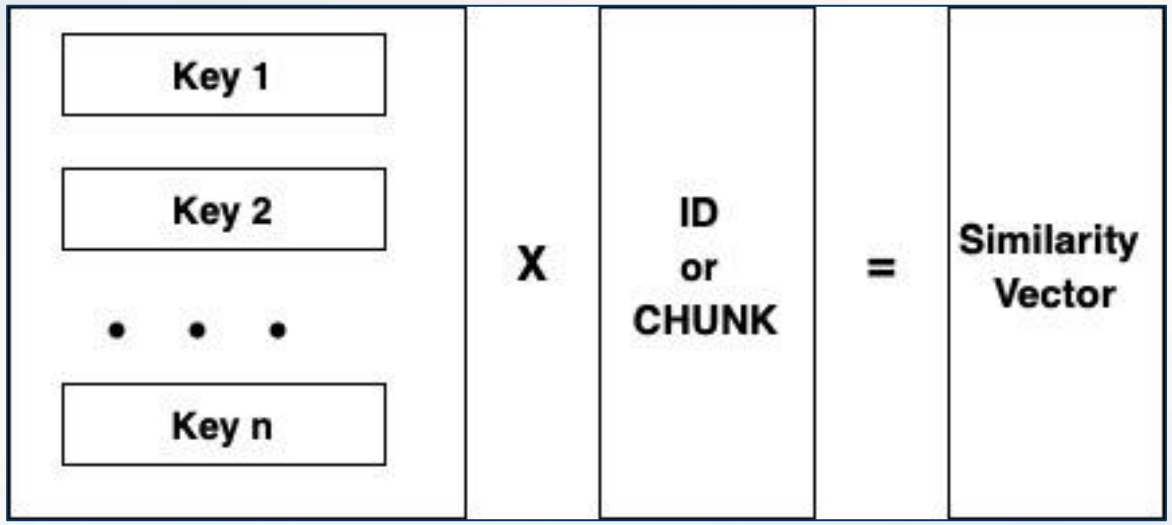


Data Flow of the System



How do we calculate similarity?

Dot Product



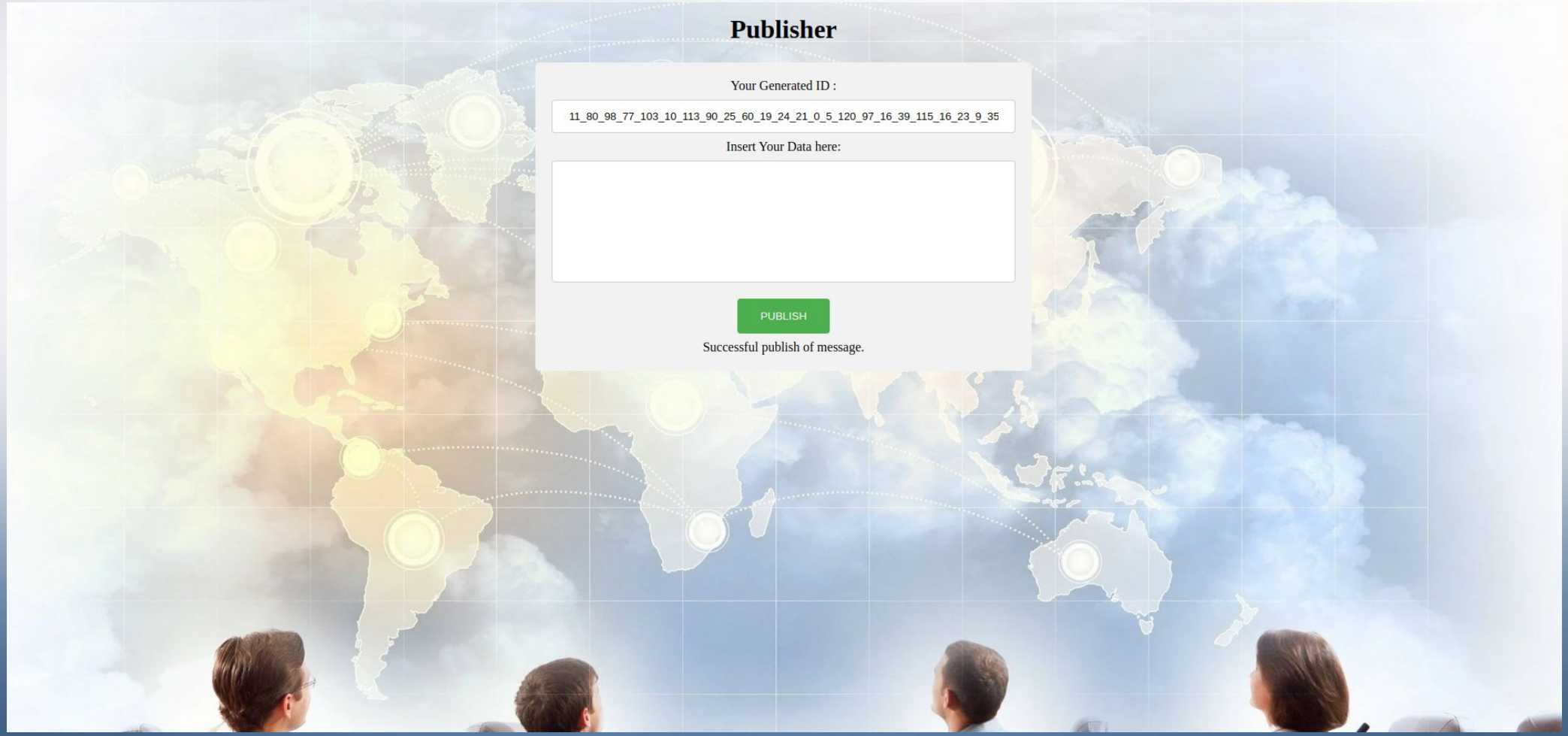
In a result we select a maximum element of similarity vector that gives a correct key

Performance Evaluation

- Achievements:
  - Well documented code
  - Fully implemented (server, publisher client, subscriber client, encryption)
  - Fully-functional and user-friendly GUI (web application)
- Room for improvement
  - Scaling the project for additional clients
  - Two-way publishing-subscribing
  - Host on AWS EC2 Instance instead of local host

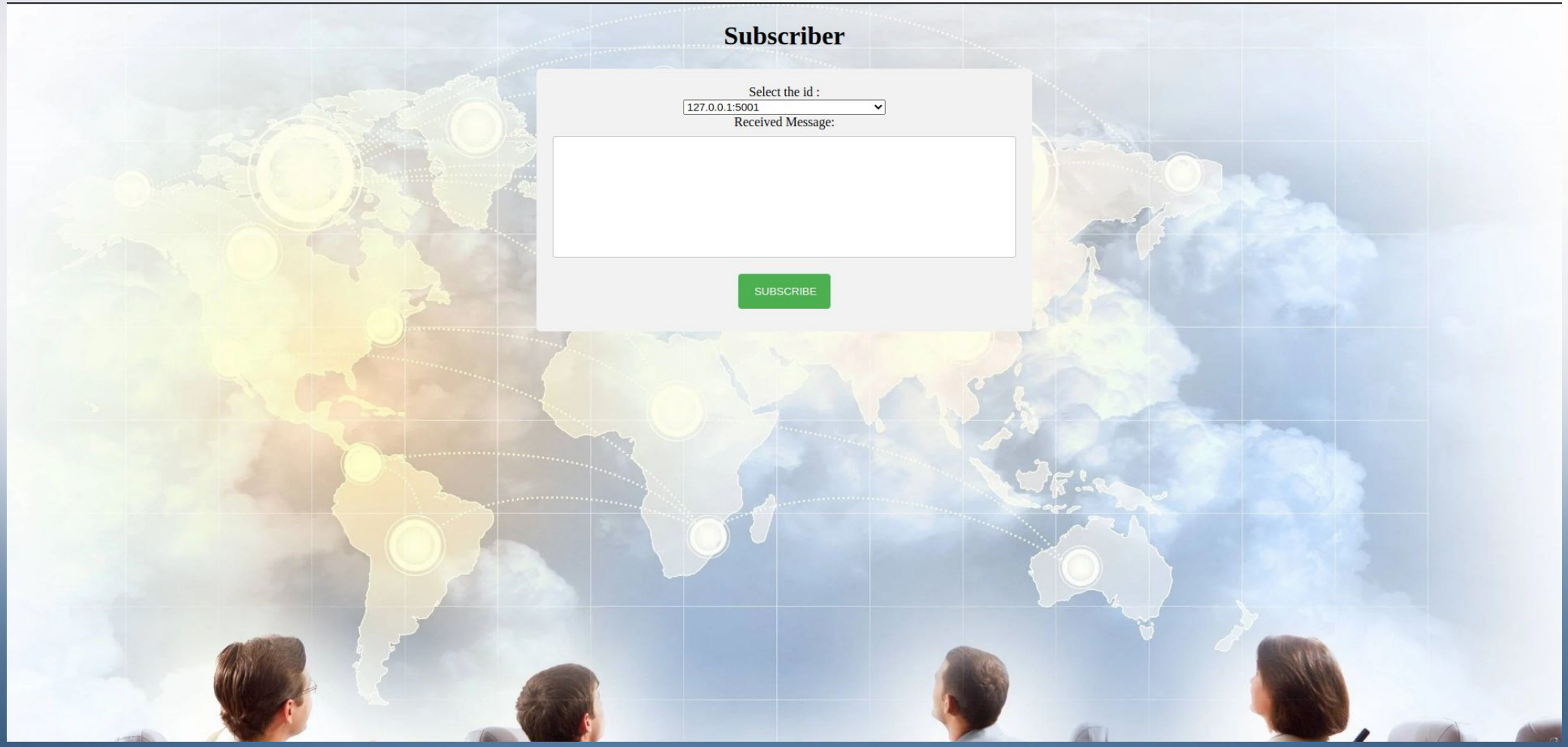
Implementation & Demo

https://github.com/TheCsr/DistributedApplication



Implementation & Demo

https://github.com/TheCsr/DistributedApplication



Thank you for your attention!!!