SMBUD 2021 - Project work 2

Aman Gabba - 10793117
Andrea Cerasani - 10680486
Giovanni Demasi - 10656704
Pasquale Dazzeo - 10562130
Vlad Marian Cimpeanu - 10606922

POLITECNICO
MILANO 1863

Contents

1 Introduction 3
- 1.1 Problem Specification 3
- 1.2 Hypothesis 3

2 ER diagram 5

3 Document diagram 6

4 Dataset description 7

5 Queries and Commands 8
- 5.1 Queries 8
  - 5.1.1 Last expiring certificate 8
  - 5.1.2 Expiration date of certificate 9
  - 5.1.3 People vaccinated by a nurse 9
  - 5.1.4 Doctor who performed an anamnesi 10
  - 5.1.5 Covid history of a person 10
  - 5.1.6 Vaccination amount per period 10
- 5.2 Commands 11
  - 5.2.1 New vaccination 11
  - 5.2.2 Expiration date update 12
  - 5.2.3 Revoke certificate 12

6 Generator database 13
- 6.1 Place generation 13
- 6.2 Certificate generation 13
- 6.3 Sanitary operator generation 13
- 6.4 Authorized body generation 13

7 Application description 14

8 User guide 15

9 Conclusion 15

10 References and Sources 15

1 Introduction

1.1 Problem Specification

The aim of this project was to design a 'query document data structure' in MongoDB for supporting a certification application for COVID-19. The database must register all the necessary information about the users including their tests and vaccination status in order to know if a certain person can be considered 'harmless' and so able for instance to visits some public spaces. The application using this database will be able to exploit all the data coming from test results and vaccination status.

1.2 Hypothesis

The assumptions taken into account are the following:

- The recovery document has been introduced in the document DB to distinguish the usual negative test result to the one did after an infection to consider a certain person healed from Covid-19. Using this document can be computationally useful because otherwise, to know if a person has a recovery certificate, it would be necessary to scan all the tests and check the relative results and dates to deduce an infection and a consequent healing. It has been also assumed that the recovery certificate doesn't get automatically released, but it is registered manually by whom of competence after a specific iter. The recovery certificate was not expressly required by the problem specification but it has been introduced because it can be used as support to calculate the expiration date of the certificate.

- Authorized bodies has been implemented as an autonomous collection and a pointer has been used in the document Place instead of a sub-document because they are few and they are repeated a big amount of times. Even if redundancy can be accepted in MongoDB, the information contained in this document are not widely used, for this reason having multiple copies of them could be inefficient.

- It has been assumed that the database only contains documents related to the Italian state. Thanks to this the government rules are well known and well defined and it allows to have the expiration date directly in the certificate document since rules changes rarely happen. In this last case the update would be a little time consuming but this choice is widely justified considering the fact that checking the validity of a certificate is one of the main reason of our database and having it directly in the document allow to access it instantaneously without any computation.

- As said in the problem description, the designed database would be mainly used to check the validity of a determined certificate. For this reason it has been decided to make a collection of certificates (with possibly more than one associated to the same person) instead of making a single one for every person containing all the relevant data. This choice allows to make the certificate access faster since they are indexed by UIC and it anyway allows to reconstruct the Covid history of a given person by looking for all his/her certificates also indexed by fiscal code. In conclusion, this choice allows to have the same database that would be obtained with the other approach but simplify the queries for which the database would be mainly used.

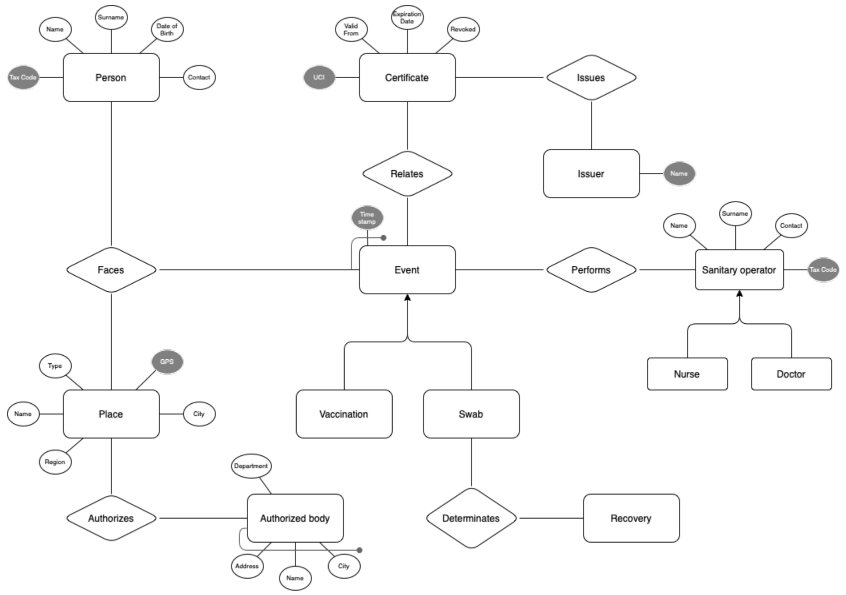
- The following database has been designed with the awareness of existence of the Covid certification, so the structure thought has been taken into consideration thanks to its efficency in the most used cases. If the same database would have to be designed without knowing the existence of the certification adding it later maybe it would have make to a different and adapted, and probably also less efficient, design.

- The following design has been chosen due to make the research of a certificate more efficient. Otherwise, by making an unique dossier for every person, finding a desired certificate by UCI would have meant finding all the people with the same name, surname and date of birth and scan all their certificates (dossier sub-document) looking at the one with the given UCI. The tax code can't be used because, in real cases, it would not be exposed in the certification due to privacy reasons.

2 ER diagram

The designed ER diagram contains the following entities: Person, Place, Authorized body, Certificate, Event and Sanitary operator. Nurse and Doctor have been designed as sub-entities of Sanitary Operator, Vaccine and Swab have been designed as sub-entities of Certificate and the Recovery entity is related to a swab.

Every person, identified by an unique tax code, can faces some events, like for instance a vaccination, in determinates places with unique GPS coordinates. Through an event a person can obtain a certificate with unique UCI useful for checking its valitidy. Every certification is issued by an authorized issuer and all the events are performed by prepares sanitary operators.

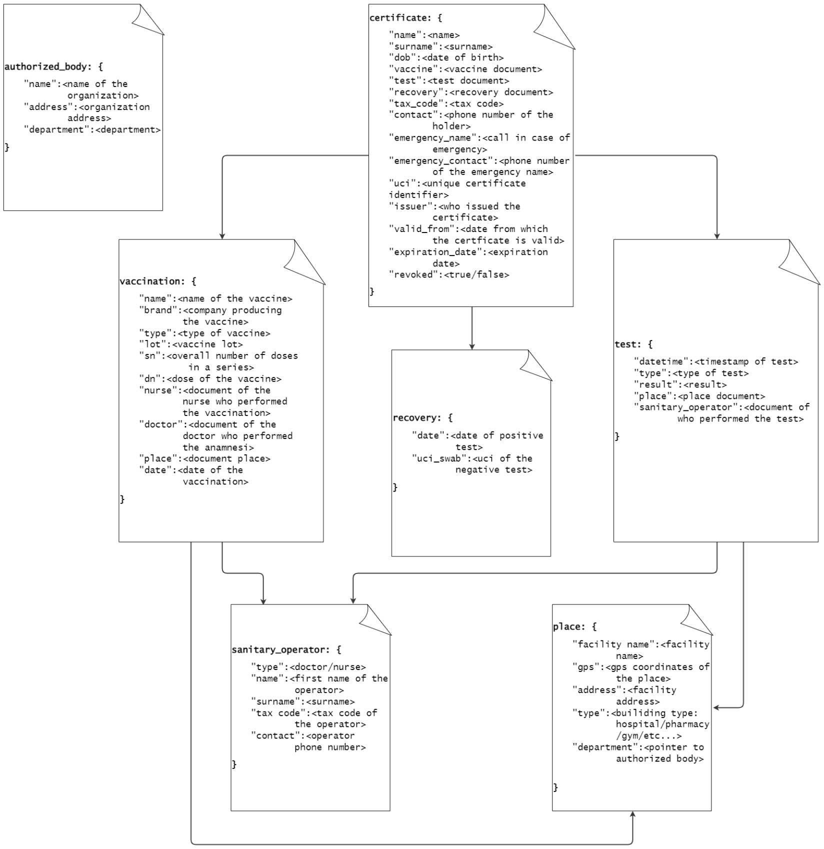


3 Document diagram

The designed Document diagram is shown below. Certificate and Authorized body are represented as collections of documents.

The certificate document doesn't represent an unique person but it is specific for a determined event. It can have one between Recovery, Test and Vaccination as sub-document and these last two have both Place and Sanitary operators as sub-documents which contain all the information related to the place and the doctor/nurse involved in the specific event.

The Recovery document has been implemented to make the recovery certificate validation easier and faster. Otherwise, doing this validation through the scan of the tests information would have been more complex due to the possibility of a person to do an unlimited amount of tests. With this design choice the time complexity of validate a recovery certificate has been reduced.



```
authorized_body: {
    "name":<name of the
            organization>
    "address":<organization
            address>
    "department":<department>
}
```

```
certificate: {
    "name":<name>
    "surname":<surname>
    "dob":<date of birth>
    "vaccine":<vaccine document>
    "test":<test document>
    "recovery":<recovery document>
    "tax_code":<tax code>
    "contact":<phone number of the
            holder>
    "emergency_name":<call in case of
            emergency>
    "emergency_contact":<phone number
            of the emergency name>
    "uci":<unique certificate
    identifier>
    "issuer":<who issued the
            certificate>
    "valid_from":<date from which
            the certficate is valid>
    "expiration_date":<expiration
            date>
    "revoked":<true/false>
}
```

```
vaccination: {
    "name":<name of the vaccine>
    "brand":<company producing
            the vaccine>
    "type":<type of vaccine>
    "lot":<vaccine lot>
    "sn":<overall number of doses
             in a series>
    "dn":<dose of the vaccine>
    "nurse":<document of the
            nurse who performed
            the vaccination>
    "doctor":<document of the
            doctor who performed
            the anamnesi>
    "place":<document place>
    "date":<date of the
            vaccination>
}
```

```
recovery: {
    "date":<date of positive
            test>
    "uci_swab":<uci of the
            negative test>
}
```

```
test: {
    "datetime":<timestamp of test>
    "type":<type of test>
    "result":<result>
    "place":<place document>
    "sanitary_operator":<document of
            who performed the test>
}
```

```
sanitary_operator: {
    "type":<doctor/nurse>
    "name":<first name of the
            operator>
    "surname":<surname>
    "tax code":<tax code of
            the operator>
    "contact":<operator
            phone number>
}
```

```
place: {
    "facility name":<facility
            name>
    "gps":<gps coordinates of
            the place>
    "address":<facility
            address>
    "type":<builiding type:
            hospital/pharmacy
            /gym/etc...>
    "department":<pointer to
            authorized body>

}
```

As shown above, every certificate has an unique certification number and contains all the information about the person to whom it belongs like for instance name, surname and tax code. It also has information about the period while it can be used like the validity starting date and the expiration date.

Certificates also have a boolean value called Revoked with a False default value, which can be switched to True in case a certification has been released due to a mistake, allowing in this way the possibility to make it unusable.

Vaccination, Reovery and Test contains the relevant information about the performance, like for instance the date of the event and the type of tests or vaccine.

Sanitary operator and Place instead have information about when an event has happened and by whom it has been performed. This information can be useful, for instance in case of mistakes or doubts, to track down the sanitary operator or the doctor who performed the vaccination or test.

4 Dataset description

The Dataset has been built by making a script in Python, using some useful packages like: `random-italian-person` that automatically generates people with random attributes (this package has been modified due to our necessity); the official `PyMongo` driver for Python for the communication with the Database since it allows to make queries directly from Python.

It has been also used a CSV document, downloaded directly from the official Italian Government github repository, of all the vaccination centres (due to the limited scale of the to make database only some regions have been taken into consideration).

CSV files for vaccines information and authorized bodies have been built by hand due to necessity with the relevant information.

5 Queries and Commands

In the following chapter all the queries and commands parameters (part of the code to substitute with desired values) will be highlighted with **magenta** bold text.

Some parameters information can be useful for different queries or commands so they are written here to avoid writing them multiple times:

- The UCI (unique certificate identifier) is an alphanumeric string and must be in the following format:
  "01ITAXXXXXXYYYXXXYXXYXYYXYXXXXXXYXY" where X is a digit and Y is a capital letter.
- The tax code ("codice fiscale" in italy) is a 16 digits alphanumeric code and follow the usual italian format.
- Dates must be in the following format YYYY-MM-DD.

5.1 Queries

5.1.1 Last expiring certificate

The following query, given the tax code of a person, returns the date of his last expiring certificate and the UCI of such certificate. This query can be used to find out if a person has any valid green pass or not.

```
db.certificates.find(
    {
        'tax_code': "tax code",
        'expiration_date' : {$exists: true}
    },
    {
        'uci': 1,
        'expiration_date': 1
    }
).sort({"expiration_date": -1}).limit(1)
```

5.1.2 Expiration date of certificate

The following query, given a UCI of a certificate, returns its expiration date. This can be useful for instance for the control of certification validity for the access of public spaces.

```
db.certificates.find(
    {
        'uci': "uci"
    },
    {
        "expiration_date": 1,
        "_id": 0
    }
)
```

5.1.3 People vaccinated by a nurse

This query can be used to find all the people (and therefore the related certificates with their information) that get vaccinated by a given sanitary operator in a certain period of time. It can be useful for instance in case, probably due to a mistake, a sanitary operator accidentally injects a saline solution instead of a vaccine, consequentially all the people must be tracked down and warn and their certificates must be revoked.
The tax code is the fiscal code of the nurse who performed the vaccination, min date and max date are the two dates which delimit the period of time.

```
db.certificates.find(
    {
        "vaccination.nurse.tax_code": "tax code",
        "vaccination.date": {$gte: ISODate('min date'),
                             $lte: ISODate('max date')}
    },
    {
        "_id": 0,
        "name": 1,
        "surname": 1,
        "tax code": 1,
        "contact": 1,
        "uci": 1,
        "vaccination.date": 1
    }
)
```

5.1.4 Doctor who performed an anamnesi

This query returns the tax code of the doctor who performed the anamnesi during the last vaccination of a specific person and the related date. It can be used for instance, in case someone die in the next days of a vaccination, to track down the doctor who performed the anamnesi to see if there is any correlation between the death and the anamnesi of the doctor.

The tax code parameter is the fiscal code of the vaccinated person.

```
db.certificates.find(
    {
        "tax_code": "tax code"
    },
    {
        "vaccination.doctor.tax_code": 1,
        "vaccination.date": 1
        _id: 0
     }
).sort({"vaccination.date": -1}).limit(1)
```

5.1.5 Covid history of a person

The following query returns the entire Covid history (all the vaccinations and tests information) of a given person. The tax code is the fiscal code of the person whose history you want to look for.

```
db.certificates.find(
    {
        "tax code": "tax code",
    },
    {
        "test": 1,
        "recovery": 1,
        "vaccination": 1,
        "_id": 0
    }
)
```

5.1.6 Vaccination amount per period

This query returns the amount of vaccination certificates released in a desired period. Min date and max date parameters are the two dates which delimit the period of time.

```
db.certificates.countDocuments(
    {"vaccination.date": {$gte: ISODate('min date'),
                          $lte: ISODate('max date')}
                          }   )
```

5.2 Commands

5.2.1 New vaccination

This query add a new vaccination (and the relevant information) for a person.

```
db.certificates.insertOne({
  'name' : "person name",
  'surname' : "person surname",
  'dob' : "person date of birth",
  'tax_code' : "person tax code",
  'contact' : "person mobile number",
  'emergency_name' : "name of emergency contact",
  'emergency_contact' : "mobile number of emergency contact",
  'uci' : "certificate uci",
  'vaccination' : {
    'revoked' : False,
    'name' : "vaccine name",
    'brand' : "vaccine brand",
    'type' : "vaccine type",
    'lot' : "vaccine lot",
    'sn' : "vaccine sn",
    'dn' : "vaccine dn",
    'issuer' : "Italian Ministry of Health",
    'nurse' : {
      'type' : "Nurse",
      'name' : "nurse name",
      'surname' : "nurse surname",
      'tax_code' : "nurse tax code",
      'contact' : "nurse mobile number"
    },
    'doctor' : {
      'type' : "Doctor",
      'name' : "doctor name",
      'surname' : "doctor surname",
      'tax_code' : "doctor tax code",
      'contact' : "doctor mobile number"
    },
    'place': {
      'building_name' : "building name",
      'type' : "building type",
      'region' : "building region",
      'gps' : "builfing coordinates",
      'city' : "building city",
      'authorized_by' : "building authorized body"
    },
    'date' : "vaccination date",
    'expiration_date': "certificates expiration date"
  }
})
```

5.2.2 Expiration date update

This query changes the expiration date of all the vaccination certificates related to the second dose. It can be used for instance in case of government rules changes to extend the validity of the certificates. Days is the amount of days (integer) that will increase or decrease the validity of the certificate. Sign can only be + (for increasing) or - (for decreasing).

```
db.certificates.updateMany(
    {"vaccination.dn": 2},
    [{ $set: { "vaccination.expiration_date": {
    $add: ["$vaccination.expiration_date", sign 24*3600 * 1000 * days]}}
    }]
)
```

5.2.3 Revoke certificate

The command proposed is used to revoke a certificate under a certain condition. The utility of such command, for instance, could be to revoke all the certificates related to vaccinations performed by a given nurse in a specific day for the same motivation explained in the nurse query in subsection 5.1.3.

```
db.certificates.updateMany(
    {
        "vaccination.date": new Date("YYYY-MM-DD"),
        "vaccination.nurse.tax_code": "nurse tax code"
    },
    {
        "$set": {"revoked": true}
    }
)
```

6 Generator database

The "MongoDB-assignment/main.py" file is responsible of generating a random database. In order to correctly run the generator, it is mandatory to store the MongoDB connection string (with the correct username and password) in a .txt file named connection-string.txt saved in the directory MongoDB_assignment.
This script exploits the following packages: pandas, numpy, PyMongo and python-codicefiscale.

6.1 Place generation

The CSV file "datasets/locations.csv" has been used to generate all the places related to tests and vaccinations. Due to database scale limitation the generation of places has been limited to the Lombardy region.

6.2 Certificate generation

In order to consistently generate certificates information about people have been randomly generated by using our adapted version of the random italian person package. A function has been implemented in order to generate random UCI (by using a real italian format).

6.3 Sanitary operator generation

Sanitary operators information have been randomly generated by using our adapted version of the random italian person package.

6.4 Authorized body generation

The CSV file "datasets/auth_bodies.csv" has been used to generate all the authorized bodies documents. Due to database scale limitation their generation has been limited to only some of them present in the Lombardy region.

7 Application description

Goal of the web application `"webapp/App.py"` is to give some information about the Covid-19 history and certifications about a specific person. The user, after having inserted his tax code to login, can use the application in order to:

- Visualize his/her certifications (valid or expired).
- Visualize his entire Covid history through documents related to his Covis vaccination, test or recovery.

8 User guide

The application is meant to work on every operating system since it is executed on a browser. As the application has been entirely developed in Python, it is not operative system dependent.
The user must check to have correctly installed, in the virtual environment, the following pacakges: `Flask, Flask-PyMongo, bson, PyMongo, pandas, numpy and python-codicefiscale`.

9 Conclusion

Some interesting conclusions can be drawn from the development of this project: Document databases are, if well designed, easy to use and really scalable. Thanks to a good db structure most queries can be easily written in a compact way.

A foundamental property of MongoDB databases is the need to optimize data access, which puts the focus on query patterns. This mean that a good practice for MongoDB databases design is to consider how users will query the data and how often.

10 References and Sources

- Random-italian-person package: https://pypi.org/project/random-italian-person
- PyMongo package: https://docs.mongodb.com/drivers/pymongo/
- Flask package: https://flask.palletsprojects.com/en/2.0.x/
- Italian Government repository: https://github.com/italia/covid19-opendata-vaccini