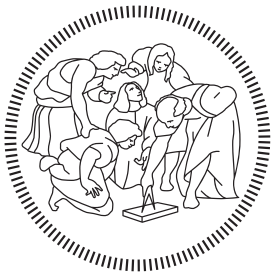
AY 2021/2022



Politecnico di Milano

SMBUD 2021 - Project work delivery 1

ImmunoPoli

Group 20

Margherita	Musumeci	(10600069)
Matteo	Nunziante	(10670132)
Piero	Rendina	(10629696)
Andrea	Sanchini	(10675541)
Enrico	Zuccolotto	(10666354)

Professor
Marco Brambilla

November 14, 2021

Contents

1 Introduction

At the beginning of 2020, the COVID-19 pandemic arose. It entailed many restrictions, lockdowns, and freedom limitations all around the world. Keeping track of all possible contacts among people is a crucial step to help governments and countries address the worldwide viral spreading. Furthermore, collecting and understanding data is the key to have a clear snapshot of all the people currently dealing with the virus, including also vaccines efficacy and contagion rate. The project aims to provide an efficient database for storing all information about people and an application to query it and perform additional analysis. The database is based on a graph data structure and makes usage of the NoSQL paradigm. It stores data coming from various sources, such as data collected either through inspecting public building access or data provided by contact tracing applications in addition to basic personal information about people.

2 Assumptions

The project relies on the hypotheses described below:

1. All information provided by people is truthful.
2. No one visits a place knowing that their last test is positive.
3. The limit for the number of vaccine doses injectable is *two*.
4. There is no house inhabited by underage people only.
5. The people are prevented from changing their residence address.
6. The period ranging from virus incubation to test outcome acknowledgement is *seven days*. It entails that the people who had contact with the positive person, up to seven days before the test, have been potentially exposed to the virus. To be declared safe, each user involved in the exposures needs a negative swab at least seven days after the contact.
7. Since the *covid exposure* relationship represents a possible contact with the virus, for each user, the actual contagion refers to the last positive swab.
8. Every address is associated with just one house or public place.
9. The database includes only entries related to indoor places. Those are assumed to be accessible through a covid certificate vaccination scan. The registration before entering is feasible also by giving own email. Moreover, for those not equipped with the certificate, access is allowed by providing their references.

3 Conceptual model of the database

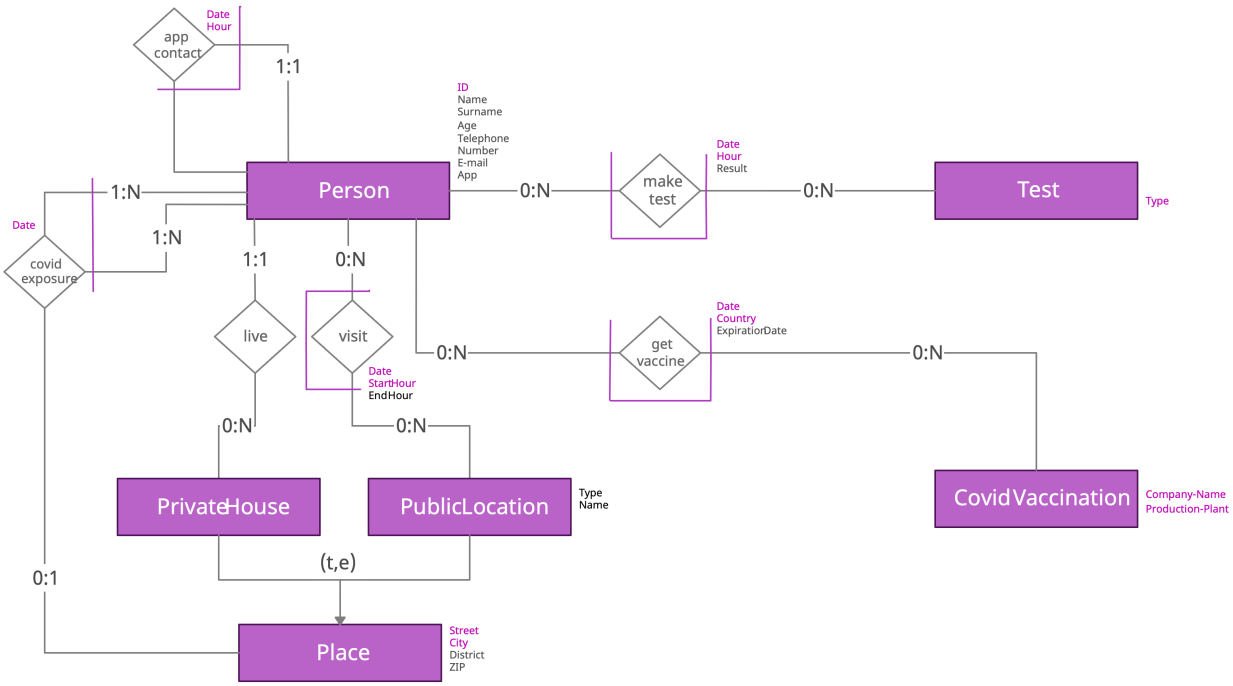


Figure 1: *Entity Relationship model*

3.1 Entities

- **Person:** man or woman identified by a unique ID. The ID is also the key for logging into the app. The property *app* is a boolean attribute used to identify whether or not the person has any type of contact tracing application.
- **Place:** general concept of place characterized by an address.
- **Private house:** place that hosts either a group of flatmates or a family. The name of the house corresponds to the owner's name and surname.
- **Public place:** represents indoor places accessible providing personal information for traceability purposes.
- **Covid Vaccination:** the COVID-19 vaccines. They are identified through their company name and the production plant from where they come.
- **Test:** the COVID-19 tests. They are identified through their class. The distinction is between the tests that detect the viral presence through the *PCR*, *Molecular*, *Antibody* (to check the presence of the virus antagonists) and in the end, the *Rapid* ones.

3.2 Relationships

- The ***covid exposure*** relationship represents a *possible viral infection*. Once a positive test is inserted into the database, the system triggers a command[1] to easily find all the people at risk of contagion and to add an instance of the relationship for all of them. People are obliged to make a swab, and in case of a negative result after at least seven days, the relationship is deleted. Note that if the exposure is detected at home or by contact tracking app, the relationship will have just the date as an attribute.

- The ***live*** relationship binds a person with where he resides.

- The ***visit*** relationship involves all the public places where it is more likely that personal information is collected. It serves both realism and simplicity purposes because we don't need to take care of outdoor spots.

- The ***app contact*** relationship gathers all contacts detected by any tracking system that supports localization and applications installation, regardless of its type.

- The ***make test*** relationship links the user with a test he has made. It contains information such as the hour, the date and the outcome.

- The ***get vaccine*** relationship connects the user to the vaccine he got. As well as *make test*, it is characterized by the date and hour of the injection.

[1] see section 6 for further details about the command.

4 Dataset description

The dataset is drawn from a random generator. It allows enforcing parameters such as the number of visits, tests, covid vaccinations, families and the probability of being positive.
The generator sequentially builds the graph starting from people, places, families and groups of flatmates.
Families are sets of people with at most two different surnames, whereas flatmates are combined randomly.
Afterwards, it adds all the vaccinations ensuring that the first dose of vaccine grants a one-month validity certification. The second vaccination provides one-year validity certification employing the same vaccine as the first one.
Then, it builds instances of *make test* relationships. Every time a positivity status is confirmed, we check the previous contacts and the visited places to look for any *covid exposure* relationships. Moreover, to ensure the assumption in section 2.2, those who have been tested positive are prevented from going out from then on.

5 Queries

5.1 Find undirected contacts stored in the *app contact* relationship.

```
MATCH (p1:Person)-[:APP_CONTACT*]→(p2:Person) RETURN p1 , p2
```

5.2 Find the mean age of the infected[2] people in the last ten days.

```
MATCH (p:Person)-[r:MAKE_TEST]→(t:Test)
WHERE r.result = "Positive" AND r.date ⩾ date() - duration ({Days: 10})
RETURN AVG(toFloat(p.age)) AS average
```

5.3 Find people that live with someone who has been tested positive in the last ten days.

```
MATCH (p:Person)-[:LIVE]→(h:House)←[:LIVE]-(i:Person)-[m:MAKE_TEST]→(t:Test)
WHERE m.result = "Positive" AND m.date ⩾ date() - duration({Days: 10})
RETURN p
```

5.4 Find the houses where resides someone tested positive in the last ten days.

```
MATCH (h:House)←[:LIVE]-(p:Person)-[r:MAKE_TEST {result: "Positive"}]→(t:Test)
WHERE r.date ⩾ date() - duration({days: 10})
RETURN h
```

[2]We refer to infected people as the positive-tested in the last ten days.

5.5 Find the number of test performed in the last month.

```
MATCH (p:Person)-[m:MAKE_TEST]→(t:Test)
WHERE date() ⩽ m.date AND m.date ⩾ date() - duration({days: 30})
RETURN COUNT(m)
```

5.6 Find the number of infected people in a given month.

```
MATCH (p:Person)-[m:MAKE_TEST {result:"Positive"}]→(t:Test)
WHERE m.date.month = 11
RETURN COUNT(m)
```

5.7 Find the number of vaccinations performed in a given month.

```
MATCH (p:Person)-[g:GET_VACCINE]→(v:Vaccine)
RETURN COUNT(g), g.date.month
```

5.8 Find the number of people with a single vaccine dose.

```
MATCH (p:Person)-[g:GET_VACCINE]→(v:Vaccine)
WITH count(g) AS vaccination, p as peopleVaccinatedOnce
WHERE vaccination = 1
RETURN peopleVaccinatedOnce
```

5.9 Find the number of people vaccinated by CAP.

```
MATCH (p:Person)-[g:GET_VACCINE]→(v:Vaccine), (p)-[l:LIVE]→(h:House)
WITH h AS house, p AS person
RETURN COUNT(DISTINCT(person)) AS vaccinated, house.CAP ORDER BY vaccinated DESC
```

5.10 Find the first five places according to the rate of positive people in the last month.

```
MATCH (t:Test)←[m:MAKE_TEST {result:"Positive"}]-(p:Person)-[v:VISIT]→(l:Location)
WHERE  v.date ⩽ m.date ⩽ v.date + duration({Days: 10}) AND v.date ⩾ date() - duration({Days: 30})
WITH COUNT(DISTINCT(p)) AS pos, l

MATCH (p1:Person)-[v1:VISIT]→(l)
WHERE  v1.date ⩾ date() - duration({Days: 30})
WITH COUNT(DISTINCT(p1)) AS vis, pos, l

RETURN pos*1.0/vis AS rate, l.name ORDER BY rate DESC LIMIT 5
```

5.11 Find the people exposed to the virus who have not been tested yet.[3]

```
MATCH (p:Person)-[inf:COVID_EXPOSURE]→(i:Person)
WHERE NOT EXISTS {
    MATCH (p)-[inf:COVID_EXPOSURE]→(i), (p3:Person)-[m:MAKE_TEST]→(t:Test)
    WHERE m.date > inf.date AND id(i) = id(p3)
}
RETURN i
```

5.12 Find the percentage of infected people that got at least one dose of vaccine.

```
MATCH (v:Vaccine)←[g:GET_VACCINE]-(p:Person)-[m:MAKE_TEST{result: "Positive"}]→(t:Test)
MATCH (v)←[g1:GET_VACCINE]-(p1: Person)
WHERE m.date > g.date
RETURN (COUNT(DISTINCT(p)))*100/COUNT(DISTINCT(p1)) AS rate, v.name
```

5.13 Find the number of people positive after being exposed to the virus.

```
MATCH (i:Person)-[inf:COVID_EXPOSURE]→(p:Person)-[m:MAKE_TEST{result: \"Positive\"}]→(Test)
WHERE m.date > inf.date + duration({days: 1})
RETURN p, ID(p)
```

5.14 Find not vaccinated people.

```
MATCH (p:Person)
WHERE NOT((p)-[:GET_VACCINE]→())
RETURN p , ID(p)
```

5.15 Find the number of contact detected by a tracking application for ten people.

```
MATCH (p:Person)-[a:APP_CONTACT]→(:Person)
RETURN COUNT(a), p.name LIMIT 10
```

5.16 Find the worst case of covid exposures tree caused by a positive person.

```
MATCH g1 = (p1:Person)-[relationship1:COVID_EXPOSURE*1..2]→(p2:Person)
WHERE ID(p1) = $id AND
ALL ( idx in range(1 , size(relationships(g1)) - 1)
    WHERE (relationships(g1))[idx - 1].date ⩽ (relationships(g1))[idx].date
    )

MATCH g2 = (p2)-[relationship2:COVID_EXPOSURE*0..2]→(p5:Person)
WHERE
ALL (idx in range(1 , size(relationships(g2)) - 1)
    WHERE (relationships(g2))[idx - 1].date ⩽ (relationships(g2))[idx].date
    )
MATCH ()←[t1:MAKE_TEST{result: "Positive"}]-(p3:Person)-[ce:COVID_EXPOSURE]→(p4:Person)-
[t2:MAKE_TEST{result: "Positive"}]→()
WHERE t1.date ⩽ t2.date AND ce.date ⩽ t1.date AND (ce IN relationships(g1) OR ce IN relationships(g2))
RETURN DISTINCT p3 , ID(p3) , ce , ce.name , ce.date , p4 , ID(p4)
```

[3] They haven't made a swab from the date of the last covid exposure until now.

6 Commands

As already seen in section 4, the generator exploits several basic commands to build the graph database. Concerning the application, it triggers neo4j library routines to manage the database changes and to guarantee its consistency. In this section, we report the behaviour of the most relevant commands and explore their employment within the Python code.
Below, there is the list of commands:

- **Creation of COVID exposures command.** It is triggered whenever the government employee inserts a positive test. At first, it collects all people IDs. Then, it builds the relationships according to the IDs found.

```
"MATCH (pp:Person)-[:LIVE]->(h:House)<-[:LIVE]-(ip:Person) "
"WHERE ID(pp) = $id AND ip <> pp AND NOT (ip)<-[:COVID_EXPOSURE]-(pp)"
"RETURN DISTINCT ID(ip);"
```

Figure 2: *Query that retrieves either flatmates' or relatives' IDs of a positive person.*

```
"MATCH (pp:Person)-[r1:APP_CONTACT]->(ip:Person) "
"WHERE ID(pp) = $id AND (r1.date > date($date) OR (r1.date = date($date) AND r1.hour >= time($hour))) "
"AND (r1.date < date($date) + duration({days:7}) OR (r1.date = date($date)+duration({days:7}) AND "
"r1.hour <= time($hour))) "
"AND NOT "
"(pp)-[:COVID_EXPOSURE{date: r1.date}]->(ip)"
"RETURN DISTINCT ID(ip) , r1.date;"
```

Figure 3: *Query that retrieves app contacts IDs for a positive person.*

```
"MATCH (pp:Person)-[r1:VISIT]->(l:Location)<-[r2:VISIT]-(ip:Person) "
"WHERE ID(pp) = $id AND ip <> pp AND (r1.date > date($date) OR (r1.date = date($date) AND r1.start_hour >= time($hour))) "
"AND (r1.date < date($date) + duration({days:7}) OR (r1.date = date($date)+duration({days:7}) AND "
"r1.end_hour <= time($hour))) AND r2.date = r1.date AND "
"((r1.start_hour < r2.start_hour AND r1.end_hour > r2.start_hour) OR "
"(r2.start_hour < r1.start_hour AND r2.end_hour > r1.start_hour)) AND NOT "
"(pp)-[:COVID_EXPOSURE{name: l.name , date: r1.date}]->(ip)"
"RETURN DISTINCT ID(ip) , r1.date , l.name;"
```

Figure 4: *Query that retrieves people met by a positive person in a certain place.*

```
"MATCH (pp:Person) , (ip:Person) "
"WHERE ID(pp) = $id AND ID(ip) = $ipid "
"CREATE (pp)-[:COVID_EXPOSURE{date: date($date) , name: $name}]->(ip);"
```

Figure 5: *Command run over all previously collected IDs to instantiate the exposures.*

- **Deletion of possible exposures command.** It is triggered whenever the government employee inserts a negative test. It looks for the exposures that may have involved the negative tested person and, it deletes all of them.

```
"MATCH (p:Person) - [inf:COVID_EXPOSURE]-> (i:Person) - [m: MAKE_TEST{result: \"Negative\"}]->(:Test) "
"WHERE m.date >= inf.date + duration({days: 7}) and id(i) = $id "
"DELETE inf "
```

Figure 6: *Deletion command. Note that the search is done by ID.*

- **Update of personal information command**. Differently from the others above, this command is called when a generic user is modifying his information. It supports email and phone number changes.

```
"MATCH (p: Person) "
"WHERE id(p) = $ID "
"SET p.mail = $MAIL, p.number = $NUMBER "
"RETURN p"
```

Figure 7: *Update of telephone number and email provided the person ID.*

- **Creation of a new test command.** It adds the given *date, hour and result* as attributes of the relationship.

```
"MATCH (p:Person) , (t:Test) "
"WHERE ID(p) = $ID AND ID(t) = $testId "
"MERGE (p)-[:MAKE_TEST{date:date($date) , hour: time($hour) ,result:$result}]->(t); "
```

- **Creation of a new visit instance command.** It is activated whenever a location manager adds data related to a *visit*. The *date* is set by default to the current one, whereas the user has to specify the *hour* as attribute of the relationship.

```
"MATCH (p:Person) , (l:Location) "
"WHERE p.mail = $mail AND id(l) = $locationId "
"MERGE (p)-[:VISIT {date: date($date) , start_hour: time($startHour) , end_hour: time($endHour)}]->(l); "
```

7 Application

The application is the database entry point, and it is projected to be run either by a government employee, a location manager or a generic user. The starting page shows the ways to access it.

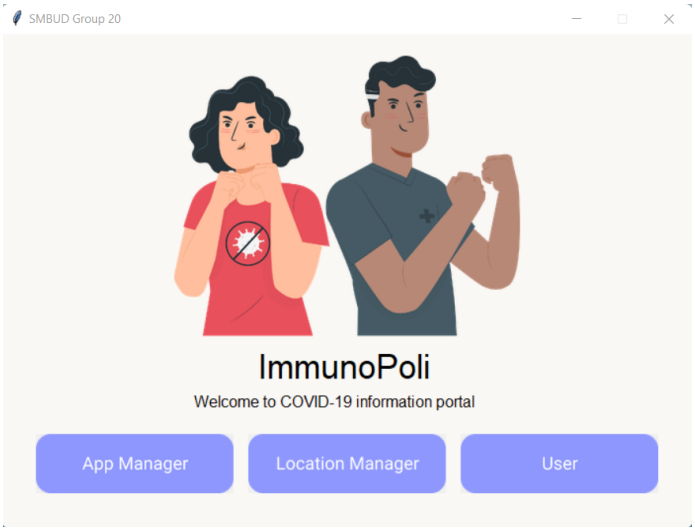


Figure 8: *starting page*

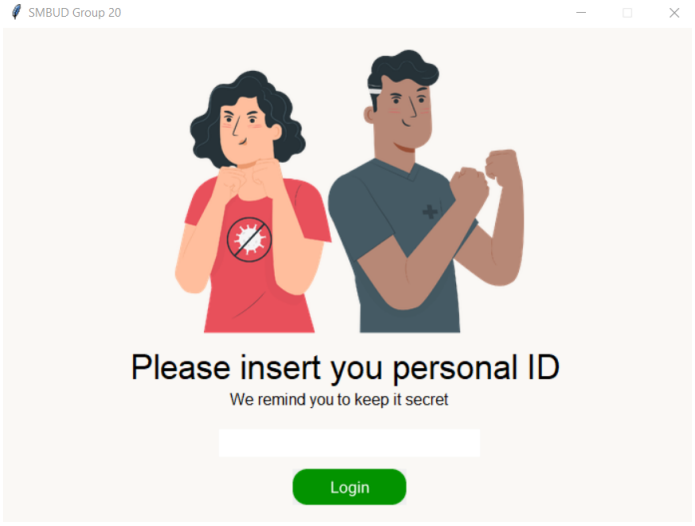


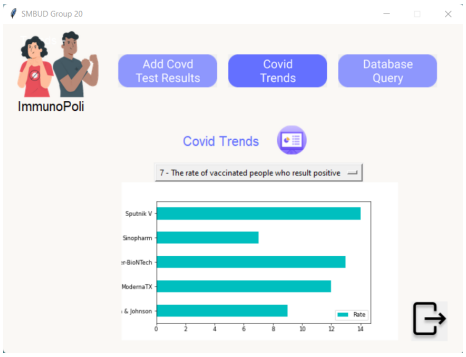
Figure 9: *login page*

The **app manager** option is designed for an employee whose task may consist in:

- adding information concerning COVID tests;
- monitoring COVID trends[4];
- performing useful database queries.

(a) *Number of vaccinated people organized by CAP.*

(b) *Positive people trend over a twelve month horizon.*



(c) *Percentage of positive people after at least one vaccination grouped by vaccine.*

(d) *Number of tests performed each month.*

Figure 10: *some of the analysis that a government employee can perform from his interface.*

[4]The trends that are shown are drawn from a random database. By running them now, it may end up with different results

The **location manager** can register the people entrance inside a place. The access is granted by inserting the location ID.

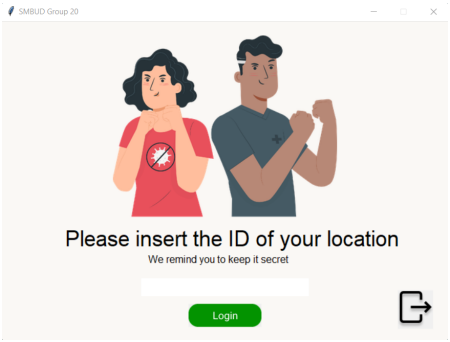


Figure 11: *login instruction for location managers.*

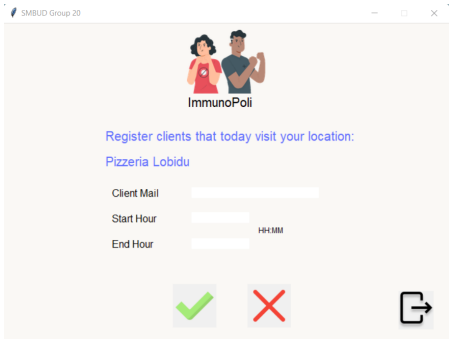
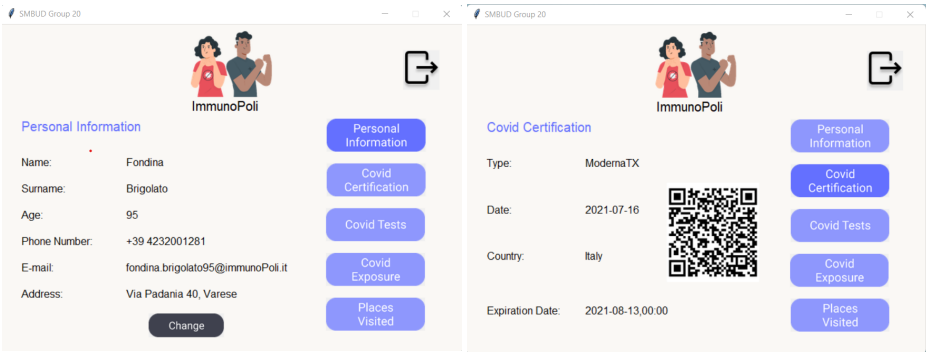


Figure 12: *dashboard for a location manager.*

The **generic user** has access to features that enable:

- changing his/her personal information;
- examining the places that he/she has visited and the risk[5] linked to them;
- checking his/her covid exposures (the date and the place if they are available);
- seeing the history of his/her tests and related outcomes.



(a) *Example of the personal information stored.*

(b) *Example of covid certification lasting four weeks.*

Figure 13: *examples of a dashboard for a generic user.*

[5]The risk factor is the ratio between the number of tested positive people within ten days from the user's visit and the number of visitors in the last month.

8 Resources

To access the application the IDs are needed. Here there is the list of the IDs to sign in.

User/App Manager	Location Manager
4027	4590
4058	4560
4084	4551
4307	4623
3951	4592
4045	4588

To add new tests is required one of the following ID.

Test Name	Test ID
Rapid	4653
Molecular	4654
PCR	4655
Antibody	4656

All the contents covered in this report are available at the link below:
https://github.com/matteoNunz/ImmunoPoli