**Healthcare Passport**

**Jason Scherrer, Connor Pautz, Deniz Kerim**

**Problem Statement**

The idea of this project is to empower people with their own data.  The database will be a record of past doctors visits, current medications, allergens, etc. We are looking to create this with the intention of increasing transparency of records for ones’ self, allowing the owner to be able to pull up and search their medical data all in one local place. The database will contain information about a patient, doctors, hospitals, medication, and doctor reports, and will be able to update doctors and patients information as well as return information about all of the entities. Furthermore, it will be able to create a new patients and doctors, as well as a new report for each visit. A database system is essential for this project because of how much data your hospital visits requires. You visit the hospital so often that being able to store that information into a database is essential for future recollection and knowledge.

**Conceptual Database Design**

A picture containing text

Description automatically generated

**Logical Database Design**

A screenshot of a cell phone

Description automatically generated

Patient:

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Constraint** |
| Full name | CHAR(80) | NOT NULL |
| Social Security Number | INTEGER | Primary Key |
| Phone number | INTEGER |  |
| Date of Birth | INTEGER | NOT NULL |
| DoctorID | INTEGER | Foreign Key |
| Gender | CHAR(80) | NOT NULL |
| Med\_ID | INTEGER | Foreign Key |
| Age | INTEGER |  |
| Vaccine\_Name | CHAR(80) |  |
| Date of Shot | CHAR(80) |  |

Medication:

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Constraint** |
| Report\_ID | INTEGER | Primary key |
| Medicine\_ID | INTEGER | Primary Key |

Report:

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Constraint** |
| Med\_ID | INTEGER | Foreign Key |
| Report\_ID | INTEGER | Primary Key |
| SSN | INTEGER | Foreign key |
| Doc\_ID | INTEGER | Foriegn Key |
| Purpose of Visit | CHAR(80) | NOT NULL |
| Patient Info | CHAR(80) | NOT NULL |
| Height | INTEGER | NOT NULL |
| Weight | INTEGER | NOT NULL |
| Allergies | CHAR(80) |  |

Doctor:

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Constraint** |
| Specification | CHAR(80) | NOT NULL |
| Address | VARCHAR(255) | NOT NULL |
| Name | CHAR(80) | NOT NULL |
| Doctor\_ID | INTEGER | Primary Key |

Hospital

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Constraint** |
| Address | VARCHAR(255) | Primary Key |
| Name | CHAR(80) | NOT NULL |

Department

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Constraint** |
| Department Name | CHAR(80) | Primary Key |
| Address | VARCHAR(255) | Primary Key |

**Application Program Design**

PATIENT:

Add\_patient()

    name = prompt for patient name

    social = prompt for social

    date\_of\_birth = prompt for DOB

    gender = prompt for gender

    phone\_number = prompt for phone\_number

get\_patient()

    patient\_to\_return = prompt for social

    return all attributes for patient\_to\_return

update\_patient()

    patient\_to\_update = prompt for social

    if(social exists)

        // modify various attributes for patient to update.

    else

        prompt to create new patient

delete\_patient()

    patient\_to\_delete = prompt for social

    if(patient\_to\_delete)

        execute delete query starting at topmost users table and cascading downwards.

DOCTOR:

Add\_doctor()

    specification = prompt for specification

    address = prompt for address

    name = prompt for name

    doctor\_id = prompt for doctor\_id  
    hospital = get\_hospital()

    Department = get\_department()

get\_doctor()

    doctor = prompt for doctor\_id

    return all attributes for doctor

update\_doctor()

    doctor = prompt for doctor\_id

    if(doctor exists)

        // modify variosu attributes for doctor to update

    else

        prompt to create new doctor

delete\_doctor()

    doctor = prompt for doctor\_id

    if(doctor exists)

        execute delete query starting at topmost users table and cascading downwards.

MEDICATION:

create\_med()

    medicine = prompt for id

delete\_med()

    medicine = prompt for id

    if(medicine exists)

        execute delete query starting at topmost users table and cascading downwards.

REPORTS:

create\_report()

    report\_id = prompt for report id

    purpose\_of\_visit = prompt for purporse of visit

    patient\_id = prompt for patient id

    if(patient\_id exists)

        patient\_info = all attributes for the patient

    else

        prompt to create a patient

get\_report()

    report\_id = prompt for report id

    return all attributes for the report

delete\_report()

    report = prompt for id

    if(report exists)

        execute delete query starting at topmost users table and cascading downwards.

HOSPITAL:

create\_hospital()

    address = prompt for address

    name = prompt for name

get\_hospital()

    address = prompt for address

    return all attributes for hospital with address

delete\_hospital()

    hospital = prompt for hospital

    if(hospital exists)

        execute delete query starting at topmost users table and cascading downwards.

DEPARTMENT:

create\_department()

    department\_name = prompt for department name

    address = prompt for address

get\_department()

    name = prompt for department name

    return all address with department name

delete\_department()

    department = prompt for name

    if(department exists)

        execute delete query starting at topmost users table and cascading downwards.

**Installation Instructions**

Intended Operating System: tested on MacOS and Windows 10

Installing system: use a python package manager to install the following:

* Flask
* Flasksql
* Flask\_wtf
* Sql\_alchemy

From here, from a command line you can run **flask run** inside of the healthcare passport folder in order to run the website and the database on your local machine.

**User Manual (Connor)**