**Healthcare System:**

1. Frameworks to be used – Flask, Python
2. FHIR Library

**Project Outline:**

1. Healthcare API will provide patient details based on the requested query parameter like patient ID.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S. No** | | **API** | **Fields** | **Return Fields** |
|  | http://127.0.0.1:5000/patientinfo? | | patient\_id=<id> | 'PatientUID', 'NameFamily', 'NameGiven', 'DoB', 'Gender' |
|  | http://127.0.0.1:5000/patientconditions? | | patient\_id=<id> | 'ConditionText', 'ConditionOnsetDates' |
|  | http://127.0.0.1:5000/patientobservations? | | patient\_id=<id> | 'ObservationText', 'ObservationValue', 'ObservationUnit', 'ObservationDate' |
|  | http://127.0.0.1:5000/patientprocedure? | | patient\_id=<id> | 'ProcedureText', 'ProcedureDates', 'PatientUID' |
|  | http://127.0.0.1:5000/patientencounter? | | patient\_id=<id> | 'EncountersText', 'EncounterLocation', 'EncounterProvider', 'EncounterDates', |
|  | http://127.0.0.1:5000/patientmedication? | | patient\_id=<id> | 'MedicationText', 'MedicationDates' |
|  | http://127.0.0.1:5000/patientclaim? | | patient\_id=<id> | 'Immunization', 'ImmunizationDates' |

Other possible improvements:

Storing Data:

1. The json files could be loaded into Databases like MySql for faster access by providing partitioning ( Key Partitioning, Hash Partitioning, Range Partitioning)
2. Different segments of information of patients can be stored under separate tables and further it can be accessed through joints like INNER JOINTS, OUTER JOINTS etc. for increasing the performance of API .

Encrypting Data:

1. Since data can be prone to vulnerability it could be encrypted with Base64 or other advanced cryptographic techniques before sending over the network.

User based access:

1. Since there could be different groups of user who uses patients records, these data could be restructured into further. For example, a hospital may access a bunch of patient records who got treatment from their hospital. So the data may be restructured by grouping data based on the hospital name or ID.
2. Similarly a research organization might want to see the impact of their immune boosting vaccine, so the data could be again restructured to suit the need of end user.

**Constructing library for reusable codes:**

1. A library can be built for function like filtering and restructuring patient records based on the end user level like ( for researchers, hospitals, governments etc)
2. Certain process flow could be implemented using threads and process to achieve multiprocessing and to improve performance of application.

**Simple application for public users:**

1. A login/signup page for public users where they can register themselves using their ID proof.
2. A login/signup page for medical practitioners who can access their patient’s historical records.
3. Public users can login and view their medical history like appointments attended at different hospitals, medications received, medical documents.
4. Medical practitioners will be able to update patient records only for the treatment received from them and will be able to go through the medical history of patients.
5. This application could be used as a centralized system for maintaining the medical records of patients, thereby making it more reliable to track the health of patients.

**Statistics data automation for tracking health of common public :**

1. Cron jobs can be build to automate statistical data.
2. Information regarding major diseases outbreak in regular basis.
3. Information regarding immune vaccines given.
4. Information regarding areas getting infected very frequent