OER

```
Improving Education Through Personalised Learning.
```

GOAL

```
Give High quality Search Results thourgh Artifial Intelligence.
```

OBJECTIVES

```
Acquire Data
Clean Data
Store Data
Data Syncing
Index Data
High quality search results based on user query
```

PREREQUISITE

- `Ubuntu 18.04 LTS` `PostgreSQL 14` `MongoDB >4.0` `Redis Server` `ElasticSearch =>7.0` `pgsync` `Storage => 500GB` `RAM => 32GB` `CORES => 16`

POSTGRESQL

Storage , Read , Write and Index.

1.1 Setup

```
Install postgre >= 9.4 as elastic search  is mandatory.
Older postgre versions wont work well with syncqing data to elasticsearch >=7.1.
```

1.2 Dependencies

```
sudo apt-cache search postgresql | grep postgresql
sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-pgdg main" > /etc/apt/sources.list.d/pgdg.list'
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
sudo apt -y update
```

1.3 Installation

```
sudo apt -y install postgresql-14
```

1.4 Conditions

```
#[ENABLE] postgresql service
   sudo systemctl enable postgresql
#[RESTART] postgresql service
   sudo systemctl restart postgresql
#[STOP] postgresql service
   sudo systemctl stop postgresql
#[STATUS] postgresql service
   sudo systemctl status postgresql
```

1.5 Version

```
postgresql --version
```

1.7 Config

```
sudo nano /etc/postgresql/14/main/postgresql.conf
#WRITE ACCESS LOGS LEVEL
   [BEFORE] wal_level = ??
   [AFTER]  wal_level = logical
#REPLICATION SLOTS
   [BEFORE] # max_replication_slots = ??
   [AFTER]  max_replication_slots = 1
#WRITE ACCESS LOGS SIZE
   [BEFORE] max_slot_wal_keep_size = ??
   [AFTER]  max_slot_wal_keep_size = 100GB
#RESTART
   sudo systemctl restart postgresql
#STATUS
   sudo systemctl status postgresql
```

1.8 Ports

```
postgre --5432
sudo ufw allow 5432/tcp
sudo ufw allow 5432/udp
```

REDIS

Queueing , Publishing , Subscribing and Multithreading.

2.1 Setup

```
Install redis-server version >= 6.0 as multithreading is mandatory.
Older redis-server versions are single threaded and this introduces blocking which is
not good for a distributed production system.
```

2.2 Dependencies

```
sudo apt install default-jre
sudo apt install pkg-config
sudo add-apt-repository ppa:chris-lea/redis-server
sudo apt-get update
```

2.3 Install

```
sudo apt-get install redis-server -y
```

2.4 Coditions

```
#[ENABLE] redis-server service
   sudo systemctl enable redis-server.service
#[RESTART] redis-server service
   sudo systemctl restart redis-server.service
#[STOP] redis-server service
   sudo systemctl stop redis-server.service
#[STATUS] redis-server service
   sudo systemctl status redis-server.service
```

2.5 Version

```
redis-server --version
```

2.7 Config

```
sudo vi /etc/redis/redis.conf
#REMOTE CONNECTION
   [BEFORE] bind 127.0.0.1
   [AFTER]  bind 0.0.0.0
#MULTHTHREADING
   [BEFORE] # io-threads 4
   [AFTER]  io-threads [TOTAL THREADS - 2]
#READS
   [BEFORE] io-threads-do-reads no
   [AFTER]  io-threads-do-reads yes
#RESTART
   sudo systemctl restart redis-server.service
#STATUS
   sudo systemctl status redis-server.service
```

2.8 Ports

```
redis --6379
sudo ufw allow 6379/tcp
sudo ufw allow 6379/udp
```

MONGODB

Storage, unstructured data, profiles and authentication.

3.1 Setup

```
Install mongodb version >= 5.0.
```

3.2 Dependencies

```
sudo apt install default-jre
sudo apt install pkg-config
wget -qO - https://www.mongodb.org/static/pgp/server-5.0.asc | sudo apt-key add -
sudo nano /etc/apt/sources.list.d/mongodb.list
deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/5.0
multiverse
sudo apt update
```

3.3 Install

```
sudo apt install mongodb-org
```

3.4 Version

```
mongo
db.version()
```

3.5 Conditions

```
#[ENABLE] mongo service
   sudo systemctl enable mongod
#[RESTART] mongo  service
   sudo systemctl restart mongod
#[STOP] mongo service
   sudo systemctl stop mongod
#[STATUS] mongo service
   sudo systemctl status mongod
```

3.7 Config

```
sudo lsof -i | grep mongo
sudo vi /etc/mongodb.conf
#REMOTE CONNECTION
   [BEFORE] bind 127.0.0.1
   [AFTER]  bind 127.0.0.1,[MONGO_SERVER_IP]
#RESTART
```

```
   sudo systemctl restart mongodb
#STATUS
   sudo systemctl status mongodb
```

3.8 Ports

```
mongodb --27017
sudo ufw allow 27017/tcp
sudo ufw allow 27017/udp
```

ELASTICSEARCH

Indexing

4.1 Setup

```
Install elasticsearch version >= 7.0 as fror pgsync intergration with postgres 14.
Older elasticsearch versions are single threaded and this introduces blocking which is
not good for a distributed production system.
```

4.2 Dependencies

```
curl -fsSL https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee -a
/etc/apt/sources.list.d/elastic-7.x.list
sudo apt update
```

4.3 Install

```
sudo apt install elasticsearch
```

4.4 Coditions

```
#[ENABLE] elasticsearch service
   sudo systemctl enable elasticsearch
#[RESTART] elasticsearch service
   sudo systemctl restart elasticsearch
#[STOP] elasticsearch service
   sudo systemctl stop elasticsearch
#[STATUS] elasticsearch service
   sudo systemctl status elasticsearch
```

4.5 Version

```
elasticsearch --version
```

4.7 Config

```
sudo nano sudo nano /etc/elasticsearch/elasticsearch.yml
#NETWORK CONFIGURATION
    network.host: 0.0.0.0
    http.port: 9200
    transport.host: localhost
    transport.tcp.port: 9300

#RESTART
   sudo systemctl restart elasticsearch
#STATUS
   sudo systemctl status elasticsearch
```

4.8 Ports

```
elasticsearch --9200
sudo ufw allow 9200/tcp
sudo ufw allow 9200/udp
```

4.9 Testing

```
curl -X GET 'http://localhost:9200'
```

PGSYNC

Realtime Syncing Pipeline

5.1 Setup

```
    pip3 install pgsync
```

5.2 Config

```
    sudo nano /PATH_TO/.env
    sudo nano /PATH_TO/schema.json
```

5.3 Index Db

```
    bootstrap --config /PATH_TO/schema.json
    pgsync
```

5.4 List indices

```
#[ELASTICSEARCH]
     curl -X GET "localhost:9200/_cat/indices?pretty"
```

5.5 Get Specific Index

```
#[ELASTICSEARCH]
    curl -XGET "https://localhost:9200/_idxbedb" -d'
    curl -XGET "https://localhost:9200/<index name>" -d'
```

5.7 Delete Specific Index

```
    curl -XDELETE localhost:9200/idxtedb
    curl -XDELETE localhost:9200/<index name>
```

5.8 Delete Specific Index

```
#[POSTGRES]
    rm.postgre_idxtedb
    rm .<database name>_<index name>
```

6.0 DATABASE

Clean , Structure and Store Data.

6.1 TEACHERS FUTURES

```
[LOGIN TO POSTGRE DB]
psql u -postgres
CREATE TABLE TEACHERS_FUTURES_DB(
    id SERIAL PRIMARY KEY,
    Course VARCHAR  NULL,
    Author VARCHAR  NULL,
    Activity VARCHAR NULL,
    Week VARCHAR  NULL,
    Title VARCHAR  NULL,
    Description VARCHAR  NULL,
    Keywords VARCHAR  NULL,
    Document VARCHAR  NULL,
    Date_Created date NOT NULL default CURRENT_DATE
);

   #alter table teachers_futures_db add column Date_Created date not null default
CURRENT_DATE;

   postgres=# copy teachers_futures_db
(Course,Author,Activity,Week,Title,Description,Keywords,Document) from 'PATH_TO/ACTT
Breakdown - Week 1.csv' DELIMITER ',' CSV HEADER;
   COPY 63
```

```
   postgres=# copy teachers_futures_db
(Course,Author,Activity,Week,Title,Description,Keywords,Document) from 'PATH_TO/ACTT
Breakdown - Week 2.csv' DELIMITER ',' CSV HEADER;
   COPY 22
   postgres=# copy teachers_futures_db
(Course,Author,Activity,Week,Title,Description,Keywords,Document) from 'PATH_TO/ACTT
Breakdown - Week 3.csv' DELIMITER ',' CSV HEADER;
   COPY 22
   postgres=# copy teachers_futures_db
(Course,Author,Activity,Week,Title,Description,Keywords,Document) from 'PATH_TO/ACTT
Breakdown - Week 4.csv' DELIMITER ',' CSV HEADER;

CREATE TABLE TEACHERS_FUTURES_VIDEO_DB(
   id SERIAL PRIMARY KEY,
   Course VARCHAR  NULL,
   Author VARCHAR  NULL,
   Activity VARCHAR NULL,
   Week VARCHAR  NULL,
   Title VARCHAR  NULL,
   Description VARCHAR  NULL,
   Keywords VARCHAR  NULL,
   Document VARCHAR  NULL,
   Date_Created date NULL default CURRENT_DATE
);
copy teachers_futures_video_db
(Course,Author,Activity,Week,Title,Description,Keywords,Document,Date_Created) from
'PATH_TO/ACTT Text Breakdown - ACTT Video.csv' DELIMITER ',' CSV HEADER;

CREATE TABLE TEACHERS_FUTURES_AUDIO_DB  (
   id SERIAL PRIMARY KEY,
   Course VARCHAR  NULL,
   Author VARCHAR  NULL,
   Activity VARCHAR NULL,
   Week VARCHAR  NULL,
   Title VARCHAR  NULL,
   Description VARCHAR  NULL,
   Keywords VARCHAR  NULL,
   Document VARCHAR  NULL,
   Date_Created date NULL default CURRENT_DATE
);
copy teachers_futures_audio_db
(Course,Author,Activity,Week,Title,Description,Keywords,Document,Date_Created) from
'PATH_TO/ACTT Text Breakdown - ACTT Audio-2.csv' DELIMITER ',' CSV HEADER;

CREATE TABLE TEACHERS_FUTURES_DB(
   id SERIAL PRIMARY KEY,
   Course VARCHAR  NULL,
   Author VARCHAR  NULL,
   Activity VARCHAR NULL,
   Week VARCHAR  NULL,
   Title VARCHAR  NULL,
   Description VARCHAR  NULL,
```

```
   Keywords VARCHAR  NULL,
   Document VARCHAR  NULL,
   Date_Created date NULL default CURRENT_DATE
   );
copy teachers_futures_db
(Course,Author,Activity,Week,Title,Description,Keywords,Document,Date_Created) from
'PATH_TO/ACTT Text Breakdown - ACTT.csv' DELIMITER ',' CSV HEADER;
```

6.2 OPENLIBRARY

```
    [LOGIN TO POSTGRE DB]
    psql u -postgres

   psql postgres < /PATH_TO/openlibrary-db.sql
   ALTER TABLE editions DROP COLUMN work_key;

   COPY works FROM '/PATH_TO/[works_or_authors_editions_dataset].csv' DELIMITER E'\t'
QUOTE '|' CSV;
   COPY authors FROM '/PATH_TO/[works_or_authors_editions_dataset].csv' DELIMITER
E'\t' QUOTE '|' CSV;
   COPY editions FROM '/PATH_TO/[works_or_authors_editions_dataset].csv' DELIMITER
E'\t' QUOTE '|' CSV;

   ALTER TABLE editions ADD COLUMN work_key;

   insert into editionisbn13s select distinct key,  jsonb_array_elements(data-
>'isbn_13')->>0  from editions where  key is not null and data->'isbn_13'->0 is not
null;

   select e.data->>'title' "EditionTitle", e.data->'languages'->0->>'key'
"Language",e.data->>'publish_date' "DateUpdated", e.data->>'subtitle'
"EditionSubtitle" , e.data->>'subjects'
   "Subjects",a.data->'name' "author" from editions e join editionisbn13s ei on
ei.edition_key = e.key join works w on w.key = e.work_key join authors a on a.key =
w.data->'authors'->0->'author'->>'key'
   where e.data->'languages'->0->>'key'='/languages/eng' OR e.data->'languages'->0-
>>'key' = '/l/eng' limit 3;

   copy (select e.data->>'title' "EditionTitle", e.data->'languages'->0->>'key'
"Language",e.data->>'publish_date' "DateUpdated", e.data->>'subtitle'
"EditionSubtitle" , e.data->>'subjects'
   "Subjects",a.data->'name' "author" from editions e join editionisbn13s ei on
ei.edition_key = e.key join works w on w.key = e.work_key join authors a on a.key =
w.data->'authors'->0->'author'->>'key'
   where e.data->'languages'->0->>'key'='/languages/eng' OR e.data->'languages'->0-
>>'key' = '/l/eng' limit 3) to '/PATH_TO/open_library_export.csv' With CSV DELIMITER
E'\t';

   CREATE TABLE BOOKS_ENG_DB(
         id SERIAL PRIMARY KEY,
         title VARCHAR  NULL,
```

```
        language VARCHAR  NULL,
        date_updated VARCHAR NULL,
        abstract VARCHAR  NULL,
        subject VARCHAR  NULL,
        authors VARCHAR  NULL
   );

INSERT INTO BOOKS_ENG_DB(title, language, date_updated, abstract, subject, authors,
notes, description, book_key,covers)select e.data->>'title' , e.data->'languages'->0-
>>'key' ,e.data->>'publish_date',
e.data->>'subtitle', e.data->>'subjects',a.data->'name',e.data->'notes'-
>>'value',e.data->'description'->>'value',w.key,w.data->'covers'->>0 from editions e
join editionisbn10s ei on ei.edition_key = e.key
join works w on w.key = e.work_key join authors a on a.key
= w.data->'authors'->0->'author'->>'key'  where e.data->'languages'->0-
>>'key'='/languages/eng' OR e.data->'languages'->0->>'key' = '/l/eng';
```

6.3 X5GON

```
[LOGIN TO MONGO VALIDATA IF DATA EXISTS FROM DB TEST AND COLLECTIONS X5GON]
mongo 192.168.8.212:27017 -u "AlbusDumbledore5" -p "SherbetPops22" --
authenticationDatabase 'admin'
    show databases
    test.x5gon.count()
[LOGIN TO MONGO VALIDATA IF DATA EXISTS]
mongoexport --host 192.168.8.212:27017 -u "AlbusDumbledore5" -d test -c x5gon --
forceTableScan  -p "SherbetPops22"  --authenticationDatabase 'admin' --type=csv --out
x5gon.csv --fields 'record_data.title,
record_data.description,record_data.type,record_data.url,record_data.website,record_data

record_data.provider.domain,record_data.content_ids.0,record_data.license.short_name
,record_data.license.disclaimer ,record_data.license.url'

[LOGIN TO POSTGRE DB]
psql u -postgres

    CREATE TABLE X5GON(
        id SERIAL PRIMARY KEY,
        title VARCHAR  NULL,
        description VARCHAR  NULL,
        data_type VARCHAR NULL,
        url VARCHAR  NULL,
        website VARCHAR  NULL,
        language VARCHAR  NULL,
        creation_date VARCHAR  NULL,
        retrieved_date VARCHAR  NULL,
        provider_name VARCHAR  NULL,
        provider_id VARCHAR  NULL,
        provider_domain VARCHAR  NULL,
        content_ids VARCHAR  NULL,
        license_name VARCHAR  NULL,
```

```
        license_disclaimer VARCHAR  NULL,
        license_url VARCHAR  NULL
    );
    COPY
x5gon(title,description,data_type,url,website,language,creation_date,retrieved_date,prov

    FROM 'PATH_TO/x5gon.csv' (FORMAT csv, HEADER, DELIMITER ',');

    CREATE TABLE X5GON_EN_DB(
        id SERIAL PRIMARY KEY,
        title VARCHAR  NULL,
        language VARCHAR  NULL,
        description VARCHAR NULL,
        item VARCHAR  NULL,
        website VARCHAR  NULL,
        authors VARCHAR  NULL,
        creation_date DATE  NULL,
        retrieved_date DATE  NULL,
        provider_name VARCHAR  NULL,
        provider_id VARCHAR  NULL,
        provider_domain VARCHAR  NULL,
        content_ids VARCHAR  NULL,
        license_name VARCHAR  NULL,
        license_disclaimer VARCHAR  NULL,
        license_url VARCHAR  NULL,
        data_type VARCHAR  NULL
    );
    INSERT INTO  X5GON_EN_DB(title, language,description,item ,
website,authors,creation_date,provider_id,provider_domain,content_ids,license_name,licer
 data_type)select DISTINCT

title,language,description,url,website,provider_name,creation_date::timestamp::date,prov
 data_type from x5gon where language='en';

    CREATE TABLE AUDIO_EN_DB(
        id SERIAL PRIMARY KEY,
        title VARCHAR  NULL,
        language VARCHAR  NULL,
        description VARCHAR NULL,
        item VARCHAR  NULL,
        website VARCHAR  NULL,
        authors VARCHAR  NULL,
        creation_date DATE  NULL,
        retrieved_date DATE  NULL,
        provider_name VARCHAR  NULL,
        provider_id VARCHAR  NULL,
        provider_domain VARCHAR  NULL,
        content_ids VARCHAR  NULL,
        license_name VARCHAR  NULL,
        license_disclaimer VARCHAR  NULL,
        license_url VARCHAR  NULL,
        data_type VARCHAR  NULL
```

```
    );
    INSERT INTO  AUDIO_EN_DB(title, language,description,item ,
website,authors,creation_date,provider_id,provider_domain,content_ids,license_name,licer


title,language,description,item,website,provider_name,creation_date::timestamp::date,pro
 from x5gon_en_db where data_type='audio';

    CREATE TABLE VIDEOS_EN_DB(
        id SERIAL PRIMARY KEY,
        title VARCHAR  NULL,
        language VARCHAR  NULL,
        description VARCHAR NULL,
        item VARCHAR  NULL,
        website VARCHAR  NULL,
        authors VARCHAR  NULL,
        creation_date DATE  NULL,
        retrieved_date DATE  NULL,
        provider_name VARCHAR  NULL,
        provider_id VARCHAR  NULL,
        provider_domain VARCHAR  NULL,
        content_ids VARCHAR  NULL,
        license_name VARCHAR  NULL,
        license_disclaimer VARCHAR  NULL,
        license_url VARCHAR  NULL,
        data_type VARCHAR  NULL
    );
    INSERT INTO  VIDEOS_EN_DB(title, language,description,item ,
website,authors,creation_date,provider_id,provider_domain,content_ids,license_name,licer
  title,

language,description,item,website,provider_name,creation_date::timestamp::date,provider_
 from x5gon_en_db where data_type='video';

    CREATE TABLE TEXT_EN_DB(
        id SERIAL PRIMARY KEY,
        title VARCHAR  NULL,
        language VARCHAR  NULL,
        description VARCHAR NULL,
        item VARCHAR  NULL,
        website VARCHAR  NULL,
        authors VARCHAR  NULL,
        creation_date DATE  NULL,
        retrieved_date DATE  NULL,
        provider_name VARCHAR  NULL,
        provider_id VARCHAR  NULL,
        provider_domain VARCHAR  NULL,
        content_ids VARCHAR  NULL,
        license_name VARCHAR  NULL,
        license_disclaimer VARCHAR  NULL,
        license_url VARCHAR  NULL,
        data_type VARCHAR  NULL
```

```
    );
    INSERT INTO  TEXT_EN_DB(title, language,description,item ,
website,authors,creation_date,provider_id,provider_domain,content_ids,license_name,licer


title,language,description,item,website,provider_name,creation_date::timestamp::date,pro

    from x5gon_en_db where data_type='text';
```

6.4 SEMANTIC SCHOLAR

```
    psql  -U postgres -h 127.0.0.1 --password scholar;
    CREATE TABLE PAPERS_EN_DB(
        id SERIAL PRIMARY KEY,
        title VARCHAR  NULL,
        paperAbstract VARCHAR  NULL,
        authors VARCHAR NULL,
        year VARCHAR  NULL,
        s2Url VARCHAR  NULL,
        pdfUrls VARCHAR  NULL,
        journalName VARCHAR  NULL,
        doiUrl VARCHAR  NULL,
        fieldsOfStudy VARCHAR  NULL
    );
    jq -c . *.json | spyql -Otable=papers_en_db "SELECT json->title, json-
>paperAbstract AS paperabstract, json->authors, json->year, json->s2Url AS
    s2url,json->pdfUrls AS pdfurls, json->journalName AS journalname, json->doiUrl AS
doiurl,json->fieldsOfStudy AS fieldsofstudy FROM json TO sql" | psql  -U postgres -h
127.0.0.1 --password scholar

    CREATE TABLE ADVANCED_PAPERS_EN_DB(
        id SERIAL PRIMARY KEY,
        title VARCHAR  NULL,
        paperAbstract VARCHAR  NULL,
        authors VARCHAR NULL,
        year VARCHAR  NULL,
        s2Url VARCHAR  NULL,
        pdfUrls VARCHAR  NULL,
        journalName VARCHAR  NULL,
        doiUrl VARCHAR  NULL,
        fieldsOfStudy VARCHAR  NULL
    );
    INSERT INTO ADVANCED_PAPERS_EN_DB (title, paperAbstract ,authors, year , s2Url
,pdfUrls,journalName,doiUrl ,fieldsOfStudy )select  title, paperAbstract ,authors,
year , s2Url ,
    pdfUrls,journalName,doiUrl ,fieldsOfStudy  from PAPERS_EN_DB
    where length(pdfUrls) >2;
```

6.5 STATS

```
    X5GON
    postgres=# SELECT  provider_domain , count(*) FROM X5GON_EN_DB WHERE
provider_domain is not null GROUP BY  provider_domain;
                provider_domain              | count
    -----------------------------------------+-------
    http://campus.unibo.it                   |  3589
    http://madoc.univ-nantes.fr/             |    29
    https://av.tib.eu/                       |    33
    https://cnx.org                          |  7546
    https://media.upv.es                     |   164
    https://medienportal.siemens-stiftung.org |   652
    https://ocw.mit.edu/                     | 44963
    https://openlibrary.ecampusontario.ca/   |   197
    https://www.canal-u.tv                   |     5
    https://www.engageny.org/                |  4595
    https://www.oerafrica.org/               |   317
    https://www.openlearnware.de/            |   135
    http://videolectures.net/                | 23840
    (13 rows)

    \COPY (SELECT  provider_domain , count(*) FROM X5GON_EN_DB WHERE  provider_domain
is not null GROUP BY  provider_domain) TO 'PATH_TO/X5GON_EN_DB.csv' With CSV DELIMITER
',' HEADER;
    #SEMANTIC SCHOLAR
    \COPY (SELECT   journalname , count(*) FROM advanced_papers_en_db WHERE
journalname is not null GROUP BY  journalname) TO 'PATH_TO/ADVANCED_PAPERS_EN_DB.csv'
With CSV DELIMITER ',' HEADER;
```

7.0 DATABASE

Queing ,Syncing and Indexing.

6.1 Synqing