Music/Myapp Application

1. Download tools

1. Docker Desktop
2. pgAdmin
3. git
4. vs code
5. dbeaver-ce
6. Angular

2. Download docker images

### postgres:latest

1. litmuschaos/mongo

You can do it by the following commands

docker pull postgres

docker pull mongodb/mongodb-community-server

3. DOWNLOAD JAVA

version 24

Create database in pgAdmin

mydatabase

Install angular 19 or 20

1. Instal NodeJS

<https://nodejs.org/en/download>

or

1. sudo apt install npm

# Download and install nvm:

curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.40.2/install.sh | bash

# in lieu of restarting the shell

\. "$HOME/.nvm/nvm.sh"

# Download and install Node.js:

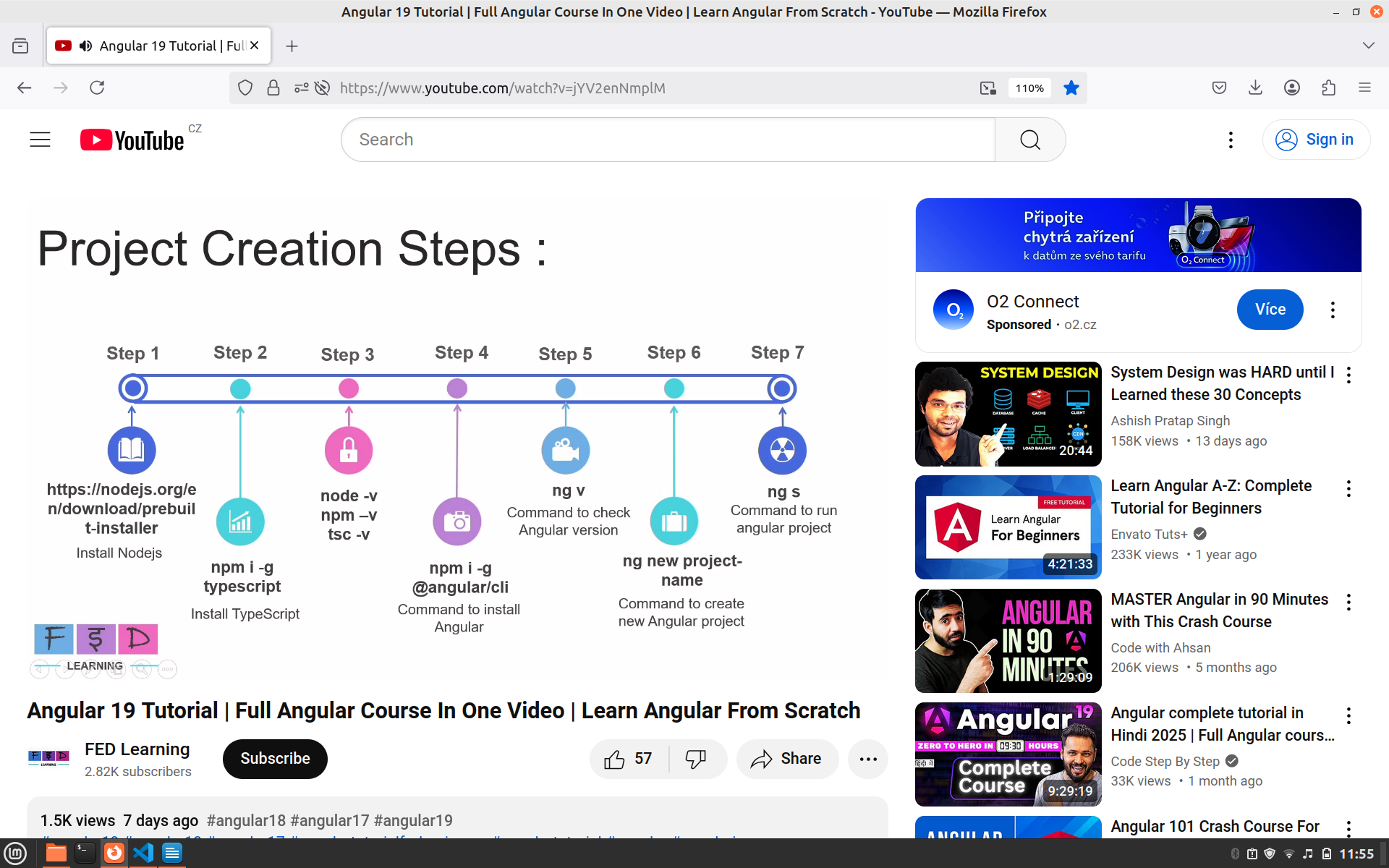
nvm install 22

# Verify the Node.js version:

node -v # Should print "v23.11.0".

nvm current # Should print "v23.11.0".

# Verify npm version:

npm -v # Should print "10.9.2".

npm I -g typescript

1. install typescript extension to visual studio code

First run docker dependencies

donwload docker images

**Postgres container () I am not sure you need the following command**

docker run --hostname=467ca158f369 --mac-address=1a:22:4b:7e:6b:f6 --env=POSTGRES\_PASSWORD=password --env=PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/lib/postgresql/17/bin --env=GOSU\_VERSION=1.17 --env=LANG=en\_US.utf8 --env=PG\_MAJOR=17 --env=PG\_VERSION=17.5-1.pgdg120+1 --env=PGDATA=/var/lib/postgresql/data --volume=/var/lib/postgresql/data --network=bridge -p 5432:5432 --restart=no --runtime=runc -d postgres:latest

################## RUN ##########################

**Download music and myapp projects**

**git clone** [**https://github.com/ahnsys1/MYAPP\_AND\_MUSIC.git**](https://github.com/ahnsys1/MYAPP_AND_MUSIC.git)

**MyApp**

**mongo-myapp-client-release**

**myapp-backend-development**

**myapp-jpa-client-release**

**now let’s start backend first**

**myapp-backend-development/mvn clean install**

**myapp-backend-development/java -jar target/backend-0.0.1-SNAPSHOT.jar**

**mongo-myapp-client-release/npm i**

**mongo-myapp-client-release/ng s**

**OR: depending if you want to store data in MongoDB or in PostrgeSQL**

**myapp-jpa-client-release/npm i**

**myapp-jpa-client-release/ng s**

**npm i uuid --dave**

**npm i jquery --save**

npm i datatables.net --save

npm i datatables.net-dt --save

npm i angular-datatables --save

npm i @types/jquery --save-dev

**npm i bootstrap5 --save**

**ng add @ng-bootstrap/ng-bootstrap**

**npm list -g**

**/home/vj/.nvm/versions/node/v23.11.0/lib**

**├── @angular/cli@19.2.6**

**├── corepack@0.32.0**

**├── npm@10.9.2**

**├── typescript@5.8.3**

**└──** [**uuid@11.1.0**](mailto:uuid@11.1.0)

**Manual for integrating datatables into Angular project**

[**https://nipuanandawansha.medium.com/angular-datatables-with-angular-19-4ee54d989639**](https://nipuanandawansha.medium.com/angular-datatables-with-angular-19-4ee54d989639)

**npm install json-server**

**npm i @types/datatables.net --save-dev**

**npm i json-server**

**Angular\_19/src/REST-endpoints$**

**json-server -w users-data.json**

**angular-datatables-master**

[**https://github.com/l-lin/angular-datatables/tree/master**](https://github.com/l-lin/angular-datatables/tree/master)

**npm start**

**this starts development server**

**localhost:4200**

**npm i –save @types/jquery**

**ng add @angular/material**

**PROJECT EXPORT**

**git archive --format=zip --output zipfile.zip development**

**json-server -w employee-data.json**

json-server -p 4000 -w users-data.json

http://localhost:4000/users

**server**

**java -agentlib:jdwp=transport=dt\_socket,server=y,suspend=y,address=8000 -jar target/backend-0.0.1-SNAPSHOT.jar**

**DOCKER**

**MONGODB**

docker pull mongo

**docker run -d --name my-mongodb -p**

**27017:27017 mongo**

**docker exec -it my-mongodb mongosh**

**/home/vj/Desktop/Angular/myapp-backed/backend**

**run without docker**

**mvn clean install**

**java -jar target/backend-0.0.1-SNAPSHOT.jar**

**run with docker**

**docker build -t myapp-backend:1.0.0 .**

**docker run -p 9090:9090 myapp-backend:1.0.0**

**docker container stop myapp-backend:1.0.0**

**run without docker**

**~/Desktop/Angular/myapp$**

**ng build**

**ng s or npm start**

**run with docker**

**docker build -t myapp-web:1.0.0 .**

**docker run -p 80:80 myapp-web:1.0.0**

**docker container stop myapp-web:1.0.0**

**MUSIC**

**docker build -t music:1.0.0 .**

**docker run -p 8080:8080 music:1.0.0**

**docker container stop music:1.0.0**

***LIST OF Images and Containers***

**docker images**

**docker container ps**

**removes all stopped contianers**

**docker container prune**

**remove all stopped containers**

docker container prune

**Show containers**

**docker container ps -a**

**docker run -p 9090:9090 myapp-backend**

**docker run -p 80:80 myapp-web**

**use mydb**

db.createCollection("users")

db.users.insertOne({name: "John Doe", email: "john@example.com"})

**TEST USE OF PERSISTENCE**

**docker stop my-mongodb**

docker rm my-mongodb

**TEST DB FIND**

**docker exec -it my-mongodb mongo**

use mydb

db.users.find()

**Where is the Docker volume?**

/var/lib/docker/volume

Volumes are also stored as part of the host file system, which is managed by Docker. On Linux, volumes are stored in “**/var/lib/docker/volume**”.

[**https://shamsfiroz.medium.com/running-mongodb-in-docker-a-practical-guide-2b4cf39d5d60**](https://shamsfiroz.medium.com/running-mongodb-in-docker-a-practical-guide-2b4cf39d5d60)

[**https://www.baeldung.com/ops/docker-mounting-multiple-volumes**](https://www.baeldung.com/ops/docker-mounting-multiple-volumes)

[**https://www.mongodb.com/resources/products/compatibilities/spring-boot**](https://www.mongodb.com/resources/products/compatibilities/spring-boot)

[**https://www.baeldung.com/spring-data-**](https://www.baeldung.com/spring-data-)

[**mongodb-tutorial**](https://www.baeldung.com/spring-data-mongodb-tutorial)

**RUN PGADMIN**

**https://www.pgadmin.org/docs/pgadmin4/latest/container\_deployment.html**

docker pull dpage/pgadmin4

**docker run -p 180:180 \ -e 'PGADMIN\_DEFAULT\_EMAIL=user@domain.com' \**

**-e 'PGADMIN\_DEFAULT\_PASSWORD=SuperSecret' \**

**-e 'PGADMIN\_CONFIG\_ENHANCED\_COOKIE\_PROTECTION=True' \**

**-e 'PGADMIN\_CONFIG\_LOGIN\_BANNER="Authorised users only!"' \**

**-e 'PGADMIN\_CONFIG\_CONSOLE\_LOG\_LEVEL=10' \**

**-d dpage/pgadmin4**

**2025-05-27 21:22:40.942 | Error: Database is uninitialized and superuser password is not specified.**

**2025-05-27 21:22:40.942 | You must specify POSTGRES\_PASSWORD to a non-empty value for the**

**2025-05-27 21:22:40.942 | superuser. For example, "-e POSTGRES\_PASSWORD=password" on "docker run".**

**2025-05-27 21:22:40.942 |**

**2025-05-27 21:22:40.942 | You may also use "POSTGRES\_HOST\_AUTH\_METHOD=trust" to allow all**

**2025-05-27 21:22:40.942 | connections without a password. This is \*not\* recommended.**

**2025-05-27 21:22:40.942 |**

**2025-05-27 21:22:40.942 | See PostgreSQL documentation about "trust":**

**2025-05-27 21:22:40.942 | https://www.postgresql.org/docs/current/auth-trust.html**

**How to integrate backend with frontend (myapp, myapp-backed);**

**1. ng build for myapp (web)**

**2 Copy myapp/dist/myapp/browser content into myapp-backedsrc/main/resources/static**

**3. mvn clean install in project myapp-backend**

**4. java -jar target/backend-0.0.1-SNAPSHOT.jard a docker image**

**Angular material**

**ng add @angular/material**

**KAFKA**

**working directory:  
/opt/kafka/bin**

**VYTVORENI TOPICU**

**./kafka-topics.sh --bootstrap-server localhost:9092 --topic myapp-topic --create --partitions 3 --replication-factor 1**

**VYPSANI VSECH TOPICU**

**/opt/kafka/bin $ ./kafka-topics.sh --bootstrap-server localhost:9092 --list**

**Delete topic**

**kafka-topics.sh --bootstrap-server localhost:9092 --delete --topic myapp-topic**

-92 --delete --topic first\_topic --bootstrap-server localhost:9092 --delete --topic f092 --delete --topic first\_topic