**Task Manager App**

This project contains Restful api’s for GoKapture task management system.

File structure:

* Node\_modules: This folder contains all the packages needed for node app to run

.env: This is an environment file which will hold environment variables

* Compose.yaml: This is a docker compose file which creates docker services.
* Dockerfile: This will be used to create an image of node app.
* Package-lock.json: This file contains documenting the precise dependencies, sub-dependencies, and installation paths of node packages.
* Package.json: This file contains dependency listings and version of node packages
* Server.js: This file contains Javascript code for api’s.
* task\_manager.postman\_collection.json: This file contains postman api’s test cases for each functionality.

**Compose.yaml**

This file contains 2 services:

**mysql**: This service uses mysql image to build container. Password for root user is set to “Password” and a default “task\_manager\_db” database is created. Container port 3306 is mapped system port 3306. A volume is defined to store mysql db data.

**nodejs-app**: It uses Dockerfile to build nodeapp container. Port 3000 of container is mapped to port 3000 of system. A volume is mounted to store app data.

**Mysql database:**

Task\_manager\_db has 2 tables

1. **Users**: Columns [id, username, password] id is the primary key and username should be unique and it is case insensitive.
2. **Tasks**: Columns[id, title, description, status, priority, due date, created at, updated at and user id] id is the primary key and title should be unique.

**Server.js**

Docker creates an internal networking for the containers so the containers can be accessed using host “host.docker.internal”.

Node app will be listening on port 3000 and all the api’s will be accessible at “http://localhost:3000”

**Api’s**

POST: /api/register : To register new user.

* Input will be username and password which should be incorporated into a https request.
* If registration is successful then a jwt token in sent to the user.
* Passwords will hashed.

GET: /api/authenticate: To authenticate registered user.

* Input will be username and password which should be incorporated into a req object.
* If authentication is successful then a jwt token in sent to the user.

POST: /api/tasks : To create new tasks.

* Inputs title, description, status, priority, due\_date, username need to be incorporated into a https request.
* It will create a new record and store the task.

GET: /api/tasks : To fetch tasks

* Input will be passed as parameters.
* Based on the input it will fetch all the records or filter out record on status, priority, and due date or search tasks by title or description.

PUT: /api/tasks/:taskId: To update tasks

* TaskId should be passed as query parameter and task details should be passed in https request
* Based on the values task will be updated.

DELETE: /api/tasks/:taskId :Delete a task

* Input will be taskid passed as query parameter.
* Given task will be deleted.

**Steps to setup task\_management\_app:**

Run **docker compose up** to setup the services and app will be accessible at<http://localhost:3000>

**To Create a docker image of node app**

Run **docker build -t my-nodejs-app .**

This command will create a docker image named “my-nodejs-app”.