Tasks

# Task 1

* We need to provide a mechanism of auto migrations. For that we should use for it
  + Umzug is used to track all migrations in sequelize. Added mechanism as follows: -
  + Should check if the migrations are pending and ask user to perform all migrations using prompt-confirm.
  + If the user types yes it should perform all migrations and start the server
  + If the user says no it should not perform the migrations the server should not start.
  + If the migrations are already performed then it should directly run the server.

# TASK 2

Created the feature of multiple usage of ports using portastic i.e., if the project is already running on some port and if the other user tries to run the project on same port. It will prompt a confirmation for available free port and asks user to run that on specific port

# TASK 3

* Prepared a setup for managing APIS.
* Create a folder called API on the root path and for every individual module create a folder for e.g.: - suppose you create a module for customer than a folder named customer must be created inside api folder and inside this customer folder it must contain three folders mainly controller, middleware, services with a route.json file.
* The file structure should look like this
  + Root/api
  + Root/api/customer
  + Root/api/customer/controller
  + Root/api/customer/services
  + Root/api/customer/middleware
  + Root/api/customer/routes.json

And the routes.json file will contain details as follows

[{

        "method": "get",

        "url": "/",

        "globalMiddlewares": "['isLogedIn','all global middlewares']",

        "middlewares": "['all local middleware goes here']",

        "controller": "testshowData"

}]

# TASK 4

* Used npm run dev when working in development environment or else npm start (configured in package.json file)
* Logged different color for error and warning in console using chalk.

# TASK 5

* Create a npm package what all it does is create the basis setup by writing some command on the cli also when I update the code and push it to GitHub it automatically pushes the code to npm registry also I have used GitHub Actions to overcome that challenge
* It also installs the dependencies automatically which are specified in the dependencies array.
* Also now the when running clizard-init the prompt will ask about the user’s database details whether he wants it for development or production. And will after that update the username, database, password field in database.json file

# TASK 6

* Error handling in express i.e., the server should not crash when the error occurs it should display in the console.
* Used documentation of express and found that it can be achieved by using next(err).

# Task 7

* Researched on what do it mean by corn. It is a job scheduler which manages and runs job when specified according to respective time.
* Used npm module called node-cron. The node-corn module is tiny task scheduler in pure JavaScript for node.js based on GNU crontab. This module allows you to schedule task in node.js using full crontab syntax.

# TASK 8

* Configured file upload setup using multer as a npm package. Also provided file filtration to overcome the challenge of which file to accept and which to not.