***Creating fastify todo-apis:***

[Fast and low overhead web framework, for Node.js | Fastify](https://fastify.dev/)

DATABASE USED: MONGODB

STEPS:

* npm install -g fastify-cli
* fastify generate myproject
* cd myproject
* npm install fastify-plugin mongoose
* created a db by the name mydb in mongodb compass and copy the connection string
* now start with app.js and import the required dependenices like fastify, mongoose and setup a basic server that listens to PORT=3000 then write a code to connect to mongodb database using mongoose.connect function , keeping all the code within a same asynchronous startServer function.
* Build a todoSchema to define the structure of a database in the models folder
* Now define the routes in routes/todos.js file and import the mongodb model i.e. todo.js and keep all the required routes in a asynchronous function routes and export it using fastify-plugin
* Now come to app.js and register the routes (plugin) using fastify.register function. (keeping that within async startServer function only)
* Now install REST Client to test the apis
* Start the server using node app.js or npm run dev or npm start and then the console will be displayed telling ,that you have connected to the database and routes has been registered (only if you placed console.log() after those functions)
* Now test your apis using rest client easily within the same window.

Mongoose built-in functions used:

* todo.save()
* ObjectID.isValid(id)
* ToDo.findById(id)
* …todo.toObject()
* ToDo.findByIdAndUpdate(id,{title},{new: true})
* ToDo.find()
* Todo.findByIdAndUpdate( id, { completed: true, dateOfCompletion: Date.now()},

new: true } );

* Todo.findByIdAndDelete(id)

**REQUEST TO TEST THE APIS:**

# GET Request to Retrieve All Todos

GET http://localhost:3000/todos

# GET Request to Retrieve a Specific Todo

GET http://localhost:3000/todos/64dc9c039cffec3db6282e95

# POST Request to Create a New Todo

POST http://localhost:3000/todos

Content-Type: application/json

{

"title": "Buy groceries"

}

# PUT Request to Update a Todo

PUT http://localhost:3000/todos/64dc9c039cffec3db6282e95

Content-Type: application/json

{

"title": "Buy fruits"

}

# PUT Request to Mark a Todo as Completed

PUT http://localhost:3000/todos/64dc9c039cffec3db6282e95/markComplete

Content-Type: application/json

{

"completed": true

}

# DELETE Request to Delete a Todo

DELETE http://localhost:3000/todos/64dc9c039cffec3db6282e95