goto to backend folder

right click open terminal

npm init -y ----this will create a package.json template

code . ----- open folder in vs code

open terminal

npm i --save-dev typescript ---- to install typescript

npx tsc --init ---necessary for typescript

npm i express

npm i --save-dev @types/express

npx tsc to convert ts code to js since we cant directly node "server.ts"

\*\*\*\*\*\*\*\*\*\*\*\*\*\* THIS WE DO IN PRODUCTION \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

to make server.js mimic main file

in tsconfig.json uncomment outdir and insert ./dist

then do npx tsc in terminal

dist folder will be created to run the node = node dist/server.js

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\* THIS WE DO IN DEVELOPMENT\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-To constantly make changes on port without doing npx tsc we use DevDependency of nodemon

npm i --save-dev nodemon

-To use nodemon with Typescript we need node ts

npm i -D ts-node

now finally npx nodemon src/server.ts

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-D(--save-dev)

now install more devdependency = npm i -D eslint

to setup eslint = npx eslint --init

-It checks if there are some common typos or undefined errors

npx eslint .est .ts

-we can set it up in package json as a short key

"lint":"eslint . --ext .ts",

-If we have an undefined port our project will still work to solve it we download an ENVALID dependency

and create util folder and make a validateEnv file

after creating the file import in file where env is used and replace

process.env ===> env.\*\*\*\*

then we make models in src folder

we make mongoose schema there

**SOME TYPESCRIPT STUFF FOR MODELS**

type List = InferSchemaType<typeof listSchema>;

export default model<List>("Todo List", listSchema);

for the promise we can use .find().exec() also we need to add async function

also we cant use it in top level that’s why we did .then() and .catch in server.ts

# MIDDLEWARES :

# ENHANCING ERROR HANDLING

Firstly we download few dependency: npm i http-errors

And a dev dependency : npm i -D @types/http-errors

import createHttpError ,{isHttpError} from 'http-errors';

# EXPRESS ERROR HANDLER

For async we add try-catch for sync function we don’t have to

app.get('/', async (req, res, next) => {

   try {

      throw Error("HACKES");

      const todolist = await listModel.find().exec();

      res.status(200).json(todolist);

   } catch (error) {

      next(error)

   }

});

// eslint-disable-next-line @typescript-eslint/no-unused-vars

app.use((error: unknown, req: Request, res: Response, next: NextFunction) => {

   console.error(error);

   let errorMessage = 'An Unknown Error occured';

   let statuscode = 500;

   if(isHttpError(error)) {

      statuscode =error.status;

      errorMessage=error.message;

   }

   res.status(statuscode).json({ error: errorMessage })

});

# UNDEFINED ROUTES

We write it above error handler and below route

app.use((req,res,next) => {

   next(createHttpError(404,"Endpoint Not Found !"))

});

For additional logging we use another dependency called as morgan

npm I morgan

then: npm i --save-dev @types/morgan

app.use(morgan('dev'));

just have to use above line.

# FRONTEND

npx create-react-app frontend --template typescript

npm i react-bootstrap bootstrap

paste css in index.tsx:

import 'bootstrap/dist/css/bootstrap.min.css';