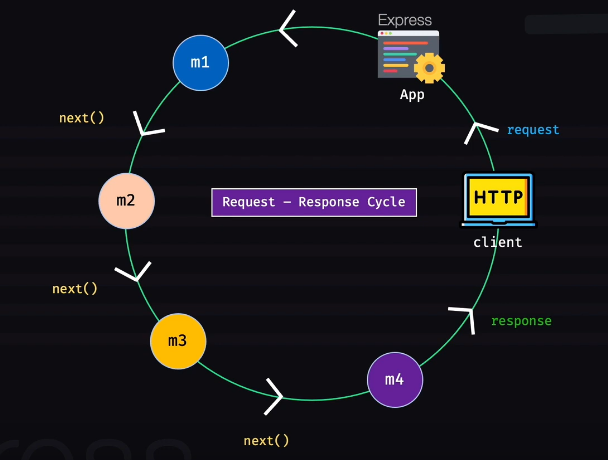
**Middleware Concept**

Ajke amra express js er shobche important concept er ekta concept middleware niye discuss korbo. Express js muloto middleware er upor base korei kaj kore.

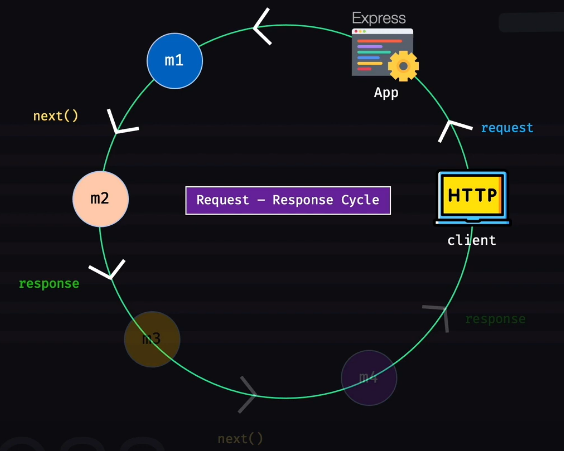
Middleware hocche ekta simple js function jar kache request ebong response object er access ache. Ebong tar kache next name ekta function er access ache.

Middleware bujhte hole request ebong response cycle ta bujhte hobe. Mone kori, yellow icon ta holo http client. Ei client amar express app e ekta request kore. Ei request ta amar express app e ashar pore client ekta response er jonno wait kore thake. Ei response ta deyar age amar express app bivinno function er moddho diye ghurte pare.

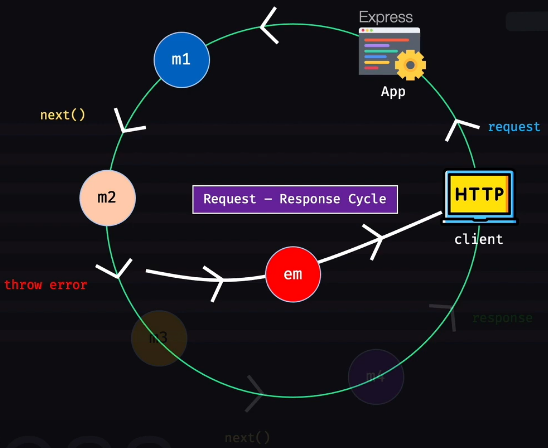
For e.g. express app er kache request ta ashlo. Then she shei request ta niye m1 namok ekta middleware function er kache pathaya dilo. Ei m1 besh kichu kaj kore ekta next function call kore dilo. Er fole process ta porer middleware function m2 er kache chole gelo. M2 function kichu kaj kore next function call kore dilo ebong poroborti middleware m3 er kache pathaya dilo. then m3 ebong tarpor m4 er kaj sheshe m4 middleware amader client ke response diye dilo. request ebong response cycle er middleware evabei kaj kore.



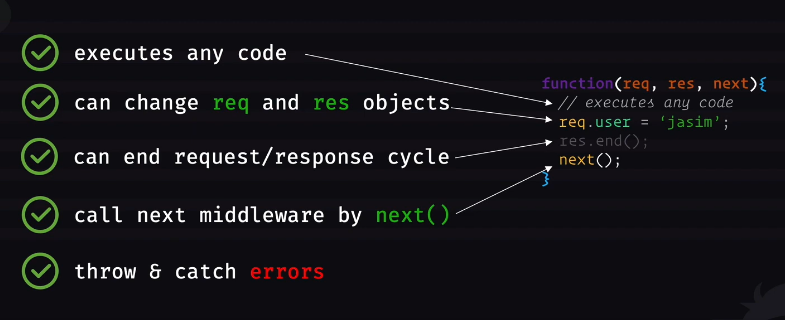
Joto Khushi toto middleware ami amar application e rakhte pari.



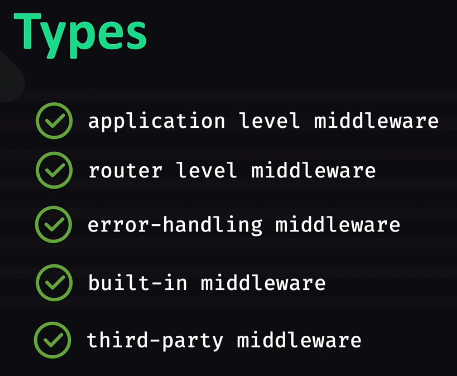
Mone kori, ei jaygatay m2 middleware next function call korlo na. jekono jayga thekei jekono middleware function next() call na kore response (res) call kore dite pare. Orthat kono middleware theke jodi res.send() ba res.end() call hoy sheikhanei middleware er kaj shesh hoye jay ebong directly client er kache response chole jay. Orthat jekono middleware jekono shomoy next call na kore response ke end kore deyar khomota rakhe. In that case chain e thaka poroborti middleware functiongula ar call hobe na.



Echara jekono middleware chaile error throw korte pare. For e.g. m2 middleware jodi kono error throw kore process ta porer middleware gulate ar jabe na. amader ekta error handling middleware em banate hobe. Jokhon kono middleware kono error throw korbe tokhon amader error handling middleware em sheta catch korbe. catch kore handle kore finally client ke response pathay dibe je ‘there was an error’ or something like this. Ar jodi amra error handling middleware na rakhi ebong kono middleware (m1,m2,m3…) kono error throw kore tahole amader application ta crash kore jabe.



Right side e ache ekta simple middleware function. Tar kache req, res ebong next function er access ache. She jekono dhoroner js code execute korte pare. Middleware function ta request ebong response object ke change korte pare. Notun property add korte pare. She jekono jayga theke next function call na kore request ebong response cycle ke end kore dite pare. Orthat res.json(), res.end() ba res.send() kore response ke end kore dite pare. Echara next() call kore process ta porer middleware er kache pathaya dite pare. Ebong catch & throw error korte pare.



Express application e ki ki type er middleware thakte pare?

1. Application level middleware amra app.use() er maddhome use korbo.
2. Router create kore router.use() kore router level middleware use korte pari.
3. Error handling middleware ta ektu age dekhlam.
4. Express.json(), express.static() --- egula built in middleware.
5. CookieParser() ekta third party middleware. Express e joto dhoroner third party package use korbo beshirvagi amader ekta middleware dibe. Amra application er jekono jaygay shegula use korte parbo.

Now we’ll dive right into the examples….

Index.js …

*const* express = *require*("express");

*const* app = *express*();

app.*get*("/about", (*req*, *res*) *=>* {

  res.*send*("This is about page");

});

app.*listen*(3000, () *=>* {

  console.*log*("Listening on port 3000");

});

Ekhon ekhane ekta middleware function anbo. MyMiddleware name ekta function likhbo. ei function parameter akare req, res ar next ney. Porer middleware e jawar dorkar hole next call korbe. ar jodi myMiddleware ei response diye dey tahole ar next call korar dorkar nai. But by default shob middleware ei tinta jinish pay. Ekhon myMiddleware e kichu kaj kore ami porer middleware e pathay dite pari. Porer middleware bolte amar /about route er callback function. Etao kintu ekta middleware. myMiddleware er function body lekhar por app ke bole dite hobe eta use koro. App.use(myMiddleware) kore app ke bole dibo myMiddleware use korte.jehetu app myMiddleware use kore shehetu Jokhoni kono client er kach theke server er kache request ashbe tokhon first e myMiddleware function ta call hobe. Then /about route er middleware e jabe.

Index.js…

*const* express = *require*("express");

*const* app = *express*();

*const* *myMiddleware* = (*req*, *res*, *next*) *=>* {

  console.*log*("I am logging");

*next*();

};

app.*use*(myMiddleware);

app.*get*("/about", (*req*, *res*) *=>* {

  res.*send*("This is about page");

});

app.*listen*(3000, () *=>* {

  console.*log*("Listening on port 3000");

});

Server on kore /about route e hit korle console e ‘I am logging’ lekha ashtese. Abar postman e ‘this is about page’ o lekha paitesi.

Ekhon myMidleware1 ebong myMiddleware2 name duita middleware banabo.

Index.js …

*const* express = *require*("express");

*const* app = *express*();

*const* *myMiddleware1* = (*req*, *res*, *next*) *=>* {

  console.*log*("I am logging 1");

*next*();

};

*const* *myMiddleware2* = (*req*, *res*, *next*) *=>* {

  console.*log*("I am logging 2");

*next*();

};

app.*use*(myMiddleware1);

app.*use*(myMiddleware2);

app.*get*("/about", (*req*, *res*) *=>* {

  res.*send*("This is about page");

});

app.*listen*(3000, () *=>* {

  console.*log*("Listening on port 3000");

});

Server on kore /about route e hit korle console e ‘I am logging 1’ ebong ‘I am logging 2’ duitai ashtese.

Ekhon arekta middleware likhbo logger name.

Index.js …

*const* express = *require*("express");

*const* app = *express*();

*const* *logger* = (*req*, *res*, *next*) *=>* {

  console.*log*(

    `${new *Date*(Date.*now*()).*toLocaleString*()} -- ${req.method} -- ${

      req.originalUrl

    } -- ${req.protocol} -- ${req.ip}`

  );

*next*();

};

app.*use*(logger);

app.*get*("/about", (*req*, *res*) *=>* {

  res.*send*("This is about page");

});

app.*get*("/home", (*req*, *res*) *=>* {

  res.*send*("This is home page.");

});

app.*listen*(3000, () *=>* {

  console.*log*("Listening on port 3000");

});

Ei logger middleware er kaj holo kono request ashle tar details ta log kora. App.use(logger) er fole amar pura application ta ei logger ke use korbe. amar application er shomosto route ei logger er moddho diye jabe. Eta bojhar jonno ekta /home route rekhechi. Ekhon server on kore /about kingba /home jei route ei hit kori na keno console e log details dekhte pacchi like ‘10/9/2022, 1:09:38 PM -- GET -- /home -- http -- ::1’. Ekhane logger er moddhe amra next call na kore res.end() kore response end kore dite pari. Shekehtre /about kingba /home route er CB ta call hobe na. just loging hobe. Log details console e dekhabe.

Next function er moddhe kono value dile like next(‘hello’) dile postman e error show korbe. but console e log details ebong tar niche ‘hello’ lekha show korbe. so next er moddhe faka dite hobe.

Ekhane app.use(logger) korechi. So application level e ami ei logger middleware ke use korsi. Echara amra router level eo middleware use korte pari.

Index.js …

*const* express = *require*("express");

*const* app = *express*();

*const* adminRouter = express.*Router*();

*const* *logger* = (*req*, *res*, *next*) *=>* {

  console.*log*(

    `${new *Date*(Date.*now*()).*toLocaleString*()} -- ${req.method} -- ${

      req.originalUrl

    } -- ${req.protocol} -- ${req.ip}`

  );

*next*();

};

adminRouter.*use*(logger);

adminRouter.*get*("/dashboard", (*req*, *res*) *=>* {

  res.*send*("This is admin dashboard.");

});

app.*use*("/admin", adminRouter);

app.*get*("/about", (*req*, *res*) *=>* {

  res.*send*("This is about page.");

});

app.*listen*(3000, () *=>* {

  console.*log*("Listening on port 3000");

});

Ekhane app ke bole dilam /admin route pele adminRouter subapp ta use korte. Abar adminRouter subapp ke bole dilam logger middleware ta use korte. Orthat logger middleware ta pura app e use hobe na, only subapp e use hobe. Ekhon /about route e hit korle just postman e response ashe “This is about page”. But console e log details ashena. orthat logger middleware ta call hoy na. onnodike /admin/dashboard url e hit korle postman e response ashe “This is admin dashboard.” Abar console e log details o ashe. Orthat logger call hoy. Abar localhost:3000/admin url e hit korle postman e error ashe. But console log details ashe.

Ekhon amra third party logger dekhbo. For e.g. amra cookie parser use korbo. ‘npm install cookie-parser’ kore sheta install kore fellam. Const cookieParser = require(‘cookie-parser’) kore aamader application e parser ta niye ashlam.

*const* express = *require*("express");

*const* cookieParser = *require*("cookie-parser")

*const* app = *express*();

app.*use*(*cookieParser*())

ei cookieParser jokhon ami require korchi tokhon sheta ekta third party library. Ei cookieParser er moddhe ja pacchi sheta ke call kore dile she ekta middleware return kore. Shetakei ami application level e use kortesi.

So amra app level, router level ebong third party level e middleware er kaj dekhlam. Echara built in middleware for e.g. express.json() use korte pari.

Now at last error handling middleware dekhbo. Logger function er last e next call korsi. Sheta na kore ‘throw new Error(‘This is an error’) --- eta likhe error throw kore dite pari. Error jehetu throw korchi shehetu response end hocche na. ami jodi ei error take arekta middleware diye catch na kori amar application crash korbe. so erokom catch korar jonno common middleware banate hole amra simple error handling middleware like errorMiddleware nibo. Eta onnanno middleware function er motoi. But eta ekta extra parameter ‘err’ ney.

Index.js …

*const* express = *require*("express");

*const* app = *express*();

*const* adminRouter = express.*Router*();

*const* *logger* = (*req*, *res*, *next*) *=>* {

  console.*log*(

    `${new *Date*(Date.*now*()).*toLocaleString*()} -- ${req.method} -- ${

      req.originalUrl

    } -- ${req.protocol} -- ${req.ip}`

  );

  throw new *Error*("This is an error.");

};

adminRouter.*use*(logger);

adminRouter.*get*("/dashboard", (*req*, *res*) *=>* {

  res.*send*("This is admin dashboard.");

});

app.*use*("/admin", adminRouter);

app.*get*("/about", (*req*, *res*) *=>* {

  res.*send*("This is about page.");

});

*const* *errorMiddleware* = (*err*, *req*, *res*, *next*) *=>* {

  console.*log*(err. message);

  res.*status*(500).*send*("there was a server side error.");

};

adminRouter.*use*(errorMiddleware);

app.*listen*(3000, () *=>* {

  console.*log*("Listening on port 3000");

});

So jokhoni ei 4 parameterala middleware ta amar application e use korbo tokhoni express bujhe jabe eta holo error handling middleware. Ekhon logger theke throw kora error ta errorMiddleware er err e chole ashbe. errorMiddleware e console e err ta dekhanor bebostha korlam. Erpor response takeo end kore dilam. Jehetu logger ta ami adminRouter e use kortesi ebong logger thekei error throw kortese shehetu adminRouter.use(errorMiddleware) kore adminRouter ke bole dilam errorMiddleware ta use korte. now server on kore postman e /admin/dashboard url e hit korle postman er response section e lekha ashtese ‘there was a server side error.’ Ebong console eo error message (jeta logger function er moddhe likhe disilam --- this is an error) show kortese. So logger middleware er moto onno jekeno middleware theke error throw kore error handling middleware (errorMiddleware) diye catch korte parbo.

Finally arekta jinish dekhbo.

Ekhon emon onek shomoy hote pare logger function ta bairer kono file e ache ba ei file ei ache but logger ta te ami bairer theke kono argument ba data pass korte chai orthat ami jodi ekta configurable middleware banate chai tahole ki korte hobe? In that case amader ekta normal functioni dite hobe ja jekono data receive kore ebong function ta ekta middleware ke return kore.

Index.js …

*const* express = *require*("express");

*const* app = *express*();

*const* adminRouter = express.*Router*();

*const* *loggerWrapper* = (*options*) *=>* {

*return* *function* (*req*, *res*, *next*) {

    if (options.log) {

      console.*log*(

        `${new *Date*(Date.*now*()).*toLocaleString*()} -- ${req.method} -- ${

          req.originalUrl

        } -- ${req.protocol} -- ${req.ip}`

      );

*next*();

    } else {

      throw new *Error*("Failed log");

    }

  };

};

adminRouter.*use*(*loggerWrapper*({ log: false }));

adminRouter.*get*("/dashboard", (*req*, *res*) *=>* {

  res.*send*("This is admin dashboard.");

});

app.*use*("/admin", adminRouter);

app.*get*("/about", (*req*, *res*) *=>* {

  res.*send*("This is about page.");

});

*const* *errorMiddleware* = (*err*, *req*, *res*, *next*) *=>* {

  console.*log*(err.message);

  res.*status*(500).*send*("there was a server side error.");

};

adminRouter.*use*(errorMiddleware);

app.*listen*(3000, () *=>* {

  console.*log*("Listening on port 3000");

});

Logger function ta ke shoriye loggerWrapper name ekta function nilam. Ei function theke main middleware ta return korlam. Main middleware ta bairer theke asha value true naki false shetar upor depend kore value dibe. Jodi true hoy then log details dibe. Ar jodi false hoy tahole error throw kore error handling middleware er kache process ta pathay dibe.