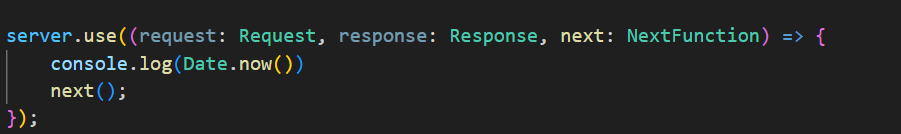
**Node.js – 2023-05-04**

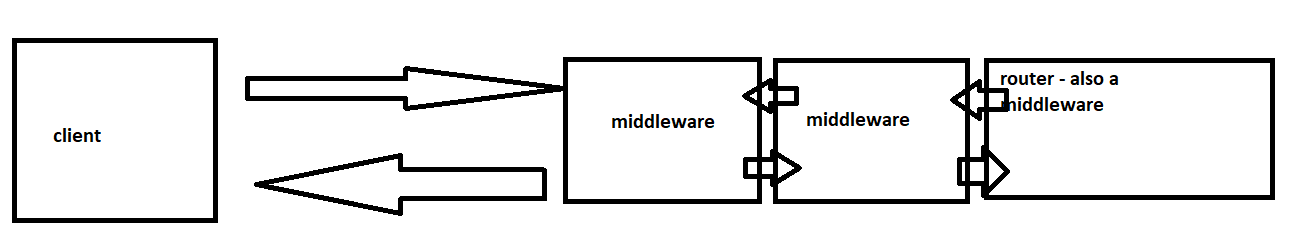
Middleware

אלו פונקציות המופעלות בצורה אוטומטית, כאשר בתוכן יש לי גישה ל context מסוים.



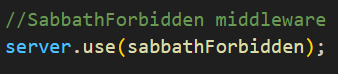
כמה דברים חשובים שצריך לדעת על middleware

1. בדרך כלל, middleware מתאפיין בכך שהוא מקבל data רלוונטי (במקרה שלנו response ו request ) וגם פונקציה, אשר תפקידה לעצור את ה flow הנוכחי או לאפשר לו להמשיך הלאה.
2. אם אין יותר לאן להעביר את ה flow, הוא עוצר. תמיד באותה הדרך. כלומר – יש שרשרת של פונקציות שנקראות אחת אחרי השניה ( המיקום הוא מאוד רלוונטי). האחרונה שולחת בחזרה ל client את התגובה. בעצם אנחנו מקבלים את הבקשה מה client, עושים "מסג" ל data ושולחים לו את ה response
3. חייבים תמיד לקרוא לפונקציה ( במקרה שלנו next) כדי לעבור לפונקציה הבאה בשרשרת. או לחילופין, חייבים להחזיר תגובה כלשהי ל client ; אחרת, ה client לא יקבל שום דבר (שום תגובה) ויהיה timeout מהשרת.
4. כל השינויים שנעשו בפרמטרים המוזרקים ( במקרה שלנו req, res, next ) – יהיו זמינים גם בפונקציה בשרשרת הבאה.
5. במקרה של ה express middleware ה request ו ה response, הם שונים בכל בקשה (האובייקטים שונים לחלוטין).

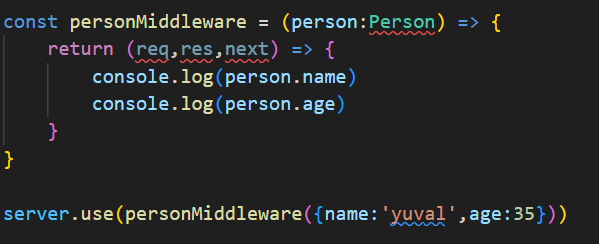


כלומר – כדי לא להמשיך הלאה ( ל middleware הבא), אלא להחזיר מידע כלשהו, אנחנו מחזירים response. ה controllers שלנו הם גם Middleware, אבל מאחר והם routes, אנחנו מחוייבים להחזיר מהם תגובה לclient או שגיאה.

1. ניתן לממש Middlewares בקובץ הראשי (app.ts) – אשר מתבצע מנקודת ה biding ועד סוף השרשרת.



1. אנחנו גם יכולים לכתוב פונקציה שמחזירה middleware ( במידה ואנחנו רוצים להעביר configuration or data ) .



1. ברמת route ספציפי 

Catch all middleware

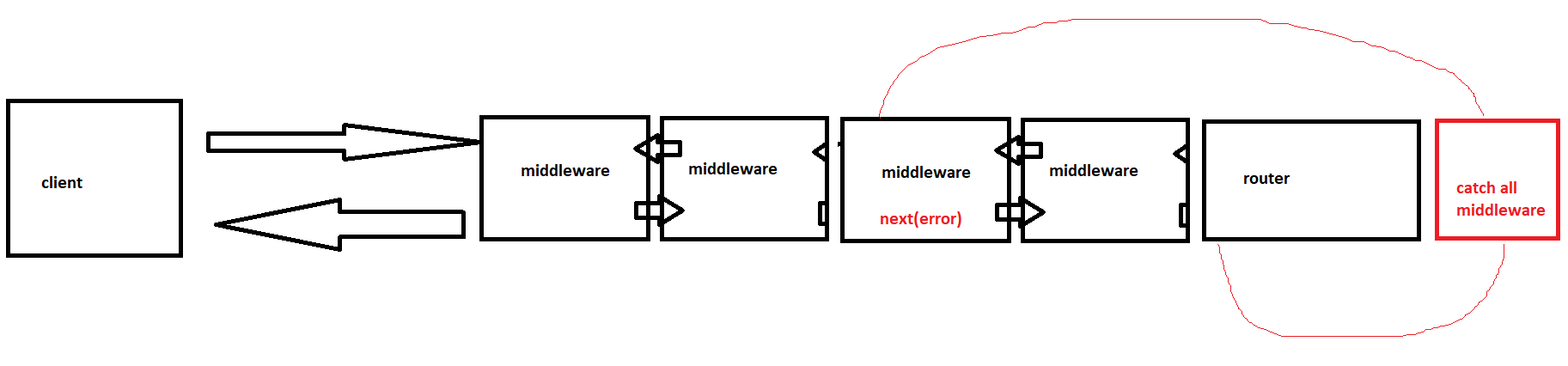
Catch-all זה מונח המתאר מצב שיש אזור אחד בקוד, שכל החריגות מגיעות אליו.

Catch-all middleware , זה הוא middleware מיוחד, המכיל 4 arguments, כאשר הראשון הוא ה error שהוקפץ (השאר זהים – req, res, next ) .

ה middleware חייב להירשם אחרון ב binding (כלומר בסוף התהליך, אחרי כל שאר ה middlewares וה routes )

הוא מופעל כאשר מעבירים argument כלשהו ל next().

בדרך כלל נהוג להעביר אובייקט עם סטטוס השגיאה ותיאור השגיאה.



טיפול בשגיאות

יש מספר סוגים של שגיאות שעלינו לטפל בצד שרת.

עבור השגיאות השונות הללו, עלינו להחזיר ללקוח הודעה מדויקת.

לצורך כך, אנו בונים מחלקות לתיאור השגיאות.

סוגי השגיאות הנפוצות שלרוב יהיו בשרתים:

1. קריסה בשרת (שרת לא היה בסדר) סטטוס : 500
2. גלישה ל ROUTE שלא קיים ( 404)
3. בקשה של RESOURCE שלא קיים (404)
4. שגיאות ולידציה – POST, PUT, PATCH, בדרך כלל תלוי במודל (SCHEMA ) של אובייקט כלשהו (400)

חוקי הוולידציה מוגדרים במחלקות ה MODEL (בגלל שיש המון בדיקות חוקיות).

Validation library – npm I joi