Day 14 - Middlewares

1. Middle wares can be reused.
2. Ticket buyer, seller and guard example.
3. Code:

```
const express = require('express');

const app = express();

function isOldEnough(age) {
  if (age >= 14) {
    return true;
  } else {
    return false;
  }
}

// app.use(isOldEnoughMiddleware);

function isOldEnoughMiddleware(req,res,next){
  const age = req.query.age;
  if (age >= 14){
    next();
  } else {
    res.status(401).json({
      msg : "You are not old enough to access this ride :("
    })
  }
}

app.get("/ride2", isOldEnoughMiddleware, function(req,res){
  res.json({
    msg : "You are old enough for this ride, enjoy!"
  })
})

app.get('/ride1', function(req, res) {
  if (isOldEnough(req.query.age)) {
    res.json({
      msg: "You have successfully ridden the ride 1"
    });
  } else {
    res.status(411).json({
      msg: "Sorry, you are not of age yet! :("
    });
  }
});
app.listen(3000);
```

4. Important points:

This code is fine but it only has one problem here which is it does not have any check marks.
EX: a Person of specific age, or gender is getting allowed to get a ride in the fair.
Hence let's introduce a function which returns a boolean for this checking.

1. function returns a boolean if age >= 14
   this was one way of checking via a ticket checker.
2. two ways to call a middleware:
   a. to call the middleware in a function
   b. to call it using app.use(isOldEnoughMiddleware) at the start of a function call -->

the
ORDER matters here.

5. Rate limiting middlewares:
   A single user can’t bombard your servers.
   Hence these come in place to protect server crashing.

```
//--> Rate limiting middleware
app.use(function (req, res, next) {
  const userId = req.headers["user-id"];
  //getting the userId info from the header
  if (numberOfRequestsForUser[userId]) {
    numberOfRequestsForUser[userId] = numberOfRequestsForUser[us
    if (numberOfRequestsForUser[userId] > 5) {
      res.status(404).json({ msg: "No Entry" });
    } else {
      next();
    }
  } else {
    numberOfRequestsForUser[userId] = 1;
    next();
  }
});
```

In the real world it happens through IPV6 protocol, where IP’s are tracked.

6. Error Handling Middleware
   This is used to define an error and handle it.

```
app.use(function(err,req,res,next){
});
```