We have an issue with throwing errors due to validation. This is due to the behavior of the **express-validator** library. If validation fails in a validation chain, **express-validator** doesn't throw an error of type Error, instead it creates an array of errors. Each of these errors is a JavaScript object (henceforth I will call it ERR). However, we want our desired program flow to be such that whenever any error is thrown, then it should be caught by the global error handler. In this case, it isn't possible for the global error handler to catch these ERRs since they aren't of type Error. A few ways of dealing with the situation are given below:

1) If we are using only a single validator in the validation chain, then we can extract the "msg" key from ERR and create a CustomError object with it along with an HTTP error code. However, the ERR contains a lot of information. This information will not be shown in the CustomError and it will also be against the desired program flow.

2) If the validation chain contains more than one function, then the strategy outlined in 1) may not work in a situation where more than 1 function in the validation chain throws an error. While an array of CustomError objects with the "msg" keys from each of the ERRs can be created, when that array is passed to next(), then it will not be able to read the CustomError objects in it and will throw a default HTTP 500 Internal Server Error.

**3) We could define an ExpressValidationError class by extending the Error class such that it could accept the array of ERRs as an argument.**

4) We could use the validator.js library and create validation chain functions by using if-else statements or conditional operators, by using the validator functions directly. This way we can define CustomError for each type of validation failure.

The highlighted approach is being followed.