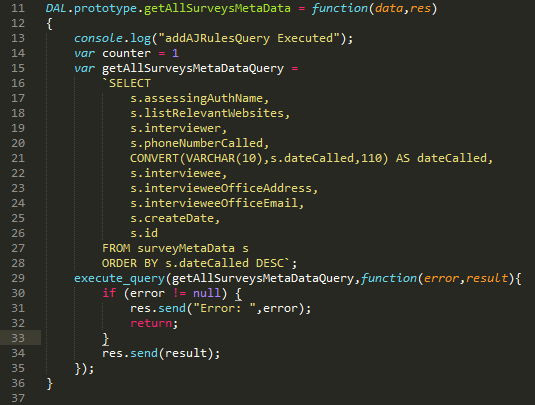
**Coding Conventions**

**1. Use alias instead of table name in the query.**

**Don’t do the following.**

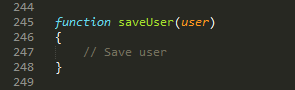


**Do the following.**

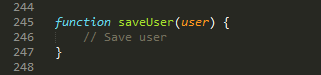


**2. Open braces should be on the same line.**

**Don’t do the following.**



**Do the following.**



**3. Logging on the console.**

**Use console.log() for debugging purposes only. It should be removed before committing the changes. If you need debugging on the server, add console.log() and debug it. When debugging is completed then remove it and commit the changes.**

**4. Use try/catch for exception handling.**

**Try/catch should be used so that to avoid crashing the application. A meaningful message should be sent to the client side when an exception occurs.**

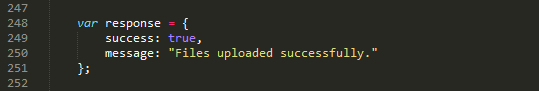
**5. Use server side pagination.**

**Server side pagination should be implemented when working with large amount of data. It improves the performance by retrieving only those records from database that need to be displayed to the user.**

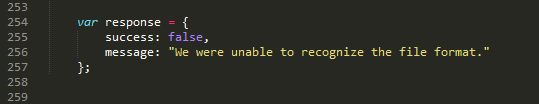
**6. Use a standard response object.**

**A standard response object should be sent to the client.**

**An example of a success response:**

****

**An example of a failure response:**

****

**7. Catching exception in BLL.**

**We usually use try/catch in BLL to catch exceptions occurred at lower levels (e.g. DAL).**

**8. DAL should not send response to the client directly.**

**We should not send a response to the client directly from DAL. Result should be returned to BLL and BLL will send response to the client.**

**9. Exception object should not be sent to client.**

**Exception object (specially sent by DAL) should not be sent to the client because it can contain sensitive information like database or table name.**

**10. Preventing SQL injection.**

**Don’t concatenate parameters to the query directly, pass parameters to the query instead.**

**11. Use constants for common things.**

**Use constants for date/time and messages and use them in other places.**

**12. Preventing CQL injection.**

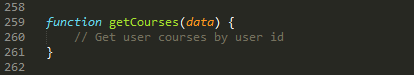
**Don’t concatenate parameters to the cypher query directly, pass parameters to the query instead.**

**13. Avoid unused variables.**

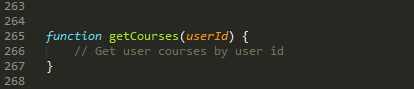
**Avoid creating a variable that you are not using in your code.**

**14. Use meaningful names for identifiers.**

**Don’t do the following (parameter “data” is not descriptive).**

****

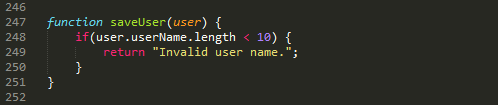
**Do the following (parameter “userId” is descriptive).**

****

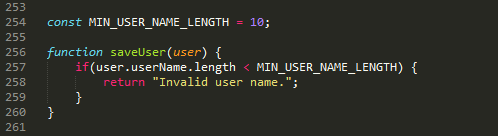
**15. Avoid using hard coded values (use constants).**

**Avoid using hard coded values in the statements, use constants instead.**

**Don’t do the following.**

****

**Do the following.**



**16. Use procedures.**

**Use procedures for bulk entries if possible.**

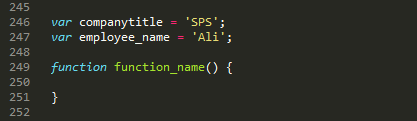
**17. Log exceptions.**

**Save exception details to database.**

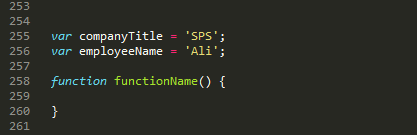
**18. Follow the following naming conventions.**

**a. Use lower camel case for variables, properties and functions.**

**Don’t do the following.**

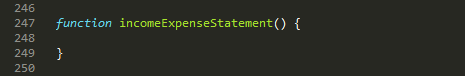


**Do the following.**

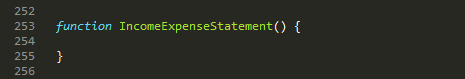


**b. Use upper camel case for class name.**

**Don’t do the following.**

****

**Do the following.**

****

**c. Use uppercase for constants.**

**Constants should be declared with all uppercase characters with underscore between words.**

**Don’t do the following.**

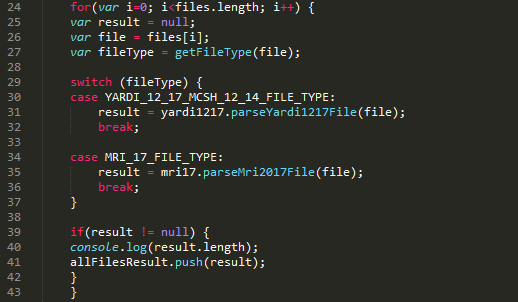
****

**Do the following.**

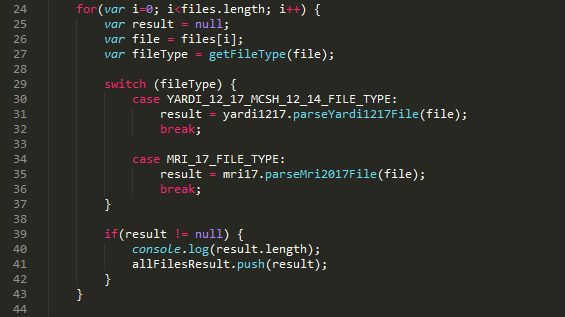


**19. Use indentation.**

**Don’t do the following.**

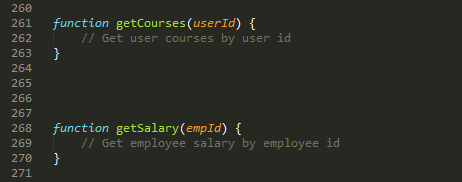
****

**Do the following.**

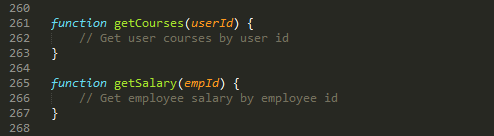


**20. Use one blank line between two functions.**

**Don’t do the following.**



**Do the following.**



**21. Add comments.**

**If you have implemented a complex logic, it’s better to explain it by adding comments in the code.**

**22. Document code.**

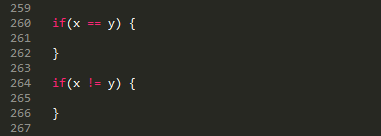
**Document the code so that we would be able to generate documentation for our project.**

**23. Remove commented out code.**

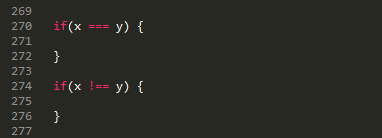
**It’s better to remove commented out code so that the project will be clean.**

**24. Use triple equality operator.**

**Don’t do the following.**



**Do the following.**



**25. Shift email code to util module.**

**Write email sending code in util module so that it can be used from other functions as well.**

**26. Add validations in parsers.**

**Add validations in parsers. For example, if we don’t find required fields we will send an error to the client.**