TABLE OF CONTENT

W03: JAVASCRIPT

1. **THEORY**
2. **Function**
    a. Function Statement
    b. Function Expression
    c. Function Declaration
    d. Anonymous function
    e. Named Function Expression
    f. Functional Programing
    g. Higher order function
    h. First class function
3. Advantages and disadvantages of JS
4. Scope, Lexical scope
5. Prototype
6. **Closure**
    a. Disadvantage
    b. Uses
7. Garbage collection
8. **Hoisting**
    a. TDZ
    b. let, const vs var
    c. Function vs arrow function
9. Call Apply Bind
10. This Keyword
11. **String Methods**
    a. Length
    b. toUpperCase, LowerCase
    c. Trim
    d. Pad
    e. charAt
    f. Split
    g. Concat
    h. substring
12. **Array Methods**
    a. Map
    b. Filter
    c. Reduce
    d. Find
    e. Sort
    f. Foreach
    g. Push
    h. Pop
    i. Shift
    j. Unshift
    k. Slice
    l. Splice
13. **Object Methods**
    a. freeze
14. Callback and callback hell
15. **Promise**
    a. Promise.all
    b. Promise.allSettled
    c. Promise.race
    d. Thenable
    e. Finally
    f. Catch
16. Async await
17. Spread and Rest Operator
18. DOM, BOM
19. Call stack
20. Event loop
21. **ES6 and its features**
    a. Let, Var, Const
    b. Ternary operator
    c. Arrow function
    d. Template literals
    e. Default Parameters
    f. Classes
    g. Modules
    h. Iterators
    i. Object & Array Destructuring
22. **Primitive and non-primitive**
    a. Pass by value and pass by reference
23. Message queue
24. Life
25. Generator
26. **Prototype**
    a. Prototype chain
    b. Prototypal Inheritance
27. JavaScript is dynamically types
28. Currying
29. **Type Casting**
    a. Implicit (Coercion)
    b. Explicit (Conversion)
30. Microtask queue

31. Shallow copy
32. Deep copy
33. Immutable

34. VS

a. == and ===
b. Let, const, var
c. Synchronous vs asynchronous
d. While vs do while
e. Foreach Vs Map
f. Parameters, Arguments
g. for in, for of
h. Undefined, Null
i. Keywords & Identifiers

W04: NODE.JS EXPRESS

THEORY

1. What is Node.js
2. why v8 Engine
3. Advantages & Disadvantages of Node.js
4. How node works
5. Node Module System
6. REPL, Cli
7. NPX
8. Globals
    a. __dirname
    b. __filename
    c. Module
    d. Process
9. Modules
    a. Core Modules.
    b. local Modules.
    c. Third-party Modules.
    d. module.exports:{}
    e. require
    f. ESM
        i. import and export
10. NPM
    a. local and global
    b. npm init
    c. npm install or i
11. Nodemon
    a. scripts
        i. start
        ii. dev
    b. npm run dev
12. package.json
13. package-lock.json
14. Event loop
15. Event Queue
16. Events
    a. Events emitter
    b. Http module
17. Streams
    a. type of streams
        i. writable, readable, duplex, transform
    b. createReadStream()
    c. pipe()

18. HTTP

    a. https
    b. How does it work?
    c. request response cycle
    d. Stateless protocol
        i. Local storage, Sessions and Cookies
    e. Request
        i. General (start line)
            1. method/target/version
        ii. header
        iii. body
    f. Response
        i. General (start line)
            1. version/statuscode/statustext
        ii. header
            1. content type
        iii. body
            1. requested resource
    g. HTTP Methods
        i. GET
        ii. POST
        iii. PUT
        iv. DELETE
    h. Idempotent
    i. Headers
    j. Status code
        i. 1xx: Informational
        ii. 2xx: Success
            1. 200 - Success
            2. 201 - Success and created
        iii. 3xx: Redirect
            1. 301: moved to new URL

            2. 304: not changed
        iv. 4xx: Client Error
            1. 401: Unauthorised
            2. 402: 402 Payment Required
            3. 403: Forbidden
            4. 404: page not found
        v. 5xx: Server Error
    k. MIME type
    l. HTTP v2
    m. TCP and IP

19.EXPRESS

20. npm install express –save
21. app = express()
    a. get()
        i. status()
        ii. send()
        iii. sendFile()
    b. post()
        i. express.urlencode()
        ii. Form vs JS
    c. put()
    d. patch()
    e. delete()
    f. all()
    g. use()
    h. listen()
22. Static files
    a. public
    b. express.static()

23.API
    a. json()
24. Params, Query String
25. Route Parameter
26. Query string/url Parameter

27.MIddleware
    a. what is middleware
    b. used for what?
    c. req, res, next
    d. next()
    e. app.use in middleware
    f. passing two middleware
    g. external
        i. morgan npm

28. Routing
    a. router
    b. express.Router()

29. Core Express

    a. Session
        i. i express-session
        ii. secret
        iii. resave
        iv. saveUninitialized
        v. destroy()
    b. Cookies
        i. i cookie-parser
    c. Core middleware
    d. Core routing
    e. Build own API
    f. Core views
    g. database integration

30. EJS

    a. i ejs
    b. server side rendering
    c. view engine
    d. render()
    e. <% %>, <%- %>, <%= %>
    f. partials

31. Rest API
    a. RESTful
32. fragment identifier

33. VS

34. API vs HTTP
35. API vs SSR
36. HTTP vs HTTPS
37. URIs vs URLs vs URNs
38. Session vs Cookies
39. GET vs POST
40. PUT vs PATCH
41. SSL vs TLS

42. GOOD TO KNOW

43. Build-in Modules (only imp)
    a. os
    b. path
        i. join()
        ii. basename()
        iii. resolve()
    c. fs
        i. fs sync

   ii. - readFileSync()
   iii. - writeFileSync()
   iv. **fs async**
   v. - readFile()
   vi. - writeFile()

d. http
   i. createServer()
      1. url
      2. listen()
      3. write()
      4. writeHead()
      5. end()

e. util
   i. promisify

f. events
   i. on()

W05: MONGODB

1. **THEORY**
2. What is Database?
3. SQL(relational) vs NoSQL ()
4. What is MongoDB?
5. Run on JS Engine
6. How does mongoDB work?
7. Non-relational Document based
8. Advantage and Disadvantages
9. BSON
10. MongoDB Structure
11. MongoDB architecture
12. JSON vs BSON
13. MongoDB shell
14. CRUD Operations
15. Cursor, Iterate a Cursor
16. Time to Leave
17. Maximum Document Size : 16Mb
    a. GridFS
18. **Data types in MongoDB (BSON)**
    a. ObjectId
        i. timestamp
        ii. random value
        iii. incrementing counter
    b. String
    c. Int, longInt, Double
    d. Array, Object
    e. Boolean
    f. Date
    g. Decimal128
    h. Regex
    i. Javascript
        i. with scope
        ii. without scope
    j. MinKey, MaxKey
    k. Binary data
19. Cursor
    a. cursor methods
    b. - toArray
    c. - forEach
20. **Collection**
    a. db
    b. db.createCollection(collectionName)
    c. show collections
    d. renaming Collection
21. **Documents**
    a. adding new Documents
    b. Nested Documents
        i. advantage
22. **Inserting Document**
23. Insert One and Many
24. what are the additional methods used for inserting
25. **Finding / Querying**
    a. find()
        i. iterate (it)
        ii. pretty()
    b. findOne({ *filter* })
    c. finding In nested Array
        i. "*field.field*"
        ii. match
        iii. exact match
        iv. multiple match
    d. Array
        i. finding in specific order
        ii. without regard to order
        iii. query by array index
        iv. query by array length
    e. **Projection**
        i. explicitly include fields
    f. Null, $type: 10, $exists
26. **Filtering**
    a. find(*filter*)
    b. find(*{filter}, {fieldsToGet}*)
27. **Method Chaining**
    a. count()
    b. limit()
    c. sort(1 or -1)
    d. skip()
28. **Operators** (denoted by $)
    a. {$gt: number} $gte
    b. $lt, $lte
    c. $or $and $not
    d. $in: [1,2,3], $nin: [1,2]
    e. $all
    f. $set, $unset

   g. $elemMatch
   h. $slice
   i. $size
   j. $inc: 1, $inc: -1
   k. $pull, $push
   l. $each [1, 2]
   m. $eq, $ne
   n. $currentDate
   o. $exists
   p. $expr
   q. $rename

29. **Deleting**
   a. deleteOne({ *field:value* })
   b. deleteMany()
   c. remove()
30. **Updating**
   a. updateOne({*whichObject*} , {$set: {*field: value, field: value*} })
   b. updateMany()
   c. replaceOne()
   d. incrementing & decrementing
   e. adding and remove from array
   f. upsert
31. **bulkWrite()**
   a. ordered: false
   b. ordered vs unordered
32. **Commands**
   a. mongosh
   b. db
   c. show dbs
   d. db.stats
33. **Aggregation**
   a. How does it work
   b. advantages
   c. distinct
   d. Aggregate stages
      i. $match
      ii. $group
         1. grouping by
         2. -nested field
         3. -multiple field
      iii. $sort
      iv. $count
      v. - other ways to count
      vi. - client and server side count
      vii. $limit, $skip
      viii. $out
      ix. $project
      x. $lookup
      xi. $unwind
      xii. allowDiskUse: true
   e. “$name” vs ”name”
   f. Accumulators Operators
      i. $sum, $avg, $max, $min
   g. Unary Operators
      i. $type, $lt $gt $or $and $multiply
   h. Aggregation Pipeline
      i. How does aggregation pipeline work?
      ii. memory limit : 100mb
         1. spill to disk
   i. Map Reduce
      i. for what is it used?
      ii. find sum, avg
34. **Indexes**
   a. pros and cons of Indexes
   b. createIndex({ *filed: value* })
   c. options when creating Index
      i. background: true
      ii. unique: true
      iii. name: “*<indexName>*”
   d. getIndex()
   e. dropIndex(), dropIndexes
   f. reIndex()
   g. rename Index
   h. hiding index
   i. Types of Indexes
      i. Single Field Index
      ii. Compound Index
      iii. Multikey Index
      iv. Text Index
      v. Geospatial, Hashed, Clustered Index
35. **Schema**
   a. pros and cons of using schema

b. optional schema

36. **Relationships**
   a. embedding
   b. referencing
   c. one-to-one
   d. one-to-many
   e. one-to-squillions
   f. many-to-many
37. **Replication**
   a. replica set
   b. advantage and disadvantages of replication
   c. primary and secondary nodes
   d. arbiter
   e. Replication lag
   f. operation log (oplog)
   g. types of replication
38. **Sharding**
   a. advantages and disadvantages
   b. **Sharding Architecture**
      i. What is Mongos/Router
      ii. Config Server
   c. **Types of sharding**
      i. Hashed sharding
      ii. Ranged sharding
      iii. Zone Sharding
   d. **Shard key**
      i. shard hotspots
      ii. normal shard key
      iii. hashed shard key
   e. Vertical and horizontal scaling
   f. Zones
   g. mongos
   h. auto balancer
39. **Cluster**
   a. types of cluster
   b. config servers
40. **Data Modeling**
   a. linking vs embedding
41. **Transactions**
   a. ACID Tra nsaction
   b. A- Atomicity
   c. C- Consistency
   d. I - Isolation
   e. D - Durability
42. **VS**
   a. $or vs $in
   b. drop() vs remove()
   c. findAndModify() vs findOneAndUpdate()
   d. Primary key vs secondary key
   e. join vs lookup
   f. dot notation vs nested form
   g. $currentDate vs $$NOW
   h. delete() vs remove()
   i. bulkWrite vs InsertMany
   j. replace vs update
   k. shard vs node vs cluster
   l. Aggregation Pipeline vs Map Reduce
   m. vertical scalability vs horizontal scalability
   n. load balancer vs sharding
   o. odm vs driver
   p. stage operator vs accumulator operator
   q. normal shard key vs hashed shard key
   r. aggregate([$count:"tota"]) vs find({}).count()
   s. replication vs replica set
   t. transaction vs query
   u. scaling up vs scaling down
   v. config servers vs mongos
43. What is a MongoDB driver?
44. **INTERVIEW QUESTION**
45. What to do when your quireing becomes slow?
46. What to do when your files are getting very big?
47. How to condense large volumes of data?
48. How to search for text in MongoDB?
49. How does MongoDB schema changes?
50. **GOOD TO KNOW**
51. Atomicity
52. Type Bracketing
53. Dot Notation

54. Cursor behaviour
55. Aggregation Pipeline
56. Retryable Writes and Reads
57. MongoDB CRUD Concepts
58. B-Tree
59. ACID compliance
60. Mongoose
61. Network Components
    a. load balancer
    b. firewall
62. CAP Theorem
63. Mongo Utilities
    a. mongoexport
    b. mongoimport
    c. mongodump
    d. mongorestore
    e. mongostat
    f. mongotop
    g. mongooplog

OTHERS

1. **SASS**
2. @import "../node_modules/bootstrap/scss/bootstrap";
3. @use & @forward