Convention over configuration Like ng new --default=true, so this make frameworks more easier .

So that’s why there is full stack developer, its start with front end and back end on (for example c++), after that frameworks get better and better , so then start to know there is front end and back ends developer then frameworks more flexible so now it’s possible to have full stack developer.

* Benefits of full stack:
  + Low Frequency for team member-developer.
  + Better achievement for the company.
  + Time estimation more elastic to the client. (Knows all the changes will done on the app).
* NodeJS introduces JavaScript to back end, able to run JavaScript out of the browser .
* First attempt to run JavaScript out of the browser done by Microsoft, Classic asp=>aspx run JavaScript out the browser.

Mongo DB:

1. Open source, document store, NoSQL DB, schema less db
2. Backup-Replica-Clone
   1. Replication primary node (mongod: mogo driver),
      1. Contains two DBs (DataStore and operation log)
      2. App using the primary-> request com to mongodb then do the required operation on the database then write doing on the operation log db.
   2. Secondary node : the same as the primary but its not active .
   3. A heartbeat signal between all nodes
   4. Replication signal between primary and the secondary’s nodes.
   5. This we call it ReplicaSet
   6. Fault tolerance :
      1. Arbiter Node: voting
      2. Regular Node: Primary, Secondary
         1. One fault tolerance (if the primary failed or/and one secondary failed then the second node is chosen as primary) (two secondaries’ nodes)
         2. Tow fault tolerance (three secondary node)
         3. switching from one primary node to another primary node (second node) so there is time between switching is this case mogo driver will buffer the command and send it to DB when the primary is chosen.
      3. Delayed node: (Backup node, disaster node). (Data updated not frequently)
      4. Hidden node: Perform analytics
   7. Replication Read preferences:
      1. Read from primary
      2. Read from secondary
      3. Read from nearest node:
   8. Replication write concerns:
      1. Write Zero: return success immediately
      2. Write One: return success only after successful write to primary
      3. Write Two: return success only after successful write to primary and at least one secondary.
   9. New MongoClient(“mongodb://localhost:27017,localhost:27018,localhost:27019,localhost:27020/meanGeams?w=1&readpreferance=secondary”)
3. Scalability
   1. Glass Shard, shard of glass
   2. Sharding is scall out
   3. Router -> Shard 1 -Shard 2 – Shard 3 (Cluster)
      1. Router decide where the peace of data is on which shard
         1. **A mongo database is a cluster consist of several shard each node could be replica set.**
         2. The document exists one shard
         3. Odd even distributions (on id) Hashing
      2. Add Shard
         1. Add new node
         2. Take data from other nodes then go online then remove the data from the other shards
      3. Remove shard
         1. Transfer data to another nodes
         2. Remove the node
      4. Hashing Function
         1. Hash code for each node
         2. When remove or add shard, we rehash.
         3. Hash functions accept shard key to generate hash
      5. Mongo db Transaction
4. \_id epack , processor , time , counter
5. **Red black structure tree (Search)**
6. **Hashing**