Day 1 - 08 Mar 2017 21:00 -

Init Voting Project

Starting coding for the application. I have a node.js project before, so what I have done is reuse web structure of that project for the new Voting web application.

Delete, modify, cleanCode and something like this.

Show 'Hello World' in a website.

Problem - failed to lookup view

- **Problem**: I face a problem when I try to run my new project for the first time: *Error*: Failed to lookup view "/pages/info" in views directory "/Users/Lenny/Desktop/Interview/BBC/voting/views"_
- **Process**: ...still search the solution via Google. I think it is just a typo error or some stupid mistake. But I can't find out...must solve it today.
- **Solution**: in controller/v1/info.js file, remove '/' before '/pages/info' res.render('pages/info'). I know it must be a typo error...

Create Data Model Schema and Create Model test

I will always create **Model Test** for the models in the database, I have finished the data models for **Vote**, **User**, **Candidate**. Finish models test or leave it to tomorrow.

Have finish model test:

- Candidate Model should be able to save a user to the DB: 22ms
- Candidate Model should not be able to save a non-candidateID User to the DB: 2ms
- User Model should be able to save a user to the DB: 3ms
- User Model should not be able to save a ton-userID User to the DB: 2ms
- User Model should not be able to save a maxVote-lagger-than-3 User to the DB: 0ms
- Vote Model should be able to save a vote to the DB: 2ms
- Vote Model should not be able to save a non-userID vote to the DB: 1ms
- Vote Model should not be able to save a non-candidateID vote to the DB: 1ms

8 passing (80ms)

Day 2 - 09 Mar 2017 19:00 -

voting API

Before start the API design and test, I make more models tests and make sure the database can follow the rules I create.

Have finish 3 more model test:

- Candidate Model should not be able to save same candidateID Candidate to the

- User Model should not be able to save same userID User to the DB: 2ms
- Vote Model should be able to save the same vote to the DB: 2ms

I like TDD development so I can know what I have done and is correct. So the first step is to create API test.

The project is not big but I design a structure for my application. **models** folder to pull the model schema. **api/v1** dir has the API path. **proxy** dir handle the DB operation from HTTP request through API. For me, GET API is the easiest to create and test. So I create a GET votes API Test to verify my app structure that can work fine.

Problem - Call Back && Multi Save

- **Problem**: In my original thought, I think when a user posts a vote, Vote will save a vote, User will save candidate and maxVote, Candidate will add one more vote. When I try to implement it. I find it is very complex when they are lots of calls back. Even I use Promise to solve callback, I still need to find out how to manage these operations.
- **Process**: I think the most important part is digging Promise and see how it handles multi callback. Have tried some implementation but it doesn't work well. Keep trying.
- **Thought**: Do I really need to use a callback to save multi data for a different object? Users don't need to know the valid/invalid vote result in this requirement. Can we update User/Candidate database without callback?
- **Solution**: Might find out tonight.

Create required API - POST vote & GET candidates/ Create API Test to test behaviour as well:

- /api/v1/voting/candidates/ GET should list and return all votes: 62ms
- /api/v1/votes/ GET should list and return all votes: 15ms
- /api/v1/votes/ POST should create a vote for a new vote and return 201: 24ms
- /api/v1/votes/ POST should not create a vote for an invalid vote and return 400: 6ms

Day 3 - 10 Mar 2017 21:00 -

Finish Basic Data Proxy with Test

The first is to finish the problem - call back and multi save. Look some documents about call-back and Promise. I have some idea about how to design this part better. Before implementation, I will make some TDD tests first for basic data proxy.

Basic DB Proxy Test:

- Candidate Model Proxy createCandidate should create a new candidate with 1 vote: 2ms
- Candidate Model Proxy updateVoteCandidate should increase 1 for votes number: 10ms
- Candidate Model Proxy updateValideVoteCandidate should increase 1 for valid votes number: 3ms
- User Model Proxy findUser should return one user if find a user: 11ms
- User Model Proxy createUser should create a new user with 1 maxVote and one candidate: 2ms
- User Model Proxy validVote should return false if maxVote is equal/larger then 3: 3ms
- User Model Proxy validVote should return true if maxVote is less than 3: 4ms
- User Model Proxy updateUser should add the second candidate if maxVote equal 1: 5ms
- User Model Proxy updateUser should add the third candidate if maxVote equal 2: 7ms
- Vote Model Proxy updateCandidate should save a new candidate if can not find candidateID: 3ms
- Vote Model Proxy updateCandidate should return one more vote when finding candidateID: 15ms

Implement Post voting API with multi-save correctly

I have to say the TDD is so powerful and so suitable for web development when It need deal with lots of call-back and Promise in one API. It helps a lot to correct my code.

Have same issue about multi save. I thought the code should be ok but the test failed... Will solve it tomorrow.... I spent lots of time to solve this issue but the solution is **fix the typo**... I know when you can not find the reason and if you think your design is correct, it always is the typo issue!!!!!!!

Day 4 - 11 Mar 2017 9:30 -

Implement Post voting API with multi-save correctly

I am still working on the part as there is a typo bug and block a whole night... I have to say the TDD is so great... Because the database proxy I design is from simple to complex. It helps me a lot to find the problem or wrong design in complicated DB operation proxy. Do no have too much detail.... Most of the job is create Test -> design -> implementation -> test ->modification -> pass. Lots of typo or wrong use of promise. And I should finish proxy and API I need before I move to UI design.

Problem - '/app/v1/votes/ POST should create a vote for update user and candidate with validVote and return 201' failed because it won't add valideVote

- **Problem**: In my original thought, when there is a new vote, it will update user then update candidate.But if a user already has 2 maxVote, it will update maxVote to 3, and then when it update candidate, candidate check updated user first and it will consider it as an invalid user (update user before to 3), so it won't update validVote for a candidate.
- **Process**: Well, my I modify the model, modify the user proxy, modify the voting proxy but I find the logic is not easy to handle. And it makes more test fail...I wanna solve it in another branch with this issue.
- **Thought**: I think a better design can save lots of time. The answer is just in the corner.
- **Solution**: Just update Candidate first, then update the user. I have a good design so it saves lots of time to make this change.

Complicated Voting DB Proxy Test:

- Voting Proxy For modifyUser: modifyUser should return a new user when can't find this userID: 10ms
- Voting Proxy For modifyUser: modifyUser should update user with new candidate: 8ms
- Voting Proxy For modifyCandidate: modifyCandidate should create a new candidate: 11ms
- Voting Proxy For modifyCandidate: modifyCandidate should add vote and validVote for a valid user: 11ms
- Voting Proxy For modifyCandidate: modifyCandidate should add vote without validVote for invalid user (maxVote >= 3): 10ms
- Voting Proxy For multiSave: multiSave should return a new user and a new candidate for a Vote: 18ms
- /API/v1/votes/ POST should create a vote for a new vote with new user and candidate then return 201: 34ms
- /API/v1/votes/ POST should create a vote for update user and candidate with validVote and return 201: 31ms
- /API/v1/votes/ POST should create a vote for update invalid user and candidate without validVote and return 201:

24ms

Simple responsive UI Design

Start the front-end UI design. It will be Use node **ejs** view engine with Jquery/Bootstraps/Ajax framework. Nothing too much to talk about. There will have 2 main folders for UI.

1. **Public**: Some js/CSS I created for better display.
2. **Views**: ejs files for creating HTML. **Partials** folder includes some needed components to reduce duplicated code. **Pages** directory has two main pages, one is index.ejs('/voting/') page for front-end design test. Second one is main.ejs('/voting/main') page. This page is for votes created by users and gets candidate result(countMeUp).

Won't have too much problem. Most of the issue are typo mistake. Website debugs tools in Chrome and FireFox have their own advantages and they help a lot to fix the front-end issue.

- Chrome: Better to find the syntax errors in JS files
- FireFox: Better to debug Restful HTTP request.

Have finished the static *index* page and dynamic AJAX RestRequest pages. I will improve a better UI and then start the algorithms.

Day 5 - 12 Mar 2017 9:30 -

Today is the last day, and I will start working on the big data algorithm. Before start it... I need to create a large number of data. I user gulp script to create two task 'createVotes' and test 'bigData'. I should find a proper way to create data.

The requirement is not **big data** as well. **I am in the wrong direction for 4 days**. I need to modify what I have written. And back to the correct directions.

The details are in the algorithmProcess.md

Well, it is not a pure algorithm challenge, it should be big data Back to big data challenge.

Start big data challenge

I might not have enough time to finish the challenges. Well, Nodejs is an async language. I need to solve call-back all the time. And I face this small issue. Mongoose: Query.prototype.stream() is deprecated in mongoose >= 4.5.0, use Query.prototype.cursor() instead.

Here is some methods that can get results at the beginning:

1. Iterate the votes, create a list for userID with votes times. If the votes time is larger than 3, that vote will be invalid. This is O^2 time, O(1) space algorithm. Iterate votes and users, then do the add operation each time for the candidates.

2. Iterate the votes again. But because the userID is the number in this project. We can create an array as long as the votes number. User userID as an index to insert maxVotes int value (1, 2, 3). So, we just use the index to get the votes times and do the valid add operation for candidates. This is O(1)time but O^2 space algorithms. The array MAX_VALUE = 2^31-1 = 2147483647, it is enough for 1000,0000 votes.

But I am afraid that it might have time-out issue and memory issue. Test the second method first.

Improvement the algorithms.

- Can we do parallel computing? Iterates the votes from start to mid and from end to mid together to save the time?

Problem - Stop big data challenges, I think it is the wrong direction and I can't solve it.

- **Problem**: My test data is 100,000 votes, just 1/10 of the 10,000,000 the scenario one. But only iterate 100,000 votes need more than 1 sec. I have no idea how to do 10,000,000 data iterate.
- **Solution**:

Stop the project right now. I think I might in the wrong direction. Because if is a big data challenges. Why it provides the percentages of candidates?