**Database questions:**

Apis\_bish\_user

1. Why save location but not zipcode + latitude / longitude combination to use for computation? **(In mobile app we take location not zipcode or latitude / longitude)**
2. Why is there both id and user\_id in user table? Shouldnt this be id only? **(Here two table one is django table and other is user created table id is a pk and user\_id is a FK of django table)**
3. What is registration\_id? **(Registration\_id is use for notification here we save device registration id )**

Apis\_businessdetail

1. Is Business\_id column the id from yelp? **(yes)**
2. Why are we saving Category\_name from our data here? Shouldnt this be a table of just business data from yelp? So my assumption is saving yelp\_cat instead of category\_name here.

Apis\_businesslike / Apis\_businessvote / Apis\_votebybusiness / Apis\_votelikeself

1. Please explain to me why we have four tables with basically the same data.
   1. Apis\_businesslike we can get the like information from the Apis\_votelikeself table from like boolean.
   2. Seems to me like only thing missing from votelikeself table is latitude, longitude columns and it will contain all information from four tables combined.
   3. Apis\_votebybusiness is not even being used so why can we not delete this one?
   4. Apis\_votelikeself
      1. What is buss\_pk (confusing naming)
      2. Is id the yelp id? (confusing naming) **(if we change the field name so in mobile app we can’t fetch the data)**
      3. Why do we need the name, location, Business\_display\_address, yelp\_cat(this should be inside business data table) if we have a Business data table, can we not just have the Business id here and get the other information in the api if we needed it? **(in Mobile app coding we need simple text so we add all field if we call data from Business data table so some complication in mobile app coding)**
      4. What is status mean? **(if status is false so record not display admin can change the status from web view)**
      5. Why do we call it Business\_id\_id and not Business\_id? (Same with User\_id\_id) **(we call Business\_id as a FK but acroding to django function the change table name Business\_id\_id same as User\_id\_id)**
      6. Why do we need the Category\_name in this table if we have a Categories table and we have the category\_id in here? Cant we just have the category\_id only?**(because of display in mobile app)**
      7. Why do we have username, userimage column here? Can we just have user\_id and fetch from user table? **(we fetch in python but complicated in mobile app)**
      8. What is the column duplicate do?**(if duplicate is true so we vote in another post which is already created by other user and we not delete if unvoted this post)**

Why do we have api\_hashtable?

1. Is this to save hash values to tables? Please elaborate on possible use case. **(Here we save hash tag value when we use in vote process)**

Apis\_notification\_table **(This table for pop up notification in mobile so all filed are required which is display in notification)**

1. Why not just have user\_id
2. Why is data and time character varying and other tables use timestamp or other format
3. Whats post\_id?
4. Whats userid\_id?
5. Whats postuserid\_id?

Do we use auth\_group / auth\_group\_permission/ auth\_permission / auth\_user / auth\_user\_groups / auth\_user\_user\_permissions? If yes please explain to me how please.**(these table created by dajngo automatically)**

Home\_category\_detail

1. Why does the id start at 56? **(because we delete data not the table)**
2. Why is there no created\_at column? **(because all category added by admin and admin is single person)**
3. I believe users can create new category if not existing. So why do we not have a user\_id column? **(if category not add then new category add in voting process but admin add category without vote in web portal )**

**API** **questions**:

Im not familiar with python / django. I see that the views is where we define the method and its signature. For instance if we have a route for a simple get and we want to add another route to get the same information but for a specific category using an id (type, user\_id, etc) as params. Do you reuse the same get route and switch on whether or not it contains params in the request or do you write a whole new route?

Please explain to me the reason we have three: bishproject vs bishproject1 vs home. Is one used only for accessing admin panel? If we no longer need the admin panel can we remove one?

**(bishproject is main project and bishproject1 is backup code and home is for web view)**

1. user\_ins = get\_object\_or\_404(User, id=User\_id)
2. bishuser\_ins = get\_object\_or\_404(Bish\_user, user\_id=user\_ins)

Why do we have User and Bish\_user? Shouldnt we only have one user table?

**(Because User table created by django and filed are fixed we can’t add or edit field in django table and we need another field so we create two different table**

Line 185 apis/views.py

   if Category\_detail.objects.filter(Category\_name=Category\_name).exists():

     pass

   else:

     Category\_name\_value = Category\_detail(Category\_name=Category\_name)

     Category\_name\_value.save()

*# \_\_\_\_\_\_\_\_\_\_\_\_\_\_check user are block or not\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

   user\_ins = get\_object\_or\_404(User, id=User\_id)

   bishuser\_ins = get\_object\_or\_404(Bish\_user, user\_id=user\_ins)

   if bishuser\_ins.activate == False:

     return JsonResponse({"status":"0", "message":"You are blocked by admin."})

*# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

   else:

Are we checking if user is allowed to make a post? So do we allow them to save a new category then block them from the rest of actions?

**(We check if category name already exists so pass from the code otherwise we save category name as a new category)**

For photos I see that you save the endpoint for the photo. How do we handle multiple photos saved per restaurant or post? Do we just have an array?

**(But from mobile app we attach only one photo so we not use array)**

Business\_vote\_confirm:

What is postuser? **(postuser is the user name who create the post)**

Why do we need phototype if we are just always sending in 1 as boolean? **(if we get phototype = 1 so it’s means we get a photo in base64 format other wise we get photo as a photoname from old post)**

     if postuser == "":

       postuserid = ""

       vote\_instance=""

       postuser = User\_id

     else:

Can you explain these lines? Will we ever not have user information when we hit this endpoint?

Line 269 if user has a post with category but not with this business would old\_post\_self be null or an empty array? What would it be?

Line 272-274 can you explain? We look in BusinessVote to find a post with current category + Vote = true + current user + duplicate = true. What is duplicate mean? Why do we then update the Business vote to false? **(duplicate=True means we vote on a post which is already add like vote on another user’s post and duplicate=False means create a new post for vote)**

Then else in 277

           BusinessVote.objects.filter(Category\_id=category\_ins).filter(Business\_vote=True).filter(

             User\_id=bishuser\_ins).update(Business\_vote=False)

So here we find all businessVote posts with current category and has true for voted flag and then update it to business vote = false? I was assuming this is to replace the currently voted post for the same category but in the UI we ask them to confirm override. How are we checking that here? **(according to doc if we vote the same category in another post so my old vote are removed so we check if we already vote in this category so change vote status as false)**

This route is very long and it just has series of if and else and hard to read. Is it at all possible to separate some functionality or sections into named methods?

In 282 im assuming this is the currently voted new business? But what if they never had a post related to this restaurant?

Lets walk through 282-296 and Im wondering if we can cut this down by cutting down some tables.

Im a little confused on ln 314, we get all Business Vote posts that are not voted then iterate through each and update its total\_vote to count of posts currently voted (true) in its category name. Im assuming that data.id points to the yelp id for the business (when we say id I usually assume its the primary key of that table so can we update this to be more clear on what it is?). So I understood this as find all votes with yelpid then check that business\_vote is true then filter for category name to find count from all users. Does the next line update all posts with the combination with the count from above? Can you walk me through the logic here? **(we update all posts with the combination with the count because in mobile code we need the count in ever post if we send code in another array so it is complicated to handle)**

*# UPDATING TOTAL\_VOTES*

*#  Here generic id points to yelpid im assuming*

           num = BusinessVote.objects.filter(id=data.id).filter(Business\_vote=True).filter(Category\_name=data.Category\_name).count()

           BusinessVote.objects.filter(id=data.id).filter(Category\_name=data.Category\_name).update(total\_vote=num)

         data = VoteLikeforSelf.objects.filter(Business\_vote=False)

         for data in data:

           num = BusinessVote.objects.filter(id=data.id).filter(Business\_vote=True).filter(Category\_name=data.Category\_name).count()

           VoteLikeforSelf.objects.filter(id=data.id).filter(Category\_name=data.Category\_name).update(total\_vote=num)

*# ----------------------------------------------------------------------------------------------*

         data12 = BusinessVote.objects.filter(Business\_vote=True).filter(Category\_name=Category\_name)

         for data12 in data12:

           num = BusinessVote.objects.filter(id=data12.id).filter(Category\_name=data12.Category\_name).filter(Business\_vote=True).count()

           data = BusinessVote.objects.filter(id=data12.id).filter(Category\_name=data12.Category\_name)

           for data in data:

             BusinessVote.objects.filter(id=data.id).filter(Category\_name=data.Category\_name).update(total\_vote=num)

         data12 = VoteLikeforSelf.objects.filter(Business\_vote=True).filter(Category\_name=Category\_name)

         for data12 in data12:

           num = VoteLikeforSelf.objects.filter(id=data12.id).filter(Category\_name=data12.Category\_name).filter(Business\_vote=True).count()

           data = VoteLikeforSelf.objects.filter(id=data12.id).filter(Category\_name=data12.Category\_name)

           for data in data:

             VoteLikeforSelf.objects.filter(id=data.id).filter(Category\_name=data.Category\_name).update(total\_vote=num)

So I want to see if we can cut down on this section also if we decide to cut down on the tables. I want to discuss this with you.

       if BusinessVote.objects.filter(id = Business\_id).filter(Category\_id=category\_ins).filter(Business\_vote=False).filter(userid=postuser).filter(User\_id=postuserid).exists():

         vote\_instance = get\_object\_or\_404(BusinessVote, id=Business\_id, Category\_id=category\_ins,

                           Business\_vote=False, userid=postuser, User\_id=postuserid)

       else:

         vote\_instance = get\_object\_or\_404(BusinessVote, id = Business\_id,Category\_id=category\_ins,Business\_vote=True,userid=postuser, User\_id=postuserid)

 Since we just voted for this combination, will we ever have a case where the Business\_vote = False? **(for old voted post which is create by me because self post not remove just change the vote status)**

     if BusinessDetail.objects.filter(Business\_id=Business\_id).filter(Category\_name=Category\_name).exists():

In my opinion the BusinessDetail should be a single source of truth about the restaurant data and shouldnt know anything about the votes. So I think instead of the voted category\_name, we should just save the yelp\_cat.

So above it sends back JSON saying the vote is complete **(this message if vote is successfully done)**

         return JsonResponse({"status": "1", "message": "You have successfully vote for this category.",

                    "Business\_name": str(Business\_name), "total\_vote": str(total)})

But then here it does another check and you can potentially get this **(this message if already vote on same post)**

     if VoteLikeforSelf.objects.filter(Category\_id=category\_ins).filter(User\_id=bishuser\_ins).filter(Business\_vote=True).filter(Business\_id=business\_ins).exists():

*# pass*

       return JsonResponse({"status": "0", "message": "You are voted on this post."})

     else:

Can you explain the flow here?

What is duplicatepost = postuser point to? **(we check this post is duplicate or new , postuser is a user id which is already post this information)**

What is total\_vote\_gte? I dont see that column **(Here we check if total vote count is grater then equal 2 so notification message send to there mobile)**

      if BusinessVote.objects.filter(id=Business\_id).filter(Category\_name=Category\_name).exists():

*# --------------------------------------------------------------------------------------------*

       data = BusinessVote.objects.filter(Business\_vote=False)

       for data in data:

         num = BusinessVote.objects.filter(id=data.id).filter(Business\_vote=True).filter(Category\_name=data.Category\_name).count()

         BusinessVote.objects.filter(id=data.id).filter(Category\_name=data.Category\_name).update(total\_vote=num)

       data = VoteLikeforSelf.objects.filter(Business\_vote=False)

       for data in data:

         num = VoteLikeforSelf.objects.filter(id=data.id).filter(Business\_vote=True).filter(Category\_name=data.Category\_name).count()

         VoteLikeforSelf.objects.filter(id=data.id).filter(Category\_name=data.Category\_name).update(total\_vote=num)

*# ----------------------------------------------------------------------------------------------*

       if BusinessVote.objects.filter(Business\_vote=True).filter(id=Business\_id).filter(Category\_name=Category\_name).exists():

         num = BusinessVote.objects.filter(id=Business\_id).filter(Category\_name=Category\_name).filter(Business\_vote=True).count()

       else:

         num = BusinessVote.objects.filter(id=Business\_id).filter(Category\_name=Category\_name).filter(Business\_vote=False).count()

       data = BusinessVote.objects.filter(id=Business\_id).filter(Category\_name=Category\_name)

       for data in data:

         BusinessVote.objects.filter(id=data.id).filter(Category\_name=data.Category\_name).update(total\_vote=num)

       if VoteLikeforSelf.objects.filter(Business\_vote=True).filter(id=Business\_id).filter(Category\_name=Category\_name).exists():

         num = VoteLikeforSelf.objects.filter(id=Business\_id).filter(Category\_name=Category\_name).filter(Business\_vote=True).count()

       else:

         num = VoteLikeforSelf.objects.filter(id=Business\_id).filter(Category\_name=Category\_name).filter(Business\_vote=False).count()

       data = VoteLikeforSelf.objects.filter(id=Business\_id).filter(Category\_name=Category\_name)

       for data in data:

         VoteLikeforSelf.objects.filter(id=data.id).filter(Category\_name=data.Category\_name).update(total\_vote=num)

I see this at the end but dont we do something very similar before? I did not look at it too closely but can you explain?

I have many many more questions for this current route but I think that if we redesign the db side after we talk, it will be more clear to understand so I will hold off until then. I feel that we update multiple tables with almost identical information but I want to confirm with you first. Also lets do a walk through here.

Also do we use a column called duplicate to prevent duplicates instead of adding a restriction in the db?

Line 357 I believe the category here should save the yelp cat if the business id does not exist in the business detail table. I dont think that we need to save the voted category to that table.

Business\_like\_confirm ln 657 - 761

I dont see any issues here with using one table instead of four separate. Do you agree?

*# Here we check for category + business vote + user post has a like already*

*#  Guessing if the user unvotes a post this exist will be true*

     if BusinessLike.objects.filter(Category\_id=category\_ins).filter(Business\_vote=business\_vote\_ins).filter(Business\_id=business\_ins).filter(User\_id=bishuser\_ins).filter(Like=get\_like).exists():

*# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

       if vote == False:

*# Guessing this is how we handle unvote by deleting.*

         businesslike\_instance = get\_object\_or\_404(BusinessLike, Category\_id=category\_ins,Business\_vote=business\_vote\_ins,Business\_id=business\_ins,User\_id=bishuser\_ins)

         businesslike\_instance.delete()

         business\_instance = get\_object\_or\_404(BusinessVote, Category\_id=category\_ins,

                             Business\_id=business\_ins,username=username)

         total\_like = business\_instance.total\_like

         total\_like = total\_like - 1

         BusinessVote.objects.filter(Category\_id=category\_ins).filter(Business\_id=business\_ins).filter(username=username).update(total\_like=total\_like)

         VoteLikeforSelf.objects.filter(Category\_id=category\_ins).filter(Business\_id=business\_ins).filter(username=username).update(total\_like=total\_like,like=False)

Lets just use one table so when the user unvotes a post, instead of deleting we can just update the like column to false. Now when the row has both vote as false and like as false, we should delete that row.

Also I dont understand why we do the vote / get\_like as opposite booleans. Cant we just check once if the category + business\_vote + business\_id + user\_id + vote = true exists instead of checking twice in line 700 and line 717. Also since we fill the logo if its already liked, would this case ever happen? If user tapped a filled heart icon, it should just unvote instead of getting an error I assume.

Hash\_value: Since we dont currently use this HashTable can you tell me a possible use case of this method? If it is to store hashed values into tables I think we can use this for audits. But please let me know use case.

Line 822-841 As I brought up above in db questions, I still dont know why we fetch from BusinessLike, BusinessVote, and VoteLikeForSelf separately. Seems like we can just use one table since we have similar information stored in each table anyways.

I dont see us hitting this route, what and how is the notification\_data\_value route used?

I see that we hit notification-value with user\_id, what are we getting back here? Ln858-869 what does this produce?

Overall I thought the naming convention is all over the place and could use a refactor in the db / api. I have some suggestions on the naming and also on the structures for the additional work for phase 2 on the back end side. Would love to put that together after this meeting to be discussed.