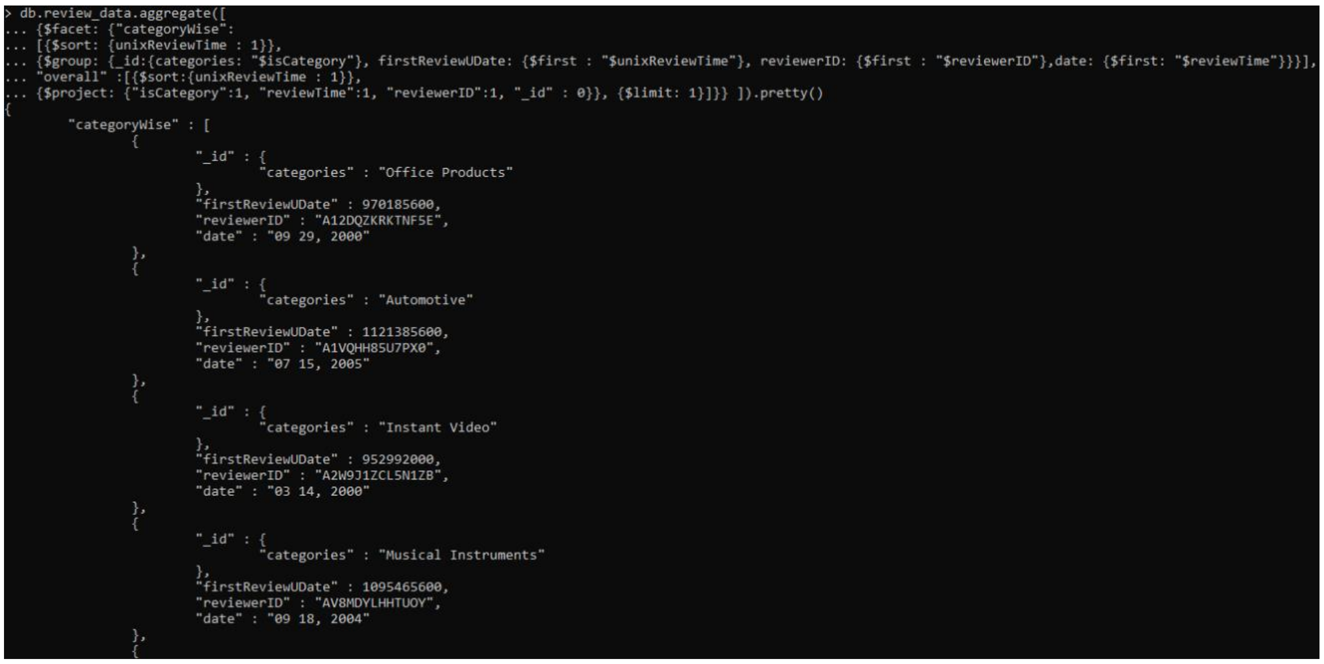
Mongo DB

Mini-Project 4
CIS8045
Unstructured Data Management
Master's in Information Systems Fall 2018
Georgia State University

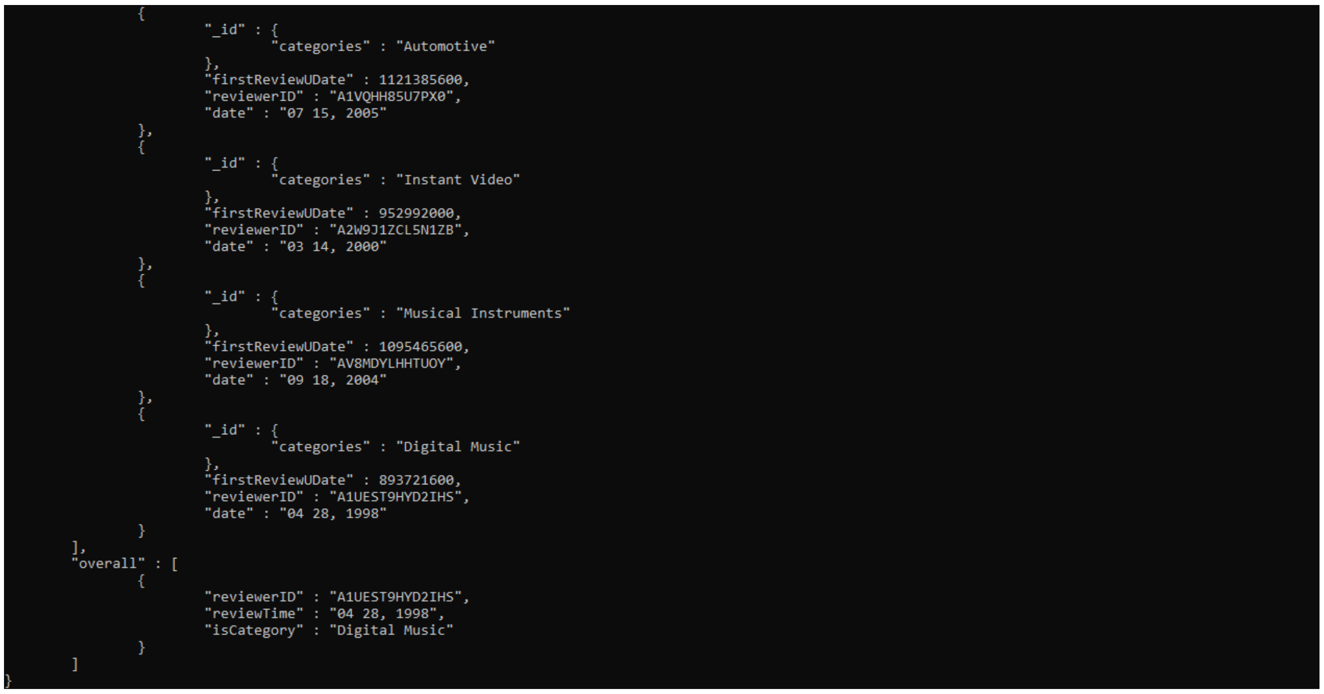
Rahul Sharma
Shruthi Jabshetty
Shrikant Patel
Deepali Pareek
Anubhav Pradhan

Purpose 1:

a- Identify the reviewer with the oldest review overall and within a category.
db.review_data.aggregate([
{$facet: {"categoryWise":
[{$sort: {unixReviewTime : 1}},
{$group: {_id:{categories: "$isCategory"}, firstReviewUDate: {$first : "$unixReviewTime"}, reviewerID: {$first : "$reviewerID"},date: {$first: "$reviewTime"}}}],
"overall" :[{$sort:{unixReviewTime : 1}},
{$project: {"isCategory":1, "reviewTime":1, "reviewerID":1, "_id" : 0}}, {$limit: 1}]}}]).pretty()



```
> db.review_data.aggregate([
... {$facet: {"categoryWise":
... [{$sort: {unixReviewTime : 1}},
... {$group: {_id:{categories: "$isCategory"}, firstReviewUDate: {$first : "$unixReviewTime"}, reviewerID: {$first : "$reviewerID"},date: {$first: "$reviewTime"}}}],
... "overall" :[{$sort:{unixReviewTime : 1}},
... {$project: {"isCategory":1, "reviewTime":1, "reviewerID":1, "_id" : 0}}, {$limit: 1}]}} ]).pretty()
{
        "categoryWise" : [
                {
                        "_id" : {
                                "categories" : "Office Products"
                        },
                        "firstReviewUDate" : 970185600,
                        "reviewerID" : "A12DQZKRKTNF5E",
                        "date" : "09 29, 2000"
                },
                {
                        "_id" : {
                                "categories" : "Automotive"
                        },
                        "firstReviewUDate" : 1121385600,
                        "reviewerID" : "A1VQHH85U7PX0",
                        "date" : "07 15, 2005"
                },
                {
                        "_id" : {
                                "categories" : "Instant Video"
                        },
                        "firstReviewUDate" : 952992000,
                        "reviewerID" : "A2W9J1ZCL5N1ZB",
                        "date" : "03 14, 2000"
                },
                {
                        "_id" : {
                                "categories" : "Musical Instruments"
                        },
                        "firstReviewUDate" : 1095465600,
                        "reviewerID" : "AV8MDYLHHTUOY",
                        "date" : "09 18, 2004"
                },
                {
```

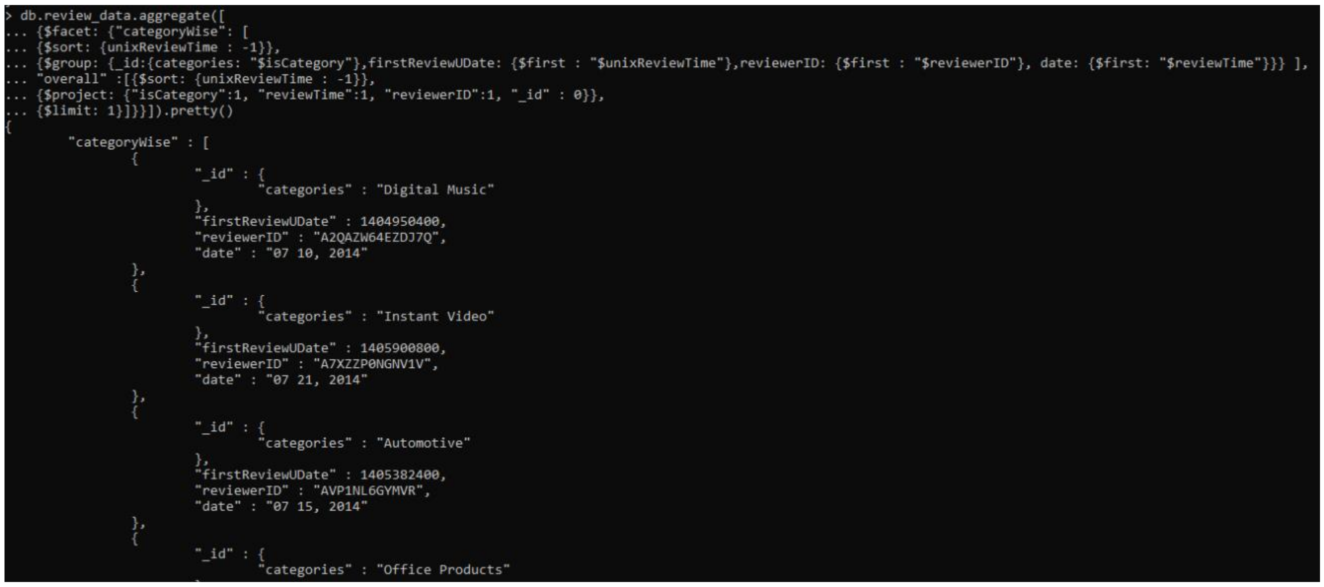


```
                {
                        "_id" : {
                                "categories" : "Automotive"
                        },
                        "firstReviewUDate" : 1121385600,
                        "reviewerID" : "A1VQHH85U7PX0",
                        "date" : "07 15, 2005"
                },
                {
                        "_id" : {
                                "categories" : "Instant Video"
                        },
                        "firstReviewUDate" : 952992000,
                        "reviewerID" : "A2W9J1ZCL5N1ZB",
                        "date" : "03 14, 2000"
                },
                {
                        "_id" : {
                                "categories" : "Musical Instruments"
                        },
                        "firstReviewUDate" : 1095465600,
                        "reviewerID" : "AV8MDYLHHTUOY",
                        "date" : "09 18, 2004"
                },
                {
                        "_id" : {
                                "categories" : "Digital Music"
                        },
                        "firstReviewUDate" : 893721600,
                        "reviewerID" : "A1UEST9HYD2IHS",
                        "date" : "04 28, 1998"
                }
        ],
        "overall" : [
                {
                        "reviewerID" : "A1UEST9HYD2IHS",
                        "reviewTime" : "04 28, 1998",
                        "isCategory" : "Digital Music"
                }
        ]
}
```

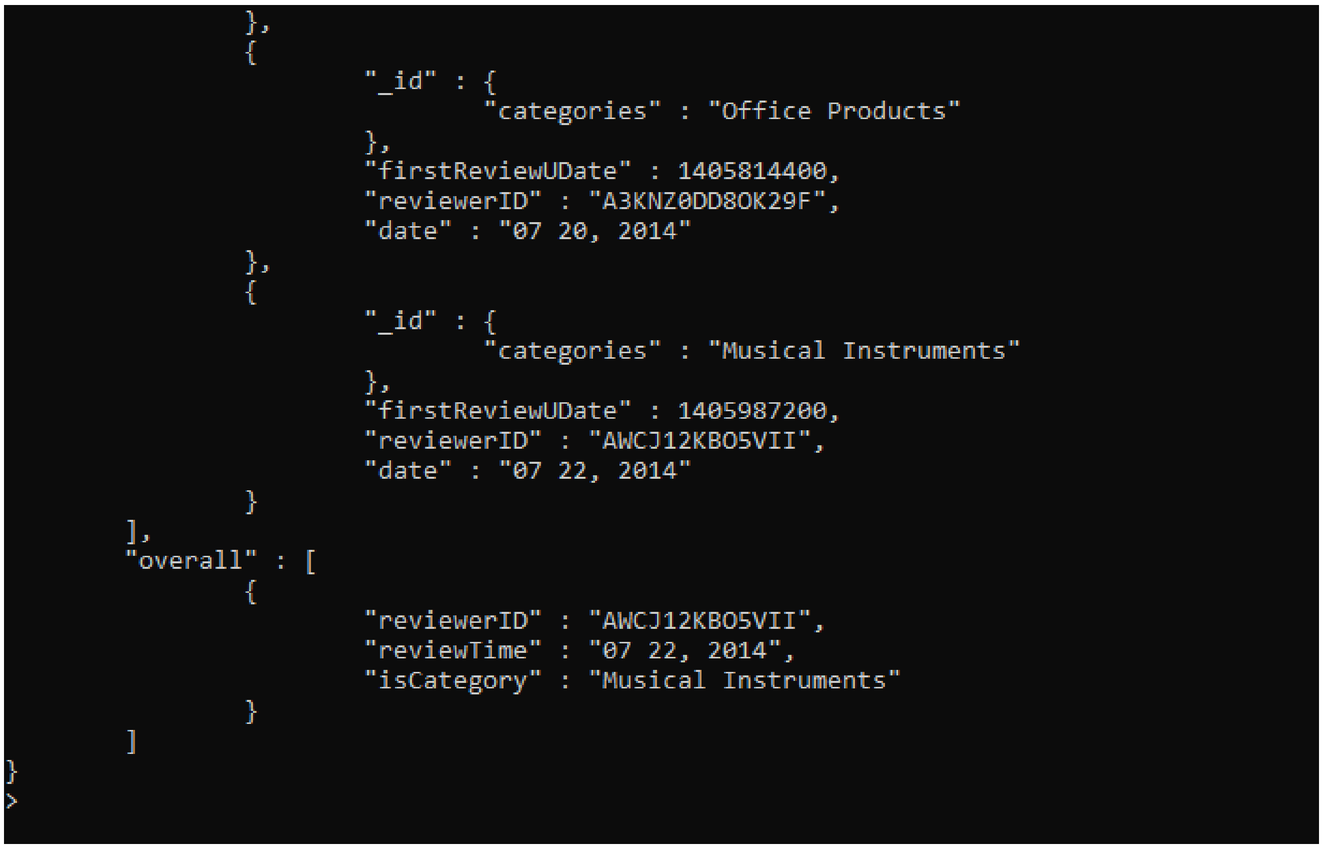
Summary-
The collection is faceted to include multiple aggregation functions (Overall and Category-wise) within a single aggregation stage. To get the oldest reviewer per category. The collection is sorted in the ascending order and grouped by category. ‘$first’ retrieves the first ‘Unix Review Time’, first ‘Reviewer id’ and first ‘Review time’ for each category. To get the oldest reviewer over all, the collection is sorted (ascending order) on the basis of Unix Time and the values of ‘Categories’, ‘Review Time’, ‘Review Id’ are projected with the display limited to just one values each with the help of $limit function. $Pretty helps organize the output.

b- Identify the reviewer with the newest review overall and within a category.

```
db.review_data.aggregate([
{$facet: {"categoryWise": [
{$sort: {unixReviewTime : -1}},
{$group: {_id:{categories: "$isCategory"},firstReviewUDate: {$first : "$unixReviewTime"},reviewerID: {$first : "$reviewerID"}, date: {$first: "$reviewTime"}}} ],
"overall" :[{$sort: {unixReviewTime : -1}},
{$project: {"isCategory":1, "reviewTime":1, "reviewerID":1, "_id" : 0}},
{$limit: 1}]}}]).pretty()
```



```
> db.review_data.aggregate([
... {$facet: {"categoryWise": [
... {$sort: {unixReviewTime : -1}},
... {$group: {_id:{categories: "$isCategory"},firstReviewUDate: {$first : "$unixReviewTime"},reviewerID: {$first : "$reviewerID"}, date: {$first: "$reviewTime"}}} ],
... "overall" :[{$sort: {unixReviewTime : -1}},
... {$project: {"isCategory":1, "reviewTime":1, "reviewerID":1, "_id" : 0}},
... {$limit: 1}]}}]).pretty()
{
        "categoryWise" : [
                {
                        "_id" : {
                                "categories" : "Digital Music"
                        },
                        "firstReviewUDate" : 1404950400,
                        "reviewerID" : "A2QAZW64EZDJ7Q",
                        "date" : "07 10, 2014"
                },
                {
                        "_id" : {
                                "categories" : "Instant Video"
                        },
                        "firstReviewUDate" : 1405900800,
                        "reviewerID" : "A7XZZP0NGNV1V",
                        "date" : "07 21, 2014"
                },
                {
                        "_id" : {
                                "categories" : "Automotive"
                        },
                        "firstReviewUDate" : 1405382400,
                        "reviewerID" : "AVP1NL6GYMVR",
                        "date" : "07 15, 2014"
                },
                {
                        "_id" : {
                                "categories" : "Office Products"
```

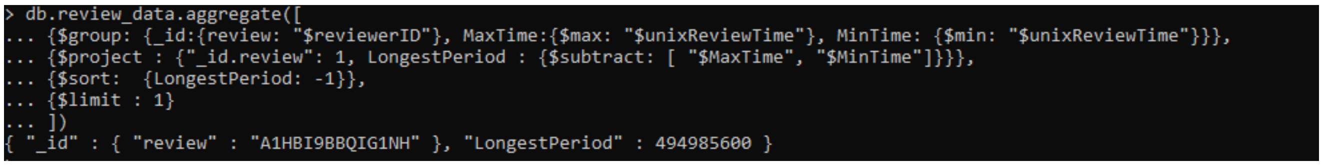


```
                },
                {
                        "_id" : {
                                "categories" : "Office Products"
                        },
                        "firstReviewUDate" : 1405814400,
                        "reviewerID" : "A3KNZ0DD8OK29F",
                        "date" : "07 20, 2014"
                },
                {
                        "_id" : {
                                "categories" : "Musical Instruments"
                        },
                        "firstReviewUDate" : 1405987200,
                        "reviewerID" : "AWCJ12KBO5VII",
                        "date" : "07 22, 2014"
                }
        ],
        "overall" : [
                {
                        "reviewerID" : "AWCJ12KBO5VII",
                        "reviewTime" : "07 22, 2014",
                        "isCategory" : "Musical Instruments"
                }
        ]
}
>
```

Summary-
The collection is faceted to include multiple aggregation functions (Overall and Category-wise) within a single aggregation stage. To get the newest reviewer per category, the collection is sorted in descending order basis Unix review time and grouped by category. ‘$first’ retrieves the first ‘Unix Review Time’, first ‘Reviewer id’ and first ‘Review time’ for each category. To get the newest reviewer overall, the collection is sorted in descending order basis the Unix Time and the values of ‘Categories’, ‘Review Time’, ‘Review Id’ are projected with the display limited to just one values each with the help of ‘limit’ function. $Pretty helps to organize the output.

c- Identify the reviewer who has been reviewing for the longest period of time.

db. review_data.aggregate([
{$group: {_id:{id: "$reviewerID"}, MaxTime:{$max: "$unixReviewTime"}, MinTime: {$min: "$unixReviewTime"}}},
{$group: {_id:{id: "$reviewerID "},LongestPeriod: { $subtract:[{"MaxTime", "MinTime"}] }}},
{$sort: {LongestPeriod: -1}}, {$limit: 1}])

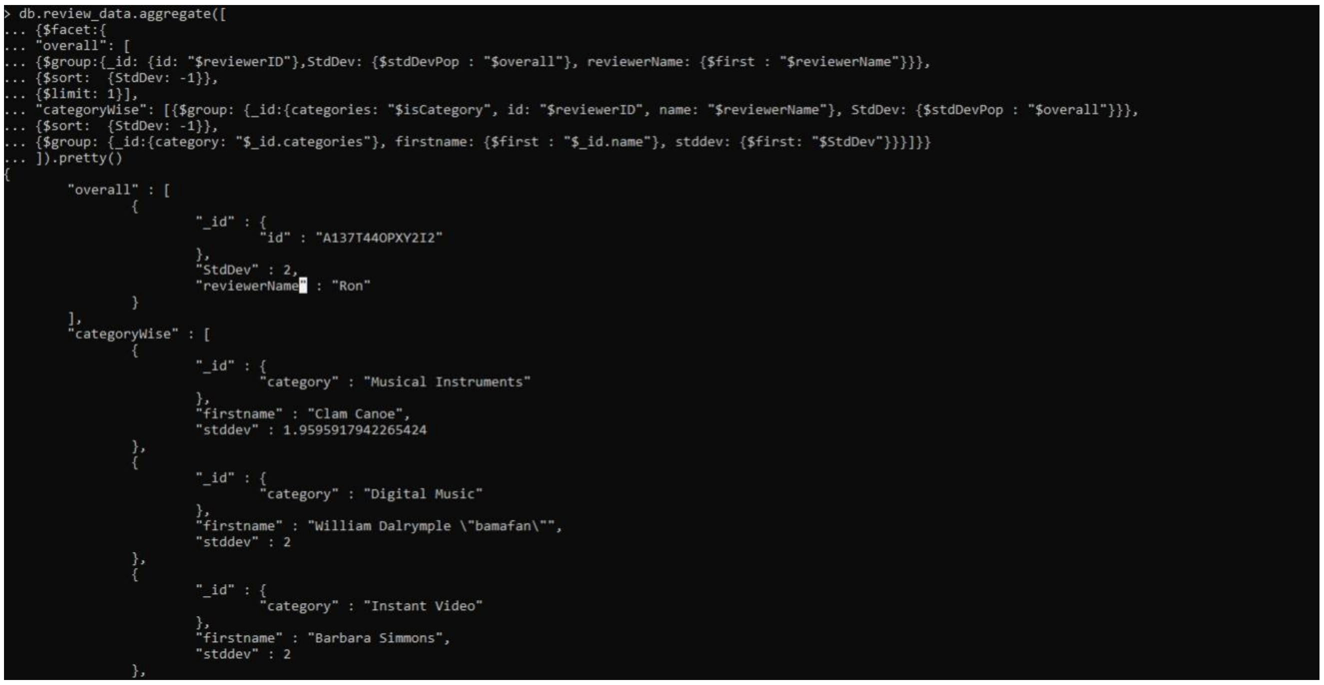


```
> db.review_data.aggregate([
... {$group: {_id:{review: "$reviewerID"}, MaxTime:{$max: "$unixReviewTime"}, MinTime: {$min: "$unixReviewTime"}}},
... {$project : {"_id.review": 1, LongestPeriod : {$subtract: [ "$MaxTime", "$MinTime"]}}}},
... {$sort:  {LongestPeriod: -1}},
... {$limit : 1}
... ])
{ "_id" : { "review" : "A1HBI9BBQIG1NH" }, "LongestPeriod" : 494985600 }
```

Summary-
The documents in the collection is grouped by reviewer id. $max and $min helps retrieve the maximum and the minimum values of Unix review time for each reviewer id. The seconds $group takes the output of the first group as its input and groups the document basis reviewer id. $subtract helps fetch the difference between the minimum and the maximum Unix time for each reviewer. The output is then sorted in descending order and $limits prints the first row with the details of reviewer who has been reviewing for the longest period of time.

d- Identify the reviewer whose reviews deviate the most (largest standard deviation of review rating) overall and within a category.

```
db.review_data.aggregate([
{$facet:{ "overall": [
{$group:{_id: {id: "$reviewerID"},StdDev: {$stdDevPop : "$overall"}, reviewerName: {$first :
"$reviewerName"}}}, {$sort:  {StdDev: -1}}, {$limit: 1}],
"categoryWise": [{$group: {_id:{categories: "$isCategory", id: "$reviewerID", name: "$reviewerName"},
StdDev: {$stdDevPop : "$overall"}}},
{$sort:  {StdDev: -1}},
{$group: {_id:{category: "$_id.categories"}, firstname: {$first : "$_id.name"}, stddev: {$first: "$StdDev"}}}]}}
]).pretty()
```



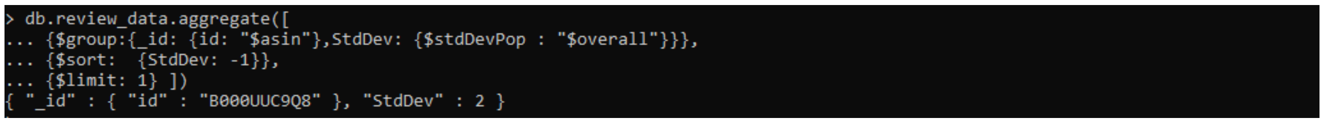
```
> db.review_data.aggregate([
... {$facet:{
... "overall": [
... {$group:{_id: {id: "$reviewerID"},StdDev: {$stdDevPop : "$overall"}, reviewerName: {$first : "$reviewerName"}}},
... {$sort:  {StdDev: -1}},
... {$limit: 1}],
... "categoryWise": [{$group: {_id:{categories: "$isCategory", id: "$reviewerID", name: "$reviewerName"}, StdDev: {$stdDevPop : "$overall"}}},
... {$sort:  {StdDev: -1}},
... {$group: {_id:{category: "$_id.categories"}, firstname: {$first : "$_id.name"}, stddev: {$first: "$StdDev"}}}]}}
... ]).pretty()
{
        "overall" : [
                {
                        "_id" : {
                                "id" : "A137T44OPXY2I2"
                        },
                        "StdDev" : 2,
                        "reviewerName" : "Ron"
                }
        ],
        "categoryWise" : [
                {
                        "_id" : {
                                "category" : "Musical Instruments"
                        },
                        "firstname" : "Clam Canoe",
                        "stddev" : 1.9595917942265424
                },
                {
                        "_id" : {
                                "category" : "Digital Music"
                        },
                        "firstname" : "William Dalrymple \"bamafan\"",
                        "stddev" : 2
                },
                {
                        "_id" : {
                                "category" : "Instant Video"
                        },
                        "firstname" : "Barbara Simmons",
                        "stddev" : 2
                },
```

Summary-
The collection is faceted to include multiple aggregation functions (Overall and Category-wise) within a single aggregation stage. The collection is grouped by reviewer Id. Standard deviation function helps calculate the standard deviation in the ratings for each user. $first returns the first element for each reviewer id to get the reviewer who has the highest standard deviation. $limit is added to get the reviewer with the highest

standard deviation. The second aggregation is used to find the highest standard deviation in ratings per category. Here, the collection is grouped on the basis of category, $stdDevPop calculates the standard deviation in the rating. The collection is then sorted in the descending order on the basis of standard deviation and then grouped by category. $first returns the first element per category and pretty helps organize the output.

e- Identify the product with the most divergence in the ratings of reviews received (largest deviation of rating).

db.review_data.aggregate([
{$group:{_id: {id: "$asin"}, StdDev: {$stdDevPop : "$overall"}}},
{$sort: {StdDev: -1}}, {$limit: 1}])

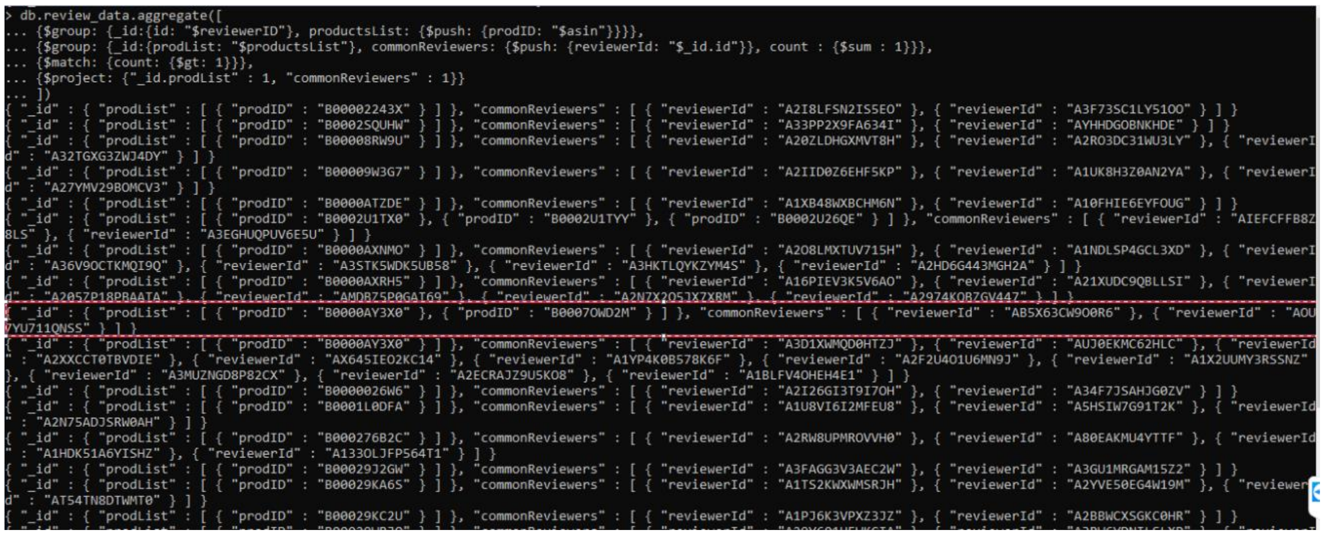


```
> db.review_data.aggregate([
... {$group:{_id: {id: "$asin"},StdDev: {$stdDevPop : "$overall"}}},
... {$sort:  {StdDev: -1}},
... {$limit: 1} ])
{ "_id" : { "id" : "B000UUC9Q8" }, "StdDev" : 2 }
```

Summary-
The collection is grouped on the basis of product id, and standard deviation is applied on the grouped ratings. The results are then sorted on the basis of standard deviation in descending order with the limit one to display just the product with largest deviation

f- Identify the reviewers that reviewed a common set of products.

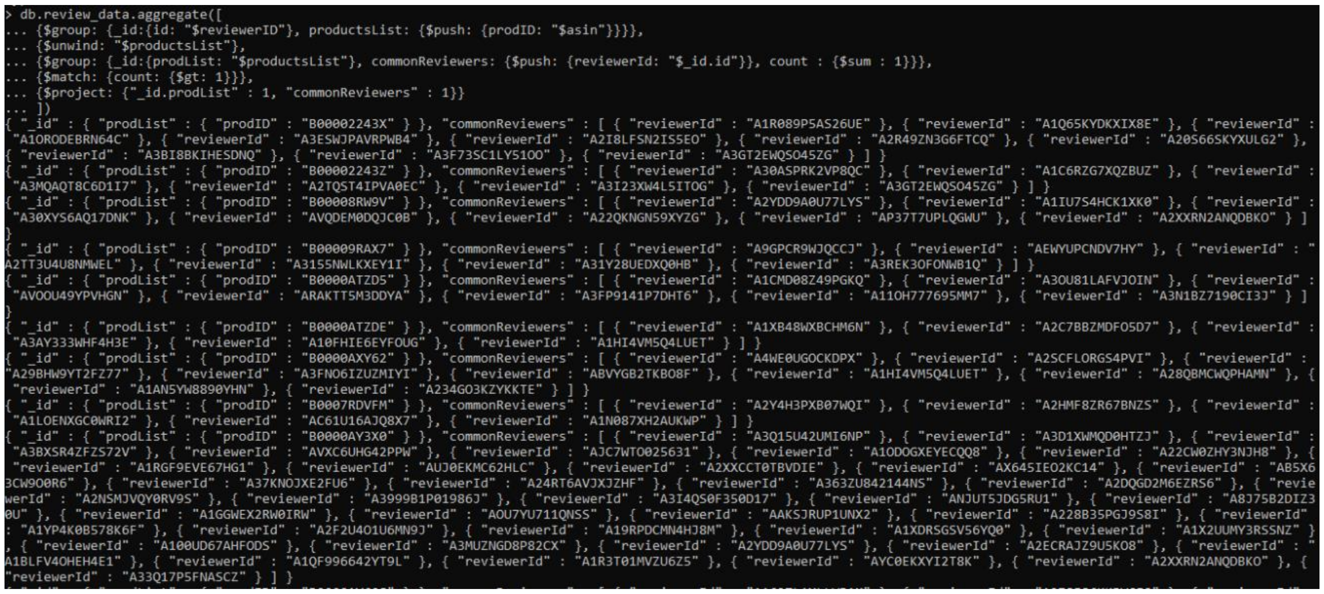
db.review_data.aggregate([
{$group: {_id:{id: "$reviewerID"}, productsList: {$push: {prodID: "$asin"}}}},
{$group: {_id:{prodList: "$productsList"}, commonReviewers: {$push: {reviewerId: "$_id.id"}}, count : {$sum : 1}}},
{$match: {count: {$gt: 1}}},
{$project: {"_id.prodList" : 1, "commonReviewers" : 1}}
])



```
> db.review_data.aggregate([
... {$group: {_id:{id: "$reviewerID"}, productsList: {$push: {prodID: "$asin"}}}},
... {$group: {_id:{prodList: "$productsList"}, commonReviewers: {$push: {reviewerId: "$_id.id"}}, count : {$sum : 1}}},
... {$match: {count: {$gt: 1}}},
... {$project: {"_id.prodList" : 1, "commonReviewers" : 1}}
... ])
{ "_id" : { "prodList" : [ { "prodID" : "B00002243X" } ] }, "commonReviewers" : [ { "reviewerId" : "A2I8LFSN2IS5EO" }, { "reviewerId" : "A3F73SC1LY51OO" } ] }
{ "_id" : { "prodList" : [ { "prodID" : "B0002SQUHW" } ] }, "commonReviewers" : [ { "reviewerId" : "A33PP2X9FA634I" }, { "reviewerId" : "AYHHDGOBNKHDE" } ] }
{ "_id" : { "prodList" : [ { "prodID" : "B00008RW9U" } ] }, "commonReviewers" : [ { "reviewerId" : "A20ZLDHGXMVT8H" }, { "reviewerId" : "A2RO3DC31WU3LY" }, { "reviewerI
d" : "A32TGXG3ZWJ4DY" } ] }
{ "_id" : { "prodList" : [ { "prodID" : "B00009W3G7" } ] }, "commonReviewers" : [ { "reviewerId" : "A2IID0Z6EHF5KP" }, { "reviewerId" : "A1UK8H3Z0AN2YA" }, { "reviewerI
d" : "A27YMV29BOMCV3" } ] }
{ "_id" : { "prodList" : [ { "prodID" : "B0000ATZDE" } ] }, "commonReviewers" : [ { "reviewerId" : "A1XB48WXBCHM6N" }, { "reviewerId" : "A10FHIE6EYFOUG" } ] }
{ "_id" : { "prodList" : [ { "prodID" : "B0002U1TX0" }, { "prodID" : "B0002U1TYY" }, { "prodID" : "B0002U26QE" } ] }, "commonReviewers" : [ { "reviewerId" : "AIEFCFFB8Z
8LS" }, { "reviewerId" : "A3EGHUQPUV6E5U" } ] }
{ "_id" : { "prodList" : [ { "prodID" : "B0000AXNMO" } ] }, "commonReviewers" : [ { "reviewerId" : "A2O8LMXTUV715H" }, { "reviewerId" : "A1NDLSP4GCL3XD" }, { "reviewerI
d" : "A36V9OCTKMQI9Q" }, { "reviewerId" : "A3STK5WDK5UB58" }, { "reviewerId" : "A3HKTLQYKZYM4S" }, { "reviewerId" : "A2HD6G443MGH2A" } ] }
{ "_id" : { "prodList" : [ { "prodID" : "B0000AXRH5" } ] }, "commonReviewers" : [ { "reviewerId" : "A16PIEV3K5V6AO" }, { "reviewerId" : "A21XUDC9QBLLSI" }, { "reviewerI
d" : "A205ZP18PBAATA" }, { "reviewerId" : "AMDBZ5P0GAI69" }, { "reviewerId" : "A2N7X2O5JX7XRM" }, { "reviewerId" : "A2974KOBZGV447" } ] }
{ "_id" : { "prodList" : [ { "prodID" : "B0000AY3X0" }, { "prodID" : "B0007OWD2M" } ] }, "commonReviewers" : [ { "reviewerId" : "AB5X63CW9O0R6" }, { "reviewerId" : "AOU
7YU711QNSS" } ] }
{ "_id" : { "prodList" : [ { "prodID" : "B0000AY3X0" } ] }, "commonReviewers" : [ { "reviewerId" : "A3D1XWMQD0HTZJ" }, { "reviewerId" : "AUJ0EKMC62HLC" }, { "reviewerId
" : "A2XXCCT0TBVDIE" }, { "reviewerId" : "AX645IEO2KC14" }, { "reviewerId" : "A1YP4K0B578K6F" }, { "reviewerId" : "A2F2U4O1U6MN9J" }, { "reviewerId" : "A1X2UUMY3RSSNZ"
}, { "reviewerId" : "A3MUZNGD8P82CX" }, { "reviewerId" : "A2ECRAJZ9U5KO8" }, { "reviewerId" : "A1BLFV4OHEH4E1" } ] }
{ "_id" : { "prodList" : [ { "prodID" : "B0000026W6" } ] }, "commonReviewers" : [ { "reviewerId" : "A2I26GI3T9I7OH" }, { "reviewerId" : "A34F7JSAHJG0ZV" } ] }
{ "_id" : { "prodList" : [ { "prodID" : "B0001L0DFA" } ] }, "commonReviewers" : [ { "reviewerId" : "A1U8VI6I2MFEU8" }, { "reviewerId" : "A5HSIW7G91T2K" }, { "reviewerId
" : "A2N75ADJSRW0AH" } ] }
{ "_id" : { "prodList" : [ { "prodID" : "B000276B2C" } ] }, "commonReviewers" : [ { "reviewerId" : "A2RW8UPMROVVH0" }, { "reviewerId" : "A80EAKMU4YTTF" }, { "reviewerId
" : "A1HDK51A6YISHZ" }, { "reviewerId" : "A133OLJFP564T1" } ] }
{ "_id" : { "prodList" : [ { "prodID" : "B00029J2GW" } ] }, "commonReviewers" : [ { "reviewerId" : "A3FAGG3V3AEC2W" }, { "reviewerId" : "A3GU1MRGAM15Z2" } ] }
{ "_id" : { "prodList" : [ { "prodID" : "B00029KA6S" } ] }, "commonReviewers" : [ { "reviewerId" : "A1TS2KWXWMSRJH" }, { "reviewerId" : "A2YVE50EG4W19M" }, { "reviewer
d" : "AT54TN8DTWMT0" } ] }
{ "_id" : { "prodList" : [ { "prodID" : "B00029KC2U" } ] }, "commonReviewers" : [ { "reviewerId" : "A1PJ6K3VPXZ3JZ" }, { "reviewerId" : "A2BBWCXSGKC0HR" } ] }
```

Summary-
The collection is first grouped on the basis of reviewer ID, $push is used to form a list of product Ids reviewed by each reviewer. The result is then grouped on the basis of Product Id list (formed in the first group by) and a new array of reviewer Id who reviewed the same list of products is formed. Count is used to get the number of common reviewers. $match is used to avoid the condition where there are no common reviewers. We then project the product list and the common reviews. This query provides the common reviewers for a given set of products and to find the list of common reviewers per product we can use the below query.

```
db.review_data.aggregate([
{$group: {_id:{id: "$reviewerID"}, productsList: {$push: {prodID: "$asin"}}}},
{$unwind: "$productsList"},
{$group: {_id:{prodList: "$productsList"}, commonReviewers: {$push: {reviewerId: "$_id.id"}}, count : {$sum : 1}}},
{$match: {count: {$gt: 1}}},
{$project: {"_id.prodList" : 1, "commonReviewers" : 1}}
])
```

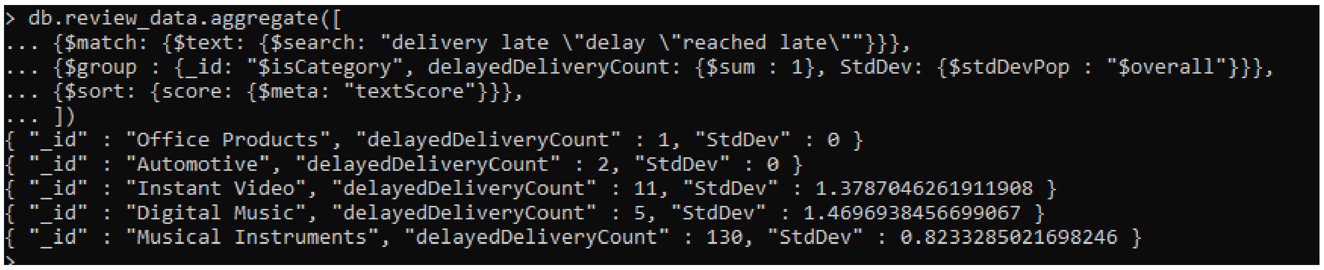


```
> db.review_data.aggregate([
... {$group: {_id:{id: "$reviewerID"}, productsList: {$push: {prodID: "$asin"}}}},
... {$unwind: "$productsList"},
... {$group: {_id:{prodList: "$productsList"}, commonReviewers: {$push: {reviewerId: "$_id.id"}}, count : {$sum : 1}}},
... {$match: {count: {$gt: 1}}},
... {$project: {"_id.prodList" : 1, "commonReviewers" : 1}}
... ])
{ "_id" : { "prodList" : { "prodID" : "B00002243X" } }, "commonReviewers" : [ { "reviewerId" : "A1R089P5AS26UE" }, { "reviewerId" : "A1Q65KYDKXIX8E" }, { "reviewerId" :
 "A1ORODEBRN64C" }, { "reviewerId" : "A3ESWJPAVRPWB4" }, { "reviewerId" : "A2I8LFSN2IS5EO" }, { "reviewerId" : "A2R49ZN3G6FTCQ" }, { "reviewerId" : "A20S66SKYXULG2" },
{ "reviewerId" : "A3BI8BKIHESDNQ" }, { "reviewerId" : "A3F73SC1LY51OO" }, { "reviewerId" : "A3GT2EWQSO45ZG" } ] }
{ "_id" : { "prodList" : { "prodID" : "B00002243Z" } }, "commonReviewers" : [ { "reviewerId" : "A30ASPRK2VP8QC" }, { "reviewerId" : "A1C6RZG7XQZBUZ" }, { "reviewerId" :
 "A3MQAQT8C6D1I7" }, { "reviewerId" : "A2TQST4IPVA0EC" }, { "reviewerId" : "A3I23XW4L5ITOG" }, { "reviewerId" : "A3GT2EWQSO45ZG" } ] }
{ "_id" : { "prodList" : { "prodID" : "B00008RW9V" } }, "commonReviewers" : [ { "reviewerId" : "A2YDD9A0U77LYS" }, { "reviewerId" : "A1IU7S4HCK1XK0" }, { "reviewerId" :
 "A30XYS6AQ17DNK" }, { "reviewerId" : "AVQDEM0DQJC0B" }, { "reviewerId" : "A22QKNGN59XYZG" }, { "reviewerId" : "AP37T7UPLQGWU" }, { "reviewerId" : "A2XXRN2ANQDBKO" } ]
}
{ "_id" : { "prodList" : { "prodID" : "B00009RAX7" } }, "commonReviewers" : [ { "reviewerId" : "A9GPCR9WJQCCJ" }, { "reviewerId" : "AEWYUPCNDV7HY" }, { "reviewerId" : "
A2TT3U4U8NMWEL" }, { "reviewerId" : "A3155NWLKXEY1I" }, { "reviewerId" : "A31Y28UEDXQ0HB" }, { "reviewerId" : "A3REK3OFONWB1Q" } ] }
{ "_id" : { "prodList" : { "prodID" : "B0000ATZD5" } }, "commonReviewers" : [ { "reviewerId" : "A1CMD08Z49PGKQ" }, { "reviewerId" : "A3OU81LAFVJOIN" }, { "reviewerId" :
 "AVOOU49YPVHGN" }, { "reviewerId" : "ARAKTT5M3DDYA" }, { "reviewerId" : "A3FP9141P7DHT6" }, { "reviewerId" : "A11OH777695MM7" }, { "reviewerId" : "A3N1BZ7190CI3J" } ]
}
{ "_id" : { "prodList" : { "prodID" : "B0000ATZDE" } }, "commonReviewers" : [ { "reviewerId" : "A1XB48WXBCHM6N" }, { "reviewerId" : "A2C7BBZMDFO5D7" }, { "reviewerId" :
 "A3AY333WHF4H3E" }, { "reviewerId" : "A10FHIE6EYFOUG" }, { "reviewerId" : "A1HI4VM5Q4LUET" } ] }
{ "_id" : { "prodList" : { "prodID" : "B0000AXY62" } }, "commonReviewers" : [ { "reviewerId" : "A4WE0UGOCKDPX" }, { "reviewerId" : "A2SCFLORGS4PVI" }, { "reviewerId" :
"A29BHW9YT2FZ77" }, { "reviewerId" : "A3FNO6IZUZMIYI" }, { "reviewerId" : "ABVYGB2TKBO8F" }, { "reviewerId" : "A1HI4VM5Q4LUET" }, { "reviewerId" : "A28QBMCWQPHAMN" }, {
 "reviewerId" : "A1AN5YW8890YHN" }, { "reviewerId" : "A234GO3KZYKKTE" } ] }
{ "_id" : { "prodList" : { "prodID" : "B0007RDVFM" } }, "commonReviewers" : [ { "reviewerId" : "A2Y4H3PXB07WQI" }, { "reviewerId" : "A2HMF8ZR67BNZS" }, { "reviewerId" :
 "A1LOENXGC0WRI2" }, { "reviewerId" : "AC61U16AJQ8X7" }, { "reviewerId" : "A1N087XH2AUKWP" } ] }
{ "_id" : { "prodList" : { "prodID" : "B0000AY3X0" } }, "commonReviewers" : [ { "reviewerId" : "A3Q15U42UMI6NP" }, { "reviewerId" : "A3D1XWMQD0HTZJ" }, { "reviewerId" :
 "A3BXSR4ZFZS72V" }, { "reviewerId" : "AVXC6UHG42PPW" }, { "reviewerId" : "AJC7WTO025631" }, { "reviewerId" : "A1ODOGXEYECQQ8" }, { "reviewerId" : "A22CW0ZHY3NJH8" }, {
 "reviewerId" : "A1RGF9EVE67HG1" }, { "reviewerId" : "AUJ0EKMC62HLC" }, { "reviewerId" : "A2XXCCT0TBVDIE" }, { "reviewerId" : "AX645IEO2KC14" }, { "reviewerId" : "AB5X6
3CW9O0R6" }, { "reviewerId" : "A37KNOJXE2FU6" }, { "reviewerId" : "A24RT6AVJXJZHF" }, { "reviewerId" : "A363ZU842144NS" }, { "reviewerId" : "A2DQGD2M6EZRS6" }, { "revie
werId" : "A2NSMJVQY0RV9S" }, { "reviewerId" : "A3999B1P01986J" }, { "reviewerId" : "A3I4QS0F350D17" }, { "reviewerId" : "ANJUT5JDG5RU1" }, { "reviewerId" : "A8J75B2DIZ3
0U" }, { "reviewerId" : "A1GGWEX2RW0IRW" }, { "reviewerId" : "AOU7YU711QNSS" }, { "reviewerId" : "AAKSJRUP1UNX2" }, { "reviewerId" : "A228B35PGJ9S8I" }, { "reviewerId"
: "A1YP4K0B578K6F" }, { "reviewerId" : "A2F2U4O1U6MN9J" }, { "reviewerId" : "A19RPDCMN4HJ8M" }, { "reviewerId" : "A1XDRSGSV56YQ0" }, { "reviewerId" : "A1X2UUMY3RSSNZ" }
, { "reviewerId" : "A100UD67AHFODS" }, { "reviewerId" : "A3MUZNGD8P82CX" }, { "reviewerId" : "A2YDD9A0U77LYS" }, { "reviewerId" : "A2ECRAJZ9U5KO8" }, { "reviewerId" : "
A1BLFV4OHEH4E1" }, { "reviewerId" : "A1QF996642YT9L" }, { "reviewerId" : "A1R3T01MVZU6Z5" }, { "reviewerId" : "AYC0EKXYI2T8K" }, { "reviewerId" : "A2XXRN2ANQDBKO" }, {
"reviewerId" : "A33Q17P5FNASCZ" } ] }
```

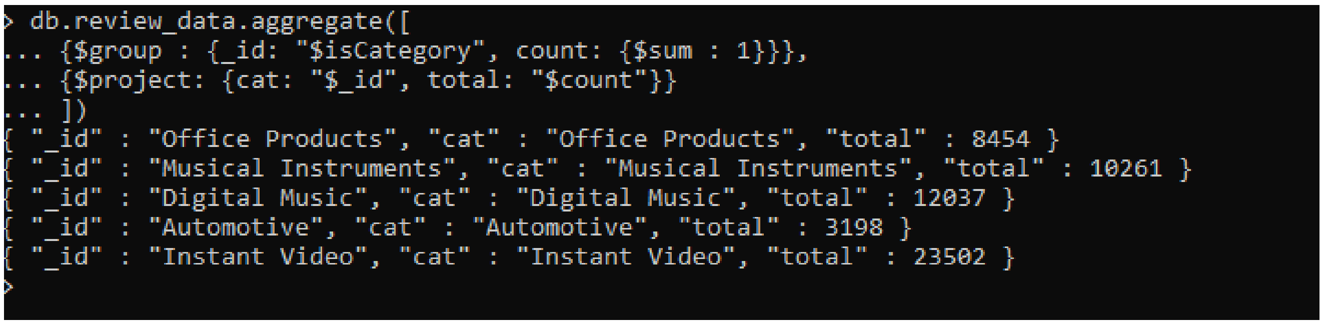
Summary-
Here, $unwind is used to find the common set of reviewers for a single product id.

g) Find the number of products per category (as per their review text) that faced delivery related issues

```
db.review_data.aggregate([
{$match: {$text: {$search: "delivery late \"delay \"reached late\""}}},
{$group : {_id: "$isCategory", delayedDeliveryCount: {$sum : 1}}},
{$sort: {score: {$meta: "textScore"}}},
])
db.review_data.aggregate([
{$group : {_id: "$isCategory", count: {$sum : 1}}},
{$project: {cat: "$_id", total: "$count"}}
])
```



```
> db.review_data.aggregate([
... {$match: {$text: {$search: "delivery late \"delay \"reached late\""}}},
... {$group : {_id: "$isCategory", delayedDeliveryCount: {$sum : 1}, StdDev: {$stdDevPop : "$overall"}}},
... {$sort: {score: {$meta: "textScore"}}},
... ])
{ "_id" : "Office Products", "delayedDeliveryCount" : 1, "StdDev" : 0 }
{ "_id" : "Automotive", "delayedDeliveryCount" : 2, "StdDev" : 0 }
{ "_id" : "Instant Video", "delayedDeliveryCount" : 11, "StdDev" : 1.3787046261911908 }
{ "_id" : "Digital Music", "delayedDeliveryCount" : 5, "StdDev" : 1.4696938456699067 }
{ "_id" : "Musical Instruments", "delayedDeliveryCount" : 130, "StdDev" : 0.8233285021698246 }
>
```



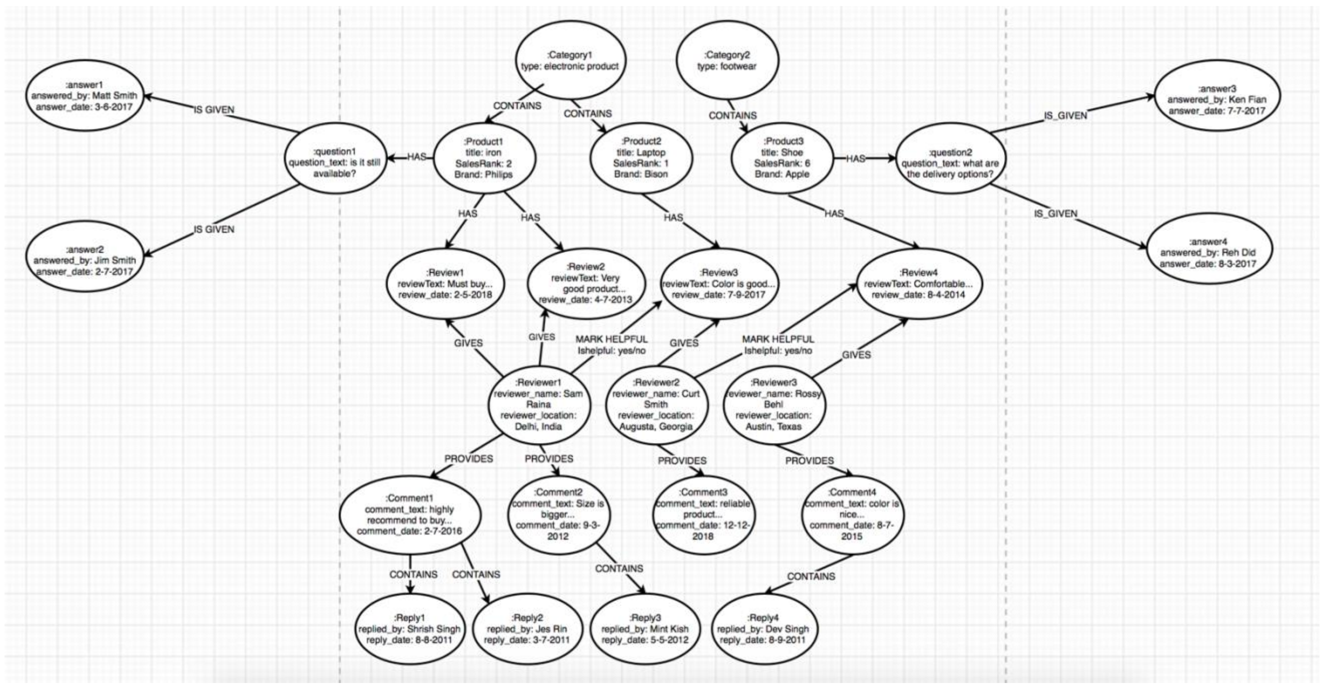
```
> db.review_data.aggregate([
... {$group : {_id: "$isCategory", count: {$sum : 1}}},
... {$project: {cat: "$_id", total: "$count"}}
... ])
{ "_id" : "Office Products", "cat" : "Office Products", "total" : 8454 }
{ "_id" : "Musical Instruments", "cat" : "Musical Instruments", "total" : 10261 }
{ "_id" : "Digital Music", "cat" : "Digital Music", "total" : 12037 }
{ "_id" : "Automotive", "cat" : "Automotive", "total" : 3198 }
{ "_id" : "Instant Video", "cat" : "Instant Video", "total" : 23502 }
>
```

Summary-

The first query aggregates the entire collections and matches for strings like (“delivery late /"delay /"reached late”) and tries to find reviews where user has expressed displeasure for the delivery related issues and the standard deviation in their ratings. The second query gives us the total number of reviews per category. Combined output of both the queries can be used to find customers who are facing delivery related issues.

Purpose 2:

The Graph Model-


:Category1 type: electronic product
:Category2 type: footwear
:answer1 answered_by: Matt Smith answer_date: 3-6-2017
IS GIVEN
:question1 question_text: is it still available?
HAS
:Product1 title: iron SalesRank: 2 Brand: Philips
CONTAINS
CONTAINS
:Product2 title: Laptop SalesRank: 1 Brand: Bison
CONTAINS
:Product3 title: Shoe SalesRank: 6 Brand: Apple
HAS
:question2 question_text: what are the delivery options?
IS_GIVEN
:answer3 answered_by: Ken Fian answer_date: 7-7-2017
IS GIVEN
:answer2 answered_by: Jim Smith answer_date: 2-7-2017
HAS
HAS
HAS
HAS
IS_GIVEN
:answer4 answered_by: Reh Did answer_date: 8-3-2017
:Review1 reviewText: Must buy... review_date: 2-5-2018
:Review2 reviewText: Very good product... review_date: 4-7-2013
:Review3 reviewText: Color is good... review_date: 7-9-2017
:Review4 reviewText: Comfortable... review_date: 8-4-2014
GIVES
GIVES
MARK HELPFUL Ishelpful: yes/no
GIVES
MARK HELPFUL Ishelpful: yes/no
GIVES
:Reviewer1 reviewer_name: Sam Raina reviewer_location: Delhi, India
:Reviewer2 reviewer_name: Curt Smith reviewer_location: Augusta, Georgia
:Reviewer3 reviewer_name: Rossy Behl reviewer_location: Austin, Texas
PROVIDES
PROVIDES
PROVIDES
PROVIDES
:Comment1 comment_text: highly recommend to buy... comment_date: 2-7-2016
:Comment2 comment_text: Size is bigger... comment_date: 9-3-2012
:Comment3 comment_text: reliable product... comment_date: 12-12-2018
:Comment4 comment_text: color is nice... comment_date: 8-7-2015
CONTAINS
CONTAINS
CONTAINS
CONTAINS
:Reply1 replied_by: Shrish Singh reply_date: 8-8-2011
:Reply2 replied_by: Jes Rin reply_date: 3-7-2011
:Reply3 replied_by: Mint Kish reply_date: 5-5-2012
:Reply4 replied_by: Dev Singh reply_date: 8-9-2011

Graph data model is used to visualize the data sets in a graphical manner. We use nodes and relationships to find connection between the data and use a simple and fast retrieval approach for a better understanding. The approach allows better problem solving as we can easily traverse through the nodes. The properties can reside inside the nodes or the relationships.

A graph model consists of-

Nodes- Nodes are usually used to represent the entities and they may contain the properties in the form of name-value pairs of data. They also contain the key values of the properties. In the above graph, nodes like Product, category and reviewers give us the entities and contain their respective properties

Relationships- Relationships connect two related nodes. A relationship has a source node and a target node that shows the direction with the connecting arrows. For example, in the above graph, a category has multiple Products, where the relationship is represented between Category and Product.

Properties- These are the name-value pairs that can be stored on a node or a relationship. Properties allow to store relevant data about the node or relationship on which they are stored.

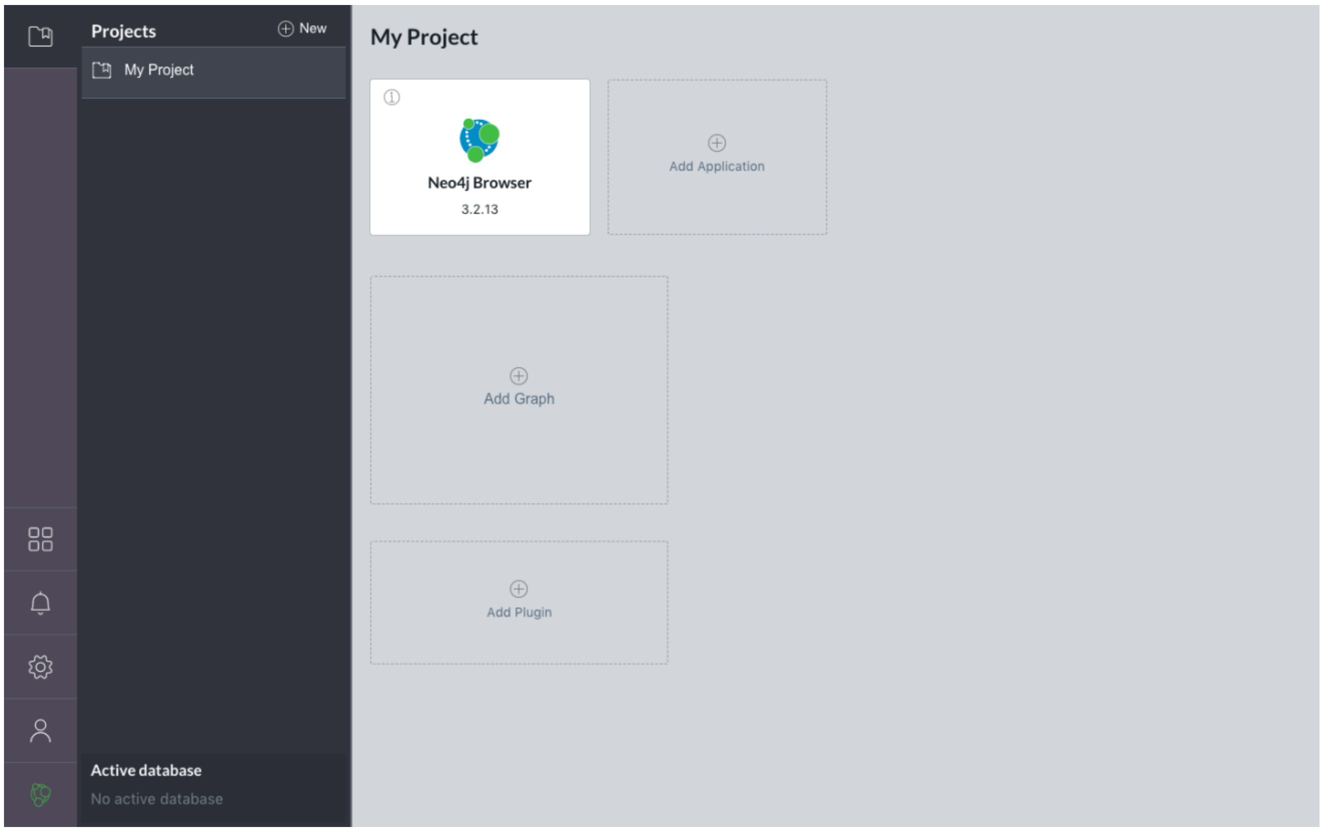
Following are the nodes, properties, the relations and relationship properties in above presented graph data model-

Sr.No	Nodes	Properties	Relationships	Relationship attribute
1	Category	type	contains	
2	Product	title SalesRank Brand	has	
3	question	question_text	is given	
4	answer	answer_by answer_date		
5	review	reviewText review_date		
6	reviewer	reviewer_name reviewer_location	gives provides mark helpful	ishelpful
7	comment	comment_text comment_date	contains	
8	reply	replied_by reply_date		

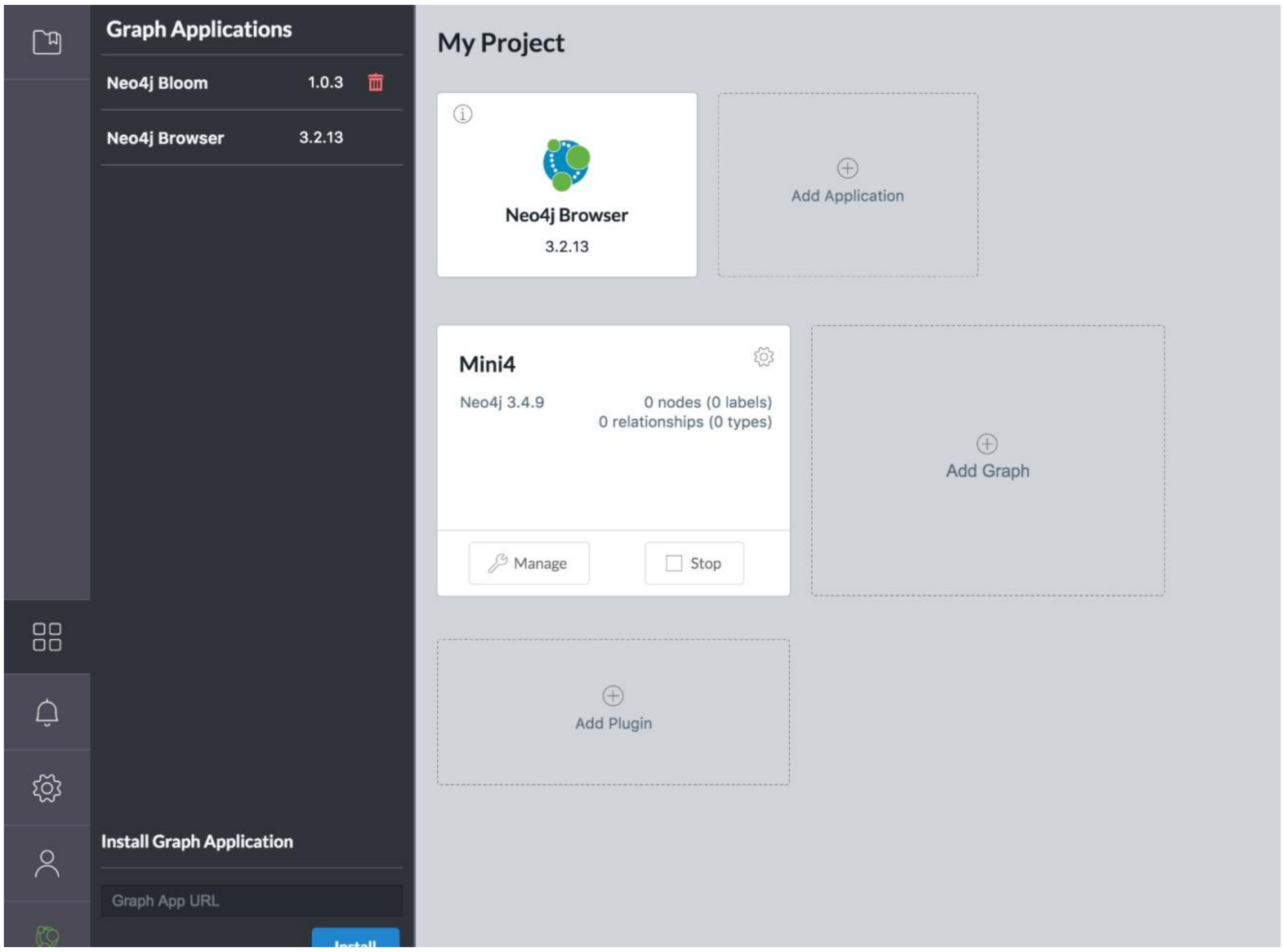
<u>Purpose 3:</u>

Installation of Neo4J

- Member- Deepali Pareek



- Member- Shrikant Patel


Graph Applications

Neo4j Bloom	1.0.3
Neo4j Browser	3.2.13

Install Graph Application

Graph App URL

My Project

Neo4j Browser
3.2.13

Add Application

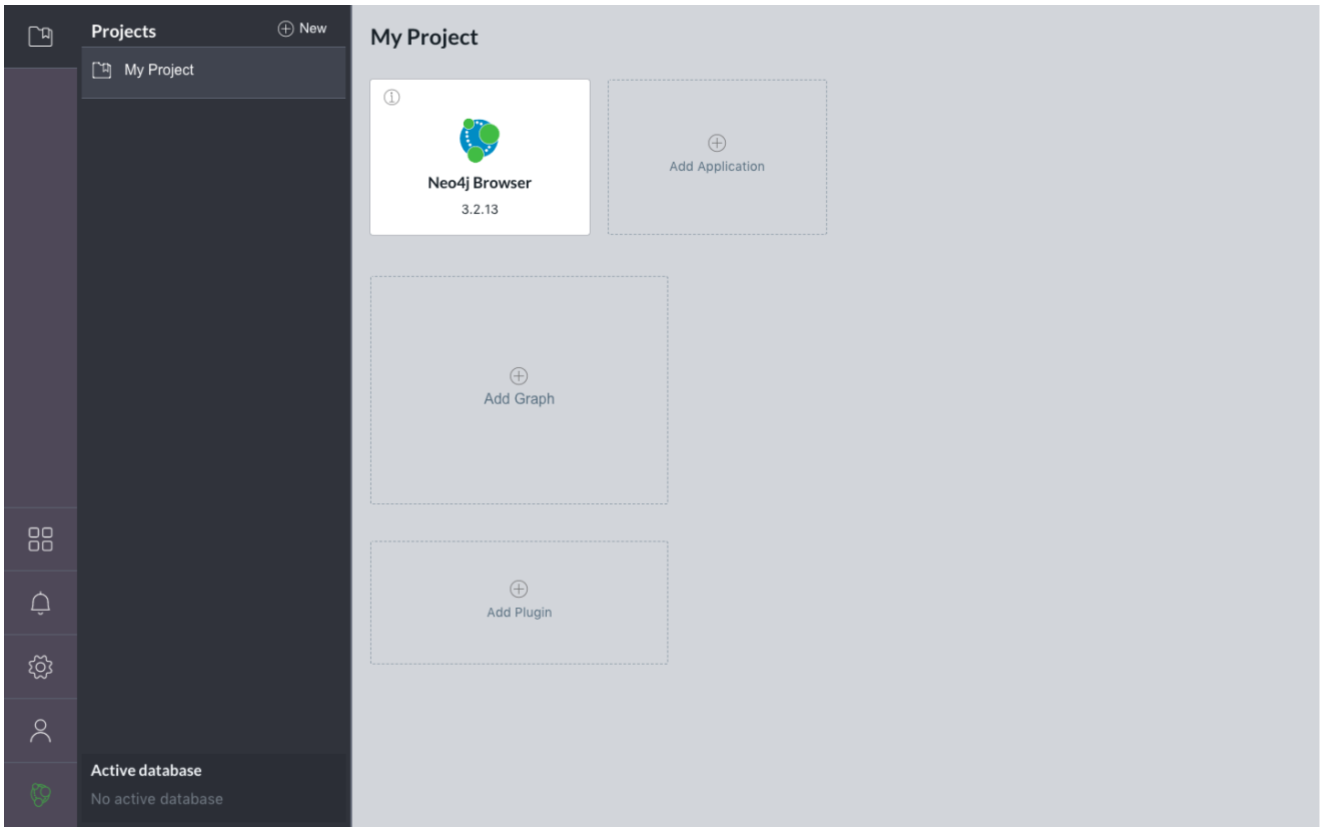
Mini4
Neo4j 3.4.9
0 nodes (0 labels)
0 relationships (0 types)

Manage
Stop

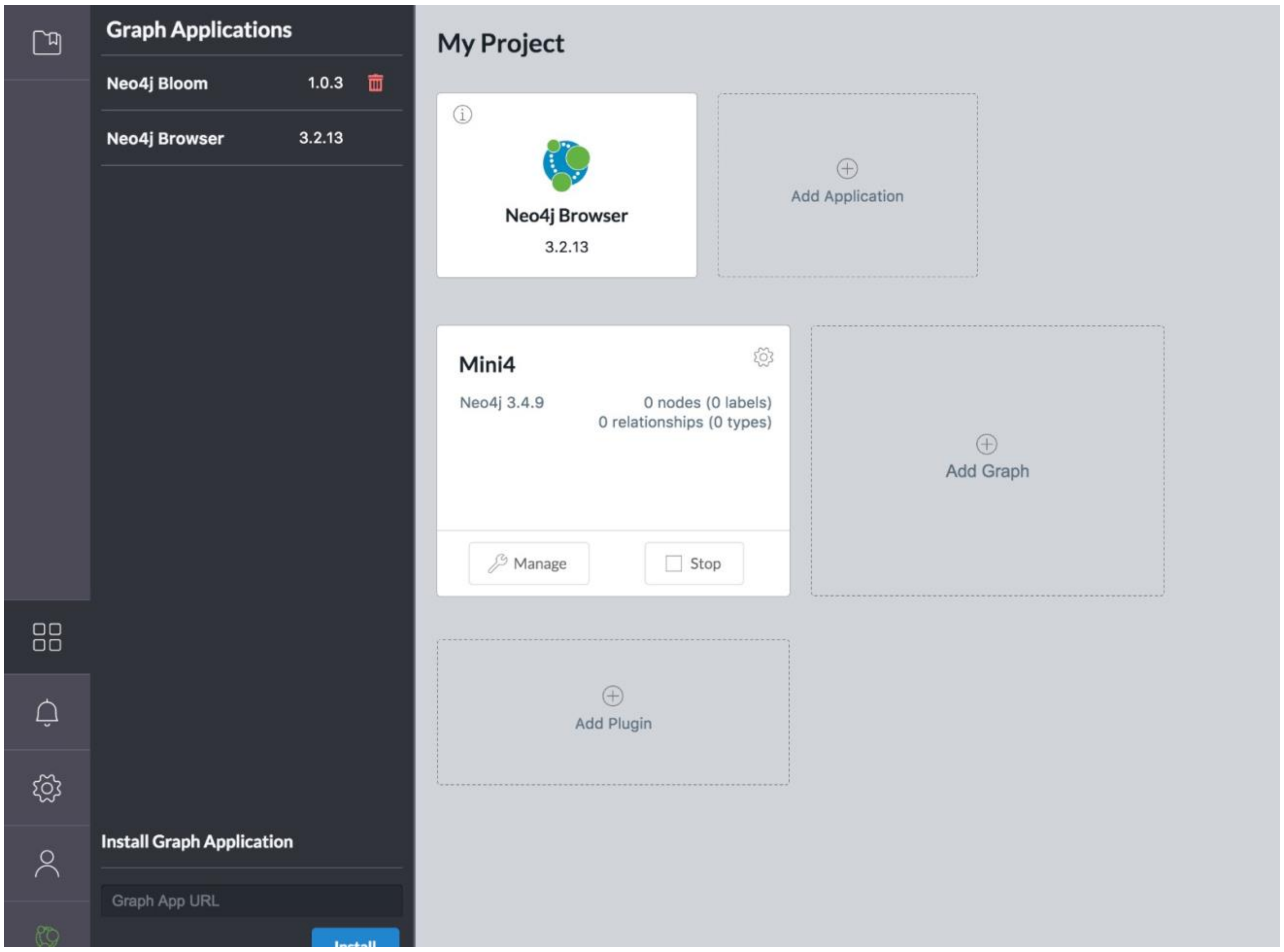
Add Graph

Add Plugin

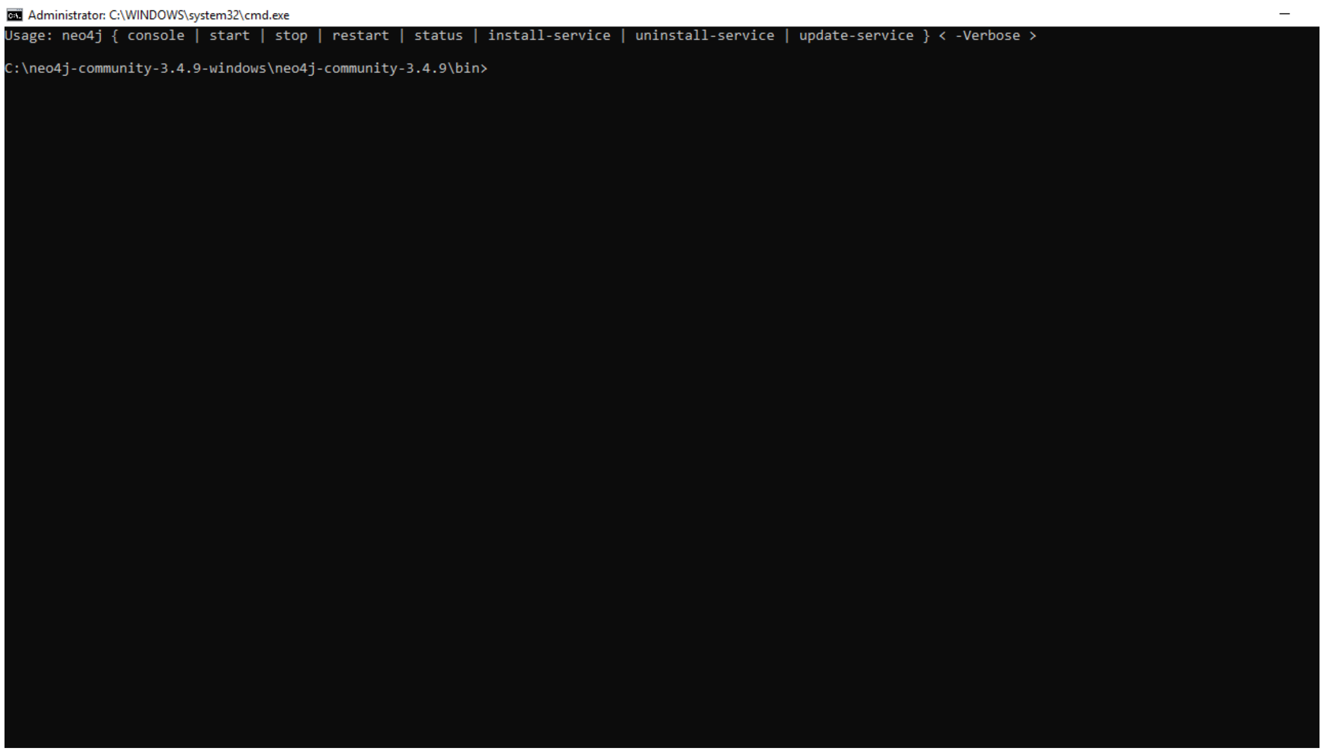
- Member- Rahul Sharma



- Member - Shruthi Jabashetty



- Anubhav Pradhan



```
Usage: neo4j { console | start | stop | restart | status | install-service | uninstall-service | update-service } < -Verbose >

C:\neo4j-community-3.4.9-windows\neo4j-community-3.4.9\bin>
```