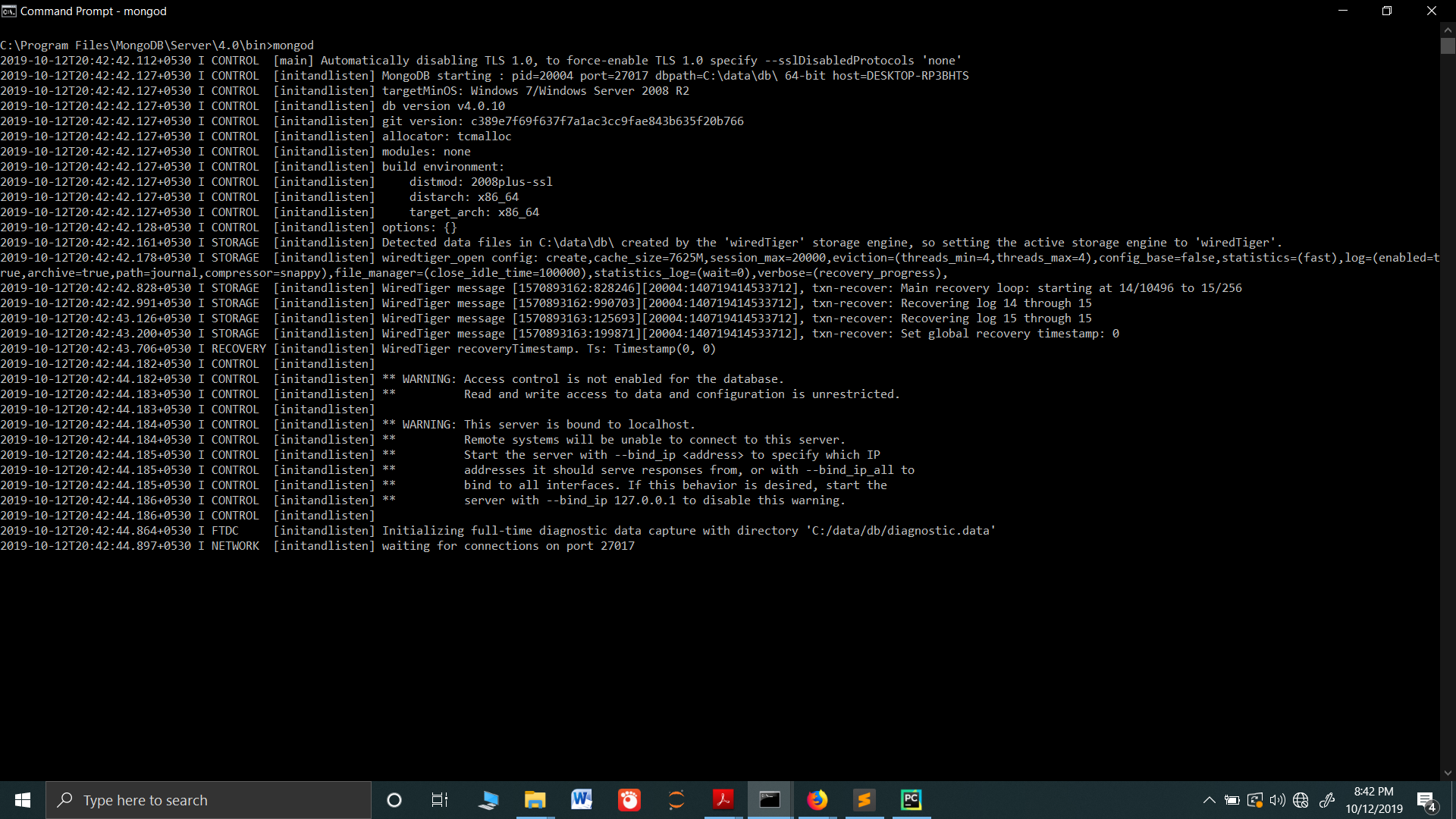
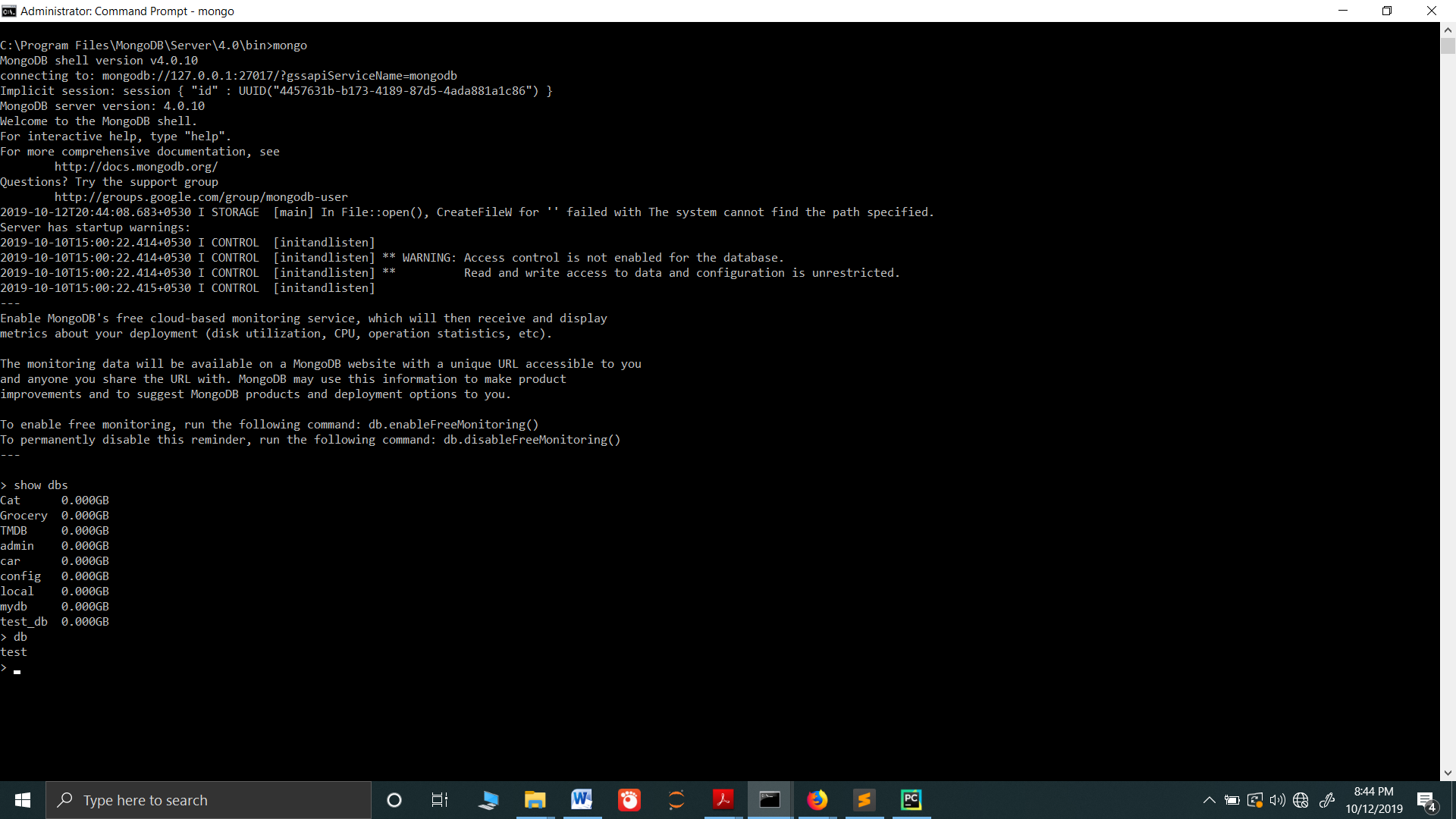
Problem Statement 1:





Problem Statement 2:

use Mongo\_DB\_Project

1: Customer Collection schema

db.createCollection("Customers", {  
 validator: {  
 $jsonSchema: {  
 bsonType: "object",  
 required: [ "customer\_id","customer\_name","contact\_number", "address","birth\_date" ],  
 properties: {  
  
 customer\_id: {  
 bsonType: "string",  
 description: "must be a string and is required"  
 },  
 customer\_name: {  
 bsonType: "string",  
 description: "must be a string and is required"  
 },  
 contact\_number: {  
 bsonType: "int",  
 description: "must be a string and is required"  
 },  
 address: {  
 bsonType: "object",  
 required: [ "streetAddress","city","state","postalCode"],  
 properties: {  
 streetAddress: {  
 bsonType: "string",  
 description: "must be a string if the field exists"  
 },  
 city: {  
 bsonType: "string",  
 "description": "must be a string and is required"  
 },  
 state:{  
 bsonType: "string",  
 description: "must be a string if the field exists"  
 },  
 postalCode:{  
 bsonType: "int",  
 description: "must be a integier if the field exists"  
 }  
 }  
 },  
 birth\_date:{  
 bsonType:"date",  
 "description": "must be a date and is required"  
 }  
 }  
 }  
 }  
})

2: Order collection schema

db.createCollection("Orders", {  
 validator: {  
 $jsonSchema: {  
 bsonType: "object",  
 required: [ "order\_id","order\_name","category", "brand","items","order\_date","shipping\_date"],  
 properties: {  
 order\_id: {  
 bsonType: "string",  
 description: "must be a string and is required"  
 },  
 order\_name: {  
 bsonType: "string",  
 description: "must be a string and is required"  
 },  
 category: {  
 bsonType: "object",  
 required: [ "category\_id","category\_name"],  
 properties: {  
 category\_id: {  
 bsonType: "string",  
 description: "must be a string if the field exists"  
 },  
 category\_name: {  
 bsonType: "string",  
 description: "must be a string if the field exists"  
 },  
 }  
 },  
 brand: {  
 bsonType: "object",  
 required: [ "brand\_id","brand\_name"],  
 properties: {  
 brand\_id: {  
 bsonType: "string",  
 description: "must be a string if the field exists"  
 },  
 brand\_name: {  
 bsonType: "string",  
 description: "must be a string if the field exists"  
 },  
 }  
 },  
 items: {  
 bsonType: "object",  
 required: [ "items\_id","items\_name","price"],  
 properties: {  
 items\_id: {  
 bsonType: "string",  
 description: "must be a string if the field exists"  
 },  
 items\_name: {  
 bsonType: "string",  
 description: "must be a string if the field exists"  
 },  
 price: {  
 bsonType: "int",  
 description: "must be a integer if the field exists"  
 }  
 }  
 },  
 order\_date:{  
 bsonType: "date",  
 description: "must be a string if the field exists"  
 },  
 shipping\_date:{  
 bsonType: "date",  
 description: "must be a string if the field exists"  
 },  
 }  
 }  
 }  
})

3: Categories collection schema

db.createCollection("Categories", {  
 validator: {  
 $jsonSchema: {  
 bsonType: "object",  
 required: [ "category\_id","category\_name","brands"],  
 properties: {  
 category\_id: {  
 bsonType: "string",  
 description: "must be a string and is required"  
 },  
 category\_name: {  
 bsonType: "string",  
 description: "must be a string and is required"  
 },  
 brands: {  
 bsonType: "object",  
 required: [ "brand\_id","brand\_name","items"],  
 properties: {  
 brand\_id: {  
 bsonType: "string",  
 description: "must be a string if the field exists"  
 },  
 brand\_name: {  
 bsonType: "string",  
 description: "must be a string if the field exists"  
 },  
 items:{  
 bsonType: "object",  
 required: [ "items\_id","items\_name","price"],  
 properties: {  
 items\_id: {  
 bsonType: "string",  
 description: "must be a string if the field exists"  
 },  
 items\_name: {  
 bsonType: "string",  
 description: "must be a string if the field exists"  
 },  
 price: {  
 bsonType: "int",  
 description: "must be a integer if the field exists"  
 }  
 }  
 }  
 }  
 }  
 }  
 }  
 }  
})

Problem Statement 3:

>use admin  
>db.createUser({user: "Adam", pwd: "adam", roles: ["readWrite","dbAdmin"]});

db.Customers.insert(  
 [  
 {  
 "customer\_id": "CUST001",  
 "customer\_name": "John",  
 "contact\_number": "9158385438",  
 "address":  
 {  
 "streetAddress": "709 Honey Creek Dr.",  
  
 "city": "New York",  
 "state": "NY",  
 "postalCode": "10028"  
 },  
 "birth\_date": "12-12-1993"  
 },  
 {  
 "customer\_id": "CUST002",  
 "customer\_name": "James",  
 "contact\_number": "9158385234",  
 "address":  
 {  
 "streetAddress": "3 South Sherman Street Astoria",  
 "city": "New York",  
 "state": "NY",  
 "postalCode": "11106"  
 },  
 "birth\_date": "13-12-1994"  
 },  
 {  
 "customer\_id": "CUST003",  
 "customer\_name": "Smith",  
 "contact\_number": "8758385438",  
 "address":  
 {  
 "streetAddress": "73 Pacific St. Forest Hills",  
 "city": "New York",  
 "state": "NY",  
 "postalCode": "11375"  
 },  
 "birth\_date": "10-12-1993"  
 },  
 {  
 "customer\_id": "CUST004",  
 "customer\_name": "Linda",  
 "contact\_number": "7858385438",  
 "address":  
 {  
 "streetAddress": "812 Thatcher Court Yonkers",  
 "city": "New York",  
 "state": "NY",  
 "postalCode": "10701"  
 },  
 "birth\_date": "10-12-1994"  
 },  
 {  
 "customer\_id": "CUST004",  
 "customer\_name": "Mary",  
 "contact\_number": "7875085438",  
 "address":  
 {  
 "streetAddress": "93 Bayport Ave. South Richmond Hill",  
 "city": "New York",  
 "state": "NY",  
 "postalCode": "11419"  
 },  
 "birth\_date": "10-12-1994"  
 }  
 ]  
  
)

db.Orders.insert([  
{  
 "order\_id": "ORD001",  
 "orderer\_name": "John",  
 "category": {  
 "category\_id": "CAT001",  
 "category\_name": "mobile"  
 },  
 "brand":  
 {  
 "brand\_id": "BRN002",  
 "brand\_name": "Samsung"  
 },  
 "items":  
 {  
 "item\_id": "005",  
 "item\_name": "Samsung Galaxy J7",  
 "price": 11890  
 },  
 "order\_date": "12-12-2017",  
 "shipping\_date": "17-12-2017"  
},  
{  
 "order\_id": "ORD002",  
 "orderer\_name": "James",  
 "category": {  
 "category\_id": "CAT003",  
 "category\_name": "TV"  
 },  
 "brands":  
 {  
 "brand\_id": "BRN006",  
 "brand\_name": "Philips"  
 },  
 "items":  
 {  
 "item\_id": "021",  
 "item\_name": "32PFL3931/V7",  
 "price": 15999  
 },  
 "order\_date": "23-12-2017",  
 "shipping\_date": "27-12-2017"  
},  
{  
 "order\_id": "ORD003",  
 "orderer\_name": "John",  
 "category": {  
 "category\_id": "CAT001",  
 "category\_name": "mobile"  
 },  
 "brand":  
 {  
 "brand\_id": "BRN002",  
 "brand\_name": "Samsung"  
 },  
 "items":  
 {  
 "item\_id": "005",  
 "item\_name": "Samsung Galaxy J7",  
 "price": 11890  
 },  
 "order\_date": "12-12-2017",  
 "shipping\_date": "17-12-2017"  
},  
{  
 "order\_id": "ORD004",  
 "orderer\_name": "Smith",  
 "category": {  
 "category\_id": "CAT002",  
 "category\_name": "computer"  
 },  
 "brand":  
 {  
 "brand\_id": "BRN004",  
 "brand\_name": "Apple"  
 },  
 "items":  
 {  
 "item\_id": "013",  
 "item\_name": "Apple iMac",  
 "price": 82000  
 },  
 "order\_date": "12-12-2017",  
 "shipping\_date": "17-12-2017"  
}  
])

db.Categories.insert(  
[{  
 "category\_id": "CAT001",  
 "category\_name": "mobile",  
 "brands": [  
 {  
 "brand\_id": "BRN001",  
 "brand\_name": "Sony",  
 "items":[  
 {  
 "item\_id": "001",  
 "item\_name": "Xperia R1",  
 "price": 13990  
 },  
 {  
 "item\_id": "002",  
 "item\_name": "Xperia XZ",  
 "price": 46990  
 },  
 {  
 "item\_id": "003",  
 "item\_name": "Xperia XA",  
 "price": 24990  
 },  
 {  
 "item\_id": "004",  
 "item\_name": "Xperia Z5",  
 "price": 42990  
 }  
 ]  
 },  
 {  
 "brand\_id": "BRN002",  
 "brand\_name": "Samsung",  
 "items":[  
 {  
 "item\_id": "005",  
 "item\_name": "Samsung Galaxy J7",  
 "price": 11890  
 },  
 {  
 "item\_id": "006",  
 "item\_name": "Samsung Galaxy S8",  
 "price": 52999  
 },  
 {  
 "item\_id": "007",  
 "item\_name": "Samsung Galaxy Note",  
 "price": "$1000"  
 },  
 {  
 "item\_id": "008",  
 "item\_name": "Samsung Galaxy A9",  
 "price": 22499  
 }  
 ]  
 }  
 ]  
},  
{  
 "category\_id": "CAT002",  
 "category\_name": "computer",  
 "brands": [  
 {  
 "brand\_id": "BRN003",  
 "brand\_name": "Dell",  
 "items":[  
 {  
 "item\_id": "009",  
 "item\_name": "Dell OptiPlex",  
 "price": 32200  
 },  
 {  
 "item\_id": "010",  
 "item\_name": "Dell Inspiron",  
 "price": 26990  
 },  
 {  
 "item\_id": "011",  
 "item\_name": "Dell AIO",  
 "price": 23876  
 }  
 ]  
 },  
 {  
 "brand\_id": "BRN004",  
 "brand\_name": "Apple",  
 "items":[  
 {  
 "item\_id": "013",  
 "item\_name": "Apple iMac",  
 "price": 82000  
 },  
 {  
 "item\_id": "014",  
 "item\_name": "Apple MGEN2HN/A",  
 "price": 52849  
 }  
 ]  
 }  
 ]  
 },  
 {  
 "category\_id": "CAT003",  
 "category\_name": "TV",  
 "brands": [  
 {  
 "brand\_id": "BRN005",  
 "brand\_name": "LG",  
 "items":[  
 {  
 "item\_id": "017",  
 "item\_name": "49LJ52KT",  
 "price": 659990  
 },  
 {  
 "item\_id": "018",  
 "item\_name": "49LJ53IT",  
 "price": 48990  
 }  
 ]  
 },  
 {  
 "brand\_id": "BRN006",  
 "brand\_name": "Philips",  
 "items":[  
 {  
 "item\_id": "021",  
 "item\_name": "32PFL3931/V7",  
 "price": 15999  
 },  
 {  
 "item\_id": "022",  
 "item\_name": "43PUTT7791",  
 "price": 43490  
 },  
 {  
 "item\_id": "023",  
 "item\_name": "55PFL5059/V7",  
 "price": 56920  
 },  
 ]  
 }  
 ]  
 }  
])

db.Customers.find().pretty()  
  
db.Orders.find().pretty()  
  
db.Categories.find().pretty()

db.Categories.update({'brands.items.item\_naame':'Dell OptiPlex'},{$set:{'brands.items.price':33000}})

Problem Statement 4:

>mongoexport -d Mongo\_DB\_Project -c Categories --type=json --fields brands --out 'desktop/mydump/students/brands.json'

>mongoimport -d Mongo\_DB\_Project -c Categories --type json --file 'desktop/mydump/students/brands.json'

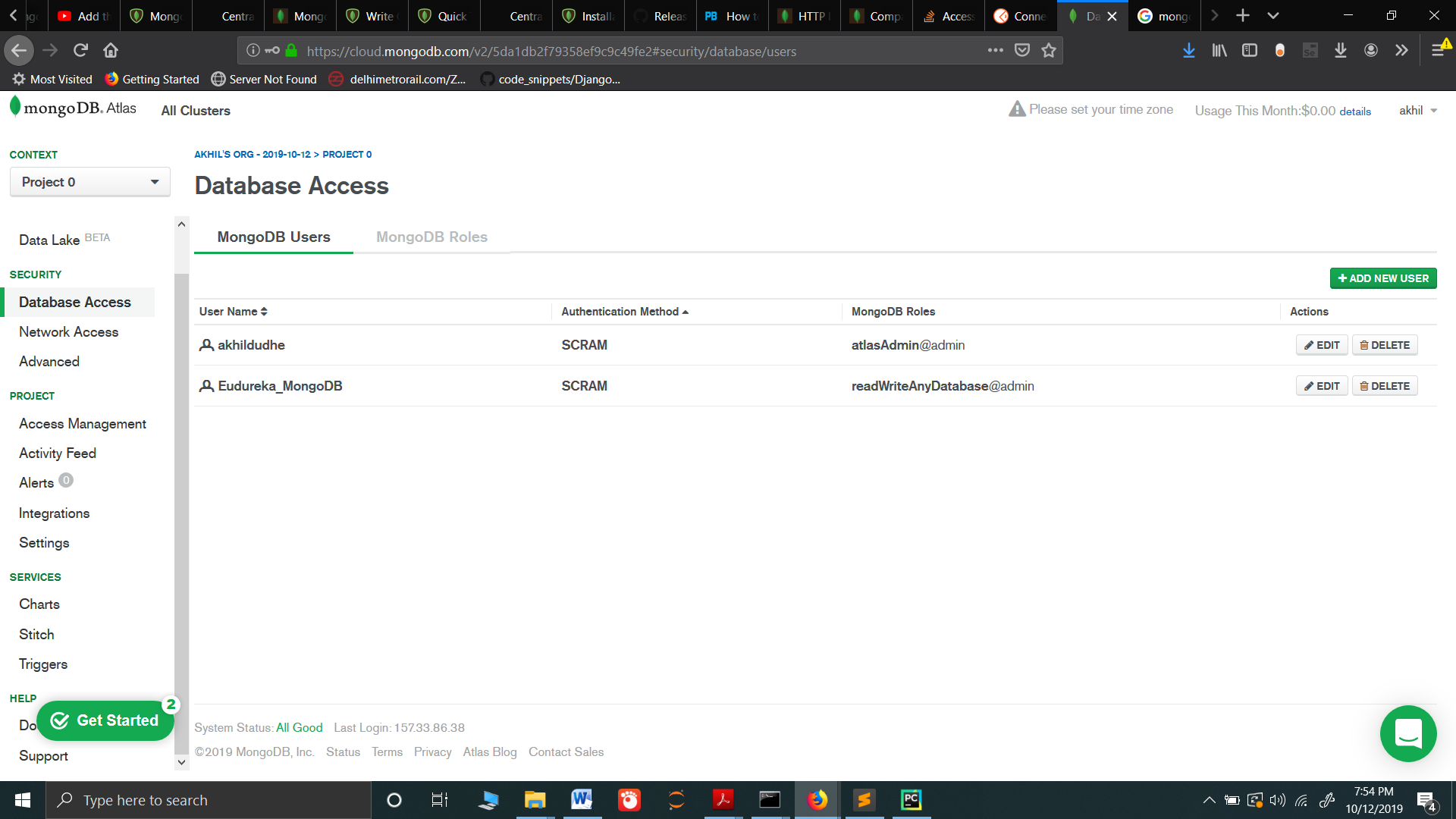
Problem Statement 5:

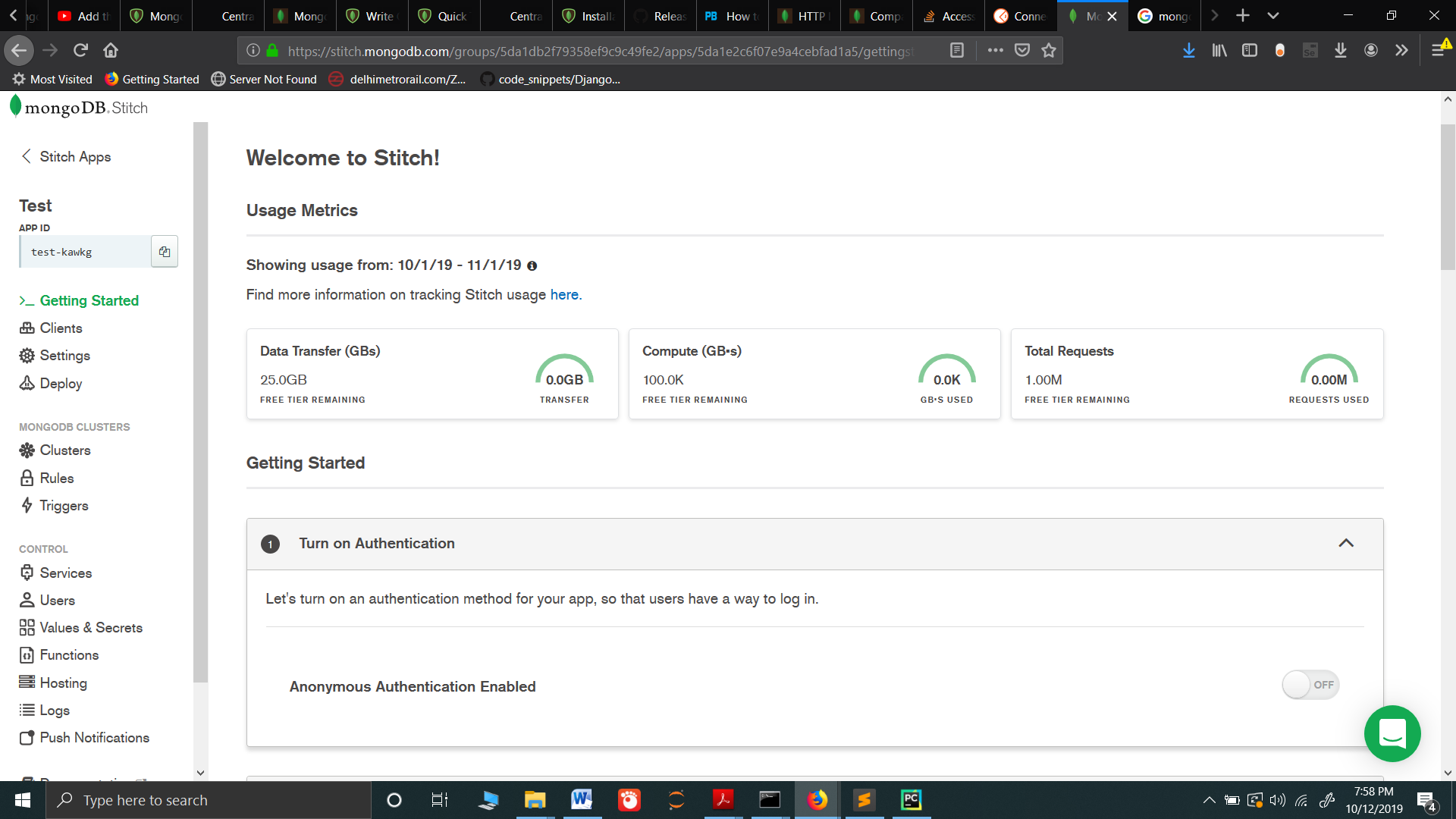
db.Categories.ensureIndex({'brands':1})

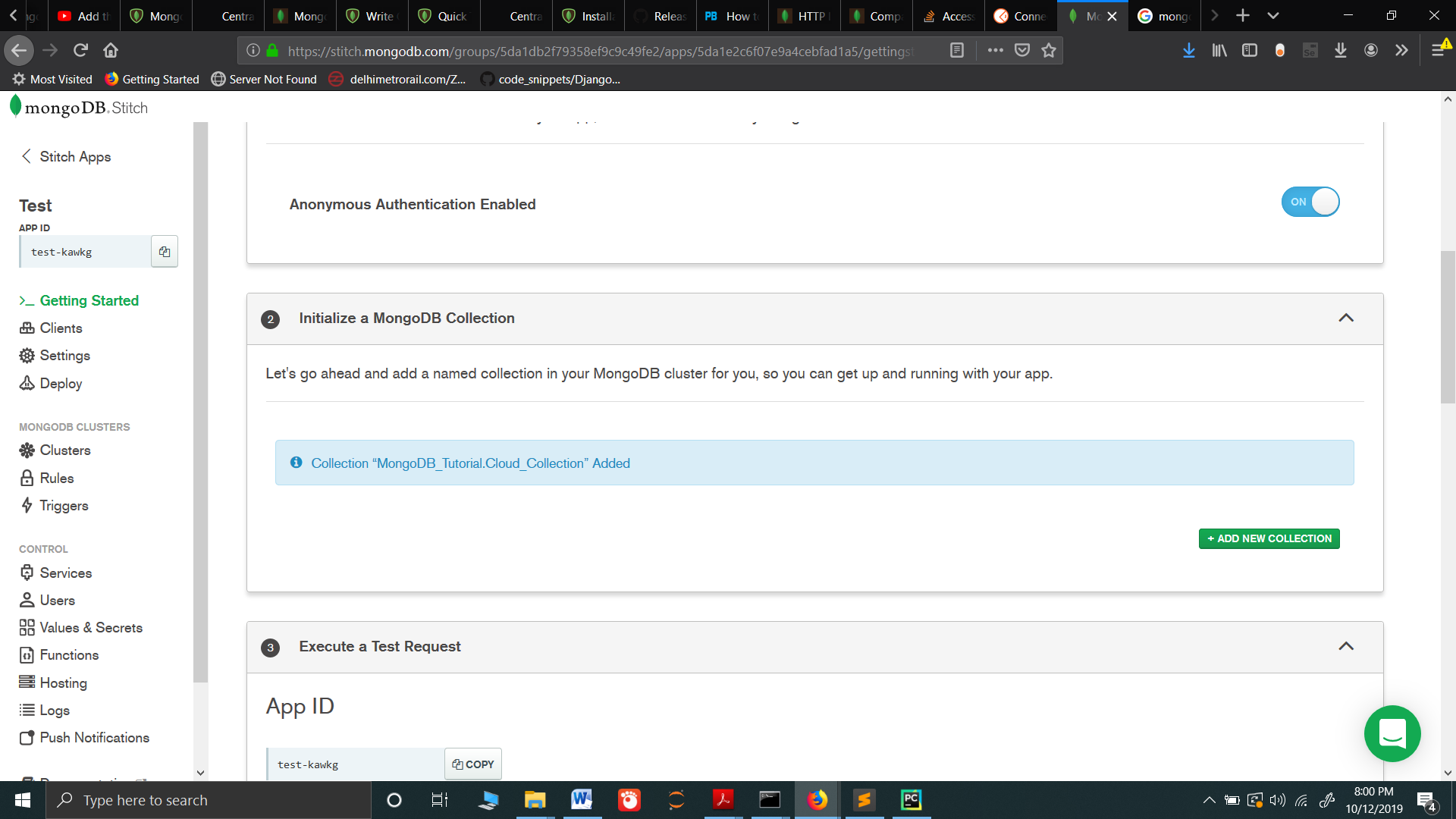
Problem Statement 6:

db.Categories.aggregate**({**$group**:** **{**\_id**:** ''**,** TotalPrice**:** **{** $sum**:** '$brands.items.price' **}}})**

Problem Statement 7:







Problem Statement 8:

db.runCommand( { buildInfo: 1 } )

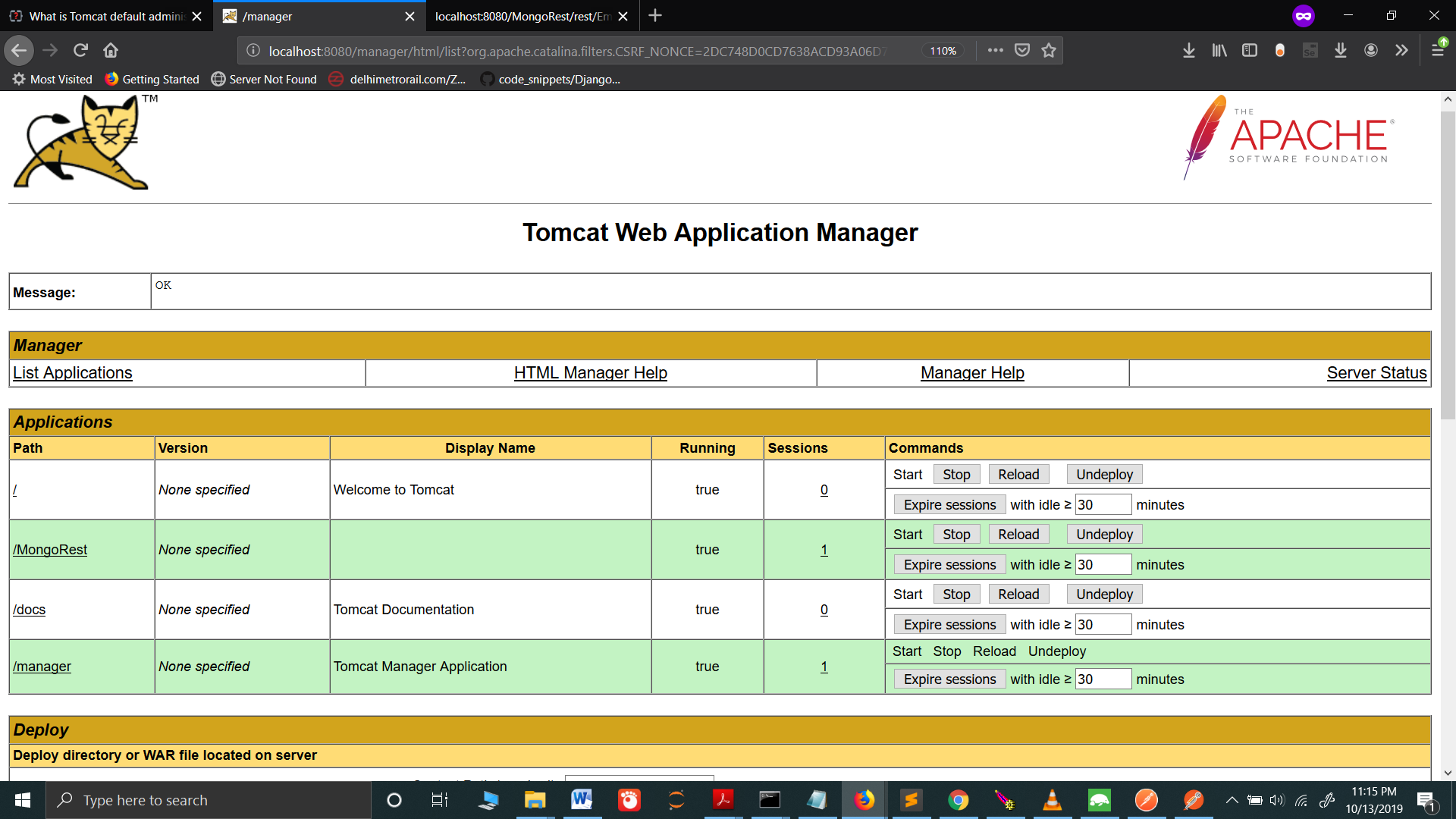
db.runCommand( { collStats : "brands", scale: 1024 } )

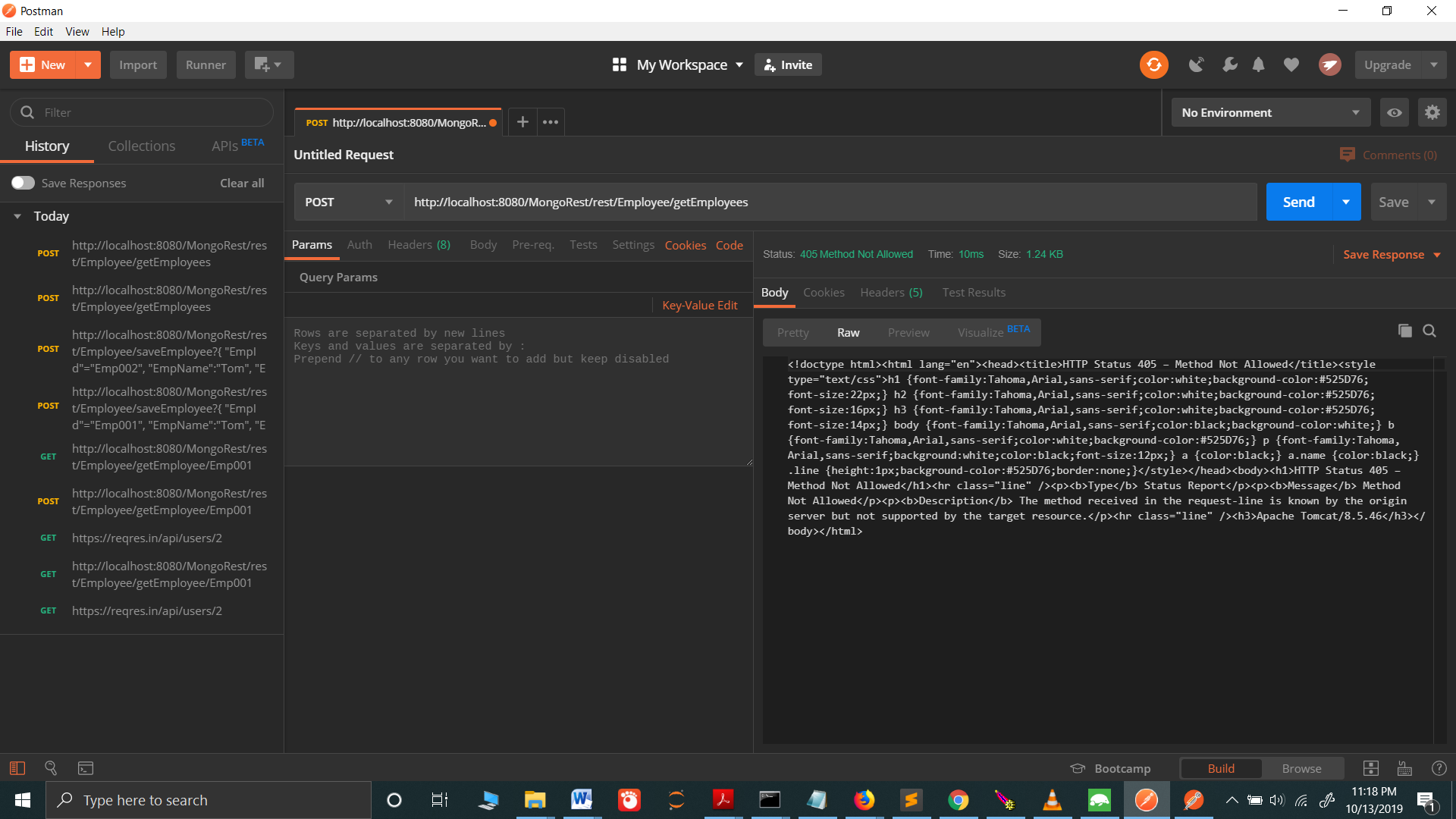
rs.status()

sh.status()

db.runCommand( { serverStatus: 1 } )

Problem Statement 9:





Problem Statement 10:

rs1---> 47017  
rs2---> 47018  
rs3---> 47019  
  
>start /b mongod --replSet rs --dbpath rs1 --port 47017 --logpath logs\rs1.log --logappend  
>start /b mongod --replSet rs --dbpath rs2 --port 47018 --logpath logs\rs2.log --logappend  
>start /b mongod --replSet rs --dbpath rs3 --port 47019 --logpath logs\rs3.log --logappend  
>rs.status()  
>rs.initiate()  
>rs.status()  
>rs.add('Pc\_name:47018')  
>rs.add('Pc\_name:47019')