

HU Extension Assignment 02
 Handed out: 09/11/2015
 09/18/2015

E-90 Cloud Computing
 Due by 11:59 PM EST

Problem 1: Access the S3 storage area using tools inside AWS Console and display an image that is available for public viewing.

Create a Bucket in S3

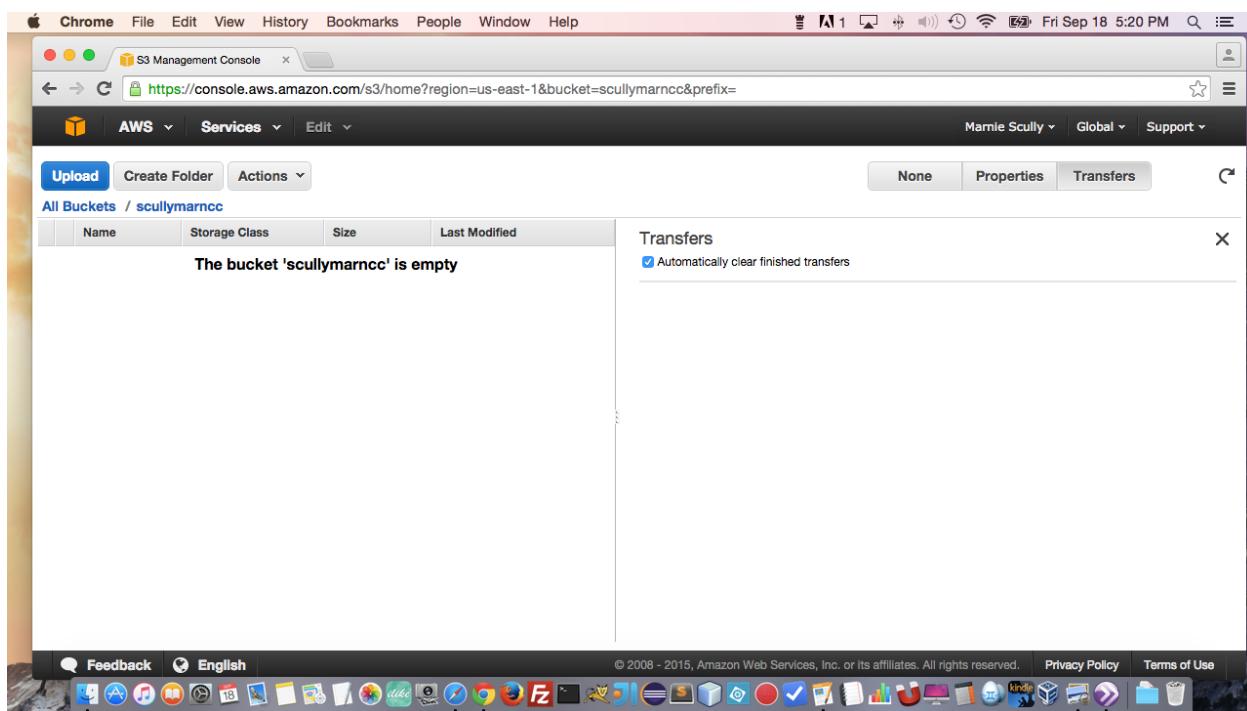
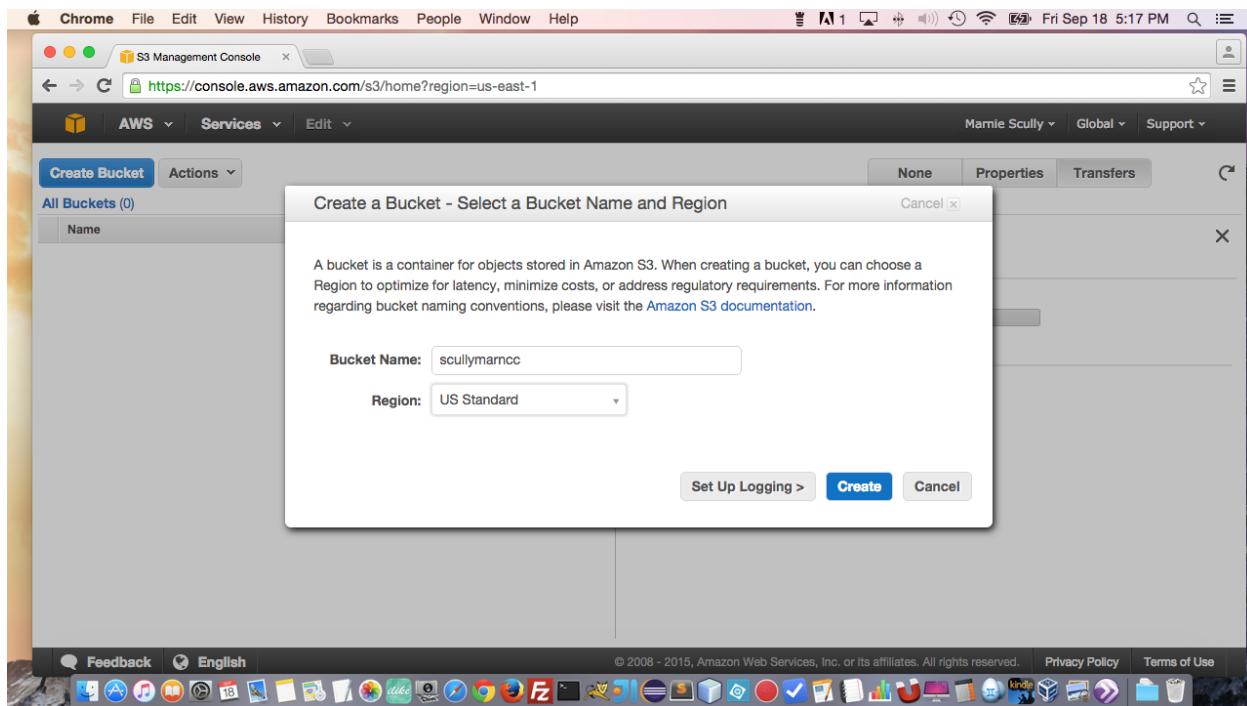
Select S3

Then Create a Bucket

Give Bucket Unique Name a choose region

The screenshot shows the AWS Management Console interface. At the top, there's a navigation bar with links for Chrome, File, Edit, View, History, Bookmarks, People, Window, Help, and a user dropdown for Marnie Scully. Below the navigation bar is the AWS logo and a Services dropdown. The main content area is titled "Amazon Web Services" and lists various services under categories like Compute, Storage & Content Delivery, Database, Developer Tools, Management Tools, Security & Identity, Mobile Services, Application Services, and Resource Groups. On the right side, there's a sidebar titled "Additional Resources" with links for Getting Started, AWS Console Mobile App, AWS Marketplace, and AWS Lambda. A vertical sidebar on the far right shows a file tree with folders labeled "Problem 1 Images", "Problem 2 Images", "Problem 3 Images", and "Problem 4 Images".

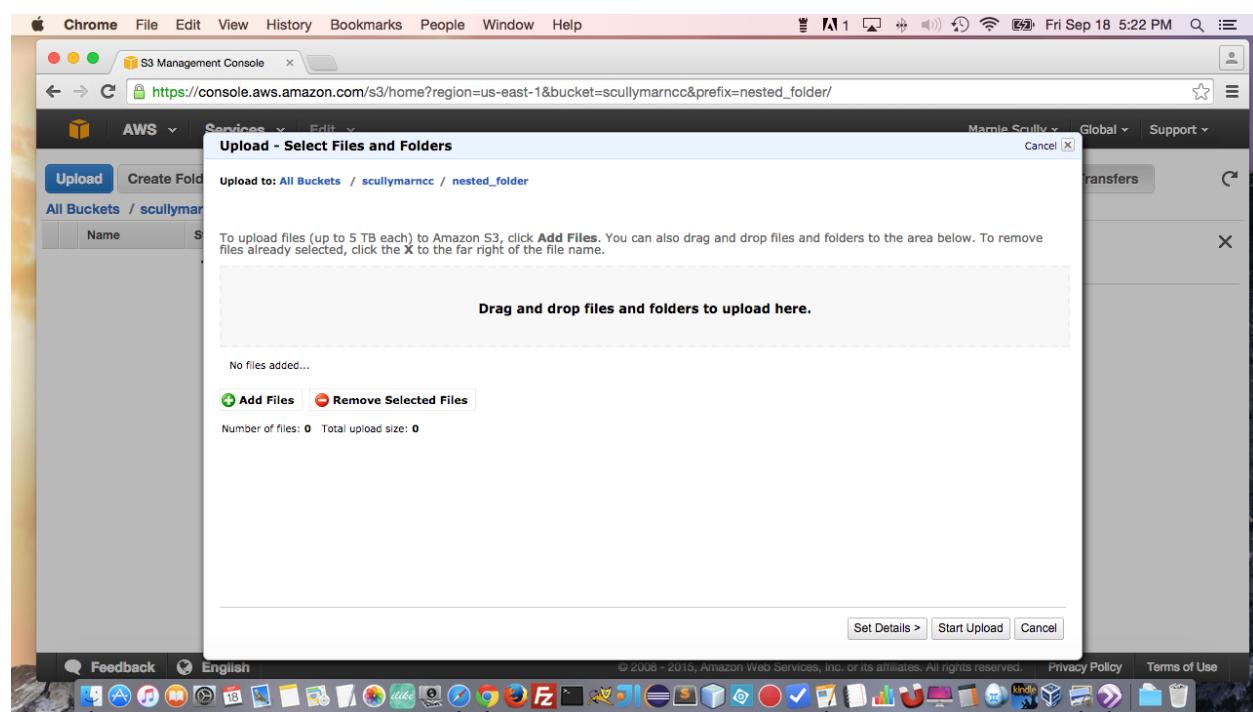
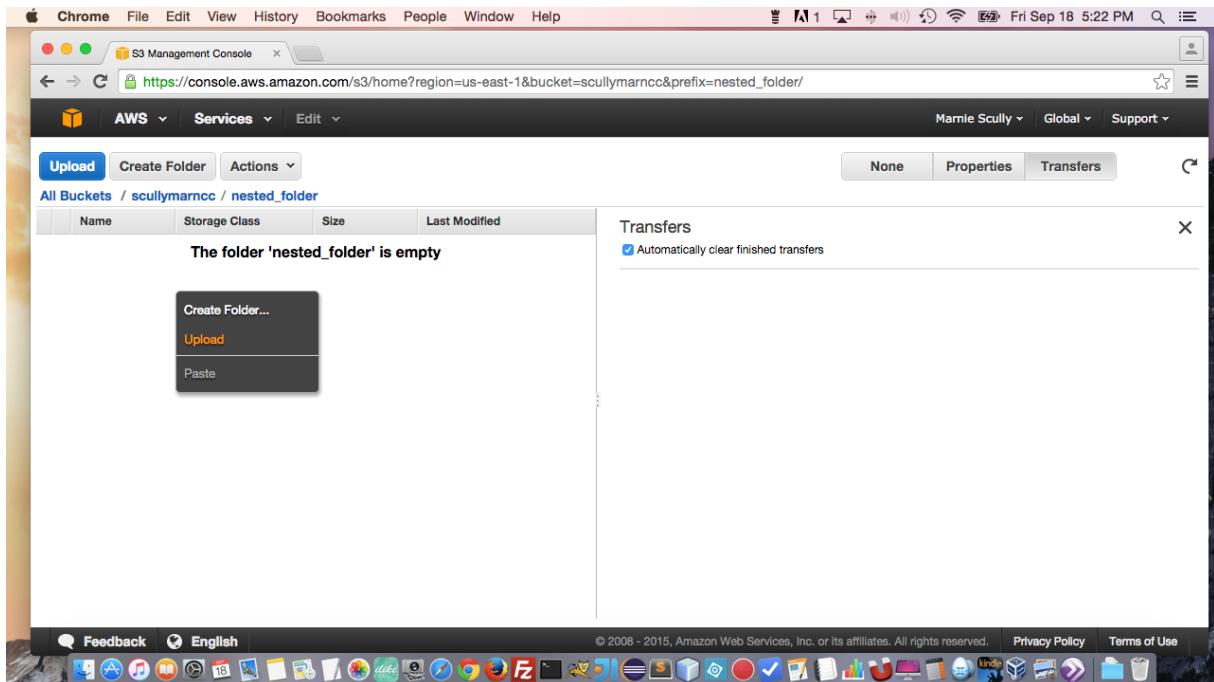
The screenshot shows the S3 Management Console. At the top, it says "Welcome to Amazon Simple Storage Service". It explains that Amazon S3 is storage for the Internet designed to make web-scale computing easier for developers. It highlights the benefits of being highly scalable, reliable, secure, fast, and inexpensive. Below this, it states that objects can be stored from 1 byte to 5 terabytes each. It also mentions that each object is stored in a bucket with a unique key. There's a "Create Bucket" button at the bottom left. The right side has an "Additional Information" section with links for Getting Started Guide, Documentation, and All S3 Resources. The bottom of the screen features a "Feedback" link, language selection (English), and standard footer links for Privacy Policy and Terms of Use.

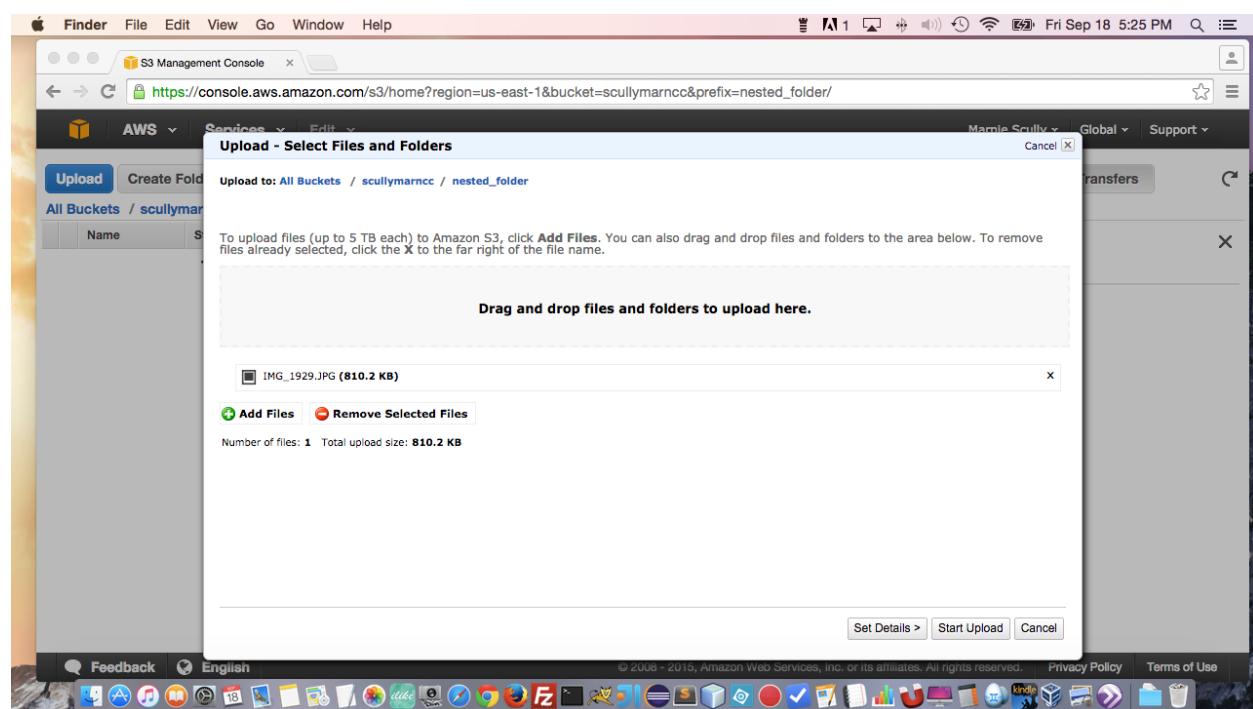
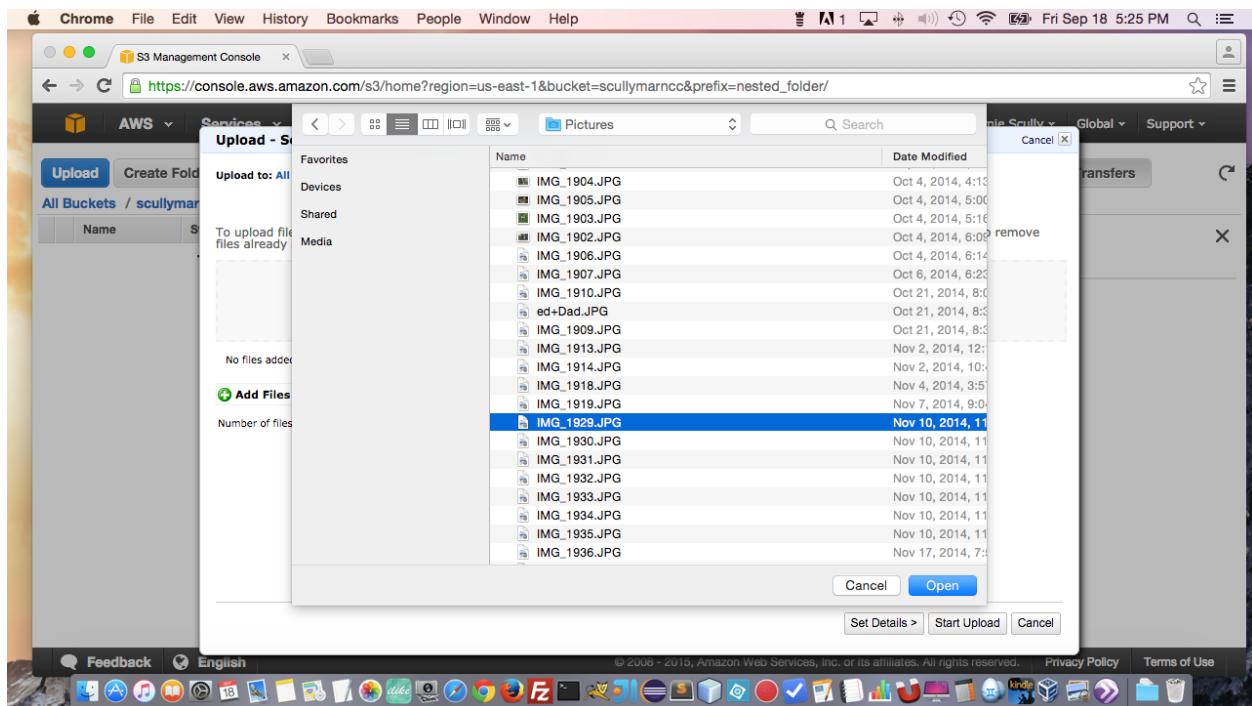


Create a Double-nested folder (a folder within a bucket)

The screenshot shows the AWS S3 Management Console interface. The URL in the address bar is <https://console.aws.amazon.com/s3/home?region=us-east-1&bucket=scullymarncc&prefix=>. The main content area displays a table of objects in the bucket 'scullymarncc'. The table has columns: Name, Storage Class, Size, and Last Modified. There is one object listed: 'nested_folder'. In the top navigation bar, there are tabs for 'Upload', 'Create Folder', and 'Actions'. To the right of the table, there is a 'Transfers' sidebar with the option 'Automatically clear finished transfers' checked. The bottom of the screen shows the Mac OS X dock with various application icons.

Upload a photograph





The screenshot shows the AWS S3 Management Console interface. The URL in the address bar is https://console.aws.amazon.com/s3/home?region=us-east-1&bucket=sculymarncc&prefix=nested_folder/. The main content area displays a table with one row, indicating that the folder 'nested_folder' is empty. The table columns are Name, Storage Class, Size, and Last Modified. A message at the top of the table says 'The folder 'nested_folder' is empty'. To the right of the table, there is a 'Transfers' sidebar with a status message: 'Starting' (0%) and 'Upload: Uploading IMG_1929.JPG to sculymarncc'. The bottom of the screen shows the Mac OS X dock with various application icons.

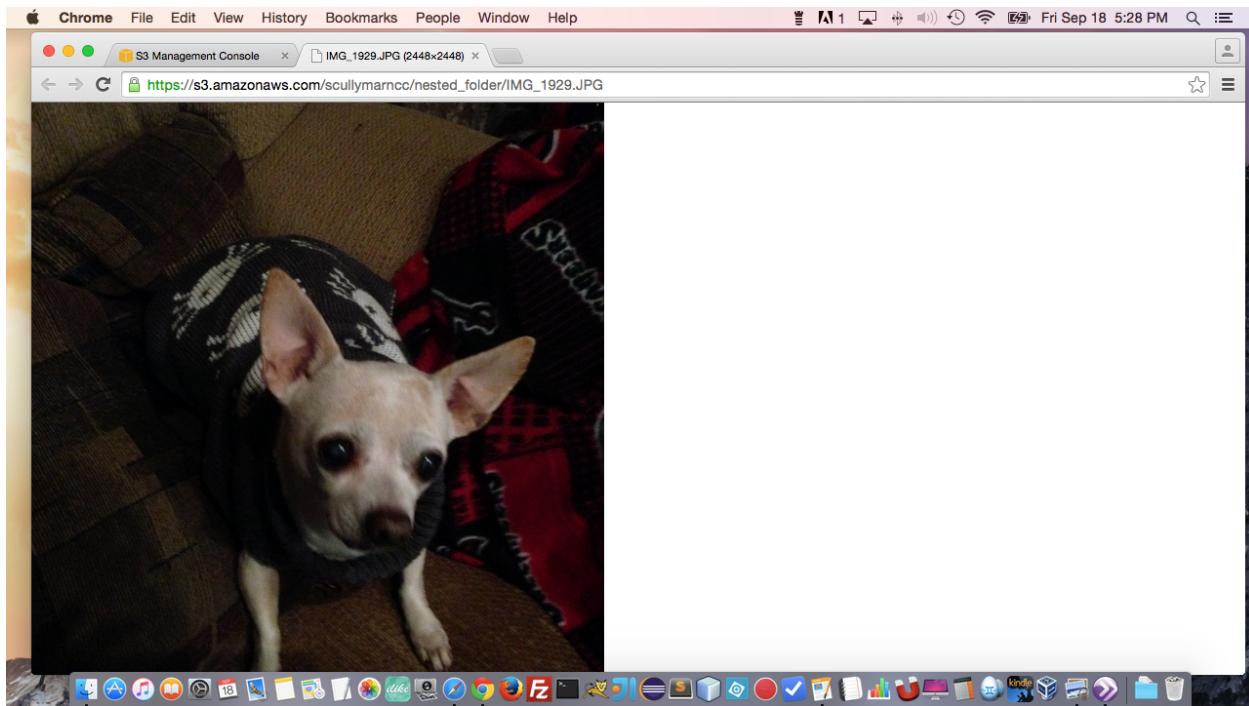
This screenshot shows the same AWS S3 Management Console interface as the previous one, but now the 'nested_folder' contains a single file named 'IMG_1929.JPG'. The file details are shown in the table: Name (IMG_1929.JPG), Storage Class (Standard), Size (810.2 KB), and Last Modified (Fri Sep 18 17:26:16 G). The 'Transfers' sidebar shows the upload has completed successfully. The Mac OS X dock is visible at the bottom.

Make the photograph publicly accessible

The screenshot shows the AWS S3 Management Console in a web browser. The URL is https://console.aws.amazon.com/s3/home?region=us-east-1&bucket=scullymarncc&prefix=nested_folder/. The page displays a table of files in the 'nested_folder' of the 'scullymarncc' bucket. One file, 'IMG_1929.JPG', is selected. On the right, detailed information about this file is shown, including its size (810.2 KB), storage class (Standard), and last modified date (Fri Sep 18 17:26:16 G). Below this, the 'Permissions' section is expanded, showing the current grantee as 'marniescally'. A checkbox for 'Open/Download' is checked, while 'View Permissions' is also checked. There is a link to 'Edit Permissions'.

This screenshot shows the same AWS S3 Management Console interface after changes have been made. The 'Grantee' dropdown has been set to 'Everyone'. The 'Open/Download' checkbox is checked, and the 'View Permissions' checkbox is unchecked. The 'Edit Permissions' button is visible below the permissions table. The rest of the page, including the file list and basic metadata, remains the same as in the previous screenshot.

Prove that you can see it in your browser

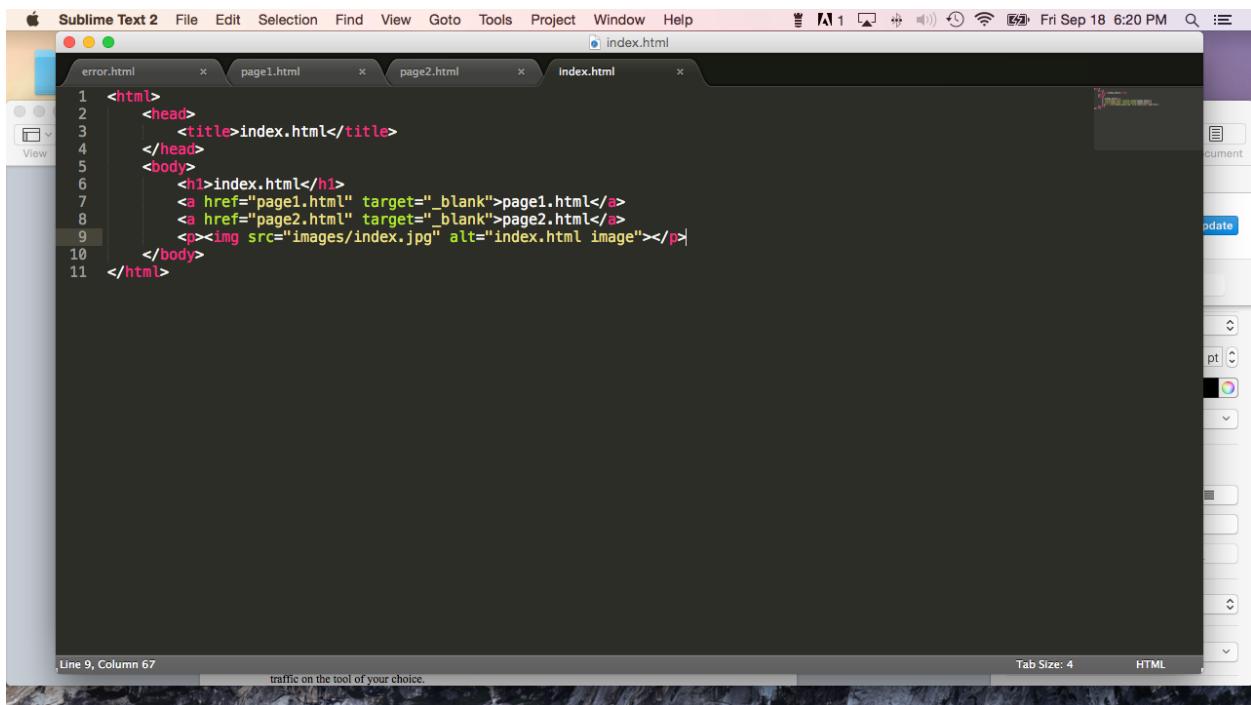


The Url:

https://s3.amazonaws.com/scullymarncc/nested_folder/IMG_1929.JPG

Problem 2: Create a tiny Web site (html files and images) on AWS using S3.

Create web pages for a web site with one index.html page and one error.html page.
Index.html contains two links to two simple static pages



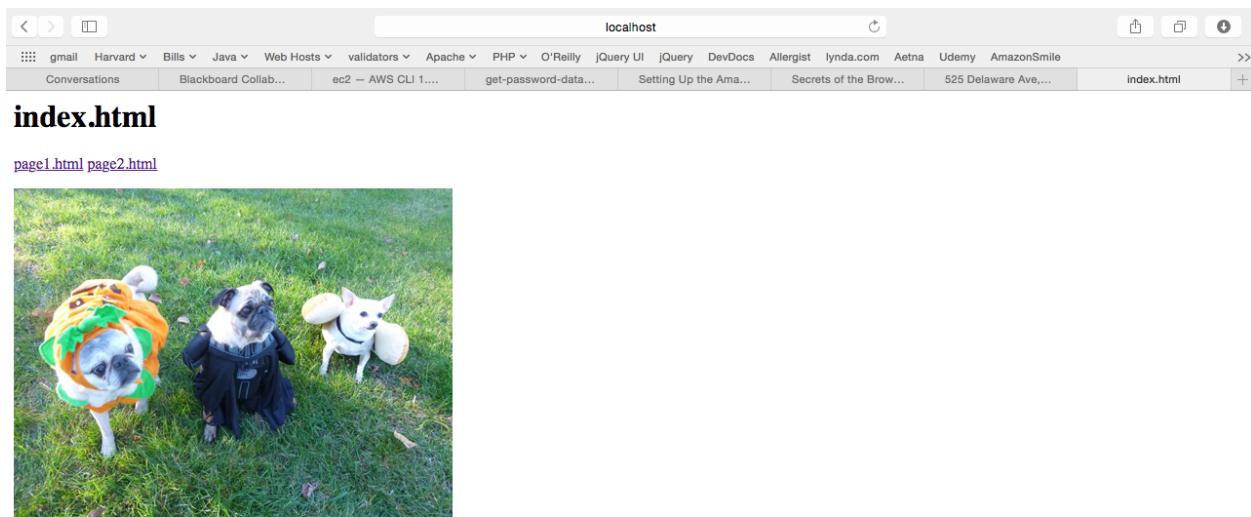
```
Sublime Text 2 File Edit Selection Find View Goto Tools Project Window Help
error.html x page1.html x page2.html x index.html x
Fri Sep 18 6:20 PM
1 <html>
2   <head>
3     <title>index.html</title>
4   </head>
5   <body>
6     <h1>index.html</h1>
7     <a href="page1.html" target="_blank">page1.html</a>
8     <a href="page2.html" target="_blank">page2.html</a>
9     <p></p>
10   </body>
11 </html>
```

Line 9, Column 67

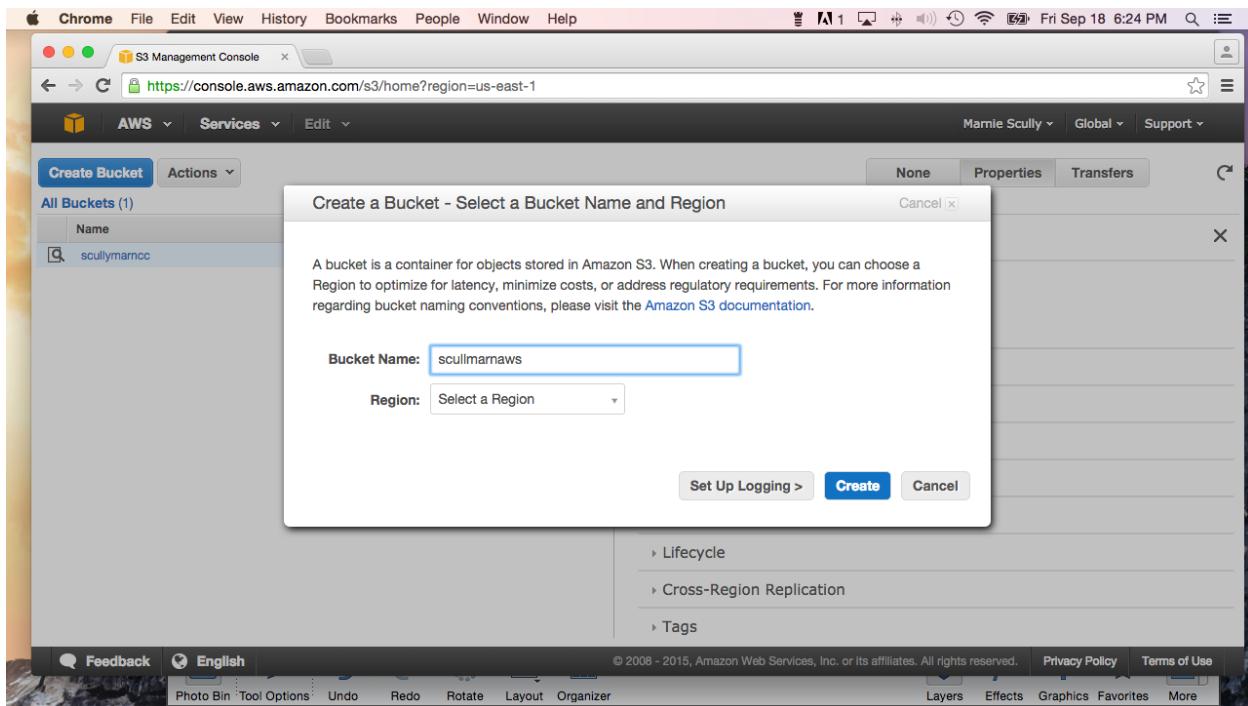
traffic on the tool of your choice.

Tab Size: 4 HTML

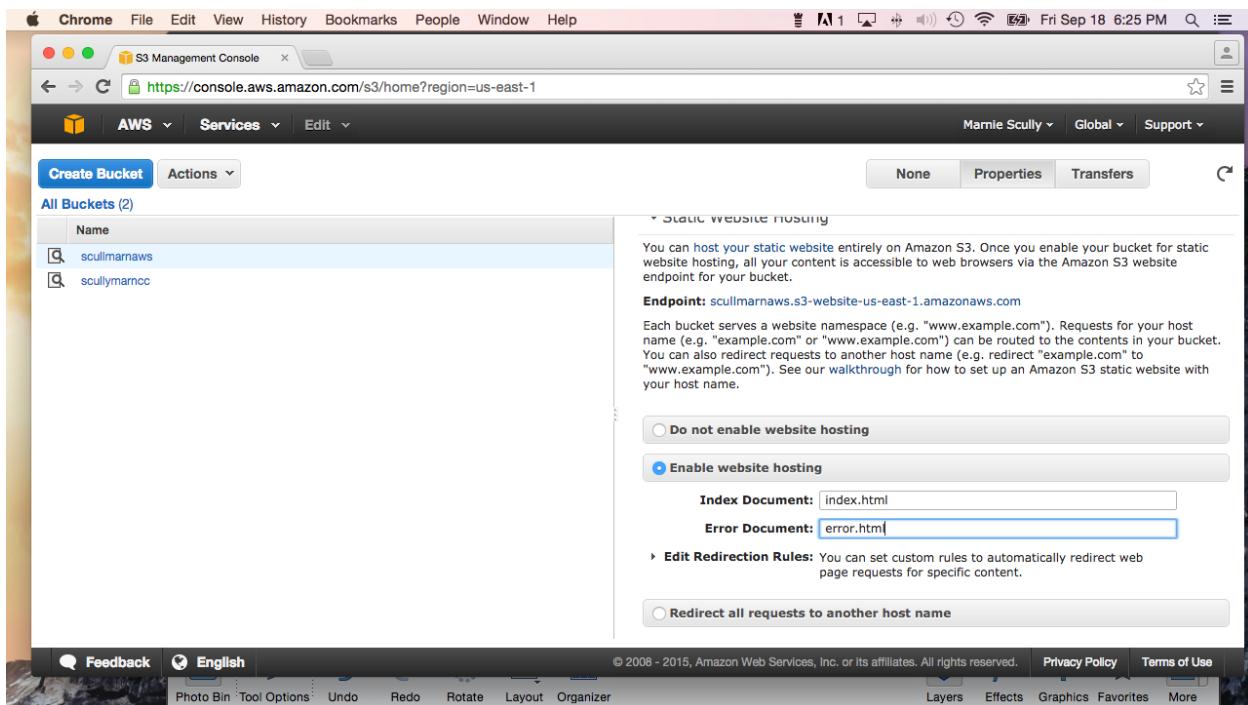
Check your site.



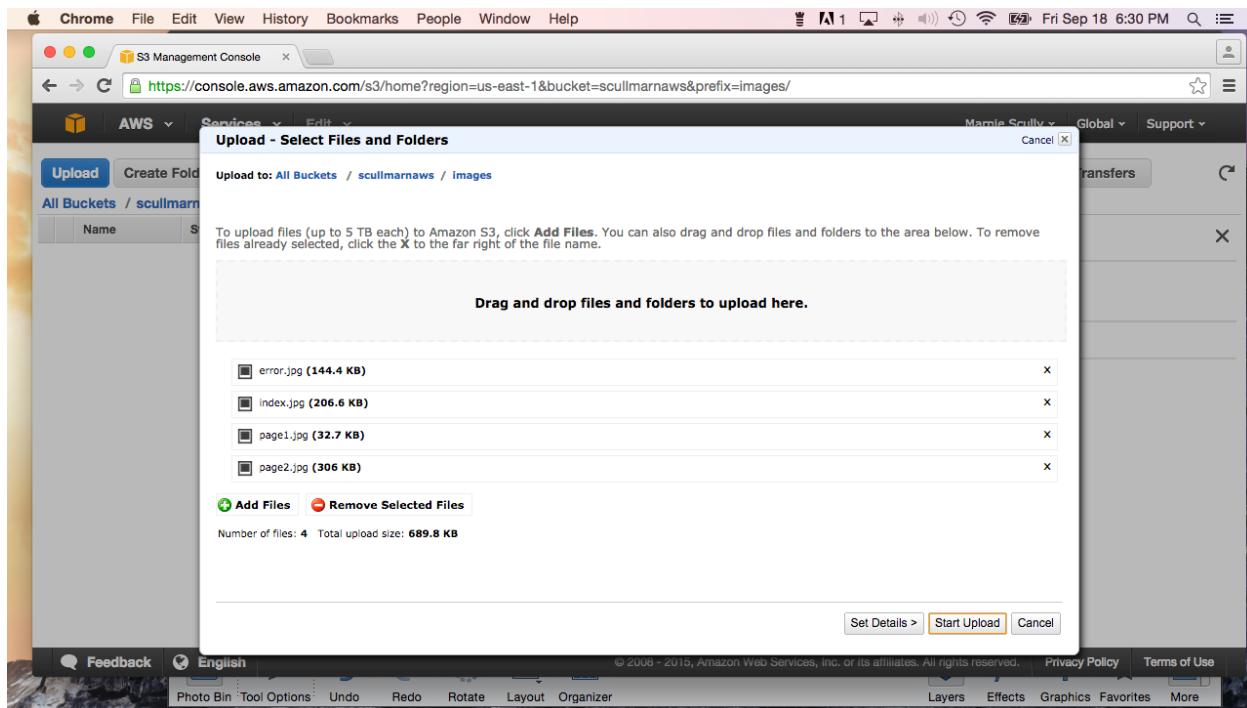
Create a bucket in S3.



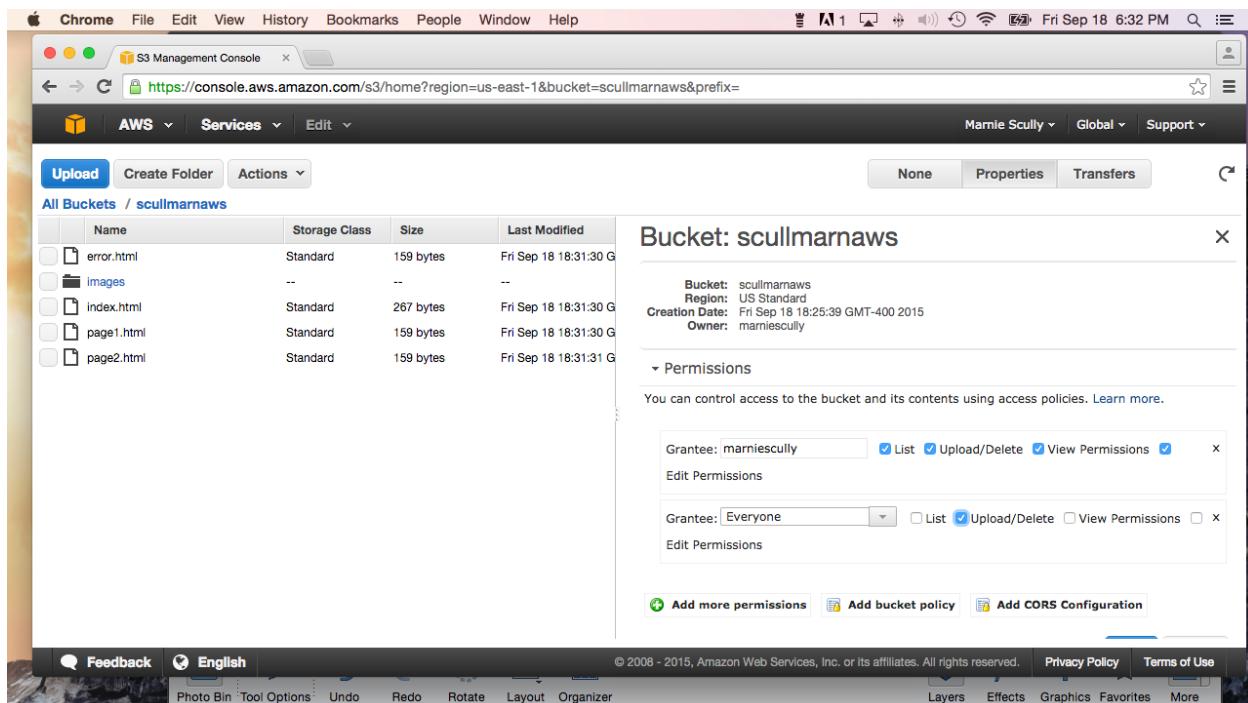
Select that bucket and in the right hand corner of the S3 console page select Properties. Under Properties, select Static Website Hosting and Enable website hosting. Add your index and error files here.



Upload your files into your bucket.



Provide appropriate permissions for public viewing.



Each file and folder must have permissions set to have it publicly available.

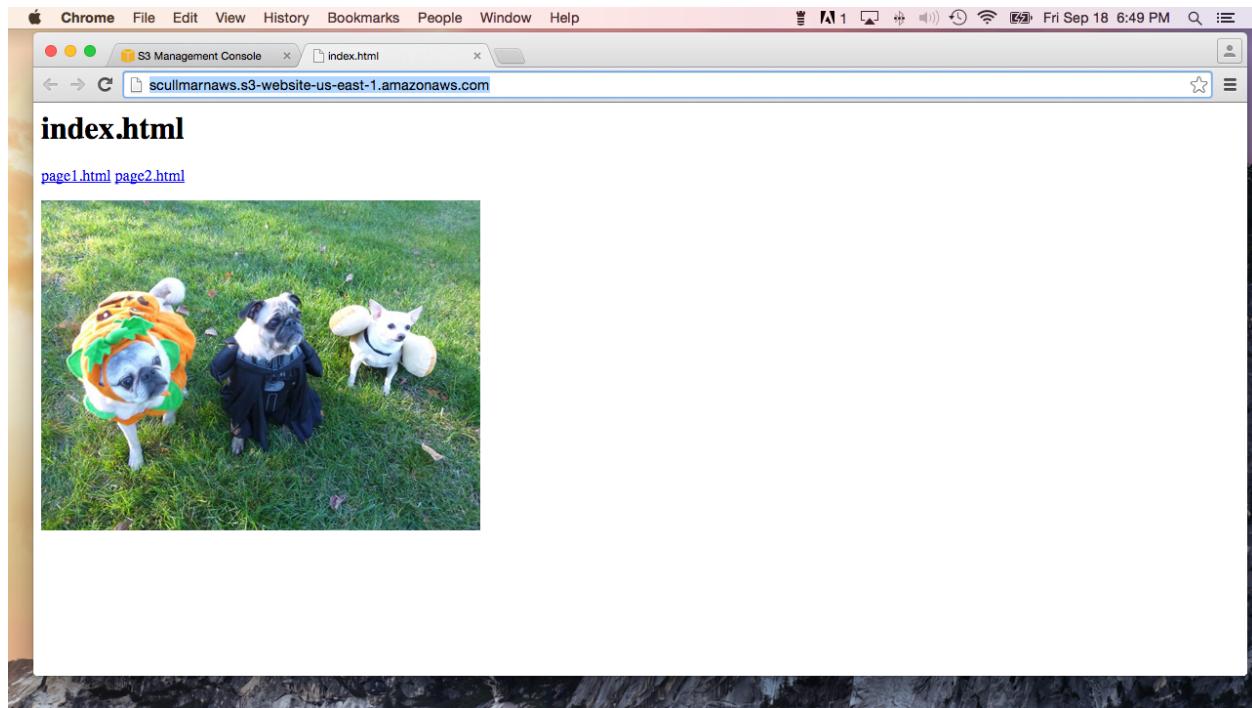
The screenshot shows the AWS S3 Management Console interface. The left sidebar shows the navigation path: All Buckets / scullmarnaws / images. The main content area displays a table of files:

| | Name | Storage Class | Size | Last Modified |
|-------------------------------------|-----------|---------------|----------|-----------------------|
| <input type="checkbox"/> | error.jpg | Standard | 144.4 KB | Fri Sep 18 18:30:39 G |
| <input checked="" type="checkbox"/> | index.jpg | Standard | 206.6 KB | Fri Sep 18 18:30:40 G |
| <input type="checkbox"/> | page1.jpg | Standard | 32.7 KB | Fri Sep 18 18:30:44 G |
| <input type="checkbox"/> | page2.jpg | Standard | 306 KB | Fri Sep 18 18:30:44 G |

On the right side, there is a 'Permissions' panel. It shows two sections: 'Grantee: marniescully' with 'Open/Download' and 'View Permissions' checked, and 'Grantee: Everyone' with the same permissions checked. There is also a link to 'Edit Permissions'. At the bottom of the panel are 'Save' and 'Cancel' buttons.

The screenshot shows the AWS S3 Management Console interface. The left sidebar shows the navigation path: All Buckets / scullmarnaws. The main content area displays a table of files. A context menu is open over the 'images' folder, with the 'Make Public' option highlighted. Other options in the menu include 'Delete', 'Initiate Restore', 'Cut', 'Copy', and 'Properties'. To the right of the table, there is a 'Transfers' section with a checkbox for 'Automatically clear finished transfers'.

Show that your Web site is publicly accessible.



The URL. <http://scullmarnaws.s3-website-us-east-1.amazonaws.com/>

Problem 3: Capture the HTTP requests sent from your Browser and the HTTP responses for uploading a file to one of your S3 buckets. Describe and illustrate the process of capturing HTTP traffic on the tool of your choice.

Run Google Chrome Network Monitoring Tool

The screenshot shows the AWS S3 Management Console interface. On the left, there's a sidebar with options like 'Upload', 'Create Folder', and 'Actions'. The main area displays a list of files in a folder named 'nested_folder'. The bottom right corner of the screen shows the Google Chrome developer tools Network tab, which is monitoring the traffic for the current session. The log shows several requests and responses, including a POST request for file upload.

Upload a file to your S3 bucket

The screenshot shows the 'Upload - Select Files and Folders' dialog in the AWS S3 Management Console. It lists several files in the 'Pictures' folder, with 'IMG_1874.JPG' selected. At the bottom, there are buttons for 'Add Files' and 'Remove Selected'. The developer tools Network tab at the bottom shows a log entry for an XHR finished loading POST request, indicating the file was successfully uploaded.

Capture your traffic showing the upload.

The screenshot shows the AWS S3 Management Console with a file named "IMG_1929.JPG" selected for upload. The "Transfers" section shows an "Starting" status with one upload task listed: "Upload: Uploading IMG_1874.JPG to scullymarncc". The Network tab of the developer tools shows several requests, including "DeliverHttp" and "V4Sign" requests for file parts and metadata. The status bar at the bottom right of the browser window says "Starting".

The screenshot shows the AWS S3 Management Console with the same file selection. The "Transfers" section now shows a completed upload: "Uploaded 1.37 MB (124 KB/sec) 55.74s". The Network tab shows the final "PollHttp" request indicating the upload is finished. The status bar at the bottom right of the browser window says "Completed".

Stop your monitoring. Take a screen snap of your traffic.

The screenshot shows the AWS S3 Management Console. On the left, the 'All Buckets' list shows 'scullymarncc / nested_folder' containing 'IMG_1874.JPG' (2.4 MB) and 'IMG_1929.JPG' (810.2 KB). On the right, the 'Transfers' section shows a list of 25 requests transferred. The Network tab is open, displaying a timeline of network activity. The timeline shows several requests, mostly PollHttp and V4Sign, originating from 'Console.Bun...' and 'Console.Bun...'. The requests are timestamped between 20000 ms and 80000 ms. Below the timeline, a detailed table lists each request with columns for Name, Status, Type, Initiator, Size, Time, and Timeline. The table shows 31 requests transferred in total.

This screenshot is similar to the one above, showing the AWS S3 Management Console with the same bucket contents. The Network tab is open, showing a more detailed timeline of requests. A specific request to 'https://s3-console-us-standard.console.aws.amazon.com/PollHttp/run' is highlighted. The detailed table below the timeline shows 31 requests transferred, with the highlighted request taking 1.7 ms. The table includes columns for Name, Status, Type, Initiator, Size, Time, and Timeline.

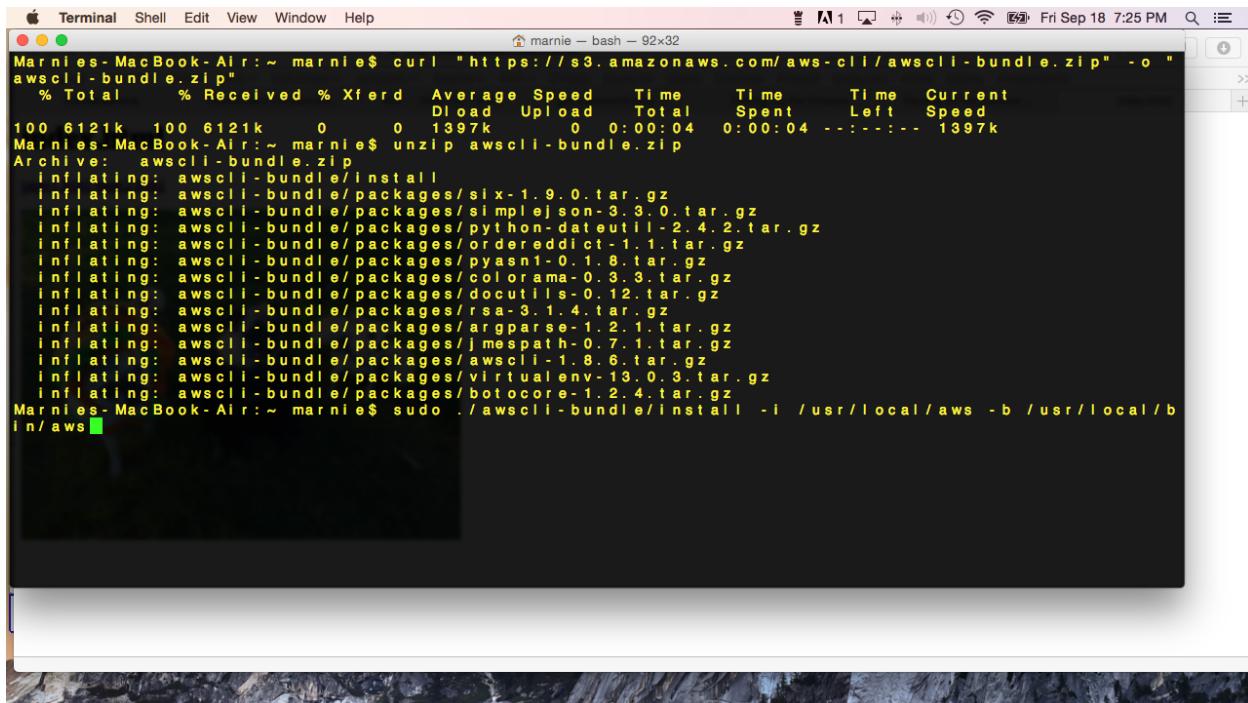
Show your file in your bucket too.

The screenshot shows the AWS S3 Management Console interface. At the top, there's a navigation bar with links for Chrome, File, Edit, View, History, Bookmarks, People, Window, Help, and a user icon. Below the navigation bar, the URL is https://console.aws.amazon.com/s3/home?region=us-east-1&bucket=scullymarncc&prefix=nested_folder/. The main content area displays a table of files in the 'scullymarncc' bucket under the 'nested_folder'. The table has columns for Name, Storage Class, Size, and Path. There are two files listed: 'IMG_1874.JPG' and 'IMG_1929.JPG', both in Standard storage class with a size of 2.4 MB. The total size for the folder is 810.2 MB. Below the table, there's a detailed timeline log showing various requests from the AWS console. The log includes columns for Name, Status, Type, Initiator, Size, Time Latency, and Timeline. The log shows multiple requests for files like 'V4Sign', 'Time?', 'Tattle', and 'PollHttp', all returning 'OK' status with varying response times and sizes. At the bottom of the timeline, it says '31 requests | 21.7 KB transferred'.

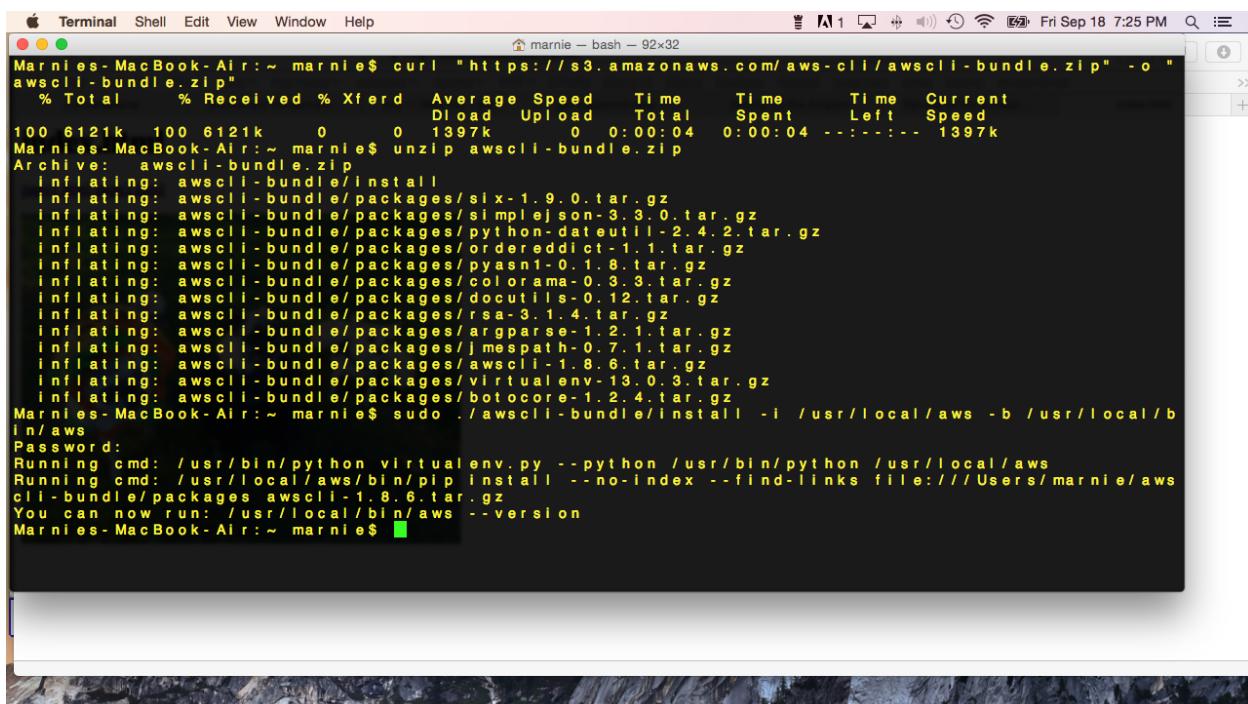
Problem 4:

Verify Java installed

```
j Marnies-MacBook-Air:~ marnie$  
java version "1.8.0_40"  
Java(TM) SE Runtime Environment  
Java HotSpot(TM) 64-Bit Server  
Marnies-MacBook-Air:~ marnie$
```



```
Marnie's-MacBook-Air:~ marnie$ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle.zip" -o "awscli-bundle.zip"
% Total    % Received % Xferd  Average Speed   Time   Time     Time  Current
          Dload  Upload Total Spent   Left Speed
100 6121k  100 6121k    0     0  1397k      0  0:00:04  0:00:04  --:--:-- 1397k
Marnie's-MacBook-Air:~ marnie$ unzip awscli-bundle.zip
Archive: awscli-bundle.zip
  inflating: awscli-bundle/install
  inflating: awscli-bundle/packages/six-1.9.0.tar.gz
  inflating: awscli-bundle/packages/simplejson-3.3.0.tar.gz
  inflating: awscli-bundle/packages/python-dateutil-2.4.2.tar.gz
  inflating: awscli-bundle/packages/ordereddict-1.1.tar.gz
  inflating: awscli-bundle/packages/pyasn1-0.1.8.tar.gz
  inflating: awscli-bundle/packages/colorama-0.3.3.tar.gz
  inflating: awscli-bundle/packages/docutils-0.12.tar.gz
  inflating: awscli-bundle/packages/rsa-3.1.4.tar.gz
  inflating: awscli-bundle/packages/argparse-1.2.1.tar.gz
  inflating: awscli-bundle/packages/jmespath-0.7.1.tar.gz
  inflating: awscli-bundle/packages/awscli-1.8.6.tar.gz
  inflating: awscli-bundle/packages/virtualenv-13.0.3.tar.gz
  inflating: awscli-bundle/packages/botocore-1.2.4.tar.gz
Marnie's-MacBook-Air:~ marnie$ sudo ./awscli-bundle/install -i /usr/local/aws -b /usr/local/bin/aws
```



```
Marnie's-MacBook-Air:~ marnie$ curl "https://s3.amazonaws.com/aws-cli/awscli-bundle.zip" -o "awscli-bundle.zip"
% Total    % Received % Xferd  Average Speed   Time   Time     Time  Current
          Dload  Upload Total Spent   Left Speed
100 6121k  100 6121k    0     0  1397k      0  0:00:04  0:00:04  --:--:-- 1397k
Marnie's-MacBook-Air:~ marnie$ unzip awscli-bundle.zip
Archive: awscli-bundle.zip
  inflating: awscli-bundle/install
  inflating: awscli-bundle/packages/six-1.9.0.tar.gz
  inflating: awscli-bundle/packages/simplejson-3.3.0.tar.gz
  inflating: awscli-bundle/packages/python-dateutil-2.4.2.tar.gz
  inflating: awscli-bundle/packages/ordereddict-1.1.tar.gz
  inflating: awscli-bundle/packages/pyasn1-0.1.8.tar.gz
  inflating: awscli-bundle/packages/colorama-0.3.3.tar.gz
  inflating: awscli-bundle/packages/docutils-0.12.tar.gz
  inflating: awscli-bundle/packages/rsa-3.1.4.tar.gz
  inflating: awscli-bundle/packages/argparse-1.2.1.tar.gz
  inflating: awscli-bundle/packages/jmespath-0.7.1.tar.gz
  inflating: awscli-bundle/packages/awscli-1.8.6.tar.gz
  inflating: awscli-bundle/packages/virtualenv-13.0.3.tar.gz
  inflating: awscli-bundle/packages/botocore-1.2.4.tar.gz
Marnie's-MacBook-Air:~ marnie$ sudo ./awscli-bundle/install -i /usr/local/aws -b /usr/local/bin/aws
Password:
Running cmd: /usr/bin/python virtualenv.py --python /usr/bin/python /usr/local/awscli-bundle/packages/awscli-1.8.6.tar.gz
Running cmd: /usr/local/aws/bin/pip install --no-index --find-links file:///Users/marnie/awscli-bundle/packages/awscli-1.8.6.tar.gz
You can now run: /usr/local/bin/aws --version
Marnie's-MacBook-Air:~ marnie$
```

A screenshot of a Mac OS X desktop. In the foreground, a Terminal window is open with the command `aws configure`. The output shows AWS Access Key ID, AWS Secret Access Key, Default region name, and AWS_DEFAULT_REGION set to us-east-1. Behind the Terminal, a browser window displays a HAR (HTTP Archive) file analysis interface. The HAR file is titled "Secrets of the Browser Dev..." and includes sections for "Save at:" (with options for Firebug format, chrome, and Permalinks), "Bug developer and the creator of the HAR used for this purpose.", and social sharing links.

```
marnie$ aws configure
AWS Access Key ID [None]: AKI AI WURWCPTW4VVYLQ
AWS Secret Access Key [None]: Z6VtdCaul v54w00pxu1QKQeoqc5wZxJPXk2w93Sb
Default region name [None]: us-east-1
Default output format [None]: table
marnie$ export AWS_ACCESS_KEY_ID=AKI AI WURWCPTW4VVYLQ
marnie$ export AWS_SECRET_ACCESS_KEY=Z6VtdCaul v54w00pxu1QKQeoqc5wZxJPXk2w93Sb
marnie$ export AWS_DEFAULT_REGION=us-east-1
marnie$
```

Create a new key pair using AWS command line (CLI) tools.

A screenshot of a Mac OS X desktop showing a Terminal window. The command `aws ec2 create-key-pair --key-name CliKeyPair --query 'KeyMaterial' --output text > CliKeyPair.pem` is being run. The terminal window title is "marnie - bash - 86x24".

```
marnie$ aws ec2 create-key-pair --key-name CliKeyPair --query 'KeyMaterial' --output text > CliKeyPair.pem
```

A screenshot of a Mac OS X desktop. In the foreground, a Terminal window is open with the command `aws ec2 describe-key-pairs --key-name CliKeyPair`. The output shows a single key pair named "CliKeyPair" with a specific key fingerprint. Behind the Terminal, a browser window displays a HAR (HTTP Archive) file analysis interface. The HAR file is titled "Secrets of the Browser Dev..." and includes sections for "Save at:" (with options for Firebug format, chrome, and Permalinks), "Bug developer and the creator of the HAR used for this purpose.", and social sharing links.

```
marnie$ aws ec2 describe-key-pairs --key-name CliKeyPair
{
    "DescribeKeyPairs": {
        "KeyPairs": [
            {
                "KeyFingerprint": "3d:55:9f:ff:43:85:0e:87:1d:9d:88:f0:1e:99:fd:19:23:53:16:d8",
                "KeyName": "CliKeyPair"
            }
        ]
    }
}
```

Subsequently, use that new key to create a new Linux instance ami-008db468 .
 Your instance should be of the EBS root type.

```
marnie - bash - 86x36
marnie$ aws ec2 run-instances --image-id ami-008db468 --count 1
--instance-type t1.micro --key-name CliKeyPair --security-groups RDPGroup
+-----+
| RunInstances
+-----+
| OwnerId
| ReservationId
+-----+
| Instances
+-----+
| AmiLaunchIndex
| Architecture
| ClientToken
| EbsOptimized
| Hypervisor
| ImageId
| InstanceId
| InstanceType
| KernelId
| KeyName
| LaunchTime
| PrivateDnsName
| PrivateIpAddress
| PublicDnsName
| RootDeviceName
| RootDeviceType
| SourceDestCheck
| StateTransitionReason
| SubnetId
| VirtualizationType
| VpcId
+-----+
| Monitoring
+-----+
| State
| disabled
+-----+
marnie - bash - 86x36
+-----+
| InstanceType
| KernelId
| KeyName
| LaunchTime
| PrivateDnsName
| PrivateIpAddress
| PublicDnsName
| PublicIpAddress
| RootDeviceName
| RootDeviceType
| SourceDestCheck
| StateTransitionReason
| SubnetId
| VirtualizationType
| VpcId
+-----+
| BlockDeviceMappings
+-----+
| DeviceName
| /dev/sda1
+-----+
| Ebs
+-----+
| AttachTime
| 2015-09-19T00:45:41.000Z
| DeleteOnTermination
| True
| Status
| attached
| VolumeId
| vol-9633d976
+-----+
| Monitoring
```

Tell us in which region you are working in. (us-east-1b)

```
  PublicDnsName |  ec2-54-175-31-34.compute-1.amazonaws.com
  PublicIp      |  54.175.31.34
+-----+
| Placement
+-----+
| AvailabilityZone | us-east-1b
| GroupName
| Tenancy
+-----+
| SecurityGroups
```

While the instance is starting, inquire about its health and status using AWS CLI tools.

```
marnie - bash - 86x36
  DescribeInstanceStatus
+-----+
| InstanceStatuses
+-----+
| AvailabilityZone | InstanceId
+-----+
| us-east-1b | i-fbcd92b
+-----+
| InstanceState
+-----+
| Code | Name
+-----+
| 16 | running
+-----+
| InstanceStatus
+-----+
| Status | ok
+-----+
| Details
+-----+
| Name | Status
+-----+
| reachability | passed
+-----+
| SystemStatus
+-----+
| Status | ok
+-----+
| Details
+-----+
| Name | Status
+-----+
| reachability | passed
+-----+
Marnie's MacBook Air: ~ marnie$
```

Also check the status of the instance in your AWS Console.

The screenshot shows the AWS Management Console with the EC2 Instances page open. A single instance, `i-febcd92b`, is listed as `t1.micro` in the `Instance Type` column, located in `us-east-1b` in the `Availability Zone` column, and is currently `running` in the `Instance State` column. The `Status Checks` column shows `2/2 checks ...` and `None`. The `Public DNS` column shows `ec2-54-175-31-34.com`. Below the instance table, there are three CloudWatch metrics graphs: `CPU Utilization (Percent)`, `Disk Reads (Bytes)`, and `Disk Read Operations (Operations)`. The `CPU Utilization` graph shows a sharp drop from approximately 100% to 0% between 00:30 and 01:00 on 9/19. The `Disk Reads` and `Disk Read Operations` graphs show very low activity, near zero.

Terminate your instance, using AWS CLI Tools.

```
marnie$ aws ec2 terminate-instances --instance-ids i-febcd92b
marnie$ Terminating instances
+-- TerminatingInstances
+-- InstanceId
+-- i-febcd92b
+-- CurrentState
+-- Code      Name
+-- 32       shutting-down
+-- PreviousState
+-- Code      Name
+-- 16       running
marnie$
```

Problem 5:

Create a Windows instance ami-cd9339a6 with: 64-bit, root device type: EBS (or you can use Instance store), t1.micro using AWS Command Line tools.

```
marnie$ aws ec2 run-instances --image-id ami-cd9339a6 --count 1
--instance-type t1.micro --key-name CLIKeyPair --security-groups RDPGroup
[...]
RunInstances
+-----+
| OwnerId | 413513583861 |
|-----+
| ReservationId | r-8a58f577 |
|-----+
| Instances |
|-----+
| AmiLaunchIndex | 0 |
| Architecture | x86_64 |
| ClientToken | False |
| EbsOptimized | xen |
| Hypervisor | ami-cd9339a6 |
| ImageId | i-98ff9a4d |
| InstanceId | t1.micro |
| KeyName | CLIKeyPair |
| LaunchTime | 2015-09-19T01:25:45.000Z |
| Platform | windows |
| PrivateDnsName | ip-172-31-0-146.ec2.internal |
| PrivateIpAddress | 172.31.0.146 |
| PublicDnsName | |
| RootDeviceName | /dev/sda1 |
| RootDeviceType | ebs |
| SourceDestCheck | True |
| StateTransitionReason | |
| SubnetId | subnet-0becce57c |
| VirtualizationType | hvm |
| VpcId | vpc-91498915 |
|-----+
| Monitoring |
|-----+
| State | disabled |
|-----+
```

Add a policy for RDP 3389 if it is not defaulted in security group for that AMI.

```
marnie$ aws ec2 describe-key-pairs --key-name CLIKeyPair
[...]
DescribeKeyPairs
+-----+
| KeyFingerprint | 3d:55:9f:ff:43:85:0e:87:1d:9d:88:f0:1e:99:fd:19:23:53:16:d8 |
| KeyName | CLIKeyPair |
|-----+
marnie$ aws ec2 create-security-group --group-name RDPGroup --description "RDP group"
[...]
CreateSecurityGroup
+-----+
| GroupId | sg-52f9ed35 |
|-----+
```

```
marnie$ aws ec2 describe-security-groups --group-names RDPGroup
{
    "DescribeSecurityGroups": {
        "SecurityGroups": [
            {
                "Description": "RDP group",
                "GroupId": "sg-52f9ed35",
                "GroupName": "RDPGroup",
                "OwnerId": "413513583881",
                "VpcId": "vpc-81498915",
                "IpPermissionsEgress": {
                    "IpProtocol": "-1"
                },
                "IpRanges": [
                    {
                        "CidrIp": "0.0.0.0/0"
                    }
                ]
            }
        ]
    }
}
```

```
marnie$ aws ec2 authorize-security-group-ingress --group-name RDGroup --protocol tcp --port 3389 --cidr 203.0.113.0/24
Marnie$
```

The screenshot shows a Mac OS X desktop with a Terminal window and a browser window. The Terminal window is titled "marnie - bash - 86x24" and displays the output of the command "aws ec2 describe-security-groups". The browser window is titled "Untitled - Edited" and shows a detailed view of the security group configuration, including its description ("RDP group"), group ID ("sg-52f9ed35"), owner ID ("413513563861"), and VPC ID ("vpc-914989f5"). It also lists the IP permissions, which include a rule for port 3389 using TCP. The browser interface includes a "Share" button with social media icons.

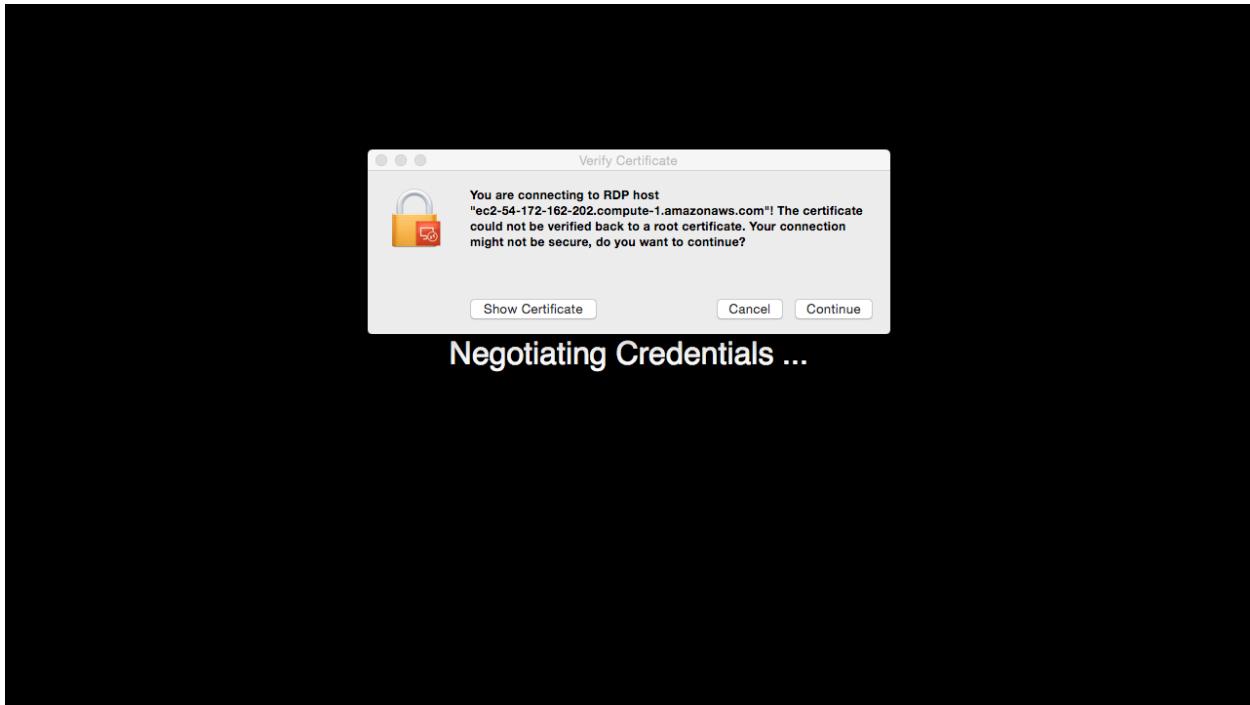
```
marnie - bash - 86x24
aws ec2 describe-security-groups
{
    "SecurityGroups": [
        {
            "Description": "RDP group",
            "GroupId": "sg-52f9ed35",
            "GroupName": "RDPGroup",
            "OwnerId": "413513563861",
            "VpcId": "vpc-914989f5",
            "IpPermissions": [
                {
                    "FromPort": 3389,
                    "IpProtocol": "tcp",
                    "ToPort": 3389
                }
            ],
            "IpRanges": [
                {
                    "CidrIp": "203.0.113.0/24"
                }
            ],
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1"
                }
            ]
        }
    ]
}
```

Request remote password using AWS CLI tools.

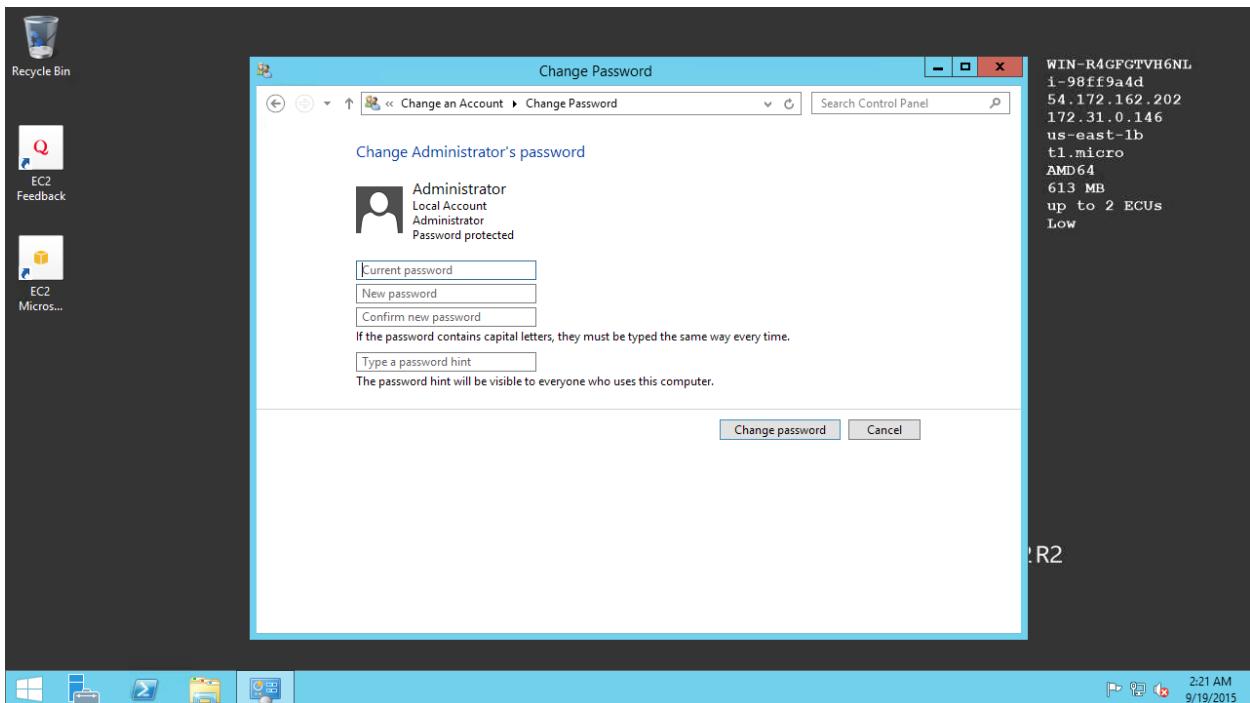
The screenshot shows a Mac OS X desktop with a Terminal window. The terminal is titled "marnie - bash - 86x36" and displays the command "aws ec2 get-password-data --instance-id i-98ff9a4d --private-launch-key CliKeyPair.pem". The output shows the "Get PasswordData" section, which includes the instance ID ("i-98ff9a4d") and the password data ("aa72QzqS*7"). The timestamp is listed as "2015-09-19T01:29:13.000Z".

```
Marnie's-MacBook-Air:~ marnie$ aws ec2 get-password-data --instance-id i-98ff9a4d --private-launch-key CliKeyPair.pem
{
    "GetPasswordData": {
        "InstanceId": "i-98ff9a4d",
        "PasswordData": "aa72QzqS*7",
        "Timestamp": "2015-09-19T01:29:13.000Z"
    }
}
Marnie's-MacBook-Air:~ marnie$
```

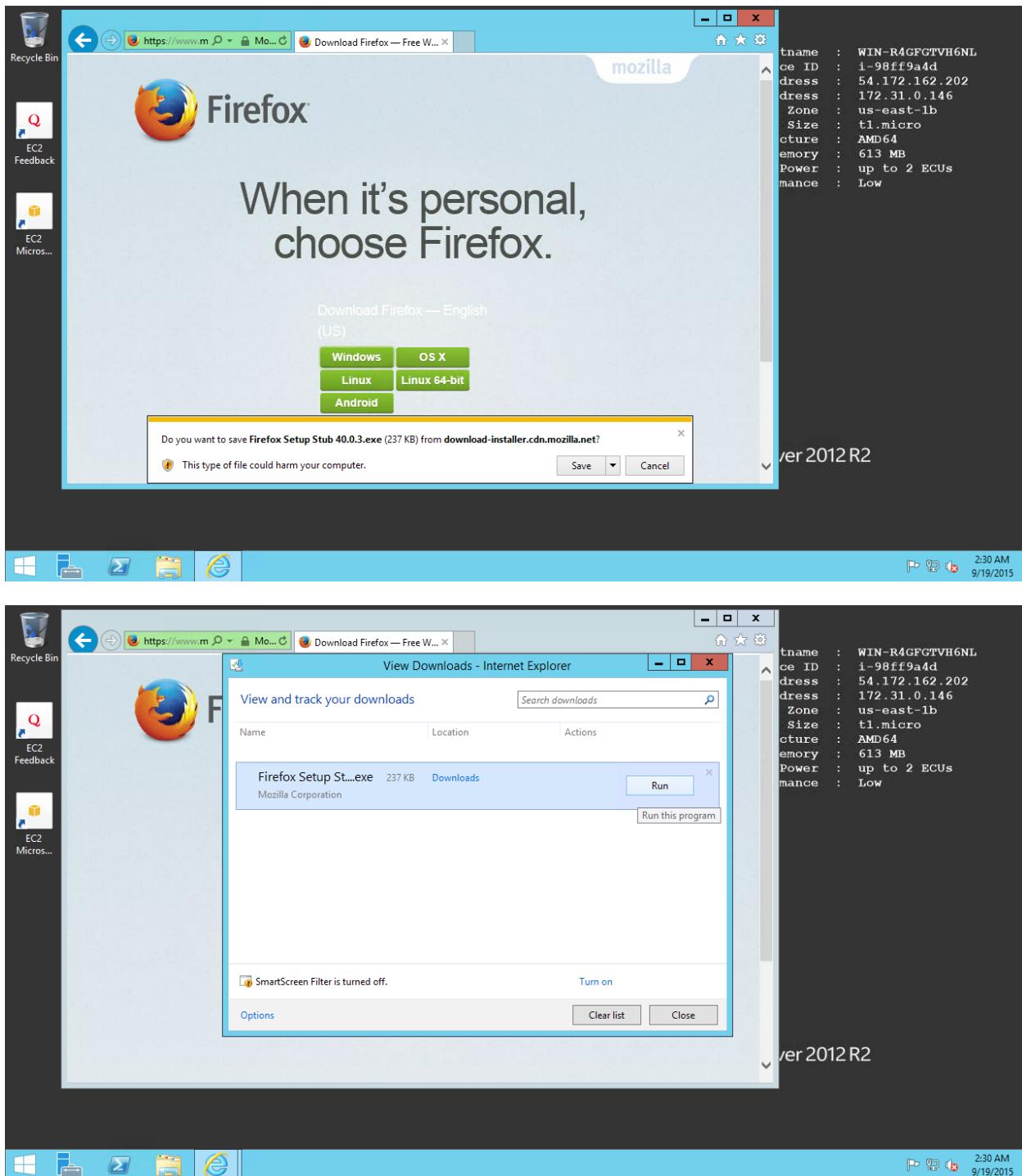
You have to use Remote Desktop Protocol and port 3389 to connect to your instance.



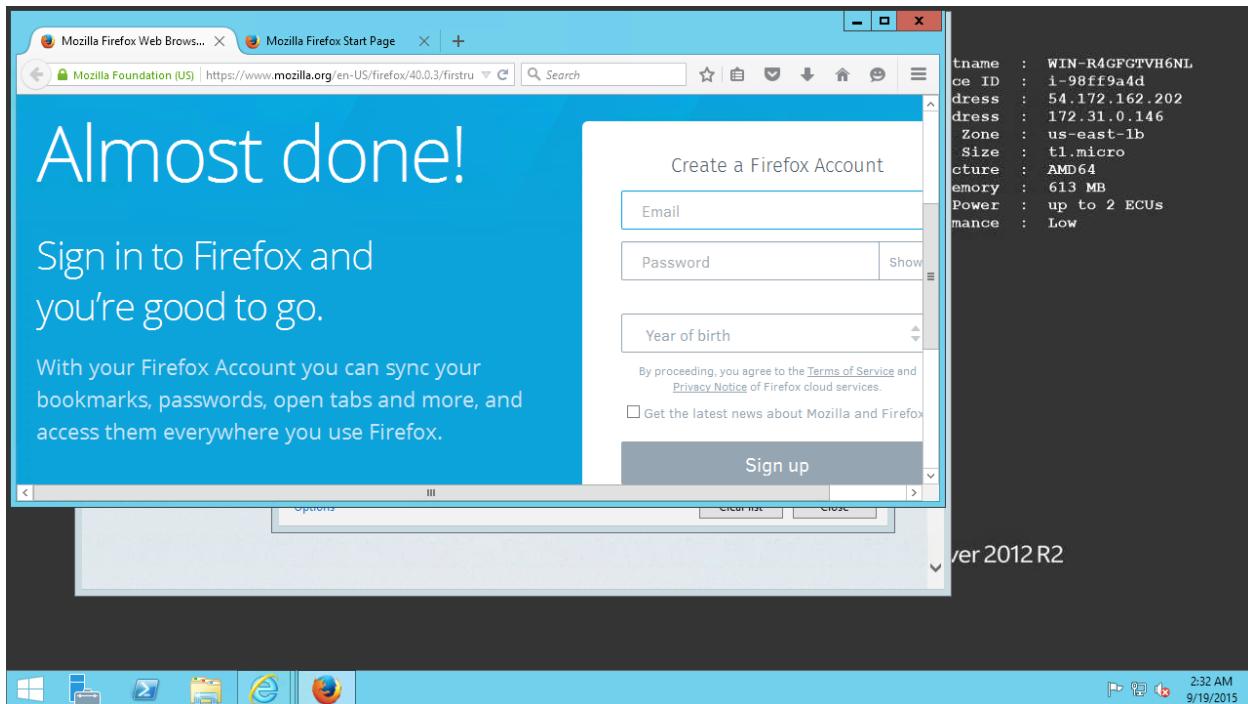
Change the password to something you could remember.



Next, open a browser and download any software you wish.



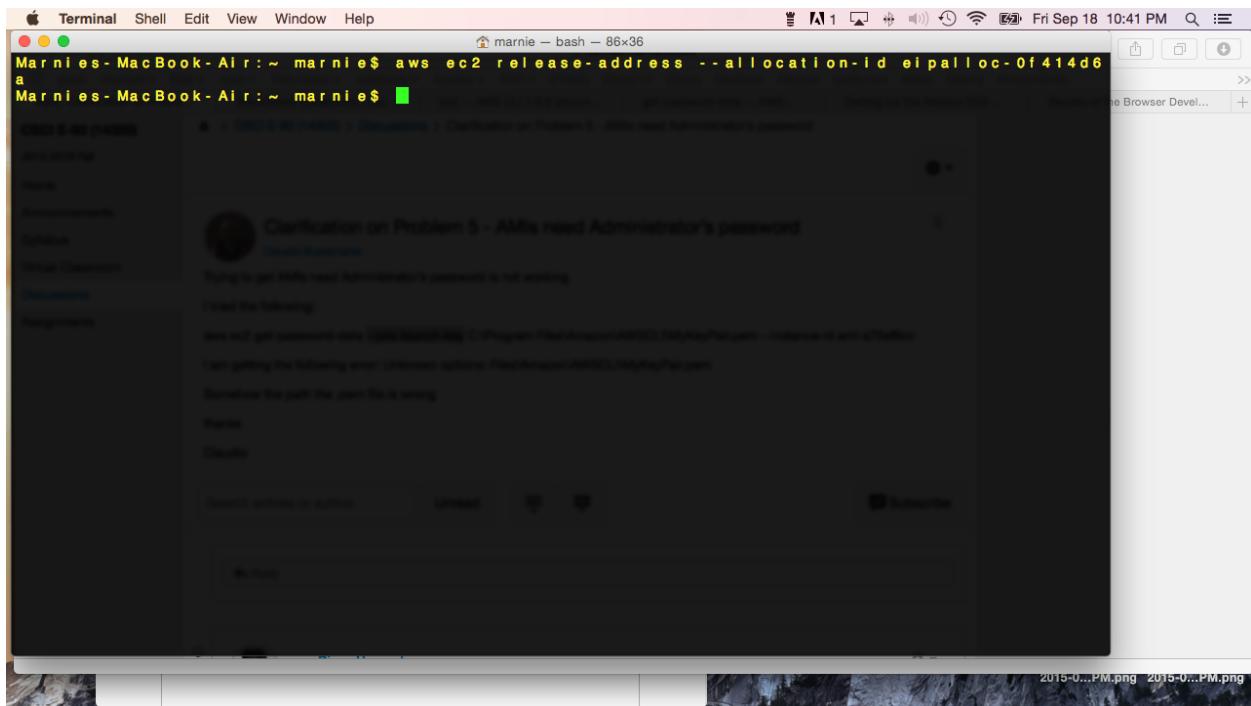
Demonstrate your added software in a screen shot.



Acquire an Elastic IP address (could you do it with AWS Tools?) and associate it with your instance.

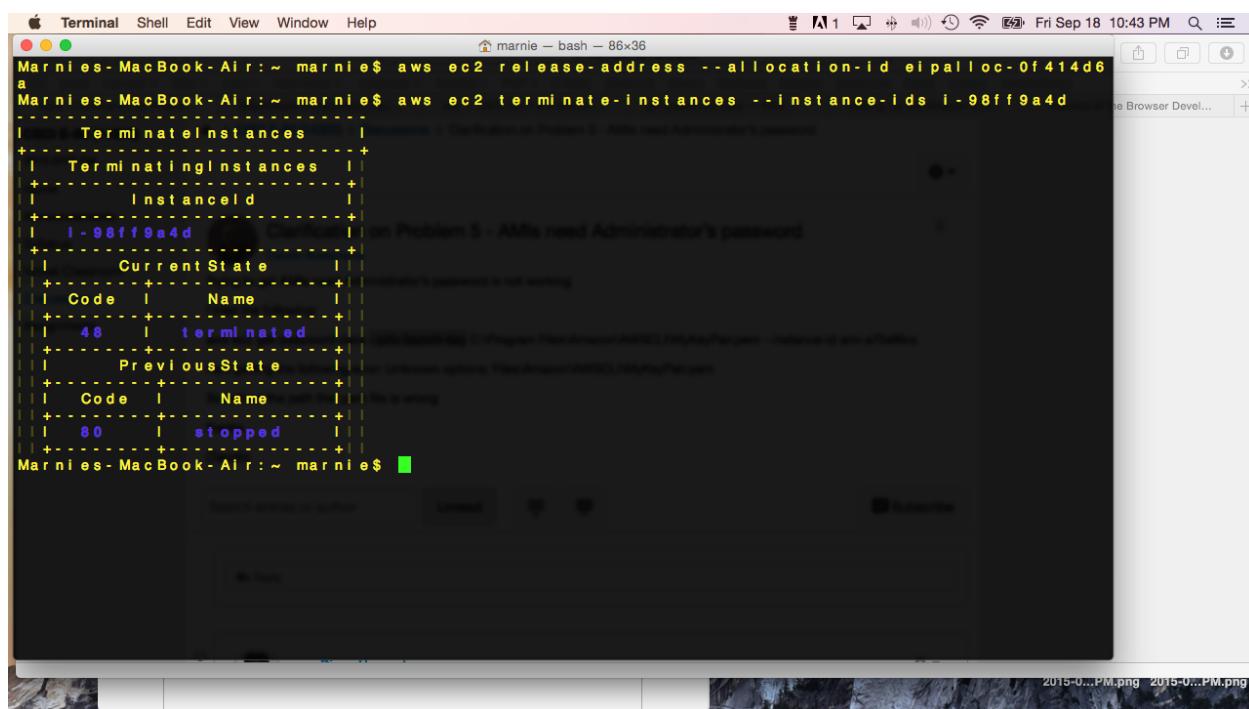
```
marnie@Marnies-MacBook-Air: ~ marnie$ aws ec2 allocate-address
-----
|           AllocateAddress           |
+-----+-----+-----+
| AllocationId | Domain | PublicIp |
+-----+-----+-----+
| elipalloc-0f414d6a | vpc | 52.21.84.111 |
+-----+-----+-----+
Marnie@Marnie-MacBook-Air: ~ marnie$ aws ec2 associate-address --instance-id i-98ff9a4d --public-ip 52.21.84.111
-----
|           AssociateAddress          |
+-----+-----+
| AssociationId | elipassoc-c98419af |
+-----+-----+
Marnie@Marnie-MacBook-Air: ~ marnie$
```

Release your Elastic IP address.



A screenshot of a Mac OS X desktop. At the top, there's a menu bar with 'Terminal', 'Shell', 'Edit', 'View', 'Window', and 'Help'. The system status bar shows the date and time as 'Fri Sep 18 10:41 PM'. In the center, a terminal window titled 'marnie — bash — 86x36' is open, showing the command 'aws ec2 release-address --allocation-id eipalloc-0f414d6'. Below the terminal is a browser window displaying a page about 'Clarification on Problem 5 - AMIs need Administrator's password'. The page contains text and a 'Submit' button. The desktop background is a dark image of a landscape.

Terminate your instance using AWS CLI Tools. Capture all dialogues.



A screenshot of a Mac OS X desktop. At the top, there's a menu bar with 'Terminal', 'Shell', 'Edit', 'View', 'Window', and 'Help'. The system status bar shows the date and time as 'Fri Sep 18 10:43 PM'. In the center, a terminal window titled 'marnie — bash — 86x36' is open, showing the command 'aws ec2 terminate-instances --instance-ids i-98ff9a4d'. Below the terminal is a browser window displaying a page about 'Clarification on Problem 5 - AMIs need Administrator's password'. The page contains text and a 'Submit' button. The desktop background is a dark image of a landscape.

Problem 6:

Show your Instances are stopped from AWS Console.

The screenshot shows the AWS EC2 Management Console interface. On the left, the navigation pane is open with the 'Instances' section selected. In the main content area, a table lists instances. One instance, with the ID i-98ff9a4d, is shown in blue and has a red 'stopped' status indicator. Below the table, several CloudWatch metrics are displayed as line graphs: CPU Utilization (Percent), Disk Reads (Bytes), Disk Read Operations (Operations), Disk Writes (Bytes), Disk Write Operations (Operations), and Network In (Bytes). The CPU Utilization graph shows a sharp spike from 0% to approximately 80% between 9/19 02:00 and 9/19 02:30. The other metrics show minimal activity.

Show your Month-to-Date Top Services by Spend (cost amounts) for your Amazon Services (EC2, S3, ...) from your Billing & Cost Management Page.

The screenshot shows the AWS Billing Management Console. The left sidebar includes options like Bills, Cost Explorer, Budgets, Payment Methods, and Payment History. The main dashboard features a 'Spend Summary' section with a large '\$0.00' and a chart showing costs for 'Last Month' and 'Month-to-Date'. To the right, a 'Month-to-Date Spend by Service' chart is displayed, showing a single entry: 'No Amount Due'. Below this chart is a table titled 'Month-to-Date Top Services by Spend' which lists services and their amounts: DataTransfer (\$0.00), kms (\$0.00), EC2 (\$0.00), and Tax (\$0.00). A total row shows 'Total' with '\$0.00'.