



lab title

Setting up a NodeJS Development Environment on AWS EC2

V1.00



Course title

AWS Certified Developer Associate



Table of Contents

Contents

Table of Contents.....	1
About the Lab	1
Creating an IAM User, Group and Role	1
Creating a Security Group.....	1
Creating an EC2 instance.....	1
Connecting to your EC2 instance using SSH.....	1
Transferring files to an EC2 instance using SFTP	1
Setting Up the GIT Version Control System	1
Integrating SFTP EC2 Connection and GIT with an IDE.....	1

About the Lab



These lab notes are to support the instructional videos on Setting up a NodeJS Server on EC2 in the BackSpace AWS Certified Developer course.

We will first use the Identity and Access Management (IAM) service to create a user and a developers group for user. Permissions will be set for the developers group and users inside the group will inherit the permissions. We will also create a role with permissions that will allow our EC2 Linux server to access AWS resources within the account.

We will then:

- Create an EC2 Linux instance and connect to that instance using SSH.
- Transfer files using SFTP.
- Integrate SFTP and Git version control into an IDE.

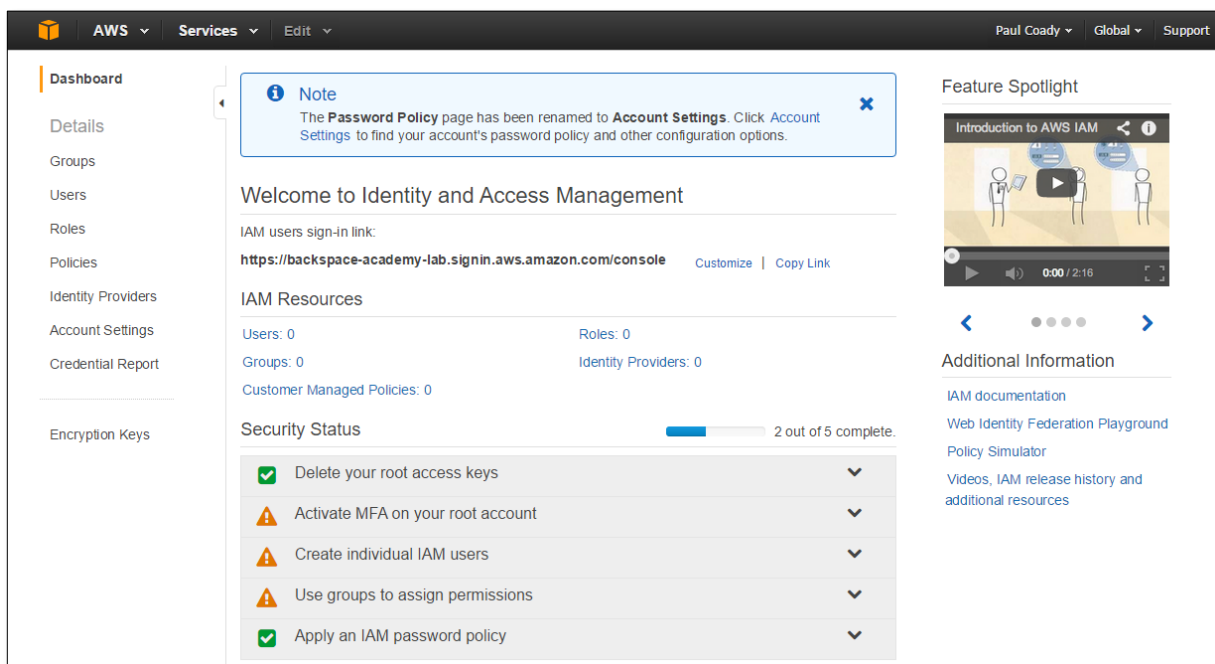
Please note that AWS services change on a weekly basis and it is extremely important you check the version number on this document to ensure you have the latest version with any updates or corrections.



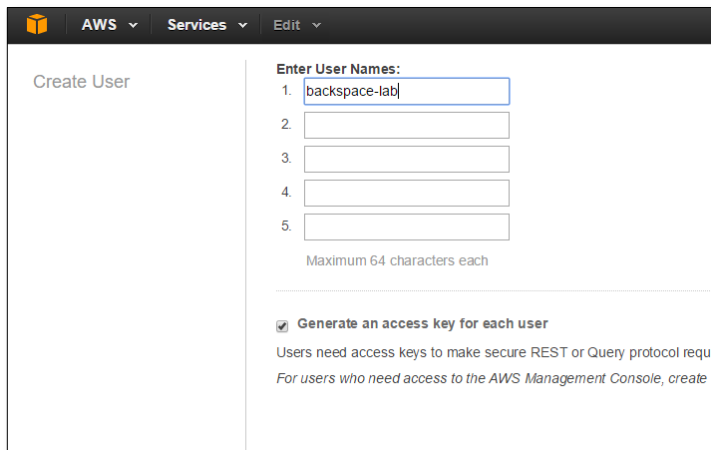
Creating an IAM User, Group and Role

In this section we will use the Identity and Access Management (IAM) service to create a user and a developers group for user. Permissions will be set for the developers group and users inside the group will inherit the permissions. We will also create a role with permissions that will allow our EC2 Linux server to access AWS resources within the account.

Select the IAM Console



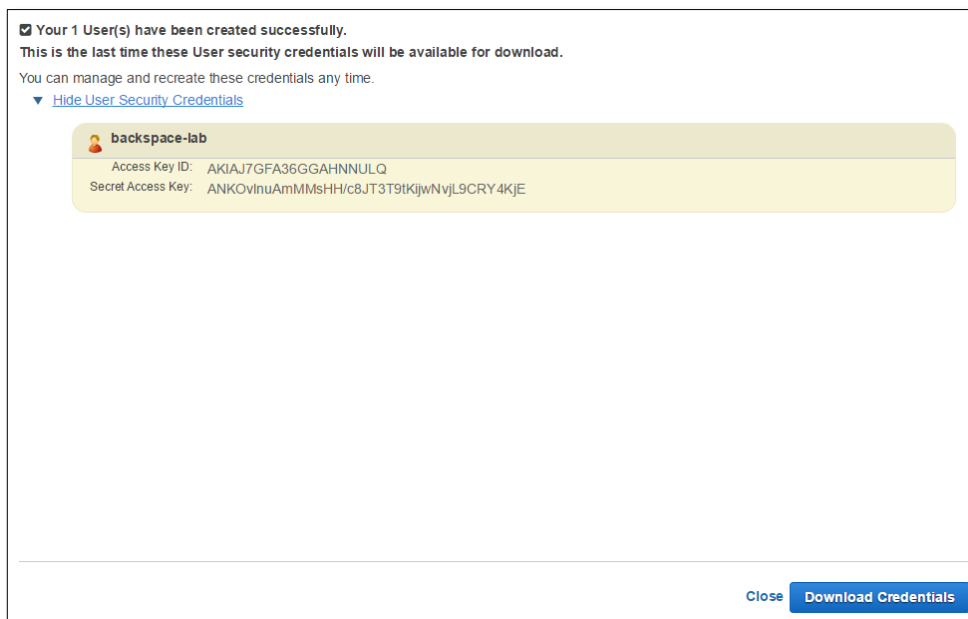
Click “Users” then “Create New Users”. Call the user backspace-lab.



The screenshot shows the AWS IAM 'Create User' console. The 'Enter User Names' field contains 'backspace-lab'. The 'Generate an access key for each user' checkbox is checked. The console also displays a note about access keys and a link to 'Hide User Security Credentials'.

Click Create.

Click Download Credentials. Save this file somewhere we will need it later.



The screenshot shows the AWS IAM 'Download Credentials' console. It displays the user 'backspace-lab' and its Access Key ID and Secret Access Key. The console also includes a 'Close' button and a 'Download Credentials' button.

User	Access Key ID	Secret Access Key
backspace-lab	AKIAJ7GFA36GGAHNNULQ	ANKOvinuAmMMsHH/c8JT3T9tKjwNvjL9CRY4KjE

Click Close.

Click on “Groups” then select “Create New Group”. Call the group Developers.

The screenshot shows the 'Set Group Name' step of the 'Create New Group Wizard'. On the left, a sidebar lists the steps: 'Step 1: Group Name' (active), 'Step 2: Attach Policy', and 'Step 3: Review'. The main area is titled 'Set Group Name' and includes the instruction 'Specify a group name. Group names can be edited any time.' Below this, there is a 'Group Name' label and a text input field containing 'Developers'. A note below the input field states: 'Example: Developers or ProjectAlpha' and 'Maximum 128 characters'.

Click “Next Step”.

Search for Administrator Access and select.

The screenshot shows the 'Attach Policy' step. At the top, it says 'Select up to two policies to attach to the group.' Below this is a filter section with 'Filter: Policy Type' and a search input containing 'admin', which shows 'Showing 5 results'. A table lists the policies:

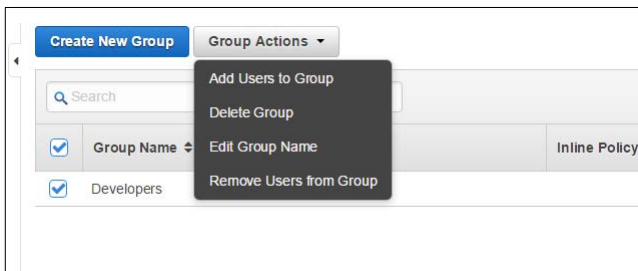
	Policy Name	Attached Entities	Creation Time	Edited Time
<input checked="" type="checkbox"/>	AdministratorAccess	0	2015-02-07 05:39 UTC+1100	2015-02-07 05:39 UTC+...
<input type="checkbox"/>	AmazonAPIGatewayAdmini...	0	2015-07-10 03:34 UTC+1000	2015-07-10 03:34 UTC+...
<input type="checkbox"/>	AmazonWorkSpacesApplica...	0	2015-04-10 00:03 UTC+1000	2015-04-10 00:03 UTC+...
<input type="checkbox"/>	ServiceCatalogAdmin	0	2015-07-10 03:19 UTC+1000	2015-07-10 03:19 UTC+...
<input type="checkbox"/>	ServiceCatalogAdminReadO...	0	2015-07-10 03:21 UTC+1000	2015-07-10 03:21 UTC+...

At the bottom right, there are three buttons: 'Cancel', 'Previous', and 'Next Step'.

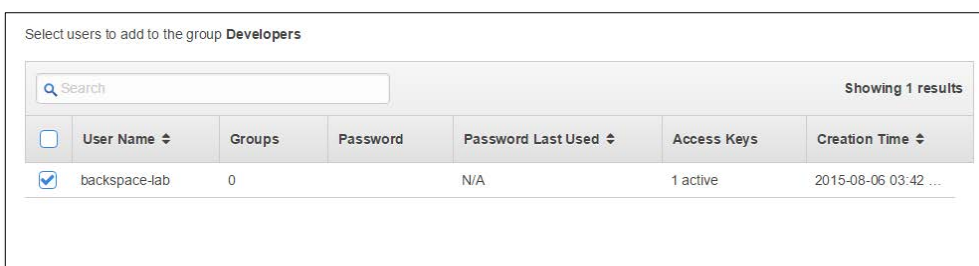
Click “Next Step”.

Click “Create Group”.

Select the new group and select “Add users to group” from Group Actions.



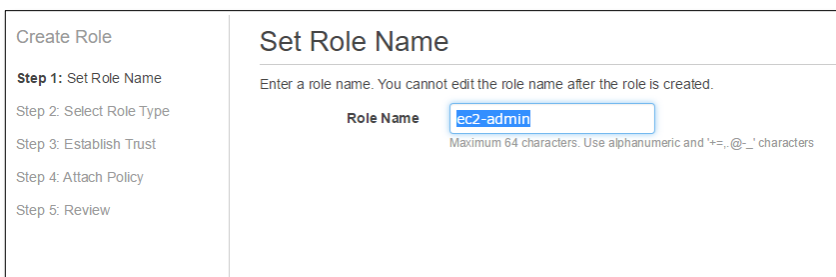
Select the backspace-lab user and click “Add users”



The user is now added to the Developers group and has inherited administrator access from the group.

Click on “Roles” and select “Create new role”.

Call the role ec2-admin.



Click “Next Step”.

Select “Amazon EC2 - Allows EC2 instances to call AWS services on your behalf.”



Search for Administrator Access and select.






Attach Policy

Select up to two policies to attach to the role.

Filter: Policy Type

admin

Showing 5 results

		Policy Name	Attached Entities	Creation Time	Edited Time
<input checked="" type="checkbox"/>		AdministratorAccess	1	2015-02-07 05:39 UTC+1100	2015-02-07 05:39 UTC+...
<input type="checkbox"/>		AmazonAPIGatewayAdmini...	0	2015-07-10 03:34 UTC+1000	2015-07-10 03:34 UTC+...
<input type="checkbox"/>		AmazonWorkSpacesApplica...	0	2015-04-10 00:03 UTC+1000	2015-04-10 00:03 UTC+...
<input type="checkbox"/>		ServiceCatalogAdmin	0	2015-07-10 03:19 UTC+1000	2015-07-10 03:19 UTC+...
<input type="checkbox"/>		ServiceCatalogAdminReadO...	0	2015-07-10 03:21 UTC+1000	2015-07-10 03:21 UTC+...

Now click “Create Role”

You have now created a role that can be assigned to an EC2 instance to access AWS resources.

Creating a Security Group

In this section we will create a security group that can be assigned to our EC2 NodeJS server to restrict access from the internet.

Go to the EC2 console.

Click on “Security Groups” and select “Create Security Group”.

Call your security group WebServerSG.

Select the default VPC.

Select the inbound tab and add the following rules:

Inbound			
Source	Protocol	Port Range	Comments
0.0.0.0/0	TCP	80	Allow inbound HTTP access to the web servers from anywhere
0.0.0.0/0	TCP	443	Allow inbound HTTPS access to the web servers from anywhere
0.0.0.0/0	TCP	8080	Allow inbound HTTP access to the web servers from anywhere
My IP (your home network's public IP address range)	TCP	22	Allow inbound SSH access to Linux instances from your home network (over the Internet gateway)

Outbound

Destination	Protocol	Port Range	Comments
All traffic	TCP	All	Allow outbound traffic from the EC2 instance

Type ⓘ	Protocol ⓘ	Port Range ⓘ	Source ⓘ	
SSH ▾	TCP	22	My IP ▾ 0.0.0.0/0	✕
HTTP ▾	TCP	80	Anywhere ▾ 0.0.0.0/0	✕
HTTPS ▾	TCP	443	Anywhere ▾ 0.0.0.0/0	✕
Custom TCP Rule ▾	TCP	8080	Anywhere ▾ 0.0.0.0/0	✕

Add Rule

Cancel Save

Click “Create” to create the security group.

▶ Creating an EC2 instance

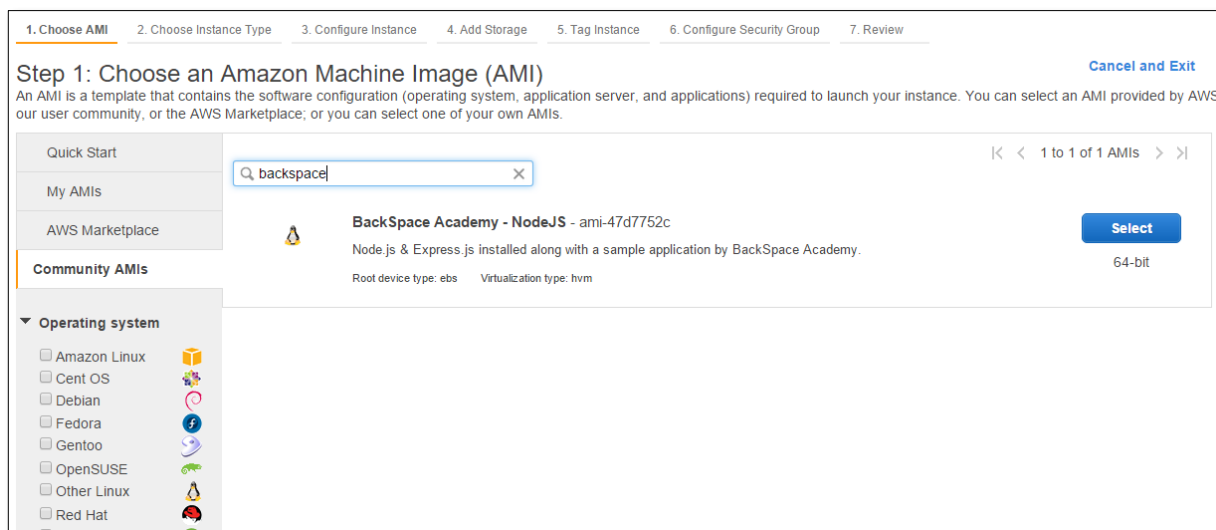
In this section we will create an EC2 instance from an AMI containing NodeJS. We will also bootstrap our instance to run a Linux bash script to set up firewall settings and update the operating system. We will also assign the IAM role and security group we created earlier.

Go to “Instances”

Click Launch Instance.

Select the Community AMIs tab.

Search for the BackSpace NodeJS AMI.



Click Select.

Select a t2 micro instance.

Click “Next: Configure Instance Details”

Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](#) about instance types and how they can meet your computing needs.

Filter by: **All instance types** **Current generation** [Show/Hide Columns](#)

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
<input checked="" type="checkbox"/>	General purpose	t2.micro <small>Free tier eligible</small>	1	1	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.small	1	2	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.large	2	8	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	m4.large	2	8	EBS only	Yes	Moderate

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Configure Instance Details](#)

Select the default VPC

Enable “Auto assign public IP”

Select IAM role “ec2-admin”

Check “Protect against accidental termination”

Expand the “Advanced Details” section.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 6. Configure Security Group 7. Review and Launch

Step 3: Configure Instance Details

Auto-assign Public IP (i) **Enable**

IAM role (i) **ec2-admin** [Create new IAM role](#)

Shutdown behavior (i) **Stop**

Enable termination protection (i) ☒ Protect against accidental termination

Monitoring (i) ☐ Enable CloudWatch detailed monitoring
Additional charges apply.

Tenancy (i) **Shared tenancy (multi-tenant hardware)**
Additional charges will apply for dedicated tenancy.

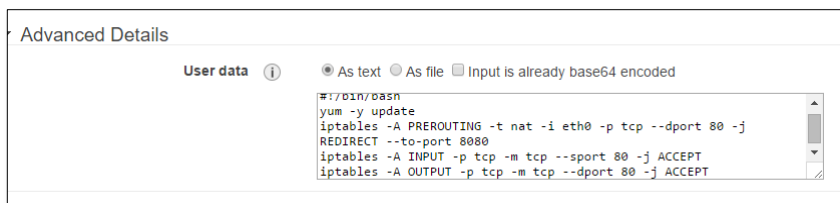
▼ Advanced Details

User data (i) ☒ As text ☐ As file ☐ Input is already base64 encoded

(Optional)

In “User Data” we now have to add our bash script to set up the firewall settings that is run when the instance is launched:

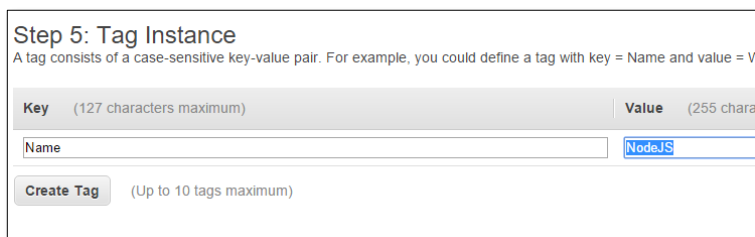
```
#!/bin/bash
yum -y update
iptables -A PREROUTING -t nat -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 8080
iptables -A INPUT -p tcp -m tcp --sport 80 -j ACCEPT
iptables -A OUTPUT -p tcp -m tcp --dport 80 -j ACCEPT
```



Click “Next add storage”

Click “Next tag instance”

Give it the name NodeJS



Click “Next configure security group”

Select your existing WebServerSG you created earlier.

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: ☐ Create a **new** security group
☒ Select an **existing** security group

Security Group ID	Name	Description	Actions
<input type="checkbox"/> sg-65f9c601	default	default VPC security group	Copy to new
<input checked="" type="checkbox"/> sg-69ad2c0e	WebServerSG	Web Server security group	Copy to new

Inbound rules for sg-69ad2c0e (Selected security groups: sg-69ad2c0e)

Type ⓘ	Protocol ⓘ	Port Range ⓘ	Source ⓘ
SSH	TCP	22	203.206.165.58/32
HTTP	TCP	80	0.0.0.0/0
HTTPS	TCP	443	0.0.0.0/0

[Cancel](#) [Previous](#) [Review and Launch](#)

Click “Review and Launch”

Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your instance and complete the launch process.

AMI Details [Edit AMI](#)



BackSpace Academy - NodeJS - ami-47d7752c

Node.js & Express.js installed along with a sample application by BackSpace Academy.

Root Device Type: ebs Virtualization type: hvm

Instance Type [Edit instance type](#)

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
t2.micro	Variable	1	1	EBS only	-	Low to Moderate

Security Groups [Edit security groups](#)

Security Group ID	Name	Description
sg-69ad2c0e	WebServerSG	Web Server security group

All selected security groups inbound rules

[Cancel](#) [Previous](#) [Launch](#)

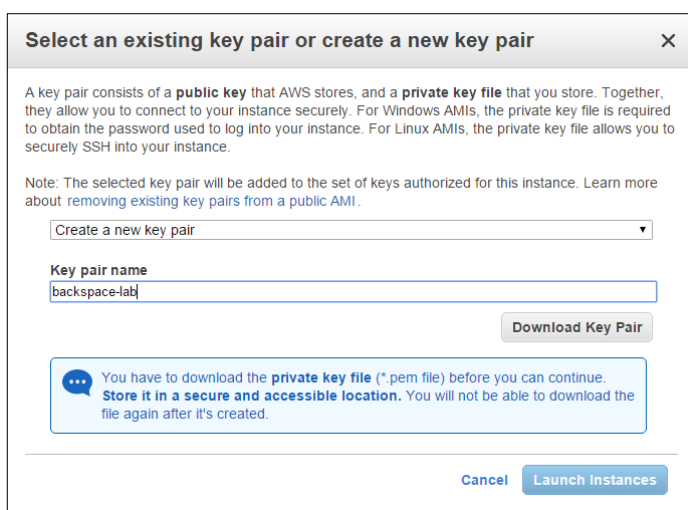
Click “Launch”

Select “Create a new key pair”

Call the key pair backspace-lab.

Create a directory on your windows system at C:\KeyPairs

Download the key backspace-lab.pem file to C:\KeyPairs

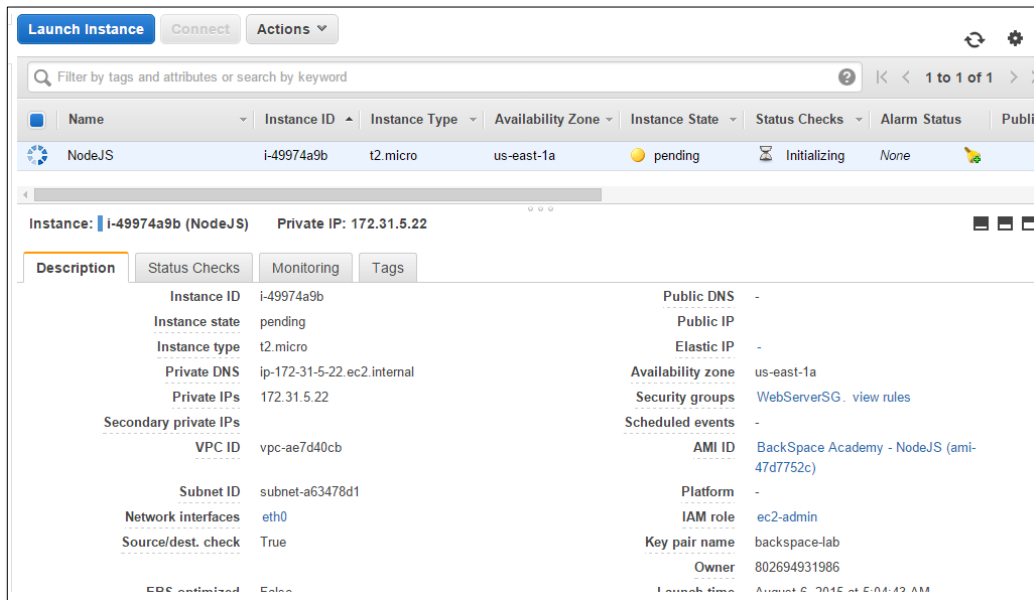


The screenshot shows a dialog box titled "Select an existing key pair or create a new key pair" with a close button (X) in the top right corner. The dialog contains the following elements:

- Text:** "A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance."
- Note:** "Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about [removing existing key pairs from a public AMI](#)."
- Dropdown menu:** A dropdown menu with the text "Create a new key pair" and a downward arrow.
- Text input:** A text input field labeled "Key pair name" containing the text "backspace-lab".
- Button:** A button labeled "Download Key Pair" located to the right of the text input field.
- Message box:** A blue-bordered message box with a speech bubble icon containing the text: "You have to download the **private key file** (*.pem file) before you can continue. **Store it in a secure and accessible location.** You will not be able to download the file again after it's created."
- Buttons:** At the bottom, there are two buttons: "Cancel" and "Launch Instances".

Click “Launch Instance”

Click “View Instance”



You have now created an EC2 server ready to go with NodeJS, Express and the AWS SDK.

▶ Connecting to your EC2 instance using SSH

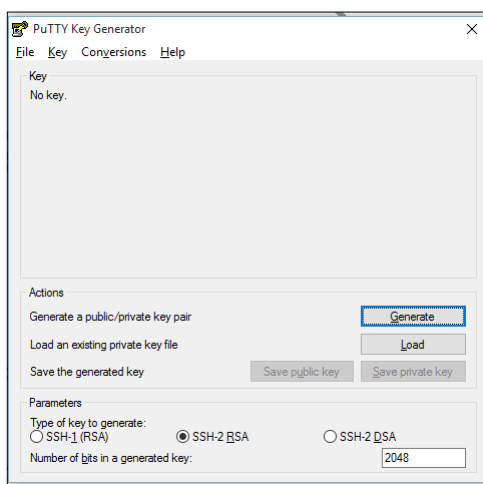
In this section we will connect from our Windows desktop to our EC2 instance using SSH and Putty. Mac OSX and Linux have SSH support without installing an additional client software.

Go to <http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html> and download the following executable files:

Putty (putty.exe)

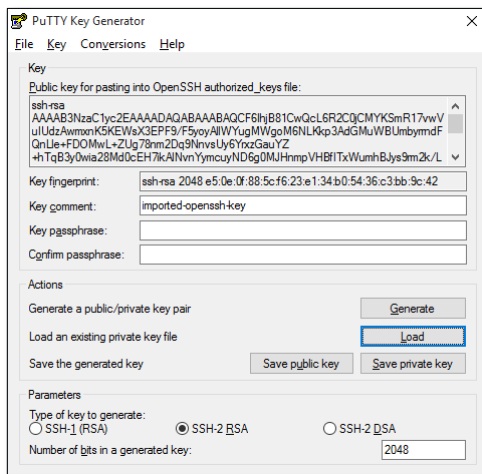
Putty Key Generator (puttygen.exe)

When they have downloaded run puttygen.exe



We need to convert our backspace-lab.pem to a ppk file suitable for Putty.

Click load and select "All files" and select the backspace-lab.pem from C:\KeyPairs



Click "Save Private Key"

Click "yes" to save without passphrase

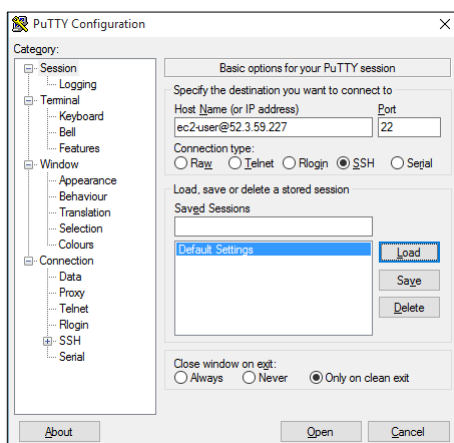
Save as backspace-lab to C:\KeyPairs

Close Puttygen

Go back to the EC2 console and copy your instances Public IP

Now run Putty.exe

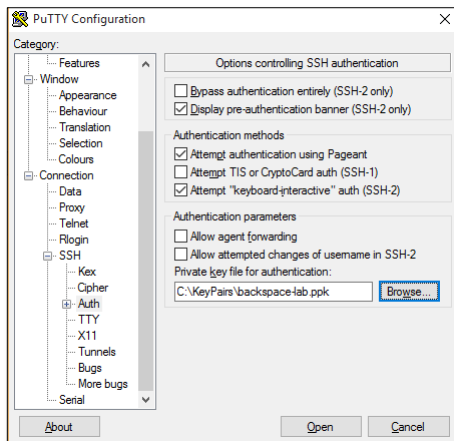
Input the hostname as ec2-user@(your Public IP) and port as 22



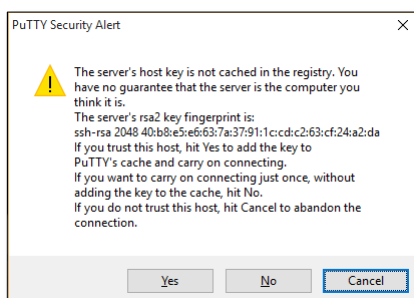
Click on SSH in the directory tree to expand.

Click on Auth in the directory tree.

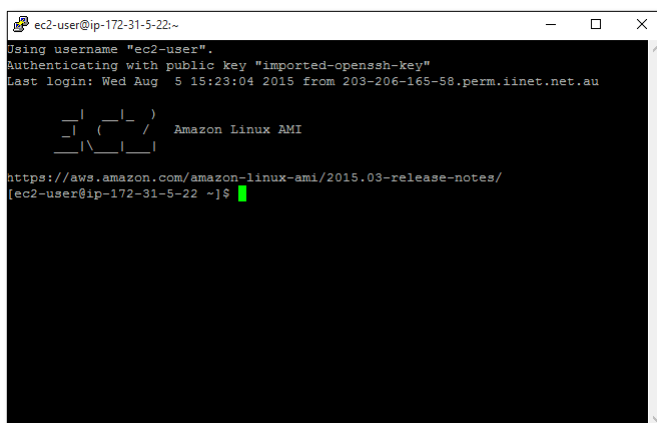
Click on "browse" and select the backspace-lab.ppk file



Click "Open"



Click "Yes"

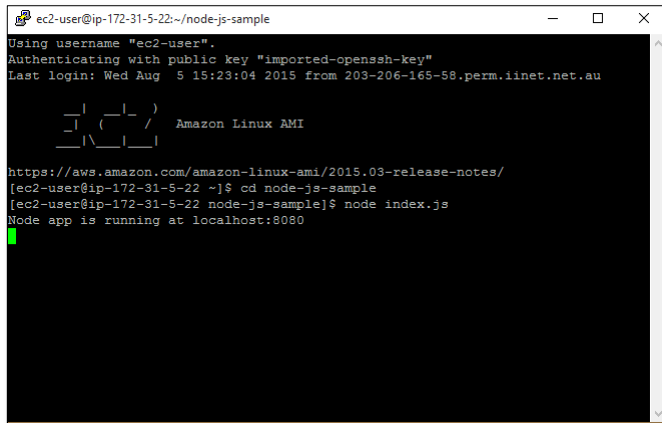


You are now connected to your EC2 instance.

Now run the sample NodeJS app with the following commands:

```
cd node-js-sample
```

```
node index.js
```



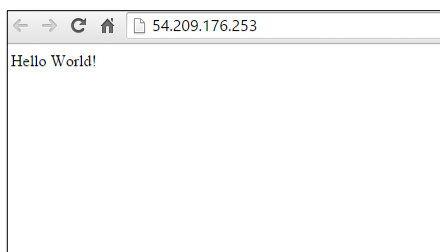
```
ec2-user@ip-172-31-5-22:~/node-js-sample
Using username "ec2-user".
Authenticating with public key "imported-openssh-key"
Last login: Wed Aug  5 15:23:04 2015 from 203-206-165-58.perm.iinet.net.au

 _ _ | ( _ | /   Amazon Linux AMI
 _ _ | \ _ | _ |

https://aws.amazon.com/amazon-linux-ami/2015.03-release-notes/
[ec2-user@ip-172-31-5-22 ~]$ cd node-js-sample
[ec2-user@ip-172-31-5-22 node-js-sample]$ node index.js
Node app is running at localhost:8080
```

Your NodeJS app is now running.

Point your browser to your instance Public IP address and you will see the standard “Hello World!”

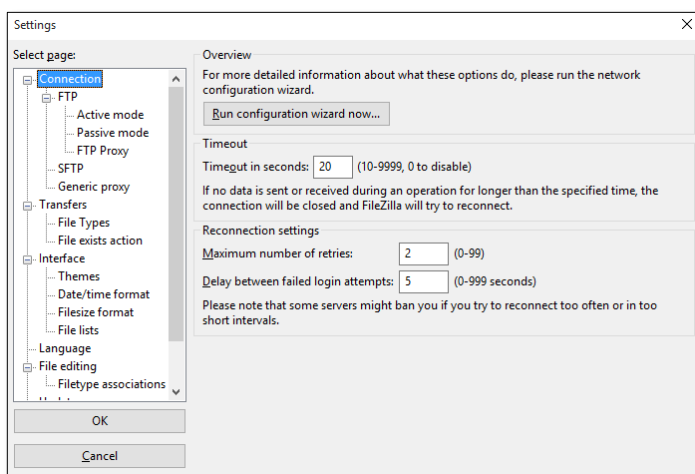


▶ Transferring files to an EC2 instance using SFTP

In this section we will set up FileZilla to allow us to transfer files to our EC2 instance using SFTP. The instructions are for Windows although FileZilla is available for Mac OSX and Linux also.

Open FileZilla

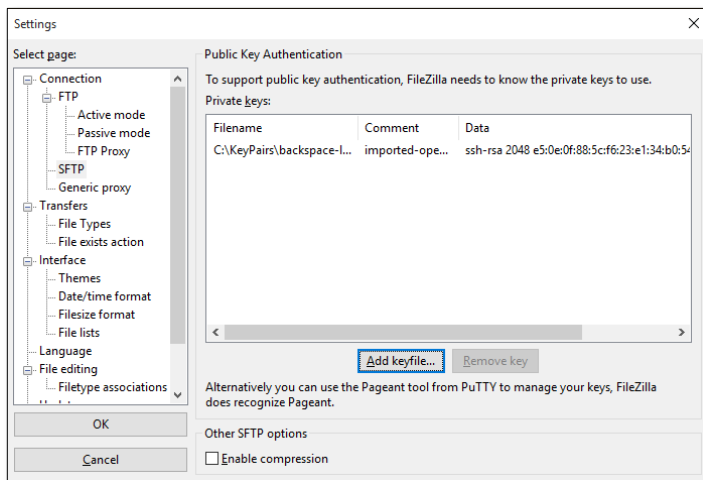
Go to "Edit" -> "Settings"



Click on "SFTP"

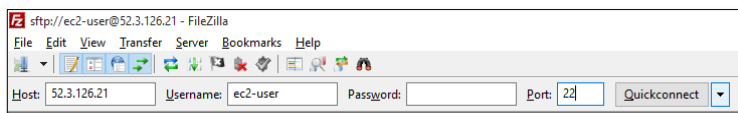
Click "Add Keyfile"

Select the backspace-lab.ppk (not pem) file.



Click OK

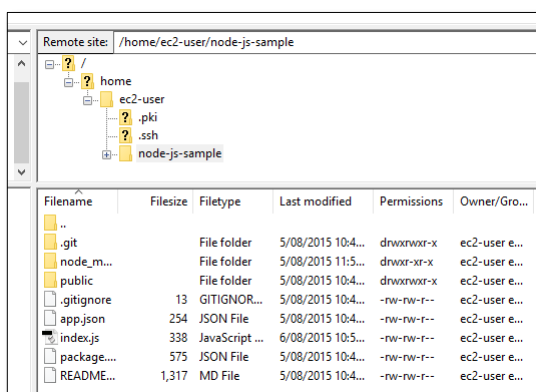
Enter your EC2 instance public IP, username ec2-user and port 22.



Click "Quick Connect"

You will then be connected to the EC2 instance.

Navigate to the node-js-sample folder to see the sample app.



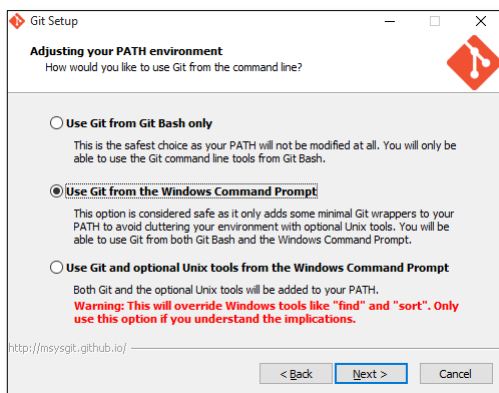
▶ Setting Up the GIT Version Control System

In this section we will set up the GIT Version Control System and integrate it into the Atom.io Integrated Development Environment (IDE) to allow us to keep track of changes to our code.

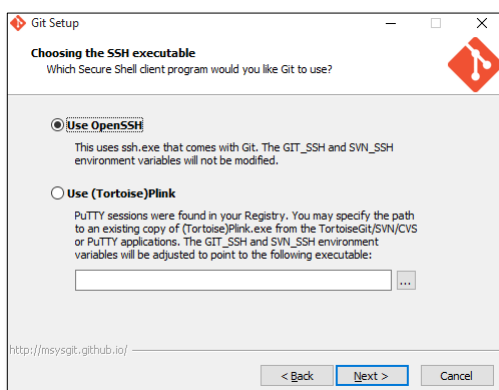
Download GIT for Windows

<https://git-scm.com/download/win>

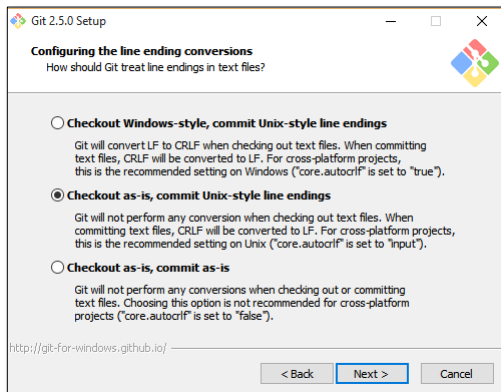
Select "Use GIT from the Windows Command Prompt", click Next



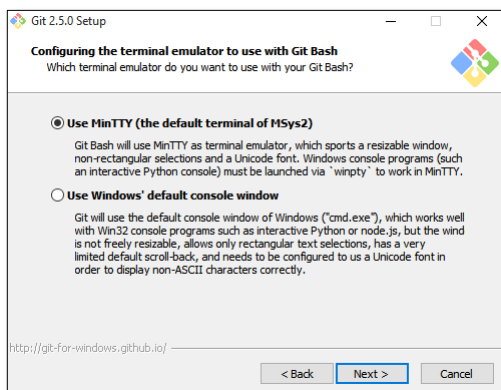
Select OpenSSH, click Next



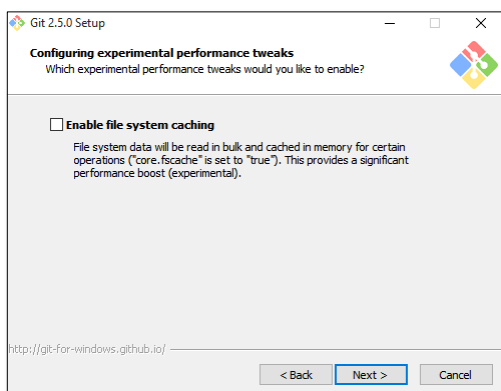
Select Checkout as-is, commit Unix-style line changes style, click Next



Select Use MinTTY, click Next



click Next



Finish installation!

▶ Integrating SFTP EC2 Connection and GIT with an IDE

In this section we will set up the Atom.io Integrated Development Environment (IDE) to allow syncing of files to our EC2 instance using SFTP.

Download and install Atom.io

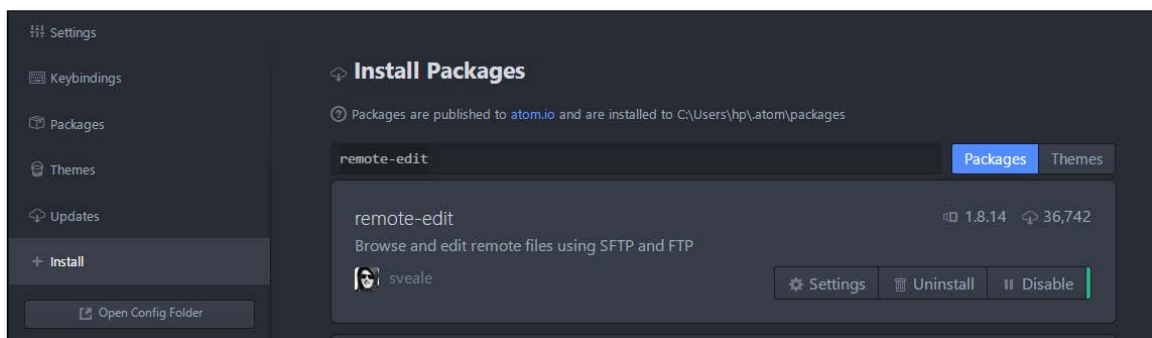
Open the Atom IDE.

Go to “File” -> “Settings”

Select “Install”

Search for the “remote-edit” package.

Install the package.



Go to “Packages” from the top menu then select “Remote Edit” then “Add new host SFTP”

Click on “Private Key”

Enter the path to your backspace-lab.ppk (not pem) file.

Enter your EC2 public IP for hostname (IP address only without http://)

Enter node-js-sample/ for default directory

Enter username ec2-user

Enter port 22

Connection settings

Hostname:
52.5.22.137

Default directory:
node-js-sample/

Username:
ec2-user

Port:
22

Authentication settings

User agent Private key Password

Private key path:
C:\KeyPairs\backspace-lab.ppk

Private key passphrase:

Passphrases are stored in cleartext! Consider using an ssh-agent instead. Leave passphrase field blank if key does not require passphrase.

Additional settings

Alias:

Save Cancel

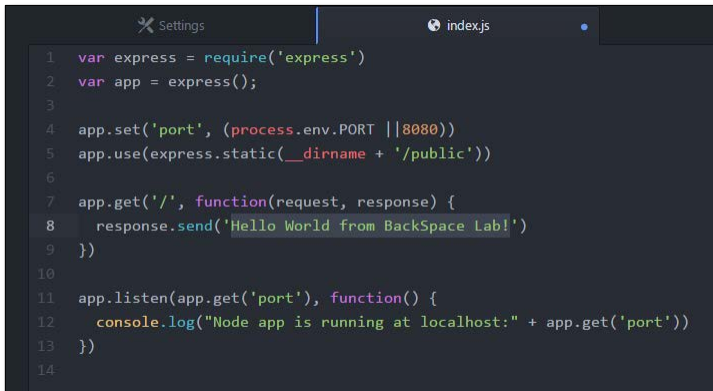
Click Save

Go to “Packages” from the top menu then select “Remote Edit” then “Browse Hosts”

Select the SFTP host.

Select the “index.js” file.

Change the response to “Hello World from BackSpace Lab!”



```
1 var express = require('express')
2 var app = express();
3
4 app.set('port', (process.env.PORT || 8080))
5 app.use(express.static(__dirname + '/public'))
6
7 app.get('/', function(request, response) {
8   response.send('Hello World from BackSpace Lab!')
9 })
10
11 app.listen(app.get('port'), function() {
12   console.log("Node app is running at localhost:" + app.get('port'))
13 })
14
```

Press  +  to save back to the EC2 instance.

NOTE: If you have made a mistake and need to edit your settings, “browse hosts” again and press:

 +  to edit the host.

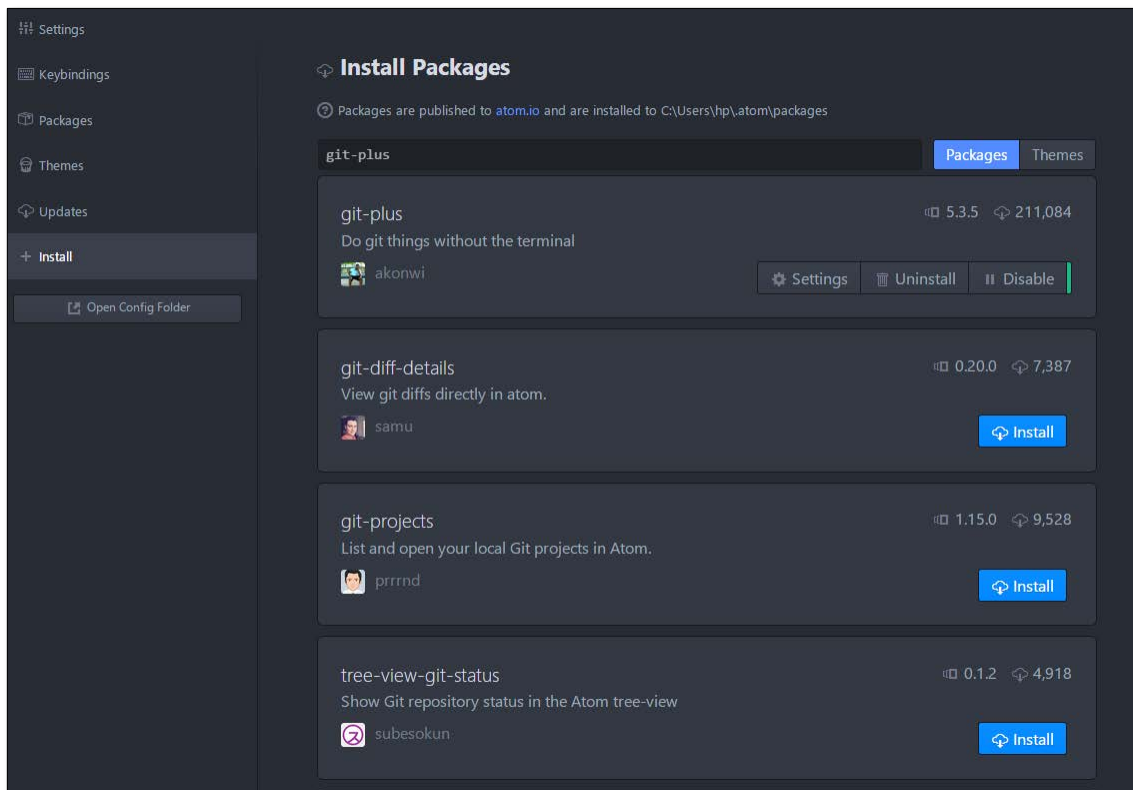
Now run your node app and check your browser to see the changes.

Now we will integrate GIT into our IDE

Go to File – Settings

Click Install.

Search for git-plus and install.



Other useful packages:

- linter
- linter-jshint
- linter-htmlhint
- linter-csslint
- project-manager