

# Cloud Computing Exercise – 1

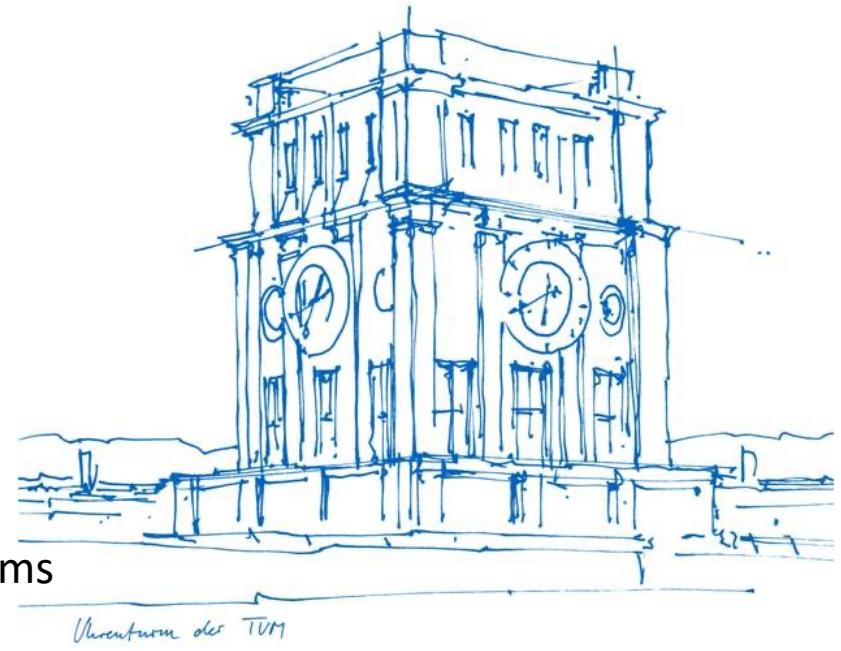
## Application Development using Node.js

Anshul Jindal (M.Sc. Informatics)

[anshul.jindal@tum.de](mailto:anshul.jindal@tum.de)

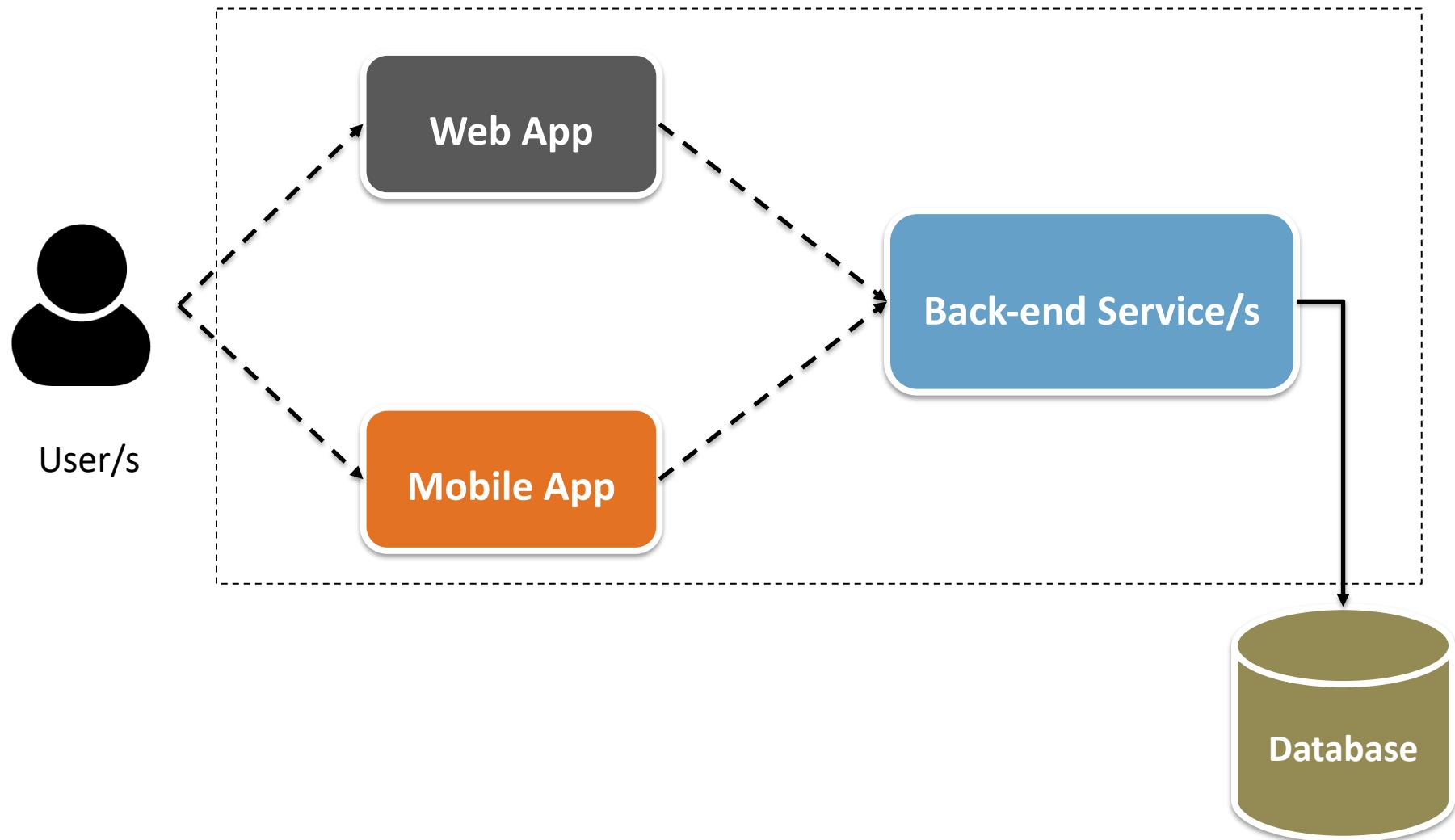
Chair of Computer Architecture and Parallel Systems

*Technical University of Munich (TUM), Germany*

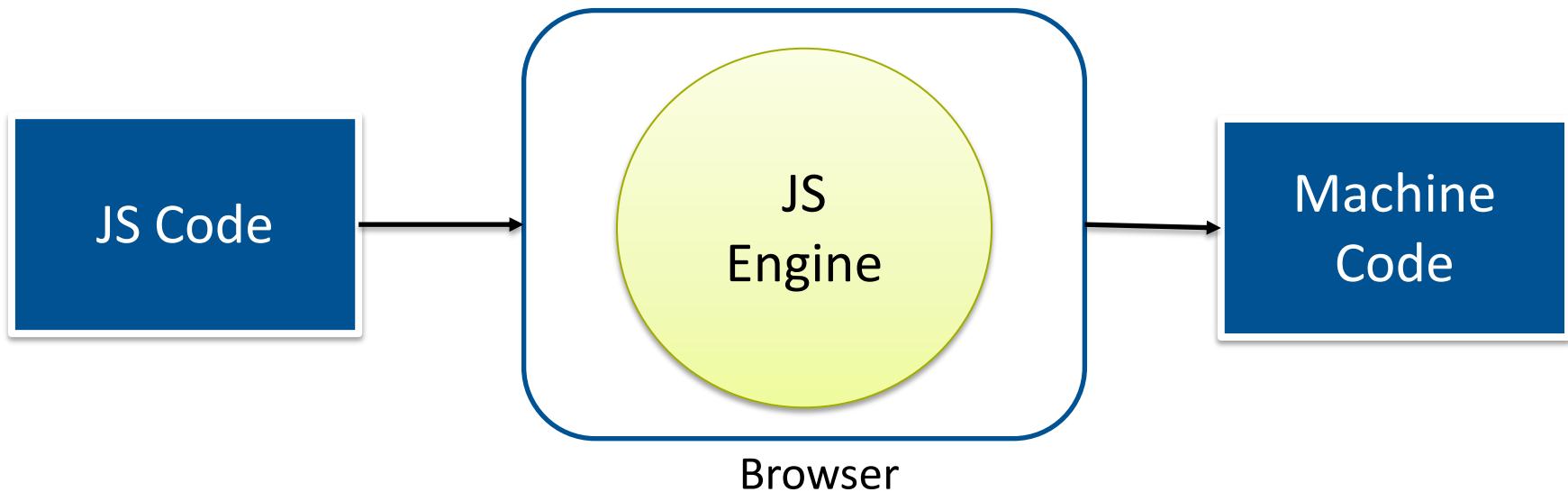


# Introduction to Node.js

# Example of a Request to an Application Flow



# How JavaScript (JS) is running on a browser ?



## JS Engine in Browsers



Microsoft edge  
has Chakra



Firefox  
has Spider Monkey



Chrome  
has V8

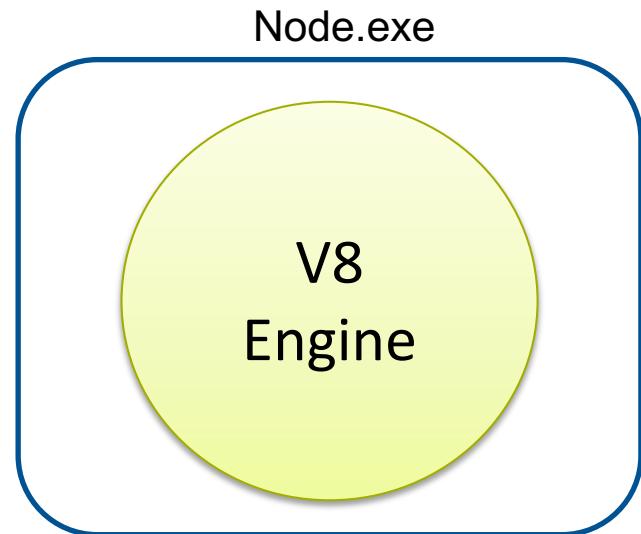
# Node

- Browser provides Runtime Environment for JS code.
- Till 2009 only way to run JS code was inside a browser.
- Ryan Dahl inventor of Node.js in 2009 [1]



Chrome  
has V8

Took V8 Engine

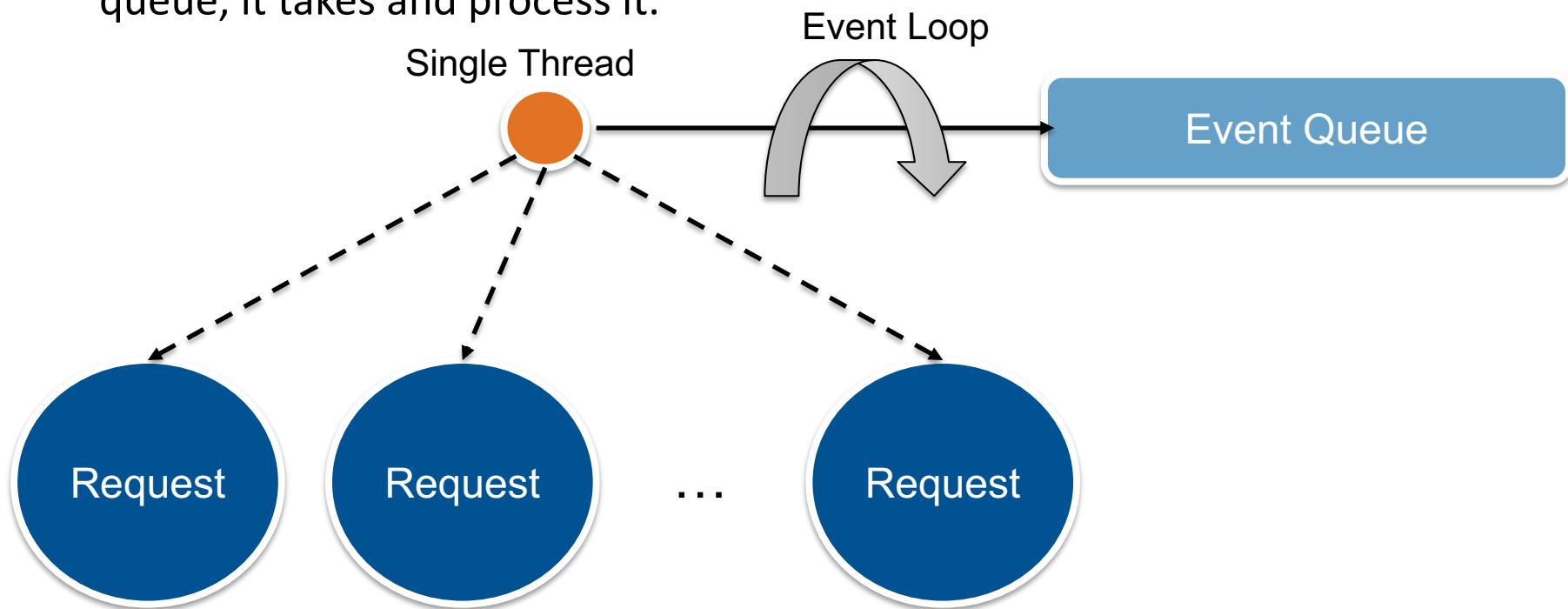


Embedded V8 inside a  
C++ program called as  
Node.exe

- Similar to browser Node is a Runtime Environment for JS code.
- “.js” was added to Node to just name the product.

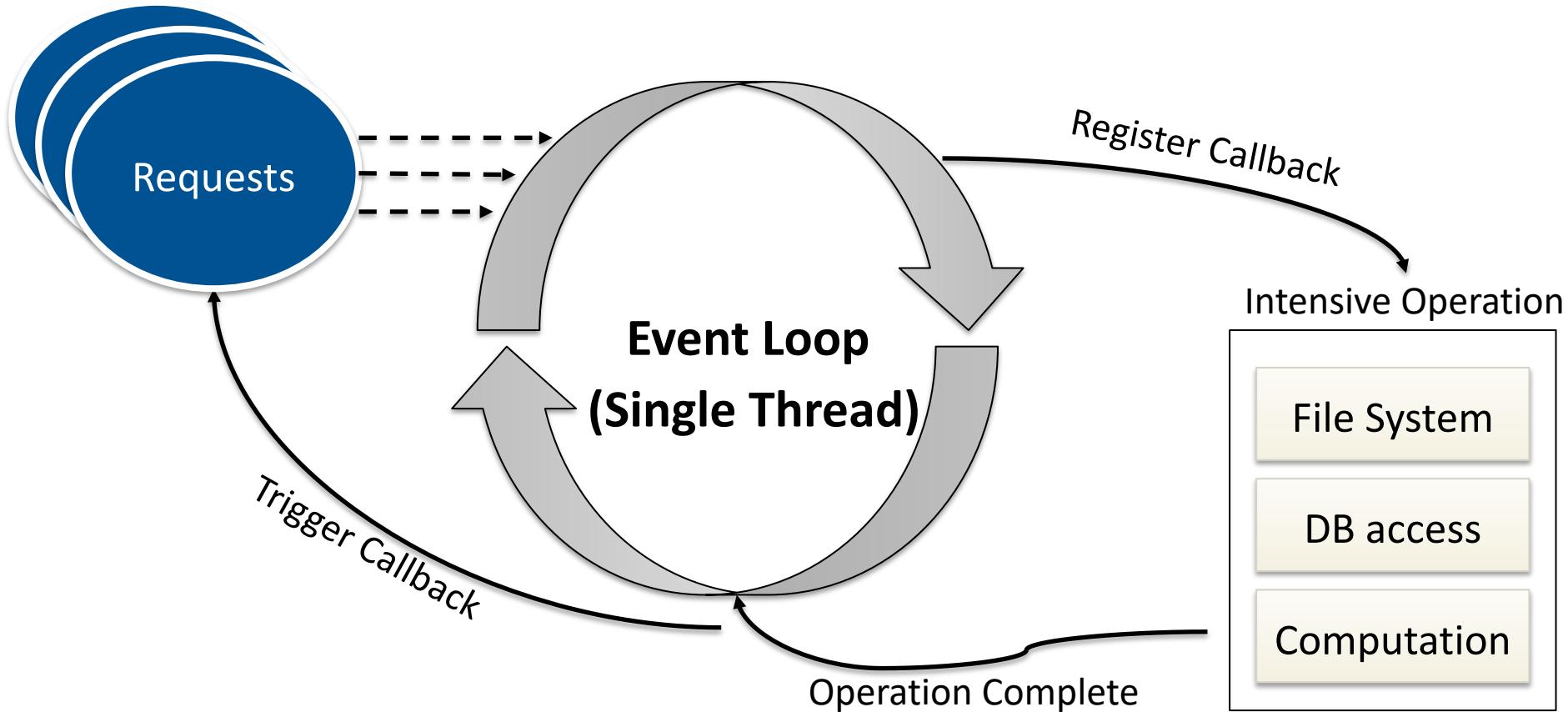
# Node.js Non-Blocking/Asynchronous Nature

- A single thread is used to handle multiple requests.
- If a request needs to query a Database, thread does not wait for the database to return data.
- It moves to the next request for handling.
- When the database result is ready, its added to event queue.
- Node monitors this queue continuously and whenever it finds an event in this queue, it takes and process it.



# Node.js Event Loop

The event loop simply iterate over the event queue which is basically a list of events and callbacks of completed operations.



*“A platform built on Chrome’s JavaScript runtime for easily building fast, scalable network applications. [2]”*

- Core in c++ on top of Chrome v8 engine.
- Can handle thousands of Concurrent connections with Minimal overhead (cpu/memory) on a single process.
- It’s NOT a web framework, and it’s also NOT a language.
- It is a Runtime Environment for JS code.
- Works on single thread using non-blocking I/O calls

# When and when not to use Node.js



## When to use ?

- **Real time applications** : applications that have to process a high volume of short messages requiring low latency.
- **Fast and scalable environment:**
  - ability to process many requests with low response times.
  - makes it a great fit for modern web applications that carry out lots of processing on the client's side. Ex. single-page applications.

## When not to use ?

- **CPU-heavy jobs:**
  - Node.js is based on an event-driven, non-blocking I/O model, and uses only a single CPU core.
  - CPU-heavy operations will just block incoming requests, rendering the biggest advantage of Node.js useless.
- **Simple Web Applications:** Using Node.js would be superfluous for simple HTML applications in which you don't need a separate API, and all data comes directly from the server.

# PayPal Case Study [3]

- PayPal, a worldwide online payments system, were using the Java on server side and JavaScript on the browser side.
- This gave them lots of problems and eventually they moved their backend development from Java to JavaScript and Node.js.
- They found that, the Node.js app was:

Built almost **twice as fast with fewer people**

Written in 33% fewer lines of code

Constructed with 40% fewer files

Doubled the number of requests per second served

35% decrease in the average response time for the same page.

# Node Package Manager (NPM) [4]

- Used to install node programs/modules
- Easy to specify and link dependencies
- Modules get installed inside “**node\_modules**” folder

```
npm install express
```

// install express module

```
npm install express --save
```

// install express module and add in **package.json**

```
npm install -g express
```

// install express globally

- Popular Modules
  - **Express** – Web development framework
  - **Connect** – Extensible HTTP server framework
  - **Socket.io** – Server side component for WebSocket
  - **Mongo/Mongoose** – Wrappers to interact with mongo database

# package.json

- Present inside the root folder of your package/application
- Tells npm how the package is structured and what all dependencies need to be installed.

```
{  
  "name": "clientapp",  
  "main": "clientServer.js",  
  "repository": {  
    "type": "git",  
    "url": ""  
  },  
  "scripts": {  
    "start": "node clientServer.js"  
  },  
  "author": "Anshul Jindal",  
  "license": "",  
  "dependencies": {  
    "child_process": "^1.0.2",  
    "express": "~4.0.0"  
  },  
  "homepage": "",  
  "version": "1.0.0",  
  "description": "client application for CCS"  
}
```

Main file of the app

Called when **npm start** command is run

List of all dependencies

- REST – REpresentational State Transfer
- Resources Based – Deals with things instead of actions (like a person, address)
- Representations –

Means how resources get manipulated.

Transferred between client and server (Typically JSON or XML)

- 6 constraints, which makes a web service – a true RESTful API [5]

## 1. Uniform Interface –

- HTTP actions (GET, POST, PUT, DELETE)
- Uniform Resource Identifiers (URIs) (resource names)
- HTTP response (status, body)

## 2. Stateless –

- Server contains no client state (No session, no history)
- Each request contains enough context to process message.

## 3. Client-server –

- Client application and server application MUST be able to evolve separately without any dependency on each other

## 4. Cacheable –

- caching can be applied to resources when applicable and these resources MUST be declared cacheable.
- Caching can be implemented on the server or client side.

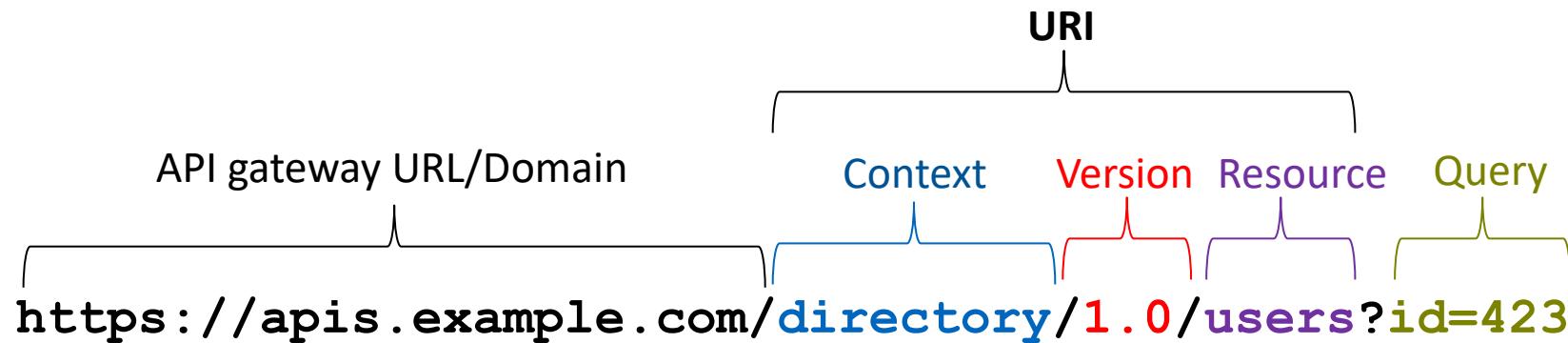
## 5. Layered system –

- Uses layered system architecture where for example we deploy the APIs on server A, and store data on server B and authenticate requests in Server C.

## 6. Code on demand (optional) –

- Most of the time we will be sending the static representations of resources in form of XML or JSON.
- Can return executable code to support a part of your application e.g. clients may call API to get a UI widget rendering code.

# REST API example



```
{  
    "id": "423",  
    "name": "xyz",  
    "status": "student"  
}
```

**Example JSON Response**

# MongoDB

- MongoDB is an open-source document database that provides high performance, high availability, and automatic scaling [6].
- NoSQL Database, uses JSON like documents to store information.
- A record in MongoDB is a **document**, which is a data structure composed of field and value pairs.

```
{  
    "id": "423",  
    "name": "xyz",  
    "status": "student"  
}
```

field:value  
field:value  
field:value

- **Collection:** A grouping of MongoDB documents.
- **Databases** hold collections of documents.
- More Information [here](#)

# Installation

# Installation of Node and NPM

- Get a recent version of Node.js by adding a PPA (personal package archive) maintained by NodeSource for Ubuntu\*

```
wget -qO- https://deb.nodesource.com/setup_10.x | sudo -E bash -
```

- You can now install the Node.js package.

```
sudo apt-get install -y nodejs
```

- For some npm packages to work (such as those that require building from source), you will need to install the build-essentials package

```
sudo apt-get install -y build-essential
```

- Test Node: `node -v` (This should print a version number)
- Test NPM: `npm -v` (This should print NPM's version number)

\*For other OS check: <https://nodejs.org/en/download/>

# Install Mongo Database



Check here for installation steps for MongoDB

<https://docs.mongodb.com/manual/administration/install-community/>

Lets see some code...

# Directory Structure of the Provided Application\*



.editorconfig	Used for editor configuration, like Indentation, spaces etc.
eslintrc.json	
package.json	For handling dependencies
server.js	Main file for running back-end web server
models books.js index.js	Contains schema of the object to be stored in MongoDB
public scripts app.js	For HTML file styling and front-end JS
styles styles.css	
views index.html	Contains HTML file

\*Code is based upon ECMAScript 6 specification.

# Models – Books.js

- We are using [Mongoose](#) to connect to MongoDB. See queries [here](#).
- Schema Used in this application:

```
const mongoose = require('mongoose'),  
Schema = mongoose.Schema;
```

Importing Mongoose

```
const BooksSchema = new Schema({  
    title: String,  
    author: String,  
    releaseDate: String,  
    genre: String,  
    rating: String,  
    language: String  
});
```

Creating a schema for the object that will be stored inside the mongo database

```
const BooksModel =  
mongoose.model('books', BooksSchema);
```

Creating the model **books** and exporting it. The name **books** will be used by mongo DB for creating collection

```
module.exports = BooksModel;
```

# Models – Index.js

- Similarly we can have more schemas as shown in previous slide in different files.
- We can now consolidate all schemas in one file and export them for further use.

```
const mongoose = require("mongoose");
mongoose.connect( process.env.MONGODB_URI ||  
  "mongodb://localhost:27017/booksData", {  
useNewUrlParser: true } );
```

Connection to Mongo DB  
which is running locally at  
**localhost:27017** and  
database name as  
**booksData**

```
module.exports.books =  
require("./books.js");
```

We export our **books**  
schema as **books** module  
from here.

# server.js

- This is the main back-end service file.
- It is responsible for starting web server and contains all the API endpoints.

```
const express = require('express');  
const app = express();
```

Express is used to create web server.  
More Info [here](#)

extract the entire body portion of an incoming request stream and exposes it on req.body.

```
app.use(express.urlencoded({extended: true}));
```

Accepts only UTF-8 encoding of the body

```
app.use(express.json());
```

Accepts any type of encoding of the body

```
// Set Static File Directory
```

```
app.use(express.static(__dirname + '/public'));
```

Directory for static files like CSS files or images

# server.js continue..

## Routes/Endpoints

- **Root Endpoint “/” [GET]**

```
app.get('/', function homepage(req, res){  
    res.sendFile(__dirname +  
        '/views/index.html');  
});
```

- **API information Endpoint “/api” [GET]**

GET request

```
app.get('/api', (req, res) => {  
    res.json([...])  
});
```

Notice these two calls.. These both are same

request

response

res)

On accessing Root endpoint like on <http://example.com/> calls the function **homepage** which is sending the **index.html** file

Called on endpoint **/api** like on <http://example.com/api>, which is sending some **JSON** information

# server.js continue..

- All stored books information Endpoint “/api/books” [GET]

```
app.get('/api/books/' , (req, res) => {
```

Empty means to get all

```
    db.books.find({} , function (err, books) {  
        if (err) throw err;
```

uses the books model and query to mongo database to get all objects stored.

```
        res.json(books);
```

**books** is an array of found JSON objects and it is returned back to the user.

```
    } );  
});
```

- Similarly there are other API endpoints for POST, PUT, DELETE

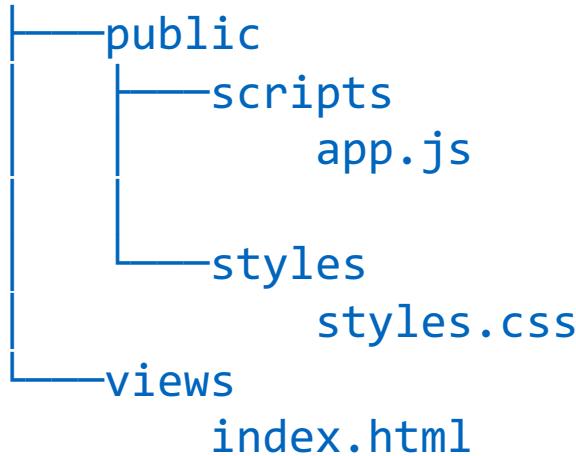
# server.js continue..

- To Listen on a particular port for the webserver.
- Here it runs on port 80.
- Webserver can be accessed at <http://localhost:80> from browser

If environment variable is present otherwise use 80

```
app.listen(process.env.PORT || 80, () => {  
  console.log('Express server is up and  
running on http://localhost:80/');  
}) ;
```

# Public and Views Directory



- Here the code related to front-end is specified like how to style and display the webpage.
- You can access the html webpage at the address <http://localhost:80> , once the server is running.

# Running the application

# Download the Provided Application

1. Download the provided application source zip file from Moodle.
2. Unzip the file

```
sudo apt-get install unzip
```

```
unzip cloud-computing-exercise-app.zip -d cloud-computing-exercise-app
```

# Start MongoDB

1. Open a Terminal and make the current directory as “cloud-computing-exercise-app”

`cd cloud-computing-exercise-app`

2. Make a directory with the name “data”. This will store all mongo database information

`mkdir data`

3. Run the following command to start mongo database

`mongod --dbpath=./data`

```
[initandlisten] MongoDB starting : pid=18352 port=27017 dbpath=data 64-bit host=DESKTOP-670NKOH
[initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
[initandlisten] db version v3.4.7
[initandlisten] git version: cf38c1b8a0a8dca4a11737581beafef4fe120bcd
[initandlisten] OpenSSL version: OpenSSL 1.0.1u-fips  22 Sep 2016
[initandlisten] allocator: tcmalloc
[initandlisten] modules: none
[initandlisten] build environment:
[initandlisten]   distmod: 2008plus-ssl
[initandlisten]   distarch: x86_64
[initandlisten]   target_arch: x86_64
[initandlisten] options: { storage: { dbPath: "data" } }
```

# Start Node.js Application

1. Open another Terminal and make the current directory as “cloud-computing-exercise-app”

```
cd cloud-computing-exercise-app
```

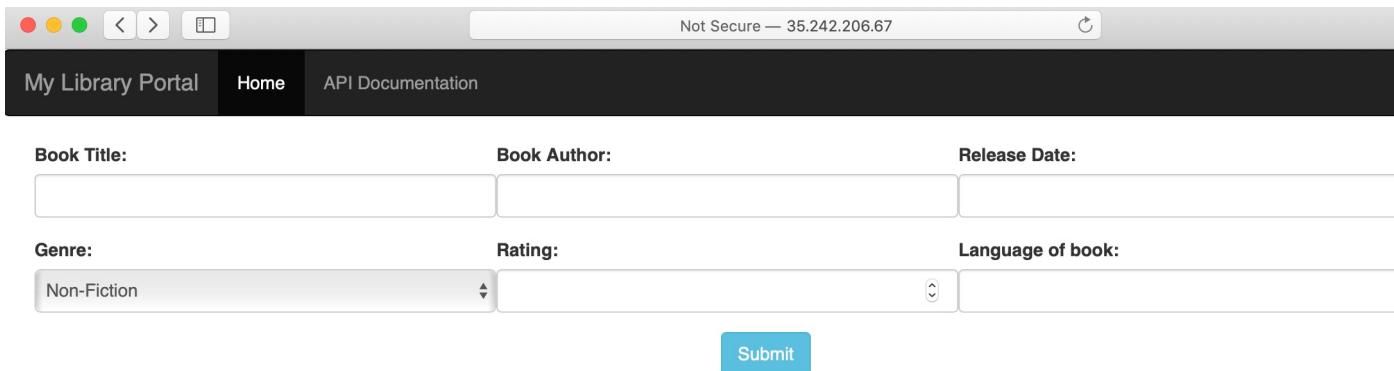
2. Install All node.js dependencies present inside **package.json** file

```
npm install
```

3. Start the application

```
sudo node server.js
```

4. You can now check the browser at address <http://localhost:80>



The screenshot shows a web browser window with the following details:

- Address Bar:** Not Secure — 35.242.206.67
- Header:** My Library Portal, Home, API Documentation
- Form Fields:**
  - Book Title: [Input Field]
  - Book Author: [Input Field]
  - Release Date: [Input Field]
  - Genre: Non-Fiction [Dropdown: Non-Fiction]
  - Rating: [Input Field]
  - Language of book: [Input Field]
- Buttons:** Submit

## All Stored Books Information

# Tasks to be completed

# Tasks to be completed

1. Document all your api endpoints in a simple hardcoded JSON object in the “/api” endpoint. Only some are provided as an example, write others

```
app.get('/api', (req, res) => {
    // TODO: Document all your api endpoints below as a
    res.json({
        message: 'Welcome to my app api!',
        documentationUrl: '', //leave this also blank for
        baseUrl: '', //leave this blank for the first exe:
        endpoints: [
            {method: 'GET', path: '/api', description: 'Des'},
            {method: 'GET', path: '/api/profile', descriptio},
            {method: 'GET', path: '/api/books/', descriptio}
            // TODO: Write other API end-points description
        ]
    })
});
```

# Tasks to be completed Continue..

2. Complete the `/api/profile` endpoint. You can add here fake information too, to make it more interesting like Name as Jon Snow, homeCountry as winterfell ☺

```
// TODO: Fill the values
app.get('/api/profile', (req, res) => {
  res.json({
    'name': '',
    'homeCountry': '',
    'degreeProgram': '', //informatics or
    'email': '',
    'deployedURLLink': '', //leave this bl
    'apiDocumentationURL': '', //leave th
    'currentCity': '',
    'hobbies': []
  })
}) ;
```

# Tasks to be completed Continue..

3. Complete other missing APIs. Study mongoose queries [here](#).

- **/api/books [POST]** : To store new book information and return the stored information as JSON.

```
app.post('/api/books/' , (req, res) => {  
  
    /*  
     * New Book information in req.body  
     */  
    console.log(req.body);  
    /*  
     * TODO: use the books model and create a new object  
     * with the information in req.body  
     */  
    /*  
     * return the new book information object as json  
     */  
    var newBook = {};  
    res.json(newBook);  
}) ;
```

# Tasks to be completed Continue..

3. Complete other missing APIs. Study mongoose queries [here](#).

- **/api/books/:id [PUT]** : To Update a book information based upon the provided id and new information. After updating return the updated JSON.

```
app.put('/api/books/:id', (req, res) => {
  /*
   * Get the book ID and new information of book from
   * the request parameters
   */
  const bookId = req.params.id;
  const bookNewData = req.body;
  /*
   * TODO: use the books model and find using the bookId and
   * update the book information
   */
  /*
   * Send the updated book information as a JSON object
   */
  var updatedBookInfo = {};
  res.json(updatedBookInfo);
});
```

# Tasks to be completed Continue..

3. Complete other missing APIs. Study mongoose queries [here](#).

- **/api/books/:id [DELETE]**: To delete a book information based upon the id.  
Return the delete book information as the JSON

```
app.delete('/api/books/:id', (req, res) => {
  /*
   * Get the book ID of book from the request
   * parameters
   */
  const bookId = req.params.id;
  /*
   * TODO: use the books model and find using
   * the bookId and delete the book
   */
  /*
   * Send the deleted book information as a JSON object
   */
  var deletedBook = {};
  res.json(deletedBook);
});
```

# Submission

# Submission Instructions

To submit your application results you need to follow this :

1. Open the Cloud Class server url : <https://cloudcom.caps.in.tum.de/>
2. Login with your provided username and password.
3. After logging in, you will find the button for **exercise1**
4. Click on it and a form will come up where you must provide
  - VM IP on which your application is running

## Example:

10.0.23.1

5. Then click submit.
6. You will get the correct submission from server if everything is done correctly.

**Deadline for submission is three weeks from now (exact date and time is mentioned on the server)**

# Cloud Registration Link

- Google Cloud Platform: Will be added on Piazza and Moodle
- Microsoft Azure: <https://azure.microsoft.com/de-de/free/students/>
- AWS: [https://www.awseducate.com/registration#APP\\_TYPE](https://www.awseducate.com/registration#APP_TYPE)

# GCP VM Creation

# GCP Registration Link



- Click on the given GCP registration link: Will be added on Piazza and Moodle
- Instructions:

Will be added on Piazza and Moodle

# GCP Registration

- After clicking the link, fill the below shown information.
- Use official TUM email id only.

The screenshot shows a registration form for Google Cloud Platform Education Grants. The header reads "Cloud Platform Education Grants" and describes the program as providing credits via the Google Cloud Platform Education Grants program. The main message is to thank the user for their interest and to fill out the form to receive a coupon code. The form fields include "First Name" and "Last Name" (both empty), "School Email" (with a dropdown menu showing "@tum.de" and a link to contact the course instructor at [anshul.jindal@tum.de](mailto:anshul.jindal@tum.de) if the domain is not listed), and a statement about agreeing to share information with the educational institution and course instructor. A "Submit" button is at the bottom, and a "Privacy Policy" link is at the bottom right.

Cloud Platform Education Grants

Use credits provided to you via the Google Cloud Platform Education Grants program to access Google Cloud Platform. Get what you need to build and run your apps, websites and services.

Thank you for your interest in Google Cloud Platform Education Grants. Please fill out the form below to receive a coupon code for credit to use on Google Cloud Platform.

First Name

Last Name

School Email

@tum.de ▾

If you do not see your domain listed, please contact your course instructor: [anshul.jindal@tum.de](mailto:anshul.jindal@tum.de)

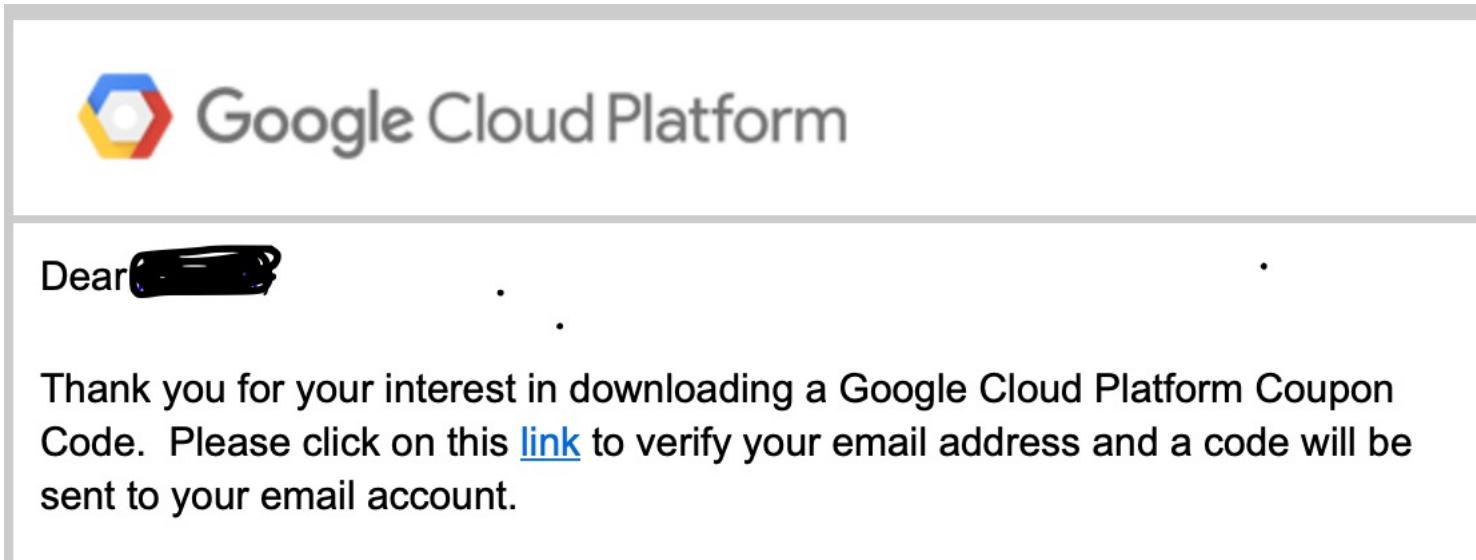
By clicking "Submit" below, you agree that we may share the following information with your educational institution and course instructor ([anshul.jindal@tum.de](mailto:anshul.jindal@tum.de)): (1) personal information that you provide to us on this form and (2) information regarding your use of the coupon and Google Cloud Platform products.

Submit

Privacy Policy

# GCP Registration Cont..

- After submitting the information you will receive an email like this:

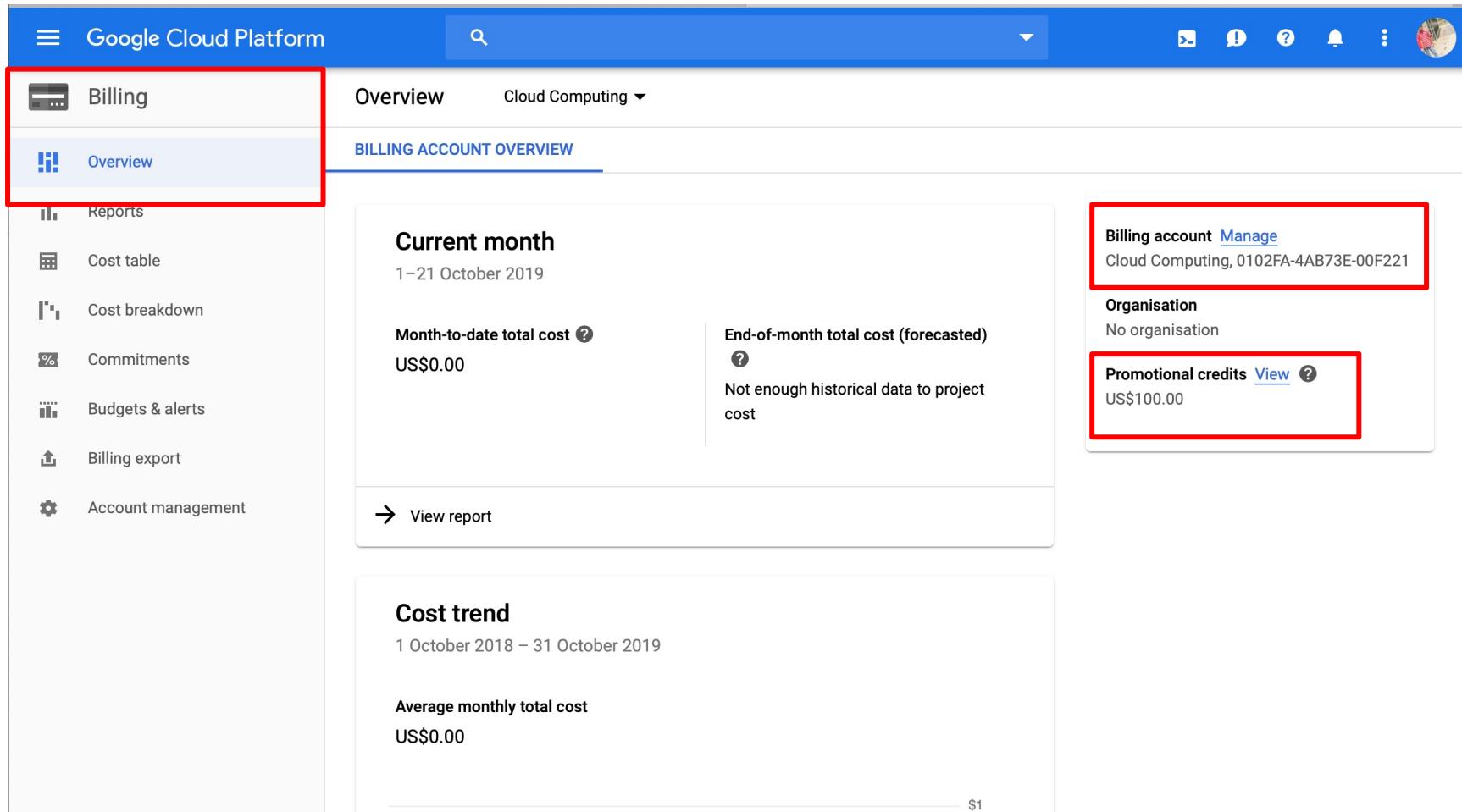


- Verify the email id and you will then get the coupon code to your email.
- Click the link in the new email from GCP to redeem it .

# Login into GCP : Home page



Below is the homepage displayed, after logging into GCP



The screenshot shows the Google Cloud Platform Billing Overview page. The left sidebar has a red box around the 'Billing' section, which contains 'Overview' (selected), 'Reports', 'Cost table', 'Cost breakdown', 'Commitments', 'Budgets & alerts', 'Billing export', and 'Account management'. The main content area has a red box around the 'BILLING ACCOUNT OVERVIEW' section. It displays the 'Current month' (1–21 October 2019), 'Month-to-date total cost' (US\$0.00), and 'End-of-month total cost (forecasted)' (US\$0.00). A note says 'Not enough historical data to project cost'. Below this is a 'View report' button. To the right, there's a 'Billing account Manage' section (Cloud Computing, 0102FA-4AB73E-00F221) and a 'Promotional credits View' section (US\$100.00). The bottom part shows a 'Cost trend' chart from 1 October 2018 to 31 October 2019, with an average monthly total cost of US\$0.00.

# Start a VM: Step1

The screenshot shows the Google Cloud Platform Billing dashboard. A red arrow points from the top-left corner to the three-line menu icon in the top-left corner of the main content area. Another red arrow points from the top-left corner to the 'Billing' link in the left sidebar. A third red arrow points from the top-left corner to the 'VM instances' link in the 'Compute' menu dropdown.

**Billing account:** Manage  
Cloud Computing, 0102FA-4AB73E-00F221

**Organisation:** No organisation

**Promotional credits:** View ?  
US\$100.00

**End-of-month total cost (forecasted):** ?  
Not enough historical data to project cost

**Average monthly total cost:** US\$0.00

**October 2019**

**VM instances**

- Instance groups
- Instance templates
- Sole-tenant nodes
- Disks
- Snapshots
- Images
- TPUs
- Committed use discounts
- Metadata
- Health checks
- Zones
- Network endpoint groups
- Operations
- Security scans
- Settings

**Compute**

- App Engine**
- Compute Engine**
- Kubernetes Engine**
- Cloud Functions**

**Home**

**Marketplace**

**Billing**

**APIs & Services**

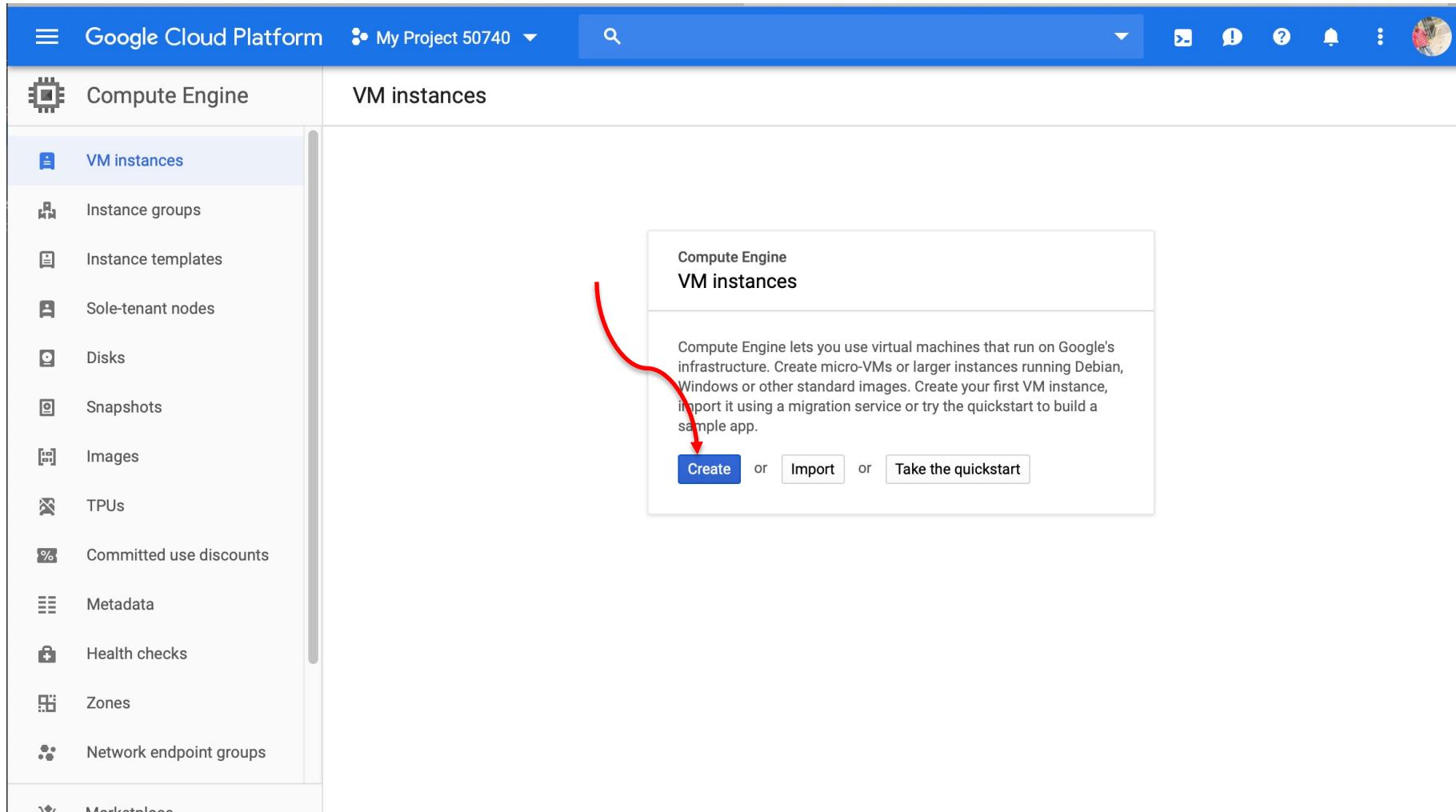
**Support**

**IAM & admin**

**Getting started**

**Security**

# Start a VM: Step2



The screenshot shows the Google Cloud Platform Compute Engine VM instances page. The left sidebar lists various Compute Engine services: VM instances, Instance groups, Instance templates, Sole-tenant nodes, Disks, Snapshots, Images, TPUs, Committed use discounts, Metadata, Health checks, Zones, Network endpoint groups, and Marketplace. The 'VM instances' link is highlighted in blue. The main content area is titled 'Compute Engine VM instances' and contains a brief description of what Compute Engine is and how to get started. At the bottom of this section are three buttons: 'Create', 'Import', and 'Take the quickstart'. A red arrow points from the 'Create' button in the main content area to the 'Create' link in the sidebar menu.

# Start a VM: Step 3

## Create an instance

To create a VM instance, select one of the options:

### New VM instance

Create a single VM instance from scratch

### New VM instance from template

Create a single VM instance from an existing template

### Marketplace

Deploy a ready-to-go solution onto a VM instance

Name ?  
instance-1

Region ? Zone ?  
europe-west3 (Frankfurt) europe-west3-c

Machine configuration ?

Machine family  
**General-purpose**

Machine types for common workloads, optimised for cost and flexibility

Generation  
First  
Powered by Skylake CPU platform or one of its predecessors

Machine type  
**g1-small (1 vCPU, 1.7 GB memory)**

vCPU  
1 shared core      Memory  
1.7 GB

CPU platform and GPU

Container ?  
 Deploy a container image to this VM instance. [Learn more](#)

Boot disk ?

New 10 GB standard persistent disk  
Image  
Ubuntu 18.04 LTS      Change

Identity and API access ?

Service account ?  
Compute Engine default service account

Select region and zone

\$17.04 monthly estimate

That's about \$0.023 hourly

Pay for what you use: No upfront costs and per second billing

Details

Select machine type

Select OS and Disk space

# Start a VM: Step4

Identity and API access [?](#)

Service account [?](#)  
Compute Engine default service account

Access scopes [?](#)  
 Allow default access  
 Allow full access to all Cloud APIs  
 Set access for each API

**Firewall** [?](#)  
Add tags and firewall rules to allow specific network traffic from the Internet.  
 Allow HTTP traffic  
 Allow HTTPS traffic

**Availability policy**

**Preemptibility**  
A preemptible VM costs much less, but lasts only 24 hours. It can be terminated sooner due to system demands. [Learn more](#).

Off (recommended)

**On host maintenance**  
When Compute Engine performs periodic infrastructure maintenance, it can migrate your VM instances to other hardware without downtime

Migrate VM instance (recommended)

**Automatic restart**  
Compute Engine can automatically restart VM instances if they are terminated for non-user-initiated reasons (maintenance event, hardware failure, software failure and so on)

On (recommended)

[Less](#)

You will be billed for this instance. [Compute Engine pricing](#) ↗

**Create** **Cancel**

Equivalent REST or command line



# Start a VM: Step5

- SSH into VM

The screenshot shows the Google Cloud Compute Engine interface for managing VM instances. On the left, a sidebar menu includes 'Compute Engine' (selected), 'VM instances' (selected), 'Instance groups', 'Instance templates', and 'Cloud Build triggers'. The main area is titled 'VM instances' with a 'CREATE INSTANCE' button and various management icons. A search bar labeled 'Filter VM instances' and a 'Columns' dropdown are also present. A table lists VM instances, with one row highlighted by a red box and a red arrow pointing to the 'SSH' button in the 'Connect' column. The table columns are: Name, Zone, Recommendation, In use by, Internal IP, External IP, and Connect. The highlighted row shows 'instance-1' in the 'Name' column, 'europe-west3-c' in the 'Zone' column, and the 'SSH' button in the 'Connect' column.

Name	Zone	Recommendation	In use by	Internal IP	External IP	Connect
instance-1	europe-west3-c			10.156.0.2 (nic0)	35.242.206.67	SSH

# Run and access the application

- Run the given application as mentioned in the instructions.
- Go to URL : [http://VM\\_PUBLIC\\_IP:80](http://VM_PUBLIC_IP:80)

The screenshot shows a web browser window with the following details:

- Address Bar:** Not Secure — 35.242.206.67
- Navigation Bar:** My Library Portal, Home, API Documentation
- Form Fields (Top Row):**
  - Book Title: (input field)
  - Book Author: (input field)
  - Release Date: (input field)
- Form Fields (Second Row):**
  - Genre: Non-Fiction (dropdown menu)
  - Rating: (dropdown menu)
  - Language of book: (input field)
- Buttons:** Submit (blue button)

## All Stored Books Information

# Run and access the completed application

- Follow the same instructions to run the application.
- Go to URL : [http://VM\\_PUBLIC\\_IP:80](http://VM_PUBLIC_IP:80) and test the queries.

The screenshot shows a web browser window with the address bar displaying "Not Secure — 35.242.206.67". The navigation bar includes "My Library Portal", "Home", and "API Documentation". The main content area contains a form for adding a book:

Book Title: abc	Book Author: asd	Release Date: ad
Genre: Fiction	Rating: 2	Language of book: adad
<input type="button" value="Submit"/>		

## All Stored Books Information

Title : abcsd

Author: asddd

Release Date: addd

Genre: fiction

Rating: 2d

Language: adadd

[Delete](#)

[Edit](#)

Title : abc

Author: asd

Release Date: ad

Genre: fiction

Rating: 2

Language: adad

[Delete](#)

[Edit](#)

# Further Information

- There are many source code management systems out there -- But we recommend you to use GitLab at LRZ [https://gitlab.lrz.de/users/sign\\_in](https://gitlab.lrz.de/users/sign_in)
- Editors (You can get the student licensed version for them):
  - WebStorm <https://www.jetbrains.com/webstorm/>
  - Visual Studio Code <https://code.visualstudio.com/>
- For further Node.js learning you check this  
[https://www.youtube.com/watch?v=-u-j7uqU7sI&list=PL6gx4Cwl9DGBMdKFn3HasZnnAqVjzHn\\_&index=1](https://www.youtube.com/watch?v=-u-j7uqU7sI&list=PL6gx4Cwl9DGBMdKFn3HasZnnAqVjzHn_&index=1) tutorial

# AWS VM Creation

# AWS Management Console



aws Services Resource Groups 🔍 🔔 Login Name Oregon Support

**List of services**

Find a service by name or feature (for example, EC2, S3, or V/M storage)

Recently visited services

- CodeStar
- Billing
- EC2
- IAM
- VPC

All services

Build a solution

Get started with simple wizards and automated workflows.

- Launch a virtual machine  
With EC2  
~2-3 minutes
- Build a web app  
With Elastic Beanstalk  
~6 minutes
- Build using virtual servers  
With Lightsail  
~1-2 minutes
- Connect an IoT device  
With AWS IoT
- Start a development project  
With CodeStar
- Register a domain  
With Route 53  
~3 minutes

See more

**Current Region**

Helpful tips

- Manage your costs  
Monitor your AWS costs, usage, and reservations using AWS Budgets. [Start now](#)
- Create an organization  
Use AWS Organizations for policy-based management of multiple AWS accounts. [Start now](#)

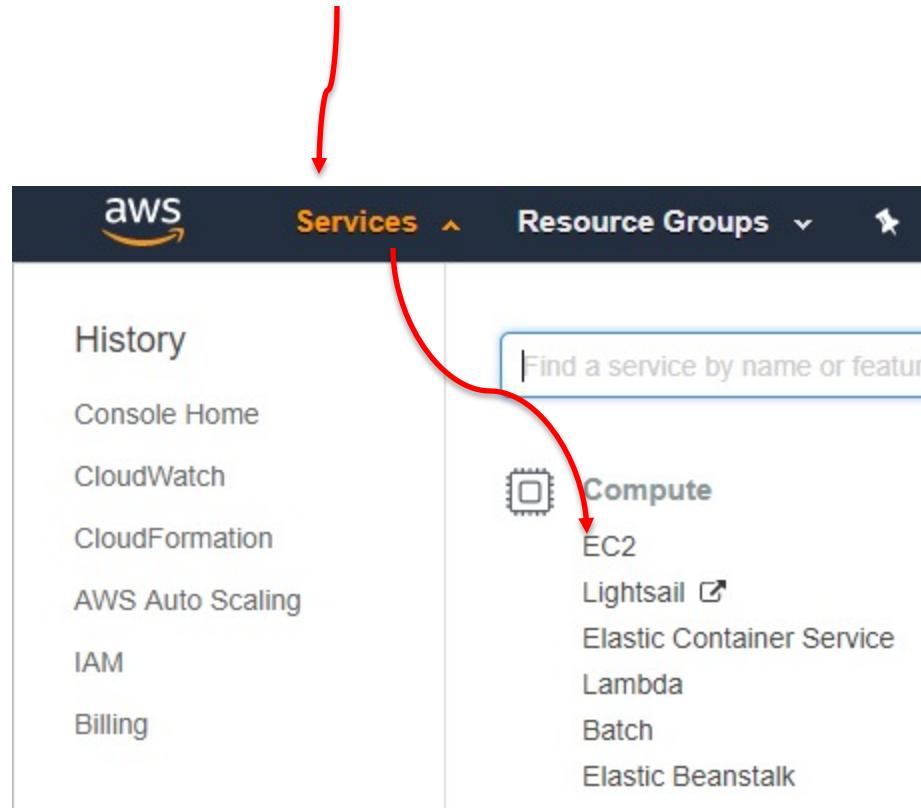
Explore AWS

Machine Learning with Amazon SageMaker  
The fastest way to build, train, and deploy machine learning models. [Learn more](#)

Amazon Relational Database Service (RDS)  
RDS manages and scales your database for

# Start EC2 Instance

4. Now we will be starting the EC2 instance. Click on services and choose EC2



# EC2 Dashboard

- After clicking on EC2, you would go into EC2 dashboard. Now here click on **Launch Instance**

The screenshot shows the AWS EC2 Dashboard. The left sidebar has a navigation menu with the following items:

- EC2 Dashboard** (selected)
- Events
- Tags
- Reports
- Limits
- INSTANCES** (expanded)
- Instances
- Launch Templates
- Spot Requests
- Reserved Instances
- Dedicated Hosts
- Scheduled Instances
- IMAGES** (expanded)
- AMIs
- Bundle Tasks
- ELASTIC BLOCK STORE** (expanded)

The main content area is titled "Resources" and displays the following summary:

Amazon EC2 Resources	Count
Running Instances	0
Dedicated Hosts	0
Volumes	0
Key Pairs	1
Placement Groups	0

A callout box in the center says: "Learn more about the latest in AWS Compute from AWS viewing the EC2 Videos."

The "Create Instance" section contains the following text and a "Launch Instance" button:

To start using Amazon EC2 you will want to launch a virtual EC2 instance.

**Launch Instance**

A large red curved arrow points from the text "Now here click on **Launch Instance**" in the slide notes to the "Launch Instance" button on the dashboard.

# Launching an EC2 Instance : Step 1 (Selecting AMI)



## 6. Select an AMI. We will be selecting here **Ubuntu Server 18.04 LTS (HVM), SSD Volume Type**

### Step 1: Choose an Amazon Machine Image (AMI)

[Cancel and Exit](#)

AMI Details		
Red Hat	Red Hat Enterprise Linux 7.5 (HVM), SSD Volume Type - ami-28e07e50 Red Hat Enterprise Linux version 7.5 (HVM), EBS General Purpose (SSD) Volume Type	<a href="#">Select</a> 64-bit (x86)
SUSE Linux	SUSE Linux Enterprise Server 15 (HVM), SSD Volume Type - ami-0de02b68de6f5f732 SUSE Linux Enterprise Server 15 (HVM), EBS General Purpose (SSD) Volume Type. Public Cloud, Advanced Systems Management, Web and Scripting, and Legacy modules enabled.	<a href="#">Select</a> 64-bit (x86)
Ubuntu	Ubuntu Server 18.04 LTS (HVM), SSD Volume Type - ami-0bbe6b35405ecebdb Ubuntu Server 18.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical ( <a href="http://www.ubuntu.com/cloud/services">http://www.ubuntu.com/cloud/services</a> ).	<a href="#">Select</a> 64-bit (x86)

# Launching an EC2 Instance : Step 2 (Instance Type)



7. Choose an Instance type. We will be selecting here **t2.micro**. For instance types list check [here](#).

## 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	IPv6 Support
<input type="checkbox"/>	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
<input checked="" type="checkbox"/>	General purpose	t2.micro	1	1	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	General purpose	t2.small	1	2	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	General purpose	t2.large	2	8	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	General purpose	t2.xlarge	4	16	EBS only	-	Moderate	Yes

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

# Launching an EC2 Instance : Step 3 (Instance Config.)



8. Configure the Instance details. You can leave everything to default.

## Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of instances  Launch into Auto Scaling Group [\(i\)](#)

Purchasing option [\(i\)](#)  Request Spot instances

Network [\(i\)](#) vpc-73810c0b (default) [\(i\)](#) [Create new VPC](#)

Subnet [\(i\)](#) No preference (default subnet in any Availability Zone) [\(i\)](#) [Create new subnet](#)

Auto-assign Public IP [\(i\)](#) Use subnet setting (Enable) [\(i\)](#)

IAM role [\(i\)](#) None [\(i\)](#) [Create new IAM role](#)

Shutdown behavior [\(i\)](#) Stop [\(i\)](#)

Enable termination protection [\(i\)](#)  Protect against accidental termination

Monitoring [\(i\)](#)  Enable CloudWatch detailed monitoring  
Additional charges apply.

Tenancy [\(i\)](#) Shared - Run a shared hardware instance [\(i\)](#)  
Additional charges will apply for dedicated tenancy.

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Add Storage](#)

# Launching an EC2 Instance : Step 4 (Add Storage)



- Next step is to add the storage to the VM, we will leave here also to default **8GiB SSD**.

## Step 4: Add Storage

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. [Learn more](#) about storage options in Amazon EC2.

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination	Encrypted
Root	/dev/sda1	snap-03c91645beefa0b0d	8	General Purpose S	100 / 3000	N/A	<input checked="" type="checkbox"/>	Not Encrypted

**Add New Volume**

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. [Learn more](#) about free usage tier eligibility and usage restrictions.

[Cancel](#) [Previous](#) **Review and Launch** [Next: Add Tags](#)

# Launching an EC2 Instance : Step 5 (Add Tags)

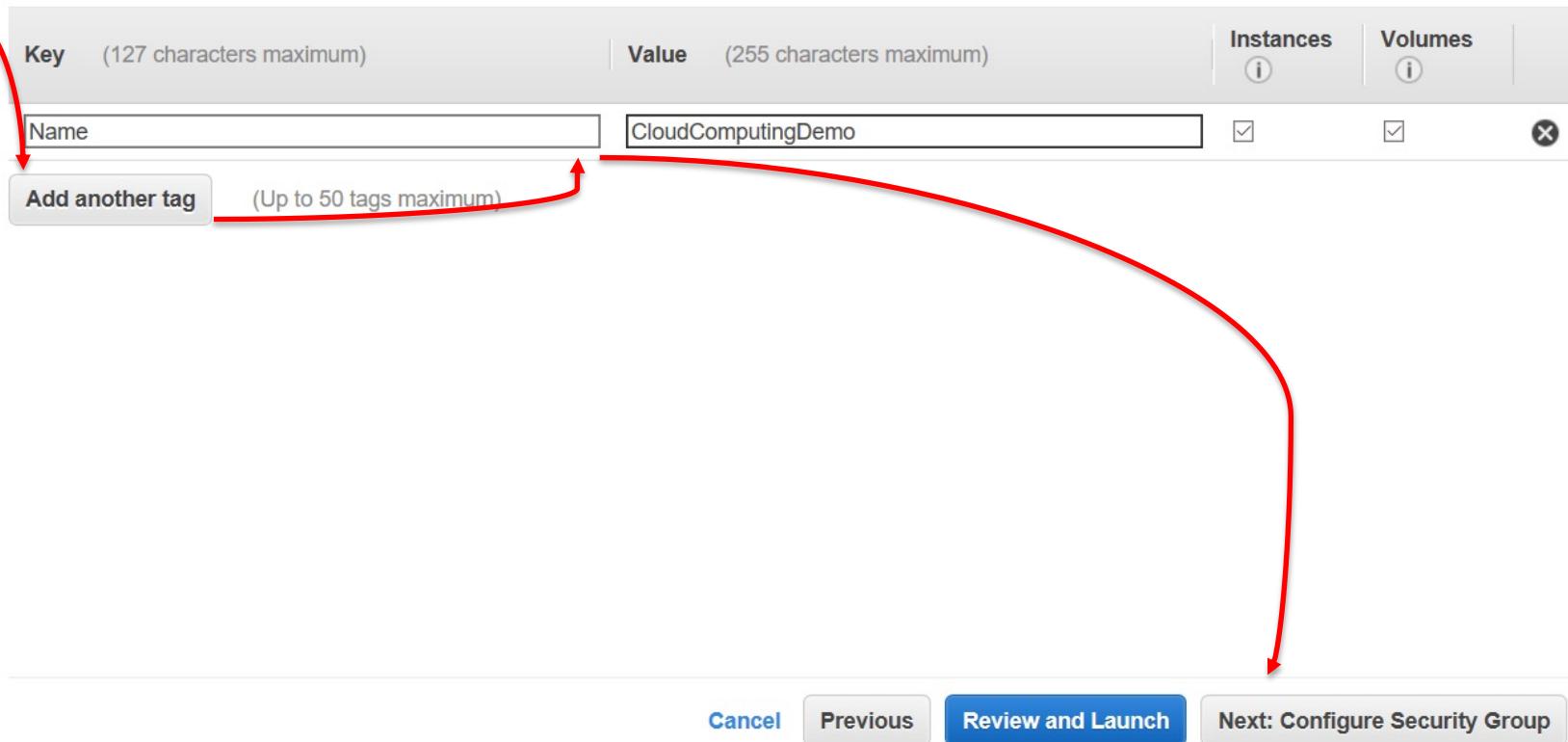
10. Here we can add tag to our Instance. It's key value pair. We will be adding one tag which is the **Name** as shown below. You can give any name.

## Step 5: Add Tags

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.

A copy of a tag can be applied to volumes, instances or both.

Tags will be applied to all instances and volumes. [Learn more](#) about tagging your Amazon EC2 resources.



The screenshot shows the 'Add Tags' step of the EC2 instance launch wizard. A red arrow points from the 'Name' key input field to the 'Add another tag' button. Another red arrow points from the 'CloudComputingDemo' value input field to the 'Review and Launch' button. A large red curved arrow starts from the bottom right and points towards the 'Add another tag' button.

Key (127 characters maximum)	Value (255 characters maximum)	Instances <span style="font-size: small;">i</span>	Volumes <span style="font-size: small;">i</span>
Name	CloudComputingDemo	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Add another tag (Up to 50 tags maximum)

Cancel Previous Review and Launch Next: Configure Security Group

# Launching an EC2 Instance : Step 6 (Configure SG)



11. We will create here a security group

- Allow from all IPs the SSH port **22**
- Allow from all IPs the port **80** (Our Application will be hosted on this port)

## 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 name:

Description:

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop
Custom TCP I	TCP	80	Anywhere 0.0.0.0/0, ::/0	App access

Add Rule

**Warning**

Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

**Cancel** **Previous** **Review and Launch**

# Launching an EC2 Instance : Step 6 (Review & Launch)

12. Now all the configuration is done, you can review your configuration. After review click on 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.

 **Improve your instances' security. Your security group, launch-wizard-1, is open to the world.**  
Your instances may be accessible from any IP address. We recommend that you update your security group rules to allow access from known IP addresses only.  
You can also open additional ports in your security group to facilitate access to the application or service you're running, e.g., HTTP (80) for web servers. [Edit security groups](#)

**AMI Details** [Edit AMI](#)

 **Ubuntu Server 18.04 LTS (HVM), SSD Volume Type - ami-0bbe6b35405ecebdb**  
**Free tier eligible** Ubuntu Server 18.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).  
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

[Cancel](#)

[Previous](#)

[Launch](#)

# Launching an EC2 Instance : Step 7 (Download Key)



13. If you don't have already a key pair **create a new one** and download the pem file. You can give any name to this file.

Select an existing key pair or create a new key pair X

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: 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](#).

▼

**Key pair name**

**Tip:** 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.

---

# Launching an EC2 Instance : Instance Started



The Instance has started, click on the name to see it's status

## Launch Status



Your instances are now launching

The following instance launches have been initiated: [i-05becf8f26e261627](#) [View launch log](#)



# Launching an EC2 Instance : Instance Status

Check for the instance status and it's public ip.

The screenshot shows the AWS EC2 Instances page. A single instance is listed:

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks
CloudComputingDemo	i-05becf8f26e261627	t2.micro	us-west-2a	running	Initializing

The instance details page for i-05becf8f26e261627 is shown. The instance state is running, and its public DNS is ec2-34-217-36-129.us-west-2.compute.amazonaws.com. The public IP is 34.217.36.129. The private IP is 172.31.39.153. The instance type is t2.micro, and it is located in the us-west-2a availability zone. It is associated with the security group launch-wizard-1.

Instance ID	i-05becf8f26e261627
Instance state	running
Instance type	t2.micro
Elastic IPs	
Availability zone	us-west-2a
Security groups	launch-wizard-1 <a href="#">view inbound rules</a> <a href="#">view outbound rules</a>
Public DNS (IPv4)	ec2-34-217-36-129.us-west-2.compute.amazonaws.com
IPv4 Public IP	34.217.36.129
IPv6 IPs	-
Private DNS	ip-172-31-39-153.us-west-2.compute.internal
Private IPs	172.31.39.153
Secondary private IPs	

# Launching an EC2 Instance : Instance SSH Login (Linux)

For Logging into instance

- Linux or mac:
  - ssh -i "KEY\_FILE\_PATH" ubuntu@PUBLIC\_IP

For Example:

```
ssh -i "aws_key.pem" ubuntu@34.217.36.219
```

# Launching an EC2 Instance : Instance SSH Login (Win.)



## For Windows:

- Download **Putty** from here

<https://www.chiark.greenend.org.uk/~sgtatham/putty/>

- Download **PuttyGen**

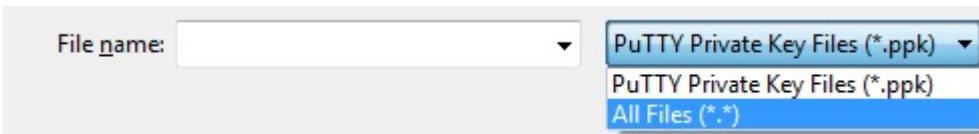
- 32 bit <https://the.earth.li/~sgtatham/putty/latest/w32/puttygen.exe>
- 64 bit <https://the.earth.li/~sgtatham/putty/latest/w64/puttygen.exe>

- After Download open puttygen

- Under **Type of key to generate**, choose **RSA**

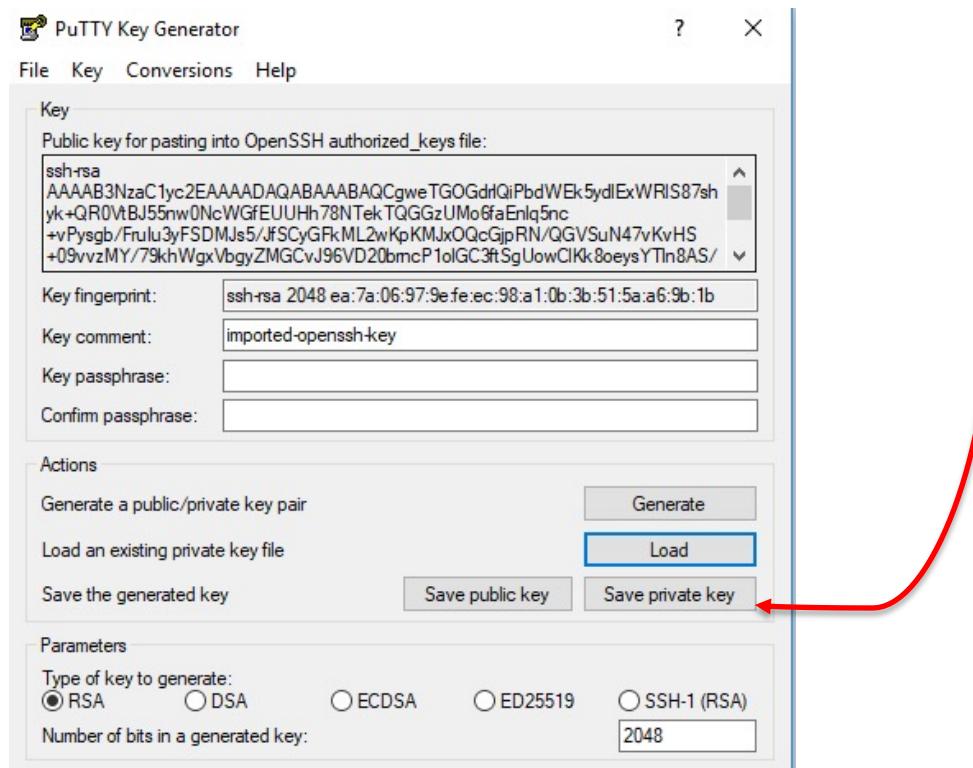


- Choose Load. By default, PuTTYgen displays only files with the extension .ppk. To locate your .pem file, select the option to display files of all types.



# Launching an EC2 Instance : Instance SSH Login (Win.)

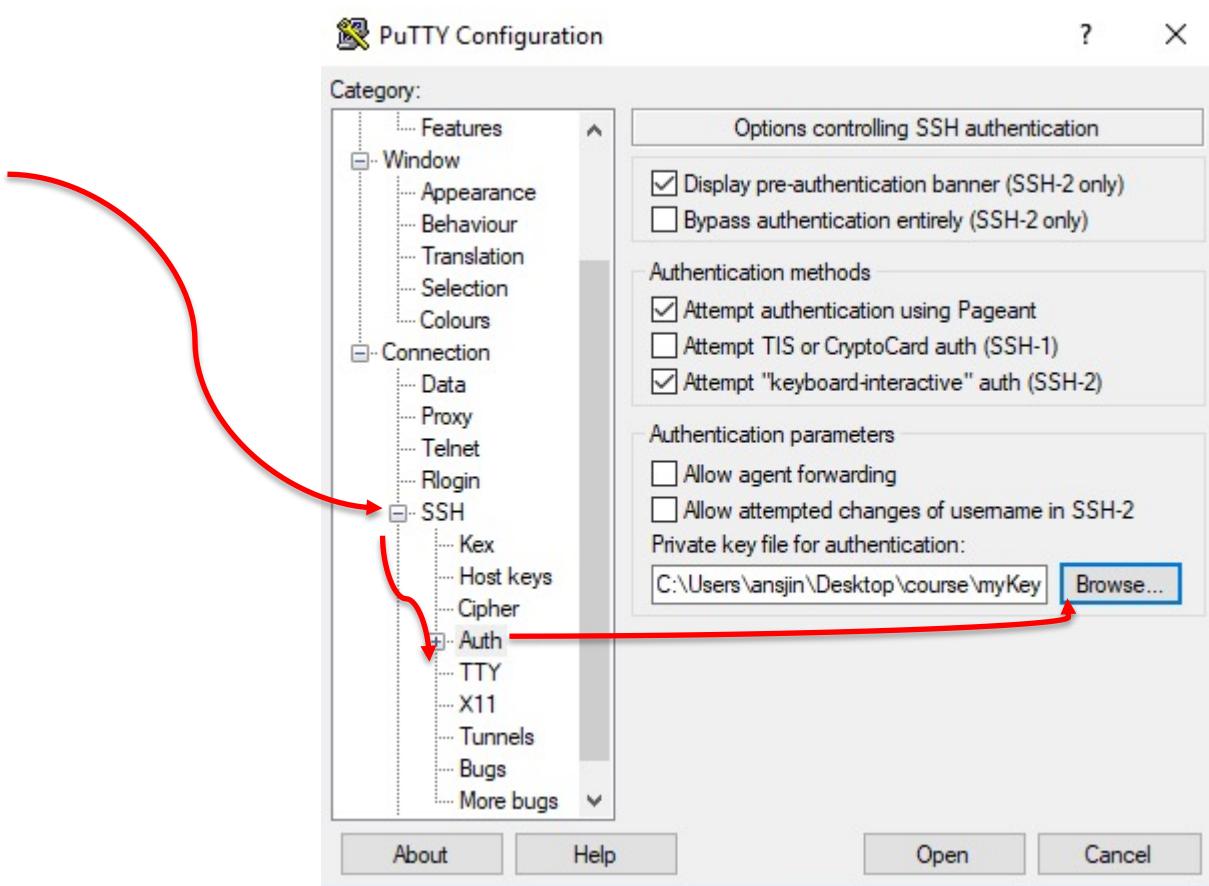
- Select your .pem file for the key pair that you specified when you launched your instance, and then choose Open. Choose OK to dismiss the confirmation dialog box.
- Choose **Save private key** to save the key in the format that PuTTY can use. PuTTYgen displays a warning about saving the key without a passphrase. Choose **Yes**.



# Launching an EC2 Instance : Instance SSH Login (Win.)

- Specify the same name for the key that you used for the key pair (for example, my-key-pair). PuTTY automatically adds the .ppk file extension.

- Now, Open Putty
- Click on **SSH - > Auth** on left panel and browse for the ppk file



# Launching an EC2 Instance : Instance SSH Login (Win.)

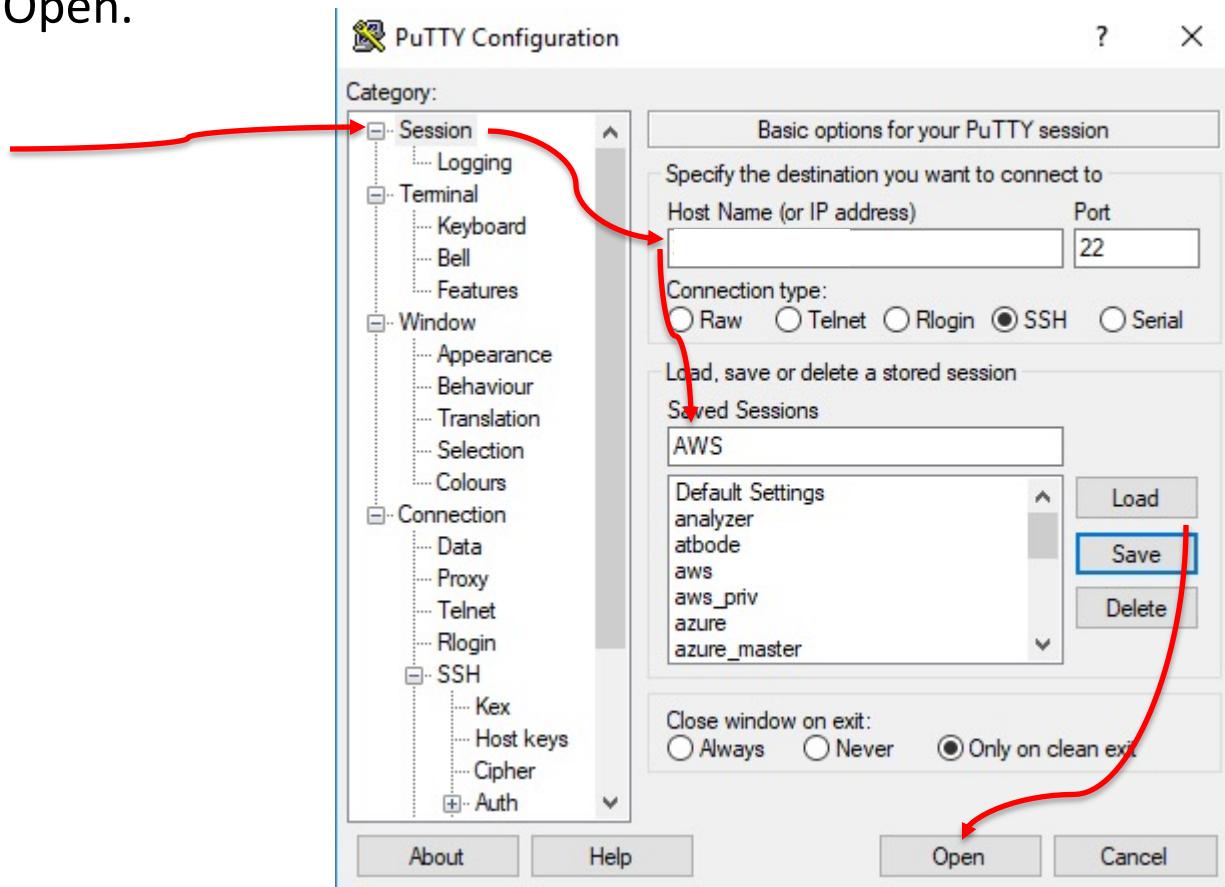


3.Click on Session

4.Add the **hostname** which is the **public IP** of your instance

5.Write a name to save the configuration for future use, click save

6.Click Open.



7.Use username  
**ubuntu**

# Instance Stop/Reboot/Termination

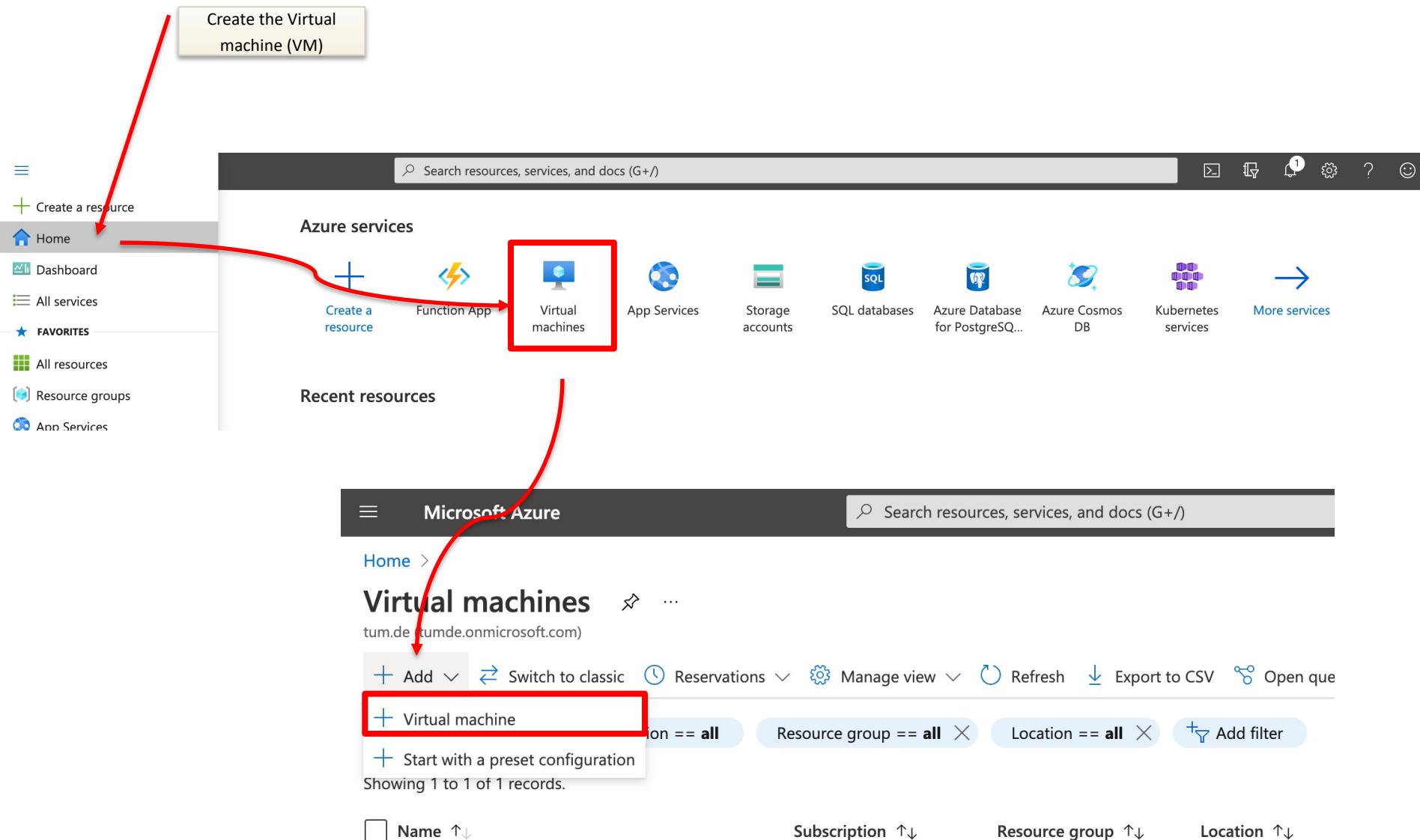
You can either stop or reboot or terminate the instance from the EC2 Dashboard

The screenshot shows the AWS EC2 Dashboard. On the left, there's a sidebar with various navigation links. A red arrow points from the 'Instances' link in the sidebar to the 'Actions' button in the main header area. Another red arrow points from the 'Actions' button to the open dropdown menu. The dropdown menu is titled 'Actions' and contains several options: Connect, Get Windows Password, Create Template From Instance, Launch More Like This, Instance State, Instance Settings, Image, Networking, and CloudWatch Monitoring. The 'Instance State' option is highlighted in orange. Below the dropdown, the instance details for 'i-05becf8f26e261627' are displayed, including its Public DNS and IP addresses. The instance is currently running.

Description	Value	Description	Value
Instance ID	i-05becf8f26e261627	Public DNS (IPv4)	ec2-34-217-36-129.us-west-2.compute.amazonaws.com
Instance state	running	IPv4 Public IP	34.217.36.129
Instance type	t2.micro	IPv6 IPs	-
Elastic IPs		Private DNS	ip-172-31-39-153.us-west-2.compute.internal
Availability zone	us-west-2a	Private IPs	172.31.39.153
Security groups	launch-wizard-1, view inbound rules, view outbound	Secondary private IPs	

# Azure VM Creation

# Azure Dashboard : Step 1 (Create VM)



# Azure Create VM : Step 2 (Create a Resource Group)



Home > Virtual machines >

## Virtual machines

tum.de (tumde.onmicrosoft.com)

+ Add Switch to classic ...

Filter for any field...

Name ↑↓

Subscription ↑↓



No virtual machines to display

Create a virtual machine that runs Linux or Windows.  
Select an image from the marketplace or use your own  
customized image.

Learn more about Windows virtual machines ↗ Learn  
more about Linux virtual machines ↗

## Create a virtual machine

Basics

Disks

Networking

Management

Advanced

Tags

Review + create

Create a virtual machine that runs Linux or Windows. Select an image from Azure marketplace or use your own customized image. Complete the Basics tab then Review + create to provision a virtual machine with default parameters or review each tab for full customization. [Learn more ↗](#)

### Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \* ⓘ

Azure for Students

Resource group \* ⓘ

(New) Resource group

Create new

### Instance details

Virtual machine name \* ⓘ

A resource group is a container that holds related resources for an Azure solution.

Name \*

cloud-computing-ex1

OK

Cancel

Region \* ⓘ

Availability options ⓘ

Image \* ⓘ

Azure Spot instance ⓘ

Size \* ⓘ

### Administrator account

Authentication type ⓘ

SSH public key

Password

Create a resource group. It contains all the resources belonging to the resource i.e VM, we are creating

# Azure Create VM : Step 3 (Select region, VM Image and Size)

## Create a virtual machine

Create a virtual machine that runs Linux or Windows. Select an image from Azure marketplace or use your own customized image. Complete the Basics tab then Review + create to provision a virtual machine with default parameters or review each tab for full customization. [Learn more](#)

### Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \* ⓘ



Azure for Students



Resource group \* ⓘ

cloud-computing-ex1



[Create new](#)

### Instance details

Virtual machine name \* ⓘ

cc-ex1

[Give a name to VM](#)

Region \* ⓘ

(Europe) Germany West Central

Select the region where the VM will be created.

Availability options ⓘ

No infrastructure redundancy required



Image \* ⓘ

Ubuntu Server 18.04 LTS - Gen1

Select the VM operating system image (here it is Ubuntu 18.04)

[See all images](#)

Azure Spot instance ⓘ



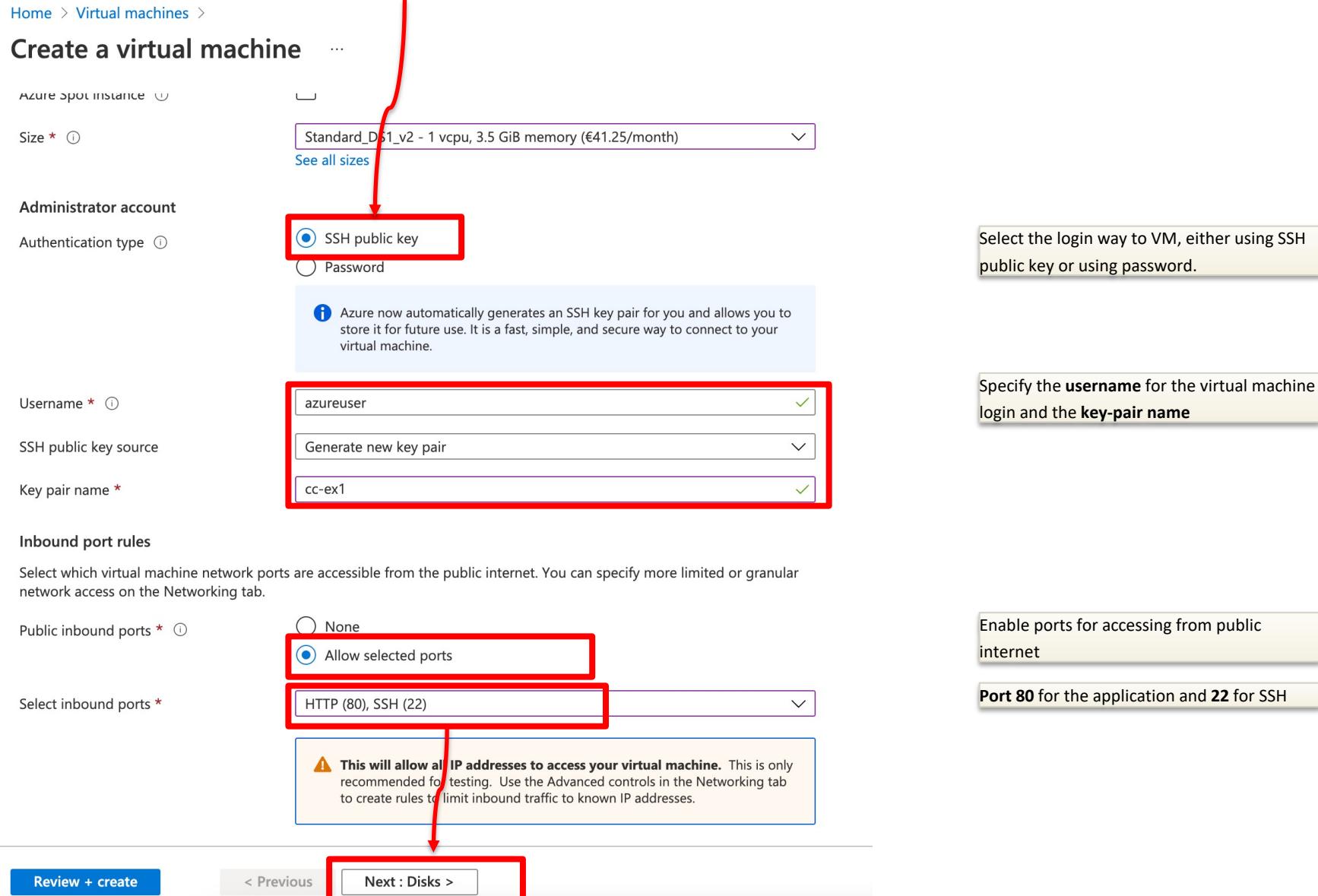
Size \* ⓘ

Standard\_DS1\_v2 - 1 vcpu, 3.5 GiB memory (€41.25/month)

Select the size of VM i.e the number of cores and RAM allocated to VM

[See all sizes](#)

# Azure Create VM : Step 4 (Select SSH key and enable ports)



The screenshot shows the Azure portal interface for creating a virtual machine. A red arrow points from the top right towards the 'SSH public key' section. Another red box highlights the 'SSH public key' radio button, which is selected. A callout box states: 'Select the login way to VM, either using SSH public key or using password.' Below this, a red box highlights the 'Username' field containing 'azureuser', the 'Generate new key pair' dropdown, and the 'Key pair name' field containing 'cc-ex1'. A callout box says: 'Specify the username for the virtual machine login and the key-pair name.' A red box highlights the 'Allow selected ports' radio button under 'Public inbound ports', and another red box highlights the 'HTTP (80), SSH (22)' dropdown. A callout box says: 'Enable ports for accessing from public internet' and 'Port 80 for the application and 22 for SSH'. A red arrow points from the bottom right towards the 'Next : Disks >' button.

Home > Virtual machines >

## Create a virtual machine

Azure Spot instance 

Size \*  Standard\_D61\_v2 - 1 vcpu, 3.5 GiB memory (€41.25/month)   
[See all sizes](#)

Administrator account

Authentication type  SSH public key  Password

 Azure now automatically generates an SSH key pair for you and allows you to store it for future use. It is a fast, simple, and secure way to connect to your virtual machine.

Username \*  azureuser 

SSH public key source  Generate new key pair 

Key pair name \* cc-ex1 

Inbound port rules

Select which virtual machine network ports are accessible from the public internet. You can specify more limited or granular network access on the Networking tab.

Public inbound ports \*  None  Allow selected ports

Select inbound ports \*  HTTP (80), SSH (22)

 This will allow all IP addresses to access your virtual machine. This is only recommended for testing. Use the Advanced controls in the Networking tab to create rules to limit inbound traffic to known IP addresses.

[Review + create](#) [< Previous](#) [Next : Disks >](#)

# Azure Create VM : Step 5 (Select OS Disk type)



Home > Virtual machines >

## Create a virtual machine

Basics Disks Networking Management Advanced Tags Review + create

Azure VMs have one operating system disk and a temporary disk for short-term storage. You can attach additional data disks. The size of the VM determines the type of storage you can use and the number of data disks allowed. [Learn more](#)

**Disk options**

OS disk type \* ⓘ Standard HDD

The selected VM size supports premium disks. We recommend Premium SSD for high IOPS workloads. Virtual machines with Premium SSD disks qualify for the 99.9% connectivity SLA.

Encryption type \* ⓘ (Default) Encryption at-rest with a platform-managed key

Enable Ultra Disk compatibility ⓘ  Ultra disk is not supported for the selected VM size Standard\_DS1\_v2 in germanywestcentral.

**Data disks**

You can add and configure additional data disks for your virtual machine or attach existing disks. This VM also comes with a temporary disk.

LUN	Name	Size (GiB)	Disk type	Host caching

Create and attach a new disk Attach an existing disk

Advanced

Use managed disks ⓘ

Ephemeral OS disk ⓘ

[Review + create](#) < Previous Next : Networking >

Select the **OS disk type**, you can choose SSD or HDD. As we are not hosting any production application, so here HDD is selected to save some money.

Keep this to default

# Azure Create VM : Step 6 (Network Settings)



Home > Virtual machines >

## Create a virtual machine

Basics Disks Networking Management Advanced Tags Review + create

Define network connectivity for your virtual machine by configuring network interface card (NIC) settings. You can control ports, inbound and outbound connectivity with security group rules, or place behind an existing load balancing solution.  
[Learn more](#)

### Network interface

When creating a virtual machine, a network interface will be created for you.

Virtual network \* ⓘ

(new) cloud-computing-ex1-vnet

[Create new](#)

Subnet \* ⓘ

(new) default (10.0.0.0/24)

[Create new](#)

Public IP ⓘ

(new) ex1-ip

[Create new](#)

NIC network security group ⓘ

None

Basic

Advanced

Public inbound ports \* ⓘ

None

Allow selected ports

Select inbound ports \*

HTTP (80), SSH (22)

Keep this to default, automatically created by Azure

Check the ports 80 and 22 are enabled.

⚠ This will allow all IP addresses to access your virtual machine. This is only recommended for testing. Use the advanced controls in the Networking tab to create rules to limit inbound traffic to known IP addresses.

Accelerated networking ⓘ

The selected VM size does not support accelerated networking.

[Review + create](#)

< Previous

Next : Management >

# Azure Create VM : Step 7 (Management Settings)



Home > Virtual machines >

## Create a virtual machine

Enable basic plan for free (i)

This will apply to every VM in the selected subscription

Enabled by default for adding basic monitoring to your VM

### Monitoring

Boot diagnostics (i)

Enable with managed storage account (recommended)

Enable with custom storage account

Disable

Enable OS guest diagnostics (i)

### Identity

System assigned managed identity (i)

### Azure Active Directory

Login with AAD credentials (Preview) (i)

⚠ This preview capability is not for production use. When you sign in, verify the name of the app on the sign-in screen is "Azure Linux VM sign in" and the IP address of the target VM is correct.

### Auto-shutdown

Enable auto-shutdown (i)

11:00:00 PM

(UTC+01:00) Amsterdam, Berlin, Bern, Rome, Stockholm, Vienna

Time zone (i)

Notification before shutdown (i)

You can enable this feature of Azure, to shutdown the VMs when you will not be using them as shown here at 11:00 PM, just if you forget to shutdown. **This will save credits.**

Review + create

< Previous

Next : Advanced >

# Azure Create VM : Step 8 (Init script)

Home > Virtual machines >

## Create a virtual machine

Advanced

Add additional configuration, agents, scripts or applications via virtual machine extensions or cloud-init.

**Extensions**

Extensions provide post-deployment configuration and automation.

Extensions ⓘ Select an extension to install

**Custom data and cloud init**

Pass a cloud-init script, configuration file, or other data into the virtual machine while it is being provisioned. The data will be saved on the VM in a known location. [Learn more about custom data for VMs](#)

Custom data

Custom data on the selected image will be processed by cloud-init. [Learn more about custom data and cloud init](#)

**Host**

Azure Dedicated Hosts allow you to provision and manage a physical server within our data centers that are dedicated to your Azure subscription. A dedicated host gives you assurance that only VMs from your subscription are on the host, flexibility to choose VMs from your subscription that will be provisioned on the host, and the control of platform maintenance at the level of the host. [Learn more](#)

Host group ⓘ No host group found

Review + create < Previous Next : Tags >

This is used to specify initialization script. It is used if you want to install some software or configure VM in some way after it has started. But we are not using this feature.

# Azure Create VM : Step 9 (Add Tags to VM)



Home > Virtual machines >

## Create a virtual machine

Basics Disks Networking Management Advanced **Tags** Review + create

Tags are name/value pairs that enable you to categorize resources and view consolidated billing by applying the same tag to multiple resources and resource groups. [Learn more about tags ↗](#)

Note that if you create tags and then change resource settings on other tabs, your tags will be automatically updated.

Name ⓘ	Value ⓘ	Resource
<input type="text"/>	:	<input type="text"/> 12 selected

Specify tag name and value for the VM, this is used if you want to specify unique tags to VM. It's helpful if you have 100s of VMs but we are not specifying it.

Review + create

< Previous

Next : Review + create >

# Azure Create VM : Step 10 (Review and Create VM)



Home > Virtual machines >

## Create a virtual machine

Validation passed

Basics Disks Networking Management Advanced Tags **Review + create**

PRODUCT DETAILS

Standard DS v2 by Microsoft [Terms of use](#) [Privacy policy](#) Subscription credits apply ⓘ **0.0565 EUR/hr** [Pricing for other VM sizes](#)

TERMS

By clicking "Create", I (a) agree to the legal terms and privacy statement(s) associated with the Marketplace offering(s) listed above; (b) authorize Microsoft to bill my current payment method for the fees associated with the offering(s), with the same billing frequency as my Azure subscription; and (c) agree that Microsoft may share my contact, usage and transactional information with the provider(s) of the offering(s) for support, billing and other transactional activities. Microsoft does not provide rights for third-party offerings. See the [Azure Marketplace Terms](#) for additional details.

**⚠ You have set SSH port(s) open to the internet.** This is only recommended for testing. If you want to change this setting, go back to Basics tab.

**Basics**

Subscription	Azure for Students
Resource group	(new) cloud-computing-ex1
Virtual machine name	ex1
Region	Germany West Central
Availability options	No infrastructure redundancy required

**Create** < Previous Next > Download a template for automation

Check if validation is passed and everything is same as you have specified. If yes, click on **Create**

# Azure Create VM : Step 11 (Wait for deployment)



The screenshot shows two views of the Azure portal for a deployment named "CreateVm-Canonical.UbuntuServer-18.04-LTS-20210416122318".

**Top View (Deployment in progress):**

- Overview:** Shows the deployment status as "Deployment is in progress".
- Deployment details:** Shows deployment name, subscription, resource group, start time, and correlation ID.
- Next steps:** Includes "Setup auto-shutdown", "Monitor VM health, performance and network dependencies", and "Run a script inside the virtual machine".
- Buttons:** "Go to resource" and "Create another VM".

**Bottom View (Deployment complete):**

- Overview:** Shows the deployment status as "Your deployment is complete".
- Deployment details:** Shows deployment name, subscription, resource group, start time, and correlation ID.
- Next steps:** Includes "Setup auto-shutdown", "Monitor VM health, performance and network dependencies", and "Run a script inside the virtual machine".
- Buttons:** "Go to resource" and "Create another VM".

Red boxes highlight the deployment status messages ("Deployment is in progress" and "Your deployment is complete"). Red arrows point from the "Deployment is in progress" message to the "Wait for the deployment, until it is complete." note, and from the "Your deployment is complete" message to the "Deployment is complete, go to the resource." note.

# Azure Create VM : Step 12 (VM Details)

The screenshot shows the Azure portal interface for a virtual machine named 'ex1'. A red box highlights the 'Virtual machine' icon in the top navigation bar. Another red box highlights the 'Public IP address' field, which contains '20.52.56.18'. A third red box highlights the 'Virtual machine' section under the 'Properties' tab, which lists the following details:

Virtual machine	
Computer name	ex1
Operating system	Linux (ubuntu 18.04)
Publisher	Canonical
Offer	UbuntuServer
Plan	18.04-LTS

Below the properties table, a 'VM properties' button is visible. To the right, there is a 'Networking' section with the following information:

Networking	
Public IP address	20.52.56.18
Public IP address (IPv6)	-
Private IP address	10.0.0.4
Private IP address (IPv6)	-
Virtual network/subnet	cloud-computing-ex1-vnet/default

At the bottom left, a 'Click on Connect' button is shown, with a red arrow pointing towards the 'Connect' button in the top navigation bar.

# Azure Create VM : Step 13 (SSH login to VM)



The screenshot shows the Azure portal interface for a virtual machine named "ex1". A red box highlights the "Connect" button in the top right corner of the main content area. Another red box highlights the "SSH" tab in the navigation bar. A third red box highlights the "Connect via SSH with client" section, which contains step-by-step instructions and a private key path input field. A fourth red box highlights the "SSH login to VM" button at the bottom of the page. A fifth red box highlights the terminal window on the right side of the screen, showing the successful SSH session.

Home > CreateVm-Canonical.UbuntuServer-18.04-LTS-20210416122318 > ex1

**ex1 | Connect** Virtual machine

Search (Cmd+ /) <>

To improve security, enable just-in-time access on this VM. →

RDP SSH BASTION

Follow these instructions for logging to VM

**Connect via SSH with client**

1. Open the client of your choice, e.g. [PuTTY](#) or [other clients](#).
2. Ensure you have read-only access to the private key.  
chmod 400 azureuser.pem
3. Provide a path to your SSH private key file. ⓘ  
Private key path  
~/ssh/azureuser
4. Run the example command below to connect to your VM.  
ssh -i <private key path> azureuser@20.52.56.18

Can't connect?  
Test your connection

SSH login to VM

```
Memory usage: 5%           IP address for eth0: 10.0.0.4
Swap usage:   0%
0 packages can be updated.
0 of these updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

azureuser@ex1:~$
```

# Azure Create VM : Step 14 (Shutdown VM, after completion)

Home > CreateVm-Canonical.UbuntuServer-18.04-LTS-20210416122318 >

**ex1** Virtual machine

Search (Cmd+ /)

Connect Start Restart Stop Capture Delete Refresh Open in mobile

**Overview**

Resource group (change) : cloud-computing-ex1

Status : Running

Location : Germany West Central

Subscription (change) : Azure for Students

Subscription ID : e8f11ca9-cb84-4e8d-af38-9168ec544fba

Tags (change) : Click here to add tags

Operating system : Linux (ubuntu 18.04)

Size : Standard DS1 v2 (1 vcpus, 3.5 GiB memory)

Public IP address : 20.52.56.18

Virtual network/subnet : cloud-computing-ex1-vnet/default

DNS name : Configure

**Properties** Monitoring Capabilities (7) Recommendations Tutorials

**Virtual machine**

Computer name	ex1
Operating system	Linux (ubuntu 18.04)
Publisher	Canonical
Offer	UbuntuServer
Plan	18.04-LTS

**Networking**

Public IP address	20.52.56.18
Public IP address (IPv6)	-
Private IP address	10.0.0.4
Private IP address (IPv6)	-
Virtual network/subnet	cloud-computing-ex1-vnet/default

Shutdown VM to save credits after you are done with exercise

# Azure Create VM : Step 15 optional (deleting a resource Group)



Click on resource group

Resource group

Search (Cmd+ /) Add Edit columns Delete resource group Refresh Export to CSV Delete resource group

Overview

Subscription (change) : Azure for Students  
Subscription ID : e8f11ca9-cb84-4e8d-af38-9168ec544fba  
Tags (change) : Click here to add tags

Filter for any field... Type == all Location == all Add filter

Showing 1 to 7 of 7 records.  Show hidden types

Name	Type
cc-ex1	S
cloud-computing-ex1-vnet	V
ex1	V
ex1-ip	Pi
ex1-nsg	N
ex1686	N
ex1_OsDisk_1_54e0d199ae3c4294aad3aa36de213e88	D

# References

- [1] <http://blog.builtinnode.com/post/a-history-of-node-js>
- [2] <https://nodejs.org/en/about/>
- [3] <https://www.paypal-engineering.com/2013/11/22/node-js-at-paypal/>
- [4] <https://www.npmjs.com/about>
- [5] <https://restfulapi.net/rest-architectural-constraints/>
- [6] <https://docs.mongodb.com/manual/introduction/>

Thank you for your attention!  
Please post questions on Piazza