



‘येथे बहुतांचे हित ।’

Marathwada Mitramandal's
COLLEGE OF ENGINEERING

Karvenagar, PUNE – 411 052

*Accredited with "A++" Grade by NAAC // Accredited by NBA (Mechanical Engg. & Electrical Engg.)
Recipient of "Best College Award 2019" by SPPU // Recognized under section 2(f) and 12B of UGC Act 1956*

DEPARTMENT OF INFORMATION TECHNOLOGY

LABORATORY MANUAL

TE (INFORMATION TECHNOLOGY)

(SEMESTER – II)

**LABORATORY PRACTICE – II
[WEB APPLICATION DEVELOPMENT]**

2019 Course

**Prepared By
Dr. Bharati P. Vasgi**

Preface

Web design refers to the design of websites that are displayed on the internet. It usually refers to the user experience aspects of website development rather than software development.

Web design is the creation of websites and pages to reflect a company's brand and information and ensure a user-friendly experience. Appearance and design are incorporated as vital elements whether you're designing a website, mobile app or maintaining content on a web page. Gaining web design skills can help you in applying for roles where your creativity could help a business improve their brand, their message and their bottom line.

These development course include:

1. Design of web pages and web site using HTML, CSS and Bootstrap.
2. Design and development of web pages using JavaScript.
3. Design of web application using AJAX
4. Design of application using Angular, React, Node JS, Express JS, MongoDB
5. This lab covers the project deployment on cloud platform.

Marathwada Mitramandal's
COLLEGE OF ENGINEERING

S.No.18, Plot No.5/3, Karvenagar, Pune-52
Accredited with "A++" Grade by NAAC | Recipient of "Best College Award 2019" by SPPU
Accredited by NBA (Mechanical Engg. & Electrical Engg.)

Vision of the Institute

- To aspire for the welfare of society through excellence in science and technology.

Mission of the Institute

- Mould young talent for higher endeavors.
 - Meet the challenges of globalization.
 - Commit for social progress with values and ethics.
 - Orient faculty and students for research and development.
 - Emphasize excellence in all disciplines.
-

Department of Information Technology

Vision of the Department

- To emerge as a center of excellence in education, research and innovation in Information Technology for enrichment of society.

Mission of the Department

- To cater industry with engineers having theoretical & practical background and competent IT skills.
- To pursue advanced knowledge in the field of information technology.
- To inculcate budding IT engineers with professional and interpersonal skills.

Program Educational Objectives (PEOs)

The students of Information Technology Program after passing out will possess:

1. Adequate knowledge and skills in Information Technology for implementation of complex problems with innovative approaches.
2. Inclination and technical competency towards professional growth in the field of Information Technology.
3. Ethics and value based interpersonal skills to facilitate lifelong learning and societal contributions.

Program Outcomes (POs)

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcomes (PSOs)

Information Technology Graduates will be able to:

1. Develop quality computer applications by applying principles of software engineering.
2. Pursue advancement in the field of data engineering.

Marathwada Mitra Mandal's

COLLEGE OF ENGINEERING

S.No.18, Plot No.5/3, Karvenagar, PUNE -52

Accredited with "A++" Grade by NAAC // Accredited by NBA (Mechanical Engg. & Electrical Engg.)

Recipient of "Best College Award 2019" by SPPU // Recognized under section 2(f) and 12B of UGC Act 1956

DEPARTMENT OF INFORMATION TECHNOLOGY

List of Experiments

Academic Year: 2023-24

Semester: II

Course Code & Name: Lab Practice II (WAD)

Class: TE

Course Coordinator: Dr. Bharati P. Vasgi

Sr. No.	Expt. No.	Experiment Title
<i>As per SPPU Syllabus</i>		
1	1. a	Create a responsive web page which shows the ecommerce/college/exam admin dashboard with sidebar and statistics in cards using HTML, CSS and Bootstrap
2	1. b	Write a JavaScript Program to get the user registration data and push to array/local storage with AJAX POST method and data list in new page.
3	2.a	Create version control account on GitHub and using Git commands to create repository and push your code to GitHub.
4	2.b	Create Docker Container Environment (NVIDIA Docker or any other).
5	2.c	Create an Angular application which will do following actions: Register User, Login User, Show User Data on Profile Component
6	3.a	Create a Node.JS Application which serves a static website.
7	3.b	Create four API using Node.JS, ExpressJS and MongoDB for CURD Operations on assignment 2.C.
8	4.a	Create a simple Mobile Website using jQuery Mobile.
9	4.b	Deploy/Host Your web application on AWS VPC or AWS Elastic Beanstalk. Mini Project Develop a web application using full stack development technologies in any of the following domains: 1. Social Media 2. Ecommerce 3. Restaurant 4. Medical 5. Finance 6. Education
<i>Content Beyond Syllabus</i>		
10	5	Design an application illustrating use of JSON
<i>Virtual Lab</i>		
NA		

Course Code: 314458

Course Name: LABORATORY PRACTICE – II (WAD) (2019 Course)

ASSIGNMENT NO.1

Problem Statement:

a. Create a responsive web page which shows the ecommerce/college/exam admin dashboard with sidebar and statistics in cards using HTML, CSS and Bootstrap Removing document

Tools / Environment:

HTML,CSS, Bootstarp

Related Theory:

- I. HTML
 - HTML stands for Hyper Text Markup Language
 - HTML is the standard markup language for creating Web pages
 - HTML describes the structure of a Web page
 - HTML consists of a series of elements
 - HTML elements tell the browser how to display the content
 - HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc

A simple HTML Document

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

- The <!DOCTYPE html> declaration defines that this document is an HTML5 document
- The <html> element is the root element of an HTML page
- The <head> element contains meta information about the HTML page
- The <title> element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)
- The <body> element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.
- The <h1> element defines a large heading

- The <p> element defines a paragraph
- II. CSS

- CSS stands for Cascading Style Sheets
- CSS describes how HTML elements are to be displayed on screen, paper, or in other media
- CSS saves a lot of work. It can control the layout of multiple web pages all at once
- External style sheets are stored in CSS files
- CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.

Simple CSS example

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background-color: lightblue;
}

h1 {
  color: white;
  text-align: center;
}

p {
  font-family: verdana;
  font-size: 20px;
}
</style>
</head>
<body>

<h1>My First CSS Example</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

III. Bootstrap

Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites. It solves many problems which we had once, one of which is the cross-browser compatibility issue. Nowadays, the websites are perfect for all the browsers (IE, Firefox, and Chrome) and for all sizes of screens (Desktop, Tablets, Phablets, and Phones).

Why Bootstrap?

- Faster and Easier Web Development.

TE IT – WAD (2019 Course) Lab Manual

- It creates Platform-independent web pages.
- It creates Responsive Web-pages.
- It is designed to be responsive to mobile devices too.
- It is Free! Available on www.getbootstrap.com
- Simple Example

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Bootstrap Example</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min
.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
</head>
<body>

  <div class="container">
    <h1>My First Bootstrap Page</h1>
    <p>This is some text.</p>
  </div>

</body>
</html>
```

Bootstrap CDN

```
<!-- Latest compiled and minified CSS -->
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css
">

<!-- jQuery library -->
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>

<!-- Latest compiled JavaScript -->
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>
```

b. Write a JavaScript Program to get the user registration data and push to array/local storage with AJAX POST method and data list in new page.

AJAX stands for Asynchronous JavaScript and XML. AJAX is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS, and JavaScript.

Ajax uses XHTML for content, CSS for presentation, along with Document Object Model and JavaScript for dynamic content display.

Conventional web applications transmit information to and from the server using synchronous requests. It means you fill out a form, hit submit, and get directed to a new page with new information from the server.

With AJAX, when you hit submit, JavaScript will make a request to the server, interpret the

results, and update the current screen. In the purest sense, the user would never know that anything was even transmitted to the server. XML is commonly used as the format for receiving server data, although any format, including plain text, can be used. AJAX is a web browser technology independent of web server software.

A user can continue to use the application while the client program requests information from the server in the background. Intuitive and natural user interaction. Clicking is not required, mouse movement is a sufficient event trigger. Data-driven as opposed to page-driven.

AJAX is based on the following open standards –

- Browser-based presentation using HTML and Cascading Style Sheets (CSS).
- Data is stored in XML format and fetched from the server.
- Behind-the-scenes data fetches using XMLHttpRequest objects in the browser.
- JavaScript to make everything happen

AJAX cannot work independently. It is used in combination with other technologies to create interactive webpages.

- JavaScript
 - Loosely typed scripting language.
 - JavaScript function is called when an event occurs in a page.
 - Glue for the whole AJAX operation.
- DOM
 - API for accessing and manipulating structured documents.
 - Represents the structure of XML and HTML documents.
- CSS
 - Allows for a clear separation of the presentation style from the content and may be changed programmatically by JavaScript
- XMLHttpRequest
 - JavaScript object that performs asynchronous interaction with the server.

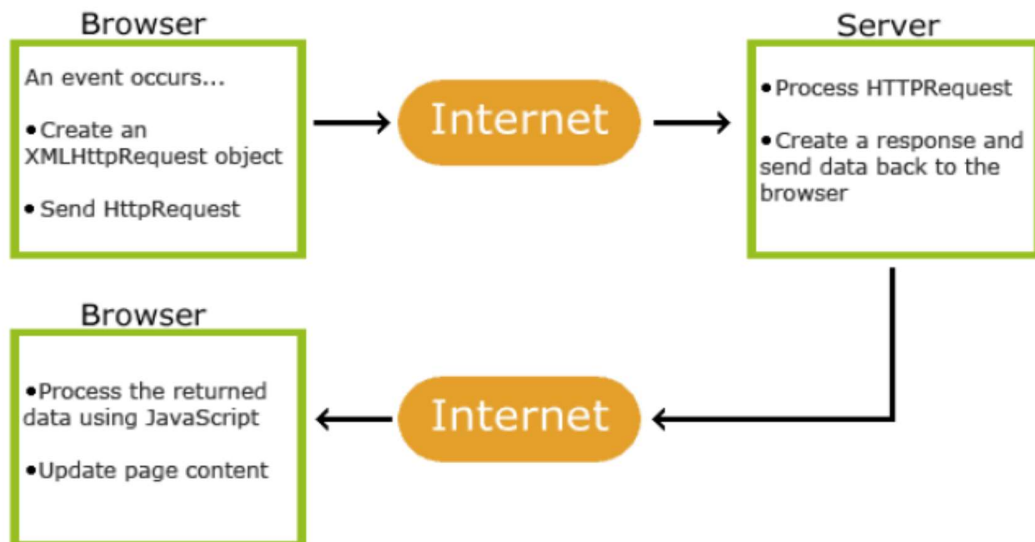


Figure 1. How AJAX works

AJAX – Events : onreadystatechange Event Properties

Property	Description
onReadyStateChange	It is called whenever readystate attribute changes. It must not be used with synchronous requests.
readyState	represents the state of the request. It ranges from 0 to 4. <ul style="list-style-type: none"> • 0: request not initialized (open() is not called.) • 1: server connection established (open is called but send() is not called.) • 2: request received (send() is called, and headers and status are available.) • 3: processing request (Downloading data; responseText holds the data.) • 4: request finished and response is ready (The operation is completed fully.)
Status	200: "OK" 403: "Forbidden" 404: "Page not found"

XMLHttpRequest object properties

Property	Description
readyState	An integer from 0..4. (0 means the call is uninitialized, 4 means that the call is complete.)
onreadystatechange	Determines the function called when the objects readyState changes.
responseText	Data returned from the server as a text string (read-only).
responseXML	Data returned from the server as an XML document object (read-only).
status	HTTP status code returned by the server
statusText	HTTP status phrase returned by the server

XMLHttpRequest object methods

Method	Description
open('method', 'URL', asyn)	Specifies the HTTP method to be used (GET or POST as a string, the target URL, and whether or not the request should be handled asynchronously (asyn should be true or false, if omitted, true is assumed).
send(content)	Sends the data for a POST request and starts the request, if GET is used you should call send(null).
setRequestHeader('x','y')	Sets a parameter and value pair x=y and assigns it to the header to be sent with the request.
getAllResponseHeaders()	Returns all headers as a string.
getResponseHeader(x)	Returns header x as a string.
abort()	Stops the current operation.

Conclusion:

In this assignment, we learned how to create responsive web page using HTML and CSS.

ASSIGNMENT NO. 2

Problem Statement:

- a. Create version control account on GitHub and using Git commands to create repository and push your code to GitHub.**
- b. Create Docker Container Environment (NVIDIA Docker or any other).**
- c. Create an Angular application which will do following actions: Register User, Login User, Show User Data on Profile Component**

Tools / Environment:

GIT, Docker, Angular

Related Theory:

Part a.

1. What is Git?

Git is a popular version control system. It was created by Linus Torvalds in 2005, and has been maintained by Junio Hamano since then.

It is used for:

Tracking code changes

- Tracking who made changes
- Coding collaboration

2. What does Git do?

- Manage projects with Repositories
- Clone a project to work on a local copy
- Control and track changes with Staging and Committing
- Branch and Merge to allow for work on different parts and versions of a project
- Pull the latest version of the project to a local copy
- Push local updates to the main project

3. Working with Git

- Initialize Git on a folder, making it a Repository
- Git now creates a hidden folder to keep track of changes in that folder
- When a file is changed, added or deleted, it is considered modified
- You select the modified files you want to Stage
- The Staged files are Committed, which prompts Git to store a permanent snapshot of the files
- Git allows you to see the full history of every commit.
- You can revert back to any previous commit.
- Git does not store a separate copy of every file in every commit, but keeps track of changes made in each commit!

TE IT – WAD (2019 Course) Lab Manual

4. Why Git?

- Over 70% of developers use Git!
- Developers can work together from anywhere in the world.
- Developers can see the full history of the project.
- Developers can revert to earlier versions of a project.

5. What is GitHub

- Git is not the same as GitHub.
- GitHub makes tools that use Git.
- GitHub is the largest host of source code in the world, and has been owned by Microsoft since 2018.

6. Steps to Push and PULL version control repository to GitHub

Step No	Command	Description
1	Git Installation	Download Git from the website: https://www.git-scm.com/
2	Command line >git -version	If Git is installed, it should show something like git version X.Y
3	git config --global user.name "w3schools-test" git config --global user.email "test@w3schools.com"	Configure Git Change the user name and e-mail address to your own
4	mkdir myproject cd myproject	Creating Git Folder
5	git init	Initialize Git Initialized empty Git repository in /Users/user/myproject/.git/
6	git status	To check the status
7	git add index.html	Add file to staging environment
8	git add --all	add all files in the current directory to the Staging Environment:

TE IT – WAD (2019 Course) Lab Manual

9	git commit -m "First release of Hello World!"	The committ command performs a commit, and the -m "message" adds a message.
10	git commit -a -m "Updated index.html with a new line"	Skips staging environment
11	git log	To view the history of commits for a repository, you can use the log command
12	git <i>command</i> -help	See all the available options for the specific command
13	git help --all	See all possible commands
14	git commit -help	See help for specific command
15	git branch hello-world-images	a branch is a new/separate version of the main repository. This command creates a new branch hello-world-images
16	git checkout hello-world-images	checkout is the command used to check out/ move to a branch
17	git checkout master	Used to switch between branches
18	https://github.com/	Create a new account on github
19		Create a Repository on GitHub
20	git remote add origin https://github.com/w3schools-test/hello-world.git	Push Local Repository to GitHub
21	git push --set-upstream origin master	push master branch to the origin url,
22		go back into GitHub and see that the repository has been updated:
23	git fetch origin	fetch gets all the change history of a tracked branch/repo
24	git merge origin/master	merge combines the current branch, with a specified branch.
25	git pull origin	pull is a combination of fetch and merge

TE IT – WAD (2019 Course) Lab Manual

		It is used to pull all changes from a remote repository into the branch you are working on.
--	--	---

Part b:

Create Docker Container Environment (NVIDIA Docker or any other).

Docker Overview:

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production.

The Docker Platform:

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production.

Docker provides tooling and a platform to manage the lifecycle of your containers:

- Develop your application and its supporting components using containers.
- The container becomes the unit for distributing and testing your application.
- When you're ready, deploy your application into your production environment, as a container or an orchestrated service. This works the same whether your production environment is a local data center, a cloud provider, or a hybrid of the two.

Uses of Docker:

Fast, consistent delivery of your applications. Docker streamlines the development lifecycle by allowing developers to work in standardized environments using local containers which provide your applications and services. Containers are great for continuous integration and continuous delivery (CI/CD) workflows.

Responsive deployment and scaling

Docker's container-based platform allows for highly portable workloads. Docker containers can run on a developer's local laptop, on physical or virtual machines in a data center, on cloud providers, or in a mixture of environments.

Docker's portability and lightweight nature also make it easy to dynamically manage workloads, scaling up or tearing down applications and services as business needs dictate, in near real time.

Running more workloads on the same hardware

Docker is lightweight and fast. It provides a viable, cost-effective alternative to hypervisor-based virtual machines, so you can use more of your compute capacity to achieve your business goals. Docker is perfect for high density environments and for small and medium deployments where you need to do more with fewer resources.

Docker architecture

Refer : <https://docs.docker.com/get-started/overview/>

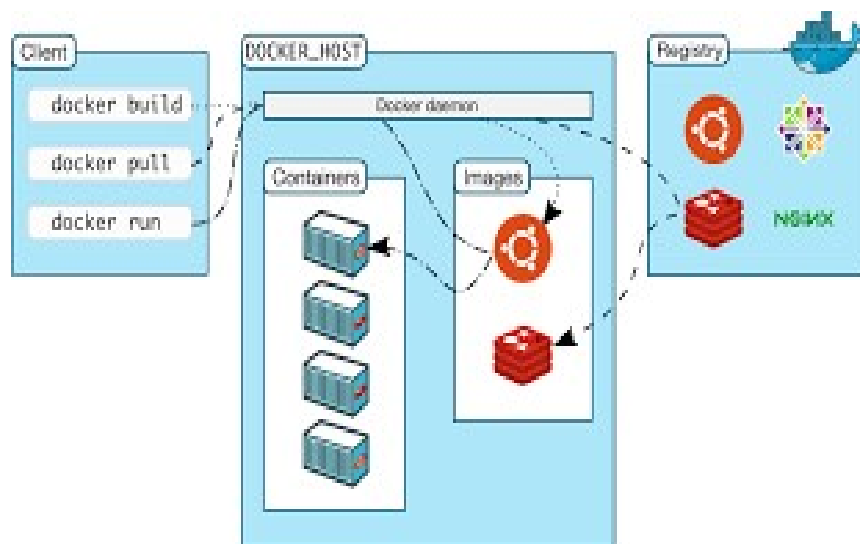


Figure 2. Docker Architecture

The Docker daemon

The Docker daemon (dockerd) listens for Docker API requests and manages Docker objects such as images, containers, networks, and volumes. A daemon can also communicate with other daemons to manage Docker services.

The Docker client

The Docker client (docker) is the primary way that many Docker users interact with Docker. When you use commands such as docker run, the client sends these commands to dockerd, which carries them out. The docker command uses the Docker API. The Docker client can communicate with more than one daemon.

Docker Desktop

Docker Desktop is an easy-to-install application for your Mac or Windows environment that enables you to build and share containerized applications and microservices. Docker Desktop includes the Docker daemon (dockerd), the Docker client (docker), Docker Compose, Docker Content Trust, Kubernetes, and Credential Helper.

Docker registries

A Docker *registry* stores Docker images. Docker Hub is a public registry that anyone can use, and Docker is configured to look for images on Docker Hub by default. You can even run your own private registry. When you use the `docker pull` or `docker run` commands, the required images are pulled from your configured registry. When you use the `docker push` command, your image is pushed to your configured registry.

Docker objects

When you use Docker, you are creating and using images, containers, networks, volumes, plugins, and other objects. This section is a brief overview of some of those objects.

Images

An *image* is a read-only template with instructions for creating a Docker container. Often, an image is *based on* another image, with some additional customization. For example, you may build an image which is based on the ubuntu image, but installs the Apache web server and your application, as well as the configuration details needed to make your application run.

You might create your own images or you might only use those created by others and published in a registry. To build your own image, you create a *Dockerfile* with a simple syntax for defining the steps needed to create the image and run it. Each instruction in a Dockerfile creates a layer in the image. When you change the Dockerfile and rebuild the image, only those layers which have changed are rebuilt. This is part of what makes images so lightweight, small, and fast, when compared to other virtualization technologies.

Containers

A container is a runnable instance of an image. You can create, start, stop, move, or delete a container using the Docker API or CLI. You can connect a container to one or more networks, attach storage to it, or even create a new image based on its current state.

By default, a container is relatively well isolated from other containers and its host machine. You can control how isolated a container's network, storage, or other underlying subsystems are from other containers or from the host machine.

A container is defined by its image as well as any configuration options you provide to it when you create or start it. When a container is removed, any changes to its state that are not stored in persistent storage disappear.

System Requirements:

TE IT – WAD (2019 Course) Lab Manual

- Windows 11 64-bit: Home or Pro version 21H2 or higher, or Enterprise or Education version 21H2 or higher.
- Windows 10 64-bit: Home or Pro 2004 (build 19041) or higher, or Enterprise or Education 1909 (build 18363) or higher.
- Enable the WSL 2 feature on Windows. For detailed instructions, refer to the Microsoft documentation.
- The following hardware prerequisites are required to successfully run WSL 2 on Windows 10 or Windows 11:
 - 64-bit processor with Second Level Address Translation (SLAT)
 - 4GB system RAM

Part c:

Useful link - <https://www.tutorialsteacher.com/angular/install-angular>

1. Angular requires a current, active LTS(long term support) or maintenance LTS version of Node.js and NPM.

install node.js <https://nodejs.org/>

It will automatically install NPM - node package manager

2. Install Angular CLI

```
npm install -g @angular/cli@latest
```

To Create Angular 2 Application Angular CLI is required

3. To create new project

through CLI go to folder of the new project

Give command as -

```
ng new project-name
```

press ENTER

The project will be created as directory structure below –

```
.angular
.git
.vscode
node_modules
src
.browserslistrc
.editorconfig
.gitignore
angular.json
karma.conf.js
package.json
package-lock.json
README.md
tsconfig.app.json
tsconfig.json
tsconfig.spec.json
```

Open folder src/app

Modify app.module.ts for form application

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
```

```
import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { FormsModule } from '@angular/forms'
```

```
@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    FormsModule,
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Open app.component.html

Write html code for form (representative code is mentioned here, modify for multiple inputs)

```
<h1>Simple Form</h1>
<form #simpleForm = "ngForm" (ngSubmit) = "getValues(simpleForm.value)">
  <input type="text" ngModel name = "user" placeholder = "Enter Name">
  <br> <br>
  <input type="text" ngModel name = "age" placeholder = "Enter age">
  <br> <br>
  <input type="text" ngModel name = "city" placeholder = "Enter city">
  <br> <br>
  <button>Get user value</button>
</form>
```

[Make changes in app.component.ts](#)

```
import { Component } from '@angular/core';
```

```
@Component({
  selector: 'app-root',
```

```
templateUrl: './app.component.html',  
styleUrls: ['./app.component.css']  
})  
export class AppComponent {  
  title = 'AngProj1';  
  getValues(val:any)  
  {  
    console.log(val);  
  }  
}
```

Here getValue() function which is called in form file is defined.
You can check inputted values through form in console.

build application

1. Use Angular CLI command `ng serve -o` to build an application. The `-o` indicates to open it automatically in the default browser.
2. Use NPM command `'npm start'` to build an application <http://localhost:4200> to see the application home page.
3. Open the terminal in VS Code from menu Terminal -> New Terminal, and type `ng serve -o` command and press enter, You can send the form contents from console to other page. On the basis of above implementation, you can design login user, show user data.

Conclusion:

In this assignment we learned about GIT, Docker and Angular

ASSIGNMENT NO. 3

Problem Statement:

- a. Create a Node.JS Application which serves a static website.
- b. Create four API using Node.JS, ExpressJS and MongoDB for CRUD Operations on assignment 2.C.

Tools / Environment:

Node.js, ExpressJS and MongoDB

Related Theory:

Create a Node.JS Application which serves a static website.

Installation : Node.js (site – Node.js), Express.js(installed through cmd)

Node.js overview

In basic terms, Node is an open source cross-platform library for server-side programming that permits clients to develop web applications rapidly. With Node, we can execute JavaScript applications or network applications. Its basic modules are engraved in JavaScript.

It is generally utilized for server applications in real-time. Node.js permits JavaScript to execute locally on a machine or a server.

Node.js gives numerous systems to utilize. One of such structures is Express.js. It is more valuable and mainstream than the different structures of Node.js.

Features of Node.js

- **Versatility:** Node is incredibly adaptable as the server reacts in a non-blocking way.
- **Zero Buffering:** Applications yield the measurements in enormous pieces. This gives the advantage of 'No buffering' to developers.
- **Network:** Node.js upholds an open-source community. This is the main explanation that numerous glorious modules have been added to Node.js applications over time.
- **Occasion driven Input and output:** APIs of Node.js are non-blocking, meaning that the server won't wait for the arrival of information from an API. Rather, it will move to another API.

Advantages of Node.js

- **Easy to learn:** Node.js is quite simple for developers to utilize and learn. Learning Node.js is less difficult than React.
- **Better Performance:** Node.js takes the code of JavaScript via Google's V8 JavaScript engine. The main advantage of this process is that it complies with the JavaScript code directly into the machine code
- **Freedom:** Node.js offers a lot of freedom when it comes to development. There are generally less constraints with Node.
- **Extended support for tools:** Another advantage of Node.js is that developers have more community support.
- **Extensible:** Node.js is known to be quite extensible. You can utilize JSON to give the degree to trade of information between the web server and the client.

- **Scalability:** Node.js makes it simple to scale applications in horizontal as well as vertical directions. The applications can be scaled even by the option of extra hubs to the current framework.

Limitations of Node.js

- **Programming interface isn't steady:** The Application Programming Interface (API) of Node can be challenging to work with. It changes regularly and doesn't remain stable.
- **No strong library support system:** JavaScript does not hold a strong library system. This limits the developers to implement even common programming tasks using Node.js.
- **Programming model is not synchronous:** Many developers find this programming model tougher in comparison to linear blocking I/O programming. In asynchronous programming, the codes become clumsier.

Express.js

"Express is a fast, opinionated minimalist web framework for Node.js" - official web site: [Expressjs.com](https://expressjs.com)

Express.js is a web application framework for Node.js. It provides various features that make web application development fast and easy which otherwise takes more time using only Node.js.

Express.js is based on the Node.js middleware module called **connect** which in turn uses **http** module. So, any middleware which is based on connect will also work with Express.js.



Figure 3: Express.js

Advantages of Express.js

1. Makes Node.js web application development fast and easy.
2. Easy to configure and customize.
3. Allows you to define routes of your application based on HTTP methods and URLs.
4. Includes various middleware modules which you can use to perform additional tasks on request and response.
5. Easy to integrate with different template engines like Jade, Vash, EJS etc.
6. Allows you to define an error handling middleware.
7. Easy to serve static files and resources of your application.
8. Allows you to create REST API server.

TE IT – WAD (2019 Course) Lab Manual

9. Easy to connect with databases such as MongoDB, Redis, MySQL

Steps :

1. Install Node.js
2. Setting up express.js
3. Structuring files
4. Creating your express server
5. Servicing your static files
6. Building your web page
7. Running your project

Open Node.js command terminal & run the following in your terminal-

Setting up express.js

1. Create a new directory for your project - `mkdir your-project-name`
2. Change into your new directory - `cd your-project-name`
3. Initialize a new Node project with defaults. This will set a `package.json` file to access your dependencies:
`npm init -y`
4. Create your entry file, `index.js`. This is where you will store your Express server: if you are working on Linux, you can run : `touch index.js`. if you are working on windows, you can edit in VSCode
5. Install Express as a dependency : `npm install express --save`
6. Edit `package.json`. Within your `package.json`, update your start script to include node and your `index.js` file.

Let `express-static-file-tutorial` is your project name

Package.json

```
{
  "name": "express-static-file-tutorial",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "start": "node index.js" // change start value as node index.js
  },
  // This will allow you to use the npm start command in your terminal to launch
  // your express server
  "keywords": [],
  "author": "Paul Halliday",
  "license": "MIT"
}
```

Structuring Your Files

To store your files on the client-side, create a public directory and include an `index.html` file

Creating Your Express Server

Edit index.js file

Index.js

```
const express = require('express');
const app = express();
const PORT = 3000;

app.use(express.static('public')); // represents application is serving static webpage in public directory

app.get('/', (req, res) => {
  res.send('Hello World!');
});

app.listen(PORT, () => console.log(`Server listening on port: ${PORT}`));
```

First of all, import the Express.js module.

In the above example, we imported Express.js module using require() function. The express module returns a function. This function returns an object which can be used to configure Express application (app in the above example).

The app object includes methods for routing HTTP requests, configuring middleware, rendering HTML views and registering a template engine.

The app.listen() function creates the Node.js web server at the specified host and port. It is identical to Node's http.Server.listen() method. Instead of Get(), post(), put() and delete() methods can be used.

Building Your Web Page – client side

Navigate to your index.html file in the public directory. Populate the file with body and image elements:

[label index.html]

```
<html>
  <head>
    <title>Hello World!</title>
  </head>
  <body>
    <h1>Hello, World!</h1>
     //download & store image in public directory
  </body>
</html>
```

(Instead of building Hello world application, building applications like student's Registration form/main page of website is recommended)

Running Your Project

In your terminal, launch your Express project
npm start

TE IT – WAD (2019 Course) Lab Manual

It will display
Server listening on port: 3000

Open your web browser, and navigate to <http://localhost:3000>.

Conclusion:

In this assignment we learned how to build application using NodeJs, ExpressJs and MongoDB

ASSIGNMENT NO. 4

Problem Statement:

- a. Create a simple Mobile Website using jQuery Mobile.
 - b. Deploy/Host Your web application on AWS VPC or AWS Elastic Beanstalk. Mini Project
- Develop a web application using full stack development technologies in any of the following domains:
1. Social Media
 2. ecommerce
 3. Restaurant
 4. Medical
 5. Finance
 6. Education
 7. Any other

Tools / Environment:

jQuery Mobile, AWS Elastic Beanstalk

Related Theory:

Part a:

jQuery Mobile

jQuery Mobile is a user interface framework, built on jQuery Core and used for developing responsive websites or applications that are accessible on mobile, tablet, and desktop devices. It uses features of both jQuery and jQueryUI to provide API features for mobile web applications. This tutorial will teach you the basics of jQuery Mobile framework. We will also discuss some detailed concepts related to jQuery Mobile.

Why Use jQuery Mobile?

- It creates web applications that it will work the same way on the mobile, tablet, and desktop devices.
- It is compatible with other frameworks such as PhoneGap, Whitelight, etc.
- It provides a set of touch-friendly form inputs and UI widgets.

Features of jQuery Mobile

- It is built on jQuery Core and "write less, do more" UI framework.
- It is an open source framework, and cross-platform as well as cross-browser compatible.
- It is written in JavaScript and uses features of both jQuery and jQuery UI for building mobile-friendly sites.

Download jQuery Mobile

When you open the link <https://jquerymobile.com/>, you will see there are two options to download jQuery mobile library.



Figure 4: jQuery Installation

Click the *Stable* button, which leads directly to a ZIP file containing the CSS and JQuery files, for the latest version of jQuery mobile library. Extract the ZIP file contents to a jQuery mobile directory.

This version contains all files including all dependencies, a large collection of demos, and even the library's unit test suite. This version is helpful to getting started.

Part b:

What is Cloud Computing?

Cloud computing is a term referred to storing and accessing data over the internet. It doesn't store any data on the hard disk of your personal computer. In cloud computing, you can access data from a remote server.

What is AWS?

The full form of AWS is Amazon Web Services. It is a platform that offers flexible, reliable, scalable, easy-to-use and, cost-effective cloud computing solutions.

AWS is a comprehensive, easy to use computing platform offered Amazon. The platform is developed with a combination of infrastructure as a service (IaaS), platform as a service (PaaS) and packaged software as a service (SaaS) offerings.

It is a secure cloud services platform, offering compute power, database storage, content delivery and other functionality to help businesses scale and grow.

In simple words AWS allows you to do the following things- Running web and application servers in the cloud to host dynamic websites.

History of AWS

- 2002- AWS services launched
- 2006- Launched its cloud products
- 2012- Holds first customer event
- 2015- Reveals revenues achieved of \$4.6 billion
- 2016- Surpassed \$10 billion revenue target

TE IT – WAD (2019 Course) Lab Manual

- 2016- Release snowball and snowmobile
- 2019- Offers nearly 100 cloud services
- 2021- AWS comprises over 200 products and services

Important AWS Services

Amazon Web Services offers a wide range of different business purpose global cloud-based products. The products include storage, databases, analytics, networking, mobile, development tools, and enterprise applications, with a pay-as-you-go pricing model.

Applications of AWS services

Amazon Web services are widely used for various computing purposes like:

- Web site hosting
- Application hosting/SaaS hosting
- Media Sharing (Image/ Video)
- Mobile and Social Applications
- Content delivery and Media Distribution
- Storage, backup, and disaster recovery
- Development and test environments
- Academic Computing
- Search Engines
- Social Networking

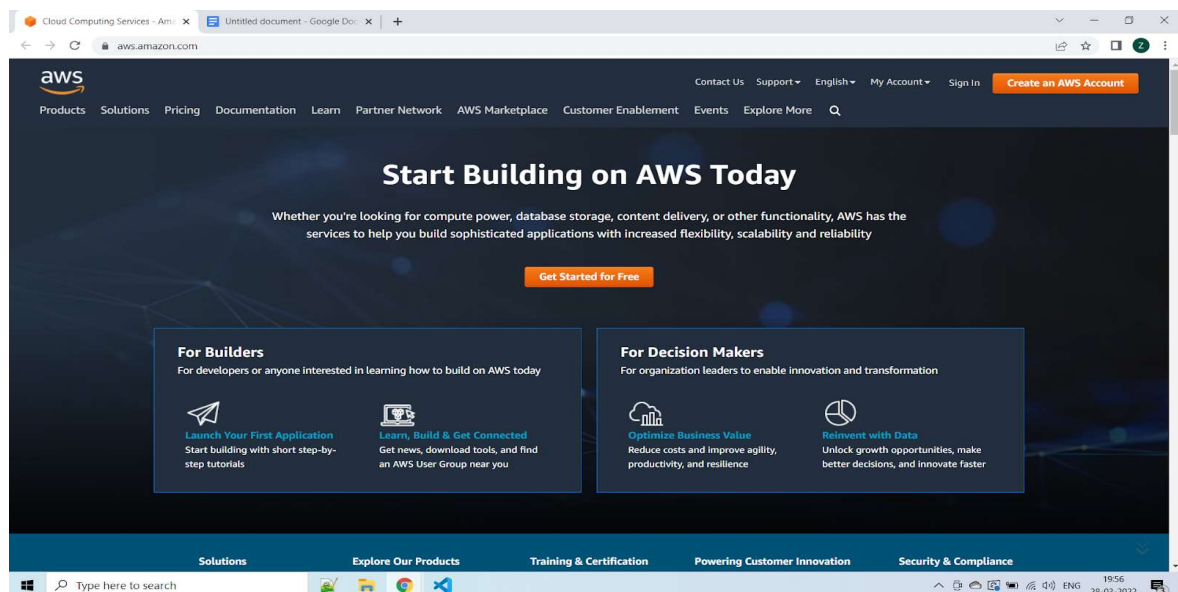
Creating an AWS Account is the first step you need to take in order to learn Amazon Web Services.

Steps to follow are as follows :

Step 1 – Visiting the Signup Page

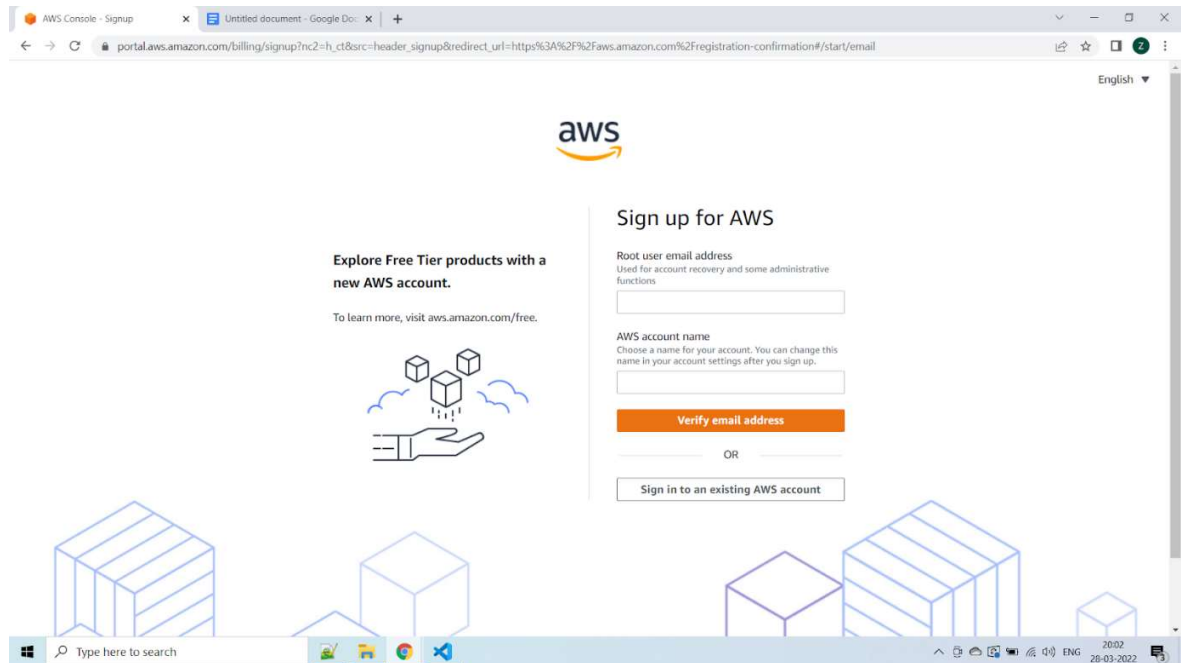
Go to <https://aws.amazon.com>

You should see something like below:



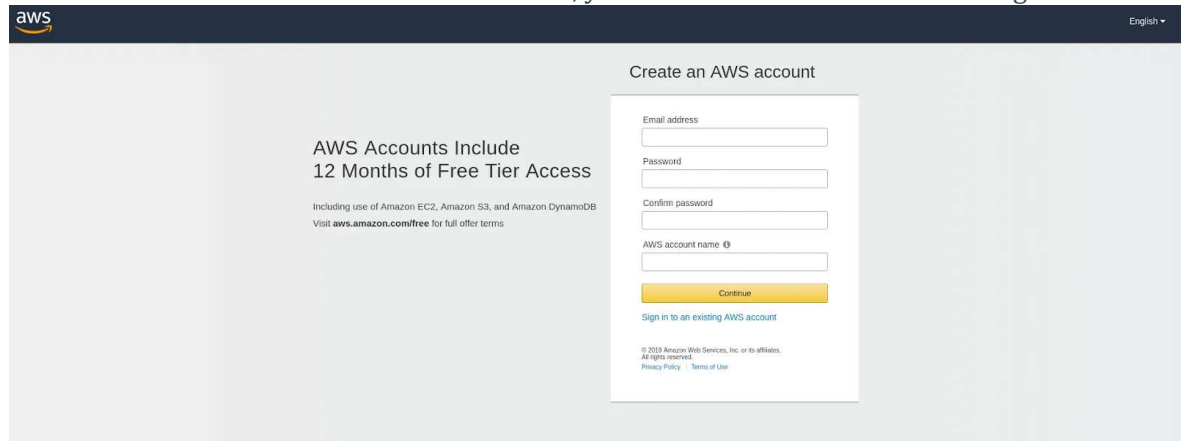
TE IT – WAD (2019 Course) Lab Manual

In order to continue, click the **Complete Sign Up** button in the middle of the screen or on the top right corner of the screen. You will see the below screen.



Step 2 – Entering User Details

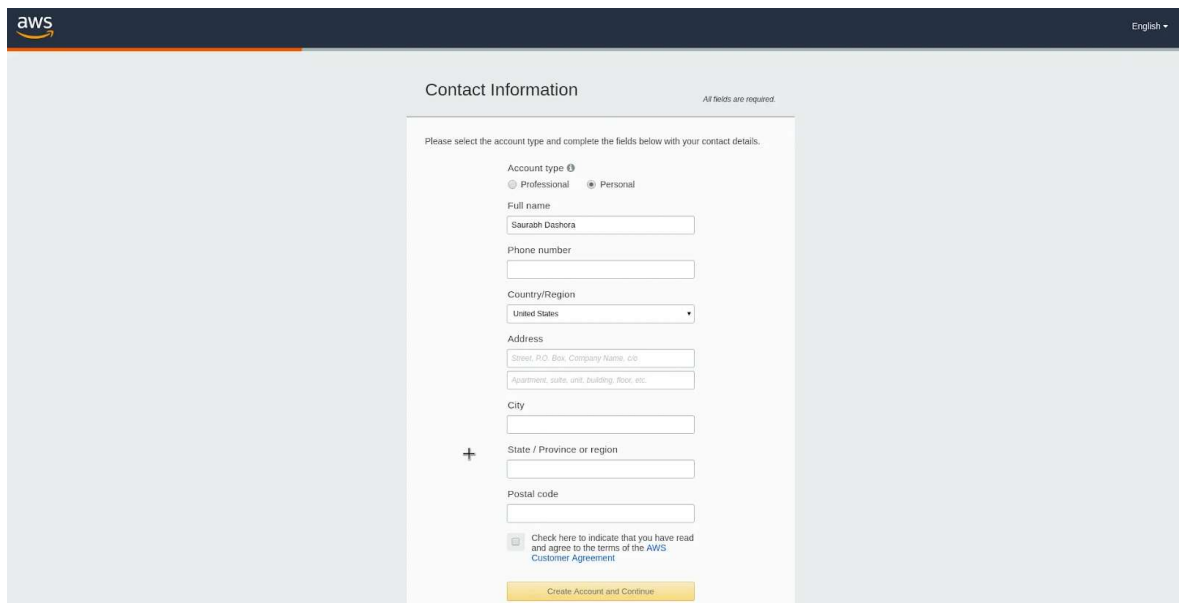
After you have chosen to **Create a new AWS account**, you will see the below screen asking for few details.



You can fill up the details as per your requirements and click **Continue**.

Next you will be asked to fill up your contact details such contact number, country, address and so on. You should fill them up properly because your contact number is important for further steps.

TE IT – WAD (2019 Course) Lab Manual

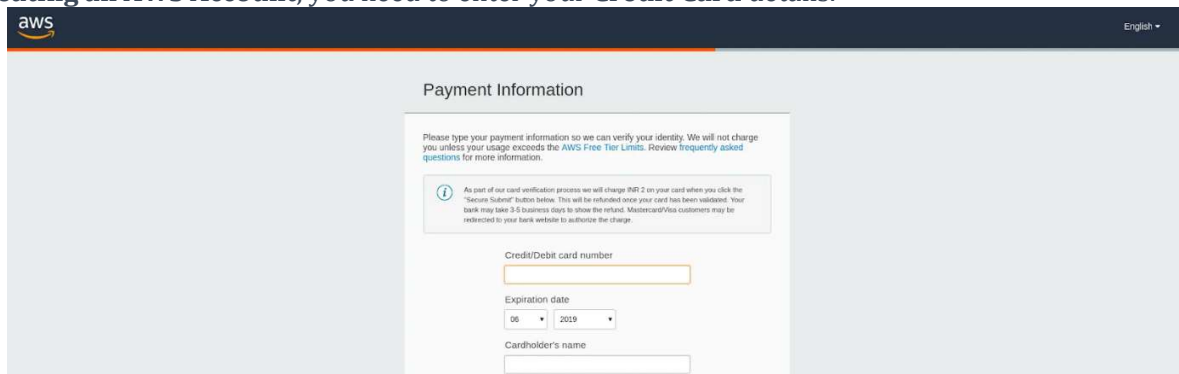


The screenshot shows the 'Contact Information' step of the AWS account creation process. The header includes the AWS logo and a language dropdown set to 'English'. The main heading is 'Contact Information' with a note 'All fields are required.' Below this, a sub-header says 'Please select the account type and complete the fields below with your contact details.' The form includes radio buttons for 'Account type' (Professional and Personal, with Personal selected). Fields for 'Full name' (filled with 'Saurabh Dashora'), 'Phone number', 'Country/Region' (dropdown set to 'United States'), 'Address' (split into 'Street, P.O. Box, Company Name, etc.' and 'Apartment, suite, unit, building, floor, etc.'), 'City', 'State / Province or region' (with a plus icon to the left), and 'Postal code' are present. A checkbox for 'Check here to indicate that you have read and agree to the terms of the AWS Customer Agreement' is checked. A yellow 'Create Account and Continue' button is at the bottom.

After filling up the details, click on the **Create Account and Continue** button at the bottom of the form.

Step 3 – Filling up the Credit Card details

For **Creating an AWS Account**, you need to enter your **Credit Card** details.



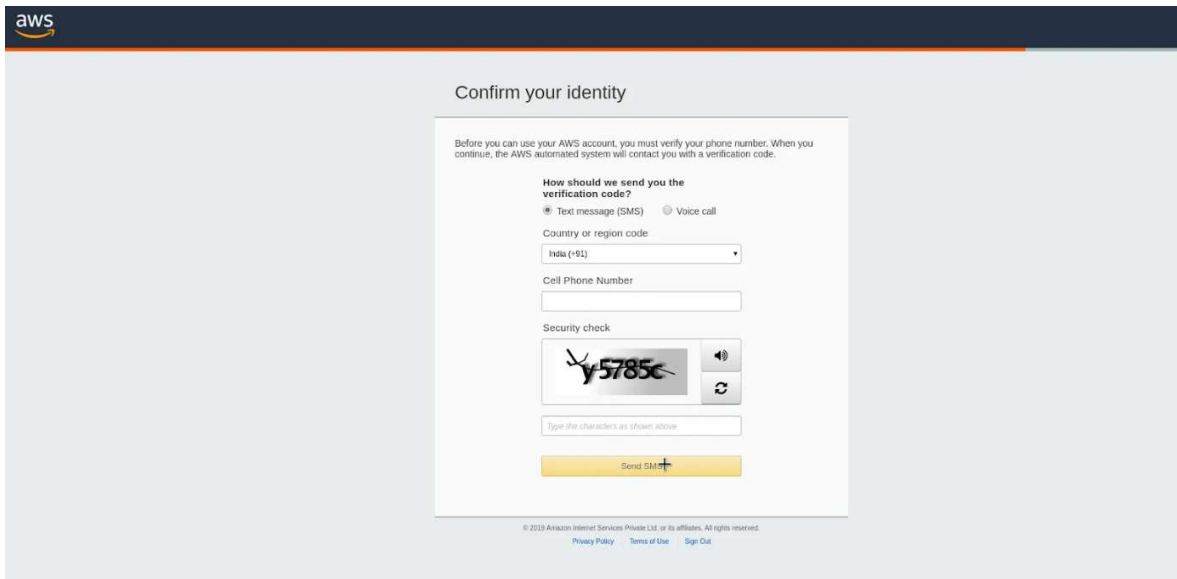
The screenshot shows the 'Payment Information' step of the AWS account creation process. The header includes the AWS logo and a language dropdown set to 'English'. The main heading is 'Payment Information'. A sub-header says 'Please type your payment information so we can verify your identity. We will not charge you unless your usage exceeds the AWS Free Tier Limits. Review frequently asked questions for more information.' Below this is a blue information box with a warning icon and text: 'As part of our card verification process we will charge \$0.01 on your card when you click the "Secure Submit" button below. This will be refunded once your card has been validated. Your bank may take 3-5 business days to show the refund. Mastercard/Visa customers may be redirected to your bank website to authorize the charge.' The form includes fields for 'Credit/Debit card number', 'Expiration date' (with dropdowns for month '06' and year '2019'), and 'Cardholder's name'.

After entering the details, click on **Secure Submit** button. It might take a while to process the request depending on your bank/credit card company servers.

Step 4 – Identity Confirmation

Once the credit card details are confirmed, you will need to complete the **Identity Confirmation** step. You will see the below screen:

TE IT – WAD (2019 Course) Lab Manual



aws

Confirm your identity

Before you can use your AWS account, you must verify your phone number. When you continue, the AWS automated system will contact you with a verification code.

How should we send you the verification code?

☒ Text message (SMS) ☐ Voice call

Country or region code

India (+91)

Cell Phone Number

Security check

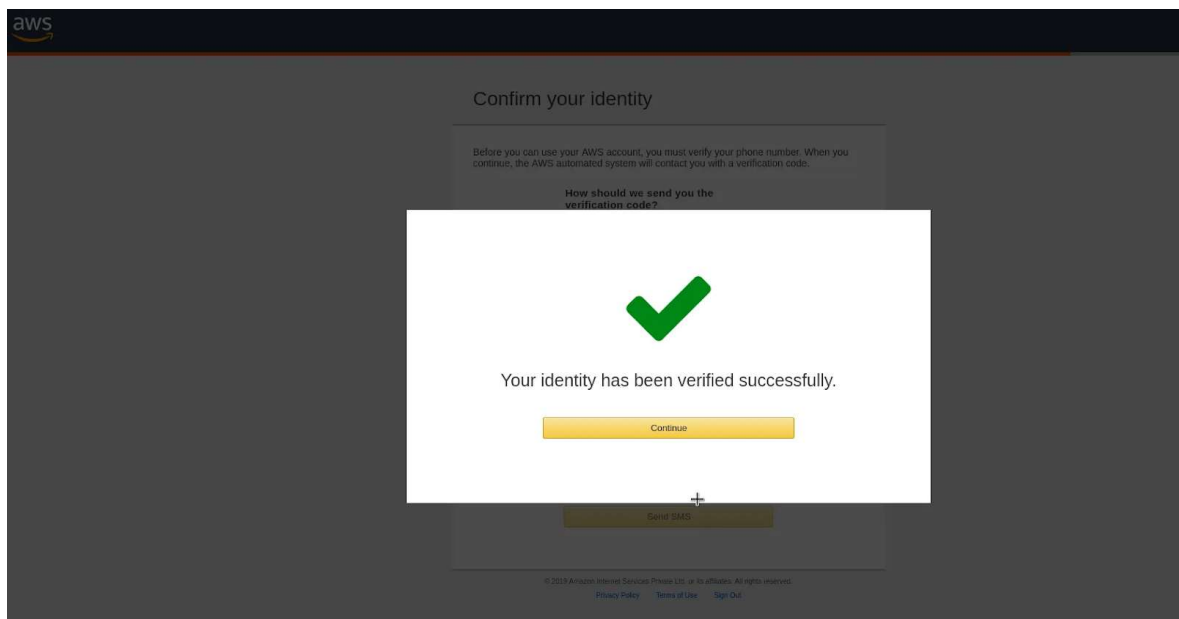
y5785c

Type the characters as shown above

Send SMS

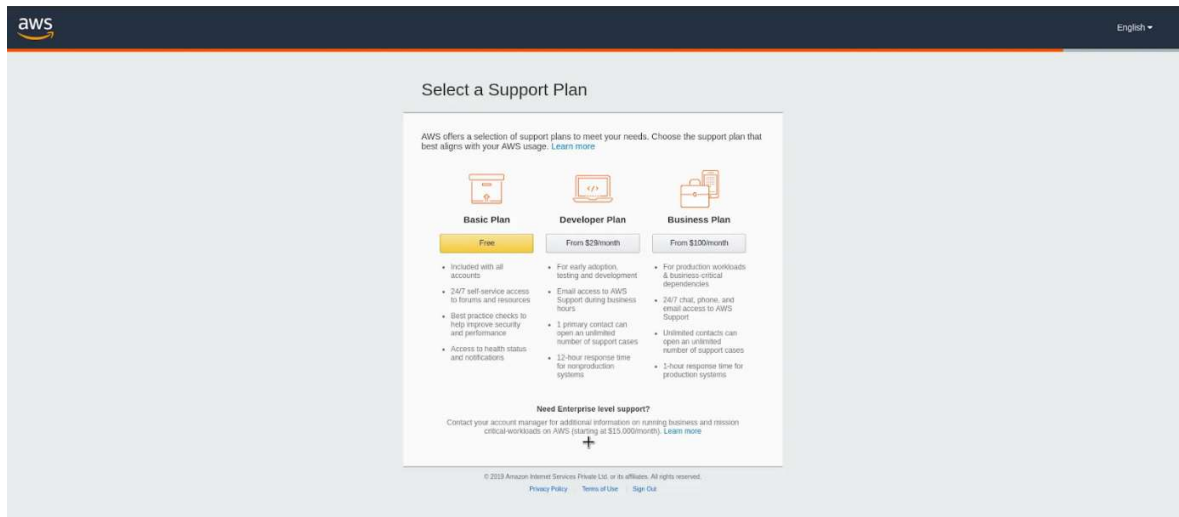
© 2019 Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.
[Privacy Policy](#) [Terms of Use](#) [Sign Out](#)

Once you have verified successfully, you should see a screen like below:

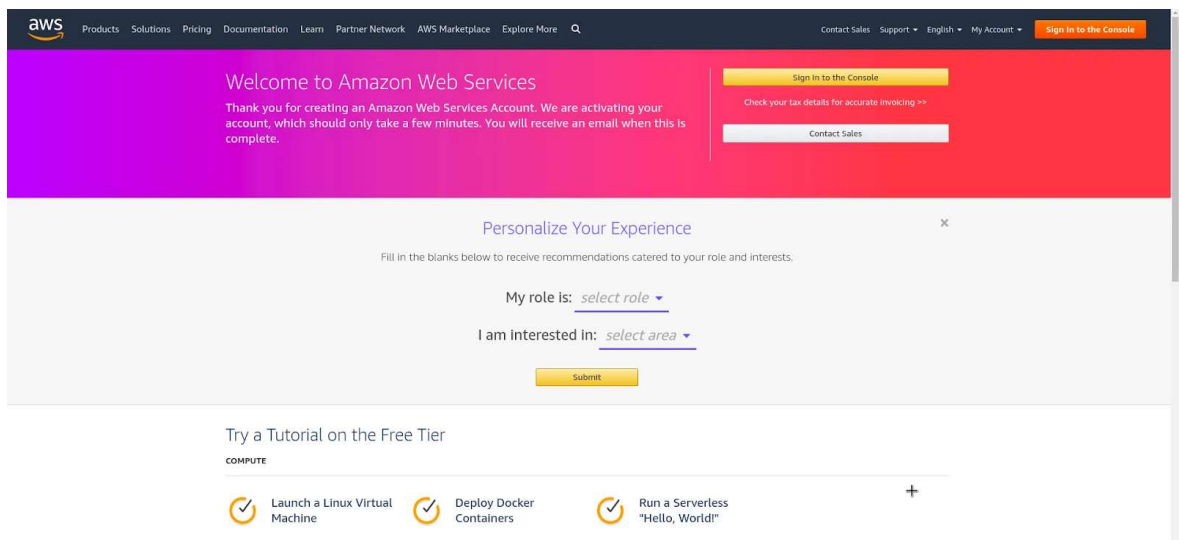


TE IT – WAD (2019 Course) Lab Manual

Step 5 – Selecting a Support Plan

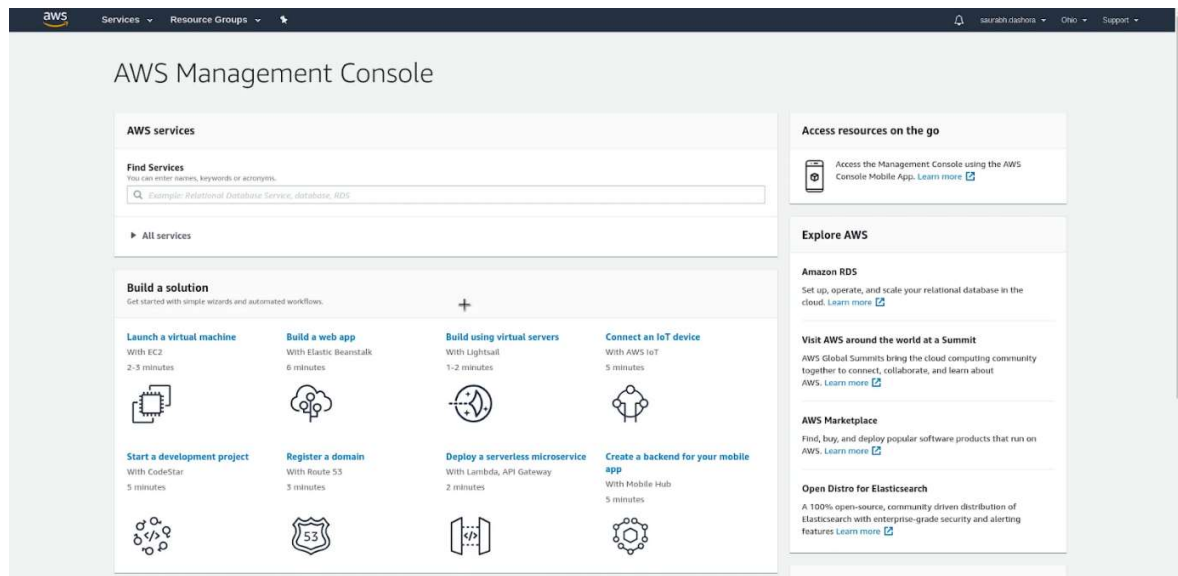


Go for **Basic Plan**. It is Free of cost and great for learning purposes. The other plans are **Developer Plan** and a **Business Plan**. But both of them are paid options. Once you select your plan, you will see the below **Welcome** screen. From here on, you can Sign in to your **AWS Console**.



Finally, after logging in, you should be able to see the **AWS Management Console** as below:

TE IT – WAD (2019 Course) Lab Manual



If you have reached this far, you have successfully finished **Creating an AWS Account**.

Deployment Steps:

- Step 1: Launch a Windows Server Amazon EC2 instance.
- Step 2: Configure your source content to deploy to the Windows Server Amazon EC2 instance.
- Step 3: Upload your "hello, world!"
- Step 4: Deploy your Hello World application.
- Step 5: Update and redeploy your "hello, world!"
- Step 6: Clean up your "hello, world!"

Conclusion:

In this assignment, we learned jQuery mobile and deployment of application on AWS