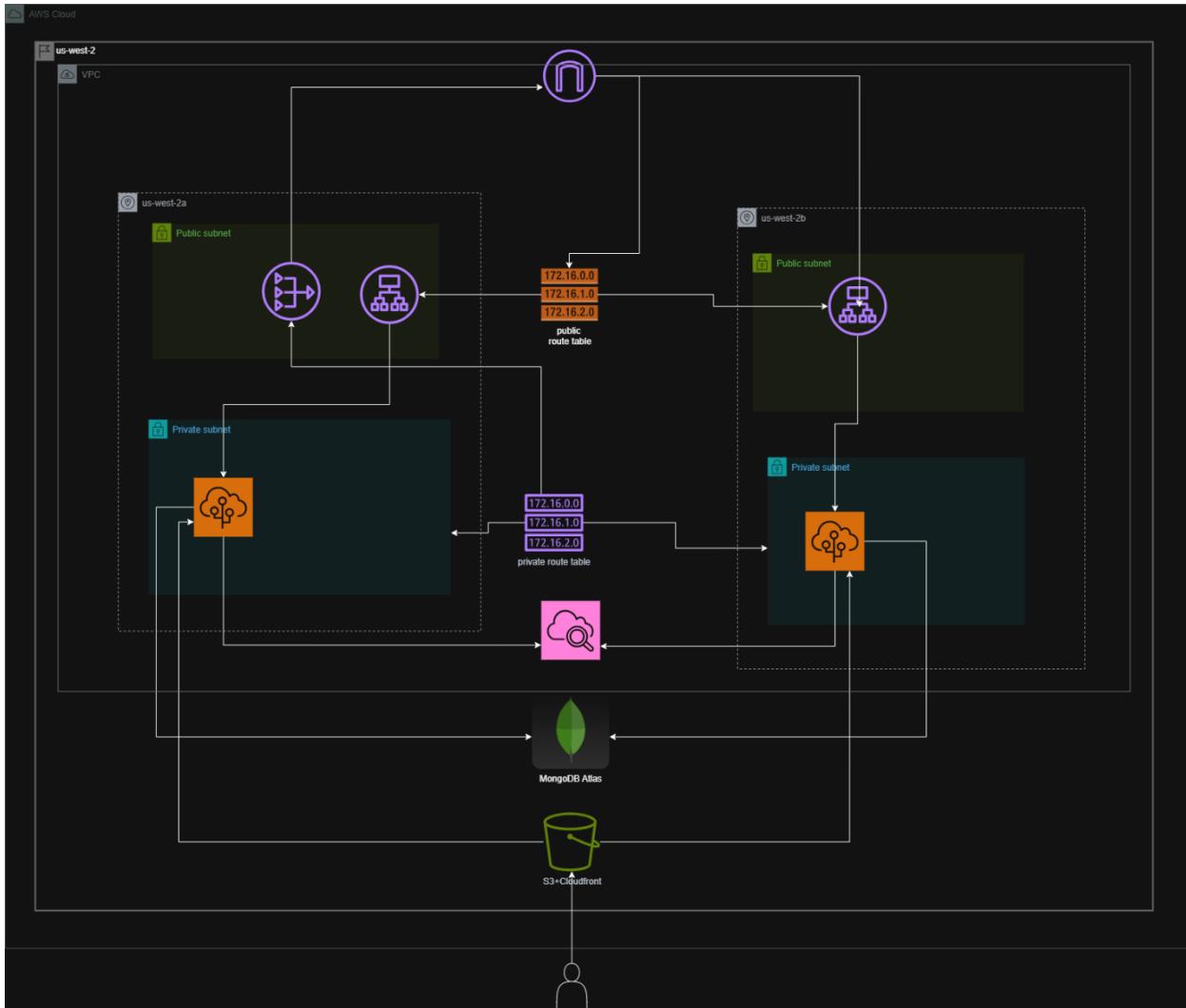

Sardar Noor Ul Hassan
Cloud Intern

Deploying a Production-Ready Node.js Application on AWS Elastic Beanstalk (Three-Tier Architecture)

1. Project Overview

This project demonstrates the end-to-end deployment of a **production-ready Node.js backend application** using **AWS Elastic Beanstalk**, integrated with a managed **MongoDB database (MongoDB Atlas)** and later extended into a **three-tier architecture** by introducing a **static frontend hosted on Amazon S3 and Amazon CloudFront**.

2. Architecture Summary (High Level)



Architecture Type: Three-Tier Architecture

Layers:

1. Presentation Layer (Frontend)

- Static frontend hosted on Amazon S3
- Delivered globally via Amazon CloudFront (HTTPS)

2. Application Layer (Backend)

- Node.js REST API
- Deployed on AWS Elastic Beanstalk (Node.js platform)

- Exposed via Elastic Beanstalk load-balanced environment

3. Data Layer (Database)

- MongoDB Atlas (managed MongoDB)
- Secure connection via MongoDB URI stored in environment variables

3. Prepare Node.js Application Code with Necessary Dependencies

The backend application was prepared locally with:

- Node.js runtime
- Express.js framework
- MongoDB (Mongoose) integration
- JWT-based authentication
- Modular folder structure:
 - controllers/
 - routes/
 - models/
 - middleware/
 - config/
 - utils/

All required dependencies were defined in:

- package.json
- package-lock.json

Name	Date modified	Type	Size
config	6/27/2025 6:23 PM	File folder	
controllers	7/16/2025 2:55 AM	File folder	
middleware	6/27/2025 6:23 PM	File folder	
models	7/16/2025 2:55 AM	File folder	
node_modules	12/30/2025 6:46 PM	File folder	
routes	6/29/2025 10:44 PM	File folder	
scripts	6/27/2025 6:23 PM	File folder	
utils	6/27/2025 6:23 PM	File folder	
.env	12/29/2025 6:23 PM	ENV File	1 KB
.gitignore	6/27/2025 6:23 PM	Text Document	1 KB
muawinbackend.zip	12/30/2025 4:39 AM	WinRAR ZIP archive	111 KB
package.json	7/16/2025 2:55 AM	JSON Source File	1 KB
package-lock 2.json	6/27/2025 6:23 PM	JSON Source File	72 KB
package-lock.json	12/30/2025 6:46 PM	JSON Source File	226 KB
README.md	6/27/2025 6:23 PM	Markdown Source...	4 KB
server 2.js	6/27/2025 6:23 PM	JavaScript File	44 KB
server.js	7/16/2025 2:55 AM	JavaScript File	3 KB

4. Create an Elastic Beanstalk Application in AWS Management Console

An Elastic Beanstalk application was created from the AWS Management Console.

Key actions:

- Navigated to Elastic Beanstalk service
- Created a new application
- Assigned a meaningful application name
- Selected **Web server environment**

The screenshot shows the AWS Elastic Beanstalk console interface. The top navigation bar includes the AWS logo, a search bar, and account information: 'cloudelligent-sandbox (5046-4907-6991)', 'AdministratorAccess/sardar.hassan'. Below the navigation bar, the left sidebar shows 'Elastic Beanstalk' with sections for 'Applications', 'Environments', and 'Change history'. Under 'Application: muawin-eb', there are links for 'Application versions' and 'Saved configurations'. The main content area displays the 'Application muawin-eb environments (1) Info' section. It lists one environment: 'Muawin-eb-env-1' with status 'Ok', created on 'December...', running version 'v1.2' on 'Node.js 2...', and a 'Supported' status. A 'Create new environment' button is also present. The bottom of the page includes standard AWS footer links: CloudShell, Feedback, Privacy, Terms, and Cookie preferences.

The second part of the screenshot shows the 'Muawin-eb-env-1' environment details page. The left sidebar is identical. The main content area has tabs for 'Events', 'Health', 'Logs', 'Monitoring', 'Alarms', 'Managed updates', and 'Tags'. The 'Events' tab is selected, showing 42 events. The right side of the page is divided into 'Environment overview' and 'Platform' sections. In the 'Environment overview', it shows 'Health' (Ok), 'Domain' (Muawin-eb-env-1.eba-uyyghj8.us-west-1.elasticbeanstalk.com), and 'Application name' (muawin-eb). In the 'Platform' section, it shows 'Platform' (Node.js 24 running on 64bit Amazon Linux 2023/6.7.1), 'Running version' (v1.2), and 'Platform state' (Supported). The bottom of the page includes standard AWS footer links: Events (42), Filter events by text, property or value, Time, Type, Details, Privacy, Terms, and Cookie preferences.

5. Configure Elastic Beanstalk Environment with Platform Node.js

The Elastic Beanstalk environment was configured with:

- Platform:** Node.js
- Environment type:** Load balanced
- Proxy server:** Nginx

- **VPC:** noor-vpc
- **Subnets:** Private subnets for EC2 instances

Elastic Beanstalk automatically provisioned:

- EC2 instances
- Application Load Balancer
- Auto Scaling Group
- IAM roles
- Security groups

The screenshot shows the AWS Elastic Beanstalk Configuration page for the environment "Muawin-eb-env-1".

VPC: A dropdown menu shows the selected VPC is "vpc-052f3413af93f88fc | (10.0.0.0/16) | noor-vpc". There is a "Create VPC" button.

Public IP address: A checkbox labeled "Enable" is checked.

Instance subnets: A table lists subnets across availability zones:

Availability Zone	Subnet	CIDR	Name
us-west-1b	subnet-07d884bf4f33e50b	10.0.4.0/24	n-private-subnet-2
us-west-1a	subnet-099216f597c25261c	10.0.1.0/24	n-public-subnet-1
us-west-1b	subnet-0c934af216ddfa9e3	10.0.3.0/24	n-public-subnet-2
us-west-1a	subnet-0e555b8e468917ea3	10.0.2.0/24	n-private-subnet-1

Instance summary for i-09727933f36857a54 (Muawin-eb-env-1)

Updated less than a minute ago

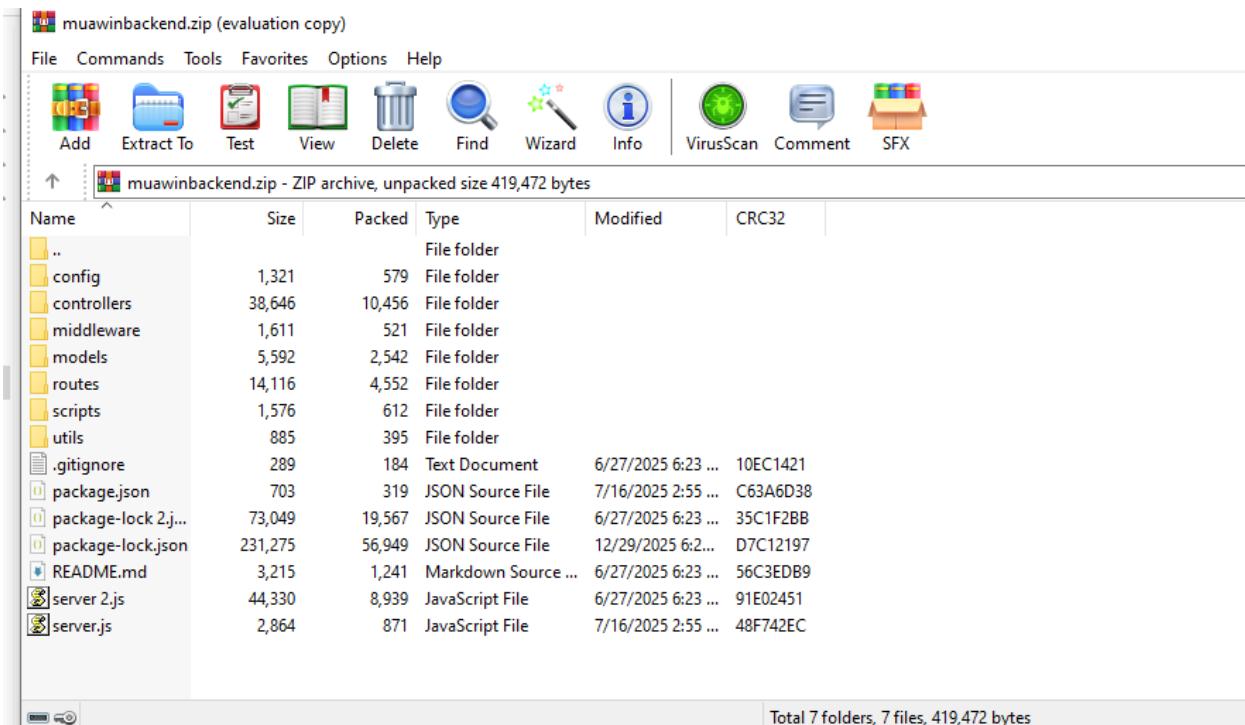
Instance ID i-09727933f36857a54	Public IPv4 address –	Private IPv4 addresses 10.0.2.165
IPv6 address –	Instance state Running	Public DNS –
Hostname type IP name: ip-10-0-2-165.us-west-1.compute.internal	Private IP DNS name (IPv4 only) ip-10-0-2-165.us-west-1.compute.internal	Elastic IP addresses –
Answer private resource DNS name –	Instance type t3.micro	AWS Compute Optimizer finding No recommendations available for this instance.
Auto-assigned IP address –	VPC ID vpc-052f3413af93f88fc (noor-vpc)	Auto Scaling Group name awseb-e-2kqgdv78uy-stack-AWSFAutoScalingGroup-IG3VQ7dkYI-i
IAM Role aws-elasticbeanstalk-ec2-role	Subnet ID subnet-0e555b8e468917ea3 (n-private-subnet-1)	

6. Package Application Code as a ZIP File for Deployment

The backend application was packaged as a ZIP archive:

- Included all application source files
- Excluded node_modules
- Included package.json and package-lock.json
- .env file was **not relied upon** for production secrets

This ZIP file became the deployment artifact for Elastic Beanstalk.



7. Upload and Deploy the Application Package to Elastic Beanstalk

The ZIP package was uploaded to the Elastic Beanstalk environment.

Elastic Beanstalk handled:

- Application extraction
- Dependency installation
- EC2 instance provisioning
- Application startup

After deployment, the environment transitioned to a **Running** state.

8. Monitor Deployment Status and Logs via Elastic Beanstalk Dashboard

Deployment and runtime monitoring were performed using:

- Elastic Beanstalk **Events**
- Elastic Beanstalk **Health dashboard** Application and instance logs

Events **Health** **Logs** **Monitoring** **Alarms** **Managed updates** **Tags**

Overall health Info

Requests / second	2XX responses	3XX responses	4XX responses	5xx responses
0.2	2	-	-	-

P99 latency(ms)	P90 latency(ms)	P75 latency(ms)	P50 latency(ms)	P10 latency(ms)
1	1	1	1	1

Enhanced instance health (1) Info

Instance ID	Status	Running time	Deployment ID	Requests/sec	2x:
i-09727933f3...	Ok	19 hours, 59 m...	3	0.2	2

aws Search [Alt+S] 76 United States (N. California) cloudelectric-sandbox (5046-4907-
AdministratorAccess/sarda)

Elastic Beanstalk > Environments > Muawin-eb-env-1

Elastic Beanstalk

- Applications
- Environments
- Change history

► Application: muawin-eb

▼ Environment: Muawin-eb-env-1

- Go to environment
- Configuration
- Events**
- Health
- Logs
- Monitoring
- Alarms

Health Ok

Environment ID e-2kqgqv78uy

Domain Muawin-eb-env-1.eba-uyyghj8.us-west-1.elasticbeanstalk.com

Application name muawin-eb

Platform Node.js 24 running on 64bit Amazon Linux 2023/6.7.1

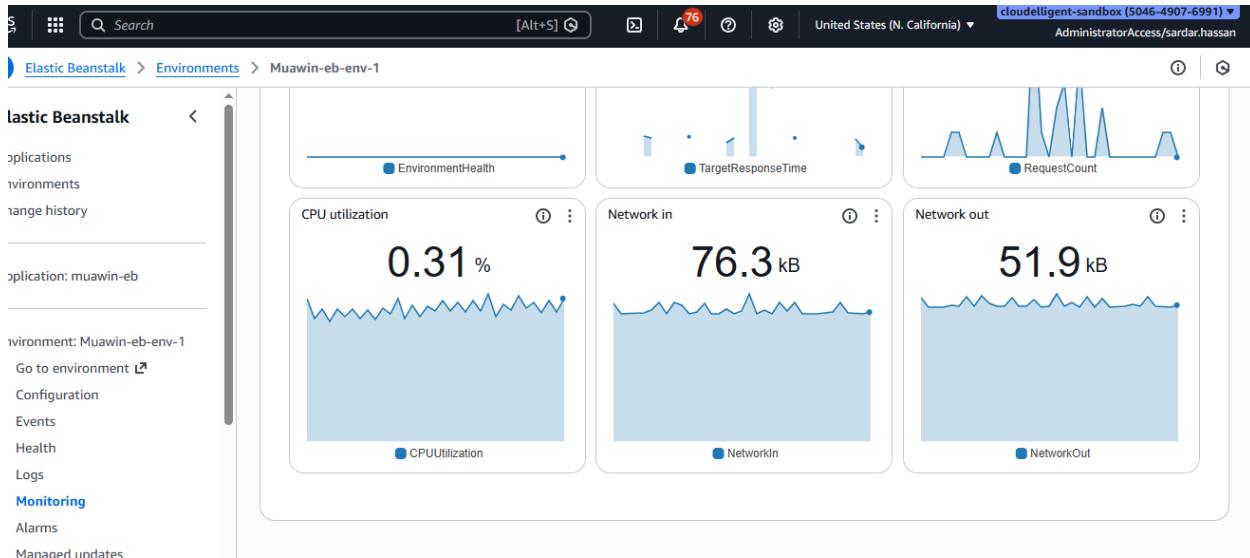
Running version v1.2

Platform state Supported

Events **Health** **Logs** **Monitoring** **Alarms** **Managed updates** **Tags**

Events (42) Info

Time	Type	Details
December 30, 2025 04:48:16 (UTC+5)	INFO	Environment health has transitioned from Info to Ok. Application update completed 50 seconds ago and took 54 seconds.



9. Configure Environment Variables and Scaling Options

Sensitive configuration values were securely configured using **Elastic Beanstalk environment properties**, including:

- MONGODB_URI
- JWT_SECRET
- NODE_ENV=production

These environment variables replaced local .env usage.

The screenshot shows the AWS Elastic Beanstalk Environment Properties configuration page. It includes sections for X-Ray enabled (Disabled) and Environment properties. The Environment properties table lists the following variables:

Source	Key	Value
Plain text	CORS_ORIGIN	https://dp xenmzaw9k b.d.cloudfront.net
Plain text	JWT_SECRET	1234567890abcdefghijklmnopqrstuvwxyz...
Plain text	MONGODB_URI	mongodb+srv://Noor:noor1234@task9-...
Plain text	NODE_ENV	development

10. Test the Deployed Application URL

The backend application was tested using:

- Browser

- API endpoints

The image contains two screenshots of browser windows. Both windows show a JSON response with a "message" field and a "version" field, followed by a "endpoints" object. The "endpoints" object contains various API endpoints such as auth, users, zones, tasks, announcements, assignedTasks, tickets, files, locations, categories, notifications, and applicants. A note at the bottom of the JSON states: "API documentation will be available soon".

```

{
  "message": "Welcome to Muawin API",
  "version": "1.0.0",
  "endpoints": {
    "auth": "/api/auth",
    "users": "/api/users",
    "zones": "/api/zones",
    "tasks": "/api/tasks",
    "announcements": "/api/announcements",
    "assignedTasks": "/api/assigned-tasks",
    "tickets": "/api/tickets",
    "files": "/api/files",
    "locations": "/api/locations",
    "categories": "/api/categories",
    "notifications": "/api/notifications",
    "applicants": "/api/applicants"
  }
}

API documentation will be available soon
  
```

The first screenshot is from a CloudFront distribution URL: `dpxenmzaw9kbd.cloudfront.net/api/`. The second screenshot is from an Elastic Beanstalk environment URL: `muawin-eb-env-1.eba-uuyghj8h.us-west-1.elasticbeanstalk.com/api`.

11. Frontend Deployment (Three-Tier Extension)

To complete the three-tier architecture:

- Frontend static build files were generated
- Uploaded to an Amazon S3 bucket
- S3 static website hosting remained disabled
- Amazon CloudFront distribution was created

The screenshot shows the AWS CloudFront console with the distribution `muawin-cdn-t9` selected. The left sidebar includes sections for CloudFront, Distributions, Policies, Functions, Static IPs, VPC origins, SaaS, Telemetry, and Reports & analytics. The main area displays the distribution's details, including its domain name (`dpxenmzaw9kdb.cloudfront.net`), billing plan (Free plan, \$0/month), ARN, and last modified date (December 29, 2025). Below the details, the **General** tab is active, showing settings like Name (`muawin-cdn-t9`), Description, Price class (Use all edge locations / best performance), Alternate domain names, Standard logging (Available with the Pro plan), and Default root object (`index.html`). At the top, a progress bar indicates an upload process for 16 files (1.9 MB total) is at 89% completion.

CloudFront Behaviors:

- Default (*) → S3 (Frontend)
- `/api/*` → Elastic Beanstalk backend

The screenshot shows the 'Behaviors' tab selected in the AWS CloudFront console. There are two behaviors listed:

Precedence	Path pattern	Origin or origin group	Viewer protocol	Cache policy	Origin request	Response headers
0	/api/*	muawin-eb-e...	Redirect HTTP	Managed-Caching	Managed-AllViewers	-
1	Default (*)	muawin-eb-b...	Redirect HTTP	Managed-Caching	-	-

This enabled:

- HTTPS for frontend
- Secure routing of API calls to backend
- Elimination of mixed-content errors

12. CloudFront Cache Invalidation

CloudFront cache invalidation was used to:

- Force refresh updated frontend files
- Clear cached assets after frontend changes

Invalidation paths used:

/*

This ensured the latest frontend build was served globally.

The screenshot shows the AWS CloudFront Invalidations console. At the top, there are two navigation items: 'E1FNP2DJOFJOMD' and 'I6T7W2I1MDD2LA2FJX3AMLKIQJ'. On the right side, there is a 'Copy to new' button. Below this, the 'Invalidation details' section is displayed. It includes a 'Date created' field showing 'December 29, 2025 at 11:51:09 PM UTC', an 'Object paths' field showing '/', and a 'Status' field showing 'Completed' with a green checkmark icon. A vertical scroll bar is visible on the left side of the page.

13. Update Application Code and Redeploy as Needed

Application updates followed a repeatable process:

1. Update code locally
2. Rebuild frontend or backend
3. Re-zip backend (if applicable)
4. Upload new version to Elastic Beanstalk
5. Invalidate CloudFront cache (frontend)

Elastic Beanstalk versioning allowed safe rollouts.

The screenshot shows the AWS Elastic Beanstalk Application versions page. The URL in the address bar is 'https://console.aws.amazon.com/elasticbeanstalk/applications/muawin-eb/application-versions'. The page title is 'Application versions (3)'. There are three versions listed: v1.2 (created on December 30, 2025), v1.1 (created on December 29, 2025), and v1 (created on December 29, 2025). Each version has a checkbox next to it, and the 'Actions' dropdown menu is visible above the table. The table columns are 'Version label', 'Description', 'Date created', and 'Source'. A search bar at the top allows filtering by application version. A horizontal scrollbar is present at the bottom of the table area.

Version label	Description	Date created	Source
v1.2	—	December 30, 2025 04:45:55 (UTC+5)	1767051954631
v1.1	—	December 29, 2025 23:58:24 (UTC+5)	1767034703364
v1	—	December 29, 2025 23:26:11 (UTC+5)	1767032767976

14. Set Up Monitoring and Alerts for Elastic Beanstalk Environment Health

Elastic Beanstalk health monitoring was used to track:

- Instance health
- HTTP error rates
- Application responsiveness

Health transitions (OK → Severe → OK) were analyzed and resolved by:

- Fixing health check paths
- Ensuring backend root responses were valid

This validated production readiness.

CloudWatch > **Alarms** > awseb-e-c2vpmckna7-stack-AWSEBCloudwatchAlarmHigh-TYFuxBBBoodj

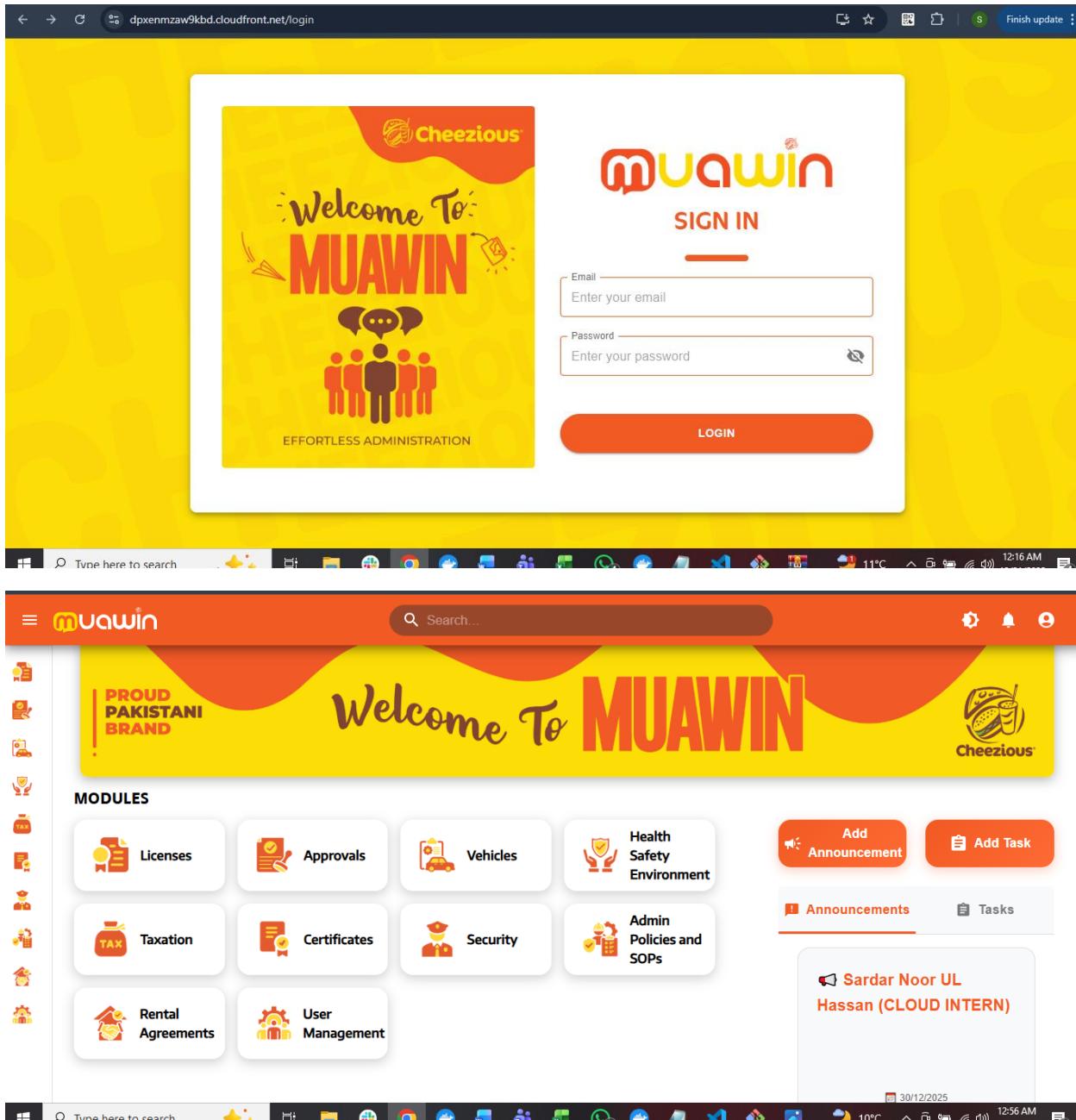
Details

Name	awseb-e-c2vpmckna7-stack-AWSEBCloudwatchAlarmHigh-TYFuxBBBoodj	State	OK
Type	Metric alarm	Threshold	NetworkOut > 6000000 for 1 datapoints within 5 minutes
Description	ElasticBeanstalk Default Scale Up alarm	Actions	Actions enabled
		Last state update	2025-12-30 17:45:11 (UTC)
		Statistic	Average
		Period	5 minutes
		Datapoints to alarm	1 out of 1
		Missing data treatment	Treat missing data as missing
		Percentiles with low samples	evaluate
		ARN	arn:aws:cloudwatch:us-west-1:504649076991:alarm:awseb-e-c2vpmckna7-stack-AWSEBCloudwatchAlarmHigh-TYFuxBBBoodj

15. Final Outcome

At the end of this project:

- Backend Node.js API is live on Elastic Beanstalk
- Database is securely connected via MongoDB Atlas
- Frontend is served globally using S3 + CloudFront
- HTTPS is enforced end-to-end
- Application follows a clean three-tier architecture
- Deployment is scalable, monitored, and production-ready



16. Skills & Technologies Demonstrated

- AWS Elastic Beanstalk
- Node.js Production Deployment
- MongoDB Atlas Integration
- Amazon S3 Static Hosting

- Amazon CloudFront
- HTTPS & Security Best Practices
- Environment Variable Management

17. Conclusion

This project successfully transformed a local Node.js application into a **fully functional, cloud-hosted, production-grade system** using AWS managed services.