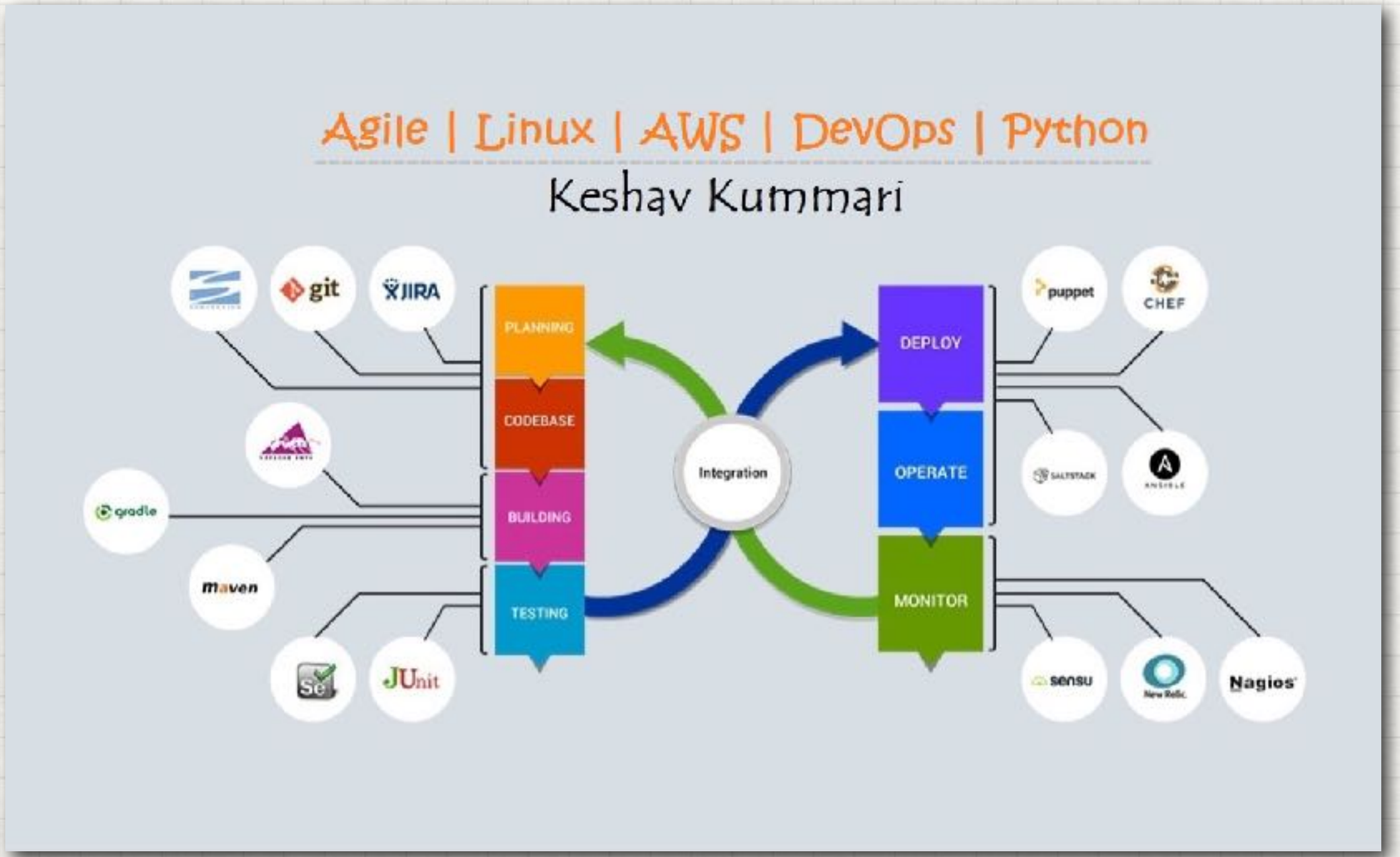
AWS

AWS - ELASTIC BEANSTALK



AWS ELASTIC BEANSTALK

1. WHAT IS ELASTIC BEANSTALK?

- AWS Elastic Beanstalk is an easy-to-use service for **deploying** and **scaling web applications** and **services** developed with **Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker** on familiar servers such as **Apache Tomcat, Nginx, Passenger, and IIS**.

- You can simply **upload your code** and **Elastic Beanstalk** automatically handles the **deployment**, from **capacity provisioning**, **load balancing**, **auto-scaling** to **application health monitoring**.

- At the same time, you **retain full control** over the **AWS resources powering** your application and can access the **underlying resources** at any time.

- There is no additional charge for Elastic Beanstalk **- you pay only for** the **AWS resources** needed to **store** and **run your applications**.

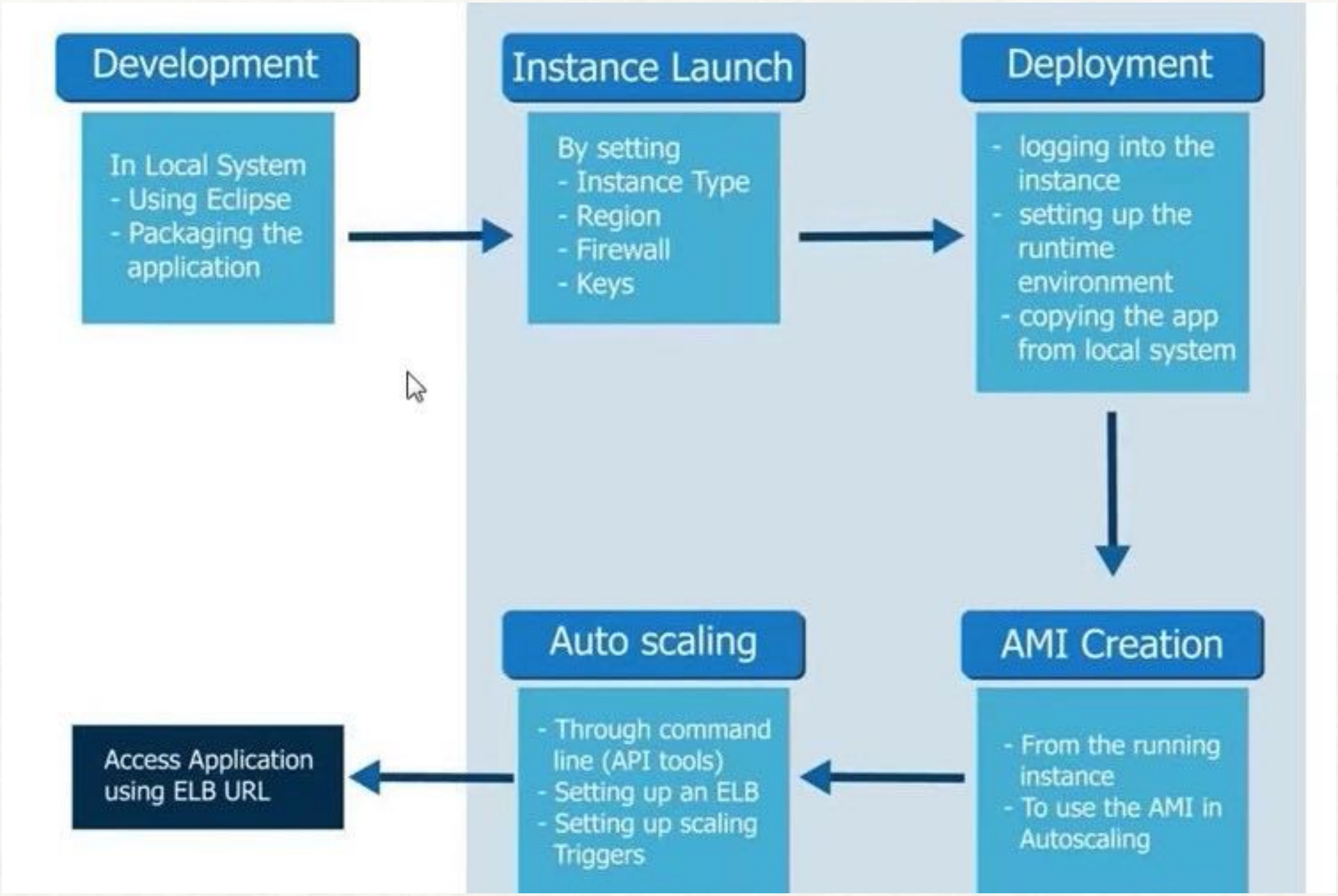
- With Elastic Beanstalk, you can **deploy, Monitor,** and **scale** an application quickly.

- It provides **developers** or **end users** with the ability to **provision application infrastructure** is an almost **transparent way**.

- It has a highly **abstract focus** towards **infrastructure**, focusing on **components** and **performance - not configuration** and **specifications**.

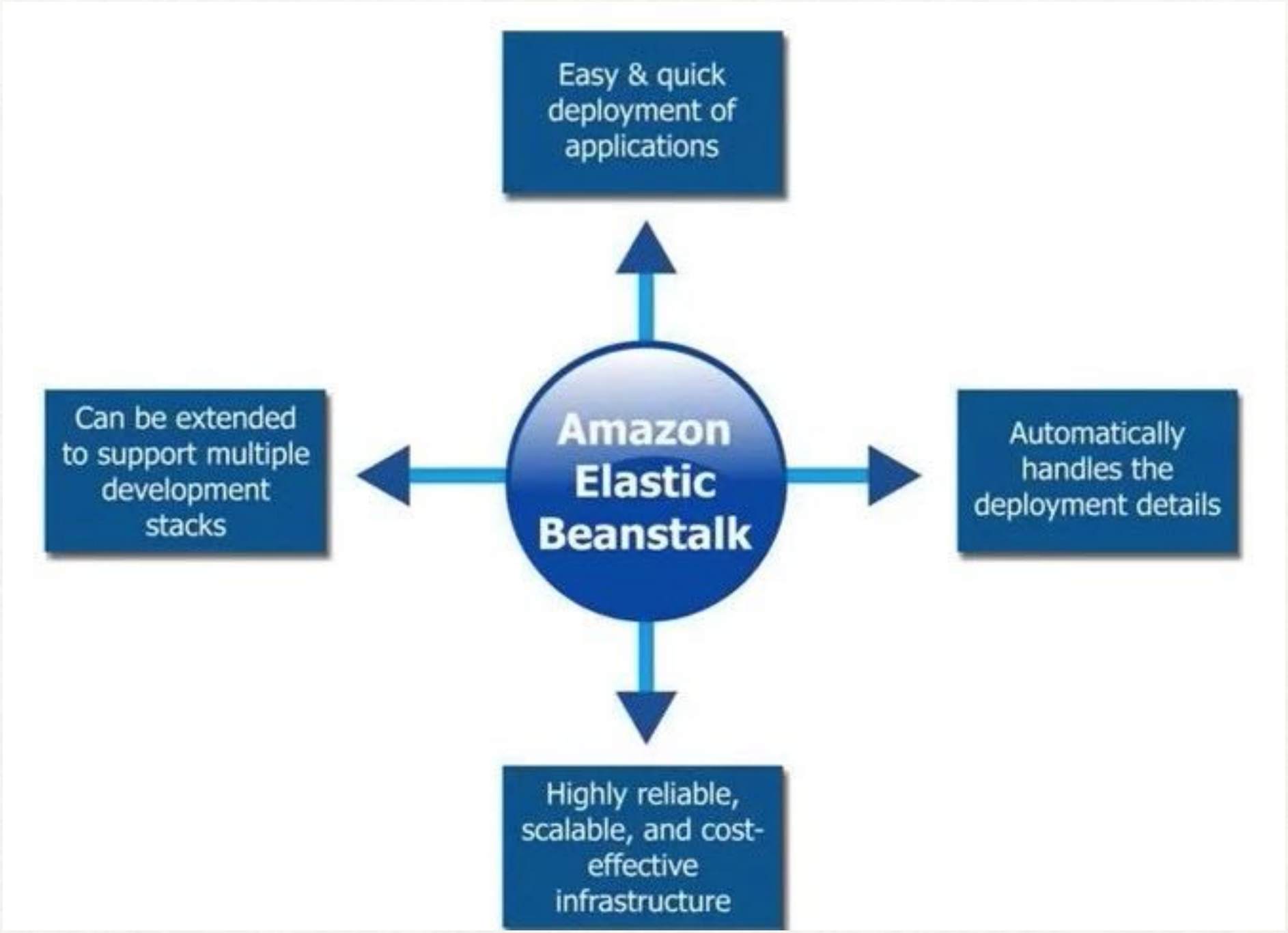
- It attempts to **remove, or significantly** simplify infrastructure management, allowing **applications to deployed** into **infrastructure environments easily**.

DEVELOP & DEPLOY FLOW



Development & Operation Work Flow using Elastic Bean Stack

USAGES OF ELASTIC BEANSTACK



EB - FLOW

BEANSTALK KEY ARCHITECTURE COMPONENTS

- Applications are the high level structure in beanstalk.
- Either your entire Application, is one EB Application, or
- Each logical component of your Application, can be a EB Application or a EB environment within an application.

EB

BEANSTALK KEY ARCHITECTURE COMPONENTS

- Applications can have multiple environments(Dev, Test, Acceptance, & Production)
- or functional type(Front-End,Back-End)
- Environments are either single instance or scalable
- Environments are either web server environments or worker environments

Elastic Bean Stak

BEANSTALK KEY ARCHITECTURE COMPONENTS

- Application Versions are unique packages which represent versions of apps.
- An Application is uploaded to Elastic beanstalk as an application bundle i.e. .zip
- Each "Application" can have many versions 1:M relationship.
- Application versions can deployed to environments within an application.

AWS EB

ELASTIC BEANSTALK ADVANTAGES

1. Developer retain full control over the AWS resources
2. Root access to your EC2 instances
3. Easily manage configuration changes in one place
4. Use any database - Amazon RDS, Amazon SimpleDB, or Oracle
5. Create custom AMIs
6. Run other services side-by-side in EC2
7. Easily move your application out of Elastic Beanstalk

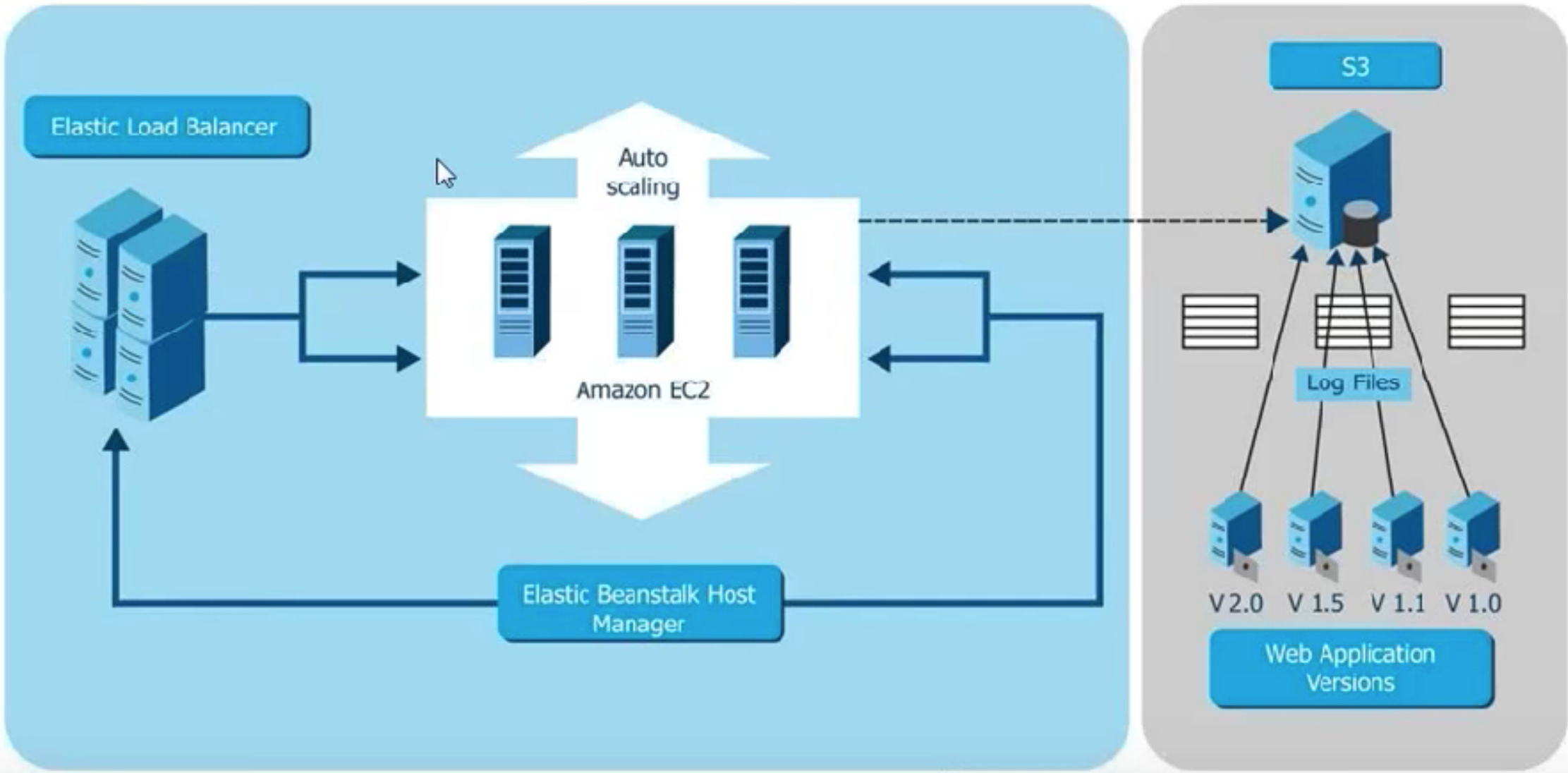
HIGHLIGHTS OF EB

- Easy to Use
  Automated Scalability
- Complete Control
- Flexible
- Reliable
- Free of Cost

Why Not Use My Own App Server AMI?

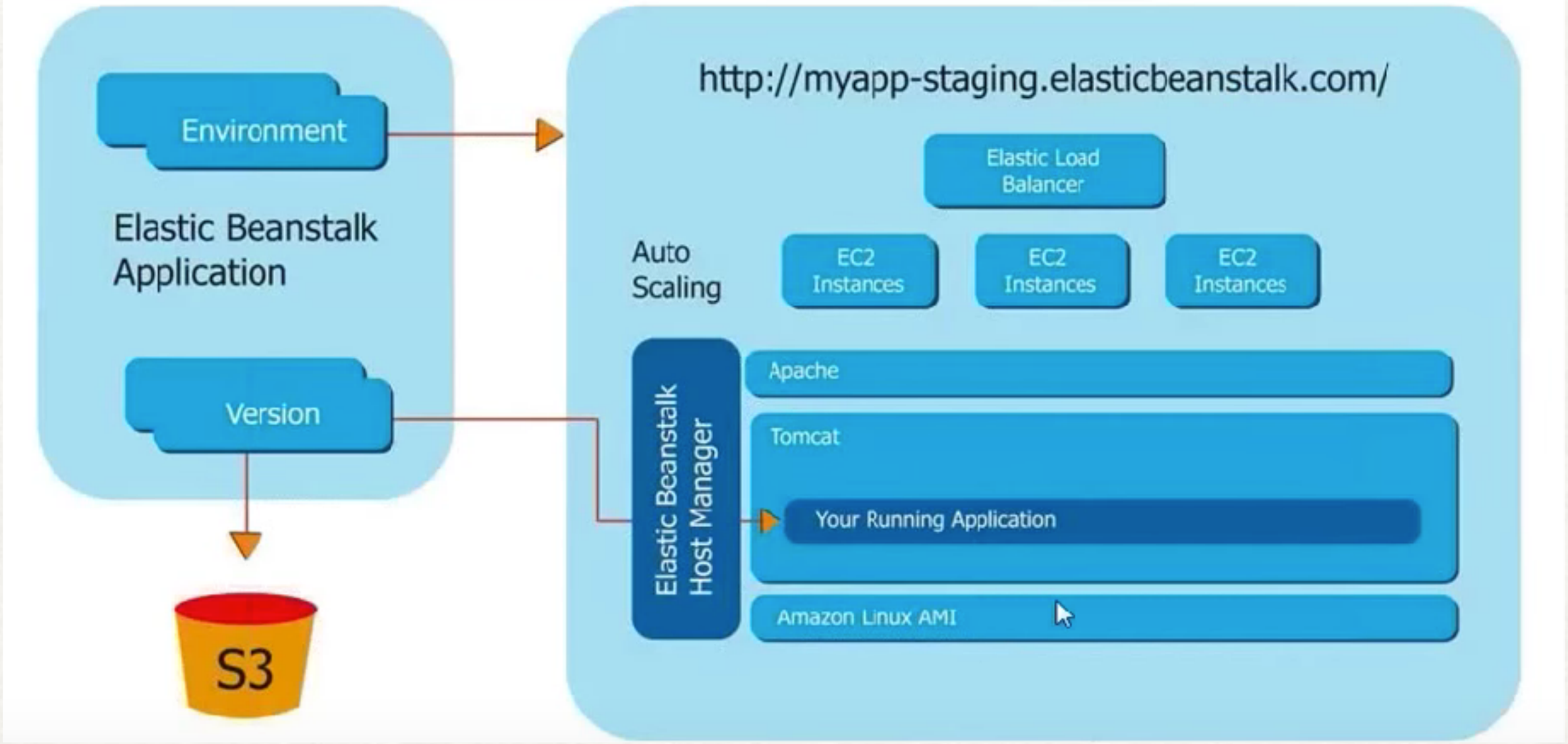
- Automated Provisioning & De-Provisioning of Environments
- Automated version deployment(Including Rollback)
- Managed Environments settings
- Built-in monitoring and notifications
  - Application Health and Other Important Events
- Basic log file rotation to Amazon S3
- Easy troubleshooting
  - Snapshot logs
  - Restart application server

FLOW OF AWS ELASTIC BEANSTALK



Architecture of EB

EXAMPLE OF AWS EB

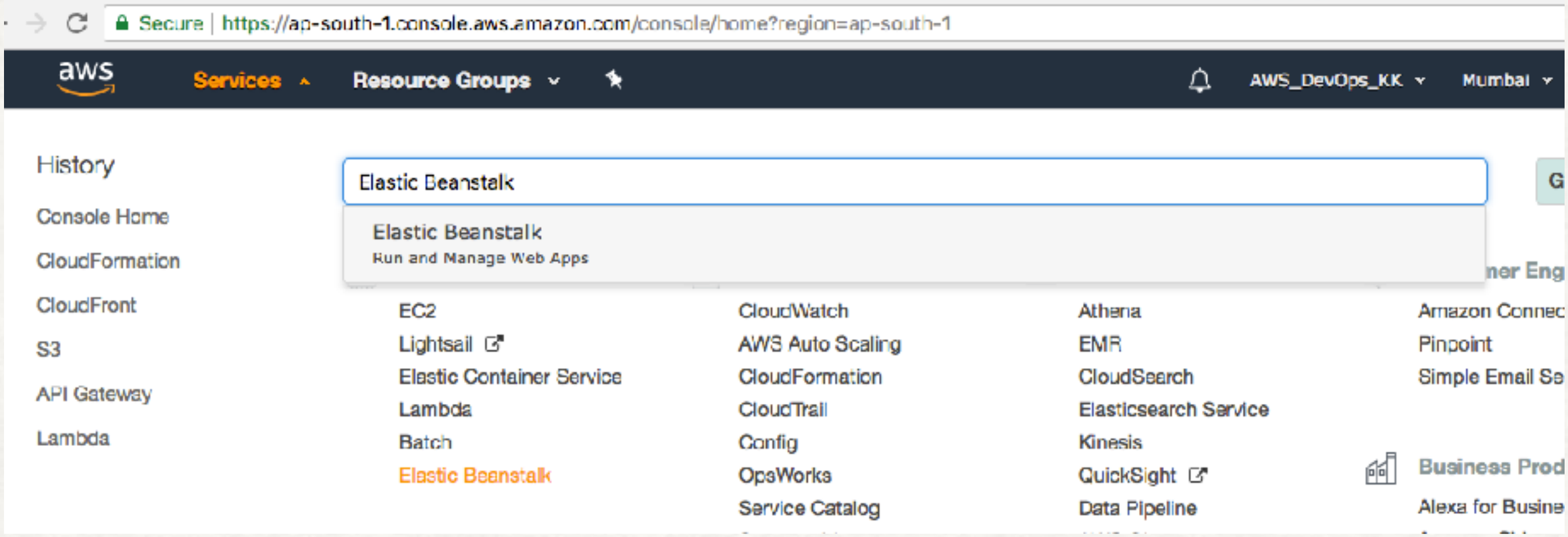


EB FLOW

HOW DO I ACCESS ELASTIC BEANSTALK?

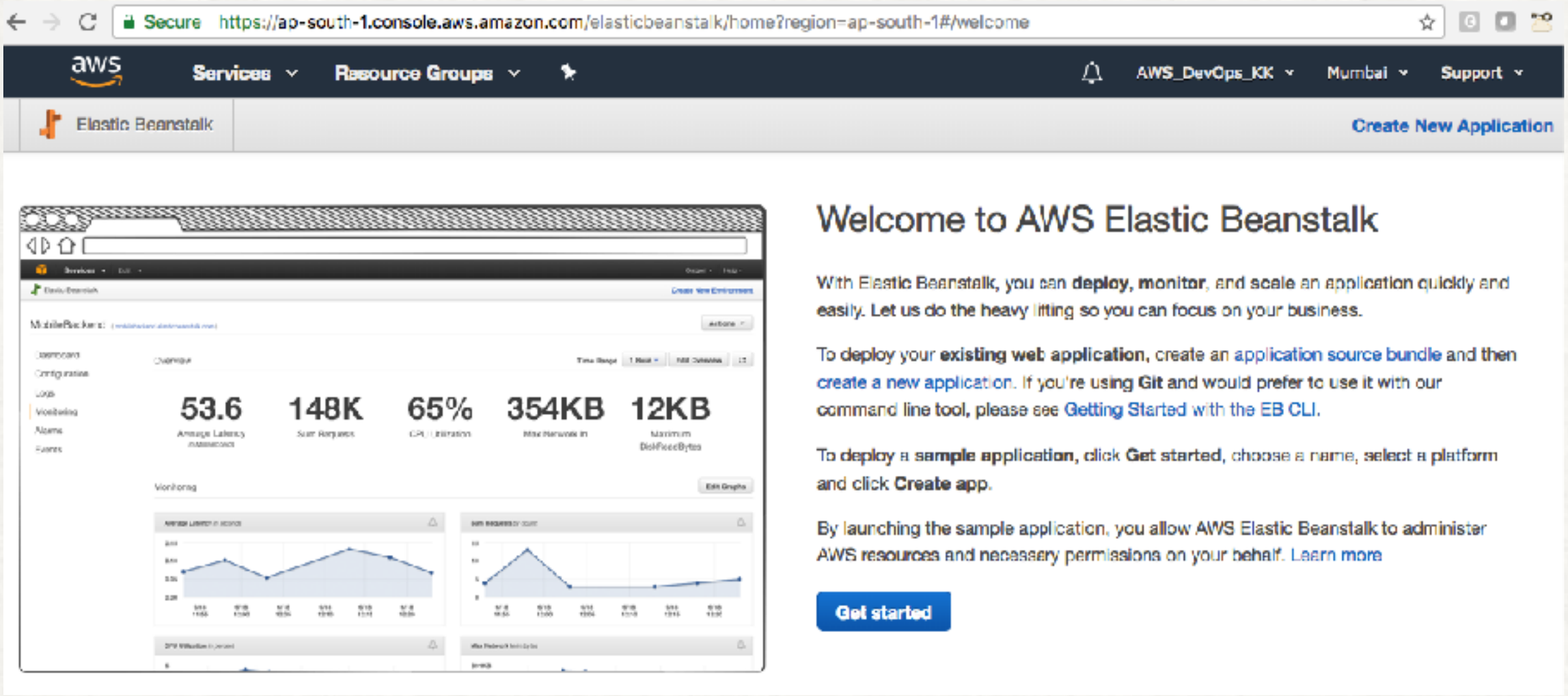
- AWS Management Console
- AWS Toolkit for Eclipse, Visual Studio
- Command Line Tools
- SDK's and API

STEP BY STEP PRACTICAL STEPS OF EB



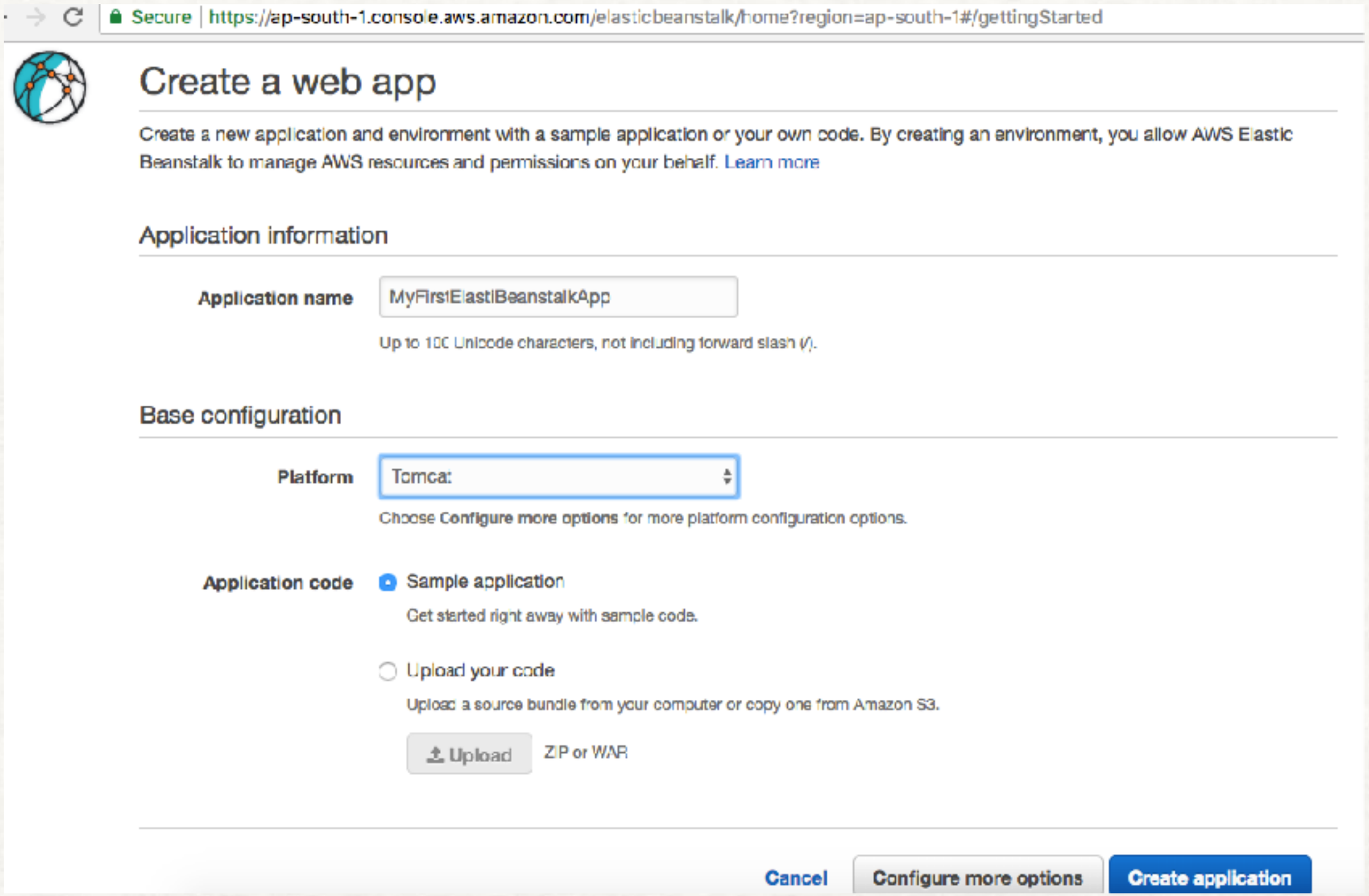
Step-1 : Go to AWS Management Console & Search for EB

STEP-2 : CLICK ON "GET STARTED"



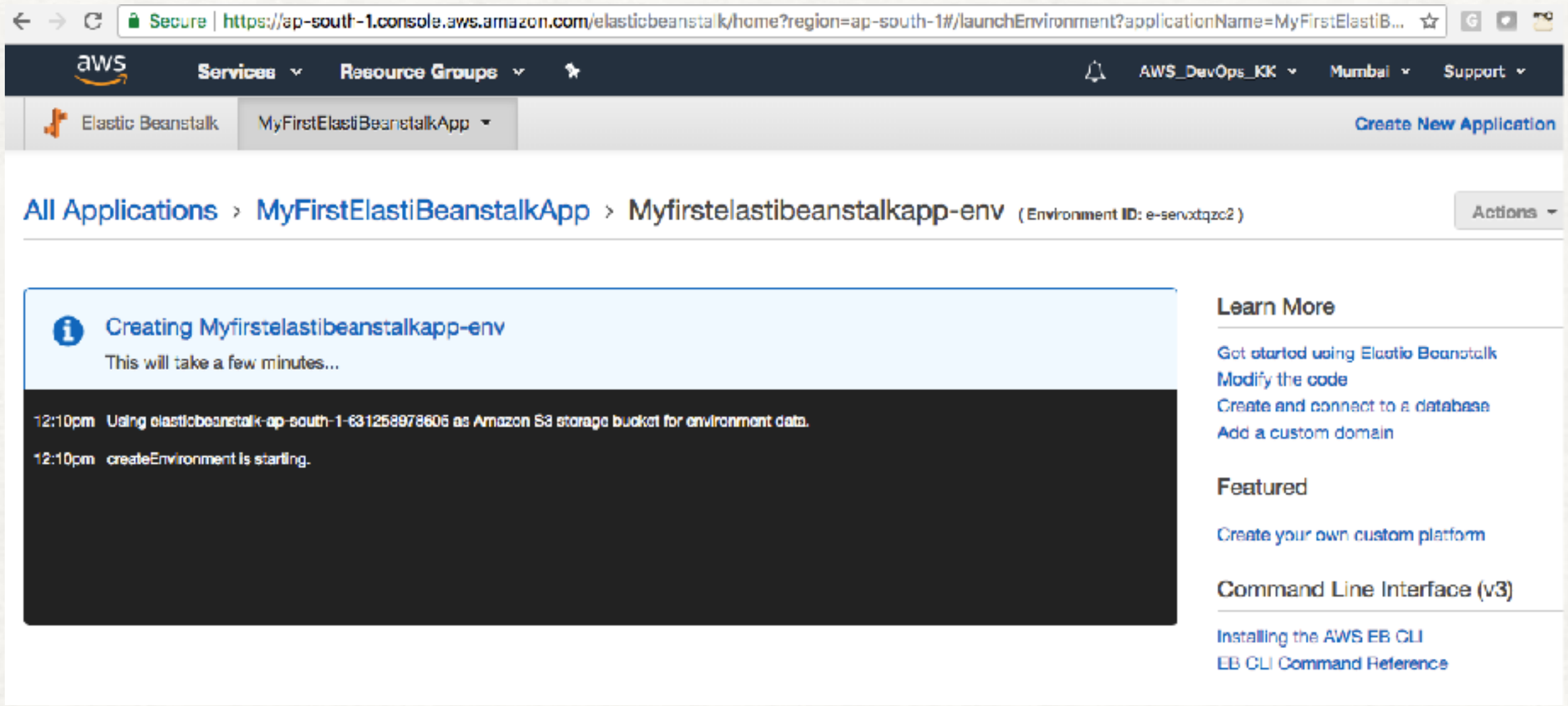
EB

STEP-3 : PROVIDE APPLICATION NAME & PLATFORM



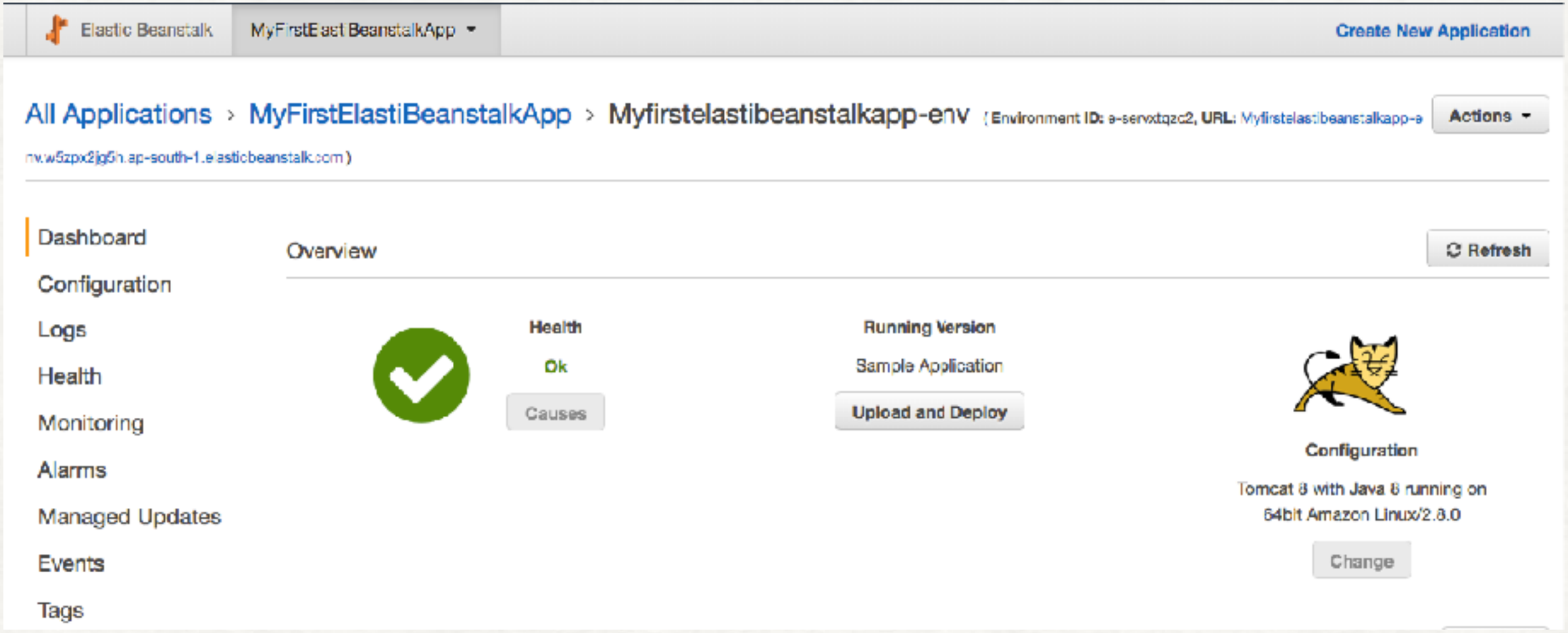
Click on “Create Application”

STEP-4 : IT'S CREATING NOW!



Observe the output

STEP-5 : IT'S CREATED!



Navigate each tab which is located at Left Side Panel

STEP-6: GO THROUGH ALL THE LEFT SIDE PANEL TABS

- Dashboard
- Configuration
- Logs
- Health
- Monitoring
- Alarms
- Managed Updates
- Events
- Tags

Events

Refresh

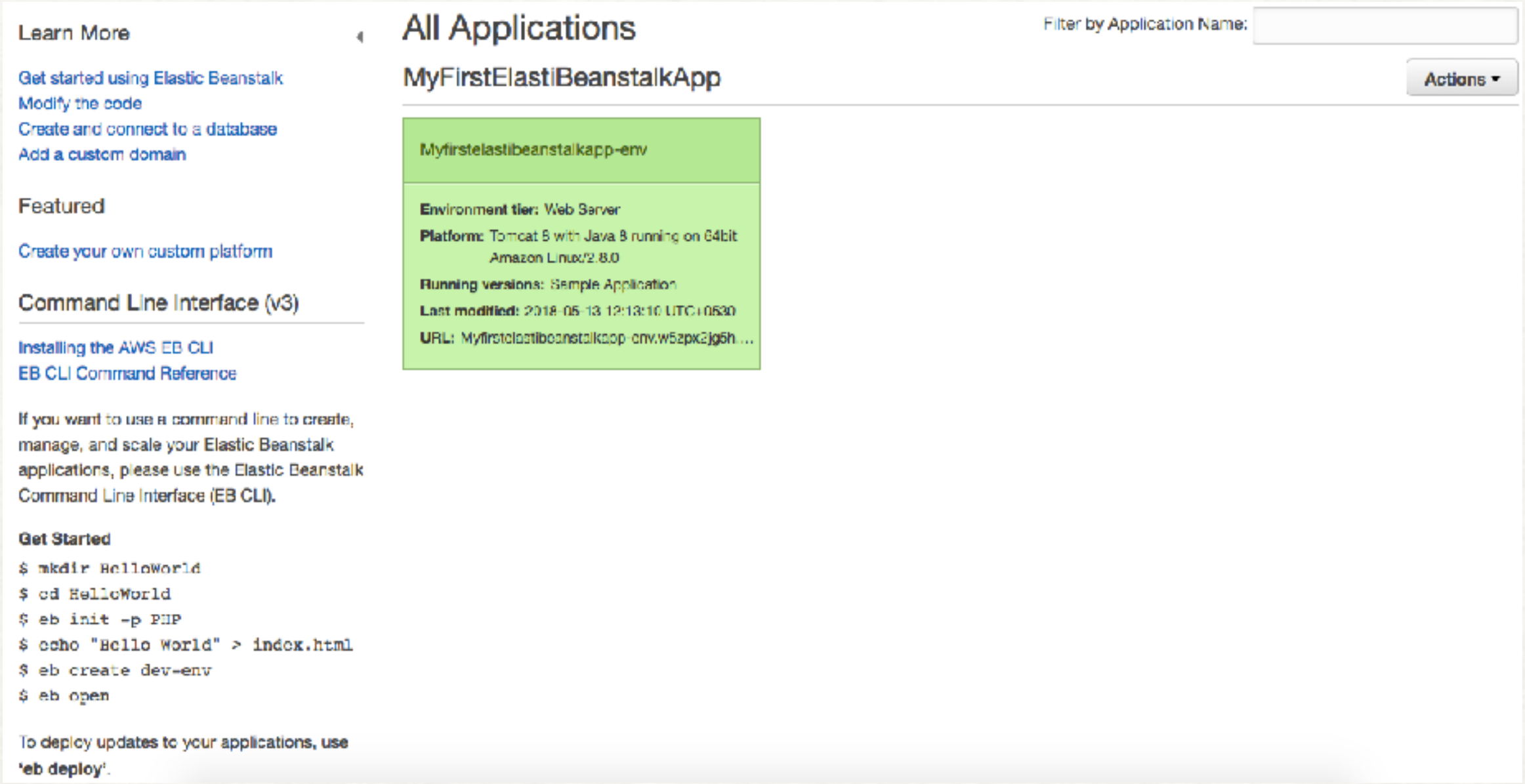
Severity TRACE

2018-03-25 12:15:00 UTC+0530 — 2018-05-13 12:18:00 UTC+0530

Time	Type	Details
2018-05-13 12:13:10 UTC+0530	INFO	Successfully launched environment: Myfirstelastibeanstalkapp-env
2018-05-13 12:12:45 UTC+0530	INFO	Environment health has transitioned from Pending to Ok. Initialization completed 9 seconds ago and took 2 minutes.
2018-05-13 12:11:46 UTC+0530	INFO	Added instance [i-00d1524174c7f2250] to your environment.
2018-05-13 12:11:37 UTC+0530	INFO	Waiting for EC2 instances to launch. This may take a few minutes.
2018-05-13 12:10:49 UTC+0530	INFO	Created EIP: 52.66.134.146
2018-05-13 12:10:48 UTC+0530	INFO	Environment health has transitioned to Pending. Initialization in progress (running for 24 seconds). There are no instances.
2018-05-13 12:10:33 UTC+0530	INFO	Created security group named: awseb-e-servxtqzc2-stack-AWSEBSecurityGroup-U1W7LNN3AGW5
2018-05-13 12:10:09 UTC+0530	INFO	Using elasticbeanstalk-ap-south-1-631258978605 as Amazon S3 storage bucket for environment data.
2018-05-13 12:10:08 UTC+0530	INFO	createEnvironment is starting.

We are looking at one of the TAB output i.e. Events

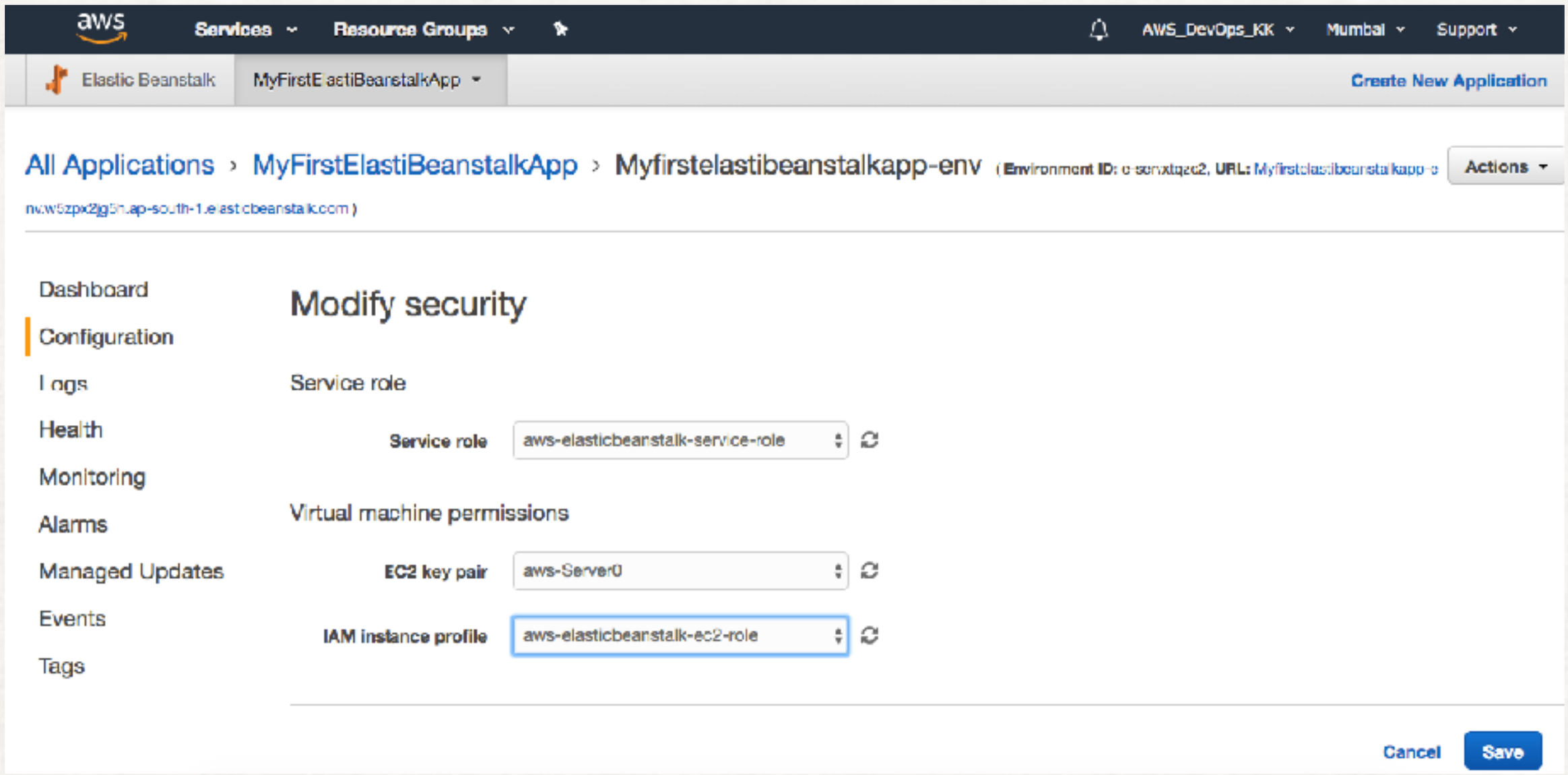
STEP-7 : CHECK THE CLI COMMANDS



AWS CLI Commands

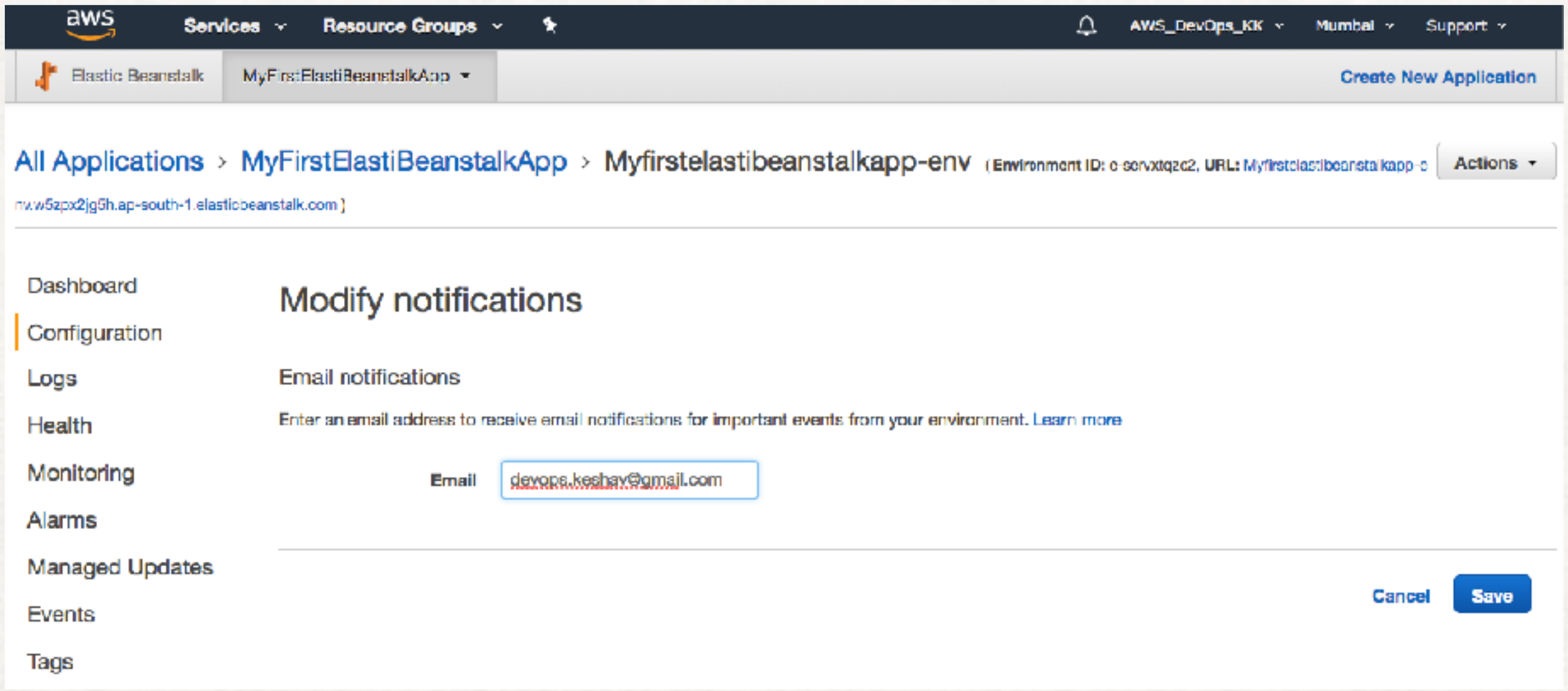
STEP-8: DO REQUIRED CONFIGURATION CHANGES

- Add “Private Key”



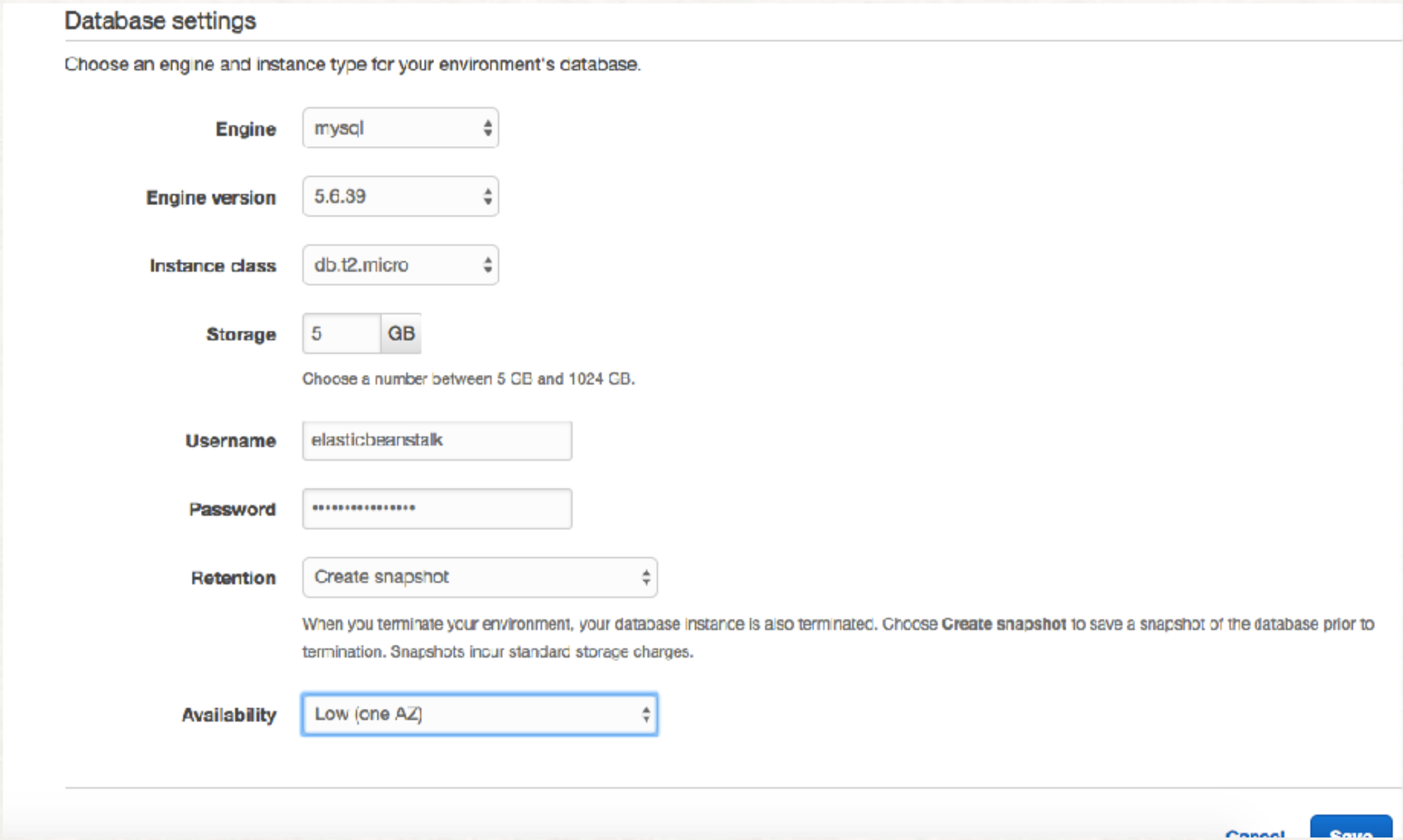
Click on Save

STEP-9 : ADD EMAIL ID FOR ALERTS



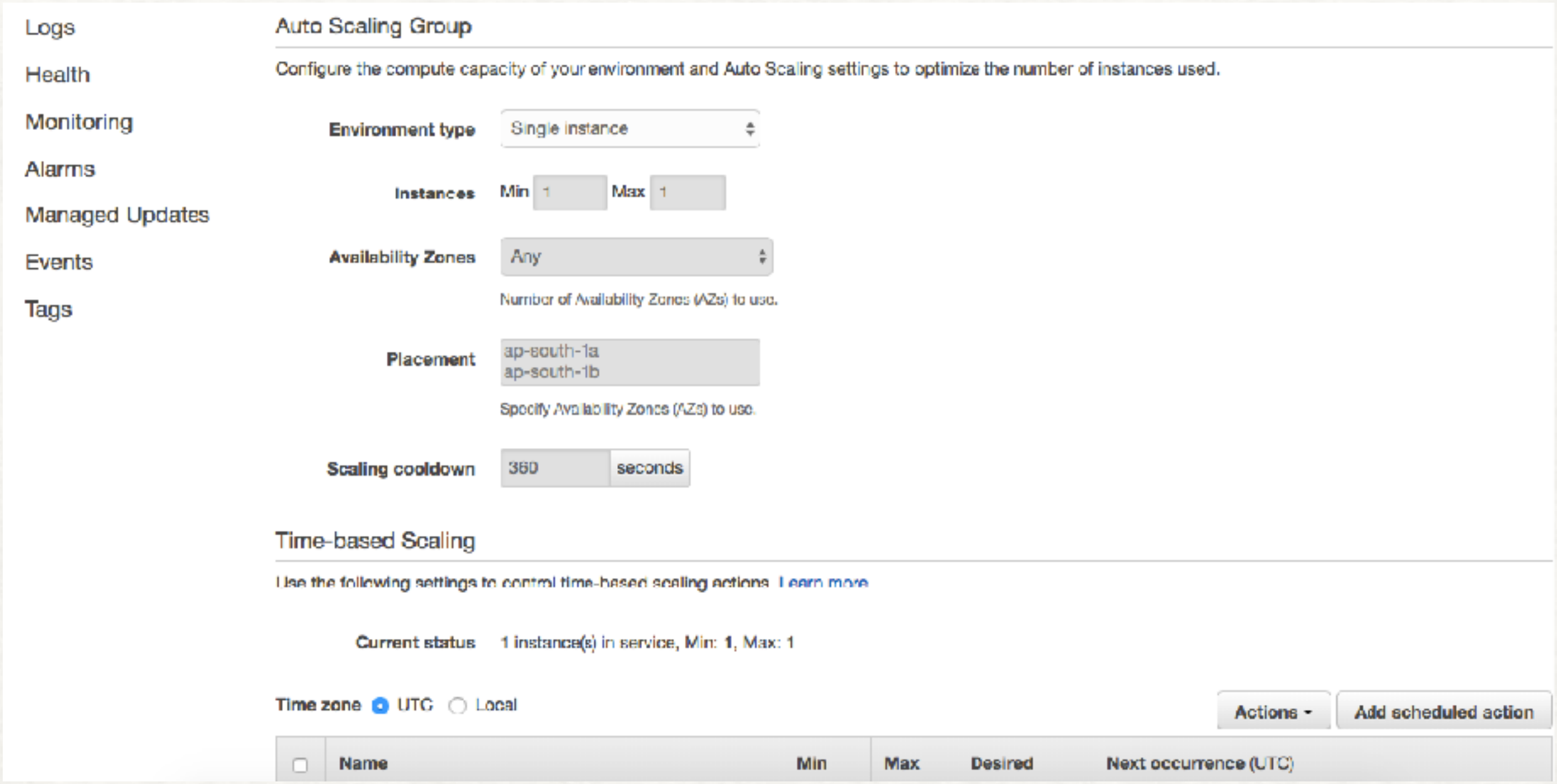
Click on “Save”

STEP-10 : ADD DB PARAMETERS



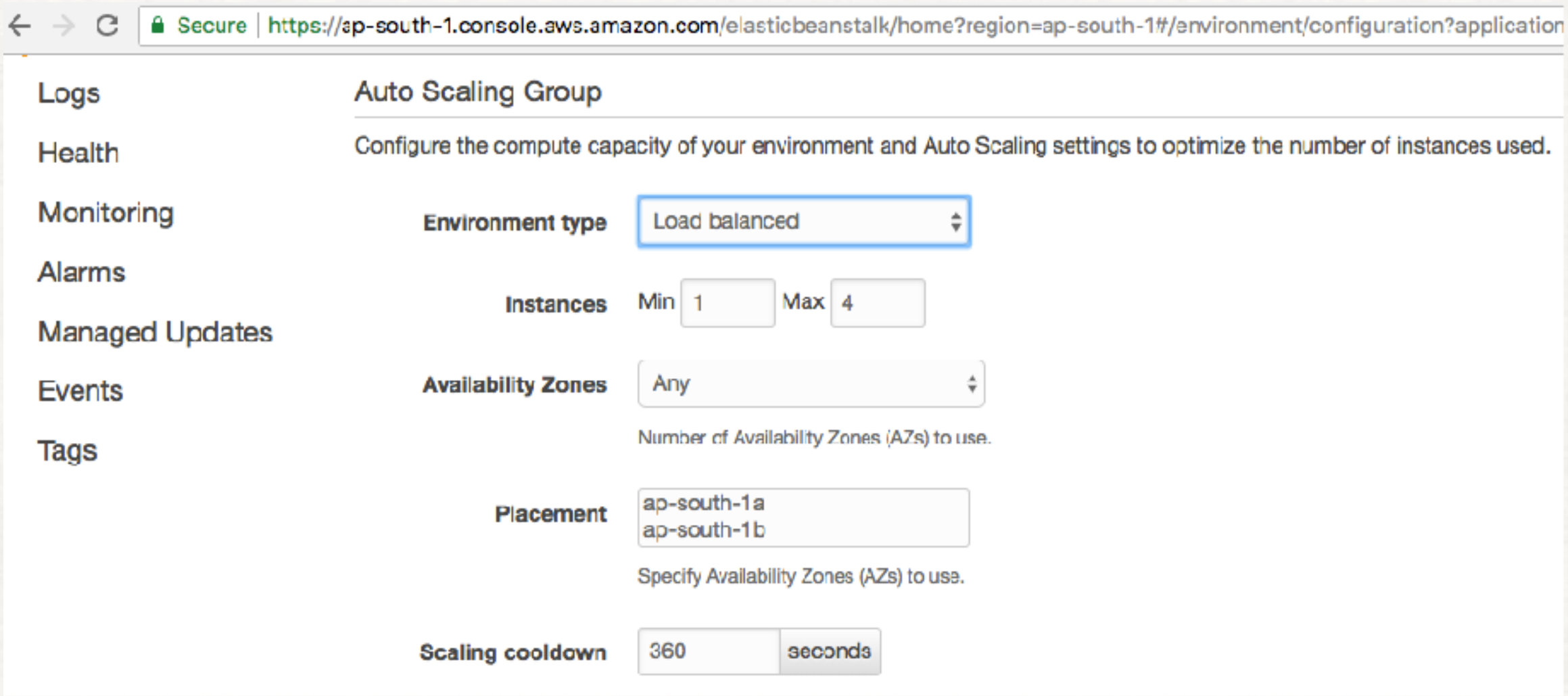
Click on “Save”

STEP-11: MODIFY THE AUTO SCALING GROUP AS PER THE REQUIREMENT



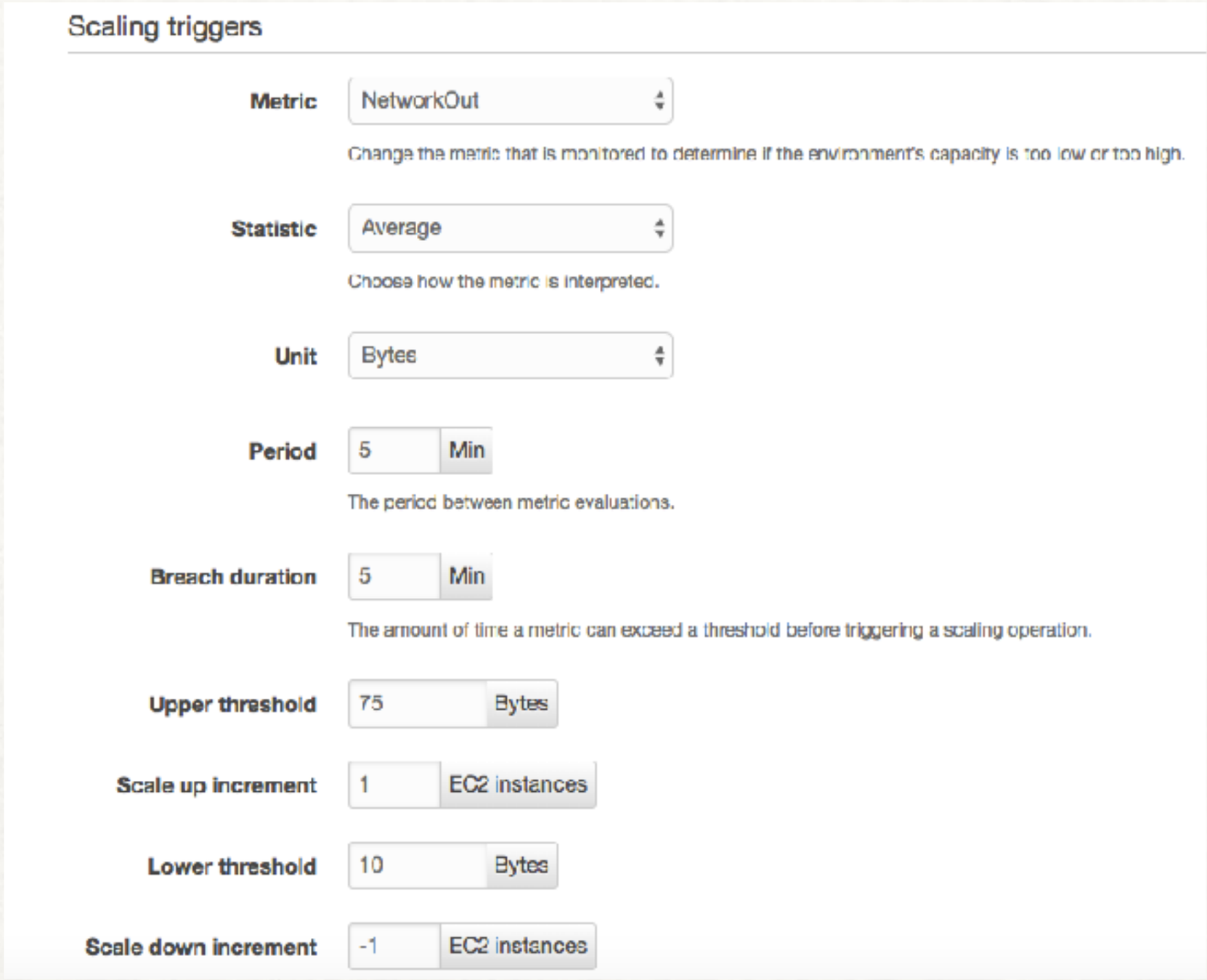
Edit required things

STEP-12 : ADD AUTOSCALING GROUP



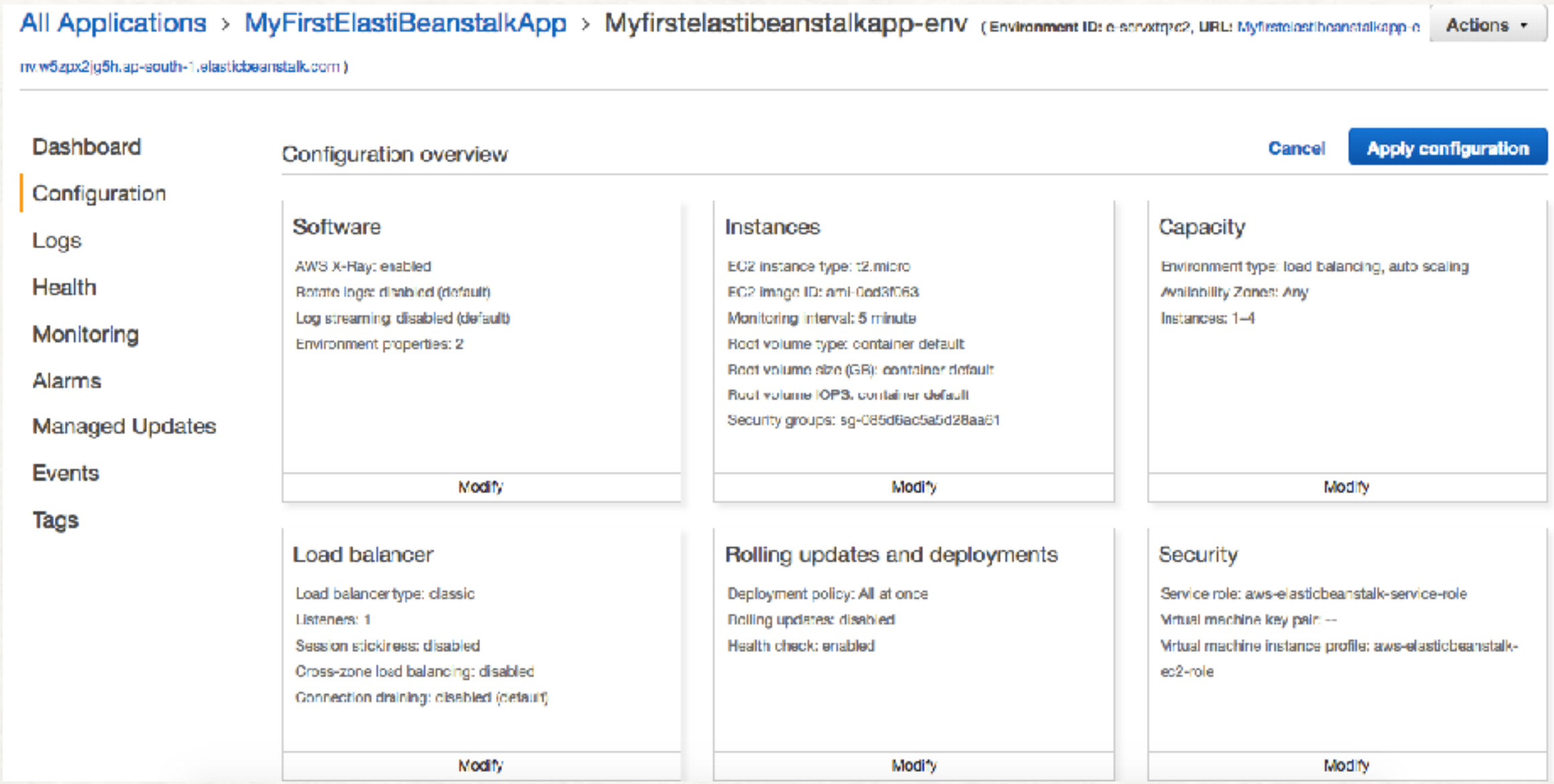
Select Load Balancer & Add the AutoScaling

ADD AUTO SCALING



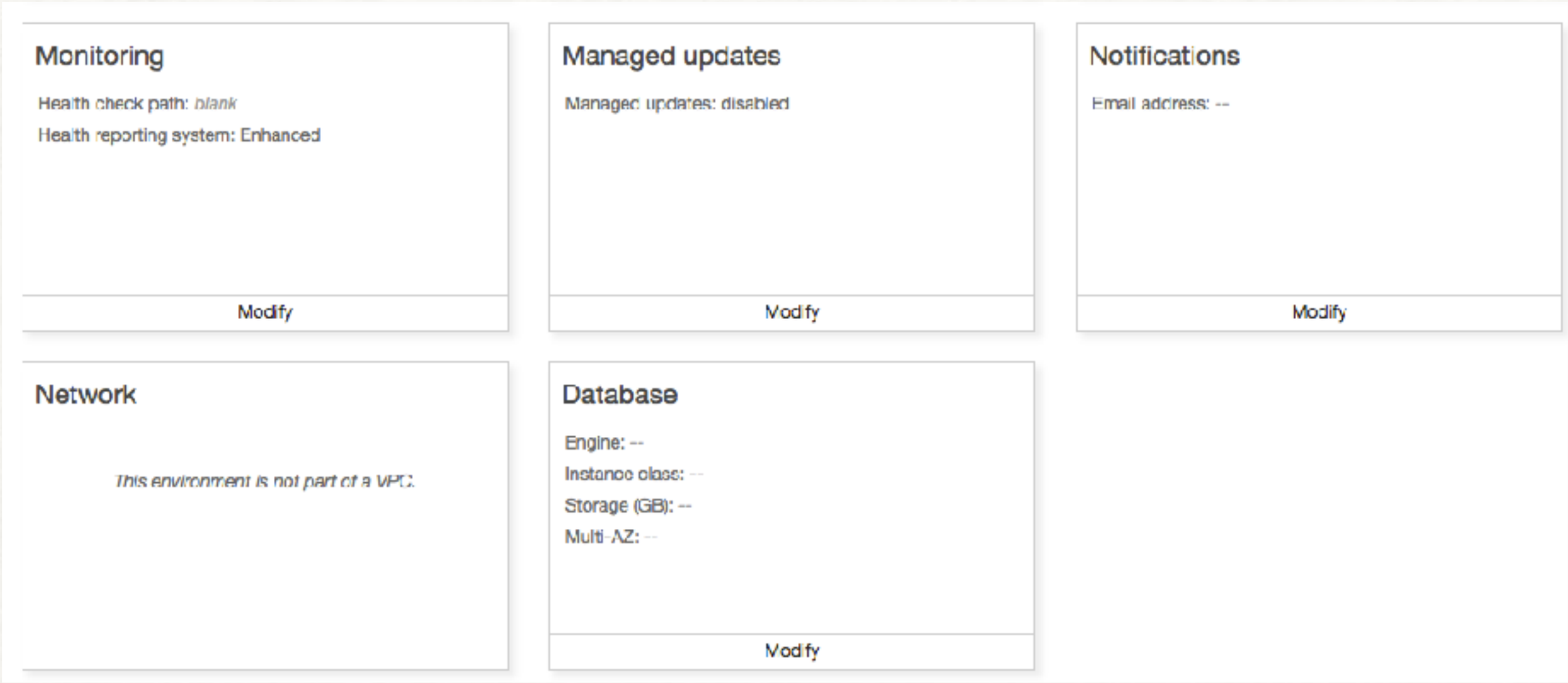
Click on “Save”

STEP-13 : PART OF CONFIGURATION TAB, WE HAVE MODIFIED FEW



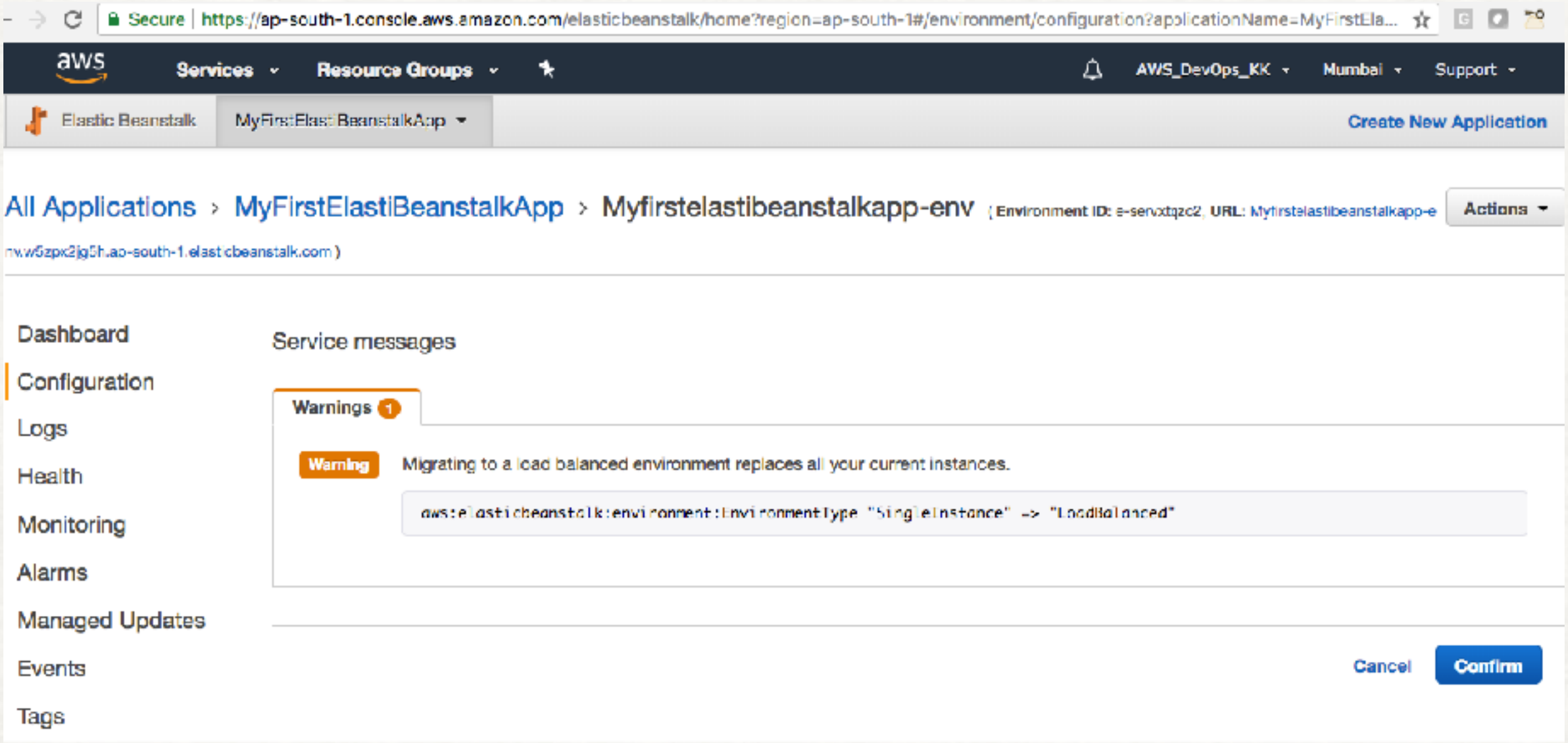
Scroll Down

PAGE 2



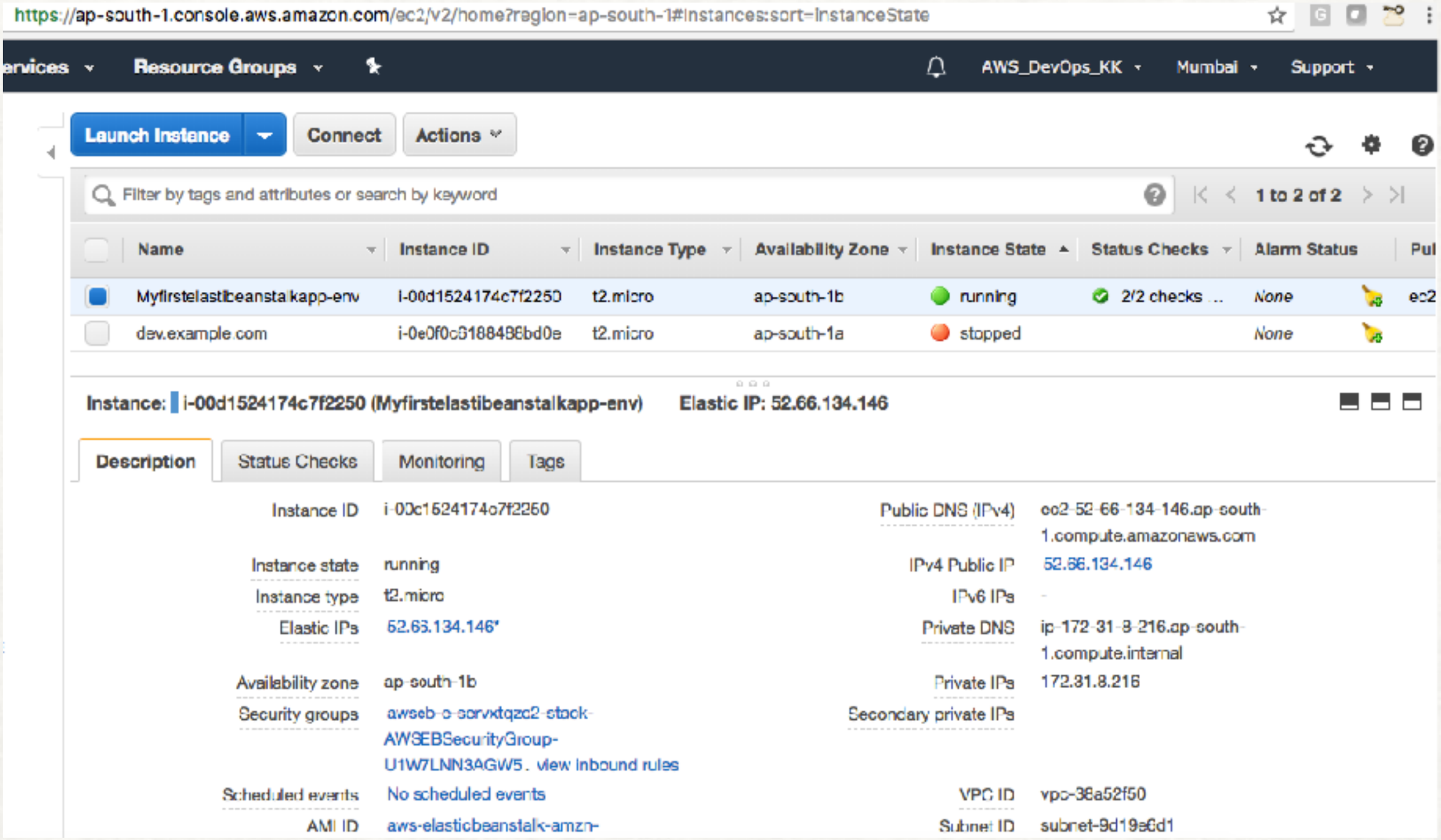
Scroll Up & Click on “Apply Configuration”

STEP-14 : APPLYING ALL THE CONFIGURATION CHANGES WERE MADE BY US



Click on “Confirm”

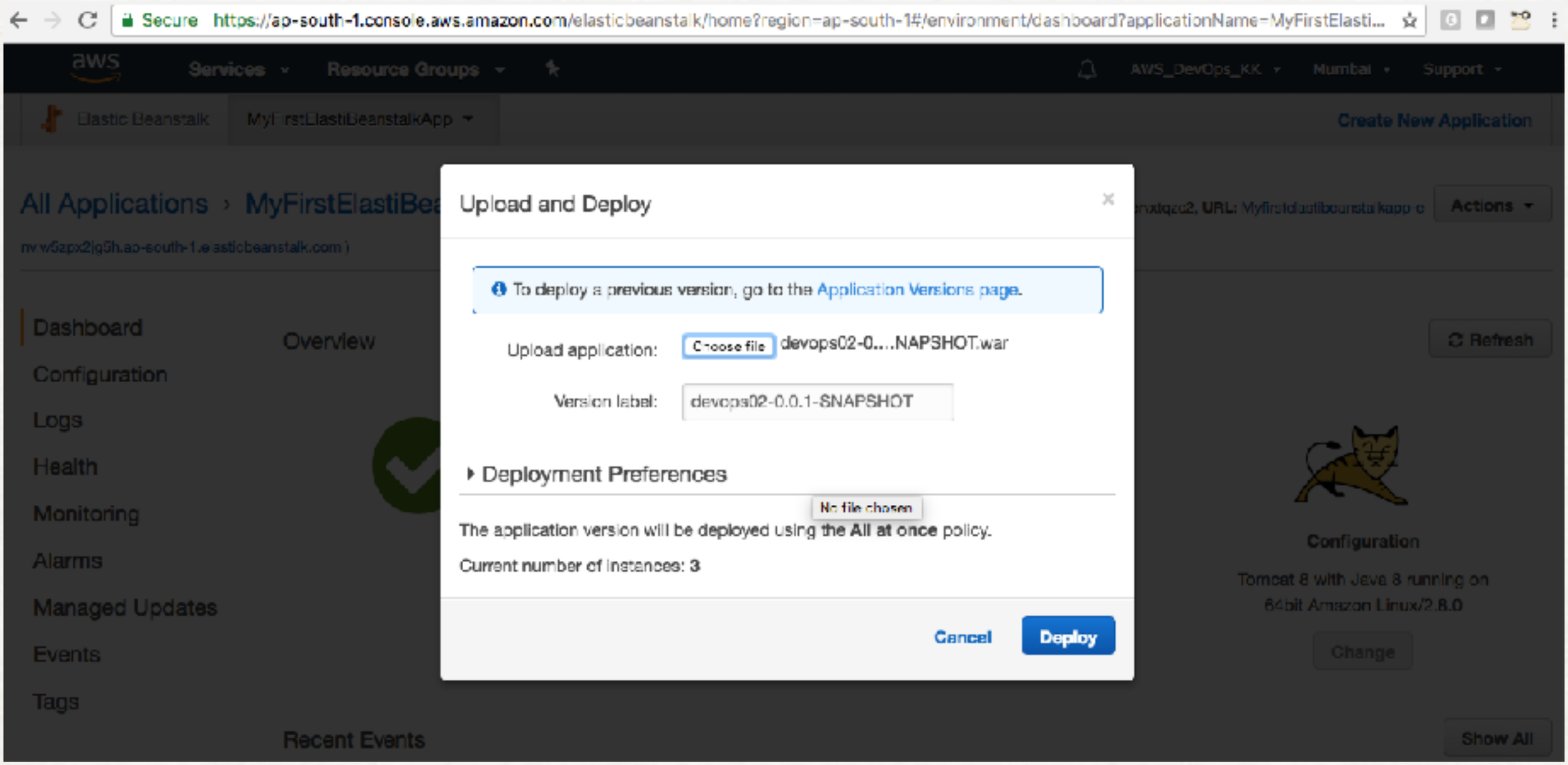
STEP-15 : CROSS CHECK THE RESOURCES



EC2 Instance is created

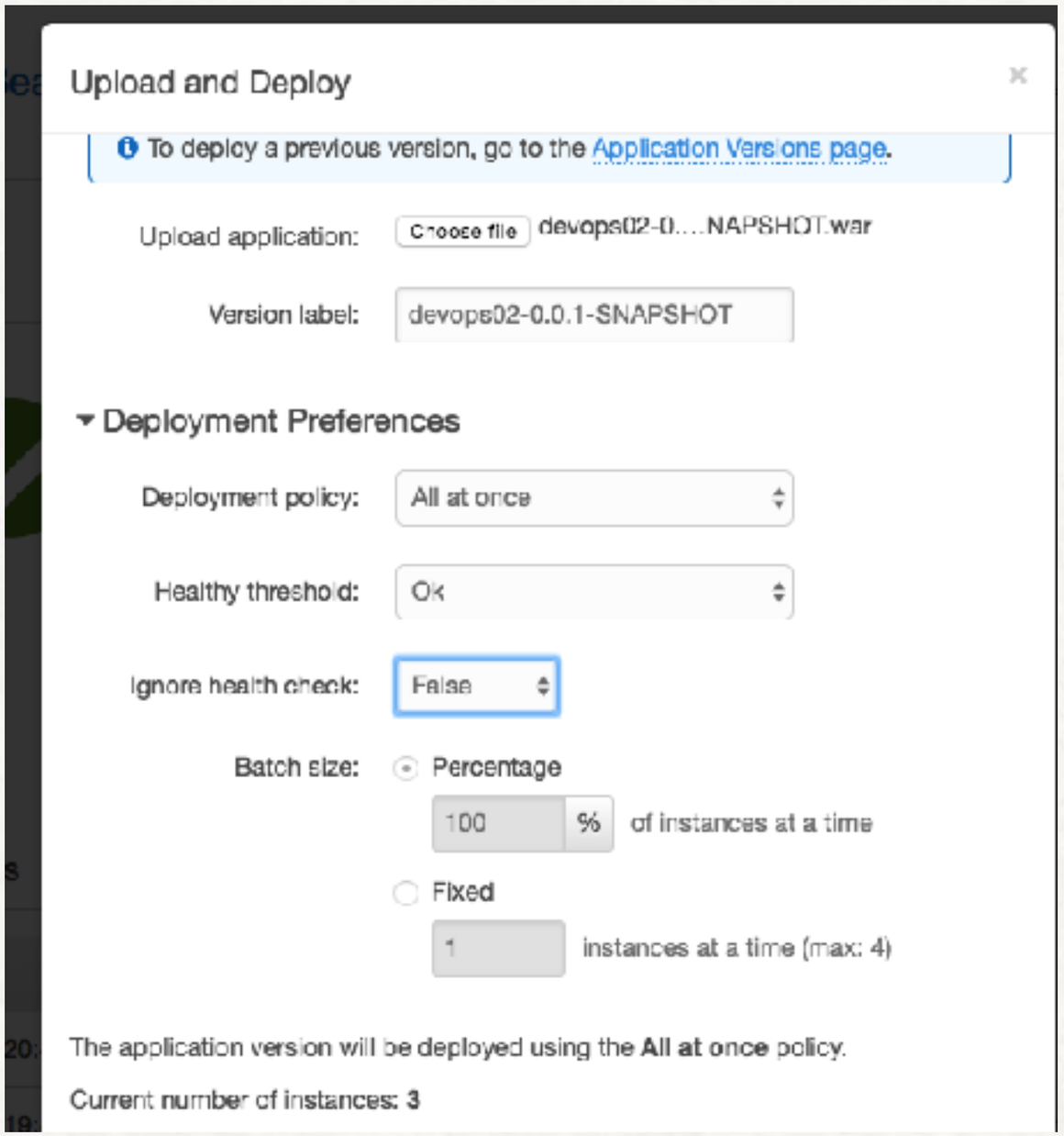
STEP-16 : CLICK ON UPLOAD & DEPLOY A JAVA BUILD

- On my Local Machine I have Have build i.e. devops02-0.0.1-SNAPSHOT.war file



Upload a .war file

Click on Deployment Preference & Check the configuration



Click on Deploy

STEP-17 : CLICK ON APPLICATIONS VERSIONS & CHECK THE LIST OF VERSIONS

Secure https://ap-south-1.console.aws.amazon.com/elasticbeanstalk/home?region=ap-south-1#/application/versions?applicationName=MyFirstElastiBea...

aws Services Resource Groups AWS_DevOps_KK Mumbai Support

Elastic Beanstalk MyFirstElastiBeanstalkApp Create New Application

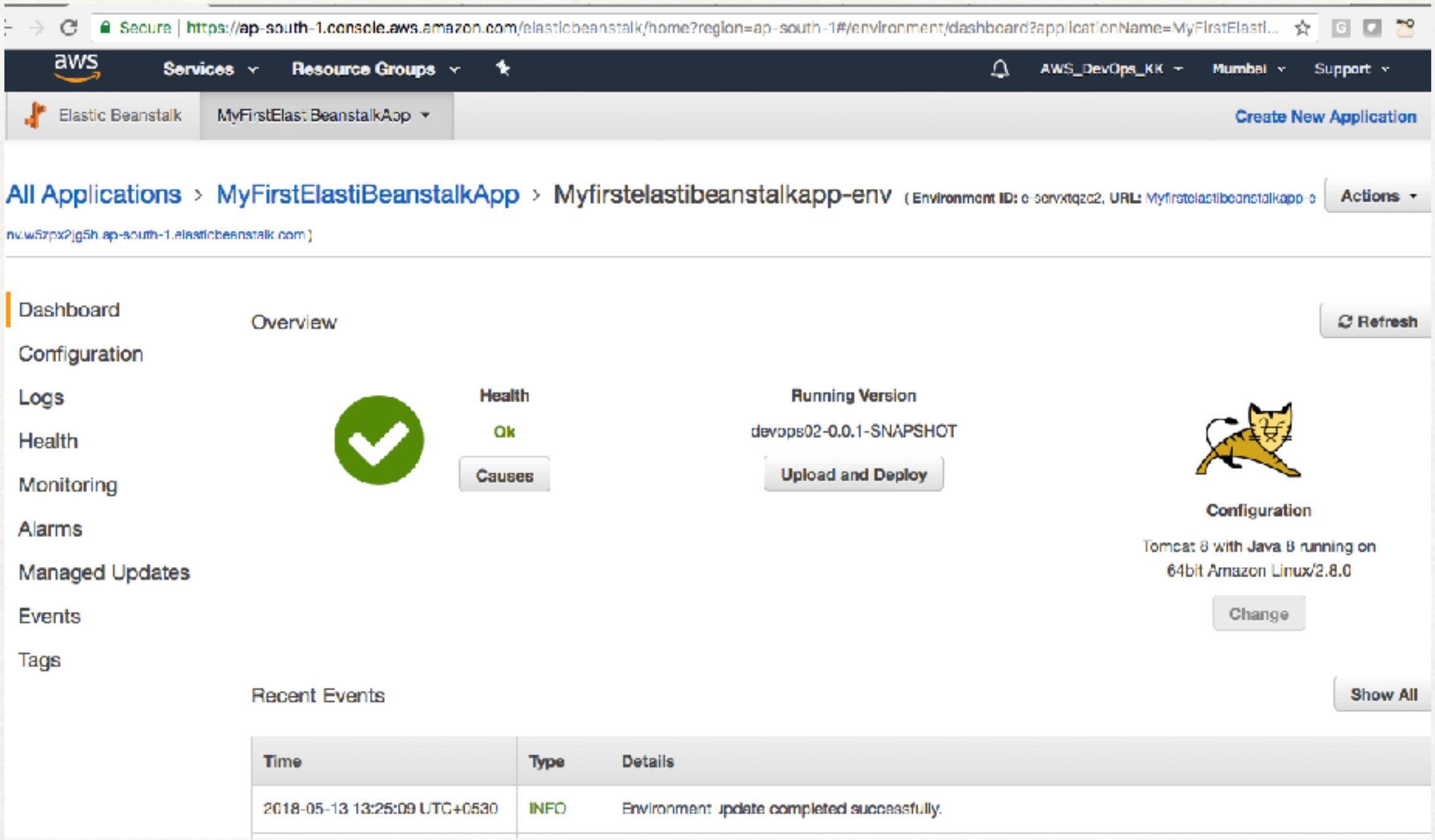
All Applications › MyFirstElastiBeanstalkApp

Environments
Application versions
Saved configurations

Settings Delete Deploy Upload Refresh

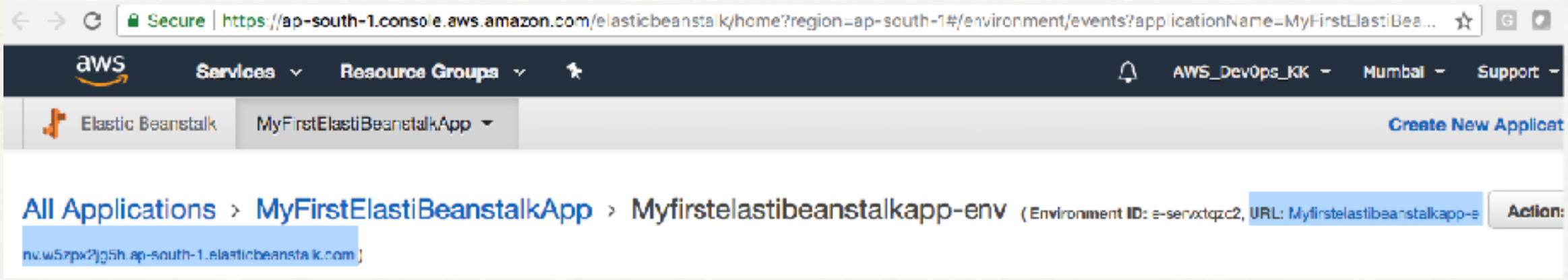
Version Label	Description	Date Created	Source	Deployed To
devops02-0.0.1-SNAPSHOT		2018-05-13 13:23:58 UTC+0530	20181330?w-devops02-0.0.1-SNAPSHOT.war	Myfirstelastibeanstalkapp-env
Sample Application		2018-05-13 12:10:07 UTC+0530	Sample Application	

STEP-18 : JAVA BUILD HAS BEEN DEPLOYED TO TOMCAT



Go to Events check the steps

COPY THE URL & GO TO BROWSER & CHECK



http://myfirstelastibeanstalkapp-env.w5zpx2jg5h.ap-south-1.elasticbeanstalk.com/

Java Application is Up!

SUMMARY & EXAM TIPS

- You can have multiple versions of your applications
- Your applications can be split in to tiers(Web Tier/Application Tier/Database Tier)
- You can update your application
- You can update your configuration
- Updates can be 1 instance at a time, a % of instances or an immutable update
- You pay for the resources that you use, but Elastic Beanstalk is free
- If elastic beantalk creates your RDS database then it will delete it when you delete your application.
- If not then the RDS instance stays
- Know what languages are supported!

- Apache Tomcat for Java applications
- Apache HTTP Server for PHP applications
- Apache HTTP Server for Python applications
- Nginx or Apache HTTP Server for Node.js applications
- Passenger or Puma for Ruby applications
- Microsoft IIS 7.5, 8.0 for .NET applications
- Java SE
- Docker
- GO