**OPERATIONS**

**MANUAL**

*ESMZone Beta*

**Education Sports Mentoring Inc.**

C:\stsdev\esmzone\web-app\images\logo.pngJune, 2010

**Revision Sheet**

|  |  |  |
| --- | --- | --- |
| **Release No.** | **Date** | **Revision Description** |
| Rev. 0 | 6/18/10 | First Draft Operations Manual |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**OPERATIONS MANUAL**

**TABLE OF CONTENTS**

Page #

1 GENERAL INFORMATION 1-1

1.1 System Overview 1-1

1.2 Project References 1-2

1.3 Authorized Use Permission 1-2

1.4 Points of Contact 1-2

1.4.1 Information 1-2

1.4.2 Coordination 1-2

Provide a list of organizations that require coordination between the project and its specific support function (e.g., installation coordination, security, etc.). Include a schedule for coordination activities. 1-2

1.4.3 Help Desk 1-2

Provide help desk information, including responsible personnel phone numbers for emergency assistance. 1-2

1.5 Organization of the Manual 1-2

Provide a list of the major sections of the Operations Manual (1.0, 2.0, 3.0, etc.) and a brief description of what is contained in each section. 1-2

1.6 Acronyms and Abbreviations 1-2

2 SYSTEM OPERATIONS OVERVIEW 2-1

2.1 System Operations 2-1

2.2 Software Inventory 2-1

2.3 Information Inventory 2-1

2.3.1 Resource Inventory 2-1

2.3.2 Report Inventory 2-1

2.4 Operational Inventory 2-1

2.5 Processing Overview 2-1

2.5.1 System Restrictions 2-2

2.5.2 Waivers of Operational Standards 2-2

2.5.3 Interfaces with Other Systems 2-2

2.6 Communications Overview 2-2

2.7 Security 2-2

3 BuILD description 3-1

3.1 Build Inventory 3-1

3.2 Build Description 3-2

3.2.1 Noble Avatar 3-2

3.2.2 ESMZone 3-3

3.2.2.1 Run Interrupt Checkpoints 3-4

3.2.2.2 Diagnostic Procedures 3-4

3.2.2.3 Error Messages 3-5

4 RUN DESCRIPTION 4-7

4.1 Run Inventory 4-7

4.2 Data Loads 4-7

4.2.1 Schools 4-7

4.3 Run Inventory 4-8

4.4 Run Description 4-9

4.4.1 Test/Production Environment 4-10

4.4.1.1 Starting the System 4-10

4.4.1.2 Run Interrupt Checkpoints 4-10

4.4.1.3 Set-Up and Diagnostic Procedures 4-10

4.4.1.4 Error Messages 4-11

4.4.1.5 Restart/Recovery Procedures 4-11

**1.0 GENERAL INFORMATION**

# GENERAL INFORMATION

## System Overview

Explain in general terms the system and the purpose for which it is intended. The description shall include:

1. Major functions performed by the system
2. Describe the architecture of the system in non-technical terms (e.g., client/server, web-based, etc.)
3. User access mode (e.g., graphical user interface)
4. Responsible organization
5. System name or title
6. System code
7. System category:
8. *Major application*: performs clearly defined functions for which there is a readily identifiable security consideration and need
9. *General support system*: provides general ADP or network support for a variety of users and applications
10. Operational status:
11. Operational
12. Under development
13. Undergoing a major modification
14. General description
15. System environment or special conditions

## Project References

Provide a list of the references that were used in preparation of this document in order of importance to the operations staff.

## Authorized Use Permission

Provide a warning regarding unauthorized usage of the system and making unauthorized copies of data, software, reports, and documents, if applicable. If waiver use or copy permissions need to be obtained, describe the process.

## Points of Contact

### Information

Provide a list of the points of organizational contact (POCs) that may be needed by the document user for informational and troubleshooting purposes. Include type of contact, contact name, department, telephone number, and e-mail address (if applicable). Points of contact may include, but are not limited to, development/maintenance POC and operations POC.

### Coordination

# Provide a list of organizations that require coordination between the project and its specific support function (e.g., installation coordination, security, etc.). Include a schedule for coordination activities.

### Help Desk

# Provide help desk information, including responsible personnel phone numbers for emergency assistance.

## Organization of the Manual

# Provide a list of the major sections of the Operations Manual (1.0, 2.0, 3.0, etc.) and a brief description of what is contained in each section.

## Acronyms and Abbreviations

Provide a list of the acronyms and abbreviations used in this document and the meaning of each.

**2.0 SYSTEM OPERATIONS OVERVIEW**

# SYSTEM OPERATIONS OVERVIEW

## System Operations

Provide a brief description of the operation of the system, including its purpose and uses.

## Software Inventory

Provide an inventory of the system’s programs. Include each program’s full name and identification, as well as the security characteristics of the software. Identify those programs necessary to continue or resume operation of the system in a degraded or an emergency situation.

## Information Inventory

Provide an overview of the information included in the system and refer to Appendix 1 of **Handbook 2229.1 Records Disposition Schedule for Automated Systems** for retention information for these inventories.

### Resource Inventory

List all permanent files and databases that are referenced, created, or updated by the system. Include the file names and database names, specific file identification, storage media and required storage capacity, as well as security considerations. Identify those files and databases necessary to continue or resume operation of the system in a degraded or an emergency situation.

### Report Inventory

List all reports produced by the system. Include the following information for each report listed:

1. Security considerations
2. Media (hard copy, electronic media)
3. Frequency of reporting
4. Typical report volume
5. Software that produces the report, if applicable

## Operational Inventory

Identify any infrastructure hardware and/or software support related to the operation of this system, including peripheral and resource requirements (e.g., network management software).

## Processing Overview

Provide information that is applicable to the processing of the system.

### System Restrictions

Identify any system restrictions imposed on this system (e.g., times of day when system can be run).

### Waivers of Operational Standards

Describe any waivers that are or will be filed to exempt the operation of the system from operational standards already followed.

### Interfaces with Other Systems

Describe operational interfaces to other systems (e.g., input data for this system comes from the same source and on the same physical media shared by another system).

## Communications Overview

Describe or depict the communications network necessary to operate the system.

## Security

Describe the security considerations associated with the system.

**3.0 BUILD DESCRIPTION**

# BuILD description

ESMZone.com is web based application built on the Grails framework. Grails is a rapid application development framework that leverages the strength of the Java Virtual Machine running in a server environment. Grails strives to take the complexity out of Java development while still leveraging robust java components. Key Java components include Hibernate for database access and Spring for the MVC framework. While Java is utilized for many of the components underling Grails, programming in Grails is done in the Groovy language. Groovy is a rapid application development alternative to Java running in the JVM. While Groovy is similar to Java it’s a dynamic language which allows runtime evaluation. This greatly improves development speed but at some cost to performance.

## Build Inventory

One prominent feature of Grails is its use of a plug-in approach to adding functionality. With many plug-ins available to the developer, functionality that previously required work to developer or integrate 3rd party class libraries not can be added with ease. Examples include search, fancy UI components, and AJAX support.

The basic building blocks of the system are Grails and Groovy

Grails.1.2.1

Groovy-1.7.0

EMSZone leverage several plugins to provide functionality and run in a Tomcat servlet container using MYSQL for the database.

Grails Plugins

plugins.bubbling=2.1.2 (AJAX extensions to YUI)

plugins.fckeditor=0.9.4 (HTML editor not currently used in the production build)

plugins.grails-ui=1.2-SNAPSHOT (UI components use primarily in the admin)

plugins.hibernate=1.2.1 (Database framework)

plugins.jquery=1.4.1.1 (AJAX alternative to YUI used by some components)

plugins.jsecurity=0.4.1 (Provides login security)

plugins.mail=0.9 (Provides support for email)

plugins.nested-set=0.1 (provides support for tag hierarchy not currently used in production)

plugins.richui=0.7 (provides some AJAX components)

plugins.searchable=0.5.5 (provides search functionality)

plugins.tomcat=1.2.1 (Support for Tomcat)

plugins.yui=2.7.0.1 (Primary AJAX library used for remote forms and other UI components)

The database used is MYSQL 5.0

The development environment is Windows Vista using the Springsource Tool Suite (STS). STS is a customized version of Eclipse created by Springsource to support Java development using the Spring framework and Grails.

Noble Avatar

The generate avatar feature of ESMZone is provided by software from Noble Games. Noble provided java applet code that was modified to work with ESMZone. This java applet runs in the user’s browser and allows the user to create and upload an avatar to the server. The avatar is build using JDK 1.5 to give it compatibility with users that have JRE 1.5 or higher on the PC. Noble avatar software leverages a library file which provides user interface components (jgoodies-forms-1.0.4.jar). When running applets in a browser, the applet must be signed for the security features of many browsers to allow the applet to run. Ideally, the applet should be signed by a recognized authority so the browser can indicate this to the user. When signing noble avatar, you must sign both the avatar applet (avatar.jar) and the interface library (jgoodies-forms-1.0.4.jar) for the browser to acknowledge the applet is fully signed.

## Build Description

To build and test locally in a development environment, download STS from springsource. Add the plugins for Grails and Groovy development. Add in the subversion plugin to connection to the Codesion repository.

SprintSource Tool Suite 2.3.0.RELEASE

Plugins:

Grails 1.2.1

Groovy-Eclipse Feature 2.00

Subclipse 1.6.6

Install MYSQL 5.0

### Noble Avatar

The Noble Avatar applet is build using the Eclipse IDE in this project. The source code is stored in Codesion. If the code needs to be changed follow the direction below to recompile, sign and copy to the ESMZone project. Normally, the applet will not change and the latest copy (jar files) will be stored in the ESMZone code repository ready to be packaged and released along with ESMZone.jar.

Repository Root: <https://esmzone.svn.cvsdude.com/avatar/trunk>

1. Checkout the project from the repository root.
2. Build the applet
3. The built classes will be created in the bin directory under the avatar source code directory.
4. Package the class in a jar file using the command:
   1. Change directory to the bin directory under avatar
   2. jar cvf avatar.jar \*
   3. Copy avatar.jar to the esmzone/web-app directory
5. Sign the avatar.jar
   1. First you must setup a development keyfile or get a production file from a signing authority like Thawte.
   2. To generate a development key, go to the esmzone/web-app directory execute the following command
      1. keytool –genkeypair –alias signFiles –keystore esmzone
      2. Enter a keystore password
      3. Complete the information about your organization

Last Name: Moss

FirstName: Andy

Organization Unit: ESMZone

Organization: ESM Inc

Locality: Norwalk

State: CT

Country: US

* + 1. Enter a password for signFiles
  1. Sign the jar file
     1. jarsigner -keystore esmzone avatar.jar signFiles
  2. If not already done, sign the jgoodies jar file in the same way
     1. jarsigner -keystore esmzone jgoodies-forms-1.0.4.jar signFiles

### ESMZone

ESMzone is a standard Grails 1.2 application. The build can be done either in the STS IDE or from the command line (ex. for test and production environments). Since the commands are essentially the same this description will focus on the command line. The product of the build is generally a “war” file that can be deployed on the Tomcat server. Deployment is simply copying the new esmzone.war to the web-apps directory under Tomcat.

Test and production have identical directory structures and build procedures. Make sure the latest version is on the Trunk branch of SVN (Subversion).

Directory Structure

All data and code are stored on an Amazon EBS volume and not on the AMI. This means that the EC2 instance can be started and stopped without loss of the database or application data.

/ebs1 : the EBS volume is mounted at this location

/ebls1/opt : all 3rd party software is in this subdirectory

/ebs1/TcatServer6 : the version of Tomcat (customized by Mulesoft) that runs test and production

/ebs1/TcatServer6/bin : the directory where the startup and stop scripts for Tomcat are

/ebs1/TcatServer6/webapps: the directory where the application esmzone.war is placed

/ebs1/TcatServer6/logs: the directory where Tomcat system logs are created and can be viewed.

/ebs1/www/esmzone: The directory where source code is kept, checked out to and built.

Build Steps

* Insure the latest code is checked into SVN.
* Connect to the desired server (test or production) using your SSH client as the root user.

Using SecureCRT 6.5.0 (Van Dyke Software), you setup a connection to either [www.esmzone.com](http://www.esmzone.com) (production) or test13.esmzone.com (test). Be sure to assign the public key file to the connection. Under SSH2 settings, highlight public key in the Authentication box. Click the properties button and select “esmzone.pem” from the directory where you stored the public key (provided to you by the system administrator as generated by your Amazon EC2 account).

* Change directory to esmzone: *cd /ebs1/www/esmzone*
* Checkout the latest code: *svn update*
* Build the new war: *grails prod war*

Deployment Steps

* Connect to the desired server as root
* Login as the tomcat user: *su – tomcat*
* Go to the web-app directory: *cd ../webapps*
* Run the update script to copy the war: *./* *updateEsmzone.sh*
* The system will start tailing the log (/logs/cataling.out). You should see:

INFO: Deploying web application archive esmzone.war

when the war starts being deployed

* When the war is fully deployed you will see the following debug information

driverClassName=com.mysql.jdbc.Driver

minEvictableIdleTimeMillis=1800000

maxOpenPreparedStatements=-1

removeAbandoned=false

minIdle=0

defaultReadOnly=false

initialSize=0

timeBetweenEvictionRunsMillis=1800000

maxIdle=8

maxWait=-1

testOnBorrow=true

username=esmzone

url=jdbc:mysql://localhost/esmzone

numActive=0

defaultTransactionIsolation=-1

maxActive=8

logWriter=java.io.PrintWriter@ebf5ae

accessToUnderlyingConnectionAllowed=false

password=esmzone13

testWhileIdle=false

connection=jdbc:mysql://localhost/esmzone, UserName=esmzone@localhost, MySQL-AB JDBC Driver

class=class org.apache.commons.dbcp.BasicDataSource

testOnReturn=false

validationQuery=SELECT 1

numTestsPerEvictionRun=3

poolPreparedStatements=false

logAbandoned=false

numIdle=6

defaultAutoCommit=true

removeAbandonedTimeout=300

defaultCatalog=nul

#### Run Interrupt Checkpoints

Identify and describe the acceptable interrupt points within the system to permit manual or semiautomatic verification of intermediate results, to provide the user with intermediate results for other purposes, or to permit a logical break if higher priority jobs are submitted.

#### Diagnostic Procedures

Verify that the application is running.

From a new browser instancer bring up the application:

<http://www.esmzone.com/esmzone>

Check the Memory Usage

TCatServer6 comes with a console. Run the console and check the memory usage. Currenlty the console runs better on Firefox. If you want to run it on IE you must be in compatibility mode.

<http://www.esmzone.com/console>

Login as admin

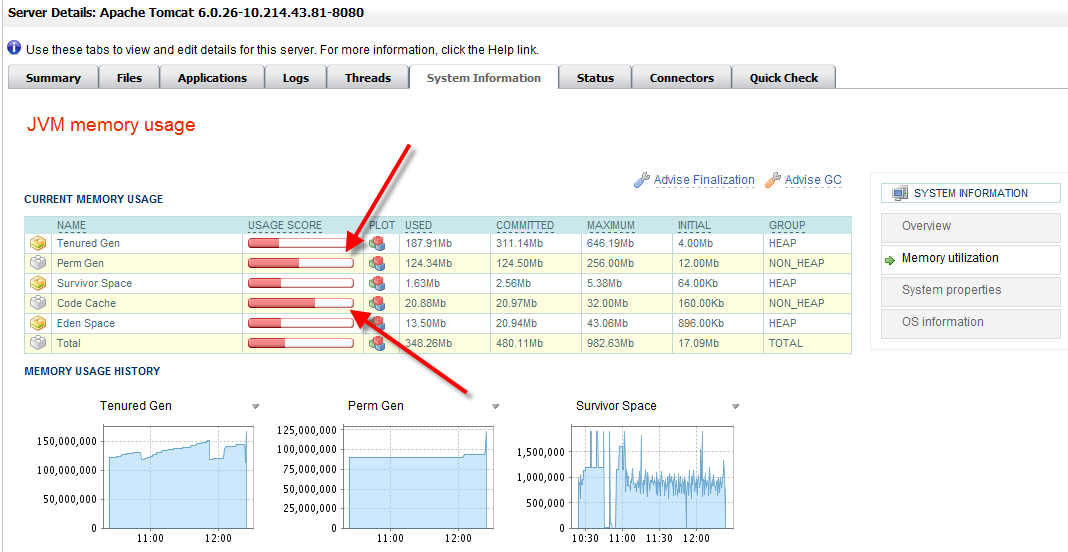
Select the server from the list (it should be the only one)

Click on the System Information Tab at the top.

Click on the Memory Utilization Tab on the right

Look at the Perm Gen and Code Cache

If either of these has a usage score bar nearing 80+ % full, you should restart the server.



Check the Latest Release of Esmzone

Make sure the code released is the latest release. If you run out of memory during a release, the latest version may not get deployed. Restart the Tomcat server and re-run the updateEsmzone.sh script in the webapps directory as described in the deployment section.

#### Error Messages

The primary error message you are likely to see in the Cataling.out log is a memory error. If this occurs restart the server.

**java.lang.OutOfMemoryError: PermGen space**

**4.0 RUN DESCRIPTION**

# RUN DESCRIPTION

## Run Inventory

The system is running on a custom AMI that is based on a standard Amazon AMI for Fedora Linux Release 8.

Mysql 5.0

Mulesoft TCatServer6 (which comes with Suns JVM 1.6.0)

Grails 1.2.1

Java Development Kit (JDK) 1.7.0 IcedTea Runtime Environment for Fedora (build 1.7.0-b21)

## Data Loads

### Schools

In order to load school data, the system must populate both the school and address tables. The address is used to display the city and state of schools along with their name to help difference generically named schools. In order to load school data, you must populate a table called school\_load with the data necessary to populate the tables.

School Load Table

* id
* version
* name
* street
* state
* city
* postal\_code
* type

The school type must be one of the following (PRESCHOOL, ELEMENTARY, MIDDLE, JUNIOR\_HIGH, HIGH, VOCATIONAL, COLLEGE, GRADUATE, OTHER).

Type: If you have a list of schools but not the specific details you can just use “OTHER” for all schools. Currently the type is not used in the system but there for future use.

Version: This is a system field and should always be loaded with 0.

All other fields are required. If you do not have data for street or postal\_code you can put default values in (ex. street = “NA”) . Only the Name, City and State are used in the system.

Load Steeps

1. Setup a CSV file with the required fields (all but id which is populated by the system).
2. Start the esmzone system in the environment you wish to load the data into. You may load new schools into.
3. Clear out the current school\_load table (delete \* from school\_load)
4. Import the CSV file of schools into the school\_load table using a tool like DbVisualizer.
5. Login to ESMZone as the admin on the system you are loading.
6. Run the following command from a browser against the system you are loading:

<http://URL/esmzone/admin/schooolLoad/load?max=NN>

where URL is the system you are loading (ex. [www.esmzone.com](http://www.esmzone.com)) and NN is the number of schools you want to load. Currently 10,000 is the maximum you can load in a single run. If you have more than 10,000 simply split the load data into two CVSs and repeat the process again.

## Run Inventory

ESMZone is a Grails 1.2 application running on the JVM under TCatServer6 (a version of Tomcat). All hosting is done on Amazon EC2. There are currently two servers, one for test and one for production.

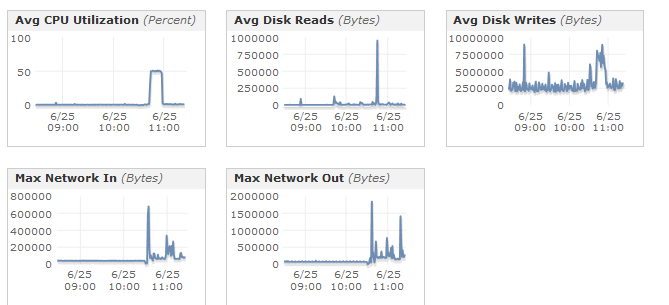
The Amazon account is under [admin@esmzone.com](mailto:admin@esmzone.com)

* Test Server: test13.esmzone.com is bound to the Elastic IP (184.73.239.80) and a 10 Gig Elastic Block Storage - EBS (vol-f7bf6b9e). It is running on a small EC2 instance in the us-east-1d availability zone.
* Production: [www.esmzone.com](http://www.esmzone.com) is bound to the Elastic IP (184.73.186.120) and a 50 Gig Elastic Block Storage (vol-ddbd2cb4). It is running on a medium EC2 instance.

Note: The instance and the EBS must be in the same availability zone.

Monitoring

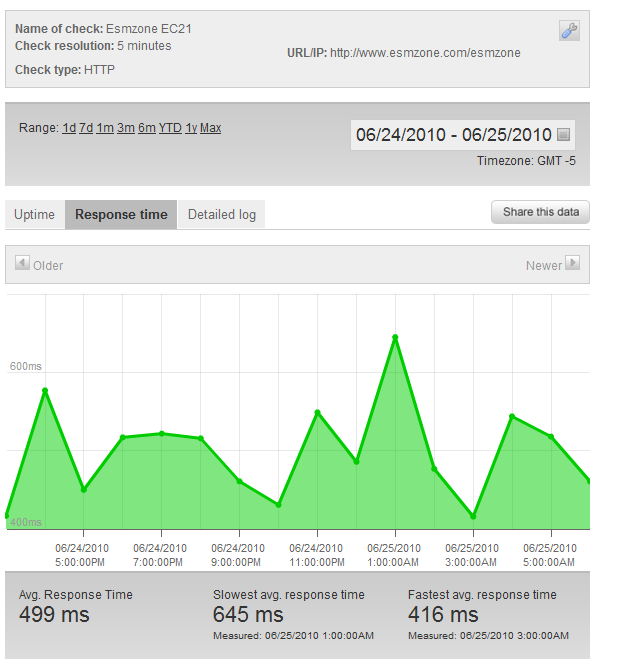
The production instance has an extra service CloudWatch enabled which monitors the health of the server



In addition a uptime monitoring servers has been setup to notify if the server is not responding. The service is called Pingdom and a free account has been setup under [dbrandt@precidix.com](mailto:dbrandt@precidix.com)/Test13. This account can monitor a single server and send emails or alerts to an iPhone app of outages. It also maintains and resports uptime statistics. Checks are done every 5 minutes.

The free account also include 20 SMS messages, however, due to this limitation, email and the Iphone app are being used for notification. (www.pingdom.com)

Currently on the email notification list are [andy@esmzone.com](mailto:andy@esmzone.com), [rick@esmzone.com](mailto:rick@esmzone.com) and [dbrandt@precidix.com](mailto:dbrandt@precidix.com).



## Run Description

### Test/Production Environment

The runtime container currently used in Test and Production is Mulesofts TCatServer6. This is a customized version of Tomcat that supports Mulesofts monitoring and clustering add-ons. We currently use their Console to monitor memory usage.

#### Starting the System

Steps to Run

1. To run the application, connect to the desired instance as described in section 3.2.2.
2. Login to the tomcat user: *su – tomcat*
3. Change directory to the TcatServer6/bin directory: *cd ../bin*
4. Check if the server is already running: ps –ef | grep java
   1. If the server is running you should see the following:

tomcat 5895 1 1 Jun16 ? 02:07:37 /ebs1/opt/TcatServer6/jre/bin/java -Djava.util.logging.config.file=/ebs1/opt/TcatServer6/conf/logging.properties -Xmx700m -XX:MaxPermSize=256m -XX:+HeapDumpOnOutOfMemoryError -Dcom.sun.management.jmxremote -Djava.net.preferIPv4Stack=true -Djava.awt.headless=true -Dgalaxy.data=/ebs1/opt/TcatServer6/galaxy-data -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djava.endorsed.dirs=/ebs1/opt/TcatServer6/endorsed -classpath /ebs1/opt/TcatServer6/bin/bootstrap.jar -Dcatalina.base=/ebs1/opt/TcatServer6 -Dcatalina.home=/ebs1/opt/TcatServer6 -Djava.io.tmpdir=/ebs1/opt/TcatServer6/temp org.apache.catalina.startup.Bootstrap start

* 1. If the server is not running you should not see any processes.

1. If you wish to restart the server (for example if memory is low), execute the shutdown command: ./shutdown.sh
2. Check to see the server has completely shutdown by issues the command in step 4 above until no processes show up.
3. Startup the server with the startup command: ./startup.sh
4. Check the server has completed starting up. Change to the log directory and tail the log:
   1. *cd ../logs*
   2. *tail –f catalina.out*
5. Verify the server is completed the startup. It will load esmzone.war and display the output shown in section 3.2.2. Next it will run the console.war. When the startup is completed the following output will display:

INFO: Server startup in 99373 ms

#### Run Interrupt Checkpoints

Identify and describe the acceptable interrupt points within the system to permit manual or semiautomatic verification of intermediate results, to provide the user with intermediate results for other purposes, or to permit a logical break if higher priority jobs are submitted.

#### Set-Up and Diagnostic Procedures

#### Error Messages

#### Restart/Recovery Procedures

In the event that the EC2 instance crashes or become unresponsive, you can start a new instance, associate the Elastic Block Store volume and the Elastic IP to a new instance. These steps can be done from the Amazon management console on the web. Once started, you will need to login to the instance and mount the Elastic Block Store to the file system. After this, you can following the steps above in the run description.

The current production AMI (instance to start) is ami-f6f41c9f (esmzone/BetaRelease.manifest.xml).

The production EBS volume is vol-ddbd2cb4

The production Elastic IP is i-c6b5a6ad (184.73.186.120)

Steps to start a new Instance

1. From the Amazone management console, select Instances from the menu on the left
2. Select Launch Instance from the button above the list of instances.
3. Select My AMIs and choose ami-f6f41c9f (esmzone/BetaRelease.manifest.xml)
4. If this is production chose the us-east-1a availability zone (because the EBS volume is in that zone). If this is test choose the us-east-1d availability zone (the EBS volume for test is in that zone)
5. Choose instance type small for test and High-CPU Medium for production.
6. Leave the other defaults and then click continue.
7. Check enable CloudWatch monitoring if this is a production instance and click continue.
8. Select the “esmzone” keypair for security and click continue.
9. Leave the default security group selected and click continue.
10. Verify all the setting are correct and click Launch.
11. It will take some time to complete the launch.
12. Once that is done the instance status will indicate “Running”. Note the instance Id for the next steps.
13. Next you must associate the EBS and Elastic IP
14. Click on the Volumes option to the right.
15. Verify the volume you will use (vol-f7bf6b9e – 10G for test, vol-ddbd2cb4 – 50G for production) are not in-use. If they are then the instance may still be running and must be stopped from the instance list. This can be done by right clicking on the instance and selecting terminate.
16. Right click on the volume and select attach. Attach the volume to the instance you just started (you will need to select the instance id).
17. Once the volume is attached it will indicate in-use.
18. Next attach the Elastic IP in the same way. Goto the elastic ip section and right click on the appropriate IP. Select associate IP and pick the appropriate instance id.
19. Next use your SSH client to attach to that instance. This should already be setup since the IP has not changed. See section 3.2.2 for SSH connection.
20. Logged in as root, mount the EBS volume to the appropriate mount points in the system.

*mount /var/lib/mysql*

*mount /ebs1*

1. Stop apache if it is running. a*pachectl stop (note this can be run regardless to insure apache is stopped)*
2. Start MYSQL to insure it is using the new mount points: */etc/init.d/mysqld restart*
3. Verify apache is not running by attempting to get the server from your browser (ex. http://www.esmzone.com). You should get a 404 (site not found).
4. Verify that MYSql is using the correct database: *mysqld esmzone –uesmzone –pesmzone13*
5. You should see the MYSQL prompt:

Welcome to the MySQL monitor. Commands end with ; or \g.

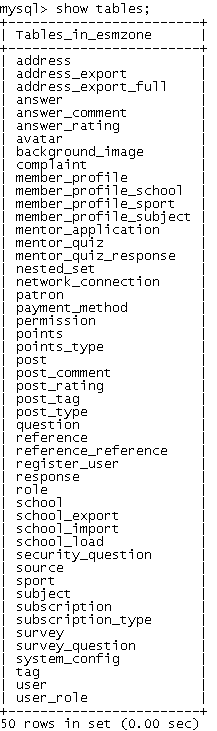
Your MySQL connection id is 2188

Server version: 5.0.45 Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>

1. To verify you are in the correct database you can check the table list: *show tables; (note the semicolon)*



1. Refer to section 4.4.1 for string the Esmzone system.