# Project Setup

## Project E-mail Account

In order to use external services such as Amazon AWS we made a project account that should be passed along to future developers of this project.

Address: [molecularMission@gmail.com](mailto:molecularMission@gmail.com)

Password: RiTSeniorProject

In addition, as of the 2015-2016 team, the AWS account uses the same email and password for the sake of convenience. Please feel free to change this if you choose to do so.

**Important Note:** The Amazon AWS account and free tier usage was set up in early May 2016, and the Amazon’s free tier is only free for 1 year. The 2016-2017 senior project team will need to decide if it is best to stick with Amazon and have the sponsor pay for usage or use some other means to host the server and game.

## Game

Git Repository: <https://kgcoe-git.rit.edu/theriostopes/risotopes-U5> (This is a private repository hosted through the College of Engineering. You will need to log in using your main RIT accounts and have either Prof Vallino, or Prof Lanzafame(the sponsor) add you to the repo.)

Download and install the latest version of the Unity game engine.

Note: You will need to make a free Unity account in order to access the download link.

Clone the game’s Git repository and open the project’s root folder in Unity. You should have access to everything in the game including scenes, scripts, models, etc.

## Server

Git Repository: <https://kgcoe-git.rit.edu/theriostopes/student-manager> (This is a private repository hosted through the College of Engineering. You will need to log in using your main RIT accounts and have either Prof Vallino, or Prof Lanzafame(the sponsor) add you to the repo.)

Clone the server’s Git repository and follow the instructions in the README.md file for setting up the server. Downloading and using Vagrant and virtualbox is highly recommended for this project as Vagrant makes it easy to deploy to Amazon AWS.

Note: It is recommended that the development machine for the server is Unix based (Mac OS X or Linux) since all of the prior development of the server has been done in a Unix command line environment. If this is no longer feasible please feel free to spend some time to setup a working development environment and update the server documentation for working on non-Unix based machine.

# Project Deployment

Both the game and the server are deployed on an Amazon EC2 instance (See the [Project E-mail Account](#h.k9qqrxvsq3h) for AWS account info). Therefore, this enables us to not only host the website client, but also host the game executable files since Amazon EC2 provides 30GB of space in the free tier at the time of this writing.

## Preparing the Game

1. Make new directories called ‘Molecular\_Mission\_<OS>’ where <OS> is either Mac, Windows, or Linux.
2. Next, build the game for Windows, Mac, and Linux using a universal processor architecture (Universal or x86) and place the necessary build files in the corresponding folder that was just created.
3. Compress those three folders as a .zip and move the zip files to ‘<server\_root>/public/views/game/game\_executable’, where <server\_root> is the root directory for the server side source code.

## Preparing for Amazon Deployment

1. Create new users in the Identity and Access Management section of the AWS console for the people that you want to have direct access to the AWS instance.
   1. Note: Make sure to generate access keys for each user, and make sure you can access the access key and secret key later on. (When I did this I was able to download a .csv file that had this information, so I recommend downloading it as this information is needed later.)
2. In the EC2 console click on Key Pairs under Network and Security and generate a new key pair.
   1. Note: Be sure to save the .pem file as it is used when accessing the Amazon instance later on.
3. Also under Network and Security create a new security group and set the inbound ports for SSH, HTTP, and HTTPS as well as open up ports 8080 and 27017.
   1. Note: You might be able to use the security group that is already configured.

## Deploying through Vagrant

1. Once Vagrant has been set up on the development machine run the following command line commands.
   1. vagrant plugin install vagrant-aws
   2. vagrant box add dummy <https://github.com/mitchellh/vagrant-aws/raw/master/dummy.box>
2. After that, create environment variables for:
   1. AWS\_KEY - Your IAM user’s access key
   2. AWS\_SECRET - Your IAM user’s secret key
   3. AWS\_KEYNAME - The name of the Key Pair made in EC2
   4. AWS\_KEYPATH - The file path to the .pem file
   5. Note: If you are not sure how to create environment variables look up the command to do so. Also, these are necessary so that AWS credentals stay out of the Vagrantfile that is in the server repository.
3. If something changes in terms of the AWS API or the Vagrant-AWS plugin then modify the Vagrantfile in the server repo as necessary.
4. Start up an AWS instance with: vagrant up --provider=aws
5. To ssh into the instance use: vagrant ssh
6. To terminate the instance (so that it is no longer availble or used for billing purposes) use: vagrant destroy

**Important Note:** If an instance is running and you only need to update the game files stored in the instance use the following SCP command to update those files:

scp -r -i <AWS\_KEYPATH> <LOCAL\_PATH\_TO\_GAME\_ZIP> <AWS\_PATH\_TO\_GAME\_ZIP>

With that said, you WILL need to SCP the game executables over to the Amazon instance so that the game can build with the correct URL to the instance (it is located in the game’s ConnectionUtility.cs). It is recommended that you create scripts that will handle the SCP commands for you once you build the game provided that you have already have an instance running on Amazon.

# Amazon Billing

This section is for mainly for Joe so that he can monitor Amazon’s billing.

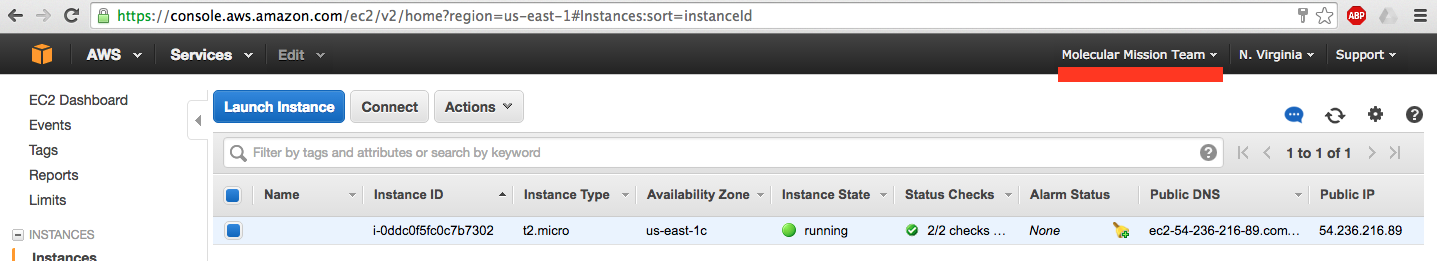
Amazon Login Page:

<https://console.aws.amazon.com/console/home>

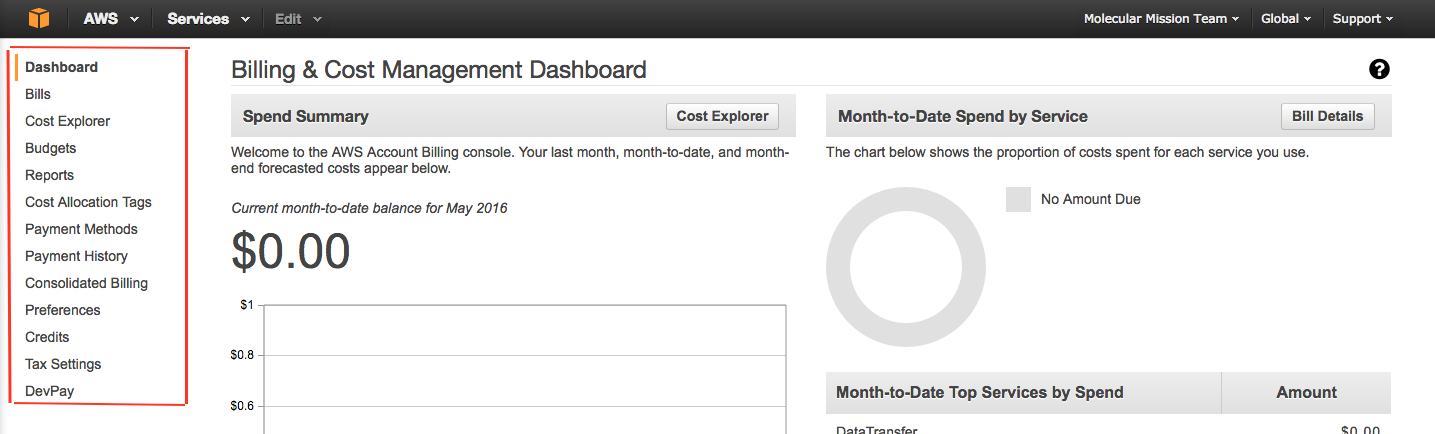
Use the credentials listed in the [Project E-mail Account](#h.k9qqrxvsq3h) section.

## To view billing information

1. Click on ‘Molecular Mission Team’ as indicated by the red line in the Screenshot 1 below.
2. Click on ‘Billing and Cost Management’
3. Navigate to specific billing related pages using the navigation pane on the left, as highlighted by Screenshot 2.



Screenshot 1 - AWS page when logged in



Screenshot 2 - Billing Dashboard and Navigation