Content Delivery Web Server

The objective of this project is to determine the best method to setup a content delivery web server. The content delivery web server is to [*fill in with intended use*].

**Q: What are our current needs?**

A: These are the following requirements for the system:

* Client authentication
* SSH terminal access
* Demo environment
* Production environment

**Q: How are we going to manage our code?**

A: All code will be managed through a Version Control System (Git) which will sync with the IDE that we will be using (Eclipse or AWS Cloud9).

**Q: What will the difference be between the sandboxes versus production?**

A: AWS Elastic Beanstalk allows for the creation of multiple environments when multiple versions of an application need to be run. Environments could be created for development, integration, and production.

**Q: How do we host a sandbox?**

A: Amazon EC2 will be used to create and run virtual machines in the cloud.

<https://aws.amazon.com/getting-started/tutorials/launch-windows-vm/>

**Q: How will this system be integrated with the current WF file system and terminal access?**

A: The Amazon EC2 instance can connect to the Remote Desktop Protocol to access all files. Amazon EC2 can be connected to using SSH.

**Q: What frameworks and packages are we using?**

A: This is what we are currently using:

OS: Linux, Ubuntu

HTTP: Apache HTTP Server

Databases: SqlServer

Languages: JS, PHP

File Transfer Protocol: WinSCP

**WIP Architecture**

This section is a high-level overview of what the framework for the demo-server will be. Please see below for details on how the technology works.

1. Git will manage the code that is written.
2. Django (Python) will be used as the backend setup for user-logins, user management along with a set of templates to display HTML titles. NodeJS will be used to add real-time functionality.
3. AWS Elastic Beanstalk will be used to deploy the web applications.
   1. AWS Elastic Beanstalk can run multiple versions of the same environment for testing and production. The following will be set up by AWS EB.
      1. AWS EC2 will be used for computing (Virtual Machines).
      2. AWS VPC will be used for security and isolation of web apps.
      3. AWS RDS will be used as a database
      4. AWS S3 will be used to store data

Version Control System (VCS)

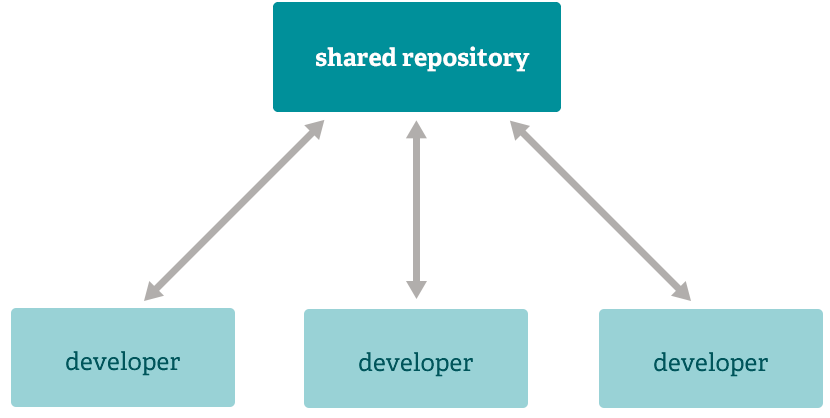
**Git**

Git is a distributed open-source version control system that stores code, tracks revision history, merges code changes, and reverts to earlier code versions.

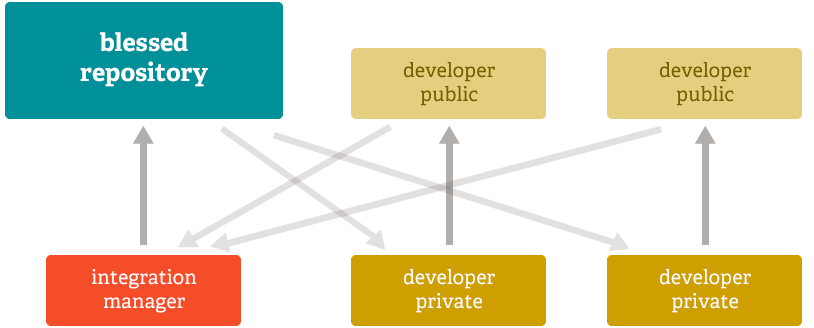
* Stores source code and full development history locally in a repository
  + Create a branch (copy of source code) that can be worked on in parallel to the main version
  + Commit changes to save progress or merge your branch back into the main version
  + Every commit will be stored in a snapshot and compared to previous versions with a viewable operation (diff)

Features:

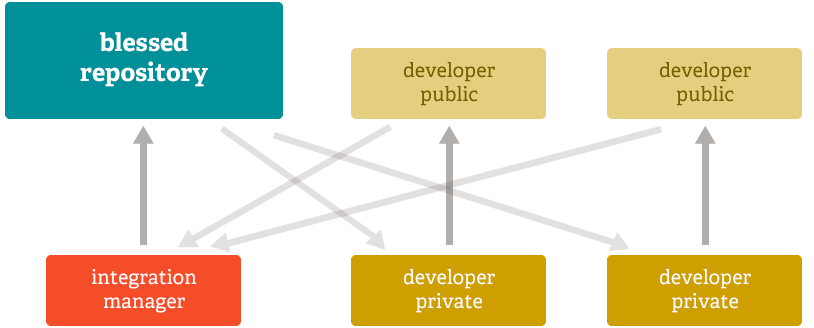
* Branching and merging
  + Frictionless context switching – ideal for testing ideas, applying patching and merging
  + Role-based codelines – branch for production and another for testing
  + Feature based workflow – new branches for each new feature that can be switched on/off, delete, or merge to main branch
  + Disposable experimentation
* Small and fast
  + All operations performed locally (no communication to centralized server)
* Distributed
  + Multiple backups – no single point of failure unless only single copy of repository
  + Set-up allows for different workflows
    - Subversion-style workflow



* + - Integrated manager workflow



* + - Dictator workflow



* Data assurance
  + Data model ensures cryptographic integrity of every file and commit
* Staging area
  + Intermediate area where commits can be formatted and reviewed before completion
  + Choose between adding portions of modified file or add all changes to staging area

*More research will be done when the need for this service arises.*

Bitbucket

Github

AWS CodeCommit

AWS CodeCommit is a highly scalable, managed source control service that hosts private Git repositories. *This service is not needed unless the developer team grows and the Git repositories need to be highly scalable.*

Features:

* Collaboration, encryption, access control, high availability & durability, unlimited repositories, easy access & integration, notifications & custom scripts

Content Delivery (Platform as a Service)

**AWS Elastic Beanstalk**

AWS Elastic Beanstalk is a service for deploying and scaling web applications and services. Deployment will be automatically handled with capacity provisioning, load balancing, auto-scaling, and application health monitoring).

* Application can be deployed through AWS Management Console, Git repository, or IDE (AWS Toolkit for Eclipse allows for management in the IDE)
* Developers retain full control over AWS resources powering application
  + Select OS requirements, available database, availability zone
* Supports multiple running environments that is independently configured and runs on their own AWS resources
  + Could be for integration testing, pre-production, and one for production
  + New environment could be created for each Git branch
* Database and Storage
  + Amazon S3 bucket will be created and all files will be copied from local client to S3


        Elastic Beanstalk Architecture Diagram
      

\*USED WITHOUT CLOUDFRONT DISTRIBUTION

[AWS Elastic Beanstalk + Amazon Relational Database Service (RDS)](https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/using-features.managing.db.html?icmpid=docs_elasticbeanstalk_console)

* Add database instance to Elastic Beanstalk environment
  + MySQL, PostgreSQL, Oracle, or SQL Server database during or after environment creation
* Database instance can be launched outside environment for production environment
  + Connect to database from multiple environments and tear down environment without affecting the database instance
* Can be used with Microsoft SQL Server, Oracle, or other relational databases running on EC2

[AWS Elastic Beanstalk + Amazon Virtual Private Cloud](https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/vpc.html?icmpid=docs_elasticbeanstalk_console)

* Define virtual network in isolated section of the AWS cloud
* New class of web applications can be deployed on Elastic Beanstalk
  + Internal web applications (ex. Recruiting application)
  + Web applications connecting to on-premises database (using VPN connection)
  + Private web service back ends
* Instances must have access to internet directly or through NAT device


      Elastic Beanstalk and VPC Topology with Amazon RDS
    

Deploying EB application with Amazon RDS in a VPC using a NAT gateway

Web Frameworks w/ Authentication Tools

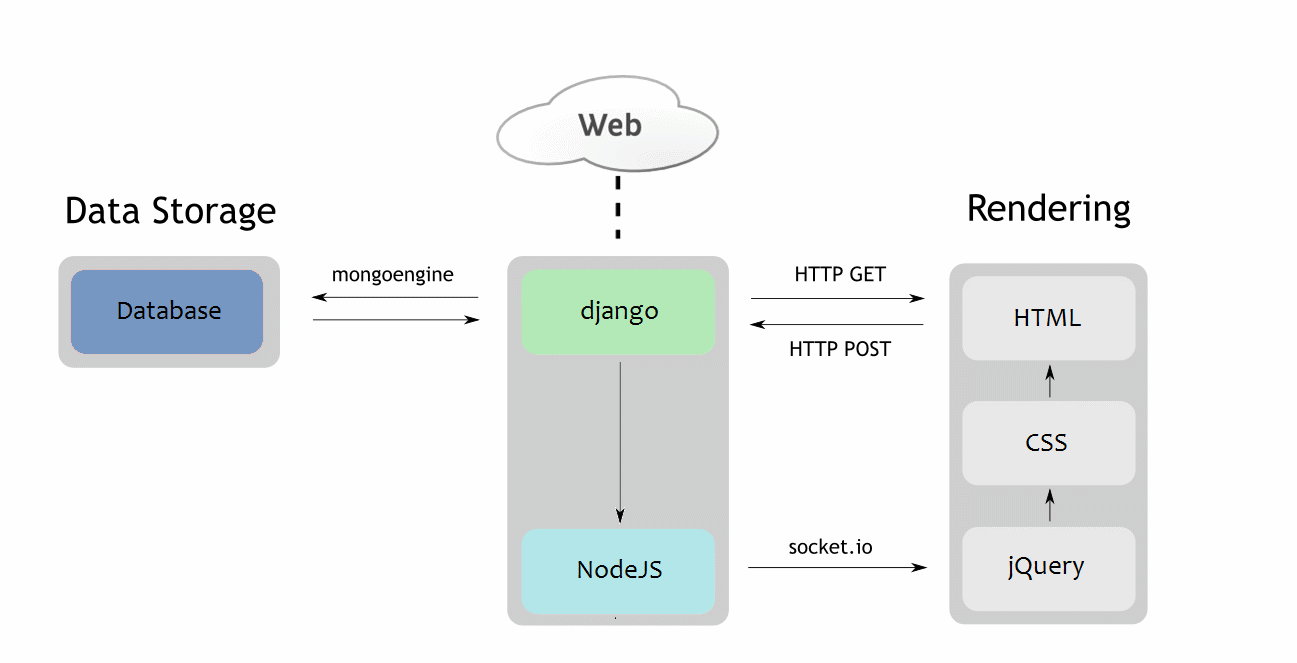
**Django**

High-level Python web framework.

* Take Web applications from concept to launch (Python code)
  + Features such as user authentication, content administration, site maps, RSS feeds
* User authentication – verifies who a use claims to be
* User authorization – determines what an authenticated user is allowed to do
  + Users
  + Permissions w/ binary flags to designate what tasks users can do
  + Groups w/ generic application of labels and permissions to users
  + Configurable password hashing
  + Forms and view tools for logging in users or restricting content
  + Strength checking
  + Throttling of login attempts
  + Authentication through third-parties (OAuth)

[Django + NodeJS (+ Socket.io + Redis)](http://www.cuelogic.com/blog/how-to-use-both-django-nodejs-as-backend-for-your-application/)

* Django will be used to handle maintenance functionality such as user-logins, user management and a set of templates to display titles
* NodeJS will build the real time data retrieval
  + Server part of library runs on top of NodeJS which will provide high performance event-driven framework to manage message exchange with client
* Redis will be used as message bus with architect socket-io server to send messages between Django and NodeJS
  + Socket.io enables real-time bidirectional event-based communication
  + Redis is an in-memory database structure store



Passport JS

* Authentication middleware for Node
  + 300+ authentication strategies
  + Easily handle success and failure
  + Dynamic scope and permissions
* Used in any Express.js based web application

AWS SDK for JS in Node.js (STS class)

Integrated Development Environment (IDE)

Eclipse w/ AWS Toolkit

**AWS Cloud9**

*Not hosted in Canada so will not be used until it is.*

AWS Cloud9 is a cloud IDE for writing, running, and debugging code.

* Terminal access to AWS
* Code through a browser – no setup required
  + Create developer environment on new Amazon EC2 instance or connect it to your own Linux server through SSH
  + Fast start on projects – packaged with tooling for 40+ programming languages
  + Easy collaboration on code
* Build serverless applications – switch between local and remote execution of code
* Connect and run in remote server (SSH) requires ability to reach existing instance or server over public internet using SSH

[AWS Cloud9 + Amazon Lightsail](https://docs.aws.amazon.com/cloud9/latest/user-guide/lightsail-instances.html)

* Launch new Amazon EC2 instance preconfigured with:
  + Popular app or framework – WordPress, MySQL, PHP, Node.js, Nginx, Drupal, Joomla
  + Linus distribution – Ubuntu, Debian, FreeBSD, openSUSE
* Create environment, connect to new instance, open IDE

Databases

**Amazon Relational Database Service**

Amazon RDS is a managed relational database service that uses familiar database engines and handles routine database tasks such as provisioning, patching, backup, recovery, failure detection, and repair.

* For use with MySQL, Microsoft SQL Server, PostgreSQL, MariaDB, Oracle, and Amazon Aurora

PostgreSQL

MySQL

Amazon Aurora

MongoDB

Integrated Services (used with Amazon Elastic Beanstalk)

**Amazon Elastic Compute Cloud**

Amazon EC2 is a web service that provides secure, resizable compute capacity in the cloud.

* Launch instances with variety of operating systems, load with custom application environment, manage access permission and run with as many or few systems as needed
* Instances are virtual machines created and run on the cloud
  + Specify software and specifications of the instance
  + Choose instance type (CPU, memory, storage and networking capacity)
  + Connect to instance using a Remote Desktop Protocol client
    - Instance can be connected through SSH <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AccessingInstancesLinux.html>

**Amazon Simple Storage Service**

Amazon S3 is object storage built to store and retrieve any amount of data from anywhere (web sites and mobile apps).

* Designed to deliver 99.999999999% durability and provide comprehensive security and compliance capabilities that meet regulatory requirements
  + Distribute data across three physical facilities (minimum) with data replicated to any region
  + Supports 3 types of encryption
* Build applications that make use of Internet storage

**Amazon Virtual Private Cloud**

Amazon VPC lets you provision a logically isolated section of the AWS cloud to launch AWS resources in a custom defined virtual network.

* Grants complete control over virtual networking environment
  + Selection of own IP address ranges, creation of subnets, and configuration of route tables and network gateways
  + Could have hardware VPN connection between current datacenter and VPC and leverage the AWS cloud as extension of datacenter
  + Define own network space and control how network and Amazon EC2 resources are exposed to the internet

Pricing

“AWS offers you a pay-as-you-go approach for pricing for over 70 cloud services. With AWS you pay only for the individual services you need, for as long as you use them, and without requiring long-term contracts or complex licensing.”

EC2 – On-Demand Pricing (Canada Central region)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Instance | vCPU | CPU Credits /h | Mem (GiB) | Price |
| T2.nano | 1 | 3 | 0.5 | $0.0064 / h |
| T2.micro | 1 | 6 | 1 | $0.0128 / h |
| T2.small | 1 | 12 | 2 | $0.0256 / h |
| T2.medium | 2 | 24 | 4 | $0.0512 / h |

S3 (Canada Central region)

|  |  |
| --- | --- |
| Standard Storage | Price |
| First 50 TB / Month | $0.025 / GB |

RDS – PostgreSQL (Canada Central region)

|  |  |
| --- | --- |
| Standard Instance – Current Generation | Price |
| Db.t2.micro | $0.02 / h |
| Db.t2.small | $0.04 / h |
| Db.t2.medium | $0.08 / h |

VPC (Canada Central region)

* $0.05 per VPN Connection-hour
* AWS PrivateLink Pricing
  + $0.011 per VPC Endpoint per AZ
  + $0.01 per GB data processed
* NAT Gateway Pricing
  + $0.05 per NAT gateway
  + $0.05 per GB data processed

Security

<https://aws.amazon.com/security/?hp=tile>