



FAG EL-GAMOUS

WEBSITE PROJECT

PREPARED BY CADE LOAR,
BRANDON JACOBS, YONGHEE LEE,
RILEY MARSHALL

IN ASSOCIATION
WITH BYU



Table of Contents

1. Group Information	3
2. URL and User Credentials	3
3. Solution Details	
a. IS 404	5
b. IS 413	8
c. IS 413	8
d. IS 455	9
4. Executive Summary	11



Group Information

Group Information	
Group Number	4-9
Group Members	Cade Loar
	Brandon Jacobs
	Yonghee Lee
	Riley Marshall

URL and User Credentials

	URL
Website URL	https://fag-el-gamous.party/
.ipynb URL (Data Cleaning)	https://drive.google.com/file/d/1m8rQ5LMzGqQHl32Gwi4rFMCsQEepCwc2/view?usp=sharing
.ipynb URL (Supervised Determining Sex)	https://drive.google.com/file/d/1DZXpK4YNt066W2f37_izlmtSAWBR7ui2/view?usp=sharing
.ipynb URL (Supervised Head Direction)	https://colab.research.google.com/drive/1Bxgq8gfgEb8cvWiXG4ylZ8m3y-ewbwR7?usp=sharing
.ipynb URL (Supervised API Code)	https://colab.research.google.com/drive/1Ky1J0zLJKdEPGR7fe_BfGGmVfuqWvo0j?usp=sharing
.ipynb URL (Unsupervised Model)	https://colab.research.google.com/drive/1OFva9AMqgdKfvbLwV1_gSq7zmtB5Ym2F?usp=sharing
Google Colab	https://drive.google.com/drive/folders/19vJPhZQflvBjE796Y-gNZCX1KQKKnUS?usp=sharing

	URL	GitHub Branch
Repository URL	https://github.com/brandongjacobs1/fag-el-gamous	main

Username	Password	Role	2FA/MFA Used (if any)
ctloar@gmail.com		Admin	TOTP
giveusanA@ta.com	TA-password-123	Admin	None
research@ta.com	Researcher-TA-123	Researcher	None
test@ta.com	Testing-Testing-123	None	None

Solution Details

IS 404

Existing Solution

Amazon RDS: We utilized Amazon RDS to store all the data regarding burials and all the data related to our users. This made it easier for us to manage the data and keep track of it. Additionally, we ensured that the user-related data was encrypted to ensure data security.

Amazon Secrets Manager: To keep our parameters from being publicly available, we utilized AWS Secrets Manager. By passing in the parameters, we were able to protect them from unauthorized access and ensure that they were only accessed by authorized users and applications.

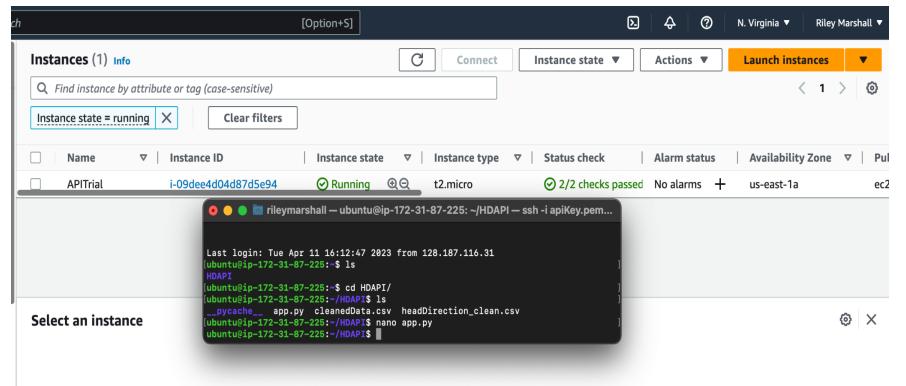
The screenshot shows the Amazon RDS Databases console. At the top, there's a header with a search bar and navigation links. Below it, a banner with an info icon suggests creating a Blue/Green deployment. The main area is titled 'Databases' and contains a table with one row. The row details a database named 'intex-database-1' which is an PostgreSQL instance in the 'us-east-1d' region, using db.t3.micro storage, and is currently 'Available'. There are 'Actions' and 'Clone' buttons at the bottom of the table row.

The screenshot shows the Amazon Elastic Beanstalk environments console. A banner at the top informs about the redesigned console. The main area is titled 'All environments' and lists a single environment named 'Intex-2-env'. The environment is marked as 'Green' and is associated with the application 'intex-2'. It was created on April 10, 2023, at 09:54:19 UTC-0600. The URL listed is 'Intex-2-env.eba-dg3begx.us-east-1.elasticbeanstalk.com'. There are 'Actions' and 'Create a new environment' buttons at the top of the table.

Amazon Elastic Beanstalk: Our website's main bulk was hosted on Elastic Beanstalk, an AWS service that makes it easy to deploy, manage, and scale web applications. This allowed us to easily manage the website and ensure that it was always up and running. Within the elastic beanstalk, we are using services such as Load Balancer and Auto Scaling.

Cloudflare: We used Cloudflare to buy our domain name and obtain a TLS certificate, which ensures secure communication between our website and our users. Additionally, Cloudflare provides various security features such as DDoS protection and web application firewall, which helps protect our website from attacks.

Amazon EC2 Instance: To host the API that we use for our models, we utilized an EC2 instance, which is a virtual machine that can be used to run applications. By using an EC2 instance, we were able to ensure that our API was always running and could handle the incoming requests.



The screenshot shows the AWS EC2 Instances console. A single instance named "APITrial" is listed, showing it is "Running". The instance ID is "i-09dee4d04d87d5e94". The instance type is "t2.micro" and it has "2/2 checks passed". The availability zone is "us-east-1a". Below the table, a terminal window is open on the instance, displaying a login session and some command-line activity related to an API endpoint.

```
Last login: Tue Apr 11 16:12:47 2023 from 128.187.116.31
[rileymarshall@ip-172-31-67-228:~]$ cd HDAPI/
[rileymarshall@ip-172-31-67-228:~/HDAPI]$ ls
__Pycache__ app.py clearedData.csv headDirection_clean.csv
[rileymarshall@ip-172-31-67-228:~/HDAPI]$ nano app.py
[rileymarshall@ip-172-31-67-228:~/HDAPI]$
```

Amazon Elastic IP: The IP address of both the application and the API on the EC2 instance needs an elastic IP. The IP address of our API is called in our program so that has to stay consistent.

Service Recommendations

Below is a list of AWS services that we did not include in our web application but that may be beneficial.

1. Amazon API Gateway: This service allows you to create, publish, and manage APIs, making it easier to connect to the model running on the API and expose its functionality to the web app.
2. Amazon S3: This service is an object storage service that can be used to store and retrieve data for the web app. You can store static files like images, videos, and documents, as well as dynamic data like user uploads and application data.
3. Amazon CloudFront: This service is a content delivery network that can be used to deliver content to users with low latency and high transfer speeds. It can improve the performance of your web app by caching and distributing content from S3 and other sources.

4. Amazon Lambda: This service allows you to run code without provisioning or managing servers. You can use it to execute the API or perform background tasks, such as processing uploaded data or generating reports.
5. Amazon Route 53: This service is a scalable domain name system (DNS) that can be used to route traffic to the web app. It can help you ensure high availability and low latency for the users by routing traffic to the closest available server.

Estimated Budget

Based on the AWS monthly budget calculator results provided below, the estimated monthly budget to provide this application with approximately 100 daily users is around \$271.15 USD. This includes hosting the API with an estimated cost of \$1.79 USD, database management using Amazon RDS with an estimated cost of \$186.16 USD, utilizing AWS Secrets Manager with an estimated cost of \$10.13 USD, and using Elastic Load Balancing with an estimated cost of \$31.03 USD. Additionally, the estimated cost of a single Beanstalk Instance is \$42.05 USD. It is important to note that the actual monthly budget may vary depending on usage and the number of daily users. Nevertheless, based on the given information, an estimated budget of \$271.15 USD per month is a reasonable estimation for supporting approximately 100 daily users.

The screenshot shows the AWS Pricing Calculator interface. At the top, there are buttons for 'Contact Sales' (with a phone icon), 'Feedback', 'English' (with a dropdown arrow), and 'Create an AWS Account'. Below the header, there's a summary table:

Upfront cost 0.00 USD	Monthly cost 271.15 USD	Total 12 months cost 3,253.82 USD <small>Includes upfront cost</small>
--------------------------	----------------------------	---

On the right side of the summary table are two buttons: 'Get started for free' and 'Request a quote'. Below the summary table is a section titled 'My Estimate' with a table of resources and their costs:

My Estimate					
<input type="button" value="Duplicate"/> <input type="button" value="Delete"/> <input type="button" value="Move to"/> <input type="button" value="Create group"/> <input type="button" value="Add support"/> <input type="button" value="Add service"/>					
<input type="text" value="Find resources"/> < 1 > ⌂					
<input type="checkbox"/>	Service Name	Upfront cost	Monthly cost	Description	Region
<input type="checkbox"/>	Amazon EC2	0.00 USD	1.79 USD	Host our API	US East (Ohio)
<input type="checkbox"/>	Amazon RDS for MySQL	0.00 USD	186.16 USD	Database	US East (Ohio)
<input type="checkbox"/>	AWS Secrets Manager	0.00 USD	10.13 USD	-	US East (Ohio)
<input type="checkbox"/>	Elastic Load Balancing	0.00 USD	31.03 USD	Part of Elastic Beanstalk	US East (Ohio)
<input type="checkbox"/>	Amazon EC2	0.00 USD	42.05 USD	Beanstalk Instance	US East (Ohio)

At the bottom of the page, there are links for 'Privacy', 'Site terms', 'Cookie preferences', and a copyright notice: '© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.'

IS 413

Almost all of the requirements for our web application for IS 413 are apparent by visiting the website or looking at our code. One thing to note is that we used an API to call our models that are being hosted on a separate EC2 instance rather than incorporating the models right into the ASP.NET files

IS 414

Security Features

Most of the security features are apparent on our web application or within our code. There are only a couple things to note here. The first is that we implemented additional roles in our RBAC. We have an unauthorized user and different authorized user roles. The two main authorized roles are researcher and admin. Only a researcher can edit and delete data while only an admin can visit the administration pages. Another thing to note is how credentials are handled. By default, ASP.NET makes a SQLite database within the application to store user information. Even though the passwords are hashed, other user information can be publicly seen when we post our project to github. To handle this, we created a database in AWS to store all user information. The last thing to note is that we obtained our TLS certificate from cloudflare after we bought our domain name.

Egypt Data Protection Law

In 2020, Egypt passed a Data Protection Law. Below are our recommendations on how to comply with this law

- Appoint a Data Protection Officer (DPO): The law requires certain organizations to appoint a DPO who will be responsible for ensuring compliance with the law.
- Develop Data Protection Policies and Procedures: Develop a policy and procedure that addresses how personal data is collected, processed, stored, and secured in compliance with the law. This should include guidelines for obtaining consent, managing access to data, responding to data breaches, and other related areas.

- Respond to Data Breaches: Have a data breach response plan in place that outlines how they will respond to and manage a data breach. This includes notifying affected individuals and authorities as required by law.

IS 455

Below are specifics on how we met and went above and beyond the requirements for both the supervised and unsupervised models. We also included a general overview of combining and cleaning the data.

Data Combination & Cleaning

The code for our data cleaning and combing can be found above. We downloaded every table given to us as a .csv then combined them using foreign key relationships. We then did some primary cleaning of the data by getting rid of columns with missing records and deleting extra columns (columns containing foreign keys that weren't needed).

Supervised

We have two supervised trained models. One model predicts the head direction of the mummy. The second model predicts the sex of the mummy. The second model was what we did to go above and beyond the requirements. For both models, we began by cleaning the data that was combined before. The data used in the model that predicts head direction was filtered to only include rows where head direction was not null. The same goes with data in the model predicting sex, but filtered down to only include rows with non null values for sex.

After both sets of data were filtered, we then dropped columns that had null values for roughly 25% of the data. After columns were dropped, we filled in the rest of the null values with either the mean of the column or the mode, depending on if the column was numeric or not. This gave us two data sets with no null values. We then used this data to create a model using a decision tree.

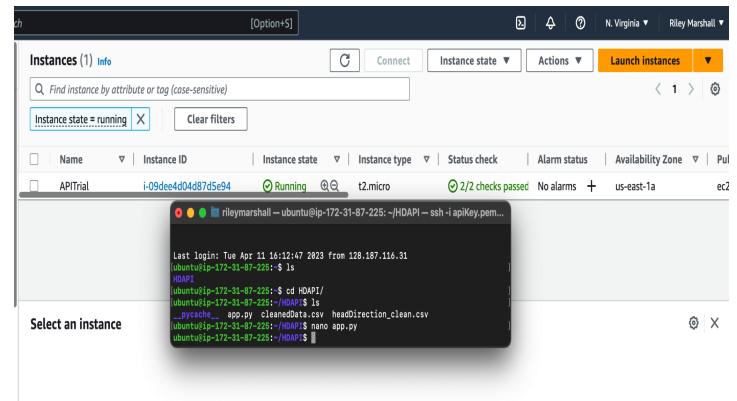
We thought having a user enter in all the columns to get a prediction for both head direction and sex was too much. We decided to narrow it down to seven fields for the head direction prediction and six fields for the sex prediction, as seen on our website. This way a user is able to enter in enough information for the model to make an accurate prediction but not overwhelm the user. These fields were chosen based on the instructions for the project

(factors relating to the burial itself) and based on the fields that had the most influence in determining what the model was predicting.

We then took both models and made them into a FastAPI. That code can be seen above. We put all the data we needed to and the FastAPI python file into an AWS EC2 instance and configure it to automatically run the file when the instance is running. The EC2 instance and a view of the files inside can be seen in the photo on the right.

Unsupervised

In our research on Unsupervised models, our primary objective was to investigate the hypothesis that burial practices in the cemetery underwent a change over time. To achieve this, we employed two distinct models and performed rigorous clustering analysis, leveraging a range of inputs such as burial depth, burial orientation (e.g. head direction), age at death, and sex variable.



Given the heterogeneous nature of our data, incorporating varying scales and mixed data types, we selected the Gower distance measure and the agglomerative clustering algorithm to perform our hierarchical clustering analysis. Prior to analysis, we filtered our dataset to include only relevant variables. Furthermore, we imputed missing data values by dropping columns with null values and replacing the remaining null values with either the mean or mode of the column, depending on the nature of the data. This preprocessing step ensured that our dataset was complete and ready for clustering analysis.

Subsequently, we leveraged our preprocessed data to construct a hierarchical clustering model, utilizing K-means clustering for optimal results. Our approach successfully enabled us to uncover and delineate meaningful patterns and trends within the data, despite the inherent complexity of our dataset.

Executive Summary

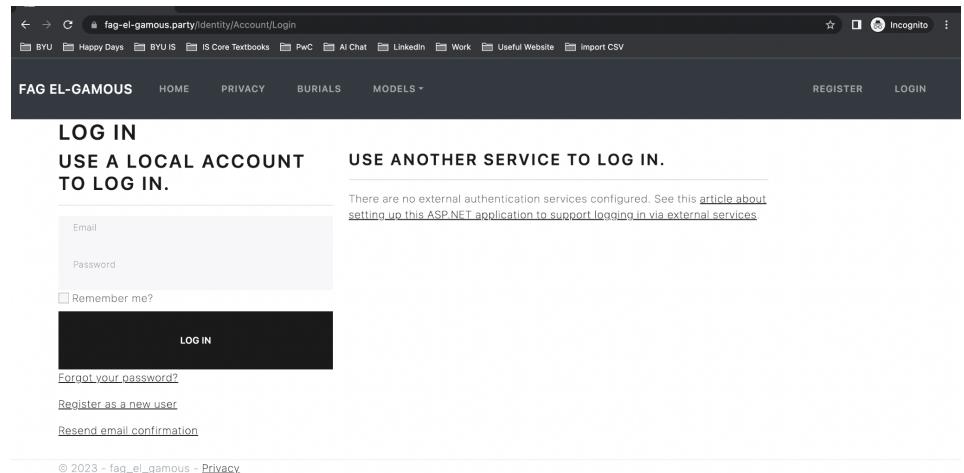
This project proved to be a multifaceted and gratifying endeavor, which not only tested our team's mettle but also rewarded us with valuable insights and lessons. Despite encountering various obstacles, such as the need to pivot our models and continually debug our codebase, our group's collective efforts culminated in the successful development of our web application. Our unwavering commitment to excellence and determination to surmount challenges along the way enabled us to produce a solution that goes beyond the expected standards.

Our web application boasts an array of features that make it a comprehensive solution for online data manipulation and analysis. Among its key functionalities are robust security protocols, a user login and control system, and the ability to manipulate and analyze data with ease.

Our security measures include the use of encrypted passwords, two-factor authentication, and role-based access control, which ensures that sensitive data is only accessible to authorized personnel. Moreover, our website features a TLS certificate, and users are prompted to verify their intentions before deleting any data.

Upon visiting our website, users are prompted to register and create an account. Once registered, users' information is securely stored in our database, enabling them to log in and out as needed. Depending on their assigned roles, users can access different parts of the website. Researchers can add, edit, and delete data related to mummies, while admins can manage user information through the administration pages. These roles are also stored in our database, ensuring that access to sensitive data is restricted to authorized personnel.

The primary objective of our web application is to display



data and make predictions using models built from the provided information. Our user-friendly interface allows anyone, whether registered or not, to filter and browse data related to burials. With over ten filters to choose from, users can quickly find the information they need. For each burial record, a detailed page is available that displays all available information.

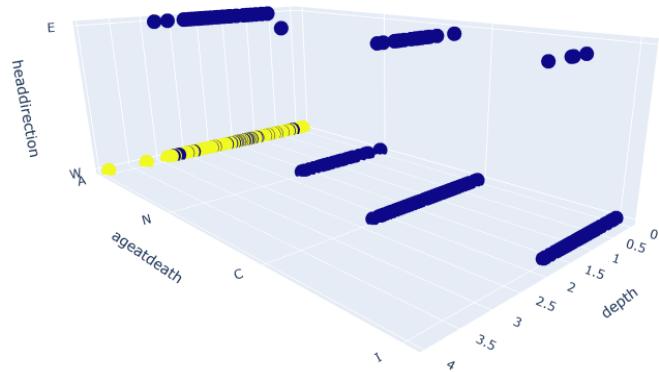
The screenshot shows a web application interface for predicting head direction and sex of mummies. The 'HEAD DIRECTION PREDICTION' section contains fields for Depth, South to Head, South to Feet, West to Head, West to Feet, Area, and Wrapping. The 'SEX PREDICTION' section contains fields for East/West, Depth, Hair Color, Wrapping, Age at Death, and Samples Collected. Both sections have a 'PREDICT' button. Below the form, the results are displayed: HEAD DIRECTION: and SEX:.

Our site has an area for a user to predict either the head direction of a mummy or the sex of a mummy. If predicting the head direction, a user needs to input seven fields of information. If predicting the gender, a user needs to input six fields of information. After the submit button is pressed, a post request is sent to an API with the field information as the main body and a prediction is

returned. This prediction is then displayed underneath the form that the user filled out.

After training both supervised and unsupervised models, we ‘unearthed’ some interesting findings within the data. The supervised data shows that the biggest influence in determining sex is the field titled “South To Feet”. The biggest influence in determining head direction is the depth the mummy is buried. Although both these fields are the most influential factors in determining their respective variables, neither are significant. There is no significant correlation, according to our supervised data, in any of the fields when trying to determine sex or head direction.

Additionally, users can access our visualizations which were created using a cluster analysis algorithm to provide an intuitive and interactive way to explore the data. These visualizations revealed several patterns



and insights on the burial practices in the cemetery.

Firstly, our analysis showed that age at death, head direction, and depth were the most critical factors that influenced the clustering of the burials. Our analysis showed burials of adults, infants, and newborns were clustered together in cases where head direction was towards the west. This finding suggests a potential cultural or religious significance in burying individuals with their heads pointed towards the west.

Moreover, as the burial depth decreased, we observed a higher number of cases where the head direction was towards the west. This finding further reinforces the potential cultural or religious significance of this burial practice and may suggest a shift in burial practices over time.

Overall, our findings highlight the importance of utilizing data analysis in uncovering meaningful insights and patterns in complex datasets. These findings can inform future research on the burial practices in the cemetery and provide a more comprehensive understanding of the cultural and social factors that shaped these practices.

The development of our multifaceted web application was a challenging but ultimately rewarding experience for our team. Our perseverance in overcoming obstacles such as the need to pivot our models and constant debugging ultimately enabled us to produce a sophisticated solution that surpassed the expected standards. The website boasts a plethora of features, including robust security measures, an intuitive user login and control system, and the ability to manipulate and analyze data. These functionalities make our web application an excellent choice for users and archaeologists alike.