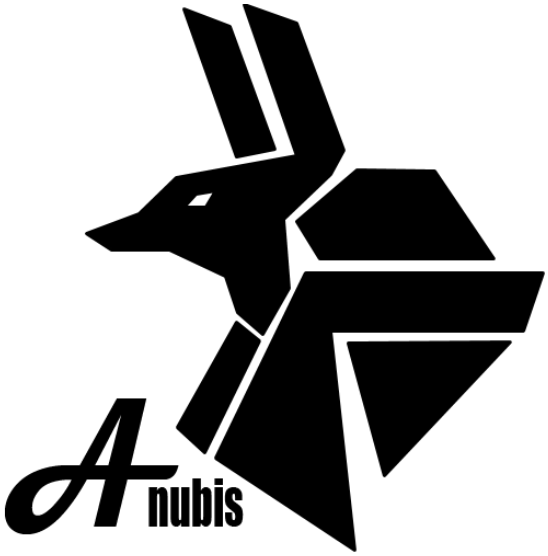


Technology Feasibility

Team Anubis

10/27/16

Sponsor:

Cindy Brown, NAU University Development
on behalf of the NAZ Animal Welfare Task Force

Faculty Mentor:

Steven Jacobs, Lecturer SICCS

Frankie Berry
Steven Gruenewald
Marjorie Hahn
Riley Shelton
Matthew Siewierski

Table of Contents

Introduction

We are Team Anubis. Our members include Frankie Berry, Steven Gruenewald, Marjorie Hahn, Riley Shelton, and Matthew Siewierski. Our project is to create a website for the NAZ Animal Welfare Task Force that users can use to input data about themselves and be matched to a pet that fits the adopter's lifestyle. The project sponsor, Cindy Brown, is the Assistant Director of Development for Student Success at NAU, and the former director of the Second Chance Center for Animals in Flagstaff.

This document is used to explore how feasible this project will be, and compare what technologies can be used to solve this problem. We begin with technological challenges, where we discuss what the system needs to do, and any challenges that might be faced. Then, in the technical analysis section, we discuss the challenges in a bit more detail, and look at a few approaches that can be taken to solve them. We also discuss which choice we went with, and how we plan on proving the feasibility of using the chosen technologies. We then end the document with a conclusion that summarizes what was discussed in the document.

Problem Statement

The problem that we are trying to solve is many NAU students that decide to adopt a pet while they are going to school then realizing they cannot provide for it properly then returning it to the shelter. Changing homes can be traumatizing to the animals, so this website will help adopters choose an animal that they will keep permanently, or, alternatively, screen them to not be a pet owner while in school.

We will be producing a website that potential pet adopters can use to find a pet they will take care of for a longer period of time.
Features:

- Shelter users can add or change pet information as needed.
- Users can create an account and answer questions about their home, lifestyle, and pet preferences.
- Users will then be given a list of pets that match the information collected from the questions.
- Users can also view a list of all animals available for adoption.
- Admins can add additional fields to the database, and additional shelters to the system.

Technological Challenges

The need for a system that allows animals and users to get matched to one another based on characteristics including lifestyle and personality. Does lead to several challenges both in the interface and the backend of the system. The following are the technological challenges identified for the system below:

- The system will need a secure interaction and user authentication on our web application.
- The system will need a method to communicate with a backend database.
- The system will need a way for taking the information from a database and display it in a clean and organized layout.
- The system will need a way to provide a responsive interface onto a wide variety of platforms (personal computer, tablet, smartphone) that allows users to create a profile including information on lifestyle and personality.
- The system will need a way to find potential animal matches within the system.
- The system will need to match users to animals based on profiles.
- The system will need a way to ingest large database files such as csv (comma separated value), excel sheets.
- The system will need to allow users to administrators to update the database and to modify question fields on the profiles of users and animals.
- The system will need to be integrated with a content management system for easy editability and maintainability for the end user.

Technical Analysis

Framework

Intro

The first challenge is the system will need a secure interaction and user authentication on our web application. This is needed for all users to login to the system and or to create a profile. Due to the possibility of entering sensitive information being entered into the system the security is needed. The entire system is driven by use of this entered information in the various components of the system. Secondly the system will need a method to communicate with a backend database. This is for the storage of the information being collected from the profiles being created and the input from shelters for later analysis and retrieval. Another challenge is the system will need a way to provide a responsive interface onto a wide variety of platforms (personal computer, tablet, smartphone). Allowing for a similar experience across all the devices, so there will be little to no learning curve from one device to another. Finally, the system will need a way for taking the information from a database and display it in a clean and organized layout. With all the system's information being stored externally, the system needs to retrieve information. The cleaner the layout the easier it is for the end user to see information and understand it completely. So that there will be no surprises and all the information is visible.

The main challenge and component of the system will need a way to find potential animal matches within the system. The system will analyze the information put into the system to find matched between animals and users, based on information in the profiles. More precisely the

system will need to match users to animals based on the information contained in the profiles created by either themselves or the shelter. The goal is to have an animal matched to a user that has some common traits between each other. So that the adoption is based on the needs and characteristics of each other, rather than the simple physical characteristics of the animal that are pleasing on the eyes of the user. For example a highly active dog that loves children, would potentially get matched to a young family that has yard, leading to an adoption that has the potential to last years.

Alternatives

- Apache Cordova - a mobile application development framework which supports responsive design on multiple devices through converting web applications to a mobile form. [2]
- Django - a diverse web framework which takes an alternative approach to web development through Python as its primary language. [5]

Alternatives Compared		
Method	**Pros**	**Cons**
Apache Cordova	● Portability ● Easy to learn with prior web development knowledge ● Quick testing ● Develop applications using JavaScript	● Many devices to account for ● Requires powerful computer to run emulator(s) ● Lacking detailed documentation
Django	● Easy-to-understand Model/View/Control layout ● Allows Python programming in web environment ● Built-in Object-Relational Mapping ● Builds on HTML, CSS and JavaScript ● Bootstrap compatible ● Security built into framework	● Slight overhead in using Python ● Learning curve to understand framework

Chosen Approach

We decided to use Django because it provides a nice clean, easy-to-use interface, and since we are not developing this as a mobile application, we will not be using Apache Cordova. Django is nice since it builds on typical web development languages including HTML, CSS, and JavaScript with Python in the background, so it allows for easy web building with the Bootstrap

library. Another key benefit of Django is that it allows for development of a responsive interface with the mix of Python and Bootstrap. It also allows for the implementation of the matching easily with Python.

Feasibility

For the challenges of matching users to animals based on profiles, finding potential animal matches within the system and building a responsive interface onto a wide variety of platforms (personal computer, tablet, smartphone) that allows users to create a profile including information on lifestyle and personality we will programmatically solve these problems using Django's web environment Python API, which our team has a considerable amount of experience in[5]. To solve the security challenge Django natively supports XXS(cross site scripting protection), Cross Site Request Forgery(CSRF), Clickjacking Protection and Is deployed behind HTTPS[5]. The taking the information from a database and display it in a clean and organized layout we will be taking it from the existing database using the Python API and displaying using HTML and CSS [5].

Conclusion

With the need for a clean and responsive interface at the same time robust and easy to use. It was easy to select the Django framework for our system. It has the ability for the needed security for the information that we are ingesting from various sources. It also allows for our responsive web application to be developed with ease using the combination of HTML, CSS, JavaScript and Python. It will bring together all the other various technologies into a cohesive system.

Database Integration

Intro

A challenge for the backend database is the system will need a way to ingest large database files such as csv (comma separated value), excel sheets. This is needed for the initial connection of a shelter to the system, so that all the existing animal information will be ingested. Another need for this ability is that if for some reason the shelter needs to take in many animals at once they have the ability to put it into the system all at once.

Alternatives

- MongoDB - a non-relational document-oriented database program. [8]
- MySQL - a relational database management system. [1]

Alternatives Compared		
Method	**Pros**	**Cons**

MongoDB	● Non-relational database ● Not at risk for SQL injection ● Allows separation of large databases into smaller and more efficiently managed parts ● Low cost	● Data size generally larger than other databases ● Need to join data manually ● Does not handle database transactions well
MySQL	● Allows immediate manipulation of data ● Built-in administrative functions ● Portability	● SQL injection security risks ● Rigid format ● Difficulty in interfacing

Chosen Approach

We decided to use mongoDB because it is not as rigid as MySQL, and does not have the risk of SQL injections. This allows us to have more freedom with how we choose to store the data that we are ingesting.

Feasibility

For the method to communicate with a backend database challenge, we will use Django-nonrel to set up communication between Django and our Mongo database. [4] The way to ingest large database style files challenge will be tackled using the Django MongoDB Engine. [9]

Conclusion

With choosing mongoDB as our backend database system, it will allow us to easily ingest the information from the shelters various systems with easy. Django allows for easy integration of mongoDB with a simple plugin. Without the forced relationships that are a commonality with MySQL it allows for the freedom that we need due to the variety of information that will be stored. It will allow us to store the needed information with ease while allowing the necessary freedom on how we are to retrieve the information in our systems matching tool.

Maintainability

Intro

A challenge for the long term maintainability is that the system will need to be integrated with a content management system for easy editability and maintainability for the end user. This would just make the system able to be maintained by a normal non technical user. So that they may have someone on their end update the pages with current information so that it can stay relevant and ultimately helpful for the users. Another challenge for the maintainability is that the

system will need to allow users and administrators to update the database and to modify question fields on the profiles of users and animals. For this system to be able to be implemented by other shelters in different areas, there may be the need for additional information that is legally required in that area to be added. This should allow the system to be more flexibility and overall last longer.

Alternatives

- Divio - a content management system intended to be used alongside Django with built-in integration. [3]
- WordPress - an independent content management system used for different websites, ranging from personal to professional content. [7]

Alternatives Compared		
Method	**Pros**	**Cons**
Divio	● Well-supported and integrated with Django ● Easy-to-use content management system ● Built-in group management system with access rights ● Database integration supported ● Easy to maintain	● Reliant on Django framework ● Requires prior knowledge of storage and memory needs ● Limited support on lower tier plans
WordPress	● Easy-to-use content management system ● Google Ads integration ● Reliable reputation	● Limited maintainability ● Expensive ● Requires software updates regularly

Chosen Approach

We decided to use Divio, because it works well with Django, supports database integration, and is easy to use. We decided not to use WordPress because it is harder to maintain, and is not as secure as Divio. The server will have the following specification allowed by the custom server option provided by Divio. The server will have 512Mb of RAM, 40Gb of disk space, 80Gb of bandwidth, there will be two instances of the server running to reduce the chance of the site being down. There is also that ability to use there out of the box solution for a business, it has the same basic functionality just offers additional support and resources.

Feasibility

For the ability to allow administrators to update the database and to modify question fields on the profiles of users and animals and allow users to administrators to update the database and to modify question fields on the profiles of users and animals the Divio content management system provides an easy to use Django CMS version updates. [3]

Conclusion

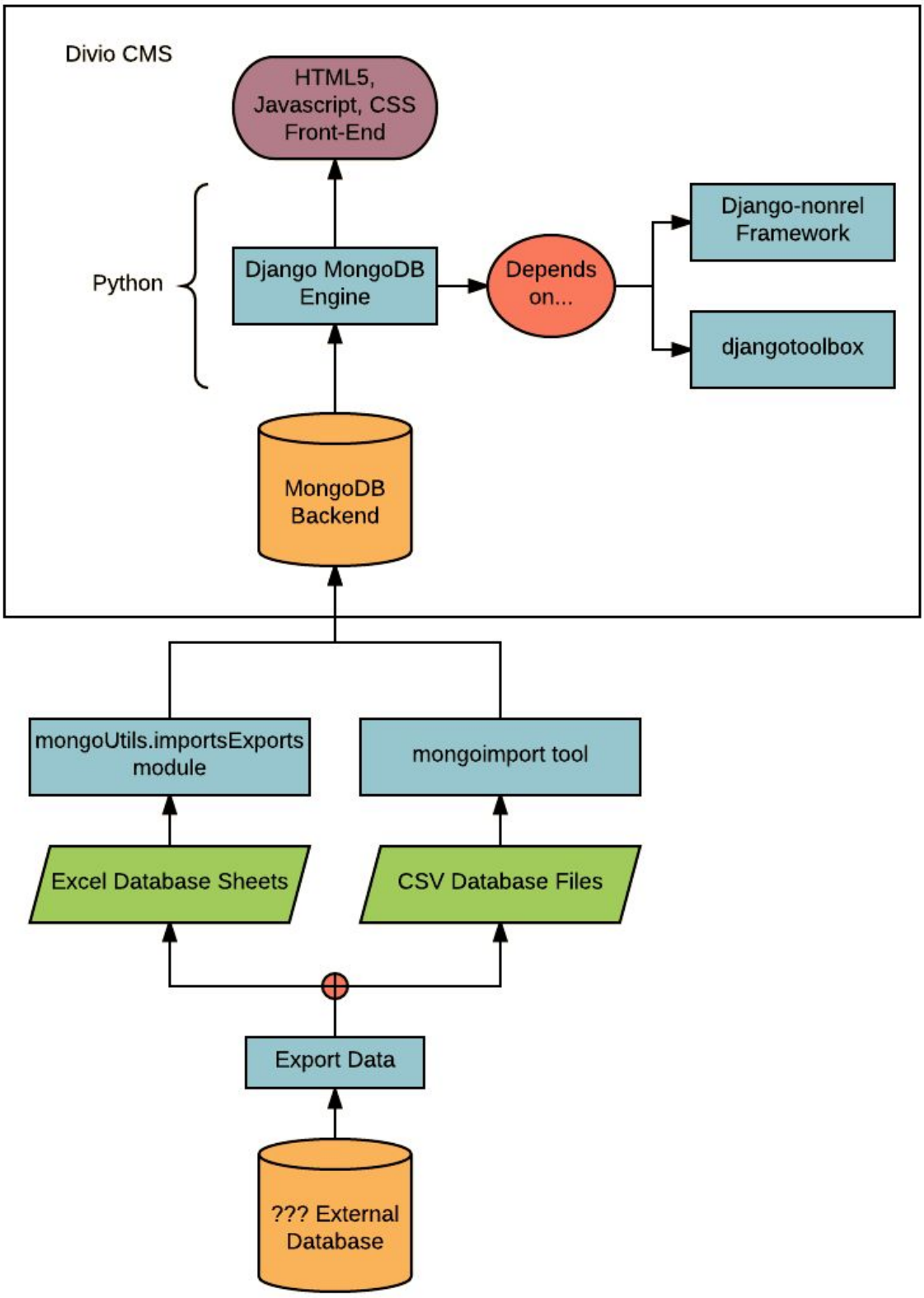
Maintainability was a key necessity, due to the need for a non technical user to be able to update and maintain the site. Divio offers that ability easily due to it being built around the Django framework, It offers the ability to have a drag and drop content editing so anyone should be able to update it. Another key element for using Divio offers to all clients an SSL certificate with any of their packages to reinforce the native security in Django.

Technology Integration

In order to get our system to function the way that we want it to, we are going to face several integration challenges that are outlined below. Fortunately, there are plenty of technologies available to us that will allow us to accomplish this project. The first big challenge will be integrating existing databases from animal shelters with our own database. Another challenge will be integrating a MongoDB database with the Django web framework since, unfortunately, MongoDB is not a default supported Django database. More fortunately, Divio, our content management system and Django are already very well integrated with each other, so that will not be an issue for us to solve.

System Integration Diagram:

This diagram is a visualization of how all the technologies that we will be using for this project will mesh together. The ins and out of which are described below.


Divio CMS
HTML5, Javascript, CSS Front-End
Python
Django MongoDB Engine
Depends on...
Django-nonrel Framework
djangotoolbox
MongoDB Backend
mongoUtils.importsExports module
mongoimport tool
Excel Database Sheets
CSV Database Files
Export Data
??? External Database

Integration Issue 1: Integrating Existing Databases with our MongoDB Database

MongoDB has a built in import functionality that we can use to import CSV files to a MongoDB database. [10] There is also a mongoUtil import utility that can be used to import an Excel spreadsheet to MongoDB. [11] As long as the existing animal databases can be exported to an Excel sheet or CSV file, then we will be capable of importing them to the database that we will be creating. If worst comes to worse, we can write our own parser or script to transfer existing data to our database. Written records will ultimately have to be moved over by hand.

Integration Issue 2: Connecting Back-end Database with Web Framework

We are going to use a MongoDB engine for Django that will allow us to use Django's object-relational mapping, admin site, authentication, site, session, and caching frameworks with MongoDB. [9] In order to utilize this engine, we will be using a fork of Django that is capable of supporting non relational databases like Mongo. [4] We will also be using its corresponding API toolbox, djangotoolbox, which will provide several utilities for non-relational Django applications and backends. [6]

Conclusion

The problem that we are trying to solve is how quickly animals are being adopted and then thrown away by students who decide they cannot take care of them. Our solution is to create a friendly website for students to get matched to the perfect animal for them, and thus decrease the number of animals that get returned to shelters. In order to get one step closer to making this solution a reality, we have put together this technological feasibility document to make sure that we are technically capable of building this project. Below is a table summarizing each technical issue we are facing, and what technologies we are planning to use in order to solve them.

Technological Challenges Summary	
Issue	Solution
We need a web frame that will allow us to easily build and create the website.	Django, an easy to understand Python based web framework.
The content management system that we use must be easily maintainable for our client.	Divio, an easy to use CMS that is compatible with Django.
The database must be flexible and easy to build upon.	MongoDB, a flexible, non-relational database.
We must be able to connect our chosen	We will use a third party engine,

database to our web framework.	mongodb-engine, to integrate MongoDB with Django.
We need to be able to move over existing databases to the database that we are creating.	MongoDB provides import functionality for moving over existing data.

We are confident that we will implement this project given the technologies described above. One uncertainty that still remains is how the current animal data is being kept. However, we have prepared ourselves with a few alternative solutions for moving over that data, so whatever the format may be, it should not be a problem.

References

1. About MySQL [Website]. (2016). Retrieved from: https://www.mysql.com/about/.
2. Architectural Overview of Cordova Platform [Website]. (2016). Retrieved from: https://cordova.apache.org/docs/en/latest/guide/overview/.
3. Divio: Platform for Python/Django [Website]. (2016). Retrieved from: https://www.divio.com/en/.
4. Django-nonrel [Computer software]. (2015). Retrieved from: https://github.com/django-nonrel/django.
5. Django: The Web Framework for Perfectionists with Deadlines [Website]. (2016). Retrieved from: https://www.djangoproject.com/.
6. djangotoolbox [Computer software]. (2015). Retrieved from: https://github.com/django-nonrel/djangotoolbox/.
7. How does Django CMS compare with WordPress? [Website]. (2016) Retrieved from: http://support.divio.com/migration/other-cmss/how-does-django-cms-compare-with-wordpress.
8. MongoDB [Website]. (2016). Retrieved from: https://www.mongodb.com/.
9. MongoDB integration for Django [Computer Software]. (2015). Retrieved from: https://github.com/django-nonrel/mongodb-engine.
10. mongoimport functionality [Website]. (2016). Retrieved from: https://docs.mongodb.com/manual/reference/program/mongoimport/#bin.mongoimport.
11. mongoUtils.importExports module [Website]. (2014). Retrieved from: http://miloncdn.appspot.com/docs/mongoUtils/mongoUtils.importsExports.html.