Research

Web Frameworks

The project will be deployed as a web application and to avoid developing commonaly used features like database abstration models, user autherization and routing I have decided to use a webframe work. There are many webframe works available. Below I will discuss the different types and their advantages and disaadvantages.

The first webframe work I am going to look at is Django. Django is a high level web framework built with python. It has a alot of built in features that it comes with out of the box. These features are bundled into componets called apps. Apps can be added to your django project and they can automatic a lot of common tasks involved in web development. Tasks such as user authorization and authetencation and much more. Django claims to be extremly fast, scalable and secure [1]. Companies like instagram, youtube, spotify and bitbucket use django [1].

### Pros

* **Models**: Django has an inbuilt feature where you can define models in a python class. Each field in the model can be converted to a database table. This allows you to define your relational database as a class. You can migrate your models and django will setup a database with models you defined with a database of your choosing. The most powerful and useful feature about django models are once you migrate your models django will provide you with API to query your database through python. This is particually useful when combined with views which are discuesed below. [2]
* **Urls**: Another useful feature of django is its url dispatcher. Django alows you to define your own custom urls and map them to a view which will handle the request.[3]
* **Views**: Views are how djangos handles requests from the user. When a user goes to a certian url you defined in your url dispatcher it will be mapped to a view. The view will process the request and return a respone e.g. a html page. Views are a big selling point for my project when combined with django model API it will allow me to make request to my database to retrieve data to be visualized. [4]
* **Uses Python**: Django uses python which I am very familuar with and has a lot of great libraries.
* **Security**: Django includes a securty layer that helps prevent Cross-site request forgery and sql ingections. [5]
* **Admin page**: Django provides you with a built in admin page that allows you to manage your models, users and groups.
* **GeoDjango**: Geodjango is a feature that comes with django that help you make spacial queries to a database. You can define spacial datatypes like multi-polygons and provides you with a model API to make spacial queries e.g. return the conntry point(x,y) is in.

### Cons

* **Everything include**: Django has a lot of features include by default which allow the user

Flask is a web framework built with python. It is commonly called a micro framework because unlike django it doesn’t come with user autherization or a database abstraction model. However these features can be added to flask using extension. Extension are components at automate alot of common task in web development like user autherization. This way you can build a web application with just the functionality and not have any unwanted modules and bloatware. [6]

### Pros

* **Extensions**: Extension allow you to create a minimal app with just the components you need.
* **Python**: Uses python which has a lot of great libriaries

### Cons

* **Documentation**: Flasks documention isnt to the same standard as djangos’
* **Project Structure**: While flask is a great and compact webframe work it doesn’t provide you with the same structure as django. Django provides you with a clean structured folder structure when you create a new project.

Ruby on rails is a web framework like flask and django. It automates a lot of the common task in web development and is built on the model view controller MVC philosophy. It uses ruby has its main programming language. It is classifified as an optinionated framework meaning you are encourged to develop your application in way they want to. [7]

### Pros

* **Clean code**: Ruby enforces the MVC design pattern and forces you to write cleaner more maintainable code.
* **Community**: Ruby on rails has a great community behind it with lots of posts on stackoverflow.

### Cons

* **Not flexible**: Ruby on rails is not as flexibale as Flask you can’t easily add extensions to it and it doesn’t have a inbuilt spacial component for dealing with spacial data like django does with geodjango.

After a lot of consideration I have decided django would be the best fit for this project. It provides a lot of commonly need tasks built in and an database abstraction model to easly perform queries to a database and the main reason I have chosen django is because of its geodjango extension app which will allow me to and geo spatial data.

Databases

For this application I will be dealing with a lot of large data sets. The criminal records data set just the city of chicago is 2GB. Will this isn’t too much if I were to added in more cities and towns this dataset could be large very quickly. One of the solutions to dealing with the larges datasets is normalizating the data (only if using a relational database). Another big aspect of my application is going to be performace. I want the user to be able to filter crimes and display them on a map as quickly as possible. Taking this into consideration I would need a database with fast read speeds. Another big part of this application will be spatial data like border polygons. I will need a database that can handle these datatypes. In summary the main attributes I am looking for in a database is read speed, scalibilty and can handle spatial data. I have to consider two types of databases SQL and no-SQL. The advantages of a no-SQL database is that it is more flexiable and you can dont have to spend too much time creating relational schemes. Although someone people may see this as a disadvantage depending on the project. No sql also can store data as a json object making it easier to parse and send out in a reqiest response. SQL on the other hand scales better than no-SQL because common values can be stored ina table and referenced with a key. SQL databases generally provide more flexablity for querying data. However this deponds on how you design your scheme. I am leaning towards towards a SQL database because it will provide a more flexible querys, scales well and as a added bonus I am more familar with SQL database. However below I will explore possible options.

The first database I looked at was postgres. Postgresql is a retional open source database it has been ACID complient since 2011. Postgresql is highly extensionable and has a big communtity behind it and a lot of documentation [8]. Postgresql caught my eye because of its PostGIS extension. This extension alows you to make spatial querys, for example is point(x,y) in ireland. This could be very useful for my project if I only want to display crimes in a certain county for town. Another big advantage of postgis is that it has spatial datatypes like polygon and multipolygon which can be used to store information about borders which can then be visualized on a map.

MongoDB is also a possible option for this project. It is a no-SQL database and is used by many large companies such as facebook, ebay, google, sap and many more [9]. The advantages of mongodb is that it is scalable, what this mean in my situation is that data can be stored in documents that are simular to json. When I want to add a new information that might have a new column for two I can just added it to the database with little to no hassle on the other hand in a SQL database I would have to alter the database table and possibily perform normalization to make sure the database is up to industry standard. This would cost extra development time. Another advantage of mongdo is that because it doesn’t have relations there is no need to developed a relational database model that I would most likely have to reference every no and then to make sure I am querying the right data and depending on how I design it I may need to re-design the model if I want to remove or add new columns. This is not a huge issue but something to consider as it could save me development time. In terms of query performance mongdb would be faster than a SQL database the downside is that if you want to perform more complex queries that need very specific data then noSQL might not be the best option.

In this project I am leaning towards mongodb because it is easy to set up and scales well with newly added data it is also not restricted to a relation schema. The one downside to mongodb is that it doesn’t have any fully featured extensions for handling geo spatial data like multipolygons for borders. While this data can still easily be stored in mongodb postgresql comes with an extension called postgis which provides custom datatypes another benifit to postgres is that it works well withs geodjangos model api. This means I can easly perform geo spatial queries from django to a postgres database with the postgis extension. For this reason I have decided to choose postgresql for this project.

DATASETS

A major factor in this project is going to be what datasets I choose. The datasets need to span span over a decent peroid of time be accutarate and have a lot of information. One of the first places I looked was the UK. The UK has a good dataset with lost of inforamation as well as longitude and latitude positions and a small description of the crime. The UK also has census data which can be used to filter crimes in areas that have a low deprevation index. A deprevation index is a measurement from one to ten of how aflunent or disadvanteged an area is. This is based of census data which includes poverty, population, type of employment/skill and a few others. This can be used to better visualize criminal data in certain areas.

Another possible candinate for datasets is the city of Chicago in America. The reason I am considering this city as a possiblilty is because it has a detailed dataset on crimes going back to 2001 census data that calculates deprevation index of areas and chicago as lot of dataset relation to public facities like transport, education, enviroment and many more [10]. In terms of datasets relating to crimes and census data Chicago and the UK are that different, however Chicago as a lot of other useful datasets that could become useful for visualizing criminal data after more research about what causes crime, which will be discussed below.

Orginally I conisdered ireland and a possiblity however while ireland has excellent census data its dataset on crimes is not very accurate and reports crime by garda divions which isn’t very useful for visualizations. It is also marked as “under reservation” by the cental statistics office, due to the data not beign accurate. [11]

I have chosen to use Chicago as the focus point for this application

After a quick anyalsis of the data I found there was very little missing values except for the comunity areas, where 8% of the values are missing. Community areas are areas in chicago that are used for gathering census data. Luckily using geodjango I can query the database of comunity area borders and using the longitude an latitude of the crimes and update the missing values.



Fontend Frameworks

# React

## Pros

## Cons

# AngularJs

## Pros

## Cons

# Bootstrap

## Pros

## Cons

Dev Tools

Mapping

# Leaflet

## Pros

## Cons

# Mapbox

## Pros

## Cons

# **ArcGis**

## Pros

## Cons

Hosting

# Docker

## Pros

## Cons

# Ngix

## Pros

## Cons

# Apache

## Pros

## Cons

refrences

(1) <https://www.djangoproject.com/>

(2) <https://docs.djangoproject.com/en/2.2/topics/db/models/>

(3) <https://docs.djangoproject.com/en/2.2/topics/http/urls/>

(4) <https://docs.djangoproject.com/en/2.2/topics/http/views/>

(5) https://docs.djangoproject.com/en/2.0/topics/security/

(6) http://flask.palletsprojects.com/en/1.1.x/

(7) https://guides.rubyonrails.org/getting\_started.html

(8) https://www.postgresql.org/about/

(9) https://www.mongodb.com/

(10) https://data.cityofchicago.org/

(11) <https://www.cso.ie/en/statistics/crimeandjustice/>