UniMap: Using Python's Django Framework to Identify On-Campus Traffic

CS4991 Capstone Report, 2024

Maseel Shah
Computer Science
The University of Virginia
School of Engineering and Applied Science
Charlottesville, Virginia USA
dda5us@virginia.edu

ABSTRACT
On college campuses, students often struggle to find the best times to visit areas like libraries and gyms, as these places are typically crowded at popular times. To solve this issue, my team and I created UniMap, a web-based solution using Python's Django Framework so students can use a scale of one to five to rank the current level of crowding at a given location. Building the website required the use of the Google Maps API to help users find the locations they wanted to rank or see in terms of traffic. Furthermore, it required the use of Django models track user rankings. This project allows students to more effectively plan visits across the UVA campus, including libraries, gyms, and restaurants, avoiding crowded areas, saving them a wasted trip. To make this project more accessible and useful in the future, it will need to be turned into a mobile app.

1. INTRODUCTION

College students have access to many different buildings across their university campuses, including gyms, libraries, dining halls, classrooms, and more. However, multiples of these locations exist across the campus, some busier than others. Not knowing how crowded a certain gym is or how many people are in certain study spaces can pose a large issue for students. For example, if a student goes to a specific gym but it turns out to be busy, it may be difficult to commute to a different gym across the campus. Use of the UniMap application can save time by helping students to identify busy places and find facilities that are less busy.

2. RELATED WORKS

Google's busyness ranking feature is similar to the UniMap ranking system. In this feature, users can select certain locations and determine how busy the area is on average during different times of the day across the week. Google calculates this by aggregating data from users who share their location. Using this information, Google determines at what times of certain days most people show up, providing a bar chart reference (Google, 2024). While this feature is similar to UniMap in the sense that it gives users some information about how busy areas are, it is not a live, current representation showing how busy locations are in real-time. Google's feature uses aggregated data over time to identify trends about when more people are likely to be in an area. By contrast, UniMap allows users to share in real-time the traffic at certain locations, as well as rank how busy the location is. UniMap rankings last for three hours, reiterating the importance of real-time data collection. The UniMap project also allows users to create new locations that may not exist on the map, whereas Google's version does not account for low-profile areas.

Another source that heavily contributed to this project was a journal article detailing use of the Django Python framework when developing web applications. This helped me understand specific and popular use cases when Django is used, including popular companies that use it. Furthermore, it outlined the different architectural patterns Django uses, including Model View Controller; and it explained how different parts of the Django file convention correlate to the MVC design pattern (Thakur & Jadon, 2023).

3. PROJECT DESIGN

To help students avoid traveling to excessively busy locations across their university campus, we built the UniMap website to enable students to share in real-time with their peers how busy locations across campus are. This involved building a website using Python's Model View Controller Django backend framework along with JavaScript, and HTML/CSS for the front end of the website.

We first had to ensure all users could be properly authenticated. This authentication process involved creating an account with Google Cloud in order to use the Google Authentication API and allow users to sign in with their Google accounts. This created a necessary separation between admin and regular users, as only admin users can approve location suggestions on the platform.

Next, we utilized a similar process to gain access to the Google Maps API and display a map of locations on the website. Once we had established the API connections, we could then add the main functionalities of the project.

The first functionality was the ability to create and select certain locations on the map view. We used the Google Maps API and JavaScript to pinpoint certain areas of the map so users could rank the busyness or see the busyness level. These selections were then saved in a Django model built to keep track of locations. This included tracking the locations' latitude and longitude, as well as using separate markers for locations that were/were not verified by admins. Unverified locations are only visible to the admins and the person making the request.

Next, we built a model to track the busyness rankings of the locations. To build this feature, we used a simple Django poll model that tracked the location being ranked, which user made the ranking, one (least busy) to five (most busy) ranking, and when the time the ranking was made. This was an integral feature because each ranking lasted for only three hours before being deleted from the database to keep the busyness level at its most accurate state.

The user information is kept private in the backend to ensure that the user who makes the ranking does not skew results. Their vote is updated if they change it, ensuring that they only provide one ranking per location. This was an efficient manner to handle the problem of spam voting by a single user.

All of this information was then tallied to display as anonymous votes on a bar graph for a given location on the map, including the average busyness rating. We built this feature with JavaScript. To efficiently display the Django model results on the front end, the values had to be converted to JSON and then parsed in the JavaScript templates.

Finally, to publish the project, the website had to be hosted using Heroku and the database converted to a Postgres Database as opposed to an SQLite database. The final version of the ranking page is shown in Figure 1 below.

Figure 1: Voting Page on UniMap

4. RESULTS

Using Python's Django Framework, we developed a full stack application helping students identify hotspots across their university campus. Using UniMap, students can be more efficient when selecting places to study, gyms for workouts and restaurants for meals. This is a direct result of using the live ranking system to understand how crowded locations are. Knowing when a library is less crowded can help students find a more efficient study space and be more focused. Knowing what gyms are less busy can allow for more efficient workouts, as there is no need to spend extra time waiting for equipment to be free. Finding restaurants that have a lower rush allows students to get their food faster.

Most importantly, finding places to do work can be one of a student's most challenging tasks. The use of the UniMap application allows for time efficiency because students can use the application in a strategic manner to find an optimal study space. Furthermore, they no longer need to worry about walking twenty minutes across campus just to find that the location where they wanted to work is overcrowded. The UniMap tool can be used as a more accurate supplement to the preexisting Google busyness ranker because UniMap uses data from users currently at the locations and not an average data sample collection.

5. CONCLUSION

Living in an era of unprecedented connection and accessibility to the internet, sharing live information about the status of certain locations is inevitable. The result of this issue is the creation of UniMap, a website where users can rank and see the busyness of a given location, makes it possible to provide feedback others can use to make a decision on whether or not to visit that location.

Completing the UniMap project was important to help cater to the needs of students who have difficulty determining optimal times to visit facilities across their university campuses. Rather than using an average collection of busyness factors, they can rely on their peers for accurate, real-time information. The three-hour lifespan of the ranking, along with the application's ability to prevent spam votes from a single user helps provide accuracy to UniMap's polling system. Furthermore, UniMap is beneficial for users because they are able to request and suggest new locations to be added to the map. This suggestion feature is important for college students because there are niche places across a university campus that a map built by a big company may not recognize.

6. FUTURE WORK

To expand upon this work and make it more usable, the UniMap application should be transitioned into a mobile application. This makes it more user-friendly because students are more likely to consult their phones than their laptops when traveling to diverse campus locations. Furthermore, navigating to a URL is far more difficult than opening a simple app on one's phone.

While the accuracy of the busyness ranker is maintained by limiting users to one ranking per location, being in the general vicinity of the location is not enforced. In order to fix this issue, the application needs an update ensuring that the user can only submit a ranking if they are in that general area.

Last, UniMap only allows users to vote on buildings in locations, but not necessarily specific rooms or floors within the buildings. Adding a functionality differentiating certain domains specific to different areas of a building would be very useful.

REFERENCES

Google. (2024). Get information about busy areas from Google Maps. Google maps help. https://support.google.com/maps/answer/11323117?hl=en#:~:text=To%20determine%20popular%20times%2C%20wait,enough%20visits%20from%20these%20users.

Thakur, P., & Jadon, P. (2023). Django: Developing web using Python. 2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE). https://doi.org/10.1109/icacite57410.2023.10183246