NavigateLA 28

TL;DR

Mohammed Faizaan Muzawar
University of California, Riverside
Student ID: 862467277
mmuza004@ucr.edu

Ajit Singh
University of California, Riverside
Student ID: 862465889
asing349@ucr.edu

Aditya Gambhir
University of California, Riverside
Student ID: 862466997
agamb031@ucr.edu

Samara Miramontes
University of California, Riverside
Student ID: 862059976
smira009@ucr.edu

ABSTRACT

This project developed an app to assist tourists visiting the 2028 LA Olympics. It will provide recommendations for nearby attractions, restrooms, parks, and transit stops using spatial data and big data technologies such as Hadoop for spatial analysis. The project will address the challenges of dynamic routing and context-aware recommendations, integrating large GIS datasets to enhance the tourist experience.

1 INTRODUCTION

The Navigate LA28 project aims to deliver a state-of-the-art tourist assistance application for the 2028 Los Angeles Olympics. Tourists in unfamiliar cities often struggle with navigation—especially during large-scale events characterized by changing traffic conditions, shifting crowd distributions, and evolving transit schedules. By incorporating big data and spatial analysis technologies, the proposed application will provide personalized, recommendations for attractions, transit routes, and essential amenities such as restrooms and parks. Its adaptive routing and interactive capabilities are designed to respond dynamically to environmental changes, enhancing the overall tourist experience.

1.1 Applications

This application supports various scenarios for visitors attending the LA28 Olympics. Users can obtain customized recommendations for attractions, optimal transit routes, and nearby amenities, thereby maximizing the value of their visit. Continuous integration of data ensures that users remain informed about transit delays, traffic congestion, and crowd densities. In addition, app usage dashboards reveal valuable insights into tourist preferences, enabling stakeholders to fine-tune their services. Together, these features demonstrate the power of big data in addressing complex, real-world challenges while improving user satisfaction.

1.2 The Big Data Problem

Managing the extensive and dynamic datasets required by Navigate LA28 is inherently challenging. The application must process large-scale spatial information, including GIS maps, transit updates, and attraction data. Traditional static platforms are ill-equipped for efficiently analyzing data of this volume and complexity, necessitating scalable big data solutions. By leveraging technologies such as Hadoop for spatial analytics, the project ensures efficient data storage, retrieval, and processing. As a result, the system can deliver timely, accurate, and context-aware recommendations to users.

The remainder of this report is structured as follows: Section 2 defines the problem statement and objectives of the project. Section 3 presents a literature review of relevant technologies and methodologies. Section 4 details the datasets, data preprocessing steps, and task-specific implementations using tools like HDFS. Section 5 discusses evaluation results, highlighting query-based insights and performance metrics for dynamic features. Finally, Section 6 summarizes our findings and proposes directions for future work.

2 PROBLEM STATEMENT AND OBJECTIVES

The Navigate LA28 project addresses the complexity of guiding tourists through Los Angeles during the 2028 Olympics by employing big data techniques, spatial analysis, and personalized recommendations. Its primary goal is to enhance user experiences by integrating diverse data sources to provide immediate insights, route planning, and location-based guidance.

2.1 Dataset

The datasets used in this project can be accessed via the following link: Dataset Link

2.2 Objectives

The project's objectives include identifying the most visited attractions, transit stops, and amenities by examining user search trends and interaction data. Mapping the geographic distribution of popular destinations based on user locations will help uncover regional preferences and usage patterns.

In addition, the project seeks to develop a routing algorithm that can predict and generate an optimal visitation sequence for various attractions. This algorithm will factor in user preferences, travel time, and ensuring that recommendations remain personalized and dynamically updated.

Finally, the project aims to build interactive dashboards to classify user behaviors and analyze demographic trends. These visual tools will provide stakeholders with essential insights into user engagement, helping them improve overall service offerings.

3 LITERATURE SURVEY

3.1 Big Data and Spatial Data Processing

3.1.1 GIS Tools for Hadoop. This work introduces a suite of GIS tools optimized for the Hadoop ecosystem, enabling large-scale spatial data processing. Key functionalities include spatial indexing and geoprocessing operations like spatial joins and clustering. Integrating with HDFS and MapReduce, these tools facilitate scalable spatial analytics—a critical capability for Navigate LA28, which must handle extensive GIS datasets.

3.1.2 Spatial Hadoop. Spatial Hadoop extends Hadoop with native spatial primitives and operations. Features such as R-tree and Quadtree indexing improve query efficiency, while MapReduce optimizations enhance performance for tasks like nearest-neighbor searches and spatial joins. Navigate LA28 can leverage these capabilities to efficiently manage spatial queries and provide location-based recommendations.

3.2 Real-Time Data Integration

3.2.1 Dynamic GIS with Real-Time Data. This framework merges GIS with real-time data streams, enabling continuous spatial analysis. Through real-time data collection, in-memory processing, and live map updates, applications can reflect current conditions on the fly. For Navigate LA28, these techniques ensure timely and precise information on transit conditions and crowd densities.

3.2.2 Real-Time Data Processing with Big Data. This research explores frameworks like Apache Kafka and Spark Streaming for low-latency data handling. Methods such as windowing and event-driven processing support continuous analysis of live data. Applied to Navigate LA28, these technologies ensure the delivery of reliable, real-time updates on transit and event conditions.

3.3 Context-Aware and Personalized Recommendations

3.3.1 A Real-Time Context-Aware Recommendation System. This study outlines a system that adapts recommendations based on factors like location, time, and user preferences. By incorporating context detection, real-time processing, and dynamic user profiles, the approach can be adopted by Navigate LA28 to tailor attractions and route suggestions according to each tourist's unique situation.

3.3.2 Real-Time Personalized Recommendation Techniques. Focusing on user profiling and event-driven collaborative filtering, this research enables instant, personalized suggestions. Implementing these methods in Navigate LA28 will enhance user satisfaction through timely, context-relevant recommendations.

3.4 Geospatial Analysis and Map-Based UI

3.4.1 Mapping Techniques for Real-Time Data. This work discusses strategies such as dynamic layering and live data updates for visualizing real-time GIS information. These techniques allow maps to represent traffic, crowd density, and other shifting variables crucial for Navigate LA28's interactive interface.

3.4.2 JavaScript Libraries for Geospatial Analysis. Libraries like Leaflet and Mapbox enable responsive, real-time maps with customizable data overlays. These tools are well-suited for building Navigate LA28's user interface, allowing tourists to easily explore attractions, routes, and amenities on an intuitive, map-based platform.

4 METHODOLOGY

We followed the following implementation for the project.

4.1 Data Processing

In this case, the data was already relatively clean, so minimal preprocessing was required. The initial step involved identifying and removing duplicate entries in the bus stop dataset. These duplicates, representing the same stop recorded multiple times, could have influenced analyses such as total stop counts or spatial clustering. To address this, duplicate rows were checked by comparing unique identifiers like bus stop IDs, location coordinates, or stop names. Only the first occurrence of each duplicate was retained, ensuring that each stop was uniquely represented.

Next, centroids were calculated for polygon geometries within the park boundaries dataset. Instead of working directly with these often complex polygons, representing each park boundary as a single point provided a simpler way to conduct spatial analyses, such as determining proximity to bus stops. Before performing these operations, attention was given to ensuring that the coordinate reference systems (CRS) were consistent across all datasets, allowing for seamless spatial integration.

Apache Spark is utilized for its high-speed in-memory processing capabilities, which are crucial for handling large geospatial datasets and performing analytics.

4.1.1 Data Description. This project utilizes three primary datasets, each offering valuable spatial information about different aspects of Los Angeles. The first dataset, all_places.csv, comprises details about recreation spaces across the city. It includes attributes such as the name of the recreation space, its street address, and the latitude and longitude coordinates that precisely situate it within the urban landscape.

The second dataset, all_restrooms.csv, focuses on public restrooms in Los Angeles. These data points were sourced through web scraping and contain the name and address of each restroom facility, along with latitude and longitude coordinates that enable precise geospatial analyses of restroom accessibility and distribution.

The third dataset, busstops.shp, is a shapefile obtained from the LA Metro website. It contains comprehensive spatial data about all bus stops within Los Angeles, as well as the bus lines serving each stop. The attributes include a unique stop number, the bus line number, the direction of travel for that line, and the name of the bus stop. Latitude and longitude coordinates provide a geospatial reference, while the geometric point representation offers a consistent spatial framework. Notably, if a single bus stop serves multiple lines, there will be multiple records, each corresponding to a different line number.

4.1.2 Data Storage. For the storage of raw datasets and any outputs generated during the analysis, the Hadoop Distributed File

System (HDFS) will be employed. This approach ensures efficient handling of large spatial datasets and facilitates scalable, distributed computations.

4.2 Query-Based Features

The queries developed for this project are designed to help users find spatial information relevant to their current position and interests in Los Angeles. By integrating SparkSQL for efficient large-scale data processing and GeoPandas for precise spatial analysis, these queries deliver insightful and actionable results. The following sections describe each query and its purpose.

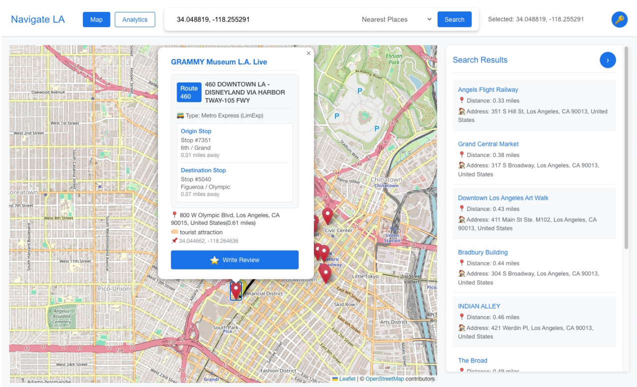

Figure 1: Nearest Attractions

4.2.1 Query 1: Find Nearest "n" Attractions from User Location. This query enables users to identify the nearest "n" recreational attractions from their current location, where "n" is a user-defined parameter. Implemented using SparkSQL, it efficiently processes spatial data to calculate distances between the user's location and various attractions. As a result, users can quickly discover and prioritize nearby recreation spaces in Los Angeles that cater to their specific interests or constraints.

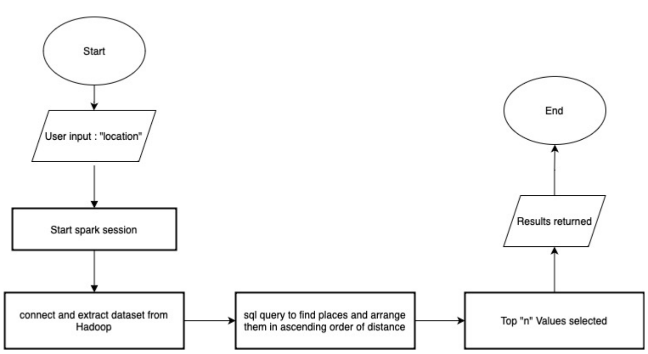

Figure 2: Query Architecture

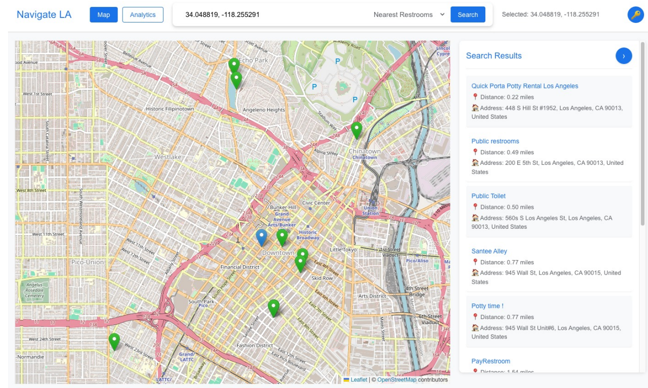

Figure 3: Nearest Restrooms

4.2.2 Query 2: Find Restrooms Near User Location or Attraction Location. For individuals searching for public restrooms, this query identifies the facilities closest either to their current location or a chosen recreational attraction. By leveraging GeoPandas for spatial analysis and SparkSQL for large-scale data handling, the query provides an effective solution to locate restroom accessibility in proximity to key points of interest. This combination ensures that users can easily find necessary amenities without extensive manual searching.

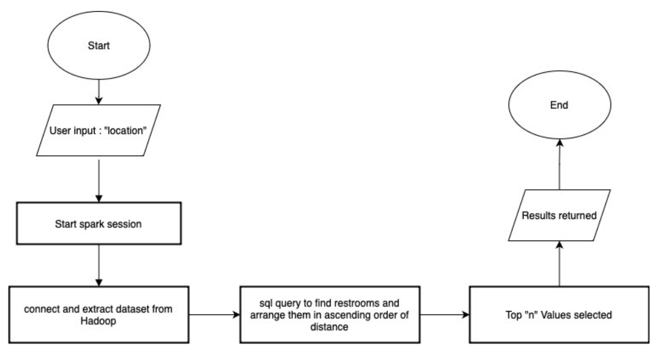

Figure 4: Query Architecture

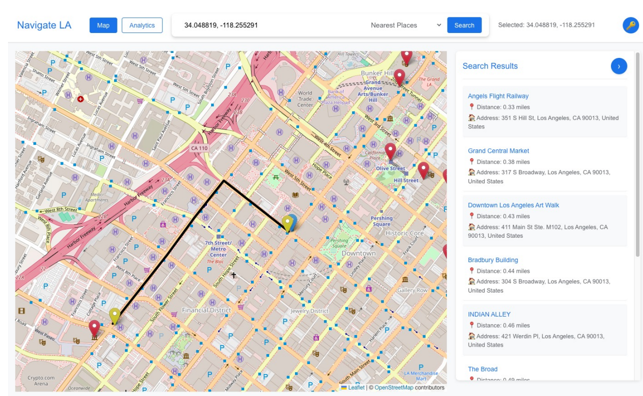

Figure 5: Bus Routes

4.2.3 *Query 3: Find Bus Stops and Direct Bus Route within a 2-Mile Radius.*
This query focuses on public transportation convenience. It discovers bus stops within a 2-mile radius of the user's current position and connects them directly to bus stops located within a 2-mile radius of a selected historic, recreational, or tourist area. Through the integration of spatial data from the busstops.shp dataset, SparkSQL for robust data management, and GeoPandas for spatial operations, users can identify accessible transit options that simplify travel between significant locations in Los Angeles.

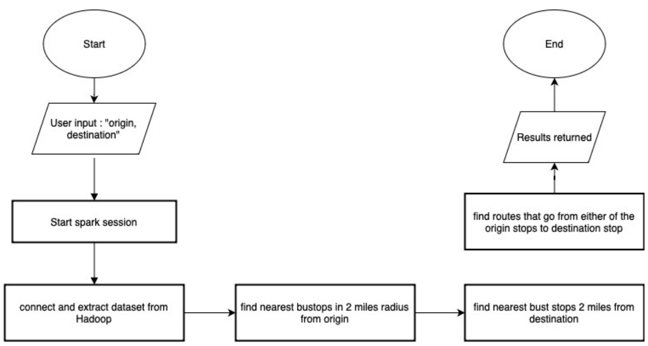

Figure 6: Query Architecture

4.3 Optimization and Analysis Features

The following sections describe the big data optimizations and analysis features designed to enhance the overall user experience. By leveraging advanced spatial computing methods, routing algorithms, and interactive dashboards, these features aim to improve both the functionality and the intelligence of the application.

4.3.1 *Creating an Attraction Visit Plan.*
This component focuses on determining an optimal sequence of attractions for tourists to visit, starting from a specified location. To achieve this, GIS datasets—including attraction locations, park boundaries, and transit stops—are integrated and prepared for analysis. Distances and travel times between attractions are calculated using geospatial libraries such as GeoPandas and Shapely, while user preferences, including attraction types, estimated visit durations, and starting points, are also incorporated. The problem is solved using routing algorithms like A* for identifying shortest paths, and the resulting route is presented dynamically on an interactive map interface built with tools like Leaflet or Mapbox integrated into React.

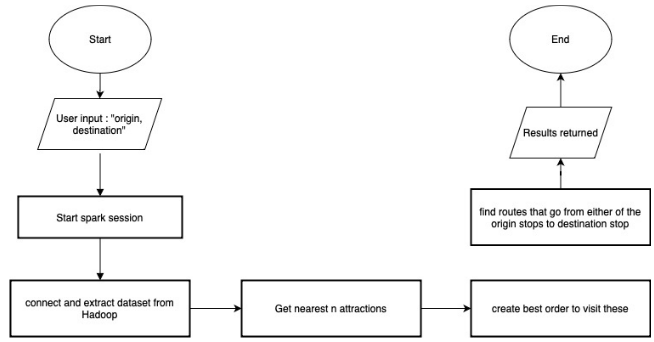

Figure 7: Query Architecture

4.3.2 *App Usage Dashboards.*
The second feature provides analytical insights into app usage, highlighting popular attractions and examining user demographics. Data preparation involves collecting app usage logs, including search history, geographic user distributions, and session information. Key metrics, such as the most searched attractions, frequently queried regions, and common navigation patterns, are derived from these data. The implementation leverages React for building the user interface and charting libraries like Chart.js for visualizing data. Interactive filters are included to segment metrics by demographics, regions, and time frames, enabling the presentation of user-friendly statistics—such as total searches, session durations, and user preferences—on dynamic dashboards. These insights help stakeholders understand user behavior and optimize both content delivery and user engagement strategies.

4.4 Web Application

The web application supporting this project integrates a robust back-end, a dynamic front-end, and a big data infrastructure that leverages Hadoop Distributed File System (HDFS) and Spark. All components are containerized with Docker, ensuring consistency, scalability, and ease of deployment. The system has been designed to handle geospatial queries, provide updates, and deliver predictive analytics results to the user interface.

4.4.1 *Back-end.*
The back-end, developed using FastAPI, manages incoming requests and data processing tasks. Environment variables for database connections, API secret keys, and other configuration details are stored securely in an .env file and loaded at runtime. The Docker configuration defined in docker-compose.yml exposes port 8000 for API access. The back-end offers APIs to handle geospatial data queries and predictive analytics, making it possible for the front-end to access updates. For each query type—such as locating nearby tourist attractions or determining optimized travel routes—dedicated endpoints are available in the routes/ directory. Predictive analytics tasks, carried out by Spark, store their results in a format that these endpoints can easily retrieve and return as JSON responses. The back-end also manages user authentication, validates inputs, and coordinates with Hadoop and Spark for distributed geospatial computations.

API Design.

- RESTful API endpoints are defined for various functionalities such as route planning, point-of-interest searches, and user authentication.
- The API adheres to modern best practices for security and scalability.

Data Flow Management.

- The backend interacts with Spark to process user requests and retrieve relevant data from HDFS.
- FastAPI's asynchronous capabilities ensure that multiple requests are handled concurrently without compromising performance.

Error Handling.

- Robust error handling mechanisms ensure smooth user experience and system reliability.
- Logging and monitoring are implemented to track and resolve issues promptly.

4.4.2 Front-end. The front-end is built with React, situated in the client/ directory, and employs Redux for state management, with corresponding slices defined in the src/slices/ directory. Interactive map-based visualizations, rendered using React Leaflet, allow users to engage with geospatial data seamlessly. The front-end's homepage introduces the project's functionalities, guiding users through various tools to discover tourist spots, navigate optimal routes, and analyze data-driven insights. Instead of loading all data at once, the front-end makes targeted API calls in response to user actions, such as applying filters or selecting specific map locations. This on-demand data retrieval enhances responsiveness and ensures a more efficient user experience. Filters, including radius, categories, and date ranges, help users customize their results, which are then displayed dynamically on the interface.

User Interface Design.

- The interface features an intuitive layout with interactive maps, search functionality, and route visualization.
- Components are modular, allowing for future enhancements and updates.

Integration with Backend.

- React communicates with the FastAPI backend using RESTful API calls.

Cross-Platform Compatibility.

- The frontend is designed to work seamlessly across various devices and screen sizes, ensuring accessibility for all users.

4.4.3 Hadoop Distributed File System (HDFS). HDFS serves as the backbone for storing geospatial data, configured with a Hadoop namenode container defined in the docker-compose.yml file. It relies on the bde2020/hadoop-namenode image and provides accessibility via ports 9870 for the web UI and 9000 for data operations. Configuration files are located in the hadoop/conf/ directory. Storing geospatial data in HDFS and partitioning it for parallel processing enables efficient batch queries and aggregations using Spark. This approach significantly reduces latency, especially when analyzing trends such as user activity over time.

4.4.4 Spark. Spark, configured with a master container based on the bitnami/spark image and defined within docker-compose.yml, is exposed on port 8080 for monitoring and management. Spark's primary function is to handle large-scale geospatial datasets for predictive tasks. By identifying tourist hotspots, optimizing travel routes, and performing other analytical jobs, Spark processes and refines data stored in HDFS. The resulting insights are then written back to HDFS for retrieval by the back-end APIs, ensuring that front-end users always have access to up-to-date, data-driven recommendations.

4.4.5 Docker. All system components—front-end, back-end, Hadoop, and Spark—are containerized with Docker. The docker-compose.yml file orchestrates these services, connecting them through the navigate_la_28 network. Volume mappings maintain persistent storage for logs, configurations, and data. This containerized approach allows individual services to be rebuilt or restarted independently, minimizing disruptions. Debugging is facilitated by the ability to access logs through simple commands like docker-compose logs <service_name>.

Container Structure.

- Separate Docker containers are created for the frontend (React), backend (FastAPI), Hadoop, and Spark components.
- Each container is configured with the necessary dependencies and settings.

Orchestration with Docker Compose.

- Docker Compose manages the lifecycle of all containers, enabling easy startup, shutdown, and scaling.
- The `docker-compose.yml` file defines the interconnections between components.

Overall, this integrated web application architecture leverages containerized microservices, big data frameworks, and modern web technologies to deliver responsive, data-driven geospatial experiences.

4.5 Workflow Integration

The integration of all components follows a well-defined workflow, ensuring smooth communication and functionality:

(1) **User Interaction:** Users access the application through the React frontend to search for routes, locations, or points of interest.
(2) **Request Handling:** The frontend sends user requests to the FastAPI backend via API calls.
(3) **Data Processing:** The backend forwards the request to Spark, which processes the data stored in HDFS and returns the results.
(4) **Response Delivery:** The backend formats the results and sends them to the frontend, where they are displayed interactively to the user.

5 RESULTS AND PERFORMANCE EVALUATION

In this section, we present the results of our analyses and evaluate the system's performance. The focus is on assessing query-based tasks designed to provide spatial insights, evaluating both accuracy and efficiency. Each query-based scenario is accompanied by clear objectives, metrics, and visualization strategies, ensuring that our evaluation covers usability, scalability, and alignment with ground truth or benchmark data.

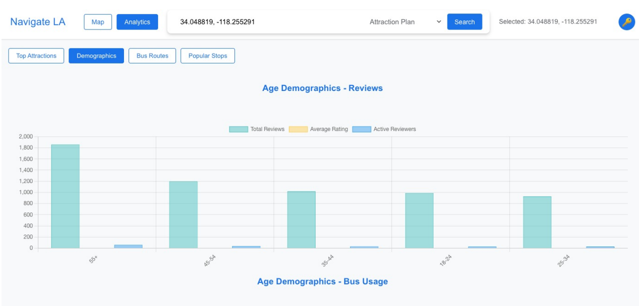

Figure 8: Age Demographics

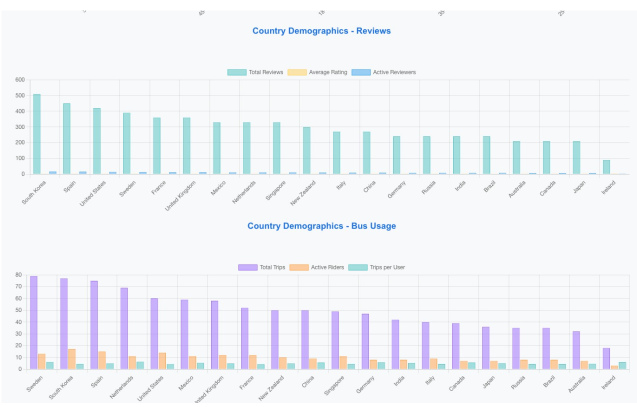

Figure 9: Country Demographics

Figure 10: Bus Line Popularity and Frequent Routes by Stop

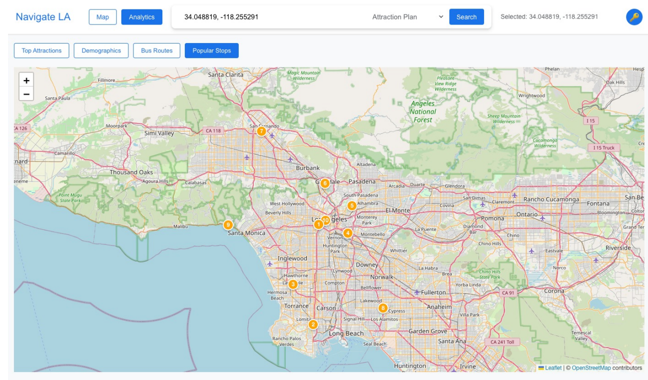

Figure 11: Popular Bus Stops

5.1 Query-based Tasks

The following subsections detail the methodology and outcomes of the query-based tasks performed. Visual representations, including map-based overlays, help verify the spatial relationships and ensure that the queries meet user expectations in real-world scenarios.

5.1.1 Q1: Find Nearest "n" Attractions from User Location. The primary objective of this query is to determine how accurately and quickly the system can retrieve the nearest "n" attractions relative to a user's location. To achieve this, we measure response time, ensuring the query executes efficiently. Additionally, we assess accuracy by validating calculated distances against benchmark datasets or known ground truth, and we test scalability by evaluating performance under varying numbers of attractions and users. Visualization involves placing markers on a map to confirm the correct spatial relationships between the user's location and the identified attractions.

5.1.2 Q2: Find Restrooms Near User or Attraction Location. In this task, we aim to assess the effectiveness of identifying restrooms in close proximity to either a user's current position or a specified attraction. Precision and recall metrics are employed to ensure the system correctly identifies nearby facilities, while response time evaluations gauge the speed of handling extensive restroom datasets. To further measure the feature's usability, user tests are conducted to confirm that the restrooms meet accessibility requirements. The results are visualized by displaying restroom locations as pins on a map, overlaid onto user or attraction positions, facilitating an intuitive evaluation of spatial alignment and ease of use.

5.1.3 Q3: Find Bus Stops and Direct Bus Routes. This query focuses on determining the efficiency and accuracy of locating bus stops and identifying direct transit routes that connect areas surrounding both the user's location and chosen attractions. We verify route feasibility by cross-referencing identified routes and bus stops with official transit maps and schedules. Response time is tracked to ensure timely delivery of route options within the specified two-mile radius. To gauge practical value, user feedback is collected regarding the utility of the suggested routes. Visualization involves presenting bus stops, user or attraction vicinities, and transit routes on an interactive map, enabling stakeholders to confirm spatial accuracy, interpretability, and overall usefulness of the provided recommendations.

5.2 Optimization and Analysis Tasks

In evaluating the Attraction Visit Plan feature, several criteria will be considered to determine its efficacy. These include response time, route accuracy when compared against expert-recommended paths, and user satisfaction based on surveys and feedback. By incorporating user inputs such as preferred attraction types and visit durations, the feature can be tested for its capacity to deliver tailored travel plans. Visual representations of these optimized routes on interactive maps will further highlight their practicality and contribute to a positive user experience.

The App Usage Dashboards will be assessed in terms of accuracy, usability, and the meaningfulness of the insights they deliver. Key metrics encompass the fidelity of displayed data relative to raw application logs, the relevance of insights to various stakeholders, and system load times for refreshing dashboard information. User feedback on the dashboards' navigability and filtering options will help gauge overall effectiveness. Data visualizations, including bar charts, pie charts, and demographic heatmaps, will be employed to illuminate usage patterns, user distributions, and frequently searched attractions, ensuring that the dashboards present actionable intelligence in a clear and accessible manner.

5.3 Evaluation Framework

The evaluation framework for the Navigate-LA-28 system ensures all components function seamlessly, covering the frontend, backend, data storage, processing, and containerization. The **frontend** is tested to confirm proper loading, with all UI components rendering without errors, and API connectivity verified to ensure smooth communication with the backend. Key **navigation features**, such as location search and route visualization, are evaluated for usability. On the backend, the **FastAPI server** is assessed for successful initialization and correct API responses to user queries like route suggestions and location-based data. The **data handling** processes, including **Hadoop HDFS initialization** and the `move_to_hdfs.sh` data upload script, are tested to ensure accessibility and proper storage of datasets. Similarly, **Spark initialization** and **job execution** are validated to confirm efficient processing of test queries, including filtering and aggregation tasks.

Additionally, the framework evaluates the containerization and orchestration setup. **Docker services** are tested to ensure all containers (frontend, backend, Hadoop, Spark) start correctly and communicate without issues. The **environment variables** are checked for proper configuration in the `.env` files. All components successfully passed these evaluations, ensuring the system is robust and reliable.

Test ID	Component	Evaluation Criteria	Result
1.1	Frontend Loading	React frontend loads without errors, displays the homepage, and all UI components render properly.	Pass
1.2	API Connectivity	Frontend successfully communicates with backend API endpoints (e.g., fetch location data).	Pass
1.3	Navigation Features	Users can search locations, view routes, and interact with the map without errors.	Pass
2.1	Backend Initialization	FastAPI server starts correctly without error messages.	Pass
2.2	API Response Validation	API endpoints return correct and timely responses for user requests (e.g., route suggestions).	Pass
3.1	Hadoop HDFS Initialization	Hadoop services start successfully, and the HDFS is accessible.	Pass
3.2	Data Upload to HDFS	The `move_to_hdfs.sh` script successfully uploads test data to HDFS.	Pass
4.1	Spark Initialization	Spark cluster initializes correctly, and jobs can be submitted without errors.	Pass
4.2	Spark Job Execution	Spark processes test queries successfully (e.g., filtering and aggregation tasks).	Pass
5.1	Docker Services	All Docker containers (frontend, backend, Hadoop, Spark) start correctly and are interconnected.	Pass
5.2	Environment Variables	All required environment variables are correctly configured in `.env` files for each service.	Pass
6.1	Full System Workflow	The system completes the end-to-end workflow: user interaction → backend API → Spark processing → frontend display.	Pass

Table 1: Evaluation Framework for Navigate-LA-28

6 CONCLUSION

In conclusion, this project aims to support future tourists visiting Los Angeles during the 2028 Olympic Games by providing them with integrated, location-based insights and services. Recognizing the city's complexity and sprawling layout, our platform aggregates diverse datasets—from recreational attractions and restrooms to transit stops—and makes them accessible through a unified query interface. By leveraging advanced geospatial analyses, data processing tools, and interactive visualizations, we empower visitors to discover optimal routes, identify nearby amenities, and ultimately navigate Los Angeles with greater ease and confidence.

7 AUTHOR CONTRIBUTIONS

7.1 Mohammed Faizaan Muzawar

Focused on integrating the queries with the backend and building the UI for all the functions.

7.2 Ajit Singh

Focused on writing queries, making flow diagrams and initial ideation and also solving dependencies issues for some tasks.

7.3 Aditya Gambhir

Focused on getting queries to work and providing information for the report.

7.4 Samara Miramontes

Focused on writing reports, helping with queries, and other general tasks.

REFERENCES

[1] ESRI. *GIS Tools for Hadoop.* Accessed: November 2024. Available: https://esri.github.io/gis-tools-for-hadoop/
[2] SpatialHadoop. *SpatialHadoop.* Accessed: November 2024. Available: https://spatialhadoop.cs.umn.edu/
[3] Cloudera Community. *Geospatial Data Analysis in Hadoop.* Accessed: November 2024. Available: https://community.cloudera.com/t5/Community-Articles/Geospatial-Data-Analysis-in-Hadoop/ta-p/244881

[4] Aji, A. M., Wang, F., Vo, H. T., Lee, R., Liu, Q., Zhang, X., and Saltz, J. H. 2013. Hadoop-GIS: A High Performance Spatial Data Warehousing System over MapReduce. *Proceedings of the VLDB Endowment*, 6(11), 1009–1020.

[5] Eldawy, A., and Mokbel, M. F. 2015. The Era of Big Spatial Data. *The VLDB Journal*, 24(4), 731–760.

[6] Bagnoli, C., Comandini, V., Montangero, M., Morzenti, A., and Rossi, M. 2020. Big Spatial Data Analytics on Cloud-Based Platforms for Mobility Analysis: Performance Evaluation and Design Guidelines. In *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 1029–1038.

[7] Kumar, D., Gade, K., Ho, S., and Mithun, M. 2016. Large Scale Geospatial Analysis Using Apache Spark. *arXiv preprint arXiv:1602.09076*. Accessed: November 2024.

[8] Xie, H., Yang, J., and Kou, J. 2020. A Novel Distributed Geospatial Data Storage and Processing System Based on Apache Spark. In *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 3801–3809.

[9] Garcia-Garcia, A., Moreno-Fernandez, A., Martinez-Sober, M., Jimenez-Molina, J. A., and Garcia-Garcia, J. 2013. Enhancing Urban Traffic Management with Big Data Technologies. *arXiv preprint arXiv:1311.7676*. Accessed: November 2024.

[10] Fang, X., and Liu, X. 2020. Distributed Spatial-Temporal Big Data Processing for Smart Cities: A Case Study Using Hadoop. In *Geographic Information Science, Lecture Notes in Computer Science*, vol. 11886, 49–64.

[11] Yu, M., Wu, Q., Ma, S., Chen, F., and Zhuang, Y. 2018. Big Data in Transportation: Applications and Challenges. *Transportation Research Part C: Emerging Technologies*, 86, 2–17.

[12] ESRI. *Data Science and Analytics for GIS Professionals*. ESRI User Conference. Accessed: November 2024. Available: https://proceedings.esri.com/library/userconf/proc13/tech-workshops/tw_354.pdf

[13] ESRI. *Big Data Analytics and GeoAnalytics Tools in ArcGIS*. ESRI User Conference. Accessed: November 2024. Available: https://proceedings.esri.com/library/userconf/proc13/tech-workshops/tw_357.pdf

[14] Bian, X., and Wang, C. 2020. Intelligent GIS Data Collection and Analysis Using Big Data Techniques. *Journal of Ambient Intelligence and Humanized Computing*, 11(10), 4149–4163.

[15] Ahmad, M. 2024. Connecting the Dots: Harnessing OpenStreetMap for Big Data Analytics and Market Insights. *Big Data Analytics Techniques for Market Intelligence*, pp. 19.

[16] Quvvatov, B. 2024. SQL Databases and Big Data Analytics: Navigating the Data Management Landscape. *Development of Pedagogical Technologies in Modern Sciences*, 3(1), 117–124. Retrieved from http://www.econferences.ru/index.php/dptms/article/view/11708.

[17] Udeh, H. A., Orieno, O. H., Daraojimba, O. D., Ndubuisi, N. L., and Oriekhoe, O. I. 2024. Big Data Analytics: A Review of Its Transformative Role in Modern Business Intelligence. *Computer Science IT Research Journal*, 5(1), 219–236. https://doi.org/10.51594/csitrj.v5i1.718.

[18] Torre-Bastida, A. I., Del Ser, J., Laña, I., Ilardia, M., Bilbao, M. N., and Campos-Cordobés, S. 2018. Big Data for Transportation and Mobility: Recent Advances, Trends and Challenges. *IET Intelligent Transport Systems*, First published: 17 July 2018. https://doi.org/10.1049/iet-its.2018.5188.

[19] Unknown. 2020. Big Data Analytics in Production and Operations Management: Current Trends and Future Directions. *International Journal of Production Economics*. https://doi.org/10.1016/j.ijpe.2020.107868.

[20] Ghofrani, F., He, Q., Goverde, R. M. P., and Liu, X. 2018. Recent Applications of Big Data Analytics in Railway Transportation Systems: A Survey. *Transportation Research Part C: Emerging Technologies*, 90, 226–246. https://doi.org/10.1016/j.trc.2018.03.010.

[21] Welch, T. F., and Widita, A. 2019. Big Data in Public Transportation: A Review of Sources and Methods. *Transport Reviews*, 39(6), 795–818. https://doi.org/10.1080/01441647.2019.1612841.