Team JSR

Members: Remitio

Project Description`

Supply Scout is a supply-finding app that provides real-time inventory updates for nearby stores. It categorizes stores based on the type of supplies they carry and integrates with maps to help users locate their desired items quickly and efficiently. The app features an intuitive interface, ensuring users can easily navigate and find the supplies they need without confusion.

Requirements Summary:

1.UI Design- The ease of use and aesthetic appeal of the app's interface, including navigation, search functionality, and overall user experience.

2.GPS - The precision of the app's GPS and map integration to show accurate user location and nearby stores.

3. Performance- The speed and reliability of the app, including loading times, responsiveness to user actions, and stability under various conditions.

4. User Registration and Login - Users should be able to create accounts and log in using email, social media, or phone number.

Design Space

Important Characteristics of the Task Performed by Users:

What requirements may be difficult to realize?

Integrating real-time inventory updates from a diverse range of stores may be challenging due to the lack of standardized APIs and varying levels of technological infrastructure across different retailers. Ensuring that every store can provide up-to-date inventory information might require developing flexible integration methods and potentially accommodating manual updates for smaller stores without automated systems. This complexity is further heightened by the need to maintain synchronization without overloading the system, making it a technically demanding requirement to implement effectively.

What are some tradeoffs that you should or did explore?

A significant tradeoff involves balancing real-time data accuracy with system performance. Continuously updating inventory information ensures users have the most current data but can place a substantial load on the system, potentially affecting overall performance and user experience. To address this, strategies such as optimizing data refresh rates and employing caching mechanisms were explored to maintain a balance between providing up-to-date information and ensuring the app remains responsive and efficient. Additionally, finding the right balance between a feature-rich interface and usability was crucial to avoid overwhelming users while still offering comprehensive functionality.

Which tasks will be easiest to support?

User registration and login functionalities are among the easiest tasks to support due to the availability of established methods and frameworks. Implementing secure and efficient authentication processes using services like OAuth can streamline this aspect of development. Furthermore, store categorization is relatively straightforward, as it involves organizing stores based on predefined types and item categories, which can be managed effectively using a structured database system.

Which tasks will be hardest to support?

The most challenging task to support is real-time inventory updates, which require continuous synchronization with a wide variety of store inventory systems that may differ significantly in their capabilities and formats. Ensuring accurate, up-to-date information in real-time necessitates robust integration solutions tailored to each system's specifics. Additionally, maintaining precise and reliable location services poses difficulties, particularly in environments with poor GPS signals or dense urban areas. Implementing advanced tracking technologies and algorithms is necessary to provide users with accurate location data and seamless navigation.

Design Summary

In developing Supply Scout, we considered various design alternatives to balance complexity, usability, and scalability. One key decision involved inventory updates. We debated between fully automated updates through direct integration with store management systems and manual updates by store staff. While automation ensures accuracy and real-time data, it is technically challenging due to the diversity of store systems. Conversely, manual updates are simpler but

can be unreliable. We opted for a hybrid approach, combining both methods to balance feasibility and accuracy.

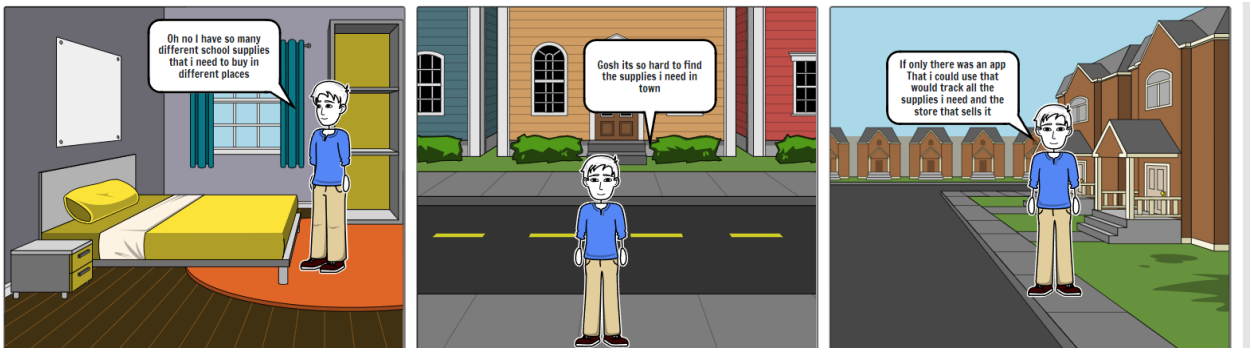
We also weighed a comprehensive feature set against a minimal viable product (MVP). A feature-rich interface with advanced search filters and extensive customization would offer more functionality but risked complicating the user experience. An MVP focusing on core functionalities such as basic search, location integration, and essential inventory updates ensures a user-friendly interface and quicker deployment. We chose the MVP approach initially, planning to add features incrementally based on user feedback to maintain balance and usability.

For location services, we considered third-party map integration versus developing a custom mapping solution. Third-party services like Google Maps provide reliable and accurate data with quicker implementation, despite potential dependency and cost issues. Custom mapping offers more control but requires significant development time and expertise. We opted for third-party integration to leverage existing infrastructure and focus on other core functionalities.

Regarding data storage, we evaluated centralized versus distributed approaches. Centralized storage simplifies data management and security but can become a bottleneck as the user base grows, affecting performance. Distributed storage offers better scalability and reliability but introduces complexity in synchronization. Initially, we chose centralized storage for its simplicity and data integrity, with plans to explore distributed solutions as the app scales.

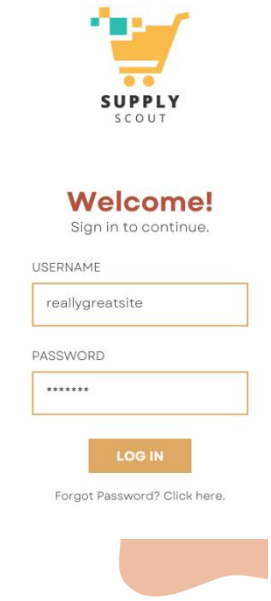
The Designs

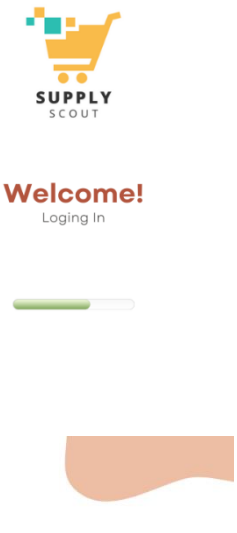
Pov Buyer:

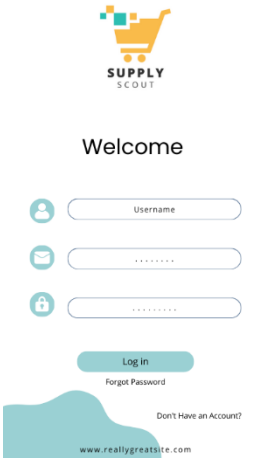


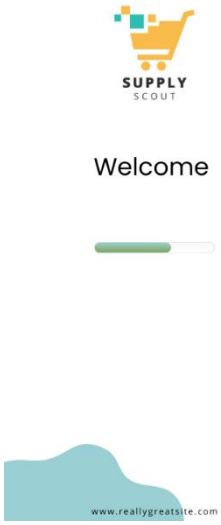
Pov Seller:

Log in Registration:









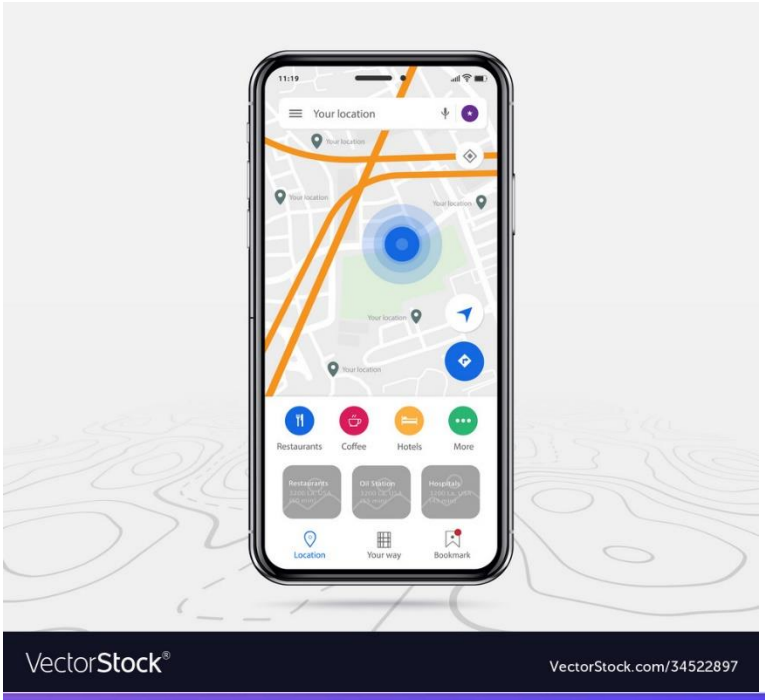
Advantages:

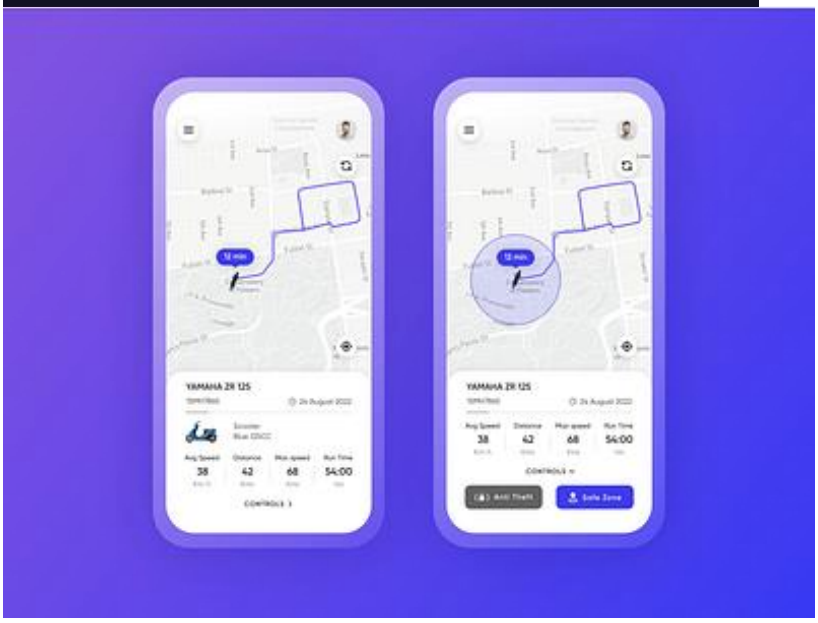
- Easy to look at
- Clean

Disadvantages

- Can still be improved upon and there are better options

Gps Landing page for users:





Advantages:

- Easy to navigate
- User friendly
- Clear use of GPS

Disadvantages:

- Not much design
- May be hard for others to look at
- Prone to misclicks

Requirement changes:

During the design process of Supply Scout, we modified, added to, and removed several requirements and usability criteria based on practical considerations and feedback. Initially, we aimed for fully automated real-time inventory updates. However, due to the diverse range of store inventory systems and the technical challenges of achieving full automation, we adopted a hybrid approach. This combined automated updates for technologically capable stores with manual updates for others, ensuring broader compatibility and more reliable data.

We also re-evaluated our initial plan for a comprehensive

feature set that included advanced search filters and extensive customization options. To ensure a faster deployment and a user-friendly experience, we decided to focus on developing a Minimal Viable Product (MVP) with essential features such as basic search, location integration, and core inventory updates. This allowed us to prioritize core functionalities and ensure the app remained intuitive for users.

Several new requirements emerged during the design process. For instance, we added a manual inventory update interface for stores. This was necessary to accommodate stores that do not have automated systems but still need to provide accurate inventory data. Additionally, we incorporated an in-app notification system to keep users informed about inventory alerts and promotions within the app environment, enhancing user engagement and experience. Recognizing the importance of continuous improvement, we also added a user feedback mechanism to gather insights and iteratively refine the app based on real user needs.

Conversely, we removed some initial requirements to streamline development and focus on delivering a functional product. We decided against developing a custom mapping solution due to the complexity and time

required, opting instead for reliable third-party map integration that offered quicker implementation and robust functionality. We also scaled back extensive user customization options, choosing to concentrate on essential features and maintaining a straightforward interface.

These requirement changes arose from practical considerations, technical constraints, and the goal of delivering a functional, scalable, and user-friendly app that meets essential user needs while allowing for future enhancements based on user feedback and evolving requirements.