Boston University
Electrical & Computer Engineering
EC464 Capstone Senior Design Project

User's Manual

Pharma AI Procurement Optimizer (PAPO)

Submitted to

Gordon Hubbard
Beth Israel Deaconess Medical Center
330 Brookline Ave
Boston, MA 02215
(617) 667-8862
ghubbard@bidmc.harvard.edu

by

Team 6
PAPO

Team Members

Bora Bulut bbulut@bu.edu
Taha Ababou hababou@bu.edu
Manuel Segimon manuelsp@bu.edu
Joel Akerman akermanj@bu.edu
Zaiyan Muhammad monem@bu.edu

Submitted: 04/15/2024

PAPO User’s Manual

Table of Contents

Executive Summary

The nature of the problem at hand revolves around the client's manual and time-consuming drug procurement processes, characterized by the need for constant evaluation of alternatives based on price, quantity, and packaging. Additionally, the criticality of specific drugs necessitates a precise and timely restocking strategy. Our final deliverable aims to address these challenges comprehensively through an AI-driven procurement solution.

This solution comprises a three-tiered technical approach: continuous stock monitoring and replenishment based on live feed analysis, intelligent recommendations for alternative drug replacements drawn from the pharmacy's database, and the development of a user-friendly interface based on client feedback. The proposed model offers innovative features such as dynamic stock monitoring, intelligent alternative recommendations prioritized by cost, dosage preference, and packaging, and a user-centric interface tailored to streamline the buyer's decision-making process.

This solution is poised to significantly optimize the client's drug procurement workflow, ensuring efficiency, accuracy, and responsiveness to critical drug restocking requirements.

1 Introduction

Welcome to the user manual for the Pharma AI Procurement Optimizer system, developed by Team 6 in EC464 at Boston University's Electrical & Computer Engineering department. This guide is designed to assist you in navigating and leveraging our system to enhance your pharmacy procurement processes. Our solution integrates cutting-edge technology with practical applications, aiming to optimize pharmaceutical procurement and management.

The Pharma AI Procurement Optimizer system was conceived in response to the growing need for pharmacies to improve efficiency, reduce costs, and ensure the availability of necessary medications. The project leverages advanced algorithms and database technologies to provide a seamless and user-friendly platform for managing pharmaceutical data. The heart of our system is a sophisticated cost-effective algorithm that helps identify the most financially optimal drug replacements available on the market.

Our platform utilizes a robust online database to securely store user information while ensuring real-time updates and high availability. The integration of API functionality facilitates essential user interactions such as registration, login, profile management, and more, enhancing the overall user experience.

From the user's perspective, our system offers several standout features, including a cost-effective algorithm that automatically calculates the weighted average prices of drug replacements, helping users find cost-effective alternatives quickly. The algorithm supports adjustments based on three different price segments, providing tailored results that align with budget constraints. The system features an intuitive front-end design, including a straightforward login page and a comprehensive main dashboard where users can input drug codes and receive a list of suitable replacements in descending order of suitability. Secure and reliable access ensures the integrity and security of user data with encrypted login credentials and real-time database updates, and with advanced logging capabilities, every user action is logged securely, aiding in troubleshooting and enhancing security by monitoring login attempts and user activities.

While the system is designed to be as robust and user-friendly as possible, users may encounter several challenges. The first one is data security risks. Although our database employs advanced security measures, the sensitive nature of pharmaceutical data requires users to maintain strong cybersecurity practices to prevent unauthorized access. In addition, due to algorithm complexity, users may need some initial assistance to fully understand how to manipulate the algorithm's settings for optimal results. Lastly, ensuring compatibility with existing hardware and software infrastructure may require technical adjustments.

The remainder of this manual is organized into several sections to help you effectively utilize the Pharma AI Procurement Optimizer system. These include System Overview and Installation, Operation of the Project, and Technical Background. Each section is designed to provide comprehensive information and support, ensuring you can maximize the benefits of the Pharma AI Procurement Optimizer system in your operations.

2 System Overview and Installation

2.1 Overview block diagram

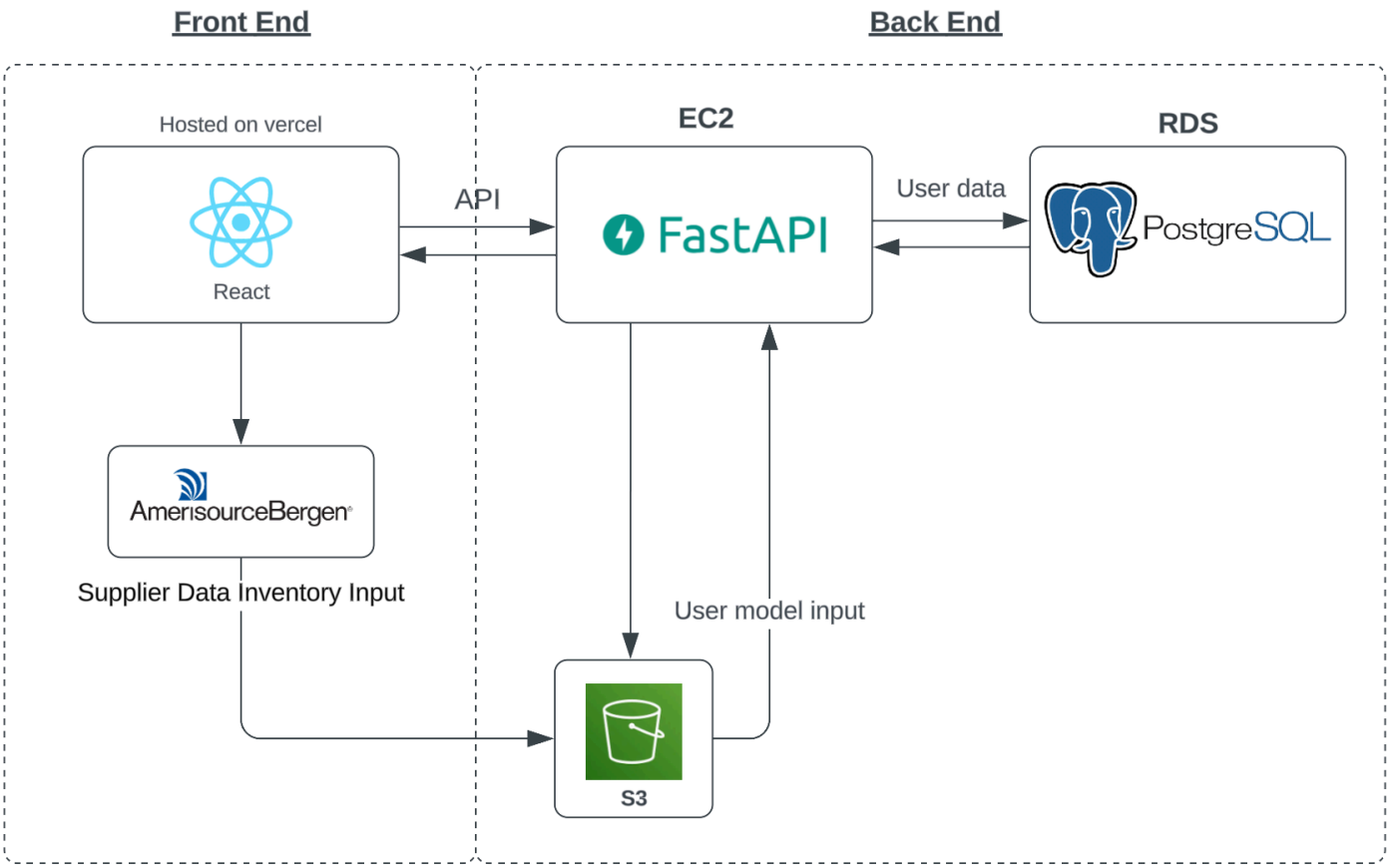

Figure 2.1 The block diagram represents the system architecture of the Pharma AI Procurement Optimizer, showing a front-end built with React and hosted on Vercel for user interaction, a back-end comprising a FastAPI running on an AWS EC2 instance for API management, a PostgreSQL database on AWS RDS for storing user data, and an AWS S3 bucket for storing user model inputs. It also indicates integration with supplier data from AmerisourceBergen for inventory management.

2.2 User Interface

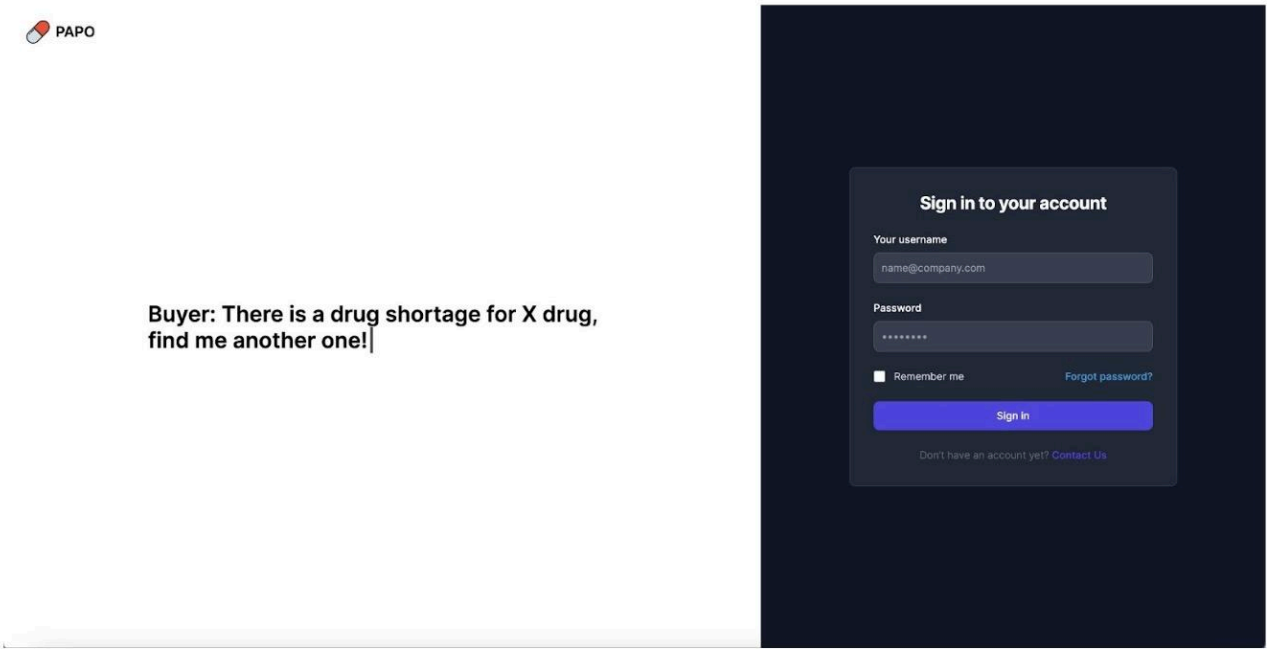

Figure 2.2 PAPO's secure login page, featuring a clean and straightforward design for user authentication. Users can enter their credentials to access the system's comprehensive drug procurement and management tools.

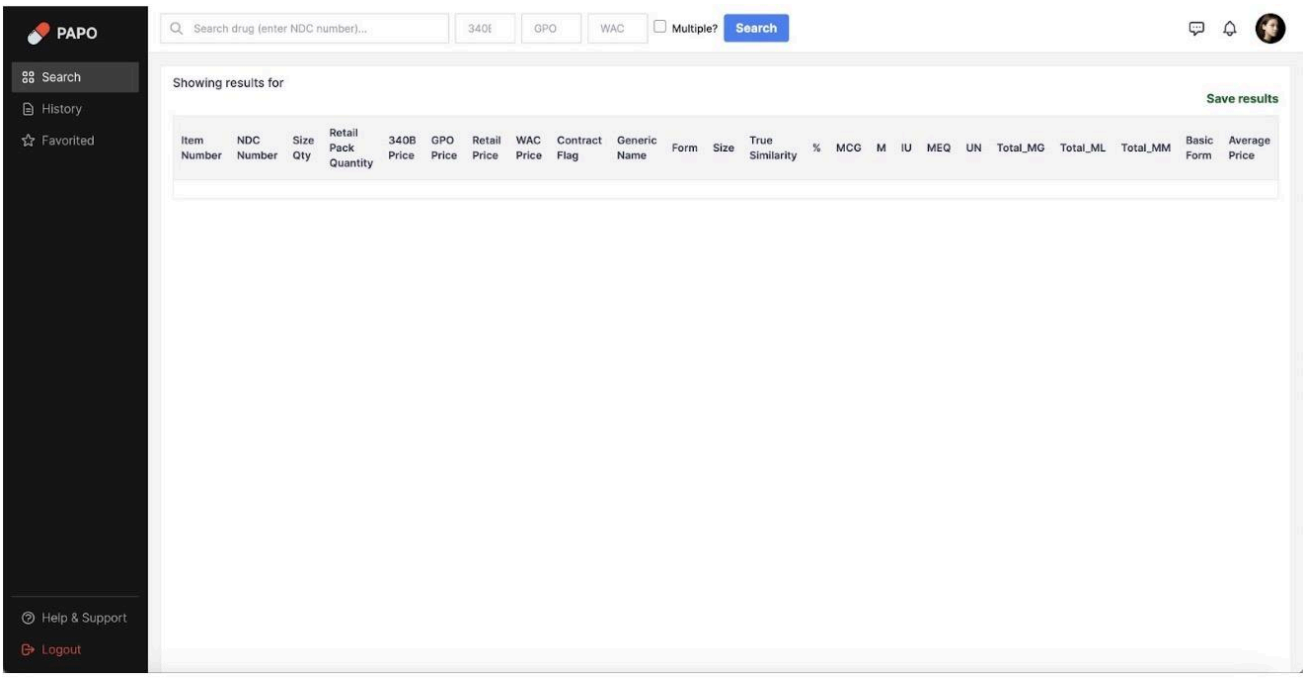

Figure 2.3 The dashboard page of PAPO, where users can search for drugs by entering the NDC number, view detailed results including pricing and package size, and utilize advanced filters such as multiple pill selection for precise and tailored search outcomes.

2.3 Software Description

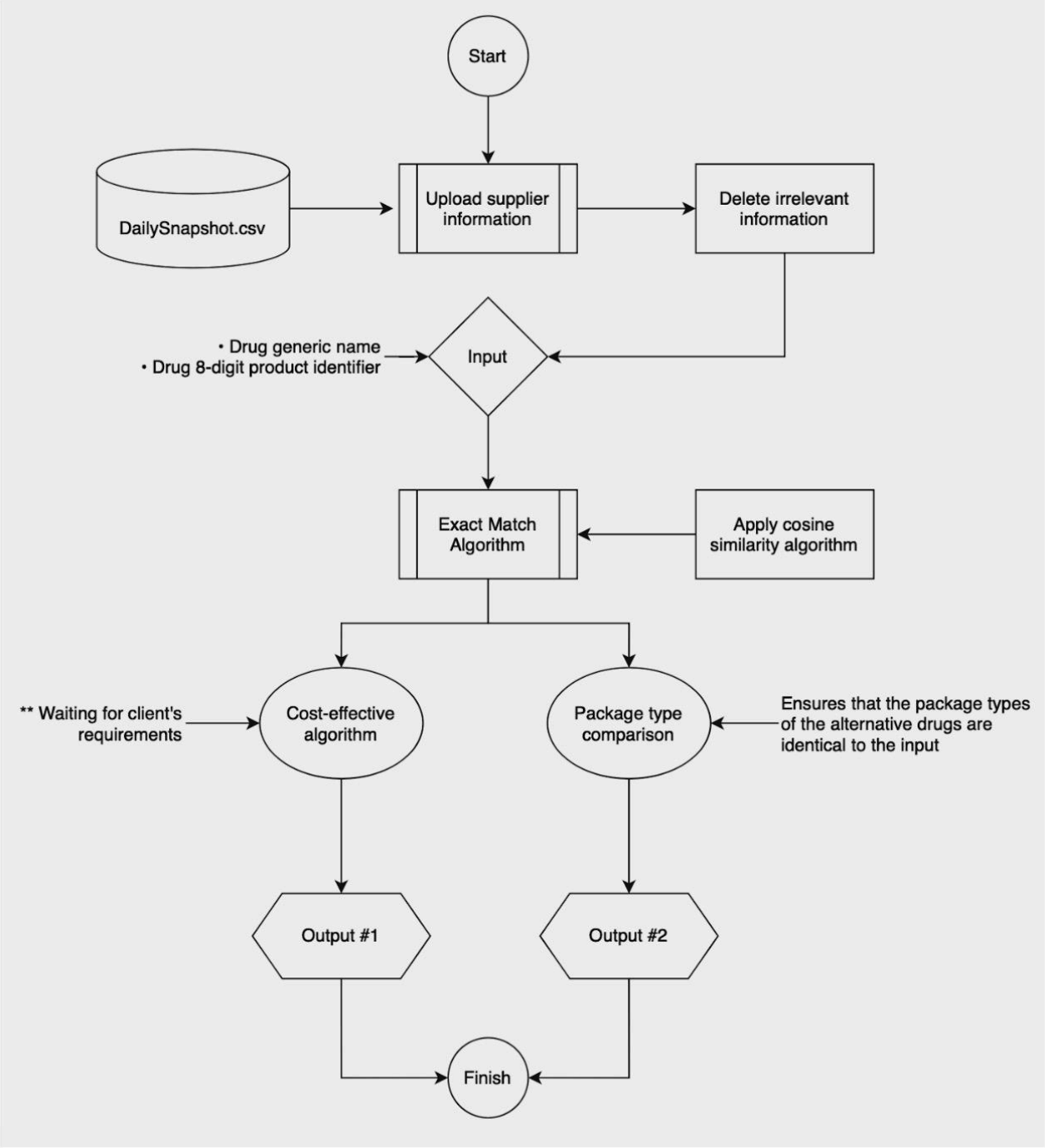

Figure 2.4 The flowchart illustrates the operational workflow, starting with uploading supplier information and cleansing data. Users input drug details, triggering the Exact Match and cosine similarity algorithms. The system then waits for client-specific requirements before applying the cost-effective algorithm and package type comparison to provide precise drug alternatives.

2.4 Installation, Setup, and Support

Setting up the PAPO system is a process that begins with preparing the underlying database infrastructure. This involves an administrator logging into AWS and locating the PostgreSQL database designed to store user data. The admin must verify that the database service is operational and that security settings are appropriately configured to allow connections, ensuring the database is set to 'publicly available' to permit required access.

Following the database setup, attention shifts to configuring the development and testing environments. This is a critical phase where the systems are prepared to mirror the production settings as closely as possible. The configuration process encompasses the installation and updating of all necessary software, databases, and network configurations, ensuring a smooth transition from development to deployment. Every dependency and library involved in the project must be thoroughly checked and updated to the latest versions to avoid any compatibility or security issues.

The next step involves the preparation of data and scripts crucial for testing. The 'Daily Snapshot.csv', a data file provided by the client containing the test dataset, needs to be stored within the API's accessible reach. This file is essential as it contains the data points against which the PAPO system will be tested.

Once the data is in place, a comprehensive environment verification is conducted. It is essential to ensure that the API is not just operational but also responsive. A responsive API is indicative of a healthy back-end system capable of handling requests efficiently and is vital for the PAPO system's performance.

Lastly, a browser setup is necessary for interfacing with the PAPO system. A stable and reliable web browser is launched, and the navigator is directed to the PAPO project's website. The URL, https://papo-pearl.vercel.app/home, serves as the entry point to the system's user interface, where functionality and performance can be assessed.

In summary, the setup process for PAPO is multifaceted, involving database activation, environment configuration, data preparation, system verification, and browser setup. Each of these steps is interconnected, forming a comprehensive preparatory procedure essential for the seamless operation of the Pharma AI Procurement Optimizer.

3 Operation of the Project

3.1 Operating Mode 1: Normal Operation

In the normal operation mode of PAPO, users interact with the system under standard conditions, accessing features designed to manage and optimize pharmaceutical procurement processes.

Access and Authentication

1. The user starts by accessing the PAPO system through the provided URL.
2. Upon loading the website, the user is prompted to enter their credentials on the login page.
3. Once the correct credentials are submitted, the user is granted access to the main dashboard.

Using the Main Dashboard

The main dashboard is the central hub for all procurement activities.

Here, the user can:

1. Search for drugs by entering the National Drug Code (NDC).
2. View and sort the inventory based on various parameters like price, quantity, and supplier.
3. Access detailed drug information and historical pricing data.
4. The system responds by displaying the requested information, allowing for further actions such as order placement or alternative drug suggestions.

Procurement Actions

To perform procurement actions, the user can:

1. Select a drug from the list.
2. Specify quantity and supplier preferences.
3. Confirm the selection to proceed to the ordering process.
4. The system will process the request and confirm successful actions or prompt for additional information if needed.

Abnormal Results and Troubleshooting

Abnormal results in this mode might include:

1. Error messages or failed searches due to incorrect NDC entries.
2. Delays in response or timeout errors, potentially caused by server issues.

If such issues occur, users should first refresh the page or attempt to re-enter information. Persistent issues may require contacting support.

Exiting Normal Operation

To exit the normal operation mode:

1. The user can simply log out by clicking the 'Logout' button available on the dashboard.
2. To stop all operations, close the browser or the browser tab where PAPO is running.

Safety and Security Notices

1. Users should always log out after completing their session to maintain security.
2. Avoid using the PAPO system on public or unsecured networks to prevent unauthorized access.

By adhering to these guidelines, users can ensure a smooth and efficient experience with the PAPO system under normal operation conditions.

3.2 Operating Mode 2: Abnormal Operations

In the abnormal operation mode, users may encounter unexpected system behaviors or outputs. This section of the user manual describes the steps to identify and recover from such states.

Abnormal states can be identified when the system behaves unexpectedly. Examples include:

1. The system returns no results for a known drug NDC.
2. The user interface does not update or respond after user actions.
3. Error messages are displayed indicating a failed process or data retrieval issue.
4. The system performance is significantly slower than usual, causing delays in operations.

To recover from an abnormal state, the user should:

1. Note down any error codes or messages displayed.
2. Attempt to refresh the web page to reset the session.
3. Check network connectivity to ensure the system can access external data sources and services.
4. Clear the browser cache to resolve any potential conflicts with outdated information.

When input data is out of expected range or format, the system will typically provide a validation error message. The user should:

1. Review the input data for accuracy.
2. Re-enter the information, ensuring that it conforms to the required format and data type.
3. If the problem persists, report the issue to the support team with detailed input data and error messages received.

To exit abnormal operation mode, the user may need to:

1. Log out of the system entirely.
2. Restart their browser or computer to clear any systemic issues.
3. If the system is unresponsive, use task manager or equivalent to force close the browser.

3.3 Safety Issues

Safety and security are paramount for PAPO, especially considering the sensitive nature of the pharmaceutical data it handles. During normal operations, the primary safety concern revolves around ensuring the confidentiality, integrity, and availability of data. Unauthorized access to user accounts can lead to privacy breaches or incorrect procurement decisions. To mitigate this, the system employs robust authentication protocols, and users are encouraged to follow best practices for password creation and account management. Additionally, the system has been designed to prevent accidental data exposure through features such as automatic timeouts and secure logout processes.

In abnormal operations, such as system errors or out-of-range data inputs, the risk of data corruption or unintended data leaks increases. The system is designed to enter a safe mode or log out the user if it detects a critical error to prevent further issues. Regular backups and redundant systems are in place to preserve data integrity and ensure recovery in the event of a failure.

Regarding the broader system, PAPO is part of a larger healthcare management ecosystem. A failure within PAPO could potentially disrupt the supply chain, leading to delays in drug procurement, which could affect patient care. This potential impact on the larger healthcare system underscores the need for continuous monitoring, rapid response protocols, and a well-defined disaster recovery plan.

It's important to note that while the system does not directly interact with physical safety elements like fire or hazardous materials, an indirect risk could arise if the system's outputs were to be used incorrectly, leading to procurement of the wrong medications. Therefore, the system includes checks to minimize the risk of incorrect data processing.

4 Technical Background

The technical background of PAPO revolves around a sophisticated suite of software tools and principles designed to streamline the process of pharmaceutical procurement. The technical approach combines the power of modern web technologies, data analytics, and cloud computing to create an efficient and user-friendly experience for users who may not be technically inclined. The system is composed of several detailed components, each playing a critical role in the functionality of the system.

At its core, PAPO employs a web-based application framework powered by React, a JavaScript library for building user interfaces with a high degree of interactivity and state management. React’s component-based architecture enables the modular development of the PAPO dashboard, which reacts in real time to user inputs without the need for page reloads, improving the overall efficiency and user experience. Utilized for its efficient update and rendering capabilities, React allows for a dynamic user experience with an interface that updates in real-time without the need for full page refreshes. It's designed to handle the complex state management required for the various user interactions within PAPO. React is used to craft the elements and operations within the user interface, which is designed to provide a seamless and intuitive user experience while maintaining a technical backbone that ensures performance and security. Beginning with the login page, the input fields for username and password are managed by state variables within the React framework, facilitating real-time state management and ensuring a reactive user input experience. Client-side validation is integrated to provide immediate feedback on input errors, enhancing user interaction and preventing unnecessary server requests. When form data is submitted, secure HTTPS POST requests convey credentials to the backend for verification. A successful login results in a secure token—typically a JWT—which is stored to manage the user session, ensuring a seamless and secure user experience. Error handling on the login page is discreet and informative, designed to maintain security while guiding users through login issues. Transitioning to the dashboard, the user is greeted with a navigation panel that simplifies access to the application's diverse features, from search functionalities to historical data review. The dashboard’s central component, a robust search bar, interacts with the backend API to fetch and display drug information based on user queries. Results are rendered in a tabular format with options to sort and filter, ensuring a data-rich yet comprehensible interface. Interactivity is paramount, with the dashboard responding instantaneously to user actions thanks to React's virtual DOM, which optimizes the update and render cycle for on-screen elements. Action buttons are wired with event handlers, translating user clicks into meaningful operations like order placements or data exports. Advanced state management techniques, possibly leveraging Redux or the Context API, maintain the synchronicity between the application’s state and the user interface.

On the server side, PAPO leverages FastAPI, a modern, fast web framework for building APIs with Python that provides backend functionality. FastAPI is known for its performance benefits and ease of use, offering asynchronous request handling and automatic data validation which increases the reliability and scalability of the application. It provides a robust set of endpoints for Create, Read, Update, and Delete (CRUD) operations essential for data manipulation and retrieval. New user accounts can be created within the system, along with the addition of new pharmaceutical data as required. The system can retrieve and display information such as drug details, pricing, availability, and user account information. User profiles and drug information can be modified as needed, allowing for real-time updates to the data as market conditions change. Users with the appropriate permissions can remove entries from the database, such as outdated user accounts or obsolete drug information.

Data storage and management rely on PostgreSQL, a robust and secure object-relational database system, which is hosted on Amazon RDS (Relational Database Service). This setup allows for reliable and scalable storage solutions, essential for handling the vast amounts of data involved in pharmaceutical procurement. The user information stored within the PAPO database is comprehensive, facilitating not only user authentication but also providing a means to manage access and maintain security. The database stores username and password. Credentials are securely stored and are essential for user authentication. Passwords are hashed for security. User ID and center ID are also stored. These identifiers uniquely distinguish each user and their associated work center within the system, ensuring that data and actions are correctly attributed. Another important stored data is IsLocked. This is a boolean flag indicating whether a user's account is locked, a critical feature for security protocols that restrict access after multiple failed login attempts.

The system is deployed on an AWS EC2 instance, ensuring that the necessary computing resources are available on-demand and can be scaled according to the user traffic and data processing needs. This cloud-based deployment also adds a layer of resilience and geographic redundancy to the service.

PAPO places heavy emphasis on operation logs. It maintains a detailed log of activities and errors in a file named backend.log within the API. This log captures a wealth of information including API call outcomes, authentication attempts, and system errors, which is crucial for both security auditing and troubleshooting.

Underpinning the PAPO system's functionality are algorithms specifically tailored to the pharmaceutical industry's needs. These include:

Exact Match Algorithm: Ensures that the drugs searched for by the user match exactly with the database records, reducing the likelihood of errors in drug identification.

Cosine Similarity Algorithm: Applied to textual data to find drugs that are most similar to the user's input, based on their active ingredients or other defining characteristics. This is a principle derived from vector space models in computational linguistics and information retrieval.

Cost-Effective Algorithm: This component takes into account various pricing parameters to suggest the most cost-effective drug options. It uses a weighted average approach, which is rooted in statistical analysis to determine the significance of each price type in the final calculation.

In the realm of data security, PAPO incorporates industry-standard encryption and security protocols to ensure that all user data remains confidential and secure. By following principles like least privilege access and secure coding practices, the system minimizes vulnerabilities and protects against unauthorized access.

From a technical perspective, PAPO is grounded in established principles of computer science and software engineering, tailored to meet the specific demands of pharmaceutical procurement. It seamlessly combines these elements to provide a reliable and secure platform for its users, with an emphasis on the practical application of these technical principles to improve procurement outcomes.

5 Relevant Engineering Standards

PAPO adheres to a variety of engineering standards that ensure the system is robust, secure, and efficient, aligning with best practices in software development and operational protocols.

PAPO is built on modern software architecture principles, using high-level frameworks like React for the frontend and FastAPI for the backend. The development team follows established coding standards and practices, such as the PEP 8 style guide for Python and the AirBnB style guide for JavaScript. These standards ensure that the code is not only efficient and secure but also maintainable and scalable. Version control is managed through Git, with strict branching and merging policies to maintain the integrity of the codebase.

In terms of communication, PAPO ensures data integrity and security by adhering to HTTP/HTTPS protocols, particularly SSL/TLS for data encryption during transmission. This is crucial for protecting sensitive user data and authentication tokens that are transmitted between the client and server. The system's API interactions are structured around the REST architectural style, which simplifies communications and enhances the scalability of server-client interactions.

PAPO is deployed on cloud services provided by Amazon Web Services (AWS), utilizing EC2 for compute capacity, RDS for database services, and S3 for data storage. These services comply with high operational standards, including ISO/IEC 27001 for security management. They provide the scalability and reliability necessary to support the application's demands. The operational environment also follows the best practices for disaster recovery and data backup to ensure high availability and data persistence.

Given that PAPO handles sensitive pharmaceutical data, it complies with relevant governmental and industry regulations, such as the Health Insurance Portability and Accountability Act (HIPAA) in the U.S., which sets the standard for protecting sensitive patient data. The application also aligns with the General Data Protection Regulation (GDPR) for users in the European Union, ensuring that data handling, storage, and processing meet these stringent regulations.

The engineering team also keeps abreast of updates in web development and security standards published by organizations such as the World Wide Web Consortium (W3C) and the Open Web Application Security Project (OWASP). Adhering to these standards helps the system mitigate common security vulnerabilities and stay updated with the latest in web technologies and practices.

6 Cost Breakdown

Project Costs for Production of Beta Version				
	Quantity	Description	Unit Cost	Extended Cost
1	12	Amazon Virtual Private Cloud	$4 - $10	$48 - $120
2	1	Domain Name Service	$14	$14
			Beta Version-Total Cost	$62 - $134

This project has two expenses in total. Although the team has adopted the use of the free tier of AWS, Amazon still charges the team for its cloud service use, which allows us to host the database and store necessary data. Currently, we are being charged $4 for our storage use rate. However, this number is likely to go up as we expand the software to other facilities and centers. More frequent use of the software, coupled with more registered users, will result in more storage use. We are estimating the monthly rate for more storage use to be in the $4 - $10 range.

Additionally, we are paying $14 annually to own the papoai.net domain name. We believe this catchy URL will draw more users to the platform and encourage them to actively use it. Therefore, it is worth paying this amount. Thus, for the next year, the total cost of the beta version of our product is estimated to be in the range of $62 - $134.

7 Appendices

[Appendices include supplemental information for the User that would distract if included in the regular sections.

7.1 Appendix A - Specifications

Requirement	Value, range, tolerance, units
Accuracy of Cost Analysis Algorithm	≥ 98% accuracy
Response Time of Cost Analysis Calculations	≤ 2 seconds
Login Page Authentication	100% accuracy
Login Page Response Time	≤ 2 seconds
Main Page Functionality	100% compliance
Audit Log Storage	100% accuracy

7.2 Appendix B – Team Information

Bora Bulut
Worked primarily on database development, QA testing, and technical documentation. Will be pursuing MEng in ECE at Cornell Tech following graduation.

Manuel Segimon
Operated as Head of Continuous Development, working on backend and algorithm development, including exact match, cost-effective, and package analysis algorithms.

Joel Akerman
Worked as Product Manager, overseeing the whole product development lifecycle and presenting the prototypes.

Taha Ababou
Worked on front-end development, UI design, and API formation. Will be pursuing MS in Statistical Practice at Boston University following graduation.

Zaiyan Muhammad

Operated as Technical Project Manager in charge of logistics and created roadmaps, Gantt Charts to ensure the team hands in the deliverables by the deadline.