InfoPulse Software System (IPSS)

v1.0

Software Requirements Specification

Author: Joshua Bonner

Contact: jbb5882@psu.edu

Master of Software Engineering

SWENG 894 (SUM 2024) – Capstone Experience

1. Introduction

1.1. Purpose

This document provides a detailed description of the requirements for the IPSS (InfoPulse Software System) web application. The purpose of this application is to offer users an advertisement-free platform to view the latest news based on searches using NewsAPI sources. This document is intended for developers, project managers, and stakeholders.

1.2. Mission Statement

IPSS will provide users with a curated feed of the latest news on subjects they are interested in in a web application free from advertisements and other clutter that detract from the experience.

1.3. Scope

IPSS will feature user authentication, personalized news feeds, search functionalities, and secure data handling. The GUI will be built using the Vue.js framework leveraging its Composition API, while the backend implementation will leverage Python's FastAPI framework with Pydantic models. User data and search preferences will be stored in a PostgreSQL Database and news article metadata content will be stored in an Elastic Search index for quick retrieval, persisted through user sessions.

1.4. Definitions, Acronyms, and Abbreviations

- **1.4.1. IPSS:** InfoPulse Software System
- **1.4.2. API:** Application Programming Interface
- **1.4.3. GUI:** Graphical User Interface
- **1.4.4. UI/UX:** User Interface/User Experience
- **1.4.5. JSON:** JavaScript Object Notation
- **1.4.6. TLS:** Transport Layer Security

1.5. References

- **1.5.1. Vue.js –** https://vuejs.org/guide/introduction
- **1.5.2. Fast API –** https://fastapi.tiangolo.com/
- **1.5.3. Pydantic –** https://docs.pydantic.dev/latest/
- **1.5.4. PostgreSQL –** https://www.postgresql.org/docs/current/intro-whatis.html
- **1.5.5. Elastic Search –** https://www.elastic.co/about/
- **1.5.6. NewsAPI –** https://newsapi.org/

2. System Overview

2.1. Description

IPSS is a web application leveraging the latest in web technologies to provide the user with quick access to the most current news pulled from NewsAPI, one of the most popular sources available to developers with API endpoints to cover any need. Users can utilize this web application to keep themselves up to date on the latest information on a particular topic without the normal distractions, like popups or advertisements, at a quick glance and promptly.

2.2. Rationale

Today, web applications that provide the news in some form or another are generally riddled with advertisements and many other useless clutter that detract from the original purpose of the site. With IPSS, the idea is to create a simple and intuitive UI that provides the user with a feed of curated information based on their searches without these detractions as quickly as possible so they can move on with their day well-informed.

2.3. Target Audience

The intended audience for IPSS is busy individuals who want to gather information relatively quickly and efficiently without advertisements and clutter that prevents them from accessing the information as quickly as possible. The idea is for a user to access the web application by logging in, search for a particular topic or by keyword, and be presented with the latest news about the subject. No popups, advertisements, and other distractions period.

2.4. Key Features

2.4.1. User Authentication and Account Management
2.4.2. Personalized news feed based on user searches and preferences
2.4.3. Topic and Keyword search for news articles
2.4.4. Secure data storage and handling
2.4.5. Real-time notifications for updated news feeds

2.5. Constraints

2.5.1. Limited to NewsAPI endpoints and sources
2.5.2. General browser limitations given typical network traffic
2.5.3. Users must generally understand how to use web applications

2.6. Assumptions and Dependencies

2.6.1. The application assumes reliable internet connectivity for fetching news articles

2.6.2. Dependencies include Vue, FastAPI, PostgreSQL, ElasticSearch, and NewsAPI

3. Software Requirements

3.1. Functional Requirements

3.1.1. User Authentication and Account Management

3.1.1.1. IPSS shall authenticate each user by providing authentication mechanisms to access the web application.

3.1.1.2. IPSS shall reject unauthenticated login attempts outright.

3.1.1.3. IPSS shall provide a method to create a new user account.

3.1.1.4. IPSS shall provide a mechanism to recover or reset the user's authentication credentials.

3.1.1.5. IPSS shall store authentication credentials securely by encoding the user's sensitive information.

3.1.1.6. IPSS shall allow the user to log out securely.

3.1.2. News Content and Search Functionality

3.1.2.1. IPSS shall provide the most recent articles by searched topic or keyword sorted by popularity.

3.1.2.2. IPSS shall save user preferences to enable default search customization.

3.1.2.3. IPSS shall allow the user to save search criteria for later use.

3.1.2.4. IPSS shall organize searched articles based on user preferences.

3.1.2.5. IPSS shall reach out to NewsAPI data sources for articles relating to the search criteria provided by the user.

3.1.2.6. IPSS shall allow the user to view an article's content.

3.1.2.7. IPSS shall provide alerts for when their news feed is updated.

3.1.2.8. IPSS shall provide an application dashboard for the user.

3.1.2.9. IPSS shall provide a means to search for articles from the dashboard.

3.1.2.10. IPSS shall provide search tabs to organize all the articles returned in a search tab.

3.1.2.11. IPSS shall provide cards within the search tabs that house the content of an article.

3.1.2.12. IPSS shall provide a page to view the content of an article.

3.1.3. Compatibility

3.1.3.1. IPSS shall be compatible with major web browsers (e.g., Chrome, Firefox, Edge)

Functional Requirements / User Stories Mapping:

REQ #	User Story
3.1.1.1	As a user, I would like to log in to the IPSS by providing account my credentials which allow me to view my personal dashboard
3.1.1.2	As a user, I would like my account to be secured from unauthenticated login attempts
3.1.1.3	As a user, I would like to be able to create an account so that I may utilize the application
3.1.1.4	As a user, I would like to be able to retrieve my login credentials in case I forget them
3.1.1.5	As a user, I would like my login credentials to be safely stored to prevent unauthenticated login attacks
3.1.1.6	As a user, I would like to be able to log out of the application
3.1.2.1	As a user, I would like to be able to view a list of relevant articles based on the search criteria I provide
3.1.2.2	As a user, I would like to be able to set preferences for default search behavior and have it persist between sessions
3.1.2.3	As a user, I would like to be able to customize my search with additional parameters that will curate the articles that are provided by the application
3.1.2.4	As a user, I would like to be able to view a list of relevant articles based on the search criteria I provide
3.1.2.5	As a user, I would like articles to be provided by NewsAPI
3.1.2.6	As a user, I would like to view the article content I selected from the dashboard article card
3.1.2.7	As a user, I would like to be informed when my dashboard feed has been updated
3.1.2.8	As a user, I need to be presented with a dashboard element that will allow me to interact with the web application including initiating an article search, saving search criteria, and logging out
3.1.2.9	As a user, I would like to be able to provide a keyword or topic which can be searched upon
3.1.2.10	As a user, I would like to be able to organize my searches by tabs
3.1.2.11	As a user, I would like to access articles the IPSS provides in an organized fashion within the search tabs as cards
3.1.2.12	As a user, I would like to view the article's content
3.1.3.1	As a user, I would like to utilize a web browser of my choice to access the IPSS application

3.2. Nonfunctional Requirements

The following non-functional requirements will facilitate an intuitive and functional application in accordance with stakeholder expectations. Typical modern web applications adhere to specific performance metrics and capabilities that will be included in the IPSS.

3.2.1. User Interface

3.2.1.1. IPSS shall provide a web application GUI that is intuitive and free of advertisements, following standard UI/UX design methodologies.

3.2.2. Performance

3.2.2.1. IPSS shall load search content within 2 seconds.

3.2.2.2. IPSS shall maintain latency no greater than 100 milliseconds.

3.2.3. Security

3.2.3.1. IPSS shall encrypt all sensitive user data in transit and at rest.

3.2.3.2. IPSS shall provide failover mechanisms with verbose error statements.

3.2.4. Documentation

3.2.4.1. IPSS shall provide documentation for both end users and developers.

4. Significant Algorithmic Component

The core theme of the IPSS is to facilitate a simple and burdenless search engine for news articles based on search criteria provided by the user in a simple and intuitive user interface. Implementing this search functionality is crucial to the application's success. Typically, in other applications, the user is bombarded with advertisements, clutter, or locked behind paywalls that prevent them from digesting the content of the articles they search for using other popular search engines. For IPSS, the mission statement is clear,

"*IPSS will provide users with a curated feed of the latest news on subjects they are interested in in a web application free from advertisements and other clutter that detract from the experience*."

With this in mind, a clear picture of why the application is necessary and to whom it would serve is presented. Not everyone has time to deal with all the fuss associated with various news outlets, i.e. setting up an account, paying for each article they digest, or dealing with popup advertisements. Therefore, our solution is the answer to that problem.

Making use of open-source software is the forte of a good application and its development team. I have decided that leveraging a popular open-source solution by the name of NewsAPI would be a great start to satisfying the major requirements for this project. Below is a sequence diagram that will provide some detail as to how the interactions will look between the various clients and servers.

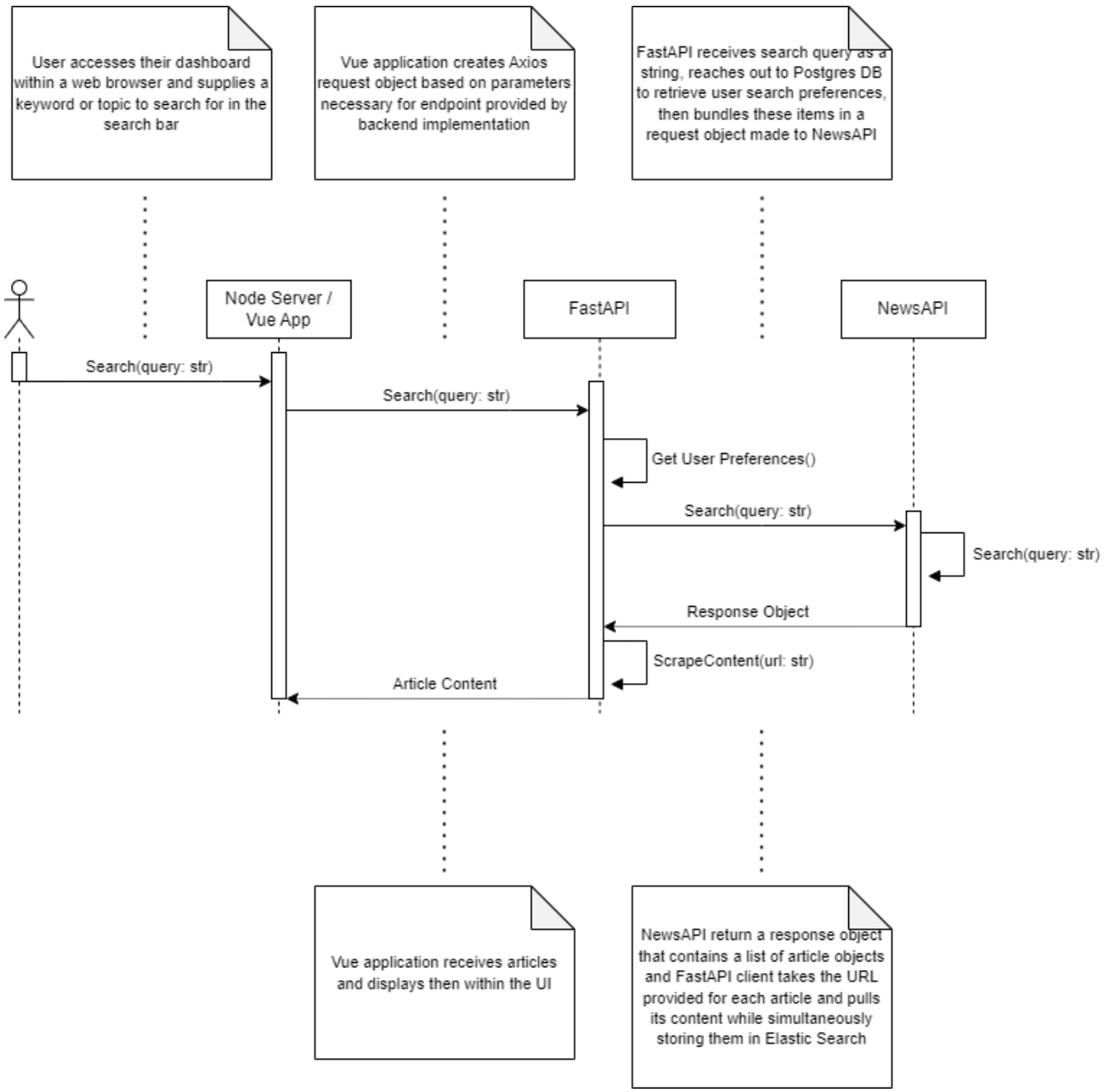

Figure 1: Sequence Diagram for major search functionality between various IPSS clients

In this diagram, a few major interactions are taking place between various clients and servers within IPSS, I will briefly describe them. The first step in the process is for the user to have accessed the application and provided a topic or keyword in the user dashboard's search bar within the Vue application. From there, a basic Axios request object is created which includes the query string provided and passes this object to the backend FastAPI server. Assuming the user has set search preferences, those preferences are utilized to create a subsequent request object that is then sent to NewsAPI. NewsAPI will then return a list of articles based on the search criteria provided. The URL provided for each article is then utilized to scrape all of the <p> tags for its content. At this time they are stored in ElasticSearch for persistence and then served up to the Vue application UI for the user to digest. The user is then able to choose an article card to view the contents of an article free from advertisements and other distractions.

This solution as described will satisfy all foundational requirements posed by the stakeholders and will provide the user with a quick and efficient way of browsing the latest news on topics they are interested in with little to no fuss.