Software Design Document (SDD) for PGWatch2 Enhancements

ANAND GOPAN G

ARJUN P

K PAVITRA

1. Introduction

This document outlines the design and implementation details for the enhancements made to PGWatch2 and PGWatch v3. The primary goal of these updates is to improve the monitoring experience, enhance usability, and introduce new features that offer better data accessibility and visualization.

2. Scope

The enhancements focus on:

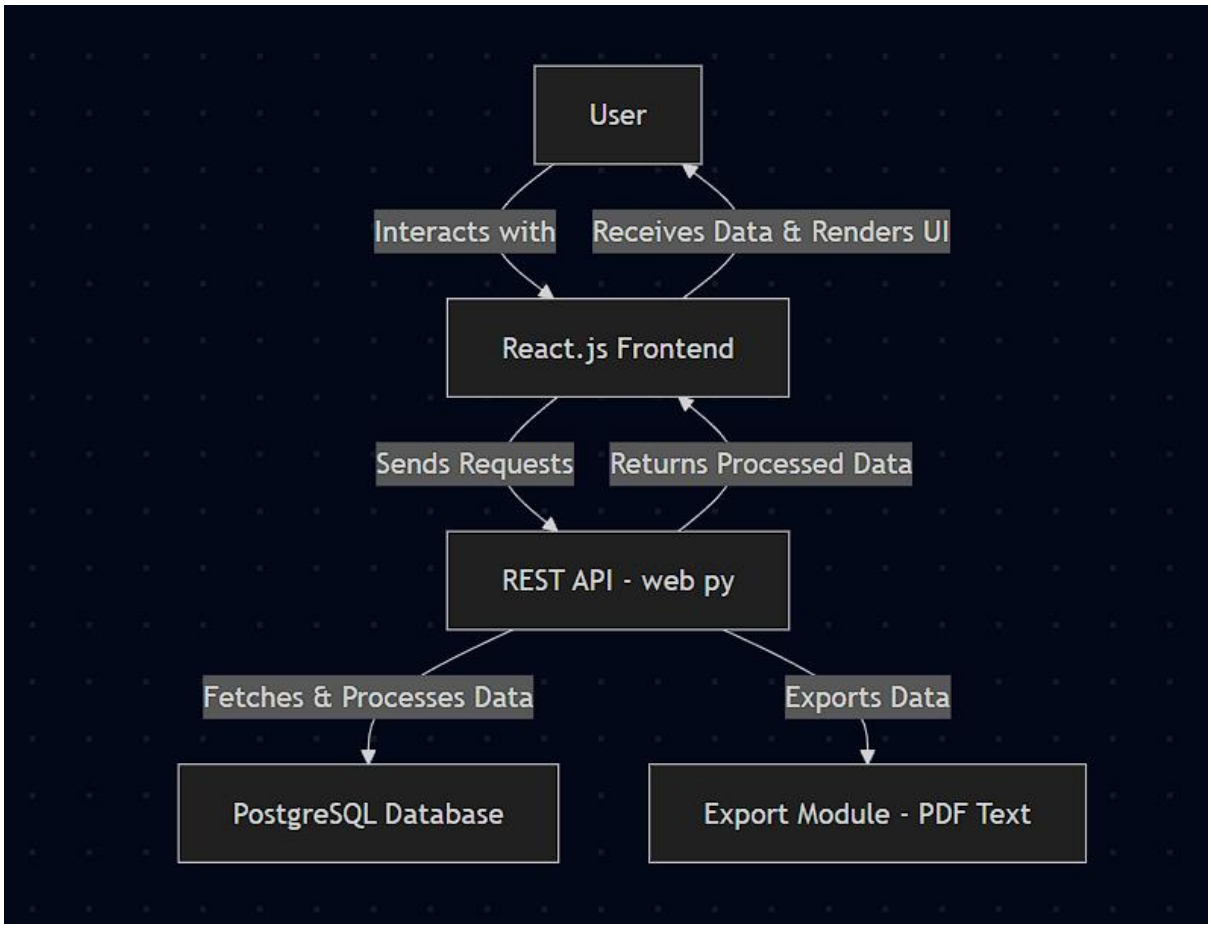
- Real-time Metrics Fetching & PDF Export (pgwatch v3)
- UI Enhancements & Login Feedback (pgwatch v3 & pgwatch2)
- Metrics Export as Text File (pgwatch2)
- Light/Dark Mode (pgwatch2)

3. System Architecture

3.1 Architectural Overview

The PGWatch2 enhancements follow a **client-server architecture**:

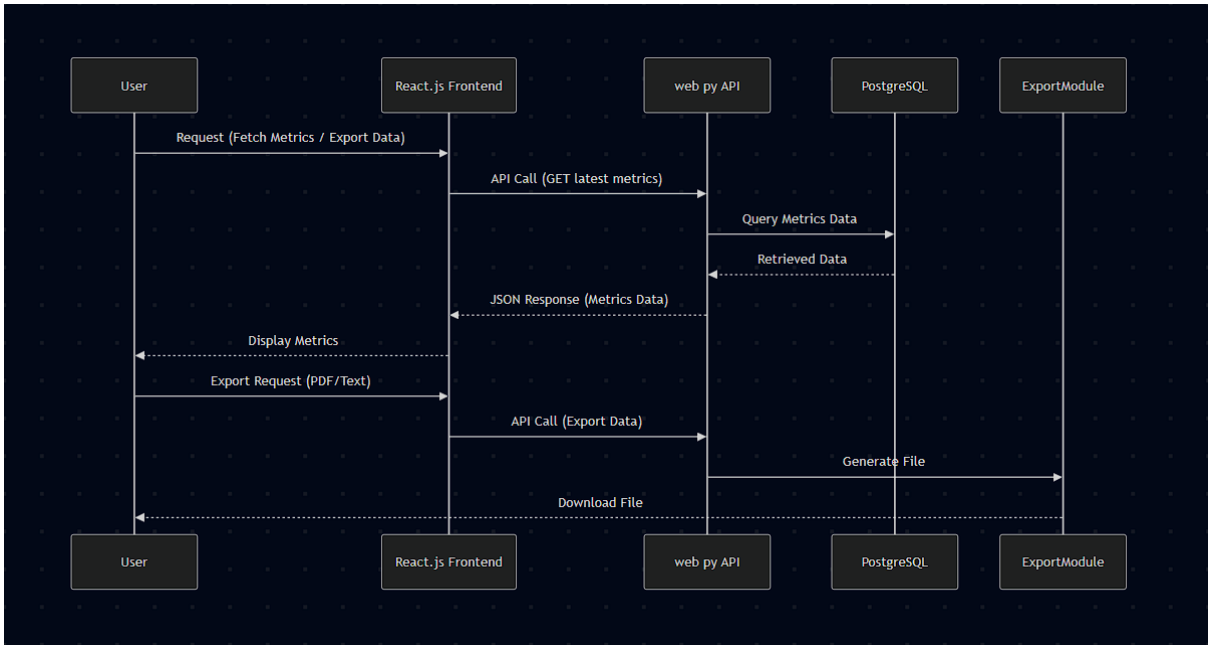
- **Frontend:** Developed using **React.js**, responsible for UI enhancements and user interactions.
- **Backend:** Implemented in **Python (web.py)** with **PostgreSQL** for data storage and processing.
- **APIs:** RESTful endpoints facilitate communication between the frontend and backend for fetching metrics, exporting data, and handling user authentication.


User
Interacts with
Receives Data & Renders UI
React.js Frontend
Sends Requests
Returns Processed Data
REST API - web py
Fetches & Processes Data
Exports Data
PostgreSQL Database
Export Module - PDF Text

3.2 Data Flow Diagram

The following describes the high-level data flow within the system:

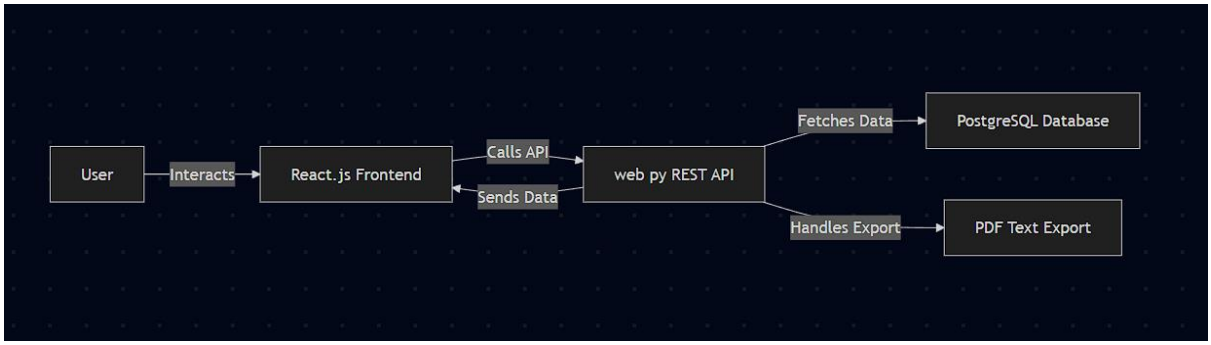
1. **User Action:** A user initiates a request (e.g., fetching real-time metrics, exporting data, toggling dark mode).
2. **Frontend Processing:** The request is sent via API to the backend.
3. **Backend Processing:** The backend retrieves relevant data from PostgreSQL, processes it, and returns a response.
4. **Frontend Rendering:** The UI updates accordingly (e.g., displaying metrics, saving files, changing themes).



3.3 Component Diagram

The system is divided into the following components:

- **User Interface (React.js):** Handles user interactions and displays data.
- **REST API Layer (web.py):** Manages HTTP requests and responses.
- **Database (PostgreSQL):** Stores and retrieves monitored metrics data.
- **Export Module:** Supports exporting metrics in PDF or text format.



4. Real-time Metrics Fetching & PDF Export (pgwatch v3)

Overview

The **Latest Metrics Button** allows users to fetch the most recent database metrics, including Transactions Per Second (TPS), Queries Per Second (QPS), buffer hit ratio, and more. Users can also export these metrics as a **PDF** for easy sharing and documentation.

Implementation Details

Frontend Changes

- **New React Component:** LatestMetricsButton.tsx
- **UI Enhancements:** Integrated into SourcesGridActions.tsx
- **PDF Export:** Utilized jspdf for formatting and download.

Backend Changes

- **New API Endpoint:** GET /latest-metrics to retrieve the latest database metrics.
- **Optimized SQL Queries:** Efficient retrieval of recent monitoring data.
- **Data Processing & Formatting:** Ensured structured JSON responses.

Technical Impact

- **Security & Authentication:** Token-based authentication.
- **Performance Considerations:** Optimized query execution.

Future Enhancements

- Filtering options for metrics.
- Graphical visualization in exported PDFs.

5. UI Enhancements & Login Feedback (pgwatch v3 & pgwatch2)

Overview

This update modernizes the authentication interface, improves user experience, and enhances security.

Implementation Details

Login Form Enhancements

- **Before:** Basic static layout, minimal styling, no interactive feedback.
- **After:** Modernized UI using **Material-UI**, improved validation via **Yup**, animated transitions, and **snackbar notifications** for login feedback.

Code Improvements

- **State Management:** useState for UI state handling.
- **Modularization:** Separated components for better maintainability.
- **Improved Styles:** TSS-React used for styling.
- **Accessibility Enhancements:** Better focus states and spacing.

Testing Checklist

✅ Hover effects and transitions ✅ Responsive layout ✅ Snackbar notifications

Future Improvements

- Theming support.
- Two-factor authentication.

6. Metrics Export as Text File (pgwatch2)

Overview

Users can now download a summary of monitored database metrics in a text file format for offline review.

Implementation Details

UI Changes (webpy/templates/stats-summary.html)

- **New "Download as Text" button** added next to "Show".
- Styled using Bootstrap's btn btn-secondary class.

Backend Modifications (webpy/web.py)

- **Logic for Text File Export:** Implemented a download parameter.
- **Generated .txt file** containing structured database statistics.

Sample Code Snippet

```
@logged_in
@cherrypy.expose
def stats_summary(self, **params):
    download = params.get('download')
    if dbname and download:
        text_content = f"Database Overview for {dbname}\n"
        text_content += "=" * 50 + "\n\n"
        for key, values in data:
            text_content += f"{key}\t{values[0]}\t{values[1]}\t{values[2]}\n"


        cherrypy.response.headers['Content-Type'] = 'text/plain'
        cherrypy.response.headers['Content-Disposition'] = f'attachment;
filename="{dbname}_overview.txt"'
        return text_content
```

Benefits

- Quick export of metrics for offline review.
- Lightweight and dependency-free data retrieval

.

7. Light/Dark Mode (pgwatch2)

Overview

This enhancement introduces a **Dark/Light Mode Toggle**, allowing users to switch themes for improved UI experience.

Implementation Details

CSS Variables (static/custom.css)

```
:root {
    --bg-primary: #ffffff;
    --bg-secondary: #f8f9fa;
    --text-primary: #212529;
}
[data-theme="dark"] {
    --bg-primary: #212529;
    --bg-secondary: #343a40;
    --text-primary: #f8f9fa;
}
```

JavaScript for Theme Toggle

```
document.addEventListener('DOMContentLoaded', function() {
    const savedTheme = localStorage.getItem('theme') || 'light';
    document.documentElement.setAttribute('data-theme', savedTheme);
});

function toggleTheme() {
    const currentTheme = document.documentElement.getAttribute('data-theme');
    const newTheme = currentTheme === 'light' ? 'dark' : 'light';
    document.documentElement.setAttribute('data-theme', newTheme);
    localStorage.setItem('theme', newTheme);
}
```

HTML Changes

```
<li class="nav-item">
    <a class="nav-link" href="#" onclick="toggleTheme(); return false;">
        <i id="theme-icon" class="fas fa-moon"></i>
    </a>
</li>
```

Benefits

✅ Improved visual experience with theme switching. ✅ Persistent theme preferences. ✅ Intuitive UI toggle button.

Testing Checklist

✅ Theme persists after reload
✅ Toggle button updates instantly
✅ UI elements adapt properly in both themes

8. Conclusion

The **PGWatch2 Enhancements** outlined in this document introduce significant usability improvements, modern UI design, and additional data retrieval capabilities. These updates enhance monitoring efficiency, improve user interaction, and provide a seamless experience for database administrators.

Future iterations may include further UI refinements, expanded export options, and additional security enhancements.