# **GSoC’25 Keploy App Dashboard**

# **Prototype Software Requirements Specification (SRS)**

## **1. Introduction**

### **1.1 Purpose**

The purpose of this document is to define the functional and non-functional requirements for the **Keploy Dashboard with Metrics and Charts**. This project aims to develop a **real-time, interactive dashboard** for monitoring GitHub activity, providing analytics, data visualizations, and insights for developers.

### **1.2 Document Conventions**

* **Shall** indicates a mandatory requirement.
* **Should** indicates a recommended requirement.
* **May** or **Optional** indicates a flexible requirement.

### **1.3 Intended Audience and Reading Suggestions**

This document is intended for:

* GSoC mentors and developers at **Keploy**.
* Project contributors and future maintainers.
* UI/UX designers and testers.

### **1.4 Project Scope**

This project will provide:

* Real-time GitHub activity tracking (commits, PRs, merges, etc.).
* Customizable dashboard with interactive charts and graphs.
* Secure authentication using GitHub OAuth.
* High-performance architecture using Golang, Redis, and MongoDB.
* AI-driven insights for trend analysis.
* Real-time updates via **Socket.IO**.

## **2. Overall Description**

### **2.1 Product Perspective**

The dashboard will act as an **extension of GitHub's analytics**, offering better visualization and data-driven insights for developers. It will integrate **GitHub API** and **Keploy's testing framework** for enhanced observability.

### **2.2 Product Functions**

The application shall provide the following functionalities:

* **User Authentication:** Secure login with GitHub OAuth.
* **Activity Tracking:** Fetch GitHub user activity in real time.
* **Custom Dashboards:** Users can filter, sort, and analyze data.
* **Data Visualization:** Interactive charts and reports.
* **Performance Optimization:** Uses Redis caching for efficiency.
* **AI Insights:** Uses machine learning models for predictive analysis.

### **2.3 User Characteristics**

The target users include:

* Open-source developers and contributors.
* Software engineers tracking GitHub metrics.
* Teams managing software development workflows.

### **2.4 Constraints**

* Limited **GitHub API rate limits** (will optimize using caching and batch requests).
* **OAuth authentication restrictions** (must comply with GitHub policies).
* **Real-time data handling** must be optimized for performance.

## **3. Specific Requirements**

### **3.1 Functional Requirements**

| **ID** | **Requirement Description** |
| --- | --- |
| FR-01 | The system shall allow users to authenticate via GitHub OAuth. |
| FR-02 | The system shall fetch GitHub activity logs (PRs, commits, merges). |
| FR-03 | Users shall be able to filter and sort activity logs by repo, time, and type. |
| FR-04 | The system shall generate interactive graphs using **Chart.js/Recharts**. |
| FR-05 | The dashboard shall support real-time updates using **Socket.IO**. |
| FR-06 | The system shall cache API responses using **Redis** to improve performance. |
| FR-07 | AI-driven insights shall be provided for user activity trends. |
| FR-08 | Users shall be able to customize dashboard layouts and views. |

### **3.2 Non-Functional Requirements**

| **ID** | **Requirement Description** |
| --- | --- |
| NFR-01 | The system shall be built using **Next.js** for optimized performance. |
| NFR-02 | The backend shall be implemented in **Golang (Go Fiber framework)**. |
| NFR-03 | Response time for API calls shall not exceed **1 second** for cached data. |
| NFR-04 | The system shall support up to **10,000 concurrent users**. |
| NFR-05 | All API requests shall be encrypted using **HTTPS and OAuth tokens**. |
| NFR-06 | The system shall be deployable using **Docker and Kubernetes**. |

### **3.3 External Interface Requirements**

#### **3.3.1 User Interfaces**

* **Login Page:** GitHub OAuth authentication.
* **Dashboard:** Displays graphs, analytics, and filters.
* **Settings Page:** Allows customization and user preferences.

#### **3.3.2 Hardware Interfaces**

* Cloud-based deployment with **minimum 2 vCPUs and 8GB RAM**.
* Supports both **desktop and mobile web browsers**.

#### **3.3.3 Software Interfaces**

* **GitHub API:** Fetches repository and user activity data.
* **Redis:** Caching mechanism for faster API response.
* **MongoDB:** Stores user data and custom settings.
* **Auth.js:** Manages authentication securely.

### **3.4 Performance Requirements**

* API response time should be **≤ 1 second for cached data**.
* Charts should render with **≤ 500ms latency**.
* System should handle **up to 100 GitHub API requests per minute**.

## **4. Appendices**

* **Prototype:** [Live Demo](https://chatgpt.com/c/67ec35c9-2ae8-800f-894b-e322425a09a2#)
* **Project Roadmap:** [Notion Todo Roadmap](https://chatgpt.com/c/67ec35c9-2ae8-800f-894b-e322425a09a2#)
* **Wireframes:** [WireFrames Documentation](https://chatgpt.com/c/67ec35c9-2ae8-800f-894b-e322425a09a2#)
* **Flowchart:** [GSoC App Metric Flowchart](https://chatgpt.com/c/67ec35c9-2ae8-800f-894b-e322425a09a2#)