**Software Design Description**

**SysAnatomy: A Reporting Tool for Linux OS**

Version <1.0>

3rd March 2016

Contents

[**1.** **Introduction** 3](#_Toc445382488)

[1.1 Purpose 3](#_Toc445382489)

[1.2 Scope 3](#_Toc445382490)

[1.3 Audience 3](#_Toc445382491)

[1.4 Document revision history 4](#_Toc445382492)

[**2.** **Design overview** 4](#_Toc445382493)

[2.1 Objective 4](#_Toc445382494)

[2.2 Approach 4](#_Toc445382495)

[2.3 Design Overview 5](#_Toc445382496)

[**3.** Topology Design 6](#_Toc445382497)

[**4.** External Design 6](#_Toc445382498)

[**5.** Internal Design 7](#_Toc445382499)

[5.1 Hardware / Software Requirements 7](#_Toc445382500)

[5.2 Development Environment 7](#_Toc445382501)

[5.3 Proposed Components 7](#_Toc445382502)

[5.3.1 Process Specific Data 7](#_Toc445382503)

[5.3.2 CPU/Memory Specific Data 8](#_Toc445382504)

[5.3.3 Report Generation 8](#_Toc445382505)

[5.4 Error Handling & Recovery 8](#_Toc445382506)

[5.5 Testing 8](#_Toc445382507)

[5.5.1 Test Environment 8](#_Toc445382508)

[5.5.2 Unit Testing 8](#_Toc445382509)

[5.5.3 Functional Testing 8](#_Toc445382510)

[5.5.4 UX Testing 8](#_Toc445382511)

[**6.** User Interface Design 9](#_Toc445382512)

[a. Home Page 9](#_Toc445382513)

[**7.** Database Design 10](#_Toc445382514)

[**8.** Future Scope 11](#_Toc445382515)

[7.1 Storage of User profiles 11](#_Toc445382516)

[7.2 Latest Software Version 11](#_Toc445382517)

[**Appendix A: Acronyms and Abbreviations** 12](#_Toc445382518)

[**Appendix B: Products & Tools** 13](#_Toc445382519)

# **Introduction**

## Purpose

The purpose of this design document is to outline the technical details of an online metrics collection tool. Furthermore, it provide an overview for the implementation of online metrics collection tool.

Its main purpose is-

1. Provide the link between functional specification and technical details.
2. Provide detailed description and functionality of various components and their interactions.

## Scope

The system collects metrics related to all applications, processes running in one particular device and stores it in a relational database. The system holds data associated with all active processes in a device. This data is processed, organized and stored in a single relational database, giving access to key performance metrics on a customizable, real-time dashboard.

## Audience

This document is intended for the project sponsor, project development team, technical architect, database designer and quality assurance team.

Project sponsor will critique and guide through the phases of the project and provide important tweaking suggestions and tips.

The project development team will use this document as a reference for development. This will make sure that everyone in the team is on the same page and work efficiently.

Testers are responsible for making sure that the final product conforms to the design document.

## Document revision history

|  |  |
| --- | --- |
| **Major Changes** | **Date** |
|  |  |

# **Design overview**

## Objective

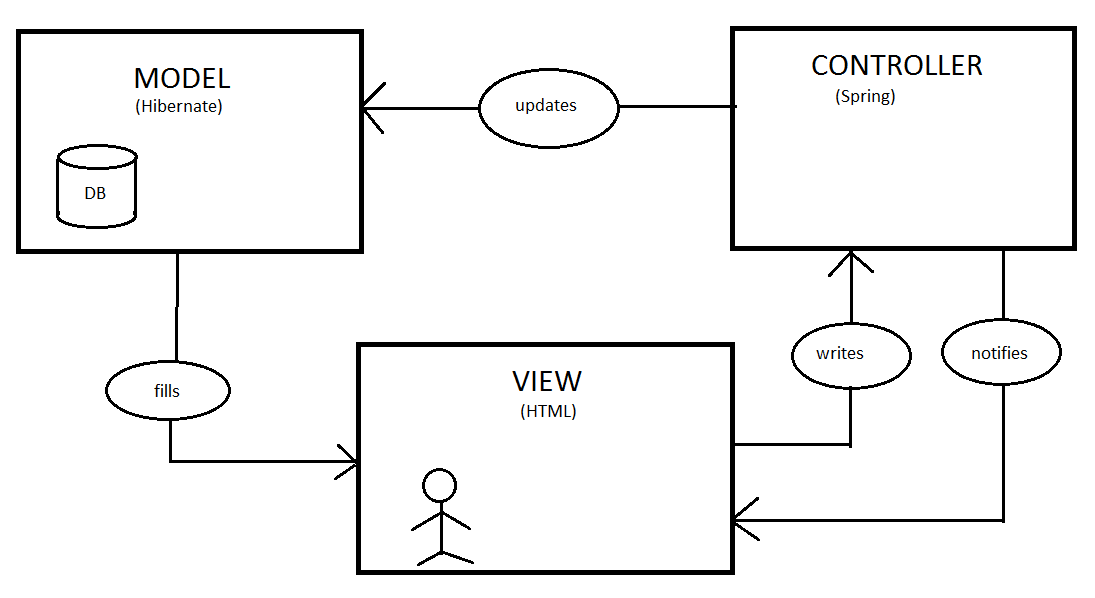
Provide a platform that enables the users to better understand the co-relation between the programs running in their devices with the performance of the system.

## Approach

This document is created and will be updated during multiple phases of the project life cycle. The project follows an Agile Methodology for development. Agile method of product life cycle helps to adapt to the unpredictable changes in requirements or otherwise in the product.

This methodology consists of two weekly Scrum meetings to keep track of updates and discussion about achieving goals. Scrum meetings are conducted by scrum master, who may change every two weeks. Scrum master acts as a reporting authority for collecting information from each team member, document them and manage them in bug tracker. Every cycle's output and goals are discussed with the project sponsor, who when approved ensures implementation is being carried out as per the requirements.

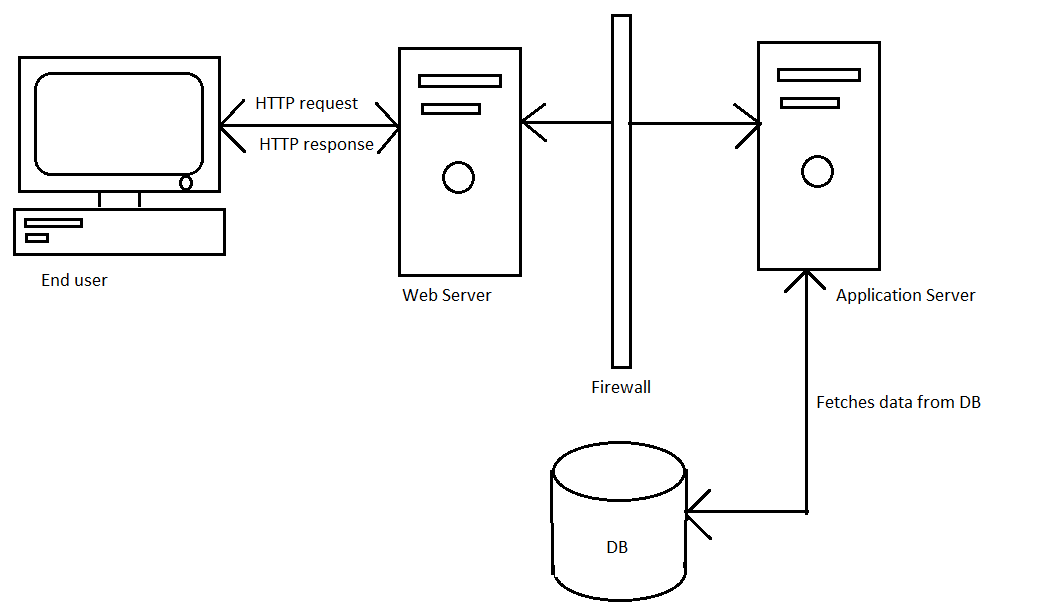
## Design Overview



The above diagram shows the Model View Controller architecture of the system being developed. The end user generates HTTP requests from a web browser. This request is analyzed and processed by the MVC to produce an appropriate response, which is returned back to the user. The internal functioning is hidden from the end user. The user can only see the view as shown in the diagram above.

The controller works as the manager who directs and regulates all the functionalities. It makes changes to the model and notifies the view about the same. The view updates itself and renders a fresh view.

# Topology Design



As shown above, the end user uses his device to connect to our servers via HTTP requests and responses. He interacts with the forms and generates new data that is to be processed and stored in the Database.

The Front-end or UI would be made using HTML, JQuery and JavaScript. The view has the important role of rendering the model data. It also requests updates from the model.

The Controller of the MVC model would make use of JSP to implement the business logic. It would serve functionalities such as defining the application behavior. It would map http requests to model updates. Further, it would be responsible for selecting the view as response for each functionality.

While the Model component of the MVC will be implemented using Spring and Hibernate in Java. It encapsulates application data and application functionality. It also notifies the view of any of the changes. Additionally, it provides interface for state queries.

The data collected from the user’s devices will be stored in SQL Server, a Relational Database. This data will then be processed and analyzed to generate reports and metrics.

# External Design

# Internal Design

# Hardware / Software Requirements

|  |  |
| --- | --- |
| **CPU** | X86 or higher, any speed. |
| **Operating System** | Linux. |
| **Main Memory** | Minimum: 8MB  Recommended : 16 MB |
| **Display** | Color Required.  Minimum : 256 Colors  Recommended: 16K colors are better. |
| **Disk Space** | As required for any browser. |

# Development Environment

|  |  |
| --- | --- |
| **Compiler** | javac |
| **IDE** | NetBeans |
| **Source code repository** | https://github.com/rsaranya/Project-SysAnatomy |
| **Defect Tracker** | https://github.com/rsaranya/Project-SysAnatomy/issues |
| **Build Process** | Maven |
| **Release Process** | Deploy in a Web/ Application Server like JBoss or Apache Tomcat |

# Proposed Components

## Process Specific Data

This module deals with collection of data related to processes running on the system. This includes the identifier of the process, its parent information, name, memory utilization and the owner. This information gives an insight into each process and their behavior in the system.

Extracting this data, cleaning and transforming it into a generic format helps in converting data into information. This information can be used for generating reports by calculating various metrics. These metrics helps us in understanding the effect the process has in the system or vice versa.

## CPU/Memory Specific Data

This module deals with collecting CPU related data like memory utilization, processing speed, number of processors available for processing, support for hyper threads, logical processors This information gives an insight into how the system is performing.

Extracting patterns and variations in the CPU/memory related data to represent in graphical format like heat maps and graphs for easier understanding.

## Report Generation

All the collected metrics can be used to create customize able reports, which can be printed, saved or exported to csv format.

# Error Handling & Recovery

Apache Log4j 2, an open source logging system will be used to generate logs for the system. The logs can be used to detect failures, if any, during the working of the application. The logs generated will be stored at the server.

# Testing

* + 1. **Test Environment**: Multiple LINUX operating systems.
    2. **Unit Testing**: As and when the modules are developed they will be tested.
    3. **Functional Testing**: Test suites will be defined in this phase using Junit to test the functional correctness of each module.
    4. **UX Testing:** On deployment the system will be tested by peers to get appropriate feedback on their perception.

# User Interface Design

## Home Page

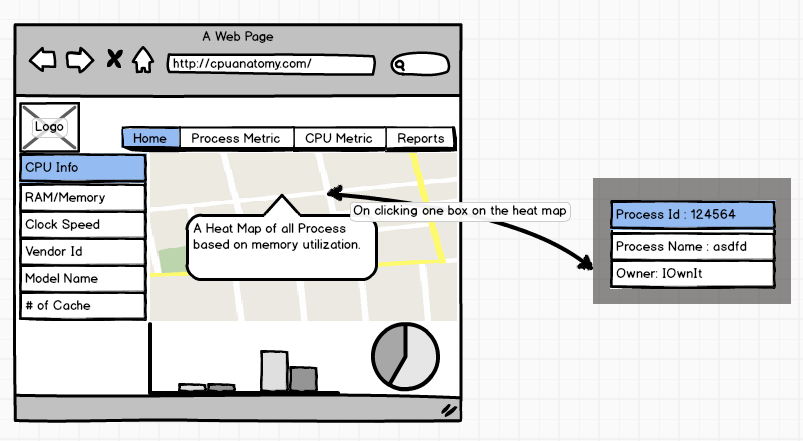


Fig5.1: Main Page containing links to other pages. Contains a snapshot of users device information

# Database Design

***Table Name:*** REQUEST

|  |  |  |
| --- | --- | --- |
| RQ\_SYSTEM\_ID | VARCHAR(50) | unique system identifier |
| RQ\_TOKEN\_ID | INT | auto-generated token id |

***Table Name:*** PROCESS\_INFO

|  |  |  |
| --- | --- | --- |
| RQ\_TOKEN\_ID | INT | token id to track a session |
| PI\_PROCESS\_ID | INT | process id for a particular process |
| PI\_PROCESS\_NAME | VARCHAR(50) | name of the process |
| PI\_PPID | INT | parent process id |
| PI\_PROCESS\_OWNER | VARCHAR(50) | owner of the process |
| PI\_MEM\_USAGE | VARCHAR(50) | memory usage of the process |
| PI\_RAM | VARCHAR(50) | speed utilization |
| PI\_CLOCK\_SPEED | VARCHAR(50) | clock speeds |
| PI\_THREAD | VARCHAR(50) | number of active threads |
| PI\_CACHE | VARCHAR(50) | cache usage for the process |

***Table Name:*** CPU\_INFO

|  |  |  |
| --- | --- | --- |
| CI\_TOKE\_ID | INT | token id to track a session |
| CI\_IP\_ADD | VARCHAR(50) | ip address of the user |
| CI\_NUM\_PROC | VARCHAR(50) | number of processors on the user cpu |
| CI\_MEM\_SIZE | VARCHAR(50) | memory size of the user cpu |
| CI\_RAM | VARCHAR(50) | speed of the user cpu |
| CI\_CLOCK\_SPEED | VARCHAR(50) | clock speed of the user cpu |
| CI\_THREAD | VARCHAR(50) | number of threads running on the cpu |
| CI\_CACHE | VARCHAR(50) | amount of cache available on cpu |

# Future Scope

## Storage of User profiles

This module deals with generating metrics that compare the users registered devices and generate reports using this information. A unique identifier will be used to keep track of user specific information.

Each visitor to the site may choose to create an account with us. This way he/she can register multiple devices with us. We will provide him with a useful and detailed comparison of various metrics for all his devices that run our application.

## Latest Software Version

Provide with information about latest updates available for various applications installed in the user’s device.

# **Appendix A: Acronyms and Abbreviations**

|  |  |
| --- | --- |
| **HTML** | Hypertext Markup Language |
| **MVC** | Model View Controller |
| **UI** | User Interface |
| **HTTP** | Hypertext Transfer Protocol |
| **JSP** | Java Server Pages |

# **Appendix B: Products & Tools**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Software/  Tool | Version | | Source | Description |
| JRE | | 8 | http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html | Runtime Environment to run Java Programs |
| Maven | | 3.3.9 | https://maven.apache.org/ i | Build manager for the Java Program |
| Eclipse | | 4.5.1 | https://eclipse.org/ | IDE for developing Java based programs |
| Tortoise git | | 1.8.16.0 | https://tortoisegit.org/ | User Interface for operating Git |
| Git hub | | 10.9 | https://github.com/ | Repository for source code |
| Creatly | | - | https://creately.com/ | Designing software used to create Use Case Diagrams |