**WFD PATCH MANAGER DESIGN DOCUMENT**

**(An automated solution for applying WFD patches)**

|  |  |
| --- | --- |
| Version | 1.0 |
| Author | [mayur.kapoor@kronos.com](mailto:mayur.kapoor@kronos.com) [sagar.saxena@kronos.com](mailto:sagar.saxena@kronos.com) |

**WFD Patch Manager**

**Introduction**WFD patch manager is automated one-click solution for applying patches in WFD services (WFM, Reporting, Open AM, PostgreSQL etc). It will provide a user interface layout which interacts with user and perform required patching actions parallelly on multiple Right-Scale deployments on behalf of user.

**Audience**

Release Team, One support Team, Automation Team

**Intent to deploy**

* To apply patches in a project (CUST/CFN) in a single click
* To get rid of manual intervention
* To avoid any type of human error while applying patches
* To reduce time and effort both, during patching activity

**Technology/Platform**

Python 3.8.3

Python flask Framework

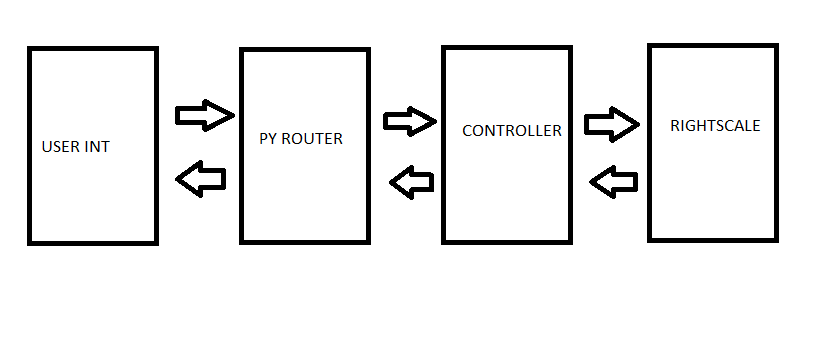
REST API

HTML/CSS

**Application Workflow (Module-wise)**

Application has three major components.

* User Interface
* Flask Router
* Controller



User Interface takes inputs from User Form, brings it to python flask router which according to the requirement, calls Controller Module.

Controller Module uses REST APIs to fetch details from Right-Scale and performs post actions on the Right-scale.

Controller brings back the response or details of the Right-Scale resources to Python Router and which sends it to User Interface.

**In Brief Application Workflow**

**Pre-requisites**

User needs to create patch right script (Shell/Python/Ruby etc) before using WFD Patch Manager.

Right-Scale token should be kept handy to login on WFD Patch Manager.

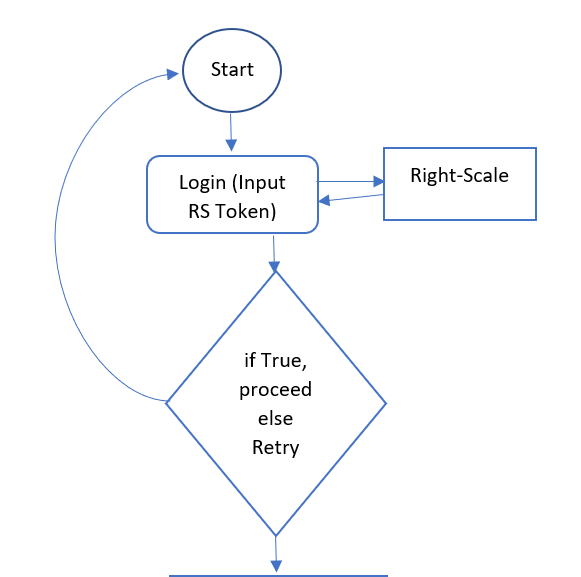
Below is the explanation of each phase with a phase wise flowchart on the next page followed by complete flowchart of the application.

Lines in blue are in-brief explanation of how particular phase works.

**Login Phase:**

User needs to hit application URL, a user form will ask for Refresh Token and on entering user will be authenticated.

* An API POST call will be executed which brings access token (valid for 2 hrs.) with respect to Refresh Token entered, if API response is successful control moves to Discovery Phase else it will go for Retry Page.
* On successful Response, access token will be stored in session parameter for future use.

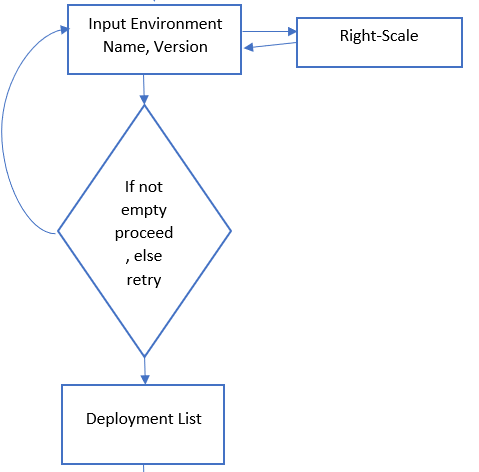


**Discovery Phase:**

User needs to enter filters for service name and version. All Deployments which match the filters will be fetched and displayed to user in tabular format.

* An API GET call will be executed with filters entered, List of Deployment Objects will be received and displayed on HTML page.

***Note:* *This phase can be modified to discover according to Right-scale deployment tag and it depends upon the user requirement.***

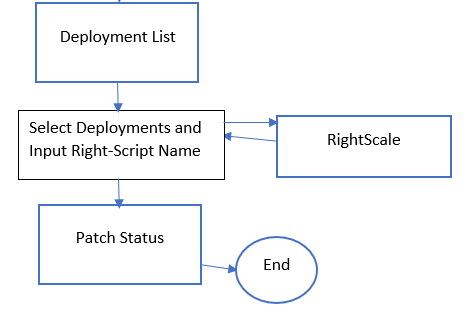


**Execution Phase:**

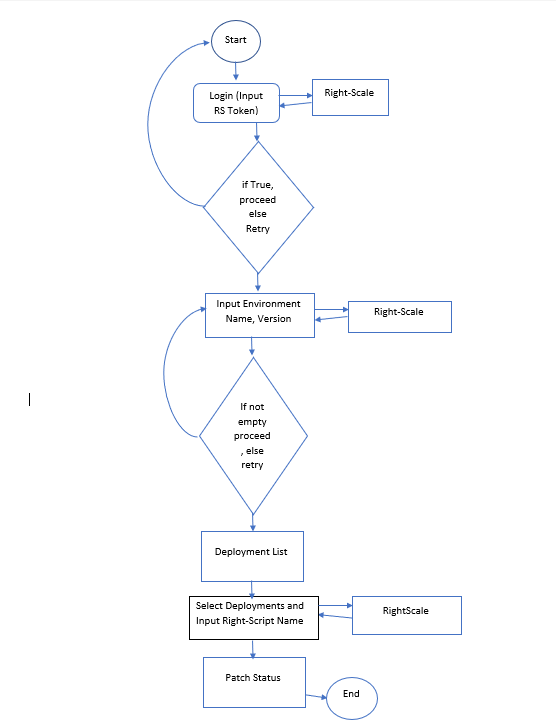
User needs to select deployments on which patch is required to be applied and enter Right-Script Name.

On confirming, Right-Script with entered name will get executed on all selected Deployments.

* An API POST call will be executed with list of selected deployments, Right-script name and Right-script will be executed on selected deployments.

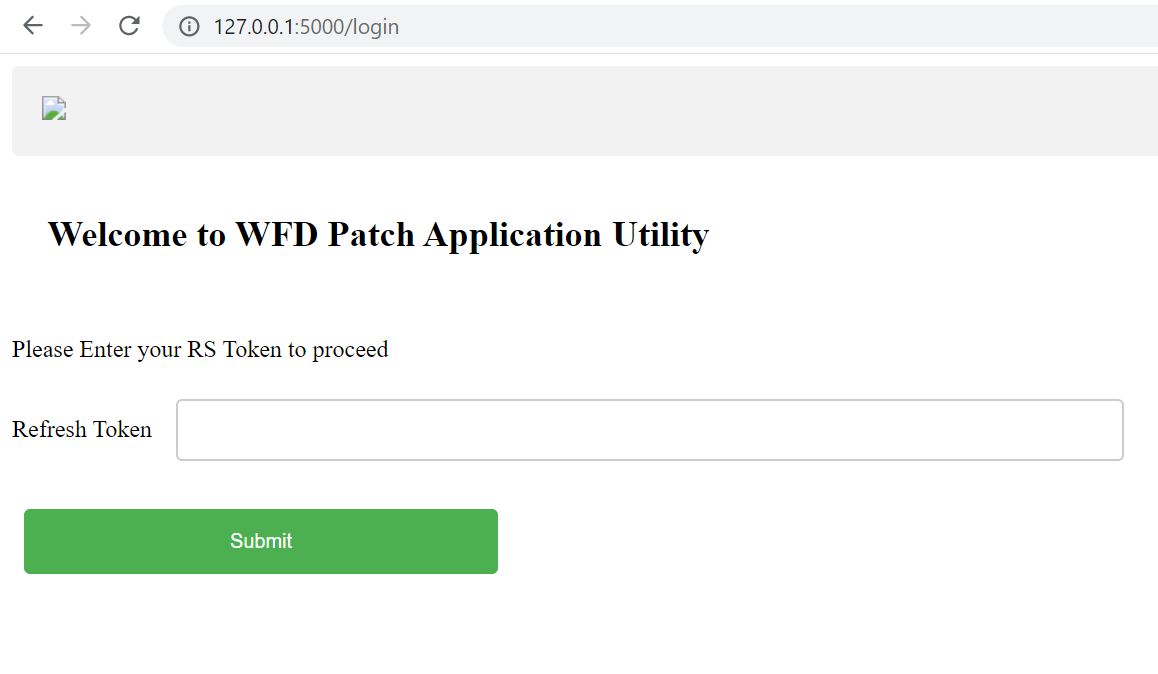


**Complete Flowchart of the Application**

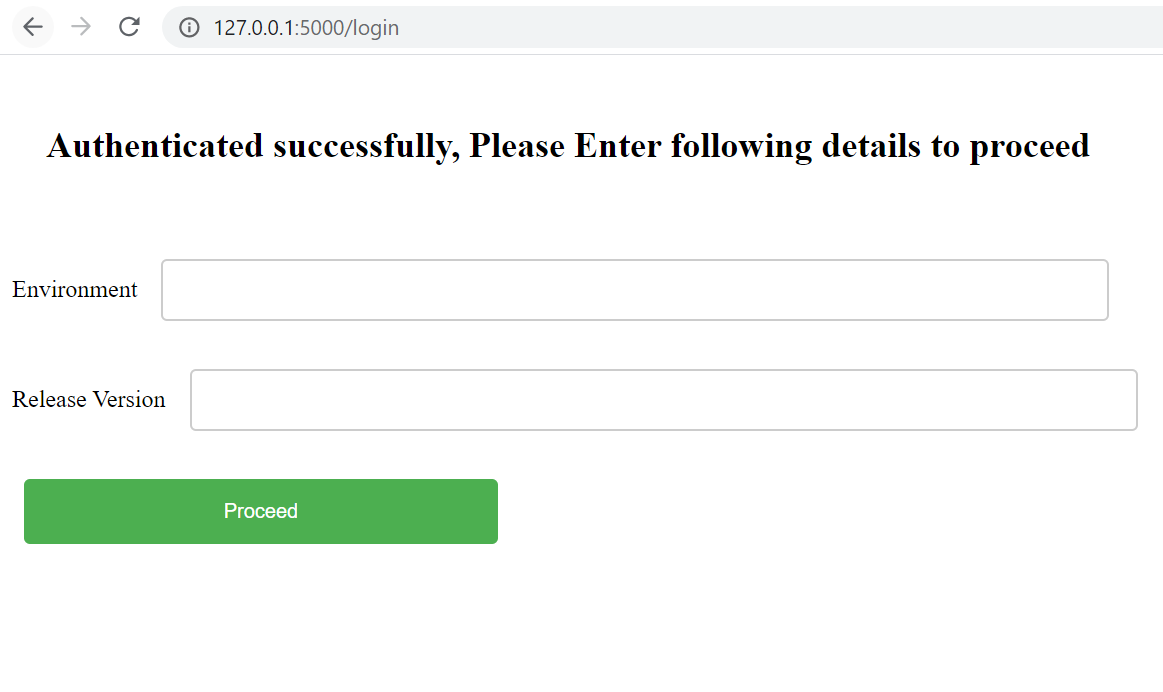


**SNAPSHOTS OF ACTUAL PROJECT WORKFLOW**

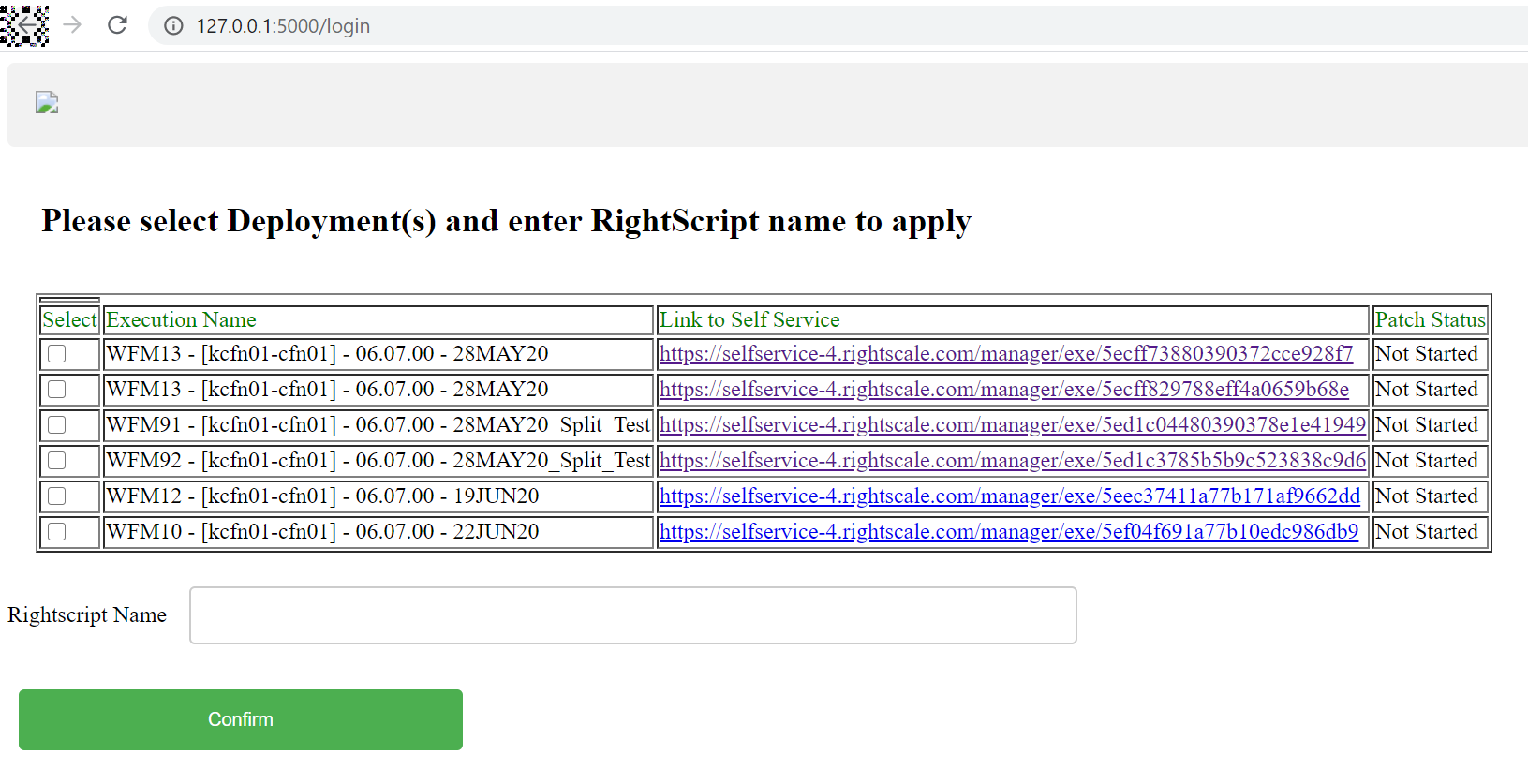
**Login Phase:**



**Discovery Phase:**



**Execution Phase:**



**This project is ready for WFM service right now and can be later extended for other services as well.**

**This document has been designed to explain the basic workflow of the project, a high-level design can be prepared if this design gets approved.**

**Current Patch Use-Cases**