Dive Manager

Table Of Contents

**[1. Dive Manager](https://docs.spring.io/spring-boot/docs/current-SNAPSHOT/reference/htmlsingle/" \l "boot-documentation) Overview**

**[2. What](https://docs.spring.io/spring-boot/docs/current-SNAPSHOT/reference/htmlsingle/" \l "getting-started) Dive Manager used for**

**[3. Core](https://docs.spring.io/spring-boot/docs/current-SNAPSHOT/reference/htmlsingle/" \l "using-boot) Dive Manager Concepts**

**4. Dive Manager Tutorial**

**5.** **Technical requirement**

## [1. Dive Manager](https://docs.spring.io/spring-boot/docs/current-SNAPSHOT/reference/htmlsingle/" \l "boot-documentation) Overview

## Dive Manager is user for the Administration and Management of services.

## [2. What](https://docs.spring.io/spring-boot/docs/current-SNAPSHOT/reference/htmlsingle/" \l "getting-started) Dive Manager used for

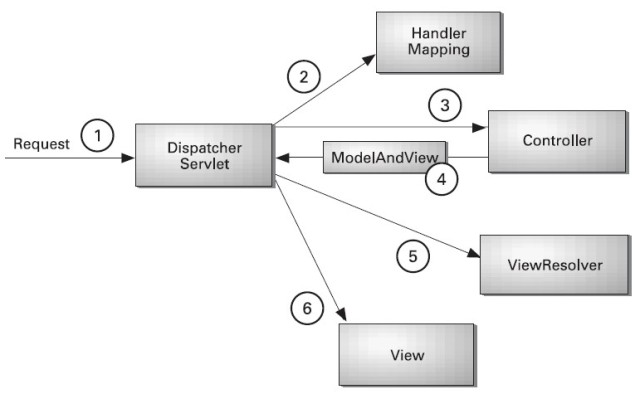
## Using this application we can

* Dashboard – Display Grafana Graphs
* Resource Management
* Predictive Models - Predictive Modeling using OpenScoring-Engine – a PMML approach
* Lamda Models - Publish a machine learning engine to the web as a service using Apache PredictionIO
* Manage Ldap User
* Schedule Job

## [3. Core](https://docs.spring.io/spring-boot/docs/current-SNAPSHOT/reference/htmlsingle/" \l "using-boot) Dive Manager Concepts

Dive Manager is developed using Spring Web MVC framework.

* Spring’s web MVC framework is, like many other web MVC frameworks, request-driven, designed around a central Servlet that dispatches requests to controllers and offers other functionality that facilitates the development of web applications.
* Spring’s DispatcherServlet however, does more than just that.
* It is completely integrated with the Spring IOC container and as such allows you to use every other feature that Spring has.



**Figure 3.1. The request processing workflow in Spring Web MVC (high level)**

* As displayed in the figure, all the incoming request is intercepted by the DispatcherServlet that works as the front controller.
* The DispatcherServlet gets entry of handler mapping from the xml file and forwards the request to the controller.
* The controller returns an object of ModelAndView.
* The DispatcherServlet checks the entry of view resolver in the xml file and invokes the specified view component.

## 4. Dive Manager Tutorial

Dive manager is integrated with LDAP login. After we login we get the page where we can view the Models published, Scheduler List and Web service assigned to User List. It is used for management below tasks.

* **Dashboard**
* **Resource Management**
* **Predictive Models**
* **Jobs**
* **Lambda Models**
* **User**

**Dashboard**

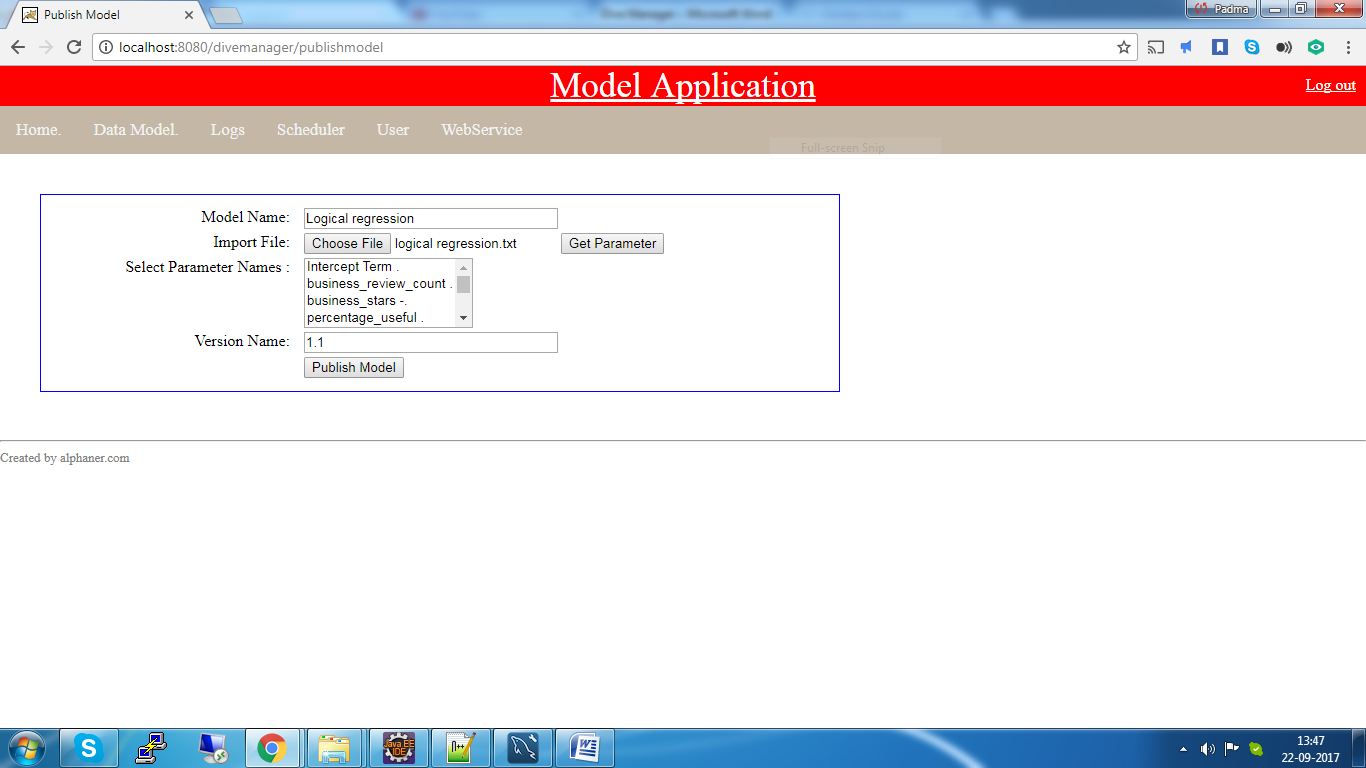
After logging in Dive Manager Dashboard page is dispalyed shown below. It contains system details, show grafana graphs and Displays docker containers services and Images.

**4.1. Data Model**

Data Model is used Publish Model.

* In this page we select model file published in an algorithm and select the needed parameters.
* After submission the data is stored in database with selected parameter with the related values.

Below the view of Data Model .

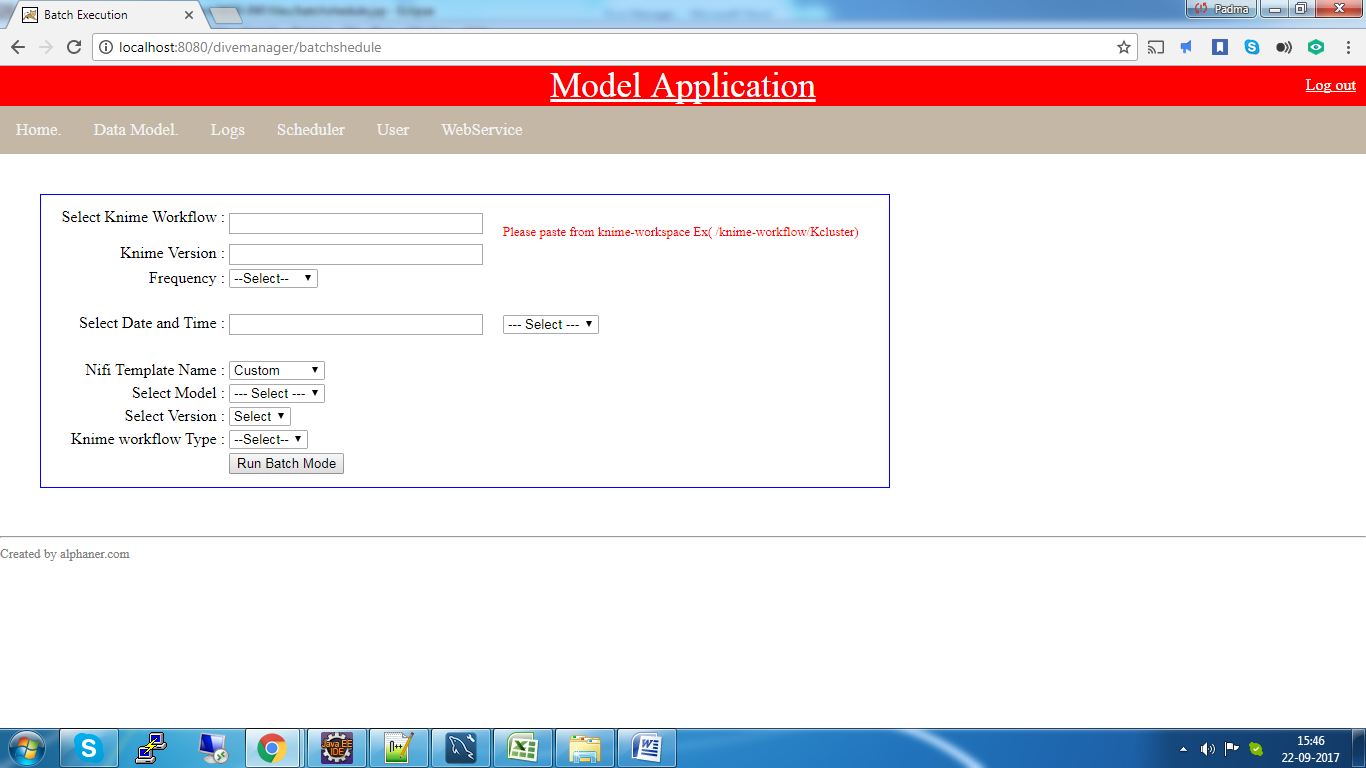


**Figure 4.1. Data Model**

**4.2. Scheduler**

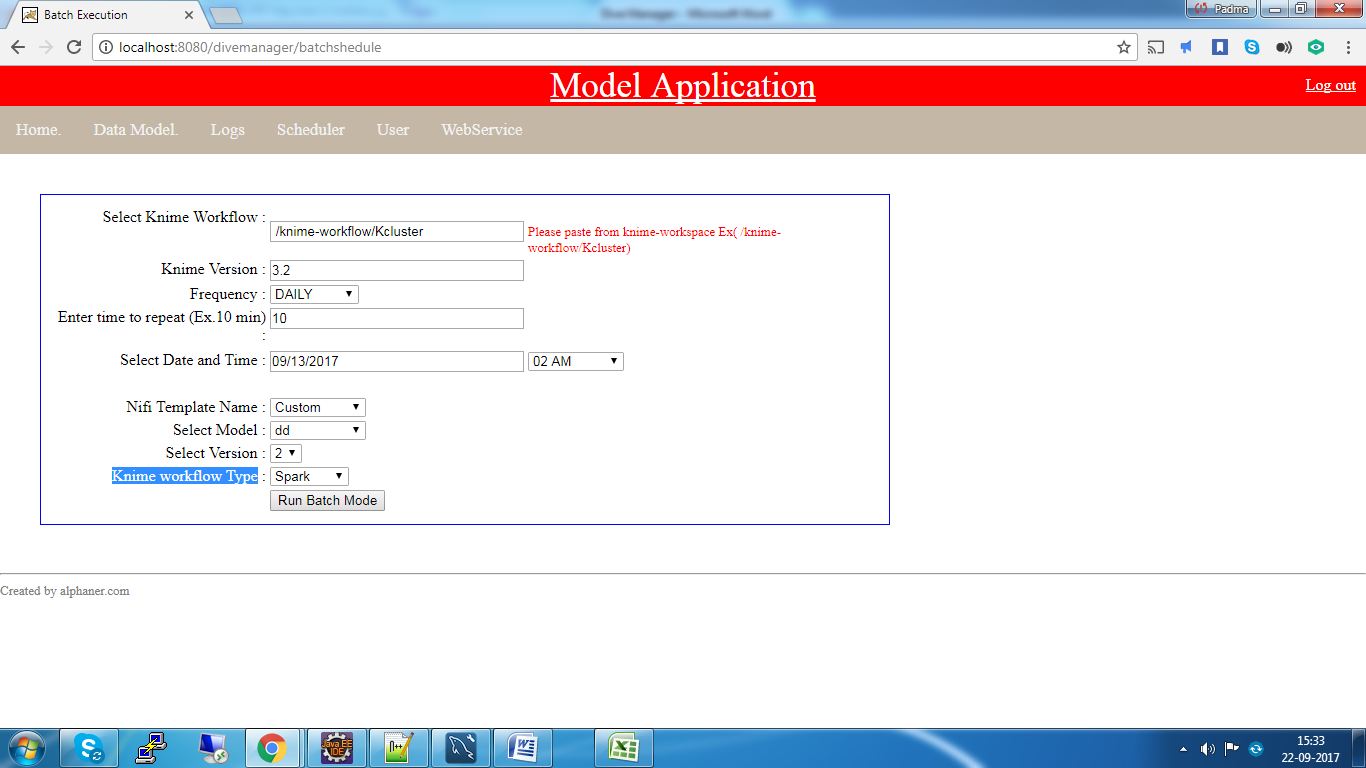
The Scheduler is used to create Batch Scheduler using the below parameters.

* *Select Knime Workflow*
* *Knime Version*
* *Frequency*
* *Enter time to repeat (Ex.10 min) (Hidden)*
* *Select Date and Time*
* *Nifi Template Name*
* *Select Model*
* *Select Version*
* *Knime Workflow Type*



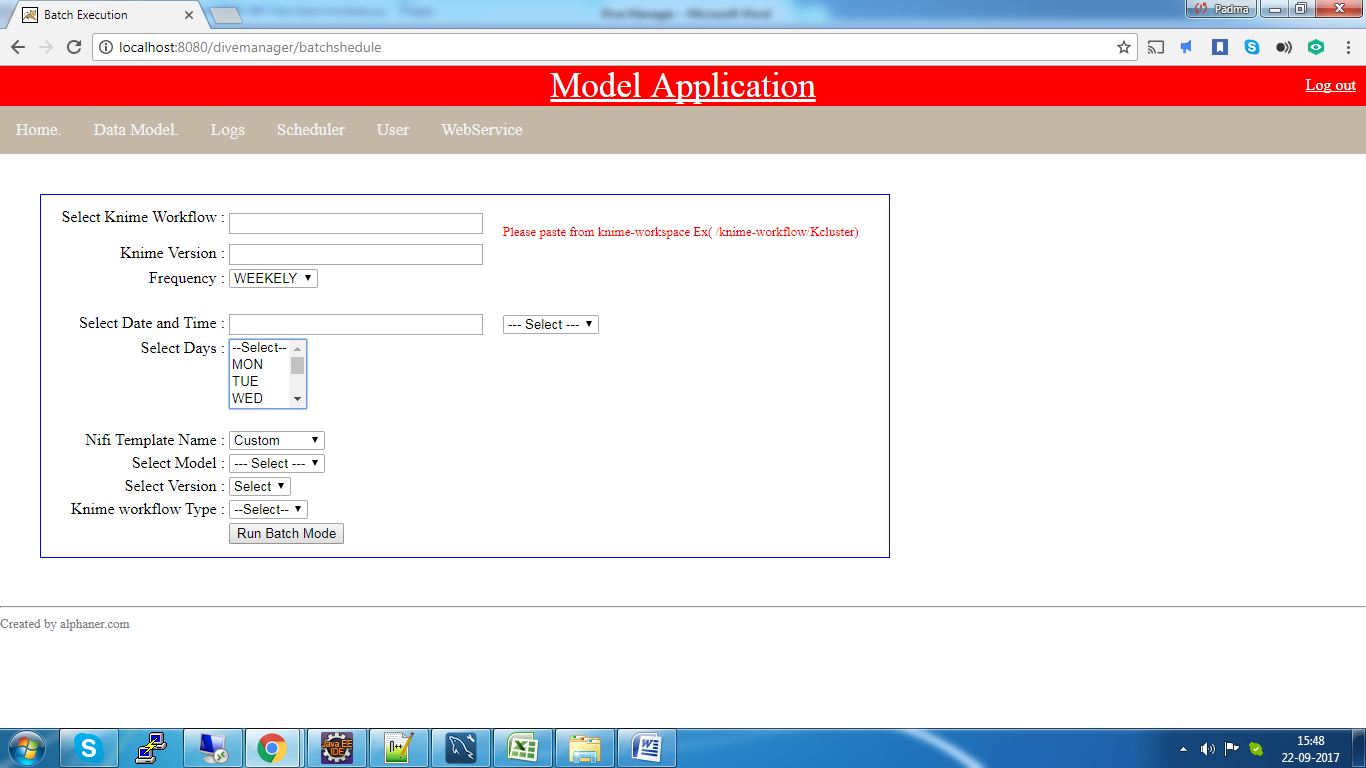
**Figure 4.2. Batch Scheduler**

* User need to enter the path of Knime Workflow starting from "**/knime-workspace".**
* Version of Knime installed in system should be entered.
* Frequency field has three parameters. **DAILY,WEEKLY and MONTHLY**
* If the frequency is DAILY user need to enter the time to repeat field. ex 10 shown below



**Figure 4.2. 1 Batch Scheduler (When Frequency is DAILY)**

* If the frequency is WEEKLY user need to select the days of week in the field show



**Figure 4.2. 2 Batch Scheduler (When Frequency is WEEKLY)**

* User needs to select date and time for scheduler run
* Nifi Template consists of two types ex. Standard and Custom.
* User can select the model published y him.On selection of model vesrion name would be dispalyed.
* Knime workflow Type has four option to select. **Knime,R,Spark and Pyhton**

After entering of all the mandatory fields user need to click on **Run Batch Mode**.

Once the operation is Successful following two files would be created in system.

**1. Knime\_KMeans\_Clustering\_2.0.txt** (ProductName\_VresionName.txt)

This is the text file created in the below path

**/root/nifi-1.3.0/parameter**

The file contains the command to run the Batch Scheduler.

ex.

knime/knime\_3.4.0/knime -consoleLog -application org.knime.product.KNIME\_BATCH\_APPLICATION -nosplash -workflowDir="/knime/knime-workspace/Knime-KMeans"

**2.** **Knime\_KMeans\_Clustering\_2.0.ksh** (ProductName\_VresionName.ksh)

The script file is created in below path

**/root/nifi-1.3.0/parameter/shellscript**

The file contains the below contents.

ex.

load,DAILY

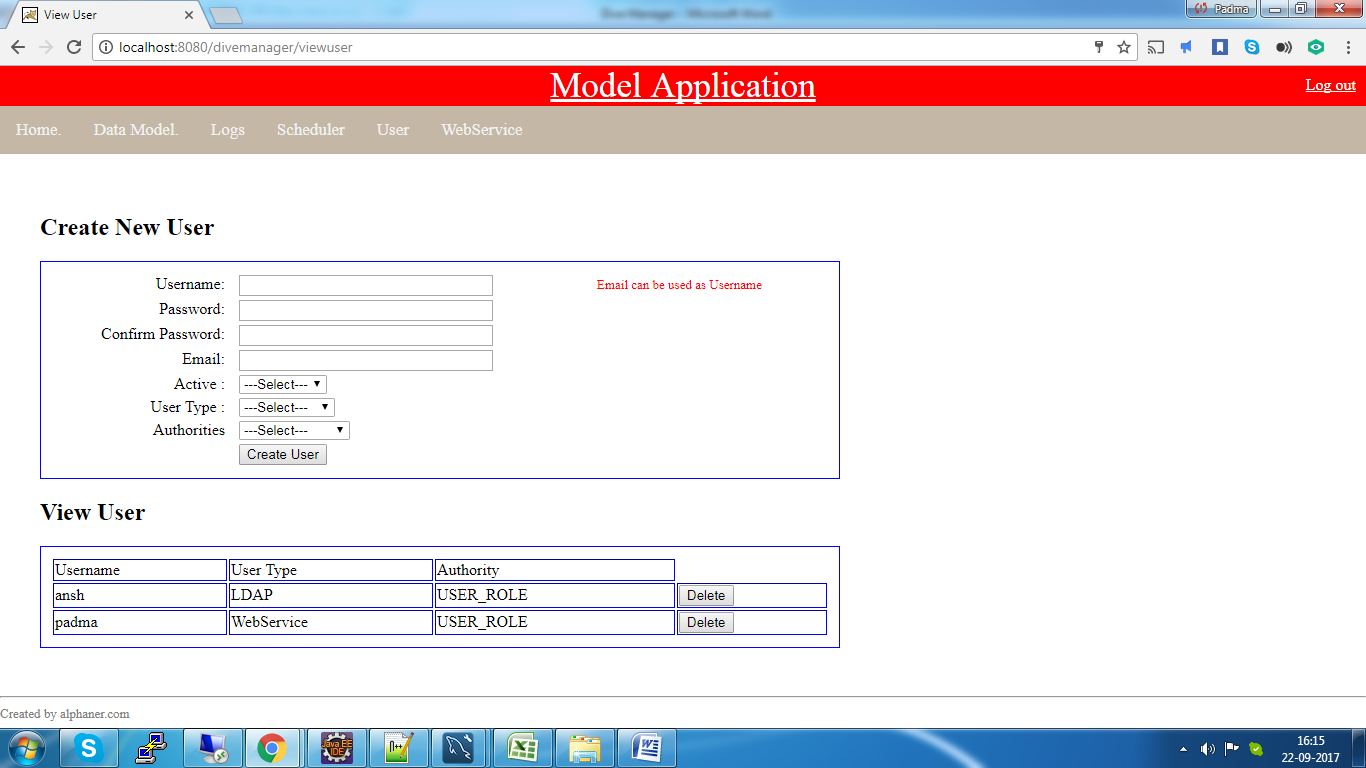
submitjob,/root/nifi-1.3.0/parameter/shellscript/Knime\_KMeans\_Clustering\_2.0.ksh

#/knime/knime\_3.4.0/knime-consoleLog-application org.knime.product.KNIME\_BATCH\_APPLICATION -nosplash -workflowDir="/knime/knime-workspace/Knime-KMeans"

**4.3. User**

In the user page we can create a new user and also manage the user accounts.

User page looks as below.



**Figure 4.3.1 User**

User page contains

* Create New User
* View User

**4.3.1 Create New User**

In the create new user we can create two types of users. One is LDAP and other is web service. To create new user we need to enter below fields.

* *Username*
* *Password*
* *Confirm Password*
* *Email*
* *Active*
* *User Type*
* *Authorities*
* We can user email as Username field .
* Password for authentication .
* Email address of user.
* We can enable or disable using Active field.
* Authorities has the option to select if the user is admin or normal.

After successful creation of user ,if the user is LDAP the below text file would be created .

**userinfo.txt** in the following location.

It would contain username and email address of user.

**4.3.View User**

This would displays the list of users created. We have option to delete the selected user.

**4.4. *Web Services***

* We need to proved user to permission to access web services.
* Need to select a user in selection field where the all the users of type web service would be displayed.
* Once the user selected ,need to specify which model name with respective version would be selected.
* Clicking on Create Model Service would allow the selected user to access specified web service.
* Below is the page which shows the creation.



**Figure 4.4.1 Webservice**

**5 Technical requirement**

## 5. 1 Client requirement

As Dive Manager is a web application it could be run on most of custom operating systems via a web browser. We tested PATH with Chrome 1.0 and IE 7.0 and found no problem.

## 5.2 Server Requirement

The following technologies are required for Dive Manager server.

* Mysql 5.1
* Java 1.7
* Tomcat 5