**PROJECT DOCUMENTATION**

SUBJECT**:** **Pelton Turbine Experiment Simulation using javascript**

PREPARED BY**: Jatin Kumar**

**1. INTRODUCTION**

The Pelton Turbine Experiment Simulation is the JavaScript version of the pre-existing simulator programmed in Java3D. The simulator runs on all the modern web-browsers like Chrome, Firefox, Edge etc., and does not need any requirement of any third party application to run, unlike the previous version.

**The following is the information related to the Tech-Stack used in the project:**

* Programming Language Used: **JavaScript**
* Markup Language Used: **HTML5**
* Style-Sheet Language Used: **CSS3S**
* External Libraries/Frameworks Used: **SVG.js, Canvas.js and jQuery.**

**FILES:**

1. Codes/rotate.html
2. Codes/style.css
3. Codes/java.js
4. Libraries/svg.js
5. Libraries/jquery.js
6. Libraries/canvasjs.min.js

**2. OVERALL DESCRIPTION**

**2.1 BUTTONS:**

**2.1.1 START BUTTON:**

FUNCTIONALITY: Starts the animation of the simulation.

IMPLEMENTATION: The “START” button has been given ID “start” in the rotate.html file.The click on the button leads to function call for the function “animate()” and also passes the id of button ,the function call begins the simulation and the button gets disabled after the click.

**2.1.2 RESET BUTTON:**

FUNCTIONALITY: Refreshes the webpage rotate.html

IMPLEMENTATION: The “RESET” button has been given ID “reset” in the rotate.html file. The click on the button leads to function call for the function

“reload()” which reloads the webpage which resets the simulation and enables the start button once again.

**2.1.3 “CALCULATIONS” BUTTON:**

FUNCTIONALITY: Hyperlink to the Calculations section of the webpage.

IMPLEMENTATION: The “CALCULATIONS” button has been given ID “calc” in the rotate.html file. The button act as a hyperlink to jump to calculation section of the webpage with id=”calculation-buttons”.

**2.1.4 “SUBMIT” BUTTON**:

FUNCTIONALITY: It is input element which submits the values entered in the from and verifies that neither a field is empty nor contains a negative value.

IMPLEMENTATION: The “SUBMIT” button has been given ID “submit” in the rotate.html file. On click of this button, which is of input type the values are checked for empty values and negative values and get store.

**2.1.5 “RESET” BUTTON:**

FUNCTIONALITY: Clears the values of the form and the graph.

IMPLEMENTATION: The “RESET” button has been given ID “reset” and its button type has been set to “button” in the rotate.html file. The click on the button leads to function call for the function “reload()” which reloads the webpage which resets the simulation.

**2.1.6 “PLOT GRAPH” BUTTON:**

FUNCTIONALITY: Clears the values of the form and the graph.

IMPLEMENTATION: The “RESET” button has been given ID “reset” and its button type has been set to “button” in the rotate.html file. The click on the button leads to function call for the function “reload()” which reloads the webpage which resets the simulation.

**2.2 THE SIMULATION**

The simulation mainly SVG.js library to draw the outlines and the perform horizontal or downward water animations of the experiment. Upward water animations have been programmed using jQuery.

**2.2.1 HTML STRUCTURE**

The simulation has been programmed for mainly two elements in the rotate.html file namely <head> and <body>.

The <head> part includes the script files including JavaScipt libraries used and the JavaScript script used.

The <body> part contains two <div> elements containing the simulation part and the calculation + graph part.

**2.2.2 OUTLINES OF THE EXPERIMENT (SVG.js)**

IMPLEMENTED BY: Function **construct()** in script.js file.

An SVG drawing variable “value” has been declared in the java.js file as a global variable which creates an SVG area of size 1800x800 px.

*polyline() / polygon() functions of SVG.js have been used to draw setup.*

**2.2.3 WATER ANIMATION**

IMPLEMENTED BY: Function **animate().**Function animate() is called when the onclick event of “START” button occurs.

**Part 1: USING SVG.js**

For horizontal and downward water animations and the rotation of the turbine.

**Concept used:** Water of a particular shape is drawn for a negligible size (not visible) and its dimensions are increased in a direction using animate().size() function of SVG.js

**Part 2: USING jQuery**

For upward water animations

**Concept used:** These <div> elements have been styled in style.css file initially with negligible height. After respective waiting time for each element, height changes accordingly using animation() function of jQuery, in the animate() function inside java.js

**2.3 CALCULATIONS**

**2.3.1 PERFORMING CALCULATIONS**

IMPLEMENTED BY: Function b**()**.

A form for filling the values of experiment has been created by using HTML and CSS. The values of the placeholders are used in the b() function by using DOM property.

Variables Used for Reading 1:

1. Q (Water Rate)
2. H (Water Head)
3. n (Revolution Frequency)
4. x (Length)
5. m (Mass)

Variables Used for Reading 2:

1. Q2 (Water Rate)
2. H2 (Water Head)
3. n2 (Revolution Frequency)
4. x2 (Length)
5. m2 (Mass)

The form ensures that the values entered are positive and not null.

The **input power (input\_power)** and the **output power (output\_power)** are calculated using the variables above.

The answer,**efficiency** of the Pelton Turbine is calculated using the ratio of the output power to the input power.

**2.3.2 GRAPH (USING Canvas.js)**

HTML: Placed inside the <div> element with ID “chartContainer”

IMPLEMENTED BY: Function **b()**.

plotGraph() function takes four variables as arguments -

1. input\_power
2. input\_power2
3. output\_power
4. output\_power2

When Plot Graph button is clicked without any values filled in the form,an empty graph gets plotted and the graph between input\_power and output\_power gets plotted when all the values are filled properly.