Development of Web Application For Virtual Laboratory

Kausar Fatima[1], Bikranta Sarkar[2], Suhrid Das[3], Shankha Shubhra Mukherjee[4], Sudip Mandal[5*], Mirwaiz Rahaman[6]
[1, 2, 3, 4, 5, 6] Dept of Electronics and Communication Engineering
[1, 2, 3, 4, 5, 6] Jalpaiguri Government Engineering College, Jalpaiguri, India, PIN-735102

Abstract- *A Web Application is employed that attempts to make available the holistic procedure of laboratory work in an Online Platform. The project begins with a prototype of the Electronics and Communication Laboratory. This agenda has an immense application in today's world, especially in times when pandemics like COVID-19 have forced the world to search for alternative solutions. The web app provides a virtual platform to the students for practicing and understanding the laboratory experiments present in their coursework. The site comprises labs from multiple streams and subjects, to include students from different demographics to make use of this convenient resource. The web app will provide the students with all the theories related to the experiment along with requisite diagrams and graphs which will give them the feel of a real laboratory.*

Keywords- Virtual lab, PN diode, JavaScript, React.js

I. INTRODUCTION

Due to COVID-19 the colleges went on lockdown, so most of the classes were interrupted. Students had to attend classes in online mode, which at the beginning was a great idea to continue the studies but as time passed people realized that the online mode is harming the very essence of education– the laboratory practicals. The need of the hour was a robust and undemanding interface which would be easy to work with and accessible to all. Hence, the purpose of developing an E-lab web application.

The problem was to provide a practical electronics lab to the students who are home-stricken so that their practical studies don't hamper. So, we devised a method that could help thousands of students who are not able to attend classes during the lockdown. So, we proceeded methodologically by modeling the electronics lab in a virtual environment, where students can interact with the practicals just as in a real practical lab. So our main challenge was to design a system that would be user friendly and as good as a real world lab.

The app also enables the students to do the required calculations for a particular experiment and it is designed in such a way that it will revert back the outputs, after receiving the necessary inputs. The corresponding input and output values will get updated into the table as a new row. Also, in real time we will get the values plotted in the graph. JavaScript [1], [2], [3] has been utilized for the implementation of the virtual lab.

II. ELEMENTS OVERVIEW

The students are supposed to read and understand the objective of the particular experiment during the virtual lab. Initially, the required theory and diagrams of the respective experiments are provided. After this, there are options to input sample values of the parameters and obtain the corresponding result. The navigation bar displays all the experiments in a sequential order using a tabular design. If the student clicks on any experiment, then a personalized drop down menu, specifically designed for that individual experiment, will pop up. Further, clicking on any particular subsection of that menu will open the corresponding webpage in the right hand side of the web browser. Hence, overall generalized web application for the virtual lab consists of following elements:

2.1 Navigation Bar:

The design includes two navigation bars on the web app- one at top and another one at left side. The top one helps in navigating back to the homepage. For the link to the pages of experiments, link tags have been used instead of anchor tags, as it helps in avoiding redundant calls to the server every time the link is clicked. The left vertical bar helps in navigation through different experiments. Every experiment has a drop down menu to access different subparts of the experiment.

2.2 Experiment Tab:

Every experiment tab has three main parts- a diagram, theory, and graph. The diagram is at the top of the experiment, we have used img tag for the image and it is centered at the top of the tab. Just after the image we have the theory part, which is mostly static text with some equation related image. Here we display the equation based on which

we will later calculate the outputs. The lower part of the tab has input output sections with table and graph too.

2.3 Graph and Table:

This part is designed in such a way that when an input is given its corresponding output is displayed that gets calculated on the basis of the previously provided formula. Both the values get inserted into the table as a new row. Further they get plotted into the graph simultaneously. For displaying the output at the table as well as in the graph respective states are declared in the app.

III. THEORETICAL CONCEPT: SIMULATION OF IV CHARACTERISTIC OF PN DIODE

Initially, a simple laboratory experiment on Basic Electronics i.e. "Observation of IV Characteristics of PN Diode" has been selected for the web application development of Virtual Lab. Before going through the detailed approach of web application, let's get the preliminary concept of the experiment.

The relationship between the diode current and voltage is non-linear or exponential in nature. The current (I_d) of a PN junction diode can be given as [9].

$$I_d = I_s(e^{\frac{qV_d}{\eta KT}} - 1) \quad (1)$$

where I_s is the reverse saturation current of the diode, V_d is the diode voltage, η is the diode ideality factor, K is the Boltzmann constant (1.3806503×10^{-23} J/K), T is the temperature of the junction in Kelvin, and q is the electron charge (1.60217646×10^{-19} C).

It can be seen that the value of reverse saturation current and ideality factor are approximately $3.0\text{x}10^{-9}$A and 1.37 respectively at 297 K temperature for 1N4007 diode. These values can be further used for modeling of electronics systems made of 1N4007 PN diodes.

IV. FRAMEWORKS USED

Following technologies are used for the development of web app for virtual laboratory. For the whole application **React.js** [6], [7], [8] and **Chart.js** [4], [5] are used.

4.1 React.js

It has following advantages.

- The requirement was of a reactive web app where small changes corresponding to the user behavior can be observed without much work under the hood. Thereby, making the app lightweight and easy to use.
- React.js is an open-source JavaScript library that is used for building user interfaces specifically for single-page applications. It's used for handling the view layer for web apps.
- React allows developers to create large web applications that can change data, without reloading the page. The main purpose of React is to be fast, scalable, and simple. It works only on user interfaces in the application.

Important features of React are as follows

4.1.1. React Hooks: Tags in React have been used to take in the inputs, so that we can easily apply input values' validation in future. The app reads the input from the object emitted when the form value is submitted. React hooks are a feature which is available in react versions of 16.8 and above, Hooks help to use state and other react features without a class. These are functions which hook into React state and lifecycle features from function components and cannot be used inside a class. Informally, it helps to react and reflect on the changes after the project is rendered once.

- *UseState:* It is a hook which allows us to have state variables in functional components. UseState is implemented to maintain some local component states, since these are used to plot the graph in real time.
- UseEffect: The useEffect Hook allows to perform side effects in the components. Some examples of side effects are: fetching data, directly updating the DOM, and timers.

4.1.2 React Router: Routing is a process in which a user is directed to different pages based on their action or request. ReactJS Router is mainly used for developing Single Page Web Applications. React Router is used to define multiple routes in the application. When a user types a specific URL into the browser, and if this URL path matches any 'route' inside the router file, the user will be redirected to that particular route.

4.1.3 React Router DOM: React Router DOM is an npm package that enables us to implement dynamic routing in a web app. It allows us to display pages and allow users to navigate them. It is a fully-featured client and server-side routing library for React. React Router Dom is used to build single-page applications i.e. applications that have many pages

or components but the page is never refreshed instead the content is dynamically fetched based on the URL. This process is called Routing and it is made possible with the help of React Router Dom.

4.2.Chart.js

Chart.js is a popular open source library that helps us to plot data in web applications. It is highly customizable, and a powerful tool when it comes to expressing scientific facts in a graphical way. The components of the graph are implemented using a npm package called chart.js. Inside the code of the graph some important parameters are registered such as Linear Scale, Point Element, Line Element which are used internally in the webpack of React.js.

V. WORKING PRINCIPLE

During the implementation of virtual lab, following functions are involved.

1. Charts(): It is the react component that gets exported as the complete chart, which is later called in this web application.
2. Truncate Decimals(number, digits): It takes two inputs 'number' and 'digits'. It returns a truncated value of the 'number' to its 'digits' decimal place.
3. Calculate(input1, input2): It takes two inputs 'input1' and 'input2'. Here the input 'input1' takes the value of [Vf] and 'input2' is set to 0.000003 while calling the function to return the value of [If]. Within this function the value of [If] is computed using the Eqn. 1, where [Is], q and v are predefined in the function. React states used:-

    - vdin: Stores the current state of [Vd] taken from input.
    - idin: Stores the current state of [Id] which is then passed in the truncateDecimals() function.
    - xaxis: Stores the current state of the point to be plotted along x-axis.
    - yaxis2: Stores the current state of the point to be plotted along y-axis.
    - tabhtml: Stores the current state where the internal structure of the table to be plotted is defined.

Next, flowing objects are used for the implementation of the virtual lab

1. data: Within this object we define labels which are 'xaxis' and define another object within it called 'datasets' which stores basic properties of the plotted graph such as color of the plot, background color etc.
2. options*:* Within this object we set the parameter of scale by setting the y-axis beginning with zero false.

Ultimately, graph rendering features has been used to visualize the outputs. The code that renders the graph is present in a function called EnterHandler(). This function has several states which helps in memorizing previous data whenever we get a new value, these states are maintained using react hooks. The states that are kept here are the input voltage, the output current, the x-axis and y-axis points and also a state related to the table row.

Firstly, the input is taken using react form using its 'onChange' property which emits objects whenever there is a change in the input value. The last emitted object is used to get the value from its 'target' attribute. This value is passed to call the input react hook which changes the current input value and based on this current input the current output is calculated using the calculate() function and finally as the state changes, the output value is visible next to the input value.

Another hook is employed to update the table; the two values are passed to the hook which adds them as a new row in the table. At the same time, these values are passed to two more hooks which add them into x-axis and y-axis datasets respectively. After the data is inserted, the state changes and the new point appear in the graph. And the process goes on as we insert new values.

VI. RESULTS

This website https://jgec-e-lab-online.netlify.app/ provides a virtual platform to the students for practicing and understanding the laboratory experiments present in their coursework. The site will comprise of labs from multiple streams and various subjects, so that no niche of students feel excluded from this online phenomenon.

This navigation bar displays all the experiments in a sequential order using a tabular design. If the student clicks on any experiment, then a personalized drop down menu, specifically designed for that individual experiment, will pop up. Further, clicking on any particular subsection of that menu will open the corresponding webpage in the right hand side of the web browser.

Fig.1: Home page of the Website with navigation Bar

After clicking the particular experiment number, corresponding theory, equations, typical graphs and diagrams of the respective experiments are provided. Here, we are using forms in React

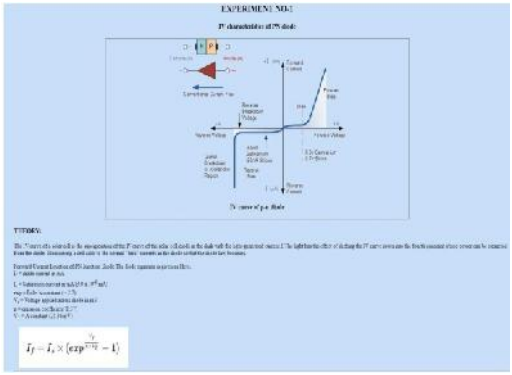

Fig. 2: Theory of the related experiment

After this, there are options to input sample values of the parameters (i.e. Vd) and obtain the corresponding result. For displaying the output, we declare states in our react app such that in this case the input is changed, the output will be changed simultaneously.

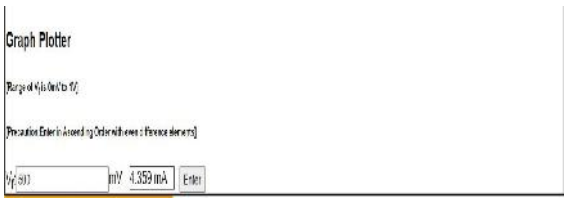

Fig. 3: Processing Results and Showing the Outputs

There is a button next to the input form, on clicking it we can insert the input as well as output values into the table. We are using a react hook to update the table; we pass the input value and the output value that we got after calculation to the hook which binds them as a token. The state changes, the table is rendered again with the new row present at the end of previously entered data.

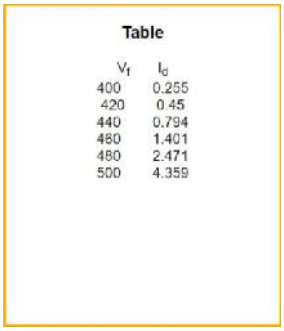

Fig. 4: Store the output in Tabular format

As the data is entered in the table we can see the respective plotting in the graph. We are using react hooks to show the plottings as the data set changes.

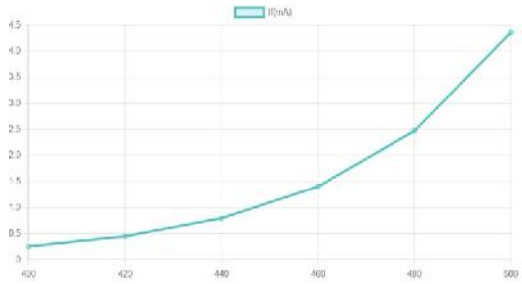

Fig. 5: Plot the Output as Graph

VII. CONCLUSION

The current model is indeed a small prototype of the greater idea. The project aims to bring down the hassle of completing University level Experimental Work with utmost precision in the comfort of homes or the hostel dorms and within fingertips. In future we can add a backend support to our existing application which can be used for user authentication and admin panel. In addition to that, we can also store the data submitted by the students in a central database.

REFERENCES

[1] Eloquent JavaScript, https://eloquentjavascript.net/Eloquent_JavaScript.pdf
[2] JavaScript, https://developer.mozilla.org/en-US/docs/Web/JavaScript
[3] Javascript: The Definitive Guide - Master the World's Most-Used Programming Language, Seventh Edition
[4] Chart.js, https://www.chartjs.org/docs/latest/
[5] Chart.js:Line Graphs, https://www.w3schools.com/js/tryit.asp?filename=tryai_chartjs_lines
[6] The Road to Learn React, by Robin Wieruch
[7] Learning React, by Alex Banks, Eve Porcello

[8] React documentation, https://reactjs.org/
[9] S. Mandal, S. Majumdar, S. Barman and S. Haldar, “Parametric Optimization of PN Junction Diode Using Flower Pollination Algorithm”, International Journal of Emerging Engineering Research and Technology, vol. 5, Issue 9, pp. 32-36, 2017.