BeiGene

Vue and webR Integration for Serverless Local Statistical Analysis in a Single HTML File

BeiGene

Kaiping Yang

2025.03.28

Catalogue

BeiGene

PART 1

Research Background and Abstract

Research Background

01

With the advancement of web technologies, there is a growing demand for **statistical analysis applications** that can be deployed **without a server**, running directly on the user's **local machine**.

02

Leveraging the capabilities of modern browsers' **JavaScript** and the powerful statistical features of the **R language** makes it possible to perform local statistical analysis.

Research Abstract

This study introduces how to build a pure client- side statistical analysis application in a single HTML file using **Vue** and **webR**.

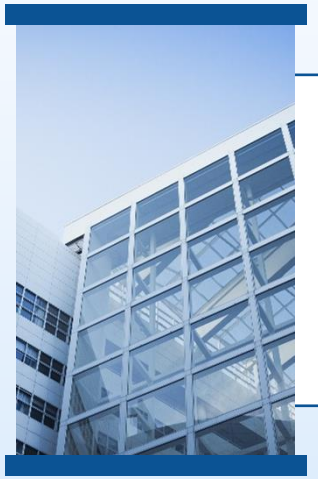
Vue is used for responsive front- end interactions, while webR calls the R language runtime environment, allowing users to experience the complete statistical analysis process by simply opening an HTML file.

BeiGene

PART 2

Introduction to webR

What is webR?



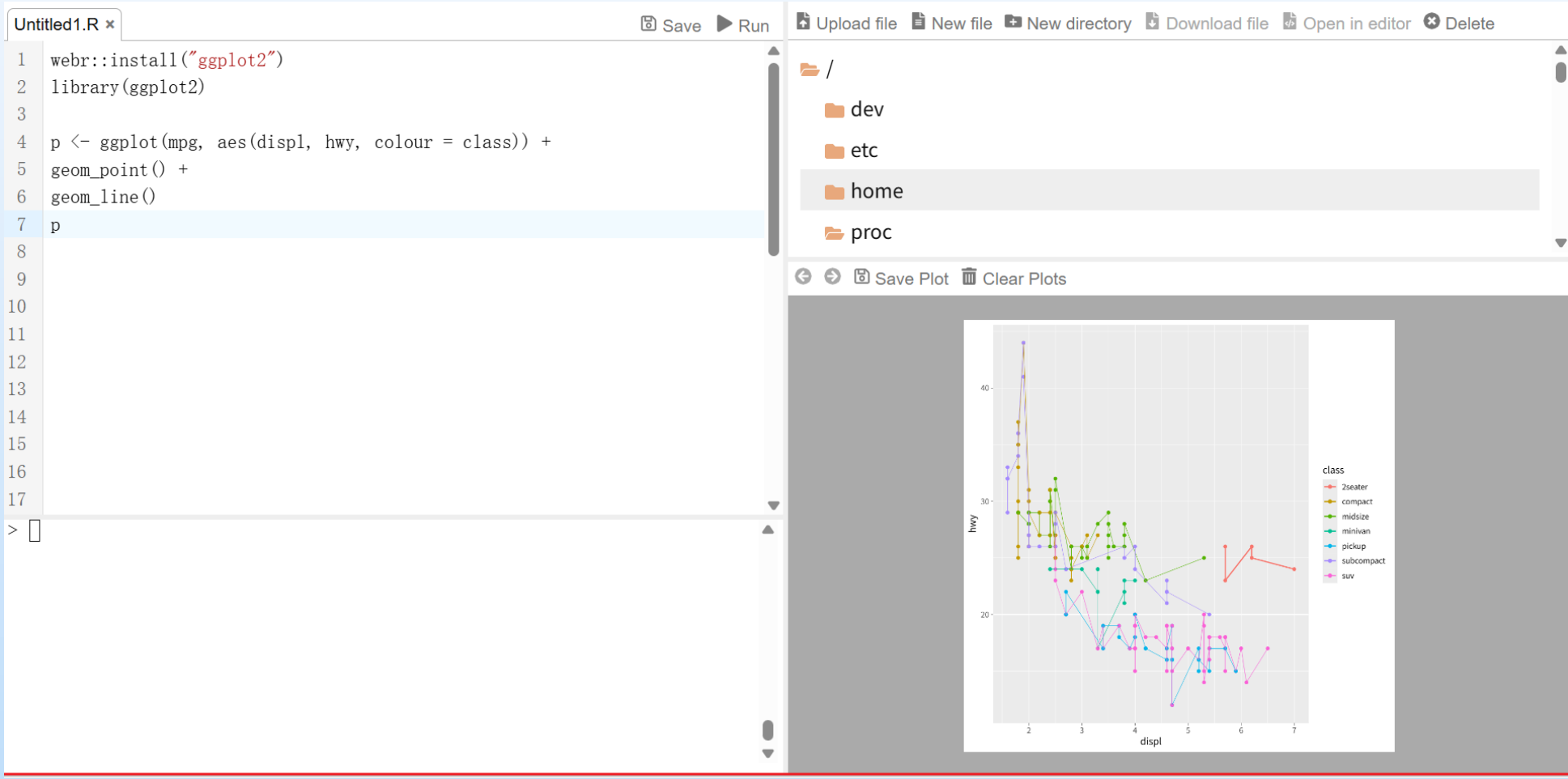
webR is an implementation of the R language on **WebAssembly** (WASM), enabling R to run in **browsers**.

- **Eliminates** the need for an R server
- Reducing deployment
- Reducing maintenance costs
- supports commonly used R packages such as ggplot2 and plotly

webR REPL

The webR REPL app provides a simple R environment directly in your web browser.
The app can be accessed at https://webr.r-wasm.org/v0.2.0/
and includes sections for

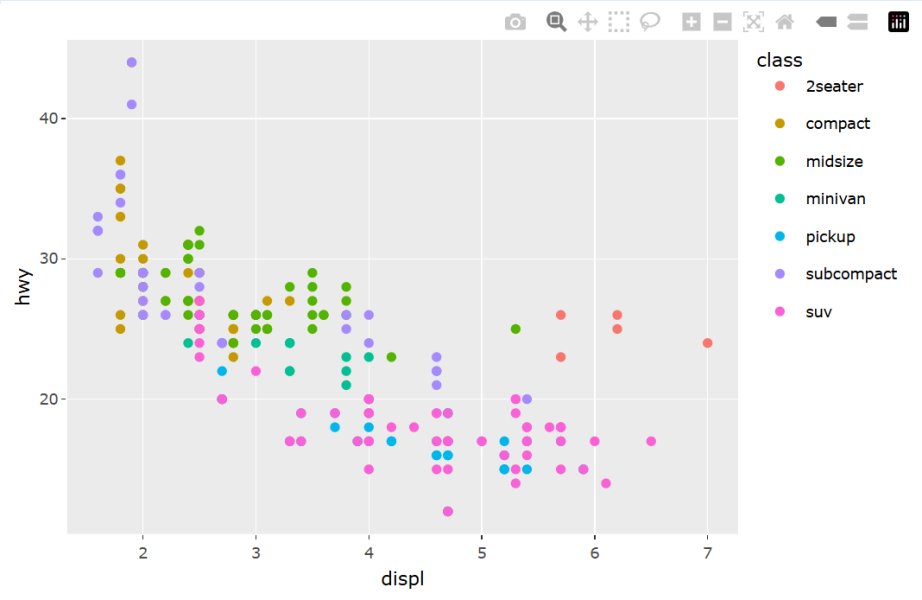
- R console input/output,
- code editing,
- file management,
- graphics device output.



Code Example

Interactive charts
with ggplot2 and plotly

Using the plotly R package, ggplot2 output can be converted into interactive figures powered by plotly.js.




```html
<html>
  <head>
    <title>Plotly Example</title>
    <script
      src="https://cdnjs.cloudflare.com/ajax/libs/plotly.js/2.26.2/plotly.min.js"
      charset="utf-8"
    ></script>
  </head>
  <body>
    <div>
      <pre><code id="out">Loading webR, please wait...</code></pre>
    </div>
    <script type="module">
      import { WebR } from 'https://webr.r-wasm.org/latest/webr.mjs';
      const webR = new WebR({ interactive: false });
      await webR.init();
      const outElem = document.getElementById('out');
      outElem.innerText = 'Loading plotly, please wait...';
      await webR.installPackages(['jsonlite', 'ggplot2', 'plotly'], true);
      outElem.innerText = 'Generating plot, please wait...';
      const plotlyData = await webR.evalRString(`
library(plotly)
library(ggplot2)
p <- ggplot(mpg, aes(displ, hwy, colour = class)) +
  geom_point()
plotly_json(p, pretty = FALSE)
`);
      outElem.replaceChildren();
      Plotly.newPlot('out', JSON.parse(plotlyData), {});
    </script>
  </body>
</html>
```

init

package

eval R code

BeiGene

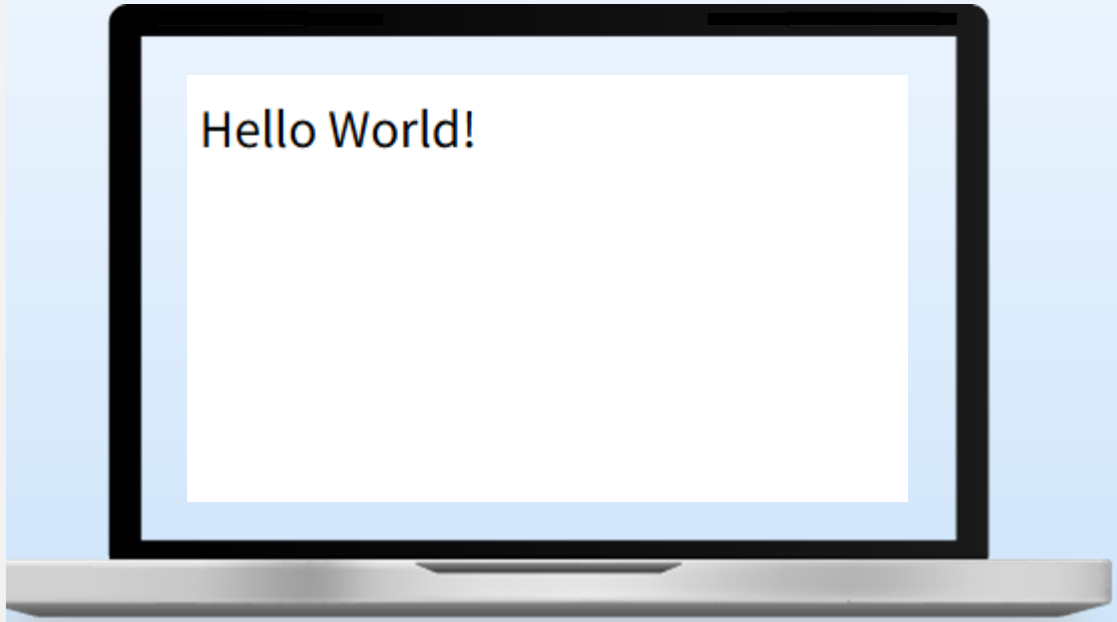
PART 3

Vue and elementUI

Introduction of Vue and elementUI

Vue Hello world

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Vue 2 Hello World</title>
    <!-- Include Vue 2 from a CDN -->
    <script src="https://cdn.jsdelivr.net/npm/vue@2.7.9"></script>
</head>
<body>
    <!-- The div where the Vue instance will be mounted -->
    <div id="app">
        {{ message }}
    </div>

    <script>
        // Create a new Vue instance
        var app = new Vue({
            // Mount the Vue instance to the element with id 'app'
            el: '#app',
            // Define the data properties for the Vue instance
            data: {
                // The message to display
                message: 'Hello World!'
            }
        });
    </script>
</body>
</html>
```



Vue + elementUI Hello world


```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Element UI Hello World</title>
    <!-- Include Vue 2 from a CDN -->
    <script src="https://cdn.jsdelivr.net/npm/vue@2.7.9"></script>
    <!-- Include Element UI CSS -->
    <link rel="stylesheet" href="https://unpkg.com/element-ui/lib/theme-chalk/index.css">
    <!-- Include Element UI JS -->
    <script src="https://unpkg.com/element-ui/lib/index.js"></script>
</head>
<body>
    <!-- The div where the Vue instance will be mounted -->
    <div id="app">
        <!-- Use Element UI's Button component to display "Hello World" -->
        <el-button type="primary" @click="sayHello">Hello World</el-button>
    </div>
    <script>
        // Create a new Vue instance
        new Vue({
            // Mount the Vue instance to the element with id 'app'
            el: '#app',
            // Define methods for the Vue instance
            methods: {
                // Method to alert "Hello World"
                sayHello() {
                    alert('Hello World!');
                }
            }
        });
    </script>
</body>
</html>
```

BeiGene

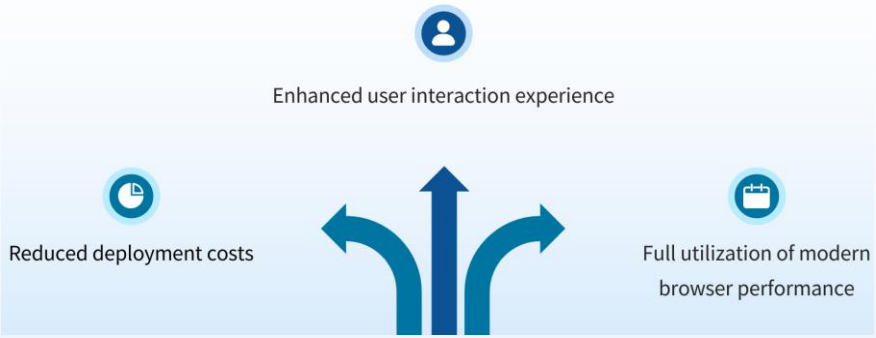
PART 4

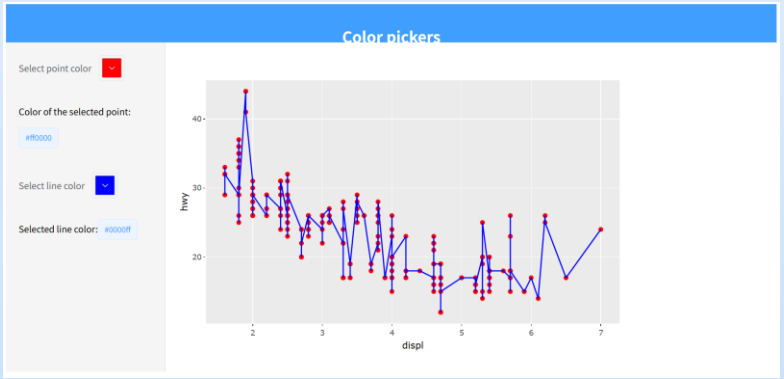
Demonstration

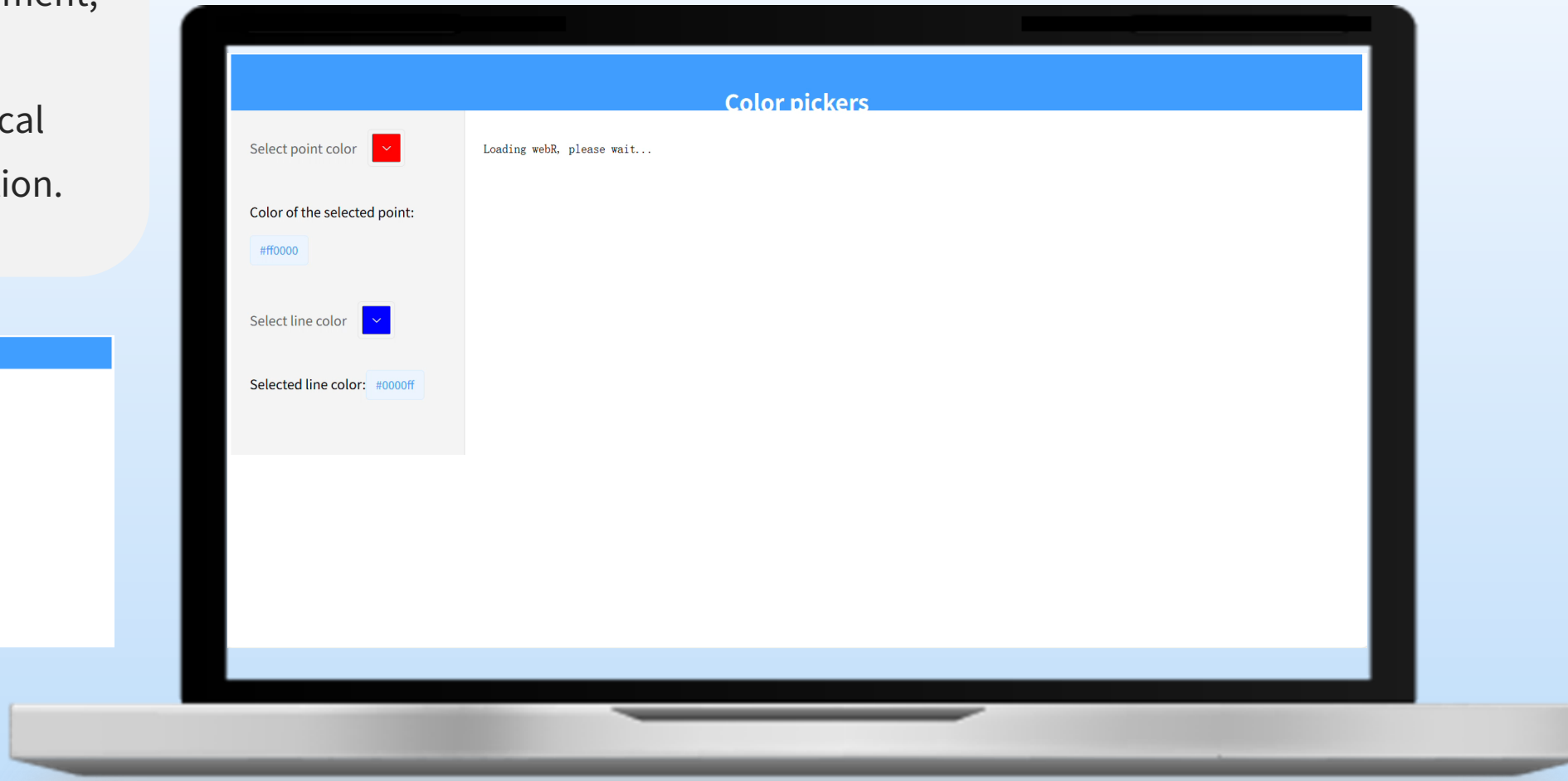
Integration Objectives

Utilize **Vue** for front- end interactions and state management,

and **webR** for backend statistical calculations and chart generation.







Code Structure Explanation

01

HTML PartUses Element UI to build page layout, including header, sidebar (color picker), and main content area (chart display).

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Element UI Example</title>
    <!-- Include Vue 2 from a CDN -->
    <script src="https://cdn.jsdelivr.net/npm/vue@2.7.9"></script>
    <!-- Include Element UI CSS -->
    <link rel="stylesheet" href="https://unpkg.com/element-ui/lib/theme-chalk/index.css">
    <!-- Include Element UI JS -->
    <script src="https://unpkg.com/element-ui/lib/index.js"></script>
    <!-- Include Plotly -->
    <script src="https://cdnjs.cloudflare.com/ajax/libs/plotly.js/2.26.2/plotly.min.js" charset="utf-8"></script>
    <style>
        .el-header {
            background-color: #409EFF;
            color: white;
            text-align: center;
            line-height: 60px;
        }
        .el-aside {
            background-color: #f5f5f5;
            border-right: 1px solid #dcdcdc;
            padding: 20px;
        }
        .el-main {
            padding: 20px;
        }
    </style>
</head>
```

Code Structure Explanation

01

HTML PartUses Element UI to build page layout, including header, sidebar (color picker), and main content area (chart display).

```html
<body>
    <div id="app">
        <el-container>
            <el-header>
                <h2>Color pickers</h2>
            </el-header>
            <el-container>
                <el-aside width="250px">
                    <el-form>
                        <el-form-item label="Select point color">
                            <el-color-picker v-model="pointColor"></el-color-picker>
                            <p>Color of the selected point:<el-tag>{{ pointColor }}</el-tag></p>
                        </el-form-item>
                        <el-form-item label="Select line color">
                            <el-color-picker v-model="lineColor"></el-color-picker>
                            <p>Selected line color:<el-tag>{{ lineColor }}</el-tag></p>
                        </el-form-item>
                    </el-form>
                </el-aside>
                <el-main>
                    <div>
                        <pre><code id="out">Loading webR, please wait...</code></pre>
                    </div>
                </el-main>
            </el-container>
        </el-container>
    </div>
```

Code Structure Explanation

02

Vue PartInitializes Vue instance, sets reactive data pointColor and lineColor, watches color changes, and triggers chart updates.

```
<script type="module">
    new Vue({
        el: '#app',
        data() {
            return {
                pointColor: '#ff0000',
                lineColor: '#0000ff'
            };
        },
        watch: {
            pointColor(newColor) {
                this.updatePlot();
            },
            lineColor(newColor) {
                this.updatePlot();
            }
        },
        mounted() {
            this.initWebR();
        },
        methods: {
```

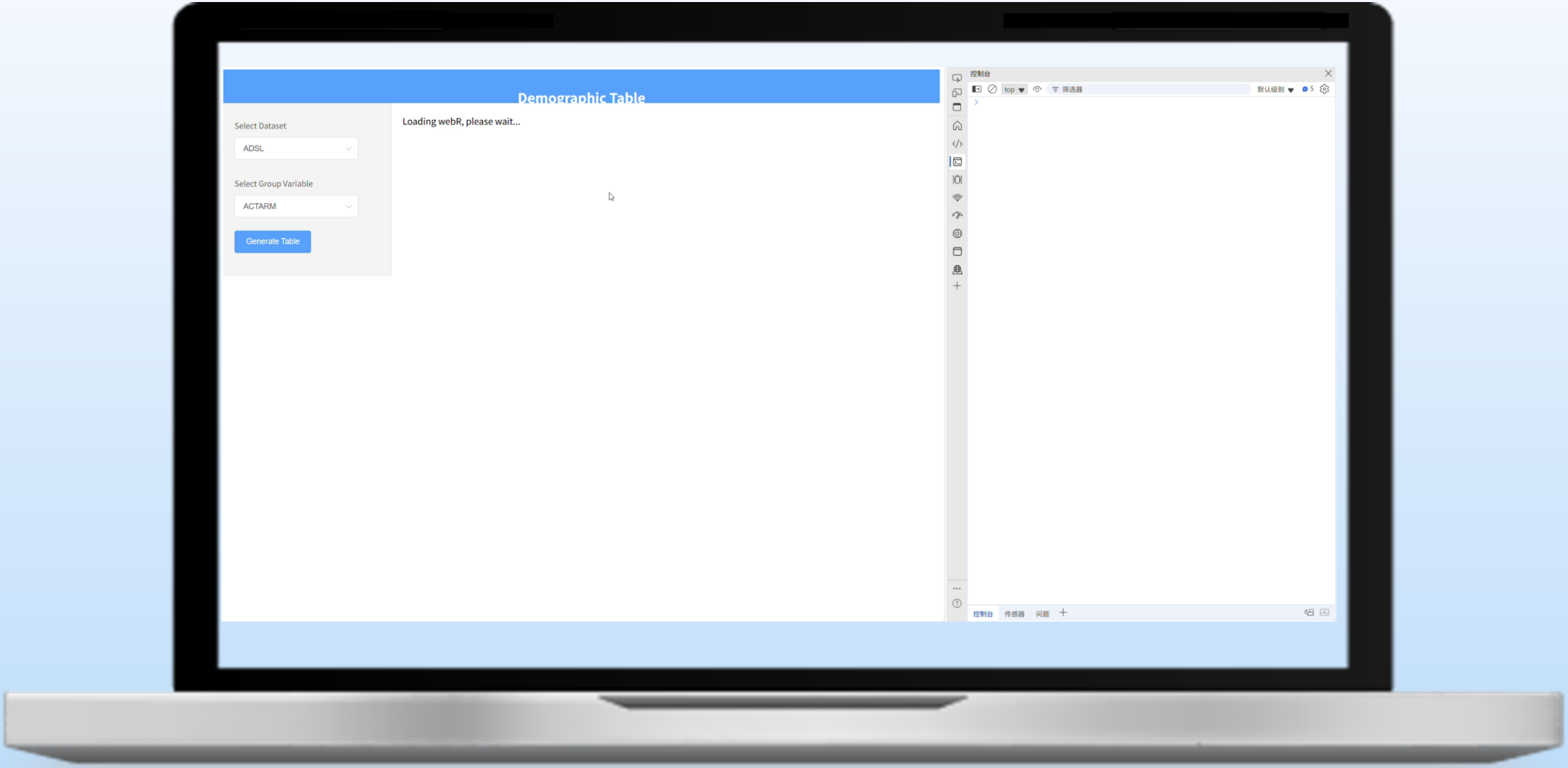
Code Structure Explanation

02

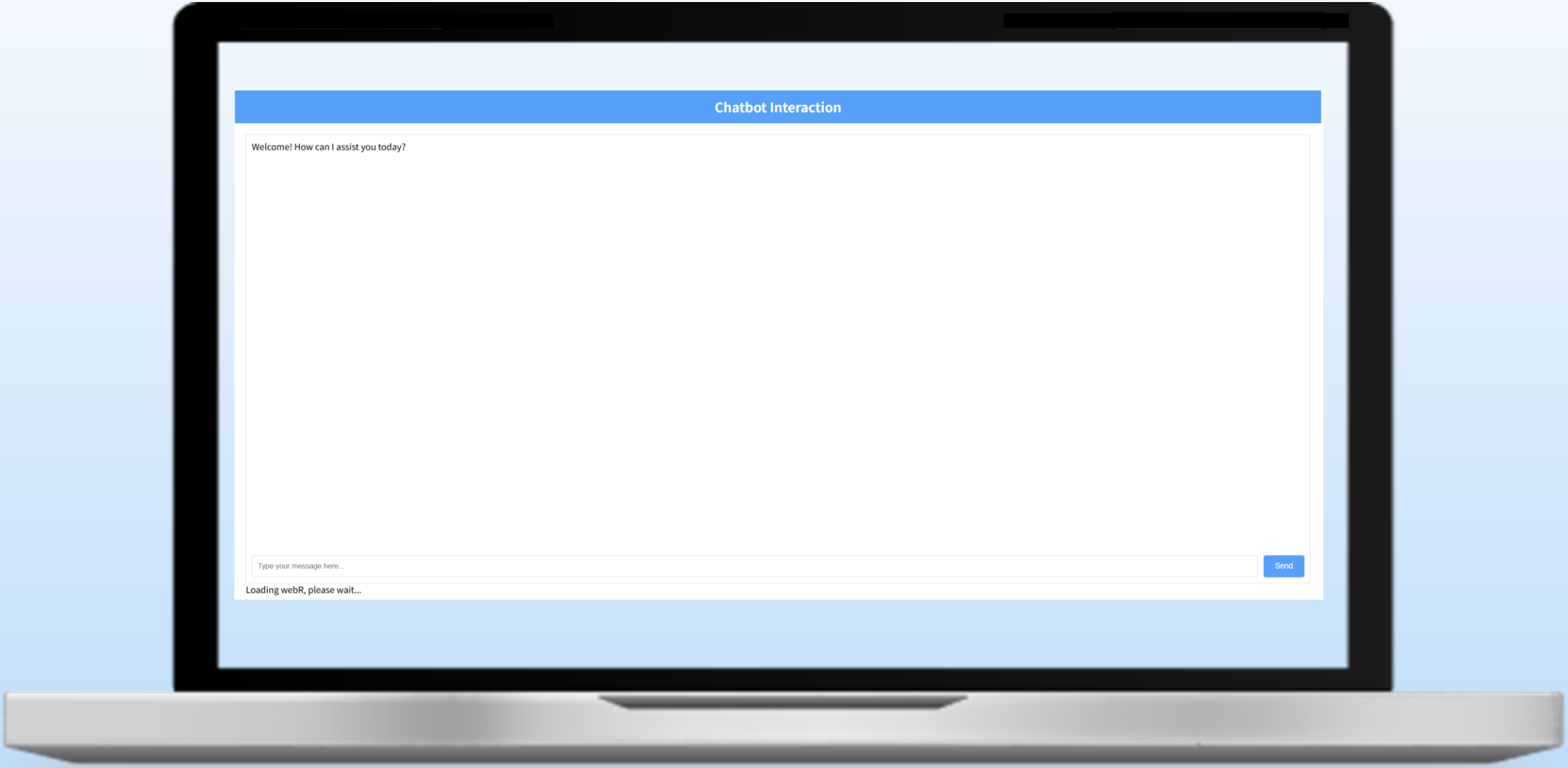
webR PartInitializes webR in Vue's mounted hook, loads R packages, and generates charts.

```
            methods: {
                async initWebR() {
                    const { WebR } = await import('https://webr.r-wasm.org/latest/webr.mjs');
                    this.webR = new WebR({ interactive: false });
                    await this.webR.init();
                    const outElem = document.getElementById('out');
                    outElem.innerText = 'Loading plotly, please wait...';
                    await this.webR.installPackages(['jsonlite', 'ggplot2', 'plotly'], true);
                    outElem.innerText = 'Generating plot, please wait...';
                    this.updatePlot();
                },
                async updatePlot() {
                    const rcode = `
                        library(plotly)
                        library(ggplot2)

                        p <- ggplot(mpg, aes(displ, hwy, colour = class)) +
                        geom_point(colour = '${this.pointColor}') +
                        geom_line(colour = '${this.lineColor}')

                        plotly_json(p, pretty = FALSE)
                    `;
                    const plotlyData = await this.webR.evalRString(rcode);
                    const outElem = document.getElementById('out');
                    outElem.replaceChildren();
                    Plotly.newPlot('out', JSON.parse(plotlyData), {});
                }
            }
        });
    </script>
</body>
</html>
```

Demographic Table



AI Chatbot Interaction



BeiGene

PART 5

Conclusion

Conclusion

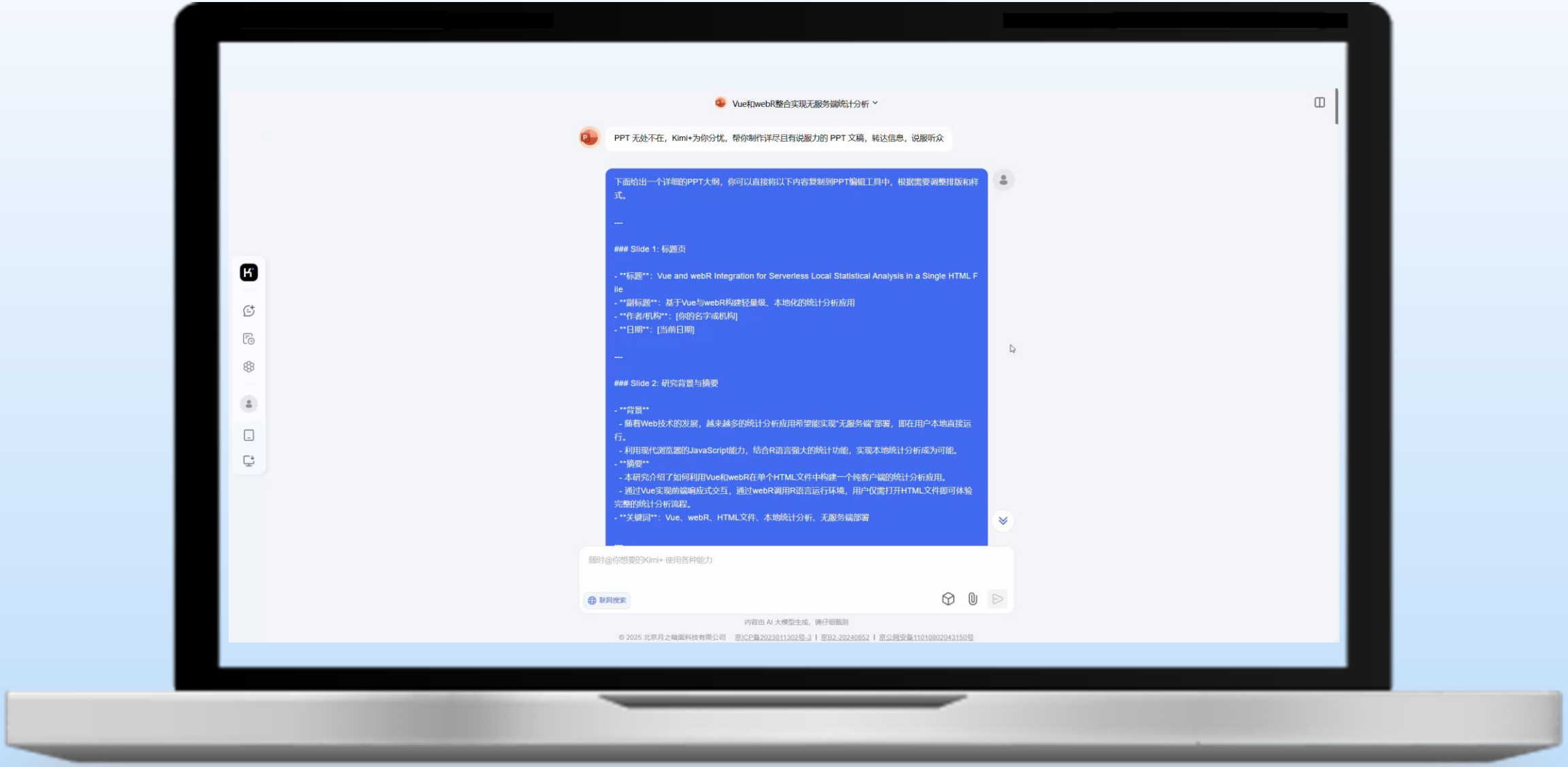
01

By integrating Vue and webR, we have achieved a **single- file**, localized solution for statistical analysis.

02

This solution leverages modern web technologies to provide powerful statistical analysis and data visualization **without server support**.

Colored egg: Kimi slides



BeiGene

Thank you for listening!

Email: kaiping.yang@beigene.com