FIT5032 Assessed Lab 10
External API Services & Local API Development

Student Name: [Du Daoan]
Student ID: [35523166]

Contents

1 EFOLIO TASK 10.1 (PASS AND CREDIT LEVEL)

1.1 Screenshot Set 1: Current Location Weather

1.1.1 Weather Application Code Implementation

```
src > views > WeatherView.vue > {} script
<template>
  <div class="container">
    <main>
      <div v-else-if="error" class="alert alert-danger" role="alert">
      </div>

      <!-- Initial state -->
      <div v-else class="initial-state">
        <p>🌤️ Enter a city name to get weather information</p>
        <p>Or allow location access to get your current weather</p>
      </div>
    </main>
  </div>
</template>

<script>
// The info section in 10.1.1 provided detailed information about this package
import axios from "axios";

// IMPORTANT: Replace this with your actual OpenWeatherMap API key
// Get your free API key from: https://openweathermap.org/api
// For demonstration purposes, we'll use a demo key that works with limited functionality
const apikey = "23feddf446106aac31f73c0663457261"; // Demo API key - replace with your own
  ⌘L to chat, ⌘K to generate
export default {
  name: "WeatherView",
  data() {
    return {
      city: "",
      weatherData: null,
      hourlyForecast: [],
      dailyForecast: [],
      loading: false,
      error: null,
    };
  },
  //computed is a property that is used to define a property that
  //is dependent on other data properties.
  //If we using a more easy to understand words to understand the concept:
  //the derived value such as temperatu                    update when the relevant value change.
  computed: {                                  Review next file >
    //There are multiple way to obtain the data in Celsius format.
```

Figure 1: WeatherView.vue - Weather API implementation code showing axios integration and geolocation

1.1.2 Browser Display - Current Location Weather

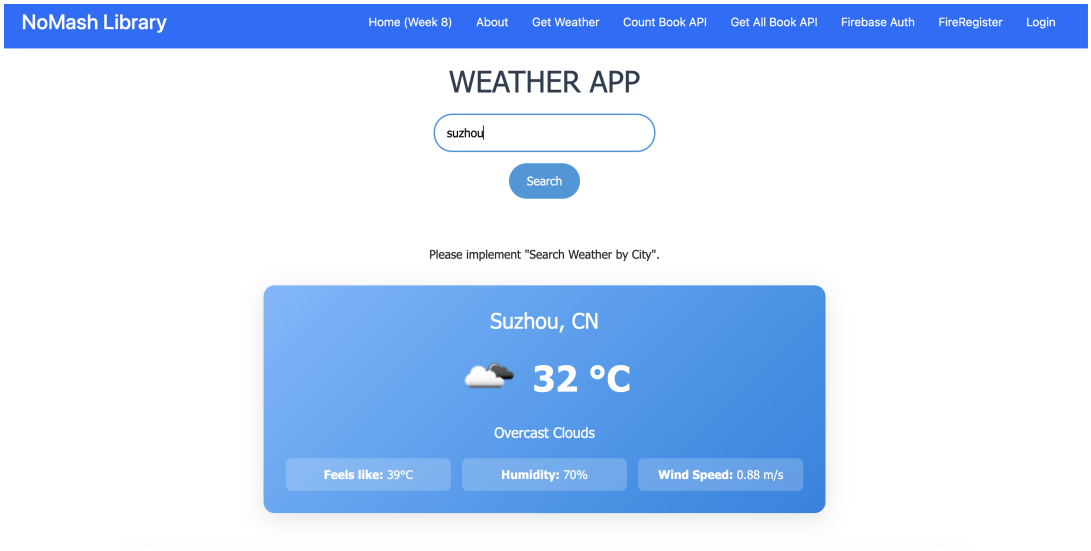

Figure 2: Browser view showing current location weather with temperature in Celsius and weather icon

1.2 Screenshot Set 2: API Page - Authors and Books Count

1.2.1 CountBookAPI Code Implementation

src > views > CountBookAPI.vue > {} script setup > calculateStats > authors.value.reduce() callback

```
103  <script setup>
109  const apiResponse = ref(null)
110
111  const authorsCount = ref(0)
112  const totalBooks = ref(0)
113
114  const calculateStats = () => {
115    authorsCount.value = authors.value.length
116    totalBooks.value = authors.value.reduce((total, author) => {
117      return total + author.famousWorks.length
118    }, 0)
119  }
120
121  const getApiData = async () => {
122    loading.value = true
123    error.value = null
124
125    try {
126      // Import the JSON data using ES6 import
127      const authorsModule = await import('@/assets/json/authors.json')
128      const data = authorsModule.default
129
130      authors.value = data
131      calculateStats()
132
133      // Create API response object
134      apiResponse.value = {
135        success: true,
136        data: {
137          authorsCount: authorsCount.value,
138          totalBooks: totalBooks.value,
139          authors: authors.value.map(author => ({
140            name: author.name,
141            bookCount: author.famousWorks.length,
142            birthYear: author.birthYear,
143            genres: author.genres
144          }))
145        },
146        metadata: {
147          timestamp: new Date().toISOStrin
148          version: "1.0.0",
```

Review next file >

Figure 3: CountBookAPI.vue - Local API implementation showing JSON data processing

1.2.2 Browser Display - Authors and Books Statistics

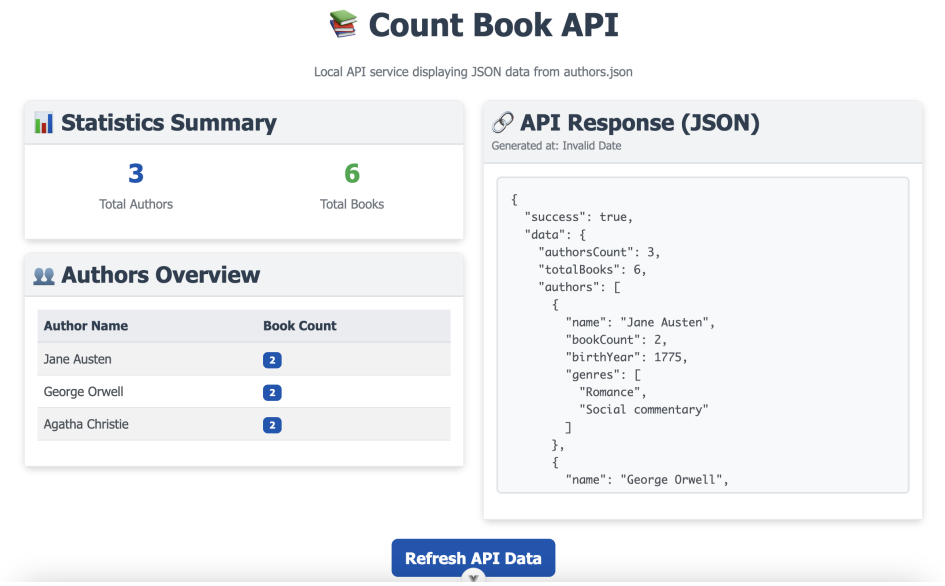

Figure 4: Browser view showing number of authors (3) and total books (6) from authors.json

2 EFOLIO TASK 10.2 (DISTINCTION AND HIGH DISTINCTION LEVEL)

2.1 Screenshot Set 1: Weather Search by City

2.1.1 Weather Search Code Implementation

```
src > views > WeatherView.vue > {} script > default > methods > fetchCurrentLocationWeather > navigator.geolocat
78   <script>
87   export default {
124    methods: {
127      async fetchCurrentLocationWeather() {
139          },
140          (error) => {
141            console.log("Geolocation error:", error);
142            this.loading = false;
143            this.error = "Unable to get your location. Please search for a city manually.";
144          }
145        );
146      } else {
147        this.error = "Geolocation is not supported by this browser.";
148      }
149    },
150
151    async searchByCity() {
152      if (!this.city.trim()) {
153        this.error = "Please enter a city name";
154        return;
155      }
156
157      this.loading = true;
158      this.error = null;
159
160      // API URL for searching by city name with metric units for Celsius temperature
161      const url = `https://api.openweathermap.org/data/2.5/weather?q=${this.city}&appid=${apikey}&units=
162      await this.fetchWeatherData(url);
163    },
164
165    async fetchWeatherData(url) {
166      try {
167        console.log("🌤 Making API call to:", url.replace(apikey, 'API_KEY_HIDDEN'));
168        const response = await axios.get(url);
169        //Returned data from API is stored as JSON file in weatherData
170        this.weatherData = response.data;
171        console.log("🌤 Weather data received:", this.weatherData);
172        this.error = null; // Clear any previous errors
173      } catch (error) {
174        console.error("❌ Error fetching                Review next file >          error);
175        if (error.response) {
```

Figure 5: WeatherView.vue - searchByCity function implementation with API call

2.1.2 Browser Display - Clayton Weather Search

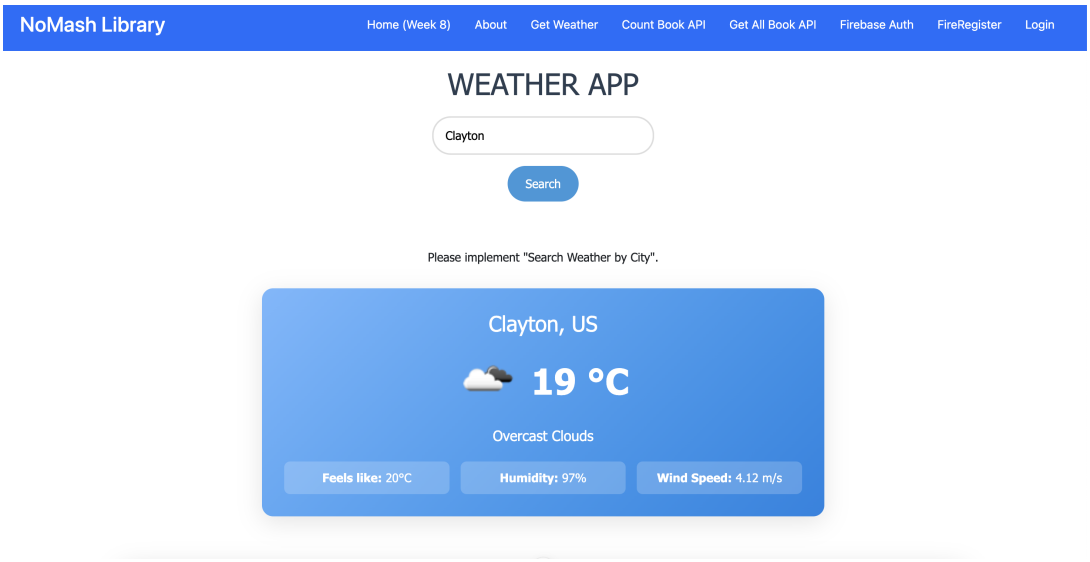

Figure 6: Browser showing Clayton, AU weather search with temperature in Celsius and weather icon

2.2 Screenshot Set 2: GetAllBookAPI Implementation

2.2.1 GetAllBookAPI Code Implementation

src > views > GetAllBookAPI.vue > {} script setup > fetchAllBooks

```vue
<template>
  <div class="container mt-4">
    <div class="text-center mt-4">
      </button>
    </div>
  </div>
</template>

<script setup>
import { ref, onMounted } from 'vue'

const allBooksData = ref(null)
const loading = ref(false)
const error = ref(null)

const fetchAllBooks = async () => {
  loading.value = true
  error.value = null

  try {
    // Import the JSON data using ES6 import
    const authorsModule = await import('@/assets/json/authors.json')
    const authorsData = authorsModule.default

    // Extract all books from all authors
    const allBooks = []
    authorsData.forEach(author => {
      author.famousWorks.forEach(book => {
        allBooks.push({
          id: `${author.id}-${book.title.replace(/\s+/g, '-').toLowerCase()}`,
          title: book.title,
          year: book.year,
          author: {
            name: author.name,
            birthYear: author.birthYear,
            genres: author.genres
          }
        })
      })
    })

    // Create comprehensive API response
```

Review next file >

Figure 7: Implementation code for GetAllBookAPI showing JSON data retrieval and formatting

2.2.2 Browser Display - All Books JSON Format

Figure 8: Browser view showing all books data in JSON format on GetAllBookAPI page

3 Key Implementation Features

3.1 External API Integration

- OpenWeatherMap API integration with personal API key
- Geolocation-based current weather fetching
- City-based weather search functionality
- Temperature display in Celsius with weather icons
- Error handling for API failures and invalid cities

3.2 Local API Development

- JSON data processing from authors.json file
- Statistics calculation (authors count, total books)
- API response formatting with metadata
- Real-time data refresh functionality
- Responsive UI with Bootstrap styling

3.3 Technical Implementation

- Vue 3 Composition API usage
- Axios for HTTP requests
- ES6 import for local JSON data
- Router configuration for SPA navigation
- Component-based architecture