To develop a web-based app that takes user input and generates a downloadable PDF file, you'll need a combination of front-end and back-end technologies. Here's a step-by-step guide along with the technologies you might consider:

1. **Front-End Development:**
   * **HTML/CSS:** For creating the user interface and styling.
   * **JavaScript:** For handling user interactions and making asynchronous requests to the server.
2. **Back-End Development:**
   * **Server-Side Language:** Choose a server-side programming language. Common choices include:
     + **Node.js (JavaScript):** Using frameworks like Express.js.
     + **Python:** Using frameworks like Flask or Django.
     + **Ruby:** Using Ruby on Rails.
     + **Java:** Using Spring or another framework.
     + **PHP:** Using frameworks like Laravel.
3. **User Input Handling:**
   * Use HTML forms to collect user input.
   * Validate user input on the client side (JavaScript) for responsiveness and on the server side for security.
4. **PDF Generation:**
   * Use a library to generate PDF files dynamically. Some popular options include:
     + **jsPDF (JavaScript):** A client-side library that allows you to generate PDF files.
     + **PDFKit (Node.js):** A server-side PDF generation library for Node.js.
     + **ReportLab (Python):** A Python library for creating PDF files.
5. **Communication Between Front-End and Back-End:**
   * Use AJAX or fetch API to send user input to the server asynchronously.
   * Set up routes or endpoints on the server to handle these requests.
6. **Server-Side Logic:**
   * Process user input, generate the PDF file using the chosen library, and save it on the server temporarily.
   * Return a response to the client with a link or the generated PDF file.
7. **File Storage (Optional):**
   * If you want to store generated PDFs, you may need a file storage solution (local or cloud-based) and handle file uploads and downloads.
8. **Deployment:**
   * Choose a hosting platform for your web application. Common choices include AWS, Heroku, Netlify, or Vercel.
   * Configure your server and deploy your application.
9. **Security Considerations:**
   * Implement proper input validation to prevent malicious input (XSS, SQL injection, etc.).
   * Secure your server and connections, especially if handling sensitive information.
10. **Testing:**

* Test your application thoroughly to ensure that it works correctly on various browsers and devices.
* Implement unit testing and integration testing as needed.

1. **Documentation:**

* Document your code and create user documentation if necessary.

Remember that this is a general guide, and the specific technologies and libraries you choose may depend on your preferences and project requirements. Always stay updated with the latest best practices in web development and security.