To deploy an app with a React front end and Flask back end on a LAN (Local Area Network) server, you'll need to follow these general steps:

1. Set up your LAN server: Prepare a computer or server within your LAN environment to host your application. This server should have the necessary infrastructure, such as an operating system (e.g., Linux), a web server (e.g., Nginx or Apache), and Python installed.

2. Build your React front end: Create your React application by developing the front-end code using tools like Create React App. Write your components, styles, and any necessary logic for the user interface.

3. Build your Flask back end: Develop your Flask application to handle the server-side functionality and API endpoints. Implement the necessary routes, database interactions, and any other back-end logic required for your application.

4. Bundle and optimize your React app: Use a bundler like webpack to package your React front-end code into a production-ready bundle. Optimize the bundle for performance by minifying and compressing the files.

5. Set up a web server: Configure your LAN server's web server (e.g., Nginx or Apache) to serve your React front-end files. This involves setting up the server to listen on a specific port and serve the bundled static files generated in the previous step.

6. Configure your Flask app: Set up your Flask application to run on the LAN server. Ensure that it listens on a specific port and handles the necessary routes and API endpoints. Consider using a WSGI server like Gunicorn to run your Flask app.

7. Connect the front end and back end: Ensure that your React app can communicate with the Flask back end by making API requests to the appropriate endpoints. Use the appropriate libraries (e.g., Axios) to handle these requests.

8. Test your application: Verify that your application works correctly by testing it on your LAN. Access the LAN server's IP address or hostname from other devices within the network to check if the app is accessible and functional.

9. Secure your application: Depending on your requirements, implement security measures such as authentication and authorization to protect your application and its data.

10. Monitor and maintain: Set up monitoring and logging mechanisms to keep track of your application's performance and troubleshoot any issues that may arise. Regularly update and maintain both your front-end and back-end code to ensure security patches and bug fixes are applied.

Remember that this is a high-level overview, and the specific implementation details may vary based on your server configuration, network setup, and application requirements.