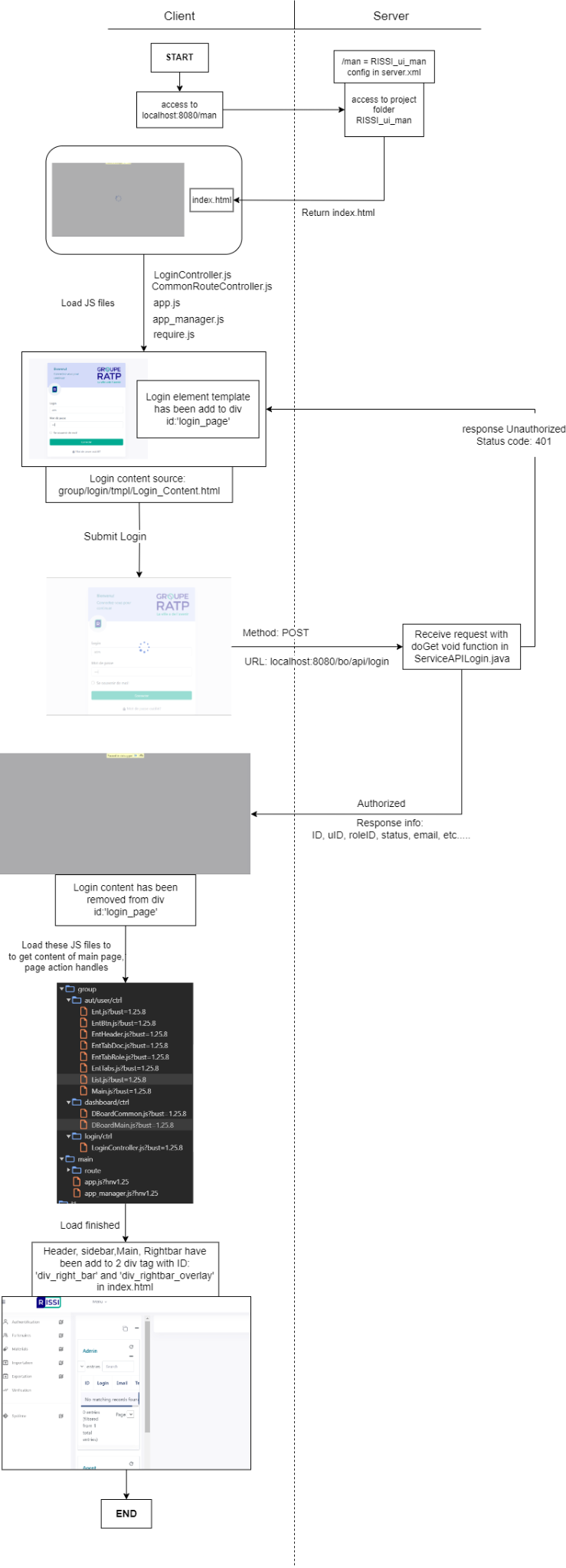
**Tổng hợp kết quả nghiên cứu**

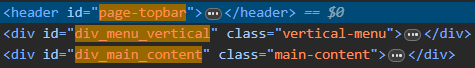
* **Thứ tự các mục:**

1. Đăng nhập.
2. Sơ lượt luồng chính sau khi đăng nhập.
3. Luồng hiển thị dữ liệu ra table.
4. SVNew.
5. SVDel.
6. Đính kèm file.
7. Lock thực hiện sửa đổi dữ liệu
8. **Đăng nhập:**

****

* **Client:**
  + - **Giao diện đăng nhập:**
* Truy cập localhost:8080/man sẽ thực hiện load file app.js, require.js, app\_manager.js, CommonRouteController.js, LoginController.js.
* app.js và app\_manager.js để thực hiện các hành động như khai báo các biến chứa đường dẫn đến folder được sử dụng trong cả project, handle hành động từ người dùng tương tác với trang web, handle data từ api, init view cho page, khai báo biến App và AppCommon, các function kiểm tra kết nối, reload trang...vv
* require.js thư viện hỗ trợ thuật toán và các function khác.
* CommonRouteController.js để khai báo các key api hay dùng, xử lý history, xử lý controller.
* LoginController.js thực hiện kiểm tra user đã đăng nhập chưa từ dữ liệu được lưu trong localstorage(bộ nhớ local phía client), thực hiện đăng nhập, đăng ký và đăng xuất cho user, file này được define và sử dụng thư viện jquery, import nội dung của phần tử đăng nhập từ file Login\_Content.html và nội dung của phần tử đăng ký từ file Register\_Content.html.
* **Giao diện Dashboard(home):**
  + Truy cập localhost:8080/man sẽ thực hiện load file app.js, require.js, app\_manager.js, LoginController.js, CommonRouteController.js, DBoardCommon.js, DBoardMain.js
* DBoardMain.js để init view

DBoard\_Main.html, DBoard\_Rightbar.html

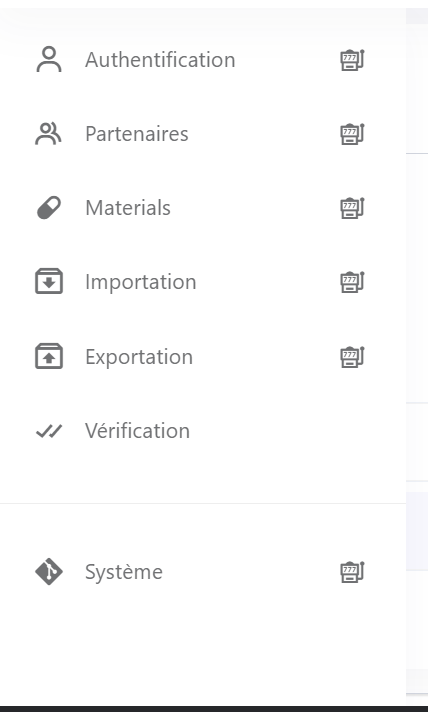


Hình 2.1: DBoard\_Main để chứa topbar, vertical menu và main content

sử dụng thư viện handlebars để compile, init dữ liệu cho giao diện

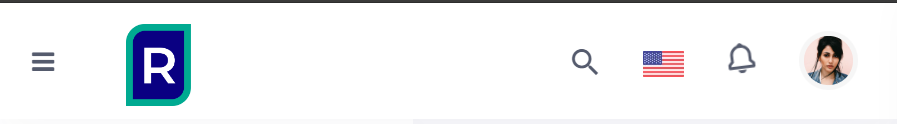
* DBoardCommon để init view

DBoard\_Sidebar.html



Hình 2.2: Side bar nằm trong thẻ div có id là div\_menu\_vertical, trong layout DBoard\_Main

DBoard\_Header.html



Hình 2.3: Header nằm trong thẻ div có id là page-topbar, trong layout DBoard\_Main

DBoard\_Notification.html

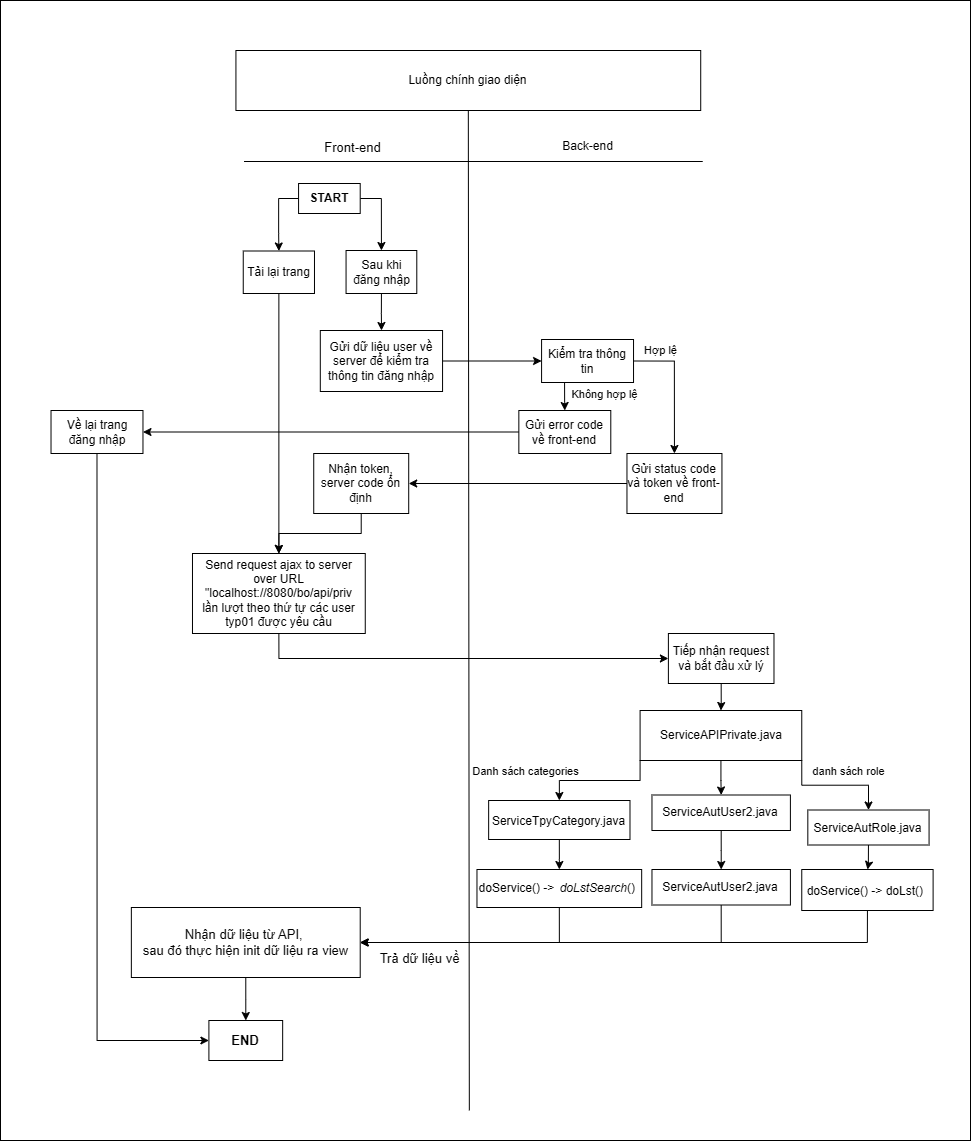
sử dụng thư viện handlebars để compile, init dữ liệu cho giao diện

* **Server:**
* Xác thực dữ liệu user với file ServiceAPILogin.java, với request mapping là /api/login, khi request tới ServiceAPILogin.java thì những yêu cầu có phương thức POST sẽ được sử lý bởi hàm doGet, encode bycrypt, convert information trong request dạng JSON, lấy từng thông tin ra và xác thực và chuyển đổi dữ liệu sang json và gửi về client.

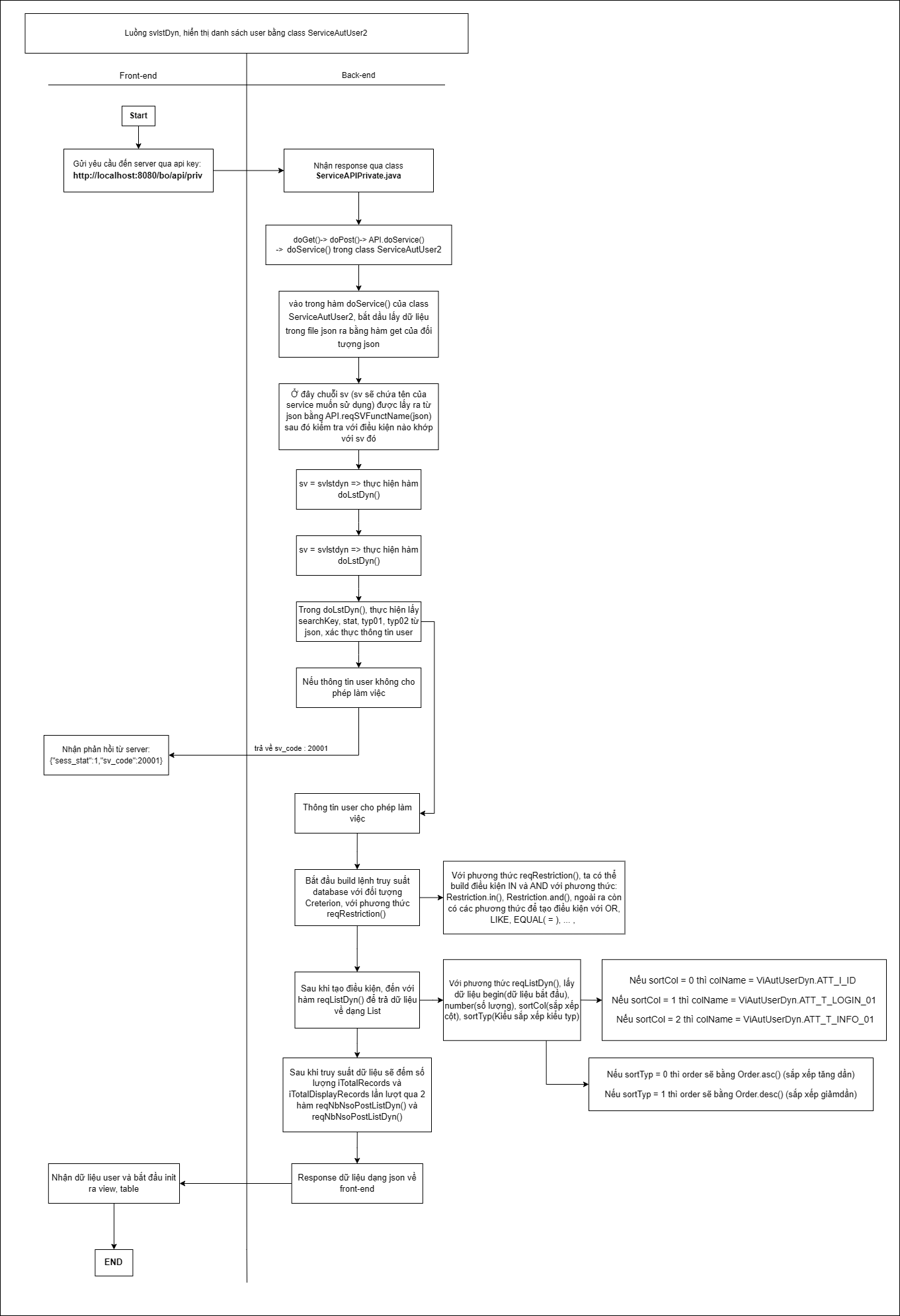
1. **Phân tích quy trình hoạt động:**

* Sau khi truy cập localhost:8080/man thì các file js check trạng thái đăng nhập của user từ localstorage thì bắt đầu init view, event handle function thì người dùng bắt đầu nhập thông tin đăng nhập(username và password)
* Sau khi nhập xong nhấn enter hoặc truy cập bằng nút connect trên giao diện thì sẽ bắt đầu gửi lấy dữ liệu từ các trường input, và gửi dữ liệu đến api để xác thực, quá trình này được handle bởi function do\_lc\_Login() trong file LoginController.js, function này sẽ dùng đến đối tượng App, contribute network và sử dụng do\_lc\_ajax\_bg để thực hiện gửi dữ liệu về api bằng key localhost:8080/bo/api/login..
* Sau khi dữ liệu được gửi về api thì file có tên ServiceAPILogin.java (file này được request mapping xử lý với key /api/login), sau đó bắt đầu xử lý dữ liệu với hàm doGet, trong hàm này, kiểm tra request trong ip có bị Locked hay không bằng hàm canIPLocked, trả về kiểu dữ liệu boolean
* Sau đó sử dụng đến thư viện Json để thực hiện chuyển đổi dữ liệu trong request sang json để tách thông tin.
* Sau khi tách thông tin từ request thì bắt đầu xác thực
* Có 2 trường hợp:
  + **TH1:** Thông tin không hợp lệ:
  + Nếu thông tin không hợp lệ thì hàm doAuthenticate() sẽ thực hiện throw BadCredentialsException và gửi response không hợp lệ về cho phía client
  + Phía client sau khi nhận phản hồi không hợp lệ sẽ hiển thị thông báo sai thông tin không thể đăng nhập
* **TH2:**  Thông tin hợp lệ:
* Nếu thông tin hợp lệ thì sẽ lấy user token, user information, chuyển đổi về dạng json và doResponse về cho phía client với thông tin hợp lệ và đăng nhập thành công
* Phía client sau khi nhận được phản hồi thành công sẽ bắt đầu load file js, init view, Route quản lý các url và controller cho dashboard, cập nhật giao diện, ẩn login content đi và thêm dashboard content vào.

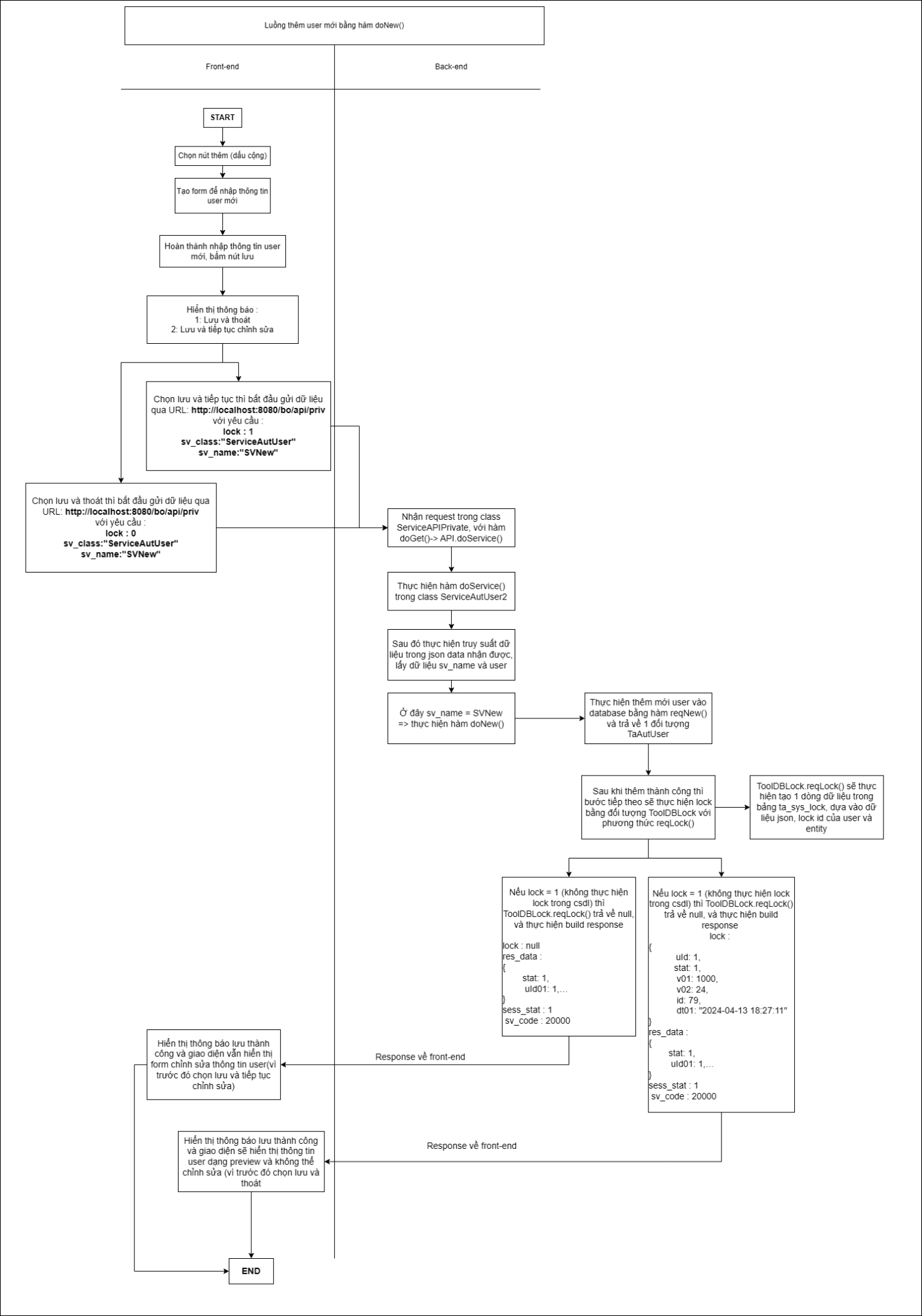
**II. Sơ lượt luồng chính sau khi đăng nhập:**



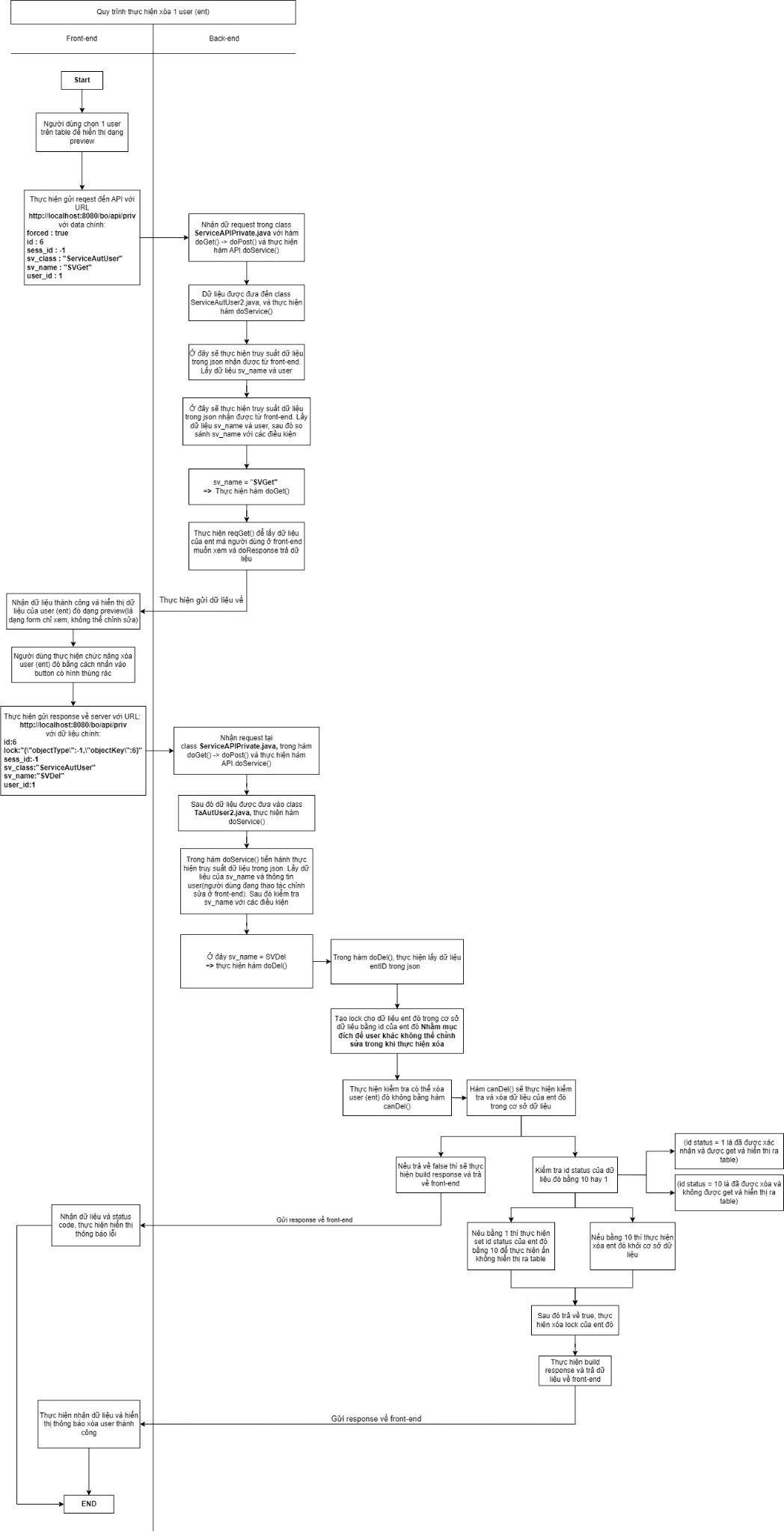
**III. Luồng hiển thị dữ liệu ra table:**

****

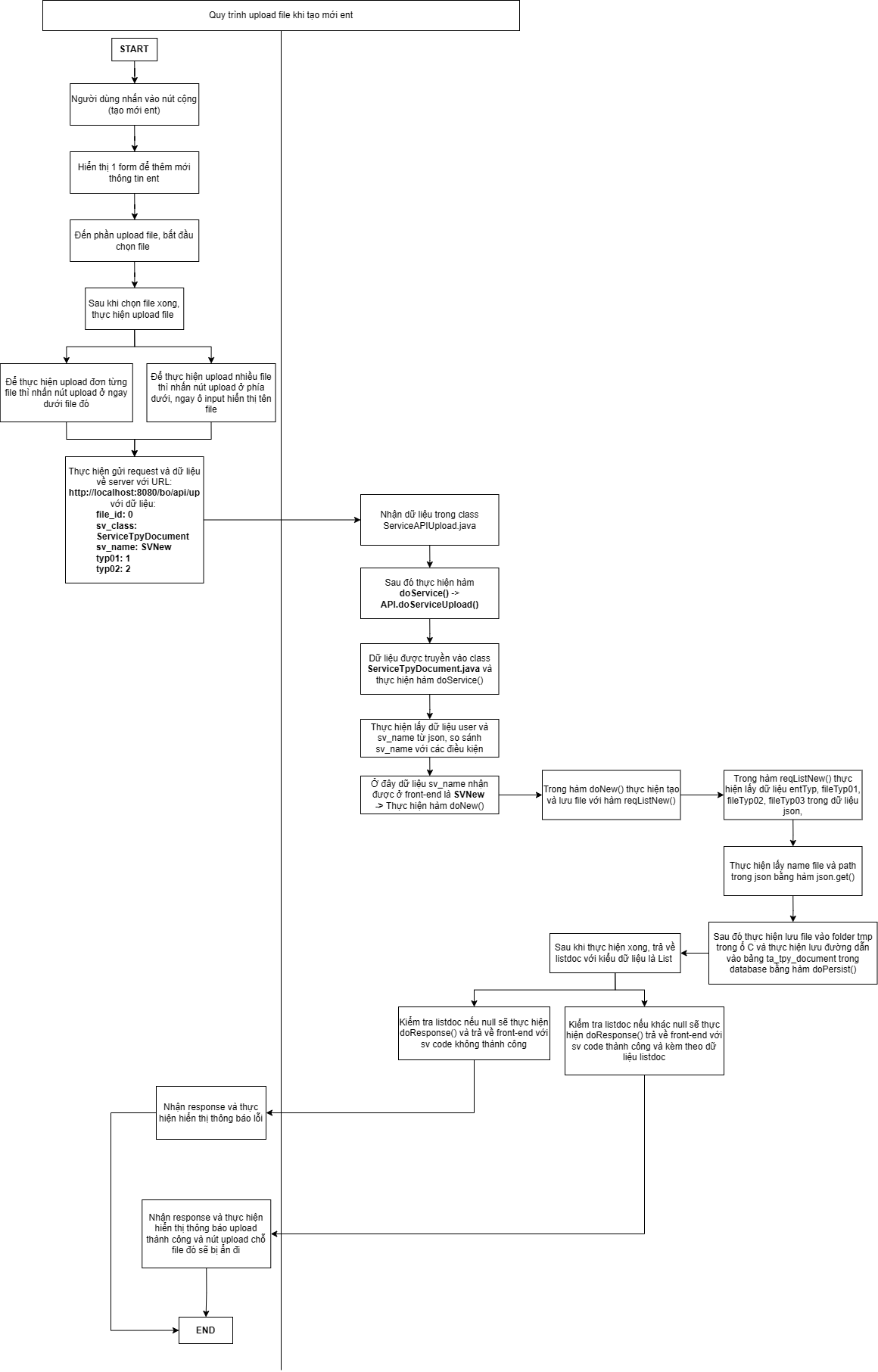
**IV. Luồng SVNew (thực hiện thêm mới 1 user):**



**V. Luồng SVDel (thực hiện xóa 1 user):**

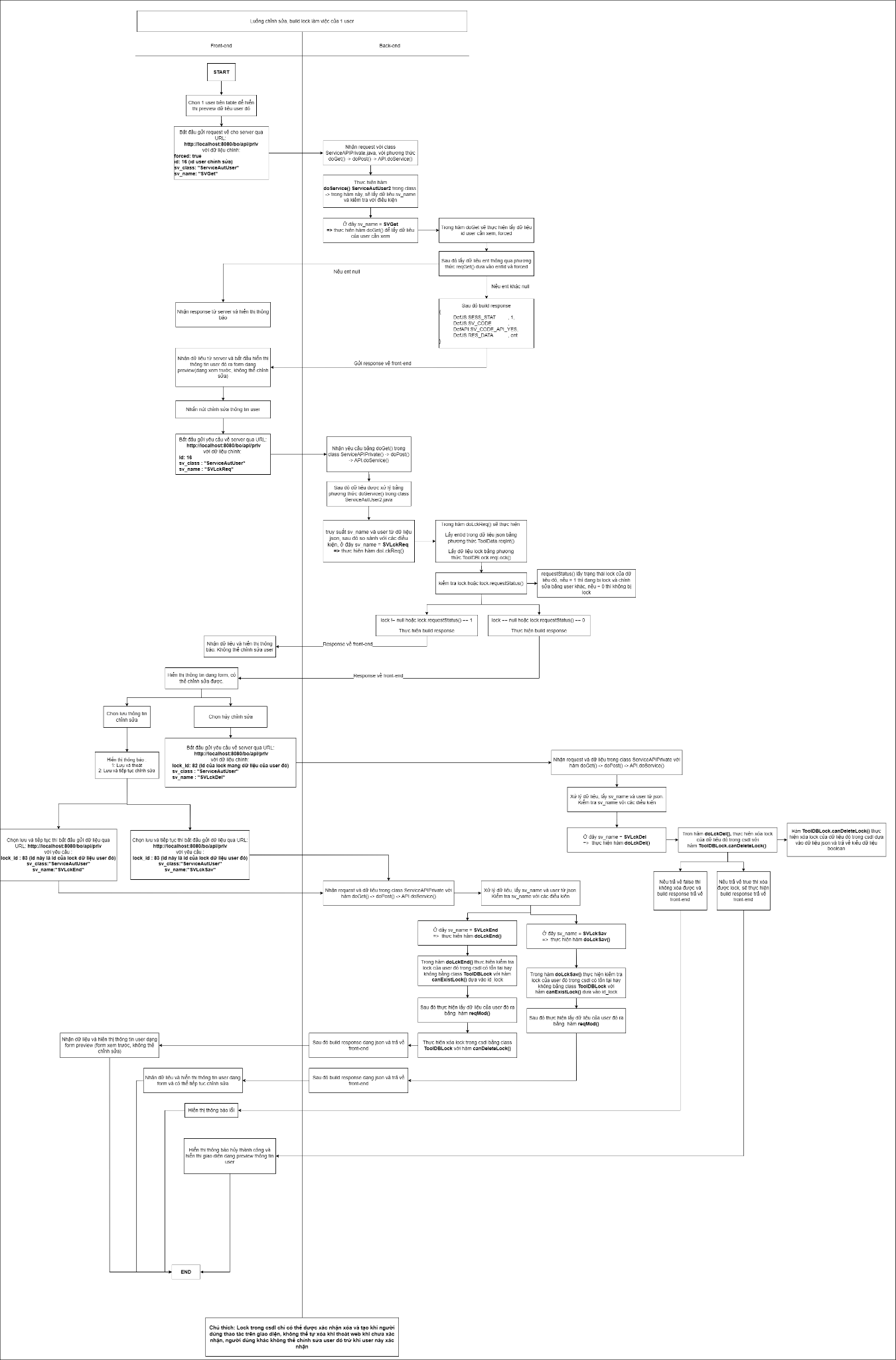
****

**VI. Upload file của 1 user:**



**VII. Luồng Lock và chỉnh sửa dữ liệu user:**

**(Sơ đồ có thể bị mờ nên em để link ở đây để thầy có thể xem rõ hơn:** <https://drive.google.com/file/d/1vA9mRHXRnH-ZhnMcIIcFhexso-LTN6ow/view?usp=sharing> **)**

****