A fix for the itch:

My resolution was adding CSRF Tokens to protect user sessions of armbook from CSRF attacks. To do this I added a token query parameter to add_friend and del_friend requests so that I can compare the session token to the query parameter token. If not equivalent, I throw an error. **Since the adversary will not know the randomized token query parameter is, they can't successfully add friends through my click jacking attack performed in step1 of act3.** I found information on how to do this through: https://web.archive.org/web/20211011172345/https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html#token-based-mitigation and its implementation on https://www.xenonstack.com/insights/stateful-and-stateless-applications (highly recommend reviewing both of these for deeper understanding.

Login.php assigns a randomly generated token value to the session, $_SESSION['token'] = bin2hex(random_bytes(32)), which is used to prevent undesired actions to authenticated users by providing an unpredictable token value (changes per session). One flaw of this method is that if an adversary has control of a proxy you are using to connect, they can obtain the parameter. I add the token as a query parameter in home.php and search.php. I implement token check in add_friend.php and del_friend.php (NOTE: this check can be implemented in other areas of armbook as well)

Edited Files:

login.php # Added token to SESSION

home.php # Added token to add_friend query

search.php # Added token to add_friend and del_friend query (recursively searching for add friend let me find this)

add_friend.php # Added token check

del_friend.php # Added token check

NOTE: Instead of a writeup describing just how the previous issue would be fixed I decided to fix it too.