Secure Programming
Assignment: "SP ++, (SPPP)"
Due in 3 parts: Nov. 4, Nov. 16, Dec. 2 (at 23:11, 11:11 PM)

Overview:

Create a Web service (server) to allow multiple, concurrent, users to share "items" which may be pictures, small videos, text and "binary" files. Users of the service will be "authenticated", and allowed several types of interactions (services) from SPPP.

There are two types of SPPP "users": general users and administrator.

General users who have been authenticated may: View, Put, Get, and Remove, (as well as edit) items: pictures, videos, text, and other format "binary" (raw) files.

Commands:

When an item is "put" into a group it will have an item name (up to 68 characters, "internationalized" (UTF-8 or similar)), the creator's name, an item descriptor (up to 1000 characters or symbols, internationalized), and time and date of creation and time and date of last access.

A "get" will retrieve the item from the service and copy to the user's local device.

A "remove" deletes the item (only the creator or admin may remove an item).

A "view", (which is the default user interaction), shows all items in all groups that a user is a member of, to the user. (As well as "meta" information, the creator's name, times and dates, and the description.)

Registration and Authentication:

A general user must register, she must register with her actual (not login) name of 80 characters/symbols (internationalized), a 16 character login name (ASCII, or any format you choose), a "sufficiently" good, long password, and the names (or selections) of groups she wishes to join.

Group names and general users must be accepted by the admin before being activated (asynchronously).

The admin must accept (allow) user (who have previously registered), requested groups, be able to create new groups, and be able to set some limits of space consumption: limits per individual user, limits per group, and limits per item.

Implementation:

Users (as well as admin) may operate independently (asynchronously).

Items (group files) should be stored in a "reasonable", retrievable form ("probably" SQL).

All passwords, protection, etc must be "secure".

You may implement this service in any reasonable programming language, data base, software middleware, infrastructure, and services.

Implementation is encouraged in a "safe" programming language (or must be made "safe"), the programming language "Rust" is strongly encouraged.

You should (of course) block, avoid, thwart, evade, hinder, and prevent attacks: script attacks, overflows, web/http/html, injections, timing attacks, passwords and authentication, and other "well known" vulnerabilities.

Of course, this service must be available "on the web", that is on a publicly available server (service). This may be done utilizing university or department computing resources, public (or private) "clouds", or your own web hosting.

Details:

All work must be your own, you may use information (or code) from books, articles, and the internet AS LONG as you completely give reference citations.

All work should be done individually, if for some reason you wish to work with one other person, you may make suggestion of non-trivial extensions, you must/should suggest these yourself, otherwise I might suggest very interesting, but perhaps "challenging" extensions (multi-factor authentication and public-private key encryption and load performance considerations, for example. You will not get a "menu" of suggestions, just a list of requirements.)

Submit:

You should submit this assignment (and you may be asked to demo parts) in three separate submissions.

Part 1 should have basic registration, and implement functionality for "put" and "view" (not all "put" options need to be complete, you should have at a minimum "putting" an item, creators name, time and date.

Part 2 should have completed "put", "view" and "get" functionality and be hosted on a remote server.

Part 3 should be all functionality, complete.