**Security**

**Change Default User on RaspBery Pi.**

Our raspberry pi is going to be exposed to the internet, this means it would open to hackers who can use the default username & password and can access it without permission.

**Changing default password.**

* Changing password is a top priority in the security of the app
* This is done in the raspberry pi terminal using the *sudo raspi-config*

**Changing default username.**

* All raspberry pi’s come with default username *pi*

To add a new user in raspberry pi we used *sudo adduser takudzwa\_karigomba98.* A password is then prompt to be entered for the new user.The new user profile now has a home directory at */home/takudzwa\_karigomba98/.*

To add more security to our pi: We added our profile to the sudo group. This allows to give sudo permissionsusing the command:

* *sudo usermod -a -G adm,dialout,cdrom,sudo,audio,video,plugdev,games,users,input,netdev,gpio,i2c,spi takudzwa\_karigomba98.*

To make sure sudo forces a require password we used: *sudo visudo /etc/sudoers.d/010\_pi-nopasswd* and changed the user entry to *pi ALL=(ALL) PASSWD: ALL.*

**Password Hashing**

In the absence hashing, any password stored within our application’s database could get robbed if the database is breached. Hackers can promptly use the information to compromise our application and the accounts of all the users within our database, if they use passwords that aren’t strong or aren’t unique.

Chart, diagram

Description automatically generatedPassword hashing will carry out a one-way transformation on the passwords that are going to be stored in the database for the application. This security feature will be added to make sure personal information is not leaked, as we don’t want non authorized personal’s to read personal information in plain text. Once the passwords are stored hash in our database, it is will be very difficult or hackers/non authorized personals to read the passwords and turn the hashed password back into the original password. Figure 1(Google Images)

**Log-in Interface**

**Passwords.**

Many users are prone to making weak passwords.

* A regex pattern can help user’s make better password*"^(?=.\*[A-Za-z])(?=.\*\d)[A-Za-z\d]{8,}$".* This regex pattern will validate password strength. If the user makes the password without the minimum requirements they will be prompt to do it again until they have matched the regex pattern and receive this error “***Minimum eight characters, at least one letter and one number”.***
* If a user enters the wrong password or the wrong username, the feedback on the screen will say wrong email/password. This will give security to the log-in system as a if unauthorized person tried to log into someone’s profile, they will not know which wrong the email or password.
* Many applications have the option to stay signed so the user never has to enter their details ever again. Our application will use the remember option instead. This feature adds to the security e.g If user uses loses their phone or it gets stolen they wont be stayed signed in. The application will only remember the email.
* Passwords can easily be seen if entering on a open/big screen. Putting an option for cover password can also help the user to cover their private information. This gives the user to have two options. It will help the user from making password error as they can switch between the two options