PasswordProtectionProgram

Generated by Doxygen 1.8.13

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

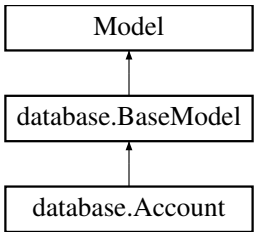
Chapter 4

Class Documentation

4.1 database.Account Class Reference

SQLite table to store passwords.

Inheritance diagram for database.Account:



Static Public Attributes

- **ID** = peewee.PrimaryKeyField()
- **AccName** = peewee.CharField(unique=True)
- **AccType** = peewee.CharField()
- **UserName** = peewee.CharField(null=True)

4.1.1 Detailed Description

SQLite table to store passwords.

Use peewee orm library to create a table class that stores accounts

Parameters

AccID	Account ID and Primary Key
AccType	Type of Account used
UserName	Account Username

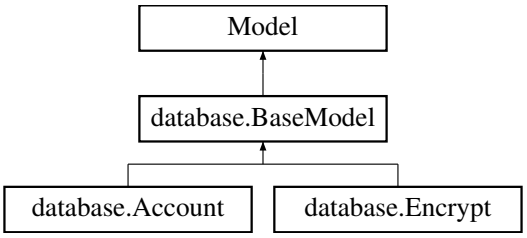
The documentation for this class was generated from the following file:

- database.py

4.2 database.BaseModel Class Reference

Base Model for database connection.

Inheritance diagram for database.BaseModel:



Classes

- class Meta

4.2.1 Detailed Description

Base Model for database connection.

All other Tables will connect automatically to our database

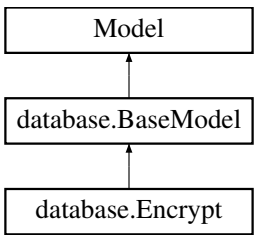
The documentation for this class was generated from the following file:

- database.py

4.3 database.Encrypt Class Reference

SQLite table to store hash keys and hash values.

Inheritance diagram for database.Encrypt:



Static Public Attributes

- **ID** = peewee.ForeignKeyField(Account, to_field='ID', primary_key=True, on_delete='CASCADE')
- **HashVal** = peewee.CharField()
- **HashKey** = peewee.FixedCharField(44, unique=True)

4.3.1 Detailed Description

SQLite table to store hash keys and hash values.

use peewee orm library to create a table class that stores hash values and hash keys in a database. This table is not accessabile via the application.

Parameters

Eid	Encrypted Password ID and Foreign key from Account ID
HashVal	Hashed value of Password
HashKey	Key to Decrypt Password

The documentation for this class was generated from the following file:

- database.py

4.4 database.BaseModel.Meta Class Reference

Static Public Attributes

- **database** = db

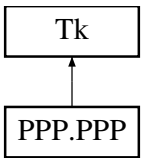
The documentation for this class was generated from the following file:

- database.py

4.5 PPP.PPP Class Reference

An ADT that represents the GUI.

Inheritance diagram for PPP.PPP:



Public Member Functions

- def __init__ (self, args)

  PPP constructor.
- def showHomeScreen (self, state)

  Home Screen.
- def checkPW (self, frame, label, entry, state)

  Check Password.
- def showEntry (self, frame, detailFrame)

  Show entry.
- def showPWPage (self, args)

  Password Management Screen.
- def addEntry (self, scrollFrame, detailFrame, pwd)

  Add entry to database and display in scrollbar frame.
- def viewDetails (self, idnum, frame)

  Displays details of entry.
- def userManual (self, frame)

  Displays instructions on how to use product.
- def destroyF (self, frame)

  Destroys frame.
- def update (self, i, new, pw, msg)

  Updates database.

Public Attributes

- **view**
- **copy**
- **gen**
- **delete**
- **logo**

4.5.1 Detailed Description

An ADT that represents the GUI.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 __init__()

```
def PPP.PPP.__init__ (
              self,
              args )
```

PPP constructor.

Initializes a PPP GUI object using a variable argument list

Parameters

∗*args*	A variable argument list that contains information that should be dsplayed in GUI

4.5.3 Member Function Documentation

4.5.3.1 addEntry()

```
def PPP.PPP.addEntry (
              self,
              scrollFrame,
              detailFrame,
              pwd )
```

Add entry to database and display in scrollbar frame.

Adds entry to the database and the display

Parameters

scrollFrame	The frame on the left which displays the entries as buttons
detailFrame	The frame on the right which displays details of each entry
∗*pwd*	Variable list of entries from the user when he/she adds an entry

4.5.3.2 checkPW()

```
def PPP.PPP.checkPW (
              self,
              frame,
              label,
              entry,
              state )
```

Check Password.

Checks if password is good for registration or correct for logging in, depending on input

Parameters

frame	The frame which called the method, so it can be updated to show something else upon entering a right password
label	The label that will be updated to show the user what he/she needs to do
entry	The entered password
state	Tells if we are checking for password matching (for LogIn) or criteria (for Registering)

4.5.3.3 destroyF()

```
def PPP.PPP.destroyF (
              self,
              frame )
```

Destroys frame.

Recursively destroys each widget in frame frame Frame you wish to destroy

4.5.3.4 showEntry()

```
def PPP.PPP.showEntry (
              self,
              frame,
              detailFrame )
```

Show entry.

Shows entries that already exists as a button on the left scrolling frame

Parameters

frame	The frame that displays button
detailFrame	The frame that will show further details if button on left frame is clicked

4.5.3.5 showHomeScreen()

```
def PPP.PPP.showHomeScreen (
              self,
              state )
```

Home Screen.

Displays the first window which can either be a registration frame or log in frame

Parameters

state	List of what goes in the frame, namely if it is a register frame or log in frame

4.5.3.6 showPWPage()

```
def PPP.PPP.showPWPage (
```

```
                self,
                args )
```

Password Management Screen.

Where user can add and view account information

Parameters

**args*	A variable argument list

4.5.3.7 update()

```
def PPP.PPP.update (
                self,
                i,
                new,
                pw,
                msg )
```

Updates database.

when user makes change to password or username, it gets updated in database

Parameters

i	The id number of the entry
new	The updated entry
pw	Is it a password or username? Boolean value msg Update the message to show that changes were successfully made

4.5.3.8 userManual()

```
def PPP.PPP.userManual (
                self,
                frame )
```

Displays instructions on how to use product.

Step by step instructions and link to user guide pdf

Parameters

frame	The frame on which the manual is to be displayed

4.5.3.9 viewDetails()

```
def PPP.PPP.viewDetails (
              self,
              idnum,
              frame )
```

Displays details of entry.

Displays details of entry (type, name, username, password), called when button for entry is clicked

Parameters

idnum	The id number of the entry that was clicked
frame	The frame in which to display the details on

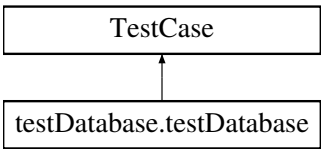
The documentation for this class was generated from the following file:

- PPP.py

4.6 testDatabase.testDatabase Class Reference

This class is used to test the functions in database.py.

Inheritance diagram for testDatabase.testDatabase:



Public Member Functions

- def setUp (self)

  This method creates a list of keys using the Encrypt module as well as create empty tables.
- def tearDown (self)

  This method deletes the tables in the database.
- def test_FRDB4 (self)

  This method tests the Get functions by checking that all Get functions return equal outputs "Test".
- def test_FRDB1 (self)

  THis method tests the insertion into an empty database, as this is the master password.
- def test_FRDB2_1 (self)

  This method tests the Update password Function.
- def test_FRDB2_2 (self)

  This function tests the Update username Function.
- def test_FRDB3 (self)

  This function test the Delete row Function.

Public Attributes

- **keys**

4.6.1 Detailed Description

This class is used to test the functions in database.py.

4.6.2 Member Function Documentation

4.6.2.1 test_FRDB3()

```
def testDatabase.testDatabase.test_FRDB3 (
              self )
```

This function test the Delete row Function.

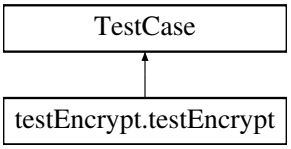
The documentation for this class was generated from the following file:

- testDatabase.py

4.7 testEncrypt.testEncrypt Class Reference

This class is used to test the functions in Encrypt.py and the functional requirements FR8 and FR11.

Inheritance diagram for testEncrypt.testEncrypt:



Public Member Functions

- def setUp (self)

  This method creates various passwords and their associated keys.
- def test_cryptDecode1 (self)

  This method is used to test if the decoded password matched the original password (FR-E-1 & FR-E-2).
- def **test_cryptDecode2** (self)
- def **test_cryptDecode3** (self)

Public Attributes

- **p1**
- **p2**
- **p3**
- **k1**
- **k2**
- **k3**
- **e1**
- **e2**
- **e3**

4.7.1 Detailed Description

This class is used to test the functions in Encrypt.py and the functional requirements FR8 and FR11.

4.7.2 Member Function Documentation

4.7.2.1 test_cryptDecode1()

```
def testEncrypt.testEncrypt.test_cryptDecode1 (
              self )
```

This method is used to test if the decoded password matched the original password (FR-E-1 & FR-E-2).

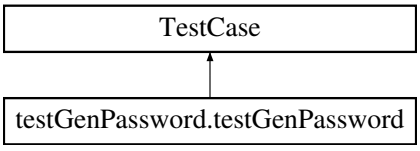
The documentation for this class was generated from the following file:

- testEncrypt.py

4.8 testGenPassword.testGenPassword Class Reference

This class is used to test the functions in GenPassword.py and functional requirement FR6.

Inheritance diagram for testGenPassword.testGenPassword:



Public Member Functions

- def setUp (self)

  This method creates various passwords and their associated keys.
- def test_GenPass1 (self)

  These methods are used to test if the generated string is 8 characters long and contains the necessary security features.
- def **test_GenPass2** (self)
- def **test_GenPass3** (self)

Public Attributes

- **p1**
- **p2**
- **p3**

4.8.1 Detailed Description

This class is used to test the functions in GenPassword.py and functional requirement FR6.

4.8.2 Member Function Documentation

4.8.2.1 test_GenPass1()

```
def testGenPassword.testGenPassword.test_GenPass1 (
              self )
```

These methods are used to test if the generated string is 8 characters long and contains the necessary security features.

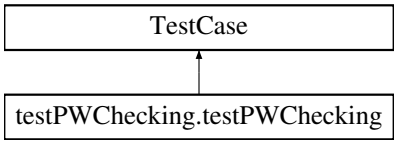
The documentation for this class was generated from the following file:

- testGenPassword.py

4.9 testPWChecking.testPWChecking Class Reference

This class is used to test the functions in PWChecking.py.

Inheritance diagram for testPWChecking.testPWChecking:



Public Member Functions

- def setUp (self)

  This method creates various passwords for creation as well as checking upon login.
- def test_FRMP1 (self)

  This method is used to test FR-MP-1, which checks that a password with at least 8 characters, at least 1 uppercase and at least one number returns True.
- def test_FRMP2 (self)

  This method is used to test FR-MP-2, which checks that a password without any numbers gives the appropriate message.
- def test_FRMP3 (self)

  This method is used to test FR-MP-3, which checks that a password with less than 8 characters gives the appropriate message.
- def test_FRMP4 (self)

  This method is used to test FR-MP-4, which checks that a password without uppercase gives the appropriate message.
- def test_FRMP5 (self)

  This method is used to test FR-MP-5, which checks that an empty password gives the appropriate message.
- def test_FRMP6 (self)

  This method is used to test FR-MP-6, which checks that an empty password upon login gives the appropriate message.
- def test_FRMP7 (self)

  This method is used to test FR-MP-7, which checks that a password upon login that is same as actual returns True.
- def test_FRMP8 (self)

  This method is used to test FR-MP-8, which checks that a password upon login that is not the same as actual gives the appropriate message.

Public Attributes

- **mp1**
- **mp2**
- **mp3**
- **mp4**
- **mp5**
- **actual**
- **mp6**
- **mp7**
- **mp8**

4.9.1 Detailed Description

This class is used to test the functions in PWChecking.py.

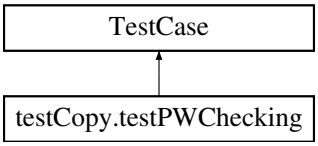
The documentation for this class was generated from the following file:

- testPWChecking.py

4.10 testCopy.testPWChecking Class Reference

This class is used to test the functions in Copy.py.

Inheritance diagram for testCopy.testPWChecking:



Public Member Functions

- def setUp (self)

  This method creates a tkinter window.
- def test_FRN5 (self)

  This method is used to test FR-N-5, which tests the copy function.

Public Attributes

- **root**
- **c1**

4.10.1 Detailed Description

This class is used to test the functions in Copy.py.

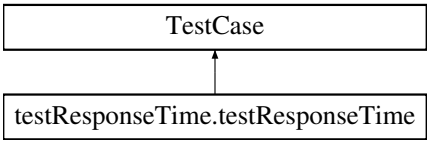
The documentation for this class was generated from the following file:

- testCopy.py

4.11 testResponseTime.testResponseTime Class Reference

This class is used to test that performance requirements are met.

Inheritance diagram for testResponseTime.testResponseTime:



Public Member Functions

- def test_NFR_PER_1 (self)

  Test processing time of reset timer.
- def test_NFR_PER_2 (self)

  Test error time of incorrect password.

4.11.1 Detailed Description

This class is used to test that performance requirements are met.

4.11.2 Member Function Documentation

4.11.2.1 test_NFR_PER_1()

```
def testResponseTime.testResponseTime.test_NFR_PER_1 (
              self )
```

Test processing time of reset timer.

If ran for too long, inactivity method throws error. Note in test report.

The documentation for this class was generated from the following file:

- testResponseTime.py

Chapter 5

File Documentation

5.1 Constants.py File Reference

The constants being used in the GUI file (fonts, colours)

Variables

- string **Constants.BGC** = "#cccccc"
- string **Constants.BG** = "#383A39"
- string **Constants.FG** = "#A1DBCD"
- tuple **Constants.LARGE** = ("Helvetica", 16)
- string **Constants.screen_size** = "820x640"
- string **Constants.EYE** = "icons/eye.gif"
- string **Constants.COPY** = "icons/copy.gif"
- string **Constants.GENERATE** = "icons/flash.gif"
- string **Constants.DELETE** = "icons/delete.gif"
- string **Constants.LOGO** = "icons/ppp.gif"
- string **Constants.ICON** = "icons/ppp.ico"
- string **Constants.WELCOME** = "Welcome to the Password Protection Program"
- dictionary **Constants.REGISTER**
- dictionary **Constants.LOGIN**
- list **Constants.FIELDS** = ["Name", "Type", "Username", "Password"]
- int **Constants.ERR_TIME** = 1000
- int **Constants.PROC_TIME** = 1000
- string **Constants.USERMANUAL** = "..\\Doc\\UserGuide\\UserGuide.pdf"
- string **Constants.INSTRUCTIONS**
- int **Constants.INACTIVITY** = 30000

5.1.1 Detailed Description

The constants being used in the GUI file (fonts, colours)

Author

Suhavi Sandhu

Date

November 10, 2017

5.1.2 Variable Documentation

5.1.2.1 INSTRUCTIONS

```
string Constants.INSTRUCTIONS
```

Initial value:

```
1 =  "1) Add entries on the top-left hand corner!\
2                 2) Generate a username/password using the lightning icon.                          \
3                 3) View entries by clicking the buttons with an eyeball.                           \
4                 4) Edit entry by viewing and updating on this side of the screen.                  \
5                 NOTE: If you are still having trouble, read the user manual by clicking below. "
```

5.1.2.2 LOGIN

```
dictionary Constants.LOGIN
```

Initial value:

```
1 =  {
2     "prompt": "Enter your master password",
3     "loggedIn": True }
```

5.1.2.3 REGISTER

```
dictionary Constants.REGISTER
```

Initial value:

```
1 =  {
2     "prompt": "Create a master password to start using the application. Must have lowercase, uppercase,
       numbers and at least 8 characters",
3     "loggedIn": False }
```

5.2 Copy.py File Reference

Copies the text after the user clicks a button.

Functions

- def Copy.copy (text)

  Copies the text.

5.2.1 Detailed Description

Copies the text after the user clicks a button.

Author

Suhavi Sandhu

Date

November 10, 2017

5.2.2 Function Documentation

5.2.2.1 copy()

```
def Copy.copy (
              text )
```

Copies the text.

Parameters

text	The text to be copieds

5.3 database.py File Reference

PPP_database

Classes

- class database.BaseModel

  Base Model for database connection.
- class database.BaseModel.Meta
- class database.Account

  SQLite table to store passwords.
- class database.Encrypt

  SQLite table to store hash keys and hash values.

Functions

- def database.CreateTables ()
  Instantiate new empty tables.
- def database.DropTables ()
  Delete tables.
- def database.Insert (N, T, U, Hv, Hk)
  Insert new Account Instance and Encrypt Instance.
- def database.GetAll ()
  get all queries
- def database.GetId (id_)
  search tables with AccId
- def database.GetT (Atype)
  Get Table Rows with Account Type.
- def database.GetN (Aname)
  Get Table Row with Account name.
- def database.Delete (id_)
  Delete Table Row with ID.
- def database.UpdateU (Aid, U)
  Update Table Row with ID.
- def database.UpdateP (Aid, Hv)
  Update Password with ID.

Variables

- **database.db** = peewee.SqliteDatabase('pppDatabase.db')

5.3.1 Detailed Description

PPP_database

Author

Joseph Lu, luy89

Date

20/10/2017

5.3.2 Function Documentation

5.3.2.1 CreateTables()

```
def database.CreateTables ( )
```

Instantiate new empty tables.

Encrypt Table should also be reset when Account is reset

5.3.2.2 Delete()

```
def database.Delete (
              id_ )
```

Delete Table Row with ID.

Parameters

id	Account ID

5.3.2.3 DropTables()

```
def database.DropTables ( )
```

Delete tables.

Encrypt Table should also be reset when Account is reset

5.3.2.4 GetId()

```
def database.GetId (
              id_ )
```

search tables with AccId

Parameters

Id	Account Id

5.3.2.5 GetN()

```
def database.GetN (
              Aname )
```

Get Table Row with Account name.

Parameters

Aname	Account Name

5.3.2.6 GetT()

```
def database.GetT (
              Atype )
```

Get Table Rows with Account Type.

Parameters

Atype	Account type

5.3.2.7 Insert()

```
def database.Insert (
              N,
              T,
              U,
              Hv,
              Hk )
```

Insert new Account Instance and Encrypt Instance.

Parameters

N	Account Name
T	Account Type
U	username
Hv	Hash Value
Hk	Hash Key

5.3.2.8 UpdateP()

```
def database.UpdateP (
              Aid,
              Hv )
```

Update Password with ID.

Parameters

Aid	Account Id (Encrypted ID)
Hv	new Hash Value

5.3.2.9 UpdateU()

```
def database.UpdateU (
              Aid,
              U )
```

Update Table Row with ID.

Parameters

Aid	Account Id
U	new Username
Hv	new Hash Value

5.4 Encrypt.py File Reference

This module handles the key generation for and encryption/decryption of passwords.

Functions

- def Encrypt.generKey ()

  This function generates a unique) key (for encoding user passwords) using Python's Fernet library.
- def Encrypt.cryptEncode (key, passw)

  This function uses a key to encrypt an input string.
- def Encrypt.cryptDecode (key, encrypted)

  This function uses the saved key to decrypt the encrypted user password stored in the database.

5.4.1 Detailed Description

This module handles the key generation for and encryption/decryption of passwords.

Author

Shabana Dhayananth

Date

October 15, 2017

5.4.2 Function Documentation

5.4.2.1 cryptDecode()

```
def Encrypt.cryptDecode (
            key,
            encrypted )
```

This function uses the saved key to decrypt the encrypted user password stored in the database.

Parameters

key,encrypted	refer to the key that was used to encrpt the password and the encrypted password

Returns

decrypted password in string format

5.4.2.2 cryptEncode()

```
def Encrypt.cryptEncode (
              key,
              passw )
```

This function uses a key to encrypt an input string.

Fernet uses symmetric encryption on the input key

Parameters

key,passw	refer to the key to be used to encypt and the password to be encrypted

Returns

encrypted password in byte format

5.4.2.3 generKey()

```
def Encrypt.generKey ( )
```

This function generates a unique) key (for encoding user passwords) using Python's Fernet library.

key derived from a string that is run through the kdf (key derivation function)

Returns

key that will be used to encode the user passwords (32 bytes)

5.5 GenPassword.py File Reference

This module is used to generate a random password containing alphanumeric characters.

Functions

- def GenPassword.genPass ()

  This function generates a random password consisting of upper case, lower case alphanumeric characters.
- def GenPassword.genPassCrypt ()

  This function generates a random password consisting of upper case, lower case alphanumeric characters.

5.5.1 Detailed Description

This module is used to generate a random password containing alphanumeric characters.

Author

Shabana Dhayananth

Date

October 27, 2017

5.5.2 Function Documentation

5.5.2.1 genPass()

```
def GenPassword.genPass ( )
```

This function generates a random password consisting of upper case, lower case alphanumeric characters.

default random number generator's sequences can be reproduced, in case SystemRandom() is not available on user system

Returns

random password consisting of 8 characters

5.5.2.2 genPassCrypt()

```
def GenPassword.genPassCrypt ( )
```

This function generates a random password consisting of upper case, lower case alphanumeric characters.

Same as genPass() but uses SystemRandom() to generate random numbers so sequences are not reproducible

Returns

random password consisting of 8 characters

5.6 PPP.py File Reference

The graphical user interface for a password manager.

Classes

- class PPP.PPP

  An ADT that represents the GUI.

Functions

- def PPP.inactive ()

  Activates when user is inactive for more than x time.
- def PPP.reset_timer ()

  resets the timer

Variables

- **PPP.app** = PPP()
- **PPP.timer** = None

5.6.1 Detailed Description

The graphical user interface for a password manager.

Author

Suhavi Sandhu

Date

November 10, 2017

5.6.2 Function Documentation

5.6.2.1 inactive()

```
def PPP.inactive ( )
```

Activates when user is inactive for more than x time.

Currently shows login by default, can easily add database check

5.7 PWChecking.py File Reference

Check Passwords

Functions

- def PWChecking.checkMP (password)

  Checks master password at creation.
- def PWChecking.checkLogIn (entered, actual)

  Checks master password at login.

5.7.1 Detailed Description

Check Passwords

Author

Suhavi Sandhu

Date

November 10, 2017

5.7.2 Function Documentation

5.7.2.1 checkLogIn()

```
def PWChecking.checkLogIn (
              entered,
              actual )
```

Checks master password at login.

Verifies that the entered password matches the actual

Parameters

entered	The password entered by the user
actual	The real master password

5.7.2.2 checkMP()

```
def PWChecking.checkMP (
              password )
```

Checks master password at creation.

Verifies that the password meets criteria

Parameters

password	The password that is being checked

5.8 testCopy.py File Reference

This file unittests the functions used to copy words.

Classes

- class testCopy.testPWChecking

  This class is used to test the functions in Copy.py.

5.8.1 Detailed Description

This file unittests the functions used to copy words.

Author

Suhavi Sandhu

Date

November 28, 2017

5.9 testDatabase.py File Reference

This file unittests the functions used for.

Classes

- class testDatabase.testDatabase

  This class is used to test the functions in database.py.

5.9.1 Detailed Description

This file unittests the functions used for.

Author

Joseph Lu

Date

November 16, 2017

5.10 testEncrypt.py File Reference

This file unittests the functions used for key generation, encryption and decryption.

Classes

- class testEncrypt.testEncrypt

  This class is used to test the functions in Encrypt.py and the functional requirements FR8 and FR11.

5.10.1 Detailed Description

This file unittests the functions used for key generation, encryption and decryption.

Author

Shabana Dhayananth

Date

November 16, 2017

5.11 testGenPassword.py File Reference

This file unittests the the function used to generate random strings for username and password.

Classes

- class testGenPassword.testGenPassword

  This class is used to test the functions in GenPassword.py and functional requirement FR6.

5.11.1 Detailed Description

This file unittests the the function used to generate random strings for username and password.

The randomness of the values cannot be tested but the requirements for the string can be tested.

Author

Shabana Dhayananth

Date

November 28, 2017

5.12 testPWChecking.py File Reference

This file unittests the functions used to test password checking at registration and login.

Classes

- class testPWChecking.testPWChecking
  This class is used to test the functions in PWChecking.py.

5.12.1 Detailed Description

This file unittests the functions used to test password checking at registration and login.

Author

Suhavi Sandhu

Date

November 28, 2017

5.13 testResponseTime.py File Reference

To test that the application responds within the appropriate time.

Classes

- class testResponseTime.testResponseTime
  This class is used to test that performance requirements are met.

5.13.1 Detailed Description

To test that the application responds within the appropriate time.

Author

Suhavi Sandhu

Date

December 6, 2017

Index