

## PasswordProtectionProgram

Generated by Doxygen 1.8.13



# Contents

<b>1</b>	<b>Hierarchical Index</b>	<b>1</b>
1.1	Class Hierarchy . . . . .	1
<b>2</b>	<b>Class Index</b>	<b>3</b>
2.1	Class List . . . . .	3
<b>3</b>	<b>File Index</b>	<b>5</b>
3.1	File List . . . . .	5
<b>4</b>	<b>Class Documentation</b>	<b>7</b>
4.1	database.Account Class Reference . . . . .	7
4.1.1	Detailed Description . . . . .	7
4.2	database.BaseModel Class Reference . . . . .	8
4.2.1	Detailed Description . . . . .	8
4.3	database.Encrypt Class Reference . . . . .	8
4.3.1	Detailed Description . . . . .	9
4.4	database.BaseModel.Meta Class Reference . . . . .	9
4.5	PPP.PPP Class Reference . . . . .	9
4.5.1	Detailed Description . . . . .	10
4.5.2	Constructor & Destructor Documentation . . . . .	10
4.5.2.1	__init__() . . . . .	10
4.5.3	Member Function Documentation . . . . .	11
4.5.3.1	addEntry() . . . . .	11
4.5.3.2	checkMP() . . . . .	11
4.5.3.3	matchPassword() . . . . .	11
4.5.3.4	showCreateMP() . . . . .	12
4.5.3.5	showEntry() . . . . .	12
4.5.3.6	showLogin() . . . . .	13
4.5.3.7	showPWPage() . . . . .	13
4.5.3.8	viewDetails() . . . . .	13

<b>5</b>	<b>File Documentation</b>	<b>15</b>
5.1	database.py File Reference . . . . .	15
5.1.1	Detailed Description . . . . .	16
5.1.2	Function Documentation . . . . .	16
5.1.2.1	CreateTables() . . . . .	16
5.1.2.2	Delete() . . . . .	16
5.1.2.3	DropTables() . . . . .	16
5.1.2.4	GetId() . . . . .	17
5.1.2.5	GetN() . . . . .	17
5.1.2.6	GetT() . . . . .	17
5.1.2.7	Insert() . . . . .	17
5.1.2.8	UpdateP() . . . . .	18
5.1.2.9	UpdateU() . . . . .	18
5.2	Encrypt.py File Reference . . . . .	18
5.2.1	Detailed Description . . . . .	19
5.2.2	Function Documentation . . . . .	19
5.2.2.1	cryptDecode() . . . . .	19
5.2.2.2	cryptEncode() . . . . .	19
5.2.2.3	generKey() . . . . .	20
5.3	GenPassword.py File Reference . . . . .	20
5.3.1	Detailed Description . . . . .	20
5.3.2	Function Documentation . . . . .	21
5.3.2.1	genPass() . . . . .	21
5.3.2.2	genPassCrypt() . . . . .	21
5.4	PPP.py File Reference . . . . .	21
5.4.1	Detailed Description . . . . .	22
5.5	PWChecking.py File Reference . . . . .	22
5.5.1	Detailed Description . . . . .	22
5.5.2	Function Documentation . . . . .	22
5.5.2.1	checkLogIn() . . . . .	22
5.5.2.2	checkMP() . . . . .	23
	<b>Index</b>	<b>25</b>

# Chapter 1

## Hierarchical Index

### 1.1 Class Hierarchy

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

database.BaseModel.Meta . . . . .	9
Model	
database.BaseModel . . . . .	8
database.Account . . . . .	7
database.Encrypt . . . . .	8
Tk	
PPP.PPP . . . . .	9



## Chapter 2

# Class Index

### 2.1 Class List

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

<a href="#">database.Account</a>	SQLite table to store passwords . . . . .	7
<a href="#">database.BaseModel</a>	Base Model for database connection All other Tables will connect automatically to our database	8
<a href="#">database.Encrypt</a>	SQLite table to store hash keys and hash values . . . . .	8
<a href="#">database.BaseModel.Meta</a>	. . . . .	9
<a href="#">PPP.PPP</a>	An ADT that represents the GUI . . . . .	9





## Chapter 3

# File Index

### 3.1 File List

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

<a href="#">database.py</a>		
	PPP_database . . . . .	15
<a href="#">Encrypt.py</a>		
	Password Encryption . . . . .	18
<a href="#">GenPassword.py</a>		
	Generate Random Password . . . . .	20
<a href="#">PPP.py</a>		
	The graphical user interface for a password manager . . . . .	21
<a href="#">PWChecking.py</a>		
	Check Passwords . . . . .	22



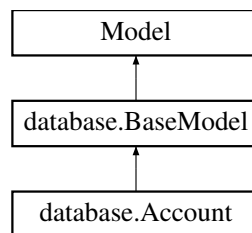
## Chapter 4

# 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()
- **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

<i>AccID</i>	<a href="#">Account</a> ID and Primary Key
<i>AccType</i>	Type of <a href="#">Account</a> used
<i>UserName</i>	<a href="#">Account</a> Username

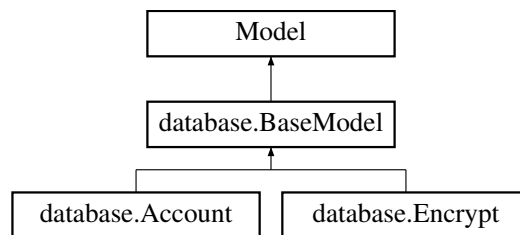
The documentation for this class was generated from the following file:

- [database.py](#)

## 4.2 database.BaseModel Class Reference

Base Model for database connection All other Tables will connect automatically to our database.

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.

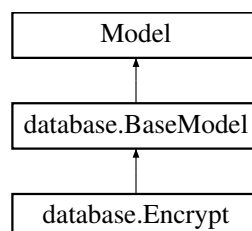
The documentation for this class was generated from the following file:

- [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(10)

#### 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 accessible via the application.

#### Parameters

<i>Eid</i>	Encrypted Password ID and Foreign key from <a href="#">Account</a> ID
<i>HashVal</i>	Hashed value of Password
<i>HashKey</i>	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

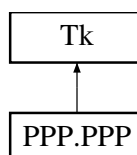
The documentation for this class was generated from the following file:

- [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 `showLogin` (self, kwargs)  
*Log In Screen.*
- def `matchPassword` (self, frame, kwargs)  
*Match Password.*
- def `showCreateMP` (self, args)  
*Master Password Creation Screen.*
- def `checkMP` (self, frame, kwargs)  
*Check Master Password.*
- def `showEntry` (self, frame, detailFrame)  
*Show entry.*
- def `showPWPage` (self, args)  
*Password Management Screen.*
- def `addEntry` (self, scrollFrame, canvas, detailFrame, pwd)  
*Add entry to database and display in scrollbar frame.*
- def `viewDetails` (self, idnum, frame)  
*Displays details of entry.*

## Public Attributes

- **accounts**
- **img**

### 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

<b>*args</b>	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,
    canvas,
    detailFrame,
    pwd )
```

Add entry to database and display in scrollbar frame.

Adds entry to the database and the display

##### Parameters

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

#### 4.5.3.2 checkMP()

```
def PPP.PPP.checkMP (
    self,
    frame,
    kwargs )
```

Check Master Password.

Checks if password meets criteria for master password creation

##### Parameters

<i>frame</i>	The frame which called the method, so it can be updated to show something else upon entering a satisfactory password
<i>**kwargs</i>	A variable argument list, in this case takes the label that tells you if incorrect password and entry from user

#### 4.5.3.3 matchPassword()

```
def PPP.PPP.matchPassword (
    self,
```

```
        frame,  
        kwargs )
```

Match Password.

Checks if password matches master password stored in database

#### Parameters

<i>frame</i>	The frame which called the method, so it can be updated to show something else upon entering the correct password
<i>**kwargs</i>	A variable argument list, in this case takes the label that tells you if incorrect password and entry from user

#### 4.5.3.4 showCreateMP()

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

Master Password Creation Screen.

Displays the master password creation frame

#### Parameters

<i>*kwargs</i>	A variable argument list to provide extra functionality to the frame
----------------	--

#### 4.5.3.5 showEntry()

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

Show entry.

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

#### Parameters

<i>frame</i>	The frame that displays button
<i>detailFrame</i>	The frame that will show further details if button on left frame is clicked



#### 4.5.3.6 showLogin()

```
def PPP.PPP.showLogIn (
    self,
    kwargs )
```

Log In Screen.

Displays the log in frame

##### Parameters

<i>*kwargs</i>	A variable argument list to provide extra functionality to the frame
----------------	--

#### 4.5.3.7 showPWPage()

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

Password Management Screen.

Where user can add and view account information

##### Parameters

<i>*args</i>	A variable argument list
--------------	--------------------------

#### 4.5.3.8 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

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

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

- [PPP.py](#)

## Chapter 5

# File Documentation

### 5.1 database.py File Reference

PPP\_database

#### Classes

- class [database.BaseModel](#)  
*Base Model for database connection All other Tables will connect automatically to our database.*
- 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](#) (Id, N, T, U, Hv, Hk)  
*Insert new [Account](#) Instance and [Encrypt](#) Instance.*
- def [database.GetId](#) (id\_)  
*search tables with AcclId*
- 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')

### 5.1.1 Detailed Description

PPP\_database

#### Author

Joseph Lu, luy89

#### Date

20/10/2017

### 5.1.2 Function Documentation

#### 5.1.2.1 CreateTable()

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

Instantiate new empty tables.

Encrypt Table should also be reset when Account is reset

#### 5.1.2.2 Delete()

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

Delete Table Row with ID.

#### Parameters

<i>id</i>	Account ID
-----------	------------

#### 5.1.2.3 DropTables()

```
def database.DropTable ( )
```

Delete tables.

Encrypt Table should also be reset when Account is reset

#### 5.1.2.4 GetId()

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

search tables with AcclId

##### Parameters

<i>Id</i>	Account Id
-----------	------------

#### 5.1.2.5 GetN()

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

Get Table Row with Account name.

##### Parameters

<i>Aname</i>	Account Name
--------------	--------------

#### 5.1.2.6 GetT()

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

Get Table Rows with Account Type.

##### Parameters

<i>Atype</i>	Account type
--------------	--------------

#### 5.1.2.7 Insert()

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

Insert new Account Instance and Encrypt Instance.

**Parameters**

<i>N</i>	Account Name
<i>T</i>	Account Type
<i>U</i>	username
<i>Hv</i>	Hash Value
<i>Hk</i>	Hash Key

**5.1.2.8 UpdateP()**

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

Update Password with ID.

**Parameters**

<i>Aid</i>	Account Id (Encrypted ID)
<i>Hv</i>	new Hash Value

**5.1.2.9 UpdateU()**

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

Update Table Row with ID.

**Parameters**

<i>Aid</i>	Account Id
<i>U</i>	new Username
<i>Hv</i>	new Hash Value

**5.2 Encrypt.py File Reference**

Password Encryption

**Functions**

- def [Encrypt.generKey](#) ()

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

- def `Encrypt.cryptEncode` (key, passwd)

*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.*

### 5.2.1 Detailed Description

#### Password Encryption

##### Author

Shabana Dhayananth

##### Date

October 15, 2017

### 5.2.2 Function Documentation

#### 5.2.2.1 `cryptDecode()`

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

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

##### Parameters

<code>key, encrypted</code>	refer to the key that was used to encript the password and the encrypted password
-----------------------------	---

##### Returns

decrypted password

#### 5.2.2.2 `cryptEncode()`

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

This function uses a key to encrypt an input string.

Fernet uses symmetric encryption on the input key

**Parameters**

<code>key,passw</code>	refer to the key to be used to encrypt and the password to be encrypted
------------------------	---

**Returns**

encrypted password

**5.2.2.3 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)

## 5.3 GenPassword.py File Reference

Generate Random Password

**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.*

### 5.3.1 Detailed Description

Generate Random Password

**Author**

Shabana Dhayananth

**Date**

October 27, 2017



### 5.3.2 Function Documentation

#### 5.3.2.1 `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

#### 5.3.2.2 `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

## 5.4 PPP.py File Reference

The graphical user interface for a password manager.

### Classes

- class [PPP.PPP](#)  
*An ADT that represents the GUI.*

### Variables

- string **PPP.BGC** = "#cccccc"
- tuple **PPP.LARGE** = ("Helvetica", 16)
- string **PPP.BG** = "#383A39"
- string **PPP.FG** = "#A1DBCD"
- **PPP.app** = `PPP()`

### 5.4.1 Detailed Description

The graphical user interface for a password manager.

**Author**

Suhavi Sandhu

**Date**

November 10, 2017

## 5.5 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.*

### 5.5.1 Detailed Description

Check Passwords

**Author**

Suhavi Sandhu

**Date**

November 10, 2017

### 5.5.2 Function Documentation

#### 5.5.2.1 checkLogIn()

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

Checks master password at login.

Verifies that the entered password matches the actual

**Parameters**

<i>entered</i>	The password entered by the user
<i>actual</i>	The real master password

**5.5.2.2 checkMP()**

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

Checks master password at creation.

Verifies that the password meets criteria

**Parameters**

<i>password</i>	The password that is being checked
-----------------	------------------------------------



# Index

- `__init__`
    - PPP::PPP, 10
- `addEntry`
  - PPP::PPP, 11
- `checkLogIn`
  - PWChecking.py, 22
- `checkMP`
  - PPP::PPP, 11
  - PWChecking.py, 23
- `CreateTables`
  - database.py, 16
- `cryptDecode`
  - Encrypt.py, 19
- `cryptEncode`
  - Encrypt.py, 19
- `database.Account`, 7
- `database.BaseModel`, 8
- `database.BaseModel.Meta`, 9
- `database.Encrypt`, 8
- `database.py`, 15
  - `CreateTables`, 16
  - `Delete`, 16
  - `DropTables`, 16
  - `GetId`, 16
  - `GetN`, 17
  - `GetT`, 17
  - `Insert`, 17
  - `UpdateP`, 18
  - `UpdateU`, 18
- `Delete`
  - database.py, 16
- `DropTables`
  - database.py, 16
- `Encrypt.py`, 18
  - `cryptDecode`, 19
  - `cryptEncode`, 19
  - `generKey`, 20
- `genPass`
  - GenPassword.py, 21
- `genPassCrypt`
  - GenPassword.py, 21
- `GenPassword.py`, 20
  - `genPass`, 21
  - `genPassCrypt`, 21
- `generKey`
  - Encrypt.py, 20
- `GetId`
  - database.py, 16
- `GetN`
  - database.py, 17
- `GetT`
  - database.py, 17
- `Insert`
  - database.py, 17
- `matchPassword`
  - PPP::PPP, 11
- PPP.PPP, 9
- PPP.py, 21
- PPP::PPP
  - `__init__`, 10
  - `addEntry`, 11
  - `checkMP`, 11
  - `matchPassword`, 11
  - `showCreateMP`, 12
  - `showEntry`, 12
  - `showLogIn`, 12
  - `showPWPage`, 13
  - `viewDetails`, 13
- PWChecking.py, 22
  - `checkLogIn`, 22
  - `checkMP`, 23
- `showCreateMP`
  - PPP::PPP, 12
- `showEntry`
  - PPP::PPP, 12
- `showLogIn`
  - PPP::PPP, 12
- `showPWPage`
  - PPP::PPP, 13
- `UpdateP`
  - database.py, 18
- `UpdateU`
  - database.py, 18
- `viewDetails`
  - PPP::PPP, 13