Address Book Interface Specifications

TABLE OF CONTENTS

0 Interface Layers

L0:(You are here)
L1: Application
L2: UserInterface, AuditLog
L3: Commands
L4: User
L5: Authorization, Database, AddressEntry, UserEntry, Encryption

1 User

1.1 User [singleton]

1.1.1 Name: getInstance

Syntax

static User getInstance()

Input

none

Output

none

Return

Returns the single instance of User

Description

This function retrieves the single instance of the user to be used, based on the singleton pattern it replaces the public constructor so that only one User object is created.

1.1.2 Name: setUser

Syntax

void setUser(UserEntry entry, String DBKey)

Input

entry

A UserEntry class representing the body of user data

DBKey

Encryption key used for AES

Output

none

Return

Returns the single instance of User

Description

This function retrieves the single instance of the user to be used, based on the singleton pattern it replaces the public constructor so that only one User object is created.

1.1.2 Name: getUserId

Syntax

String getUserId()

Input

none

Output

none

Return

Returns the username of the current user

Description

This function returns the username of the user currently stored in the User class

1.1.3 Name: encrypt

Syntax

String encrypt(String data)

Input

data

An ascii String representation of data to be cryptographically encrypted

Output

none

Return

An ascii String representation of a encrypted form of data

Description

This call uses internal cryptographic techniques to encrypt the data input, using end-to-end encryption for optimal security. This will be used to prepare data for secure and integrous transportation and storage.

1.1.4 Name: decrypt

Syntax

String decrypt(String data)

Input

data

An ascii String representation of encrypted data to be cryptographically decrypted

Output

none

Return

An ascii String representation of a decrypted form of the data

Description

This call uses internal cryptographic techniques to decrypt the data input, using end-to-end encryption for optimal security. This will be used to receive data from secure and integrous transportation and storage, and render it into a form usable by the application.

1.1.1 Name: getAuthorization

Syntax

int getAuthorization()

Input

none

Output

none

Return

Returns the integer representation of the privilege level associated with the current user.

Description

This function retrieves the level of privilege associated with the user. This can be expected to map to one of three values {0:null/no user, 1:user, 2:admin}. This is for RBAC concerning command permissions.

2 Application

2.1 Application [static]

2.1.1 Name: run

Syntax

void run()

Input

none

Output

none

Return

This method doesn’t return anything

Description

This function drives the basic functionality of the address book program. Handles logging, command input / execution, authorization, and authentication, and error reporting.

3 Authorization

3.1 Authorization [static]

3.1.1 Name: verify

Syntax

static boolean verify(Command command)

Input

command

The command to authorize

Output

None

Return

Returns a boolean that is false if authorization fails and true if authorization is successful

Description

This function is used to make sure whoever is using the system is authorized to use a command before it is executed.

4 Audit Log

4.1 AuditLog [singleton]

4.1.1 Name: getInstance

Syntax

static AuditLog getInstance()

Input

none

Output

none

Return

Returns the single instance of AuditLog

Description

This function retrieves the single instance of the audit log to be used, based on the singleton pattern it replaces the public constructor so that only one AuditLog object is created.

4.1.2 Name: logCommand

Syntax

void logCommand(Command command, boolean authorized)

Input

command

The command that was attempted

authorized

Whether the user was authorized to execute the command or not

Output

None

Return

None

Description

This function logs when a user accesses the system through a command and whether the execution of a command was authorized, this is done for non-repudiation. If command code is null, no logging is necessary.

5 Database

5.1 AddressDatabase

5.1.1 Name: get

Syntax

AddressEntry get(String UserId, String recordId, Decryptor decrypt)

Input

id

The id of the user connected to the requested address entry

recordId

The id associated with the requested address entry

decrypt

Function to decrypt information

Output

none

Return

An AddressEntry object populated with information associated with the entry id and user id

Description

This method retrieves address book information for reading.

5.1.2 Name: set

Syntax

void set(String userId, AddressEntry entry, Decryptor decrypt, Encryptor encrypt)

Input

userId

The id of the user connected to the address entry

entry

The address entry information

decrypt

Function to decrypt information

encrypt

Function to encrypt information

Output

none

Return

This method doesn’t return anything

Description

This method writes the information stored in the provided address entry object into the address book of the specified user.

5.1.3 Name: Entry Exists

Syntax

boolean exists(String UserId, String recordId, Decryptor decrypt)

Input

userId

The id of the desired user

recordId

The id associated with the requested address entry

Output

none

Return

If an entry exists in the database

Description

This method checks if a record is available in the database

5.1.4 Name: Database Full

Syntax

boolean isFull(String UserId,Decryptor decrypt)

Input

none

Output

none

Return

If a database is full

Description

This method checks if the number of entries in the db is the max number of entries (256)

5.1.5 Name: Entry Delete

Syntax

boolean deleteRecord(String UserId, String recordId, Decryptor decrypt, Encryptor encrypt)

Input

userId

The id of the desired user

recordId

The id associated with the requested address entry

Output

none

Return

If an entry exists in the database

5.2 UserDatabase

5.2.1 Name: User Get

Syntax

UserEntry get(String userId)

Input

userId

The id of the desired user

Output

none

Return

An UserEntry object populated with information associated with the specified user. Returns null if no such user exists

Description

This method retrieves user information for reading.

5.2.2 Name: User Exists

Syntax

boolean exists(String userId)

Input

userId

The is of the desired user

Output

none

Return

If a user exists in database

Description

This method checks if a user is available in the database

5.2.3 Name: Database Full

Syntax

boolean isFull()

Input

none

Output

none

Return

If a database is full

Description

This method checks if the number of users in the db is the max number of users (7)

5.2.4 Name: User Set

Syntax

void set(UserEntry entry)

Input

entry

The user entry information

Output

none

Return

none

Description

This method writes the information stored in the provided user entry object into the location in the database specified by the username. If no entries for the specified username exist and there are less than max entries. Then one will be created and populated with the information held in the user entry.

6 AddressEntry

1.1 AddressEntry [immutable]

6.1.1 AddressEntry: constructor

Syntax

AddressEntry(String recordID, String SN, String GN, String PEM, String WEM, String PPH, String WPH, String SA, String CITY, String STP, String CTY, String PC)

Input

recordID

The identifying string to access the record, must be unnique

SN

The surname of the person who the address corresponds to

GN

The given name of the person whom the address corresponds to

PEM

The personal email of the person whom the address corresponds to

WEM

The work email of the person whom the address corresponds to

PPH

The personal phone number of the person whom the address corresponds to

WPH

The work phone number of the person whom the address corresponds to

SA

Street Address for the person whom this corresponds to

CITY

The city of the person whom the address corresponds to

STP

State or providence of the person whom the address corresponds to

CTY

The country of the person whom the address corresponds to

PC

The postal code of the person whom the address corresponds to

Output

none

Return

AddressEntry

The constructor will spit out an AddressEntry with the name, address, and phone as fields

Description

This function is the constructor function for the AddressEntry class. It will collect a name, address, and phone number in order to create an AddressEntry object.

6.1.2 Name: Constructor ii

Syntax

AddressEntry(String stringAddressEntry)

Input

stringAddressEntry

AddressEntry in string form

Output

none

Return

AddressEntry object

Description

This function creates a user object from a string that is the same format as generated by AddressEntry.toString()

6.1.3 Name: recordID [Field]

Syntax

final String recordID

Description

This is the field to hold the Id number of the record

6.1.4 Name: Surname [Field]

Syntax

```
final String SN
```

Description

This is the field to hold the surname.

6.1.5 Name: Given Name [Field]

Syntax

```
final String GN
```

Description

This is the field to hold the given name of the person in the record

6.1.6 Name: Personal Email [Field]

Syntax

```
final String pem
```

Description

This is the field to hold the work phone number

6.1.7 Name: Work Email [Field]

Syntax

```
final String WEM
```

Description

This is the field to hold the work email.

6.1.8 Name: Personal Phone Number [Field]

Syntax

```
String String pph
```

Description

This is the field to hold the personal phone number

6.1.9 Name: Work Phone [Field]

Syntax

```
final String wph
```

Description

This is the field to hold the work phone number

6.1.10 Name: Street address [Field]

Syntax

```
final String sa
```

Description

This is the field to hold the Street address

6.1.11 Name: City [Field]

Syntax

```
final String CITY
```

Description

This is the field to hold the city of the address record.

6.1.12 Name: State or Province [Field]

Syntax

```
final String stp
```

Description

This is the field to hold the state or province

6.1.13 Name: Country [Field]

Syntax

```
final String CTY
```

Description

This is the field to hold the country of the person in the address record

6.1.14 Name: Postal Code [Field]

Syntax

```
final String PC
```

Description

This is the field to hold the postal code of the address.

6.1.15 Name: toString

Syntax

```
String toString()
```

Input

None

Output

None

Return

String form of AddressEntry

Description

This function is used to convert the AddressEntry into something storable like
“Bob;Smith;Robert;bobsmith@mail.edu;;8805551212;;;;;;;”

7 UserEntry

7.1 UserEntry

7.1.1 Name: Constructor

Syntax

UserEntry(String userid)

Input

userid

Username of the user

password

Password of the user

Output

Set flag for password to be changed next time this user logs in

Return

UserEntry object

Description

This function generates a new user account object.

7.1.2 Name: Constructor ii

Syntax

UserEntry(String stringUserEntry)

Input

stringUserEntry

UserEntry in string form

Output

none

Return

UserEntry object

Description

This function creates a user object from a string that is the same format as generated by UserEntry.toString()

7.1.3 Name: userid [field]

Syntax

final String userid

Description

The username of the user.

7.1.4 Name: password [field]

Syntax

final String password

Description

The string that connects with the username to allow the user to access their account. This field uses a hash function on the password so that it is encrypted.

7.1.5 Name: toString

Syntax

String toString()

Input

None

Output

None

Return

String form of UserEntry

Description

This function is used to convert the user entry into something storable

7.2 AdminEntry

7.2.1 Name: Constructor

Syntax

AdminEntry(UserEntry entry)

Input

entry

The entry to create admin from

Output

none

Return

AdminEntry object

Description

This function generates an admin account entry in the system.

8 User Interface

8.2 UserInterface

8.2.1 Name: getNextCommand

Syntax

Command getNextCommand()

Input

None

Output

None

Return

Returns the input from the user in string form

Description

This function is used to get input from the user and turn it into a command using the parser

8.2.2 Name: sendResponse

Syntax

void sendResponse(String response)

Input

response

The string to show to the user

Output

None

Return

None

Description

This function is used to send feedback or information to the user

9 Command

9.1 Command [abstract]

9.1.1 Name: *constructor*

Syntax

Command(String input, int authRequirement, String authorizedCode, String unauthorizedCode)

Input

input

Input for the command

authRequirement

The authorization level requirement (0:none,1:user,2:admin)

authorizedCode

String representation of authorization for the log

unauthorizedCode

String representation of unauthorization for the log

Output

None

Return

Returns an object representation of the LoginCommand

Description

Creates a Command object.

9.1.2 Name: getLogCode

Syntax

String getLogCode(boolean isAuthorized)

Input

isAuthorized

Boolean value representing whether or not the user has the authorization to use this Command

Output

none

Return

A two character String code for the Audit Log, as defined in the command line interface.

Description

Accessor for convenient retrieval of Command specific log codes. Must be implemented in subclasses

9.1.3 Name: execute [abstract]

Syntax

String execute() throws CommandException

Input

None

Output

to be determined by subclass implementation

Return

A string to show the user on completion of the command
Throws a CommandException upon failure.

Description

Executes command represented by this object. This class is abstract and thus must be overloaded in all subclass implementations of this class.

9.1.4 Name: validateInput

Syntax

boolean validateInput(String input, int maxSize)

Input

input
Input to be validated
maxSize
Maximum size of the input

Output

None

Return

Boolean value: True if input is of valid format, else false.

Description

Helper method for validating input conforms with requirements outlined in design document:

i) Is no larger than the Maximum size of input for an input
ii) Is alphanumeric
iii) Is nonempty

9.1.4 Name: validateInput

Syntax

boolean validateInput(String input)

Input

input

Input to be validated

Output

None

Return

Boolean value: True if input is of valid format, else false.

Description

Helper method for validating input conforms with requirements outlined in design document:

i) Is alphanumeric

ii) Is nonempty

9.2 Login extends Command

9.2.1 Name: *constructor*

Syntax

Login(String input)

Input

input

Input of command from the user, to be parsed

Output

none

Return

Returns an object representation of the LoginCommand

Description

This constructor takes int input in the form “LIN <userID> <password>” this is parsed by the execute

9.2.2 Name: execute

Syntax

String execute() throws CommandException

Input

None

Return

Returns If the credentials are correct it will return "Login successful". If credentials are not correct it will return "The username or password is invalid".

Description

This function checks if the username and password inputted by the user exist in the credentials database. If the username and password match up and are correct, the user is granted access to their account inside the application.

9.3 Logout

9.3.1 Name: *constructor*

Syntax

Logout()

Input

none

Output

none

Return

Returns an object representation of the Logout command.

9.3.2 Name: execute

Syntax

String execute() throws CommandException

Input

none

Output

none

Return

If successful, String message indicating success otherwise throws a CommandException

Description

This function logs a user off of their account and closes access to the account from the device at that time.

9.4 ChangePassword extends Command

9.4.1 Name: *constructor*

Syntax

ChangePassword(String input)

Input

currentPassword

The current password for the user in the Application

newPassword

The new password to change

Output

none

Return

Returns an object representation of the ChangePassword command.

Description
This function creates an instance of the ChangePassword class.

9.4.2 Name: execute

Syntax
String execute() throws CommandException
Input
none
Output
Password is changed in UserDatabase
Return
String message indicating success
Throws a CommandException if it fails

Description

This function changes the password of the current user. It verifies that the current password of the current user matches the currentPassword. If so, the password is updated to reflect the newPassword.

9.5 AddRecord extends Command

9.5.1 Name: *constructor*

Syntax
AddRecord(String input)
Input
input
Input for the command from the user

Output
none

Return

Returns an object representation of the AddRecord command.

Description

This constructs a class to add a record using the input from the user in the form ADR <recordID> [<field1=value1> <field2=value2> ...]

9.5.2 Name: execute

Syntax

String execute() throws CommandException

Input

none

Output

user's address book is updated

Return

If successful, String message indicating success otherwise throws a CommandException

Description

This function adds a new address entry into the address book associated with the user. It uses the parameters passed into the constructor in the creation of the new entry.

9.6 DeleteRecord extends Command

9.6.1 Name: *constructor*

Syntax

AddRecord(String input)

Input

input

Input for the command from the user

Output

none

Return

Returns an object representation of the AddRecord command.

Description

This constructs a class to remove a record using the input from the user in the form DER <recordID>

9.6.2 Name: execute

Syntax

String execute() throws CommandException

Input

none

Output

none

Return

none

Description

This function is a command that a user can use to delete a record from his/her own address book.

9.7 EditRecord extends AddRecord

9.7.1 Name: execute

Syntax

String execute() throws CommandException

Input

none

Output

none

Return

none

Description

This function is a command that a user can use to edit a record in his/her own address book.

9.8 GetRecord extends Command

9.8.1 Name: execute

Syntax

String execute() throws CommandException

Input

none

Output

none

Return

AddressEntry

The address entry of the person the user is looking for

Description

This function is a command that a user can use to delete a record from his/her own address book.

9.9 ImportDatabase extends Command

9.9.1 Name: execute

Syntax

String execute() throws CommandException

Input

None

Output

user's address book replaced

Return

This method doesn't return anything

Description

This method loads an address book database into the user's account, overwriting any pre-existing database.

9.10 ExportDatabase extends Command

9.10.1 Name: execute

Syntax

String execute() throws CommandException

Input

None

Output

None

Return

This method doesn't return anything

Description

This method writes all of the information stored in the user's address book database into an external file.

9.11 AddUser extends Command

9.11.1 Name: execute

Syntax

String execute() throws CommandException

Input

None

Output

None

Return

None

Description

If parameters don't exist throw CommandException
This method adds a user to the system

9.12 DeleteUser extends Command

9.12.1 Name: execute

Syntax

String execute() throws CommandException

Input

None

Output

None

Return

None

Description

This method removes a user from the system
If user of id or parameter id doesn't exist throws CommandException

9.13 DisplayAuditLog extends Command

9.13.1 Name: execute

Syntax

String execute() throws CommandException

Input

None

Output

Displays the audit log interactively for the user

Return

None

Description

If user is not admin throws AuthException
This method displays the audit log of the system to the admin
All string arguments passed to the constructor are ignored

9.14 Help extends Command

9.14.1 Name: execute

Syntax

String execute() throws CommandException

Input

None

Output

None

Return

String representation of the command line syntax of one or more commands.

Description

Help command which shows the user the syntax necessary to use the system.

9.15 CommandException extends Exception

Description

This exception represents an Exception occuring during the logic of the Command execution.

10 Encryption

10.1 Encryption

10.1.1 Name: BCrypt hash

Syntax

String hashBCrypt(String data)

Input

data

Data that is to be hashed

Output

none

Return

A string that has hashed the data

Description

The hash function will use the Bcrypt algorithm to provide unpredictable hash strings.

10.1.2 Name: SHA256 hash

Syntax

String hashSHA256(String data)

Input

data

Data that is to be hashed

Output

none

Return

A string that has hashed the data

Description

The hash function will use the SHA256 algorithm to provide unpredictable hash strings

10.1.3 Name: BCrypt check hash

Syntax

boolean checkBCrypt(String unhashed, String hashed)

Input

Unhashed

Data to be checked

Hashed

Data that is hashed to be checked against

Output

none

Return

If the two strings are the same

Description

This function compares two bcrypt hashes against each other

10.1.4 Name: encrypt

Syntax

String encrypt(String data, String key)

Input

data

Data that is to be hashed

key

Key used for RSA encryption

Output

none

Return

A string that has hashed the data with the chosen key

Description

This function takes the input data and uses the key to return encrypted data.

10.1.5 Name: decrypt

Syntax

String decrypt(String data, String key)

Input

data

Encrypted data

key

Key used for decryption

Output

none

Return

A string of decrypted data

Description

This function is used to decrypt the data back to its initial state.