CSCI-231 Server-side Development
Quiz 1 Name ______________________________

Please read the entire problem description before writing a solution.

Suppose you have a file named **DatabaseError.js** that contains the following:

```
export default class DatabaseError extends Error {
    constructor(func, message, code) {
        super(message)
        this.func = func   // function
        this.code = code
    }
```

And suppose you have a file named **index.js** that contains the following:

```
import * as db from './database.js'

try {
    db.addUser({ username: 'joe', password: 'test1234!' })
    db.addUser({ username: 'alice', password: 'test1234!' })

    db.setPassword('alice', 'abc123')

    const alice = db.getUser('alice')
    console.log(alice)

    db.deleteUser('joe')

    const arr = db.getAllUsers()
    console.log(arr)
}
catch (err) {
    console.log(`${err.func}: ${err.message} - ${err.code}`)
}
```

As you can see from the source code shown above, **index.js** imports an object as **db** from a module named **database.js** and utilizes the imported object to store, modify, delete, and retrieve objects in the database.

Write the source code for the **database.js** module. Use the code shown above and the descriptions below to understand the functions provided by the module.

The module should satisfy the following requirements.

1. The module stores in memory a collection of objects. Each user object in the collection models a user and has a **username** property that holds a *non-empty string* and a **password** property that holds a *non-empty string*.

2. The module provides a function named **addUser** that takes an object as an argument. If the object passed to the function is not of the proper form that can be stored in the collection (See #1), or has invalid property values, or if there is already an object in the collection with the same **username** as the object passed into the function then the function throws an instance of **DatabaseError**; otherwise the object passed into the function is added to the collection.

3. The module provides a function named **setPassword** that takes 2 strings as arguments. The first string holds a user's username, and the second string holds a new password. If either argument is not a non-empty string or if the username does not correspond to a user in the collection then the function throws an instance of **DatabaseError**; otherwise the **password** property - of the object in the collection whose username matches the username passed into the function - is changed to the password string passed into the function.

4. The module provides a function named **getUser** that takes a string as an argument. The function throws an instance of **DatabaseError** if there does not exist an object in the collection that has a **username** property that is equal to the string passed into the function; otherwise it returns the object in the collection whose **username** property is equal to the string passed into the function.

5. The module provides a function named **deleteUser** that takes a string as an argument. The function throws an instance of **DatabaseError** if there does not exist an object in the collection that has a **username** property that is equal to the string passed into the function; otherwise it deletes from the collection the object whose **username** property is equal to the string passed into the function.

6. The module provides a function named **getAllUsers** that returns an array containing all of the objects in the collection.

When throwing an instance of **DatabaseError** use reasonable and useful values for **func**, **message**, and **code**.