**WriterSpot**

**Summary**

A text-based program in Java for writing and sharing stories, poems and other written works which can be used by anyone with basic knowledge of computers. The program will help the user in keeping their written work safe and available everywhere and every time and can also be made public if they wish thus would eliminate the need of a paper-based products and making it easier to get help on it from others. It also consists of a game StoryTeller in which users can add a sentence of their own to an ongoing story started by the admin. The program will be using file handling to handle data.

**Use Case Analysis**

**The program will start with the main menu screen:**

Welcome to WriterSpot. A place for people who want to write and share with others.

Please choose from following options:

1. Sign Up

2. Sign In

3. Exit

Choice:

**The sign-up screen asks for username and password**

Sign Up

Enter a Username:

Password:

**After Checking if username doesn’t already exist, it creates the account and asks for logging in:**

Writer created successfully. You will have to put login credentials to sign in.

--------------------------------------------------

Welcome to the login section. Please enter following

Username:

Password:

**After the writer logs in, they have following options:**

1. View public entries **(Shows public entries if exist)**

2. Create new entry **(Asks for new entry content and creates if doesn’t exist)**

3. List all entries **(List entries belonging to writer)**

4. Edit old entry **(Option to edit an existing entry)**

5. Delete old entry **(Option to delete an existing entry)**

6. Open StoryTeller game **(Runs the StoryTeller game)**

7. Save work and Log Out

8. Exit

Choice:

**The StoryTeller game shows all the existing stories and asks the writer to choose the story they wish to play with.**

**Data Design**

As the program will be using file handling, several files will be used to store data.

**Writers.dat**: Stores username, password and entries of each writer in a Vector of Writer objects

**stories.dat**: Stores the name of each story created by the admin in a Vector of String objects

**[storyName].dat**: Each story has its own file, saves a Vectors of username and storyline in another Vector.

Each Writer Object contains Entry objects if there are any entries.

Algorithm

***\*\*Goals\*\****

Create a program for user to write anything and see what others wrote.

***\*\*Input\*\****

Opens a file, Writers.dat , stores the data in a vector writers.

Opens a file, stories.dat , stores the data in a vector stories.

***\*\*Output\*\****

Opens the file Writers.dat, outputs the vector writers in it.

***\*\*Steps\*\****

***- \*\*\*Program Algorithm\*\*\****

1. \*Open\* the \*input\* file\*Writers.dat\* , reads data into vectors \*writers\*.

2. Show \*mainMenu()\*.

3. If \*choice\* is sign up, ask for \*username\* and \*password\* and create if doesn’t exist already.

4. If \*choice\* is sign in, ask for \*username\* and \*password\*. Check for \*login\* credentials. Open \*writerMenu\* if \*writer\*. Show \*adminMenu\* if credentials match to admin.

5. If \*choice\* is exit, \*confirm\* exit and end program if \*yes\*.

***- \*\*\*Entry.java\*\*\*:***

\*Entry(String userEntry, boolean isPublic)\*:

\*construct\* object.

\*getEntry()\*:

return \*object.entry\* as String.

\*setEntry(String userEntry)\*:

set \*object.entry\* to userEntry.

\*getPublicStatus()\*:

return \*object.isPublic\* as boolean.

\*setPublicStatus(boolean isPublic)\*:

set \*object.isPublic\* to \*isPublic\*

***- \*\*Writer.java\*\*\*:***

\*Writer(String username, String password):

\*construct\* object.

\*getUsername()\*:

return \*object.username\* as String.

\*setUsername(String username)\*:

set \*object.username\* to username.

\*getPassword()\*:

return \*object.password\* as String.

\*setPassword(String password)\*:

set \*object.password\* to \*password\*.

\*getEntries()\*:

return \*object.entries\* as Vector.

\*newEntry(String userEntry, boolean isPublic)\*:

a. Check if \*userEntry\* already exists.

b. if \*true\*, print \*already exists\*.

c. if \*false\*, make new \*Entry\* object and add to \*object.entries\*.

\*delEntry()\*:

a. if size of \*entries\* is 0, print "no entries\*.

b. else, call \*object.listEntries\* to print \*entries\*.

c. Ask \*entry\* number to delete\*

d. Check if input is valid, delete if \*true\*.

e. If \*invalid input\* print invalid input.

\*editEntry()\*:

a. if size of \*entries\* is 0, print "no entries\*.

b. else, call \*object.listEntries\* to print \*entries\*.

c. Ask \*entry\* number to delete\*

d. Check if input is valid, ask for \*editedEntry\* if \*true\*., set \*object.entry\* to \*editedEntry\*.

e. If \*invalid input\* print invalid input.

\*listEntries()\*:

a. if size of \*entries\* is 0, print \*no entries\*.

b. run \*for\* loop to traverse through \*object.entries\*

c. Get \*entry\* and \*status\* from each object in \*object.entries\*.

d. Print \*entry\* and \*status\*.

- \*\*\*Menu.java\*\*\*:

Open \*Writers.dat\* and read vector and store in \*writers\*.

call \*mainMenu\* to run program.

UML

