Software Requirements Document for Family Album

Functional Requirements

Developed By:

Group #2

James Blake

Sunny Zeng

Kai Wu

October 7, 2010

1. **About this document**

This document describes the functional requirements for the software for a family album.

1. **Functional Requirements**

Each functional requirement is given in the following format:

*Index:*

*Name:*

*Purpose:*

*Input parameters:*

*Action:*

*Output parameters:*

*Exceptions:*

*Remarks:*

*Cross-references:*

*Index* refers to a unique index assigned to this functional requirement. It will be used for cross-referencing this requirement from other requirements and also from other documents such as the design document.

*Name* is a descriptive name given to this functional requirement. This name does not need to be unique.

*Purpose* is a short description (in a line or two) of the function. It is used to quickly understand the functionality and is also used to search the required function when all the functionalities are browsed through.

*Input parameters* refers to a set of parameters that the given function accepts as input. These parameters are required in order to design and implement the current function. No type information will be included for parameters in this document.

*Output parameters* refers to a set of parameters that are output/exhibit by the current function, when implemented. No type information will be included for parameters in this document.

*Action* refers to a set of tasks or activities that must be performed in order to satisfy this requirement. These tasks/activities are listed in no particular order. The design will take care of sequencing the tasks.

*Exceptions* refers to a set of conditions, each of which indicates a situation in which the function will stop. Notice that this column only lists a set of exceptions that might occur but does not suggest any action that must be taken when the exceptions occur. These actions will be included in the design document.

*Remarks* includes a set of comments that explain more about the functionality. It also describes hints to the designer and implementer that are suggested by the requirements analyst.

*Cross-references* refers to a set of other functional requirements that are related to the current function

* 1. **Functional Requirements for the Family Album**

*Index:*

*Name:*

*Purpose:*

*Input Parameters:*

*Action:*

*Output parameters:*

*Exceptions:*

*Remarks:*

*Cross-References:*

*Index:* FA.0

*Name:* InitializeFamilyAlbum

*Purpose:* To initialize a family album with empty set of entries except for the name of the album

*Input Parameters:* Album Name

*Action:* Initialize the family album with all fields empty except for the name of the album

*Output parameters:* None

*Exceptions:* Album Name is already taken.

*Remarks:* This function will be used when a new family album is created

*Cross-References:*None

*Index:* FA.1

*Name:* SelectFamilyAlbum

*Purpose:* To select which family album is desired

*Input Parameters:* Album Name

*Action:* Activates the family album

*Output parameters:* None

*Exceptions:* Album Name is not in database

*Remarks*: This function activates the family album that was entered. Brings needed data into memory.

*Cross-References:* None

**2.2 Functional Requirements for the \_\_\_\_\_\_\_\_\_\_\_**

*Index:* FP.0

*Name:* AddProfile

*Purpose:* To add a profile into the album.

*Input Parameters:* last name, first name, birth date, death date, description

*Action:* Ensure that no profile contains same last name, first name and birth date exists in the album.

Ensure that birth date is in the format of MM/DD/YYYY.

Ensure that death date is either empty or in the format of MM/DD/YYYY.

Ensure that description is in the range of 0 to 256 characters.

Update the family album by adding a profile with all the input entries into the family album.

*Output parameters:* None

*Exceptions:* Either last name, first name, or birth date is empty.

The combination of last name, first name, and birth date already exists in the album.

Birth date is not in the format of MM/DD/YYYY.

Death date is not empty but not in the format of MM/DD/YYYY.

Description is out of 0 to 256 characters range.

Album is full.

*Remarks:* The album may be sorted on last name, first name, or birth date so that searching for profile would be easier.

Death date and description are not required to add a profile.

*Cross-References*: None

*Index:* FP.1

*Name:* ModifyProfile

*Purpose:* To modify a profile in the family album*.*

*Input Parameters:* last name, first name, birth date, death date, description

*Action:* Ensure that the profile contains same last name, first name and birth date exists in the album.

Ensure that birth date is in the format of MM/DD/YYYY.

Ensure that death date is either empty or in the format of MM/DD/YYYY.

Ensure that description is in the range of 0 to 256 characters.

Update the profile by overwriting the new last name, first name, birth date, death date, and/or description into the profile.

*Output parameters:* None

*Exceptions:* Either last name, first name, or birth date is empty.

Birth date is not in the format of MM/DD/YYYY.

Death date is not empty but not in the format of MM/DD/YYYY.

Description is out of 0 to 256 characters range.

*Remarks:* The album may be sorted on last name, first name, or birth date so that searching for profile would be easier.

Death date and description are not required to modify a profile. If are not provided, those fields are assumed to be empty.

If death date is updated, this function may make use of FP.2.

If description is updated, this function may make use of FP.3.

*Cross-References:* FP.2, FP.3

*Index:* FP.2

*Name:* AddDeathDate

*Purpose:* To add the death date of this people into the profile

*Input Parameters: death date*

*Action:* Ensure that the death date input is in the format of MM/DD/YYYY

*Update the profile by adding the death date*

*Output parameters:* None

*Exceptions:* the format of the death date does not follow MM/DD/YYYY

*Remarks:* This function will be used when the user of this software choose to enter the death date of this people

*Cross-References:* None

*Index:* FP.3

*Name:* AddDescription

*Purpose:* To add the description of this people into the profile

*Input Parameters: description*

*Action:* Ensure that the description of the person is 256 or few characters

*Ensure that the description of the person can’t be null.*

*Update the profile by adding the description*

*Output parameters:* None

*Exceptions:* the size of the description is 0 or bigger than 256 characters

*Remarks:* This function will be used when the user of this software choose to enter the death date of this people

*Cross-References:* None

*Index:* FP.4

*Name:* SearchAlbum

*Purpose:* To search a specific album

*Input Parameters: album name*

*Action:*

Get a specific album by searching album name

*Output parameters:* Album

*Exceptions:* no such album, which has the specific attribute.

*Remarks:* This function will be used when the user of this software find an album

*Cross-References:* None

*Index:* FP.5

*Name:* SearchProfile

*Purpose:* To search a specific profile from the family album

*Input Parameters: last name or first name or birth date*

*Action: Ensure that the format of the birth date is MM/DD/YYYY if it is not null*

Get a specific profile by searching the last name, first name or birth date one by one.

*Output parameters:* profile

*Exceptions:* no such profile, which has the specific attributes.

*Remarks:* This function will be used when the user of this software desire to search a specific profile

*Cross-References:* None

*Index:* JB.2

*Name:* DeleteProfile

*Purpose:* To delete an individual profile from the family album

*Input Parameters:* Last name, first name, birth date

*Action:* Ensure that a person with that exact last name, exact first name, and exact birth date exists in the database. Update the family album by removing all data associated with this profile.

*Output parameters:* None

*Exceptions:* No person matching with that exact last name, exact first

name, and exact birth date exists in the database

*Remarks:* None

*Cross-References:* None

*Index:* JB.3

*Name:* AddPhoto

*Purpose:* Adds a .jpg file to profile

*Input Parameters:* Last name, first name, birth date, photo

*Action:* Ensure that a person with that exact last name, exact first name, and exact birth date exists in the database.

Add photo to that profile’s list of photos

*Output parameters:* None

*Exceptions:* No person matching with that exact last name, exact first

name, and exact birth date exists in the database.

*Remarks:* This function will add a photo to a profile

*Cross-References:* None

*Index:* JB.4

*Name:* AddTextDoc

*Purpose:* Adds a .txt file to profile

*Input Parameters:* Last name, first name, birth date, text document

*Action:* Ensure that a person with that exact last name, exact first name, and exact birth date exists in the database.

Add text document to that profile’s list of text documents

*Output parameters:* None

*Exceptions:* No person matching with that exact last name, exact first

name, and exact birth date exists in the database.

*Remarks:* This function will add a text document to a profile

*Cross-References:* None

*Index:* JB.5

*Name:* AddGraphics

*Purpose:* Adds a .png file to profile

*Input Parameters:* Last name, first name, birth date, graphic

*Action:* Ensure that a person with that exact last name, exact first name, and exact birth date exists in the database.

Add graphics to the profile’s list of graphics

*Output parameters:* None

*Exceptions:* No person matching with that exact last name, exact first

name, and exact birth date exists in the database.

*Remarks:* This function will add a graphic to a profile

*Cross-References:* None

*Index:* JB.6

*Name:* ViewPhoto

*Purpose:* Opens a .jpg file

*Input Parameters:* Photo

*Action:* Ensure that the photo exists.

Open photo.

*Output parameters:* None

*Exceptions:* The photo does not exist

*Remarks:* This function will open a photo

*Cross-References:* None

*Index:* JB.7

*Name:* ViewTextDoc

*Purpose:* Opens a .txt file

*Input Parameters:* Text Document

*Action:* Ensure that the text document exists.

Open text document.

*Output parameters:* None

*Exceptions:* The text document does not exist

*Remarks:* This function will open a text document

*Cross-References:* None

*Index:* JB.8

*Name:* ViewGraphics

*Purpose:* Opens a .png file

*Input Parameters:* Graphic

*Action:* Ensure that the graphic exists.

Open graphic.

*Output parameters:* None

*Exceptions:* The graphic does not exist

*Remarks:* This function will open a graphic

*Cross-References:* None

*Index:* JB.9

*Name:* DeletePhoto

*Purpose:* Deletes a .jpg file from a profile

*Input Parameters:* Photo

*Action:* Ensure that the photo exists.

Delete photo from that profile’s list of photos

*Output parameters:* None

*Exceptions:* Photo does not exist

*Remarks:* This function will delete a photo from a profile

*Cross-References:* None

*Index:* JB.10

*Name:* DeleteText

*Purpose:* Deletes a .txt file from a profile

*Input Parameters:* Text Document

*Action:* Ensure that the text document exists.

Delete text document from that profile’s list of text document

*Output parameters:* None

*Exceptions:* Text document does not exist

*Remarks:* This function will delete a text document from a profile

*Cross-References:* None

*Index:* JB.11

*Name:* DeleteGraphic

*Purpose:* Deletes a .png file from a profile

*Input Parameters:* Graphic

*Action:* Ensure that the graphic exists.

Delete graphic from that profile’s list of graphics

*Output parameters:* None

*Exceptions:* Graphic does not exist

*Remarks:* This function will delete a graphic from a profile

*Cross-References:* None

*Index:* JB.12

*Name:* SetParent

*Purpose:* Updates a profile with a new parent

*Input Parameters:* Child Last Name, Child First Name, Child Birth Date, Parent Last Name, Parent First Name, Child Birth Date

*Action:* Ensure that a person with that exact child last name, exact child first name, and exact child birth date exists in the database.

Ensure that a person with that exact parent last name, exact parent first name, and exact parent birth date exists in the database.

Update child’s profile with parent information

Update parent’s profile with child information

*Output parameters:* None

*Exceptions:* Parent or child is not in database.

Child already has two parents

*Remarks:* This function will update both the child’s and parent’s profile respectively.

*Cross-References:* None

*Index:* JB.13

*Name:* RemoveParent

*Purpose:* Updates a profile to remove child-parent link

*Input Parameters:* Child Last Name, Child First Name, Child Birth Date, Parent Last Name, Parent First Name, Child Birth Date

*Action:* Ensure that a person with that exact child last name, exact child first name, and exact child birth date exists in the database.

Ensure that a person with that exact parent last name, exact parent first name, and exact parent birth date exists in the database.

Update child’s profile with parent information

Update parent’s profile with child information

*Output parameters:* None

*Exceptions:* Parent or child is not in database.

*Remarks:* This function will update both the child’s and parent’s profile

respectively.

*Cross-References:* None