

Picture Publisher

Group 15

Fredrik Lundberg flundbe@kth.se

Jakob Nordlander jakobno@kth.se

Dan Wester dweste@kth.se

Per Wiklander pwik@kth.se

Design Document	2
1. Introduction	2
1.1 The purpose and scope of this document	2
1.2 Intended audience	2
1.3 Version	2
1.4 References to other documents	2
1.5 Important Terms and Acronyms	2
1.6 Abstract	4
2. System Overview	5
2.1 General Description	5
2.2 Overall Architecture Description	5
2.3 Detailed Architecture	6
3. Design considerations	10
3.1 Assumptions and Dependencies	10
3.2 General Constraints	11
4. Graphical User Interface	12
4.1. An Overview of the system from a user’s perspective	12
4.2. The GUI forms and functional requirements associated with them	13
4.3. Names of controls and fields, methods/procedures and triggers for each form	27
5. Design Details	32
5.1 CRC Cards	32
5.2 Class Diagram	38
5.4 Interaction Diagrams	39
5.5 Detailed Design	43
5.6 Package Diagram	61
5.7 References to RD	62
6. Functional Test Cases	63

Design Document

1. Introduction

1.1 The purpose and scope of this document

This document is intended to be used in the development of the “Picture Publisher” software. After reading this document it should be possible to plan the project and estimate the workload. It should help in structuring classes and methods and databases and in how to connect all parts of the system.

1.2 Intended audience

- The project team members
- The customer (i.e. At this moment ourselves but if anyone decides to purchase the system they become the customer)
- The project supervisor

1.3 Version

Version 1.0

1.4 References to other documents

Requirements Document version 1.1

1.5 Important Terms and Acronyms

A

Album

A collection of pictures stored in the file system

Aperture

A hole or an opening through which light is admitted. (wikipedia)

C

Client

The application used to publish pictures.

Color space

The way the colors of the pictures is saved, example on different color spaces is Adobe RGB and sRGB

Compression

Data compression on digital images. (wikipedia)

D

Database

A collection of records stored in a computer in a systematic way, so that a computer program can consult it to answer questions.

E

EJB3 (Enterprise Java Bean 3.0)

A managed, server-sided component for modular construction of enterprise applications

Exposure

The total amount of light allowed to fall on the sensor during the taking of a photograph. (wikipedia)

F

FNumber

The maximum usable aperture of a lens (wikipedia)

FocalLength

The distance to the focal point

K

Keywords

A type of metadata involving the association of descriptors with objects. Here: a collection of short strings describing a picture's content (e.g. "Eiffel Tower", "car sun beach", "John's family" etc.)

M

Manufacturer

The manufacturer of the camera used to take the picture

Metadata

Information about the content of the specified data (e.g. the date, camera type, camera objective or subject of an image file)

Metering mode

The way in which a camera determines the correct exposure. (wikipedia)

Model

The model of the camera

O

Orientation

Specification of how the picture is turned

P

PictureID

A unique number for the picture

Pixel

A single point in a graphic image

Publisher

A user who uploads pictures to his account using the client (e.g. a photographer sharing pictures of cars from a car show)

R

Resolution

The level of detail of an image

T

Tag

The expression for attaching a keyword to a picture

Thumbnail

A tiny version of a picture used as an icon.

Tomcat Web Server

Tomcat Web Server is a system that allows java applications to interact with users through web pages.

U

User

A user is anyone using the system, either a viewer or a publisher (e.g. someone viewing his friend's holiday pictures or someone publishing those holiday pictures)

V

View

Area containing information or items that can be moved.

Viewer

A user who views the pictures uploaded by a publisher (e.g. someone viewing his friend's holiday pictures)

1.6 Abstract

This document describes the expected way to carry out the implementation of the Picture Publisher, including Graphical User Interface, Class Diagram and detailed description of method and Java-classes that will be implemented. This document will also give an overall architecture description as well as a detailed view of the architecture with the data flows and control flows at various levels of the design.

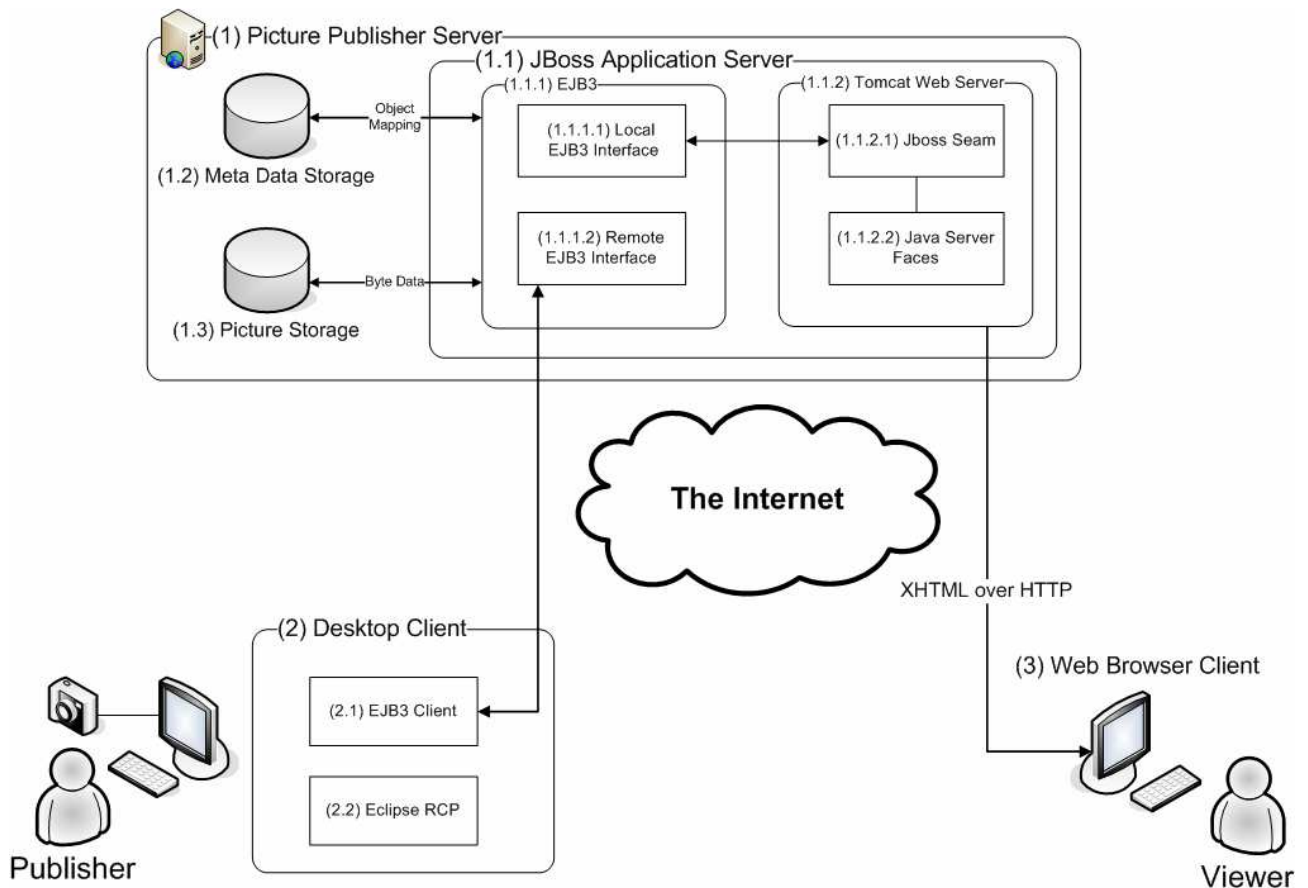
2. System Overview

2.1 General Description

Picture publisher is a system for storing and sharing pictures by uploading them to a central server. This is done by using a client installed on the user's computer. To view the pictures, the only thing needed is a standard web browser. Since locating a specific picture through its filename is quite hard, we intend to make use of keywords and metadata. The metadata contains information about the picture, the keywords describe the picture. Metadata is added by publishers, keywords can be added by both publishers or by viewers.

The design is supposed to be oriented toward usability, and since we have a limited range of functions this should be quite straight forward. I.e. we do not have to make a system of menus and sub menus. Icons will be used as far as possible.

2.2 Overall Architecture Description



- **A client for uploading pictures**

The client allows the user to upload one or more pictures to the system's server. It will be possible to download exact copies of these pictures and this means that the system can be used as an alternative storage unit. The

pictures can be tagged to make it easier for the users to find them.

- **The server**

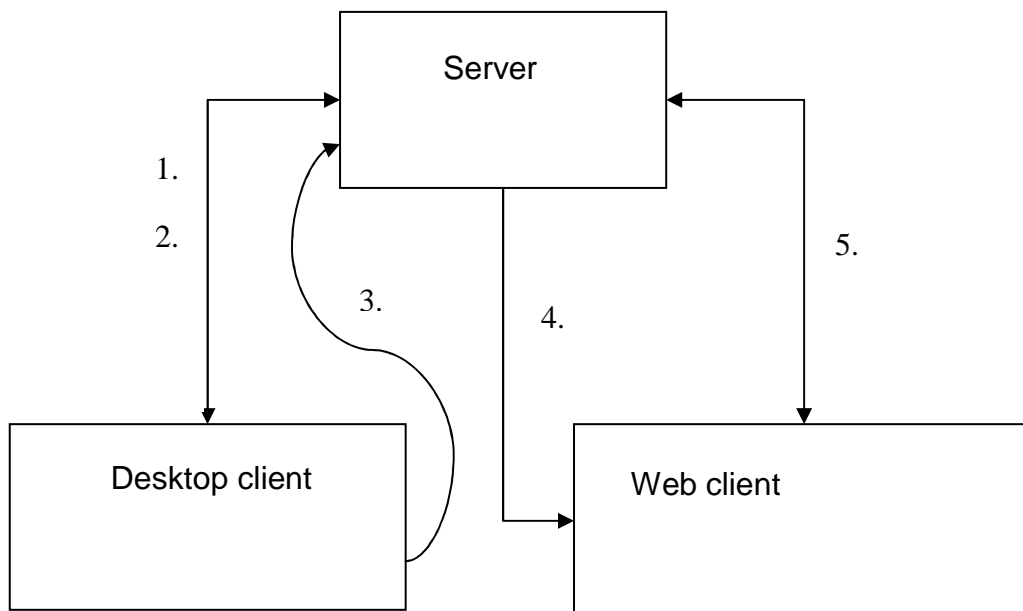
The server is where the pictures and their metadata are stored. The server communicates with the desktop client and serves images to the web clients using a web server.

- **The ability to view pictures given an Internet connection and a browser**

The system allows users to view pictures by using a web browser, it also makes it possible for people who view the pictures to tag them.

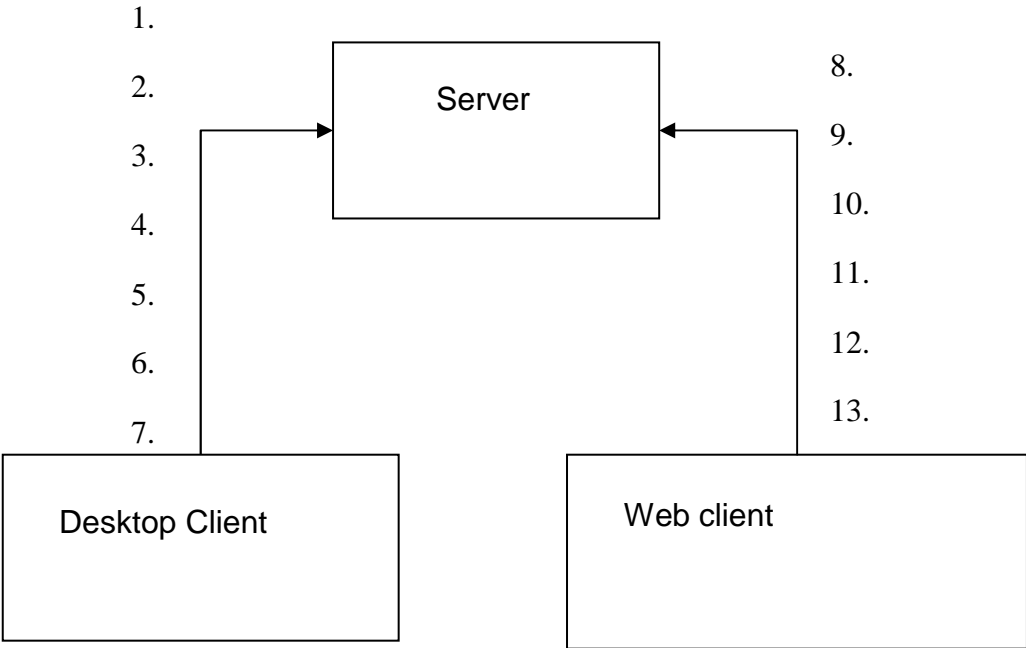
2.3 Detailed Architecture

Dataflow overview



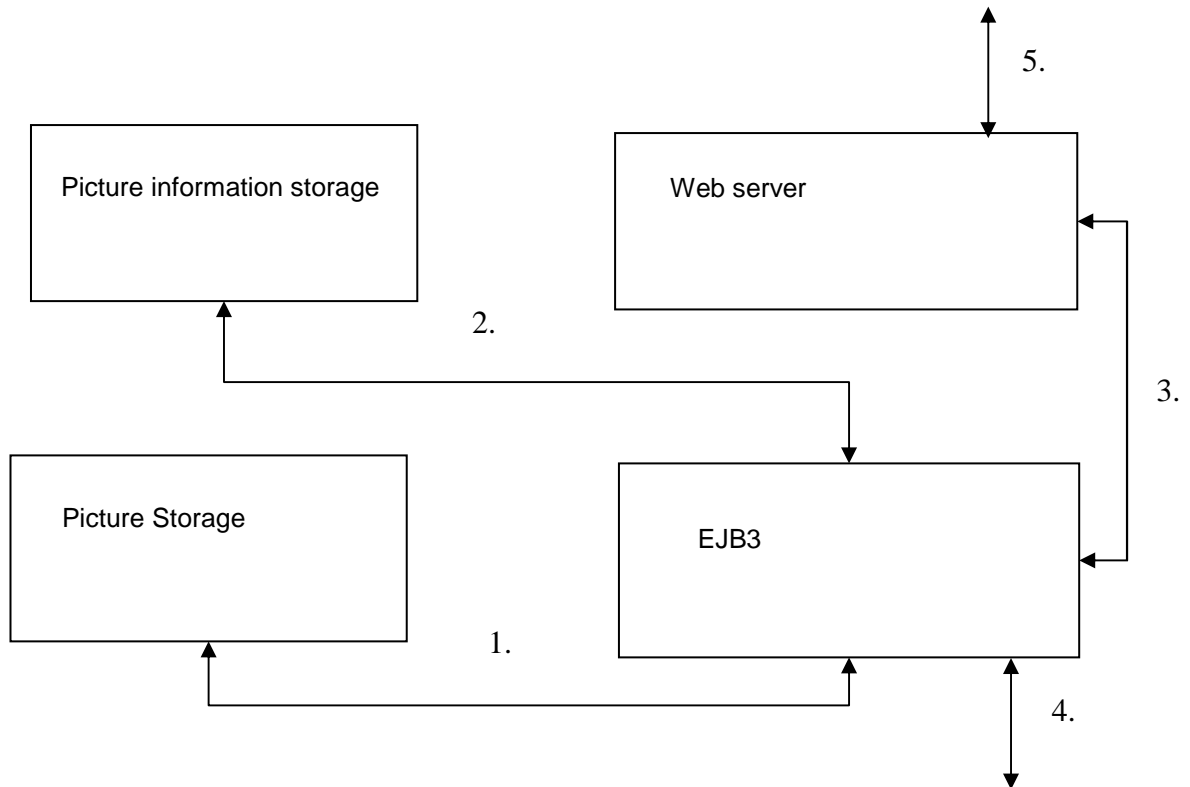
1. Pictures can be uploaded and downloaded to/from the server
2. Metadata and keywords can be uploaded and downloaded to/from the server
3. The desktop client can send login information to the server to verify who is uploading data
4. The server can send the requested pictures to the web client
5. Keywords can be viewed and added to a picture

Control flow overview



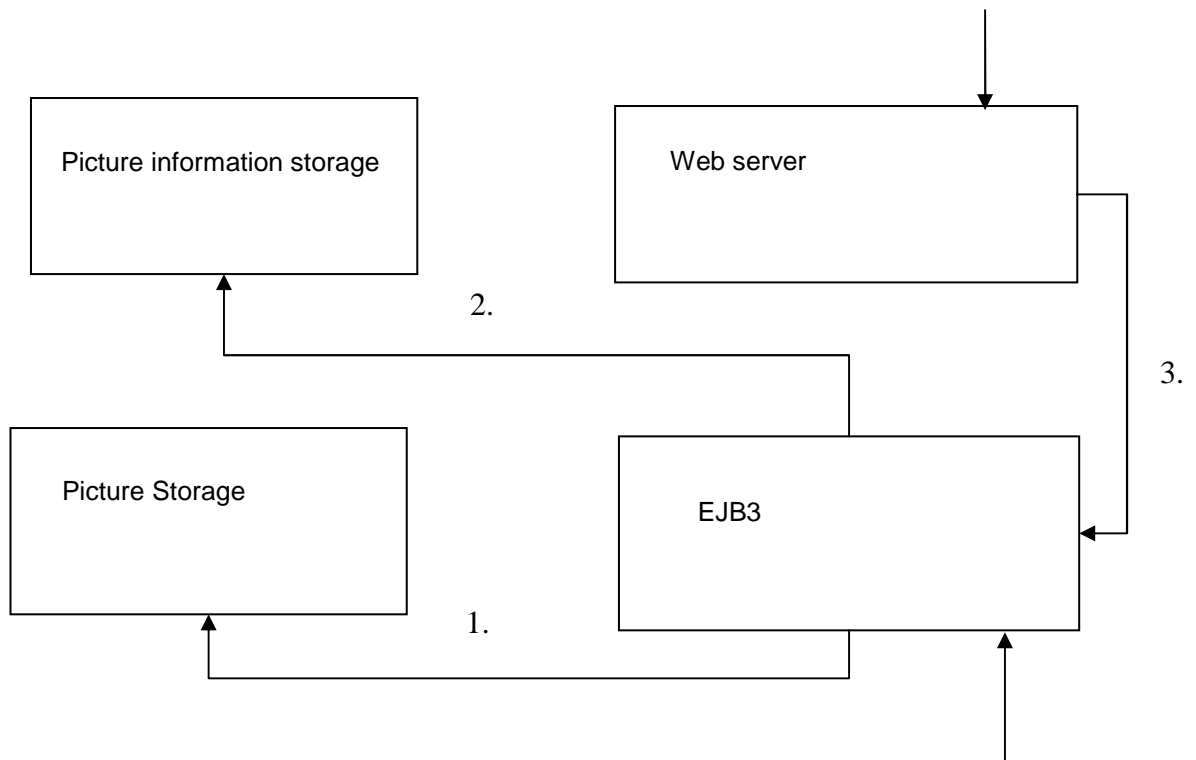
- 1. Access restrictions
- 2. Removing a picture
- 3. Download restrictions
- 4. Login information
- 5. Downloading exact copies
- 6. Adding metadata
- 7. Sorting pictures into albums
- 8. Account creation
- 9. Searching
- 10. Removing accounts
- 11. Downloading exact copies
- 12. Adding keywords
- 13. Viewing pictures through a standard web browser

Dataflow in the server



1. Pictures are sent between the EJB3 application and the picture storage
2. Metadata and keywords is sent between the EJB3 application and the picture information storage
3. Pictures and keywords are sent between the web server and the EJB3 application
4. Data is sent and received to/from the desktop client
5. Data is sent and received to/from the web client

Control flow in the server



1. The EJB3 application can tell the picture storage to store, remove or return a picture.
2. The EJB3 application can tell the metadata storage to store, remove or return metadata.
3. The web server can tell the EJB3 application to return a picture. Furthermore it can tell the EJB3 application to store or return metadata

3. Design considerations

3.1 Assumptions and Dependencies

- Related software and hardware
 - The server part of the system should run on any platform that supports running a Java JVM of version 1.5 or higher. We will test the server on Windows XP, Windows Vista and Gentoo Linux.

 - The desktop client will run on any platform that supports running Eclipse RCP applications. At the time of writing, the supported platforms are (according to the Eclipse website)
 - Microsoft Windows XP, x86-32, Win32
 - Microsoft Windows Vista, x86-32, Win32
 - Red Hat Enterprise Linux 4.0 update 2, x86-32, GTK
 - Red Hat Enterprise Linux 4.0 update 2, x86-64, GTK
 - SUSE Linux Enterprise Server 10, x86-32, GTK
 - Sun Solaris 10, SPARC, GTK
 - IBM AIX 5.3, Power, Motif 2.1
 - Red Hat Enterprise Linux 4.0 update 2, Power, GTK
 - SUSE Linux Enterprise Server 10, Power, GTK
 - Apple Mac OS X 10.4, Universal, Carbon

 - We will test the desktop client software on Windows XP, Windows Vista and Apple Mac OS X 10.4.

 - The web client requires a modern standards compliant web browser. Examples of such browsers are:
 - Mozilla Firefox 1.5 - 2
 - Microsoft Internet Explorer 6 – 7
 - Safari
 - Opera 9

We will test the web client in Mozilla Firefox and Internet Explorer.

- There are no special hardware requirements for the different parts of the system other than those imposed by the operating system under which the system is run. One thing to keep in mind is that the amount of storage space available to the server software will decide the amount of pictures that can be stored.

If the server is run on another machine than the client (which is typically the case) there has to be a network connection between those two machines.

- End-user characteristics
 - Our users have some computer experience. They are generally interested in photography and regularly use software like Adobe Photoshop. They will use the software both at home and at work, school or when traveling.
- Possible and/or probable changes in functionality
 - We have started to feel that the requirement “Adding another account to use in the client” (page 9 in the RD) is unnecessary and will most likely not implement that.

3.2 General Constraints

- **Picture access time for viewer** (page 17 in the RD)
Design impact: To cut down on server processing time the different versions of the image files are created in the client before everything is uploaded to the server. (see *section 5.5: ImportAction* in this document).
- **Specified amount of space** (page 18 in the RD)
Design impact: Before a picture is added on the server the system checks that the amount of available storage space is sufficient (see *section 5.5: PictureService* in this document).
- **The Picture Publisher shall be in compliance with PUL, personuppgiftslagen** (page 19 in the RD)
Design impact: Personal information about the users of the system is only stored when needed and after getting permission from the user. No personal information about users is shown to other users of the system.

4. Graphical User Interface

4.1. An Overview of the system from a user's perspective

The Picture Publisher Client

When starting the program the main window appears. This window holds all information for handling the program. This includes folder views with views for thumbnails of the pictures for both the home folder and the Picture Publisher folder (server folder). Below the album view is an input field for search strings; this is used for searching for pictures.

When the user clicks file -> login a window appears and the user is prompted for login name and password. If the login fails, a red text indicates that the user has entered an incorrect login name or password. Login is required for uploading or downloading pictures and viewing pictures located on the server.

When a thumbnail is selected (clicked on) it appears in a larger format (640x480) and a metadata table together with a keyword table appears at the bottom of the interface. These tables can be altered before uploading to the server/downloading to the home folder.

Uploading and downloading is done by selecting one or more pictures from the thumbnail view and then pressing the upload or download buttons or dragging thumbnails between the thumbnail views.

Adding and removing albums in the server folder is done by right-clicking with the mouse. This also makes it possible to set pictures and albums as public or private. Setting download restrictions for pictures in the thumbnail view is done by right-clicking with the mouse.

Help is provided by clicking the "help" button in the file menu.

The Picture Publisher Web Page

When accessing the website the user can log in, search for pictures, download the Picture Publisher Client and create an account by signing up. The Search function can separate keywords attached to pictures by the Publisher or by the Viewers. When a successful login is executed the webpage displays the text "You are logged in". If the login was unsuccessful a red text indicates "login incorrect". If the user is logged in he can remove the account by clicking "remove account". This action will take him to a page that informs of the consequences and can proceed or cancel.

If the user presses Sign Up a page is displayed that prompts the user for an alias, password and e-mail. If any form does not hold the correct data the red text "incorrect sign up" is displayed.

When the user performs a search the results are displayed in thumbnails and clicking a picture makes it appear in a larger format and the choice of tagging it is activated.

The user who is unfamiliar with the webpage layout and functions can get assistance by clicking help which displays the help file.

4.2. The GUI forms and functional requirements associated with them

The client login window [form 1]

Functional requirements

none



The image shows a screenshot of a Windows-style application window titled "Picture Publisher Client". The window contains a login form with the following elements:

- Title: "Picture Publisher Client Login"
- Input field: "Login name" followed by a text input box.
- Input field: "Password" followed by a text input box.
- Button: A green button labeled "Login" with a black arrow pointing to the right.

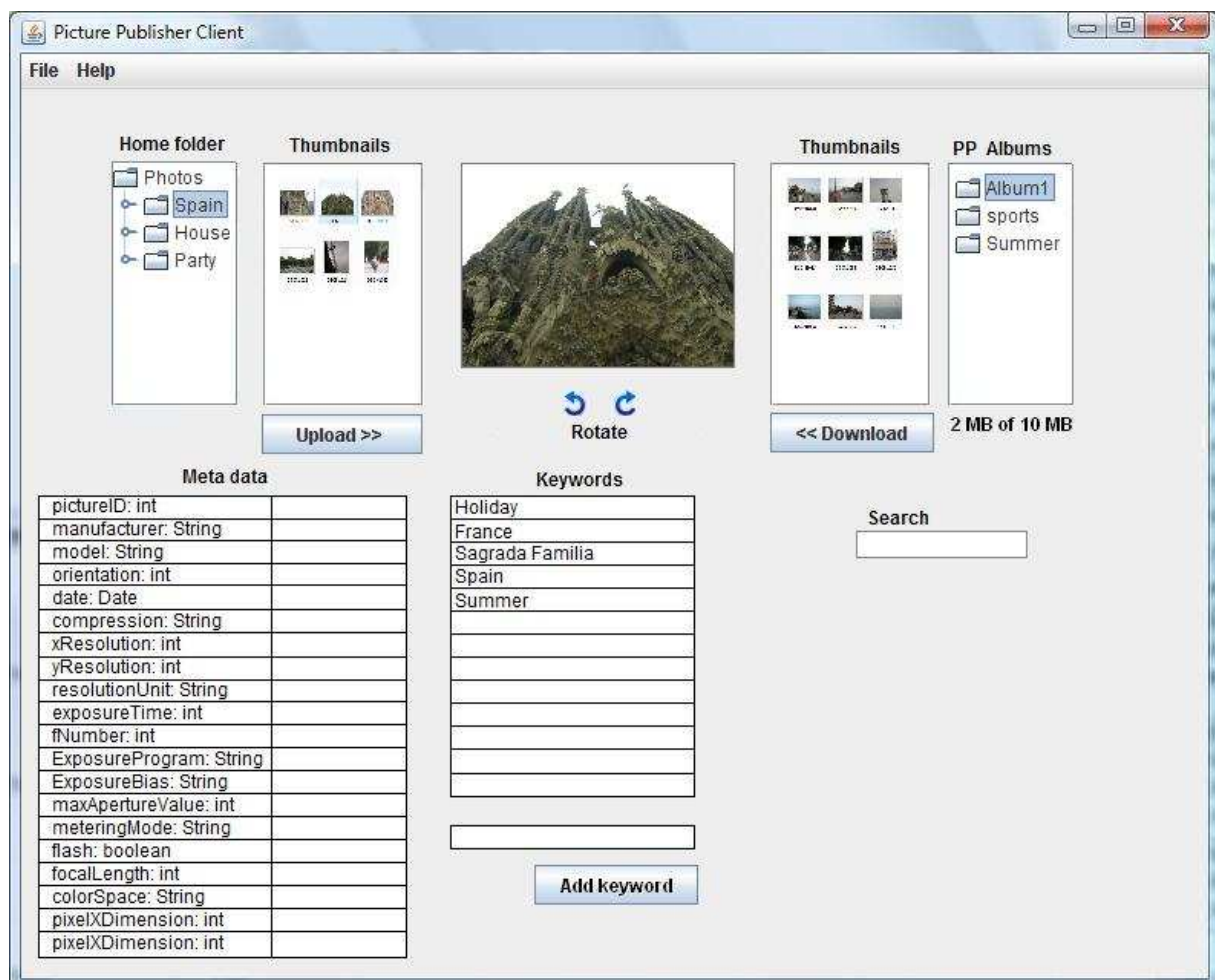
The "incorrect login" window [form 2]



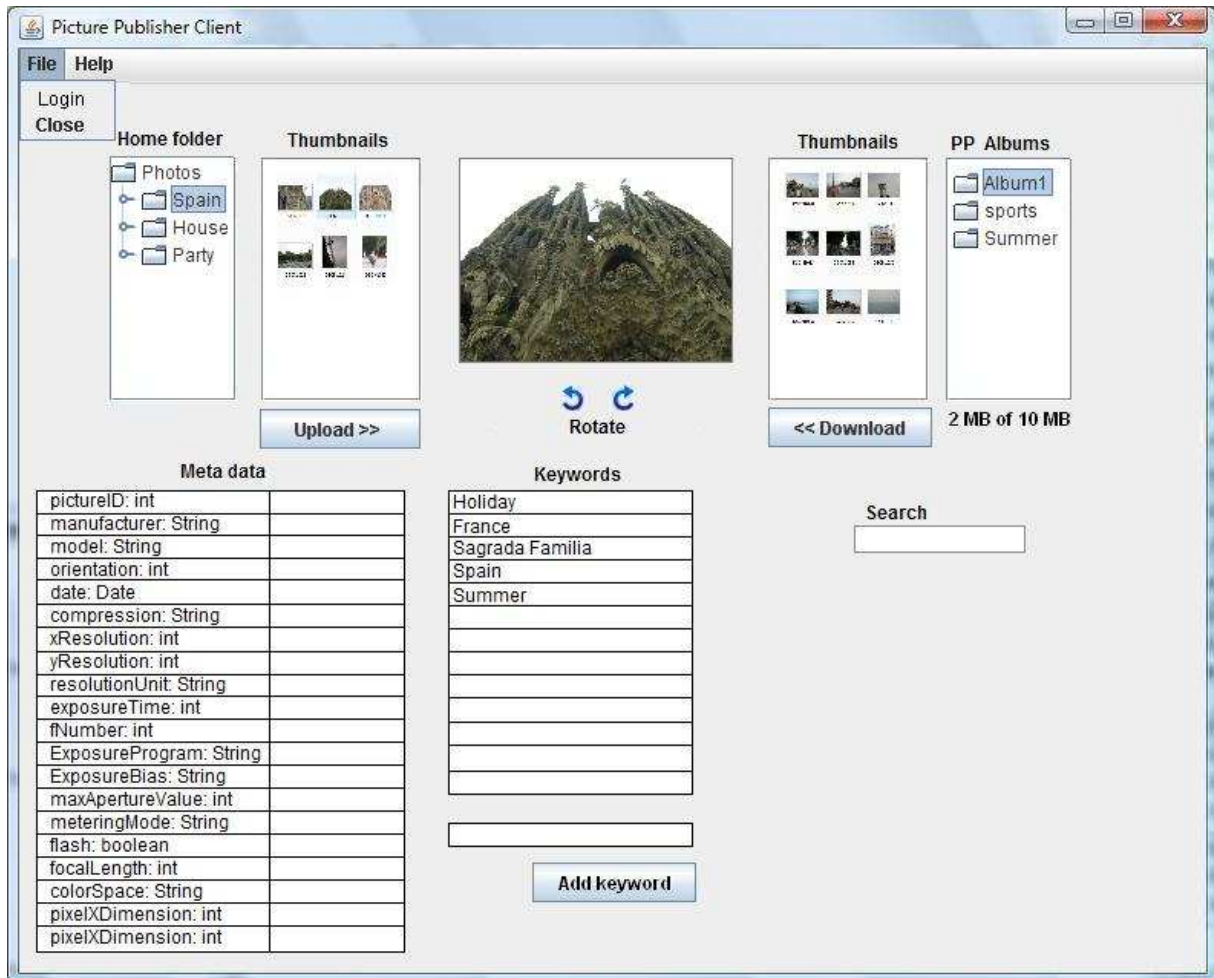
The main window [form 3]

Functional requirements:

- Picture downloading
- Picture storage
- Rotating pictures
- Adding keywords to pictures
- Sorting pictures
- Picture searching
- User input of metadata
- Removing a picture
- Upload quota



The main window with the file menu [form 4]
Functional requirements:



The Picture Publisher Webpage

The empty webpage [form 8]

Functional requirements:

Viewing Pictures

Separation of keywords

“How to” document

Picture searching



The “login incorrect” webpage [form 9]

Functional requirements:

See form 8

The screenshot displays the 'Picture Publisher' website interface. At the top, there is a teal header with the site name 'Picture Publisher' on the left, and links for 'New? Sign up!' and 'Download client' on the right. Below the header is a green navigation bar containing search options: 'Use publisher keywords' and 'Use viewer keywords', a search input field with a 'Search' button, and login fields for 'Alias' and 'Password' with a 'Login' button. A blurred image of a train is visible on the right side of the header. The main content area is white and features a red error message 'Login incorrect' with a 'Help!' link. On the left, under the heading 'Search result', there are two empty rounded rectangular boxes. On the right, under the heading 'Tag this picture', there is one empty rounded rectangular box.

The online sign up [form 10]

Functional requirements:

Account Creation

Personal Information Access



The screenshot shows a web page for 'Picture Publisher'. The header has a teal background with the text 'Picture Publisher' and a link 'Download client'. Below the header is a green bar. The main content area is white and contains the text 'Sign up now!'. There are three input fields for 'Alias', 'Password', and 'Email'. Below these fields is a 'Sign up' button. At the bottom, there is a checkbox labeled 'Allow your personal information to be accessed'. The right side of the page is a grey vertical bar.

Picture Publisher [Download client](#)

Sign up now!

Alias

Password

Email

[Sign up](#)

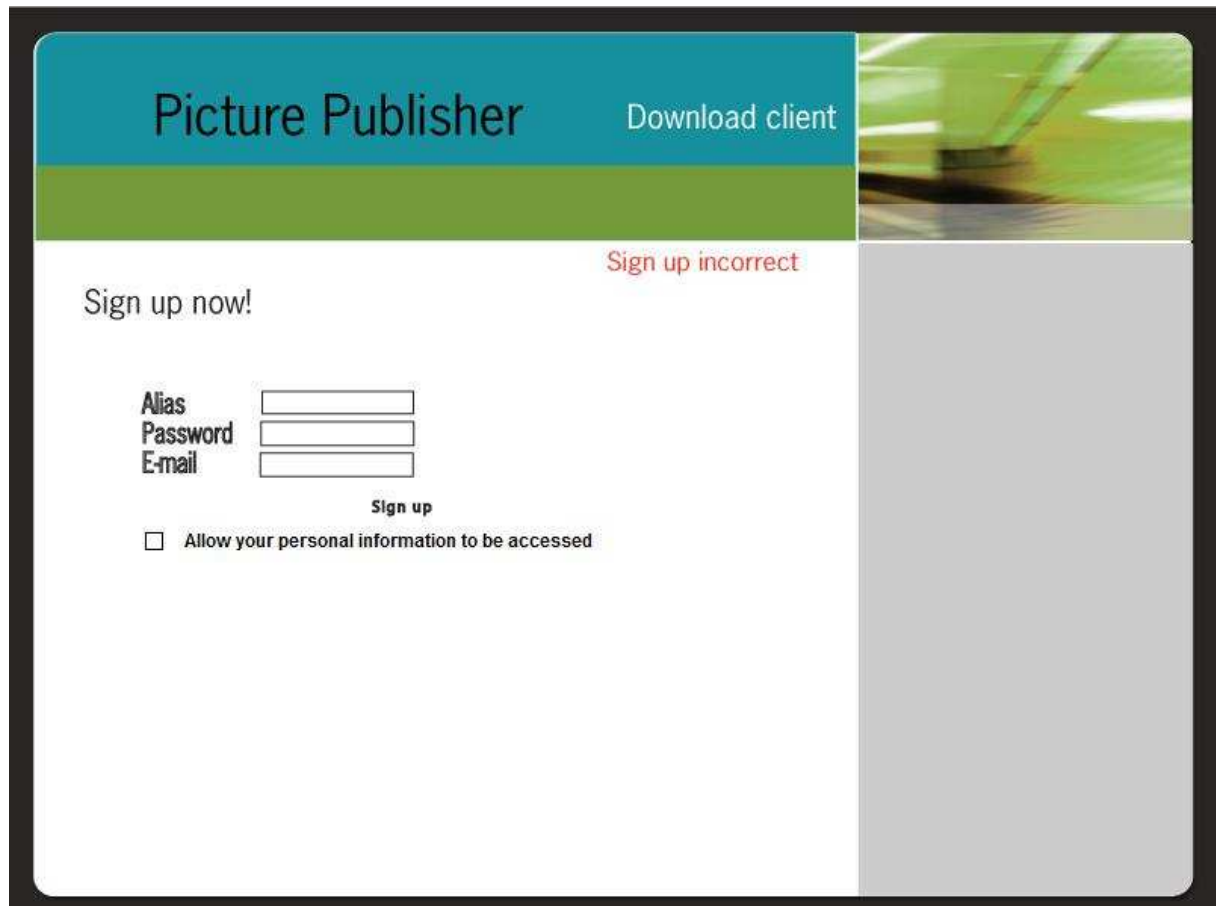
Allow your personal information to be accessed

The online sign up is incorrect [form 11]

Functional requirements

Account Creation

Personal Information Access



Picture Publisher [Download client](#)

Sign up now! Sign up incorrect

Alias
Password
E-mail

Sign up

Allow your personal information to be accessed

The webpage displays search results and that the user is logged in [form 12]

Functional requirements:

“How to” document

Viewing Pictures

Adding keywords to pictures through the web

Picture searching

Separation of keywords

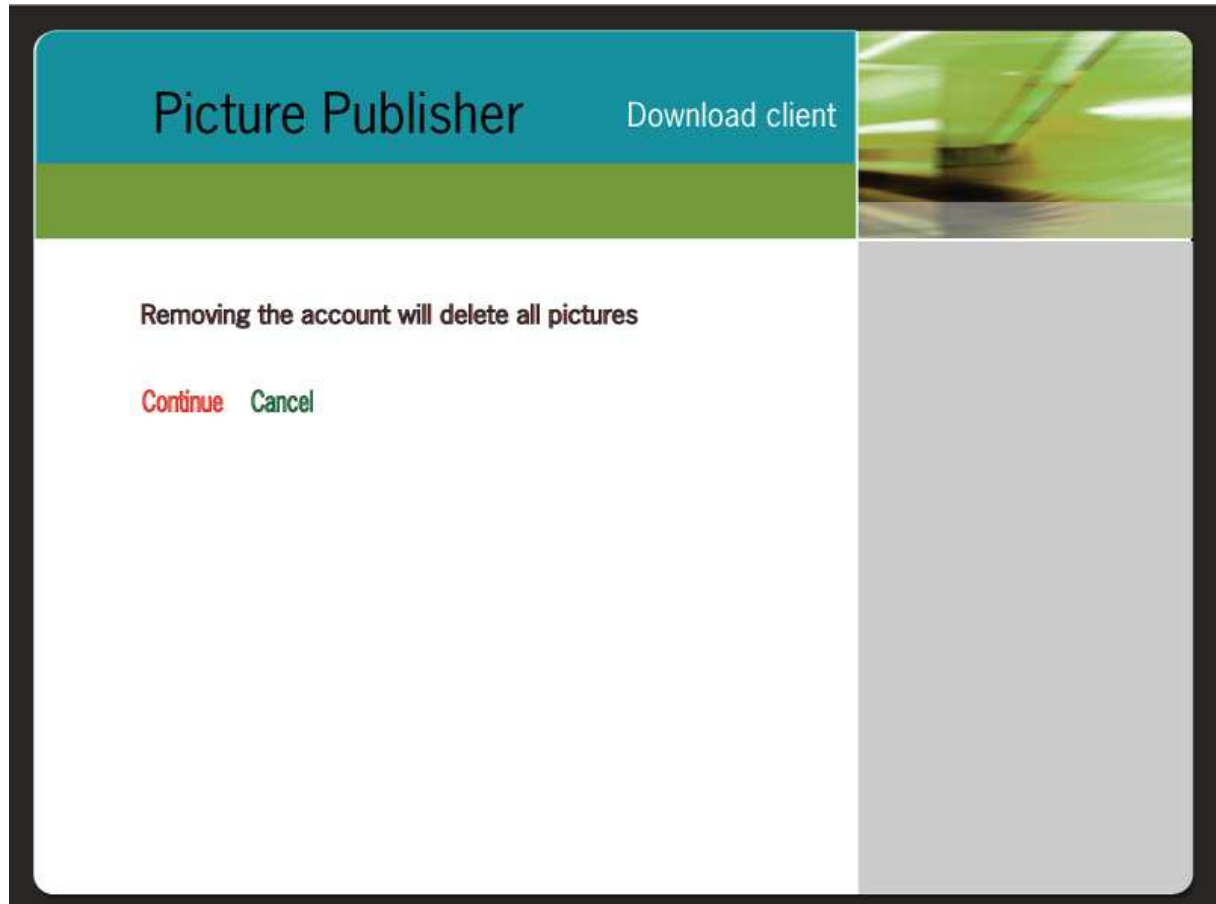
Removing accounts



The webpage displays the remove account form [form 13]

Functional requirements:

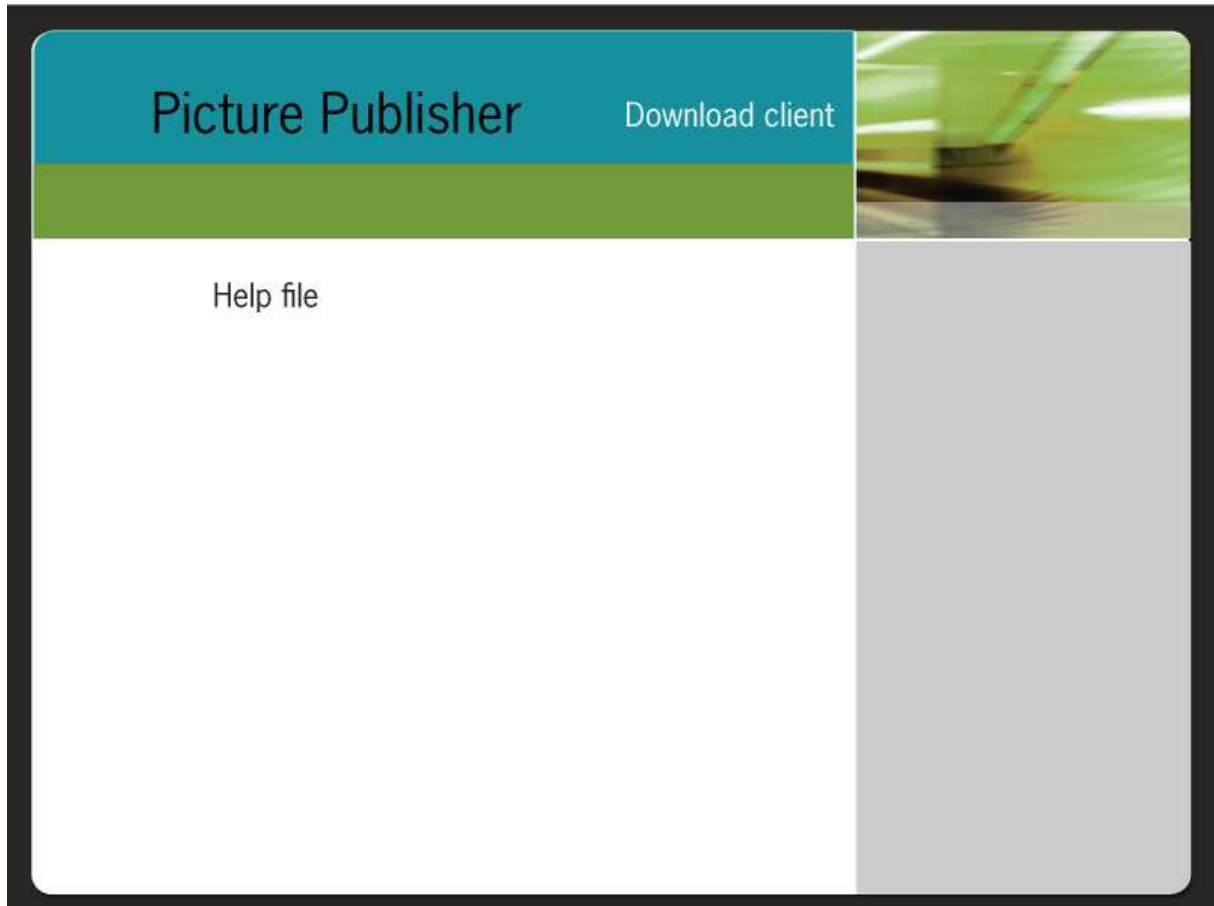
Removing accounts



The webpage displays the help form [form 14]

Functional requirements:

“How to” document



4.3. Names of controls and fields, methods/procedures and triggers for each form

The Picture Publisher Client

Form 1

Names: The form has a login field called "Login name" and a password field called "Password". The form also has a button called "Login Button".

Triggered by: The user presses "login" from the file menu

Controls

Login Button: A login attempt is made with the data from "Login name" and "Password".

Form 2

Names: The form has a login field called "Login name" and a password field called "Password". The form also has a button called "Login Button".

Triggered by: The data from "User name" or "Password" was incorrect

Controls

Login Button: A login attempt is made with the data from "Login name" and "Password".

Form 3

Names: Home folder view is called "Home folder"
Local thumbnail view is called "Local Thumbnails"
The upload button is called "Upload"
The picture view is called "Picture"
The rotate buttons are called "Rotate Left" and "Rotate right"
The server thumbnail view is called "Server Thumbnails"
The download button is called "Download"
The folder view called "Album"
The meta data view is called "Meta data"
The keyword view is called "Keywords"
The add keyword button is called "Add keyword"
The search view is called "Search"
The file menu is called "File"
The help menu is called "Help"
The home folder view is called "Home folder view"
The local thumbnails view is called "Local thumbnails view"
The picture view is called "Picture View"
The PP albums view is called "Album view"
The server thumbnails view is called "Server thumbnails view"

Triggered by: Program startup, loads information to the views

Controls

Home folder view:	Calls the method UpdateThumbnailsView
Local thumbnails view	Calls the method UpdatePictureView, updateMetadataView, updateKeywordsView
Server thumbnails view	Calls the method UpdatePictureView, updateMetadataView, updateKeywordsView
Album view	Calls the method UpdateThumbnailsView
Upload:	Calls the method ImportAction
Rotate left:	Calls the method RotateAction(int direction)
Rotate right:	Calls the method RotateAction(int direction)
Download:	Calls the method ExportAction
Meta data:	Calls the method UpdateMetadataAction
Add keyword:	Calls the method AddKeywordAction
Search:	Calls the method SearchAction
File:	Displays the file menu
Help:	Displays the help menu

Form 4

Names: The same as form 3 and
The login item called "login"
The close item called "close"

Triggered by: The user presses "File" from the file menu

Controls

Login: Displays form 1
Close: Calls System.close()

Form 5

Names: The same as form 3 and
The help item called "Help file"

Triggered by: The user presses "Help" from the help menu

Controls

Help file: Displays the help file

Form 6

Names: The same as form 3 and
The item called "Add album"
The item called "Remove album"
The menu item called "Set access restrictions"
The item called "Private"
The item called "Public"

Triggered by: The user right clicks in the "Album view"

Controls

Add album: Calls the method AddAlbumAction
Remove album: Calls the method RemoveAlbumAction
Private: Calls the method SetAccessPrivateAction

Triggered by: The Sign Up link was clicked

Controls

Sign Up Button Attempts to sign up the user with the data from the fields.

Form 11

Names:

The alias field is called "Alias"

The password field is called "Password"

The E-mail field is called "E-mail"

The Sign Up button is called "Sign Up button"

Triggered by: The Sign Up was incorrect

Controls

Sign Up Button Attempts to sign up the user with the data from the fields.

Form 12

Names:

- The sign up link is called "Sign Up"
- The download client link is called "Download"
- The search field is called "Search string"
- The search button is called "Search"
- The alias field is called "Alias"
- The password field is called "Password"
- The login button is called "Login button"
- The search result view is called "Search result"
- The picture view is called "Picture"
- The tag table is called "Tag table"
- The tag field is called "Tag field"
- The tag button is called "Tag"
- The remove account button is called "Remove account"
- The help button is called "Help button"

Triggered by: The user performs a search

Controls

Sign Up:	Calls form 10
Download:	Downloads the client
Search	Performs a search with the data from "Search string"
Search result	Picture view and tag table are updated
Picture	The picture is shown in its original size
Tag	The data from "Tag field" is inserted into the "tag table" and connected to the picture.
Remove account	Form 13 is displayed
Help button	Form 14 is displayed

Form 13

Names:

- The Cancel button is called "Cancel"
- The Continue button is called "Continue"

Triggered by: The "remove button" was clicked

Controls

Cancel:	Cancels the removal
Continue	Removes the account and all the pictures associated to it

Form 14

Names: The text document is called "help file"

Triggered by: The "help button" was clicked

Controls: None

5. Design Details

5.1 CRC Cards

Picture	
Know picture id Know which keyword this picture has Know which metadata this picture has	Keyword Metadata

Keyword	
Know which keyword it is	

Album	
Know the album id Know the album name Know the album description Know which pictures that belongs to this album	Picture

Metadata	
Know the metadata of a picture	

FolderView	
Display folders from the local file system and albums from the server	Album

ThumbnailsView	
Display thumbnails of the pictures in the selected folder	FolderView

PictureView	
Display the selected picture	ThumbnailView

MetadataView	
Display metadata for the selected picture	PictureView Metadata

KeywordView	
Display keywords for the selected picture	PictureView Keyword

Reference to RD: Page: 8 Section: User Requirements Definition
Header: Rotating Picture

RotateAction	
Rotate a picture in the requested direction	Metadata PictureView

Reference to RD: Page: 17 Section: System Requirements Specification
Header: Picture Storage

ImportAction	
Send a picture and two alternate versions to the server	PictureService PictureUtils

Reference to RD: Page: 8 Section: User Requirements Definition
Header: Picture Downloading

ExportAction	
Download the requested picture from the server	PictureService ThumbnailsView

Reference to RD: Page: 9 Section: User Requirements Definition
Header: Removing a picture

RemovePictureAction	
Remove a specified picture	PictureService ThumbnailsView

Reference to RD: Page: 9 Section: User Requirements Definition
Header: User input of metadata

UpdateMetadataAction	
Update the metadata for the selected picture	PictureService

Reference to RD: Page: 8 Section: User Requirements Definition
Header: Sorting Pictures

AddAlbumAction	
Add an album that you can put pictures in	PictureService FolderView

Reference to RD: Page: 8 Section: User Requirements Definition
Header: Sorting Pictures

MovePictureAction	
Move a picture from one album to another	PictureService

Reference to RD: Page: 8 Section: User Requirements Definition
Header: Sorting Pictures

RemoveAlbumAction	
Remove an album	PictureService FolderView

Reference to RD: Page: 8 Section: User Requirements Definition
Header: Adding Keywords to Pictures

AddKeywordAction	
Add a keyword to a picture	PictureService KeywordView

RemoveKeywordAction	
Remove a keyword	PictureService KeywordView

Reference to RD: Page: 9 Section: User Requirements Definition
Header: Picture searching Reference to RD: Page: 19 Section: System

Requirements Specification
Header: System behavior in Picture Searching

SearchAction	
To find and return the pictures that match the given attributes	PictureService ThumbnailView

Reference to RD: Page: 17 Section: System Requirements Specification
Header: Optimization of image quality

PictureUtils	
Resize a picture to requested resolution	

Reference to RD: Page: 17 Section: System Requirements Specification
Header: Picture Storage

Reference to RD: Page: 8 Section: User Requirements Definition
Header: Picture Downloading

Reference to RD: Page: 9 Section: User Requirements Definition
Header: Removing a picture

Reference to RD: Page: 8 Section: User Requirements Definition
Header: Adding Keywords to Pictures

Reference to RD: Page: 8 Section: User Requirements Definition
Header: Sorting Pictures

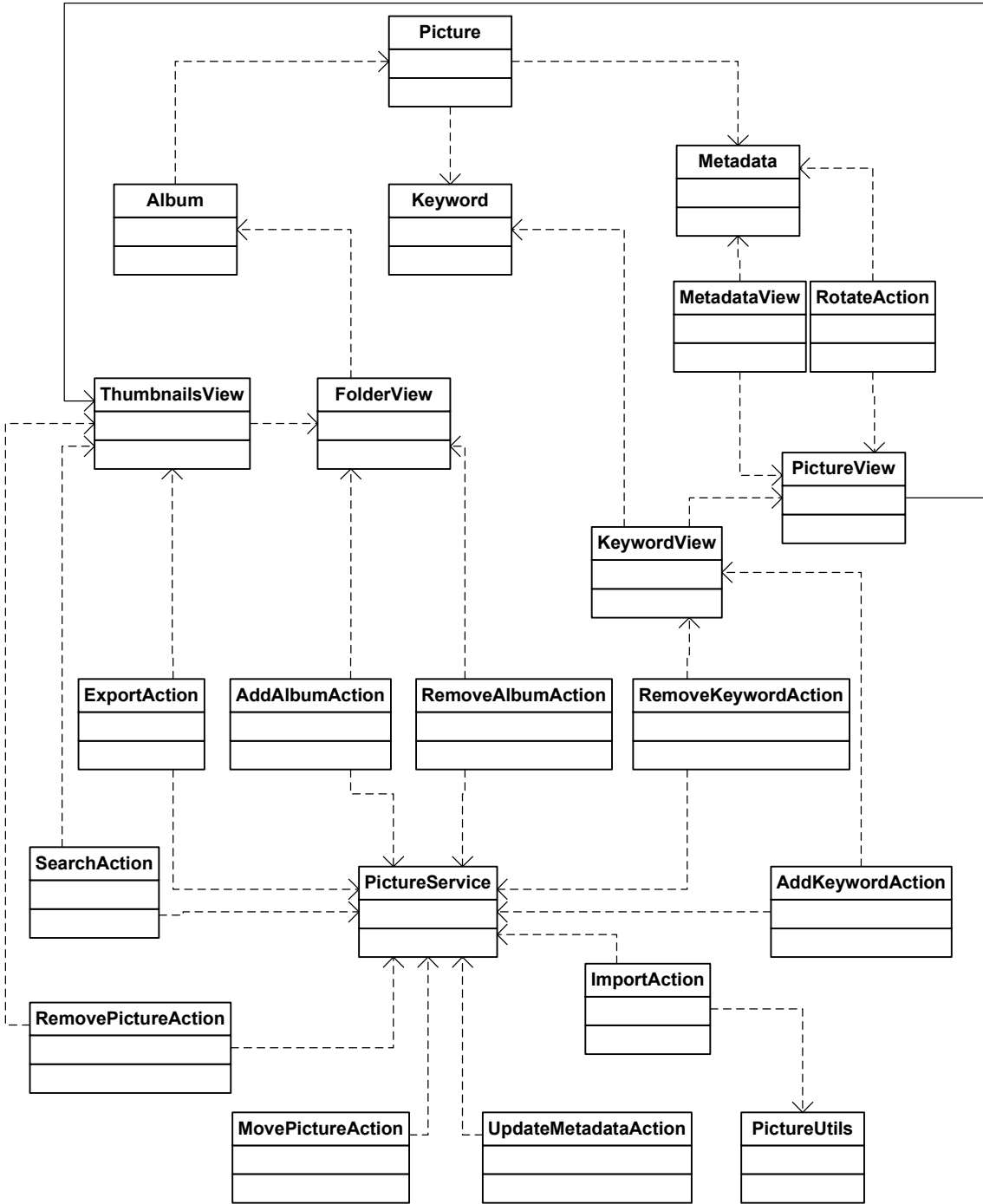
Reference to RD: Page: 9 Section: User Requirements Definition
Header: Picture searching

Reference to RD: Page: 19 Section: System Requirements Specification
Header: System behavior in Picture Searching

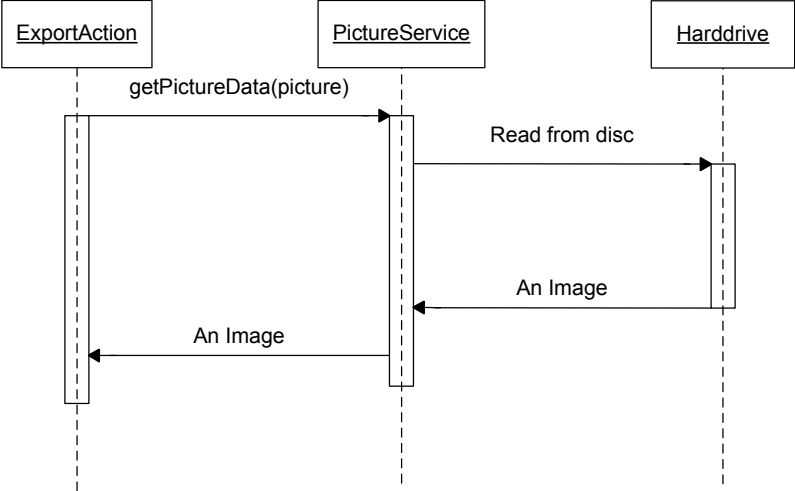
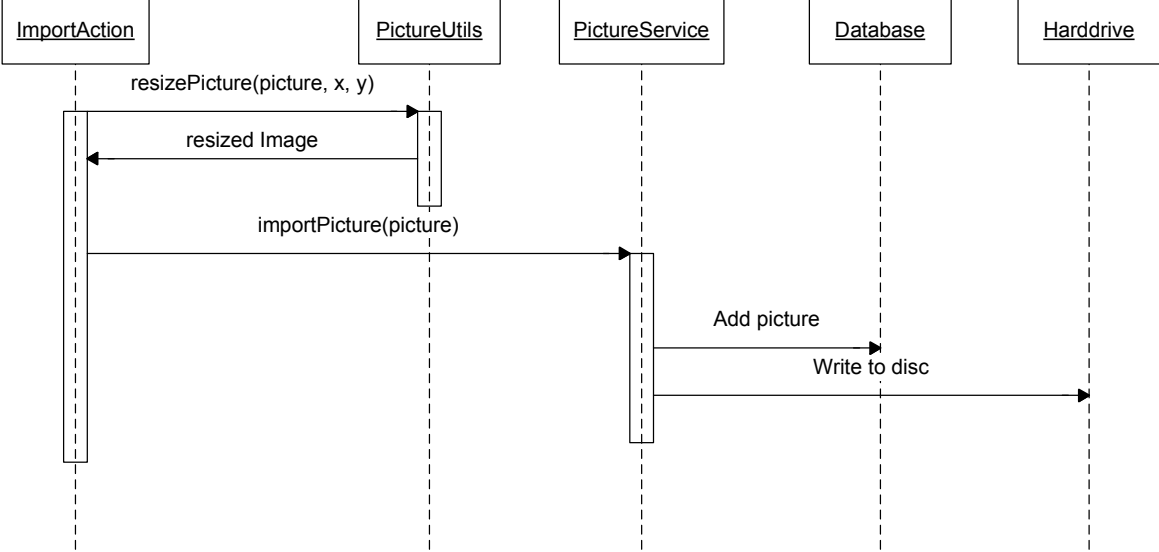
Reference to RD: Page: 9 Section: User Requirements Definition
Header: User input of metadata

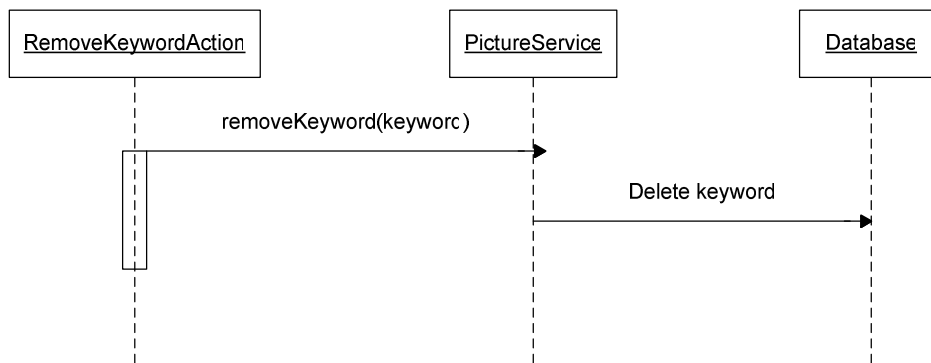
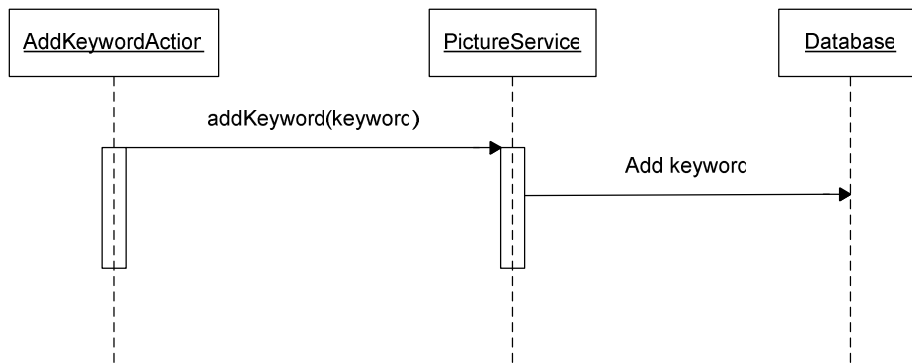
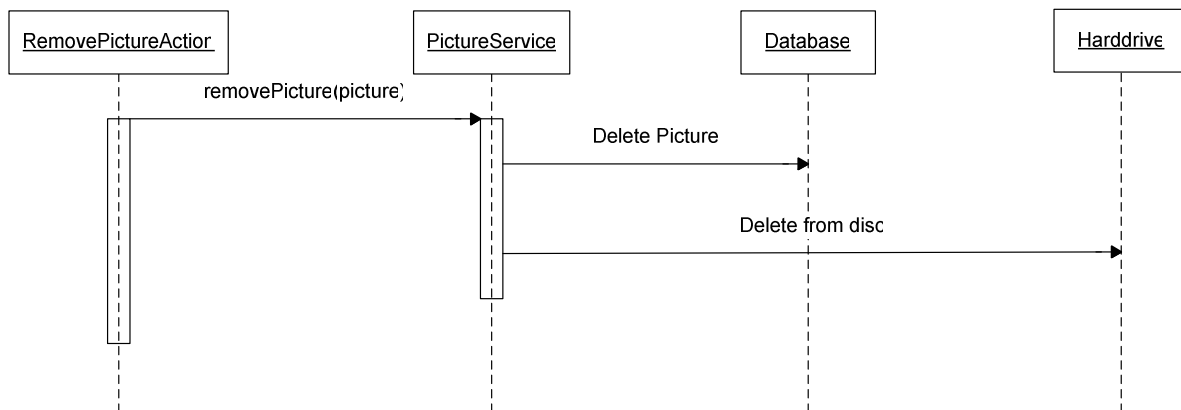
PictureService	
Add picture object to the database and the picture files to the file system	
Send a picture to the client	
Remove a picture object from the database and remove the picture files from the file system	
Add a keyword to the database	
Remove a given keyword from the database	
Add an album to the database	
Remove an album from the database	
Add a specified picture to a album	
Remove a specified picture from a album	
Update metadata for a given picture	
Perform a search	

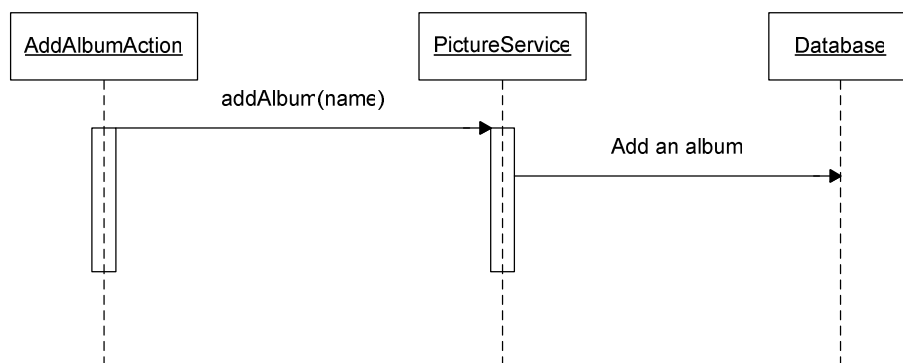
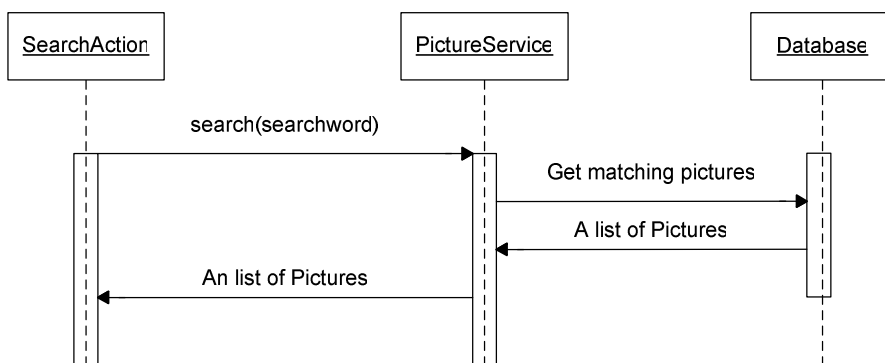
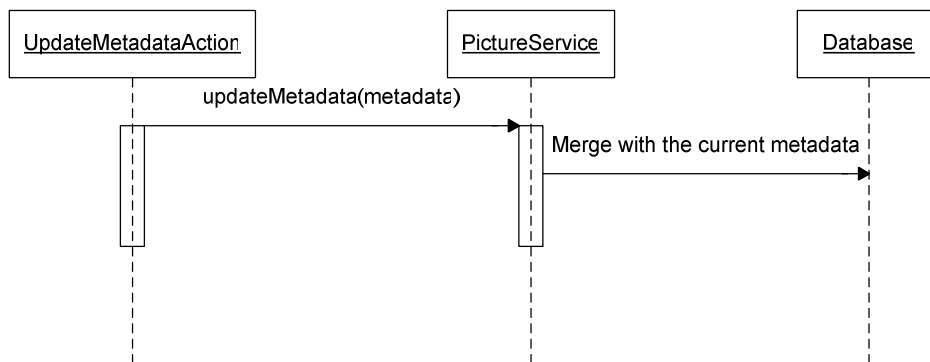
5.2 Class Diagram

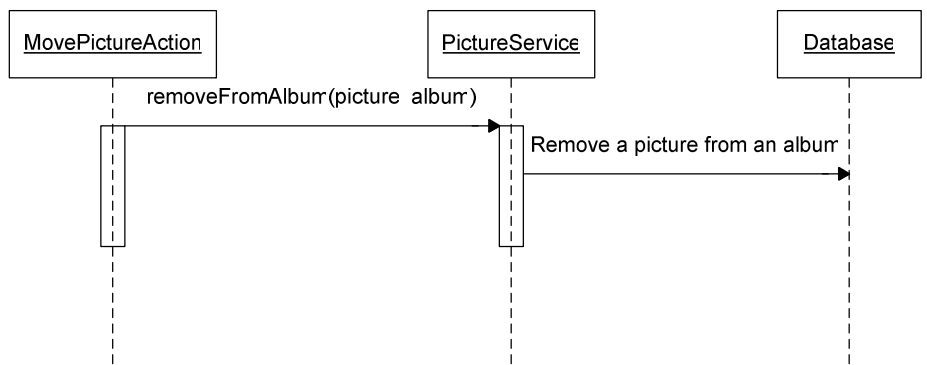
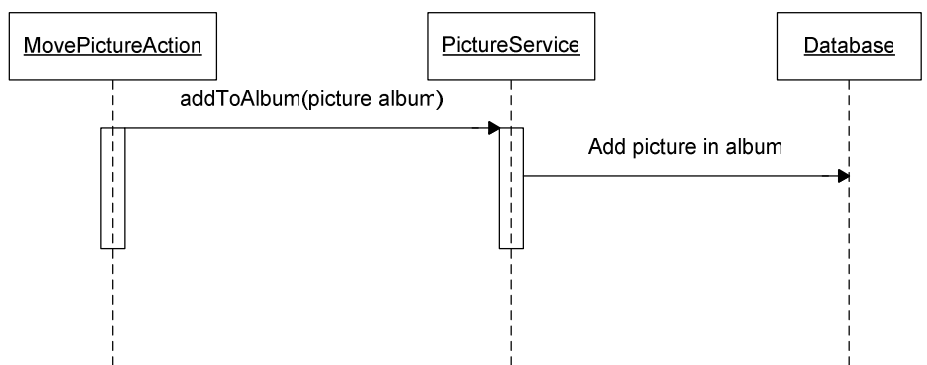
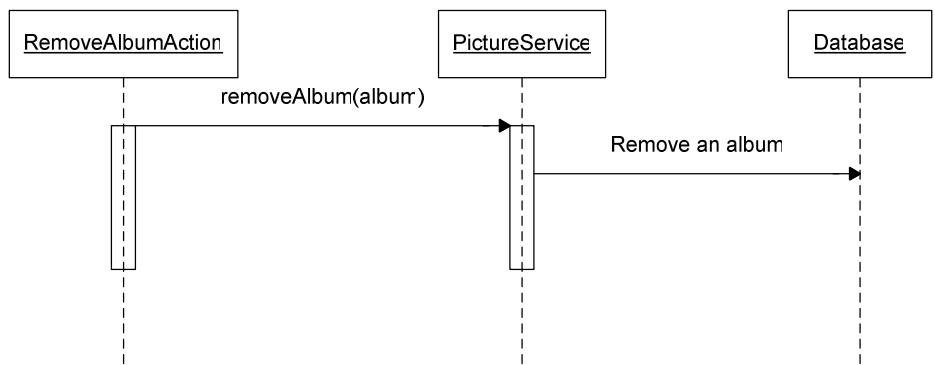


5.4 Interaction Diagrams









5.5 Detailed Design

Database Design

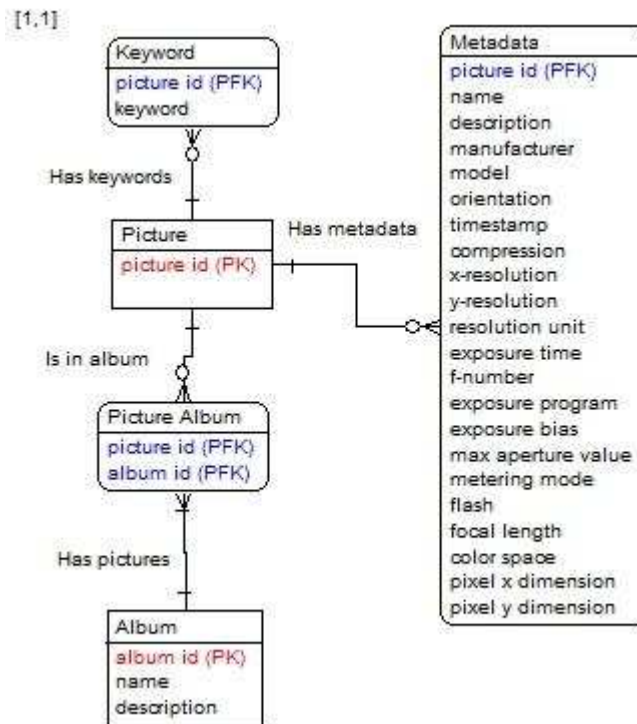


Table name	Table type	Primary key	# columns
Album	independent	album_id	3
Keyword	dependent	picture_id	2
Metadata	dependent	picture_id	22
Picture	independent	picture_id	1
Picture_Album	dependent	picture_id, album_id	2

Table: Album

Table name	Album
Primary key	pk_Album

Columns:

Key	Column name	Domain	Data type	N ¹⁾	U ²⁾	C ³⁾	D ⁴⁾
PK	album_id		Integer	YES	NO	NO	NO
	name		Varchar(n) (255)	NO	NO	NO	NO
	description		Text	NO	NO	NO	NO

Relationships:

Constraint name	Relationship type	Parent table	Child table	Card.
Has pictures	Identifying	Album	Picture_Album	1:N

Table: Keyword

Table name	Keyword
Primary key	pk_Keyword

Columns:

Key	Column name	Domain	Data type	N ¹⁾	U ²⁾	C ³⁾	D ⁴⁾
PFK	picture_id		Integer	YES	NO	NO	NO
	keyword		Varchar(n) (20)	NO	NO	NO	NO

Relationships:

Constraint name	Relationship type	Parent table	Child table	Card.
Has keywords	Identifying	Picture	Keyword	1:N

¹ Not null

² Unique

³ Check

⁴ Default

Table: Metadata

Table name	Metadata
Primary key	pk_Metadata

Columns:

Key	Column name	Domain	Data type	N ¹⁾	U ²⁾	C ³⁾	D ⁴⁾
PFK	picture_id		Integer	YES	NO	NO	NO
	name		Varchar(n) (255)	NO	NO	NO	NO
	description		Text	NO	NO	NO	NO
	manufacturer		Varchar(n) (255)	NO	NO	NO	NO
	model		Varchar(n) (255)	NO	NO	NO	NO
	orientation		Bit (1)	NO	NO	NO	NO
	timestamp		Timestamp(p) (0)	NO	NO	NO	NO
	compression		Varchar(n) (20)	NO	NO	NO	NO
	x_resolution		Varchar(n) (20)	NO	NO	NO	NO
	y_resolution		Varchar(n) (20)	NO	NO	NO	NO
	resolution_unit		Varchar(n) (20)	NO	NO	NO	NO
	exposure_time		Varchar(n) (20)	NO	NO	NO	NO
	f_number		Varchar(n) (20)	NO	NO	NO	NO
	exposure_program		Varchar(n) (20)	NO	NO	NO	NO
	exposure_bias		Varchar(n) (20)	NO	NO	NO	NO
	max_aperture_value		Varchar(n) (20)	NO	NO	NO	NO
	metering_mode		Varchar(n) (20)	NO	NO	NO	NO
	flash		Varchar(n) (20)	NO	NO	NO	NO
	focal_length		Varchar(n) (20)	NO	NO	NO	NO
	color_space		Varchar(n) (20)	NO	NO	NO	NO
	pixel_x_dimension		Varchar(n) (20)	NO	NO	NO	NO
	pixel_y_dimension		Varchar(n) (20)	NO	NO	NO	NO

Relationships:

Constraint name	Relationship type	Parent table	Child table	Card.
Has metadata	Identifying	Picture	Metadata	1:N

¹ Not null
² Unique
³ Check
⁴ Default

Table: Picture

Table name	Picture
Primary key	pk_Picture

Columns:

Key	Column name	Domain	Data type	N ¹⁾	U ²⁾	C ³⁾	D ⁴⁾
PK	picture_id		Integer	YES	NO	NO	NO

Relationships:

Constraint name	Relationship type	Parent table	Child table	Card.
Has metadata	Identifying	Picture	Metadata	1:N
Has keywords	Identifying	Picture	Keyword	1:N
Is in album	Identifying	Picture	Picture_Album	1:N

Table: Picture_Album

Table name	Picture_Album
Primary key	pk_Picture_Album

Columns:

Key	Column name	Domain	Data type	N ¹⁾	U ²⁾	C ³⁾	D ⁴⁾
PFK	picture_id		Integer	YES	NO	NO	NO
PFK	album_id		Integer	YES	NO	NO	NO

Relationships:

Constraint name	Relationship type	Parent table	Child table	Card.
Is in album	Identifying	Picture	Picture_Album	1:N
Has pictures	Identifying	Album	Picture_Album	1:N

¹ Not null
² Unique
³ Check
⁴ Default

Method Design

Picture

- id: int
- keywords : Set<Keyword>
- metadata : Metadata

Picture()

getId(): int
setId(int id): void
getKeywords(): Set<Keyword>
setKeywords(Set<Keyword> keywords): void
getMetadata(): Metadata
setMetadata(Metadata metadata): void

Metadata

- picture: Picture
- description: String
- name: String
- manufacturer: String
- model: String
- orientation: int
- date: Date
- compression: String
- xResolution: int
- yResolution: int
- resolutionUnit: String
- exposureTime: int
- fNumber: int
- exposureProgram: String
- exposureBias: String
- maxApertureValue: int
- meteringMode: String
- flash: boolean
- focalLength: int
- colorSpace: String
- pixelXDimension: int
- pixelYDimension: int

Metadata(Picture picture)

getColorSpace(): String
setColorSpace(String colorSpace): void
getCompression(): String
setCompression(String compression): void
Date getDate(): Date

setDate(Date date): void
getDescription(): String
setDescription(String description): void
getExposureBias(): String
setExposureBias(String exposureBias): void
getExposureProgram(): String
setExposureProgram(String exposureProgram): void
getExposureTime(): int
setExposureTime(int exposureTime): void
isFlash(): boolean
setFlash(boolean flash): void
getFNumber(): int
setFNumber(int number): void
getFocalLength(): int
setFocalLength(int focalLength): void
getManufacturer(): String
setManufacturer(String manufacturer): void
getMaxApertureValue(): int
setMaxApertureValue(int maxApertureValue): void
getMeteringMode(): String
setMeteringMode(String meteringMode): void
getModel(): String
setModel(String model): void
getName(): String
setName(String name): void
getOrientation(): int
setOrientation(int orientation): void
getPictureID(): int
setPictureID(int pictureID): void
getPixelXDimension(): int
setPixelXDimension(int pixelXDimension): void
getPixelYDimension(): int
setPixelYDimension(int pixelYDimension): void
getResolutionUnit(): String
setResolutionUnit(String resolutionUnit): void
getXResolution(): int
setXResolution(int resolution): void
getYResolution(): int
setYResolution(int resolution): void

Keyword

- picture: Picture
 - keyword: String
-

Keyword(Picture picture, String keyword)

getKeyword():String
setKeyword(String keyword): void
getPicture(): Picture
setPicture(Picture picture): void

Album

- id: int
 - name: String
 - description: String
 - pictures : List<Picture>
-

Album()

getPictures(): List<Picture>
addPicture(Picture picture): void
removePicture(Picture picture)
getDescription(): String
setDescription(String description): void
getName(): String
setName(String name): void

ImportAction

- none
-

ImportAction(String path)

Name: run()

Parameters: none

Return value: none

Description: Sends a picture and two alternate versions of it to the server

Database: none

Pre-condition: There exists a picture to upload

Validity checks:

The path is a picture

False: Throw InvalidArgumentException

Post-condition: The picture and the alternate versions of the picture are uploaded to the server

Calls: PictureService.importPicture, PictureUtils.resizePicture

Called by: ThumbnailsView

ExportAction

- none

Name: run()

Parameters: none

Return value: An image

Description: Downloads a picture from the server to the user's local computer

Database: none

Pre-condition: The picture to download exists

Validity checks: none

Post-condition: The picture is stored locally on the user's computer

Calls: PictureService.getPictureData

Called by: ThumbnailsView

RemovePictureAction

- none

Name: run()

Parameters: none

Return value: none

Description: Removes a picture from the server

Database: none

Pre-condition: The picture to be removed exists

Validity checks: none

Post-condition: The picture data is removed from the file system and the picture object is removed from the database

Calls: PictureService.removePicture

Called by: PictureView, ThumbnailsView

RotateAction

- none

RotateAction(int direction)

Name: run()

Parameters: none

Return value: none

Description: Rotates the picture in the specified direction

Database: none

Pre-condition: The picture that we want to rotate exist

Validity checks: none

Post-condition: The metadata defining the rotation of the specified picture is updated to the correct value

Calls: none

Called by: PictureView

AddAlbumAction

- none

AddAlbumAction(String name)

Name: run()

Parameters: none

Return value: none

Description: Adds an album

Database: none

Pre-condition: none

Validity checks: none

Post-condition: An album object is created in the album table in the database

Calls: PictureService.addAlbum

Called by: FolderView

RemoveAlbumAction

- none

Name: run()

Parameters: none

Return value: none

Description: Removes an album

Pre-condition: The album exists

Validity checks: none

Post-condition: The specified album object is removed from the album table in the database

Calls: PictureService.removeAlbum

Called by: FolderView

AddKeywordAction

- none

AddKeywordAction(Picture picture, String keyword)

Name: run()

Parameters: none

Return value: none

Description: Adds a keyword to a picture

Database: none

Pre-condition: The picture you add the keyword to exists

Validity checks: none

Post-condition: The keyword object is created in the database

Calls: Pictureservice.addKeyword

Called by: KeywordView

RemoveKeywordAction

- none

RemoveKeywordAction(Keyword keyword)

Name: run()

Parameters: none

Return value: none

Description: Removes a keyword from a picture

Database: none

Pre-condition: The keyword to remove exists

Validity checks: none

Post-condition: The keyword object is removed from the database

Calls: PictureService.removeKeyword

Called by: KeywordView

SearchAction

- none

SearchAction(String searchPhrase): void

Name: run()

Parameters: none

Return value: none

Description: Performs a search for pictures where the description, keywords or metadata contains the search phrase

Database: none

Pre-condition: none

Validity checks: none

Post-condition: none

Calls: PictureService.search

Called by: Thumbnailsview

PictureService

- none

PictureService(): void

Name: importPicture

Parameters:

Picture picture

A picture object to be stored in the database

Image original

The original picture to be stored on in the file system for later use

Image copy

A lower resolution copy of the original to be stored in the file system that will be displayed on the web

Image thumbnail

A thumbnail version of the original picture to be stored in the file system also used to be displayed on the web

Return value: none

Description: Import a picture into the system

Database: Picture

Pre-condition: There exists a picture to add

Validity checks:

Images are of the expected type

false: Throws InvalidArgumentException

The amount of available storage space is sufficient

false: Throws OutOfSpaceException

Post-condition: A picture is added

Called by: ImportAction.run()

Name: getPictureData

Parameters:

Picture picture

The picture object representing the image to be returned

Return value: Image

Description: Get the picture data for the given picture

Database: none

Pre-condition: The picture to download exists

Validity checks: none

Post-condition: none

Called by: ExportAction.run()

Name: removePicture

Parameters:

Picture picture

A picture object representing the picture to be removed

Return value: none

Description: Remove the picture from the database and file system.

Database: Picture, Metadata, Picture_Album

Pre-condition: The picture to be removed exists

Validity checks: none

Post-condition: The picture is removed

Calls: none

Called by: RemovePictureAction.run()

Name: addKeyword

Parameters:

Keyword keyword

The keyword to be added

Return value: none

Description: Add a keyword to the database.

Database: Keyword

Pre-condition: There exists a keyword to add

Validity checks: none

Post-condition: A keyword is added

Calls: none

Called by: AddKeywordAction.run()

Name: removeKeyword

Parameters:

Keyword keyword

A keyword object representing the keyword to be removed

Return value: none

Description: Remove the given keyword from the database

Database: Keyword

Pre-condition: There exists a keyword to remove

Validity checks:

Post-condition: A keyword is removed

Calls: none

Called by: RemoveKeywordAction.run()

Name: addAlbum
Parameters:
String name
 The name of the album
Return value: none
Description: Add a new album
Database: Picture_Album
Pre-condition: There exists an album to add
Validity checks: none
Post-condition: An album is added
Calls: none
Called by: AddAlbumAction.run()

Name: removeAlbum
Parameters:
Album album
 An album object representing the album to be removed
Return value: none
Description: Remove the album from the database.
Database: Picture_Album
Pre-condition: There exists an album to remove
Validity checks: none
Post-condition: An album is removed
Calls: none
Called by: RemoveAlbumAction.run()

Name: addToAlbum
Parameters:
Picture picture
 A picture object representing the picture to be added
Album album
 An album object representing the album where we want to add the
picture
Return value: none
Description: Adds the given picture to the given album
Database: Picture_Album
Pre-condition: There exists a album to add the picture into and there exists a picture
to add
Validity checks: none
Post-condition: A picture is added in the album
Calls: none
Called by: MovePictureAction.run()

Name: removeFromAlbum

Parameters:

Picture picture

A picture object representing the picture to be removed from an album

Album album

An album object representing the album from which we want to remove a picture

Return value: none

Description: Remove the given picture from the given album

Database: none

Pre-condition: There exists a picture to remove in the album we specified

Validity check: none

Post-condition: The required changes are made to the Picture_Album table.

Calls: none

Called by: MovePictureAction.run()

Name: updateMetadata

Parameters:

Metadata metadata

A metadata object representing the metadata to update

Return value: none

Description: Merge the given metadata with the metadata with the same id in the database.

Database: Metadata

Pre-condition: none

Validity checks: none

Post-condition: Changes are made to the metadata table in the database.

Calls: none

Called by: UpdateMetadataAction.run, RotateAction.run

Name: search

Parameters:

String searchPhrase

The phrase to search for

Return value: none

Description: Return a list of pictures that match the given search phrase

Database: Picture

Precondition: none

Validity checks: none

Post-condition: none

Calls: none

Called by: SearchAction.run()

UpdateMetadataAction

- none

UpdateMetadataAction(Metadata data): void

Name: run()

Parameters: none

Return value: none

Description: Update the metadata for the currently selected picture

Database: none

Precondition: There exists metadata to update and a picture is selected

Validity checks: none

Post-condition: The metadata is updated

Calls: PictureService.updateMetadata.

Called by: MetadataView

MovePictureAction

- none

MovePictureAction(Picture picture, Album album): void

Name: run()

Parameters: none

Return Value: none

Description: Moves a picture from one album to another

Database: none

Pre-condition: none

Validity checks: none

Post-condition: none

Calls: PictureService.removeFromAlbum, PictureService.addToAlbum

Called by: FolderView

PictureUtils

- none

Name: `resizePicture()`

Parameters:

Image image

The image to resize

int xResolution

The resolution in horizontal direction

int yResolution

The resolution in vertical direction

Return value: Image

Description: Creates a resized version of the given image

Database: none

Pre-condition: There exists a picture to resize

Validity checks: none

Post-condition: The metadata of the picture is updated

Calls: none

Called by: `ImportAction.resizePicture`

FolderView

- none

Description: Creates the objects seen in the folderview

`createPartControl(): void`

Name: `updateFolderView`

Parameters: none

Return value: none

Description: Updates the folders seen in the folderview

Database: none

Pre-condition: A folderview has been updated

Validity checks: none

Post-condition: none

Calls: none

Called by: `AddAlbumAction`, `RemoveAlbumAction`, `MovePictureAction`

ThumbnailsView

- none

Description: Creates the objects seen in the thumbnailsview
createPartControl (): void

Name: updateThumbnailsView

Parameters: none

Return value: none

Description: Updates the thumbnails seen in the thumbnailsview

Database: none

Pre-condition: A thumbnailsview has been updated

Validity checks: none

Post-condition: none

Calls: none

Called by: ImportAction, ExportAction, RemovePictureAction

PictureView

- none

Description: Creates the objects seen in the pictureview
createPartControl(): void

Name updatePictureView

Parameters: none

Return value: none

Description: Updates the picture seen in the pictureview

Database: none

Pre-condition: none

Validity checks: none

Post-condition: none

Calls: none

Called by: RotateAction, ThumbnailsView

MetadataView

- none

Description: Creates the objects seen in the metadataview
createPartControl(): void

Name: updateMetadataView

Parameters: none

Return value: none

Description: Updates the picture seen in the metadataview

Database: none

Pre-condition: none

Validity checks: none

Post-condition: none

Calls: none

Called by: updateMetadataAction

KeywordView

- none

Description: Creates the objects seen in the keywordview
createPartControl(): void

Name: updateKeywordView

Parameters: none

Return value: none

Description: Updates the picture seen in the keywordview

Database: none

Pre-condition: none

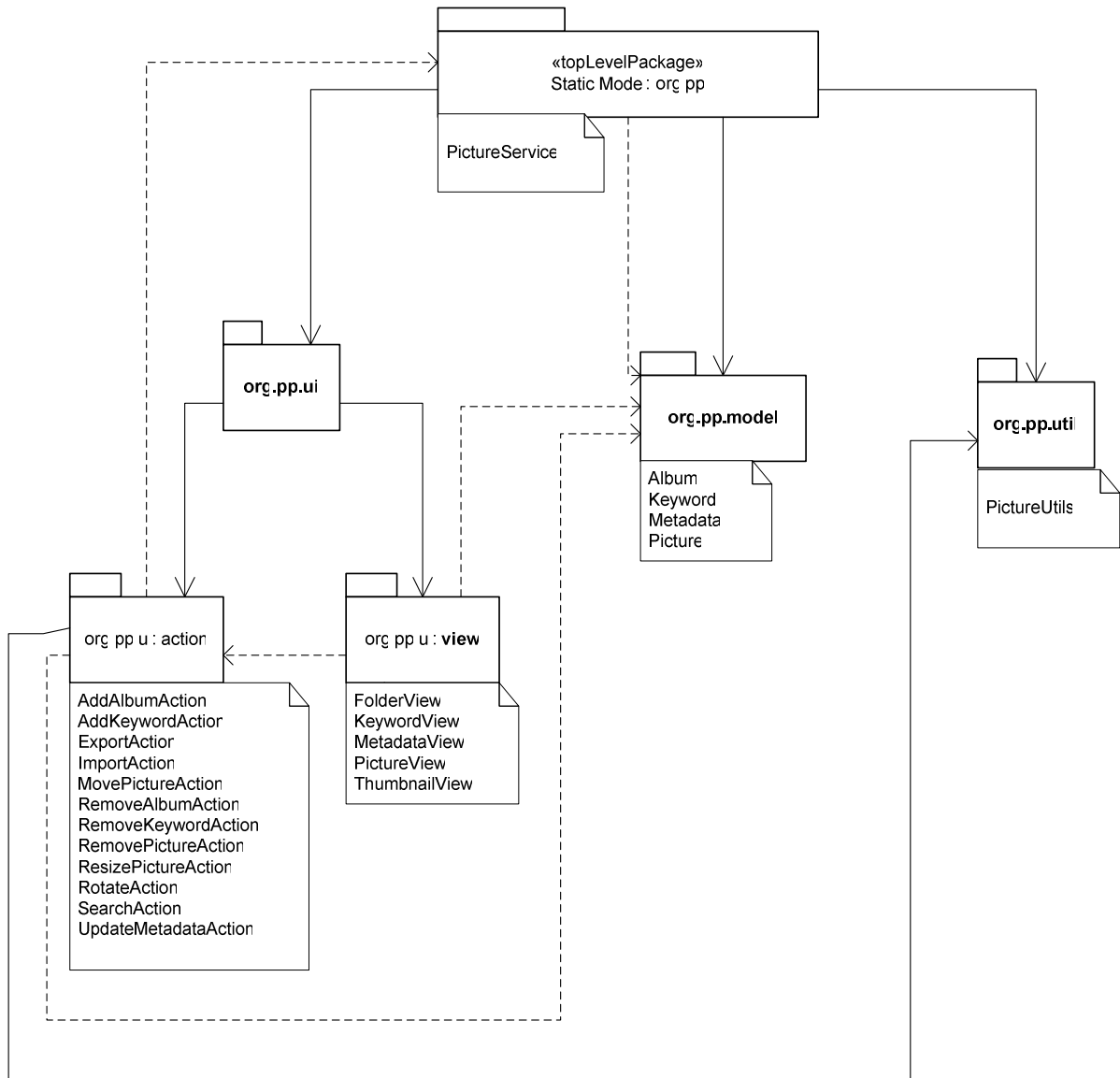
Validity checks: none

Post-condition: none

Calls: none

Called by: addKeywordAction, removeKeywordAction

5.6 Package Diagram



5.7 References to RD

Requirement from the RD	Implementation in the DD
Rotating Picture Page 8	RotateAction Page 34
Picture Storage Page 17	ImportAction Page 34 PictureService Page 37
Picture Downloading Page 8	ExportAction Page 34 PictureService Page 37
Removing a picture Page 9	RemovePictureAction Page 34 PictureService Page 37
User input of metadata Page 9	UpdateMetadataAction Page 34 PictureService Page 37
Sorting Pictures Page 8	AddAlbumAction Page 35 MovePictureAction Page 35 RemoveAlbumAction Page 35 PictureService Page 37
Adding keywords to pictures Page 8	AddKeyWordAction Page 35 PictureService Page 37
Picture Searching Page 9	SearchAction Page 36 PictureService Page 37
System behavior in picture searching Page 19	SearchAction Page PictureService Page 37
Optimization of image quality Page 17	PictureUtils Page 36

6. Functional Test Cases

Item 1 is an explanation of the functionality of the requirement

Item 2 is a reference to where in the RD you can find the requirement

Item 3 is the input that might be needed to test the functionality

Item 4 describes how you can check that the functionality actually happened

After this there is a step by step explanation of how to test the functionality

Picture Downloading

1. The publisher can download a picture from the server through the client.
2. Page 8 Picture Downloading
3. A picture object representing the picture to be downloaded
4. A exact copy of the picture can now be found in the local file system

Pre-condition: The publisher is logged in and using the client

1. Select a picture to be downloaded from the server by clicking on it
2. Press the download button
3. Verify that an exact copy of the picture is transferred into the folder on the local computer that is currently selected

Rotating pictures

1. The publisher can use the client to rotate a picture either clockwise or counter clockwise
2. Page 8 Rotating pictures
3. The picture in the picture view and a direction to rotate
4. The picture is rotated in the specified direction

Pre-condition: The publisher is logged in and using the client

1. Select the picture you want to rotate
2. Click either the rotate clockwise or the rotate counter clockwise button
3. Verify that the picture is rotated 90 degrees in the specified direction

Adding Keywords to Pictures

1. The publisher can attach keywords to a picture
2. Page 8 Adding Keywords to pictures
3. A keyword to add and a picture to add it to
4. A keyword is added in the keyword list

Pre-condition: The publisher is logged in and is using the client

1. Select a picture to add a keyword to
2. Enters the keyword into the keyword field
3. Press the add keyword button
4. Verify that the keyword is shown in the keyword list

Sorting Pictures

1. The publisher can add a picture in an album
2. Page 8 Sorting Pictures
3. An album to add the picture to and a picture to add
4. The picture is added to the album, and if you select the album, the picture is visible

Pre-condition: The publisher is logged in and using the client

1. Select one or more pictures by clicking on the picture(s) (ctrl + clicking if more than one)
2. Drags the pictures into the desired album
3. Verify that you can see the pictures if you select the album you dragged them to

Access Restrictions

1. The publisher can decide if his pictures are public or private
2. Page 8 Access Restrictions
3. A picture to set access restrictions for
4. If a picture is set as private it should not be possible to view it through the webpage

Pre-condition: The publisher is logged in and using the client

1. Select a picture
2. Right click on the picture
3. Click on set "Access Restrictions"
4. Click on the desired restriction type
5. Verify that it is no longer possible to view the picture through the webpage

Account Creation

1. The user can create an account
2. Page 8 Account Creation
3. Alias, password and e-mail
4. An account that can be used to log in to the Picture Publisher system

Pre-condition: The user is browsing the Picture Publisher webpage

1. Click on the sign up now button
2. Fill in the alias, password and e-mail fields
3. Verify that you can now log in

Viewing Pictures

1. Anyone is able to view uploaded pictures that are not access restricted. This is done on the website
2. Page 8 Viewing Pictures
3. A search string (something that exists so that the search actually returns a result)
4. A picture

Pre-condition: Browsing the Picture Publisher website

1. Search for a picture
2. Click one of the thumbnails from the search result
3. Verify that a picture is displayed

Picture searching

1. You can search for pictures and when searching for pictures the pictures returned match the search string
2. Page 9 Picture Searching
3. A search phrase
4. The pictures that match the search phrase are displayed

Pre-condition: The user is logged in

1. Enter the search phrase into the search field and press enter
2. Clicks on one of the thumbnails in the result
3. Verify that you can find the search phrase in either the metadata or the keywords

User input of metadata

1. The publisher can input meta data about the picture
2. Page 9 User input of metadata
3. The metadata to add to the picture
4. The metadata is changed and you can see the updated metadata in the metadata field

Pre-condition: The user is logged in

1. Select a picture
2. Click on one field in the metadata view that you wish to change
3. Enter the changes (or erase the incorrect data)
4. Press enter
5. Select another picture and then reselect the picture you changed metadata for
6. Verify that the metadata is still changed

Removing accounts

1. You can remove your own account
2. Page 9 Removing Accounts
- 3.
4. The account is removed and can no longer be used to log in. The pictures added by that account is deleted

Pre-condition: The user is browsing the webpage and is logged in

1. Click on the remove account button
2. Confirm that you really want to remove your account
3. Verify that you can no longer log in using the removed account and that you can no longer find the pictures you added with that account

Removing a picture

1. The publisher can remove a picture that he has previously published in the system
2. Page 9 Removing a picture
3. The picture to remove
4. The picture is removed and can no longer be seen, neither on the web or by browsing the server through the client

Pre-condition: The user is logged in and using the client

1. Select the picture you want to remove from the server
2. Click the “delete”-button
3. Verify that you can no longer see the picture on the server, also go to the webpage and perform a search that should return the picture and verify that it is not displayed

Download restrictions

1. The publisher can decide if it shall be possible to download the original picture or not
2. Page 9 Download restrictions
3. The picture to set restrictions for and what restriction to set
4. If a restriction is set you can no longer download the original version of that file

Pre-condition: The user is logged in and using the client

1. Select a picture that is on the server
2. Right click on the picture
3. Select “Download restrictions”
4. Set it to the desired restriction (yes if it is possible to download the original no otherwise, default is no)
5. Go to the Picture Publisher webpage and verify that you can no longer download the original version of a picture

Picture Storage

1. When you upload pictures an exact copy of the picture uploaded is stored
2. Page 17 Picture Storage
3. A picture to upload
4. The picture is stored on the server

Pre-condition: The user is logged in and using the client

1. Select one or more pictures in your local file system by clicking on it (holding down ctrl + clicking if you want more than one)
2. Press the upload button
3. Verify that an exact copy is now stored in the directory on the server that is currently selected

Upload quota

1. The administrator can decide how many megabytes a user is allowed to upload
2. Page 9 Upload quota
3. What the quota should be
4. When the quota is reached the publisher should no longer be able to upload more then the specified amount

Pre-condition: The user is logged in and using the client

1. Check how much space you have left (you can see this under the servers folder view)
2. Upload some large pictures until you have reached the quota
3. Verify that you can no longer upload pictures when you would have exceeded the quota

System behavior in Picture Searching

1. When searching for pictures the system look at the pictures metadata and keyword
2. Page 17 System behavior in picture searching
3. A picture that you have attached keywords to and a picture you have changed the metadata for
4. When you search for the keywords you attached the picture should be found and when you search for the metadata you changed the picture should be found

Pre-condition: The user is logged in and using the client and there exist uploaded picture(s)

1. Select a picture on the server, we'll call this picture p1
2. Add the keyword "test16" to the selected picture
3. Select another picture on the server, we'll call this picture p2
4. Change the metadata field "Manufacturer" into "PPTest"
5. Search for "test16" and verify that p1 is in the search result
6. Search for "PPTest" and verify that p2 is in the search result

Separation of keywords

1. The system keeps track of which keywords are added by publishers and which keywords have been added by other users
2. Page 11 Separation of keywords
3. You need two pictures, one that has a keyword attached by a publisher and one that has the same keyword attached by a viewer
4. If you choose to only search for pictures tagged by publishers only that picture should be visible

Pre-condition: The user is logged in and using the client and there exist uploaded picture(s)

1. Select a picture on the server by clicking on it, we'll call this picture p1
2. Add the keyword "test" to it
3. Go to the Picture Publisher website
4. Log in
5. Find a picture (not the same one as you used in the client), we'll call this picture p2
6. Add the keyword "test" to that picture
7. Search for test and verify that both p1 and p2 are found
8. Uncheck the "Use viewer keywords"
9. Search for test again and verify that only p1 is found

Optimization of image quality

1. The picture displayed on the website is optimized for viewing in a web browser
2. Page 17 Optimization of image quality
3. A picture that is larger than 800x600
4. If you look at the picture on the website it is smaller

Pre-condition: The user is logged in and using the client

1. Upload a picture that is larger than 800x600, we'll call it p1
2. Add the keyword "test18" to the picture
3. Go to the Picture Publisher website
4. Search for "test18"
5. Click on p1
6. Verify that p1 is now smaller