# Picture Box Design Document

By: Maximilian Mihoc

Student Number: C12728559

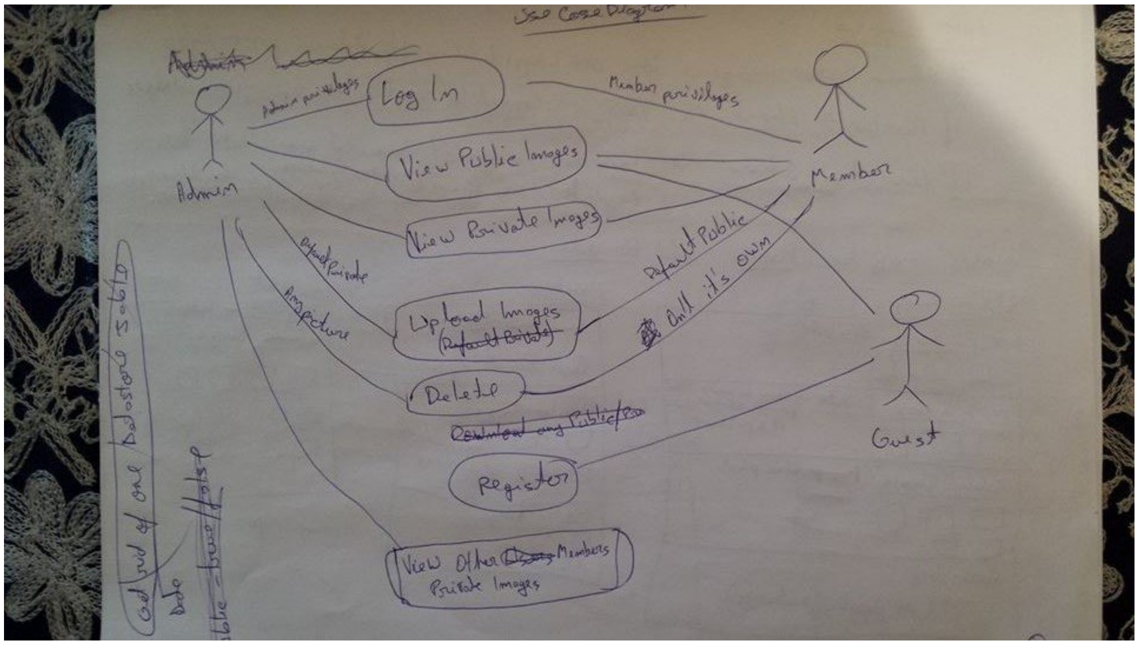
## About Picture Box

Picture box is a web application for uploading and sharing pictures with other members. This application is available on google app engine and can be run from the following link:

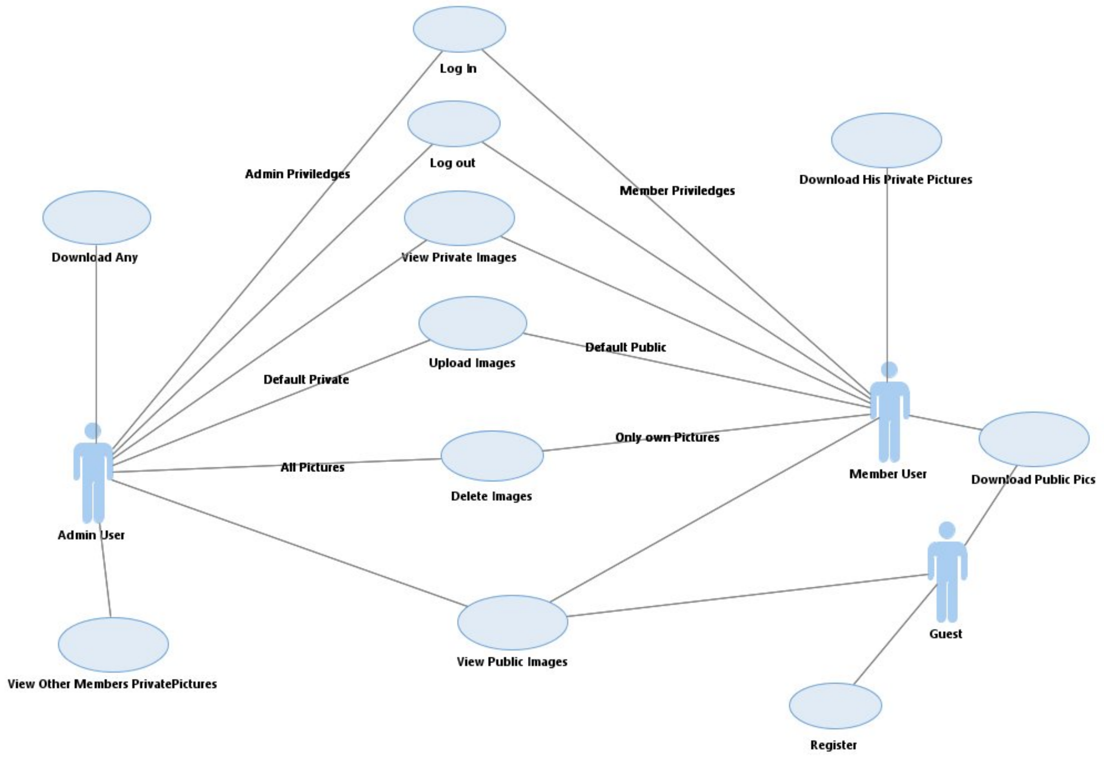
http://ieditmihocmaximilianpicturebox.appspot.com/

## Design

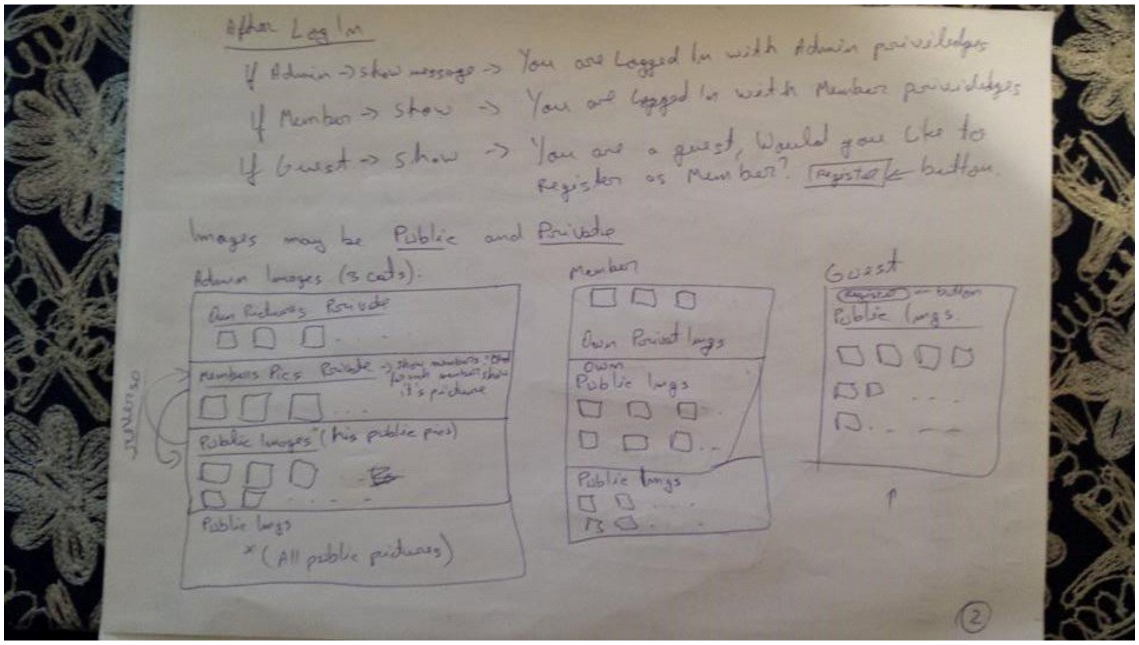
When I first started this application, I defined the most important requirements that I was going to implement on paper using a use case diagram and this is how the first it looks like:



This diagram contains the possible users for my application (Admin, Member or Guest) and the features that the application. Later I created that use case diagram using RSA (Rational Software Architect) and I implemented a more complex use case diagram to represent my requirements.

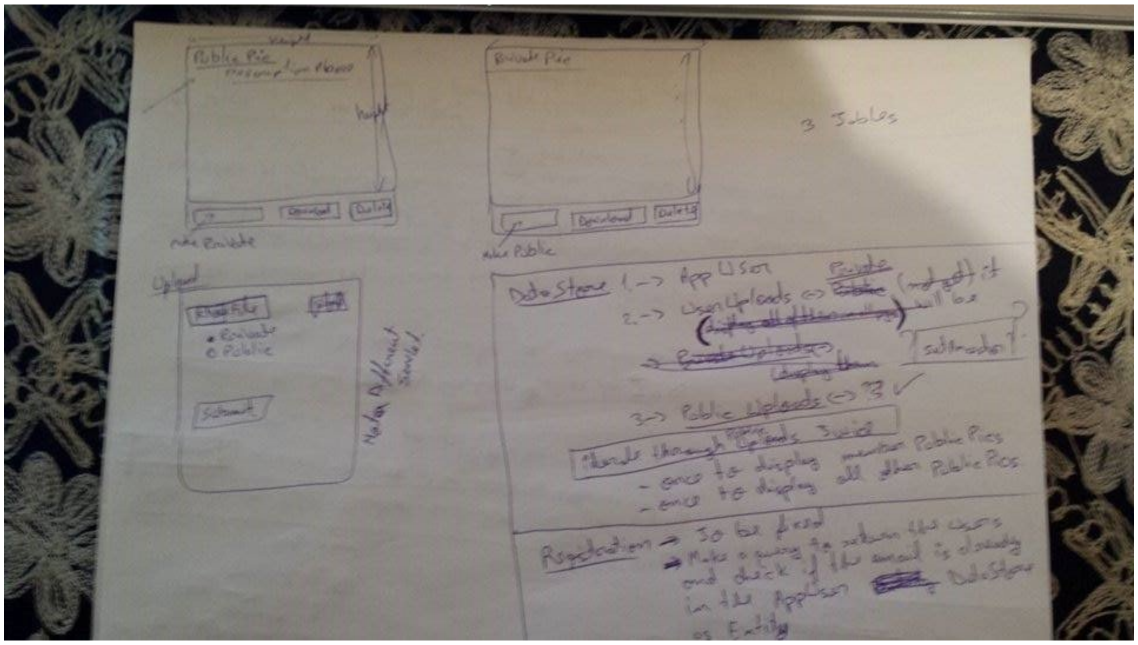


Tacking this diagram into consideration, I designed how I want the user interface for my application to look like on paper. Only the main pages from my application are designed on paper: admin page, member page and guest page.


After Log In
If Admin -> show message -> You are Logged In with Admin priviledges
If Member -> show -> You are Logged In with Member priviledges
If Guest -> show -> You are a guest, Would you like to Register as Member? Register <- button.
Images may be Public and Private
Admin Images (3 cats).
Own Pictures Private
Members Pics Private -> show members' (for each member) show it's picture
Public Images (his public pics)
Public Imgs
*(All public pictures)
Member
Own Privat Imgs
Own Public Imgs
Public Imgs
Guest
Register - button
Public Imgs.
2

First diagram represents admin page, second diagram represents member page and third diagram represents guest page.

On each page I had to display private and public images and I decided that each should be represented like in the following diagram:



## Features of the application

The above use case diagram is representing all the features that are achieved in the application.

- There are three classes of users
  1. Administrator
     - Can Log in with Administrator privileges
     - Can Log out
     - Can View all Public and Private pictures
     - Can upload Pictures (Default Private)
     - Can Delete any Picture
  2. Member
     - Can Log in with Member Privileges
     - Can Log out
     - Can view all Public Pictures and only his Private Pictures
     - Can upload pictures (Default Public)
     - Can delete only his pictures (Private or Public)
  3. Guest
     - Does not have to log in
     - Can view Public Pictures
     - Can choose to Register

## Code Explanation

What I think it was the hardest part in my application was realising how to store pictures as private and public and how to map this pictures with different users. To do this I created 3 Kinds in google app engine Datastore: AppUser, UserUpload (for private pictures) and Public Uploads (for public pictures).

AppUser kind is containing the members of my application, which will have email address and username. I was thinking to create a separate kind for administrators, but because I only have 2 administrators, I decided not to do that. For administrators, I created a list of strings where I am storing their email address. Based on the email address, the administrators will be identified when they log in into the application.

UserUpload Kind is containing the private pictures. Each picture will have a description, a blob key, and a user, which is the owner. When a user is uploading a picture, he can choose to upload that picture as private and in this case, the picture will be stored in this table.

Public uploads kind is the same as user upload but this is containing the public pictures. If a user wants to upload public pictures, this is where they will be stored.

The code that is deciding where to upload a picture, UserUpload or Public Upload, is in Upload class. When the user is choosing one or more files to upload and in clicking the button, upload file, the code from upload class will be executed. The files keys will be saved in a list of blob keys and for each key, new Entities will be created with 3 properties: user (owner), description and blob key. The new entity will be either a UserUpload or PublicUploads, depending if the user choose to be private or public.

In order to make difference between users, I creates 3 separate classes for 3 separate classes of users. So my application is containing an Admin class, a Member class and a guest Class. The difference between these 3 types of users is in the start of the application. If someone wants to use the application without logging in, he is considered to be a guest and Guest class will be executed. If a user will log in into the application, his email will be checked to see if it is an admin or a member. Based on that, Admin or Member classes will be executed. The reason I decided to use 3 separate classes for that is because for each user should be implemented a piece of functionality and they can be well represented in this way.

In each class, I am building some queries which will return information from the GAE datastore. For example, In Admin class, the content of both kinds, UserUpload and PublicUploads have to be returned, because Admin should be able to see all the pictures. The content of UserUpload will be stored in a list of maps called uploads and the content of PublicUploads will be stored in a list of maps called publicUploads. These 2 lists are placed into session, and they will be used in the jsp page to be displayed to the user, using JSTL. In the Member class, from the UserUpload Kind, only the pictures associated with the user will be returned, using a filter when the query is created, and placed in a list. The content of PublicUploads will be returned in the same way as in Admin clss.

Delete Class it is used to delete pictures from the application. When a user press the delete button for a picture, a string with the blob key will be send to the delete class for deletion.

Register class is used if a user wants to use my application and he is not registered. If he choose to register, he can click on the registration link available in guest page, he will have to write a username and his email address and click on register button. His details will be returned to the Register class, a check is going to be made to see if his email already exists in the database and if not, he will successfully register and redirected to member page. If his email already exist in database, user won't be registered again and he will still be redirected to member page. In order to check if the email already exists, I create a query to return all emails from the AppUser kind on GAE, put all emails in a list of strings and check if the new email is in there.

In order to get information in the JSP pages, the lists were placed into session and retrieved from there using JSTL tags. Using these tags I was able to display the lists contents.

# Implementation

## Build

The application is available on GitHub here: https://github.com/MaximilianMihoc/dt2283cloud-CA2

In order to build this application, a git clone command should be used like the following: git clone https://github.com/MaximilianMihoc/dt2283cloud-CA2.git

After the repository was cloned, make sure the current branch is develop branch, this is the branch that will contain all the code. If for some reason the current branch is another one, the checkout and pull commands should be used.

```
git checkout develop

git pull
```

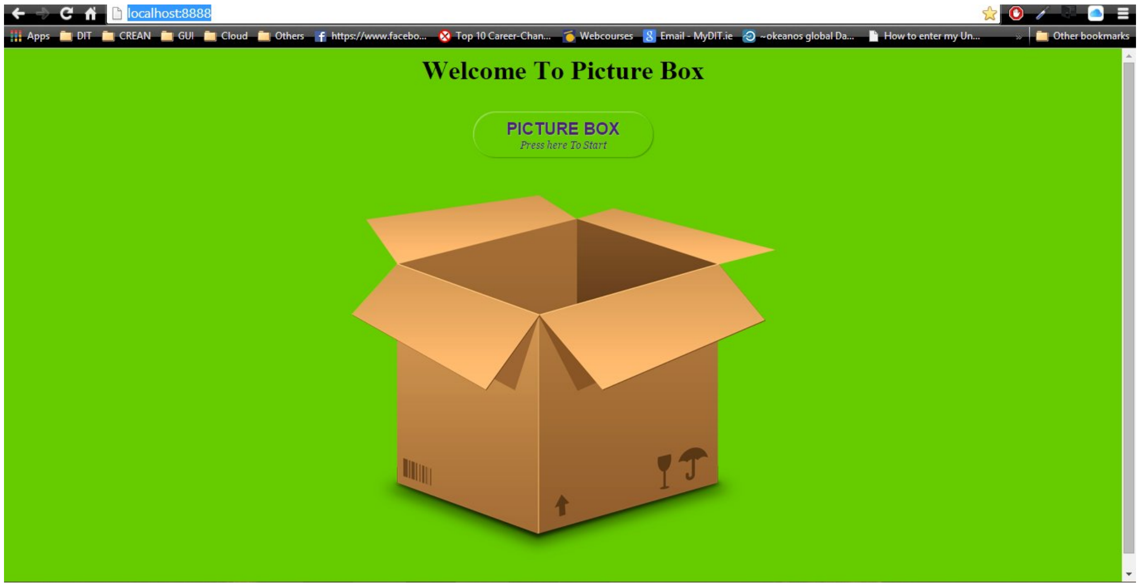
After this is done, the complete code should be in a new folder called dt2283cloud-CA2 and inside that can be found a Documentation folder and the Picture box Folder.

## Deploy

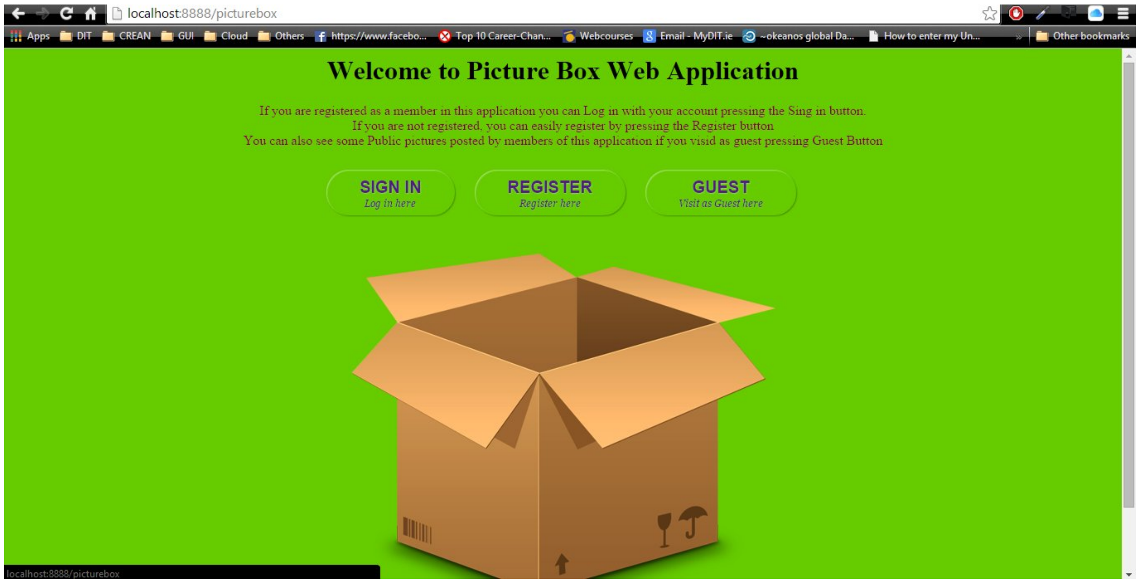
After the above steps are completed, the Application can me imported into eclipse. Eclipse will need google plugin installed and a java JDK version 7 in order to run. Import project by clicking File > Import, from there choose to import a General Project, Existing Projects Into Workspace and import the project that was just downloaded from GitHub. After this step, all code should be available in eclipse.

## Test Plan

In order to test this application, it will be required to be run as local application by right click on the application in eclipse, Run as > Web application. After this step, the application should be available in http://localhost:8888/ and the following page should be displayed:



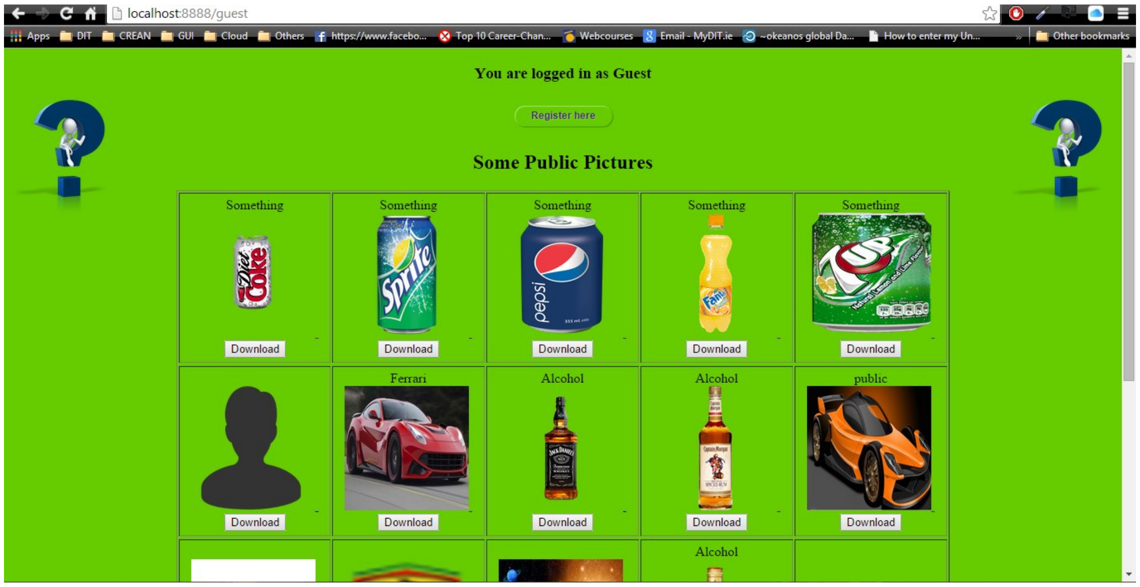
This represents the first page of my application, it is what a user will see first time when he will interact with the application. Pressing The Picture box button, the application will start and the user will be redirected here:


localhost:8888/picturebox

Welcome to Picture Box Web Application

If you are registered as a member in this application you can Log in with your account pressing the Sing in button.
If you are not registered, you can easily register by pressing the Register button
You can also see some Public pictures posted by members of this application if you visid as guest pressing Guest Button

Here, the user can login as member or admin, he can visit the page as guest or he can register if he is not yet registered. There is no need for registration if the user wants to visit the page as guest. By clicking on the Guest button, he will be redirected to the guest page where he will be able to see public pictures posted by other users that can be also downloaded. To see a picture in normal size, the user can click on the picture.


localhost:8888/guest

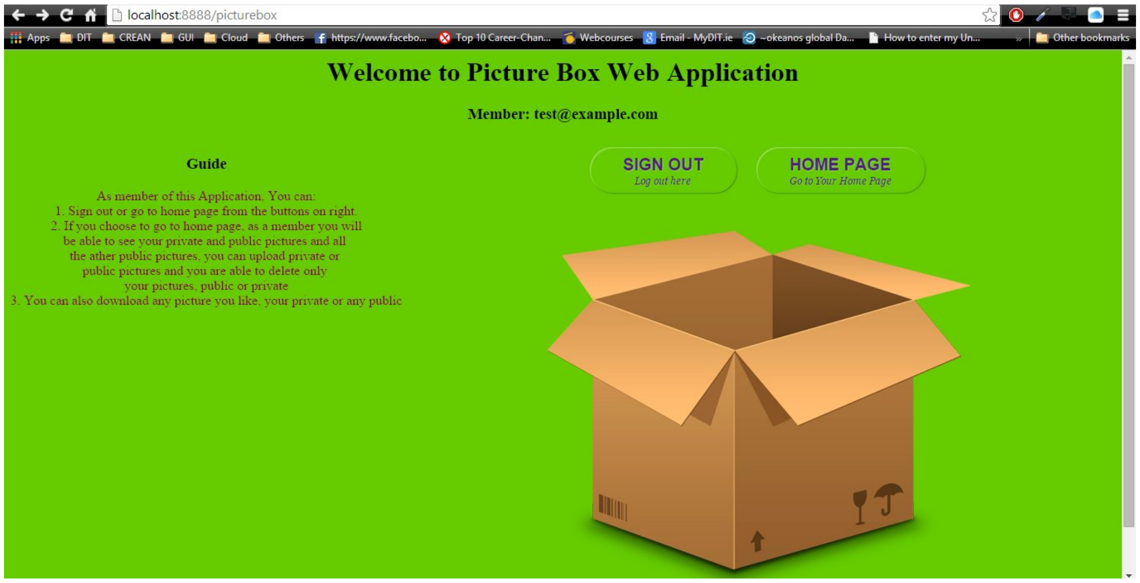
You are logged in as Guest

Some Public Pictures

| | | | | |
|---|---|---|---|---|
| Something<br>Download | Something<br>Download | Something<br>Download | Something<br>Download | Something<br>Download |
| Download | Ferrari<br>Download | Alcohol<br>Download | Alcohol<br>Download | public<br>Download |
| | | | Alcohol | |

If a user is already registered for this application, he can log in with his credentials and he will be redirected to member page, if he is a member, or admin page, if he has an admin account.

When he logs in with a member account, he will see the following page. The user with email address test@example.com, is registered as a member of this application on localhost. This registration can be done in the register page.


localhost:8888/picturebox

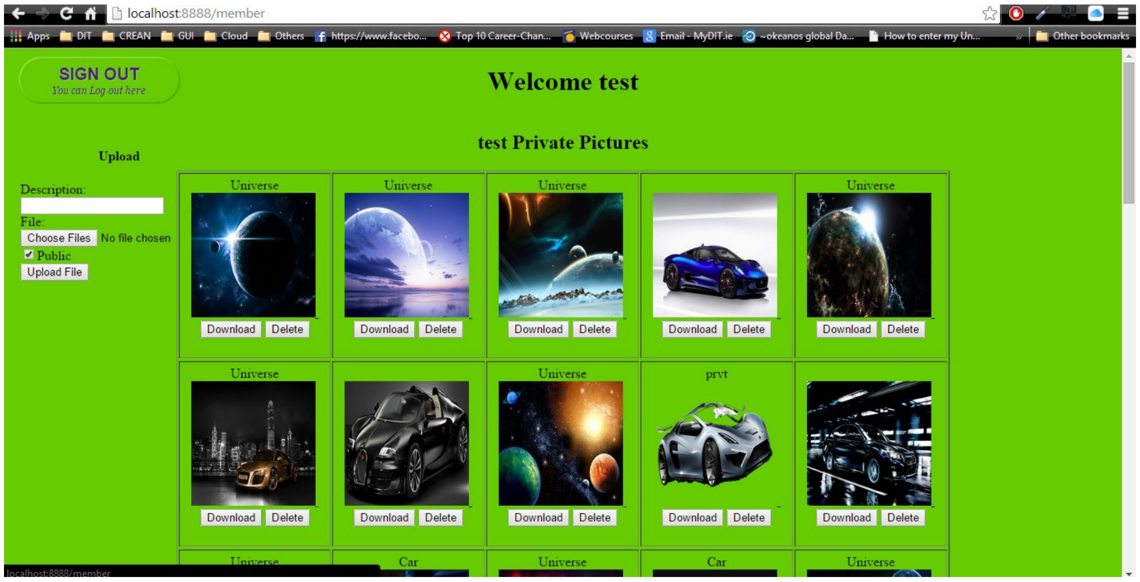
Welcome to Picture Box Web Application

Member: test@example.com

Guide

As member of this Application. You can:
1. Sign out or go to home page from the buttons on right.
2. If you choose to go to home page, as a member you will be able to see your private and public pictures and all the ather public pictures, you can upload private or public pictures and you are able to delete only your pictures, public or private
3. You can also download any picture you like, your private or any public

In the right, the user can see some guidelines that will give him some hints on what he can do in this application. By clicking the Home Page button, he will be redirected to his page, where he will be able to see hi private and public pictures, if he has, and he can also see some other public pictures posted by other members. His member page will look like this:


localhost:8888/member

Welcome test

test Private Pictures

Upload

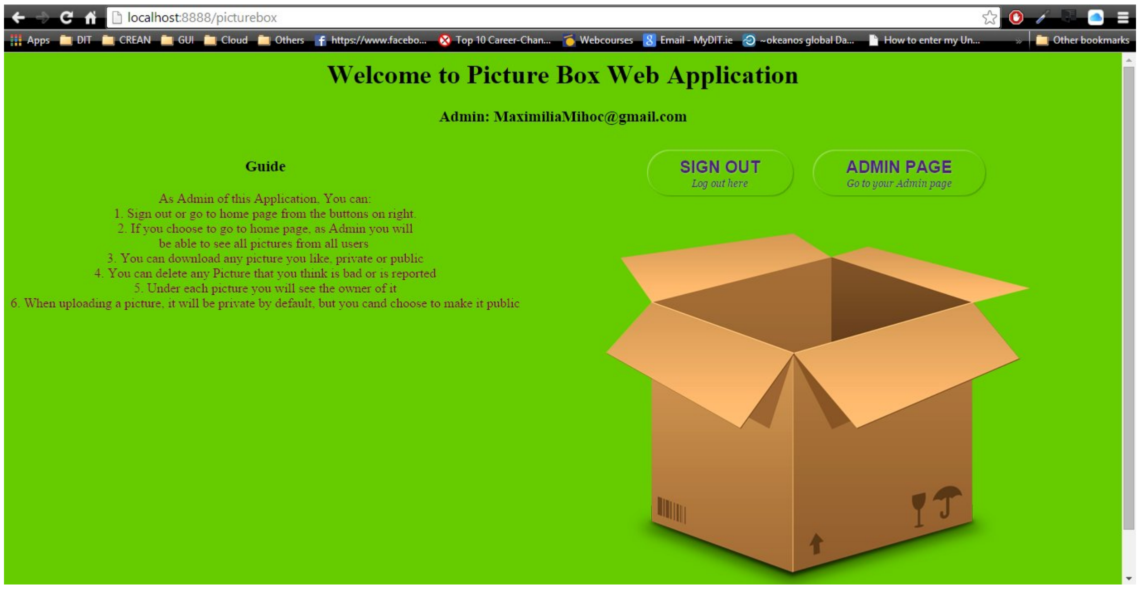
Description:

File: Choose Files No file chosen

Public [✓]

A member can upload one or many pictures, public or private, can delete only his public or private pictures and he can also download pictures.

If the user will log in with an administrator account, he will be redirected to another page, where he will be able to do all things a member can do, plus some extra. Logging in with MaximiliaMihoc@gmail.com or mark.deegan@dit.ie the user will be logged in as administrator.



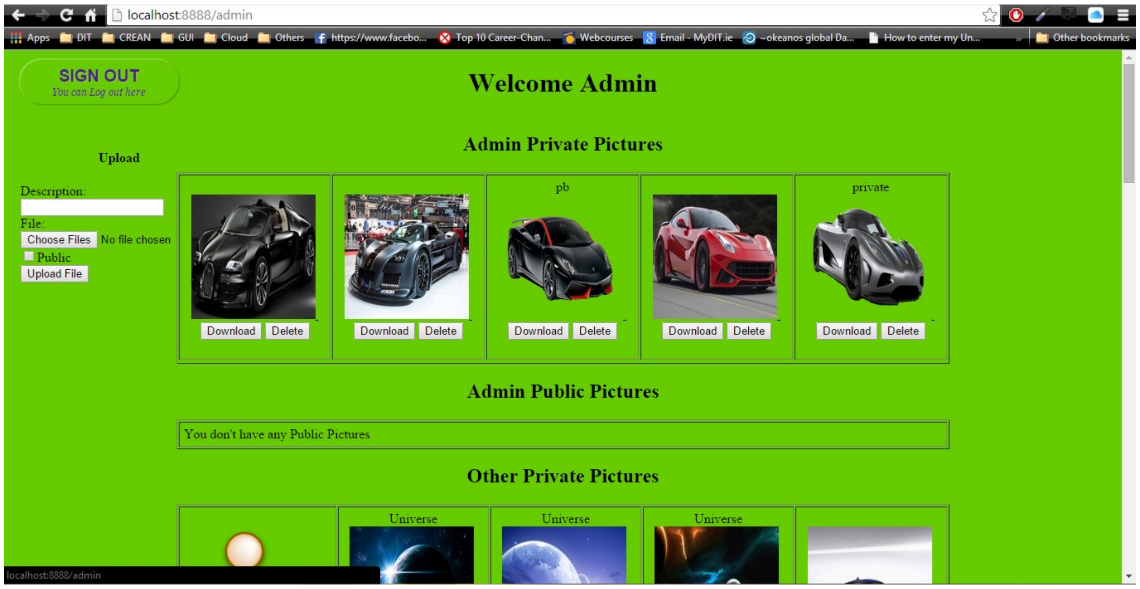
Welcome to Picture Box Web Application

Admin: MaximiliaMihoc@gmail.com

Guide

As Admin of this Application. You can:
1. Sign out or go to home page from the buttons on right.
2. If you choose to go to home page, as Admin you will
be able to see all pictures from all users
3. You can download any picture you like, private or public
4. You can delete any Picture that you think is bad or is reported
5. Under each picture you will see the owner of it
6. When uploading a picture, it will be private by default, but you cand choose to make it public

Some guidelines are presented in the left hand side explaining what and how the user can do things. By clicking the admin page button in the right, user will be redirected to admin page where he can do whatever he wants. As administrator, it will be possible to download any picture, delete any picture, own picture or a picture uploaded by another member, and upload public or private pictures. An admin can also see the other users private or public pictures in this page, as I mentioned before, and under each picture, he can also see the owner of it, in case is needed.



Welcome Admin

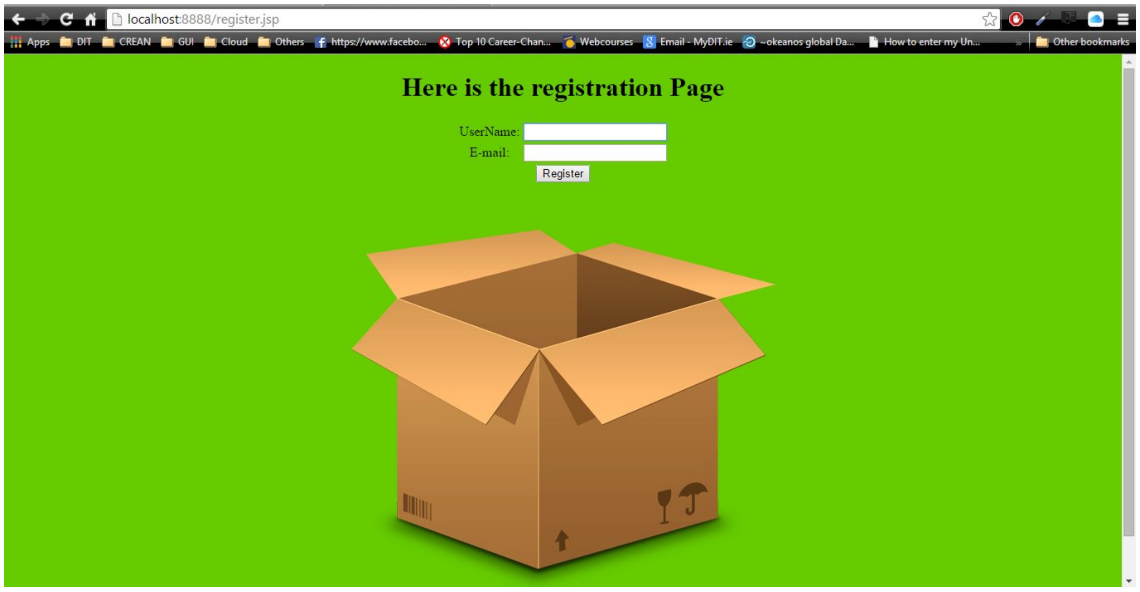
Admin Private Pictures

Admin Public Pictures

You don't have any Public Pictures

Other Private Pictures

If a new user comes in and he wants to register, the registration page can be easily accessed by clicking the register button and the following page will be shown:


localhost:8888/register.jsp

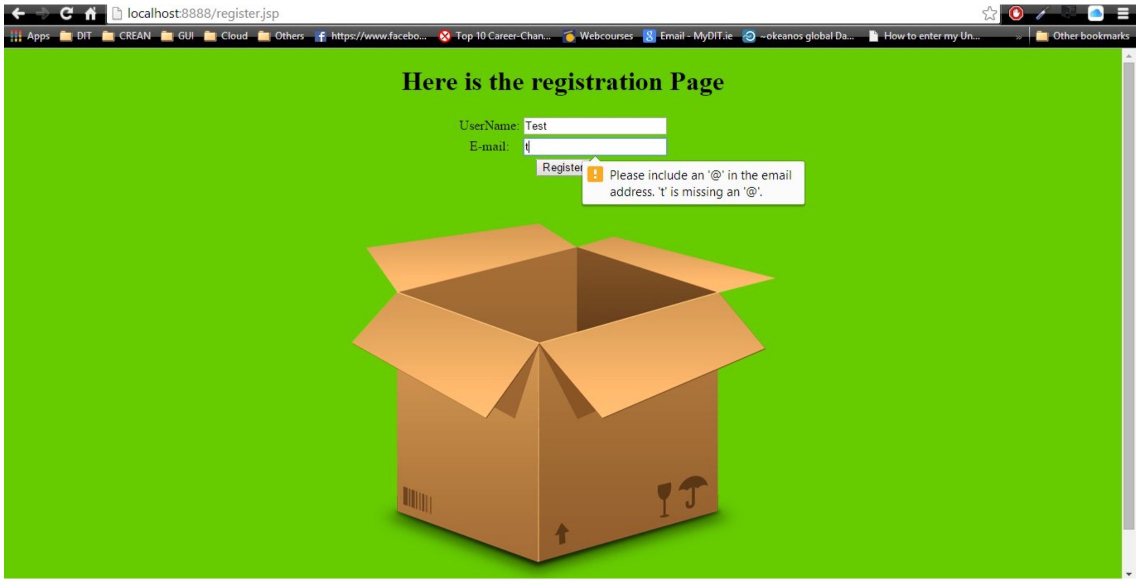
Here is the registration Page

UserName:

E-mail:

Username and email address will be required in order to register with this application. Some error checking are included here so that the user will have to enter a valid email address. If a user that enters his email address and username, is redirected to the main page, The Portal, after he clicks the register button, it means that the registration has completed successfully.

For example if a user will enter the user name Test, and the email address t, he will see the error:


localhost:8888/register.jsp

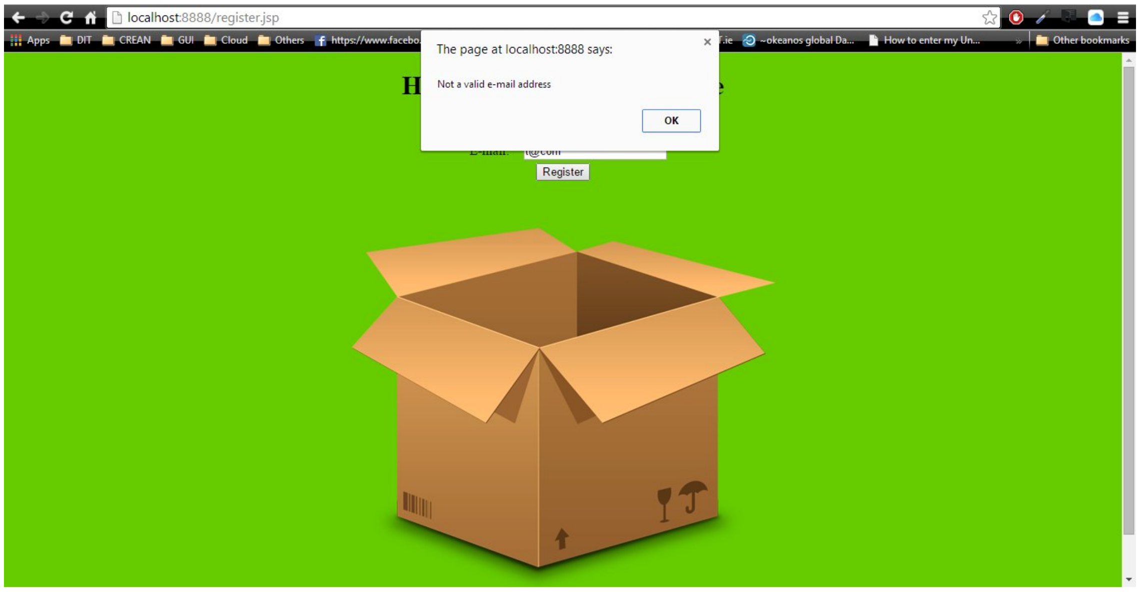
Here is the registration Page

UserName: Test

E-mail: t

Please include an '@' in the email address. 't' is missing an '@'.

For username Test and email t@com, he will get another error:



The page at localhost:8888 says:

Not a valid e-mail address

OK

## Git Hub

GitHub was used during the development of this application. In GitHub, in the repository dt2283cloud-CA2 can be found all code and some documentation of this application.

https://github.com/MaximilianMihoc/dt2283cloud-CA2

During development, the develop branch was used as a master branch of this application. Every time when a new piece of functionality was added to the application, another branch was created from the develop branch, and when the functionality was working, that branch was merged back into the develop branch and so on.

## References

- The design of the buttons is from: http://designmodo.com/create-css3-buttons/
- The book “Programming Google App Engine, 2nd Edition” by Dan Sanderson, was used for documentation and implementation.