# Introduction

Today there are relatively many mobile applications for expense tracking. It goes without saying, that one of the core features that all the expense tracking applications have is a possibility to manually submit basic expense information (eg. total sum, merchant/company etc.). But with the development of modern technologies, some expense tracking applications introduced an opportunity to take pictures of purchase receipt in order to automatically recognize and extract basic expense information from the image and place the extracted information into user’s expense list inside the application. From the users’ perspective, such functionality may be quite useful, as it prevents them from manual expense submission in favor of automatic receipt processing. The most well-known examples of applications of such kind are Expensify[1] and Xpenditue[2]. But at the same time, these applications are proprietary and what is more, they are not free of charge when it comes to extensive usage of expenses extraction from receipt image. By the time the author started with this thesis, he has not found any noted open-source expense tracking software with the functionality of expense recognition from a receipt image. Because of this, the author decided to implement such mobile application, which would allow expense data recognition from images of receipts issued in Estonia, using open-source technology stack.

## Problem and background

The main problem the author focuses in this work is to find out the best designing solution for expense tracking mobile applciation, focusing on possibility of expense information recognition from Estonian receipts as well as usability and security aspects of the applicaiton.

The reason why this work may be considered useful for the broader audience is that as a result of it appered a mobile application, which can simplify users’ everyday expenses tracking and thus to help planning personal budget more sensibly.

The knowledge gained during development of this applciation and pointed out in this work may be useful from the perspective of a software engineer, as it covers the following topics:

* Comparison of potential solutions regarding the architecture and design of such kind of application
* Methods of receipt digital image processing in order to improve recognition of the text of receipt
* Reliability evaluation of the application’s function of expense information recognition from the aspect of Tesseract for OCR and OpenCV for image preprocessing usage.

## Goals

The main goal of this work is to implement a mobile application with the following functionality:

1. Possibility to take picture of a receipt, recognize total sum of the purchase and the company the purchase was from based on the receipt image, compose an expense entry from this information an place it into user’s expense list.
2. Possibility for the user to see his own expenses.
3. Possibility to insert expense entry manually, which can be useful in case recognition of expense from receipt has failed due to some reason.
4. Possibility to make pictures of receipts while device is not connected to the internet and make expense recognition from those pictures later.
5. Graphics? (to be implemented)

## Methodology

To achieve the goal the author implemented the Android client-side application and the server side application written in Java programming language. Both client and server are designed in an object-oriented manner. The server side is put to communicate with the database (PostgreSQL) and an external web serice of the e-business register. Both service and client are also using third-party libraries.

## Thesis overview

//to do: add when is ready.

# Theory

## Image preprocessing before optical charecter recogniton

In the author’s mobile application, the picture of receipt used for further exepnse information extraction is taken using the mobile device’s camera. As not all the mobile devices have high resolution cameras, this means that the quality of the captured receipt image itself may not be good enough to perform successful optical charecter recogniton. Besides that, the recognition stage is made even harder by the fact that cash registers use mainly either thermal printers with thremal paper or dot matrix printers for receipt printing. The problem of the receipts printed with the thermal printer is that they tend to fade with time, and the problem of the dot-matrix-printed receipts is that the charecters are composed of small dots, which makes it hard for the OCR engine to determine them correctly. As a result, in order to enchance the quality of the image with the purpose of better OCR results, image preprocessing should be applied.

### Conversion to grayscale

The first measure to be taken is order to enchance image for further OCR is converting color image to grayscale image. Grayscale image is an image in which the value of each [pixel](http://en.wikipedia.org/wiki/Pixel) it carries only [intensity](http://en.wikipedia.org/wiki/Luminous_intensity) information. Images of this sort, also known as [black-and-white](http://en.wikipedia.org/wiki/Black-and-white), are composed of shades of [gray](http://en.wikipedia.org/wiki/Grey), varying from black at the weakest intensity to white at the strongest. (Stephen Johnson (2006). [*Stephen Johnson on Digital Photography*](http://books.google.com/books?id=0UVRXzF91gcC&pg=PA17&dq=grayscale+black-and-white-continuous-tone&ei=XlwqSdGVOILmkwTalPiIDw). O'Reilly. [ISBN](http://en.wikipedia.org/wiki/International_Standard_Book_Number) [0-596-52370-X](http://en.wikipedia.org/wiki/Special:BookSources/0-596-52370-X)) Conversion of the image to grayscale is important from the perspective of making further image processing stages faster.

([http://research.ijcaonline.org/volume55/number10/pxc3882784.pdf p. 52](http://research.ijcaonline.org/volume55/number10/pxc3882784.pdf%20p.%2052)).

### Denoising

The second stage of preprocessing is denoising. Noise is considered to be either error in the pixel value or an erroneous bit pattern with no significance in the output image, which is takes place during the acqusition process of the image. The noise may be amplified by the digital corrections of the camera or tools removing blur or increasing contrast of the images.

The most trivial denoising method is replacing the color of the pixel with an average of the colors of nearby pixels. But in practice it does not often lead to the desired result, as similar pixels are not always close to each other.  Thus it proves to be more reliable to scan a vast portion of the image in order to find all the pixels that really resemble the pixel to be denoised. Denoising is then done by calculating the average color of these most similar pixels. The similarity is evaluated by comparing a whole window around each pixel, and not just the color. The filter performing such kind of operation is called non-local means filter (Antoni Buades, Bartomeu Coll, and Jean-Michel Morel, Non-Local Means Denoising, [Image Processing On Line](http://dx.doi.org/10.5201/ipol), [1](http://dx.doi.org/10.5201/ipol.2011) (2011).<http://dx.doi.org/10.5201/ipol.2011.bcm_nlm>). This filter is applied by the author in his practical part of the work using the OpenCV library.

### Morphological closing

As it was mentioned above, the quality of receipt print is quite frequently of low level. It is particularly reflected in the fact that some of the printed symbols contain gaps, which isolate parts of a single symbol. As a result, the OCR engine tend to misinterpret separate parts of a single symbol as separate symbols. For the puprose of linking the parts of a symbol separated with the gaps, an operation of morphological closing(<http://docs.opencv.org/doc/tutorials/imgproc/opening_closing_hats/opening_closing_hats.html>) should be applied.

This operation is comprised of two basic morphological operations: dilation, followed by erosion.(http://docs.opencv.org/doc/tutorials/imgproc/erosion\_dilatation/erosion\_dilatation.html) As a result of applying morphological closing, parts of the symbol which were separate before tend to become connected as well as dots inside the symbols tend to get removed.

<http://opencv-python-tutroals.readthedocs.org/en/latest/py_tutorials/py_imgproc/py_morphological_ops/py_morphological_ops.html>

### Thresholding

The operation of thresholding is considered to be a simple method of image segmentation. It takes a grayscale image as an input and transforms it into a binary image, comprised of only two colors, according to the following logic: if the value of a pixel is below the threshold value, then the pixel is assigned the minum value (e.g. black), otherwise the maximum value is assigned to the pixel (e.g white).

The most trivial way of thresholding is global thresholding. It is done by specifying a global threshold value and comparing each pixel of the image with this value. However, this approach proves not to be consistent in most cases, as light distribution on the image is usually uneven – one part of the image is lighted, while another part remains dark. Therefore the best approach would be so called adaptive thresholding, which would use separately calculated threshold values by statistically checking the intensity values of the of the surrounding pixels of each pixel in specified area of the image.

As a result, different threshold values are calculated for different regions of the image and this gives better results for images with varying illumination. The stage of threshholding is an important stage of preprocessing before OCR, as it creates black-and-white image ready to serve as input for recognition by the OCR engine.

## Optical character recognition

Optical charecter recognition, or simply OCR, is the machine recognition of printed or handwritten characters from an image. As a rule, OCR systems can recognize text with different fonts, both typewriter and computer printed characters, but some complex OCR systems can even recognize handwritten text. (http://www.pcmag.com/encyclopedia/term/48267/ocr)

There are various approaches used for OCR engines design:

* Matrix matching: Each charecter of the image is represented as a pattern and is compared to stored glyphs. Such kind of recognition is not suitable in case text with different fonts can be found in the same image.
* Feature extraction: Each character is represented by a set of features (height, width, density, lines, loops etc.). The absence or presence of exact features are used to determine the character.
* Neural Networks: The main idea of this approach is to train an artificail neural network with a set of training data input. //todo add

(<http://www.cisjournal.org/journalofcomputing/archive/vol4no6/vol4no6_2.pdf> )

# Implementation

## Choice of the architecture

Before the start of the implementation the author pointed out 3 possible ways of implementing the mobile application, all having their pros and cons:

1. Develop a mobile application with all the functionality and business logic in it. In such case there is no server side at all. All the stages of images preprocessing before OCR, OCR itself, expense information extraction from the OCR result as well as storage of the user expenses is done on the device where the application is installed.

Pros of this solution:

1. No need to implement server side and thus its is to possibile to avoid additional complexity in the development process. There will be no need to design server-based authentication mechanism, session handling and data retrieval logic because all the functionality would be in the mobile application itself and all the data would stored locally on the device.
2. The mobile application does not need internet connection, as all the functionality of it would be available offline.

Cons of this solution:

1. As all the user’s expense entries are stored on the device only, there is no way for the user using multiple devices to have a single account with all of his expenses present on all the devices automatically. In order to equally keep track on all of his expenses on multiple devices, user would have to manually duplicate the expense entries, which is very inconvenient.
2. If, for example, the device storing all the user’s expenses data becomes unusable or if a restore of device’s factory settings is done, then all the user’s expenses data would get lost, which is definitely an example of bad user experience.
3. As all the receipt image preprocessing stages and OCR are performed on the device, it implies that the application will have to include dependencies for both image preprocessing and OCR libraries, which will make application size much bigger (according to the authors estimations, up to 60 mb). Large application size may prevent users from downloading and installing it.
4. If in the future there is a plan to introduce support for the other mobile platforms besides Android (e.g. iOS, Windows Phone), there might occur problems, as there is no guarantee that the libraries used in Android can be easily integrated into the application for other platforms in the same way.
5. Develop a mobile application (client) and a server the mobile client would communicate with, however in this case the client is still “fat”, i.e. it still contains a large part of business logic. The process of image preprocessing before OCR, OCR and expense information extraction from OCR result would be still performed on the client side, while the server would be used for storing and retrieving expense information.

Pros of this solution:

1. User may have a single account and use it with numerous devices and all the expenses data is synchronized with the server. There is no threat of losing the expense data as in case of the previous solution.
2. As the receipt image preprocessing and OCR with the following expense information extraction takes places on the client side, there is no need to push the whole image to the server side in order to make expense extraction from that. Instead of that, only the extracted expense information itself is posted to the server.

Such kind of approach acts in favour of descreasing of internet traffic intensity,

as the payload size for the post request with the expense data is around several hundred bytes instead of several megabytes in case of the whole image

Cons of this solution: same as the points c) and d) of the 1st solution.

1. Develop a mobile application (client) and server. The mobile client should be thin and all the business logic should be located on the server side. The expense displayed to the user are retrieved with a request to the server, the same goes for submissions of receipt images for OCR and further expense information extraction.

<http://cdn.intechopen.com/pdfs-wm/11405.pdf>

<http://static.googleusercontent.com/media/research.google.com/en//pubs/archive/33418.pdf>

http://www.ipol.im/pub/art/2011/bcm\_nlm/