

Optimization of CAPS user interface

Ulrik Enevoldsen

Kongens Lyngby 2012
IMM-B.Eng-2010-84

Technical University of Denmark
Department of Informatics and Mathematical Modeling
Building 321, DK-2800 Kongens Lyngby, Denmark
Phone +45 45253351, Fax +45 45882673
reception@imm.dtu.dk
www.imm.dtu.dk

Summary

In this thesis a new and better user interface for the existing Nets A/S product, Card Acquiring Processing System CAPS, based on a number of common used scenarios is described.

The design process is a user centered process and the new design is based on the user's needs, routines and the workflows where CAPS is used as a tool. The new design should increase the usability and streamline the user's workflows.

In the analysis the user's needs is mapped by user research, where the users are observed using the existing system. The key findings are that the users work in several workspaces, they use a lot of time navigation to different screens, and there is no help memorizing data.

The user needs and observations are mapped into the actual requirements for the design. Different solutions are discussed and the most suitable, a solution, where all changes can be done from one page and where workspaces are incorporated, is chosen.

A conceptual design is created to fulfill the requirements and a prototype is implemented.

The prototype is evaluated by conducting some usability tests and an expert evaluation. These evaluations show that the new user interface generally meets the requirements and points out some issues for the further development.

Also a small test comparing the prototype and CAPS were performed, and this test clearly shows improvement in the time used completing a task using the prototype.

I denne rapport bliver en ny og bedre brugergrænseflade, for et antal af hyppigt brugte skærbilleder, til det eksisterende Nets A/S, Card Acquiring Processing System, CAPS beskrevet.

Den anvendte design proces er bruger-centreret, og det nye design er således baseret på brugernes behov, rutiner samt de arbejdsgange hvor brugerne bruger CAPS som et arbejdsredskab. Det nye design skal øge systemets brugervenlighed samt effektivisere brugernes arbejdsprocesser.

I analysen bliver brugernes behov kortlagt via en brugerundersøgelse hvor brugerne bliver observeret mens de arbejder med CAPS systemet. De vigtigste resultater fra denne undersøgelse viser, at brugerne har brug for at arbejde med flere ting på samme tid, brugerne bruge meget tid på at navigere rundt mellem forskellige skærbilleder, og der er ingen hjælp til lettere at huske data.

Brugernes behov og observationerne heraf bliver overført til de faktiske krav til systemets design. Forskellige løsninger til problemstillingen diskuteres og den mest hensigtsmæssige vælges. Den valgte løsning går ud på at lave et design hvor alle ændringer kan ske fra samme side, samt at implementere workspaces.

Et konceptuelt design bliver udviklet således at det opfylder de givne krav til system, og en prototype implementeret.

Prototypen bliver evalueret ved at udføre nogle usability tests på denne. Samtidig laves der også en ekspert evaluering. Disse evalueringer viser at den nye brugergrænseflade opfylder kravene, og brugerne generelt er meget tilfredse. Evalueringer pointerer også visse svagheder i designet, som kan arbejdes med i den videre udvikling af systemet.

Der er også lavet en lille test hvor prototypen og CAPS bliver sammenlignet. Denne test viser klar en forbedring af tiden brugt på en opgave, når brugerne bruger prototypen.

Preface

This thesis was prepared at Department of Informatics Mathematical Modeling, Technical University of Denmark in collaboration with Nets Denmark A/S as partial fulfillment of the requirements for acquiring the Bachelor of Engineering (IT Engineering) degree.

The work on the thesis was done from September 1st, 2011 to January 20th, 2012, which corresponds to 18 weeks. The workload corresponds to 20 ECTS points.

The thesis supervisors are Michael Kai Petersen, Department of Informatics Mathematical Modeling, Technical University of Denmark and Morten Dines Andersen, Nets Denmark A/S.

Kongens Lyngby, January 2012
Ulrik Enevoldsen

Contents

Summary.....	i
Resumé	ii
Preface	iii
Chapter 1 Introduction.....	1
Chapter 2 Analysis	4
2.1 The CAPS System.....	4
2.1.1 Existing functionality	5
2.2 User research	6
2.2.1 Observations.....	6
2.2.2 User expectations	7
2.3 Requirements.....	8
2.3.1 General requirements	8
2.3.2 Observed requirements.....	8
2.4 Analysis of solutions.....	9
2.5 Use case descriptions.....	10
2.5.1 Add workspace	10
2.5.2 Change workspace.....	10
2.5.3 Search for merchant	11
2.5.4 Show merchant overview	11
2.5.5 Show merchant information	11
2.5.6 Edit merchant information	12
2.5.7 List output recipients.....	12
2.5.8 Edit output recipient	13
2.5.9 Create output recipient	13
Chapter 3 Conceptual design	14
3.1 Visual framework.....	14
3.1.1 Colors.....	14
3.1.2 Font.....	15

3.2	General layout.....	16
3.3	Organizing the content	17
3.3.1	Option bar.....	17
3.3.2	Workspaces	19
3.3.3	Navigation.....	22
3.4	Merchants.....	24
3.4.1	Search input.....	24
3.4.2	Search results	27
3.5	Overview	30
3.5.1	Merchant information	30
3.5.2	Merchant objects.....	33
Chapter 4	Implementation	36
Chapter 5	Evaluation	37
5.1	Usability test	37
5.1.1	Test results.....	37
5.2	Analytic evaluation	39
5.3	Statistic evaluation.....	43
Chapter 6	Discussion	45
Chapter 7	Conclusion	46
Literature	47
Appendix A	Use case descriptions.....	48
A.1	Remove workspace.....	48
A.2	List settlement agreements	48
A.3	Edit settlement agreement.....	48
A.4	Create settlement agreement	49
A.5	List bank accounts.....	49
A.6	Edit bank account	50
A.7	Create bank account	50
A.8	List merchant's customer groups	50
A.9	Add merchant to customer group	51
Appendix B	Evaluation data	52
Appendix C	CD content	53
Appendix D	Installation guide	54

This thesis will document the development of a new user interface for the web-based CAPS Card Acquiring Processing System. The background for this project is that Nets wants a new and smarter user interface for CAPS. The existing user interface is over ten years old and has been has never redesigned. The new user interface should increase the effectiveness of the users work and be more pleasant to work with.

CAPS is a system that helps acquirers handle their card business.¹ The system handles settling money between an issuer and a merchant.² This means that the system contains information of card types the merchant can handle, how transactions will be settled and administration of various fees to the acquirer. The system also handles advices, account printouts and chargeback advisements between the acquirer and the merchant.

The acquirer and merchant enter a settlement agreement. The acquirer sets up the merchant in CAPS storing the information from the settlement agreement. Now a cardholder can use his credit card at the merchant. The credit card is issued to the cardholder by an issuer. The merchant sends the payment transaction to Nets' backend where it is processed. If the transaction is approved backend sends the transaction to CAPS. Now CAPS uses the information stored in the system to do the settlement of money between the issuer and the merchant through backend. The flow can be seen in the figure below.

¹ An acquirer is a financial institution accepting card payments for a merchant.

² An issuer is a bank issuing a credit card to a cardholder.

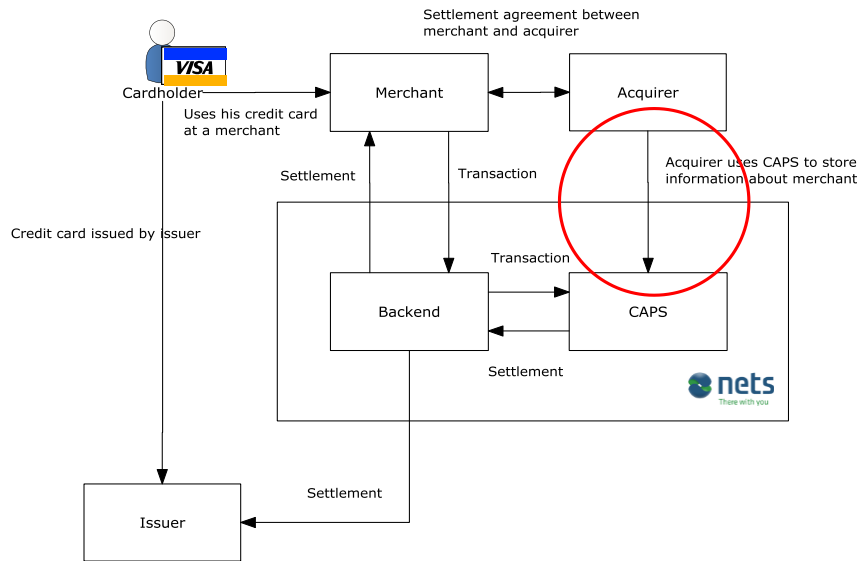


Figure 1 shows the settlement flow between merchant and issuer.

The focus of the project is designing a new and better user interaction between the acquirer and CAPS. Because CAPS is a very large system I have decided to limit the project and focus on some chosen use cases. The goal is to design and implement a prototype with a new user interface where it is possible to execute some of the most commonly used use cases, like changing address and emails for a merchant.

The process of designing user interfaces is an iterative process where the user is in focus. We need to understand the needs of the user and the domain of the system in order to make the perfect solution.

An iteration starts with identifying needs and establishing requirements. Next a conceptual design and a prototype are made reflecting the needs and requirements obtained. Lastly a user evaluation of the design and prototype is conducted, to see if the user's needs are fulfilled. Based on the results from the evaluation phase the next iteration is started. Here new requirements are identified and so forth. The process will continue until the buyer of the software is satisfied with the evaluation. The figure below illustrates this design process.³

³ Figure is inspired by the ISO 13407 human-centered design lifecycle model. [1] UsabilityNet. (2006). ISO 1347. [online]. Available from: <http://www.usabilitynet.org/tools/13407stds.htm> [Accessed 10 January 2012].

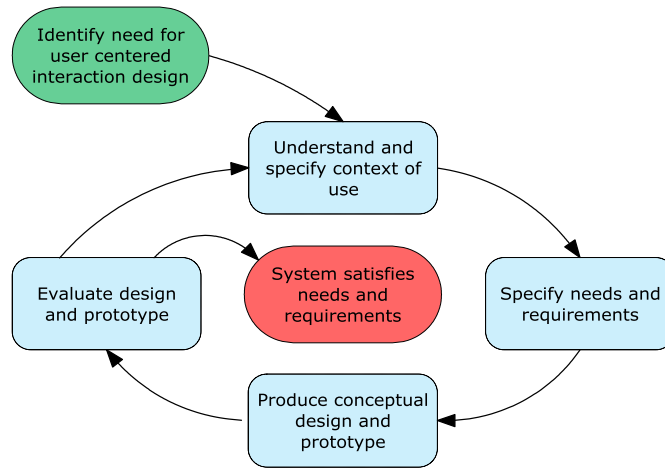


Figure 2 shows the iterative user centered interaction design process.

This thesis will describe the first iteration of the process of designing a new user interface for CAPS. First an analysis of the existing functionality in CAPS is made and requirements for the new design are revealed by observing the users. A new design is created and a prototype is implemented. Finally the prototype is usability tested and analytically evaluated.

2.1 The CAPS System

The CAPS system is a very large system with several hundred of different screens, where information can be viewed and altered. The system is used on a daily basis by the 600 employees in Nets Operations that handles the acquirer business.

The department Customer Service Administration DK and their tasks involving CAPS will be used as a reference group throughout the project. This reference group of users is called case handlers. They use CAPS on an everyday basis as of their main work tools.

The case handler has some kind of interaction with a merchant where the merchant request some changes to their settlement agreement and provides some kind of identification in form of a merchant number or vat number.

The case handler carry out the changes using CAPS. To do this the case handler has to find the merchant in CAPS using the information given and then navigate to the right page to make the changes. After the change is made the case handler typically goes to the overview page to check everything is as it should be. This change request flow is illustrated below.

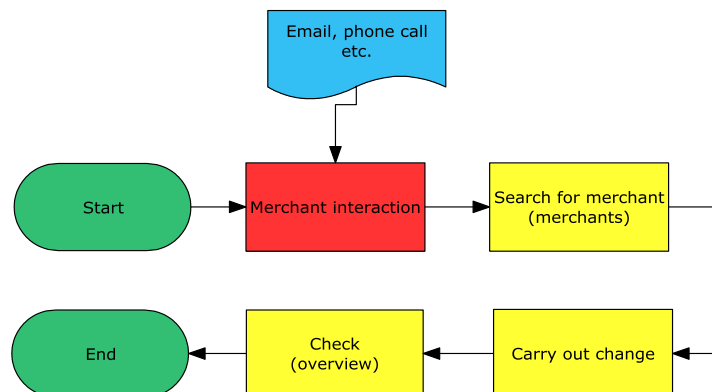


Figure 3 shows the workflow when a merchant requests a change to their settlement agreement.

Most merchant has agreements with several acquirers, to be able to accept different cards e.g. Dankort and Visa. The case handlers administrate all these acquirers and the data needs to be updated with all the different acquirers the merchant has an agreement with. This means that the users have to work with several acquirers at the same time.

Some typical tasks are, update the address for a merchant and add a new email for advices also called output recipients.

The scope of this project will be to redesign the user interface and implement a prototype where it is possible to perform the simple change request workflow for bank accounts, settlement agreements, output recipients, customer groups and basic merchant information.

2.1.1 Existing functionality

Here the existing screens of CAPS used to perform the actions in the scope of this project are described. Each row in the table below corresponds to a screen in the existing system.

Screen	Description
Search for merchant	The user searches for a merchant with a given acquirer.
Edit merchant	The user edits data for a merchant with a given acquirer.
Overview	The system shows the overview screen for a merchant.
List output recipients	The system lists the current output recipients for the merchant.
Edit output recipient	The user edits data for an existing output recipient.
Create output recipient	The user creates a new output recipient.
List customer groups for merchant	The system lists the customer groups for the merchant.
Search merchants with customer group	The user searches for merchants with a specific customer group.
List merchants with customer group	The system lists the merchants with the given customer group.
Add customer group	The user adds a customer group to the merchant.
List settlement agreements	The system lists the current settlement agreements for the merchant.
Edit settlement agreement	The user edits data for an existing settlement agreement.
Create settlement agreement	The user creates a new settlement agreement.
List bank accounts	The system lists the current bank accounts attached the merchant.
Edit bank account	The user edits data for an existing bank account.
Create bank account	The user creates a new bank account.

Table 1 Description of the screens and functionality associated with the change request workflow.

2.2 User research

To understand what the users' needs and to determine the requirements for a new CAPS user interface some research needs to be conducted. There are several ways to obtain information about the users' needs and expectations to a system.

Generally the best way to gather data from the users is direct observation. Surveys can be hard to analyze and personas does not show us what the users actually do with the system.

On site user visits, watching the user use the product, gives the best overview over the needs and requirements for the new software. The observer can be in the background without intervening and putting the user under any kind of pressure. The user feels secure, and does what he normally does, because he is in his normal environment. This makes it easier for the observer to locate the needs for the system. Furthermore questions can be asked if anything is unclear about the user's behavior. The observer can also ask other questions and start a dialog to learn what the user's expectations for the system are.

In this project some users from the reference group Customer Service Administration DK have been observed in their daily work to learn how CAPS is working and used by the users.⁴

2.2.1 Observations

The observations listed here are things that could be important for the new design of a new user interface.

- ☒ The users use several browser windows when editing a merchant, one for the AMEX acquirer and one for the Visa acquirer. The user does this to avoid typing the acquirer id every time he has to change between the different parts of the process.
- ☒ The users remember the acquirer ids. There is a help function, but the help text comes in a popup menu that do not make easy to find what you want. New users of the system often have a paper with the different acquirer ids.
- ☒ Some users use a low screen resolution, but have big screens 24'. The lowest observed was 1024 x 768 pixels.
- ☒ The users use the overview screen to check that the information for a merchant is correct. But if they find an error they have navigate to the correct screen using the menu to fix the error.

⁴ A video example how the CAPS user interface is currently used is available on the CD. Appendix C CD content. [C] Video: *caps_example.avi*

- ☒ If the user has to change more than one parameter for a merchant he has to go through the search picture every time. The acquirer and merchant number is filled but the user still has to click or press enter.
- ☒ Experienced users of CAPS tend to use the keyboard as much as possible, in order to work quicker.
- ☒ The users often have to edit the same information for many merchants e.g. a chain of shops ask to get another type of output recipient. This is a boring task and can easily result in errors from the user.
- ☒ When the users need help for filling in field with a predefined value e.g. a zip code. The help is loaded in a pop up window, with no search assistance.

2.2.2 User expectations

During the observation and the talks with the users of the reference group their expectations to a new interface were revealed. The users complained a lot over the responsiveness of the application, long wait times etc.

Unfortunately the wait times, is caused by the different calls to the backend and a new user interface cannot do anything about this problem. This is an issue the common user has trouble understanding. Underneath the main points of the user expectations is listed.

- ☒ A new user interface should have a more modern and simple design, like fancy web applications as Facebook, Twitter etc.
- ☒ A new user interface should give a better indication of loading time when something is processing.
- ☒ The new user interface must minimize the number of mouse clicks.

2.3 Requirements

This section describes the requirements for the new user interface. The requirements are based on the observations from the user research and the general requirements to the system from the product owners. Below is a vision statement from the product owners.

We want CAPS to have a new and more intuitive user interface. The user should use a minimum of mouse clicks to navigate the application and use less time completing his tasks. We imagine a solution where the user can change a lot of data in one screen.

2.3.1 General requirements

The general requirements to the new user interface are based upon the vision statement from the product owners.

- The system should be more satisfying for the user to use.
- The layout and design should be redesigned.
- It is important that the new design is neutral as several acquirers will use the system.
- When using the new user interface the number of mouse clicks for completing a task should be less than with the old user interface.
- It should be possible to change all the information you need in one screen.
- The system must still support easy access to several languages.

2.3.2 Observed requirements

Here the observations done in the user research is mapped into specific requirements for the system.

- The user must be able do his job properly in one browser window.
- There must better solutions for auto completion of various fields.
- The user must only encounter one search screen per merchant.
- The overview screen must offer more effective shortcuts.
- The content of the pages should be able to be seen in the screen resolution 1024 x 768 pixels without the user having scroll horizontally.
- The solution should offer as much keyboard interaction as possible.

2.4 Analysis of solutions

There are a few different ways to solve the problems given by the requirements.

One solution to make a more efficient workflow for the users is to implement a process engine, where workflows can be defined telling the system which screens to use to complete a particular workflow. The workflow would then be completed by pressing a next button every time a step in the flow was executed.

This solution would prevent the users from navigation through the menu and definitely save some mouse clicks. Furthermore the users could avoid the use of more than one browser window as the process engine would be able to switch between the different acquirers used in the task. This could be a solution but it would require building a brand new module to CAPS from scratch to handle the process engine and that would be a larger project.

To solve the problem concerning that the users perform the same task multiple times in the same workday, a solution could be a module performing the changes automatically. This module could for instance read excel documents with a well defined structure. The needed parameters would be acquirer numbers, merchant numbers, the fields that need to be changed and some input for the fields present. When the document is interpreted the changes could be done automatically by the system.

This solution could also save a lot of time and mouse clicks for the users as they could collect incoming request and run CAPS one time in the end of the day. With this solution it would also be possible to make rollbacks a lot easier. This idea could create a more efficient product, but again it would require big changes to the backend and the focus of this project is to make change to the user interface.

The solution I will work on focuses on making a simple design, where the user has to go through a minimum of different screens to complete the task he is assigned. There will be a simple search page to search for merchants at different acquirers. It will be possible to make all simple changes to a merchant from the overview screen, where the changeable items for a merchant will open inside. This will increase the efficiency of all workflows using CAPS because all unnecessary screens wait time and mouse clicks will be cut out.

To solve the problem concerning working with different acquirers at the same time, workspaces will be introduced. It will be possible to open several workspaces in the same main page. This will be implemented using tabs. This should also increase the general efficiency when working with the system because the acquirers always will be visible and easy to change between.

2.5 Use case descriptions

In this section the suggested solution and requirements for the application is mapped to use cases to describe the new system behavior. The use cases cover the scenario where a user has the search for a merchant at two acquirers and see the overview page for each to correct something.

Only full use cases used for changing simple merchant data and adding, editing output recipients are described in this section. The rest can be seen in Appendix A.

2.5.1 Add workspace

Use case: Add workspace

Actors: User, CAPS, Backend

Summary: The user needs to search for a merchant at an acquirer that he has not got an open workspace for.

Pre-condition: CAPS is loaded.

Basic course of events:

- 1 The user clicks the add workspace button.
- 2 The system adds another workspace to the screen.

Post-condition: Another workspace is added to the screen.

2.5.2 Change workspace

Use case: Change workspace

Actors: User, CAPS

Summary: The user wants change his current workspace to a workspace for a different acquirer.

Pre-condition: CAPS is loaded and more than one workspace is open.

Basic course of events:

- 1 The user clicks workspace tab he wants to switch to.
- 2 The system changes to the clicked workspace.

Extensions:

- 1a The user uses the keyboard CTRL+right/left arrow to navigate the workspaces.

Post-condition: The current workspace has changed.

2.5.3 Search for merchant

Use case: Search for merchant

Actors: User, CAPS, Backend

Summary: The user wants to find a merchant from a given acquirer.

Pre-condition: The user clicks the link merchants and loads the search page.

Basic course of events:

- 1 The user chooses the acquirer and search date.
- 2 The user fills in his search criteria's and clicks the search button.
- 3 The system searches for a merchant with the given parameters.
- 4 The system shows a list of found merchants.

Extensions:

4b The system cannot find any merchants with the given search criteria's and gives an error message.

4c The system gives an error and shows a relevant error message.

Post-condition: The user is has found the merchant he was looking for.

2.5.4 Show merchant overview

Use case: Show merchant overview

Actors: User, CAPS, Backend

Summary: The user has found a merchant by using the search and wants to see detail about this merchant.

Pre-condition: The user has found a merchant. (*Search for merchant*)

Basic course of events:

- 1 The clicks the merchant he wants to see or overview link in the menu.
- 2 The system loads the merchant overview page by loading merchant information, settlement agreements, bank accounts, output recipients and customer groups.

Extensions:

2b If a merchant is not chosen when using the menu to navigate to the overview page, the system redirects to the search page.

Post-condition: The overview page is loaded.

2.5.5 Show merchant information

Use case: Show merchant information

Actor: User, CAPS, Backend system

Summary: The user loads the overview page to see merchant information like name, address etc.

Pre-condition: The user has found the merchant he wants to see. (*Show merchant overview*)

Basic course of events:

1 The system loads and shows merchant information.

Extensions:

1a The system gives an error and shows a relevant error message.

Post-condition: The merchant information is shown in the overview page.

2.5.6 Edit merchant information

Use case: Edit merchant

Actor: User, CAPS, Backend

Summary: The user wants to edit information about a merchant.

Pre-condition: The merchant information is loaded. (*Show merchant information*)

Basic course of events:

1 The user presses the edit merchant button.

2 The system shows the merchant form.

3 The user types in his changes and saves the data.

Extensions:

2a The system gives an error and shows a relevant error message.

3b The system gives an error and shows a relevant error message.

Post-condition: The user has edited some information for a merchant.

2.5.7 List output recipients

Use case: List output recipients

Actor: User, CAPS, Backend system

Summary: The user wants to see the output recipients for a merchant.

Pre-condition: The user has found the merchant he wants to see. (*Show merchant overview*)

Basic course of events:

1 The system lists the output recipients for the merchant.

Extensions:

1a The system gives an error and shows a relevant error message.

Post-condition: The output recipients are listed in the overview page.

2.5.8 Edit output recipient

Use case: Edit output recipient

Actor: User, CAPS, Backend system

Summary: The user wants to edit an output recipient for a merchant.

Pre-condition: The user has listed the output recipients for a merchant. (*List output recipients*)

Basic course of events:

- 1 The user presses the output recipient he wants to edit.
- 2 The systems show the output recipient form.
- 3 The user edits the information and saves.

Extensions:

- 2a The system gives an error and shows a relevant error message.
- 3b The system gives an error and shows a relevant error message.

Post-condition: The user has edited an output recipient for a merchant.

2.5.9 Create output recipient

Use case: Create output recipient

Actor: User, CAPS, Backend system

Summary: The user wants to create a new output recipient for a merchant.

Pre-condition: The user has listed the output recipient for a merchant. (*List output recipients*)

Basic course of events:

- 1 The user presses the create output recipient button.
- 2 The systems show the output recipient form.
- 3 The user types in data and saves.

Extensions:

- 2a The system gives an error and shows a relevant error message.
- 3b The system gives an error and shows a relevant error message.

Post-condition: The user has created an output recipient for a merchant.

In this chapter the design and functionality of the new user interface will be described.

3.1 Visual framework

For the general layout I will create a visual framework, so every page has the same basic layout. With consistent color, font, placements of buttons and layout the user will know where to find things and don't have to learn more than one layout. A strong visual framework for the application will help creating a better brand for CAPS. The whole application should graphically hang together.

The visual framework will build on the logo and header that looks like the previous. It will stay alike since the user should recognize that it is still the same product as before.



Figure 4 shows the new header.

3.1.1 Colors

The colors used in the application follow the ones used in the logo. The blue shaded colors will be used for things the users should pay attention to. The gray shades will be used for button and backgrounds.



Figure 5 shows the color scheme.

3.1.2 Font

The font type used in the design is Verdana. This is the standard font used in Nets A/S. The headings have the same color as the logo, labels and help text are gray shaded and plain text is black.

Heading 1	Text	Help text
Heading 2	Menu	Labels

Figure 6 shows the different fonts used in the design.

3.2 General layout

The layout will contain a header placed in the top of the page, an options section placed under the header, a navigation bar in the left side of the page and a content area. The header and options section is distinguished by the colors and size of their areas. The navigation area is recognized by the alignment and similarity of the menu items and the space to the other sections. This makes it look like its own group. The content should draw the most of the users' attention. To achieve this, the content area fills most of the page and is easily distinguished from the other page elements.⁵

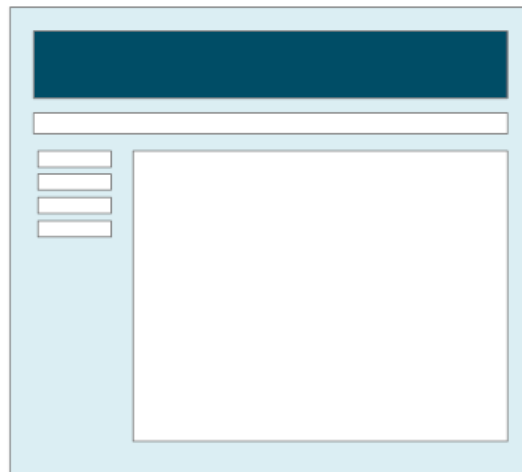


Figure 7 shows the general page layout.

We want to use as much as the screen as possible for our content. The users have big screens that support 1920 x 1280 pixels but we also want to support laptops and users that use lower screen resolutions. To do this the page layout will be centered in a wrapper with a width of 1024 pixels. The page layout is centered in a grey background to draw the users' attention.

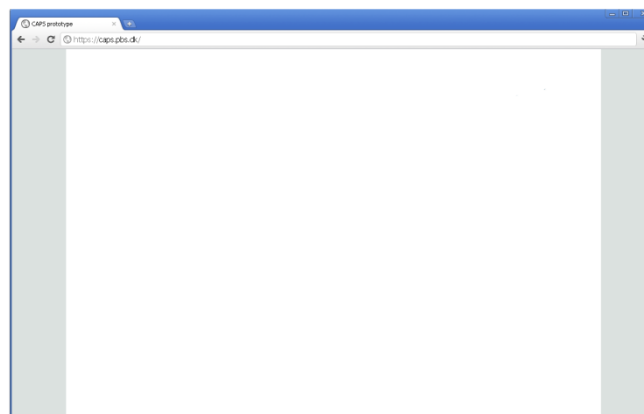


Figure 8 shows the layout page in a browser window with a width over 1024 pixels.

⁵ Using the Gestalt Principles. [2] Tidwell, Jenifer. (2011) *Designing Interfaces*. 2nd Edition. Sebastopol: O'Reilly. Page 139.

3.3 Organizing the content

The users mostly works with three different acquirers open at the same time, and their task makes them jump around between them, checking and copying information. Therefore the application should have the possibility to load the content of several pages into one main page as separate workspaces. The users should also have the possibility to always change the language of the application.

3.3.1 Option bar

The main page has an option bar where the user is able to change the applications language to his preferred. The option bar also shows information about the user that is logged into the system. The language selector works like as normal select box where a list of options drop down when clicking at the button. When a new value is selected by either clicking or navigation with the keyboard the application will change the language. When the language is changed the whole page will reload, and any changes not saved will be lost.

The selector provides small icons a long with a text. The icons should help the user understand the context of the language selector quicker as pictures is easier to associate things with than a text.

3.3.1.1 Conceptual design

The following illustrations will show how the option bar looks and functions.

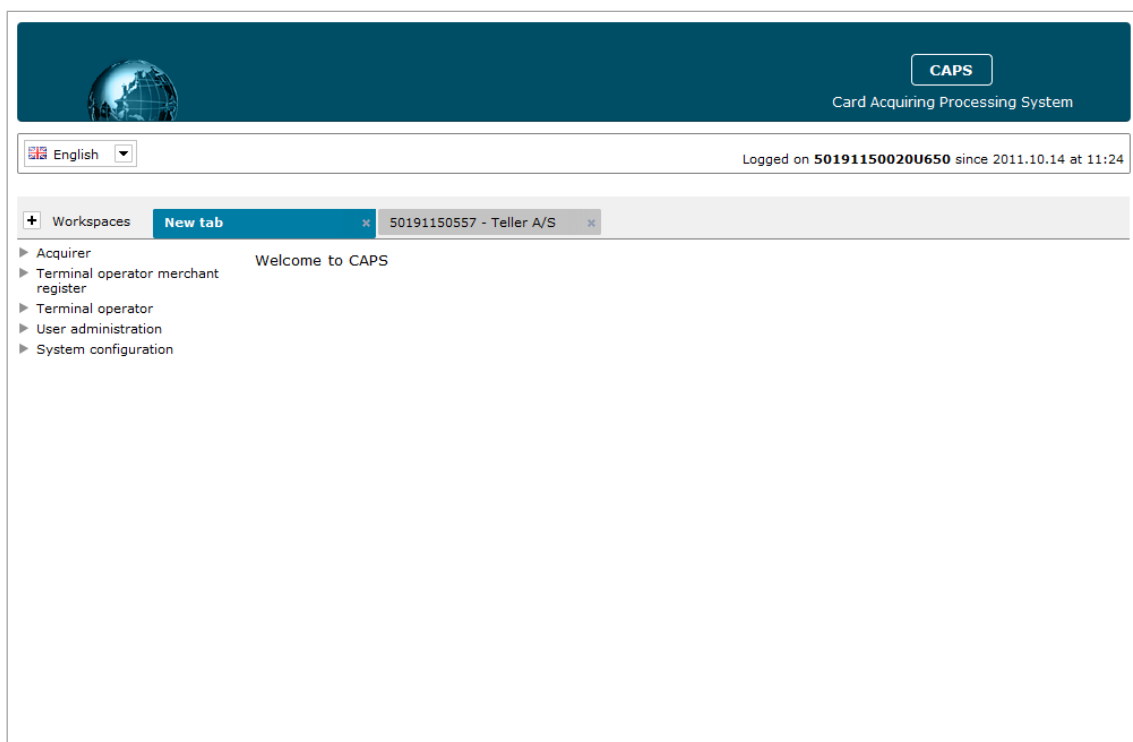


Figure 9 shows the welcome page.

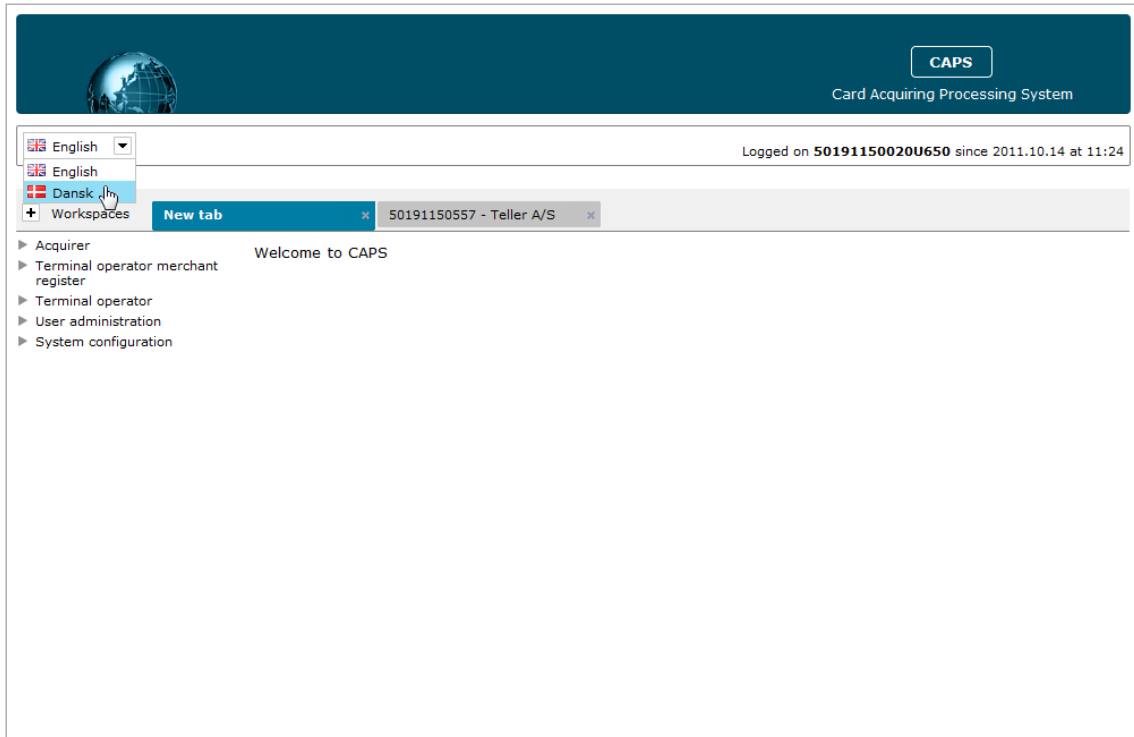


Figure 10 shows the language selector in action.

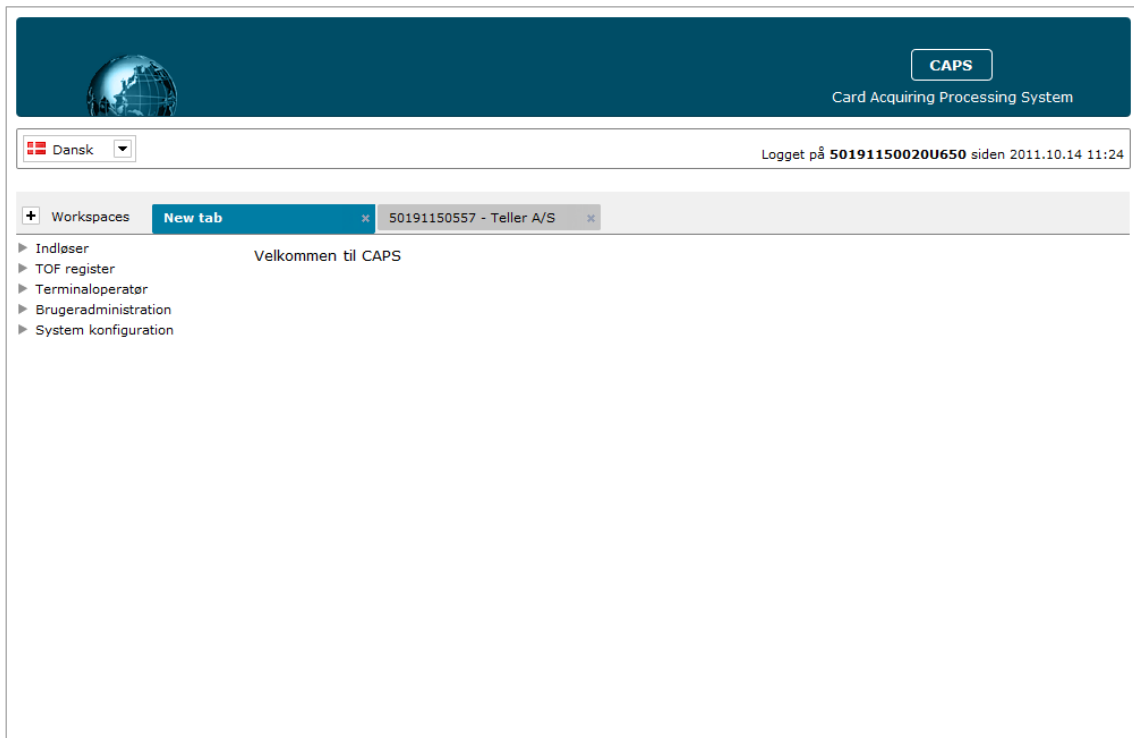


Figure 11 show the welcome page after a new language has been chosen.

3.3.2 Workspaces

To achieve that, the main page can load several pages, the application will include tabs. Each tab represents a workspace and will show the content of a loaded page. The user will be able to respectively open, close and browse the workspaces as he needs. The tab solution will help the user separate the different acquirers he is working on and easily switch between them. To enforce this, the labels on the tabs will change and show the name of the acquirer used in the workspace. The tabs opened will be stored in the user's session and will be remembered if the user reloads the main page.

The workspaces acts like other known tabs like 'browser tab'. By pressing the 'add new workspace button' a new workspace is opened. By clicking the 'close workspace button' on a tab the workspace will close. The user can easily jump between workspaces by clicking the tabs or by navigating CTRL + left/right arrow keys. The tabs can also be moved around so the order is changed.

The current workspace is highlighted and the tab is attached to the content of the page. This is done to make clear which workspace currently is in use. The workspaces not shown are grey and are not attached to the content of the page.

With the implementation this workspace solution the users only have to use one browser window and tab to work effectively with the CAPS application. The navigation between the different workspaces is more effective and will save the users for mouse clicks and mouse navigation. When content is loaded into a workspace it will be done asynchronously so the main page does not have to be reloaded. This will give the user a feeling of a more smooth application where the load time is minimal.

3.3.2.1 Conceptual design

The following illustrations will show how the workspace solution is designed.

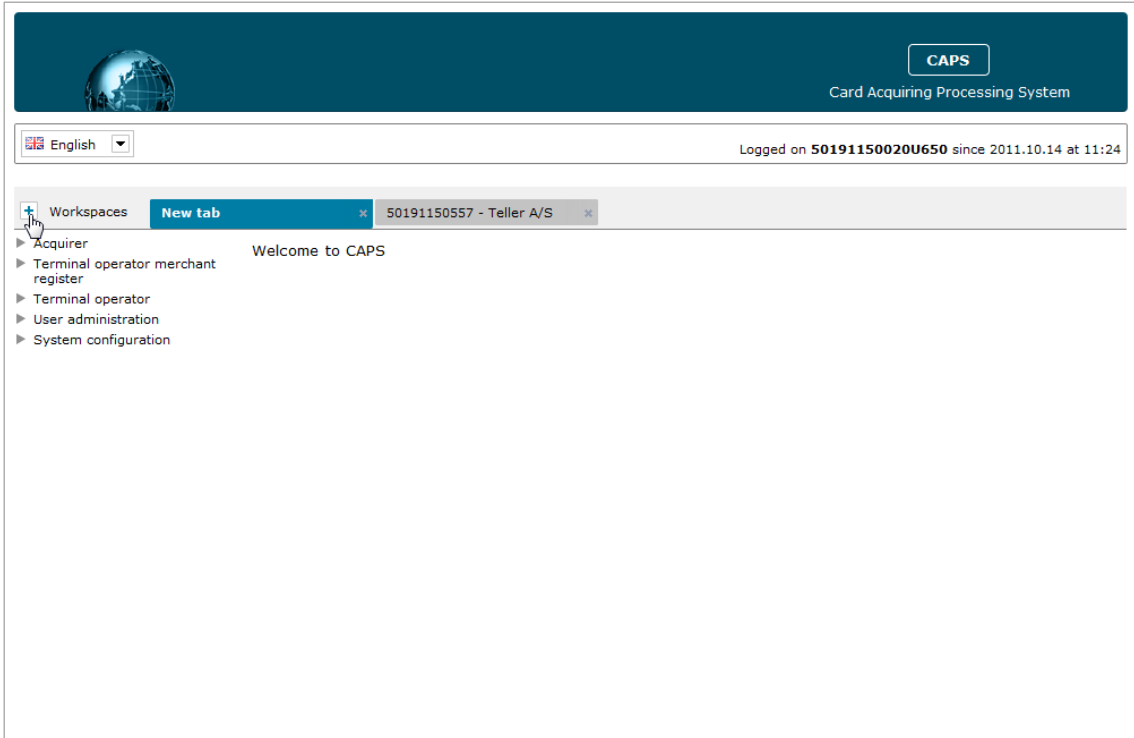


Figure 12 shows the add workspace button.

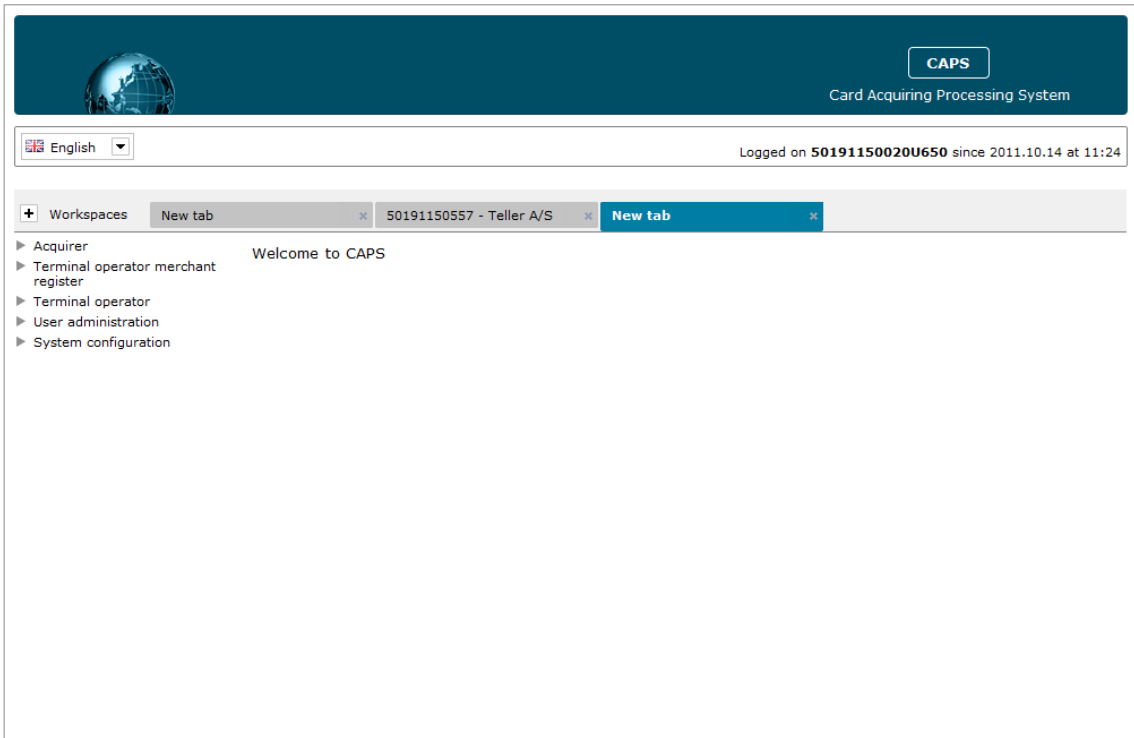


Figure 13 shows the newly added workspace.

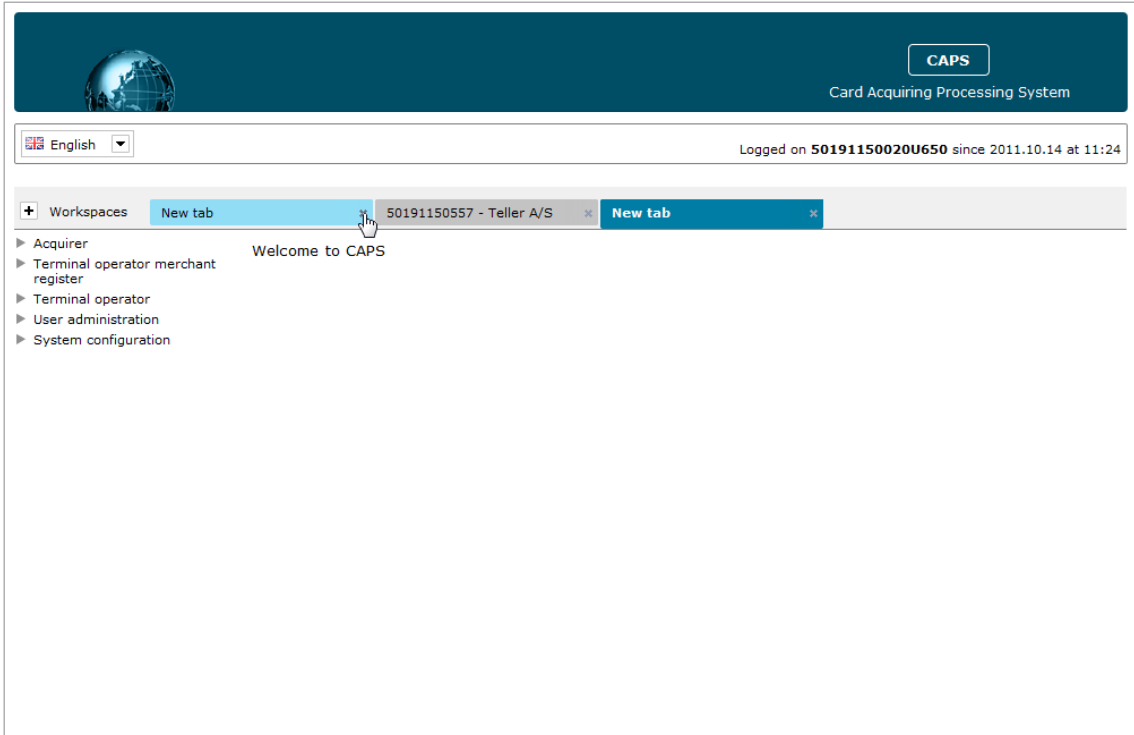


Figure 14 shows the close workspace button.

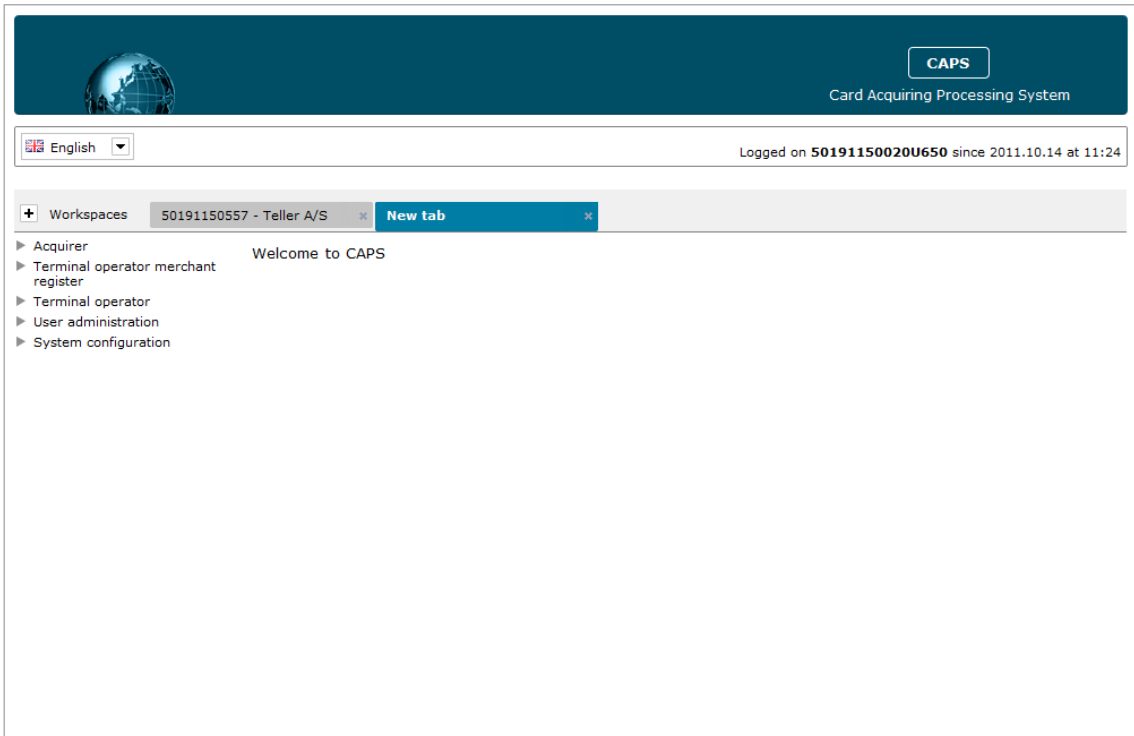


Figure 15 shows the workspaces after closing the workspace.

3.3.3 Navigation

For the general page navigation, there will be a global navigation bar in the left side of the screen. This navigation menu is found in every workspace and is used by the users to navigate the application. The global navigation is to be fully connected, so a user can reach any page from his current page. The existing CAPS menu has a large hierarchy and the new menu will have the same hierarchy.

A way to fulfill these requirements is to implement a tree menu. The user can open more than one menu item at the same time and access all menu items. Generally the three structured menu gives the user a good overview of the content because of hierarchal layout and is easy to navigate. The tree menu is fully connected and saves some time compared to a partial connected menu where the user has to load a page to get the next level of the menu.

The tree menu works by clicking the menu items. If the menu item is closed the menu item will expand and if it is expanded it will close. The triangle pointing left indicates that the menu item is expandable and the triangle pointing down shows that the menu item is expanded. If the menu item is a leaf it has no graphic. Clicking a leaf menu item will open this page in the currently selected workspace. Once the page is loaded the menu items will be bold so the user can identify where he currently is in the application and navigate where he wants.

3.3.3.1 Conceptual design

The illustrations below show the functionality of the menu.

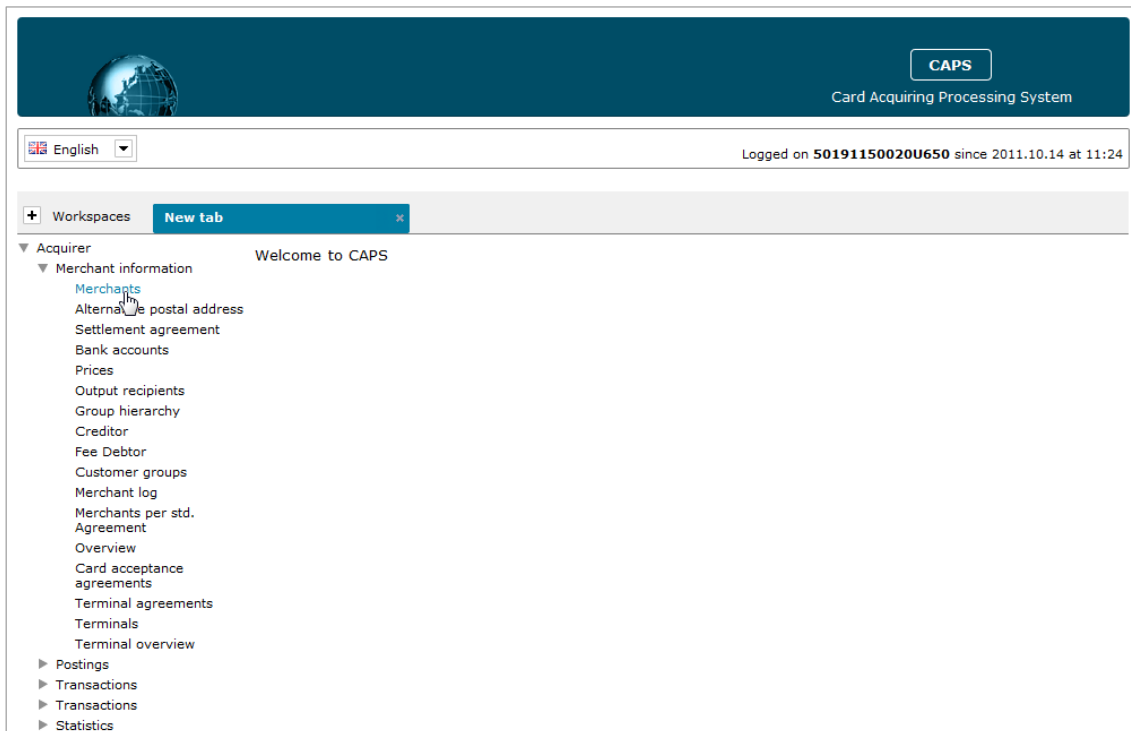



Figure 16 shows how the menu looks when folded out.


CAPS
Card Acquiring Processing System

English ▼
Logged on 50191150020U650 since 2011.10.14 at 11:24

+ Workspaces
Merchants
x

▼ Acquirer

▼ Merchant information

- Merchants**
- Alternative postal address
- Settlement agreement
- Bank accounts
- Prices
- Output recipients
- Group hierarchy
- Creditor
- Fee Debtor
- Customer groups
- Merchant log
- Merchants per std.
- Agreement
- Overview
- Card acceptance agreements
- Terminal agreements
- Terminals
- Terminal overview

- ▶ Postings
- ▶ Transactions
- ▶ Transactions
- ▶ Statistics

Merchants

Search for merchants
Create merchant Super create merchant

Acquirer

Choose acquirer.

Search date

Choose date.

Search

9

Search for merchant number, company reg. no., telephone no., terminal operator merchant no, business name, legal name, zip code, country code, merchant agreement no., business code.

Figure 17 shows the menu after loading content into a workspace.

23

3.4 Merchants

The merchant page has two functions. The user can create new merchants from this page or search for existing merchants. The page contains two main elements a search box and a result box. Above the search box is a group of buttons concerning the creation of merchants. The alignment and grouping of these buttons shows that the buttons are associated and they are alternatives to the search. The result box is placed under the search box. Above the result box two buttons is placed, these are used for exporting the search results. Beneath the result box there will be buttons used for pagination. The buttons are associated with the results, but they provide an alternative action. The overall design is shown in the figure below.

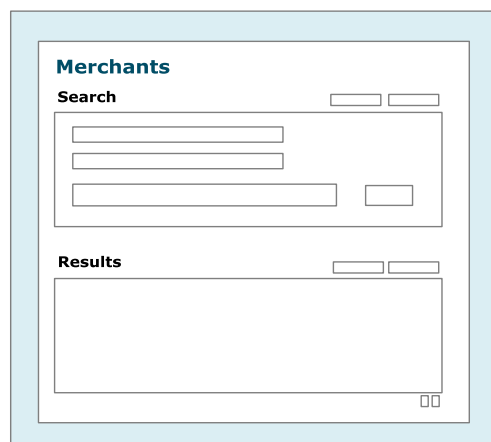


Figure 18 shows the intended layout for the search page.

3.4.1 Search input

The search box is an input form. The user has to choose an acquirer, search date and give some input in the search field to search for a merchant. The alignment of the labels and the input fields should make the user recognize them as one line and make it possible to separate them from each other. The label is made bold to highlight the name of the field. This will draw the user's attention and by the name of the label he should be able to understand how to fill the input.

The new search page is very different from the old design so to help the users there is a help text underneath each input field. This help text helps the user to understand the context of the input field if the label is not enough. The help text is separated from the label so users who knows what to do can ignore it and focus on the label and input control.

The acquirer input field is a combo box consisting of a text field and a dropdown list. The textbox functions as an auto completing search field where the string typed in is matched with the options in the select list. This approach will save the user a lot of time when typing the eleven character long acquirer id. Also it will reduce the user's memory burden as he only has to remember parts of the acquirer id or the name of

the acquirer. Furthermore type errors will be avoided. This field has to be filled out during a search. If a search is performed without choosing an acquirer the field will be marked with an error to show that the field is required.

The search date input field is a text field where the user types in the search date. As a default the current date is filled in. This will save the user finding the current date somewhere else. Next to the date field a calendar icon is placed. This icon serves two purposes. It is a metaphor helping the user to interpret that the field is a date field. When the user clicks the icon a calendar is unfolded, and the user is able to find the search date he wants.

The search input field is a normal text field where the user will type in the search parameters. The search input field is bigger than a standard text input. This is to draw the user's attention and indicate that this is the most important search parameter. Next to the input field is the search button. The search button is bigger than the other buttons visible to show that this will actual perform the search. The big search field and search button should make the user associate to a search engine like Google etc. The whole input box works like a normal form and the search can also be submitted by pressing enter when one of the input fields has focus.

3.4.1.1 Conceptual design

The following illustrations will show how the search is working.

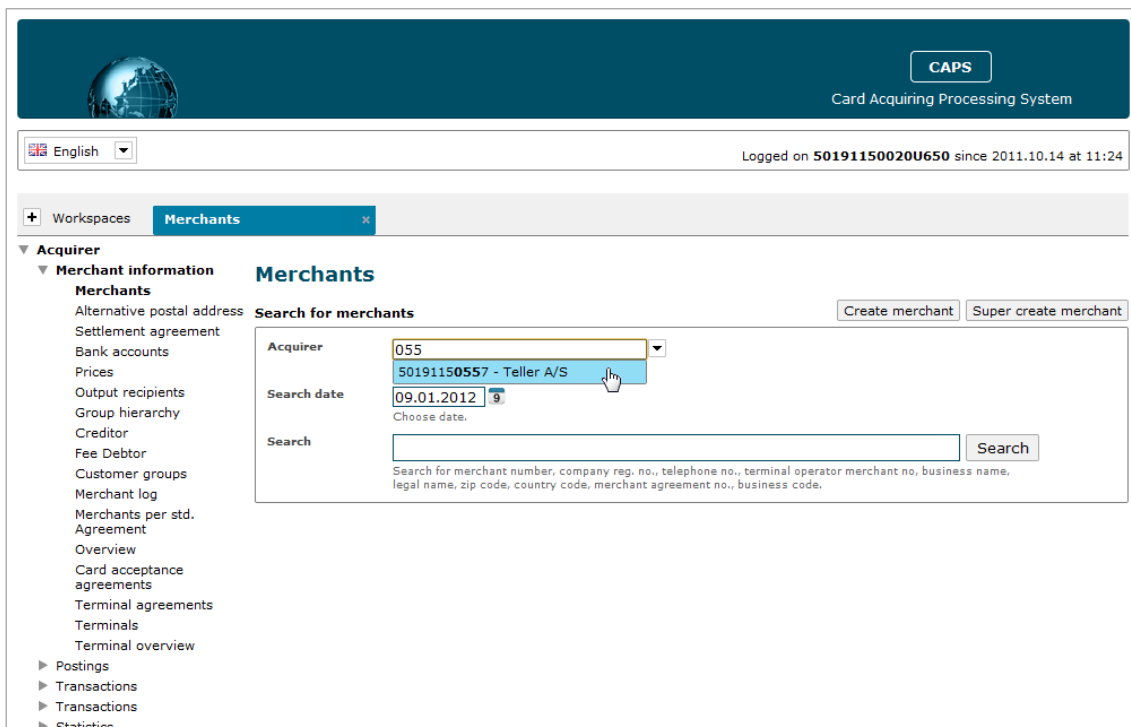


Figure 19 shows the auto completion in work.

The screenshot shows the CAPS (Card Acquiring Processing System) interface. At the top, there is a header with a globe icon and the text "CAPS Card Acquiring Processing System". Below the header, there is a language selector set to "English" and a login status "Logged on 50191150020U650 since 2011.10.14 at 11:24". The main content area is titled "Merchants" and includes a sidebar with navigation options like "Acquirer", "Merchant information", "Postings", "Transactions", and "Statistics". The "Search for merchants" section contains a form with the following fields:

- Acquirer:** A dropdown menu showing "50191150557 - Teller A/S".
- Search date:** A date input field showing "09.01.2012" with a calendar icon.
- Search:** An empty text input field.

A date picker calendar is open, showing the month of "January 2012". The calendar grid includes days of the week (Mo, Tu, We, Th, Fr, Sa, Su) and dates from 1 to 31. A tooltip is visible over the calendar, containing the text: ", telephone no., terminal operator merchant no, business name, agreement no., business code."

Figure 20 shows the date picker.

This screenshot shows the same CAPS interface as Figure 20, but with the search form filled out. The "Search" input field now contains the value "700001". A mouse cursor is pointing at the "Search" button. The tooltip text is now: "Search for merchant number, company reg. no., telephone no., terminal operator merchant no, business name, legal name, zip code, country code, merchant agreement no., business code."

Figure 21 shows the search page with filled input fields.

3.4.2 Search results

When the users submit his search the result box becomes visible and a load bar is shown. Furthermore the chosen acquirer will be the new name of the workspace, and this workspace will remember this acquirer until a new search in the workspace with a different acquirer is done. This naming of the workspaces will help the user to distinguish the workspaces with different acquirers from each other. If the search input from the user matches a merchant number the application will direct the user directly to the overview page of that merchant.

The loading indicator makes it clear for the user that something is happening and that he has to be patient. When the search results are rendered the load bar is removed and the results are filled in. The search is done asynchronously which means that the whole page will be responsive while the search is performed.

The merchants found in the search are listed in a table. Each row represents a merchant. A merchant has a lot of fields, so the columns' data should be the ones that at most commonly used to identify a merchant. To visually separate each rows, adjacent rows are colored differently. This makes each row a visual object due to the similarity of the rows and the closure of the columns.⁶

CAPS is a large system and most searches will give a large number of merchants found. To help the user identify exactly the merchant he is looking for it is possible to sort each column. When a search gives many resultsk, the result table is broken in to several parts and only a part of it is shown. This is done to make it easier for the user to get an overview and to prevent very long render times. For navigation between the different pages of results pagination is used. The pagination is positioned under the result box and shows which page the user is currently on and gives the option to navigate between them.

When a merchant in the result list is clicked the overview page of this merchant is loaded and the merchant number will be attached to the workspace. This means that the user will be able to go between different screens without having to make a new search. The merchant number attached to the workspace will be overwritten when a new merchant is found via a search.

3.4.2.1 Conceptual design

The following illustrations show the design of the result box.

⁶ Referring to the Gestalt Principles. [2] Tidwell, Jenifer. (2011) *Designing Interfaces*. 2nd Edition. Sebastopol: O'Reilly. Page 139.

CAPS
Card Acquiring Processing System

English

Logged on 50191150020650 since 2011.10.14 at 11:24

Workspaces 50191150557 - Teller A/S

Acquirer

Merchant information

Merchants

Alternative postal address
Settlement agreement
Bank accounts
Prices
Output recipients
Group hierarchy
Creditor
Fee Debtor
Customer groups
Merchant log
Merchants per std.
Agreement
Overview
Card acceptance agreements
Terminal agreements
Terminals
Terminal overview

Postings
Transactions
Transactions
Statistics

Search for merchants Create merchant Super create merchant

Acquirer: 50191150557 - Teller A/S
Choose acquirer.

Search date: 09.01.2012
Choose date.

Search: Fakta Search

Search for merchant number, company reg. no., telephone no., terminal operator merchant no, business name, legal name, zip code, country code, merchant agreement no., business code.

Search result

Figure 22 shows the load bar after the search button has been pressed.

Alternative postal address
Settlement agreement
Bank accounts
Prices
Output recipients
Group hierarchy
Creditor
Fee Debtor
Customer groups
Merchant log
Merchants per std.
Agreement
Overview
Card acceptance agreements
Terminal agreements
Terminals
Terminal overview

Postings
Transactions
Transactions
Statistics
Acquirer main data
Terminal operator merchant register
Terminal operator
User administration
System configuration

Search for merchants Create merchant Super create merchant

Acquirer: 50191150557 - Teller A/S
Choose acquirer.

Search date: 09.01.2012
Choose date.

Search: Fakta Search

Search for merchant number, company reg. no., telephone no., terminal operator merchant no, business name, legal name, zip code, country code, merchant agreement no., business code.

Search result Export to Excel Export to CSV

Merchant number	TO merchant no.	Business name	Zip code	Start date	Expiry date
7000001	1000001	Fakta 1	6200 Aabenraa	06.07.2007	
7000002	1000002	Fakta 2	6200 Aabenraa	06.07.2007	
7000003	1000003	Fakta 3	6200 Aabenraa	06.07.2007	
7000004	1000004	Fakta 4	6200 Aabenraa	06.07.2007	
7000005	1000005	Fakta 5	6200 Aabenraa	06.07.2007	
7000006	1000006	Fakta 6	6200 Aabenraa	06.07.2007	
7000007	1000007	Fakta 7	6200 Aabenraa	06.07.2007	
7000008	1000008	Fakta 8	6200 Aabenraa	06.07.2007	
7000009	1000009	Fakta 9	6200 Aabenraa	06.07.2007	
7000010	1000010	Fakta 10	6200 Aabenraa	06.07.2007	
7000011	1000011	Fakta 11	6200 Aabenraa	06.07.2007	
7000012	1000012	Fakta 12	6200 Aabenraa	06.07.2007	
7000013	1000013	Fakta 13	6200 Aabenraa	06.07.2007	
7000014	1000014	Fakta 14	6200 Aabenraa	06.07.2007	
7000015	1000022	Fakta 15	6200 Aabenraa	06.07.2007	

21 merchants found. 1 2 Next

Figure 23 shows the search result.

Alternative postal address **Search for merchants** [Create merchant](#) [Super create merchant](#)

Settlement agreement
Bank accounts
Prices
Output recipients
Group hierarchy
Creditor
Fee Debtor
Customer groups
Merchant log
Merchants per std. Agreement
Overview
Card acceptance agreements
Terminal agreements
Terminals
Terminal overview

► Postings
► Transactions
► Transactions log
► Statistics
► Acquirer main data
► Terminal operator merchant register
► Terminal operator
► User administration
► System configuration

Acquirer Choose acquirer.

Search date Choose date.

Search

Search for merchant number, company reg. no., telephone no., terminal operator merchant no., business name, legal name, zip code, country code, merchant agreement no., business code.

Search result [Export to Excel](#) [Export to CSV](#)

Merchant number	TO merchant no.	Business name	Zip code	Start date	Expiry date
7000001	1000001	Fakta 1	6200 Aabenraa	06.07.2007	
7000002	1000002	Fakta 2	6200 Aabenraa	06.07.2007	
7000003	1000003	Fakta 3	6200 Aabenraa	06.07.2007	
7000004	1000004	Fakta 4	6200 Aabenraa	06.07.2007	
7000005	1000005	Fakta 5	6200 Aabenraa	06.07.2007	
7000006	1000006	Fakta 6	6200 Aabenraa	06.07.2007	
7000007	1000007	Fakta 7	6200 Aabenraa	06.07.2007	
7000008	1000008	Fakta 8	6200 Aabenraa	06.07.2007	
7000009	1000009	Fakta 9	6200 Aabenraa	06.07.2007	
7000010	1000010	Fakta 10	6200 Aabenraa	06.07.2007	
7000011	1000011	Fakta 11	6200 Aabenraa	06.07.2007	
7000012	1000012	Fakta 12	6200 Aabenraa	06.07.2007	
7000013	1000013	Fakta 13	6200 Aabenraa	06.07.2007	
7000014	1000014	Fakta 14	6200 Aabenraa	06.07.2007	
7000015	1000022	Fakta 15	6200 Aabenraa	06.07.2007	

21 merchants found. [1](#) [2](#) [Next](#)

Figure 24 shows a highlighted merchant.

3.5 Overview

The overview page is the page where all information on a merchant can be view and altered. In the top of the page the unchangeable information like the merchant number and acquirer id is displayed. There is also a group of buttons used for deleting or resigning the merchant. These are placed for themselves as they are rarely used. The rest of the page consists of things that can be altered like merchant information, bank accounts, settlement agreements, output recipients and customer groups. These items are standard in the overview page and should give a quick overview of the merchant. The overall design is shown in the figure below.

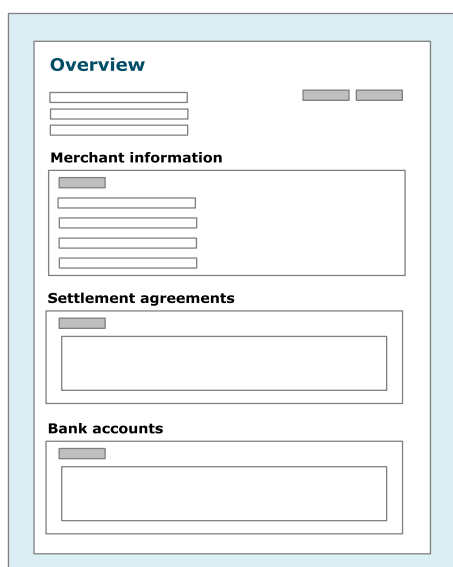


Figure 25 shows the layout for the overview page.

The idea is that when user clicks an item in the menu that is not currently shown in the overview page it will open in this page. If it is already shown in the overview page the application will scroll to the right place.

All boxes in the page are loaded asynchronously. This means that only this part of the entire page needs to be rendered. This should save some time and make the rest of the application usable while loading. Furthermore a load bar is shown when something is loading to indicate that something is in progress.

3.5.1 Merchant information

When the user opens the overview page not all merchant information is shown. Only the fields giving a quick overview of the merchant is shown. A merchant has a lot of fields, so if everything were to be shown the user could not easy create an overview. Besides that the full merchant information would also consume too much space. If the user wants to see the hidden fields he simply clicks the show all button in bottom of the merchant information box.

If the some merchant information needs to be altered the user simply clicks the edit button. The edit button is placed in the top of the merchant information box, so it is easy for the user to locate, he does not have to scroll to find it. When the edit button is clicked the edit merchant form is loaded. If the user wants to cancel the editing he can click the close button in the top right corner. The placement of this button is close to the mouse after pressing the edit button. If the user regrets his action then the navigation to the close button is short.

The form itself has the, by CAPS, required fields. All fields where the user has to choose a value from some predefined set are made as a combo box with auto completion. The start date and end date field are placed next to each other to save space and has a date picker to assist the user. When the user wants to save is entry he can press enter if one of the field is in focus, or he can go to the bottom of the form where the save button is located. The save button is located in the end of the form to force the user to check all the input fields. The save button is bigger than the rest of the buttons in the page to show that this will end a transaction.

When the mouse is over the form the box is highlighted and it is possible to close the form by pressing the escape button on the keyboard. The highlighting of the box will give the user the change to see what is closed when pressing the escape button.

3.5.1.1 Conceptual design

The following illustrations show merchant information boxes in the overview page.

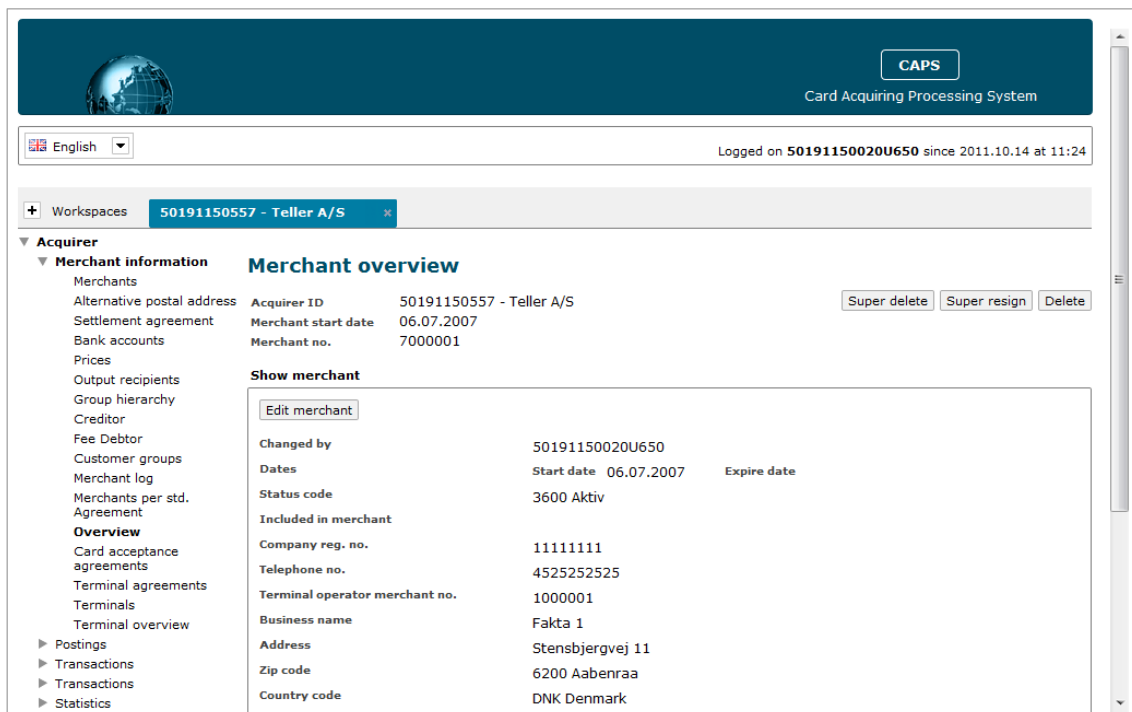


Figure 26 shows the top of the overview page.

▼ Acquirer

▼ Merchant information **Merchant overview**

Merchants

Alternative postal address Acquirer ID 50191150557 - Teller A/S Super delete Super resign Delete

Settlement agreement Merchant start date 06.07.2007

Bank accounts Merchant no. 7000001

Prices

Output recipients

Group hierarchy

Creditor

Fee Debtor

Customer groups

Merchant log

Merchants per std.

Agreement

Overview

Card acceptance agreements

Terminal agreements

Terminals

Terminal overview

► Postings

► Transactions

► Transactions

► Statistics

► Acquirer main data

► Terminal operator merchant register

► Terminal operator

► User administration

► System configuration

Show merchant

Changed by 50191150020U650

Dates Start date 06.07.2007 Expire date

Status code 3600 Aktiv

Included in merchant

Company reg. no. 11111111

Telephone no. 4525252525

Terminal operator merchant no. 1000001

Business name Fakta 1

Address Stensbjergvej 11

Zip code 6200 Aabenraa

Country code DNK Denmark

Business code

Contact person Ulrik Enevoldsen

Settlement agreement

Standard agreement no. Basic standard agreement Rate agreement no. Batch method Start date Expire date

1 Alm. Aftale med Dagl.advis. No 8 100 procent 1 KIB bundt. 01.01.2011 01.01.2020

Figure 27 shows the show merchant box

▼ Acquirer

▼ Merchant information **Merchant overview**

Merchants

Alternative postal address Acquirer ID 50191150557 - Teller A/S Super delete Super resign Delete

Settlement agreement Merchant start date 06.07.2007

Bank accounts Merchant no. 7000001

Prices

Output recipients

Group hierarchy

Creditor

Fee Debtor

Customer groups

Merchant log

Merchants per std.

Agreement

Overview

Card acceptance agreements

Terminal agreements

Terminals

Terminal overview

► Postings

► Transactions

► Transactions

► Statistics

► Acquirer main data

► Terminal operator merchant register

► Terminal operator

► User administration

► System configuration

Edit merchant

Changed by 50191150020U650

Dates Start date 06.07.2007 Expire date

Status code 3600 Aktiv

Included in merchant

Hierarchy 2 NIVEAU 02

Role type 4 Aftaleindehaver

Company reg. no. 11111111

Telephone no. 4525252525

Alternative customer no.

Terminal operator merchant no. 1000001

Legal name Fakta 1

Business name Fakta 1

Short name Fakta 1

Address Stensbjergvej 11

Zip code 6200 Aabenraa

Region

Country code DNK Denmark

Language code DAN Danish

Business code

Currency for maximum amount to withhold

Figure 28 shows the top of the edit merchant box and the cancel button.

Figure 29 shows the bottom of the edit merchant box and the save button.

3.5.2 Merchant objects

Settlement agreements, bank account, output recipients and customer groups can be attached to a merchant. All of these have their own space in the overview page and share the same functionality.

The box has a table where the attached objects are shown. The design of the table is the same as described for the result box in the search page. The design is the same to minimize the cognitive burden of the user. When the same design and functionality is the same, the user can recognize the table and he instantly know how it works.

The user can edit the object by clicking the row. This will open the filled edit form. The object can only be opened one time. This is to prevent inconsistent data. It is also possible to delete the object clicking the red delete button. An object open for editing cannot be deleted. The create button opens a form where a new objects can be created. When a new object is created the table showing the attached objects is also updated.

The forms works the same way as described for the edit merchant form.

3.5.2.1 Conceptual design

The following illustrations show the design and functionality of the settlement agreement, bank account, output recipient and customer group boxes. Only the design of the settlement agreement is fully shown.

System configuration

Settlement agreement

Create settlement agreement

Standard agreement no.	Basic standard agreement	Rate agreement no.	Batch method	Start date	Expire date
1 Alm. Aftale med Dagl.advis.	Yes	8 100 procent	1 KIB bundt.	01.01.2011	01.01.2020

Bank accounts

Create bank account

Standard agreement no.	Currency	Account no.	Settlement currency	Start date	Expire date
*	***	111111111111	DKK Danske kroner	15.01.2010	

Output recipients

Create output receiver

Media	Email	Start date	Expire date
9 Advis. e-mail pdf	enevoldsen11@gmail.com	01.01.2011	01.01.2020
4 Adv.enkeltr.(,)	enevoldsen11@gmail.com	01.01.2011	01.01.2020

Customer groups

Add customer group

Customer group	Priority	Start date	Expire date
1 Test	1	15.1.2012	

Figure 30 shows the bottom of the overview page.

System configuration

Settlement agreement

Create settlement agreement

Standard agreement no.	Basic standard agreement	Rate agreement no.	Batch method	Start date	Expire date
1 Alm. Aftale med Dagl.advis.	Yes	8 100 procent	1 KIB bundt.	01.01.2011	01.01.2020

Create settlement agreement

Changed by: 50191150020U650

Dates: Start date: 15.1.2012, Expire date: []

Standard agreement no. []

Rate agreement no. []

Batch method []

Fee payment []

Basic standard agreement

Create

Bank accounts

Create bank account

Standard agreement no.	Currency	Account no.	Settlement currency	Start date	Expire date
*	***	111111111111	DKK Danske kroner	15.01.2010	

Output recipients

Create output receiver

Figure 31 shows the creation of a settlement agreement.

System configuration

Settlement agreement

Create settlement agreement

Standard agreement no.	Basic standard agreement	Rate agreement no.	Batch method	Start date	Expire date
1 Alm. Aftale med Dagl.advis.	Yes	8 100 procent	1 KIB bundt.	01.01.2011	01.01.2020

Create settlement agreement

Changed by: 50191150020U650

Dates: Start date: 15.1.2012, Expire date: []

Standard agreement no.: []

Rate agreement no.: []

Batch method: []

Fee payment: []

Basic standard agreement:

Create

Bank accounts

Create bank account

Standard agreement no.	Currency	Account no.	Settlement currency	Start date	Expire date
*	***	111111111111	DKK Danske kroner	15.01.2010	

Output recipients

Create output receiver

Figure 32 shows error messages from the create settlement agreement.

System configuration

Settlement agreement

Create settlement agreement

Standard agreement no.	Basic standard agreement	Rate agreement no.	Batch method	Start date	Expire date
1 Alm. Aftale med Dagl.advis.	Yes	8 100 procent	1 KIB bundt.	01.01.2011	01.01.2020

Edit settlement agreement

Changed by: 50191150020U650

Dates: Start date: 01.01.2011, Expire date: 01.01.2020

Standard agreement no.: 1 Alm. Aftale med Dagl.advis.

Rate agreement no.: 8 100 procent

Batch method: 1 KIB bundt.

Fee payment: []

Basic standard agreement:

Save

Bank accounts

Create bank account

Standard agreement no.	Currency	Account no.	Settlement currency	Start date	Expire date
*	***	111111111111	DKK Danske kroner	15.01.2010	

Output recipients

Create output receiver

Figure 33 shows the edit settlement agreement.

The prototype is implemented using the Java web application framework Play!.⁷ The different pages are implemented in HTML styled using CCS style sheets. All interaction in the pages are implemented with JavaScript. For implementing JavaScript in the prototype the jQuery library was used.⁸

The prototype has no connection to the real backend system. So the prototype runs with test data saved in a transient in memory database provided by the Play! Framework.

To install the application please see Appendix D Installation guide.

⁷ <http://www.playframework.org/>

⁸ <http://jquery.com/>

The following chapter will describe the evaluation of the prototype.

5.1 Usability test

The prototype is tested to see how the users interact with the new design. The usability test should determine whether the prototype can be used to achieve the tasks the way it was designed to do. Some users are asked to complete some scenarios using the prototype. Their behavior is observed to learn if they are using the tool as the design expects them to. Also their comments and overall experience will be used in the evaluation.

In the usability test the users is asked to change the merchants address and add another output recipient using the prototype and following the change request workflow described in chapter 2.1 The CAPS System.

Scenario

A merchant sends an email to customer service providing a new address and a new email to send notifications to. The merchant has settlement agreements with two different acquirers so he also provides the acquirer numbers and associated merchant numbers.⁹

5.1.1 Test results

The users are very happy with the look and feel of the new user interface. Most find the new interface more satisfying to use. The users especially like that all things can be controlled from the overview page. Some issues was found observing the users use the prototype. These are listed in the table below.

⁹ A video of this usability test is available on the cd. See Appendix C CD content. [D] Video: *usability_test.avi*

The issues are labeled with a severity which identifies how big the problem is.

Severity	Issue	Recommendation
High	The users do not use the auto completion. They tend to use the dropdown function instead.	There should be some kind of help text to let the user know that he can perform a search.
Medium	Some users find it annoying that the workspace tabs are not visible if one has to scroll the page.	Move the scroll bar inside the tab area. This will make the workspace tab stay in the top of the screen.
Low	The customer groups are highlighted as they are editable but customer groups cannot be edited.	Remove hover effect from customer groups.
Medium	The users do not use the built in short cuts.	In the welcome page the new design and shortcuts could be introduced. Also a dedicated help page could be a solution.
Medium	Some users find it annoying that one has to fold out the menus.	When the users click the acquirer menu point it will automatically open the search page.

Table 2 shows issue observed in the usability test.

5.2 Analytic evaluation

An analytic evaluation is an evaluation where a usability expert inspects the prototype and compares it with usability principles. These usability principles are when used in an analytic evaluation called heuristics. The heuristics are scored and the expert gives his comments and recommendations. The original set of heuristics for evaluation user interfaces were developed by Jakob Nielsen and his colleagues. A revised version of these is used in the evaluation.¹⁰

The grading in this evaluation is from 0-1 where 1 completely satisfies the heuristic.

Visibility of system status

The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

Evaluation:

User is always able to find out where he is in the application. The current workspace is highlighted and the menu point the user is currently using is bold. When the system is loading a load bar appears to show the user that something is happening.

Score: 1.0

Match between system and the real world

The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

Evaluation:

The terms and language used through the whole application matches the language used in the acquiring business. So the users should be able to understand labels and menu points.

Score: 1.0

User control and freedom

Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

¹⁰ [4] Nielsen, Jakob. (2005). Ten *Usability Heuristics*. [online]. Available from: http://www.useit.com/papers/heuristic/heuristic_list.html [Accessed 10 January 2012].

Evaluation:

Generally the user has good possibilities to go back and forth between screens without losing the current state of the application. E.g. the workspace remembers the current acquirer. The user can cancel the creation and editing of bank account, output recipients etc.

There is one exception. If the user does not click the right merchant in the result list after doing a search there is no way back. The user has to do the whole search again. There should be some kind of undo in this case.

Score: 0.9

Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

Evaluation:

The different pages, actions, words always mean the same thing.

Score: 1.0

Error prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.

Evaluation:

Error prevention is implemented in the design some places. When a user is editing an output recipient this is object is displayed in the table and has a delete button. But if the button is clicked nothing happens. The same output recipient cannot be opened at the same time.

Error prevention could have been used when filling in forms. Here errors could be prevented by checking if required fields are empty real time. An alternative solution is to make a help text telling the user from the start that this is a required field.

Score: 0.8

Recognition rather than recall

Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

Evaluation:

All actions and options are visible and important information is remembered from screen to screen.

Help texts is only provided in the search page. It is not clear for the users how to use the tabs and auto completion. A little help text to get them started would be good practice.

Score: 0.5

Flexibility and efficiency of use

Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

Evaluation:

All the auto completing combo fields are accelerators. The experienced user knows exactly what to write in these and can do it quicker with the help of auto completion. The inexperienced will use the drop down functionality to find the value.

Score: 1.0

Aesthetic and minimalist design

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

Evaluation:

Every page has an aesthetic and minimalist design. Only relevant data is shown.

Score: 1.0

Help users recognize, diagnose, and recover from errors

Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

Evaluation:

When the user submits a form and something is missing the application will mark the field to show something is missing. If e.g. an email address does not have the correct format message informing the user will be shown.

Score: 1.0

Help and documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Evaluation:

There is no documentation built in the system.

Score: 0.0

5.3 Statistic evaluation

Here the number of mouse clicks and time for completing the task used in the usability test is compared against data gathered from similar scenarios using the old CAPS system.¹¹

When comparing the time it takes to complete the same task using the prototype and the current CAPS user interface, we see that the new interface is in average 23 seconds quicker. See the diagram below.

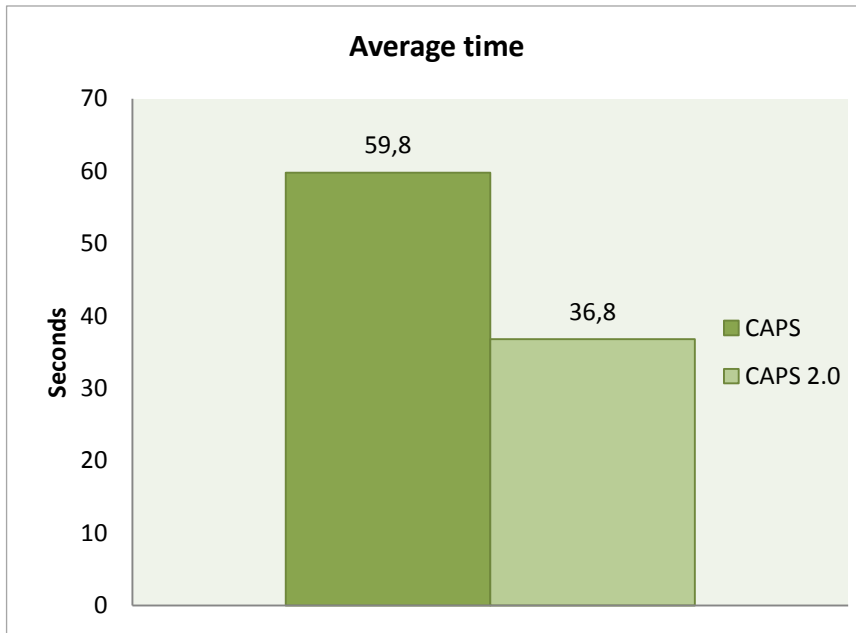


Figure 34 shows the comparison of time used to do the test task.

When it comes to the number of mouse clicks the user had to use to complete the task, we see that the users in average used approximately 3 mouse clicks less when using the new user interface. See the diagram below.

¹¹ The data can be seen in Appendix B Evaluation data.

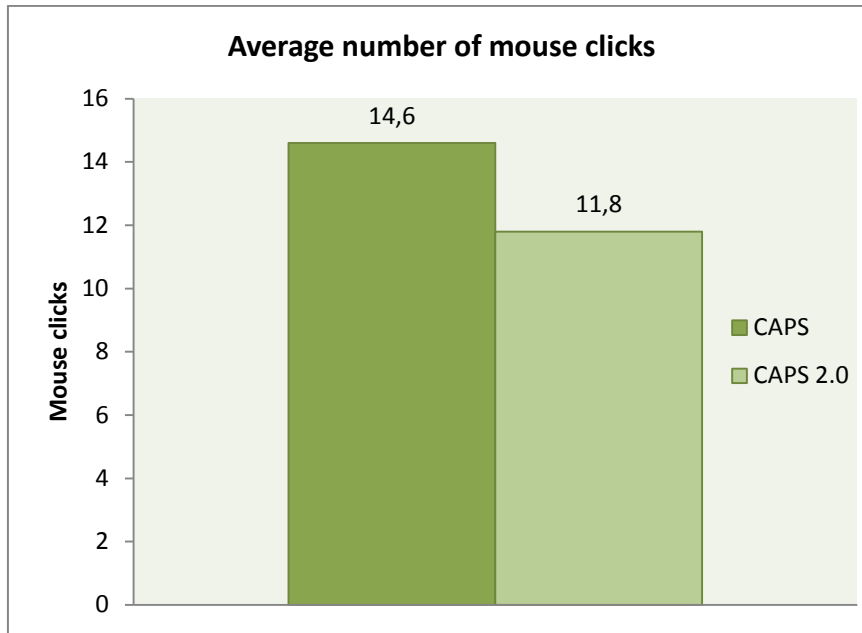


Figure 35 shows the comparison of mouse clicks used to finish the test task.

The results show that the new user interface has increased the performance of the change request workflow. With the new design the user has to go through fewer screens and use fewer mouse clicks to complete a given task..

The new user interface clearly improves the change request workflow. The evaluation shows that a lot of time is saved when doing a simple task. This is primarily due to the new overview page where all the merchants' data can be changed and viewed. If the solution were to be implemented in the real business a lot of man hours used to handle the merchants requests could be saved.

This project really shows the power of the user centered design process. If the user research is conducted properly and the data collected is interpreted the right way, the chance of making an effective user interface fitting the users actual needs is very high.

The workspace solution is a good idea but one could argue that the users should just use the built in tabs in their browser. With the current state of the prototype this argument is true but later on the workspace idea could become powerful. The order and number of workspaces could be saved on the user's profile. Furthermore the items shown in the overview page and the order of these could be saved for each workspace. This way the solution would be more customizable and could fit the individual user's needs. When opening the application the user's preferred workspaces could be loaded from the start.

The user and expert evaluation shows that there are still things that could improve the prototype in the next iteration of the interface design process. The user's does not use the built in shortcuts intended. This must be improved as it has a big potential to save resources. There some other issues but nothing that cannot easily be fixed. Generally the users are very happy with the new design.

The ideas from this project are to be presented for the CAPS product owners. And they will decide if the concepts from this project are to be implemented for the CAPS product. The work done in this project only covers a small part of the CAPS. So it would be obvious to look in to how other group of users use CAPS in their work to make the new design fit to all user's needs.

This thesis presents a new user interface for the CAPS system. Due to the size of existing CAPS systems the scope of the project was defined to focus on how to optimize a specific work flow providing a smarter designed user interface.

The problem definition in the introduction saying, *“The new user interface should increase the effectiveness of the users work and be more pleasant to work with.”* has been analyzed and a solid solution was made.

The project was done using following the user centered design process and describes the first iteration of this process.

The users were observed to map their needs and expectations to a new user interface. From the observations specific requirements for the user interface were made. The requirements were analyzed and a solution fitting the requirements was chosen. The solution and requirements were then mapped into use cases describing the how the new user interface should behave.

All use cases defined in the analysis were designed and described in the conceptual design and later implemented in a running prototype.

An evaluation was conducted to show that the new user interface in fact is better than the old. This is proved by the user’s satisfaction using the new user interface and the comparison made between the two systems.

Literature

- [1] UsabilityNet. (2006). *ISO 1347*. [online]. Available from: <http://www.usabilitynet.org/tools/13407stds.htm> [Accessed 10 January 2012].
- [2] Tidwell, Jenifer. (2011) *Designing Interfaces*. 2nd Edition. Sebastopol: O'Reilly.
- [3] Rogers, Yvonne. Sharp, Helen. And Preece, Jenny. (2007) *Interaction design : beyond human-computer interaction*. 2nd Edition. West Sussex: Wiley.
- [4] Nielsen, Jakob. (2005). *Ten Usability Heuristics*. [online]. Available from: http://www.useit.com/papers/heuristic/heuristic_list.html [Accessed 10 January 2012].

A.1 Remove workspace

Use case: Remove workspace

Actors: User, CAPS, Backend

Summary: The user wants to remove a workspace he is not using at the moment.

Pre-condition: CAPS is loaded and more than one workspace is open.

Basic course of events:

- 1 The user clicks the close workspace button at the tab he wants to close.
- 2 The system removes the particular workspace from the screen.

Post-condition: The unwanted workspace is removed from the screen.

A.2 List settlement agreements

Use case: List settlement agreements

Actor: User, CAPS, Backend system

Summary: The user wants to see the settlement agreements attached to a merchant.

Pre-condition: The user has found the merchant he wants to see. (*Show merchant overview*)

Basic course of events:

- 1 The system lists the settlement agreements for the merchant.

Extensions:

- 1a The system gives an error and shows a relevant error message.

Post-condition: The settlement agreements are listed in the overview page.

A.3 Edit settlement agreement

Use case: Edit settlement agreement

Actor: User, CAPS, Backend system

Summary: The user wants to edit a settlement agreement for a merchant.

Pre-condition: The user has listed the settlement agreements for a merchant. (*List settlement agreements*)

Basic course of events:

- 1 The user presses the settlement agreement he wants to edit.

2 The systems show the settlement agreement form.

3 The user edits the information and saves.

Extensions:

2a The system gives an error and shows a relevant error message.

3b The system gives an error and shows a relevant error message.

Post-condition: The user has edited a settlement agreement for the merchant.

A.4 Create settlement agreement

Use case: Create settlement agreement

Actor: User, CAPS, Backend system

Summary: The user wants to attach a new settlement agreement to a merchant.

Pre-condition: The user has listed the settlement agreements for a merchant. (*List settlement agreements*)

Basic course of events:

1 The user presses the create settlement agreement button.

2 The systems show the settlement agreement form.

3 The user types in data and saves.

Extensions:

2a The system gives an error and shows a relevant error message.

3b The system gives an error and shows a relevant error message.

Post-condition: The user has created and attached a new settlement agreement to the merchant.

A.5 List bank accounts

Use case: List bank accounts

Actor: User, CAPS, Backend system

Summary: The user wants to see the bank accounts attached to a merchant.

Pre-condition: The user has found the merchant he wants to see. (*Show merchant overview*)

Basic course of events:

1 The system lists the bank accounts for the merchant.

Extensions:

1a The system gives an error and shows a relevant error message.

Post-condition: The bank accounts attached to the merchant are listed in the overview page.

A.6 Edit bank account

Use case: Edit bank account

Actor: User, CAPS, Backend system

Summary: The user wants to edit a bank account attached to a merchant.

Pre-condition: The user has listed the bank accounts attached to the merchant. (*List settlement agreements*)

Basic course of events:

- 1 The user presses the bank account he wants to edit.
- 2 The systems show the settlement agreement form.
- 3 The user edits the information and saves.

Extensions:

- 2a The system gives an error and shows a relevant error message.
- 3b The system gives an error and shows a relevant error message.

Post-condition: The user has edited a settlement agreement for the merchant.

A.7 Create bank account

Use case: Create bank account

Actor: User, CAPS, Backend system

Summary: The user wants to attach a new bank account to a merchant.

Pre-condition: The user has listed the bank accounts for a merchant. (*List bank accounts*)

Basic course of events:

- 1 The user presses the create bank account button.
- 2 The systems show the bank account form.
- 3 The user types in data and saves.

Extensions:

- 2a The system gives an error and shows a relevant error message.
- 3b The system gives an error and shows a relevant error message.

Post-condition: The user has created and attached a new bank account to the merchant.

A.8 List merchant's customer groups

Use case: List merchant's customer groups

Actor: User, CAPS, Backend system

Summary: The user wants to see all customer groups a merchant is member of.

Pre-condition: The user has found the merchant he wants to see the customer groups for. (*Show merchant overview*)

Basic course of events:

- 1 The systems shows list of the attached customer groups for the merchant.

Extensions:

- 1a The system gives an error and shows a relevant error message.

Post-condition: The user sees the customer groups for the merchant.

A.9 Add merchant to customer group

Use case: Add merchant to customer group

Actor: User, CAPS, Backend system

Summary: The user wants to add a merchant to a customer group.

Pre-condition: The user has listed the customer groups for a merchant. (List merchant's customer groups)

Basic course of events:

- 1 The user presses the add customer group button.
- 2 The systems the customer group form.
- 3 The system adds the merchant to the customer group.

Extensions:

- 3b The customer group does not exist and the system show a relevant error message.
- 3c The system gives some kind of error and shows a relevant error message.

Post-condition: The user has added a merchant to a customer group.

CAPS 2.0	Test 1	Test 2	Test 3	Test 4	Test 5
<i>Time in seconds</i>	50	35	33	30	36
<i>Mouse clicks</i>	13	9	11	13	13

Table 3 shows time and mouse clicks used for the test of CAPS 2.0

CAPS	Test 1	Test 2	Test 3	Test 4	Test 5
<i>Time in seconds</i>	60	55	57	61	66
<i>Mouse clicks</i>	17	10	15	17	17

Table 4 shows time and mouse clicks used for the test if CAPS.

[A] PDF version of this thesis: *dip10_84.pdf*

[B] Source code: *caps-prototype.zip*

[C] Video: *caps_example.avi*

[D] Video: *usability_test.avi*

This guide will help you to run the CAPS prototype. *Note that Java should be installed on your machine.*

1. Download the Play! Framework and unzip it on your computer.¹² *C:\ would be preferable.*
2. Unzip the source code, caps-prototype.zip in your play-1.2.4 folder.
3. Open the command prompt and navigate to your play-1.2.4 folder. *C:\play-1.2.4*
4. Use the command *play start caps-prototype* to start the application.
5. Open your browser and enter <http://localhost:9000>.

Enjoy!

¹² <http://www.playframework.org/download>