
UNICONTA

New Online Systems ApS



Sebastian Damgaard, Programmer
e-mail: sd@uniconta.com

SEBASTIAN DAMGAARD
ERHVERVSAKADEMI SJÆLLAND
INTERNSHIP REPORT
20/01/17-14/04/17

Table of content

Introduction.....	2
The Company.....	2
What is ERP?.....	2
Development Department	3
Development Method and Environment	4
Development Method	4
Trello.....	5
Meetings and Planning	6
Development Environment	7
Architecture.....	7
Tools	7
Work Assignments.....	8
CVR/VAT API	8
Payment Formats	9
Export e-economic vouchers and import them to Uniconta.....	10
Making the importing tools part of Uniconta.....	11
Employee Commission	12
Reflection.....	12
Assignments.....	12
Workflow	13
Denmark vs. India	13
Conclusion	14
Bibliography.....	15
Appendix.....	15
View	15
Controller 1.....	16
Controller 2.....	16
Model	16
Letter of Recommendation:	17

Introduction

I did my internship at the company New Online Systems ApS, which I was lucky enough to get because my dad, Erik Damgaard, is CEO and lead developer¹. The fact that he was my dad had me considering whether it would be a good or a bad idea, seeing that some people may consider it “an easy route”. Since he is a great developer, I decided that it would still be a great learning experience.

I started at their office in Ballerup where I met the sales and support team. There was only one other developer here since the rest were in India and the lead developer was in Brazil. The Danish developer, Lasse, and the lead developer, Erik, did all the business logic while the Indians did all the User Interface (UI) logic. On the 08/03/17 I therefore went to India for the remainder of the internship, so I could learn and better myself in both areas.

The Company

The company has made a “*lightning-fast cloud-based ERP system*”. It is founded and created by Erik Damgaard who, with his thirty years’ experience in developing accounting systems, has now launched Uniconta which aims to “*make it accessible and simple for the user while maintaining the highest technological development and business practicality*”².

What is ERP?

ERP (Enterprise Resource Planning) is used for core business processes through software and technology. There are a lot of different ERP systems with different features, such as finance, accounting and bookkeeping, marketing and sales, production planning and more. The big companies will need ERP systems with a lot of functionalities while the smaller companies will go for simplistic ERP systems with less functionalities.

¹ I will refer to him as Erik, lead developer or CEO to remain professional.

² Source: <http://www.uniconta.com/en/about-us/>

Uniconta is trying to position itself in the middle of this spectrum, where small companies can just do their accounting work and bigger companies can do inventory management and more as well.

This is done while upholding the accessible and straightforward design for the user.

Item	Item name	Item type	Cost price m...	Cost price	Sales price...	Sales price...	Sales price...	Purchase...	In stock	Unit	D
Bremsekabel	Bremsekabel per meter	Item	Fixed cost price	5.00	20.00				-9.00	Meter	
Cykel01	Cykel Standard	Item	Fixed cost price	1,000.00	5,000.00				122.00	Pcs	
Cykel02	Specialized Tarmac SL4	Item	Fixed cost price	10,000.00	27,000.00				-25.00	Quant...	
SlangerMTB	10 stk slanger MTB	Item	Fixed cost price	50.00	200.00				147.00	Packa...	

Figure 1 shows a company's item inventory screen.

Development Department

As mentioned in the beginning, the development department was split into two areas. In Denmark and in Brazil, we were creating business logic/functionality for the users. This could be creating payment files for the various banks format, which was one of my tasks while doing the internship. Then there is the development team in India where we create all the user interface (picture of UI can be seen in figure 1). The whole architecture and almost all the business logic is done by Erik, but after the launch, Erik spends a lot of time communicating with users, fixing errors, handling requests and so forth.

The image shows two side-by-side screenshots of CSV files. The left window is titled "BEC Format Test i CSV" and the right is "Nordea Format Test i CSV". Both windows display a grid of data with columns numbered 1 through 9. The data in the BEC format includes account numbers, dates, and amounts. The Nordea format includes similar data but also includes bank names and reference numbers.

Figure 2 shows two different bank payment formats. On the left is BEC, which is used by various banks, and on the right, is Nordea's bank format.

Development Method and Environment

The work/development method was special in Uniconta and very different from what I know from school. Since the product owner is also a developer, it meant that there was no work method, at least what we are familiar with. The environment we were working in was Visual Studio, .NET and C#, which we know well from school.

Development Method

As mentioned above, the lead developer is also the project owner, which means that the different tasks would be given from him. I believe you could define it as a self-crafted agile process, where the lead developer handles the value of the "user stories"³ or user requests. The developers then get a new assignment when finished with the previous assignment. The closest agile approach to this I believe is XP (Extreme Programming), which is very core software engineering based, but as the Agile samurai says, "I will be sharing with you teachings and innovations from all the agile

³ A user story is an informal description of one or more features of a software system from the perspective of a user.

*methods and several we had to invent ourselves. Read them, study them, challenge them, and take from them what you need*⁴. Meaning only your company/project can know what is right for you.

Trello

In the beginning, we didn't use any SCRUM or Kanban board. As time progressed, we got recommended a tool called Trello, which is *"boards, lists, and cards that enable you to organize and prioritize your projects in a fun, flexible and rewarding way"*⁵. Trello is like the SCRUM board but can be used by support and sales as well with their tasks. It was still in the testing face when I left, but as you see in figure 3 below, we have lists for errors, requests, postponed, in progress and update⁶. Inside the list is then a card that shows the title of a task and as seen in figure 4 below, you can go into the card, read a description, add an attachment, comment on it and more. That it came so late in Uniconta's life, was due to Uniconta getting bigger and Erik giving more tasks to us other developers. Meaning that we needed a workplace where we could see all assignments and errors. After understanding and creating the Trello board, it was unfortunately nearing the end of my internship.

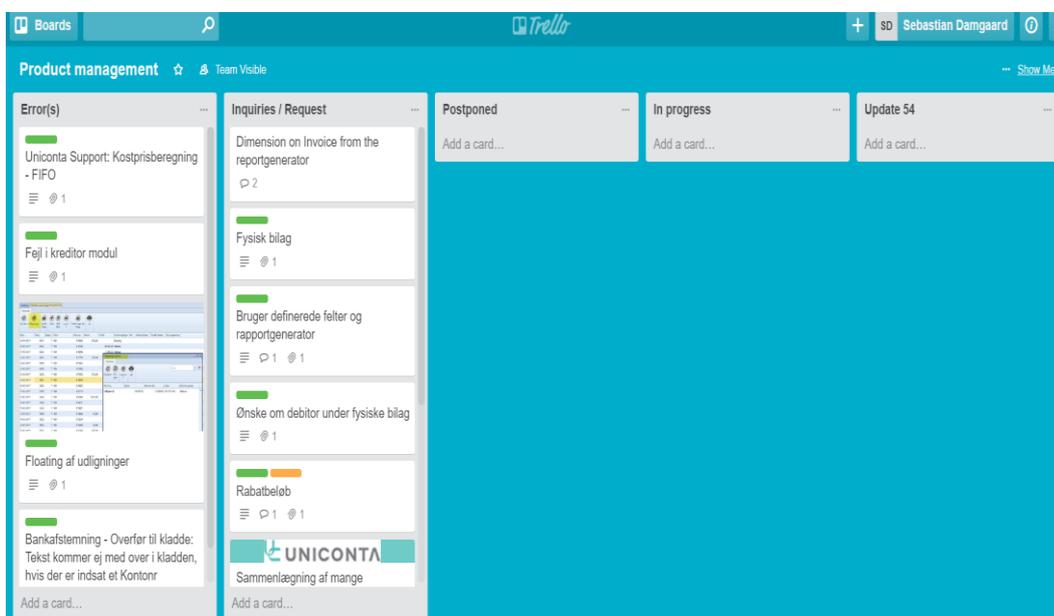


Figure 3 shows Uniconta Trello board with lists and cards

⁴ Source: The Agile Samurai: How Agile Masters Deliver Great Software, page 24

⁵ Source: <https://trello.com>

⁶ Update means it is ready for the next updated Uniconta version release.

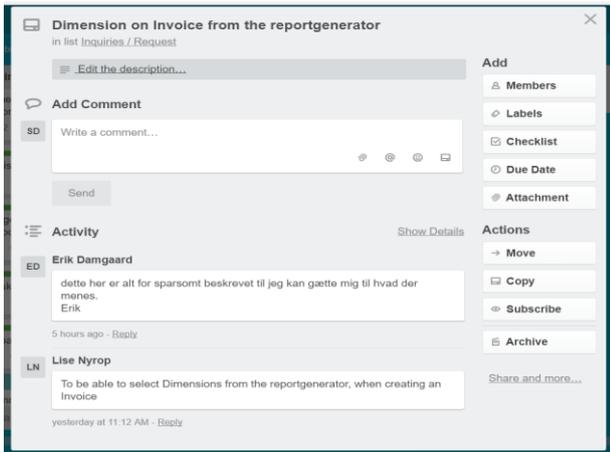


Figure 4 shows an example of what a card could look like

Meetings and Planning

The first one and a half months, when I was working in Denmark, we didn't have any development meetings but a meeting with support and the developers (Lasse and me) once every 1-2 weeks. Here the support team would get feedback on what partners and customers thought of their support. We would also talk about the communication between development and support, since support often reported errors or request to the developers. The development team would talk about what they had been working on this week and what they would be doing the next week, what setbacks we had and what was ready for release/next update.

The other meeting, was a monthly meeting where sales, support, development and executives would meet. Sales would talk about the new user count, revenue, partners and all the stuff that would lift Uniconta's⁷ revenue and popularity. Support would talk about the issues mostly discussed with the users, educational seminars and more. We, the programming team, would then talk about the assignments we had done, what could be done to make our workflow better (this is how the use of Trello came to be) and feedback from the users of the new features we had created.

These meetings would work as our retrospect of the week/month and would be used to optimize our workflow.

⁷ Since Uniconta is the only product of New Online Systems ApS, I will be referring the company's name as Uniconta.

The team in India is not a part of these meetings. They would only communicate with the other developers. Mostly if we needed them to create a module/page or other UI assignments. This meant that the meetings would not really help them.

Development Environment

The development was very like what I was taught and worked with in school. Working with Visual Studio (VS), .NET and C# amongst working with API's, Team Foundation and MSSQL. There was one tool that I had not heard of before which is called DevExpress.

Architecture

All the development is done in Visual Studio in one solution. It is separated into different projects because of the size of the solution where each project has their tasks to do. The SQL project communicates with the database and creates the different classes, a link is then added to the API, which is used by the Silverlight and Windows API to create a model class, that has the properties the WPF and Silverlight Client uses.

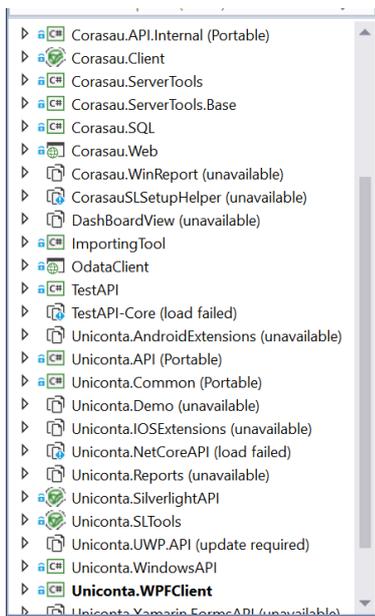


Figure 5 shows some of the Uniconta solution. Corasau is the old name of Uniconta.

Tools

One of the tools Uniconta use is Microsoft SQL Server Management Studio where the database and tables are created and maintained. While I was working with Uniconta, I only got to add fields to one table. Erik was the only one working on the database because he was cautious of giving

“the responsibility” to the other developers. In one assignment, I needed a new table we then created together, so I could see how he had designed the database and how it was connected to the Uniconta solution.

Another tool is Team Foundation Server which we used to maintain version control. Team Foundation creates a workplace on your computer and every change you make, can then be pending for the other team members to get to their workplace. Other than that, if two members have edited in the same class, it will tell you and make sure you don't override another member's code.

Then there is DevExpress, which is a company that creates UI controls for WinForms, WPF, ASP.Net, Silverlight and much more, to create a nice user experience. Some of the popular controls offered are Ribbons, DataGrid and PDFViewer. They also offer more than 40 custom themes for business applications as well as all controls can be localized using satellite resource assemblies.

Work Assignments

When starting I was working with an assignment given by Erik but when I was done a problem had arisen. The other programmer, Lasse, who worked from Denmark, was sick. This meant that the first one and a half weeks he wasn't there. When I then had to get a new assignment, the non-programming employees threw a lot of economical assignments my way. Since they couldn't explain, programming wise, what they wanted and because they had so many specific requests, it became overwhelming with all the tasks at hand. We managed to sort it out and when Lasse came back, I could ask him for help if there was something I didn't understand.

CVR/VAT API

The first task was to implement a CVR/VAT API that could retrieve a company's information by typing their CVR/VAT number. This had been a customer request for a while so they had already found an API I could use which was CVRAPI.dk.

The first version of the CVR implementation made it possible for the users to input their CVR, the places where it needed to be filled out, and the rest of the information, such as company name, email, phone, address etc., would then be filled out automatically.

This was a free service so when the users wanted more information, Uniconta then bought a license to their REST service. This made it possible to get more information, search by more criteria's and get a list of companies in return when searching by name, since it before just returned the first one found.

```
1 {
2   "vat": "33266928",
3   "address": {},
22  "companyform": {},
27  "companystatus": {},
31  "contact": {},
37  "status": {},
43  "industrycode": {},
47  "life": {
48    "start": "2010-11-17",
49    "end": null,
50    "name": "New Online Systems ApS",
51    "adprotected": true
52  },
53  "accounting": {},
342 "capital": {},
348 "shareholder": {},
352 "info": {},
362 "secondarynames": [],
365 "industrycodes": [],
366 "subsidiaries": [],
369 "employment": [],
523 "participants": []
859 }
```

Figure 6 shows what the REST service returns when searching with CVR/VAT (only life is expanded because of space)

Payment Formats

Perhaps the biggest request from the users I managed, was the need to write their payments to a file that then could be read and accepted by various banks. This was something the users needed to run their company and Uniconta was missing this feature.

I started by researching what the type of file I needed to create was, which was a comma-separated-file (CSV) or a fixed length file. It was tricky to figure out how to make it most efficient since there had to be around 100 fields inside these files and the payment looked different depending on if it was a standard, "Indbetalingskort"⁸, foreign payment etc. I choose to do the code using the architecture Model-View-Controller (MVC)⁹. In the View the user will select which format they require, it will then go to the first Controller and search the list of payments to check

⁸ Indbetalingskort is a Danish payment way often used for bills.

⁹ Go to appendix to see the architecture with pictures.

what type of payment it is (standard, foreign etc.). It will then go to the second Controller which creates the Model object and writes the payment objects out to a file¹⁰.

The first formats I created was Nordea and Danske Bank but it came apparent that there were more formats that needed to be created. I then restructured everything to make a base class for the Model class. It holds the information that all formats need (amount, from- and to account, currency etc.) and the Model class, with the information for the specific format (Nordea, Jyske Bank etc.) only, then inherits from this base class. I also created a base class for the controllers that checks the information of the payments are correct, creates the parts of the object that is the same for every format and more. I would describe it like the Template Method Design Pattern, where an algorithm can be overridden to allow the different subclasses their own behavior. The base class algorithm would still be followed but would be extended.

```
public class BankFormatBase
{
    public bool SharedCodeForCreateBankFormatFields(CreditorTransPayment tran,
        CreditorPaymentFormat paymentFormat,
        BankStatement bankAccount, DanishFormatFieldBase danishFields) {...}

    public bool SharedCheckIBAN(DanishFormatFieldBase field, CreditorTransPayment tran) {...}

    public bool SharedCheckFIK(DanishFormatFieldBase field, CreditorTransPayment tran) {...}

    public string ShortenWordToCriteria(string value, int lengthAllowed) {...}

    public static async Task<bool> CheckBankInfo(CrudAPI api, CreditorPaymentFormat paymentFormat,
        List<CreditorTransPayment> listOfPayments) {...}

    public static async Task<BankStatement> SharedCheckPaymentInfo(List<CreditorTransPayment> trans, CrudAPI api,
        CreditorPaymentFormat paymentFormat) {...}
}
```

Figure 7 shows the base class for the controllers

Export e-economic vouchers and import them to Uniconta

Erik had developed a solution for importing all your data from C5 and e-economic to Uniconta, since a lot of users wanted to convert from these ERP systems to Uniconta. The user of C5 or e-economic would export all their data to a file(s) and the importing tool will then take the information of these files and import them to the right places in Uniconta. The only problem was that their scanned vouchers would not be exported, which holds information that is valuable for

¹⁰ Figure 2 shows what two files might look like.

their business (invoices, bills, documents etc.). I therefore received a task to find out if it was possible to get these vouchers and if it was, then create a solution to import them to Uniconta.

I started by looking at e-conomic's REST and SOAP services. Their REST service was relatively new and lacked a lot of the possibilities but their SOAP service did not. I managed to find a method to get all physical vouchers from e-conomic. I then found a method to get all the accounts and checked which account had a voucher number that matched the voucher number of one of the vouchers previously found. I then used Uniconta's API to add the voucher to the matching account in Uniconta or create the account before adding, if it does not exist.

Making the importing tools part of Uniconta

After the project, to export and import scanned vouchers, was creating, we decided the importing tools should also be built into Uniconta.

One of the Indian programmers, Anurag, made it possible for users to create a company from the imported data in the page where you normally create a company. I then created a screen that would write out the different files and tell when they were done one by one. I added a progress bar to follow the time it would take.

Inside the tool module I then created a page, as seen in figure 8, where you could input the information needed to get the vouchers from e-conomic and as with the other part, which creates the company, it would open a window with an information textbox and a progress bar.

Other than that, the code was already created and basically just needed to be copy pasted.

Import	Terminate	Help
e-conomic SecretToken	<input type="text"/>	
e-conomic GrantToken	<input type="text"/>	
Company name	<input type="text"/>	
From date	01/05/2017	
To date	01/05/2017	

Figure 8 shows the screen where you put in information to get the e-conomic vouchers

Employee Commission

Some customers wanted a page where it would be possible to maintain and calculate employee commission. This was an interesting task because it was the first assignment where I had to create pages from scratch.

The page adds an employee and give him/her an item or item group. When calculating the commission to the employee, it will then go into the item inventory and see the price of the item and the quantity sold by that employee. If the item does not have a price, it will look at the item's item group and see if there is a price there.

The company also have the possibility to just give an item a fixed-price or rate, choose if it is revenue or margin, and the calculation will be done by looking at the criteria's, as seen in figure 9.

```
private CalCommissionClient CalculateCommissionInvTran(InvTransClient tran, EmployeeCommission commission)
{
    double amount;

    if (commission.FixedPrice != 0)
        amount = -commission.FixedPrice * tran._Qty;
    else
    {
        if (commission.IsRevenue)
            amount = tran._NetAmount() * commission._Rate / -100d;
        else
            amount = tran.Margin * commission._Rate / -100d;
    }
    if (amount == 0)
        return null;

    var calCom = new CalCommissionClient();
    calCom._Employee = tran._Employee;
    calCom._Item = tran._Item;
    calCom._Account = tran._DCAccount;
    calCom._InvoiceNumber = tran._InvoiceNumber;
    calCom._Commission = Math.Round(amount, 2);
    return calCom;
}
```

Figure 9 shows how the calculation of the commission is done

Reflection

As mentioned in the introduction, I was nervous going into this because I wasn't sure if it was going to affect me that my dad was the boss. I quickly discovered that it was not going to be an issue. Everyone was very nice to me and the VP's were not afraid to give me assignments, to the extend where I was overwhelmed in the beginning.

Assignments

The programming aspect wasn't that problematic. Both the CVR and getting e-conomic scanned vouchers required the understanding of using REST and SOAP API's. Since we had been working

and learning a lot of this in school, it didn't really become a problem. The tools we used was something I was used to working with as well.

The biggest issue for me was understanding the economical parts of the project. If I had to create the code for writing out to files I had to ask myself, how/where do I find the information they need inside Uniconta? What does Payment ID mean and does it mean the same in Uniconta as the bank? In the beginning the other Danish programmer, Lasse, was sick. This meant that I had no one, other than Erik over skype, that could do the translation of this economical assignment to a programming assignment. This shook my confidence in the beginning and I felt that I could not deliver on the level they expected. When Lasse finally came back, he gave me a warm welcome by explaining what this and that meant, which was relieving because after that I was developing nonstop and loved it.

Workflow

The workflow was something I had to get used to. In SCRUM, there is the SCRUM master who takes care of the talk with the bosses and then "translates" that to the developers. When I was the only programmer, the bosses in Denmark threw many tasks at me. They weren't developers and therefore had no idea the pressure that put on me. This got fixed when Lasse came back from being absent and could be the "SCRUM master" that translates what they wanted to me.

At a meeting, I brought up that it was quite overwhelming getting all these tasks and keeping track of them all, I therefore introduced the use of the SCRUM Board. It seemed like they didn't really take it to heart but two weeks later one of the employees introduced Trello, which I talked about earlier, and they loved it when they saw it in practice. Even though I didn't get to work with it a lot, I think it will help the workflow in Uniconta.

Denmark vs. India

In Denmark, my work was mostly focused on creating and understanding the business logic and following the development in Uniconta from a business perspective.

In India, my work was similar to the work I did in Denmark, the difference was that I got to work with the UI as well and learning DevExpress.

In India my work was very programming focused. We discussed assignments and helped each other but other than that no meetings were held and the business part of Uniconta was replaced with pure programming. It was an interesting way to work because it meant that I furthered my development skills more in India than in Denmark. On the other hand, my knowledge on how a business works and is run came from my time in Denmark.

Conclusion

I feel my programming skills and what I have been taught in school, gave me the knowledge to work out in “the real world”. The only issue where I felt behind when starting was my knowledge of XAML, which the school covered little of. I was taught a lot by the Indian team in this and I had improved my skills gradually when my internship was over.

It felt like the workflow lacked some organization from the development perspective. I believe that they have been used to sending everything to Erik and he would decide the importance and order of the tasks and then do them. Since I had never worked like this before I was a little shocked by all the tasks in the beginning. It helped a lot with Trello since they could put all the tasks in there instead of dumping everything on me.

Finishing this internship left me with an understanding of how important the workflow and communication inside a company is. It was one of the things school has focused on and I understand why it is important. I feel confident in going out to work after I get my degree. My understanding of programming, especially Visual Studio, C# and .NET, covers, to some extent, what is needed to complete tasks for a company, IF(!) the communication of the assignment is done the right way.

Bibliography

1. Zealand Institute of Business and Technology, Curriculum, Computer Science, August 2014
2. <http://anbo-easj.dk/internship/index.html>
3. Extreme Programming: https://en.wikipedia.org/wiki/Extreme_programming
4. ERP: https://en.wikipedia.org/wiki/Enterprise_resource_planning
5. The Agile Samurai: How Agile Masters Deliver Great Software, Jonathan Rasmusson, 2010
6. Uniconta: <http://www.uniconta.com/en/about-us/>
7. Trello: <https://trello.com>
8. DevExpress: <https://www.devexpress.com>

Appendix

View

```
else if (paymMethod == ExportFormatType.Nordea_CSV)
{
    NordeaPaymentFormat.CheckWhichNordeaTransferTask(trans, api, paymentFormatRec);
}
else if (paymMethod == ExportFormatType.DanskeBank_CSV)
{
    DanskeBankPayFormat.CheckIfDanishOrForeignTransferTask(trans, api, paymentFormatRec);
}
else if (paymMethod == ExportFormatType.BankData)
{
    BankDataPayFormat.CheckIfDanishOrForeignBankDataTransferTask(trans, api, paymentFormatRec);
}
else if (paymMethod == ExportFormatType.BEC_CSV)
{
    BECPayFormat.CheckWhichNordeaTransferTask(trans, api, paymentFormatRec);
}
```

Controller 1

```

50         if (tran._PaymentMethod == PaymentTypes.IBAN)
51         {
52             var countryCode = tran.PaymentId.Substring(0, 2);
53             if (countryCode.Contains("DK") || countryCode.Contains("dk"))
54             {
55                 var newProp = fileFormat.CreateDomesticFormatField(tran, paymentFormat, bankStatement);
56                 if (newProp != null) listoNordeaProperties.Add(newProp);
57             }
58             else
59             {
60                 var newProp = fileFormat.CreateForeignFormatField(tran, paymentFormat, bankStatement);
61                 if (newProp != null) listoNordeaProperties.Add(newProp);
62             }
63         }
64     }
65     else if ((tran._PaymentMethod != PaymentTypes.IBAN) &&
66             (tran._PaymentMethod != PaymentTypes.VendorBankAccount))
67     {
68         var newProp = fileFormat.CreateIndbetalingskortFormatField(tran, paymentFormat, bankStatement);
69         if (newProp != null)
70             listoNordeaProperties.Add(newProp);
71     }
72     else if (tran._PaymentMethod == PaymentTypes.VendorBankAccount)
73     {
74         var newProp = fileFormat.CreateDomesticFormatField(tran, paymentFormat, bankStatement);
75         if (newProp != null) listoNordeaProperties.Add(newProp);
76     }
77 }
78 if (listoNordeaProperties.Count > 0)
79 {
80     var sfd = new SaveFileDialog
81     {
82         Filter = UtilFunctions.GetFilteredExtensions(FileExtensionsTypes.CSV)
83     };
84     var userClickedSave = sfd.ShowDialog();
85     if (userClickedSave != true)
86         return;
87     try
88     {
89         using (var stream = sfd.OpenFile())
90         {
91             using (var sw = new StreamWriter(stream))
92             {
93                 fileFormat.StreamToDomesticFile(listoNordeaProperties, sw);
94                 fileFormat.StreamToForeignFile(listoNordeaProperties, sw);
95                 fileFormat.StreamToIndbetalingskortFile(listoNordeaProperties, sw);
96             }
97             stream.Close();
98         }
99     }
100     catch (Exception ex)
101     {
102     }
103 }

```

Controller 2

```

public partial class CreateNordeaFileFormatBase : BankFormatBase
{
    public DanishFormatFieldBase CreateDomesticFormatField(CreditorTransPayment tran,
        CreditorPaymentFormat paymentFormat,
        BankStatement bankAccount) {...}

    public bool CheckDomesticPaymentType(DanishFormatFieldBase field, CreditorTransPayment tran) {...}

    public void ListsThatHoldDomesticInfo(DanishFormatFieldBase field, CreditorTransPayment tran) {...}

    public void StreamToDomesticFile(List<DanishFormatFieldBase> listoOfDanskeBankPayments, StreamWriter sw) {...}
}

```

Model

```

public class NordeaFormatFields : DanishFormatFieldBase
{
    public string TextCode;
    public string BriefFreeAdvice;
    public string StretchedAdvice;
    public string ExchangeRateReference;
    public string CostCode;
    public string ExchangeRate;
}

```

Letter of Recommendation:

Sebastian har været i praktik hos os, New Online Systems, i 3 måneder fra 20/1-2017 til 14/4-2017. De første 1 ½ måneder arbejdede Sebastian på vores kontor i Ballerup og den sidste del af perioden, på vores kontor i New Delhi i Indien.

I New Online Systems udvikler vi programmet Uniconta, som er et online ERP system til små og mellemstore virksomheder.

Første opgave for Sebastian var at lave en integration mellem vores kundekartotek og CVR registeret. Sebastian søgte på egen hånd information om hvordan CVR registerets web API fungerede og lavede en velfungerende integration med Uniconta hvor, når brugeren har indtastet en virksomheds CVR nummer, så blev felterne til navn, adresse m.m. udfyldt automatisk.

Næste opgave var at bygge en fil integration til danske bankers betalings system. Vi viste Sebastian et skærmbillede i Uniconta, hvor fakturaer ligger og venter på at blive betalt. Opgaven gik på at danne betalingsfiler, som kan læses af banken. Sebastian søgte selv informationen og dannede betalingsfiler til 4 forskellige danske banker. Sebastian udbyggede skærmbillederne i Uniconta med menuer og dialoger. Desuden oprettede han en SQL tabel til opsætning af præferencer og gemte data i dem. Til sidst blev alle tekster oprettet i vores sprog database. Sebastian udførte opgaven selvstændigt og med vejledning fra os, når det drejede sig om at følge kode standarder og om at bruge vores interne systemer, som ikke er dokumenteret.

Næste opgave var at lave en integration til konkurrerende ERP produkt, som også kører online. Opgaven var, via konkurrentens API, at hente alle indscannede bilag, som en kunde vil have brug for, hvis han ville skifte fra konkurrentens system og til Uniconta. Sebastian søgte selv informationen om konkurrentens API, og byggede derefter integration. Han udviklede desuden på egen hånd en brugergrænseflade, og via Uniconta's API, blev de indscannede bilag gemt i Uniconta, men en reference til bilags transaktion.

Sidste opgave var at danne en provisions afregning for sælgere i Uniconta systemet, som er baseret på de produkter og fakturaer, som en sælger har ansvar for. Med instruktion for os omkring hvordan beregningen skulle udføres, dannede Sebastian selv skærmbilleder til opsætning af beregnings præference og efterfølgende visning af den beregnede provision.

Alle opgaver blev løst, frigivet i produktet og bruges i dag af vores kunder i det daglige.

Sebastian forstår at omsætte en opgaveformulering til en færdig løsning. Undervejs forstår han at søge information omkring grænseflader og efterfølgende at implementere løsningen, med et objekt orienteret design, som er let at læse og vedligeholde. Jeg kan på det varmeste anbefale Sebastian i enhver virksomhed, som udvikler software i Microsoft teknologier. Han behersker disse teknologier meget fint.

Hilsen

Erik Damgaard Nielsen, Direktør.

