Computer Science - Spring 2018

Final Project - SC-T-404-LOKA

CATS

Project Management System for //JÖKULÁ

Andri Karel Júlíusson

Sigurður Marteinn Lárusson

Skúli Arnarsson

Smári Björn Gunnarsson

Instructor:
Birgir Kristmannsson

Examiner:
Björgvin Sigurðsson

Table of contents

Figures

Tables

Introduction

This report has been prepared as part of our final project for Reykjavík University. The team members are Andri Karel Júlíusson, Sigurður Marteinn Lárusson, Skúli Arnarsson and Smári Björn Gunnarsson. The goal of the project was to create a “full stack” web application titled “Cats”. This web application facilitates planning projects, building a folder structure of projects, creating a list of tasks for each project and provides a space to store data for each individual project. It also provides a time report for time that has been worked on each individual project that can be displayed to customers of //JÖKULÁ as well as the cost of projects through a dedicated log in service for buyers.

1 The Project

1.1 Project Description

The project ("Cats") is a project management and time registration system that contains seven user groups (Roles). These groups include an Administrator, Human resources officer, Project manager, Financial officer, Salesman, Employee, Customer administrator and Customer employee. Each user group has their own dedicated user interface and their own user dashboard. The main focus of this project will be on completing the tasks for three of these user groups, that is the Administrator, Project manager and the Employee. The Customer employee user group will not be implemented but minimal emphasis will be placed on the implementation of the Human resources officer, Financial officer, Salesman and Customer administrator.

The flow of the system should be as follows: The Human Resource officer can create a new user and give him one or more roles. A salesman should then be able to create a sales offer for a project and send it to a customer. If the project is accepted by the customer the salesman marks the sales offer as “accepted” and sends it to a Project manager. The Project manager receives a requirements list and registers the tasks present in the requirements list into the system within the appropriate project. The Project manager then assigns tasks to employees within an appropriate division (Programming, Design, Marketing, etc). When employees within these divisions complete their tasks, they mark them as ready for QA and send them back to the appropriate Project managers dashboard so that he can assess the quality of the delivered task. The Project manager should then be able to send tasks back for improvement along with a message. If the task does not require any further improvement the Project manager can mark the task as “ready for invoice”. The Project manager then reviews the time it took to complete the task and either accepts the hours worked as they are logged or explicitly changed the amount of worked hours to a lower value if they are not justifiable for billing. When the time logged for the tasks has been accepted by a Project manager the tasks should automatically appear on the dashboard of the customer, so that he can monitor the progress of the project in real time. On his dedicated user interface the customer has the chance to make comments or give feedback on time registrations that are made to his project before the invoice is sent for billing. The Financial Officer should then be able to use the aforementioned information to create invoices.

The system in it’s latest state can be accessed on cats.jokula.is.

1.2 Deliverables

These are the deliverables from the work we did on the final project.

1. A front end that is connected to a back end web service and is implemented in accordance to the design provided by //JÖKULÁ.
2. A back end web service that complies to the requirements of the project.
3. A user manual.
4. A development guide
5. Documentation of the API
6. Documentation of the services.

1.3 About //JÖKULÁ

//JÖKULÁ is a company that offers digital marketing solutions, from traditional advertising to complex websites. Within the company, there are 10 employees in three departments as well as contractors. The employees have a considerable amount of knowledge and experience. The company takes on projects in entrepreneurship, text-making, design, programming, market analysis and more.

1.4 Future of the system

We have implemented most of the key features that the system was originally intended to support. The only A priority user stories missing are the ones linked to the salesman aspect of the system. The salesman was not one of the roles that //JÖKULÁ prioritized and we decided not to implement that role fully. This was mainly due to the complexity of the salesman role and its reliance on customers being an active member in the system (which would probably not be implemented). This is something that will most likely be revisited in the future. One of the requirements for the end system is to support a file system. This allows employees to share files with each other by uploading or them onto the system’s file server. Another feature that would be preferable in the future is the ability to allow customers to log into the system. This would allow customers to keep track of projects that //JÖKULÁ is working on for them. This gives them a chance to see if tasks are broken down correctly, if tasks contain a correct description and keep track of the time that is logged on each task from beginning. The customer would also be notified, most likely by email, when the project reaches a certain milestone.
All of the features that are mentioned above (except for the salesman feature) are categorized as B and C requirements and so they were not a priority for us.

1.4.1 Considerations for future development.

- Making sure that all routes are properly inaccessible for unauthorized users.
- Make use of continuous integration and continuous deployment if work continues on the system.
- Perform user testing with a range of users of different ages, technical backgrounds and familiarity with the system.
- Use sockets for real time, bidirectional communication between web clients and servers, to instantly notify and update information for users as soon as changes occur the system.
- Use Vuex state manager to avoid passing too many props to child components and to keep a centralized persistent store of information, accessible to users with correct access rights.
- If the system were to be brought to a larger scale we would like the UI to be viewable in tablets and mobile devices, instead of simply having a desktop first design.

2 User groups

There are several user groups that interact with the system. These user groups are implemented explicitly in the system in the form of roles. Each user is assigned one or more roles at the time when a new user is registered in the system. These user groups all serve a different purpose within the system and have different permissions, privileges and user interfaces. The user groups and a description for each user group can be seen in the table below (Table 2.1):

User Group	Description	Implemented
Admin	Administrator of the system, can create new users, divisions and change a users role. Is essentially a master role with all permissions.	Yes
Customer Administrator	This group can create Customer accounts for other employees of the same company to see the status of an ongoing project. Can send messages to the company and can see contract details.	No
Customer	This group can view the status of projects he/she has access to.	Yes, partially
Employee	An employee working at the company using the system. Can see tasks assigned to him/her and related projects. Can clock in and out of tasks.	Yes
Financial Officer	Can view the status of projects and tasks and see the charged hours for each task. Can view customers and the contacts for each customer.	Yes, partially
Human Resources Officer	An employee of the Human resources department of the company. Can create, edit and delete employees as well as view information about them.	Yes
Project Manager	The manager of a project. Can manipulate projects and tasks (create, edit or delete). He can also assign tasks to employees and view time reports for each project. Finally since the Salesman was not fully implemented, the Project manager can also create, edit and delete customers.	Yes
Salesman	A salesman of the company, can create, edit and delete customers. Was not fully implemented.	Yes, partially

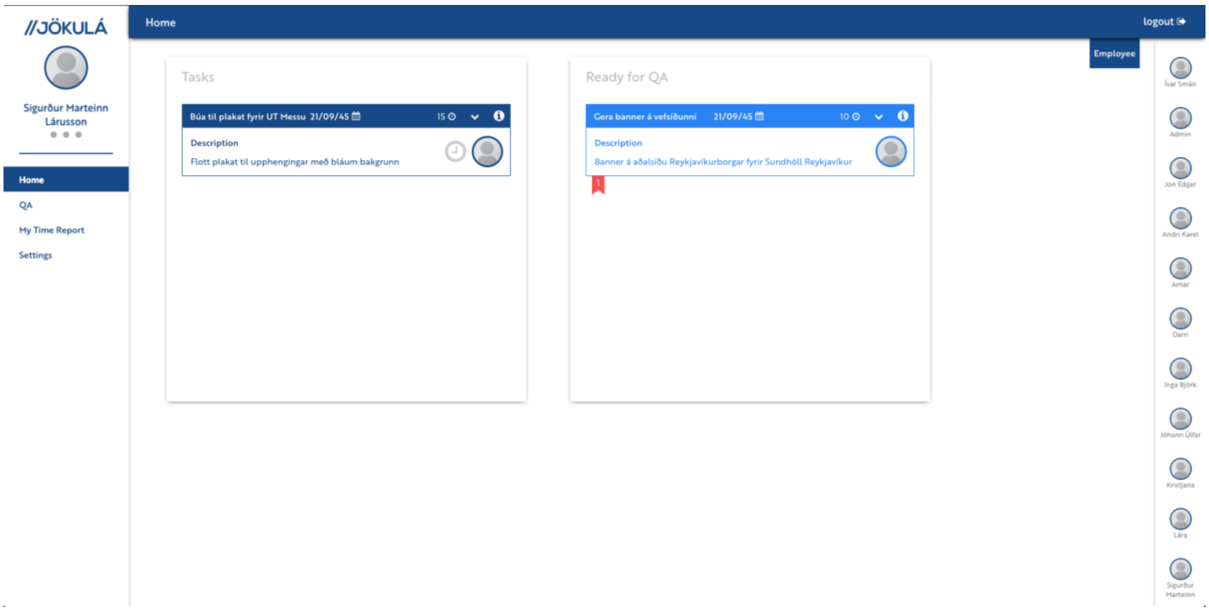
Table 2.1: User groups

3 System Overview

This section shows the main functionality of the system in detail with accompanying images for reference. This can be seen as an enumeration of all the features of the system and not as a manual. For instruction on how to use the system please see the User Manual.

3.1 Employee

The employee home page(seen in figure 3.1) has two boxes. The one on the left has the employees assigned tasks and on the box on the right the tasks he has completed and sent for QA (Quality Assurance). There may be a red banner at the bottom of the “ready for QA” tasks implying they have already been sent to QA before and returned back to the employee for improvement.

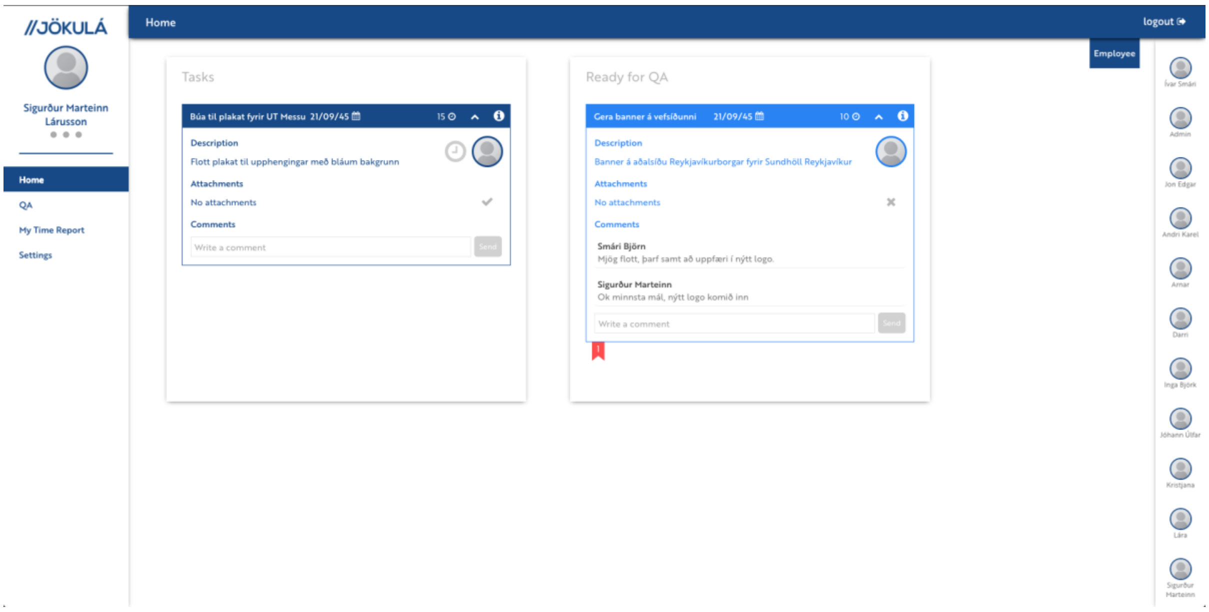


Búa til plakat fyrir UT Messu 21/09/45 15
Description
Flott plakat til upphengingar með bláum bakgrunn

Gera banner á vefsíðunni 21/09/45 10
Description
Banner á aðalsíðu Reykjavíkurborgar fyrir Sundhöll Reykjavíkur

Figure 3.1: The employees homepage

If the employee presses the arrow down button in the header of the tile, he expands the tile showing additional information (see figure 3.2). The employee can then see attachments and comments, if there are any. As well as either a cross or check mark, used to either send the task to or from ready for QA.

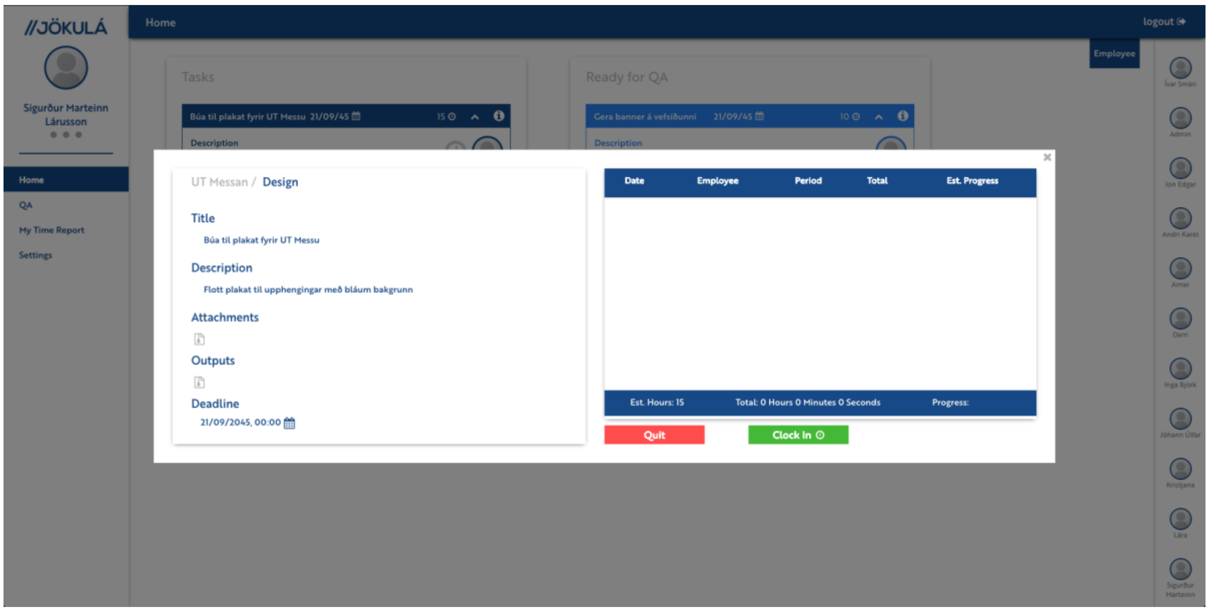


Búa til plakat fyrir UT Messu 21/09/45 15
Description
Flott plakat til upphengingar með bláum bakgrunn
Attachments
No attachments
Comments
Write a comment Send

Gera banner á vefsíðunni 21/09/45 10
Description
Banner á aðalsíðu Reykjavíkurborgar fyrir Sundhöll Reykjavíkur
Attachments
No attachments
Comments
Smári Björn
Mjög flott, þarf samt að uppfæri í nýtt logo.
Sigurður Marteinn
Ok minnsta mál, nýtt logo komið inn
Write a comment Send

Figure 3.2: Expanded tiles employee homepage

Figure 3.3 is the view that the employee will get after pressing the information button on the task he wishes to clock into. He can then click on "clock in" on the lower right hand corner which creates a new timestamp and sets its starting time.



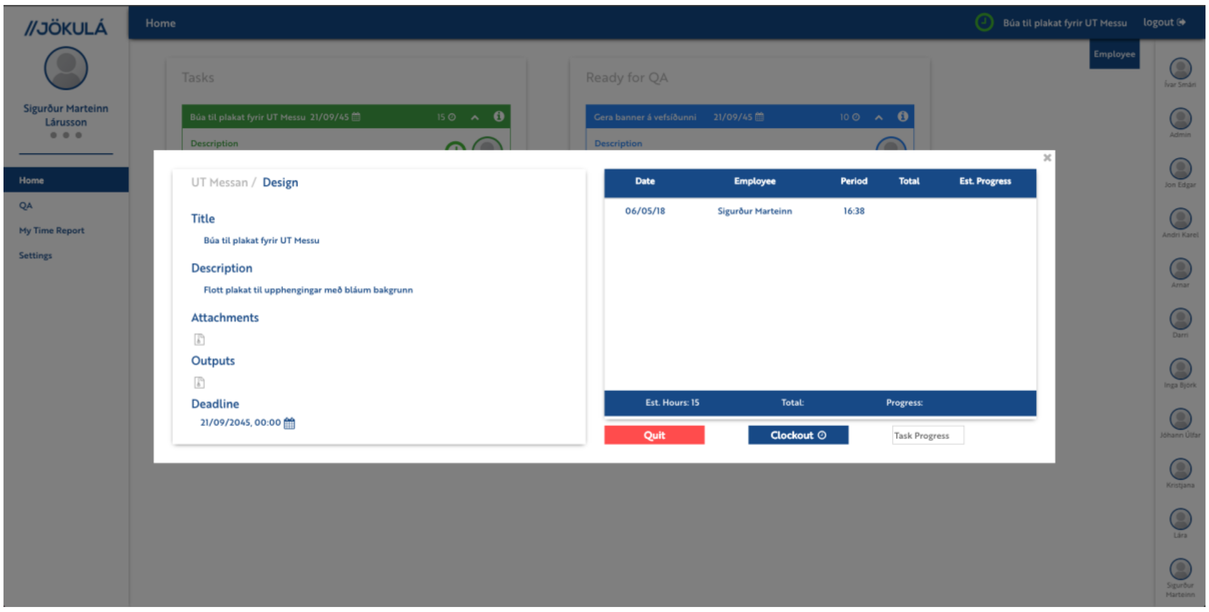
UT Messan / Design
Title
Búa til plakat fyrir UT Messu
Description
Flott plakat til upphengingar með bláum bakgrunn
Attachments
Outputs
Deadline
21/09/2045, 00:00

Date	Employee	Period	Total	Est. Progress

Est. Hours: 15 Total: 0 Hours 0 Minutes 0 Seconds Progress:

Figure 3.3: Employee task details

When the employee stops working on the task he can clock out of the task. He will be prompted to fill in the progress of the task before clocking out (see figure 3.4).



UT Messan / Design

Title

Búa til plakat fyrir UT Messu

Description

Flott plakat til upphengingar með bláum bakgrunn

Attachments

Outputs

Deadline

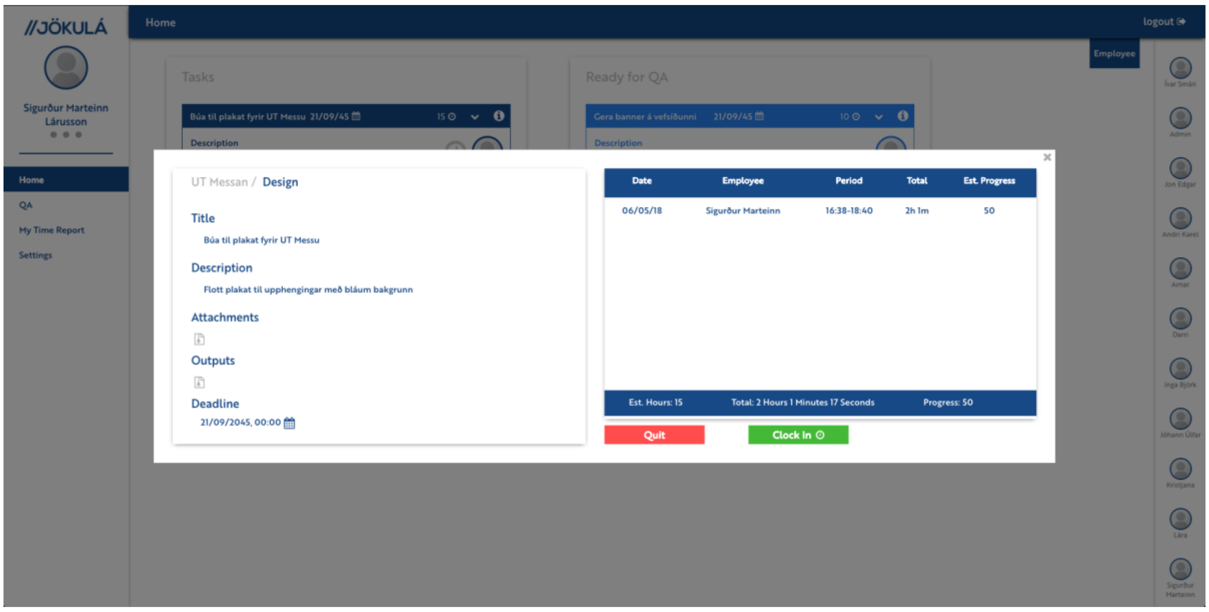
21/09/2045, 00:00

Date	Employee	Period	Total	Est. Progress
06/05/18	Sigurður Marteinn	16:38		

Est. Hours: 15 Total: Progress:

Figure 3.4: Employee task details clocked in

After clocking out the employee can see the total time that he worked each session and the total time that has been worked on the task (see figure 3.5).



UT Messan / Design

Title

Búa til plakat fyrir UT Messu

Description

Flott plakat til upphengingar með bláum bakgrunn

Attachments

Outputs

Deadline

21/09/2045, 00:00

Date	Employee	Period	Total	Est. Progress
06/05/18	Sigurður Marteinn	16:38-18:40	2h 1m	50

Est. Hours: 15 Total: 2 Hours 1 Minutes 17 Seconds Progress: 50

Figure 3.5: Employee task details clocked out

Figure 3.6 shows how the employee's homepage looks when he is clocked into a task. The task he is clocked on to turns green and he has a green clock and title of the task in his navigation bar indicating that he is clocked in. Clicking on the clock in the navigation bar is also a shortcut to open the tasks details so he can easily clock out of the task. If the employee is clocked in on a task there is no clock icon on other tasks, this restricts him from being clocked in on multiple tasks.

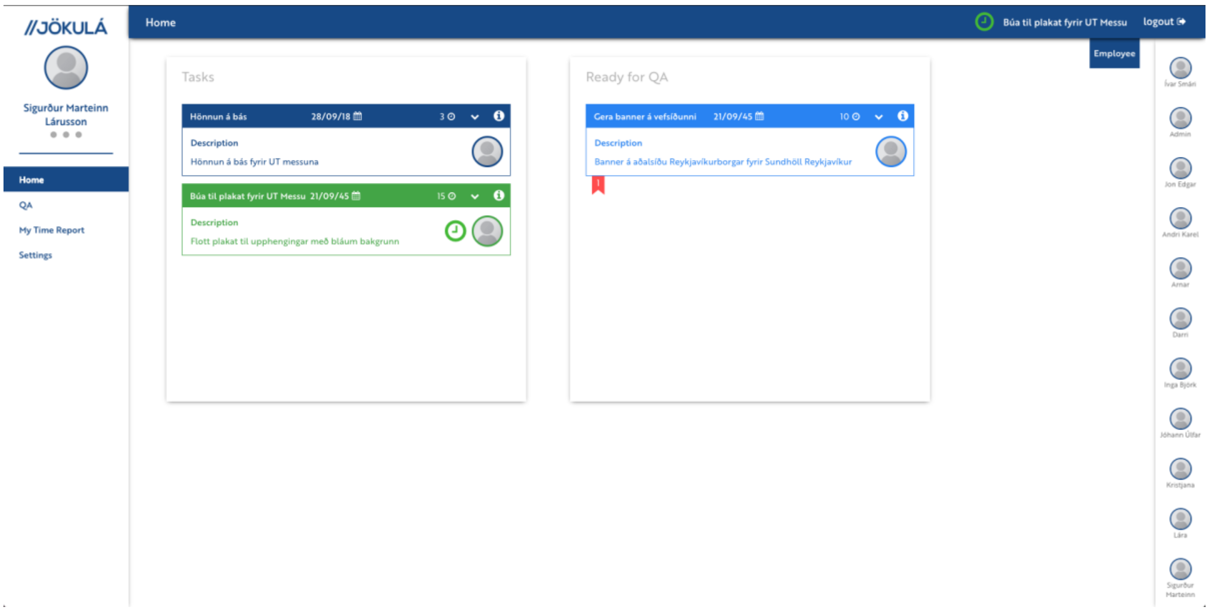

//JÖKULÁ
Sigurður Marteinn Lárusson
Home
QA
My Time Report
Settings
Home
Búa til plakat fyrir UT Messu
logout
Employee
Tasks
Hönnun á bás 28/09/18 3
Description
Hönnun á bás fyrir UT messuna
Búa til plakat fyrir UT Messu 21/09/45 15
Description
Flott plakat til upphengingar með bláum bakgrunn
Ready for QA
Gera banner á vefsíðunni 21/09/45 10
Description
Banner á aðalsíðu Reykjavíkurborgar fyrir Sundhöll Reykjavíkur
Ívar Smári
Admin
Jon Edgar
Andri Karel
Arnar
Darri
Inga Björk
Jóhann Úlfar
Kristjana
Lára
Sigurður Marteinn

Figure 3.6: Employee homepage clocked in

If the user is clocked into a task he is unable to clock into another before first clocking out of the previous task (see figure 3.7).

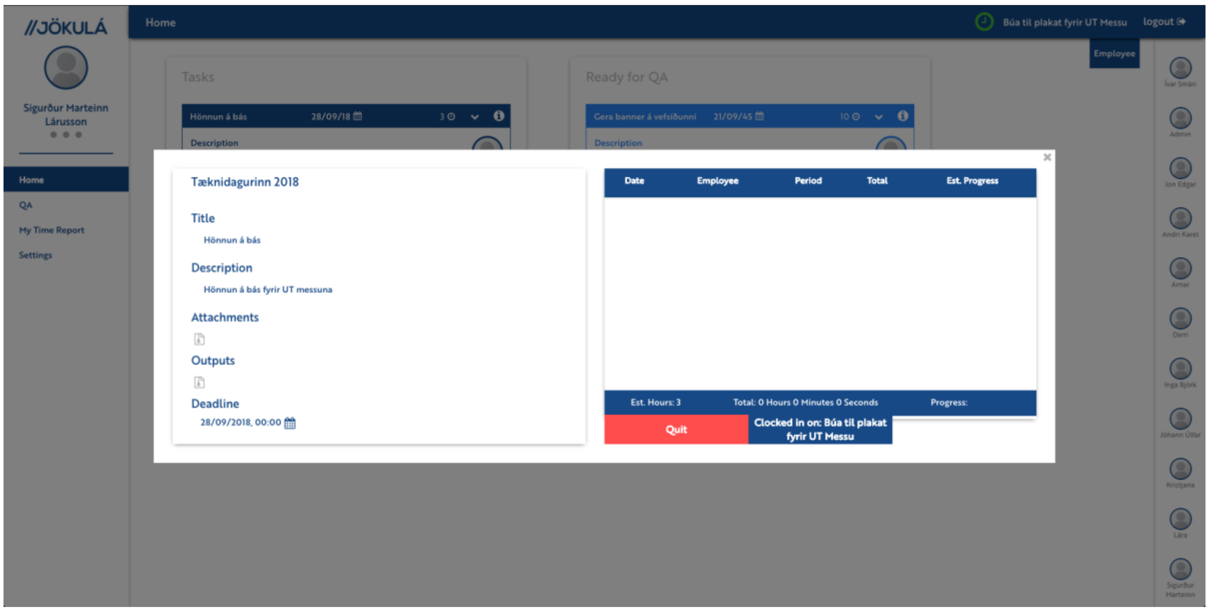
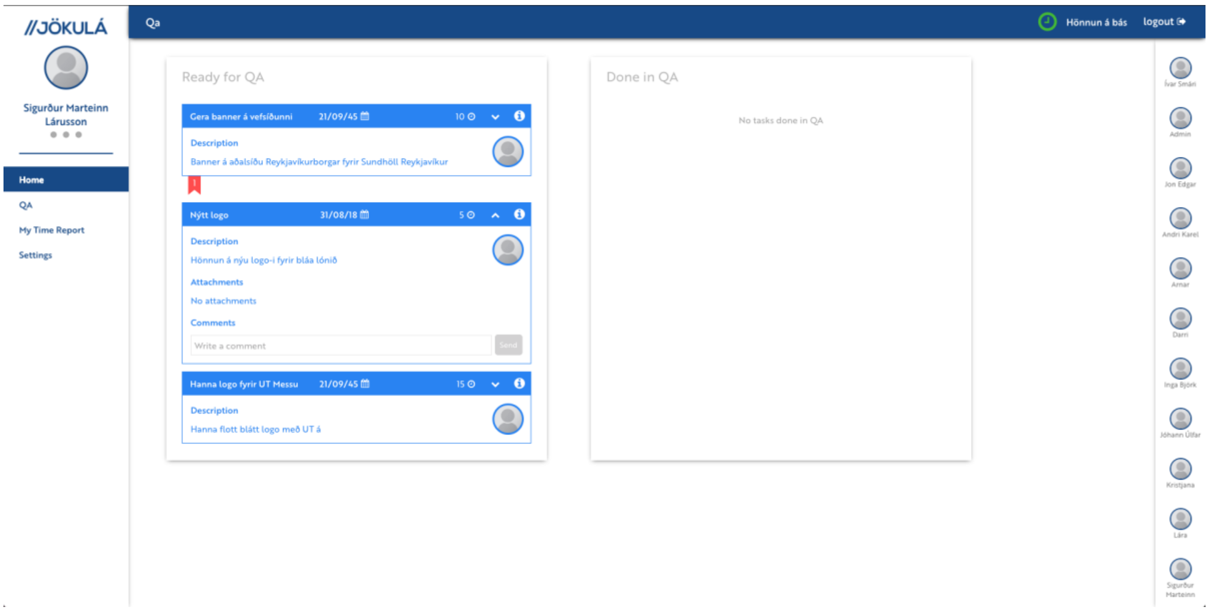


Figure 3.7: Employee task details clocked in on other

All employees are able to see the QA view, although only the product manager can accept or decline QA (see figure 3.8). All employees can comment on the tasks if they have feedback that they would like to provide.



Qa

Sigurður Marteinn Lárusson

Ready for QA

Gera banner á vefsíðunni 21/09/45 10

Description

Banner á aðalsíðu Reykjavíkurborgar fyrir Sundhöll Reykjavíkur

Nýtt logo 31/08/18 5

Description

Hönnun á nýu logo-i fyrir bláa lónið

Attachments

No attachments

Comments

Write a comment

Hanna logo fyrir UT Messu 21/09/45 15

Description

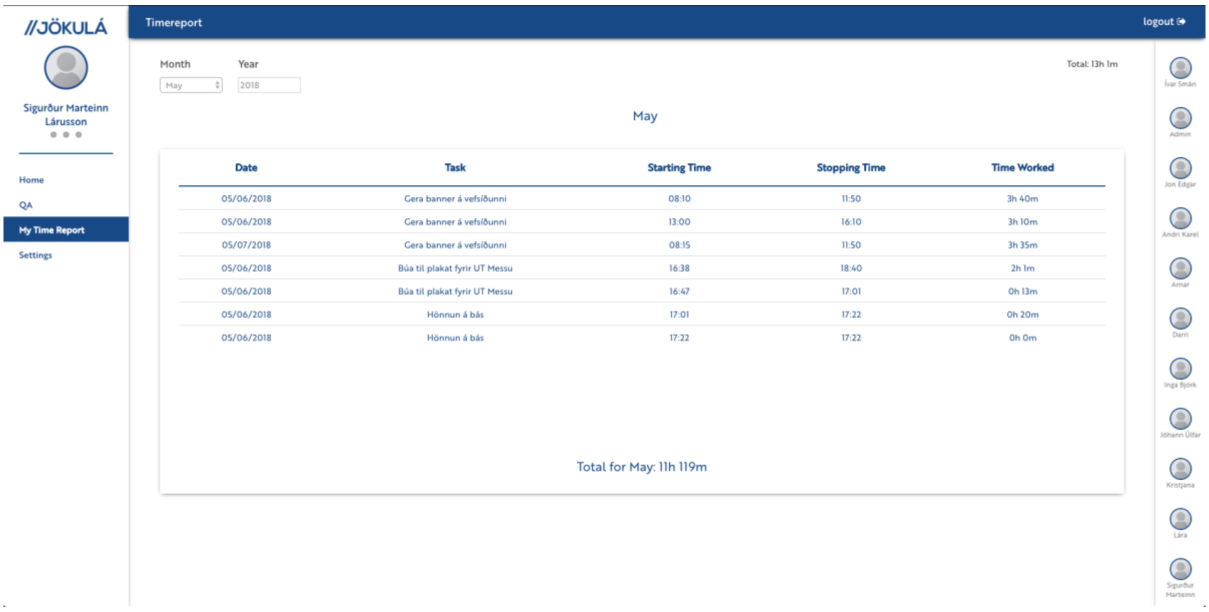
Hanna flott blátt logo með UT á

Done in QA

No tasks done in QA

Figure 3.8: Employee ready for QA

All employees can access their time reports to see the hours they have logged(see figure 3.9).



Timereport

Sigurður Marteinn Lárusson

Month: May   Year: 2018   Total: 13h 1m

May

Date	Task	Starting Time	Stopping Time	Time Worked
05/06/2018	Gera banner á vefsíðunni	08:10	11:50	3h 40m
05/06/2018	Gera banner á vefsíðunni	13:00	16:10	3h 10m
05/07/2018	Gera banner á vefsíðunni	08:15	11:50	3h 35m
05/06/2018	Búa til plakat fyrir UT Messu	16:38	18:40	2h 1m
05/06/2018	Búa til plakat fyrir UT Messu	16:47	17:01	0h 13m
05/06/2018	Hönnun á bás	17:01	17:22	0h 20m
05/06/2018	Hönnun á bás	17:22	17:22	0h 0m

Total for May: 11h 119m

Figure 3.9: Employees time report

If the employee wishes to change his profile picture or his password he can navigate to the settings page. To update the password he needs to type a new password twice to avoid an unexpected typo (see figure 3.10).

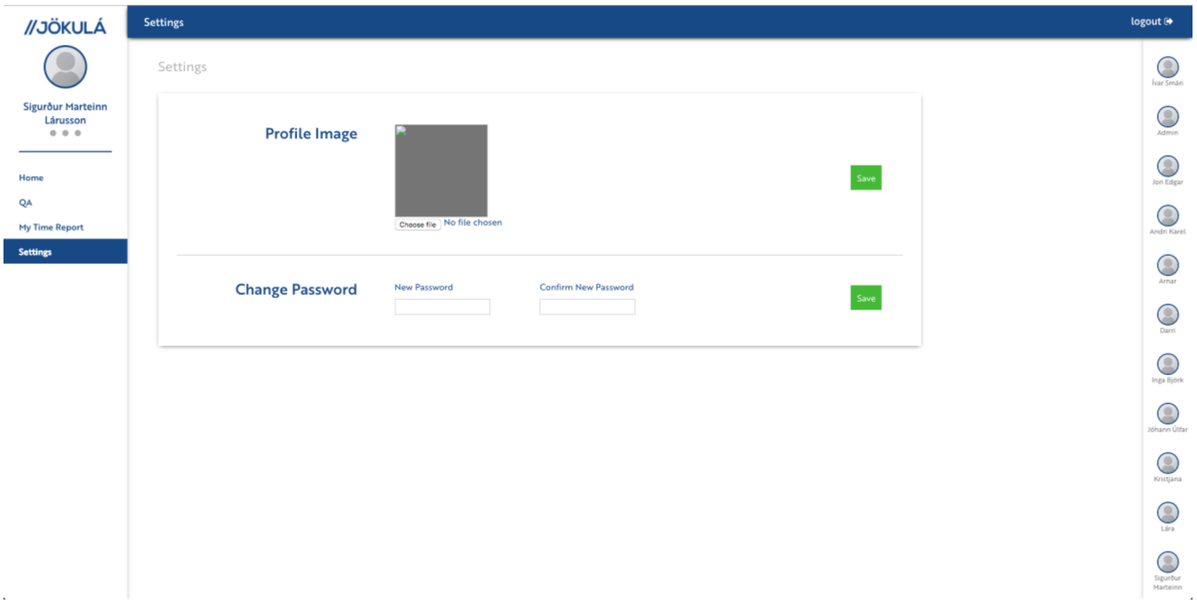


Figure 3.10: Employee settings page

If the employee wishes to change their profile picture they can click on “choose file” and pick a picture from their local machine (see figure 3.11).

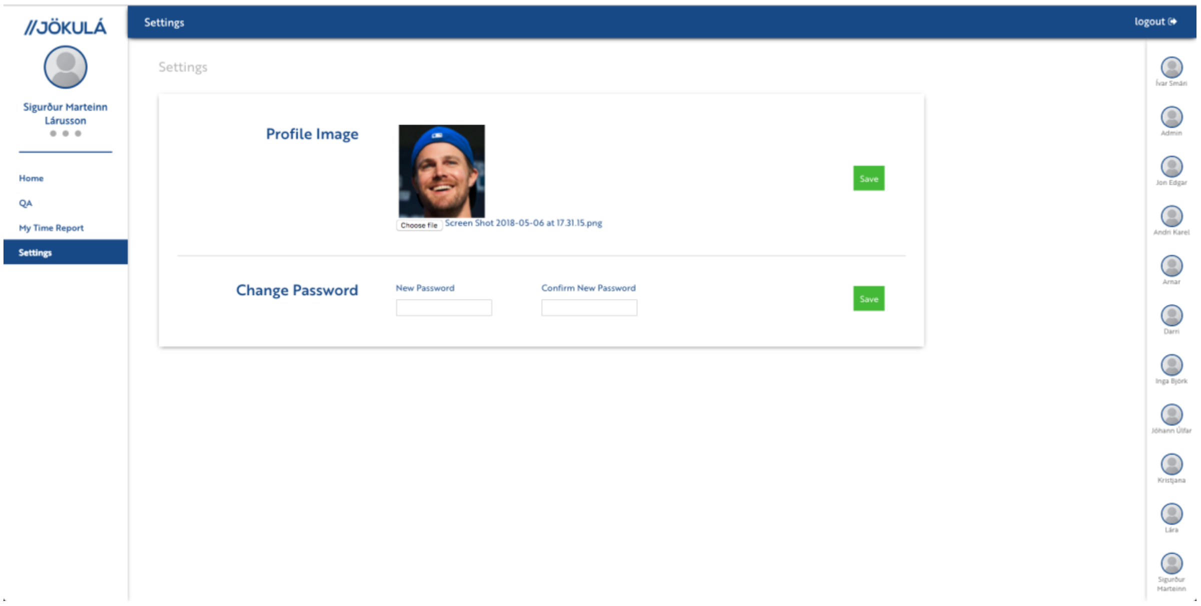
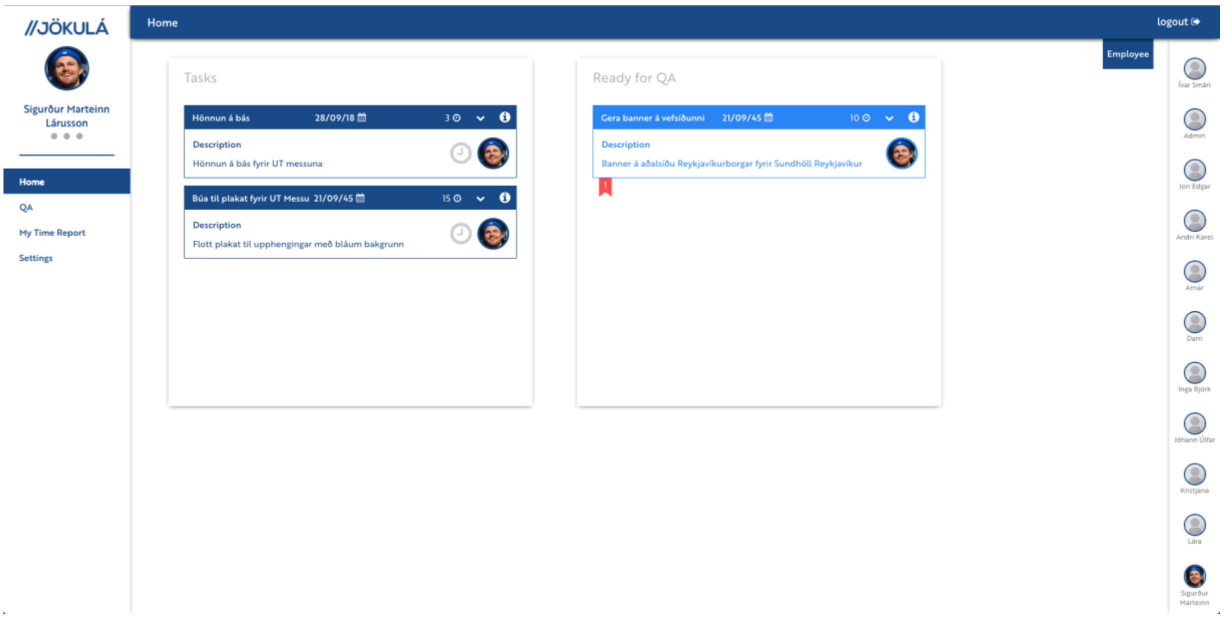


Figure 3.11: Employee new profile picture

After clicking on save, the picture is saved as the current users profile picture (see figure 3.12).



Tasks

Hönnun á bás 28/09/18 3
Description
Hönnun á bás fyrir UT messuna

Búa til plakat fyrir UT Messu 21/09/45 15
Description
Flott plakat til upphengingar með bláum bakgrunn

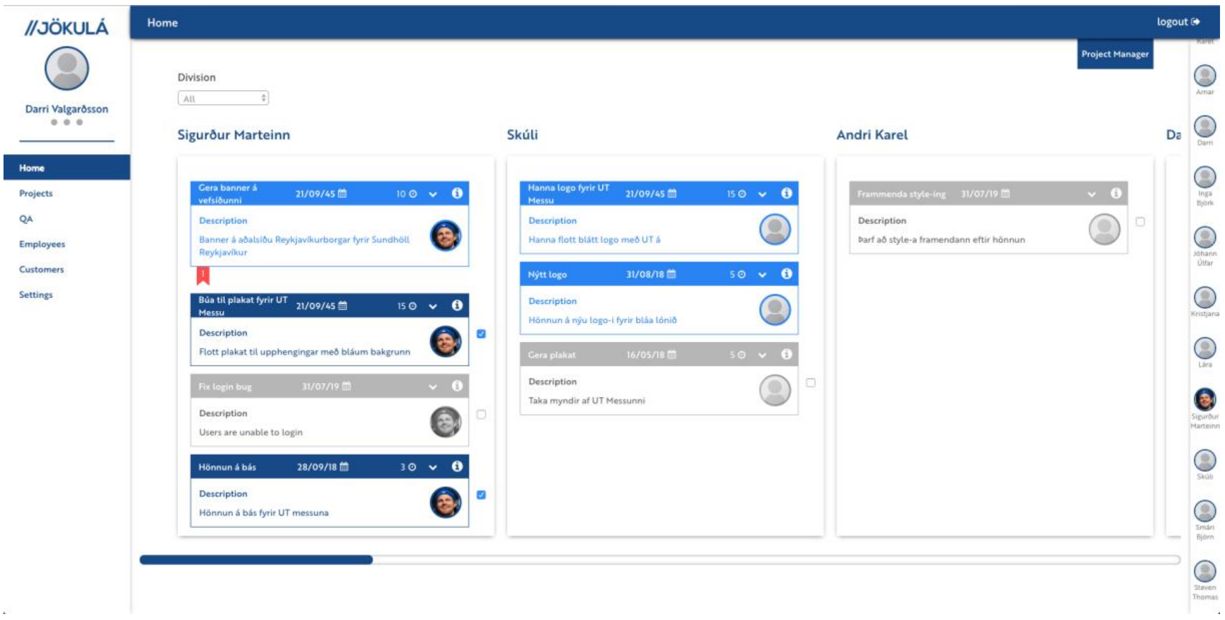
Ready for QA

Gera banner á vefsíðunni 21/09/45 10
Description
Banner á aðalsíðu Reykjavíkurborgar fyrir Sundhöll Reykjavíkur
1

Figure 3.12: Employee dashboard with new profile picture

3.2 Project manager

The project manager’s homepage gives him an overview of all assigned tasks with a checkbox to indicate if the employees themselves can see the tasks or not. Grey tasks are assigned but not visible, dark blue tasks are assigned and visible and light blue tasks are assigned and ready for QA. The project manager cannot toggle the visibility anymore if the task has been sent to QA (see figure 3.13).



Division
All

Sigurður Marteinn

Gera banner á vefsíðunni 21/09/45 10
Description
Banner á aðalsíðu Reykjavíkurborgar fyrir Sundhöll Reykjavíkur
1

Búa til plakat fyrir UT Messu 21/09/45 15
Description
Flott plakat til upphengingar með bláum bakgrunn

Fix login bug 31/07/19
Description
Users are unable to login

Hönnun á bás 28/09/18 3
Description
Hönnun á bás fyrir UT messuna

Skúli

Hanna logo fyrir UT Messu 21/09/45 15
Description
Hanna flott blátt logo með UT á

Nýtt logo 31/08/18 5
Description
Hönnun á nýu logo-i fyrir bláa lónið

Gera plakat 16/05/18 5
Description
Taka myndir af UT Messunni

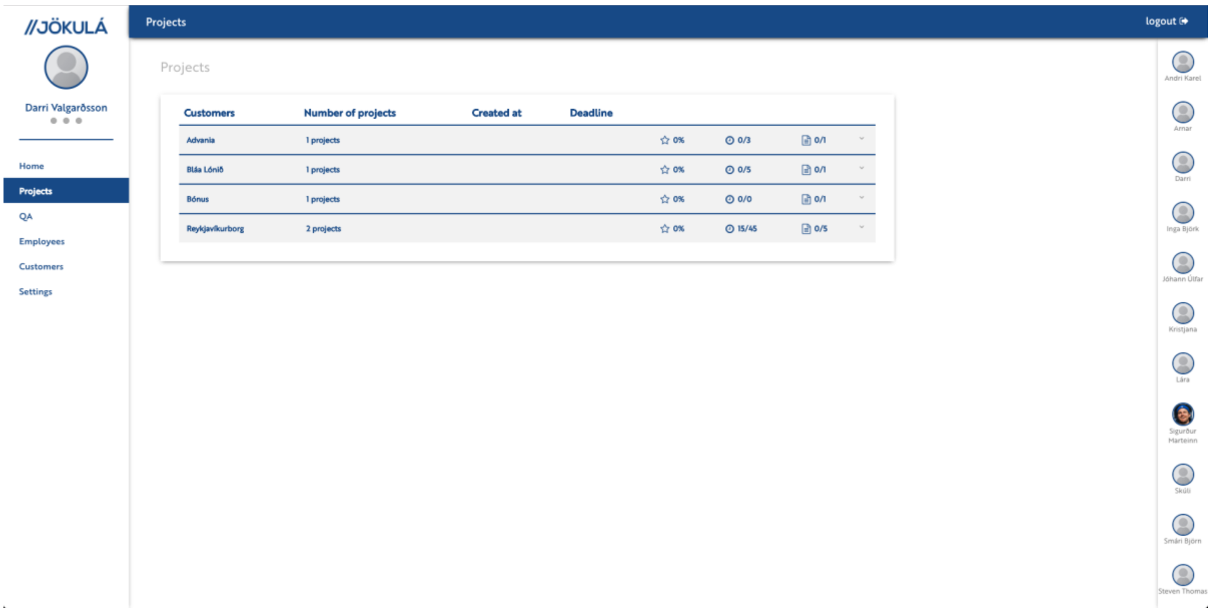
Andri Karel

Frammenda style-ing 31/07/19
Description
Þarf að style-a framendann eftir hönnun

Da

Figure 3.13: Project manager dashboard

Figure 3.14 shows the first view the project manager gets when looking at projects. From there he can either click on a row in the table to expand that row and see all projects belonging to that customer or he can click on the number of projects which will take him to another view with all projects belonging to that customer.

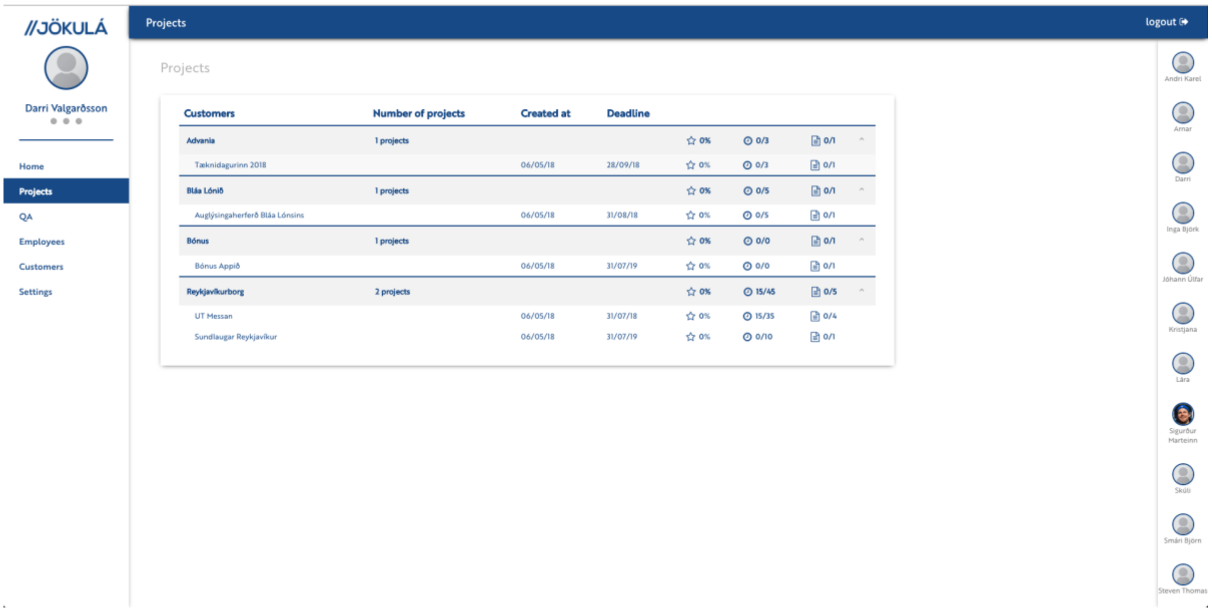


Projects

Customers	Number of projects	Created at	Deadline			
Advania	1 projects			☆ 0%	◷ 0/3	🗎 0/1
Bláa Lónið	1 projects			☆ 0%	◷ 0/5	🗎 0/1
Bónus	1 projects			☆ 0%	◷ 0/0	🗎 0/1
Reykjavíkurborg	2 projects			☆ 0%	◷ 15/45	🗎 0/5

Figure 3.14: Project manager project view

Figure 3.15 shows the project managers projects view after expanding the customer table.



Projects

Customers	Number of projects	Created at	Deadline			
Advania	1 projects			☆ 0%	◷ 0/3	🗎 0/1
Tæknidagurinn 2018		06/05/18	28/09/18	☆ 0%	◷ 0/3	🗎 0/1
Bláa Lónið	1 projects			☆ 0%	◷ 0/5	🗎 0/1
Auglýsingaherferð Bláa Lónsins		06/05/18	31/08/18	☆ 0%	◷ 0/5	🗎 0/1
Bónus	1 projects			☆ 0%	◷ 0/0	🗎 0/1
Bónus Appið		06/05/18	31/07/19	☆ 0%	◷ 0/0	🗎 0/1
Reykjavíkurborg	2 projects			☆ 0%	◷ 15/45	🗎 0/5
UT Messan		06/05/18	31/07/18	☆ 0%	◷ 15/35	🗎 0/4
Sundlaugar Reykjavíkur		06/05/18	31/07/19	☆ 0%	◷ 0/10	🗎 0/1

Figure 3.15: Project manager project view expanded

The project manager will get a view of all projects belonging to a specific customer if he clicks on the number of projects in the projects view (see figure 3.16).

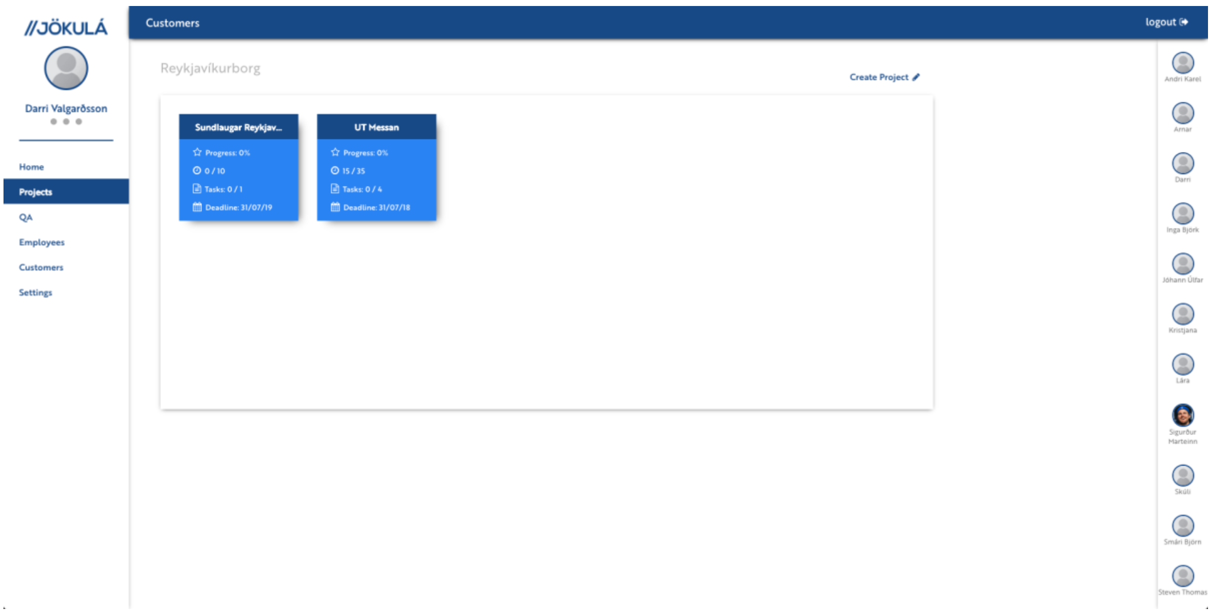


Figure 3.16: Project manager projects by customer view

The specific project view (see figure 3.16) will appear once the project manager has selected a project. The views will continue to look the same as this when going deeper into the folder structure. If for example “design” were chosen here the same view would appear with Design’s sub-projects and tasks.

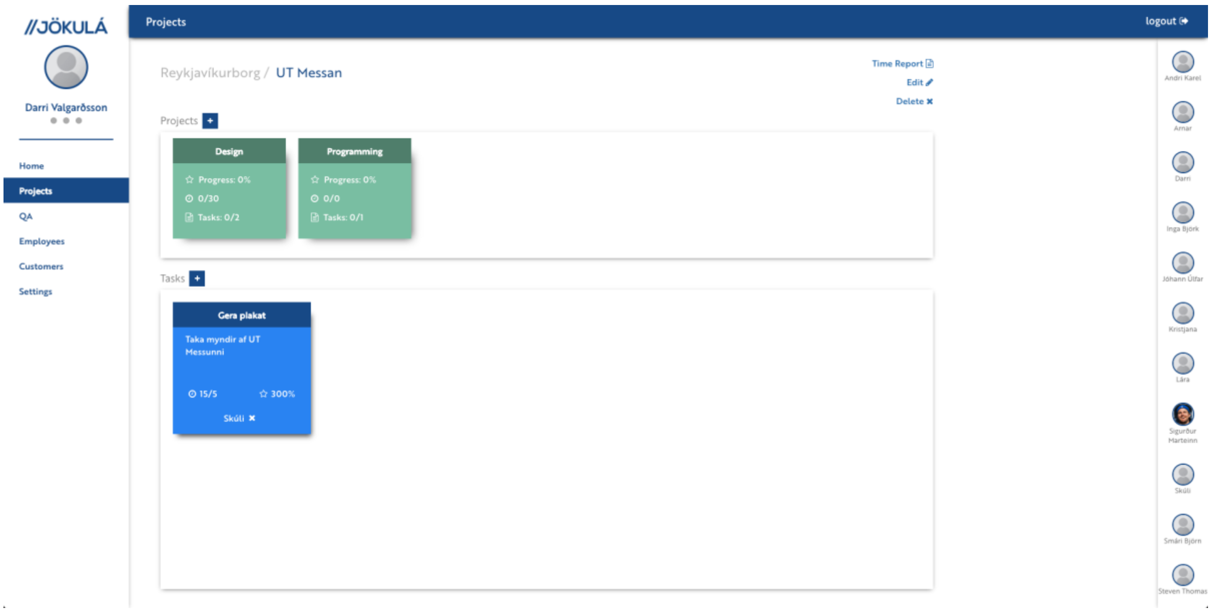
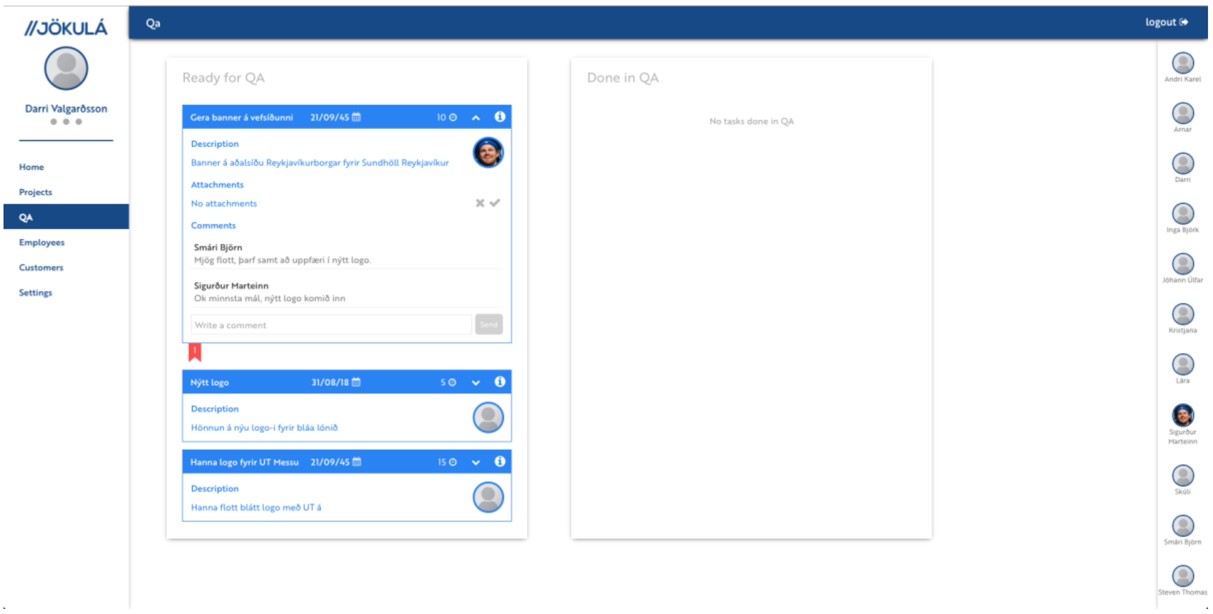


Figure 3.16: Project manager specific project view

The project manager has a QA view as all other employees, except with the distinction of having accept/reject buttons to either accept a task as “Done in QA” or send it back, preferably with a comment to the employee telling them what needs fixing, aslo incrementing the QA counter (see figure 3.17).



Ready for QA

Gera banner á vefsíðunni 21/09/45 10

Description
Banner á aðalsíðu Reykjavíkurborgar fyrir Sundhöll Reykjavíkur

Attachments
No attachments

Comments

Smári Björn
Mjög flott, þarf samt að uppfæri í nýtt logo.

Sigurður Marteinn
Ok minnsta mál, nýtt logo komið inn

Write a comment Send

1

Nýtt logo 31/08/18 5

Description
Hönnun á nýu logo-i fyrir bláa lónið

Hanna logo fyrir UT Messu 21/09/45 15

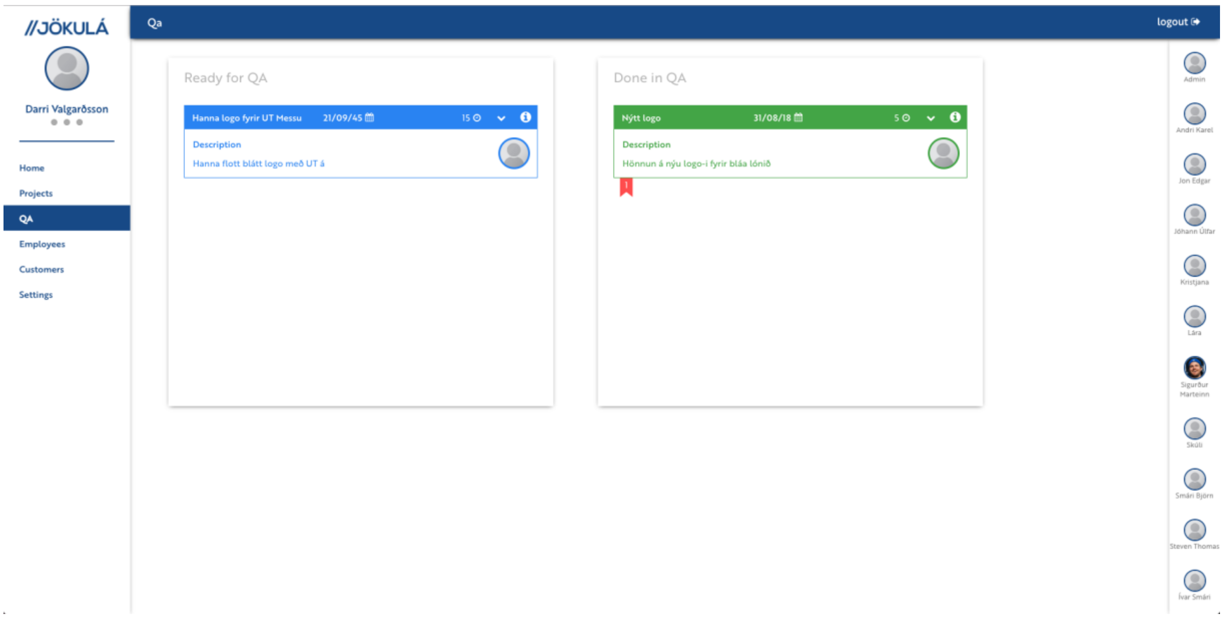
Description
Hanna flott blátt logo með UT á

Done in QA

No tasks done in QA

Figure 3.17: Project manager ready for QA

Figure 3.18 shows a project manager's view when a task has been moved to Ready for QA.



Ready for QA

Hanna logo fyrir UT Messu 21/09/45 15

Description
Hanna flott blátt logo með UT á

Done in QA

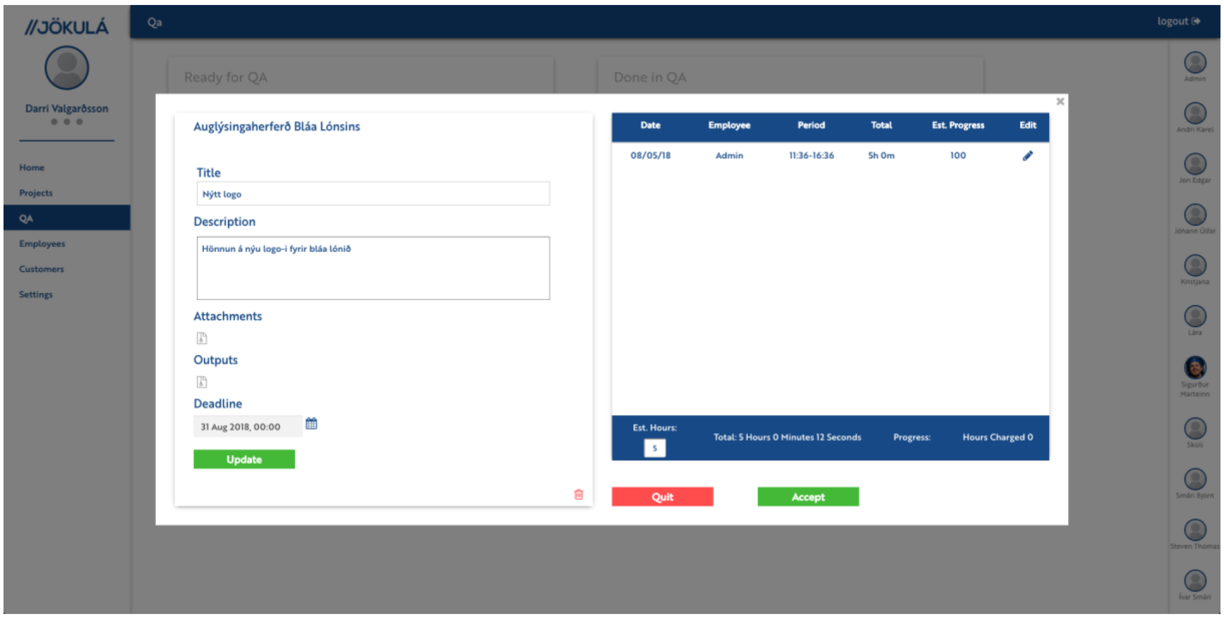
Nýtt logo 31/08/18 5

Description
Hönnun á nýu logo-i fyrir bláa lónið

1

Figure 3.18: Project manager done in QA view

Figure 3.19 shows a project managers task details view after the task has been sent to Done in QA.



Auglýsingaherferð Bláa Lónsins

Title: Nýtt logo

Description: Hönnun á nýu logo-i fyrir bláa lónið

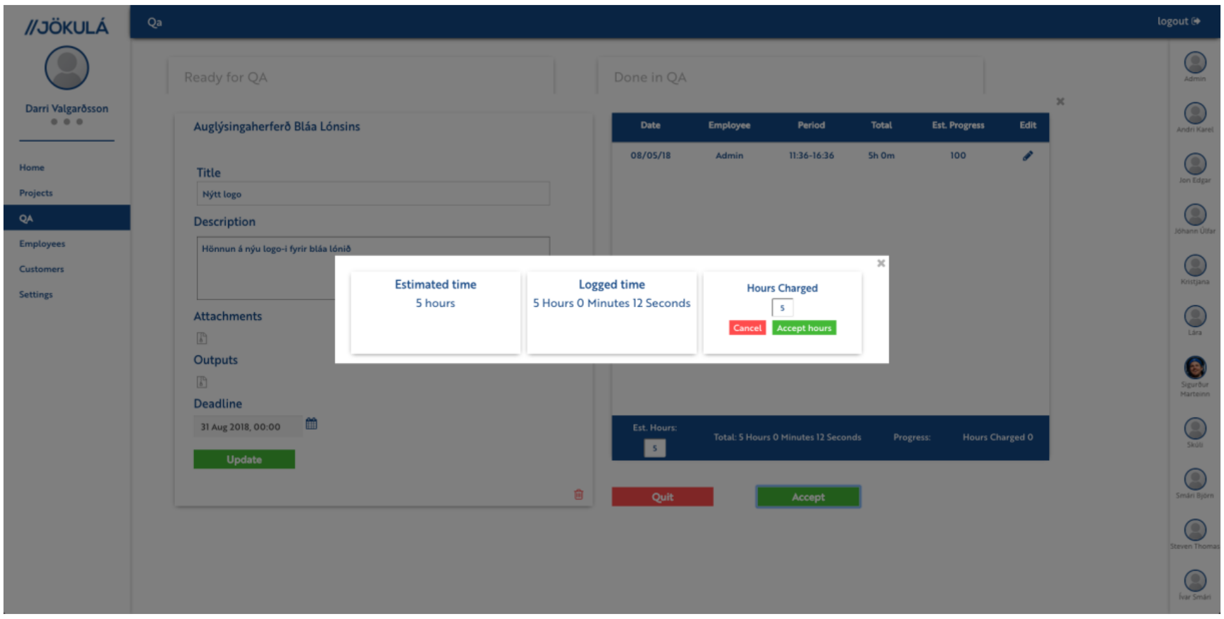
Deadline: 31 Aug 2018, 00:00

Date	Employee	Period	Total	Est. Progress	Edit
08/05/18	Admin	11:36-16:36	5h 0m	100	

Est. Hours: 5 Total: 5 Hours 0 Minutes 12 Seconds Progress: Hours Charged 0

Figure 3.19: Project manager done in QA task detail

When the project manager clicks on accept in a task in done in QA, a modal appears (see figure 3.20) where he has to evaluate how many hours should be invoiced on the task. He can choose however many hours he judges to be correct. The default is the estimated hours on the task as that will be in accordance to the agreed upon hours with the customer.



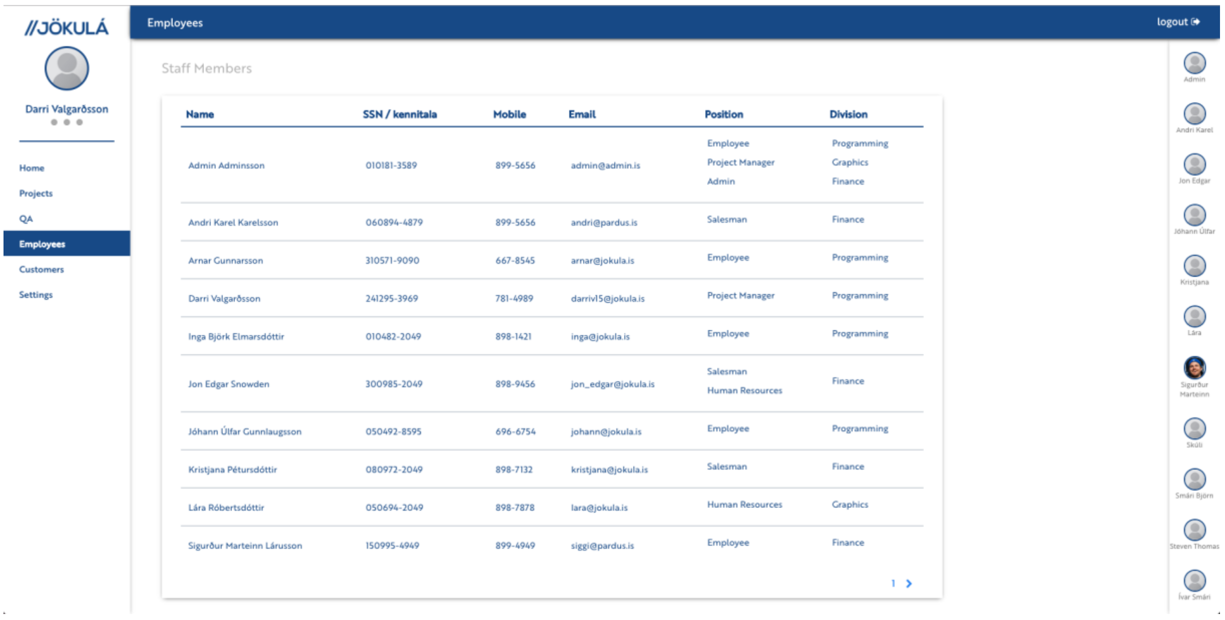
Estimated time: 5 hours

Logged time: 5 Hours 0 Minutes 12 Seconds

Hours Charged: 5

Figure 3.20: Project manager - accept hours

The project manager has an “Employees” view (see figure 3.21). He has access to read employees info but no update/delete rights.

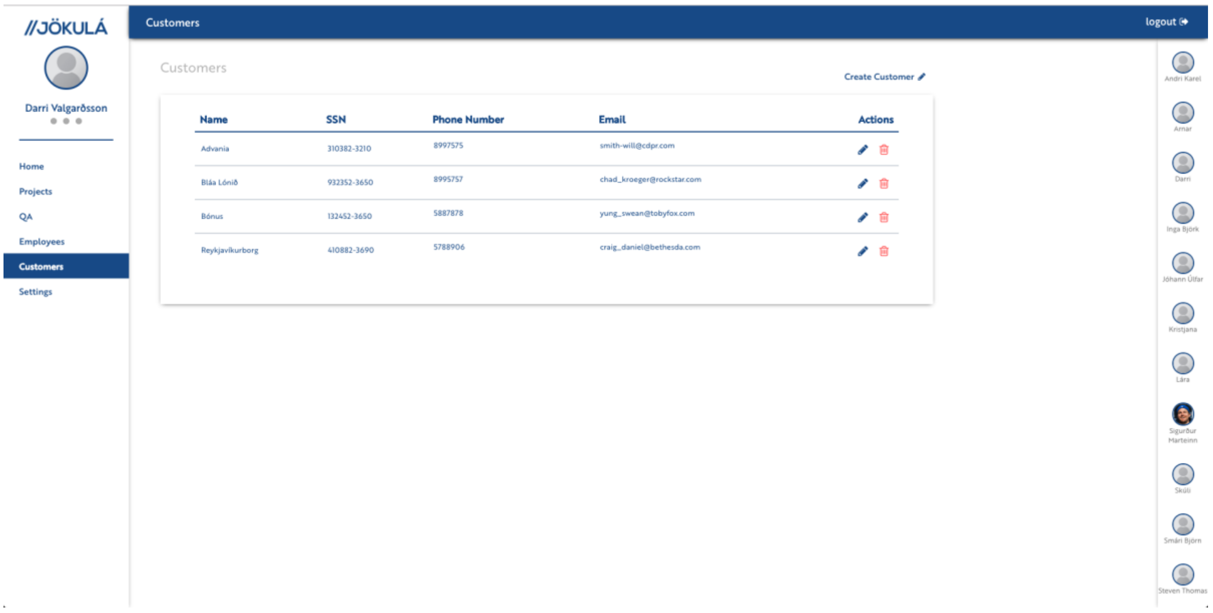


Staff Members

Name	SSN / kennitala	Mobile	Email	Position	Division
Admin Adminsson	010181-3589	899-5656	admin@admin.is	Employee Project Manager Admin	Programming Graphics Finance
Andri Karel Karelsson	060894-4879	899-5656	andri@pardus.is	Salesman	Finance
Arnar Gunnarsson	310571-9090	667-8545	arnar@jokula.is	Employee	Programming
Darri Valgarðsson	241295-3969	781-4989	darrivl5@jokula.is	Project Manager	Programming
Inga Björk Elmarsdóttir	010482-2049	898-1421	inga@jokula.is	Employee	Programming
Jon Edgar Snowden	300985-2049	898-9456	jon_edgar@jokula.is	Salesman Human Resources	Finance
Jóhann Úlfar Gunnlaugsson	050492-8595	696-6754	johann@jokula.is	Employee	Programming
Kristjana Pétursdóttir	080972-2049	898-7132	kristjana@jokula.is	Salesman	Finance
Lára Róbertsdóttir	050694-2049	898-7878	lara@jokula.is	Human Resources	Graphics
Sigurður Marteinn Lárusson	150995-4949	899-4949	siggi@pardus.is	Employee	Finance

Figure 3.21: Project manager employee view

The project manager has a customers view (see figure 3.22). This view has a general overview showing the customers name and ssn, the phone number and email of the main contact.

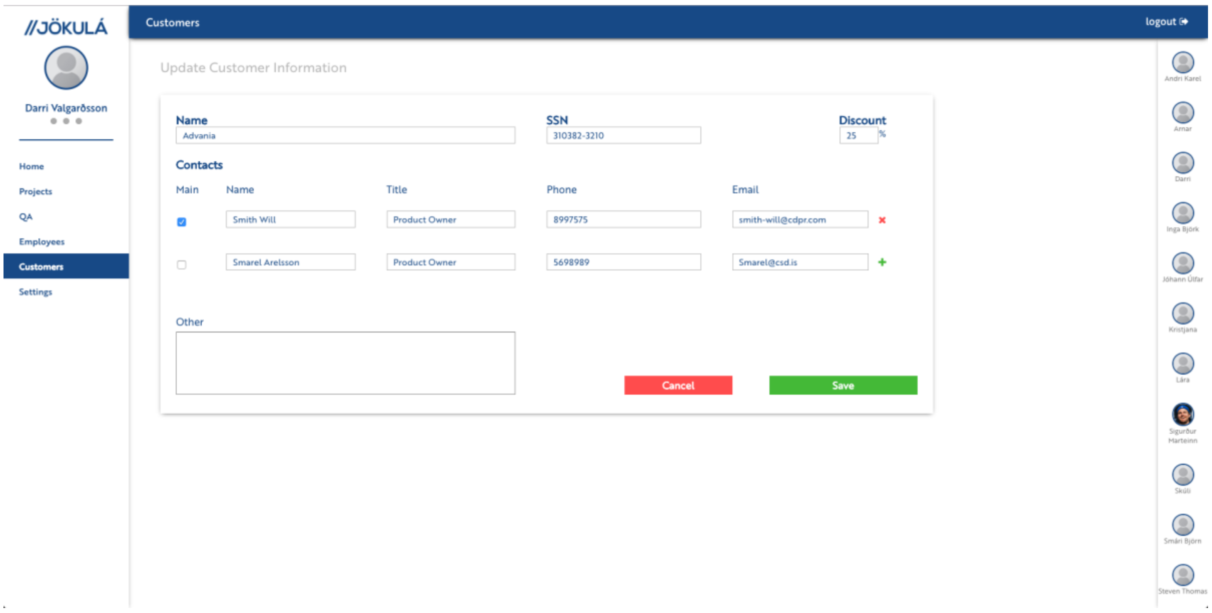


Customers

Name	SSN	Phone Number	Email	Actions
Advania	310382-3210	8997575	smith-will@cdpr.com	
Bláa Lónið	932352-3650	8995757	chad_kroeger@rockstar.com	
Bónus	132452-3650	5887878	yung_swean@tobyfox.com	
Reykjavíkurborg	410882-3690	5788906	craig_daniel@bethesda.com	

Figure 3.22: Project manager customer view

When editing, the project manager can set the basic info for the customer as well as set contacts for the customer (see figure 3.23). In cases where the customer has multiple contacts he can pick a contact to be the main contact and who's info will be displayed by default on the customers frontpage.



Update Customer Information

Name: Advania
SSN: 310382-3210
Discount: 25 %

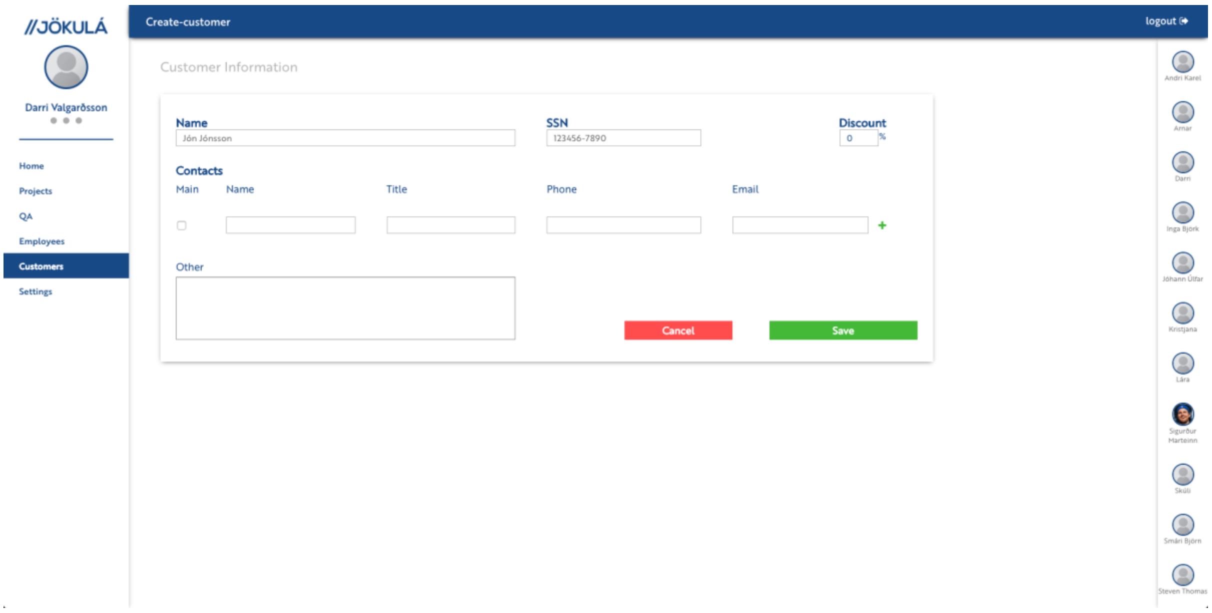
Contacts

Main	Name	Title	Phone	Email
✓	Smith Will	Product Owner	8997575	smith-will@cdpr.com
☐	Smarel Arelsson	Product Owner	5698989	Smarel@csd.is

Other

Figure 3.23: Project manager edit customer

Figure 3.24 shows the Create Customer form for project manager.



Customer Information

Name: Jón Jónsson
SSN: 123456-7890
Discount: 0 %

Contacts

Main	Name	Title	Phone	Email
☐				

Other

Figure 3.24: Project manager new customer

The project manager also has a settings page (see figure 3.25) where he can upload a picture to use as his profile picture or change his password.

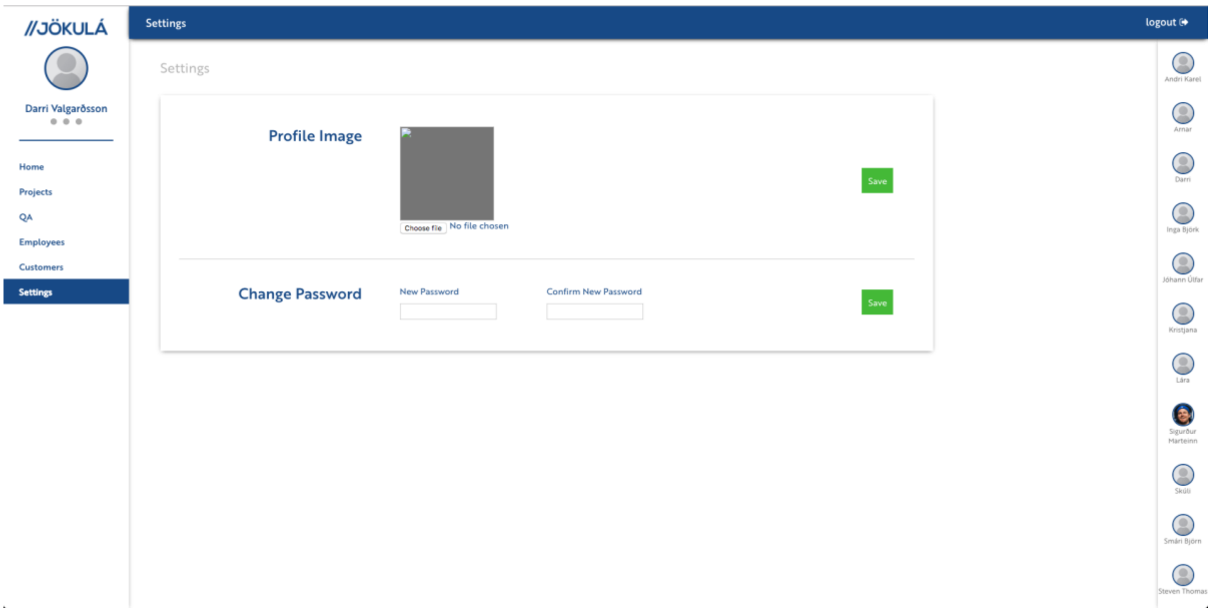
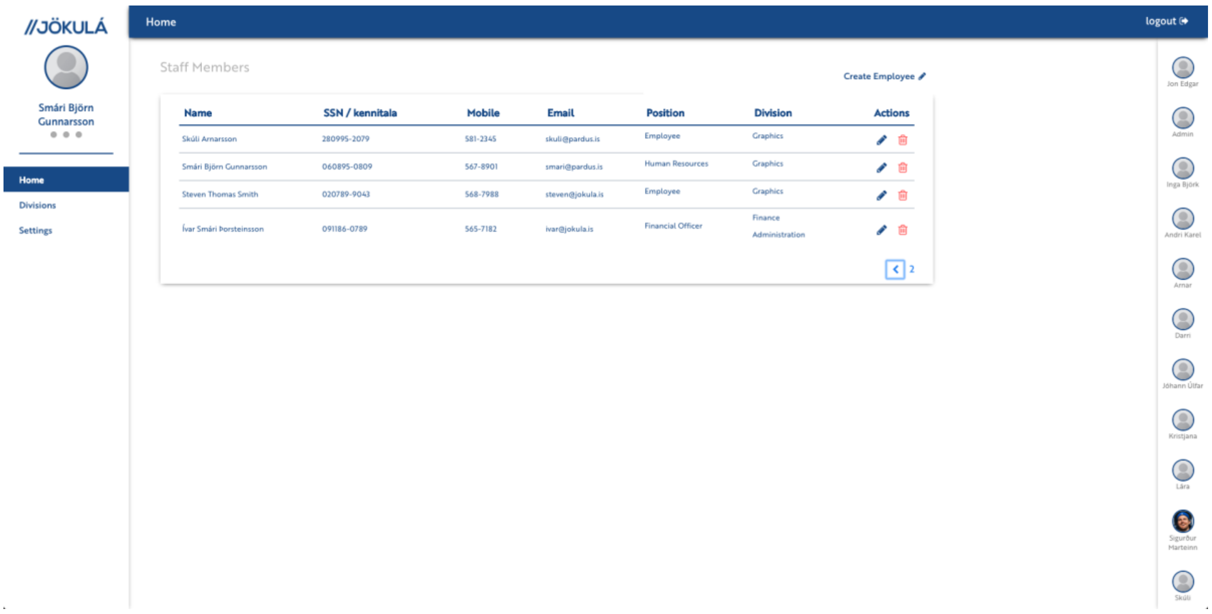


Figure 3.25: Project manager settings

3.3 Human Resources

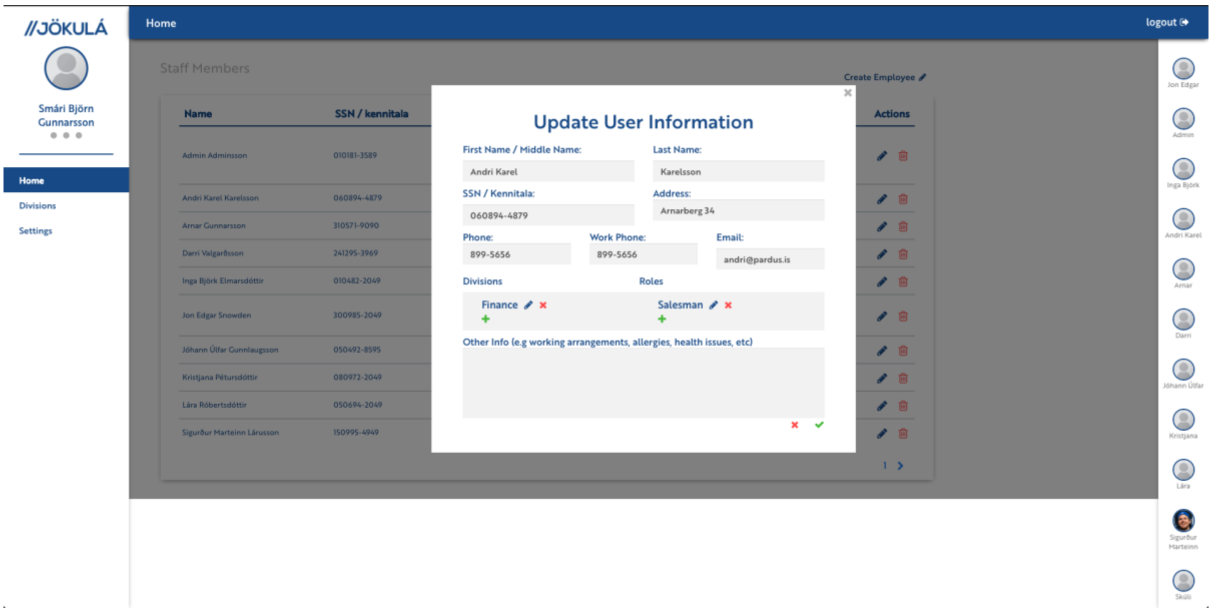
The homepage for the Human Resources Officer is the Staff members page (see figure 3.26) where the HR employee has CRUD access to all staff.



Name	SSN / kennitala	Mobile	Email	Position	Division	Actions
Skúli Arnarsson	280995-2079	581-2345	skuli@pardus.is	Employee	Graphics	
Smári Björn Gunnarsson	060895-0809	567-8901	smari@pardus.is	Human Resources	Graphics	
Steven Thomas Smith	020789-9043	568-7988	steven@jokula.is	Employee	Graphics	
Ívar Smári Þorsteinsson	091186-0789	565-7182	ivar@jokula.is	Financial Officer	Finance Administration	

Figure 3.26: Human resources homepage

The update employee form for the HR employee (see figure 3.27).



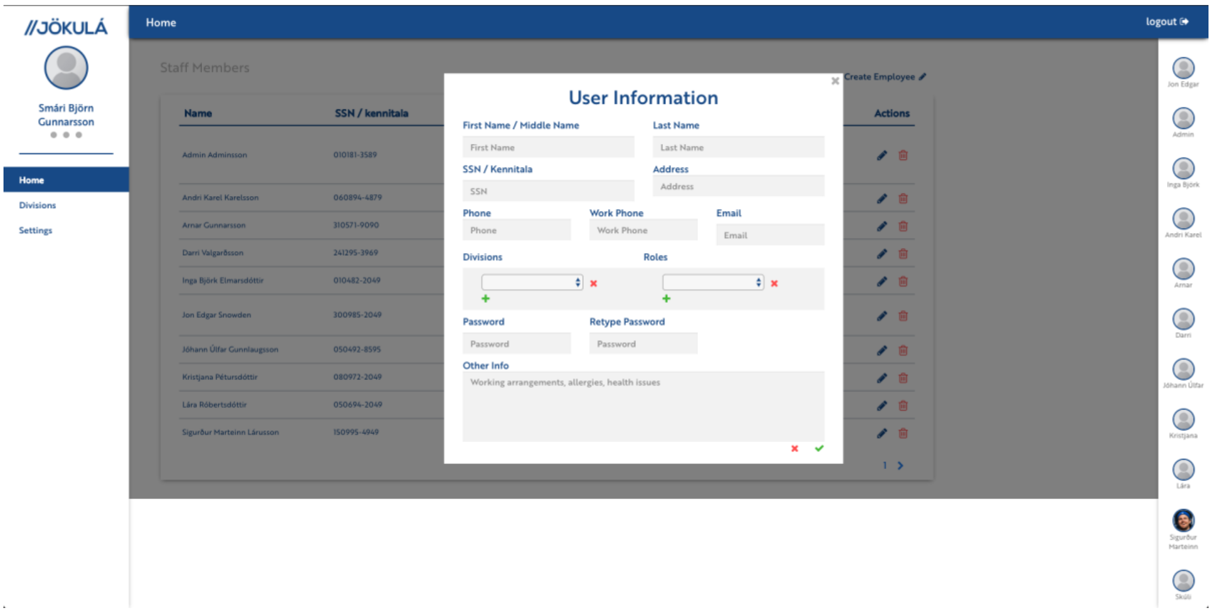
Name	SSN / kennitala	Actions
Admin Adminsson	010181-3589	
Andri Karel Karelsson	060894-4879	
Arnar Gunnarsson	310571-9090	
Darri Valgarðsson	241295-3969	
Inga Björk Elmarsdóttir	010482-2049	
Jon Edgar Snowden	300985-2049	
Jóhann Úlfar Gunnlaugsson	050492-8595	
Kristjana Pétursdóttir	080972-2049	
Lára Róbertsdóttir	050694-2049	
Sigurður Marteinn Lárusson	150995-4949	

Update User Information

First Name / Middle Name: Andri Karel
Last Name: Karelsson
SSN / Kennitala: 060894-4879
Address: Arnarberg 34
Phone: 899-5656
Work Phone: 899-5656
Email: andri@pardus.is
Divisions: Finance
Roles: Salesman
Other Info (e.g working arrangements, allergies, health issues, etc)

Figure 3.27: Human resources edit employee

The create employee form for the HR employee (see figure 3.28).



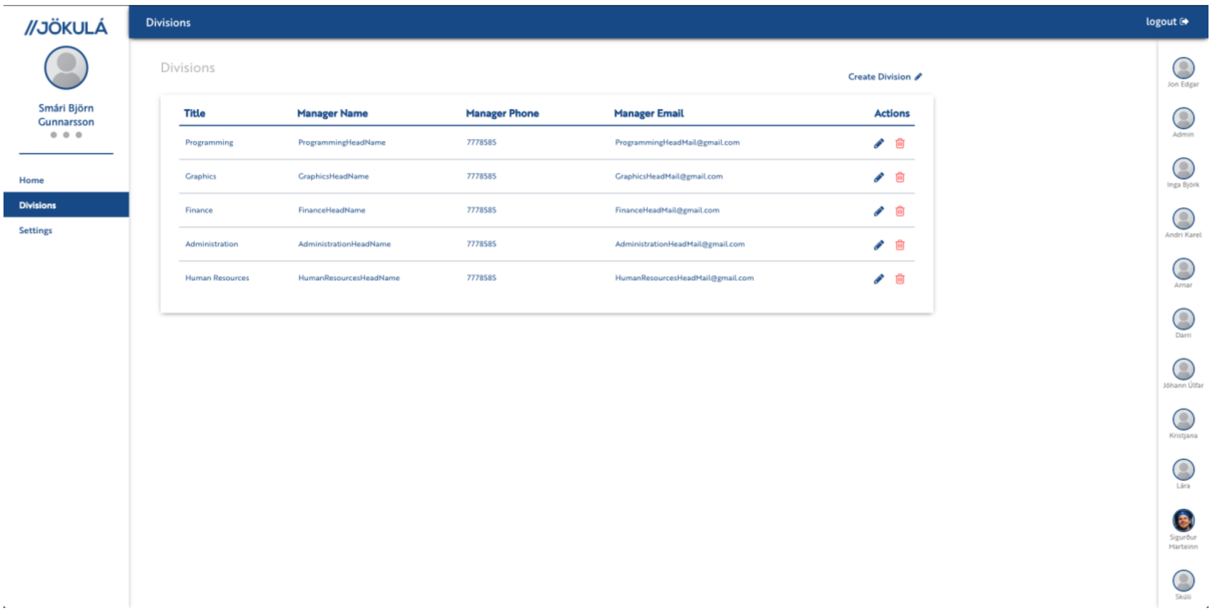
Name	SSN / kennitala	Actions
Admin Adminsson	010181-3589	
Andri Karel Karelsson	060894-4879	
Arnar Gunnarsson	310571-9090	
Darri Valgarðsson	241295-3969	
Inga Björk Elmarsdóttir	010482-2049	
Jon Edgar Snowden	300985-2049	
Jóhann Úlfar Gunnlaugsson	050492-8595	
Kristjana Pétursdóttir	080972-2049	
Lára Róbertsdóttir	050694-2049	
Sigurður Marteinn Lárusson	150995-4949	

User Information

First Name / Middle Name: First Name
Last Name: Last Name
SSN / Kennitala: SSN
Address: Address
Phone: Phone
Work Phone: Work Phone
Email: Email
Divisions
Roles
Password: Password
Retype Password: Password
Other Info: Working arrangements, allergies, health issues

Figure 3.28: Human resources create employee

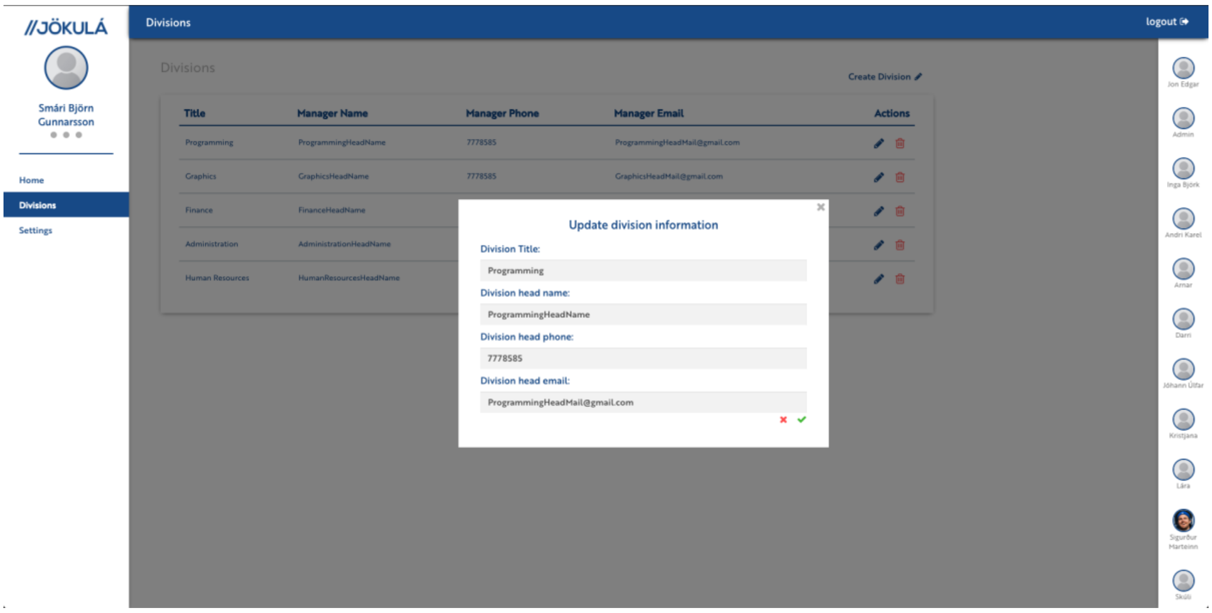
The human resources employee has CRUD access to Divisions (see figure 3.29).



Title	Manager Name	Manager Phone	Manager Email	Actions
Programming	ProgrammingHeadName	7778585	ProgrammingHeadMail@gmail.com	
Graphics	GraphicsHeadName	7778585	GraphicsHeadMail@gmail.com	
Finance	FinanceHeadName	7778585	FinanceHeadMail@gmail.com	
Administration	AdministrationHeadName	7778585	AdministrationHeadMail@gmail.com	
Human Resources	HumanResourcesHeadName	7778585	HumanResourcesHeadMail@gmail.com	

Figure 3.29: Human resources divisions view

The update form for divisions for the HR employee (see figure 3.30).



Update division information

Division Title: Programming

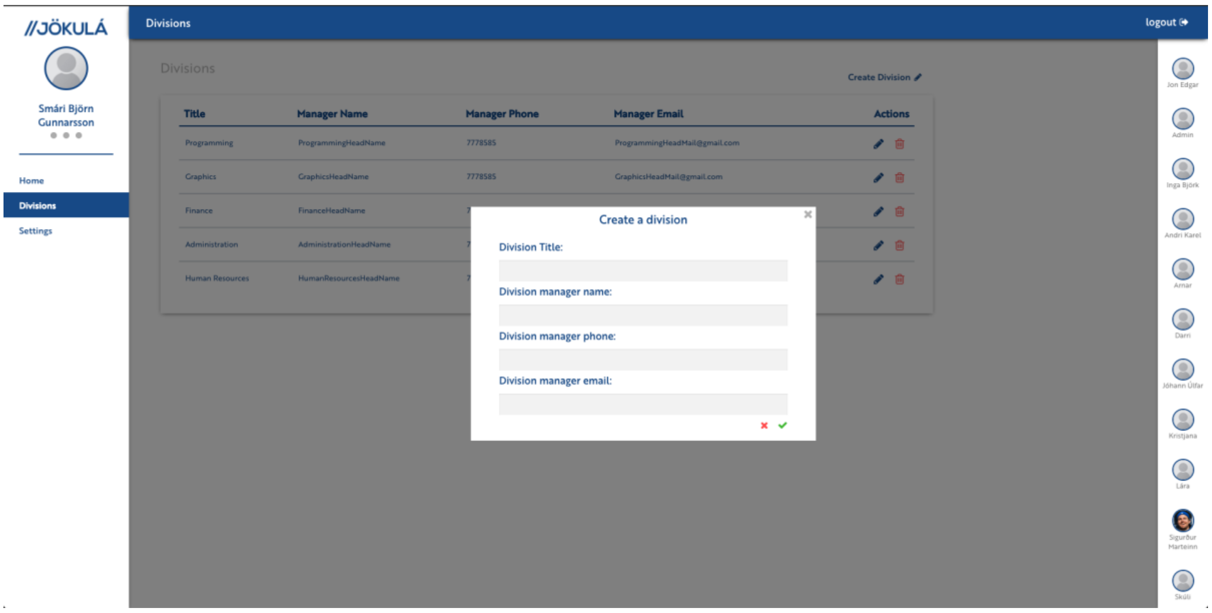
Division head name: ProgrammingHeadName

Division head phone: 7778585

Division head email: ProgrammingHeadMail@gmail.com

Figure 3.30: Human resources update division

The create form for divisions for the HR employee (see figure 3.31).



Divisions

Divisions

Create Division

Title	Manager Name	Manager Phone	Manager Email	Actions
Programming	ProgrammingHeadName	7778585	ProgrammingHeadMail@gmail.com	
Graphics	GraphicsHeadName	7778585	GraphicsHeadMail@gmail.com	
Finance	FinanceHeadName	7		
Administration	AdministrationHeadName	7		
Human Resources	HumanResourcesHeadName	7		

Create a division

Division Title:

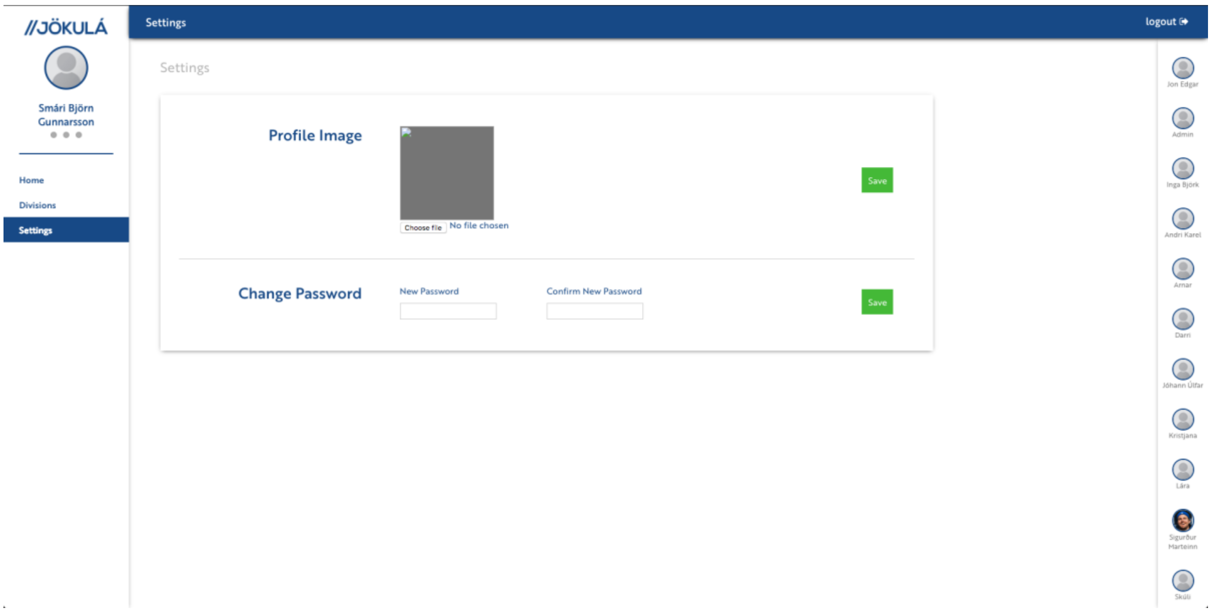
Division manager name:

Division manager phone:

Division manager email:

Figure 3.31: Human resources create division

The settings page for the HR employee (see figure 3.32).



Settings

Settings

Profile Image

Choose file No file chosen

Save

Change Password

New Password

Confirm New Password

Save

Figure 3.32: Human resources settings

3.4 Salesman

We did not implement all functionality for the salesman. But if the system were completed with all B and C tasks the salesman frontpage would display on overview of offers that he could negotiate and accept/decline. When offers would be accepted they would be broken down into sub-projects and tasks by the project manager (see figure 3.33).

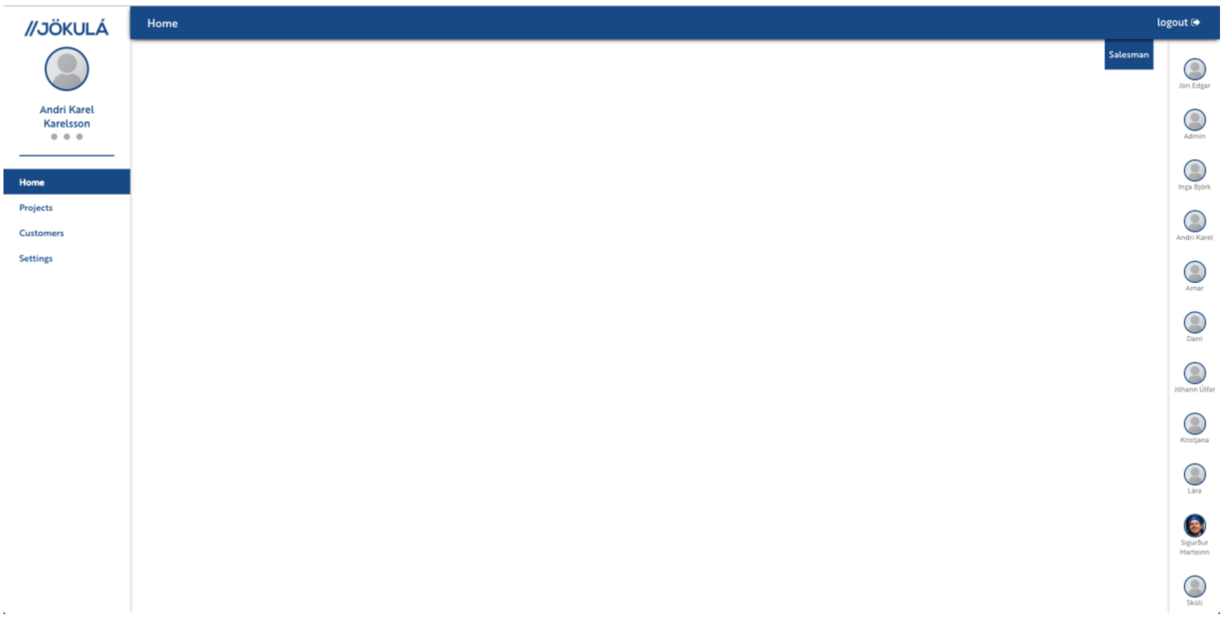
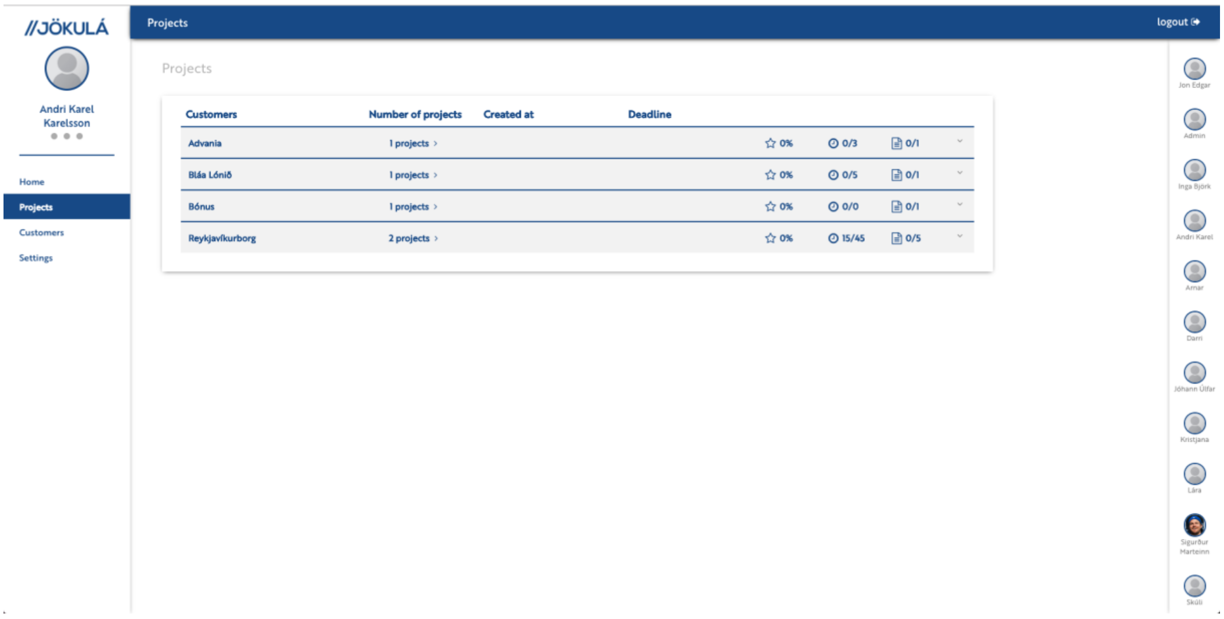


Figure 3.33: Salesman homepage

As with the project manager the salesman has access to the projects view (see figure 3.34). But only read access.

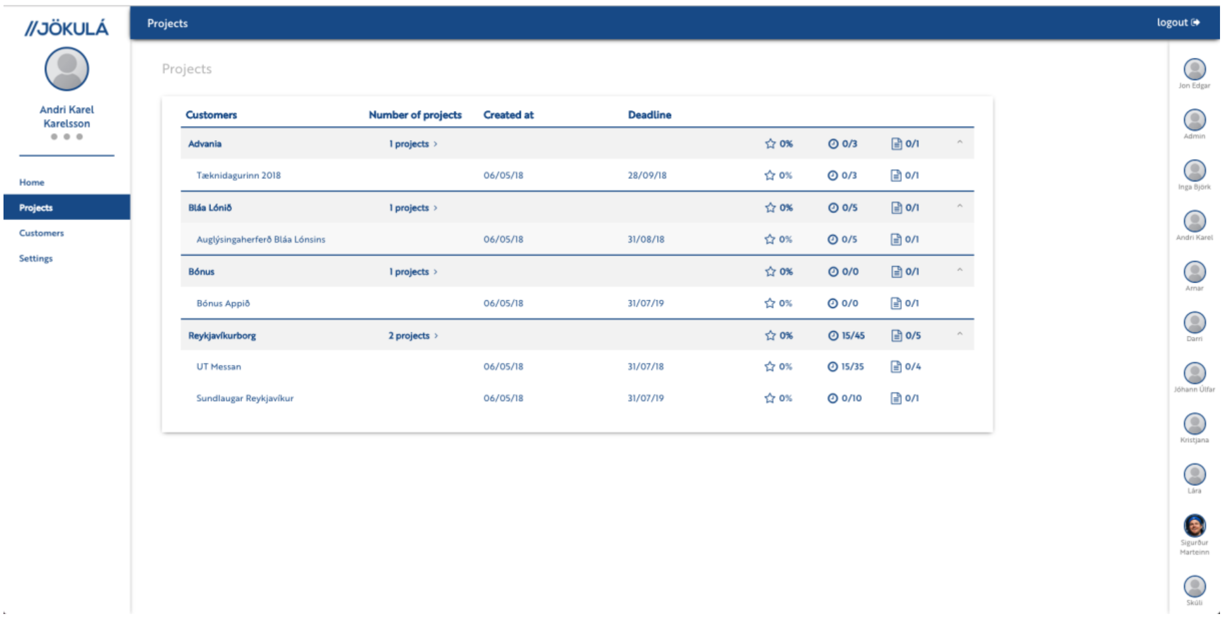


Projects

Customers	Number of projects	Created at	Deadline			
Advania	1 projects ›			☆ 0%	0/3	0/1
Bláa Lónið	1 projects ›			☆ 0%	0/5	0/1
Bónus	1 projects ›			☆ 0%	0/0	0/1
Reykjavíkurborg	2 projects ›			☆ 0%	15/45	0/5

Figure 3.34: Salesman projects view

Expanded table view of the projects page for salesman (see figure 3.35).

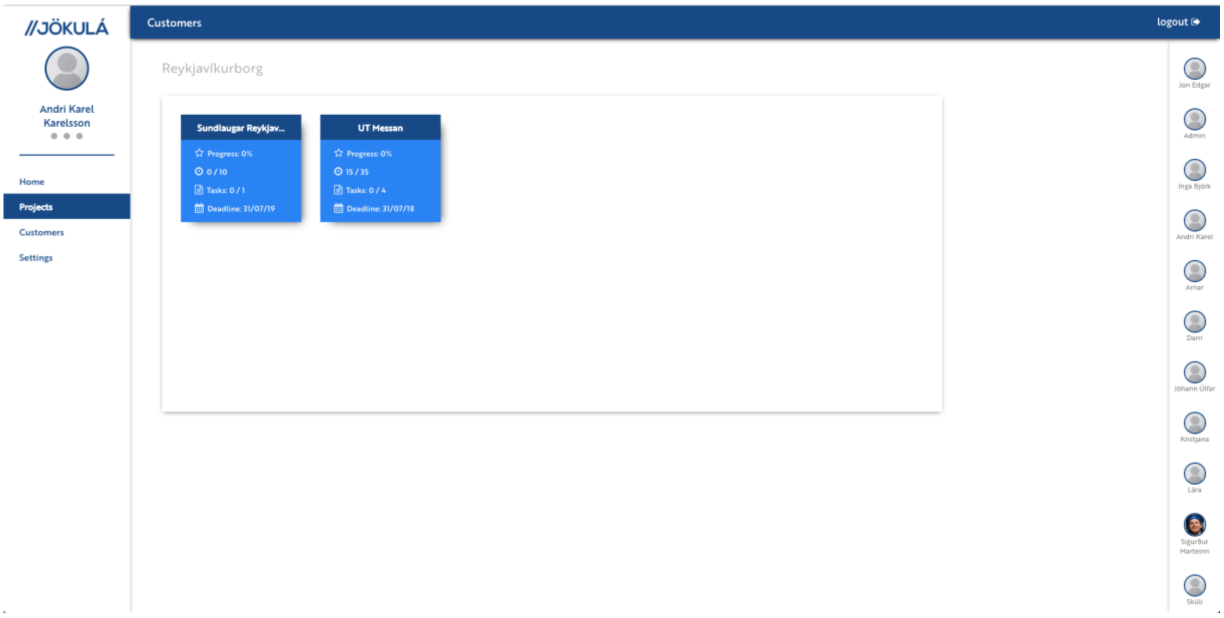


Projects

Customers	Number of projects	Created at	Deadline			
Advania	1 projects ›			☆ 0%	0/3	0/1
Tæknidagurinn 2018		06/05/18	28/09/18	☆ 0%	0/3	0/1
Bláa Lónið	1 projects ›			☆ 0%	0/5	0/1
Auglýsingaherferð Bláa Lónsins		06/05/18	31/08/18	☆ 0%	0/5	0/1
Bónus	1 projects ›			☆ 0%	0/0	0/1
Bónus Appið		06/05/18	31/07/19	☆ 0%	0/0	0/1
Reykjavíkurborg	2 projects ›			☆ 0%	15/45	0/5
UT Messan		06/05/18	31/07/18	☆ 0%	15/35	0/4
Sundlaugar Reykjavíkur		06/05/18	31/07/19	☆ 0%	0/10	0/1

Figure 3.35: Salesman projects expanded

The more detailed view for tasks belonging to a specific customer for salesman (see figure 3.36).



Reykjavíkurborg

Sundlaugar Reykjav...
Progress: 0%
0 / 10
Tasks: 0 / 1
Deadline: 31/07/19

UT Messan
Progress: 0%
15 / 35
Tasks: 0 / 4
Deadline: 31/07/18

Figure 3.36: Salesman projects by customers

The view for a specific project (see figure 3.37). The salesman can read sub projects and tasks, but does not have create, update or delete access.

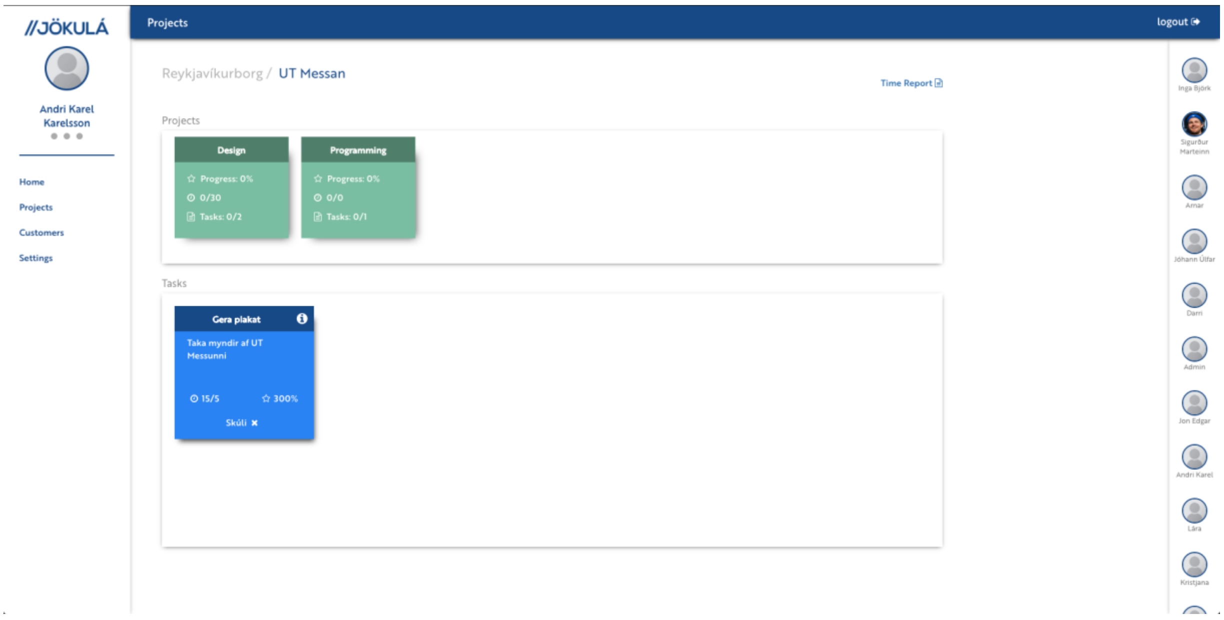
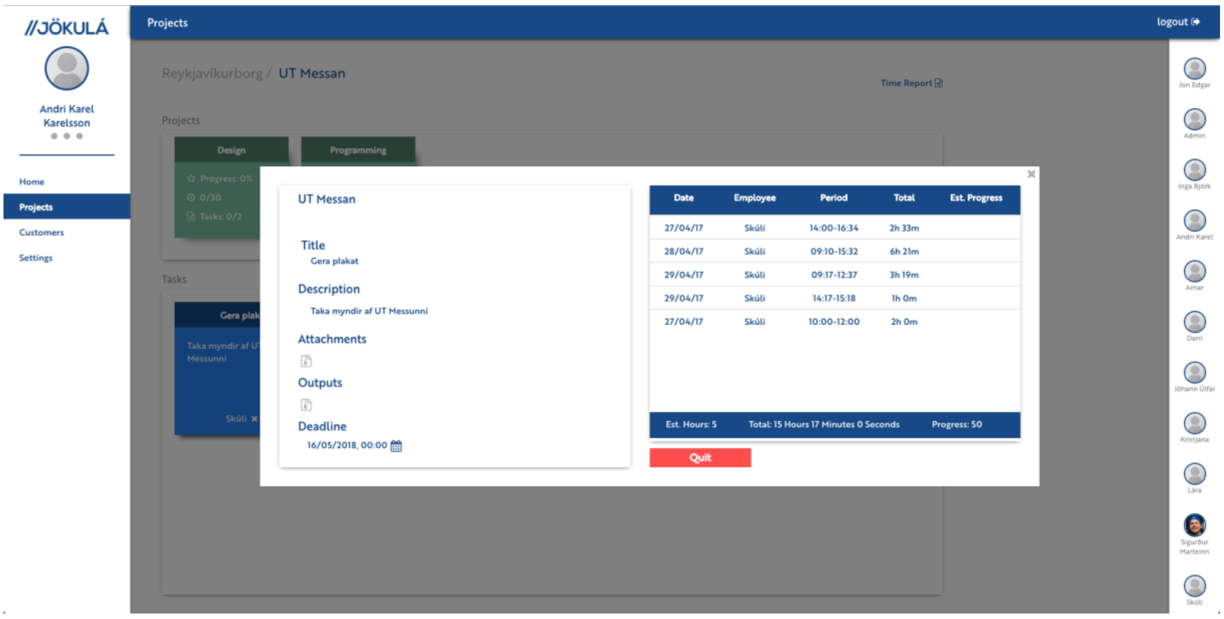


Figure 3.37: Salesman specific project view

He also has read access to tasks (see figure 3.38) but not create, update or delete access.



UT Messan

Title
Gera plakat

Description
Taka myndir af UT Messunni

Attachments

Outputs

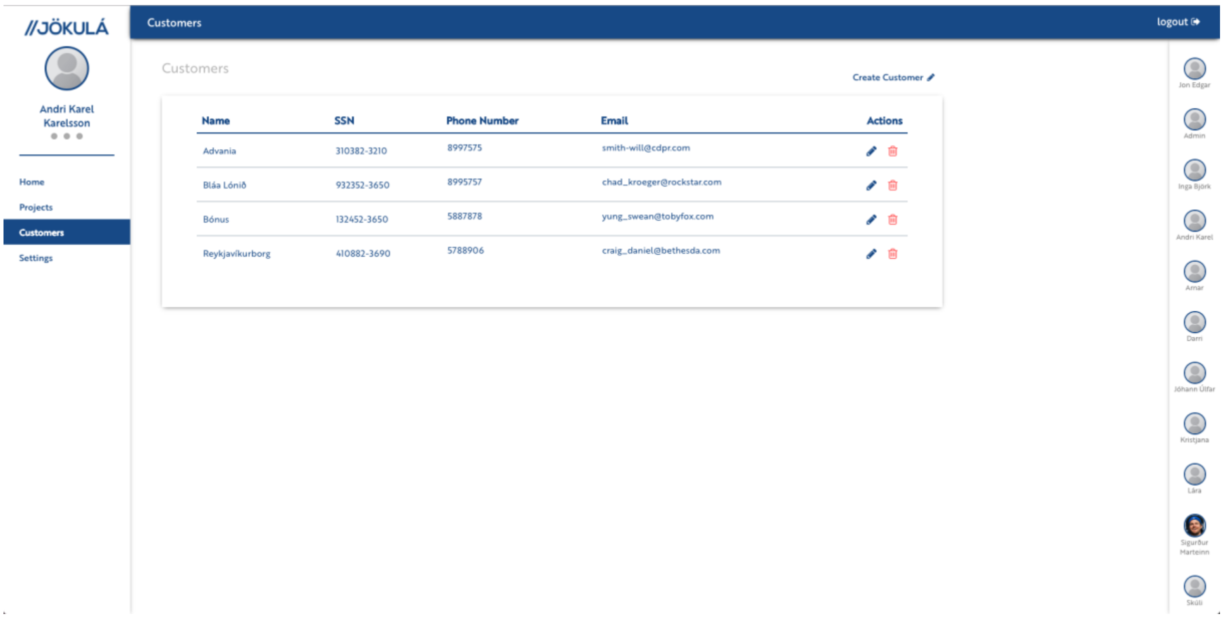
Deadline
16/05/2018, 00:00

Date	Employee	Period	Total	Est. Progress
27/04/17	Skúli	14:00-16:34	2h 33m	
28/04/17	Skúli	09:10-15:32	6h 21m	
29/04/17	Skúli	09:17-12:37	3h 19m	
29/04/17	Skúli	14:17-15:18	1h 0m	
27/04/17	Skúli	10:00-12:00	2h 0m	

Est. Hours: 5 Total: 15 Hours 17 Minutes 0 Seconds Progress: 50

Figure 3.38: Salesman project details

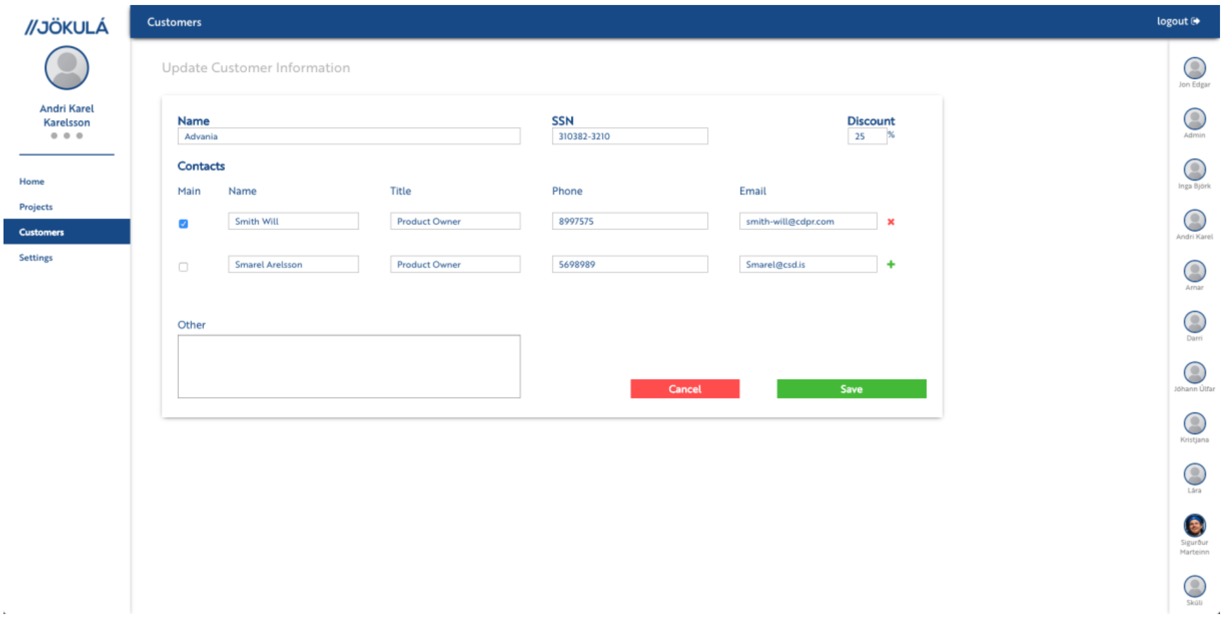
The salesman has complete CRUD access to customers (see figure 3.39).



Name	SSN	Phone Number	Email	Actions
Advania	310382-3210	8997575	smith-will@cdpr.com	
Bláa Lónið	932352-3650	8995757	chad_kroeger@rockstar.com	
Bónus	132452-3650	5887878	yung_swean@tobyfox.com	
Reykjavíkurborg	410882-3690	5788906	craig_daniel@bethesda.com	

Figure 3.39: Salesman customer view

The update form for customers for the salesman (see figure 3.40).



Name: Advania
SSN: 310382-3210
Discount: 25 %

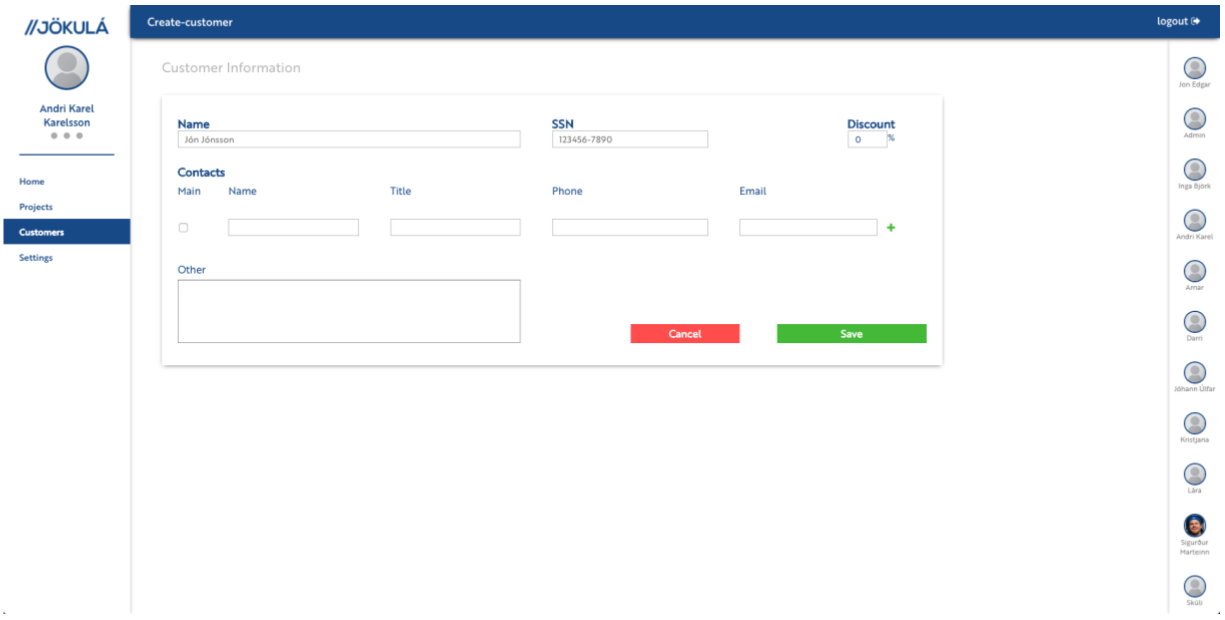
Contacts

Main	Name	Title	Phone	Email
✓	Smith Will	Product Owner	8997575	smith-will@cdpr.com
	Smarel Arelsson	Product Owner	5698989	Smarel@csd.is

Other

Figure 3.40: Salesman update customer

The create form for customers for the salesman (see figure 3.41).



Name: Jón Jónsson	SSN: 123456-7890	Discount: 0 %

Figure 3.41: Salesman create customer

The salesman settings page (see figure 3.42).

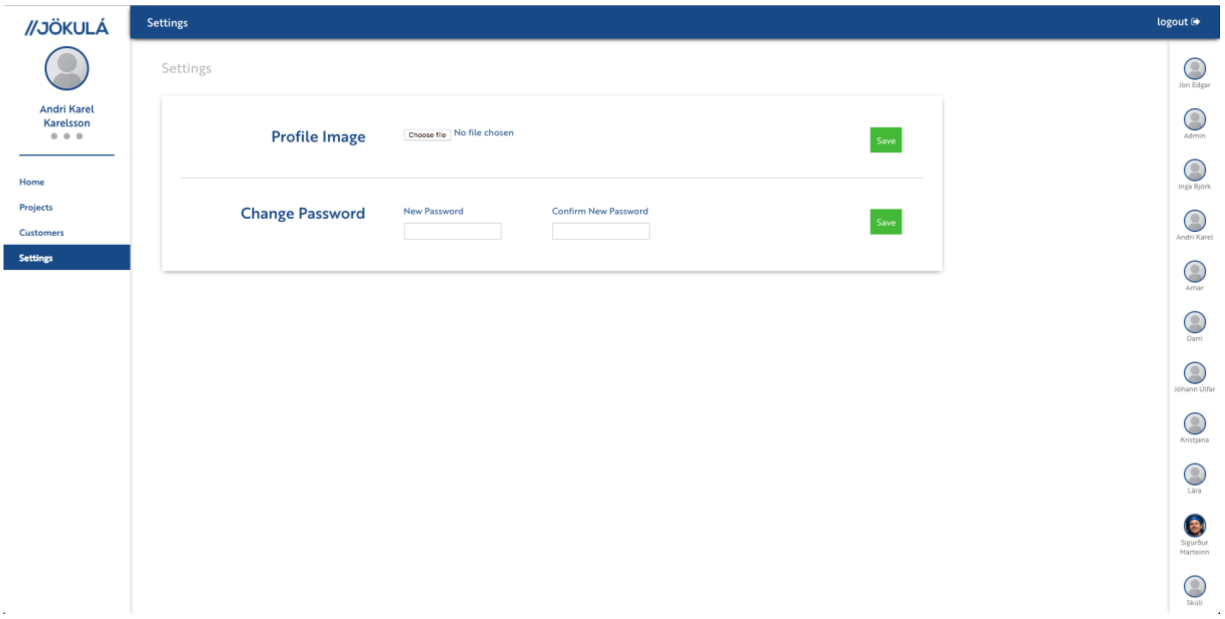
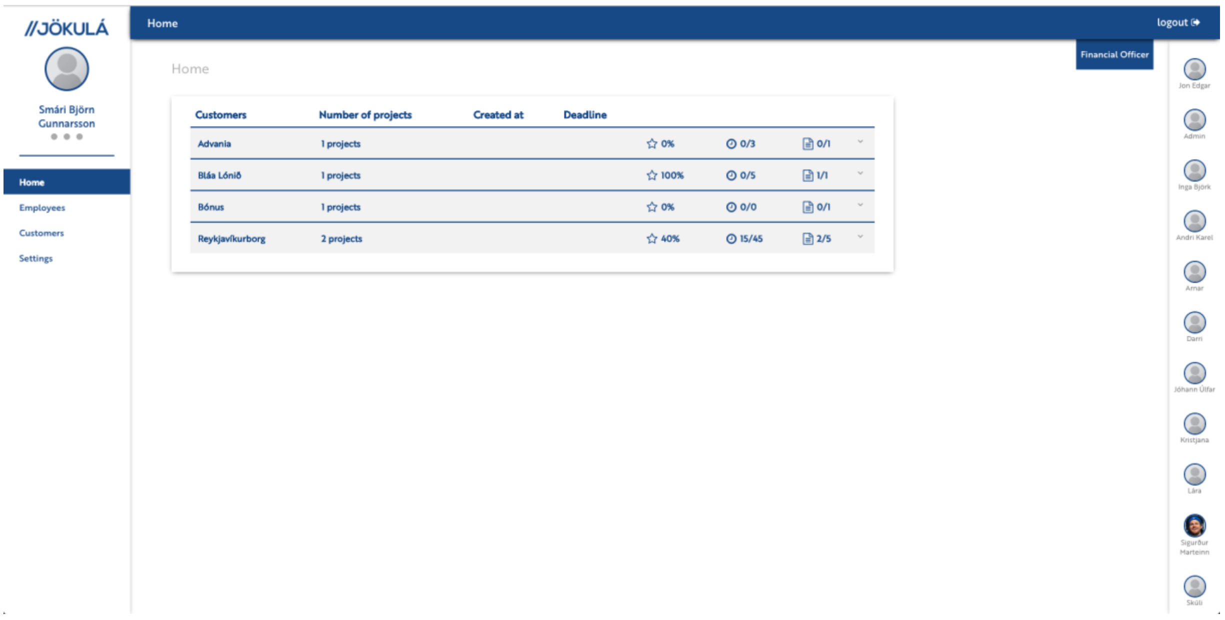


Figure 3.42: Salesman settings

3.5 Financial officer

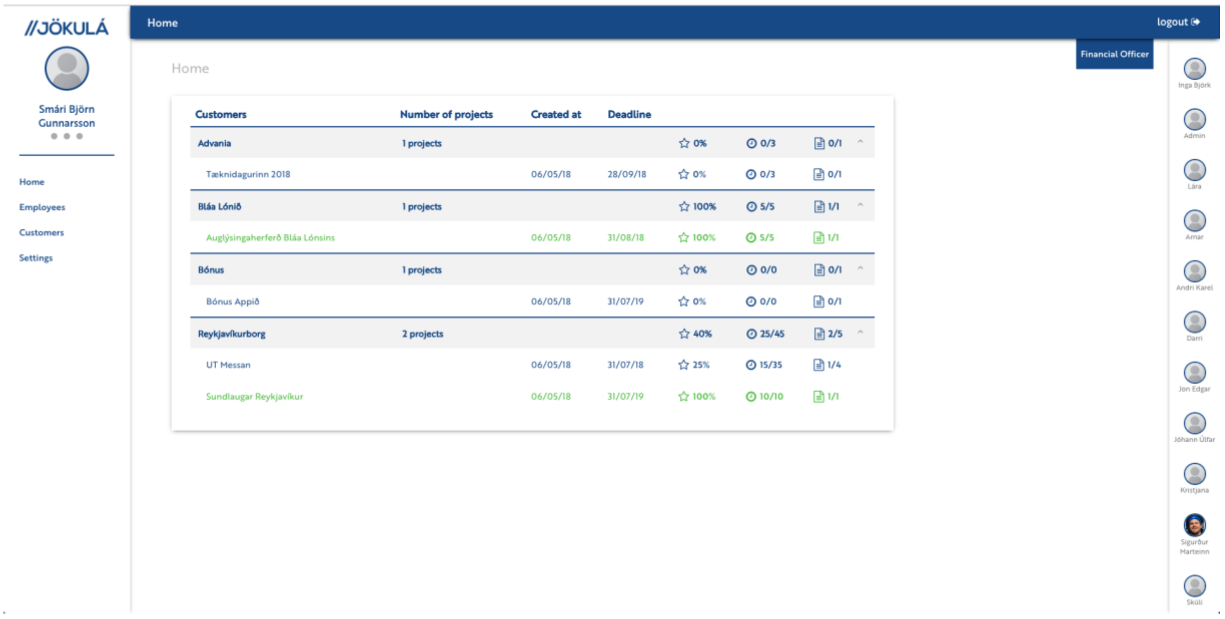
The Financial Officer’s homepage is a table of all customers that behaves as usual except when he has chosen a project he will get a detailed time report of tasks worked on that project so he can invoice it (see figure 3.43).



Customers	Number of projects	Created at	Deadline			
Advania	1 projects			☆ 0%	0/3	0/1
Bláa Lónið	1 projects			☆ 100%	0/5	1/1
Bónus	1 projects			☆ 0%	0/0	0/1
Reykjavíkurborg	2 projects			☆ 40%	15/45	2/5

Figure 3.43: Financial officer homepage

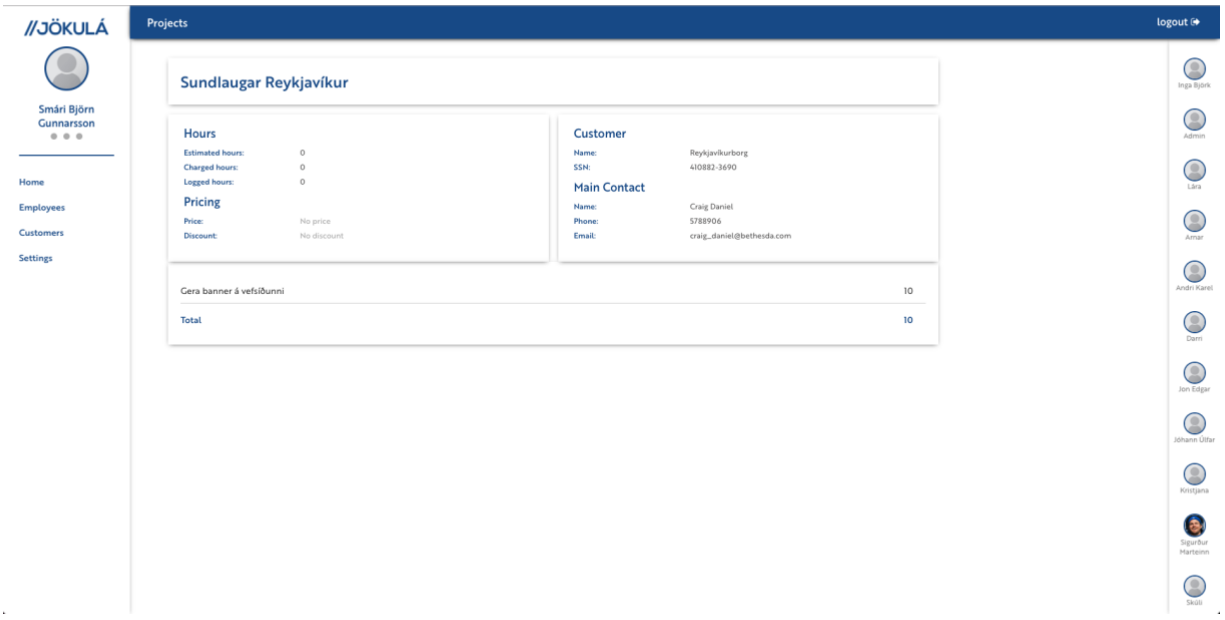
The financial officer can see a more detailed overview if he expands the customers to see their projects and their progress status (See figure 3.44).



Customers	Number of projects	Created at	Deadline			
Advania	1 projects			☆ 0%	0/3	0/1
Tæknidagurinn 2018		06/05/18	28/09/18	☆ 0%	0/3	0/1
Bláa Lónið	1 projects			☆ 100%	5/5	1/1
Auglýsingaherferð Bláa Lónsins		06/05/18	31/08/18	☆ 100%	5/5	1/1
Bónus	1 projects			☆ 0%	0/0	0/1
Bónus Appið		06/05/18	31/07/19	☆ 0%	0/0	0/1
Reykjavíkurborg	2 projects			☆ 40%	25/45	2/5
UT Messan		06/05/18	31/07/18	☆ 25%	15/35	1/4
Sundlaugar Reykjavíkur		06/05/18	31/07/19	☆ 100%	10/10	1/1

Figure 3.44: Financial officer homepage expanded customers

The financial officers main view will come if he choses a project from his homepage. He will then end on a view (see figure 3.45) where he gets the customers details so he can invoice them and the hours worked. He gets details about how much time is being invoiced on each task to be able to break the invoice down if needed.



Sundlaugar Reykjavíkur

Hours

Estimated hours:	0
Charged hours:	0
Logged hours:	0

Pricing

Price:	No price
Discount:	No discount

Customer

Name:	Reykjavíkurborg
SSN:	410882-3690

Main Contact

Name:	Craig Daniel
Phone:	5788906
Email:	craig_daniel@bethesda.com

Gera banner á vefsíðunni	10
Total	10

Figure 3.45: Financial officer timereport for invoice

The financial officers settings page (See figure 3.46)

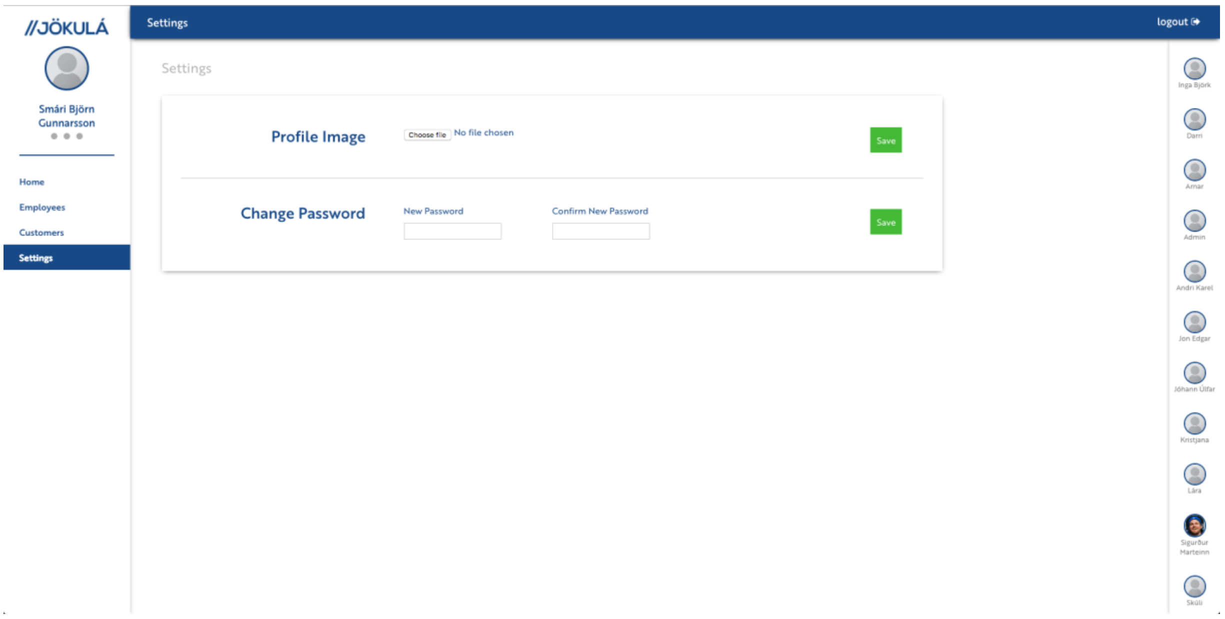


Figure 3.46: Financial officer settings

3.6 Admin

All of the admins functionality has been seen before in by other roles. The admin has CRUD access to everything. The only views he will not be able to are those specific to a role, such as the financial managers final time report or the project managers frontpage to be able to hide or show tasks. The admin can however assign the role of financial officer or project manager to himself giving him access.

4 Development Environment

//JÖKULÁ decided which development environment and technologies we would use for the project. They use the same development environment for their own projects and have a good understanding of the technologies. //JÖKULÁ decided that the project would be written in Node.js while using the Express.js framework. The database that is used is PostgreSQL but we use Bookshelf.js as an ORM on top of PostgreSQL. We use GitHub for source control. Below is a more detailed list of all the technologies used in our system.

4.1 Back end

- The back end is written using **Node.js** v8.9.* LTS runtime environment using the **Express.js** Web application framework.
- The web service is designed using the **REST** architecture style.
- Authentication is implemented using **Passport** with **local strategy.**
- The database is a **PostgreSQL** database.
- **Bookshelf.js** is an ORM that maps the **PostgreSQL** database.
- **Bookshelf.js** uses **Knex.js** as a query builder.
- Unit testing is executed using the **Mocha.js** test framework.
- Code coverage reports are generated with **Istanbul.**
- **JSHint** is used to enforce coding standards and to detect errors and potential problems in code.
- Build scripts are set up using **Gulp** task runner.
- Tests are written with **"Behaviour Driven Development"** in mind.
- We strive to use **Ecmascript 6** where it is supported.
- **NPM** is used for package management.
- **Webpack.js** is used as a module bundler and watcher.
- **SuperTest** test framework is used for agent driven authentication and permission testing.
- **CircleCI** runs unit tests when the master branch is updated.

4.2 Front end

- Written in **Vue.js** and uses the **Vue Router** framework.
- Has a centralized store using **Vuex** state management library.
- CSS is written using **Less.**

- We strive to use **Ecmascript 6. Babel** compiles code to a supported version for browsers that do not support Ecmascript 6.
- **Pug** is used template engine for HTML code.

4.3 Version Control

The **git** version control system is used along with **GitHub** as a source code management system. We use branching as much as possible where we keep the master branch clean and push only fully functional code onto it. We keep a development branch where we keep new features that are not fully tested. We merge the development branch into the master branch when the development branch reaches stable state. When working on new features, each developer works on a dedicated feature branch and merges it with the development branch when the feature is completed.

5 Work Arrangement

5.1 Methodology

The team used the Scrum methodology when implementing the system. That includes working in sprints, managing a product backlog and sprint backlogs as well as making burndown charts for each sprint and for the entire project.
We decided that each sprint should be two weeks long. Each sprint started with taking several user stories and dividing them into tasks that went into the sprint backlog for that particular sprint. Each task was then assigned to one or more team member.

5.2 Scrum

When using Scrum defining certain roles within the team is a necessity. The roles in our team were assigned in the following manner:

Name	**Role**
Georg Kristinsson	Product Owner
Smári Björn Gunnarsson	Scrum Master
Andri Karel Júlíusson	Scrum Team Member
Sigurður Marteinn Lárusson	Scrum Team Member
Skúli Arnarsson	Scrum Team Member

Table 5.1: Scrum Roles

We used Microsoft Excel Online for managing our product backlog and sprint backlogs. The product backlog consists of user stories and for each user story we determined it’s priority and tried to estimate how many story points it was. We used planning poker to determine the number of story points for each story in the backlog. We use Google Drive to manage our documentation.

5.3 Sprint Arrangements

The sprints were arranged in the following way: Each sprint was two weeks long until the exam period started. Over the exam period there was one sprint that was three weeks long, and finally there were three one week long sprints at the end of the semester. The first sprint started on January 29th.

The team met at Thursdays and Fridays each week in the workspace provided to us by //JÖKULÁ at their office. If for some reason we couldn't meet at the days we decided that we would meet on weekends to catch up.

5.4 Workplace Arrangements

We were provided with our own workspace at //JÖKULÁ and access to the building at all times. Each team member was also provided with an office desk and chair, a keyboard, a mouse and a screen to connect their laptops to. Each team member brought their own laptop to work on for the project. Members of the team also had access to everything else in the building, including the kitchen, bathrooms and meeting rooms. The team also had access to the programmers and the other staff of //JÖKULÁ if they had any questions or needed any advice.

5.5 Meetings

5.5.1 Scrum

We started every day that we worked on the project with a daily scrum meeting where every member of the team had to attend. These were short meetings where each member of the team explained shortly what he had been working on since the last meeting, what he intended to do for the day and if he had any problems that prevented him from completing his tasks. We also dedicated a part of the meeting to updating the scrum board for the current sprint so each team member could keep track of the overall progress and see what the other team members were doing.

At the end of each sprint we held retrospective meetings for that particular sprint. The purpose of these meetings was to reflect on the sprint's progress, what could have gone better and if there were any problems during the sprint. In these meetings we always kept in mind what we learned from previous retrospective meetings and sprints. This information then helped us in planning the following sprints.

5.5.2 Other meetings

Every week we met with the product owner on Thursdays at 14:00. The main purpose of these meetings was to make sure the project was going in the right direction and that all team members were on the same page.

We also planned meeting with two of the programmers working for //JÖKULÁ each Friday at 13:00. The programmers performed a short code review on the project and point out things that might be better or different. Unfortunately these programmers were quite busy over the semester so this was often not the case, although we did manage to meet them a few times and get some pointers from them.
Finally we met with our instructor every week on Thursdays at 13:00. The purpose of these meetings was to get feedback from him about the project or to get his help if we had some issues regarding the project, //JÖKULÁ or other members of the team.

	Sun	Mon	Tue	Wed	Thu	Fri	Sat
10:00-12:00	Backup Day				Work on project	Work on project	Backup Day
13:00-14:00	Backup Day				Meeting with instructor 13:00	Code review 13:00	Backup Day
14:00-15:00	Backup Day				Meeting with product owner 14:00	Work on project	Backup Day
14:00-17:00	Backup Day				Work on project	Work on project	Backup Day

Table 5.2: The teams weekly schedule.

5.6 Time Logging

Time logging was done using an Excel Online spreadsheet where every member kept their own log. For each entry in the log, each team member wrote down what they were working on, when whey worked on a particular task, how much time he spent working on it along with an optional comment. Each record in the sheet was then categorized by it's type, the categories included Programming, Documentation and Meetings. At the end of each day each member would also review the time log for the day and make sure all entries had been logged correctly.

5.7 Documentation

When starting the project the team decided right away to try to document everything as well as possible. This goal was reached to some degree. All our documents are stored on Google Docs in a shared folder where every member of the team can access them at any time.

5.7.1 API & Service Documentation

Documentation for routes and services are written in a comment above each route and service. These comments are written in the following format in order to be rendered correctly in a browser:

```
/**
 * @api {get} /user/:id Request User information
 * @apiName GetUser
 * @apiGroup User
 * @apiParam {Number} id Users unique ID
 * @apiError (4xx) 400 Object contains bad data
 * @apiSuccess {String} firstname Firstname of the User
 * @apiSuccess {String} lastname  Lastname of the User
 */
```

Figure 5.1: API Documentation example

Documentation can be generated from documentation comments by executing `'npm run document'`.

The documentation for the project can be found in the Documentation folder found in root directory of the project itself. Documentation for the API can we viewed in a browser by opening index.js file found in the api folder inside the Documentation folder. Documentation for the services are viewable under the services folder inside the Documentation folder. The documentation for the services is written in the markdown markup language and are easily viewable using a markdown viewer of your preference.

5.7.2 Development Guide

The development Guide is a guide meant for new developers that are starting to work on the project. It details the prerequisites needed for the project, how to get it up and running as well as the overall structure of it.

5.7.3 User Manual

The User Manual includes simple instructions on how to use the more complex interfaces of the system.

5.8 Ownership of the product

The product including all code and documentation entirely belongs to //JÖKULÁ.

6 Planning and Progress

In this section the overall plan for the semester is explained, how the sprints were distributed over the semester and how long each sprint was. Each sprint is then explained in more detail in Appendix 1. This section also details the time logged by each member of the team and how the logged hours are divided between them as well as how much time was logged in each category.
Here we also shortly describe our requirements: what requirements are in the product backlog and which of them we finished. The requirements themselves can be found in Appendix 3.

6.1 Sprint Schedule

This is the schedule that we prepared for sprints along with goals that we set to achieve in each sprint. We were able to keep to the sprint schedule for the most part. The front end fell a bit behind schedule but were able to rectify that problem by moving two of our team members exclusively to the front end. The system was close to a minimum viable product at the end of sprint 5 as we had scheduled. However we would not have been comfortable presenting the system as a MVP until after the 6th sprint, when the front end started to take shape.

Sprint #	Begins	Ends	Main Goals
0	15/01	28/01	**Design**: Everyone has a good overview of the requirements. **Documentation**: Project schedule, Product backlog, Work plan. **Project**: Setup required dependencies and environment.
1	29/01	11/02	**Design**: Wireframes, database tables. **Documentation**: Risk analysis. Wireframes, progress plan and burndown charts. **Project**: Authentication, basic routes, basic views. **1st status meeting.**
2	12/02	25/02	Back end functions for employee,divisions and roles, basic front end to show functionality of server & Database implementation
3	26/02	11/03	**Project**: Back end functions for customer, projects, front end implementation on customer & projects
4	12/03	1/04	**Project:** Task back end/front end implementation. Link employees to tasks.

			Front end views for different roles. Basic styling on client **2nd status meeting**
5	2/04	22/04	**Project:** Timestamp implementations in back end/front end. Project manager able to assign tasks to employees. Minimum viable product ready
6	23/04	29/04	Exam period. Team all work on front end, all back end functions nearly complete. Employees can log time on tasks. Project manager can see stats about projects
7	30/04	6/05	**Documentation**: Development guide, Instruction manual, Handoff documentation. Bug fixing **Preparation for 3 and final status meeting**
8	7/05	11/05	**Documentation:** Final report for the project **Programming:** Minor bug fixing before handin

Table 6.1: Sprint Schedule.

6.2 Requirements

We got a document from //JÖKULÁ describing the permissions and roles of the system and the actions available to each role and we translated that into our product backlog. We estimated each story in story points where each story point represented some arbitrary amount of time. We estimated the stories using planning poker. Planning poker works in the way that each member of the team reflects on the task and estimates how many story points it is on his own. All team members then have to show their estimate and explain the reason they picked their estimate. If the team members disagree about some estimate they should state their argument and then estimate again on their own. This is done until an agreement has been reached for every user story. When we had translated all the requirements into the backlog and estimated them we assigned a priority to them ranging from A to C, with A being the highest priority. Our backlog was then essentially complete and was updated frequently over the projects duration. The backlog can bee seen in Appendix 3 in this report.

We estimated that we would be able to implement all of the A priority user stories in the product backlog. Ideally we also wanted to implement the B priority stories, however we believed that the C priority stories would not be implemented as a part of this course but they may be implemented if //JÖKULÁ decides to continue the project.

We ended up completing most of the A priority stories (except the ones related to the salesman role), most of the B priority stories as well and some of the C priority stories.

The total number of story points was 368.
The number of story points for A priority stories was 196.

6.3 Distribution of working time

The estimated required time to complete the project is about 1300 to 1400 hours according to //JÖKULÁ.
The graph and table below explain how the hours worked were divided between categories. It is worth noting that the testing category probably has higher amount of hours worked than is presented in our data due to members sometimes registering hours spent testing as hours spent programming. This happened on occasion because the team decided that when programming the back end the tests for that particular back end function should always be written right after. This resulted in both types of categories being registered as programming.

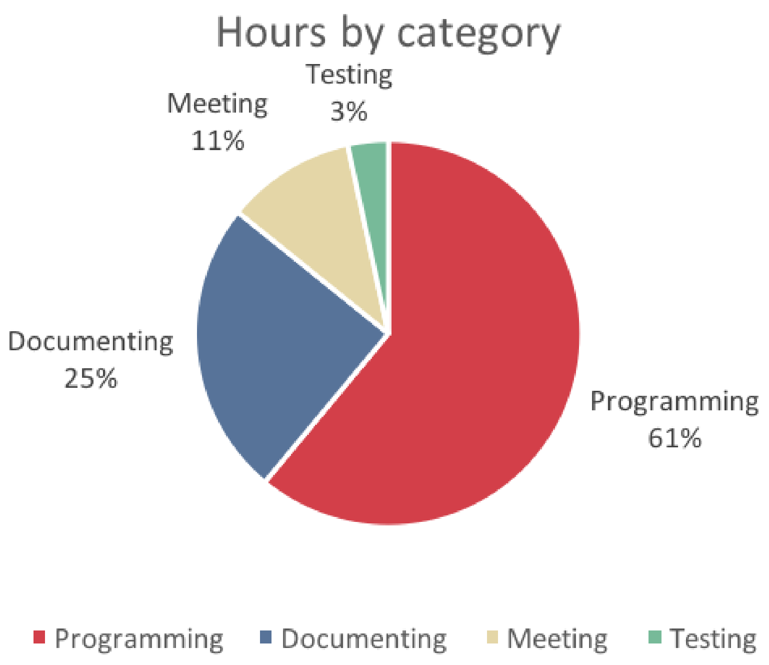


Figure 6.1: Breakdown of hours worked per category.

Number	Category	Hours
1	Programming	830.25
2	Documenting	337
3	Meeting	147
4	Testing	46
	Total	**1360.25**

Table 6.2: Breakdown of hours worked per category.

On the graph and table below we can see how the the hours spent on the project were divided between members of the team. The initial goal we set for hours worked on the project was 300 hours per team member. Every member of the development team reached the goal that was initially set.

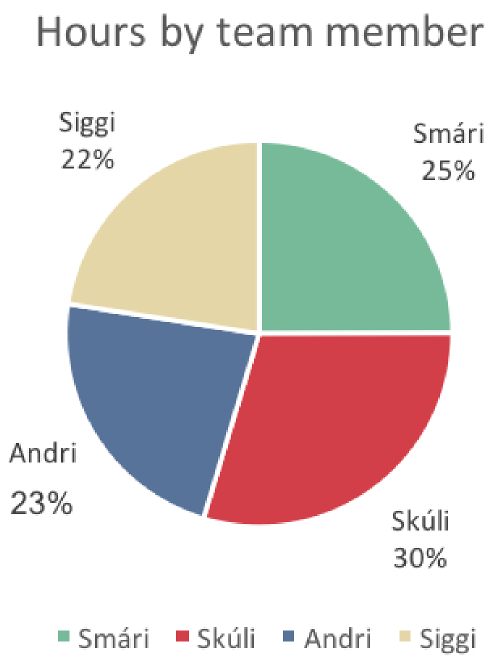


Figure 6.2: Breakdown of hours worked on the project by team members.

Andri	Siggi	Skúli	Smári	Total
310.5	306.5	403.75	339.5	**1360.25**

Table 6.3: Breakdown of hours worked on the project by team members.

7 System Design

The front end of the system is carefully implemented with the design we received from //JÖKULÁ in mind. The design went through a few iterations where the style was updated and features and components were refined. The design and implementation were made with the desktop first design strategy. This decision was made because the systems intended use is in an office environment.
The development team created diagrams that were used as a reference for implementing the system. These diagrams are an entity diagram for the database, a container diagram to show a high-level shape of the architecture, a flow diagram that in relation to different stages a project goes through and actions that change that a project is in and finally a component diagram that shows the individual parts of the system in greater detail.

7.1 Entity Diagram

The entity diagram shows the design of the database. We regularly updated the diagram as the database tables changed. Maintaining the database and creating database tables was a story that was present in most of the sprints and it was never completed until the latter part of our time working on the project. We used Bookshelf.js as an ORM while working on a postgreSQL database.

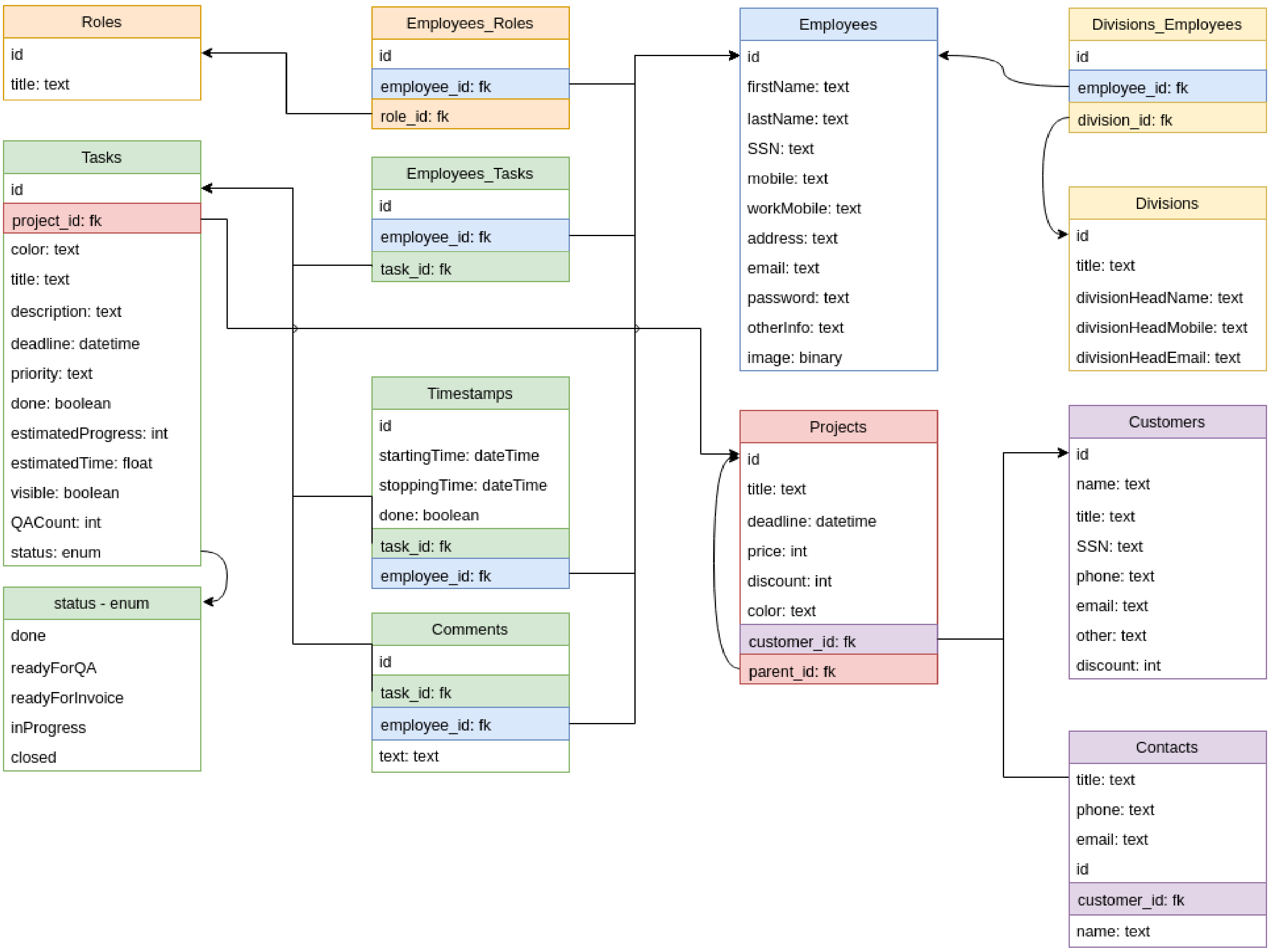


Figure 7.1: Entity Diagram

7.2 Container Diagram

This diagram show a high-level design of the architecture of the system, from the database layer to the presentation layer. The diagram shows how different components of the system interact. The container contains the components of the system itself. The front end communicates with the back end through API calls to a web service. The back end then calls the service layer that contains the logic that processes the requested action. The service calls the data access layer that writes to or reads information from the database. The information is then passed all the way back up into the front end. Currently the system is isolated, that is, the system does not communicate with any external services. Plans for the future include communication with an email server and the in-house file system at //JÖKULÁ.

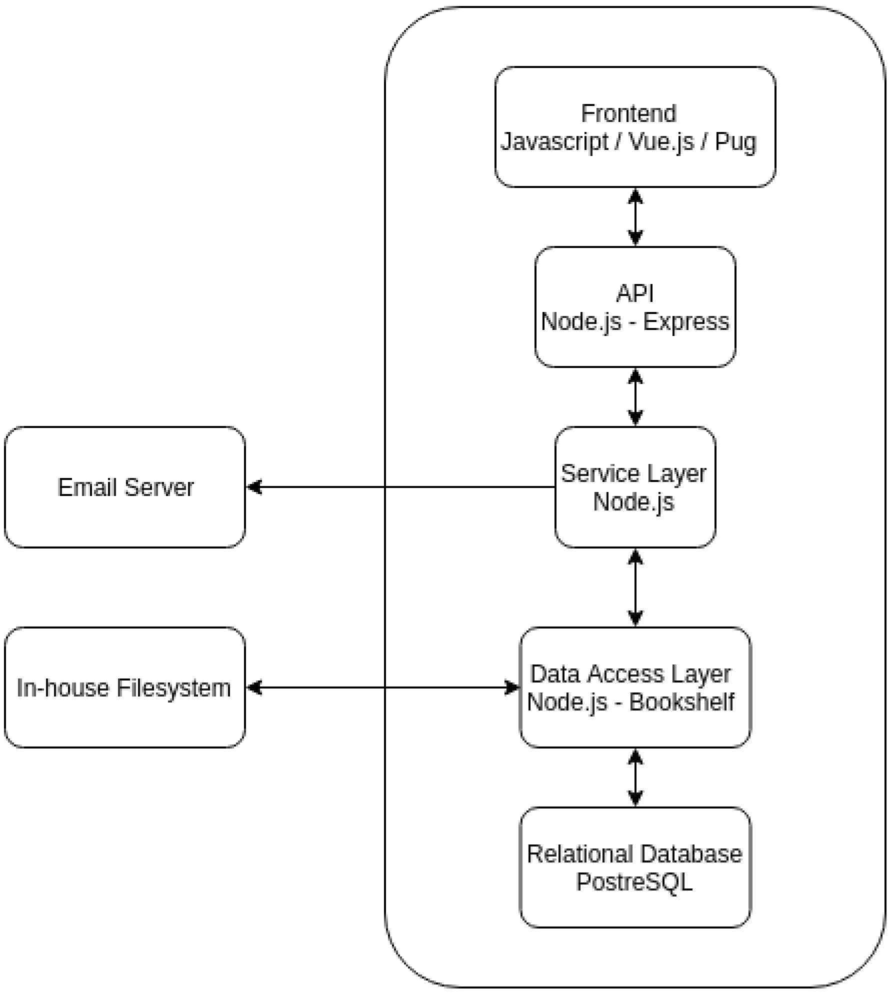


Figure 7.2: Container Diagram

7.3 Component Diagram

The component diagrams shows the interactions between components in more detail. The diagram shows in which layer individual components of the system belong along with the main technologies used within each layer. On the left side of the diagram there are arrows that indicate how different layers of the system communicate. Components in the front end show all the implemented User groups. The REST / API layer shows all the API services and the service layer shows all the services that are present in the system.

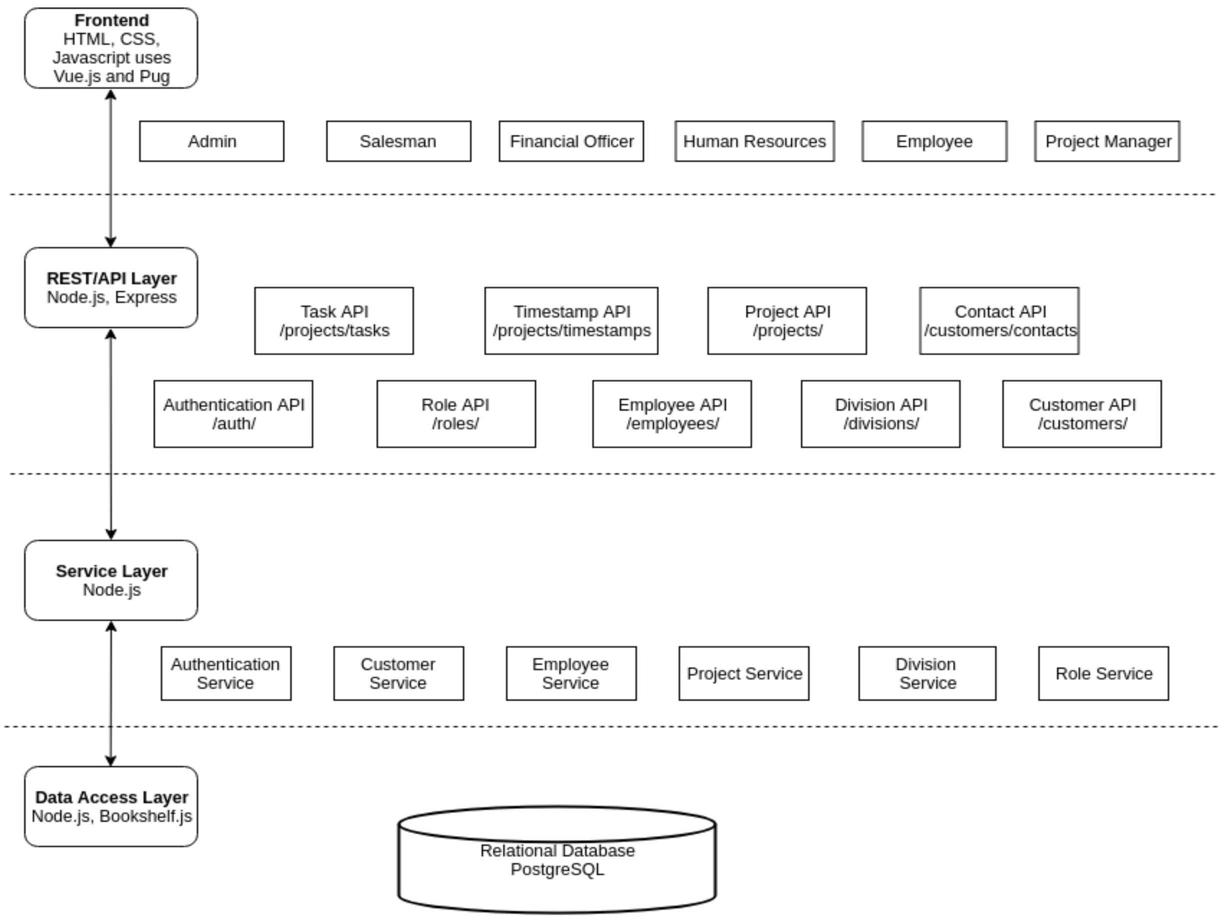


Figure 7.3: Component Diagram

7.4 Flow Diagram

The flow diagram shows the different stages that a Project goes through, from when the salesman creates the sales offer until a financial officer creates an invoice to be charged to a customer. Included in the diagram are functionalities that are expected to be present in the completed system. The green rounded box means that the colored action or role has been fully implemented. The orange rounded box means that it was not implemented and the yellow rounded box means that it was partially implemented.

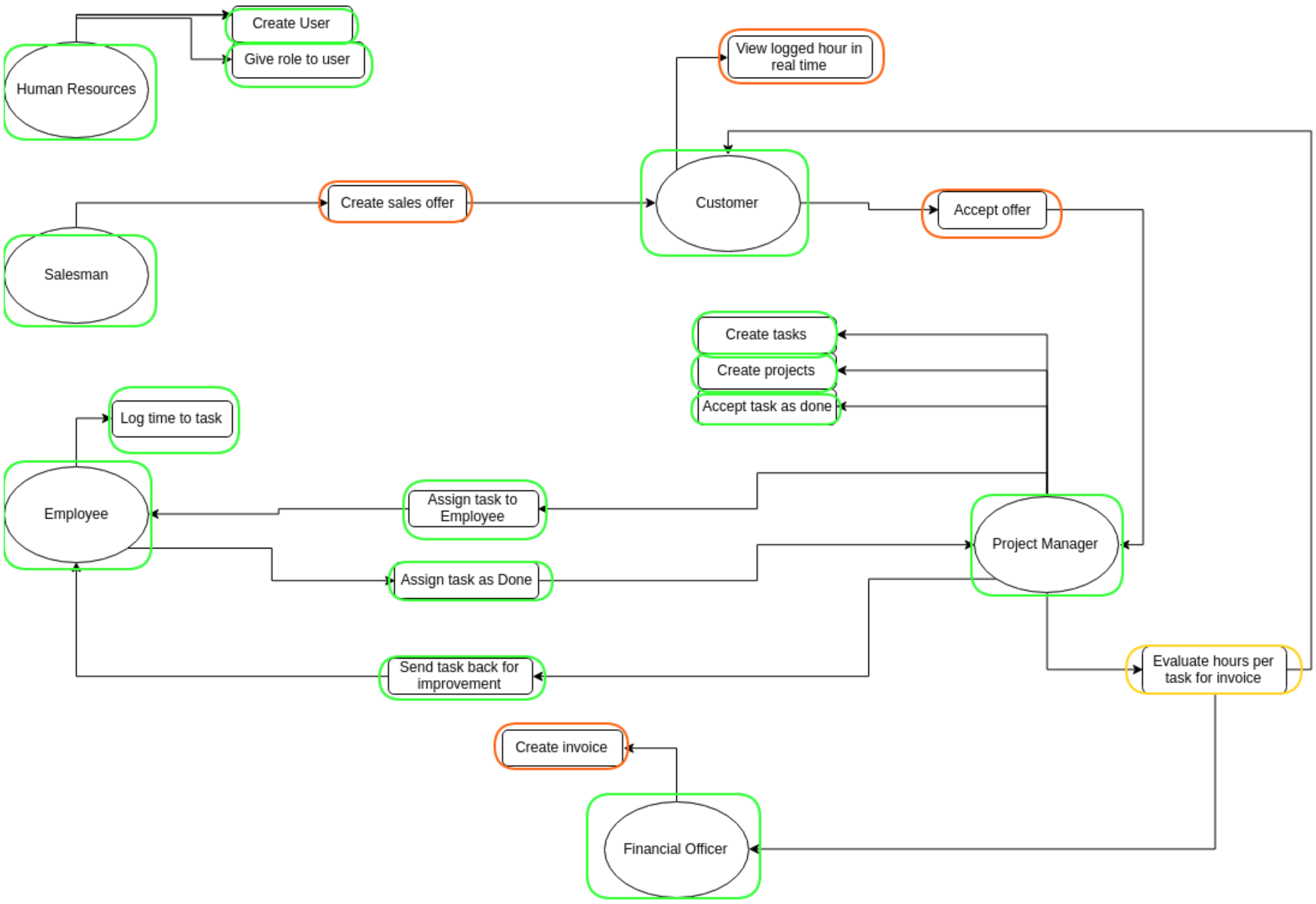


Figure 7.4: Flow diagram

8 Testing

8.1 Unit tests

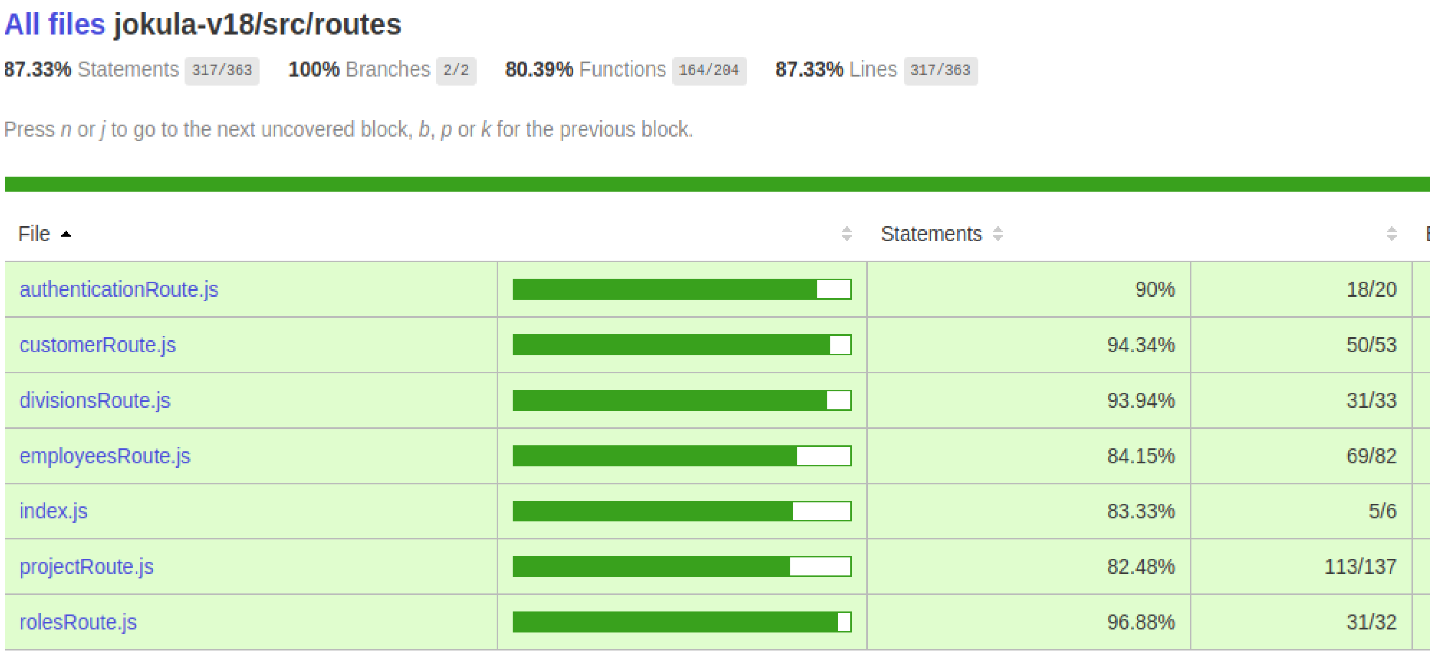
Unit test were written for all the routes and services in the system. The purpose of the unit tests is to ensure that the back end of the system always return the data it should and that all mutations of data are sound. The unit tests also make it much easier to pinpoint where a fault lies when something goes wrong and ensure the system works as expected after making some large change to it. The unit tests are executed on a mock database because they will not pass unless specific seed data is present and also because the unit tests create data that shouldn't exist in a production database. The unit tests were written using Mocha.js. The number of unit tests currently stands at 219, every one of them is passing.

8.2 Code coverage

At the time this was written the code coverage for the system stands at 87.33% for the API and 84.47% for the services. We decided to settle for 80-90% statement coverage because striving for 100% statement coverage would be wasted effort. A decision was made to put more emphasis on having effective tests that target a wide range of functionality in the back end of the system, instead of getting as many lines of code as we could. The system generates code coverage reports each time the unit tests are run with the command

```
'npm run test'.
```

The code coverage reports are generated in an HTML format and are easily viewed in a browser. The reports can be found under the coverage folder located in the projects root directory.



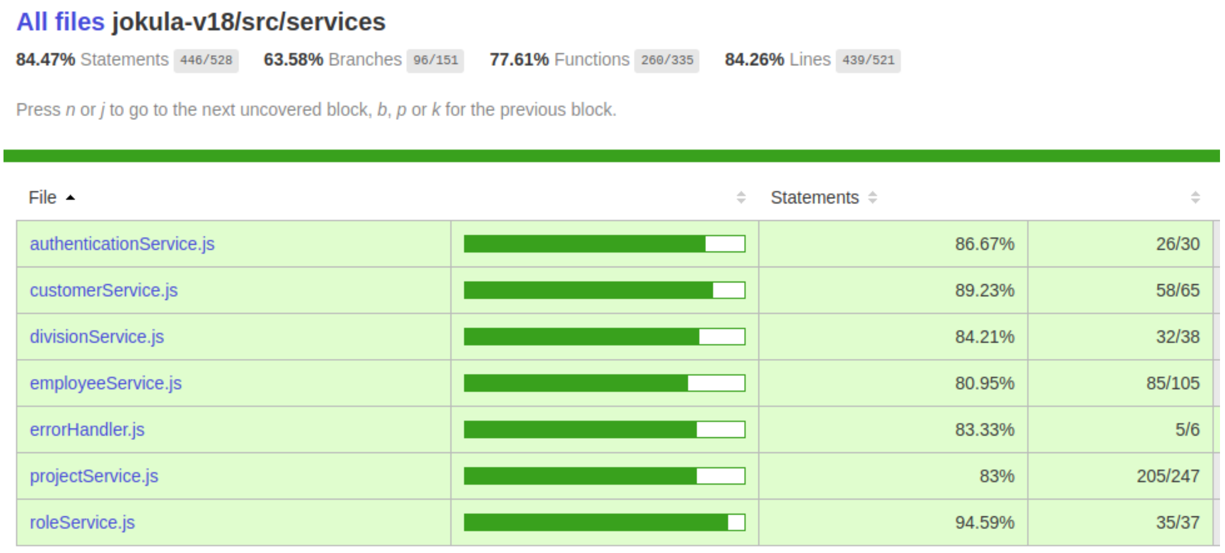
All files jokula-v18/src/routes

87.33% Statements 317/363 100% Branches 2/2 80.39% Functions 164/204 87.33% Lines 317/363

Press *n* or *j* to go to the next uncovered block, *b*, *p* or *k* for the previous block.

File		Statements	
authenticationRoute.js		90%	18/20
customerRoute.js		94.34%	50/53
divisionsRoute.js		93.94%	31/33
employeesRoute.js		84.15%	69/82
index.js		83.33%	5/6
projectRoute.js		82.48%	113/137
rolesRoute.js		96.88%	31/32

Figure 8.1: Shows code coverage for each route.



All files jokula-v18/src/services

84.47% Statements 446/528 63.58% Branches 96/151 77.61% Functions 260/335 84.26% Lines 439/521

Press *n* or *j* to go to the next uncovered block, *b*, *p* or *k* for the previous block.

File		Statements	
authenticationService.js		86.67%	26/30
customerService.js		89.23%	58/65
divisionService.js		84.21%	32/38
employeeService.js		80.95%	85/105
errorHandler.js		83.33%	5/6
projectService.js		83%	205/247
roleService.js		94.59%	35/37

Figure 8.2: Shows code coverage for each service.

8.3 User testing

We held several informal user testing sessions with the staff of //JÖKULÁ. In these sessions the user was asked to navigate the system and imitate the normal procedure the user would go through in the Excel version of the system they currently use at //JÖKULÁ. While this was happening the user was asked to speak aloud any thoughts that came to mind and one of the team members was writing down everything that was said and done in the session.

We also tested the system regularly ourselves by simply writing down tasks to complete, then solving the tasks and pretending we were users of the system with no knowledge of it. This might seem unnecessary or inefficient but we thought it was really helpful as a guideline when we weren't sure about some feature and wanted to make decisions fast without having to schedule a session with the //JÖKULÁ staff.

Finally we performed some informal user tests on some of our relatives and friends with the same method as we did with the //JÖKULÁ staff.

The results of all these tests were quite beneficial to us and became valuable in designing and implementing key features of the system. During one of the user tests with the staff at //JÖKULÁ we got some pointers from the project manager about how the project manager home dashboard should look. He pointed out that we would like to be able to see each employee on his dashboard and also be able to make tasks visible for employees from there. This was something that we had not thought of ourselves and was a good addition to our system. The employees at //JÖKULÁ also suggested that it would be great be able to see which users are online on the employee sidebar. This is something that we had actually thought of ourselves but thought that this was not a priority.

9 Post mortem

In this section we will mention what went well and what have learned during our time working on the project as well as mentioning what could have gone better.

We feel like we all learned a lot working on the project. We got experience working with Vue.js, Node.js and Bookshelf.js as well as developing a front end system that communicates with our server.

We had all learned about the scrum methodology before but most of us had never really used scrum on any type of real project before. We all agree that working using Scrum worked out extremely well. Scrum kept the chaos in check as well as keeping the team focused on a limited number of tasks at once. The team adapted well to these new technologies and we were all able to work independently pretty early on, but every team member was always ready to help when another ran into trouble.

When we started the project no member of the team had any experience working with the technology stack that //JÖKULÁ had assigned us. We had to spend a great amount of time to get familiar with technologies like Vue.js, Knex and Bookshelf.js. Planning the sprints went well but sometimes we underestimated the user stories in our sprints, those situations were mentioned on our risk analysis report. Occasionally we started working on the project without having completely written down the sprint backlog for that week, that was never a serious issue but we realize that is bad practice. We were all extremely busy during this semester and most of our school courses required us to spend a lot of time working on assignments and thereby neglecting the final project. We were still able to meet every week and we made up for time lost during the last 4 weeks of the semester.

It is important to mention that we did not have a clear vision for the system when we had started to work on the project. We had to start by implementing back end functions for the system before starting working on front end development, during this time we had not given the front end much thought, but as soon as we went full force working on the front end we realized how the system would operate from a user perspective. Lastly we will mention that one of our A requirements was to implement a salesman role for the system. We did include that role in the system but the salesman has no real function. We would have liked to have registered the salesman role as a B-requirement in the backlog initially, its functionality relies heavily on customers being active in the system, which in itself was a B-requirement. We also were not able to implement continuous integration, as it was always in our sideview and we always had more important functions to implement so it was always pushed to the side as “something we will do later”, we ended up trying to use CircleCI as continuous integration and got some help from the CTO of //JÖKULÁ but we were unsuccessful in getting it working successfully. When we were nearing the end we also noticed that one other A requirement story is not implemented, that is, to be able to filter tasks and projects. This is something that was overlooked by us all. We did however implement a sorting system for employees, customers and divisions.

Final words

In this report we introduced our project “Cats”. We covered what the systems features are as well as what technologies we used to be able to implement the system. We mentioned that we built the system from the ground up, so we think that it is important to mention what technologies we used. In the report we also describe what work methodology we used and how we used that to plan our work. We also dedicate a chapter in the report to show how the system works, step by step, as different user and try to explain what actions the user can take on each page. The report also covers a “post mortem” for the whole projects, this sections mentions what went well when working on the project and also what should have been done better or differently.

We all learned a great deal while implementing the system and we think that it is great that the school offers students to step outside the comfort zone of a classroom, and learn while working in a professional environment.

We would like to thank //JÖKULÁ for all the help and the cooperation that they provided and for always being ready to answer any questions that we had as soon as they came up. We would also like to thank our instructor Birgir for meeting with us regularly and providing really useful feedback on our project.

Appendix 1: Sprint Details

This report details the progress of our project and what we did in Sprint Zero and Sprint One.

Sprint Zero

In this sprint we laid the groundwork for the project. We were introduced to the project by //JÖKULÁ and had several meetings with them discussing the project. In collaboration with //JÖKULÁ we created and estimated the user stories for the project as well as setting up the environment for each team member so they would be ready to start working on the project. We received help setting up a “template” for the project from a programmer working at //JÖKULÁ. We spent some time getting familiar with the technology stack that we will be using to implement the project.
We did this by experimenting each on our own and trying to understand the frameworks and programming languages that are used.

	Andri	**Sigurður**	**Skúli**	**Smári**	**Total**
Hours	21	34	42	31	128

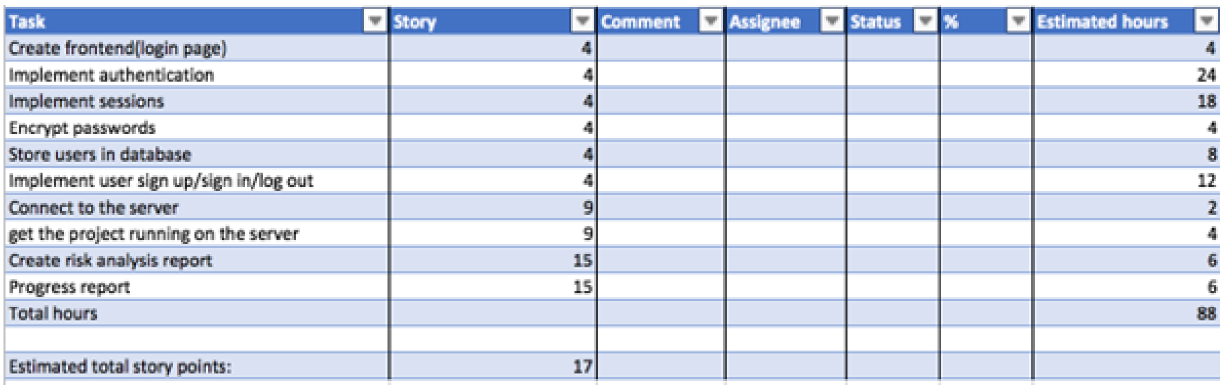
Table A1.1: Sprint zero. Hours worked in sprint

Sprint One

Tasks

We assigned user stories to a sprint backlog as well as creating tasks for each story. The tasks were as follows:

Total estimated hours: 88



Task	Story	Comment	Assignee	Status	%	Estimated hours
Create frontend(login page)	4					4
Implement authentication	4					24
Implement sessions	4					18
Encrypt passwords	4					4
Store users in database	4					8
Implement user sign up/sign in/log out	4					12
Connect to the server	9					2
get the project running on the server	9					4
Create risk analysis report	15					6
Progress report	15					6
Total hours						88
Estimated total story points:	17					

Figure A1.1: Sprint one Tasks

Since this was our first actual sprint we decided to take some time at the beginning of the sprint to plan the sprint properly and create templates for sprint backlogs and burndown charts that we will use in the coming sprints. Because of this and it being the first sprint the tasks we chose only add up to 88 hours.

Burndown Chart

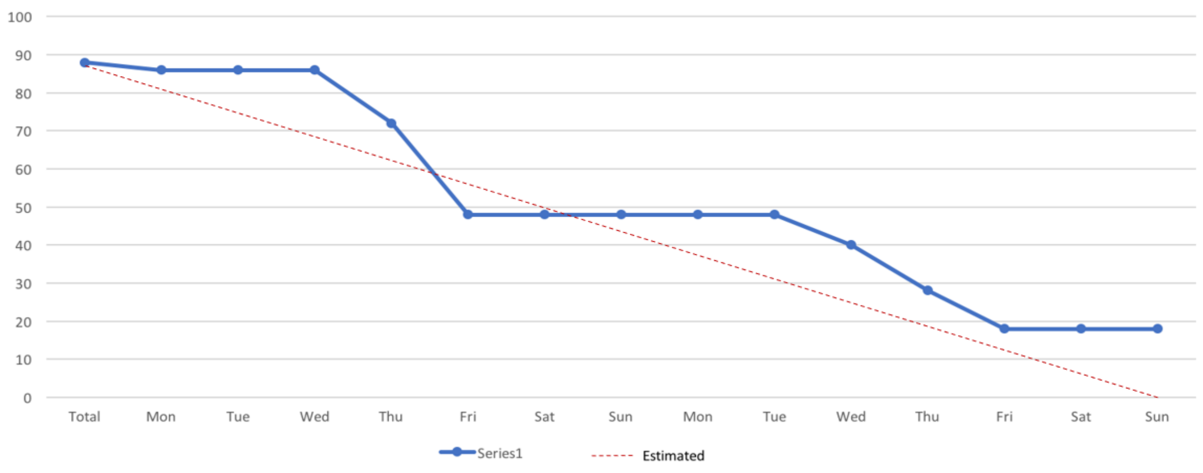


Figure A1.2: Sprint one Burndown

As we can see on our burndown chart, the progress line does not finish at 0. This is because the hand in is two business days before a status meeting and our sprint has three days left.

	Andri	Sigurður	Skúli	Smári	Total
Hours	37	32	16	29	114

Table A1.2: Sprint one. Hours worked in sprint.

Retrospective

These were the notes that we took from the retrospective meeting for sprint one.

Negative

- Reports felt rushed and were completed last minute. We should have reserved more of our time writing our reports well at the cost of programming.
- Tasks were too big (live stories). We picked stories for our backlog that could not be finished completely until a later time. An example of such a task is “creating database tables”.
- We would have wanted to break up the large stories into smaller tasks so that we could give a precise estimate of the time it takes to complete them.
- We spent too much time working of tasks that were not registered in the sprint backlog, should have picked stories on the backlog instead.
- Daily standups were not held every day. We should have followed the Scrum methodology more precisely.
- We should work on the project on a “catch-up” day stay and stay longer Thursdays since Fridays we often leave early on fridays and don't work our required 8 hours.

Positive

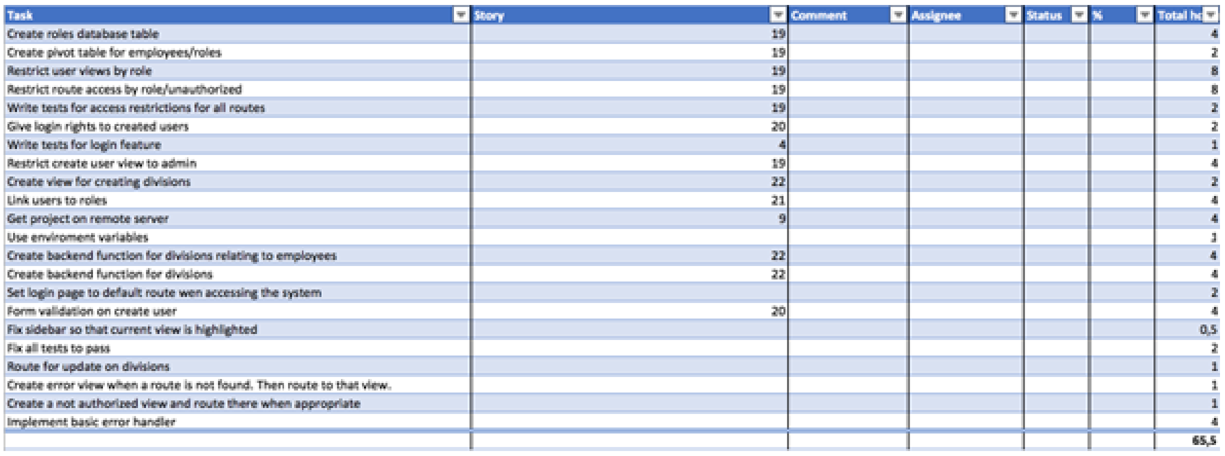
- Starting tasks was easy.
- We are very happy with the sprint document that we made.
- The extra time we spent on reports will save us much time in the future.

Sprint Two

Tasks

The tasks for sprint two were the following:

Total estimated hours: 65.5



Task	Story	Comment	Assignee	Status	%	Total h
Create roles database table	19					4
Create pivot table for employees/roles	19					2
Restrict user views by role	19					8
Restrict route access by role/unauthorized	19					8
Write tests for access restrictions for all routes	19					2
Give login rights to created users	20					2
Write tests for login feature	4					1
Restrict create user view to admin	19					4
Create view for creating divisions	22					2
Link users to roles	21					4
Get project on remote server	9					4
Use enviroment variables						1
Create backend function for divisions relating to employees	22					4
Create backend function for divisions	22					4
Set login page to default route wen accessing the system						2
Form validation on create user	20					4
Fix sidebar so that current view is highlighted						0,5
Fix all tests to pass						2
Route for update on divisions						1
Create error view when a route is not found. Then route to that view.						1
Create a not authorized view and route there when appropriate						1
Implement basic error handler						4
						65,5

Figure A1.3: Sprint two Tasks

Some tasks were not completely finished at the end of the sprint, most notably the task of getting the system to a remote server. That proved to be harder than we expected.

Burndown Chart

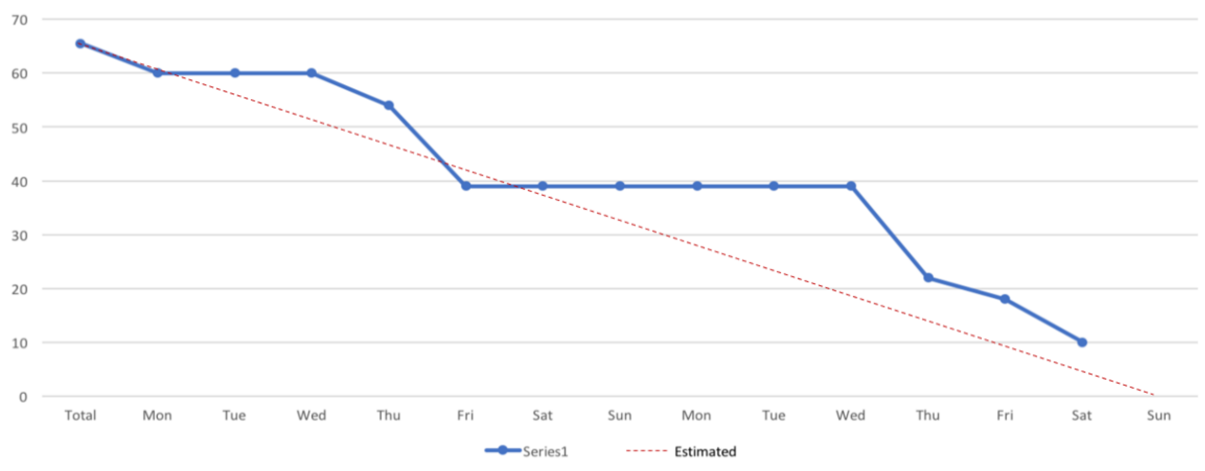


Figure A1.4: Sprint two Burndown

The hours left in this sprint were estimated for the task of setting the project up on a remote server. Other than that the tasks were completely finished.

	Andri	Sigurður	Skúli	Smári	Total
Total	30	13.5	16.75	23	83.25

Table A1.3: Sprint two. Hours worked in sprint.

Retrospective

These were the notes that we got from the retrospective meeting for sprint two.

Negative

- Some tasks required other tasks to be done that were not on the sprint backlog.
- Managing the sprint backlog was not as much of a priority as it should be, some tasks were updated some time after they were finished which reduced accuracy
- Standups not daily, sometimes missed.
- Frequent switching between tasks without finishing them.
- Important tasks were assigned few story points but took a long time.

Positive

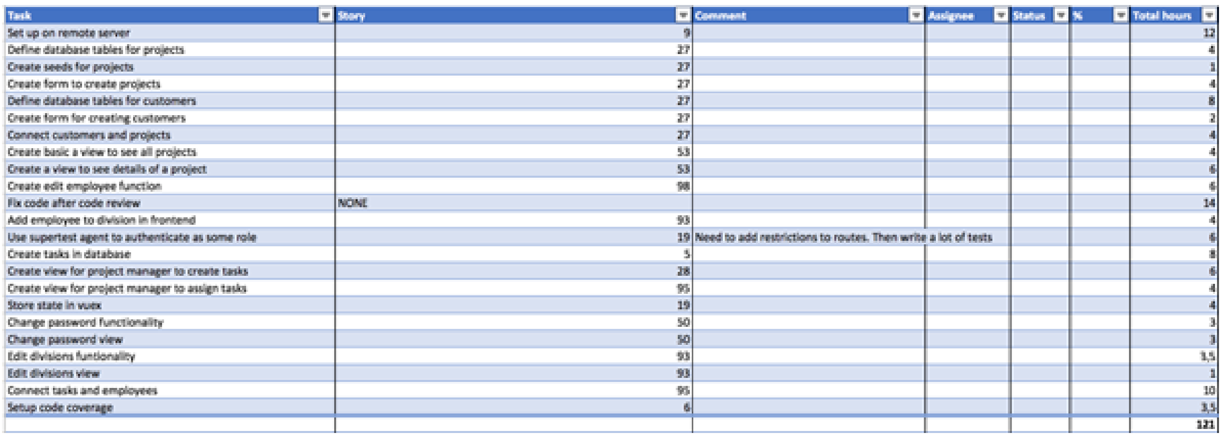
- This sprint was better than the last.
- Managed to finish almost everything we wanted.
- Estimated the stories accurately and the tasks were well defined.
- Followed the sprint backlog closely.

Sprint Three

Tasks

The tasks for sprint three were the following:

Total estimated hours: 121



Task	Story	Comment	Assignee	Status	%	Total hours
Set up on remote server	9					12
Define database tables for projects	27					4
Create seeds for projects	27					1
Create form to create projects	27					4
Define database tables for customers	27					8
Create form for creating customers	27					2
Connect customers and projects	27					4
Create basic a view to see all projects	53					4
Create a view to see details of a project	53					6
Create edit employee function	98					6
Fix code after code review	NONE					14
Add employee to division in frontend	93					4
Use supertest agent to authenticate as some role	19	Need to add restrictions to routes. Then write a lot of tests				6
Create tasks in database	5					8
Create view for project manager to create tasks	28					6
Create view for project manager to assign tasks	95					4
Store state in vuex	19					4
Change password functionality	50					3
Change password view	50					3
Edit divisions funtionality	93					3,5
Edit divisions view	93					1
Connect tasks and employees	95					10
Setup code coverage	6					3,5
						121

Figure A1.5: Sprint three Tasks

Burndown Chart

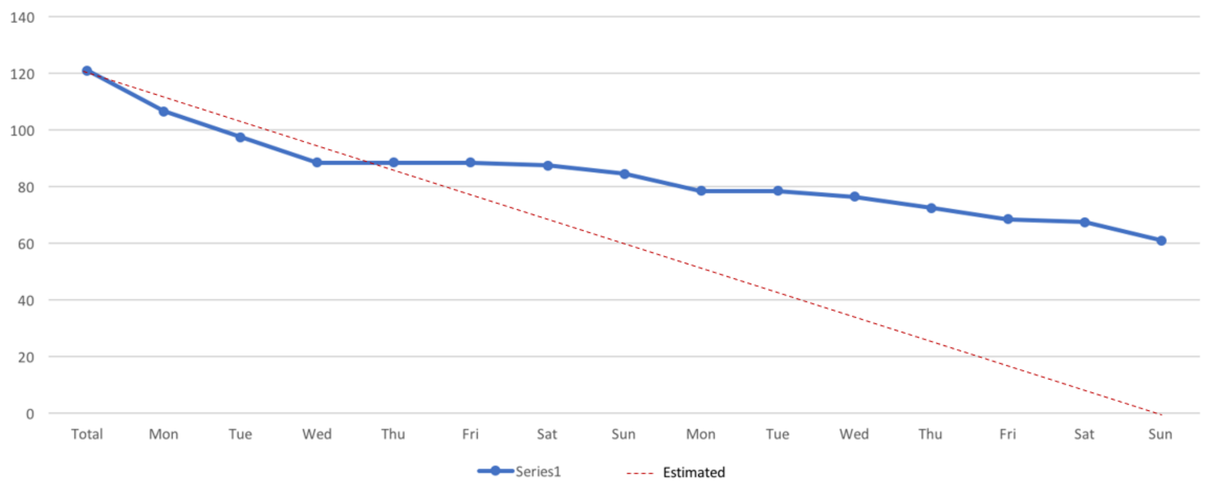


Figure A1.6: Sprint three Burndown

This sprint didn’t go exactly as planned. We lost a whole day of work and all had deadlines in other courses on top of that. We also should have picked tasks with a higher priority, for instance implementing the changing password functionality was something we has as a task in the sprint and then decided to put on the backburner as it is a feature that is not critical at this moment in time. We completed most back end and some front end tasks concerning projects and divisions. Leaving table connections and front end views for the next sprint and the changing password functionality for a future sprint.

	Andri	Sigurður	Skúli	Smári	Total
Hours	20	9	46	29	104

Table A1.4: Sprint three. Hours worked in sprint.

Retrospective

Negative

- Full day was lost
- Making up for it was difficult because we were all really busy
- Updating the sprint backlog was somewhat neglected
- The sprint backlog was not fully finished when we started working in the sprint
- Functions in the back end are often written without writing API docs and tests for them
- The sprint wasn’t planned enough and started badly
- There were many distractions and a lot to do outside the project

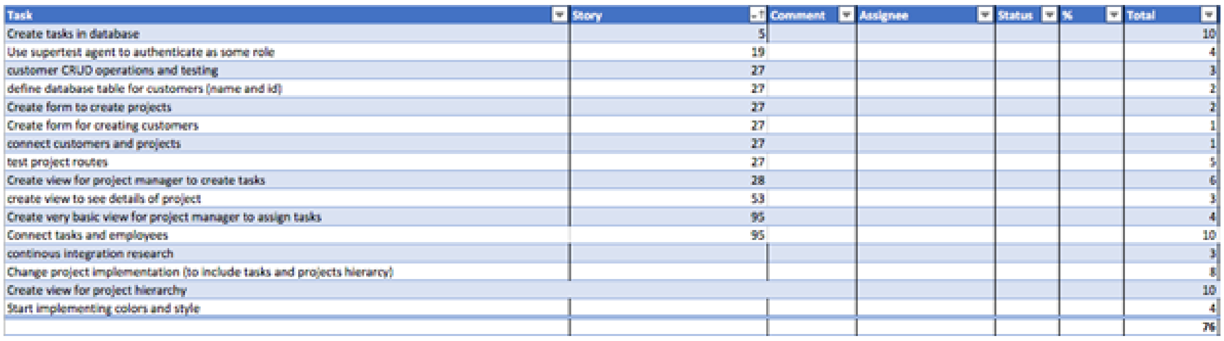
Positive

- Everyone is able to work independently
- We have gained knowledge using the frameworks

Sprint Four

Tasks

Total estimated hours: 76



Task	Story	Comment	Assignee	Status	%	Total
Create tasks in database	5					10
Use supertest agent to authenticate as some role	19					4
customer CRUD operations and testing	27					3
define database table for customers (name and id)	27					2
Create form to create projects	27					2
Create form for creating customers	27					1
connect customers and projects	27					1
test project routes	27					5
Create view for project manager to create tasks	28					6
create view to see details of project	53					3
Create very basic view for project manager to assign tasks	95					4
Connect tasks and employees	95					10
continous integration research						3
Change project implementation (to include tasks and projects hierarcy)						8
Create view for project hierarchy						10
Start implementing colors and style						4
						76

Figure A1.7: Sprint four Tasks

Burndown Chart

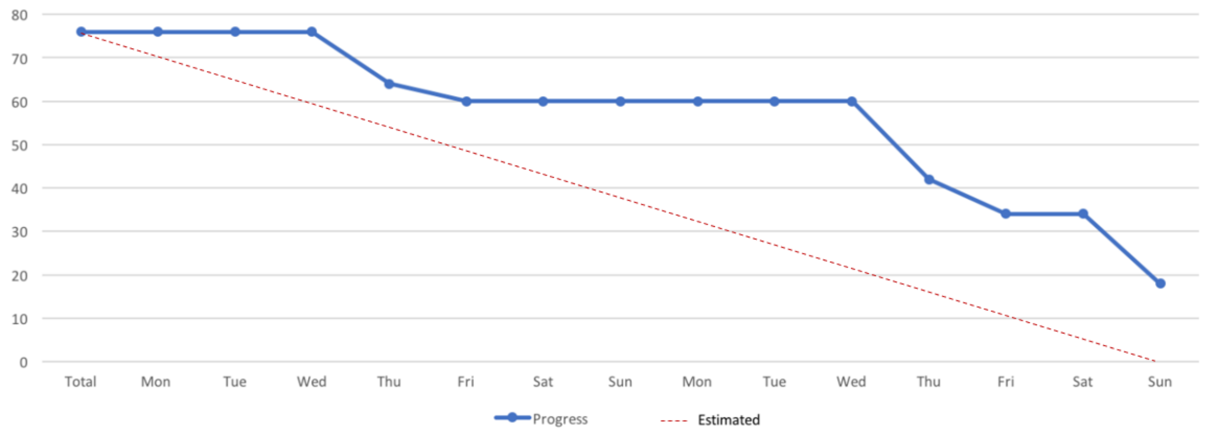


Figure A1.8: Sprint four Burndown

The Easter vacation was during this sprint. We completed most of the tasks, leaving behind creating the functionality behind the project hierarchy we had envisioned in the system. We decided to move this into the next sprint since we wanted to discuss this further when we were all present.

	Andri	Sigurður	Skúli	Smári	Total
Hours	11	22	38.5	24	95.5

Table A1.5: Sprint four. Hours worked in sprint.

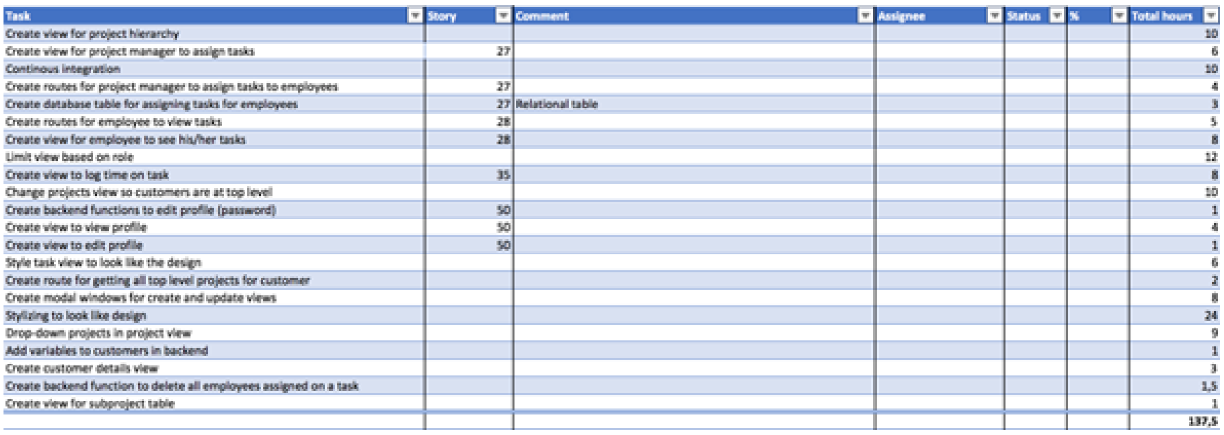
Retrospective

- What we did went well, however, the members of the team were all absent since there was the Easter break in school.
- The sprint was three weeks long but we didn’t work on the project as much as we would have liked.
- The team decided to split in two: front end and back end. We thought that this went well and we want to continue doing this.

Sprint Five

Tasks

Total estimated hours: 137.5



Task	Story	Comment	Assignee	Status	%	Total hours
Create view for project hierarchy						10
Create view for project manager to assign tasks	27					6
Continous integration						10
Create routes for project manager to assign tasks to employees	27					4
Create database table for assigning tasks for employees	27	Relational table				3
Create routes for employee to view tasks	28					5
Create view for employee to see his/her tasks	28					8
Limit view based on role						12
Create view to log time on task	35					8
Change projects view so customers are at top level						10
Create backend functions to edit profile (password)	50					1
Create view to view profile	50					4
Create view to edit profile	50					1
Style task view to look like the design						6
Create route for getting all top level projects for customer						2
Create modal windows for create and update views						8
Stylizing to look like design						24
Drop-down projects in project view						9
Add variables to customers in backend						1
Create customer details view						3
Create backend function to delete all employees assigned on a task						1,5
Create view for subproject table						1
						137,5

Figure A1.9: Sprint five Tasks

Burndown chart

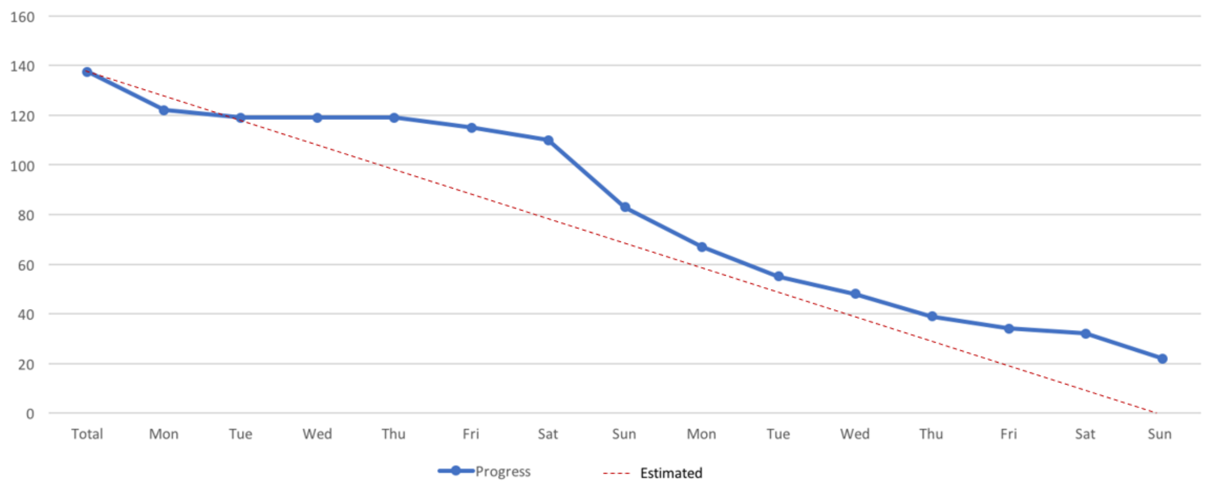


Figure A1.10: Sprint five Burndown

This sprint went well. The hours left over are mostly related to the Continuous Integration task. We have spoken to the CTO at //JÖKULÁ and he is looking into what CI he prefers we use and has offered to help us integrate it. Other than that only a few hours were left on views.

	Andri	Sigurður	Skúli	Smári	Total
Hours	58	41.5	46	48.5	194

Table A1.6: Sprint five. Hours worked in sprint.

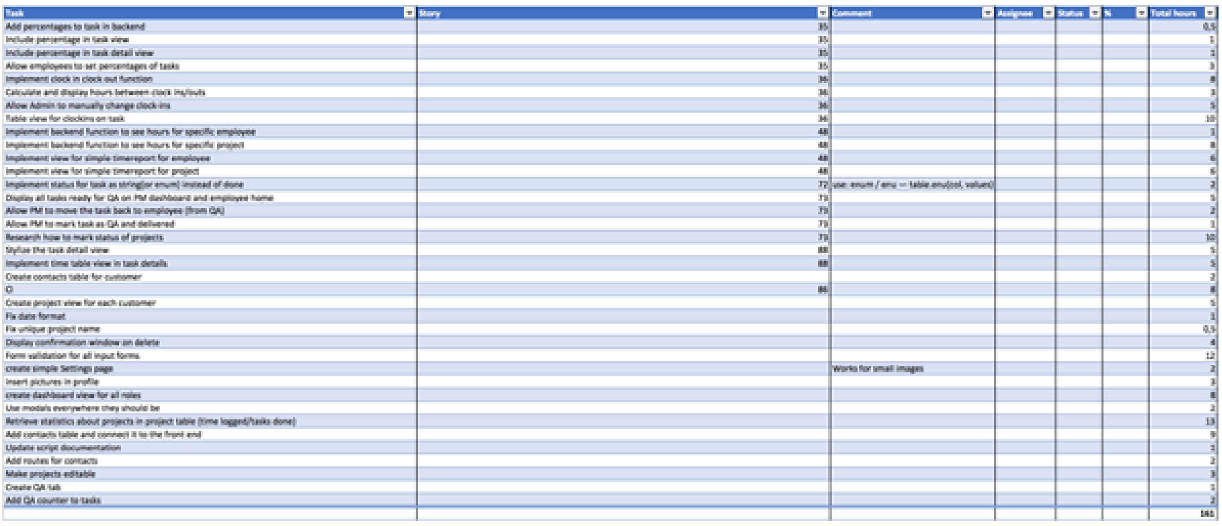
Retrospective

- We decided to keep the team in two groups, one for front end and one for back end. However this was not ideal because there were so many tasks left in the front end but few in the back end so all of us switched to the front end team.
- Replicating the design went great.
- We had exams but despite that we managed to much more than in previous sprints.
- We did not set up CI in this sprint but we want to set it up in the next sprint.

Sprint Six

Tasks

Total estimated hours: 161



Task	Story	Comment	Assignee	Status	%	Total hours
Add percentages to task in backend	35					0,5
Include percentage in task view	35					1
Include percentage in task detail view	35					1
Allow employees to set percentages of tasks	35					3
Implement clock in clock out function	36					8
Calculate and display hours between clock ins/outs	36					1
Allow Admin to manually change clock-ins	36					5
Table view for clockins on task	36					10
Implement backend function to see hours for specific employee	48					1
Implement backend function to see hours for specific project	48					8
Implement view for simple timereport for employee	48					6
Implement view for simple timereport for project	48					6
Implement status for task as string(or enum) instead of done	72	use enum / enu = table.enu(col, values)				2
Display all tasks ready for QA on PM dashboard and employee home	73					5
Allow PM to move the task back to employee (from QA)	73					2
Allow PM to mark task as QA and delivered	73					1
Research how to mark status of projects	73					10
Stylize the task detail view	88					5
Implement time table view in task details	88					5
Create contacts table for customer						2
CI	86					8
Create project view for each customer						5
Fix date format						1
Fix unique project name						0,5
Display confirmation window on delete						4
Form validation for all input forms						12
create simple Settings page		Works for small images				2
Insert pictures in profile						3
create dashboard view for all roles						8
Use modals everywhere they should be						2
Retrieve statistics about projects in project table (time logged/tasks done)						13
Add contacts table and connect it to the front end						9
Update script documentation						1
Add routes for contacts						2
Make projects editable						3
Create QA tab						1
Add QA counter to tasks						2
						161

Figure A1.11: Sprint six Tasks

Burndown chart

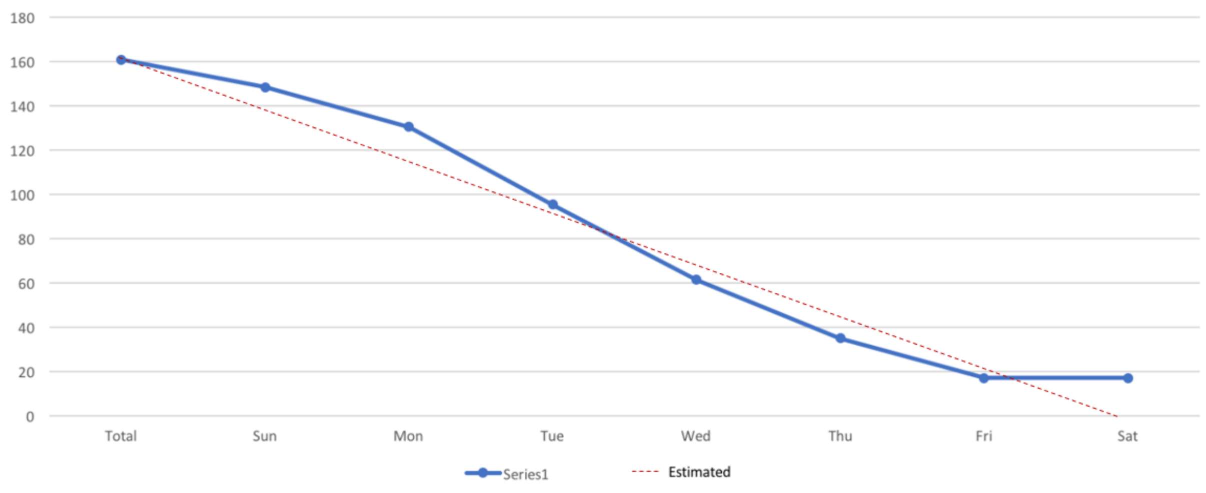


Figure A1.12: Sprint Six Burndown

We were happy with the progress in this sprint. We are all at //JÖKULÁ working all day and the progress is good. The system is taking the shape of the original vision both cosmetically and functionally. The only tasks that we did not complete were CI and Form validation thought they are both on their way. //JÖKULÁ chose CircleCi but we have been unsuccessful at integrating it with their help. Questions arose about bringing in a CircleCi expert but we decided to put that on hold to better focus on other tasks. We used part of the final days of the sprint to walk through the system as different roles verifying the system and looking for features that need improving on.

	Andri	Sigurður	Skúli	Smári	Total
Hours	48	57.5	54	55	214.5

Table A1.7: Sprint six. Hours worked in sprint.

Retrospective

Negative

- We were unable to get our CI up and running
- Since we have all mostly been developing front end, documenting in the back end and unit testing has been neglected
- Some of the functionality that should have been implemented in the back end was implemented in the front end

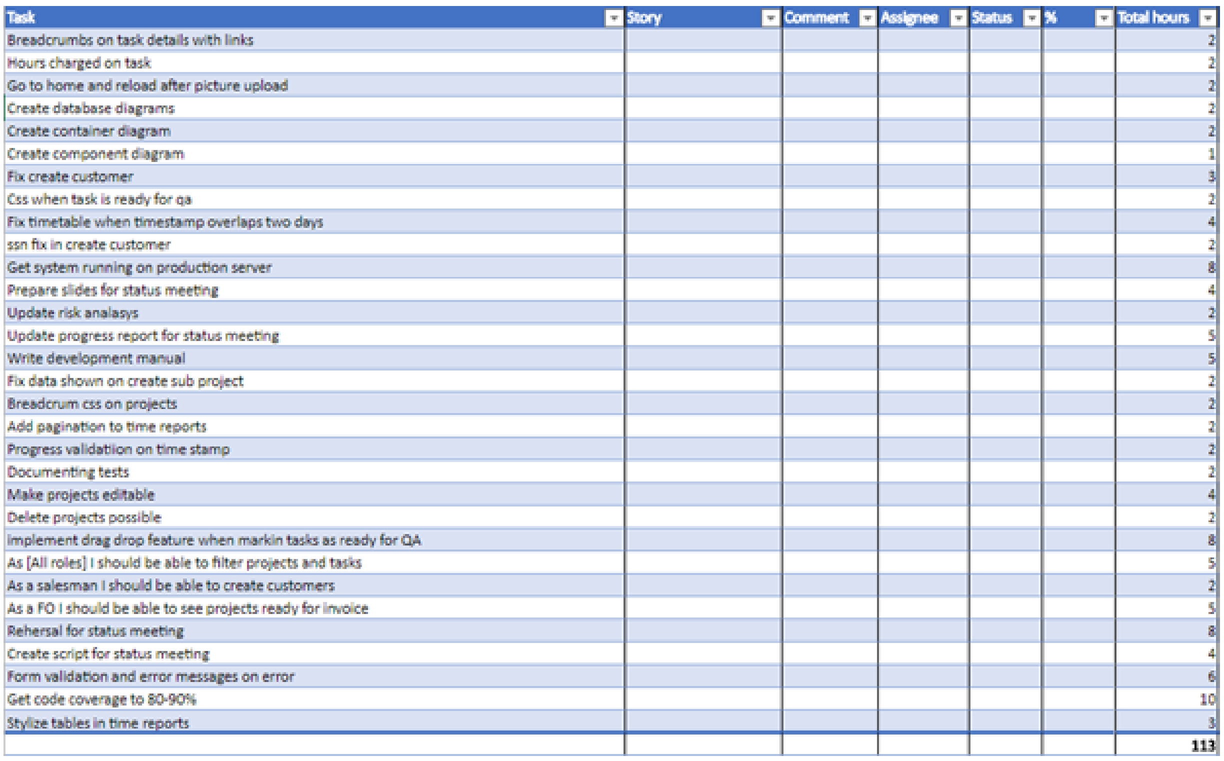
Positive

- The sprint itself went well
- We were able to finish the tasks we set ourselves for the sprint early
- Because of that we were able to find time to test the system acting as different roles to find possible bugs
- The system is taking shape both cosmetically and functionally
- We have all been able to apply all of our time to the Project
- Making communication and pair programming easy
- We have been able to keep to our work schedule

Sprint Seven

Tasks

Total estimated hours: 113



Task	Story	Comment	Assignee	Status	%	Total hours
Breadcrumbs on task details with links						2
Hours charged on task						2
Go to home and reload after picture upload						2
Create database diagrams						2
Create container diagram						2
Create component diagram						1
Fix create customer						3
Css when task is ready for qa						2
Fix timetable when timestamp overlaps two days						4
ssn fix in create customer						2
Get system running on production server						8
Prepare slides for status meeting						4
Update risk analasys						2
Update progress report for status meeting						5
Write development manual						5
Fix data shown on create sub project						2
Breadcrum css on projects						2
Add pagination to time reports						2
Progress validatiion on time stamp						2
Documenting tests						2
Make projects editable						4
Delete projects possible						2
Implement drag drop feature when markin tasks as ready for QA						8
As [All roles] I should be able to filter projects and tasks						5
As a salesman I should be able to create customers						2
As a FO I should be able to see projects ready for invoice						5
Rehersal for status meeting						8
Create script for status meeting						4
Form validation and error messages on error						6
Get code coverage to 80-90%						10
Stylize tables in time reports						3
						113

Figure A1.13: Sprint seven Tasks

Burndown chart

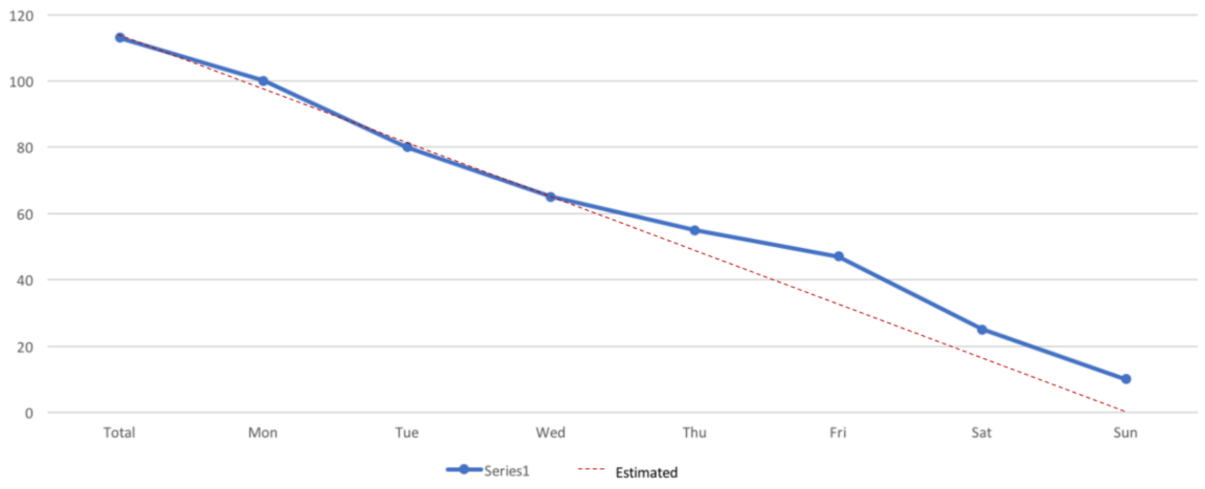


Figure A1.14: Sprint seven Burndown

In this sprint we had our final status meeting. The sprint was centered around it. We focused on creating diagrams, updating reports and creating a story for the meeting. We also made sure the system was up to spec for the meeting. We tried to go through the system to try and expose any bugs. In the days after the meeting we finished any outstanding tasks, the only ones we didn’t finish were a drag and drop feature (which was pretty optimistic but we wanted to give it a go so we had it in the sprint) and filtering for projects and tasks.

	Andri	**Sigurður**	**Skúli**	**Smári**	**Total**
Hours	48.5	55	70.75	58	232.25

Table A1.8: Sprint seven. Hours worked in sprint.

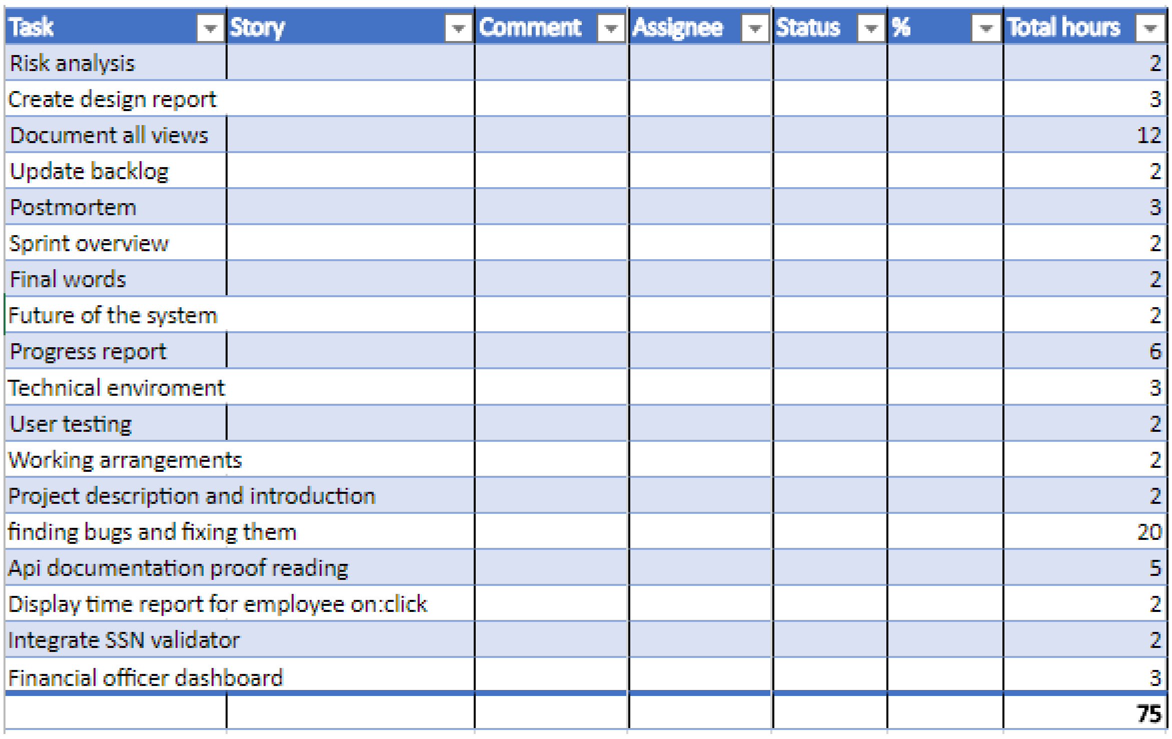
Retrospective

- Preparing for the status meeting went well
- The status meeting itself also went well.
- We had time to fix bugs, which went well.
- We had already finished many tasks before planning the sprint.
- All A tasks that we want to accomplish are done as well as a lot of B tasks

Sprint Eight

Tasks

Total estimated hours: 75



Task	Story	Comment	Assignee	Status	%	Total hours
Risk analysis						2
Create design report						3
Document all views						12
Update backlog						2
Postmortem						3
Sprint overview						2
Final words						2
Future of the system						2
Progress report						6
Technical enviroment						3
User testing						2
Working arrangements						2
Project description and introduction						2
finding bugs and fixing them						20
Api documentation proof reading						5
Display time report for employee on:click						2
Integrate SSN validator						2
Financial officer dashboard						3
						75

Figure A1.15: Sprint eight Tasks

Burndown chart

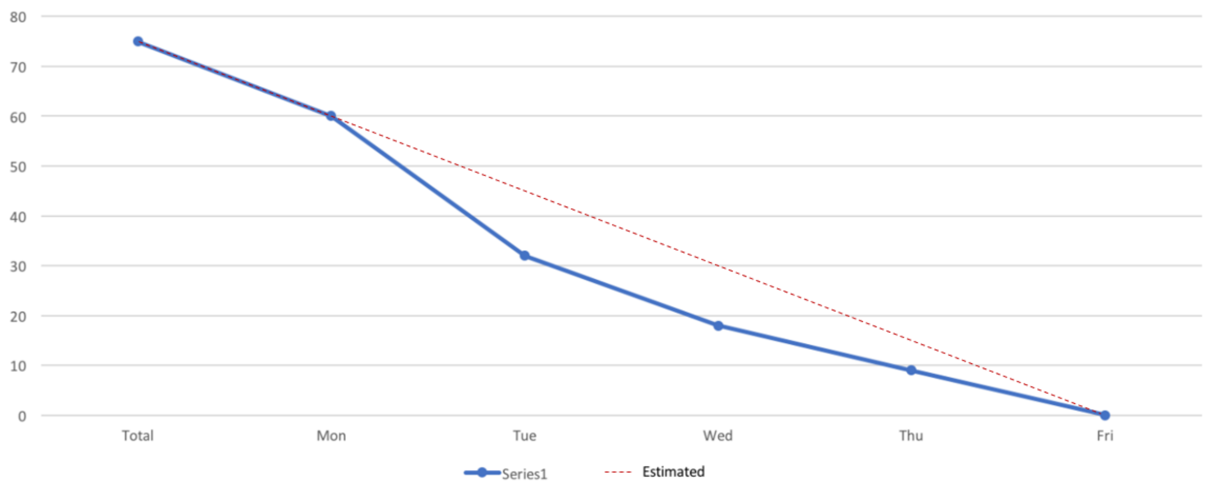


Figure A1.16: Sprint eight Burndown

Most of the sprint was used to finish our final report and for some last minute bug fixes of the system. The sprint was only 5 days long since we had to hand in our final project on Friday.

Working on the report went well and we think we were able to finish all of the chapters in the report that we set out to do. We were also able to fix a few minor things like bugs in the CSS as well as validation on forms. We added some nice-to-have features like shortcuts to each employees time report from the project manager view and made the financial officers dashboard look better. All in all this sprint went well.

	Andri	Sigurður	Skúli	Smári	Total
Hours	42	42	45	42	171

Table A1.9: Sprint eight. Hours worked in sprint.

Retrospective

- Would have liked to spend more time on bug fixing however making the report was time consuming.
- Finishing the report was not as fun as programming.
- We didn’t implement any new features in this sprint.
- Small things that needed fixing were fixed in this sprint.
- Would have liked to finish the final presentation as well in this sprint instead of doing it over the weekend.
- Writing the report went well.

Appendix 2: Risk Analysis

We calculate the risk level of each risk by multiplying together the probability and severity of it, each being a factor from 1 to 5. We estimated these factors together as a team, each team member giving input to be as accurate as possible. We then decided what would be our precautions for each risk as well as the responsible party.
If we encounter some of these risks later we will fill in the Fixed, When and How columns.

Risk level = Probability * Severity = (1 to 5) * (1 to 5)

Nr.	Risk	Severity	Likelihood	Risk level	Responsible Party	Precaution
1	Underestimating user stories	4	5	20	Everyone	Learn from past sprints, and reevaluate stories accordingly. This could be done at the start of each sprint
2	Not all team members have the same understanding of the system	4	4	16	Everyone	Make sure that the team is on the same page. The team must strive to help each other. Each team member must not be afraid to ask
3	Other courses take too much time from the project	4	4	16	Everyone	Use the weekends for either the project or the other courses /
4	The team does not	3	5	15	Everyone	Stay focused on sprint tasks, Scrum master uses daily standup to enforce

	stick to sprint tasks.					this.
5	Team members encounter pitfalls	5	3	15	Everyone	Get help from someone that is familiar with the problem.Try another approach to the implementation
6	The end system might not be what the product owner expected.	5	2	10	Everyone	Include the product owner as much as possible. Get feedback often to realize any potential design mistakes.
7	The system will not be ready by deadline (All user stories)	2	5	10	Everyone	Stick to the plan. Get help when encountering pitfalls. Prioritize crucial features of the system over secondary features
8	Difficulty adapting to a new technology environment	5	2	10	Everyone	Each team member should ask for help when necessary.
9	Team cannot meet	3	3	10	Everyone	Try and reschedule to another day or add another work day to a following week.

10	Broken code pushed to github	3	3	10	Everyone	Fix the code and push again, let the other team members know immediately
11	Sprint task are assigned unevenly between team members	4	2	8	Scrum master	Letting each other know when assigning a task. Assigning one task at a time.
12	Communication difficulties between team members	4	2	8	Everyone	Keeping our Daily standup meetings.
13	Teammember is unavailable	1	5	5	Team member	Team member works overtime to make up for lost time

Table A2.1: Risk Analysis table

Risk Log

In this log we log all the risks we have encountered, when they were encountered and how they were dealt with.

Risk Nr.	Comment	When	Solution
3		Sprint 2	Took an extra day to catch up, it put us back on schedule
3		Sprint 3	Happened again this sprint, we decided to work overtime the rest of the semester to make sure we would

			finish everything
5	Encountered a pitfall when setting up the project on a remote server	Sprint 3	We got some help from the staff at //JÖKULÁ in setting up on a server but didn’t manage to finish it in this sprint. We put the matter on hold.
13	There was an easter break during sprint 4 and none of us were at the same place	Sprint 4	We chose tasks that were easy to implement on our own and postponed tasks that would require help or opinions from other members
1	We underestimated the complexity of the front end and appearance of the application	Sprint 4 and 5	We split the team into two groups, one group would purely focus on the front end. Later all team members switched to the front end group to finish it up which worked nicely.

Table A2.2: Risk log

Appendix 3: Product Backlog

Backlog nr.	Division	Priority	Story Points	Sprint nr.	Category	Story	Comment	Status	%
1	Project	A	1	0	Meeting	Meeting with //JÖKULÁ about the project		Done	100
2	Design	A	3	0	Design	As a developer I want to me able to see an UML diagram for the database	Database design	Done	100
3	Project	A	1	0	Meeting	Introduction to project backlog		Done	100
4	Programming	A	13		Project	As a user I should be able to login	Authentication	Done	100
5	Programming	A	13	5	Server	Database tables	Implement database in code	Done	100
7	Programming	A	5	0	Project	Setup environment for development	Everything related to the setup of the environment for development of the project	Done	100
9	Programming	A	3	2	Project	Setup remote server	Setup a remote server to host front-end and back-end of system	Done	100
15	Project	A	6		Documentation	Create risk analysis report/Progress report		Done	100
16	Project	A	3	2	Documentation	Verkskipulag/Verklýsing		Done	100
19	Programming	A	13		Project	As a user I should be able to have different roles with different permissions		Done	100
20	Programming	A	2		Project	As an admin I should be able to create users		Done	100
21	Programming	A	5		Project	As an admin I should be able to assign roles to users		Done	100
22	Programming	A	2		Project	As an admin I should be able to create divisions		Done	100

23	Programming	A	5		Project	As a salesman I should be able to create offers			
27	Programming	A	20		Project	As a project manager I should be able to create project (or confirm pending projects)	Confirming pending projects put on hold		80
28	Programming	A	8		Project	As a project manager I should be able to assign task to employees	Within their divisions	Done	100
33	Programming	A	3		Project	As an employee I should be able to see tasks assigned to me	Filtered by division. We need to implement some kind of folder structure.	Done	100
35	Programming	A	3		Project	As an employee I should be able to mark task as done or % of done		Done	100
36	Programming	A	5		Project	As an employee I should be able to log time worked on each task		Done	100
48	Programming	A	5		Project	As [All roles] I should be able to retrieve time reports	Time reports for each employee, for each project etc.	Done	100
50	Programming	A	1		Project	As a user I should be able to change my password		Done	100
53	Programming	A	1		Project	As [All roles] I should be able to see all my projects		Done	100
54	Programming	A	8		Project	As [All roles] I should be able to clock in for each task	From home page and detailed page	Done	100
60	Programming	A	5		Project	As an admin should be able to see all users and sort them		Done	100
66	Programming	A	1		Project	As a salesman I should be able to see and change status of offers	Accepted, Pending, Rejected		
67	Programming	A	8		Project	As a project manager I should be able to add tasks to projects	Including description, estimated hours, deadline	Done	100
68	Programming	A	1		Project	As a project manager I should be able to change tasks		Done	100
70	Programming	A	2		Project	As a project manager I can edit projects		Done	100
72	Programming	A	1		Project	As a project manager I should be able to see projects (and tasks) ready for	A project should be sent to to QA when all	Done	100

						QA	tasks are 100% done		
73	Programming	A	1		Project	As a project manager I should be able to change the status of a project as "QA AND DELIVERED"	Need to decide what to do when done in QA	Done	100
74	Programming	A	1		Project	As a project manager I can set a project/tasks as "ready for invoice"		Done	100
76	Programming	A	3		Project	As [All roles] I should be able to filter projects and tasks			
79	Programming	A	1		Project	As a project manager I should be able to see assigned tasks for all employees		Done	100
86	Programming	A	2		Project	As an employee I should be able to see my tasks that are ready for QA		Done	100
87	Programming	A	8		Project	As an employee I should be able to drag and drop tasks to mark them as ready for QA	Needs drag and drop feature		80
88	Programming	A	8		Project	As an employee I should be able to see a more detailed view on a task	Description, hours logged, (attachments)	Done	100
92	Programming	A	3		Project	As [All roles] I should be able to see my available functions in a navbar at all times		Done	100
93	Programming	A	5		Project	As an HR I should be able to create, edit or delete divisions		Done	100
94	Programming	A	2		Project	As an HR I should have an overview of divisions	Project managers, phone numbers etc	Done	100
95	Programming	A	20		Project	As a PM I should be able to create a task hierarchy within a project.	This hierarchy should allow all combinations of folders and tasks	Done	100
96	Programming	A	3		Project	As a PM I should be able to create customers		Done	100
98	Programming	A	8		Project	As an HR I should be able to create, edit or delete Employees		Done	100
200	Programming	A	3		Project	As a salesman, FO, PM I should be able to see and update contacts for each customer		Done	100
c	Scripting	A	5		Project	Set up CI	Continuous integration.	Troub	50

							Builds. bundles, tests, deploys. The docker VM is unable to execute unit tests.	le	
x	Programming	A	2		Project	As a salesman I should be able to create customers		Done	100
24	Programming	B	1		Project	As a salesman I should be able to send offers to customer	Offers put on hold for the time being		
25	Programming	B	3		Project	As a salesman I should be able to forward accepted offers to the project manager	Have to implement salesman		
26	Programming	B	1		Project	As a project manager I should be able to see accepted offers (pending projects)	Front-end, server, routes and filtering data		
29	Programming	B	13		Project	As a project manager I should be able to see hours worked on a done tasks	See hours for each task by employee, day, etc (filtering)	Done	100
32	Programming	B	1		Project	As a user I should be able to change my password		Done	100
~~34~~	~~Programming~~	~~B~~	~~3~~		~~Project~~	~~As an employee I should be able to see open tasks~~	~~Tasks that have not been assigned to anyone in the project~~		
37	Programming	B	1		Project	As an HR I should be able to create users		Done	100
38	Programming	B	1		Project	As an HR I should be able to assign roles		Done	100
39	Programming	B	1		Project	As a FO I should be able to see projects ready for invoice	Partially implemented in the final project		50
42	Programming	B	3		Project	As a project manager I should be able to change hours back in time		Done	100
47	Programming	B	3		Project	As a [All roles] I should be able to see info about all projects	Status, etc.	Done	100
55	Programming	B	13		Project	As [All roles] I should be able to access a file system for each project			
58	Programming	B	2		Project	As [All roles] I should be able to see a list of employees		Done	100

61	Programming	B	8		Project	As a HR I should be able to see and change employment contract info	C: Upload the contract file itself. Possibly notifications when contracts expire, could pose serious complications		
62	Programming	B	2		Project	As a salesman I should be able to change offers	Change price, tasks, if the offer is monthly, discount, which customer etc.		
63	Programming	B	1		Project	As a salesman I should be able to copy offers			
75	Programming	B	1		Project	As an FO I can mark the project as "invoiced"		Done	100
78	Programming	B	3		Project	As [All roles] I should be able to see past and future tasks	Future tasks meaning assigned for another day		50
80	Programming	B	2		Project	As a customer I should be able to see an overview of my open projects, filtered by months			
81	Programming	B	5		Project	As a customer I should be able to see a timesheet of all done tasks in my projects			
83	Programming	B	2		Project	As [All roles] I should be able to input/edit information on customers	Except for employee, HR	Done	100
84	Programming	B	2		Project	As [All roles] I should be able to view information on customers		Done	100
89	Programming	B	8		Project	As an employee/PM I should be able to attach files to tasks			
97	Programming	B	2		Project	As a project manager i want to see how many times the same task has been sent into QA		Done	100
6	Programming	C	2		Project	As a developer I want to be able to see code coverage		Done	100
8	Programming	C	3		Project	As a developer I want to develop using CI	Setup continuous integration in development		50

10	Programming	C	2		Project	Create seeder files	Fills the database with fake data	Done	100
11	Project	C	3		Documentation	Create Hand-Off documentation		Done	100
12	Project	C	3		Documentation	Create development guide		Done	100
13	Project	C	3		Documentation	Create instruction manual		Done	100
14	Project	C	3		Documentation	Create burndown chart	Both for each sprint and the entire project	Done	100
17	Project	C	3		Documentation	Context Diagram		Done	100
18	Project	C	3		Documentation	Component diagram		Done	100
30	Programming	C	8		Project	As a project manager I should be able to validate hours on done tasks	Validate the hours worked on a task for a specific project	Done	100
31	Programming	C	8		Project	As a project manager I should be able to assess a task and either validate it or send it back to the employee		Done	100
40	Programming	C	5		Project	As a customer I should be able to see all approved hours for project			
41	Programming	C	3		Project	As a customer I should be able to make comments projects/ time logging			
43	Programming	C	2		Project	Employees and project managers should be able to comment on projects	Creates a conversation on each project	Done	100
49	Programming	C	1		Project	As [All roles except employee] I should be able to see conditions and agreements for offers	NEED INFO		
51	Programming	C	5			As a user I should be able to change my personal settings		Done	100
59	Programming	C	3		Project	As a project manager I should be able to see the skill set of each employee	Possibly visible for more user roles		

65	**Programming**	**C**	**3**		**Project**	**As a salesman I should be able to export offers to pdf**			
85	**Programming**	**C**	**3**		**Project**	**As a customer I want to be able to receive notifications on a project's status**			
90	**Programming**	**C**	**8**		**Project**	**As [All roles] I should see announcements on my home page**	**Possibly filtered by role,projects etc.**		
94	**Programming**	**C**	**5**		**Project**	**As a customer I should be able to sign up to get an account**			
97	**Programming**	**A**	**13**		**Project**				
			Total: 396						

Table A3.1: The product backlog for the project

HÁSKÓLINN Í REYKJAVÍK
REYKJAVÍK UNIVERSITY