

UNIVERSITY OF TROMSØ

# INF-2900 Group Project - ScheduleIT

by

Andrea Spreafico - asp005@post.uit.no

Stephan Abrahamsen - sab004@post.uit.no

Eva Garcia Domingo - edo015@post.uit.no

Emil Nysted Hagen - eha092@post.uit.no

Siarhei Kulakou - sku019@post.uit.no

A thesis submitted in partial fulfillment for the degree of Computer Science  
Computer Science

in the

Faculty of Computer Science  
Department of Computer Science

May 2017

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Aim of the project . . . . .	1
1.2	Aim of the course . . . . .	1
<b>2</b>	<b>Background Theory</b>	<b>3</b>
2.1	Agile Software Development . . . . .	3
2.2	Ruby on Rails . . . . .	4
<b>3</b>	<b>Analysis and Design</b>	<b>5</b>
3.1	Web application architecture . . . . .	5
3.2	Models design . . . . .	5
3.2.1	User . . . . .	5
3.2.2	Session . . . . .	6
3.2.3	Timetable . . . . .	6
3.2.4	Event . . . . .	6
3.2.5	Participant . . . . .	6
3.2.6	Contact . . . . .	6
<b>4</b>	<b>Team Work</b>	<b>7</b>
4.1	How we apply Agile process . . . . .	7
4.2	Communication between the team . . . . .	7
4.3	Struments and application used . . . . .	7
4.3.1	Github . . . . .	7
4.3.2	Pivotal Tracker . . . . .	7
<b>5</b>	<b>Discussion</b>	<b>8</b>
<b>6</b>	<b>Conclusions</b>	<b>9</b>
6.1	Evaluation . . . . .	9
6.2	Further Works on the project . . . . .	9
6.3	Further Works on the team work process . . . . .	9
	<b>Bibliography</b>	<b>10</b>

# Chapter 1

## Introduction

### 1.1 Aim of the project

Thinking of actual students' problems, we tried to find a solution to some of them. We have developed a whole online platform for any group of people able to share a calendar. Our aim was to provide them with a tool that could organize different schedules, being able to show each schedule separately or as a whole calendar that includes all the events. Although our first target was students, the platform can be used by everyone who need an organizer. The only requirement a person needs is creating an account and he or she will be able to create a timetable and share it with people to join it.

People who join a timetable will be able to see the whole calendar but, obviously, they will not be able to edit it. by doing this, we have ensured the safety of a timetable. In addition, in order to maintain the confidentiality of our users, the content of the website cannot be seen by someone without an account.

With the aim of helping new users to create accounts and to learn how to use the application, we have completely filled the Help and About pages. Furthermore, in case some question could not be answered through these pages, an email account has been created for contacting us.

### 1.2 Aim of the course

One of the purposes of this project was to learn how to work together in a cooperative environment. Some of the challenges we have successfully faced to are: sharing code, dealing with different schedules or finding a common idea of what to do and how to do it. We have learnt that having a meeting once per week is unavoidable for sharing ideas,

problems and struggles, asking for help and being updated.

Finally, another important objective consisted on learning to use the framework Ruby on Rails. We found that it is such a powerful web developer, hard to use for the first time, but very sensitive to any change. That is, with so few code lines you can make big changes in your application. In addition, compared with others frameworks, it is quite easy to put together the logic with the graphics: Ruby helps you as much as a software can.

## Chapter 2

# Background Theory

What the reader needs to know in order to understand the rest of the report. Examiners like to know that you have done some background research and that you know what else has been done in the field (where relevant). Try to include some references. Related work (if you know of any) What problem are you solving? Why are you solving it? How does this relate to other work in this area? What work does it build on?

### 2.1 Agile Software Development

## 2.2 Ruby on Rails

Ruby on Rails, or simply Rails, is a server-side web application framework written in Ruby under the MIT License. Rails is a model–view–controller (MVC) framework, providing default structures for a database, a web service, and web pages. [1]

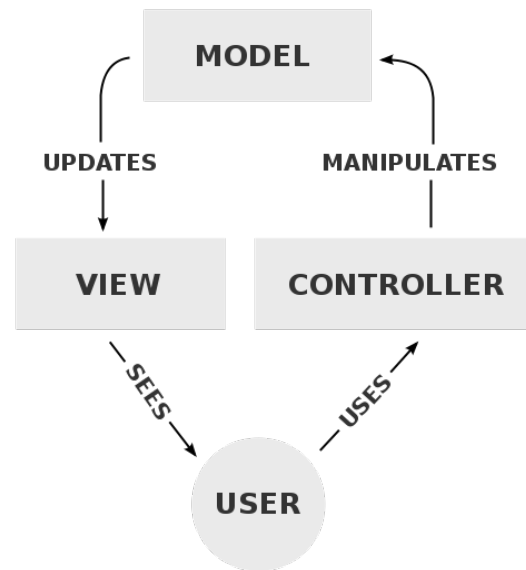


FIGURE 2.1: Diagram of interactions within the MVC pattern.

**Source:** <https://en.wikipedia.org/wiki/Model-view-controller>

### The Model:

- Contains data for the application (often linked to a database)
- Contains state of the application (e.g. what orders a customer has)
- Contains all business logic
- Notifies the View of state changes (\*\* not true of ROR, see below)
- No knowledge of user interfaces, so it can be reused

### The View:

- Generates the user interface which presents data to the user
- Passive, i.e. doesn't do any processing
- Views work is done once the data is displayed to the user.
- Many views can access the same model for different reasons

### The Controller:

- Receive events from the outside world (usually through views)
- Interact with the model
- Displays the appropriate view to the user

<https://stackoverflow.com/questions/1931335/what-is-mvc-in-ruby-on-rails>

## Chapter 3

# Analysis and Design

Concentrate on explaining the decisions made and the reasons for them.

### 3.1 Web application architecture

### 3.2 Models design

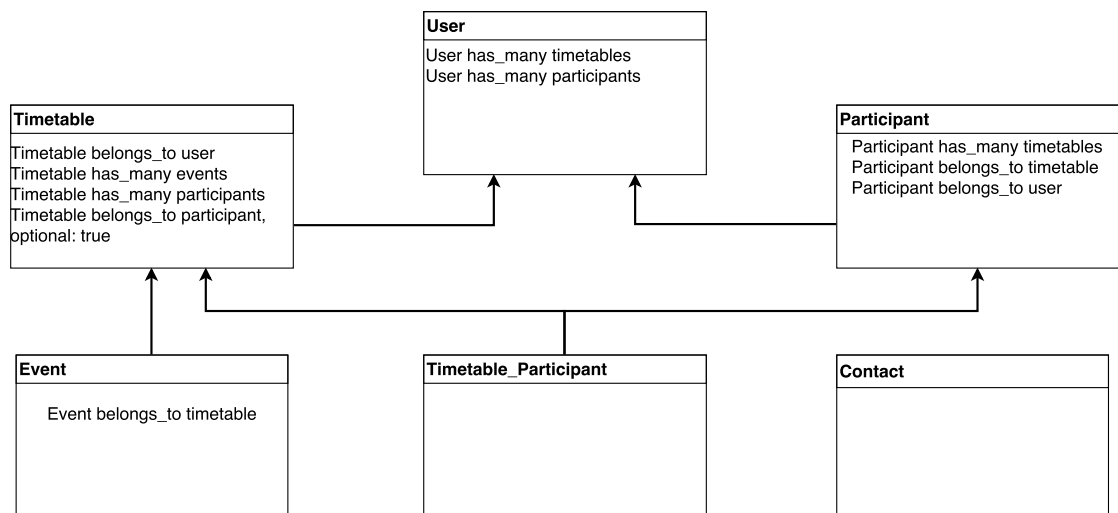


FIGURE 3.1: Implemented models.

#### 3.2.1 User

The user model is responsible for keeping track of the application's user information. It is directly related to the timetable and participant models, because each user must be able to own several timetables, as well as participating.

The rest of the model is set up to ensure that the user information is correct. It validates the user e-mail, uses a Bcrypt function to ensure that all users have passwords, and in turn, hashes the password string so that no plaintext passwords are stored in the database. An added feature also allows the user to upload an avatar (profile picture) for their profile.

### **3.2.2 Session**

### **3.2.3 Timetable**

The timetable

### **3.2.4 Event**

### **3.2.5 Participant**

### **3.2.6 Contact**

The contact model is a small and simple one, but it contains the necessary parts for a proper e-mail header. Whenever a user submits a new contact form, the model validates the fields of the form, as well as one "invisible" field, used for catching spam bots. This "spam catcher" works by refusing to submit forms where the field *nickname* contains any characters. Since the field being hidden by some CSS code, we know that a human can't see this field, unless they have access to the code, in which case it's most likely a robot of some kind.



## Chapter 4

# Team Work

### 4.1 How we apply Agile process

### 4.2 Communication between the team

### 4.3 Struments and application used

#### 4.3.1 Github

#### 4.3.2 Pivotal Tracker

## Chapter 5

## Discussion

## **Chapter 6**

# **Conclusions**

What have you achieved? Give a critical appraisal (evaluation) of your own work - how could the work be taken further (perhaps by another student next year)?

### **6.1 Evaluation**

### **6.2 Further Works on the project**

### **6.3 Further Works on the team work process**

# Bibliography

- [1] Wikipedia. Ruby on rails — wikipedia, the free encyclopedia, 2017. [Online; accessed 25-May-2017].