

# The calendar application

Web applications DAT076 / DIT126

Gabriella Thorén, Niklas Côté, Marcus Mathiason and Erik Tran

# Introduction

The application acts as a calendar application. The application allows users to register themselves to the database to then be able to login and create their own calendars. Depending on whether a user has read only or write access to a calendar, he/she will be able to add events to the calendar.

Github repo: <https://github.com/niklascote/WebApplicationProject>

## User cases

1. Register user
2. Log in
3. View calendars
4. Create calendar
5. Create event
  - a. Quickly add event with a limited detail via the calendar view
  - b. Add fully detailed event by clicking on the “more information”-button when adding event
  - c. Create recurrent events by checking the checkbox and specifying repetition range and frequency
6. View events
  - a. Current user’s event
  - b. Events shared with the current user
7. Invite other users to event
8. Edit event
9. Delete event
  - a. Remove this occurrence  
Removes the selected occurrence.
  - b. Remove all occurrences  
Removes all occurrences of the selected event.
  - c. Remove only from your schedule  
Removes shared event from the invited users calendar, and not for owner or other participants.
10. Invite users to calendar (Public calendar)
11. Event notifications
12. Edit profile settings
13. Log out

# User manual

## Home page / Log in

The users is greeted by a simple start page which gives the options of logging in or signing up depending if it's an old or new user. If the users inputs are valid then the user is redirected to the Calendar page upon clicking Login after registering or if already registered. If user clicks Sign up the user will be redirected to the Register page.

A screenshot of a web application's login page. The page has a light gray background. At the top center, the word "CALENDAR" is displayed in a bold, black, sans-serif font. Below it, there are two input fields: the first is labeled "Username (email)" and the second is labeled "Password". Both labels are in a small, gray, sans-serif font. Below the password field is a "Login" button with a gray gradient and a black border. At the bottom center, there is a link that says "Not registered? Sign up" in a small, gray, sans-serif font.

## Register

Upon clicking the "Sign up" button, you are greeted with the register screen. The user is prompted to enter his or her first name, last name, phone, email, password and if he or she wants notifications or not. The time zone is fetched automatically from the client. Upon clicking Register user, the user is automatically logged in and brought to the Calendar view. If the back button is clicked, the user is redirected to the home page.

# Register

Firstname:	
Lastname:	
Phone:	
Email:	
Password:	
Timezone:	GMT+1
Notifications:	true

Register user

[Back](#)

## Calendar

When the user has been signed in the calendar view is directly shown. If the user already has calendars assigned to him or her, the first one in the list will be shown. If the user has no calendars, a new empty calendar will be created for that user. The user can see all planned events in the calendar. Some event may be created by the user itself, and others may have been created by other user's that have invited the current user to join.

Gabriella Thorén Profile

Logout

March 2019

Su	Mo	Tu	We	Th	Fr	Sa
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

CALENDARS

Create new calendar

today

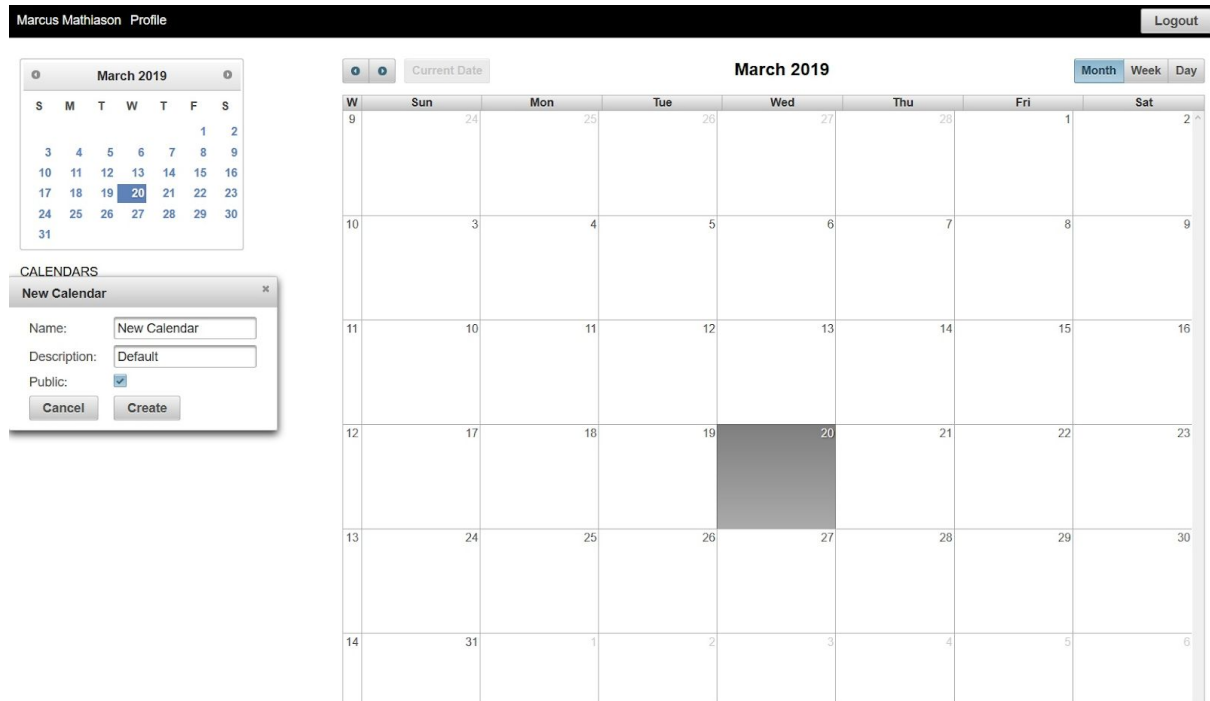
March 2019

month week day

W	Sun	Mon	Tue	Wed	Thu	Fri	Sat
9	24	25	26	27	28	1	2
10	3	4	5 event	6	7	8	9
11	10	11 test tst	12	13	14	15	16
12	17	18	19	20	21	22	23
13	24	25	26	27	28	29	30
14	31	1	2	3	4	5	6

## Create new calendars

Upon clicking the “Create new calendar” button, a dialog is opened prompting the user for the name, description and public access status of the calendar being created.



## Add events

The user can add an event to the calendar by pressing a date in the calendar.

Gabriella Thorén Profile Logout

March 2019

Su	Mo	Tu	We	Th	Fr	Sa
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

CALENDARS

Create new calendar

today

March 2019

month week day

W	Sun	Thu	Fri	Sat
9	2	27		2
10		6	7	8
11	1	13	14	15
12	1	20	21	22
13	24	25	26	27
14	31	1	2	3

Event Details

Event name:

Start date:

End date:

All Day:

Calendar:

Location:

Recurrent:

Repeat every:

Repeat # of times:

More information

Save

If the user want to add more details to the event, besides the ones shown in the previous image, he or she can press the “More information”-button, which will redirect the user to a more detailed page. In the detailed view, the user can specify some more information, for example invite other users to the event.

Event | Save Delete

Event name

Location

Description

07/03/2019 00:00

07/03/2019 23:59

All day

Reminder: None

Calendar: Testkalender

Attendees

Invite users...

anton svaren

Can view

lily lily

Can view

If the user wants to save the changes, he or she must press the “Save”-button. If the user do not want to save the changes he or she can press the “Delete”- or “x”-button. All these actions will redirect the user to the calendar view again.

## Adding recurring events

To create a recurring event, simply check the “Recurrent” checkbox and specify how the interval and how many times you would like the event to repeat.

The screenshot shows a web application interface for a calendar. At the top, there is a header with 'Marcus Mathiason Profile' and a 'Logout' button. On the left, there is a sidebar with a 'March 2019' calendar view and a list of calendars: 'Empty', 'Empty', 'test', 'test2', 'Empty', 'Empty', and 'Marcus' calendar'. Below this list is a 'Create new calendar' button. The main area displays a 'March 2019' calendar grid. An 'Event Details' dialog box is open, showing the following fields: 'Event name: test', 'Start date: 11/03/2019 00:00', 'End date: 11/03/2019 23:59', 'All Day: ☒', 'Calendar: Marcus' calendar', 'Location: ', 'Recurrent: ☒', 'Repeat every: Week', and 'Repeat # of times: 2'. There are 'More information' and 'Save' buttons at the bottom of the dialog box.

This screenshot shows the same calendar application after the event has been created. The 'Event Details' dialog box is no longer present. Instead, two blue bars representing the event 'test' are visible on the calendar grid. The first bar spans from Monday, March 11th to Tuesday, March 12th. The second bar spans from Monday, March 18th to Tuesday, March 19th. The sidebar and header remain the same as in the previous screenshot.

## Update events

If the user wants to update the details of an event, he or she can select the event in the calendar view. The edit view will then be shown. The details about the event can only be changed if the user is the owner of the event, or has write permissions to an event which he or she has been invited to.

Event | Save | Delete

tst

Location

Description

11/03/2019 08:00

11/03/2019 08:30

☒ All day

Reminder: None
Calendar: Testkalender

### Attendees

Invite users...

Gabriella Thorén

Owner

test test

Can view

### *Write permissions*

Event | Delete from your schedule

test

Location

Description

11/03/2019 00:00

11/03/2019 23:59

☒ All day

Reminder: None
Calendar: Testkalender

### Attendees

Invite users...

test test

Owner

Gabriella Thorén

Can view

### *Read permissions*

## Delete events

If the user wants to delete an event he or she can select an event in the calendar view.

If the user is the owner of the event, he or she has three options.



Event | Save Delete

tst

Location

Description

11/03/2019 08:00
11/03/2019 08:30
☒ All day

Reminder: None
Calendar: Testkalender

### Attendees

Invite users...

Gabriella Thorén
Owner

test test
Can view

This occurrence
All occurrences
Only from your schedule

If the user selects “This occurrence” only this occasion of the event will be deleted. This is necessary for cases where the occasion is repeated over multiple days, and the user does not want to delete all the occasions.

If the users selects “All occurrences”, all occasions of the event will be deleted. If the event is repeated over multiple days, all the repeating occasions will be removed.

If the users selects “Only from your schedule”, the event will only be removed for the current users. If the current user has been invited to an event created by another user, the function will only delete it from the current user’s calendar, and not affect anyone else invited to the event.

## Profile

In the calendar view there will consists a Profile-button in the menu bar. This button will take you to the profile view and there you can see your profile information.

Calendar

Erik Tran

Edit

Basic Information

First Name	Erik
Last Name	Tran
Email Address	erik@mail.se
User ID	1


Additional Information

Phone Number	12321387
Time Zone	GMT +1
Allow Notifications	true

## Edit profile settings

In the profile view there are two modes available, and they are view-mode and edit-mode. The page consists a Edit-button. Clicking this button will turn from view-mode into edit-mode. In edit-mode you will be able to edit profile settings. After you are done editing, click Save-button to save, or Cancel-button to cancel.

Calendar



Erik Tran

Save

Cancel

Basic Information

First Name

Erik

Last Name

Tran

Email Address

erik@mail.se

User ID

1

Additional Information

Phone Number

12321387

Time Zone

GMT +1

Allow Notifications

true

# Design

## Frameworks and other technologies used:

### Frameworks

- Java EE
- Java Server Faces
- Primefaces
  - Primefaces themes

### Libraries

- Omnifaces
- Prettyfaces
- Lombok
- Eclipselink

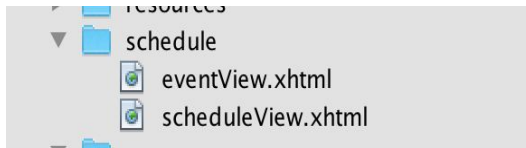
We chose to work with Java EE and Java Server Faces since we had used it in the labs and felt that less time would be spent on learning new frameworks if we chose ones that we all already had worked with prior to the project. We chose Primefaces because it features components useful for writing a calendar application, which was perfect for the project.

Omnifaces was used at first, to make use of the time libraries it features, but after time zone was switched over to being fetched with javascript, it was instead used for its converter. Prettyfaces was used to simplify navigation to the different html files. Lombok was used for its `@Getter` and `@Setter` annotations. Eclipselink was used to simplify conversations with the database.

## Filter

```
<!--Filter-->
<filter>
    <filter-name>sec</filter-name>
    <filter-class>com.WebApplicationProject.control.SecureFilter</filter-class>
</filter>

<filter-mapping>
    <filter-name>sec</filter-name>
    <url-pattern>/schedule/*</url-pattern>
</filter-mapping>
```



These lines does so every request to the “/schedule/\*.xhtml” folder goes through the SecureFilter class which is in “com.WebApplicationProject.control” package.

SecureFilter Class implements Filter.

```
String loginURI = request.getContextPath()+"/index.xhtml";

boolean loggedIn = (session != null && session.getAttribute("email") != null) ;

if (!loggedIn) {
    if(loginURI.indexOf("/scheduleView.xhtml")>=0 || loginURI.indexOf("/eventView.xhtml")>=0){
        response.sendRedirect(request.getServletContext().getContextPath() + "/index.xhtml");
    } else
        chain.doFilter(req, resp);
}
}
```

The SecureFilter class checks if there is a session and if the session has an attribute and saves this as a boolean named “loggedIn” and then redirects the application user depending if the “loggedIn” variable is true or false.

## Authentication Control for Login

```
<h:head>
<h:outputStylesheet library="css" name="default.css" />
<title>Login to Calendar</title>
</h:head>
<h:form styleClass="loginView">
<div>
<p:inputText type="text"
value="#{authController.email}"
required="true"
placeholder="Username (email)"/>

<p:inputText type="password"
value="#{authController.pass}"
required="true"
placeholder="Password"/>

</div>
<div>
<p:commandButton
ajax="false"
value="Login"
action="#{authController.login}"/>

</div>
</h:form>
```

AuthController class is a requestscoped bean which controls if the users email and password is correct or not. The input of the bean is received in loginView.xhtml as the picture above and is used in AuthController as below.

```

public String login() throws IOException {
    FacesContext context = FacesContext.getCurrentInstance();
    if (validate(email, pass)) {
        loggedIn = true;
        HttpSession session = SessionUtil.getSession();
        session.setAttribute("email", email);
        return "/schedule/scheduleView";
    } else {
        context.addMessage(
            null,
            new FacesMessage(FacesMessage.SEVERITY_WARN,
                "Incorrect Email and/or Password",
                "Please enter correct Email and Password"));
        return "/index.xhtml";
    }
}

```

If the validation of email and password is correct, a session will get the attribute of the email and the user will be redirected to the scheduleView.xhtml else a failure message will be displayed and a redirection to the welcome page will be done.

```

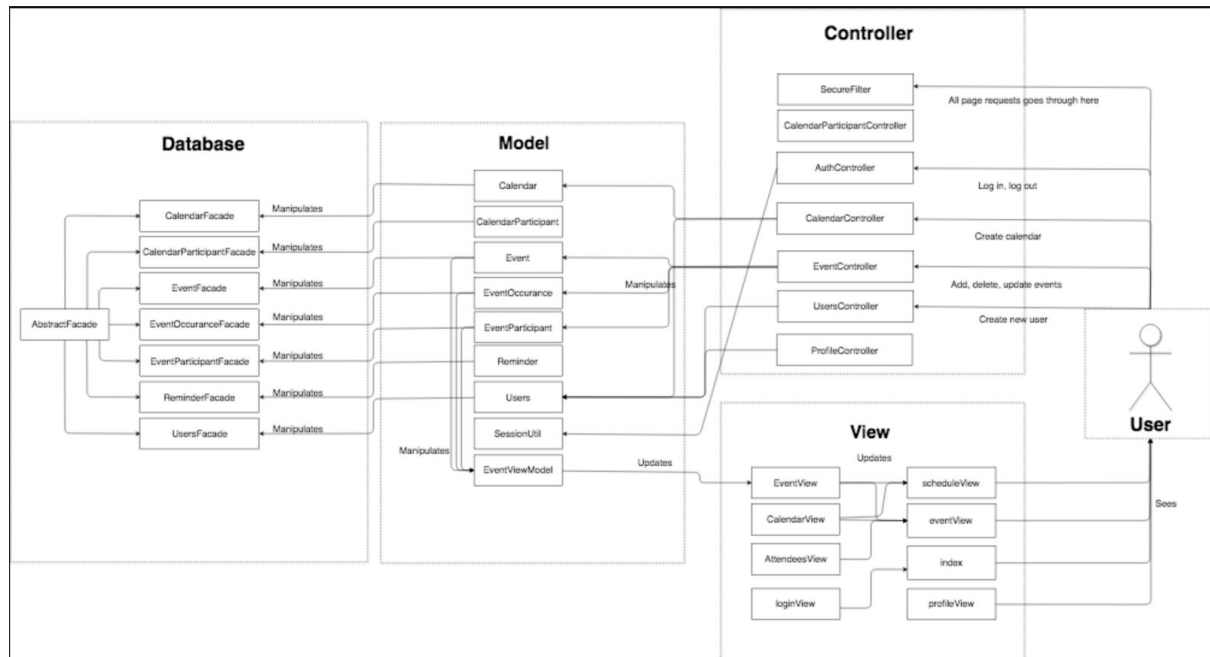
private boolean validate(String email, String pass) {
    Users user = ufacade.users(email);

    return !(user == null
        || (user.getPassword() == null ? pass != null : !user.getPassword().equals(pass)));
}

```

The validation method makes a request to the UsersFacade and stores it as a User, the UsersFacade makes a query to the database to get all Users with the same email as the input variable in AuthController, because there is only one user with that email the query it will return a List<Users> with only one user in it and then return the user, the user will be used as a temporary user in AuthController validation method.

# Application flow



# Responsibilities

Tasks were set up in a Scrum board at the start of the project. Every project group member then assigned tasks to themselves to work on until done, after which, the process was repeated.

Marcus:

Code:

- Created register view - Create.xhtml
- Created home page - index.html
- Calendar controller
  - Create new calendar
    - New calendar is created if user has no calendars already, if user has calendars in collection, fetch collection
- Event controller
  - Add recurrent events
  - Delete events
- User controller
  - Time zone functionality, originally fetched through javascript in Create.xhtml
- Calendar view
  - Functionalities
    - Create new calendar
    - View calendar list of public calendars
    - Create event
      - Create recurring events

Report:

- Wrote the sections that handles the parts of the project I have worked on.

Niklas:

Code:

- SecureFilter, a class which acts as a Login Filter for all pages in schedule folder, long investigation process of filters
- LoginView, view for input parameters for login
- AuthController, handles input from LoginView and checks if user input is valid
- SessionUtil, handles the session of which will be set if user input is valid

- Notification in ScheduleView, JavaScript which creates a simple notification

## Report

- UserManual - Home Page / Login
- Design - Filter and Authentication Control for Login

## Gabriella Thorén:

- Created the Maven-project
- Created the basic file structure
- Added the dependencies:
  - Lombok
  - Java EE
  - Omnifaces
  - Primefaces and themes for Primefaces
  - Prettyfaces
  - Javax persistence
- Created database structure
- Created entities for:
  - Calendar.java
  - CalendarParticipant.java
  - Event.java
  - EventOccurance.java
  - EventParticipant.java
  - Reminder.java
  - Users.java
- Calendar view
  - CSS-styling
  - HTML-structure
  - Functionalities:
    - View current user's calendars
    - View current user's event
    - View events shared with the current user
    - Add event
- Event view
  - CSS-styling
  - HTML-structure
  - Functionalities:
    - Add event
    - Share events with other users
    - Edit event
    - Remove event
      - Remove this occurrence



- Remove all occurrences
  - Remove only from your schedule
- Report writing
  - Written the parts in the report concerning things that I have been involved with.

Erik Tran:

Code:

- Helped Niklas with the Login functionality (AuthController.java and loginView.xhtml)
- Edited index.xhtml with jsf tags to avoid any kind of html injections.
- Connecting inputs and styled pages with primefaces (on index.xhtml, loginView.xhtml, profileView.xhtml)
- Created profileView.xhtml and styled it with primefaces and jsf.
- Created ProfileController.java
  - Manipulates the User-table on database
- Created Edit-button with functionality that changes the page from “view-mode” into “edit-mode” on profile page.
- Created Save-button on profile page
  - Updates the database.
  - Turns back into “view-mode” after button clicked
- Created Cancel-button on profile page
  - Resets the input values back to default values (values before they were edited)
  - Turns back into “view-mode” after button clicked

Report

- Profile section
- Edit profile settings