StuderaMera
Requirement Analysis Document

Date: 23/10/2020

Authors: Izabell Arvidsson, Ida Dahl, Julia Jönmark, Hanna Söderström

This version overrides all previous versions

Contents

1. Introduction

The project aims to create a computer based application that students can use to help manage their time. The application is divided into three parts, a timer, a calendar, and a to do-list. The timer is for study sessions the user might have, where the user can decide how long the sessions will be. The calendar is for the user to add different events, making it easier to keep track of activities. The to do-list is to help the user structure different tasks that may exist. All these features to help students structure their study time.

The application is adapted to desktops and will be sized to fit most displays.

Once the application is started the user will be greeted by a first side and three options. The three choices are the three main functions in the program. The first choice is a timer. On this page the user will be able to use the timer for study sessions. The user can choose how long to study for, for how long the breaks will be and how many repetitions. The timer will count down and as the study session is progressing, a flower will grow. The flower will only grow during when the user is studying and not during rest time. If the user chooses to terminate the study session before the timer is done the flower dies. This page will also have a tips-button, where information about different study methods and techniques.

The second choice is a to do-list. On this page the user can create different lists of tasks the user has to finish. The user will be able to set a date and time for the tasks. There is also a checklist where the user can put in information, to later check the tasks of when finished. The user will also be able to delete lists that are either finished or not relevant.

The third and last choice is a calendar. On the calendar page the user will be able to see the current month, to move back and forth between months and to create events. When creating an event the user will have to put in a date, a time, and a title. The event will then be placed in the same date as the date the user put in, though all the information won't be displayed. It's when the user clicks on the event that a window pops up to show the information about the event.

1.2 Definitions, acronyms and abbreviations

Timeline - Makes it possible to update the property values along the progress of time

Spinner - A function built in SceneBuilder, which this application uses to build the graphical part, that makes sure that the user can choose between different values to start the timer with

Design pattern- is a general, reusable solution to problems that often occurs in software design

Observer Pattern- A design pattern which makes it possible for multiple objects to be notified when an event happens to an object who they are observing without having multiple dependencies

int- stores integer type data, is primitive

Integer- stores integer types, is a class type

String- is a sequence of characters

Getter- a method that reads value of a variable

Setter- a method that updates value of a variable

Definition of Done- The acceptance criteria that are common for all user stories

ScrollPane- A function built in SceneBuilder, which this application uses to build the graphical part, that allows the application to lay tasks on each other

Checkbox- A function built in SceneBuilder, which this application uses to build the graphical part, that allows the user to check off tasks

Escape Hatch- A way for the user to get to a know place, in this application it is the first page

2. Requirements

2.1 User Stories

High Priority:

User Story
Implemented: Yes
Story Identifier: T008
Story name: Study sessions

Description
As a student who must study, I want a timer to help me concentrate better

Functional
-Will I be able to start a timer?
-Will I be able to see the timer counting down?

User Story
Implemented: Yes
Story Identifier: T007
Story name: Stop timer

Description
As a busy person I want to be able to stop the timer so if I have something else to do I can stop studying

Functional
-Can I stop the timer by clicking a button?

User Story
Implemented: Yes
Story Identifier: T001
Story name: Timer intervals

Description
As a person I want to be able to decide my own intervals for when to study and when to relax since I have difficulty focusing.

Functional
-Can I choose for how long to study?

-Can I choose for how long to rest?
-Will I be able to see how much time I have left?
-Will I be able to see a difference between study and rest time?

Non-functional:

Availability:
-Is there a limit for how long I can study?
-Is there a limit for how long I can rest?

User Story
Implemented: Yes
Story Identifier: T005
Story name: Countdown

Description
As a student I want a countdown clock so that I know if it is time to study or relax

Functional
-Can I see if it's study or rest time?
-Can I see the time ticking down?

User Story
Implemented: Yes
Story Identifier: P001
Story name: Schedule

Description
As a student I want to be able to build my own schedule so that I can easier structure my week

Functional
-Will I be able to give the events different colours?
-Will I be able to edit already existing events?
-Will I be able to delete existing events?

Security:
-Are other people able to view my schedule?

User Story
Implemented: Yes
Story Identifier: P004
Story name: Add events

Description
As a (kind of) structured person I want a button where I can add activities to my schedule so that I can have a summary over my week

Functional
-Can I add events to the schedule?

User Story
Implemented: Yes
Story Identifier: P006
Story name: Save events

Description
As a sometimes forgetful person I want to be able to save my events so that I don't forget them

Functional
-Will I be able to save an event by pressing a button?

User Story
Implemented: Yes

Story Identifier: P008
Story name: Add to do-list

Description
As a structured person I want a button where I can

Functional
-Will I be able to add tasks to to-do-lists?

User Story
Implemented: Yes
Story Identifier: P002
Story name: To do-list

Description
As a student I want to be able to make a to do-list so that I don't miss something

Functional
-Will I be able to edit existing tasks?
-Will I be able to see an overview of the task?
-Will I be able to check off tasks?

Security:
-Are other people able to view my to-do-lists?

User Story
Implemented: Yes
Story Identifier: P007
Story name: Save tasks

Description
As a sometimes forgetful person I want to be able to save my tasks so that I don't forget what I have to do for the day/week

Functional
-Can I save my tasks by pressing a button?

Medium Priority:

User Story
Implemented:
Story Identifier: P009
Story name: Edit tasks

Description
As a person I want to be able to edit already existing tasks

Functional
-Will I be able to edit tasks that are already saved?

Non-functional:
Availability:
-Will there be a button that indicates that the task is editable?

User Story
Story Identifier: P010
Story name: Edit events

Description
As a person I want to be able to edit events that are save to the calendar so that if an event changes I can adjust the event

Functional
-Can I edit events after they are saved?

User Story
Implemented: No, due to not enough time
Story Identifier: P003
Story name: Overview for calendar

Description
As a student I would like to see an overview over what my schedule looks like

Functional
-Can I see an overview of my day?
-Can I delete events from the overview?
-Can I edit events from overview?

Security:
-Are other people able to see my overview?

User Story
Implemented: No, due to not enough time
Story Identifier: P005
Story name: Overview for to do-list

Description
As a student I want an overview over the tasks I must do during the week or day

Functional
-Will I be able to see an overview the different to do-lists
-Can I delete to do-lists via the overview?
-Can I check off the tasks?

User Story
Implemented: Yes

Story Identifier: A002
Story name: Escape Hatch

Description
As a person who is not very technical, I want a button that easy takes me back to the front page

Functional
-Will I be able to always find my way back to the first page?
-Will the changes I made be saved if I leave the page?

Non-functional:
Availability:
-Will the escape hatch be available everywhere?

User Story
Implemented: Yes
Story Identifier: T006
Story name: Booster

Description
As a somewhat unmotivated student I want to get some kind of visual booster to keep studying

Functional
-Will the visual booster be visual on the screen at all times?

Non-functional:
Availability:
-Can I leave my study session without any consequences regarding the booster?

User Story
Implemented: Yes
Story Identifier: T004
Story name: Motivation

Description
As a person who often leaves applications, I want something visual that makes me stay

Functional
-Can I do something else in the application while the timer is ongoing?

Non-functional:
Availability:
-Will the calendar be available?
-Will the to do-lists be available?

User Story
Implemented: Yes
Story Identifier: T002
Story name: Set intervals

Description
As a student I want to be able to select my own intervals for studying since it will help me get more motivated

Functional
-Can I choose a specific time to study and rest for?
-Can I see how much I have left of the interval?

Non-functional:
Availability:
- Can I set an uneven number for both study and rest time?

Low Priority:

User Story
Implemented: Yes
Story Identifier: T003
Story name: Tips and techniques

Description
As a person how wants help with staying focused, I want to get tips and facts about different methods how to keep focus since I want to be more effective

Functional
-Can I find help on how to better my studying techniques?

Non-functional:
Availability:
-Will I be able to add my own methods?
-Will I be able to find the sources?

User Story
Implemented: Yes
Story Identifier: A001
Story name: Help

Description
As a person I want a help button where I can go if I get stuck

Functional
-Can I get help if I get stuck?

Non-functional:
Availability:
-Can I access the help page from anywhere?

User Story
Implemented: Yes
Story Identifier: A003
Story name: Beginner

Description
As a beginner of the application I want a tool that can help me if need be

Functional
-Will I easily see where I can go?
-Will there be a walkthrough of the application?

Non-functional:
Availability:
-Will I be able to get help from anywhere in the application?

2.2 Definition of Done

Not only must the user stories meet the acceptance criterion for just that story, but they must also meet the mutual acceptance criterion. The common acceptance criterion we have used for these user stories are:

-They should be tested
-They should be reviewed

2.3 User interface

The user interface is created to have a consistent design throughout the whole application. This is made so to make the user feel more comfortable and to faster get to know the applications and its features. Once the user starts the program the first page will appear with three choices. The three different buttons lead to either the timer, the to-do-list, or the calendar.

Figure 1: First Page

2.3.1 Timer

By clicking on the “Till timer”-button the user will be sent to the timer page. On this page the user will be greeted by three different spinners, a “Starta”-button, and a button with a light bulb. With the spinners help the user can decide for how long to study or rest. The user does so by clicking on the tiny arrows that the spinners have. There will be a limit for how long to study, which is 100 minutes. The same goes for rest time, where the limit is 40 minutes. Another limit that is relevant to both times is that the spinners takes 5 minutes in one step. This means the user will only be able to choose a time that ends in either 0 or 5. There is a third spinner, and this on is for repetitions. Here the user can decide for how many times to repeat the study and rest time.

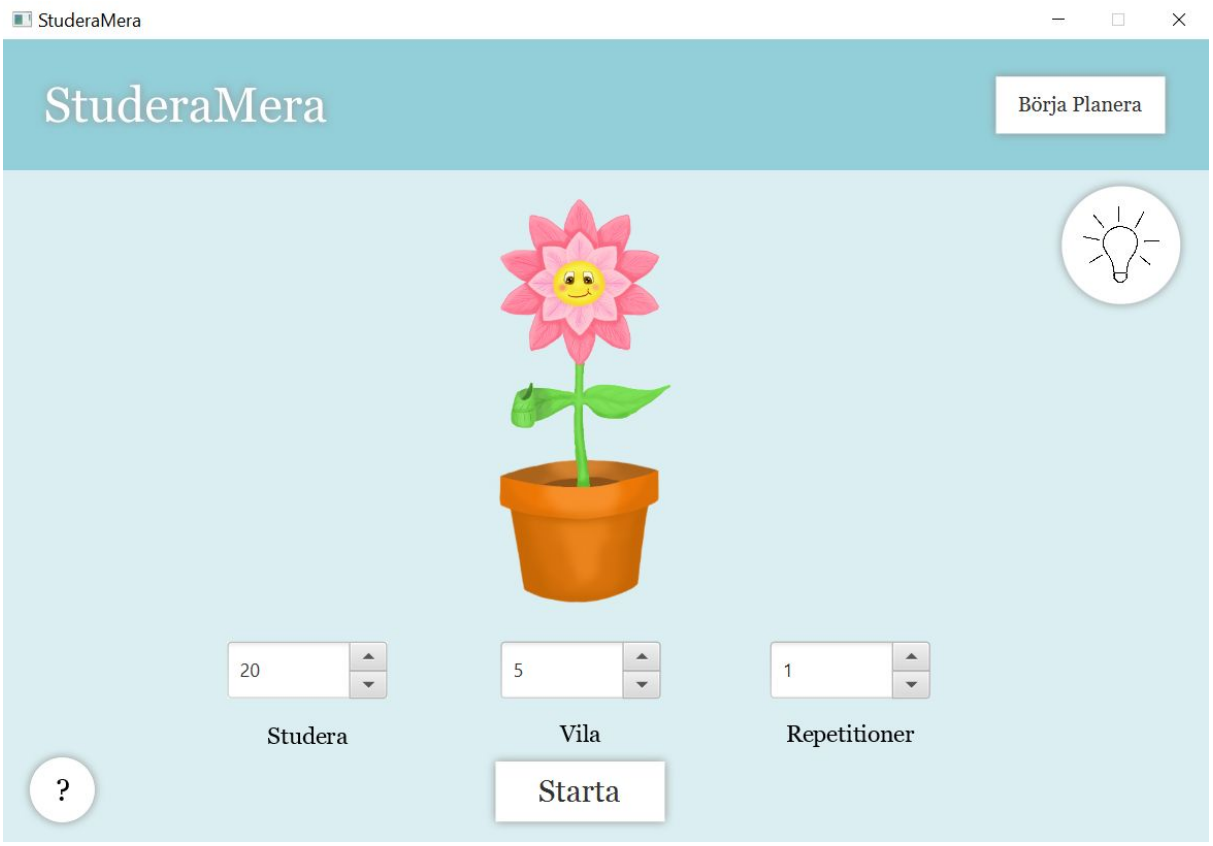

Figure 2: Start page for the timer

Once the user has chosen a time for both study and rest, all the user must do to start the timer is click the “Starta”-button. Once the button is pressed the user will be taken to a new page and the timer will start to countdown. During the study period an image of a flower will appear on the screen. The flower will grow successively with the time chosen for study and the number of repetitions. The application is coded so that the flower will not grow during rest time.

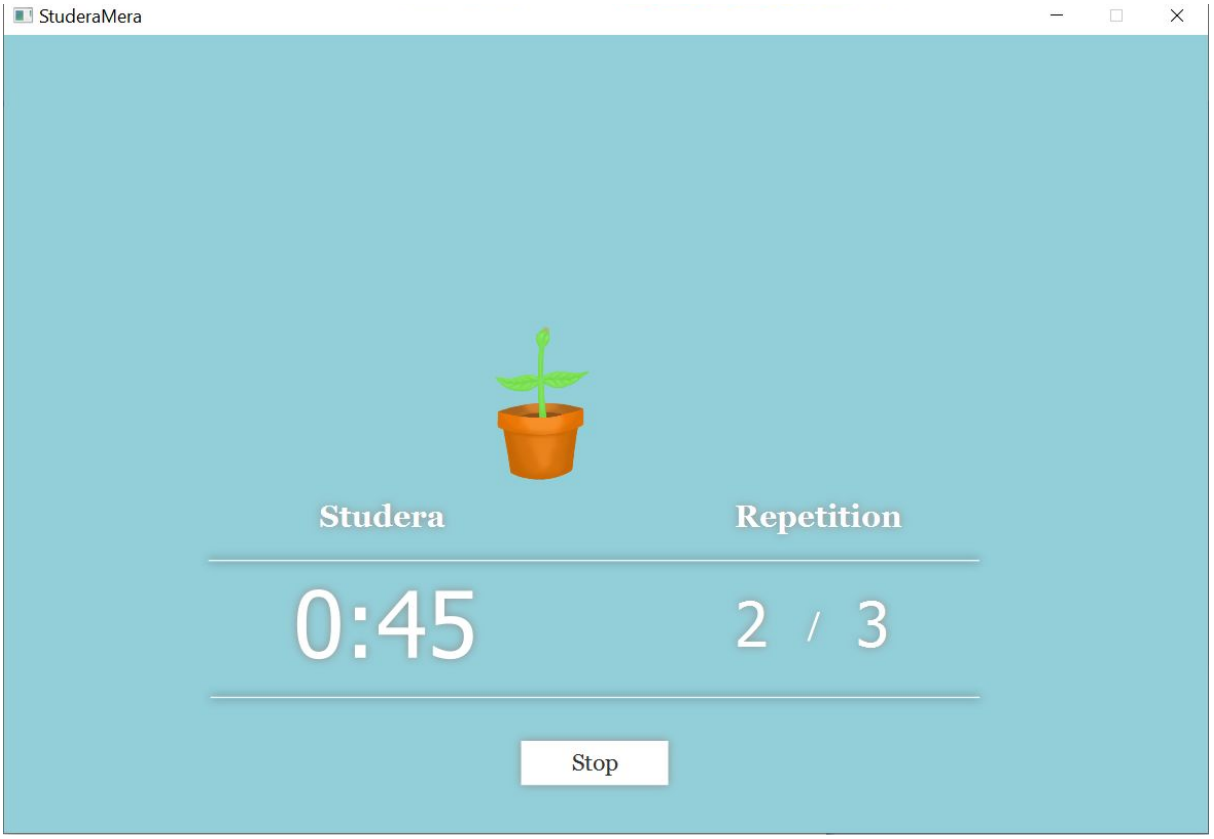

Figure 3: The timer is running

When the study period is completed a new window appears to show the user that the session is done. The meaning of this is go give the user a feeling accomplishment. Once the “Tillbaka till startsidan”-button is pressed the user will be taken back to the timer page.

Figure 4: The study session is done

If the user wishes to end the session before the time has run out, the user can do so by pressing the “Stop”-button. The countdown will then be paused, and a window will appear, asking the user if the session is to be ended. If pressed no, the user will go back to the timer and the countdown begins again. If pressed yes, a window will appear with a dead flower and a message that lets the user know they failed the study session. The message also reminds the user to think of a good time to study for, especially if the time chosen was not working.

Figure 5: The user has decided to end the study session before time was out

By pressing the “Jag förstår”-button the user will be taken back to the timer page. Here the user can decide on a new time or go to the calendar or to do-list.

2.3.2 To do-list

If the user chooses to go to the to-do-list, the user will be met by an overview of the to-do-list. Here the user can add, change, and delete tasks.

Figure 6: The to do-list page

To add a task the user must press “+”-button. A window will then pop up where the user can write specifics for just that task. The task can have a title, a description, and a checklist. The user will be required to fill in a deadline and a time in which the task will be completed, to be able to add the task.

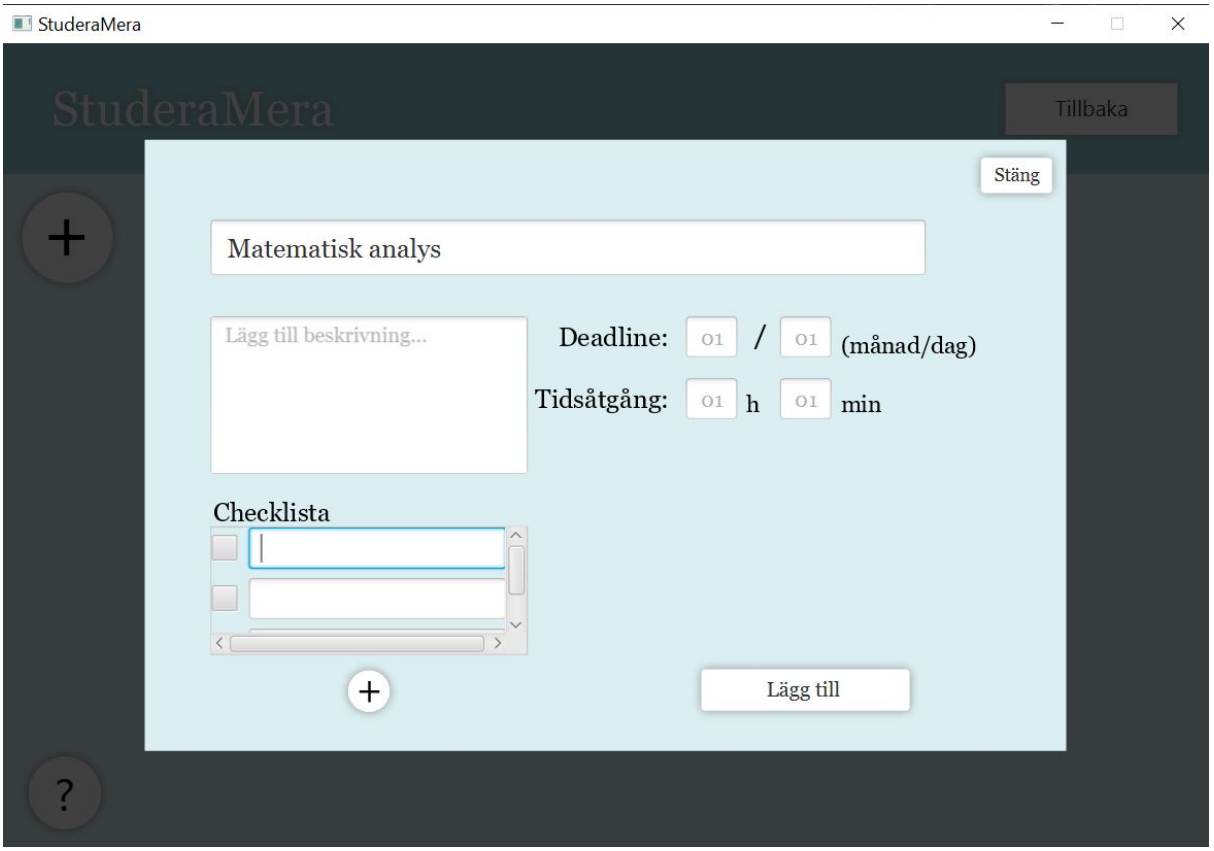

Figure 7: Adding a task

After all fields are filled in the user must press “Lägg till” and the task is saved. The task will then be seen in the grey scrollpane. If the user by chance wrote wrong or the task itself changed the user can change the task by clicking on it.

Figure 8: The task is added to the to do-list

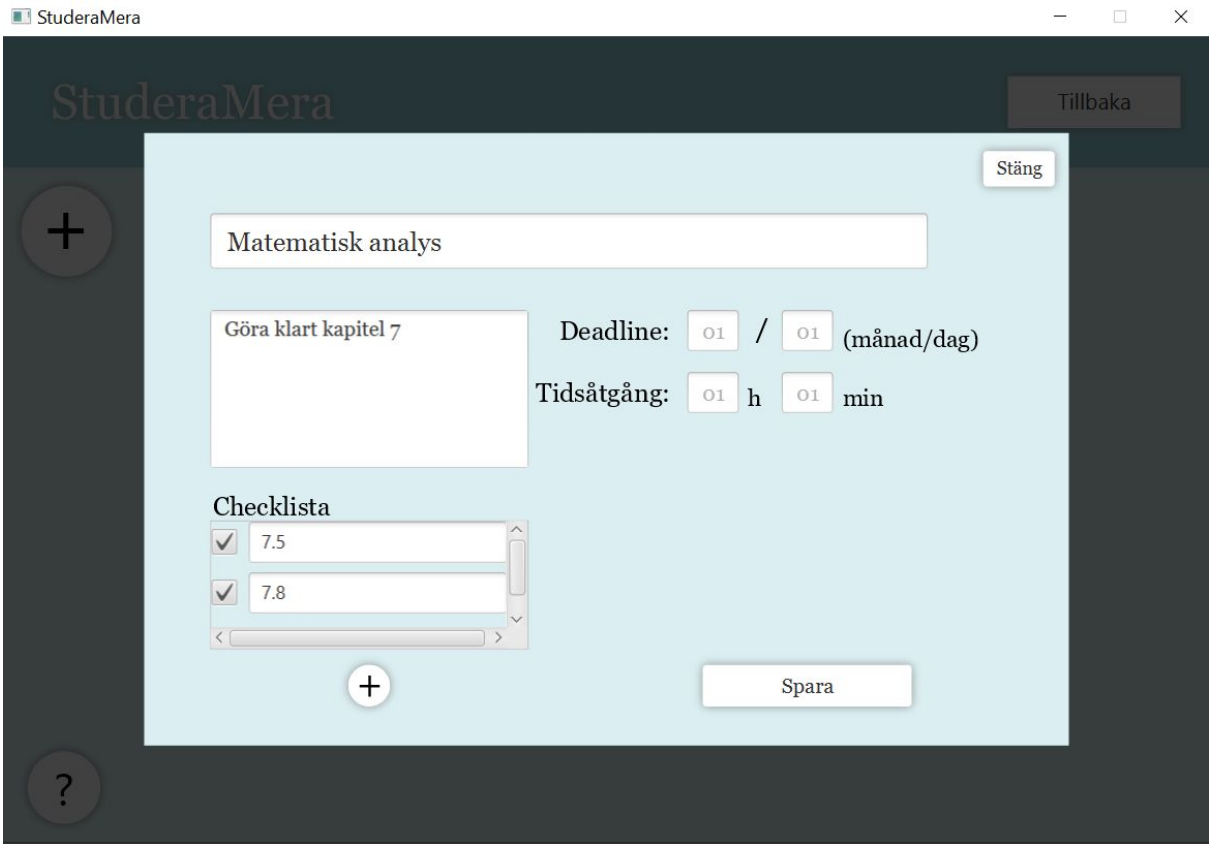

Figure 9: Changes are made to a task

In the overview of the to-do-list page the user can see the different tasks that have to be completed. On the task there is information regarding how many of the task's checklists are done and if the whole task is completed. There is also an image of a trash can. By clicking on the image the task will be deleted. The user can check off the whole task by clicking on the grey checkbox and uncheck it by clicking on the checkbox again.

Figure 10: A task is checked off

2.3.3 Calendar

By clicking on the “Till kalender”-button the user is taken to a page that has an overview of the current month. The current day has a different colour then the other days, to indicate that is the day.

Figure 11: The calendar page

Here the user can add events to certain days by pressing on the “+”-button. The user can then give the event a title, a description, a location and a colour. The event requires a date so that it can be placed on the same date that the user decided. The event is saved once the user clicks on the “Lägg till”-button.

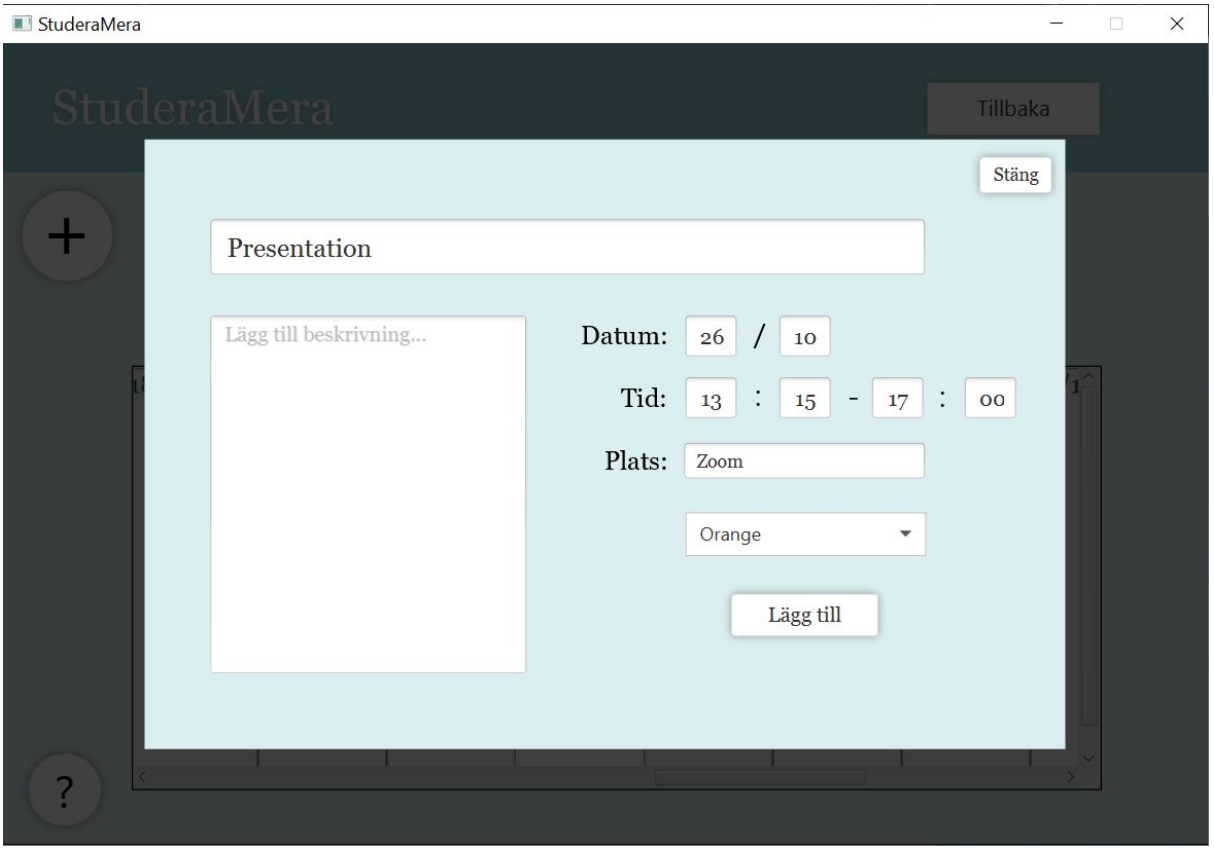

Figure 12: Adding an event

Figure 13: The event is added to the calendar

If pressed the “Föregående”-button changes the overview to the month before. There is not a limit for how long back the user can go. The same applies to the “Nästa”-button, which changes to the next month. To make the user aware of the change, the date under the header “Din kalender” changes to the corresponding month.

2.3.4 Help page

The help page can be reached from every page, except when the timer is running. By pressing the button with a question mark, in the lower left corner, the user will be transferred to the help page. Here the user can get information on often asked questions, contact and references. From here the user can go back to the first page to then decide what to do.

Figure 14: The help button is circled in red

2.3.5 Back button

On every page the user can find a “Tillbaka”-button. This button will take the user back to the first side, where the user will have three choices.

Figure 15: The back button is marked in red

3. Domain model

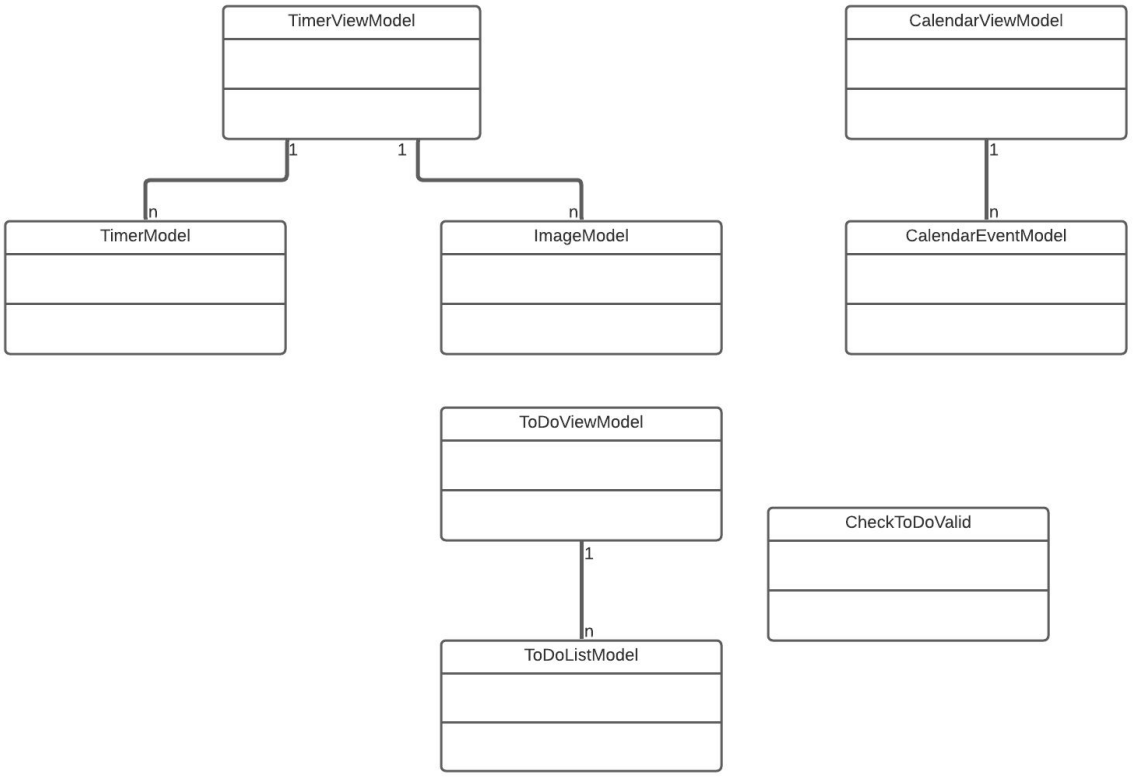

Figure 16: domain diagram

3.1 Class responsibility

CalendarEventModel

This class holds the constructor for a calendarEvent as well as necessary setters and getters.

CalendarViewModel

This class is responsible for the logic behind saving calendar events while the program is running and between startups. It also makes sure that the calendarEvents are displayed and in the correct FlowPane.

CheckToDoValid

This class is not dependent on other classes. Its responsibility is to check the status of the checklists and checkboxes it gets from the ToDoListView class in its methods.

ImageViewModel

This is the model that is responsible for sending the right number for the image that the TimerView is supposed to show. Takes in the amount of studyseconds that has passed and the total studytime in seconds and returns an int that the TimerView receives.

TimerModel

The model that is responsible for the converting of the time in form of integer to a string which will be shown by the TimerView.

TimerViewModel

Handles the functionality of the timer and creates timelines which will be going through the class' methods whilst running. It sends updates about all the different parts connected to the timer to the TimerView, through the Observer pattern, which updates the graphics.

ToDoListModel

The ToDoListModel-class creates a new todo-list with the data it got from the user from the ToDoViewModel. A todo-list contains a name, description, timeline, deadline and checklists. By calling on the class getters or setters the values can be reached and changed.

ToDoViewModel

This class toDoLists handles the functionality regarding the todo-lists. It has a list of all the todo-lists that it has created and saves them when the program shuts down and writes them back up when the program starts again. It has dependencies on ToDoListModel and the ListInToDoView because the class creates an instance of them.

4. References

Oracle.com, “ScrollPane”, (2020), Recived 2020-10-21 from
https://docs.oracle.com/javase/8/javafx/api/javafx/scene/control/ScrollPane.html

Oracle.com, “Timeline”, (2020), Recived 2020-10-21 from
https://docs.oracle.com/javase/8/javafx/api/javafx/animation/Timeline.html

Codejava.net, “Java Getter and Setter Tutorial”, (2019), Recived 2020-10-21 from
https://www.codejava.net/coding/java-getter-and-setter-tutorial-from-basics-to-best-practices

Tutorialspoint.com, “Difference Between an Integer And Int in Java”, (2019), Recived 2020-10-21 from
https://www.tutorialspoint.com/difference-between-an-integer-and-int-in-java

Wikipedia.org, “Software Design Pattern”, (2020), Recived 2020-10-21 from
https://en.wikipedia.org/wiki/Software_design_pattern

Medium.com, “The Definition of Done, What Does ‘done’ Actually Mean?”, (2017), Recived 2020-10-21 from
https://medium.com/@dannysmith/the-definition-of-done-what-does-done-actually-mean-ef1e5520e153