High-Level Design

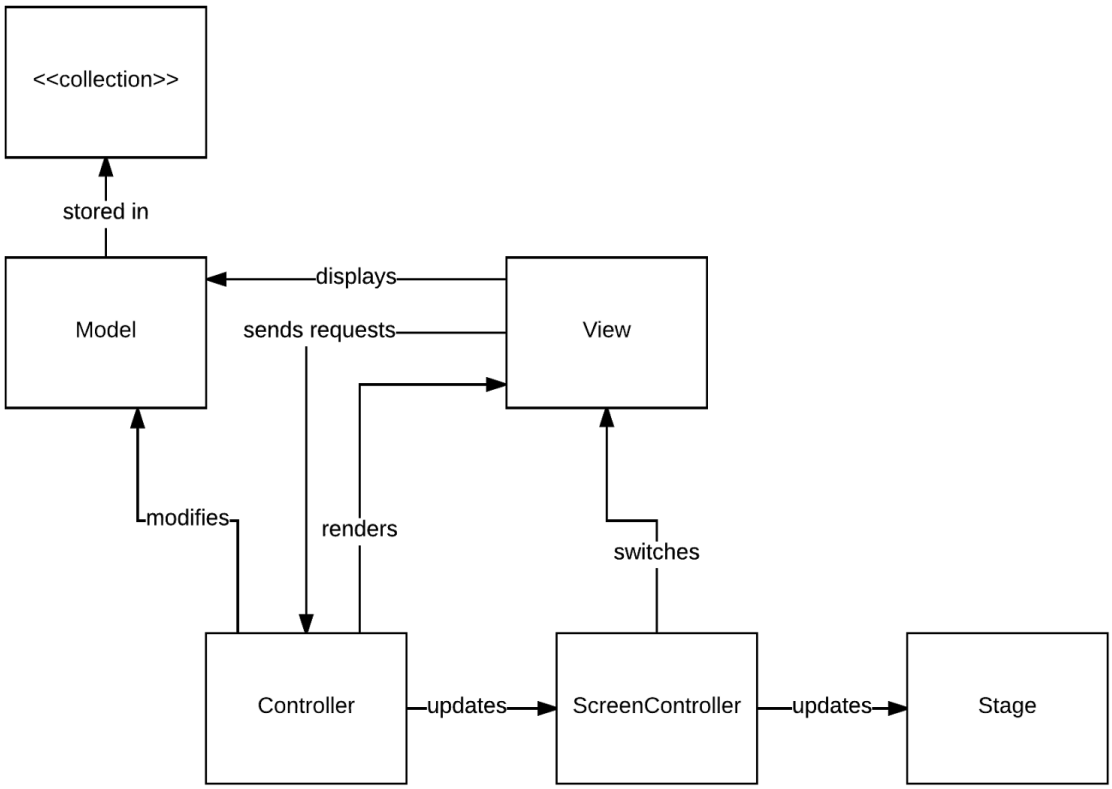
Introduction

High Level Design (HLD) is our global overview and design of the system. It covers the construction, layout, and database design of the system. The links between the system's different functions and modules are described at High Level Design.

The purpose of this High level design document is to attach the essential details to the procedure of our current project, so that we delineate an appropriate model of the required application's architecture. This is based on the use cases which are created in the analysis.

At first, a high level class diagram and its features are presented. After that, Robustness diagrams and the related functions are described at this document.

Class Diagram



This diagram represents an overview of the classes in the application. The application will be developed using a MVC structure, dividing Model, View and Controller. Models are stored in a separate storage class; various views are shown through a separate ScreenController class.

Models:

- Timeline**:** This class holds the timeline attributes, all particular events and their details. It verifies upon any action the validity of the information provided by the user, and decides on how to proceed.
- Event**:** This class stores event attributes, and holds a reference to its own timeline and priority/position in that specific timeline. This class's duration depends on the pop up window option for duration and non duration events.

Views:

- New timeline: screen to enter information about new timeline
- New event: screen to enter information about new event
- Timeline details: screen where the timeline is shown
- Event details: screen showing details about existing events, after clicking on them.

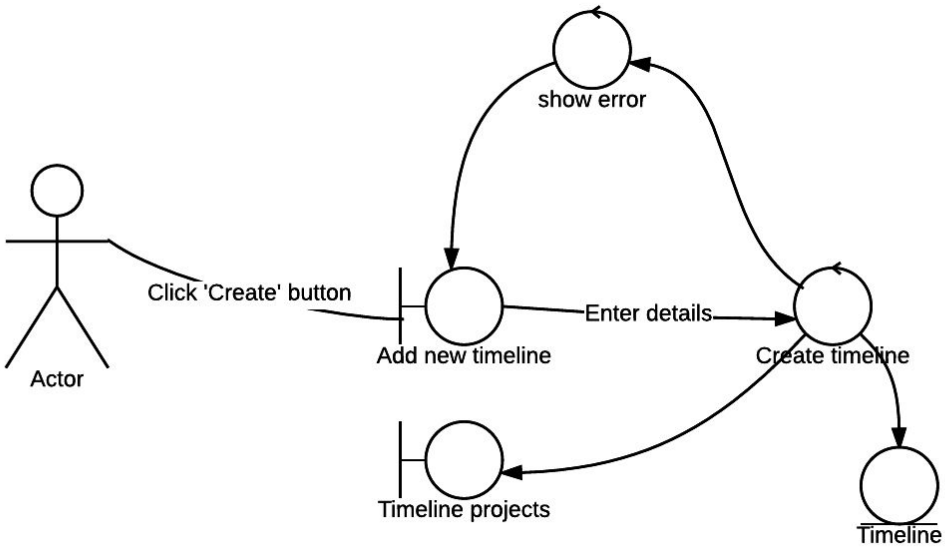
- Timeline overview: showing a list of all created timelines.

Java-FX will be used to create a graphical representation of data. In combination with Java-FX, Scenebuilder will be used to create FXML files. These FXML files represent the Views.

Robustness diagrams

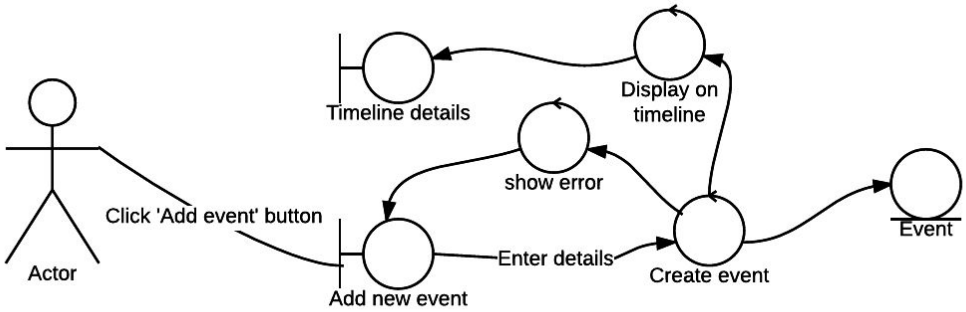
Robustness diagrams function as bridges between the analysis and design. Using the use cases from the analysis, a high level layout of the processes inside the application is shown, which will ultimately lead to the required classes and methods in the class diagram.

Add new timeline:



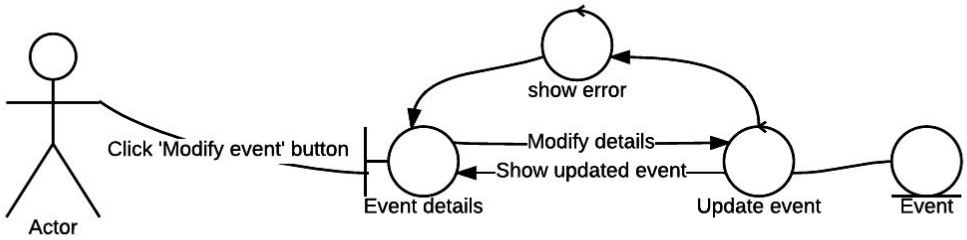
1. The user starts the application, and decides to create a new timeline.
2. A window is shown where the user enters the details about the timeline, such as title, start and end date and an optional description.
3. Then, the user clicks OK, after which the system checks if the entered details are correct. If not, an error will be shown, and the user will return to the new timeline screen to change details.
4. If all details are correct, the system creates a Timeline object, after which the timeline is shown on the canvas.

Add new event:



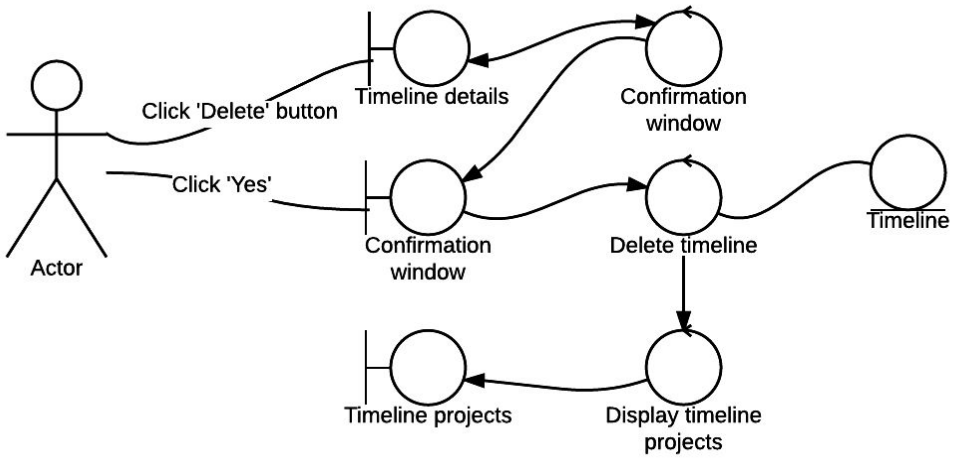
1. In the timeline details screen, the user clicks the <<Add event>> button. A new screen will be shown to the user where they can enter the details of the new event.
2. After the user clicks OK, the system checks the entered details for errors. If any errors occur, the user is shown a warning and the system returns to the “Add new event” screen.
3. Else, if all the details are correct, the system creates an event object, and adds it to a collection it in the timeline.

Modify event:



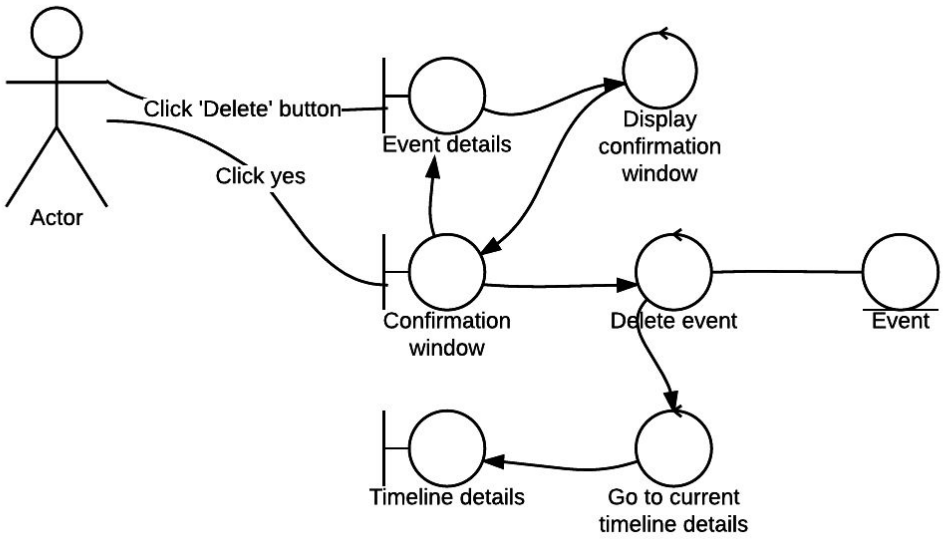
1. The user wants to modify an event. In the event details screen, he clicks the button <<Modify event>>.
2. The user is changes the details of the event, and clicks OK.
3. The system checks whether the details are correct, and then updates the event object.

Delete timeline:



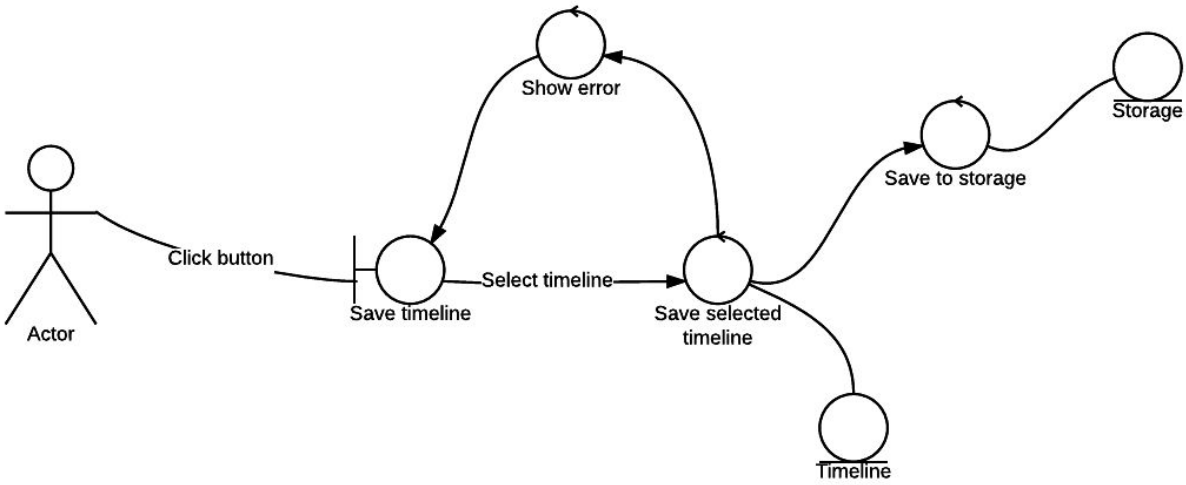
1. At the timeline overview, the user selects the timeline to delete, opens that timeline to go to the timeline details screen. In that screen, he clicks the “Delete timeline” button.
2. A confirmation window is shown to the user. If the user cancels, he is taken back to the timeline details screen.
3. Else, the user clicks OK. The system deletes the timeline object and all its event objects.

Delete event:



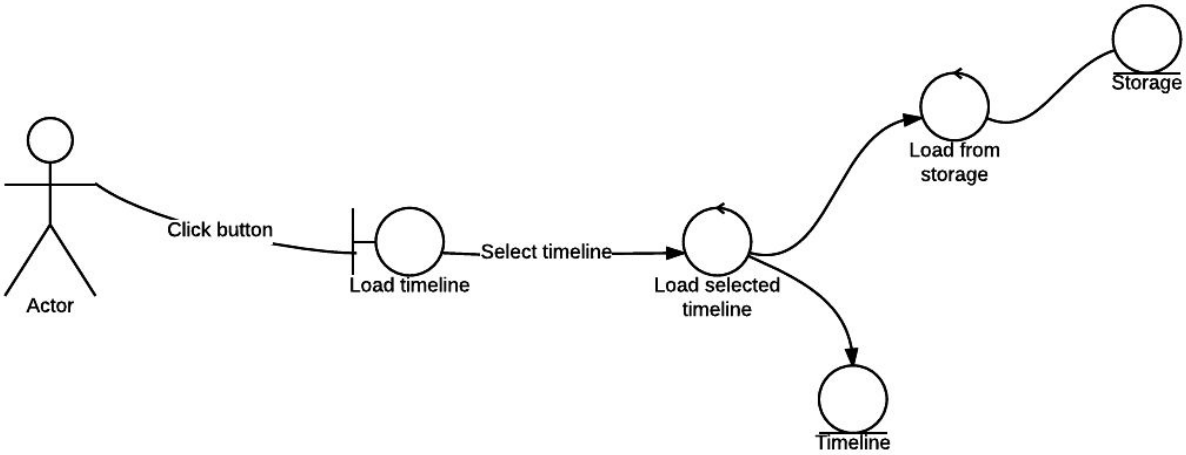
1. The user selects the event that they want to delete. He is taken to the event details window, where he then clicks on the “Delete event” button.
2. The user must confirm the deletion. If the user cancels, the user is taken back to the event details screen.
3. Else, the user clicks OK. The system deletes the event object and takes the user to the timeline details screen.

Save timeline



1. Two ways of saving are possible:
    a. The user selects the timeline to save, right-clicks on that timeline, and clicks “Save timeline”. Then, they are taken to the save screen, where the location of the timeline can be entered.
    b. The user goes to the Menu bar at the top of the window, and clicks File > Save timeline. This populates a list of currently opened timelines. Then, they select the timeline to save. Afterward, they are taken to the save screen, where the location of the timeline can be entered.
2. After the user has entered the location and pressed OK, the system saves the timeline as a datafile to the defined location.

Load timeline:



1. The user goes to File > Load timeline in the Menu bar.
2. A window is opened where the user selects the timeline to be loaded.
3. The system opens the timeline.