Timeline Manager
Implementation Plan

1. Key milestones

Milestone	Date
Iteration start	19/04/2017
Improve use case descriptions & add pre & post conditions.	27/04/2017
Fix FXML issues.	27/04/2017
Implement Add timeline use case.	27/04/2017
Implement Add event use case.	27/04/2017
Implement Save timeline use case.	27/04/2017
Iteration stop	27/04/2017

2. High-level objectives

- Add timeline use case is implemented.
- Add event use case is implemented.
- Save timeline use case is implemented.
- Issues from previous iteration is solved.

3. Work Item assignments

The following Work Items will be addressed in this iteration:

Name or key words of description	Reference material	Assigned to (name)	Hours worked
Improve use case descriptions & add pre & post conditions. Fix test cases.	Make the use case descriptions more robust so it's easier to get a grip on what's expected when implementing it. Example: postcondition when deleting timeline, does it remove a file or just the timeline object?	Milan	
Make the timeline represent a grid.	This means that if a timeline object were to be added to the timeline view it should be drawn as a grid. This can be done by extending the grid class and in the timeline constructor make it take form.	Basel & Henry	
Make form appear when pressing create timeline and when filled in create timeline.	This means when a create timeline button is created a form should appear and when filled in it should create a timeline object.	Gina & Wael	
Fix FXML issues.	Basically remove all the fixed width & height properties of the elements, also	Dimitris	

	some elements are overlapping each other were not needed like the label over the search bar. The fxml is now added to our project so you can run it with java and test your changes.		
Make the model function as expected & add listener to each controller.	Making observable objects in model. Listeners in each controller.	Oskar	
Finish the IO classes if there needs more work.	To make the save timeline use case work just check so these classes are connected and works as intended.	Oskar	
Add unit tests to all classes that allows it.	Self explanatory.	Oskar	

4. Issues

Issue	Status	Notes

5. Evaluation criteria

To be edited….

- **97% of system-level test cases passed.**
- **Walkthrough of iteration build with Departments X and Y received favorable response.**
- **Favorable response to technical demo.**

6. Assessment

[Use this section for capturing and communicating results and actions from assessments, which are typically done at the end of each iteration. If you don't do this, the team may not be able to improve the way they develop software.]

Assessment target	[This could be the entire iteration or just a specific component]
Assessment date	
Participants	
Project status	[For example, express as Red, Yellow, or Green.]

- **Assessment against objectives**

  [Document whether you addressed the objectives as specified in the Iteration Plan.]

- **Work Items: Planned compared to actually completed**

  [Summarize whether all Work Items planned to be addressed in the iteration were addressed, and which Work Items were postponed or added.]

Assessment against Evaluation Criteria Test results

[Document whether you met the evaluation criteria as specified in the Iteration Plan. This could include information such as “Demo for Department X was well-received, with some concerns raised around usability,” or “495 test cases were automated with a 98% pass rate. 9 test cases were deferred because the corresponding Work Items were postponed.”]

- **Other concerns and deviations**

  [List other areas that have been evaluated, such as financials, or schedule deviation, as well as Stakeholder feedback not captured elsewhere.]