Abstract

This document describes the software requirements for the SensUs Digital Platform. The requirements for this Software Requirements Document (SRD) satisfy the requirements in the User Requirements Document[1]. This document also complies with the ESA software standard.
# Contents

1 Introduction
   1.1 Purpose .................................................. 9
   1.2 Scope .................................................. 9
   1.3 Definitions and Abbreviations ......................... 10
      1.3.1 Definitions .................................. 10
      1.3.2 Abbreviations ............................... 14
   1.4 List of references ................................... 14
   1.5 Overview ............................................... 15

2 General Description ....................................... 16
   2.1 Relation to Current Projects ......................... 16
   2.2 Relation to Predecessor and Successor Projects ..... 16
   2.3 Function and Purpose ................................ 16
      2.3.1 Online viewers and Event Attendees .......... 16
      2.3.2 Online Participation Management ............. 17
      2.3.3 Event Management ............................ 17
   2.4 Environment .......................................... 18
   2.5 Relation to Other Systems ........................... 19
      2.5.1 Google OAuth .................................. 19
      2.5.2 YouTube for Videos and Live Streaming ..... 19
      2.5.3 Image Hosters ................................ 19
      2.5.4 Social Networking ............................ 19
      2.5.5 Donation Payment Services .................... 20
   2.6 General constraints .................................. 20
      2.6.1 Usability ....................................... 20
      2.6.2 Security ....................................... 20
      2.6.3 Extendability ................................ 21
      2.6.4 Performance .................................. 21
   2.7 Model Description .................................... 21
      2.7.1 Environment Model ............................ 22
      2.7.2 Class Diagram ................................. 24
      2.7.3 Data Model ..................................... 35
      2.7.4 Model-View-Controller Model .................. 42
      2.7.5 Sequence Diagrams ............................ 43
3 Software requirements

3.1 Functional requirements

3.1.1 Example

3.1.2 User - Unauthenticated User

3.1.3 User - Authenticated User

3.1.4 User - Team Member

3.1.5 User - SensUs Personnel

3.1.6 User - Website Administrator

3.1.7 Team

3.1.8 Team - SensUs Organization

3.1.9 Charity

3.1.10 Question

3.1.11 Answer

3.1.12 Vote

3.1.13 Tag

3.1.14 LiveStream

3.1.15 Measurement

3.1.16 Graph

3.1.17 Video

3.1.18 Picture

3.1.19 Link

3.1.20 Statistics

3.1.21 Mapping

4 Traceability Matrix

4.1 Mapping of Software Requirements to User Requirements

4.2 Mapping of User Requirements to Software Requirements

A Appendix - UI Layouts

A.1 Home Page

A.1.1 Uploading

A.2 Voting

A.2.1 Casting Votes

A.2.2 Voting Statistics

A.2.3 Heat Map

A.3 Explore

A.4 Teams

A.4.1 Team Info Page

A.5 Q&A
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.6 Login</td>
<td>136</td>
</tr>
<tr>
<td>A.7 Dashboard</td>
<td>137</td>
</tr>
<tr>
<td>A.8 Team Dashboard</td>
<td>137</td>
</tr>
<tr>
<td>A.9 Data Dashboard</td>
<td>138</td>
</tr>
<tr>
<td>A.10 Event Dashboard</td>
<td>139</td>
</tr>
<tr>
<td>A.11 Q&amp;A Dashboard</td>
<td>139</td>
</tr>
<tr>
<td>A.12 Control Dashboard</td>
<td>141</td>
</tr>
<tr>
<td>B Transitions</td>
<td>142</td>
</tr>
<tr>
<td>B.1 Unauthorized-user transitions</td>
<td>142</td>
</tr>
<tr>
<td>B.2 Authorized-user transitions</td>
<td>142</td>
</tr>
<tr>
<td>B.3 Diagram</td>
<td>143</td>
</tr>
</tbody>
</table>
DOCUMENT STATUS SHEET

GENERAL

Document title: Software Requirements Document v1.0.2
Identification: SRD/1.0.2
Document status: Draft 1.0.2

DOCUMENT HISTORY

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Author(s)</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0.1</td>
<td>16-05-2017</td>
<td>J.M.A. Boender</td>
<td>Document layout</td>
</tr>
<tr>
<td>0.1.0</td>
<td>16-05-2017</td>
<td>J.M.A. Boender</td>
<td>Draft section 1.</td>
</tr>
<tr>
<td>0.2.0</td>
<td>02-06-2017</td>
<td>R. Morel, C.C. Weibel</td>
<td>Draft Sections 2.1 through 2.6. Draft of Environment model in Section 2.7.</td>
</tr>
<tr>
<td>0.3.0</td>
<td>03-06-2017</td>
<td>R. Morel, C.C. Weibel</td>
<td>Implement feedback from N. Zannone.</td>
</tr>
<tr>
<td>0.5.0</td>
<td>07-06-2017</td>
<td>R. Morel</td>
<td>Draft of Class Diagram in Section 2.7.</td>
</tr>
<tr>
<td>0.6.0</td>
<td>13-06-2017</td>
<td>C.C. Weibel</td>
<td>Draft of UI mockup Appendix and Transitions Appendix.</td>
</tr>
<tr>
<td>0.7.0</td>
<td>15-06-2017</td>
<td>R. Morel, C.C. Weibel</td>
<td>Draft of Section 3 Requirements done.</td>
</tr>
<tr>
<td>0.8.0</td>
<td>16-06-2017</td>
<td>R. Morel</td>
<td>Draft of generated traceability matrix done.</td>
</tr>
<tr>
<td>0.9.0</td>
<td>16-06-2017</td>
<td>R. Morel, C.C. Weibel</td>
<td>Implement feedback from N. Zannone.</td>
</tr>
<tr>
<td>0.10.0</td>
<td>18-06-2017</td>
<td>R. Morel, C.C. Weibel</td>
<td>Draft of ER Diagram done. Complete descriptions for class diagram and ER model.</td>
</tr>
<tr>
<td>Version</td>
<td>Date</td>
<td>Authors</td>
<td>Notes</td>
</tr>
<tr>
<td>---------</td>
<td>------------</td>
<td>-----------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>0.11.0</td>
<td>20-06-2017</td>
<td>R. Morel</td>
<td>Implement feedback from N. Zannone.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C.C. Weibel</td>
<td></td>
</tr>
<tr>
<td>0.12.0</td>
<td>23-06-2017</td>
<td>H.R. Galioulline</td>
<td>Sequence Diagrams done.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>W.M.W.R. Verlaek</td>
<td></td>
</tr>
<tr>
<td>1.0.0</td>
<td>28-06-2017</td>
<td>H.R. Galioulline</td>
<td>Implement feedback from N. Zannone.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R. Morel</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>W.M.W.R. Verlaek</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>C.C. Weibel</td>
<td></td>
</tr>
<tr>
<td>1.0.1</td>
<td>04-07-2017</td>
<td>R. Morel</td>
<td>Revise document according to changes in URD.</td>
</tr>
<tr>
<td>1.0.2</td>
<td>06-07-2017</td>
<td>H.R. Galioulline</td>
<td>Final round of implementing feedback from N. Zannone, W. Verlaek, and J. Ubbink.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R. Morel</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>W.M.W.R. Verlaek</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>C.C. Weibel</td>
<td></td>
</tr>
</tbody>
</table>
## DOCUMENT CHANGE RECORDS

Significant changes since version 1.0.1

<table>
<thead>
<tr>
<th>Section</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.7.5</td>
<td>Fix grammar and usage mistakes in sequence diagram descriptions. Fix oversized diagrams.</td>
</tr>
<tr>
<td>3.1</td>
<td>Add ‘static’ keyword to missing function requirements (making it inline with the class diagram).</td>
</tr>
<tr>
<td>3.1.8</td>
<td>Removed SR-28 as it was a duplicate. SR-28 is now skipped as a requirement identifier.</td>
</tr>
<tr>
<td>4</td>
<td>Regenerate the traceability matrix, in accordance with URD changes.</td>
</tr>
</tbody>
</table>
1 INTRODUCTION

1.1 PURPOSE

This Software Requirements Document (SRD) contains the software requirements for the SensUs Digital Platform. These requirements are a translation of the user requirements found in Section 3 of the User Requirements Document (URD) [1]. The formulation of these user requirements in the URD [1] was a joint effort between SensUs and Valedictorian, while the software requirements in this document will be solely formulated by Valedictorian with approval by SensUs. The URD [1] specified what the SensUs Digital Platform will allow users to do, whereas the SRD will specify how this will be implemented by the SensUs Digital Platform.

1.2 SCOPE

Valedictorian is a group of software engineers working on behalf of the TU/e for SensUs. SensUs is an organization that organizes an annual competition between different universities and colleges across the globe to demonstrate the design and effectiveness of various biosensors. The software system to be developed will be a website, and possibly also a mobile application, that acts as a platform to further engage all interested parties (both attendees and online viewers) in the event.

The main purpose of the SensUs Digital Platform is to provide online viewers with an experience as close as possible to actually attending the event. This will be realized by having live video streams and highlights of past live video streams of events available, along with additional information about the competition on the website. These streaming-related features will also be available on the mobile application. In this manner, online viewers will have access to the same information and ongoings of the events as attendees. Furthermore, for any attendee of the event, the website and mobile application will provide an additional source of information of ongoing activities, the participating teams, the different awards, and the judges. Since it will be impossible to be everywhere at once, this will also allow attendees to catch up with any interesting missed activities. Finally, the website and mobile application will act as a medium to allow online viewers (and offer attendees an additional format) to engage with the participating teams. This engagement will be a mixture of asking the teams interesting questions, sharing videos of the event, and voting for the teams on the different awards. Overall, the SensUs Digital Platform will provide another dimension of engagement for everyone interested in following the competition.
1.3 DEFINITIONS AND ABBREVIATIONS

1.3.1 DEFINITIONS

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Answer Video</td>
<td>A video link containing an answer to a question video.</td>
</tr>
<tr>
<td>Answer Video Status</td>
<td>An answer video can either be rejected, approved or unapproved. That status determines whether a video can be viewed or not.</td>
</tr>
<tr>
<td>Award</td>
<td>A prize that can be won by a team participating in the SensUs competition.</td>
</tr>
<tr>
<td>Biomarker</td>
<td>A molecule, gene, etc. that acts as an indicator of a particular physiological or pathological state of an organism. The biomarker for SensUs 2017 is NT-proBNP. This is a protein that is a biomarker for heart failure.</td>
</tr>
<tr>
<td>Biosensor</td>
<td>A device of a team to measure the concentration of a biomarker in a sample.</td>
</tr>
<tr>
<td>Biosensor data</td>
<td>Data entry for a specific team, containing the following data: the sample number, the sample volume used, the measured concentration, the time-to-result, the measured signal value.</td>
</tr>
<tr>
<td>Chrome</td>
<td>A web browser developed and maintained by Google.</td>
</tr>
<tr>
<td>Correlation Scatter Plot</td>
<td>A scatter plot containing data points mapping the difference between the actual concentration of different samples and the measured concentrations of the samples by each group.</td>
</tr>
<tr>
<td>Dose Response Curve</td>
<td>A graph mapping the dose amount against the response.</td>
</tr>
<tr>
<td>Event Attendee</td>
<td>A user of the SensUs Digital website who is physically present at the SensUs competition event.</td>
</tr>
<tr>
<td>Highlight Video</td>
<td>A short video of a live stream highlighting a certain part.</td>
</tr>
<tr>
<td>iDeal</td>
<td>A payment method that transfers money directly between bank accounts - very popular in the Netherlands.</td>
</tr>
<tr>
<td>Live Video Stream</td>
<td>A video showing live content.</td>
</tr>
<tr>
<td>Logged-in User</td>
<td>A user of the SensUs Digital website who has a Google account associated with them. Has extra permissions to vote on teams and submit videos.</td>
</tr>
<tr>
<td>Main stream</td>
<td>A live video stream designated by SensUs to be the central live video stream of the event.</td>
</tr>
<tr>
<td>Matching Rules</td>
<td>A set of rules on how to properly match concentrations to teams.</td>
</tr>
<tr>
<td>Term</td>
<td>Definition</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Online Viewer</td>
<td>A user of the SensUs Digital website who is not physically present at the SensUs competition event.</td>
</tr>
<tr>
<td>Partial ID</td>
<td>An identifier for a specific “sample concentration” indicating the actual concentration level and measurement number, but not the team it is matched with.</td>
</tr>
<tr>
<td>Partners</td>
<td>Organizations that help sponsor the SensUs event.</td>
</tr>
<tr>
<td>PayPal</td>
<td>A payment system using accounts that are linked either to a bank account or credit card. Accounts can then easily transfer money electronically to each other.</td>
</tr>
<tr>
<td>Pre-approved video</td>
<td>A video shows up immediately on the website after being uploaded without being approved by SensUs personnel. However, a pre-approved can still be rejected and deleted by SensUs personnel.</td>
</tr>
<tr>
<td>Question Tag</td>
<td>A key word associated with a question video to link it to a team or SensUs. This tag will be used to identify which team (or SensUs) the question is addressed to.</td>
</tr>
<tr>
<td>Question Video</td>
<td>A video link containing a question (only shows up on the Q&amp;A dashboard during processing).</td>
</tr>
<tr>
<td>Question Video Status</td>
<td>A video status (pending, rejected, being answered, coupled) determines if the Q&amp;A video shows up in the Q&amp;A page or not, if it is in the process of being answered and if the video is coupled with an Answer video or not.</td>
</tr>
<tr>
<td>Regular Video</td>
<td>A video link that only shows up on the control room dashboard when being processed. This can be a video link to a related topic of biomarkers, a live stream link or some other type of video link related to the competition.</td>
</tr>
<tr>
<td>Regular Video Status</td>
<td>A video can either be rejected, approved or unapproved. That status determines whether a video can be viewed or not.</td>
</tr>
<tr>
<td>Regular Video Tags</td>
<td>A tag that categorizes the video based on the type of content and determines under which filters the video will show up.</td>
</tr>
<tr>
<td>Reserve Sample</td>
<td>A sample not associated with a specific team. Will be supplied to a team if necessary.</td>
</tr>
<tr>
<td>Role</td>
<td>Short for &quot;User Role&quot; - definition listed below.</td>
</tr>
<tr>
<td>Term</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Sample</td>
<td>A substance containing a certain concentration of a biomarker. During the competition, samples are provided to the teams, and the teams test the samples in their biosensors. The teams do not know the concentration of biomarker in their samples.</td>
</tr>
<tr>
<td>Sample Database</td>
<td>A database containing information associated with each of the samples. This information includes values such as the actual concentrations of each sample, the team the sample is assigned to and any recorded measurements (by a team) of each sample.</td>
</tr>
<tr>
<td>SensUs Backend</td>
<td>The component in the environment that primary deals with data access.</td>
</tr>
<tr>
<td>SensUs Frontend</td>
<td>The component in the environment that is primary concerned with the presentation layer.</td>
</tr>
<tr>
<td>SensUs Member</td>
<td>A member of the SensUs organization involved in organizing the SensUs event.</td>
</tr>
<tr>
<td>SensUs Personnel</td>
<td>A user of the SensUs Digital website who is associated with SensUs, has all the permissions of a team member as well as some additional permissions.</td>
</tr>
<tr>
<td>SensUs-Related Social Media Activity</td>
<td>Mentions, hashtags and posts of SensUs on all social media platforms.</td>
</tr>
<tr>
<td>Stable</td>
<td>The behavior, functionality, specification or API is considered 'final' for that version (no more changes).</td>
</tr>
<tr>
<td>Stock team photo</td>
<td>The photo that represents the team as a whole.</td>
</tr>
<tr>
<td>Submitted Photo</td>
<td>A photo submitted by an authenticated member to be shown on the SensUs Digital website.</td>
</tr>
<tr>
<td>Summary Graph of Biosensor Data</td>
<td>A correlation scatter plot where the x-axis is real concentration and the y-axis is measured concentration over all teams' biosensor data.</td>
</tr>
<tr>
<td>Text Answer</td>
<td>A text answer submitted by a team member to a question (question could be either text or video).</td>
</tr>
<tr>
<td>Text Answer Status</td>
<td>An text answer can either be rejected, approved or unapproved. That status determines whether the answer can be viewed or not (paired with the question).</td>
</tr>
<tr>
<td>Team Member</td>
<td>A user of the SensUs Digital website who is associated with a participating team and has additional permissions relative to a logged-in User.</td>
</tr>
<tr>
<td>Team Photo</td>
<td>A photo submitted by a team member to be shown on the photo gallery.</td>
</tr>
<tr>
<td>Term</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Team Photo Status</td>
<td>A team photo submitted by a team member can either be rejected, approved or unapproved. That status determines whether the team photo can be viewed or not.</td>
</tr>
<tr>
<td>Team Poster</td>
<td>A poster supplied by the team for the SensUs competition.</td>
</tr>
<tr>
<td>Team Summary Information</td>
<td>A compact summary stating the name, logo, picture, and country of origin for each participating teams.</td>
</tr>
<tr>
<td>Text Question</td>
<td>A question submitted by a user to a specific team. The team the question is directed to will be handled in the tag system.</td>
</tr>
<tr>
<td>Text Question Status</td>
<td>A status (pending, rejected, being answered, coupled) determines if the Q&amp;A text question has been answered or not.</td>
</tr>
<tr>
<td>Ubuntu</td>
<td>A distribution of the Linux operation system that can be used to run web servers.</td>
</tr>
<tr>
<td>Unauthenticated User</td>
<td>An online viewer or event attendee using the SensUs Digital website who has the lowest level of permission. Some features of the SensUs Digital website are not available to this user until the user becomes authenticated.</td>
</tr>
<tr>
<td>User Role</td>
<td>The different roles a user can have in the SensUs Digital website. These include: unauthenticated user, authenticated user, SensUs Personnel, or Web Administrator. All these are treated in more detail in section 2.4 of this document.</td>
</tr>
<tr>
<td>Video</td>
<td>Either a question video or a regular video.</td>
</tr>
<tr>
<td>Video State</td>
<td>A video state determines if that video is a question video, answer video or regular video.</td>
</tr>
<tr>
<td>Viewing statistics</td>
<td>The total number of views across all videos and the location of those views.</td>
</tr>
<tr>
<td>Viewing statistics status</td>
<td>Can either be &quot;visible&quot; or &quot;invisible&quot;.</td>
</tr>
<tr>
<td>Voting Heatmap</td>
<td>A graphical representation of voting data across regions on the globe.</td>
</tr>
<tr>
<td>Website administrator</td>
<td>A user of the SensUs Digital website who is has the highest level of permissions. Everything that SensUs personnel can do, a website administrator can also do.</td>
</tr>
</tbody>
</table>
1.3.2 ABBREVIATIONS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESA</td>
<td>European Space Agency</td>
</tr>
<tr>
<td>TU/e</td>
<td>Eindhoven University of Technology</td>
</tr>
<tr>
<td>URD</td>
<td>User Requirements Document</td>
</tr>
<tr>
<td>SRD</td>
<td>Software Requirements Document</td>
</tr>
<tr>
<td>URL</td>
<td>Uniform Resource Locator</td>
</tr>
<tr>
<td>Q&amp;A</td>
<td>Question and Answer</td>
</tr>
</tbody>
</table>

1.4 LIST OF REFERENCES


1.5 OVERVIEW

The remainder of this document is organized into three chapters. The first chapter (Chapter 2) is the general description of the SensUs Digital Platform. The general description will be split into seven subsections describing the different aspects of the SensUs Digital Platform. Section 2.1 will be about the relation of this project to other current projects, and Section 2.2 will contain the context of this project in relation to predecessor and successor projects. Section 2.3 will be about the general functionality and purpose of the SensUs Digital Platform. This will give an overview and context for the main categories and specifics of the requirements. In Section 2.4, the operational environment will be detailed. Section 2.5 will be about how the SensUs Digital Platform relates to other existing systems. Next, Section 2.6 will be about the general constraints that will have to be taken into account. Section 2.7 will go over the logical model of the SensUs Digital Platform. Altogether, Chapter 2 will set the stage and context to help understand the specific requirements in the next chapter. Chapter 3 will be the section containing the functional requirements. This will be the formal description of exactly how the system will satisfy the users requirements contained in the URD [1]. In order to make sure that we do meet all necessary requirements in the URD [1], Chapter 4 will contain a traceability matrix matching each of the functional requirements to its corresponding user requirement in the URD [1]. In this way, we can organize and easily keep track of all the user requirements in the URD [1] that have been implemented by requirements in this SRD.
2 GENERAL DESCRIPTION

2.1 RELATION TO CURRENT PROJECTS

The SensUs Digital Platform is a project specifically designed for the SensUs competition. The SensUs Digital Platform is not based on any one specific system, but rather draws inspiration from different components of other social websites (such as Facebook, Twitter and Reddit). The combination of these platforms and their components is not encompassed by a comparable event-monitoring, live-streaming, donation-handling system. In terms of core functionality, the SensUs Digital Platform could be hypothetically replaced by a pre-existing system, that would offer similar critical functionality (such as Facebook or YouTube) but not contain all features that the SensUs Digital Platform would offer.

2.2 RELATION TO PREDECESSOR AND SUCCESSOR PROJECTS

The SensUs Digital Platform’s only significant related project is the SensUs contest which took place in 2016. After the contest, it became apparent for the need of a platform to manage all aspects of the event. There is, of course, no relation to successors due to the SensUs Digital Platform still being in its nascency. SensUs has made it clear that they intend to use the SensUs Digital Platform in the following years. Because of this, the SensUs Digital Platform has to be designed with both scalability and extensibility in mind. Furthermore, it is possible that SensUs Digital Platform will extend beyond the initial functionality implemented by Valedictorian in the following years.

2.3 FUNCTION AND PURPOSE

The SensUs Digital Platform will act as the main system to not only help all people interested in the SensUs competition find out more and participate, but also act as a tool to help SensUs personnel manage the people participating online and the event as a whole. From this perspective, we can categorize the purpose of all functionalities into three categories: providing interested viewers and event attendees with information, providing SensUs personnel with tools to manage online participation/interaction, and provide SensUs personnel with tools for management of the event as a whole. Brief descriptions of the functionality the SensUs Digital Platform will need to satisfy are given below:

2.3.1 ONLINE VIEWERS AND EVENT ATTENDEES

- *Real-time Information*: Both online viewers and event attendees will have access to real-time data about the current event. This way they will be able to keep up-to-date on the latest results of the competition and other activities happening during the event.
• **Background Information**: Both online viewers and event attendees will be able to see static information related to the event. This will allow curious viewers and attendees to find out more about the event and who was involved.

• **Playback**: Online viewers and event attendees will be able to play back videos, which will be recordings of the event itself and videos submitted by participating users. The videos will be available throughout the SensUs Digital website.

• **Voting**: Online viewers and event attendees will be able to vote for their teams of choice.

• **Donation**: Online viewers and event attendees will be able to donate to their team of choice.

• **Mobile Application**: Online viewers and event attendees will be able to use a mobile application if they are using a mobile device, which will offer limited functionality relative to the SensUs Digital website.

### 2.3.2 ONLINE PARTICIPATION MANAGEMENT

• **Social Media**: SensUs Personnel will be able to check social media mentions about the SensUs event and respond directly through the SensUs Digital website to these mentions.

• **Monitoring Data**: SensUs Personnel will have access to the status and workload information of server running the SensUs Digital website. This will hopefully allow SensUs Personnel to realize when the SensUs Digital website is close to failure, and enable them to take evasive action.

• **Video Management**: SensUs Personnel will be able to monitor videos uploaded to the SensUs Digital website in order to make sure only appropriate submissions make it to the SensUs Digital website.

• **Q&A**: SensUs Personnel will be able to monitor all questions and answers submitted. Again, this will ensure only appropriate questions and answers get displayed on the SensUs Digital website.

### 2.3.3 EVENT MANAGEMENT

• **Biosensor Data**: SensUs Personnel will be able to enter biosensor data gathered in order to keep all live graphs up-to-date. Since this data has to be manually entered, these graphs will be considered only “semi-live,” i.e. there might be significant delays before the gathered data is available for online viewers.

• **Technical Evaluation**: SensUs Personnel will have a tool to automate the matching between samples and teams. This will be essential in the organization of the competition, and will have a significant impact on the time it takes to set up this part of the event.
In this way, the SensUs Digital Platform will have a three-fold role during the SensUs competition and will be absolutely critical to its smooth management and successful execution.

## 2.4 ENVIRONMENT

The SensUs Digital Platform is made up of the SensUs Digital website and the SensUs Digital mobile application. Both the SensUs Digital website and SensUs Digital mobile application have been designed to run on most modern web browsers and most modern mobile operating systems respectively:

- **SensUs Digital website**: The browsers supported are Google Chrome v57, FireFox v52, Safari v8, Internet Explorer v11, and Microsoft Edge v40.

- **SensUs Digital mobile application**: The mobile operating systems supported are Android 4.1 and higher, and iOS 7.0 and higher, and it will run in portrait mode.

The SensUs Digital Platform is to be operated by the following types of users:

- **Unauthenticated User**: People who visit the SensUs Digital website without logging in. They will be able to browse most of the website content.

- **Authenticated User**: People who visit the website and log in. They will have the added functionality to vote for teams, submit questions, post pictures, and post videos.

- **Team Member**: Members of participating teams at the event. They will have access to the same functionality as an Authenticated User, with additional access to the custom tools for team-specific functions.

- **SensUs Personnel**: Party trusted by the SensUs organization running the competition. SensUs personnel will have all permissions that team members will have with the addition of being able to use a set of private dashboards to manage the event and user interaction.

- **Website Administrator**: Works with SensUs to maintain the SensUs Digital website. They will have all permissions of SensUs personnel with the additional ability to change user roles (to authenticated users, team members, SensUs personnel and website administrators). They will also have their own dashboard to manage the administration. Data entry duties will also be relegated to administrators.

- **Mobile User**: An online viewer or event attendee using the SensUs Digital mobile application to access the SensUs Digital Platform. The SensUs Digital mobile application will have a subset of the functionality of the SensUs Digital website. The SensUs Digital mobile application will only act as an accessory to the SensUs Digital website.

A detailed description of these users can be found in the URD [1] in sections 2.4.2 - 2.4.7.
2.5 RELATION TO OTHER SYSTEMS

The SensUs Digital Platform is stand-alone and does not relate to any other system developed by SensUs, however it does connect to a number of third-parties for services such as social networking, live streaming, etc. As the SensUs Digital Platform is a new concept, and it was not used in the SensUs contest of the previous year, no other systems depend upon SensUs Digital Platform. This is subject to change should the SensUs Digital Platform be extended in the times to come.

2.5.1 GOOGLE OAUTH

Instead of directly implementing and supporting authentication of user accounts we will use the OAuth service of Google to try to limit the number of accounts that can be associated with a person using the site, in order to minimize spam, and double voting. Using OAuth and restricting it to a single provider makes for easy implementation while only using Google’s service makes it more difficult for people to use multiple accounts to, for example, vote twice.

2.5.2 YOUTUBE FOR VIDEOS AND LIVE STREAMING

In order to avoid the need to create our own live-streaming system, we have opted to use YouTube’s existing live-streaming service. We will be embedding links to the relevant live streams throughout the SensUs Digital website. This saves a significant amount of time and money, while providing a proven solution that will be able to scale to many more users in the future. In addition we allow users to upload videos associated with the event. We selected YouTube as the main video hosting platform we want to support due to its dominance among video hosting services and the familiarity most users will have with YouTube.

2.5.3 IMAGE HOSTERS

To support users uploading images associated with the event we chose to support direct links to images hosted elsewhere, thereby avoiding the burden of storing images in our database. Thus, it is the user’s responsibility to find a suitable image hosting platform. For user convience we intend to integrate with the API of the Imgur hosting service, allowing direct uploading of images through the interface of the SensUs Digital Platform.

2.5.4 SOCIAL NETWORKING

The social networks we provide integration for are Facebook, Twitter and LinkedIn. Each of these services provide an API that allows us to display all messages from and to, the SensUs account from the respective social network. Since all three of these social networks are popular platforms (in 2017, at the time of writing the current document), by integrating their services into the SensUs Digital website we intend to be able to reach a wider audience.
2.5.5 DONATION PAYMENT SERVICES

To implement donations we will need money transfer services. For Dutch users we selected iDeal as the most appropriate money transfer service, as it is supported by all major banks in the Netherlands. For international donors we consider integration with PayPal to be sufficient, as it is the biggest international payment service.

2.6 GENERAL CONSTRAINTS

The SensUs Digital Platform should be easily utilized by all the users listed in section 2.4. It is especially important that authenticated/unauthenticated users can navigate the website with ease because they represent the public viewers (whose attention is critical to the success of the event). To avoid voter fraud, security is of utmost importance, as well as extensibility to allow SensUs to expand upon the SensUs Digital Platform in the future. We will expand on each of these important constraints, starting with usability.

2.6.1 USABILITY

The usability of the website is a main priority because the viewer base for the event will include a wide range of people, both technical and non-technical. To this end, we have a usability requirement that a user must be able to find the desired content they are searching for within a minimum time of five minutes. Ideally, the user should be able to find the content within a minute, which also gets translated to a requirement but at a lower priority level. We will achieve a high degree of usability by keeping the website relatively clean and simple, with minimalist-style user interfaces.

2.6.2 SECURITY

Security is important for a number of reasons, among them being able to prevent users from voting multiple times via multiple login, etc. and to ensure pages are only accessed by users with the proper authorization. Each user that wishes to access more functionality than provided to the unauthenticated users will be required to log in with their Google account. This will prevent DDoS-style uploading behaviour as each user must link an account to the website and hence it comes with a risk to behave inappropriately. By requiring an account associated to be associated to any user submitting to the website it is also possible to ban users consistently uploading inappropriate content. Not only will using the Google authentication services make implementation easy, but also it will mean that we do not need to store any passwords due to the OAuth 2.0 authentication scheme allowing users to be authenticated through a third party (Google).
2.6.3 EXTENDABILITY

As the SensUs contest is intended to be repeated yearly and SensUs envisions that the SensUs Digital Platform will be extended in the future, extendability through modular code is paramount to future success of the SensUs Digital Platform. Due to this, the SensUs Digital Platform is being developed using React with Redux and Laravel. The structure these frameworks provide allows for easy extendability of the platform.

Front-end extendability

The front-end of the SensUs Digital Platform is being developed in React with Redux. React allows us to easily add, remove, or swap out components, without affecting the rest of the application. This modular approach unquestionably makes for lower effort extensibility.

Back-end extendability

The back-end of the SensUs Digital Platform is being developed using Laravel. This framework allows us to easily separate our front-end and back-end code by using a templating library. Having a clear separation between font-end and back-end code means that we can usually edit one without affecting the other. Adding new functionality to the back-end is also quite simple, because Laravel provides a routing system that is meant to be extensible.

2.6.4 PERFORMANCE

The client SensUs is planning to use the SensUs Digital Platform with thousands of users accessing content concurrently. To achieve a reasonable level of scalability, along with usability, it is necessary for the SensUs Digital website to maximize performance (such as page loading times) to reduce server loads. We will do this by loading all of the SensUs Digital website resources for the website upon visiting the page. This corresponds to the one-page web application concept, which will minimize the waiting times for switching among pages and thereby keep latency low.

2.7 MODEL DESCRIPTION

In order to be able to give complete and precise requirements for the high-level design of the system, we will use several models to illustrate this high-level design. In this section, we will go over the environment model, the class diagram, Entity Relationship model and the sequence diagrams.
2.7.1 ENVIRONMENT MODEL

The following diagram depicts the logical components of the SensUs Digital Platform, as well as the important services the platform will need to be able to interoperate with. Each logical unit is described below.

FIGURE 1: ENVIRONMENT MODEL.

SENSUS DIGITAL WEBSITE

This will be the main component of the SensUs Digital Platform. This will be what the front-end users see when requesting content. The modularization between the SensUs Digital website and the back-end allows for the SensUs Digital website and SensUs Digital mobile application to access the same back-end logic and database. The main users for this component will be the unauthenticated and authenticated users. Based on the hierarchy of users, team members, SensUs personnel and website administrators will also be able to use this component, even though they are not the primary target. Also, this will be the main component that will interact with all social media API’s, the streaming database and the Imgur API. The other components will also have access to these API’s, however most of the calls will be made from this component.

SENSUS MOBILE APPLICATION

This will be an accessory component of the SensUs Digital Platform. Because of modularization between the SensUs Digital website and the back-end, the SensUs Digital mobile application will have access to the same back-end logic and database. This makes developing the SensUs Digital mobile application significantly easier. However, the mobile users will only have access to the SensUs Digital mobile application and thus only have that limited set of functionality.
SENSUS DIGITAL DASHBOARD

The SensUs Digital Dashboard is a separate view within the SensUs Digital website. The reason it is separate from all other views in the website is that only SensUs personnel and team members have access to it (with SensUs personnel having more options than team members). However, SensUs personnel and team members will be able to navigate back and forth between the dashboard and the SensUs Digital website (and the user hierarchy will have all relevant functionality available).

BACK-END DATABASE

The database will be the component that stores all video links, votes, descriptions and other data displayed on the website. Because this will contain all important data, it will be one of the most critical components of the system.

BACK-END LOGIC AND LARAVEL

The back-end logic will govern the exchange between the front and back-end of the SensUs Digital Platform. In order to improve database security (e.g. protect against SQL injections, cross site scripting attacks, et cetera), the logic will be built on top of the Laravel framework and built in security features to formulate queries and process responses from queries. This will make dealing with the database much easier.

SOCIAL MEDIA API

To facilitate more user engagement through social media, we make use of three popular social media platforms. We will use three APIs to embed Facebook feeds, Twitter feeds and LinkedIn feeds on the SensUs Digital website. Feeds from all these social media platforms will be used to show users the current social activity going on of the competition. And they will be connected to the front-end because all the calls will be made from components in that category.

STREAMING API

Using an external service for live video streaming removes the need to invest in large amounts of networking and storage infrastructure. This API will be used to embed live video stream content. And depending on the streaming platform used this could be either YouTube, Twitch, or some other service. This once again will used by the front-end when a user requests a video to be played.

IMGUR API

To avoid storing images on the platform itself, we will rely on an external service. Imgur is a website that allows for free uploading, storage and retrieval of any appropriate images. We
plan on using an API to embed Imgur images on the SensUs Digital website. Again, any calls made to the Imgur API will be done from the front-end (since the back-end has nothing to do with retrieving images, only storing links).

**GOOGLE OAuth API**

To simplify user verification, we will use Google’s authentication services. This API will be used to allow the SensUs Digital Platform to verify the user’s identity through his or her Google account when logging in. This will be used by the backend logic when verifying users who want to authenticate themselves, or do any other action that requires some type of authorization.

**PAYMENT API**

This will include the iDeal API and PayPal API in order to allow the SensUs Digital website to accept donations from users. These two payment methods were chosen because they are the most common nationally and internationally respectively. iDeal is the most common payment method in the Netherlands, and PayPal is common internationally. In this way, the majority of users will be able to make a donation easily. The APIs will be called from the logic in the back end because this is where the controller will be to facilitate the transaction.

**2.7.2 CLASS DIAGRAM**

This section describes the logical view of the SensUs Digital Platform. The model that we use is the class diagram in Figure 2. The attributes and the operations of each entity are described in this section. In Section 3 of this document, the formal requirements on each of the attributes and operations are stated.
User - Unauthenticated User

The Unauthenticated User class embodies the basic user who has yet to log into the SensUs SensUs Digital website.

- `login(email): Void`
  Performs the login procedure of OAuth and stores in the back end that the user is now logged in.

- `shareEvent(service): Void`
  A function that allows the user to share the event to one of the above social networks.

User - Authenticated User

The Authenticated User class embodies the user who has logged into the SensUs SensUs Digital website. These users gain additional functionality over an Unauthenticated User and thus have additional attributes and methods associated with them.

- `name: String`
  The name of the user, as identified by Google OAuth.

- `email: String`
  The email of the user, as required by Google OAuth.

- `remember_token: String`
  The cookie identifier used by Google OAuth to keep a user logged in.

- `banned: Boolean`
  A Boolean identifying whether the user has been banned or not.

- `logout(): Void`
  A function that perform the logout procedure and modifies the back end to reflect that the user is logged out.

- `castVote(teamA, teamB): Void`
  A function that casts/updates a vote for two teams, for this particular user. If the user has already cast their vote in the past their vote gets updated.

- `endorseQuestion(question, state): Void`
  A function whereby a user endorses a question. Is idempotent.
**User - Team Member**

The Team Member User represents a user who will be part of one of the competing teams during the SensUs competition. They will have functionalities specific to biosensor data along with the standard Authenticated User functionality.

- *addComment(question, text, state): Void* A function whereby a team member can add/remove a comment to/from a question.

**User - SensUs Personnel**

The SensUs Personnel class represents a SensUs employee who will be involved with monitoring/handling specific parts of the SensUs Digital website, such as answering questions, but they cannot perform all the actions of the Website Administrator.

- *ban(user, banState): Void* A function that either bans or unbans the user from logging in again.

**User - Website Administrator**

The Website Administrator represents a user who will be all aspects of the SensUs SensUs Digital website. They will have access to all functionalities.

- *changeRole(user, role): Void* A function that changes the user’s role to role, thereby giving or taking capabilities from the user.

**Team**

The Team represents one of the teams who will be part of the SensUs competition.

- *name: String.* The name of the team.
- *picture: URL.* A URL linking to a picture of the team.
- *country: String.* The name of the team's country of origin.
- *university: String.* The name of the team's university.
- *description: String.* The description for the team, which serves as its background information.
Team - SensUs Organization

The SensUs Organization Team represents SensUs as a whole. Viewing the SensUs organization as a Team allows for questions/media to be directed at SensUs as if they were another team. Note that for the SensUsOrganization class there will only be one instance.

- **introVideo**: URL. The link to the introduction video for the SensUs organization.
- **memberInfo**: [(String, URL)]. An array containing the biographical information and a picture of each member of the SensUs organization.
- **eventProgram**: [String]. An array containing the event program for each day.
- **judges**: [(String, URL)]. An array containing the biographical information and a picture of each judge for the SensUs Digital event.
- **partners**: [(String, URL)]. An array containing the biographical information of and a link to each partner for the SensUs Digital event.
- **awards**: [String]. An array containing the descriptions for each award.
- **getSocialActivities()**: [SocialMessage] A function that retrieves all the social media messages associated with the SensUs organization. The social networks supported are Facebook, Twitter, and LinkedIn.
Charity

The Charity class encompasses donations, donation stats, and information specific to the charity the class represents.

- **name**: String.
  The name of the charity.

- **amountRaised**: (Number, Number).
  The total amount raised for this charity and by the corresponding team.

- **update(newName)**: Void
  A function that updates the name for the Charity.

- **sendMoney(serviceProvider, teamAmount, charityAmount)**: Void
  A function that executes the donation action of a user, using the `serviceProvider` specified, with the donation directed to both a `Team` and the team's `Charity`.

Question

The Question class represents any question users pose towards a specific team and submit via the SensUs SensUs Digital website.

- **title**: String.
  The title of the question.

- **description**: String.
  The description of the question.

- **status**: QuestionStatus.
  The status containing whether the answer is Unapproved, Approved, Rejected, Coupled-Approved, and Coupled-Unapproved.

- **date**: Date.
  The date and time of when this question was posted.

- **post(title, desc, link, team)**: Void
  A function that posts a given question to the database.

- **update(status, team)**: Void
  A function that updates a given question's status and associated team.

- **addAnswer(answer)**: Void
  A function that couples an answer to the question.

- **delete()**: Void
  A function that deletes a given question from the database.
**Answer**

The Answer class represents the response given to a given question of the Question class. The selected team at which the question was pointed will be able to create an answer and link it to the corresponding question.

- **title**: String.
  The title of the answer.

- **description**: String.
  The description of the answer.

- **status**: AnswerStatus.
  The status containing whether the answer has been answered or not.

- **update(status)**: Void
  A function that updates a given answer’s status.

- **delete()**: Void
  A function that deletes a given answer from the database.

**Vote**

The Vote class contains all of the statistics associated with users voting for specific teams.

- **statisticsVisible**: Boolean.
  Static attribute, shared among all votes, that sets whether the viewer statistics for votes should be visible.

- **getTotals()**: [(Team, Number)]
  A function that returns the count statistics per team.

- **setStatVisibility(statVis)**: Void
  A function that modifies the statistics visibility state for all votes.

**Tag**

The Tag class trivially contains the tag name of the given media.

- **name**: String.
  The name of the tag.
**LiveStream**

The LiveStream class represents the video being streamed from the event to be accessed by users on the SensUs SensUs Digital website.

- **title**: String.
  The title of the live stream.

- **description**: String.
  The textual description of the live stream.

- **amountViewers()**: Number
  A function that retrieves the current viewer amount from the livestream hoster.

- **update()**: Void
  A function that updates a given live stream's title or description.

**Measurement**

The Measurement class encompassed the data points collected by the biosensors of each team. This data will be graphed in a number of different locations across the website.

- **number**: Number.
  The number associated with the sample.

- **assignedConcentration**: Number.
  The concentration assigned to the sample.

- **sampleConcentration**: Number.
  The concentration of the specific sample.

- **sampleVolume**: Number.
  The volume of the specific sample.

- **timeToResult**: Number.
  The time to attain the measurement result.

- **signalValue**: Number.
  The value ascertained by the signal of a specific biosensor.

- **reportedConcentration**: Number.
  The concentration measured by the teams (different from actual concentration).

- **update(num, assCon, samCon, volume, timeToResult, signal, repCon)**: Void
  A function that updates a Measurement entry.
• **delete()**: Void
  A function that deletes a Measurement entry

**Graph**

A Graph contains data points to be plotted on a set of axis. This data has statistical properties such as standard deviations.

• **type**: GraphType.
  The type of this graph.

• **visible**: Boolean.
  Flag that determines whether the graph is visible or not.

• **axisVisible** (Boolean, Boolean).
  Two flags that determine whether the x-axis and the y-axis are visible, respectively.

• **getDataPoints()**: [Measurement]
  A function that retrieves the measurement data used by this graph. Can be used to count the number of data points.

• **correlation()**: Number
  A function that returns the correlation for the data points.

• **getAverage()**: Number
  A function that returns the average for the data points.

• **getStdDev()**: Number
  A function that returns the standard deviation for the data points.

• **setVisibility(visState, xState, yState)**: Void
  A function modifies the visibility states for a graph.

**Video**

The Video class represents the videos that will be uploaded to the SensUs SensUs Digital website that will be displayed primarily on the Explore page.

• **statisticsVisible**: Boolean.
  Static attribute, shared among all videos, that sets whether the view counts and current number of viewer statistics should be visible.

• **title**: String.
  The title of the video.
• **description**: String.
   The description of the video.

• **status**: MediaStatus.
   The status of a given media.

• **post**(title, desc, link, status): Void
   A function that posts a given video to the database. The status argument should be Unapproved when not submitted by SensUsPersonnel.

• **update**(title, desc, link, status, tags): Void
   A function that updates a given video in the database.

• **delete**(): Void
   A function that deletes a given video from the database.

• **setStatVisibility**(statVis): Void
   A function modifies the statistics visibility state for all videos.

• **filterTeam**(team): Void
   A function filters all the videos for the team.

**Picture**

The Picture class represents the pictures that will be uploaded to the SensUs SensUs Digital website that will be displayed primarily on the Explore page.

• **title**: String.
   The title of the picture.

• **status**: MediaStatus.
   The status of the picture.

• **post**(title, link): Void
   A function that posts a given picture to the database.

• **update**(title, status, link): Void
   A function that updates a given picture's title or status.

• **delete**(): Void
   A function that deletes a given picture from the database.
**Link**

A Link is a symbolic connection to external media content.

- **sourceType**: SourceType.
  
The source of the link, whether it is

- **contentId**: String.
  
The identifier unique to the source linking to the media content, e.g. the identifier for a YouTube video.

**Statistics**

The Statistics class contains up-to-date stats relevant to the SensUs Digital website.

- **numVisitors**: Number.
  
The amount of visitors that have accessed the SensUs Digital website.

- **uptime**: Number.
  
The uptime of the server hosting the SensUs Digital website

- **respTime**: Number.
  
The average response time of the SensUs Digital website.

- **availableRAM**: Number.
  
The available RAM (in bytes) of the server hosting the SensUs Digital website.

**Mapping**

A Mapping is a legend linking concentration values to a specific sample IDs.

- **associatedTeam**: String.
  
The team associated with the mapping.

- **sampleIDs**: [Number].
  
The set of IDs associated with the samples to be measured.

- **concentrationValues**: [Number].
  
The set of concentration values to be linked with the corresponding sample IDs.

- **currentMapping**: [(Number, Number)].
  
The current mapping.

- **reserveSampleConcentrations**: [(Number, [Number])].
  
The reserve sample concentrations used by each team.
• `generateMapping(sampleIDs, concentrations): Mapping`
  A function that generates mappings from a set of sample IDs and concentration values according to mapping rules for generation.

• `isValidMapping(): Boolean`
  A function that checks if the current mapping satisfies the mapping rules.

2.7.3 DATA MODEL

The data model describes what data we will be storing and how that data is structured. Our data model is illustrated as a Entity Relationship diagram, with crowfoot notation, in Figure 3. Note that this model is merely a representation of the data structure that complies with the URD [1], but this does not imply the full setup of the eventual database. What follows is a description of all the entities in our model with an explanation of their attributes.

UnauthenticatedUser

An UnauthenticatedUser does not have any data associated with it. Such a user only has simple capabilities, not expressed in the data model, such as viewing the site, logging in, and sharing on social media.

AuthenticatedUser

To support user authentication and authorization, the AuthenticatedUser entity contains all the data that defines a user that is not anonymous.

• `name: String`.
  The name of the user, as identified by Google OAuth.

• `email: String`.
  The email of the user, as required by Google OAuth.

• `rememberToken: String`.
  The cookie identifier used by Google OAuth to keep a user logged in.

• `banned: Boolean`.
  A Boolean identifying whether the user has been banned or not.

TeamMember

A TeamMember does not carry additional data versus an AuthenticatedUser, except for the user's relation to a team.

**SensUsPersonnel**

A SensUsPersonnel only needs to be able be distinguished from the other users, such that the role of the user can be checked as part of the preconditions for actions, but carries the exact same as an AuthenticatedUser.
WebsiteAdministrator

The WebsiteAdministrator has the data layout of an AuthenticatedUser, except that it must have a distinguishable role from its superclass (when implemented).

Team

The Team entity stores the identifying name of a team along with descriptive information for interested viewers.

- **name**: String.
  The name of the team.

- **picture**: URL.
  A URL linking to a picture of the team.

- **country**: String.
  The name of the team’s country of origin.

- **university**: String.
  The name of the team’s university.

- **description**: String.
  The description for the team, which serves as its background information.

- **poster**: URL.
  The link to the Team’s poster.

SensUsOrganization

The SensUsOrganization is in many of the same relations as a Team and hence derives from it. It stores a team’s information as well as the organizational information about the event.

- **introVideo**: URL.
  The link to the introduction video for the SensUs organization.

- **memberInfo**: [(String, URL)].
  An array containing the biographical information and a picture of each member of the SensUs organization.

- **eventProgram**: [String].
  An array containing the event program for each day.

- **judges**: [(String, URL)].
  An array containing the biographical information and a picture of each judge for the SensUs Digital event.
• **partners**: [(String, URL)].
  An array containing the biographical information of and a link to each partner for the SensUs Digital event.

• **awards**: [String].
  An array containing the descriptions for each award.

**Charity**

A basic representation of a Charity where only the name of the charity and the money that has been raised are being stored. Any additional information would depend on the payment provider (not modeled in this document).

• **name**: String.
  The name of the charity.

• **amountRaised**: (Number, Number).
  The total amount raised for this charity and by the associated team.

**Question**

A Question entity contains data on the question being posed as well as some basic statistical data.

• **title**: String.
  The title of the question.

• **description**: String.
  The description of the question.

• **status**: QuestionStatus.
  The status containing whether the answer is Unapproved, Approved, and Rejected, Coupled-Approved, Coupled-Unapproved.

• **date**: Date.
  The date and time of when this question was posted.

**Answer**

The Answer entities carry the answer themselves as well as a indicator whether the answer should be user visible or not, which can be changed by team member users (and users higher up in the user hierarchy).
• **title**: String.
The title of the answer.

• **description**: String.
The description of the answer.

• **status**: AnswerStatus.
The status containing whether the answer has been answered or not.

**Vote**

A *Vote* carries no data by itself, the importance of the entity comes from linking a user to exactly two teams, which constitutes a vote.

**Tag**

A *Tag* is a basic descriptive label for used in relation with multiple entities to indicate the type of the content for that entity.

• **name**: String.
The name of the tag.

**LiveStream**

The *LiveStream* entities store basic descriptive information and have their actual stream information stored in a *Link*.

• **title**: String.
The title of the live stream.

• **description**: String.
The textual description of the live stream.

**Measurement**

A result obtained by a team for a given test on a sample is stored in a *Measurement*. Pertinent information on the test result are stored as attributes of this entity.

• **number**: Number.
The number associated with the sample.

• **assignedConcentration**: Number.
The concentration assigned to the sample.
- **sampleConcentration**: Number.
  The concentration of the specific sample.

- **sampleVolume**: Number.
  The volume of the specific sample.

- **timeToResult**: Number.
  The time to attain the measurement result.

- **signalValue**: Number.
  The value ascertained by the signal of a specific biosensor.

- **reportedConcentration**: Number.
  The concentration measured by the teams (different from actual concentration).

### Graph

The measurement data are presented to the users in graphs. These Graph entities can represent different kinds of graphs, i.e. by its type attribute, in addition it has visibility options for the graph stored. The Graph relations encode which data needs to be related to the user and the relation to Team signifies for which team data is being plotted.

- **type**: GraphType.
  The type of this graph.

- **visible**: Boolean.
  Flag that determines whether the graph is visible or not.

- **xAxisVisible**: Boolean.
  A flag that determines whether the x-axis is visible.

- **yAxisVisible**: Boolean.
  A flag that determines whether the y-axis is visible.

### Video

A Video contains descriptive information about its linked video. It can be associated with multiple Teams and Tags. Its relation to AuthenticatedUser encodes the uploader of the video.

- **title**: String.
  The title of the video.

- **description**: String.
  The description of the video.
• **status**: MediaStatus.
The status of a given media

**Picture**

Similarly to a Video entity, a Picture as a (descriptive) type, a status encoding whether it should be shown on site, and relations signifying its type of content (tags), its uploader (to a user), related teams, and, of course, the actual Link.

• **title**: String.
The title of the picture.

• **status**: MediaStatus.
The status of the picture

**Link**

A Link is a general entity which is responsible to store a URL to a resource. It stores the type of the resource.

• **sourceType**: SourceType.
The source of the link, whether it is

• **contentId**: String.
The identifier unique to the source linking to the media content, e.g. the identifier for a YouTube video.

**Statistics**

Site-wide statistics for operational concerns are captured by the Statistics.

• **numVisitors**: Number.
The amount of visitors that have accessed the SensUs Digital website.

• **uptime**: Number.
The uptime of the server hosting the SensUs Digital website

• **respTime**: Number.
The average response time of the SensUs Digital website.

• **availableRAM**: Number.
The available RAM (in bytes) of the server hosting the SensUs Digital website.
**Mapping**

A mapping stores how the SensUs organization is going to allocate samples to teams.

- `associatedTeam`: String.
  The team associated with the mapping.

- `sampleIDs`: [Number].
  The set of IDs associated with the samples to be measured.

- `concentrationValues`: [Number].
  The set of concentration values to be linked with the corresponding sample IDs.

- `currentMapping`: [(Number, Number)].
  The current mapping.

- `reserveSampleConcentrations`: [(Number, [Number])].
  The reserve sample concentrations used by each team.

### 2.7.4 MODEL-VIEW-CONTROLLER MODEL

The SensUs SensUs Digital Platform utilizes Laravel (PHP) on the back end and React (JS) with Redux on the front end. Because of these frameworks, we have used a model-view-controller model architecture as displayed in Figure 4. The descriptions for the entities are as follows:

- **User** - The user of the controller who interacts with the SensUs Digital website and the observer of the view.

- **View** - The SensUs Digital website itself, which has been implemented in HTML and CSS.

- **Model** - Contains data that is queried according to commands from the controller and shown in the view.

- **Controller** - Manipulates data in the model through the React Javascript files

- **SensUs Website Client** - Interface to the back-end data

- **SensUs Website Server** - Handles requests and updated made via the front end.

- **MySQL Database** - Stores data common to various parts of the SensUs Digital website
2.7.5 SEQUENCE DIAGRAMS

This section describes the sequence diagrams derived from the use cases in Appendix A of the URD [1]. Each use case in the URD is translated to a sequence diagram, following the numbering of the URD for the use cases, with sequence diagram A.0.1 being the exception as it deals with the initial loading of the website, which is not described in the URD.
Since the SensUs Digital website is designed as a web-app, the initial load of the website is different from page loads while on the website. The whole web-app is loaded in the initial load, containing the different views for the different pages. When a new page is requested, no request is made to the server, but the page is displayed using the initially loaded application. Only API requests are made when requesting new pages to just load the data of the page, not the layout. For all following sequence diagrams, it is assumed that the website is already loaded, and the new page in the use case only needs to make the API requests to load the data of the view.

Note that in the diagram, “Website resources” is a simplification of the multiple requests made for several types of resources, including HTML, images, stylesheets and scripts.

When loading the website it is possible that the user sends an authentication token when the user was already logged in from a previous visit. This token will then be checked with the database to see whether it is valid. In the case the token is valid, the returned website resources will have small changes to display the user information on the webpage.

**Goals:** To load and display the SensUs digital website.

**Preconditions:** The user is not banned on the website.

**Summary:** A user loads the SensUs digital website.

**Priority:** Must have.
A.1.1 LOGGING IN – Figure 6

The user selects the option to log in available from anywhere on the website. After receiving the login request, the SensUs server redirects the user to the Google OAuth login feature. The user then logs in, and if the authentication is successful, then Google responds with a successful authorization code, which can then be exchanged for a login token. The SensUs server then uses this token to load the user’s profile details on the website and store the details in the database. This view is then prepared for the user and displayed by the webpage. If the authorization was unsuccessful, then the user is redirected to the page they were previously on.

**Goals:** To log in.

**Preconditions:** The user has a Google account.

**Summary:** A user logs in.

**Priority:** Must have.
A.1.2 LOGGING OUT – Figure 7

A user selects the log out option available from anywhere on the website. This request is then passed to the SensUs server which then responds with a successful logout message and redirects the user to the home page.

Goals: To log out.
Preconditions: The user is logged in.
Summary: A user logs out.
Priority: Must have.

FIGURE 7: A.1.2 LOGGING OUT
A.1.3 CHANGING ACCOUNT ROLE – Figure 8

A web administrator changes a user's account role, and is then asked to confirm the change. After the changes are saved, the webpage passes the new role onto the server which then instructs a role change in the database for the respective client. Once the database has been successfully updated, the webpage displays a view with the updated changes.

**Goals:** To change a given user's role.

**Preconditions:** The user is a website administrator and has navigated to the page with administrator options.

**Summary:** A website administrator can change a given user's role.

**Priority:** Must have.

---

**FIGURE 8: A.1.3 CHANGING ACCOUNT ROLE**
A.2.1 ACCESSING A FULLSCREEN EVENT SUMMARY – Figure 9

A website administrator selects the event statistics, the request is then passed to the server and the event statistics are retrieved from the server. The server then prepares the view with the event summary which the webpage then displays to the user in fullscreen.

Goals: To access a fullscreen view of the event statistics.
Preconditions: The user is logged in as SensUs personnel.
Summary: The event statistics are accessed by SensUs personnel.
Priority: Must have.

FIGURE 9: A.2.1 ACCESSING A FULLSCREEN EVENT SUMMARY
A.3.1 UPLOADING STOCK TEAM PHOTO – Figure 10

SensUs personnel selects the option to upload a new stock photo for a given team. The webpage responds with a file navigation view. The user then selects an image file, to which the webpage responds with a confirmation request. If the user confirms the change, the request is passed on the server and the stock team photo is updated in the database. After this, the user is shown the page with the updated stock team photo. If the user decides to cancel the change, the view with the original stock team photo is shown.

Goals: To upload a stock team photo.
Preconditions: The user is a team member and has navigated to the team dashboard.
Summary: A team member uploads a new team photo to their team view.
Priority: Must have.
A.3.2 DELETING STOCK TEAM PHOTO – Figure 11

SensUs personnel selects the option to delete a stock photo for a given team, to which the webpage responds with a confirmation request. If the user confirms the change, the request is passed on the server and the stock team photo is deleted in the database. After this, the user is shown the page with the removed stock team photo. If the user decides to cancel the change, the view with the original stock team photo is shown.

Goals: To delete a stock team’s photo.
Preconditions: The user is a team member and has navigated to the team dashboard.
Summary: A team member deletes a stock team photo from their team view.
Priority: Must have.

FIGURE 11: A.3.2 DELETING STOCK TEAM PHOTO
A.3.3 UPLOADING PHOTO TO PHOTO GALLERY – Figure 12

SensUs personnel selects the option to upload a new photo to the photo gallery for a given team. The webpage responds with a file navigation view. The user then selects an image file, to which the webpage responds with a confirmation request. If the user confirms the upload, the request is passed on the server and the photo is added to the database. After this, the user is shown the team dashboard. If the user decides to cancel the change, the image is not added to the database and the user is shown the team dashboard.

Goals: To upload a team photo to their team photo gallery.
Preconditions: The user is a team member and has navigated to the team dashboard.
Summary: A team member uploads a photo to their photo gallery.
Priority: Must have.
A.3.4 DELETING PHOTO FROM PHOTO GALLERY – Figure 13

SensUs personnel selects the option to delete a photo from the photo gallery for a given team, to which the webpage responds with a confirmation request. If the user confirms the deletion, the request is passed on the server and the photo is deleted in the database. After this, the user is shown the team dashboard. If the user decides to cancel the deletion, the photo is not deleted and the team dashboard is shown.

Goals: To delete a photo from their photo gallery.
Preconditions: The user is a team member and has navigated to the team dashboard.
Summary: A team member can delete a photo from their photo gallery.
Priority: Must have.

FIGURE 13: A.3.4 DELETING PHOTO FROM PHOTO GALLERY
A.3.5 EDITING TEAM BACKGROUND INFORMATION – Figure 14

SensUs personnel selects a given team. The webpage responds with the view for that specific team. The user then selects the option to edit the team background information, to which the webpage responds with a confirmation request. If the user confirms the change, the request is passed on the server and the information is updated in the database. After this, the user is shown the team dashboard with the updated background information. If the user decides to cancel the change, the information is not updated in the database and the user is shown the team dashboard with the original information.

Goals: To edit the background information shown on a given team’s view.
Preconditions: The user is a team member and has navigated to the team dashboard.
Summary: A team member edits the team’s background information.
Priority: Must have.
A.4.1 VOTING WHEN AUTHENTICATED – Figure 15

An authenticated user selects exactly two distinct teams to vote for. The webpage makes the voting option available. The user submits the vote and the webpage responds with a confirmation view. If the user decides to confirm, the vote counters are updated in the database, and the user is shown the voting page view. If the user decides to cancel, the vote counters are not updated and the user is shown the voting page view.

**Goals:** To vote for two teams.

**Preconditions:** The user is authenticated and has navigated to the voting page.

**Summary:** A two team vote is cast by an authenticated user.

**Priority:** Must have.

---

**FIGURE 15:** A.4.1 VOTING WHEN AUTHENTICATED
A.4.2 VOTING WHEN UNAUTHENTICATED – Figure 16

An unauthenticated user selects exactly two distinct teams to vote for. The webpage makes the voting option available. The user submits the vote and the webpage responds with a login view. The user then follows the login procedure outlined in A.1.1. After the login procedure is completed, the webpage displays a confirmation view. If the user decides to confirm, the vote counters are updated in the database, and the user is shown the voting page view. If the user decides to cancel, the vote counters are not updated and the user is shown the voting page view.

**Goals:** To vote for two teams.

**Preconditions:** The user is unauthenticated and has navigated to the voting page.

**Summary:** A two team vote is cast by an initially unauthenticated user.

**Priority:** Must have.

---

**FIGURE 16:** A.4.2 VOTING WHEN UNAUTHENTICATED
A.5.1 DONATING TO A TEAM – Figure 17

A user selects a team to donate to, and the webpage shows payment options. The user then selects a payment option and subsequently the amount to donate to the team and charity. The webpage then makes the donation submission available and the user submits the donation. The webpage then displays a view with the external payment service of choice which the user selects. The user then follows the external payment procedure. If the payment was successful, the donation statistics are updated in the database and the user is shown a successful payment view, after which they may easily navigate back to the donation view by closing the successful payment message. If the payment was unsuccessful then the database is not changed and an unsuccessful payment is shown to the user, which they can close to find themselves on the donation page.

Goals: To donate to a team.

Preconditions: The user must have a PayPal account or a bank account supporting iDeal.

Summary: A user donates to a team.

Priority: Must have.
FIGURE 17: A.5.1 DONATING TO A TEAM
A.6.1 INPUTTING BIOSENSOR DATA – Figure 18

SensUs personnel selects the option to input biosensor data, to which the webpage responds with an input data form. The user then fills in the form and submits, after which the server shows a confirmation request. If the user confirms the input the biosensor data is updated in the database and the user is redirected to the biosensor dashboard with updated biosensor data. If the user cancels, the database is not updated and the webpage displays the original biosensor dashboard view.

Goals: To input biosensor data.

Preconditions: The user is logged in as SensUs personnel and has navigated to the biosensor dashboard.

Summary: SensUs personnel updates graph information by manually inputting biosensor data collected from different teams.

Priority: Must have.
A.6.2 RETRIEVING BIOSENSOR DATA – Figure 19

SensUs personnel select the option to retrieve biosensor data. If the data is present, then it is retrieved from the database and shown to the user. If the data is not present, there is an attempt to retrieve it from the database but the user is shown an error.

Goals: To retrieve biosensor data.

Preconditions: SensUs personnel retrieves biosensor data which was previously input.

Summary: The user is logged in as SensUs personnel and has navigated to the biosensor dashboard.

Priority: Must have.

![Diagram: A.6.2 Retrieving Biosensor Data]
A.7.1 SHARING THE EVENT ON SOCIAL MEDIA – Figure 20

A user selects the option to share the event through Facebook. This request is passed on to the SensUs server, which then passes on the request to a Facebook server. The Facebook server responds with a view to show the user. The user then has the option to edit the default message and selects the share option. The share information is then sent to Facebook which then responds with a success or failure. If the share is successful, the user is notified of the success. If the share is unsuccessful, the user is notified of the failure.

Goals: To share the event to Facebook.
Preconditions: The user has a Facebook account and has navigated to the home page.
Summary: The event is shared to the user’s Facebook feed.
Priority: Could have.

FIGURE 20: A.7.1 SHARING THE EVENT ON SOCIAL MEDIA
A.7.2 LOGGING IN ON THE SOCIAL MEDIA DASHBOARD – Figure 21

SensUs personnel selects the social media platform they would like to log in to. This request is passed on to the SensUs server and subsequently to the social media platform server. The social media platform responds with a login form, which the user fills in and submits. If the login is successful then the social media feed is shown, and otherwise the user is notified of the error and the feed is not shown.

**Goals:** To login to a given social media account through the centralized social media management dashboard.

**Preconditions:** SensUs personnel logs in to a given social media account through the centralized social media management dashboard.

**Summary:** The users who is logged in as SensUs personnel has an account of the given social media platform and has navigated to the social media dashboard.

**Priority:** Could have.

**FIGURE 21:** A.7.2 LOGGING IN ON THE SOCIAL MEDIA DASHBOARD
A.8.1 SUBMITTING REGULAR VIDEOS – Figure 22

An authenticated user selects the option to submit a video, and the webpage shows a video submission view. The user then inputs video information and submits. The webpage responds with a confirmation request. If the user confirms and the video information was valid, then the video is passed on to the video control room and added to the database. The user is notified that the submission was successful and is redirected to the explore page. If the user confirms and the video information was incorrectly input, then the database is not modified, the video is not passed on to the control room and the user is notified of the incorrect input. Then the user is redirected to the explore page. If the user decides to cancel the submission, then they are redirected to the explore page.

Goals: To submit a regular video.
Preconditions: The user must be authenticated, must have a URL to the video being submitted and has navigated to the explore tab.
Summary: A user submits a regular video.
Priority: Must have.
FIGURE 22: A.8.1 SUBMITTING REGULAR VIDEOS
A.8.2 SUBMITTING A QUESTION VIDEO VIA THE EXPLORE PAGE – Figure 23

An authenticated user selects the option to submit a question video, and the webpage shows a question video submission view. The user then inputs the question video information and submits. The webpage responds with a confirmation request. If the user confirms and the question video information was valid, then the question video is passed on to the video control room and added to the database. The user is notified that the submission was successful and is redirected to the explore page. If the user confirms and the question video information was incorrectly input, then the database is not modified, the question video is not passed on to the control room and the user is notified of the incorrect input. Then the user is redirected to the explore page. If the user decides to cancel the submission, then they are redirected to the explore page.

Goals: To submit a question video.
Preconditions: The user must be authenticated, must have a URL to the video being submitted and has navigated to the explore tab.
Summary: A user submits a question video from the explore page.
Priority: Must have.
FIGURE 23: A.8.2 SUBMITTING A QUESTION VIDEO VIA THE EXPLORE PAGE
A.8.3 SUBMITTING A TEAM VIDEO – Figure 24

SensUs personnel selects the team page, and the webpage shows the respective team view. The user then inputs the video information and submits. The webpage responds with a confirmation request. If the user confirms and the team video information was valid, then the team video is added to the database. The user is notified that the submission was successful and is shown the team dashboard. If the user confirms and the team video information was incorrectly input, then the database is not modified and the user is notified of the incorrect input. If the user decides to cancel the submission, they are shown the team dashboard.

Goals: To submit a video associated with a team.
Preconditions: The user is SensUs personnel and has navigated to the team dashboard.
Summary: SensUs personnel submits a team video.
Priority: Must have.
FIGURE 24: A.8.3 SUBMITTING A TEAM VIDEO
A.8.4 DELETING A TEAM VIDEO – Figure 25

SensUs personnel selects the team page, and the webpage shows the respective team view. The user then selects the option to delete the team video. The webpage responds with a confirmation request. If the user confirms then the team video is removed from the database. The user is notified that the deletion was successful and is shown the team dashboard. If the user decides to cancel the deletion, they are shown the team dashboard.

**Goals:** To delete a team video.

**Preconditions:** The user is SensUs personnel, a team video has been submitted previously and the user has navigated to the team dashboard.

**Summary:** SensUs personnel deletes a team video.

**Priority:** Must have.

![Diagram of the process for deleting a team video](image)

**FIGURE 25:** A.8.4 DELETING A TEAM VIDEO
A.8.5 CHANGING REGULAR VIDEO STATUS – Figure 26

SensUs personnel selects a video to edit the status of. The webpage displays the available status options. The user then selects an option and is shown a confirmation request by the webpage. If the user confirms the status change, then the status for that video is changed in the database, and the user notified of the successful update. If the user cancels the status change, then the database does not get updated and the control room view is shown to the user.

**Goals:** To change the status of a regular video.

**Preconditions:** The user is logged in as SensUs personnel and has navigated to the control room dashboard.

**Summary:** SensUs personnel can change the status of all regular videos from the control room dashboard.

**Priority:** Must have.
A.8.6 CHANGING REGULAR VIDEO TAG – Figure 27

SensUs personnel selects a video to edit the tags of. The webpage displays the tagging edit form. The user then edits the tags, submits the form and is shown a confirmation request by the webpage. If the user confirms the tag change, then the tags for that video are changed in the database, and the user is notified of the successful update. If the user cancels the tag change, then the database does not get updated and the control room view is shown to the user.

**Goals:** To change the tag of a regular video.

**Preconditions:** The user is logged in as SensUs personnel and has navigated to the control room dashboard.

**Summary:** SensUs personnel can change the status of all regular videos from the control room dashboard.

**Priority:** Must have.
A.8.7 EDITING REGULAR VIDEO TITLE – Figure 28

SensUs personnel selects a video to edit the title of. The webpage displays the title editing form. The user then edits the title, submits the form and is shown a confirmation request by the webpage. If the user confirms the title change, then the title for that video is changed in the database, and the user is notified of the successful update. If the user cancels the title change, then the database does not get updated and the control room view is shown to the user.

**Goals:** To change the title of a regular video.

**Preconditions:** The user is logged in as SensUs personnel and has navigated to the control room dashboard.

**Summary:** SensUs personnel can change the title of a regular video from the control room dashboard.

**Priority:** Must have.

![A.8.7 Editing Regular Video Title Diagram](image-url)
A.8.8 EDITING REGULAR VIDEO DESCRIPTION – Figure 29

SensUs personnel selects a video to edit the description of. The webpage displays the description editing form. The user then edits the description, submits the form and is shown a confirmation request by the webpage. If the user confirms the description change, then the description for that video is changed in the database, and the user is notified of the successful update. If the user cancels the description change, then the database does not get updated and the control room view is shown to the user.

Goals: To change the title of a regular video.
Preconditions: The user is logged in as SensUs personnel and has navigated to the control room dashboard.
Summary: SensUs personnel can change the title of a regular video from the control room dashboard.
Priority: Must have.
A.8.9 DELETING A REGULAR VIDEO – Figure 30

SensUs personnel selects a video to delete. The webpage shows a confirmation request. If the user confirms the deletion, then the video is removed from the database, and the user is notified of the successful deletion. If the user cancels the deletion, then the database is not changed, and the user is shown the control room dashboard view.

Goals: To delete a regular video.
Preconditions: The user is logged in as SensUs personnel and has navigated to the control room dashboard.
Summary: SensUs personnel deletes a regular video from the control room dashboard.
Priority: Must have.
A.9.1 CHANGING QUESTION VIDEO TO REGULAR VIDEO – Figure 31

To change a question video to a regular video, a Team Member loads the Q&A page. The user selects a question to change to a regular video, triggering a confirmation dialog. Only once confirmed, a request is sent to the server to change the question video to a regular video question. Assuming the user has the right permissions, the server updates the database to remove the question and change the related video’s state to a regular video. The server sends back a response when finished, after which the webpage will update its content to reflect the removed question. In the case the confirmation message is canceled, nothing is sent to the server and the original page is shown again.

Goals: To change the state of a question video to a regular video.
Preconditions: The user is a team member.
Summary: A team member changes the state of a question video from their team-specific Q&A dashboard.
Priority: Must have.
A.9.2 SENDING QUESTION VIDEO TO ANOTHER TEAM – Figure 32

To send a question video to another team, a Team Member loads the Q&A page. The user selects the question to send, which shows a form where the user can select the team to send the question to. After the user selects a team a confirmation dialog shows up. Only once confirmed, a request is sent to the server to send the question. Assuming the user has the right permissions, the server updates the database to update the team of the question. The server sends back a response when finished, after which the webpage will update its content to reflect the updated question. In the case the confirmation message is canceled, nothing is sent to the server and the original page is shown again.

**Goals:** To send a question video to the Q&A dashboard of another team.

**Preconditions:** The users is logged in as a team member.

**Summary:** A team member changes a question video’s intended team.

**Priority:** Must have.
A.9.3 QUESTION VIDEO RESERVATION – Figure 33

To reserve a video question, a Team Member loads the Q&A page. The user selects the question to reserve, which triggers a confirmation dialog. Only once confirmed, a request is sent to the server to reserve the question. Assuming the user has the right permissions, the server updates the database to mark the question as reserved. The server sends back a response when finished, after which the webpage will update its content to reflect the now reserved question. In the case the confirmation message is canceled, nothing is sent to the server and the original page is shown again.

Goals: To reserve the question video to be answered, setting the video's status to "being answered."

Preconditions: The user is a team member.

Summary: To ensure that only one team member answers the question, it sets the video's status to "being answered."

Priority: Must have.
A.9.4 PAIRING QUESTIONS AND ANSWERS – Figure 34

To pair a question with an answer, a Team Member loads the Q&A page. The user selects the question to pair (answer), showing a form in which the user can input an answer video URL. If the URL is invalid, an error is shown and the user can update the URL until a valid URL has been entered. Once submitted, a confirmation dialog is shown. Only once confirmed, a request is sent to the server to link the answer to the question. Assuming the user has the right permissions, the server creates the answer in the database and links the answer to the question. The server sends back a response when finished, after which the webpage will update its content to reflect the now answered question. In the case the confirmation message is canceled, nothing is sent to the server and the original page is shown again.

**Goals:** To pair a question video with an answer video.

**Preconditions:** The user is a team member, the video to be answered has status “being answered by team”.

**Summary:** A team member can couple a question video with an answer video from the Q&A dashboard.

**Priority:** Must have.
FIGURE 34: A.9.4 PAIRING QUESTIONS AND ANSWERS
A.9.5 CHANGING A QUESTION VIDEO FROM "BEING ANSWERED" TO "PENDING" – Figure 35

To set the status of a question video back to pending, a Team Member loads the Q&A page. The user selects the question to cancel answering, which triggers a confirmation dialog. Only once confirmed, a request is sent to the server to cancel the answering of the question. Assuming the user has the right permissions, the server updates the database to mark the question as pending again. The server sends back a response when finished, after which the webpage will update its content to reflect the now pending question. In the case the confirmation message is canceled, nothing is sent to the server and the original page is shown again.

Goals: To send a video back to pending.
Preconditions: The user is a team member.
Summary: A team member changes a question video’s status from “Being answered by team” to “pending”.
Priority: Must have.

FIGURE 35: A.9.5 CHANGING A QUESTION VIDEO FROM "BEING ANSWERED" TO "PENDING"
A.9.6 REJECTING QUESTION VIDEOS – Figure 36

To reject a question video, a Team Member loads the Q&A page. The user selects the question to reject, which triggers a confirmation dialog. Only once confirmed, a request is sent to the server to reject the question. Assuming the user has the right permissions, the server updates the database to mark the question as rejected. The server sends back a response when finished, after which the webpage will update its content to reflect the now rejected question. In the case the confirmation message is canceled, nothing is sent to the server and the original page is shown again.

Goals: To reject a question video.
Preconditions: The user is a team member.
Summary: A team member rejects a question video.
Priority: Must have.

FIGURE 36: A.9.6 REJECTING QUESTION VIDEOS
To edit a question video title, SensUs personnel loads the Q&A page. The user selects the question to edit, showing a form to edit the question. The user can edit the title in the form, finish by submitting the form, thereby triggering a confirmation message. When confirmed, a request is made to the server to update the question. Assuming the user has the right permissions, the server updates the question in the database, sending back a response when finished. After that, the view will update reflecting the updated question. If the confirmation message was canceled, the original view is shown again.

Goals: To change the title of a question video.

Preconditions: The user is logged in as SensUs personnel.

Summary: SensUs personnel can change the title of a question video.

Priority: Must have.
A.9.8 EDITING QUESTION VIDEO DESCRIPTION – Figure 38

To edit a question video description, SensUs personnel loads the Q&A page. The user selects the question to edit, showing a form to edit the question. The user can edit the description in the form, finish by submitting the form, thereby triggering a confirmation message. When confirmed, a request is made to the server to update the question. Assuming the user has the right permissions, the server updates the question in the database, sending back a response when finished. After that, the view will update reflecting the updated question. If the confirmation message was canceled, the original view is shown again.

Goals: To change the description of a question video.
Preconditions: The user is logged in as SensUs personnel.
Summary: SensUs personnel can change the description of a question video from the Q&A dashboard.
Priority: Must have.
FIGURE 38: A.9.8 EDITING QUESTION VIDEO DESCRIPTION
A.9.9 DELETING A QUESTION VIDEO – Figure 39

To delete a question, SensUs personnel loads the Q&A page. The user selects the question to delete. If the question already has an answer, the user is asked whether they also want to delete the answer video. After the user chooses whether they want to delete the answer video a confirmation dialog is shown, asking whether the user wants to delete the question. When confirmed, a request is made to the server to delete the question. Assuming the user has the right permissions, the server deletes the question in the database, optionally preceded by deleting the answer. After deletion is done, the server sends back a response, followed by the view updating its content to reflect the deleted question and possibly answer. If the confirmation message was canceled, the original view is shown again.

Goals: To delete a question video.
Preconditions: The user is logged in as SensUs personnel.
Summary: SensUs personnel deletes a question video from the Q&A dashboard.
Priority: Must have.
FIGURE 39: A.9.9 DELETING A QUESTION VIDEO
A.9.10 CHANGING QUESTION VIDEO TAG – Figure 40

To change a video question tag, SensUs personnel loads the Q&A page. The user selects the question to edit, showing a form to edit the question. The user can add and remove tags from the question in the form, finish by submitting the form, thereby triggering a confirmation message. When confirmed, a request is made to the server to update the question. Assuming the user has the right permissions, the server updates the question in the database, sending back a response when finished. After that, the view will update reflecting the updated question. If the confirmation message was canceled, the original view is shown again.

Goals: To change the tag of a question video.
Preconditions: The user is logged in as SensUs personnel.
Summary: SensUs personnel can change the tag of a question video from the Q&A dashboard.
Priority: Must have.
FIGURE 40: A.9.10 CHANGING QUESTION VIDEO TAG
A.9.11 SENDING A TEXT QUESTION TO ANOTHER TEAM – Figure 41

To send a text question to another team, a Team Member loads the Q&A page. The user selects the question to send, which shows a form where the user can select the team to send the question to. After the user selects a team a confirmation dialog shows up. Only once confirmed, a request is sent to the server to send the question. Assuming the user has the right permissions, the server updates the database to update the team of the question. The server sends back a response when finished, after which the webpage will update its content to reflect the updated question. In the case the confirmation message is canceled, nothing is sent to the server and the original page is shown again.

Goals: To send a text question to the Q&A dashboard of another team.
Preconditions: The user is a team member, desired text question has status “pending”.
Summary: A team member changes a text question’s intended team.
Priority: Must have.

FIGURE 41: A.9.11 SENDING A TEXT QUESTION TO ANOTHER TEAM
A.9.12 RESERVING TEXT QUESTION – Figure 42

To reserve a text question, a Team Member loads the Q&A page. The user selects the question to reserve, which triggers a confirmation dialog. Only once confirmed, a request is sent to the server to reserve the question. Assuming the user has the right permissions, the server updates the database to mark the question as reserved. The server sends back a response when finished, after which the webpage will update its content to reflect the now reserved question. In the case the confirmation message is canceled, nothing is sent to the server and the original page is shown again.

Goals: To reserve a text question to be answered.
Preconditions: The user is a team member.
Summary: A team member indicates that a text question is being answered, by changing it’s status to "being answered".
Priority: Must have.

FIGURE 42: A.9.12 RESERVING TEXT QUESTION
A.9.13 ANSWERING A TEXT QUESTION – Figure 43

To answer a text question, a Team Member loads the Q&A page. The user selects the question to answer, which shows an answer form. The user inputs the answer, updating the form. After submitting the form, a confirmation dialog is shown. Only once confirmed, a request is sent to the server to answer the question. Assuming the user has the right permissions, the server updates the database to add the answer to the question. The server sends back a response when finished, after which the webpage will update its content to reflect the now answered question. In the case the confirmation message is canceled, nothing is sent to the server and the original page is shown again.

Goals: To answer a text question.
Preconditions: The user is a team member, the text question to be answered has the status “being answered by team”.
Summary: A team member can submit an answer for a text question from the Q&A dashboard.
Priority: Must have.
FIGURE 43: A.9.13 ANSWERING A TEXT QUESTION
A.9.14 UNRESERVING A TEXT QUESTION – Figure 44

To set the status of a text question back to pending, a Team Member loads the Q&A page. The user selects the question to cancel answering, which triggers a confirmation dialog. Only once confirmed, a request is sent to the server to cancel the answering of the question. Assuming the user has the right permissions, the server updates the database to mark the question as pending again. The server sends back a response when finished, after which the webpage will update its content to reflect the now pending question. In the case the confirmation message is canceled, nothing is sent to the server and the original page is shown again.

Goals: To set the status of a text question back to pending.
Preconditions: The user is a team member, the desired text question has status “being answered by team”.
Summary: If a team member has already reserved a video to answer, but then realizes for some reason that he/she won’t be able to answer it for some reason, then they can release it back to allow others to answer it.
Priority: Must have.

FIGURE 44: A.9.14 UNRESERVING A TEXT QUESTION
**A.9.15 REJECTING TEXT QUESTION** – Figure 45

To reject a text question, a Team Member loads the Q&A page. The user selects the question to reject, which triggers a confirmation dialog. Only once confirmed, a request is sent to the server to reject the question. Assuming the user has the right permissions, the server updates the database to mark the question as rejected. The server sends back a response when finished, after which the webpage will update its content to reflect the now rejected question. In the case the confirmation message is canceled, nothing is sent to the server and the original page is shown again.

**Goals:** To reject a text question.

**Preconditions:** The user is a team member.

**Summary:** A team member can reject a text question from the Q&A dashboard. In this way team members can reject text questions that are not good quality questions, but should be kept for some other reason. Sets the text question’s status to "rejected".

**Priority:** Must have.

---

**FIGURE 45: A.9.15 REJECTING TEXT QUESTION**
A.9.16 FILTERING REGULAR VIDEOS – Figure 46

After the user loads the explore page, an option to filter videos by tag is presented. The videos will then get filtered client-side, after which the view updates with the now filtered videos.

Goals: To filter available videos on a tag in the filter menu.

Preconditions: None.

Summary: A view of all videos with that tag are presented.

Priority: Must have.

FIGURE 46: A.9.16 FILTERING REGULAR VIDEOS
A.9.17 CHANGING TEXT QUESTION TAG – Figure 47

To change a text question tag, SensUs personnel loads the Q&A page. The user selects the question to edit, showing a form to edit the question. The user can add and remove tags from the question in the form, finishing by submitting the form, triggering a confirmation message. When confirmed, a request is made to the server to update the question. Assuming the user has the right permissions, the server updates the question in the database, sending back a response when finished. After that, the view will update reflecting the updated question. If the confirmation message was canceled, the original view is shown again.

**Goals:** To change a tag of a text question.

**Preconditions:** The user is logged in as SensUs personnel.

**Summary:** SensUs personnel can change the tag of a text question from the Q&A dashboard.

**Priority:** Must have.
FIGURE 47: A.9.17 CHANGING TEXT QUESTION TAG
A.9.18 DELETING A QUESTION – Figure 48

To delete a question, SensUs personnel loads the Q&A page. The user selects the question to delete, triggering a confirmation message. When confirmed, a request is made to the server to delete the question. Assuming the user has the right permissions, the server deletes the question in the database, sending back a response when finished. After that, the view will update with the deleted question removed from the view. If the confirmation message was canceled, the original view is shown again.

Goals: To delete a text question.
Preconditions: The user is logged in as SensUs personnel.
Summary: SensUs personnel deletes a text question from the Q&A dashboard.
Priority: Must have.
A.9.19 FILTERING QUESTIONS – Figure 49

To filter the questions, SensUs personnel loads the Q&A page. The user selects the tag to filter on, after which the questions will be filtered on the client-side with no server interaction to save server load. Once filtered, the view updates with the filtered questions.

**Goals:** To filter questions on various tags.

**Preconditions:** The user is authenticated as SensUs personnel.

**Summary:** SensUs personnel filters questions on a tag.

**Priority:** Must have.
A.9.20 BANNING A USER – Figure 50

To ban a user from the website, SensUs personnel loads the page to edit account roles in the dashboard. The SensUs personnel user selects the (un-banned) user to ban, which triggers a confirmation dialog. Only once confirmed, a request is sent to the server to ban the user. Assuming the user has the right permissions, the server updates the database to mark the user as banned, removing access of the user to the platform. The server sends back a response when finished, after which the webpage will update its content to reflect the now banned user. In the case the confirmation message is canceled, nothing is sent to the server and the original page is shown again.

Goals: To ban a given user from submitting further content.
Preconditions: The user is logged in as SensUs personnel, the user to ban is currently not banned and the user has navigated to the page with administrator options.
Summary: SensUs personnel can ban certain users. This way, SensUs personnel can prevent spam and other inappropriate material.
Priority: Must have.
FIGURE 50: A.9.20 BANNING A USER
A.9.21 UN-BANNING A USER – Figure 51

To un-ban a user from the website, SensUs personnel loads the page to edit account roles in the dashboard. The user selects the (banned) user to un-ban, which triggers a confirmation dialog. Only once confirmed, a request is sent to the server to un-ban the selected user. Assuming the user has the right permissions, the server updates the database to mark the selected user as un-banned, allowing the user to access the platform again. The server sends back a response when finished, after which the webpage will update its content to reflect the now un-banned user. In the case the confirmation message is canceled, nothing is sent to the server and the original page is shown again.

Goals: To un-ban a user so they can submit further content.

Preconditions: The user reversing the ban is authenticated as SensUs personnel and the user to be un-banned is banned.

Summary: SensUs personnel can un-ban certain users that are banned by accident or are allowed to be un-banned again.

Priority: Must have.
FIGURE 51: A.9.21 UN-BANNING A USER
3 SOFTWARE REQUIREMENTS

This section contains all software requirements for the SensUs Digital Platform. All requirements have been prioritized using the MoSCoW model [2]. The model is as follows:

- **M Must have**: These requirements are critical for the system to function and absolutely must be included in the final product.

- **S should have**: These requirements are desirable for the system to function as desired, but can be left out if there are not sufficient resources to do so.

- **C could have**: These requirements are good possible capabilities for the system to have, but only should be done if there is extra time left or if it is trivially easy to implement.

- **W Won't have**: These requirements are possibilities, but fall outside the scope of this project. However, they could be realized in some later version of the product.

3.1 FUNCTIONAL REQUIREMENTS

3.1.1 EXAMPLE

<table>
<thead>
<tr>
<th>SR - number</th>
<th>function or name of variable</th>
<th>Description of requirement</th>
<th>Input (only relevant for functions)</th>
<th>Precondition (only relevant for functions)</th>
<th>Postcondition (only relevant for functions)</th>
</tr>
</thead>
</table>

The class diagram in Section 2.7.2 acts as the basis for organizing our requirements. The software requirements are derived from the functional user requirements as specified in Section 3.1 of the URD.

3.1.2 USER - UNAUTHENTICATED USER

3.1.2.1 Operations

**SR - 1** *must have*

*login(email)*: Void

*email*: String The email that uniquely identifies the (Google OAuth) user.

A function that performs the login procedure of OAuth and stores in the back-end that the user is now logged in.

*Precondition*: The unauthenticated user is in possession of a valid Google OAuth account and is not banned.
Postcondition: The user has all capabilities of an authenticated user on the site and any capabilities associated with the user’s specific role.

SR - 2 should have

`shareEvent(service): Void`

`service: ServiceType`  The service on which to share the SensUs event. `ServiceType` contains Facebook, Twitter, and LinkedIn.

A function that allows the user to share the event to one of the above social networks.

Precondition: The user owns an account to one of the above social networks.

Postcondition: The user either cancels the sharing through the service provider’s form or the user completes the form and shares the event.

3.1.3 USER - AUTHENTICATED USER

SR - 3 must have

AuthenticatedUser extends UnauthenticatedUser.

Any capability that is afforded to an UnauthenticatedUser, by the requirements in this document, is also afforded to an AuthenticatedUser.

3.1.3.1 Attributes

SR - 4 must have

`name: String`. The name of the user, as identified by Google OAuth.

SR - 5 must have

`email: String`. The email of the user, as required by Google OAuth.

SR - 6 must have

`remember_token: String`. The cookie identifier used by Google OAuth to keep a user logged in.

SR - 7 must have

`banned: Boolean`. A Boolean identifying whether the user has been banned or not.

3.1.3.2 Operations

SR - 8 must have

`logout(): Void`
A function that performs the logout procedure and modifies the back end to reflect that the user is logged out.

**Precondition**: The user is logged in.

**Postcondition**: The user becomes unauthenticated and no longer has any of the capabilities reserved to users of authenticated role and above.

SR - 9

`castVote(teamA, teamB): Void`

`teamA: Team`  The first of the two teams for whom the user votes.

`teamB: Team`  The second of the two teams for whom the user votes.

A function that casts/updates a vote for two teams, for this particular user. If the user has already cast their vote in the past their vote gets updated.

**Precondition**: The user is logged in.

**Postcondition**: If no prior vote was cast a new `Vote` is created and associated with this user and the two teams, else the vote for this user is updated for the new teams (removing the vote for the old teams).

SR - 10

`endorseQuestion(question, state): Void`

`question: Question`  The question which the user is endorsing.

`state: Boolean`  Whether the question is being endorsed or un-endorsed.

A function whereby a user endorses a question. Is idempotent.

**Precondition**: None.

**Postcondition**: The user’s endorsement status for the question is updated.

### 3.1.4 USER - TEAM MEMBER

SR - 11

TeamMember extends AuthenticatedUser.

Any capability that is afforded to a AuthenticatedUser, by the requirements in this document, is also afforded to a TeamMember.

### 3.1.4.1 Operations

SR - 12

`addComment(question, text, state): Void`

`question: Question`  The question which the TeamMember is modifying by adding/removing a comment.

`text: String`  The comment’s text that is being added.

`state: Boolean`  Whether the comment is being added or removed.

A function whereby a team member can add/remove a comment to/from a question.
Precondition: None.
Postcondition: The question’s comments are updated to reflect the arguments used for this function.

3.1.5 USER - SENSUS PERSONNEL

SR - 13  
SensUsPersonnel extends TeamMember.
Any capability that is afforded to a TeamMember, by the requirements in this document, is also afforded to a SensUsPersonnel user.

3.1.5.1 Operations

SR - 14  
ban(user, banState): Void  
user: AuthenticatedUser  The user whose banned state is being modified.  
banState: Boolean  The new banned state of the user.

A function that either bans or unbans the user from logging in again.
Precondition: The user exists on the back-end, and the user is not a WebsiteAdministrator user.
Postcondition: The user’s banned state will reflect the banState argument, and the user will correspondingly be able/no longer be able to log in. If the user becomes banned then the user will be logged out as soon as possible, provided the user was still logged in.

3.1.6 USER - WEBSITE ADMINISTRATOR

SR - 15  
WebsiteAdministrator extends SensUsPersonnel.
Any capability that is afforded to a SensUsPersonnel user, by the requirements in this document, is also afforded to a WebsiteAdministrator user.

3.1.6.1 Operations

SR - 16  
changeRole(user, role): Void  
user: AuthenticatedUser  The user whose role is being modified.  
role: UserType  The new role (i.e. user type, AuthenticatedUser, TeamMember, etc.) for this user.

A function that changes the user’s role, thereby giving or taking capabilities from the user.
Precondition: The user exists on the back end.

Postcondition: The role of the user is changed to role and the user hereafter has all capabilities associated with the new role.

3.1.7 TEAM

3.1.7.1 Attributes

SR - 17
name: String.
The name of the team.

SR - 18
picture: URL.
A URL linking to a picture of the team.

SR - 19
country: String.
The name of the team’s country of origin.

SR - 20
university: String.
The name of the team’s university.

SR - 21
description: String.
The description for the team, which serves as its background information.

SR - 22
poster: URL.
The link to the Team’s (digital) poster.

3.1.7.2 Operations

SR - 23
update(desc, picURL): Void
desc: String  The new description for the team.
picURL: String  The new picture URL for the team.
A function that updates the description and picture for the team.

Precondition: The user initiating the description and picture change action is a SensUsPersonnel user.

Postcondition: The Team’s description has value desc and Team’s picture has value picURL.
SR - 24  
*must have*

`pictures()`: [Picture]

A function that returns all the Pictures that are in the many-to-many relation with this Team.

**Precondition**: None.

**Postcondition**: All the returned Pictures are associated with the team and there are no Pictures also associated with the team that are not in the return value.

SR - 25  
*must have*

`questions()`: [Question]

A function that returns all the Questions that are in the many-to-many relation with this Team.

**Precondition**: None.

**Postcondition**: All the returned Questions are associated with the team and there are no Questions also associated with the team that are not in the return value.

SR - 26  
*should have*

`getCharity()`: Charity

A function that returns the Charity associated with this Team.

**Precondition**: None.

**Postcondition**: The unique Charity associated with this Team is returned.

### 3.1.8  TEAM - SENSUS ORGANIZATION

SR - 27  
*must have*

SensUsOrganization extends Team.

Any capability that is afforded to a Team, by the requirements in this document, is also afforded to the SensUsOrganization team.

#### 3.1.8.1  Attributes

SR - 29  
*should have*

`introVideo`: URL.

The link to the introduction video for the SensUs organization.

SR - 30  
*should have*

`memberInfo`: [(String, URL)].

An array containing the biographical information and a picture of each member of the SensUs organization.

SR - 31  
*must have*

`eventProgram`: [String].
An array containing the event program for each day.

**SR - 32** must have

`judges: [(String, URL)]`.

An array containing the biographical information and a picture of each judge for the SensUs Digital event.

**SR - 33** must have

`partners: [(String, URL)]`.

An array containing the biographical information of and a link to each partner for the SensUs Digital event.

**SR - 34** must have

`awards: [String]`.

An array containing the descriptions for each award.

### 3.1.8.2 Operations

**SR - 35** should have

`getSocialActivities(): [SocialMessage]`

A function that retrieves all the social media messages associated with the SensUs organization. The social networks supported are Facebook, Twitter, and LinkedIn.

**Precondition:** None.

**Postcondition:** The relevant social media messages are visible to the user initiating the retrieval action.

### 3.1.9 CHARITY

#### 3.1.9.1 Attributes

**SR - 36** should have

`name: String`.

The name of the charity.

**SR - 37** should have

`amountRaised: (Number, Number)`.

The total amount raised for this charity, and by the associated team.

#### 3.1.9.2 Operations

**SR - 38** should have

`update(newName): Void`

`newName: String` The new name for the charity.
A function that updates the name for the Charity.

**Precondition:** The user initiating the name change action is a SensUsPersonnel user.

**Postcondition:** This Charity's name has value *newName*.

```
SR - 39
sendMoney(serviceProvider, teamAmount, charityAmount): Void
```

`serviceProvider`: ServiceProvider  
The money transfer service the user would like to use. Either PayPal or iDeal.

`teamAmount`: Amount  
The amount of money to donate by the user to the team.

`charityAmount`: Amount  
The amount of money to donate by the user to the charity.

A function that executes the donation action of a user, using the `serviceProvider` specified, with the donation directed to both a Team and the team's Charity.

**Precondition:** The user initiating the action is logged in.

**Postcondition:** The donation(s) have been submitted to the service provider and the amountRaised is updated accordingly.

### 3.1.10  QUESTION

#### 3.1.10.1  Attributes

```
SR - 40
title: String.
The title of the question.

SR - 41
description: String.
The description of the question.

SR - 42
status: QuestionStatus.
The status containing whether the answer is Unapproved, Approved, and Rejected, Coupled-Approved, Coupled-Unapproved.

SR - 43
viewCount: Number.
The number of views for this question.

SR - 44
date: Date.
The date and time of when this question was posted.
```
3.1.10.2 Operations

SR - 45  
**static post**(*title, desc, link, team*): Question  
**title**: String  
The title for the question.  
**desc**: String  
The description for the question.  
**link**: Link  
The link for the question (optional).  
**team**: Team  
The team to which the question is directed.  

A function that posts a given question to the database.  
**Precondition**: The user initiating the post action is an AuthenticatedUser.  
**Postcondition**: The database contains the question with its fields set according the arguments, and with the status set to Unapproved, and date set to the current time and date. If a valid link is supplied the question becomes associated with that link.

SR - 46  
**update**(status, team): Void  
**status**: QuestionStatus  
The new status for the question.  
**team**: Team  
The team to which the question is directed.  

A function that updates a given question's status and associated team.  
**Precondition**: The user initiating the update action is a TeamMember user for the team associated with the question or is a SensUsPersonnel user.  
**Postcondition**: The database contains the question with the updated status and team.

SR - 47  
**addAnswer**(answer): Void  
**answer**: Answer  
The answer that is to be associated with this question.  

A function that couples an answer to the question.  
**Precondition**: The user initiating the update action is a TeamMember user for the team associated with the question or is a SensUsPersonnel user.  
**Postcondition**: The database contains the question-answer association.

SR - 48  
**delete**(): Void  
A function that deletes a given question from the database.  
**Precondition**: The user initiating the delete action is a SensUsPersonnel user.  
**Postcondition**: The database no longer contains the question specified and any associated answer is removed from the database.

3.1.11 ANSWER
3.1.11.1 Attributes

SR - 49  
**title:** String.  
The title of the answer.

SR - 50  
**description:** String.  
The description of the answer.

SR - 51  
**status:** AnswerStatus.  
The status containing whether the answer is Pending, Approved, and Rejected.

3.1.11.2 Operations

SR - 52  
**update(status):** Void  
**status:** AnswerStatus  
The new status for the answer.

A function that updates a given answer’s status.  
**Precondition:** The user initiating the update action is a SensUsPersonnel user.  
**Postcondition:** The database contains the answer with the updated status.

SR - 53  
**delete():** Void  
A function that deletes a given answer from the database.  
**Precondition:** The user initiating the delete action is a SensUsPersonnel user.  
**Postcondition:** The database no longer contains the answer. Any question that has a relation to this answer has the relation removed.

3.1.12 VOTE

3.1.12.1 Attributes

SR - 54  
**static statisticsVisible:** Boolean.  
Static attribute, shared among all votes, that sets whether the viewer statistics for votes should be visible. The statistics themselves, such as location of users, will be retrieved from the authentication service used.
3.1.12.2 Operations

SR - 55 must have

static getTotals(): [(Team, Number)]
A function that returns the count statistics per team.

Precondition: None. Postcondition: All the returned counts are associated with the corresponding teams and there are no votes that are unaccounted for in the counts returned.

SR - 56 should have

static setStatVisibility(statVis): Void
statVis: Boolean The new statistics visibility state for all votes.
A function that modifies the statistics visibility state for all votes.

Precondition: The user initiating the visibility change action is a SensUsPersonnel user.
Postcondition: The static vote statisticsVisible attribute reflects the statVis argument.

3.1.13 TAG

3.1.13.1 Attributes

SR - 57 must have

name: String.
The name of the tag.

3.1.14 LIVESTREAM

3.1.14.1 Attributes

SR - 58 must have

title: String.
The title of the live stream.

SR - 59 must have

description: String.
The textual description of the live stream.

3.1.14.2 Operations

SR - 60 should have

amountViewers(): Number
A function that retrieves the current viewer amount from the livestream hoster.

Precondition: The livestream hoster provides information about view count.
Postcondition: The current amount of viewers (according to the livestream hoster) is returned.
3.1.15 MEASUREMENT

3.1.15.1 Attributes

SR - 61  
*must have*

*number*: Number.
The number associated with the sample.

SR - 62  
*must have*

*assignedConcentration*: Number.
The concentration assigned to the sample.

SR - 63  
*must have*

*sampleConcentration*: Number.
The concentration of the specific sample.

SR - 64  
*must have*

*sampleVolume*: Number.
The volume of the specific sample.

SR - 65  
*must have*

*timeToResult*: Number.
The time to attain the measurement result.

SR - 66  
*must have*

*signalValue*: Number.
The value ascertained by the signal of a specific biosensor.

SR - 67  
*must have*

*reportedConcentration*: Number.
The concentration measured by the teams (different from actual concentration).

3.1.15.2 Operations

SR - 68  
*must have*

*update*(num, assCon, samCon, volume, timeToResult, signal, repCon): Void

*num*: Number  The new number associated with the measurement.

*assCon*: Number  The new assigned concentration associated with the measurement.

*samCon*: Number  The new sample concentration associated with the measurement.

*volume*: Number  The new sample volume associated with the measurement.

*timeToResult*: Number  The new time for attaining the result associated with the measurement.

*signal*: Number  The new signal value associated with the measurement.

*repCon*: Number  The new measured concentration associated with the measurement.

A function that updates a Measurement entry.
Precondition: The user initiating the update action is a SensUsPersonnel user.

Postcondition: The database contains the measurement with the updated information.

SR - 69  
delete(): Void

A function that deletes a Measurement entry.

Precondition: The user initiating the delete action is a SensUsPersonnel user.

Postcondition: The database no longer contains the measurement.

3.1.16 GRAPH

3.1.16.1 Attributes

SR - 70  
type: GraphType.

The type of this graph. GraphType consists of Correlation, AverageVolumeTeam, AverageTimeTeam, NumPointsTeam, DoseResponse.

SR - 71  
visible: Boolean.

Flag that determines whether the graph is visible or not.

SR - 72  
axisVisible: (Boolean, Boolean).

Two flags that determine whether the x-axis and the y-axis are visible, respectively.

3.1.16.2 Operations

SR - 73  
getDataPoints(): [Measurement]

A function that retrieves the measurement data used by this graph. Can be used to count the number of data points.

Precondition: None.

Postcondition: All the returned Measurements are associated with the graph and there are no Measurements also associated with the graph that are not in the return value.

SR - 74  
correlation(): Number

A function that returns the correlation coefficient for the data points.

Precondition: None.

Postcondition: The returned value represents the correlation of the measurements associated with the graph.
getAverage(): Number

A function that returns the average for the data points (where whether the average is defined is graph type dependent).

Precondition: None.
Postcondition: The database contains the measurement with the updated information.

getStdDev(): Number

A function that returns the standard deviation for the data points (where whether the standard deviation is defined is graph type dependent).

Precondition: None.
Postcondition: The database contains the measurement with the updated information.

setVisibility(visState, xState, yState): Void

visState: Boolean The new visibility state for the graph in general.
xState: Boolean The new visibility state for the x-axis of the graph.
yState: Boolean The new visibility state for the y-axis of the graph.

A function modifies the visibility states for a graph.

Precondition: The user initiating the visibility change action is a SensUsPersonnel user.
Postcondition: The graph's visible attribute and axisVisible attribute reflect the arguments.

3.1.17 VIDEO

3.1.17.1 Attributes

static statisticsVisible: Boolean.
Static attribute, shared among all videos, that sets whether the view counts and current number of viewer statistics should be visible. The statistics themselves will be retrieved from the video hosting services.

title: String.
The title of the video.

description: String.
The description of the video.
tags: [Tag].
The tags of the video.

**SR - 82**

*status: MediaStatus.*

The status containing whether the video is Unapproved, Approved, and Rejected.

### 3.1.17.2 Operations

**SR - 83**

*static post(title, desc, link, status): Video*

- **title:** String  
  The title for the video.
- **desc:** String  
  The description for the video.
- **link:** Link  
  The link for the video.
- **status:** MediaStatus  
  The status for the video.

A function that posts a given video to the database. The status argument should be Unapproved when not submitted by SensUsPersonnel.

**Precondition:** The user initiating the upload action is an AuthenticatedUser. The status argument should be Unapproved when the submitting user is not a SensUsPersonnel user.

**Postcondition:** The database contains the video with its fields set according the arguments.

**SR - 84**

*update(title, desc, link, status, tags): Void*

- **title:** String  
  The new title for the video.
- **desc:** String  
  The new description for the video.
- **link:** Link  
  The new link for the video.
- **status:** MediaStatus  
  The new status for the video.
- **tags:** [Tag]  
  The new tags for the video.

A function that updates a given video in the database.

**Precondition:** The user initiating the update action is a SensUsPersonnel user.

**Postcondition:** The database contains the video with its fields set according the arguments.

**SR - 85**

*delete(): Void*

A function that deletes a given video from the database.

**Precondition:** The server is running and the video exists in the database.

**Postcondition:** The database no longer contains the video specified.

**SR - 86**

*static setStatVisibility(statVis): Void*

- **statVis:** Boolean  
  The new statistics visibility state for all the videos.

A function modifying the statistics visibility state for all videos.

**Precondition:** The user initiating the visibility change action is a SensUsPersonnel user.
Postcondition: The static Video statisticsVisible attribute reflects the statVis argument.

SR - 87

\textit{static filterTeam(team): [Video]}

\textit{team: Team} The team to find the videos for.

A function filters all the videos for the team.

Precondition: None.

Postcondition: The returned videos are associated with this team and no videos associated with team are missing in the return value.

3.1.18 PICTURE

3.1.18.1 Attributes

SR - 88

\textit{title: String.}

The title of the picture.

SR - 89

\textit{status: MediaStatus.}

The status containing whether the picture is either Unapproved, Approved, and Rejected.

3.1.18.2 Operations

SR - 90

\textit{static post(title, link): Picture}

\textit{title: String} The title for the new picture.

\textit{link: Link} The link for the new picture.

A function that posts a given picture to the database.

Precondition: The user initiating the upload action is a AuthenticatedUser.

Postcondition: The database contains the picture with status Unapproved.

SR - 91

\textit{update(title, status, link, tags): Void}

\textit{title: String} The new title for the picture.

\textit{status: MediaStatus} The new status for the picture.

\textit{link: Link} The new link for the picture.

\textit{tags: [Tag]} The new tags for the picture.

A function that updates a given picture’s title or status.

Precondition: The user initiating the update action is a SensUsPersonnel user.

Postcondition: The database contains the picture with the updated title, status and link and
associated tags.

SR - 92

*delete*(): Void

A function that deletes a given picture from the database.

**Precondition**: The picture exists in the database.

**Postcondition**: The database no longer contains the picture specified.

---

### 3.1.19 LINK

#### 3.1.19.1 Attributes

**SR - 93**

*sourceType*: SourceType.

The source of the link, whether it is YouTube, Imgur, etc.

**SR - 94**

*contentId*: String.

The identifier unique to the source linking to the media content, e.g. the identifier for a YouTube video.

---

### 3.1.20 STATISTICS

#### 3.1.20.1 Attributes

**SR - 95**

*numVisitors*: Number.

The amount of visitors that have accessed the SensUs Digital website.

**SR - 96**

*uptime*: Number.

The uptime of the server hosting the SensUs Digital website.

**SR - 97**

*respTime*: Number.

The average response time of the SensUs Digital website.

**SR - 98**

*availableRAM*: Number.

The available RAM (in bytes) of the server hosting the SensUs Digital website.

---

### 3.1.21 MAPPING
3.1.21.1 Attributes

SR - 99  
*associatedTeam*: Team.  
The team associated with the mapping.

SR - 100  
*sampleIDs*: [Number].  
The set of IDs associated with the samples to be measured.

SR - 101  
*concentrationValues*: [Number].  
The set of concentration values to be linked with the corresponding sample IDs.

SR - 102  
*currentMapping*: [(Number, Number)].  
The current mapping.

SR - 103  
*reserveSampleConcentrations*: [(Number, [Number])].  
The reserve sample concentrations used by each team.

3.1.21.2 Operations

SR - 104  
*static generateMapping(sampleIDs, concentrations)*: Mapping  
*sampleIDs*: [Number]  
The samples to use for the generation.  
*concentrations*: [Number]  
The concentration to be mapped to.  
A function that generates a mapping from a set of sample IDs and concentration values according mapping rules for generation.

**Precondition**: The sample IDs and concentration values allow a valid bijection.  
**Postcondition**: Each sample ID of the returned Mapping is mapped to a unique concentration value.

SR - 105  
*isValidMapping()*: Boolean  
A function that checks if the current mapping satisfies the mapping rules.

**Precondition**: The set of rules is populated and valid, and a current mapping has already been generated.  
**Postcondition**: False is returned unless all rules in the rule set are satisfied, in which case True is returned.
## 4 TRACEABILITY MATRIX

### 4.1 MAPPING OF SOFTWARE REQUIREMENTS TO USER REQUIREMENTS

<table>
<thead>
<tr>
<th>SRD</th>
<th>URD</th>
</tr>
</thead>
<tbody>
<tr>
<td>SR-1</td>
<td>URF-1, URFM-1</td>
</tr>
<tr>
<td>SR-2</td>
<td>URF-84, URF-85, URF-86</td>
</tr>
<tr>
<td>SR-3</td>
<td>URF-4</td>
</tr>
<tr>
<td>SR-4</td>
<td>URF-1, URF-2</td>
</tr>
<tr>
<td>SR-5</td>
<td>URF-3</td>
</tr>
<tr>
<td>SR-6</td>
<td>URF-1, URF-2</td>
</tr>
<tr>
<td>SR-7</td>
<td>URF-1</td>
</tr>
<tr>
<td>SR-8</td>
<td>URF-2, URFM-2</td>
</tr>
<tr>
<td>SR-9</td>
<td>URF-68, URF-69, URF-70, URFM-4, URFM-61</td>
</tr>
<tr>
<td>SR-10</td>
<td>URF-127, URF-128, URF-130, URF-131, URF-133, URF-134</td>
</tr>
<tr>
<td>SR-11</td>
<td>URF-4</td>
</tr>
<tr>
<td>SR-12</td>
<td>URF-174, URF-175, URF-176</td>
</tr>
<tr>
<td>SR-13</td>
<td>URF-5</td>
</tr>
<tr>
<td>SR-14</td>
<td>URF-8, URF-9</td>
</tr>
<tr>
<td>SR-15</td>
<td>URF-6</td>
</tr>
<tr>
<td>SR-16</td>
<td>URF-7A, URF-7B</td>
</tr>
<tr>
<td>SR-17</td>
<td>URF-45, URF-67, URFM-10</td>
</tr>
<tr>
<td>SR-18</td>
<td>URF-45</td>
</tr>
<tr>
<td>SR-19</td>
<td>URF-45</td>
</tr>
<tr>
<td>SR-20</td>
<td>URF-45</td>
</tr>
<tr>
<td>SR-21</td>
<td>URF-45, URFM-10</td>
</tr>
<tr>
<td>SR-22</td>
<td>URF-48</td>
</tr>
<tr>
<td>SR-23</td>
<td>URF-45, URF-47, URF-64, URF-65</td>
</tr>
<tr>
<td>SR-24</td>
<td>URF-46</td>
</tr>
<tr>
<td>SR-25</td>
<td>URF-163, URF-173</td>
</tr>
<tr>
<td>SR-26</td>
<td>URF-46</td>
</tr>
<tr>
<td>SR-27</td>
<td>URF-6</td>
</tr>
<tr>
<td>SR-29</td>
<td>URF-109</td>
</tr>
<tr>
<td>SR-30</td>
<td>URF-52, URF-53</td>
</tr>
<tr>
<td>SR-31</td>
<td>URF-55, URFM-12</td>
</tr>
<tr>
<td>SR-32</td>
<td>URF-56, URF-57</td>
</tr>
<tr>
<td>SR-33</td>
<td>URF-50, URF-51</td>
</tr>
<tr>
<td>SR-34</td>
<td>URF-58</td>
</tr>
<tr>
<td>------</td>
<td>----------------</td>
</tr>
<tr>
<td>SR-35</td>
<td>URF-42, URF-87, URF-88, URF-89</td>
</tr>
<tr>
<td>SR-36</td>
<td>URF-49</td>
</tr>
<tr>
<td>SR-37</td>
<td>URF-35</td>
</tr>
<tr>
<td>SR-38</td>
<td>URF-49, URF-66A</td>
</tr>
<tr>
<td>SR-39</td>
<td>URF-35, URF-71, URF-72, URF-73, URF-74, URF-75</td>
</tr>
<tr>
<td>SR-40</td>
<td>URF-145</td>
</tr>
<tr>
<td>SR-41</td>
<td>URF-90B, URF-145</td>
</tr>
<tr>
<td>SR-43</td>
<td>URF-44A</td>
</tr>
<tr>
<td>SR-44</td>
<td>URF-44A, URF-129, URF-132</td>
</tr>
<tr>
<td>SR-46</td>
<td>URF-139, URF-140, URF-141, URF-142, URF-145, URF-146, URF-147, URF-148, URF-149, URF-150</td>
</tr>
<tr>
<td>SR-47</td>
<td>URF-147, URF-148, URF-151, URF-152</td>
</tr>
<tr>
<td>SR-49</td>
<td>URF-145</td>
</tr>
<tr>
<td>SR-50</td>
<td>URF-145</td>
</tr>
<tr>
<td>SR-51</td>
<td>URF-124, URF-126</td>
</tr>
<tr>
<td>SR-52</td>
<td>URF-157, URF-158, URF-159, URF-161, URF-162</td>
</tr>
<tr>
<td>SR-53</td>
<td>URF-155, URF-156</td>
</tr>
<tr>
<td>SR-54</td>
<td>URF-39, URF-40, URF-41, URF-43</td>
</tr>
<tr>
<td>SR-55</td>
<td>URF-36, URF-37</td>
</tr>
<tr>
<td>SR-56</td>
<td>URF-40, URF-41</td>
</tr>
<tr>
<td>SR-57</td>
<td>URF-103</td>
</tr>
<tr>
<td>SR-58</td>
<td>URF-11, URF-12, URF-13, URF-14, URF-44B, URF-44C, URFM-3</td>
</tr>
<tr>
<td>SR-59</td>
<td>URF-11, URF-12, URF-13, URF-14, URF-44B, URF-44C</td>
</tr>
<tr>
<td>SR-60</td>
<td>URF-33</td>
</tr>
<tr>
<td>SR-61</td>
<td>URF-15, URF-76A, URF-77, URF-83C, URFM-11</td>
</tr>
<tr>
<td>SR-63</td>
<td>URF-15, URF-76A, URF-77, URF-83C, URFM-11</td>
</tr>
<tr>
<td>SR-64</td>
<td>URF-15, URF-76A, URF-77, URF-83C, URFM-11</td>
</tr>
<tr>
<td>SR-65</td>
<td>URF-15, URF-76A, URF-77, URF-83C, URFM-11</td>
</tr>
<tr>
<td>Number</td>
<td>URF-IDs</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>SR-68</td>
<td>URF-76A, URF-76B, URF-83B</td>
</tr>
<tr>
<td>SR-69</td>
<td>URF-78</td>
</tr>
<tr>
<td>SR-70</td>
<td>URF-15, URF-16, URF-20, URF-21, URF-22, URF-23, URF-23, URF-24, URF-79, URF-80, URF-81, URF-83A</td>
</tr>
<tr>
<td>SR-71</td>
<td>URF-25, URF-26, URF-27, URF-28, URF-29, URF-30A, URF-30B</td>
</tr>
<tr>
<td>SR-72</td>
<td>URF-25, URF-26, URF-27, URF-28, URF-29, URF-30A, URF-30B</td>
</tr>
<tr>
<td>SR-73</td>
<td>URF-17, URF-82</td>
</tr>
<tr>
<td>SR-74</td>
<td>URF-19</td>
</tr>
<tr>
<td>SR-75</td>
<td>URF-21, URF-81</td>
</tr>
<tr>
<td>SR-76</td>
<td>URF-18</td>
</tr>
<tr>
<td>SR-77</td>
<td>URF-27, URF-28, URF-29, URF-30A</td>
</tr>
<tr>
<td>SR-78</td>
<td>URF-31, URF-32, URF-33, URF-34, URF-38, URF-43, URF-96, URF-110</td>
</tr>
<tr>
<td>SR-79</td>
<td>URF-90A, URF-110, URF-118, URF-119, URF-120</td>
</tr>
<tr>
<td>SR-80</td>
<td>URF-90A</td>
</tr>
<tr>
<td>SR-83</td>
<td>URF-93, URF-107, URF-108, URFM-5</td>
</tr>
<tr>
<td>SR-84</td>
<td>URF-100, URF-101, URF-102, URF-103, URF-104, URF-105</td>
</tr>
<tr>
<td>SR-86</td>
<td>URF-40, URF-41</td>
</tr>
<tr>
<td>SR-87</td>
<td>URF-117</td>
</tr>
<tr>
<td>SR-88</td>
<td>URF-63</td>
</tr>
<tr>
<td>SR-89</td>
<td>URF-60, URF-61, URF-62, URF-63</td>
</tr>
<tr>
<td>SR-90</td>
<td>URF-59</td>
</tr>
<tr>
<td>SR-91</td>
<td>URF-60, URF-61, URF-62, URF-66C</td>
</tr>
<tr>
<td>SR-92</td>
<td>URF-66B</td>
</tr>
<tr>
<td>SR-93</td>
<td>URF-90A</td>
</tr>
<tr>
<td>SR-94</td>
<td>URF-90A</td>
</tr>
<tr>
<td>SR-95</td>
<td>URF-183</td>
</tr>
<tr>
<td>SR-96</td>
<td>URF-184</td>
</tr>
<tr>
<td>SR-97</td>
<td>URF-185</td>
</tr>
<tr>
<td>SR-98</td>
<td>URF-186</td>
</tr>
<tr>
<td>SR-99</td>
<td>URF-182</td>
</tr>
<tr>
<td>SR-100</td>
<td>URF-177, URF-178</td>
</tr>
<tr>
<td>SR-101</td>
<td>URF-177, URF-178</td>
</tr>
</tbody>
</table>
4.2 MAPPING OF USER REQUIREMENTS TO SOFTWARE REQUIREMENTS

<table>
<thead>
<tr>
<th>URD</th>
<th>SRD</th>
</tr>
</thead>
<tbody>
<tr>
<td>URF-1</td>
<td>SR-1,SR-4,SR-6,SR-7</td>
</tr>
<tr>
<td>URF-2</td>
<td>SR-4,SR-6,SR-8</td>
</tr>
<tr>
<td>URF-3</td>
<td>SR-5</td>
</tr>
<tr>
<td>URF-4</td>
<td>SR-3,SR-11</td>
</tr>
<tr>
<td>URF-5</td>
<td>SR-13</td>
</tr>
<tr>
<td>URF-6</td>
<td>SR-15,SR-27</td>
</tr>
<tr>
<td>URF-7A</td>
<td>SR-16</td>
</tr>
<tr>
<td>URF-7B</td>
<td>SR-16</td>
</tr>
<tr>
<td>URF-8</td>
<td>SR-14</td>
</tr>
<tr>
<td>URF-9</td>
<td>SR-14</td>
</tr>
<tr>
<td>URF-11</td>
<td>SR-58,SR-59</td>
</tr>
<tr>
<td>URF-12</td>
<td>SR-58,SR-59</td>
</tr>
<tr>
<td>URF-13</td>
<td>SR-58,SR-59</td>
</tr>
<tr>
<td>URF-14</td>
<td>SR-58,SR-59</td>
</tr>
<tr>
<td>URF-16</td>
<td>SR-70</td>
</tr>
<tr>
<td>URF-17</td>
<td>SR-73</td>
</tr>
<tr>
<td>URF-18</td>
<td>SR-76</td>
</tr>
<tr>
<td>URF-19</td>
<td>SR-74</td>
</tr>
<tr>
<td>URF-20</td>
<td>SR-70</td>
</tr>
<tr>
<td>URF-21</td>
<td>SR-70,SR-75</td>
</tr>
<tr>
<td>URF-22</td>
<td>SR-70</td>
</tr>
<tr>
<td>URF-23</td>
<td>SR-70</td>
</tr>
<tr>
<td>URF-24</td>
<td>SR-70</td>
</tr>
<tr>
<td>URF-25</td>
<td>SR-71,SR-72</td>
</tr>
<tr>
<td>URF-26</td>
<td>SR-71,SR-72</td>
</tr>
<tr>
<td>URF-27</td>
<td>SR-71,SR-72,SR-77</td>
</tr>
<tr>
<td>URF-28</td>
<td>SR-71,SR-72,SR-77</td>
</tr>
<tr>
<td>URF-29</td>
<td>SR-71,SR-72,SR-77</td>
</tr>
<tr>
<td>URF</td>
<td>SR-Numbers</td>
</tr>
<tr>
<td>------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>URF-30A</td>
<td>SR-71,SR-72,SR-77</td>
</tr>
<tr>
<td>URF-30B</td>
<td>SR-71,SR-72</td>
</tr>
<tr>
<td>URF-31</td>
<td>SR-78</td>
</tr>
<tr>
<td>URF-32</td>
<td>SR-78</td>
</tr>
<tr>
<td>URF-33</td>
<td>SR-60,SR-78</td>
</tr>
<tr>
<td>URF-34</td>
<td>SR-78</td>
</tr>
<tr>
<td>URF-35</td>
<td>SR-37,SR-39</td>
</tr>
<tr>
<td>URF-36</td>
<td>SR-55</td>
</tr>
<tr>
<td>URF-37</td>
<td>SR-55</td>
</tr>
<tr>
<td>URF-38</td>
<td>SR-78</td>
</tr>
<tr>
<td>URF-39</td>
<td>SR-54</td>
</tr>
<tr>
<td>URF-40</td>
<td>SR-54,SR-56,SR-86</td>
</tr>
<tr>
<td>URF-41</td>
<td>SR-54,SR-56,SR-86</td>
</tr>
<tr>
<td>URF-42</td>
<td>SR-35</td>
</tr>
<tr>
<td>URF-43</td>
<td>SR-54,SR-78</td>
</tr>
<tr>
<td>URF-44A</td>
<td>SR-43,SR-44</td>
</tr>
<tr>
<td>URF-44B</td>
<td>SR-58,SR-59</td>
</tr>
<tr>
<td>URF-44C</td>
<td>SR-58,SR-59</td>
</tr>
<tr>
<td>URF-45</td>
<td>SR-17,SR-18,SR-19,SR-20,SR-21,SR-23</td>
</tr>
<tr>
<td>URF-46</td>
<td>SR-24,SR-26</td>
</tr>
<tr>
<td>URF-47</td>
<td>SR-23</td>
</tr>
<tr>
<td>URF-48</td>
<td>SR-22</td>
</tr>
<tr>
<td>URF-49</td>
<td>SR-36,SR-38</td>
</tr>
<tr>
<td>URF-50</td>
<td>SR-33</td>
</tr>
<tr>
<td>URF-51</td>
<td>SR-33</td>
</tr>
<tr>
<td>URF-52</td>
<td>SR-30</td>
</tr>
<tr>
<td>URF-53</td>
<td>SR-30</td>
</tr>
<tr>
<td>URF-55</td>
<td>SR-31</td>
</tr>
<tr>
<td>URF-56</td>
<td>SR-32</td>
</tr>
<tr>
<td>URF-57</td>
<td>SR-32</td>
</tr>
<tr>
<td>URF-58</td>
<td>SR-34</td>
</tr>
<tr>
<td>URF-59</td>
<td>SR-90</td>
</tr>
<tr>
<td>URF-60</td>
<td>SR-89,SR-91</td>
</tr>
<tr>
<td>URF-61</td>
<td>SR-89,SR-91</td>
</tr>
<tr>
<td>URF-62</td>
<td>SR-89,SR-91</td>
</tr>
<tr>
<td>URF-63</td>
<td>SR-88,SR-89</td>
</tr>
<tr>
<td>URF-64</td>
<td>SR-23</td>
</tr>
<tr>
<td>Valedictorian</td>
<td>Software Requirements Document</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------</td>
</tr>
</tbody>
</table>

<p>| URF-65        | SR-23                         |
| URF-66A       | SR-38                         |
| URF-66B       | SR-92                         |
| URF-66C       | SR-91                         |
| URF-67        | SR-17                         |
| URF-68        | SR-9                          |
| URF-69        | SR-9                          |
| URF-70        | SR-9                          |
| URF-71        | SR-39                         |
| URF-72        | SR-39                         |
| URF-73        | SR-39                         |
| URF-74        | SR-39                         |
| URF-75        | SR-39                         |
| URF-76B       | SR-68                         |
| URF-78        | SR-69                         |
| URF-79        | SR-70                         |
| URF-80        | SR-70                         |
| URF-81        | SR-70,SR-75                   |
| URF-82        | SR-73                         |
| URF-83A       | SR-70                         |
| URF-83B       | SR-68                         |
| URF-84        | SR-2                          |
| URF-85        | SR-2                          |
| URF-86        | SR-2                          |
| URF-87        | SR-35                         |
| URF-88        | SR-35                         |
| URF-89        | SR-35                         |
| URF-90B       | SR-41,SR-45                   |
| URF-91        | SR-45                         |
| URF-92        | SR-45                         |
| URF-93        | SR-83                         |
| URF-94        | SR-85                         |
| URF-96        | SR-78,SR-82                   |</p>
<table>
<thead>
<tr>
<th>URF-97</th>
<th>SR-82</th>
</tr>
</thead>
<tbody>
<tr>
<td>URF-98</td>
<td>SR-82</td>
</tr>
<tr>
<td>URF-99</td>
<td>SR-82</td>
</tr>
<tr>
<td>URF-100</td>
<td>SR-84</td>
</tr>
<tr>
<td>URF-101</td>
<td>SR-84</td>
</tr>
<tr>
<td>URF-102</td>
<td>SR-84</td>
</tr>
<tr>
<td>URF-103</td>
<td>SR-57,SR-84</td>
</tr>
<tr>
<td>URF-104</td>
<td>SR-84</td>
</tr>
<tr>
<td>URF-105</td>
<td>SR-84</td>
</tr>
<tr>
<td>URF-106</td>
<td>SR-85</td>
</tr>
<tr>
<td>URF-109</td>
<td>SR-29</td>
</tr>
<tr>
<td>URF-110</td>
<td>SR-78,SR-79</td>
</tr>
<tr>
<td>URF-111</td>
<td>SR-81</td>
</tr>
<tr>
<td>URF-112</td>
<td>SR-81</td>
</tr>
<tr>
<td>URF-113</td>
<td>SR-81</td>
</tr>
<tr>
<td>URF-114</td>
<td>SR-81</td>
</tr>
<tr>
<td>URF-115</td>
<td>SR-81</td>
</tr>
<tr>
<td>URF-116</td>
<td>SR-81</td>
</tr>
<tr>
<td>URF-117</td>
<td>SR-87</td>
</tr>
<tr>
<td>URF-118</td>
<td>SR-79</td>
</tr>
<tr>
<td>URF-119</td>
<td>SR-79</td>
</tr>
<tr>
<td>URF-120</td>
<td>SR-79</td>
</tr>
<tr>
<td>URF-121</td>
<td>SR-81</td>
</tr>
<tr>
<td>URF-122</td>
<td>SR-81</td>
</tr>
<tr>
<td>URF-123</td>
<td>SR-42</td>
</tr>
<tr>
<td>URF-124</td>
<td>SR-51</td>
</tr>
<tr>
<td>URF-125</td>
<td>SR-42</td>
</tr>
<tr>
<td>URF-126</td>
<td>SR-51</td>
</tr>
<tr>
<td>URF-127</td>
<td>SR-10</td>
</tr>
<tr>
<td>URF-128</td>
<td>SR-10</td>
</tr>
<tr>
<td>URF-129</td>
<td>SR-44</td>
</tr>
<tr>
<td>URF-130</td>
<td>SR-10</td>
</tr>
<tr>
<td>URF-131</td>
<td>SR-10</td>
</tr>
<tr>
<td>URF-132</td>
<td>SR-44</td>
</tr>
<tr>
<td>URF-133</td>
<td>SR-10</td>
</tr>
<tr>
<td>URF-134</td>
<td>SR-10</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td>URF-135</td>
<td>SR-45</td>
</tr>
<tr>
<td>URF-136</td>
<td>SR-45</td>
</tr>
<tr>
<td>URF-138</td>
<td>SR-45</td>
</tr>
<tr>
<td>URF-139</td>
<td>SR-46</td>
</tr>
<tr>
<td>URF-140</td>
<td>SR-46</td>
</tr>
<tr>
<td>URF-141</td>
<td>SR-46</td>
</tr>
<tr>
<td>URF-142</td>
<td>SR-46</td>
</tr>
<tr>
<td>URF-146</td>
<td>SR-46</td>
</tr>
<tr>
<td>URF-147</td>
<td>SR-46, SR-47</td>
</tr>
<tr>
<td>URF-148</td>
<td>SR-45, SR-46, SR-47</td>
</tr>
<tr>
<td>URF-149</td>
<td>SR-46</td>
</tr>
<tr>
<td>URF-150</td>
<td>SR-46</td>
</tr>
<tr>
<td>URF-151</td>
<td>SR-47</td>
</tr>
<tr>
<td>URF-152</td>
<td>SR-47</td>
</tr>
<tr>
<td>URF-153</td>
<td>SR-48</td>
</tr>
<tr>
<td>URF-154</td>
<td>SR-48</td>
</tr>
<tr>
<td>URF-155</td>
<td>SR-53</td>
</tr>
<tr>
<td>URF-156</td>
<td>SR-53</td>
</tr>
<tr>
<td>URF-157</td>
<td>SR-52</td>
</tr>
<tr>
<td>URF-158</td>
<td>SR-52</td>
</tr>
<tr>
<td>URF-159</td>
<td>SR-52</td>
</tr>
<tr>
<td>URF-161</td>
<td>SR-52</td>
</tr>
<tr>
<td>URF-162</td>
<td>SR-52</td>
</tr>
<tr>
<td>URF-163</td>
<td>SR-25</td>
</tr>
<tr>
<td>URF-164</td>
<td>SR-42</td>
</tr>
<tr>
<td>URF-165</td>
<td>SR-42</td>
</tr>
<tr>
<td>URF-166</td>
<td>SR-42</td>
</tr>
<tr>
<td>URF-167</td>
<td>SR-42</td>
</tr>
<tr>
<td>URF-168</td>
<td>SR-42</td>
</tr>
<tr>
<td>URF-169</td>
<td>SR-42</td>
</tr>
<tr>
<td>URF-170</td>
<td>SR-42</td>
</tr>
<tr>
<td>URF-171</td>
<td>SR-42</td>
</tr>
<tr>
<td>URF-172</td>
<td>SR-42</td>
</tr>
<tr>
<td>URF-173</td>
<td>SR-25</td>
</tr>
<tr>
<td>URF-174</td>
<td>SR-12</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td>URF-175</td>
<td>SR-12</td>
</tr>
<tr>
<td>URF-176</td>
<td>SR-12</td>
</tr>
<tr>
<td>URF-177</td>
<td>SR-100,SR-101,SR-104</td>
</tr>
<tr>
<td>URF-178</td>
<td>SR-100,SR-101,SR-104</td>
</tr>
<tr>
<td>URF-179</td>
<td>SR-105</td>
</tr>
<tr>
<td>URF-180</td>
<td>SR-102</td>
</tr>
<tr>
<td>URF-181</td>
<td>SR-103</td>
</tr>
<tr>
<td>URF-182</td>
<td>SR-99</td>
</tr>
<tr>
<td>URF-183</td>
<td>SR-95</td>
</tr>
<tr>
<td>URF-184</td>
<td>SR-96</td>
</tr>
<tr>
<td>URF-185</td>
<td>SR-97</td>
</tr>
<tr>
<td>URF-186</td>
<td>SR-98</td>
</tr>
<tr>
<td>URFM-1</td>
<td>SR-1</td>
</tr>
<tr>
<td>URFM-2</td>
<td>SR-8</td>
</tr>
<tr>
<td>URFM-3</td>
<td>SR-58</td>
</tr>
<tr>
<td>URFM-4</td>
<td>SR-9</td>
</tr>
<tr>
<td>URFM-5</td>
<td>SR-83</td>
</tr>
<tr>
<td>URFM-6</td>
<td>SR-45</td>
</tr>
<tr>
<td>URFM-7</td>
<td>SR-45</td>
</tr>
<tr>
<td>URFM-8</td>
<td>SR-45</td>
</tr>
<tr>
<td>URFM-9</td>
<td>SR-45</td>
</tr>
<tr>
<td>URFM-10</td>
<td>SR-17,SR-21</td>
</tr>
<tr>
<td>URFM-12</td>
<td>SR-31</td>
</tr>
<tr>
<td>URFM-61</td>
<td>SR-9</td>
</tr>
</tbody>
</table>
A  APPENDIX - UI LAYOUTS

In this appendix we present mock-up pages for the views that we intend to implement. Note that these mockups might not all be implemented in the final product as some are derived from should have requirements from the URD [1].

A.1  HOME PAGE

The user is presented with the home page when they visit the SensUs Digital website, as illustrated in Figure 52. As the user has not been logged-in yet, they only have the options to visit the unrestricted pages through the tabs in the header. On the home page itself, the user can view the main live stream and read various tidbits describing different aspects of the contest, such as an explanation of the event, the number of teams and where they are from, and the chosen biomarker. From this view, the user can initiate an upload via the upload button pinned to the left side of the page.
A.1.1 UPLOADING

The upload modal (Figure 53) provides users the option to upload a picture, video, or question. The type of upload is indicated with a button-style selection, and the URL and title belonging to the upload are entered into the corresponding text fields. The user can also provide an optional description before submitting the upload link via the “Submit” button.

A.2 VOTING

On the “Vote” page, the user will be presented with the description of the prizes available for teams to win, as the mockup in Figure 54 illustrates. Further down the page, the user will have
the option to vote, view the voting statistics, and global heat map.

### A.2.1 CASTING VOTES

A set of tiles are presented to the user, each containing a specific team's image and name, see Figure 55. To view information about the team, the info button in the upper-right corner, upon being clicked, will present the user with a description and picture of the team as a modal, see Figure 56.

![Team Info](image)

To cast a vote for the two teams of choice, two tiles can be selected and the "Cast Vote" button can be clicked. If a user wishes to deselect a team they have already chosen, they can click the selected team once again to remove their selection.
A.2.2 VOTING STATISTICS

In Figure 57 we have a user view when the user has already voted. Such a user will have the ability to view statistics about the total votes collected for each team.

A.2.3 HEAT MAP

In addition to voting statistics, there will be a heat map that displays vote frequency based on country (see Figure 58). The geolocation is done based on IP address of the voter.
A.3 EXPLORE

On the explore page, Figure 59, all videos are displayed in a tiled format based on a chosen filter. The user can opt to filter on tag type or media type, with sorting options such as “Random” and “Newest First”. There is a continuous loading of videos (so called “infinite scrolling”) as the user scrolls down the page.

A.4 TEAMS

The “Teams” page, as shown in Figure 60, provides a card for each participating team containing the name of the team, associated university, and country of origin. Each card is clickable and
links to their associated team info page.

### A.4.1 TEAM INFO PAGE

![Team Info Page](image)

**FIGURE 61: TEAM INFO PAGE**

Each informational team page (see Figure 61) contains an enlarged picture of the team, with a team video and biosensor chart displayed side by side. Below are various tabs with a description of the team, questions and answers directed to/from the team, associated media, social feed, and an option to support the team via donation.

### A.5 Q&A

![Q&A Page Layout](image)

**FIGURE 62: Q&A PAGE LAYOUT**
The Q&A page provides a form for a user to ask questions to the teams, as can be seen in Figure 62. A question must contain a title and an associated team. Optionally, a URL and/or textual description can be provided in the correct fields. To enable the URL and text fields, the respective buttons must be pressed. Finally, a question can be sent to the team by clicking the “Submit” button. All the approved questions and answers will be displayed on this page as well.

A.6 LOGIN

When the user clicks on the “Login” tab, it will redirect them to a standard Google login form (see Figure 63). As we use Google login for our users, this view will redirect back to the home page when the user successfully authenticates.
A.7 DASHBOARD

The dashboard page, as the mockup in Figure 64 illustrates, acts as a portal for team member users (and users with more privileges) to access dashboards within their specified privileges. Each of the dashboards displayed are clickable and redirect the user to the corresponding dashboard page.

A.8 TEAM DASHBOARD

The “Team” dashboard is only accessible to administrators, see Figure 65. It contains the button, “Select an image”, to edit the current team image along with an editable description...
associated with the team. The team info can be updated via the “Update” button, and videos can be uploaded via the “Upload” button. A list of team videos will be displayed in a list-style view with an option to delete the video from public access through the “Remove” button.

A.9 DATA DASHBOARD

![Data Dashboard Image]

**FIGURE 66: DATA DASHBOARD PAGE LAYOUT**

The “Data” dashboard contains an in-browser, spreadsheet-style editor for biosensor data. Figure 66 contains an early version. There is an option for teams to upload “.csv” files to populate the grid. Each cell can be edited, and new rows and columns can be added at will.
A.10 EVENT DASHBOARD

The “Event” dashboard of Figure 67 contains three main components: statistics, slideshow preview, and display options. The statistics will display relevant graphs from the event, and can be selected/deselected to enable/disable their presence in the slideshow preview. New statistics can be added to the left rail with the “Add Item” button. The slideshow preview scrolls through the statistics in a gallery format, which can be full-screened (among other displayed options) for exhibition at the event.

A.11 Q&A DASHBOARD

The “Q&A” dashboard of Figure 68 contains a list of questions and answers. Each question has a status indicating whether it is waiting to be answered, approved, rejected, or claimed. The dashboard also shows a section for questions that have already been answered and approved.
The Q&A dashboard (see Figure 68) contains a list of all questions submitted to the SensUs Digital website. For ease of navigability, all questions can be viewed in the “All” tab, but specific subsets of these questions can be observed in their corresponding tabs, i.e. “Unapproved” for unapproved questions, “Approved” for approved questions, and so on. The main color for a question indicates the status of the question itself and the small circle on the right indicates the status of any associated answer.

Upon expanding a question, the question title, status, description, and optional video can be viewed, as in Figure 69. If there is an answer already associated with the selected question, it will appear in the “Answer” drop-down menu.

The status of the question (pending, approved, or rejected) can be changed by selecting the
new desired status. A confirmation modal (see Figure 70) will appear to warn the user of the effects associated with the change, from which a user can cancel or confirm the action.

**A.12 CONTROL DASHBOARD**

![Control Dashboard Page Layout](image)

The “Control” dashboard contains a list of media titles on the left rail, followed by a display of the media with media info, see Figure 71. The media can be a video or image, each with a number of descriptors. They can be filtered by type (all, videos, pictures), state (all, approved, unapproved, rejected), and sorted on a number of selectors (uploaded, name, type). To the right of the media, there are buttons to update, upload, and refresh. The media URL, title, description, tags, associated teams, and status can be changed by altering their corresponding fields and submitting via the “Submit” button at the bottom of the page. The media can also be deleted via the “delete” button.
In the “Upload” tab, a picture/video can be uploaded by entering a source URL, title, and other descriptors, as shown in Figure 72. Finally, the “Submit” button must be pressed to complete the submission.

B TRANSITIONS

We will now describe the actions the user can take to switch between the interfaces previously described. All of the layouts are accessible via the web browser, so at any time the user can close their browser and terminate connection with the SensUs Digital website. Note that the browsing session will be maintained and that the user can resume their logged in after starting up their browser again.

B.1 UNAUTHORIZED-USER TRANSITIONS

When a user first lands on the website, he will arrive at the home page with the option to access any of the pages linked by tabs in the header. Since a first-time user has not authenticated he will be limited to the tabs “Vote Now”, “Explore”, “Teams”, “Q&A”, “Social”, and “Login”. To return to the home page from any of these tabs, the SensUs logo can be clicked. It should be noted that navigation from each page to any other page is mediated by the tabs in the header.

B.2 AUTHORIZED-USER TRANSITIONS

To access pages only available to authorized users, the user must login, which requires clicking the "Login" tab and providing login credentials to the Google sign-in page. When the user has successfully logged in, they will be redirected to the SensUs SensUs Digital website, and will
have an extra tab “Dashboard” should they have the required permissions to access the dashboards. Within the “Dashboard” tab, each of the available dashboards to their user type are viewable/clickable in the dashboard landing page as well as the (now blue) header. Each of the dashboards are linked through the header, and the tabs accessible to only unauthorized users can be restored by navigating to the “SensUs Home” tab.

B.3 DIAGRAM

![Diagram](image)

**FIGURE 73: TRANSITION DIAGRAM**

The above diagram illustrates the pages the user can visit from each page in the website. The squares are the possible states, i.e. shown pages, with the colors indicating the permission level that is required for a page (red indicating an unauthenticated user). The circles are transitions, which happen through an interaction with the menu, with the arrows indicating that a transition is possible from a page. To access a certain page the user must have the permission level associated with that color, i.e. at least that user level. The Main Website is available to any user (from unauthenticated users up to website admins), indicated by the red color. To move from the Home Page to any page on the Dashboard it is required that the user is at least a Team Member, indicated by the orange color for the arrows from the Main Website to the Dashboard and the arrows colors for any arrow to non-Home Page pages in Dashboard.