# System design document for Grey Matter

Aline Eikeland, Hugo Ganelius, Viktor Johansson,
Emil Lundqvist, Felix Oliv

date
version

## 1 Introduction

Give an introduction to the document and your application.

### 1.1 Definitions, acronyms, and abbreviations

Definitions etc. probably same as in RAD

Standalone application - A application that runs locally on the device and doesn't require anything else to be functional.

## 2 System architecture

The most overall, top level description of your application. If your application uses multiple components (such as servers, databases, etc.), describe their responsibilities here and show how they are dependent on each other and how they communicate (which protocols etc.)

You will to describe the 'flow' of the application at a high level. What happens if the application is started (and later stopped) and what the normal flow of operation is. Relate this to the different components (if any) in your application.

GrayMatter is a mostly standalone application, and all the logic is therefore built into the app. However to be able to compare your score with other players, the app needs a internet connection to download other players data which is located on a server.

### 2.1 Flow of the application

When the application is started the method on-Create in the Main-activity class is called. This method will instantiate the main-page XML file which displays the main-page to the,

## 3 System design

Draw an UML package diagram for the top level for all components that you have identified above (which can be just one if you develop a standalone application). Describe the interfaces and dependencies between the packages. Describe how you have implemented the MVC design pattern.

Create an UML class diagram for every package. One of the packages will contain the model of your application. This will be the design model of your application, describe in detail the relation between your domain model and your design model. There should be a clear and logical relation between the two. Make sure that these models stay in 'sync' during the development of your application.

Describe which (if any) design patterns you have used.

The above describes the static design of your application. It may sometimes be necessary to describe the *dynamic* design of your application as well. You can use an UML *sequence diagram* to show the different parts of your application communicate an in what order.

## 4 Persistent data management

If your application makes use of persistent data (for example stores user profiles etc.), then explain how you store data (and other resources such as icons, images, audio, etc.).

## 5 Quality

- Describe how you test your application and where to find these tests. If applicable, give a link to your continuous integration.
- List all known issues.
- Run analytical tools on your software and show the results. Use for example:
  - Dependencies: STAN or similar.
  - Quality tool reports, like PMD.

NOTE: Each Java, XML, etc. file should have a header comment: Author, responsibility, used by ..., uses ..., etc.

## 5.1 Access control and security

If you applications has some kind of access control, for example a login, of has different user roles (admin, standard, etc.), then explain how you application manages this.

# 6 References

List all references to external tools, platforms, libraries, papers, etc. The purpose is that the reader can find additional information quickly and use this to understand how your application works.