Design 1

Violation of MVC Layers 1

Usage of helper objects between view and model 1

Rich OO domain model 1

Clear responsibilities 2

Sound invariants 2

Overall code organization & reuse, e.g views 2

Coding style 2

Consistency 2

Intention-revealing names 2

Do not repeat yourself 2

Exception, testing null values 2

Encapsulation 2

Assertion, contracts, invariant checks 2

Utility methods 2

Documentation 2

SRS-Document 2

Understandable 2

Intention-revealing 3

Describe responsibilities 3

Match a consistent domain vocabulary 3

Test 3

Clear and distinct test cases 3

Number/coverage of test cases 3

Easy to understand the case that is tested 3

Well crafted set of test data 3

Readability 3

Review Group 7 Mensa@Unibe

# Design

## Violation of MVC Layers

The usage of “Model-View-Controller” as a software architectural pattern, which separates the representation of information from the interaction whit it, is clearly realised over the entire project. In general the pattern is provided by the architectural structure of an android application.

Clear structured source tree in “Model” and “Controller” packages so also the “Views” in the activity layout Folder.

## Usage of helper objects between view and model

Good usage of helper objects between view and model.

## Rich OO domain model

The code contains a lot of useful domain models. The problems attributes are clearly shown and modelled in the code. The Mensa object is built trough a builder in and implemented in proper OOP manner.

The SRS document could contain more relationships to the modelled domain objects (there are more domain model in the code as in the diagram ).

## Clear responsibilities

Verantwortungen sind klar unterteilt unter größtmöglicher nutzung der Android API

## Sound invariants

Habe keine überprüfungen gefunden

## Overall code organization & reuse, e.g views

Sieht gut aus

# Coding style

## Consistency

Top

## Intention-revealing names

Namen der Klassen und Methoden sind gut gewählt

## Do not repeat yourself

Kommentare wiederholen teils nur was der Methodenname scho aussagt.

## Exception, testing null values

Tries mit Logausgabe. Es wird nach nullvalues geprüft.

## Encapsulation

Sieht aus als ist alles private was private sein sollte

## Assertion, contracts, invariant checks

Einige assertions etc. gut umgesetzt

## Utility methods

Hat es einige.

# Documentation

## SRS-Document

Missing a table of contents at the beginning of the Document.

The SRS-Document was kept up to date in a good manner.

The purpose is clearly defined and understandable.

Under The point Stakeholders is not clear from where does the knowledge comes, that 75% of students at the Unibe are android user?

The System overview contains all of the implemented features.

In general the use cases are very short but they contain the necessary information’s.

In point 2.2 what does it mean “average intelligence”.

Good overview of requirements, missing newest acquired knowledge about the app.

## Understandable

The java-doc are mostly present ant helpful. No presence of java-doc in the java activity`s. In some cases the documentation is redundant, mostly there is a redundancy between a @return statement and the overview or description of the method.

Example:

/\*\*

\* actually reads the whole file and returns it as string

\* @return the file content as a string

\*/

/\*\*

\* reads the user's account email and returns a hash of it

\* @return the md5 hash of the user's email

\*/

## Intention-revealing

Gibt Details an wenn es welche anzugeben gibt.

## Describe responsibilities

Jop

## Match a consistent domain vocabulary

Jop

# Test

## Clear and distinct test cases

Ja

## Number/coverage of test cases

Tests für die wichtigsten Klassen. Einige kleinere wurden aussen vor gelassen.

## Easy to understand the case that is tested

Übersichtlich und dazu gut dokumentiert

## Well crafted set of test data

Mockobjekte vorhanden. Also ja

## Readability

Sauber und gut dokumentiert.