Software Architecture Document (SAD)

Project: PharmaPartners

Project team: S3-DB03T group 3

Team members:

Name	E-mail
Bart Blaak	b.blaak@student.fontys.nl
Sibe van Etten	sibe.vanetten@student.fontys.nl
Hugo Mkandawire	h.mkandawire@student.fontys.nl
Maarten Reimus	m.reimus@student.fontys.nl
Kacper Serewiś	k.serewis@student.fontys.nl
Kevin van der Wouw	kevin.vanderwouw@student.fontys.nl

Client: Pharmapartner
Version: 2.0
Version date: 14.09.2020
Status: Concept

Document History

Version	Changes	Author	Date
1.0	Document setup		07/09/2020
2.0	Created and added diagrams with the corresponding description.		14/9/2020
2.1	Class diagram		

Content

DOCUMENT HISTORY 2

C1 PREFACE 4

C2 SYSTEM CONTEXT (C1) 5

C3 DEPLOYMENT DIAGRAM 6

C4 COMPONENTS ERROR! BOOKMARK NOT DEFINED.

C5 CLASS DIAGRAMS AND SEQUENCE DIAGRAMS 8

C6 PERSISTENCY PER COMPONENT 10

C7 SPECIFICATION OF INTERFACES 12

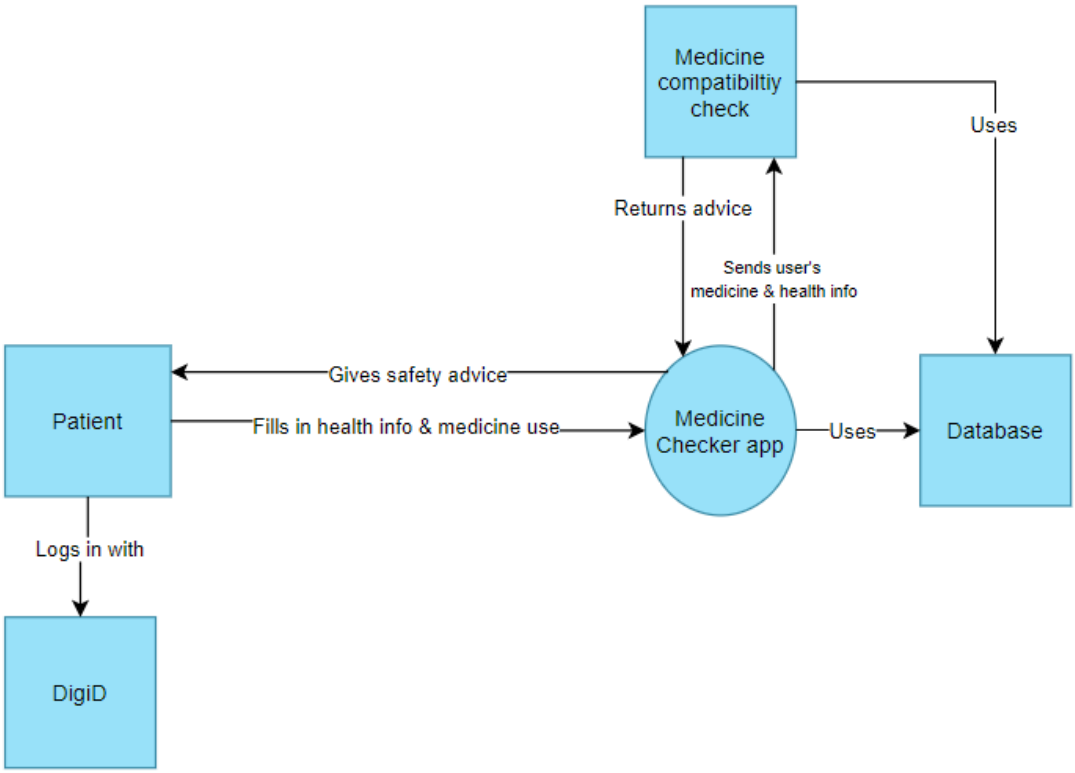
C1 Preface

Prescribing the right medication can be a tricky thing. Some medicines do not work well together and might cancel each other out, or worse. The goal of the Pharmapartners application we are building is to avoid such things from happening. In the application the user will be able to check the compatibility of the medication he or she is taking, giving a warning if one or more medicines does not work well together.

In this document we will show and describe the architecture of the application. What is the System context, how are the different components connected and what are the persistency's of each component? This and more we will make clear in this document. For a description of the more non-functional requirements, please see the Analysis document.

C2 System Context (C1)

The following diagram will show am indication of how the system, and its different components, interacts with each other on a basic level.

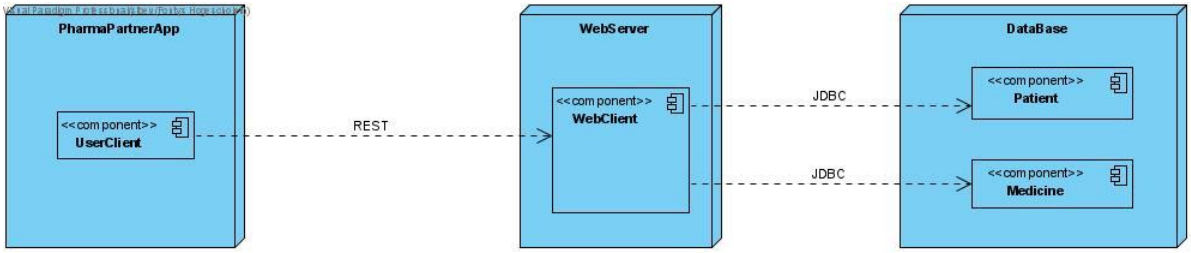


This context diagram shows the following:

- The patient logs in with DigiD.
- The patient fills in their medicine combinations and health information into the medicine checker app.
- The medicine checker app uses the “medicine combination analysis” algorithm on the webserver, which then returns some advice to the app.
- The advice is displayed to the user.
- Both the app and analysis use the database because the patient’s personal information and health information is stored in there.

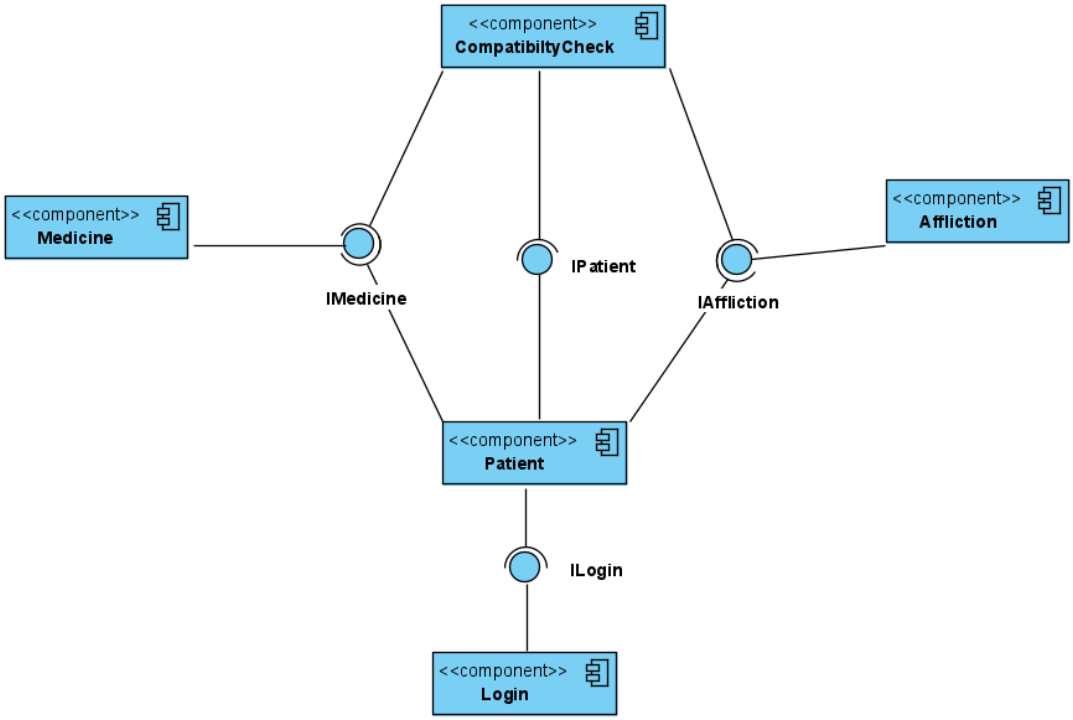
C3 Deployment Diagram

We can take a look at the deployment now we know more about our system context. The basic setup is using three nodes. One as our client (the frontend), one as the backend (the main system) and our database for storage. The frontend and backend are able to communicate using the rest protocol. Because we are developing in Java, we are using the Java DataBase Connectivity driver.



C4 Components

The following diagram gives a basic understanding of the required components in our application. We will go over them one at a time.



Login: The login component is responsible for the authentication of the users.

Patient: The patient component is responsible for the information regarding the user. We cannot compare medicines when we do not know what type of diseases or other medicine usage someone is up to.

Medicine: The medicine component contains information regarding a medicine and its usage.

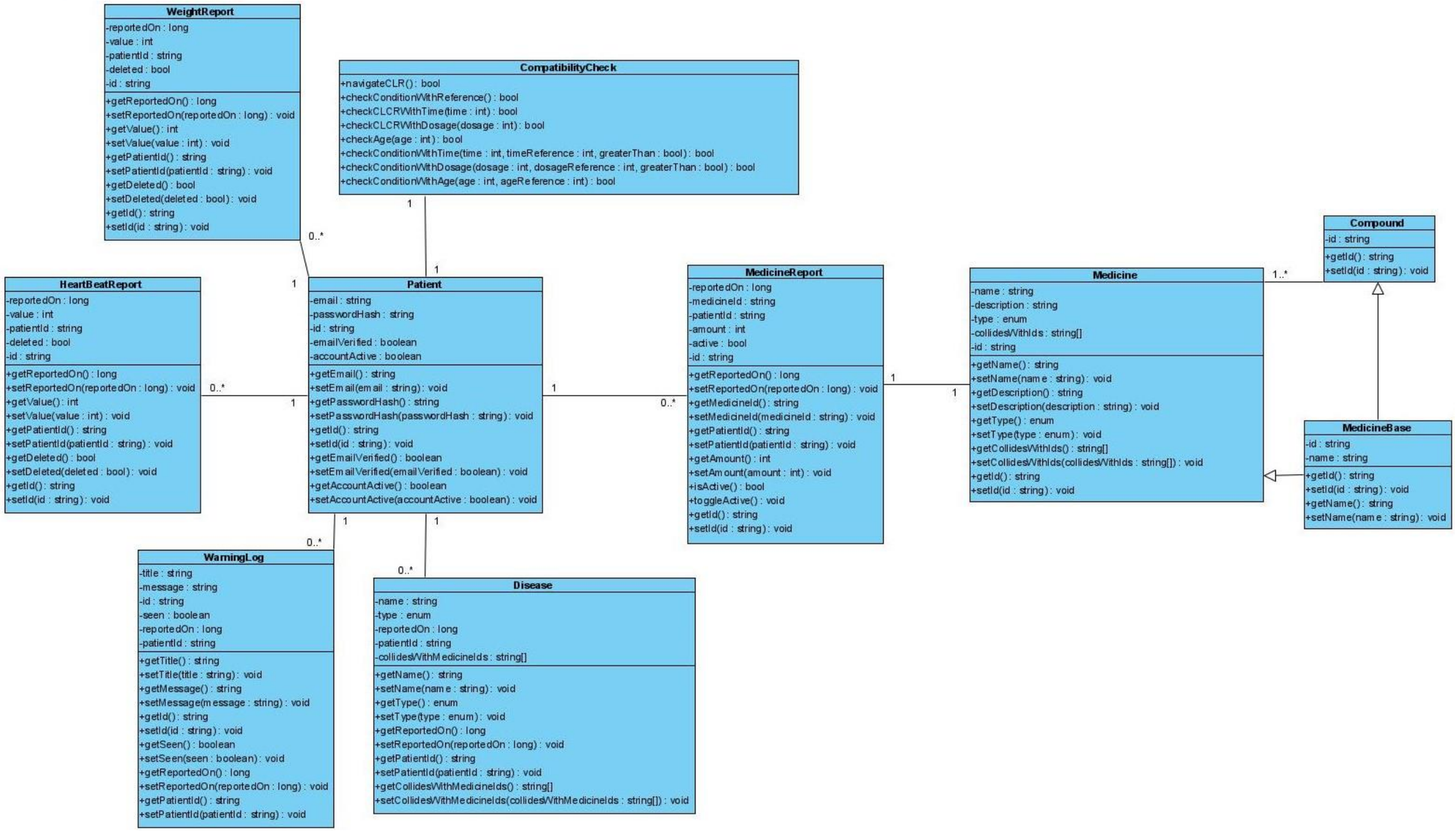
Affliction: The affliction component is for people who are currently ill and need extra checks in order to provide a safe medicine result.

CompatibiltyCheck: This component is the root of the system (looking at our mvp). It is responsible for providing a fast and reliable result to our users regarding their diseases and medicine usage. This is also where most of the business logic will take place.

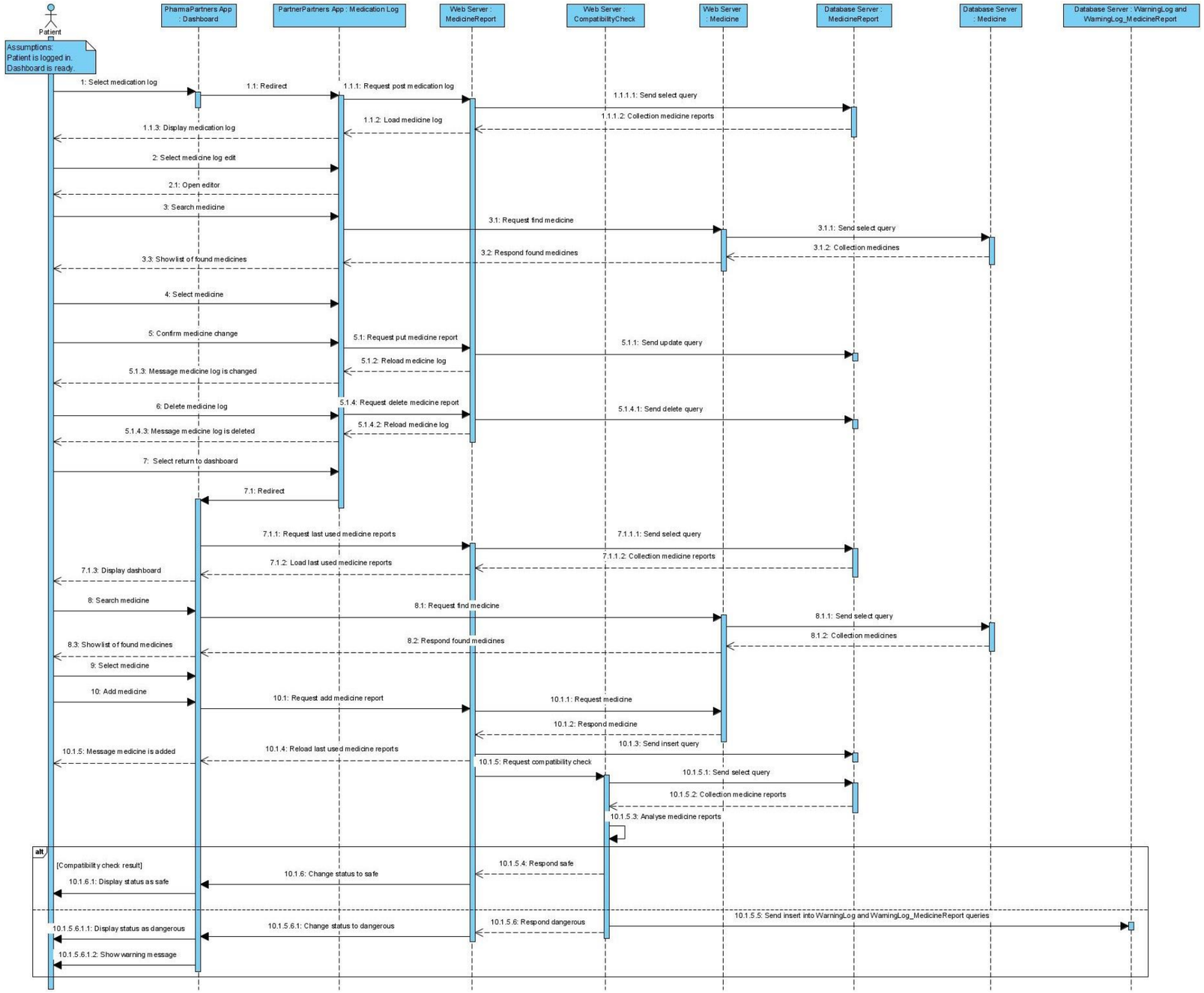
C5 Class Diagrams and sequence diagrams

In this chapter you can see our class diagram and sequence diagrams

Class diagram



Sequence diagram - medicine



Notes of sequence diagram – medicine

Notice: Assume the patient is logged in and the dashboard is ready to use.

1. The patient wants to check the medication log to see what medicines he took in the past. The app will show the medication log, after the patient selects the medicine log in the dashboard.

2. The patient wants to edit the medication report in the medication log. The app opens the text editor, after the patient selects the medication report edit.

3. The patient searches a specific medicine by writing in the text editor. The app will show a list of found medicines the patient can select.

4. The patient selects the specific medicine in the list of found medicines. The app removed the list and only shows one medicine in the text editor.

5. The patient confirms the changed medicine. The app shows the message the medicine is changed. The medicine report has updated with another medicine in the medicine log.

6. The patient removes the medication report from the medication log. In the medication log the patient can remove any medication report.

7. The patient can return to the dashboard by select 'go back'. The app will load the dashboard with a last used medicine reports list.

8. The patient searches a specific medicine by writing in the search engine box. The app will show a list of found medicines the patient can select.

9. The patient selects the specific medicine in the list from the search engine. The app removed the list and only shows one medicine in the box.

10. The patient adds the medicine to the medication log. The app will add the medicine in the last used medicine reports list. The app will also check the added medicine with recent medicine reports. If it is safe, then the status of compatibility check will remain as safe, else the status will display dangerous and show a warning message for bad compatibility between medicines.

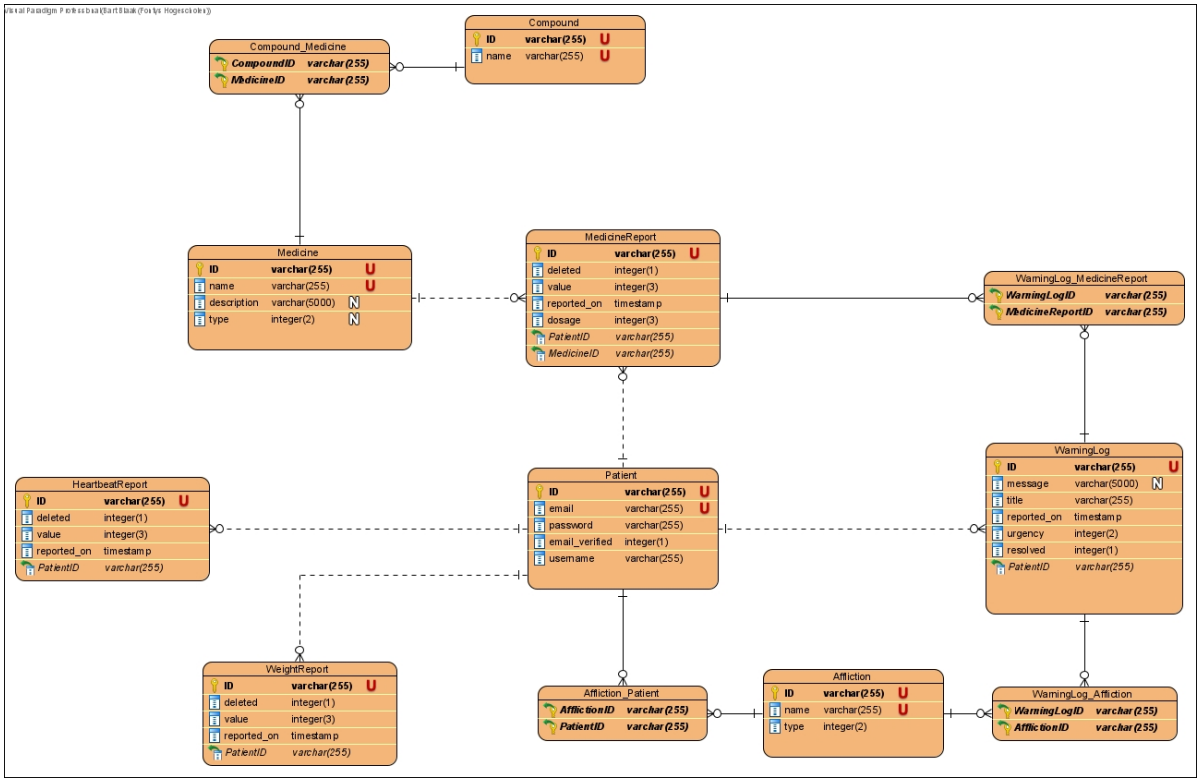
C6 Persistency per component

The following ERD gives you insight into our data layout.
We are going to implement this layout into our program using database.

Some entities do have relation with another ones:
Every patient does have from none to many: “Heartbeat”-, “Medicine”- and “WeightReports”. But also none to many “Afflictions” & “WarningLogs”.

So does “WarningLog” have one “Affliction” and one “MedicineReport”.
And finally MedicineReport does have one Medicine.

All data in this ERD will be saved into the database when it is generated by a component and can be fetched or updated by any other component.



C7 Specification of interfaces

Our specification of interfaces is based on REST and is created using Swagger, all endpoints and data models can be found here:

https://app.swaggerhub.com/apis-docs/K4CZP3R/PharmaPartners/0.0.1