****

**SW-Spec. 2**

SW-Spec. 2 EWS

**Version 1.0**

Historie der Dokumentversionen

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Datum | Autor | Änderungsgrund / Bemerkungen |
| 0.1 | 20.03.18 | Daniel Ladwig | Ersterstellung |
| 1.0 | 24.04.18 | Daniel Ladwig | Neues Artefakt Design, Trennung von Pflichtenheft |
|  |  |  |  |
|  |  |  |  |

Inhaltsverzeichnis

[Historie der Dokumentversionen 2](#_Toc512368935)

[Inhaltsverzeichnis 2](#_Toc512368936)

[Feindesign 3](#_Toc512368937)

[1 Feinplanung der Datenbank 3](#_Toc512368938)

[1.1 Installation 3](#_Toc512368939)

[1.2 Entity Relationship Model (ERM) 3](#_Toc512368940)

[2 Desktopanwendung 5](#_Toc512368941)

[2.1 Idee: 5](#_Toc512368942)

[2.2 Probleme/Gedanken: 5](#_Toc512368943)

[3 Server 6](#_Toc512368944)

[4 App 7](#_Toc512368945)

# Feindesign

# Feinplanung der Datenbank

## Installation

Als Datenbanksystem soll SQLite 3 verwendet werden. Dabei handelt es sich um ein gemeinfreies, relationales Datenbanksystem. Installiert wird diese auf einem RasberryPI. Auf dem RasberryPI läuft eine Serveranwendung, welche auf die Datenbank zugreift.

## Entity Relationship Model (ERM)

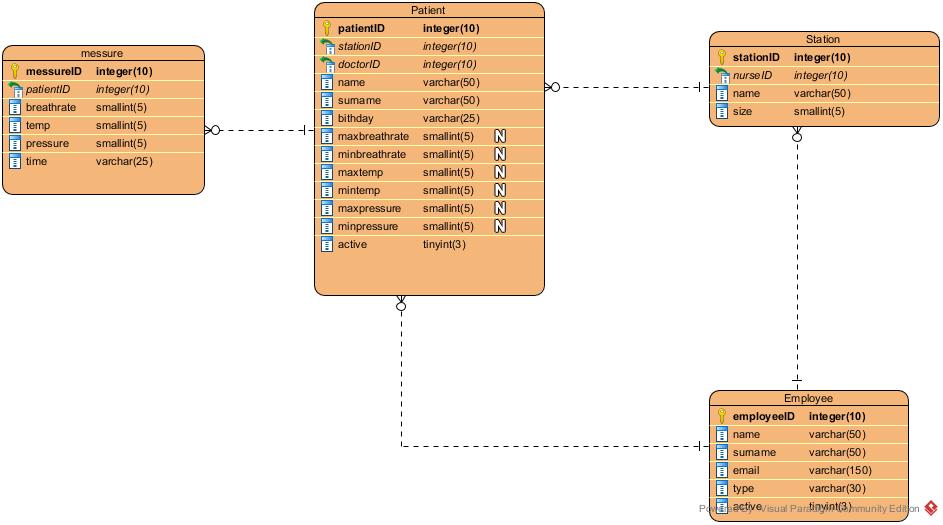


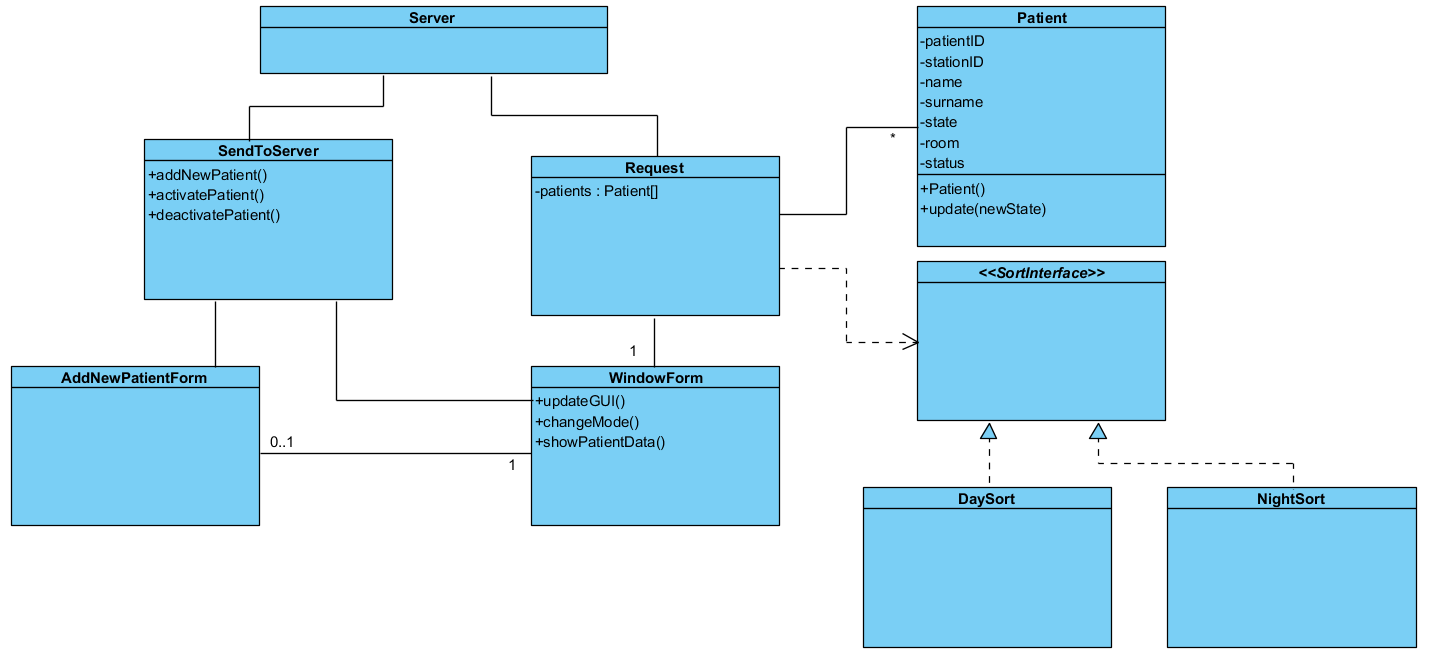
Abbildung ERM der Datenbank

SQLite 3 hat eigene Datentypen, welche nicht im ERM beinhaltet sind. Wird in einem SQL-Skript ein Standarddatentyp verwendet, wird dieser implizit in einen von SQLite 3 zur Verfügung gestellten Datentypen gecastet. Falls der Datentyp genauer durch einen Parameter spezifiziert wird, wird dies von SQLite 3 ignoriert.   
Um in SQL-Skripten eine gewisse Unabhängigkeit zu gewährleisten, werden dennoch die im ERM angegeben Datentypen verwendet.

|  |  |
| --- | --- |
| **Standartdatentyp** | **SQLite-3-Datentyp** |
| INT INTEGER TINYINT SMALLINT MEDIUMINT BIGINT UNSIGNED BIG INT INT2 INT8 | INTEGER |
| CHARACTER(20) VARCHAR(255) VARYING CHARACTER(255) NCHAR(55) NATIVE CHARACTER(70) NVARCHAR(100) TEXT CLOB | TEXT |
| BLOB | BLOB |
| REAL DOUBLE DOUBLE PRECISION FLOAT | REAL |
| NUMERIC DECIMAL(10,5) BOOLEAN DATE DATETIME | NUMERIC |

Eine der Hürden von SQLite 3 ist, dass es keinen eigenen Datentyp zur Verfügung stellt um ein Datum abzuspeichern. Daher sollen sämtlich Zeitpunkte als ISO8601 Strings ("YYYY-MM-DD HH:MM:SS.SSS") abgespeichert werden. SQLite 3 liefert entsprechende Funktionen um den Umgang mit diesem Format einfach zu halten.

# Desktopanwendung



Ergänzungen zum Klassendiagramm der Desktopanwendung:

## Idee:

Die Request-Klasse soll ständig auf eine Anfrage des Servers warten. Kommt eine, werden die übermittelten Daten in eine Patientenliste eingetragen/geupdated und gegebenenfalls sortiert. Dieses Array wird an die WindowForm-Klasse weitergegeben und dort dem entsprechenden Modus an der UI angezeigt. WindowForm kann dabei in die verschiedenen Moden wechseln. Im Patientendaten-Modus, können Patienten aktiviert beziehungsweise deaktiviert werden und neue Patienten hinzugefügt werden. Bei letzterem wird eine neue Form geöffnet – AddNewPatientForm und dort ein Formular mit den entsprechenden einzutragenden Daten angezeigt.

## Probleme/Gedanken:

Im aktuellen Klassendiagramm fehlen noch einige Methoden und Attribute. Diese werden im Laufe der Zeit hinzugefügt und/oder gegebenenfalls entfernt.   
Zudem müssen noch einige Dinge geklärt werden, beispielsweise wie und wo die sortierte Patientenliste gehalten wird (aktuell in der Request-Klasse) oder wie der genaue Ablauf, bei einer State-Änderung zwischen Tag- und Nachtmodus stattfindet.   
Auch muss noch geklärt werden, wo die Sortierung stattfindet und wie die sortierte Patientenliste abgespeichert wird.

# Server

# App