OpenEyes - Clinical Events

Editors: G W Aylward

Version: 0.9:

Date issued: 24 February 2010

Target Audience

General Interest	
Heathcare managers	
Ophthalmologists	✔
Developers	✔

Amendment Record

Issue	Description	Author	Date
0.9	Draft	G W Aylward	22/04/2010

Table of Contents

Introduction

The heart of an electronic patient record is the ability to view previous records of a patient, preferably organised in a logical and helpful manner. Open Eyes has the concept of an ‘event’, which is any distinct clinical entity which can happen to a patient during their care. There are a large number of possible types of event, and they are grouped in ‘episodes’, which corresponds to the NHS definition of a consultant episode. A real world example would be treatment of cataract. Such an episode would include the following events; Referral letter, clinical examination, pre-assessment, operation, post-operative visit and letters to the GP.

Data Structure

There is a multiplicity of possible events, all of which have very different data structures, and there is a need to plan for additional types of event that may be introduced in the future, and are therefore as yet undefined. It is proposed to accommodate these requirements by making use of specialised data structure which is indicated in the entity diagram in figure 1. The essential design feature is that the common fields for any event (data, patient identifier, user identifier) are stored in a table which also includes a reference to a separate table containing the particular data for the event, and a reference scripts to edit and display the event. The field definitions for the episode and events tables are shown in the following tables

Episodes table:

Field	Type	Comments
episode_id	INT UNSIGNED NOT NULL AUTO_INCREMENT	Primary Key, 4 billion
patient_id	INT UNSIGNED NOT NULL	Foreign key referencing patients
specialist_id	SMALLINT UNSIGNED NOT NULL	Foreign key referencing specialists
specialty_id	SMALLINT UNSIGNED NOT NULL	Foreign key referencing specialties
status	ENUM('Open', 'Closed') DEFAULT 'Open'	Status of the episode
startdate	DATETIME NOT NULL	Start date and time
enddate	DATETIME	End date and time (if closed). Presence of an date in this field signifies closure

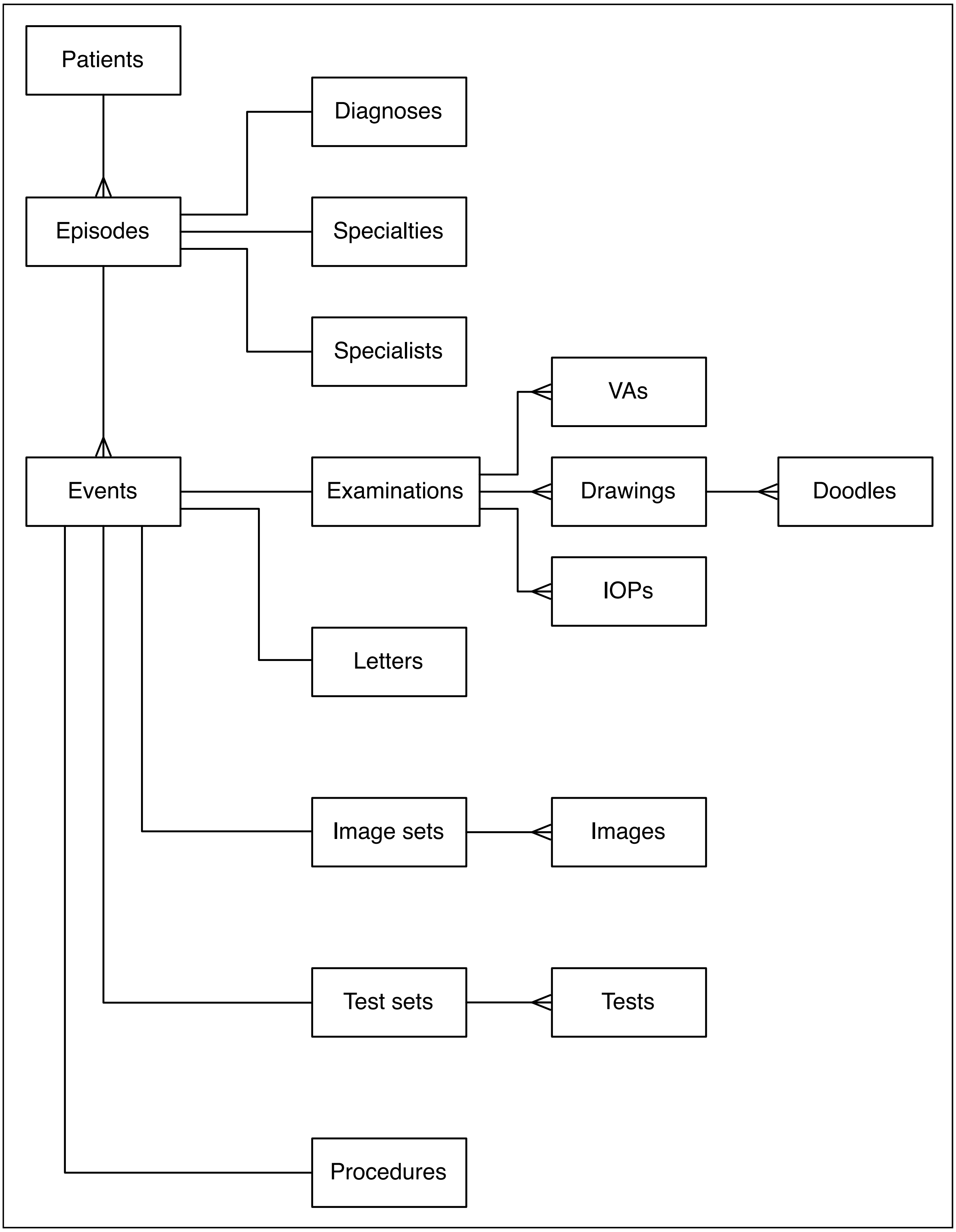
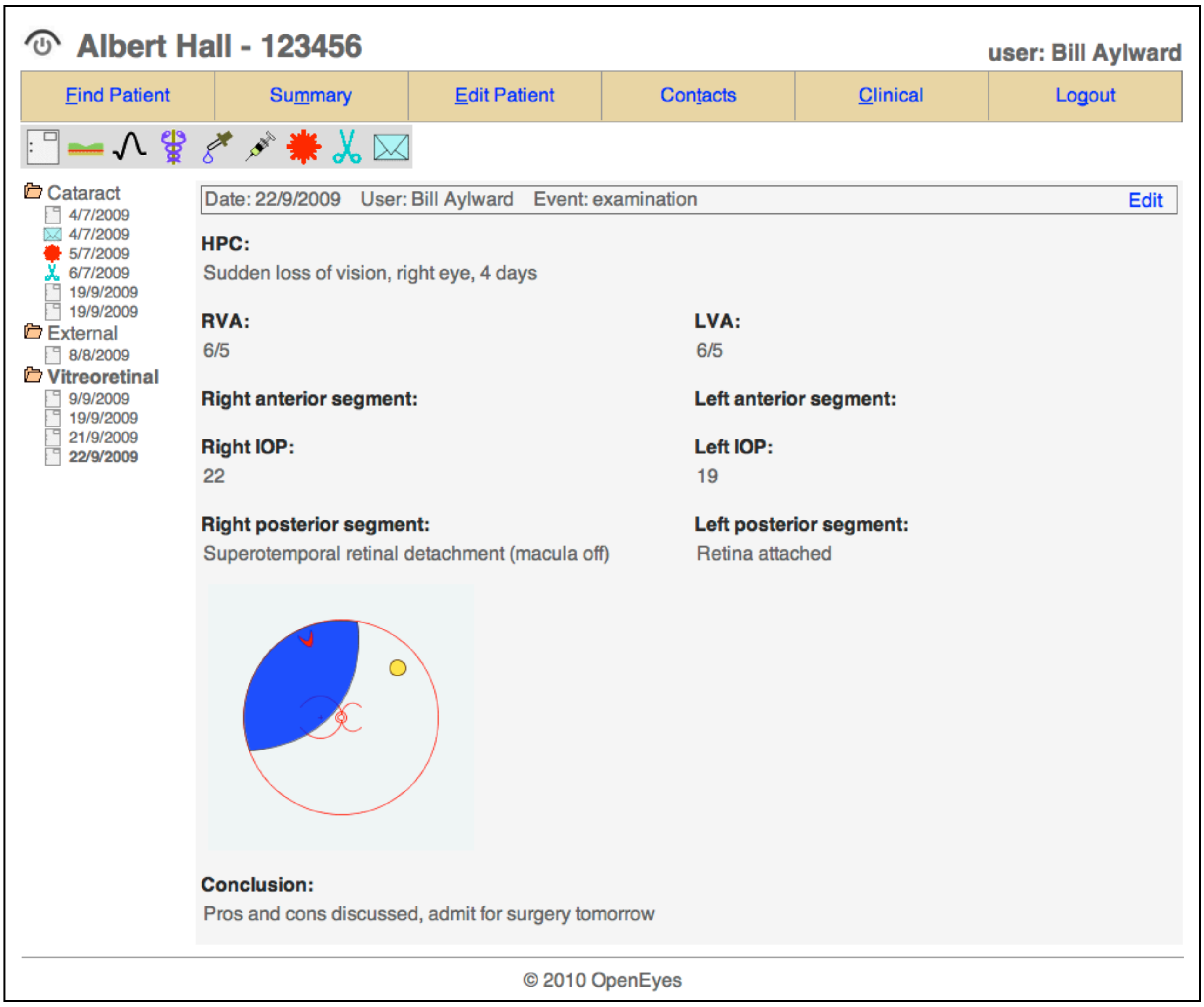


Figure 1. Entity diagram showing the relationship between the events table and other tables holding particular details pertaining to the event.



Albert Hall - 123456

user: Bill Aylward

Date: 22/9/2009 User: Bill Aylward Event: examination

HPC:
Sudden loss of vision, right eye, 4 days

RVA:
6/5

LVA:
6/5

Right anterior segment:

Left anterior segment:

Right IOP:
22

Left IOP:
19

Right posterior segment:
Superotemporal retinal detachment (macula off)

Left posterior segment:
Retina attached

Conclusion:
Pros and cons discussed, admit for surgery tomorrow

© 2010 OpenEyes

Figure 2. Screen shot showing proposed layout for the display of clinical events.The main area to the bottom right of the screen is used to display the event selected in the sidebar. Small icons in the sidebar indicate the type of event.

Events table

Field	Type	Comments
event_id	INT UNSIGNED NOT NULL AUTO_INCREMENT	Primary Key, 4 billion
episode_id	INT UNSIGNED NOT NULL	Foreign key referencing the episode to which this event belongs
user_id	SMALLINT UNSIGNED NOT NULL	Foreign key referencing users indicating the user responsible for this event
datetime	DATETIME NOT NULL	Date and time of the event

Field	Type	Comments
type	ENUM('examination', 'diagnosis', ...') DEFAULT 'examination'	Less storage than using a string, but using an enum means there is a need to modify the table when adding a new type of event

The 'type' field in the events table corresponds to the name of a table and/or a script which will handle the particular data pertaining to that event.

Types of event

A minimum set of events will cover all current clinical interactions with ophthalmology patients. A list of possible events are listed in a table in Appendix 1. Some events will link to other tables where information is more logically stored. For example an ophthalmic examination will consist of certain core items such as visual acuity, and some more specific items which are rarely performed (for example exophthalmometry). It is proposed to store some core items in separate tables to reduce storage requirements, and to aid analysis. For example, analysis of trends in IOP will be easier if the values are stored in a separate table with a patient identifier as an additional foreign key. This approach will allow data item like visual acuity to be measured and stored in a variety of tables, yet still be available for analysis

Displaying and editing events

A proposed user interface for displaying events is shown in the screen shot in figure 2. The display is obviously only available in patient mode. A full list of events is displayed in the sidebar on the left, grouped by episodes which can be clicked open or closed by the user. When an event is selected, the details are displayed in the main area. A toolbar just beneath the header allows the addition of new events. When editing a new event (or an existing one if permitted), an alternative display (based within a form) is used to allow the user to edit fields. The ability to edit a saved clinical examination would be subject to local rules, but one suggestion would be to allow editing for up to three hours after saving to allow contemporaneous changes to be made.

Coding the events screen

The display and coding of the interface illustrated in figure 2 is probably the most complex screen that will the required in OpenEyes. Therefore a modular structure is proposed whereby the code is split into a framework which displays and handles the collection of events, and specific modules for each type of event which 'plug in' to the framework. This arrangement has the advantage of allowing encapsulation of the specific coding related to an event type. In practical terms, the event coding is written as a PHP class specific to one type of event. The class has three major methods called display, edit and save which will carry out the appropriate actions. A flow chart based on this outline is shown in figure 3.

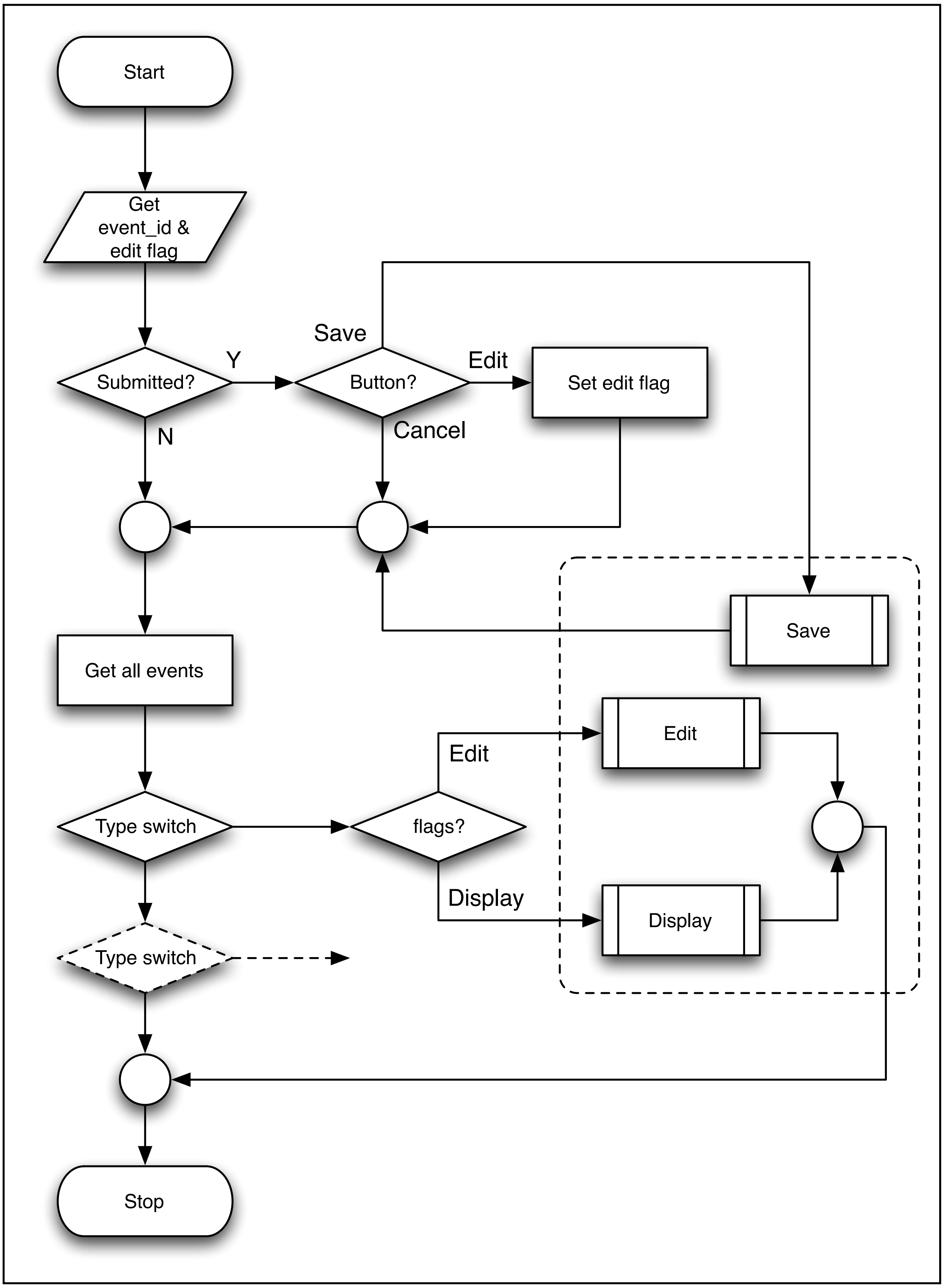


Figure 3. Flow chart for the PHP script to display events. Form submission is detected by a hidden variable. The functions in the area surrounded by the dotted line are supplied by a PHP class specific to the event in question.

Appendix 1. List of events

The following table lists events with a single word type (corresponding to the enum value in the events table)

Type	Description
examination	A clinical examination (The specialty is defined by the enclosing episode)
refraction	A refraction
orthoptics	An orthoptic assessment
diagnosis	A clinical diagnosis
ffa	Fluorescein angiogram
icg	Indocyanine green angiogram
oct	Ocular coherence tomogram
field	A visual field
ultrasound	Ultrasound examination
xray	A plain radiograph
ctscan	Computerised tomography
mriscan	Magnetic resonance imaging
bloodtest	Any blood investigation
prescription	A prescription for drugs
preassess	Pre-operative assessment
anaesth	Anaesthetic assessment
amdapplication	AMD application for funding
amdinjection	AMD injection
injection	Other intravitreal injection
laser	Laser photocoagulaton
letterin	A letter or communication received
letterout	A letter or communication sent
cvi	CVI form or registration

Appendix 2 - SQL statements

The following SQL statements can be used to create the tables discussed in this manual.

Episodes table

```
CREATE TABLE episodes (
episode_id INT UNSIGNED NOT NULL AUTO_INCREMENT,
patient_id INT UNSIGNED NOT NULL,
specialist_id SMALLINT UNSIGNED NOT NULL,
specialty_id SMALLINT UNSIGNED NOT NULL,
status ENUM('Open', 'Closed') DEFAULT 'Open',
startdate DATETIME NOT NULL,
enddate DATETIME,
PRIMARY KEY (episode_id)
)
```

Events table

```
CREATE TABLE events (
event_id INT UNSIGNED NOT NULL AUTO_INCREMENT,
episode_id INT UNSIGNED NOT NULL,
user_id SMALLINT UNSIGNED NOT NULL,
datetime DATETIME NOT NULL,
type ENUM(''examination', 'refraction', 'orthoptics', 'diagnosis', 'ffa', 'icg',
     'oct', 'field', 'ultrasound', 'xray', 'ctscan', 'mriscan', 'bloodtest',
     'prescription', 'preassess', 'anaesth', 'amdapplication', 'amdinjection',
     'injection', 'laser', 'letterin', 'letterout', 'cvi') NOT NULL DEFAULT
     'examination',
PRIMARY KEY (event_id)
)
```