Bayer's Patient Finder wireframe design

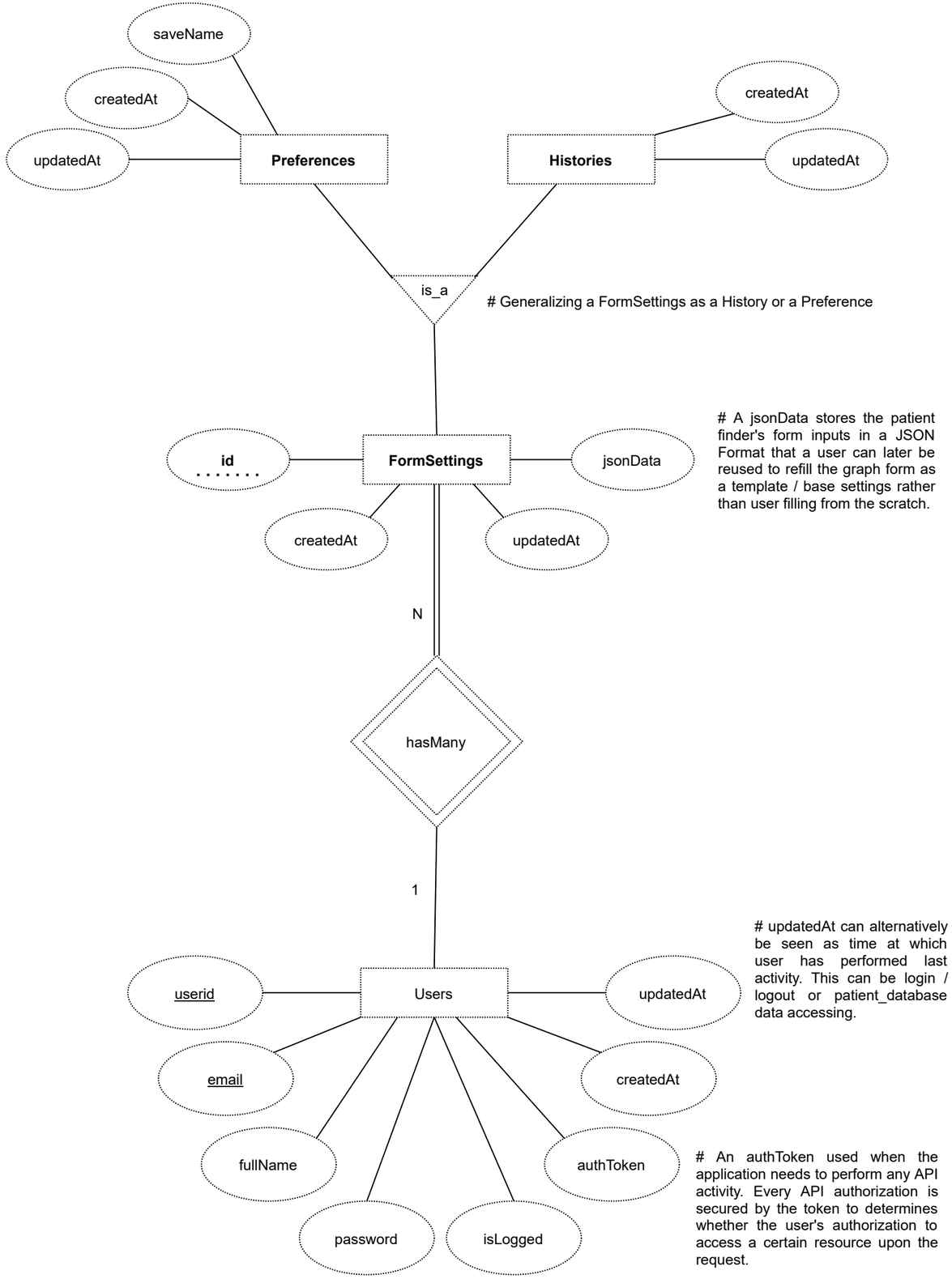
Please note that the scrollbars on fixed to the screen elements are only present to improve screen responsiveness in this design.

Short Description about the ER- diagram database for User preferences:
This product is targeted to build a database, whose design includes following properties:

1. Each user of Patient Finder application can save their Patient Finder form definition saved within the FormSettings entity.
2. Hence, A single user can have many Form Settings, that is stored in a way that is uniquely identifiable. The configuration related to inputs are stored into the database as a raw JSON format named jsonData.
3. A form settings can be generalized into two categories:
    1. Preferences: A FormSettings configurations that are manually saved by a user, so that users can default certain settings like a template rather than filling the long form from scratch. This is a means for users to efficiently work with the Patient Finder application, gaining a bonus interface experience.
    2. History: A FormSetting configuration that are stored automatically when a user clicks for graph generation (or data visualization) so that the potential product owner or respective users of the application can track what they have accessed across their product usage.
4. Note: Difference between a user's Preference and History lies at the part where Preference comes as a feature for assistance in working with the system and History as a feature for tracking their Patient Finder data accesses.

Also, definition of user must include a userid, email, their Full name and a secure password for securely accessing their account. Each time a user gets logged into their account they are generated with a unique accessToken. This user accessToken can be sent to as a authorization mechanism for every action they perform on their account and data accesses. If the access token is mismatch from what's in the database, then the Backend API must not carry out that request as they are marked as *Unauthorized Action* by default.

When a user logs in successfully, they are immediately marked as Logged in (i.e., isLogged becomes true). The time for any last changes made to their account are also saved (as updatedAt). This feature is also consistent for all other entities part of our database.



Database Design for a Patient Finder:

Database: patient_database (say)

Built with the Sequelize ORM Migration support

Users	userid (PK)	email (UK)	fullName	password	defaultPreferenceId	isLogged	authToken	createdAt	updatedAt

FormSettings	userid (CK)	id (CK)	jsonData	createdAt	updatedAt

Preferences	userid (CK)	id (CK)	savedName	createdAt	updatedAt

Histories	userid (CK)	id (CK)	createdAt	updatedAt

Exisiting in the database (say patient_database)
Assumption: Required to pre-exist inside the database of choice

label_info	label	name	label_type	label_val

patients_info	patid	medical_condition	treatment	ip_mi	ip_stroke	ip_hf	ip_cv	dial_dx	px_trans	dial_dx	dial_dx	...
...	dial_cpt_rev	lab_hba1c	lab_egfr	lab_uacr	lab_egfr_uacr	pv_card	pv_endo	pv_gp	pv_neph	pv_urol	...
...	px_dial	sex	race	paytyp	prdtyp	state	eligeff	eligend	ckd	diab	...
...	yob	indexdt	yr	pat_age	pop	agegrp	div1	reg			

May require
optimizations on schema
by Normalizations
(Design Coming soon)

Thoughts:
This optimization can be done by models designed at Backend API using the ORM designing.

For References,

Bayer Patient Finder Backend API Documentation:
https://github.com/RPG-coder/bayer-njit-backend/blob/master/documentation/API%20Documentation%20for%20Bayer%20Patient%20Finder.pdf

Bayer Patient Finder Repository for Backend:
https://github.com/RPG-coder/bayer-njit-backend

Bayer Patient Finder Repository for Frontend:
https://github.com/sp2728/bayer-njit-frontend

More Updates are coming soon.