LOGICAL FRAMEWORK FOR ADMINS AND PARTICIPANTS

# User authentication & access control

When an **admin or master user** opens the application, they must first **log in** using their user ID oremail and password.  
Authentication will verify credentials against the users table and assign a ***role\_id***.

* **Participants (patients, subjects)** do **not** log in (See more in Questions)  
  Instead, they access their tasks using **unique, personalized links** generated via tokens.

# Roles and permissions

There are two user roles in the system:

|  |  |
| --- | --- |
| **Role** | **Permissions** |
| **Master** | Full system control — can create, update, and delete all tables, manage projects, admins, and global settings. Only a few users should have this role. |
| **Admin** | Limited project-based control — can manage only assigned projects, their participants, and related data. Cannot see or modify other projects. |

Each **admin** can manage one or more projects.  
Each **project** can include multiple **subjects (participants)** and multiple **protocols** (sets of tasks with specific order and parameters).

# Admin and master interfaces

After login, users see a **dashboard.**  
The displayed elements depend on the user’s role.

1. **Admin UI**

Upon login, the admin sees:

* Summary statistics for all their projects
* A **project selector** (mandatory before proceeding)
* Available actions:

|  |  |
| --- | --- |
| **Action** | **Description** |
| 1. **Add / Manage Subjects** | View, add, update, or remove subjects from the project. |
| 1. **Create / Assign Protocols** | Design or assign testing protocols to selected subjects. |
| 1. **Data Explorer (optional)** | Explore collected data and progress. |

1. **Master UI**

Master users see everything the Admin sees, plus:

* Manage all **admins and projects**
* System-wide controls (e.g., protocol libraries, configuration, permissions)

# Subject management

Now the confusing part starts, I need to rethink and adjust it.

The **Add / Manage Subjects** section provides a table of subjects within the selected project.

Actions:

1. **Add a new subject to the project**

Opens **Subject personal info** form. After filling the data:

* Saves to the ***subjects*** table.
* Creates a corresponding record in ***project\_subjects***.

The admins can see not only the **Subject personal info** form, but as well at the beginning the information about generating unique link for participants (contains unique token), where participants can fill the personal info by their own. Or admin can fill out all data manually on behalf of the participant.

1. **Update subject personal info**

* Select subject → opens **Subject personal info** form prefilled from database.
* Admin edits and saves → updates ***subjects*** table and refreshes ***updated\_at***.

1. **Assign protocol to existing subject**

Redirects to the **Create / Assign Protocols** page.

1. **End the protocol assignment for subject/subjects** –

* Select protocol → select subject(s) → updates ***participant\_protocols***: sets ***end\_date*** and ***is\_active*** = false.

# b) Protocol creation and assignment

**Create / Assign Protocols** page is spil into two panels:

* Left: available tasks
* Right: current protocol configuration
  + Option to change ***current\_protocol*** name, add questionnaire, change protocol language

Workflow:

1. **Select or create a protocol:**
   1. Existing Protocol:

* Update version → overwrite → save updates to the ***protocols*** table with ***version*** incremented
* Rename and save → creates new protocol -> save to ***protocols*** as new entry
  1. New Protocol:
* Define name, language, tasks, and parameters.

1. **Save protocol** (updates ***protocols*** and ***protocol\_tasks*** tables)
2. **Select subjects** listed for given project (updates ***participant\_protocols*** table)
3. Generate **unique participant links** (based on ***unique\_token***) and optionally send them to each participant.

Additionally, there is button of view current protocol (protocol will be displayed in test mode and data won’t be saved).

# Participant interface

Participants access their test session via the unique link generated for them.

No login is required. (See more in Questions)

The interface loads assigned protocol (tasks in defined order with specified parameters, language, questionnaire, etc.

# Subject personal info form

A **form-type page** allowing subjects to provide demographic or clinical data.  
Values are auto-saved to the database.

# Questions?

* **FREQUENCY DEFINITION:** Which layer should it be defined on: projects, protocols or participants? Where should the user select it, and should this option be available always?
* **TOKEN STRUCTURE:** This is the tricky part – should there be a unique token for each project/protocol and subject? What about the unique link created from that? If not, participants would need to log in and it would be easier for me.
  + Pros: No login required for participants, personalised and secure.
  + Cons: Admins must send separate URLs to each participant.
* **SUBJECT PERSONAL INFO FORM ALONG WITH PROTOCOL:** Should the option be allowed to send the personal info form along with the protocol? Participants would only need one link, but this would complicate the logic, depending on the token structure.

DATABASE DBML

***/////////////////////////////////////////////////////////////***

***// USERS & ACCESS CONTROL***

***/////////////////////////////////////////////////////////////***

***Table users {***

***id integer [pk, increment]***

***email varchar [not null, unique]***

***password\_hash varchar(255) [not null]***

***full\_name varchar***

***role\_id integer [ref: > roles.id, not null]***

***created\_at timestamp [not null, default: `now()`]***

***updated\_at timestamp [default: `now()`]***

***}***

***Table roles {***

***id integer [pk, increment]***

***name varchar [not null, unique, note: 'master, admin']***

***description text***

***}***

***Table user\_projects  {***

***id integer [pk, increment]***

***user\_id integer [ref: > users.id, not null]***

***project\_id integer [ref: > projects.id, not null]***

***assigned\_at timestamp [default: `now()`]***

***indexes {***

***(user\_id, project\_id) [unique]***

***}***

***}***

***/////////////////////////////////////////////////////////////***

***// PROJECT STRUCTURE***

***/////////////////////////////////////////////////////////////***

***Table projects {***

***id integer [pk, increment]***

***name varchar [not null, unique]***

***description text***

***start\_date date [not null]***

***end\_date date***

***is\_active boolean [not null, default: true]***

***frequency varchar [note: 'e.g. daily, weekly, monthly']***

***country varchar***

***contact\_person varchar***

***created\_at timestamp [not null, default: `now()`]***

***created\_by integer [ref: > users.id]***

***updated\_at timestamp***

***updated\_by integer [ref: > users.id]***

***}***

***Table project\_subjects {***

***project\_id integer [not null, ref: > projects.id]***

***subject\_id integer [not null, ref: > subjects.id]***

***enrolled\_at timestamp [default: `now()`]***

***added\_by integer [ref: > users.id]***

***primary key (project\_id, subject\_id)***

***}***

***/////////////////////////////////////////////////////////////***

***// PROTOCOLS & TASK DEFINITIONS***

***/////////////////////////////////////////////////////////////***

***Table protocols {***

***id integer [pk, increment]***

***name varchar [not null]***

***language\_id integer [not null ,ref: > languages.id]***

***description text***

***version integer [not null, default: 1]***

***questionnaires\_id integer [ref: > questionnaires.id]***

***created\_at timestamp [not null, default: `now()`]***

***created\_by integer [ref: > users.id]***

***updated\_at timestamp [default: `now()`]***

***updated\_by integer [ref: > users.id]***

***indexes {***

***(name, version) [unique]***

***}***

***}***

***Table questionnaires {***

***id integer [pk, increment]***

***questions jsonb [not null]***

***version integer [default: 1]***

***updated\_at timestamp [default: `now()`]***

***}***

***Table project\_protocols {***

***project\_id integer [not null, ref: > projects.id]***

***protocol\_id integer [not null, ref: > protocols.id]***

***primary key (project\_id, protocol\_id)***

***}***

***Table protocol\_tasks {***

***id integer [pk, increment]***

***protocol\_id integer [not null, ref: > protocols.id]***

***task\_id integer  [not null, ref: > tasks.id]***

***task\_order integer [not null]***

***params jsonb [note: 'Admin-defined overrides for duration, topic, phoneme, etc. vs each param as new column??']***

***// each param could be as new column, when new task is added, new columns would be created as string with default null value ??***

***indexes {***

***(protocol\_id, task\_order) [unique]***

***}***

***}***

***Table tasks {***

***id integer [pk, increment]***

***key varchar [not null, unique, note: 'e.g. monologue, reading, phonation']***

***type\_id integer [not null, ref: > task\_types.id ,note: 'id of voice, visual, cognitive']***

***recording\_mode jsonb [not null]***

***params jsonb [note: 'JSON schema of editable parameters - names not values']***

***illustration text***

***updated\_at timestamp [default: `now()`]***

***}***

***Table task\_types {***

***id integer [pk, increment]***

***type varchar [unique, note: 'voice, visual, cognitive']***

***updated\_at timestamp [not null, default: `now()`]***

***}***

***Table languages {***

***id integer [pk, increment]***

***code varchar [unique, not null, note: 'e.g. en, cs, de']***

***name varchar [not null]***

***}***

***/////////////////////////////////////////////////////////////***

***// SUBJECTS & PROJECT PARTICIPATION***

***/////////////////////////////////////////////////////////////***

***Table subjects {***

***id integer [pk, increment]***

***external\_id varchar [unique, note: 'Optional external ID if linked to hospital or registry']***

***full\_name varchar [not null]***

***birth\_date date [not null]***

***sex varchar(10)***

***contact\_email varchar***

***contact\_phone varchar***

***notes text***

***created\_at timestamp [not null, default: `now()`]***

***updated\_at timestamp***

***}***

***Table participant\_protocols {***

***id integer [pk, increment]***

***project\_id integer [not null, ref: > projects.id]***

***subject\_id integer [not null]***

***protocol\_id integer [not null]***

***unique\_token varchar [unique, not null, note: 'UUID or hash to reconstruct the URL on the backend']***

***start\_date date [not null]***

***end\_date date***

***is\_active boolean [default: true]***

***}***

***Ref: participant\_protocols.(project\_id, subject\_id) > project\_subjects.(project\_id, subject\_id)***

***Ref: participant\_protocols.(project\_id, protocol\_id) > project\_protocols.(project\_id, protocol\_id)***

***/////////////////////////////////////////////////////////////***

***// SESSIONS (specific protocol runs)***

***/////////////////////////////////////////////////////////////***

***Table sessions {***

***id integer [pk, increment]***

***participant\_protocol\_id integer [ref: > participant\_protocols.id, not null]***

***session\_date timestamp [not null, default: `now()`]***

***progress jsonb [note: 'Stores completed task IDs and timestamps']***

***completed boolean [default: false]***

***// additional notes (medication,....)***

***environment\_notes text***

***}***

***Table recordings {***

***id integer [pk, increment]***

***session\_id integer [ref: > sessions.id, not null]***

***protocol\_task\_id integer [ref: > protocol\_tasks.id, not null]***

***repeat\_index integer [not null, default: 1, note: '1..n repetition count per session/task']***

***recording\_url varchar [not null]***

***duration\_seconds integer***

***created\_at timestamp [default: `now()`]***

***indexes {***

***(session\_id, protocol\_task\_id, repeat\_index) [unique]***

***}***

***}***