# Description of Database Structure:

Users can create an account just inserting their full name, email address and a password to log in.

Account -> user ID to identify two users with same name (unique integer or string autogenerated), full name (String limited to 50 chars), email address (verified through a regex), password (this is both parsed using the framework and I guess it will take care of it)

Users can create a project. Many users can take part to the same project and the single user can work on more projects at once.

The project must have a name, it holds the day of creation, it also contains the list of users working at that project and a list of tags to show what skills are involved and to group them.

Project -> project ID (unique/ autogenerated increasing an int number), project name (String limited in chars controlled using php), project date of creation (DATETIME, since it is the date of creation it is automatically added when inserting in the database), members (members id, the relationship is many to many so a new ‘table’ will be required with the member id and the project id), tags (I will probably add a new table with simply a list of tags as entries)

Relation many to many = many users can work at many projects at the same time.

Tags, the main idea is to have a list of tags (string) users can retrieve their tags for the project from or add in case the tag required is not listed.

Every project will have a list of logs holding the time a user worked at that project. Every log will be linked to a specific project. It must have the date and the time the user started working that day at that project and the number of hours the user did that task. It is possible to summarise the work done.

Log -> log ID (autogenerated index integer), project ID (since every single log refers to a project), user ID (because multiple user can contribute to a project, I want to save who is working at that time), date (another DATATIME but not added automatically when creating because people can log time worked in a different day), time started (to keep a log of the time of the day when the user started working), hours spent (an integer to calculate the total of hours), note ( a simple summary of the log, string limited to 250 chars)

Relation one to many = A project is made of many logs, but every single log is linked only to a specific project; so, no new table will be required since I can query logs by projects

Every log can be linked to some type of content. The user can add details in form of text, images like screenshots, links to the web URLs and files like PDFs or Word documents.

Content -> Log ID (integer), text, links (both strings, it may contain multiple paragraphs so limit to 2500 chars or even more), image, file( images files like pdfs or word documents are uploaded using the framework, there might be a limit to the size to avoid malicious files or big files that can slow down the website)

Relation one to many = A log can keep more files saved but a specific file is part of only of a single time entry. In this case a new table for the relationship is not required.

# URL Scheme:

loggingtool.co.uk/users

POST -> Add a new user with the information added when registering an account

loggingtool.co.uk/users/uniqueUserIdorName

GET -> List the searched user information to show his profile

loggingtool.co.uk/projects/uniqueUserId

GET -> List all the project for the specific user

POST -> Add a new project with the information added from that user as creator (returns project id and info)

loggingtool.co.uk/logs/uniqueProjectId

GET -> list all the information (logs) about a specific project

POST -> Add a new log with the information added in that project

loggingtool.co.uk/content/uniqueLogtId

GET -> list all the information and additional text, files of that specific log (retrieve content)

POST -> Add content, a file, image or text to the log specified

A picture containing monitor, television, control, remote

Description automatically generated

1

many

1

many

many

many