JWT COMPATIBLE

HelixTrack Core

The Core module for the Helix Track.

Development

The HelixTrack Core has been developed and tested on AltBase Linux distribution.

Before you start

Clone the project, then, initialize and update the Git submodules.

Note: Some subprojects (submodules) may be dependant on its own Git submodules. For those, it is required to do the init and update as well.

Using the `clone` script

To do this automatically execute the following:

```
(test -e ./clone || wget https://raw.githubusercontent.com/Helix-Track/Core/main/clone -O clone) && chmod +x ./clone && ./clone
```

Note: It is required to execute the script from empty directory where you whish to clone the HelixTrack project.

Executing inititialisation scripts

Tbd.

Opening the project

From the root of the project execute:

```
./open
```

Note: The `open` command expects that Visual Studio Code is present on the system and available though the `code` command.

Testing the project

From the root of the project execute:

```
./test
```

It will execute all the Testable system components.

Database

The system database

The `Definition.sqlite` represents the system database. It contains all the tables and initial data required for the system to work.

The DDL directory contains all major SQL scripts required to initialize the database.

Convention used for the SQl script is the following:

- The main version scripts:

`Definition.VX.sql` where X represents the version of the database (1, 2, 3, etc).

- Migration scripts:

`Migration.VX.Y.sql` where X represents the version of the database (1, 2, 3, etc) and Y the version of the patch (1, 2, 3, etc).

All SQL scripts are executed by the shell and the `Definition.sqlite` is created as a result.

Scripts and tools

Tbd.

All scripts and tools required for the system to initialize (database, generated code, etc.)

Tbd.

Developers documentation

Documentation can be found here.