P4 Planning Doc - Tessitura Report Catalog

# App Title

Tessitura Report Catalog

# Description

Tessitura Report Catalog is a web application to manage report documentation for all in-house custom reports developed for Tessitura CRM application.

Tessitura (http://tessituranetwork.com) is the core business application we use in our company for online ticket selling, venue management, fundraising, customer relationship management and other ecommerce related operations. To meet our end users’ demand we do lot of inhouse customizations, and created hundreds of custom reports and utilities. The one main challenge our end users and developers face is keeping track of all of these customizations. There are online documentation for all the canned reports and other standard Tessitura functionalities but nothing for the in-house custom solutions. Tessitura Report Catalog web application will meet this demand by creating an online repository for all the in-house reports which then we can hook up with the main Tessitura application so that a user can pull a report documentation on demand using context sensitive help from a Tessitura client application.

This will also help our in-house developers to find the relevant system documentation for troubleshooting and further enhancement purposes.

If there is enough time, the application may also have the features to view all favorite reports by a user; view and save a report detail page as PDF; view and save a searched report listing page as PDF; option to add, edit and view data dictionary/glossary page(s) which will help users & developers to use common lingo; a contact page, and a page to add and update system/reference data without relying on database interface.

# Target Audience

The users and developers of Tessitura CRM application. The initial plan is to implement it in my current organization. If it becomes successful and after a couple of months of usage when everything will become streamlined I may share the application with other Tessitura licensee organizations to be locally installed and used if anyone is interested. Currently there are around 600 Tessitura licensees around the globe.

# Essential Features

* Add a new report
* View list of reports with elaborated search/filtering options
* View a specific report detail
* Edit an existing report
* Add a report revision history by a developer
* Edit/delete a report revision history by a developer
* Add comments to a report; rate and mark a report as favorite by a user
* Edit/delete comments to a report, update rating and mark/unmark a report as favorite by a user

# Non-essential Features

* View “My Favorite” reports by a user
* Add data dictionary/glossary for commonly used terms
* Edit data dictionary/glossary for commonly used terms
* Contact page with send message via email option
* View and save a report list as PDF
* View and save a report detail page as PDF
* Interface to add system or reference data
  + Report Framework (SSRS, InfoMaker, Other)
  + Code Location (SP, Embedded SQL, Other)
  + Report Category
  + User Group/ Roles

# **To do**

[ ] Setup

[ ] Create app

[ ] Version control in Github

[ ] Deploy to production (Digital Ocean) server

[ ] Set up subdomain p4.guddi.ca

[ ] Homepage

[ ] Welcome/intro note

[ ] Logo/Company Name/Application Title

[ ] Link to the “Listing page of reports with elaborated filtering options”

[ ] Top level menu and submenu with link to other pages

[ ] Login

[ ] Register

[ ] Reports

[ ] Page to search a report/solution

[ ] Page to view all reports

[ ] Page to view favorite reports

[ ] Page to comment on a report

[ ] Page to view my (logged in user) comments along with edit, delete options

[ ] Page to rate a report and mark a report as favorite

[ ] Page to view my (logged in user) ratings along with edit, delete options

[ ] Glossaries

[ ] Page to view all glossary terms along with edit, delete options

[ ] Page to create a new report

[ ] Developers

[ ] Page to view all reports along with edit, delete options

[ ] Page to create a new report

[ ] Page to add revision history to a report

[ ] Page to view my (logged in user) revisions history along with edit, delete options

[ ] Finalize

[ ] Test all features

[ ] Observe someone else use the application

[ ] Validate, spell check

[ ] Update Github reports

[ ] Final deployment on live site

# Route Plan (Essential Features)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Purpose** | **Method** | **Route Name** | **Action** | **URI** |
| Homepage | GET | page.home | index | / |
| Contact | GET | page.contact | create | /contact |
| Contact | POST | page.contactPost | send | /contact |
| Listing of reports | GET | reports.index | index | /reports/ |
| Show a report detail | GET | reports.show | show | /reports/{reportId?} |
| Display favorite reports | GET | reports.my\_fave\_reports | index | /my-fav-reports |
| Search a report | GET | reports.search | create | /search |
| Display search result | GET | reports.search\_result | index | /search-result |
| List reports for developers | GET | reports-dev.index | index | /reports-dev |
| Show form to add a new report | GET | reports-dev.create | create | /reports-dev/create |
| Process form to add the report | POST | reports-dev.store | store | /reports-dev |
| Show a report with technical information for a developers | GET | reports-dev.show | show | /reports-dev/{id} |
| Show form to edit a report | GET | reports-dev.edit | edit | /reports-dev/{id?}/edit |
| Process form to edit the report | PUT | reports.update | update | /reports-dev/{id?}/edit |
| Show form to delete a report | GET |  | show | /reports-dev/{id}/delete |
| Process form to delete a report | DELETE |  | delete | /reports-dev/{id} |
| Show all my (logged in user) comments | GET | comments.my\_comments | index | /my-comments |
| Show form to create a comment | GET | comments.create | create | /comments/create |
| Process form to create a comment | POST | comments.store | store | /comments |
| Show form to display a comment | GET | comments.show | show | /comments/{id} |
| Show form to edit a comment | GET | comments.edit | edit | /comments/{id}/edit |
| Process form to edit the comment | PUT | comments.update | update | /comments/{id} |
| Show form to delete a comment | GET | comments.delete | delete | /comments/{id}/delete |
| Process form to delete a comment | DELETE | comments.destroy | destroy | /comments/{id} |
| Show all my (logged in user) revisions | GET | revisions.my\_revisions | index | /my-revisions |
| Show form to create a revision | GET | revisions.create | create | /revisions/create |
| Process form to create the revision | POST | revisions.store | store | /revisions |
| Show an individual revision | GET | revisions.show | show | /revisions/{id} |
| Show form to edit a revision history | GET | revisions.edit | edit | /revisions/{id?}/edit |
| Process form to edit the revision history | PUT | revisions.update | update | /revisions/{id}/edit |
| Show form to add a revision history | GET | revisions.delete | delete | /revisions/{id}/delete |
| Process form to delete a revision history | DELETE | revisions.destroy | destroy | /revisions/{id} |
| Show all my (logged in user) ratings | GET | ratings.my\_ratings | index | /my-ratings |
| Show form to create a rating | GET | ratings.create | create | /ratings/create |
| Process form to create a rating | POST | ratings.store | store | /ratings |
| Show form to display a rating | GET | ratings.show | show | /ratings/{id} |
| Show form to edit a rating | GET | ratings.edit | edit | /ratings/{id}/edit |
| Process form to edit the rating | PUT | ratings.update | update | /ratings/{id} |
| Show form to delete a rating | GET | ratings.delete | delete | /ratings/{id}/delete |
| Process form to delete a rating | DELETE | ratings.destroy | destroy | /ratings/{id} |
| Show all glossary terms | GET | glossaries.index | index | /glossaries |
| Show form to create a glossary | GET | glossaries.create | create | /glossaries/create |
| Process form to create the glossary | POST | glossaries.store | store | /glossaries |
| Show an individual glossary term | GET | glossaries.show | show | /glossaries/{id} |
| Show form to edit a glossary | GET | glossaries.edit | edit | /glossaries/{id?}/edit |
| Process form to edit the glossary | PUT | glossaries.update | update | /glossaries/{id}/edit |
| Show form to delete a glossary term | GET | glossaries.delete | delete | /glossaries/{id}/delete |
| Process form to delete a glossary term | DELETE | glossaries.destroy | destroy | /glossaries/{id} |

*(Attention* ***Kate****: I broke the database object naming conventions suggested in the class here. Had a discussion in Piazza with* ***Susan*** *on this. Here is the Piazza discussion link:* [*https://piazza.com/class/iqiwxxw3sex3r2?cid=155*](https://piazza.com/class/iqiwxxw3sex3r2?cid=155)*. Thanks!*

***Update (Nov 29, 2016):*** *I thought about it. Since it will be a completely separate DB from our business app, I changed my plan to stick to Laravel convention instead - at least for the duration of this course, so that I can take advantage of some Laravel built-in functionalities.)*

# Database Tables

**Table name:** reports

**Description:** Will store individual report doc

**Fields:**

* (int, primary key, auto\_increment) **id**
* (varchar(255)) **name**
* (text) **description**
* (varchar(255)) **tess\_report\_id**
* (varchar(255)) **definition file**
* (text) **sql\_proc**
* (varchar(255)) **database**
* (varchar(255)) **keywords**
* (text) **note\_general**
* (text) **note\_technical**
* (int) **category\_id**
* (int) **framework\_id**
* (int) **type\_id**
* (boolean) **schedulable**
* (boolean) **inhouse**
* (boolean) **verified**
* (boolean) **published**
* (datetime) **first\_implemnetation\_dt**
* (datetime) **last\_update\_dt**
* (boolean) **discontinued**
* (boolean) **active**
* (timestamp) **created\_at**
* (timestamp) **updated\_at**
* (int) **created\_by**
* (int) **updated\_by**

**Table name:** screenshots

**Description:** Will store screenshot file names for a report

**Fields:**

* (int, primary key, auto\_increment) **id**
* (int) **report\_id**
* (varchar(255)) **file\_name**
* (varchar(255)) **file\_type**
* (varchar(255)) **caption**
* (varchar(255)) **description**
* (boolean) **active**
* (timestamp) **created\_at**
* (timestamp) **updated\_at**

**Table name:** categories

**Description:** Will store report category names

**Fields:**

* (int, primary key, auto\_increment) **id**
* (varchar(255)) **name**
* (varchar(255)) **description**
* (boolean) **default**
* (boolean) **active**
* (timestamp) **created\_at**
* (timestamp) **updated\_at**

**Table name:** frameworks

**Description:** Will store report frameworks

**Fields:**

* (int, primary key, auto\_increment) **id**
* (varchar(255)) **name**
* (varchar(255)) **description**
* (boolean) **default**
* (boolean) **active**
* (timestamp) **created\_at**
* (timestamp) **updated\_at**

**Table name:** types

**Description:** Will store report types

**Fields:**

* (int, primary key, auto\_increment) **id**
* (varchar(255)) **name**
* (varchar(255)) **description**
* (boolean) **default**
* (boolean) **active**
* (timestamp) **created\_at**
* (timestamp) **updated\_at**

**Table name:** tessareas

**Description:** Will store Tessitura area for reports

**Fields:**

* (int, primary key, auto\_increment) **id**
* (varchar(255)) **name**
* (text) **description**
* (boolean) **default**
* (boolean) **active**
* (timestamp) **created\_at**
* (timestamp) **updated\_at**

**Table name:** report\_tessarea

**Description:** Pivot table between reports and tessareas

**Fields:**

* (int, primary key, auto\_increment) **id**
* (int) **report\_id**
* (int) **tessarea\_id**
* (boolean) **active**
* (timestamp) **created\_at**
* (timestamp) **updated\_at**

**Table name:** comments

**Description:** Table to store a user comments.

**Fields:**

* (int, primary key, auto\_increment) **id**
* (int) **report\_id**
* (int) **user\_id**
* (varchar(255)) **description**
* (datetime) **comment\_dt**
* (boolean) **active**
* (timestamp) **created\_at**
* (timestamp) **updated\_at**
* (timestamp) **deleted\_at**

**Table name:** ratings

**Description:** Table to store a user comments.

**Fields:**

* (int, primary key, auto\_increment) **id**
* (int) **report\_id**
* (int) **user\_id**
* (int) **rating**
* (boolean) **favorite**
* (boolean) **active**
* (timestamp) **created\_at**
* (timestamp) **updated\_at**
* (timestamp) **deleted\_at**

**Table name:** revisions

**Description:** Table to store a revision history.

**Fields:**

* (int, primary key, auto\_increment) **id**
* (int) **report\_id**
* (int) **user\_id**
* (varchar(255)) **description**
* (timestamp) **revision\_dt**
* (boolean) **active**
* (timestamp) **created\_at**
* (timestamp) **updated\_at**
* (timestamp) **deleted\_at**

**Table name:** glossaries

**Description:** Will store data dictionary/glossary items.

**Fields:**

* (int, primary key, auto\_increment) **id**
* (varchar(255)) **term**
* (text) **definition**
* (boolean) **active**
* (timestamp) **created\_at**
* (timestamp) **updated\_at**
* (int) **created\_by**
* (int) **updated\_by**

**Table name:** users (Laravel built-in)

**Description:** Table to store users

**Fields:**

* (int, primary key, auto\_increment) **id**
* (varchar(255)) **name**
* (varchar(255)) **email**
* (varchar(255)) **password**
* (varchar(100)) **remember\_token**
* (timestamp) **created\_at**
* (timestamp) **updated\_at**

**Table name:** password\_resets (Laravel built-in)

**Description:** Table to store users

**Fields:**

* (varchar(255)) **email**
* (varchar(255)) **token**
* (timestamp) **created\_at**

**Table name:** roles

**Description:** Table to store application roles.

**Fields:**

* (int, primary key, auto\_increment) **id**
* (varchar(255)) **name**
* (varchar(255)) **description**
* (boolean) **default**
* (boolean) **active**
* (timestamp) **created\_at**
* (timestamp) **updated\_at**

**Table name:** role\_user

**Description:** Pivot table between the roles table and the users table.

**Fields:**

* (int, primary key, auto\_increment) **id**
* (int) **role\_id**
* (int) **user\_id**
* (boolean) **active**
* (timestamp) **created\_at**
* (timestamp) **updated\_at**

**Table name:** appobjects

**Description:** Will store object details for this web application

**Fields:**

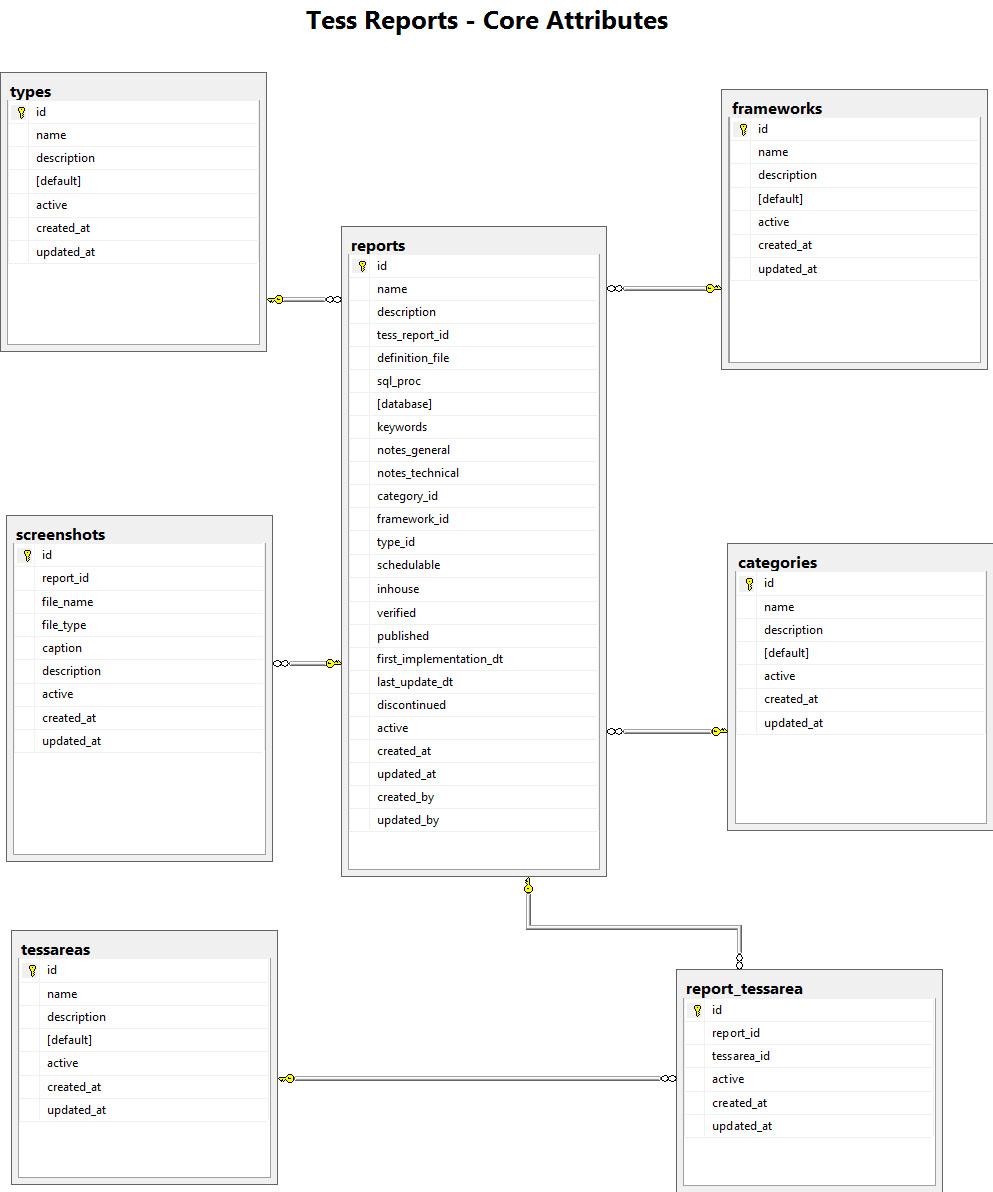
* (int, primary key, auto\_increment) **id**
* (varchar(255)) **name**
* (text) **description**
* (boolean) **default**
* (boolean) **active**
* (timestamp) **created\_at**
* (timestamp) **updated\_at**

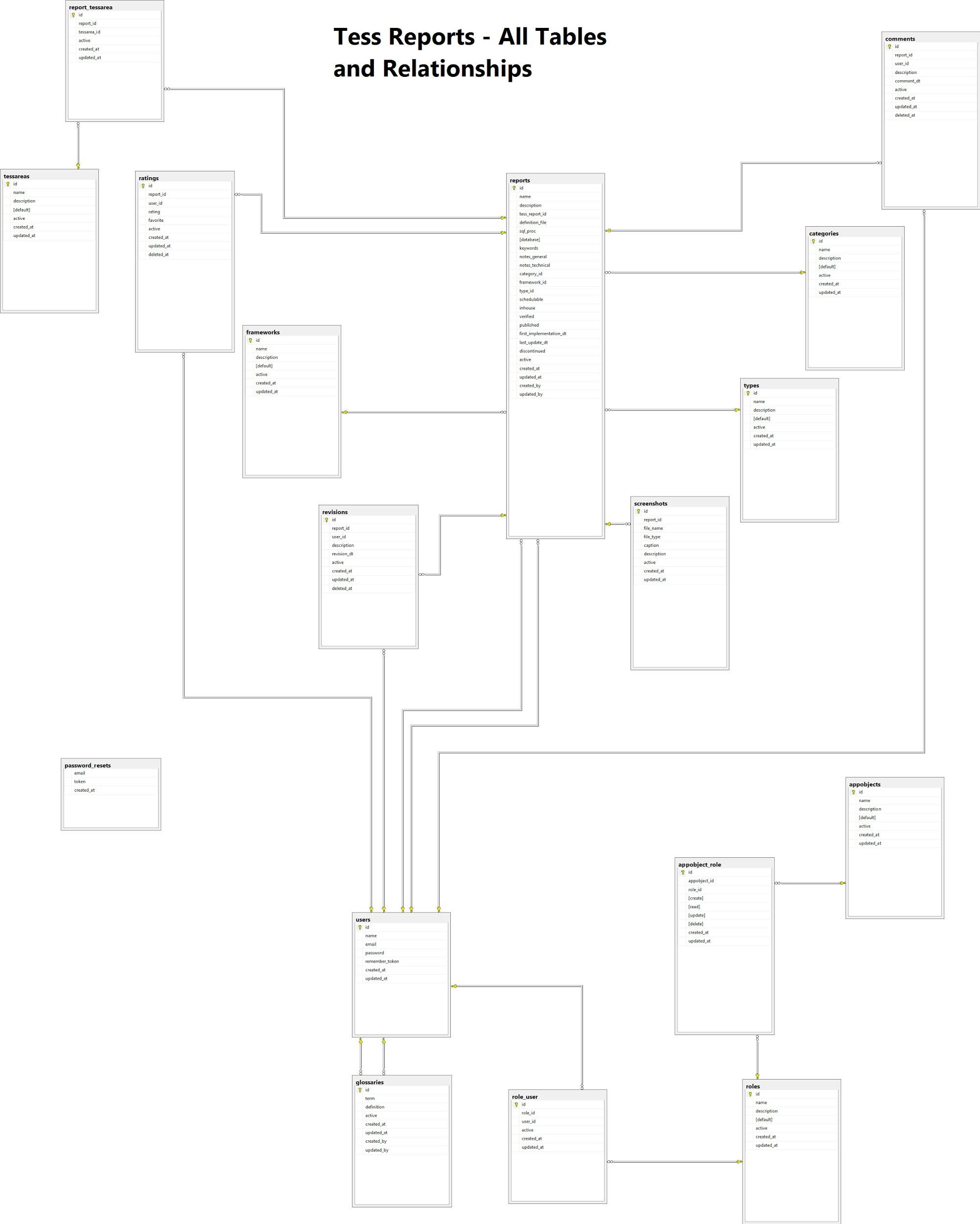
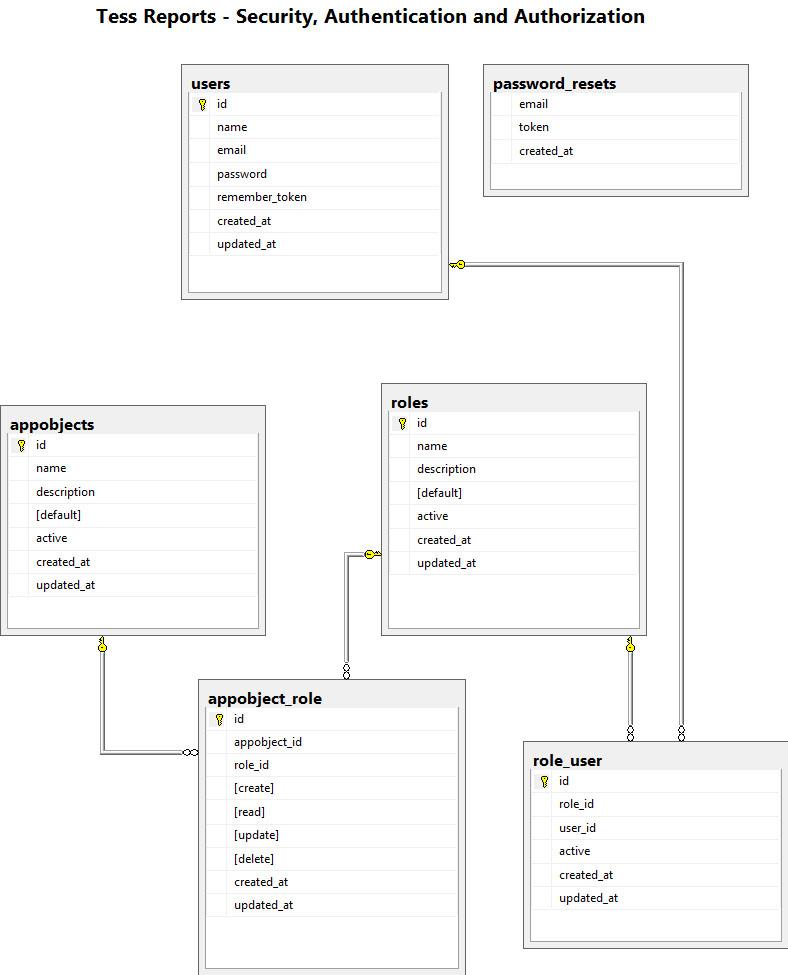
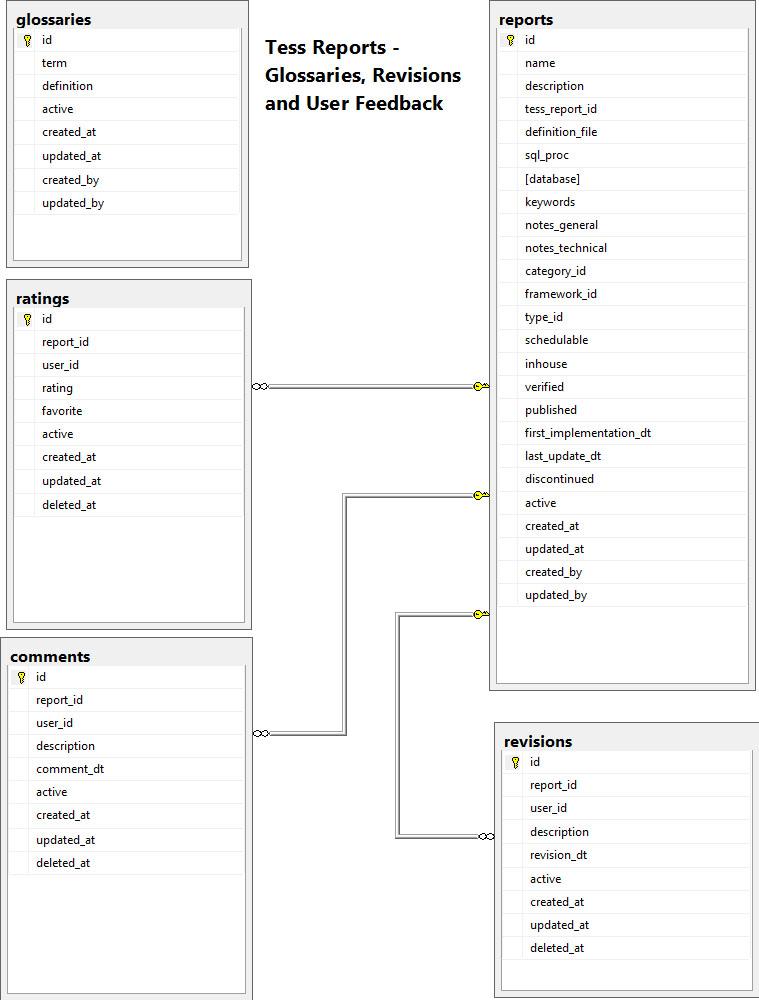
**Table name:** appobject\_role

**Description:** Pivot table between appobjects and roles for authorization purpose

**Fields:**

* (int, primary key, auto\_increment) **id**
* (int) **appobject\_id**
* (int) **role\_id**
* (boolean) **create**
* (boolean) **read**
* (boolean) **update**
* (boolean) **delete**
* (timestamp) **created\_at**
* (timestamp) **updated\_at**





Misc

1. General viewing of a report will not be password protected to accommodate linking from the core business application (yet to finalize this decision). All other functionalities will be password protected.
2. All system/reference data are quite static and hardly will require an up-date. Even if they ever get updated the frequency will be very low - may be once in a couple of years. This update will be done by the developers. So no front end entry/update screen is necessary at this point. That’s why I added them as non-essential features.
3. Though for the class project purpose I will keep a signup screen for the users eventually there will be only login screen for the users but no sign-up screen as the application won’t remain open to public. Only designated users will have access to this application. The users will be added by the admin staff.
4. Screenshot file names in the screenshots table will be saved with file extension but without any folder location. All screenshots will be pulled from a particular folder of the application.
5. Keywords to easily find a report will be saved as a comma separated value string in the reports table for simplicity purpose instead of putting them in a separate table.
6. Since Laravel built functionalities don’t support role based security, I extended this part to add my own role based security. Look at the “Tess Reports - Security, Authentication and Authorization” diagram above for details.