High level design:

FormStorage has been divided into 2 subcomponents:

* forms
* errors

All store and retrieve related functionality is present in **forms** package

Error handling functionality is present in **errors** package

Each have their own Blueprints registered with the top-level app

**Database used is SQLITE and SQLAlchemy is used for interaction with the database.**

**forms:**

**routes.py:**

This is the core module defining all the APIs. This defines the following APIs:

*/store – This has the GET and POST end points. It first renders the html containing the form elements for the user to enter the data into the form fields. On submit it stores the entered data in the database and calls the storedid API with the id of the stored form data.*

*/storedid/<id> - This API takes an id parameter and searches if a form with that id is present in the database. If it is present it displays the id with the form contents in json format else returns a 404*

*/retrieve – This API is used to retrieve the data from the database. This lets user to query database using either or both of the value and localisable columns. If user wants to view all form fields, he may leave both the text boxes blank. This renders a html with text boxes for value and localisable fields. On submit, this calls the display API described below*

*/display/<string:value>/<string:localisable> - This takes value and localisable arguments. These can be left blank if all the form fields need to be retrieved without any filter.*

**forms.py:**

This module contains the classes required for the forms for the data to be entered by the user for store and retrieve.

**errors:**

This component handles the exceptions that might occur. Any new error to be handled can be added to this module

e.g.,

404 – which handles the exception occurred when data we try to retrieve is not present

**Config.py:**

This is outside of the components described above. This module contains configuration details which is passed during app creation.

This contains secret key as it is required for WTF package and also connection details to database(Sqlite). We can plug in a different database if needed here.

**Models.py**

This module contains the classes corresponding to the tables that needs to be created. Class attributes correspond to table columns.

This project requires two tables. Form and Formfield .

Form table has a backreference to Formfield table for faster accessibility when a form with a particular id needs to be retrieved.

**Templates:**

This folder contains all the html files needed for the project if it is run from a browser.