**Introduction**

Welcome to the e-consent/frontend module! This module is specifically designed to cater to the frontend aspect of the application. It consists of two submodules, namely e2e and src. The e2e submodule focuses on initiating the entire frontend process, while the src submodule encompasses all the code related to the frontend of the econsent app.

In this README, we will primarily focus on the src submodule, as it plays a crucial role in the frontend development of the econsent app. We will explore the different sections within the src module, where each section has its own set of tasks to accomplish.

1. Getting Started
2. Modules
   1. Consents
   2. Login
   3. Patients
   4. researchStudy
3. Assets
4. Environment

Let's begin by getting familiar with the src module and its various sections.

**Getting Started**

To get started, let's explore the different sections within the src module:

1. **App**: This section encompasses all the HTML, CSS, and JavaScript files. It houses the Angular modules, configurations, services, and common webpage files.

2. **Assets**: Here, you'll find static documents and files like supporting images and static values that are used throughout the application.

3. **Environments**: This section allows you to specify the deployment environment for the application.

4. **Browser Lists**: It provides information about the supported browsers for the application.

5. **Main.ts**: This file serves as the trigger for the entire application, initiating its execution.

By understanding these sections within the src module, you'll gain insights into the structure and components of the frontend of the econsent app.

**Config:**

The "config" section encompasses fundamental and commonly used values that are utilized across the entire application. It serves as a repository for essential configurations and settings that are shared among different components or modules within the application.

**all-constant.config.ts**

The provided code includes several TypeScript enums, which are used to define named values representing specific types or categories. Let's explore each enum and its respective values:

1. `**ConsentListConstant**` enum:

- `MAXIMUM\_CONSENT\_LENGTH`: Denotes the maximum length of a consent, set to 16.

- `MAX\_ROWS\_PER\_PAGE`: Indicates the maximum number of rows per page, set to 10.

- `MAX\_NUM\_ITEM\_SHOW\_AT\_ONCE`: Represents the maximum number of items to display at once, set to 10.

- `NO\_ITEM\_SHOW\_MAX\_REACHED`: Signifies the number of items to show when the maximum limit is reached, set to 10.

2. `**StatusCodes**` enum:

- `CODE\_200`, `CODE\_201`, `CODE\_400`, `CODE\_401`, `CODE\_404`, `CODE\_409`, `CODE\_500`, `CODE\_502`, `CODE\_503`: Represents commonly used HTTP status codes in web development, each associated with its respective numeric value.

3. `**LoginRoles**` enum:

- `TECHNICIAN`, `ADMIN`, `RESEARCHER`, `PATIENT`: Represents different roles for users within a login system, with each role represented by a string value.

4. `**DropdownConstants**` enum:

- `MAX\_NUM\_ITEM\_SHOW\_AT\_ONCE`: Indicates the maximum number of items to display in a dropdown at one time, set to 3.

- `NO\_ITEM\_SHOW\_MAX\_REACHED`: Denotes the number of items to show when the maximum limit is reached, set to 10.

5. `**GenericColors**` enum:

- Represents a collection of predefined colors used in the application, with each color represented by a hexadecimal code.

These enums offer a centralized approach to define and refer to constant values across the codebase, simplifying the maintenance and modification of these values in a single location.

**common.config.ts**

The provided code consists of two TypeScript enums: ShowErrorMessage and ShowInfoMessage. These enums are used to define different messages for displaying error and informational notifications in an application.

The ShowErrorMessage enum includes messages for various error scenarios. Each error message is associated with a specific key or code, such as UNAUTHORISED\_ERROR\_401 representing an unauthorized login error, CONFLICT\_ERROR\_409 indicating a conflict during an action, and BAD\_REQUEST\_ERROR\_400\_500\_502 denoting a general error with a request.

The ShowInfoMessage enum contains messages for displaying informational notifications. These messages cover a range of scenarios, such as the creation, update, and deletion of consent forms and research studies, success and failure messages for different operations, patient-related messages, and more.

By using these enums, developers can easily reference and display the appropriate error or information message based on specific application scenarios. The messages are designed to provide clear and concise feedback to users about the outcome of their actions or the state of the application.

The "Modules" section comprises four components:

1. Consents: Responsible for managing all aspects related to consent forms, including creation, update, and deletion.

2. Login: Manages the login functionality for various types of users accessing the application.

3. Patients: Handles the display and management of patient-related information and features.

4. Research Study: Deals with the creation of research studies and the addition of consent forms associated with them.

In summary, the "Modules" section encompasses these four components, each focusing on specific functionalities and tasks within the application.

let us take a look on each module

**Consents Component:**

In the "Consents" component, there are several features and functionalities:

**Consent Lists:** This feature manages and handles all the consent forms that have been created. It provides a way to view and organize the existing consent forms. (selector name: app-consent-list)

A screenshot of a computer

Description automatically generated

**Create/Update Consent:** This functionality is responsible for creating new consent forms or updating existing ones. It allows users to input the necessary information and configure the details of the consent form. (selector name: app-create-update-consent)

A screenshot of a computer

Description automatically generated

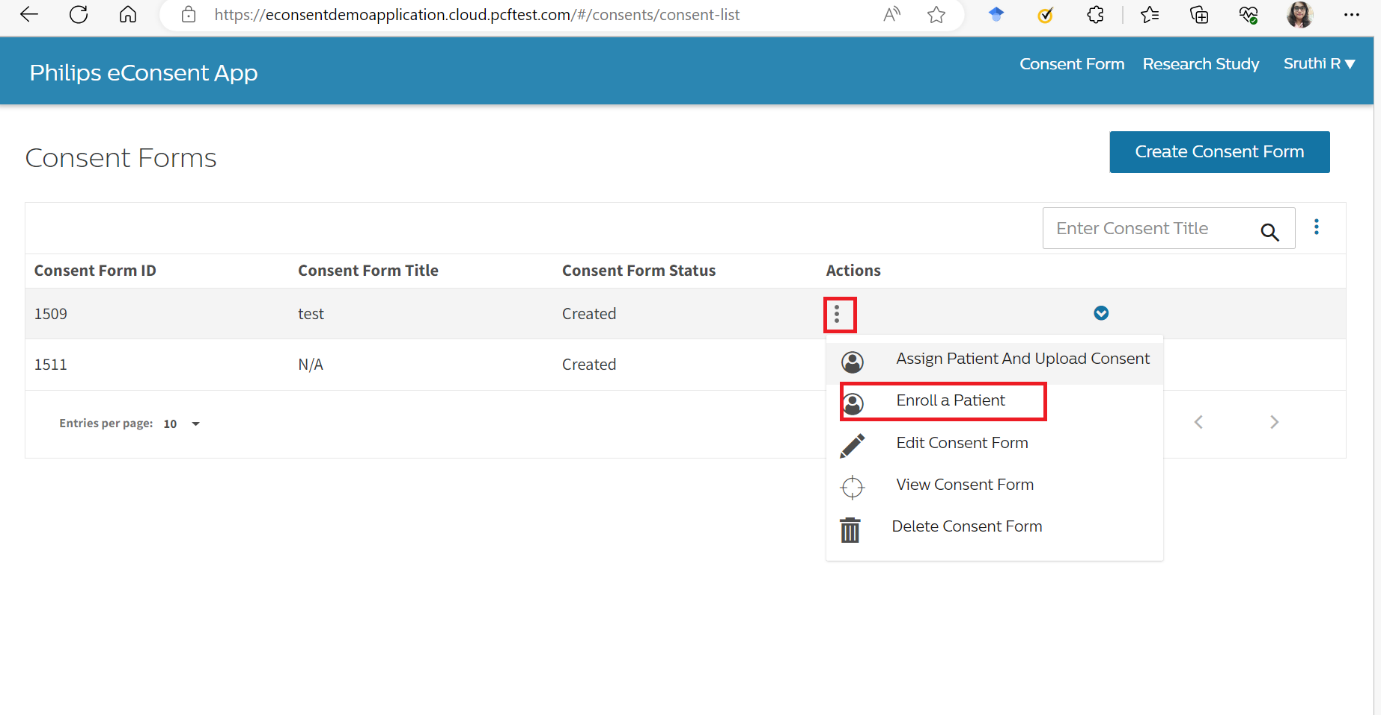
A screenshot of a computer

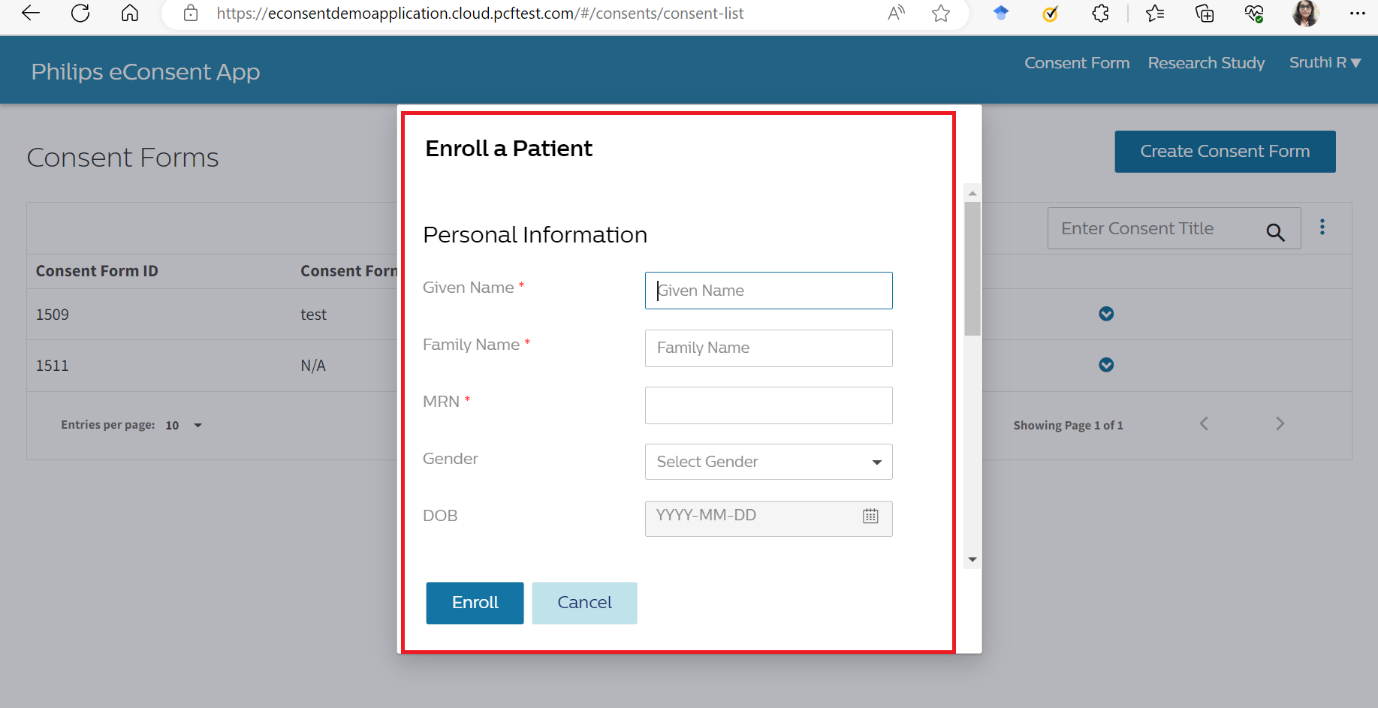
Description automatically generated

A screenshot of a computer

Description automatically generated

**Enroll Patient:** This feature enables the creation of new patients or enrolling existing patients into a specific consent form. It facilitates the association of patients with their respective consent forms. (selector name: app-enroll-patient)

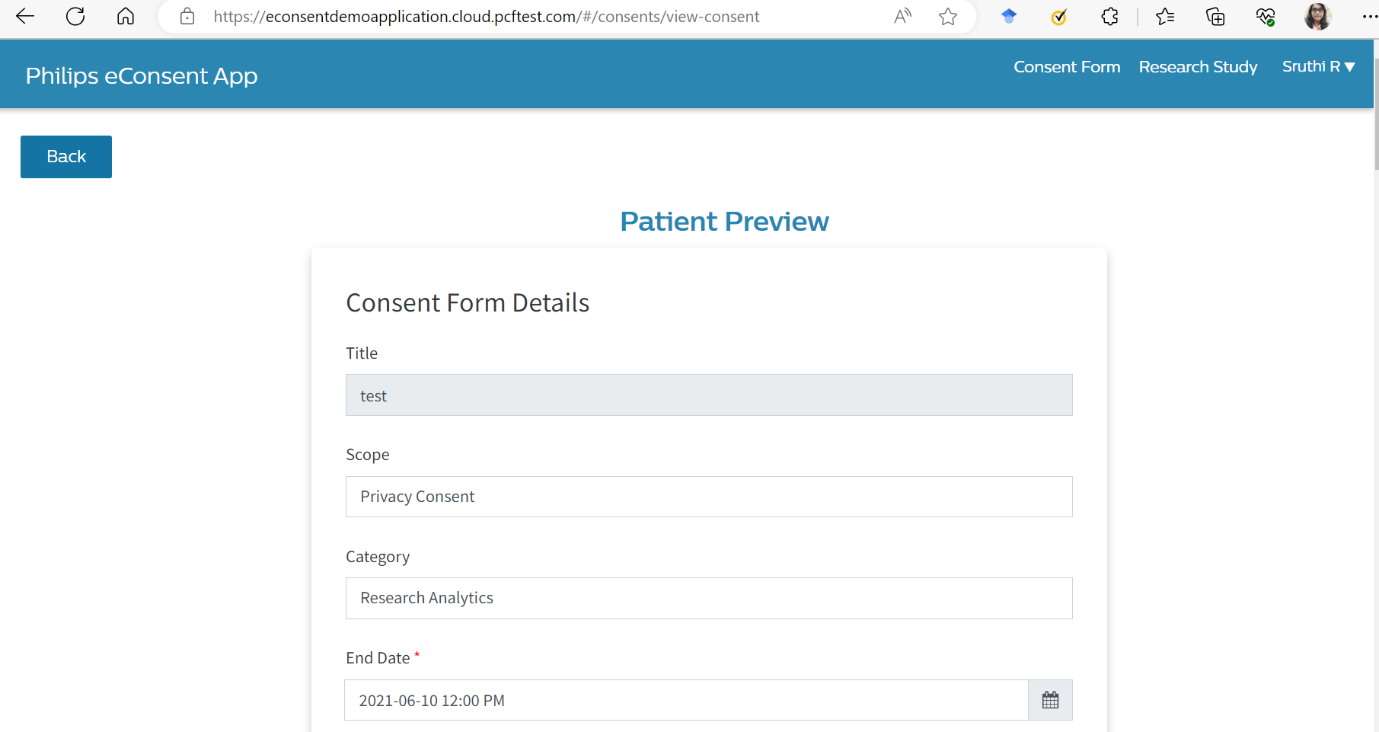




**View Consent:** Once a consent form is created, this component allows users to view the details and content of the consent form. It provides a user-friendly interface to navigate and review the information contained within the consent form. (selector name: app-view-consent)

A screenshot of a computer

Description automatically generated



Overall, the "Consents" component handles the management, creation, updating, enrollment of patients, and viewing of consent forms in the application.

Login module handles the login of each user. Based on the roles provided in the config list it redirects to different pages.

(Selector name : login-screen-example)

If we login as technician or admin or researcher then consents component functions are called)

If we login as patient patients component is called.

A screenshot of a computer login screen

Description automatically generated

This patient module has 5 components

Create update patient : in this module the the process of creating new patient account is initiated (selector name : app-create-update-patient)

Patient consent lists : this modules tells what are all the consent forms are there for the particular patient.

(selector: 'app-patient-consent-list')

Renew consent : here patient can extend the time for their consent.

(selector: 'app-view-patient')

Renew consent : here patient can getback their consent.

(selector: 'app-view-patient')

View-patient : the patient can view their details

(selector: 'app-view-patient')

Cdal-research study

A screenshot of a computer

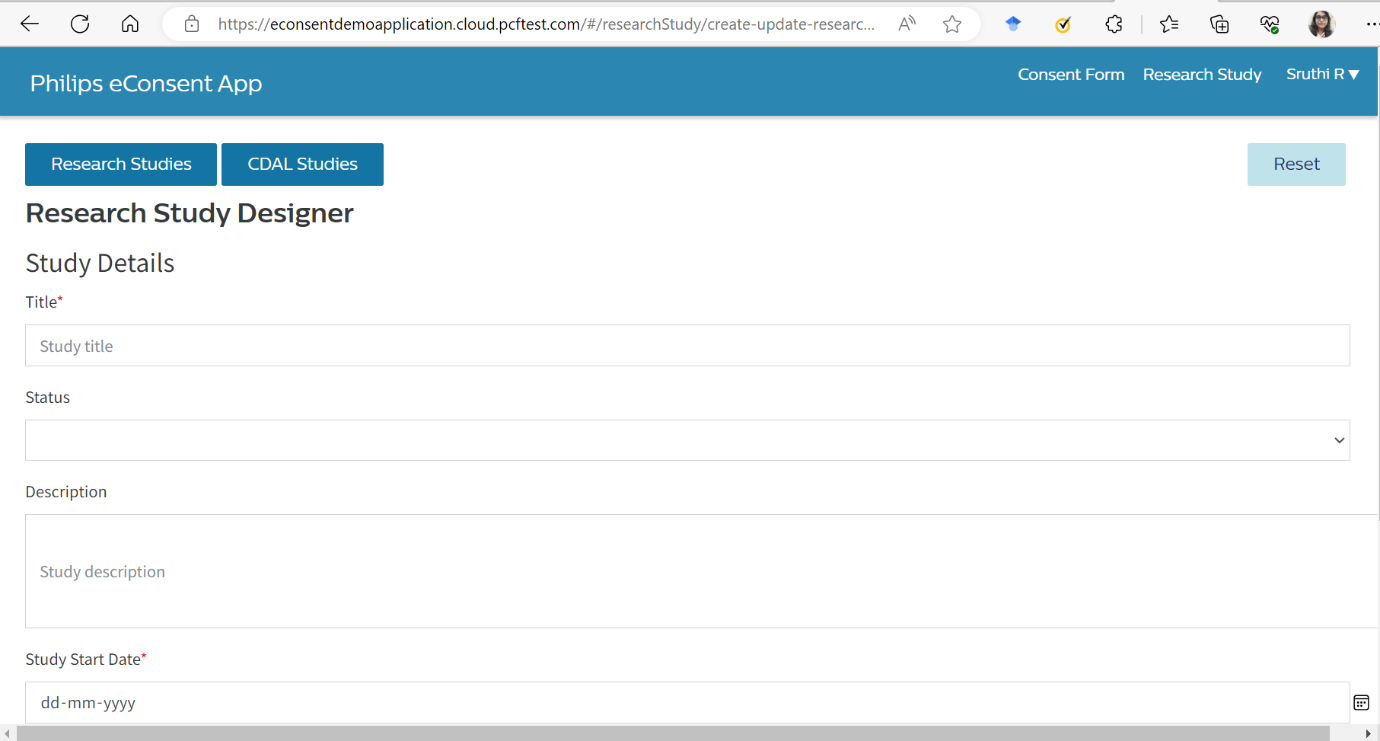
Description automatically generated

Create-update researchStudy : in this component we create a research study.

selector: 'app-create-update-researchstudy'

A screenshot of a computer

Description automatically generated

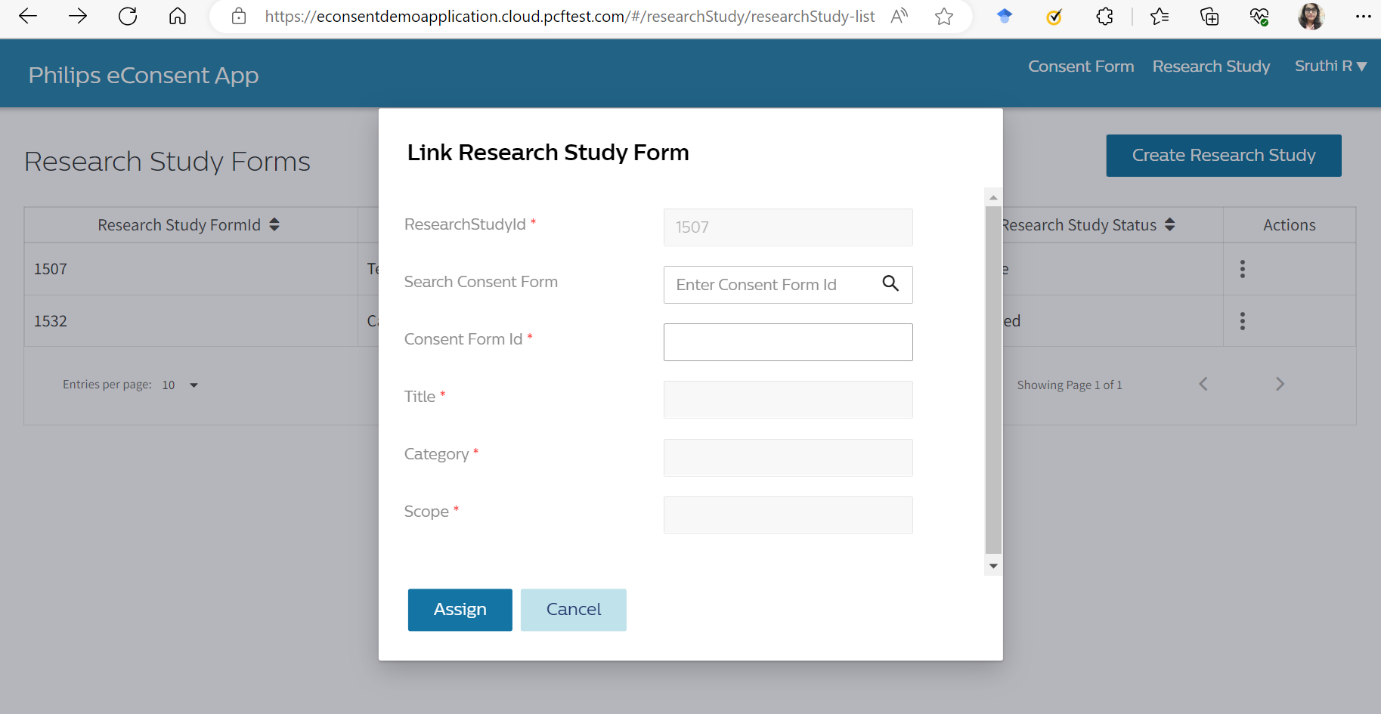


Link-study this module talks about adding a consent for to the research.

selector: 'app-link-study

A screenshot of a computer

Description automatically generated

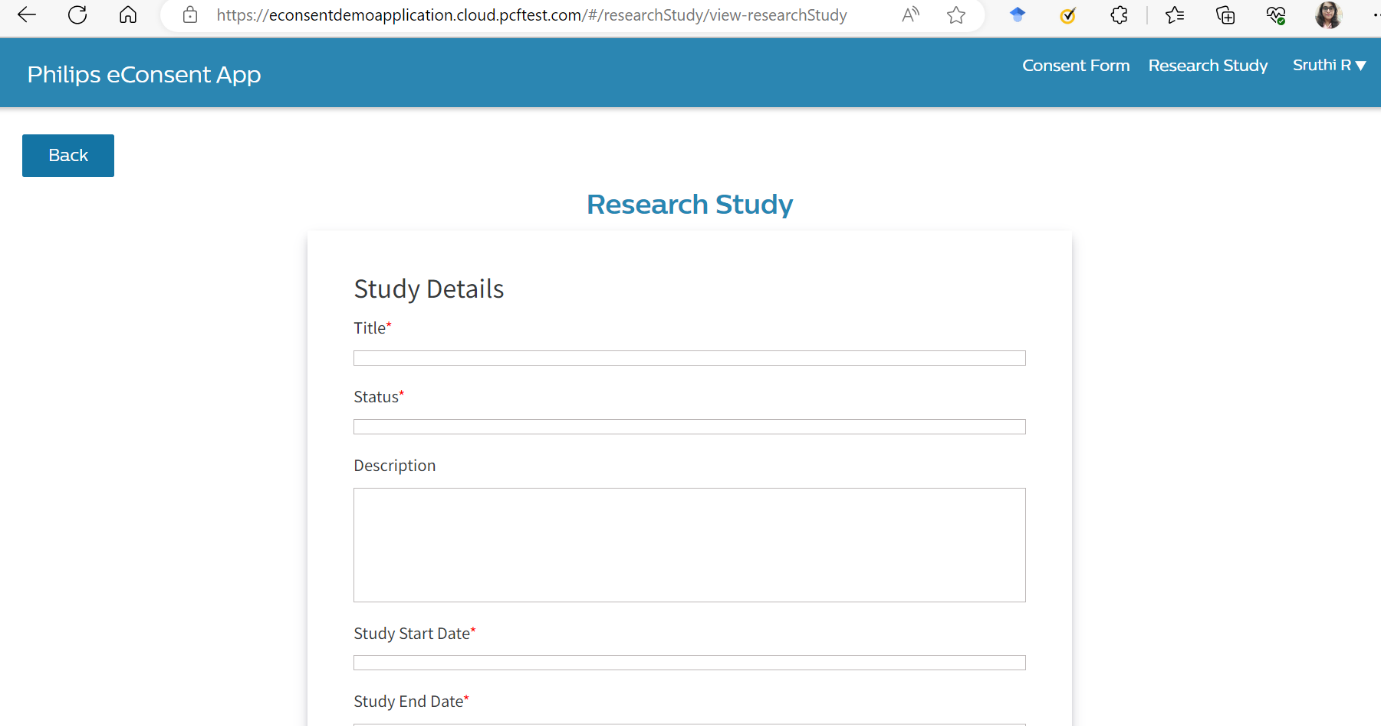


researchStudy : this component talks about what are all the research are already created.

A screenshot of a computer

Description automatically generated

View research study : this component shows you how a reseaarch study will look like.



Assets:

Here, you'll find static documents and files like supporting images and static values that are used throughout the application.

Environment:

If the environment is set to true the aplication will run in t site

If it set to false our application will run in local