R&D UC/SE PORTAL

Team Details:

- **B.G.V.Shiva (ee220002029@iiti.ac.in)**
- **P Madan (cse220001043@iiti.ac.in)**
- **M. Sai Varshit (cse220001044@iiti.ac.in)**

About this project:

The project aims to develop client-based custom software that is to be used by the employees of Research and Development Department (R&D) of IIT INDORE. Through the virtue of this software, the employees would be able to store the data of the grants, sanctions, expenses in a month wise balance sheet, annual balance sheet and in other custom documents like UCR and UCNR for all the projects/deals respectively without any interference or ambiguity between any two projects.

Objective:

As described, the objective of this project is to develop and deploy a portal to provide a comprehensive solution for managing the finance aspects of project data within the department.

Its primary functionalities include:

- Efficient storage and organization of grant, sanction and expense data of the project deals.
- Generation of month wise balance sheets and annul sheets.
- Creation of custom documents such as SOE, UCR and UCNR for individual projects or deals.
- Ensuring data integrity and preventing any ambiguity or overlap between different projects.

Technologies Used:

1. HTML: Used for creating the structure of web pages.

2. JavaScript: Implemented dynamic and interactive features on the client-side.

3. CSS: Employed styling and enhancing the presentation of web pages.

4. Ajax: Implemented for making asynchronous HTTP requests to the server, enabling dynamic updates without page reloads.

5. Python: Used as the primary backend programming language for server-side logic.

6. Django: Framework utilized for building web applications with Python, providing features like URL routing, database management, and template rendering.

Features:

1. Seamless data saving without the need to refresh the page, through the virtue of AJAX.

2. Enhanced page loading speed for improved user experience.

3. Integration of Google login for secure authentication.

4. Convenient download option for SOE/UCR/UCNR sheets covering any time interval within the project's start and end years.

5. Versatile viewing options available in both list and card styles for projects.

6. Intuitive search functionality enables users to easily find specific projects.

7. Efficient sorting capabilities based on alphabetical order and start year for streamlined project management.

8. Allow users to dynamically add new rows to the monthly sheet and rename them according to their preferences. Additionally, provide functionality for users to delete existing rows as needed, offering greater flexibility and customization in managing project data.

Running the project on a local machine/environment:

1. Download the folder from the GitHub link and ensure that Python and Django are installed on your local computer. Unzip the file and open it in a code editor.

2. Open the terminal and navigate to the folder containing the manage.py file using the following command.

- cd Rnd

3. Start the Django development server by running the command:

- python manage.py runserver

Note (to the Developers):

Initially the column “Sanctioned Amount” in the monthly sheet is conceived as “Grant Amount” while the column “Released Amount” is conceived as “Sanctioned Amount” and the logics of all the functions were written on those id’s and class's names. However at the latter stage, the frontend display name of the column “Grant Amount” is changed to “Sanctioned Amount” and the display name of the column “Sanctioned Amount” is changed to “Released Amount” while the id’s and class’s names weren’t altered.

VIEWS:

1.Duration(date1, date2):

Description: Calculates the duration between two dates in terms of financial years and months.

Methods:

- datetime.strptime(date, "%Y-%m-%d"): Converts the date string to a datetime object/
- Calculates the start and end months and years.
- Determines the financial year start and end indices based on the provided dates.

Returns a dictionary containing the count of financial years, financial year start and end indices, start and end months.

2. Createfile(Project_file_name, period_range):

-

Description: Creates a file with given project details and initializes data for each period range.

Methods:

- Iterates over the period range and initializes data for each table.
- Creates a JSON object with the initialized data.
- Writes the JSON object to a file.

3.index(request):

-Description: Handles the index page where users can input project details.

-Methods
:

- Handles POST requests where project details are received.
- Validates project details and checks if the project already exists.
- Calculates project duration and financial years using the duration function.
- Creates a project details object and saves it to the database.
- Calls createfile to create a file for the project.
- Renders the monthly page with relevant details.

5. mastersheet(request, project_id):

- Description: Renders the master sheet page for a specific project.

- Methods:

- Obtains the project ID and retrieves the corresponding project details.
- Parses and processes data from project files.
- Renders the master sheet page with relevant data.

6. get_monthewise_exp(project_id, financialYearStartYear, start_year1):

- Description: Retrieves month-wise expenses for a given project and financial year.

- Methods:

- Fetches project details from the database.
- Loads data from the project file.
- Extracts month-wise expense data for the specified financial year.
- Returns a dictionary containing month-wise expenses.

7. get_grants(project_id):

- Description: Retrieves total grants for each period of a given project.

- Methods:

  - Fetches project details from the database.
  - Loads data from the project file.
  - Iterates over the duration of the project and extracts total grants for each period.
  - Returns a list of total grants.

8. soe(request, project_id, period):

- Description: Renders the statement of expenditure (SOE) page for a specific project and period.

- Methods:

  - Obtains the project ID, financial year start year, and current period.
  - Retrieves project details and relevant data from files.
  - Calculates the financial year start year based on the current period.
  - Calls get_monthewise_exp and get_grants to obtain month-wise expenses and total grants.
  - Prepares data for rendering the SOE page.

9. save_table_data(request, project_id):

- Description: Saves table data for a project to a file.

- Methods:

  - Retrieves the project using the project ID.
  - Obtains the file path based on the project details.
  - Decodes JSON data from the request body and saves it to the file.
  - Returns a JSON response indicating success or failure.

10. fill(request, project_id):

- Description: Renders the filling page with project data for editing.

- Methods:

  - Retrieves the project details using the project ID.
  - Reads table data from the project file.

- Parses and prepares data for rendering the filling page.
- Renders the filling page with project data.

11. count_keys(d):

- Description: Counts the number of keys in a dictionary recursively.

- Methods:

  - Recursively counts the number of keys in the dictionary.
  - Returns the total count of keys.

12. save_tables_to_file(request, project_id):

- Description: Saves tables data to a file for a specific project.

- Methods:

  - Retrieves the project details using the project ID.
  - Obtains the file path based on the project details.
  - Decodes JSON data from the request body and saves it to the file.
  - Returns a JSON response indicating success or failure.

13. save_table_data_to_file(request):

- Description: Saves table data received from the request to a text file.

- Methods:

  - Parses JSON data from the request body.
  - Processes and saves the data to a text file.
  - Returns a JSON response indicating success or failure.

14. save_data_to_file(request):

- Description: Saves JSON data received from the request to a text file.

- Methods:

- Parses JSON data from the request body.
- Converts the JSON data to a text format.
- Overwrites the file with the new data.
- Returns a JSON response indicating success or failure.

15. logout(request):

- Description: Logs out the user and redirects to the homepage.

- Methods:

  - Logs out the user using auth_logout.
  - Renders the homepage template.

16. save_table_data1(request, project_id):

- Description: Saves table data received from the request to a text file specific to the project.

- Methods:

  - Parses JSON data from the request body.
  - Processes and saves the data to a text file associated with the project.
  - Returns a JSON response indicating success or failure.

17. project_search(request):

- Description: Searches for projects based on the query string and renders the project list template.

- Methods:

  - Retrieves the query string from the request.
  - Filters projects based on the query using Q objects.
  - Renders the project list template with the filtered projects.complete_task(request, project_id):

18. mark_project_completed(request, project_id):

- Description: Marks a project task as completed and redirects to the project list view.

- Methods:

  • Retrieves the project using the project ID.

  • Sets the task field of the project to "completed" and saves it.

  • Redirects to the project list view.

19. project_listwise(request):

- Description: Renders the project list template with an option to filter and sort projects.

- Methods:

  • Fetches all projects from the database.

  • Applies filtering based on the filter option selected.

  • Sorts projects based on the specified column and order.

  • Renders the project list template with the sorted and filtered projects.

20. save_as_pdf(request):

- Description: Converts base64 image data to a PDF file and provides a download link.

- Methods:

  • Parses JSON data from the request body.

  • Converts base64 image data to binary and saves it as a temporary image.

  • Generates a PDF with the image using ReportLab.

  • Deletes the temporary image and saves the PDF as a temporary file.

  • Provides a download link for the PDF.

21. download_pdf(request):

- Description: Serves the generated PDF file for download.

- Methods:

  • Retrieves the PDF file path from the request.

  • Creates an HTTP response with the PDF content type and attachment disposition.

  • Writes the PDF file content to the response.

22. submit_project_data(request, project_id):

- Description: Updates project data based on the submitted form data.

- Methods:

  - Retrieves the project object using the project ID.
  - Updates project fields with the submitted form data.
  - Saves the updated project object.
  - Redirects to the project list view upon success or handles the error accordingly.

23. delete_project(request, project_id):

- Description: Deletes a project based on the provided project ID.

- Methods:

  - Retrieves the project object using the project ID.
  - Deletes the project object.
  - Renders the project list template with updated project data.SOE_navigation(request, project_id):

24. SOE_navigation(request, project_id):

- Description: Renders the SOE navigation template for navigating through financial years.

- Methods:

  - Retrieves the existing project object using the project ID.
  - Generates financial years based on the project's start and end years.
  - Renders the SOE navigation template with financial year data.

25.Homepage(request):

- Description:

  This function is responsible for rendering the homepage template/

- Methods:

- Returns the homepage template.

26.login(request):

- Description:

This function handles POST requests where the username and password are obtained. It attempts to authenticate the user using Django's authenticate method. If authentication is successful, it redirects the user to the homepage; otherwise, it displays an error message and renders the login page.

- Methods:

1. Handles POST requests.
2. Attempts user authentication.
3. Redirects to the homepage upon successful authentication.
4. Displays error message and renders login page upon authentication failure.

27.project_list(request):

- Description: Renders a list of projects with filtering options.

- Methods:

- Retrieves all project details from the database.
- Filters projects based on the provided filter option (if any).
- Renders the project list page with the filtered projects.

28.ucr(request, project_id):

- Description: Renders the UCR (User Change Request) page for a specific project.

- Methods:

- Obtains the project ID and renders the UCR page.

29.monthly(request):

- Description: Handles the monthly page where users can input project details.

- Methods:

- Handles POST requests where project details are obtained.

- Calculates the project duration based on start and closure dates.
- Creates or updates a project record in the database.
- Renders the monthly page with relevant details.

JAVA SCRIPT methods pagewise:

Filling

1. addEventListener('DOMContentLoaded', function ()):

- Description: Listens for the DOMContentLoaded event, indicating that the HTML document has been fully loaded and parsed.

- Methods:

- Initializes table data and values.

- Adds event listeners for table manipulation functions.

2. initializeTableData(table):

- Description: Initializes data for a specific table by creating a JavaScript object to hold input field values.

- Methods:

    - Retrieves input fields of type number within the table

.

- Creates a JavaScript object to store the input field values.

3. initializeTableValues():

- Description: Initializes the values of all input fields of type number to zero.

- Methods:

    - Retrieves input fields of type number

.

- Sets their values to zero.

4. addRows(tableId, numRows):

- Description: Dynamically adds rows to a table based on the provided table ID and the number of rows to add.

- Methods:

- Retrieves the table and its tbody element.

    - Inserts new rows with appropriate input fields.

5. updateCellValue():

- Description: Updates the value of a specific cell in a table based on user input.

- Methods:

    - Retrieves user input values

.

- Finds the corresponding cell and updates its value.

6. deleteRow():

- Description: Deletes the last row from all tables on the page.

- Methods:

- Finds the last row in each table and removes it.

7. addRow():

- Description: Adds a new row to all tables on the page.

- Methods:

    - Constructs HTML for a new row

.

    - Appends the new row to each table.

8. updateTable(tableId):

- Description: Updates the table data and recalculates totals and closing balances for a specific table.

- Methods:

    - Retrieves table data

.

    - Recalculates total amounts, expenses, and closing balances

.

- Updates table display.

9. getTableData(tableId):

- Description: Retrieves data from input fields within a specific table and returns it as a JavaScript object.

- Methods:

    - Retrieves input fields within the table

.

- Constructs a JavaScript object to store field values.

10. calculateAndDisplayTotalExpenses(tableId):

- Description: Calculates and displays total expenses for each row in a specific table.

- Methods:

- Iterates through rows, calculates total expenses, and updates corresponding input fields.

11. calculateAndDisplayClosingBalance(tableId):

- Description: Calculates and displays closing balance for each row in a specific table.

- Methods:

 - Retrieves sanctioned amount and total expenses for each row.

- Calculates closing balance and updates corresponding table cells.

12. calculateGrantAmount(table, columnIndex):

- Description: Calculates the total grant amount for a specific column in a table.

 - Methods:

  - Iterates through rows, calculates grant amounts, and returns total.

13. calculateSanctionedAmount(table, columnIndex):

 - Description: Calculates the total sanctioned amount for a specific column in a table.

- Methods:

- Iterates through rows, calculates sanctioned amounts, and returns total.

14. calculateAndDisplayTotalAmount(tableId):

- Description: Calculates and displays total amount for each column in a specific table.

- Methods:

- Iterates through columns, calculates total amounts, and updates corresponding table cells.

15. convertToInputFieldName(monthText):

- Description: Converts text representation of a month to its corresponding input field name.

- Methods:

  - Splits month text to get abbreviation.

  - Returns corresponding input field name.

16. saveTableDataToDjango(data, project_id):

- Description: Sends table data to a Django backend for saving using a POST request.

- Methods:

- Constructs POST request with table data.

- Sends request to Django backend.

17. getCookie(name):

- Description: Retrieves the value of a specific cookie by its name from the browser's cookies.

- Methods:

- Parses browser cookies to find the specified cookie by name.

- Returns its value.

Login

1. toggleShowSI():

- Description: Toggles the visibility of the password input field for sign-in.

- Methods:

    - Retrieves the sign-in password input field and its associated show/hide icon

.

- Changes the input field type between "password" and "text" to show or hide the password.

- Updates the show/hide icon accordingly

.

- Adds or removes a CSS class for styling purposes.

2. toggleShowSU():

- Description: Toggles the visibility of the password input field for sign-up.

- Methods:

- Retrieves the sign-up password input field and its associated show/hide icon

.

- Changes the input field type between "password" and "text" to show or hide the password.

- Updates the show/hide icon accordingly

.

- Adds or removes a CSS class for styling purposes.

3. create.addEventListener("click", () => {...}):

- Description: Handles the click event for the "Create Account" button to switch between sign-in and sign-up forms.

- Methods:

- Retrieves the containers for the sign-in and sign-up forms

.

- Hides the sign-in container and displays the sign-up container

.

- Applies CSS classes for animation effects to transition between the two forms.

4. log.addEventListener("click", () => {...}):

- Description: Handles the click event for the "Log In" button to switch between sign-up and sign-in forms.

- Methods:

- Retrieves the containers for the sign-in and sign-up forms

.

- Hides the sign-up container and displays the sign-in container.

- Applies CSS classes for animation effects to transition between the two forms.

Masterhseet

1. getcellvalue(cellId):

- Description: Retrieves the value of a specific cell in the data table corresponding to the grant amount.
- Methods:

- Splits the cell ID to extract the table key, row, and column indices

.

- Checks if the data for the specified column and row exists in the tablesData object.

- Returns the grant amount value if found, otherwise returns "Not found".

2. getCellValue0(cellId):

- Description: Retrieves the value of a specific cell in the data table corresponding to the sanctioned amount.

- Methods: Similar to
getCellValue(cellId), but retrieves the sanctioned amount value.

3. getCellValue1(cellId):

- Description: Retrieves the value of a specific cell in the data table corresponding to the expenses.

- Methods: Similar to
getCellValue(cellId), but retrieves the expenses value.

4. fillCells0():

- Description: Fills the cells of the data table with the sanctioned amount values.

- Methods:

    - Retrieves the table body and rows of the data table

.

- Iterates over each row and cell, updating the cell content with the sanctioned amount value.

5. fillCells():

- Description: Fills the cells of the data table with the grant amount values.

- Methods: Similar to fillCells0(), but fills cells with grant amount values.

6. fillCells1():

- Description: Fills the cells of the data table with the expenses values.

- Methods: Similar to fillCells0(), but
fills cells with expenses values.

7. fillCells2():

- Description: Fills the cells of the data table with the text values
.

- Methods:

    - Retrieves the rows and cells of the data table
.

    - Identifies
text cells and updates their content with text values from the tablesData object.

8. updateCellValue():

- Description: Updates the value of a specific cell in the data table.

- Methods:

- Retrieves the total cell value for each row from the tablesData object.

- Updates corresponding cells in the data table with the total cell values.

9. deleteLastRow():

- Description: Deletes the last row from the data table
.

- Methods:

- Retrieves the table body and checks for the last row
.

- Deletes the last row if it's
not the "Total" row and updates the file accordingly.

10. updateFile():

- Description: Updates the file with the contents of the data table.

- Methods:

- Initializes an empty cellContents object.

- Iterates over each row and cell in the data table, updating the cellContents object.

- Sends the updated cellContents object to the server via AJAX for saving to the database.

11. fillTableWithData():

- Description: Fills the data table with parsed data retrieved from the server.

- Methods:

    - Retrieves the table body and the total row
.

- Iterates over each row in the parsed data and fills the corresponding cells with data.

    - If a row does not exist, adds a new row and fills
it with data.

12. initializeRowData():

- Description: Initializes the row data object with the current content of the data table.

- Methods:

- Retrieves the table body and initializes an empty object.

- Iterates over each row and cell in the data table, storing the content in the object.

13. addRow():

- Description: Adds a new row to the data table.

- Methods:

  - Retrieves the table body and the total row.

  - Creates a new row element and sets its ID.

  - Adds cells to the new row for each column, including inputs for date columns.

  - Inserts the new row before the total row in the table body.

  - Updates the row data object with the content of the new row.

14. Event Listener:

- Description: Listens for the DOMContentLoaded event and triggers various functions.

- Methods:

  - Fills cells of the data table with different types of data.

  - Calls fillTableWithData() twice to ensure data consistency.

- Updates the total row with parsed data.

- Updates cell values based on the row data object.

  - Listens for clicks on the update button and updates table data accordingly.

- Sends updated table data to the server using fetch API.

15. sendToServer(data, project_id):

- Description: Sends data to the server using the Fetch API.

- Methods:

- Constructs the URL for the server endpoint based on the project_id.

- Sets the method to 'POST' for sending data

.

- Constructs headers with content type as JSON and includes CSRF token

.

- Converts the data object to JSON format and includes it in the request body

.

- Handles the response from the server, logging the result or any errors to the console.

16. updateTotalCell():

- Description: Updates total cells in the table based on the content of the table.

- Methods:

- Iterates over each row of the table to update the total cells

.

- Retrieves the total cell and corresponding cells containing values for summation.

- Calculates the sum of values in the cells and updates the total cell with the computed sum.

- Handles the computation separately for different row types (e.g., 1-row-${j}, 0-row-${j}, row-${j}).

17. getCookie(name):

- Description: Retrieves the value of the specified cookie from the browser's cookie storage.

- Methods:

    - Parses the document's cookie string to extract individual cookie key-value pairs
.

- Iterates over each cookie to find the one with the specified name.

- Returns the value of the cookie if found, otherwise returns null.

- Utilizes decodeURIComponent to handle URL encoding in cookie values.

18. Initialization:

- Description: Initializes variables and performs calculations when the DOM content is loaded.
- Methods:

- Retrieves the project ID from the DOM.

    - Iterates over budget heads and financial years
to calculate total grants, sanctions, and expenses.

- Updates the corresponding total cells and sums for each financial year.

    - Calculates the total funds granted, sanctioned, and expended
across all years.

- Initializes input fields with zero values for specific columns.

19. Update Total Balance:

- Description: Recalculates and updates the total balance amount.

- Methods:

- Iterates over budget heads to compute the balance amount for each row.

- Retrieves total funds, expenses, and committed expenditures for each row.

- Calculates the balance amount and updates input fields accordingly.

- Updates the total balance amount across all domains.

20. Calculate Total Balance:

- Description: Calculates and updates the total balance amount.

- Methods: - Similar to
the update total balance function but only calculates the total balance without updating input fields.

21. Initialize Zero:

- Description: Initializes specific input fields with zero values.

- Methods:
  - Finds input fields corresponding to specific columns and initializes them with zero values.

26.updateTotalRow :

Description:
- The function updates the total row in a data table with calculated sums based on the provided parsed data.

. Methods:
- Selecting Total Row:

- Selects the total row from the table's tbody element.

- Logs an error if the total row is not found and exits the function.

- Calculating Column Sums:

- Loops through columns 5 to 6 (inclusive) to calculate the sum for each column.

- Iterates through each row in the parsed data and accumulates the values for the corresponding column.

- Updates the total cell in the total row with the calculated sum.

- Calculating Budget Head Sums:

- Iterates through each budget head.

- Loops through each row in the parsed data and accumulates the values for the corresponding budget head.

- Updates the total cell in the total row with the calculated sum.

- Update Cell Values: - Calls the updateCellValue function after updating the total row.

3. Additional Notes:

- The function uses the provided parsed data to calculate the sums.

- It updates the total row with the calculated sums for specific columns and budget heads.

Project list

1. showAllProjects:

-

Description: Redirects to the same page without any filter.

- Method: Updates the window location to the project list URL without any filter parameter.

2. showOngoingProjects:

-

Description: Redirects to the same page with a filter for ongoing projects.

- Method: Updates the window location to the project list URL with a filter parameter set to "ongoing".

3. showCompletedProjects:

- Description: Redirects to the same page with a filter for completed projects.

- Method: Updates the window location to the project list URL with a filter parameter set to "completed"
.
 - Additional: Also modifies the CSS classes of HTML elements ongoing and completed, adding/removing the "active1" class. However, the code to remove the "active1" class from ongoing is missing.

4. confirmDelete:

- Description: Displays a confirmation dialog to the user before deleting a project.

- Method: Uses the confirm function to show a confirmation dialog with the message "Are you sure you want to delete this project?". Returns true if the user clicks OK, allowing the deletion to proceed. Returns false if the user clicks Cancel, preventing the default action (deleting the project).

Project listwise

1. confirmSubmission:

- Description: Displays a confirmation dialog to the user before submitting a form.
- Parameters:
  - projectId: The ID of the project associated with the form.

- Method:
  - Shows a confirmation dialog with the message "Are you sure you want to submit?".

    - If the user confirms (clicks OK), the form with the ID form_projectId (constructed dynamically based on the projectId) is submitted.
      - If the user cancels (clicks Cancel), the form submission is not executed.

SOE/UCR/UCNR navigation

1. Dropdown Initialization:

- Method: initializeDropdown

- Description: Initializes the SOE/UCR/UCNR dropdown menu by populating it with periods retrieved from the backend.

2. Populating Dropdown:

- Method: populateDropdown

- Description: Populates the dropdown menu with periods retrieved from the backend. Ea
ch period is represented as an anchor element within the dropdown menu.

3. Dropdown Toggle:

- Method: toggleDropdown

- Description: Toggles the visibility of the dropdown menu when the SOE/UCR/UCNR dropdown button is clicked.

4. Dropdown Close on Click Outside:

- Method: closeDropdownOnClickOutside

- Description: Closes the dropdown menu when the user clicks outside of it. This method ensures that the dropdown closes to improve user experience and navigation.

5. Event Listeners:

- Method: addEventListeners

- Description: Adds event listeners to the SOE/UCR/UCNR dropdown button and the window to handle toggling and closing the dropdown menu.

References:

- Django Documentation.
- JavaScript Documentation.
- Google Auth Documentation.
- AI tools.