**Riddhish Atul Bharadva**

**MSC-Computing (Cloud Computing)**

**Assignment 2 (Task Manager)**

**Student Id: 3009299**

**Assignment Description:**

Aim of this assignment is to create a task manager where users will be able to create a new task board post login to their user account. Each task board may or may not (When task board is generated) have tasks. Each user that creates task board will be able to invite another registered user available in our database. Task board admin can invite or remove user to his created task board. Main objective of this assignment is to fetch all the data using Key. Hence, I am not using query() to fetch and filter data in this Assignment.

**Files / Pages:**

In this assignment, I have created 2 types of Files i.e. files with “.py” extension and files with “.html” extension. Files with extension “.py” are responsible for backend working of this project and files with “.html” are responsible for frontend or UI working of this project. Below are the files in detail description of each.

HTML files involved are:

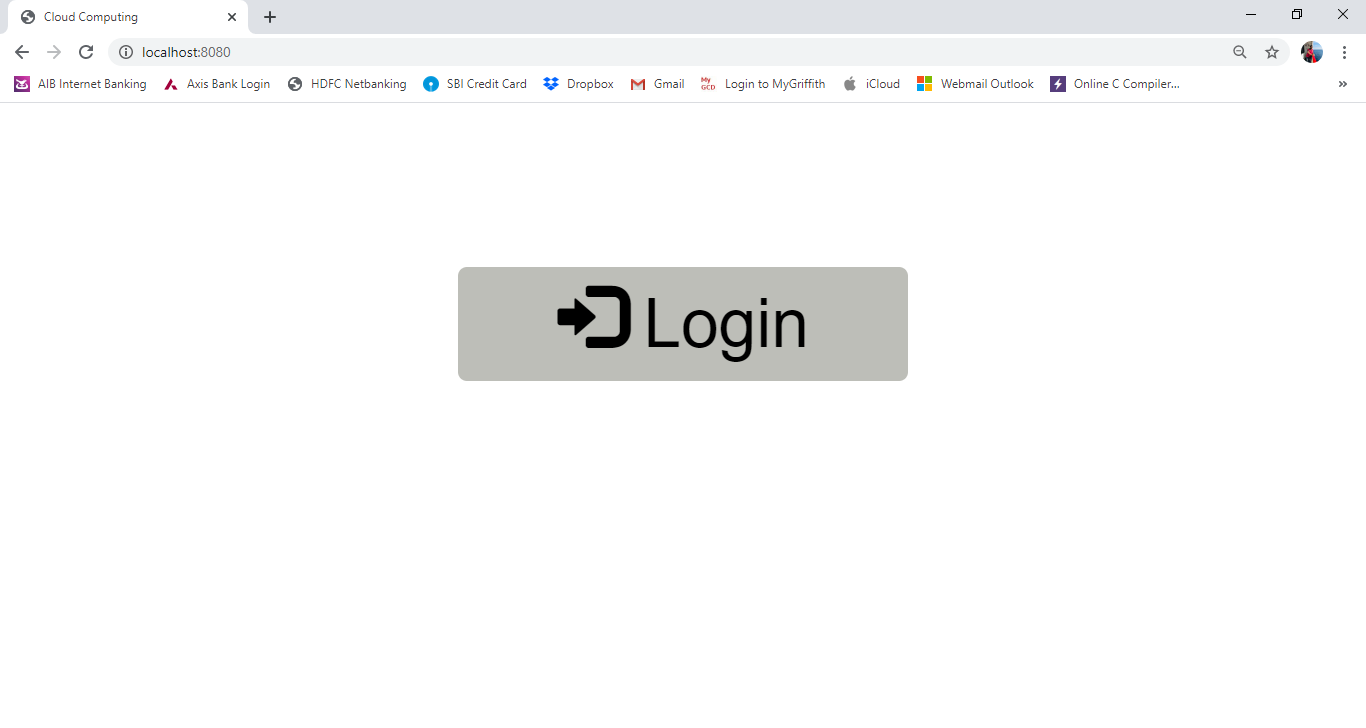
1. Main.html
2. TaskBoardData.html
3. EditTask.html

Python files involved are:

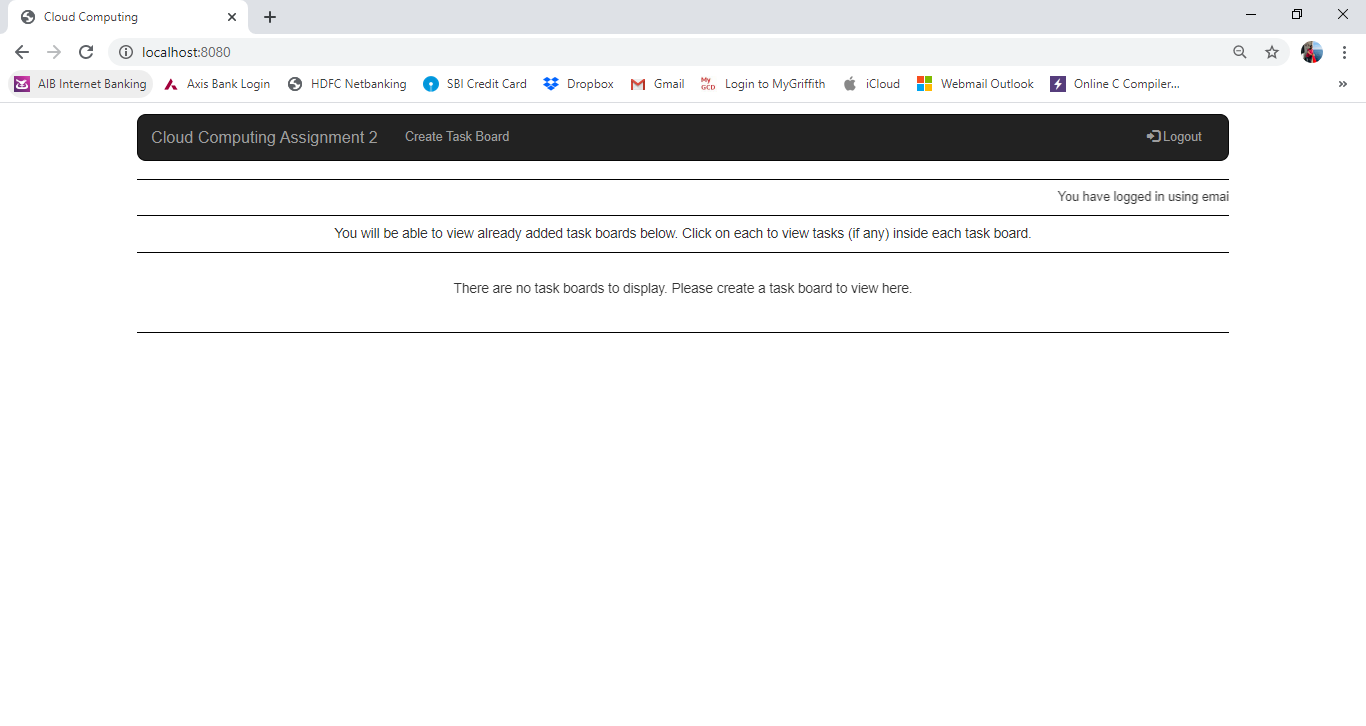
1. Main.py
2. TaskBoardData.py
3. EditTask.py
4. UserDB.py
5. TaskBoardDB.py
6. TaskDB.py

**Main.html**

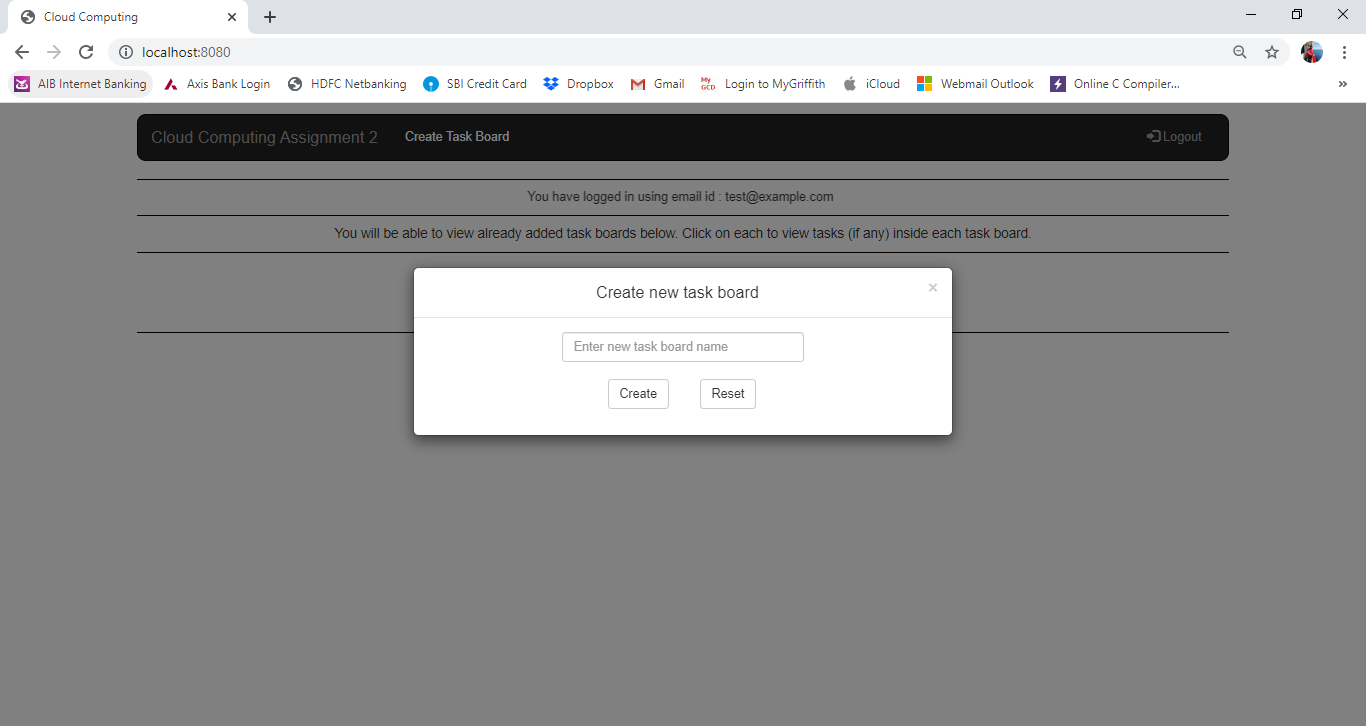
Main.html is the front end page which will be displayed to user when he will start this web application. In case user is not logged in, Main.html: Image 1 will be displayed else, Main.html: Image 2 will be displayed.



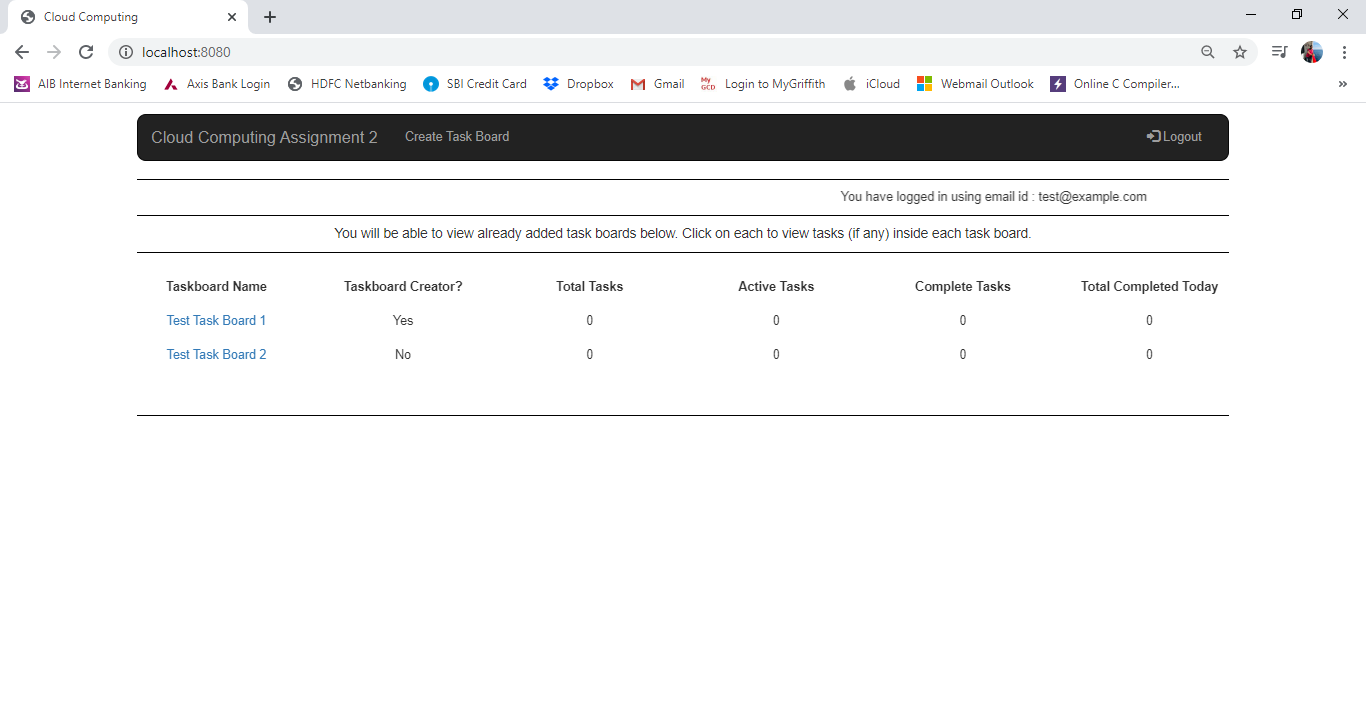
Main.html: Image 1



Main.html: Image 2



Main.html: Image 3



Main.html: Image 4

As shown in above Images 1, 2, 3, 4, etc. in case user have just logged in for first time and no task board have been created, Main page displayed will be as shown in Image 2. User can now click on “Create Task Board” to create a new task board.

As soon as user clicks on “Create Task Board”, “Create new task board” model will be displayed as shown in Image 3. User is expected to provide with task board name he/she wishes to create and click on “Create” button.

If user have already created task board or is being invited by another user, main page displayed is shown in Image 4. All the taskboard names are hyperlinks clicking on which takes users into detailed view of taskboard.

User will also be able to view “Taskboard Creater, Total Tasks, Active Tasks, Complete Tasks, Total Completed Today”.

Taskboard Creator: This shows if the logged in user is creator of respective task board displayed in Taskboard Name or not.

Total Tasks: This shows total number of tasks in taskboard.

Active Tasks: This shows total number of tasks currently active in taskboard.

Complete Tasks: This shows total number of tasks completed in taskboard.

Total Completed Today: This shows total number of tasks completed today from number of completed tasks in taskboard.

**Main.py**

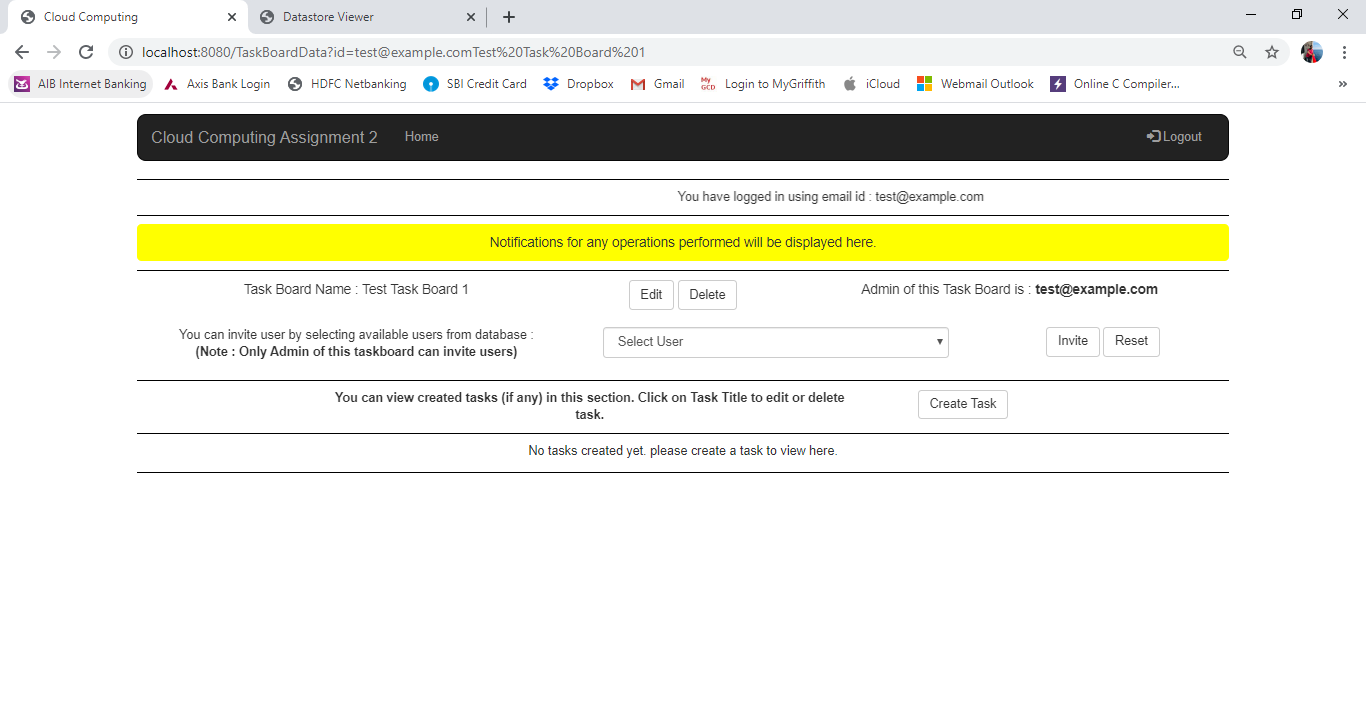
In Main.py, I am fetching data from all required datastores. On main page, I am displaying data like taskboard name, task counts, etc. Hence, I need to fetch data from all the models in Main.py.

After fetching data, I am calculating all the values of Total Task, Task Completed, Active Tasks, Task Completed Today, etc. and also passing them all along with database’s fetched data to html page in order to display same in required format.

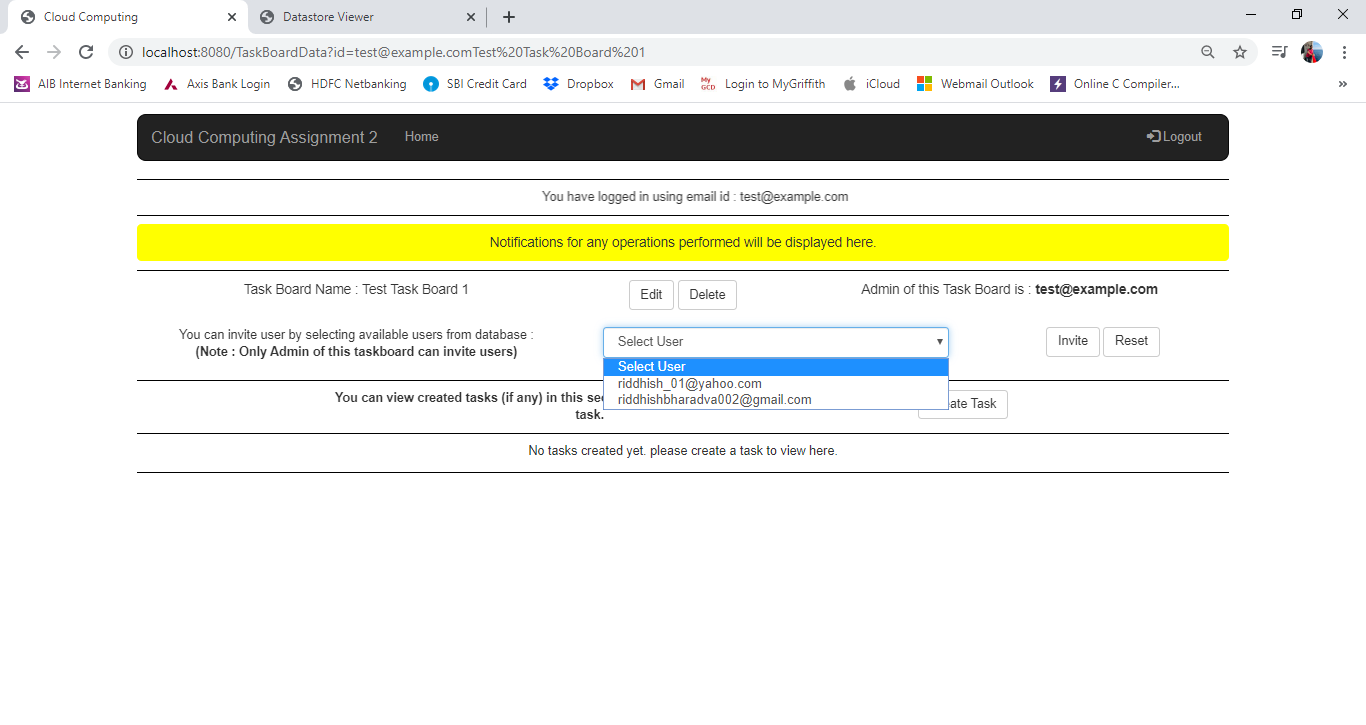
I have written a logic in post method of Main.py to create a new taskboard. As per logic, before creating a new taskboard, it will check if the taskboard with same name exists in database or not? If it exist, new taskboard with provided name will not be created else, a new taskboard with provided name will be created.

**TaskBoardData.html**

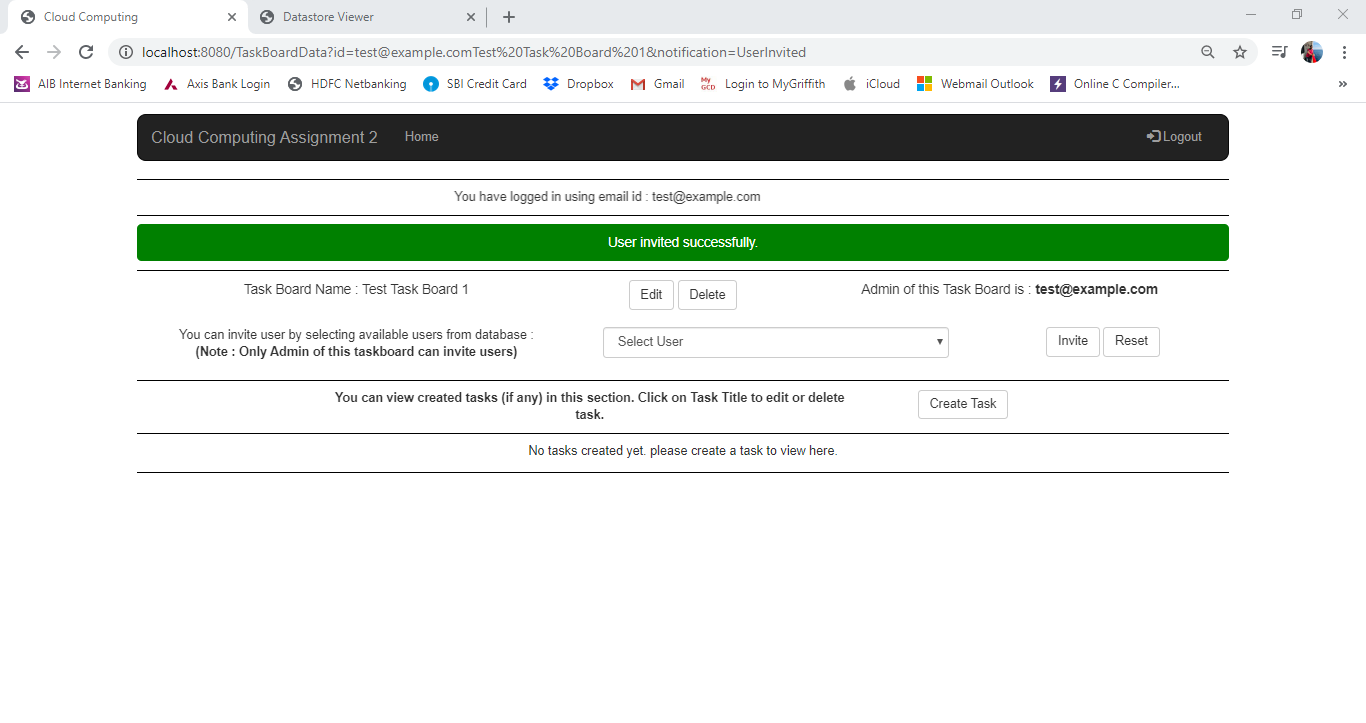
TaskBoardData.html contains a lot of functionalities within a single page. These includes Inviting users in taskboard, Editing Taskboard name, removing users, Deleting Taskboard, creating tasks, editing tasks, assigning a task to user, marking complete or incomplete, deleting a task, etc. All these functionalities are defined within same html page by calling different condition in backend .py page. Below are the screenshots of this page with different functionalities.



TaskBoardData.html: Image 1

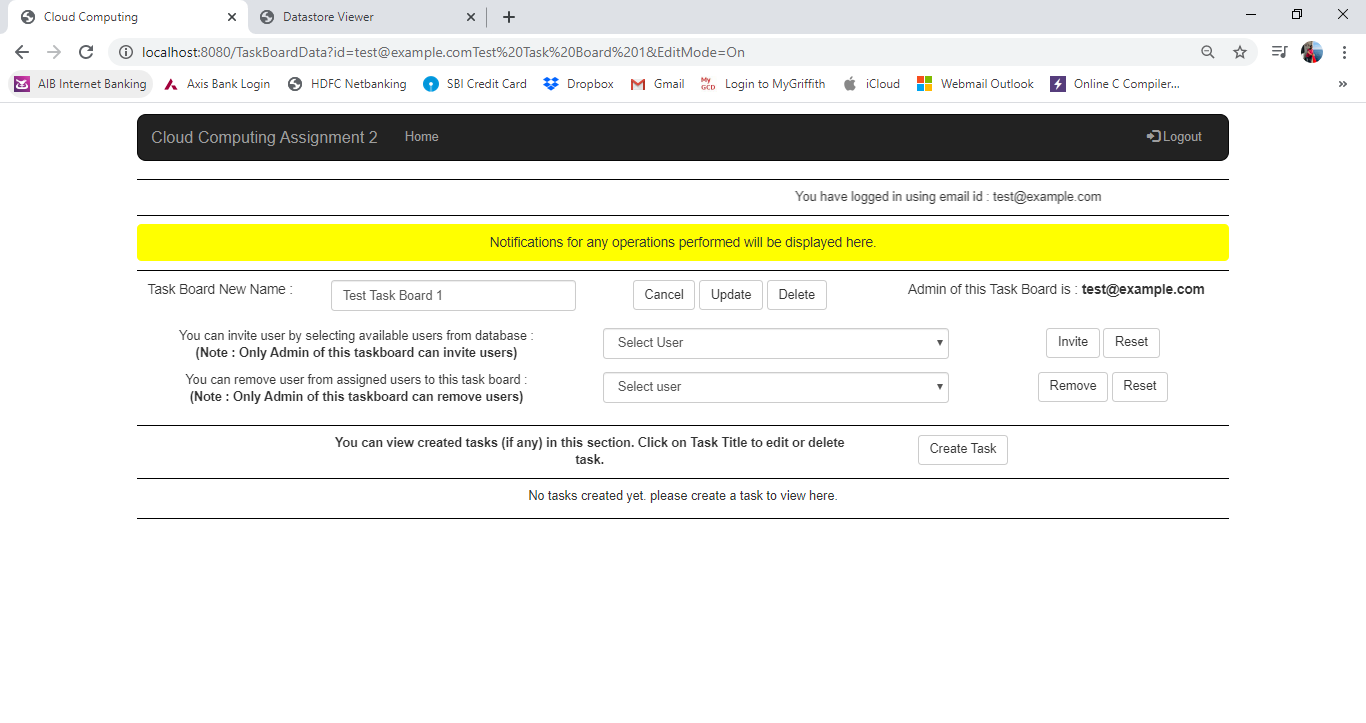


TaskBoardData.html: Image 2.1



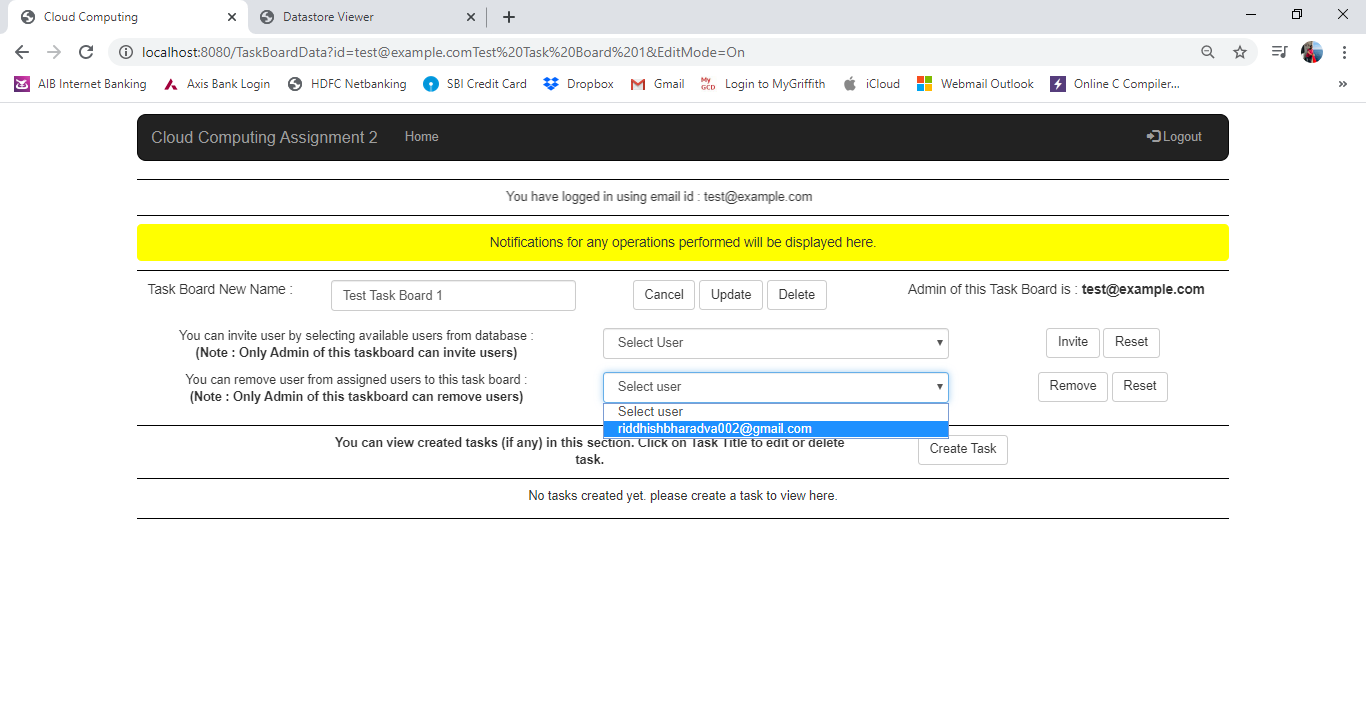
TaskBoardData.html: Image 2.2

As seen in above Image 2.1 and 2.2, any available user in database can be invited to created taskboard from given selection box.

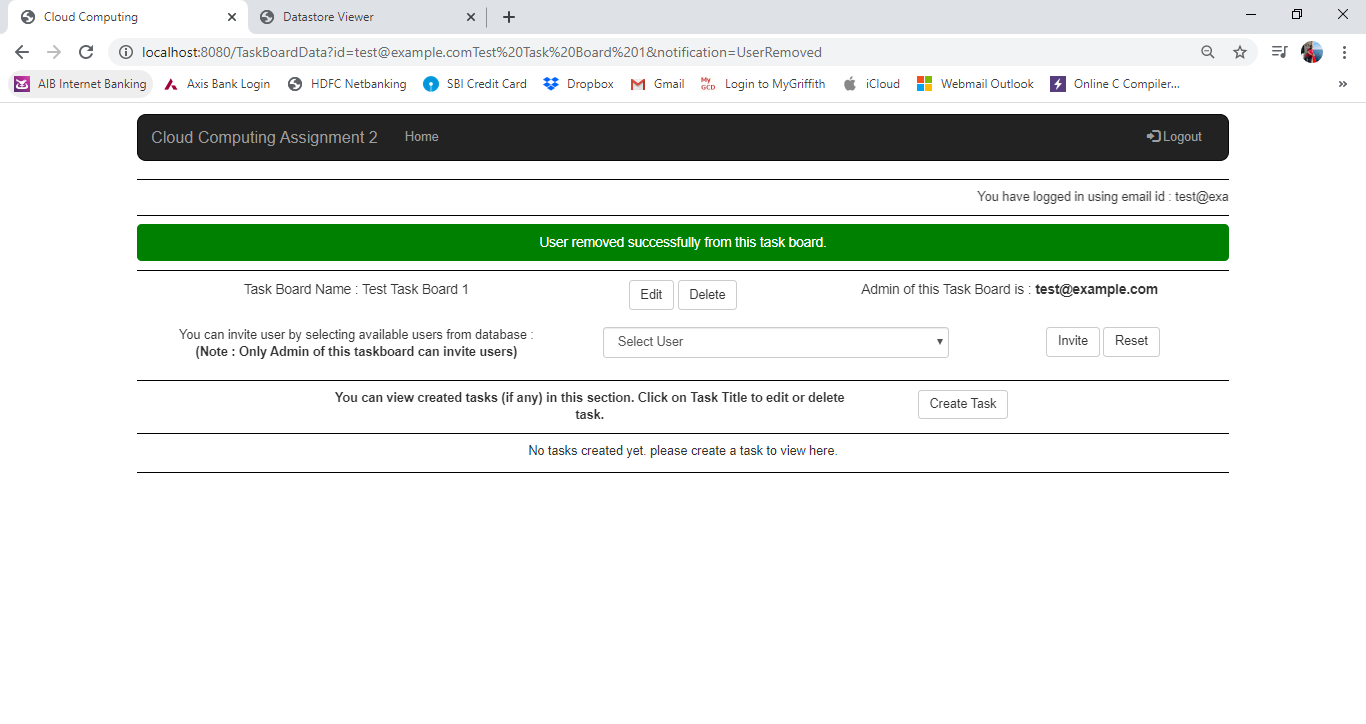


TaskBoardData.html: Image 3

As shown in Image 3, Renaming taskboard, removing access of users from taskboard and deleting taskboard can be performed on clicking “Edit” button.

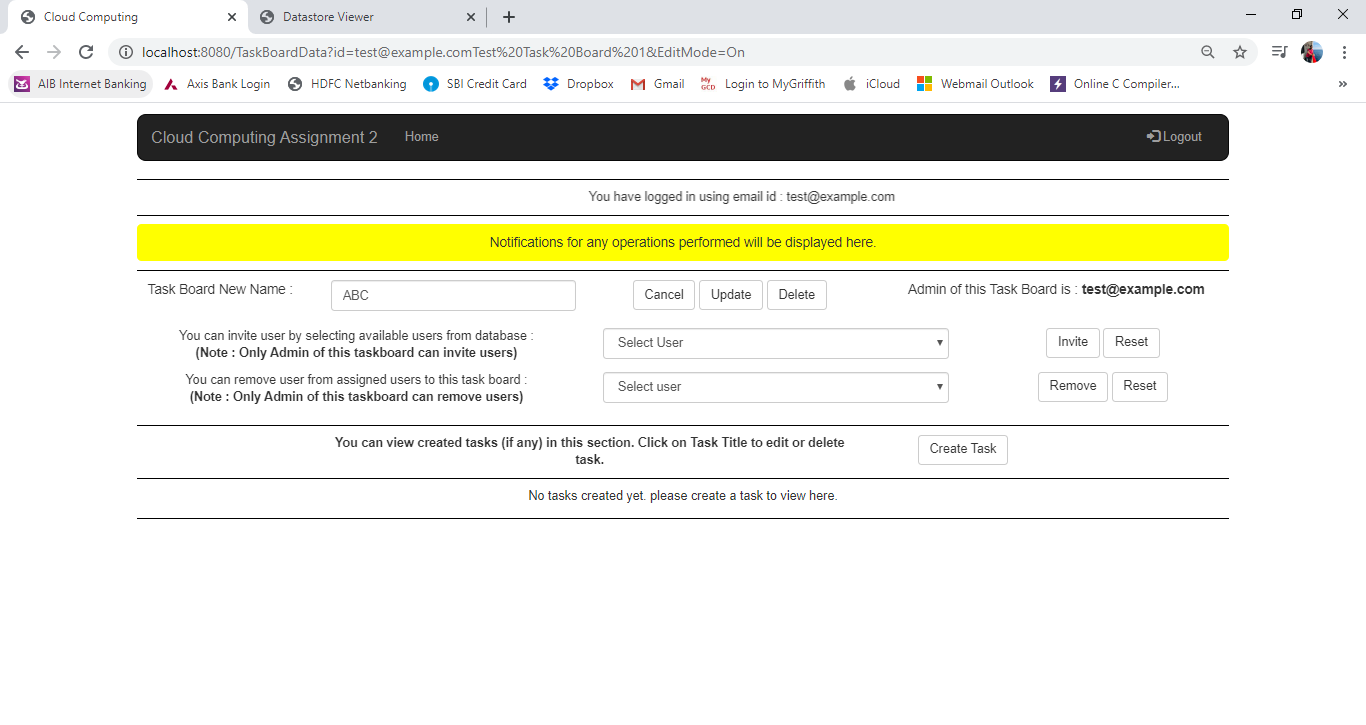


TaskBoardData.html: Image 4.1

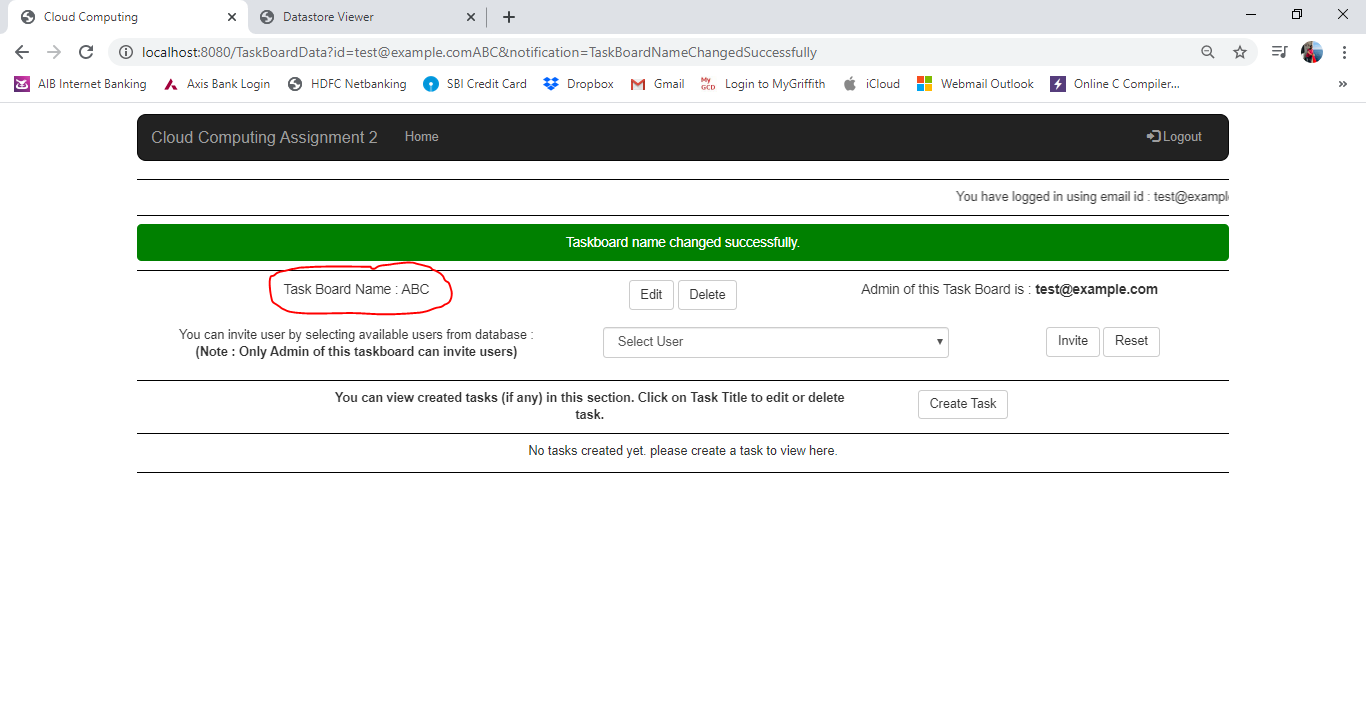


TaskBoardData.html: Image 4.2

As per Image 4.1 and 4.2, user can be removed by selecting available users from selection box. Only admins have authorization to invite or remove users from created taskboards.

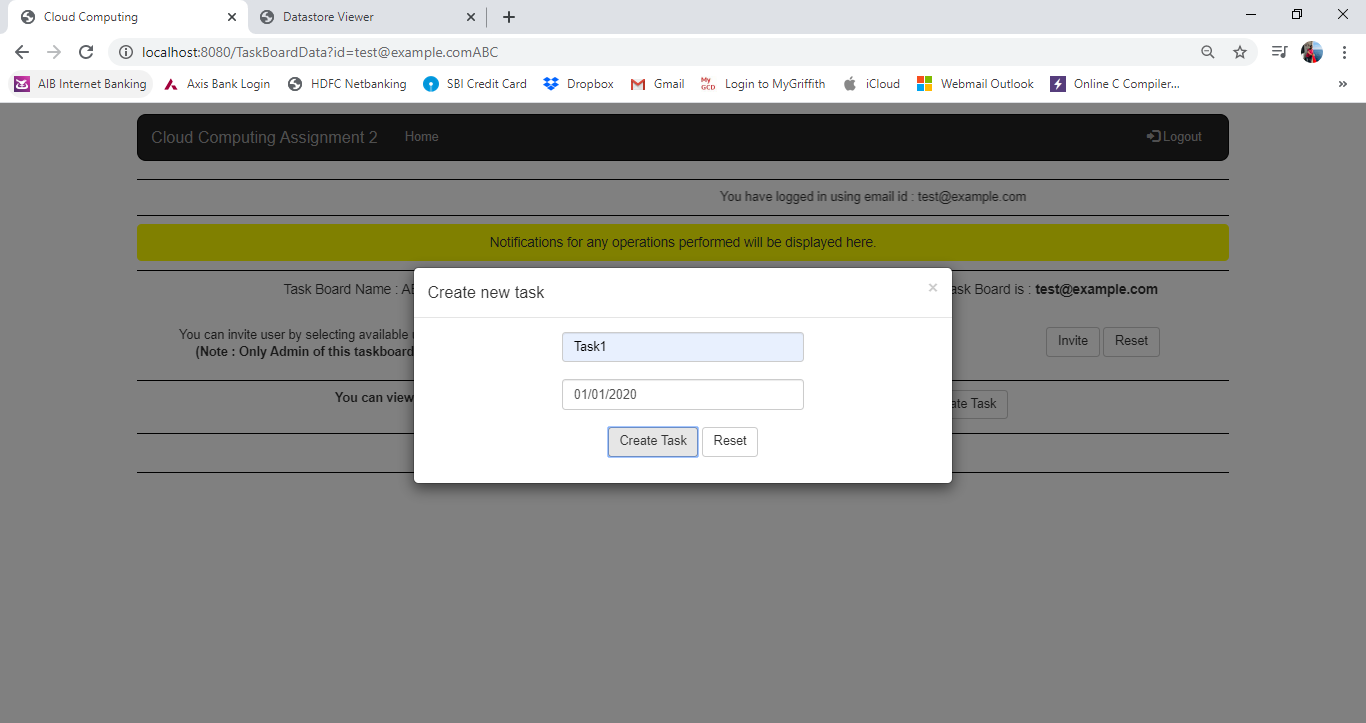


TaskBoardData.html: Image 5.1

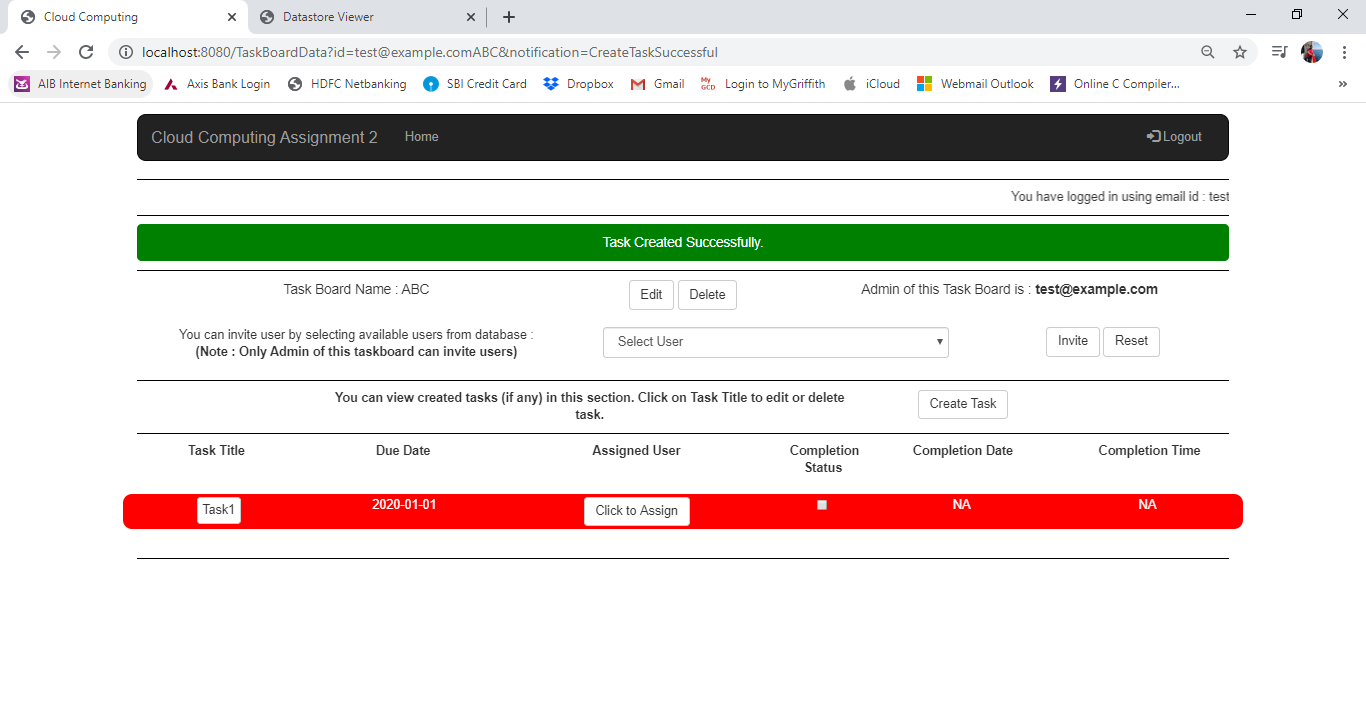


TaskBoardData.html: Image 5.2

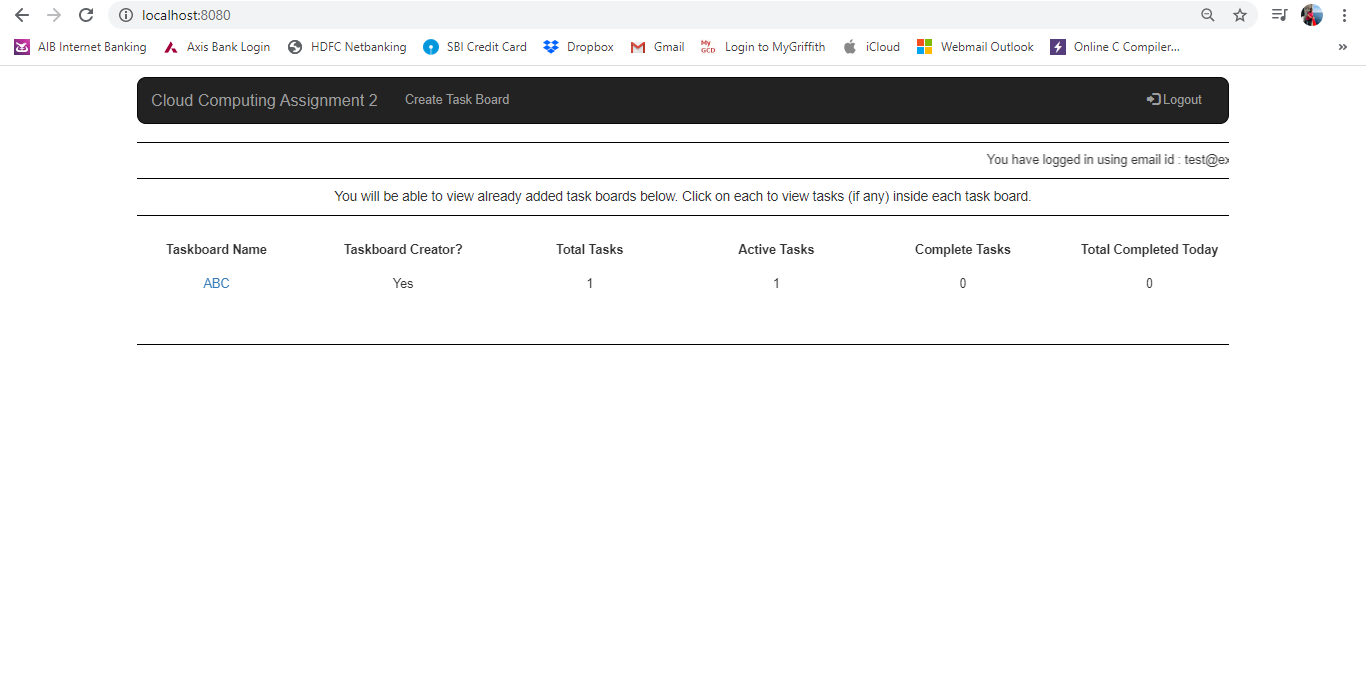
In image 5.1 and 5.2 above, it is clear that any taskboard can be renamed by entering new name of taskboard in given textbox and clicking on Update button.



TaskBoardData.html: Image 6.1

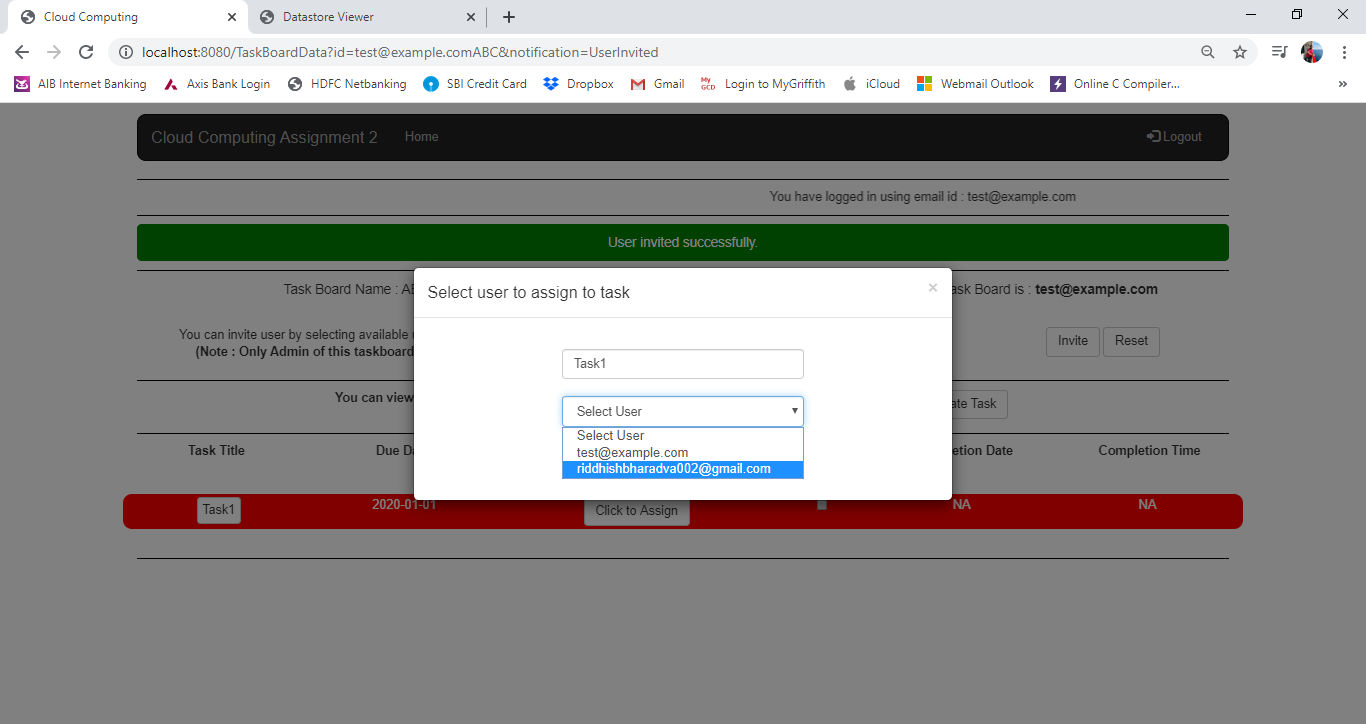


TaskBoardData.html: Image 6.2

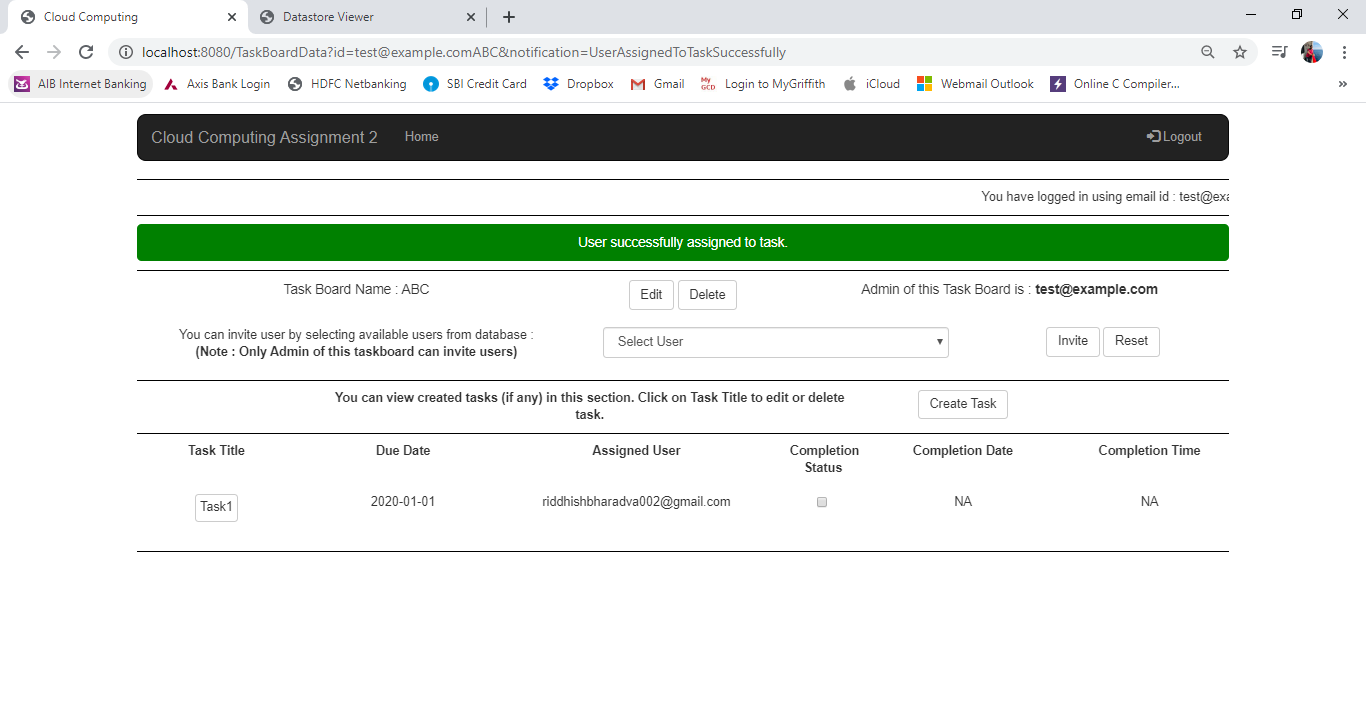


TaskBoardData.html: Image 6.3

In image 6.1, 6.2 and 6.3, I have shown how a new task is created. On clicking Create Task Button, a model will pop up and user is required to give Task name and due date as input. Post clicking Create Task Button again, a new task will be created in taskboard and the values of Total Task, Active Task will be changed on Home page as shown in Image 6.3.

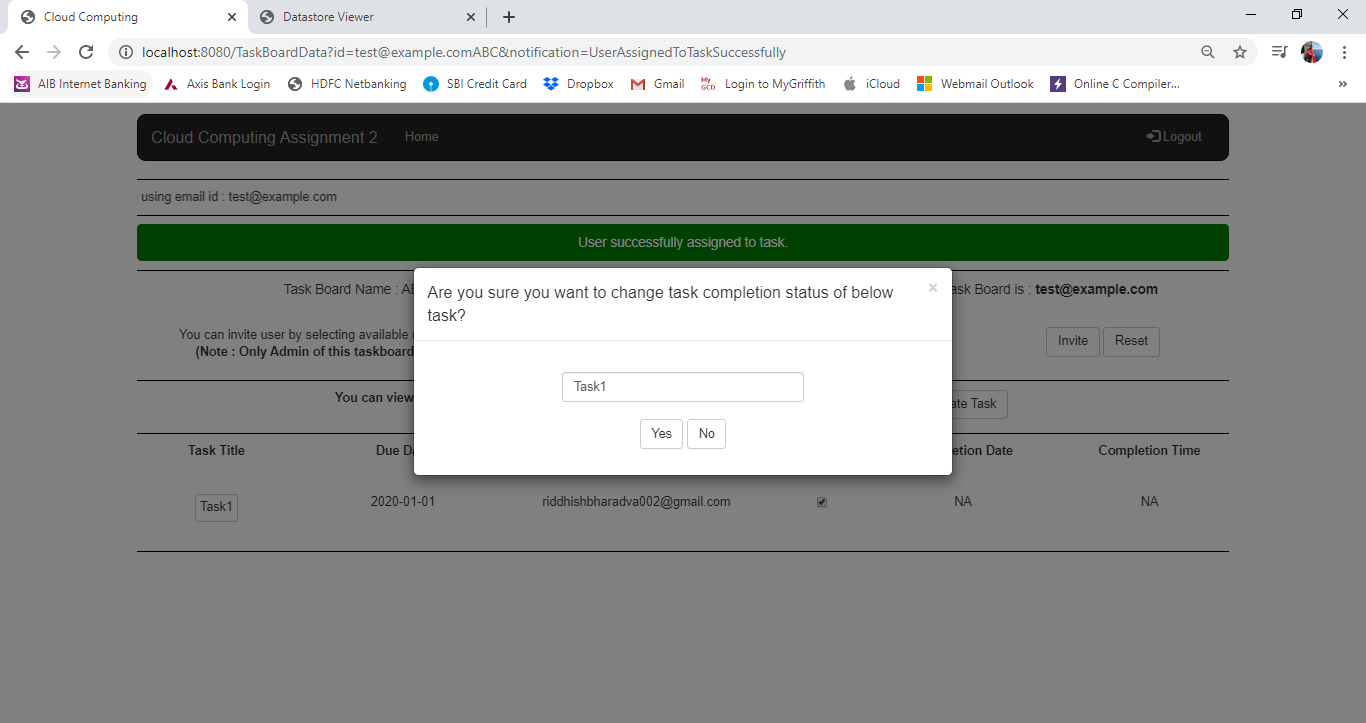


TaskBoardData.html: Image 7.1

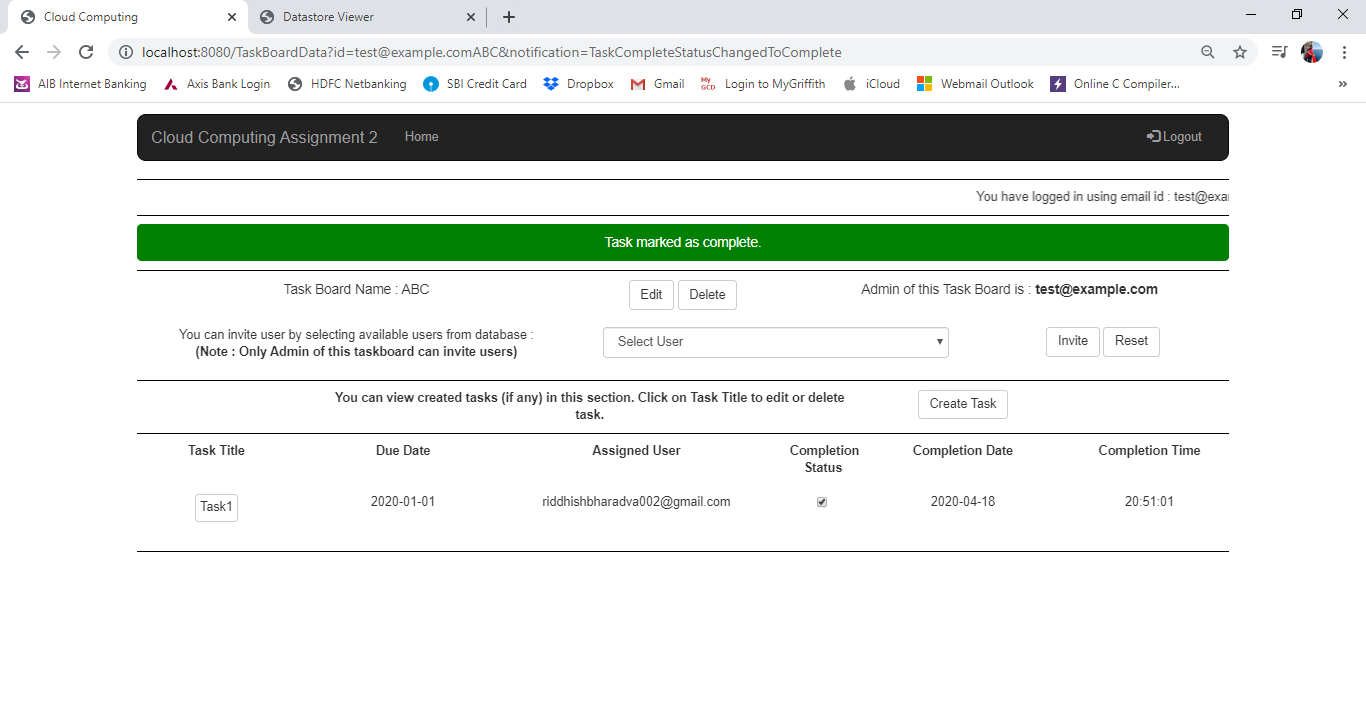


TaskBoardData.html: Image 7.2

On clicking on “Click to Assign” button, a model will pop up and user needs to select email id to whom he/she wish to assign the displayed task. On clicking Assign button, task will be assigned to selected user and same will be displayed on taskboard details page as shown in Image 7.2.

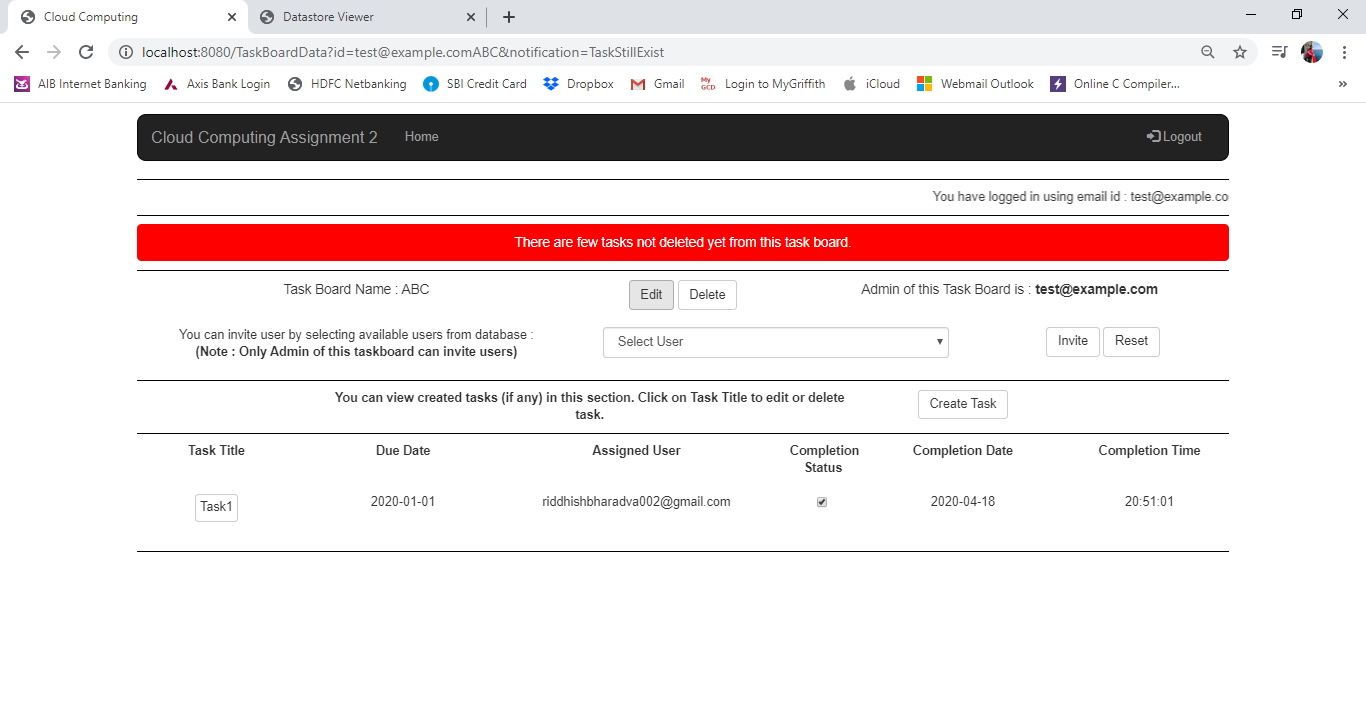


TaskBoardData.html: Image 8.1

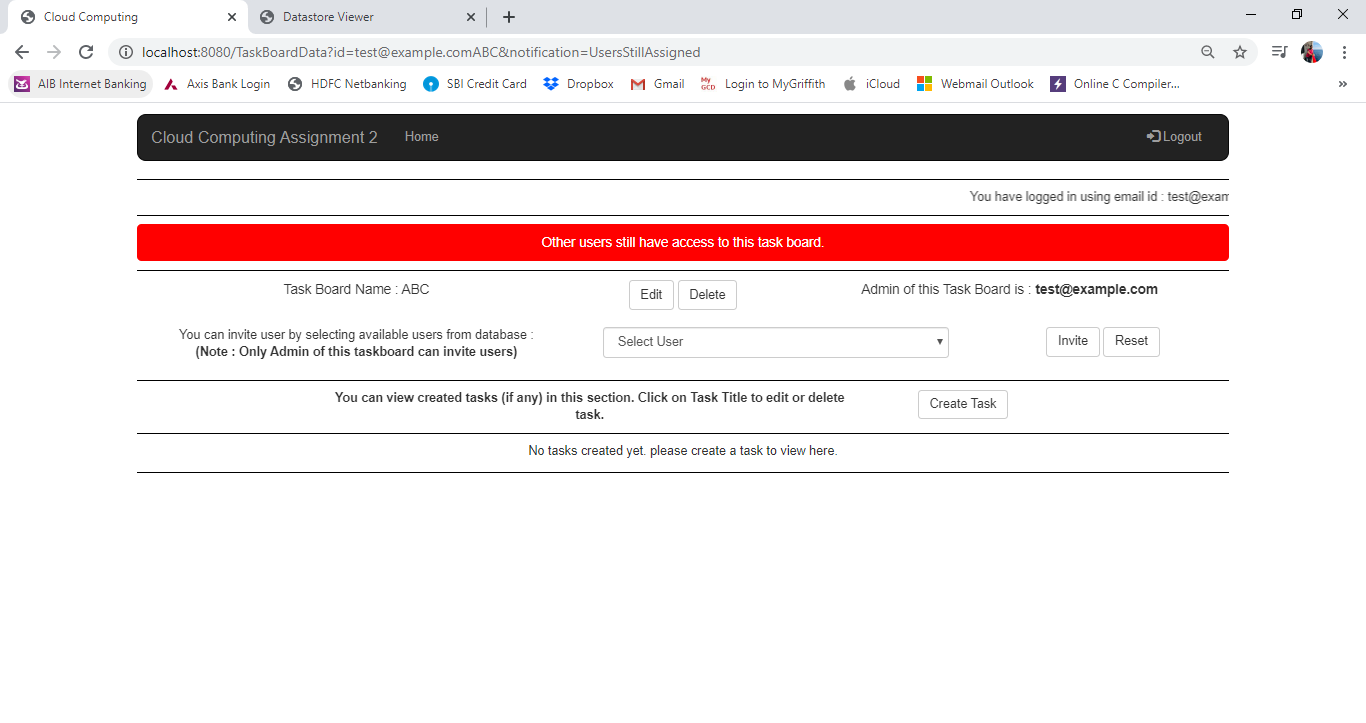


TaskBoardData.html: Image 8.2

In Image 8.1, 8.2, I have shown how the task will be complete. On clicking check box it will prompt for confirmation from user. If user clicks on Yes, task will be complete and date and time of completion will be displayed else it will not be complete.



TaskBoardData.html: Image 9.1



TaskBoardData.html: Image 9.2

As shown in Image 9.1 and 9.2, on taskboard cannot be deleted unless all the tasks are deleted and also none of other users except creator of taskboard have access to same.

**TaskBoardData.py:**

TaskBoardData.py consists of 2 functions:

1. def get(self)
2. def post(self)

Below is a brief of each functions in more detail:

1. def get(self): In this method, I have handled notification and the edit mode of page. I have declared a variable notification and EditMode. I am fetching data from url variable. In case url variable is blank (which is expected to be blank in case user is redirected from another page), then notification will be default. Same is the case for EditMode, in case of blank the edit functionalities on .html page will be hidden from user. Next, I am fetching data from TaskBoardDB and TaskDB and passing to .html page to display required data. Next, I am fetching all available users from database to send it .html page. This is to display available users to be invited to taskboard.
2. def post(self): In this method many functionalities have been handled using if conditions. Whenever a button has been clicked, request goes to post method and then it is handled using if and elif conditions. First condition handled is to invite user in taskboard. When an invite button has been clicked, first check performed is if the currently loggedin user is creator (Admin) of this taskboard or not? If he is, it is checked if the selected user already have access to this taskboard or not? Else if user is not an admin, user will be redirected to TaskBoardData.html page with appropriate value of notification passed in url. In case user is creator and If user does not have access to taskboard, user email id is granted access to same else it redirects user to main page and appropriate notification value is passed in url in both cases.

Next condition handled here is to remove user. When user clicks on remove button, first check performed is if loggedin user is creator of the taskboard or not? If found true, it will proceed to check for selected user to be removed from taskboard. If found in list, it will remove access of user from user email id and also change value in TaskDB table. In case any task is assigned to user to be removed access from, all those tasks will be marked as Not assigned before removing access of user from taskboard. User is then redirected to TaskBoardData.html page with appropriate notification value in both cases.

Next condition handled is to create a new task in taskboard. User is expected to provide task name and task due date. These values are fetched in py page and date is converted from string format to date format and check if the given task name already exist in TaskDB or not? If exist then it will redirect user to TaskBoardData.html page with appropriate notification. Else if not found, a new task entry will be created and task will be marked as Not Complete by default and no user will be assigned to same.

Next condition handled is to check task as complete / incomplete. When checkbox is checked / unchecked, this elif condition is triggered. First it is checked if task is marked as complete, it changes it to incomplete else it marks as complete. Logic here is when a new task will be created, it will be marked as incomplete. Hence, when check box is checked, this function get triggered and marks task as complete and vice versa is done and appropriate notification value is passed on TaskBoardData.html page.

Next condition handled is to assign user to task. Task title is passed on to this condition along with user email id. Task title is then checked to be found in list of title and position is saved in a variable. The passed user email id is assigned to same position in list of assigned task list.

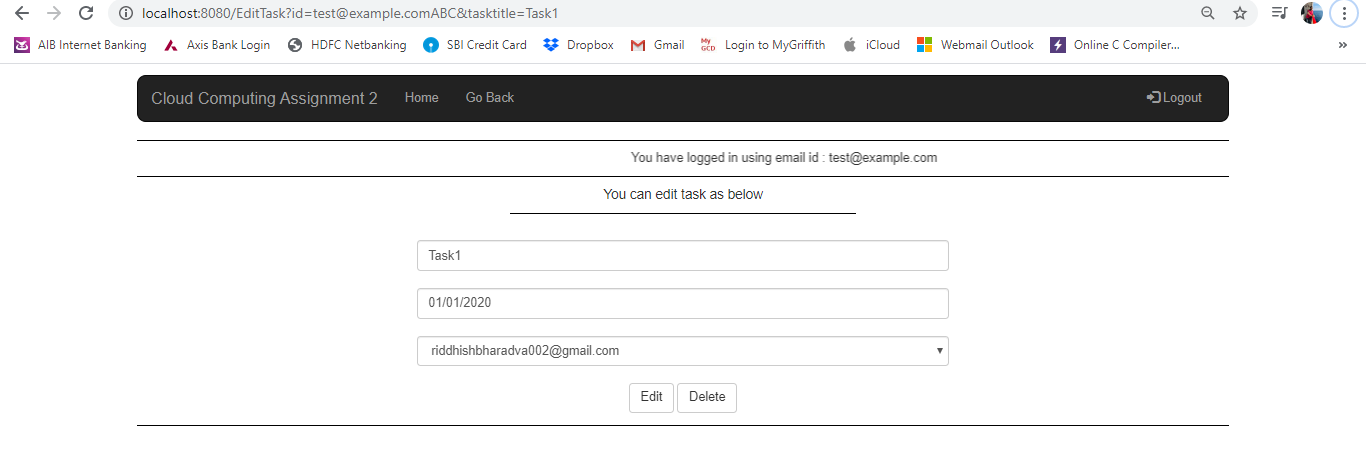
Next condition handles deletion of taskboard. Before deleting taskboard, it is checked if taskboard contains any tasks? If yes it will not be deleted else if no task is found, it will check for if any user have access to this taskboard or not? If yes, taskboard will not be deleted and user will be redirected to TaskBoardData.html page with appropriate notification in all cases. If no user have access to this taskboard, taskboard will be deleted from TaskBaordDB and also key of taskboard will be removed from UserDB and user will be redirected to Home page.

Next condition handles edit button. When edit button is clicked, value of EditMode variable is changed to “On” and same is passed in url and user is redirected to TaskBoardData.html page.

Next and final condition handled is to rename taskboard. Here, new name provided by user is checked if no taskboard for same creator must exist. If not exist, a new key will be generated with creator email id and new name of taskboard. Now, all data from old taskboard name will be copied to newly created key in both db tables TaskBoardDB and TaskDB and then values with old taskboard name will be deleted from both tables. User will be redirected to TaskBoardData.html with appropriate value of notification.

**EditTask.html**

EditTask.html have 2 functionalities related to task. First is to edit details of any selected task and delete the same.



EditTask.html: Image 1

As shown in Image above, if user clicks Edit button, given details will be updated in TaskDB and if he/she clicks on delete button, selected task will be deleted from TaskDB and user will be redirected to TaskBoardData.html page with proper notification value in both cases.

**EditTask.py**

EditTask.py page consist of logics to perform 2 operations on tasks. First is to edit the selected task and another is to delete the selected task.

In case of editing task, it checks if same task name is not already present in TaskDB. In case it is present, it will update only due date and assigned user values in TaskDB table for selected task else it will rename the selected task with new task name.

**Data structure & Models:**

I have created 3 models and 3 datastore tables in this project. The same are described as below:

Models:

1. UserDB
2. TaskBoardDB
3. TaskDB

Datastore Tables:

1. UserDB
2. TaskBoardDB
3. TaskDB

In all the data models, I have imported ndb library to connect and interact with database.

1. UserDB:

In this model, there is 1 datastore table declared i.e. UserDB. UserDB consist of Key, user\_Email and TB\_Key fields. Key is default field in all the datastore tables created in google app engine.

1. user\_Email is declared as String. This is because, email id may contain characters and numbers as well along with special characters like @,.,\_, etc.
2. TB\_Key is also declared as String. I am storing the keys of all taskboards for which particular user have access to. These values contains keys of all the taskboards user have access to irrespective of if user is creator of that taskboard or not.
3. TaskBoardDB:

In this model, there is 1 datastore table that contains all the details information about TaskBoards. It contains of fields like TBName, Admin\_Email, Users\_Email and TaskConnect. All the fields except TaskConnect are strings i.e. StringProperty.

1. TBname is used to store names of taskboards.
2. Admin\_Email is used to store email id of user who is creator of that taskboard. This field data will never be changed as the creator of taskboard will always be same even in case taskboard name is changed.
3. Users\_Email contains list of email ids of users having access to that particular taskboard. This can be changed based on access to users for that particular taskboard. There will always be 1 value in this field which will never be changed and that value will be email id of owner. Logic is to keep track of all users including creator having access to taskboard.
4. TaskConnect is declared as StructuredProperty to connect TaskBoardDB and TaskDB. This contains links to each field of TaskDB from TaskBoardDB.
5. TaskDB:

TaskDB model consists of 1 datastore table TaskDB. Fields in TaskDB are TaskTitle, TaskDueDate, TaskAssignedUser, TaskCompleteStatus, TaskCompleteDate, TaskCompleteTime.

1. TaskTitle – This field stores title of task.
2. TaskDueDate – This field stores due date of task. Due date is decided at the time of creation of the task. It is having DateTimeProperty.
3. TaskAssignedUser – This field stores email id of user to which this task is assigned.
4. TaskCompleteStatus – This field stores the completion status. This is designed to be like binary field. 0 represents task is incomplete and 1 indicates that the task is complete.
5. TaskCompleteDate – This contains date on which a task is marked as complete. I have declared as String. The reason is when task is not complete, I am storing “Not Complete” string.
6. TaskCompleteTime – This contains time at which a task is marked as complete. I have declared as String. The reason is when task is not complete, I am storing “Not Complete” string.

**Design Decisions:**

1. I have decided to keep user UI interactive and easy to understand. I have taken help of bootstrap to make the web-page user friendly.
2. I have kept key of taskboard and task datastore as taskboard creator email id plus taskboard name as this is the only unique combination in this application. No user can create 2 taskboards of same name. But, can have 2 taskboards of same name in case he/she is invited to taskboard by another user. In such case 2 taskboard names can be same with 2 different creators of taskboard.
3. I am keeping key of my taskboards and tasks same so as to keep it easy to fetch all task data when I pull my taskboard data with same key.
4. I am using structured key to link taskboard datastore with my task datastore.
5. While checking task to complete / incomplete, I have written a script to pass task title on model and have used script to not to allow user change value in textbox displayed while completing / incompleting task for which checkbox has been selected / deselected. This not only makes it easy to pass value, but also prevents from taking user on another page before passing value to python script.
6. I am also keeping values of key of taskboard created / having access to taskboard in user datastore.
7. I am also keeping email ids of users having access to taskboard in taskboard datastore for further use.
8. I am keeping datatype of 2 fields (TaskCompleteDate, TaskCompleteTime) as StringProperty instead of DateTimeProperty as I will be storing “Not Complete” when task is not complete else, I will be storing date and time of completion.