Team Name: The Four
Team Members: [Sergio Nogueira Jr. - https://github.com/seno7509] / [Askar Alaskar - https://github.com/moal2839] / [Mohammad Ghiasy - https://github.com/paimang**]**
Project Name: Planneradora
Team Number: 006
Section: 014
GitHub Repository: https://github.com/CSCI-3308-CU-Boulder/3308SP21_section014_6/

Project Milestone 4

- Revised List of Features:

  -Tasks Feature:
  This feature includes the addition and deletion of tasks for users using the firebase database. Tasks themselves hold the information about the name of the task, date, difficulty, and description of the task. This feature also includes the checking off of tasks which means that when a user finishes a task they are able to click a button to mark that task as completed. View the tasks and the details about them as well as view a history of the tasks that the user has completed.

  -Login/Registration Feature:
  This feature includes the registration of a new user if they don't already exists in The firebase database and logging in if the user already has an account. This feature also includes the ability to log out and choose to login using a different account.

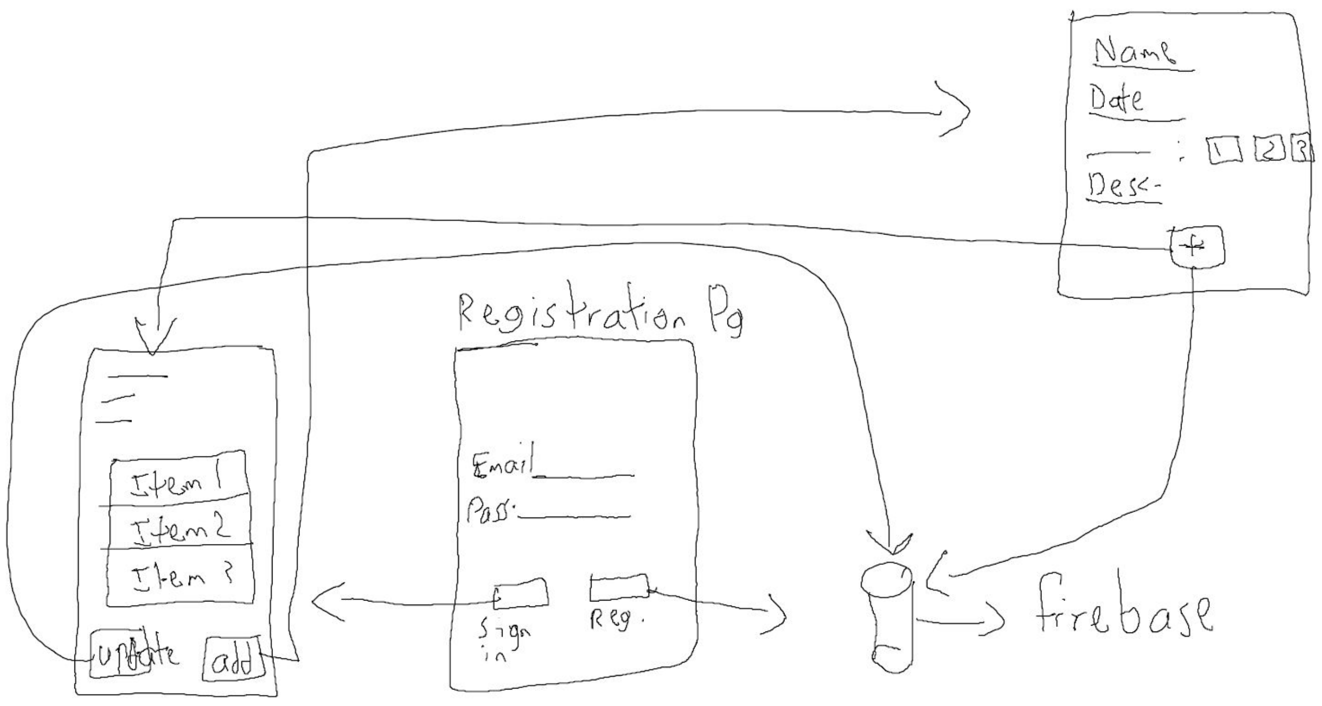
  -Modification Feature:
  This feature allows the user to change attributes of a task such as changing when a task is due, the task name, and description.

  -Progression and Level System Feature:
  This feature includes the user earning experience points when finishing tasks and those points go towards leveling up the user's account. After certain progression of levels the user can earn badge icons that can be equipped and viewed on the home page.

-Family System Feature:

This feature allows parent user's to be able to view and monitor what tasks their kids have finished.

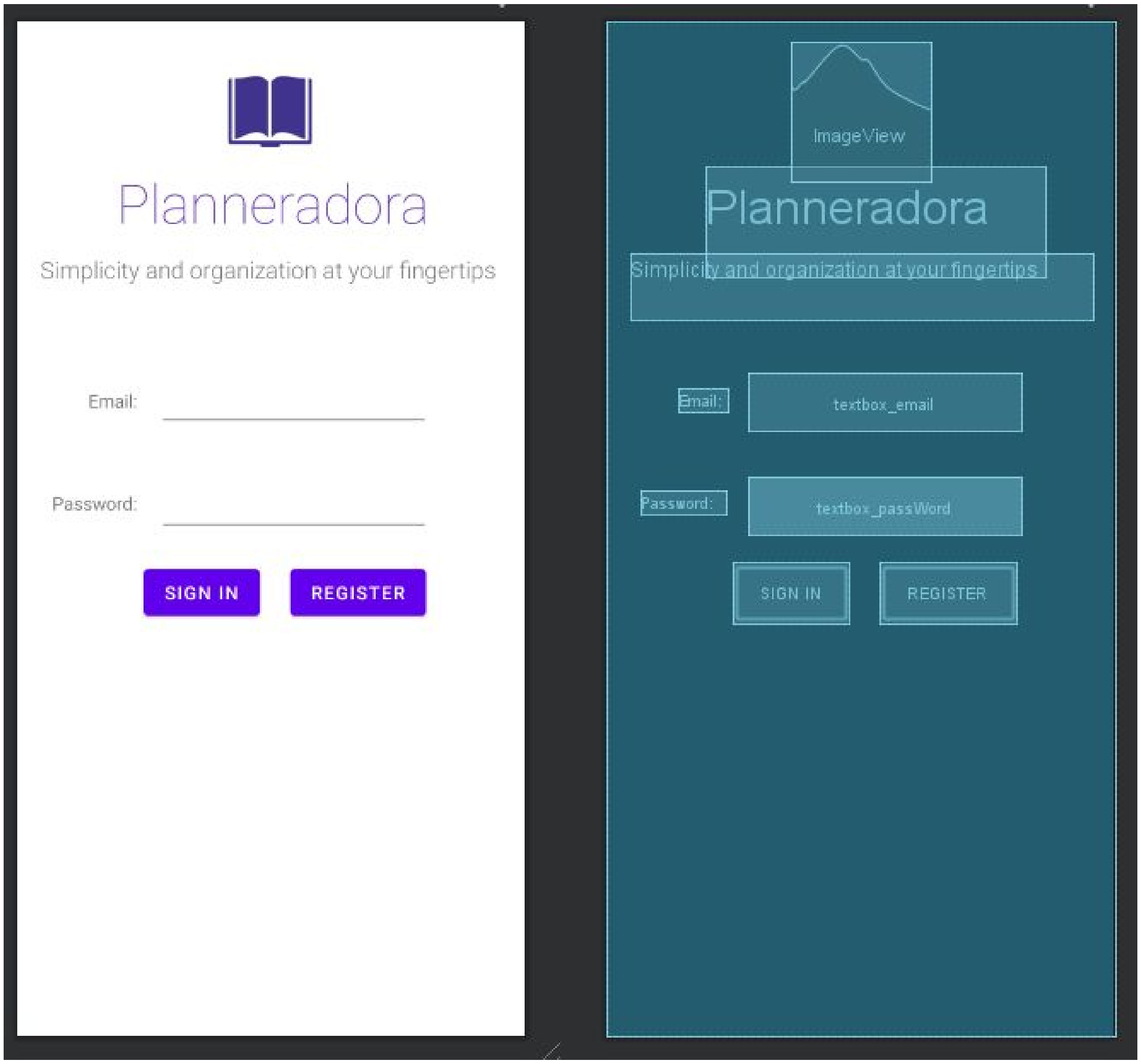
- Architecture Diagram:



A note in android studio these pages are made with xml script. In xml you can add buttons, input and anything you see on the screen much like html and change their attributes with the script. When adding a new xml page like the home page, login/registration page, and add task page you can link that xml to a java file where you will get the id of the components of the xml objects like the button and input and setup the logic when the user interacts with those components also from the java backend you can get an instance of the realtime database of firebase and use it to add or remove information from the database like when the user clicks on the registration button on the registration page in java an instance of firebase is created and you can setup a sign in user functionality which will add the user to the database similarly the sign in button will go to the database and check if the entered email and password exist and match what is on the database.

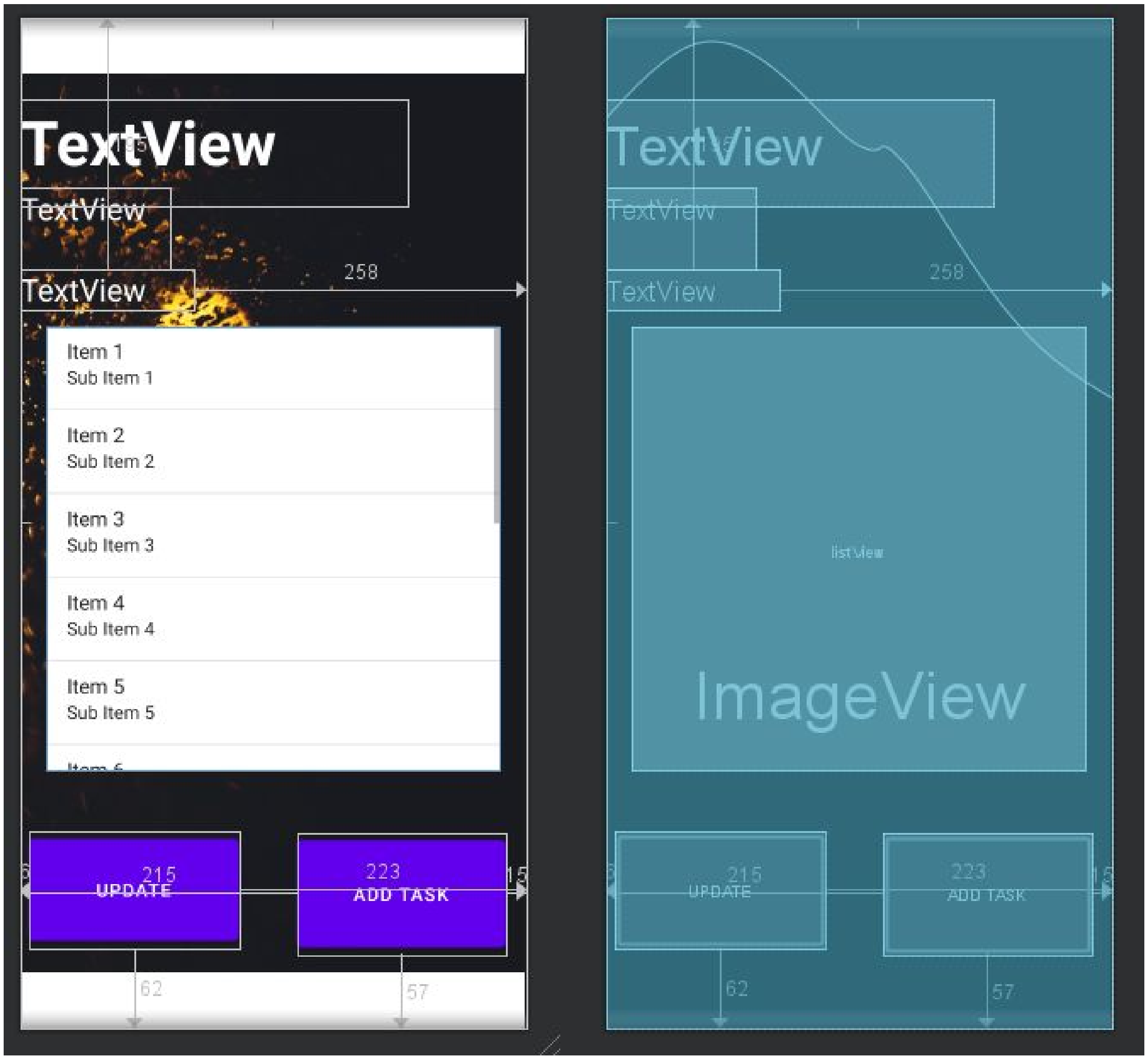
- Front End Design:

  -Login/Registration Page:

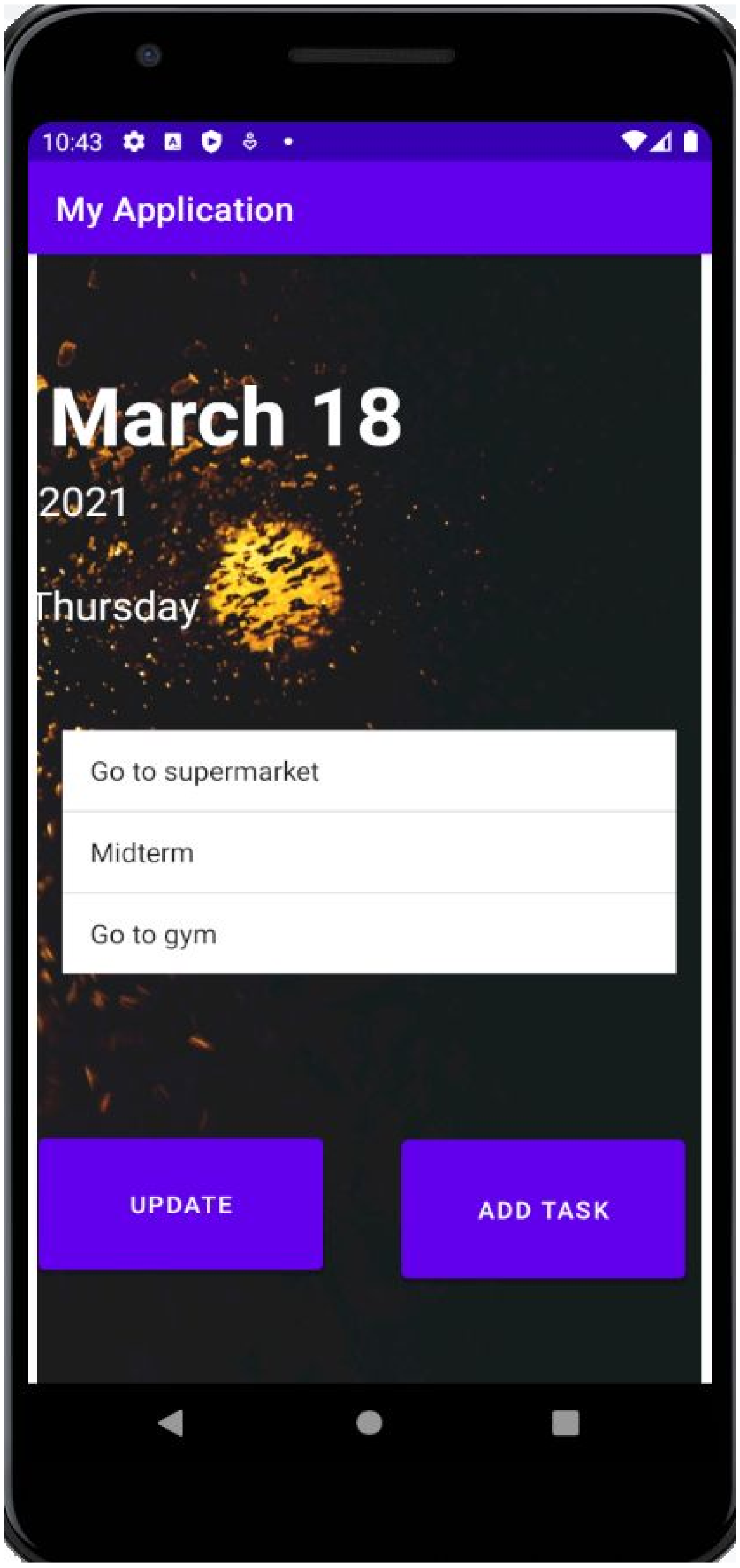


This is page contains two edit text fields which in android studio is user input one gets the email and the other gets the password. If it is a new user then they will click on the register button which will verify that their email is valid and add them to the database. If they have been added then they can click on the sign in button and it will sign them in to the application where then they will land on the home page.

-Home Page:



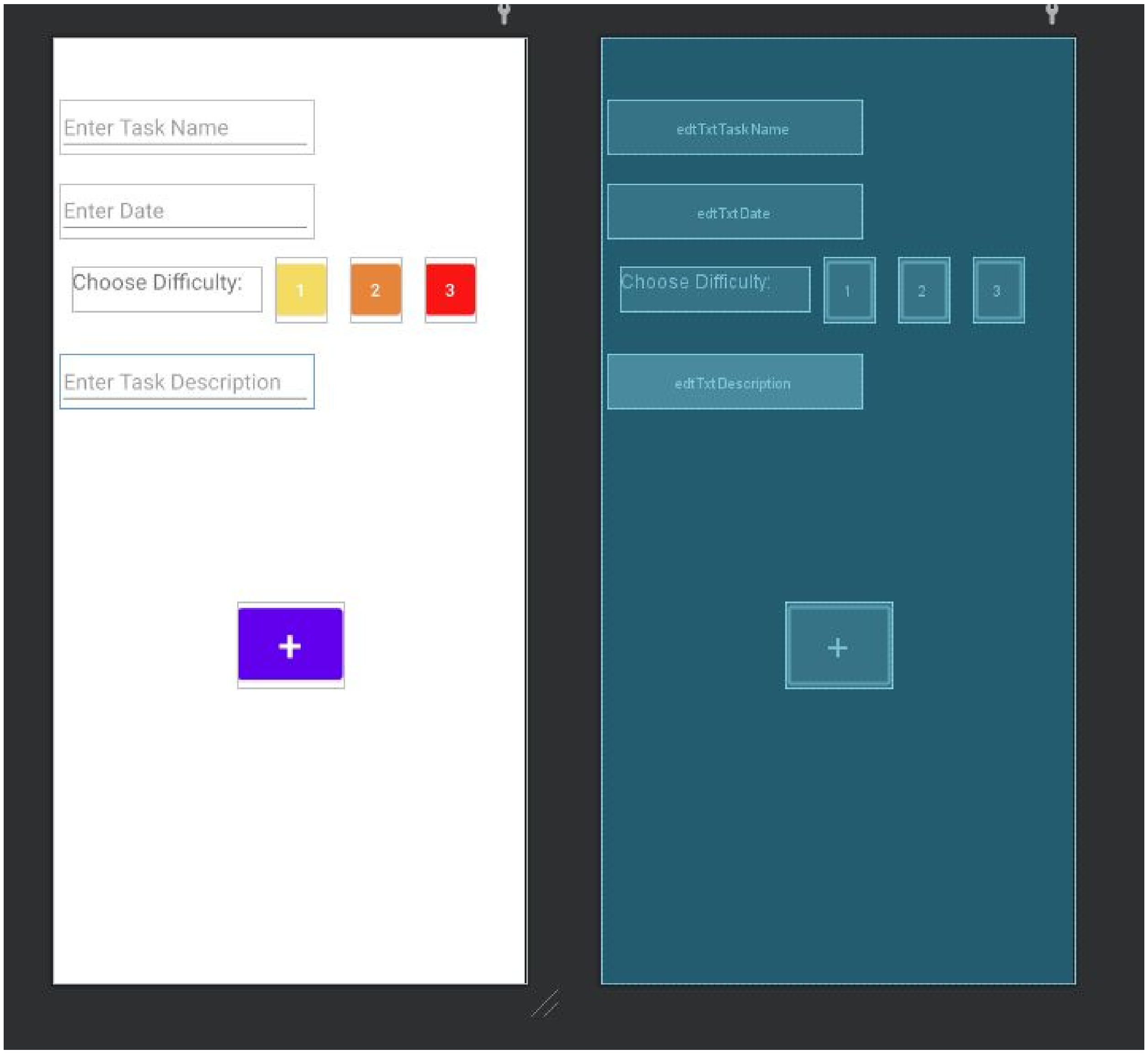
A view on how it looks when the application is running:



The three textviews on the top get the date of today using functionality from android studio when the java on the backend is being run. There are two buttons in which the user

Can interact with the add task button once clicked transitions the user to the add task page where they can enter information about the task and add it. The update button updates the listview on the home page and add any existing tasks as cards with a text that displays the name of the task, if the user has no tasks then it will show an alert message at the bottom of the screen saying that no new tasks have been added. Once a task has been added and is viewable on the home page then the user can hold them to delete the task and it will be removed from the listview.

-Add Task Page:



In this page the user can enter the name, description and date of the task as string input and then select one of the three difficulties by pressing on one of the buttons, The button with the plus sign at the bottom of the page should then add the task with all its information to the database and then go back to the home page.

- Web Service Design:

  -The only api that is being used in the application is google's api which gets and verifies that the email passed in when registering for a new account is a valid email and then adds it to the firebase database.

- Database Design:

  In this application we are using Firebase database and we are only storing to types of data The first is users and the second is tasks. A user is related to tasks in that a user can have many tasks but a task is only connected to one user. The user holds the attributes, email (string), password(string), id(integer), level(integer), xp(integer), and an array of tasks. The task holds the attributes name (string), date (string), difficulty (integer), description (string).

- Challenges:

  -The adding and deletion of tasks was a challenge at the beginning because of the unfamiliarity with android studio and java. We had trouble making this functionality work because everytime we added the information about a task when we go back to the home page and run the java associated with that page it would think that we are starting the activity again as if it is the first time we enter the app and not going back to it so it we would end up losing the information that we were storing about the task and the listview wouldn't update in the home page. This issue was resolved by marking the add task page xml as a child of the home page xml and making some variables that were used to hold information about the tasks and update the view in the home page java to be publicly accessible by the other activities java after making those changes this issue was resolved and the view would update after adding a new task.

  -There was a challenge of figuring out how to make the database work and understanding it. That issue was resolved and we are working with Firebase and the information about users regarding to logging in and registering is working with the database. The user is able to login/register and this feature is working as it was intended.

-There were some few bugs that came up after fixing the issue with the home page not getting updated with the new tasks that were being added. Some of those bugs were that when the user clicks on the update button it would sometime add duplicate tasks to the view, or override previous tasks, and there was also the bug were when the user deletes a task and then the user clicks on the update button again it will still update the view with the deleted tasks. This issue was resolved by fixing some logic in the backend with java.

- Individual Contributions:

  -JIRA: https://csci-3308-spring21-6.atlassian.net/jira/software/projects/THEF/boards/1

  Askar Alaskar: Fixed the bugs with the tasks where it would add duplicate tasks to the view when the update button was clicked, and fixed the bug where some tasks were being overwritten and fixed the bug where previously removed tasks would still show up when the user would click on the update button.

  Payman Ghiasy: Created a date picker html, for picking date when clicked it will show up a xml with dates like calendar, trying to figure out how to make the task list to assign to specific user.

  Sergio Nogueira: Re-initiated Firebase user db, created independent (non-API) queries for user registration and sign-in (insertion and selection), implemented front-end graphical optimizations, removal of progressbar, more detailed catch functions for debugging user error. Implemented application logo, name, and description. Re-factored username input to email input instead.