Project Report

Hazir

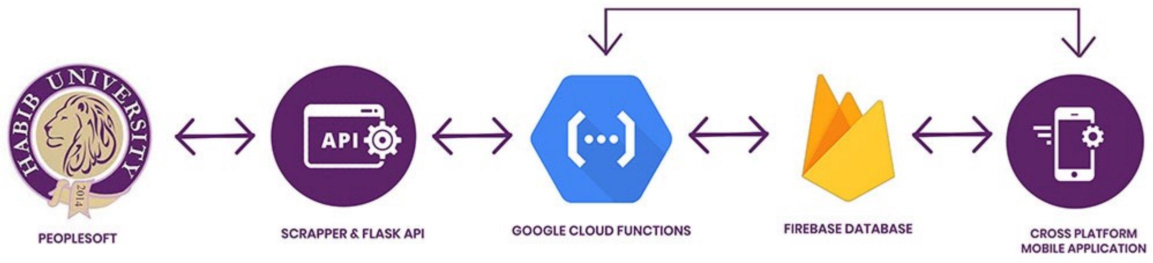
S. Hasan Faaz Abidi

Overview

The idea of Hazir is inspired to solve, perhaps, the most basic yet crucial problem in regard to student's performance at Habib which is the issue of attendance. It's serious than we can think or comprehend. Considering all this, we came up with an idea to create an attendance app that collects your data from PSCS and notifies the user about their current attendance status. It will also keep a track of your absences and will tell you if you are on the verge of dropping from the course or not. It's a simple idea but it can have a great impact. Students don't have to check their attendance by logging in on PSCS and click on several pages to get at the actual attendance roster. They don't even have to open the app daily; app will generate a notification if the information updates. On the other side, instructors and RO will have fewer requests and complains from the students, unlike now.

Architecture

This how all the webservices and mobile application are working together to create a seamless flow of data.

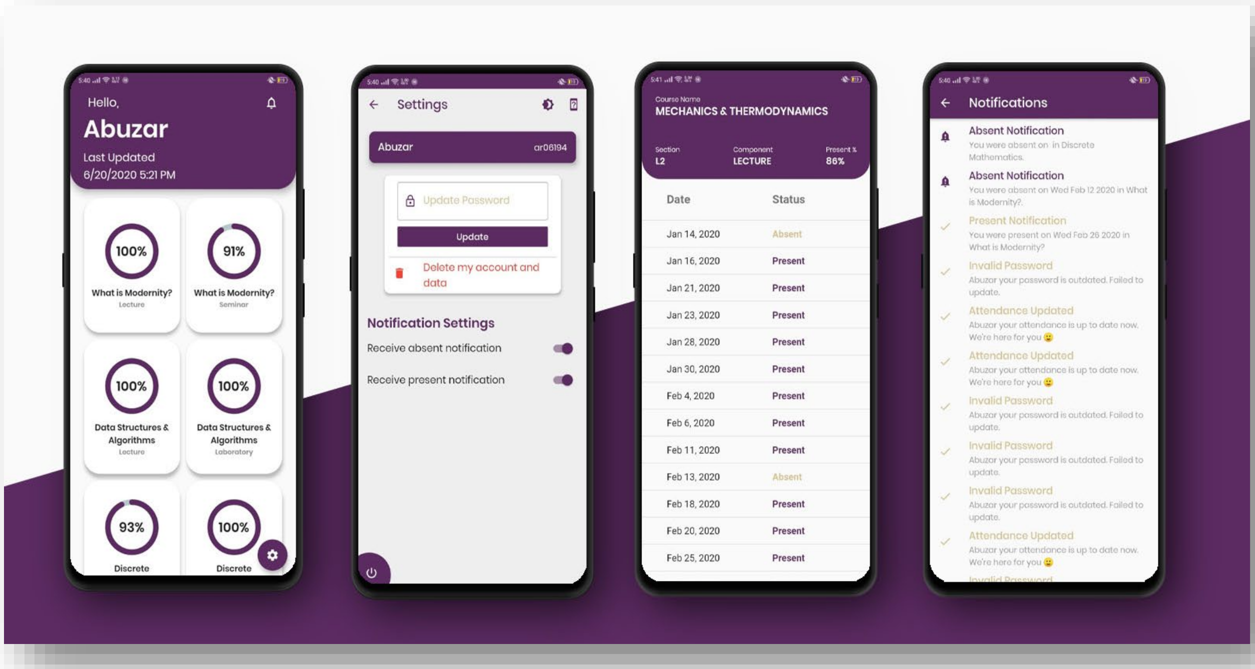


Implementation of DSA techniques:

- We used DFS for the insertion of values in our JSON tree from our Flask API
- We are using JSON Tree in hierarchy throughout all the services.
- We have used nested dictionaries and we did their manipulations.
- We have used and we did list manipulation

Interface/Output/Results

Below are the few mockups of our mobile application.



Hello,

Abuzar

Last Updated
6/20/2020 5:21 PM

100% What is Modernity? Lecture

91% What is Modernity? Seminar

100% Data Structures & Algorithms Lecture

100% Data Structures & Algorithms Laboratory

93% Discrete

100% Discrete

Settings

Abuzar ar06194

Update Password

Update

Delete my account and data

Notification Settings

Receive absent notification

Receive present notification

Course Name
MECHANICS & THERMODYNAMICS

Section	Component	Present %
L2	LECTURE	86%

Date	Status
Jan 14, 2020	Absent
Jan 16, 2020	Present
Jan 21, 2020	Present
Jan 23, 2020	Present
Jan 28, 2020	Present
Jan 30, 2020	Present
Feb 4, 2020	Present
Feb 6, 2020	Present
Feb 11, 2020	Present
Feb 13, 2020	Absent
Feb 18, 2020	Present
Feb 20, 2020	Present
Feb 25, 2020	Present

Notifications

Absent Notification
You were absent on in Discrete Mathematics.

Absent Notification
You were absent on Wed Feb 12 2020 in What is Modernity?.

Present Notification
You were present on Wed Feb 26 2020 in What is Modernity?

Invalid Password
Abuzar your password is outdated. Failed to update.

Attendance Updated
Abuzar your attendance is up to date now. We're here for you 🙂

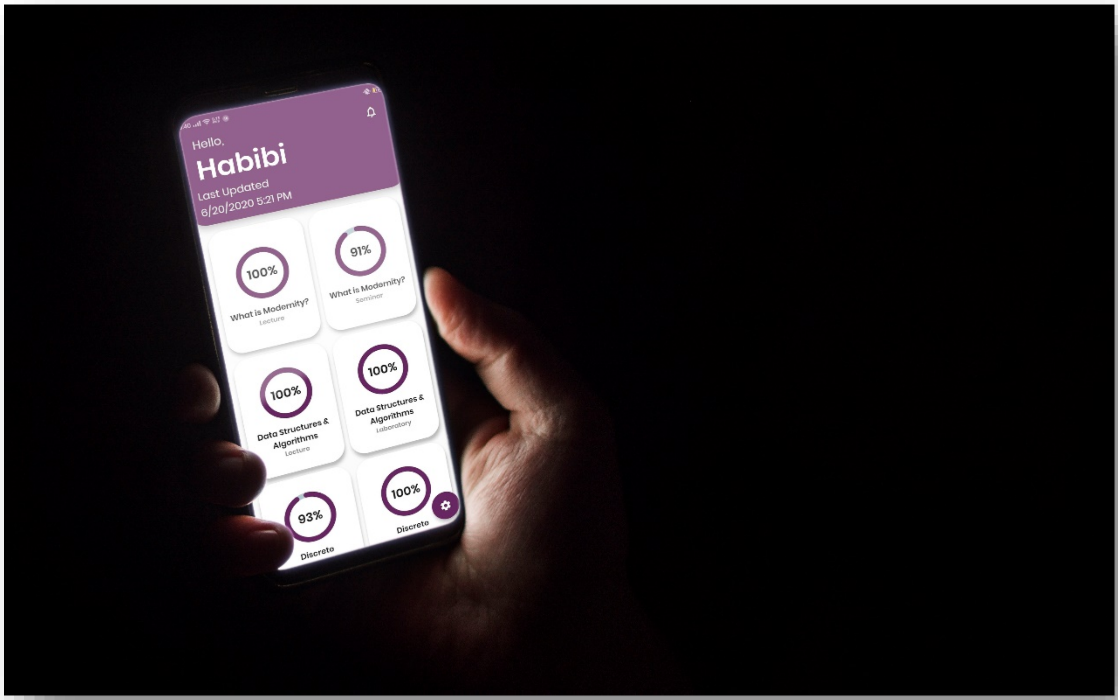
Attendance Updated
Abuzar your attendance is up to date now. We're here for you 🙂

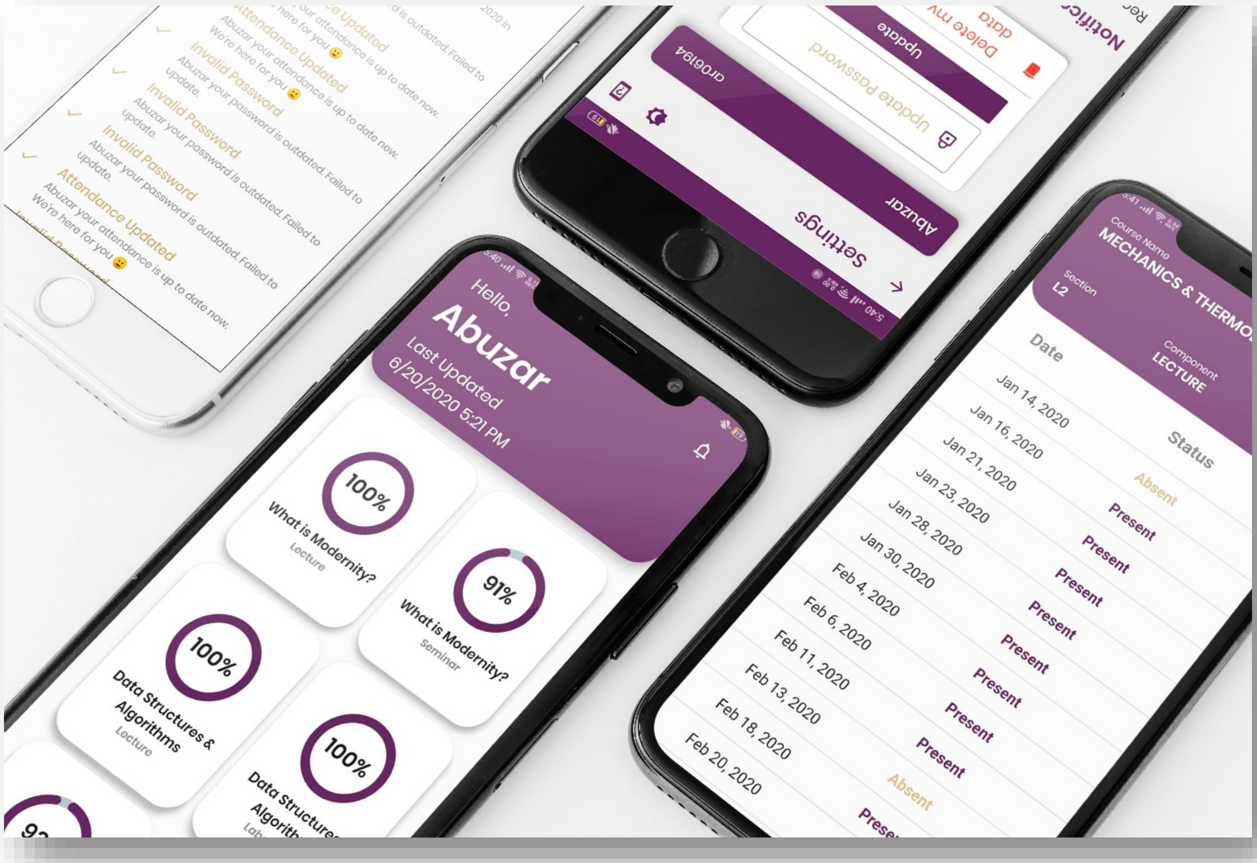
Invalid Password
Abuzar your password is outdated. Failed to update.

Invalid Password
Abuzar your password is outdated. Failed to update.

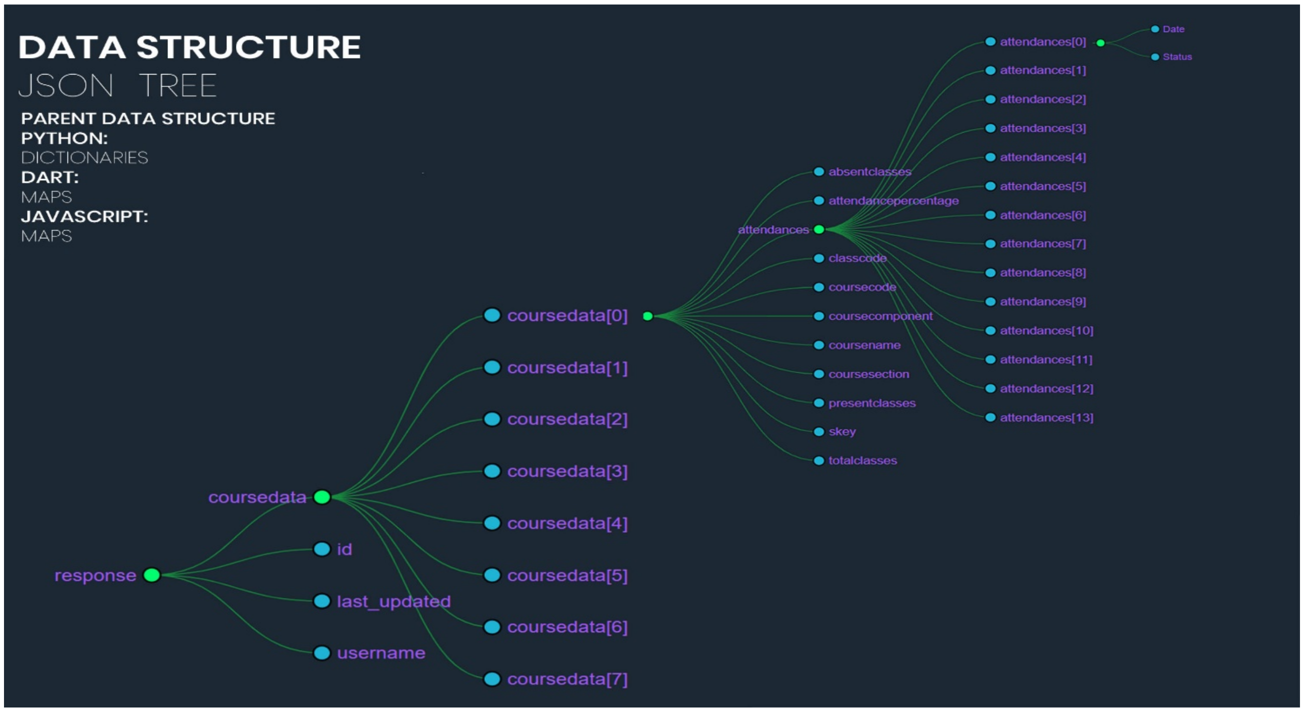
Invalid Password
Abuzar your password is outdated. Failed to update.

Attendance Updated
Abuzar your attendance is up to date now. We're here for you 🙂





API Response:



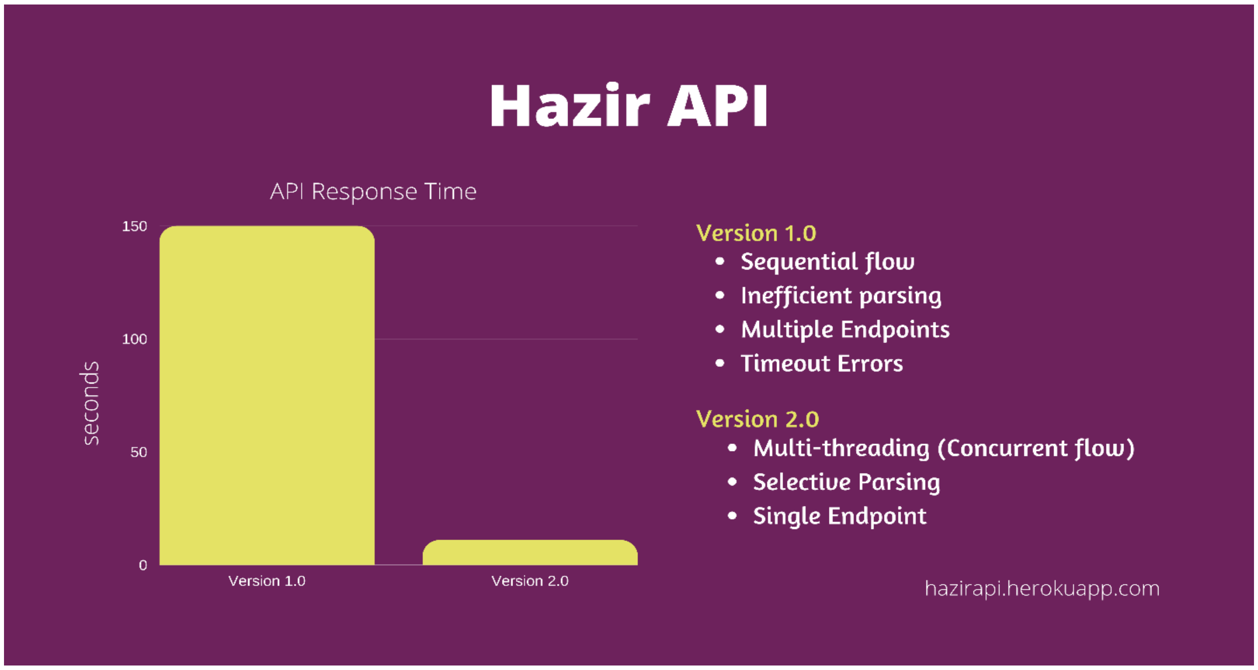
Challenges Addressed

Although almost every aspect of this project was a challenge for us. We knew nothing about how to design a proper architecture for creating such service. This is the reason why we mostly did try and error. We tried many approaches and then changed our path. But I want to highlight the most important and significant challenge that we faced and solved.

The most important challenge that we addressed was the optimization of our API. Without this API, this entire would not be possible.

Initially we created an API with my basic scraper that was taking data from all the courses. The problem that we were facing with that API i.e version 1.0 was that it was taking too long to respond. The response time for the version 1.0 was about 150 seconds i.e 2.5 minutes. Due to this we were facing even more issues in our architecture. We were unable to deploy this API to any of the webservices because no service was allowing us have timeout time of more than 150 seconds. To solve this timeout error, we tried making segments of our data. We created multiple endpoints to avoid timeout errors. But this approach was not efficient. It got more complex and harder to manage.

After that, I came across the concept of multi-threading. Multi-threading was the best fit to our API because it is mostly used for this specific purpose of sending multiple requests to web. So, we implemented multi-threading in our scraper. And it was a game changer. Our response time reduces significantly. Response time went from 150 seconds to around 30 seconds. This optimization was good but we needed to reduce our response time even more. So, we also added selective parsing in our scraper. In version 1.0 we were not selecting tags and then parsing it to beautiful soup, but when we did this and throw away all the garbage data, our response time got even lower. After this, our response time was/is on average 12 seconds. In 12 seconds our API will get all the users data from PSCS servers and structure that in proper JSON tree.



Completeness

The project we presented in demo was complete. All the outcomes from our project proposals have been met. This is a complete project according to our project proposal.