# Approach（30 min）

The tasked described in the file could be concluded as a Single Page Application(SPA), whose main function is working as a domain-specified search engine. The naivest approach would be designing several web forms collecting all the required and optional search criteria from user, and a “search” button to gather all the information to build an HTTP request to the server. Results should be presented in a form, which is disabled before response if obtained from the server, where each column’s header should be a button which would sort the corresponding column when clicked(first click ascending, second click descending and restore at third click). Ajax shall be a good tool for collecting data and building HTTP request. Bootstrap is a great framework for responsive component design and React would be a good fit for the state changes and action the application should take.

As the file states that the result “meet user-specified filter criteria” and given the fact that the web server only provides APIs of searching by name and by name and country. And that should be the only filters the application provides. Otherwise the front end would be doing the job which should be done by back end.

Also we need error handling in case the server is down or response is a empty list so we have nothing to show.

Several questions pop into my head during designing. The first thing would be that the page should be responsive that it can fit both portable end and work station. And then it is required to provide Sorting function. So what exactly sorting algorithm required for each field? Should it be just alphabetically since most of the columns are string? Or shall I provide specific sorting approach for each column? For example, we can sort “country” by continent and generally sort the result by ranking. Can sorting be applied to multiple field of the result? If so, how should we judge the priority? And what if certain value is missing like the situation below?

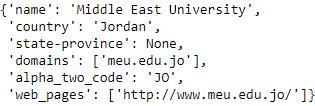


Figure 1: state-province field is missing

Then I will take some details into consideration. Like how many results should be displayed in one page. Shall the result be divided into different pages or just design “show full content” button? And shall we provide the reset button so the user can reset their search filter criteria and sorting requirement?

At last, we should use some validation process to reduce the server side pressure: invalid search patterns should be add and checked before the request is sent to the server. But it would be a problem to set such a pattern without prior knowledge.

After the development is finished, we should be testing the front end before releasing it or merging it to another project. Selenium would be good choice to do such a job.

# High Level Design(25 min)

A close up of a logo

Description automatically generated

# Update API(20 min)

We can design a button, located in the corner, to perform this function. A pop-up including forms can be shown to the user by adding a listener to the button. Each form is responsible for one field of the updating data. Submit and cancel button should be displayed at the bottom, where cancel button should close the pop-up.

Submit button should do several things. First it should collect the information user provided in the forms. Second it should do the mandatory check to make sure the data to be uploaded to the database is consistent, where the checking rules can be obtained from the documentation of the API. Then, it should build a post request and send it to the server. And last it should check the status returned from the server and tell the user if the data is successfully added to the server.

As the server would be providing consistent result to everyone. We want to make sure the data added by user is valid and correct. So it’s a question that if every user could have this right to add data to the server.

User authentication should be involved here. Each user can have a certain amount of opportunities to add new record to the server. Every successful adding grants him/her a new chance and failed ones not. What’s more, to prevent malicious DDOS attack, submit button should be disabled until the server returns the adding result.

# Work Before Release(10min)

Unit and integration test should be applied before release to make sure the application is functional and consistent. Performance test then should be applied to make sure the website

Static resources and dynamic ones should be separated. Static resources should be deployed in CDN.

Version control should be well-managed.