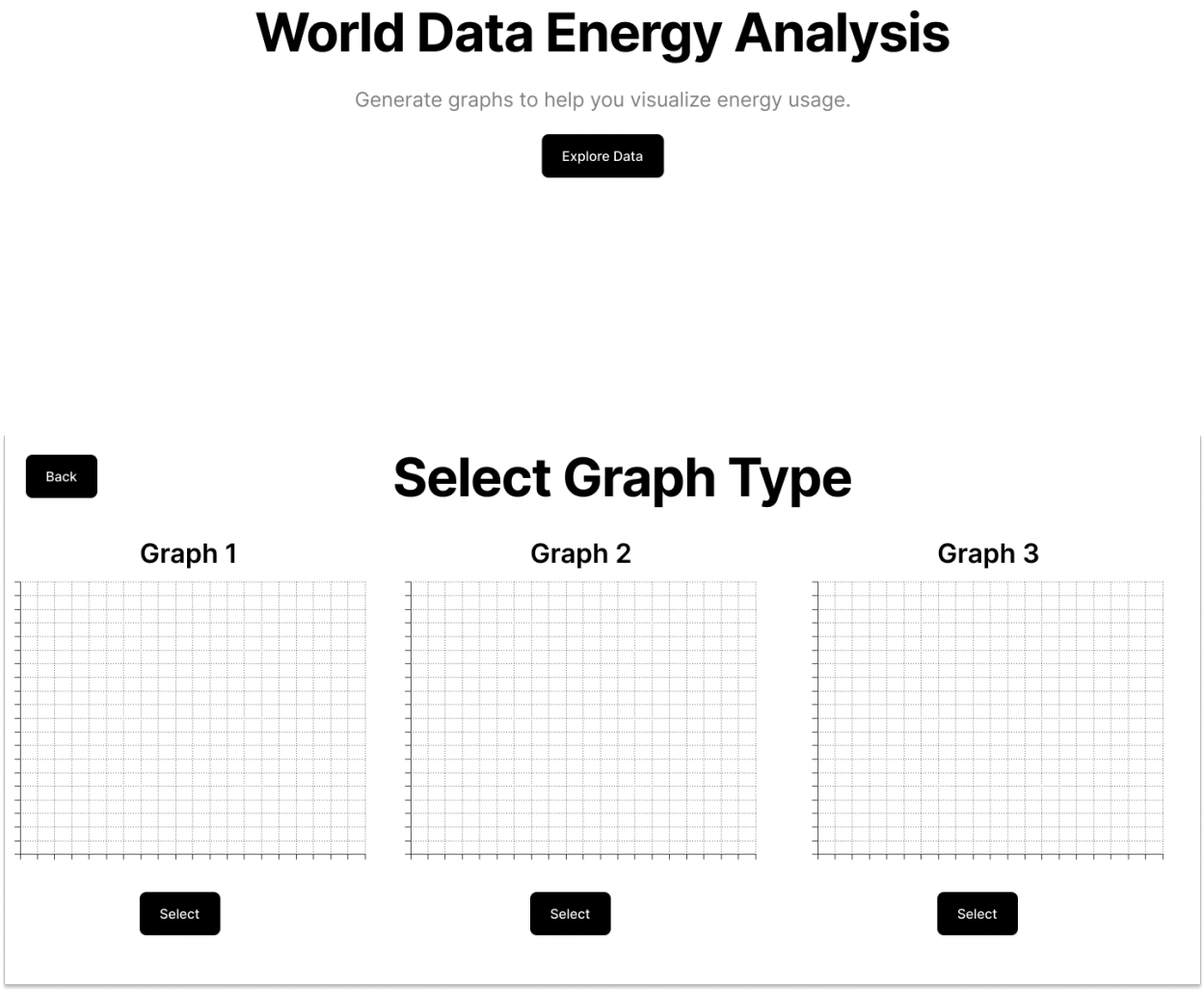
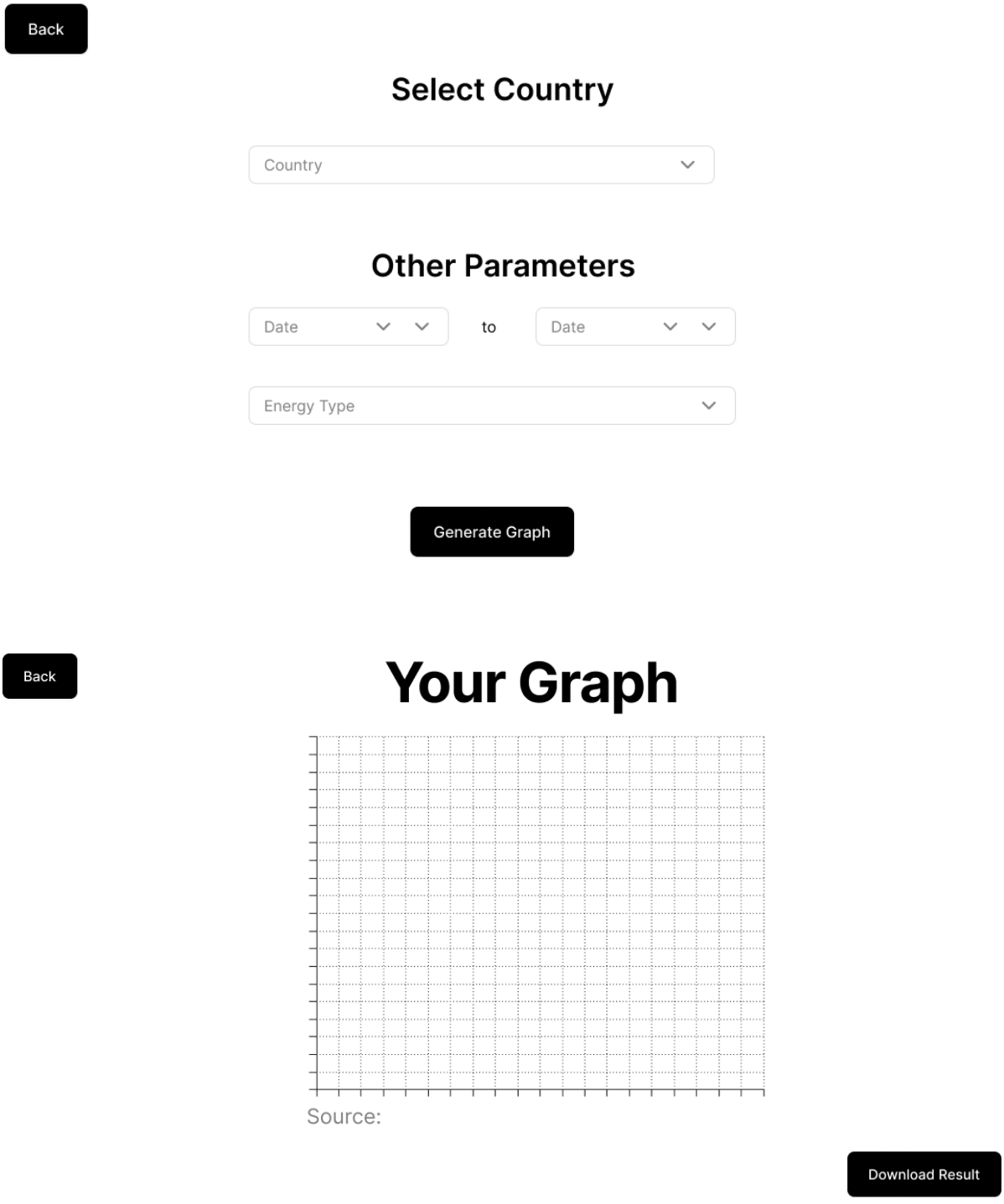
First submissions:

Database:

Client-Side:





Server-Side:

- To hand-in:

- A list of all the server routes that will be implemented with details:
    - o Path, method
    - o Parameters expected and format

- A proof of concept of one route that:
    - o Calls a python script by passing parameters
    - o Returns an image in the response

• **Path**: /welcome
• **Method**: GET
• **Purpose**: Get path for the welcoming page that simply has a button. This is a get method that has no need for format and parameters since it has nothing but a button. The button will be coded in the client side to navigate to the following page.
• **Parameters**: This has no need for parameters since the page would only have a button that goes to the next page.
• **Response**:
- 200 status code upon success, the message body will say welcome user.
- 400 status code upon failure. The message body will say: The server cannot or will not process the request due to an apparent client error

• **Path**: /selectGraph
• **Method**: GET
• **Purpose**: This route works on the second page of the website. The page has multiple different graph examples with a button underneath them. The user can choose one of the graphs by clicking the button underneath that graph, where they will be navigated to another page. A page they can fill the queries and get all the data in their selected graph.
• **Parameters**: This page has no parameters since its only function is to show images already on there and a button that will navigate to the next page.
• **Response**:

- 200 status code upon success, the message body will say Successful navigation.
- 400 status code upon failure. The message body will say: The server cannot or will not process the request due to an apparent client error

• **Path**: /infoGraph
• **Method**: POST
• **Purpose**: Post path for taking all the queries you've put in this page on the client side. The queries are country, from and to dates, and type of energy. Once the user submits the queries by clicking the button underneath queries, the server takes the queries and generates a graph based on the provided data. The user is then directed to the next page where they can view the generated graph.
• **Parameters**: Use body parameters. That way, you can simply type the queries inside the json body and get the answer.
• **Response**:
- 200 status code upon success, the message body will show: all the result from the selected/typed query in the json body.
- 400 status code upon failure. The message body will say: The server cannot or will not process the request due to an apparent client error

• **Path**: /infoGraph
• **Method**: POST
• **Purpose**: Post path for taking all the queries you've put in this page on the client side. The queries are country, from and to dates, and type of energy. Once the user submits the queries by clicking the button underneath queries, the server takes the queries and generates a graph based on the provided data. The user is then directed to the next page where they can view the generated graph.
• **Parameters**: Use body parameters. That way, you can simply type the queries inside the json body and get the result in the body in writing.
• **Response**:
- 200 status code upon success, the message body will show: all the results from the selected/typed query in the json body.

- 400 status code upon failure. The message body will say: The server cannot or will not process the request due to an apparent client error

• **Path**: /showGraph
• **Method**: GET
• **Purpose**: GET path for taking all the queries you've put in and turn them into a graph that you will be shown in the result body. This is where the user, if wished, could download the graph on their device. The graph shown will be the one selected.
• **Parameters**: Body parameter. This is basically the same function as the previous, but instead of getting the answer in writing. The data will be shown in image.
• **Response**:
- 200 status code upon success, the message body will show: The generated graph with all the correct data.
- 400 status code upon failure. The message body will say: The server cannot or will not process the request due to an apparent client error