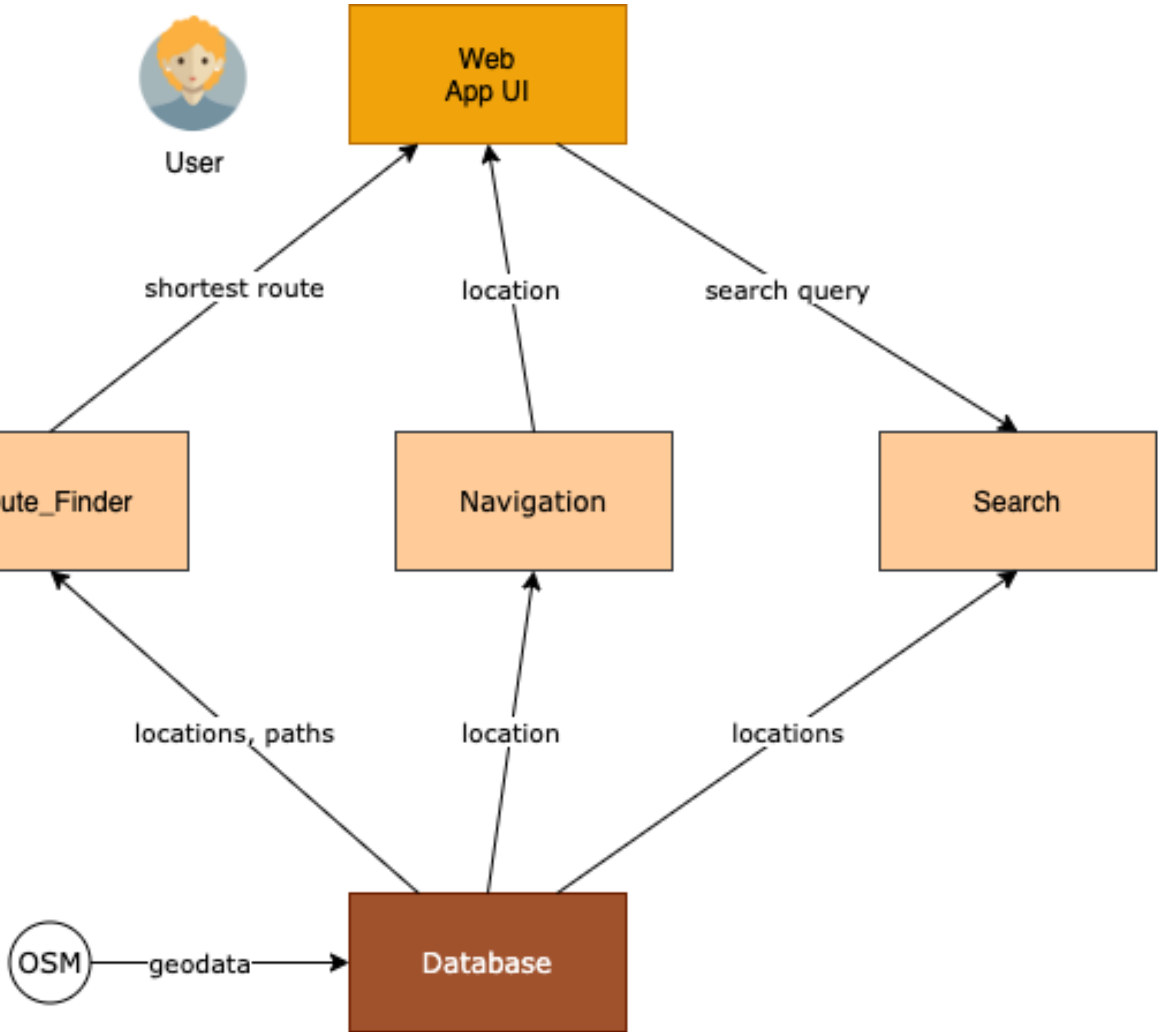
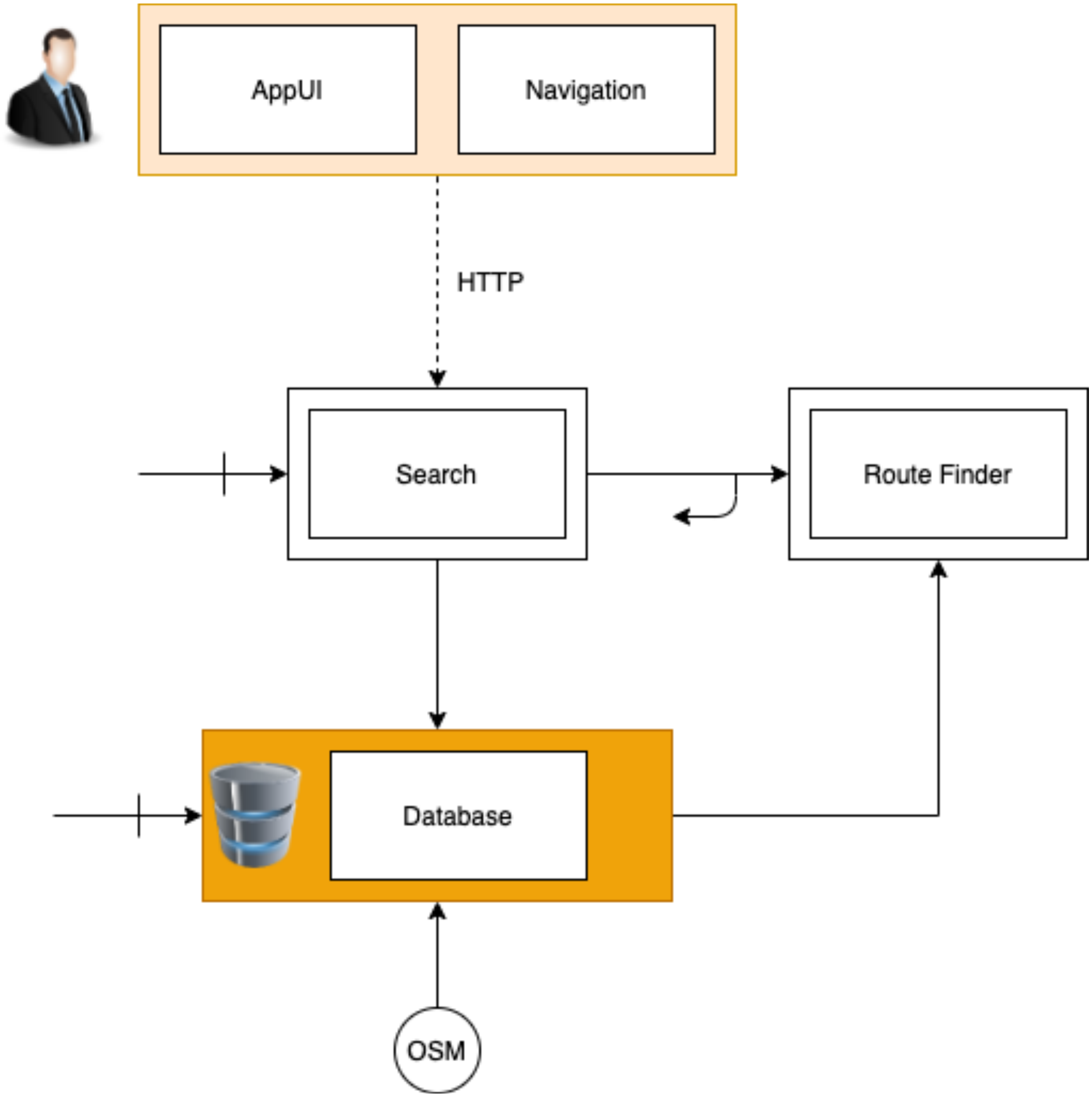
Conceptual Architecture



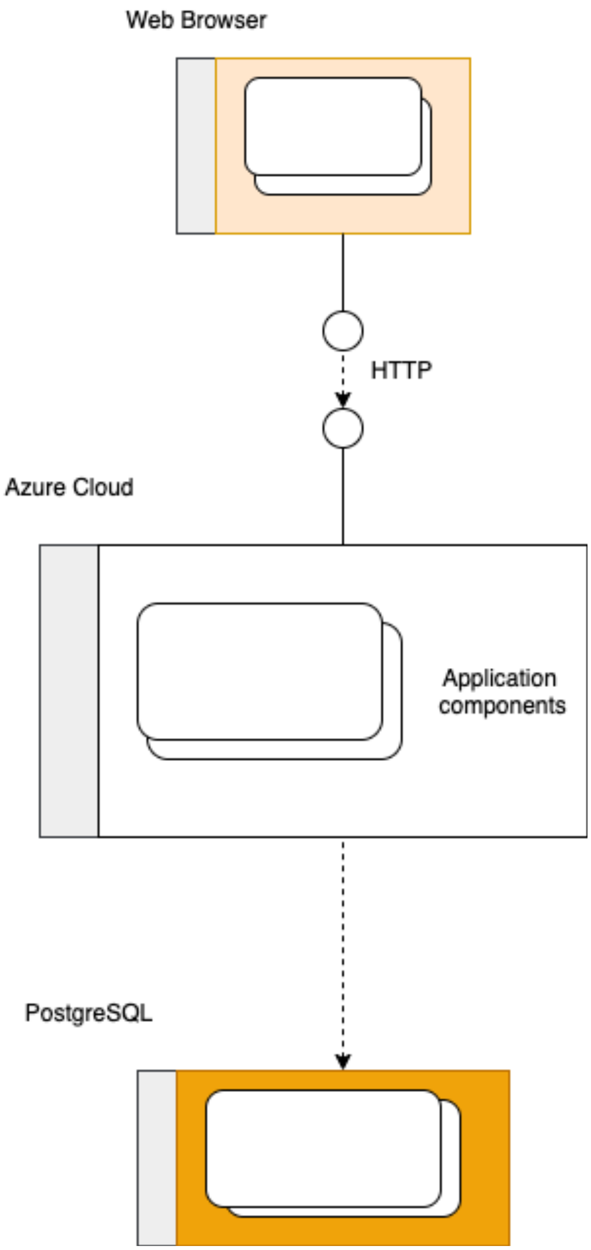
The first step in defining our conceptual architecture is identifying the stakeholders and the external system, which in our case is the user and the OSM dataset. The user needs an interface in order to interact with the system, therefore there must be a WebUI to do so.
Since the main functionality of the web application is to calculate the navigational path,
the data storage is going to be taken care of by the database. Even though the database is not an active component, the majority of the data is still flowing from the database to the other components. The information flow between the database and the route finder is the places of interest and their interconnection. The information flow between the database and the search component is the desired location. The navigation component also receives data from the database in the format of a geographical point. The user invokes a search query through the WebUI in order to get the desired information. The navigation component transmits the point of interest, while the route finder is calculation the shortest path to that same point.

Execution Architecture

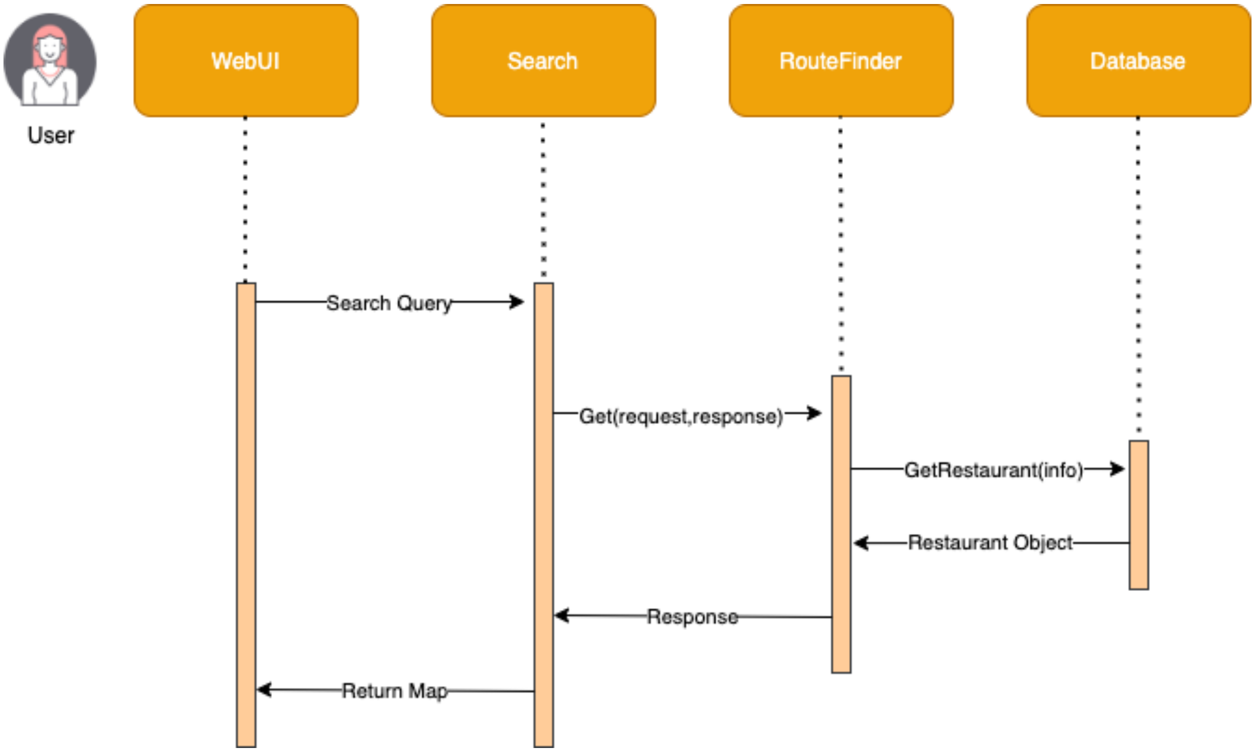


We need to decide where the conceptual components reside in the execution architecture. The WebUI and Navigation components work as user-activated components and reside in the users device. Search and Database wait for requests from other components so they are service components. The WebUI sends a HTTP request with a desired location to the Search component which in turn sends a callback request to the RouteFinder for the desired location. In order for the RouteFinder to calculate the shortest route it retrieves the current location, the desired location and related paths from the database. Once the information is retrieved, it calculates a path and sends a response to the search component which then sends the shortest path to the WebUI which then displays the shortest path to the user.

Implementation Architecture



The web browser contains the app components and is connected to an API which is then connected to the HTTP interface. Such flow connects the web browser to the cloud. Then, PostgreSQL is used as a relational database management system. These container components include multiple application components needed for the functionalities of the web application.



An alternate view of the implementation architecture can be seen in the following sequence diagram. The user initiates a search query for an eatery through the web application's user interface, which is then requested as a 'get' function with parameters relating to the request type and response. The RouteFinder component then requests the appropriate eatery which can be a restaurant or a cafe considering the users' preference and distance. After the database has processed the request, a corresponding restaurant object is returned as a direct response on a graphical map on the UI.