Starting with front end since that’s what every web application must have, if we don't have a high requirement of user-interfaces, we can just use plain JavaScript along with HTML and CSS. But if we want the front end to be more reliable and responsive, or we want to repurpose it to be mobile someday, we can choose from a variety of third-party frameworks such as Angular or React.

For back-end, we need to determine the scale of this web application first. For example, how many active users we are targeting at? How many requests we are making? Thus, we can have the QPS number. Based on this number, we can choose which type of database we are going to adopt: SQL(1k QPS) or NoSQL(10k QPS). When we build this as console app, we basically only make one request at a time, but a full-scaled web application will serve a lot more concurrent request from the client side and the server side will need to return different kinds of data. If we need to read the databases a lot more often which will likely cause a bad network traffic, we might consider caching the data that we retrieve from the DB and in the consequent request without hitting the database. We will probably set up different servers to handle different kinds of request/service. In this case, we may also want to use a load balancer to distribute client requests or network load efficiently across multiple servers.