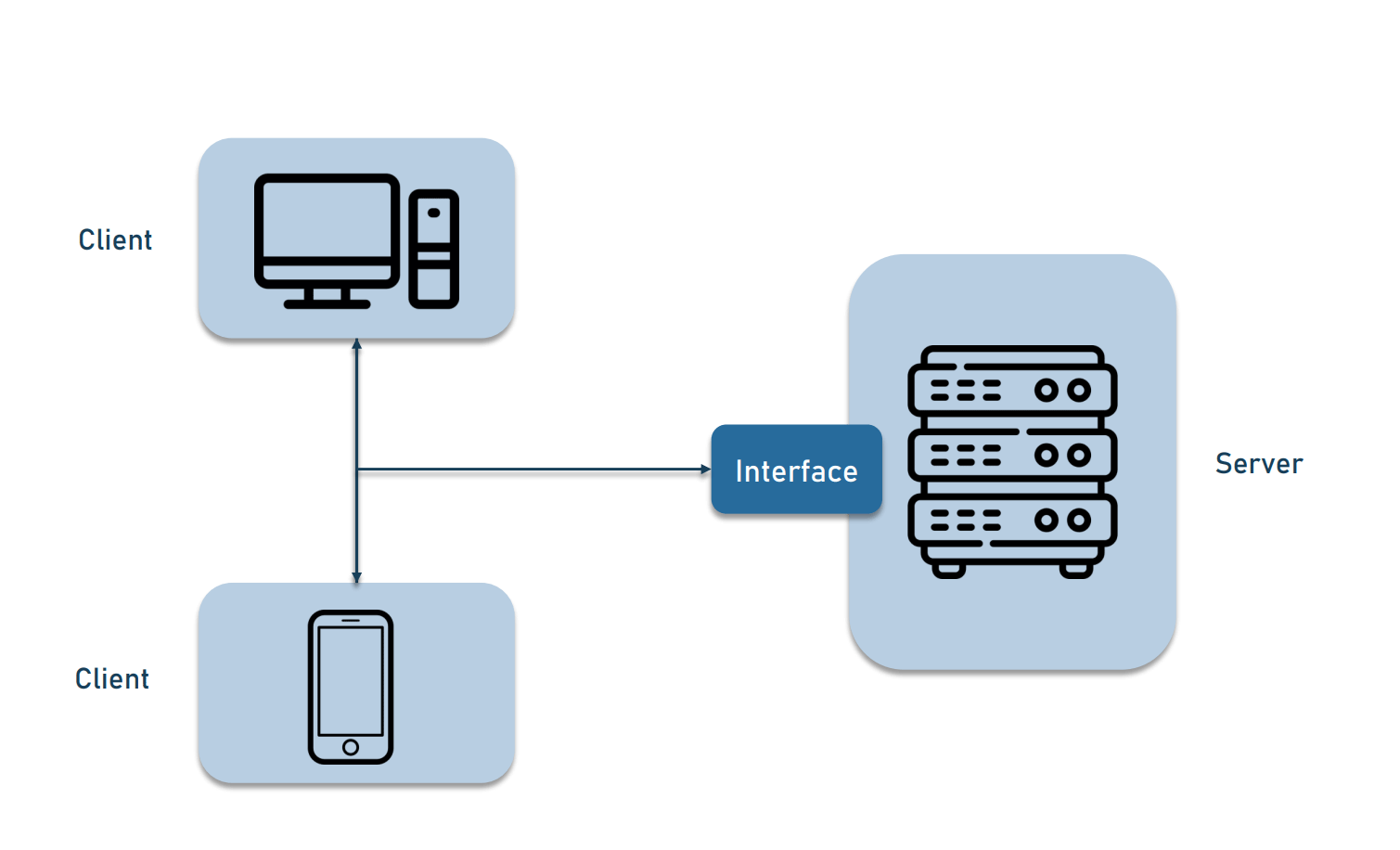
L’applicazione è divisa in due macrosistemi che sono indipendenti l’uno dall’altro:

* Backend: parte del sistema che agisce sul server ed è controllata utilizzando il framework Django, scritta in linguaggio Python
* Frontend: parte che viene presentata all’utente, ossia l’applicazione web vera e propria con cui egli interagisce, realizzata principalmente tramite la libreria React.js e scritta in Javascript (nella forma JSX)

Questi due sistemi, che rappresentano rispettivamente il server e il client, interagiscono tra di loro tramite chiamate asincrone ai metodi più comuni del protocollo http (GET, POST, PUT, DELETE) da parte del client verso il server, in un modello architetturale chiamato REST (REpresentational State Transfer) e tramite un interfaccia definita come API (Application Programming Interface)



Questo tipo di architettura permette di avere un’applicazione che può presentarsi in modo diverso all’utente a seconda del dispositivo utilizzato e di avere anche client completamente diversi dalle tipiche pagine web che possono usufruire delle risorse messe a disposizione dal server, purché concordino sull’interfaccia.

Le applicazioni REST interagiscono tramite metodi del protocollo HTTP tipicamente con questa convenzione (nel nostro caso risorsa e collezione sono equivalenti):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| URL | Metodi | | | |
| GET | PUT | POST | DELETE |
| http://fitcourse/api/risorsa | **Restituisce** una collezione di elementi | **Sostituisce** un’intera collezione con un’altra | **Crea** un nuovo elemento nella collezione | **Elimina** l’intera collezione |
| http://fitcourse/api/risorsa/id | Restituisce l’elemento della collezione tramite identificativo | Sostituisce (o aggiorna) l’elemento indirizzato | Tratta l’elemento come una collezione e crea un elemento | Elimina l’elemento indirizzato dalla collezione |

Nel nostro caso le possibilità con sfondo rosso nella tabella non sono state utilizzate (e tipicamente non vengono utilizzate).