**UZROCI:**

* **Website Traffic (posecenost sajta)**
* **Trosenje resursa servera (zavisi od trenutnog broja korisnika i od ponudjenih funkcionalnosti sajta koje u zavisnosti od slozenosti mogu da salju velike zahteve serveru)**
* **Web Hosting (u zavisnosti od prve dve stavke neophodno je nadograditi web hosting kako bi se smanjilo vreme odgovora servera)**

**OPTIMIZACIJA CSS-A**

**OPTIMIZACIJA JAVASCRIP KODA**

**OPTIMIZACIJA PHP KODA**

1. Koristiti gotove PHP funkcije sto je moguce vise
2. Koristiti single quotes, umesto “” koristiti ’’
3. U uslovu koristiti === umesto ==
4. Koristiti u uslovu *isset()* umesto *count( ),* *strlen( ),* *sizeof( )*
5. Ne ponavljati istu funkciju(sa povratnom vrednoscu) vise puta vec je dodeliti nekoj promenljivoj koja ce se koristiti umesto f-je
6. Funkcije za stringove poredjane po brzini od najbrze: *strtr, str\_replace, preg\_replace*
7. Ne ponavljati istu funkciju(sa povratnom vrednoscu) vise puta vec je dodeliti nekoj promenljivoj koja ce se koristiti umesto f-je
8. Koristiti switch case
9. Izbegavati koriscenje return u definiciji funkcije, npr:

*function test(&$input){*

*$input = .................*

*}*

*$x;*

*test($x);*

1. Koristiti objektno programiranje sto vise moguce
2. Koristiti nasledjivanje klasa i metode iz klase naslednice, tj. smanjiti broj kreiranih objekata sto je vise moguce
3. Iskljuciti prikazivanje debugging poruka
4. Koristiti tehnologije za kesiranje
5. Posle koriscenja podataka iz baze zatvarati konekciju, radi oslobadjanja maemorije
6. Smanjiti broj zahteva upucenih bazi, tj. smanjiti broj query-ja sto je vise moguce
7. Umesto XML-a u web servisima koristiti JSON niz funkcijama *json\_encode( )* i *json\_decode( )*
8. Koristiti apsolutnu putanju sto je vise moguce nego relativnu
9. Koristiti . pri spajanju stringova kada nisu dodeljeni promenljivoj, u suprotnom bolje je koristiti ,
10. Koristiti prilikom testiranja timestamp tj. $\_SERVER['REQUEST\_TIME']