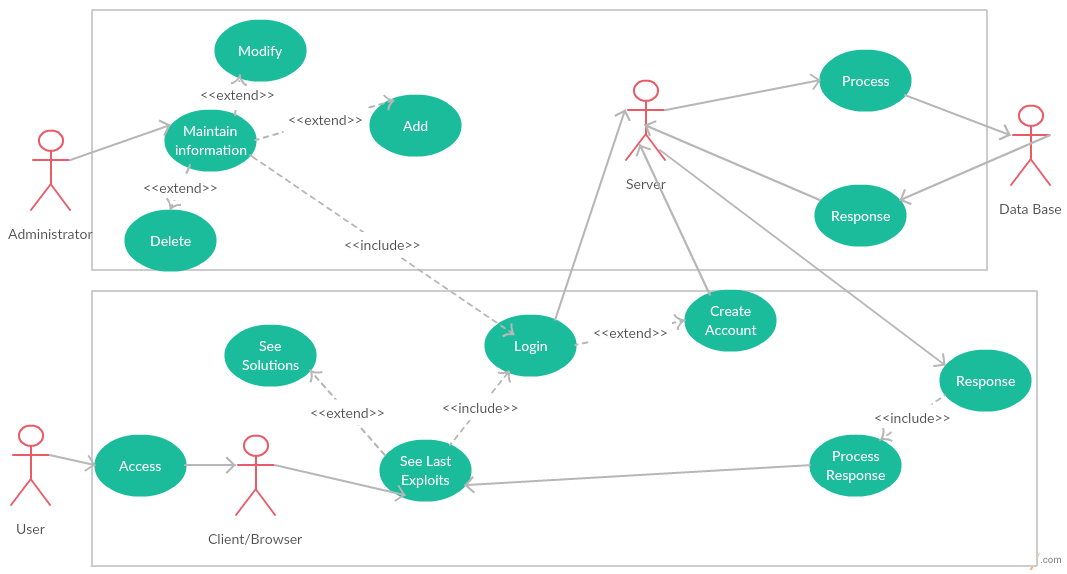
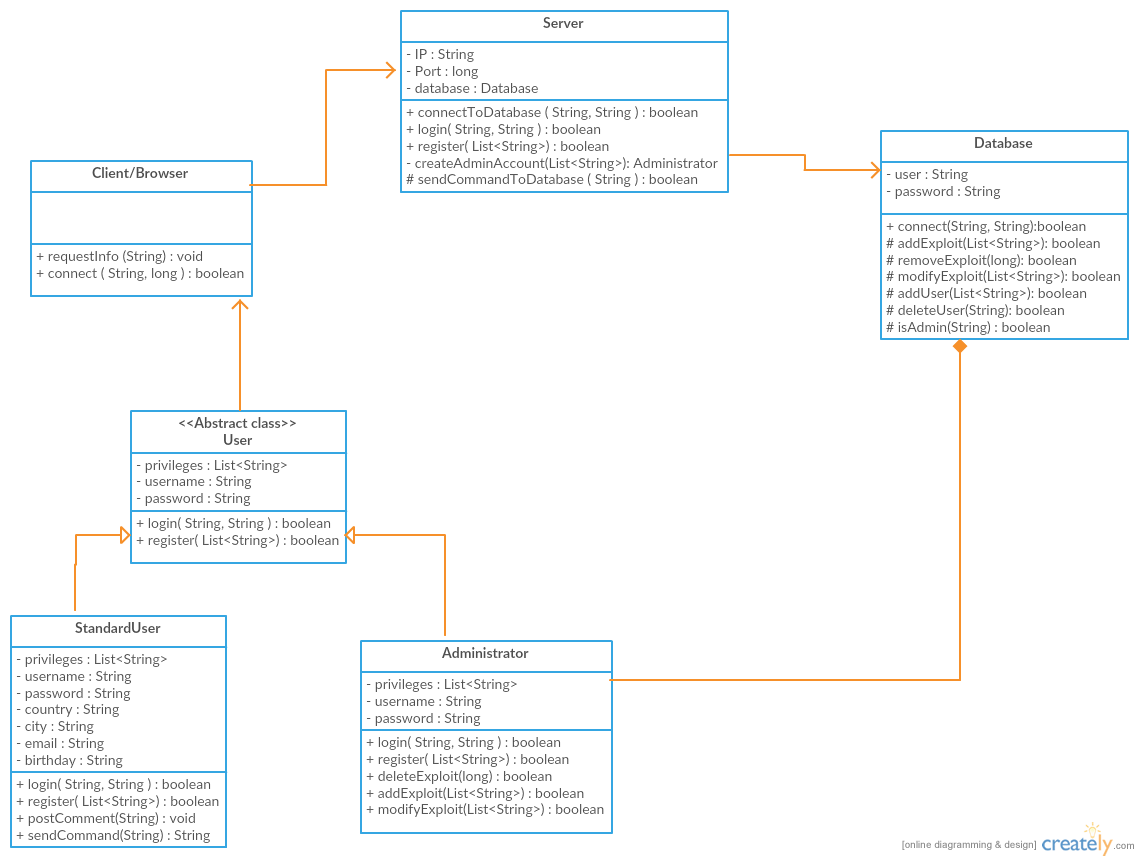
**Web App Security Alerter**

1. **Diagrame**



**2. Etapele intermediare ale dezvoltarii proiectului**

I. Realizarea fisei cerintelor pentru a putea dezvolta ideea de ansamblu a proiectului

a.Construirea diagramei use-case, care urmareste modelul fisei.

b. Construirea diagramei de clase - actorii vor fi clasele, iar scenariile de utilizare vor fi in mare parte transformate in metode.

II. Luarea deciziilor privind abordarea problemei

a. Alegerea limbajului de programare (PHP)

b. Alegerea design-ului website-ului - aceasta alegere se va face in concordanta cu nevoile utilizatorului si cerintele problemei : responsive, intuitiv, atragator si simplu de folosit.

c. Brainstorming privind securitatea aplicatiei web - abordarea situatiilor de genul SQL Injection ...

d. Documentare privind API-urile ce vor fi folosite, bibliotecile si framework-ul de dezvoltare.

e. Impartirea task-urilor pe module pentru dezvoltarea ulterioara.

III. Dezvoltare

a. Dezvoltarea front-end-ului (impartirea task-urilor pe module; fiecare dezvoltator va realiza cate un segment; header-ul si footer-ul vor fi template-uri

care vor fi incluse la fiecare pagina din website, schimbandu-se doar main-ul)

b. Construirea bazei de date (realizarea diagramei de clase, pachetelor, tabelelor)

c. Realizarea conexiunii intre baza de date, front-end si back-end.

d. Dezvoltarea back-end-ului.

i. Instalarea server-ului Apache.

ii. Testarea conexiunii cu baza de date si front-end-ul.

iii. Scrierea codului.

IV. Integrarea: preluarea modulelor de la dezvoltatori si combinarea lor in vederea rezultarii unei aplicatii functionale.

V. Testarea: dupa integrare, testarea va urmari eventualele bug-uri rezultate din integrare (poate unele module nu vor fi compatibile, anumite use-case-uri neacoperite, etc)

VI. Rezolvarea bug-urilor

Obs. Etapele V si VI se vor face ciclic, insemnand ca dupa rezolvara bug-urilor, se vor face un nou rand de teste deoarece exista posibilitatea ca atunci cand bug-urile vor fi, altele vor aparea.

**3. Consideratii preliminare privind structura (modelarea) datelor si provenienta lor (servicii web, API-uri publice)**

I. Folosirea API-ului website-ului exploit-db si a SearchSploit-ului : https://github.com/offensive-security/exploit-database

II. Dezvoltarea unui API public care va oferi solutii pentru exploit-urile gasite pentru a minimiza sansa vulnerabilitatii software a aplicatiei noastre si de asemenea, pentru a face mai usoara comunicarea dintre client si server si pentru o dezvoltare ulterioara facila a server-ului.

III. Respectarea paradigmei REST.

IV. Implementarea notificarii in timp real folosind Pusher - https://pusher.com/tutorials/realtime-notifications