**Funcționalități**

Aplicația pe care o voi dezvolta este un sistem de bug tracking menit să țină evidența bug-urilor ce pot apărea în timpul dezvoltării unei aplicații software. Aplicația va oferi caracteristicile de bază pe care le deține orice bug tracker însă va încerca să ofere noi funcționalități care vor ușura munca dezvoltatorului.

Utilizatorii vor avea posibilitatea de a se înregistra pentru a putea folosi aplicația, putându-se mai apoi autentifica folosind numele de utilizator și parola alese. De asemenea, manager-ul proiectului va putea adăuga noi membrii la echipa sa. Pentru aceștia se va crea un cont de utilizator cu o parolă generată automat. Această parolă va fi trimisă prin e-mail noilor utilizatori la adresa specificată la adăugare, ei având ulterior posibilitatea de a o modifica.

Aplicația va putea urmări mai multe proiecte la care lucrează un dezvoltator, oferindu-i acestuia posibilitatea de a administra bug-urile apărute în cadrul tuturor proiectelor sale. Acesta va putea adăuga fie bug-uri, fie noi funcționalități la care trebuie să se lucreze în viitor, permițând astfel să se facă o distincție clară între acestea. Toate acestea vor putea fi organizate în milestone-uri (etape importante din ciclul de viață al proiectelor care trebuie îndeplinite până la o anumită dată prestabilită), care vor fi adăugate de către manager-ul de proiect.

În formularul care va trebui completat la adăugarea unui nou task, se va putea specifica, pe lângă un titlu sugestiv, o descriere amănunțită, și va exista posibilitatea de a adăuga fișiere sau fotografii auxiliare. De asemenea, acum se va putea selecta gradul de severitate pe care îl are respectiva problemă (low, medium, high, critical). Aceasta va putea fi atribuită unui anumit milestone, specificându-se astfel data la care va trebui să fie rezolvată. Va exista posibilitatea de a alege un responsabil pentru rezolvarea bug-ului, însă acest lucru nu va fi obligatoriu. Pentru o grupare a task-urilor în funcție de anumite categorii relevante pentru echipa de dezvoltatori, se vor putea adăuga diferite etichete ce constau din cuvinte cheie sugestive.

O altă modalitate de a adăuga bug-uri va fi prin trimiterea de e-mail-uri de către utilizatori, fie aceștia dezvoltatori sau clienți. Astfel, subiectul e-mail-ului va reprezenta titlul noului task, iar conținutul va fi descrierea adiacentă. Prin această funcționalitate se va putea primi feedback din partea clienților mult mai rapid și mai eficient.

O dată ce un bug este asignat unui dezvoltator, acesta va trece din starea “open”, în starea “in progress”, sugerând astfel că s-a început rezolvarea lui. La finalul dezvoltării, acesta va trece în starea “in testing” și abia după ce testarea s-a finalizat cu succes bug-ul va putea fi considerat “closed”.

Pentru a putea fi la curent cu toate modificările aduse bug-urilor, dezvoltatorii implicați vor primi notificări privind aceste schimbări. Cei care vor primi notificări vor fi persoanele care au creat respectivul bug, sunt asignate lui sau au scris comentarii în legătură cu acesta. Următoarele schimbări vor declanșa trimiterea de notificări: noi comentarii sau fișiere atașate task-urilor, un nou task asignat dezvoltatorului în cauză, modificarea stării unui task sau închiderea unui task în care dezvoltatorul este implicat.

Dezvoltatorii vor avea posibilitatea de a adăuga notițe pentru fiecare din task-urile proprii, putând astfel să consemneze idei pentru studii ulterioare. De asemenea, se va pune la dispoziție accesul la un motor de căutare (Bing) pentru a căuta mai facil răspunsuri la întrebări tehnice sau de altă natură. Acestea vor putea fi salvate în detaliile adiacente fiecărui task, nefiind nevoie de o recăutare următoare sau de adăugarea unui bookmark în browser-ul personal.

În pagina de listare a task-urilor, se vor putea aplica diferite filtre pentru a ușura căutarea în funcție de o categorie. Vor exista filtre predefinite, însă dezvoltatorul va avea de asemenea posibilitatea de a-și crea propriile filtre personalizate. Task-urile vor putea fi afișate în funcție de milestone-ul la care sunt asignate, etichetele pe care le dețin, categoria din care fac parte, gradul de severitate, starea etc.

Aplicația va oferi de asemenea posibilitatea de a genera rapoarte ce evidențiază volumul de muncă parcurs sau viitor. Se va putea vizualiza și lista de activități care au fost întreprinse în fiecare zi, iar manager-ul va putea primi prin e-mail un sumar al activităților la o perioadă de timp selectată de el: zilnic, săptămânal sau lunar.

**Platforma de dezvoltare**

Aplicația de față este o aplicație web-based, fiind construită folosind framework-ul ASP.NET MVC 5. Am ales să folosesc ASP.NET deoarece acesta este susținut de puternica platformă .Net care oferă deosebit de multe facilități. Server-ul ce va găzdui aplicația va fi Internet Information Services (IIS).

MVC este un model arhitectural compus din trei componente interconectate: Model, View, Controller. Modelul conține reprezentarea informației cu care utiizatorii interacționează. Aceasta poate fi constituită de view model, ce reprezintă datele transmise între view și controller, sau de domain model, care conține datele de bază ale aplicației, precum și operațiile de manipulare ale acestora. View-ul ajută la redarea informației într-o interfață grafică, iar controller-ul, care procesează cererile primite, efectuează operațiile necesare pe model și selectează view-ul care trebuie redat utilizatorului.

Principalul motiv pentru care am ales să folosesc modelul arhitectural MVC este faptul că acesta respectă principiul separării intereselor (separation of concerns). Fiecare componentă este bine definită și independentă: logica de manipulare a datelor este conținută numai în model, logica de afișare a datelor numai în view, iar codul care se ocupă cu cererile utilizatorului este conținut doar în controller. Astfel, aplicația devine mult mai ușor de menținut și extins, oferind și posibilitatea de a scrie cod curat și organizat. De asemenea, MVC înlesnește realizarea de unit teste și îmbunătățește search engine optimization. Modelul arhitectural MVC oferă posibilitatea dezvoltatorului să aleagă șabloanele URL pe care link-urile aplicației sale să le respecte, făcându-le astfel mult mai ușor de căutat și înțeles.

Pentru a asigura persistența datelor, am ales să folosesc Microsoft SQL Server 2012. În ceea ce privește manipularea datelor din baza de date, platforma .Net pune la dispoziție ca instrument de Object Relațional Mapping, Entity Framework. Versiunea folosită este Entity Framework 6, iar modalitatea de creare a tabelelor și de manipulare a datelor este Code First. Pentru a utliza Entity Framework Code First, este necesar doar să creezi clasele ce reprezintă entitățile de bază ale aplicației, creându-se mai apoi automat tabelele bazei de date. Pentru a manipula datele, se folosesc instanțe ale claselor în cauză.

Pentru o mai facilă mentenabilitate a aplicației, am ales să folosesc dependency injection. Acesta presupune ca fiecare componentă să nu cunoască detalii despre o alta cu care trebuie să comunice, ci să comunice cu aceasta prin interfețe. Acest lucru este realizat printr-un dependency injection container care asignează clasele necesare fiecărei interfețe specificate. Container-ul ales este Ninject.

În ceea ce privește interfața cu utilizatorul, pe lângă HTML, CSS, Javascript, jQuery și Ajax, voi folosi și jQueryUI și Bootstrap, care adaugă un plus de interactivitate și atractivitate aplicației.