1. **Introducere**
   1. **Contextul**

Într-o lume în care tehnologia evoluează tot mai mult, în care aceasta își pune din ce în ce mai mult amprenta pe viața noastră de zi cu zi, medicina este una din ariile care s-a bucurat de o atenție mare. Deși de-a lungul ultimilor ani, au apărut multe tehnologii noi care revoluționează atât speranța de viață a oamenilor cât și munca și pregătirea medicilor și a doctorilor, unele aspecte chiar foarte minore au rămas neglijate. Deși am ajuns să operăm la o precizie de microni, să avem roboți asistenți sau să învățăm pe holograme, lucruri care acum nu mult timp păreau luate din filme, am uitat un aspect foarte important: pacientul.

* 1. **Motivația**

În momentul actual, aplicațiile medicale care sunt oferite pe piață sunt împărțite în două mari categorii: aplicațiile pentru folosința cadrelor medicale și aplicațiile pentru folosința tuturor. Fiecare categorie are avantajele și dezavantajele sale.

Pentru cadrele medicale ce lucrează într-un mediu privat, aplicațiile sunt deobicei doar de gestiune a pacienților, de a face programări etc. fără a da prea multe opțiuni pacientului. Cele din mediul spitalelor sau clinicilor de stat sunt la fel, și deși au o bază de date comună și folositoare în multe cazuri, aceastea sunt gândite în mare parte ca o aplicație de gestiune economică, fără a fi gândite pentru nevoile medicilor sau a pacienților. În plus rareori sunt acestea ținute la zi cu nevoile curente sau oferită mentenanță, și din păcate aspectul vizual și interfața cu utilizatorul lasă de dorit.

Pentru pacienți, sau simpli oameni preocupați de starea lor de sănătate în schimb, exista o multitudine de aplicații venite în folosul lor. De la aplicații embeded, web sau mobile, până la o combinație dintre acestea. Deși aeste aplicații de cele mai multe ori sunt ținute la zi cu nevoile utilizatorilor, au un aspect plăcut și sunt ușor de utilizat, acestea sunt lipsite de o bază de date specializată, completată de un cadru medical specializat. Un alt dezavantaj este faptul că deobicei aceste aplicații sunt axate pe un domeniu singular, iar pentru date mai complete ar trebui ca utilizatorul să aibă o multitudine de aplicații instalate.

Problema generală este că deși aceste două tipuri de aplicații există de sine stătătoare, ele nu funcționează concomitent sau cel puțin nu pot interacționa. Orice pacient ar trebui să poată vedea istoricul său medical, eventualul tratament primit, să poată interacționa cu cadrele medicale în cazul în care acesta este internat sau îi este oferită o anumită îngrijire medicală de un anume tip. În același timp, cadrul medical specializat trebuie să poată să își desfășoare activitatea în continuare într-un mod cât mai rapid și eficient.

* 1. **Descrierea proiectului**

Având în vedere nevoile generale ale utilizatorilor și a cadrelor medicale și a probleme identificate mai sus, am găsit următoarele funcționalități și caracteristii:

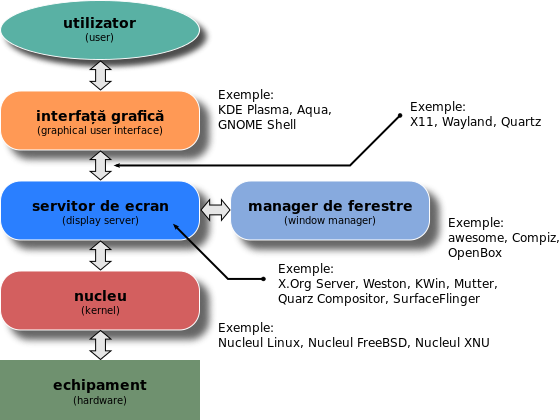
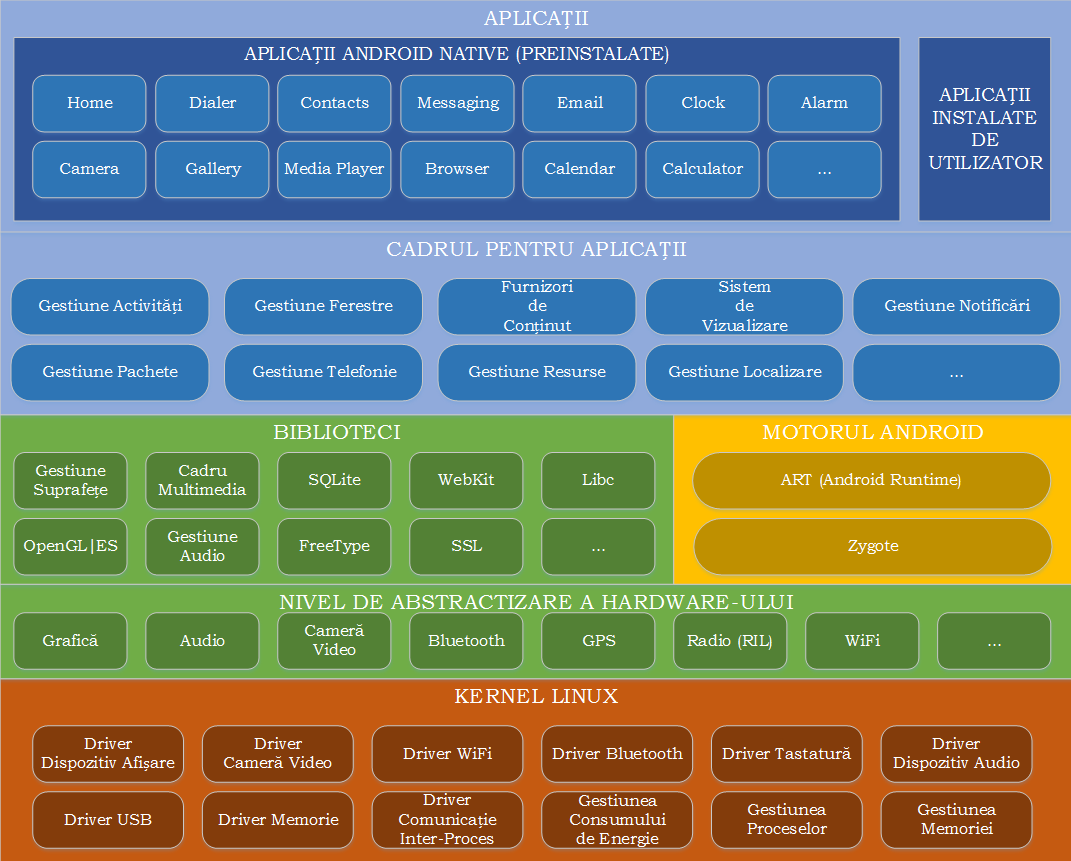
* **Mobilitate**
  + Aplicația trebuie să fie în variantă mobilă, atât pentru cadrele medicale cât și pentru pacienți (utilizatori).
  + Cadrele medicale ar trebuie să poate avea în orice moment acces la toate datele necesare, și să nu trebuiască să fie limitate de o anume locație.
  + Pacienții (utilizatorii) ar trebui să poată avea acces oricând la datele lor medicale, reușind astfel să introducem o întreaga bibliotecă de acte, analize și istorice medicale intr-un singur telefon sau tabletă.
* **Costuri reduse pentru platforme**
  + Având în vedere faptul că această aplicație ar trebui să poate fi accesibilă de marea masă de oameni dar și instituții medicale, aceasta trebuie implementată pe o platforma disponibilă aproape oricui.
* **Acces la propriul istoric medical**
  + Pacientul ar trebui să poată accesa istoricul medical fără a fi nevoit să îl introducă el sau să folosească multe aplicații. Acesta ar trebui să poată accesa direct baza de date folosită de cadrul medical.
* **Cererea asistenței medicale**
  + Pe timpul internării, pacientul ar trebui să poată cere asistență medicală fără a trebui să se deplaseze, mai ales că în unele cazuri această deplasare este imposibilă
* **Implementarea funcționalităților necesare cadrelor medicale**
  + Aplicația trebuie să dispună în continuare de facilitățile care le au medicii în momentul actual, poate chiar să le extindă
* **O interfață prietenoasă**
  + Aplicația trebuie să aibă o interfață ușor de înțeles și utilizat și să ofere informațiile necesare oricărui tip de utilizator.
* **Împărțirea istoricului medical cu rude sau prieteni**
  + Aplicația ar trebui să permită împărțirea istoricului medical cu alți membrii ai familiei sau prieteni apropiați, mai ales pentru persoanele vârstnice.

Luând în considerare caracteristicile de mai sus, aplicația va avea următorul format:

* **Utilizarea sistemului Android**
  + Aplicația va fi dezvoltată pentru dispozitive cu sisteme de operare Android pentru a rezolva chiar 2 caracteristice chieie:
    - Platformele ce rulează Android sunt mobile
    - Platformele ce rulează Android impun costuri reduse pentru platformă. Acestea se găsesc cu prețuri variabile dar sunt și platforme foarte ieftine ce vor putea rula aplicația. Desigur, acest lucru este în comparație cu restul platformelor populare.
* **Aplicația va avea trei tipuri de utilizatori**
  + Pentru a îndeplini cerințele acces la propriul istoric medical, cererea asistenței medicale și împărțirea istoricului medical cu rude sau prieteni, aplicația va avea trei tipuri de utilizatori, fiecare având diferite nivele de acces și funcții specifice. Cele trei tipuri de utilizatori vor fi:
    - Cadru medical
    - Pacient
    - Familie
* **Folosirea unor elemente vizuale plate**
  + Pentru a asigura o interfață vizuale plăcută, aplicația va conține doar elemente plate și familiare din setul “Material Design Android”.

1. **Fundamentare teoretică**
   1. **Sistemul de operare Android**

Android este un sistem de operare și o platformă pentru telefoane mobile, tablete, ceasuri deștepte și mai nou și alte dispozitive precum televizoare și mașini. Acesta este bazat pe nucleul Linux însă permite dezvoltatorilor să scrie aplicații în limbajul Java prin intermediul unor biblioteci dezvoltate de Google sau după caz, de alte terțe. Deși la început, sistemul de operare nu era în totalitate publicat, început cu 2008 acesta a devenit produs software cu licență de tip Open Source, motiv în plus pentru dezvoltatori să investească timp și să creeze diverse aplicații ieșite din comun. [[/]](http://www.eweek.com/mobile/google-open-sources-android-on-eve-of-g1-launch)



**Arhitectura Sistemului de Operare Android** [**[/]**](https://ocw.cs.pub.ro/courses/_media/eim/laboratoare/laborator01/android_architecture.png?w=600&tok=743a89)

Capacitățile platformei sunt numeroase, făcând această platformă și mai potrivită pentru orice tip de aplicație mobilă. Printre caracteristicile suportate actul regăsim [[/]](https://developer.android.com/about/index.html) [[/]](https://www.engadget.com/2007/11/12/googles-android-os-early-look-sdk-now-available/) :

* Configurații dispositive - adaptabilă la configurații mai mari, VGA, biblioteci grafice 2D și biblioteci grafice 3D.
* Stocare de date
* Conectivitate – suportă tehnologii de conectivitate incluzând GSM/EDGE/HSDPA/LTE, CDMA, EV-DO, UMTS, Bluetooth și Wi-Fi.
* Mesagerie instant
* Navigatorul de web
* Mașina virtuală Dalvik
* Suport media
* Suport hardware additional - poate utiliza camere video/foto, touchscreen, GPS, accelerometru, cititor de amprente, diverse periferice și grafică accelerată 3D.
* Mediu de dezvoltare
* Piața Android
* Multi-touch
* Și altele...
  1. **Android Studio**

**2.2.1 Noțiuni generale** [**[/]**](https://developer.android.com/studio/intro/index.html)

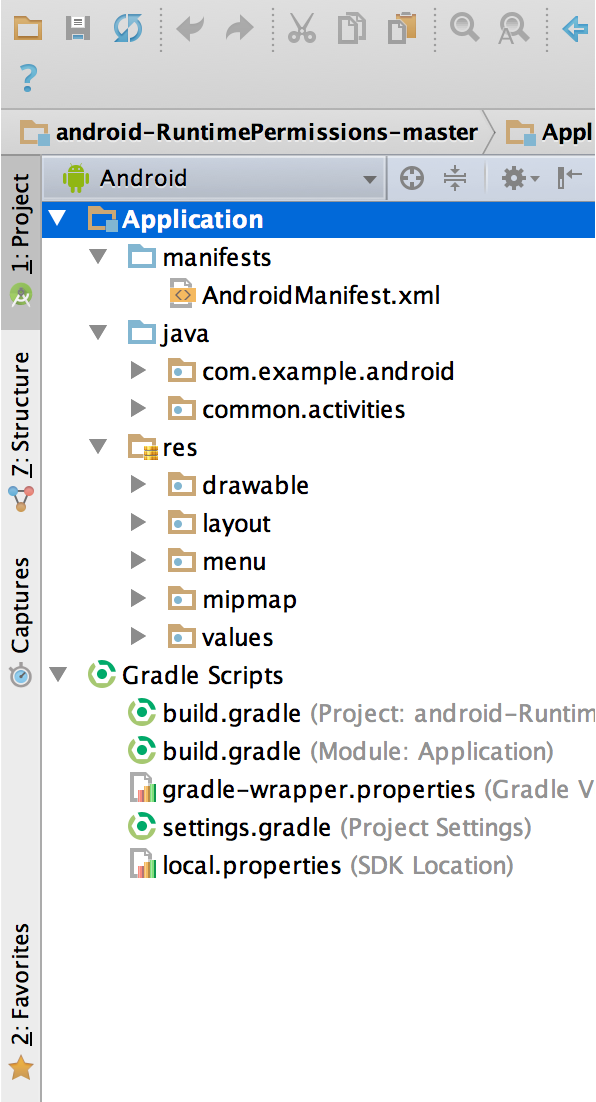
Mediul de dezvoltare Android Studio a fost anunțat in 2013 la conferința Google I/O și a fost realizat ca și o alternativă a mediului de dezvoltare Eclipse. Acesta este bazat pe mediul de dezvoltare Java a celor de la InteliJ IDEA. În scurt timp, acesta a devenit mediul oficial de dezvoltare a aplicațiilor Android.

Android Studio oferă următoarele facilități:

* Un sistem flexibil de construcție bazat pe Gradle
* Un emulator rapid și bogat în caracteristici
* Un mediu unificat în care vă puteți dezvolta pentru toate dispozitivele Android
* Rulare Instantă pentru a împinge modificările aplicației în desfășurare fără a construi un nou APK
* Șabloane de coduri și integrare GitHub pentru a vă ajuta să construiți caracteristici comune ale aplicațiilor și să importați coduri de probă
* Instrumente și cadre de testare extinse
* Instrumente pentru scanare pentru a prinde performanța, gradul de utilizare, compatibilitatea versiunilor și alte probleme
* Suport C ++ și NDK
* Suport încorporat pentru Google Cloud Platform, facilitând integrarea serviciului Google Cloud Messaging și a aplicației Engine Engine

**2.2.2 Structura proiectului**

Fiecare proiect din Android Studio conține



unul sau mai multe module cu fișiere de cod

sursă și fișiere de resurse. Tipurile de module

includ:

* Module de aplicații Android
* Module de bibliotecă
* Modulele Google App Engine

În mod implicit, aplicația Android Studio

afișează fișierele de proiect în vizualizarea

proiectului Android, după cum se arată în

figura 2.1. Această vizualizare este organizată

de module pentru a oferi acces rapid la

fișierele sursă cheie ale proiectului.

Toate fișierele de construcție sunt

vizibile la nivelul superior în cadrul

scripturilor de grad și fiecare modul

de aplicație conține următoarele dosare:

* Manifestul: conține fișierul

AndroidManifest.xml.

* Java: conține fișierele de cod sursă

Fig 2.1 -

Java, inclusiv codul de testare JUnit.

* Res: conține toate resursele non-cod,

cum ar fi layout-uri XML, șiruri de caractere UI și imagini bitmap.

* + 1. **Interfața cu utilizatorul**

Fereastra principală Android Studio este alcătuită din mai multe zone logice identificate în figura 2.1.

1. Bara de instrumente vă permite să efectuați o gamă largă de acțiuni, inclusiv difuzarea aplicației și lansarea de instrumente Android.
2. Bara de navigare vă ajută să navigați prin proiectul dvs. și să deschideți fișierele pentru editare. Acesta oferă o vedere mai compactă a structurii vizibile în fereastra proiectului.
3. Fereastra de editor este locul în care creați și modificați codul. În funcție de tipul de fișier curent, editorul se poate schimba. De exemplu, când vizualizați un fișier de aspect, editorul afișează Editorul de Layout.
4. Bara de ferestre a instrumentului rulează în afara ferestrei IDE și conține butoanele care vă permit să extindeți sau să restrângeți ferestrele individuale ale uneltelor.
5. Ferestrele de instrumente vă oferă acces la anumite sarcini, cum ar fi gestionarea proiectelor, căutarea, controlul versiunilor și multe altele. Le puteți extinde și le puteți restrânge.
6. Bara de stare afișează starea proiectului dvs. și IDE în sine, precum și orice avertismente sau mesaje.

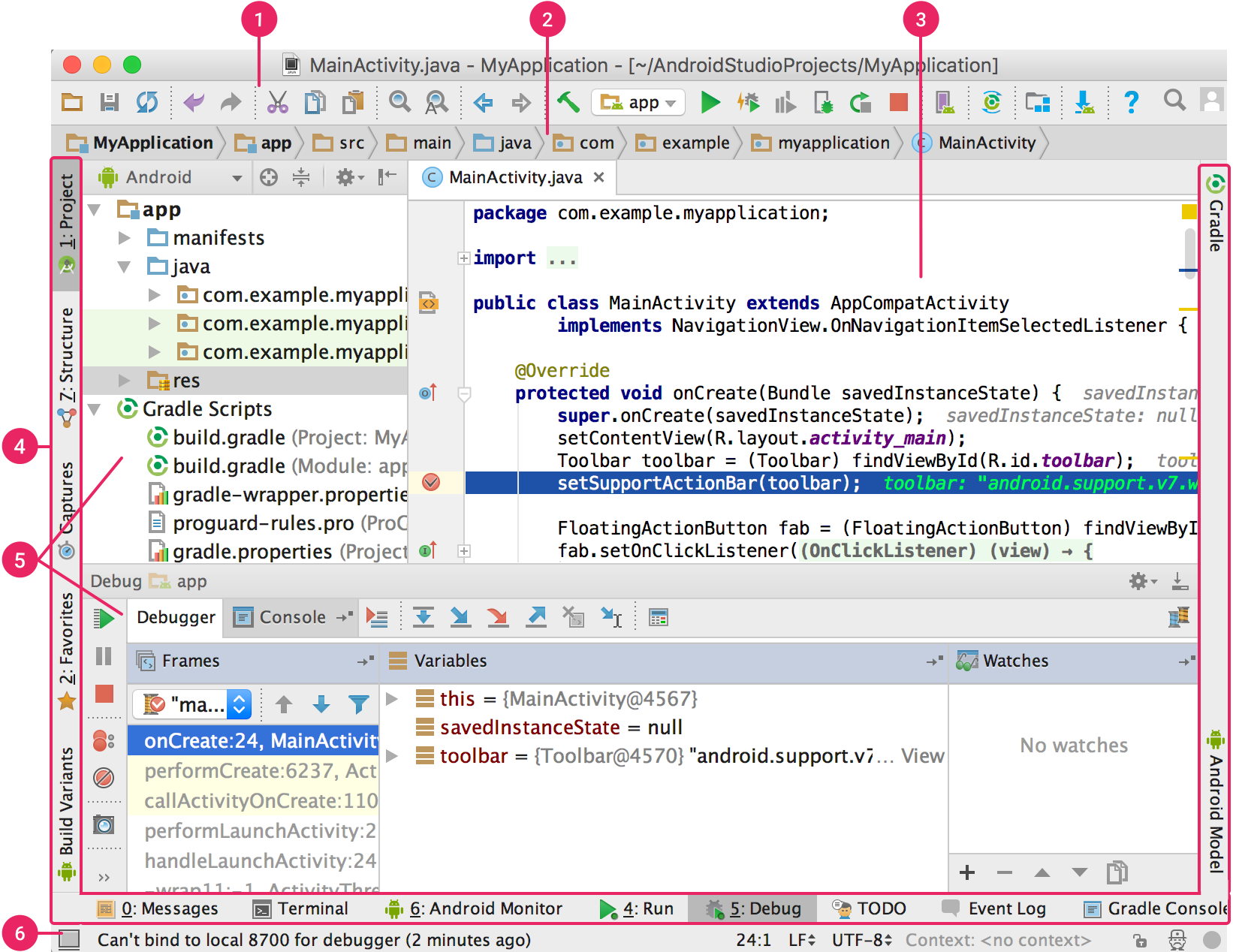


Fig 2.1 -

**2.3 Limbajul de programare Java**

Java este un limbaj de programare orientat-obiect, conceput de către James Gosling la începutul anilor ʼ90, fiind lansat în 1995. Cele mai multe aplicații distribuite sunt scrise în Java, iar noile evoluții tehnologice permit utilizarea sa și pe dispozitive mobile de genul telefon, agenda electronică, palmtop etc. În felul acesta se creează o platformă unică, la nivelul programatorului, deasupra unui mediu eterogen extrem de diversificat. Acesta este utilizat în prezent cu succes și pentru programarea aplicațiilor destinate intranet-urilor [Ștefan Tanasa, Cristian Olaru, Stefan Andrei, "Java de la 0 la expert", Polirom, 2003.].

Limbajul împrumută o mare parte din sintaxă de la C și C++, dar are un model al obiectelor mai simplu și prezintă mai puține facilități de nivel jos. Un program Java compilat, corect scris, poate fi rulat fără modificări pe orice platformă care e instalată o mașină virtuală Java. Acest nivel de portabilitate (inexistent pentru limbaje mai vechi cum ar fi C) este posibil deoarece sursele Java sunt compilate într-un format standard numit cod de octeți care este intermediar între codul mașină (dependent de tipul calculatorului) și codul sursă.

Mașina virtuală Java este mediul în care se execută programele Java. În prezent, există mai mulți furnizori de JVM, printre care Oracle, IBM, Bea, FSF. În 2006, Sun a anunțat că face disponibilă varianta sa de JVM ca open-source.

**2.3 Baze de date**

**2.3.1 Noțiuni generale**