|  |  |  |
| --- | --- | --- |
|  | UNIVERSITY OF CRAIOVA  FACULTY OF AUTOMATION, COMPUTERS AND ELECTRONICS  COMPUTERS AND INFORMATION TECHNOLOGY DEPARTMENT |  |

DEGREE PROJECT

Bochiș Florian Celestin

SCIENTIFIC COORDINATOR

Şef lucrări dr. ing. Marius Marian Adrian

JULY 2016

CRAIOVA

|  |  |  |
| --- | --- | --- |
|  | UNIVERSITY OF CRAIOVA  FACULTY OF AUTOMATION, COMPUTERS AND ELECTRONICS  COMPUTERS AND INFORMATION TECHNOLOGY DEPARTMENT |  |

“Support Application for International Support Mobilities”

Bochiș Florian Celestin

SCIENTIFIC COORDINATOR

Şef lucrări dr. ing. Marius Marian Adrian

JULY 2016

CRAIOVA

*"We were promised flying cars, and instead what we got was 140 characters"*

~ Peter Thiel~

**DECLARAȚIE DE ORIGINALITATE**

Subsemnatul **Bochiș Florian Celestin**, student la specializarea Calculatoare în Limba Engleză din cadrul Facultății de Automatică, Calculatoare și Electronică a Universității din Craiova, certific prin prezenta că am luat la cunoştinţă de cele prezentate mai jos şi că îmi asum, în acest context, originalitatea proiectului meu de licenţă:

* cu titlul “Support Application for International Support Mobilities”,
* coordonată de Şef lucrări dr. ing. Marius Marian Adrian,
* prezentată în sesiunea IULIE 2016.

La elaborarea proiectului de licenţă, se consideră plagiat una dintre următoarele acţiuni:

* reproducerea exactă a cuvintelor unui alt autor, dintr-o altă lucrare, în limba română sau prin traducere dintr-o altă limbă, dacă se omit ghilimele şi referinţa precisă,
* redarea cu alte cuvinte, reformularea prin cuvinte proprii sau rezumarea ideilor din alte lucrări, dacă nu se indică sursa bibliografică,
* prezentarea unor date experimentale obţinute sau a unor aplicaţii realizate de alţi autori fără menţionarea corectă a acestor surse,
* însuşirea totală sau parţială a unei lucrări în care regulile de mai sus sunt respectate, dar care are alt autor.

Pentru evitarea acestor situaţii neplăcute se recomandă:

* plasarea între ghilimele a citatelor directe şi indicarea referinţei într-o listă corespunzătoare la sfărşitul lucrării,
* indicarea în text a reformulării unei idei, opinii sau teorii şi corespunzător în lista de referinţe a sursei originale de la care s-a făcut preluarea,
* precizarea sursei de la care s-au preluat date experimentale, descrieri tehnice, figuri, imagini, statistici, tabele et caetera,
* precizarea referinţelor poate fi omisă dacă se folosesc informaţii sau teorii arhicunoscute, a căror paternitate este unanim cunoscută și acceptată.

Data, Semnătura candidatului,

|  |  |  |
| --- | --- | --- |
|  | UNIVERSITATEA DIN CRAIOVA  Facultatea de Automatică, Calculatoare şi Electronică  Departamentul de Calculatoare și Tehnologia Informației | Aprobat la data de  …………………  Şef de departament,  Prof. dr. ing.  Marius BREZOVAN |

**PROIECTUL DE DIPLOMĂ**

|  |  |
| --- | --- |
| Numele și prenumele studentului/-ei: | Bochiș Florian Celestin |
| Enunțul temei: | “Support Application for International Support Mobilities” este o aplicație web pentru studenți și profesori prin intermediul căreia se urmărește simplificarea procesului de înregistrare în cadrul unui contract de studiu într-o țară străină. |
| Datele de pornire: | Studiul Java; Apache Maven; Apache Tomcat server; MySQL database; Hibernate; Bootstrap; AngularJS; REST Services; |
| Conținutul proiectului: | Introduction – Application concepts, Purpose, Motivation.  Technologies and frameworks - Theoretical aspects.  System Design and Implementation of the project.  The Application (User Manual). |
| Material grafic obligatoriu: | Prezentare PowerPoint, Figuri, Print-Screen-uri, Cod Sursa. |
| Consultații: | Periodice |
| Conducătorul științific  (titlul, nume și prenume, semnătura): | Şef lucrări dr. ing. Marius Marian Adrian |
| Data eliberării temei: | 26.01.2016 |
| Termenul estimat de predare a proiectului: | 18.07.2016 |
| Data predării proiectului de către student și semnătura acestuia: | 15.07.2016 |

|  |  |  |
| --- | --- | --- |
|  | UNIVERSITATEA DIN CRAIOVA  Facultatea de Automatică, Calculatoare şi Electronică  Departamentul de Calculatoare și Tehnologia Informației |  |

**REFERATUL CONDUCĂTORULUI ȘTIINȚIFIC**

|  |  |
| --- | --- |
| Numele și prenumele candidatului/-ei: | **Bochiș Florian Celestin** |
| Specializarea: | Calculatoare în limba engleză |
| Titlul proiectului: | “Support Application for International Support Mobilities” |
| Locația în care s-a realizat practica de documentare (se bifează una sau mai multe din opțiunile din dreapta): | În facultate □ |
| În producție □ |
| În cercetare □ |
| Altă locație: [*se detaliază*] |

În urma analizei lucrării candidatului au fost constatate următoarele:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nivelul documentării | | Insuficient  □ | Satisfăcător □ | Bine  □ | Foarte bine  □ |
| Tipul proiectului | | Cercetare  □ | Proiectare  □ | Realizare practică □ | Altul  [*se detaliază*] |
| Aparatul matematic utilizat | | Simplu  □ | Mediu  □ | Complex □ | Absent  □ |
| Utilitate | | Contract de cercetare □ | Cercetare internă □ | Utilare  □ | Altul  [*se detaliază*] |
| Redactarea lucrării | | Insuficient  □ | Satisfăcător □ | Bine  □ | Foarte bine  □ |
| Partea grafică, desene | | Insuficientă  □ | Satisfăcătoare □ | Bună  □ | Foarte bună  □ |
| Realizarea practică | Contribuția autorului | Insuficientă  □ | Satisfăcătoare □ | Mare  □ | Foarte mare  □ |
| Complexitatea  temei | Simplă  □ | Medie  □ | Mare  □ | Complexă  □ |
| Analiza cerințelor | Insuficient  □ | Satisfăcător □ | Bine  □ | Foarte bine  □ |
| Arhitectura | Simplă  □ | Medie  □ | Mare  □ | Complexă  □ |
| Întocmirea specificațiilor funcționale | Insuficientă  □ | Satisfăcătoare □ | Bună  □ | Foarte bună  □ |
| Implementarea | Insuficientă  □ | Satisfăcătoare □ | Bună  □ | Foarte bună  □ |
| Testarea | Insuficientă  □ | Satisfăcătoare □ | Bună  □ | Foarte bună  □ |
| Funcționarea | Da  □ | Parțială  □ | Nu  □ | |
| Rezultate experimentale | | Experiment propriu  □ | | Preluare din bibliografie  □ | |
| Bibliografie | | Cărți | Reviste | Articole | Referințe web |
| Comentarii  și  observații | |  | | | |

În concluzie, se propune:

|  |  |
| --- | --- |
| ADMITEREA PROIECTULUI  □ | RESPINGEREA PROIECTULUI  □ |

Data, Semnătura conducătorului științific,

**PROJECT SUMMARY**

“Support Application for International Support Mobilities” is a web application created for both students and professors, which simplifies the process of registration for studying abroad. The application offers account creation, account management, contract creation, contract visualization, data uploading, data visualization, news adding and the possibility to visualize an entirely institution architecture with faculties, departments and sections providing external links and data about them.

In the developing process the following languages were used:

* Java programming language - on the server side.
* HTML 5, Bootstrap 3, jQuery, CSS and AngularJS - on the client side.

In the developing process the following tools and api-s were used:

* Spring tool suite (eclipse based IDE)
* Spring framework
* MySql Database
* Hibernate ORM
* Apache Maven
* Apache Tomcat 7 Server
* JAX-RS / Jersey REST services
* TinyMCE
* ABN Tree

The application is constructed using the 3 Tier Architecture pattern dividing the application into three parts: data tier, logic tier and presentation tier.

The application has have 5 levels of access:

1) Guest: Doesn’t require registration, can visualize an institution architecture and all the details of it, read news.

2) BRI: Can visualize the student uploads, add contracts, visualize all contracts and add news.

3) Coordinator: Can accept contracts of students that applied to his/her institution, visualize the uploads of a student in the same institution, add news.

4) Student: Can upload documents, apply to a contract, confirm contract.

5) Administrator: Can visualize the accounts of others and can delete them if they affect the application in any way, is responsible with the application email address and can send informative emails through it.

\* All other roles have the guest capabilities

***Key Words***: Study abroad, Support Application, BRI, Student, Java, Bootstrap, HTML, AngularJS, 3 Tier Architecture.

**Acknowledgements**

I acknowledge with gratitude my indebtness to Professor Marius Marian Adrian for giving me the opportunity to work on this project and gave valuable guidance for preparing this project.

I would like to thank those people who directly or indirectly helped me to enhance my practical knowledge in the field of software programming and express my sincere gratitude to all those who share valuable thoughts with me.

This is being my first effort, the possibilities of errors and omissions in its contents and presentation cannot be completely ruled out. I shall, however, be very grateful to my teachers, colleagues and other readers from their suggestions for its improvement.

CONTENTS

[**Chapter 1. INTRODUCTION** 15](#_Toc456319438)

[1.1 Purpose 15](#_Toc456319439)

[1.2 Motivation 15](#_Toc456319440)

[**Chapter 2. TECHNOLOGIES AND FRAMEWORKS** 16](#_Toc456319441)

[2.1 Java (programming language) 16](#_Toc456319442)

[2.1.1 Principles 16](#_Toc456319443)

[2.1.2 Versions 17](#_Toc456319444)

[2.1.3 Java platform 17](#_Toc456319445)

[2.1.4 Performance 18](#_Toc456319446)

[2.1.5 Automatic memory management 18](#_Toc456319447)

[2.1.6 Criticism 20](#_Toc456319448)

[2.2 MySQL Database 21](#_Toc456319449)

[2.2.1 Features 22](#_Toc456319450)

[2.3 Hibernate ORM 24](#_Toc456319451)

[2.4 Apache Maven 27](#_Toc456319452)

[2.5 Apache Tomcat 29](#_Toc456319453)

[2.6 JAX-RS / Jersey Rest Services 31](#_Toc456319454)

[2.6.1 JAX-RS 31](#_Toc456319455)

[2.6.2 Jersey 32](#_Toc456319456)

[2.7 Spring tool suite 33](#_Toc456319457)

[2.8 Spring framework 34](#_Toc456319459)

[2.8.1 Inversion of control container (dependency injection) 35](#_Toc456319460)

[2.8.2 Data access framework 36](#_Toc456319461)

[2.8.3 Transaction management framework 38](#_Toc456319462)

[2.9 HTML 5 39](#_Toc456319463)

[2.10 Bootstrap 3 42](#_Toc456319464)

[2.11 jQuery 44](#_Toc456319465)

[2.12 Cascading Style Sheets 46](#_Toc456319466)

[2.13 AngularJS 49](#_Toc456319467)

[2.14 TinyMCE 51](#_Toc456319468)

[**Chapter 3. SYSTEM DESIGN AND IMPLEMENTATION** 52](#_Toc456319469)

[3.1 System requirements 52](#_Toc456319470)

[3.2 Use case diagram 54](#_Toc456319471)

[3.3 System Architecture 55](#_Toc456319472)

[3.3.1 Three tier architecture 55](#_Toc456319473)

[3.4 Maven application structure 57](#_Toc456319474)

[3.4.1 Persistence module 58](#_Toc456319475)

[3.4.2 Service module 64](#_Toc456319476)

[3.4.3 Web Service module 67](#_Toc456319477)

[3.4.4 E-mail confirmation 71](#_Toc456319478)

[3.4.5 Account Recovering 72](#_Toc456319479)

[3.4.6 Application security 74](#_Toc456319480)

[**Chapter 4. Conclusions** 77](#_Toc456319481)

[**Chapter 5. Annex** 78](#_Toc456319482)

[5.1 User Manual 78](#_Toc456319483)

[**Chapter 6. References** 83](#_Toc456319484)

FIGURES LIST

Figure 2.1 Java virtual machine code execution

Figure: 2.2 Java virtual machine specs

Figure 2.3 Hibernate layers

Figure 2.4 Spring framework components

Figure 2.5 Dependency injection

Figure 2.6 HTML structure

Figure 2.7 HTML5 API-s taxonomy and status

Figure 2.8 jQuery Dom selecting

Figure 2.10 CSS declaration block

Figure 2.11 AngularJS features

Figure 2.12 TinyMCE features

Figure 3.1 Use-case diagram

Figure 3.2 Three tier architecture

Figure 3.3 Entities Class Diagram

Figure 3.4 Dao-s class diagram

Figure 3.5 Enhanced entity–relationship model

# INTRODUCTION

## Purpose

The main purpose of “Support Application for International Support Mobilities” is to enhance student mobility through online support. It offers to students a platform where they can easily see and analyze the options they got if they want to study abroad for a period of time. Moreover it aims to simplify the work of coordinators by allowing them to check online if a student is eligible for a contract.

## Motivation

A web-based application is any application that uses a website as the interface. Users access the application from any computer connected to the Internet using a standard browser, instead of using an application that has been installed on their local computer.

Unlike traditional applications, web systems are accessible anytime, anywhere, via a PC with an Internet connection, putting the user in charge of where and when they access the application.

The user interface of web-based applications is easy to customize, making it easier to update the look and feel of the application, or to customize the presentation of information to different users.

In addition to customizing content for specific users, content can also be customized for presentation on any device connected to the internet, including gadgets like mobile phones, smart TVs, consoles, PDAs etc. further extending the user’s ability to receive and interact with information.

Accessible anywhere, easily customizable, accessible for a range of devices these are the main reasons that motivated me to build a web application for my final year project.

The need for a platform where students can easy observe and apply to contracts for studying abroad, made me to choose “Support Application for International Support Mobilities” as a subject for my web application.

# Java_logoTECHNOLOGIES AND FRAMEWORKS

## Java (programming language)

“Java is a general-purpose computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture. As of 2016, Java is one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by James Gosling at Sun Microsystems (which has since been acquired by Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun relicensed most of its Java technologies under the GNU General Public License. Others have also developed alternative implementations of these Sun technologies, such as the GNU Compiler for Java (bytecode compiler), GNU Classpath (standard libraries), and IcedTea-Web (browser plugin for applets).

The latest version is Java 8, which is the only version currently supported for free by Oracle, although earlier versions are supported both by Oracle and other companies on a commercial basis.

### Principles

There were five primary goals in the creation of the Java language:

* It must be "simple, object-oriented, and familiar".
* It must be "robust and secure".
* It must be "architecture-neutral and portable".
* It must execute with "high performance".
* It must be "interpreted, threaded, and dynamic".

### Versions

As of 2015, only Java 8 is supported ("publicly"). Major release versions of Java, along with their release dates:

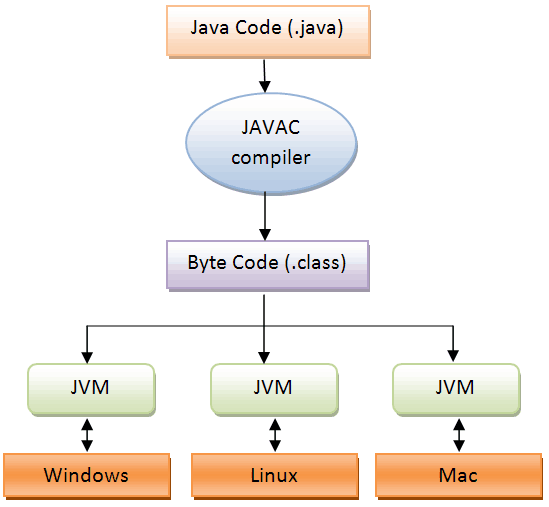
* JDK 1.0 (January 21, 1996)
* JDK 1.1 (February 19, 1997)
* J2SE 1.2 (December 8, 1998)
* J2SE 1.3 (May 8, 2000)
* J2SE 1.4 (February 6, 2002)
* J2SE 5.0 (September 30, 2004)
* Java SE 6 (December 11, 2006)
* Java SE 7 (July 28, 2011)
* Java SE 8 (March 18, 2014)

### Java platform

One design goal of Java is portability, which means that programs written for the Java platform must run similarly on any combination of hardware and operating system with adequate runtime support. This is achieved by compiling the Java language code to an intermediate representation called Java bytecode, instead of directly to architecture-specific machine code. Java bytecode instructions are analogous to machine code, but they are intended to be executed by a virtual machine (VM) written specifically for the host hardware. End users commonly use a Java Runtime Environment (JRE) installed on their own machine for standalone Java applications, or in a web browser for Java applets.

Standard libraries provide a generic way to access host-specific features such as graphics, threading, and networking.

The use of universal bytecode makes porting simple. However, the overhead of interpreting bytecode into machine instructions makes interpreted programs almost always run more slowly than native executables. However, just-in-time (JIT) compilers that compile bytecodes to machine code during runtime were introduced from an early stage. Java itself is platform-independent, and is adapted to the particular platform it is to run on by a Java virtual machine for it, which translates the Java bytecode into the platform's machine language.



**Figure 2.1 Java virtual machine code execution**

### Performance

Programs written in Java have a reputation for being slower and requiring more memory than those written in C++. However, Java programs' execution speed improved significantly with the introduction of just-in-time compilation in 1997/1998 for Java 1.1, the addition of language features supporting better code analysis (such as inner classes, the StringBuilder class, optional assertions, etc.), and optimizations in the Java virtual machine, such as HotSpot becoming the default for Sun's JVM in 2000. With Java 1.5, the performance was improved with the addition of the java.util.concurrent package, including Lock free implementations of the ConcurrentMaps and other multi-core collections, and it was improved further Java 1.6.

Some platforms offer direct hardware support for Java; there are microcontrollers that can run Java in hardware instead of a software Java virtual machine, and ARM based processors can have hardware support for executing Java bytecode through their Jazelle option (while its support is mostly dropped in current implementations of ARM).

### Automatic memory management

Java uses an automatic garbage collector to manage memory in the object lifecycle. The programmer determines when objects are created, and the Java runtime is responsible for recovering the memory once objects are no longer in use. Once no references to an object remain, the unreachable memory becomes eligible to be freed automatically by the garbage collector. Something similar to a memory leak may still occur if a programmer's code holds a reference to an object that is no longer needed, typically when objects that are no longer needed are stored in containers that are still in use. If methods for a nonexistent object are called, a "null pointer exception" is thrown.

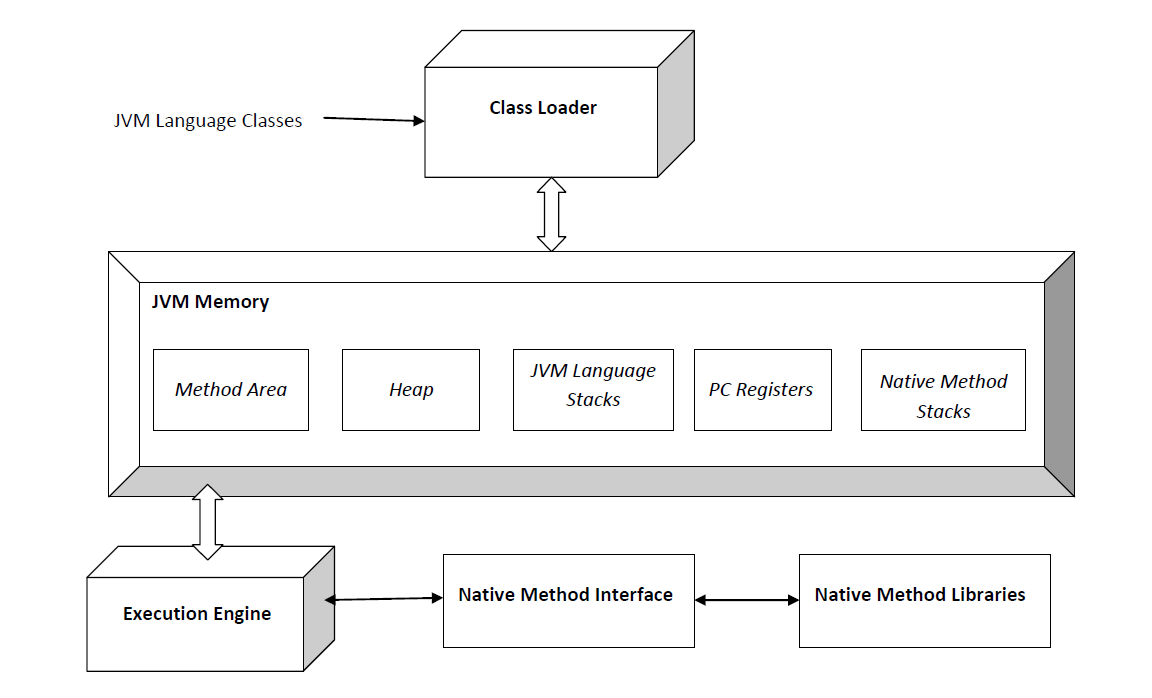
One of the ideas behind Java's automatic memory management model is that programmers can be spared the burden of having to perform manual memory management. In some languages, memory for the creation of objects is implicitly allocated on the stack, or explicitly allocated and deallocated from the heap. In the latter case the responsibility of managing memory resides with the programmer. If the program does not deallocate an object, a memory leak occurs. If the program attempts to access or deallocate memory that has already been deallocated, the result is undefined and difficult to predict, and the program is likely to become unstable and/or crash. This can be partially remedied by the use of smart pointers, but these add overhead and complexity. Note that garbage collection does not prevent "logical" memory leaks, i.e., those where the memory is still referenced but never used.

Garbage collection may happen at any time. Ideally, it will occur when a program is idle. It is guaranteed to be triggered if there is insufficient free memory on the heap to allocate a new object; this can cause a program to stall momentarily. Explicit memory management is not possible in Java.

Java does not support C/C++ style pointer arithmetic, where object addresses and unsigned integers (usually long integers) can be used interchangeably. This allows the garbage collector to relocate referenced objects and ensures type safety and security.

As in C++ and some other object-oriented languages, variables of Java's primitive data types are either stored directly in fields (for objects) or on the stack (for methods) rather than on the heap, as is commonly true for non-primitive data types (but see escape analysis). This was a conscious decision by Java's designers for performance reasons.

Java contains multiple types of garbage collectors. By default, HotSpot uses the parallel scavenge garbage collector. However, there are also several other garbage collectors that can be used to manage the heap. For 90% of applications in Java, the Concurrent Mark-Sweep (CMS) garbage collector is sufficient. Oracle aims to replace CMS with the Garbage-First collector (G1).



**Figure: 2.2 Java virtual machine specs**

### Criticism

Criticisms directed at Java include the implementation of generics, speed, the handling of unsigned numbers, the implementation of floating-point arithmetic, and a history of security vulnerabilities in the primary Java VM implementation HotSpot.” [[1]](#endnote-1)

## C:\Users\lovit\Desktop\logo-mysql-170x115.pngMySQL Database

“MySQL is an open-source relational database management system (RDBMS). In July 2013, it was the world's second most widely used RDBMS, and the most widely used open-source client–server model RDBMS. Its name is a combination of "My", the name of co-founder Michael Widenius' daughter, and "SQL", the abbreviation for Structured Query Language. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation. For proprietary use, several paid editions are available, and offer additional functionality.

MySQL is a popular choice of database for use in web applications, and is a central component of the widely used LAMP open-source web application software stack (and other "AMP" stacks). LAMP is an acronym for Linux, Apache, MySQL, Perl/PHP/Python. Free-software open-source projects that require a full-featured database management system often use MySQL. Applications that use the MySQL database include: TYPO3, MODx, Joomla, WordPress, phpBB, MyBB, Drupal and other software. MySQL is also used in many high-profile, large-scale websites, including Google (though not for searches), Facebook, Twitter, Flickr, and YouTube.

On all platforms except Windows, MySQL ships with no GUI tools to administer MySQL databases or manage data contained within the databases. Users may use the included command line tools, or install MySQL Workbench via a separate download. Many third party GUI tools are also available.

MySQL is written in C and C++. Its SQL parser is written in yacc, but it uses a home-brewed lexical analyzer. MySQL works on many system platforms, including AIX, BSDi, FreeBSD, HP-UX, eComStation, i5/OS, IRIX, Linux, OS X, Microsoft Windows, NetBSD, Novell NetWare, OpenBSD, OpenSolaris, OS/2 Warp, QNX, Oracle Solaris, Symbian, SunOS, SCO OpenServer, SCO UnixWare, Sanos and Tru64. A port of MySQL to OpenVMS also exists.

The MySQL server software itself and the client libraries use dual-licensing distribution. They are offered under GPL version 2, beginning from 28 June 2000 (which in 2009 has been extended with a FLOSS License Exception) or to use a proprietary license.

Support can be obtained from the official manual. Free support additionally is available in different IRC channels and forums. Oracle offers paid support via its MySQL Enterprise products. They differ in the scope of services and in price. Additionally, a number of third party organisations exist to provide support and services, including MariaDB and Percona.

MySQL has received positive reviews, and reviewers noticed it performs extremely well in the average case and that the developer interfaces are there, and the documentation (not to mention feedback in the real world via Web sites and the like) is very, very good. It has also been tested to be a fast, stable and true multi-user, multi-threaded sql database server.

### Features

MySQL is offered under two different editions: the open source MySQL Community Server and the proprietary Enterprise Server. MySQL Enterprise Server is differentiated by a series of proprietary extensions which install as server plugins, but otherwise shares the version numbering system and is built from the same code base.

Major features as available in MySQL 5.6:

* A broad subset of ANSI SQL 99, as well as extensions
* Cross-platform support
* Stored procedures, using a procedural language that closely adheres to SQL/PSM
* Triggers
* Cursors
* Updatable views
* Online DDL when using the InnoDB Storage Engine.
* Information schema
* Performance Schema that collects and aggregates statistics about server execution and query performance for monitoring purposes.
* A set of SQL Mode options to control runtime behavior, including a strict mode to better adhere to SQL standards.
* X/Open XA distributed transaction processing (DTP) support; two phase commit as part of this, using the default InnoDB storage engine
* Transactions with savepoints when using the default InnoDB Storage Engine. The NDB Cluster Storage Engine also supports transactions.
* ACID compliance when using InnoDB and NDB Cluster Storage Engines
* SSL support
* Query caching
* Sub-SELECTs (i.e. nested SELECTs)
* Built-in Replication support (i.e. Master-Master Replication & Master-Slave Replication) with one master per slave, many slaves per master.Multi-master replication is provided in MySQL Cluster, and multi-master support can be added to unclustered configurations using Galera Cluster.
* Full-text indexing and searching
* Embedded database library
* Unicode support
* Partitioned tables with pruning of partitions in optimizer
* Shared-nothing clustering through MySQL Cluster
* Multiple storage engines, allowing one to choose the one that is most effective for each table in the application.
* Native storage engines InnoDB, MyISAM, Merge, Memory (heap), Federated, Archive, CSV, Blackhole, NDB Cluster.
* Commit grouping, gathering multiple transactions from multiple connections together to increase the number of commits per second.
* The developers release minor updates of the MySQL Server approximately every two months. The sources can be obtained from MySQL's website or from MySQL's GitHub repository, both under the GPL license.” [[2]](#endnote-2)

## Hibernate ORM



“Hibernate ORM is concerned with helping your application to achieve persistence. So what is persistence? Persistence simply means that we would like our application’s data to outlive the applications process. In Java terms, we would like the state of (some of) our objects to live beyond the scope of the JVM so that the same state is available later.

Specifically, Hibernate ORM is concerned with data persistence as it applies to relational databases (RDBMS). In the world of Object-Oriented applications, there is often a discussion about using an object database (ODBMS) as opposed to a RDBMS. We are not going to explore that discussion here. Suffice it to say that RDBMS remain a very popular persistence mechanism and will so for the foreseeable future.

'Object-Relational Impedance Mismatch' (sometimes called the 'paradigm mismatch') is just a fancy way of saying that object models and relational models do not work very well together. RDBMSs represent data in a tabular format (a spreadsheet is a good visualization for those not familiar with RDBMSs), whereas object-oriented languages, such as Java, represent it as an interconnected graph of objects. Loading and storing graphs of objects using a tabular relational database exposes us to 5 mismatch problems:

* Granularity

Sometimes you will have an object model which has more classes than the number of corresponding tables in the database (we says the object model is more granular than the relational model). Take for example the notion of an Address

* Subtypes (inheritance)

Inheritance is a natural paradigm in object-oriented programming languages. However, RDBMSs do not define anything similar on the whole (yes some databases do have subtype support but it is completely non-standardized)

* Identity

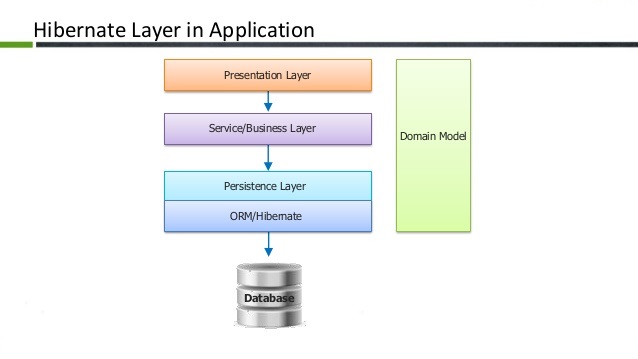
A RDBMS defines exactly one notion of 'sameness': the primary key. Java, however, defines both object identity a==b and object equality a.equals(b).

* Associations

Associations are represented as unidirectional references in Object Oriented languages whereas RDBMSs use the notion of foreign keys. If you need bidirectional relationships in Java, you must define the association twice. Likewise, you cannot determine the multiplicity of a relationship by looking at the object domain model.

* Data navigation

The way you access data in Java is fundamentally different than the way you do it in a relational database. In Java, you navigate from one association to an other walking the object network. This is not an efficient way of retrieving data from a relational database. You typically want to minimize the number of SQL queries and thus load several entities via JOINs and select the targeted entities before you start walking the object network.



**Figure 2.3 Hibernate layers**

In addition to its own "native" API, Hibernate is also an implementation of the Java Persistence API (JPA) specification. As such, it can be easily used in any environment supporting JPA including Java SE applications, Java EE application servers, Enterprise OSGi containers, etc. Hibernate enables you to develop persistent classes following natural Object-oriented idioms including inheritance, polymorphism, association, composition, and the Java collections framework. Hibernate requires no interfaces or base classes for persistent classes and enables any class or data structure to be persistent.

Hibernate supports lazy initialization, numerous fetching strategies and optimistic locking with automatic versioning and time stamping. Hibernate requires no special database tables or fields and generates much of the SQL at system initialization time instead of at runtime. Hibernate consistently offers superior performance over straight JDBC code, both in terms of developer productivity and runtime performance.

Hibernate was designed to work in an application server cluster and deliver a highly scalable architecture. Hibernate scales well in any environment: Use it to drive your in-house Intranet that serves hundreds of users or for mission-critical applications that serve hundreds of thousands.

Hibernate is well known for its excellent stability and quality, proven by the acceptance and use by tens of thousands of Java developers. Hibernate is highly configurable and extensible.” [[3]](#endnote-3)

## Apache Maven



“Maven, a Yiddish word meaning accumulator of knowledge, was originally started as an attempt to simplify the build processes in the Jakarta Turbine project. There were several projects each with their own Ant build files that were all slightly different and JARs were checked into CVS. Maven provides a standard way to build the projects, a clear definition of what the project consisted of, an easy way to publish project information and a way to share JARs across several projects.

The result is a tool that can now be used for building and managing any Java-based project.Maven makes the day-to-day work of Java developers easier and generally help with the comprehension of any Java-based project.

Maven’s primary goal is to allow a developer to comprehend the complete state of a development effort in the shortest period of time. In order to attain this goal there are several areas of concern that Maven attempts to deal with:

* Making the build process easy
* Providing a uniform build system
* Providing quality project information
* Providing guidelines for best practices development
* Allowing transparent migration to new features

While using Maven doesn’t eliminate the need to know about the underlying mechanisms, Maven does provide a lot of shielding from the details. Maven allows a project to build using its project object model (POM) and a set of plugins that are shared by all projects using Maven, providing a uniform build system. Once you familiarize yourself with how one Maven project builds you automatically know how all Maven projects build saving you immense amounts of time when trying to navigate many projects.

Maven provides plenty of useful project information that is in part taken from your POM and in part generated from your project’s sources. For example, Maven can provide:

* Change log document created directly from source control
* Cross referenced sources
* Mailing lists
* Dependency list
* Unit test reports including coverage

As Maven improves the information set provided will improve, all of which will be transparent to users of Maven. Other products can also provide Maven plugins to allow their set of project information alongside some of the standard information given by Maven, all still based on the POM.

Maven aims to gather current principles for best practices development, and make it easy to guide a project in that direction. For example, specification, execution, and reporting of unit tests are part of the normal build cycle using Maven. Current unit testing best practices were used as guidelines:

* Keeping your test source code in a separate, but parallel source tree
* Using test case naming conventions to locate and execute tests
* Have test cases setup their environment and don’t rely on customizing the build for test preparation.

Maven also aims to assist in project workflow such as release management and issue tracking. Maven also suggests some guidelines on how to layout your project’s directory structure so that once you learn the layout you can easily navigate any other project that uses Maven and the same defaults.

Maven provides an easy way for Maven clients to update their installations so that they can take advantage of any changes that been made to Maven itself. Installation of new or updated plugins from third parties or Maven itself has been made trivial for this reason.

You may have heard some of the following things about Maven:

* Maven is a site and documentation tool
* Maven extends Ant to let you download dependencies
* Maven is a set of reusable Ant scriptlets

While Maven does these things, these are not the only features Maven has, and its objectives are quite different.

Maven does encourage best practices, but we realise that some projects may not fit with these ideals for historical reasons. While Maven is designed to be flexible, to an extent, in these situations and to the needs of different projects, it can not cater to every situation without making compromises to the integrity of its objectives.” [[4]](#endnote-4)

## Apache Tomcat



“Apache Tomcat, often referred to as Tomcat, is an open-source web server developed by the Apache Software Foundation (ASF). Tomcat implements several Java EE specifications including Java Servlet, JavaServer Pages (JSP), Java EL, and WebSocket, and provides a "pure Java" HTTP web server environment in which Java code can run. Tomcat is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation, released under the Apache License 2.0 license, and is open-source software.

Components:

Tomcat 4.x was released with Catalina (a servlet container), Coyote (an HTTP connector) and Jasper (a JSP engine).

* Catalina is Tomcat's servlet container. Catalina implements Sun Microsystems' specifications for servlet and JavaServer Pages (JSP). In Tomcat, a Realm element represents a "database" of usernames, passwords, and roles (similar to Unix groups) assigned to those users. Different implementations of Realm allow Catalina to be integrated into environments where such authentication information is already being created and maintained, and then use that information to implement Container Managed Security as described in the Servlet Specification.
* Coyote is a Connector component for Tomcat that supports the HTTP 1.1 protocol as a web server. This allows Catalina, nominally a Java Servlet or JSP container, to also act as a plain web server that serves local files as HTTP documents. Coyote listens for incoming connections to the server on a specific TCP port and forwards the request to the Tomcat Engine to process the request and send back a response to the requesting client. Another Coyote Connector, Coyote JK, listens similarly but instead forwards its requests to another web server, such as Apache, using the JK protocol. This usually offers better performance.
* Jasper is Tomcat's JSP Engine. Jasper parses JSP files to compile them into Java code as servlets (that can be handled by Catalina). At runtime, Jasper detects changes to JSP files and recompiles them.

Three new components were added with the release of Tomcat 7:

-Cluster has been added to manage large applications. It is used for load balancing that can be achieved through many techniques. Clustering support currently requires the JDK version 1.5 or later.

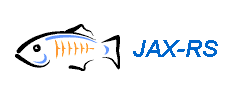
- A high-availability feature has been added to facilitate the scheduling of system upgrades (e.g. new releases, change requests) without affecting the live environment. This is done by dispatching live traffic requests to a temporary server on a different port while the main server is upgraded on the main port. It is very useful in handling user requests on high-traffic web applications.

- It has also added user- as well as system-based web applications enhancement to add support for deployment across the variety of environments. It also tries to manage sessions as well as applications across the network.

Tomcat is building additional components. A number of additional components may be used with Apache Tomcat. These components may be built by users should they need them or they can be downloaded from one of the mirrors.

Tomcat 7.x implements the Servlet 3.0 and JSP 2.2 specifications. It requires Java version 1.6, although previous versions have run on Java 1.1 through 1.5. Versions 5 through 6 saw improvements in garbage collection, JSP parsing, performance and scalability. Native wrappers, known as "Tomcat Native", are available for Microsoft Windows and Unix for platform integration.” [[5]](#endnote-5)

## JAX-RS / Jersey Rest Services



### JAX-RS

“JAX-RS: Java API for RESTful Web Services (JAX-RS) is a Java programming language API that provides support in creating web services according to the Representational State Transfer (REST) architectural pattern. JAX-RS uses annotations, introduced in Java SE 5, to simplify the development and deployment of web service clients and endpoints.

From version 1.1 on, JAX-RS is an official part of Java EE 6. A notable feature of being an official part of Java EE is that no configuration is necessary to start using JAX-RS. For non-Java EE 6 environments a (small) entry in the web.xml deployment descriptor is required.

JAX-RS provides some annotations to aid in mapping a resource class (a POJO) as a web resource. The annotations include:

@Path specifies the relative path for a resource class or method.

@GET, @PUT, @POST, @DELETE and @HEAD specify the HTTP request type of a resource.

@Produces specifies the response Internet media types (used for content negotiation).

@Consumes specifies the accepted request Internet media types.

In addition, it provides further annotations to method parameters to pull information out of the request. All the @ \*Param annotations take a key of some form which is used to look up the value required.

@PathParam binds the method parameter to a path segment.

@QueryParam binds the method parameter to the value of an HTTP query parameter.

@MatrixParam binds the method parameter to the value of an HTTP matrix parameter.

@HeaderParam binds the method parameter to an HTTP header value.

@CookieParam binds the method parameter to a cookie value.

@FormParam binds the method parameter to a form value.

@DefaultValue specifies a default value for the above bindings when the key is not found.

@Context returns the entire context of the object (for example @Context HttpServletRequest request).” [[6]](#endnote-6)

### Jersey



“Developing RESTful Web services that seamlessly support exposing your data in a variety of representation media types and abstract away the low-level details of the client-server communication is not an easy task without a good toolkit. In order to simplify development of RESTful Web services and their clients in Java, a standard and portable JAX-RS API has been designed. Jersey RESTful Web Services framework is open source, production quality, framework for developing RESTful Web Services in Java that provides support for JAX-RS APIs and serves as a JAX-RS (JSR 311 & JSR 339) Reference Implementation.

Jersey framework is more than the JAX-RS Reference Implementation. Jersey provides it’s own API that extend the JAX-RS toolkit with additional features and utilities to further simplify RESTful service and client development. Jersey also exposes numerous extension SPIs so that developers may extend Jersey to best suit their needs.

Goals of Jersey project can be summarized in the following points:

* Track the JAX-RS API and provide regular releases of production quality Reference Implementations that ships with GlassFish;
* Provide APIs to extend Jersey & Build a community of users and developers; and finally
* Make it easy to build RESTful Web services utilising Java and the Java Virtual Machine.” [[7]](#endnote-7)

## Spring tool suite



“The Spring Tool Suite is an Eclipse-based development environment that is customized for developing Spring applications. It provides a ready-to-use environment to implement, debug, run, and deploy your Spring applications, including integrations for Pivotal tc Server, Pivotal Cloud Foundry, Git, Maven, AspectJ, and comes on top of the latest Eclipse releases.

Included with the Spring Tool Suite is the developer edition of Pivotal tc Server, the drop-in replacement for Apache Tomcat that's optimized for Spring. With its Spring Insight console, tc Server Developer Edition provides a graphical real-time view of application performance metrics that lets developers identify and diagnose problems from their desktops.

The Spring Tool suite supports application targeting to local, virtual and cloud-based servers. It is freely available for development and internal business operations use with no time limits, fully open-source and licensed under the terms of the Eclipse Public License.” [[8]](#endnote-8)

## 

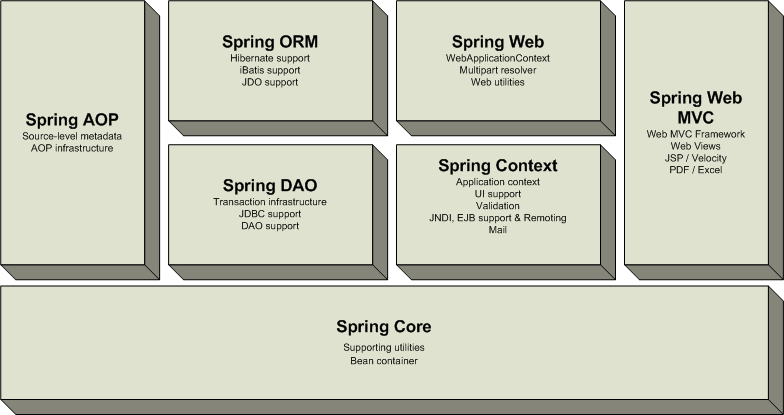


## Spring framework

“The Spring Framework is an application framework and inversion of control container for the Java platform. The framework's core features can be used by any Java application, but there are extensions for building web applications on top of the Java EE platform. Although the framework does not impose any specific programming model, it has become popular in the Java community as an alternative to, replacement for, or even addition to the Enterprise JavaBeans (EJB) model. The Spring Framework is open source.

The Spring Framework includes several modules that provide a range of services:

* Spring Core Container: this is the base module of Spring and provides spring containers (BeanFactory and ApplicationContext).
* Aspect-oriented programming: enables implementing cross-cutting concerns.
* Authentication and authorization: configurable security processes that support a range of standards, protocols, tools and practices via the Spring Security sub-project (formerly Acegi Security System for Spring).
* Convention over configuration: a rapid application development solution for Spring-based enterprise applications is offered in the Spring Roo module
* Data access: working with relational database management systems on the Java platform using JDBC and object-relational mapping tools and with NoSQL databases
* Inversion of control container: configuration of application components and lifecycle management of Java objects, done mainly via dependency injection
* Messaging: configurative registration of message listener objects for transparent message-consumption from message queues via JMS, improvement of message sending over standard JMS APIs
* Model–view–controller: an HTTP- and servlet-based framework providing hooks for extension and customization for web applications and RESTful Web services.
* Remote access framework: configurative RPC-style marshalling of Java objects over networks supporting RMI, CORBA and HTTP-based protocols including Web services (SOAP)
* Transaction management: unifies several transaction management APIs and coordinates transactions for Java objects
* Remote management: configurative exposure and management of Java objects for local or remote configuration via JMX
* Testing: support classes for writing unit tests and integration tests



**Figure 2.4 Spring framework components**

### Inversion of control container (dependency injection)

Central to the Spring Framework is its inversion of control (IoC) container, which provides a consistent means of configuring and managing Java objects using reflection. The container is responsible for managing object lifecycles of specific objects: creating these objects, calling their initialization methods, and configuring these objects by wiring them together.

Objects created by the container are also called managed objects or beans. The container can be configured by loading XML files or detecting specific Java annotations on configuration classes. These data sources contain the bean definitions that provide the information required to create the beans.

Objects can be obtained by means of either dependency lookup or dependency injection.[12] Dependency lookup is a pattern where a caller asks the container object for an object with a specific name or of a specific type. Dependency injection is a pattern where the container passes objects by name to other objects, via either constructors, properties, or factory methods.

In many cases one need not use the container when using other parts of the Spring Framework, although using it will likely make an application easier to configure and customize. The Spring container provides a consistent mechanism to configure applications and integrates with almost all Java environments, from small-scale applications to large enterprise applications.

The container can be turned into a partially compliant EJB 3.0 container by means of the Pitchfork project. Some criticize the Spring Framework for not complying with standards. However, SpringSource doesn't see EJB 3 compliance as a major goal, and claims that the Spring Framework and the container allow for more powerful programming models. You do not create an object, but describe how they should be created, by defining it in the Spring configuration file. You do not call services and components, but tell which services and components must be called, by defining them in the Spring configuration files. This makes the code easy to maintain and easier to test through IoC.



**Figure 2.5 Dependency injection**

### Data access framework

Spring's data access framework addresses common difficulties developers face when working with databases in applications. Support is provided for all popular data access frameworks in Java: JDBC, iBatis/MyBatis, Hibernate, JDO, JPA, Oracle TopLink, Apache OJB, and Apache Cayenne, among others.

For all of these supported frameworks, Spring provides these features

* Resource management - automatically acquiring and releasing database resources
* Exception handling - translating data access related exception to a Spring data access hierarchy
* Transaction participation - transparent participation in ongoing transactions
* Resource unwrapping - retrieving database objects from connection pool wrappers
* Abstraction for BLOB and CLOB handling

All these features become available when using template classes provided by Spring for each supported framework. Critics have said these template classes are intrusive and offer no advantage over using (for example) the Hibernate API directly. In response, the Spring developers have made it possible to use the Hibernate and JPA APIs directly. This however requires transparent transaction management, as application code no longer assumes the responsibility to obtain and close database resources, and does not support exception translation.

Together with Spring's transaction management, its data access framework offers a flexible abstraction for working with data access frameworks. The Spring Framework doesn't offer a common data access API; instead, the full power of the supported APIs is kept intact. The Spring Framework is the only framework available in Java that offers managed data access environments outside of an application server or container.

While using Spring for transaction management with Hibernate, the following beans may have to be configured:

* Data Source
* SessionFactory
* HibernateProperties
* TransactionManager

Other points of configuration include:

* An AOP configuration of cutting points.
* Transaction semantics of AOP advice.

### Transaction management framework

Spring's transaction management framework brings an abstraction mechanism to the Java platform. Its abstraction is capable of:

* working with local and global transactions (local transaction does not require an application server)
* working with nested transactions
* working with savepoints
* working in almost all environments of the Java platform

In comparison, JTA only supports nested transactions and global transactions, and requires an application server (and in some cases also deployment of applications in an application server).

The Spring Framework ships a PlatformTransactionManager for a number of transaction management strategies:

* Transactions managed on a JDBC Connection
* Transactions managed on Object-relational mapping Units of Work
* Transactions managed via the JTA TransactionManager and UserTransaction
* Transactions managed on other resources, like object databases

Next to this abstraction mechanism the framework also provides two ways of adding transaction management to applications:

* Programmatically, by using Spring's TransactionTemplate
* Configuratively, by using metadata like XML or Java annotations (@Transactional, etc.)

Together with Spring's data access framework — which integrates the transaction management framework — it is possible to set up a transactional system through configuration without having to rely on JTA or EJB. The transactional framework also integrates with messaging and caching engines.” [[9]](#endnote-9)

## HTML 5



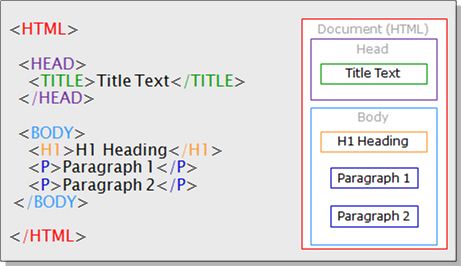
“HTML5 is a markup language used for structuring and presenting content on the World Wide Web. It is the fifth and current version of the HTML standard.

It was published in October 2014 by the World Wide Web Consortium (W3C) to improve the language with support for the latest multimedia, while keeping it both easily readable by humans and consistently understood by computers and devices such as web browsers, parsers, etc. HTML5 is intended to subsume not only HTML 4, but also XHTML 1 and DOM Level 2 HTML.

HTML5 includes detailed processing models to encourage more interoperable implementations; it extends, improves and rationalizes the markup available for documents, and introduces markup and application programming interfaces (APIs) for complex web applications.[5] For the same reasons, HTML5 is also a candidate for cross-platform mobile applications, because it includes features designed with low-powered devices in mind.

Many new syntactic features are included. To natively include and handle multimedia and graphical content, the new <video>, <audio> and <canvas> elements were added, and support for scalable vector graphics (SVG) content and MathML for mathematical formulas. To enrich the semantic content of documents, new page structure elements such as <main>, <section>, <article>, <header>, <footer>, <aside>, <nav> and <figure>, are added. New attributes are introduced, some elements and attributes have been removed, and others such as <a>, <cite> and <menu> have been changed, redefined or standardized.

The APIs and Document Object Model (DOM) are now fundamental parts of the HTML5 specification and HTML5 also better defines the processing for any invalid documents.



**Figure 2.6 HTML structure**

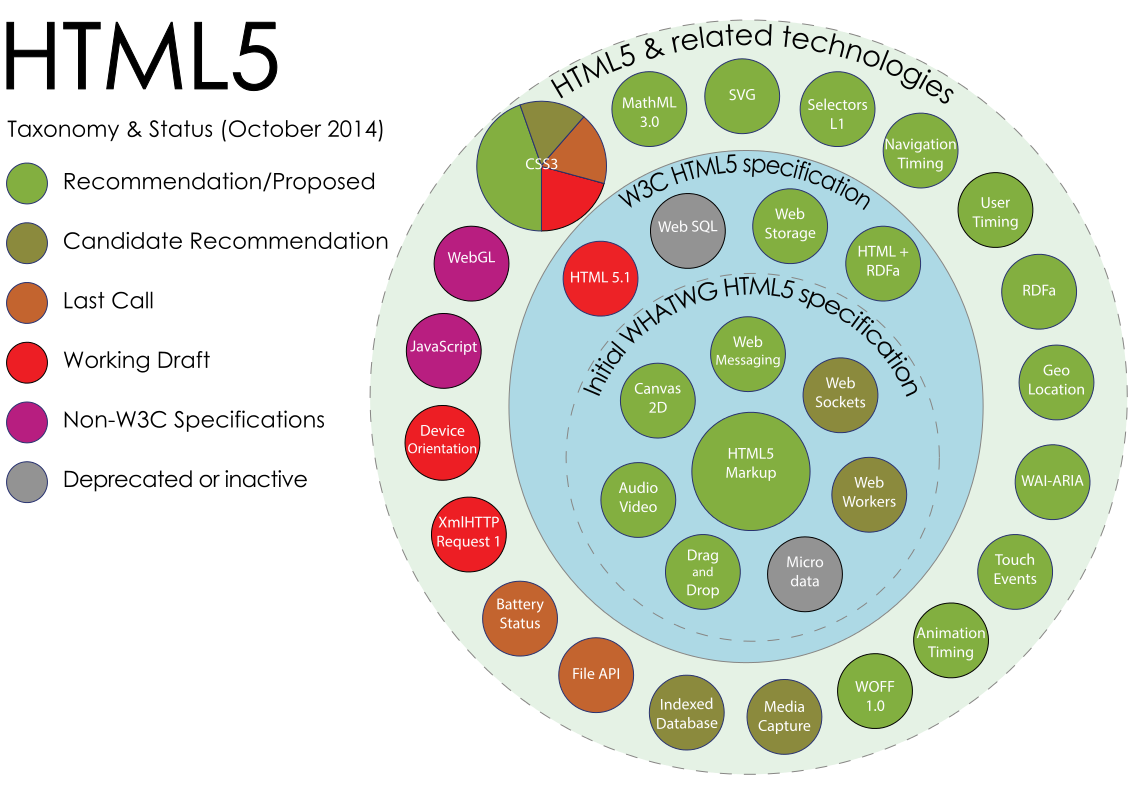
The Web Hypertext Application Technology Working Group (WHATWG) began work on the new standard in 2004. At that time, HTML 4.01 had not been updated since 2000, and the World Wide Web Consortium (W3C) was focusing future developments on XHTML 2.0. In 2009, the W3C allowed the XHTML 2.0 Working Group's charter to expire and decided not to renew it. W3C and WHATWG are currently working together on the development of HTML5.

While some features of HTML5 are often compared to Adobe Flash, the two technologies are very different. Both include features for playing audio and video within web pages, and for using Scalable Vector Graphics. However, HTML5 on its own cannot be used for animation or interactivity – it must be supplemented with CSS3 or JavaScript. There are many Flash capabilities that have no direct counterpart in HTML5. See Comparison of HTML5 and Flash.

Although HTML5 has been well known among web developers for years, its interactive capabilities became a topic of mainstream media around April 2010 after Apple Inc's then-CEO Steve Jobs issued a public letter titled "Thoughts on Flash" where he concluded that "Flash is no longer necessary to watch video or consume any kind of web content" and that "new open standards created in the mobile era, such as HTML5, will win". This sparked a debate in web development circles suggesting that, while HTML5 provides enhanced functionality, developers must consider the varying browser support of the different parts of the standard as well as other functionality differences between HTML5 and Flash. In early November 2011, Adobe announced that it would discontinue development of Flash for mobile devices and reorient its efforts in developing tools using HTML5.

HTML5 introduces elements and attributes that reflect typical usage on modern websites. Some of them are semantic replacements for common uses of generic block (<div>) and inline (<span>) elements, for example <nav> (website navigation block), <footer> (usually referring to bottom of web page or to last lines of HTML code), or <audio> and <video> instead of <object>.[34][35][36] Some deprecated elements from HTML 4.01 have been dropped, including purely presentational elements such as <font> and <center>, whose effects have long been superseded by the more capable Cascading Style Sheets.[37] There is also a renewed emphasis on the importance of DOM scripting (e.g., JavaScript) in Web behavior.

The HTML5 syntax is no longer based on SGML despite the similarity of its markup. It has, however, been designed to be backward compatible with common parsing of older versions of HTML. It comes with a new introductory line that looks like an SGML document type declaration, <!DOCTYPE html>, which triggers the standards-compliant rendering mode. Since 5 January 2009, HTML5 also includes Web Forms 2.0, a previously separate WHATWG specification.” [[10]](#endnote-10)



**Figure 2.7 HTML5 API-s taxonomy and status**

## Bootstrap 3



“Bootstrap is a free and open-source front-end web framework for designing websites and web applications. It contains HTML- and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. Unlike many web frameworks, it concerns itself with front-end development only.

Bootstrap is the second most-starred project on GitHub, with over 95 thousand stars and more than 40 thousand forks.

Bootstrap, originally named Twitter Blueprint, was developed by Mark Otto and Jacob Thornton at Twitter as a framework to encourage consistency across internal tools. Before Bootstrap, various libraries were used for interface development, which led to inconsistencies and a high maintenance burden. According to Twitter developer Mark Otto:

"A super small group of developers and I got together to design and build a new internal tool and saw an opportunity to do something more. Through that process, we saw ourselves build something much more substantial than another internal tool. Months later, we ended up with an early version of Bootstrap as a way to document and share common design patterns and assets within the company."

After a few months of development by a small group, many developers at Twitter began to contribute to the project as a part of Hack Week, a hackathon-style week for the Twitter development team. It was renamed from Twitter Blueprint to Bootstrap, and released as an open source project on August 19, 2011. It has continued to be maintained by Mark Otto, Jacob Thornton, and a small group of core developers, as well as a large community of contributors.

On January 31, 2012, Bootstrap 2 was announced. This release added the twelve-column grid layout and responsive design components, as well as changes to many of the existing components. The Bootstrap 3 release was announced on 19 August 2013, moving to a mobile first approach and using a flat design.

On October 29, 2014, Mark Otto announced Bootstrap 4 was in development. The first alpha version of Bootstrap 4 was deployed on August 19, 2015.

Bootstrap is compatible with the latest versions of the Google Chrome, Firefox, Internet Explorer, Opera, and Safari browsers, although some of these browsers are not supported on all platforms. Since version 2.0 it also supports responsive web design. This means the layout of web pages adjusts dynamically, taking into account the characteristics of the device used (desktop, tablet, mobile phone). Starting with version 3.0, Bootstrap adopted a mobile-first design philosophy, emphasizing responsive design by default. The version 4.0 alpha release added Sass and Flexbox support.

Bootstrap is modular and consists essentially of a series of Less stylesheets that implement the various components of the toolkit. A stylesheet called bootstrap less includes the components stylesheets. Developers can adapt the Bootstrap file itself, selecting the components they wish to use in their project. Adjustments are possible to a limited extent through a central configuration stylesheet. More profound changes are possible by the Less declarations. The use of Less stylesheet language allows the use of variables, functions and operators, nested selectors, as well as so-called mixins. Since version 2.0, the configuration of Bootstrap also has a special "Customize" option in the documentation. Moreover, the developer chooses on a form the desired components and adjusts, if necessary, the values of various options to their needs. The subsequently generated package already includes the pre-built CSS style sheet.

As of Bootstrap 4, Sass will be used for stylesheets instead of Less. Grid system and responsive design comes standard with a 1170 pixel wide, grid layout. Alternatively, the developer can use a variable-width layout. For both cases, the toolkit has four variations to make use of different resolutions and types of devices: mobile phones, portrait and landscape, tablets and PCs with low and high resolution. Each variation adjusts the width of the columns.” [[11]](#endnote-11)

## jQuery



“ jQuery is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML. jQuery is the most popular JavaScript library in use today, with installation on 65% of the top 10 million highest-trafficked sites on the Web. jQuery is free, open-source software licensed under the MIT License.

jQuery's syntax is designed to make it easier to navigate a document, select DOM elements, create animations, handle events, and develop Ajax applications. jQuery also provides capabilities for developers to create plug-ins on top of the JavaScript library. This enables developers to create abstractions for low-level interaction and animation, advanced effects and high-level, themeable widgets. The modular approach to the jQuery library allows the creation of powerful dynamic web pages and Web applications.

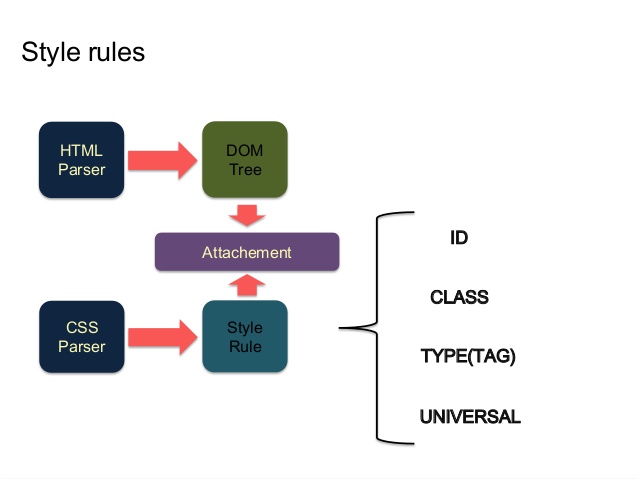
The set of jQuery core features—DOM element selections, traversal and manipulation—enabled by its selector engine (named "Sizzle" from v1.3), created a new "programming style", fusing algorithms and DOM data structures. This style influenced the architecture of other JavaScript frameworks like YUI v3 and Dojo, later stimulating the creation of the standard Selectors API.

Microsoft and Nokia bundle jQuery on their platforms. Microsoft includes it with Visual Studio for use within Microsoft's ASP.NET AJAX and ASP.NET MVC frameworks while Nokia has integrated it into the Web Run-Time widget development platform.jQuery has also been used in MediaWiki since version 1.16.

jQuery has two usage styles:

* Via the $ function, which is a factory method for the jQuery object. These functions, often called commands, are chainable as they all return jQuery objects.
* Via $.-prefixed functions. These are utility functions, which do not act upon the jQuery object directly.

Access to and manipulation of multiple DOM nodes in jQuery typically begins with calling the $ function with a CSS selector string. This returns a jQuery object referencing all the matching elements in the HTML page. $("div.test"), for example, returns a jQuery object with all the div elements of class test. This node set can be manipulated by calling methods on the returned jQuery object or on the nodes themselves.



**Figure 2.8 jQuery Dom selecting**

jQuery, at its core, is a DOM (Document Object Model) manipulation library. The DOM is a tree-structure representation of all the elements of a Web page and jQuery simplifies the syntax for finding, selecting, and manipulating these DOM elements. For example, jQuery can be used for finding an element in the document with a certain property (e.g. all elements with an h1 tag), changing one or more of its attributes (e.g. color, visibility), or making it respond to an event (e.g. a mouse click). jQuery also provides a paradigm for event handling that goes beyond basic DOM element selection and manipulation. The event assignment and the event callback function definition are done in a single step in a single location in the code. jQuery also aims to incorporate other highly used JavaScript functionality (e.g. fade ins and fade outs when hiding elements, animations by manipulating CSS properties).

The advantages of using jQuery are:

* **Encourages separation of JavaScript and HTML:** The jQuery library provides simple syntax for adding event handlers to the DOM using JavaScript, rather than adding HTML event attributes to call JavaScript functions. Thus, it encourages developers to completely separate JavaScript code from HTML markup.
* **Brevity and clarity:** jQuery promotes brevity and clarity with features like chainable functions and shorthand function names.
* **Eliminates cross-browser incompatibilities:** The JavaScript engines of different browsers differ slightly so JavaScript code that works for one browser may not work for another. Like other JavaScript toolkits, jQuery handles all these cross-browser inconsistencies and provides a consistent interface that works across different browsers.
* **Extensible:** New events, elements, and methods can be easily added and then reused as a plugin.” [[12]](#endnote-12)

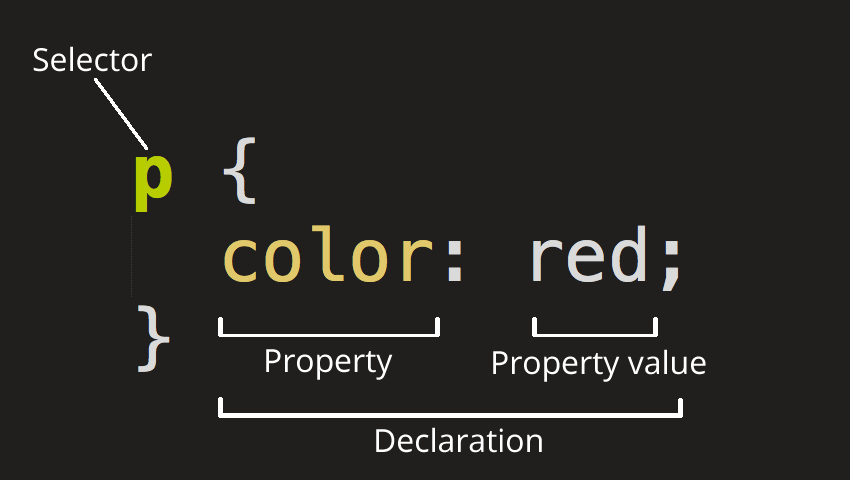
## http://1.bp.blogspot.com/-YTiH-WSka84/UQNlvIwQZsI/AAAAAAAAW9E/7FzoNxea6X4/s200/css3-logo.png Cascading Style Sheets

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

CSS is designed primarily to enable the separation of document content from document presentation, including aspects such as the layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

This separation of formatting and content makes it possible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (when read out by a speech-based browser or screen reader) and on Braille-based, tactile devices. It can also be used to display the web page differently depending on the screen size or device on which it is being viewed. Readers can also specify a different style sheet, such as a CSS file stored on their own computer, to override the one the author has specified. Changes to the graphic design of a document (or hundreds of documents) can be applied quickly and easily, by editing a few lines in the CSS file they use, rather than by changing markup in the documents.

The CSS specification describes a priority scheme to determine which style rules apply if more than one rule matches against a particular element. In this so-called cascade, priorities (or weights) are calculated and assigned to rules, so that the results are predictable. The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) text/css is registered for use with CSS by RFC 2318 (March 1998). The W3C operates a free CSS validation service for CSS documents.



**Figure 2.9 CSS selector**

CSS has a simple syntax and uses a number of English keywords to specify the names of various style properties. A style sheet consists of a list of rules. Each rule or rule-set consists of one or more selectors, and a declaration block.

In CSS, selectors are used to declare which part of the markup a style applies to by matching tags and attributes in the markup itself.

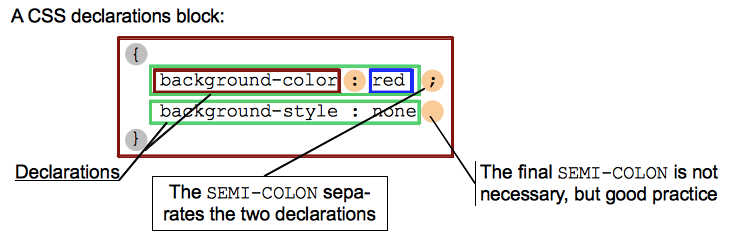
Selectors may apply to:

* all elements of a specific type, e.g. the second-level headers h2
* elements specified by attribute, in particular:
* id: an identifier unique within the document
* class: an identifier that can annotate multiple elements in a document
* elements depending on how they are placed relative to others in the document tree.

Classes and IDs are case-sensitive, start with letters, and can include alphanumeric characters and underscores. A class may apply to any number of instances of any elements. An ID may only be applied to a single element.

Pseudo-classes are used in CSS selectors to permit formatting based on information that is not contained in the document tree. One example of a widely used pseudo-class is :hover, which identifies content only when the user "points to" the visible element, usually by holding the mouse cursor over it. It is appended to a selector as in a:hover or #elementid:hover. A pseudo-class classifies document elements, such as :link or :visited, whereas a pseudo-element makes a selection that may consist of partial elements, such as ::first-line or ::first-letter.

Selectors may be combined in many ways to achieve great specificity and flexibility. Multiple selectors may be joined in a spaced list to specify elements by location, element type, id, class, or any combination thereof. The order of the selectors is important. For example, div .myClass {color: red;} applies to all elements of class myClass that are inside div elements, whereas .myClass div {color: red;} applies to all div elements that are in elements of class myClass.



**Figure 2.10 CSS declaration block**

A declaration block consists of a list of declarations in braces. Each declaration itself consists of a property, a colon (:), and a value. If there are multiple declarations in a block, a semi-colon (;) must be inserted to separate each declaration.

Properties are specified in the CSS standard. Each property has a set of possible values. Some properties can affect any type of element, and others apply only to particular groups of elements.

Values may be keywords, such as "center" or "inherit", or numerical values, such as 200px (200 pixels), 50vw (50 percent of the viewport width) or 80% (80 percent of the window width). Color values can be specified with keywords (e.g. "red"), hexadecimal values (e.g. #FF0000, also abbreviated as #F00), RGB values on a 0 to 255 scale (e.g. rgb(255, 0, 0)), RGBA values that specify both color and opacity (e.g. rgba(255, 0, 0, 0.8)), or HSL or HSLA values (e.g. hsl(000, 100%, 50%), hsla(000, 100%, 50%, 80%)).” [[13]](#endnote-13)

## AngularJS logo AngularJS

AngularJS (commonly referred to as "Angular" or "Angular.js") is a complete JavaScript-based open-source client and server-side web application framework mainly maintained by Google and by a community of individuals and corporations to address many of the challenges encountered in developing single-page applications. The JavaScript components complement PhoneGap, the framework used for developing cross-platform mobile apps. It aims to simplify both the development and the testing of such applications by providing a framework for client-side model–view–controller (MVC) and model–view–viewmodel (MVVM) architectures, along with components commonly used in rich Internet applications.

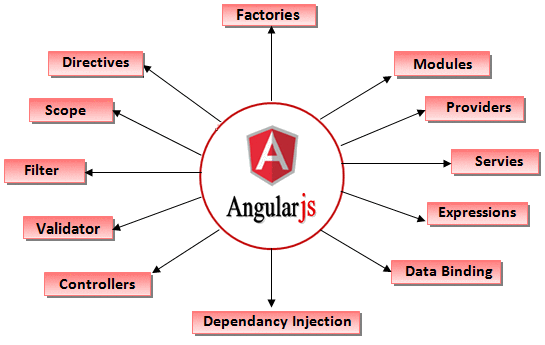
The AngularJS framework works by first reading the HTML page, which has embedded into it additional custom tag attributes. Angular interprets those attributes as directives to bind input or output parts of the page to a model that is represented by standard JavaScript variables. The values of those JavaScript variables can be manually set within the code, or retrieved from static or dynamic JSON resources. According to JavaScript analytics service Libscore, AngularJS is used on the websites of Wolfram Alpha, NBC, Walgreens, Intel, Sprint, ABC News, and approximately 8,400 other sites out of 1 million tested in July 2015.

AngularJS is the frontend part of the MEAN stack, consisting of MongoDB database, Express.js web application server framework, Angular.js itself, and Node.js runtime environment. AngularJS is built on the belief that declarative programming should be used to create user interfaces and connect software components, while imperative programming is better suited to defining an application's business logic. The framework adapts and extends traditional HTML to present dynamic content through two-way data-binding that allows for the automatic synchronization of models and views. As a result, AngularJS de-emphasizes explicit DOM manipulation with the goal of improving testability and performance.

AngularJS's design goals include:

* to decouple DOM manipulation from application logic. The difficulty of this is dramatically affected by the way the code is structured.
* to decouple the client side of an application from the server side. This allows development work to progress in parallel, and allows for reuse of both sides.
* to provide structure for the journey of building an application: from designing the UI, through writing the business logic, to testing.

Angular implements the MVC pattern to separate presentation, data, and logic components. Using dependency injection, Angular brings traditionally server-side services, such as view-dependent controllers, to client-side web applications. Consequently, much of the burden on the server can be reduced.



**Figure 2.11 AngularJS features**

Angular uses the term "scope" in a manner akin to the fundamentals of computer science.

Scope in computer science describes when in the program a particular binding is valid. The ECMA-262 specification defines scope as: a lexical environment in which a Function object is executed in client-side web scripts; akin to how scope is defined in lambda calculus. As a part of the "MVC" architecture, the scope forms the "Model", and all variables defined in the scope can be accessed by the "View" as well as the "Controller". The scope behaves as a glue and binds the "View" and the "Controller".

In Angular, "scope" is a certain kind of object that itself can be in scope or out of scope in any given part of the program, following the usual rules of variable scope in JavaScript like any other object. When the term "scope" is used below, it refers to the Angular scope object and not the scope of a name binding.” [[14]](#endnote-14)

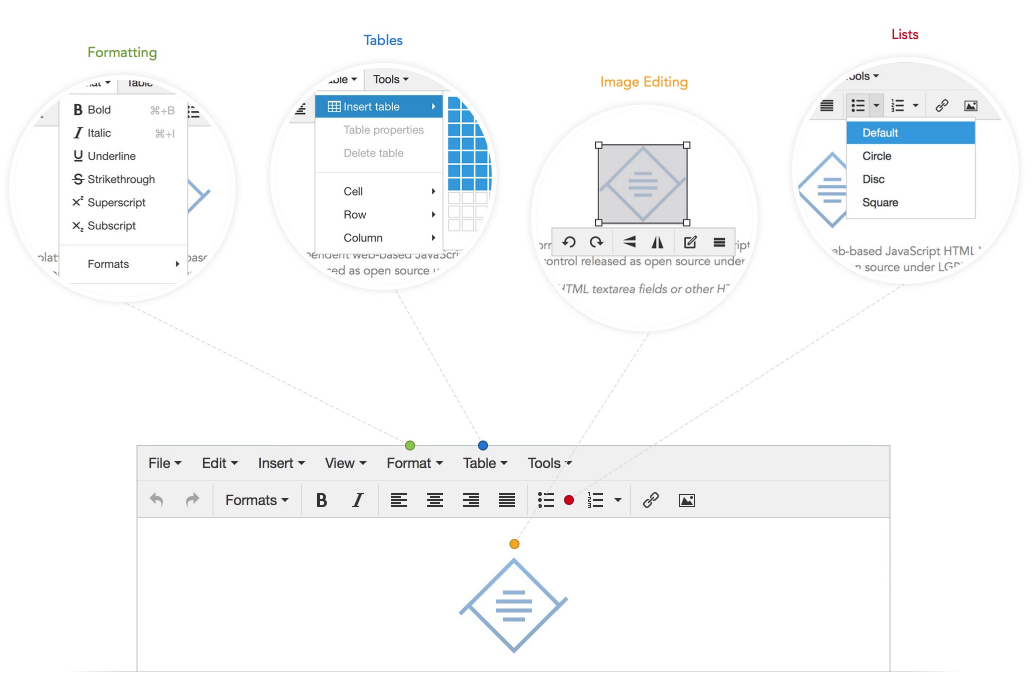
## https://www.tinymce.com/images/glyph-tinymce@2x.png TinyMCE

“TinyMCE (Tiny Moxiecode Content Editor) is a platform-independent, browser-based WYSIWYG editor control, written in JavaScript and released as open-source software under the LGPL by Ephox. It has the ability to convert HTML textarea fields or other HTML elements to editor instances. TinyMCE is designed to easily integrate with content management systems, including Django, Drupal, Joomla!, WordPress, and SOY CMS.

It offers HTML formatting tools, like bold, italic, underline, ordered and unordered lists, different types of alignments, in-line placement of images and videos, etc. It allows users to edit HTML documents online. The different options can be configured at the time of integration with a project, which improves the flexibility of a project.

TinyMCE is compatible with multiple browsers, including Internet Explorer, Mozilla Firefox, Safari, Opera and Google Chrome, across multiple operating systems.

TinyMCE includes an extensive API for custom integration.” [[15]](#endnote-15)



**Figure 2.12 TinyMCE features**

# SYSTEM DESIGN AND IMPLEMENTATION

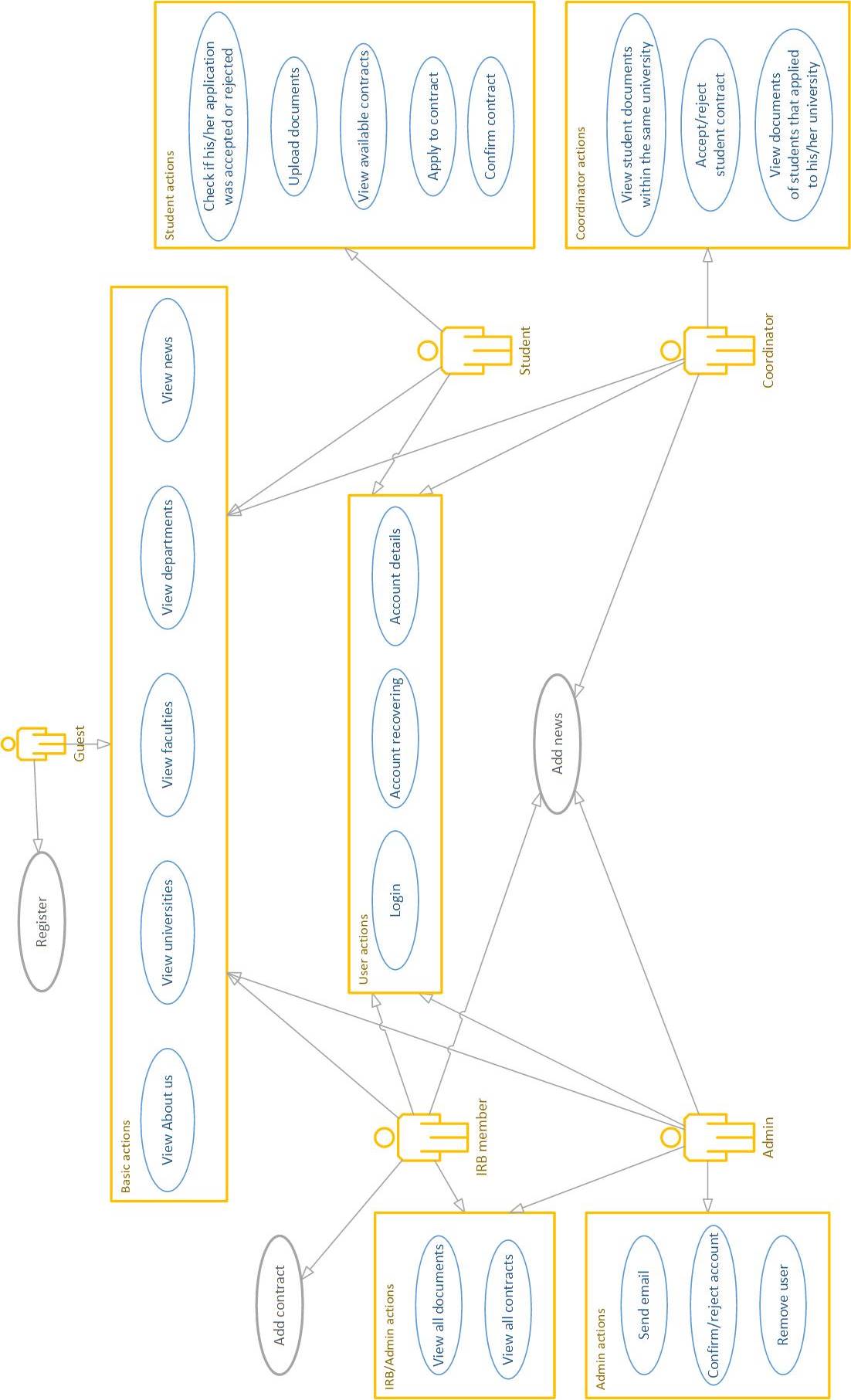
“Support Application for International Support Mobilities” is a web application created for both students and professors, which simplifies the process of registration for studying abroad. The application offers account creation, account management, contract creation, contract visualization, data uploading, data visualization, news adding and the possibility to visualize an entirely institution architecture with faculties, departments and sections providing external links and data about them.

## System requirements

“Support Application for International Support Mobilities” has the following functional requirements:

* Possibility to view the structure of a partner university:
  + Details: name, code, website
  + Contact person list: name, role, didactic grade, doctorate, email, mobile number, interior number 1, interior number 2
  + Legal representants list: name, function, didactic grade, doctorate
  + List with all partner faculties within one university
* Possibility to view the structure of a partner faculty:
  + Details: name, code, website
  + Contact person list: name, role, didactic grade, doctorate, email, mobile number, interior number 1, interior number 2
  + List with all departments within one faculty
* Possibility to view the structure of a department:
  + Details: name, code, website
  + Contact person list: name, role, didactic grade, doctorate, email, mobile number, interior number 1, interior number 2
  + List with all sections of a department
* Possibility to see the news
* Possibility to see details about the website, contact address
* Application account:
  + Account creation for students, coordinators, International Relations Bureau member
  + Login in the application
  + Account recovering through e-mail
  + See account details
* Student account:
  + Can upload documents in .pdf format: CV, transcript of grades, recommendation letter, motivational letter
  + Can view contracts available to him/her
  + Can apply to contracts
  + Can view if he/she was accepted or rejected
  + Can confirm an accepted contract
* Coordinator account:
  + Can view the documents of the students within the same university
  + Can accept/reject contracts of the students that apply to a faculty within his university
  + Can view the documents of the students that applied to a contract within his university
  + Can add news
* IRB member:
  + Can view the documents of all students
  + Can add contracts
  + Can add news
* Admin:
  + Can view the documents of all students
  + Can confirm/reject an user account
  + Can delete an user account
  + Can send e-mails via application e-mail
* Account e-mail confirmation

## Use case diagram



**Figure 3.1 Use-case diagram**

## System Architecture

“In software engineering, multitier architecture (often referred to as n-tier architecture) is a client–server architecture in which presentation, application processing, and data management functions are physically separated. The most widespread use of multitier architecture is the three-tier architecture.

N-tier application architecture provides a model by which developers can create flexible and reusable applications. By segregating an application into tiers, developers acquire the option of modifying or adding a specific layer, instead of reworking the entire application. A three-tier architecture is typically composed of a presentation tier, a domain logic tier, and a data storage tier. While the concepts of layer and tier are often used interchangeably, one fairly common point of view is that there is indeed a difference. This view holds that a layer is a logical structuring mechanism for the elements that make up the software solution, while a tier is a physical structuring mechanism for the system infrastructure.”

### Three tier architecture

Three-tier architecture is a client–server software architecture pattern in which the user interface (presentation), functional process logic ("business rules"), computer data storage and data access are developed and maintained as independent modules, most often on separate platforms. It was developed by John J. Donovan in Open Environment Corporation (OEC), a tools company he founded in Cambridge, Massachusetts.

Apart from the usual advantages of modular software with well-defined interfaces, the three-tier architecture is intended to allow any of the three tiers to be upgraded or replaced independently in response to changes in requirements or technology. For example, a change of operating system in the presentation tier would only affect the user interface code. Typically, the user interface runs on a desktop PC or workstation and uses a standard graphical user interface, functional process logic that may consist of one or more separate modules running on a workstation or application server, and an RDBMS on a database server or mainframe that contains the computer data storage logic. The middle tier may be multitiered itself (in which case the overall architecture is called an "n-tier architecture").

Three-tier architecture:

* Presentation tier

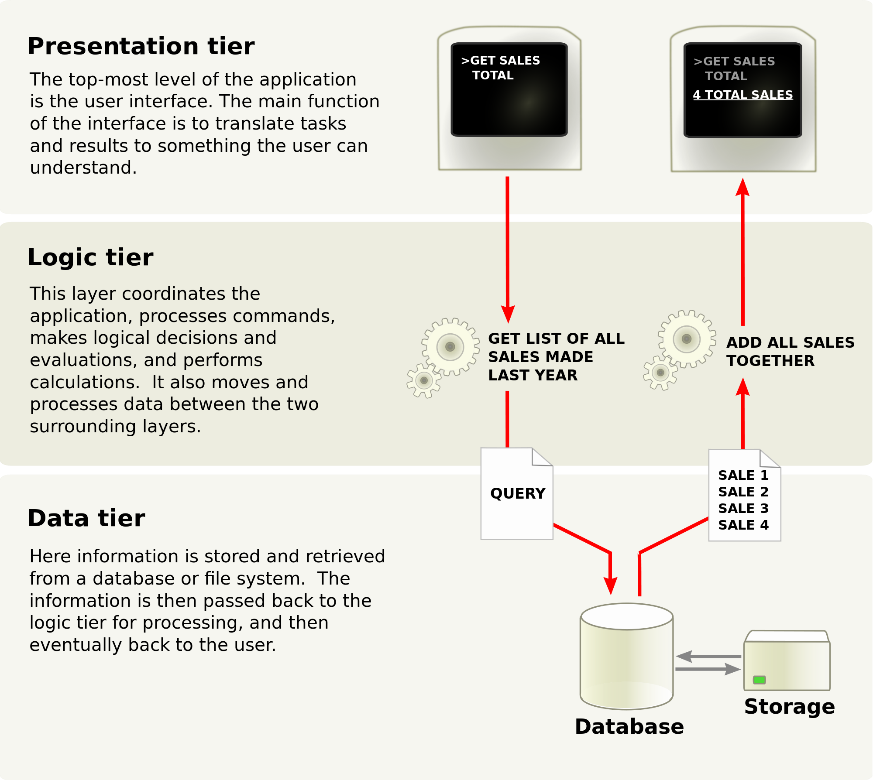
This is the topmost level of the application. The presentation tier displays information related to such services as browsing merchandise, purchasing and shopping cart contents. It communicates with other tiers by which it puts out the results to the browser/client tier and all other tiers in the network. (In simple terms it is a layer which users can access directly such as a web page, or an operating systems GUI)

* Logic tier (business logic, application tier, or middle tier)

The logical tier is pulled out from the presentation tier and, as its own layer, it controls an application’s functionality by performing detailed processing.

* Data tier

The data tier includes the data persistence mechanisms (database servers, file shares, etc.) and the data access layer that encapsulates the persistence mechanisms and exposes the data. Avoiding dependencies on the storage mechanisms allows for updates or changes without the application tier clients being affected by or even aware of the change.” [[16]](#endnote-16)



**Figure 3.2 Three tier architecture**

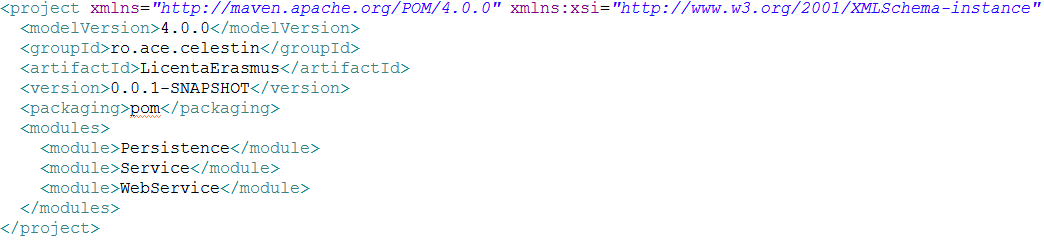
In “Support Application for International Support Mobilities” the bottom tier “Data tier” is called Persistence, the middle tier “Logic Tier” is called Service and the top tier “Presentation tier” is called WebService.

## Maven application structure

Maven supports dependency management and will retrieve dependency transitively for me and gives me the tooling I need to manage the complexity**inherent to dependency management:** I can analyze a dependency tree, control the versions used in transitive dependencies and exclude some of them **if**required. In Maven parlance, a module is just a subproject for a given “parent” Maven project. This organizational construct allows us to create a project that is compartmentalized into a collection of smaller modules.

“Support Application for International Support Mobilities” is structured in 3 maven modules

* Persistence module
* Service module
* WebService module



### Persistence module

Persistence module has two configuration files in .xml format:

* persistence.xml used to configure hibernate
* spring.xml used to configure the spring framework

Persistence.xml:

Spring.xml

****

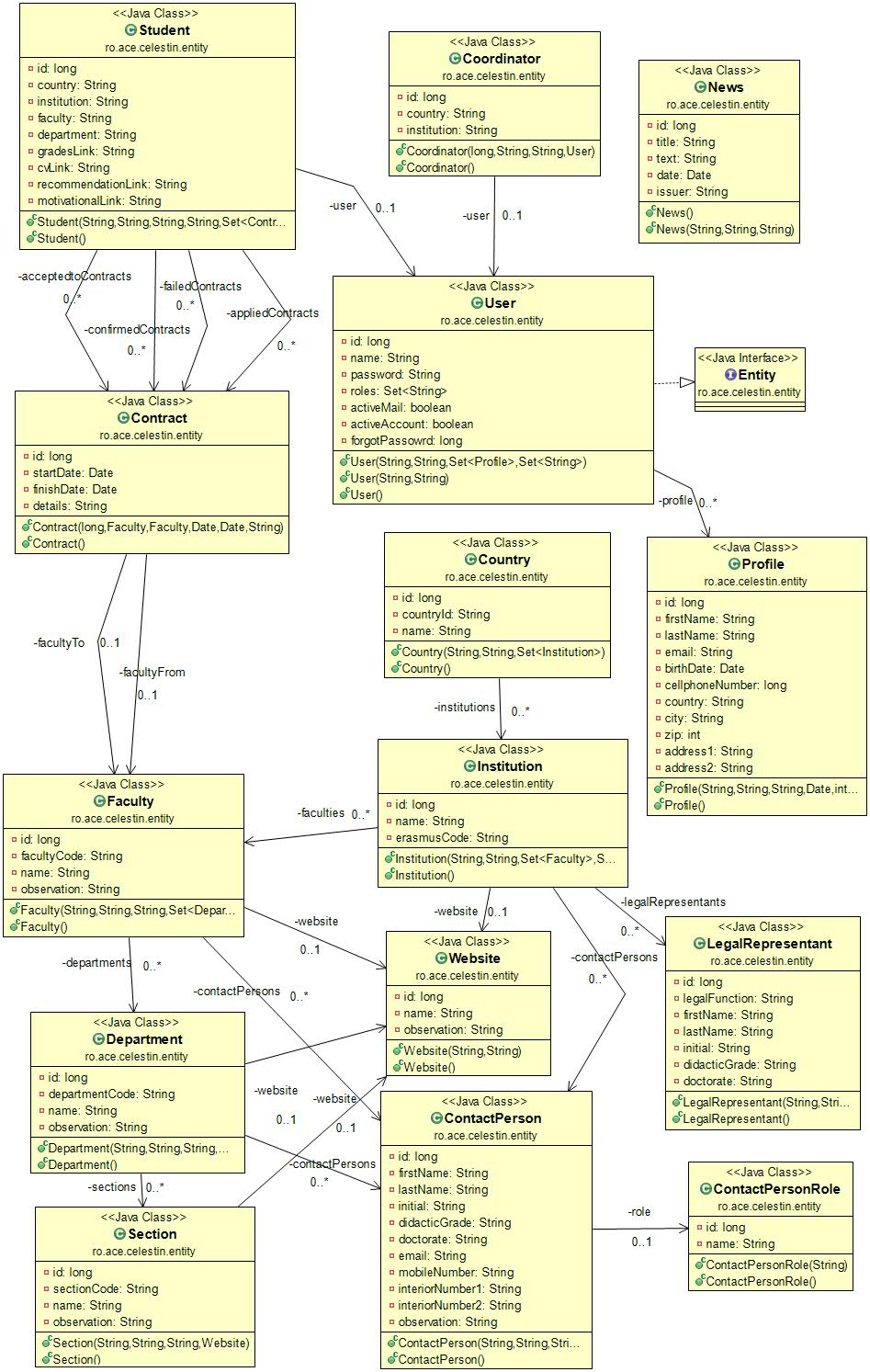
Persistence module has two packages named “entity” and “dao”, it also has a class named “Constants” filled with static final members consisting of names for tables and relations between them.

The first package “entity” package contains classes representing the java class representation of the database tables.

Entity classes are defined by the following:

* They are annotated with *@Entity* annotation from JPA used by hibernate to identify them and with *@Table* annotation used to define the name of the table representation in the database.
* Every class has an id data member annotated with the *@Id* annotation making it primary key in the database and with *@GeneratedValue* for auto-generating the id value of a database entry.
* *@Column* annotation is used on a class data member to add additional information that will processed by hibernate when a table is created. Information like the length restriction of an entry or if it is nullable or not etc.
* Annotations like *@ManyToOne, @OneToOne, @ManyToMany* are used to define the relations between two database tables.
* *@JoinColumn* annotation is basically used to define the column name of a foreign key in a table.

Basically entity classes are simple POJOs that have additional information added through the annotations used.

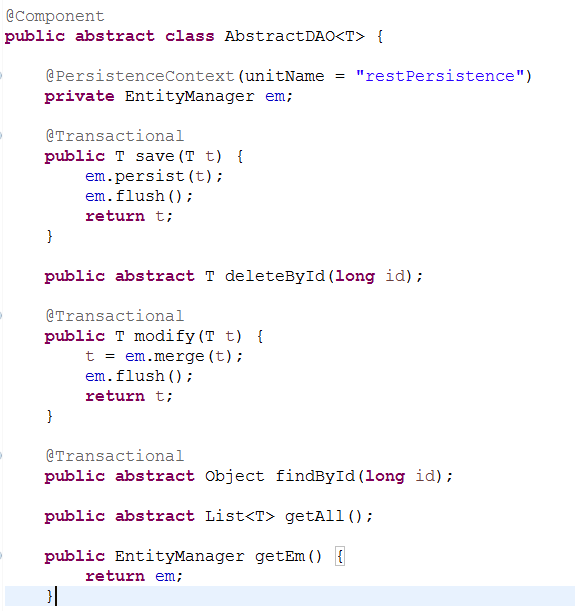


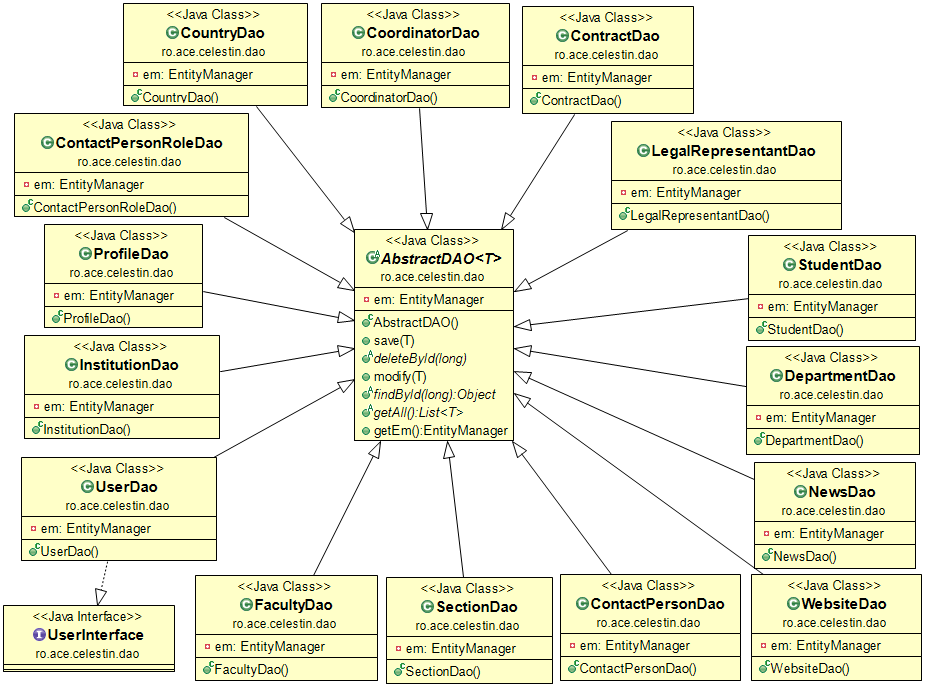
**Figure 3.3 Entities Class Diagram**

The second package “dao” package contains classes representing the data access objects required to access the elements from the database. Classes are annotated with the @Component annotation making them beans for spring framework. All classes have the “Entitymanager” data member from JPA used as a transaction manager by the spring framework.

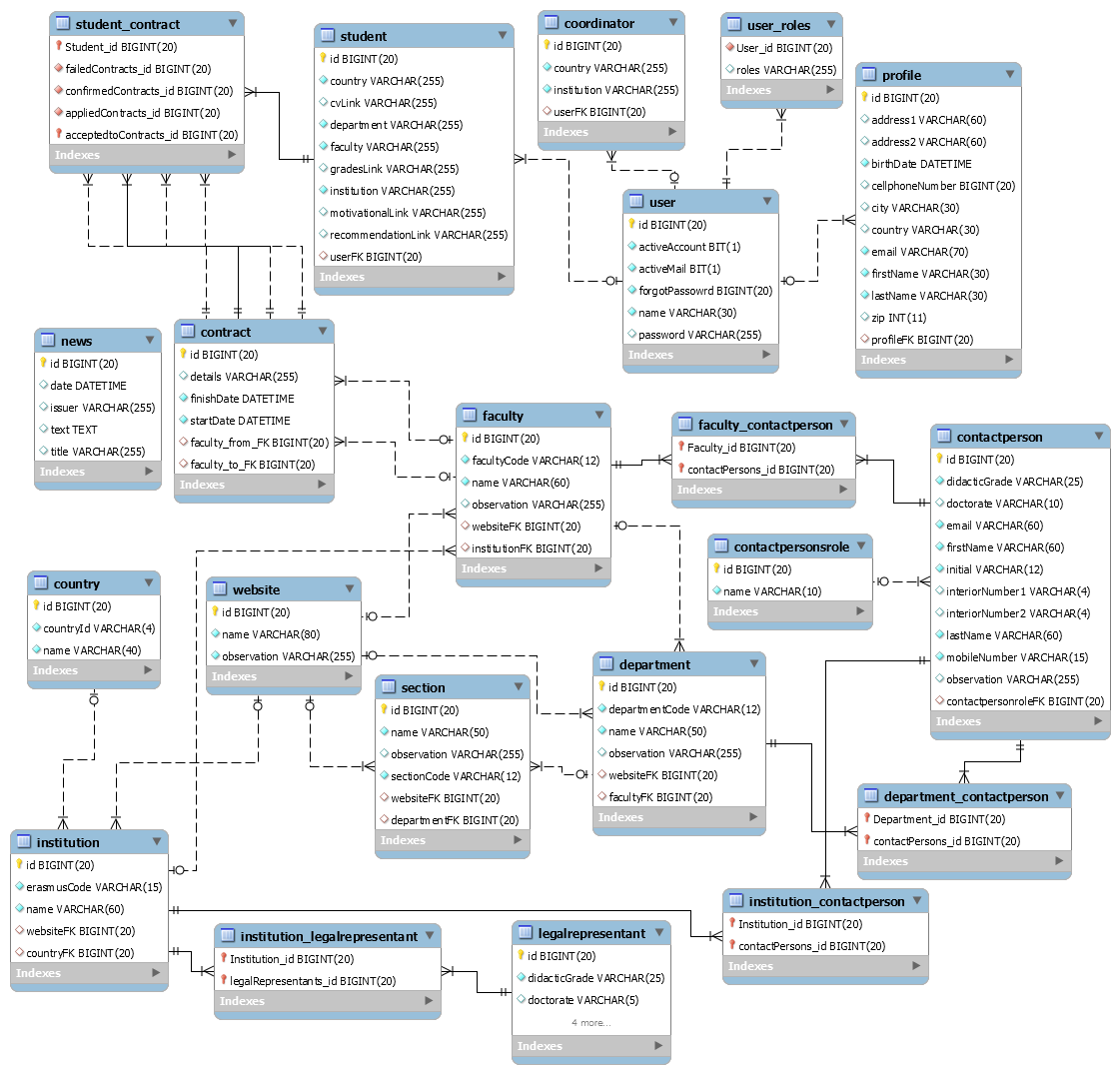
Each entity class has a corresponding dao class, dao classes use entity classes to provide functionality within the application.

Because every “dao” has common methods like persist and merge, I created an abstract template class named AbstractDAO that every class extends.





**Figure 3.4 Dao-s class diagram**



**Figure 3.5 Enhanced entity–relationship model**

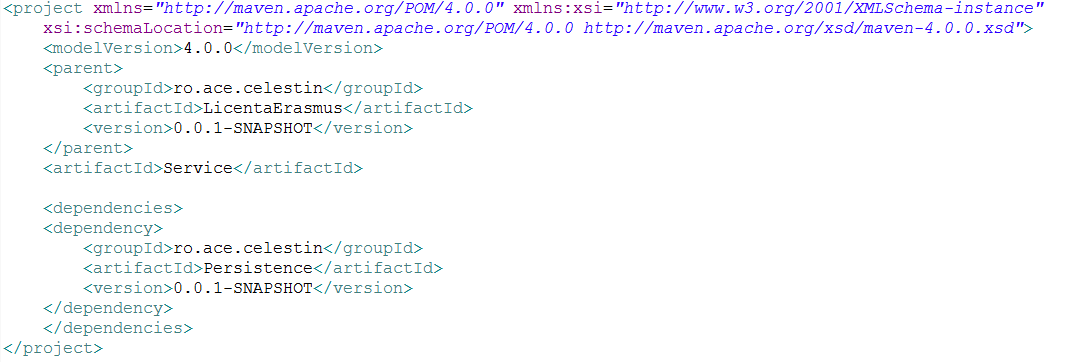
### Service module

The service module contains two packages:

* “dto” package where data transfer objects are stored.
* “service” package for the services of the application.

The service module has a dependency on the persistence module, declared in its project object model (“pom.xml”) because it needs it to perform its actions.

The pom.xml of the service module:



The Service module has two packages named “dto” and “service”.

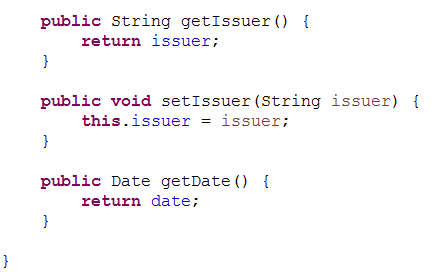
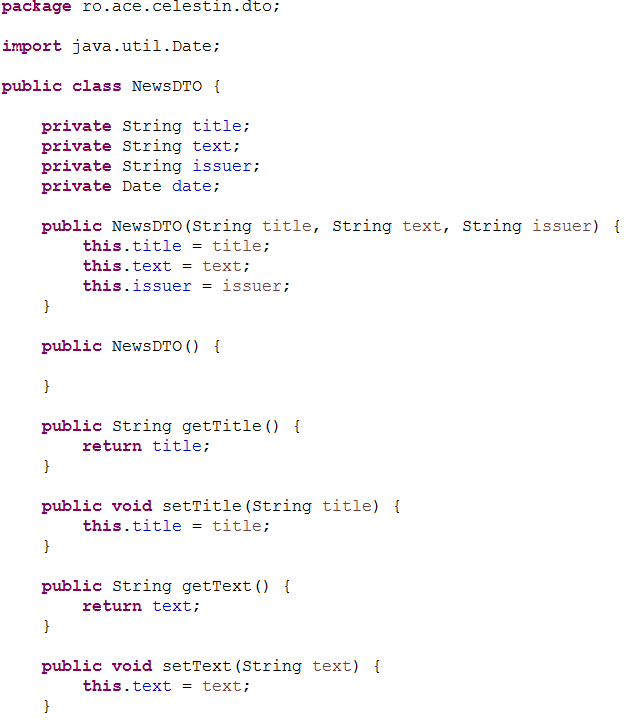
The first package “dto” contains classes representing data transfer objects that are used to encapsulate data, and send it from one subsystem of an application to another.

DTOs main benefit is that they reduce the amount of data that needs to be sent across the wire.

In “Support Application for International Support Mobilities” DTOs are used by the Service layer to transfer data between itself and the UI layer (WebService).

DTOs are simple POJOs that do not contain any business logic or method implementation

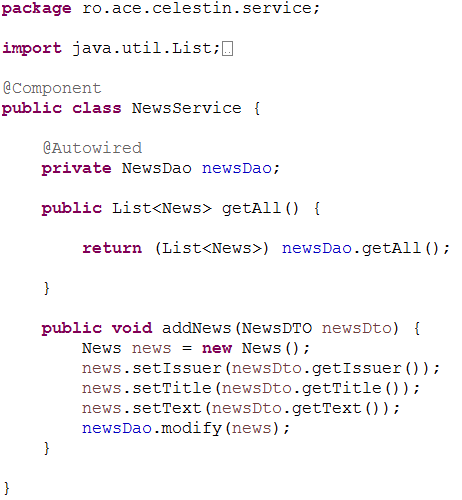
DTO example:



The second package “service” contains classes that provide a way for the client to interact with some functionality in the application.

For example NewsService will allow you to add news via addNews() method and get all the news from the database via getAll() method.

Service example:



Service classes are annotated with the *@Component* annotation making them beans within the spring framework. Every service will make use of the DAO classes via the *@Autowired* annotation.

When the program starts, spring will look for a class that matches the property of a member annotated with the *@Autowired* annotation in the applicationContext and will inject it automatically.

### Web Service module

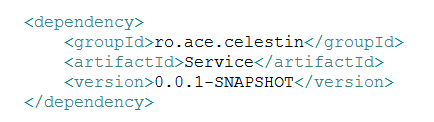
The third module respectively the last one is the WebService module in which we have two packages “restService” and “utils”.

WebService module is a dynamic web project at its roots. So it has a “webapp” folder where all the front end part of the application stands.

“webapp” folder contains:

* html pages
* cascade style sheet files
* java script files (angular js)
* images
* resources ( json objects)
* a folder named StudentDocuments where all the documents uploaded by students go.

In its pom.xml WebService module has a dependency on the Service module.



Because WebService it’s a dynamic web project it has a web.xml in the WEB-INF folder that holds the configuration of our web application. It instructs the servlet container (tomcat) which classes to load, what parameters to set in the context, and how to intercept requests coming from browsers.

The application web.xml file.



In the “restservice” package we have classes that communicate directly to the user. They will consume the information a user will send and will give him a response if he wants something from the server.

This interchange of information is done with the help of JAX-RS rest methods like:

* POST
* GET
* UPDATE
* DELETE

Classes from the “restservice” package are annotated with the *@Component* annotation making them beans for the spring framework to manage. Every class has a *@Path* annotation that will define a parameter in the url, this way all the actions performed on the url-s a class provides, will start with this parameter. It is not obligatory to add this annotation on every class but it is considered a good practice.

This classes will make use of the classes from the Service module by autowiring them via spring framework annotation *@Autowired.*

Methods within the classes of the “restservice” package have multiple annotations and will usually return a Response object.

* *@GET / @POST* annotations representing the rest methods.
* *@Path* defining the URL where information can be accessed.
* *@Produces* used to define the format of the returned data
* *@Consumes* used to define the format data is received within the method
* *@Transactional* used to perform actions transactionaly on the database facilitating the LAZY loading fetchtype used by hibernate.

Bellow is the “NewsWebService” class where we can observe all the details presented above.



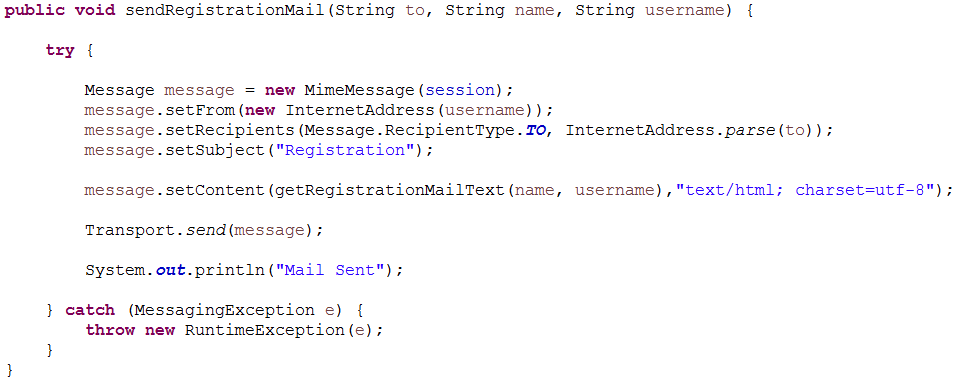
In the second package the “utils” package I stored some classes that will help me perform so logic of the application.

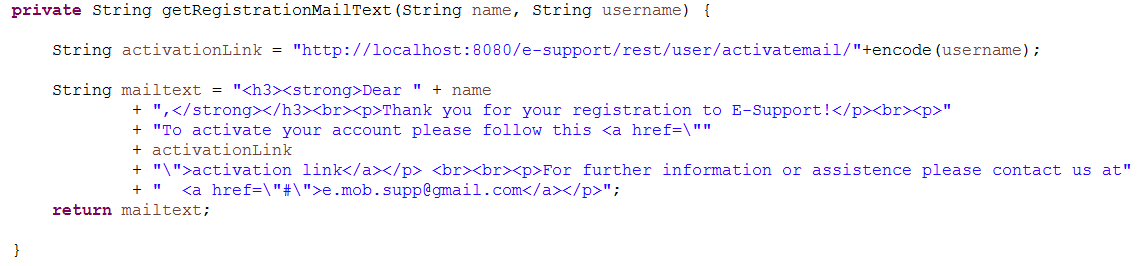
To be more specific there is stored the “EmailSender” class that will perform actions that consist in setting up the email service, constructing email message and send mail.

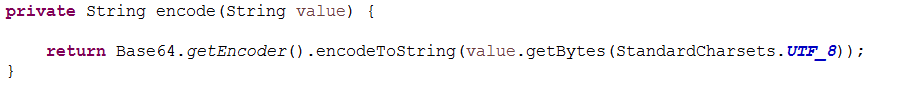


### E-mail confirmation

When a user completes the registration form, he/she will be asked to add his/her email address. After form verification an email will be sent to the user e-mail address asking him/her to confirm the e-mail address.



E-mail confirmation is done by creating an URL to a resource with the encoded username of the user as a parameter in it. The resource link is then added into the e-mail content of the e-mail that is sent to the user.

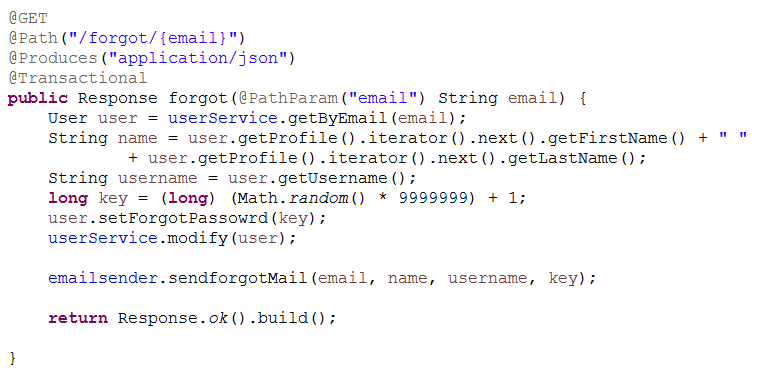
Encoding the username:

When the user accesses the link, a boolean value will be changed in the database making his e-mail active.

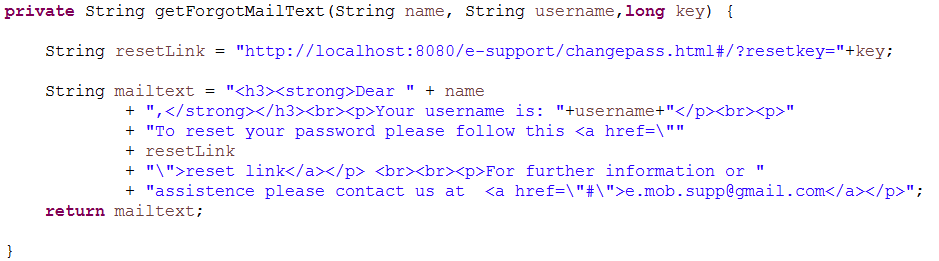
### Account Recovering

In “Support Application for International Support Mobilities” it is possible to recover your account in case you forgot your credentials but still remember and have access to the e-mail address you used in the registration process.

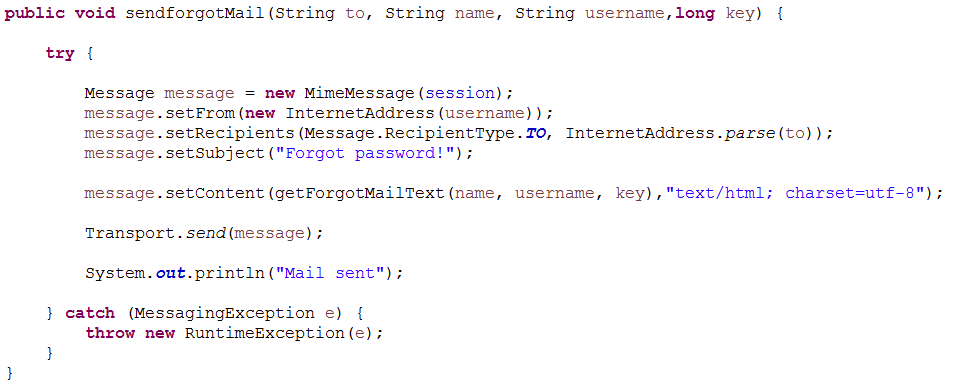
When a user requests account recovering by submitting his/her e-mail address in the forgot account form, a value in his/her user database entry will change from 0 to an autogenerated value.



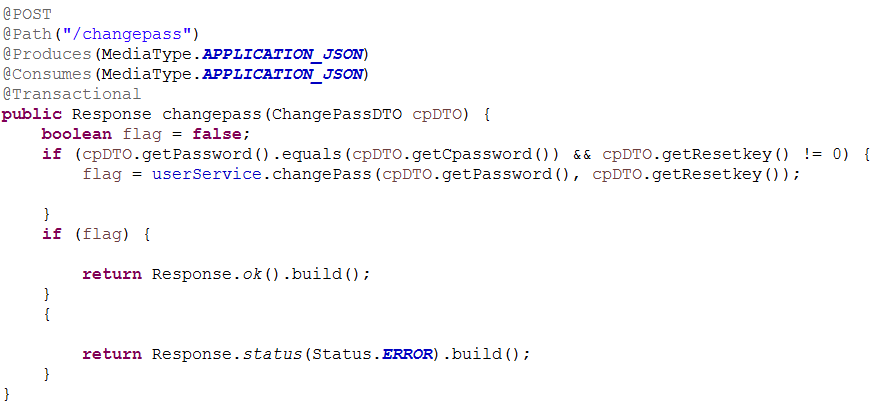
Then an e-mail content is created that contains the username and a password reset link. The password reset URL has the autogenerated value as a parameter in it.



Then an e-mail is sent to the user with the content created earlier.



When the user accesses the reset password link in the e-mail, he/she is redirected to a page where is asked to create a new password. After successfully password creation, his/her password field is updated in the database and the database value with the autogenerated value created earlier will change back to 0.



### Application security

In the WebService module we got a spring configuration file named applicationContext.xml where we define the security of the application. In applicationContext.xml we map a security filter that will be accessed every time a user request a secure method.

Secured methods can be easily defined here with the following structure:

<security:intercept-url method=*"RESTMETHOD"* pattern=*""* access=*"hasRole('userrole')"* />

ApplicationContext.xml



The security filter is a class from the restservice package named “AuthenticationFilter” mapped as a bean with the id “authenticationTokenProcessingFilter”.



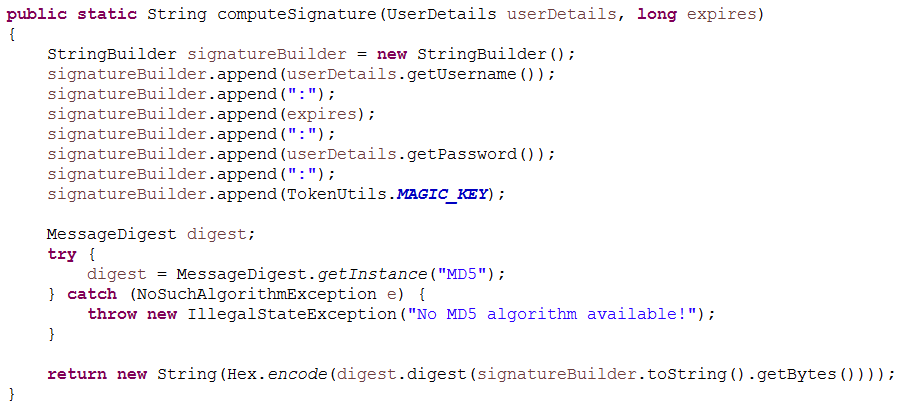
The authentication in application is realized with an authentication token that will be stored as a cookie when a user logs into the application. TokenUtils class deals with all the action performed on the token.

Creating the token:

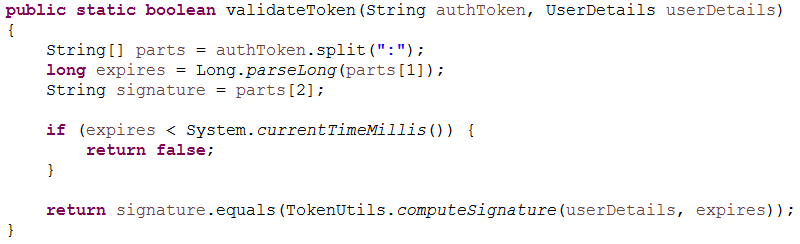


The token is created using the user credentials the expiration time and a secret key. It is then hashed using the MD5 hashing algorithm and hex encoded so it can be used as a cookie value.

ComputeSignature function that helps us to create the token:



Function to verify if a token has not expired yet:



# Conclusions

“Support Application for International Support Mobilities” is an application created for both students and teachers which has its main scope to eliminate the paperwork and to provide the users with information they need to begin the journey of studying abroad. In the end I can say that it completed its main purpose and that it provides some additional functionalities meant to make its users much more informed about this subject.

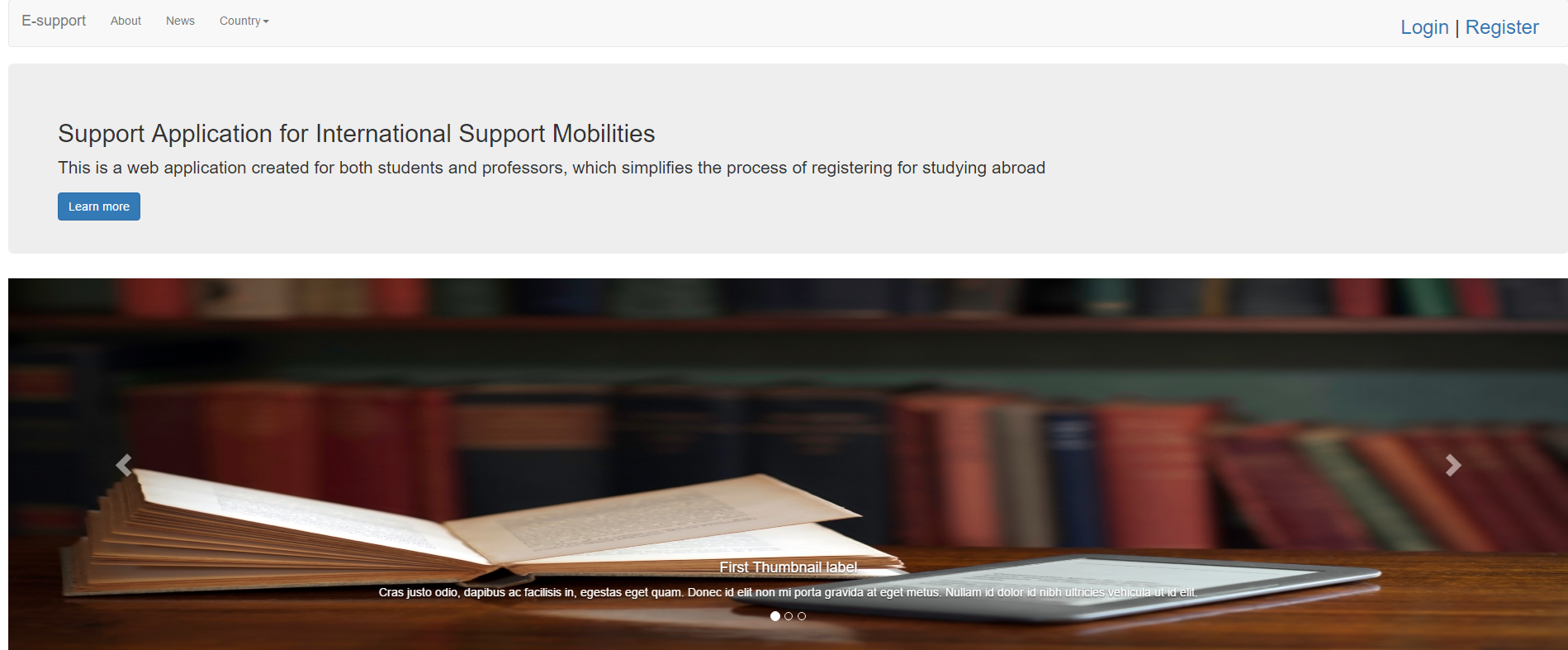
The development process of this application was very enthusiastic for me because it gave me the opportunity to research and use a lot of new technologies and APIs. I consider it a step ahead in my future career as a software developer.

For further development I suggest the implementation of a tool that will make the process of adding new institution architectures easier, because now this whole process it’s very tedious and can easily be compromised by a simple misunderstanding.

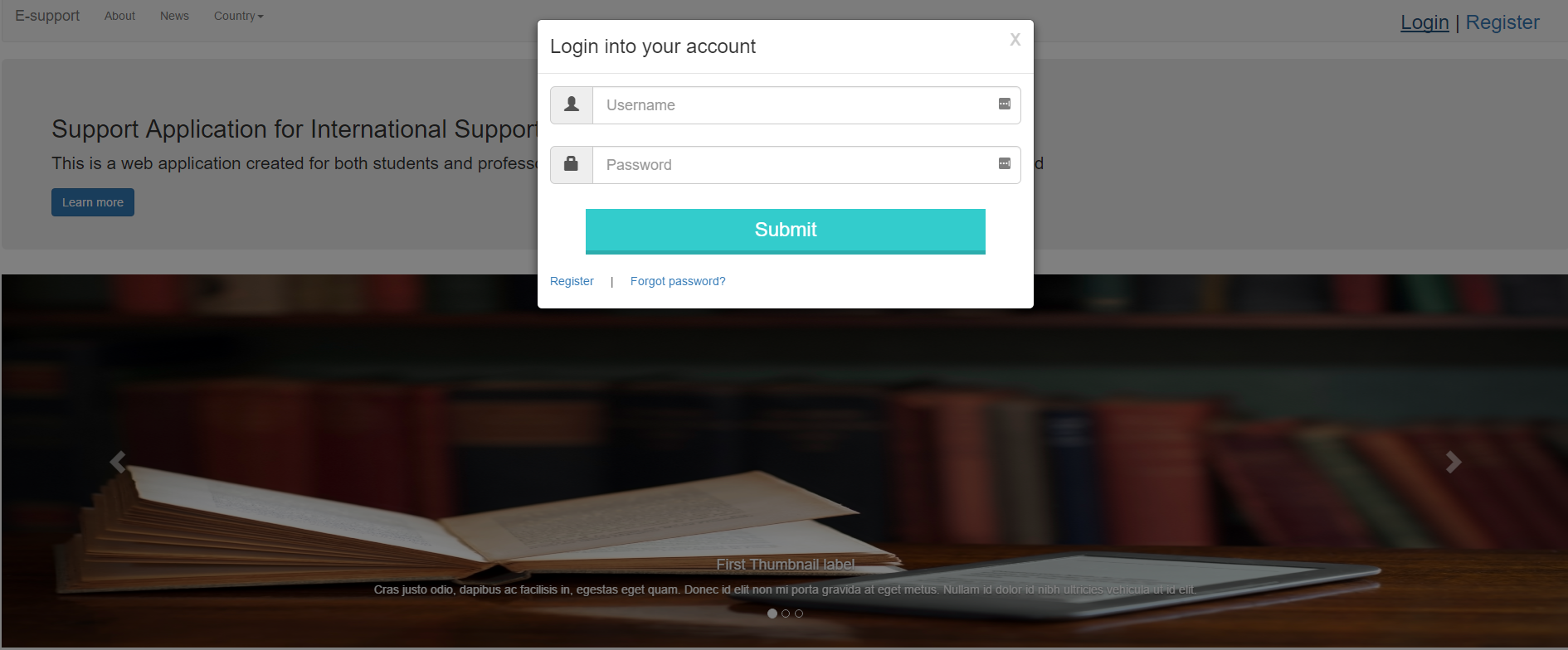
# Annex

## User Manual

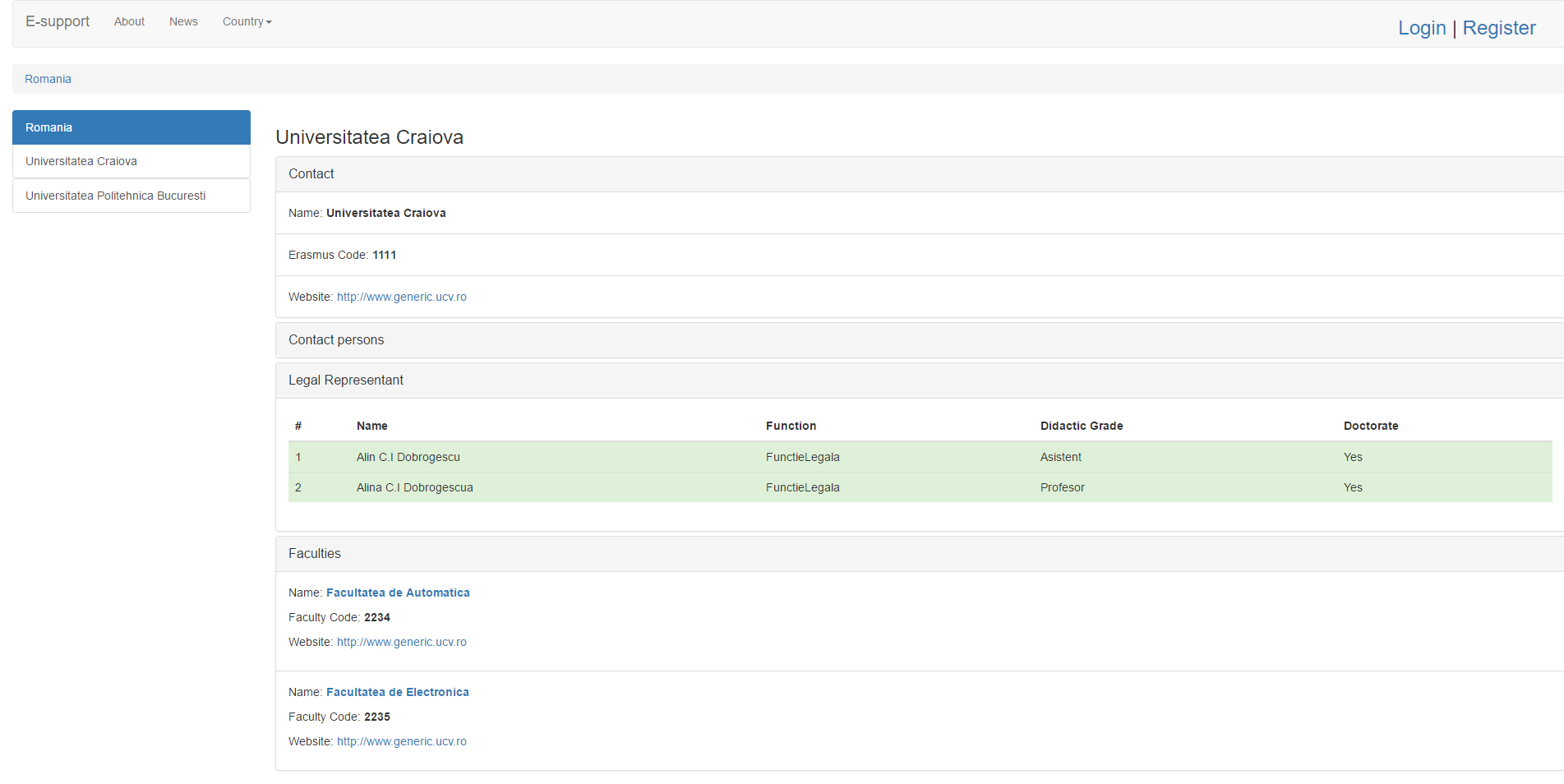
The entry point of the application is the index page.



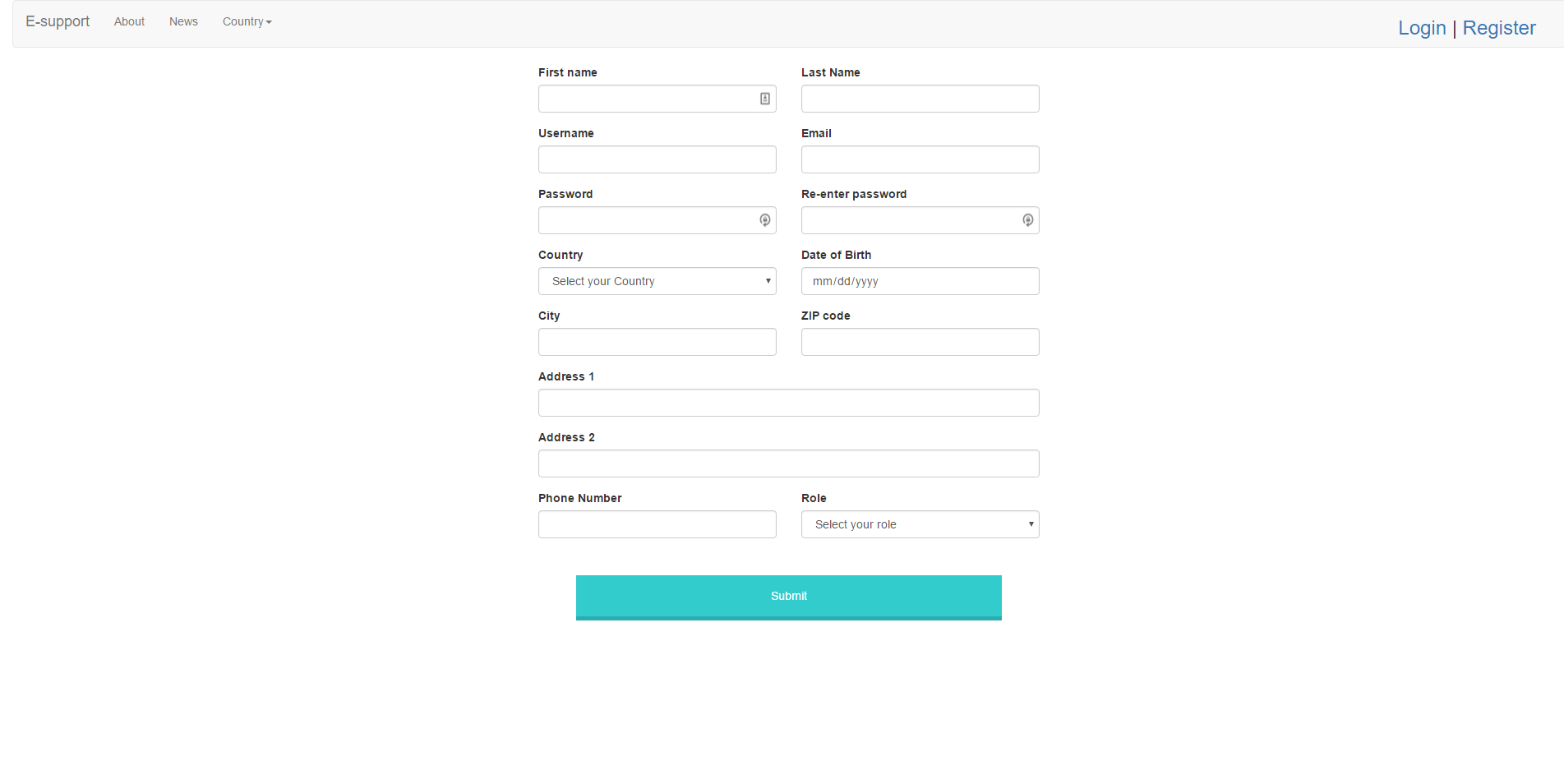
Login modal window. User needs to enter the credentials in order to log into the application.



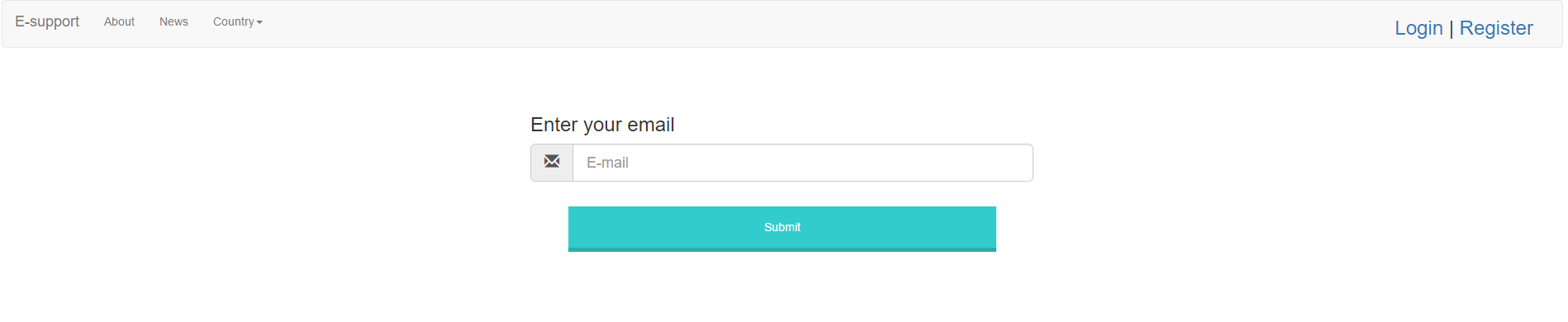
Selecting a country from the Country dropdown menu will redirect you to a page where the institutions from that country will be presented.



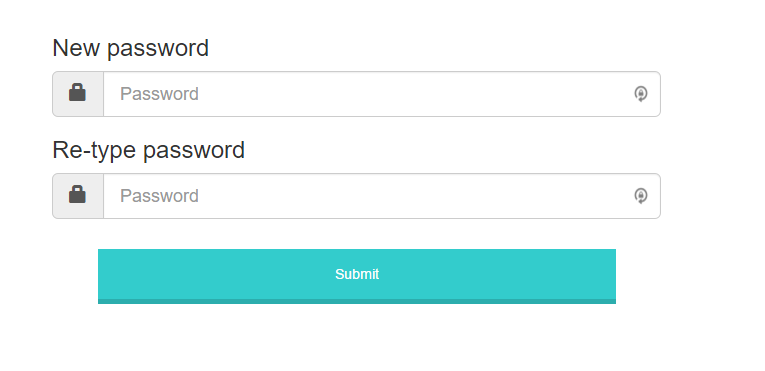
The registration form a user must complete in order to create an account.



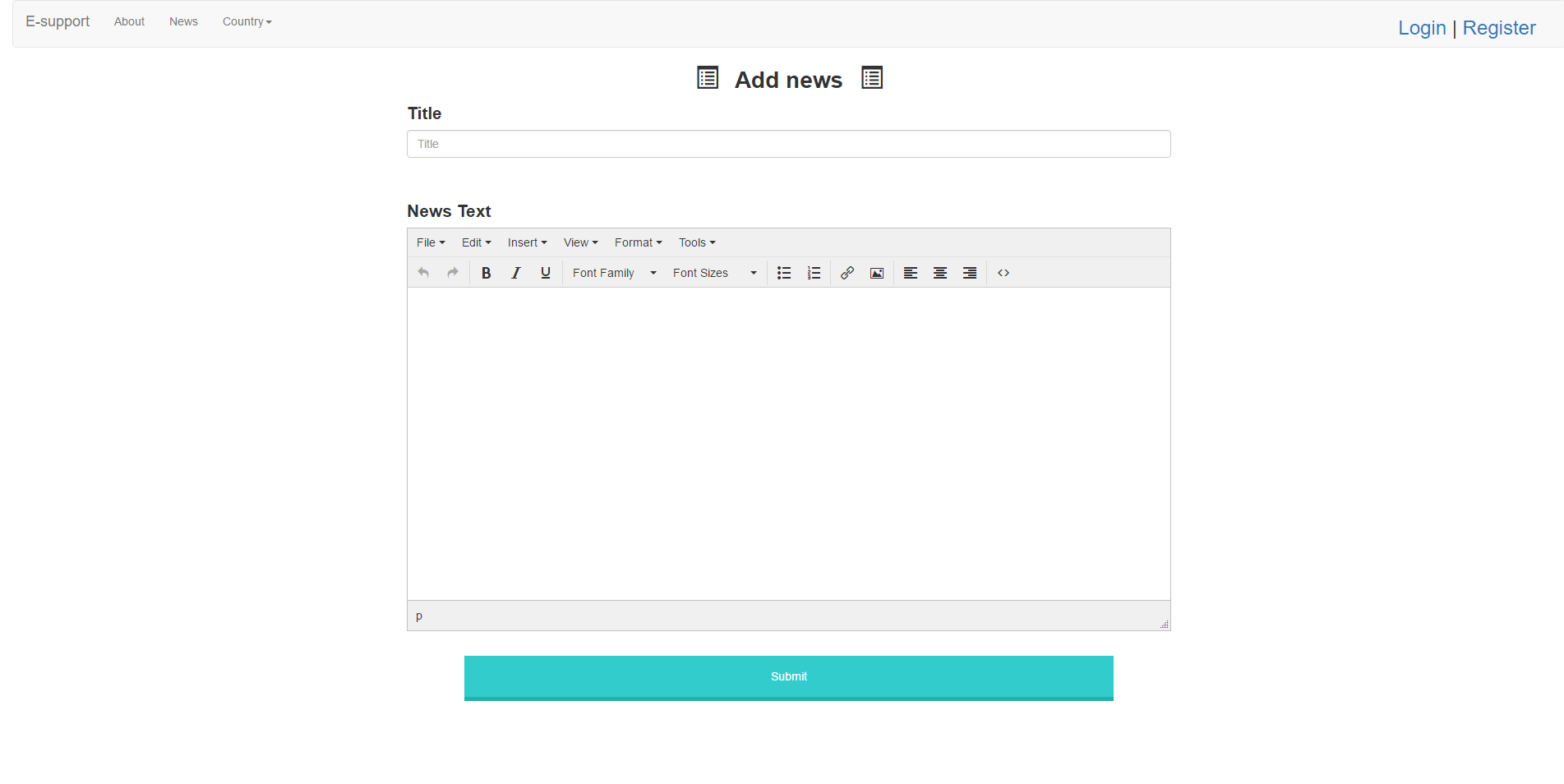
In the forgot password page the user is asked to enter his e-mail in order to recover his account.



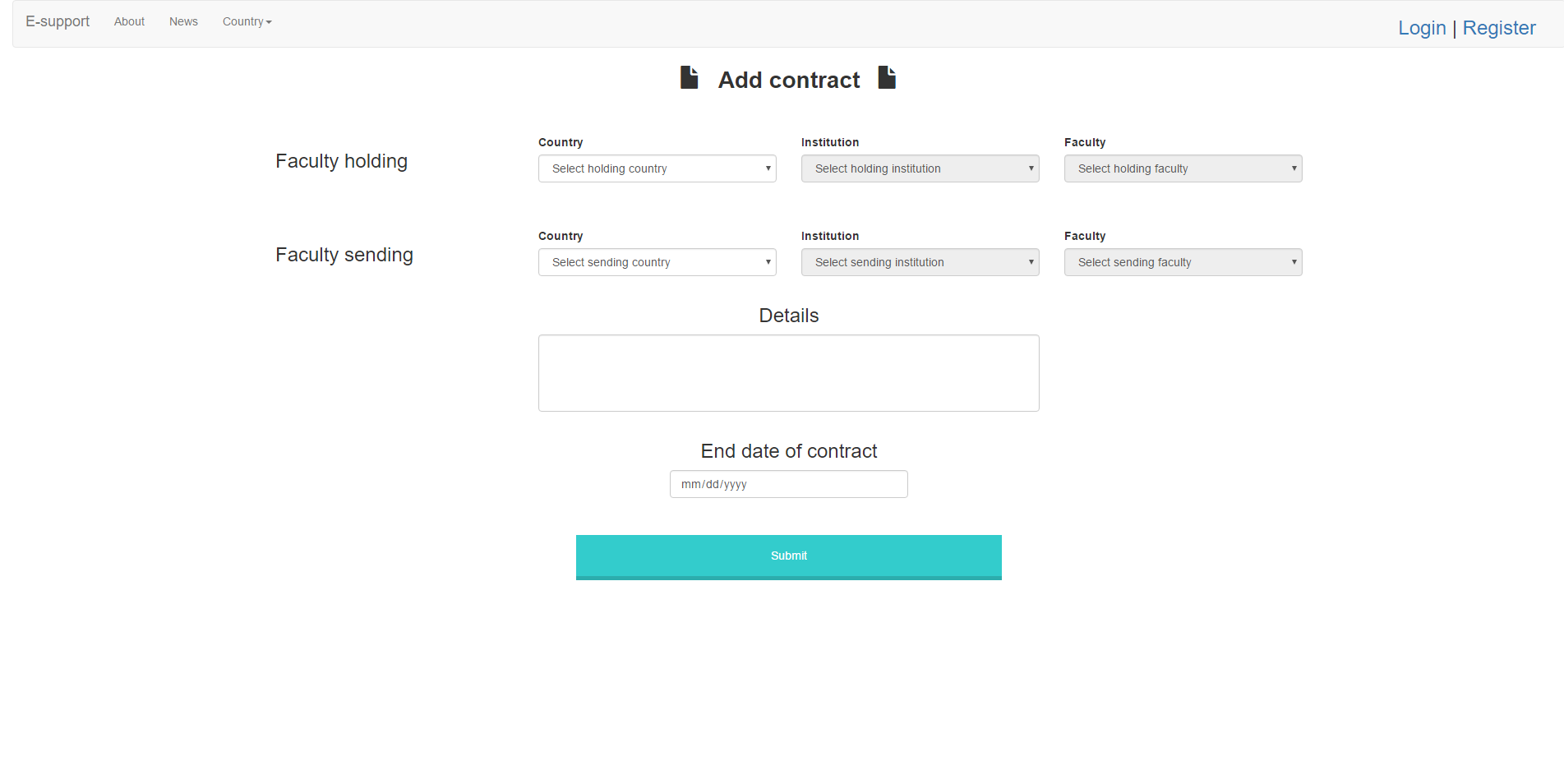
When a user accesses his reset password link from mail inbox he is prompted to this page. After completing this form his password will be updated.



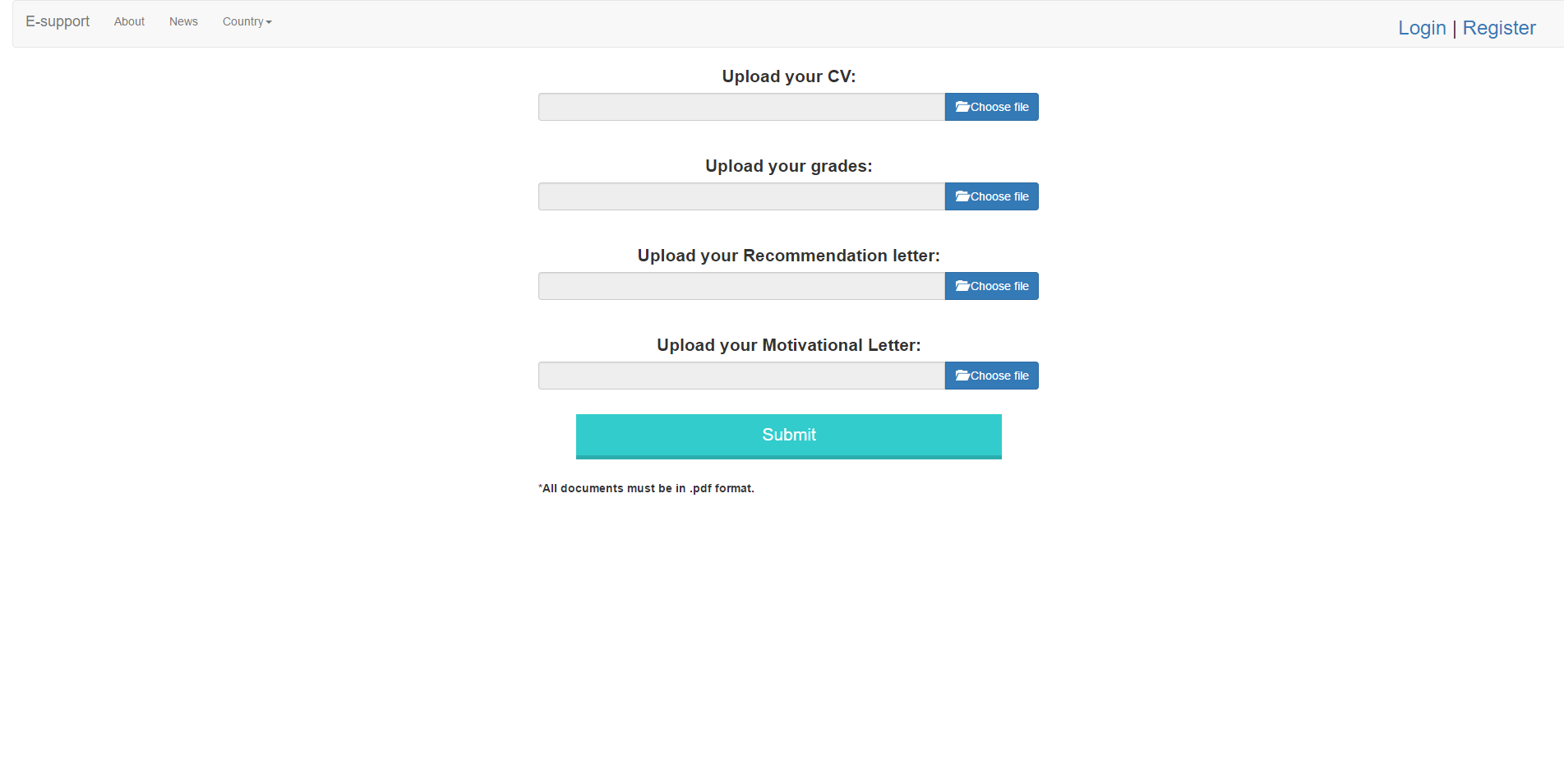
In this page a user can add news by completing the *Title* and *News Text* fields.



On the add contract page an IRB member can add a new contract after he selects the holding and the sending faculties and chooses an end date of the contract.



On the upload data page, a student can upload his documents. He must upload all four of them in order to complete the action.



# References

1. https://en.wikipedia.org/wiki/Java\_(programming\_language) [↑](#endnote-ref-1)
2. https://en.wikipedia.org/wiki/MySQL [↑](#endnote-ref-2)
3. http://hibernate.org/orm/ [↑](#endnote-ref-3)
4. https://maven.apache.org/what-is-maven.html [↑](#endnote-ref-4)
5. https://en.wikipedia.org/wiki/Apache\_Tomcat [↑](#endnote-ref-5)
6. https://en.wikipedia.org/wiki/Java\_API\_for\_RESTful\_Web\_Services [↑](#endnote-ref-6)
7. https://jersey.java.net/ [↑](#endnote-ref-7)
8. https://spring.io/tools/sts [↑](#endnote-ref-8)
9. https://en.wikipedia.org/wiki/Spring\_Framework [↑](#endnote-ref-9)
10. https://en.wikipedia.org/wiki/HTML5 [↑](#endnote-ref-10)
11. https://en.wikipedia.org/wiki/Bootstrap\_(front-end\_framework) [↑](#endnote-ref-11)
12. https://en.wikipedia.org/wiki/JQuery [↑](#endnote-ref-12)
13. https://en.wikipedia.org/wiki/Cascading\_Style\_Sheets [↑](#endnote-ref-13)
14. https://en.wikipedia.org/wiki/AngularJS [↑](#endnote-ref-14)
15. https://en.wikipedia.org/wiki/TinyMCE [↑](#endnote-ref-15)
16. https://en.wikipedia.org/wiki/Multitier\_architecture [↑](#endnote-ref-16)