Documentation for Delib, course project on ITP II

Team members:
Daniel Kalinin,
Elena Patrusheva,
Anastasia Pichka,
Susanna Gimaeva
BS1-5 students

Innopolis, 2018

Introduction and Purpose

This document describes the architecture and design for the Delib project. Delib is a web site project for all users who are fond of reading books and magazines and would like to have access to the IU library. Everyone can register, see all the books our library can offer and check them out. If you are a registered user, you will get the response if the library has available copies of the document you need and if there is a book you are searching for.

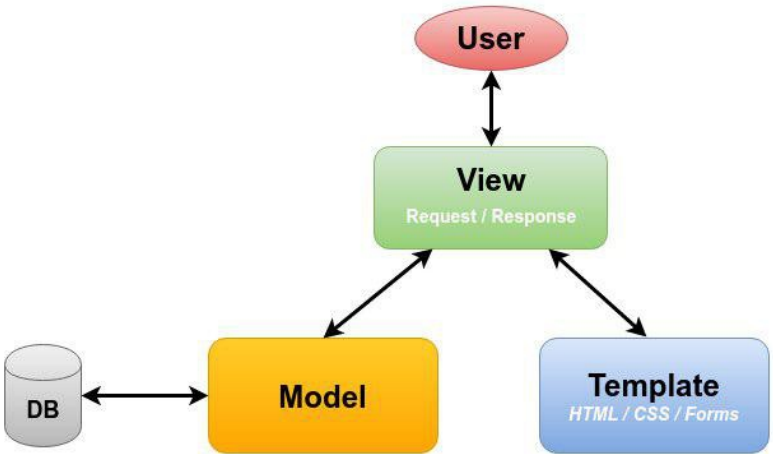
The purpose of the document is to describe the project in a way that addresses the interests and concerns of both users and developers. The document will identify major system components and describe their static attributes and dynamic patterns of interaction.

Django framework was chosen to implement the project as a web site because it is suitable for projects like this and it met the developers needs. Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel.

Among all the pros of Django the most important are as follows:

- It is free and open source.
- It was designed to help developers take applications from concept to
- completion as quickly as possible.
- It takes security seriously and helps developers avoid many common security mistakes.
- It is flexibly scalable.

System Design for Django projects



Here is what happens when a visitor lands on a Django page:

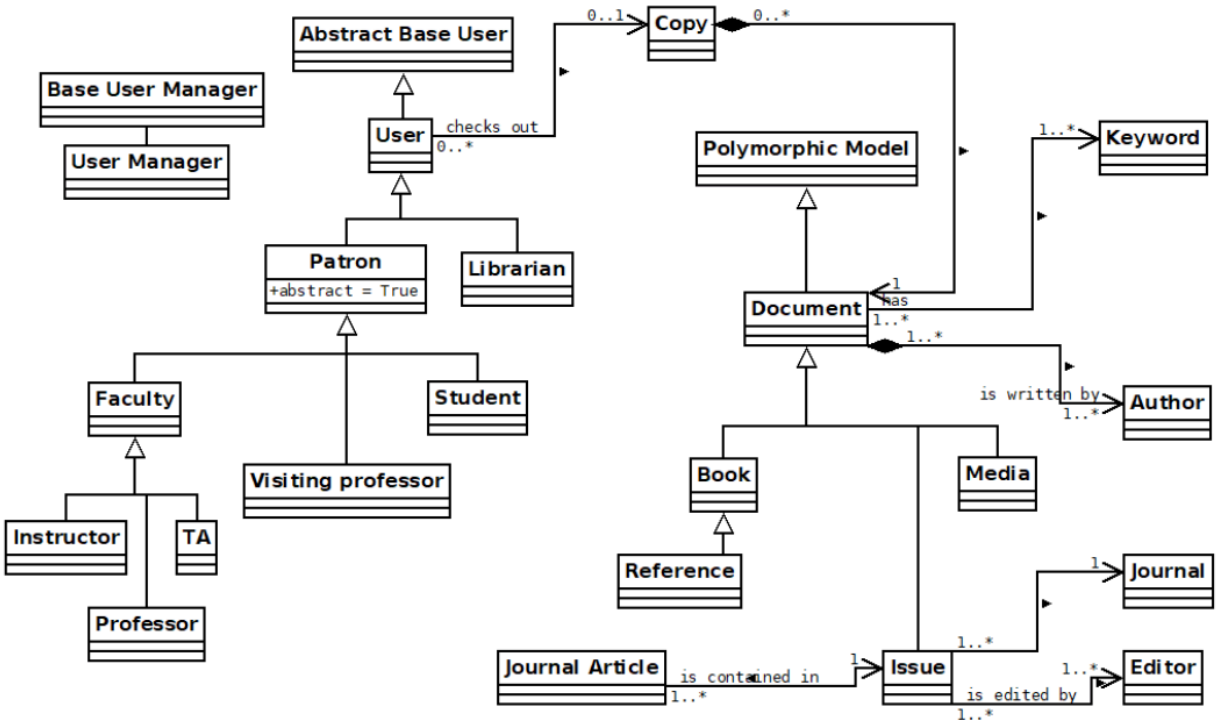
1.First, Django consults the various URL patterns created by a developer and uses the information to retrieve a view.

2.The view then processes the request, querying the database if necessary.

3.The view passes the requested information on to the template.

4. The template then renders the data in a layout a developer has created and displays the page.

High-Level Class Design (uml-diagram)



(see annex 1 - the whole uml-diagram)

Components description

Users

There is a special class User, which contains the information that library needs to know about each person of the system (name, surname, address, etc) and methods and fields for enabling authentication.

The class UserManager helps us to create a user.

There are two types of users in our implementation (they are inherited from class User):

Patron:

The methods of classes of documents (search for, check out, return documents) are available for objects and descendants of class Patron. There are three subtypes of patrons (users of these classes have different possibilities of using the library, e.g checking out period, number of times to renew the book, etc):

- Faculty, representing all the teaching body (e.g. professors, instructors, TAs).
- Students of the university. A student cannot be a faculty member.
- Visiting Professors. VPs are not part of the Faculty body.

Librarian:

The class contains methods that are available only for managing patrons by librarians(add user, delete user, modify user). The methods of classes of documents, such as check overdue document, check out documents, add copy, remove copy can be used by users of class Librarian.

Documents

There is a special class Document, which contains information that library needs to know about each document of the system(title, authors, keywords, price) and methods for checking out, returning documents(for all users), adding/deleting/ modifying document(for users of class Librarian). The methods of the class check the type of user (methods are available only for a certain type of user, meet the requirements of library system).

Class Author contains the author's name of the document.

Class Keyword contains document keywords.

The main asset of the library are documents. The application contains the following type of documents:

Book:

The class Book is inherited from Document. It also contains publisher, edition, year of publication and tells whether the book is a bestseller. A check out period is different for several types of users. There is a method that checks the type of user and makes it possible to take the book for a certain time for different users

Journal Articles:

There is a class Journal that contains the publisher of the journal.

Class Editor contains the editor's name of the issue.

There is a class Issue which is inherited from class Document and contains the journal, editors and publication date. There is a method for determining the check out period.

There is a class Journal Article which contains the title of the journal, issue and the authors.

Media:

The class is a descendant of the class Document and contains a method that make it possible to take the media for a certain time

Reference:

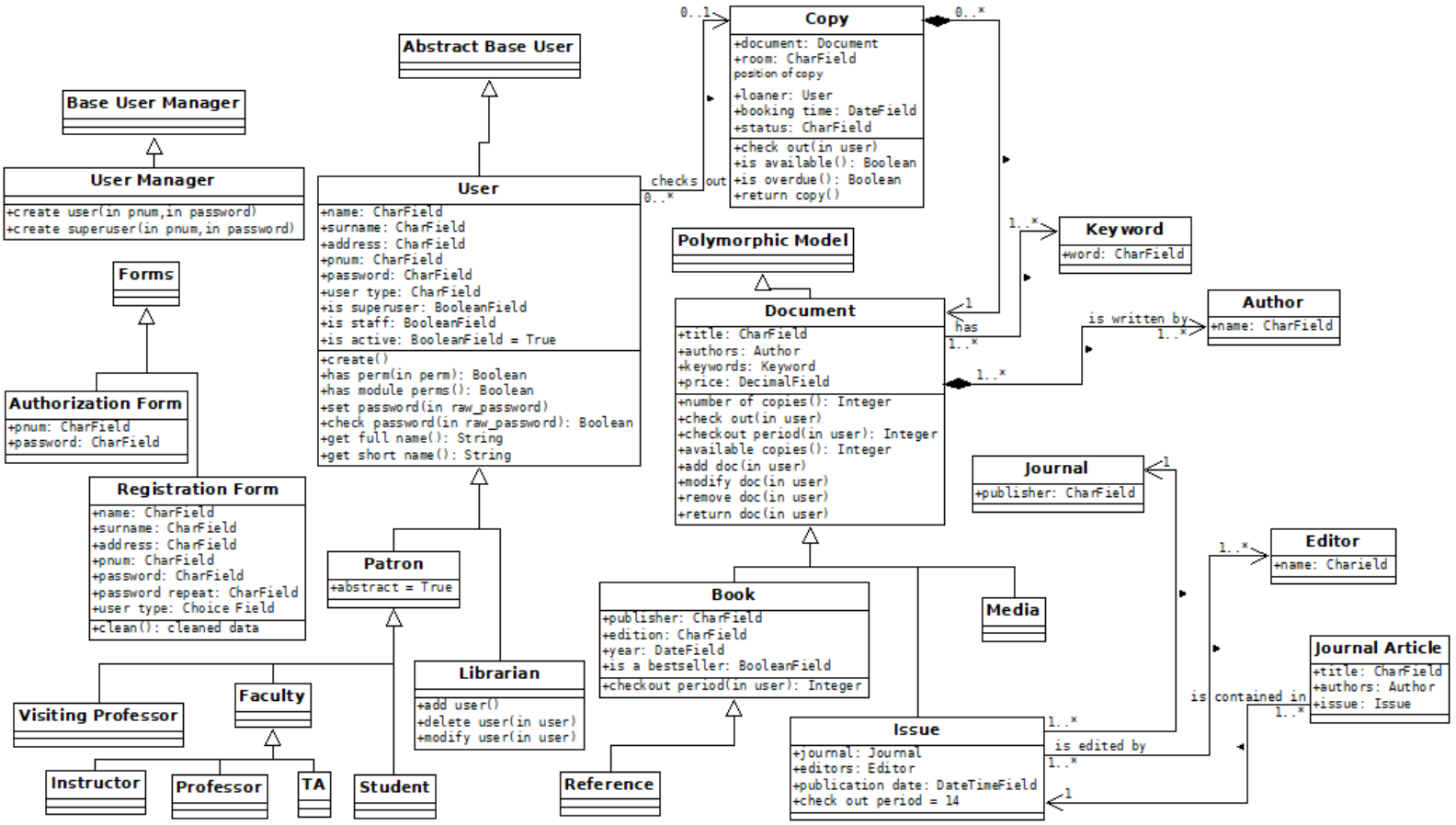
The objects of these class are not available for checking out

Copies

A library may have several copies of each document. The class Copies contains a place where the copy is stored, user that checks(if it is checking out) it and loan status.There are methods that help check out copy, return copy, etc.

In case of multiple requests for a document there is a queue of users, according to which they are served. The priority is as follows: Students, Instructors, TAs, Visiting Professors, Professors.

Annex1


Base User Manager
User Manager
+create user(in pnum,in password)
+create superuser(in pnum,in password)
Abstract Base User
User
+name: CharField
+surname: CharField
+address: CharField
+pnum: CharField
+password: CharField
+user type: CharField
+is superuser: BooleanField
+is staff: BooleanField
+is active: BooleanField = True
+create()
+has perm(in perm): Boolean
+has module perms(): Boolean
+set password(in raw_password)
+check password(in raw_password): Boolean
+get full name(): String
+get short name(): String
Copy
+document: Document
+room: CharField
position of copy
+loaner: User
+booking time: DateField
+status: CharField
+check out(in user)
+is available(): Boolean
+is overdue(): Boolean
+return copy()
checks out
0..1
0..*
0..*
Polymorphic Model
Keyword
+word: CharField
1..*
Document
+title: CharField
+authors: Author
+keywords: Keyword
+price: DecimalField
+number of copies(): Integer
+check out(in user)
+checkout period(in user): Integer
+available copies(): Integer
+add doc(in user)
+modify doc(in user)
+remove doc(in user)
+return doc(in user)
has
1
1..*
1..*
Author
+name: CharField
is written by
1..*
Forms
Authorization Form
+pnum: CharField
+password: CharField
Registration Form
+name: CharField
+surname: CharField
+address: CharField
+pnum: CharField
+password: CharField
+password repeat: CharField
+user type: Choice Field
+clean(): cleaned data
Patron
+abstract = True
Librarian
+add user()
+delete user(in user)
+modify user(in user)
Visiting Professor
Faculty
Instructor
Professor
TA
Student
Book
+publisher: CharField
+edition: CharField
+year: DateField
+is a bestseller: BooleanField
+checkout period(in user): Integer
Media
Reference
Journal
+publisher: CharField
1
Editor
+name: Charield
1..*
Journal Article
+title: CharField
+authors: Author
+issue: Issue
is contained in
1..*
Issue
+journal: Journal
+editors: Editor
+publication date: DateTimeField
+check out period = 14
1..*
is edited by
1..*
1