SET09103 Coursework 1 - Books Catalogue Development Report

Written by: Piotr Kubicki

Written for: Simon Wells

**Introduction**

This document describes development process of books catalogue. Application will be available online and will allow users to search catalogue using different criteria, view book details, show book cover if provided and allow to comment and rate the book. Application should use database to store data.

**Requirements**

Functional

Application must:

* show all available books
* allow to select different categories
* allow to select by different authors
* provide search facility
* display book details
* display author details
* allow user to rate and leave comment under any book
* be available online

Non-functional

Application must:

* be finished before deadline
* use appropriate colour scheme
* run in Levinux environment
* be builded using Python Flask
* be intuitive and easy to use
* accessible

**Resources and materials required**

* covers for books
* author portrait photos
* details about books
* details about authors
* machine with working instance of levinux operating system
* icons

**Information sources**

**Legal requirements**

Application will contain book covers and books authors photography but those will be used only for presentation purpose as a part of university assignment.

All software, frameworks and libraries must be used accordingly to their licenses.

**Design**

**Top level use case diagram**



**Use case scenarios**

**Use case 1:** User looking for a book by title

**Actor:** User

**Use case overview:** Actor visits page to find some details about Terry Pratchett “Small Gods” book.

**Trigger:** Actor opened page typing base url.

**Precondition 1:** Server is up and running

**Precondition 2:** Actor use machine with internet access

**Basic flow:** Display details of Terry Pratchett “Small Goods” book

**Description:** This scenario describes situation when user have internet access and server is up and running. In this scenario wanted book exists in database. This is a main success scenario.

1. System display books in alphabetical order by title.
2. Actor enter “Small Gods” into search bar select search by title and click search button.
3. Client send request to server.
4. Server return response with searching results.
5. Client display results to the Actor.
6. Actor select Terry Pratchett “Small Gods” book.
7. Client send request to the server.
8. Server returns response with book details.
9. Client display book details to the Actor.

**Termination outcome:** Actor close the page.

**Use case 2:** User looking for an author books

**Actor:** User

**Use case overview:** Actor visit page to find books written by his/her favourite witter.

**Trigger:** Actor opened page typing base url.

**Precondition 1:** Server is up and running

**Precondition 2:** Actor use machine with internet access

**Basic flow:** Display all books written by selected witter.

**Description:** Scenario describes situation when user have internet access and server is running. Actor looking for a not read book written by his/her favourite author.

1. System display books in alphabetical order by title.
2. Actor select author from page menu to unwind authors panel.
3. Actor select Terry Pratchett from panel.
4. Client send request to the server.
5. Server return response with results to the client.
6. Client display results in alphabetical order by title.

**Termination outcome:** Actor close the page.

**Back end design**

Database design

Database will store information about authors, books and ratings with comments. Every author may have many books. Every book may have one to many authors. Every comment must have only one book. Because many to many relations between authors and books tables exists, one more pivot table will be required. This database will allow to get all books related with selected author or authors and vice versa. Developer will also easily find all comments related with selected book. Database will also contain genres table to help order books by their genre. It will be help to generate categories menu panel and may be used to store additional information such as genre description in the future.

Database design diagram



Application design

API functions

Index (‘/’) GET function will return an index template with pagination and objects to display sorted in alphabetical order by book title. It will also return an array of displayed items sorted in the same order. That array will be used later in detailed view.

Search (‘/search?keyword=keyvalue…’) GET function will be used to send database query with requested key words. Function will get keywords from request ulr arguments. This function will behave differently depends from the arguments given. Function will allow search by using multiple filters such as title, author name. Function will return template with results sorted in alphabetical order by title, it will also contain array of items id’s sorted in the same order.

ShowByName (‘/authors/<name>’) GET function will redirect user to search function with ‘author’ filter and name as argument.

ShowByCategory (‘/categories/<name>’) GET function will redirect user to search function with ‘category’ as filter and category name as argument.

Default function will catch all not supported url’s and show 404 error page with link to home page.

Comment (‘/books/<id>/comments’) POST function will take user comment together with rating and add new entry to the database. Function will return updated book rating.

**Front end design**

Application will use two main layouts, collection and item layout. Collection layout will be used on the main page, to display search results and other url’s where display of multiple items is required. Item view layout will be used whenever details about the item are required.

Colours

Web application will use monochromatic colour view different shades of blue and white colour for backgrounds and objects, with black and white fonts used where appropriate.

Colours codes:

|  |  |  |
| --- | --- | --- |
| **RGB** | **HEX** |  |
| 0 103 204 | 0067CC |  |
| 41 85 127 | 29557F |  |
| 0 129 255 | 0081FF |  |
| 76 167 255 | 4CA7FF |  |
| 0 64 127 | 00407F |  |

Typography

Application will use sans-serif type fonts for long text to improve readability as this type of the fonts are better for screens. Headings and large characters will use serif type fonts to support visual design. Fonts used here have to be appropriate for wide range of users.

Long text, descriptions: Tahoma

Font-family: Tahoma, Geneva, sans-serif;

Headings: Georgia

Font-family: Georgia, Serif;

Storyboard

Sitemap

**Tests**

**Future**

Add login facility.

Responsive design for wider range of devices.

**Project evaluation**

**Personal evaluation**