Social Books Web Application

Presented By:

Awais Akbar, Yang Wang, and Karthick Pandi,

Department of Computer Science

Maynooth University, Ireland

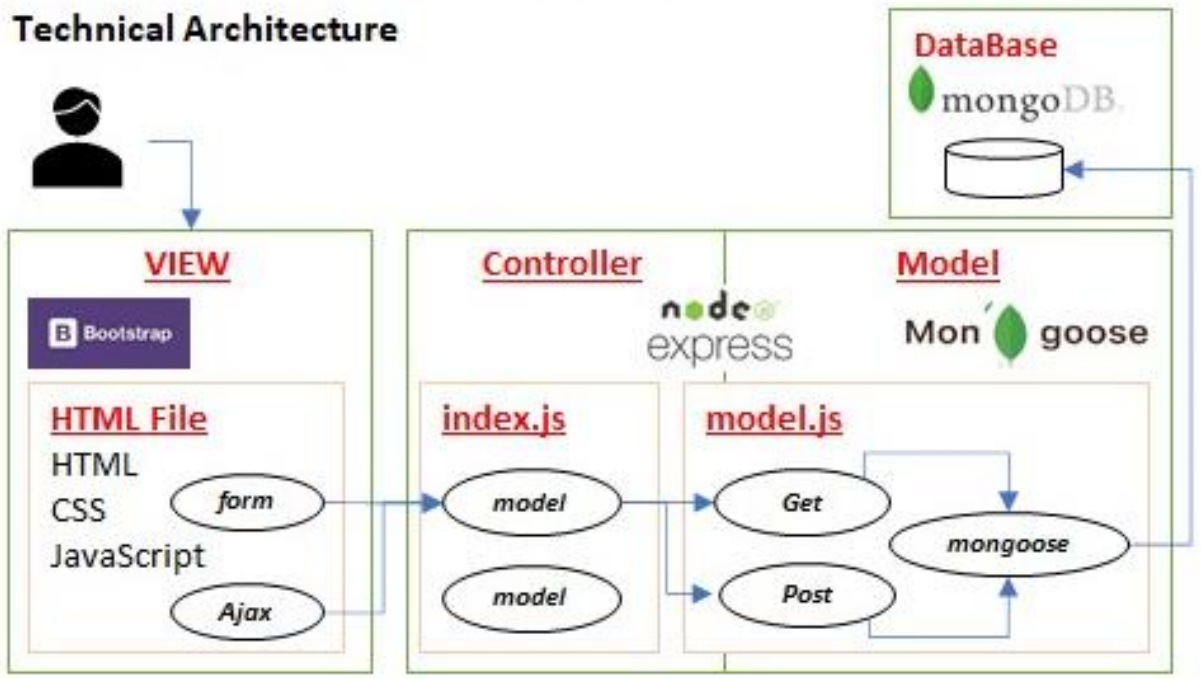
Maynooth, Co. Kildare, Ireland

Outline

- Part – I
  - Introduction
  - Technical Architecture
  - Technologies Used
  - User Stories
- Part – II
  - Use Case Diagram
  - Web App Architecture
  - User Interaction
  - Responsive Web Design
- Part – III
  - Cloud Deployment Architecture
  - Agile Ideas for Project Management
  - MongoDB Blackmail Issue

Maynooth University
National University of Ireland Maynooth

Technical Architecture



Maynooth University
National University of Ireland Maynooth

Technologies Used

Front End

HTML/CSS + Bootstrap Framework + Java Script

Back End

API Service Layer - Express JS/Node

ODM - Mongoose

Database

Mongo DB

Jenkins

Automated Software Development Process

Used for Continuous Integration & Continuous Delivery

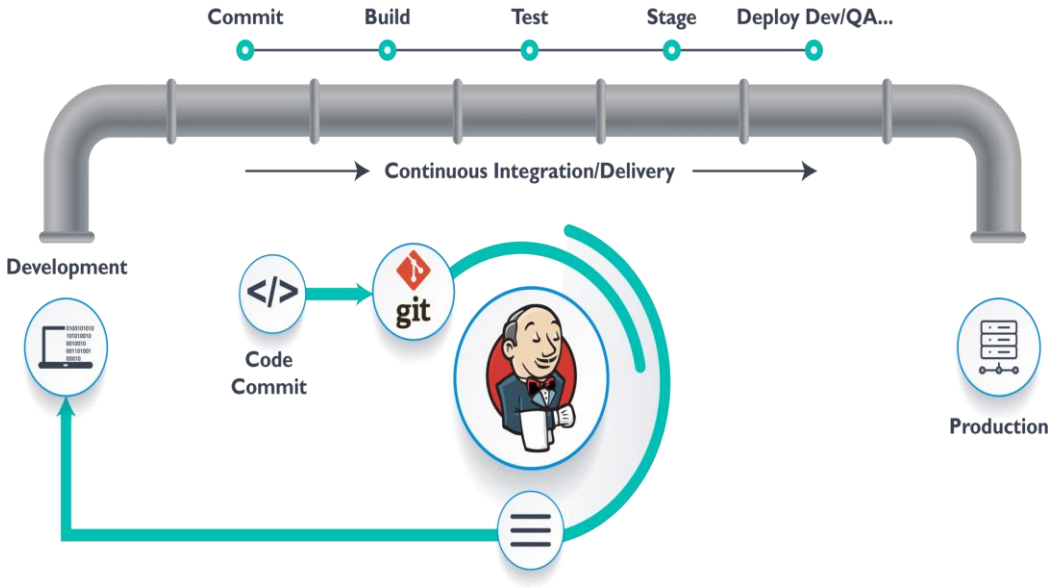
Continuous Integration

Will Check any breakage if new commits were made into the main branch

Continuous Deployment

Starts when CI gets done

Deploy changes to the testing environment



User Stories

User Management :

Registering with the **username and password** for portal login

User can able to login into book portal.

Able to **Reset or Change password**

Adding Books :

Title, Author, Year, Abstract, ISBN , Personal tags are used for books adding

Maynooth University
National University of Ireland Maynooth

User Stories

Personal Section

Books in **Personal Library** can be viewed basic on topics/metadata tags.

Each books relate to a topic

All books under the **topics** are getting listed

Social Section

In Social section where **recommendation** happens if they share at least **three** books in common.

Nothing we can know about the user other than the books which he is interested.

Book Management

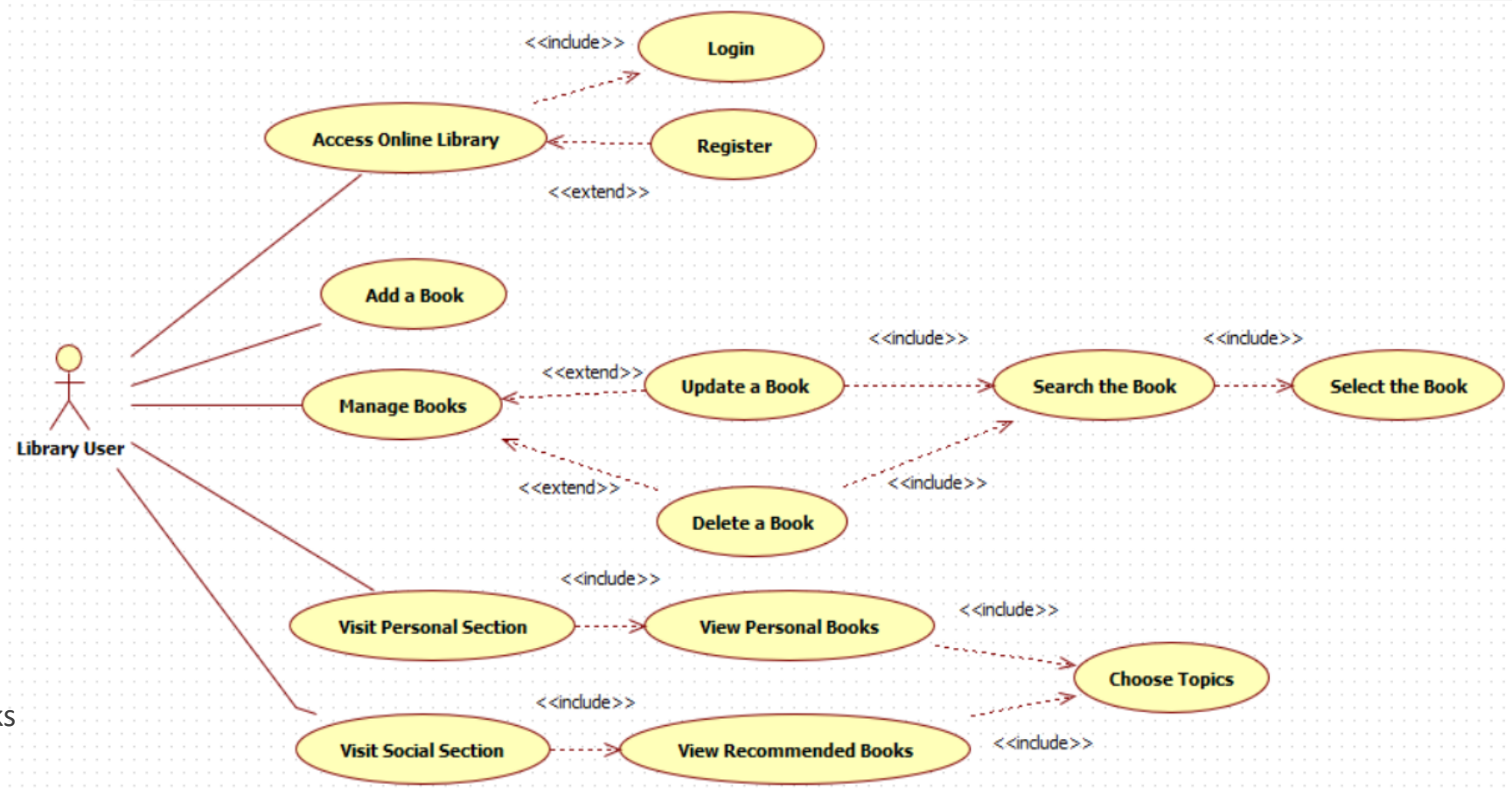
User can able to Select , Update, Delete and search the books from the library portal.

Maynooth University
National University of Ireland Maynooth

PART - II

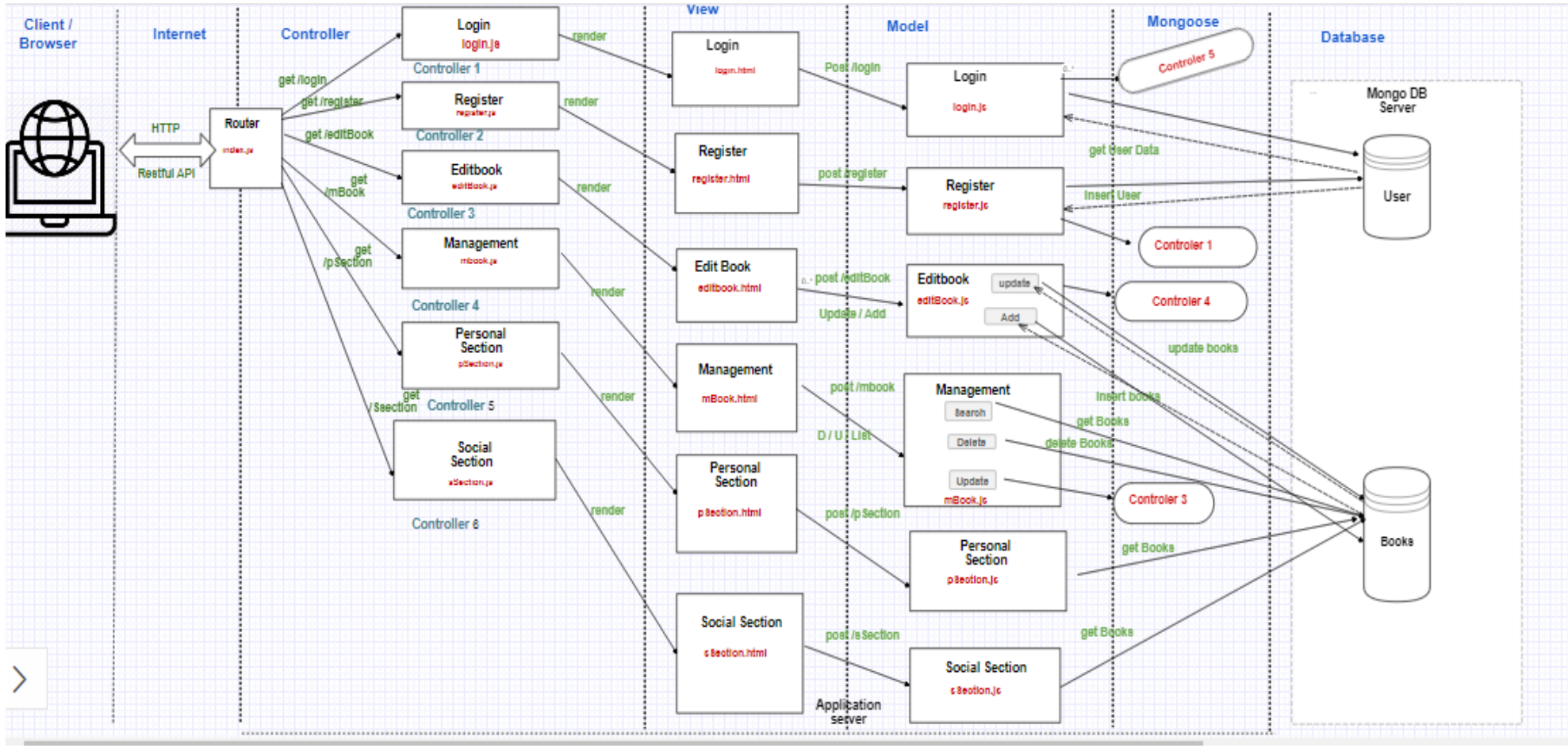
Use Case Diagram

- Access Online Library
  1. Register
  2. Login
- Add a Book
- Manage Books
  1. Search Book
  2. Select Book
     1. Update Book
     2. Delete Book
- Visit Personal Section
  1. Choose Metadata Tags
  2. View Personal Books
- Visit Social Section
  1. Choose Topics
  2. View Recommended Books



Web App Architecture

- Used MVC
  - To separate app into 3 components:
  - Logic, Interface, Database
- Model
  - Shows MongoDB data model of our library app
  - Constraints & formats to store data e.g JSON format
  - Represents collections e.g., user, book
  - Stores a book when user's request passes through controller to model
- View
  - Presents data to the users using .html files e.g., viewing personal or social library books
- Controller
  - Controls users' requests
  - Generates appropriate response
  - Gets triggered when user passes any request e.g request to view a book

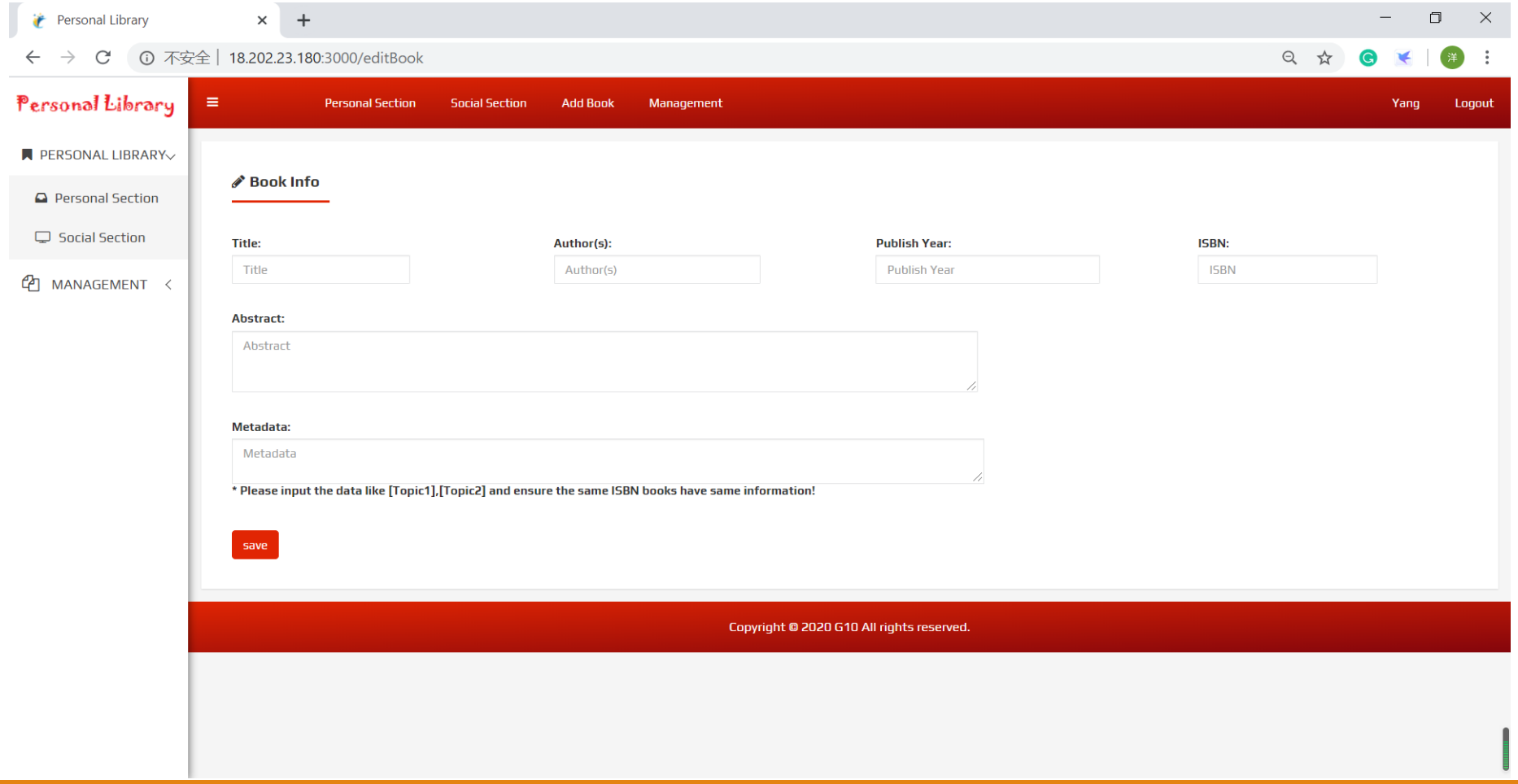


User Interaction – Add a Book

- Books can be added into personal library using this page
- ISBN Validation
  - Book not added in case of invalid ISBN
  - Accepts both 10 & 13 digits ISBNs
  - Used JavaScript regular expression to validate ISBN
  - Register
  - Examples of one 13 digit accepted ISBN:

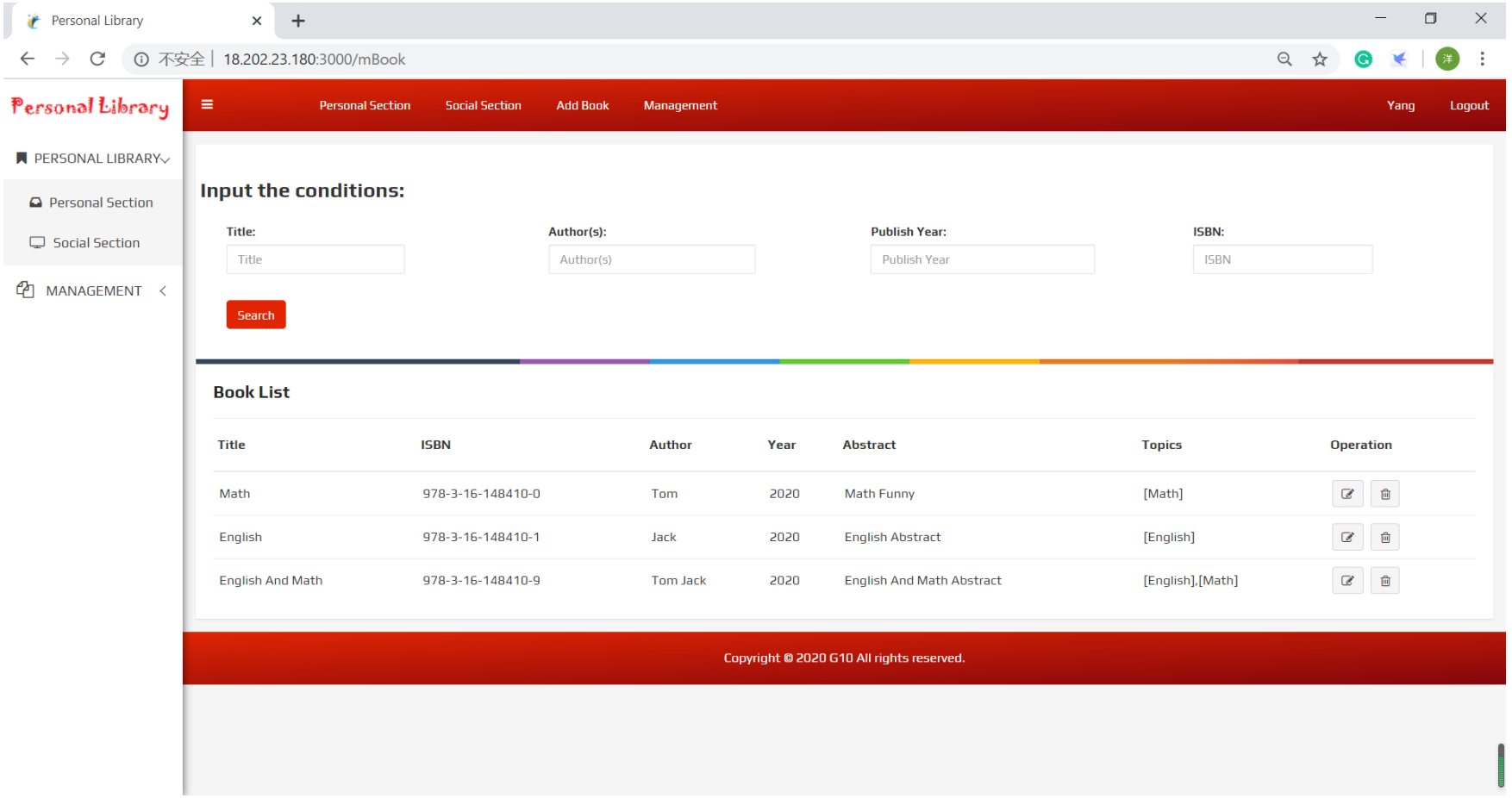
    978-1-4842-3896-7
  - Examples of one 10 digit accepted ISBN:

    0-13-187248-6
- Metadata
  - Can add more than one metadata tags / topics
  - Accepted Format : [Topic 1], [Topic 2]



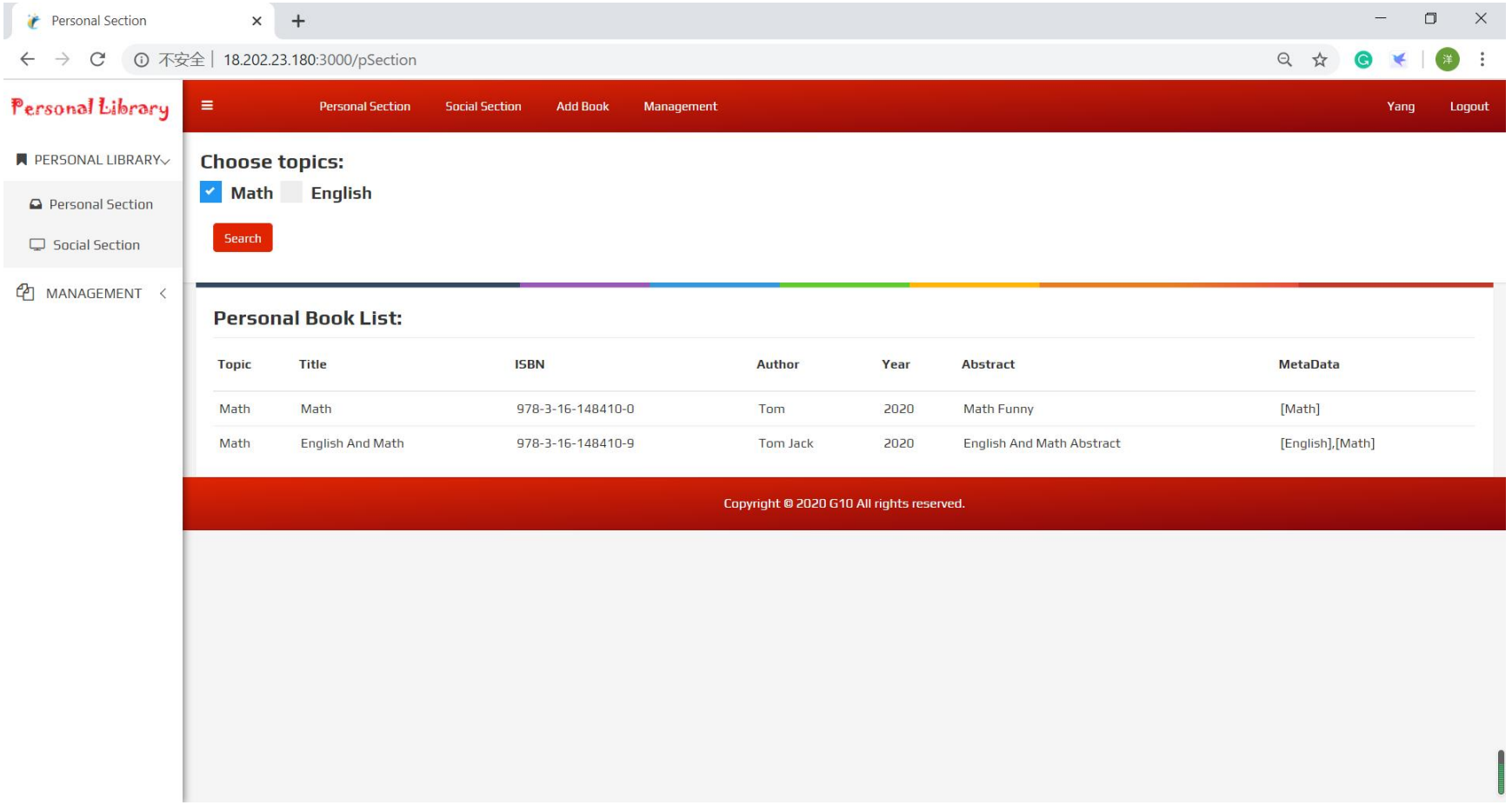
User Interaction – Book Management

- Different Operations can be performed on personal library
  - Search books
  - Edit Book Details
  - Delete Books



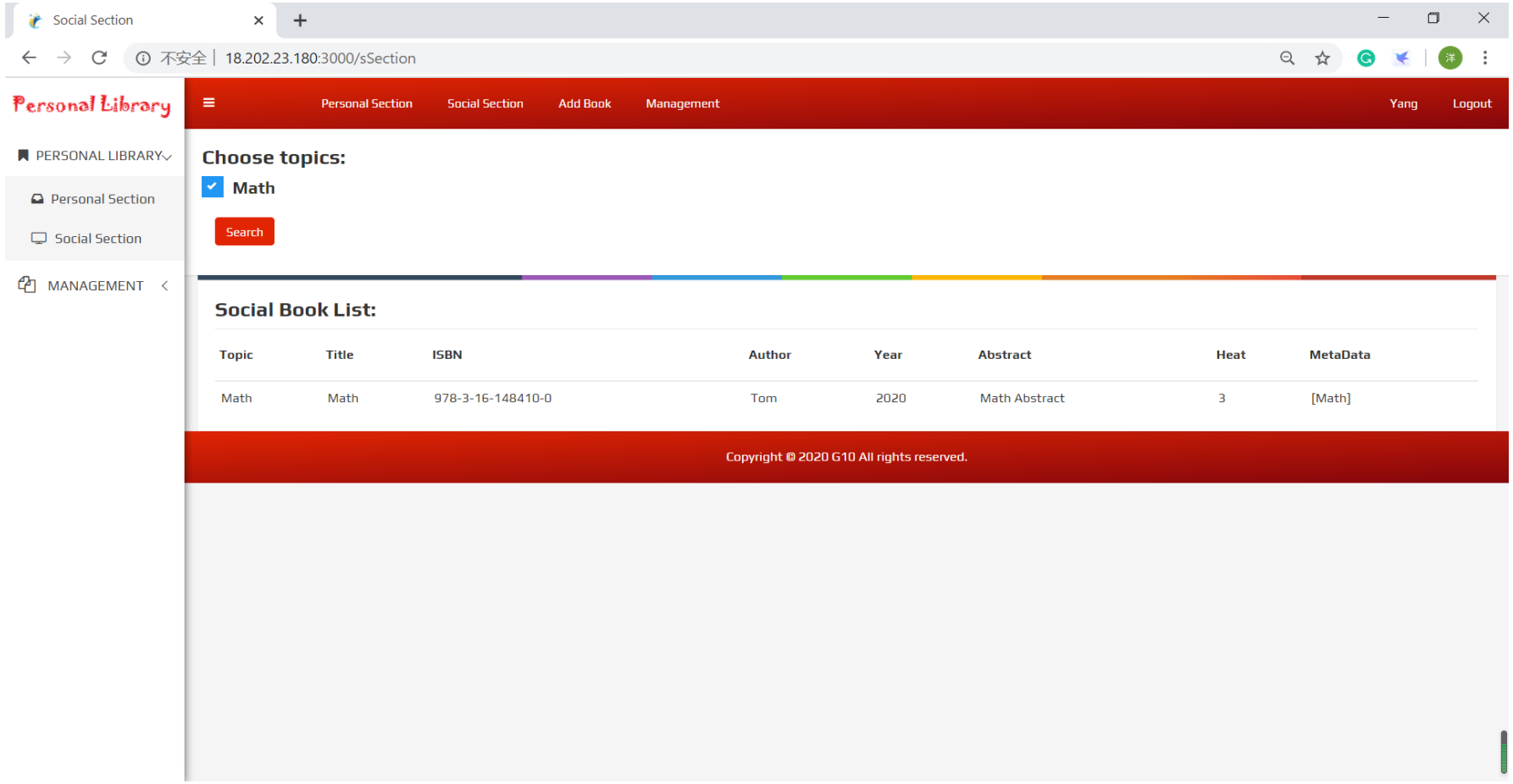
User Interaction – Personal Section

- Books added into personal library are shown here
- Topics / metadata tags can be selected for viewing books under a particular topic



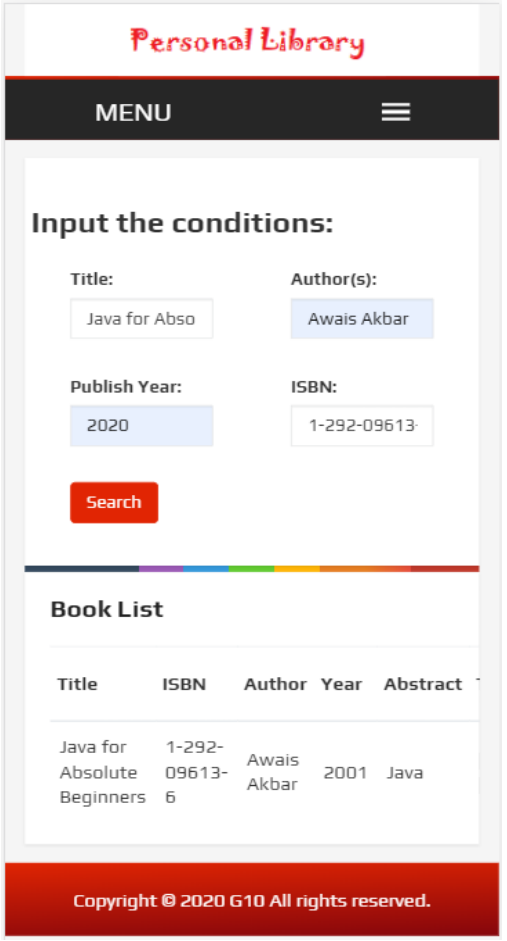
User Interaction – Social Section

- Books from other like-minded book lovers are shown here
- Books are recommended when user has three common books with another user
- Books from other user will be shown then
- Topics / metadata tag can be selected for viewing books under a particular topic



Mobile-Friendly (Responsive) Web Design

- Used CSS @media rule
  - Includes a CSS properties block when a particular condition is met
- Layout changes depending on the device
- Doesn’t require zooming to make the text readable
- Works well on mobiles and tabs



PART - III

Cloud Deployment Architecture

Used AWS to deploy the App and Db Servers

1. Available Zone:

   Used Two AZ to deploy the servers on two physical data centres.

   (Europe (Ireland) eu-west-1 and Europe (London) eu-west-2)

2. Private Subnet(EC2):

   Deployed different type servers on separate subnets and EC2.

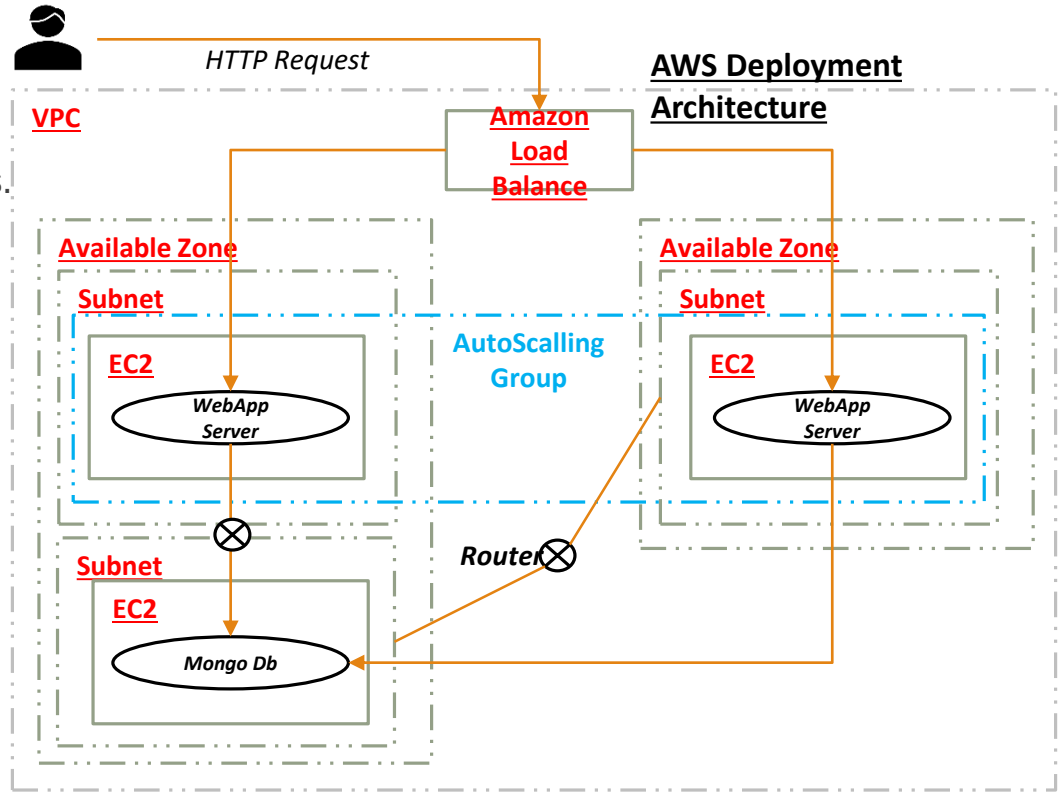
   Controlled with security groups from IPs, Protocols and ports.

3. Application Load Balance:

   Optimized the use of resources and avoid overload.

4. Auto Scaling:

   Improved fault tolerance, availability and reduce the cost.



Agile ideas for Project Management

Used some agile ideas to manage the project.

1. Requirement Analysis:

   Divided all functions into small modules to be suitable for Agile management.

   Every module was managed by separate JavaScript and HTML file.

   Used Restful API to access each module and plugged in or out freely.

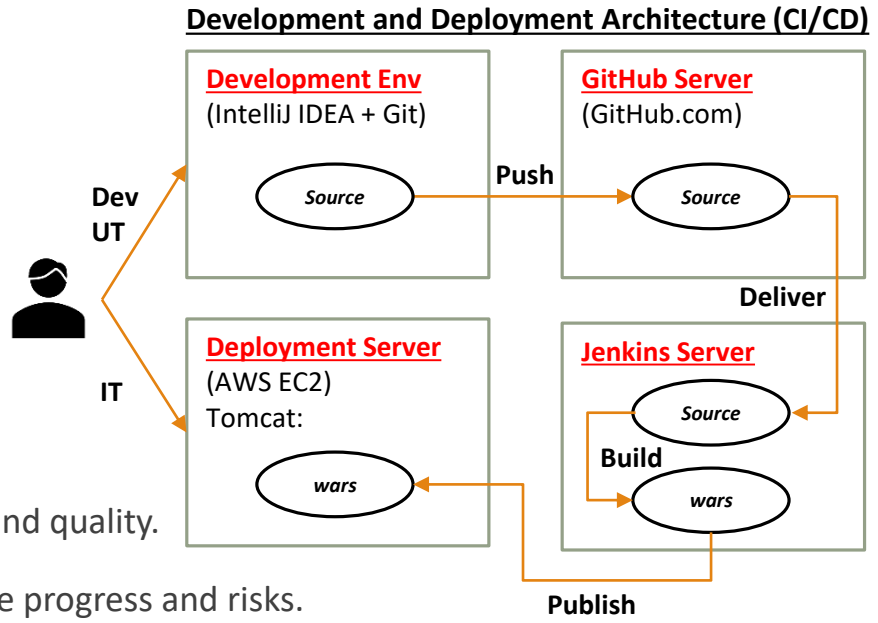
2. Team Management:

   Used the team structure (Scrum master + 2 members) to ensure the progress and quality.

   Operated the start-up, preview and review meetings every week to manage the progress and risks.

3. CI/CD - Continuous integration and continuous delivery:

   Used the Git and Jenkins to manage all project phases to deploy the services on servers automatically.



The blackmail trouble of MongoDB

1. Trouble:

   Several times we found that the data has been deleted from our database,

   and received a mail to let us pay Bitcoin to recovery the data.

2. Reason:

   To use the local tool to manage Database, We exposed the server on the public subnet.

   However, the default configuration of MongoDB does not have any security policy,

   the hackers can scan the database port(27017) to operate the database remotely.

3. Solution:

   Moved the MongoDB into private subnet, and closed the port(27017) to avoid the access from Internet.

   Meanwhile, added new policies to limited the unnecessary access.

Thanks for your attention

Maynooth University
National University of Ireland Maynooth