

IIHT

DIGITAL BOOKS

Table of Contents

[1. Problem Statement 3](#__RefHeading___Toc639_3426528727)

[2. Wireframes 4](#__RefHeading___Toc641_3426528727)

[3. Application architecture 5](#__RefHeading___Toc643_3426528727)

[Possible rest clients 5](#__RefHeading___Toc675_3426528727)

[4. Cloud Architecture 6](#__RefHeading___Toc645_3426528727)

[5. Tool Chain 7](#__RefHeading___Toc647_3426528727)

[6. Business Requirements 8](#__RefHeading___Toc649_3426528727)

[7. Proposed Rest Endpoints 9](#__RefHeading___Toc651_3426528727)

[8. Key Rubrics (Expected Deliverables) 10](#__RefHeading___Toc653_3426528727)

[A. As an application developer: 10](#__RefHeading___Toc655_3426528727)

[B. Debugging & Troubleshooting 10](#__RefHeading___Toc657_3426528727)

[C. Code Quality/Optimizations 10](#__RefHeading___Toc659_3426528727)

[9. Platform 11](#__RefHeading___Toc661_3426528727)

[10. Methodology: Agile 12](#__RefHeading___Toc663_3426528727)

# 1. Problem Statement

Build **D****igitalBooks** app which takes traditional books a step further, combining text with visual and audio elements to make authors' publications truly multimodal. Authors can write down their thoughts and assemble a collection of original or curated content ranging from photos, drawings, and images to audio and video clips -- in some cases, even animated text.

And while **D****igital Book** app can magically use images of autobiographical presentations or fantastical tales which are certainly options. It is also important to think beyond personal narratives to how authors might share the "stories/experiences" of their learning on any topic. And beyond author presentations and publications, plenty of students, teachers, doctors, engineers can jump on board, to create dynamic books and presentations that serve as instructional tools.

Build digital book app. Below are the different Microservices, which need to be developed and deployed.

1. User Microservice

Managed roles

* + 1. Guest user
    2. Reader
    3. Author

1. Book Microservice

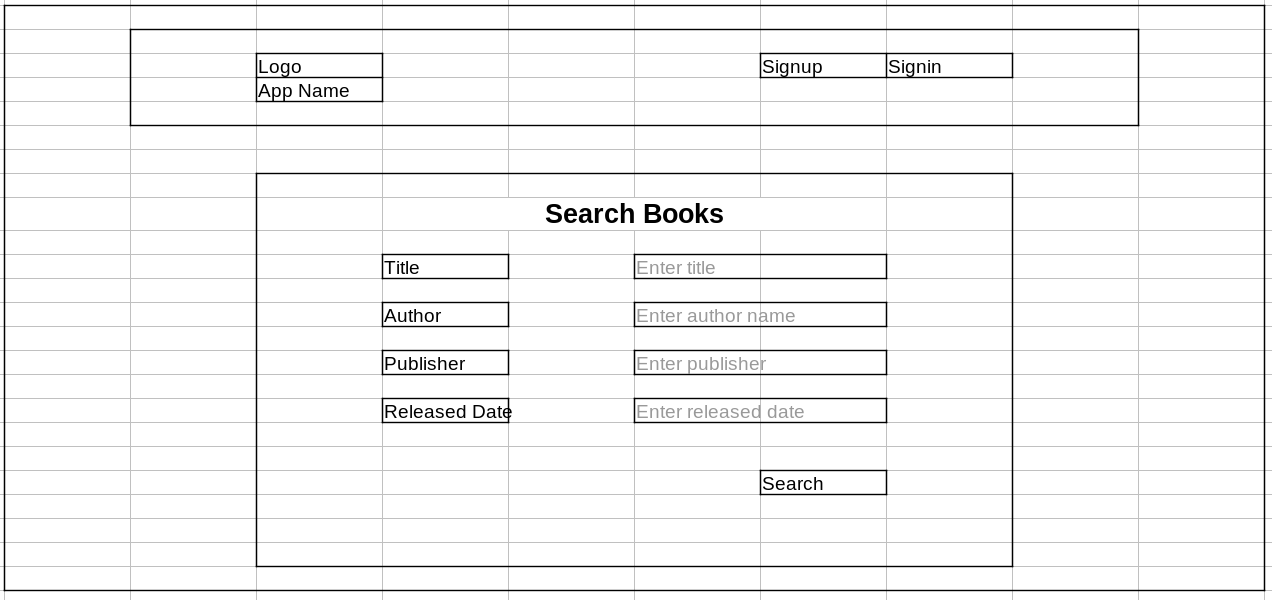
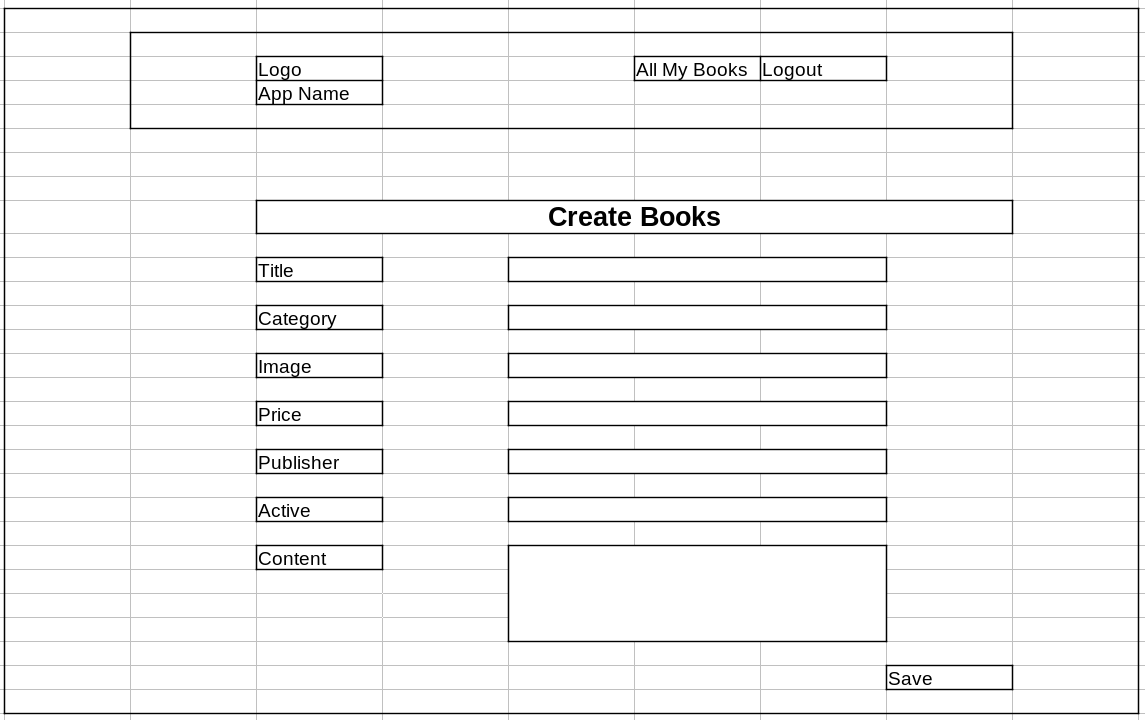
Crud application

* + 1. Guest user can search book, create account and get login
    2. Author can search/add/edit/block book
    3. Reader can search/subscribe/unsubscribe book

The scope includes developing the application using tool chain mentioned below.

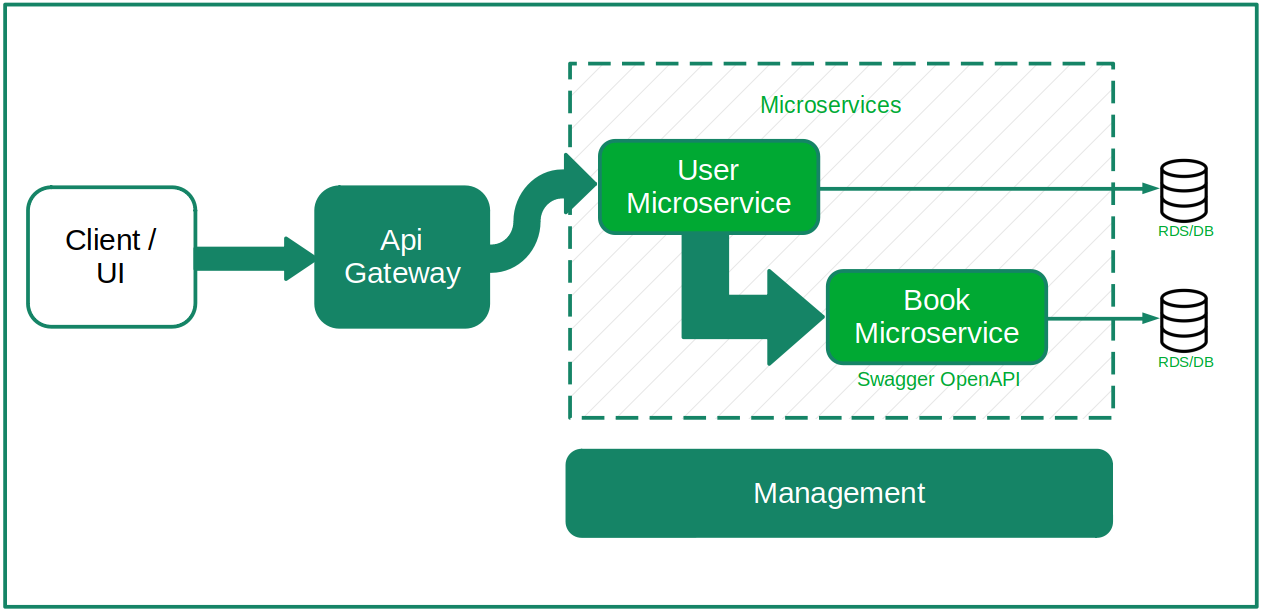
# 2. Wireframes

UI needs improvisation and modification as per given use case.



# 3. Application architecture

Compute and Integration/Presentation/Networking and Content Delivery



## Required services

* User microservice
* Book microservice

## Possible rest clients

We will use below **clients** for our microservice applications

* Postman
* Postwoman (hoppscotch.io)
* Swagger
* Rest Client - plugin
* Angular app
* Other microservice applications

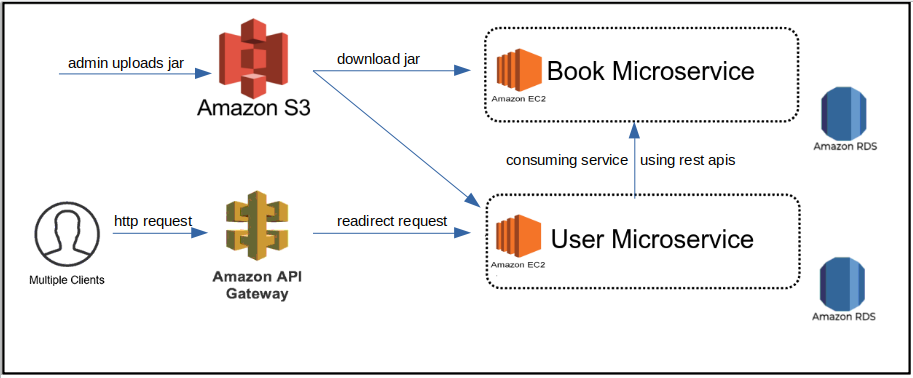
Any client from below list can consume our microservice (we will not use them):

* React app
* Jmeter
* Android app
* Ios app
* .Net application
* python application
* many more…

# 4. Cloud Architecture

S3

Api gateway → EC2 → RDS



# 5. Tool Chain

|  |  |  |
| --- | --- | --- |
| **Competency** | **Skill** | **Skill Detail** |
| Engineering Mindset | Networking and Content Delivery |  |
|  | DevOps |  |
| Programming Languages | Application Language | Java |
| Products & Frameworks | Presentation | HTML5, Bootstrap, CSS, Responsive web desiging |
|  |  | Angular, directives, components, services, routing and Material |
|  |  | NgRx and Route guard can be used.(Optional) |
|  |  | Javascript/Typescript |
|  | Compute & Integration | Spring Boot |
|  | Database & Storage | MySQL |
|  | Governance & Tooling | Git |
|  |  | Junit |
|  |  | Mockito |
| Engineering Quality |  |  |
| Platform | Cloud Tools | AWS EC2 |
|  |  | RDS-MySQL/Aurora |
|  |  | AWS API Gateway |
|  |  | AWS CloudWatch |

# **6. Business Requirements**

As an application developer, develop microservices with below guidelines:

|  |  |  |
| --- | --- | --- |
| User  Story # | User Story Name | User Story |
| US\_01 | Guest User Features | * As a guest user I want to search for books based on **title, category, author, price**,so I can see search result containing book logo, title, author, publisher, price, published date, category. * As a guest user, I want to create an account. I can create account as either **reader** or **author** * As a guest user, I want to get login to my account. |
| US\_02 | Reader Features | * As a reader, I want to search for books based on **title, category, author, price**,so I can see search result containing book logo, title, author, publisher, price, published date, category. * As a reader, (from the above search result) I want to select a book (which is not blocked) and subscribe to it. I want to get a unique **subscription\_id** in response after subscription is done. * As a reader, I can read only subscribed books. * As a reader, I can not read blocked book or book which is not subscribed. * As a reader, I want to   + view history of all previous subscriptions   + view invoice using subscription\_id   + cancel the subscription within 24 hrs |
| US\_02 | Author Features | 1. As an author, I want to create/edit book so that reader can subscribe to it. 2. As an author, I want to create book with below properties    * **logo**: image    * **title**: Spiderman is back    * **category**: comic    * **price**: 24    * **author**: {current user name}    * **publisher**: Moon publisher    * **published date**: 22/04/2022    * **chapters/content**    * **active:** true 3. As an author, I want to block and unblock a book. 4. When book is blocked    * It will not be shown in Search results for the reader.    * Readers (who have already subscribed to this book) should get notification about the unavailability of book. |

# **7. Proposed Rest Endpoints**

If you think rest endpoints need improvisation and modification as per given use case, you can make necessary changes.

|  |  |  |
| --- | --- | --- |
| **GET** | /api/v1/digitalbooks/search?category&title&author&price&publisher | Guest, Reader and Author can search books |
| **POST** | /api/v1/digitalbooks/sign-up | Guest can create account as either reader or author |
| **POST** | /api/v1/digitalbooks/sign-in | Guest can get login |
| **POST** | /api/v1/digitalbooks/{book-id}/subscribe  payload: {bookId, reader {email / pk}} | Reader can subscribe to a book |
| **GET** | /api/v1/digitalbooks/readers/{emailId}/books | Reader can fetch all subscribed books |
| **GET** | /api/v1/digitalbooks/readers/{emailId}/books/{subscription-id} | Reader can fetch a subscribe book |
| **GET** | /api/v1/digitalbooks/readers/{emailId}/books/{subscription-id}/read | Reader can read book content |
| **POST** | /api/v1/digitalbooks/readers/{emailId}/books/{subscription-id}/cancel-subscription | Reader can cancel the subscription within 24 hrs of subscription |
| **POST** | /api/v1/digitalbooks/author/{author-id}/books  Payload: {logo, title, category, price, author, publisher, published date, chapters/content, active} | Author creates a book |
| **PUT** | /api/v1/digitalbooks/author/{author-id}/books/{book-id}  Payload: {logo, title, category, price, author, publisher, published date, chapters/content, active} | Author can edit a book |
| **POST** | /api/v1/digitalbooks/author/{author-id}/books/{book-id}?block=yes | Author can block a book |
| **POST** | /api/v1/digitalbooks/author/{author-id}/books/{book-id}?block=no | Author can unblock a book |

# **8. Key Rubrics (Expected Deliverables)**

## A. As an application developer:

* + 1. Develop the application as a microservice architecture.
    2. Ensure package Structure for project is like **com.****digitalbooks.\*** with proper naming conventions for package and beans.
    3. Use **application.properties** or **application.yaml** file to maintain all spring boot config.
    4. Implemented the package structure - Controller, Interface, Service, DAO, Testing, Validation, Security etc
    5. Implementation as follows:
       1. Use Domain Driven Design to implement distributed architecture
       2. Follow Single Data Store per microservice practice
       3. Document REST endpoints with OpenAPI/ Swagger
       4. Expose all rest Endpoints using aws API Gateway

## B. Debugging & Troubleshooting

* + 1. Generate bug report & error logs - Report must be linked with final deliverables which should also suggest the resolution for the encountered bugs and errors.

## C. Code Quality/Optimizations

* + 1. Associates should write clean code that is readable
    2. Associate should use the Code Analyzer (PMD/SonarQube) to ensure code quality and standard code style.

# 9. Platform

* + All jar files must be uploaded into S3.
  + All jar files must be executed in EC2 instance.
  + All APIs from all jar files must connect with RDS MySQL
  + All APIs from all jar files must be configured in API gateway and deployed in stages
  + All deployed APIs in API gateway must be consumed from Frontend

# 10. Methodology: Agile

* + As an application developer, use project management tool along to update progress as you start implementing solution.
  + As an application developer specify the estimation and planning as a part of Agile process.
  + As an application developer, the scope of discussion with mentor is limited to:
    1. Q/A
    2. New Ideas, New feature implementations and estimation.
    3. Any development related challenges
    4. Skill Gaps
    5. Any other pointers key to UI/UX and Middleware Development