AUGEN
SOFTWARE VISION

Technical Test

VERSION 1.1

FOR

Frontend Angular

Authors and contributors: Winston Chow

Date: 2021

1 Overview

This exercise has been designed to test your front and backend development skills. The results will help us evaluate your fit for the development role at Augen Software Group.

1.1 Timeframes

We expect this test to take anywhere from **8-12 hrs** depending on your skill level. As part of the exercise, you will need to give a summary breakdown of the time you have spent and any design decisions made.

1.2 Your work

We hope that you will do this test to the best of your ability without copying or referring to outside assistance. If you are unsure about task, feel free to email\skype the operations manager (operations.manager@augensoftwaregroup.com)

Alternatively, you can make assumptions if your implementation is slightly different from the functionality requested. Please ensure those assumptions are stated in your output.

2 Assignment

2.1 Background

A bookstore wants to grow their market size and decides to use ecommerce as a means, they want to build a website for their customers to browse a variety of books, buy it online and ship it via different delivery services. The store coordinates with delivery vendors of motorbike, train and aircraft for their service. The delivery cost varies based on the delivery service and season, i.e. it is cheaper during summer due to low demand however more expensive during school time due to high demand. Following is its calculation:

Delivery cost = base cost * factor

Service	Base cost	Factor		
		June to Aug	Sep	Other months
Motorbike	$5	0.5	1.5	1
Train	$10	0.8	1.8	1
Aircraft	$20	0.8	2	1

As customer confirms to buy, the store sends an email to the customer briefing the delivery info so that the customer is aware of who is delivering, when and where the package arrives. Each delivery service has its own info:

Service	Delivery info
Motorbike	Driver name: {driver name} Mobile: {phone number} Delivery date: {dd/MM/yyyy} Cost: {delivery cost}
Train	Train no: {train no} Station of arrival: {station name} Date of arrival: {dd/MM/yyyy} Cost: {delivery cost}
Aircraft	Flight no: {flight no} Gate of arrival: {gate number} Date of arrival: {dd/MM/yyyy} Cost: {delivery cost}

2.2 Your assignment:

As a developer, you will be responsible for building a **SPA** for that bookstore so that:

- User is able to browse a variety of books
- User is able to search by book title or author
- User is able to view details of a book
- User is able to buy it. Upon confirmation, your app needs to generate the delivery info for the selected delivery service

Back end requirements:

-
- Framework: Java (spring framework and spring boot), or .Net Core, or NodeJS or PHP
- Use a build tool like Maven or Gradle if implementing with java.
- Write unit tests
- Design your code so that the book enquiry can be easily substituted by other source of data e.g. 3rd Api, CSV, JSON …
- For the sake of this test, use Google book API for book enquiry

https://www.googleapis.com/books/v1/volumes?q={search text}


```
{
 "kind": "books#volume",
 "id": "P2GS1m876L4C",
 "etag": "LEXOZCXk4bA",
 "selfLink": "https://www.googleapis.com/books/v1/volumes/P2GS1m876L4C",
 "volumeInfo": {
  "title": "Flower Confidential",
  "subtitle": "The Good, the Bad, and the Beautiful",
  "authors": [
   "Amy Stewart"
  ],
  "publisher": "Algonquin Books",
  "publishedDate": "2008-03-18",
  "description": "Award-winning author Amy Stewart takes readers on an around-the-world, behind-the-scenes look at the
orse—to achieve perfection. She tracks down the hybridizers, geneticists, farmers, and florists working to invent, manu
nd sturdier than anything nature can provide. There's a scientist intent on developing the first genetically modified bl
reated the most popular lily; a breeder of gerberas of every color imaginable; and an Ecuadorean farmer growing exquisit
nd, at every turn she discovers the startling intersection of nature and technology, of sentiment and commerce.",
 "imageLinks": {
  "smallThumbnail": "http://books.google.com/books/content?id=P2GS1m876L4C&printsec=frontcover&img=1&zoom=5&edge=curl&source=gbs_api",
  "thumbnail": "http://books.google.com/books/content?id=P2GS1m876L4C&printsec=frontcover&img=1&zoom=1&edge=curl&source=gbs_api"
 },
```


Front end requirements:

- JS Framework: Angular 8+
- CSS framework: your choice, however it must be CSS preprocessor (SASS, SCSS…)
- Application of a state management library would be a plus
- Application of ES6 or Typescript would be a plus

Deliver your test:

- Upload your code to your Github
- In README.md provide:
    - How to build & run your server app and SPA on a local machine.
    - Reasons your application of design / patterns, i.e. why did you use it, what benefit it brings to your code, how it makes your code loosely coupling, how it makes your code comply to SOLID…
    - Provide a summary breakdown of the approximate time that you have spent developing the solution e.g.

      Example: Time summary:

        - Preparation: 1 hour
        - Coding: 5 hours (including tests)
        - Styling 30 minutes
        - Building and testing 30 minutes

      Grand total: 7 hours

Bonus Requirement (This is optional. Complete if you have available time)

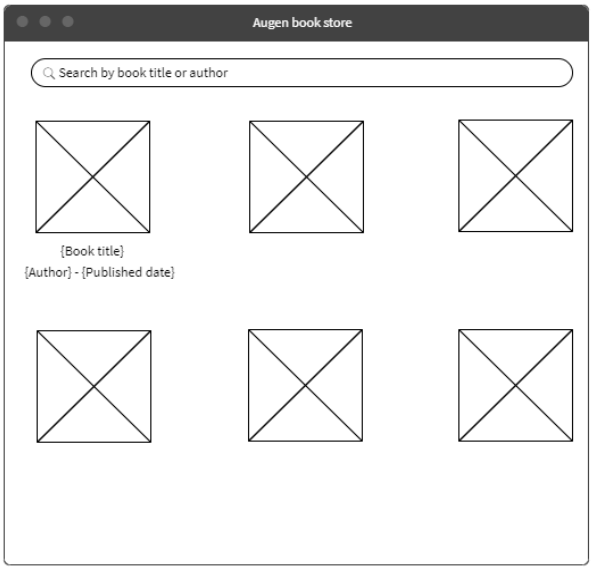
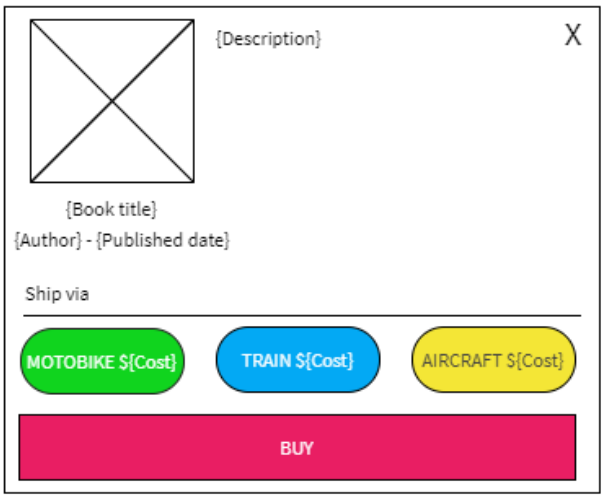
- As user confirms to buy, your API needs to generate the delivery info for the selected delivery service as described above. You need to design a **DeliveryInfoGenerator**, which is capable of generating a string representing the delivery info of the selected service. You can use random values for the corresponding arguments for the sake of this test (i.e. use random driver name, mobile phone and delivery date for motorbike). Your endpoint may resemble below:

```
public class BuyBookModel
{
    public string DeliveryService { get; set; }
    public double DeliveryCost { get; set; }
}

[HttpPost]
public string BuyBook([FromBody] BuyBookModel model)
{
    // random driver name, mobile phone and delivery date if selected service is motorbike. Same as for other services
    string driveName = ...;
    string mobilephone = ...;
    DateTime deliveryDate = ...;

    // use DeliveryInfoGenerator to generate the delivery info
    var info = _deliveryInfoGenerator...
    return info;
}
```

Additional Info

- No database / data persistence required
- Application of design / patterns is **critical** to this test

Mockups:

Screen	Description
Landing page 	- Upon searching, display books as thumbnails alongside its title, author and published date. - Each line accommodates as many items as its width can - Add paging (this is not shown in the mockup) or alternative ways to manage the performance on the initial load. - Clicking on item results in a details popup
Details popup 	- Display book info as mockup - Display cost of each delivery service - Icon 'X': close popup - Button 'BUY': close popup and show a confirmation. Call Buy endpoint if completing Bonus Requirement.

How your test is evaluated:

We evaluate your test against the following points:

- Your ability to spot problems in code and apply design / patterns to sort them out
- Your reasoning over your application of design / patterns
- Your written instructions and notes
- Minimum coding requirements evaluated:
    - Unit tests and integration tests.
    - Error design and management.
    - Import optimisation and indenting.
    - Input data validation.
    - No hardcoded data.
    - Consistency in naming (methods and classes)
    - Commentinga
-